УДК 007:681.512.2

*Д.И. Каширин, И.Ю. Каширин*

# МОДЕЛИ ПРЕДСТАВЛЕНИЯ ЗНАНИЙ В СИСТЕМАХ ИСКУССТВЕННОГО ИНТЕЛЛЕКТА

*В статье предлагается обзор наиболее известных, показавших свою эффективность, моделей представления знаний, рассматриваются их отличительные особенности, определяющие применимость в решении различных классов задач искусственного интеллекта.*

***Ключевые слова:*** *представление знаний, искусственный интеллект, онтологии.*

**Введение.** Модели представления знаний для описания семантики документов, баз данных и прикладных расчетно-логических задач начали разрабатываться с 70-х годов XX века. Различные подходы отличались собственной спецификой, более всего заметной на синтаксическом уровне формального инструментария описания смысла. Свое применение эти подходы находили в вопросно-ответных [1] и естественно-языковых системах [2], а также в интеллектуальных решателях задач и системах управления [3-5]. Проблематика этого направления получила название “системы представления знаний” (knowledge based systems, knowledge representation systems and languages) [6, 7]. К наиболее известным теоретическим концепциям можно отнести следующие:

- семантические сети [2, 3, 7, 8];
- концептуальные сети [9];
- ситуационное управление [10];
- логическое представление знаний [11];
- универсальный семантический код [12];
- Ф-язык [13];
- фреймы для представления знаний [5];
- растущие пирамидальные сети [14];
- признаковые структуры [15];
- онтологии для описания семантики [16].

**Цель настоящей статьи:** обзорный анализ известных моделей знаний с выявлением соответствующих классов задач, в которых эти модели наиболее эффективны.

**1. Основные понятия, используемые в моделях представления знаний.** К основным понятиям моделей знаний относятся понятия, характеризующие структуру знаний. Перечислим их далее.

Индивид (денотат, объект, экземпляр) – существующий в единственном экземпляре представитель своего класса, которому можно присвоить уникальный индекс (например, “инвентарный номер”).

*Концепт* может быть эксплицитно (перечислением) задан множеством конкретных *индивидов* или имплицитно (содержательно) определен с помощью описания через другие уже известные концепты.

Свойство (роль, признак, слот) - один из атрибутов концепта, характеризующих его с какой-либо точки зрения. Для концептов этот атрибут может иметь имя и не иметь точного значения, но должен иметь область определения значений. Для индивидов свойство имеет точное значение.

Область определения значений свойств (ограничитель, фацет) - множество возможных конкретных значений свойства, заданное либо их перечислением, либо пределами значений, либо описанное через другие, уже известные области значений (типы, домены).

Отношение - множество пар или троек, или кортежей n-й размерности, задающее свойства концептов и индивидов, проявляемые ими при взаимодействии с другими концептами или индивидами. Принадлежность конкретного кортежа этому множеству говорит о существовании свойств у элементов кортежа, проявляемых в их взаимодействии.

Ситуативная структура (ситуация, сложное отношение, фрейм) - система (множество) отношений, определенных на конечном множестве концептов или индивидов, называемых в этом случае “элементами ситуации” или “элементами сложного отношения”. Ситуативная структура может предполагать существование у ее элементов каких-либо свойств.

Базовые понятия сведены в таблицу 1, где рассматривается соответствие терминологии для различных моделей данных.

**Таблица 1**

| Модель знаний | Концепт | Индивид | Свойство | Область значений | Ситуативная структура |
|---|---|---|---|---|---|
| Семантические сети | Понятие | Экземпляр | Свойство | Область значений | Подсеть |
| Концептуальные сети | Категория | Объект | Атрибут | Шкала | Концептуа-лизация |
| Ситуационное управление | Понятие | Объект | Роль | Множество значений | Ситуация |
| Логическое представление знаний | Индивидная переменная | Константа | Предикат | Область значений | Терм |
| Универсальный семантический код | Понятие | Экземпляр | Свойство | - | Формула |
| Ф-язык | Понятие | Объект | Свойство | - | Формула |
| Фреймы для представления знаний | Имя фрейма | Экземпляр | Слот | Фацет | Фрейм |
| Растущие пирамидальные сети | Концептор | Денотат | Признак | Область значений | Пирамида |
| Признаковые структуры | Имя структуры | Структура-экземпляр | Элемент | Область значений | Структура |
| Онтологии для описания семантики | Концепт | Индивид | Свойство | Фацет | Структура |

**2. Краткая характеристика моделей представления знаний.** Семантические сети (СС) представляют собой описание знаний средствами теории графов. Сеть представляется множеством вершин, соответствующих понятиям, свойствам и значениям свойств, а направленные ребра (дуги) - типизированным отношениям.

Существует множество подходов к типизации СС. Можно выделить, например, следующие разновидности СС [17]:

- И/ИЛИ-деревья;
- родовидовая таксономия;
- таксономия отношения «часть-целое»;
- событийные сети;
- сети смешанного типа;
- сети перекрестного поиска;
- сети наследования и умолчаний;
- многосвязные иерархии.

На рисунке 1 приведен пример сети смешанного типа, с помощью которой можно описывать основные знания об игроке баскетбольной команды.

Важным положительным свойством СС является их универсальность с точки зрения применения в различных интеллектуальных задачах, обусловленная близостью к общей теории графов [18]. С другой стороны, фундаментальный характер СС не позволяет использовать их непосредственно, т.е. без соответствующей прикладной доработки, для какой-либо конкретной задачи.

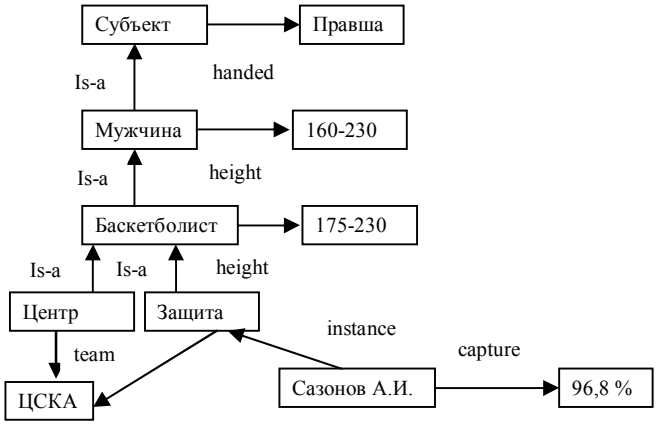


**Рисунок 1 - Пример семантической сети смешанного типа**

Концептуальные сети (КС) – средство графического представления знаний, основанное на базовой типизации понятий (категорий) и их атрибутов. Типизация базовых категорий, которые считаются изначально «понятными» для системы представления знаний, характерна для большинства моделей знаний. В КС, например, классы понятий обозначаются через мнемонику

“PP”, действия, совершаемые представителями PP, − через “ACT”. Особо выделяются концептуальные синтаксические правила для описания концептуализаций (рисунок 2).

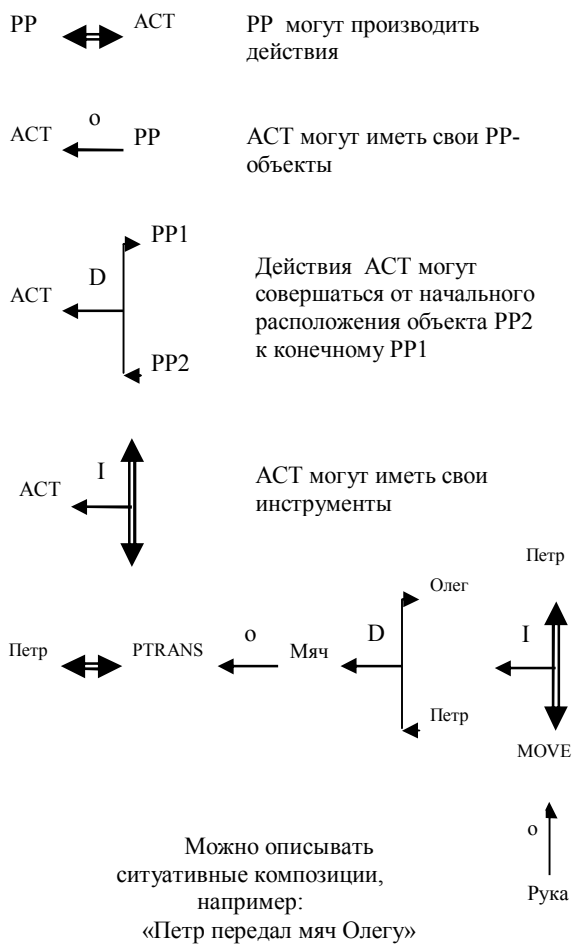


**Рисунок 2 - Концептуальная сеть**

КС, располагая весьма мощными механизмами представления знаний, в то же время сложны для инженерного использования и организации таксономий как основных структурообразующих элементов инженерии знаний.

Ситуационное управление (СУ) – теория, использующая графические и языковые средства для интеллектуального управления сложными объектами. К графическим средствам относятся дискретные ситуационные сети [19] и специализированные семантические сети СУ. Основным инструментарием в СУ является язык СУ. Например, семантика следующего выражения на языке СУ:

$(((q\ \rho\ i_1)r_1((u\ \rho\ v)\ r_{10}\ 100))\ r_{59}\ d_1)$ –
«Изготовлено 100 автомобилей ВАЗ 2109»
может быть описана таблицей 2.

СУ является стройной апробированной тео-рией, используемой в системах интеллектуаль-ного управления сложными системами. В то же время эта теория не предназначалась, например, для применения в задачах информационного поиска.

**Таблица 2**

| **Наименование обозначения** | **Тип обозначения** | **Мнемоника** |
|---|---|---|
| Автомобиль | Объект | q |
| Иметь имя | Отношение | ρ |
| ВАЗ 2109 | Экземпляр | $i_1$ |
| Находиться в состоянии | Отношение | $r_{59}$ |
| Быть готовым | Свойство | $d_1$ |
| Мера | Свойство | u |
| Иметь | Отношение | $r_1$ |
| Штука | Тип значения свойства | v |
| Мера - значение | Отношение | $r_{10}$ |
| 100 | Значение свойства | 100 |

Логическое представление знаний – описание фактографических знаний в языке какой-либо логики.

Простейшим случаем является описание посредством исчисления предикатов первого порядка, например:

$\exists x \exists y\ \text{Learn}(x, y) \rightarrow \text{Student}(x)\ \&\ \text{Discipline}(y)$,

что соответствует выражению «студент x изучает дисциплину y», в котором предикатами являются Learn, Student, Discipline.

Для оперирования знаниями в логической форме разработано множество алгоритмических языков, наиболее известными из которых являются языки семейства Prolog [11].

Логическое представление знаний, базирующееся на прикладных исчислениях предикатов первого порядка, пригодно для решения известных задач теории доказательств. Любое отклонение в описании семантики от классического исчисления требует использования других методов. Более сложными случаями являются использование псевдофизических [20] и нечетких [21, 22] логик, специализированных структур с типизированными элементами.

Универсальный семантический код (УСК) – модель знаний, ориентированная на схемы представления знаний, использующие специализированные структуры с типизированными элементами.

Примером такой схемы является представление ситуации, в которой есть активный субъект $[S_{A,P}\ ]$ действия (он совершает действие), есть пассивный объект действия $[O_{A,P}\ ]$ (над ним совершается действие), и есть медиатор $[M_{S,O}]$ (условия происходящего взаимодействия между субъектом и объектом).

Субъект X также может рассматриваться как причина, а объект Z, − как следствие некоторого процесса, происходящего через посредничество условий или состояний Y:

$$[S_{A,P}] \Leftrightarrow X \leftrightarrow Y \leftrightarrow Z \Leftrightarrow [O_{A,P}]$$
$$\Uparrow$$
$$[M_{S,O}]$$

Такие схемы могут описывать весьма широкий класс описательных и ситуативных структур [10], но ограниченных рамками базовых понятий УСК.

Ф-язык – языковое средство описания смысла, используемое для диалоговых систем, не только содержащее возможность представления родовидовых отношений, но и предполагающее автоматическое построение обратных отношений [13]. В то же время, эта концепция не получила широкого распространения в силу отсутствия эффективных программных реализаций соответствующего инструментария.

Фреймы для представления знаний – концептуальное средство описания понятий, отношений, свойств и ситуативных структур, основанное на Is-a (AKO) таксономии.

Фрейм является главной единицей представления знаний и описывает некоторую сущность посредством слотов как прикрепленным к сущности свойствам. Свойства наследуются, пополняются и уточняются при описании сущностей-наследников. Экземпляры фреймов описывают реальные предметы, явления и ситуации в реальном мире. Значения слотов могут вычисляться в процессе работы с фреймами с помощью специальных присоединенных к слотам процедур-демонов и процедур-слуг.

Реорганизация системы фреймов может автоматически производиться при добавлении, изменении или удалении элементов фреймовых структур. Это описывается соответствующими условиями: If-Added, If-Modified, If-Removed.

Далее приведен простейший пример фрейма «MOMENT» в синтаксисе FRL-0 [5, 23], описывающего текущее значение времени с его структурой, характеристиками и возможностью автоматизированного формирования экземпляра фрейма.

```
MOMENT
  AKO   TIME
        MINUTE
  HOUR $IF-NEEDED(ASK) [TYPE: REQUEST]
                  $IF-ADDED (DAYTIME-
                EQUATION)
                  $REQUIRE (INTEGER-RANGE 0
                23) [TYPE: SYNTAX]
                                (DAYTIME-
                              AGREEMENT)
  DAY   $IF-NEEDED(ASK) [TYPE: REQUEST]
                  $IF-ADDED(WEEKDAY-
                EQUATION)
                  $REQUIRE(INTEGER-RANGE 1
                31) [TYPE: SYNTAX]
                  (DAY-MONTH-AGREEMENT)
                  (WEEKDAY-AGREEMENT)
SDEFAULT(CALENDAR-DAY (NOW))
```

Теория фреймов является достаточно общепринятой и широко используется в современной инженерии знаний. Недостатком ее использования является трудность логической формализации полученных на ее основе новых программных средств.

Растущие пирамидальные сети (РПС) – ациклические ориентированные графы с вершинами-истоками (рецепторами) и вершинами-концепторами. Концепторы могут описывать понятия на основе разнообразных свойств (значений признаков), задаваемых рецепторами как источниками данных, и формируются в процессе работы сети. В начале работы в РПС есть только рецепторы. При появлении конкретных признаков рецепторы переходят в «состояние возбуждения». Этот процесс распространяется по сети вверх. Концептор активизируется, если все его признаки «возбуждены». Важной процедурой является правильная идентификация понятия $A_i$, которое не должно быть спутано с другими. Для этого служат контрольные вершины понятий с характеристиками вершин $m_i$. В качестве примера приводится рисунок 3 [14].

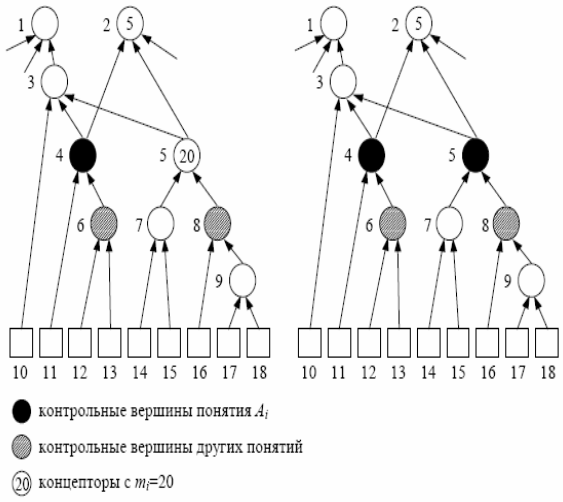


**Рисунок 3 - Фрагмент РПС**

Весьма важным свойством РПС является то, что они позволяют не только распознавать отдельные понятия и ситуации, но и структурировать их взаимосвязи. Это дает возможность автоматического построения различных таксономий. Таким образом, РПС дают возможность решать одну из важных задач инженерии знаний,

дальнейшая работа со знаниями может реализовываться в рамках других концепций.

Признаковые структуры (ПС) – формализм, объединяющий фреймовый и логический подходы в рамках иерархии вложенных кортежей.

Для ПС в отличие от логических термов снимаются следующие ограничения:

- структуры, размеченные символически, не используют явного указания позиций аргументов;
- для структур не требуется указывать фиксированное число составляющих их аргументов;
- устраняются различия между идентификаторами функций и аргументов;
- переменные и взаимные ссылки аргументов обрабатываются отдельно.

На рисунке 4 приведен пример ПС, описывающей человека, находящегося в браке, полученной композицией (операция «$\cup$») признаков.

$$\begin{bmatrix} Type : Person \\ Name : John \\ Age : 23 \end{bmatrix} \cup$$

$$\begin{bmatrix} \mathrm{Pr}\, ofession : builder \\ wife : \begin{bmatrix} Name : Mary \\ Age : 22 \end{bmatrix} \end{bmatrix} \Rightarrow \begin{bmatrix} Type : Person \\ Name : John \\ Age : 23 \\ \mathrm{Pr}\, ofession : builder \\ wife : \begin{bmatrix} Name : Mary \\ Age : 22 \end{bmatrix} \end{bmatrix}$$

**Рисунок 4 - Пример композиции ПС**

Важным свойством ПС является возможность использования для них алгоритмов унификации, позволяющих структурировать и сопоставлять описываемые этими ПС знания. Снятие перечисленных ранее ограничений, однако, ведет к неразрешимости логик, основанных на ПС.

Онтологии для описания семантики – концептуальные модели предметных областей, использующие одну или несколько таксономий и включающие описание понятий, отношений, свойств понятий и ограничений на значения свойств.

Онтологии являются одним из современных средств описания семантики предметных областей (в том числе для описания информационных ресурсов), в котором использованы основные результаты известных ранее моделей знаний. В то же время онтологический подход достаточно прозрачен для его реализации в рамках простых логических формализмов, обладающих свойством разрешимости.

Этот подход принят в качестве одной из базовых методологий при реализации онтологических систем.

**3. Семейство языков OWL для описания онтологий в Semantic Web.** Современная концепция языка OWL (Web Ontology Language) была предложена в форме рекомендаций рабочей группой по Web-онтологиям консорциума W3C в 2004 году. OWL создан как язык для описания Web-онтологий с их приложениями (примерами в прикладных областях) на основе проектирования классов и отношений между ними, которые присущи электронным документам и приложениям.

Язык OWL стал результатом работ по анализу существующих моделей знаний [23], развитию известных программных средств XML, RDF(RDFS) [24], OIL [25], DAML [26] (DAML+OIL) и применению наработок в логике DL [23-26]. По своему типу он является языком разметки и не имеет исполняемых операторов. Фактически в OWL онтологии описываются как типизированные данные. Для работы с OWL (вывод, унификация, структуризация) используются внешние программные средства, реализующие работу с описаниями и ресурсами, которые структурированы этими описаниями.

OWL последовательно расширяется от более простых его подмножеств к сложным по схеме OWL-Lite $\subseteq$ OWL-DL $\subseteq$ OWL-Full.

OWL-Lite позволяет описывать классы с их свойствами как некоторые словари. Описания классов содержат указание суперклассов по родовидовому наследованию и аннотации классов как комментарии. Свойства описываются ограничениями их значений, которые могут быть определены через имена других классов или типы данных, в том числе данных из некоторого ряда констант.

OWL-DL использует OWL-Lite как подмножество и дает возможность дополнительно описывать аксиомы логики DL, т.е., например, объединение (union), пересечение (intersection), дополнение (complement) и разобщение (disjoint) классов, указывать сложные комбинированные типы свойств, включая утверждения об эквивалентности и подмножествах свойств. Важным фактором применимости OWL-DL является сохранение разрешимости и полноты логики, записываемой в рамках его конструкций.

OWL-Full является наиболее открытым языком описания онтологий, в котором допускается нарушение полноты и разрешимости логики DL, для включения в онтологию новых механизмов работы со знаниями.

Далее на примерах рассматриваются основные конструкции OWL.

owl:Thing – имя самого общего класса (кон-

цепта, соответствующего «⊤»).

<owl:Class rdf:ID="Механизм"/> - описание наиболее общего класса «Механизм», что соответствует выражению Механизм $\subseteq \top$ в DL.

Следующие строки позволяют описать класс «Троллейбус» как подкласс Механизма:

```
<owl:Class rdf:ID="Троллейбус">
    <rdfs:subClassOf
rdf:resource="#Механизм"/>
</owl:Class>,
```

что соответствует Троллейбус $\subseteq$ Механизм в DL.

Конструкция

```
< Троллейбус rdf:ID="Троллейбус
Номер А 208 ВС 62" />
```

задает конкретный троллейбус с номером госрегистрации. Далее можно задать дополнительную информацию о созданном индивиде:

```
<owl:Thing rdf:about="# Троллейбус
Номер А 208 ВС  62">
        <Муниципалитет_города
rdf:resource="#Рязань" />
</owl:Thing>
```

В OWL выделяются свойства, выраженные через отношения между индивидами определенного класса и значениями из XML-схемы данных (DatatypeProperty) и между индивидами двух классов (ObjectProperty).

```
<owl:DatatypeProperty
rdf:ID="Иметь_номер">
        <rdfs:domain
rdf:resource="#Троллейбус" />
        <rdfs:range
rdf:resource="http://www.w3.org/
        2008/XMLSchema#string"/>
</owl:DatatypeProperty>
<owl:ObjectProperty
rdf:ID="Использоваться_муниципалитетом">
        <rdfs:domain
rdf:resource="#Троллейбус " />
        <rdfs:range rdf:resource="#Город" />
</owl:ObjectProperty>
```

Свойства также имеют типизацию для соответствующих им отношений:

– FunctionalProperty: индивид имеет только одно значение свойства;

– InverseFunctionalProperty: инверсное функциональному свойство;

– SymmetricProperty: свойство основано на симметричном отношении;

– TransitiveProperty: свойство основано на транзитивном отношении;

– subPropertyOf: подсвойство другого свойства;

– inverseOf: свойство, инверсное другому свойству.

Классы могут иметь ограничения, задаваемые при описании их свойств с помощью отношений с другими классами [27]. Основными видами ограничений являются:

- ограничения квантификации (someValuesFrom, allValuesFrom),
- ограничения мощности множеств (cardinality, minCardinallity, maxCardinality),
- ограничения значений (hasValue).

Например, следующее описание:

```
<owl:Restriction> <owl:onProperty
rdf:resource="#Пассажиры" />
        <owl:someValuesFrom
rdf:resource="#Горожане" />
</owl:Restriction>
```

задает свойство отношением с индивидами «Пассажиры», ограниченное тем, что некоторые из пассажиров являются горожанами. Если нужно указать, что пассажирами являются исключительно граждане, оплатившие проезд, можно использовать другое ограничение:

```
<owl:Restriction>
<owl:onProperty rdf:resource="#Пассажиры" />
        <owl:allValuesFrom
rdf:resource="#Оплативший_проезд" />
</owl:Restriction>
```

Такую характеристику троллейбуса, как максимальное число мест, можно задать с помощью ограничения maxCardinality:

```
<owl:Class rdf:ID="Число_мест">
<rdfs:subClassOf>
        <owl:Restriction>
            <owl:onProperty
          rdf:resource="#Пассажиры"/>
            <owl:maxCardinality
          rdf:datatype="&xsd;nonNegativeI
          nteger">60
            </owl:maxCardinality>
        </owl:Restriction>
      </rdfs:subClassOf>
</owl:Class>
```

Для определения, например, класса «Рязанцы» с ограничением используется описание соответствующего подкласса.

```
<owl:Class rdf:ID="Рязанцы">
        <rdfs:subClassOf>
            <owl:Restriction>
                <owl:onProperty
            rdf:resource="#Горожане" />
                <owl:hasValue
            rdf:resource="#Рязань" />
            </owl:Restriction>
```

```
        </rdfs:subClassOf>
</owl:Class>
```

Стандарт OWL-DL предусматривает операции пересечения, объединения и дополнения. Например, троллейбусом является вид транспорта, принадлежащий муниципалитету и приводимый в движение электричеством.

```
<owl:Class rdf:ID="Троллейбус">
        <owl:intersectionOf
rdf:parseType="Collection">
 <owl:Class
rdf:about="#Муниципальный_транспорт" />
  <owl:Class
rdf:about="#Электродвижимый_транспорт" />
        </owl:intersectionOf>
</owl:Class>
```

Все трамваи и троллейбусы являются муниципальными транспортными средствами. Это можно показать с помощью объединения классов unionOf.

```
<owl:Class
rdf:ID="Муниципальный_транспорт">
        <owl:unionOf
rdf:parseType="Collection">
                <owl:Class
        rdf:about="#Трамвай" />
                <owl:Class
        rdf:about="#Троллейбус" />
        </owl:unionOf>
</owl:Class>
```

Класс концепта «Водитель» может быть определен как дополнение (complementOf) класса «Пассажир», поскольку водители - это люди, не являющиеся пассажирами:

```
<owl:Class rdf:ID="Водитель">
        <owl:complementOf
rdf:resource="#Пассажир" />
</owl:Class>
```

OWL-DL также позволяет задавать классы, принимающие значение на определенном множестве. Такие классы задаются с помощью конструкции owl:oneOf. Например, можно создать класс, содержащий перечисление всех троллейбусных депо города:

```
  <owl:Class rdf:ID="Троллейбусное_депо">
  <owl:oneOf rdf:parseType="Collection">
          <owl:Thing rdf:about="#Депо-1"/>
          <owl:Thing rdf:about="#Депо-2/>
          <owl:Thing rdf:about="#Депо-3/>
  </owl:oneOf>
  </owl:Class>
```

Для того чтобы показать, что троллейбус, автобус и маршрутное такси являются разными видами транспорта, нужно использовать оператор owl:disjointWith:

```
<owl:Class rdf:ID="Троллейбус">
        <owl:disjointWith
rdf:resource="#Автобус"/>
        <owl:disjointWith
rdf:resource="#Маршрутное_такси"/>
</owl:Class>
```

Кроме того, существует возможность явно указать, что класс эквивалентен другому классу. Классы считаются эквивалентными, если они содержат одинаковые наборы индивидных объектов. Для обозначения эквивалентности классов используется конструкция owl:equivalentClass, для свойств – owl: equivalentProperty, для индивидов - owl:sameAs.

```
<owl:Class rdf:ID=" Маршрутное_такси">
        <owl:equivalentClass
rdf:resource="&другая_онтология;Маршрутка
"/>
</owl:Class>
```

**Заключение.** Многочисленные примеры описания различных прикладных онтологий на языке OWL, существующие в настоящее время [28-30], позволяют убедиться в том, что этот язык не только может использоваться для описания семантик, но и уже достаточно распространен как достаточно универсальный общепринятый международный стандарт.

### *Библиографический список*


1. *Белоногов Г.Г., Кузнецов Б.А.* Языковые средства автоматизированных информационных систем. - М.: Наука, 1983. – 288 с.

2. *Попов Э.В.* Искусственный интеллект: Кн.1. Системы общения и экспертные системы: справочник / под ред. Э.В. Попова - М.: Радио и связь, 1990. – 464 с.

3. *Нильсон Н.* Принципы искусственного интеллекта. - М.: Радио и связь, 1985. – 372 с.

4. *Поспелов Д.А.* Логико-лингвистические модели в системах управления. - М.: Энергоиздат, 1981. – 231 с.

5. *Минский М.* Фреймы для представления знаний. – М.: Энергия, 1979. – 151 с.

6. *Гаврилова Т.А., Хорошевский В.Ф.* Базы знаний интеллектуальных систем. − СПБ: Питер, 2001 – 384 с.

7. *Слэйгл Дж.* Искусственный интеллект. - М.: Мир, 1973. – 319 с.

8. *Загорулько Ю.А.* Семантические сети и системы продукций. Методическое пособие. Изд-во НГУ. Новосибирск, 1996. 46 с.

9. *Шенк Р.* Обработка концептуальной информации. – М.: Энергия, 1980. – 360 с.

10. *Поспелов Д.А.* Ситуационное управление. Теория и практика. - М.: Наука, 1986. – 288 с.

11. *Братко И.* Язык PROLOG (Пролог): алгоритмы искусственного интеллекта. – М.: Издательский дом “Вильямс”, 2007. – 640 с.

12. *Мартынов В.В.* Основы семантического кодирования. - М.: ЕГУ, 2001. - 140 с.

13. *Брябрин В.М.* Ф-язык – формализм для представления знаний в интеллектуальной диалоговой системе // Прикладная информатика. Сб. статей. – М., 1981. – С. 73-103.

14. *Гладун В.П.* Растущие пирамидальные сети // Новости искусственного интеллекта, 2004, № 1, С. 30-40.

15. *Knight K. Unification: A* Multidisciplinary Survey. - ACM Computing Surveys. - 1989. - V.21. - N1. - pp. 93-124.

16. *Abdul-Ghafour S., Ghodous P., Shariat B., Perna E.A* Common Design-Features Ontology for Product Data Semantics Interoperability // IEEE/WIC/ACM International Conference on Web Intelligence (WI'07), 2007. - pp. 443-446.

17. *Bullinaria J.A., Huckle C.C.* Modelling lexical decision using corpus derived semantic representations in a connectionist network. In Proceedings of the Fourth Neural Computation and Psychology Workshop, 1997.

18. *Кормен Т.Х.* и др. Алгоритмы для работы с графами // Алгоритмы: построение и анализ. 2-е изд. — М.: «Вильямс», 2006. — С. 1296.

19. *Клыков Ю.И.* Ситуационное управление большими системами. – М.: Энергия, 1974. – 134 с.

20. *Кандрашина Е.Ю., Литвинцева А.В., Поспелов Д.А.* Представление знаний о времени и пространстве в интеллектуальных системах. – М.: Наука, 1989.

21. *Bloch I., Petrosino A., Tettamanzi A.G.B.* Fuzzy Logic and Applications // Lecture Notes in Computer Science, Vol. 3849, 2006.

22. *Masulli F., Mitra S., Pasi G.* Applications of Fuzzy Sets Theory // Lecture Notes in Computer Science, Vol. 4578, 2007.

23. *Goldstein I. A* Frame Representation Language, AI Memo 333, MIT, Cambridge, MA, 1976.

24. *Fensel D., Lausen H., Polleres A., Bruijn J., Stollberg M., Roman D., Domingue J.* Enabling Semantic Web Services. The Web Service Modeling Ontology. - Springer-Verlag, Berlin Heidelberg 2007 – 188 p.

25. *Fensel D., Harmelen F., Horrocks I., McGuinness D., Patel-Schneider P.F.* OIL: An Ontology Infrastructure for the Semantic Web. IEEE Intelligent Systems, 16(2), 2001. - 38-45pp.

26. *Bryson J.,* at al. Agent-Based Composite Services in DAML-S. - 2002, - Режим доступа: www.cs.batch.ac.uk/~jjb/ftp/springer-daml.pdf.

27. *Bruijn J., Polleres A., Lara R., Fensel D.* OWL DL vs. OWL Flight: Conceptual modeling and reasoning on the SemanticWeb. In Proceedings of the 14th International World Wide Web Conference (WWW2005), Chiba, Japan, 2005.

28. *Артемьева И.Л., Высоцкий В.И., Решетаненко Н.В.* Модель онтологии предметной области (на примере органической химии) // НТИ, сер.2. 2005. № 8. С.19-27.

29. *Клещёв А.С., Москаленко Ф.М., Черняховская М.Ю.* Модель онтологии предметной области "Медицинская диагностика". В 2-х частях // НТИ. Сер. 2. Часть 1: 2005. №.12. С.1-7. Часть 2: 2006. № 2. С. 19-30.

30. *Холюшкин Ю.П., Гражданников Е.Д.* Системная классификация археологической науки (элементарное введение в археологическое науковедение). Новосибирск: Изд-во ИДМИ Минобразования, 2000.