УДК 004.682

АВТОМАТИЗАЦИЯ СИНТЕЗА СОСТАВНЫХ ОНТОЛОГИЧЕСКИХ ПАТТЕРНОВ СОДЕРЖАНИЯ

П.А. Ломов

Институт информатики и математического моделирования технологических процессов Кольского научного центра Российской академии наук, Апатиты, Мурманская область, Россия
lomov@iimm.ru

Аннотация

Применение онтологических паттернов проектирования становится распространенным подходом в онтологическом инжиниринге. Онтологические паттерны являются формализацией проверенных решений, которые могут быть повторно использованы при разработке онтологий. В данной статье основное внимание сосредоточено на одной разновидности онтологических паттернов – онтологических паттернах содержания, которые представляют собой небольшие фрагменты онтологий, формализующие обобщённые ситуации предметной области (например, участие в событии, исполнении роли, наличие частей у объекта и др.). Паттерны содержания используются в качестве строительных блоков при разработке онтологии. В таком случае они, как правило, могут быть расширены, специализированы, объединены разработчиком для получения составного паттерна содержания, который предоставит более сложный онтологический фрагмент, позволяющий обеспечить представление знаний о некотором объекте предметной области с необходимой степенью детализации. Однако выполнения таких композиций паттернов содержания часто не сводится к простому объединению соответствующих им онтологических фрагментов, а требуют их предварительной модификации и последующего связывания их элементов особым образом. Это может потребовать от разработчика наличия знания семантики того или иного паттерна, используемого в композиции, а также навыков онтологического инжиниринга и тем самым усложнить применение паттернов. В данной работе рассматривается проблема автоматизации подбора подходящих паттернов содержания на основе преопределённых отношений между ними и последующего синтеза на их основе составного паттерна в соответствии с требования задачи разработчика онтологии. Рассматривается пример синтеза составного онтологического паттерна содержания с использованием предложенной процедуры.

Ключевые слова: *онтология, онтологические паттерны, онтологический инжиниринг, автоматизация синтеза.*

Цитирование: *Ломов, П.А.* Автоматизация синтеза составных онтологических паттернов содержания / П.А. Ломов // Онтология проектирования. – 2016. – Т. 6, №2(20). - С. 162-172. – DOI: 10.18287/2223-9537-2016-6-2-162-172.

Введение

Несмотря на то, что на данный момент существует множество программных средств [1-4] и технологий онтологического инжиниринга [5-8], разработка онтологий остаётся сложной задачей. В связи с этим повторное использование (reuse) эффективных практик решения классов задач онтологического моделирования является многообещающим подходом. Одной из его реализаций является использование онтологических паттернов проектирования (Ontology Design Patterns, ODP). Основные преимущества и недостатки применения паттернов при разработке и последующем сопровождении онтологий были рассмотрены автором в работе [9].

В соответствии с [10] ODP представляют эффективные решения типовых проблем онтологического моделирования. В рамках ODP для каждого такого решения, помимо его описания, даётся обобщённое определение проблемной ситуации, в которой необходимо его применять, а также представляются последствия использования данного решения. Это позволяет разработчикам не тратить время на поиск и тестирование разных вариантов решения проблемы моделирования, а выбрать, исходя из поставленной перед ними задачи, и реализовать вариант, предлагаемый тем или иным ODP. Таким образом, применение ODP позволяет сократить затраты труда и времени, а также гарантировать качество полученной в итоге онтологии за счёт применения проверенных решений в процессе её разработки.

Существует несколько типов ODP, каждый из которых ориентирован на решение различных видов задач онтологического моделирования. В данной статье основное внимание сосредоточено на проблемах использования онтологических паттернов содержания (Content ontology Design Patterns, CDP), которые представляют возможные варианты представления знаний предметной области (ПрО) в виде наборов классов онтологий, связанных отношениями, то есть фрагментов онтологий. Для каждого CDP задаётся обобщённое определение ситуации (General Use Case, GUC), в которой необходимо его применять. Например, участие в событии или исполнении роли, наличие частей у объекта и другие. Исходя из GUC, для паттернов содержания также задаются наборы квалификационных вопросов (Competency questions). Каждый из таких вопросов может быть преобразован в запрос к онтологии, ответ на который гарантированно будет получен в случае реализации соответствующего паттерна.

Общая схема применения ODP, и в частности паттернов содержания, представлена в технологии экстремального проектирования онтологий (eXtreme Design methodology, XD). Она подразумевает поиск разработчиком необходимого паттерна содержания в специализированном каталоге по ключевым словам и квалификационным вопросам, импортирование соответствующего ему онтологического фрагмента в разрабатываемую онтологию, последующая его конкретизация путём определения наследников его элементов и тестирование полученного фрагмента онтологии. Разумеется, разработчик может не обнаружить паттерна, полностью отвечающего требованиям конкретной задачи. В таком случае ему может потребоваться комбинация нескольких из них. Например, определение действия, порядок и состав этапов которого зависит от его участников и требует комбинирования паттернов содержания «Участие» (Participation), «Часть» (PartOf), «Последовательность» (Sequence) и «Ситуация» (Situation).

При этом заметим, что использование комбинации нескольких паттернов в ряде случаев не сводится к простому объединению соответствующих им фрагментов онтологий, а требует их перестройки и связывания их отношениями особым образом. Это предъявляет к разработчику дополнительные требования наличия специализированных знаний и навыков онтологического инжиниринга. Тем самым снижается польза от использования технологии паттернов. Представление же в каталоге всего множества комбинаций нецелесообразно в виду их большого числа.

Для решения этой проблемы в данной работе рассматриваются специфические вопросы комбинирования паттернов содержания и предлагается технология генерации составного паттерна по требованиям пользователя. Её применение позволит автоматизировать подбор комбинаций паттернов и компоновку соответствующих им онтологических фрагментов с образованием в итоге фрагмента онтологии составного паттерна содержания, позволяющего обеспечить должное представление знаний ПрО.

1 Общая процедура синтеза составного паттерна содержания

В общем случае процедура синтеза составного паттерна содержания - X состоит из следующих шагов:

1) пользователь выбирает начальный паттерн. Соответствующий ему фрагмент онтологии становится основой для формирования фрагмента для синтезируемого паттерна X;
2) среди понятий, представленных классами в онтологическом фрагменте паттерна X, пользователь выбирает то, описание которого он хотел бы расширить другими паттернами;
3) исходя из выбранного понятия и паттерна, к которому оно относится, формируется и представляется пользователю набор предлагаемых паттернов (Set of offered patterns, SOP);
4) пользователь выбирает желаемый паттерн из предложенных для добавления в синтезируемый паттерн X;
5) по желанию пользователя осуществляется повтор шагов 2) - 4) или завершение процедуры переходом к шагу 6);
6) пользователь осуществляет необходимую правку полученного паттерна X.

Заметим, что первый шаг процедуры требует того, чтобы пользователь был знаком с паттернами содержания для выбора из них подходящего для решения поставленной задачи моделирования. В случае затруднений с таким выбором по умолчанию предлагается паттерн «Виды сущностей» (Type of entities). Он имеет простую структуру и определяет с помощью классов своего фрагмента онтологии основные типы сущностей ПрО, такие как «Качество» (Quality), «Объект» (Object), «Событие» (Event), «Абстрактный объект» (Abstract). Особенностью его использования при синтезе является возможность для пользователя добавлять отдельные классы этого паттерна. Например, пользователь может добавить в синтезируемый паттерн X только класс «Объект» данного паттерна и далее расширять этот класс другими паттернами.

При формировании набора предлагаемых паттернов на каждой итерации учитываются следующие отношения [11, 12] между паттернами:

- Отношение «**специализирует**» (specialization) - Паттерн *P2* специализирует другой паттерн *P1*, когда по крайней мере один из классов или одно из свойств в *P2* является подклассом или подсвойством некоторого класса или свойства *P1*. При этом в остальном *P1* и *P2* идентичны.
- Отношение «**имеет часть**» (has-components) проявляется в следующих случаях:
  - Паттерн *P2* расширяет (extends) паттерн *P1*, т.е. когда *P2* содержит *P1* целиком и добавляет новые классы или свойства;
  - Паттерн *P3* интегрирует паттерны (integrates) *P2* и *P1*, т.е. когда *P3* включает *P1* и *P2*;
  - Паттерн *P3* объединяет (merges) паттерны *P2* и *P1*, т.е. когда *P3* включает *P1* и *P2* и задает хотя бы одно отношение между двумя классами из *P1* и *P2*.
- Отношение «**соотносится**» (related) между паттернами *P1* и *P2* устанавливается в случае, если они имеют некоторые общие классы или наследуют некоторые классы друг друга. При этом в остальном *P1* и *P2* различны.

Учёт приведённых отношений производится при формировании набора предлагаемых паттернов для некоторого паттерна *X – SOP(X)* по следующим правилам:

- **Правило детализации описания** – если паттерн *A* специализирует паттерн *X*, то *A* включается в *SOP(X)*;
- **Правило детализации описания части** – если паттерн *B* часть паттерна *X*, а паттерн *A* специализирует паттерн *B*, то паттерн *A* включается в *SOP(X)*;

- **Правило расширения описания** – если паттерн *A* соотнесён с паттерном *X* через выбранное пользователем понятие, то он включается в множество *SOP(X)*;
- **Правило описания целого** – если паттерн *A* содержит паттерн *X* в качестве части, то *A* включается в *SOP(X)*.

При добавлении нового паттерна в синтезируемый пользователю представляются на выбор различные варианты добавления в виде:

$$AO_i = \{action_n(\{EConcept_j\}, \{AConcept_k\})\},$$

где $\{EConcept_j\}$ – подмножество понятий синтезируемого паттерна и $\{AConcept_k\}$ подмножество понятий добавляемого паттерна, над которыми будет произведено действие - $action_n$.

Действия зависят от конкретных паттернов и могут включать переименование понятия синтезируемого паттерна с учётом добавляемого, определение добавляемого понятия как супер/подкласса для существующего или его замещения и другие. Например, при расширении понятия «Событие» (Event) посредством паттерна «Коллекция» (Collection) с целью представления его в виде некоторого множества, будет предложено сменить имя у понятия «Событие» на «КоллекцияСобытие» (EventCollection).

2 Особенности использования некоторых паттернов при синтезе

Проведение синтеза с использованием некоторых общих паттернов содержания имеет некоторые особенности. Одним из таких паттернов является «Ситуация» (Situation). Так как он используется в качестве основы (как часть и/или предок) для некоторых других общих паттернов содержания (далее будем называть их «Ситуационные паттерны»), то приведенные далее приемы применимы и к ним. Ситуационные паттерны применяются в случае необходимости представления n-арного отношения (n>2) между сущностями ПрО. Например, представление статуса объекта, участвующего в некотором событии, или наличие у пациента диагноза с определённой вероятностью. В этом случае отношения будет тернарными. Ввиду того, что на сегодняшний день распространенным средством описания онтологий является язык веб-онтологий (Ontology Web Language, OWL), который позволяет задавать только бинарные отношения, представление отношений большей арности производится с помощью определения «реифицированного отношения» (reified relation). Данное «отношение» является специальным классом «Ситуация» (Situation), который имеет бинарные отношения «задан-для» (isSettingFor) со всеми участниками n-арного отношения (компонентами ситуации), (рисунок 1).

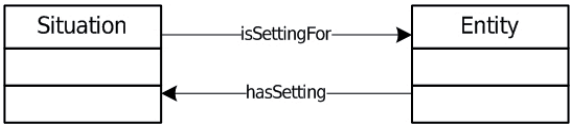


Рисунок 1 – Диаграмма классов CDP «Ситуация» (Situation)

Заметим, что обычно при использовании данного паттерна отношение «задан-для» заменяется более конкретным отношением-наследником, указывающим на роль соответствующего понятия в данной ситуации. Таким образом, все компоненты ситуации – участники n-арного отношения связаны между собой через отношения с понятием ситуации.

При выполнении синтеза с использование ситуационных паттернов возможны следующие случаи:

1) ситуационный паттерн применяется к понятию синтезируемого паттерна и при этом в последнем нет других ситуационных паттернов, добавленных ранее;

2) ситуационный паттерн применяется к понятию синтезируемого паттерна, которое уже является компонентом ситуации, заданной другим ситуационным паттерном;
3) ситуационный паттерн применяется к понятию синтезируемого паттерна, которое не является компонентом ситуации, заданной другим ситуационным паттерном. При этом среди паттернов, добавленных на предыдущих шагах, есть ситуационные.

В первом случае пользователю предлагаются на выбор несколько вариантов добавления, определяющих какие из понятий следует включить в ситуацию, определяемую добавляемым паттерном.

Во втором случае варианты добавления будут включать выполнение:

- **поглощения** (absorption) ситуации, заданной в добавляемом ситуационном паттерне. В этом случае её компоненты будут включены в существующую в синтезируемом паттерне;
- **включения** (inclusion) ситуации, заданной в добавляемом ситуационном паттерне, в виде компонента в существующую в синтезируемом паттерне.

Поглощение следует выполнять, когда при будущем использовании онтологии, включающей синтезируемый паттерн, не предполагается выделять отдельно контекст ситуации (набор ее компонентов и отношений между ними) добавляемого ситуационного паттерна. Это позволит уменьшить в онтологии количество классов-ситуаций, которые являются в большей степени искусственными в том смысле, что часто не соответствуют понятиям ПрО.

Включение ситуации в виде компонента существующей следует выполнять, когда её контекст является значимым. К таким случаям относится использование при синтезе паттерна «Последовательность» (Sequence) или его производных. С помощью этого паттерна можно задать порядок следования некоторых сущностей за счет определения между ними отношений «следует» (follows), «следует-непосредственно» (directlyFollows), «предшествует» (precedes), «предшествует-непосредственно» (directlyPrecedes) (рисунок 2).

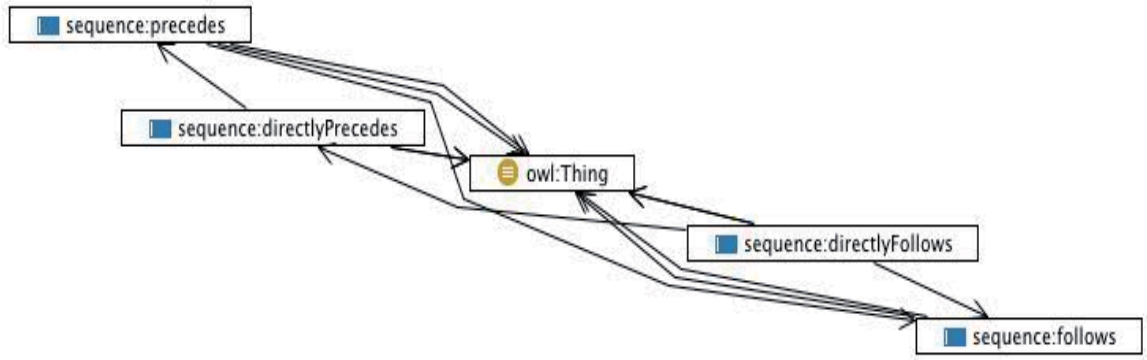


Рисунок 2 – Диаграмма классов CDP «Последовательность» (Sequence)

При добавлении данного паттерна в существующую ситуацию производится его приведение к ситуационному паттерну (рисунок 3).

Это предполагает представление заданной в нем последовательности в виде класса-ситуации «СитуацияПоследовательность» (SequenceSit), включающей единственный компонент – первый элемент (FirstSegment), который в свою очередь ссылается через отношение «имеет-следующей-сегмент» (hasNextSeg) на следующий и на упорядочиваемую сущность (Entity).

Полученный класс-ситуация «СитуацияПоследовательность» (SequenceSit) становится при добавлении компонентом существующей. Таким образом, удается сохранить заданный порядок на элементах.

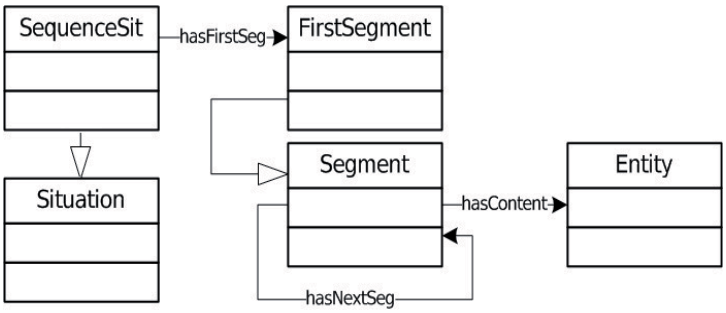


Рисунок 3 – Диаграмма классов CDP «Последовательность» (Sequence), преобразованного в ситуационный паттерн

Рекомендуется также выполнять включение при добавлении ситуационных паттернов, задающих сложные свойства. Такие свойства обычно представляют атрибуты объектов ПрО с дополнительной характеристикой (вероятностью, точностью и др.) или включают несколько аспектов (например, компания имеет ежемесячную прибыль 1 000 000 руб, которая снижается). Данные свойства также представляются в виде подклассов класса «Ситуация», имеющих специальное наименование в терминах ПрО (например, «ТрендИзмененияДохода»). Эти подклассы будут потеряны в случае выполнения поглощения таких ситуаций, так как в этом случае их компоненты станут компонентами существующей ситуации, а они сами станут не нужны и будут отброшены. Это может отрицательно сказаться на полноте представления понятийной системы ПрО в онтологии.

В третьем случае пользователю будут предложены варианты добавления двух видов. Первые будут предполагать добавление ситуации с включением в неё различных комбинаций понятий, не являющихся компонентами других ситуаций в синтезируемом паттерне. Вторые будут представлять возможные включения в существующие или поглощения ими добавляемой ситуации.

3 Пример синтеза составного онтологического паттерна содержания

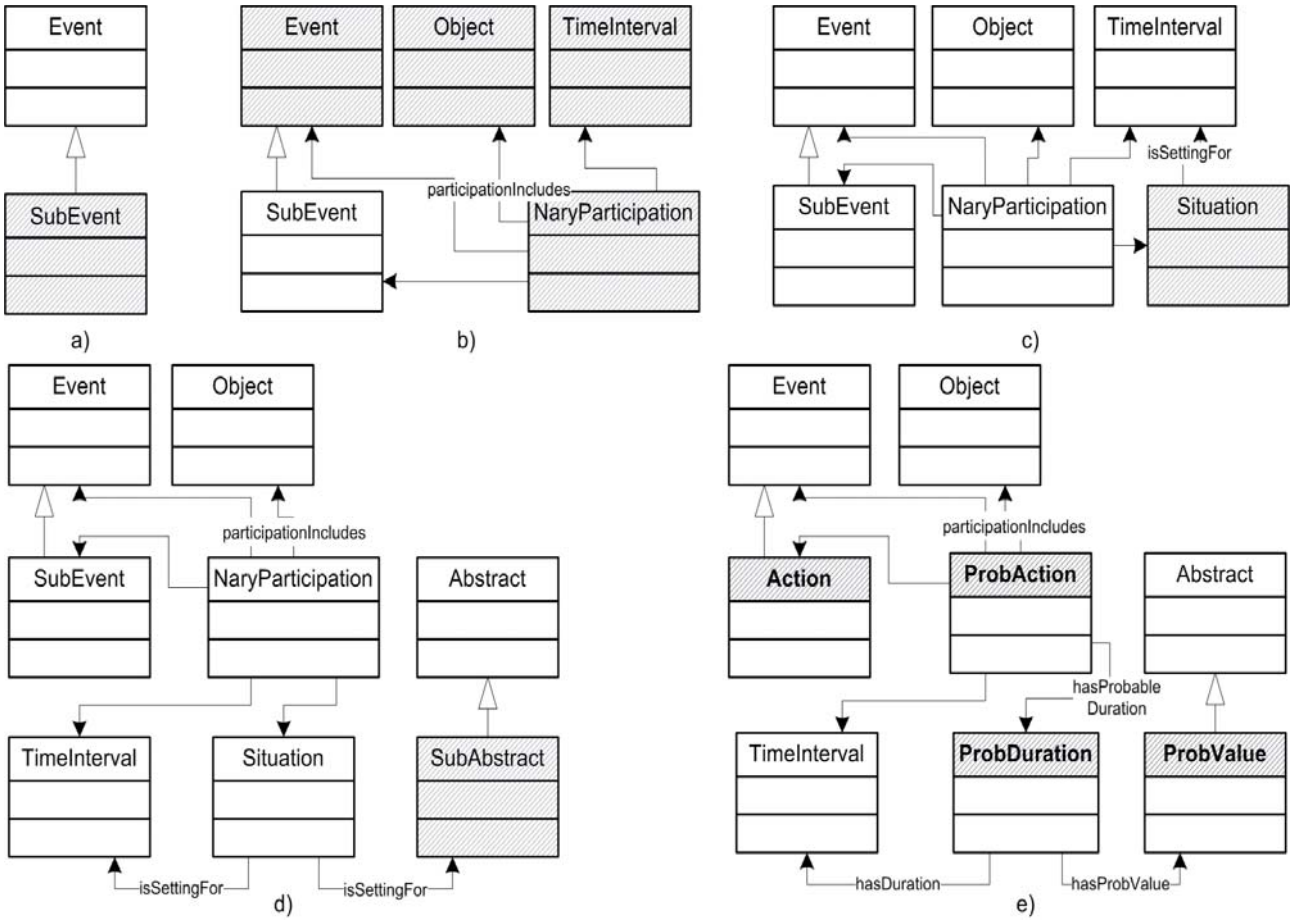
В качестве примера рассмотрим синтез паттерна для представления в онтологии информации о вероятной продолжительности выполнения работ при разработке вебсайта различными разработчиками. Данная информация может быть полезна для планирования деятельности в фирме, специализирующийся на создании веб-ресурсов или же при подборе исполнителей на бирже фрилансеров.

Если абстрагироваться от описания задачи в терминах ПрО, то в данном случае нужно представить действие, состав участников которого определяет его вероятную продолжительность. Подчеркнём, что в данном случае недостаточно просто создать классы, представляющие действие, участника и продолжительность и соединить их отношениями. Необходимо определить между ними тернарное отношение, тем самым объединить их в единый контекст. В результате синтеза будет получен инвариантный по отношению к ПрО паттерн содержания.

Дальнейшее применение этого абстрактного паттерна заключается в его специализации, то есть определении для его классов и отношений наследников, соответствующих конкретным объектам и отношениям ПрО. В нашем случае такими наследниками могут быть класс «Кастомизация интерфейса» (Design customization) - подкласс действия, представляющий один из видов работ при создании веб-ресурса, и «Веб-разработчик» (Web-developer) – под-

класс объекта, представляющий исполнителя задачи. Рассмотрим сначала процедуру синтеза инвариантного к ПрО составного паттерна.

Существует несколько способов его получения. Далее будет рассмотрен один из возможных. В качестве исходного на первом шаге используем паттерн «Виды сущностей» (TypeOfEntities). Из данного паттерна используем его класс «Событие» (Event), ввиду того, что действие (Action), которое мы хотим представить, будет рассматриваться как его подкласс. Таким образом, добавление паттерна «Виды сущностей» подразумевает включение в синтезируемый класса «Событие» и создание его подкласса - «ПодСобытие» (SubEvent). Данный класс генерируется автоматически, его имя может быть изменено на последнем шаге процедуры (рисунок 4а).



a), b), c), d), e) – виды (шаги) формирования составного онтологического патттерна

Рисунок 4 – Пример синтеза составного онтологического паттерна

Множество предлагаемых паттернов на данном шаге будет иметь вид:

SOP_1(*SubEvent*) = {*Componency, PartOf, Timeinterval, Region, Sequence, Bag, List, Collection, Set, TypesOfEntities, CoParticipation, NaryParticipation, ParticipantRole, Participation, Classification, Situation*}.

На следующем шаге для расширения понятия «ПодСобытие» выбирается ситуационный паттерн «N-арное участие» (NaryParticipation), так как он определяет необходимые атрибуты события (участники и продолжительность). Предлагаемый способ добавления будет следующим:

$AO_1 = \{addSitComponent_n(\{SubEvent\},\{NaryParticipation\})\}$.

Данный способ включает класс «ПодСобытие» в ситуацию, заданную добавляемым паттерном «N-арное участие». Синтезируемый паттерн примет вид (рисунок 4b).

Заметим, что в полученном на данном шаге паттерне уже имеется класс, который определяет временной интервал действия – «Временной интервал» (TimeInterval). Однако нам необходимо добавить дополнительную характеристику – его вероятность. Для этого надо сформировать ещё одно тенарное отношение с помощью паттерна «Ситуация». Для этого выбираем понятие «Временной интервал» для расширения.

Множество предлагаемых паттернов на данном шаге будет иметь вид:

$SOP_2(TimeInterval)$ = {*Componency, ActingFor, PartOf, Parameter, Timeinterval, Bag, Sequence, List, Set, CoParticipation, NaryParticipation, ParticipantRole, Participation, Classification, Situation, Collection, TypesofEntities*}.

Выбираем паттерн «Ситуация» (Situation) для добавления. Предлагаемые варианты добавления будут заключаться в поглощении добавляемой ситуации или её включении с внесением в неё различных комбинаций понятий синтезируемого паттерна:

$AO_1 = \{absorbSit(\{NaryParticipation\},\{Situation\})\}$,

$AO_{2-4} = \{includeSit(\{NaryParticipation\},\{Situation\}),$

$addSitComponent(\{TimeInterval\},\{Situation\})/$

$addSitComponent(\{TimeInterval, SubEvent\},\{Situation\})/...\}$

Выбираем вариант AO_2 – включение ситуации с добавлением в неё понятия «Временной интервал» (TimeInterval). Синтезируемый паттерн примет вид, показанный на рисунке 4c.

На следующем шаге нам необходимо дополнить добавленную ситуацию понятием, которое будет обозначать вероятность. Для этого выбираем понятие «Ситуация» (Situation) для расширения и паттерн «Виды сущностей» (TypeOfEntities) для добавления. Среди предложенных вариантов выбирается вариант с добавлением понятия «Абстрактный объект» (Abstract), которое будет использоваться для представления вероятности:

$AO_n = \{addSitComponent(\{Situation\},\{SubAbstract\})\}$.

В данном случае, как и при прошлом использовании паттерна «Виды сущностей», для класса, представляющего выбранное понятие, автоматически создаётся подкласс – «ПодАбстрактный объект» (SubAbstract), который и добавляется в ситуацию (рисунок 4d).

Синтез паттерна завершается корректировкой наименований отношений и концептов (рисунок 4e). Полученный паттерн позволяет отвечать на квалификационные вопросы паттернов, использованных для его синтеза, а также на вопросы, определяемые заданной ситуацией. Например, «Какова вероятность данной длительности действия с этими исполнителями?». Таким образом, в результате был синтезирован паттерн, позволяющий представлять в онтологии более сложные описания знаний ПрО.

Специализируем полученный паттерн, определив подклассы «Кастомизация интерфейса» (Design customization) и «Веб-разработчик» (Web-developer), соответствующие терминам ПрО (рисунок 5).

Использование данного паттерна в рамках онтологии, например биржи фрилансеров, позволит формализовать предложения по выполнению одной конкретной работы по кастомизации дизайна сайта разными исполнителями с определённой вероятной продолжительностью. На основе этой информации можно впоследствии производить подбор исполнителя или определять возможность завершения данной работы в отведённое время.

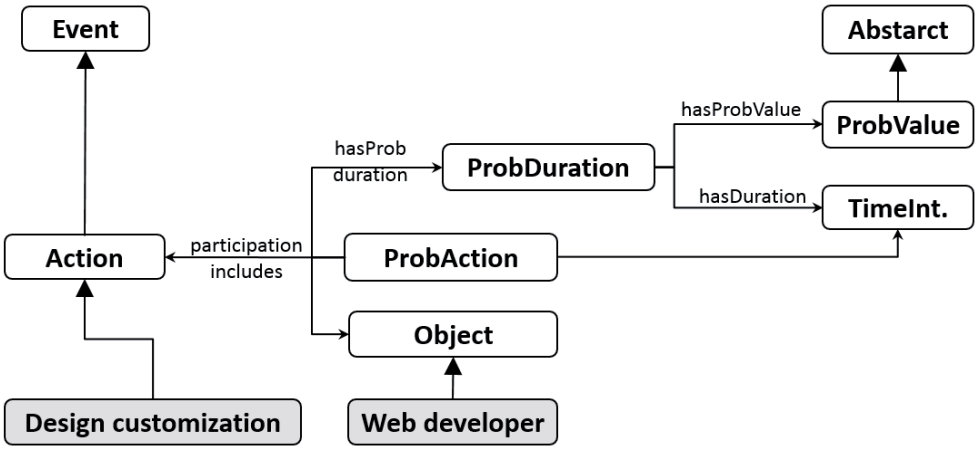


Рисунок 5 – Специализация составного онтологического паттерна

4 Заключение

В статье рассмотрена проблема синтеза составных онтологических паттернов содержания для их последующего применения при решении задач представления знаний ПрО. Определена общая процедура синтеза, а также рассмотрены некоторые случаи комбинирования общих паттернов содержания. Подбор паттернов, используемых при синтезе, производится по предложенным правилам на основе отношений между паттернами.

На основе полученных результатов планируется разработка программного средства, которое позволит облегчить экспертам формирование набора паттернов для представления знаний ПрО с необходимой детализацией.

Благодарности

Работа выполнена при поддержке Российского фонда фундаментальных исследований – грант 15-07-03321.

Список источников

[1] ***Liebig, T.*** OntoTrack: Combining Browsing and Editing with Reasoning and Explaining for OWL Lite Ontologies The Semantic Web, / Liebig, T., McIlraith, S., Plexousakis, D., Harmelen, F. OntoTrack // Proceedings of ISWC 2004, Springer Berlin Heidelberg, 2004, 3298, p. 244-258.

[2] ***Motta, E. A*** Novel Approach to Visualizing and Navigating Ontologies / Motta, E., Mulholland, P., Peroni, S., d'A qui n, M., Manuel Gomez-Perez, J., Mendez, V., Zablith, F. // Proceedings of the 10th International Conference on The Semantic Web - Volume Part I, Springer-Verlag, 2011, p. 470-486.

[3] ***Tsarkov, D.*** Chainsaw: a Metareasoner for Large Ontologies / Tsarkov, D., Palmisano, I. // Proceedings of the 1st International Workshop on OWL Reasoner Evaluation (ORE-2012), Manchester, UK, July 1st, 2012, 2012

[4] ***Bijan, P***. CropCircles: Topology Sensitive Visualization of OWL Class Hierarchies / Bijan, P., Wang, T., Golbeck, J. // The Semantic Web - ISWC 2006, 5th International Semantic Web Conference, ISWC 2006, Athens, GA, USA, November 5-9, 2006, Proceedings, 2006, p. 695-708.

[5] ***Suárez-Figueroa, M***. The NeOn Methodology framework: A scenario-based methodology for ontology development Applied Ontology / Suárez-Figueroa, M., Gómez-Pérez, A., Fernández-López, M. // Applied ontology 10(2):107-145, September 2015, 10, p. 107-145.

[6] ***Guarino, N., Welty, C***. An overview of OntoClean / Guarino, N., Welty, C. // In Staab, S. and Studer, R. (Eds), Handbook on Ontologies, Springer, Berlin, 2004, pp. 151-172

[7] ***Malzahn, N.*** Collaborative Ontology Development - Distributed Architecture and Visualization / Malzahn, N., Weinbrenner, S. Hüsken, P. Ziegler, J. Hoppe, H.// Proceedings of the German E-Science Conference, Max Planck Digital Library, 2007

[8] ***Amal, Z., Roger, N.,*** Building Domain Ontologies from Text for Educational Purposes / Amal, Z., Roger, N., // IEEE Transactions on Learning Technologies, vol.1, no. 1, pp. 49-62, January-March 2008, doi:10.1109/TLT.2008.12

[9] ***Ломов, П.А***. Применение паттернов онтологического проектирования для создания и использования онтологий в рамках интегрированного пространства знаний / П.А. Ломов // Онтология проектирования – 2015. - Том 5, №2(16). – с.233-245. - DOI: 10.18287/2223-9537-2015-5-2-233-245.

[10] ***Gangemi A***. Ontology Design for Interaction in a Reasonable Enterprise / Gangemi A., Presutti V., // Handbook of Ontologies for Business Interaction, 2007.

[11] ***Gangemi, A.,*** Ontology Design Patterns for Semantic Web Content. // Proceedings of the Fourth International Semantic Web Conference, Galway, Ireland, pp. 262-276, 2005. Springer.

[12] ***Ломов, П.А***. Использование отношений между онтологическими паттернами содержания при работе с онтологиями / П.А. Ломов // Информационные системы и технологии. 2016. № 2 (94). С. 30-39. ISSN 2072-8964.

AUTOMATION OF SYNTHESIS OF COMPOSITE CONTENT ONTOLOGY DESIGN PATTERN

P.A. Lomov

Institute for Informatics and Mathematical Modeling of Technological Processes,
The Kola Science Center of RAS, Apatity, Myrmansk region, Russia
lomov@iimm.ru

Abstract

Using of Ontology Design Patterns (ODPs) become useful for development and reengineering ontologies. ODPs represent encodings of best practices supporting ontology construction by facilitating reuse of proven solution principles. In this paper, we focus on Content ODPs (CDPs), which represent small ontology fragments that encode general use cases (e.g. participation in event, role playing, parts of object.). Content CDPs are used as building blocks during ontology development. In such cases they could be specializes, extends, integrates by user to obtain new composite CDP which would allow to provide more expressive representation of domain concept in the ontology is being developed. But to make such CDP composition it is not enough to simply join together corresponding ontology fragments. Often it is need to change them and link their elements in special way. It demands that developer understands the meaning of patterns structure and has additional skills of ontology engineering and therefore it may reduce the benefits of using CDPs. Therefore in this paper automatic selection of suitable CDPs on the base of the predefined relations between them and the procedure for synthesis of new composite CDP are suggested. At the end of the paper an example of the synthesis of composite CDP for a particular task is considered.

Key words: *ontology, ontology design patterns, ontology engineering, synthesis automation.*

Citation: *Lomov PA.* Automation of synthesis of composite content ontology design pattern. Ontology of designing. 2016; 6(2): 162-172. DOI: 10.18287/2223-9537-2016-6-2-162-172.

References

[1] *Liebig T., McIlraith N., Plexousakis D., Harmelen F*. OntoTrack: Combining Browsing and Editing with Reasoning and Explaining for OWL Lite Ontologies The Semantic Web в ISWC 2004, Springer Berlin Heidelberg, 2004, 3298, pp.244-258.

[2] *Motta E., Mulholland P., Peroni S., d'A quin M., Gomez-Perez J., Mendez V., Zablith F*. A Novel Approach to Visualizing and Navigating Ontologies Proceedings of the 10th International Conference on The Semantic Web - Volume Part I, Springer-Verlag, 2011, pp.470-486.

[3] *Tsarkov D., Palmisano I.* Chainsaw: a Metareasoner for Large Ontologies Proceedings of the 1st International Workshop on OWL Reasoner Evaluation (ORE-2012), Manchester, UK, July 1st, 2012.
[4] *Bijan P., Wang T., Golbeck J.* CropCircles: Topology Sensitive Visualization of OWL Class Hierarchies The Semantic Web - ISWC 2006, 5th International Semantic Web Conference, ISWC 2006, Athens, GA, USA, November 5-9, 2006, Proceedings, pp.695-708.
[5] *Suárez-Figueroa M., Gómez-Pérez A., Fernández-López M.* The NeOn Methodology framework: A scenario-based methodology for ontology development Applied ontology 10(2):107-145, September 2015, 10, p. 107-145.
[6] *Guarino N., Welty C*. An overview of OntoClean. In Staab, S. and Studer, R. (Eds), Handbook on Ontologies, Springer, Berlin, 2004, pp.151-172.
[7] *Malzahn N., Weinbrenner S., Hüsken P., Ziegler J., Hoppe H*. Collaborative Ontology Development - Distributed Architecture and Visualization Proceedings of the German E-Science Conference, Max Planck Digital Library, 2007.
[8] *Zouaq A., Nkambou R*. Building Domain Ontologies from Text for Educational Purposes, IEEE Transactions on Learning Technologies, vol.1, no. 1, pp.49-62, January-March 2008, DOI:10.1109/TLT.2008.12
[9] *Lomov PA*. Application of ontology design patterns to development and use of ontologies in an integrated knowledge space [In Russian]. Ontology of designing. 2015; V.5, 2(16): 233-245. DOI:10.18287/2223-9537-2015-5-2-233-245.
[10] *Gangemi A., Presutti V.,* Ontology Design for Interaction in a Reasonable Enterprise // Handbook of Ontologies for Business Interaction, 2007.
[11] *Gangemi A*., Ontology Design Patterns for Semantic Web Content. Proceedings of the Fourth International Semantic Web Conference, Galway, Ireland, pp. 262-276, 2005. Springer.
[12] *Lomov PA.* Using the relationship between patterns of ontological content when working with ontologies [In Russian]. Information Systems and Technology [Informacionnye sistemy i tekhnologii]. 2016. № 2 (94). P.30-39. ISSN 2072-8964.

Сведения об авторе

Ломов Павел Андреевич, 1984 г.р., окончил Кольский филиал Петрозаводского государственного университета (2006), кандидат технических наук, научный сотрудник Института информатики и математического моделирования технологических процессов Кольского научного центра РАН. Области научных интересов: представление знаний, онтологическое моделирование, Semantic web, информационная безопасность.
Lomov Pavel Andreevich, (b.1984) PhD, research associate of Institute for Informatics and Mathematical Modelling of Technological Processes of the Kola Science Center RAS. Research interests: knowledge representation, ontological modeling, Semantic web, information security.