

ООО «ТриниДата»

А. Беглер, Р. Шебалов

РУКОВОДСТВО ПО СОЗДАНИЮ И ПОДДЕРЖКЕ СЕМАНТИЧЕСКОЙ МОДЕЛИ
с помощью платформы виртуализации данных АрхиГраф.MDM

Екатеринбург, 2022

Содержание

О чём это руководство	3
Общие вопросы	4
Что такое семантическая модель?	4
Зачем нужна семантическая модель?	6
Как создать семантическую модель?	7
Как использовать сторонние онтологии?	10
Принципы и приемы построения семантической модели	15
Выделение набора концептов	15
Именованье и определение сущностей	16
Группировка концептов и определение связей между ними	17
Создание свойств	20
Работа с индивидуальными объектами классов	23
Моделирование отдельных видов информации	27
Создание классов характеристик объектов	27
Моделирование темпоральных характеристик и отношений	28
Моделирование состояний	30
Моделирование адресов	31
Моделирование контрагентов	33
Моделирование активов	35
Инструменты для работы с информацией в семантической модели	37
Управление терминологическим аппаратом	37
Контроль логической целостности	39
Поиск объектов-дубликатов	39
Правила вывода новой информации	40
Организация работы с семантической моделью	42
Команда и заинтересованные лица	42
Оценка онтологии	44
Управление изменениями	45
Документирование	47
Заключение	49

О чём это руководство

Несмотря на бесчисленное количество отечественных и зарубежных публикаций, посвященных вопросам онтологического моделирования и графов знаний, проблемы методики построения семантических моделей на практике остаются актуальными. Не в последнюю очередь это связано с почти полным отсутствием работ, являющихся доступным и достаточно подробным руководством по созданию семантических моделей. Наше руководство призвано хотя бы отчасти восполнить этот пробел. Безусловно, в нём нет исчерпывающего описания всех возможных техник работы с онтологиями, правил их создания и способов применения. Это попросту невозможно, да, собственно, и не нужно в таком виде. Нашей главной целью является описание набора практических действий и приемов моделирования, которые бы позволили получить конкретный результат, а именно семантическую модель, пригодную для решения бизнес-задач в контексте определенного бизнес-ландшафта. Расширение арсенала техник моделирования – вопрос совершенствования бизнес-практик, который каждый специалист и каждая компания решает по-своему.

Это руководство подготовлено коллективом компании «ТриниДата» и адресовано специалистам заказчиков семантических моделей, которые самостоятельно занимаются их внедрением, поддержкой и развитием, используя в первую очередь наше программное обеспечение. Несмотря на то, что мы затрагиваем процесс создания модели, для погружения непосредственно в разработку семантических моделей мы рекомендуем обращаться к специализированной литературе — ряд ключевых работ, на которые мы опираемся, указан в ссылках.

Описанный в руководстве подход и конкретные приёмы моделирования используются нами при создании семантических моделей для наших заказчиков. Где возможно, мы используем для иллюстрации примеры, основанные на тех дизайн-решениях, которые зарекомендовали себя в наших семантических моделях. Примеры представлены в диаграммах, для визуальной нотации использована предназначенная для семантических моделей палитра Grafoo редактора yEd (подробнее про инструменты мы расскажем в заключительном разделе). Перечисленные на диаграммах наборы сущностей являются примерными, они могут быть скорректированы при построении конкретной семантической модели. При использовании сущностей из внешних онтологий добавлен вариант перевода на русский язык.

В первом разделе рассказывается про потенциал использования семантических моделей, их особенности по сравнению с другими видами моделей, а также общую процедуру создания. Следующий раздел посвящён конкретным принципам и приёмам, используемым на этапе разработки модели. Затем, в разделе «Моделирование отдельных видов информации», показано, как описанные приёмы применяются для решения задач, часто встречающихся в практике промышленного моделирования. Во всех этих разделах говорится преимущественно об объектах и явлениях физического мира. Далее, в разделе «Инструменты для работы с информацией в семантической модели», рассматриваются дополнительные слои информации, присутствующие в семантической модели – моделирование словаря, который используется для обозначения реальных объектов и правил, по которым они взаимодействуют. В заключительном разделе кратко описана организация работы с моделью в условиях предприятия.

Общие вопросы

В этом разделе мы дадим определение термину *семантическая модель* и опишем процесс создания такой модели. Затронем фундаментальные принципы, лежащие в основе модели, понимание которых объясняет некоторые особенности формального языка представления моделей и облегчает понимание процесса их создания. Отдельный фрагмент посвящён повторному использованию других моделей – этот шаг может сэкономить достаточное количество усилий при разработке.

Что такое семантическая модель?

Не вдаваясь в подробное перечисление многочисленных определений семантических моделей, оговоримся, что (1) под семантической моделью мы понимаем формализованное описание системы концептов, их отношений и свойств, которые характерны для некоторой предметной области¹ и (2) термины *семантическая модель* и *онтологическая модель* мы употребляем как синонимы. Использование этих терминов указывает на технологию создания модели в соответствии со спецификациями W3C, консорциума Всемирной паутины² (RDF/RDFS/OWL), но не на предметное содержание модели – о разнице семантических моделей в зависимости от их содержания, а также об использовании упомянутых технологий поговорим в разделе «Принципы и приёмы построения семантической модели».

Наиболее часто **онтологическая модель определяется как «формальная явная спецификация общей концептуализации»**³. За каждым словом в этом определении стоит определённый принцип, который нужно учитывать при создании модели:

– *формальная* означает машиночитаемая, то есть созданная с использованием набора технологий, предполагающих автоматическую обработку. Большинство из этих технологий представлено в «пироге семантической паутины» на рисунке 1 и описывается спецификациями консорциума W3C;

– *явная* означает, что элементы семантической модели (классы, свойства, индивидуальные объекты) должны иметь однозначное смысловое и функциональное наполнение. То есть, одно и то же понятие в модели всегда должно обозначать одно и то же явление или объект реального мира. Для достижения такого эффекта семантическая модель имеет встроенные средства аннотирования, целесообразно также использовать средства внешнего описания структуры и содержания модели;

– под *спецификацией* понимается набор объектов и отношений между ними, определяющих рассматриваемую предметную область;

– *общей* модель делает то, что она разделяема специалистами предметной области. Это значит, что модель создается в тесном взаимодействии с профильными экспертами, обладающими необходимыми компетенциями относительно смыслового наполнения и связей бизнес-сущностей и каталогов нормативно-справочной информации (НСИ), которые предполагается реализовать в качестве элементов модели;

– *концептуализация* – это упрощённое видение той части мира, которую мы хотим представить для определённых целей. Это означает, что не существует единственно верной и исчерпывающе правильной семантической модели. Модель – это всегда компромисс,

¹ См. развернутую интерпретацию природы онтологий, например, здесь: Гаврилова, Т. А., Кудрявцев, Д. В., Муромцев, Д. И. (2016). Инженерия знаний. Модели и методы. СПб: Лань. С. 190–193.

² World Wide Web Consortium (W3C) – Консорциум Всемирной паутины, международная организация, созданная для разработки и внедрения Интернет-стандартов. Сайт организации: <https://www.w3.org/>.

³ Gruber, T. R. (1995). Toward principles for the design of ontologies used for knowledge sharing. International Journal of Human-Computer Studies, 43, 907–928.

предполагающий учет текущего объема доступной информации, текущих компетенций разработчиков модели и ее пользователей, текущих и перспективных особенностей ИТ-ландшафта организации, а также бизнес-задачи, для решения которой модель создаётся.

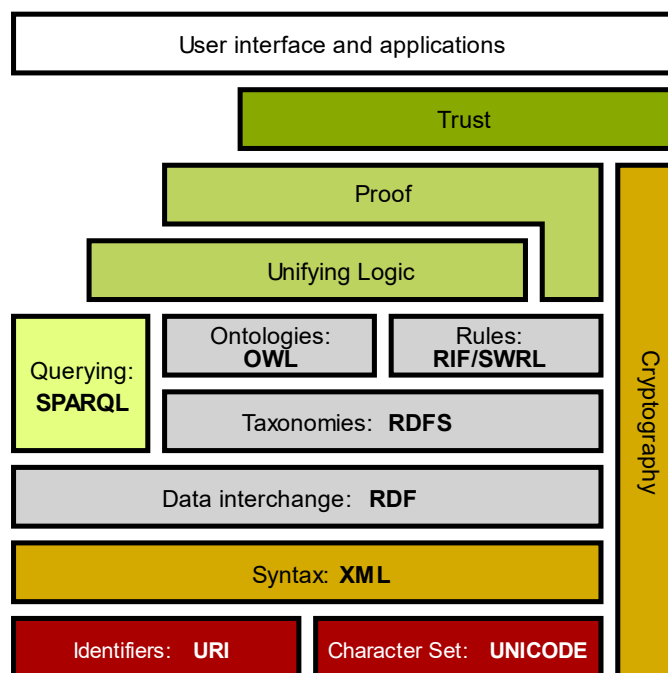


Рис. 1. Стек технологий семантической паутины.
Изображение подготовлено на основе материалов W3C, источник: Wikipedia

От того, насколько стройно, непротиворечиво и целостно будет создана семантическая модель, зависит ее жизнеспособность, а также эффективность тех прикладных компонентов, которые планируется к ней подключить для работы с данными. При этом необходимо учитывать **фундаментальное предположение об открытости мира**⁴ – создавая модель, мы предполагаем, что она заведомо неполна и всегда есть дополнительная информация, которую в неё можно включить. Из этого предположения следует, что:

- семантическая модель не может быть создана сразу целиком. Ее реализация предполагает итеративный подход;
- если в модели что-то не сказано, это не означает, что это не верно – семантическая модель обретает стабильность не сразу, а после достижения соглашения между всеми основными заинтересованными сторонами (экспертами, бизнес-пользователями, специалистами по данным, ИТ-архитекторами и пр.). Обычно такое соглашение требует некоторого времени. Стабильность семантической модели относительна, и в своем зрелом состоянии работа с моделью может регулироваться административными методами;
- семантическая модель должна предполагать возможность расширения и дополнения новыми фрагментами, описывающими смежные области знания (домены) и новые бизнес-процессы. Поэтому в начале ее разработки целесообразно переиспользовать сторонние онтологии.

При этом, в некоторых проектах закрытый мир подразумевается для облегчения работы с моделью и правилами логического вывода. В таких случаях это должно быть явно указано в описании семантической модели.

⁴ Это лишь одно из фундаментальных положений семантической паутины, наиболее релевантное для разработки онтологических моделей внутри компании. Более подробно о его предпосылках и следствиях см. в Allemang, D., Hendler, J. (2011). Semantic Web for the Working Ontologist (330 с.).

Зачем нужна семантическая модель?

Семантическая модель является информационным «каркасом», который обеспечивает структурирование и связность данных, а также предоставляет инструментарий для их описания.

Содержанием модели может быть как описание предметной области (точнее, наших представлений о ней), так и описание лексики (используемого в компании словаря для описания предметной области), правил преобразования или обработки данных, происхождения данных и др. Семантическая модель предметной области реализуется так, чтобы она максимально полно отражала наши представления о реальных объектах, явлениях, процессах, для описания которых она предназначена. Полнота такого отражения традиционно обозначается как «выразительность». Сильной стороной этого подхода является возможность использовать при создании структурных элементов модели понятия, которые являются органичными для самой предметной области и бизнес-практики. Это означает, что семантическая модель не выстраивает искусственных терминологических барьеров между реальной жизнью и способами ее семантического описания. Все, что используется в деятельности предприятия или организации, может быть смоделировано, а значит – может стать объектом автоматизированной обработки.

Важно понимать, что понятия, которые мы используем в семантической модели, указывают на объекты реального мира и выражаются словами языка. Эти **тройственные отношения между предметом, понятием и словом** впервые отметил Готлоб Фреге⁵ и далее развили Чарлз Кей Огден и Айвор Армстронг Ричардс⁶, представив их в формате семиотического⁷ треугольника (см. рисунок 2). *Слово* – это знак, оно обозначает какой-то *предмет* (который представляет собой значение слова) и воплощает *понятие* (смысл слова), которое, в свою очередь, отсылает к предмету. В семантической модели может быть представлен реальный мир (точнее, наш текущий взгляд на реальный мир), на объекты которого указывают понятия, которые воплощены в словах – написанных или произнесенных с использованием набора знаков. Также в модели могут быть представлены отношения между словами, которые используются для обозначения объектов реального мира – тогда мы создаём лексическую модель, или тезаурус. По умолчанию мы будем говорить о моделях первого типа, описывающих реальный мир, и отдельно выделять разделы, в которых речь идёт про лексические модели.

В скобках отметим, что способность формировать понятия, без которой построение семантических моделей было бы невозможным, основана на очень интересной особенности человеческого мышления – типизации эмпирического опыта. Она обеспечивает взаимодействие не с самими объектами реального мира, которые нас окружают, а с обобщенными их репрезентациями в сознании – концептами, которые, в свою очередь, имеют материальное воплощение в виде знаков (слов, символов и пр.). То есть, семантическая модель является инструментом структурирования наборов концептов, нормализации способов их знакового воплощения (преодолением омонимии, полисемии, синонимии) и, таким образом, управления доступом к информации о материальных объектах.

⁵ Frege, G. (1982). Über Sinn und Bedeutung. In: Zeitschrift für Philosophie und philosophische Kritik. Band 100, 25–50; переведено как: Фреге, Г. (1977). Смысл и денотат. В сборнике Семиотика и информатика, выпуск 8, с. 181–210.

⁶ Ogden, C. K., Richards, I. A. (1923). The meaning of meaning: A study of the influence of thought and of the science of symbolism. Harcourt, Brace & World, Inc.

⁷ Семиотика – наука о знаках, в том числе изучает процесс корректной передачи сообщения от отправителя к получателю. Семантические модели в этом контексте обеспечивают решение проблемы потери информации, возникающей при кодировании и декодировании передаваемого сообщения, не касаясь при этом проблемы потери информации, возникающей из-за проблем в передающем канале.

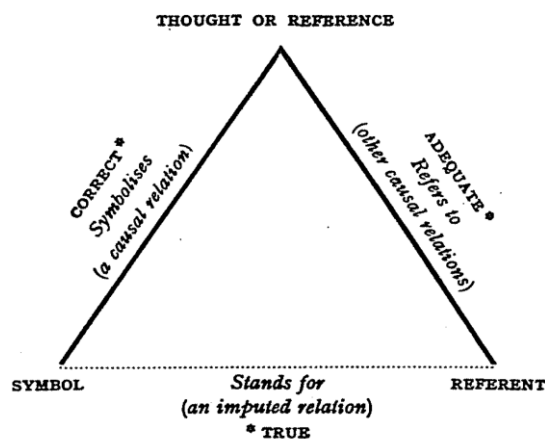


Рис. 2. Семиотический треугольник. Слово (Symbol) обозначает (Stands for) Предмет (Referent) и воплощает (Symbolises) Понятие (Thought of Reference), которое отсылает к (Refers to) Предмету (Referent). Изображение из оригинальной работы Огдена и Ричардса, источник: Wikipedia

Создавая семантическую модель и описывая таким образом наше видение мира, мы приобретаем новое понимание тех явлений, которые описываем. Семантическая модель является, с одной стороны, описательной и этим отличается и от предсказательных моделей, например, регрессионных, и от моделей, по которым строятся объекты реального мира, таких как конструкторские чертежи⁸. С другой стороны, сам процесс создания модели, формализации явлений реального мира, изменяет наше представление о них. Важная особенность семантических моделей: они создаются и для людей, и для машин одновременно. То есть модель призвана и упростить понимание мира для человека, и позволить машине работать с ней.

Модель может создаваться с разными целями. В данном руководстве мы подразумеваем, что модель создается для обработки данных о конкретных объектах и событиях в структуре, описанной этой моделью. Одна из особенностей онтологического моделирования состоит в том, что в нем нет разрыва между концептуальным, логическим и физическим уровнями моделирования данных, привычного для системных аналитиков: наоборот, данные физически хранятся в соответствии именно с той структурой модели, которая задана на концептуальном уровне.

Как создать семантическую модель?

Работу по созданию семантической модели можно разделить на три фазы: (1) подготовка к созданию модели; (2) собственно разработка; (3) поддержка и развитие. Жизнеспособность модели во многом зависит от подготовительной фазы, потому что именно на ней определяется цель создания модели, которая определяет необходимые в модели сущности и предпочтительные приёмы моделирования. Последующие разделы данного руководства посвящены разработке и поддержке модели, а в данном разделе перечислим подготовительные шаги, которые имеет смысл выполнять в любом проекте.

Шаг 1. Определение целей разработки семантической модели и границ предметной области, которая будет представлена в семантической модели.

Цели создания семантической модели могут быть связаны с решением различных бизнес-задач. Их спектр достаточно широк: от упорядочения работы с каталогами НСИ до анализа цепочек

⁸ Guizzardi, G., Proper, H. A. (2021). On Understanding the Value of Domain Modeling. In Proceedings of 15th International Workshop on Value Modelling and Business Ontologies (VMBO 2021). Краткое содержание статьи: <https://www.youtube.com/watch?v=OTVouXTDKGs>.

поставок, от разработки корпоративной поисковой системы и подбора аналогов до выявления аномалий и построения аналитических прогнозов, и многое другое. Главная задача состоит в том, чтобы максимально конкретно определить эти цели и сформировать образ желаемого результата.

Цели будут напрямую определять тот фрагмент предметной области, который должна описывать разрабатываемая семантическая модель. Например, если с помощью семантической модели планируется обеспечить связывание информации о поставщиках продукции, представленной в различных корпоративных системах, то предметная область должна включать, как минимум, сведения о контрагентах, договорах, фактах поставок, складах и складских остатках, логистических цепочках, товарной номенклатуре. Не стоит пытаться сразу описать в семантической модели весь объем деятельности предприятия, по крайней мере на начальном этапе ее разработки. Это может оказаться слишком трудозатратно и вряд ли даст нужный результат. Модель может быть расширена на новые фрагменты предметной области позднее.

В процессе разработки придерживаться содержания модели, соответствующей цели, помогают **требования к онтологии**⁹. Это отдельный документ, в который обычно входит:

- описание цели, содержания и языка реализации семантической модели;
- предполагаемое использование модели и конечные пользователи;
- нефункциональные требования: общие аспекты онтологии, не относящиеся к её контенту, в частности, должна ли разрабатываемая модель включать какие-либо стандарты, должна ли быть многоязычной, какую систему именования она должна использовать;
- функциональные требования, обычно в формате *компетентностных вопросов*: формулируется набор вопросов, на которые должна быть способна ответить онтология, например, «Когда в последний раз осуществлялась поверка датчиков температуры?» или «Кто отвечает за проведение анализа результатов проб определённого типа?» – вопросы должны отражать конкретные задачи, которые должна решать разрабатываемая модель, они предназначены конкретизировать цель.

Шаг 2. Определение круга заинтересованных лиц и подразделений организации, которые имеют отношение к разработке семантической модели и к использованию бизнес-приложений, основанных на семантической модели; специалистов, ответственных за сопровождение разработки семантической модели.

Четко определенные на предыдущем шаге цели и границы предметной области позволят выделить основных стейкхолдеров и профильных экспертов. Заинтересованные лица со стороны компании, бизнес-пользователи разрабатываемой модели привлекаются к разработке для контроля содержания модели, а также для понимания удобства конечных пользователей. Без полноценного привлечения профильных экспертов к созданию модели ее выразительные возможности рискуют оказаться недостаточными, что сделает неэффективными трудозатраты на разработку: несмотря на то, что ключевым аспектом создания модели является её соответствие заявленным целям и вписанность в бизнес-процесс, она всё же при этом должна иметь практическую значимость и отражать текущее положение дел в предметной области, что и обеспечивается привлечением профильных экспертов. Кроме этого, к созданию онтологической модели должны привлекаться инженеры по знаниям – они обеспечивают согласованность модели, правильность её построения и принятие дизайн-решений для моделирования конкретных сущностей.

⁹ Приведены на основе шаблона такого документа, представленного в Suarez-Figueroa, M. C., Gómez-Pérez, A., Motta, E., Gangeni, A. (2012). *Ontology Engineering in a Networked World*. <https://doi.org/10.1007/978-3-642-24794-1>.

Взаимодействие заинтересованных лиц при разработке семантической модели может быть организовано с помощью принятых в компании средств и способов. Они могут включать проведение с установленной регулярностью рабочих встреч и совещаний, создание отдельных рабочих пространств на площадках для групповой работы в формате электронных досок, корпоративного мессенджера и пр.

Шаг 3. Выбор основных инструментов для разработки семантической модели, описания процесса разработки.

В состав такого инструментария входит редактор семантической модели, графовая база данных (точнее, выражаясь техническим языком, хранилище триплетов – RDF triple store), средства анализа качества данных и пр. Существует несколько редакторов онтологий, большая часть которых являются проприетарными¹⁰. Среди свободно распространяемых наибольшей известностью пользуется редактор Protégé в его десктопной¹¹ и веб-версиях¹². Этот редактор поддерживает стандарты W3C, обладает обширной документацией на английском языке и поддерживается достаточно большим сообществом пользователей. Однако сфера использования Protégé – это в основном научные и научно-практические проекты, частные разработки энтузиастов, интересующихся вопросами семантического моделирования.

Очевидно, что в случае с решением корпоративных задач оптимальным вариантом является использование целостной экосистемы, предоставляющей комплексное решение по развертыванию набора программных продуктов, обеспечивающих редактирование семантической модели, организацию хранения данных и их виртуализацию, интеграцию с имеющимися в компании СУБД, реализацию различных инфраструктурных задач. Платформа «АрхиГраф» является именно такой экосистемой¹³, часть которой составляют специализированные приложения, напрямую адресуемые к онтологии, обеспечивающей описание структуры данных¹⁴.

Шаг 4. Подготовка документации, описывающей методику разработки и принципы использования семантической модели.

Хорошей практикой является подготовка методики построения семантической модели – документа, в котором фиксируются принципы и подходы к созданию семантической модели, а также правила ее администрирования и доработки.

В методике моделирования необходимо зафиксировать задачи построения семантической модели, ее структуру, правила присвоения идентификаторов (в виде URI) для различных компонентов модели – классов, свойств, объектов, принятые правила выделения классов и подклассов, модель классов верхнего уровня и их смысловую нагрузку, правила реализации в модели свойств и связей объектов классов. Следует также описать процессы управления изменениями модели и НСИ, а также подходы к организации работы по управлению изменениями модели, принципы оценки жизненного цикла модели и ее рефакторинга. В том случае если с помощью семантической модели планируется работать с временной и пространственной информацией – описать в методике способы хранения темпоральных и геопространственных данных. Рассмотрению принципов и приёмов моделирования, подходов к моделированию различных сущностей и организации работы с моделью посвящены соответствующие разделы руководства.

¹⁰ См. краткий перечень редакторов онтологий, например, здесь: https://www.w3.org/wiki/Ontology_editors.

¹¹ <https://protege.stanford.edu/>

¹² <https://webprotege.stanford.edu/>

¹³ <https://trinidata.ru/platform.htm>

¹⁴ Подробнее о построении дата-центричных корпоративных архитектур см.: https://trinidata.ru/mdm_archigraph.htm и https://trinidata.ru/data_centric_architecture.htm.

Как использовать сторонние онтологии?

Формирование наборов понятий (концептов), составляющих семантическую модель, и определение их взаимосвязей является достаточно непростой задачей, особенно если ее приходится проводить «с чистого листа». Для того чтобы упорядочить концепты и организовать такую их структуру, которая допускала бы расширение семантической модели без необходимости кардинально ее переделывать, целесообразно использовать сторонние онтологии.

Сторонние онтологии – результат работы различных научно-исследовательских коллективов, состоящих из специалистов по философии, теории информации, онтологическому инжинирингу, управлению знаниями, а также экспертов в различных предметных областях. Среди сторонних онтологий для переиспользования наиболее полезны онтологии верхнего уровня и предметные онтологии.

Онтология верхнего уровня (Upper Ontology) позволяет сформировать абстрактный и максимально универсальный информационный каркас, который можно конкретизировать по мере развития семантической модели и который позволит связать в единую структуру практически любые концепты. Онтологии этого типа включают такие классы, как «Субъект», «Объект», «Пространственный объект», «Временной регион», «Качество», «Событие», «Деятельность» и др. Такой каркас выполняет три функции¹⁵: (1) разрабатываемые на его основе онтологии могут дальше повторно использоваться как в разных подразделениях компании, так и её партнёрами, и всем профессиональным сообществом; (2) позволяет избежать ситуации, когда при разработке с опорой на существующие данные онтология отражает только структуру данных, а не объектов реального мира за ними; (3) позволяет легко интегрировать несколько онтологий на её основе или с отображением на неё.

На данный момент разработано более десятка онтологий верхнего уровня, каждая из которых имеет свои особенности применения. Наиболее известными из них являются Basic Formal Ontology (BFO), Descriptive Ontology for Linguistic and Cognitive Engineering (DOLCE), General Formal Ontology (GFO), Suggested Upper Merged Ontology (SUMO) и др¹⁶.

Невозможно однозначно определить, какая из онтологий верхнего уровня является наиболее «правильной» и «полной». Скорее, вопрос стоит ставить иначе: какая из онтологий в наибольшей степени отвечает целям и задачам реализации разрабатываемой семантической модели, и какая является в достаточной степени развитой? Мы в своей практике под развитостью понимаем активность поддержки онтологии со стороны экспертного сообщества, наличие возможностей по нативной интеграции смежных онтологий более низкой степени абстракции с онтологией верхнего уровня, в том числе на уровне идентификаторов ее элементов. С учетом этих критериев в настоящее время достаточную степень зрелости демонстрирует онтология Basic Formal Ontology (BFO)¹⁷, которая чаще всего используется специалистами компании «ТриниДата» при построении информационных моделей промышленного применения.

¹⁵ Smith, B. (2019). Introduction to Basic Formal Ontology. URL: <https://www.youtube.com/watch?v=p0buEjR3t8A>

¹⁶ Подробнее см., например: https://en.wikipedia.org/wiki/Upper_ontology#; Partridge, C., Mitchell, A., Cook, A., Sullivan, J., West, M. (2020). A Survey of Top-Level Ontologies-to inform the ontological choices for a Foundation Data Model. Version 1. Construction Innovation Hub.

¹⁷ <https://basic-formal-ontology.org/>; <https://github.com/BFO-ontology>; <https://groups.google.com/g/bfo-discuss>. См. также монографию Arp, R., Smith, B., Spear, A. D. (2015). Building Ontologies with Basic Formal Ontology. MIT press, в которой изложены концептуальные основания разработки BFO, принципы использования онтологии и лучшие практики онтологического инжиниринга с использованием BFO.

В онтологии BFO все сущности разделены на две группы: представленные полностью в каждый момент времени «Континуанты» («Continuant») и протяжённые во времени «Оккurrentы» («Occurrent»). Континуанты представляют собой объекты, их характеристики и роли, а также пространства. Оккurrentы – это процессы, их границы, а также пространственно-временные сущности. Отношения между сущностями вынесены в отдельную онтологию — Relation Ontology (RO) и содержат отношения между континуантами, например, «является частью» («part of») или «расположен в» («located in») и континуантов с оккurrentами, например, «происходит в» определённое время («at»).

Предметная онтология (Domain Ontology) представляет собой семантическую модель, специфичную для какой-либо области знания (медицины, биологии, бизнеса и финансов, военного дела, геологоразведки, информационной безопасности, лингвистики, культуры и искусства и пр.). Предметные онтологии могут разрабатываться по отдельности¹⁸ или в рамках целостных проектов¹⁹.

Использование устоявшихся предметных онтологий позволяет «не изобретать колесо», т. е. не придумывать наборы классов, их подклассов и связи между ними, что существенно сокращает трудозатраты на разработку семантической модели. Это дает возможность достаточно быстро перейти к созданию той части модели, которая непосредственно описывает предметную область с учетом специфики данной организации. Положительным итогом такой практики является возможность провести бесшовное сопряжение сторонних онтологий и корпоративной семантической модели. Кроме того, использование сторонних онтологий помогает снизить вероятность ошибок и неточностей моделирования, многих из которых трудно избежать при создании действительно сложных моделей.

Немаловажным преимуществом применения сторонних онтологий также является и тот факт, что они позволяют использовать чужой опыт моделирования фрагментов действительности, т. е. фактически расширить диапазон возможностей аналитика как в части типов данных (классов), так и отношений между ними (связей классов). Так, например, Organization Ontology содержит 9 базовых классов и 35 свойств классов²⁰; онтология Friend of a Friend (FOAF) включает 13 базовых классов и 62 их свойства²¹; онтология SOSA/SSN содержит 19 базовых классов и 38 их свойств²².

Сложностью повторного использования сторонних онтологий является их поиск и оценка качества: онтологий верхнего уровня существует всего несколько, они достаточно известны в сообществе, в том числе доступна информация об их использовании в реальных проектах. С предметными онтологиями ситуация обратная: не всегда легко найти подходящую, а после нахождения отдельной задачей становится оценка её качества. Для поверхностного поиска онтологий существуют открытые хранилища (см. ссылку 18), при необходимости осуществить более глубокий поиск можно воспользоваться подходом на основе структурированного поиска литературы²³ – в нём по ключевым словам ищутся статьи, презентующие онтологические модели, затем отбираются релевантные, в которых ищутся уже непосредственно ссылки на онтологическую модель и её описание. Данный подход достаточно трудоёмок, однако применять его каждый раз нет необходимости – в процессе работы формируется набор словарей, которые достаточно часто

¹⁸ <https://lov.linkeddata.es/dataset/lov/vocabs>; <https://www.ebi.ac.uk/ols/ontologies>; <http://aber-owl.net/ontology/#/>; <https://bartoc.org/vocabularies>

¹⁹ <https://obofoundry.org/>

²⁰ <https://www.w3.org/TR/vocab-org/>

²¹ <http://xmlns.com/foaf/spec/>

²² <https://www.w3.org/TR/vocab-ssn/#>

²³ Campos, P. M. C., Reginato, C. C., Almeida, J. P. A., Barcellos, M. P., de Almeida Falbo, R., Silva Souza, V. E., Guizzardi, G. (2020). Finding reusable structured resources for the integration of environmental research data. *Environmental Modelling & Software*, 133, 104813.

используются (например, онтологии времени или единиц измерения) и более углублённый поиск бывает необходим только при вхождении в новую область. Кроме этого, наиболее общеупотребимые модели известны инженерам по знаниям, а в некоторых областях знаний и профильным экспертам.

Приведем небольшой пример. Допустим, необходимо построить семантическую модель, которая бы позволила описать организацию и те бизнес-объекты, с которыми она взаимодействует в определенной производственной сфере. Типологически таких объектов очень много: от ее организационных подразделений и работников до контрагентов и контрольно-надзорных органов, от рабочих мест и единиц оборудования до цепочек поставок сырья и материалов, и т. д. В связи с этим целесообразно использовать онтологию верхнего уровня, которая будет дополняться предметной онтологией (с учетом специализированных задач) и далее информационными сущностями, представляющими элементы бизнес-модели данной организации. Концептуально такая «стыковка» информационных сущностей может быть представлена в следующем виде:

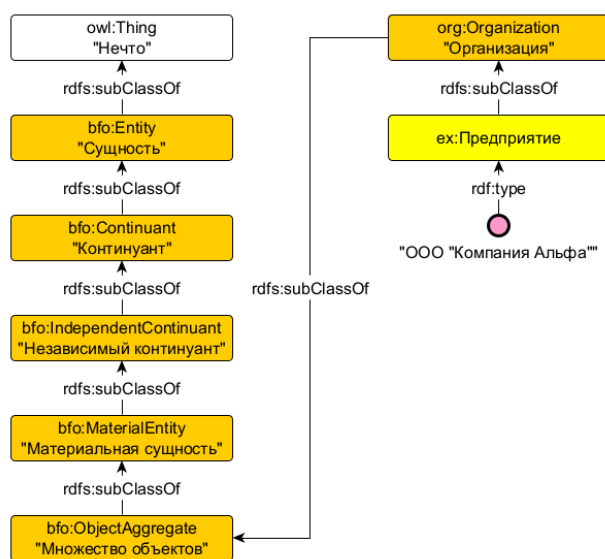


Рис. 3. Сочетание элементов разных онтологий в составе единой семантической модели. В рыжих прямоугольниках классы из сторонних онтологий; приставки *bfo:*, *org:* и *owl:* отсылают к онтологиям, в которых определены импортированные сущности, подробнее про именование в разделе «Именование и определение сущностей»

На диаграмме приведена взаимосвязь классов трех онтологий: BFO, Organization Ontology и онтология условной компании («ex» – от англ. example, «пример»), а также индивидуальный объект класса *ex:Предприятие* – «ООО "Компания Альфа"». Класс *ex:Предприятие* определён в онтологии компании, а далее он также определён как подкласс класса сторонней онтологии *org:Organization*. Класс «Организация» (*org:Organization*) предназначен для сущностей, «представляющих собой собрание людей в сообщество или другую социальную, коммерческую или политическую структуру»²⁴. Класс компании *ex:Предприятие* сужает это определение до коммерческой структуры.

Онтология Organization Ontology – предметная онтология, описывающая организации. Чтобы связать эту онтологию (и уточняющие её классы разрабатываемой онтологии компании) проводится анализ классов выбранной онтологии верхнего уровня BFO. Решение о том, что класс

²⁴ Определение класса Organization из спецификации онтологии The Organization Ontology: «Represents a collection of people organized together into a community or other social, commercial or political structure. The group has some common purpose or reason for existence which goes beyond the set of people belonging to it and can act as an Agent», источник: <https://www.w3.org/TR/vocab-org/#class-organization>.

может быть определен как подкласс (`rdfs:subClassOf`) класса «Множество объектов» (`bfo:ObjectAggregate`), принимается на основе того, что организация, по определению `org:Organization`, представляет собой «собрание людей», то есть, некоторое объединение материальных объектов, которые существуют независимо от других объектов и в ходе существования относительно стабильны – именно так определён класс `bfo:ObjectAggregate` в онтологии BFO²⁵.

Остальная часть иерархии на рисунке 3 представляет собой фрагмент онтологии BFO вплоть до максимально возможного уровня абстракции – сущности «Нечто», экземплярами которой в языке OWL является что угодно. Так, «Множество объектов» в BFO является «Материальной сущностью» (чем-то, имеющим физическое воплощение), которая представляет собой подвид «Независимого континуанта» (сущности, представленной полностью в единый момент времени и не нуждающейся ни в какой другой сущности для своего существования – в отличие, например, от качеств, которым нужны носители, чтобы проявиться). «Независимый континуант» вместе с не представленными на схеме «Специфически зависимым» (теми самыми качествами) и «Общезависимым континуантом» образуют класс «Континуантов», которые являются «Сущностями» онтологии BFO.

Описание в семантической модели разнообразия типов оборудования, которое используется на предприятии, может быть реализовано через создание классов, показанных на следующей диаграмме:

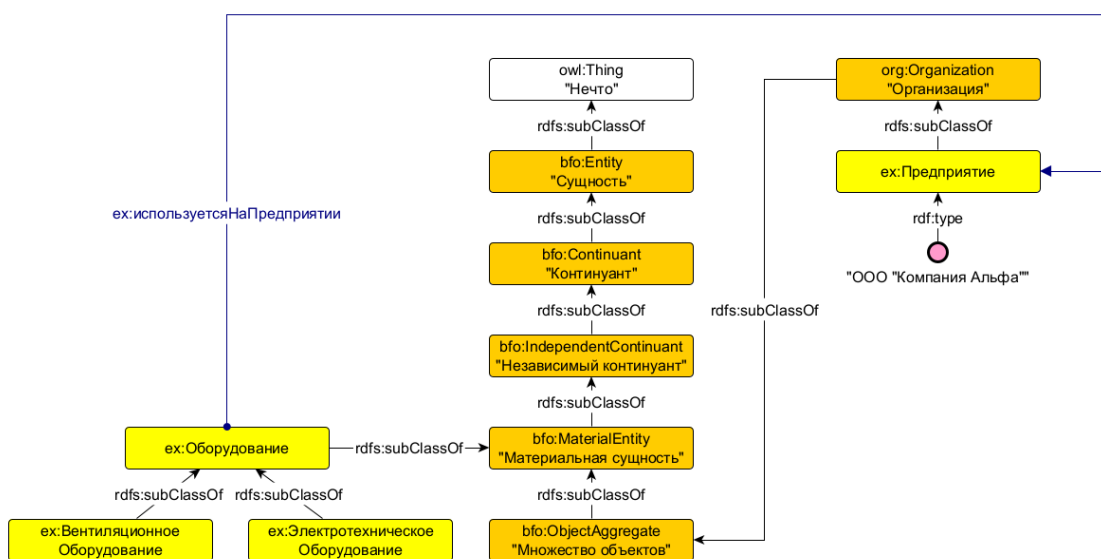


Рис. 4. Типы оборудования в семантической модели

Класс `ex:Оборудование` может быть определен как подкласс `bfo:MaterialEntity`²⁶ в силу семантики, которой обладает каждый из этих классов, и смысловой связи между концептами «Оборудование» и «Материальная сущность», так как любое оборудование материально.

Другие распространенные классы предметной области могут быть определены как подклассы BFO или их эквиваленты следующим образом (примерный перечень, должен быть уточнен в зависимости от семантики классов конкретной онтологической модели):

²⁵ Одно из определений данного класса в онтологии BFO: «An object aggregate is a material entity that is made up of a collection of objects and whose parts are exactly exhausted by the objects that form this collection».

²⁶ Ср. одно из определений данного класса в BFO: «Material entities (continuants) can preserve their identity even while gaining and losing material parts».

Класс предметной области	Класс BFO/IAO ²⁷
ex:ФизическоеЛицо	bfo:Object
ex:ТехническоеМесто	bfo:SpecificallyDependentContinuant
ex:Событие	bfo:ProcessBoundary
ex:Процесс	bfo:Process
ex:Документ	iao:Document
ex:Файл массива данных	iao:DataSet
ex:Изображение	iao:Image
ex:Исполнение роли	bfo:Role

В результате мы получаем возможность описать взаимосвязь различных классов, входящих в целостную информационную модель по принципу иерархии, а также выразить более специфичную семантику отношений – например, установить связь между оборудованием и предприятием, на котором оно используется (ex:используетсяНаПредприятии)²⁸. Подобная организация представления информации не только обладает практически безграничной выразительностью, но и позволяет использовать гибкие способы извлечения необходимой информации в машиночитаемом виде.

Способы использования сторонних онтологий могут быть разными. В языке OWL предусмотрено несколько вариантов для такого использования²⁹: (1) прямой импорт всей сторонней онтологии с помощью `owl:import`; (2) прямое повторное использование отдельных сущностей сторонней онтологии с использованием аннотационного свойства `rdfs:isDefinedBy` – при таком методе сущности определяются в разрабатываемой онтологии, а в указанном свойстве даётся ссылка на источник; (3) не прямое использование через связь необходимых классов сторонней онтологии с помощью `rdfs:subClassOf` – с помощью него класс разрабатываемой онтологии связывается с соответствующим надклассом сторонней онтологии. Эти способы могут комбинироваться в рамках одной семантической модели.

При использовании онтологий верхнего уровня или предметных онтологий в платформе Архиграф необходимо физически импортировать эти онтологии в создаваемую модель собственными средствами платформы (конструкция `owl:import` для ссылки на сторонние онтологии на момент написания этого Руководства не поддерживается). Для этого в редакторе Архиграф.Мир предусмотрены пункты меню «Импорт/экспорт OWL» и «Импорт/экспорт NTriples». В интернете можно найти файлы большинства распространенных онтологий в одном из этих форматов, при необходимости – усечь эти файлы путем удаления ненужных элементов, и импортировать в Архиграф.Мир.

²⁷ Information Artifact Ontology (IAO) – онтология, описывающая информационные объекты. Она является расширением BFO на более специфическую предметную область. См.: <https://github.com/information-artifact-ontology/IAO>; <https://raw.githubusercontent.com/information-artifact-ontology/IAO/v2020-12-09/iao.owl>.

²⁸ В данном примере в целях упрощения мы сознательно не отражаем временной аспект отношений по поводу использования оборудования в деятельности предприятия.

²⁹ Carriero, V. A., Daquino, M., Gangemi, A., Nuzzolese, A. G., Peroni, S., Presutti, V., Tomasi, F. (2020). The Landscape of Ontology Reuse Approaches, (section 1), 1–18. <https://doi.org/10.3233/ssw200033>.

Принципы и приемы построения семантической модели

Описанные выше методологические подходы к проектированию семантической модели, очевидно, должны быть реализованы через конкретные приемы моделирования, т.е. практические шаги и операции по ее построению. Не претендуя на исчерпывающую полноту описания, остановимся на тех из них, которые являются наиболее часто используемыми в практике промышленного моделирования³⁰.

Если рассматривать семантическую модель с точки зрения функциональной природы информации, то в ней выделяются два уровня: TBox (Terminology Box, «хранилище терминов») и ABox (Assertion Box, «хранилище фактов/суждений»). TBox – уровень описания структуры представлений, он включает определения классов и свойств; ABox – уровень фактографических данных, включает индивидуальные объекты модели. Разработку семантической модели целесообразно начинать с TBox. Ниже мы опишем приёмы для моделирования предметной области вместе с необходимым для построения онтологий каркасом формального языка, предоставляемого спецификациями технологического стека семантической паутины: Resource Description Framework (RDF)³¹, RDF Schema (RDFS)³² и OWL 2 Web Ontology Language³³.

Кроме описания предметной области, в семантической модели могут присутствовать области или слои, необходимые для решения специфических задач:

- слой лексических данных (необходим для создания глоссария терминов или решений по обработке текстов на естественном языке);
- слой данных, описывающих алгоритмы работы прикладного программного обеспечения (необходим, в частности, если в модели содержится описание структуры интерфейсов пользовательских приложений или соответствие элементов модели таблицам СУБД систем-источников);
- слой правил логического вывода;
- слой метаданных (информации о происхождении данных, сведения об авторах, дате и времени редактирования сущностей семантической модели и др.).

В данном разделе мы будем иногда касаться этих слоёв, но речь пойдёт преимущественно о моделировании объектов и явлений реального мира и отношений между ними. Слои лексических данных и данных логического вывода будут упомянуты в подразделах раздела «Инструменты для работы с информацией в семантической модели», слой метаданных рассмотрим в заключительном разделе, а слой описания алгоритмов останется за рамками данного руководства.

Выделение набора концептов

После того как определена предметная область, которую необходимо реализовать в семантической модели, нужно сформировать составляющий ее набор понятий (концептов), указывающих на объекты реального мира, которые мы хотим описать в модели. Физически это

³⁰ О приемах и техниках моделирования см., например: Гаврилова, Т. А., Кудрявцев, Д. В., Муромцев, Д. И. (2016). Инженерия знаний. Модели и методы (324 с.). СПб: Лань.; Цуканова, Н. И. (2014). Онтологическая модель представления и организации знаний. Учеб. пособие для вузов. М.: Горячая линия–Телеком.; Муромцев, Д. И. (2007). Онтологический инжиниринг знаний в системе Protégé. СПб: СПб ГУ ИТМО, 62; Rudnicki, R., Smith, B., Malyuta, T., Mandrick, C. W. (2016). Best Practices of Ontology Development October 25, 2016. CUBRC, Inc.; Noy, N. F., McGuinness, D. L. (2001). Ontology development 101: A guide to creating your first ontology. URL: https://protege.stanford.edu/publications/ontology_development/ontology101.pdf.

³¹ <https://www.w3.org/TR/rdf-primer/>

³² <https://www.w3.org/TR/rdf-schema/>

³³ <https://www.w3.org/TR/owl2-primer/>

выражается в том, что аналитик составляет перечень слов³⁴, обозначающих сущностно самостоятельные элементы (концепты), относящиеся к данной предметной области. Например, если требуется описать организационную структуру предприятия, такими концептами могут быть: «Департамент», «Отдел», «Дисциплина», «Сектор», «Лаборатория», «Цех», «Склад», «Должность», «Кадровая единица» и т. д. Необходимо учитывать, что при работе с концептами мы имеем дело не со словами как единицами лексической системы естественного языка – на данном шаге мы разрабатываем именно объектную модель предметной области, а не лексическую, то есть нас интересуют понятия, а не слова, которыми они описаны в языке. Использование слов при этом неизбежно, потому что они репрезентируют понятия. На этом шаге целесообразно выделять концепт по принципу его функциональной уникальности, не вдаваясь в тонкости обозначения понятия разными словами в языке³⁵: это позволит избежать случаев, когда в семантическую модель могут попасть разные обозначения одного и того же понятия (как, например, «Санитарная книжка» и «Санкнижка», «Исполнитель работ» и «Исполняющий работу», «Автоматизированная система управления технологическим процессом» и «АСУ ТП»).

На данном этапе работы с моделью могут использоваться любые инструменты, вплоть до текстовых редакторов – задача этого этапа найти релевантные концепты, не приступая пока к моделированию средствами формального языка. Однако уже на этапе выявления концептов разумно использовать хорошие практики именования, а также сразу давать концептам определения – это сэкономит время в дальнейшем, а также позволит сохранить содержание концептов неизменным на протяжении разработки.

Именованье и определение сущностей

Именованье сущностей в онтологии должно обеспечивать и человеко- и машиночитаемость. Второе обеспечивается тем, что каждая сущность в модели (класс, свойство, индивидуальный объект) обладает своим *уникальным идентификатором* – URI (Uniform Resource Identifier). Ключевое отличие такого идентификатора от URL (Uniform Resource Locator) в том, что он не должен меняться при перемещении ресурса по другому адресу в сети. В общем виде URI любой сущности в онтологии выглядит так: http://www.someurl.ru/ontologies/example_onto.owl#Colour, где «http://www.someurl.ru/ontologies/example_onto.owl» – это пространство имён онтологии, «example_onto.owl» – сама онтология, «Colour» – идентификатор конкретной сущности (в данном случае – класса). Пространство имён онтологии для улучшения человекочитаемости зачастую обозначается префиксом, в приводимых выше примерах это префикс «ex:», то есть, когда мы видим сущность ex:Краска, мы понимаем, что данная сущность взята из онтологии example_onto.owl.

³⁴ Напомним здесь про тройственные отношения между предметом, понятием и словом: мы моделируем предметы (объекты реального мира), которые в модели представлены понятиями (смысловыми элементами, указывающими на соответствующие объекты) и воплощены в словах (наборах знаков). По сути, в модели объектов физического мира мы оперируем понятиями, но так как понятия неизбежно представлены в какой-либо знаковой системе, здесь мы говорим про перечень слов.

³⁵ Подобная аналитическая работа, безусловно, носит до известной степени субъективный характер, однако от этого она не перестает быть необходимой. Субъективность может быть нивелирована за счет инструментов внешней валидации тех концептуальных выводов, к которым пришел отдельно взятый аналитик – как с помощью обсуждений с коллегами/экспертами прототипа модели, так и обращения к объективным свидетельствам (корпоративной документации, справочникам и энциклопедиям и пр.). Более того, построение любых семантических моделей субъективно по определению хотя бы потому, что модель содержит в себе не реальные объекты действительности, а наши представления о них и их свойствах. Подробнее о соотношении объективного и субъективного при построении онтологий, см.: Горшков, С. (2016). Введение в онтологическое моделирование. Екатеринбург: ООО «ТриниДата». URL: <https://trinidata.ru/files/SemanticIntro.pdf>.

При выборе URI важно учитывать:

- обеспечение *дереференцируемости*, то есть, будет ли URI сущности эквивалентен её URL'у – другими словами, будет ли сущность из онтологии доступна по ссылке, которая образована её идентификатором. Деререференцируемость считается хорошей практикой, однако важно помнить об обеспечении стабильности URI – они не должны меняться в случае перемещения файла с онтологией по другому адресу в сети;
- названия сущностей могут отделяться знаком решётки (как в примере) или слэшем. Оба варианта допустимы, однако знак слэша позволяет деререференсировать каждую сущность в отдельную веб-страницу, тогда как при использовании решетки несколько концептов будут описаны на одной веб-странице;
- иногда в самом пространстве имён разделяются схема и экземпляры;
- хорошая практика делать URI каждой сущности также человекочитаемым (как в примере, URI класса «Краска» содержит слово «Colour»), однако допустим любой уникальный идентификатор.

По умолчанию в языке OWL не предполагается уникальность URI для каждого индивидуального объекта, то есть, если про сущности в модели явно не сказано, что они разные, то они могут быть одинаковыми. При разработке промышленных моделей для удобства может быть сделано предположение об уникальности соответствия URI и сущности реального мира, которую он обозначает.

На этапе выделения концептов выбор URI ещё не актуален, но кроме URI у каждой сущности в составе семантической модели может быть *человекочитаемое имя*. Такое имя для классов и экземпляров принято записывать с заглавной буквы в начальной форме (Именительный падеж, единственное число); для отношений – со строчной буквы, с глаголом единственного числа настоящего времени (например, «имеет дату», «является исполнителем»). Именно так лучше именовать концепты с самого начала работы с ними.

Хорошей практикой является не только формирование перечня концептов, но и фиксация их определений. Это полезно для документирования будущей семантической модели, а также для того, чтобы при работе с моделью можно было точно определить, какой смысл ее разработчики вложили в каждый концепт. Это поможет принимать решения о необходимости введения новых концептов в модель. Источниками определений могут быть как общедоступные словари, справочники и энциклопедии, так и отраслевые и корпоративные документы – стандарты, глоссарии терминов, руководящие документы, согласованное мнение экспертов/устоявшееся в корпоративной культуре определение и пр.

Для человекочитаемых имён и описаний концептов предусмотрены формальные свойства:

- `rdfs:label` для человекочитаемого наименования элемента;
- `rdfs:comment` для человекочитаемого комментария к элементу, зачастую используется для сохранения определения;
- `rdfs:seeAlso` для дополнительной информации об элементе;
- `rdfs:isDefinedBy` для информации об источнике описания элемента.

Группировка концептов и определение связей между ними

Этот шаг предполагает объединение концептов в некоторые общности, которые условно можно назвать «группами». Здесь важно соблюдать принцип логической целостности и непротиворечивости, т. е. объединять их по одному основанию. Например, если мы моделируем онтологию принадлежностей для рисования, то в группу «Краска», очевидно, должны войти концепты «Акварель», «Гуашь», «Пастель» и т. д. и не должны попасть «Кисть для живописи»,

«Мольберт», «Холст». Возможно, не для всех выделенных концептов можно будет подобрать группу – это означает, что нам пока не хватает информации для определения их места в модели.

Концепты, попавшие в группы, должны быть подвергнуты логической проверке на предмет того, является ли каждый и любой из объектов, обозначенных данным концептом, одновременно и объектом, обозначенным концептом более высокого уровня. Возвращаясь к нашему примеру, необходимо задать вопрос, является ли любая «Акварель» краской? (Да) И далее: является ли любая «Краска» принадлежностью для рисования? (В общем случае – нет, т. к. существуют краски, предназначенные для других целей; однако, если наша модель заведомо описывает только принадлежности для рисования, то да). Если на подобные вопросы даны утвердительные ответы, значит, иерархические отношения между концептами определены верно. Если нет – значит, мы неточно определили концепт (подобрали «слово») и/или неточно определили вышестоящий концепт. Сложность таких логических умозаключений заключается, в частности, в том, что при проверке корректности иерархии требуется учитывать смыслы концептов и общую цель построения данной семантической модели. Практически это выражается в том, что ответы «Да» или «Нет» следует формулировать на основе понимания того, что в данной модели мы под краской понимаем предмет, используемый для живописи, а не, например, окрашивания волос.

В семантической модели такие группы называются *классами* и их иерархия обеспечивается отношением класс-подкласс, `rdfs:subClassOf`. Оно обладает формальной семантикой, на основе которой работают машины логического вывода. Она состоит в том, что если два класса А и В связаны отношением `rdfs:subClassOf`, то есть класс А является подклассом класса В, то каждый экземпляр класса А должен также являться экземпляром класса В. Эта характеристика отношения класс-подкласс называется *транзитивностью*, и описанная выше проверка необходима для её обеспечения.

Выделение классов семантической модели и определение их иерархии может проводиться несколькими способами³⁶:

- «снизу вверх»: от классов предметного уровня к классам высокого уровня;
- «сверху вниз»: от классов более высокого уровня абстракции (включая онтологию верхнего уровня и предметную онтологию) к классам более низкого уровня;
- «из середины»: от ключевых терминов предметной онтологии, для каждого из которых затем определяются более абстрактные и более конкретные классы³⁷.

Выбор того или иного способа, по большому счету, зависит от личных предпочтений аналитика, а также от объема доступной информации при реализации конкретного проекта. Возможны ситуации, когда по какому-либо фрагменту предметной области пока недостаточно информации для того, чтобы можно было провести детальное моделирование, и тогда приходится формировать иерархию более общего характера. После получения необходимой информации семантическая модель может быть дополнена или подвергнута рефакторингу.

Фактически, выделяя концепты и организуя их в классы по определенным правилам логики, аналитик проводит классификацию (типизацию) сущностей, т. е. описывает классы объектов и выстраивает иерархические отношения между ними (класс-подкласс). Методологически такая типизация опирается на теорию множеств³⁸.

³⁶ Uschold, M., Grüninger, M. (1996). Ontologies: Principles, Methods and Applications. Knowledge Engineering Review, 11(2), 93–136.

³⁷ См. об этом, например: Гаврилова, Т. А., Кудрявцев, Д. В., Муромцев, Д. И. (2016). Инженерия знаний. Модели и методы. СПб: Лань. С. 211–212.

³⁸ Не углубляясь подробно в методологию определения множеств и классов, сделаем отсылку к работе Горшков, С. В., Кралин, С. С., Муштак, О. И., Гумеров, С. З., Мирошниченко, М. Г., Гребешков, А. Ю., Шибалов,

В онтологическом моделировании каждый концепт можно отнести одновременно к двум или более классам – это позволяет реализовать *фасетную классификацию*, когда классификация производится параллельно по нескольким основаниям. Например, краску как принадлежность для рисования можно также рассматривать и как товар, который продается в художественном магазине. Важно, что при этом краска не перестает быть краской в том смысле, как мы его определили в контексте данной семантической модели, однако семантическая модель позволяет нам описать краску как концепт в нескольких «разрезах». Графически такая фасетная классификация может быть представлена следующим образом:

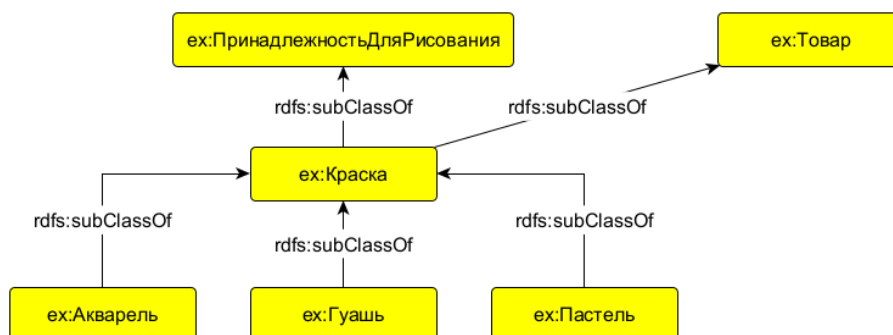


Рис. 5. Фасетная классификация элементов модели

Этот прием моделирования (называемый еще *множественным наследованием*) придает нашей онтологии бóльшую выразительность и позволяет воспользоваться некоторыми преимуществами, что будет проиллюстрировано ниже. Важно, что при таком наследовании всё ещё должна соблюдаться семантика отношения `rdfs:subClassOf`: в примере на рисунке 5 все конкретные гуаши, акварели и пастели должны являться одновременно и товарами, и принадлежностями для рисования.

Выделение концептов, организацию их в классы и надклассы необходимо проводить до достижения той степени выразительности, которая достаточна на данном этапе моделирования с точки зрения решения поставленной прикладной задачи. Безусловно, в один или два приема трудно описать весь набор необходимых концептов и классов – семантическая модель достраивается постепенно, по мере необходимости и по мере получения новой информации.

Помимо иерархических отношений на этом шаге полезно описать отношения «часть-целое» (например, `ex:Ручка ex:являетсяЧастью ex:КистьДляРисования` – это так называемое мерееологическое деление), а также выделить иные отношения между концептами, основываясь на целях построения данной семантической модели, т. е. ее прагматике, прикладном значении. Такими прагматическими отношениями могут быть отношения связности (`ex:Краска ex:имеетФирмуПроизводителя ex:ФирмаПроизводитель`), отношения соответствия (`ex:Кисть ex:используетсяДля ex:МаслянаяКраска`) и любые другие, позволяющие описать смыслы, релевантные для данной предметной области и цели построения семантической модели. Эти отношения не определены в языках семантической паутины, однако некоторые из них определены в онтологиях верхнего уровня или предметных онтологиях. Так, группа мерееологических отношений «part of» определена в онтологии BFO, а в онтологии организации Organization Ontology определены такие отношения, как «member of» (для обозначения членов организации).

Р. Ю. (2019). Онтологическое моделирование предприятий: методы и технологии. Екатеринбург: Издательство Уральского университета. URL: <https://trinidadata.ru/files/EnterpriseModeling.pdf>. В частности, см. главу 5.

Создание свойств

Свойства необходимы, чтобы обеспечить выражение разнообразных характеристик, которыми могут обладать объекты, входящие в классы. Строго говоря, объекты группируются в соответствующие классы (множества) именно потому, что они обладают однотипными свойствами. При этом важно, чтобы свойства, на основании которых объекты включаются в классы, были *жесткими* (rigid), т. е. объект обладал ими на всем протяжении своего существования. Однако при построении семантической модели, которая является некоторого рода абстракцией, аналитики зачастую пренебрегают представлениями о том, что объекты с их свойствами первичны, и рассуждают о «свойствах классов» вместо «свойств объектов классов».

Также свойства могут быть формализованы на основе слов, которые были сформированы при описании предметной области и которые не были определены в качестве классов семантической модели³⁹. Третьим способом выявления свойств является анализ связей классов модели и оценка необходимости формализации каких-либо отношений между объектами.

Свойства в семантической модели обладают несколькими особенностями:

1. Свойства наследуются объектами от всех своих надклассов. Например, если мы определим для класса `ex:Кисть` свойства `ex:имеетМатериалЩетины` и `ex:общаяДлина`, то эти свойства окажутся применимы для всех объектов всех подклассов этого класса. То есть, и все малярные кисти, и все кисти для рисования будут характеризоваться материалом щетины:

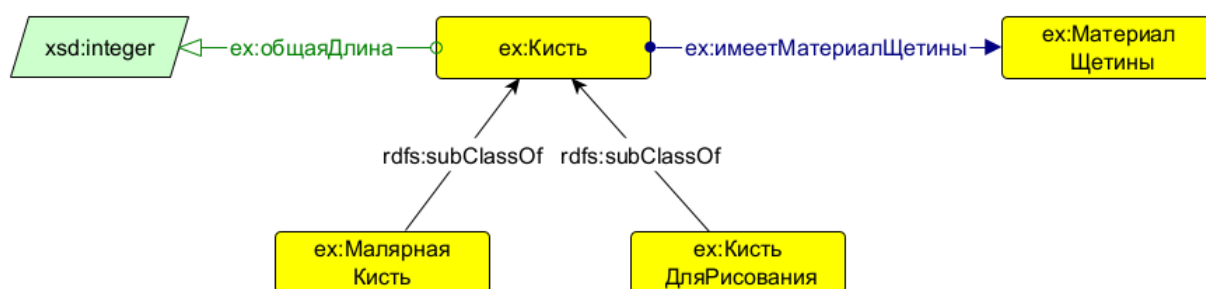


Рис. 6. Свойства класса `ex:Кисть`

2. Одно и то же свойство может быть определено для разных классов семантической модели. Например, свойство «высота» может быть отнесено к объектам класса «Дерево» и объектам класса «Дом». Важно, что технически такое свойство будет иметь один идентификатор (URI), и семантически это одна и та же высота – например, высота объектов, измеренная от земли в метрах (а не над уровнем моря в километрах). При этом следует стремиться определять свойство для максимально высокого класса в иерархии, принимая во внимание, что свойство должно быть применимо к объектам всех его подклассов. В противном случае модель рискует стать семантически избыточной, когда область применения свойства будет включать несколько классов одного уровня, что усложнит работу с этим свойством в будущем.

Эти особенности позволяют семантическим моделям быть лаконичными без потери выразительности и более эффективными и гибкими, чем реляционные. Допустим, нам

³⁹ В связи с этим могут возникать ситуации, когда приходится принимать решение о том, как оптимально смоделировать какую-либо сущность – как класс (тип) или как свойство. См. об этом: Горшков, С. В., Кралин, С. С., Муштак, О. И., Гумеров, С. З., Мирошниченко, М. Г., Гребешков, А. Ю., Шибалов, Р. Ю. (2019). Онтологическое моделирование предприятий: методы и технологии. Екатеринбург: Издательство Уральского университета. URL: <https://trinidad.ru/files/EnterpriseModeling.pdf>. С. 144–146.

потребовалось найти, какие объекты могут быть упакованы в ящик длиной 50 см. В случае с семантической моделью для этого достаточно будет сделать один запрос с условием на длину, в ответ будут возвращены все подходящие объекты. В случае с реляционными моделями потребуется создавать специальную логику, потому что для каждой таблицы поле «Длина» – это самостоятельное поле, и база данных «не знает» о том, что поле с таким названием в каждой таблице означает одну и ту же характеристику.

3. Свойства обладают областью применения и диапазоном значений:

- *область применения* (domain) – это класс (или классы), объекты которых могут обладать значениями данного свойства;
- *диапазон значений* (range) – это те типы значений, которые может принимать свойство.

Свойства могут принимать литеральные значения, или их значениями могут быть объекты каких-либо классов (в том числе и того класса, который установлен в области применения данного свойства). К литеральным значениям относятся строки, логические типы данных, целые числа, числа с плавающей запятой, даты и пр. Разновидности литеральных значений описываются спецификациями RDFS/OWL⁴⁰, использующими в этой части спецификацию XSD⁴¹.

Соответственно, с точки зрения хранимых данных, семантическая модель поддерживает два типа свойств: *литеральные* (owl:DatatypeProperty) и *ссылочные* (owl:ObjectProperty). На приведенной выше диаграмме свойство ex:общаяДлина является литеральным, и имеет тип данных «целое число» (xsd:integer); свойство ex:имеетМатериалЩетины является ссылочным, в диапазон его значений входят объекты класса ex:МатериалЩетины.

4. Свойства могут иметь ограничения на количество хранимых значений. Каждое свойство семантической модели по умолчанию может иметь неограниченное количество значений для каждого объекта, а также может не хранить ни одного значения (т. е. быть необязательным для заполнения). Однако в некоторых случаях, исходя из конкретных бизнес-задач, может быть полезно установить ограничения на минимальное и максимальное количество значений данного свойства – определить так называемые минимальную и максимальную кардинальность.

Например, если необходимо описать в семантической модели такое свойство краски, как цвет, можно использовать ссылочное свойство ex:имеетЦвет, установив для него минимальное количество значений равным «1». Технически это будет означать, что мы не сможем создать/отредактировать в модели ни одного объекта класса ex:Краска и его подклассов, не указав значения свойства ex:имеетЦвет. Если нам необходимо указать в семантической модели, сколько осталось краски как товара, предназначенного для продажи, можно классу ex:Товар определить литеральное свойство ex:остатокДляПродажи, установив для него минимальное количество значений равным «0», а максимальное – равным «1»⁴². При работе с моделью это будет означать, что нам необязательно указывать, сколько краски каждого типа имеется в свободной продаже в текущий момент времени, если нам по каким-то причинам недоступна эта информация, а если информация доступна – это всегда одно конкретное значение.

⁴⁰ <https://www.w3.org/TR/2012/REC-owl2-quick-reference-20121211/>

⁴¹ <https://www.w3.org/TR/xmlschema11-1/>; <https://www.w3.org/TR/xmlschema11-2/>

⁴² Необходимо учитывать, что при разработке программного обеспечения, работающего с данными семантической модели, значение owl:maxCardinality > 1 будет иметь следствием то, что программисту потребуется описывать обработку значения такого свойства как массива, что может создавать определенные сложности. Поэтому рекомендуется во всех возможных случаях устанавливать owl:maxCardinality = 1.

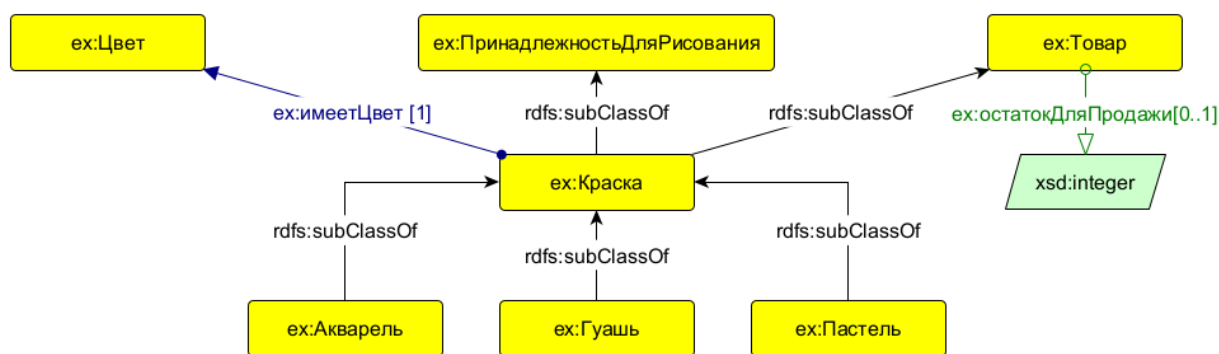


Рис. 7. Литеральное и ссылочное свойства с ограничением на количество значений

5. Свойства обладают собственными характеристиками, которые определяют «поведение» свойства в процессе логического вывода. Язык OWL предоставляет возможность использовать несколько характеристик свойств⁴³:

- *симметричность*: если объект А связан с объектом В симметричным свойством, то и объект В будет связан с объектом А таким же свойством. Например, свойство «имеет супруга» является симметричным;
- *асимметричность*: если объект А связан с объектом В асимметричным свойством, то объект В не может быть связан с объектом А таким же свойством, например, «имеет ребёнка»;
- *дизъюнктивность*: два объекта не могут быть одновременно связаны дизъюнктивными свойствами, например, объект А не может быть связан с объектом В одновременно свойствами «имеет супруга» и «имеет ребёнка»;
- *рефлексивность*: таким свойством объект может быть связан сам с собой, например, свойство «имеет родственника» может быть рефлексивным, т. к. каждый человек является собственным родственником;
- *иррефлексивность*: таким свойством объект не может быть связан сам с собой, например, человек не может быть собственным супругом, значит, такой характеристикой должно обладать свойство «имеет супруга»;
- *функциональность*: может быть не более одного объекта В, с которым объект А может быть связан функциональным свойством, «имеет супруга» является таким свойством;
- *обратная функциональность*: только с одним объектом А объект В может быть связан этим свойством, свойство «имеет супруга» (по крайней мере в нашей культуре) является также и обратно функциональным свойством;
- *транзитивность*: если объекты А и В связаны этим свойством, и объекты В и С связаны таким свойством, то объекты А и С тоже будут связаны таким свойством, `rdfs:subClassOf` пример такого свойства.

В платформе АрхиГраф данный функционал реализован через конфигурацию свойств в настройках платформы виртуализации данных АрхиГраф.MDM⁴⁴ и создании соответствующих правил логического вывода в системе управления знаниями АрхиГраф.СУЗ⁴⁵.

6. Семантические модели включают «стандартные» свойства, то есть такие, которые определены какой-либо спецификацией W3C, а не созданы разработчиком данной модели. К таким свойствам относятся, например, упомянутые в предыдущих разделах и определённые в спецификации RDF Schema `rdfs:subClassOf`, `rdfs:label`, `rdfs:comment`,

⁴³ Примеры взяты из спецификации OWL 2: https://www.w3.org/TR/owl2-primer/#Property_Characteristics.

⁴⁴ <https://trinidata.ru/files/ArchiGraphMDMAdminGuide.pdf>

⁴⁵ <https://trinidata.ru/files/ArchiGraphSUZUserGuide.pdf>

`rdfs:seeAlso`, `rdfs:isDefinedBy`. При необходимости семантическая модель может расширяться за счет специализированных онтологий, предоставляющих дополнительный арсенал свойств для описания ее элементов⁴⁶.

7. Семантические модели поддерживают многоязычность. Технически это выражается в том, что для одного объекта модели могут быть заданы значения каждого строкового свойства на разных языках. Это позволяет обеспечить доступность информации о содержании семантической модели носителям этих языков средствами самой модели. Также это дает возможность создавать мультязычные пользовательские интерфейсы бизнес-приложений, которые работают с семантической моделью, без использования дополнительных программных средств (автоматических переводчиков и пр.).

Например, в онтологии Time Ontology⁴⁷ пятница как день недели описывается следующим образом:

```
:Friday
  rdf:type :DayOfWeek ;
  rdfs:label "Friday"@en ;
  skos:prefLabel "Freitag"@de ;
  skos:prefLabel "Friday"@en ;
  skos:prefLabel "Piątek"@pl ;
  skos:prefLabel "Sexta-feira"@pt ;
  skos:prefLabel "Vendredi"@fr ;
  skos:prefLabel "Venerdì"@it ;
  skos:prefLabel "Viernes"@es ;
  skos:prefLabel "Vrijdag"@nl ;
  skos:prefLabel "Пятница"@ru ;
  skos:prefLabel "الجمعة"@ar ;
  skos:prefLabel "星期五"@zh ;
  skos:prefLabel "金曜日"@ja .
```

Здесь строковое свойство `skos:prefLabel` онтологии SKOS используется для записи множества обозначений сущности «Пятница как день недели» на языках, отличных от английского. Для обозначения этой сущности «по умолчанию», на английском языке использовано стандартное свойство `rdfs:label`.

Работа с индивидуальными объектами классов

Индивидуальный объект класса семантической модели – это такой ее элемент, который обладает уникальным значением идентификатора, может обладать определенными значениями свойств и входить в один или несколько классов (множеств). Иначе говоря, *индивидуальный объект* (`owl:NamedIndividual`) является уникальным экземпляром какого-либо класса (или классов) объектов. Для связи экземпляров с соответствующими классами используется отношение `rdf:type`, определённого в спецификации RDF. Обычно индивидуальные объекты отражают в модели сведения о конкретных объектах моделируемого мира.

Вопрос о том, чем по своей природе является индивидуальный объект и какие приемы помогают определить, как оптимально смоделировать какую-либо сущность (как индивидуальный объект или класс) является не таким уж простым, и в общем случае зависит от прагматики семантической модели. Основной принцип здесь состоит в том, что в качестве индивидуальных

⁴⁶ См., например: <https://www.dublincore.org/specifications/dublin-core/dcmi-terms/>.

⁴⁷ <https://www.w3.org/TR/2020/CR-owl-time-20200326/>; <https://www.w3.org/2006/time>

объектов модели мы представляем сущности, которые входят во множества (классы) и имеют конкретные значения свойств – то есть такие сущности, факты о которых мы хотим предоставлять в модели. Здесь большую роль играет цель создания модели. Так, например, для модели лакокрасочного производства важна информация о составляющих частях краски (их хранении, транспортировке и т. д.) – в этом случае экземплярами станут пигменты и другие составные части, а для модели магазина красок такая информация избыточна – здесь скорее всего потребуется описать конкретные экземпляры упаковок красок различных типов. При этом мы можем использовать свойство, которое позволяет уникальным образом идентифицировать их в ряду однородных объектов. Таким свойством может быть серийный номер упаковки, как показано на следующей диаграмме:

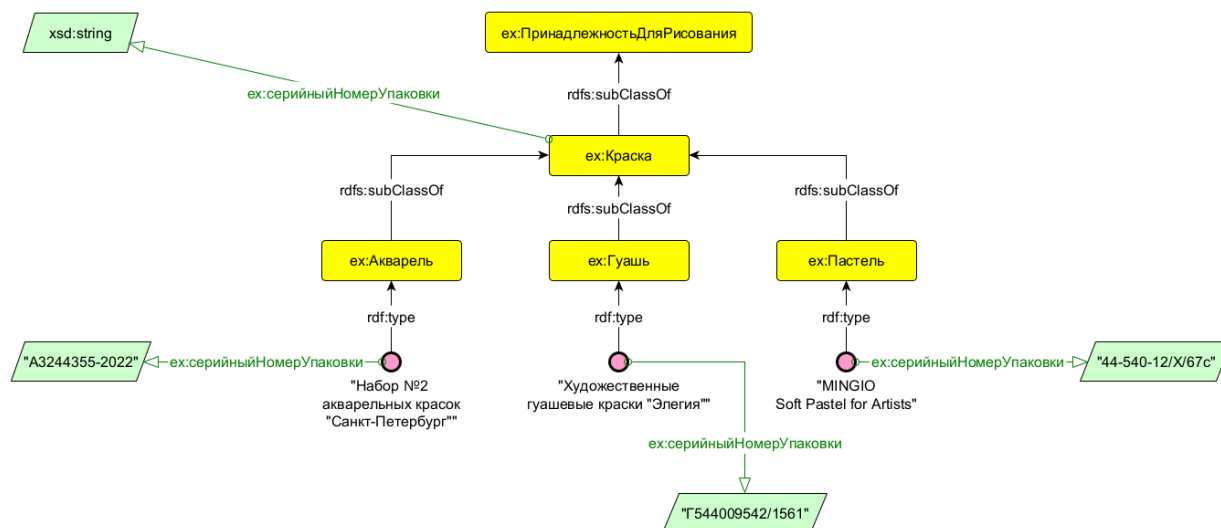


Рис. 8. Уникальные значения свойства индивидуальных объектов классов

Литеральное свойство `ex:серийныйНомерУпаковки` определено для надкласса классов `ex:Акварель`, `ex:Гуашь`, `ex:Пастель`, т. к. оно является общим для данных классов. Это свойство наследуется от их надкласса `ex:Краска` и для каждого объекта подклассов может принимать уникальные значения⁴⁸.

Поскольку классы организованы по принципу иерархии, их индивидуальные объекты являются членами не только того класса, в который они непосредственно входят, но и всех надклассов данного класса. Это позволяет использовать свойства любого из надклассов иерархии для описания данного объекта. Дополним приведенную выше диаграмму следующим образом:

⁴⁸ Данный пример является условным и используется только для демонстрации возможностей семантического моделирования. Любые совпадения с реальными типами и видами лакокрасочной продукции, а также их характеристиками и производителями случайны.

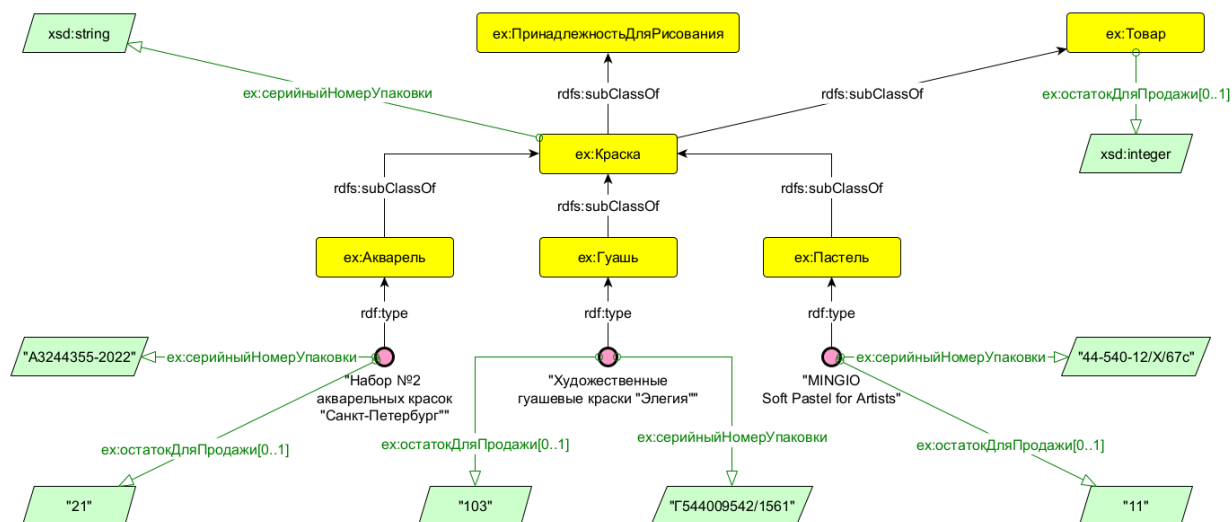


Рис. 9. Значения свойств индивидуальных объектов от двух надклассов

Индивидуальные объекты классов `ex:Акварель`, `ex:Гуашь`, `ex:Пастель` могут обладать значениями свойств, унаследованных не только от класса `ex:Краска`, но и от его надкласса – класса `ex:Товар`.

Управлять множественностью вхождения объектов в разные классы можно не только со стороны иерархии классов, но и самих объектов. Мы уже упоминали о том, что каждый индивидуальный объект может одновременно относиться к нескольким классам. Результат такого вхождения не будет отличаться от случая, рассмотренного выше.

Если в семантической модели имеются индивидуальные объекты, которые необходимо описать в совершенно разных аспектах, т. е. использовать разные ветки классов и надклассов с их свойствами, целесообразно включить их в несколько классов. Например, город может рассматриваться в аспекте муниципальной и административно-территориальной принадлежности, а также в аспекте туристической привлекательности. Очевидно, что в обоих этих аспектах данный город сохраняет свою идентичность (остается одной и той же сущностью):

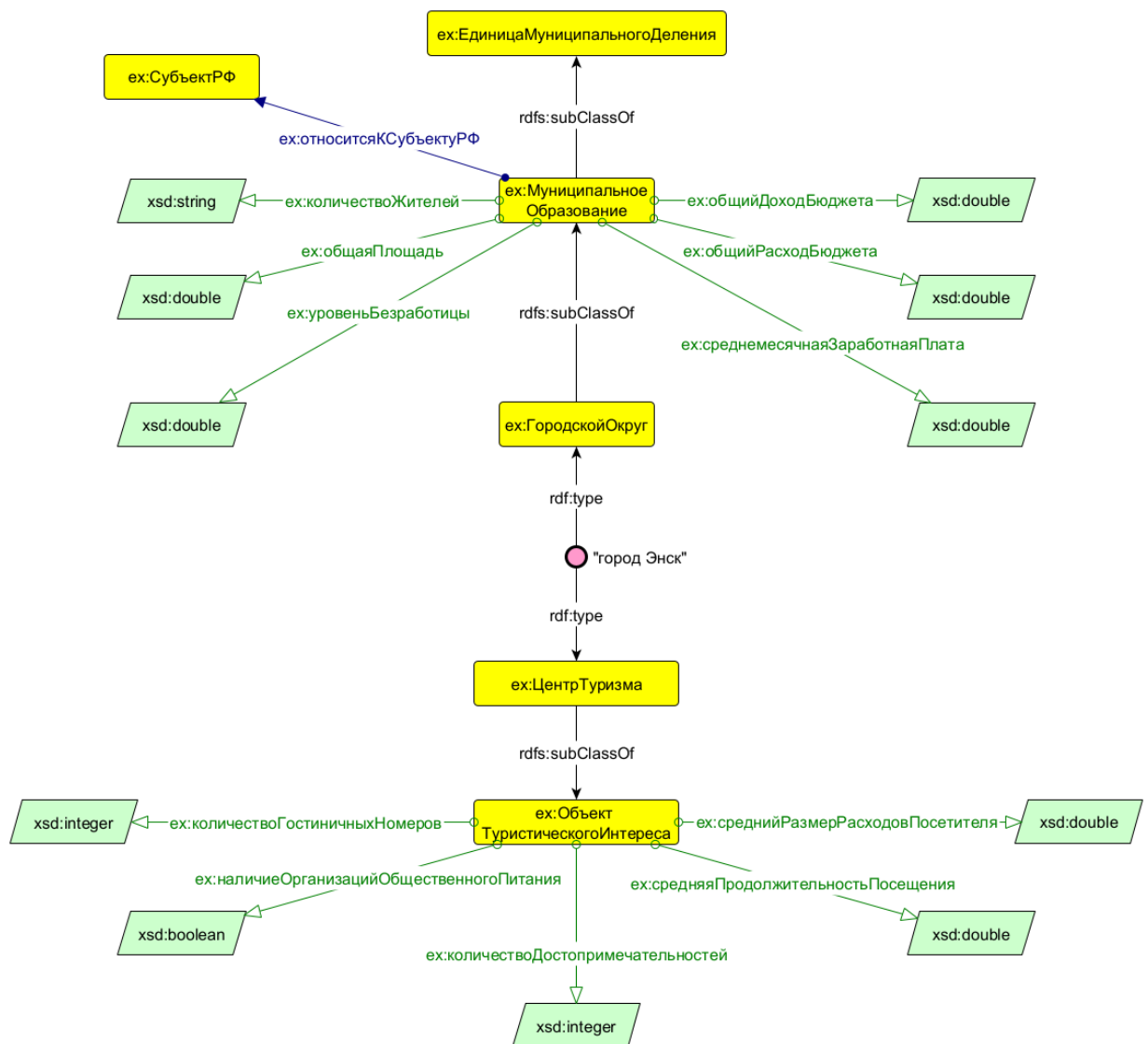


Рис. 10. Включение индивидуального объекта в несколько классов

В результате индивидуальный объект «город Энск» может быть описан с помощью всей совокупности свойств классов обеих веток – как единица муниципального деления и как объект туристического интереса⁴⁹. Этот прием моделирования увеличивает выразительность индивидуальных объектов онтологии.

⁴⁹ Внимательный читатель может заметить, что город Энск как городской округ должен называться «городской округ "город Энск"», что вызовет «номинативный конфликт», если его одновременно рассматривать как туристический объект (подобное наименование может вызвать недоумение у гостей города и вряд ли будет способствовать его туристической привлекательности). Подобные коллизии в семантической модели разрешаются за счет использования свойств онтологии SKOS (в частности, skos:prefLabel и skos:altLabel). См. подробнее: <https://www.w3.org/TR/swbp-skos-core-guide/>, <https://www.w3.org/TR/skos-primer/>.

Моделирование отдельных видов информации

Создание семантических моделей может предполагать использование некоторых паттернов, которые позволяют описать типовые сценарии представления информации. Рассмотрим некоторые частные примеры таких паттернов. В качестве иллюстраций в этом разделе представлены дизайн-решения, которые мы используем в разработке своих моделей. Они могут быть повторно использованы при разработке других моделей, однако в подавляющем большинстве случаев будут нуждаться в адаптации к целям разрабатываемой модели.

Создание классов характеристик объектов

Помимо классов, описывающих бизнес-объекты, в семантической модели зачастую необходимы классы, объекты которых используются при характеристике этих бизнес-объектов. Например, кисти для рисования могут обладать такими характеристиками:

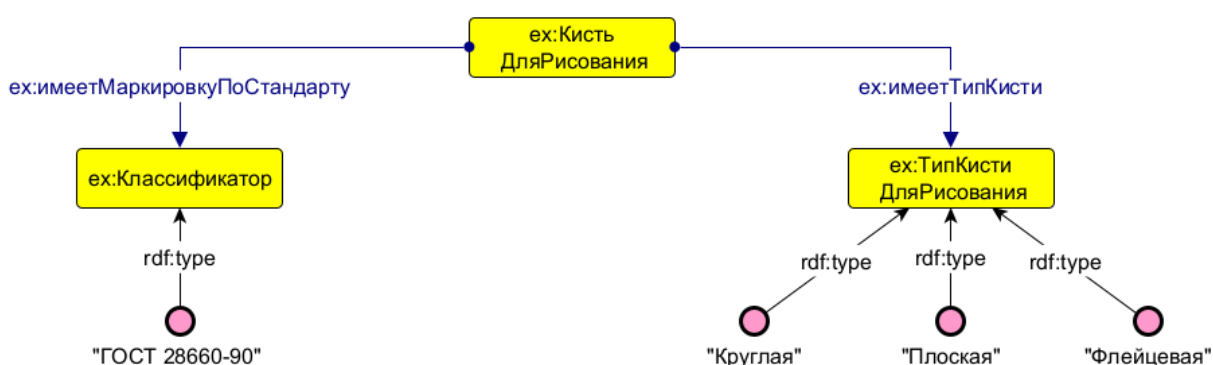


Рис. 11. Использование класса НСИ

В данном случае, класс характеристик `ex:ТипКистиДляРисования` включает три объекта, которые могут выступать в качестве значений свойства `ex:имеетТипКисти` класса `ex:КистьДляРисования`.

Возникает вопрос – почему в этом случае мы использовали ссылку `ex:имеетТипКисти` на объекты классификатора, а не создали подклассы класса `ex:КистьДляРисования`? Ответы на подобные вопросы должны содержаться в методике моделирования, принятой в рамках конкретного проекта. Мы считаем, что подклассы целесообразно выделять, если объекты подкласса могут иметь значения свойств, которыми не обладают объекты надкласса (например, есть какая-то особая характеристика именно плоских кистей). Предпочтение ссылке на характеристику следует отдавать, если у неё есть значения собственных свойств – т. е. если нам нужно выразить факты обо всех круглых или плоских кистях. В принципе, собственными значениями свойств может обладать и класс, но такой паттерн моделирования (он называется *ripping*) не желательно применять без необходимости.

В составе онтологической модели такие характеристики могут быть подклассами соответствующих классов выбранной онтологии верхнего уровня. Например, в онтологии ВФО они будут подклассами «Качества» – свойства объекта, которое не существует само по себе, но проявляется в соответствующем объекте (например, цвет или форма) или «Предрасположенности» – свойства объекта, проявляющегося в процессе какой-либо деятельности (например, должность человека или режим работы прибора). В приведённом примере плоскость или круглость кисти не существуют сами по себе, это «Качества», которые проявляются в определённой кисти, при этом в разных кистях могут проявляться по-разному. Сама

кисть при этом будет подклассом «Объекта» в BFO – такой сущности, которая существует независимо, то есть, не требует никакой другой сущности, чтобы проявиться.

Другой тип характеристик объектов представлен нормативными данными – такие характеристики часто не отражаются напрямую в структуре объекта, но важны для его функционирования. Это, например, категории земель или степени защиты приборов: первые описывают, для чего могут использоваться определённые территории, вторые – насколько прибор защищён от проникновения пыли и воды. Эта информация ничего не говорит нам о физическом представлении земель и приборов, но важна для процессов, в которых они могут участвовать. Ещё одна важная характеристика такой информации состоит в том, что обычно её регламентирует какой-то нормативный документ: закон, подзаконный акт, стандарт, техническая спецификация и т. п. В терминологии онтологии верхнего уровня BFO и её расширения IAO классы, соответствующие таким документам, станут подклассами «Информационной сущности» в онтологии IAO и могут быть объединены в подкласс `ex:СправочныеДанные` или `ex:Классификатор`⁵⁰. Экземпляром такого класса может быть, например, документ ГОСТ, в соответствии с которым маркирована данная кисть. В таком случае экземпляры класса `ex:КистьДляРисования` и «ГОСТ 28660-90» будут связаны свойством `ex:имеетМаркировкуПоСтандарту`. Сами значения, описанные нормативными документами (в приведённом примере – коды маркировок), будут экземплярами другого подкласса «Информационной сущности» (например, `ex:КодМаркировкиКисти`).

Различие между классами характеристик бизнес-объектов и классами НСИ состоит еще и в том, что первые практически постоянно пополняются новыми объектами в ходе выполнения бизнес-процессов в текущей деятельности организации. Классы НСИ являются в этом смысле более стабильными и «закрытыми»: изменение состава их индивидуальных объектов происходит при изменениях соответствующих нормативных источников или корректировках классификаций, и подразумевается, что эти классы не содержат никаких объектов, кроме явно перечисленных.

Моделирование темпоральных характеристик и отношений

До сих пор мы сознательно не затрагивали в наших примерах моделирование темпоральных (временных) свойств и отношений для упрощения восприятия, однако при работе с реальной семантической моделью избежать описания изменений объектов и их свойств во времени не удастся. Иначе выразительные возможности такой модели окажутся существенно ограниченными, и она не будет выполнять одну из своих главных задач – представлять бизнес-объекты и бизнес-процессы в их взаимосвязи. Не вдаваясь в феноменологическую природу времени и представлений человека о нем⁵¹, сконцентрируемся на том, как можно описать в семантической модели сущности, имеющие временную протяженность (длительность).

⁵⁰ О методологии и методиках разработки классификаторов см.: Горшков, С. В., Кралин, С. С., Муштак, О. И., Гумеров, С. З., Мирошниченко, М. Г., Гребешков, А. Ю., Шебалов, Р. Ю. (2019). Онтологическое моделирование предприятий: методы и технологии. Екатеринбург: Издательство Уральского университета. URL: <https://trinidad.ru/files/EnterpriseModeling.pdf>. С. 91–104.

⁵¹ Применительно к вопросам онтологического моделирования см. об этом, например: Sowa, J. F. (1999). Knowledge representation: logical, philosophical and computational foundations. Brooks/Cole Publishing Co.; Keet, M. (2018). An introduction to ontology engineering. Cape Town: Maria Keet. URL: <https://open.umn.edu/opentextbooks/textbooks/590>; Горшков, С. В., Кралин, С. С., Муштак, О. И., Гумеров, С. З., Мирошниченко, М. Г., Гребешков, А. Ю., Шебалов, Р. Ю. (2019). Онтологическое моделирование предприятий: методы и технологии. Екатеринбург: Издательство Уральского университета. URL: <https://trinidad.ru/files/EnterpriseModeling.pdf>. С. 72–79.

Представление в семантической модели сущностей, описывающих временные характеристики, требует использования классов «Процесс» и «Событие». Сами по себе эти классы носят абстрактный характер, их объекты описывают факты, свидетельствующие о том, что с определенными объектами модели что-то произошло (или происходит) в определенный момент времени. Процессы, таким образом, представляют собой изменения, происходящие с объектами.

Принципиальная разница между процессами и событиями заключается в «мгновенности»: процессы имеют некоторую длительность, которая ограничена моментом начала и может быть ограничена моментом окончания. «Может быть» – потому, что момент окончания не обязательно точно известен, но всегда подразумевается. События длительности не имеют (или ею можно пренебречь в конкретных бизнес-контекстах), они случаются в определенный момент времени.

Описание средствами семантической модели процессов и событий предполагает установление нескольких связей между объектами классов, участвующих в них. В случае с моделированием «мгновенных» событий классу «Событие» целесообразно определить свойство `ex:имеетДату` (со значением типа `xsd:date`) или `ex:имеетДатуВремя` (со значением типа `xsd:dateTime`), а также связи с объектами, участвующими в этом событии, например:

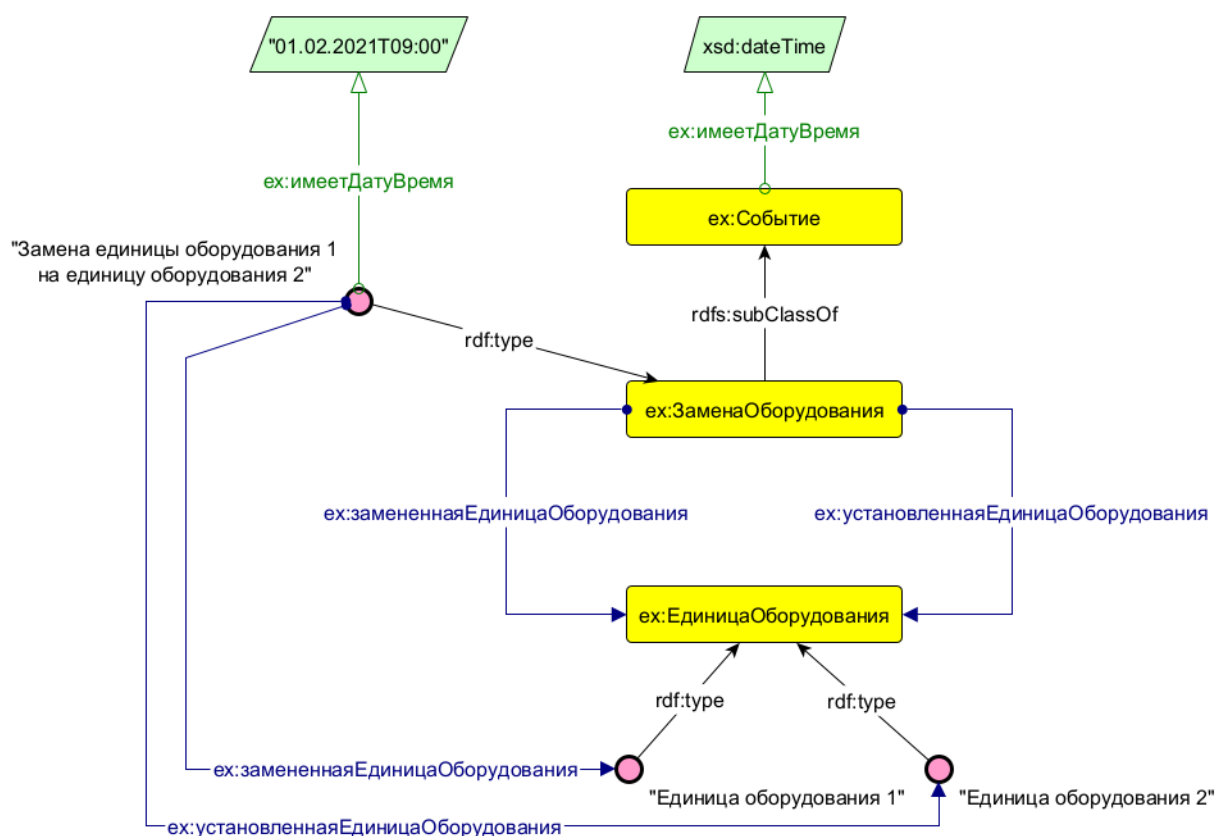


Рис. 12. Представление конкретного события в семантической модели

Обратим внимание, что дата события, произошедшего с объектом – это не свойство объекта «Единица оборудования 1», с которым произошло событие, а свойство объекта, представляющего само событие «Замена единицы оборудования 1 на единицу оборудования 2». Это так, потому что время замены одной единицы оборудования на другую ничего не говорит нам о характеристиках, присущих самой заменённой единице оборудования. Кроме того, сама эта заменённая единица оборудования может дальше участвовать в других событиях. Определять темпоральные свойства непосредственно самим бизнес-объектам (физическим лицам, единицам оборудования и пр.) в подобных случаях было бы неверно и потому,

что задачей является не описание их жизненного цикла, а фиксация изменения бизнес-ландшафта, в котором они задействованы. Кроме этого, благодаря описанному подходу в семантической модели появляются информационные объекты, с которыми можно работать, как с «журналом событий»⁵².

Моделирование состояний

В отличие от процесса, который отражает непосредственно изменение объекта, состояние – это что-то, присущее объекту определённый период времени. Состояние объекта может меняться, но при этом само по себе стабильно и не является изменением как таковым: сотрудник компании может менять штатные должности, и в каждой должности у этого сотрудника будут разные функции, но стабильные в рамках конкретной должности. Смена сотрудника, замещающего ту или иную штатную должность – это событие с точки зрения бизнес-процесса и временная характеристика с точки зрения семантической модели. Нахождение сотрудника на должности – это осуществление определённого набора функций с точки зрения бизнес-процесса и характеристика состояния с точки зрения семантической модели.

Вводя в модель класс «Состояние», мы должны определить его место по отношению к другим классам. В онтологическом смысле состояние – это характеристика объекта, которая проявляется в определённом процессе. Так, характеристика сотрудника «исполнять должность инспектора» будет проявляться в процессе «проведение инспекции». В онтологии верхнего уровня VFO такие характеристики объектов называются «Исполняемыми сущностями» и включают в себя «Роли» (например, быть врачом), «Предрасположенности» (например, для меди – проводить электрический ток) и «Функции» (для двигателя – преобразовывать энергию в работу).

Например, если нам необходимо выразить, что физическое лицо на протяжении какого-то времени занимало должность в структурном подразделении компании, это можно сделать следующим образом:

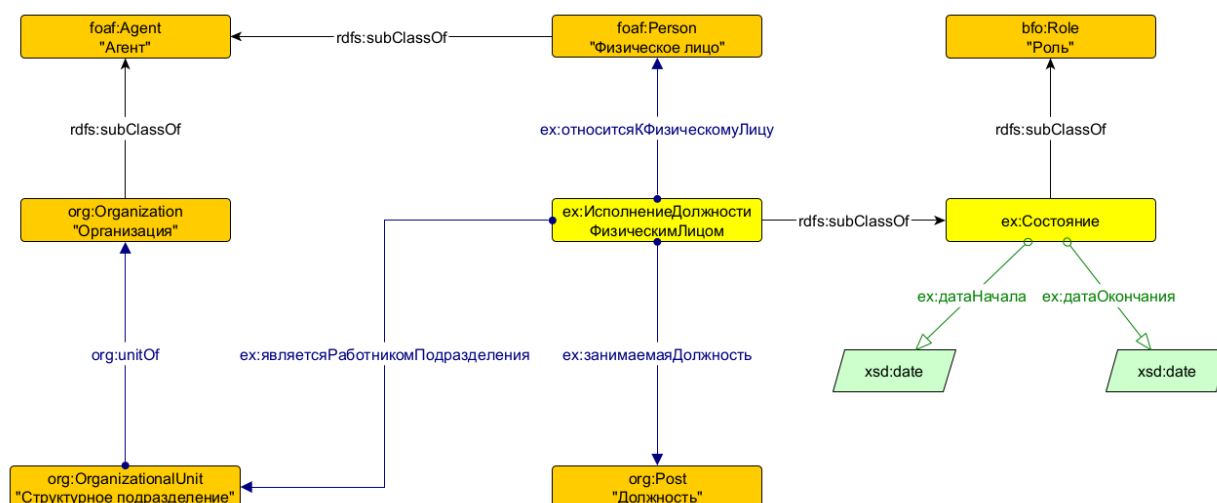


Рис. 13. Модель исполнения должности физическим лицом

Класс `ex:ИсполнениеДолжностиФизическимЛицом` используется для записи темпоральных отношений между объектами классов `foaf:Person` и `org:Post` с помощью свойств `ex:датаНачала` и `ex:датаОкончания`. Дополним данный TBox объектами ABox:

⁵² См. подробнее: Горшков, С. (2016). Введение в онтологическое моделирование. Екатеринбург: ООО «ТриниДата». URL: <https://trinidata.ru/files/SemanticIntro.pdf>. С. 135–137.

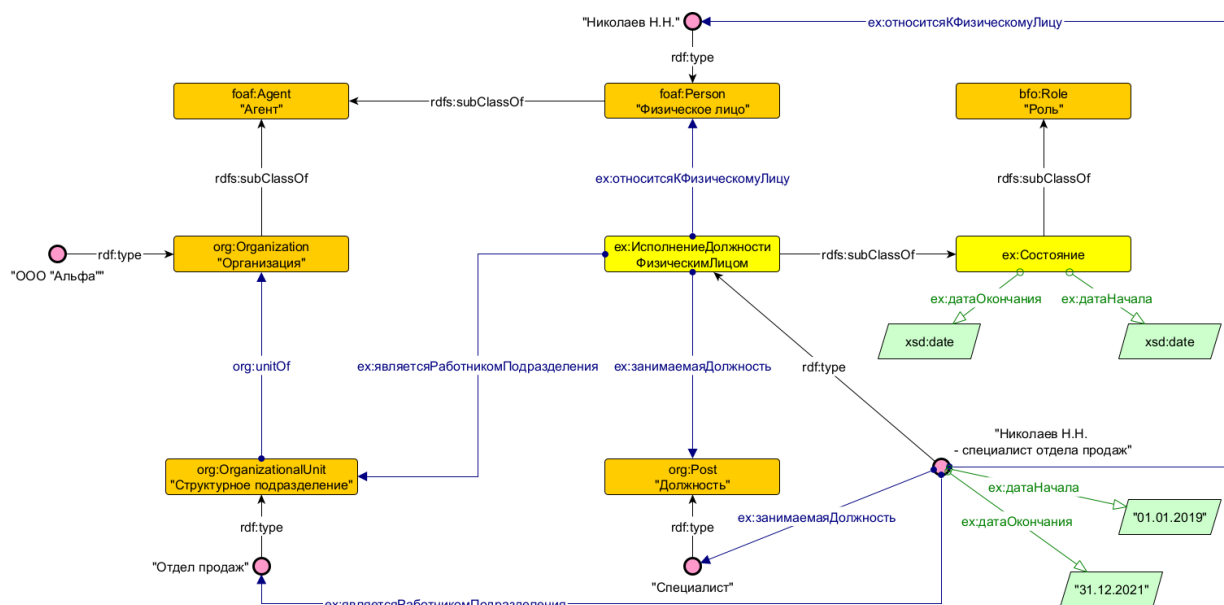


Рис. 14. Сведения об исполнении определенной должности конкретным физическим лицом

Индивидуальный объект «Николаев Н.Н. – специалист отдела продаж» класса `ex:ИсполнениеДолжностиФизическимЛицом`, как следует из диаграммы, агрегирует информацию о том, кто, на протяжении какого периода, в каком отделе и в какой должности исполнял свои обязанности. Читаемое наименование объекта может быть любым, его основное назначение – дать описание связей объектов семантической модели в разрезе времени.

Использование приема моделирования, который заключается в создании промежуточных классов, необходимых для связывания объектов других классов, является достаточно распространенным и часто применяется для описания темпоральных отношений.

Моделирование адресов

Указание адресов мест расположения бизнес-объектов является достаточно распространенной задачей для информационных систем. Решить эту задачу можно несколькими способами. Например, можно описать адрес с помощью строкового свойства («адрес», «адрес полный», «адрес строкой» и т. п.) или набора строковых свойств, каждый из которых будет хранить определенный компонент адреса («город», «улица», «дом», «помещение» и пр.). Такие свойства можно определить классам соответствующих бизнес-объектов и заполнять необходимой информацией. В том случае, если адресов относительно немного (до нескольких сотен) и наборы значений пополняются/обновляются относительно редко, этого может быть достаточно.

Второй вариант описания адресных свойств – формирование информационного набора сведений об адресе с помощью онтологического представления `vCard`⁵³. В результате можно получить адресный «пакет», который можно связать с классом необходимых бизнес-объектов свойством-ссылкой:

⁵³ <https://www.w3.org/Submission/vcard-rdf/>

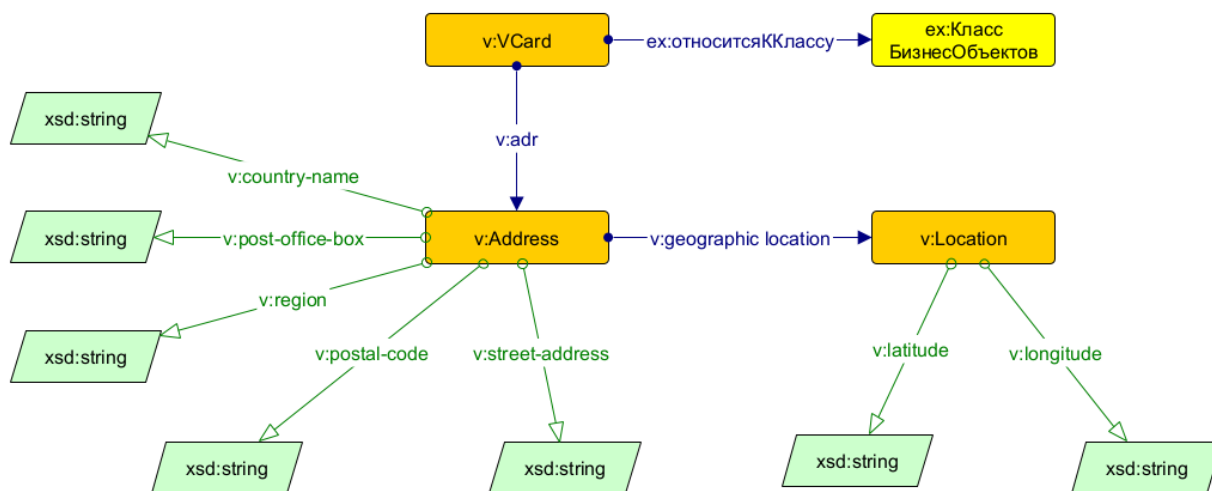


Рис. 15. Примерное представление адресной модели vCard (фрагмент)

Преимуществом использования такого или похожего подхода является возможность описывать случаи, когда несколько бизнес-объектов имеют один и тот же физический адрес, что делает модель данных более экономичной, а управление изменениями адресных данных – менее трудозатратным и более надежным в плане предупреждения появления ошибок.

Третий вариант – описать адресные данные с помощью внешнего классификатора. В случае с российскими адресами таким классификатором является Государственный адресный реестр (ГАР)⁵⁴, который является официальным источником сведений об адресах. В семантической модели в ветке НСИ необходимо создать набор классов, соответствующих типам объектов ФИАС – например, `ex:МуниципальноеОбразование`, `ex:Регион`, `ex:Район`, `ex:Улица`, `ex:Здание`, `ex:Помещение`, `ex:Машиноместо` и т. д. Эти классы должны быть снабжены свойствами, которые будут хранить значения полей базы данных ФИАС ГАР. Затем следует импортировать в семантическую модель базу данных ФИАС ГАР в объеме тех регионов, с адресами которых планируется работать, или полностью⁵⁵. В результате в семантической модели будет размещена структурированная информация адресного типа, на которую смогут ссылаться свойства бизнес-объектов:

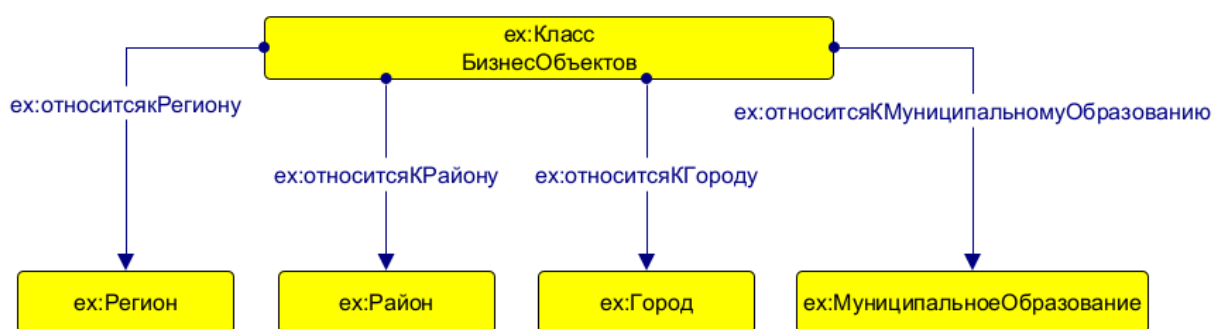


Рис. 16. Модель семантического описания адресной информации

⁵⁴ <https://fias.nalog.ru/>

⁵⁵ В приведённом примере объекты классификатора становятся частью разрабатываемой онтологии. Другой вариант работы с классификатором – реинжиниринг не онтологического ресурса в онтологию. При таком подходе нужно один раз создать онтологию на основе данных классификатора, а затем импортировать её в разрабатываемую модель. В таком случае у онтологии классификатора будет собственное пространство имён (например, `fiasgar`: вместо `ex`).

Классификатор ГАР обновляется регулярно дважды в неделю, поэтому он может использоваться в ситуациях, когда необходима высокая точность и детализация адресного описания объекта. Обратной стороной такой практики является необходимость частой актуализации справочника и необходимость хранить в онтологии большое число импортированных адресных объектов.

При использовании любого из описанных вариантов нужно предусмотреть возможность присвоения адресу определённого типа – зачастую у объектов, особенно агентов, имеется несколько адресов, которые по-разному задействованы в бизнес-процессах. Например, у физического лица может быть адрес регистрации, адрес фактического проживания и почтовый адрес, и это могут быть три разных адреса. Для случаев, когда адреса реализованы набором свойств, эта проблема может быть решена дополнением названия свойства типом адреса, например, `ex:улицаРегистрации` и `ex:улицаПроживания`, или `ex:адресПолныйРегистрации` и `ex:адресПолныйПочтовый`. Для случаев с адресами-объектами также добавляется отношение с указанием типа адреса, только оно связывает бизнес-объект с адресным объектом, например, вместо единого свойства `v:adr` можно использовать такие свойства как `ex:имеетАдресНахождения` или `ex:имеетАдресЮридический` (такой способ мы считаем предпочтительным). Альтернативный вариант – создавать адресные объекты для каждого типа адресов (например, класса `ex:ЮридическийАдрес` вместо `v:Address`) – менее предпочтителен, так как фактический адрес нахождения одного объекта может совпадать с юридическим адресом другого, то есть, свойства адресов «быть юридическим» или «быть адресом нахождения» не характеризуют сами адреса.

В результате размещения в семантической модели структурированной информации об адресах, появляется возможность нормализовать адресные данные информационных систем компании. Для этого необходимо использовать отдельный программный обработчик, который должен будет получать на вход исходную информацию об адресе (обычно она представлена в виде строковой записи), сопоставлять ее с элементами классов, соответствующих типам данных ГАР, и выдавать на выходе наборы данных, приведенные к стандартному виду. Этими наборами можно будет обогащать семантическую модель, заполняя адресные свойства бизнес-объектов.

Моделирование контрагентов

Работа практически любой компании предполагает взаимодействие с контрагентами – другими организациями (юридическими лицами) или физическими лицами, отношения с которыми регулируются с помощью договоров. С точки зрения семантической модели, статус контрагента может быть рассмотрен как особая роль, в рамках которой контрагент приобретает определённые характеристики. При этом до, во время и после исполнения этой роли контрагент является тем, чем он был до вступления в договорные отношения – организацией или физическим лицом (на практике обычно моделируются также Индивидуальные предприниматели, как особый подкласс физических лиц – их мы здесь пропустим для лаконичности примера). Один из вариантов моделирования этого представлен ниже:

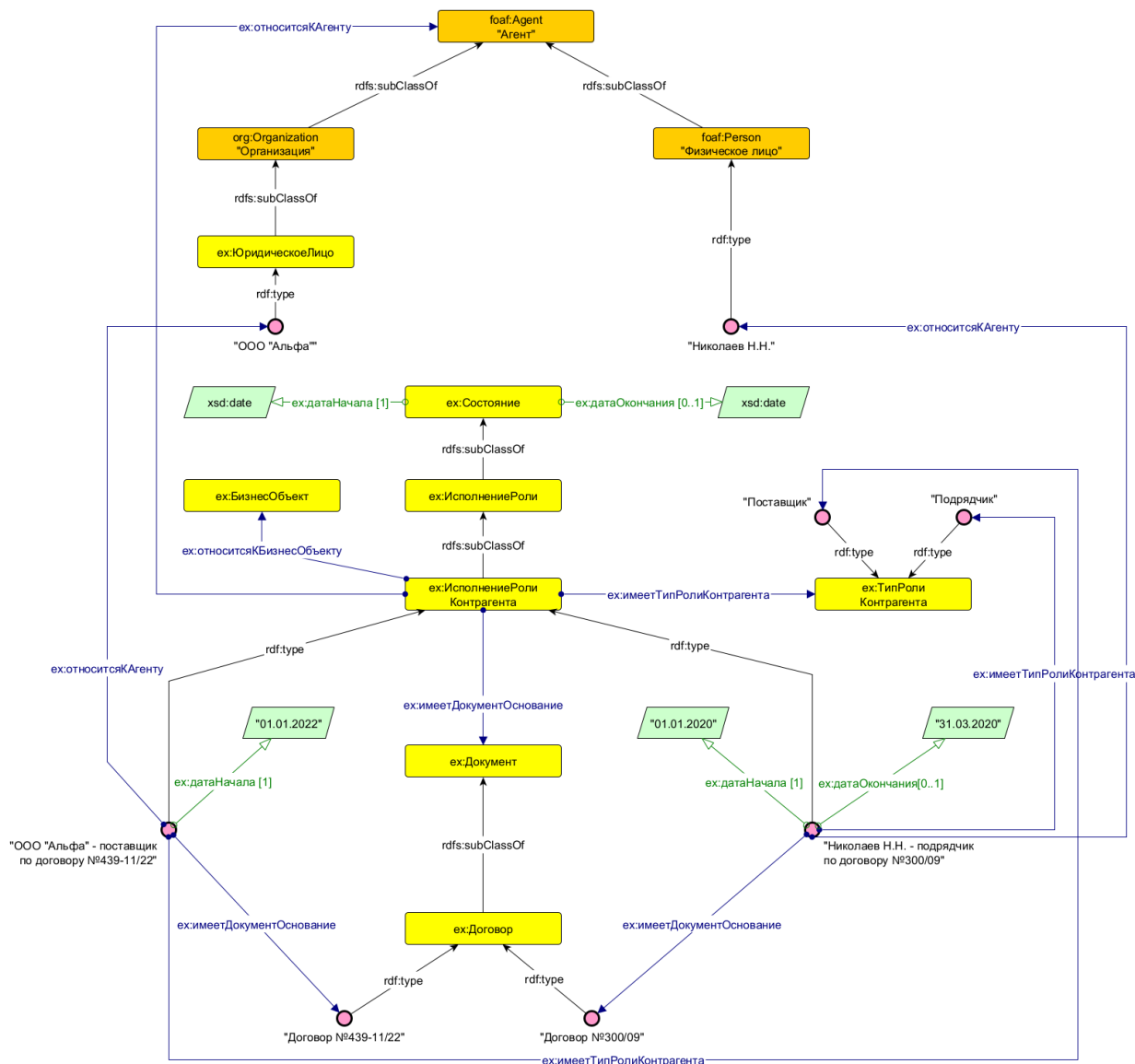


Рис. 17. Модель исполнения роли контрагента

Класс `ex:ИсполнениеРолиКонтрагента` используется для связывания контрагентов и тех бизнес-сущностей, которые участвуют в договорных отношениях на протяжении определенного периода времени. Объекты этого класса являются информационными агрегаторами, т. к. через обращение к ним можно получить набор связанной информации о целом ряде других объектов и о периоде актуальности сведений об этих объектах. Приведенная выше модель может содержать классы бизнес-объектов, которые выступают предметом отношений контрагентов – работы, поставки, оборудование, здания и сооружения и пр. На диаграмме эти классы условно обозначены как `ex:БизнесОбъект`. Класс `ex:ИсполнениеРолиКонтрагента` является подклассом `ex:ИсполнениеРоли`, который способен включать и другие подклассы, соответствующие различным ролям (например, `ex:ИсполнениеДолжности` и пр.). Объекты этого класса связываются свойством `ex:относитсяКАгенту` с теми сущностями – организациями или физическими лицами, объектами класса `Агент` онтологии FOAF – которые в данный момент исполняют роль контрагента.

Необходимым элементом семантической модели в данном случае может быть документация, описывающая отношения с контрагентами. Для представления этой информации следует использовать класс `ex:Документ` и его подклассы, типизирующие разнообразные виды

документов (договоры, акты, счета, расписки и пр.). Класс `ex:Документ` и его подклассы могут быть снабжены дополнительными набором свойств в зависимости от используемых в компании практик ведения документации (`ex:датаДокумента`, `ex:номерДоговора` и пр.).

Представленная модель контрагентов является масштабируемой и универсальной. Детализация модели может проводиться в части наборов классов бизнес-объектов, типов ролей контрагентов, типов документов и т. д.

Моделирование активов

Работа с информацией об активах (материально-технических ресурсах) в компаниях обычно ведется с использованием различных учетных информационных систем, необходимых для поддержки бизнес-процессов, таких как управление ремонтами. В то же время сильные стороны таких систем могут оказаться их слабыми местами, когда необходимо выйти за рамки подобных задач и получить, например, информацию о движении материально-технических ресурсов (единиц оборудования) в привязке к технологической схеме предприятия.

Семантическая модель позволяет не только структурировать информацию об оборудовании с нужной степенью детализации, но и описать, как используется каждая единица оборудования (ЕО) в определенный момент времени. Для этого предлагается использовать следующий паттерн:

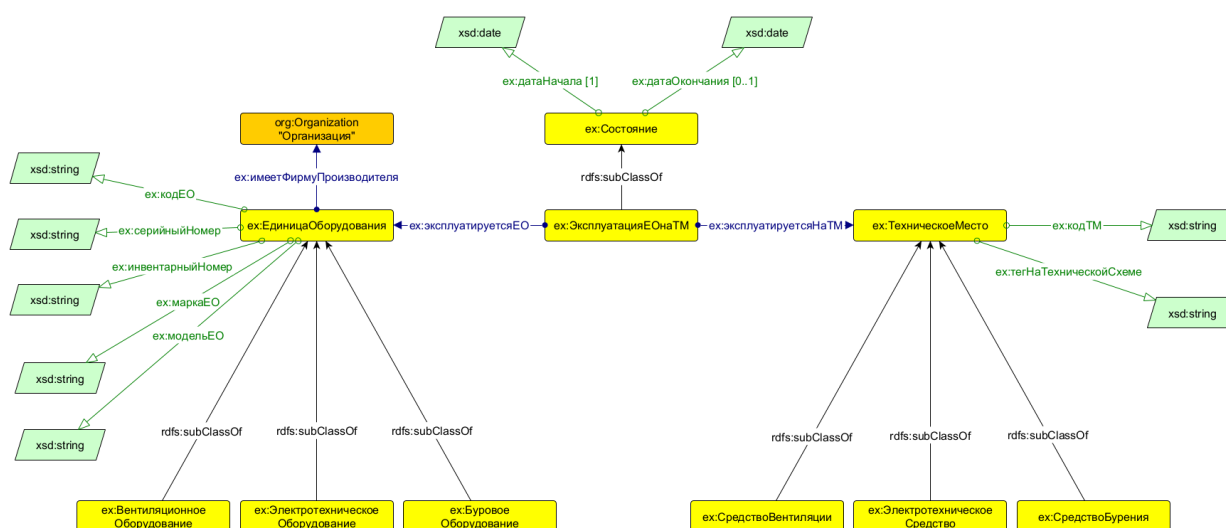


Рис. 18. Модель эксплуатации единиц оборудования на технических местах⁵⁶

В левой части диаграммы представлен вариант типологии оборудования. Индивидуальные объекты подклассов класса `ex:ЕдиницаОборудования` представляют уникальные материальные объекты, находящиеся на балансе предприятия и используемые в определенных технологических процессах.

В правой части диаграммы представлена типология технических мест (ТМ) – функциональных компонентов технологической схемы предприятия. Индивидуальные объекты подклассов класса `ex:ТехническоеМесто` представляют уникальные элементы этой технологической схемы.

Факт того, что конкретная единица оборудования эксплуатируется на определенном техническом месте на протяжении некоторого периода времени, описывается индивидуальным

⁵⁶ Перечисленные на диаграмме наборы свойств являются примерными, они могут быть скорректированы при построении конкретной семантической модели.

объектом класса `ex:ЭксплуатацияЕОнаТМ`. Эти объекты позволяют вести «историю» эксплуатации единиц оборудования в разрезе технических мест:

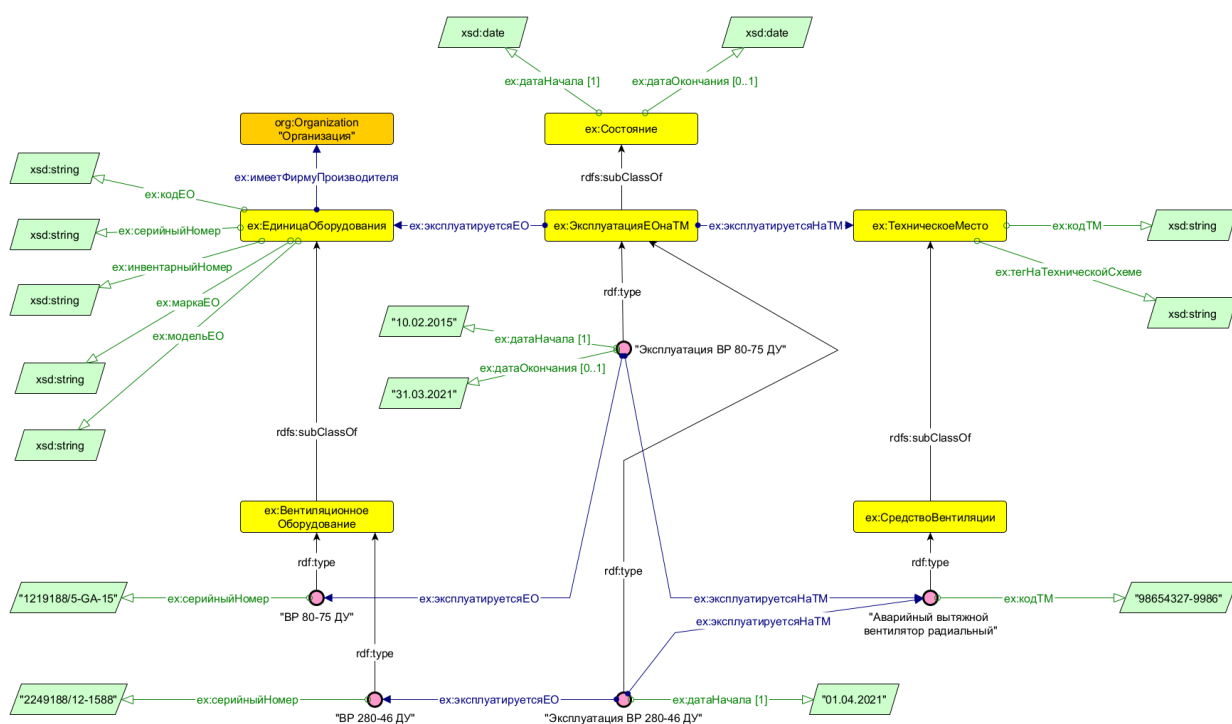


Рис. 19. История эксплуатации единиц оборудования на техническом месте

Сделав запрос к семантической модели на получение объектов класса `ex:ЭксплуатацияЕОнаТМ`, которые относятся к определенному объекту класса `ex:ТехническоеМесто`, можно будет получить объекты подкласса `ex:ЕдиницаОборудования`. Значения темпоральных свойств найденных объектов (`ex:датаНачала`, `ex:датаОкончания`) позволяют сделать вывод о том, какая ЕО эксплуатировалась и какая продолжает эксплуатироваться на данном ТМ на текущий момент.

Аналогичный подход может быть использован, если семантическая модель должна содержать описания других этапов жизненного цикла оборудования: монтаж, вывод в ремонт, замену и т. д. Достаточно создать в модели соответствующие подклассы класса `ex:Состояние` или `ex:Событие` и связать их с классами бизнес-объектов.

Инструменты для работы с информацией в семантической модели

До сих пор мы описывали работу с семантической моделью, ориентируясь на инструменты, обеспечивающие редактирование и управление ее структурой (TBox) – редактор онтологий, графовую базу данных, систему виртуализации данных. Однако существуют и другие программные инструменты, предназначенные для работы с моделью и позволяющие пополнять ABox. В данном разделе представлен их краткий обзор.

В пополнении семантической модели фактами (которое и является наполнением ABox), используются упомянутые в разделе «Принципы и приемы построения семантической модели» слои лексических данных и правил логического вывода. Напомним, что слой лексических данных используется для работы с символьным представлением объектов реального мира, обеспечивая в рамках модели создание глоссария терминов или словаря предметной области. Данный слой в качестве инструментов использует решения по обработке текстов на естественном языке. Слой правил логического вывода обеспечивает добавление новых фактов в модель на основе тех, которые уже в ней имеются. Работу данного слоя обеспечивают специализированные языки, такие как SHACL.

Управление терминологическим аппаратом

Описание набора понятий и их связей, необходимое для формирования набора классов и свойств онтологической модели, обеспечивает построение модели предметной области – семантического слоя бизнес-объектов. Как мы постарались показать в предыдущем разделе руководства, этот слой необходим для обработки информационных проекций тех объектов, процессов, зависимостей, с которыми имеют дело работники компании в своей деятельности.

В то же время в корпоративной практике часто используются информационные ресурсы, которые содержат *сведения о терминологических единицах*, организованные по принципу словарей (тезаурусов). Такие ресурсы также могут быть реализованы средствами семантического моделирования в виде набора классов, их свойств и объектов, однако прагматика у них иная – они предназначены для фиксации *знаний об используемом терминологическом, концептуальном аппарате*, а не информационной репрезентации конкретных бизнес-объектов.

Один из способов представления общих, концептуальных знаний опирается на стандарт **SKOS (Simple Knowledge Organization System)**⁵⁷, который основан на обширной методологии построения KOS-систем⁵⁸. Будучи сам по себе онтологией, SKOS предоставляет инструментарий для организации в семантической модели отдельного слоя данных, который содержит термины, описывающие понятия предметной области. При этом важно учитывать, что SKOS не предназначен для моделирования предметной области. Он выполняет референциальную функцию, т. е. с его помощью можно описать то, что люди знают о терминах и отношениях между ними.

Основными строительными элементами SKOS, которые используются для реализации этой функции, являются класс `skos:Concept`, его индивидуальные объекты – термины, а также свойства. Среди последних выделяют свойства для передачи наименований терминов (Labelling

⁵⁷ <https://www.w3.org/TR/swbp-skos-core-guide/>;
<https://www.w3.org/TR/skos-reference/>; <https://www.w3.org/TR/skos-primer/>

⁵⁸ См., например: <https://www.isko.org/index.php>.

Properties), для документирования терминов и их определений (Documentation Properties), для записи семантических отношений между терминами (Semantic Relationships)⁵⁹.

Наиболее востребованные свойства SKOS и некоторые примеры их использования приведены на следующей диаграмме:

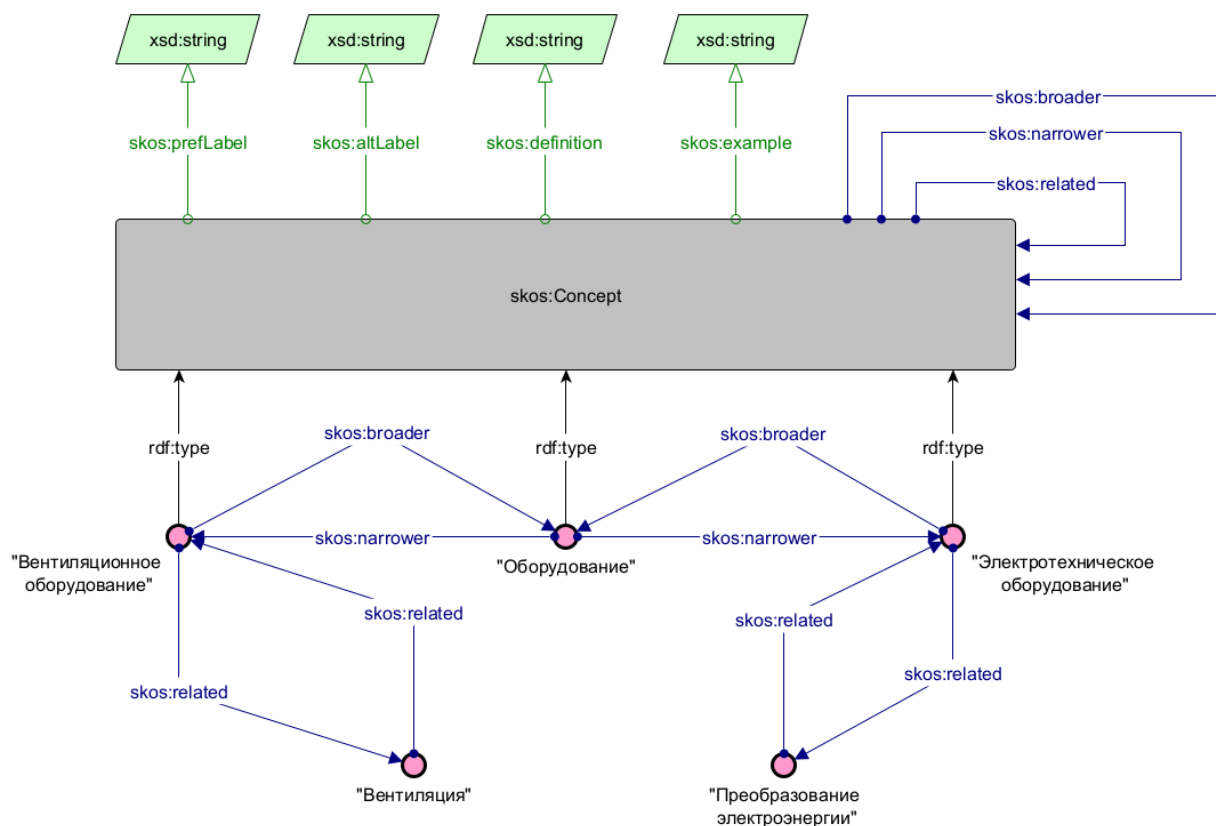


Рис. 20. Основные свойства класса `skos:Concept` и примеры терминов

Индивидуальные объекты класса `skos:Concept` – термины, которые могут иметь одно или несколько лексических обозначений: основное (`skos:prefLabel`) и неосновное/-ые (`skos:altLabel`), определение (`skos:definition`), пример употребления (`skos:example`) и другие свойства. Термины могут находиться друг по отношению к другу в различных смысловых отношениях: родовидовых (`skos:broader`, `skos:narrower`) и ассоциативной связности (`skos:related`).

С использованием SKOS можно реализовать программный продукт, позволяющий описать специализированный корпоративный словарь (глоссарий). Такой продукт позволяет создавать индивидуальные объекты класса `ex:Concept`, определять семантические отношения между ними, проводить их аннотирование и т. д. В целом глоссарий обеспечивает единое понятийное пространство компании, способствуя формированию структурированного и связного набора терминов, которые описывают составляющие и контекст ее производственной деятельности.

⁵⁹ Baker, T., Bechhofer, S., Isaac, A., Miles, A., Schreiber, G., Summers, E. (2013). Key choices in the design of Simple Knowledge Organization System (SKOS). *Journal of Web Semantics*, 20, 35–49. URL: <https://arxiv.org/abs/1302.1224>.

Также глоссарий предоставляет возможность связать термин с сущностями информационной модели предметной области, которые этот термин обозначает. Это позволяет пользователю работать не только с контентом и знаниями, но и с данными, содержащимися в онтологии⁶⁰.

Контроль логической целостности

Одной из задач создания технологий «семантической сети» (Semantic web) было обеспечение возможности формального представления в виде объектов онтологии и выполнения правил логического вывода, в том числе логических ограничений. Онтологическая модель для описания логических ограничений представлена в спецификации **Shapes Constraint Language (SHACL)**⁶¹.

Каждое правило логической целостности постулирует, что все объекты выбранного класса должны отвечать определенному *шаблону* (*shape*). Шаблон может содержать условия на наличие значений определенных свойств-ссылок на другие объекты, или на значения литеральных свойств. Приведем пример ограничения в условной нотации (?this – переменная, означающая проверяемый объект):

```
?this rdf:type ex:Оборудование.  
?this ex:имеетСертификат ex:СертификатОДопуске.  
?this ex:годПроизводства >= "2001"^^xsd:string.
```

Смысл правила заключается в том, что каждая единица оборудования, которая произведена не ранее 2001 года, должна иметь сертификат о допуске. В случае несоблюдения этих условий соответствующий объект класса ex:Оборудование считается некорректным.

Правила логической целостности могут применяться к уже созданным объектам семантической модели или к создаваемым впервые – в режиме реального времени, в момент записи объектов в хранилище. В платформе АрхиГраф правило может запрещать запись объекта, не соответствующего его условиям, или разрешать, но формировать особый объект класса sh:ValidationResult, описывающий выявленное нарушение.

Таким образом, использование правил логической целостности переводит высказывания, моделируемые в онтологии, из модальности «может быть» в модальность «должно быть». В сочетании с использованием ограничений на количество значений свойств (owl:minCardinality и owl:maxCardinality) это дает дополнительные возможности по управлению структурой и качеством данных, представленных в модели.

Правила SHACL Constraints удобнее всего создавать в специальном интерфейсе. Подробнее с приемами работы в конструкторе логических правил, разработанном компанией «ТриниДата», можно познакомиться в Руководстве пользователя Системы управления знаниями АрхиГраф.СУЗ⁶².

Поиск объектов-дубликатов

При наполнении информационной модели индивидуальными объектами нередки случаи, когда несколько информационных объектов описывают один и тот же реальный объект. Такие информационные объекты мы называем дубликатами.

⁶⁰ См. описание порядка работы с глоссарием, разработанным компанией «ТриниДата»: <https://trinidata.ru/files/ArchiGraphSUZUserGuide.pdf>.

⁶¹ <https://www.w3.org/TR/shacl/>

⁶² <https://trinidata.ru/files/ArchiGraphSUZUserGuide.pdf>

Для автоматизации выявления дубликатов могут использоваться логические правила **SHACL Rules** (правила логического вывода). Правило логического вывода в общем случае предполагает, что на основе определенных условий в отношении индивидуальных объектов установленного класса делается некоторый вывод. Например:

```
если
?this rdf:type ex:Оборудование.
?this ex:инвентарныйНомер ?number.
?equip rdf:type ex:Оборудование.
?equip ex:инвентарныйНомер ?number.
то
?this archigraph-mdm:duplicate ?equip.
```

Смысл правила заключается в том, что если существуют два объекта класса `ex:Оборудование` с одинаковым инвентарным номером, то они являются дубликатами. На самом деле это правило чуть сложнее – необходимо еще проверить, что значения переменных `?this` и `?equip` не одинаковы.

Правило работы с дубликатами может быть настроено таким образом, чтобы при создании в информационной модели нового объекта вначале проводилась проверка, не является ли он дубликатом существующего объекта. Если он оказывается дубликатом, к нему должно примениться определенное действие – запрет на создание или слияние с дублирующим объектом. Обработка дубликатов особенно полезна для случаев, когда информационная модель пополняется однотипными или содержательно похожими данными из различных корпоративных информационных систем.

Правила поиска объектов-дубликатов удобнее всего создавать в специальном интерфейсе. Подробнее с приемами работы в конструкторе правил работы с дубликатами, разработанном компанией «ТриниДата», можно познакомиться в Руководстве пользователя Системы управления знаниями *АрхиГраф.СУЗ*⁶³.

Правила вывода новой информации

Использование правил логического вывода SHACL Rules предоставляет еще одну возможность при работе с данными: автоматизированное создание новых индивидуальных объектов модели или дополнение значений свойств существующих объектов.

Правила логического вывода полезны, если нужно автоматически обогащать содержание модели на основе динамически изменяющихся данных. Например, если требуется выразить, что в результате проведения не ранее 01.01.2020 успешной поверки измерительного прибора его можно продолжать эксплуатировать, необходимо создать логическое правило следующей структуры:

```
если
?this rdf:type ex:Поверка.
?equip rdf:type ex:ИзмерительныйПрибор.
?this ex:проведенаДляПрибора ?equip.
?this ex:имеетРезультат ex:Успешно.
?this ex:дата ?date.
FILTER (?date >= "01-01-2020"^^xsd:date).
```

⁶³ <https://trinidata.ru/files/ArchiGraphSUZUserGuide.pdf>

то

```
?new rdf:type ex:ДопускКЭксплуатации.  
?new ex:относитсяКИзмерительномуПрибору ex:ИзмерительныйПрибор.  
?new ex:дата ?date.
```

В результате успешного срабатывания правила в классе `ex:ДопускКЭксплуатации` создается индивидуальный объект (обозначенный переменной `?new`), который связан с конкретным измерительным прибором, успешно прошедшим поверку в определенный день, с указанием даты поверки этого прибора. В результате, запросив объекты класса `ex:ДопускКЭксплуатации`, можно получить информацию о фактах допуска в эксплуатацию всех приборов, прошедших поверку.

Правила логического вывода удобнее всего создавать в специальном интерфейсе. Подробнее с приемами работы в конструкторе логических правил, разработанном компанией «ТриниДата», можно познакомиться в Руководстве пользователя Системы управления знаниями Архиграф.СУЗ⁶⁴.

⁶⁴ <https://trinidata.ru/files/ArchiGraphSUZUserGuide.pdf>

Организация работы с семантической моделью

В процессе создания семантической модели, а также её последующего развития необходима организационная поддержка. В неё входят следующие подпроцессы:

- оценка: созданная семантическая модель должна соответствовать изначально заявленной цели, а также не нарушать принципы построения формальных онтологий;
- управление изменениями: напомним, что онтология создаётся итеративно и предусматривает пополнение новыми знаниями на протяжении всего её жизненного цикла – данный процесс должен осуществляться контролируемо, а изменения фиксироваться;
- документирование онтологии: описание принятых при разработке дизайн-решений, создание инструкций по использованию, поддержка актуальности метаданных.

Все описанные процессы идут параллельно с разработкой онтологии: проверять соответствие онтологии требованиям, контролировать изменения и фиксировать принимаемые решения нужно в процессе разработки. Это обеспечивается командой поддержки онтологии при участии бизнес-пользователей и экспертов предметной области, взаимодействие с которыми является отдельной задачей. С роли различных членов команды мы и начнём описание организации работы с семантической моделью, а затем затронем непосредственно процессы поддержки.

Команда и заинтересованные лица

Работа с семантической моделью в корпоративных условиях предполагает участие нескольких специалистов (или даже их групп) разных типов: администратор информационной модели, эксперт предметной области, дата-стюард (эксперт по качеству данных), бизнес-пользователь. Хорошая практика – задействовать на этих ролях заинтересованных лиц, выявленных на этапе подготовки к построению модели. Кратко рассмотрим, в чем состоит специфика работы каждого.

Администратор информационной модели – это специалист, в наибольшей степени осведомленный о структуре и наполнении семантической модели, логике ее построения и принятых архитектурных решениях. Данный специалист либо сам является инженером по знаниям, либо работает во взаимодействии с инженером по знаниям. Основной функционал администратора:

- обработка новых заявок от заинтересованных лиц, касающихся дополнения/изменения модели, оценка их целесообразности, при необходимости формирование дорожной карты изменений;
- подготовка к принятию решения о внесении нового свойства/класса в семантическую модель, что предполагает поиск и/или формулирование определения, описывающего смысл новой сущности в контексте семантики предметной области;
- подготовка к удалению элемента семантической модели, что требует оценки целесообразности такого удаления, а также того, что в результате удаления целостность семантической модели не будет нарушаться;
- инициирование рефакторинга семантической модели, включая подготовку обоснования его необходимости, плана рефакторинга, оценки корректировок потоков данных, которые обрабатываются с использованием семантической модели, и/или их автоматизированного преобразования;
- работа в редакторе онтологии;
- документирование процесса изменения семантической модели.

Эксперт предметной области – специалист-консультант, ответственный за прикладное содержание онтологической модели, обеспечивает её соответствие знаниям предметной области, благодаря чему обеспечивается соответствие модели объективной реальности. Он выступает в качестве «точки правды» при принятии решений о том, что определенный класс или свойство онтологической модели должен быть включен/обновлен/удален.

Работу экспертов целесообразно организовать с учетом специфики конкретной компании и принятых в ней практик внедрения корпоративных информационных систем. В качестве наиболее полезных методик могут быть названы экспертные сессии, проблемные интервью, фокус-группы, мозговые штурмы и пр. Не менее важным способом извлечения информации, которая должна использоваться при создании информационной модели, является работа с документами – как корпоративного, так и отраслевого уровней.

Дата-стюард – работник, который осуществляет мониторинг качества данных и поддерживает наборы НСИ. Основной функционал дата-стюарда:

- работа с заявками всех заинтересованных лиц по корректировкам содержания информационной модели, включая классификаторы и справочники. Дата-стюард при необходимости может запросить консультацию у эксперта НСИ. Для каталогов НСИ могут быть определены их владельцы, с которыми дата-стюард должен согласовать изменение наборов данных в этих каталогах;

- поиск необходимых данных по запросам бизнес-пользователей;
- если от заинтересованных лиц поступает заявка на удаление данных, дата-стюард оценивает целесообразность такого удаления, в том числе наличие ссылок на предлагаемый к удалению объект;

- поиск и устранение дублирования данных в семантической модели;
- документирование процесса подачи и согласования заявок на изменение наборов данных и НСИ.

Функционал дата-стюарда похож на функционал администратора, основное отличие между ними можно провести по частям семантической модели, с которой они работают: администратор отвечает в большей мере за структуру модели (то есть, TBox), дата-стюард – за её наполнение (то есть, за ABox).

Бизнес-пользователь – это любой работник компании, имеющий доступ к информации, которая обрабатывается с использованием корпоративных информационных систем, обращающихся к семантической модели. Так как бизнес-пользователи зачастую являются конечными пользователями систем, в основе которых лежат семантические модели, их удобство и потребности должны учитываться при разработке.

Координацию работы администратора информационной модели, экспертов, дата-стюарда в масштабах компании целесообразно проводить с помощью **совета по корпоративной модели данных** – организационной единицы, к полномочиям которой могут быть отнесены следующие вопросы:

- принятие решений по внесению изменений в семантическую модель, включая ее рефакторинг;

- контроль соответствия содержания модели заявленным целям разработки модели, включая принятие решений об их изменении;

- принятие решений по корректировкам каталогов НСИ;

- реализация организационных процедур, предназначенных для поддержания качества данных (исключение дублирования, контроль непротиворечивости и логической целостности

семантической модели, разработка механизмов прослеживаемости происхождения данных, обеспечение доступности данных для бизнес-пользователей);

– участие в экспертизе технических заданий по разработке и внедрению информационных систем в компании, которые должны учитывать структуру и содержание семантической модели.

Оценка онтологии

Существует множество подходов к оценке онтологии, для более полного погружения в тему мы рекомендуем обратиться к соответствующей литературе. В рамках данного руководства мы опишем опыт оценки онтологий в условиях промышленной разработки и эксплуатации. Такая оценка включает в себя проверку технической реализации (иногда её называют верификацией) и соответствие содержания модели изначально заявленному (валидация). Можно сказать, что верификация сопоставляет модель и её спецификацию, а валидация – модель и пользовательские требования⁶⁵. **Верификация** может проводиться на нескольких уровнях⁶⁶ – архитектуры, синтаксиса и контента:

- на уровне архитектуры обеспечивается корректное взаимодействие онтологии со средой, в которой она функционирует;
- на уровне синтаксиса проверяется его правильность;
- на уровне контента – согласованность и полнота аксиом в онтологии.

На уровне синтаксиса и контента верификация может проводиться автоматически с помощью таких сервисов как OOPS! (OntOlogy Pitfall Scanner!)⁶⁷. Для такой оценки онтологии нужно загрузить её исходный код в выбранный парсер, который вернёт список ошибок. На рисунке 21 показан пример результата работы парсера OOPS!. Видно, что синтаксических ошибок не найдено, а одна из найденных ошибок (P41) касается метаданных, так как она указывает на отсутствие лицензии, т. е. информации о возможном использовании онтологии. Остальные ошибки относятся к уровню контента и включают:

- P04: «оторванные» от основного графа классы или экземпляры – в загруженной онтологии есть 4 элемента, не связанные с какими-либо другими элементами онтологии;
- P10: отсутствие аксиом дизъюнктивности;
- P11: отсутствие аксиом об области применения и диапазоне значений свойств;
- P13: для трёх свойств в онтологии не определены обратные свойства – эту ошибку данный парсер выдаёт для всех свойств, кроме явно определенных как симметричные.

Часть из найденных парсером ошибок можно исправить автоматически, например, упомянутый нами редактор аксиом Protégé способен добавлять аксиомы дизъюнктивности. Остальные ошибки нуждаются в ручном исправлении.

⁶⁵ Зачастую проверку правильности кода называют валидацией, а соответствие онтологии требованиям – оценкой. Мы в данном руководстве используем термины валидация, верификация и экспертиза как части процесса оценки чтобы было однозначно понятно, о чём идёт речь.

⁶⁶ Gómez-Pérez, A. (1996). Towards a framework to verify knowledge sharing technology. Expert Systems with Applications, 11(4 SPEC. ISS.), 519–529.

⁶⁷ <https://oops.linkeddata.es/>

[Expand All] | [Collapse All]

Results for P04: Creating unconnected ontology elements.	4 cases Minor 🟡
Results for P10: Missing disjointness.	ontology* Important 🟠
Results for P11: Missing domain or range in properties.	3 cases Important 🟠
Object and/or datatype properties without domain or range (or none of them) are included in the ontology.	
<ul style="list-style-type: none">• This pitfall appears in the following elements:<ul style="list-style-type: none">> http://webprotege.stanford.edu/R99ItWickPV3VO0oJIR7tyT> http://webprotege.stanford.edu/RBslffVzVhRs8OCBxhyJW0V> http://webprotege.stanford.edu/Rl1oTdbuieZYsn4Cb00YMP• Tip: Solving this pitfall may lead to new results for other pitfalls and suggestions. We encourage you to solve all cases when needed and see what else you can get from OOPS!	
Results for P13: Inverse relationships not explicitly declared.	3 cases Minor 🟡
Results for P41: No license declared.	ontology* Important 🟠

Рис. 21. Пример технической оценки онтологии парсером OOPS!

В платформе АрхиГраф оценка онтологий производится на основании метрик. Кроме этого, редактор онтологий АрхиГраф.Мир обеспечивает контроль ряда ошибок ещё на этапе разработки: так, синтаксические ошибки исключены благодаря тому, что пользователи системы не взаимодействуют напрямую с кодом онтологии. Также, в редакторе невозможно создать оторванный от основного графа элемент, а при попытке создать свойство без области определений и диапазона значений редактор подскажет пользователю, что их необходимо добавить. Частично оценка контента онтологии осуществляется внешними по отношению к ней инструментами, например, поиск дубликатов и контроль логической целостности, описанные в разделе «Инструменты для работы с информацией в семантической модели». Можно использовать и инструмент OOPS! – для этого нужно выгрузить онтологию из АрхиГраф.Мир в формате OWL/XML.

Валидация онтологии должна гарантировать то, что онтология действительно содержит ту модель, которую она должна представлять: её соответствие функциональным и нефункциональным требованиям, описанным на этапе подготовки к созданию модели (см. Шаг 1 в разделе «Как создать семантическую модель?»). Так, если в нефункциональных требованиях к онтологии было указано, что она должна включать информацию ГОСТов о моделируемых изделиях, то такая информация должна фактически содержаться в разработанной модели. То же верно и для языка реализации, системы именования и остальных требований. Отдельно проверяются компетентностные вопросы: вопросы, сформулированные перед началом работы на естественном языке, на этом этапе выражаются формальным языком и задаются онтологии – модель должна вернуть корректный ответ.

Проверка онтологии включает её экспертизу – возможность её использования для удовлетворения потребностей пользователя и соответствие модели состоянию дел в предметной области. Оценка возможности использования онтологии осуществляется бизнес-пользователями и основывается на описании цели и сценариях предполагаемого использования, сформулированных на этапе подготовки. Кроме этого, должна быть проведена оценка соответствия информации в онтологии состоянию дел в моделируемой предметной области – для этого привлекаются специалисты-консультанты.

Управление изменениями

Любой элемент семантической модели (класс, свойство, индивидуальный объект) обладает характеристиками, которые могут менять свои значения в ходе ее разработки и рефакторинга. Для того, чтобы иметь доступ к истории их изменения, необходимы программные решения, которые

обеспечат редактирование элементов семантической модели и хранение метаданных об изменениях, в частности времени и даты каждого изменения и его автора. Контроль таких метаданных осуществляется по-разному для структуры модели (на уровне TBox, её классов и свойств) и конкретных фактов, внесённых в неё (ABox, уровень экземпляров классов и значений свойств).

Изменение фактов относится к данным, загруженным в семантическую модель. Такие изменения должны производиться в тот момент, когда изменились соответствующие данные, их контроль может (и должен) осуществляться средствами самой модели. Например, если на склад прибыла партия красок «MINGIO Soft Pastel for Artists», то у соответствующего экземпляра класса `ex:Пастель` должно измениться значение литерального свойства `ex:остатокДляПродажи` (см. рисунок 9).

Если необходимо сохранение истории изменений, произошедших с объектом, то в модель можно внести классы, формирующие «журнал состояний». Пример такого подхода описан в разделе «Моделирование темпоральных характеристик и отношений» – экземпляры класса `ex:ЗаменаОборудования` (см. рисунок 12) фиксируют события, произошедшие с единицами оборудования. Благодаря наличию в модели факта «Замена единицы оборудования 1 на единицу оборудования 2» сохранена информация об изменениях статуса объектов «Единица оборудования 1» и «Единица оборудования 2».

Перед изменением фактов должна проводиться оценка целесообразности предполагаемых изменений: удаление экземпляров может производиться только после проверки того, что на данный элемент нет ссылок от других сущностей; добавление экземпляров и значений свойств должно осуществляться только из актуальных источников; и т. д.

Изменение структуры модели сложнее обеспечить внутренними средствами модели. В нотации RDF и языке OWL для этого предусмотрен механизм *реификации* (*reification*), позволяющий описывать триплеты с помощью других триплетов, однако такой подход не всегда применим в практике промышленной разработки – «промежуточные» триплеты, содержащие необходимые метаданные, неоправданно усложняют запросы к модели. Поэтому в разрабатываемых компанией «ТриниДата» продуктах контроль изменений структуры онтологии осуществляется внешними по отношению к онтологии средствами.

Изменение структуры модели требует тщательной оценки целесообразности. Так, перед удалением какого-либо элемента структуры семантической модели необходимо проверить его связи с другими элементами модели: например, предлагаемый к удалению класс может быть диапазоном значений или областью определения какого-то свойства. А для корректного добавления новых классов и свойств необходимо:

- формулирование определения для вносимой сущности: возможно использование справочных источников, принятых в данной отрасли и компании – стандартов, справочников, нормативных требований и пр., желательна также консультация с экспертом предметной области;
- оценка прикладных задач, для решения которых предлагается вводить новое свойство/класс в семантическую модель, в том числе определение точки зрения, которой соответствует предлагаемая к внесению в модель сущность⁶⁸;

⁶⁸ Вопрос о работе с несколькими точками зрения средствами семантических моделей рассматривается, например, здесь: Gorshkov, S., Kralin, S., Miroshnichenko, M. (2016, September). Multi-viewpoint ontologies for decision-making Support. In International Conference on Knowledge Engineering and the Semantic Web (pp. 3-17). Springer, Cham. URL: https://link.springer.com/chapter/10.1007/978-3-319-45880-9_1.

– поиск и оценка возможных противоречий или избыточности, которые могут быть вызваны дополнением семантической модели.

Платформа АрхиГраф, в состав которой входит редактор АрхиГраф.Мир и средство виртуализации данных АрхиГраф.MDM, позволяет в полном объеме работать с указанными метаданными. Общий принцип работы этих компонентов в составе платформы состоит в том, что семантическая модель разрабатывается средствами редактора онтологий, и вся история изменений модели и данных фиксируется и хранится вне онтологической модели средствами платформы. Платформа «скрывает» от пользователя темпоральный аспект данных, позволяя ему сосредоточиться на моделировании мгновенного состояния мира, но вместе с тем работает со множеством таких состояний. Как в редакторе АрхиГраф.Мир, так и при обращении к программному интерфейсу платформы можно установить момент времени, и работать с состоянием данных на этот момент – в прошлом или в будущем.

Для контроля версий объектов и триплетов в платформе предусмотрен встроенный механизм сохранения истории изменений – для каждого элемента онтологии (класса, свойства или индивидуального объекта) сохраняется время его изменения, автор и содержание изменения. Благодаря этому можно путем запроса к API платформы получить:

- информацию о наборе значений свойств каждого объекта на любой момент времени (например, в момент T1 товар был на складе, а в момент T2 он уже отсутствовал);
- информацию о существовании объектов в определенный момент времени (каждый объект онтологии был создан в какой-то момент, а также может быть удален с определенного момента);
- информацию о состоянии TBox, т. е. набор классов и свойств, определенный в онтологии на заданный момент времени;
- всю историю изменений какого-либо элемента онтологии за заданный период.

Редактор онтологий АрхиГраф.Мир поддерживает работу с перечисленными возможностями API платформы, что позволяет в интерфейсе редактора работать с набором объектов и значениями их свойств по состоянию на любой момент времени. Вся темпоральная информация хранится вне онтологии, в специальной базе метаданных платформы.

Документирование

Документирование семантической модели является столь же необходимым процессом, как и документирование любого программного продукта. Однако в случае с описанием данных онтологической природы стоит учитывать возможность семантической модели содержать в себе в том числе и метаинформацию о входящих в неё сущностях, то есть, документирование модели частично проводится средствами самой модели. Этого, однако, недостаточно для эффективной работы со сложными моделями – такие модели должны сопровождаться диаграммами, облегчающими понимание функциональных связей сущностей.

Документирование средствами семантической модели обеспечивается аннотационными свойствами. Семантическая модель позволяет проводить текстовое аннотирование любого ее элемента (класса, свойства, индивидуального объекта) с помощью стандартного свойства `rdfs:comment`. Это «самодокументирование» считается хорошей практикой при создании модели, т. к. оно необходимо для фиксации смыслов, какими наделяются элементы модели ее разработчиками. В этом отношении семантическая модель хороша своей самодостаточностью, если необходимо разобраться, с какой целью был создан и каково функциональное назначение того или иного ее элемента.

Для информации об онтологии в целом существуют модели метаданных, которые можно импортировать в разрабатываемую онтологию. Пример такой модели – набор метаданных Dublin Core⁶⁹, содержащий информацию для описания любого информационного объекта и включающий такие свойства как:

- `dc:created`, связывающее информационный ресурс с датой его создания;
- `dc:contributor`, связывающее информационный ресурс с сущностями, которые внесли вклад в его создание или изменение, здесь могут быть зафиксированы участники команды разработки семантической модели;
- `dc:description` для описания ресурса, например, в нём могут быть описаны цели создания семантической модели и даны ссылки на внешнюю документацию.

Внешняя документация семантической модели представлена в первую очередь материалами, содержащими информацию о цели, дизайн-решениях и состоянии модели. Важный элемент такой документации – диаграммы. Подготовка диаграмм с описанием структуры модели является чрезвычайно важным способом фиксации текущего и/или перспективного ее состояния. Это объясняется тем, что семантическая модель в процессе разработки постоянно трансформируется и расширяется. Диаграммы здесь выполняют несколько функций: (1) помогают визуализировать те решения, которые реализованы в модели и признаны стабильными; (2) помогают разрабатывать новые фрагменты семантической модели и проверять, как они «стыкуются» с имеющимися; (3) облегчают понимание структуры модели применительно к конкретным задачам – для их выполнения сущности в модели взаимодействуют за пределами иерархических отношений, эта информация содержится в самой модели, но на диаграмме понять её проще; (4) позволяют освежить в памяти отдельные фрагменты модели – хотя бы потому, что удержать в голове всю модель человеку чрезвычайно сложно.

В качестве инструмента для создания диаграмм семантической модели можно использовать любой редактор, позволяющий работать с блок-схемами и/или ER-диаграммами. На данный момент не существует жестко закрепленного стандарта (графической нотации), в каком требовалось бы оформлять диаграммы с описанием семантической модели. Мы рекомендуем обратить внимание на редактор yEd⁷⁰, для которого разработана дополнительная палитра инструментов Grafoo⁷¹, позволяющая описать элементы семантической модели в соответствии со стандартами OWL⁷².

⁶⁹ <https://www.dublincore.org/specifications/dublin-core/dcmi-terms/>

⁷⁰ <https://www.yworks.com/editors>

⁷¹ <https://essepuntato.it/graffoo/>

⁷² <https://essepuntato.it/graffoo/specification/>

Заключение

На этом мы останавливаем описание основных принципов и приемов подготовки семантических моделей промышленного назначения, но не потому, что больше не о чем написать. Мы рассказали о самом необходимом – о том, без чего онтологическая модель не может быть реализована для применения в решении задач бизнеса.

Безусловно, много что осталось за кадром. Это и вопросы рефакторинга семантических моделей и проблемы их масштабирования, и работа распределенных команд специалистов по развитию модели, и подходы к определению пространств имен для выделения различных функционально-семантических «блоков» внутри онтологии, и особенности архитектуры программных продуктов для работы с семантической моделью и др.

Однако авторы надеются, что предложенное вниманию читателей руководство если и не станет настольной книгой IT-директора каждой компании, то хотя бы поможет сориентироваться в тех практических моментах, на которые стоит обращать внимание при проектировании, реализации и промышленной эксплуатации решений, использующих онтологии в качестве информационного ядра. А если кто-то из заказчиков станет более отчетливо себе представлять, как можно использовать онтологии при решении конкретных бизнес-задач, мы будем считать свою задачу в существенной мере выполненной.

Авторы открыты для общения и будут благодарны за вопросы, предложения и пожелания.

Электронная почта для обратной связи: info@trinidata.ru.