OSTIS-2016
(Open Semantic Technologies for Intelligent Systems)

УДК 004.822:514

ФОРМАЛЬНОЕ СЕМАНТИЧЕСКОЕ ОПИСАНИЕ ЦЕЛЕНАПРАВЛЕННОЙ ДЕЯТЕЛЬНОСТИ РАЗЛИЧНОГО ВИДА СУБЪЕКТОВ

Шункевич Д.В.[*], Губаревич А.В.[*], Святкина М.Н.[**], Моросин О.Л.[***]

Белорусский государственный университет информатики и радиоэлектроники, г. Минск, Республика Беларусь

shunkevichdv@gmail.com

st.hubarevich@gmail.com

** *Московский государственный технический университет им. Н.Э. Баумана, г. Москва, Россия*

maria.svyatkina@gmail.com

*** *Национальный Исследовательский Университет «МЭИ», г. Москва, Россия*

oleg@morosin.ru

В работе рассматриваются формальные модели, позволяющие описывать целенаправленную деятельность субъектов различного рода с использованием семантических сетей с теоретико-множественной интерпретацией. В частности, рассматривается типология действий, понятие задачи как спецификации действия, отношения, используемые для спецификации действий. Также рассматриваются дргуие классы спецификаций действия, такие как, план, программа, решение и другие.
Ключевые слова: семантические технологии, формализация деятельности, многоагентные системы, проект OSTIS.

Введение

Целю данной работы является разработка базовых принципов описания целенаправленной деятельности различного вида субъектов на основе семантических сетей с базовой теоретико-множественной интерпретацией. Разработанные в данной работе формальные модели и средства используются при разработке различных систем, управляемых знаниями, построенных на основе *Технологии OSTIS* [Голенков, 2015]. Отметим, что разработанные модели и средства применимы для описания в семантической памяти не только деятельности субъектов в рамках самой этой памяти, но и описания деятельности субъектов, внешних по отношению к *ostis-системе* [Голенков, 2015], как в данный момент времени, так и в прошлом и будущем.

Семантическая теория деятельности подробно рассматривается в работах В.В. Мартынова [Мартынов, 1974], [Мартынов, 1977], [Мартынов, 1984], а также его учеников, в частности, в работе [Boyko, 2016]. В указанных работах подробно рассматривается семантическая типология действий, описываются подходы к формализации описания деятельности различного рода субъектов с использованием разработанного В.В. Мартыновым Универсального семантического кода (УСК). В этих работах детально рассматриваются такие понятия, как воздействие, субъект воздействия (агент), объект и другие, рассматриваемые также и в данной работе.

Кроме того, в работе [Шенк, 1980] рассматриваются такие понятия, близкие рассматриваемым в данной, как воздействие, действие (акт), деятель (субъект) и другие.

Целью данной работы, в отличие от указанных, является создание формальных моделей и средств описания целенаправленной деятельности различного вида субъектов в системах, управляемых знаниями, на основе формальных моделей представления знаний, используемых в Технологии OSTIS [Голенков, 2012], что, однако, никак не

противоречит исследованиям, описанным в указанных работах.

Данная работа является частью базы знаний системы IMS OSTIS [IMS, 2016], в рамках которой она представлена в виде раздела, описывающего соответствующую предметную область [Давыденко, 2016b].

1. Базовая типология действий

1.1. Понятие действия и типология действий

В рамках предлагаемой модели понятие ***действие*** является частным случаем понятия ***воздействие***, которое рассматривается в более общих предметных областях в рамках базы знаний IMS [IMS, 2016].

Каждому ***воздействию*** может быть поставлена в соответствие (1) некоторая *воздействующая сущность**, т.е. сущность, осуществляющая ***воздействие*** (в частности, это может быть некоторое физическое поле), и (2) некоторый *объект воздействия**, т.е. сущность, на которую воздействие направлено. Если ***воздействие*** связано с *материальной сущностью*, то его объектом воздействия является либо сама эта *материальная сущность*, либо некоторая ее пространственная часть.

В свою очередь, каждое ***действие***, выполняемое тем или иным *субъектом*, одновременно можно трактовать и как процесс решения некоторой задачи, т.е. как процесс достижения заданной цели в заданных условиях.

Предполагается, что любое ***действие***, выполняемое каким-либо *субъектом*, направлено на решение какой-либо задачи и выполняется целенаправленно. При этом явное указание *действия* и его связи с конкретной *задачей* может не всегда присутствовать в памяти. Некоторые задачи могут решаться определенными агентами перманентно, например, оптимизация базы знаний, поиск некорректностей и т.д., и для подобных задач не всегда есть необходимость явно вводить *структуру*, являющуюся формулировкой *задачи*.

Каждое ***действие*** может обозначать сколь угодно малое преобразование, осуществляемое во внешней среде либо в памяти некоторой системы, однако в памяти явно вводятся только знаки тех ***действий***, для которых есть необходимость явно хранить в памяти их спецификацию в течение некоторого времени.

При выполнении действия можно выделить следующие этапы:

- построение плана деятельности; декомпозиция (детализация) исходного действия;
- выполнение построенного плана действий;

Формальная спецификация понятия ***действие*** в SCn-коде (в том числе – классификация действий):

действие
= акция
= сделать
= работа
= процесс выполнения некоторой работы
= процесс решения некоторой задачи
= процесс достижения некоторой цели
= дело
= мероприятие
= воздействие
= целостный фрагмент некоторой деятельности
= целенаправленный процесс, управляемый некоторым субъектом
= процесс выполнения некоторого действия некоторым субъектом (исполнителем) над некоторыми объектами
<= разбиение:*
Разбиение класса действий по отношению к памяти информационной системы
=
{
- *информационное действие*
  => включение:*
  действие в sc-памяти
- *поведенческое действие*
  => включение:*
  действие во внешней среде ostis-системы
- *эффекторное действие*
  => включение:*
  эффекторное действие ostis-системы
- *рецепторное действие*
  => включение:*
  рецепторное действие ostis-системы

}
<= разбиение:*
Разбиение класса действий по отношению к текущему моменту времени
=
{
- *инициированное действие*
- *планируемое действие*
  = будущее действие
- *выполненное действие*

}

1.2. Типы действий по отношению к памяти информационной системы

Результатом выполнения ***информационного действия*** является в общем случае некоторое новой состояние памяти информационной системы (не обязательно *sc-памяти*), достигнутое исключительно путем преобразования информации, хранящейся в памяти системы, то есть либо посредством генерации новых знаний на основе уже имеющихся, либо посредством удаления знаний, по каким-либо причинам ставших ненужными. Следует отметить, что если речь идет об изменении состояния *sc-памяти*, то любое преобразование информации можно свести к ряду элементарных действий генерации, удаления или изменения

инцидентности *sc-элементов* друг относительно друга.

В случае ***поведенческого действия*** результатом его выполнения будет новое состояние внешней среды. Очень важно отметить, что под внешней средой в данном случае понимаются также и компоненты системы, внешние с точки зрения памяти, то есть не являющиеся хранимыми в ней информационными конструкциями. К таким компонентам можно отнести, например, различные манипуляторы и прочие средства воздействия системы на внешний мир, то есть к поведенческим задачам можно отнести изменение состояния механической конечности робота или непосредственно вывод некоторой информации на экран для восприятия пользователем.

Под ***эффекторным действием*** понимается действие, связанное с воздействием ostis-системы на внешнюю с точки зрения системы среду. Под ***рецепторным действием*** понимается действие, связанное с воздействием внешнего субъекта на ostis-систему, т.е. на ее датчики, рецепторы и т.п.

1.3. Типы действий по отношению к текущему моменту времени

Во множество ***инициированных действий*** входят *действия*, выполнение которых инициировано в результате какого-либо события.

В общем случае, *действия* могут быть инициированы по следующим причинам:

- *действие* инициировано явно путем проведения соответствующей *sc-дуги принадлежности* каким-либо *субъектом* (*заказчиком**). В случае *действия в sc-памяти*, оно может быть инициировано как внутренним *sc-агентом* системы, так и пользователем при помощи соответствующего пользовательского интерфейса. При этом, спецификация действия может быть сформирована одним *sc-агентом* [Шункевич, 2014], [Тарасов, 2002], а собственно добавление во множество ***инициированных действий*** может быть осуществлено позже другим *sc-агентом*.
- *действие* инициировано в результате того, что одно или несколько действий, предшествовавших данному в рамках некоторой декомпозиции, стали *прошлыми сущностями* (процедурный подход).
- действие инициировано в результате того, что в памяти системы появилась конструкция, соответствующая некоторому условию инициирования *sc-агента,* который должен выполнить данное *действие* (декларативный подход)

Следует отметить, что декларативный и процедурный подходы можно рассматривать как две крайности, использование только одной из который не является удобным и целесообразным. При этом, например, принципы инициирования по процедурному подходу могут быть полностью сведены к набору декларативных условий инициирования, но как было сказано, это не всегда удобно и наиболее рациональным будет комбинировать оба похода в зависимости от ситуации.

По сути, попадание некоторого *действия* во множество ***инициированных действий*** говорит о том, что спецификация данного *действия*, полностью сформирована, т.е. никаких дополнительных элементов, необходимых для решения поставленной задачи, не требуется, и соответствующий *sc-агент* (либо коллектив *sc-агентов,* либо внешний *субъект*) может приступать к выполнению действия. Однако стоит отметить, что с точки зрения исполнителя такая спецификация *действия* в общем случае может оказаться недостаточной или некорректной.

инициированное действие
=> включение:*
выполняемое действие
= настоящее действие
<= разбиение:*
{
- *активное действие*
- *отложенное действие*

}

Во множество ***выполняемых действий*** входят *действия*, к выполнению которых приступил какой-либо из соответствующих *субъектов*.

Попадание *действия* в данное множество говорит о следующем:

- рассматриваемое *действие* уже попало во множество *инициированных действий*.
- существует как минимум один *субъект*, условие инициирования которого соответствует спецификации данного *действия*.

После того, как собственно процесс выполнения завершился, *действие* должно быть удалено из множества ***выполняемых действий*** и добавлено во множество *выполненных действий* или какое-либо из его подмножеств.

Понятие ***выполняемое действие*** является неосновным, и вместо того, чтобы относить конкретные *действия* к данному классу, их относят к классу *настоящих сущностей*.

Во множество ***активных действий*** входят *действия*, выполнение которых осуществляется непосредственно в данный момент каким-либо *субъектом*.

Во множество ***отложенных действий*** входят *действия*, которые уже были инициированы, однако их выполнение невозможно по каким-либо причинам, например в случае, когда у исполнителя в данный момент есть более приоритетные задачи.

Во множество ***планируемых действий*** входят *действия*, начать выполнение которых запланировано на какой-либо момент в будущем.

Во множество ***выполненных действий*** попадают *действия*, выполнение которых с точки зрения завершено с точки зрения *субъекта*, осуществлявшего их выполнение. В зависимости от результатов конкретного процесса выполнения, рассматриваемое *действие* может стать элементом одного из подмножеств множества ***выполненных действий***.

Понятие ***выполненное действие*** является неосновным, и вместо того, чтобы относить конкретные *действия* к данному классу, их относят к классу *прошлых сущностей*.

Во множество ***успешно выполненных действий*** попадают *действия*, выполнение которых успешно завершено с точки зрения *субъекта*, осуществлявшего их выполнение, т.е. достигнута поставленная цель, например, получены решение и ответ какой-либо задачи, успешно преобразована какая-либо конструкция и т.д.

Если действие было выполнено успешно, то, в случае действия по генерации каких-либо знаний, к *действию* при помощи связки отношения *результат** приписывается *sc-конструкция*, описывающая результат выполнения указанного действия. В случае, когда *действие* направлено на какие-либо изменения базы знаний, *sc-конструкция*, описывающая результат действия, формируется в соответствии с правилами описания истории изменений базы знаний.

В случае, когда успешное выполнение *действия* приводит к изменению какой-либо конструкции в *sc-памяти*, которое необходимо занести в историю изменений базы знаний или использовать для демонстрации протокола решения задачи, то генерируется соответствующая связка отношения *результат**, связывающая *задачу* и *sc-конструкцию*, описывающую данное изменение.

выполненное действие
= прошлое действие
<= разбиение:*
{
- *успешно выполненное действие*
- *безуспешно выполненное действие*

}

Во множество ***безуспешно выполненных действий*** попадают *действия*, выполнение которых не было успешно завершено с точки зрения *субъекта*, осуществлявшего их выполнение, по каким-либо причинам.

Можно выделить две основные причины, по которым может сложиться указанная ситуация:

- соответствующая *задача* сформулирована некорректно;
- формулировка соответствующей *задачи* корректна и понятна системе, однако решение данной задачи в текущий момент не может быть получено за удовлетворительные с точки зрения заказчика или исполнителя сроки.

Для конкретизации факта некорректности формулировки задачи можно выделить ряд более частных классов ***безуспешно выполненных действий***, например:

- *действие*, спецификация которого противоречит другим знаниям системы (например, не выполняется неравенство треугольника);
- *действие*, при спецификации которого использованы понятия, неизвестные системе;
- *действие*, выполнение которого невозможно из-за недостаточности данных (например, найти площадь треугольника по двум сторонам);
- и другие

Для конкретизации факта безуспешности выполнения некоторого *действия* в системе могут также использоваться дополнительные подмножества данного множества, при необходимости снабженные естественно-языковыми комментариями.

Для указания приоритетности выполнения тех или иных действий дополнительно вводится понятие ***приоритет действия***.

приоритет действия
∋ *действие с очень высоким приоритетом'*
∋ *действие с высоким приоритетом'*
∋ *действие со средним приоритетом'*
∋ *действие с низким приоритетом'*
∋ *действие с очень низким приоритетом'*

Множество ***приоритет действия*** представляет собой семейство ролевых отношений, элементами которых являются *sc-дуги принадлежности*, связывающие множество поддействий в рамках декомпозиции некоторого более сложного *действия* и сами эти поддействия. Таким образом, данные ролевые отношения задают приоритетность выполнения более частных поддействий при выполнении некоторого общего действия. Приоритетность выполнения влияет на *действия*, независимые с точки зрения *последовательности действий**, и отражает влияние каждого более частного действия на качество результата выполнения общего действия.

1.4. Типы действий по атомарности

класс действий
= множество действий, однотипных в том или ином смысле
<= семейство подмножеств:*
действие
<= разбиение:*
{
- *атомарный класс действий*
- *неатомарный класс действий*

}

Принадлежность некоторого *класса действий* множеству ***атомарных классов действий*** фиксирует факт того, что при указании всех необходимых аргументов принадлежности *действия*

данному классу достаточно для того, чтобы некоторый *субъект* мог приступить к выполнению этого действия.

При этом, даже если *класс действий* принадлежит множеству ***атомарных классов действий***, не запрещается вводить более частные *классы действий*, для которых, например, заранее фиксируется один из аргументов.

Если конкретный ***атомарный класс действий*** является более частным по отношению к *действиям в sc-памяти*, то это говорит о наличии в текущей версии системы как минимум одного *sc-агента*, ориентированного на выполнение действий данного класса.

Принадлежность некоторого *класса действий* множеству ***неатомарных классов действий*** фиксирует факт того, что даже при указании всех необходимых аргументов принадлежности *действия* данному классу недостаточно для того, чтобы некоторый *субъект* приступил к выполнению этого действия, и требуются дополнительные уточнения.

2. Типология субъектов деятельности

Формальная спецификация понятия ***субъект*** в SCn-коде (в том числе – классификация субъектов):

субъект
= активная сущность
= сущность, способная самостоятельно выполнять некоторые виды действий
= агент деятельности
=> включение:*
- *Собственное Я*
- *внутренний субъект ostis-системы*
- *внешний субъект ostis-системы, с которым осуществляется взаимодействие*
- *внешний субъект ostis-системы, с которым взаимодействие не происходит*

Под ***внутренним субъектом ostis-системы*** понимается такой *субъект*, который выполняет некоторые *действия* в той же памяти, в которой хранится его знак.

К числу ***внутренних субъектов ostis-системы*** относятся входящие в нее *sc-агенты*, частные sc-машины, целые интеллектуальные подсистемы.

К числу ***внешних субъектов ostis-системы, с которыми осуществляется взаимодействие,*** относятся конечные пользователи *ostis-системы*, ее разработчики, а также другие компьютерные системы (причем, не только интеллектуальные).

3. Средства детализации процесса выполнения действий

Рассмотрим набор отношений, предназначенных для описания детализации процесса выполнения того или иного действия, то есть выделения более простых частных действий.

декомпозиция действия*
*= сведение действия ко множеству более простых взаимосвязанных действий**
∈ *отношение декомпозиции*
∈ *квазибинарное отношение*

Связки отношения ***декомпозиция действия**** связывают *действие*, и множество частных *действий*, на которые декомпозируется данное *действие*. При этом первым компонентом связки является знак указанного множества, вторым компонентом – знак более общего *действия*.

Таким образом, ***декомпозиция действия**** это *квазибинарное отношение*, связывающее действие со множеством действий более низкого уровня, к выполнению которых сводится выполнение исходного декомпозируемого действия.

Стоит отметить, что каждое *действие* может иметь несколько вариантов декомпозиции в зависимости от конкретного набора элементарных действий, которые способна выполнять та или иная система *субъектов*.

Принцип, по которому осуществляется такая декомпозиция в различных подходах к решению задач будем называть стратегией решения задач.

поддействие*
*= частное действие**
∈ *бинарное отношение*
∈ *отношение таксономии*

Связки отношения ***поддействие**** связывают *действие*, и некоторое более простое частное *действие*, выполнение которого необходимо для выполнения исходного более общего *действия*.

последовательность действий*
*= порядок действий**
*= бинарная ориентированная связка, описывающая то, какое действие может быть инициировано после завершения выполнения другого (предшествующего)**
*= бинарная ориентированная связка, описывающая передачу управления от одного (предшествующего) действия к другому (последующему)**
*= goto**
∈ *отношение порядка*
=> включение:*
- *последовательность действий при положительном результате**
- *последовательность действий при отрицательном результате**

Связки отношения ***последовательность действий**** связывают знаки *действий*, выполняющихся в какой-либо последовательности в процессе решения какой-либо задачи. При этом считается, что если два *действия* связаны данным отношением, то *действие*, стоящее в данной связке на втором месте может быть выполнено только после выполнения *действия*, стоящего в данной связке на первом месте. Таким образом, каждое

действие может быть инициировано после завершения выполнения любого из предшествующих действий.

Для обеспечения возможности синхронизации выполнения действий используется класс действий *конъюнкция предшествующих действий.*

При этом дополнительно может указываться абсолютный *приоритет действия*, характеризующий принципиальную важность действия и срочность его выполнения, не всегда зависящую напрямую от других действий, но при этом влияющую на порядок выполнения действий из некоторого множества в целом.

конъюнкция предшествующих действий
= действие, заключающееся только в ожидании установлении факта завершения всех предшествующих действий
<= включение:*
действие

Действия класса ***конъюнкция предшествующих действий*** используются в тех случаях, когда выполнение некоторого действия должно начаться только после того, как будут выполнены все предшествующие действия, а не только одно из них. После того, как все предшествующие действия выполнены, инициируются действия, следующие за ***конъюнкцией предшествующих действий***.

В некоторых случаях бывает необходимо управлять процессом выполнения какой-либо последовательности действий в зависимости от выполнения дополнительных условий. Для осуществления таких проверок вводится класс действий *проверки условия*.

проверка условия
= if-действие
= действие, направленное на установление истинности или ложности заданного высказывания
<= включение:*
действие

Действия класса ***проверка условия*** предполагают проверку истинности или ложности некоторого высказывания (условия), и после выполнения в зависимости от результата данной проверки становятся *успешно выполненными действиями* или *безуспешно выполненными действиями*.

последовательность действий при положительном результате*
*= then**
∈ отношение порядка

Переход по связкам отношения ***последовательность действий при положительном результате**** от предшествующего действия проверки условия к последующему действию происходит при условии, если указанная проверка даст положительный результат, то есть предшествующее действие станет *успешно выполненным действием*.

последовательность действий при отрицательном результате*
*= else**
∈ отношение порядка

Переход по связкам отношения ***последовательность действий при отрицательном результате**** от предшествующего действия проверки условия к последующему действию происходит при условии, если указанная проверка даст отрицательный результат, то есть предшествующее действие станет *безуспешно выполненным действием*.

4. Спецификация действий

4.1. Задача как спецификация действия

Под ***задачей*** понимается формальное описание условия некоторой задачи, то есть, по сути, формальная спецификация некоторого действия, направленного на решение данной задачи, достаточная для выполнения данного действия каким-либо *субъектом*. В зависимости от конкретного класса задач, описываться может как внутреннее состояние самой интеллектуальной системы, так и требуемое состояние внешней среды. *sc-элемент*, обозначающий *действие* входит в *задачу* под атрибутом *ключевой sc-элемент'*.

Каждая ***задача*** представляет собой спецификацию действия, которое либо уже выполнено, либо выполняется в текущий момент (в настоящее время), либо планируется (должно) быть выполненным, либо может быть выполнено (но не обязательно).

Классификация задач может осуществляться по дидактическому признаку в рамках каждой предметной области, например, задачи на треугольники, задачи на системы уравнений и т.п.

Каждая *задача* может включать:

- факт принадлежности *действия* какому-либо частному классу *действий* (например, *действие. сформировать полную семантическую окрестность указываемой сущности*), в том числе состояние *действия* с точки зрения жизненного цикла (инициированное, выполняемое и т.д.);
- описание *цели** (*результата**) *действия*, если она точно известна;
- указание *заказчика* действия*;
- указание *исполнителя* действия* (в том числе, коллективного);
- указание *аргумента(ов) действия'*;
- указание инструмента или посредника *действия*;
- описание *декомпозиции действия**;
- указание *последовательности действий** в рамках *декомпозиции действия**, т.е построение плана решения задачи. Другими словами, построение плана решения представляет собой декомпозицию соответствующего *действия* на

систему взаимосвязанных между собой поддействий;

- указание области *действия*;
- указание условия инициирования *действия*;
- момент начала и завершения *действия*, в том числе планируемый и фактический, предполагаемая и/или фактическая длительность выполнения;

Некоторые задачи могут быть дополнительно уточнены контекстом – дополнительной информацией о сущностях, рассматриваемых в формулировке *задачи*, т.е. описанием того, что дано, что известно об указанных сущностях.

Построение плана решения задачи это декомпозиция спецификации процесса решения заданной задачи на систему

Кроме этого, *задача* может включать любую дополнительную информацию о действии, например:

- перечень ресурсов и средств, которые предполагается использовать при решении задачи, например список доступных исполнителей, временные сроки, объем имеющихся финансов и т.д.;
- ограничение области, в которой выполняется *действие*, например, необходимо заменить одну *sc-конструкцию* на другую по некоторому правилу, но только в пределах некоторого *раздела базы знаний*;
- ограничение знаний, которые можно использовать для решения той или иной задачи, например, необходимо решить задачу по алгебре используя только те утверждения, которые входят в курс школьной программы до седьмого класса включительно, и не используя утверждения, изучаемые в старших классах;
- и прочее

С одной стороны, решаемые системой задачи, можно классифицировать на *информационные задачи* и *поведенческие задачи*.

С точки зрения формулировки поставленной задачи можно выделить *декларативные формулировки задачи* и *процедурные формулировки задачи*. Следует отметить, что данные классы задач не противопоставляются, и могут существовать формулировки задач, использующие оба подхода.

Формальная спецификация понятия задача в SCn-коде:

задача
<= включение:*
семантическая окрестность
=> включение:*
- *процедурная формулировка задачи*
- *декларативная формулировка задачи*

=> включение:*
- *вопрос*
- *команда*

В случае ***процедурной формулировки задачи***, в формулировке задачи явно указываются аргументы соответствующего задаче *действия*, и в частности, вводится семантическая типология *действий*. При этом явно не уточняется, что должно быть результатом выполнения данного действия. Заметим, что, при необходимости, *процедурная формулировка задачи* может быть сведена к *декларативной формулировке задачи* путем трансляции на основе некоторого правила, например определения класса действия через более общий класс.

В случае *декларативной формулировки задачи*, при описании условия задачи специфицируется цель *действия*, т.е. результат, который должен быть получен при успешном выполнении *действия*.

Под ***вопросом*** понимается задача, направленная на удовлетворение информационной потребности некоторого субъекта-заказчика

Под командой понимается спецификация инициированного действия, т.е., по сути, инициированная задача.

класс команд
<= семейство подмножеств:*
команда
=> включение:*
- *класс интерфейсных пользовательских команд*

=> включение:*
- *класс интерфейсных команд пользователя ostis-системы*

=> включение:*
- *класс команд без аргументов*
- *класс команд с одним аргументом*
- *класс команд с двумя аргументами*
- *класс команд с произвольным числом аргументов*

<= разбиение:*
{
- *атомарный класс команд*
- *неатомарный класс команд*

}

Принадлежность некоторого *класса команд* множеству ***атомарных классов команд*** фиксирует факт того, что данная спецификация является достаточной для того, чтобы некоторый субъект приступил к выполнению соответствующего действия.

При этом, даже если *класса команд* принадлежит множеству ***атомарных классов команд*** не запрещается вводить более частные *классы команд*, в состав которых входит информация, дополнительно специфицирующая соответствующее *действие*.

Если соответствующий данному *классу команд* *класс действий* является более частным по отношению к *действиям в sc-памяти*, то попадание данного класса команд во множество ***атомарных***

классов команд говорит о наличии в текущей версии системы как минимум одного *sc-агента*, условие инициирования которого соответствует формулировке команд данного класса.

Принадлежность некоторого *класса команд* множеству ***неатомарных классов команд*** фиксирует факт того, что данная спецификация не является достаточной для того, чтобы некоторый субъект приступил к выполнению соответствующего действия, и требует дополнительных уточнений.

4.2. Отношения, специфицирующие действия

Рассмотрим набор отношений, используемых для формальной спецификации действий в рамках *задачи*.

Связки отношения ***результат**** связывают *sc-элемент*, обозначающий *действие,* и *sc-конструкцию*, описывающую результат выполнения рассматриваемого действия, другими словами, цель, которая должна быть достигнута при выполнении *действия*.

Результат может специфицироваться как атомарным высказыванием, так и неатомарным, т.е. конъюнктивным, дизъюнктивным, строго дизъюнктивным и т.д.

В случае, когда успешное выполнение *действия* приводит к изменению какой-либо конструкции в *sc-памяти*, которое необходимо занести в историю изменений базы знаний или использовать для демонстрации протокола решения задачи, генерируется соответствующая связка отношения ***результат****, связывающая *задачу* и *sc-конструкцию*, описывающую данное изменение. Конкретный вид указанной *sc-конструкции* зависит от типа действия.

Связки отношения ***исполнитель**** связывают *sc-элементы*, обозначающие *действие* и *sc-элементы*, обозначающие *субъекта*, который предположительно будет осуществлять, осуществляет или осуществлял выполнение указанного *действия*. Данное отношение может быть использовано при назначении конкретного исполнителя для проектной задачи по развитию баз знаний.

В случае, когда заранее неизвестно, какой именно *субъект** будет исполнителем данного *действия*, связка отношения ***исполнитель**** может отсутствовать в первоначальной формулировке *задачи* и добавляться позже, уже непосредственно при исполнении.

Когда действие выполняется (является *настоящей сущностью*) или уже выполнено (является *прошлой сущностью*), то исполнитель этого действия в каждый момент времени уже определен. Но когда действие только инициировано, тогда важно знать:

1) кто хочет выполнить это действие и насколько важно для него стать исполнителем данного действия;
2) кто может выполнить данное действие и каков уровень его квалификации и опыта;
3) кто и кому поручает выполнить это действие и каков уровень ответственности за невыполнение (приказ, заказ, официальный договор, просьба…);

При этом следует помнить, что связь отношения *исполнитель** в данном случае также является временной прогнозируемой сущностью.

Первым компонентом связок отношения *исполнитель** является знак *действия*, вторым - знак *субъекта*-исполнителя.

Связки отношения ***заказчик**** связывают *sc-элементы*, обозначающие *действие* и *sc-элементы*, обозначающие *субъекта*, который «заинтересован» в выполнении данного действия и, как правило, инициирует его выполнение. Данное отношение может быть использовано при указании того, кто поставил проектную задачу по развитию баз знаний.

Первым компонентом связок отношения ***заказчик**** является знак *действия*, вторым - знак *субъекта*-заказчика.

Связки отношения ***инициатор**** связывают *sc-элемент*, обозначающий *инициированное действие,* и знак *субъекта*, который является инициатором данного *действия*, то есть *субъектом*, который инициировал данное *действие* и, как правило, заинтересован в его успешном выполнении.

Связки отношения ***объект**** связывают *sc-элемент*, обозначающий *действие,* и знак той сущности, над которой (по отношению к которой) осуществляется данное *действие*, например знак *структуры*, подлежащий верификации.

Связки отношения ***контекст действия**** связывают *sc-элементы*, обозначающие *действие* и *sc-структуры*, обозначающие контекст выполнения данного *действия*, то есть некоторую дополнительную информации о тех сущностях, которые входят в описание *цели**. Как правило, контекст используется для указания собственно условия некоторой задачи, того, что дано, т.е. тех знаний, которые можно использовать для вывода новых знаний при решении задачи. Таким образом, контекст непосредственно влияет на то, как будет решаться та или иная задача, при этом даже задачи соответствующие одному классу действий, могут решаться по-разному.

контекст действия*
*= задачная ситуация**
*= что дано**
*= дополнительная информация о тех сущностях, которые входят в описание цели**
= связь между некоторой задачей (формулировкой задачи) и состоянием базы знаний, возможностей и навыков некоторого субъекта,

*перед которым поставлена указанная задача**

*= связь между формулировкой задачи, т.е. описанием того, что требуется, и контекстом этой задачи, т.е. описанием имеющихся ресурсов, описанием того, что дано**

Контекст может быть представлен не только в виде атомарного фактографического высказывания, но и в виде высказывания более сложного вида. Это может быть, например:

- определение множества, используемого в описании *цели**;
- утверждение, учет которого может быть полезен в решении задач;

Первым компонентом связок отношения ***контекст действия**** является знак *действия*, вторым - знак *структуры* [Давыденко, 2016а], обозначающей контекст.

аргумент действия'
=> включение:*
- *первый аргумент действия'*
- *второй аргумент действия'*
- *третий аргумент действия'*

∈ *ролевое отношение*

Связки ролевого отношения ***аргумент действия'*** указываются в рамках конкретного действия те *sc-элементы*, которые обозначают непосредственно аргументы данного *действия*, если они явно указываются (в случае процедурной формулировки задачи).

класс аргументов*
*= класс аргументов класса команд**

*= быть классом sc-элементов, экземпляры которого являются аргументами для заданного класса команд**

=> включение:*
- *класс первых аргументов**
- *класс вторых аргументов**

Связки отношения ***класс аргументов**** связывают *классы команд* (подмножества множества *команд*), и классы *sc-элементов*, которые могут быть аргументами действий, соответствующих данному *классу команд*. В случае, когда *команды* данного класса имеют один аргумент, используется собственно отношение ***класс аргументов****, в случае, когда больше команды данного класса имеют более одного аргумента, то используются подмножества данного отношения, такие как *класс первых аргументов**, *класс вторых аргументов** и т.д.

Если для некоторого *класса команд* не указан тип какого-либо из аргументов, то предполагается, что в качестве данного аргумента может выступать любой *sc-элемент*.

Первым компонентом связок отношения ***класс аргументов**** является знак *класса команд*, вторым – знак класса *sc-элементов,* которые могут быть *аргументами действий'*, соответствующих данному *классу команд*.

4.3. Другие виды семантических спецификаций действий

Кроме задачи как спецификации действия в целом необходимо выделить еще ряд *семантических окрестностей* [Давыденко, 2016а] специфицирующих конкретное действие с той или иной стороны.

4.3.1. *План выполнения действия*

Каждый ***план*** представляет собой *семантическую окрестность*, *ключевым sc-элементом'* является *действие*, для которого дополнительно детализируется предполагаемый процесс его выполнения. Основная задача такой детализации – локализация области базы знаний, в которой предполагается работать, а также набора агентов, необходимого для выполнения описываемого действия. При этом детализация не обязательно должна быть доведена до уровня элементарных действий, цель составления плана – уточнение подхода к решению той или иной задачи, не всегда предполагающее составления подробного пошагового решения.

При описании ***плана*** может быть использован как процедурный, так и декларативный подход. В случае процедурного подхода для соответствующего *действия* указывается его декомпозиция на более частные поддействия, а также необходимая спецификация этих поддействий. В случае декларативного подхода указывается набор подцелей (например, при помощи логических утверждений), достижение которых необходимо для выполнения рассматриваемого *действия*. На практике оба рассмотренных подхода можно комбинировать.

В общем случае ***план*** может содержать и переменные, например в случае, когда часть плана задается в виде цикла (многократного повторения некоторого набора действий). Также план может содержать константы, значение которых в настоящий момент не установлено и станет известно, например, только после выполнения предшествующих ему *действий*.

Каждый ***план*** может быть задан заранее как часть спецификации *действия*, т.е. *задачи*, а может формироваться *субъектом* уже собственно в процессе выполнения *действия*, например, в случае использования стратегии разбиения задачи на подзадачи. В первом случае ***план*** *включается** в *задачу*, соответствующую тому же действию.

4.3.2. *Программа выполнения класса действий*

программа
= программа выполнения действий некоторого класса

= обобщенный план

= обобщенный план выполнения некоторого класса действий

= *обобщенный план решения некоторого класса задач*
= *обобщенная спецификация декомпозиции любого действия, принадлежащего заданному классу действий*
= *знание о некотором классе действий (и соответствующем классе задач), позволяющее для каждого из указанных действий достаточно легко построить план его выполнения*
<= *включение*:*
знание
=> *включение*:*
программа в sc-памяти

Каждая ***программа*** представляет собой обобщенный *план* выполнения *действий*, принадлежащих некоторому классу, то есть *семантическую окрестность*, *ключевым sc-элементом'* является *класс действий*, для элементов которого дополнительно детализируется процесс их выполнения.

В остальном описание ***программы*** аналогично описанию *плана* выполнения конкретного *действия* из рассматриваемого *класса действий*.

Одному *классу действий* может соответствовать несколько ***программ***.

Входным параметрам ***программы*** в традиционном понимании соответствуют аргументы, соответствующие каждому *действию* из *класса действий*, описываемого ***программой***. При генерации на основе ***программы*** *плана* выполнения конкретного *действия* из данного класса эти аргументы принимают конкретные значения.

Каждая ***программа*** представляет собой систему описанных действий с дополнительным указанием для действия:

- либо *последовательности выполнения действий** (передачи инициирования), когда условием выполнения (инициирования) действий является завершение выполнения одного из указанных или всех указанных действий;
- либо события в базе знаний или внешней среде, являющегося условием его инициирования;
- либо ситуации в базе знаний или внешней среде, являющейся условием его инициирования;

Рассмотрим пример программы базового языка программирования, ориентированного на обработку семантических сетей – *Языка SCP* (Рисунок 1). Данный пример представляет собой *scp-программу*, имеющую два аргумента (множество и элемент, который надо добавить во множество). В соответствии с алгоритмом программы осуществляется проверка того, имеется ли указанный элемент в указанном множестве, и если нет, то добавляет элемент во множество. В рамках данной программы исходный *scp-процесс* декомпозируется на три более простых подпроцесса (*поддействия**), связанных соответствующими подклассами отношения *последовательность выполнения действий**. Часть используемых в данном примере понятий идентифицировано при помощи *системных идентификаторов**, например *nrel_then*, *nrel_else* и т.д.

4.3.3. *Протокол выполнения действия*

Каждый ***протокол*** представляет собой *семантическую окрестность*, *ключевым sc-элементом'* является *действие*, для которого собственно описывается весь процесс его выполнения, то есть все более простые поддействия, в том числе те, выполнение которых, как выяснилось позже, не было целесообразным. Подразумевается, что *sc-элемент*, обозначающий данное *действие*, входит во множество *прошлых сущностей*.

Таким образом, ***протокол*** представляет собой *sc-текст*, содержащий декомпозицию рассматриваемого *действия* на поддействия с указанием порядка их выполнения, а также необходимой спецификацией каждого такого поддействия.

В отличие от *плана*, ***протокол*** всегда формируется по факту выполнения соответствующего *действия*.

4.3.4. *Решение задачи, соответствующей действию*

Каждое ***решение*** представляет собой *семантическую окрестность*, *ключевым sc-элементом'* является *действие*, для которого собственно описывается процесс его выполнения, то есть решение соответствующей задачи. Подразумевается, что *sc-элемент*, обозначающий данное *действие*, входит во множество *успешно выполненных действий*.

Таким образом, ***решение*** представляет собой *sc-текст*, содержащий декомпозицию рассматриваемого *действия* на поддействия с указанием порядка их выполнения, а также необходимой спецификацией каждого такого поддействия.

Стоит отметить, что в случае отношения ***решение**** в *декомпозиции действия** указываются только те поддействия, без которых решение поставленной задачи было бы невозможным, то есть из *протокола* исключаются ложные или избыточные шаги, проделанные в процессе поиска пути решения задачи, которые, в свою очередь, могут присутствовать при описании непосредственно текущего хода решения задачи.

Для конкретного *действия* его ***решение*** будет нестрого *включаться** в соответствующий *протокол* решения.

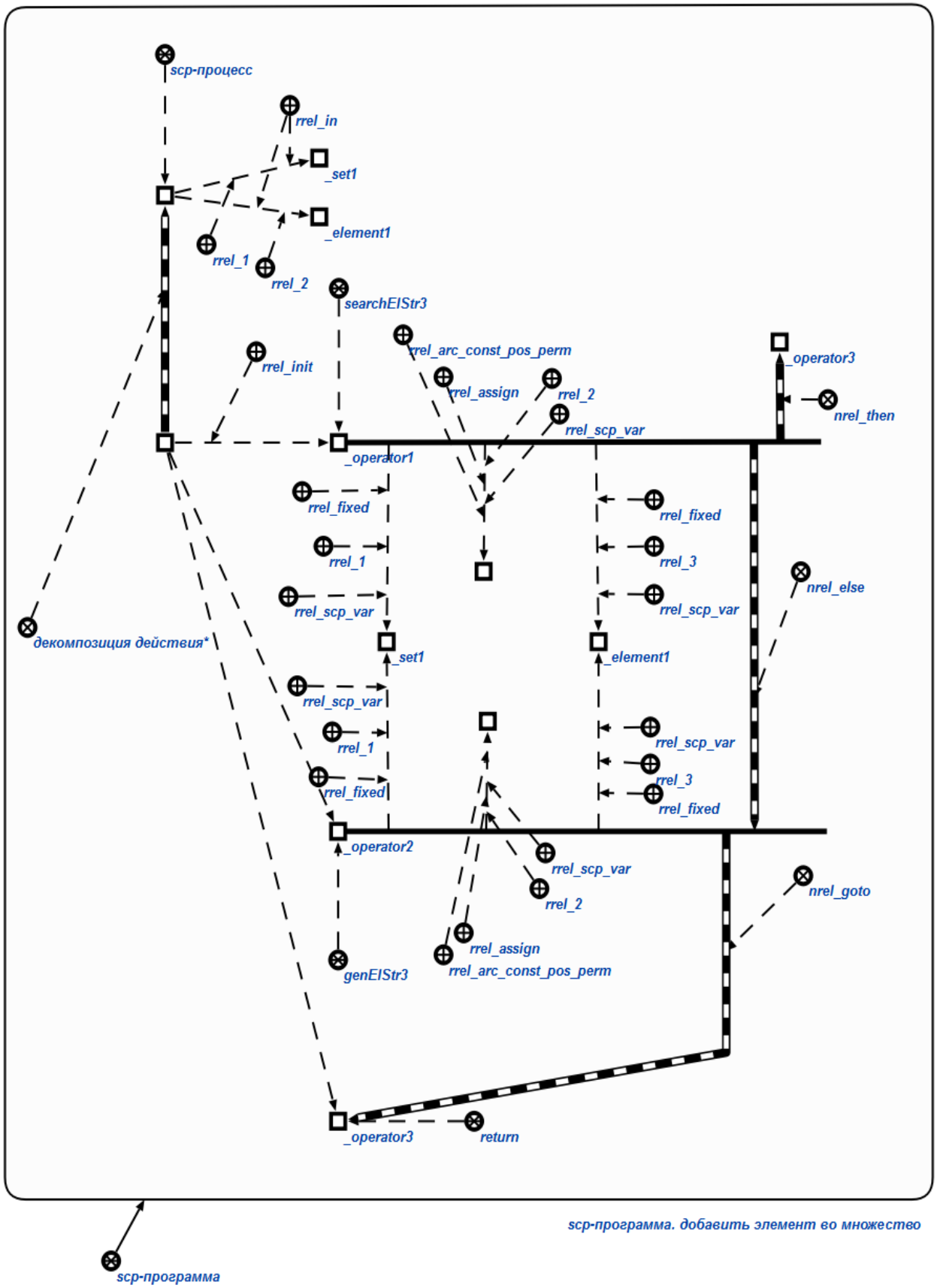

Рисунок 1 – Пример программы на языке SCP

Заключение

В работе рассмотрены базовые принципы описания целенаправленной деятельности различного вида субъектов на основе семантических сетей с базовой теоретико-множественной интерпретацией, применимые для описания в семантической памяти деятельности различного рода субъектов.

В работе представлены результаты, полученные в научных группах НИУ «МЭИ» (кафедра прикладной математики) и БГУИР (кафедра интеллектуальных информационных технологий) совместно.

Работа выполнена при поддержке гранта БРФФИ-РФФИ-М Ф15РМ-074 (номер с российской стороны - 15-57-04047 Бел-мол-а) «Методы и средства онтологического моделирования для семантических технологий проектирования интеллектуальных систем».

Библиографический список

[Boyko, 2016] Boyko, I. Semantic classification of actions for knowledge inference. Open semantic technologies for intelligent systems (OSTIS-2016): materials of VI International.sc.-tech.conf./ I. Boyko // Mn.: BSUIR, 2016

[IMS, 2016] Метасистема IMS.OSTIS [Электронный ресурс]. Минск, 2016. – Режим доступа: http://ims.ostis.net/. – Дата доступа: 15.01.2016.

[Голенков, 2015] Голенков, В.В. Семантическая технология компонентного проектирования систем, управляемых знаниями. Открытые семантические технологии проектирования интеллектуальных систем (OSTIS-2015): материалы V Междунар.научн.-техн.конф./ В. В. Голенков, Н.А Гулякина// Мн.: БГУИР, 2015

[Голенков, 2012] Голенков, В.В., Гулякина Н.А. Графодинамические модели параллельной обработки знаний: принципы построения, реализации и проектирования./В.В. Голенков, Н.А., Гулякина// Открытые семантические технологии проектирования интеллектуальных систем (OSTIS-2012): материалы Междунар. научн.-техн.конф. Мн.: БГУИР, 2012 – С.23-52

[Давыденко и др., 2016a] Давыденко, И.Т. Описание свойств различных видов знаний в системах, управляемых знаниями. Открытые семантические технологии проектирования интеллектуальных систем (OSTIS-2016): материалы VI Междунар.научн.-техн.конф./ И. Т. Давыденко [и др.] // Мн.: БГУИР, 2016

[Давыденко и др., 2016b] Давыденко, И.Т. Иерархическая структуризация баз знаний систем, управляемых знаниями. Открытые семантические технологии проектирования интеллектуальных систем (OSTIS-2016): материалы VI Междунар.научн.-техн.конф./ И. Т. Давыденко [и др.] // Мн.: БГУИР, 2016

[Мартынов, 1974] Мартынов, В.В. Семиологические основы информатики / В.В. Мартынов - Мн.: Наука и техника, 1974. – 192 с.

[Мартынов, 1977] Мартынов, В.В. Универсальный семантический код (Грамматика. Словарь. Тексты) / В.В. Мартынов – Мн.: Наука и техника, 1977. – 191 с.

[Мартынов, 1984] Мартынов, В.В. Универсальный семантический код: УСК-3 / В.В. Мартынов – Мн.: Наука и техника, 1984. – 132 с.

[Тарасов, 2002] Тарасов, В.Б. От многоагентных систем к интеллектуальным организациям / В.Б. Тарасов; – М. :Изд-во УРСС, 2002.

[Шенк, 1980] Шенк, Р. Обработка концептуальной информации. Пер. с англ. / Р. Шенк; – М.: Энергия; 1980.

[Шункевич, 2014] Шункевич, Д.В. Машина обработки знаний интеллектуальной метасистемы поддержки проектирования интеллектуальных систем. /Д.В. Шункевич// Открытые семантические технологии интеллектуальных систем (Ostis-2014): Материалы IV международной науч.-тех. конф -- Минск: БГУИР,2014. – с.93-96

FORMAL SEMANTIC DESCRIPTION OF THE VARIOS TYPES OF ACTORS PURPOSEFUL ACTION

Shunkevich D.V.[*], Hubarevich A.V.[*], Svyatkina M.N.[**], Morosin O.L.[***]

**Belarusian State University of Informatics and Radioelectronics, Minsk, Republic of Belarus*

shunkevichdv@gmail.com

st.hubarevich@gmail.com

***Bauman Moscow State Technical University, Moscow, Russia*

maria.svyatkina@gmail.com

****Moscow Power Engineering Institute*

oleg@morosin.ru

In this paper the formal models, that allowed to describe the varios types of actors purposeful action by the semantic networks with set-theoretic interpretation, are considered. Particularly, the tepology of actions, the consept of task as the action specification, relations, used for action specification are considered. At the same time taked up the other classes of action specification: plan, programm, solution, ect.

Key words: semantic technologies, the action formalisation, multy-agent systems, OSTIS progect.

Introduction

The aim of this work is to develop the basic principles of the description of the purposeful activity of various types of subjects based on semantic networks with set-theoretic interpretation. The models and instruments submitted in this work are used in the knowledge-driven systems, built on the OSTIS tecnology, development.

This work is a part of the knowledge base of the IMS OSTIS system, in which it is presented as the section, describing the relevant domain.

Main Part

This paper considered of the next sections:

- The basic typology of action:
  - The consept of action and the typology of it;
  - Types of action towards the information system's memory;
  - Types of action towards the current time;
  - Types of action atomicity;
- The typology of action subjects;
- Instruments of process of performing actions detalisation;
- Specification of actions:
  - Task as the action specification
  - Action specification relations
  - Other types of action specification