OSTIS-2015
(Open Semantic Technologies for Intelligent Systems)

УДК 004.822:514

МЕТОДИКА КОМПОНЕНТНОГО ПРОЕКТИРОВАНИЯ СИСТЕМ, УПРАВЛЯЕМЫХ ЗНАНИЯМИ

Шункевич Д.В., Давыденко И.Т., Корончик Д.Н., Губаревич А.В., Борискин А.С.

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

shunkevichdv@gmail.com

ir.davydenko@gmail.com

deniskoronchik@gmail.com

stasia@tut.by

coloss_000@mail.ru

В данной работе рассматривается методика проектирования систем, управляемых знаниями, по технологии OSTIS. Указанная технология является открытой технологией, ориентированной на разработку систем, управляемых знаниями, самого различного назначения. В работе подробно рассматривается процесс разработки систем на примере справочной системы по геометрии, описаны процессы расширения базы знаний, машины обработки знаний и пользовательского интерфейса системы.

Ключевые слова: системы, управляемых знаниями; семантические сети; базы знаний; интеллектуальный решатель задач.

Введение

В настоящее время особенно актуальной становится проблема отсутствия хорошо продуманной методики конструктивного использования опыта завершенных разработок компьютерных систем, что порождает высокую степень дублирования разработок различных компонентов этих систем. [Борисов 2013] Многократная повторная разработка уже имеющихся технических решений обусловлена либо тем, что известные технические решения плохо интегрируются в разрабатываемую систему, либо тем, что эти технические решения трудно найти. [Голенков 2012] Данная проблема актуальна как в целом в сфере разработки компьютерных систем, так и в сфере разработки систем, основанных на знаниях, поскольку в системах такого рода степень согласованности различных видов знаний влияет на возможность системы решать нетривиальные задачи. Одной из основных задач Технологии OSTIS [IMS] является решение проблемы совместимости различных компонентов и разработка средств компонентного проектирования систем, управляемых знаниями.

Для решения проблемы совместимости различных видов знаний предложены унифицированные семантические сети с теоретико-множественной интерпретацией.

При проектировании систем управляемых знаниями применяется методика компонентного проектирования, которая основывается на постоянно расширяемых библиотеках многократно используемых компонентов (типовых технических решений). В качестве примера использования такой методики в данной работы мы рассмотрим подробно процесс проектирования справочной системы по геометрии.

1. Подсистема IMS для поддержки компонентного проектирования дочерних sc-систем

В рамках интеллектуальной метасистемы IMS, ориентированной на консультационное обслуживание и поддержку разработчиков систем по Технологии OSTIS, а также накопление библиотек многократно используемых компонентов OSTIS выделяется подсистема для поддержки проектирования различных компонентов и различных классов интеллектуальных систем. Основными ее задачами являются:

- Обеспечение возможности поиска нужного компонента или набора компонентов в *Библиотеке*

многократно используемых компонентов Технологии OSTIS;

- Формирование, при необходимости, *sc-структур*, соответствующих выбранным многократно используемым компонентам;
- Обеспечение возможности транспортировки выбранных многократно используемых компонентов в *дочернюю sc-систему*;
- В случае развертывания стартовой версии *дочерней sc-системы* – обеспечение возможности установки первой базовой версии *sc-системы*, которая впоследствии может наращиваться дополнительными компонентами.

2. Этапы проектирования дочерних sc-систем

При создании стартовой версии *дочерней sc-системы* по *Технологии OSTIS* можно выделить четыре основных этапа:

- Выбор и установка платформы реализации *дочерней sc-системы*;
- Установка *Ядра sc-моделей баз знаний*, то есть набора базовых *многократно используемых компонентов sc-моделей баз знаний*, необходимых для работы даже первого прототипа *sc-системы*;
- Установка *Ядра sc-машин*, то есть набора базовых *многократно используемых компонентов sc-машин*, необходимых для работы даже первого прототипа *sc-системы*;
- Установка *Ядра sc-моделей интерфейсов*, то есть набора базовых *многократно используемых компонентов sc-моделей интерфейсов*, необходимых для работы даже первого прототипа *sc-системы*;
- Установка ядра подсистемы поддержки проектирования дочерней системы в составе проектируемой дочерней sc-системы.

После того, как собрана самая базовая конфигурация *дочерней sc-системы*, можно начинать постепенное ее наполнение либо с использованием *Библиотеки многократно используемых компонентов Технологии OSTIS*, либо путем создания своих собственных компонентов, которые, в свою очередь, могут в дальнейшем быть включены в состав указанной библиотеки.

3. Выбор и установка платформы реализации дочерней sc-системы

3.1. Выбор и установка *sc-хранилища*

В качестве платформы для разработки справочной системы по геометрии в данной работе выбрана *Web-ориентированная платформа реализации sc-систем*, основанная на специальном формате кодирования *sc-текстов*, входящая в состав *Библиотеки платформ реализации sc-систем* [IMS]. Согласно документации по данному *платформенно-зависимому многократно используемому компоненту OSTIS* [sc-machine] данный вариант реализации платформы ориентирован на операционные системы семейства Linux и сопровождается скриптами, позволяющими собрать и установить выбранную платформу на нужном сервере [Структура sc-машины].

Данный вариант реализации платформы включает в себя собственно *sc-хранилище*, а также набор трансляторов.

3.2. Установка scp-машины

После того, как на сервер была установлена выбранная реализация sc-памяти, разработчик может установить также совместимую с выбранной реализацией платформы реализацию абстрактной scp-машины. На семантическом уровне совместимость задается при помощи отношения совместимый компонент*. Таким образом, для выбранной реализации sc-памяти подойдет реализация абстрактной scp-машины на базе Web-ориентированной платформы реализации sc-систем. Исходные коды и инструкцию по установке данного платформенно-зависимого многократно используемого компонента OSTIS согласно его описанию в Библиотеке платформ реализации sc-моделей можно найти по адресу [scp-machine]. В данном варианте компонента для реализации sc-агентов используются языки C и C++.

Для корректной работы какого-либо варианта реализации абстрактной scp-машины необходимо наличие в системе соответствующего многократно используемого компонента sc-моделей баз знаний, описывающего Предметную область программ базового языка программирования, ориентированного на обработку sc-моделей баз знаний.

После того, как были установлены необходимые части платформы реализации sc-моделей, можно приступать к установке необходимых платформенно-независимых многократно используемых компонентов OSTIS.

4. Установка ядра базы знаний дочерней sc-системы

Ядро sc-моделей баз знаний рассматривается как один *неатомарный многократно используемый компонент OSTIS* и содержит в себе все фрагменты базы знаний, необходимые для корректной работы *платформы реализации sc-моделей* интеллектуальных систем, а также базовых *sc-агентов*, использование которых целесообразно в любой *дочерней sc-системе*.

Для понятности кратко перечислим фрагменты базы знаний, входящие в *Ядро sc-моделей баз знаний*:

- *Раздел. Предметная область семантических окрестностей;*
- *Раздел. Предметная область ситуативно-событийных моделей изменения состояний баз знаний;*

- *Раздел. Предметная область моделей действий, направленных на изменение состояния баз знаний;*
- *Раздел. Предметная область чисел;*
- *Раздел. Предметная область сообщений, принимаемых и передаваемых интеллектуальной системой;*
- *Раздел. SC-код*
  - *Раздел. Предметная область sc-элементов*
  - *Раздел. Предметная область sc-множеств*
  - *Раздел. Предметная область sc-отношений*
  - *Раздел. Предметная область библиографических источников*
  - *Раздел. Предметная область sc-структур*
  - *Раздел. Предметная область sc-ссылок*
  - *Раздел. Предметная область внешних sc-идентификаторов*
  - *Раздел. Предметная область физических лиц и коллективов*
- *Раздел. SCs-код*
- *Раздел. SCn-код*
- *Раздел. SCg-код*
- Базовый набор команд информационного поиска
  - *Команды запроса структуры*
    - *Запрос всех надклассов*
    - *Запрос декомпозиции*
    - *Запрос всех подклассов*
  - *Команды базовых запросов*
    - *Запрос позитивных, константных выходящих дуг с отношениями*
    - *Запрос позитивных, константных входящих дуг с отношениями*
    - *Запрос позитивных, константные выходящих дуг*
    - *Запрос позитивных, константных входящих дуг*
  - *Команды запроса семантических окрестностей*
    - *Запрос полной семантической окрестности*
  - *Команды запроса идентификаторов*
    - *Запрос всех идентификаторов*

5. Установка ядра машины обработки знаний дочерней sc-системы

Ядро машины обработки знаний *sc-систем* (*sc-машин*) включает в себя минимальный набор предметно независимых *sc-агентов информационного поиска*, необходимый для навигации по базе знаний *дочерней sc-системы*.

В текущем варианте указанное ядро включает в себя:

- *sc-агент поиска всех выходящих константных позитивных стационарных sc-дуг принадлежности;*
- *sc-агент поиска всех выходящих константных позитивных стационарных sc-дуг принадлежности с их ролевыми отношениями;*
- *sc-агент поиска всех входящих константных позитивных стационарных sc-дуг принадлежности;*
- *sc-агент поиска всех выходящих константных позитивных стационарных sc-дуг принадлежности с их ролевыми отношениями;*
- *sc-агент поиска связок декомпозиции для заданного sc-элемента;*
- *sc-агент поиска всех известных сущностей, являющихся частными по отношению к заданной;*
- *sc-агент поиска всех известных сущностей, являющихся общими по отношению к заданной;*
- *sc-агент поиска всех идентификаторов заданного sc-элемента;*
- *sc-агент поиска полной семантической окрестности заданного элемента;*
- *sc-агент поиска связок заданного отношения, компонентом которых является заданный sc-элемент;*
- *sc-агент поиска всех конструкций, изоморфных заданному образцу;*

Данный набор *sc-агентов* может быть реализован как на платформенно-независимом уровне, так и на уровне платформы. В нашем случае реализация всех перечисленных *sc-агентов* входит в состав выбранной *платформы реализации sc-моделей*, что, с одной стороны, упрощает их установку в *дочернюю sc-систему*, но с другой сильно осложняет процесс внесения изменений в состав ядра.

Чтобы обеспечить возможность задания более сложных поисковых запросов в систему необходимо добавить дополнительные *sc-агенты* из *Библиотеки многократно используемых sc-агентов OSTIS*, о чем будет сказано ниже.

Для корректной работы данных *sc-агентов* дополнительно необходимо наличие в базе знаний определенного набора *сопутствующих компонентов**.

6. Установка ядра пользовательского интерфейса дочерней sc-системы

Ядро sc-моделей пользовательских интерфейсов, как и собственно *платформа реализации sc-моделей*, представляет собой *платформенно-зависимый многократно используемый компонент OSTIS*, который должен быть совместим с выбранным вариантом платформы. Подходящий вариант реализации ядра пользовательских интерфейсов, описанный в *Библиотеке многократно используемых компонентов sc-моделей интерфейсов* может быть найден по адресу [sc-web]. Согласно описанию, выбранный компонент содержит в себе, помимо собственно ядра, следующий набор компонентов:

- Визуализатор *sc-конструкций* в *SCg-коде;*
- Редактор *sc-конструкций* в *SCg-коде;*
- Визуализатор *sc-конструкций* в *SCn-коде;*
- Набор компонентов для отображения *sc-ссылок* различных форматов (PNG, GIF, PDF, HTML, TXT и др.).

Также в выбранный ранее вариант *платформы реализации sc-моделей* по умолчанию входят

трансляторы из *sc-памяти* в формат, необходимый для SCg и SCn визуализации, а также транслятор из SCg-редактора в *sc-память*. Таким образом, установленных средств достаточно для базовой визуализации и редактирования *sc-конструкций* посредством web-интерфейса.

7. Установка ядра подсистемы поддержки проектирования дочерней sc-системы

Подсистема поддержки проектирования *дочерней sc-системы* в свою очередь состоит из

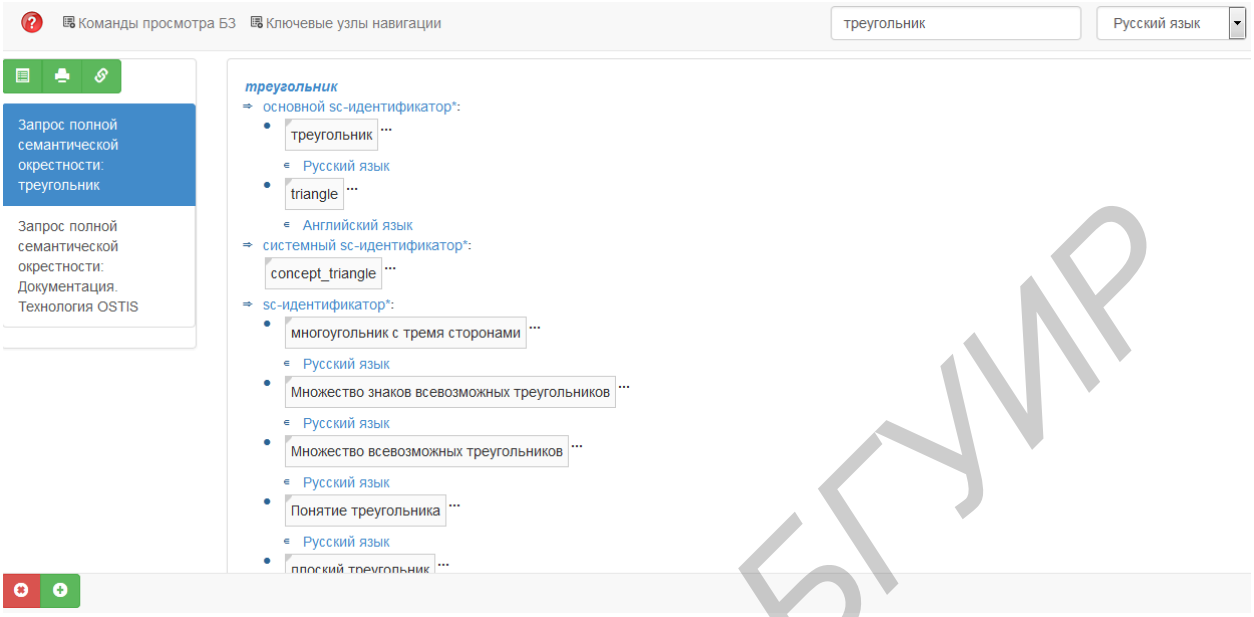

Рисунок 1 – Отображение фрагмента базы знаний в SCn-коде

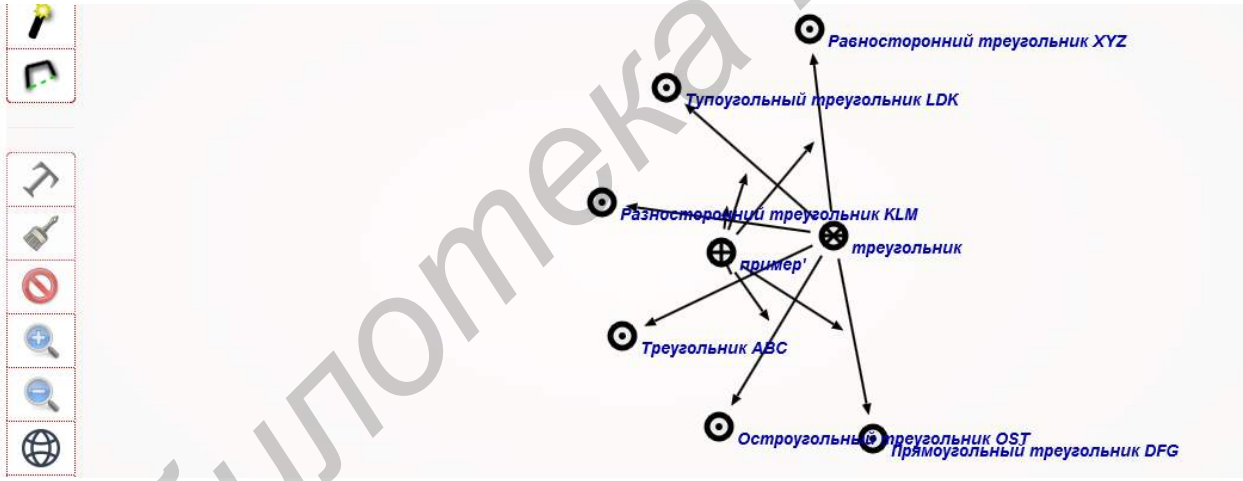

Рисунок 2 – Отображение фрагмента базы знаний в SCg-коде

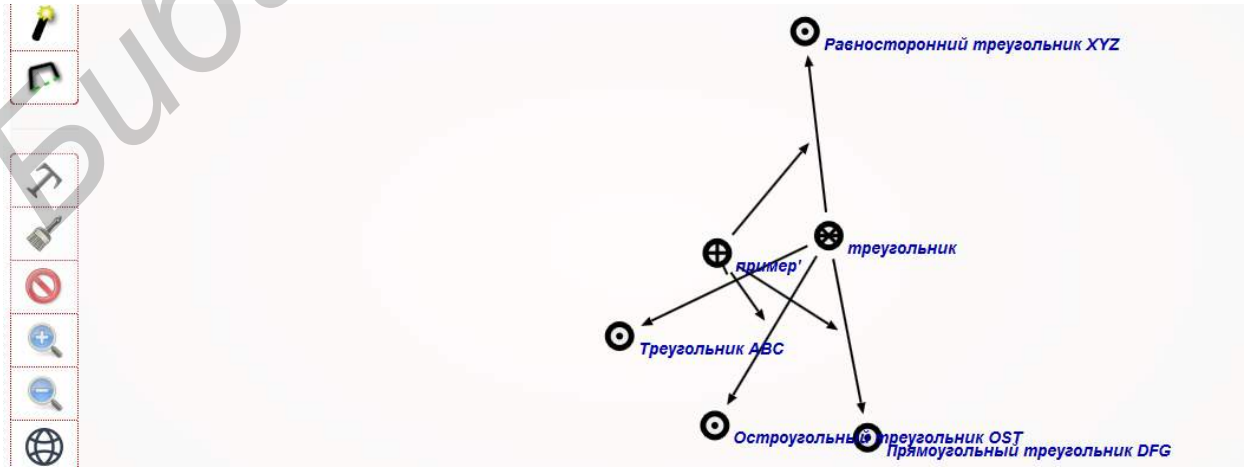

Рисунок 3 – Результат редактирования фрагмента базы знаний в SCg-коде

двух взаимосвязанных подсистем, одна их которых входит в состав самой *IMS*, а другая является *многократно используемой типовой подсистемой интеллектуальных систем*, и входит в состав минимального набора компонентов, необходимых для полноценного функционирования *дочерней sc-системы* в рамках *Глобальной базы знаний*. [Шункевич, 2015]

Ядро данной типовой подсистемы включает в себя необходимые средства интеграции *многократно используемых компонентов OSTIS* в *дочернюю sc-систему* и средства управления проектированием *дочерней sc-системы*.

8. Расширение базы знаний дочерней sc-системы

Помимо базовых многократно используемых компонентов баз знаний, входящих в состав ядра базы знаний, каждая дочерняя система может быть дополнена другими компонентами из *Библиотеки многократно используемых компонентов sc-моделей баз знаний*. Общие принципы проектирования баз знаний по *Технологии OSTIS* рассмотрены в [Давыденко, 2013]

8.1. Построение системы предметных областей и их онтологий, входящих в состав базы знаний проектируемой sc-системы

Первым и важнейшим этапом проектирования базы знаний является уточнение структуры описываемой предметной области или нескольких взаимосвязанных предметных областей. Уточнение такой структуры – это, прежде всего, уточнение класса исследуемых объектов, уточнение предмета исследования, уточнение всего семейства ключевых узлов семантической сети, представляющей предметную область. В рамках предметной области возможно выделение частных предметных областей на основе выделения подмножества из семейства классов исследуемых объектов.

При описании структуры предметной области используются ключевые узлы, входящие в состав *Предметной области предметных областей*, входящей в *Библиотеку многократно используемых компонентов баз знаний* в виде компонента, следовательно, данный компонент необходимо добавить в состав базы знаний системы по геометрии.

Пример структуры *Предметной области Геометрии Евклида*:

Предметная область Геометрии Евклида
∈ предметная область
*=> частная предметная область**:
- *Предметная область геометрических точек*
- *Предметная область линий*
- *Предметная область планарных фигур*
  *=> частная предметная область**:
  - *Предметная область прямолинейных фигур*
  - *Предметная область планарных углов*
  - *Предметная область многоугольников*
    *=> частная предметная область**:
    - *Предметная область треугольников*
    - *Предметная область четырехугольников*
  - *Предметная область вписанных планарных фигур*
  - *Предметная область кругов и окружностей*
- *Предметная область геометрических поверхностей*
- *Предметная область геометрических тел*
  *=> частная предметная область**:
  - *Предметная область многогранников и их поверхностей*
  - *Предметная область непланарных углов*
  - *Предметная область тел вращения и их поверхностей*
- *Предметная область конгруэнтности геометрических фигур*

Рассмотрение базы знаний с позиции ее соотношения с предметной областью позволяет рассматривать исследуемые объекты на различных уровнях детализации, которые отражаются в различных типах онтологий, описывающих определенное направление описания свойств объекта в рамках рассматриваемой предметной области. К таким типам таких онтологий относятся:

- *структурная спецификация предметной области* – описание связей рассматриваемой предметной области с другими предметными областями и ролей всех понятий, входящих в состав данной предметной области;
- *терминологическая онтология* – описание терминов и их синонимов ключевых понятий рассматриваемой предметной области, близких между собой терминов, этимологии терминов и правила построения идентификаторов экземпляров понятий;
- *теоретико-множественная онтология* – описание теоретико-множественных связей между понятиями рассматриваемой предметной области;
- *логическая онтология* – описание всех высказываний рассматриваемой предметной области;
- *логическая система понятий и их определений* – это структура, являющаяся надстройкой над логической онтологией, включающая описание системы определений понятий заданной предметной области с указанием набора понятий, через которые определяется каждое определяемое понятие рассматриваемой предметной области;
- *логическая система утверждений и их доказательств* – это структура, являющаяся

надстройкой над логической онтологией, включающая описание системы утверждений рассматриваемой предметной области с указанием набора утверждений, через которые доказывается каждое утверждение;

- *онтология задач и решений задач* – описание конкретных задач, рассматриваемых в заданной предметной области, и их решений;
- *онтология классов задач и способов решения задач* – описание классов задач, рассматриваемых в заданной предметной области, и способов их решений. Данная структура является надстройкой над *онтологией задач и решений задач*.

Для описания указанной структуры предметной области используются ключевые *sc-элементы*, входящие в состав *многократно используемого компонента sc-моделей баз знаний Предметная область онтологий*.

Таким образом, для всех разделов базы знаний, описывающих предметную область и ее спецификацию, имеет смысл задать типовую структуру. В качестве примера рассмотрим *Предметную область четырехугольников и ее онтологии*.

Раздел. Предметная область четырехугольников
∈ предметная область и ее онтологии
<= декомпозиция раздела:*
{

- *Предметная область четырехугольников*
- *Структурная спецификация предметной области четырехугольников*
- *Терминологическая онтология предметной области четырехугольников*
- *Теоретико-множественная онтология предметной области четырехугольников*
- *Логическая онтология предметной области четырехугольников*
- *Логическая иерархия понятий предметной области четырехугольников*
- *Логическая иерархия высказываний о предметной области четырехугольников*
- *Онтология задач и решений задач предметной области четырехугольников*
- *Онтология классов задач и способов решения задач предметной области четырехугольников*

}

Далее рассмотрим подробнее фрагменты указанных выше разделов спецификации предметной области.

8.2. Разработка фрагментов используемых предметных областей

Любая предметная область также может содержать примеры конкретных объектов исследования, т.е. экземпляры классов исследования.

Ниже приведен пример отображения в системе семантической окрестности объекта Параллелограмм ABCD.

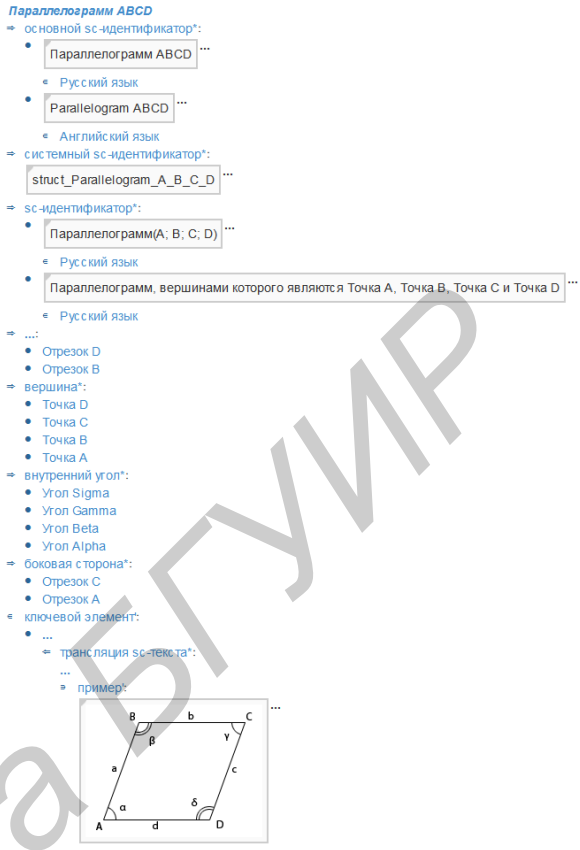

Рисунок 4 – Семантическая окрестность Параллелограмма ABCD

8.3. Разработка структурных спецификаций для каждой используемой предметной области

Рассмотрим фрагмент *Структурной спецификации предметной области четырехугольников:*

Предметная область четырехугольников
=> sc-онтология:*
Структурная спецификация предметной области четырехугольников
=[
Предметная область четырехугольников
<= частная предметная область:*
Предметная область многоугольников
∋ максимальный класс объектов исследования':
четырехугольник
∋ немаксимальный класс объектов исследования':

- *параллелограмм*
- *прямоугольник*
- *ромб*
- *квадрат*
- *трапеция*
- *равнобедренная трапеция*
- *прямоугольная трапеция*

∋ объект исследования':

- *Квадрат ABCD*

- *Четырехугольник KLMN*
- *Трапеция BMNO*

∋ *исследуемое отношение'*:

- *внутренний угол**
- *сторона**
- *высота**
- *площадь**
- *периметр**
- *средняя линия**

]

Наличие указанного фрагмента в базе знаний позволяет задавать следующие вопросы sc-системе:

- Что является объектом исследования данной предметной области?
- Какие отношения исследуются в данной предметной области?

8.4. Разработка терминологических онтологий для каждой используемой предметной области

Рассмотрим фрагмент *Терминологической онтологии предметной области четырехугольников:*

Предметная область четырехугольников
*=> sc-онтология**:
Терминологическая онтология предметной области четырехугольников
=[

четырехугольник
= многоугольник с четырьмя сторонами
= многоугольник с четырьмя углами0
*<= правила идентификации элементов**:
{

- [*Четырехугольник ⋖ суффикс ⋗*]
- [*Чет-к ⋖ суффикс ⋗*]
- [*Четырехугольник (⋖идентификатор точки⋗; ⋖идентификатор точки⋗; ⋖идентификатор точки⋗, ⋖идентификатор точки⋗)*]

}

прямоугольник
= четырехугольник с прямыми внутренними углами
*<= правила идентификации элементов**:
{

- [*Прямоугольник ⋖ суффикс⋗*]
- [*Прям-к ⋖ суффикс ⋗*]
- [*Прямоугольник (⋖идентификатор точки⋗; ⋖идентификатор точки⋗; ⋖идентификатор точки⋗, ⋖идентификатор точки⋗)*]

}

ромб
= четырехугольник с равными сторонами
=четырехугольник с конгруэнтными сторонами
*<= правила идентификации элементов**:
{

- [*Ромб ⋖ суффикс ⋗*]
- [*Ромб (⋖идентификатор точки⋗; ⋖идентификатор точки⋗; ⋖идентификатор точки⋗, ⋖идентификатор точки⋗)*]

}

квадрат
= ромб с прямыми углами
= прямоугольник с равными сторонами
= прямоугольник ∩ ромб
*<= правила идентификации элементов**:
{

- [*Квадрат ⋖ суффикс ⋗*]
- [*Квадрат (⋖идентификатор точки⋗; ⋖идентификатор точки⋗; ⋖идентификатор точки⋗, ⋖идентификатор точки⋗)*]

}

трапеция
*<= правила идентификации элементов**:
{

- [*Трапеция ⋖ суффикс ⋗*]
- [*Трапеция (⋖идентификатор точки⋗; ⋖идентификатор точки⋗; ⋖идентификатор точки⋗, ⋖идентификатор точки⋗)*]

}

равнобедренная трапеция
= трапеция с равными сторонами

прямоугольная трапеция
= трапеция с прямыми углами при боковой стороне

]

Наличие указанного фрагмента в базе знаний позволяет задавать следующие вопросы sc-системе:

- Какие синонимичные термины известны для заданного термина?
- По какому правилу идентифицируются элементы заданного множества?

8.5. Разработка теоретико-множественных онтологий для каждой используемой предметной области

Рассмотрим Фрагмент *Теоретико-множественной онтологии предметной области четырехугольников*:

Предметная область четырехугольников
*=> sc-онтология**:
Теоретико-множественная онтология предметной области четырехугольников
=[

четырехугольник
⊃ трапеция

трапеция
⊃ параллелограмм
⊃ равнобедренная трапеция

⊃ *прямоугольная трапеция*

параллелограмм
⊃*прямоугольник*
⊃*ромб*

квадрат
⊂ *прямоугольник*
⊂ *ромб*

средняя линия*
∈ *бинарное отношение*
=> *область определения**:
трапеция ∪ *отрезок*
=> *первый домен**:
трапеция
=> *второй домен**:
отрезок
]

Наличие указанного фрагмента в базе знаний позволяет задавать следующие вопросы sc-системе:

- Как классифицируется заданное понятие?
- Какие надклассы известны для заданного понятия?
- Являются ли два заданных понятия пересекающимися?
- Какова область определения заданного отношения?

8.6. Разработка логических онтологий для каждой используемой предметной области

Рассмотрим фрагмент *Логической онтологии предметной области четырехугольников*

Предметная область четырехугольников
=> *sc-онтология**:
Логическая онтология предметной области четырехугольников
=[
четырехугольник
∈ *ключевой sc-элемент'*:
Опр.(четырехугольник)
∈ *sc-определение*
<= *трансляция sc-текста**:
…
∋ *пример'*:
[***четырехугольник*** – это многоугольник с четырьмя сторонами.]
∈ *Русский язык*
…
∈ *sc-утверждение*
<= *трансляция sc-текста**:
…
∋ *пример'*:
[Сумма мер внутренних углов четырехугольника равна 360 угловых градусов.]
∈ *Русский язык*
…
∈ *sc-утверждение*
<= *трансляция sc-текста**:
…
∋ *пример'*:
[Сумма квадратов длин диагоналей четырёхугольника равна удвоенной сумме длин квадратов отрезков, соединяющих середины его противолежащих сторон.]
∈ *Русский язык*
…
∈ *sc-утверждение*
<= *трансляция sc-текста**:
…
∋ *пример'*:
[Если около четырёхугольника можно описать окружность, то сумма мер его любых противолежащих углов равна 180 угловых градусов.]
∈ *Русский язык*
]

Наличие указанного фрагмента в базе знаний позволяет задавать следующие вопросы sc-системе:

- Какие высказывания известны в рамках заданной предметной области?
- Какие высказывания являются аксиомами в рамках заданной предметной области?
- Как определяется/поясняется то или иное понятие?

8.7. Разработка логической системы понятий и их определений для каждой используемой предметной области

Рассмотрим фрагмент *Логической системы понятий и их определений предметной области четырехугольников*

Логическая онтология предметной области четырехугольников
=> *метазнание**:
Логическая система понятий и их определений предметной области четырехугольников
=[
четырехугольник
∈ *ключевой sc-элемент'*:
…
∈ *sc-определение*
<= *трансляция sc-текста**:
…
∋ *пример'*:
[***четырехугольник*** – это многоугольник с четырьмя сторонами.]
∈ *Русский язык*
<= *используемые константы**:
{
- *многоугольник*
- *сторона**
}

трапеция
∈ *ключевой sc-элемент'*:
…

∈ *sc-определение*
<= *трансляция sc-текста**:
…
∋ *пример'*:
[***трапеция*** – это четырехугольник, у которого две противолежащие стороны параллельны.]
∈ *Русский язык*
<= *используемые константы**:
{
- *четырехугольник*
- *сторона**
- *параллельность**
- *противолежащий**
}
]

Наличие указанного фрагмента в базе знаний позволяет задавать следующие вопросы sc-системе:

- Через какие понятия определяется заданное понятие?
- Какие понятия определяются на основе заданного понятия?

8.8. Разработка логической системы утверждений для каждой используемой предметной области

В разрабатываемой системе по геометрии уместно привести типовые доказательства для некоторых теорем и типовые решения для некоторых примеров задач.

Для обеспечения такой возможности необходимо добавить в систему описание *Предметной области вопросов и задач*, содержащей все необходимые ключевые узлы, то есть *Раздел. Предметная область вопросов и задач*.

Рассмотрим фрагмент *Логической системы утверждений и их доказательств предметной области четырехугольников*

Логическая онтология предметной области четырехугольников
=> *метазнание**:
Логическая система высказываний и их доказательств предметной области четырехугольников
=[
параллелограмм
∈ *ключевой sc-элемент'*:
Утв.(параллелограмм; диагональ; четырехугольник)*
∈ *sc-утверждение*
<= *трансляция sc-текста**:
…
∋ *пример'*:
[Если диагонали четырехугольника пересекаются и точкой пересечения делятся пополам, то этот четырехугольник - параллелограмм]
∈ *Русский язык*
∋ *главный ключевой sc-элемент'*:
…
=> *основное доказательство**:
Док-во.Утв.(параллелограмм; диагональ; четырехугольник)*
<= *используемые утверждения**:
{
- *Утв.(вертикальные углы*; конгруэнтность*)*
- *Утв.(треугольник; конгруэнтность*; сторона*; внутренний угол*)*
- *Опр.(внутренние накрест лежащие углы*)*
- *Опр.(параллелограмм)*
}
]

Теперь, используя упомянутые ключевые узлы, добавим в систему пример доказательства теоремы о равенстве вертикальных углов:

Рисунок 5 – Пример отображения sc-текста общей структуры доказательства, представленного в SCn-коде

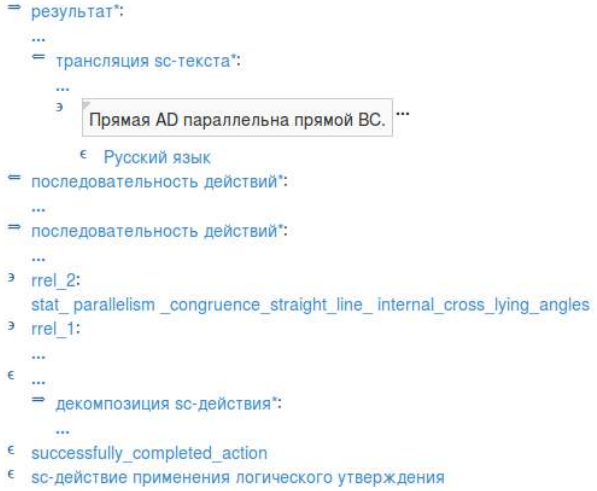

Рисунок 6 – Пример отображения sc-текста шага доказательства

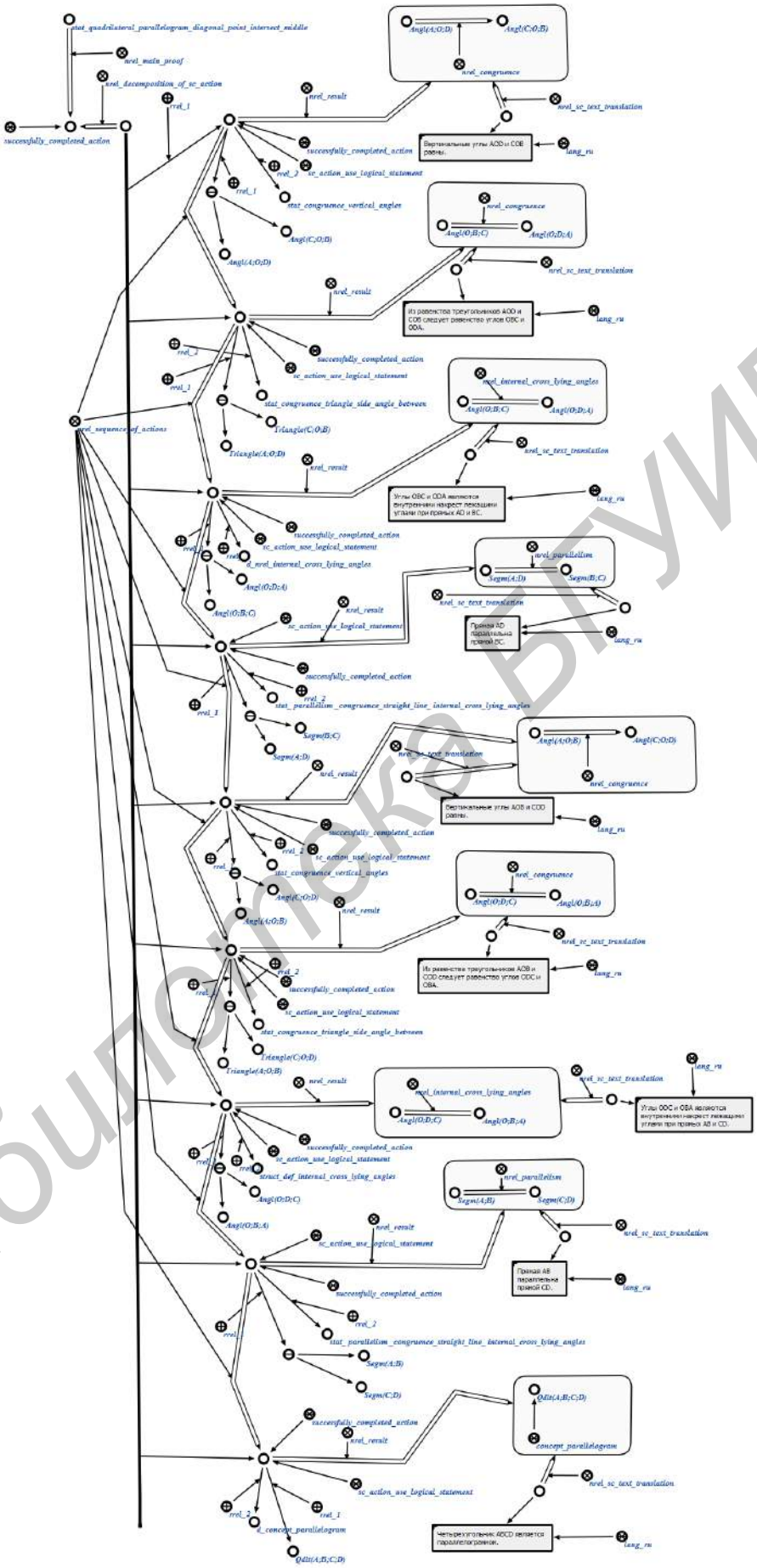

Рисунок 7 – Пример sc-текста доказательства, представленного в SCg-коде

Внесенное в систему доказательство можно просмотреть при помощи базовых средств навигации, как показано на рисунках выше, однако это не всегда удобно, поэтому имеет смысл использовать специальный *sc-агент поиска доказательства заданного утверждения*, о чем будет сказано ниже.

8.9. Разработка онтологий задач и решений для каждой используемой предметной области

Аналогичным образом можно внести в систему пример решения, например, следующей задачи: «В параллелограмме ABCD sin(C) = 5/7. AD = 7. Найдите высоту, опущенную на сторону AB».

Ниже приведет фрагмент *Онтологии задач и решений задач предметной области четырехугольников.*

Предметная область четырехугольников
=> *sc-онтология**:
Онтология задач и решений задач предметной области четырехугольников
=[
параллелограмм
∈ *ключевой sc-элемент'*:
Задача. Нахождение высоты параллелограмма по длине стороны и синусу внутреннего угла
∈ *sc-задача*
<= *трансляция sc-текста**:
…
∋ *пример'*:
[В параллелограмме ABCD sin(C) = 5/7. AD = 7. Найдите высоту, опущенную на сторону AB]
∈ *Русский язык*
=> *решение**:
Решение. Задача. Нахождение высоты параллелограмма по длине стороны и синусу внутреннего угла
<= *используемые утверждения**:
{
• *Утв.(параллелограмм; противолежащий*; внутренний угол*;конгруэнтность*)*
• *Опр.(синус*)*
}
]

Наличие указанного фрагмента в базе знаний позволяет задавать следующие вопросы sc-системе:

- Примеры каких задач известны в рамках заданной предметной области?
- Как решается заданной задача?
- С помощью каких утверждений решается заданная задача?

Формальная запись условия этой задачи на языке SCg представлена на рисунке 8.

Предлагаемый вариант решения данной задачи состоит из следующих шагов:

- $\sin(A) = \sin(C) = \frac{5}{7}$;
- $\sin(A) = \frac{5}{7} = \frac{DH}{AD} = \frac{DH}{7}$. $DH = 5$.

Решение по шагам в формальном представлено на рисунке 9.

В системе решение можно просмотреть по шагам при помощи базовых *sc-агентов информационного поиска*, однако, как и в случае с доказательствами, это не всегда удобно, поэтому имеет смысл добавить в систему *sc-агент поиска решения задачи*, позволяющий просмотреть все решение сразу.

8.10. Разработка онтологий классов задач и способов решения задач для каждой используемой предметной области

Рассмотрим фрагмент *Онтологии задач и решений задач предметной области четырехугольников.*

Онтология задач и решений задач предметной области четырехугольников
=> *метазнание**:
Онтология классов задач и способов решения задач предметной области четырехугольников
=[
Задача. Нахождение высоты параллелограмма по длине стороны и синусу внутреннего угла
∈ *sc-задача поиска заданной величины*
∈ *sc-задача на параллелограммы*
∈ *sc-задача на четырехугольники*
∈ *sc-задача планиметрии*
]

Наличие указанного фрагмента в базе знаний позволяет задавать следующие вопросы sc-системе:

- Какие классы задач известны для заданной предметной области?
- Какие способы решения класса задач известны для заданной предметной области?

Таким образом, база знаний *дочерней sc-системы* может быть пополнена как за счет использования компонентов из *Библиотеки многократно используемых компонентов sc-моделей баз знаний*, так и за счет добавления разработчиком в систему предметных знаний из рассматриваемой области.

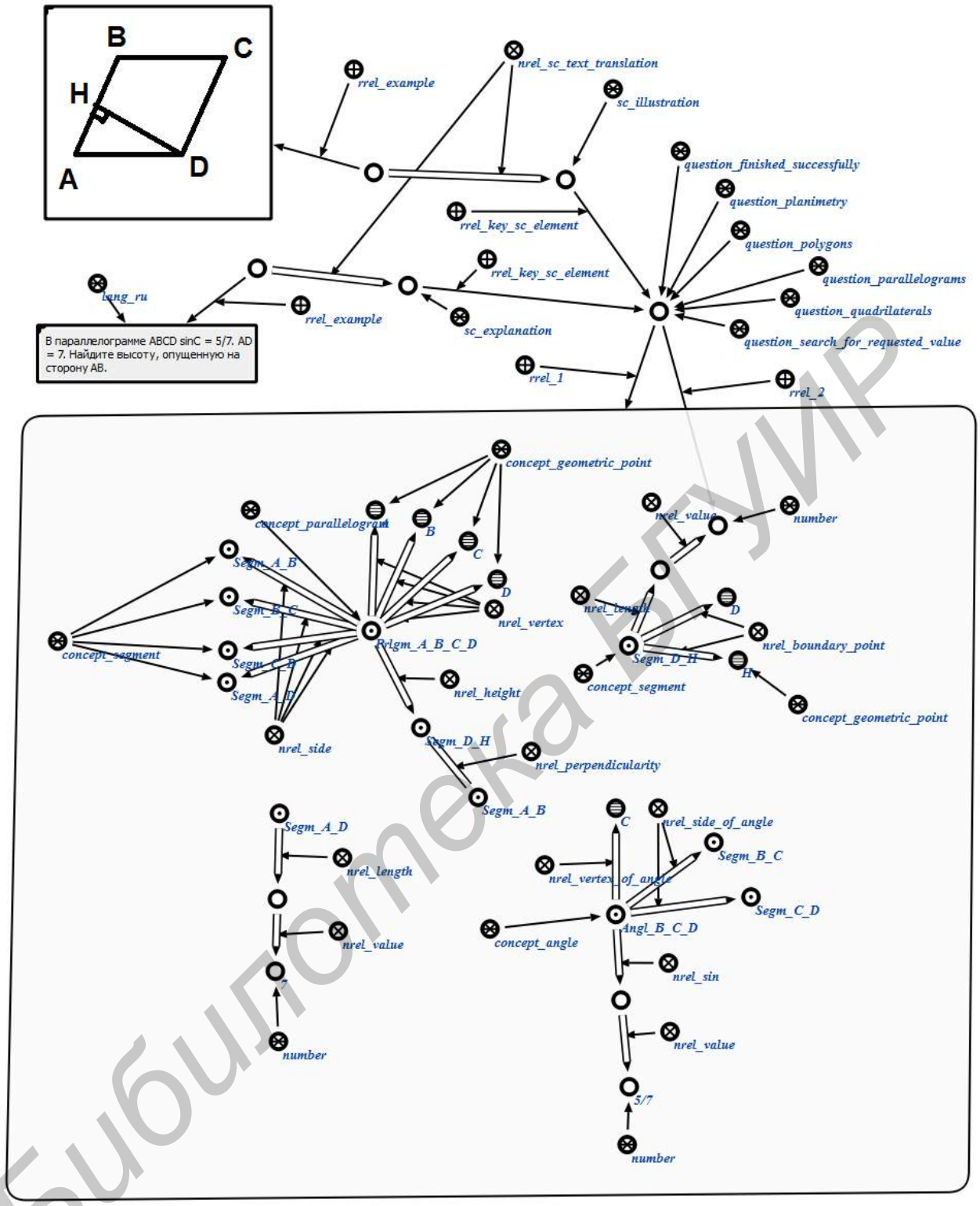

Рисунок 8 – Условие задачи, представленное в SCg-коде

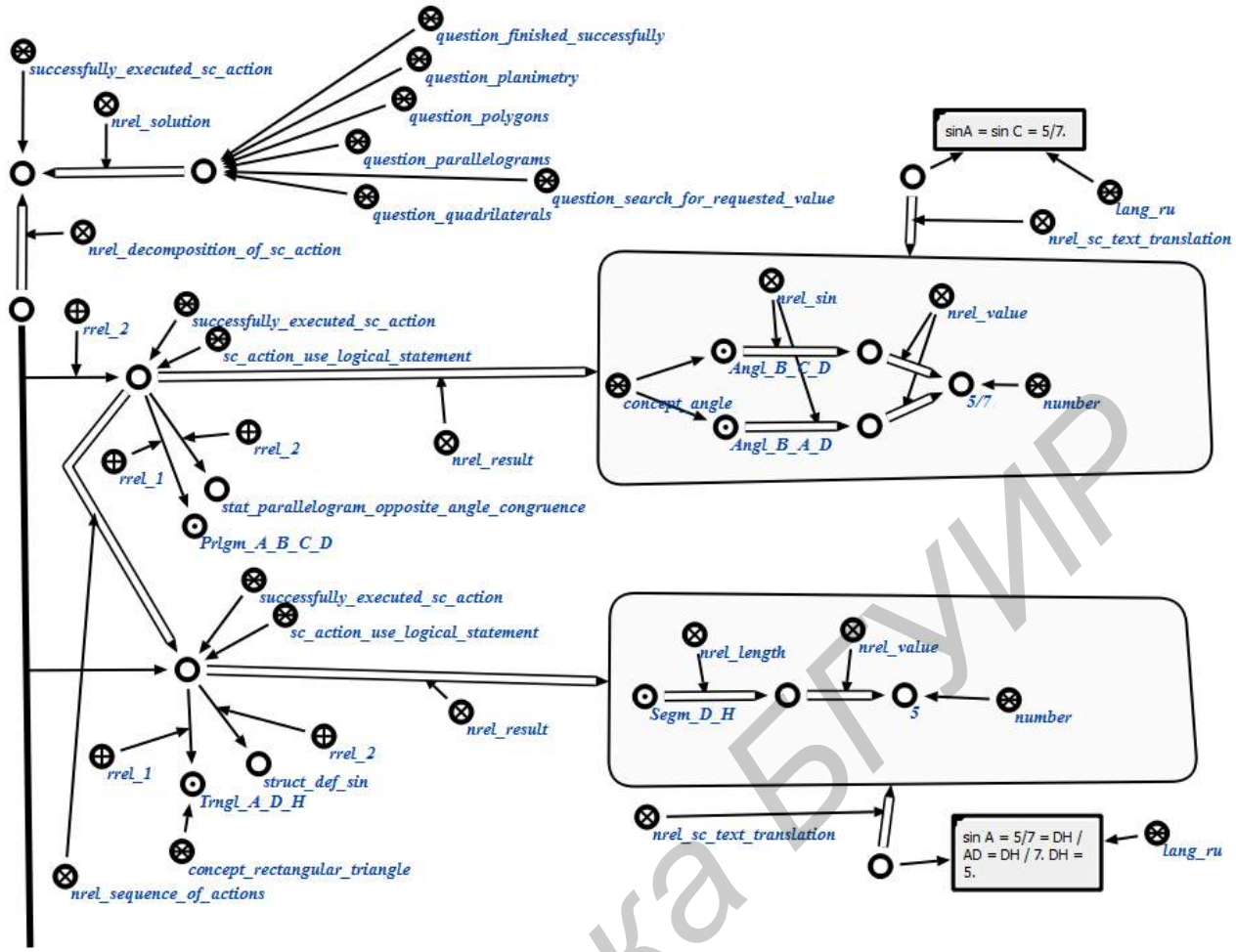

Рисунок 9 – Решение задачи, представленное в SCg-коде

9. Расширение машины обработки знаний дочерней sc-системы

Машина обработки знаний любой системы, управляемой знаниями, определяет ее функциональные возможности и во многом обеспечивает удобство работы с ее контентом. Общие принципы проектирования машин обработки знаний по *Технологии OSTIS* рассмотрены в [Шункевич, 2013].

9.1. Расширение машины обработки знаний с использованием многократно используемых компонентов OSTIS

Как было сказано ранее, в справочной системе по геометрии, например, представлены типовые доказательства некоторых теорем и решения некоторых задач. Для удобства навигации по знаниям такого рода разработаны специальные sc-агенты и описаны соответствующие им компоненты в *Библиотеке многократно используемых компонентов sc-машин*:

- *Компонент библиотеки. sc-агент поиска доказательства для заданного утверждения*;
- *Компонент библиотеки. sc-агент поиска решения заданной задач*;

Для указанных компонентов *сопутствующими компонентами** являются:

- *Компонент библиотеки. Команда поиска доказательства для заданного утверждения*;
- *Компонент библиотеки. Команда поиска решения заданной задач*;

Также *сопутствующим компонентом** для указанных является *Раздел. Предметная область вопросов и задач*, но данный компонент уже был был внедрен в систему ранее.

После того, как в систему были добавлены указанные компоненты, конечный пользователь системы получил возможность увидеть полный *sc-текст доказательства* выбранного утверждения или решения выбранной задачи.

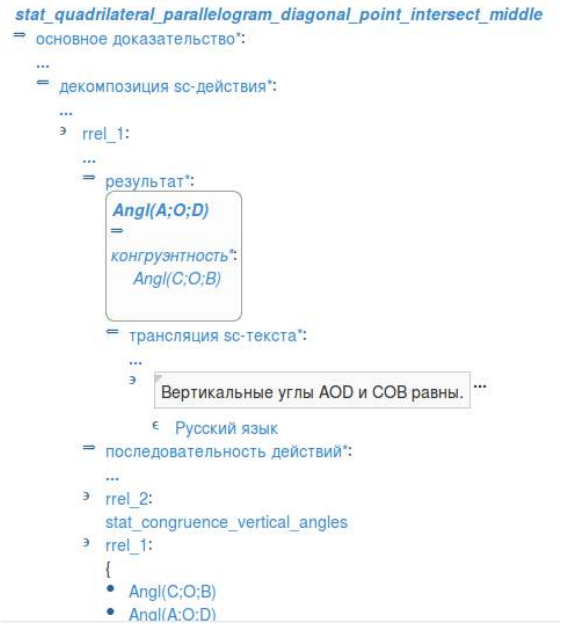

Рисунок 10 – Фрагмент результата работы *sc-агента поиска доказательства*

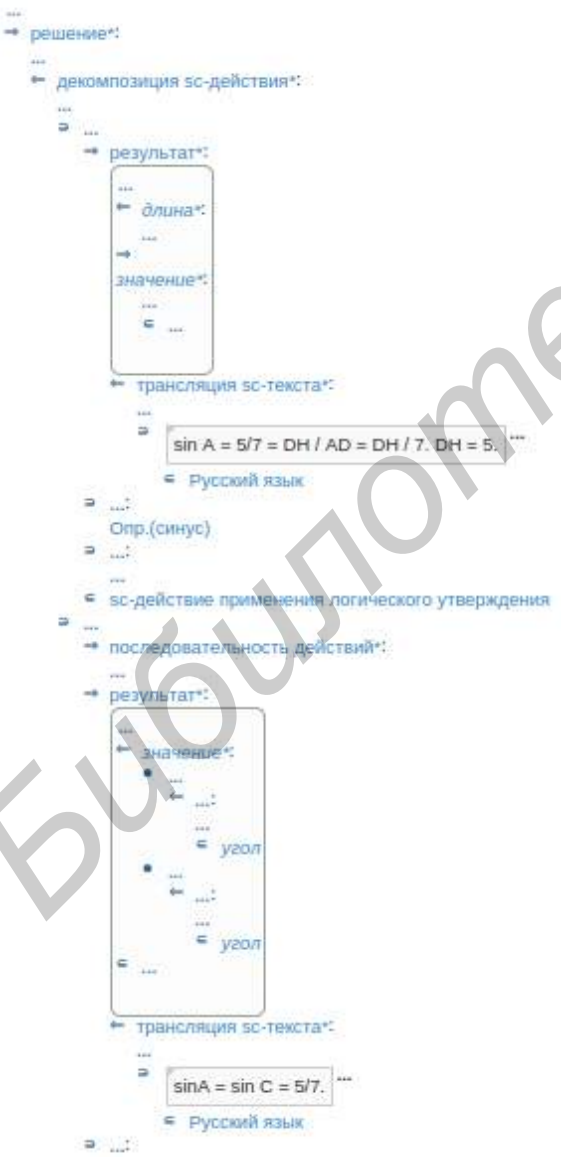

Рисунок 11 – Фрагмент результата работы *sc-агента поиска решения задачи*

9.2. Расширение машины обработки знаний на основе собственных разработок

В процессе разработки конкретной дочерней системы по *Технологии OSTIS* может оказаться, что необходимый компонент такой системы, хотя и может использоваться в целом ряде систем, но пока еще не реализован другими разработчиками или реализован, но по каким-либо причинам пока еще не вошел в состав *Библиотеки многократно используемых компонентов OSTIS*. В таком случае разработчик *дочерней sc-системы* может самостоятельно разработать нужный компонент и, поскольку *Технология OSTIS* является открытой технологией, попробовать внедрить его в состав соответствующей *библиотеки многократно используемых компонентов*.

Одним из важнейших достоинств интеллектуальной справочной системы по геометрии, является то, что она строи ответы на заданные пользователем вопросы (задачи) не только путем поиска и локализации (выделения) этих ответов в рамках текущего состояния базы знаний, но и путем генерации этих ответов, если их нет в текущем состоянии базы знаний. Интеллектуальная справочная система по геометрии, например, способна решать различные вычислительные задачи и задачи на доказательство. Для этих целей был разработан интеллектуальный решатель задач, который, однако, разрабатывался без привязки к конкретной предметной области и может войти в состав *Библиотеки типовых подсистем интеллектуальных систем*.

В процессе решения конкретной задачи решателем были выделены **6 ключевых этапов:**

- этап задания условия геометрической задачи;
- этап работы поисковых операций

  а) поиск нахождения значения указанной величины;

  б) поиск нахождения доказательства для указанного утверждения;

- этап аналогии;
- этап применения стратегий решения задач:

  а) «итерационный обход»;

  б) «обход тезаурусом»;

- этап применения правил логического вывода:

  а) применение импликативных высказываний;

  б) применение утверждений с эквивалентными частями;

- этап оптимизации сгенерированных знаний и сборки мусора.

9.2.1. *Этап задания условия геометрической задачи*

Выбранная нами модель представления знаний в виде семантических сетей предполагает, что все

ключевые понятия базы знаний и связи между ними представлены в виде узлов и дуг, то есть принимают вид связного графа. Решатель опирается на граф условия и параметры запроса. Граф условия – условие задачи, представленное на языке семантических сетей (SC). Параметры запроса – узлы графа условия, к которым обращается пользователь при постановке задачи. Предположим, что нам известны длины одного из катетов и гипотенузы, и необходимо найти длину второго катета. Параметром запроса станет узел, содержащий длину неизвестного катета.

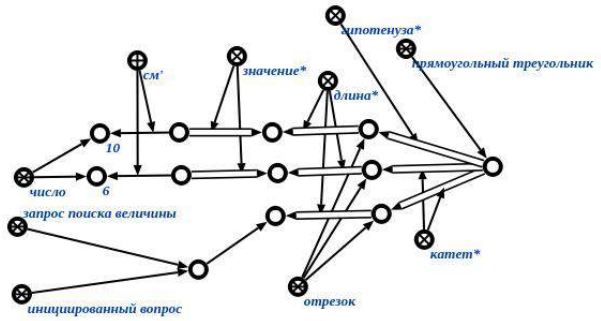

Рисунок 12 - Граф условия

9.2.2. *Этап работы поисковых операций*

Вне зависимости от типа задачи всегда имеется вероятность того, что данная задача уже была решена системой ранее или системе уже откуда-либо известен ответ на поставленный вопрос. Соответственно, первым шагом к решению поставленной задачи станет использование поисковой операции, соответствующей классу решаемой задачи.

А) Класс задач, ориентированный на поиск значения неизвестной величины, решается путём вывода значения искомой величины с пошаговым выводом действий в виде применённых в ходе решения утверждений и результатов их применения. В случае, если граф условия соответствует одному из шаблонов решённых задач, то применение операции вывода решения для заданной задачи позволит решателю с минимальными затратами времени предоставить пользователю ответ.

Б) Класс задач, ориентированный на поиск доказательства, принимает в качестве параметра запроса не узел неизвестной величины, а *sc-структура*, содержащее произвольное высказывание. В ходе решения задачи оно будет либо доказано, вследствие применения вспомогательных логических утверждений, либо опровергнуто, что будет свидетельствовать о его невыполнимости в текущем состоянии базы знаний. Если доказательство высказывания ранее было вынесено в качестве типового, то результат выполнения *операции поиска доказательства для заданного утверждения* предоставит пользователю ответ в виде декомпозиции шагов доказательства с указанием последовательности применения утверждений.

9.2.3. *Этап вывода по аналогии*

При решении задачи мы часто сталкиваемся с тем, что одну величину можно выразить через другую. Идеальным случаем является наличие между объектами бинарного неориентированного отношения, типа *конгруэнтность** или *подобие**, позволяющего на основании знаний об одном объекте делать достоверные выводы о другом. Так как основной характеристикой разрабатываемых систем является их способность к осуществлению различного рода логического вывода, то это подразумевает под собой применение системой не только стандартных алгоритмов, но и других логических схем, таких как аналогия, что позволяет сократить процесс поиска решения в разы. Пример применения аналогии для рассмотренной ранее задачи.

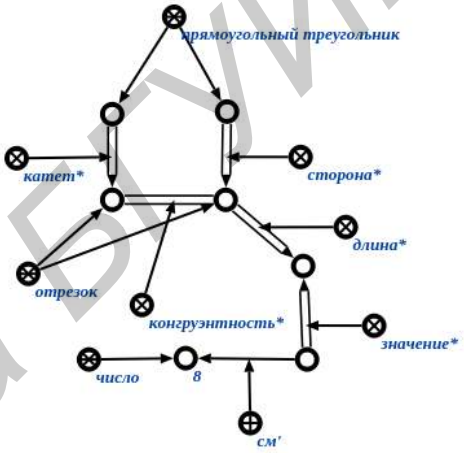

Рисунок 13 — Аналогия

9.2.4. *Этап применения стратегий решения задач*

На данном этапе осуществляется выбор между различными стратегиями решения задач, и, при необходимости, параллельный запуск различных стратегий. На данный момент интеллектуальный решатель реализует комбинированную стратегию, рассмотренную в работе [Заливако 2012].

Вначале рассматривается некоторый объект, для которого осуществляется поиск всех классов объектов, которым он принадлежит. Далее для каждого класса осуществляется поиск утверждений, справедливых для данного класса объектов. При рассмотрении каждого утверждения осуществляется попытка применить его в рамках некоторой семантической окрестности рассматриваемого объекта.

Если ни одно из утверждений класса применить не удается, то стратегия продолжает свою работу и переходит к другим объектам. Когда применения одного утверждения оказывается недостаточно для получения ответа, то стратегия запускается заново, с узла запроса, и поиск решения продолжается.

Были разработаны два возможных варианта обхода стратегии:

а) *«итерационный обход»* - попытавшись применить по разу каждое из утверждений, мы на основании того, был ли выявлен какой-то прогресс в ходе решения, то есть, сгенерированы ли какие-либо новые знания в семантической окрестности задачи, определяем, стоит ли производить ещё один обход.

б) *«обход тезаурусом»* - по мере единственного обхода мы применяем каждое из утверждений по разу и формируем словарь: в него войдут все утверждения, отсечённые на этапе оптимизации сгенерированных знаний по причине нехватки известных параметров. После успешного применения логического высказывания, мы поочерёдно применяем каждое из утверждений, находящихся в словаре, в надежде получить желаемый результат, сокращая таким образом количество утверждений, рассматриваемых на каждом шаге обхода.

Когда при *«итерационном обходе»* прогресс в решении остановится, а при *«обходе тезаурусом»* все логические утверждения окажутся пройденными, можно утверждать о том, что известных параметров для нахождения величины было недостаточно (класс задач, ориентированный на поиск неизвестной величины) или заданное утверждение не является истинным в текущем состоянии базы знаний (класс задач, ориентированный на поиск доказательства).

Для выигрыша во времени решатель пользуется специальным набором утверждений, связка которых соединена с классами объектов через отношение *основные утверждения**. Необходимость введения такой связки ещё и тем, что не все логические утверждения имеют прикладной характер: часть из них используется для дидактических целей, например при обучении, а не для применения в рамках задач.

Рисунок 14 – Стратегия

9.2.5. *Этап применения правил логического вывода*

На данном этапе происходит попытка применения утверждения, полученного на предыдущем шаге, с целью генерации в системе новых знаний. Если такое применение справедливо (например, посылка истинна) и имеет смысл (в результате применения будут сгенерированы новые знания), то осуществляется генерация новых знаний на основе одного из правил логического вывода. Если в данном контексте вывод на основе данного утверждения невозможен или нецелесообразен, решение возвращается на предыдущий этап. В случае успешного применения утверждения происходит переход к следующему этапу решения.

На данный момент решатель поддерживает возможность применения двух типов утверждений:

а) *применение импликативных высказываний* - граф условия соотносится с левой частью и, в случае успеха, дополняется *sc-конструкцией*, состоящей из *sc-констант*, содержащейся в правой части импликации.

б) *применение утверждений с эквивалентными частями* – сначала для графа условия находится часть логического утверждения, изоморфная ему, после чего к нему достраиваются новые знания, в соответствии с оставшимися частями.

Граф условия после применения теоремы Пифагора изображен на рисунке 15.

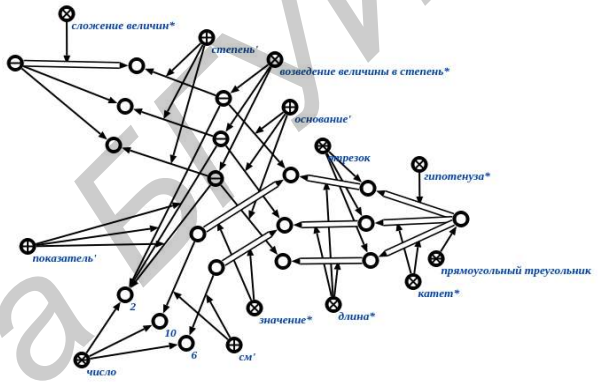

Рисунок 15 – Логика

9.2.6. *Этап оптимизации сгенерированных знаний и сборки мусора*

На данном этапе происходит интерпретация математических отношений, сгенерированных в процессе решения на предыдущем этапе, то есть попытка вычисления недостающих значений компонентов связок на основе имеющихся значений.

В существующей версии решателя возможна как обработка арифметических действий (сумма, умножение, возведение в степень, взятие корня и логарифма), так и вычисление тригонометрических выражений (синуса, косинуса, тангенса и обратных им функций). Вместе они представляют собой мощный математический аппарат, позволяющий решать обширный спектр задач, находящихся на стыке предметных областей.

Если вычислить все недостающие значения не представляется возможным, то все знания, сгенерированные на предыдущем этапе, уничтожаются и решение переходит на этап применения стратегий. В таком случае применение логического вывода для рассматриваемого на предыдущем шаге утверждения считается нецелесообразным. В конечном итоге происходит удаление конструкций, ставших ненужными и по каким-либо причинам не удаленных на предыдущих этапах решения.

Если все этапы решения выполнены успешно, то решение возвращается ко второму этапу, и в результате выполнения поисковой операции пользователь видит полученный ответ. В противном случае, процесс повторяется еще раз, то есть, мы возвращаемся к этапу стратегии.

Такой вид примет граф условия в конце решения задачи:

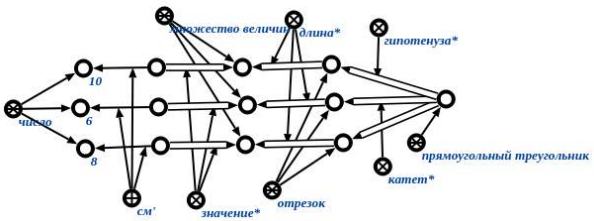

Рисунок 16 – Финальный граф

В будущем, после тестирования и полной отладки решателя, планируется внедрить его в *Библиотеку типовых подсистем интеллектуальных систем*.

Добавление компонента в *Библиотеку многократно используемых компонентов OSTIS* или обновление уже существующего может осуществлять двумя способами:

- В случае *платформенно-зависимого многократно используемого компонента OSTIS* логично использовать стандартные системы контроля версий и с их помощью с учетом прав доступа вносить предложения по изменению компонента.
- В случае *платформенно-независимого многократно используемого компонента OSTIS* предложение по добавлению или изменению компонента может формироваться прямо в *sc-памяти* средствами *SC-кода* и рассматриваться администратором базы знаний с учетом методики проектирования баз знаний (в общем случае любой *платформенно-независимый многократно используемый компонент OSTIS рассматривается* как часть базы знаний).

10. Расширение возможностей пользовательского интерфейса дочерней sc-системы

В конкретных прикладных системах, разработанных по *Технологии OSTIS* помимо *Ядра sc-моделей пользовательских интерфейсов* и средств работы с базовыми языками представления *SC-кода* может оказаться целесообразным использование в системе дополнительных средств отображения или редактирования знаний какого-либо конкретного вида. Подробно методика и средства проектирования пользовательских интерфейсов по *Технологии OSTIS* рассмотрена в [Корончик, 2013]

Для рассматриваемой в данной работе справочной системы по геометрии был разработан специализированный редактор геометрических чертежей, совместимый с выбранным вариантом реализации ядра пользовательских интерфейсов. Данный компонент поставляется в виде набора исходных кодов [github.com ясько], и может быть встроен в реализацию ядра по инструкции приведенной в описании данной реализации. [github.com sc-web]. После встраивания редактора в систему пользователь получит возможность создавать геометрические чертежи из доступных элементов, сохранять в файл и загружать из файла созданный чертеж.

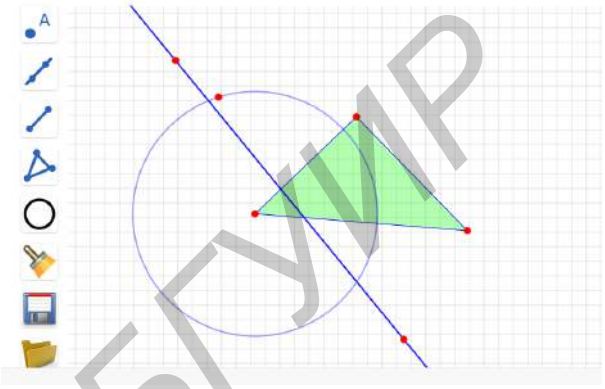

Рисунок 17 – Пример использования редактора геометрических чертежей

Заключение

В работе рассмотрена методика проектирования систем, управляемых знаниями, по *Технологии OSTIS* на примере справочной системы по геометрии. Описаны основные этапы проектирования, указаны возможные способы расширения функционала дочерней *sc-системы*.

При этом рассмотренные процессы развития базы знаний, машины обработки знаний и пользовательского интерфейса в дочерней sc-системе могут выполняться параллельно и в достаточно большой степени независимо друг от друга.

Библиографический список

[Борисов, 2014] Борисов, А.Н. Построение интеллектуальных систем, основанных на знаниях, с повторным использованием компонентов. Открытые семантические технологии проектирования интеллектуальных систем (OSTIS-2014): материалы IV Междунар.научн.-техн.конф./Д.В. Шункевич// Мн.: БГУИР, 2014, -- С.97-103

[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

[IMS] Документация. Технология OSTIS. [электронный ресурс]. – Режим доступа: http://ims.ostis.net/

[sc-machine] sc-machine. [электронный ресурс]. – Режим доступа: https://github.com/deniskoronchik/sc-machine

[Структура sc-машины] Структура sc-машины. [электронный ресурс]. – Режим доступа: https://github.com/deniskoronchik/sc-machine/wiki

[scp-machine] scp-machine. [электронный ресурс]. – Режим доступа: https://github.com/ShunkevichDV/scp-machine

[sc-web] sc-web. [электронный ресурс]. – Режим доступа: https://github.com/deniskoronchik/sc-web

[Шункевич, 2015] Шункевич, Д.В. Средства поддержки компонентного проектирования систем, управляемых знаниями. Открытые семантические технологии проектирования интеллектуальных систем (OSTIS-2015): материалы V Междунар.научн.-техн.конф./Д.В. Шункевич [и др.]// Мн.: БГУИР, 2015

[Давыденко, 2013] Давыденко, И.Т. Технология компонентного проектирования баз знаний на основе унифицированных семантических сетей. Открытые семантические технологии проектирования интеллектуальных систем (OSTIS-2013): материалы III Междунар.научн.-техн.конф./И.Т. Давыденко// Мн.: БГУИР, 2013 – С.185-190

[Шункевич, 2013] Шункевич, Д.В. Модели и средства компонентного проектирования машин обработки знаний на основе семантических сетей. Открытые семантические технологии проектирования интеллектуальных систем (OSTIS-2013): материалы III Междунар.научн.-техн.конф./Д.В. Шункевич// Мн.: БГУИР, 2013.

[Заливако, 2012] Заливако, С.С., Шункевич, Д.В. Семантическая технология компонентного проектирования интеллектуальных решателей задач. Открытые семантические технологии проектирования интеллектуальных систем (OSTIS-2012): материалы II Междунар.научн.-техн.конф. /Д.Н. Корончик// Мн.: БГУИР, 2012. – С. 297-315

[Корончик, 2013] Корончик, Д.Н. Реализация хранилища унифицированных семантических сетей. Открытые семантические технологии проектирования интеллектуальных систем (OSTIS-2013): материалы III Междунар.научн.-техн.конф. /Д.Н. Корончик// Мн.: БГУИР, 2013 – С.125-129

THE METHODOLOGY OF KNOWLEDGE BASED SYSTEM COMPONENT DESIGN

Shunkevich D.V., Davydenko I.T., Koronchik D.N., Hubarevich N.U., Boriskin A.S.

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

shunkevichdv@gmail.com

ir.davydenko@gmail.com

deniskoronchik@gmail.com

stasia@tut.by

coloss_000@mail.ru

In this paper the principles of knowledge-based systems design, based on OSTIS technology, is considered. Indicated technology is an open technology, focused on knowledge-based system design of various intelligent systems. In paper is considered the process of systems design by the example of reference system on the geometry, the processes of knowledge base increase, knowledge processing machine and human-system interface are taken up.

Key words: systems, knowledge-based, semantic networks, knowledge base, intelligent task resolving.

Introduction

Currently, the main issue of the day is absence of well thought-out principles structural using of the eventual computer systems design experience. It is the main granary of reduplication during the development of different system's components. Multiple recurring development of existing technical solutions is stipulate for the well-known technical tasks are bad integrated in developing system or it is hard to find them. And the main goal of the OSTIS technology is the issue of components compatibility and the development of knowledge-based component design tools.

To solve the problem of knowledge compatibility the semantic networks with set-theoretic interpretation are used. And as an example the process of reference system on the geometry would be described.

1. IMS subsystem for supporting sc-subsystems component design

In the context of intelligent metasystem IMS, which is directed on consulting services and supporting system developers who use the OSTIS technology, and accumulation libraries of reusable components of OSTIS single out a support system for components and categories of intelligent system design. It's main tasks are:

- Search possibility of necessary component or set of components in Library of reusable components of OSTIS technology;
- Formation, if it is necessary, sc-structures, comply with the selected reusable components;
- Possibility of transportation of selected reusable components in sc-subsystem;
- In the case of deployment basic version of child sc-system – possibility to install the first based version of sc-system, which can be increased by the added components.

2. The stages of sc-subsystem design

During creating starting version sc-subsystem on OSTIS Technology the four main steps can be marked:

- Selection and installation platform implementation of the sc-subsystem.
- Installation the Core of knowledge base sc-models, the set of based reusable components of knowledge base sc-models, needed for the work of the first sc-system prototype.
- Installation the Core of sc-machine, the set of based reusable components of sc-machine, needed for the work of the first sc-system prototype.
- Installation the Core of interface sc-models, the set of based reusable components of interface sc-models, needed for the work of the first sc-system prototype.
- Installation the core of subsystem for supporting the filial system design in the composition of estimated sc-subsystem.

Conclusion

The paper considers the principles of knowledge-based systems design, based on the OSTIS technology, by the example of the reference system on the geometry.