

**ONTOLOGY** SEMANTIC WEB  
REASONER INFERENCE  
**OBDA** DECISION MAKING SUPPORT  
CONCEPTUAL MODELING

**SPARQL**  
ENTERPRISE MODELING  
**OWL**  
KNOWLEDGE

**TRIPLE STORE GRAPH**

**ONTOLOGY** SEMANTIC WEB  
REASONER INFERENCE  
**OBDA** DECISION MAKING SUPPORT  
CONCEPTUAL MODELING

**SPARQL**  
ENTERPRISE MODELING  
**OWL**  
KNOWLEDGE

**TRIPLE STORE GRAPH**

онтологическое моделирование предприятий: методы и технологии

тринити<sup>△</sup>ата

онтологическое  
моделирование  
предприятий:

методы и технологии



# **ОНТОЛОГИЧЕСКОЕ МОДЕЛИРОВАНИЕ ПРЕДПРИЯТИЙ: МЕТОДЫ И ТЕХНОЛОГИИ**

**Монография**

Издательство Уральского университета  
Екатеринбург  
2019

УДК 005.11  
ББК Ув631  
О595

Авторский коллектив:  
С. В. Горшков (предисл., гл. 5, 6); С. С. Кралин, О. И. Муштак (гл. 1);  
С. З. Гумеров (гл. 2); М. Г. Мирошниченко (гл. 3);  
А. Ю. Гребешков (гл. 4); Р. Ю. Шебалов (приложение)

Ответственный редактор  
С. В. Горшков, директор ООО «Тринидата»

Рецензенты:

Кафедра онтологии и теории познания Уральского гуманитарного института  
Уральского федерального университета (заведующий кафедрой  
кандидат философских наук А. Г. Кислов);  
Н. Н. Непейвода, доктор физико-математических наук  
(Институт программных систем им. А. К. Айламазяна РАН)

О595 **Онтологическое моделирование предприятий: методы и технологии :**  
монография ; [отв. ред. С. В. Горшков] ; предисл. С. В. Горшкова.–  
Екатеринбург : Изд-во Урал. ун-та, 2019.– 236 с.

ISBN 978-5-7996-2580-1

Монография посвящена современным практикам применения онтологического моделирования при создании корпоративных автоматизированных систем. Рассматривается широкий круг методологических и технологических вопросов моделирования, набор современных программных средств и стандартов представления и обработки онтологических моделей. Большое внимание уделено прикладному применению этих инструментов для решения актуальных задач в области экономики и безопасности.

Работа адресована руководителям бизнес-подразделений, аналитикам и ИТ-специалистам, ответственным за цифровую трансформацию предприятий.

УДК 005.11  
ББК Ув631

ISBN 978-5-7996-2580-1

© ООО «Тринидата», 2019

## Оглавление

Предисловие .....	5
<b>Глава 1. Методологические и теоретические основания онтологического моделирования .....</b>	<b>8</b>
§ 1. Гносеологическая проблематика.....	8
§ 2. Проблемы философии языка .....	12
§ 3. Логические основания онтологий .....	16
§ 4. Semantic Web и Linked Data .....	22
Список библиографических ссылок .....	32
Дополнительная литература.....	33
<b>Глава 2. Управление результатами. Моделе-ориентированная система управления .....</b>	<b>34</b>
§ 1. Цифровизация и проблемы экономики .....	34
§ 2. Цели, задачи и структура моделе-ориентированной системы управления .....	44
§ 3. Программа цифровой трансформации .....	46
§ 4. Структура моделе-ориентированной системы управления.....	49
§ 5. Интегрированная модель подконтрольной системы .....	54
§ 6. Последовательность разворачивания модели.....	57
Список библиографических ссылок .....	63
<b>Глава 3. Практические вопросы концептуального моделирования предметных областей .....</b>	<b>66</b>
§ 1. Что такое модель и моделирование? .....	66
§ 2. Представления о пространстве и времени.....	70
§ 3. Моделирование свойств пространственно-временных объемов....	73
§ 4. Проблемы концептуального описания предметных областей ..	79
Список библиографических ссылок .....	87
<b>Глава 4. Классификация в онтологическом моделировании.....</b>	<b>88</b>
§ 1. Задача классификации и построение классификаторов.....	88
§ 2. Методы классификации объектов.....	104
§ 3. Атрибуты и идентификаторы классифицируемых объектов ..	116
Список библиографических ссылок .....	122
<b>Глава 5. Технологии и методика онтологического моделирования ..</b>	<b>124</b>
§ 1. Практические задачи предприятий и методы их решения .....	124
§ 2. Моделирование объектов.....	127
§ 3. Основные возможности стандартов моделирования RDF/RDFS/OWL.....	130

§ 4. Моделирование свойств и отношений объектов.	
Моделирование физических объектов	
и технической инфраструктуры.....	137
§ 5. Тип или свойство объекта? .....	144
§ 6. Отражение в модели точек зрения разных субъектов .....	146
§ 7. Моделирование активностей. Отражение в модели бизнес-процессов, их участников и артефактов .....	147
§ 8. Утверждения о множествах объектов. Класс или индивидуальный объект? .....	155
§ 9. Машины логического вывода и другие способы работы с моделью .....	159
Список библиографических ссылок .....	163
<b>Глава 6. Применение онтологий в автоматизированных системах ...</b>	164
Программные средства работы с онтологиями .....	164
Сценарий 1: Обеспечение доступности знаний в организации.....	168
Сценарий 2: Системы консолидации и анализа данных .....	174
Сценарий 3: Система поддержки принятия решений.....	179
Сценарий 4: Разработка программного обеспечения, управляемого онтологией .....	183
Список библиографических ссылок .....	188
<b>Приложение. Программный инструментарий онтологического моделирования. Практикум.....</b>	190
Редакторы онтологий и их возможности .....	190
Моделирование онтологий ТЭС средствами редактора onto.pro....	196
Моделирование объектов и свойств .....	202
Моделирование логических утверждений в редакторе Protégé .....	213
Работа с правилами логического вывода в среде АрхиГраф.СУЗ....	226
Список библиографических ссылок .....	232
<b>Сведения об авторах.....</b>	234

## **Предисловие**

Стремительный рост интереса к цифровым программам трансформации экономики и принятие соответствующих государственных программ ставят вопрос о выборе методов и инструментов их реализации. Критический недостаток исследований этих вопросов на русском языке, превалирование коммерческой и технологической информации над научной и методической приводят к тому, что специалистам-практикам трудно составить целостные и обоснованные представления о доступных средствах цифровизации, провести их отбор и организовать направленную деятельность по разработке и реализации программы цифровой трансформации конкретного предприятия, города, территории.

Серьезной проблемой является разрыв между академическими исследованиями в области онтологического моделирования и практикой, характеризующейся, в частности, такими проявлениями:

– Огромное количество публикаций в международных рецензируемых журналах, книг, тезисов научных конференций остаются практически не известными российским читателям; относительно небольшое число российских исследователей представляют результаты своих работ на признанных академических площадках.

– Еще меньше доля коллективов, применяющих онтологическое моделирование в практической работе по созданию автоматизированных систем и одновременно участвующих в научной работе. Это приводит к преобладанию в публичном поле маркетинговой, субъективной и не верифицированной академическим сообществом информации, повышает риск неудач прикладных проектов. Ощущается недостаток площадок для широкой экспертизы сообществом представляемых на рынке решений, методов и продуктов, обсуждения результатов их применения по принципам научной дискуссии.

– Многие архитекторы прикладных автоматизированных систем не воспринимают разработки исследовательских коллективов как источник инноваций, предпочитают «проверенные» средства «новым», «рискованным».

На последний аргумент можно возразить тем, что, во-первых, при помощи стандартных средств могут быть решены только стандартные задачи, во-вторых, онтологическое моделирование давно уже прошло стадию «новизны» и на сегодняшний день представляет собой зрелый и успешно применяемый как в мировой, так и в российской практике набор технологий и методов. В обзоре «Современные российские разработки в области онтологического моделирования»<sup>1</sup> перечислены российские коллективы, работающие в этой области по состоянию на 2018 год.

Данные обзора, к счастью, подтверждают существование активных групп специалистов, серьезно занимающихся теоретическим исследованием и практическим применением онтологического моделирования, публикующих результаты своих исследований и демонстрирующих историю успеха в различных областях экономики и управления. Их примеры доказывают, что заказчики, верно оценившие потенциал и сильные стороны онтологий, строят автоматизированные системы, обладающие принципиально новыми возможностями по сравнению с «традиционными», ориентированными на технологическую обработку не концептуализированной информации.

Задачей данной монографии является разностороннее рассмотрение комплекса вопросов, связанных с построением и использованием цифровых моделей предприятий и территорий, активов и процессов. Общей идеей, объединяющей все главы, является необходимость согласованного подхода к выбору и использованию методик моделирования, воплощения созданных моделей в автоматизированных системах и применения полученных инструментов для решения экономических и управлеченческих задач. Всех авторов объединяет также приверженность онтологическому моделированию как одному из наиболее мощных и перспективных инструментов для создания моделей сложных систем.

---

<sup>1</sup> <https://trinidata.ru/files/OntoReview.pdf>

В первой главе рассматриваются философские основания построения онтологических моделей. Вторая глава посвящена месту моделей в процессах управления, предлагает новый и смелый взгляд на средства перехода от «интуитивного» к рациональному принятию решений. Третья глава посвящена рассмотрению некоторых проблем, с которыми сталкивается аналитик, начинающий осваивать методы концептуального моделирования. В четвертой главе рассмотрены методы структурирования информации, применяющиеся при составлении классификаторов и справочников, являющихся стержнем любой модели. В пятой главе описываются функциональные возможности и технологические основы стандарта онтологического моделирования OWL, а также затрагивается широкий круг методических вопросов, возникающих при построении таких моделей. Шестая глава посвящена рассмотрению сценариев использования онтологических моделей в автоматизированных системах. Наконец, приложение освещает практические вопросы создания онтологических моделей при помощи доступных сегодня программных продуктов, включая российское программное обеспечение.

Авторы всех глав монографии – специалисты-практики, имеющие обширный и успешный опыт создания и реального применения моделей для решения экономических и управленческих задач в крупнейших корпорациях России, на уровне муниципального и государственного управления.

*Сергей Горшков,  
директор ООО «Тринидата»*

## Глава 1

# МЕТОДОЛОГИЧЕСКИЕ И ТЕОРЕТИЧЕСКИЕ ОСНОВАНИЯ ОНТОЛОГИЧЕСКОГО МОДЕЛИРОВАНИЯ

*С. С. Кралин, О. И. Муштак*

Наконец, до полного позора дело дошло тогда, когда словом «концепт» завладели информатика, маркетинг, дизайн, реклама.  
*Ж. Делёз. Что такое философия?*

Тим Бернерс-Ли говорит в одном интервью<sup>1</sup>:

So, I'd for you to have enough of a body of understanding, so when people in a Working Group stop and say «Wait, this doesn't match what I learned from Wittgenstein» that you can say «No, please go read this pamphlet, it's about philosophical engineering and it explains the philosophy of what your doing, so you won't find Wittgenstein very useful in this case or these are the bits that you will find useful.» (Я бы хотел, чтобы у вас было достаточно понимания, поэтому, когда люди в рабочей группе [рабочая группа W3C. – С. Г.] останавливаются и говорят: «Подождите, это не соответствует тому, что я читал у Витгенштейна», вы можете сказать: «Нет, пожалуйста, прочитайте эту брошюру; она посвящена философской инженерии и объясняет философию того, что вы делаете, поэтому Витгенштейн не будет очень полезным в этом случае, или вам пригодятся лишь небольшие фрагменты» [перевод наш – С. Г.].)

Эта глава не претендует быть такой «брошурой», она, скорее, некое предисловие. Вопросы, обозначенные в названии главы, будут рассмотрены с довольно традиционных позиций, включающих уместные элементарные сведения из курсов, читаемых кафедрой онтологии и теории познания одного классического университета.

Читатель будет ознакомлен с интеллектуальным инструментарием, иногда употребляемым, как оказывается, в рабочих группах W3C, и некоторыми первоначальными последствиями его применения, релевантными тематике издания.

---

<sup>1</sup> URL: <https://www.w3.org/DesignIssues/PhilosophicalEngineering.html>.

## § 1. Гносеологическая проблематика

Традиционно гносеология фокусировалась на проблемах истинности и источника знания. Проблематика истинности обсуждалась в связи с противопоставлениями знания и мнения, знания и веры. Проблематика источника находила выражение в оппозиции рационализма и эмпиризма, хотя существуют и более современные «-измы».

К формальному аппарату онтологического моделирования эти проблемы имеют мало отношения. Соответствующий аппарат не имеет встроенных средств экспликации истинности: в нем нет ни эпистемических модальностей, ни немонотонности. Что до проблематики источника, рассматриваемый аппарат строго дедуктивен.

Возможно, во всей полноте уже эпистемологических конкретизаций эти проблемы могли бы иметь отношение к применению аппарата, будь эти последние рассмотрены как своего рода «*a new kind of science*» («*science*», быть может, даже в менее утрированном смысле, чем в «*data science*»), однако применением аппарата посвящены другие главы.

Более современная гносеология уделяет больше внимания проблемам осмыслинности и смысла, и эта проблематика кажется здесь более релевантной. Цель и ценность онтологического моделирования и предоставившего ему технологическую основу проекта Semantic Web как раз и заключается в том, чтобы сделать данные и информацию знаниями.

Наиболее же релевантной в данном контексте является проблематика, относящаяся скорее к некоей прикладной гносеологии – проблематика **представления знаний**.

Знание само по себе – своего рода представление. Зачем может понадобиться его еще раз как-то представлять? Уместнее, на наш взгляд, здесь говорить об обратных процессах: опредмечивании и отчуждении знания. Цели опредмечивания и отчуждения традиционны и, возможно, неблаговидны, а средством их достижения является формализация.

Формализация заключается в моделировании знания в более простой знаковой системе, чем естественный язык, и последующих попытках получить дополнительные выгоды (или обеспечить отсутствие убытков) при помощи различных средств ее обработки. С этой точки зрения стоит говорить о двух стадиях формализации: символизации и аксиоматизации. Специфика символизации, предлагаемой проектом Semantic Web, до некоторой степени посвящен параграф «Проблемы философии языка», аксиоматизации – параграф «Логические основания онтологии».

На самом деле, не вполне понятно, какова архитектороника человеческого знания, из чего знание «состоит» и как «работает». То, каковы должны быть «примитивы» знания и их различные виды, и является центральным вопросом дисциплины «Представление знаний». Ответ на этот вопрос определяет основные характеристики формализма представления знаний:

- выразительность. Здесь определенным мерилом выступает логика предикатов первого порядка; выразительность принятого формализма представления знаний может быть как меньше, так и больше;

- вычислительная эффективность, находящаяся в обратной зависимости от выразительности (уже логика предикатов первого порядка неразрешима);

- как частный случай выразительности – способность к метаписанию и рефлексии;

- отношение к проблеме неполноты знания либо, наоборот, преыщенности истинностных оценок.

Каковы эти характеристики у формализмов представления знаний RDF и OWL, принятых в Semantic Web, будет показано в заключительном параграфе данной главы.

**Исторически** наиболее известными подходами к представлению знаний являлись фреймы и семантические сети.

Автор концепции фреймов – Марвин Минский (известный как соавтор «Перцептронов» [1], приведших в 1970-х к долговременной потере интереса к коннекционистским подходам к искусственному интеллекту).

Позднее Джон Маккарти (создатель Лиспа) и Патрик Хейес, который сейчас наиболее известен как автор рекомендаций W3C, описывавших семантику RDF 1.0 и OWL 1, сформулировали так называемую «проблему фреймов», получившую в том числе и действительно философские обобщения.

Систему фреймов можно представлять как иерархию минимальных описаний ситуаций. Человеческое сознание ориентируется в огромном потоке поступающей информации, не осмыслия огромную ее часть как не имеющую отношения к ситуации. Однако, если какая-то часть этой информации вдруг окажется необходимой, сознание способно мгновенно на нее переключиться. Таким образом, в утрированном виде проблема заключается в том, что релевантными ситуациями могут оказаться достаточно произвольные сведения, и о минимализме описания придется забыть. Проблема фреймов имеет больше отношения к вопросам построения «general AI» (искусственного интеллекта общего назначения), его поведению в динамических контекстах, и далее нами не рассматривается.

Подход семантических сетей может быть возведен к экзистенциальным графикам Пирса; законченное выражение он получил в концептуальных графах Джона Совы. Вероятно, скоро придется говорить о ренессансе этого подхода в связи с развитием графовых СУБД, работающих с моделью LPG.

Влияние этих подходов можно проследить в современных формализмах представления знаний: в Common Logic (стандартизированной как ISO/IEC 24707), в F-Logic (повлиявший на рекомендацию W3C «Rule Interchange Format»). В том и в другом случае имеет место позиционирование как форматов скорее обмена, чем собственно представления, что можно рассматривать как свидетельство неуспешности и непринятия индустрией (RDF в версии 1.0 также позиционировался как формат обмена).

Доминирующими среди формализмов представления знаний в настоящее время являются дескрипционные логики, о которых будет рассказано в следующем параграфе. Сами они достигли высокой степени зрелости, исследования обыкновенно сводятся

к попыткам добавить в них что-то из репертуара неклассических или нестандартных логических систем – например, немонотонность. К сожалению или к счастью, эти расширения не претендуют войти в состав грядущих версий стандарта OWL, основанного на дескрипционных логиках.

## § 2. Проблемы философии языка

### О функциях языка

Могут быть выделены следующие функции естественного языка:

– *коммуникативная*. Функция чистого информирования, полагается основной и порой единственной, прочие более или менее успешно могут быть отнесены к языковой прагматике;

– *императивная*. Вполне вероятно, генетически исходная; основоположник интуиционизма голландский математик Л. Э. Я. Брауэр считал ее первой не только исторически, но и логически;

– *кумулятивная*. Também может быть названа мемоизационной, если речь идет об индивидуальном языковом сознании;

– *когнитивная*. А не является ли наше мышление лишь набором речевых автоматизмов? Насколько язык определяет мышление, не ответствен ли он за существование сознания вообще?

Важно отметить, что естественный язык не распадается на соответствующие этим функциям узусы. Но какое отношение это имеет к информационным технологиям?

Историю ИТ можно представить как эволюцию некоторой системы трех тел: языков и технологий выполнения, языков и технологий хранения, языков и технологий передачи и представления. Временами они сближаются друг с другом, временами отдаляются. То и другое бывает аргументировано как соображениями производительности разработчика, так и соображениями производительности компьютера.

Стек технологий Linked Data соответствует стадии сближения: RDF является форматом как представления и передачи, так и хранения. В стеке нет специализированных компонентов исполнения,

предполагается, что исполнение в значительной степени может быть заменено логическим выводом.

В заключение следует отметить, что Semantic Web не является проектом по построению искусственного интеллекта, но лишь неким шагом навстречу ему. Semantic Web предоставляет технологии, являющиеся наиболее перспективными кандидатами на роль lingua franca для коммуникации носителей более или менее общего искусственного интеллекта, в которой они, надо полагать, будут нуждаться: стоит вспомнить, что антропогенез – процесс появления единственного общепризнанного на сегодня носителя общего интеллекта – совпадал с социогенезом.

### **О семантике**

Откройте китайскую «Википедию». Примерно так, как она выглядит для вас, выглядит русская «Википедия» для вашего компьютера. Большинство существующих программных средств не производят смысловую обработку информации, выполняя ее механическую трансформацию по жестким (императивным) алгоритмам.

Термин «семантика» означает наделение информации смыслом, который может быть кем-либо непосредственно обработан. В применении к компьютерным технологиям это значит, что программные средства должны получить те или иные возможности обработки информации с учетом ее смысла, в первую очередь с учетом многообразия связей между объектами. Стандарты RDF и OWL, о которых идет речь в этой книге, преследуют именно эту цель.

Технологическая основа стандартов состоит в том числе в использовании уникальных идентификаторов ресурсов Интернета (URI) в качестве идентификаторов сущностей. Такие идентификаторы используются для обозначения множеств информационных объектов, конкретных объектов, их свойств и связей. При этом подразумевается наличие связи между знаком (URI) и означаемым – тем объектом реального или вымышленного мира, для указания на который используется данный идентификатор.

Механизмом этой связи является *дереференсирование*. Обратившись к URI как к URL, теоретически должно быть возможно получить саму эту сущность (или, точнее, некоторое ее представление). Однако проблема в том, что HTTP предназначен для передачи документов, а не «сущностей», и стандарт, определяющий, например, способ обработки заголовка Content-Range, мог бы стать основой сюжета фантастического романа.

В трекере группы технической архитектуры W3C эта проблема известна как «ISSUE-14». Ее стандартное решение – в ответ на запрос «сущности» сервер возвращает HTTP 303 See Other и перенаправляет на документ, являющийся тем или иным медиапредставлением этой сущности (в зависимости от заголовка Accept): человекочитаемым, и тогда это обычная веб-страница, или машиночитаемым, и тогда это RDF-документ, являющийся результатом выполнения SPARQL-запроса DESCRIBE.

Быть может, однажды и появятся такие протоколы, что использующие их агенты сочтут хрисипповское «когда ты говоришь “тегла”, она проходит через твой рот» троизмом. Но пока соссюровское «означаемое» расщепляется на что-то подобное экстенсионалу и интенсионалу, гуссерлевские «сами предметы» оказываются вполне трансцендентны, а статус существования aristotelевских «первых сущностей» – примерно тем же, что у «вторых».

### О треугольнике Фреге

Отношение ресурса, идентифицируемого URI, и RDF-документа, описывающего этот ресурс, было уподоблено отношению экстенсионала и интенсионала. Что эти термины, которые окажутся полезными и в других ситуациях, обозначают?

Экстенсионал и интенсионал близко соответствуют «объему» и «содержанию» понятия в традиционной логике. Экстенсионал – объект или совокупность объектов, ассоциированных со знаковым средством; интенсионал – набор признаков, однозначно задающих этот объект или совокупность. Знаковые средства с различными интенсионалами могут иметь один и тот же экстенсионал.

В различных ситуациях употребления знакового средства может иметься в виду как его экстенсионал, так и интенсионал. Определяющим маркером интенсионального употребления является отсутствие взаимозаменяемости знаковых средств с одним и тем же экстенсионалом, но различными интенсионалами. Обычно это имеет место в контекстах знания и мнения и в модальных контекстах. Например, вряд ли можно с сохранением истинности и без принятия слишком сильных онтологических предпосылок заменить в предложении «Необходимо, что восемь равняется восьми» первое вхождение «восьми» на «число планет Солнечной системы», несмотря на то, что число планет Солнечной системы равняется восьми. В отсутствие слова «необходимо» это было бы возможно.

Считается, что экстенсионал и интенсионал, объем и содержание связаны «обратным отношением»: увеличение одного приводит к уменьшению другого. Больцано, впрочем, приводит следующий пример.

Рассмотрим понятия «человек, знающий все европейские языки» и «человек, знающий все мертвые европейские языки». Объем первого понятия очевидно меньше, чем объем второго. Однако ведь и содержание первого понятия меньше, чем содержание второго! Или же это так только с точки зрения некоей дистрибутивной семантики, а на самом деле содержание и смысл не просто мешки со словами?

К слову, попробуйте определить понятия EuropeanPolyglot и DeadEuropeanPolyglot в OWL. Это возможно, но понадобится конструкция под названием concept product. Очевидно, что Human that knows only EuropeanLanguage (в манчестерском синтаксисе) или Human  $\sqcap \forall \text{knows}.\text{EuropeanLanguage}$  (в DL-синтаксисе) не то, что нужно.

Противопоставление интенсионального и экстенсионального важно также для понимания некоторых особенностей функционирования движков вывода. Например, оно проливает свет на то, почему на уровне RDFS из

```
:hasParent rdfs:range :Person.  
:hasFather rdfs:subPropertyOf :hasParent
```

не следует

```
:hasFather rdfs:range :Person  
а на уровне OWL – следует.
```

Стоит все же отметить, что наше уподобление отношения между ресурсом, обозначаемым URI, и RDF-документом, описывающим ресурс, отношению экстенсионала и интенсионала не является вполне корректным. В частности, оно не может быть продолжено на URI, обозначающие классы. Классы вводятся вокабуляром RDFS, не указывающим для них никаких специальных способовdereferencирования. Экстенсионал URI класса – сам класс как индивид, а не множество членов класса.

### § 3. Логические основания онтологий

#### Фрагменты логики предикатов первого порядка

Гегель определял логику как изображение Бога, каков он есть в своей вечной сущности до сотворения природы и какого бы то ни было конечного духа, однако на самом деле затрагивает проблематику конечности в различных ее аспектах. Так, имеется множество фрагментов логики предикатов первого порядка, пригодных для использования людьми, не являющимися профессиональными философами или аналитиками.

Например, будущим журналистам и юристам преподают силлогистику. С помощью правил терминов и правил посылок, либо же вовсе заучивая наизусть латинские названия правильных модусов, они научаются определять, является ли умозаключения корректным.

Другой пример – хорновские дизъюнкты: фрагмент логики предикатов, лежащий в основе языка логического программирования Prolog. С использованием линейной резолюции (доказательства истинности утверждений через поиск противоречий) компьютер, а точнее абстрактная машина Уоррена, достигает поставленной ей цели.

К слову, некоторые компоненты технологического стека Semantic Web и Linked Data (язык правил SWRL, OWL-профиль OWL RL, язык запросов SPARQL в версии 1.0) весьма близки к Datalog – определенному подмножеству Prolog.

Общей чертой этих двух фрагментов является то, что они разрешимы: существует (и известен) алгоритм, позволяющий определить, является ли формула «истинной». Сама же логика предикатов первого порядка, хоть и является полной (как доказано Гёдем в теореме о полноте), в отсутствие ограничений на порядок кванторов, число переменных и пр. разрешимой не является (доказано Чёрчем и Тьюрингом).

Еще одним разрешимым фрагментом логики предикатов первого порядка является так называемый «охраняемый» (guarded) фрагмент. Говоря неформально, в охраняемом фрагменте допускается лишь ограниченная квантификация: каждый квантор должен быть «релятивизирован» предикатом от связываемой переменной. Быть может, из учебников математического анализа читатель помнит, как под  $\forall x$  иногда записывалось, например,  $x > 0$ . Этот записываемый под квантором атом является своего рода «охранником», откуда и название фрагмента. Такого рода запись всегда может быть переведена на язык логики предикатов первого порядка, однако обратный перевод возможен не всегда.

Например, формулы  $\forall x (P(x) \rightarrow \exists y Q(x, y))$  и  $\exists x (P(x) \wedge \forall y Q(x, y))$  принадлежат охраняемому фрагменту (подумайте, как записать их «в стиле Фихтенгольца»), а вот  $\forall x (P(x) \wedge \exists y Q(x, y))$  – нет.

Ограничения на структуру формул в охраняемом фрагменте дают возможность с гарантированным успехом применять так называемые табличные (родственные методу аналитических таблиц Бета) методы установления истинности.

Современная логика изучает логические системы в их взаимосвязи. Многие нестандартные и неклассические логические системы могут быть определенным переводены во фрагменты друг друга либо же во фрагменты классической пропозициональной, либо первпорядковой логики. Так называемые дескрипционные

логики, о которых пойдет речь далее, могут быть «переведены» на язык логики предикатов первого порядка, при этом их переводы, как правило, будут размещаться внутри охраняемого фрагмента, чем в основном и обусловлены разрешимость и хорошие вычислительные свойства дескрипционных логик.

### **Дескрипционные логики**

Дескрипционные логики (description logic) – формализм, возникший в рамках исследований в области представления знаний, лишенный недостатков предшествовавших подходов – отсутствия серьезных математических оснований, в частности формальной семантики (пытавшиеся изучать BPMN и другие «инструменты аналитика» поймут, о чём речь).

Причины предпочтение дескрипционные логики самой первопорядковой логике (помимо разрешимости и хороших вычислительных свойств) для целей моделирования данных и представления знаний – такие характеристики этой последней, как:

- избыточность выразительных возможностей; как следствие, трудности консенсуса в вопросе о выборе способов представления и моделирования;
- низкоуровневость и громоздкость языка; в частности, отсутствие синтаксических вариантов, которые можно использовать в качестве языков разметки.

К примеру, следующие записи в синтаксисе дескрипционных логик по крайне мере существенно короче, чем соответствующие первопорядковые:

- $\text{Human} \sqsubseteq \forall \text{hasParent}.\text{Human}$
- $\text{Parent} \equiv \text{Human} \sqcap \exists \text{hasParent}^{-}.\text{Human}$

В первопорядковом синтаксисе это могло бы быть записано, соответственно, так:

- $\forall x (\text{Human}(x) \rightarrow \forall y (\text{hasParent}(x, y) \rightarrow \text{Human}(y)))$
- $\forall x (\text{Parent}(x) \equiv \text{Human}(x) \wedge \exists y (\text{Human}(y) \wedge \text{hasParent}(y, x)))$

На естественном (русском) языке, соответственно, следующим образом:

- человек – тот, все родители кого тоже люди;
- родитель – человек, являющийся родителем хотя бы одного человека.

Итак, в дескрипционных логиках у нас есть *индивидуы, концепты и роли*. Сущности эти близко соответствуют первопорядковым *константам и одно- и двуместным предикатам*.

Кроме того, у нас есть *аксиомы*, примерно соответствующие первопорядковым *формулам*, позволяющие выражать связи между индивидами, концептами и ролями. Принято говорить о трех разновидностях аксиом (или о трех группах аксиом в их наборах):

- **ABox** (assertional box) – аксиомы с участием индивидов.

Примеры записи аксиом ABox:  $\text{Parent}(\text{John})$ ,  $\text{hasParent}(\text{Alice}, \text{John})$ ,  $\text{Alice} \neq \text{Bob}$ ,  $\text{Alice} \approx \text{Carol}$ .

- **TBox** (terminological box) – аксиомы с участием только концептов. Пример – запись  $\text{Father} \sqsubseteq \text{Parent}$ . Ее первопорядковый аналог –  $\forall x (\text{Father}(x) \rightarrow \text{Parent}(x))$ .

- **RBox** (role box) – аксиомы с участием лишь ролей. Пример – запись вида  $\text{hasFather} \sqsubseteq \text{hasParent}$ . Первопорядковый аналог ее –  $\forall x \forall y (\text{hasFather}(x, y) \rightarrow \text{hasParent}(x, y))$ .

Главное же, в дескрипционных логиках у нас есть *конструкторы концептов* и *конструкторы ролей*. С их помощью можно из имеющихся концептов и ролей создавать новые.

**Конструкторы концептов** – это традиционные «теоретико-множественные»  $\Pi$ ,  $\sqcup$  и  $\neg$  (пересечение, объединение и дополнение) и *ролевые ограничения* (*role restrictions*).

Экзистенциальные и универсальные ролевые ограничения мы уже встречались.  $\exists \text{hasParent}.\text{Human}$  можно прочитать как « тот, у кого хотя бы один родитель – человек », а  $\forall \text{hasParent}.\text{Human}$  – как « тот, чьи все родители – люди ». Дополнительно можно ввести численные ограничения вида  $\geq 3 \text{ hasParent}$  (« имеющий трех родителей или больше ») и т. д.

С помощью ролевых ограничений можно задавать у ролей области определений (domain) и области значений (range).  $\exists \text{hasParent}.\text{T}$

$\sqsubseteq$  Child означает, что все обладатели свойства «иметь родителя» будут детьми, а  $\forall \text{hasParent}.\text{Parent}$  – что все значения свойства «иметь родителя» автоматически станут родителями. Теперь из  $\text{hasParent}(\text{Bob}, \text{John})$  всегда будет следовать  $\text{Parent}(\text{John})$  и  $\text{Child}(\text{Bob})$ . В приведенных выше аксиомах Т – универсальный концепт, но можно вместо этого можно писать, например,  $(\text{Child} \sqsubseteq \neg \text{Child})$ .

**Конструкторы** новых ролей – это, например, обратная роль ( $\circlearrowleft$ ) и цепочка ролей ( $\circ$ ). Аксиома  $\text{hasChild} \equiv \text{hasParent}^\circlearrowleft$  значит то же, что  $\forall x \forall y (\text{hasParent}(x, y) \equiv \text{hasChild}(y, x))$ . Аксиому  $\text{hasParent} \circ \text{hasBrother} \sqsubseteq \text{hasUncle}$  можно перевести на язык логики предикатов первого порядка как  $\forall x \forall y \forall z (\text{hasParent}(x, y) \wedge \text{hasBrother}(y, z) \rightarrow \text{hasUncle}(x, z))$ .

Часто отдельно говорят еще о так называемых *характеристиках* ролей: рефлексивности, транзитивности, симметричности, функциональности и других. Однако характеристики ролей обычно можно записать с помощью аксиом и конструкторов. Например, транзитивность роли hasAncestor можно записать как  $\text{hasAncestor} \circ \text{hasAncestor} \sqsubseteq \text{hasAncestor}$ .

Благодаря ограниченности набора конструкторов, мы не можем записывать аксиомы, соответствующие произвольным первопорядковым формулам. В «переводе» на язык логики предикатов первого порядка мы будем получать лишь формулы из некоторого ее ограниченного фрагмента. Хочется, чтобы этот фрагмент был разрешимым и чтобы типовые задачи имели хорошую (малую) вычислительную сложность. Под типовыми задачами имеются в виду задачи определения принадлежности индивидов концептам, выполнимости концептов, их вложенности друг в друга, общей непротиворечивости набора аксиом.

В зависимости от набора разрешенных к использованию конструкторов вычислительная сложность задач будет разной. Представление о том, какова вычислительная сложность при использовании того или иного набора конструкторов

дает интерактивный «Description Logic Complexity Navigator» Евгения Золина, доступный онлайн по адресу: <http://www.cs.man.ac.uk/~ezolin/dl>.

Многие конструкторы имеют собственные однобуквенные названия. Из букв, соответствующих разрешенным к использованию конструкторам, строится название той или иной дескрипционной логики. Отдельно нужно отметить заключающую обычно в скобки  $D$  (от «domain», но можно считать, что от «datatype»). На практике она обозначает, что есть индивиды специального вида – экземпляры примитивных типов (числа, строки и т. д.).

Забегая вперед, можно сказать, что онтология в ИТ-значении – это попросту теория в дескрипционной логике в том же самом смысле, в котором говорят о теории в логике предикатов первого порядка: сигнатура и набор аксиом, описывающих функционирование внеродических символов, в нее входящих.

К слову, в многотомном Handbook of Philosophical Logic [2] нет раздела о дескрипционных логиках (как, впрочем, и о подструктурных логиках, в сфере ИТ, имеющих отношение к концепции владения в Rust). Но есть отдельный The Description Logic Handbook [3].

### **Так причем тут логика?**

По странному недоразумению моделированием иногда считается выписывание различных ограничений, которым должны удовлетворять данные. Вместо этого хотелось бы, чтобы моделирование в компьютерных науках было в чем-то сродни математическому моделированию: давало на выходе некое дополнительное знание. Пусть даже тривиальное, но избавляющее от этой тривиальности разработчика и тем более пользователя. Чтобы данные становились похожи на знания, чтобы бизнес-логика выражалась на подходящем для этого языке или же чтобы ею называлось не нечто, на 90 % относящееся к логике предикатов первого порядка.

## § 4. Semantic Web и Linked Data

### 1. Semantic Web

Эволюцию сети Интернет можно условно представить следующим образом – или же говорить о его сегментах (возможно, вложенных), формировавшихся в указанном ниже порядке:

1) «Документы в Интернете». Ключевые технологии – Gopher, FTP и т. п. Интернет является глобальной сетью для обмена локальными ресурсами.

2) «Интернет документов». Ключевые технологии – HTML и HTTP. Характер выставляемых ресурсов учитывает особенности среды их передачи.

3) «Данные в Интернете». Ключевые технологии – SOAP и REST API, XHR и пр. Эпоха интернет-приложений, потребителями ресурсов являются не только люди.

4) Этап «Интернет данных». Ключевые технологии – технологии Linked Data.

Этот четвертый этап, наступление которого предвидит Тим Бернерс-Ли, создатель ключевых технологий «Интернета документов» и директор W3C, и называется Semantic Web. Технологии Linked Data призваны сделать данные в сети Интернет машиночитаемыми и даже машинопонимаемыми.

Из дальнейшего читателю станет ясна аналогия между ключевыми понятиями второго этапа и ключевыми понятиями четвертого: аналогами URL является URI, аналогом HTML является RDF, а HTML-гиперссылкам аналогичны вхождения URI в RDF-документы.

Semantic Web – это, скорее, системное видение будущего Интернета, чем конкретный стихийный или лоббируемый тренд (хотя, конечно, находится в общем контексте развития ИТ и испытывает влияние трендов). Например, принципиальной характеристикой того, что называется «Web 2.0», считается «создаваемое пользователями содержимое». Учитывать ее призваны, в частности рекомендация W3C «Web Annotation Ontology», и такое начинание, как Solid<sup>2</sup>.

---

<sup>2</sup> <https://solid.mit.edu/>

## 2. Linked Data

Бернерс-Ли определял Linked Data как «правильно сделанный» семантический веб: совокупность подходов и технологий, позволяющую достичь его конечных целей.

Бернерс-Ли выделял следующие базовые принципы Linked Data:

1) Использование URI для именования сущностей. URI являются глобальными идентификаторами сущностей в противоположность локальным строковым идентификаторам записей. Впоследствии лучшее выражение этого принципа нашел в слогане Google Knowledge Graph: «*Things, not strings*».

2) Использование URI в схеме HTTP, чтобы их было возможно дереференсировать. Обратившись к URI по протоколу, должно быть возможно получить означаемое, стоящее за этим означающим (здесь понятна аналогия с называнием оператора \* в С). Точнее, некоторое представление этого означаемого – в зависимости от значения HTTP-заголовка `Accept`: Быть может, с наступлением AR/VR можно будет получить и *сам* этот ресурс, пока же это, вероятнее всего, будет RDF-документ, являющийся результатом SPARQL-запроса DESCRIBE.

3) Использование стандартов W3C, в первую очередь RDF(S), и SPARQL в частности, при дереференсировании URI. Этот набор стандартов и технологий будет подробно рассмотрен нами далее.

4) Использование при описании сущностей ссылок на другие URI. RDF дает возможность ограничиваться словесным описанием, и четвертый принцип призывает этого не делать. При соблюдении принципа 1 появляется возможность при описании ресурса ссылаться на другие ресурсы, в том числе «чужие», отчего данные и называются связанными. На самом деле, как мы увидим, довольно неизбежно использование словаря RDFS.

Далее описываются отдельные «слои» стека технологий Linked Data, известного также под названием Semantic Web Layer Cake.

**RDF** (Resource Description Framework) – формализм описания взаимосвязанных сущностей.

О сущностях и их взаимосвязях делаются утверждения вида «субъект-предикат-объект», называемые триплетами. В простейшем случае и субъект, и предикат, и объект – это URI. Один и тот же URI в различных триплетах может находиться в различных позициях: быть и субъектом, и объектом, и предикатом; тем самым триплеты образуют своего рода граф, называемый RDF-графом.

Субъекты и объекты могут быть не только URI, но и так называемыми *пустыми узлами*, а объекты могут быть еще и *литералами*. Литералы – экземпляры примитивных типов, состоящие из строкового представления и указания типа.

Примеры записи литералов в синтаксисе Turtle: "5.0"^^xsd:float и "five"^^xsd:string. Литералы с типом rdf:langString могут быть снабжены еще и языковым тегом, в Turtle это записывается так: "five"@en и "пять"@ru.

Пустые узлы – «канонимные» ресурсы без глобальных идентификаторов, своего рода экзистенциальные переменные, о которых, однако, могут делаться утверждения.

Часто возникает вопрос, почему, например, предикаты не могут быть пустыми узлами. Вероятная причина – желание неформально понимать и переводить на первопорядковый язык триплет «*s p o*» как нечто наподобие *p(s, o)*, где *p* – предикат, *s* и *o* – константы. Следы такого понимания есть в «LBase: Semantics for Languages of the Semantic Web», имеющем статус заметки рабочей группы W3C. При таком понимании триплет *s p []*, где [] – пустой узел, будет переведен как *p(s, x)*, где *x* – свободная переменная, но как тогда перевести *s [] o*? Имеющий статус рекомендации W3C документ «RDF 1.1 Semantics» предлагает другой способ перевода, но возможность предикатов быть пустыми узлами все равно не рассматривает.

Итак, в этом, собственно, и заключена вся суть RDF:

- субъект – это URI или пустой узел;
- предикат – это URI;
- объект – это URI, пустой узел или литерал.

RDF – абстрактная модель представления, она может быть записана (*сериализована*) в различных синтаксисах: RDF/XML, Turtle (наиболее человекочитаемый), JSON-LD, HDT.

Один и тот же RDF может быть сериализован в RDF/XML различными способами, поэтому, например, итоговый XML бессмысленно валидировать с помощью XSD или пытаться извлекать данные с помощью XPath. Равным образом JSON-LD вряд ли удовлетворит желание рядового Javascript-разработчика работать с RDF с использованием точечной и квадратно-скобочной нотации (хотя JSON-LD движется в этом направлении, предлагая механизм *фрейминга*).

Большинство синтаксисов предлагает способы сокращения длинных URI. Например, в Turtle объявление `@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>` позволит в дальнейшем вместо `<http://www.w3.org/1999/02/22-rdf-syntax-ns#type>` писать `rdf:type`.

**RDFS** (RDF Schema) – базовый словарь моделирования, вводит понятия свойства и класса и такие свойства как `rdf: type`, `rdfs: subClassOf`, `rdfs: domain` и `rdfs: range`. С помощью словаря RDFS могут быть записаны, например, следующие верные с точки зрения их смысла выражения:

<code>rdf:type</code>	<code>rdf:type</code>	<code>rdf:Property</code> .
<code>rdf:Property</code>	<code>rdf:type</code>	<code>rdfs:Class</code> .
<code>rdfs:Class</code>	<code>rdfs:subClassOf</code>	<code>rdfs:Resource</code> .
<code>rdfs:subClassOf</code>	<code>rdfs:domain</code>	<code>rdfs:Class</code> .
<code>rdfs:domain</code>	<code>rdfs:domain</code>	<code>rdf:Property</code> .
<code>rdfs:domain</code>	<code>rdfs:range</code>	<code>rdfs:Class</code> .
<code>rdfs:label</code>	<code>rdfs:range</code>	<code>rdfs:Literal</code> .

RDFS является словарем описания и моделирования, но не является языком ограничений (впрочем, официальная спецификация<sup>3</sup> и оставляет возможность подобного употребления), слово «Schema» не следует понимать в том же смысле, что и в выражении «XML Schema». Например, `:author rdfs:range :Person` означает, что

<sup>3</sup><https://www.w3.org/TR/rdf-schema/>

`rdf:type` всех значений свойства `:author – :Person`, но не означает, что об этом должно быть сказано заранее.

**SPARQL** (SPARQL Protocol and RDF Query Language) – язык запросов к RDF-данным. В базовом варианте SPARQL-запрос представляет собой набор *образцов* триплетов, с которыми соопределяются триплеты исходного графа. В образцах триплетов в позициях субъектов, предикатов и объектов могут находиться переменные.

Запрос возвратит такие значения переменных, при подстановке которых в образцы может получиться подграф опрашиваемого RDF-графа: подмножество его триплетов. Одноименные переменные в различных образцах триплетов должны иметь при этом одни и те же значения.

Например, на приведенном выше наборе из семи RDFS-аксиом следующий запрос вернет `rdfs:domain` и `rdfs:range` в качестве значений `?s` и `?p` соответственно:

```
SELECT * {  
?s ?p rdfs:Class.  
?p ?p rdf:Property.  
}
```

Следует отметить, что SPARQL декларативен и не является языком описания обхода графа (но многие RDF-хранилища предлагают способы корректировки плана выполнения запроса). В связи с этим некоторые типичные графовые задачи, например поиск кратчайшего пути, не могут быть решены на SPARQL (в том числе с использованием синтаксиса `property paths`). Некоторые RDF-хранилища предлагают специальные расширения для решения таких задач.

SPARQL не разделяет презумпцию открытости мира (см. о ней далее) и следует подходу «*negation as failure*», в нем возможны такие конструкции, как `FILTER NOT EXISTS {...}`, распределенность же данных учитывается с помощью механизма федеративных запросов.

Точка доступа SPARQL – RDF-хранилище, способное обрабатывать SPARQL-запросы, – не имеет прямых аналогов

из второго этапа развития Интернета (см. начало данного параграфа). Уподобить ее можно базе данных, на основе содержимого которой генерировались HTML-страницы, но непосредственно доступной вовне. Точка доступа SPARQL является аналогом, скорее, точки доступа API из третьего этапа, однако с двумя основными отличиями. Во-первых, она предоставляет возможность объединять несколько «атомарных» запросов в один (что считается ключевой характеристикой GraphQL), во-вторых, такой API полностью самодокументирован (то, чего пытался достичь HATEOAS).

Имеются и более «легковесные» способы организации доступа к RDF-данным, например, Linked Data Fragments (LDF) и Linked Data Platform (LDP).

**OWL** (Web Ontology Language) – формализм представления знаний, синтаксический вариант дескрипционной логики SROIQ(D) (всюду ниже правильнее говорить OWL 2, первоначальная версия OWL была основана на SHOIN(D)).

Концептам дескрипционных логик в OWL соответствуют *классы*, *ролям – свойства, индивиды* сохраняют свои прежнее название. Аксиомы также называются *аксиомами*.

Например, в манчестерском синтаксисе, использующемся в Protégé, уже известная нам аксиома  $\text{Parent} \equiv \text{Human} \sqcap \exists \text{hasParent}^{-}.\text{Human}$  будет записана так:

**Class:** Human

**Class:** Parent

**EquivalentClass:** Human and (inverse hasParent)  
        some Human

**ObjectProperty:** hasParent

Имеются и другие синтаксисы для записи OWL, например функциональный синтаксис, используемый в официальной спецификации, и OWL/XML. Кроме того, OWL может быть сериализован в абстрактный синтаксис RDF и в дальнейшем – в любой из конкретных.

OWL в отношении к RDF выступает в двояком отношении. Его, с одной стороны, можно рассматривать как некий словарь,

расширяющий RDFS. С другой – OWL все же более мощный формализм, не все конструкции которого можно записать посредством единственного RDF-триплета.

В зависимости от того, какое подмножество конструкций OWL разрешено использовать, говорят о так называемых *профилях* OWL. Стандартизованные и наиболее известные – это OWL EL, OWL RL и OWL QL. Выбор профиля влияет на вычислительную сложность типовых задач. Полный набор конструкций OWL, соответствующий SROIQ(D), называется OWL DL. Также говорят об OWL Full, в котором конструкции OWL разрешено использовать с полной свободой, присущей RDF, без семантических и вычислительных ограничений SROIQ(D). Например, нечто может быть и классом, и свойством. OWL Full неразрешим.

Ключевые принципы присоединения следствий в OWL – принятие презумпции открытого мира (open world assumption, OWA) и отказ от презумпции уникальности имен (unique name assumption, UNA). Ниже мы увидим, к чему могут приводить эти принципы, и познакомимся с некоторыми конструкциями OWL.

Пусть онтология содержит следующий фрагмент (в манчестерском синтаксисе):

```
Class: manyChildren
    EquivalentTo: Human that hasChild min 3
Individual: John
    Types: Human
Facts: hasChild Alice, hasChild Bob, hasChild Carol
```

Будет ли из сказанного следовать, что Джон многодетен? Отказ от UNA заставляет движок вывода ответить на этот вопрос отрицательно, ведь Алиса и Боб вполне могут быть одним и тем же человеком. Чтобы следование имело место, потребуется добавить такую аксиому:

```
DifferentIndividuals: Alice, Bob, Carol, John.
```

Пусть теперь фрагмент онтологии имеет следующий вид:

```
Class: manyChildren
    EquivalentTo: Human that hasChild min 3
```

**Individual:** John

**Types:** Human, manyChildren

**Facts:** hasChild Alice, hasChild Bob

**DifferentIndividuals:** Alice, Bob, Carol, John.

Будет ли эта онтология противоречивой (что может быть интерпретировано как показатель невалидности данных)? Принятие OWA заставит движок вывода ответить отрицательно. «Где-то еще» вполне может быть сказано, что Кэрол также является ребенком Джона.

Чтобы исключить возможность этого, добавим новый факт о Джоне:

**Individual:** John

**Facts:** hasChild Alice, hasChild Bob, not hasChild Carol.

Чтобы исключить появление и других детей, скажем, что все значения свойства «иметь ребенка» – люди, которых у нас всего четверо (и теперь онтология будет противоречивой):

**ObjectProperty:** hasChild

**Domain:** Human

**Characteristics:** Irreflexive

**Class:** Human

**EquivalentTo:** {Alice, Bill, Carol, John}.

Последней из аксиом мы в каком-то смысле «замкнули» мир, и обратите внимание, каким способом исключена возможность того, что Джон является ребенком самому себе.

### 3. Linking Enterprise Data

Набор подходов и технологий Linked Data первоначально предназначался для публикации данных в сети Интернет. Использование этих подходов и технологий во внутрикорпоративной среде сталкивается с рядом затруднений.

Например, в замкнутой корпоративной среде оказывается слишком слабой дедуктивная сила OWL, основанного на принятии OWA и отказе от UNA – решениях, обусловленных открытым и распределенным характером веба. Здесь возможны следующие выходы.

1. Наделение OWL семантикой, предполагающий отказ от OWA и принятие UNA, реализация соответствующего движка вывода. По такому пути идет Stardog.

2. Отказ от дедуктивных возможностей OWL в пользу движков правил – Stardog поддерживает SWRL; Jena и GraphDB предлагают собственные языки правил.

3. Отказ от дедуктивных возможностей OWL, использование для моделирования того или иного его подмножества, близкого к RDFS, – см. об этом далее.

Другая проблема: отсутствие в стеке Linked Data инструментов валидации данных, в то время как в корпоративном мире более существенное внимание уделяется проблемам качества данных. Выходы здесь следующие:

1) Использование для валидации конструкций OWL с семантикой закрытого мира и уникальности имен при наличии соответствующего движка вывода.

2) Использование SHACL, стандартизованного уже после того как перечень слоев Semantic Web Layer Cake был зафиксирован (впрочем, он может использоваться и в качестве движка правил) или ShEx.

3) Осознание того, что все в итоге делается SPARQL-запросами, создание собственного несложного механизма валидации данных с их использованием.

Впрочем, даже полный отказ от дедуктивных возможностей и инструментов валидации оставляет стек Linked Data вне конкуренции в задачах, ландшафтно сходных с открытым и распределенным вебом – в задачах интеграции данных.

Анализ внедрений показывает, что этот чаще всего используется в задачах, связанных с распределенностью и гетерогенностью данных, например при построении систем класса DWH (Data Warehouse) и MDM (Master Data Management). Такие задачи имеются в любой отрасли.

Что касается применений с отраслевой спецификой, в настоящее время технологии Linked Data и производные от них наиболее популярны в следующих отраслях:

- биомедицинские технологии (где их популярность, по-видимому, обусловлена сложностью предметной области);
- изготовление и эксплуатация сложных изделий (крупное машиностроение, добыча нефти и газа);
- финансовые организации (даже XBRL можно рассматривать как некий гибрид SDMX и онтологии RDF Data Cube);
- вопросно-ответные системы, имеющие коммерческое применение (IBM Watson, Apple Siri, отчасти Google Knowledge Graph);
- публикация структурированных данных – впрочем, это с большим основанием может быть отнесено уже к Linked Open Data.

Представление о проникновении в практику подобных технологий может дать и список пользователей RDF-хранилища Stardog<sup>4</sup> (на сайте Stardog в разделе «Customers»).

Но, как бы то ни было, в гарнитурном «Hype Cycle for Emerging Technologies» 2016 года<sup>5</sup> Enterprise Taxonomy and Ontology Management помещается в середине спуска в «долину разочарования» с перспективой выхода на «плато продуктивности» не ранее чем через 10 лет.

#### 4. Connecting Enterprise Data

В «Hype Cycle...» Gartner на 2018 год указан другой восходящий тренд – Knowledge Graphs. Произошла своего рода реинкарнация: под влиянием запросов пользователей и навыков разработчиков графовые СУБД, на которые оказалось отчасти переключено внимание первых и силы последних, начали обретать контуры и позиционирование своих «предков».

Производители практически каждой графовой СУБД объявляют свои продукты подходящей платформой для создания корпоративного «графа знаний» («linked data» порой заменяется на «connected data»), однако насколько оправданы подобные притязания?

Графовые базы данных по-прежнему асемантичны, каждая из них – все тот же data silo, то есть хранилище данных, не имеющих явно указанного машинно-обрабатываемого смысла.

---

<sup>4</sup> <https://www.stardog.com/>

<sup>5</sup> <https://www.gartner.com/newsroom/id/3412017>

Строковые идентификаторы вместо URI превращает задачу интеграции двух графовых СУБД в традиционную задачу интеграции, в то время как интеграция двух RDF-хранилищ зачастую сводится просто к объединению двух RDF-графов.

Другой аспект асемантичности – нерефлексивность графовой модели LPG, делающая затруднительным управление метаданными с использованием той же платформы.

Наконец, графовые СУБД не имеют движков вывода и движков правил. Результаты их работы могут быть воспроизведены усложнением запросов, но такое было возможно и в SPARQL, и даже в SQL.

Впрочем, ведущие RDF-хранилища не испытывают затруднений с поддержкой графовой модели LPG. Blazegraph, ставший впоследствии основой Amazon Neptune, предложил модель RDF\*, являющуюся обобщением RDF и LPG; Stardog также поддерживает язык запросов Gremlin. Ontotext GraphDB, наоборот, сравнительно недавно от этой поддержки отказалась.

Подводя итог краткому обзору состояния дел на рынке технологий и программных продуктов для работы с графиками, отметим, что с точки зрения функциональности (в том числе возможности выполнения логических вычислений) и долгосрочных перспектив использование продуктов, поддерживающих технологии стека Semantic web, представляется наиболее разумным и стратегически оправданным решением для создания корпоративных систем обработки знаний.

### **Список библиографических ссылок**

1. Минский М., Пейперт С. Персептроны (Perceptrons). М. : Мир, 1971. 261 с.
2. Handbook of Philosophical Logic / eds. D. M. Gabbay, F. Guenther. Springer, 2001. 396 p.
3. The Description Logic Handbook / eds. F. Baader, D. Calvanese Cambridge University Press, 2007. 564 p.

---

---

### **Дополнительная литература**

1. Philosophical Engineering: Toward a Philosophy of the Web /eds. H. Halpin, A. Monnin. Wiley, 2014. 216 p.
2. Semantic Web for the Working Ontologist /eds. D. Allemang, J. Hendler. Elsevier, 2011. 330 p.
3. Handbook on Ontologies /eds. S. Staab, R. Studer. Springer, 2009. 811 p.
4. Linking Enterprise Data /ed. D. Wood. Springer, 2011. 291 p.
5. Keet M. An Introduction to Ontology Engineering. College Publications, 2018. 256 p.
6. Sowa J. Knowledge Representation: Logical, Philosophical, and Computational Foundations. Brooks/Cole, 2000. 594 p.

## Глава 2

# УПРАВЛЕНИЕ РЕЗУЛЬТАТАМИ. МОДЕЛЕ-ОРИЕНТИРОВАННАЯ СИСТЕМА УПРАВЛЕНИЯ

С. З. Гумеров

## § 1. Цифровизация и проблемы экономики

Можно предположить, что социальные и экономические отношения должны описываться законами, аналогичными законам физики. Золотой век физики и ее всемерного применения в технике пришелся на XX век. Процветанию же социально-экономических отношений ведущие умы должны посвятить век текущий. И связано это процветание будет с внедрением в управление высокоточных физико-математических, инженерных методов. В противном случае мы столкнемся с интеллектуальной деградацией общества, обострением конфликтов и увеличением числа антропогенных катастроф.

Негативной особенностью сегодняшнего состояния общества является высокая инертность в выработке и принятии важных решений, низкая способность анализировать и предупреждать появление ошибок, ориентир на сиюминутные выгоды.

При этом решения в обществе и экономике разрабатываются и принимаются конкретными субъектами. Многие принимаемые решения базируются на их личных интересах и не учитывают последствий этих решений для общества и окружающей его среды. Проблема состоит не только в субъективном эгоизме, но и в недооценке замкнутости и внутренней связности системы, которая нас окружает, обеспечивая безусловную обратную связь – «бумеранг» последствий неадекватных решений.

Отсутствие практики и готовых инструментов комплексной оценки последствий принимаемых решений, учитывающих динамику внешней для интересанта среды, снижают рациональность и результативность общественных, экономических и корпоративных отношений.

Новые поколения социальных субъектов тратят ресурсы на то, чтобы, отринув накопленные знания предшественников, доказать окружающим свое превосходство, завладеть правом распоряжаться общественными благами. В этой гонке появляются новые термины для старых предметов, новые товары и услуги, позволяющие переключить внимание потребителя с насущных задач на задачи вымышленные. Такая негативистская стратегия не может дать тот устойчивый результат, теоретическое обоснование которого уже принято специалистами [1, р. 27].

Мир высоких технологий, сконцентрировавший умнейших людей планеты в Кремниевой долине, огромными темпами наращивает капитал – 177% за последние 6 лет (табл. 2.1). Спектр продукции и услуг этих мощных корпораций не решает базовых проблем общества [2], но снижает возможности научно-технологического развития традиционных отраслей экономики. Среднегодовой индекс промышленного производства Dow Jones, объединяющего базовые для экономики компании США, за этот же шестилетний период вырос всего на 60%, с 13 000 до 20 815 пунктов [3].

Таблица 2.1

**Темпы роста капитализации [4]**

Показатель	Ед. изм.	Apple Inc.	Google – Alphabet Inc.	Microsoft Corp.	Facebook	Amazon com Inc.	Всего FAAMG
Капитализация, 2017	млрд. долл.	807	685	581	504	473	<b>3050</b>
Капитализация, 2012	млрд. долл.	446	267	202	66	123	<b>1103</b>
Рост	%	81	157	188	667	284	<b>177</b>

Разумеется, сама по себе капитализация не может быть ни индикатором успешности развития общества, ни, тем более, целью управления. Однако на сегодняшний день именно капитал является одним из основных инструментов, при помощи которого

транснациональные компании со штаб-квартирами преимущественно в США, данные по которым приведены в статистической таблице, формируют ключевые тренды развития технологий для всего мира.

Как стимулировать развитие экономики в целом, обеспечивая баланс не только корпоративных интересов, но и интерес наименее консолидированной отрасли экономики – личных домохозяйств? Ведь именно благосостояние населения является источником и целью экономического процветания.

Налицо явные дисбалансы между темпами роста:

1. Новой и старой отраслей экономики.
2. Корпоративной и потребительскими сферами.
3. Развитыми и развивающимися странами.

Это создает предпосылки как для социально-экономических кризисов, так и для новых потенциалов развития.

Одним из заметных субъектов на технологическом рынке, обозначившим возможности повышения рационального использования ресурсов для реализации экономических потенциалов в традиционной (капиталоемкой) экономике, стала компания IBM, которая в 2008 году объявила о инициативе Smarter Planet. Ключевой идеей данной компании стала адаптация технологий новой экономики для решения задач экономики традиционной. Появились яркие, привлекательные лозунги данных инициатив «Digital Oil Field», «Smart City», «Smart Manufacturing» и так далее<sup>1</sup>.

Маркетинговая инициатива обрастила последователями на рынке поставщиков технологических решений и консалтинговых компаний [5]. В 2014–2015 годах к лозунгам добавились новые – «Big Data», «Machine Learning», «Artificial Intelligence», «Blockchain», «Internet of Things», «Digital Twin» и прочие. На этой волне и в российском обществе в 2017 году официально закрепились лозунги

<sup>1</sup> Стоит отметить, что трансформировать в технологический и экономический успех данную инициативу компания IBM не смогла. С 2012 по 2018 год ее выручка снизилась на 23 %, как и выручка иных технологических монстров, сформировавших в свое время капитал на производстве вычислительной техники: Hewlett-Packard (падение на 53 %), Cisco (стагнация выручки на уровне 1 %).

«Цифровая экономика» и «Умный город», подкрепленные государственным и квазигосударственным<sup>2</sup> финансированием в 3,5 триллиона рублей [6].

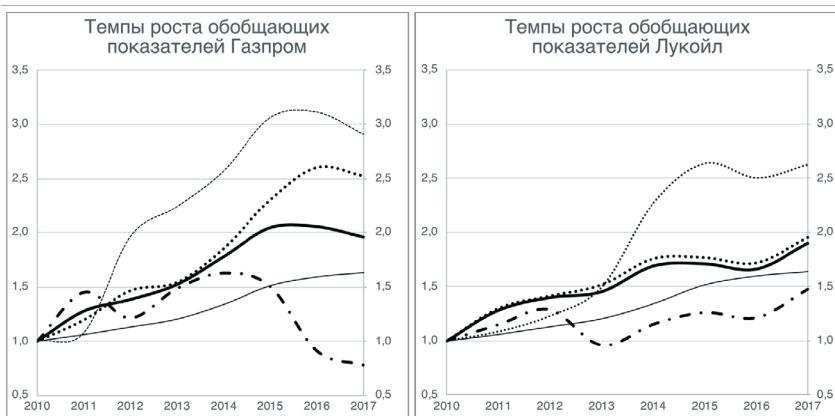
Несмотря на ажиотаж вокруг подобных технократических идей, за прошедшие 10 лет не была выработана единая идеология и технологическая платформа. Производители техники и технологий не смогли продемонстрировать существенных экономических прорывов в области традиционной экономики.

Субъекты традиционной экономики отчасти поддерживают «разогрев» технологического рынка, в особенности те сферы, которые аккумулируют капитал ввиду удачной рыночной или государственной конъюнктуры. Так, ПАО «Россети», государственная компания – естественный монополист рынка транспортировки электрической энергии в России, анонсировало инвестиционную программу в размере 1,7 триллиона рублей до 2030 года [7]. При этом цена на электроэнергию для потребителей растет и, вероятно, будет продолжать расти преимущественно за счет тарифа на транспортировку электроэнергии темпами, опережающими ВРП и доходы населения.

Нефтедобывающие компании с их инициативами по Digital Oil Field также являются ключевыми спонсорами проектов, призванных поставить новую экономику на службу старой, но оказавшихся пока безуспешными в этом направлении. Достаточно часто за проектами, подобными «Digital Oil Field», стоят политические мотивы, а не необходимость достижения значимых экономических результатов для экономики предприятия или региона. Несмотря на устойчивый рост выручки предприятий нефтегазового сектора, углеводородные компании России демонстрируют снижение экономической эффективности своей деятельности и увеличение нагрузки на экономику своих потребителей. На рис. 2.1 приведена динамика обобщающих показателей экономической эффективности по двум ключевым для российской экономики вертикально-интегрированным нефтегазовым корпорациям – госкомпании ПАО «Газпром» и частной ПАО «Лукойл».

<sup>2</sup>За счет корпораций с преимущественно государственным участием.

Себестоимость реализации добытой и переработанной нефти становится из года в год дороже (толстая сплошная линия), как и себестоимость ее производства (толстая пунктирная линия). Обе линии показывают, что темпы роста себестоимости опережают не только инфляцию, но и рост валового национального продукта Российской Федерации. Одна из составляющих операционных затрат (затраты на восстановление основных фондов) растет быстрее остальных, сигнализируя о низкой эффективности именно капитальныхложений – драйвера, в иных условиях – экономического роста.



*Рис. 2.1. Темпы роста затрат (снижения эффективности) нефтегазовых компаний РФ к уровню 2010 года: толстая пунктирная – операционные расходы на единицу добываемой продукции, включая затраты на поддержание и восстановление основных фондов; тонкая пунктирная линия – затраты на восстановление основных фондов; толстая сплошная линия показывает цену за единицу добываемой продукции для потребителя; штрихпунктирная линия показывает отношение валовой прибыли к объемам добычи; тонкая сплошная линия показывает динамику инфляции (потребительских цен).*

*Источники данных – годовые отчеты компаний: <http://www.lukoil.ru/InvestorAndShareholderCenter/RegulatoryDisclosure/AnnualReport>, <http://www.gazprom.ru/investors/disclosure/reports/>*

Ключевую долю вклада в рост себестоимости продукции обеспечивает снижающаяся эффективность капитальныхложений,

преимущественно в поддержание действующей инфраструктуры активов. Несмотря на то, что тенденция роста себестоимости связана с истощением разведанных запасов углеводородов [8], стоит отметить противоположную тенденцию на снижение стоимости добычи сланцевой нефти и газа в более сложных геологических условиях в США.

Приведенные графики свидетельствуют о снижении экономической эффективности основных отраслей экономики в Российской Федерации, увеличении нагрузки топливно-энергетического сектора на валовый внутренний продукт (рост энергоемкости ВРП) и качество жизни населения. Данный факт также подтверждает отсутствие значимых научно-технологических прорывов в отраслевой науке и производстве, но при этом формирует целевые ориентиры для возможного приложения усилий тех управленцев, которые связывает потенциал роста с более точными, цифровыми технологиями управления.

В экономике США наблюдается обратная ситуация. Использование новых, преимущественно цифровых технологий в разработке и добыче сырья приводит к снижению себестоимости добычи в среднем с 70 (2012) до 30 USD (2016) [9] за баррель нефти, росту конкурентоспособности и соответственно – росту добычи углеводородов с 2,3 до 11 млн баррелей в сутки [10].

Снижение себестоимости производства, рост конкурентоспособности на рынках возможны только при целевом развитии и вовлечении новой экономики в решение технико-экономических задач и вызовов.

На рис. 2.2 можно увидеть тенденцию динамики объемов добычи и затрат, наблюдавшуюся по крупнейшим нефтегазовым компаниям РФ. Объемы добычи растут достаточно слабо на фоне уверенного роста себестоимости продукции<sup>3</sup>. Судя по динамике экономических результатов, проекты развития российских углеводородных компаний, в том числе и по цифровизации, пока не оказали значимого влияния на экономику этих предприятий.

---

<sup>3</sup> Откорректированной на уровень фактической инфляции.

Темпы роста базовых показателей  
от уровня 2010 года

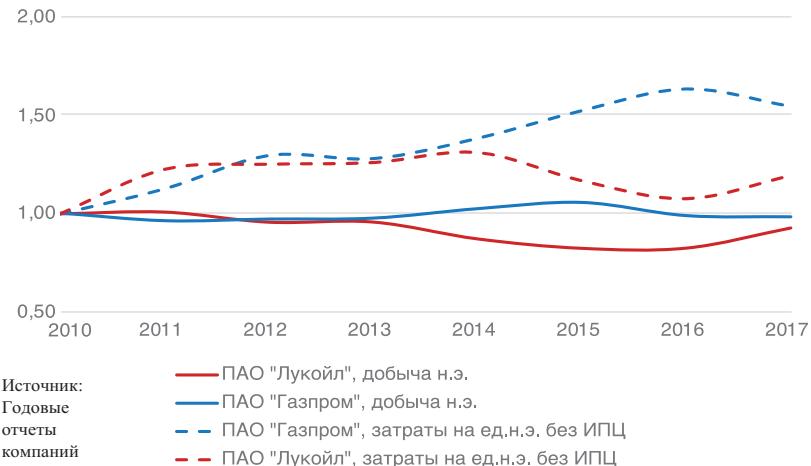


Рис. 2.2. Динамика базовых показателей объемов добычи и удельных затрат крупнейших российских нефтегазовых компаний.

Источники данных – годовые отчеты компаний: <http://www.lukoil.ru/InvestorAndShareholderCenter/RegulatoryDisclosure/AnnualReport>, <http://www.gazprom.ru/investors/disclosure/reports/>

Если же перейти от уровня национальной экономики на региональный уровень, то рост накапливаемой неэффективности в базовых сферах экономической деятельности, таких как система коммунальной инфраструктуры и энергетики, пассажирские перевозки, также нарастает [11].

В частности, при всесторонней оценке технико-экономической динамики систем коммунальной инфраструктуры, энергетики, транспортного обеспечения можно отметить следующие проблемы:

1. Диспропорции роста тарифов и себестоимости продукции по территории, отраслям и хозяйствующим субъектам.
2. Падение эффективности капитальных вложений в развитие экономической деятельности и техническое состояние основных фондов.

3. Несогласованность статистических, прогнозных и плановых сведений для изучения межотраслевых или межтерриториальных последствий государственного и корпоративного регулирования.

Очевидно, что отсутствие единства системы межотраслевого учета и оптимального планирования не позволяет обеспечить ускорение темпов роста экономики, повышает экономическую нагрузку и снижает качество жизни населения.

Основные выводы из приведенных фактов и суждений следующие:

1. Международные, преимущественно американские, компании новой экономики выигрывают конкуренцию за капитал у экономики традиционной. Ключевым инструментом, формирующим этот результат, является вовлечение в экономическую деятельность интеллектуального ресурса (человеческого капитала).

2. Несмотря на международный ажиотаж вокруг технологий по цифровизации, исследований эффективности вклада этих технологий в темпы экономического роста традиционной экономики обнаружить не удалось.

Отдельный прогресс в применении цифровых технологий в традиционном секторе показывает сланцевая нефтегазодобывача США, которая вырвалась из аутсайдеров в международные лидеры [12]. Этот успех стал возможным благодаря собственным научно-технологическим разработкам, обеспечившим предприятиям снижение себестоимости разработки и эксплуатации месторождений. Этот опыт можно отнести к успехам в собственной разработке технологий со значительным цифровым обеспечением.

Российская нефтегазовая отрасль пока не смогла продемонстрировать собственный конкурентоспособный технологический стек, и поэтому ее экономические успехи находятся под возрастающим давлением на международном рынке.

3. Российская экономика в лице ее базовых отраслей снижает свою экономическую эффективность, международную конкурентоспособность и наращивает нагрузку на потребителей в РФ через стоимость топлива [13] и производной от топлива продукции [14].

На первом цикле рост нагрузки на потребителей снижает реальные доходы населения [15], а на втором цикле приводит к снижению темпов роста экономики, к росту зависимости экономики от экспорта углеводородов [16].

Ключевые для экономики РФ корпорации на текущий момент сталкиваются с затруднениями в собственном научно-технологическом развитии из-за высокой рентабельности текущей деятельности, отсутствия позитивного опыта и мотивов для результативной работы по развитию интеллектуального капитала.

4. Инвестиции в цифровизацию традиционной экономики полезны только на возвратной основе.

Результатом данных инвестиций должен стать рост экономической эффективности деятельности традиционных секторов экономики: топливно-энергетического, жилищно-коммунального, транспортного и строительного. Количественно-обоснованная часть данного результата должна быть направлена в формирование источников финансирования нового сектора экономики (отраслевая наука и инженерия), разрабатывающего экономически целесообразные технологии.

5. Стимулирование развития новой экономики в РФ связано с реализацией новой государственной и общественной политики по росту экономической эффективности.

Данная политика реализуема только при совершенствовании и консолидации единой системы управления социально-экономическим развитием на всех уровнях государственного и общественного устройства – переходу от субъективного управления к рационально-обоснованному, цифровому управлению. Такая политика не может реализовываться только за счет рыночного регулирования.

Определение направлений разработки методов количественного управления социально-экономическим развитием (без усиления государственного участия в капитале экономики), совершенствование корпоративных систем управления, обеспечение баланса интересов акционеров предприятия, его потребителей и общества

послужило основой для создания концепции моделе-ориентированной системы управления (МОСУ) и доктрины цифровой трансформации.

Отличительной особенностью формирования новых моделей управления экономикой, в том числе и на государственном уровне, должен стать приоритет результатов над затратами, то есть эффективность в экономическом понимании. Модели управления должны обеспечивать количественно-точную связь результатов и способов распределения ресурсов по структуре экономической деятельности, учитывая мультиплекативные и обратные связи.

Формирование цифровой экономики и умных городов не должно зависеть от набора технологических решений для частных задач экономики. Задачи цифровой экономики – охватить цепочным, количественным описанием поведение социально-экономической системы и ее окружения, нарастить управляемеческую мощь по выявлению, предупреждению и устранению узких мест, сдерживающих рост эффективности. Такой подход позволит формировать долгосрочные задачи не только социально-экономического и инфраструктурного развития, но и развития научно-технологического.

Предельно точная, ресурсно-обоснованная, постановка долгосрочных целей и задач социально-экономического развития способна сформировать новый, устойчивый сектор экономики знаний.

Задача данной главы – сформулировать некоторые тезисы доктрины цифровой трансформации и концепции моделе-ориентированной системы управления традиционной экономикой, создание которых вызвано необходимостью:

- снижения диспропорций социально-экономического развития;
- выявления узких мест, снижающих устойчивость социально-экономического развития;
- повышения результативности корпоративных, государственных и межгосударственных решений;
- перевод достижений новой экономики в экономику традиционную.

## § 2. Цели, задачи и структура моделе-ориентированной системы управления

Появившаяся в 1960-е годы среди руководителей промышленности мода на АСУП привела к тому, что на многих предприятиях торопились приобретать электронно-вычислительные машины, строить свои вычислительные центры и создавать АСУ. При этом не все представляли себе истинные сложности дела. Когда начали реально ощущаться серьезные трудности, пришлось ограничиться частичной автоматизацией документооборота, бухгалтерского учета и еще некоторых второстепенных функций в управлении. Вполне понятно, что такие АСУ не могли дать ожидаемого экономического эффекта.

Специалисты уже давно пришли к выводу о том, что автоматизированная система управления предприятием будет по-настоящему эффективной только тогда, когда наряду с быстрой и точной машинной обработкой самой детализированной информации, на основе применения современных экономико-математических методов и ЭВМ на предприятии будет решаться много новых (в основном – оптимизационных) задач управления» [17].

Целью реализации доктрины цифровой трансформации (далее – доктрина) является *повышение результативности и эффективности использования ресурсов* по всей структуре международной, национальной, региональной и корпоративной экономической деятельности, ее отдельным функциям, этапам жизненного цикла объектов инфраструктуры и механизмам управления (далее рассматриваемым как аналитические срезы).

Принципиальным для доктрины является определение границ и взаимосвязей управляемого объекта с внешним миром. Для этого используется базовое понятие **подконтрольной системы** (ПС) – комплекса, осуществляющего экономическую или социально-экономическую деятельность по удовлетворению потребителей продукции (ресурсов, товаров, работ и услуг),

произведенной (транспортированной, накопленной, распределенной), с заданными параметрами качества. Каждая ПС должна рассматриваться как часть иной подконтрольной системы и состоять из отдельных подконтрольных систем<sup>4</sup>.

Инструментом достижения цели реализации доктрины является **моделе-ориентированная система управления** (МОСУ), позволяющая лицам, принимающим решения, распределять свои усилия и ресурсы на основе *количественно измеримых оценок* последствий действия или бездействия субъектов экономики.

Количественные оценки результатов и затрат являются входными и выходными характеристиками постоянно действующей и непрерывно-уточняемой **интегрированной модели ПС**. Интегрированная модель ПС связывает динамику измеренных показателей (результаты-затраты), с потенциально регулируемыми параметрами ПС, динамикой двустороннего взаимодействия ПС с внешней средой<sup>5</sup> по всей ее структуре. В отдельных источниках интегрированная модель ПС именуется цифровыми двойниками.

Целостность реализации функций управления ПС гарантирует **единую программу развития** ПС, которая количественным<sup>6</sup> образом связывает цели, задачи, меры, мероприятия и отдельные проекты текущего и планового управляющего воздействия на ПС. Единая программа развития является инструментом долго-, средне- и краткосрочного планирования, источником и инструментом приоритезации организационно-технических мероприятий для государственных, инвестиционных и производственных программ, в том числе и проектов по формированию и совершенствованию самой МОСУ.

Концептуальная схема контекста МОСУ представлена на рис. 2.3.

---

<sup>4</sup> По аналогии с понятием «Функциональный блок» языка моделирования IDEF0.

<sup>5</sup> В широкой ее трактовке – социально-экономическую, природно-антропогенную, смежные технологические процессы и субъекты.

<sup>6</sup> Здесь и далее под количественной связью понимается вычислимая, имеющая численную (цифровую) оценку.

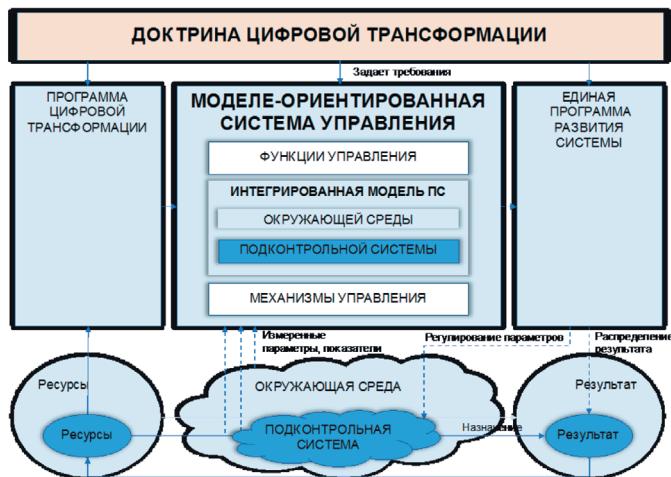


Рис. 2.3. Концептуальная схема MOSU

### § 3. Программа цифровой трансформации

Целью любой программы является достижение заранее определенного результата, в условиях ограниченности ресурсов в этом состоит общность экономических программ и программ для ЭВМ. В цифровую эпоху качество программ экономических, государственных, инвестиционных и производственных должно максимально приблизиться к качеству и культуре производства программного обеспечения. Век информационных технологий подалril обществу существенно новое понимание качества продукции, средств его проектирования, изготовления, тестирования и приемки, абсолютной формализации как языков программирования, так и данных, которые обрабатываются компьютером.

Компьютерная программа, согласно Гражданскому кодексу РФ, представленная в объективной форме совокупность данных и команд, предназначенных для функционирования ЭВМ и других компьютерных устройств с целью получения *определенного результата* (ГК РФ).

Анализ действующих государственных и инвестиционных программ российских монополистов показывает, что, несмотря на формальное наличие целевых блоков программ и выделяемых ресурсов на их достижение, цели программ не только количественно не связаны с мероприятиями, но и не описывают в полной мере технико-экономическое состояние программируемой системы. С точки зрения программной инженерии такие программы неисполнимы, а значит, ошибочны. Управленческой и экономической наукам еще предстоит впитать в себя лучшие практики программной и системной инженерий.

Формирование программ цифровой трансформации, являющихся частью единых программ развития подконтрольных систем, базируется на принципах, которые определяют порядок и структуру будущего программного документа:

**1. Основная проблема и вызов ПС** (подконтрольной системы, объекта моделирования, под которым здесь прежде всего понимается промышленное предприятие) – низкая результативность и эффективность использования ресурсов для реализации ее своего целевого предназначения.

**2. Ответ на вызов** – постоянно действующая программа цифровизации ПС, обеспечивающая переход к объективным, точным методам управления функционированием и развитием ПС, количественно обоснованные корректировки планов, структурно-технологических, территориальных схем и организационно-технических решений, производственных и инвестиционных программ.

**3. Особенность цифровой экономики ПС** – введение в систему управления и распределения ресурсов нового элемента – цифровой постоянно действующей интегрированной модели ПС («цифрового двойника»), реализующей количественно выражаемую причинно-следственную связь параметров ПС и каждого организационно-технического решения или бездействия с плановой результативностью ПС на различных горизонтах планирования и периодах времени (И иных аналитических срезах).

**4. Цель программы цифровизации** – рост эффективности, надежности и безопасности ПС.

**5. Типовой круг задач (разделов) программы:**

- повышение производительности ПС;
- снижение расходов/затрат на производство продукции ПС;
- снижение ущербов от ПС;
- снижение недоотпуска продукции ПС, с заданными параметрами качества;
- снижение вероятности причинения ущербов потребителям, внешней среде;
- снижение себестоимости продаж продукции;
- повышение конкурентоспособности на внешних рынках.

**6. Декомпозиция задач.** Задачи программы раскладываются последовательно по отдельным аналитическим срезам ПС.

**7. Индикаторы программы:** текущее, прогнозное и скользящее плановое технико-экономическое состояние ПС, соотнесенные с ними цели и задачи программы цифровизации ПС, выраженные комбинацией следующих трех базовых показателей: объемов производства, затрат и ущербов в денежном, стоимостном и энергетическом эквивалентах.

**8. Формы капитализируемых, нематериальных результатов каждого мероприятия программы цифровизации:**

- нормативно-техническая и нормативно-правовая документация;
- внедренная в ПС инновационная продукция (вещество, материал, техника, технология);
- интегрированная модель ПС и порядок мониторинга, оценки, прогнозирования, оптимизации, создания ПС с заданными характеристиками;
- банк экспериментальных, статистических данных по ПС;
- программа обучения и аттестации по результатам мероприятий;
- механизм интеграции данных по реализованному аспекту на протяжении жизненного цикла ПС.

9. **Источник финансирования программы** – фонд цифровизации ПС, пополняемый частью экономических результатов ПС, получаемых вследствие реализации программы цифровизации ПС. Первичные средства фонда, достаточные для реализации программы цифровизации ПС в первые годы, формируются из текущих средств ПС на возвратной основе.

10. **Ответственность**. У каждой цели, задачи, результата, индикатора, программы, мероприятия, проекта, аспекта, ПС и его отдельных элементов должны быть определены конкретные лица, несущие ответственность как за принятие решения и определение его плановых результатов, так и за исполнение.

#### **§ 4. Структура моделе-ориентированной системы управления**

Моделе-ориентированная система управления – один из ключевых результатов программы цифровой трансформации и инструмент обеспечения достижения его целей. Отличительной особенностью данного инструмента управления является использование результатов расчета на интегрированной модели ПС при подготовке и оценке эффективности организационно-технических решений.

В состав МОСУ входят следующие компоненты:

- единая система показателей или система координат;
- интегрированная модель ПС (подконтрольной системы);
- функции управления;
- описание структуры управления;
- механизмы управления.

**Система показателей** должна обеспечивать:

- единство измерения экономической (социально-экономической) динамики состояния ПС, сквозной по всем аналитическим срезам, подсистемам и надсистемам;
- согласованность и целостность управления поведением ПС, как во внешней среде, так и в управлении ее внутренним состоянием.

**Функциональный состав** МОСУ включает следующие задачи управления (но не ограничивается ими):

- учет и мониторинг состояния ПС;
- анализ, планирование (целеполагание, прогнозирование, выявление узких мест, угроз и потенциалов);
- оценка принимаемых решений;
- подтверждение качества исполнения решений (рис. 2.4).



Рис. 2.4. Функции управления МОСУ

Большинство предложенных функций определены Федеральным законом «О стратегическом планировании» № 172 от 28 июня 2014 года. Основные понятия, используемые данным законом, применимы на всех уровнях управления, в том числе и на уровне отдельных хозяйствующих субъектов и экономических агентов (там, где это оправдано сложностью подконтрольной системы). Существенным ограничением данного закона является неопределенность методов управления и источника технико-экономических показателей для согласования плановых решений.

Для устранения этого ограничения и обеспечения целостности управления (регулирования), что особенно важно на государственном уровне, вводится понятие *интегрированной модели*, которая позволяет хранить в функционально связном виде как исходные

сведения, так и консолидированные, отчетные, аналитические и прогнозно-плановые данные по динамике подконтрольной системы.

Иной срез интегрированной модели помогает разделить модели по признаку функционального предназначения ПС и средств производства, реализующих это функциональное предназначение. Для государственных систем, соответственно, интегрированная модель рассматривается в аспектах социально-экономической деятельности (функциональное предназначение) и природно-антропогенной системы (средства производства) (рис. 2.5).



Рис. 2.5. Аспекты интегрированной модели ПС

Природно-антропогенная система (средства производства, инфраструктура, физические объекты) имеет ограниченный ресурс и является средством обеспечения социально-экономической деятельности, что налагает ограничения и предъявляет требования к расходованию части результата экономической деятельности на поддержание надлежащего состояния данной системы.

Под *структурой управления* в контексте МОСУ понимается организационная структура предприятия, состоящая из соответствующих организационных единиц (постов, штатных единиц), на которые возложена непосредственная ответственность за динамику показателей отдельных структурных компонентов ПС, выработку, принятие и исполнение управленческих решений.

Состав *механизмов управления* можно рассмотреть в разрезе следующих компонентов:

- совершенствование и развитие системы *нормативных документов*;
- внедрение новых (*инновационных*) видов продукции, материалов, оборудования, технологий;
- развитие системы обучения, подготовки и аттестации *персонала*;
- развитие *методологии оценки, прогнозирования, мониторинга, управления, регулирования и контроля параметров и показателей*;
- получение, накопление и систематизация *экспериментальных данных* по материалам, оборудованию, технологиям и по условиям воздействия внешней среды;
- интеграция информации, получаемой на этапах жизненного цикла подконтрольной системы в *единой информационной среде* [18].

Принципиальным вопросом в предложенном разбиении механизмов управления является разделение понятий *информационного пространства* и *интегрированной модели*. Единое информационное пространство наполняется прежде всего исходной информацией, приходящей от источников данных, достоверность которой может быть верифицирована (хотя и могущей содержать ошибки и фрагменты представлений внешних по отношению к подконтрольной системе субъектов). Интегрированная модель в свою очередь решает задачи анализа, обогащения, высокуюровневой обработки, интерпретации широкого спектра данных поверх информационного пространства.

Информационное пространство должно:

- реализовать сбор, хранение, ассоциативное связывание и представление блоков информации исходной и расчетной информации о ПС и внешней среде;
- обеспечивать построение упорядоченных ассоциативных структур различного рода (набор, упорядоченный набор, граф, ориентированный граф и пр.);
- быстро строить специализированные выборки/представления через ассоциативные структуры;

– обеспечивать возможность функциональной взаимосвязи исходных данных единого информационного пространства с расчетными значениями показателей и параметров интегрированной модели;

– содержать типовые решения для регулярно повторяющихся задач;

– содержать типовые способы решения для нерегулярных задач.

Концепция МОСУ не рассматривает вопросы управления коллективом, межличностными отношениями, а ограничивается задачей количественно-обоснованного управления состоянием ПС, постепенного снижения негативного влияния человеческого фактора на результаты целевой деятельности. Отметим, что формирование целей управления и принятия нетиповых решений остается прерогативой человека, а роль МОСУ состоит только в формировании обоснованных программ достижения поставленных целей.

МОСУ не подменяет, но значительно расширяет понятие автоматизированных систем управления, фокусируя внимание проектировщиков и архитекторов систем управления на обоснованном выборе полезного контура для автоматизации – решении приоритетных социальных, экономических, технико-технологических и природно-антропогенных проблем.

Отсутствие экономических успехов от внедрения АСУ связано, прежде всего с фокусировкой на автоматизации рутинных операций, не всегда обеспечивающих даже прирост производительности труда, не говоря уже о прочих аспектах эффективности. Доля рутинного труда в наукоемких и капиталоемких отраслях (новая и старая экономика) на данный момент крайне мала и не создает значимого потенциала для реализации конкурентных преимуществ экономическим субъектам.

МОСУ обеспечивает постепенный переход от бюрократических и интуитивных принципов управления ПС, реализуемых посредством механизма поручений и доверительной делегации полномочий вышестоящего руководства, к планово-программным механизмам, количественно-обоснованных расчетами на интегрированной модели ПС.

## § 5. Интегрированная модель подконтрольной системы

Создание МОСУ подобно переходу в программной инженерии от процедурного программирования к объект-ориентированной и далее онтологической парадигме. Объектом в контексте МОСУ является ПС (подконтрольная система) и внешняя по отношению к ней среда, а описанием объекта становится *интегрированная модель* ПС. Сегодня интегрированные модели часто называют цифровыми двойниками, включая в это понятие возможности аккумуляции знаний о ПС и проведения сценарных расчетов динамики состояния ПС еще до момента принятия решения.

Интегрированная модель (см. принципиальную схему на рис. 2.6) – это расчетно-аналитический инструмент, позволяющий формировать достоверную (с учетом точности рассматриваемой модели) количественную оценку:

- динамики фактического и прогнозного состояния ПС;
- отклонений фактического состояния от ожидаемых значений;
- величины и места приложения сил, оказавших влияние на динамику фактического состояния ПС;
- последствий бездействия и реализации решений;
- экономического потенциала развития и предельных затрат на него;
- параметров вновь создаваемых или модернизируемых ПС, с новым уровнем результативности и эффективности.

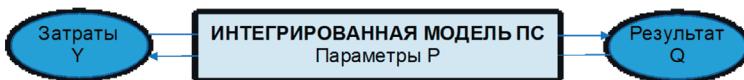


Рис. 2.6. Схема математического описания интегрированной модели

Система показателей, в которой далее математически описывается поведение ПС во внешней среде, включает в себя следующие группы:

- технико-экономические показатели;
- обобщающие показатели;
- относительные коэффициенты или параметры ПС (Р).

Отличительными признаками *технико-экономических показателей* является возможность их измерения и непосредственного регулирования. К технико-экономическим показателям ПС относятся:

- производительность ПС (установленная, располагаемая –  $Q^*$ , фактическая –  $Q$ )\*, – с заданным качеством результата;
- расходуемые ресурсы для обеспечения производительности ( $Y$ );
- ущербы для внешней среды (социальные, экологические, технико-экономический,  $U$ ).

Технико-экономические показатели могут быть представлены в натуральном, стоимостном и энергетическом выражении, в виде группы (вектора) продукции и всегда имеют свою уникальную для вида продукции размерность.

Технико-экономические показатели являются базой для определения **обобщающих показателей** динамики состояния ПС (см. рис. 2.7, табл. 2.2):

Таблица 2.2

### Обобщающие показатели состояния подконтрольной системы и способы их расчета

Показатель	Способ расчета
Надежность*	$R = Q^{\text{факт}} / Q^{\text{план}}$
Безопасность	$S = U^{\text{допустимое}} / U^{\text{фактическое}}$
Эффективность	$E = (U + Y) / Q$

\* Надежность исполнения целевого предназначения (выпуска продукции) с заранее заданными качественными характеристиками.

Обобщающие показатели являются основными для управления развитием ПС, но не доступны для прямого измерения и воздействия. *Повышение точности управления* связано с целевым обеспечением роста обобщающих показателей, определением и постоянным уточнением математического описания интегрированной модели:

$V = F(Q, Y, U, t)$  – описание результативности поведения ПС (1),  
 $G(F(P, Y, t)) = \max < R, S, E >$  – описание функции управления ПС (2).



Рис. 2.7. Система координат состояния ПС в технико-экономических и обобщающих показателях

По типам принимаемых значений показатели разделяются на ретроспективные, фактические, прогнозные (инерционные), сценарные, плановые.

По мере выявления и ранжирования отклонений поведения ПС от целевых (плановых) или прогнозных (инерционных) показателей производится уточнение (детализация) функции ПС по возможным аналитическим срезам.

Декомпозиция или агрегация функций, с определением вида функции и ее параметров, происходит по мере обработки проектных и статистических данных о динамике состояния ПС за фактический период времени. Порядок постепенного определения количественных и качественных характеристик функции ПС и декомпозиции ее по аналитическим срезам «сверху вниз» позволяет добиться целостности описания *интегрированной модели*, избежать проблем обратной интеграции функций отдельных компонент ПС в единое целое.

В общем виде математическое описание динамики состояния ПС можно записать следующим образом:

$$\Delta V = \frac{\partial F}{\partial P} \Delta P + \frac{\partial F}{\partial Y} \Delta Y + \frac{\partial F}{\partial t} \Delta t, \quad (3)$$

где  $\frac{\partial F}{\partial P} \Delta P$  – изменение параметров ПС в результате реализации организационно-технических мероприятий, чаще всего включающих в инвестиционные программы;

$\frac{\partial F}{\partial Y} \Delta Y$  – повышение эффективности использования ресурсов, реализующихся при помощи инвестиционных программ;

$\frac{\partial F}{\partial t} \Delta t$  – изменение состояния ПС в течение времени, в том числе вследствие износа и восстановления системы.

Математическое описание должно быть достаточным для обеспечения пользователя расчетами ретроспективных (включая восстановление недостающих значений показателей), фактических, прогнозных, сценарных (при условии реализации каких-либо событий или мероприятий) и плановых значений.

Предложенная система показателей и математическое описание интегрированной модели формируют требования к информационному моделированию ПС, одним из наиболее перспективных вариантов реализации которых являются методы и технологии информационного и онтологического моделирования, рассматриваемые в других главах данной монографии.

*Информационная модель* должна обеспечивать целостность выполнения расчетов на основе математической модели. В том числе она отвечает за сбор, интеграцию, хранение и предоставление данных для формирования и оценки организационно-технических решений. Вариант схемы взаимосвязи математического и информационного описаний интегрированной модели с расчетно-аналитическим комплексом представлена на рис. 2.8.

## § 6. Последовательность разворачивания модели

Ключевым системообразующим фактором, обеспечивающим целостность любой системы, является полезность результата для внешней системы [19]. Порядок разворачивания интегральной модели также связан с формированием модели, исходя из потребности внешней среды в результате, производимом ПС (подконтрольной системой).

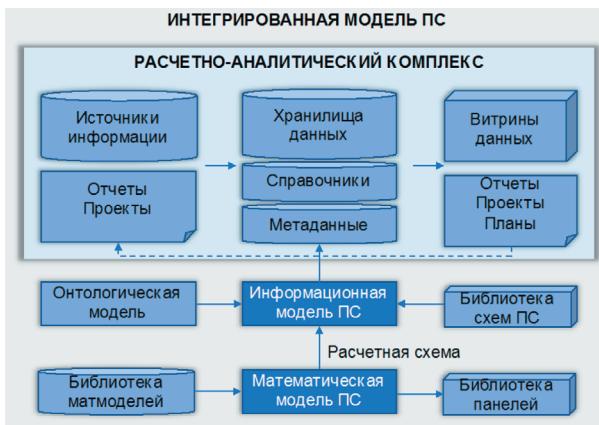


Рис. 2.8. Вариант схемы взаимосвязи компонентов интегрированной модели

В социально-экономических системах объективными факторами, определяющими развитие, является население и его благосостояние. Исходя из этого определяющего фактора, мы предлагаем использовать следующий порядок разворачивания и проведения прогнозно-плановых расчетов по интегрированной модели:

1. **Демографическая и климатические модели.** Исходными данными для разворачивания модели являются статистические наблюдения на территории, рассматриваемой как внешняя среда, с которой взаимодействует ПС.

Результатами расчета являются:

- количество населения, его расселение по территории;
- температура, осадки и прочие климатические характеристики окружающей среды, оказывающие влияние на поведение ПС и его потребителей.

2. **Модель конечного потребления** продукции, производимой ПС. Под конечным потреблением понимается потребление продукции, не используемой для дальнейшего производства.

Исходными данными для данной модели являются результаты по п. 1.

Результатами данного расчета являются ожидаемые объемы потребления основных видов продукции (ресурсов).

**3. Модель спроса, предложения и производственных возможностей.** Исходными данными для данной модели является статистика реализации, производства и запасов продукции, себестоимости производства продукции (п. 4), производственной мощности, инвестиционных планов, прогнозов конечного потребления по п. 2.

Результатами расчета являются:

- равновесные цены на реализацию продукции;
- точки безубыточности;
- области конкурентоспособности;
- предложения в планы продаж.

**4. Балансовые модели.** Комплекс моделей, обеспечивающих количественно определяемые балансы:

- производительных сил (средств производства и персонала);
- производства, распределения и использования валовой продукции;
- производственных нагрузок и мощностей;
- потребления и производства;
- выпуска-затрат (межотраслевые, межтерриториальные балансы);
- денежных ресурсов (бухгалтерские балансы).

Исходными данными для балансовых моделей являются:

- объемы потребления (п. 2);
- равновесные рыночные цены (п. 3);
- структура затрат на производство отдельной продукции.

Результатами расчетов являются:

- дисбалансы и экономические потенциалы в экономической структуре;
- стоимость и объемы производства и потребления продукции по структуре экономической деятельности ПС;
- предложения в производственные программы.

**5. Модели эффективного распределения результатов.**

Оптимизация распределения производства и реализации ассортимента продукции ПС по различным рынкам, с учетом критерия результативности.

Исходные данные:

- структура затрат на продукцию (п. 4);
- объемы и цены реализации на рынках (п. 3).

Результаты расчета:

- корректировки планов по производству, реализации продукции;
- производство и распределение капитала;
- предложения в производственные программы.

## 6. Модель развития производственных мощностей.

Исходными данными являются:

- дисбалансы по экономическим потенциалам (п. 4);
- предельные объемы капиталовложений (п. 5);
- ресурсы природно-антропогенной системы (п. 8).

Результатами расчета являются:

- оптимальные планы по развитию производственных мощностей,
- определение направлений капитальных вложений для реконструкции, строительства, модернизации;
- предложения в инвестиционные программы.

7. Модель рисков. Модель обеспечивает оценку социальных, экологических и технико-экономических рисков – степень вероятности исполнения планов реализации, производства и инвестиций.

Исходными данными для расчета является статистика отклонений в исполнении плановых показателей.

Результатами расчета является степень вероятности достижения планового результата.

8. Комплекс моделей природно-антропогенной системы. Определяет остаточные технические и природные ресурсы, используемые в процессе производства, их распределение по времени и пространству.

Исходными данными являются:

- проектная;
- исполнительская;
- эксплуатационная документация;
- статистика эксплуатации.

Результатами является оценка остаточного ресурса и распределения его по системе.

По глубине используемого математического аппарата можно сформировать следующую последовательность совершенствования точности моделей:

- статические модели, описывающие линейные взаимосвязи показателей ПС;
- динамические модели, описывающие динамику;
- агент-ориентированные модели.

Примером последовательного разворачивания интегрированной модели социально-экономической системы может послужить проект по региональному межотраслевому балансу Санкт-Петербурга, выполненный в сотрудничестве с Комитетом по энергетики и инженерного обеспечения, Комитетом по тарифам, ГУП «Водоканал Санкт-Петербурга» в 2015–2017 годах. С методикой формирования и применения данной модели можно ознакомиться в «Вестнике по тарифам Санкт-Петербурга» за июнь 2018 года [20]. Задачами данного проекта по построению межотраслевому балансу являются учет результативности, оценка, анализ, прогнозирование, оценка управленческих решений, формирование рациональных и согласованных планов развития.

В 2016–2017 годах была разработана первая экономико-математическая модель межотраслевого баланса на примере системы коммунальной инфраструктуры и энергетики г. Санкт-Петербурга (3 % ВРП, 18 % расходов граждан) – ВОБ СКИиЭ. Сформированы сводные прогнозы (за исключением жилищного комплекса) до 2050 года. Проведена оценка финансово-хозяйственной деятельности крупнейших хозяйствующих субъектов отрасли. Выявлен экономический потенциал – 35 млрд руб. в год. В 2018 году снижена совокупная нагрузка на потребителе в размере 10 млрд руб. в годовом исчислении. Проводится автоматизация данного инструмента и детализация модели до отдельных объектов потребителей, производителей, с часовой

дискретностью сбора информации (динамический, агент-ориентированный). Разработаны методические указания, готовится к рассмотрению в правительстве проект постановления по внутриотраслевому балансу системы коммунальной инфраструктуры и энергетики как ядра информационной системы.

В 2018 году начата работа по экономико-математической модели транспортно-экономического баланса в части пассажирских перевозок (13 % ВРП, 23 % от расходов населения, включая выпуск-затраты общественных перевозок и личного транспорта) – ТЭБ. Подготовлены методические документы, собраны балансы за 2012–2017 годы, подготовлен прогноз до 2024 года. Идет работа по описанию и обучению инструменту сотрудников ИОГВ и государственных перевозчиков. В процессе рассмотрения Комитетом по развитию транспортной инфраструктуры – расширение ТЭБ балансом эксплуатации, реконструкции и строительства улично-дорожной сети. Экономический *потенциал*, выявленный по транспортному экономическому балансу, составляет порядка 132 млрд рублей в годовом исчислении и связан с расширением и оптимизацией маршрутной сети, повышением привлекательности для жителей общественного пассажирского транспорта.

В августе 2018 года Комитетом по строительству администрации г. Санкт-Петербурга разработано техническое задание на баланс строительного комплекса, включая концепцию размещения производственных мощностей по строительным материалам. В процессе разработки находится техническое задание на расширение транспортно-экономического баланса по содержанию и развитию транспортной инфраструктуры.

В данной главе обозначены цели формирования доктрины цифровой трансформации по росту результативности и эффективности подконтрольной системы, а также требования к ее отдельным компонентам:

- программе цифровой трансформации;

- моделе-ориентированная системе управления;
- единой программе развития системы.

Предложенные компоненты создают основу для проектирования и постепенного развертывания моделе-ориентированной системы управления, нацеленной на поддержку лиц, принимающих решения, инструментами объективного контроля, оценки, анализа, планирования и программирования.

Архитектура МОСУ, как и интегрированной модели подконтрольной системы, нуждается в дальнейшей конкретизации. Направлениями развития представленной в этой главе концепции являются:

- 1) формирование библиотеки математических моделей социально-экономической динамики, спецификация их входных и выходных показателей и параметров регулирования;
- 2) разработка расчетных схем анализа, прогнозирования и планирования функционирования и развития систем, реализация иных функций управления;
- 3) разработка комплексных систем моделирования, исполнения интегрированных моделей в информационном пространстве;
- 4) создание пользовательских интерфейсов по визуализации динамики систем, их регулирования и управления;
- 5) развитие методов управлению на основе комплексных моделей.

### **Список библиографических ссылок**

1. Development World Commission on Environment and Our Common Future. Oxford University Press, 1987. 383 p.
2. Hooper D. The Top Twelve Grand Challenges Facing Society Today // SWHELPER. 14.01.2016. URL: <https://www.socialworkhelper.com>.
3. Index DJI. URL: <https://www.investing.com/indices/us-30-historical-data>.
4. «Expert Online». Apple, Google, Microsoft, Facebook и Amazon: слишком быстрый рост до \$ 3 трлн. URL: <http://expert.ru/2017/10/3/s3-trln-kapitalizatsii-apple-google-microsoft-facebook-i-amazon-nastorozhili-vlasti-ssha/>.

5. Morozov E., Bria F. Rethinking the Smart City. Democratizing Urban Technology. ROSA LUXEMBURG STIFTUNG, 2018. URL: [http://www.rosalux-nyc.org/wp-content/files\\_mf/morozovandbria\\_eng\\_final55.pdf](http://www.rosalux-nyc.org/wp-content/files_mf/morozovandbria_eng_final55.pdf).
6. Финансирование\_программы\_Цифровая\_экономика // TAdviser. Финансирование национального проекта Цифровая экономика. URL: <http://www.tadviser.ru/index.php/>.
7. Цифровая подстанция. «Россети» представили стратегию построения цифровой сети до 2030 года. URL: <http://digitalsubstation.com/blog/2018/02/15/laquo-rosseti-raquo-predstavili-strategiyu-postroeniya-tsifrovoy-seti-do-nnbsp-2030-goda/>.
8. Изменения и тенденции в регулировании ТЭК России и мира : аналит. центр при Правительстве РФ. III квартал 2018 года.
9. ОПЕК предсказала США лидерство по добыче нефти в мире в ближайшие 5 лет. URL: <https://www.finanz.ru/novosti/birzhevyye-tovary/opek-presdkazala-ssha-liderstvo-po-dobyche-nefti-v-mire-v-blizhayshie-5-let-1007170490>.
10. США нарастили добычу нефти до исторического рекорда. URL: <https://rg.ru/2018/07/19/dobycha-nefti-v-ssha-dostigla-istoricheskogo-maksimuma.html>
11. Гумеров С. Межотраслевой баланс системы коммунальной инфраструктуры и энергетики как инструмент подготовки и оценки государственных и управлеченческих решений в регулируемых сферах деятельности// Неделя компетенций прогнозирования и планирования–2018. : конференция. СПб. : «ПЕРО», 2018.
12. Россия в марте уступила мировое лидерство по добыче нефти США // Рос. информ. агентство. URL: <https://ria.ru/economy/20180517/1520778523.html>.
13. Рост цен на бензин: кто виноват и что делать // Рос. информ. агентство. URL: <https://ria.ru/economy/20180530/1521649535.html>
14. В Петербурге тарифы на тепло могут вырасти на 40 % // Деловой Петербург. URL: [https://www.dp.ru/a/2015/06/25/Tarifi\\_nagrejut\\_fizicheski/](https://www.dp.ru/a/2015/06/25/Tarifi_nagrejut_fizicheski/)
15. Реальные доходы россиян ускорили падение // РБК. URL: <https://www.rbc.ru/economics/17/10/2018/5bc73c559a7947517996d7d0>.
16. Доля нефтегазовых доходов бюджета России в первом полугодии составила 45,6 % // ИА RNS. URL: <https://rns.online/energy/Dolya-neftegazovih-dohodov-byudzheta-Rossii-v-pervom-polugodii-sostavila-456-2018-08-13/>.

- 
- 
17. Глушков В. М., Валах В. Я. Что такое ОГАС. М : Наука, 1981. 160 с.
  18. Аладинский В. В., Григорьева Ю. Б. Мониторинг объектов магистрального нефтепроводного транспорта // Наука и технологии трубопроводного транспорта нефти и нефтепродуктов. 2011. № 1. С. 16–21.
  19. Анохин П. К. Узловые вопросы теории функциональной системы. М. : Наука, 1980. 195 с.
  20. Временная методика формирования и применения внутриотраслевого баланса системы коммунальной инфраструктуры и энергетики Санкт-Петербурга // Вестн. комитета по тарифам Санкт-Петербурга. 2018. № 06 от 29 июня. С. 10–74.

## Глава 3

# ПРАКТИЧЕСКИЕ ВОПРОСЫ КОНЦЕПТУАЛЬНОГО МОДЕЛИРОВАНИЯ ПРЕДМЕТНЫХ ОБЛАСТЕЙ

*М. Г. Мирошниченко*

В этой главе мы рассмотрим ряд вопросов, возникающих перед аналитиком в процессе построения модели предприятия. Глава, не претендуя на глубокий анализ, касается вопросов восприятия и познания, которые оказывают существенное влияние на качество и практическую пригодность создаваемых моделей.

### **§ 1. Что такое модель и моделирование?**

Термин «концептуальное моделирование» предполагает построение модели при помощи концептов, имеющих определенное значение для всех участников создания и использования модели. Формулирование строгих определений концептов представляет значительную трудность, если аналитик начинает рассматривать различные аспекты использования этих концептов в языке и точки зрения на них. Сложнее всего формулировать определения для терминов высокого уровня абстракции, которые, как правило, используются в обычном языке сразу в нескольких значениях, существенно отличающихся в зависимости от контекста. В качестве примера попробуем дать определение термина «модель».

В обыденном представлении под моделью принято понимать некий артефакт, существующий в реальности, изучая который можно сделать выводы о свойствах другой части реальности. «Изучение реальности» подразумевает формирование в сознании некоторых представлений о ней, которые, разумеется, являются лишь приблизительным описанием определенных аспектов реальности и не тождественны ей. Таким образом, наши представления о реальности также являются своего рода моделью, выполняя над

которой мысленные операции, мы делаем выводы о последствиях развития событий в реальном мире.

Является ли модель, создаваемая аналитиком в результате моделирования и записанная в формальном виде, точным отражением представлений о реальности, сформировавшихся у него в процессе познания? Разумеется, нет – формальная модель также является лишь неким упрощенным, частичным описанием таких представлений. Эти рассуждения демонстрируют, насколько сложно дать формальное определение термину «модель». Вместе с тем его значение вполне понятно читателю, а использование одного слова для обозначения принципиально разных по своей сути моделей не вызывает проблем при восприятии текста.

Чтобы проиллюстрировать субъективность наших представлений о реальности, проведем мысленный эксперимент. Пусть существуют два физических объекта – две лопаты: лопата № 1 и лопата № 2. Сделаем обмен: черенок одной лопаты приладим к штыку второй, а штык второй насадим на черенок первой. Вопрос: после обмена мы получили две новых лопаты или же это две старых лопаты, у которых изменился состав? Если предположить, что смена состава физического объекта обязательно приводить к смене идентичности физического объекта в целом, то можно привести контраргумент: бьющий фонтан – это физический объект, но вода в нем постоянно меняется. С другой стороны, в некоторых случаях смена состава приводит к появлению новых свойств, что позволяет нам говорить о том, что старый объект закончил свое существование и возник новый. Например, если вместо воды из фонтана пойдет нефть, могут появиться основания для того, чтобы рассматривать этот фонтан как новый объект. Можно, однако, усомниться в этом и сказать, что фонтан остался прежним, но изменились его свойства. В итоге можно прийти к выводу о том, что не существует никакого «объективного» способа разрешить этот спор.

Решение о том, старый это объект или новый, принимается исключительно субъективно и зависит от pragmatики, то есть от того, как субъект намерен использовать объект и/или его модель.

Разумеется, у разных субъектов могут быть разные точки зрения на одни или те же ситуации, в которых они, соответственно, будут по-разному выделять и рассматривать объекты.

Вернемся к мысленному эксперименту с лопатами. Предположим, что принято решение считать, что лопаты как физические объекты продолжили свое существование, но в новом составе. Рассмотрим теперь задачу идентификации объектов: где из них какая? Одна лопата стоит в сарае, вторая – в амбаре, но какая из них лопата № 1? Тот же вопрос можно сформулировать иначе: лопаты обменялись черенками или штыками? Что первично в составе лопаты – черенок или штык? Если первичен штык и лопаты обменялись черенками, то лопата № 1 находится в сарае, если штыками – то в амбаре. Здесь снова нет никакого способа сделать «правильный» выбор. Любое решение «правильное» и зависит от намерений и представлений субъекта, контекста, в котором рассматривается вопрос.

Таким образом, два субъекта, даже договорившись о том, что сейчас они видят перед собой один и тот же объект, пользуясь одними и теми же принципами построения выводов, через некоторое время с большой вероятностью разойдутся в представлениях и обнаружат этот объект в разных местах пространства, при этом у них не будет «объективного» критерия для разрешения спора о том, где же на самом деле находится объект и тот ли это объект.

Приведенный выше пример объясняет, почему нельзя претендовать на «объективность» описания реальности при создании моделей. Мы описываем не реальность, а наше представление о ней. Помимо представлений о реальности, в нашем сознании могут находиться представления о несуществующих объектах (например, о единорогах). Представления формируются в процессе восприятия, который преобразует (интерпретирует) поступающую информацию, в результате чего человек, например, вместо пятен на бумаге может увидеть изображение курительной трубки. Все зависит от того, какой парадигмой пользуется человек в момент интерпретации сигналов, поступающих к нему снаружи.

Что же тогда мы называем моделью? Для передачи представлений от одного субъекту другому люди используют сигнальные системы. Сигналы звуковые, световые импульсы, вещественные объекты и так далее. Восприятие этих сигналов принимающим субъектом проходит через фильтр его представлений и формирует в сознании образы, заведомо не тождественные тем, которые существуют в сознании передающего субъекта.

Для того чтобы процесс передачи информации приводил к практическим результатам, то есть разные люди могли договориться между собой, у них должны существовать похожие наборы представлений. Такие наборы представлений о реальности принято называть парадигмами. Границы и состав парадигм условны и изменяются со временем, однако их развитие можно проследить и зафиксировать.

Например, можно рассмотреть парадигму панпсихизма, в которой объекты обладают «душой» [1]. В такой парадигме вопрос о лопатах лишен смысла, потому что достаточно спросить: сохранились ли души в лопатах, и если сохранились, то где именно? В этой парадигме идентификация объекта очевидна. Эта парадигма не так оторвана от современной реальности, как может показаться: например, при учете материальных ценностей на предприятиях используется присвоение объектам инвентарных номеров. Объект остается самим собой до тех пор, пока не изменилась та его «основная» часть, на которой нанесен инвентарный номер.

В парадигме «одушевленного мира» естественным является представление о том, что все изменения, которые происходят в природе, выполняются кем-то. Эта парадигма нашла свое отражение в нашем языке в выделении в структуре высказывания подлежащего (актора) и сказуемого – его действия. В итоге мы ищем актора в каждом высказывании, порой наделяя субъектностью неодушевленные объекты: *солнце светит, машина едет, предприятие производит товары*. Такая парадигма приводит к невозможности однозначно ответить на вопрос: кто/что производит точение детали – токарь, станок, механический блок без станины, или резец? «Правильного» ответа на этот вопрос нет и быть не может.

Еще одна парадигма, пришедшая к нам из древности, – это парадигма цикличного времени. Древние люди считали, что охота на бизонов вчера и охота на бизонов сегодня – одно и то же действие, а не два действия, похожих друг на друга. Сегодня мы пользуемся парадигмой линейного времени, в которой действие вчера не тождественно действию сегодня. Однако нам до сих пор кажется, будто действие можно повторить. Это может привести к формированию представления (даже не осознанного!) о том, что повторяющиеся конкретные действия идентичны. Такое представление поощряется средствами моделирования бизнес-процессов, где не рассматриваются различия между типовыми и конкретными операциями. В результате возникают сложности при переходе от абстрактной модели типовых бизнес-процессов к моделям конкретных бизнес-операций, например в контексте создания учетной системы.

Таким образом, при моделировании предприятия аналитик, с одной стороны, сталкивается с противоречиями, которые есть в каждой из парадигм, используемых им самим или другими аналитиками, а с другой – вынужден стыковать разные парадигмы между собой. Чтобы научиться это делать, необходимо сделать шаг назад и рассмотреть атомарные элементы – первичные концепты, на которые опирается каждая парадигма. Затем нужно понять, как на основе этих первичных концептов конструируются различные парадигмы.

## § 2. Представления о пространстве и времени

В мышлении мы пользуемся представлениями, которые описывают наш эмпирический опыт. Представления можно разделить на группы: пространственные, временные и пространственно-временные. Пространственные представления определяют наше представление о пространствах (не о пространстве), временные представления определяют наше представление о временах (не о времени), пространственно-временные формируют наше представление о пространственно-временных объемах (конечно, мы не можем говорить об абсолютном пространстве или времени).

Рассмотрим ряд основополагающих концептов, описывающих такие представления.

### ***Пространственные концепты***

1. Пространственная среда – это пространство, наблюдаемое изнутри, внутри которого мы представляем себя как наблюдателя.

Пример: пространство, из которого производится построение моделей: цех, если мы моделируем станки, или город, если мы моделируем территорию предприятия.

2. Объект – это пространство, наблюдаемое снаружи, внутреннее строение которого мы не рассматриваем в контексте нашей задачи.

Пример: станок – в контексте учета единиц оборудования, помещение – в контексте учета производственных помещений.

3. Пространственная композиция – несколько пространств, наблюдаемых снаружи, каждое из которых мы воспринимаем и осознаем индивидуально. Эти пространства могут трактоваться как объекты, множества или субстанции, они могут быть одного типа или разных типов, но суть такого рода восприятия в том, что мы одновременно осознаем их индивидуальность и взаимосвязанность.

Пример: конструкция моста в контексте проектирования моста, конструкция здания в контексте проектирования здания.

Исходя из ограничений нашего восприятия, количество таких пространств обычно не превышает 5–10. Если пространств становится больше, наше сознание склонно перестать осознавать их индивидуальность и обобщить до множества.

4. Пространственное множество. Когда количество осознаваемых пространств становится слишком большим для их одновременного индивидуального восприятия, но сознание может их дифференцировать, вместо наблюдения отдельных индивидуальных пространств сознание переключается на рассмотрение совокупности пространств одного типа.

Пример: гайки в контексте учета метизов, морские контейнеры в контексте расчета распределения масс при их погрузке на судно.

Если количество пространств продолжает увеличиваться, сознание в какой-то момент перестает их дифференцировать и переходит на восприятие сплошной субстанции.

5. Пространственная субстанция. Невозможность распознать в среде дифференцированные пространства приводит нас к понятию пространственной субстанции.

Пример: дизельное топливо в контексте учета хранимых нефтепродуктов, выход готового продукта в контексте анализа работы ректификационной колонны.

### *Временные концепты*

Концепты для времени можно определить аналогично:

1. Временна́я среда – это время, осознаваемое нами изнутри, внутри которого мы рассматриваем его элементы – под-времена.

Пример: время, в рамках которого рассматривается функционирование предприятия; время, в рамках которого рассматривается операция изготовления одного болта.

2. Событие, происшествие, операция – время между двумя состояниями, в течение которого происходит изменение в наблюдаемом пространстве. Обычно мы рассматриваем состояние до и после события, но не рассматриваем его внутреннюю структуру, так как оно не существенно для целей создания модели.

Пример: операция по изготовлению одного болта. Известно, что сначала была заготовка; прошло время – появился болт и стружка; что конкретно происходило в это время – мы не рассматриваем.

3. Временна́я композиция. Несколько времен, осознаваемых нами как индивидуальные. В качестве времен, составляющих композицию, могут выступать события, множества событий, временные субстанции. Пример модели временной композиции можно увидеть на диаграмме Ганта, где представлены модели операций.

4. Множество событий. Когда событий становится слишком много, чтобы распознать их индивидуальность, мы приходим к восприятию множества неиндивидуальных событий.

Пример: регулярная отправка грузов, регулярный прием заявок.

5. Временная субстанция. Если наши органы чувств перестают различать отдельные события, мы рассматриваем непрерывные изменения.

Пример: функция перекачки нефти, функция перемещения груза и т. д.

### ***Пространственно-временные концепты***

Чисто пространственные и чисто временные представления являются абстракциями, полезными для анализа процесса мышления. Пространство не существует вне времени и время не существует вне пространства, поэтому практически при создании моделей мы моделируем не пространство и время в отдельности, а пространственно-временные объемы (ПВО) [2]. Если наше сознание концентрируется на свойствах пространственно-временных объемов, однородных в пространстве и непрерывных во времени, мы склонны говорить о пространствах, а если на свойствах, однородных во времени и непрерывных в пространстве – то о временах.

## **§ 3. Моделирование свойств пространственно-временных объемов**

При моделировании предприятия пространственно-временные концепты необходимо использовать осознанно. Это значит, что нужно осознавать размеры моделируемых пространственно-временных объемов и размеры пространственно-временных однородностей, определяющих свойства ПВО. Например, когда моделируется функция выпуска машин, размеры моделируемого пространственно-временного объема сравнимы с размером предприятия длительностью несколько лет. При этом функция «выпуск машин» не может быть определена в интервале времени длительностью десять секунд, поскольку на этом времени невозможно описать операции, выполняемые в рамках данной функции. Минимальное время, в рамках которого имеет смысл говорить о функции «выпуск машин», – несколько дней. Эта длительность

определяет *время однородности* моделируемого свойства пространственно-временного объема, а именно – функции выпуска машин<sup>1</sup>. Таким образом, свойство «выпуск машин» определено в пространственно-временной области масштабом несколько километров на несколько лет, и может иметь (в зависимости от уровня детализации моделирования) пространственно-временные однородности размером несколько микрон на несколько дней.

В качестве еще одного примера рассмотрим физический объект – трансформатор. Достаточно одной миллисекунды, чтобы определить свойство пространственно-временного объема, воспринимаемого как физическое тело. Но сколько времени нужно, чтобы определить другое свойство – функцию трансформации напряжения? Если речь идет о частотах порядка 50 Герц, то на определение этого свойства потребуется около секунды! То есть говорить о трансформации напряжения частотой 50 Герц на промежутке времени в одну миллисекунду не имеет никакого смысла.

Обычно о такого рода деталях не задумываются, потому что масштаб времени и пространств, для которых строятся модели предприятия, сравним с масштабами человеческого тела и скоростью восприятия человеческого сознания. Все, что выходит за пределы этих границ, человек неосознанно воспринимает как нечто, что требует иного способа моделирования. Например, алгоритм, построенный в нотации BPMN [3], аналитик называет процессом, а такой же алгоритм, написанный на языке программирования, он называет программой. Единственная разница между ними в том, что при обработке программного кода движок создает инструкции на основе программного кода, а затем их выполняет, а при обработке диаграммы в нотации BPMN движок лишь создает инструкции,

---

<sup>1</sup> Рассмотрение функции как свойства пространственно-временного объема может показаться неочевидным, прежде всего из-за неоднозначности самого понятия «функции». Одно из значений этого понятия в контексте моделирования предприятий состоит в том, что «функцией» называется способность определенного пространственно-временного объема (например, производственной линии) выполнять некоторые преобразования других пространственно-временных объемов (например, сырья, продукции). В этом смысле функция рассматривается как свойство пространственно-временного объема.

но выполняют их люди. На текущий момент существует множество разных течений и направлений моделирования, каждое из которых имеет свой индивидуальный понятийный аппарат, слабо или вообще не связанный с понятийным аппаратом другого направления. Любая попытка объединить различные направления с целью создания единого понятийного аппарата обычно приводит лишь к созданию новых направлений. В связи с этим в рамках данной главы мы стараемся избегать таких популярных терминов, как процесс, система, архитектура. Эти термины имеют множество значений, некоторые из которых приняты в рамках определенных частных методов или технологий моделирования, и дифференцировать эти значения достаточно сложно.

Одним из правил моделирования является соотнесение размеров пространственно-временных объемов для всех элементов моделируемой области. Неосознанность в определении пространственно-го или временного объема часто приводит к ошибкам. Например, рассмотрим функциональную модель предприятия. В качестве элементов этой модели выступают функции, связанные между собой материальными или информационными потоками. Если моделируемые функции будут иметь несогласованные между собой пространственные или временные масштабы, могут возникнуть противоречия. Представим диаграмму в нотации IDEF0 [4], на которой одновременно смоделированы функция выпуска машин и функция перепрофилирования производства. Это – очевидная ошибка, потому что длительность функции выпуска машин меньше времени, на котором определена функция перепрофилирования производства.

Обычно в своей практике мы не сталкиваемся с подобного рода противоречиями, потому что каждый аналитик решает очень узкую и специфическую задачу, не рассматривая ее в контексте более широких или более узких задач. Иными словами, масштаб, на котором производится моделирование, для каждого специалиста достаточно узок.

Но поскольку создание онтологических моделей часто претендует на создание единой модели всего предприятия, аналитик

сталкивается с задачей перехода от одних масштабов к другим: от несколько десятков лет до наносекунд, от нескольких десятков километров до нанометров. Понятно, что шагать по этим масштабам как в сторону детализации, так и в сторону интеграции надо осознанно. Например, когда аналитик собирает статистику по операциям типа «выточить болт», какому свойству какого пространственно-временного объема будет соответствовать полученный результат? Функции «точение болтов», пространственный объем которой равен объему цеха, а временной равен времени сбора статистики. Функцию «точение болтов» можно рассматривать как часть функции «изготовление болтовых соединений». А функцию «изготовление болтовых соединений» можно считать частью конструкции операции «сборка пролета моста». В этом рассуждении происходит переход от масштаба одной операции «выточить болт», длительность которой две секунды и пространственный объем – одно рабочее место, до операции «сборка пролета моста», длительность которой равна месяцу, а пространственный объем – строительной площадке. Под осознанностью такого перехода понимается знание того, что функция может быть представлена в виде множества операций одного или нескольких типов. Кроме того, был сделан сознательный выбор конкретного способа композиции операций из огромного числа возможных, но при этом другие возможности композиции не остались незамеченными и при необходимости могут быть смоделированы.

Движение по «лестнице» масштаба вверх или вниз связано с детализацией наших представлений и их интеграцией. Часто в применении к этому движению можно услышать термины «композиция» и «декомпозиция», причем чаще второе, нежели первое. Современные стандарты моделирования подробно описывают процесс декомпозиции, но избегают рассмотрения композиции, и понятно почему. Например, при декомпозиции кажется очевидным, что буровая установка состоит из силового привода, насосного оборудования, системы для выполнения спуско-подъемных работ и так далее, но совершенно неочевидно, частью чего является она сама.

Она может быть частью предприятия, передвижного комплекса, музея. При необходимости сделать декомпозицию аналитик быстро находит «единственно верное» решение, потому что декомпозиция моделируется часто, и относительно нее существует «договоренность» (типовой способ решения задачи). При необходимости выполнить композицию аналитик теряется, потому что вопросу композиции практически не уделяется внимания, и относительно нее не существует никаких договоренностей. На самом деле как декомпозиция, так и композиция могут быть выполнены бесчисленным количеством способов. Например, буровую вышку можно декомпозировать как на агрегатные блоки, так и на функциональные системы. Существенным ограничением является то, что в современных стандартах моделирования альтернативные способы декомпозиции не рассматриваются. Это наталкивает аналитиков на мысль, что существует «единственно верный» способ декомпозиции.

Относительно недавно непонимание того, что одну и ту же бизнес-функцию можно разделить на под-функции разными способами, привело к созданию нового направления в моделировании – появился процессный подход<sup>2</sup>. До процессного подхода существовало мнение, будто функцию предприятия следует делить на под-функции в соответствии со специализацией отделов. Но никто не предполагал, что ту же функцию можно разделить и другими способами, например в соответствии с производимыми товарами или услугами. Применяя аналогию с буровой вышкой, можно сказать, что это равносильно тому, чтобы буровую вышку всегда делили на системы, пока не догадались, что ее можно поделить иначе: на агрегатные блоки. Открытием процессного подхода мог бы стать тот факт, что функцию можно поделить на части бесчисленным количеством способов, каждый из которых удобен для решения определенного класса задач. Вместо этого адепты процессного подхода назвали процессом один из способов функционального деления и критикуют другие, хотя все способы деления совершенно равноценны.

<sup>2</sup> См. описание процессного подхода: URL: <https://www.cfin.ru/itm/bpr/>.

Композиция и декомпозиция могут связывать представления разных типов. Рассмотрение всех возможных случаев выходит за рамки данной главы. Остановимся лишь на одном из них: покажем, почему функцию можно разложить на операции одного типа.

Бизнес-функция – это такая трактовка ПВО, которая описывает преобразование потоков внутри этого ПВО. Например, функция под названием «точение болтов» – это ПВО, входом которого является поток заготовок, а выходом – поток болтов. Мы рассматриваем потоки (поток заготовок и поток болтов), входящие и покидающие ПВО, трактуемый как функция, рассматриваем этот объем (цех), но не рассматриваем то, что находится внутри.

Если сравнить это описание с описанием деревянного шара, то описание функции соответствует описанию поверхности деревянного шара: мы наблюдаем как рассеивается в стороны свет, падающий на поверхность шара, но не наблюдаем, что находится под этой поверхностью. Если нужно объяснить строение шара, мы говорим, что он деревянный. Если требуется описать строение бизнес-функции, мы говорим, что внутри функции происходит точение болтов. Но «деревянный» и «точение болтов» – это названия структуры, но не ее описания. Если необходимо описать структуру шара, нам придется (в предельном случае) описать каждое волокно и его положение в пространстве. Создать такую модель практически невозможно. Однако можно создать модель его типовой структуры деревянного шара. Эта модель не будет описывать каждое волокно, но даст представление о типовом волокне и типовом расположении волокон относительно друг друга. Таким же способом можно описать структуру функции: можно не строить модель для каждой конкретной операции, но можно построить модель типовой операции точения. С точки зрения четырехмерного объема модель типового волокна и модель типовой операции аналогичны. Одно описание представляет из себя описание пространственной структуры, второе – описание временнóй. Деревянный – свойство, однородное в пространстве. Точение болтов – свойство, однородное во времени.

Таким образом, структура бизнес-функции «точение болтов» – это свойство ПВО, которое однородно во времени, структурными элементами которого являются операции типа «точение болта».

Понимание того, что такое функция и как она соотносится с операциями, позволяет точнее понять смысл ГОСТ 34, в котором описаны требования к техническому заданию на автоматизацию. В этом ГОСТе под предметом автоматизации понимается функция. Поэтому, следуя ГОСТ 34, пишут так: объектом автоматизации является функция расчета потребленного газа. Мы знаем, что функция состоит из операций одного типа. Это значит, что функция расчета потребленного газа состоит из операций по расчету потребленного газа. Далее мы говорим, что каждая операция по расчету потребленного газа вызывается пользователем информационной системы нажатием на кнопку «рассчитать потребление газа». Читатель хорошо понимает, что именно хочет заказчик от информационной системы. Однако на практике можно видеть описания, в которых говорится, что объектом автоматизации является процесс расчета потребленного газа, а далее в качестве описания процесса приводится диаграмма в нотации BPMN. О том, что такое процесс и как он связан с диаграммой, до сих пор идут споры. Чтобы избежать подобных споров, стоит пользоваться ГОСТ 34 и трактовкой понятия функции, приведенной выше.

## **§ 4. Проблемы концептуального описания предметных областей**

Наше представление о реальности складывается из представлений, каждому из которых соответствует определенный концепт, существующий в нашем сознании. Мы способны представить себе лишь то, для чего у нас есть определенный концепт. Базовый набор концептов мы приобретаем в детстве и обычно считаем, что он постоянен. Каждый концепт – это результат обобщения эмпирического опыта, закрепленный в шаблоне, который находится в нашем сознании. Мы можем создавать воображаемые миры

в своем сознании, опираясь только на эти шаблоны. Часть из этих миров в той или иной мере соответствовать реальности, а часть может быть практически оторвана от нее. Каждый субъект конструирует свои индивидуальные воображаемые миры, связанные с реальностью.

Картина мира, сформированная в сознании человека, строится при помощи концептов, одни из которых означают типы объектов («стол»), другие – являются маркерами конкретных объектов («Иван Иванович»). Так или иначе, в структуре концепта можно выделить знак (слово «стол») и означаемое (множество всех столов). Возможность передачи информации между людьми основана на том, что восприятие вербализованных концептов формирует в их сознании схожие образы: услышав слово «стол», каждый из нас представит что-то свое, но между этими образами, несомненно, будет много общего, по крайней мере с pragматической точки зрения. Вопросы отношений между знаком и означаемым, использовании знаковых систем для передачи информации между людьми детально изучены в семиотике (см., напр., [5]). Здесь мы затрагиваем их в минимальном объеме, необходимом для постановки и решения некоторых типовых задач, возникающих перед аналитиками.

С другой стороны, в некоторых случаях различия в таких представлениях могут оказаться существенными и требовать уточнения. В таких случаях процесс передачи информации продолжается с использованием дополнительных понятий, которые позволяют дифференцировать (разделить) похожие представления.

Предположим, что один субъект намерен передать другому сведения о размере некоторого объекта. Для этого он использует шкалу, состоящую из трех концептов: «больше меня», «такой же как я» и «меньше меня». Такая шкала позволяет разделить все возможные объекты на три типа в зависимости от их пространственного размера по отношению к физическим размерам говорящего. Однако точности такой шкалы может оказаться недостаточно для того, чтобы второй субъект получил практически значимую информацию

о размере объекта. Тогда первый субъект вместо слишком общей категории «меньше меня» может использовать дифференцированную шкалу: «с кошку», «с крысу», «с муравья». Чем выше требования к дифференциации представлений, тем больше требуется типов, чтобы их дифференцировать. Если типов станет слишком много, субъекты введут шкалу, использующую множество с известными свойствами, например множество натуральных или действительных чисел. Когда информация о размере объекта будет выражена через такую шкалу (например, «10 метров»), ее сложно будет интерпретировать как отнесение объекта к типу (конечно, можно представить себе множество всех объектов размером 10 метров, но такая типизация не удобна для нашего сознания). В терминологии, используемой в информационных технологиях, представленная таким образом информация выражена *значением атрибута* объекта.

Несмотря на легкость перехода от описания типов объектов к описанию атрибутов, необходимо помнить: атрибут никак не связан с типом объектов, и это кажется странным, потому что наш опыт моделирования говорит об обратном. Например, создав в базе данных таблицу «деревья», моделирующую тип объектов «деревья», аналитик создает атрибут «высота» и считает, что этот атрибут принадлежит этому типу. На это наталкивает его структура реляционной БД, в которой понятие «тип» связано с таблицей, а понятие «атрибут» связано с назначением колонки в этой таблице. Между тем понятно, что высотой могут характеризоваться не только деревья, значение высоты может быть представлено при помощи различных шкал и др., то есть на уровне концептуальной модели говорить о связи атрибутов и типов сущностей можно только в отношении того, что объекты определенных типов могут характеризоваться значениями определенных атрибутов.

При помощи типов и атрибутов можно конструировать достаточно широкий набор высказываний о конкретных объектах и их множествах. Кажется, что создать модель в виде набора такого рода высказываний довольно просто. Построив их, мы создадим

описание предприятия, и работа по моделированию будет завершена. Однако на практике возникает ряд сложностей, некоторые из которых мы рассмотрим.

### **Множественность трактовок одного высказывания**

Рассмотрим высказывание *Все автомобили в этом гараже – белого цвета*. Оно может трактоваться как простое высказывание о типе объектов (все автомобили, входящие в множество находящихся в этом гараже, входят и во множество объектов белого цвета) или как утверждение о том, что все объекты данного множества обладают общим значением атрибута «цвет». При трансляции такого высказывания в модель OWL аналитику необходимо выбрать один из этих способов, такой выбор должен быть осознанным, а критерии выбора должны входить в состав методики, принятой при создании данной модели.

### **Различие терминологии онтологического моделирования и объектно-ориентированного программирования**

Создание автоматизированных систем, использующих онтологические модели, подразумевает совместную работу аналитиков и программистов. Среди аналитиков также достаточно много тех, кто знаком с языком UML и привык к его терминологии, которая ориентирована на моделирование программного кода, написанного в парадигме объектно-ориентированного программирования (ООП). Часто приходится слышать, будто при помощи UML можно моделировать предметные области, отличные от кода, и аналитики пытаются применить его принципы при создании онтологических моделей.

Однако смысл основных терминов, таких как «объект» и «класс», в ООП и онтологическом моделировании совершенно различен. Например, в парадигме ООП является нормальным высказывание: «экземпляр данного объекта». Под «объектом» здесь понимается некий шаблон, содержащий описание структуры данных об объекте и методов взаимодействия с ним, который может

быть воплощен в виде «экземпляра объекта». Если «экземпляр объекта» в ООП является моделью объекта реального мира, то что тогда моделирует «объект»? Получается, что «объект» ООП моделирует тип объектов предметной области. Различное понимание одних и тех же терминов между разными парадигмами затрудняет взаимодействие между аналитиками, и особенно между аналитиками и разработчиками.

### **Связь атрибутов и типов**

Рассмотренное выше заблуждение о том, что атрибут принадлежит типу объектов, приводит к дублированию атрибутов в информационных моделях. Например, создадим таблицу «машины» с атрибутом «высота», затем таблицу «здания», в которой создадим одноименный атрибут «высота». Вопрос: это тот же атрибут или другой? Парадигмы ООП и реляционных баз данных запутывают аналитика в этом вопросе, заставляя думать, что высота машины и высота здания – это разные атрибуты. Парадигма онтологического моделирования позволяет моделировать атрибуты (свойства) отдельно от типов, что дает возможность описывать высоту любых объектов при помощи одного и того же атрибута (свойства).

### **Тип или свойство?**

Рассмотрим высказывание *Белая машина*. Можно представить себе образ машины с приkleенным к нему стикером, на котором написано название значения свойства: *белый*. Однако, как было сказано выше, на концептуальном уровне высказывание о свойстве объекта относительно дискретной шкалы принципиально не отличается от высказывания о его типе. Указанное выше высказывание можно представить как два высказывания о типах, первое из которых классифицирует выбранный четырехмерный пространственно-временной объем как машину, а второе – классифицирует поверхность этого объема как белую. Соединив две точки зрения, мы получаем, что, с одной стороны, выбранный объем можно классифицировать как машину, с другой – как

объем, имеющий белую поверхность. Но белая поверхность при этом не является свойством машины, как и машина не является свойством белой поверхности. Машина со стикером на борту – это образ, который заставляет нас думать, что один из методов классификации ПВО является более значимым, чем другой. Непонимание этого приводит к противоречиям, с которыми знакомы все аналитики, работавшие со структурой реляционных баз данных: при доработке модели оказывается, что бывшее значение атрибута теперь надо выделить в отдельный тип объектов, а то, что раньше моделировалось полем в таблице, теперь надо выделить в отдельную таблицу. Свойство «белая поверхность» принадлежит не машине, а ПВО, как и свойство «машина». Такой подход к моделированию дает нам два независимых утверждения, не приводящие к противоречиям при дальнейшей доработке модели.

### **«Такой же» иногда значит «похожий»**

Часто два свойства двух разных ПВО одного типа моделируются не как похожие, а как идентичные свойства. С одной стороны, два похожих велосипеда легко дифференцируются как два разных свойства двух разных ПВО, и потому моделируются разными объектами в информационной модели с разными идентификаторами. Но с двумя похожими операциями уже не все так однозначно. Можно встретить модели, в которых для двух похожих операций созданы отдельные объекты с индивидуальными идентификаторами, а можно и те, в которых похожие операции моделируются как одна тождественная операция с одним идентификатором. Если же речь идет о свойствах ПВО, описанных прилагательными, например «белый», то здесь решение однозначное: в информационной модели вы никогда не увидите двух разных «белых» с двумя различными идентификаторами, потому что кажется, что это не похожие, а идентичные свойства. Является ли это неточностью, зависит от pragmatики использования модели.

### **Активность и деятельность**

Описание изменений при помощи парадигмы деятельности предполагает наличие актора, цели, инструмента, объекта деятельности, акта деятельности, результата деятельности. Весь этот набор скопирован с нашего языка, в котором существуют подлежащее, сказуемое, дополнение. Мы уже давно не одушевляем солнце или машину, но пользуемся языком, в котором для каждого изменения в мире должен быть найден актор, ответственный за это изменение. Например, если речь идет о точении болта, то должен быть найден актор, который его точит, даже если это автоматизированная линия. Такой подход порождает неоднозначность. Кто точит болт: автоматизированная линия, станок, механический блок или резец?

Существует другая парадигма моделирования изменений, в которой нет необходимости искать несуществующего актора. Она говорит о том, что изменения происходят не потому, что их кто-то совершает, а потому, что так устроен этот мир. Это – парадигма активности. Если, пользуясь парадигмой деятельности, мы ищем актора, инструмент, объект деятельности и результат, то, пользуясь парадигмой активности, мы забываем об этих ролях, и остаются только участники. Моделируя деятельность, мы говорим: для того чтобы купить продукты, надо поехать в магазин на машине. Моделируя активность, мы говорим: частью операции по покупке продуктов является поездка в магазин или покупка продуктов предшествует поездка в магазин. Моделирование активности позволяет моделировать изменения, заменяя субъективные причинно-следственные связи на более нейтральные связи «часть–целое» и «предшествует–следует». Из модели активности всегда можно построить модель деятельности, добавив поверх модели активности модель ролей. При этом модель активности позволяет строить столько ролевых моделей, сколько есть мнений насчет того, кто или что выполняет ту или иную операцию.

Когда мы говорим, что Земля вращается вокруг Солнца, мы не задаем вопрос: кто ее вращает? Когда же мы видим, как точится болт, нам надо обязательно найти того актора, который его точит.

Нам в голову не приходит мысль, что болт точится потому же, почему вращается Земля: потому что так устроен этот мир. Мы можем выделить участников вращения Земли: Солнце, среду, Землю, гравитацию. Но невозможно сказать, что Солнце при помощи гравитации вращает Землю вокруг себя. Участниками точения болта являются станок, электрическое поле, Земля, среда и так далее. В равной степени можно сказать, что актором является станок, резец, программа и так далее, но все это не более, чем ролевая модель, построенная поверх модели активности.

На текущий момент аналитики предпочитают строить ролевые модели без построения модели активности. Аргументом является тезис о том, что якобы модель активности не отвечает на вопрос зачем она нужна, а модель деятельности отвечает. Это как если бы чертеж изгиба трубы должен был бы объяснить, зачем ее так изогнули, например для предотвращения завихрений в потоке топлива при его подаче в камеру сгорания. Как чертеж трубы не объясняет причину ее изгиба, так и модель активности не объясняет ее назначение. Однако как чертеж трубы нужен, чтобы ее изготовить, так и модель активности нужна, чтобы ее выполнить.

### **Высказывания о множествах перечисляемых, но не идентифицируемых объектов**

В модели предприятия часто приходится строить высказывания, например, следующего типа: *исполнителями данной операции по погрузке данного груза будут двое из шести грузчиков, числящиеся в составе подразделения*. Такого рода высказывания часто возникают в модели предприятия, но в информационных системах есть большая проблема с их моделированием. Способ решения этой задачи в онтологическом моделировании рассматривается в одной из следующих глав.

### **Точность и вероятность**

Строя модели предприятия, можно задать вопрос: какое время мы моделируем: прошлое, настоящее или будущее? Поскольку

настоящее моделировать невозможно (это не более чем мгновение), все наши высказывания относятся либо к будущему, либо к прошлому. Но будущее нам известно с какой-то вероятностью, а прошлое – с какой-то точностью. Поэтому при моделировании предприятия приходится делать высказывания вероятностного характера относительно класса объектов, а не относительно объектов данного класса: например, когда необходимо сказать, что среди изделий такого-то типа допускается 5% брака. Пока в широко распространенных языках моделирования нет штатного механизма для моделирования высказываний подобного рода. Конструирование высказываний о множествах объектов в онтологическом моделировании рассматривается в одной из следующих глав.

### **Пересечение объектов**

В статической трехмерной пространственной модели объекты определены, маркированы и могут пересекаться своими пространственными частями. В динамической четырехмерной модели объекты могут пересекаться не только пространственными частями, но и временными. Например, роль «директор предприятия» и физический объект «Петр Михайлович» пересекаются своими временными частями с 1 октября 2008 года по текущий день. Наступит момент, начиная с которого они перестанут пересекаться. Способ отражения таких пересечений в онтологическом моделировании также описывается далее.

### **Список библиографических ссылок**

1. Стефанов Ю., Элиаде М., Калыгин В. Миф о вечном возвращении. Изд-во «Ладомир», М., 2000. 414 с.
2. Partridge C. Business Objects: Re-Engineering for Re-Use / 2<sup>nd</sup> ed. BORO Centre, 2005. 566 p.
3. Overview of BPMN. URL: <http://www.bpmn.org>
4. Overview of IDEF0. URL: [http://www.idef.com/idef0-function\\_modeling\\_method](http://www.idef.com/idef0-function_modeling_method)
5. Семиотика : антология / сост. Ю. С Степанов. Екатеринбург : М. : Академ. проект ; Деловая книга, 2001. 701 с.

## Глава 4

# КЛАССИФИКАЦИЯ В ОНТОЛОГИЧЕСКОМ МОДЕЛИРОВАНИИ

*А. Ю. Гребешков*

## § 1. Задача классификации и построение классификаторов

Деятельность по классификации относится к логическим операциям и предусматривает как определение, так и деление понятий. Примером наиболее значимой с научной точки зрения классификации является периодическая таблица химических элементов Д. И. Менделеева. Проведение классификации как операции деления в отношении какого-то понятия традиционно предполагает, что деление производится по одному принципу (основанию). Компоненты (члены) деления должны исключать друг друга, то есть не должны пересекаться; деление производится последовательно, без скачков и разрывов; должны быть перечислены все компоненты (члены) деления [1].

В классической постановке логической задачи классификации для анализируемого пространства объектов предполагается деление на виды (роды) и формирование рода-видового отношения, вопросы которого затрагивал еще Аристотель. Предложенные им категории (основания деления понятий самого высокого уровня) можно рассматривать как один из первых обобщенных подходов к процедуре классификации. Среди категорий Аристотеля можно выделить способ высказывания о вещи, возможность существовать вне предложения (категории в виде отдельных слов), способность нести общие смыслы и понятие категории как формы бытия. В дальнейшем развитие логики, математики и философии привело к созданию различных методов и систем категорий, в частности метод формирования категорий Пирса, комбинаторный метод Лейбница. Эти и другие методы позволяют формировать онтологические категории для онтологий верхнего уровня, онтологий

прикладных областей, причем не существует общепринятого формального способа определения приоритета категорий различного уровня – для этого требуются экспертные знания и эвристики.

На основании идей Аристотеля позднее была создана первая иерархия понятий (так называемое «дерево Порфирия»), включавшая их последовательное дихотомическое деление. В дальнейшем сформировались несколько концепций классификации, такие как эссециалистская концепция, номиналистическая концепция, концептуалистический подход [2].

Эссециалистская концепция строится на принципах философии Аристотеля и Платона, согласно которым у вещи есть некая «сущность», на основе которой создается иерархия понятия, и именно классификация должна выявить эту единственную систему понятий, исходя из сущности. Номиналистическая концепция, напротив, сущности отрицают и признает единичные объекты, из которых строятся классы, что соответствует формальной теории классов. Наконец, концептуалистический подход интегрирует предыдущие две концепции, допуская, что отношения между группами изменчивы, система может развиваться, а классификация является этапом в познании объектов.

В качестве типов классификационного деления можно выделить таксономические, мереологические, пространственные и временные. Таксономическое деление может быть основано либо на видоизменении видеообразующего признака, либо на дихотомическом делении. В первом случае члены деления определяются по изменению характеристики, выступающей в качестве основания деления. Например, для деления персонала по квалификации, если исходное понятие (делимое) есть «токарь, имеющий разряд», то видовые понятия по отношению к делимому могут быть «3-й разряд», «4-й разряд», «5-й разряд», «6-й разряд», как показано на рис. 4.1. Дихотомическое деление предусматривает деление сущности понятия на два класса, понятия о которых находятся в отношении противоречия, например сигналы электросвязи делятся на аналоговые и дискретные.

Мереологический тип деления предусматривает расчленение некоторого предмета на части. Далее каждую часть можно подразделить на другие части. Например, сборочная конвейерная линия состоит из распределительного конвейера, технологических участков на конвейере с рабочими местами, приспособлений-спутников, подъемников приспособлений, спутников на рабочих местах, накопителях на конвейере, перегружателей, сборочных автоматов, контроллеров, роботизированных комплексов сборки.



Рис. 4.1. Дихотомическое, таксономическое и мереологическое деление

Пространственный тип деления предусматривает использование деления по расположению объектов в декартовой или иной системе координат. Временное разделение предусматривает использование деления по времени, соответствующего определенным наборам параметров объекта. Например, существует пространственное деление в виде флористико-географического деления, когда выделяются ландшафтные зоны, наиболее подходящие для определенных видов растений (степень эндемизма флоры) с учетом климата [3].

Онтологическое моделирование классификации обусловлено характером изучаемой предметной области. Предметная область может рассматриваться как специфическая система, например как

множество объектов классификации вместе с отношениями между объектами и между их свойствами, где между объектами существуют отношения подобия, а между их свойствами – отношения идентичности и различия [4].

В дальнейшем будем считать, что аналитическая деятельность по классификации объектов определенной предметной области или группы областей – например, если речь идет о сложном производстве – рассматривается как внесение упорядочивания в представления об определенной области или сфере деятельности. В этом смысле под *классификацией* следует понимать разделение множества объектов предметной области на подмножества на основании их сходства или различия в соответствии с принятыми методами классификации, моделью деятельности предприятия или организации [5; 6] и целью моделирования. Классифицируемыми объектами предметной области могут быть производственные фонды, продукция предприятия, показатели деятельности предприятия, система безопасности предприятия, электронный документооборот и т. п.

В результате осуществления классификации формируется классификатор. Под *классификатором* понимается систематизированный свод наименований объектов классификации, признаков классификации, классификационных группировок и их кодовых обозначений. Сложность формирования классификатора состоит в том, что он, с одной стороны, должен быть одновременно доступен для однозначного и непротиворечивого восприятия человеком, с другой – должен быть машиночитаемым. Машиночитаемость предполагает, что классификатор может быть реализован с помощью различных программных средств, в том числе с помощью онтологий, и в определенных ситуациях может применяться автоматически программными алгоритмами.

В идеальном случае после получения исходной информации об объекте, например анализа содержимого документа в электронном виде, анализа описания промышленного изделия, с помощью распознавания образов или иной технологии анализируемому объекту автоматически назначается определенная группа или

несколько групп классификатора. Это, в частности, чрезвычайно актуально для защиты прав интеллектуальной собственности при подаче заявок на патент, полезную модель, промышленный образец для корректного определения раздела и группы международной патентной классификации, к которому относится заявка.

Основной целью классификации и разработки классификаторов для цифровизации предприятия является повышение эффективности экономической деятельности. Цель достигается решением взаимосвязанных задач, среди которых можно выделить следующие:

- поддержка корпоративного тезауруса для установления семантических связей между объектами в рамках информационного обеспечения бизнес-процессов;

- обеспечение совместимости государственных, муниципальных, частных и общедоступных информационных систем и ресурсов с информационными ресурсами предприятия или организации, в том числе выполнение требований по передаче сведений в государственные информационные системы;

- реализация обмена информацией внутри предприятия или организации;

- обеспечение единства ИТ-архитектуры предприятия или организации, включающей все массивы информации;

- обеспечение сопоставимости экономических и иных данных различных подразделений и служб предприятия в целях анализа;

- систематизация информации по единым правилам для прогнозирования хозяйственно-экономической деятельности, стратегического и оперативного планирования, формирования управленческой, финансовой, бухгалтерской и иной обязательной статистической отчетности;

- поддержка деловой кооперации и взаимодействия при производстве продукции и оказании услуг, проведении аукционов и конкурсов, участии в электронных торгах;

- создание условий для унификации и стандартизации форм и средств электронного документооборота, подготовки проектно-конструкторской и эксплуатационной документации;

– гармонизация с действующими нормами и правилами, требованиями стандартов различных уровней.

Для разработки, апробации и внедрения классификаторов на предприятии или организации целесообразно создать отдельный бизнес-процесс, входящий в состав процессов управления корпоративными данными.

В основе любого классификатора находится понятие *класса*, где под классом понимается подразделение классификационного ряда, принимаемое за основное. В этом смысле общепринятое понятие класса отличается от специфических определений в рамках онтологии или объектно-ориентированного подхода, хотя смысловая близость между ними, безусловно, есть. Она заключается в том, что в рамках класса либо формируется группа объектов со схожими свойствами, либо перечисляются некоторые количественные и качественные свойства, которые позволяют относить рассматриваемый объект к определенному классу.

Если объект после анализа совокупности его признаков и характеристик тем или иным способом отнесен к какому-то классу (в рамках принятой процедуры классификации), то такой объект становится *объектом классификации*, то есть элементом классифицируемого множества. В свою очередь, признаки, по которым проводилась классификация, становятся *признаками классификации*.

Отдельно стоит выделить классификационные группировки. *Классификационной группировкой* называется подмножество объектов классификации, полученное в результате классификации. Каждый объект может принадлежать одновременно нескольким классификационным группировкам. Например, какое-либо промышленное изделие может одновременно относиться к продукции машиностроения по отраслевому делению и к товарам российского производства – по стране происхождения. Объект может входить в несколько группировок и по одному основанию, если классификатор это допускает, например компания может относиться одновременно к предприятиям сферы торговли и сферы услуг.

При проведении классификации для обеспечения краткости записи в документе и для поддержки автоматизированной обработки данных (например, с помощью бар-кода) используют *код классификации*. Кодом классификации считается совокупность знаков (символов), букв алфавита, которые вместе используются для обозначения классификационной группировки вплоть до отдельного (единичного) объекта классификации. Присвоение кода осуществляется в рамках кодирования, код имеет длину, а номер позиции знака/знаков (цифры/цифр) в коде имеет определенное содержательное значение. Коды различных классификаторов могут отличаться длиной, составом символов (знаков), а также методами кодирования.

Существует несколько методов кодирования, из которых далее рассмотрим только базовые. При *последовательном* методе кодирования знаки на каждой ступени классификации зависят от результатов разделения на предыдущих ступенях. При параллельном методе классификационные группировки кодируются независимо друг от друга определенными разрядами или группами разрядов. При порядковом методе в качестве кода используются натуральные числа. Развитием порядкового метода является *серийно-порядковый метод* кодирования, при котором кодом служат числа натурального ряда, которые разбиты на серии или диапазоны. Серии или диапазоны чисел закреплены за объектами классификации с одинаковыми признаками. Примером последнего метода является универсальная десятичная классификация (УДК) статей научной литературы, классификация патентной информации и т. п. Например, УДК 004.822 – «Сети знаний, включая семантические сети». Здесь 004 соответствует категории «Информационные технологии. Компьютерные технологии. Теория вычислительных машин и систем», 8 – категория «Искусственный интеллект», 82 – «Представление знаний».

Следует отметить, что числовое кодирование категорий классификатора налагает некоторые ограничения на его использование. Например, если в какой-то категории окажется более 10 подкатегорий, адекватно отразить их без изменения структуры кода будет

невозможно. Числовое кодирование классификационных группировок (особенно вне сферы классификации промышленных изделий) обязано своим расцветом эпохе картотек, и на сегодняшний день с учетом развития средств обработки информации теряет актуальность в контексте многих функциональных задач.

Технологически для изделий современного производства принципиально важным является как возможность классификации отдельных сборочных единиц (узлов, комплектов, компонентов), так и сопоставление содержательного описания класса с некоторыми ключевыми технологиями производства и характеристиками, которые важны для определения характера применения изделий. Так, для полупроводниковых микросхем в рамках международной классификации можно указывать класс Commercial, Industrial/Military, Space. В онтологии классификатора микросхем целесообразно указывать связь класса, например с допустимым диапазоном температур при эксплуатации, и связь с конструктивным исполнением (типов корпуса). Первому разделу классификации – Commercial – соответствуют микросхемы для диапазона температур от 0 до +70 °C в пластмассовом корпусе. Второму разделу классификации, Industrial/Military, соответствуют микросхемы для диапазона температур от –40 до +125 °C в металлокерамическом корпусе. Наконец, Space соответствует примерному диапазону температур от –200 до +150 °C, микросхема может быть изготовлена на сапфировой подложке в керамическом корпусе.

Следует иметь в виду, что в Российской Федерации на уровне отделов технического контроля действует так называемая приемка, когда в зависимости от условий тестирования производится классификация готового изделия. Известна приемка «1», когда микросхемы тестируются на предприятие-изготовителе, приемка «5» предполагает приемку заказчика, в том числе и прежде всего тесты военной приемки, наконец, приемка «9» означает специальные тесты для космоса (например, в вакуум-камере) и тесты на радиационную стойкость, что важно для ядерных электростанций. Рассмотренные классы приемок подразумевают примерно те же

диапазоны рабочих температур и типы корпусов, что и классы Commercial, Industrial/Military, Space.

Даже на таком кратком примере ясно, что часто возникает задача установления совместимости (мэппинга, установления соответствия, переходного ключа) одного классификатора с другим классификатором, причем для одной и той же классификационной группы. *Переходной ключ* представляет собой таблицу, устанавливающую соответствие каждой группировки одного классификатора другому классификатору. Совместимость классификаторов определяется общей терминологией, единством или взаимным соответствии нормативной документации и формализма описания предметной области применительно к задачам управления и учета на современном цифровизированном предприятии.

Примеры детально проработанной классификации можно найти в Общероссийском классификаторе видов экономической деятельности ОК 029-2001 и в Общероссийском классификаторе видов экономической деятельности, продукции и услуг (кроме частей I и IV) ОК 004-93 с Изменениями 4/2002, 5/2011, 6/2012.

Развитая система классификации и кодирования должна использоваться при создании единого информационного пространства управления цифровым предприятием или организацией. Применение современной системы классификации обеспечивает стандартизацию информационной модели бизнеса, позволяет достичнуть более высокого уровня автоматизации и оперативности принятия решений за счет корректного и целостного предоставления пользователю необходимой справочной, нормативной, статистической и оперативной информации, касающейся производства и технологий.

При практической реализации системы классификации и кодирования на современном цифровом предприятии с развитой ИТ-инфраструктурой целесообразно применять информационные системы, обладающие следующими функциональными возможностями:

– обеспечение доступа к классификаторам и справочникам для их просмотра и корректировки ответственным персоналом;

- обеспечение доступа к классификаторам и справочникам для контроля именования объектов классификации, их параметров с целью обеспечения соответствия принятой системы классификации и кодирования;
- подготовка выборок из классификаторов и справочников для функциональных подсистем управления предприятием;
- импорт данных внешних классификаторов и справочников, прежде всего общегосударственных/федеральных, финансовых, бухгалтерских, налоговых, международных;
- подготовка и вывод отчетов о состоянии различных классификаторов и справочников;
- администрирование системы классификации и кодирования;
- расширение состава классификаторов и/или справочников, изменение логических связей между значениями их полей;
- ввод и удаление записей из классификаторов и справочников, в том числе групповые операции ввода/удаления – по признаку значения отдельного поля или совокупности полей;
- рассылка сообщений заинтересованным пользователям об обновлении классификаторов или справочников с учетом иерархии справочной информации;
- обеспечение логической целостности совокупности классификаторов и справочников;
- блокировка ввода записей классификаторов и справочников при нарушении логической целостности системы или несанкционированном доступе к информации;
- разграничение доступа отдельных категорий пользователей к модифицируемой информации.

С учетом сказанного, при классификации и кодировании производственной продукции цифрового предприятия, прежде всего в рамках внутрикорпоративных классификаторов, целесообразно проводить работы по следующим направлениям:

- анализ действующих международных, общероссийских, межгосударственных и отраслевых классификаторов и кодификаторов, нормативно-правовых актов, руководящих документов в части

применения используемых систем и способов классификации в целях использования «как есть» или разработки дополнительной внутрикорпоративной системы классификации и кодирования предприятия;

- определение перечня необходимых классификаторов, сферы их применения, формализация оснований классификации;
- разработка процедур изменения (добавление, исключение позиций) внутрикорпоративных классификаторов;
- разработка состава, структуры, форматов информационной базы данных для поддержки внутрикорпоративных классификаторов;
- определение этапов работ по формированию внутрикорпоративных классификаторов.

Если предприятие выбирает онтологическое моделирование и семантические технологии в качестве методологического и технического базиса создания автоматизированных систем и цифровых решений, то необходимо продолжить работу по следующим направлениям:

- выбор методики представления корпоративных классификаторов в онтологии, создание необходимых структур в онтологической модели;
- выбор программных средств для поддержки корпоративных классификаторов в онтологии;
- согласование и утверждение процедур изменения фрагмента онтологии, содержащей классификаторы;
- проектирование и реализация способа распространения всех классификаторов, входящих в состав нормативно-справочной информации, во все нуждающиеся в ней корпоративные автоматизированные системы.

Безусловно, разработка системы и онтологии классификации и кодирования продукции цифрового предприятия является сложной организационной, методической и технологической задачей. На практике с решением этой задачи часто увязываются вопросы классификации и кодирования внутрикорпоративного документооборота, проектно-конструкторской документации, включая

такие вопросы, как разработка наряд-заказов, сменно-суточного задания, ведение электронного архива проектно-конструкторской и эксплуатационной документации, поиск в электронном архиве по децимальным номерам, титулам проектов, объектам проектирования и прочее.

Для хранения, обработки и предоставления сведений классификаторов часто применяются справочники (перечисления). Под *справочником* здесь понимается перечень нормализованной информации для описания и классификации объектов предметной области. Справочник включает краткие сведения о свойствах и характеристиках объектов с учетом многообразия объектов предметной области, например типов промышленной продукции, используемых проектно-конструкторских и эксплуатационных документов. В справочниках, входящих в состав нормативно-справочной информации, на практике используются нормализованные, то есть паспортные характеристики свойств объектов, которые были установлены согласно их классификационным признакам на этапе добровольной или обязательной сертификации. Также классификационные признаки устанавливаются положениями ГОСТ, ГОСТ-Р, техническими регламентами, иными нормативно-правовыми актами, часть из которых была рассмотрена выше. Информация справочников может распространяться и на общие свойства объектов классификации, когда для задания значения свойства соответствующее описание выбирается из готового справочника.

Отдельно стоит остановиться на вопросе места классификации в стандартизации деятельности современного цифрового предприятия, согласно рис. 4.2. На уровне прагматического подхода осуществляется описание бизнес-процессов предприятия, сопровождающего их документооборота и схемы обмена информационными сообщениями.

В рамках семантического подхода выполняется описание значения и смысла данных, которыми располагает предприятие, а также представление данных в форме документов. Синтаксический подход рассматривает способы кодирования элементов данных

и структуризацию документов. В рамках технологического подхода формируются номинальные, целевые и максимально достижимые показатели, определяющие свойства и возможности ИТ-инфраструктуры цифрового предприятия, способы обмена данными и документами. Наконец, стандарты безопасности обеспечивают комплексный подход к защите контента и данных как в момент передачи, так и на этапах сбора, хранения и обработки.

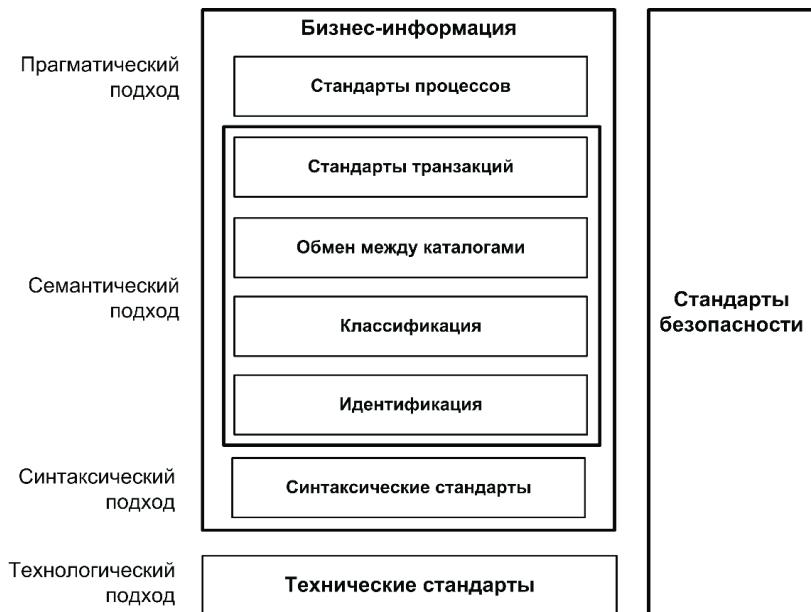


Рис. 4.2. Структура стандартов работы с данными цифрового предприятия

Семантические стандарты в целом могут ориентированными на данные (дата-ориентированные) и ориентированными на процессы (процесс-ориентированными). Дата-ориентированные стандарты предоставляют основу для единообразного представления информации о бизнесе, с упором на однозначные обозначения поставщиков, продуктов (изделий) и их свойств. Также эти стандарты используются для описания мастер-данных.

Процесс-ориентированные стандарты предназначены для единого представления структур бизнес-процессов и предоставления стандартизованной формы описания сложных рабочих/операционных процессов внутри предприятия и между предприятиями.

Некоторые стандарты (отраслевые, общегосударственные, межотраслевые) не могут быть четко отнесены к определенному типу, поскольку они стандартизируют элементы из нескольких областей. Таким образом, стандарты могут перекрывать зоны действия друг друга, дополнять друг друга или конкурировать между собой. Кроме того, могут значительно отличаться уровни удобства их использования, выполнения требований стандартизации. В настоящее время нет всеобъемлющего стандарта, охватывающего все аспекты семантической стандартизации.

Примером синтаксического стандарта является XML, который позволяет описывать структуру документа с помощью разметки с помощью тегов (меток) или хештегов. Примерами технических стандартов являются протоколы HTTP(S), TCP/IP, POP3, FTP.

Из-за увеличения сложности целей стандартизации на разных уровнях, стандарты часто используются в сочетании. Наиболее известные в мире стандартизованные формальные модели данных для описания и классификации продуктов содержатся в стандартах ISO 13584–42 и IEC 61360–2, которые используют одну и ту же модель данных.

Для аналогичных или идентичных продуктов существуют различия в структуре и формулировках стандартов классификации. Причины такого положения дел обусловлены специфическими для конкретной отрасли требованиями или внутренними потребностями предприятия. Классификации и описанные ими свойства могут быть сформированы различными способами и, следовательно, могут по-разному интерпретироваться. Для решения этих проблем Национальным германским институтом стандартизации был создан, например, онлайн-словарь стандартизованных свойств продукта<sup>1</sup> на основе формальных информационных моделей ISO

---

<sup>1</sup> Расположен на сайте [www.din.de](http://www.din.de).

13584, IEC 61360 и DIN 4002. В словаре нормированные свойства продукта объединяются вместе с их стандартизованными определениями для обеспечения обмена информацией о каталогах изделий и деталей в различных отраслях промышленности.

Кроме того, стандарты отличаются в отношении их ориентации на конкретные отрасли промышленности. Горизонтальные стандарты обеспечивают классификацию продуктов в разных отраслях. Напротив, вертикальные стандарты сосредоточены на одной конкретной отрасли и обеспечивают углубленную классификацию в пределах своей проблемной области. В результате разработано более 35 систем горизонтальной и вертикальной классификации [7].

Наиболее известными горизонтальным системами для классификации товаров и услуг являются:

– UNSPSC (United Nations Standard Products and Services Code) – стандартный код продуктов и услуг Организации Объединенных Наций, являющийся открытым, глобальным многосекторальным стандартом для классификации продуктов и сервисов;

– CPC (Central Product Classification), классификация основных продуктов, это всеобъемлющая классификация товаров и услуг, которая поддерживается Отделом статистики ООН;

– GPC (Global Product Classification), глобальная классификация продуктов, входит в состав системы GS 1, обеспечивая общий язык для управления категориями с набором общих категорий для группировки продуктов в глобальном масштабе;

– NAICS (North American Industry Classification System), Североамериканская промышленная классификация, это система классификации услуг, разработанная совместно США, Канадой и Мексикой, чтобы обеспечить сопоставимость статистических данных о деловой активности по всей Северной Америке; NAICS находится в ведении Бюро переписи населения США;

– eClass – это международный стандарт для классификации и описания продукции (материалов и услуг);

– CPV (Common Procurement Vocabulary) – общий словарь для закупок, система классификации для европейских государственных

закупок, которая поддерживается SIMAP, Système d'Information des Marchés Publics, европейским сервисом электронных закупок;

– глобальный многоязычный продукт ePDC и его классификация для электронной коммерции и электронного бизнеса – это европейская инициатива CEN по гармонизации систем классификации и многоязычных электронных каталогов и соответствующее им моделирование данных;

– система кодификации НАТО, которая управляет Организацией Североатлантического договора, как единая и общая система идентификации и классификации, призванная улучшить логистические системы и операции для участвующих стран и облегчить глобальные логистические операции;

– eOTD, Открытый технический словарь ECCMA – это международный открытый стандарт языка описания для каталогизации отдельных лиц, организаций, местонахождения производства, товаров и услуги. Он поддерживается Ассоциацией управления электронными кодами торговли (Electronic Commerce Code Management Association ECCMA), США;

– BEC (Broad Economic Categories), классификация по широким экономическим категориям, предназначена для классификации статистики торговли в области укрупненных экономических классов товаров, поддерживается ООН.

Некоторые из этих стандартов, например GPC, могут использоваться для синхронизации с основными пулами данных, такими как SINFOS Data pool (SINFOS GmbH) или Трансора (Transora, UCCnet). Эти основные пулы данных являются центральными информационными банками (узлами) для многостороннего обмена данными о товарах между деловыми партнерами.

Известные вертикальные стандарты включают следующее:

– RNTD (RosettaNet Technical Directory), технический каталог RosettaNet, содержит общие свойства для определения продуктов для включения в партнерский интерфейс RosettaNet, который определяет бизнес-процессы между торговыми партнерами в области электроники и телекоммуникаций;

– Proficlass, немецкая классификация для строительной отрасли, которая предназначена для совместной работы с сопоставимыми инициативами для обеспечения совместимости;

– PIDX (Petroleum Industry Data Exchange), обмен данными нефтяной промышленности, используется для формирования сообщений электронных бизнес-процессов в XML и EDI, а также классификации продуктов. Предоставляет стандарт сервисных компонентов с определенной номенклатурой и свойствами;

– SNITEM, Syndicat National de l'Industrie des Technologies Médicales фокусируется на индустрии медицинских технологий на международном уровне;

– гармонизированная система описания и кодирования товаров Всемирной таможенной организации (ВТО) (HS Harmonized Commodity Description and Coding System by the World Customs Organization) представляет набор номенклатуры, используемой в качестве основы семантических стандартов для упрощения процедур торговли, сбора таможенных пошлин и международной статистики торговли.

## § 2. Методы классификации объектов

Для дальнейшего рассмотрения проблем цифровизации предприятия важным является вопрос о методах классификации. Сначала рассмотрим *иерархический метод классификации*. В этом случае осуществляется последовательное разделение множества объектов на подчиненные друг другу классификационные группировки с тесными связями между классами. Множество объектов классификации, заданное, как правило, заранее, последовательно делится на подчиненные множества.

Такой подход требует определения обобщенных свойств объектов предметной области. Далее эти свойства детализируются, дробятся, уточняются в рамках конкретного классификатора. Этот подход является аналитическим, то есть построение классификатора осуществляется от общего к частному. Основную трудность

представляет не дробление или перераспределение отдельных разделов или классов, а первоначальное определение множества объектов, что является нетривиальной экспертной задачей. Очевидно, что в случае появление некоего объекта, который не входил в первоначальный контур, появляется сложная задача изменения иерархического классификатора, что, возможно, приведет к разработке совершенно нового классификатора.

Иерархический классификатор включает несколько ступеней классификации, где под ступенью понимается этап работ по классификации, в результате которого выявляется совокупность классификационных группировок или объектов классификации. Каждому этапу классификации можно назначить определенный код: АА, ББ, ВВ, ГГ и т. д., согласно схеме на рис. 4.3.

Количество этапов классификации (АА, ББ, ВВ, ГГ … и т. д.) на рис. 3, число позиций в цифровом коде, в том числе по каждому полю АА, ББ, ВВ, ГГ, целесообразно определять либо требованиями нормативно-правовых актов, либо по результату дополнительного исследования. На каждой ступени классификации деление осуществляется по наиболее значимым функциональным, техническим или иным классификационным признакам.

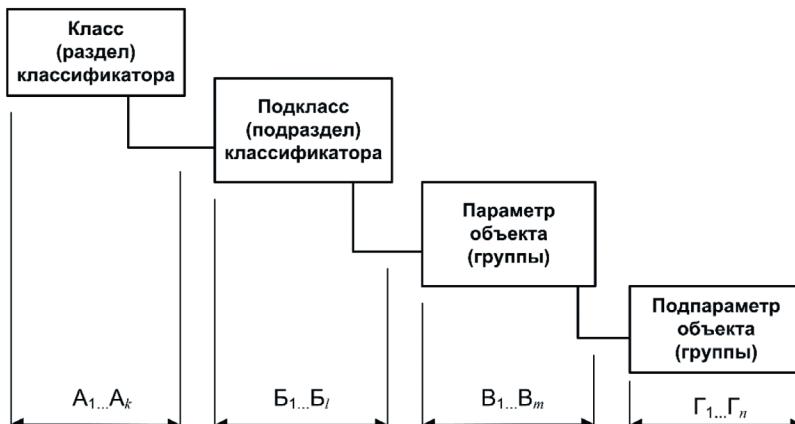


Рис. 4.3. Структура классификатора для иерархического метода

Классификация может быть завершена на любой ступени иерархической классификации. Целесообразно, чтобы класс и подкласс определялся с помощью нормативно-правовых актов различного уровня, а в отношении параметров и подпараметров может использоваться внутрикорпоративная (внутрипроизводственная) классификация.

Примером иерархического классификатора является Общероссийский классификатор управленческой документации ОК 011–93 со всеми актуальными изменениями. Структура кодового обозначения унифицированной формы документа по ОКУД: АА ББ ВВВ Г, где АА – класс форм, ББ – подкласс форм, ВВВ – регистрационный номер, Г – контрольное число. Например, код классификации 04011012 соответствует электронному платежному поручению. Во избежание дублирования информации в рамках классификатора можно использовать систему перекрестных ссылок, а также систему информационных справочников, поддерживаемых системой управления нормативно-справочной информацией.

Другим методом классификации является *фасетный метод*, при котором заданное множество объектов классификации параллельно разделяется на независимые подмножества по различным основаниям классификации. При этом разные признаки не связаны друг с другом. Фасетный метод позволяет создавать гибкую классификацию, используя при этом ограниченное число фасет (признаков) и классификационных группировок.

В контексте фасетной классификации можно говорить и о многоаспектной классификации, где под аспектом понимается точка зрения субъекта (менеджмента предприятия, собственника, аналитика, проверяющего и контролирующего органа) на объект классификации, который характеризуется одним или несколькими признаками. Достоинством метода является отсутствие жесткой классификационной структуры и зафиксированных наперед конечных группировок для классификации. В фасетном методе классификационные группировки образуются путем комбинации значений, взятых из соответствующих фасетов.

Последовательность расположения фасетов при образовании классификационной группировки задается фасетной формулой, которая может строиться как вручную, так и автоматически, согласно рис. 4.4. Количество фасетных формул определяется возможными сочетаниями признаков и может меняться по мере развития информационной системы предприятия. В результате классификатор, построенный на основе фасетного метода, соответствует следующим требованиям:

- состав признаков одного фасета не повторяется в других фасетах этого же класса;
- в состав классификатора включаются только такие фасеты и признаки, которые необходимы для решения конкретных задач.

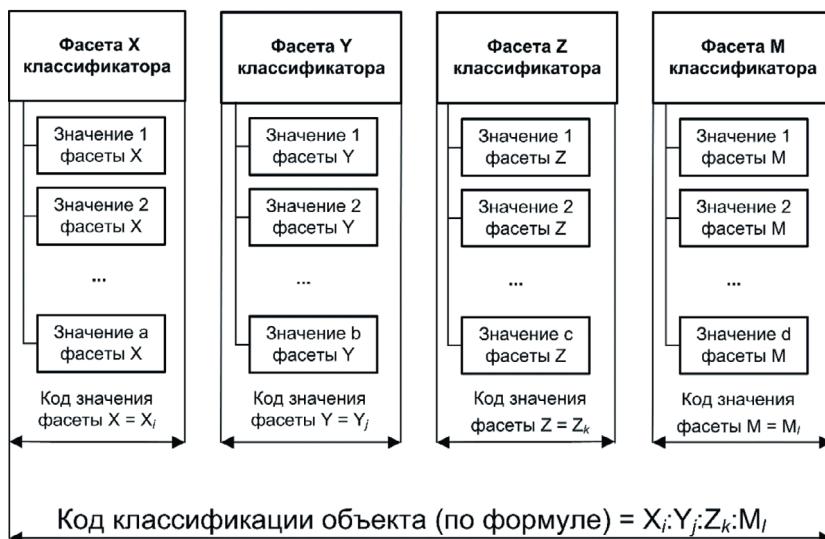


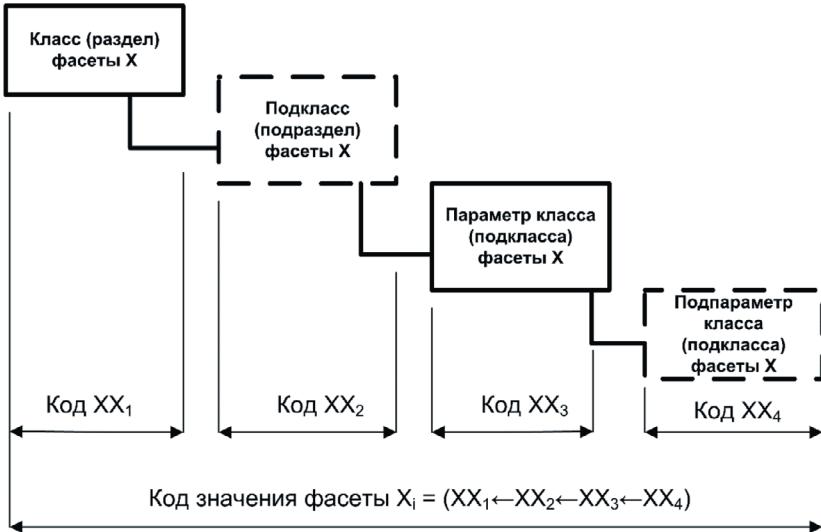
Рис. 4.4. Формирование кода классификации фасетным методом

Информация классификатора, построенная по фасетному методу, делится на два основных типа:

- первый тип – информация, характеризующая номенклатуру (перечень) объектов классификации;

– информация, подробно характеризующая свойства объектов классификации.

Внутри фасета на рис. 4.4 значения признаков образуют сложную иерархическую структуру, поскольку существует соподчиненность признаков на основании локальных классификаторов информационной системы предприятия. Поэтому при описании объекта классификации и его признаков для каждого фасета можно использовать последовательный метод формирования кода классификации, как показано на рис. 4.5, а для формирования общего классификационного признака можно использовать параллельный метод классификации. Структуры на рис. 4.4 и 4.5 позволяют обоснованно сформировать значение фасета с учетом обоих типов информации об объекте классификации.



*Примечание: «←» - отношение «является частью»*

*Рис. 4.5. Внутренняя структура значений фасета*

Онтологию классификатора формально можно определить как упорядоченное множество (кортеж) вида:

$\langle D, Rel, Attr \rangle$ ,

где  $D, D \neq \emptyset$  – конечное, непустое множество фасет (разделов классификатора), сформированных в результате анализа предметной области, которая описывается разрабатываемой онтологией;

$Rel$  (Relations) – конечное множество, определяющее типы бинарных отношений между объектами классов предметной области – фасетами;

$Attr$  (Attributes) – конечное множество, определяющее типы лiteralных (символьных, числовых, логических) свойств, значениями которых могут обладать объекты классов предметной области, то есть фасеты.

В рамках онтологии, создаваемой с использованием фасетного классификатора, сформулированные понятия и типы отношений будут использованы для формулирования истинных утверждений (аксиом) о классификации конкретных объектов и их отношениях. Моделирование истинных утверждений представляет собой ограничивающее условие, которое тем не менее обеспечивает корректное использование понятий предметной области и интерпретацию логических взаимосвязей между объектами, сформулированных при помощи введенных понятий.

Классы  $D$  формируются либо с помощью исчерпывающего описания свойств объектов предметной области, позволяющих отнести объект к данному классу при классификации, либо задаются исчерпывающим перечнем объектов, отнесенных к данному классу. Распределение объектов по классам может быть заранее неизвестно.

В случае, если отнесение объекта к определенному классу в процессе классификации определяется неотделимыми свойствами такого объекта, а свойства такого объекта измеримы, то можно обоснованно допустить наличие гипотезы  $H^{Attr}$  о существовании измеримого свойства, которое в информационной модели можно описать с помощью свойств.

Свойства объектов, входящих в класс, в онтологии можно описать с помощью множества вида:

$$Attr = \{attr_i\}, i \in Ind^{Attr},$$

где  $Attr$  – множество свойств класса объектов;

$Ind^{Attr} = \{1, 2, \dots, i, \dots, L\}$  – множество индексов для идентификации свойств объектов классификации.

Если значения свойств объектов классификации, входящих в определенный класс, измерямы, то для объектов такого класса  $D_i$  существует множество вида:

$$\{(ms_i, Smb_i, Rel_i^{Smb})\}, i = 1, \dots, S,$$

где  $ms_i$  – измерительная процедура;

$Smb_i$  – набор (множество) наперед известных символов, которыми обозначаются единицы измерения;

$Rel_i^{Smb}$  – отношения, которые приписываются свойствам объектов классов определенные символы  $Smb_i$ , где верно отношение  $ms_i : D_i \rightarrow Smb_i$ .

Наличие непустого отношения вида  $ms_i : D_i \rightarrow Smb_i$  свидетельствует о наличии измерительной процедуры, которая позволяет корректно описать свойство объекта класса, прежде всего с семантической точки зрения: например, температура не может быть быстрой.

Следует отметить, что не всегда признаки объекта класса являются измеримыми, то есть не существует отношения  $ms_i : D_i \rightarrow Smb_i$ , другими словами, полностью отсутствует множество  $Smb_i$  как множество наперед заданных символов. Это характерно для описания отношений вида «относится к ...», «действует на ...». Можно определить бинарное отношение вида  $Rel_i^{Ref} = \{\text{Имя\_Отношения}\}$  – символы, описывающие отношение, для которого в языке существует знаковое или лексическое (фразовое) обозначение. Соответственно, компонент (фасет) может участвовать в бинарном отношении  $\{\text{Имя\_Отношения}\}$ .

Состоительность онтологии определяется в том числе тем, что в предельном случае не должно быть ни одного объекта, который не был бы отнесен к определенному классу (классифицирован). При этом может быть реализована логическая ошибка, состоящая

в том, что не был выделен класс объектов, необходимый с точки зрения прагматики использования модели, а объекты такого класса были неверно (ложно) отнесены к другому, «чужому» классу.

Рассмотрим пример онтологического моделирования фасетного классификатора для системы анализа сообщений о событиях безопасности современного предприятия.

К классам, которые моделируют фасеты, относятся следующие:

- сфера безопасности;
- объект безопасности;
- объект защиты;
- способ нарушения безопасности объекта;
- зона нарушения безопасности объекта защиты.

Далее рассмотрим определения указанных классов.

*Сфера безопасности* – глобальная область, отражающая сущность защищаемых объектов безопасности и объектов защиты.

*Объект безопасности* – это среда жизнедеятельности и/или условия существования человека, безопасность которых необходимо обеспечить в рамках безопасности предприятия.

*Объектом защиты* является человек, материальный предмет, производственная среда в целом (совокупность зданий производственного назначения, микроклимат помещений), для которых поражение, повреждение, дисфункция и/или ухудшение условий эксплуатации явно или потенциально могут привести к причинению вреда (получению ущерба). Объект защиты противостоит деструктивным воздействиям благодаря совокупности своих свойств или мер/способов применения существующей защиты.

*Нарушения безопасности объекта* – деструктивные воздействия различной природы, в том числе непреодолимой силы, могущие нанести ущерб/вред объектам защиты и объектам безопасности. Нарушения безопасности могут быть описаны видом воздействия или способом нарушения безопасности. *Вид воздействия* – нанесение вреда/ущерба объекту, неразрывно связанное с природой происхождения или свойствами фактора (угрозы), нарушающего безопасность объекта. *Способ нарушения*

*безопасности объекта* – действие, состоящее в реализации угроз или рисков, которые наносят вред/ущерб объекту защиты или объекту безопасности.

*Зона нарушения безопасности объекта защиты* – территория, в границах которой происходит нарушение безопасности объекта защиты. Смысловые отношения между рассмотренными компонентами представлены на рис. 4.6.

Далее приводится краткое описание каждого фасета (компоненты) с разбивкой по категориям (подклассам), с перечнем возможных значений (указаны в скобках).

Сфера безопасности включает санитарно-эпидемиологическую обстановку, экологическую обстановку, чрезвычайные ситуации, энергоснабжение предприятия, пожарную безопасность. К объектам безопасности относятся жизнь и здоровье людей, материальные ценности, окружающая среда.

Объектами защиты являются:

- человек (персонал предприятия);
- информационные ресурсы предприятия;
- производственные здания, здания офисного типа;
- некапитальные сооружения и нежилые здания (склады);
- проезды и подъезды к предприятию, автостоянки;
- инфраструктура (водопровод, газопроводы, электросети, контактные сети, трубопровод), объекты топливно-энергетического комплекса (нефтехранилища, дизель-электростанции и т. п.);
- природные объекты вокруг предприятия (водные объекты, атмосфера, источники питьевой воды и т. п.).

Способами нарушения безопасности объекта защиты являются:

- информационное воздействие (вирусная атака, перехват управления, фишинг, DDoS атака и т. п.);
- техногенный способ (производственная травма от механизма, травма от электричества, неисправность, дефект, полный отказ оборудования);
- пожар (пожар твердых горючих веществ, пожар газов, пожар металлов, пожар энергоустановок и т. п.);

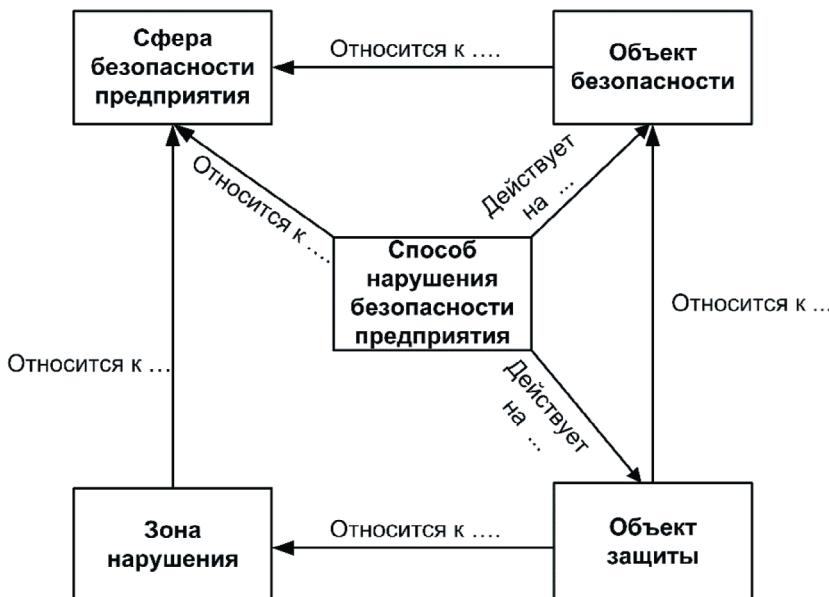


Рис. 4.6. Отношения между классами при онтологическом моделировании классификатора сообщений о нарушении безопасности предприятия

- погодное воздействие (ливневый дождь, молния, наводнение, шквал, обледенение);
- геологическая активность (проседание земли, обрушение грунта);
- взрыв (производственная авария);
- химическое воздействие (разлив ртути, разлив нефтепродуктов, выброс отравляющих веществ);
- биологический способ (отравление продуктами/напитками, отравление вредными веществами/газами, заражение опасным заболеванием, инфицирование).

К зонам нарушения безопасности объекта защиты относятся:

- территория предприятия в целом;
- отдельный объект;
- прилегающая территория.

С помощью представленного классификатора в масштабах города, региона или федерального округа можно анализировать

сообщения о состоянии безопасности промышленных предприятий. Отнесение события безопасности к определенному разделу классификации позволяет принять требуемые меры оперативного реагирования при получении сообщения о нарушении промышленной безопасности.

Еще одним методом классификации является *дескрипторная (описательная) система классификации*, которая по своему характеру близка к естественному языку и применяется для поиска информации, ведения тезаурусов и словарей. При дескрипторной системе выбирается множество ключевых слов или словосочетаний, описывающих предметную область.

Следует еще раз подчеркнуть, что на сегодняшний день не существует универсального, объективного единого метода классификации понятий реального мира. Любая классификация, прежде всего в рамках онтологий общего назначения [8], определяется точкой зрения, подходом и философским понятийным аппаратом, который используют разработчики онтологии, что особенно важно для случая слабоформализованной предметной области. При построении онтологии общего назначения решается задача начальной или первичной классификации объектов реального мира на основе классификации признаков. Здесь неизбежно возникает проблема отображения знаний о реальном мире, а не представлений о реальном мире, что можно видеть в онтологии BFO<sup>2</sup> для научной сферы, где используются разные точки зрения на объекты и их границы.

Более очевидным и, как правило, результативным, можно считать разработку системы классификации и определение классификационных признаков в рамках достаточно формализованной предметной области, где применяются относительно стабильные понятия и тезаурус, известны классификационные признаки, можно объективно указать вершину таксономии. Однако при синтезе таких систем классификации в более глобальные системы могут возникнуть существенные противоречия с учетом разнотечений авторов в части общих понятий.

---

<sup>2</sup>Расположена на сайте basic-formal-ontology.org.

Базовым критерием, который определяет успешность или неудачу предлагаемой классификации, является практический опыт применения, который доказывает или, наоборот, опровергает принятую разработчиками точку зрения на проблему классификации, в том числе в ее онтологическом аспекте. Опыт применения заключается в наполнении онтологии и классификаторов профильными данными и сведениями, что в итоге позволяет определить, все ли объекты предметной области могут быть классифицированы, а при «выпадении» определенных объектов целесообразно переработать онтологию. Примерами относительно удачных решений в различных формализованных предметных областях можно считать следующие решения:

- основанную на онтологии систему автоматической классификации и рэнкинга WEB-документов [9];
- основанную на онтологии систему автоматической классификации WEB-услуг [10];
- основанную на онтологии семантическую систему классификации спутниковых снимков с изображением крупных аварий [11];
- семантический поиск данных и основанная на онтологии семантическая классификация [12–14];
- основанную на онтологии систему классификации неаварийных событий в рамках концепции «Умный город» [15];
- основанную на онтологии систему классификации неструктурированных документов [16];
- основанную на онтологии систему классификации пациентов с лимфомой на основе семантического анализа описаний изображений и снимков пораженных участков [17];
- основанную на онтологии систему многомерной классификации экономических статей [18];
- основанная на онтологии систему классификации типов эпилепсии [19];
- систему классификацию с применением онтологии в химии [20].

Для ряда из рассмотренных примеров в целях совершенствования разработанной онтологии и классификаторов применяются методы машинного обучения.

### § 3. Атрибуты и идентификаторы классифицируемых объектов

Рассмотрим возможные атрибуты для описания объекта классификации на примере обобщенного типа продукции современного предприятия. Атрибуты описывают технические и функциональные свойства объекта, которые являются относительно устойчивыми во времени. При проекции логически выделенных атрибутов в онтологическую модель может использоваться как создание обыкновенных свойств в смысле стандарта OWL или их комбинаций, так и введение специальных классов объектов для представления значений сложных свойств описываемого объекта. Так, значение атрибута может быть представлено как график или номограмма (номограммы), описывающие зависимость характеристик устройств от различных факторов – температура, срок эксплуатации и т. п. Среди достаточно широкого набора атрибутов, характеризующих объект классификации, можно выделить следующие [21; 22].

Атрибут № 1. **Имя продукта**, присвоенное производителем; фирменное или маркетинговое обозначение.

Атрибут № 2. **Декларация производителя о соответствии стандартам** – список обязательных стандартов, которым соответствует данное устройство на российском рынке.

Атрибут № 3. **Структура продукции** – список и/или описание функциональной или физической структуры устройства с указанием деталей или сборочных единиц, которые являются составной частью данного устройства (комплекса).

Атрибут № 4. **Назначение устройства** – обозначает базовое функциональное использование или потребительскую ценность устройства.

Атрибут № 5. **Техническая информация** – ссылки на схемы устройств, планы монтажных позиций, информация инженерного характера, которая описывает детальные функции или множество функций данного устройства.

Атрибут № 6. **Физические размеры** – определяет метрические размеры (длину, ширину, высоту, объем) устройства.

Атрибут № 7. **Описание** – свойства устройства как перемещаемого объекта, в частности является ли устройство съемным, навешиваемым, монтируемым, в том числе является ли сборной единицей.

Атрибут № 8. **Вес** – определяет метрическую массу устройства для транспортировки (брutto/нетто) в единицах измерения массы.

Атрибут № 9. **Параметры электропитания** – данные производителя в отношении характера и значения напряжения, характера и значения тока питания, частоты переменного тока, потребляемой мощности, требований к предохранителям.

Атрибут № 10. **Опасные материалы** – извещение о том, где и какие опасные материалы использует производитель, включая данные об опасности лазерного излучения.

Атрибут № 11. **Версия замонтированного программного обеспечения** – указывает на возможность устройства загружать (installировать) программное обеспечение из ПЗУ (постоянного запоминающего устройства).

Атрибут № 12. **Требования к монтажной позиции** – определяет количество и характеристику монтажных позиций (слотов, площадок), необходимых для установки устройства.

Атрибут № 13. **Условия эксплуатации** – обозначает приемлемые (штатные) условия для эксплуатации устройства (внешнее оборудование помещения, внутреннее инженерно-техническое обеспечение, температурно-влажностный режим, пылевлагозащита).

Атрибут № 14. **Кабели и провода для соединений** – указание производителя на используемые типы проводов, шнурков, кабелей связи, физических цепей для монтажа устройств.

Атрибут № 15. **Взаимозаменяемость** – предоставляет возможность облегчить процесс поиска аналога устройства по принципу «такое на такое же» в рамках одного и того же производителя.

Атрибут № 16. **Совместимость** – описывает совместимость устройства «сверху вниз». Совместимость «сверху вниз» означает,

что прежняя версия устройства может быть заменена новой и никак не наоборот. Если возможно, указываются ограничения на указанную совместимость.

Перейдем от задачи классификации и описания атрибутного состава объектов к задаче их идентификации в рамках системы технического учета и паспортизации, которую рассмотрим на примере сетей, средств и сооружений связи.

В связи с созданием системы классификации дополнительной задачей является разработка принципов единой *идентификации устройств* (Equipment Identity, EI). Единая идентификация предполагает представление информации о каждом классифицированном объекте в кратком, унифицированном и функционально-ориентированном формате. Под *идентификацией* понимается процедура, позволяющая распознать конкретный экземпляр (единицу) продукции в рамках классификационной группы или объектов классификации. Идентификация может проводиться автоматически.

Одной из наиболее известных систем идентификационных номеров является глобально унифицированный стандарт GS 1. Этот тип системы открытых номеров обеспечивает всемирная уникальная и точная идентификация предметов, продуктов, активов, услуг и организаций, логистических единиц и их местоположения. Этот стандарт часто используется в системах цепочки поставок. Представление идентификаторов в форме GS 1-штрих-кодов создает основу для легкой передачи информации об объекте. Разработка была начата в 1977 году как европейская, позднее международная система EAN (European/International, Article Number) усилиями ассоциации производителей и торговых организаций из двенадцати европейских стран. Системы идентификации позволяют создать глобальное, уникальное, не совпадающее с аналогами обозначение изделия с использованием следующих основных стандартов:

– глобальный номер местоположения GLN (Global Location Number) для идентификации организаций и их единиц в их функции в качестве физических мест и юридических лиц;

- глобальный номер товарной позиции, GTIN (Global Trade Item Number) для идентификации статей, продуктов, услуг;
- код контейнера для последовательной доставки, SSCC (Serial Shipping Container Code) для идентификации элемента любого состава, применяемый для транспортировки и/или хранения контейнера;
- глобальный идентификатор возвращаемого актива, GRAI (Global Returnable Asset Identifier) для идентификации продуктов и изделий многократного использования или транспортного оборудования;
- глобальный идентификатор типа документа, GDTI (Global Document Type Identifier), используемый для комбинированного типа документа с дополнительным серийным номером для доступа к необходимой информации о базе данных для управления документами.

Идентификационные номера могут отображаться в символе штрих-кода для автоматического считывания. Этот символ основан на уникально определенных элементах данных и полях. Символы могут полностью отобразить ASCII-код, включить контрольную цифру и обладают хорошей читабельностью и возможностью печати, обеспечивая при этом высокую плотность данных.

Другими хорошо известными стандартами идентификации являются ISBN (Международный стандартный номер книги, International Standard Book Identifier) для идентификации книг в бумажном виде и ISSN для идентификации периодических изданий.

В основе системы идентификации изделий конкретного предприятия, как правило, находится универсальная система с использованием серийных номеров (Serial Number), СН. Серийный номер должен отвечать следующим требованиям:

- многофункциональность структуры номера;
- однозначность идентификации;
- устойчивость номера в контексте стабильности структуры во времени;

- возможность расширения;
- относительная простота и машиночитаемость серийного номера.

Структура и содержание серийного номера должны обеспечивать возможность функционирования уже существующих систем и продукции. Также СН должен обеспечивать возможность использования оператором геоинформационных технологий. Информация, приведенная в СН, не должна разглашать сведения, отнесенные в установленном порядке к государственной тайне. Для обеспечения задач идентификации целесообразно, чтобы СН единообразно обозначал данную реализацию (экземпляр) объекта классификации. СН должен использоваться как при ручной, так и при автоматической обработке, содержать информацию, воспринимаемую как человеком, так и автоматическим устройством обработки (машинно-считывающая информация). СН должен включать в свой состав глобально устойчивую и стабильную информацию, а также коды, соответствующие свойствам, характеристикам, отражающим специфику данного экземпляра. Например, серийные номера микросхем могут включать год и месяц производства.

Вопрос о том, каким экземплярам продукции присваивается собственный уникальный идентификатор, является предметом отдельного изучения. В целях обеспечения единства информационного пространства современного производства, можно говорить о необходимости присвоения соответствующего идентификатора каждой детали, сборочной единице, каждому экземпляру выпускаемого изделия. Однотипные экземпляры отличаются как минимум порядковым номером или датой выпуска.

Для формирования СН в целях обеспечения однозначной идентификации экземпляра продукции, который относится к объекту классификации, целесообразно использовать четыре части (позиции) идентификатора:

- условное обозначение устройства (или элемента аппаратуры), данное производителем;
- часть номера производителя;
- версия устройства, указанная производителем;

–серийный номер производителя.

*Часть номера производителя* (Manufacturer Part Number) – характеристическая информационная строка, которая присваивается каждой детали, сборочной единице или изделию его производителем. Этот уникальный номер/код может рассматриваться как способ кодирования, идентифицирующий устройство. Код может содержать основную информацию для определения функциональности устройства. Код наносится любым способом (написан, выгравирован, нанесен в виде штрих-кода) на каждое монтируемое устройство или на оборудование для монтажа; этот же номер, как правило, не обозначает версию, выпуск или релиз.

*Версия устройства, указанная производителем* (Manufacturer Equipment Version) – информация, которая присваивается деталям или сборочным единицам в целом во время сборки производителем. Этот идентификатор указывает на изменения, которые произошли по отношению к исходному оборудованию, изготовленному производителем. Например это относится к процессам сборки и соединения оборудования без существенного изменения электрических, оптических, внешних соединений или физических характеристик.

*Серийный номер производителя* (Manufacturer Serial Number) – случайная комбинация в виде строки символов, имеющая формат, заданный производителем, которая повторно не встречается ни на одном экземпляре объекта классификации и наносится на деталь, сборочную единицу, монтируемое устройство в процессе производства. Серийный номер уникален для данного производителя. Серийный номер должен содержать некоторую дополнительную информацию, связанную с производителем, например дату производства изделия; кроме того, часть информации серийного номера должна изменяться с учетом жизненного цикла изделия, например, после ремонта.

Для электронных документов, программ и контента можно использовать цифровые идентификаторы с помощью методов стеганографии, в частности цифровые отпечатки (Digital Fingerprint), стеганографические водяные знаки (Stego Watermarking). Это

особенно актуально для предотвращения утечек информации DLP (Data Leak Prevention). В последнем случае уже при создании электронного документа, содержащего конфиденциальную информацию, добавляется определенная метка, которая не изменяется, вне зависимости от количества копий и/или ревизий документа.

### Список библиографических ссылок

1. Тоноян Л. Г. Основы теории классификации в трудах Аристотеля// Логико-философские штудии. 2012. № 6. С. 28–36.
2. Субботин А. Л. Классификация // Центр гуманитарных технологий. 10.12.2011. URL: <https://gtmarket.ru/laboratory/basis/3794>.
3. Ископаемые флоры Дальнего Востока : сб. науч. статей. Владивосток : АН СССР Дальневост. науч. центр. Биол.-почв. ин-та, 1975. Т. 27 (130). 136 с.
4. Ивлев Ю. В. Логика : учебник. 4-е изд. перераб. и доп. М. : ТК Велби, Изд-во «Проспект», 2008. 304 с.
5. ПР 50.1.024-2005 Правила стандартизации. Основные положения и порядок проведения работ по разработке, ведению и применению общероссийских классификаторов. М. : Стандартинформ, 2006. 39 с.
6. Палагин А. В., Крывый С. Л., Петренко Н. Г. Онтологические методы и средства обработки предметных знаний. Луганск : Изд-во ВНУ им. В. Даля, 2012. 324 с.
7. Rebstock M., Fendel J., Paulheim H. Ontologies-Based Business Integration. Berlin, Heidelberg : Springer-Verlag, 2008. 268 p.
8. Волкова Г. А. Создание «онтологии всего». Проблемы классификации и решения // Новые информационные технологии в автоматизированных системах : материалы шестнадцатого науч.-практ. семинара. М., 2013. С. 293–300.
9. Ontology-Based automatic classification and ranking for Web documents/ J. Fang, L. Guo, Wang Xiao Dong et al. // Fourth International Conference on Fuzzy Systems and Knowledge Discovery, FSKD 2007. P. 627–631.
10. Funk A., Bontcheva K. Ontology-based categorization of Web Services with machine learning // Proceedings of the International Conference on Language Resources and Evaluation, LREC 2010. 2010. P. 482–488.
11. Ontology-based semantic classification of satellite images: case of major disasters / H. Bouyerbou, K. Bechkoum, N. Benblidia et al. // IEEE Geoscience and Remote Sensing Symposium, IGARSS 2014. 2014. P. 2347–2350.

- 
- 
12. *Dou D., Wang H., Liu H.* Semantic data mining: A survey of ontology-based approaches // Proceedings of the 2015 IEEE 9th International Conference on Semantic Computing (IEEE ICSC 2015). 2015. P. 244–251.
  13. Ontology based machine learning for semantic multiclass classification / F.-E. Calvier, M. Plantié, G. Dray et al. // TOTH: Terminologie and Ontologie: Théories et Applications, 2013, P. 100–117.
  14. *Hoppe T.* Ontology-based classification – application of machine learning concepts without learning // In «Solving Large Scale Learning Tasks: Challenges and Algorithms» / eds. S. Michaelis, N. Piatkowski, M. Stolpe. Lecture Notes in Artificial Intelligence. LNAI 9580, Springer Verlag, 2016. P. 320–33.
  15. Ontology-based classification and analysis of nonemergency Smart-city events / M. Rani, S. Alekh, A. Bhardwaj et al. // International Conference on Computational Techniques in Information and Communication Technologies (ICCTICT). 2016. 6 p.
  16. *Cheng C.K., Pan X.S., Kurfess F.* Ontology-based semantic classification of unstructured documents // Adaptive Multimedia Retrieval, International Workshop on Adaptive Multimedia Retrieval. AMR 2003, Lecture Notes in Computer Science book series (LNCS vol. 3094). 2003. P. 120–131.
  17. *Zillner S.* Towards the Ontology-based classification of Lymphoma patients using semantic image annotations // Proceedings of the Workshop on Semantic Web Applications and Tools for Life Sciences. 2009. 11 p.
  18. *Vogrinčič S., Bosnić Z.* Ontology-based multi-label classification of economic articles // Computer Science and Information Systems. 2011. Vol. 8, Iss. 1. P. 101–119.
  19. Automatic classification of epilepsy types using ontology-based and genetics-based machine learning / Y. Kassuhum, R. Perrone, De E. Momi et al. // Artificial Intelligence in Medicine. Vol. 61, Iss. 2. 2014. P. 79–88.
  20. *Hastings J., Magka D., Batchelor C.* Structure-based classification and ontology in chemistry // Journal of Cheminformatics. 2012. Vol. 4(1). 8 p.
  21. Гребешков А. Ю. Управление и технический учет ресурсов в телекоммуникациях : монография М. : ИРИАС, 2008. 268 с. ISBN 978-5-93592-037-1.
  22. Гребешков А. Ю. Управление конфигурацией и технический учет в телекоммуникациях. Организация, построение и способы организации. Germany: LAP LAMBERT Academic Publishing GmbH & Co., 2012. 428 c. ISBN 978-3-8484-9964-9.

## Глава 5

# ТЕХНОЛОГИИ И МЕТОДИКА ОНТОЛОГИЧЕСКОГО МОДЕЛИРОВАНИЯ

*С. В. Горшков*

## § 1. Практические задачи предприятий и методы их решения

Задачей данной главы является общее описание методов и принципов построения цифровых моделей активов, процессов, субъектов, необходимых для моделирования предприятий, и рассмотрение общих подходов к использованию построенных моделей для получения экономического эффекта. Технологические вопросы представления онтологических моделей по стандарту OWL затрагиваются в объеме, необходимом в связи с их влиянием на методику моделирования.

Любое предприятие представляет собой сложную систему, функционирующую в изменчивом внешнем контексте. Деятельность предприятия подчиняется интересам его владельцев, которые могут заключаться в получении операционной прибыли, наращивании капитализации предприятия, выполнении социальных или имиджевых задач. Тактику реализации этих интересов обычно определяет менеджмент предприятия, который должен трансформировать поставленные цели в программу конкретных действий и обеспечить их выполнение. В общем случае программа мероприятий должна обеспечить достижение целей в наиболее короткий срок с наименьшими затратами. Соотношение количественных оценок полученного результата и произведенных затрат на единицу времени является прямой и естественной метрикой эффективности работы предприятия.

Построение оптимальной программы действий требует от менеджмента принятия множества управлеченческих решений, причем диапазон возможных вариантов в каждой ситуации крайне широк.

Учитывая многообразие факторов, действующих как внутри, так и вне предприятия, непредсказуемый характер многих из них, общего решения задачи построения оптимальной программы действий не существует. Однако многие частные задачи, возникающие перед руководством организаций, могут быть эффективно решены при помощи современных информационных технологий. К их числу относятся задачи минимизации технологических потерь, оптимизации ремонтных программ, приложения адекватных усилий для предотвращения реализации рисков и многие другие. Решение таких задач требует воспроизведения закономерностей, описывающих поведение отдельных элементов предприятия и внешней среды, то есть создания моделей. В настоящее время активно развиваются технологии, условно объединяемые названием «цифровое моделирование». Несмотря на существенные различия, все технологии этого спектра предполагают создание «цифровых образов» объектов и субъектов, отражающих их существенные свойства и закономерности поведения в виде набора значений численных или логических формализованных параметров и правил.

Существует и широкий спектр не-экономических задач, для эффективного решения которых также требуется структурирование и обработка огромных объемов данных, описывающих структуру объектов, событий и их взаимосвязей. Примерами таких задач является организация деятельности оперативных служб по предупреждению и ликвидации последствий чрезвычайных ситуаций и происшествий, решение социальных и научно-исследовательских проблем.

Можно выделить два основных принципа создания моделей: аналитическое моделирование, когда свойства объектов и правила их взаимодействия выявляются в результате анализа и формализуются в виде алгоритмов или формул, и эмпирическое, когда не поддающиеся анализу закономерности воспроизводятся таким образом, чтобы достичь требуемой вероятности совпадения результатов моделирования с фактическим поведением реальной системы. Если моделируемую систему или объект представить как «черный ящик», то аналитические методы заключаются в воспроизведении с той или иной

степенью детальности его внутреннего механизма. Эмпирические методы состоят в предугадывании выходных параметров «черного ящика» (например, состояния системы) в ответ на определенные входные параметры на основе неточных методов, не воспроизводящих его внутренние закономерности. Каждый из этих способов имеет свои преимущества, ограничения и сферу применения.

При моделировании систем часто используются методы имитационного моделирования. В этом случае модели отдельных объектов (как аналитические, так и эмпирические) объединяются в ансамбли. Модели элементов системы, взаимодействующие между собой в рамках ансамбля, позволяют смоделировать поведение системы в целом.

Практически любая модель, построенная для решения прикладных задач, включает:

- модели объектов (в частности, в корпоративных приложениях часто речь идет о модели активов), в том числе модели окружающей среды;
- модели субъектов – акторов, обладающих собственными интересами и возможностями их осуществления;
- модели активности, то есть изменений, происходящих в рассматриваемой части реальности.

Для решения специального круга задач строятся также модели ценностных установок и оценок, суждений субъектов и др.

Онтологическое моделирование – набор методик и технологий создания моделей, заключающихся в формализации логических описаний предметных областей и их машинной обработке. Методики онтологического моделирования основываются на создании концептуальной модели, то есть формального описания моделируемой части реальности при помощи определенного терминологического аппарата. Далее модель преобразуется в машинно-читаемую форму на основе стандартов, принятых консорциумом W3C: формализмов моделирования RDF/RDFS/OWL, языка запросов к онтологическим моделям SPARQL, правил логического вывода SPIN и SWRL и др.

Далее мы рассмотрим подходы к созданию моделей объектов, субъектов и активностей с использованием инструментария онтологического моделирования, а также приведем практические примеры построения фрагментов онтологических моделей.

## § 2. Моделирование объектов

Создание модели объекта начинается с выделения объекта из окружающего мира. Выделение объекта должно быть обусловлено некой прагматикой, то есть обосновано необходимостью рассмотрения этого объекта в целях решения функциональной задачи. Приняв решение о создании модели объекта, необходимо определить его границы в пространстве и времени.

Моделирование пространственных аспектов существования физического объекта требует решения ряда вопросов. Каким образом зафиксировать нахождение объекта в пространстве? Можно указать его координаты или положение относительно других объектов. В первом случае необходимо определить систему координат, выбрать размерность описания границ объекта (точка, контур, объем), а во втором случае – ввести понятия, описывающие взаимное положение тел.

Если попытаться решить эти вопросы не интуитивно, а обосновать выбранный вариант моделирования, возникнет множество более конкретных вопросов. Нужно ли вообще описывать границы объекта или достаточно описать местонахождение и пренебречь его размерами? Могут ли другие объекты одновременно присутствовать в том же объеме пространства? Такое возможно, например, для модели, содержащей описание системы и ее составных частей (каждый объект – часть системы находится внутри объекта, представляющего систему как целое). В других случаях, таких как решение логистических задач или создание систем управления технологической инфраструктурой, возможность пространственного совпадения разных объектов может быть ограничена.

Необходимо определить границы существования объекта и во времени. Что считать моментами его возникновения

и исчезновения? В чем состоит та общность, которая сопровождает объект на протяжении его существования и заставляет считать его одним и тем же объектом?

Не существует универсальных рецептов, позволяющих найти «верные» ответы на все поставленные вопросы. В рамках задачи построения цифровой модели предприятия в каждом случае необходимо их решать, исходя из pragmatики, из того, позволит ли выбранный способ моделирования достичь целей наиболее рациональным образом.

В процессе решения подобных задач внимательному аналитику легко перейти к размышлению о природе человеческого восприятия и познания, оценить их сквозь призму тех или иных философских концепций. Действительно, поскольку создаваемые онтологические модели являются отражением образов, возникающих в мышлении аналитика, любая модель целиком является продуктом человеческого разума и не может претендовать на «правильность» в смысле «объективного» отражения реальности (мы берем эти термины в кавычки, потому что не существует заведомо верного способа отражения реальности в модели или метода оценки ее «объективности». Метрики, описывающие степень адекватности модели решаемой задаче, могут основываться на оценке отклонения картины, получаемой в результате моделирования, от картины, наблюдаемой в реальном мире в результате фактического протекания моделируемого процесса. Такие метрики неизбежно несут в себе отпечаток процессов преобразования и упрощения информации, протекающих в ходе восприятия оценивающим субъектом как результатов моделирования, так и наблюдения за процессом в реальном мире). Хотя в повседневном общении люди без труда выделяют объекты из окружающей среды, называют их и взаимодействуют с ними, необходимо помнить о том, что разные субъекты могут иметь разные точки зрения на выделение объектов, их типизацию и именование, и часто нельзя однозначно определить «единственно правильную» точку зрения даже в рамках контекста определенной задачи.

Любая модель строится на основе терминологического аппарата, то есть зависит от языка. С помощью слов конструируются высказывания, также являющиеся отражением мыслительных процессов и используемых в них образов. Любое высказывание на естественном языке несет в себе существенные элементы неопределенности и упрощения. Вместе с тем получить практически пригодные результаты моделирования можно только зафиксировав определенный терминологический аппарат и постулировав истинность некоторых высказываний.

В процессе моделирования объекта, после определения оснований его включения в модель и описания пространственно-временных границ, наступает стадия типизации и именования. Категория «тип» естественна для нашего сознания и является одним из основополагающих мыслительных механизмов, позволяющих ориентироваться в окружающем мире и быстро принимать решения. Относя каждый конкретный объект к какому-либо типу, человек тем самым распространяет на него ту информацию, которая известна ему обо всех объектах такого типа, и принимает решения в соответствии с этим. Увидев змею, человек решает: «Это – змея. Змеи ядовиты. Необходимо убежать от нее или защититься». Такой способ реакции упрощен (не все змеи ядовиты), но эффективен, поскольку если человек начнет разбираться в том, ядовита ли змея – с большой вероятностью он попадет в беду.

Итак, выделив объект из окружающего мира, человек в первую очередь стремится отнести его к одному или нескольким типам, а затем – как-то обозначить конкретный объект, чтобы отличать его от других аналогичных объектов. Эти процессы обычно протекают в подсознании и не требуют логических рассуждений. Однако при создании модели необходимо выполнять их осознанно, поскольку любая модель предназначена для использования многими людьми. Необходимо обеспечить им возможность «прочитать» модель и соотнести ее элементы с теми объектами моделируемой части реальности, которые выделят другие участники работы с моделью.

### § 3. Основные возможности стандартов моделирования RDF/RDFS/OWL

Для дальнейшего рассмотрения методологических вопросов моделирования необходимо познакомиться с основами стандартов онтологического моделирования. Получить о них более глубокую информацию можно в текстах самих стандартов, а также в специальной литературе [1–3].

В стандарте представления онтологических моделей OWL определены несколько «корневых» типов сущностей, среди которых для нас важны класс (`owl:Class`) и индивидуальный объект (`owl:NamedIndividual`). Сгруппировав ряд объектов и отнеся их к какому-либо типу, аналитик может создать класс OWL для представления этого типа в модели. Так, можно создать класс «Змея» для того, чтобы объединить все объекты, представляющие конкретных змей, или класс «Трансформатор» для того, чтобы объединить устройства трансформации электроэнергии. Конкретные объекты, выделенные аналитиком в рассматриваемом фрагменте реальности, моделируются при помощи индивидуальных объектов. Каждый индивидуальный объект может входить в один или несколько классов.

Основной структурной единицей моделей, построенных в соответствии со стандартом OWL и предшествующих ему стандартов RDF, RDFS, является триплет. Триплет – это элементарное высказывание, состоящее из трех элементов: субъекта, предиката и объекта (или, в терминах грамматики русского языка, подлежащего, сказуемого и дополнения), например: «змей является животным». На каждой позиции в таком высказывании могут находиться идентификаторы объектов или литералы: строковые, числовые, булевые выражения.

Стандарты «семантической сети» (Semantic Web), к числу которых относится OWL, используют универсальные идентификаторы ресурсов (URI) в качестве идентификаторов объектов. Ряд таких идентификаторов определен самими

стандартами. Например, тип сущности «класс» имеет идентификатор <http://www.w3.org/2002/07/owl#Class>, а тип сущности «индивидуальный объект» – идентификатор <http://www.w3.org/2002/07/owl#NamedIndividual>. Общая часть любых идентификаторов может быть определена как «префикс» (понятие, определенное в спецификациях Semantic web). Означает часть URI, общую для определенного набора ресурсов). Например, стандарт OWL определяет префикс owl как <http://www.w3.org/2002/07/owl#>. В результате замены полных префиксов на их краткие представления образуются сокращенные записи идентификаторов, такие как `owl:Class`.

Аналитик-автор онтологии может использовать собственное пространство идентификаторов, либо расширить одно из пространств стандартных онтологий, если это уместно в контексте создаваемой модели. Для использования в дальнейших примерах мы определим префикс <http://onto.pro/model> как `model`.

Теперь мы готовы к записи первых высказываний. Существует множество синтаксисов их записи, среди которых наиболее наглядны представления в виде триплетов и в виде предикатов. Высказывание, смысл которого состоит в том, что существует тип объектов «трансформаторы», записывается в виде следующего триплета:

`model:Трансформатор rdf:type owl:Class`

В этом выражении `model:` – Трансформатор – это идентификатор класса «Трансформатор», введенный автором онтологии; `rdf:type` – предикат из стандарта RDF, означающий отнесение сущности к некому типу; `owl:Class` – уже упоминавшийся нами тип «класс», определенный в стандарте OWL.

То же самое выражение можно записать в виде бинарного предиката:

`rdf:type (model:Трансформатор, owl:Class)`

Идентификаторы объектов модели обычно не демонстрируются пользователям автоматизированных систем, использующих онтологии. Вместо них пользователи видят читаемые названия,

для определения которых используется `rdfs:label` – предикат из стандарта RDFS. Задать читаемое название для объекта онтологии можно при помощи следующего триплета:

```
model:Трансформатор rdfs:label "Трансформатор"^^xsd:string
```

В третьей позиции этого триплета мы видим литеральное значение «Трансформатор», для которого указан строковый тип `xsd:string`.

Классы могут образовывать иерархии. Об отношении «надкласс–подкласс» говорят в тех случаях, когда все объекты одного класса являются одновременно и объектами другого. Например, все змеи являются животными – значит, можно утверждать, что класс «Змея» является подклассом класса «Животное». Такое отношение между классами задается при помощи предиката `rdfs:subClassOf`. Например, выделение подкласса «Силовой трансформатор» в классе «Трансформатор» выражается следующим образом:

```
model:СиловойТрансформатор rdfs:subClassOf model:Трансформатор
```

Если аналитик хочет выделить конкретный трансформатор и отнести его к описанному выше типу, он должен задать следующий набор триплетов:

```
model:актив12345 rdf:type owl:NamedIndividual  
model:актив12345 rdf:type model:Трансформатор
```

В этих выражениях `model:актив12345` – идентификатор конкретного объекта, отнесенного аналитиком к классу «Трансформатор». Идентификаторы могут присваиваться произвольным образом, единственным требованием к ним является уникальность. Из приведенного выше примера видно, что отнесение объекта к стандартному типу `NamedIndividual` и определенному аналитиком типу «Трансформатор» осуществляется при помощи одного и того же предиката `rdf:type`, а также что каждый объект может относиться одновременно к нескольким типам.

Разумеется, на практике аналитик редко работает с онтологическими моделями на уровне триплетов – все указанные операции

выполняются в редакторах онтологий, обзору которых посвящен один из следующих разделов этой книги. Тем не менее аналитику необходимо понимать структуру, в соответствии с которой представляется построенная им модель.

Подчеркнем важную деталь, касающуюся построения иерархий классов при помощи предиката `rdfs:subClassOf`. Хотя практически все программные средства моделирования онтологий отображают иерархии классов в виде дерева, некорректно рассматривать их по аналогии с папками в файловой системе. Отношение «надкласс–подкласс» подразумевает, что каждый объект подкласса является одновременно и объектом надкласса. Так, каждый объект подкласса «Человек» является одновременно объектом надкласса «Животное». Из этого следует, как мы увидим далее, что человек может обладать значениями всех свойств, применимых для описания животного. Корректной визуальной аналогией для иерархий классов (таксономий) является не дерево, а так называемые «круги Эйлера» (рис. 5.1).



Рис. 5.1. Отношения «надкласс–подкласс»

Для того чтобы описать характеристики объектов и их взаимодействие друг с другом, необходим функционал определения свойств объектов. Стандарты онтологического моделирования определяют базовый тип `owl:Property`, подклассами которого являются `owl:DatatypeProperty` и `owl:ObjectProperty`.

Первый из них предназначен для определения свойств, значениями которых являются литералы, второй – для указания связей между объектами.

Для каждого свойства можно определить набор классов, к объектам которых они применимы (предикат `rdfs:domain`) и набор классов или типов, объекты которых могут быть значениями свойств (`rdfs:range`). Например, пусть существует свойство «Дата ввода в эксплуатацию» с типом значения «дата» (используются типы значений XSD, то есть для выражения дат пригоден тип `xsd:date`), применимое к объектам класса «Трансформатор»:

```
model:ДатаВводаВЭксплуатацию rdf:type  
owl:DatatypeProperty  
  
model:ДатаВводаВЭксплуатацию rdfs:label “Дата ввода  
в эксплуатацию”^^xsd:string  
  
model:ДатаВводаВЭксплуатацию rdfs:domain  
model:Трансформатор  
  
model:ДатаВводаВЭксплуатацию rdfs:range xsd:date
```

Если свойство предназначено для указания связи между двумя объектами, его необходимо отнести к типу `owl:ObjectProperty`. Например, объявление свойства «Установлен на подстанции», которым могут обладать объекты класса «Трансформатор», диапазоном значений для которого являются объекты класса «Подстанция» (присвоим ему идентификатор `model:ПС`), может быть таким:

```
model:УстановленНаПс rdf:type owl:ObjectProperty  
  
model:УстановленНаПс rdfs:label “Установлен  
на подстанции”^^xsd:string  
  
model:УстановленНаПс rdfs:domain model:Трансформатор  
  
model:УстановленНаПс rdfs:range model:ПС
```

Отметим несколько важных технологических и методологических особенностей:

1. Любое свойство каждого объекта может иметь несколько значений, если это не запрещено его определением (минимальное и максимальное число значений свойства определяется предикатами `owl:minCardinality` и `owl:maxCardinality`). Например, можно указать два варианта читаемого названия для класса:

```
model:Tрансформатор rdfs:label  
“Трансформатор”^^xsd:string
```

```
model:Tрансформатор rdfs:label “Электрический  
трансформатор”^^xsd:string
```

2. И область применимости, и диапазон значений могут содержать несколько классов или типов XSD. Для указания таких множеств не достаточно просто указать два значения свойств `rdfs:domain` и `rdfs:range` – необходимо использовать механизм задания списков `rdf:List`, описание деталей которого выходит за рамки данной книги.
3. Определение областей применимости и диапазонов значений для свойств работает не совсем очевидным образом: с точки зрения машин логического вывода, оно не ограничивает возможность присваивать свойства объектам тех или иных классов, а, наоборот, наличие значения свойства у объекта может являться основанием для вывода о том, что он относится к тому или иному классу.

Стандарты RDF/RDFS/OWL предлагают ряд возможностей для расширенного моделирования объектов, классов и свойств, в том числе:

1. Можно определять характеристики свойств и отношений между свойствами, такие как транзитивность, инверсность и др. Например, свойство является транзитивным, если наличие связи между А и В, между В и С означает и существование такой же связи между А и С. Так, свойство «является частью» транзитивно, поскольку если А является частью В, а В является частью С, то А является частью С. Свойства *a* и *b*

являются инверсными, если наличие связи *a* между объектами А и В означает наличие связи *b* между В и А. Например, свойства «является ребенком» и «является родителем» инверсны, так как если А является ребенком В, то В является родителем А. Характеристики свойств и их применение детально рассматриваются в *Приложении*.

2. Свойства могут образовывать иерархии так же, как классы. Например, свойство «является сыном» – под-свойство для свойства «является ребенком».
3. Могут описываться отношения между классами, кроме отношений «надкласс – подкласс». Например, в модели можно постулировать, что классы А и В разъединены, то есть ни один объект класса А не является объектом класса В. Один класс может быть дополнением другого и др. Подобные отношения можно задавать и для комбинаций классов – их объединений или пересечений.
4. Классы могут определяться явным перечислением входящих в них объектов.

Эти и другие возможности, полное перечисление которых выходит за рамки задач нашей книги и может быть найдено в тексте стандартов и специальной литературе (краткий и понятный обзор функциональных возможностей стандарта OWL и правил логического вывода можно найти в [4]), предназначены прежде всего для контроля консистентности создаваемых моделей. Вводимые в модель данные не должны противоречить заданным утверждениям о классах и свойствах, в противном случае машина логического вывода выдаст ошибку. Еще одно назначение перечисленных возможностей – автоматическое дополнение известных фактов на основе правил логического вывода. Это можно проиллюстрировать на примере классического сyllogizma: если определен класс «Человек» как подкласс класса «Смертные существа», и определен объект «Сократ», входящий в класс «Человек», то запрос о том, является ли Сократ смертным существом, вернет положительный результат. Обратим внимание на то, что в этом примере свойство

объекта (смертность) задано через определение класса. Вопросы выбора способа указания свойств объектов и групп объектов рассматриваются далее.

Кроме базовых правил логического вывода, определенных стандартами, пользователь может задавать и собственные правила логического вывода – для этого предназначены рассматриваемые далее в данной главе нотации SWRL, SPIN и др. Поддерживающие эти нотации машины логического вывода могут автоматически дополнять содержимое графа (онтологии) новой информацией на основе введенных в модель фактов и определенных для нее правил.

## § 4. Моделирование свойств и отношений объектов. Моделирование физических объектов и технической инфраструктуры

В начале данной главы мы рассмотрели вопросы выделения объектов, их типизации и идентификации. Стандарт OWL предоставляет механизмы для определения типов объектов (`owl:Class`), описания индивидуальных объектов (`owl:NamedIndividual`), задания принадлежности объектов определенному типу (предикат `rdf:type`). Обратимся теперь к методическим вопросам моделирования свойств объектов, соотнося их с возможностями стандарта OWL.

Рассматривая объект, человек выделяет и анализирует его свойства: форму, цвет, размер, пространственное положение и др. Очевидно, что значения свойств могут быть выражены при помощи разных шкал. Например, размер может быть охарактеризован сочетанием числа и единицы измерения, определяющими число единиц измерения для какой-либо размерности объекта (например, диаметр = 0,5 м), сравнением с другими объектами (больше человека) или просто качественными характеристиками (большой). Цвет может быть выражен как определенным понятием (синий), так и числовым выражением, определяющим координаты в некотором цветовом пространстве (RGB #0000ff). Пространственное положение, как мы говорили выше, может быть задано значениями

координат в определенной координатной системе или относительно других объектов.

Прагматика использования модели определяет предпочтительный способ выражения характеристик объекта. Так, если в рамках определенной онтологии принято, что линейные размеры объектов всегда выражаются в метрах, и есть гарантия того, что на протяжении жизненного цикла модели не возникнет необходимости использовать другие единицы измерения, можно определить свойство

```
model:ИмеетРазмер rdf:type owl:DatatypeProperty
```

как имеющее

```
model:ИмеетРазмер rdfs:range xsd:double
```

и задавать значения этого свойства для конкретных объектов таким образом:

```
model:актив12345 model:ИмеетРазмер "0.5"^^xsd:double
```

Если требуется задавать значения одного свойства в разных единицах измерения, допустимым (хотя и не очень распространенным) вариантом может являться использование под-свойств. Например, для свойства `model:ИмеетРазмер` можно определить под-свойство `model:ИмеетРазмерВМетрах` с диапазоном значений `xsd:double` и под-свойство `model:ИмеетОтносительныйРазмер` с диапазоном значений из замкнутого класса-перечисления [`model:Малый`, `model:Средний`, `model:Большой`].

Если функциональные требования к модели вынуждают выбрать более общий и точный способ указания размеров, можно применить реификацию (reification) – прием, состоящий в разворачивании ребра графа (то есть триплета, сообщающего о значении свойства определенного объекта) в самостоятельный узел, отдельный объект с собственным идентификатором. Это позволит, например, сохранить вместе со значением свойства его единицу измерения.

Итак, чтобы реифицировать значение свойства объекта, в нашем примере необходимо объявить класс

**model:РазмерОбъекта** (который станет диапазоном значений для свойства **model:ИмеетРазмер**). Объекты этого класса могут обладать значениями свойств **model:ЗначениеРазмера** и **model:ШкалаРазмера**, первое из которых содержит числовое значение размера, а второе – идентификатор объекта, представляющего единицу измерения. Тогда приведенный выше триплет с сообщением о том, что объект с идентификатором **model:актив12345** имеет размер 0,5 м, развернется в следующую последовательность триплетов:

```
model:актив12345 model:ИмеетРазмер model:размер12345
model:размер12345 rdf:type owl:NamedIndividual
model:размер12345 rdf:type model:РазмерОбъекта
model:размер12345 model:ЗначениеРазмера
"0.5"^^xsd:double
model:размер12345 model:ШкалаРазмера model:Метр
```

В данной последовательности триплетов создается индивидуальный объект класса **model:РазмерОбъекта** с идентификатором **model:размер12345**, имеющий значения свойств **model:ЗначениеРазмера** = 0.5 и **model:ШкалаРазмера** = **model:Метр** (идентификатор единицы измерения «метр», который должен быть объявлен где-то в онтологии; можно использовать и внешнее определение метра, сославшись на одну из стандартных онтологий). Свойство **model:ИмеетРазмер** объекта **model:актив12345** теперь ссылается на объект **model:размер12345**, хотя можно создать связь и в обратную сторону – это не имеет принципиального значения, важно лишь то, как удобнее будет работать с этой информацией при помощи запросов к модели.

Такой способ моделирования позволяет решить еще одну проблему, обозначенную в начале раздела – моделирование эволюции объектов во времени. Если объект имеет размер 0,5 м не все время своего существования, и этот факт нужно отразить в модели (то есть модель должна отражать не снимок мгновенного

состояния системы, а картину ее эволюции во времени), то целесообразно рассматривать обладание значением свойства как состояние объекта. Состояние характеризуется моментами его возникновения и прекращения. Соответственно, можно объявить класс «Состояние» (например, `model:Состояние`, или взять определение состояния из одной из стандартных онтологий) и породить от него подклассы, среди которых будут такие, как «Обладание значением свойства», или более конкретно – «Обладание значением размера». В качестве области применимости для свойств «Дата возникновения состояния» и «Дата прекращения состояния» следует выбрать класс «Состояние». Это значит, что и объекты всех вложенных в него классов могут обладать значениями этих свойств.

Один из возможных подходов к моделированию состояний заключается в том, чтобы рассматривать каждую фазу состояния объекта как самостоятельный объект в пространстве-времени. Это особенно удобно в том случае, когда на каждой фазе своего существования объект входит в разные классы (выбор между классификацией и присвоением значений свойств для выражения информации об объекте обсуждается далее). В таком случае уместно создать набор индивидуальных объектов, каждый из которых будет представлять «временную часть» общего объекта и будет иметь ссылку на него. Каждая «временная часть» описывает набор характеристик объекта на определенном отрезке его существования, благодаря чему может быть интерпретирована как состояние объекта. Общность и непрерывность существования объекта отражаются в модели при помощи общего объекта, на который ссылаются все временные части и который хранит характеристики объекта, неизменные на всем протяжении его существования. Этот и другие полезные паттерны моделирования подробно обсуждаются в работе [5].

Описанные приемы моделирования являются достаточно распространенными, и в ряде стандартных онтологий верхнего уровня существуют инструменты для их реализации средствами семантических технологий. Так, онтология DOLCE содержит класс

**State**, являющийся подклассом класса **Perdurant**, предназначенный для отражения состояний. Для представления в виде объектов значений свойств, соответствующих физическим характеристикам, в ней объявлен класс **Physical Quality**, являющийся подклассом **Quality** [1]. Использование подобных онтологий верхнего уровня снижает риск ошибок моделирования за счет использования готовых определений и ограничений, однако может в некоторых случаях увеличивать сложность модели и приводить к росту вычислительной нагрузки при работе с ней.

Особое внимание следует уделить моделированию отношений объектов. Как было показано выше, отношение «является родителем – является ребенком» можно смоделировать при помощи одного или двух ребер графа, представляющих значения свойств, относящихся к типу **owl:ObjectProperty**. Можно применить реификацию аналогично описанному выше примеру, если отношение действительно не на всем протяжении существования объектов (как, например, отношение «владеть» между субъектом и собственностью). В этом случае отношение рассматривается как самостоятельный объект, а его связи с участниками отношений уместно рассматривать как роли. Так, в отношении «владеТЬ» участвуют два субъекта/объекта, один из которых выступает в роли владельца, а другой – в роли предмета владения. Этот и другие полезные паттерны моделирования на языке формальной логики, легко транслируемом в OWL, рассматриваются в основополагающей работе [6].

Для отношений также можно построить иерархию, верхние уровни которой будут описывать такие крупные классы отношений, как отношения «часть-целое» (отношения объекта и его составляющих), зависимость между отдельными объектами или отношения основного объекта со вспомогательными, описывающими его свойства. Для отношений разных классов будут определены разные типы ролей участвующих в них объектов, которые будут конкретизироваться от верхних классов иерархии отношений к нижним.

Приведем пример моделирования отношений объектов в рамках структурно-функциональной модели технической инфраструктуры. Термин «структурно-функциональная модель» подразумевает, что в модели отражаются два аспекта взаимосвязей объектов: структурные отношения, то есть вхождение одних объектов в состав других и их пространственные взаимосвязи, и функциональные отношения – зависимость исполнения функций одними объектами от работы других. Как правило, именно такие модели используются для решения широкого круга практических задач предприятий, от мониторинга состояния инфраструктуры до планирования ее оптимизации.

Пусть имеется технологическая схема, включающая трансформатор напряжения, соединенный с одной стороны с источником электропитания, а с другой – с потребителем. Эта схема описывает функциональную структуру оборудования, то есть представляет собой как бы «чертеж» или шаблон, в соответствии с которым должно быть установлено и подключено конкретное оборудование. Будем называть элементы этого шаблона «техническими местами». Конкретные устройства, устанавливаемые на технические места, будем называть «единицами оборудования». В разные периоды времени на технических местах могут быть установлены различные единицы оборудования, однако влияние таких изменений на функционирование технологической схемы в целом может и не рассматриваться. Описание изменений в схеме отложим до раздела, посвященного построению моделей активности.

Пусть на техническом месте, обозначенном на схеме как «Трансформатор Т1», в период с 01.01.2014 по 31.10.2018 был установлен трансформатор с серийным номером П12345, а начиная с 01.11.2018 его заменили на трансформатор Р67890. В модели нужно создать индивидуальные объекты, представляющие сведения как о техническом месте «Трансформатор Т1», так и о единицах оборудования П12345 и Р67890. Нужно также распределить между ними значения свойств. Далее необходимо создать объекты, описывающие факты присутствия единиц оборудования на техническом месте в разные периоды времени.

Напряжения первичной и вторичной обмоток трансформатора являются, скорее, свойствами технического места, поскольку в рамках одной технологической схемы нельзя заменить один трансформатор на другой с отличающимися характеристиками. Серийный и инвентарный номера, дата последнего освидетельствования, наоборот, будут свойствами единицы оборудования. Сведения о соединении трансформатора с источником и потребителем электропитания являются свойствами технического места, тогда как связь с контрагентом-производителем – свойством единицы оборудования.

Приведем диаграммы упрощенных фрагментов описанной модели на уровне индивидуальных объектов (рис. 5.2).

Структурно-функциональная схема технических мест



Модель единиц оборудования на техническом месте



Рис. 5.2. Диаграммы фрагментов структурно-функциональной модели

Как видно на рисунке, такой способ моделирования позволяет описывать перемещение единиц оборудования между различными техническими местами. Можно отследить как всю историю использования конкретной единицы оборудования, от ее поступления на склад до вывода из эксплуатации, так и историю замен оборудования в конкретном месте технологической схемы.

В зависимости от решаемой функциональной задачи необходимо рассматривать только технические места или места с установленными единицами оборудования. Так, для прогнозирования последствий отказа узла достаточно функциональной схемы, описанной на уровне взаимосвязей технических мест (которая позволяет установить, например, что работа потребителя П1 зависит от источника питания И1 и трансформатора Т1), а для предсказания отказов потребуются и сведения о конкретных единицах оборудования, включающие срок их эксплуатации, режимы работы и др.

Моделирование технологической инфраструктуры является распространенной, но далеко не единственной задачей в сфере моделирования объектов и их отношений. Интересным классом подобных задач является моделирование сплошных сред, необходимое для обработки геолого-геофизической информации в целях оценки объема запасов, составления оптимальных программ исследования и разработки недр. Методика моделирования геологических структур и использования таких моделей рассматривается в работе [7].

## § 5. Тип или свойство объекта?

Из предыдущего обсуждения следует, что одни и те же факты можно представлять в онтологической модели разными способами. Это относится и к отражению факта принадлежности объекта к какому-либо типу. Мы уже видели, что включение объекта в класс выражается при помощи предиката `rdf:type`, то есть с технической точки зрения (на уровне триплетов) включение объекта в класс эквивалентно присвоению свойству `rdf:type` этого объекта значения, равного идентификатору класса. С другой стороны, практически любое свойство объекта можно трансформировать в тип. Вместо того чтобы сообщать о том, что цвет данного объекта синий, можно сделать семантически эквивалентное утверждение о том, что этот объект относится к классу синих объектов. Подчеркнем, что для человека, читающего эти утверждения, они равнозначны с точки зрения ценности и объема получаемой информации.

Однако с точки зрения управления моделью и выполнения запросов к ней, разница все-таки есть. Допустим, в модели объявлен класс-перечисление «Цвет», объекты которого представляют цвета: синий, красный, зеленый и др. С помощью этого перечисления в приложении легко отобразить палитру, легко выбрать все объекты указанного пользователем цвета и др. Если вместо этого в модели будет создан набор классов, таких как «Синий объект», решение этой задачи с точки зрения программиста сильно усложнится. Следовательно, нужно выбирать в качестве оснований классификации действительно существенные признаки. Интуитивно кажется, что одни признаки более «естественно» выражаются при помощи классификации, а другие – путем присвоения значений атрибутам; можно ли вывести точный принцип выбора одного или другого решения?

Мы предлагаем простой критерий для выбора между представлением характеристики объекта как класса или индивидуального объекта. Если значение характеристики, которую необходимо присвоить, может быть уточнено (конкретизировано) или расширено, то есть спектр значений образует иерархию – есть смысл представить ее как иерархию классов и включить объект в соответствующий класс. Например, пусть необходимо указать, к какому виду учреждений относится объект «кафе “Незабудка”». Кафе является разновидностью учреждений общественного питания, которые в свою очередь являются подмножеством коммерческих учреждений. Значит, в данном случае оправданно создание иерархии классов и включение каждого индивидуального объекта в один из них.

Вторым основанием для того, чтобы предпочесть классификацию присвоению значения свойства, является появление у объекта возможных дополнительных свойств в результате того, что он обладает рассматриваемой характеристикой. Так, у кафе и ресторанов может существовать характеристика «Вид национальной кухни», у учреждений общественного питания – характеристика «Средний чек», а у коммерческих учреждений – «Часы работы». Конкретное кафе может обладать значениями всех перечисленных

характеристик как относящихся непосредственно к классу «Кафе», так и к его надклассам. Такая ситуация также является основанием для того, чтобы выразить информацию о типе объекта при путем его включения в класс.

Если ни один из указанных критериев не выполнен, то, скорее всего, следует использовать присвоение значения свойству конкретных объектов.

## § 6. Отражение в модели точек зрения разных субъектов

Онтологическая модель является отражением представлений субъекта или группы субъектов, разделяющих общую точку зрения на определенный фрагмент реальности. Как поступить в случае, если с моделью должны работать несколько групп пользователей, имеющих разные точки зрения, не сводимые к единой картине мира?

Подобные ситуации часто встречаются в бизнес-контексте. Люди разных профессий используют разный терминологический аппарат: один бизнес-пользователь может рассматривать промышленный комплекс как экономический актив, другой – как опасный производственный объект, третий – как объект бухгалтерского учета. Каждый из этих пользователей по-своему определяет границы объекта в пространстве-времени, по-разному декомпозирует его, наделяет объект разными свойствами. Вместе с тем общность данного объекта для всех пользователей несомненна и должна быть отражена в модели. В обычной корпоративной среде каждая группа пользователей работает с прикладным программным обеспечением, поддерживающим ту и только ту точку зрения, которая необходима именно им, а вопросы передачи информации между точками зрения решаются на уровне интеграционных механизмов и аналитических систем. При построении онтологической модели, как правило, целесообразно совместить несколько представлений в рамках одной модели.

Можно предложить несколько методических приемов для отражения в модели разных точек зрения на одни и те же объекты.

Первый вариант состоит в использовании приема множественной классификации, то есть отнесения каждой сущности сразу к нескольким классам. Например, описанный выше пример с предприятием может быть решен так: созданы классы «Предприятие» и «Опасный производственный объект», введены соответствующие свойства, например, «Численность персонала» для класса «Предприятие» и «Класс опасности» для класса «Опасный производственный объект». Далее можно создать индивидуальный объект, представляющий образ предприятия в онтологической модели, включить его в оба указанных класса и присвоить значения обоим свойствам.

Такой метод, однако, имеет свои ограничения: например, не учтен тот факт, что границы существования объекта как предприятия и как опасного производственного объекта могут отличаться; точки зрения практически не разделены между собой, и неизвестно, какая из них соответствует какой группе пользователей.

Другой вариант отражения множественных точек зрения состоит в создании отдельных объектов модели, каждый из которых представляет специфическую ипостась объекта, и одного объекта, выражающего сведения, общие для всех точек зрения. Такой прием аналогичен ранее описанному паттерну моделирования «временных частей» объектов. В этом случае каждый объект-часть может относиться к собственному набору классов, иметь собственные пространственно-временные границы, а также явно указанную связь с объектом модели, представляющим группу пользователей, чьей точке зрения соответствует данный объект. Недостатком метода является увеличение вычислительной сложности за счет роста числа объектов, и, следовательно, усложнения запросов для доступа к ней.

Третий вариант состоит в использовании именованных графов – функционала, позволяющего делить содержимое онтологии на несколько четко разделенных частей, элементы которых могут быть связаны между собой. Детали реализации этого варианта описаны в статье [8].

В целом следует отметить, что возможность совмещать в модели разные точки зрения, явно подчеркивая точки их соприкосновения и различия, является одним из существенных преимуществ онтологического моделирования перед другими способами представления информации. Как правило, отражение разных точек зрения позволяет реализовать важные преимущества автоматизированной системы для функциональных пользователей. В связи с этим целесообразно при построении модели уделять внимание выявлению и описанию точек зрения различных групп ее будущих пользователей, а не стремиться привести их все к единому представлению.

## **§ 7. Моделирование активностей. Отражение в модели бизнес-процессов, их участников и артефактов**

Методические и технологические приемы, обзор которых дан выше, предоставляют достаточный инструментарий для построения моделей, или «цифровых образов» объектов. В буквальном смысле «цифровым образом» можно назвать любое представление информации о каком-либо объекте в электронной форме – от его текстового описания или фотографии, до различных записей о нем в базе данных. Однако это словосочетание употребляется, как правило, для того, чтобы подчеркнуть полноту образа, необходимую для его использования в контексте среды моделирования неких процессов. Подразумевается, что «цифровой образ» должен содержать описание существенных свойств и поведения объекта, при помощи которых можно имитировать в электронной среде его взаимодействие с другими объектами и окружающей средой. Чтобы перейти от моделирования объектов и их комбинаций к моделированию процессов, протекающих во времени, необходим дополнительный методический инструментарий.

Прежде всего необходимо подчеркнуть отличие онтологического подхода к моделированию активностей от традиционных средств моделирования бизнес-процессов (BPMN и многие другие

методологии и нотации). Традиционные средства моделируют некий шаблон, в соответствии с которым должны протекать конкретные экземпляры процессов, например сценарий работы оператора колл-центра, в соответствии с которым он совершает тысячи конкретных звонков. Однако на уровне терминологического аппарата таких методологий различие между шаблоном и конкретным экземпляром процесса не всегда осознается и редко акцентируется. Далее, само понятие «процесса» неявно подразумевает наличие акторов, которые выполняют определенные действия, руководствуясь некоторыми интересами. В более широком контексте уместнее говорить о моделировании активностей: этот термин не подразумевает наличия сознательно действующих сил и пригоден для обозначения любых разворачивающихся во времени изменений (например, «вулканская активность»). Заметим также, что любой человеческий язык неявно придает субъектность источникам активностей («солнце светит»), что тоже может вносить искажения и субъективность интерпретации в картину мира, воспроизведенную аналитиком в модели<sup>1</sup>.

Будем называть активностью любые изменения, происходящие в моделируемой части реальности, подлежащие отражению в модели. Частным случаем активности является деятельность – это активность, источником которой является сознательная воля субъектов, продиктованная реализацией их интересов. Представляется правильным определение субъекта как актора, обладающего собственными интересами и возможностями для их реализации; из этого определения вытекают важные следствия о разграничении объектов и субъектов в модели. Таким образом, активностью и предметом моделирования для нас является «экземпляр процесса», а не «процесс» в терминологии распространенных методик моделирования процессов.

Охарактеризовать активность можно набором состояний системы: исходным, конечным, а также – при необходимости –

---

<sup>1</sup> Эти вопросы подробно рассмотрены М. Мирошниченко в третьей главе настоящей монографии, а также в серии публикаций, которые можно найти на ресурсе <https://habr.com/users/maxstroy/>

промежуточными. Каждое состояние представляет собой описание набора объектов и их свойств на определенный момент времени.

Простым примером активности (деятельности) может служить продажа товара. Пусть товар (актив) А принадлежал субъекту В, в момент времени  $t_1$  была совершена сделка купли-продажи, и товар перешел в собственность субъекта С. В модели должны быть созданы следующие индивидуальные объекты: субъект В, субъект С, товар (актив) А, принадлежность товара А субъекту В (обозначим этот объект АВ), принадлежность товара А субъекту С (АС). Принадлежность товара субъекту – это вид состояния, то есть можно построить такую иерархию классов: Состояние/Принадлежность/Принадлежность товара субъекту. К объектам класса «Принадлежность товара субъекту» будет применяться такой набор свойств:

- принадлежащий товар;
- субъект-владелец;
- дата и время начала владения;
- дата и время окончания владения.

Перечисленных элементов достаточно для выражения принадлежности товара до и после сделки, но как описать ее саму? Причиной изменения состояний являются события. Сделка – это вид события, поэтому мы вправе ввести классы Событие/Сделка/Купля-продажа товара. Объекты класса «Купля-продажа товара» будут обладать следующим набором свойств:

- продавец;
- покупатель;
- продаваемый товар.

– свойства объектов класса «Принадлежность товара субъекту» дополним связями со сделками, явившимися причиной возникновения и прекращения состояния:

- сделка-основание возникновения состояния;
- сделка-основание прекращения состояния.

Диапазоном значений этих свойств будет класс «Сделка» или даже «Событие», поскольку права на товар могут переходить не только в результате купли-продажи.

Итоговая диаграмма классов и свойств, связывающих их объекты, будет выглядеть так (рис. 5.3).

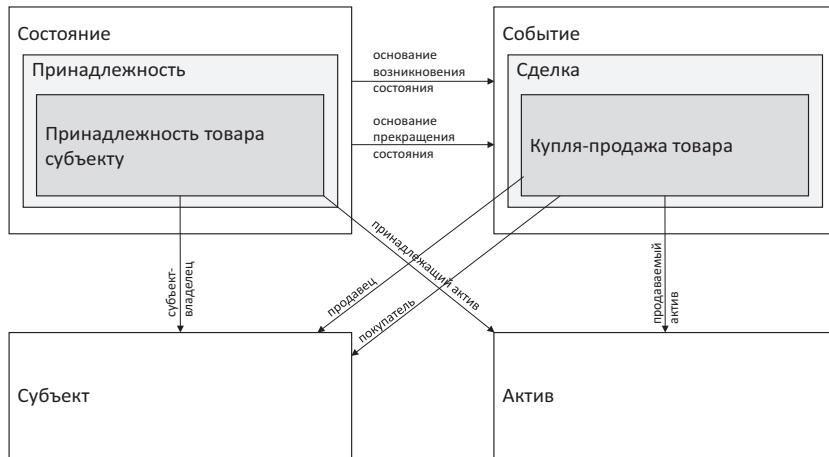


Рис. 5.3. Диаграмма классов и свойств модели продажи актива

Обратим внимание, что в этом примере каждый продаваемый товар или актив описывается как индивидуальный объект. Такой вариант моделирования пригоден для описания, например продажи крупных активов или оборудования, каждая единица которого имеет серийный номер и может быть идентифицирована. Как смоделировать продажу товара, поштучный учет которого нецелесообразен или невозможен, например бетона, продаваемого кубометрами?

Очевидный вариант решения проблемы состоит в создании класса «Партия товара», описывающего определенное количество или объем продаваемой субстанции. Можно даже смоделировать деление и объединение этих партий (оптово/розничная торговая деятельность), оперируя количественными характеристиками каждой партии. А что, если в какой-то момент нужно переключиться между видами учета? Например, организация приобрела 10 трансформаторов, не имея идентификационных данных каждого из них. С точки зрения отдела снабжения речь идет о партии

из 10 однородных единиц, каждую из которых невозможно идентифицировать в отдельности. С другой стороны, при приеме трансформаторов на склад, и тем более при их выдаче для установки на технические места, необходима идентификация и поштучный учет каждой единицы.

Для решения этой задачи необходимо определить момент, когда единицы оборудования перестают рассматриваться в составе партий товара и начинают учитываться поштучно. Далее нужно определить способ отражения этого события в модели. Примем за момент начала поштучного учета приемку оборудования на склад, когда каждой единице оборудования присваивается инвентарный номер. Значит, нужно определить подкласс «Прием оборудования на склад» класса «Событие». Свойства объектов этого класса будут связывать их с поступившей партией товара, а также с каждой из единиц оборудования, которой присвоен инвентарный номер. Дата и время приема оборудования на склад будут соответствовать дате прекращения существования объекта «Партия товара» и дате начала существования объектов – единиц оборудования.

Может возникнуть вопрос: все подобные операции учета автоматизируются в специализированном прикладном программном обеспечении, таком как системы учета ТМЦ, ERP-системы и др. В связи с чем может возникнуть необходимость создавать онтологические модели учетных операций? Один из возможных ответов заключается в решении задачи отслеживания жизненного цикла активов с целью анализа эффективности их эксплуатации, планирования ремонтов и модернизации. Единая картина движения материальных ценностей, охватывающая все этапы их приобретения и использования предприятием, может помочь в решении задач экономической оптимизации и улучшения логистики.

Более глубокий ответ состоит в том, что на современном этапе развития корпоративных информационных технологий намечается переход от привычной парадигмы автоматизации, когда компьютерные системы лишь помогают выполнять те или иные рутинные процессы, которые персонал выполнял бы и без них, к полностью

или в высокой степени автоматизированному выполнению управленческих и аналитических задач. Современные технологии предоставляют возможность решать некоторые виды подобных задач значительно эффективнее, чем мог бы их решить человек. Раз применение таких технологий экономически оправдано, их внедрение в практику будет неизбежно расширяться. При этом те предприятия, которые начнут трансформацию управленческих процессов раньше, имеют шанс получить наибольшие преимущества от нее. Для успешного выполнения такой трансформации необходимо применение целого спектра технологий и методик, который наряду с онтологическим моделированием включает использование имитационного моделирования, машинного обучения, нейросетей и др. Онтологическое моделирование является необходимым компонентом современных управляющих ИТ-систем, поскольку предоставляет наиболее полный и качественный базис для рационального постижения и воспроизведения в модели принципов функционирования технологических, экономических и социальных систем.

Можно привести следующий пример методологического отличия разработки автоматизированных систем, основанных на онтологических моделях, от традиционных корпоративных систем. Большинство операций в бизнес-процессах, таких как рассмотренные выше операции купли-продажи, сопровождаются оформлением документов, подтверждающих факт выполнения операции. Для купли-продажи такими документами являются товарные накладные, акты, счета-фактуры и др. Многие учетные системы в начале своего существования создавались для автоматизации создания подобных документов, а также для ведения бухгалтерского учета, методика которого основана именно на обработке подобных документов, в результате чего понятия документа и операции в них не разделяются. Аналогичным образом строятся и другие автоматизированные системы. Результатом такого подхода становится то, что в учетной системе журнал операций продажи может называться «журналом товарных накладных», а журнал технических

освидетельствований – «журналом актов ТО». Записи таких журналов обладают как свойствами, описывающими собственно документы (дата, номер), так и свойствами, относящимися к операции (например, результат технического освидетельствования). На первый взгляд, ничего страшного в этом нет, более того, такая структура данных может быть удобна для пользователей, которые и на уровне своих представлений могут не разделять операцию, ее результат и отражающий ее документ. Однако при работе с подобной структурой данных аналитикам и разработчикам приходится идти на компромиссы, которые могут стать причинами ошибок при построении аналитики и отчетов, дорого обойтись при расширении функционала системы. Примерами таких компромиссов могут быть: создание фиктивных документов, описывающих операции, при выполнении которых документ на самом деле не оформлялся, или искажение информации в системе в случае, если действительные параметры операции (например, дата) по каким-то причинам отличались от указанных в составленном документе.

При создании онтологических моделей, как было сказано выше, каждый класс должен иметь одну основополагающую, существенную характеристику. Класс «Акт технологического освидетельствования» должен являться подклассом класса «Документ», а класс «Технологическое освидетельствование» – подклассом класса «Операция». Высокоуровневые классы «Документ» и «Операция» разделены, то есть ни один объект не может одновременно входить в оба эти класса или их подклассы. Таким образом, при отражении технологического освидетельствования в автоматизированной системе, построенной на основе онтологической модели, необходимо будет создать два индивидуальных объекта. Один из них должен отражать операцию технологического освидетельствования, другой – оформленный в связи с этим документ. Объекты должны быть связаны между собой, например значением свойства-ссылки «Документ-результат», указывающей от операции на документ, и/или противоположно направленной ссылки «Оформлен на основании».

Разумеется, задачи моделирования состояний, активности, операций далеко не исчерпываются функциональными потребностями учетных систем. Широкий круг задач связан с моделированием изменений в технологических и физических системах. Методы моделирования динамических систем с использованием онтологического моделирования рассматриваются в работе [9]. Наряду с теоретическим обзором, затрагивающим вопросы моделирования времени и причинно-следственных отношений, последовательностей событий, построения различных видов имитационных и аналитических моделей, в этой книге рассматриваются примеры решения практических задач для широкого спектра предметных областей – от агротехники и экологии до производства и военного дела, от социальных отношений до обработки данных с групп сенсоров.

## **§ 8. Утверждения о множествах объектов. Класс или индивидуальный объект?**

Необходимость четко разграничивать классы и индивидуальные объекты создает некоторые проблемы при моделировании. Может оказаться, что сущность, в начале разработки модели принятая за индивидуальный объект, по мере ее развития должна быть преобразована в класс объектов. В теории онтологического моделирования считается, что индивидуальный объект – это сущность, которая не может быть далее воплощена (*instantiated*). Проблема состоит в том, что возможность дальнейшего воплощения, как и все остальные аспекты онтологии, зависит от точки зрения. Например, если строится иерархия типов учреждений, в которой предполагается описывать требования к различным учреждениям, то индивидуальными объектами станут типы учреждений самого нижнего уровня, поскольку требования предъявляются именно к ним, а не к конкретным учреждениям, которых в этой модели нет. Пусть необходимо выразить в модели информацию о том, что в кафе обязательно должны иметься противопожарная сигнализация

и схема эвакуации при пожаре, а на АЗС должны быть в наличии ящики с песком и противопожарный щит. Так как модель создается с целью описания требований, мы можем предположить, что в ней никогда не появятся индивидуальные объекты, представляющие конкретные АЗС или кафе – ведь нормативные требования вводятся к группам однотипных объектов, а не к конкретным объектам. Значит, в такой модели «кафе» и «АЗС», то есть мельчайшие неделимые (с точки зрения прикладной задачи) типы учреждений, могут быть представлены как индивидуальные объекты, о которых затем будут сделаны соответствующие утверждения.

Если же прикладная задача состоит в создании справочника предприятий города, то очевидно, что «кафе» и «АЗС» должны быть в ней классами. Индивидуальные объекты будут представлять конкретные кафе и АЗС, о которых будет сообщаться информация – часы работы, адреса, вид кухни и др.

На практике не всегда возможно сделать однозначный выбор между этими двумя вариантами. Более того, в ряде ситуаций необходимо рассматривать одну и ту же сущность и как класс, и как индивидуальный объект. OWL позволяет делать утверждения о взаимоотношениях классов (например, о том, что «ни одна АЗС не является клиникой»), но в большей части случаев не позволяет сообщать что-либо о значениях свойств сущностей, объявленных как класс (например о том, что «АЗС относятся к классу функциональной пожарной опасности F3.1»). Такие утверждения возможны только в рамках подъязыка OWL Full: в нем каждая сущность может являться одновременно и классом, и индивидуальным объектом. Утверждения о свойствах классов выходят за рамки логики первого порядка. Стандарт OWL версии 1 описывал несколько подъязыков, обеспечивающих разную степень выразительности. От выбора подъязыка существенно зависит скорость и функциональность вычислений, выполняемых машинами логического вывода. Для подъязыка OWL Full вычислимость не гарантируется, то есть он позволяет создавать конструкции, способные поставить в тупик машину логического вывода. Стандарт OWL 2

не содержит подъязыков, его подмножества называются профилиями. Тем не менее выражение «OWL 2 Full» иногда неформально используется для обозначения графов RDF, рассматриваемых как онтологии OWL 2 (такие графы являются только одним из возможных технических способов представления онтологий OWL 2).

Выбор подъязыка OWL или профиля OWL 2 играет роль только в случае, если для работы с моделью будет использоваться машина логического вывода. Если работать с моделью, например только при помощи SPARQL-запросов, подъязык или профиль значения не имеют. С точки зрения технологий хранения и доступа к модели в графовой СУБД (граф RDF) нет препятствий к тому, чтобы присваивать классам модели значения свойств.

Более того, классы и свойства сами могут быть значениями свойств других объектов. Пусть нужно выразить в модели утверждение о том, что все объекты класса «АЗС» должны отображаться в интерфейсе автоматизированной системы при помощи иконки «fuel.png». Можно, как описано выше, выразить это при помощи свойства «Иконка для отображения», значение которого «fuel.png» присвоить классу «АЗС». Другой вариант состоит в том, чтобы создать объект класса «Правило отображения», объявить применимые к его объектам свойства «Иконка» и «Класс отображаемых объектов», создать индивидуальный объект такого класса под названием «Правило отображения АЗС». Свойству «Иконка» этого объекта приписать значение «fuel.png», а свойству «Класс отображаемых объектов» – идентификатор класса «АЗС». В качестве диапазона значений для свойства «Класс отображаемых объектов» при его объявлении нужно указать owl:Class. Приведем несколько триплетов из этого фрагмента модели:

```
model:ОтображаемыйКласс rdf:type owl:ObjectProperty  
model:ОтображаемыйКласс rdfs:domain  
model:ПравилоОтображения  
model:ОтображаемыйКласс rdfs:range owl:Class  
model:правило12345 rdf:type owl:NamedIndividual
```

```
model:правило12345 rdf:type model:ПравилоОтображения  
model:правило12345 model:ОтображаемыйКласс model:АЗС
```

Подобные приемы, хоть и не являются общепринятыми, позволяют существенно расширить выразительность модели. Как уже упоминалось, основным аргументом против их использования является то, что на построенных с их помощью моделях невозможны вычисления при помощи большинства машин логического вывода.

Еще одной ситуацией, когда оправдано присвоение значений свойств классам и свойствам, является связывание создаваемой модели с моделью нормативно-правовых актов. В контексте многих прикладных задач важно обеспечить указание точного смысла каждого термина, применяемого в качестве названия класса или свойства. Определения этих терминов по возможности следует брать из нормативно-правовых актов (НПА), регулирующих сферу, для которой строится модель. Для обеспечения читаемости и обоснованности элементов модели желательно снабжать каждый ее элемент, введенный на основании содержания НПА, метаданными, в состав которых входит определение и ссылка на НПА-источник. Чтобы получить возможность создавать такие ссылки, необходимо создать в модели класс «Документ»/«Нормативно-правовой акт» (при необходимости – далее детализировав типы НПА), объекты которого будут представлять конкретные документы. Для указания связи элементов модели с НПА может использоваться стандартный атрибут `rdfs:isDefinedBy`.

Одним из наиболее перспективных направлений прикладного применения онтологического моделирования является моделирование требований (в том числе требований нормативных документов) и создание систем автоматизированного контроля выполнения требований. Методика моделирования источников права, отражения в модели правового понятийного аппарата, представления нормативных правил и требований подробно рассматривается в работе [7].

## § 9. Машины логического вывода и другие способы работы с моделью

Возможность использования машин логического вывода является одной из главных причин, по которой разработчики выбирают онтологии в качестве технологической основы автоматизированных систем. Машины логического вывода – это программные компоненты, работающие с графовыми СУБД (хранилищами триплетов) или файлами, содержащими онтологические модели, при помощи специального API. Их задача состоит в применении логических правил, как заданных стандартами RDF/RDFS/OWL (например, выведение факта о том, что объект входит в надкласс, если известно, что он входит в подкласс), так и правил, определяемых автором модели. Для создания пользовательских правил предназначены несколько нотаций, таких как SWRL и SPIN.

В этих нотациях аналитик может описывать произвольные правила. Проиллюстрируем их конструирование на примере. Пусть в модели содержится описание автомобилей А и В, попавших в ДТП на перекрестке. Известно, что автомобиль А двигался на красный сигнал светофора, а автомобиль В – на зеленый. Желаемый вывод состоит в том, что виновником ДТП является водитель автомобиля А.

Дорожно-транспортное происшествие является событием. На момент наступления этого события автомобили А и В находились в состоянии движения в направлениях, которые мы обозначим  $a1$  и  $b1$ . Светофор на перекрестке, где произошло ДТП, находился в состоянии, разрешающем движение в направлении  $b1$  и запрещающем его в направлении  $a1$  (рис. 5.4).

Для описания состояния светофоров на перекрестке введем класс «Состояние регулирования движения», обладающий свойствами:

- перекресток, к которому применимо состояние;
- время начала и окончания действия состояния;
- направления, в которых разрешено движение;
- направления, в которых запрещено движение.

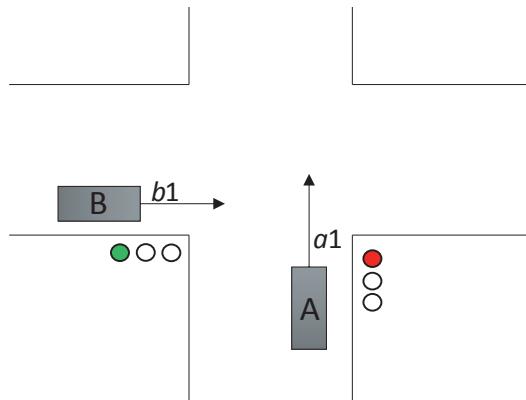


Рис. 5.4. Схема моделируемого ДТП

Напомним, что в онтологических моделях каждое свойство индивидуального объекта может иметь любое число значений. Благодаря этому можно создать один объект класса «Состояние регулирования движения», который опишет разрешение и запрет на движение сразу в нескольких направлениях.

Диаграмма классов и свойств фрагмента модели приведена на рис. 5.5.



Рис. 5.5. Диаграмма классов и свойств фрагмента модели ДТП

Приступим к конструированию правила. Синтаксис SWRL более пригоден для чтения, поэтому запись будем вести в нем. В условиях и выводах правил SWRL используются унарные и бинарные

предикаты, аргументами которых являются индивидуальные объекты или происшествия. Унарный предикат – запись вида СостояниеРегулированияДвижения (?state). Смысл такой записи состоит в том, что переменная ?state может означать любой объект, относящийся к классу СостояниеРегулированияДвижения (здесь и далее не употреблен префикс model: для краткости). Бинарный предикат – выражение вида ЗапрещеноДвижение (?state,?direction). Оно означает наличие связи ЗапрещеноДвижение между объектами в переменных ?state и ?direction.

Условие нашего правила будет состоять из двух частей. Сначала объявим все переменные, которые будут означать объекты различных классов:

СостояниеРегулированияДвижения (?state)  
НаправлениеДвижения (?direction)  
ПоложениеТранспортноСредства (?position)  
ТранспортноеСредство (?car)  
ДТП (?accident)  
Водитель (?driver)

Затем определим отношения, которыми должны быть связаны объявленные объекты:

ЗапрещеноДвижение (?state,?direction)  
ОписываетТранспортноеСредство (?position,?car)  
ДвигалосьВНаправлении (?position,?direction)  
ВМоментПроисшествия (?position,?accident)  
ПодУправлением (?position,?driver)

Оставим за пределами нашего правила условие, говорящее о том, что состояние ?state имело место на момент происшествия ?accident, чтобы не усложнять пример. Вывод правила будет состоять в утверждении о том, что водитель ?driver является виновником ДТП ?accident. В полном виде правило записывается так:

СостояниеРегулированияДвижения (?state), Направление  
Движения (?direction), ПоложениеТранспортного  
Средства (?position), ТранспортноеСредство (?car),  
ДТП (?accident), Водитель (?driver), ЗапрещеноDвижение  
(?state,?direction), ОписываетТранспортноеСредство  
(?position,?car), ДвигалосьВНаправлении (?position,  
?direction), ВМоментПроисшествия (?position,?accident),  
ПодУправлением (?position,?driver) => ИмеетВиновника  
(?accident,?driver)

В левой части правила, до знака =>, записываются все его условия, объединяемые между собой логическим «И». В правой части записывается вывод. Правило работает следующим образом: если находится комбинация индивидуальных объектов, отвечающих всем условиям (то есть для каждой переменной находится подходящий объект), то в модели создаются утверждения с участием этих объектов, указанные в выводе правила. Если существует несколько комбинаций объектов, удовлетворяющих условиям, то будет создано несколько соответствующих фактов.

Правила могут каскадироваться, то есть факты, полученные в качестве вывода одного правила, могут выступать в роли условий для следующего.

Правила SPIN функционируют по тому же принципу. Их отличие состоит в том, что условия правил помещаются в саму онтологическую модель в виде триплетов, составленных при помощи предикатов специальной метамодели для описания правил. Каждое правило может быть транслировано в SPARQL-запросы для проверки условия и вставки результирующих фактов.

У правил логического вывода по стандартам Semantic web есть ряд функциональных ограничений: в них нельзя использовать отрицание, нельзя удалять (опровергать) факты в качестве вывода и др. Для устранения этих ограничений могут дорабатываться стандартные машины логического вывода или создаваться специальные.

Правила логического вывода могут применяться для решения таких функциональных задач, как обогащение данных,

поступающих в реальном времени, согласование и очистка наборов данных, контроль выполнения требований. Одной из главных сфер их применения являются системы поддержки принятия решений. Как средство имитации рационального мышления, они позволяют быстро вычислять все возможные следствия из имеющегося широкого набора фактов. Результат вычислений может использоваться для определения диапазона рекомендуемых (или, наоборот, не рекомендуемых) действий, для прогнозирования возможных последствий тех или иных вариантов решений.

### **Список библиографических ссылок**

1. *Keet M.* An Introduction to Ontology Engineering. College Publications, 2018. 256 p.
2. *Горшков С.* Введение в онтологическое моделирование. 2016. URL: <https://trinidata.ru/files/SemanticIntro.pdf>.
3. *Гаврилова Т., Кудрявцев Д., Муромцев Д.* Инженерия знаний. Модели и методы. СПб. : Изд-во «Лань», 2016. 324 с.
4. *Kuba M.* OWL 2 and SWRL Tutorial. URL: <https://dior.ics.muni.cz/~makub/owl/>.
5. *Partridge C.* Business Objects: Re-Engineering for Re-Use. BORO Centre, 2005. 566 p.
6. *Sowa J.* Knowledge Representation. Logical, Philosophical, and Computational Foundations. Brooks/Cole, 2000. 594 p.
7. *Boer A.* Legal Theory, Sources of Law & the Semantic Web. IOS Press, 2009. 321 p.
8. *Gorshkov S., Kralin S., Miroshnichenko M.* Multi-viewpoint Ontologies for Decision-Making Support // Knowledge Engineering and Semantic Web. Vol. 649 of the series Communications in Computer and Information Science, 2016. P. 3–17
9. *Handbook of Dynamic System Modeling / ed. P. Fishwick // Chapman & Hall/CRC*, 2007. 750 p.

## Глава 6

# ПРИМЕНЕНИЕ ОНТОЛОГИЙ В АВТОМАТИЗИРОВАННЫХ СИСТЕМАХ

*С. В. Горшков*

## Программные средства работы с онтологиями

Обзор методов онтологического моделирования и возможностей соответствующих информационных технологий, приведенный в предыдущей главе, позволяет оценить применимость этих инструментов для решения прикладных задач различных классов. Спектр таких задач достаточно широк. В этой главе мы рассмотрим несколько сценариев, в которых использование онтологических моделей имеет существенные преимущества перед другими возможными технологиями. Но прежде необходимо рассмотреть арсенал инструментов для работы с онтологическими моделями в программной среде.

Как было сказано выше, спецификации консорциума W3C определяют способы структурирования информации (RDF, RDFS, OWL), способы доступа к ней (SPARQL) и синтаксис правил логического вывода (SWRL, SPIN). Онтологические модели, представленные в соответствии со стандартами RDF/RDFS/OWL, могут быть сохранены в файлы различных форматов, представляющих их содержимое как в виде наборов триплетов, так и в виде проекции на XML-структуры. Однако наиболее пригодным для практического применения в корпоративных автоматизированных системах является хранение онтологических моделей в так называемых хранилищах триплетов (RDF triple store). Этот класс программных продуктов включает графовые базы данных, позволяющие получать доступ к хранящимся в них онтологическим моделям как при помощи протокола SPARQL, работающего поверх HTTP и похожего на привычный разработчикам язык работы с реляционными

базами данных SQL, так и при помощи программного интерфейса для компонентов, написанных на Java – OWL API. Так, при помощи OWL API к хранилищам триплетов подключаются машины логического вывода (reasoner) – компоненты, обеспечивающие выполнение правил логического вывода.

Прикладные компоненты автоматизированных систем также могут работать с онтологическими моделями, размещенными в хранилищах триплетов, при помощи HTTP-интерфейса к точкам доступа SPARQL или посредством OWL API. Типичный состав компонентов автоматизированной системы с онтологическим ядром представлен на рис. 6.1.

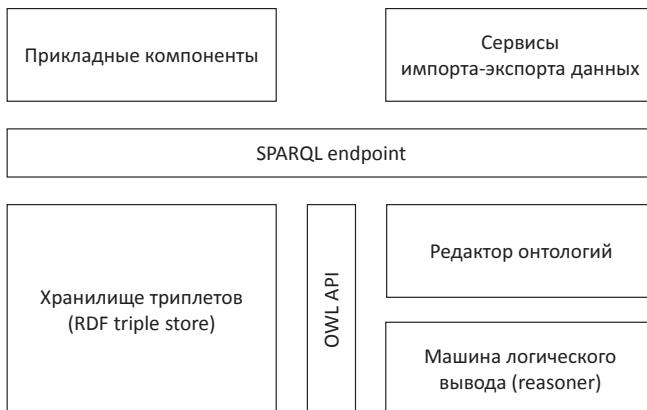


Рис. 6.1. Компоненты автоматизированной системы, использующей онтологии

Работа конечного пользователя обычно происходит только с прикладными компонентами, имеющими визуальный интерфейс, соответствующий функциональным задачам продукта, таким как поиск, обработка транзакционных данных, поддержка принятия решений и др. Прикладные компоненты, как и средства импорта-экспорта данных, взаимодействуют с онтологической моделью, находящейся в хранилище триплетов. Редактор онтологий используется для создания классов и свойств, определяющих структуру онтологической модели, а также для ввода содержимого

справочников и классификаторов. Машина логического вывода обеспечивает автоматическое дополнение информации, сосредоточенной в хранилище триплетов, новыми фактами, полученными на основании применения правил к уже имеющимся в хранилище фактам.

Производительность хранилищ триплетов обычно уступает производительности реляционных баз данных, что связано с особенностями организации информации и построения запросов на доступ к ней. В графовой базе данных строятся только общие индексы по каждому из трех элементов триплета (субъект, предикат, объект); такой подход уступает по производительности реляционным СУБД, где существует возможность строить индексы в рамках каждой таблицы, учитывая специфику структуры конкретных данных и виды запросов, которые будут их использовать. В случае, если необходимо работать с большим объемом информации о конкретных объектах, представленной в соответствии с онтологической моделью, существуют две альтернативы: использовать специальные высокопроизводительные хранилища триплетов и машины логического вывода (такие как FaCT++) или сохранить часть информации за пределами графовой СУБД в noSQL-хранилище или реляционной базе данных. Во втором случае необходимо реализовать тот или иной способ проекции структуры онтологической модели на структуру выбранного хранилища. Этого довольно легко достичь с MongoDB и другими хранилищами, работающими с JSON-объектами, или с базами данных, поддерживающими наборы колонок (существуют и попытки создать SPARQL-интерфейсы для традиционных хранилищ; из решений такого класса можно упомянуть Apache Rya – впрочем, степень их зрелости вызывает сомнения). В этом случае при изменении в онтологической модели набора свойств, применимых к сущностям того класса, объекты которого хранятся вне графовой базы, не требуется предпринимать никаких изменений в хранилище, поскольку набор свойств сущностей в нем не фиксирован. В случае использования реляционной СУБД в качестве внешнего хранилища необходимо или

автоматически обновлять структуру таблиц в соответствии с изменениями онтологической модели, что достаточно трудоемко, или применять денормализованную структуру таблиц в СУБД (например, создать таблицы с идентификаторами и типами сущностей, таблицу свойств, таблицу значений свойств для конкретных сущностей).

Использование таких приемов оправдано в случае невозможности использовать высокопроизводительные графовые СУБД или при работе с «большими данными», которые заведомо не поместятся в графовую базу. При этом теряется возможность использовать обычные машины логического вывода для дополнения данных, язык SPARQL и OWL API для доступа к ним. Однако на рынке представлены решения, абстрагирующие от прикладных компонентов систем технологические детали хранения информации и позволяющие работать с ними как с содержимым графовой СУБД независимо от того, в хранилище какого типа на самом деле расположены эти данные.

Прикладные компоненты, использующие онтологические модели, должны быть готовы к изменению набора классов и свойств модели. Одним из существенных преимуществ онтологий является то, что описание структуры данных (TBox) может быть извлечено из хранилища теми же техническими средствами, что и данные о конкретных объектах (ABox). Прикладные программные компоненты могут и должны считывать из модели набор классов, перечни свойств, применимых к объектам каждого класса, содержимое справочников, используемых для задания значений свойств объектов. Это существенно отличает приложения, основанные на онтологиях, от традиционных приложений, базирующихся на реляционных СУБД, принципах объектно ориентированного программирования и паттерне MVC, где структура данных жестко связана с программным кодом, и любое ее изменение требует обновления кода приложения.

Далее мы рассмотрим несколько сценариев применения онтологических моделей для создания автоматизированных систем

в различных прикладных областях и предназначенных для решения разных классов проблем. Их описание носит прикладной характер и иллюстрирует возможности применения рассмотренных выше технологий и методик.

## **Сценарий 1: Обеспечение доступности знаний в организации**

### **Постановка проблемы**

Любая крупная организация имеет дело с огромным объемом информации, описывающей среду и процессы ее деятельности. Эта информация используется как в текущей деятельности организации, так и в процессах анализа, принятия управленческих решений, стратегического планирования. Одна часть такой информации отражена в автоматизированных системах, другая часть находится в различных документах и электронных таблицах, расположенных в слабо структурированных файловых хранилищах, а наиболее значительная часть хранится только в головах сотрудников организации. В том числе, как правило, только опытные сотрудники располагают сведениями о том, в какой именно автоматизированной системе или файловом хранилище можно найти ту или иную информацию; таким образом, без участия этих сотрудников нельзя воспользоваться даже теми сведениями, которые представлены в электронной форме. Будем называть *знаниями* информацию, непосредственно доступную сотрудникам организации для использования (независимо от формы ее представления), а *данными* – информацию, находящуюся в различных хранилищах, требующую поиска и извлечения, переработки и осмыслиения перед практическим применением.

Процесс обучения нового сотрудника и его вхождение в деятельность организации напрямую зависит от того, насколько быстро и полно он освоит работу с массивами имеющейся в организации информации. Эффективность таких процессов зависит от коммуникативных качеств сотрудника: насколько хорошо он

сможет наладить общение с коллегами, чтобы вовремя получать требуемые ему сведения. Такая зависимость несет серьезные риски для организации, так как ставит результаты ее работы в слишком серьезную зависимость от личных качеств конкретных людей, климата в коллективе и т. д. При возникновении кризисной ситуации, которая разрешается уходом части коллектива, или в силу естественного кадрового обновления может произойти невосполнимая потеря значительной части информации, и как следствие – падение эффективности работы предприятия.

Чтобы смягчить подобные негативные эффекты, на предприятии необходимо выстроить организационную и техническую систему работы со знаниями. Поддерживающая ее автоматизированная система прежде всего должна обеспечить каталогизацию, высокоуровневое описание всей имеющейся информации, инструменты и процедуры поиска и предоставления доступа к необходимым массивам данных и конкретным объектам. Все блоки информации, с которыми работает организация, так или иначе связаны с объектами и процессами, входящими в контур ее деятельности. Соответственно, описание каждого информационного блока должно строиться в терминах, используемых для описания этих объектов и процессов. В тех же терминах формулируют свои поисковые запросы и пользователи. Онтологическое моделирование предоставляет подходящий набор инструментов для формализации набора понятий предметной области, их использования для описания блоков данных и построения поисковых запросов.

В качестве примера рассмотрим организацию, эксплуатирующую сложное производственное оборудование. При проектировании каждой производственной площадки, строительстве и монтаже оборудования, а также в ходе реконструкций, модернизаций и ремонта в организацию поступает большое количество документации от проектных организаций, поставщиков оборудования, строительно-монтажных компаний и др. Вся эта документация, скорее всего, распределена между различными файловыми архивами как масштаба всей компании, так и ее отдельных подразделений. Часть

документации остается на личных компьютерах пользователей или в их почте. Большое количество информации накапливается и в ходе обычной деятельности предприятия: она касается загрузки оборудования, сведений о поломках, простоях и текущих ремонтах, затрат на эксплуатацию.

Предположим, возникает серьезная неисправность какой-либо единицы оборудования. Инженерам и руководителям предстоит принять ряд решений: ремонт или замена? сколько времени это займет, во что обойдется простой? если менять оборудование, то на какую модель, какого поставщика? Обычно подобные вопросы решаются в ходе совещаний, и для ответов на них используются преимущественно знания (опыт) сотрудников организации. Если по какой-то причине такого опыта недостаточно, например имела место серьезная ротация кадров, возникает значительный риск принятия необоснованных решений, которые приведут к экономическим потерям. С другой стороны, опыт даже самых квалифицированных специалистов является лишь их точкой зрения, которая может не учитывать всех факторов, либо отдавать предпочтение определенным вариантам решений на основании личных убеждений.

В организации с современными бизнес-процессами и рационально построенным управлением принятие подобных решений должно быть экономически обоснованным. Обоснование должно включать оценку стоимости, потерь и достигаемого эффекта для каждого из возможных вариантов решения. Для подготовки таких обоснований сотрудникам организации понадобятся все виды перечисленных выше данных: от проектной и эксплуатационной документации до сведений о ранее выполненных ремонтах данной и аналогичных единиц оборудования, информации об опыте эксплуатации оборудования разных поставщиков, имевших место и планируемых режимов эксплуатации и др. Собрать такой объем данных в короткий срок практически нереально, если не иметь инструмента, позволяющего найти требуемую информацию без обращения к другим сотрудникам и ручного пролистывания множества папок и документов.

## Архитектура решения

Инструмент, позволяющий выполнять поиск любых данных в организации, формально представляет собой поисковую систему, но фактически имеет мало общего с распространенными решениями такого рода. Типичное решение для «управления» данными организации (на самом деле – в лучшем случае их структурированного хранения) представляет собой портал, на котором можно создавать папки, наполнять их файлами и выполнять полнотекстовый поиск по их содержимому и, возможно, метаданным. Очевидно, что в решении описанной нами задачи такое решение вряд ли поможет.

При наличии полноценного инструмента управления данными, в описанной нами ситуации пользователь может выполнить серию запросов к корпоративной системе управления знаниями, каждый из которых использует термины, зафиксированные в информационной модели, например:

- все ремонты единиц оборудования марки X модели Y за период с [...] по [...];
- сведения о моделях оборудования, имеющих такие же характеристики, как оборудование марки X модели Y;
- даты и результаты технического освидетельствования единицы оборудования № 12345.

Очевидно, что никакие инструменты полнотекстового поиска не способны обеспечить получение ответов на такие запросы. Чтобы достичь этого, необходимо:

- построить онтологическую модель предметной области, в данном случае включающую такие термины, как «ремонт», «единица оборудования», «техническое освидетельствование» и др.;
- построить набор справочников и классификаторов – в обсуждаемом примере это как минимум справочники марок и моделей оборудования;
- проиндексировать всю доступную корпоративную информацию, находящуюся как в автоматизированных системах, так и в файловых хранилищах, описав каждый блок данных в терминах

модели, используя значения справочников для указания значений свойств конкретных блоков;

– создать инструмент формулирования запросов в виде логических условий (путем распознавания введенного пользователем текста или при помощи специального конструктора запросов);

– создать механизм выполнения таких запросов на основе данных, находящихся в построенных индексах.

С точки зрения ИТ-архитектуры, автоматизированная система должна включать три основных компонента:

– репозиторий модели и нормативно-справочной информации (НСИ), хранящий формализованное описание объектов и процессов в сфере деятельности компании;

– инструмент индексации всех источников информации, создающий описания каждого объекта или блока данных в терминах модели;

– инструмент поиска, дающий пользователю возможность строить поисковые запросы в терминах модели и получать на них точные ответы.

Программные продукты, реализующие функциональность таких компонентов, представлены на рынке. Они используют графовые СУБД для хранения модели и НСИ, noSQL-хранилища для построения индексов, специальное ПО для построения запросов и получения ответов на них.

Следующим шагом в развитии подобных систем являются логические витрины данных, организующие доступ к каждому элементу содержимого источников информации (а не к блокам информации, как в описанном примере). Развитие таких систем началось с появления концепции OBDA (Ontology Based Data Access), суть которой состоит в трансляции SPARQL-запросов в серии запросов к реляционным базам данных. В широком масштабе работоспособность этой концепции продемонстрирована в рамках проекта Optique [1]. Результатом работы таких систем являются не ссылки на блоки данных, полученные из индексов, а сами данные, извлеченные из систем-источников по запросу пользователя. Логические

витрины данных также способны осуществлять слияние данных из разных источников и обрабатывать их в соответствии с определенными в модели правилами.

Однако даже при работе поисковой системы на уровне блоков информации те сведения, с которыми работает пользователь, являются корпоративными знаниями, поскольку они сформулированы в терминах предметной области и доступны для непосредственного использования. Те же сведения он мог бы получить путем расспросов других сотрудников организации, то есть обмена знаниями с другими людьми. Таким образом, система управления знаниями замещает часть коммуникативных и мыслительных функций, то есть является в полном смысле интеллектуальной автоматизированной системой.

Важно также, что при внедрении подобных систем неизбежно реализуются и организационные процессы управления знаниями – если под управлением понимать прежде всего создание и поддержку формализованного описания структуры и состава бизнес-знаний, имеющихся в организации. Построив онтологическую поисковую систему, компания становится также обладателем онтологической модели, построенной аналитиками совместно с экспертами предметной области. Эта модель важна не только как основа для структурирования и связывания корпоративных данных, но и как инструмент, обеспечивающий одинаковое понимание сотрудниками организаций используемых ими ключевых терминов. Модель предоставляет тезаурус, который используется и вне информационных систем: при постановке целей, формулировании поручений, проектировании наборов ключевых показателей и мотивационных программ, написании стандартов организации и других документов.

Постоянные изменения среды функционирования организации и ее бизнес-процессов заставляют постоянно обновлять и дополнять, поддерживать в актуальном состоянии модель, отражающую представления бизнес-пользователей. Управление моделью, в прямом смысле слова являющейся управлением знаниями,

представляет собой самостоятельный организационный процесс. В нем участвуют заинтересованные пользователи, предлагающие изменения в модель, и экспертный орган, принимающий или отклоняющий их, а также следящий за общей актуальностью модели и наборов нормативно-справочной информации.

## **Сценарий 2: Системы консолидации и анализа данных**

### **Постановка проблемы**

В предыдущем сценарии мы рассмотрели организацию поиска по внутрикорпоративным источникам данных. В холдинговых структурах часто возникает задача, которую можно посчитать схожей: интеграция сведений, полученных из автоматизированных систем филиалов и зависимых организаций (или от независимых – партнеров, дилеров, поставщиков и др.) для выполнения сводного анализа или формирования консолидированной отчетности. Существенные отличия такой задачи от рассмотренной выше составляют три фактора:

- принципиальная разнородность данных, получаемых от разных источников, в том числе невозможность интегрировать их путем введения единой нормативно-справочной информации;
- неподконтрольность изменений структуры данных в источниках;
- необходимость анализировать огромные объемы сводной информации, консолидируя их на стороне интегрирующей системы, а не оставляя в источнике по принципу логической витрины данных (из-за того, что доступ в режиме онлайн в системы-источники может отсутствовать в принципе, а получение данных от источников происходит путем периодических выгрузок).

### **Архитектура решения**

Очевидно, что решить подобную задачу можно только путем построения физического хранилища, наполняемого информацией от множества поставщиков. При помещении информации в хранилище должно происходить ее преобразование в соответствии

с общей структурой данных и единым набором справочников и классификаторов. В зависимости от организационной ситуации, преобразование данных может быть возложено на поставщиков или потребителя информации. В первом случае потребитель данных должен обеспечить публикацию машинно-читаемой структуры информации и наборов НСИ, а поставщики – на своей стороне создать инструменты преобразования данных, которые будут выполнять его на основе правил мэппинга (сопоставления) между элементами структуры данных систем-источников и модели, используемой приемником. Эти инструменты должны выявлять изменения модели со стороны приемника и оповещать о них администратора информационного обмена. Во втором случае, если у потребителя информации нет возможности требовать от поставщиков реализации таких инструментов, то ему придется создать и администрировать их на своей стороне самостоятельно.

Так или иначе, правила сопоставления элементов структуры данных целесообразно описать в самой модели. Такой подход приводит к тому, что в онтологической модели, наряду с описанием самой предметной области, появляется модель алгоритмов, применяемых для обработки информации. Возможность хранить алгоритмы вместе со структурой и содержанием самих данных представляет собой фундаментальную и крайне важную особенность, отличающую автоматизированные системы, базирующиеся на онтологиях, от любых видов традиционных ИТ-систем. Вопросы описания схем программ средствами конструктивных логик давно рассматривались в отечественной науке (см., напр., [2]). Несмотря на наличие публикаций по теме онтологического моделирования архитектуры приложений [3], эта возможность пока не до конца осознана в индустрии – однако ее потенциал огромен. Далее мы рассмотрим ряд конкретных примеров того, как она меняет подходы к созданию и поддержке автоматизированных систем.

В рамках рассматриваемой функциональной задачи необходимо хранить в модели правила смыслового соответствия элементов хранилищ данных разных систем. Хранилища могут иметь существенно

отличающуюся структуру и основываться на различных платформах: реляционных СУБД, хранилищах коллекций JSON-объектов, XML-документах и др. Каждому типу источника будет соответствовать свой фрагмент онтологии, моделирующий извлечение информации именно из него. Для реляционных баз онтология должна описывать таблицы и колонки, правила отбора данных из них, в том числе с учетом связей между таблицами; для noSQL-хранилищ это могут быть условия на извлечение JSON-объектов с определенными свойствами из заданных коллекций, а также правила получения значений конкретных информационных объектов или значений их свойств из найденных объектов JSON; для XML-документов – выражения XPath, описывающие адресацию к элементам документов, и/или шаблоны XSLT-преобразований. В любом случае, результатом применения этих правил к источникам информации является извлечение информационных объектов определенного типа, с заданным набором свойств.

На втором этапе преобразования необходимо привести извлеченные данные в соответствие с нормативно-справочной информацией, используемой в агрегирующей системе. Для этого могут применяться таблицы соответствия кодов записей, также реализованные в качестве вспомогательных объектов онтологии. Ссылки с одних извлекаемых объектов на другие нужно преобразовать, опираясь на сохраненные в агрегирующей системе идентификаторы объектов в системе-источнике. Необходимо также описать правила преобразования значений, извлекаемых из различных полей структуры данных системы-источника: может потребоваться разбиение или, наоборот, сборка значений, округление, различные виды очистки и валидации данных. Проиллюстрируем шаги описываемого процесса на простом примере (рис. 6.2).

Технологически такие правила могут быть представлены при помощи спецификации SPIN. В некоторых специфических случаях может оказаться более рациональным создать специальную метамодель описания подобных правил и реализовать программный код, интерпретирующий правила, представленные в соответствии с ней.

## Сценарий 2: Системы консолидации и анализа данных

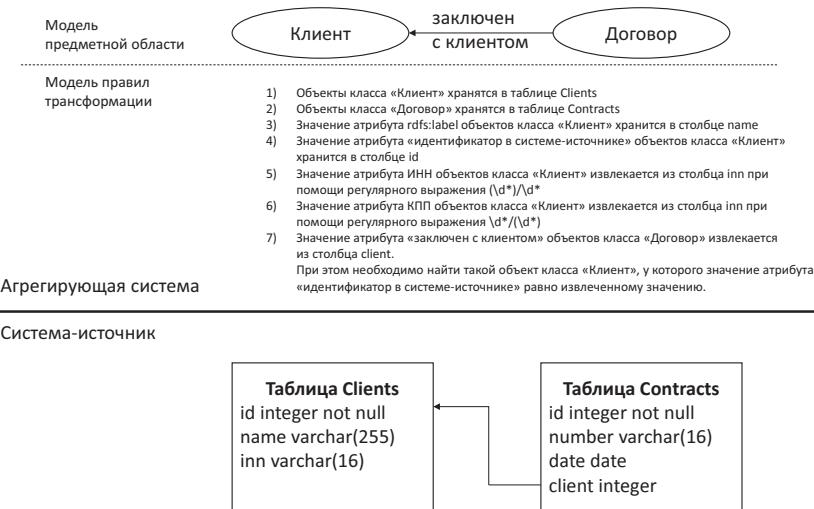


Рис. 6.2. Пример правил сопоставления структуры данных в системе-источнике и в онтологической модели

Отдельную сложность представляет сопоставление информации об одних и тех же объектах, поступающих из систем разных поставщиков. В этом случае агрегирующая система воспроизводит функционал MDM – системы управления мастер-данными, в которой создана эталонная запись для каждого объекта, сведения о котором присутствуют сразу в нескольких системах, и хранятся идентификаторы этого объекта в каждой из них. Формирование эталонного набора данных и выявление одних и тех же объектов, отраженных в разных системах, в зависимости от специфики данных выполняется полуавтоматически (сопоставление на основе определенного правила с последующей верификацией экспертом) или вручную.

При создании подобной системы нужно позаботиться об удалении из агрегирующей системы данных, удаленных из систем-источников, об обработке ошибочных ситуаций (невозможность выполнения преобразования, невозможность разрешить ссылку и т. д.). Создание системы с таким функционалом является трудоемкой задачей, требующей предельно внимательного отношения и тщательной

отладки, зато достигаемый результат обеспечивает построение таких наборов данных, которые было бы невозможно получить или надежно поддерживать при использовании другого инструментария.

После того как агрегированное представление данных построено, наступает стадия обработки информации. В зависимости от прикладной задачи обработка может происходить как при помощи императивных алгоритмов (например, отбор и ранжирование объектов по тем или иным критериям, поиск оптимальных решений, расчет балансов и др.), так и при помощи алгоритмов построения отчетов, гибко конфигурируемых пользователем или заложенных в самой модели. В последнем случае речь идет о добавлении в онтологическую модель еще одного фрагмента, описывающего структуру отчетов в терминах таблиц (документов), их разделов, строк, столбцов, алгоритмов (формул) вычисления значений для каждого элемента отчета.

Примером функциональной задачи, решаемой при помощи автоматизированной системы, построенной по описанной архитектуре, можно назвать выявление цепочек передачи ценностей между определенным набором контрагентов с целью выявления и предотвращения потерь, оптимизации маршрутов передачи, контроля экономических параметров цепочки сделок. Другим примером может быть построение целостной модели комплекса инфраструктуры или активов, эксплуатируемых группой организаций, с целью осуществления скоординированной и оптимальной политики ее развития, рационального распределения инвестиций.

Ключевыми преимуществами онтологий, реализующимися при создании как систем консолидации и анализа данных, так и чисто интеграционных решений, являются:

- техническая однородность хранения структуры и содержания данных, что обеспечивает возможность управления структурой данных при помощи тех же инструментов, которые используются для чтения и изменения самих данных;

- возможность хранения в составе одной информационной модели как обобщенного онтологического представления об объектах

и процессах, информации о которых обрабатывается агрегирующей или интегрирующей системой, так и структур, с помощью которых сведения о них представлены в системах-источниках, и правил преобразования данных;

– возможность использования правил логического вывода для преобразования и валидации данных.

## **Сценарий 3: Система поддержки принятия решений**

### **Постановка проблемы**

Назначение большинства систем поддержки принятия решений состоит в предоставлении лицу, принимающему решение, как можно более достоверной и полной информации о факторах, которые должны учитываться при принятии решения. Факторы могут включать:

- оценку возможных последствий каждого из возможных вариантов решения;
- нормативные требования и рекомендации, которые применимы к данной ситуации и должны быть соблюдены;
- мнения и интересы субъектов, затрагиваемые принимаемым решением, и др.

Ситуации принятия решений могут возникать как в контексте оперативной деятельности, так и стратегического планирования. Одним из примеров деятельности, в которой принятие решений играет особенно важную роль, является поддержка функционирования (безопасности, целостности) какой-либо системы, например промышленного предприятия. Фазы такой деятельности можно описать при помощи цикла управления безопасностью, показанного на рис. 6.3 и представляющего собой расширенную версию широко известного цикла OODA (observe, orient, decide, act):

В качестве начальной фазы цикла удобно рассмотреть реагирование на инцидент, выводящий систему из состояния нормального функционирования. Организационная единица,



Рис. 6.3. Цикл управления безопасностью

ответственная за реагирование, должна выполнить ряд действий в соответствии с типом инцидента. Целью этих действий является прекращение развития инцидента (ограничение ущерба), после чего наступит фаза возвращения объекта к нормальной работе. Для запуска программы действий (как правило, подготовленной заранее) необходимо отнести инцидент к одному из заранее определенных типов. Такой подход оправдан необходимостью выполнять реагирование в кратчайшие сроки (время на принятие решения может не превышать нескольких минут или секунд), а также необходимостью иметь нормативное обоснование программы действий. Подобные задачи актуальны при построении ситуационных центров городов, территорий, опасных производственных объектов, элементов инфраструктуры. Пример использования онтологических правил логического вывода в территориальном ситуационном центре приведен в работе [4], на железнодорожном транспорте – в работе [5].

### Архитектура решения

Одной из первых задач системы поддержки принятия решений в этом случае становится определение типа инцидента. Эта задача может выполняться с использованием правил логического вывода, принимающих в качестве предпосылок формализованное описание контекста происшествия и его сути. Преимуществом

автоматизированной системы в данном случае является возможность очень быстро обработать весь объем доступной информации, то есть учесть все предпосылки и возможные следствия. Система также может сформировать нормативное обоснование для всех предложенных ею решений. Окончательное принятие решения при этом остается за человеком, который наряду с формальными основаниями имеет также возможность руководствоваться ценностными установками.

На последующих фазах цикла управления безопасностью происходит анализ причин возникновения инцидента, формирование обоснованной программы действий по предотвращению аналогичных инцидентов в будущем и/или повышения готовности к их возникновению. В этих процессах возникают ситуации принятия стратегических решений, не столь жестко ограниченные по времени. Критерием принятия обоснованного решения является его рациональность: например, не имеет смысла тратить на предотвращение инцидентов суммы, превышающие возможный ущерб от них. В этом случае задачи системы поддержки принятия решений сводятся к оценке последствий каждого из возможных вариантов решений в терминах затрат и достигаемого эффекта, поиск варианта с оптимальным (с точки зрения заданных пользователем критериев) их соотношением. Для этого может использоваться широкий спектр методов – от уже упоминавшегося моделирования динамических систем до различных вариантов оценки частоты и возможного ущерба от инцидентов с целью определения количественного показателя риска, размерностью которого является сумма ущерба на единицу времени.

Другим характерным примером систем поддержки принятия решений являются медицинские информационные системы, предназначенные для помощи специалистам в процессе назначения обследований, постановки диагноза и назначения терапии. Нормативные документы (федеральные рекомендации, стандарты ВОЗ и др.) регламентируют спектр возможных и необходимых решений в каждой ситуации, описывая их в виде набора предпосылок

и вывода, например: «если [набор симптомов и индивидуальных факторов] то [необходимые обследования или терапия]». Такая форма представления знаний позволяет удобно создавать на их основе правила логического вывода, автоматически применяемые машиной логического вывода к описанию конкретных клинических ситуаций.

Разумеется, для формирования описаний клинических ситуаций и правил логического вывода необходима онтологическая модель, содержащая набор терминов и справочников значений, позволяющих охарактеризовать как различные клинические признаки и факторы, так и возможные варианты обследований, диагнозов, терапии, лекарственных средств и их свойств.

Медицина, поиск новых лекарств, генетические исследования являются областью, в которой применение онтологического инструментария широко признано и демонстрирует убедительные преимущества. Благодаря этому существует широкий круг развитых медицинских онтологий (OBO Foundry, FMA, Gene ontology и многие другие). Применение онтологий в медицинских системах поддержки принятия решений привлекает внимание многих исследователей и воплощается в прикладных разработках [4; 6].

Независимо от предметной области, основанные на онтологиях системы поддержки принятия оперативных решений функционируют следующим образом:

- сбор информации из разнородных источников (датчики и сенсоры, информационные сообщения от сторонних лиц и организаций, данные исследований и интерпретации их результатов, и др.);
- приведение информации в соответствие структуре онтологической модели;
- применение к полученной информации правил логического вывода для вычисления следствий из имеющихся предпосылок;
- предоставление результатов обработки информации пользователю с возможностью просмотреть логические основания каждого полученного вывода: примененные правила, исходные факты, их происхождение.

## **Сценарий 4: Разработка программного обеспечения, управляемого онтологией**

### **Постановка проблемы**

Со времен появления самых первых ЭВМ информация, хранящаяся в электронной форме, разделена на код (исполняемые инструкции) и данные. Код представляет собой императивные инструкции, выполняемые процессором ЭВМ, а данные – ту информацию, с которой выполняются операции. Первоначально вся логика выполнения программы содержалась только в коде; затем, по мере развития различных платформ для создания приложений, баз данных и промежуточного программного обеспечения, граница между кодом и данными начала размываться. На сегодняшний день исполняемый код большинства приложений представляет собой сложно организованную иерархию специализированных и достаточно абстрактных алгоритмов, порядок применения которых в значительной мере определяется настройками, то есть, в сущности, данными.

Тем не менее в программном коде прикладных автоматизированных систем все еще содержится много логики, связанной непосредственно с предметной областью и конкретными решаемыми задачами. Это связано с широким распространением парадигмы объектно ориентированного программирования и архитектуры MVC, при использовании которых в коде создаются программные объекты, описывающие свойства моделируемых объектов реального мира и операции, которые могут выполняться с этими «цифровыми образами» в виртуальной среде. Очевидным ограничением этого подхода является необходимость модифицировать код при внесении изменений в структуру модели и логику выполняемых операций. Также проблемой является то, что при создании программного обеспечения с использованием указанных технологий часто не прилагаются специальные усилия для построения модели

предметной области; модель формируется эволюционным путем, без методологии и плана. Часто авторами модели оказываются программисты, а не аналитики.

Использование онтологических моделей в программном обеспечении, в первом приближении кажется ИТ-специалистам похожим на подключение нового типа хранилища данных, с которым можно работать, не меняя принципов создания самого программного кода. Такой подход возможен и отчасти оправдан облегчением задачи освоения новых технологий для разработчиков. Однако для полного раскрытия потенциала онтологий как средства создания программного обеспечения необходимо изменение практики разработки ПО, которое влечет за собой изменение отношения программистов и аналитиков к онтологическим моделям и существенно влияет на процесс их работы.

### **Архитектура решения**

Выше были приведены примеры того, как частью онтологической модели становятся не только описания типов объектов предметной области и их связей, но и настройки, правила работы автоматизированной системы. Даже если эти правила представлены не в виде правил логического вывода SWRL или SPIN, а в виде обычных объектов онтологии, описывающих те или иные элементы алгоритмов, они фактически принимают на себя часть той роли, которую традиционно исполняет программный код.

Итак, содержимым онтологической модели могут являться:

- описания типов (классов), свойств и связей объектов;
- утверждения о конкретных объектах – «цифровых двойниках» объектов реального мира, значениях их свойств и связей;
- правила получения новых суждений об объектах на основании имеющихся сведений (правила логического вывода);
- правила взаимодействия объектов разных типов между собой в рамках имитационной модели;
- правила обработки информационных объектов различных типов в автоматизированной системе, то есть формализованные

#### Сценарий 4: Разработка программного обеспечения, управляемого онтологией

представления расчетных алгоритмов, описания пользовательских интерфейсов и т. д.;

– правила и регламенты обмена данными, правила формирования информационных сообщений, маршрутизации интеграционных потоков и многое другое.

Важно, что при создании в онтологии правил обработки информационных объектов элементы описаний этих правил будут ссылаться на классы и свойства модели предметной области. Таким образом, модель предметной области и модель процессов обработки информации в системе технологически неразрывны. При изменении модели предметной области автоматически меняются и алгоритмы обработки: так, например, при переносе класса из одного надкласса в другой могут измениться правила обработки объектов этого класса, если они отличаются для двух надклассов.

Приведем небольшой фрагмент отношений между элементами описания предметной области и описания алгоритмов работы системы (рис. 6.4).

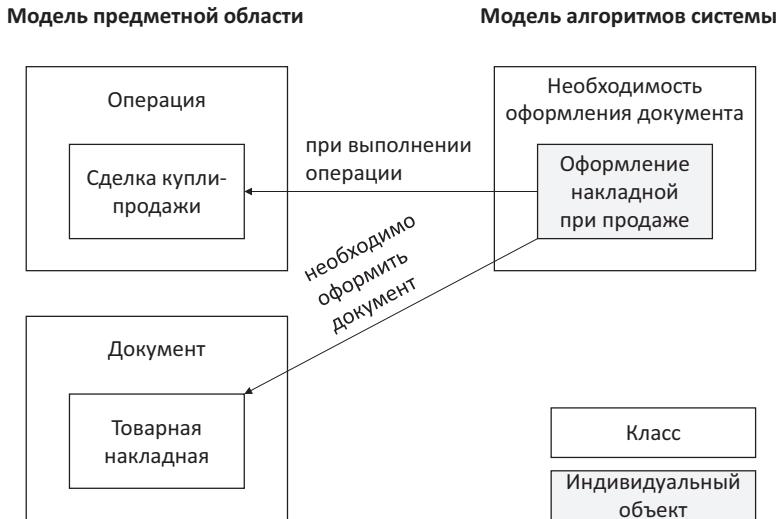


Рис. 6.4. Фрагмент модели, используемой автоматизированной системой, управляемой моделью

Элементы модели, показанные на диаграмме, интерпретируются следующим образом. Индивидуальный объект «Оформление накладной при продаже», относящийся к классу «Необходимость оформления документа», являющегося одним из подклассов класса правил, выражает тот факт, что при выполнении операции вида «Сделка купли-продажи» необходимо оформить документ типа «Товарная накладная». Вид операции и тип документов являются классами модели, то есть система создает индивидуальные объекты этих классов для отражения конкретных операций и документов. Разумеется, реальная модель включала бы, кроме показанных элементов, правила заполнения реквизитов документа, интерфейсную форму для ввода необходимых сведений, правила валидации введенных значений, правила вывода печатной формы заполненного документа.

Каким должен быть программный код, работающий с подобными моделями? На наш взгляд, конечной целью создания автоматизированных систем, управляемых моделью, является как можно большее абстрагирование программного кода от привязки к предметной области и конкретным вариациям алгоритмов обработки информации. Система, построенная по таким принципам, максимально гибка и устойчива к изменению функциональных требований, знаний о предметной области, ситуации на автоматизируемом предприятии и его задач. В идеальном случае, при изменении бизнес-процессов, влекущих за собой изменение требований к автоматизированной системе, аналитик должен отражать эти изменения в модели системы, не обращаясь к программистам и не взаимодействуя с программным кодом. Программный код должен быть готов к изменению модели, должен автоматически перестраивать свою работу (включая пользовательские интерфейсы) в соответствии с ними.

С экономической точки зрения реализация подобные системы строятся быстрее, чем традиционные, а их сопровождение и модификация (вносящие основной вклад в совокупную стоимость владения системой) оказываются существенно более выгодными.

Быстрая перестройка автоматизированных систем под требования бизнеса позволяет предприятию адекватнее реагировать на изменения внешней среды и собственных бизнес-процессов, внедрять организационные изменения, не ожидая в течение многих месяцев необходимых доработок приложений, а значит, эффективнее реализовывать свои конкурентные преимущества, повышать прибыль и снижать издержки.

Создание таких систем является не теоретической идеей, а реальностью, воплощенной в выполненных нами проектах.

Дальнейшее развитие идеи автоматизированных систем, управляемых моделью, состоит в создании алгоритмов «самообучения», благодаря которым система сможет сама модифицировать используемые ей алгоритмы в зависимости от поступающих сведений. Самым простым примером является оптимизация интерфейсных форм в зависимости от наблюдаемой реакции пользователя на работу с ними. Более сложные примеры могут включать автоматизированное формирование алгоритмов обработки информации таким образом, чтобы достигать желаемого результата наиболее оптимальным путем. Подобным образом работают рас пространенные сегодня методы машинного обучения, но коренное отличие от них предлагаемых методов состоит в том, что в процессе обучения система формирует алгоритмы, записываемые в терминах онтологической модели, верифицируемые и понятные пользователю. Это крайне важно для преодоления отчуждения между автоматизированной системой и пользователем, сохранения объяснимости ее работы для человека – эти свойства особенно важны для систем поддержки принятия решений. Пример способа реализации такого самообучающегося алгоритма приведен в работе [7].

Подводя итог нашему обзору, заметим, что методы и технологии онтологического моделирования, несмотря на более чем десятилетнюю историю существования, на момент создания этой книги еще не достигли стадии широкомасштабного распространения и полного освоения всех предоставляемых ими преимуществ.

Причина этого, на наш взгляд, состоит в высокой интеллектуальной сложности работ по созданию и применению онтологических моделей. Это несколько противоречит сложившейся в обществе тенденции к упрощению любых видов деятельности, сведению профессиональной подготовки к усвоению ограниченного набора инструкций, применимых в тех или иных ситуациях. Следствием этого является недостаток кадров для массового освоения описываемых технологий и предубеждение против них у руководителей, предпочитающих «простые и проверенные» решения.

Онтологическое моделирование остается фронтиром, достойной точкой приложения сил для исследователей и практиков, ищущих пути радикального повышения эффективности любой практической деятельности. На основании опыта, накопленного сообществом онтологов, мы вправе заявить, что вознаграждением за сложности, преодоленные на этом пути, является реализация подлинно интеллектуальных и гибких программных инструментов, позволяющих их пользователям достичь принципиально новых преимуществ.

### **Список библиографических ссылок**

1. Ontology Based Data Access in Statoil/E. Kharlamov et al.// Web Semantics: Science, Services and Agents on the World Wide Web. 2017. Vol. 44. P. 3–36.
2. Непейвода Н. Н. Некоторые семантические конструкции конструктивных схем программ // Теория алгоритмов и ее приложения (Вычислительные системы), 1989. Вып. 129. С. 49–66.
3. Грегер С. Э., Поршинев С. В. Построение онтологии архитектуры информационной системы // Фундаментальные исследования. 2013. № 10. С. 2405–2409.
4. Application of semantic knowledge management system in selected areas of Polish public administration/A. Wróblewska, A. Zięba, R. Mieńkowska-Norkiene, P. Kapłanński, P. Zarzycki // Collegium of Economic Analysis Annals, Warsaw. 2013. P. 353–368.
5. The Application of Semantic Web Technologies for Railway Decision Support/J. Lu, C. Roberts, K. Lang, A. Stirling, K. Madelin // Intelligent

---

---

Decision-making support systems: Foundations, Applications and Challenges/eds. J. Gupta, G. Forgionne, M. Mora. Springer, 2006. P. 321–341.

6. Онтология медицинской диагностики для интеллектуальных систем поддержки принятия решений/В. В. Грибова, М. В. Петряева, Д. Б. Окунь, Е. А. Шалфеева// Онтология проектирования. 2018. Т. 8, № 1(27). С. 58–73.

7. Горшков С. В., Гребешков А. Ю. Разработка способа анализа сообщений о происшествиях в автоматизированной системе управления безопасностью жизнедеятельности // Инфокоммуникац. технологии. 2018. Т. 16, № 2. С. 221–230.

# ПРОГРАММНЫЙ ИНСТРУМЕНТАРИЙ ОНТОЛОГИЧЕСКОГО МОДЕЛИРОВАНИЯ. ПРАКТИКУМ

*P. Ю. Шебалов*

## Редакторы онтологий и их возможности

Успешность реализации информационной модели во многом определяется тем инструментарием, который выбран для моделирования онтологии. В настоящее время разработано достаточно много редакторов онтологий, что, с одной стороны, предоставляет широкое поле выбора для аналитиков, с другой – может усложнить этот выбор. Обзоры редакторов онтологий приведены в работах [1; 2].

Одним из наиболее распространенных бесплатных редакторов является Protégé. Он широко востребован в академической среде и среди энтузиастов, начинающих осваивать принципы и технологии работы с онтологиями. Однако характерная для Protégé установка на работу с «низким уровнем» онтологической модели, то есть с синтаксическими конструкциями стандартов с минимальной интерфейсной помощью для пользователя, по нашему мнению, приводит к его усложнению интерфейса (рис. 1) и сужает круг аналитиков, способных воспользоваться редактором. Также удобство навигации и производительность Protégé страдают при работе с большими онтологиями.

В частности, разнесение по отдельным вкладкам классов, свойств объектов, свойств данных, свойств аннотаций, типов данных, объектов препятствует восприятию онтологии как целостной графовой структуры (рис. 2, 3).

Существует и веб-версия редактора, WebProtégé, позволяющая некоторым пользователям вести совместную работу над онтологией. Интерфейс веб-версии построен на тех же принципах, что и интерфейс основной версии Protégé.

## Программный инструментарий онтологического моделирования. Практикум

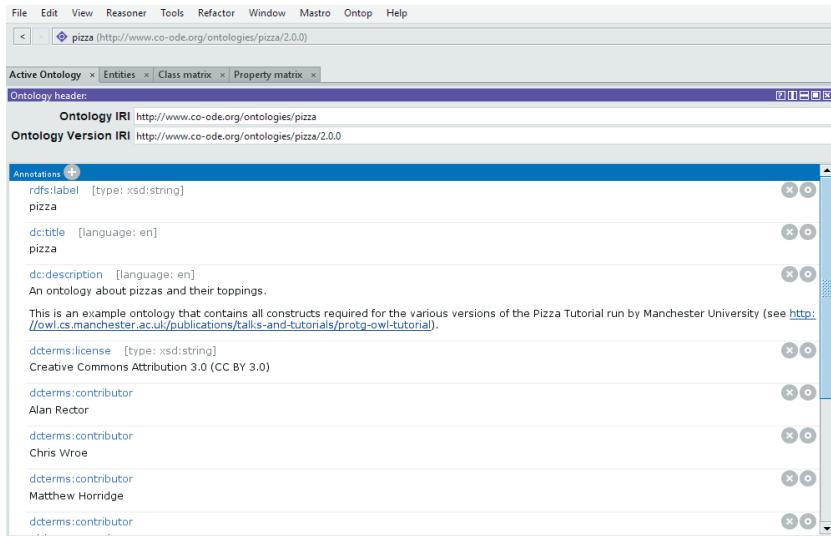


Рис. 1. Общая структура пользовательского интерфейса Protégé (фрагмент)

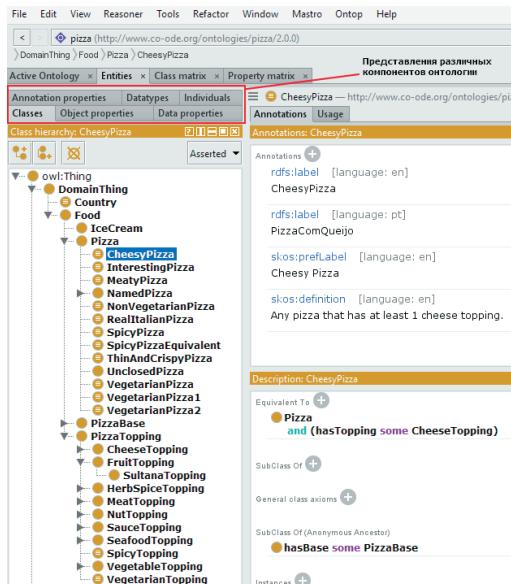
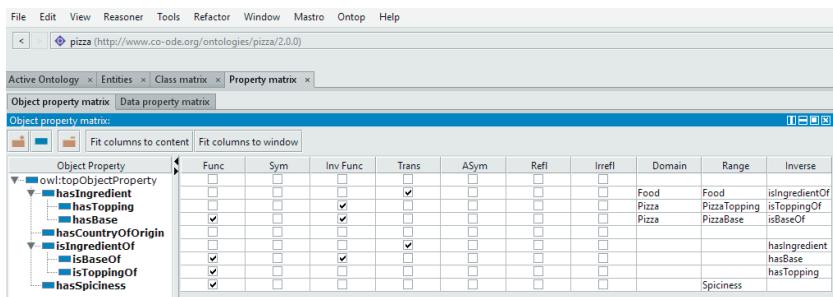


Рис. 2. Классы онтологии в Protégé (фрагмент)

## Приложение

---



The screenshot shows the Protégé interface with the following details:

- File Bar:** File, Edit, View, Reasoner, Tools, Refactor, Window, Mastro, Ontop, Help.
- Title Bar:** pizza (<http://www.co-ode.org/ontologies/pizza/2.0.0>)
- Tab Bar:** Active Ontology, Entities, Class matrix, Property matrix (selected).
- Sub-Tab Bar:** Object property matrix, Data property matrix.
- Panel Header:** Object property matrix.
- Table:** A grid showing characteristics of object properties. The columns are labeled: Func, Sym, Inv Func, Trans, ASym, Refl, Irrefl, Domain, Range, and Inverse. The rows list various properties under the owl:topObjectProperty category, such as hasIngredient, hasTopping, hasBase, hasCountryOfOrigin, isIngredientOf, isBaseOf, isToppingOf, and hasSpiciness. The table also includes columns for Food, Pizza, PizzaTopping, PizzaBase, and Spiciness.

Рис. 3. Матрица характеристик свойств в Protégé (фрагмент)

Отдельное место в ряду редакторов онтологий занимает Hozo (<http://hozo.jp/>), в первую очередь в силу того, что его разработка была мотивирована стремлением предоставить онтологический инструмент для репрезентации концепта «Роль» в контексте решения задач по информационному инжинирингу. Основополагающие подходы к разработке редактора Hozo представлены в [3; 4].

Hozo позволяет создавать онтологические модели (рис. 4), выделяя роли в качестве их отдельных элементов.

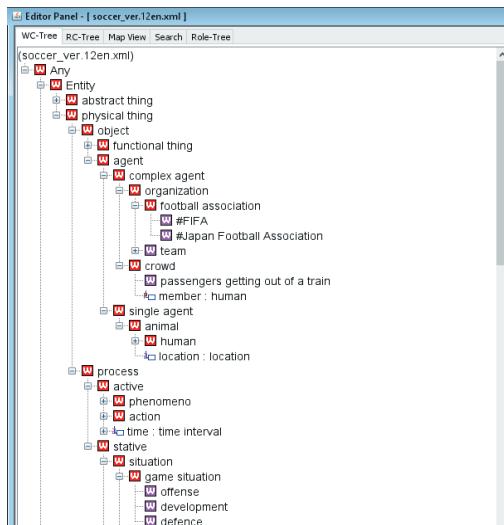


Рис. 4. Онтология футбола в Hozo (фрагмент)

## Роли в редакторе Hozo выводятся в отдельной вкладке (рис. 5).

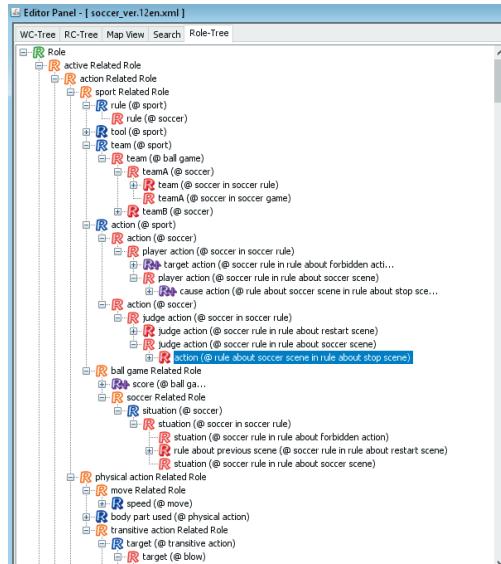


Рис. 5. Иерархия ролей в онтологии футбола в Hozo (фрагмент)

Каждая роль визуально связывается с другими сущностями модели (рис. 6).

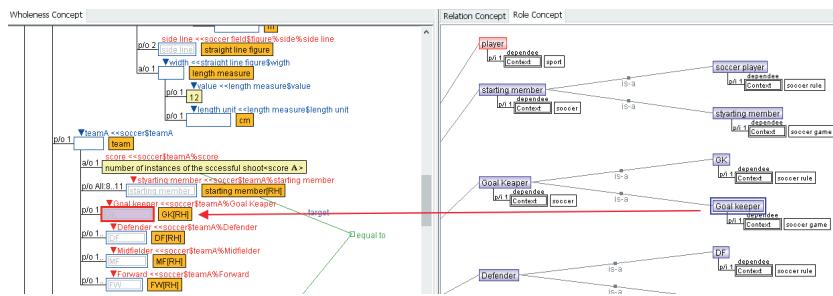


Рис. 6. Связанный просмотр сущностей и их ролей в онтологии футбола в Hozo (фрагмент)

Опора на визуальную поддержку работы с онтологией является одной из главных характеристик редактора Hozo, однако это

оказывается и его ограничением: при построении большой модели в ней становится достаточно сложно ориентироваться, и ее просмотр в пригодном для чтения масштабе затруднителен (рис. 7).

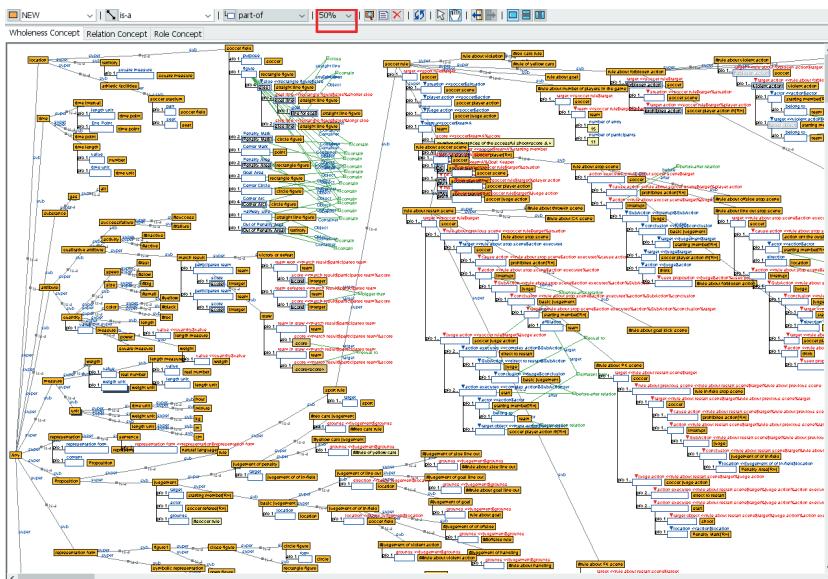


Рис. 7. Онтология футбола в Hozo в масштабе 50% (фрагмент)

По нашему мнению, в настоящее время Protégé и Hozo на рынке редакторов представляют собой достаточно удачные попытки реализовать инструмент для построения онтологии «как она есть» или «какой она должна быть». Это, безусловно, необходимо для демонстрации возможностей онтологического подхода к обработке информации, в том числе в академическом контексте, однако вряд ли эффективно, когда речь идет о реальной практике разработки онтологии и ее использовании в конкретном промышленном проекте. Особенно важно для редакторов, предназначенных для использования в таких проектах, наличие колаборативных возможностей, позволяющих нескольким пользователям совместно и одновременно вести работу над одной моделью.

Существует ряд редакторов онтологий помимо представленных выше, однако большинство из них являются коммерческими продуктами

и/или не имеют русской локализации. С функциональной же точки зрения Protégé и Нозо представляют собой два условных «полюса», относительно которых можно позиционировать остальные редакторы.

Очевидно, что выбор того или иного онтологического редактора целесообразно основывать на степени полноты набора инструментов для моделирования, обеспечивающей семантическую выразительность в соответствии с требованиями стандарта OWL, удобстве визуального интерфейса, наличии возможностей совместной работы.

Созданный коллективом российской компании «Тринидата» редактор онтологии onto.pro (<http://onto.pro/>) представляет собой веб-приложение, доступное для индивидуального бесплатного использования в «облачном» варианте. Коммерческие заказчики могут приобрести версию редактора для установки на собственные сервера. Редактор ориентирован на совместную работу аналитиков над онтологической моделью, находящейся в хранилище триплетов (RDF triple store). Редактор обеспечивает функционал для создания, редактирования и удаления классов, свойств, индивидуальных объектов, выполнения групповых операций с этими элементами, позволяет экспортить/импортировать модель через файлы формата Excel, содержит средства для облегчения рефакторинга модели, поиска элементов и др. Редактор абстрагирует пользователя от технических деталей стандартов онтологического моделирования, что расширяет круг аналитиков, способных использовать этот инструмент. Обратной стороной такого стремления является отсутствие реализации в текущей версии (на начало 2019 года) некоторых возможностей стандарта OWL, например построения иерархии свойств, указания характеристик свойств (транзитивность и др.) и классов.

В настоящее время ведется работа над новой версией редактора, призванной реализовать эти возможности при сохранении в интерфейсе стремления отразить логическую, а не технологическую структуру онтологии. Доступны руководства пользователя в краткой<sup>1</sup> и полной<sup>2</sup> версиях.

---

<sup>1</sup> <http://onto.pro/files/Modeling.pdf>.

<sup>2</sup> <https://trinidata.ru/files/OntoUserGuide.pdf>.

Далее мы рассмотрим сценарий моделирования инфраструктуры предприятия в редакторе Onto.pro, а затем познакомимся с моделированием логических утверждений при помощи инструментария Protégé.

### Моделирование онтологии ТЭС средствами редактора onto.pro

В этом разделе представлен вариант реализации средствами редактора onto.pro онтологической модели на примере онтологии тепловой электрической станции (ТЭС).

Принципиальная схема физических компонентов ТЭС в упрощенном виде представлена на рис. 8.

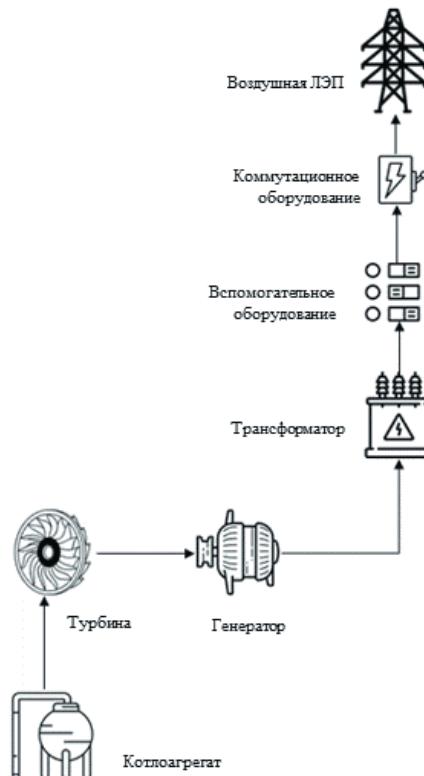


Рис. 8. Принципиальная схема компонентов ТЭС

На схеме отображен процесс взаимодействия структурно значимых компонентов оборудования, обеспечивающего генерацию, трансформацию и передачу электроэнергии. Чтобы не перегружать схему излишними деталями, в ней не представлены этапы диагностики качества электроэнергии, а также передачи ее потребителям (внутренним и внешним).

При входе в систему пользователь оказывается в рабочем пространстве, состоящем из трех основных блоков:

- набора классов онтологии, организованного в виде дерева (рис. 9);
- блока редактирования экземпляров классов и из свойств, организованного в виде списка (рис. 10);
- блока (формы) просмотра и редактирования параметров выбранного объекта класса или его свойства (рис. 11).

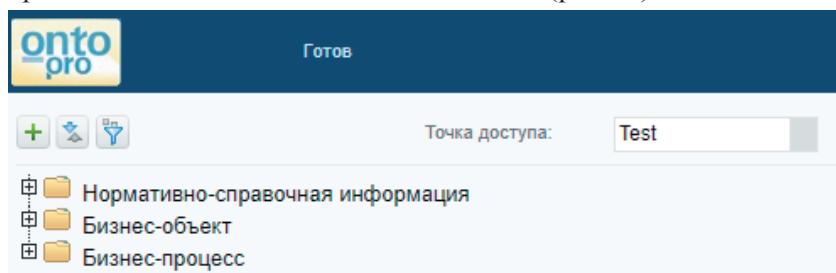


Рис. 9. Дерево классов онтологии

Объект	Свойство	Указатель	
<input checked="" type="checkbox"/> Идентификатор			Название
<input type="checkbox"/> Гост1943184			ГОСТ 19431-84 Энергетика и электрификация
<input type="checkbox"/> Гост2326978			ГОСТ 23269-78 Турбины стационарные паровые
<input type="checkbox"/> Гост2329078			ГОСТ 23290-78 Установки газотурбинные стационарные
<input type="checkbox"/> Гост242782016			ГОСТ 24278-2016 Установки турбинные паровые стационарные для привода электрогенераторов ТЭС
<input type="checkbox"/> Пуэ			Правила устройства электроустановок (ПУЭ)

Рис. 10. Список объектов выбранного класса онтологии

## Приложение

ГОСТ 23269-78 Турбины стационарные паровые

Предфикс: http://trinidat.ru/suzb/  
Автор: Ронан Шевалов

Сохранить  Удалить

Сущность - Индивидуальный объект (экземпляр)

Тип \*  Нормативные документы

Идентификатор Гост2326978

Название ГОСТ 23269-78 Турбины стационарные паровые

Определение

Определено в

Удаленный   
Поля, отмеченные \*, обязательны для заполнения.

Сохранить  Удалить

Связи объекта

Комментарии

Файлы

Рис. 11. Просмотр и редактирование параметров выбранного объекта класса

Отображение в одном блоке экземпляров класса, свойств-литералов и свойств-ссылок (рис. 12) обеспечивает компактное представление элементов онтологии.

Кнопки переключения  
Объект / Свойство-литерал / Свойство-ссылка

Объект	Свойство	Указатель
<input checked="" type="checkbox"/> Идентификатор	Название	
<input type="checkbox"/> Гост1943184	ГОСТ 19431-84 Энергетика и элэктрификация	
<input type="checkbox"/> Гост2326978	ГОСТ 23269-78 Турбины стационарные паровые	
<input type="checkbox"/> Гост2329078	ГОСТ 23290-78 Установки газотурбинные стационарные	
<input type="checkbox"/> Гост242782016	ГОСТ 24278-2016 Установки турбинные паровые стационарные для привода электрогенераторов ТЭС	
<input type="checkbox"/> Пуэ	Правила устройства электроустановок (ПУЭ)	

Рис. 12. Переключение между типами элементов

Моделирование онтологии ТЭС как разновидности промышленной онтологии предполагает выделение как минимум трех классов верхнего уровня: **Бизнес-объект**, **Бизнес-процесс**, **Нормативно-справочная информация**.

Класс **Бизнес-объект** включает в себя подклассы, описывающие сущности, характеристики которых актуальны неопределенно долгое время (рис. 13).

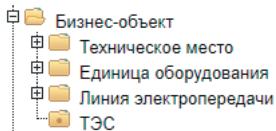


Рис. 13. Подклассы класса «Бизнес-объект»

Класс **Бизнес-процесс** включает подклассы, описывающие действия над бизнес-объектами (рис. 14). Данные действия (например, ремонт, техническое обслуживание и т. п.) имеют привязку к моменту времени, их характеристики актуальны на протяжении определенного периода.

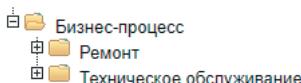


Рис. 14. Подклассы класса «Бизнес-процесс»

Разделение сущностей на объектные и процессные (условно «статические и динамические») позволяет моделировать не только компонентную составляющую промышленной онтологии, но и описывать работы, реализованные с помощью/в отношении данных компонентов.

Класс **Нормативно-справочная информация** включает отраслевые документы и справочники (рис. 15), необходимые для семантической полноты описания какого-либо объекта или свойства в онтологии. Также к данному классу относятся классификаторы и справочники параметрических характеристик бизнес-объектов (например, классы напряжения, типы изоляции, типы и марки оборудования и т. д.).

## Приложение

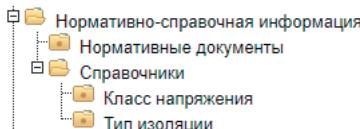


Рис. 15. Подклассы класса «Нормативно-справочная информация»

В качестве подклассов класса Бизнес-объект созданы классы Техническое место, Единица оборудования, ТЭС и ЛЭП. Последние два класса представляют образы своих физических прототипов – объектов генерации и передачи электрической энергии.

Класс Единица оборудования выступает в качестве контейнера для подклассов, включающих типы оборудования – трансформаторы, коммутационное оборудование, токоведущее оборудование и т. д. (рис. 16)

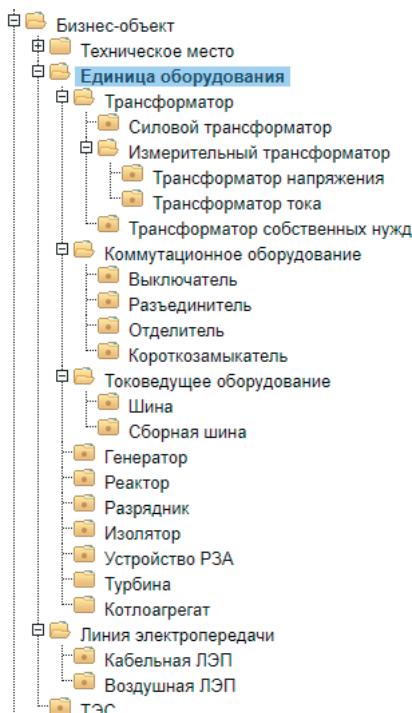


Рис. 16. Подклассы класса «Единица оборудования»

Класс **Техническое место** моделирует функциональные позиции оборудования определенного типа в общей схеме технологической инфраструктуры. Техническое место является «гнездом» для размещения конкретной единицы оборудования. Характеристики технического места описывают функциональные требования к характеристикам единицы оборудования, которая может быть установлена на это место.

На каждом техническом месте (рис. 17) устанавливаются конкретные экземпляры единиц оборудования, которые на протяжении своего жизненного цикла переживают различные процессы: диагностику, ремонты, замену, модернизацию и др. В разные периоды времени на одном техническом месте могут находиться различные единицы оборудования.

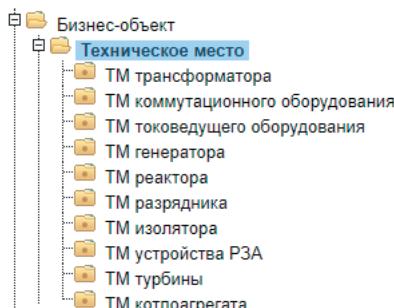


Рис. 17. Подклассы класса «Техническое место»

Выделять технические места ЛЭП и ТЭС нецелесообразно, так как эти объекты относятся к типу сложных, состоящих из множества функциональных и физических под-объектов, каждый из которых в том числе имеет свой регламент технического обслуживания и ремонта. Поэтому выделение классов ТМ ЛЭП и ТМ ТЭС вряд ли возможно без существенного снижения уровня детализации информации.

Выделение отдельных классов для учета технических мест и единиц оборудования позволяет строить не статическую картину состояния технологической инфраструктуры на определенный

момент времени, а динамическую, разворачивающуюся во времени. Такие модели позволяют определить состояние всей моделируемой системы на любой момент времени, что важно для решения аналитических и прогнозных задач.

### Моделирование объектов и свойств

Рассмотрим один из вариантов моделирования объектов, набора свойств и связей объектов между собой. Представим, что на моделируемой ТЭС находится три силовых трансформатора, два из которых установлены на технических местах, один трансформатор находится в резерве, одно техническое место вакантно. Силовые трансформаторы и их технические места обладают определенными наборами характеристик (рис. 18).

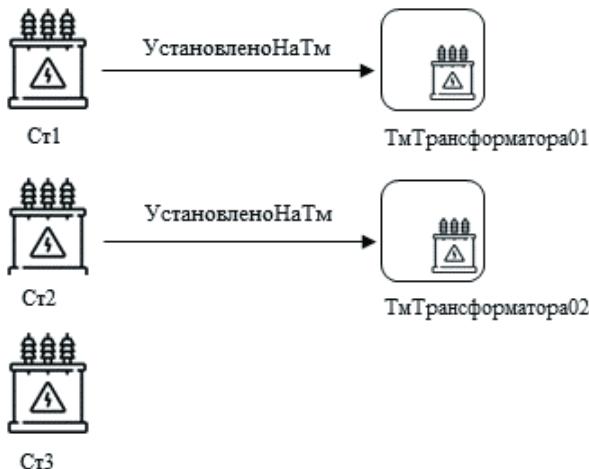


Рис. 18. Установка трансформаторов на технических места

Онтологическое описание такой ситуации предполагает несколько шагов:

1. Создать класс Силовой трансформатор (рис. 19).
2. Создать три объекта класса Силовой трансформатор (рис. 20).
3. Далее необходимо создать набор свойств объектов класса Силовой трансформатор.

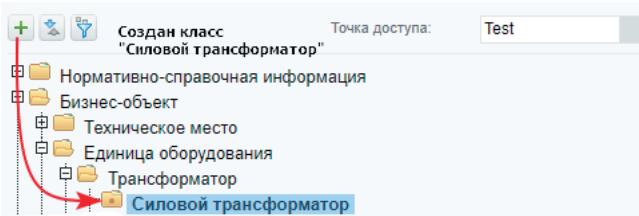


Рис. 19. Создание класса «Силовой трансформатор»

	Название	Идентификатор		
	Объект	Свойство	Указатель	
<input checked="" type="checkbox"/>	Силовой трансформатор 1	Ct1		
<input type="checkbox"/>	Силовой трансформатор 2	Ct2		
<input type="checkbox"/>	Силовой трансформатор 3	Ct3		

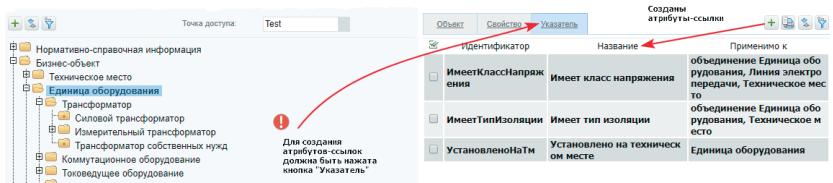
Для создания объектов должна быть нажата кнопка "Объект"

Рис. 20. Создание объектов класса «Силовой трансформатор»

Редактор onto.pro по умолчанию поддерживает наследование свойств от надклассов. На практике это означает, что если свойство-литерал или свойство-ссылка созданы для надкласса, то они становятся свойствами подклассов без дополнительных усилий пользователя. Данные свойства могут (но не обязательно должны) быть задействованы при моделировании с использованием подклассов.

Такой принцип удобен при моделировании сущностей, связанных родо-видовыми отношениями. В данном случае силовой трансформатор является видом трансформаторов, которые, в свою очередь, являются видом единиц оборудования. Поэтому свойства, применимые к объектам надкласса Единица оборудования применимы и для объектов класса Силовой трансформатор и должны быть привязаны к надклассу. Для этого при создании свойств в редакторе нужно выбрать нужный надкласс в дереве, нажать кнопку «Указатель» над списком в средней части экрана, затем нажать на расположенную правее кнопку «+» для создания нового свойства (рис. 21).

## Приложение



*Rис. 21. Создание набора свойств*

При просмотре класса Силовой трансформатор приведенные выше свойства отображаются как унаследованные (рис. 22).

Бизнес-объект \ Единица оборудования \ Трансформатор \ **Силовой трансформатор**  
Предфикс: http://trinidata.ru/suzb/  
Автор: Роман Шебалов

**Сущность - Класс**

Идентификатор	СиловойТрансформатор
Название	Силовой трансформатор

**Надклассы**

Определение	
Определено в	
Удаленный	<input type="checkbox"/>
Подкласс для *	<input type="checkbox"/> Трансформатор

**Свойства**

Имеет класс напряжения	<b>ИмеетКлассНапряжения</b> Унаследован от Единица оборудования
Имеет тип изоляции	<b>ИмеетТипИзоляции</b> Унаследован от Единица оборудования
Количество обмоток	<b>КоличествоОбмоток</b> Унаследован от Трансформатор
Находится в резерве	<b>НаходитсяВРезерве</b> Унаследован от Единица оборудования
Установлено на техническом месте	<b>УстановленоНаТм</b> Унаследован от Единица оборудования

поля, отмеченные \*, обязательны для заполнения

*Rис. 22. Унаследованные свойства*

При необходимости пользователь может создать свойства-литералы или свойства-ссылки, специфичные для данного подкласса. Например, для силовых трансформаторов таким свойством может быть ЯвляетсяАвтотрансформатором (boolean). Такое свойство

будет применено для данного класса и всех его подклассов и не влияет на набор свойств надкласса.

Свойства-литералы и свойства-ссылки имеют некоторые различия при создании. В связи с тем что свойство-литерал передает некоторое физическое значение (число, дату, текст, логическое значение), при его создании необходимо в явном виде указать тип данных.

Например, если для класса Трансформатор необходимо создать свойство КоличествоОбмоток, то типом данных для данного свойства будет целое число. Данный параметр указывается в поле «Диапазон значений» (рис. 23).

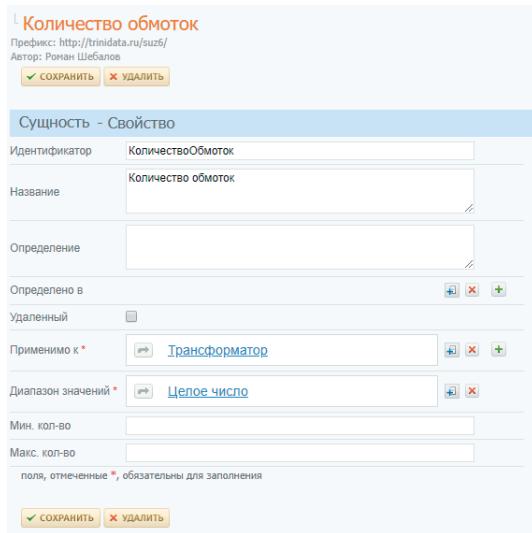


Рис. 23. Создание свойства-литерала

В полях «Мин. количество», «Макс. количество» при необходимости могут быть указаны числа, задающие ограничения на число значений (**Cardinality**) при указании конкретных значений данного свойства какого-либо объекта.

Свойство-ссылка отражают в модели логические, фактические, структурные, функциональные и другие связи между сущностями,

составляющими информационную модель. Визуально свойство-ссылка может быть представлена в виде стрелки на диаграмме, направленной от одного элемента к другому. Такие свойства обеспечивают связывание, необходимое как для описания зависимостей между элементами модели, так и для вычисления необходимых значений. Например, чтобы получить количество двухобмоточных трансформаторов, установленных на данной ТЭС и прошедших техническое освидетельствование в период с 1999 по 2004 год, необходимо получить запросом к модели объекты, представляющие двухобмоточные трансформаторы, имеющие связь с объектом ТЭС, вместе с объектами, представляющими сведения о техническом освидетельствовании. Такой запрос является классическим примером запроса к графу, результатом которого является несколько вершин графа, связанных определенными свойствами.

Создание свойства-ссылки требует выбора класса онтологии, выступающего в качестве диапазона значений, то есть целевого класса. Это подразумевает, что необходимый целевой класс должен быть создан перед созданием свойства-ссылки (рис. 24).

Л Установлено на техническом месте  
Префикс: http://timedata.ru/suzb/  
Автор: Ронан Шебалов  
Сохранить Удалить

Сущность - Свойство - указатель на объект

Идентификатор	УстановленоНаТм
Название	Установлено на техническом месте
Определение	Техническое место, на котором установлена данная единица оборудования
Определено в	<input type="checkbox"/> Удаленный
Применимо к *	<input type="checkbox"/> Единица оборудования
Диапазон значений *	<input type="checkbox"/> Техническое место
Мин. кол-во	
Макс. кол-во	

поля, отмеченные \*, обязательны для заполнения

Рис. 24. Создание свойства-ссылки

Как и для свойства-литерала, можно указать ограничение на число связей данного типа, которыми может обладать конкретный объект.

В ряде случаев несколько классов могут иметь одно и то же свойство-ссылку – например, если объекты разных классов могут выступать в одинаковом отношении к объектам третьего класса (рис. 25).

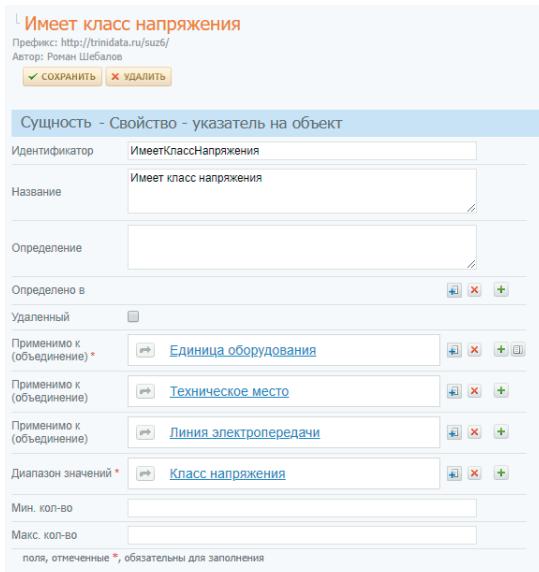


Рис. 25. Несколько классов используют одно свойство-ссылку

В данном случае это означает, что объекты классов Единица оборудования, Техническое место, Линия электропередачи и их подклассов имеют характеристику Класс напряжения, значение которой берется из справочника, хранящегося в виде набора объектов класса Класс напряжения.

4. Создать класс ТМ трансформатора (рис. 26);
5. Создать три объекта класса ТМ трансформатора (рис. 27);
6. Далее необходимо создать набор свойств объектов класса ТМ трансформатора.

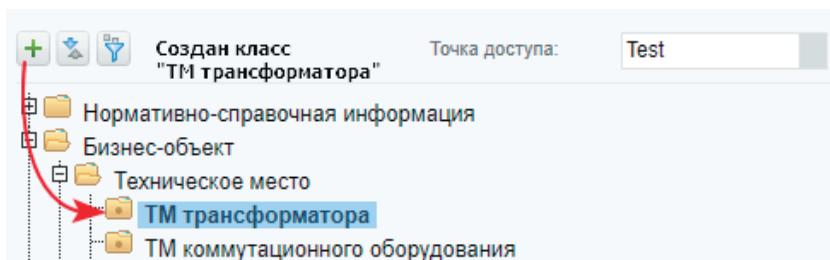


Рис. 26. Создание класса «TM трансформатора»

Объект	Свойство	Указатель	Созданы объекты класса	Идентификатор
<input checked="" type="checkbox"/>	Название			
<input type="checkbox"/>	TM трансформатора 01		ТмТрансформатора01	
<input type="checkbox"/>	TM трансформатора 02		ТмТрансформатора02	
<input type="checkbox"/>	TM трансформатора 03		ТмТрансформатора03	

Рис. 27. Создание объектов класса «TM трансформатора»

Набор свойств (литералов и ссылок) этого класса целесообразно создать у надкласса Техническое место, они наследуются подклассом TM трансформатора (рис. 28).

Идентификационный номер технического места	<a href="#">ИдентификационныйНомерТм</a> Унаследован от <a href="#">Техническое место</a>
Имеет класс напряжения	<a href="#">ИмеетКлассНапряжения</a> Унаследован от <a href="#">Техническое место</a>
Имеет тип изоляции	<a href="#">ИмеетТипИзоляции</a> Унаследован от <a href="#">Техническое место</a>
Месторасположение технического места	<a href="#">Месторасположение</a> Унаследован от <a href="#">Техническое место</a>
Наименование организации, обслуживающей TM	<a href="#">НаименованиеОбслуживающейОрганизации</a> Унаследован от <a href="#">Техническое место</a>

Рис. 28. Свойства класса «TM трансформатора», унаследованные от надкласса

7. Далее нужно заполнить значения свойств объектов классов Силовой трансформатор и TM трансформатора.

После создания (или наследования от надкласса) свойств необходимо присвоить значения этих свойств конкретным объектам (рис. 29–31).

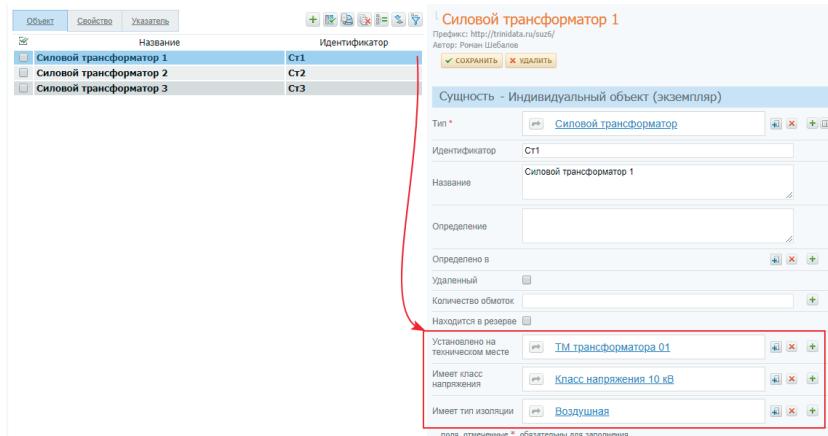


Рис. 29. Присвоение значений свойствам объекта «Силовой трансформатор 1»

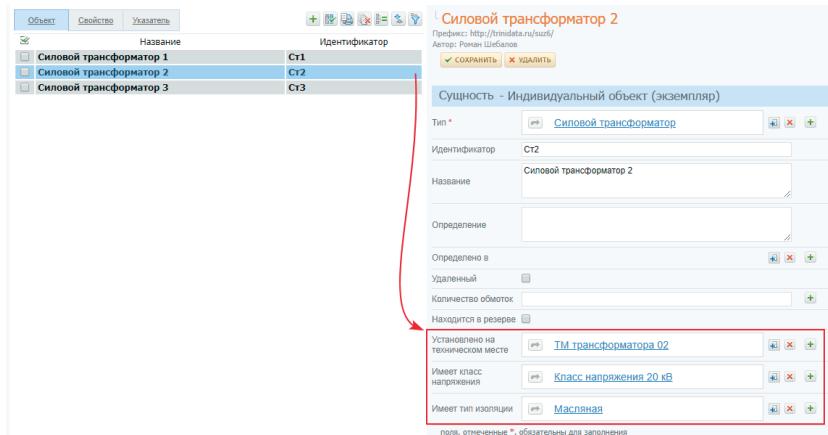


Рис. 30. Присвоение значений свойствам объекта «Силовой трансформатор 2»

Свойству-ссылке «Установлено на техническом месте» объекта «Силовой трансформатор 3» не присвоено значение, так как данный трансформатор находится в резерве и на техническое место

## Приложение

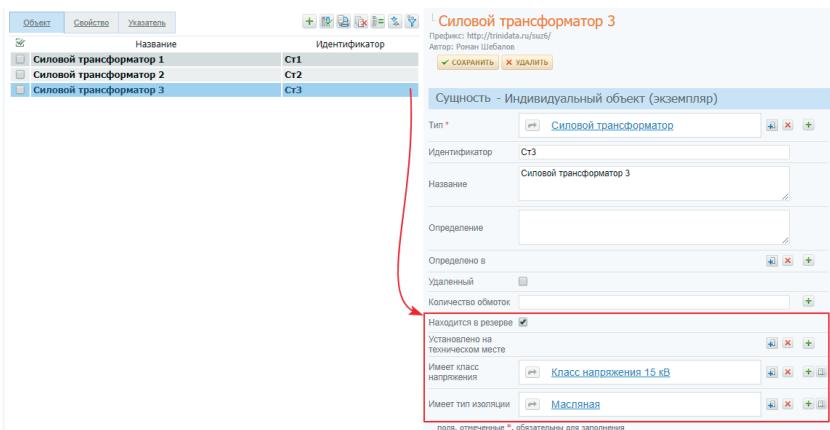


Рис. 31. Присвоение значений свойствам объекта «Силовой трансформатор 3»

не установлен. Таким образом, в результате заполнения значений свойств, силовые трансформаторы ТЭС будут описаны в онтологии в разрезе их текущих эксплуатационных параметров и функционального статуса в энергетической подсистеме ТЭС.

Чтобы просмотреть параметры технического места, на котором установлен тот или иной трансформатор, достаточно нажать на ссылку на имени объекта в форме (рис. 32).

После перехода по ссылке пользователь видит форму редактирования значений параметров выбранного технического места (рис. 33).

Если при работе с информацией об оборудовании данной ТЭС возникнет необходимость получить сведения о том, когда и кто принял решение об установке данной единицы оборудования на данном техническом месте, необходимо создать в онтологии класс, объекты которого могли бы хранить соответствующие значения.

Такой класс целесообразно назвать Установка ЕО на ТМ и сделать его подклассом класса Бизнес-процесс (рис. 34).

Цель создания данного подкласса – отделить хранение сведений о самом *оборудовании* (например, трансформаторах) от ситуаций,

## Программный инструментарий онтологического моделирования. Практикум

Силовой трансформатор 1  
Предфикс: <http://trinidata.ru/suz/>  
Автор: Роман Шебалов

Сохранить  Удалить

Сущность - Индивидуальный объект (экземпляр)

Тип *	<input type="button"/> Силовой трансформатор	<input type="button"/> <input type="button"/> <input type="button"/>
Идентификатор	Ct1	<input type="button"/> <input type="button"/> <input type="button"/>
Название	Силовой трансформатор 1	<input type="button"/> <input type="button"/> <input type="button"/>
Определение		<input type="button"/> <input type="button"/> <input type="button"/>
Определено в		<input type="button"/> <input type="button"/> <input type="button"/>
Удаленный	<input type="checkbox"/>	<input type="button"/> <input type="button"/> <input type="button"/>
Количество обмоток		<input type="button"/> <input type="button"/> <input type="button"/>
Находится в резерве	<input type="checkbox"/>	<input type="button"/> <input type="button"/> <input type="button"/>
Установлено на техническом месте	<input type="button"/> ТМ трансформатора 01	<input type="button"/> <input type="button"/> <input type="button"/>
Имеет класс напряжения	<input type="button"/> Класс напряжения 10 кВ	<input type="button"/> <input type="button"/> <input type="button"/>
Имеет тип изоляции	<input type="button"/> Воздушная	<input type="button"/> <input type="button"/> <input type="button"/>

поля, отмеченные \*, обязательны для заполнения

Рис. 32. Переход по ссылке к объекту онтологии

ТМ трансформатора 01  
Предфикс: <http://trinidata.ru/suz/>  
Автор: Роман Шебалов

Сохранить  Удалить

Сущность - Индивидуальный объект (экземпляр)

Тип *	<input type="button"/> ТМ трансформатора	<input type="button"/> <input type="button"/> <input type="button"/>
Идентификатор	ТмТрансформатора01	<input type="button"/> <input type="button"/> <input type="button"/>
Название	ТМ трансформатора 01	<input type="button"/> <input type="button"/> <input type="button"/>
Определение		<input type="button"/> <input type="button"/> <input type="button"/>
Определено в		<input type="button"/> <input type="button"/> <input type="button"/>
Удаленный	<input type="checkbox"/>	<input type="button"/> <input type="button"/> <input type="button"/>
Месторасположение технического места	B-A2-17/11	<input type="button"/> <input type="button"/> <input type="button"/>
Наименование организации, обслуживающей ТМ	ООО "Спецстроймонтаж"	<input type="button"/> <input type="button"/> <input type="button"/>
Идентификационный номер технического места	11	<input type="button"/> <input type="button"/> <input type="button"/>
Выполненный ремонт		<input type="button"/> <input type="button"/> <input type="button"/>
Имеет тип изоляции	<input type="button"/> Воздушная	<input type="button"/> <input type="button"/> <input type="button"/>
Имеет класс напряжения	<input type="button"/> Класс напряжения 10 кВ	<input type="button"/> <input type="button"/> <input type="button"/>

поля, отмеченные \*, обязательны для заполнения

Рис. 33. Параметры ТМ трансформатора 01

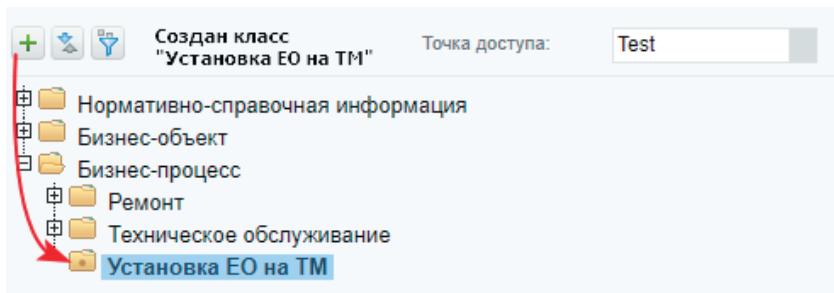


Рис. 34. Класс Установка EO на ТМ

когда нам необходимо хранить сведения о *действиях*, произведенных над этим оборудованием (установке на технических местах, ремонтах, технических освидетельствованиях и т. д.). Такое разделение необходимо, потому что за свой жизненный цикл любая единица участвует во множестве подобных действий. Информация о таких действиях является необходимой для анализа эксплуатационных, технологических, организационных и др. параметров работы любого объекта электроэнергетики.

В подклассе Установка EO на ТМ должен быть создан информационный объект, описывающий сведения о факте установки силового трансформатора на определенном техническом месте, а также о параметрах данного события (рис. 35).

Таким образом, сведения о данном трансформаторе и его техническом месте оказались обогащены информацией о событии его установки (датами, Ф.И.О. ответственного работника и наименованием структурного подразделения). Эти сведения могут быть использованы в дальнейшем, например при формировании автоматизированных отчетов о количестве и технических характеристиках трансформаторов, установленных на технических местах за определенный период времени под контролем определенного работника.

Аналогичным образом могут быть созданы классы Ремонт, Техническое обслуживание и пр., с помощью которых будут описываться соответствующие бизнес-процессы на ТЭС.

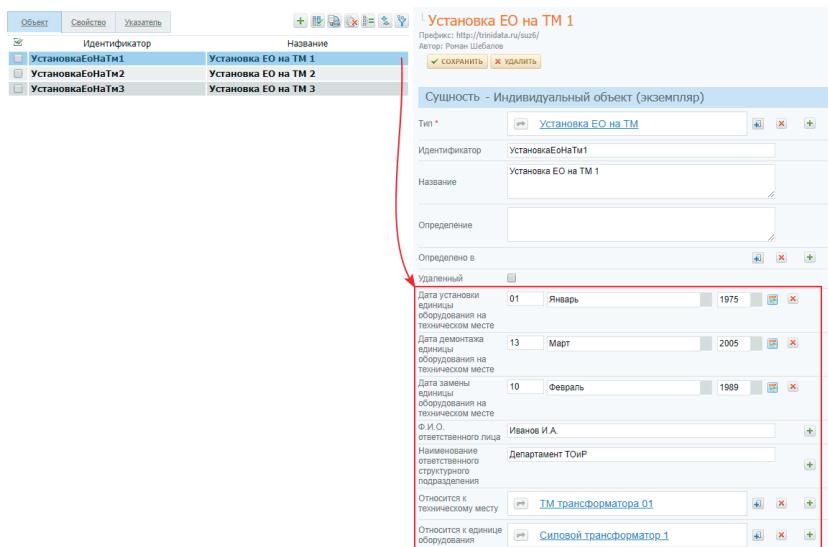


Рис. 35. Присвоение значений свойствам информационного объекта «Установка EO на TM 1»

Как демонстрирует рассмотренный пример, функциональных возможностей редактора онтологий onto.pro достаточно для создания насыщенных информационных моделей предметных областей. В реальных применениях сведения об индивидуальных объектах редко вводятся вручную – они или загружаются при помощи инструментов импорта из Excel, или поступают через различного рода интеграционные адаптеры, в частности через API продукта АрхиГраф.MDM, в связке с которым работает редактор onto.pro. Система АрхиГраф.СУЗ расширяет возможности работы с моделями, построенными в onto.pro, созданием и применением правил логического вывода. В этом продукте правила логического вывода строятся при помощи визуального конструктора.

## Моделирование логических утверждений в редакторе Protégé

Экспортируем построенную модель в редактор Protégé и продолжим знакомство с инструментами моделирования логических

утверждений. До сих пор мы описывали только один вид отношений между классами – отношение «надкласс-подкласс». Стандарт OWL позволяет конструировать и другие высказывания о классах. Например, мы можем определить новый класс Генерирующее оборудование как объединение классов Генератор и Турбина. Для этого используется предикат `owl:unionOf`. Чтобы создать такое утверждение, в редакторе Protégé создадим новый класс, и в области Description для него зададим следующее выражение (Class expression) в поле Equivalent to (рис. 36).

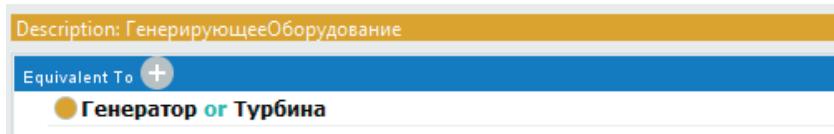


Рис. 36. Утверждение о том, что класс «Генерирующее оборудование» является объединением классов «Генератор» и «Турбина»

Далее необходимо подключить машину логического вывода, для этого нужно в меню Reasoner выбрать какую-либо из доступных машин, например HermiT, а затем в том же меню выбрать пункт Start reasoner. После этого перейдем на вкладку DL Query, чтобы выполнить запрос на получение списка индивидуальных объектов класса Генерирующее оборудование (рис. 37).

The screenshot shows the DL Query tab with the following details:

- Query (class expression):** ГенерирующееОборудование
- Results:**
  - Instances (2 of 2):
    - Генератор1
    - Генератор2
- Justifications:**
  - Show regular justifications
  - All justifications
  - Show iconic justifications
  - Limit justifications to
- Filters:**
  - Query for:
    - Direct superclasses
    - Superclasses
    - Equivalent classes
    - Subclasses
    - Instances
  - ilt filters
  - e contains

Рис. 37. Список объектов класса Генерирующее оборудование

Нажав на знак вопроса напротив утверждения о том, что объект Генератор1 входит в класс Генерирующее оборудование, увидим

объяснение – цепочку логических утверждений, благодаря которым машина логического вывода установила этот факт.

Можно также определить класс как дополнение (`owl:ComplementOf`) другого класса или набора классов. Например, определим, что класс «Кабельно-воздушная ЛЭП» является дополнением к «Воздушной ЛЭП» и «Кабельной ЛЭП», и подклассом «ЛЭП» (рис. 38). На практике это означает, что любая ЛЭП, не являющаяся воздушной или кабельной, должна являться кабельно-воздушной, поскольку других видов ЛЭП не существует (рис. 38).

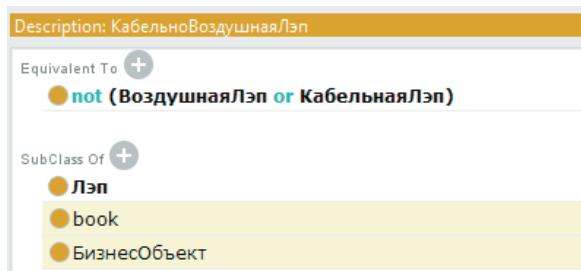


Рис. 38. Определение класса-дополнения

Для проверки заданного утверждения создадим индивидуальный объект «ЛЭП № 5» в классе «ЛЭП» и обновим данные reasoner'a (меню Reasoner → Synchronize reasoner). Можно было бы ожидать, что машина логического вывода заключит, что ЛЭП № 5 является кабельно-воздушной, однако это не так. Причиной является парадигма открытого мира, на которой основана логика работы машин логического вывода. Суть этой парадигмы состоит в том, что отсутствие какого-то факта в онтологии не означает отрицание этого факта. Так, например, в нашей онтологии не сказано, что ЛЭП № 5 является воздушной или кабельной ЛЭП, но это и не исключено. Таким образом, машина логического вывода не обладает достаточными предпосылками для включения ЛЭП № 5 в класс кабельно-воздушных ЛЭП. Чтобы это произошло, укажем в модели явно, что ЛЭП № 5 не является ни кабельной, ни воздушной, и получим нужный вывод (рис. 39).

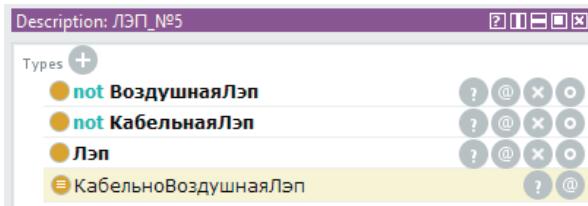


Рис. 39. Получение вывода о вхождении в класс-дополнение

Может показаться, что парадигма открытого мира усложняет практическое использование онтологических моделей, однако по сути она гарантирует отсутствие неверных выводов, основанных на недостаточных предпосылках. Это свойство онтологических моделей крайне важно для систем поддержки принятия решений, в которых должно быть исключено получение умозаключений на основании неполной информации. Если же к создаваемой системе не предъявляется таких требований, можно автоматизировать включение в модель утверждений о не вхождении объектов в классы, к которым они не отнесены явно, или использовать машину логического вывода, способную не учитывать парадигму открытого мира.

Частным случаем, когда дополнение можно использовать без утверждений о невхождении в классы, является использование классов, определяемых перечислением объектов (*owl:oneOf*). Обратимся к примеру, не связанному с энергетикой, поскольку в рамках выбранной предметной области построить класс-перечисление объектов довольно сложно.

Определим класс «Постоянный член Совбеза ООН» как перечисление из пяти объектов: Россия, США, Великобритания, Франция, Китай (рис. 40). Затем определим класс «Непостоянный член Совбеза ООН» как подкласс класса «Член Совбеза ООН» и дополнение к классу «Постоянный член Совбеза ООН» (рис. 41).

Если создать в онтологии объект, представляющий любую страну, и отнести его к классу «Член Совбеза ООН», то он автоматически будет отнесен к классу «Непостоянный член Совбеза ООН». Это произойдет потому, что класс «Постоянный

The screenshot shows a class definition in a semantic web editor. The class is named 'ПостоянныйЧленСовбезаООН' (Permanent Member of the UN Security Council). It has two annotations:

- Equivalent To**: A list of countries: Россия, США, Великобритания, Франция, Китай.
- SubClass Of**: A list containing the class 'ЧленСовбезаООН' (Member of the UN Security Council).

Рис. 40. Класс, определенный как перечисление объектов

член Совбеза ООН» закрыт: из-за того, что он определен явным перечислением входящих в него объектов, ни один другой объект входить в этот класс не может.

The screenshot shows a class definition in a semantic web editor. The class is named 'НеПостоянныйЧленСовбезаООН' (Non-Permanent Member of the UN Security Council). It has three annotations:

- SubClass Of**: A list containing the class 'ПостоянныйЧленСовбезаООН' (Permanent Member of the UN Security Council) preceded by 'not'.
- SubClass Of (Anonymous Ancestor)**: A list containing the class 'ЧленСовбезаООН' (Member of the UN Security Council) preceded by 'or'.
- Instances**: A list of countries: Египет (Egypt), Испания (Spain), Япония (Japan).

Рис. 41. Дополнение к классу-перечислению объектов

Чаще на практике можно встретить определение классов-пересечений (*owl:intersectionOf*). Пересечение означает, что членами данного класса являются все объекты, входящие одновременно во все указанные классы. Факт существования кабельно-воздушных ЛЭП можно отразить другим способом: включать представляющие их объекты одновременно в класс «Воздушная ЛЭП» и «Кабельная ЛЭП», а класс «Кабельно-воздушная ЛЭП» определить как их пересечение (рис. 42). Результат работы логического вывода о том, что индивидуальный объект-член обоих классов, является и членом класса-пересечения, показан на рис. 43.

Можно также определять правила включения объектов в классы на основании того, что они обладают определенными

## Приложение

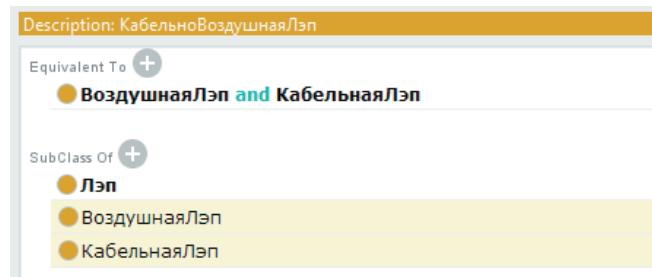


Рис. 42. Определение класса как пересечения

The screenshot shows the 'Description' tab for a rule named 'ЛЭП\_№5'. It defines the type 'КабельноВоздушнаяЛэп'. The 'Explanation' tab provides details about the rule's application, showing examples like 'ЛЭП\_№5 Type ВоздушнаяЛэп' and 'ЛЭП\_№5 Type КабельнаяЛэп'. It also shows the logical derivation path: 'КабельноВоздушнаяЛэп EquivalentTo ВоздушнаяЛэп and КабельнаяЛэп'.

Рис. 43. Работа правила логического вывода и его объяснение

свойствами (рис. 44). Например, будем относить к «Единицам оборудования» все, что установлено на «Техническом месте». Для отражения факта установки на «Техническом месте» в нашей онтологии используется предикат «Установлено на ТМ». Дадим классу «Единица оборудования» определение, показанное на рис. 44.

Созданное утверждение читается так: «Единица оборудования – это все, что установлено на каком-либо Техническом месте». Слово «каком-либо» в этом выражении соответствует квантору существования, который в OWL обозначается как Some.

Для проверки создадим объект «Генератор № 8», поместим его в класс «Бизнес-объект» и укажем, что свойство

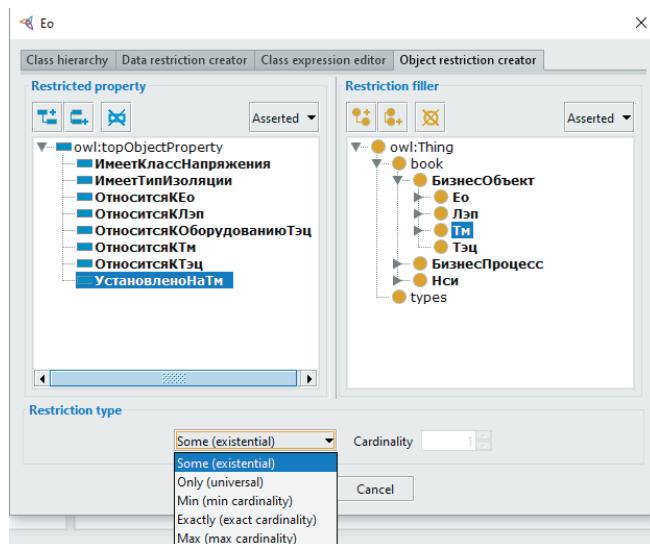


Рис. 44. Определение принадлежности классу через квантор существования

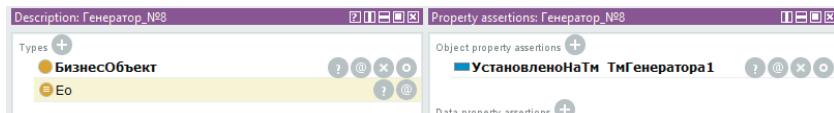


Рис. 45. Проверка вхождения в класс благодаря значению свойства

«Установлен на ТМ» этого объекта обладает значением «ТМ генератора 1» (рис. 45).

На уровне OWL при вводе такого выражения создается так называемое ограничение (**Restriction**) `owl:someValuesFrom` на свойство УстановленоНаТм, значением которого является класс ТмГенератора.

Машине логического вывода выдает заключение, что объект относится к классу «Единица оборудования» (Ео). Если посмотреть на диалоговое окно с объяснением (рис. 46), то можно увидеть, что такая цепочка рассуждений идет лишь второй в списке возможных вариантов получения данного вывода.

## Приложение

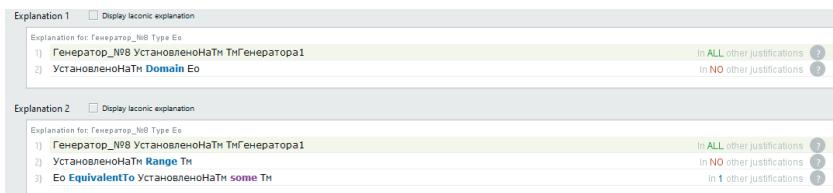


Рис. 46. Объяснение вывода о вхождении объекта в класс

Более простое объяснение состоит в том, что у свойства «Установлено на ТМ» областью применения (domain) является класс «Единица оборудования», этого достаточно, чтобы машина логического вывода относила к данному классу все объекты, обладающие любым значением указанного свойства.

На рис. 44 было показано, что наряду с квантором существования можно использовать и квантор всеобщности (Only). Смысл утверждений с этим квантором состоит в том, что все сущности данного класса связаны определенным свойством только с сущностями другого класса. Так, мы можем определить, что объекты класса Генератор связаны отношением «Установлено на ТМ» только с сущностями класса «ТМ генератора» (рис. 47).



Рис. 47. Утверждение с использованием квантора всеобщности

На уровне OWL при вводе такого выражения создается так называемое ограничение (**Restriction**) owl:allValuesFrom на свойство УстановленоНаТм, значением которого является класс ТмГенератора.

Однако в таком виде утверждение не приведет к получению вывода о том, что «Генератор № 8» является объектом класса Генератор. Причиной является парадигма открытого мира – у свойства «Установлено на ТМ» могут существовать и другие значения, во всяком случае до тех пор, пока иное не указано явно. Самый простой способ сделать это – объявить свойство «Установлено

на ТМ» функциональным (того же эффекта можно было бы добиться указанием для него максимальной кардинальности, равной 1). Функциональные свойства могут иметь только одно значение, то есть объявление свойства функциональным подразумевает, что его максимальная кардинальность равна 1. Сделав такое утверждение, получим требуемый вывод (рис. 48).

The screenshot shows a semantic modeling interface with the following details:

- Top Bar:** Description: Генератор\_Nº8, Property assertions: Генератор\_Nº8.
- Types Panel:** Shows 'БизнесОбъект' and 'Генератор' under 'Types'.
- Object property assertions:** УстановленоНаТм ТмГенератора1.
- Explanation Panel:**
  - Shows radio buttons for 'Show regular justifications' (selected) and 'Show laconic justifications'.
  - Shows radio buttons for 'All justifications' and 'Limit justifications to'.
  - Shows a dropdown menu with the value '2'.
  - Shows a checkbox for 'Display laconic explanation' which is unchecked.
- Result Area:**
  - Explanation for Генератор\_Nº8 Type Генератор:
    - Генератор\_Nº8 УстановленоНаТм ТмГенератора1
    - Functional:** УстановленоНаТм
    - ТмГенератора1 Type ТмГенератора
    - Генератор **EquivalentTo** УстановленоНаТм **only** ТмГенератора

Рис. 48. Вывод, полученный на основании выражения с квантором всеобщности

Если сформулировать более точно, то смысл объявления свойства функциональным состоит в утверждении о том, что если два значения данного свойства указывают на некие объекты, то эти объекты эквивалентны (вспомним, что один объект реального мира может иметь несколько идентификаторов в модели).

Есть возможность получать вывод о вхождении объекта в класс на основании обладания определенным значением свойства-литерала или свойства-ссылки – для этого используется ограничение `hasValue`. С помощью подобных утверждений можно, например, отнести к оборудованию магистральных сетей все оборудование классом напряжения 110 кВ и выше.

Наряду с утверждениями о классах можно делать утверждения и о свойствах. С некоторыми из них мы уже сталкивались при рассмотрении области применения и диапазона значений,

кардинальности, функциональных свойств. Коротко рассмотрим, какие еще утверждения о свойствах можно использовать в онтологических моделях.

Симметричность свойства (Symmetric property) означает, что если объект А связан этим свойством с объектом В, то и объект В связан тем же свойством с объектом А. Пример симметричного свойства – «Является супругом».

Транзитивность свойства (Transitive property) означает, что если объект А связан таким свойством с В, а В с С, то А связан тем же свойством с С. Классический пример транзитивного свойства – «Является частью». Если создать такое свойство в нашей онтологии, указать, что ТмГенератора1 является частью ТмТурбины1, а ТмТурбины1 является частью Тэс1, то будет получен вывод о том, что ТмГенератора1 является частью Тэс1 (рис. 49).

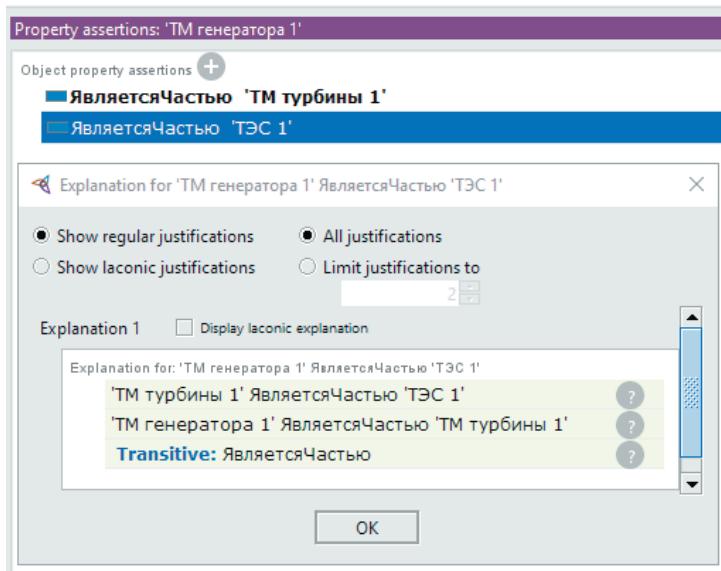


Рис. 49. Вывод, полученный на основе транзитивности свойства

Идея транзитивных свойств получила развитие в цепочках свойств (Property chains). Если три свойства P1, P2 и P3 образуют

цепочку, то если объект А связан с В связью Р1, В связан с С связью Р2, то А связан с С связью Р3. Например, если А является братом В, и В является отцом С, то А является дядей С.

Ассиметричность свойства (Assymetric property) исключает возможность того, что А и В одновременно связаны одним и тем же свойством в обоих направлениях. Так, если А является родителем В, то В не может одновременно являться родителем А (хотя и А, и В относятся к классу «Человек», а у свойства «Является родителем» и область применения, и диапазон значений указывают на этот класс).

Могут существовать пары инверсных свойств (Inverse property) такие, что если А связан с В свойством Р1, то В связан с А свойством Р2. Пример такого свойства можно создать в нашем фрагменте онтологии предметной области: ИмеетУстановленнуюЕО будет являться инверсным свойством для УстановленНаТм (рис. 50).

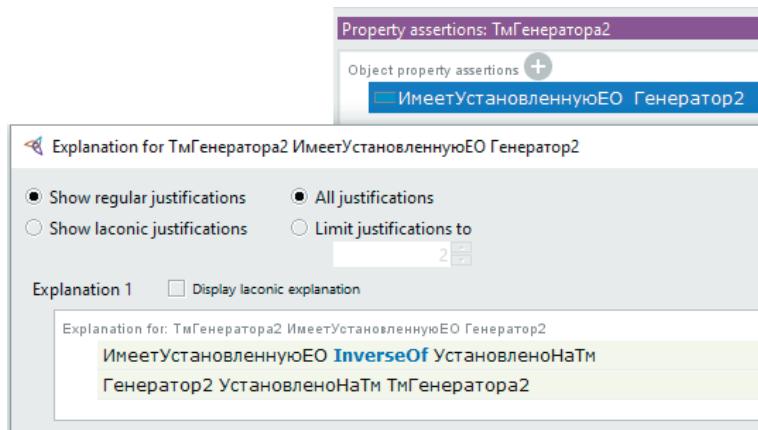


Рис. 50. Пример вывода, полученного на основании инверсного свойства

Существуют также инверсные функциональные (Inverse functional) свойства, суть которых состоит в том, что каждый объект может являться значением такого свойства только для одного объекта, например у человека может быть только одна мать. Рефлексивные (Reflexive) свойства имеют значением тот же объект,

например, любой физический объект обязательно совпадает по размеру с самим собой, поэтому «Совпадает по размеру» – рефлексивное свойство. Не рефлексивные (*Irreflexive*) свойства, в свою очередь, отрицают такую возможность (например, человек не может быть своим собственным родителем).

Наконец, существуют пары разъединенных (*Disjoint*) свойства, суть которых состоит в том, что если объект А связан с объектом В свойством Р1, то он не может быть связан с тем же объектом свойством Р1. Например, свойства «Является матерью» и «Является отцом» разъединенные, так как один человек не может одновременно являться отцом и матерью для другого.

Все богатство перечисленных возможностей предназначено прежде всего для организации логического контроля, проверки корректности вводимых знаний. Если машина логического вывода обнаруживает факты, противоречащие имеющимся в онтологии утверждениям, она выдаст сообщение об ошибке, сопровождающееся пояснением. Однако и здесь необходимо учитывать парадигму открытого мира.

В описанном выше примере мы определили свойство *УстановленоНаТм* как функциональное. Если ввести в онтологию два факта о том, что Генератор1 УстановленоНаТм ТмГенератора1 и Генератор1 УстановленоНаТм ТмГенератора2, можно ожидать, что машина логического вывода сообщит о противоречии – ведь функциональное свойство УстановленоНаТм может иметь только одно значение. Однако на практике этого не произойдет: ризонер придет к выводу, что ТмГенератора1 и ТмГенератора2 означают один и тот же объект, поскольку нигде не было сказано иного. И только если определить, что ТмГенератора1 и ТмГенератора2 являются разными индивидуальными объектами, будет отображено сообщение о неконсистентности онтологии (рис. 51).

Утверждение о различности нескольких индивидуальных объектов можно задать несколькими способами. Выше показан способ сделать это при помощи выражения *DifferentFrom*, применимого для пар объектов. Другой способ достичь того же

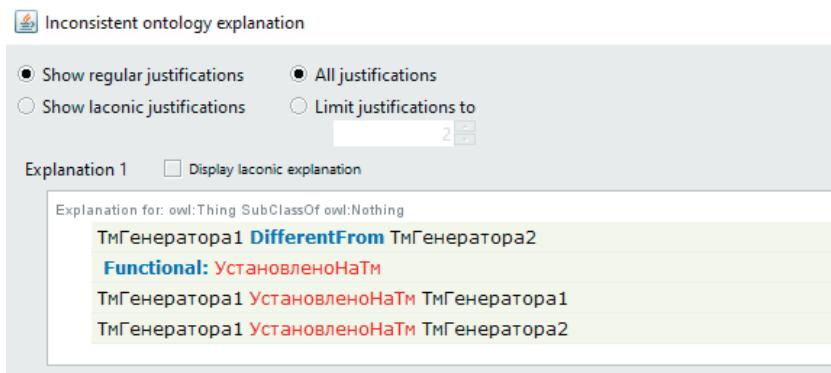


Рис. 51. Сообщение о неконсистентности онтологии в результате нарушения условий на значения функционального свойства

результата – использовать выражение **AllDifferent**, которое утверждает, что все объекты некоторой группы являются различными друг от друга. Оба этих способа неудобны тем, что требуют указания идентификаторов конкретных индивидуальных объектов. Если состав объектов постоянно меняется по ходу существования онтологии, обновление таких выражений затруднительно и ресурсоемко. Третьим и наиболее общим способом утверждения различности объектов является использование ключевых свойств. Ключевое свойство для класса задается при помощи выражения **hasKey**, и в интерфейсе Protégé отображается следующим образом (рис. 52).

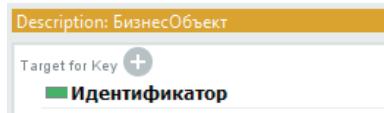


Рис. 52. Ключевое свойство «Идентификатор» для класса «БизнесОбъект»

Такое утверждение означает, что индивидуальные объекты класса **БизнесОбъект**, имеющие отличные значения свойства **Идентификатор**, означают различные объекты реального мира. Верно и обратное: если два объекта имеют одно и то же значение ключевого поля, значит, они означают один и тот же объект (что

эквивалентно использованию выражения `sameAs`). Свойство Идентификатор лучше объявить функциональным, если не требуется возможности присваивать два идентификатора одному объекту.

Правила проверки консистентности играют важную роль при создании, отладке и рефакторинге концептуальной части онтологии (TBox, Terminology box – определения классов и свойств). В промышленных применениях использовать традиционные машины логического вывода не всегда удобно, в том числе из-за их низкого быстродействия на онтологиях, содержащих большое число утверждений об индивидуальных объектах (ABox, Assertions box). Кроме того, в таких применениях часто требуется применение правил, логика которых выходит за рамки, определенные стандартом. Такую задачу решают правила SWRL и SPIN, но и их мощности не всегда достаточно. Решением может служить создание специальных машин логического вывода, одна из которых реализована в продукте компании «Тринидата» – системе управления знаниями «АрхиГраф.СУЗ».

### **Работа с правилами логического вывода в среде АрхиГраф.СУЗ**

Вернемся к онтологии, созданной в редакторе Onto.pro и размещенной в хранилище триплетов (RDF triple store). Программный продукт АрхиГраф.СУЗ позволяет исследовать такие модели при помощи поисковых запросов, выявлять связи между узлами графа знаний, а также создавать и применять правила логического вывода.

Для первого знакомства с редактором создадим простое правило: если единица оборудования НЕ установлена на каком-либо техническом месте, то есть НЕ имеет заданного значения свойства УстановленоНаМ, то ей необходимо присвоить признак «Находится в резерве». Обратим внимание, что при использовании правил SWRL нельзя использовать отрицательные утверждения, а машина применения правил, реализованная в АрхиГраф.СУЗ, позволяет это делать.

В интерфейсе редактора правил необходимо создать новое правило и задать его свойства (рис. 53).

## Программный инструментарий онтологического моделирования. Практикум

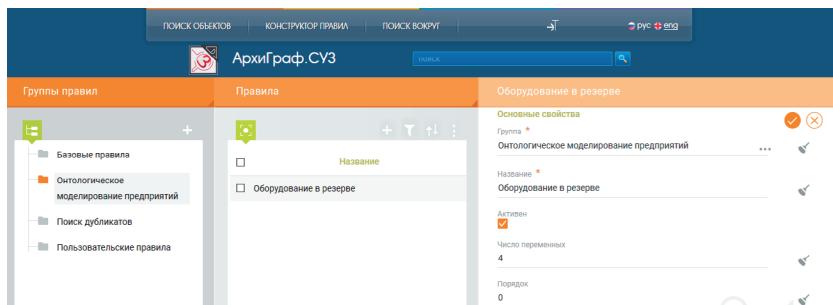


Рис. 53. Свойства правила логического вывода в интерфейсе АрхиГраф.СУЗ

После этого становится доступна область экрана, в которой нужно указать предпосылки (условия) и вывод правила (рис. 54).

The screenshot shows the rule construction interface. The top section is titled 'Оборудование в резерве' (Equipment in Reserve). It has three tabs: 'Объект' (Object), 'Отношение' (Relationship), and 'Объект или значение' (Object or Value). The 'Объект' tab is active, showing a condition: 'объект' (object) is not 'не Установлено на техническом' (not Installed on technical). The 'Отношение' tab shows a conclusion: 'имеет Находится в резерве' (has Is in reserve) is checked. The 'Объект или значение' tab shows an object 'A' selected. Below these tabs are buttons: 'Разъединить условия' (Separate conditions), 'Объединить условия' (Combine conditions), and 'Добавить условие' (Add condition). The bottom section is titled 'To' and shows a condition: 'переменная' (variable) 'A' 'имеет Находится в резерве' (has Is in reserve). It includes checkboxes for 'удалять существующие' (Delete existing) and 'Находится в резерве' (Is in reserve). The bottom of the interface has buttons: 'Добавить вывод' (Add conclusion), 'СОХРАНИТЬ' (SAVE), and 'ОТМЕНИТЬ' (CANCEL).

Рис. 54. Конструирование условий правила логического вывода в интерфейсе АрхиГраф.СУЗ

Правило на рис. 54 состоит из одного условия и одного вывода. Каждое условие и вывод, как и триплет, состоят из трех частей: субъекта, предиката и объекта. В роли субъекта (левая колонка

конструктора) может выступать конкретный объект онтологии, выбираемый по названию или идентификатору, переменная или набор переменных (использование переменных аналогично использованию выражений вида `?var` в SPARQL-запросах или правилах SWRL). Если выбрать одну из переменных, редактор отобразит поле ввода, в котором можно выбрать класс, к которому должны относиться объекты, обозначаемые этой переменной. Выбор класса не обязательен, то есть можно использовать переменные и не предопределяя класс. В показанном выше примере указано, что объекты, идентификаторы которых могут обозначаться переменной A в «логическом уравнении», должны относиться к классу «Единица оборудования».

В средней колонке конструктора выбирается предикат – в выпадающем меню отображаются только те предикаты, которые применимы к классу, выбранному на предыдущем шаге. Кроме предиката, в этом меню можно выбрать и специальные виды условий (рис. 55).

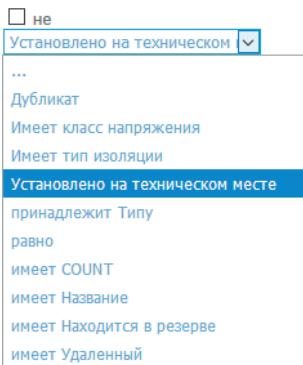


Рис. 55. Выбор предиката или специального условия

Если слева выбрать «набор переменных», в средней колонке появится возможность вычислить их разность, сумму, интервал (для переменных типа `xsd:date` или `xsd:dateTime`) и др.

Содержимое правой колонки варьируется в зависимости от варианта условия, выбранного в средней. Если выбрано

свойство-литерал, то в правой колонке отображаются варианты условий, соответствующие типу значения: больше/меньше/равно для численных типов, содержит/равно для строковых. Можно ввести новую переменную для хранения значения свойства, выбранного в средней колонке, а можно указать значение [пусто]. В нашем примере требуется именно такой вариант, поскольку нам нужно проверить отсутствие значения у свойства «Установлено на техническом месте».

Вывод правила задает структуру утверждения, которое будет сформировано в онтологии в случае, если условия правила выполняются для какой-либо комбинации объектов. В выводе нужно использовать переменные, введенные в условии. В нашем случае используется переменная A, свойству «Находится в резерве» которой присваивается значение `true`.

Для отладки правил предназначен инструмент их ручного выполнения. Инструмент позволяет увидеть утверждения, выводимые на каждом шаге применения правил. Правила применяются последовательно в соответствии с порядком, установленным в их свойствах. Применение правил происходит до тех пор, пока в ходе какой-либо итерации не окажется, что новых утверждений не сформировано. Такой порядок применения правил необходим, потому что утверждения, построенные в результате выполнения одного правила, могут являться предпосылками другого (рис. 56).

Число активных правил = 1					
Применить правила					
Номер цикла	Правило	Добавлено триплетов	Удалено триплетов	Время расчета,с	
	[Завершение работы]	-	-	0	
2	Оборудование в резерве	0	0	0	
1	Оборудование в резерве	27	0	1	
	[Подготовка к работе]	-	-	0	

Рис. 56. Процесс и результат применения правил

Утверждения, полученные в результате применения правил, помещаются в специальный именованный граф. Это необходимо, чтобы отличать их от «постулированных» утверждений, которые введены пользователем и считаются истинными. При просмотре информации об объекте в АрхиГраф.СУЗ выведенные утверждения выделяются цветом, а при наведении на них курсора мыши появляется подсказка со сведениями о том, какое правило логического вывода привело к формированию данного утверждения (рис. 57).

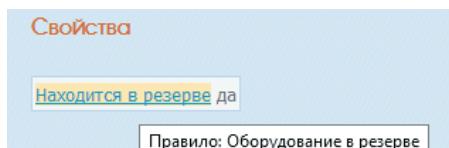


Рис. 57. Выведенный факт в интерфейсе АрхиГраф.СУЗ

Сформулируем более сложное правило. Факты выполнения ремонта отражаются в модели в виде информационных объектов, относящихся к подклассам класса «Ремонт». Они связаны свойством «Относится к оборудованию ТЭС» с единицами оборудования. Предположим, необходимо связать технические места, на которых установлены эти единицы оборудования (временным измерением онтологии пренебрежем для упрощения примера), свойством «Выполненный ремонт» с информационными объектами, представляющими ремонты.

В данном случае участниками правила будут три информационных объекта и соответствующие им переменные:

- А – ремонт;
- В – отремонтированная единица оборудования;
- С – техническое место, на котором установлена единица оборудования.

Условия правила будут выглядеть следующим образом (рис. 58).

Вывод правила состоит в установлении связей между объектами С и А (рис. 59).

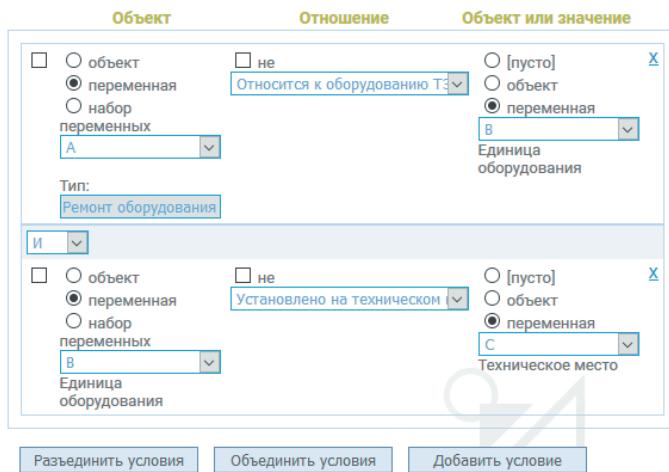


Рис. 58. Комбинация двух логических условий

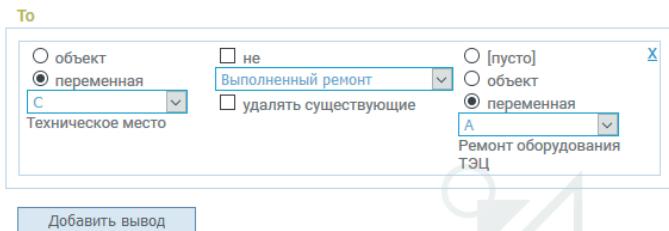


Рис. 59. Вывод правила

Правило может иметь несколько выводов, для добавления дополнительного вывода предназначена кнопка «Добавить вывод».

Конструктор правил позволяет также объединять несколько условий при помощи логических операторов. Предположим, необходимо отобрать оборудование, имеющее масляную или мало-масляную изоляцию и относящееся к классам напряжения 110 или 35 кВ. Условие будет выглядеть так (рис. 60).

Подводя итог обзору возможностей конструктора правил логического вывода в системе АрхиГраф.СУЗ, отметим, что он не требует от пользователя знакомства с синтаксисом логических правил,

## Приложение

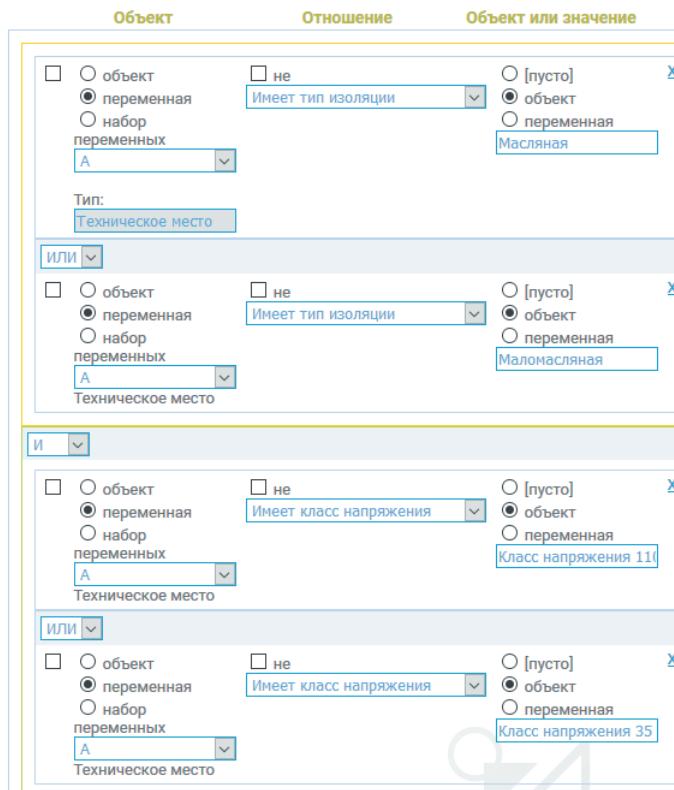


Рис. 60. Правило с двумя группами логических условий

позволяя создавать их через графический пользовательский интерфейс. Функциональность создаваемых правил превосходит возможности SWRL, в частности благодаря возможности использования отрицания при конструировании утверждений.

### Список библиографических ссылок

- Гаврилова Т.А., Кудрявцев Д.В., Муромцев Д.И. Инженерия знаний. Модели и методы : учебник. СПб., 2016. С. 300–301.
- Соловьев В.Д., Добров Б.В., Иванов В.В., Лукашевич Н.В. Онтологии и тезаурусы : учеб. пособие. Казань – Москва, 2006. С. 84–89.

---

3. Mizoguchi R., Sunagawa E., Kozaki K., Kitamura Y. A Model of Roles within an Ontology Development Tool: Hozo // Applied Ontology. 2007. Vol. 2, No. 2. Sep. P. 159–179.

4. Kozaki K., Sunagawa E., Kitamura Y., Mizoguchi R. Role Representation Model Using OWL and SWRL // Proc. of 2<sup>nd</sup> Workshop on Roles and Relationships in Object Oriented Programming, Multiagent Systems, and Ontologies, Berlin. 2007. July 30–31. P. 39–46.

---

---

## Информация об авторах

**Горшков Сергей Вадимович**, директор ООО «ТриниДата»,  
г. Екатеринбург.

*E-mail: serge@trinidata.ru*

**Гребешков Александр Юрьевич**, кандидат технических наук,  
доцент Поволжского государственного университета телекоммуникаций и информатики, г. Самара. Аналитик ООО «ТриниДата».

*E-mail: grebeshkov@trinidata.ru*

**Гумеров Сергей Зуфарович**, директор по развитию и проектному управлению ООО «Институт рациональных технологий»,  
г. Санкт-Петербург.

*E-mail: s.gumerov@inst-rt.ru*

**Кралин Станислав Сергеевич**, технический евангелист, компания «NitrosData», г. Москва.

*E-mail: stanislav.kralin@gmail.com*

**Мирошниченко Максим Григорьевич**, ведущий аналитик, компания Axbit, г. Самара.

*E-mail: maxstroy21@gmail.com*

**Муштак Оксана Игоревна**, разработчик диалоговых систем, компания «JetStyle», г. Екатеринбург.

*E-mail: mushtak.oksana@yandex.ru*

**Шебалов Роман Юрьевич**, кандидат филологических наук, аналитик ООО «ТриниДата», г. Екатеринбург.

*E-mail: shebalov@trinidata.ru*

*Научное издание*

# **ОНТОЛОГИЧЕСКОЕ МОДЕЛИРОВАНИЕ ПРЕДПРИЯТИЙ: МЕТОДЫ И ТЕХНОЛОГИИ**

Монография

Ответственный редактор *C. B. Горшков*

Корректор *E. E. Крамаревская*

Компьютерная верстка *B. B. Таскаев*

Ответственный за выпуск *H. A. Юдина*

Подписано в печать 22.03.2019. Формат 60×84 1/16.

Бумага офсетная. Гарнитура Times New Roman.

Усл. печ. л. 13,72. Уч.-изд. 11,51 л.

Тираж 200 экз. Заказ № 116.

Издательство Уральского университета  
620000, Екатеринбург-83, ул. Тургенева, 4

Отпечатано в Издательско-полиграфическом центре УрФУ  
620000, Екатеринбург-83, ул. Тургенева, 4  
Тел.: +7 (343) 358-93-06, 358-93-22  
Факс: +7 (343) 358-93-06  
E-mail: press-urfu@mail.ru  
<http://print.urfu.ru>

