Москва 2022

Идея для языка программирования искусственного интеллекта. Свойство-ориентированный подход

АЛЕКСАНДР ГУРЬЕВ, ADMIN@REAL-AI.RU

Оглавление

1. Текущая объектная модель – недостатки для ИИ 1
2. Свойство-ориентированный подход 2
2.1. Системный уровень 2
2.2. Уровень описания предметной области 4
2.3. Пример возможного синтаксиса 7
3. Характеристики, обеспечиваемые предлагаемым подходом 9
4. Заключение 11

Свободу свойствам от объекта!

1. Текущая объектная модель – недостатки для ИИ

Большинство современных языков программирования реализуют объектную модель, которая подразумевает на базовом уровне понятие объекта, обладающего определенными свойствами. Свойства объекта понимаются как именованные переменные того или иного типа. Перечень свойств может задаваться непосредственно для объекта как в прототипных языках программирования (например, в JavaScript), либо через понятие класса в класс-ориентированных языках (C++, Java, Python и многие другие).

Данный подход является чрезвычайно распространенным, большинством разработчиков воспринимается естественным, и для многих традиционных задач обеспечивает адекватное представление предметной области.

Однако, в задачах искусственного интеллекта (ИИ), моделирования мыслительных процессов, понимания смысла такой подход обладает существенным недостатком. И заключается он в реализации понятия свойства. Свойство рассматривается как неотъемлемая часть объекта и представляется в виде некоторого именованного поля внутри объекта. Два, одинаково называемых поля в разных объектах это два совершенно разных системных объекта, занимающих разное место в оперативной памяти и не имеющих ничего общего с точки зрения системы (Рис.1).

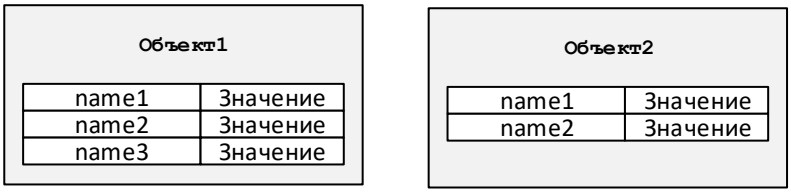

Рис.1 Структура объекта в современных языках программирования

В этом и заключается главный недостаток данного подхода при попытках описать мыслительные процессы. Человек воспринимает свойство самостоятельной сущностью и говоря, например, о цвете машины и о цвете волос, мы понимаем, что речь идет об одном свойстве – цвете.

Следовательно, для реализации алгоритмов, моделирующих мыслительные процессы человека, свойство и объект должны быть представлены в виде самостоятельных и равноправных структур.

В ходе работ над прототипом семантического анализатора RealAI (подробнее о реализации в [1]) нам пришлось реализовать объектную модель, обеспечивающую равноправное представление свойств и объектов. Она была реализована с использованием средств языка Python. Однако было бы здорово, если бы был создан язык программирования, поддерживающий данную концепцию на системном уровне, что позволило бы реализовывать интеллектуальные алгоритмы более прозрачным, лаконичным и приближенном к естественному восприятию образом. В следующем разделе описано как примерно могла бы выглядеть объектная модель при подобном, назовем его, свойство-ориентированном подходе.

2. Свойство-ориентированный подход

2.1. Системный уровень

Если мы хотим сделать свойство самостоятельной от объекта структурой, нам необходимо определиться с помощью каких понятий мы будем его определять. Предлагаемый нами ответ – с помощью понятий действия и его результата (подробнее об обосновании применения действий и их результатов для описания предметной области в [2]). То есть будем рассматривать свойство как объединение некоторого действия (которое потенциально умеет проверять это свойство) и результата выполнения этого действия. Например, свойство «цвет» подразумевает действие по проверке цвета и результат – объекты, соответствующие значениям цвета, например, «красный», «желтый» и др. Действие подразумевает также возможное наличие у него параметров, в зависимости от которых и получается тот или иной результат.

Для представления действия нам нужна именованная структура обеспечивающая объектное представление некоторой функции или активности. Забегая вперед скажем, что она и будет базовым системным элементом, на котором будут строится и которым будут представляться все другие объекты системы – описания свойств, объектов и действий. Структура этого базового системного элемента, а также его взаимосвязи с другими элементами приведены на Рис.2.

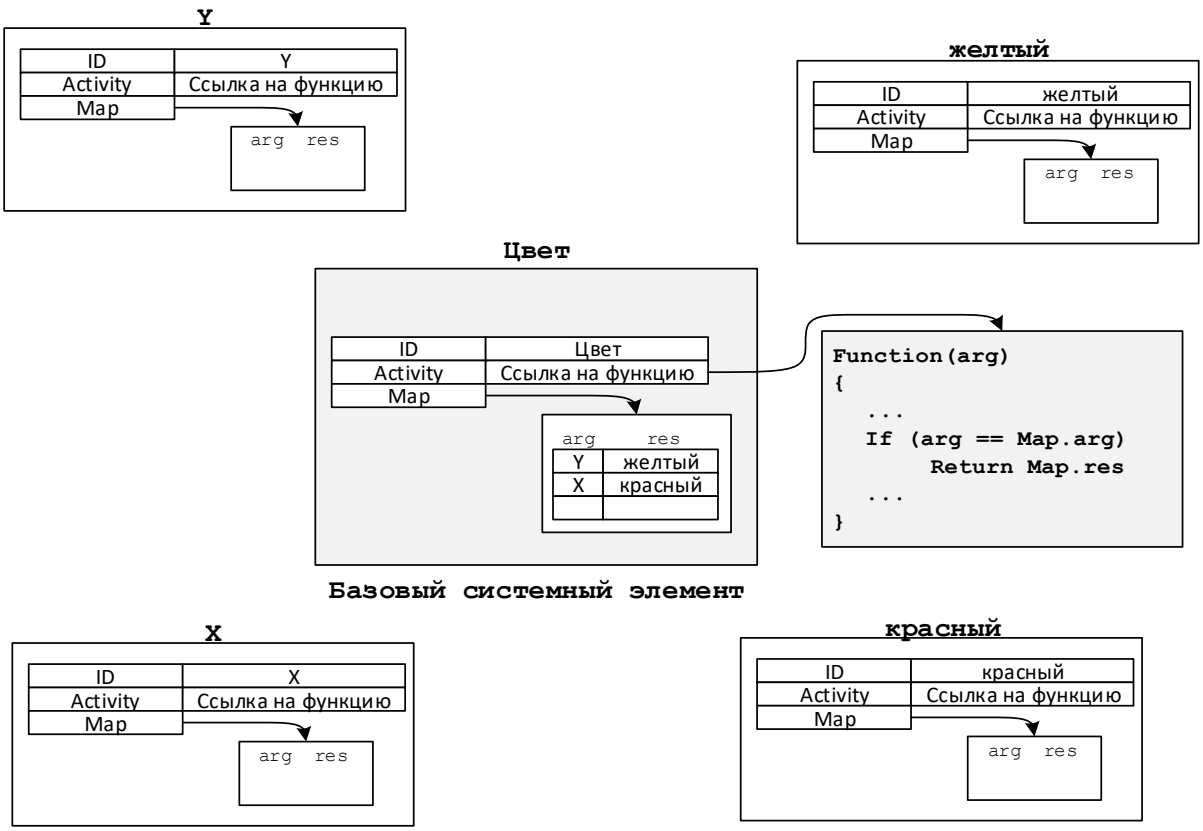

Рис.2 Структура базового системного элемента

Помимо поля активности (Activity), содержащей ссылку на функцию, определяющую эту активность, в структуре базового элемента присутствует структура мэппинга (Map), обеспечивающая сохранение знаний о результате действия при определенном параметре действия. Функция активности может использовать данные из структуры Map для определения результата действия.

Можно заметить, что знания о предметной области при таком подходе выражаются в виде изменения результата, выдаваемого функцией активности соответствующего базового элемента. Например, знание о том, что объект X красный, а объект Y – желтый сохраняется в структуре Map базового элемента «Цвет» (Рис.2), что приводит к получению результата «красный» при выполнении функции с параметром X и «желтый» при выполнении функции с параметром Y.

Данная структура оказывается чрезвычайно универсальной. В зависимости от реализации функции, входящей в состав базового системного

элемента, этот элемент может представлять понятия самого разного типа, в частности:

- Действие, оказывающее воздействие на окружающую среду - если функция выполняет определенные внешние для системы действия (например, выводит на экран «Hello world»);
- Действие, считывающее состояние окружающей среды – если функция возвращает ссылки на базовые элементы, соответствующие некоторым состояниям внешней среды (например, возвращается список ссылок на базовые элементы, соответствующие введенным символам);
- Действие возвращающее значение какого-то свойства для того или иного аргумента (например, рассмотренный выше «цвет» чего-то) – если функция возвращает ссылку на базовый элемент, основываясь либо на внутренних знаниях (сохраненных в Map, Рис.2), либо на проверке свойств аргумента;
- Абстрактный объект, «нечто», - если функция не делает ничего и этот базовый элемент не участвует в других отношениях;
- Объект с определенными свойствами, - функция не делает ничего, но объект присутствует в качестве аргумента в других базовых элементах.

Замечу, что, по моему мнению, подобная структура (но реализуемая с помощью нейронов и других нейрофизиологических механизмах) лежит в основе обработки информации мозгом человека.

2.2. Уровень описания предметной области

Рассмотренная в предыдущем разделе структура (базовый системный элемент) является строительным элементом, из которого образуются понятия более высокого уровня, которые уже непосредственно и используются при описании предметных областей, а именно:

1. *Описание свойства* (property)
2. *Описание объекта* (object)
3. *Описание объекта-действия* (action)

Для классификации объекта системы по отношению к этим понятиям предусматриваются соответствующие встроенные базовые элементы property, object и action, а также базовый элемент «есть(быть)», выполнение активности которого позволяет получить одно из этих значений.

Для формирования *описания свойства* в системе дополнительно определяются встроенные базовые системные элементы – prop_action, prop_argument, prop_result, возвращающие ссылки на базовые элементы, представляющие соответственно действие, входящее в описание свойства, аргумент этого действия и результат этого действия.

Для формирования *описания объекта* в системе дополнительно определяется встроенный базовый системный элемент – obj_properties, возвращающий ссылки на базовые элементы, представляющие описание свойств данного объекта.

С использованием подобных базовых элементов *описание свойства* «Цвет X - красный» имеет структуру, показанную на Рис.3.

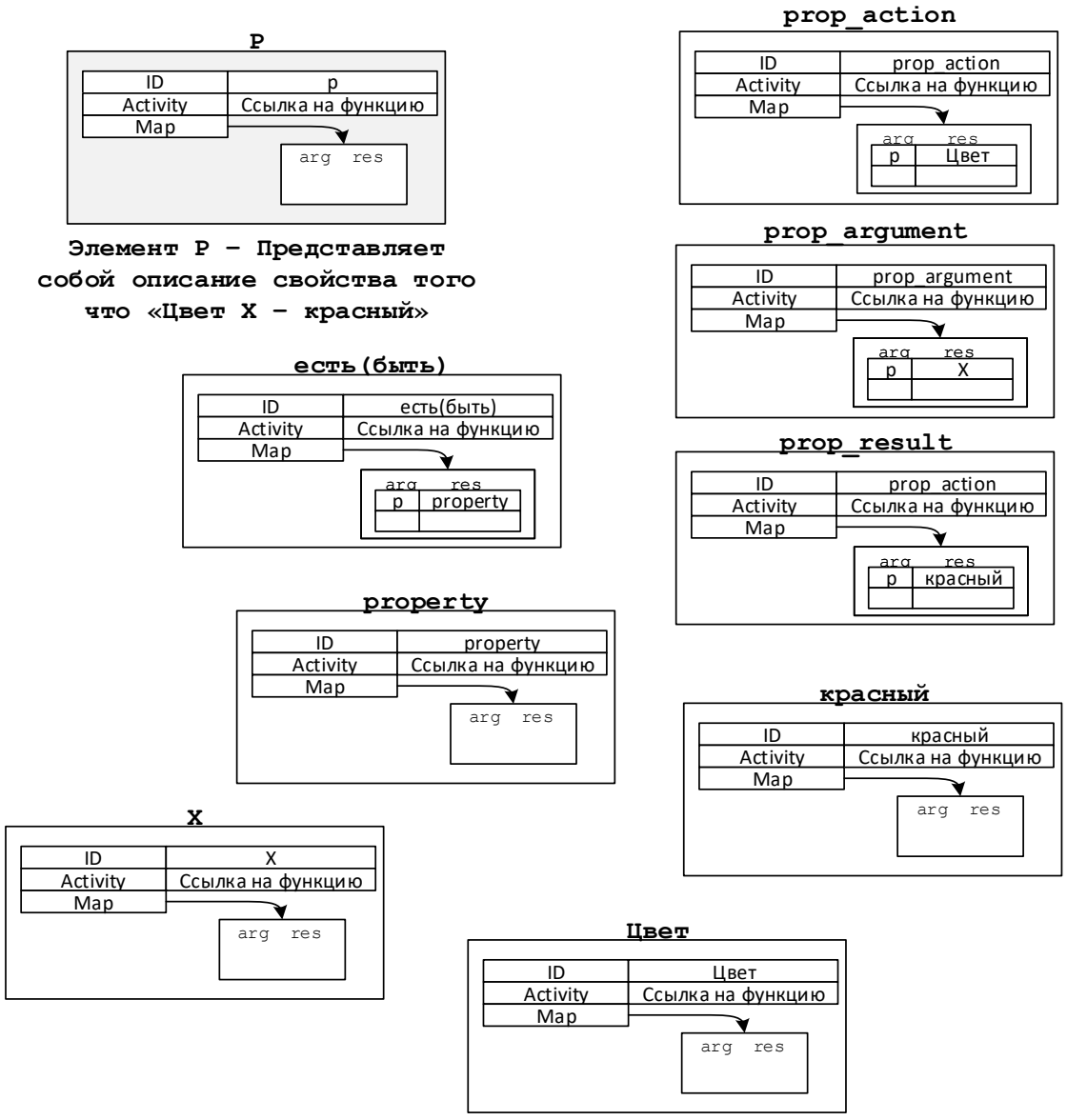

Рис.3 Описание свойства «Цвет X – красный».
Представляется элементом P.

Объекты в традиционном понимании (как нечто обладающее набором свойств) выражаются *описанием объекта*. Для примера на Рис. 4 показан элемент О, представляющий собой описание объекта, обладающего свойствами «Цвет – желтый» и «Форма – круглая».

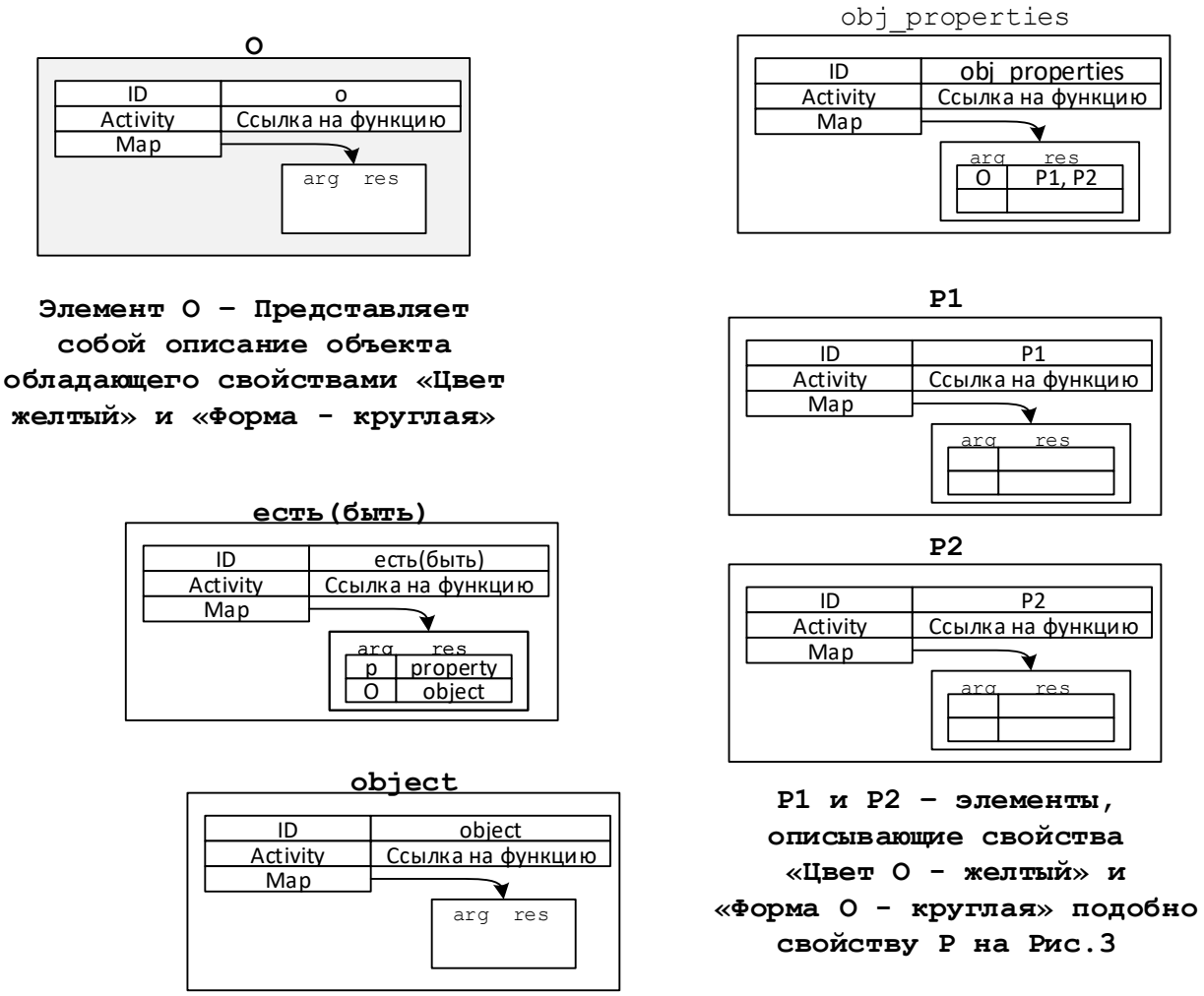

Рис.4 Описание объекта со свойствами «цвет – желтый», «форма – круглая». Представлен элементом О (структура свойств P1 и P2 не показана).

Действия в традиционном понимании, в свойство-ориентированном подходе также выражаются в виде описания объекта (описание объекта-действия), которое предполагает наличие определенного набора специфических свойств, связанных с выполнением некоторой активности – в частности, кто выполняет активность (act_actor), что за активность (act_action), объект активности (act_object), когда выполняется активность (act_time), модальность выполнения активности (хочет, может, надо) (act_mod) и др.

То есть *описание объекта-действия* – это *описание объекта,* в списке свойств которого (получаемого через obj_properties) присутствуют описания

свойств указанных видов (act_actor, act_action, act_object, act_time, act_mod и др.)

2.3. Пример возможного синтаксиса

Основными операциями, необходимыми для работы с объектами, при таком подходе будут следующие:

1. Создание базового системного элемента с определенным ID и функцией активности (типовой, по умолчанию);
2. Выполнение функции, предусмотренной базовым элементом, и получение результата в виде списка ссылок на другие элементы;
3. Манипуляции с таблицей мэппинга базового элемента – добавление, удаление, модификация записи;
4. Операции сравнения списка базовых элементов (указателей на базовые элементы);
5. Создание элементов, соответствующих описанию свойства, объекта и объекта-действия.

На Рис. 5 и Рис.6 приведен возможный вариант синтаксиса для реализации указанных операций.

A = new() – Создание базового элемента с параметрами по умолчанию. Возвращается ссылка на созданный элемент.
B = new(ID,Activity_Function) - создание базового элемента с определенными ID и функцией активности

В системе создаются как минимум следующие предопределенные базовые элементы:
is – Реализует свойство «есть(быть)»
property – Признак описания свойства
object – признак описания объекта
action – признак описания объекта-действия
prop_action – действие в описании свойства
prop_argument – аргумент в описании свойства
prop_result – результат в описании свойства
obj_properties – список свойств объекта
act_action – действие в описании объекта-действия
act_actor – актор в описании объекта-действия
act_object – объект действия в описании объекта-действия
yes – понятие «Да», в логических операциях аналог True, то есть `If(yes)` – всегда истинно
no – понятие «Нет», в логических операциях аналог False, то есть `If(no)` – всегда ложь

res = A(B) – Выполнение активности элемента **A** с параметром – **B.**
res - список указателей на результирующий список элементов.

A(B) = C – Модификация таблицы мэппинга элемента **A**, так чтобы вызов **A(B)** возвращал **C**.
A(B) += C – Модификация таблицы мэппинга элемента **A**, так чтобы вызов **A(B)** добавлял **C** к уже имеющимся записям.
A(B) = nul – Модификация таблицы мэппинга элемента **A, так** так чтобы вызов **A(B)** возвращал пустое значение.

res == C – Проверка на наличие в res элемента **C**.

P1 = new_property(action,argument,result) – Создание базового элемента представляющего описание свойства с указанными элементами `prop_action, prop_argument, prop_result` (Рис.3). Возвращается ссылка на созданный элемент. Обязателен только параметр `action`.
Эквивалентно последовательности:

```
P1 = new()
is(P1) = property
prop_action(P1) = action
prop_argument(P1) = argument
prop_result(P1) = result
```

P2 = new_property(action) - Создание базового элемента представляющего описание свойства с указанным элементом `prop_action` и остальными элементами по умолчанию (**nul**)
Значения argument и result могут быть присвоены после создания:

```
prop_argument(P2) = B
prop_result(P2) = C
```

Рис.5 Вариант синтаксиса по работе со базовыми элементами и описаниями свойств

O1 = new_object(id) – создание базового элемента представляющего описание объекта с указанным ID(Рис.4). Возвращается ссылка на созданный элемент.
Эквивалентно последовательности:

```
O1 = new(id)
is(O1) = object
```

O2 = new_object() – создание базового элемента представляющего описание объекта с автоматически назначенным ID (Рис.4). Возвращается ссылка на созданный элемент.

obj_properties(O1) += [P1,P2] – добавление в список свойств объекта O1 описаний свойств P1 и P2.

A1 = new_action(id, action) – создание базового элемента представляющего описание объекта-действия с указанным ID и элементом представляющим действие. Возвращается ссылка на созданный элемент.
Эквивалентно последовательности:

```
A1 = new(id)
is(A1) = action
P = new_property(act_action,A1,action)
obj_properties(A1) += P
```

Установление элемента B актором для действия A1 выполняется так:

```
new_p = new_property(act_actor,A1,B)
obj_properties(A1) += new_p
```

Рис.6 Вариант синтаксиса по работе с описаниями объектов и действий

Реализация данных операций, наряду с традиционными для языков программирования (условия, циклы, работа со списками и пр.) позволит создавать алгоритмы, приближенные к когнитивным функциям человека. В частности, если описать все доступные системе внешние действия в виде объектов-действий, имеющих определенные свойства, описывающие целевые состояния, достигаемые в результате их выполнения, то станет доступна реализация алгоритмов поведения, направленного на достижение определенного свойства (целенаправленность). Если же предусмотреть механизм динамического комбинирования объектов-действий, то станет возможна реализация адаптивного, самообучающегося поведения, направленного на достижение определенной цели.

3. Характеристики, обеспечиваемые предлагаемым подходом

Приведем некоторые ключевые особенности данного подхода:

- Все объекты системы, описывающие предметную область, на системном уровне представляются единообразно в виде базовых системных элементов (Рис.2). В системе имеется ряд встроенных

элементов, обеспечивающих создание основных видов объектов – описаний свойств, объектов, действий.

- Первичный элемент описания предметной области – свойство. Объект определяется как нечто, способное хранить набор свойств. Действие – подвид объекта.
- Понятия класса и экземпляра класса отсутствуют. С точки зрения излагаемого подхода класс и экземпляр класса – это объекты, различающиеся только возможным набором свойств.
- Определение объекта заключается в создании «пустого» (без свойств) объекта и наделении его определенными свойствами.
- Знания о свойствах тех или иных объектов хранятся не в этих объектах, а в свойствах.

Указанные особенности приводят к следующим свойствам системы, построенной с его использованием:

- Чрезвычайная гибкость. Для описания любого нового понятия достаточно ответить на следующие вопросы:
  - Это объект, действие или свойство?
  - Нужны ли дополнительно связанные с ним понятия?
  - Есть ли уже оно в системе?
  - Если нет, то как оно будет называться в системе?
  - Какими свойствами оно будет связано с другими объектами?

  Причем возможно поэтапное определение понятия. Определив для одной задачи понятие в целом, в дальнейшем, при необходимости, можно уточнить и разложить понятие на более детальные. Например, в качестве возможного результата свойства «статус документа» можно сначала описать понятие «утвержденный» просто как один изолированный объект, а в дальнейшем детализировать этот объект указав что оно означает, что к документу было применено действие «утверждение».
- Легкая расширяемость. Так как описание объекта максимально абстрактно («Нечто»), а новые свойства легко определяются и добавляются, то функциональность может быть расширена без необходимости внесения изменений в уже реализованные действия.
- Выразительность представления. Опыт работы над прототипом семантического анализатора RealAI позволяет говорить, что с использованием данного подхода, возможно представить любые отношения, выражаемые естественным языком.

4. Заключение

В данной статье я попытался изложить новый взгляд на объектную модель, которая может быть основой для построения систем настоящего искусственного интеллекта.

Понятно, что изложенные идеи могут быть реализованы и с использованием имеющихся языков программирования, однако, мне хотелось обратить внимание программистского сообщества на вопрос отношения объекта и его свойств.

В условиях острого недостатка новых идей, мне кажется, стоит пытаться по-новому взглянуть на базовые представления в программировании. В частности, на многие десятилетия считающееся само собой разумеющимся представление о том, что объект первичен, а свойство привязано к объекту.

Литература

[1] Гурьев А.П., «Примеры представления смысла документа через действия», https://habr.com/ru/post/575838/

[2] Гурьев А.П., «Смысл текста или представление знаний в системе, основанной на действиях» https://www.real-ai.ru/blog/wp-content/uploads/2021/04/KR_ADSN.pdf