УДК 004.658.6+004.82+004.8

ПОСТРОЕНИЕ ОНТОЛОГИИ ПРЕДМЕТНОЙ ОБЛАСТИ НА ОСНОВЕ ЛОГИЧЕСКОЙ МОДЕЛИ ДАННЫХ

А. М. Гусенков[1], Н. Р. Бухараев[2], Е. В. Биряльцев[3]

[1,2]*Казанский (Приволжский) федеральный университет, Казань;*
[3]*Центр цифровых технологий Института прикладных исследований Академии наук Республики Татарстан, Казань;*

[1]gusenkov.a.m@gmail.com, [2]boukharay@gmail.com, [3]igenbir@yandex.ru

Аннотация

Представлена технология автоматизированного построения онтологии предметной области на основе информации, извлекаемой из комментариев реляционных баз данных ПАО «Татнефть». Технология основана на построении конвертора (компилятора), транслирующего логическую модель данных Epicentre Petrotechnical Open Software Corporation (POSC), представленную в виде ER-диаграмм и набора описаний на объектно-ориентированном языке EXPRESS, в язык описания онтологий OWL, рекомендованный консорциумом W3C. Описаны основные синтаксические и семантические аспекты преобразования.

***Ключевые слова**: онтология предметной области, реляционные базы данных, POSC, OWL.*

ВВЕДЕНИЕ

Создание онтологии предметной области является, как правило, крайне трудоемким процессом, требующим участия высококвалифицированных специалистов как в конкретной предметной области, так и в области компьютерной лингвистики. Одним из широко применяемых здесь методов является концептуализация понятийного аппарата [1]. При этом мы фактически строим объектную модель некой подобласти реального мира, определяя объекты, их атрибуты и взаимосвязи. Аналогичная техника выделения базовых сущностей и связей используется и для построения логических моделей при проектировании реляционных баз данных [2]. Методологическая близость приемов, используемых при разработке онтологий и логических моделей баз данных, дает основание пред-

положить возможность использования существующих логических моделей баз данных в качестве формализованного прототипа онтологии предметной области.

В статье, на примере разработки системы интеллектуального поиска для крупной нефтедобывающей компании, предложена методика автоматизированного построения онтологии предметной области на основе ее реляционной модели.

В качестве прототипа онтологии предметной области естественно выбрать относящиеся к данной области логические модели, имеющие статус отраслевого стандарта. Одной из них является модель данных Epicentre версии 3.0 нефтетехнической корпорации Petrotechnical Open Software Corporation (POSC [3]). Она представлена в виде ER-диаграмм [2], а также набора текстовых файлов на объектно-ориентированном языке EXPRESS (ISO 10303, part 11). Такое представление ориентировано на автоматическую генерацию структур баз данных по её логической модели, а также визуальное восприятие IT-специалистами.

Для описания онтологии был выбран язык OWL (Web Ontology Language) [4, 5], разработанный рабочей группой Semantic Web Activity и рекомендованный международным консорциумом W3C [6]. Реализация онтологии выполнена на диалекте языка OWL DL, соответствующем правилам дескриптивной логики, с перспективой дальнейшей разработки системы логического вывода утверждений о данной предметной области. В статье описана схема конвертации логической модели Epicentre в язык описания онтологий OWL, учитывающая, помимо общеотраслевых стандартов, специфику нефтегазодобывающей компании ПАО «Татнефть».

МОДЕЛЬ EPICENTRE

В модели данных Epicentre определено более 1000 реально существующих технических и бизнес-объектов, связанных с разведкой и добычей нефти. В терминологии POSC-моделирования данных эти объекты названы сущностями (entities). В модели определены характеристики объектов, называемые атрибутами сущностей (entity attributes). Наиболее важными из них являются атрибуты, определяющие взаимосвязи между сущностями.

Один из важных архитектурных принципов Epicentre основан на различии между объектами, свойствами или характеристиками объектов (properties), с одной стороны, и видами деятельности (activities), с другой. Это разделение поддерживается требованиями практики: возможностью свойств объекта иметь многократные версии или описания, а также тем, чтобы каждое свойство было однозначно связано со своим собственным определением (описанием) или историей обработки.

Логическая модель данных, предлагаемая Epicentre, представляет собой набор определений сущностей и связей между ними в виде выражений на языке Express и диаграмм «сущность–связь». Каждая сущность, представленная в модели, определяется такими параметрами, как набор атрибутов, локальные правила и цепочки супертипов. Атрибут – это перечень объектов, описывающих данную сущность. В свою очередь, атрибут имеет несколько параметров, таких, как имя атрибута, список опций (ключевой, обязательный, внешний и др.), и типы связей. С сущностями также связаны правила валидации сущности – расширенные ограничения целостности данных, определяющие возможные значения атрибутов и связей.

Описание сущностей модели делятся на уровни абстракции представления объектов. Описание предметной области начинается с очень высокого уровня абстракции, на котором нефтегазовая специфика практически отсутствует. Высокоуровневая модель сущностей строится с понятий E_and_P_data, под которым понимается любой информационный объект, и Data_collection, под которым понимается произвольный набор объектов. Модель Epicentre следует принципу моделирования, при котором конкретному экземпляру сущности (entity) – объекту – обеспечивается его существование. Поэтому каждый экземпляр представляет отдельный объект и не может быть версией существующего объекта. Следовательно, новые версии характеристик объекта должны быть указаны в другом месте. В Epicentre эти характеристики связаны с наличием атрибутов или свойств (properties) сущностей.

В модели повсюду применяется архитектурный принцип разделения объектов, свойств и деятельности. Это позволяет непосредственно использовать Epicentre для многих различных по характеру моделей данных. Таким образом, понятия, которые могут рассматриваться в некоторых моделях данных как про-

стые атрибуты сущностей, в Epicentre проявляются как сущности в их полном смысле.

Для обеспечения совместимости со спецификациями POSC множество сущностей в Epicentre расширяется за счет сущностей, имеющих стандартный набор экземпляров. Подобные сущности называются справочными (reference_entities).

В модели существует три типа справочных сущностей:

- POSC Fixed – имеет фиксированное количество экземпляров, определенных POSC;
- POSC Open – имеет фиксированные экземпляры, но также возможно создание дополнительных экземпляров, не определенных POSC;
- Local – нет фиксированных определенных POSC-экземпляров, возможно лишь определение собственных экземпляров.

Все справочные сущности имеют дополнительные характеристики, позволяющие задать источник и библиографию, связанную с источником содержащейся в данном экземпляре информации. Для обозначения справочных сущностей и их типов в схеме Epicentre используется отличительная приставка Ref_ к имени сущности.

Объектно-ориентированная концепция наследования класса – важная часть архитектуры Epicentre. Так как модель данных достаточно велика, концепция наследования классов обеспечивает эффективный способ организации всех сущностей в логически связанную структуру.

Другая фундаментальная часть архитектуры Epicentre основана на понимании того, что многие сущности характеризуются своим пространственным представлением. Каждый из подобных объектов деятельности может быть связан со своим местоположением через отношения с одним или несколькими аналогичными пространственными объектами.

Модель данных Epicentre задается на языке EXPRESS и использует базовые понятия этого языка:

- **Сущность** (entity);
- **Супертип** (supertype) и **подтип** (subtype);
- **Атрибут** (attribute), явный (explicit) и инверсный (inverse);

- **Определяемый** тип данных (defined data type);
- **Простой** тип (simple type);
- **Агрегатный** тип (aggregate type);
- **Ограничение согласованности**, «где» – правило (where rule);
- **Ограничение уникальности** (uniqueness rule);
- **Схема** (schema).

Описанием сущности является определение класса модели Epicentre, на основе которого создаются экземпляры класса или объекты.

Сущность может быть подтипом сущности-супертипа и наследовать от нее атрибуты, правила и ограничения уникальности. Спецификация супертипа – это способ задания свойств данного типа, общих для всех подтипов.

Супертип может быть абстрактным (ABSTRACT), что означает, что все экземпляры должны быть определены. Если сущность не является абстрактной, то экземпляры могут быть неопределенными.

Атрибут (свойство) – это конкретная характеристика сущности. Он может быть прямым (явным), инверсным или выведенным из описания некоторой сущности. Атрибут имеет имя и тип представления.

Явный атрибут – такой атрибут, который не ссылается на описание какого-либо другого атрибута в модели.

Инверсный атрибут служит для выражения обратного направления отношения принадлежности к сущности, возникающего при задании явного атрибута.

Определение схемы представляет собой контейнер, в котором содержатся определения всех сущностей, типов и ограничений, видимых в определенной схеме на языке EXPRESS (рис. 1).

```
ENTITY schema_definition;
        name : STRING;
        types : SET OF named_type;
        global_rules : SET OF global_rule;
        UNIQUE
        Url : name;
END_ENTITY;
```

Рис. 1. Пример определения схемы на языке EXPRESS

Здесь:

name: имя схемы;

types: набор сущностей и определенных типов, объявляемых в схеме;

global rules: набор глобальных ограничений, объявленных в схеме.

Формальные утверждения:

Url: имя схемы должно быть уникально.

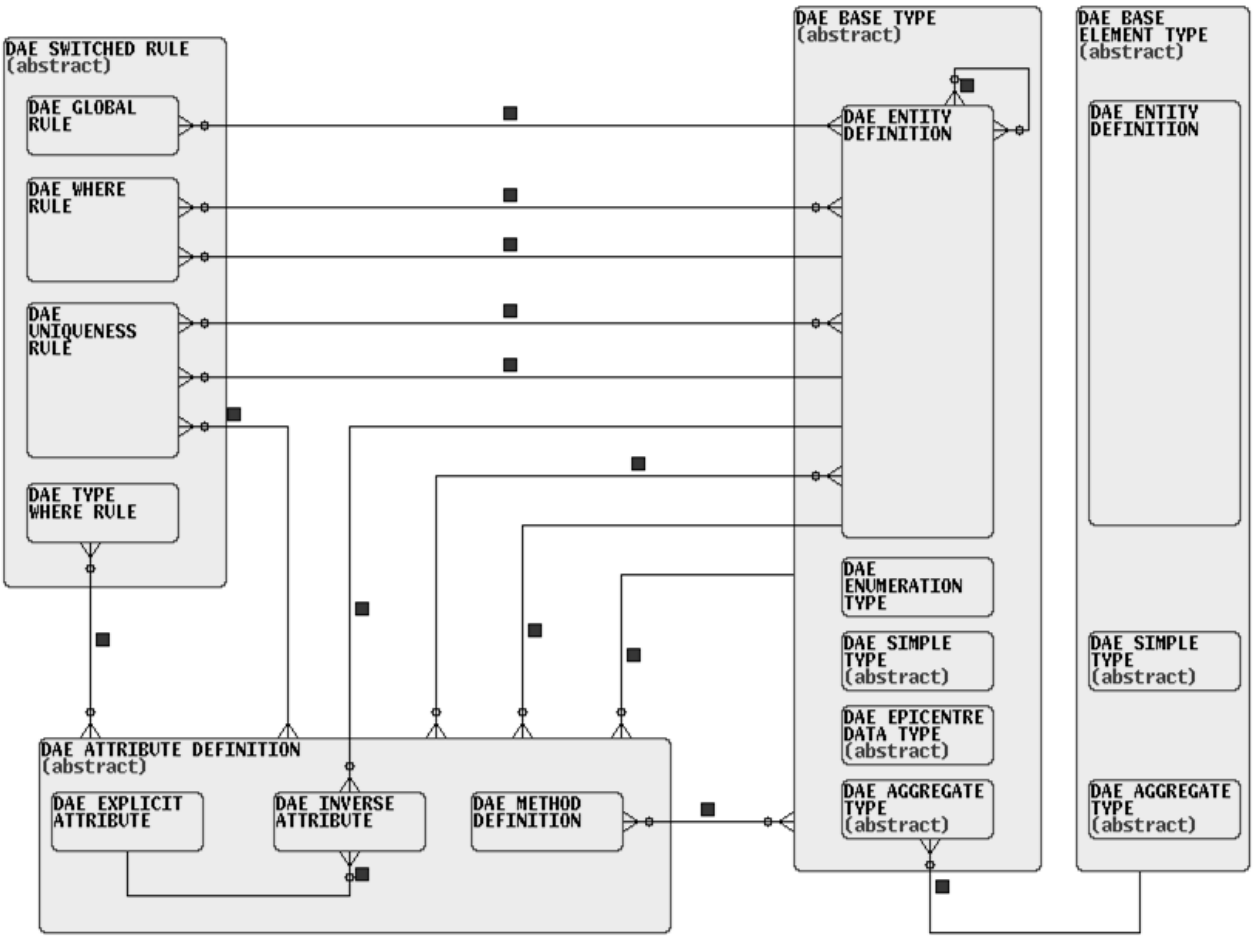


Рис. 2. Метамодель Epicentre

Ограничение уникальности указывает на комбинацию атрибутов, значения которых в совокупности однозначно идентифицируют конкретный экземпляр определяемой сущности.

Ограничение согласованности определяет ограничение, накладываемое на все экземпляры данной сущности.

Схема метамодели представлена на ER-диаграмме (рис. 2).

СТРУКТУРА OWL

Язык веб-онтологий OWL (Web Ontology Language) предоставляет возможности:

- формального определения классов и свойств этих классов;
- определения индивидов (объектов-экземпляров, представителей классов) и их свойств;
- уточнения определений классов и индивидов.

OWL максимально совместим с языками RDF [7] и RDF Schema [8]. Форматы XML [9] и RDF – часть стандарта OWL.

Основными элементами онтологии OWL являются классы, свойства, представители классов (индивиды или экземпляры) и отношения между этими представителями.

Класс – это именованная совокупность свойств, описывающих некий набор индивидов. Таким образом, классы должны соответствовать естественно определяемым наборам объектов рассматриваемой предметной области, а индивиды должны соответствовать реальным объектам, которые могут быть сгруппированы в эти классы.

Наиболее фундаментальные понятия предметной области должны соответствовать классам, которые находятся в корне различных таксономических деревьев. Каждый индивид в OWL является членом класса owl:Thing. Таким образом, каждый класс, определенный пользователем, автоматически является подклассом owl:Thing. Корневые классы, специфичные для данной предметной области, определяются простым объявлением именованного класса.

Фундаментальным таксономическим конструктором для классов является отношение «быть подклассом» rdfs:subClassOf, описывающее связь частного класса с более общим. Если X – подкласс Y, то каждый представитель X – также представитель Y. Отношение rdfs:subClassOf является транзитивным: если X – подкласс Y и Y – подкласс Z, то X – подкласс Z.

Пример: создадим определение для класса ACTIVITY (деятельность) из модели данных Epicentre. ACTIVITY наследует свойства от корневой сущности иерархии супертипов E_AND_P_DATA, представляющих данные геологической разведки и разработки месторождений.

```
<owl:Class rdf:ID=" ACTIVITY">
        <rdfs:subClassOf rdf:resource=" #e_and_p_data "/>
        ...
</owl:Class>
```

Для определения индивида достаточно объявить его членом какого-то класса.

Свойства позволяют утверждать общие факты о членах классов и особые факты об индивидах. Свойство – это бинарное отношение. Различают два типа свойств:

- свойства-значения – отношения между представителями классов и стандартными типами данных, определяемых XML- или RDF-схемой;
- свойства-объекты, определение которых ссылается на описание другого класса.

При определении свойства существует множество способов ограничить это отношение. Можно определить домен и диапазон. Свойство может быть определено как специализация (подсвойство) существующего свойства; возможны и более сложные ограничения. Свойства, как и классы, могут быть организованы в иерархию.

Существует возможность определить характеристики свойства как отношения, что обеспечивает мощный механизм их описания. Приведем некоторые характеристики свойств, важных для описания методики конвертации логической модели Epicentre в язык описания онтологий OWL.

TransitiveProperty

Если свойство P определено как транзитивное, то для любых x, y и z: P(x,y) и P(y,z) предполагают P(x,z).

SymmetricProperty

Если свойство P помечено как симметрическое, то для любых x и y: P(x,y), если P(y,x).

FunctionalProperty

Если свойство P помечено как функциональное, то для любых x, y и z: P(x,y) и P(x,z) предполагают y=z.

inverseOf

Если свойство P1 помечено как owl:inverseOf P2, то для всех x и y: P1(x,y), если P2(y,x).

Заметим, что синтаксис для owl:inverseOf берет в качестве аргумента название свойства. «A, если B» означает здесь (A предполагает B) и (B предполагает A).

InverseFunctionalProperty

Если свойство P помечено как обратно функциональное, то для всех x, y и z: P(y,x) и P(z,x) предполагает y=z.

В дополнение к обозначению характеристик свойств можно разными способами еще больше ограничить диапазон свойства в определенных контекстах. Это делается с помощью следующих ограничений свойств.

allValuesFrom

Данные ограничения являются локальными по отношению к содержащему их классу. Ограничение owl:allValuesFrom требует, чтобы для каждого представителя данного класса, который имеет данное свойство, все значения этого свойства являлись представителями класса, заданного в пункте owl:allValuesFrom.

someValuesFrom

Ограничение owl:someValuesFrom требует, чтобы для каждого представителя данного класса, который имеет данное свойство, хотя бы одно значение этого свойства являлось представителем класса, заданного в пункте owl:someValuesFrom.

Кардинальность

Параметр owl:cardinality позволяет указать количество элементов в связи. Например, свойство UNIQUE класса Name_entity определяет единственную связь (рис. 3).

В частности, в OWL DL owl:maxCardinality может использоваться для указания верхнего предела, а owl:minCardinality – для нижнего предела. В комбина-

ции друг с другом они могут использоваться для ограничения кардинальности свойства в пределах числового интервала.

```
<owl:Class rdf:ID=" Name_entity">
        <rdfs:subClassOf>
                <owl:Restriction>
                        <owl:onProperty rdf:resource="#UNIQUE"/>
                        <owl:cardinality rdf:datatype="&xsd;nonNegativeInteger">1
                        </owl:cardinality>
                </owl:Restriction>
        </rdfs:subClassOf>
</owl:Class>
```

Рис. 3. Пример определения кардинальности на языке OWL

КОНВЕРТАЦИЯ МОДЕЛИ EPICENTRE В ЯЗЫК OWL DL

При построении OWL-онтологии были использованы следующие основные подходы:

- любой сущности Epicentre соответствует простой именованный класс OWL-онтологии с сохранением в именах классов приставок, позволяющих идентифицировать сущности-свойства и сущности-справочники; все эти классы располагаются в корне таксономического дерева онтологии;
- степени связи сущностей (один-к-одному, один-ко-многим или многие-ко-многим) в OWL соответствует определение простых свойств-атрибутов, если связанная сущность не является типом данных, и свойств-значений – в противном случае; указание степени связи между классами реализовано с помощью понятия кардинальности в OWL.

В OWL отсутствуют структурные элементы, которые в полном объеме описывают определение уникальности объектов модели Epicentre. Поэтому в определение каждого класса на языке OWL добавлено новое предопределенное свойство, в котором перечислены все атрибуты, образующие уникальный ключ. Аналогичным же образом решена проблема сохранения условий ограничений.

Для каждой категории данных Epicentre построены отдельные классы OWL-онтологии. Построена формальная LR(1)-грамматика [10] модели Epicentre, на основе которой реализовано семантическое преобразование описанной на

языке EXPRESS модели Epicentre версии 3.0 в онтологию на языке OWL DL. Выполнена русификация описания сущностей и атрибутов модели Epicentre, а также соответствующих им классов и свойств на OWL. Построение онтологии реализовано на диалекте языка OWL DL, соответствующем правилам дескриптивной логики, что в дальнейшем позволит дополнить онтологию системой логического вывода.

Программная реализация конвертора модели Epicentre в язык OWL DL выполнена на языке Java с использованием генератора лексических анализаторов flex [11] и генератора синтаксических анализаторов CUP [12].

Для апробации интеграции баз данных на основе онтологий и проверки корректности построенных онтологий разработан программный интерфейс OakOwlProject 1.0, обеспечивающий навигацию и манипуляции с построенной онтологией. В качестве прототипа для реализации некоторых интерфейсных и функциональных возможностей среды OakOwlProject был выбран известный Java-проект с открытым исходным кодом Protégé [13], функционал которого был существенно расширен. Внешний вид интерфейса OakOwlProject в части его работы с онтологиями предметной области показан на рис. 4.

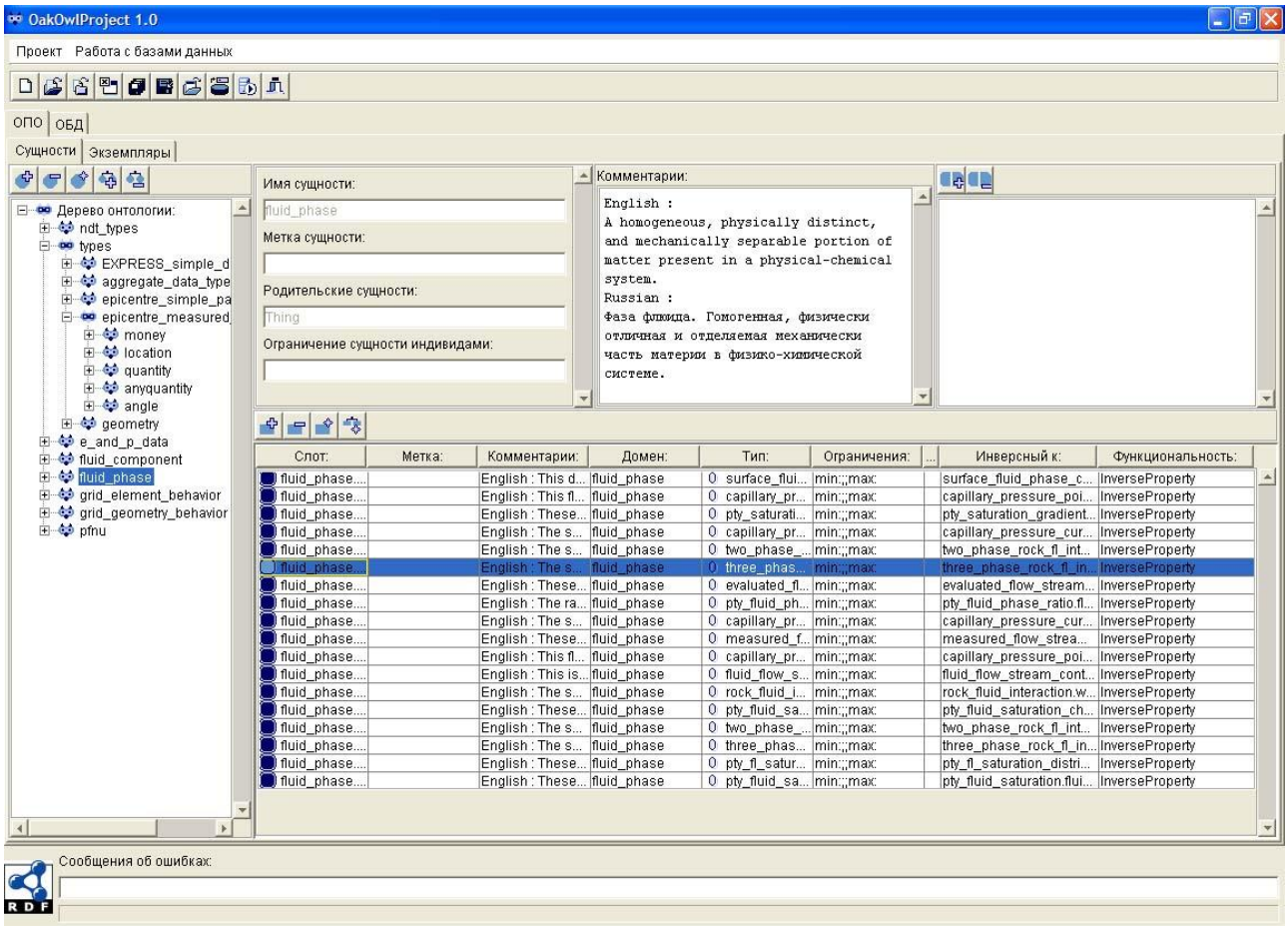


Рис. 4. Общий вид интерфейса пользователя системы OakOwlProject

Описание модели Epicentre на языке EXPRESS представляет собой последовательность описаний сущностей, каждая из которых имеет синтаксическую структуру, представленную на рис. 5.

```
ENTITY name_entity
        ABSTRACT SUPERTYPE OF (
                ONEOF(name_entity1, ..., name_entityN)
        )
        SUBTYPE OF (sub_name1, ..., sub_nameN);
        name_attribute: OPTIONAL SET[0 :?] OF type_name
        ……..
        INVERSE
                invers_attribute_name: SET[0 :?] OF invers_name_entity
                                                FOR entity_invers_type
        ……..
        UNIQUE
        si: name_attribute1, ..., name_attributeN
        WHERE условие
        ……..
END_ENTITY;
```

Рис. 5. Описаний сущностей на языке EXPRESS

В данной синтаксической конструкции используются следующие обозначения:

name_entity – имя сущности; может иметь приставку Pty_ или Ref, указывающую на сущность–свойство или справочную сущность соответственно;

name_entity1, ... , name_entityN – список сущностей;

name_attribute – имя прямого атрибута, может иметь приставку Ref_;

sub_name – имя родительской сущности, может быть обычной сущностью (приставка отсутствует) или сущностью-свойством (имеется приставка Pty_);

sub_name1, ... , sub_nameN – список сущностей sub_name;

type_name – имя типа прямого атрибута, может быть справочной сущностью (Ref_), обычной сущностью (приставка отсутствует) или именованным типом данных (приставка Ndt_);

inverse_attribute_name – имя инверсного атрибута, может быть обычной сущностью (приставка отсутствует) или сущностью-свойством (имеется приставка Pty_);

inverse_name_entity – имя сущности, которая связана обратным отношением с заданной сущностью, может быть справочной сущностью (Ref_), обычной сущностью (приставка отсутствует) или сущностью-свойством (Pty_);

entity_inverse_type – имя сущности, которая является типом прямого атрибута, соответствующего данному инверсному атрибуту, может быть обычной или справочной сущностью;

name_attribute1, ... , name_attributeN – список атрибутов.

Смысл ключевых слов, набранных прописными буквами, будет описан ниже.

На рис. 6 представлен способ конвертации сущностей в классы OWL.

```
ENTITY name_entity          =>   <owl:Class rdf:ID=" name_entity ">
...                              ...
END_ENTITY;                      </owl:Class
```

Рис. 6. Конвертация сущностей в классы OWL

В Epicentre базовыми классами являются: E_AND_P_DATA, FLUID_COMPONENT, FLUID_PHASE, GRID_ELEMENT_BEHAVIOR, GRID_GEOMETRY_BEHAVIOR, PFNU. Автоматически данные классы будут являться подклассами класса owl:Thing и располагаться в корне таксономического дерева. В оболочке OakOwlProject_1.0, например, базовые классы находятся в корне иерархии, показанной на рис. 7.

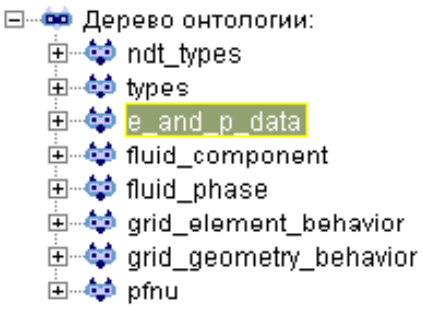


Рис. 7. Базовые классы модели

Конвертация отношений наследования показана на рис. 8. Здесь в конструкции SUBTYPE OF перечисляется список родительских сущностей для данной сущности. Данному отношению в OWL однозначно соответствует синтаксическая конструкция subclass.

```
SUBTYPE OF (                        <owl:Class rdf:ID="name_entity">
    sub_name1, ..., sub_nameN =>    <rdfs:subClassOf rdf:resource="# sub_name1" />
);                                          ...
                                            <rdfs:subClassOf rdf:resource="# sub_nameN" />
                                            ...
                                    </owl:Class>
```

Рис. 8. Конвертация отношений наследования классов

Здесь в конструкции SUBTYPE OF перечисляется список родительских сущностей для заданной сущности. Данному отношению в OWL однозначно соответствует синтаксическая конструкция subclass.

Например, сущности aircraft:

```
ENTITY aircraft
        SUBTYPE OF (  general_facility );
        UNIQUE
        si: identifier,
        identifying_facility,
        ref_existence_kind;
END_ENTITY;
```

соответствует следующий класс OWL:

```
        <owl:Class rdf:ID="aircraft">
        <rdfs:subClassOf rdf:resource="#general_facility" />
        </owl:Class>
```

В оболочке OakOwlProject_1.0 это отношение отображается в поле «Родительские сущности» (рис. 9).

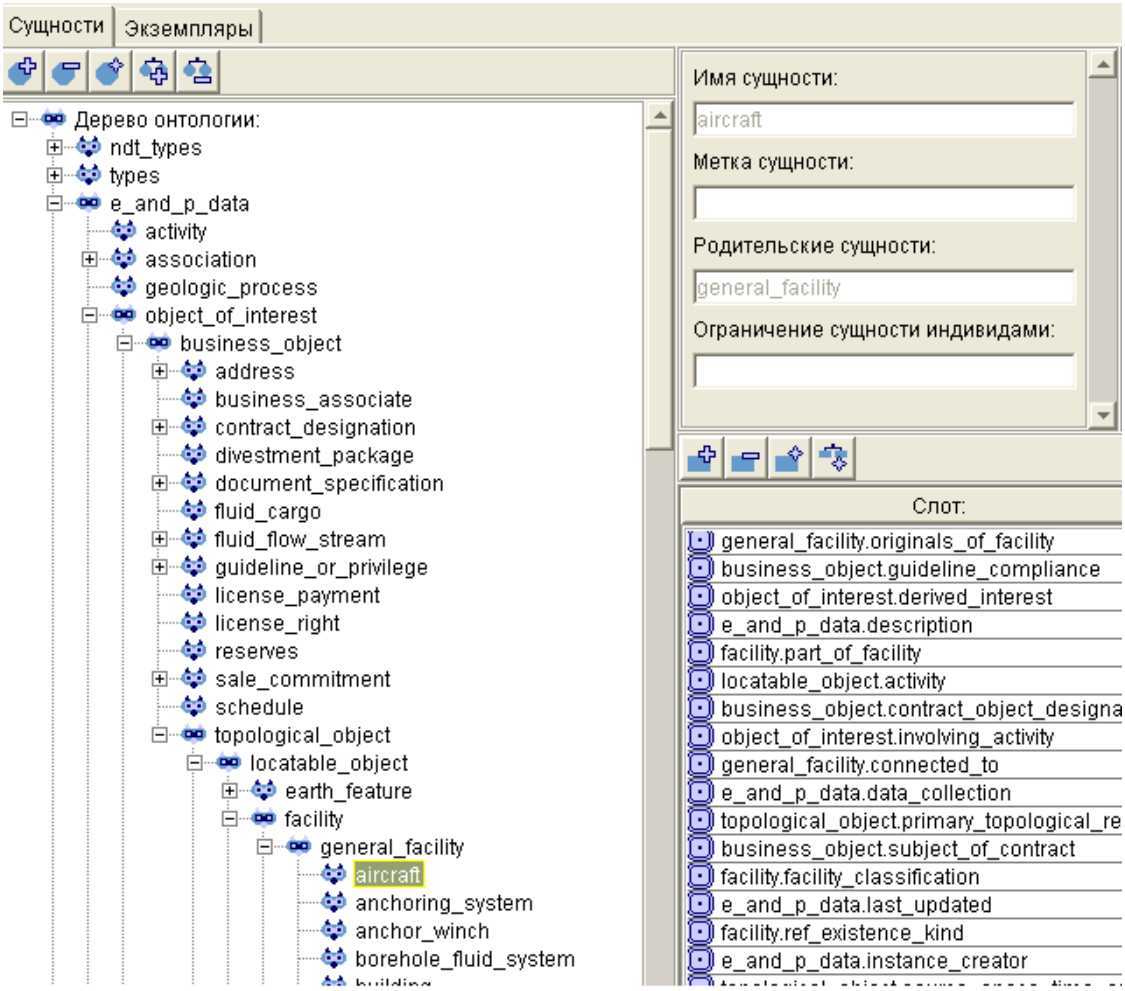


Рис. 9. Отношение наследования

Следующая синтаксическая конструкция языка EXPRESS перечисляет список сущностей, для которых данная сущность является родительской. Ключевое слово ABSTRACT указывается только для абстрактных сущностей (рис. 10).

ABSTRACT SUPERTYPE OF (ONEOF(

name_entity1, ..., name_entityN

Рис. 10. Список сущностей, для которых данная сущность является родительской

В синтаксисе языка OWL отсутствуют средства явного указания классов, которые наследуются из данного класса. Однако в онтологии OWL наследование является обязательным, и оно присутствует в таксономическом дереве онтоло-

гии. Например, в оболочке OakOwlProject_1.0 для абстрактной сущности GRID_ELEMENT_BEHAVIOR, раскрывая список, можно видеть всех ее прямых потомков (рис. 11).

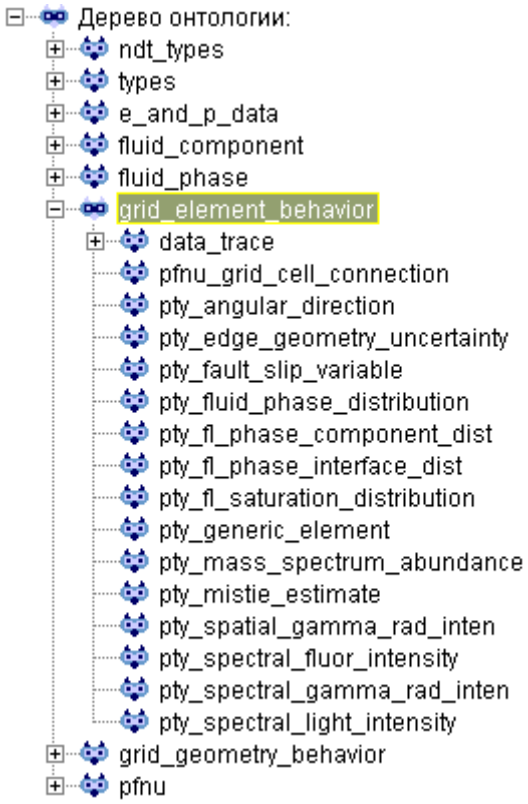


Рис. 11. Визуализация прямых потомков

После определения названия сущности и ее места в иерархии в модели Epicentre приводится список атрибутов, как прямых, так и инверсных. Определению прямого атрибута сущности соответствует синтаксическая конструкция на рис. 12.

name_attribute: OPTIONAL SET[0 :?] OF type_name

Рис. 12. Определение прямого атрибута сущности

Синтаксические конструкции OPTIONAL и SET[0:?] OF могут отсутствовать. Ключевое слово OPTIONAL определяет необязательность наличия значения атрибута, а конструкция SET[0:?] OF определяет степень связи (один-к-одному, один-ко-многим, многие-ко-многим), если атрибут является агрегатным. Такому блоку в OWL соответствует определение простых свойств-атрибутов, если свя-

занная сущность не является типом данных, и свойств-значений в противном случае. Таким образом, такой блок может быть представлен в OWL синтаксической конструкцией, показанной на рис. 13.

```
<owl:ObjectProperty rdf:ID="name_attribute">
        <rdfs:domain rdf:resource="#name_entity"/>
        <rdfs:range rdf:resource="#type_name"/>
</owl:ObjectProperty>
```

Рис. 13. Определение в OWL свойств-атрибутов и свойств-значений

Здесь имя свойства будет соответствовать имени атрибута, а отношение будет связывать исходную сущность и некоторый класс. Проблема повторения имен (например, названия атрибута-свойства и сущности-класса, с которой существует связь), решается введением точечной нотации, то есть свойство будет называться в онтологии name_entity.name_attribute. Если атрибут имеет значения некоторого типа данных, то по смыслу должно быть свойство-значение, но, так как в Epicentre типы имеют более сложную структуру, фактически получаем те же свойства-объекты.

Указание степени связи между классами, а также необязательность (OPTIONAL) можно реализовать с помощью понятия кардинальности в OWL.

```
ENTITY well_test_open_period_recovery
        SUBTYPE OF ( well_test_recovery );
        time_to_surface: OPTIONAL ndt_time;
END_ENTITY;
```

Рис. 14. Ограничение на свойство на языке EXPRESS

Так, OPTIONAL означает, что связь может быть и 0, поэтому в описании класса добавляется ограничение на свойство. Например, для сущности well_test_open_period_recovery (рис. 14) описание в OWL будет следующим (см. рис. 15). Определение самого свойства показано на рис. 16.

```
<owl:Class rdf:ID="well_test_open_period_recovery">
        <rdfs:subClassOf rdf:resource="#well_test_recovery"/>
        <rdfs:subClassOf>
                <owl:Restriction>
                        <owl:onProperty>
                                <owl:ObjectProperty rdf:resource=
                                "#well_test_open_period_recovery.time_to_surface"/>
                        </owl:onProperty>
                        <owl:minCardinality>0</owl:minCardinality>
                        <owl:maxCardinality>1</owl:maxCardinality>
                        <owl:cardinality>0</owl:cardinality>
                </owl:Restriction>
        </rdfs:subClassOf>
</owl:Class>
```

Рис. 15. Реализация степени связи на OWL

```
<owl:ObjectProperty>
        <rdf:about>#well_test_open_period_recovery.time_to_surface
        </rdf:about>
        <rdfs:domain>
                <owl:Class>
                        <rdf:about>#well_test_open_period_recovery</rdf:about>
                </owl:Class>
        </rdfs:domain>
        <rdfs:range>
                <owl:Class>
                        <rdf:about>#ndt_time</rdf:about>
                </owl:Class>
        </rdfs:range>
</owl:ObjectProperty>
```

Рис. 16. Определение свойства на OWL

Определению инверсного атрибута сущности соответствует следующая синтаксическая конструкция (рис. 17):

```
INVERSE
invers_attribute_name: SET[0 :?] OF invers_name_entity
        FOR entity_invers_type;
```

Рис. 17. Определение инверсного атрибута

Данной синтаксической конструкции в OWL можно сопоставить схему, как и в отношении прямого атрибута, только с некоторыми модификациями. Аналогичным образом реализуются и ограничения кардинальности (рис. 18).

```
<owl:ObjectProperty rdf:ID="invers_attribute_name">
        <rdfs:domain rdf:resource="#invers_name_entity"/>
        <rdfs:range rdf:resource="#entity_invers_type"/>
</owl:ObjectProperty>
```

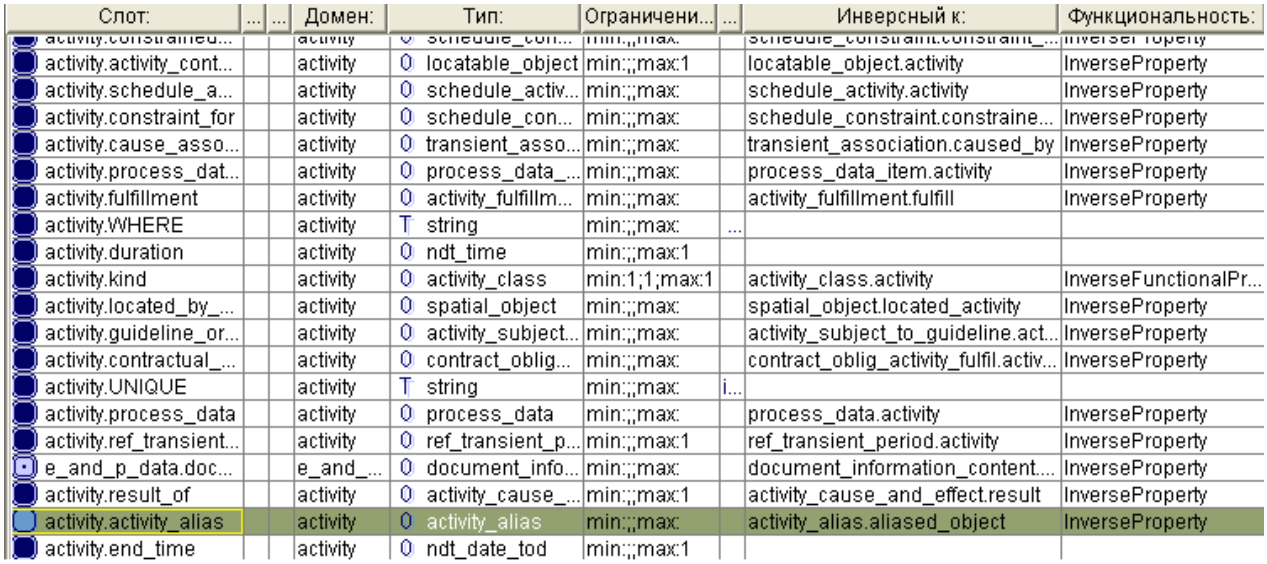
Рис. 18. Ограничения кардинальности

Так, например, у сущности activity есть атрибут activity_alias, который является инверсным к aliased_object, поэтому необходимо обозначить инверсность путем добавления в определение свойства Type: InverseProperty, а также указать, по отношению к какому свойству данное свойство является обратным (рис. 19).

```
<owl:ObjectProperty>
        <rdf:about>#activity.activity_alias</rdf:about>
        <owl:type>
                <rdf:about>#InverseProperty</rdf:about>
        </owl:type>
        <owl:inverseOf>
                <owl:ObjectProperty>
                        <rdf:about>#activity_alias.aliased_object</rdf:about>
                </owl:ObjectProperty>
        </owl:inverseOf>
</owl:ObjectProperty>
```

Рис. 19. Определение инверсного атрибута на OWL

В оболочке OakOwlProject 1.0 все атрибуты сущностей прописаны в поле «Слоты». Наличие переопределений (например, указание кардинальности либо конкретных значений) явно указывается (рис. 20).



Слот:	...	...	Домен:	Тип:	Ограничени...	...	Инверсный к:	Функциональность:
activity.activity_cont...			activity	O locatable_object	min:;;max:1		locatable_object.activity	InverseProperty
activity.schedule_a...			activity	O schedule_activ...	min:;;max:		schedule_activity.activity	InverseProperty
activity.constraint_for			activity	O schedule_con...	min:;;max:		schedule_constraint.constraine...	InverseProperty
activity.cause_asso...			activity	O transient_asso...	min:;;max:		transient_association.caused_by	InverseProperty
activity.process_dat...			activity	O process_data_...	min:;;max:		process_data_item.activity	InverseProperty
activity.fulfillment			activity	O activity_fulfillm...	min:;;max:		activity_fulfillment.fulfill	InverseProperty
activity.WHERE			activity	T string	min:;;max:	...		
activity.duration			activity	O ndt_time	min:;;max:1			
activity.kind			activity	O activity_class	min:1;1;max:1		activity_class.activity	InverseFunctionalPr...
activity.located_by_...			activity	O spatial_object	min:;;max:		spatial_object.located_activity	InverseProperty
activity.guideline_or...			activity	O activity_subject...	min:;;max:		activity_subject_to_guideline.act...	InverseProperty
activity.contractual_...			activity	O contract_oblig...	min:;;max:		contract_oblig_activity_fulfil.activ...	InverseProperty
activity.UNIQUE			activity	T string	min:;;max:	i...		
activity.process_data			activity	O process_data	min:;;max:		process_data.activity	InverseProperty
activity.ref_transient...			activity	O ref_transient_p...	min:;;max:1		ref_transient_period.activity	InverseProperty
e_and_p_data.doc...			e_and_...	O document_info...	min:;;max:		document_information_content....	InverseProperty
activity.result_of			activity	O activity_cause_...	min:;;max:1		activity_cause_and_effect.result	InverseProperty
activity.activity_alias			activity	O activity_alias	min:;;max:		activity_alias.aliased_object	InverseProperty
activity.end_time			activity	O ndt_date_tod	min:;;max:1			

Рис. 20. Визуализация атрибутов

Определению уникальности и ограничений на атрибуты сущности соответствует следующая синтаксическая конструкция (рис. 21):

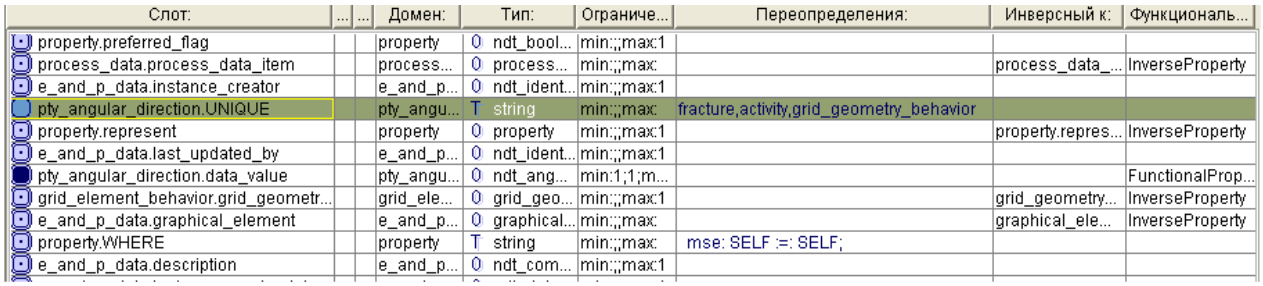
UNIQUE
si: name_attribute1, ..., name_attributeN;
WHERE *условие*

Рис. 21. Определение уникальности и ограничений на атрибуты сущности

Здесь после ключевого слова UNIQUE приведен список атрибутов, образующих уникальный ключ для данной сущности.

В OWL отсутствуют структурные элементы, которые в полном объеме описывают определение уникальности Epicentre. Поэтому в определение каждого класса на языке OWL добавлено новое свойство с именем name_entity.UNIQUE, в котором перечислены все ключевые атрибуты. Аналогичным образом решается проблема ограничений согласованности WHERE: в определение каждого класса на языке OWL добавлено новое свойство с именем name_entity.WHERE, в котором записаны соответствующие условия ограничений.

В оболочке OakOwlProject_1.0 свойства UNIQUE и WHERE, так же, как и атрибуты сущности, относятся к слотам, где в поле «Переопределения» указаны конкретные значения ключей и ограничений (рис. 22).



Слот:	...	...	Домен:	Тип:	Ограниче...	Переопределения:	Инверсный к:	Функциональ...
property.preferred_flag			property	0 ndt_bool...	min:;;max:1			
process_data.process_data_item			process...	0 process...	min:;;max:		process_data_...	InverseProperty
e_and_p_data.instance_creator			e_and_p...	0 ndt_ident...	min:;;max:1			
pty_angular_direction.UNIQUE			pty_angu...	T string	min:;;max:	fracture,activity,grid_geometry_behavior		
property.represent			property	0 property	min:;;max:1		property.repres...	InverseProperty
e_and_p_data.last_updated_by			e_and_p...	0 ndt_ident...	min:;;max:1			
pty_angular_direction.data_value			pty_angu...	0 ndt_ang...	min:1;1;m...			FunctionalProp...
grid_element_behavior.grid_geometr...			grid_ele...	0 grid_geo...	min:;;max:1		grid_geometry...	InverseProperty
e_and_p_data.graphical_element			e_and_p...	0 graphical...	min:;;max:		graphical_ele...	InverseProperty
property.WHERE			property	T string	min:;;max:	mse: SELF :=: SELF;		
e_and_p_data.description			e_and_p...	0 ndt_com...	min:;;max:1			

Рис. 22. Визуализация свойств UNIQUE и WHERE

Типы данных в модели Epicentre классифицированы на следующие категории:

- **Simple Data Types** – простые;
- **Simple Pattern Types** – простые структурные;
- **Measured Quantity Types** – метрические численные;
- **Geometry Types** – геометрические.

Для каждой категории данных в онтологии OWL построены отдельные классы, в свойствах которых использовались встроенные типы данных языка OWL.

ЗАКЛЮЧЕНИЕ

Для синтаксического описания модели Epicentre 3.0 корпорации POSC построена формальная LR(1) грамматика, которая использовалась в качестве входного файла генератора синтаксических анализаторов Java Cup. На основе этой грамматики и схемы конвертации модели Epicentre средствами языка Java построена онтология предметной области природно-технических объектов на языке OWL. Не все конструкции модели Epicentre могут быть синтаксически выражены средствами языка OWL. Для конвертации такой информации были определены специальные классы и зарезервированные свойства OWL, которые могут быть использованы семантически адекватно модели Epicentre.

Общий объем LR(1) грамматики с встроенной в нее семантикой конвертации составил порядка 30 страниц. Файл с описанием модели Epicentre на языке EXPRESS содержит около 500 страниц. Полученная модель на языке OWL имеет объем около 3500 страниц текста. Таким образом, в данной работе полностью, без потери информации, осуществлено преобразование модели Epicentre в OWL-

онтологию. Визуализация построенной онтологии предметной области позволяет наглядно убедиться в ее корректности.

Корпоративное веб-приложение ПАО «Татнефть», основанное на интеграции реляционных баз данных в виде: универсальной онтологии реляционных баз данных, онтологии предметной области, лингвистического тезауруса, и генерирующего SQL-запросы, задаваемые на русском языке, – описано в работах [14–18]. Способ реализации извлечения информации из комментариев реляционных баз данных существенно использует методы компьютерной лингвистики; подробно соответствующий алгоритм извлечения знаний изложен в этих же работах.

Методы оптимизации алгоритма построения SQL-запросов, наиболее ресурсоемкая часть которого связана с необходимостью перебора соединений таблиц, рассмотрены в работе [19].

Благодарности

Работа выполнена за счет средств субсидии, выделенной Казанскому федеральному университету для выполнения государственного задания в сфере научной деятельности, проект 1.2368.2018/ПЧ, а также при финансовой поддержке Российского фонда фундаментальных исследований, проект 18-07-00964.

СПИСОК ЛИТЕРАТУРЫ

1. *Гаврилова Т.А., Хорошевский Т.А.* Базы знаний интеллектуальных систем. СПб.: Питер, 2001. 384 с.

2. *Дейт К.Д.* Введение в системы баз данных. М.: Изд. Дом «Вильямс», 2001. 72 с.

3. *Epicentre v3.0.* URL: http://www.energistics.org/energistics-standards-directory/epicentre-archive.

4. *OWL Web Ontology Language.* URL: https://www.w3.org/TR/2004/REC-owl-features-20040210/

5. Towards the Semantic Web: Ontology-Driven Knowledge Management. Chicester, UK: John Wiley & Sons, 2003. 312 p.

6. *The World Wide Web Consortium (W3C).* URL: http://www.w3c.org.

7. *RDF 1.1 Concepts and Abstract Syntax*. URL: https://www.w3.org/TR/2014/REC-rdf11-concepts-20140225/

8. *RDF Schema 1.1.* URL: https://www.w3.org/TR/rdf-schema/

9. *Extensible Markup Language (XML).* URL: https://www.w3.org/XML/.

10. *Льюис Ф., Розенкранц Д., Стирнз Р*. Теоретические основы проектирования компиляторов. М.: Мир, 1979. 654 с.

11. *Allmon B.J., Anderson J.* Flex on Java. Manning Publications Co. Greenwich, CT, USA, ISBN: 1933988797, 2010. 264 p.

12. *CUP Parser Generator for Java.* URL: https://www.cs.princeton.edu/~appel/modern/java/CUP/

13. *Protégé.* URL: http://protege.stanford.edu/.

14. *Birialtsev E., Bukharaev N., Gusenkov A.* Intelligent search in Big Data // Journal of Physics: Conference Series. V. 913, conference 1. Published online: 25 October 2017.

15. *Гусенков А.М.* Интеллектуальный поиск сложных объектов в массивах больших данных // Электронные библиотеки. 2016. Т. 19. № 1. С. 3–39.

16. *Гусенков А., Биряльцев Е., Жибрик О.* Интеллектуальный поиск в структурированных массивах информации. LAP LAMBERT Academic Publishing. Deutschland: OmniScriptum Marceting DEU GmbH, ISBN 978-3-659-76919-1, 2015. 129 с.

17. *Гусенков А.М., Биряльцев Е.В.* Интеграция реляционных баз данных на основе онтологий // Ученые записки Казанского государственного университета. Серия Физико-математические науки. 2007. Т. 149. Кн. 2. С. 13–34.

18. *Gusenkov A., Bukharaev N., Birialtsev E.* On ontology based data integration: problems and solutions // Journal of Physics: Conference Series. V. 1203, conf. 1. 012059. URL: https://iopscience.iop.org/article/10.1088/1742-6596/1203/1/012059/meta

19. *Gusenkov A., Bukharaev N.* On Semantic Search Algorithm Optimization // New Knowledge in Information Systems and Technologies. WorldCIST'19. Advances in Intelligent Systems and Computing, V. 930. Springer, Cham, 2019. URL: https://link.springer.com/chapter/10.1007/978-3-030-16181-1_45.

BUILDING SUBJECT DOMAIN ONTOLOGY ON THE BASE OF A LOGICAL DATA MOD

A. M. Gusenkov[1], N. R. Bukharaev[2], E. V. Biryaltsev[3]

[1,2]*Kazan Federal University, Kazan, Russia;*

[3]*Center of Digital Technologies of the Institute of Applied Research of the Academy of Sciences of the Republic of Tatarstan, Kazan, Russia*

[1]gusenkov.a.m@gmail.com, [2]boukharay@gmail.com, [3]igenbir@yandex.ru

Abstract

The technology of automated construction of the subject domain ontology, based on information extracted from the comments of the TATNEFT oil company relational databases, is considered. The technology is based on building a converter (compiler) translating the logical data model of Epicenter Petrotechnical Open Software Corporation (POSC), presented in the form of ER diagrams and a set of the EXPRESS object-oriented language descriptions, into the OWL ontology description language, recommended by the W3C consortium. The basic syntactic and semantic aspects of the transformation are described.

Keywords*: subject domain ontology, relational databases, POSC, OWL*

REFERENCES

1. *Gavrilova T.A., Khoroshevskii T.A.* Bazy znanii intellektualnykh system. SPb.: Piter, 2001. 384 p.

2. *Date C.J. Deit K.D.* Vvedenie v sistemy baz dannykh. M.: Izd. Dom «Viliams», 2001. 72 p.

3. *Epicentre v3.0.* URL: http://www.energistics.org/energistics-standards-directory/epicentre-archive.

4. *OWL Web Ontology Language.* URL: https://www.w3.org/TR/2004/REC-owl-features-20040210/.

5. Towards the Semantic Web: Ontology-Driven Knowledge Management. Chicester, UK: John Wiley & Sons, 2003. 312 p.

6. *The World Wide Web Consortium (W3C).* URL: http://www.w3c.org

7. *RDF 1.1 Concepts and Abstract Syntax*. URL: https://www.w3.org/TR/2014/REC-rdf11-concepts-20140225/

8. *RDF Schema 1.1.* URL: https://www.w3.org/TR/rdf-schema/

9. *Extensible Markup Language (XML).* URL: https://www.w3.org/XML/.

10. *Lewis P., Rosenkrantz D., Stearns R.* Teoreticheskie osnovy proektirovaniia kompiliatorov. M.: Mir, 1979. 654 p.

11. *Allmon B.J., Anderson J.* Flex on Java. Manning Publications Co. Greenwich, CT, USA, ISBN: 1933988797, 2010. 264 p.

12. *CUP Parser Generator for Java.* URL: https://www.cs.princeton.edu/~appel/modern/java/CUP/

13. *Protégé.* URL: http://protege.stanford.edu/

14. *Birialtsev E., Bukharaev N., Gusenkov A.* Intelligent search in Big Data // Journal of Physics: Conference Series, V. 913, conf. 1. Published online: 25 October 2017.

15. *Gusenkov A.M.* Intellektualnyi poisk slozhnykh obieektov v massivakh bolshikh dannykh // Elektronnye biblioteki. 2016. T. 19. No 1. S. 3–39.

16. *Gusenkov A., Biryaltsev E., Zhibrik O.* Intellektualnyi poisk v strukturirovannykh massivakh informatsii. LAP LAMBERT Academic Publishing. Deutschland: OmniScriptum Marceting DEU GmbH, ISBN 978-3-659-76919-1, 2015. 129 p.

17. *Gusenkov A.M., Biryaltsev E.V.* Integratsiia reliatsionnykh baz dannykh na osnove ontologii // Uchenye zapiski Kazanskogo gosudarstvennogo universiteta. Seriia Fiziko-matematicheskie nauki. 2007. T. 149, book 2. S. 13–34.

18. *Gusenkov A., Bukharaev N., Birialtsev E.* On ontology based data integration: problems and solutions // Journal of Physics: Conference Series, V. 1203, conf. 1. 012059. URL: https://iopscience.iop.org/article/10.1088/1742-6596/1203/1/012059/meta

19. *Gusenkov A., Bukharaev N.* On Semantic Search Algorithm Optimization // New Knowledge in Information Systems and Technologies. WorldCIST'19. Advances in Intelligent Systems and Computing, V. 930. Springer, Cham, 2019. URL: https://link.springer.com/chapter/10.1007/978-3-030-16181-1_45.

СВЕДЕНИЯ ОБ АВТОРАХ

ГУСЕНКОВ Александр Михайлович – к. т. н., доцент Института вычислительной математики и информационных технологий Казанского (Приволжского) федерального университета. Области научных интересов: технологии извлечения знаний, обработка естественных языков, большие данные, интеллектуальный анализ данных.

Alexander M. GUSENKOV – assistant professor, Institute of Computational Mathematics and Information Technologies of Kazan Federal University. Ph.D. Current scientific interests: knowledge extraction technologies, Natural Language Processing, big data, data mining.

email: gusenkov.a.m@gmail.com

БУХАРАЕВ Наиль Раисович – к. ф.-м. н., доцент Института вычислительной математики и информационных технологий Казанского (Приволжского) федерального университета. Сфера научных интересов: методология IT-образования, извлечение знаний, большие данные.

Naille R. BUKHARAEV – assistant professor, Institute of Computational Mathematics and Information Technologies, Kazan Federal University. Ph.D. Current scientific interests: IT education methodology, knowledge extraction technologies, big data.

email: boukharay@gmail.com

БИРЯЛЬЦЕВ Евгений Васильевич – к. т. н., специалист в области специализированных информационных систем, автор более 50 публикаций, в том числе 5 свидетельств о регистрации программ, 3 изобретений. Заведующий Центром цифровых технологий Института прикладных исследований Академии наук Республики Татарстан.

Evgeny V. BIRYALTSEV – specialist in the field of specialized information systems, Ph. D., author of more than 50 publications, including 5 certificates of registration of programs, 3 inventions. Head of the Center of digital technologies of the Institute of applied research of the Academy of Sciences of the Republic of Tatarstan.

email: igenbir@yandex.ru

Материал поступил в редакцию 17 ноября 2019 года