DOI 10.18699/sblai2023-41
УДК 004.82

Задачный подход: на пути к доверительному искусственному интеллекту
Task approach: on the way to trusting artificial intelligence
任务方法：走向可信人工智能的途径

Е.Е.Витяев Е.Е.,
д.ф.-м.н., Институт математики им. С.Л. Соболева СО РАН,
Новосибирск, Россия.
Vityaev E.E.,
Dr. sci. Sobolev institute of mathematics,
Novosibirsk, Russia, *vityaev@math.nsc.ru*

Гончаров С.С.,
академик РАН, д.ф.-м.н., Институт математики
им. С.Л. Соболева СО РАН, Новосибирск, Россия.
Goncharov S.S.,
Academician RAS, Dr. sci. Sobolev institute of mathematics,
Novosibirsk, Russia, *s.s.goncharov@math.nsc.ru*

Гумиров В.Ш.,
Eyeline Group, Новосибирск, Россия
Gumirov V.S.,
Eyeline Group, Novosibirsk, Russia

Манцивода А.В.
д.ф.-м.н., Иркутский государственный университет,
Иркутск, Россия
Mantsivoda A.V.
Dr. sci., Irkutsk State University,
Irkutsk, Russia, *andrei@baikal.ru*

Нечесов А.В.
к.ф.-м.н., Институт математики им. С.Л. Соболева СО РАН,
Новосибирск, Россия.
Nechesov A.V.
Ph.d, Sobolev institute of mathematics,
Novosibirsk, Russia, *fxcom@yandex.ru*

Свириденко Д.И.
д.ф.-м.н., Новосибирский Государственный Университет,
Новосибирск, Россия.

Sviridenko D.I.
Dr. sci., Novosibirsk State University,
Novosibirsk, Russia, *dsviridenko47@gmail.com*

Аннотация. В данной работе авторы представляют свою оригинальную концепцию объяснимого и доверительного искусственного интеллекта (ДИИ), основу которой составляет задачный поход к искусственному интеллекту. Цель настоящей работы – показать, что задачный подход к ИИ, обобщает и развивает такие известные подходы, как агентный подход к ИИ и общий искусственный интеллект (AGI). Кроме того, в отличии от AGI, он является антропоморфным, поскольку, в том числе, формализует физиологическую Теорию Функциональных Систем (ТФС) работы мозга [15-20], описывающую целенаправленную деятельность человека.

С целью эффективной практической реализации задачного подхода в работе представлены варианты языков исполнимых спецификаций задач, которые позволяют по объяснительной и интерпретирумой спецификации задач генерировать их исполнимый код, приводящий к решениям. Спецификации задач, относящиеся к некоторой предметной области, по которым формируется соответствующий запрос к предметной области, поиск ответа на запрос и получение решения (ответа на запрос) осуществляется в рамках разработанного и представленного в работе семантического моделирования.

Используя различные языки спецификаций, разработано и представлено несколько платформ решения прикладных задач, которые, вместе с примерами решенных задач представлены в заключении работы.

Ключевые слова: искусственный интеллект, задачный подход, проблема доверия,

Abstract. In this paper, the authors present their original concept of explainable and trustworthy artificial intelligence (AI), which is based on a task approach to artificial intelligence. The purpose of this work is to show that the task approach to AI generalizes and develops such well-known approaches as the agent-based approach to AI and artificial general intelligence (AGI). In addition, unlike AGI, it is anthropomorphic, since, it formalizes the physiological Theory of Functional Systems (FTS) of brain function [15-20], which describes purposeful human activity.

For the purpose of effective practical implementation of the task approach, the work presents variants of executable task specification languages that allow, based on the explanatory and interpretable specification of tasks, to generate their executable code leading to solutions. Specifications of tasks related to a certain subject area, for which a corresponding query to the subject area is generated, the search for an answer to the query and obtaining a solution (answer to the query) is carried out within the framework of the semantic modeling developed and presented in the work.

Using various specification languages, several platforms for solving applied problems have been developed and presented, which, together with examples of solved problems, are presented in the conclusion of the work.

Keywords: artificial intelligence, task approach, problem of trust.

摘要：本文介绍基于任务方法的可解释、可信人工智能的基本概念，目的是阐述人工智能的任务方法可以概括和开发其它著名的诸如基于智能体智能和通用人工智能方法。此外，不同于通用人工智能(AGI)，任务方法是拟人化的方法，因为它将脑功能的生理功能系统理论形式化[15-20]，用于描述目的性

人类活动。

为了在实践中有效实施任务方法，本文介绍几种可执行的任务描述语言，通过这些语言对任务的描述和解释，可以生成解决问题的可执行代码。任务的描述与特定的应用领域有关，完成这些任务需要先产生面向应用领域的查询，再执行查询搜索答案，最后得到问题的解。这些操作在本项目研发的一个语义建模框架内完成。

本文介绍用不同任务描述语言开发的解决应用问题的几个平台和解决问题的实例，并给出本项工作的结论。

关键词：人工智能，任务方法，可信问题

1. Введение

После событий, связанных с «революцией ChatGPT» и открывших миру не только новые блестящие перспективы и возможности искусственного интеллекта (ИИ), но и новые риски и опасности его применения, могущих привести к неконтролируемым последствиям, тема ИИ стала чрезвычайно модной и популярной, активно обсуждаемой не только среди специалистов и профессионалов, но и в других кругах, в том числе и в правительственных. И одной из центральных проблем, обсуждаемых в связи с таким бурным развитием событий, стала проблема ***доверия*** к системам ИИ, фактически означающей признание обществом того факта, что современные системы искусственного интеллекта не могут гарантировать правильность предлагаемых ими решений задач. Т.е. признается, что эти решения могут быть ложными, ошибочными, некорректными, неточными, специально вводящими в заблуждение и т.п.

Проблема доверия к ИИ — это комплексная проблема, представляющая собой целый «клубок» проблем, препятствующих доверию человека алгоритмам ИИ. В настоящей работе будут затронуты только некоторые из этих проблем, а именно проблема «черного ящика», проблема централизации и проблема аудита, поскольку по мнению авторов решение этих проблем способно существенно повысить доверие людей к ИИ-системам.

Напомним, что ***проблема «черного ящика»*** означает то обстоятельство, что некоторые ИИ-системы, выдавая результат решения задачи, никак не объясняют его. Более того, чаще всего предъявляемой такой ИИ-системой результат либо не может быть проверен конечным пользователем вообще ни при каких условиях, либо сложность проверки результата требует огромных ресурсов и умственных затрат. И еще одно замечание, связанной с проблемой «черный ящик». Дело в том, что для полного решения обсуждаемой проблемы недостаточно придать прозрачность механизму формирования результатов ИИ-системами. Следует одновременно постараться поднять и уровень компетенций

пользователей этих систем. Ведь высокий спрос на услуги ИИ-систем поддерживается именно пользователями, которые зачастую не особо обращают внимание на проблему «черного ящика». Более того, даже если алгоритмы той же нейронной сети будут прозрачными и формирование результата будет «объяснено» некоторой специальной дополнительной программой, то далеко не факт, что предъявленное обоснование будет убедительно и понятно обычному, не искушенному пользователю. Особенно это относится к искусственным нейронным сетям с миллиардами параметров, с их огромным числом весовых коэффициентов и функций активации. Представляется, что здесь необходимо задействовать логику и логические правила, которые бы выглядели не только убедительно и правдиво, но и могли бы объяснить человеку -- из каких соображений был сформирован результат.

Другой острой проблемой современного ИИ является так называемая ***проблема централизации,*** означающая появление тенденции сосредоточения власти и контроля над ИИ в руках небольшого числа юридических лиц, что может негативно сказаться на развитии и инновациях в этой области. Более того, централизованный контроль за системами вида ChatGPT может привести к таким некорректным настройкам системы, которые могут повлечь за собой очень серьезные последствия и причинить вред как конечному пользователю системы, так и человечеству в целом. Для решения этой проблемы предлагается воспользоваться опытом применения *децентрализованных решений*. Примером может служить применение блокчейна в сфере криптовалют. Самой стабильной компьютерной сетью в мире является на сегодня сеть блокчейна Биткоина, которая стабильно функционирует с 3 января 2009 года. В принципе, блокчейны, как базы данных для хранения информации, вполне могут оказаться пригодными и для ИИ-решений. При этом, чтобы избежать известной *блокчейн-триллемы* [1], можно использовать *мультиблокчейны.* Напомним, что первые мультиблокчейны были анонсированы еще в 2017 году братьями Николаем и Павлом Дуровыми в их «Белой бумаге» для проекта TON [2]. Оказалось, что именно на различных уровнях мультиблокчейнов можно достигать необходимые показатели для масштабируемости, безопасности и децентрализации. Сами же ИИ-алгоритмы могут рассматриваться как *умные контракты*, которые исполняются на блокчейн-подструктурах мультиблокчейна, что обеспечивает защищенность, корректность и прозрачность всех операций, обучающих выборок и принятия решений.

Кроме проблем «черного ящика» и централизации для современной ИИ все более актуальной становится ***проблема ИИ-аудита***, заключающаяся в том, что даже имея доступ к алгоритмам ИИ, мы не всегда можем проверить, как принимает решения система ИИ, базирующаяся на обучении на больших массивах данных. Понятно, что для компетентного анализа таких программных решений понадобятся высоко профессиональные специалисты, которых, как

правило, не хватает и которых надо специально обучать.

В связи с указанными выше проблемами возникает опасность, что система ИИ может принимать решения, которые или непонятны пользователю, или просто не обоснованы. Именно по этой причине и возник ***объяснительный искусственный интеллект*** *(XAI),* требующий от ИИ-систем прозрачности и объяснимости всех выдаваемых ими результатов. Такое требование многие страны в настоящее время пытаются облечь в правовую форму. Так, например, в сентябре 2021 года Национальный институт стандартов и технологий США утвердил 4 принципа объяснительного искусственного интеллекта [3]:

- *Объяснение***:** ИИ должен предоставлять объяснение, которое при необходимости можно детализировать.
- *Осмысленность***:** объяснение ИИ должно быть понятно пользователю.
- *Точность***:** объяснение ИИ должно согласовываться с алгоритмами системы
- *Границы:* ИИ не должен выходить за рамки своих ограничений.

Эти принципы были учтены и китайским правительством и уже в апреле 2023 года Пекин взял курс на XAI [4]. Китай автоматически наложил ограничения на ИИ-разработки всех своих ИИ-компаний. Теперь компании разработчики будет нести ответственность за свои ИИ-решения (в том числе и за действия чат-ботов) если:

- первоначальные источники данных не легитимны.
- за результаты выдачи, которые могут принести вред пользователю системы или вводят его в заблуждение.
- за контент, который нарушал бы основные принципы социалистической партии.
- за призывы к свержению режима или действия, порочащие партию.

В России еще в 2021 году был принят национальный стандарт российской федерации по доверительному искусственному интеллекту (ГОСТ Р 59276-2020) [5]: «Системы искусственного интеллекта: способы обеспечения доверия», в котором приведена классификация факторов, влияющих на качество и способность ИИ-систем вызывать доверие на стадиях жизненного цикла. И одним из основных пунктов этого документа выступает объяснимость искусственного интеллекта. Данный стандарт распространяется только на ИИ-системы, обеспечивающий решение только конкретных практических задач и не может быть использован для систем «сильного» или «общего» ИИ.

Учитывая столь огромный интерес к проблеме доверительности и объяснительности ИИ, авторы настоящей работы представляют общественности свою оригинальную концепцию построения ***доверительного искусственного интеллекта (ДИИ),*** основу которой составляет ***задачный поход*** к искусственному интеллекту [7-12], обобщающий в определенной степени такие известные и популярные подходы, как *агентный подход* и *общий*

искусственный интеллект (AGI).

Заметим, что успехи и популярность ***агентного подхода*,** изложенного в монографии [13], во-многом обязаны удачно выбранной системе понятий, включающей различные виды «агентов» и «внешней среды». Различные задачи искусственного интеллекта рассматриваются в этом подходе как задачи взаимодействия «агента» со «средой», что позволило его авторам осуществить детальную классификацию агентов, сред и, что принципиально важно, решаемых ими *задач*. Тем самым, агентный подход фактически следует задачному подходу, но только без явного определения решаемых агентами задач и указания всех присущих понятию «задача» компонент.

Общий искусственный интеллект (Artificial General Intelligence, AGI), обзор которого приведен в [14], показал, что моделирование когнитивных процессов человека не является обязательным условием решения интеллектуальных задач. Поэтому AGI отражает тот факт, что искусственным интеллектом в той или иной степени может обладать, как человек или живой организм с высокоразвитой центральной нервной системой, так и абстрактная робототехническая система. Ведущие разработчики AGI (Бен Гёрцел, Шейн Легге, Пей Ванг) дают следующее определение AGI: «это способность решать *когнитивные задачи* в целом, действуя целенаправленно, адаптируясь к условиям среды через обучение, минимизируя риски и оптимизируя потери на достижение своих целей» [14].

Цель настоящей работы – показать, что задачный подход к ИИ, обобщая и развивая упомянутые выше подходы, и при этом являясь по своей сути ***антропоморфным*,** поскольку, опираясь на *Теорию Функциональных Систем* (ТФС) работы мозга [15-20], позволяет адекватно моделировать когнитивную целенаправленную деятельность человека, способен стать концептуальной основой доверительного ИИ и соответствующего ему инструментария.

2. *Задачный подход и его математическая теория*

2.1. *Задачный подход*

Первые попытки придать понятию «задача» точный смысл были предприняты академиком А.Н.Колмогоровым еще в 30-е годы прошлого столетия, который использовал понятие задачи для описания математической семантики интуиционистского исчисления высказываний [21]. Большой вклад в становление задачного подхода применительно к решению задач, связанных с созданием и развитием технических систем, внесла также и эмпирическая Теория Решения Изобретательских Задач (ТРИЗ) [22]. Однако наиболее обоснованное, систематическое и детальное описание концепции задачного подхода было осуществлено академиком Ю.Л.Ершовым и д.ф.н. К.Ф.Самохваловым применительно к проблемам оснований математики [7-8].

2.1.1. ***Задачный подход к основаниям математики и ИИ.*** Анализ понятия задачи применительно к основаниям математики начинается со следующих простых рассуждений, сформулированных К.Ф. Самохваловым: «Я хочу пить» – что это значит? Нет, конечно, никакой ошибки полагать, что слова «я хочу пить» означают просто вот это, где это – определенное состояние сознания, которое я переживаю сейчас и которое я именую жаждой. Но тогда возникает новый вопрос: как ощущение жажды (хотения) связано с фактическим питьем (удовлетворением хотения)? Откуда я знаю, что удовлетворить жажду можно питьем? Содержится ли в самом переживании жажды сознание того, чем эту жажду можно удовлетворить? … Знать желание не означает знать желаемое, а означает способность узнать желаемое», т.е. иметь критерий удовлетворения желания.

Таким образом, задача определена (осмыслена) тогда и только тогда, когда у нас есть *критерий решенности задачи* – критерий проверки того, что действительно ли предъявленное решение является решением задачи. В математических теориях таким критерием считается наличие *доказательства* решения задачи. Но этот критерий применим только тогда, когда в рамках самой формальной системы мы имеем как доказательство решения задачи, так и возможность убедиться средствами самой же системы, что данное доказательство действительно является решением задачи. В [7-8] было доказано, что только в «слабых» формальных системах (для которых не проходит теорема Гёделя) мы можем средствами самой формальной системы определить, является ли некоторый текст доказательством решения задачи или нет.

В результате программа Гильберта обоснования математики должна быть переформулирована следующим образом: не нужно для всей математики доказывать её непротиворечивость, как это было провозглашено в исходной программе Гильберта – это невозможно и ненужно. Надо формулировать и решать задачи в рамках и средствами соответствующих слабых формальных систем.

2.1.2. ***Понятие «задача» в искусственном интеллекте.*** Определим теперь понятие «задача», ставя своей целью рассмотрение максимально широкого круга задач, что характерно для такого направления, как ***искусственный интеллект.*** Приведем сначала неформальное определение.

Задача определена в том и только в том случае, когда в ее формулировке присутствуют:

- указание предметной области, зафиксированные в виде некой формальной модели, включая описание сигнатуры и структуры языка (онтологии), а также знания о предметной области, включая исходные данные, факты и гипотезы;
- на какой запрос (вопрос), сформулированный в задаче к предметной

области, мы должны получить ответ (решение задачи);

- критерий удовлетворения запроса – в каком случае можно считать, что ответ (решение) на запрос (вопрос) получен;

- в каком контексте следует искать ответ (решение) на запрос (вопрос) – какую цель мы преследуем, решая задачу, т.е. что мы ожидаем от полученного результата и каковы его последствия и что делать, если ответ окажется отрицательным.

Предлагаемый задачный подход предполагает, что истинное назначение искусственного интеллекта состоит в автоматизации решения задач, предполагая, что задачи формулируются в терминах исполнимых (декларативных) спецификаций, рассмотренных далее. Кроме того, введенное нами понятие задача, в отличии от AGI, как мы покажем далее, адекватно моделирует целенаправленную деятельность человека и животных в соответствии с ТФС.

2.1.3. ***Понятие «задача» в когнитивных науках***. Обобщением понятия задача в когнитивных науках является понятие ***Цели*** [15-16,20]. Цель нельзя достичь, не имея критерия её достижения, иначе всегда можно считать, что она уже достигнута. Поэтому формулировка Цели всегда должна включать критерий достижения цели. Достижение Цели дает определенный ***Результат***.

Единственной физиологической теорией, в которой достижение Цели и получение Результата рассматривается как решение мозгом задачи по удовлетворению некоторой потребности, является ***Теория Функциональных Систем*** П.К Анохина [15-20]. Эта теория также выявляет физиологические механизмы достижения цели и решения этой задачи мозгом. П.К. Анохин писал: «Пожалуй, одним из самых драматических моментов в истории изучения мозга как интегративного образования является фиксация внимания на самом действии, а не на его результатах … мы можем считать, что результатом «хватательного рефлекса» будет не само хватание как действие, а та совокупность афферентных раздражений, которая соответствует признакам «схваченного» предмета». «Совокупность афферентных раздражений» и есть критерий достижения цели в ТФС. Целью в ТФС является удовлетворение некоторой потребности: «Каждая потребность, даже при незначительном отклонении жизненно важной функции от оптимального для метаболизма уровня (в чём и проявляется потребность), немедленно воспринимается специальными рецепторными аппаратами» (критерием достижения цели).

Согласно П.К.Анохину, центральные механизмы функциональных систем, обеспечивающих целенаправленные поведенческие акты, имеют однотипную архитектуру.

2.1.2.1. ***Афферентный синтез***. Начальную стадию поведенческого акта любой степени сложности составляет афферентный синтез, включающий в себя синтез мотивационного возбуждения, памяти, обстановочной и пусковой

афферентации.

Мотивационное возбуждение. Постановка цели осуществляется возникшей потребностью, которая трансформируется в мотивационное возбуждение.

Память. Мотивационное возбуждение «извлекает из памяти» все возможные способы достижения цели, а также всю последовательность и иерархию результатов, которые должны быть получены для достижения цели некоторым конкретным способом.

Обстановочная афферентация. При достижении цели, фиксируется и та обстановка, в которой удалось получить результат. Эта обстановка фиксируется как необходимые условия наряду с мотивацией требуемые для достижения результата. Поэтому мотивационное возбуждение в данной обстановке «извлекает из памяти» только те способы достижения цели, которые возможны в данной обстановке. Таким образом, обстановочная афферентация, при взаимодействии с извлеченным из памяти опытом, определяет, что и как можно сделать в данной обстановке для достижения цели.

Пусковая афферентация. Пусковая афферентация также является обстановочной афферентацией, только связанной со временем и местом достижения результата. Пусковая афферентация отвечает на вопрос, где и когда можно достичь результат.

2.1.2.2. ***Принятие решений***. На стадии афферентного синтеза мотивационным возбуждением может быть извлечено из памяти (в данной обстановке) несколько способов достижения цели. На стадии принятия решения выбирается только один из этих способов – конкретный план действий. «Вытягивая» из памяти весь накопленный опыт, мотивационное возбуждение преобразуется в конкретную цель, определяющую способ своего достижения. Конкретная цель называется в ТФС «высшей мотивацией».

2.1.2.3. ***Акцептор результатов действия***. Мотивационное возбуждение «извлекает из памяти» также всю последовательность и иерархию результатов, которые должны быть получены для выполнения плана действий. Эта последовательность называется в ТФС акцептором результатов действия. Акцептор результатов действия представляет собой доминирующую потребность (Цель) организма, трансформированную в форме опережающего возбуждения мозга, как бы в своеобразный комплексный рецептор будущего подкрепления, являющийся критерием достижения конкретной цели.

2.1.2.4. ***Подкрепление. Санкционирующая стадия*.** Если в результате выполнения конкретного плана действий цель будет достигнута (потребность удовлетворена) и все результаты акцептора результатов действия получены, то возникает последняя санкционирующая стадия, в которой осуществляется удовлетворение потребности и занесение выполненного конкретного плана действий в память.

2.1.2.5. ***Формальная модель функциональных систем*.** Основываясь на работах [16-17,20,23] по формализации функциональных систем, была разработана формальная модель, суммирующая основные свойства функциональных систем:

1. использует формальную модель нейрона, обнаруживающую причинные связи и основанную на семантическом вероятностном выводе;
2. осуществляет постановку цели в целенаправленном поведении, формирует функциональную систему и акцептор результатов действия, которые непрерывно сверяют достигнутые результаты с ожидаемыми в акцепторе результатов действия;
3. автоматически формирует подцели и подкрепляет достижение подцелей, если их достижение увеличивает вероятность достижения конечной цели;
4. моделирует сенсорные коррекции;
5. выбирает действие в реальном режиме времени с учетом текущей ситуации и получаемой афферентации;
6. планирует достижение цели в соответствии с последовательностью и иерархией функциональных систем по достижению всех результатов, требуемых для достижения конечной цели;
7. принимает решение об определенном способе достижения цели.

Проведенные эксперименты показывают «естественность» и эффективность поведения полученных аниматов.

Данная модель успешно применялась для моделирования аниматов [20,23-25].

2.2. *Математическая теория задачного подхода - семантическое моделирование*

Перейдем теперь к формальным определениям. Определим язык спецификаций задач, опираясь на логико-математическую теорию задачного подхода - семантическое моделирование. В качестве концепции базовой модели вычислений предлагается взять концепцию ∑-*определимости вычислений* [26], дополнив ее *процедурой проверки истинности* ∑-*формул на многосортной конструктивной модели M*, рассматриваемой совместно с ее списочной надстройкой *HW(M*), и *концепцией относительной вычислимости, допускающей использование оракул*ов [27-32].

Пусть M некоторое множество. Определим по индукции множество HW(M) *наследственно конечных списков* над M:

$HW_0(M) = \{\varnothing\}$;

$HW_{n+1}(M) = HW_n(M) \cup Lisp(HW_n(M) \cup M))$;

$HW(M) = n = 0\omega HW_n(M)$,

где $Lisp\omega(X)$ есть множество всех конечных списков, состоящих из элементов множества X. Пусть $\mathcal{M}$ многосортная конструктивная модель сигнатуры σ, где

M - базисное множество этой модели и пусть (HW(M)∪M) есть базисное множество расширенной модели HW($\mathcal{M}$) сигнатуры σ* = σ ∪ {nil, head, tail, cons, conc, =, ϵ, ≤, U}, где новые функциональные символы имеют естественную интерпретацию операций со списками, а предикатные символы интерпретируются как отношения, где = есть отношение равенства списков, ϵ - отношение «быть элементом списка», ≤ - отношение «быть началом списка» и где U(HW(M)) = M. Будем называть модель *HW*($\mathcal{M}$) расширенной сигнатуры σ* *надстройкой наследственно конечных списков для модели* $\mathcal{M}$ или просто *списочной надстройкой над* $\mathcal{M}$. Далее мы будем рассматривать многосортную конструктивную модель $\mathcal{M}$ вместе со своей списочной надстройкой HW($\mathcal{M}$) как некий исходный *базовый вычислитель*, у которого все сигнатурные сущности представляют собой вычислимые конструкции. Кроме того, мы допускаем также возможность обогащения сигнатуры σ* дополнительными предикатными и функциональными символами, часть которых будет использоваться как предикатные и функциональные переменные, а у остальных в качестве их интерпретаций выступают некие *внешние* вычислимые конструкции, называемые нами в дальнейшем *оракулами*.

Естественным образом в нашем языке определяются термы и атомарные формулы сигнатуры σ*. Класс Δ0–формул сигнатуры σ* определяется как наименьший класс формул, содержащий все атомные формулы и замкнутый относительно ¬, □,□, →, □x∈a, ∃x∈a, ∀x≤a, ∃x≤a, где a – это терм. Напомним, что здесь «∈» означает «быть элементом списка», а «≤» – «быть начальным подсписком». Выражения вида ∀x∈a, ∃x∈a, ∀x≤a, ∃x≤a далее мы будем называть ограниченными кванторами.

Имея класс Δ0–формул определим теперь класс ∑–формул сигнатуры σ* как наименьший класс формул, содержащий все Δ0–формулы и замкнутый относительно ∧, ∨, ∀x∈a, ∃x∈a, ∀x≤a, ∃x≤a и неограниченного квантора существования ∃x. Заметим, что в нашем языке мы вполне корректно можем писать также формулы, у которых добавлены ограниченные кванторы вида «∀x=t» и «∃x=t», где t – терм. Нетрудно убедиться в том, что такие выражения логически эквивалентны по своему смыслу оператору присваивания «x:=t» в традиционном программировании.

Некое отношение в *HW*($\mathcal{M}$) будем называть ∑*–отношением*, если оно является ∑–определимым, и Δ_0*–отношением,* если оно Δ_0–определимо. Частично определенная функция является ∑*–функцией* (Δ_0*–функцией)* в *HW*($\mathcal{M}$) если ее график ∑–определим (Δ_0–определим).

Введем теперь очень важное для дальнейшего изложения понятие формульной определимости. Пусть задана некоторая модель $\Re$ = (R; $P^{n_0}_{m_0}, \ldots, P^{n_k}_{m_k}$). Будем говорить, что модель $\Re$ является Σ–определимой над HW($\mathcal{M}$), если существуют Σ–определимое подмножество S ⊆ HW($\mathcal{M}$) и отображение μ : S → R такие, что μ–прообразы предикатов $P^{n_0}_{m_0}, \ldots, P^{n_k}_{m_k}$ и =, а

также μ–прообразы их дополнений являются Σ–отношениями в HW($\mathcal{M}$) (как правило, с параметрами). Данное определение можно корректно обобщить и на случай моделей, в сигнатуру которых помимо предикатных входят и функциональные символы. Важность данного определения заключается в том, что оно позволяет нам формально задавать модель некой предметной области как набор ∑-определений нашего языка, т.е. как набор ∑-формул и ∑-термов. При этом допускаются рекурсивные схемы ∑-определений с некоторыми ограничениями на вхождения в них определяемых предикатов и термов. Запрос к модели предметной области мы определим также как ∑-формулу, в записи которой могут использоваться как сигнатурные конструкции базовой конструктивной модели, так и определяемые предикаты, и термы предметной области.

Приведем некоторые факты, характеризующие исключительную выразительную и вычислительную силу введённых нами понятий и конструкций [28-31,33-41]:

- Если исходная модель $\mathcal{M}$ является вычислимой (полиномиально вычислимой), то ее списочная надстройка $HW(\mathcal{M})$ также является вычислимой (полиномиально вычислимой).
- Если модель $\mathcal{M}$ полиномиально вычислима, то в ее списочной надстройке $HW(\mathcal{M})$ функции, определяемые термами, являются также полиномиально вычислимыми.
- Если исходная модель $\mathcal{M}$ является вычислимой, то в ее списочной надстройке $HW(\mathcal{M})$ ∑-отношения являются *рекурсивно перечислимыми*, а ∑-функции - *вычислимыми*. Данный факт объясняет содержание термина «формульная определимость вычислимости».
- Если модель $\mathcal{M}$ полиномиально вычислима, то в ее списочной надстройке $HW(\mathcal{M})$ истинность каждой Δ_0–формулы $\varphi(x_1, \ldots, x_n)$ полиномиально вычислима.
- Для разрешимой модели $\mathcal{M}$ и ее вычислимой списочной надстройки $HW(\mathcal{M})$ их теории, к сожалению, *неразрешимы***.**
- В модели HW($\mathcal{M}$) для n-местных ∑–отношений существует универсальное (n+1)-местное ∑–отношение $P^{n+1}(x; x_1,\ldots, x_n)$ такое, что для каждого ∑–отношения $Q^n(x_1,\ldots,x_n)$ найдется элемент **a** такой, что $Q(x_1,\ldots,x_n) \leftrightarrow P(\mathbf{a}; x_1,\ldots,x_n)$. В тоже время универсальных вычислимых функций в модели $HW(\mathcal{M})$ не существует.
- Пусть $\varphi(x_1 , \ldots, x_n; Q+)$ есть ∑-формула с позитивным вхождением предиката $Q(x_1,\ldots,x_n)$ и пусть предикат Q является n-местным отношением в списочной надстройке $HW(\mathcal{M})$. Определим оператор G на $HW(\mathcal{M})$:

$G(Q) = \{(a_1, \ldots, a_n) | HW(\mathcal{M}) \models \varphi(a_1, \ldots,a_n; Q+)\}$.

тогда оператор G является монотонным и имеет наименьшую неподвижную точку, которая ∑-определима. Отметим, что именно эта,

чрезвычайно важная, теорема Ганди позволяет при формальном определении предметных областей использовать рекурсивные конструкции. При этом, на позитивно входимый в ∑-формулу $\varphi(x_1, \ldots, x_n; Q+)$ предикатный символ Q+ часто целесообразно смотреть либо как на определяемый рекурсивно предикат, либо как на вызов оракула, допуская тем самым в нашей концепции семантической вычислимости так называемую *относительную вычислимость* – исключительно сильный инструмент моделирования, в частности сложных иерархических структур.

На практике предлагается ограничиться классом Δ_0-формул**.** Заметим, что с помощью Δ_0 –определений можно значительно повысить выразительную силу нашего исходного языка путем введения в рассмотрение так называемых *условных* и *рекурсивных* Δ_0 – *термов* [34-35]. Так, например, если φ - Δ_0-формула, а t_1 и t_2 – термы, то тройка (φ,t_1,t_2) есть условный Δ_0 *–терм,* интерпретацией которого служит следующая вычислительная схема: если φ истина, то исполняется терм t_1, в противном случае – терм t_2. С использованием различных итерационных и рекурсивных схем вычислений похожим образом определяются рекурсивные термы [35]. Обогащенный подобным конструкциями язык Δ_0–определений предстает как весьма выразительный *логико-функциональный язык моделирования*. И что очень важно – такое обогащение сохраняет исходную полиномиальную вычислимость. Более того, было показано, что при определенных условиях верно и обратное – каждая полиномиально вычислимая функция является Δ0 –определимой [39]. Для этого будем использовать более приемлемый на практике язык спецификаций задач в классе Δ_0-формул. Будем говорить, что задача ***P* формульно представима** **в** *HW*($\mathcal{M}$), если:

- *предметная область*_может быть задана в виде Δ_0-модели ***P*** сигнатуры списочной надстройки *HW*($\mathcal{M}$), т.е. как набор Δ_0-определений (еще раз напомним, что допускаются рекурсивные Δ_0–определения);

- указан *запрос* к Δ_0-модели ***P*** либо в виде Δ_0-формулы **φ**, либо в виде Δ_0-терма **t**.

- в качестве *ответа* на запрос (т.е. *решения задачи* ***P***):

- если *запрос* к Δ_0-модели ***P*** имеет вид Δ_0-формулы **φ,** то ищутся конкретизации свободных переменных **φ**, при которых формула **φ** является *истиной,*

- если *запрос* имеет вид Δ_0-терма **t,** то *вычисляется* значение Δ_0-терма **t**.

Здесь важно подчеркнуть, что истинность Δ_0-формулы запроса, получаемой подстановкой констант вместо переменных, и есть критерий решенности задачи ***P***. Заметим, что наборов констант, делающих Δ_0-формулу запроса истинной, может оказаться несколько и потому есть возможность выбора в каком-то смысле «наилучшего» решения с учетом контекста решения

задачи. В этом случае критерий решенности задачи ***P*** должен содержать и критерий выбора наилучшего ответа на запрос. Так понимаемый подход к формулировке и решению задач и носит название ***семантического моделирования***. Важно заметить, что описанный выше формализм допускает свое естественно расширение путем введения вероятностных и нечетких конструкций.

2.3. *Индуктивный вывод знаний на модели предметной области*

Как отмечалось в [41-46], предметная область может быть задана эмпирической системой $\mathfrak{I} = \langle A, \Omega \rangle$, где A – объекты Предметной Области (ПО), а Ω – множество отношений и операций, интерпретируемых в системе понятий ПО. Множество Ω представляет собой *онтологию предметной области*.

«Знания – это воспринятая, осознанная и ставшая личностно значимой информация» [47]. Поэтому для индуктивного вывода знаний необходимо *понимание* и *интерпретация* человеком предметной области и её онтологии.

Индуктивный вывод знаний на модели предметной области, осуществляемый любым методом машинного обучения, должен уметь правильно обрабатывать свойства и атрибуты объектов ПО, чтобы получить знания в терминах онтологии Ω. Следует заметить, что сами по себе числовые значения величин знания и информацию не содержат, т.к. смысл величин указывается в их интерпретации, например, 5 метров, 5 литров, 5 килограмм и т.д. Интерпретация, в частности, определяет какие математические действия можно с ними осмысленно проводить. Стоит заметить, что физические величины, измеренные в предметной области, отличной от физики, теряют свою исходную физическую интерпретацию. Рассмотрим, например, такую физическую величину как температура. Шкалы температур в нефизических областях, например, при измерении температуры тела больного в медицине, температуры почвы в сельском хозяйстве или температуры воздуха в духовке в кулинарии и т.д., должны быть разные, хотя измеряться могут одним и тем же прибором – термометром. Шкала некоторой величины, с точки зрения теории измерений, – это набор отношений и операций, которые имеет смысл производить с числовыми значениями величин в данной предметной области, т.е. это те отношения и операции, которые интерпретируемы в онтологии ПО. Можно возразить, что термометр не может измерять ничего кроме температуры. Он действительно во всех случаях измеряет физическую температуру. Но температура, как и любой другой прибор, нужны для *получения выводов (знаний) в системе понятий* (онтологии) той предметной области, к которой он относится. Физическая температура больного – есть косвенное измерение медицинской величины – уровня обмена веществ. Для почв температура интерпретируется в системе понятий физиологии растений и деятельности микроорганизмов. Физическая температуры в других предметных областях

является косвенным измерением некоторой величины, интерпретируемой в системе понятий данной предметной области, которую мы и хотим измерить. Какие отношения и операции над числовыми значениями температуры имеют смысл для этих величин, определяется уже онтологиями соответствующих ПО. Таким образом, для извлечения информации из атрибутов и величин ПО нужно определить множество интерпретируемых в онтологии Ω математических отношений и операций над этими величинами и включить их в онтологию Ω предметной области $\Im = \langle A, \Omega \rangle$.

Рассмотрим задачу обнаружения теории $Th(\Im)$ ПО, т.е. теорию эмпирической системы $\Im$. Будем предполагать, что теория $Th(\Im)$ представляет собой совокупность универсальных формул (более общий случай рассмотрен в [47]). Известно, что совокупность универсальных формул логически эквивалентна совокупности правил вида:

$$\forall x1,\ldots,x2(A1\&...\&Ak \Rightarrow A0),\ k \geq 0, \quad (1)$$

где A0,A1,…,Ak – литералы. Поэтому можно считать, что теория $Th(\Im)$ представляет собой совокупность правил вида (1).

Нетрудно видеть, что правило $C = (A1\&...\&Ak \Rightarrow A0)$ логически следует из любого подправила вида: $(Ai1\&...\&Ain \Rightarrow A0)$, где $\{Ai1,...,Ain\} \subset \{A1,...,Ak\}$, $0 \leq h < k$, то есть: $(Ai1\&...\&Ain \Rightarrow A0) \vdash (A1\&...\&Ak \Rightarrow A0)$. Тогда теорию $Th(\Im)$ можно упростить. Законом эмпирической системы $\Im = \langle A, \Omega \rangle$ будем называть истинное на $\Im$ правило C вида (1), для которого каждое его подправило уже не истинно на $\Im$. Тогда верно, что:

$L \vdash Th(M)$ [16,48].

Тогда теорию $Th(\Im)$ можно рассматривать как совокупность законов эмпирической системы.

Определим вероятность η на эмпирической системе $\Im = \langle A, \Omega \rangle$ как на модели [49]. Вероятностным законом на $\Im$ будем называть правило $(A1\&...\&Ak \Rightarrow A0)$ условная вероятность $\eta(A0\&A1\&...\&Ak)/\eta(A1\&...\&Ak)$ которого определена $(\eta(A1\&...\&Ak) > 0)$ и строго больше условных вероятностей каждого из его подправил. Сильнейшим Вероятно*стным Законом* (СВЗ) будем называть вероятностный закон C, который не является подправилом никакого другого вероятностного закона.

Индуктивный вывод вероятностного знания на предметной области $\Im = \langle A, \Omega \rangle$ осуществляется следующим семантическим вероятностным выводом.

Семантическим Вероятностным Выводом (СВВ) некоторого сильнейшего вероятностного закона Cn, предсказывающего некоторый факт G, будем называть последовательность $C1 \sqsubset C2 \sqsubset ... \sqsubset Cn$, $Ci = (Ai1\&\ldots\&Aiki \Rightarrow G)$ вероятностных законов, для которой правило Ci является подправилом правила Ci+1 и $\eta(Ci) < \eta(Ci+1)$, $i = 1,2,...n-1$.

Знания нужны прежде всего для предсказания. Определим теперь сильнейшие вероятностные законы, решающие проблему статистической

двусмысленности [48,50] и предсказывающие без противоречий.

Рассмотрим множество всех сильнейших вероятностных законов, предсказывающих некоторый факт G. Это множество можно представить семантическим вероятностным деревом вывода факта G.

Максимально Специфическим Вероятностным Законом МСВЗ(G) вывода некоторого факта G будем называть сильнейший вероятностный закон, принадлежащий некоторому семантическому вероятностному дереву вывода факта G, имеющий максимальное значение вероятности среди всех правил дерева. Множество всех максимально специфических законов МСЗ(G) для всех литералов $G \in \Omega$ обозначим через МСЗ.

Можно доказать, что $L \subset$ МСЗ [41,47-48] и поэтому множество законов МСЗ обобщает теорию $Th(\Im)$. Кроме того, МСЗ логически непротиворечиво [47-48,50] и поэтому в точном смысле является вероятностной теорией предметной области $\Im = \langle A, \Omega \rangle$. Можно доказать [47-48,50], что предсказания, получаемые индуктивно-статистическим выводом по законам из МСЗ, непротиворечивы.

Данная теория обнаружения индуктивных знаний на эмпирической системе $\Im = \langle A, \Omega \rangle$ ПО реализована в виде платформы и программной системы «Discovery», приведенной далее. С ее помощью было решено множество практических задач (http://old.math.nsc.ru/AP/ScientificDiscovery/index_rus.html).

3. *Платформенные решения задачного подхода*

В настоящее время семантическое моделирование, как одна из концепций автоматического решения интеллектуальных задач, опирается не только на методологию и теорию задачного подхода, но и имеет в своем распоряжение развитый инструментарий, ориентированный на поддержку и сопровождение следующей технологической схемы решения интеллектуальных задач:

● *ШАГ 1*. Уточняется возникшая *потребность*, выявляется и изучается связанное с ним *противоречие*, связанное с отсутствием «шаблонного» способа удовлетворения данной потребности. Отметим, что это противоречие, по своей сути, и является истинной причиной и содержанием решаемой задачи. Далее, на естественном языке формулируется *критерий успешности* преодоления выявленного противоречия (прототип *критерия решения задачи*), при необходимости осуществляется декомпозиция противоречия/задачи и определяется *контекст задачи* (почему, зачем, ближайшая надзадача, подзадачи, цель и последствия решения/нерешения задачи и т.д.).

● *ШАГ 2*. На естественном языке описываются релевантные задаче *общие знания*, строится *онтологическая модель* проблемной области (понятия, факты, правила, отношения, …), формулируется в общих и онтологических терминах класс запросов к проблемной области.

● *ШАГ 3*. В логико-вероятностных терминах семантического моделирования строится *формальная модель* проблемной области задачи и в этих же формальных терминах формулируется *запрос* и *критерий решения*.

● *ШАГ 4*. В рамках и средствами соответствующей инструментальной платформы семантического моделирования строится *компьютерная модель* задачи.

● *ШАГ 5*. С помощью компьютерной модели находятся *ответы* на запросы. Проверяется выполнимость критерия решения, *обоснованность и/или объяснимость* ответов и их *интерпретируемость.*

Что же касается технологического инструментария семантического моделирования, то созданы и активно развиваются несколько платформенных решений.

3.1. *Платформа D0SL*

Примером успешного применения задачного подхода и его логико-математической теории — семантического моделирования может служить семантическая платформа **D0SL**, представляющая собой следующее поколение так называемых Business Rules Engines (BRE) [www.d0sl.org]. D0SL, так же, как и обычная платформа BRE, позволяет реализовать бизнес-логику системы с помощью языка декларативных логических спецификаций. Проект создания платформы D0SL осуществлялся под руководством В.Ш.Гумирова.

Платформа d0sl позволяет управлять логикой поведения сложных систем с помощью языка d0sl, понятного специалисту предметной области. Платформа имеет широкую область применения — от бизнес-процессов предприятия до управления проектами или поведением автономных систем, в том числе систем искусственного интеллекта и интернета вещей. Заметим, что язык d0sl легко расширяется функциями и объектами, характерными для конкретной предметной области. Более того, d0sl сам по себе является предметно-ориентированным языком (DSL). Существование механизма расширения на уровне языка d0sl позволяет построить иерархию языков DSL, которые могут быть преобразованы в d0sl и, следовательно, могут выполняться ядром d0sl.

Что касается приложений, то язык d0sl активно используется в обработке транзакций и событий систем в телекоммуникациях, финтехе и банковском деле, что позволяет легко изменять и настраивать бизнес-логику создаваемых приложений. Использование платформы d0sl оказалось особенно эффективным при цифровой трансформации предприятий, позволяя быстро и качественно разрабатывать автономные системы поддержки принятия решений, системы предиктивной аналитики, моделирования и цифровых двойников различных объектов, в том числе самих организаций.

Следует отметить, что язык d0sl является одним из основных технологических инструментов открытой платформы Eyeline Mobilizer,

предназначенной для разработки и эксплуатации сервисов с поддержкой мобильных устройств на всех типах платформ, включая USSD/SMS/SS7, смартфоны (Android, iOS), мобильный Интернет [www.eyeline.ru]. С использованием платформы Eyeline Mobilizer разработаны решения и реализованы проекты в сфере телекоммуникаций, мобильного маркетинга и рекламы, финансов, банковских услуг, муниципального управления и мобильных платежей. В настоящее время решения и сервисы на базе платформы Eyeline Mobilizer обслуживают десятки миллионов пользователей мобильных устройств. Платформа позволяет значительно сократить расходы на разработку и эксплуатацию мобильных решений для вашего бизнеса. Примерами успешных внедрений являются: USSD-сервисы для МТС (*100#, *111# и др.) и USSD-портал Альфа-Клик *142# Альфа-Банка (Россия), InnomaNet/T-mobile (Россия), WATAGO Africa (все крупнейшие мобильные платежные системы Нигерии), проект «Парковка» Правительства Москвы (Россия), система оповещения о пропущенных звонках, Yellow Pages/SMART на Филиппинах, облачная установка для обслуживания клиентов GlobalUSSD.com, Билайн Россия (прямые деньги). трансферы) и многое другое.

3.2. *Платформа bSystem*

bSystem является платформой для построения цифровых двойников организаций и процессов. Она развивается исследовательской группой А.В.Манциводы в течение ряда лет. Платформа ориентирована на создание интеллектуальных систем управления крупными бизнес-экосистемами, цифровой трансформации предприятий и иных интегральных решений. Благодаря единой среде семантического моделирования, bSystem активно интегрирует базовый уровень онтологий с логико-вероятностным индуктивным выводом знаний и семантическим вероятностным выводом прогнозных правил и решений Е.Е. Витяева и классическим программированием.

Архитектура bSystem построена на интеграции трех уровней:

логико-вероятностного,

аналитико-онтологического,

операционно-транзакционного.

Центральный – аналитико-онтологический уровень [51-52] – контролирует основные параметры управляемого объекта, отражает текущее состояние его компонент и связей между ними, предоставляет инструменты для аналитической работы, в частности, для принятия решений операционного уровня (к ним, например, относятся задачи бизнес-аналитики). Этот уровень формирует ядро платформы, на нем идет управление онтологиями – базовыми системами знаний о предметной области, вокруг которых строится вся работа.

Над центральным уровнем надстраивается уровень логико-вероятностного вывода (ЛВВ). Он реализует функционал объясняющего искусственного

интеллекта. В рамках платформы ЛВВ отвечает за стратегические вопросы прогнозирования, принятия решений и другие задачи ИИ. Его ценность для bSystem состоит в том, что в отличие от нейронных сетей, которые в процессе обучения “накапливают” в нейронах числовые коэффициенты, не имеющие семантической интерпретации, ЛВВ в процессе обучения формирует вероятностно-логические теории (наборы закономерностей), семантически описывающие предметную область. Отсюда и способность ЛВВ объяснять свои решения. Это также в потенциале обеспечивает “самоконтроль” – способность оценить с помощью логических мета-средств достоинства и недостатки знаний, сформированных в процессе обучения. Такие возможности абсолютно недостижимы для нейронных сетей.

Наконец, операционно-транзакционный уровень – это уровень процедур и методов, изменяющих систему во времени. На этом уровне в bSystem работает объектно-ориентированный язык Libretto. С точки зрения задачного подхода, операционно-транзакционный уровень формирует среду управления жизненным циклом задач. Любой бизнес-процесс, который контролируется на этом уровне, есть не что иное, как процесс решения некоторой задачи в реальном времени (на практике – экземпляра некоторой обобщенной задачи). Функционирование цифрового двойника, разворачивающееся во времени, есть управление конгломератом взаимодействующих друг с другом активных задач, генерацией новых экземпляров задач и закрытием задач с выполненным условием решенности. И если декларативные постановка задачи и критерий её решения управляются на более высоком онтологическом уровне, то сам процесс решения через систему шагов-транзакций осуществляется на уровне операционном. При этом сами бизнес-процессы также представлены в виде объектов онтологии, что позволяет на декларативном уровне контролировать процесс решения. Таким образом, bSystem через взаимодействие онтологического и операционного уровней дает полноценную реализацию задачного подхода.

Ключевой особенностью bSystem является то, что все три уровня погружены в единое семантическое пространство, взаимодействуют друг с другом и обеспечивают вертикальное масштабирование. Данная интеграция несет мощный синергетический заряд, не только помогая улучшать количественные показатели (за счет бесшовного взаимодействия там, где обычно используется зоопарк различных информационных технологий), но и позволяя качественно изменять ситуацию. Например, такая интеграция делает возможной реализацию принципиально новой low-code технологии – объектного low-code, основанного на автоматическом синтезе кода по объектной онтологии [53]. Это принципиально отличает его от традиционного low-code, основанного на диаграммах – вариациях спецификации BPMN [54].

Центральным инструментом, обеспечивающим целостность цифровой

среды bSystem являются онтологии. Как правило, цифровой двойник – это онтология или сеть онтологий, которые играют роль ядра семантического моделирования, определяя базовые объекты и связи между объектами описываемой предметной области и решаемых задач.

Онтологию можно рассматривать как хаб, который обеспечивает взаимодействие разнообразных компонент цифрового двойника. На логико-вероятностном уровне онтологии играют роль эмпирических систем, из которых берется информация для обучения. На центральном – аналитико-онтологическом уровне – онтология является семантической основой для цифрового двойника, источником информации для аналитических манипуляций. Наконец, на процедурно-транзакционном уровне онтология служит системой типов данных для языка программирования платформы – Libretto.

Поиск подходящей архитектуры для онтологий оказался отнюдь не простой задачей. Достаточно сказать, что bSystem стала шестой (и лишь первой достаточно эффективной) попыткой построить платформу управления онтологиями такого рода. Чтобы понять, какие онтологии нужны, нами была выработана система требований, которым должна удовлетворять онтология. Приведем некоторые из них.

Во-первых, как отмечалось выше, онтологии ответственны за обеспечение вертикальной связности и единого семантического пространства платформы. Знания из онтологии должны иметь эффективную интерпретацию на всех уровнях bSystem – и в тонких методах логико-вероятностного вывода, и в аналитических модулях (например, в рамках бизнес-аналитики), и в программных процедурах, решающих традиционные алгоритмические задачи.

Во-вторых, с онтологиями будут работать люди, причем, в основном, мало понимающие в математической логике (именно трудности с пониманием формализмов стали одним из ключевых барьеров, ответственных за неудачи хорошо известной концепции Semantic Web [55]). Поскольку онтология задает базовые знания о предметной области в среде взаимодействия людей и информационных систем, то если люди онтологию не понимают, все остальное становится бессмысленным. Поэтому такое строение онтологии, которое обеспечивает интуитивное понимание онтологической информации, согласованное с каждодневным опытом пользователя, также является необходимым условием.

Далее, онтология должна обеспечить стандартизированный механизм взаимодействия с внешними оракулами (информационными системами, нейронными сетями, интернетом вещей). Например, сбор информации в системе контроллеров компании, управляющей жилым комплексом, является одной из основ функционировании её цифрового двойника. Для взаимодействия двойника с внешним миром онтология должна быть снабжена логическими

аналогами хорошо известных программных интерфейсов (API).

Наконец, если мы хотим, чтобы наши онтологии были востребованы, они должны быть эффективными. Это означает, что высокоуровневые онтологии должны иметь реализацию, сравнимую по продуктивности с реляционными базами данных. Онтология должна обладать способностью содержать миллионы сущностей, и выдавать ответы на запросы практического уровня сложности в реальном времени.

Совмещение перечисленных требований в едином формализме оказалось довольно непростой задачей (например, онтологии, базирующееся на хорошо известном языке OWL [56], не соответствуют нескольким из них). В процессе экспериментов оказалось, что в наибольшей степени к данным требованиям подходят онтологии, которые основаны на идеях объектного моделирования.

Особенность объектных моделей состоит в том, что в них четко выражены два слоя – декларативный и процедурный. Первый состоит из классов, объектов, полей или свойств объектов. Второй – из методов, определенных на уровне классов и действующих на объектах. Поскольку онтологии – это формальные описания базовых концептов, фактов и связей предметной области, а объектные модели занимаются аналогичными задачами на уровне программирования, то декларативный слой объектных моделей оказывается вполне адекватным кандидатом, чтобы превратиться в онтологию. Тем более, что суть объектных моделей весьма интуитивна, естественна и привычна людям. Это доказывается феноменально широким распространением объектно-ориентированных языков. Поэтому вполне можно выделить декларативный слой объектных моделей и, придав ему строгую логическую семантику, превратить его в вариант онтологии.

Что касается процедурного слоя, объектно-ориентированные языки обычно полны по Тьюрингу и это делает ситуацию мало контролируемой. С другой стороны, наличие процедурного слоя позволяет реализовать идею развития онтологии, управления её жизненным циклом. С этой точки зрения применение каждого метода или группы методов можно рассматривать как транзакцию, переводящую онтологию из одного состояния в другое [57]. Особенно полезна эта возможность при решении управленческих задач, поскольку такая интерпретация естественным образом определяет бизнес-процессы как цепочки транзакций в жизненном цикле онтологий [58].

Итак, чтобы превратить объектные модели в онтологии нам нужно сделать следующие шаги:

1. Описать логический формализм, соответствующий декларативному слою объектной модели. Для этого используем механизмы Δ0-определимости семантического моделирования.

2. Традиционные объектные модели короткоживущие, они содержатся только в оперативной памяти. Нам необходимо сделать их долгоживущими,

поскольку онтологии – долговременные хранилища данных и знаний. Это обеспечивает им функционал объектной базы данных.

3. Для работы с онтологией необходим язык запросов, сочетающий логическую выразительность с эффективностью, сравнимой с эффективностью SQL в реляционных базах данных. В bSystem в качестве такого языка используется декларативное ядро Libretto, имеющее строгую логическую семантику.

4. Для того чтобы обеспечить механизмы изменения и развития онтологии во времени, переформатируем процедурный уровень объектной модели в транзакционную систему управления содержимым онтологии.

5. Практика показала, что инструменты управления жизненным циклом объектов серьезно усиливают функционал по сравнению с традиционными объектными моделями, что особенно полезно при управлении бизнес-процессами. Используя транзакционные механизмы, организуем контроль за ЖЦ объектов.

6. Финальный шаг – это автономизация онтологий. Онтология сворачивается в кокон, закрывается от внешнего мира, с которым теперь общается только через строго регламентированный интерфейс (аналог программистского API).

С точки зрения задачного подхода пункт 5 реализует двухуровневое управление жизненным циклом задач, которые исполняются на нижнем операционном уровне и семантически контролируются на уровне онтологическом.

Последний шаг реализует концепцию локальной простоты [52] – построение сколь угодно сложных систем как сетей относительно простых онтологий с логически регламентированными механизмами взаимодействия. Это обеспечивает свободное горизонтальное масштабирование при построении цифровых двойников.

С информационной точки зрения второй и последний шаги превращают объектную онтологию в микросервис [59], поскольку обеспечивают два основных свойства микросервиса – автономность хранилища данных (это у нас онтология) и строгий контроль за взаимодействием с внешним миром. Это задает и общую архитектуру цифровых двойников, которые строятся на основе bSystem.

bSystem позволяет использовать принципиально новую методологию разработки [60]. За счет открытости моделирование принципиально изменяет механизмы взаимодействия

участников разработки и эксплуатации систем. Онтология непосредственно доступна для всех участвующих компетенций. Например, заказчик получает полный контроль над процессом разработки и способен дать раннюю обратную связь с мгновенной детекцией проблем. Высокоуровневая модель, благодаря

своей строгой логической семантике, одновременно выполняет роль спецификации, исполняемой документации проекта. Это принципиально отличается от черного ящика программного кода.

3.3. *Платформа DISCOVERY - онтологический подход к индуктивному выводу знаний*

Как было сказано, Витяевым Е.Е. разработан онтологический [6,42,50-51] подход к методам извлечения знаний и, реализующая его, система «Discovery». Данный подход разработан для обнаружения знаний на информации, извлеченной из данных и представленной эмпирической системой ПО. Система «Discovery», обладает следующими преимуществами по сравнению с другими методами машинного обучения:

1) снимает ограничения с используемых типов данных за счет использования любой информации, извлеченной из данных в терминах Ω;

2) снимает ограничения с используемого априорного знания путем представления его в Ω;

3) снимает ограничения с классов проверяемых гипотез за счет задания класса обнаруживаемых знаний Rule Type, определенных в Ω;

4) система Discovery обнаруживает следующие виды закономерностей:

- множество законов L и теорию Th(ℑ) эмпирической системы ℑ;
- множество вероятностных и сильнейших вероятностных законов;
- множество максимально специфических законов, предсказывающих без противоречий [48,52];

В онтологическом подходе обнаруживаемые знания полны в двух смыслах: (1) в смысле полноты извлечения информации из данных за счет использования онтологии и теории измерений; (2) в смысле полноты обнаруживаемых знаний в виде множеств правил a)-c);

Существующие методы машинного обучения не в состоянии поддерживать режим исследования данных, когда обнаруживаемая закономерность заранее неизвестна. Каждый метод машинного обучения обнаруживает свой вполне специфический класс гипотез, соответствующий его онтологии. Система «Discovery» может обнаружить произвольный класс гипотез, который может задать эксперт, и, тем самым, способна поддерживать режим исследования данных, когда класс гипотез может меняться в процессе исследования.

Система «Discovery» обнаруживает знания в терминах онтологии ПО. Интерпретируемость закономерностей может быть очень важна при принятии ответственных решений в таких областях, как медицина, финансы или военные приложения.

Система «Discovery» применялась для решения множества задач в финансовом прогнозировании [6,42], биоинформатике [64], медицине [65], анализе жульничества [66] и многих других.

3.4. *Delta-платформа*

Как упоминалось во Введении одной из острых проблем современного ИИ, помимо хорошо известной проблемы «черный ящик», является так называемая *проблема централизации*, означающая появление тенденции сосредоточения власти и контроля над ИИ в руках небольшого числа юридических лиц. Для решения этой проблемы во Введении было предложено воспользоваться опытом применения *децентрализованных решений* из сферы криптовалют. При этом сами ИИ-алгоритмы предлагалось рассматривать как *умные контракты*, которые исполняются на блокчейн-подструктурах *мультиблокчейна*, что обеспечивало бы прозрачность всех операций, прозрачность обучающей выборки и прозрачность принятия решений. И, наконец, выдвигалось требование *полиномиальной вычислимости* ИИ-алгоритмов, что имеет очевидное прикладное значение. В целом, ставилась задача создания полиномиально вычислимой (во всех смыслах) системы доверительного семантического моделирования. Но чтобы приступить к ее практическому решению потребовалось несколько лет серьезных исследований.

Введённые в 2017 году С.С.Гончаровым условные термы [34] позволили понять, что термами могут быть не только стандартные термы логики предикатов первого порядка, но и другие, корректно заданные синтаксические конструкции. Это позволило С.С.Гончарову и Д.И.Свириденко в 2019 году ввести такие термальные расширения, как ограниченно-рекурсивные термы, B-while термы, итерационные термы и т. д. [33] и показать, что все эти термальные расширения не выводят за рамки полиномиальности. Но проблема p-полноты этих расширений так ими и не была решена. Однако уже в 2022 году С.С.Гончаровым и А.В.Нечесовым была найдена финальная синтаксическая конструкция, которая завершила построение p-полного языка семантического моделирования L [39]. Чуть ранее, в 2021 году, ими же был доказан полиномиальный аналог теоремы Ганди о наименьшей неподвижной точке [36], который говорил о том, что в рамках определенной Δ_0-системы можно найти такие условия на формулы, что классический результат теоремы Ганди становится справедливым и для этой системы. При этом, что очень важно, в этой системе наименьшая неподвижная точка построенного оператора будет p-вычислимой. Это дало еще больший толчок к дальнейшему развитию языка семантического программирования L и к развитию самой формальной Δ_0-системы.

Отметим, что полиномиальный аналог теоремы Ганди позволяет погружать любые индуктивно задаваемые синтаксические конструкции внутрь формальной системы, предлагая, как кодировку, так и средства оценки сложности такого хранения. Это позволило С.С.Гончарову, Д.И.Свириденко и А.В.Нечесову описать объектно-ориентированную версию p-полного

семантического языка программирования L* [40] и приступить к разработке специального фреймворка — **Delta-платформе,** вобравшей в себя, в том числе, и весь необходимый функционал для решения выше упомянутых идей: мультиблокчейны, смарт-контракты и язык L* на базе p-полного логического языка L. При этом, на разных уровнях мультиблокчейнов, при необходимости, можно задавать различные уровни доступа пользователей, задавать уровни прозрачности как самого блокчейна, так и смарт-контрактов, исполняемых на них. В силу того, что смарт-контракты являются программами специального вида в p-полном языке L*, то это заметно снижает уровень трудозатрат при их аудите. В самом же языке L* можно описывать работу как нейронных сетей [61], так и блокчейнов [62], что несомненно является плюсом, так как не требует никаких внешних программных средств для их реализации.

С точки зрения задачного подхода в p-полном языке L* мы можем не только описать саму задачу, но в рамках и средствами этого же языка и решить ее. В силу того, что база у нас блокчейны, то децентрализованные решения строятся достаточно просто. В качестве верхнеуровнего блокчейна может быть выбран самый безопасный и децентрализованный, например, блокчейн Биткоина или Эфириума. Блокчейны более низких уровней могут не иметь децентрализации и использоваться как обычная база данных, в том числе и на домашнем компьютере одного из пользователей. А поскольку не нужно будет достигать никакого консенсуса с другими блокчейнами, то это повлечет за собой высокую скорость записи..

Хотелось бы заметить, что создание Delta-платформы в интеграции с предсказатель-ными системами типа Discovery способно дать мощный толчок к построению ИИ-алгоритмов, которые по своей природе близки к алгоритмам общего искусственного интеллекта (AGI), что обеспечивает как высокую *самообучаемость* таких систем, так и высокое *доверие* к ним обычных пользователей. В этой комбинации отмеченных выше возможностей и видится реализация основных положений задачного подхода, когда ИИ-системы могут решать, как поставленные перед ними задачи, так и самостоятельно формулировать новые задачи, которые не были явно прописаны в алгоритмах ИИ. Все это позволяет нам говорить о возможности построения доверительного «общего» ИИ в рамках и средствами задачного подхода.

4. *Заключение*

Данную статью можно рассматривать как своеобразный отчет о результатах исследований в области задачного подхода и его практических применениях, осуществляемых коллективом авторов статьи, их учениками и коллегами, которые были начаты еще в 70-80-х годах прошлого столетия. За эти годы была проделана большая работа по развитию и практическому применению положений задачного подхода применительно не только к

различным проблемам оснований математики и ИИ, но и к таким областям как когнитивные науки, нейрофизиология, медицина, цифровая трансформация предприятий, автоматизация проектирования сложных систем, цифровые двойники и т.д. Как видно из вышесказанного, удалось не только существенно развить методологические и математические положения задачного подхода, но и создать его эффективную инструментально-технологическую базу, позволившую успешно решать широкий спектр задач из различных областей: телекоммуникации, бизнес, ритейл, финтех, генетика, геология, кибербезопасность, робототехника, медицина и пр. На повестке дня – создание доверительного «общего» ИИ. И как показывает опыт предыдущих лет есть все основания, что в ближайшее время такой ИИ будет создан. В том числе в рамках и средствами задачного подхода.

1. Introduction

After the events associated with the “ChatGPT revolution” and which opened up to the world not only new brilliant prospects and possibilities of artificial intelligence, but also new risks and dangers of its use that could lead to uncontrollable consequences, the topic of AI has become extremely fashionable and popular, actively discussed not only among specialists and professionals, but also in other circles, including government ones. And one of the central problems discussed in connection with such a rapid development of events was the problem of trust in AI systems, which actually means society’s recognition of the fact that modern artificial intelligence systems cannot guarantee the correctness of the solutions they offer to problems. Those it is recognized that these decisions may be false, erroneous, incorrect, inaccurate, deliberately misleading, etc.

The problem of trust in AI is a complex problem, representing a whole “tangle” of problems that prevent people from trusting AI algorithms. In this paper, only some of these problems will be touched upon, namely the “black box” problem, the centralization problem and the audit problem, since according to the authors, solving these problems can significantly increase people’s trust in AI systems.

Let us recall that the “black box” problem means the fact that some AI systems, when producing the result of solving a problem, do not explain it in any way. Moreover, most often the result presented by such an AI system either cannot be verified by the end user at all under any conditions, or the complexity of checking the result requires enormous resources and mental costs. And one more note related to the “black box” problem. The fact is that to completely solve the problem under discussion, it is not enough to make the mechanism for generating results by AI systems transparent. At the same time, we should try to raise the level of competencies of the users of these systems. After all, the high demand for the services of AI systems is supported precisely by users, who often do not pay much attention to the “black box” problem. Moreover, even if the algorithms of the same

neural network are transparent and the formation of the result is “explained” by some special additional program, it is far from a fact that the presented justification will be convincing and understandable to an ordinary, not sophisticated user. This especially applies to artificial neural networks with billions of parameters, with their huge number of weighting coefficients and activation functions. It seems that here it is necessary to use logic and logical rules that would not only look convincing and truthful, but could also explain to a person the considerations from which the result was formed.

Another acute problem of modern AI is the so-called problem of centralization, which means the emergence of a tendency to concentrate power and control over AI in the hands of a small number of legal entities, which can negatively affect development and innovation in this area. Moreover, centralized control over systems like ChatGPT can lead to such incorrect system settings that can lead to very serious consequences and cause harm to both the end user of the system and humanity as a whole. To solve this problem, it is proposed to take advantage of the experience of using decentralized solutions. An example is the use of blockchain in the field of cryptocurrencies. The most stable computer network in the world today is the Bitcoin blockchain network, which has been operating stably since January 3, 2009. In principle, blockchains, as databases for storing information, may well be suitable for AI solutions. At the same time, to avoid the well-known blockchain trilemma [1], multi-blockchains can be used. Let us recall that the first multi-blockchains were announced back in 2017 by the brothers Nikolai and Pavel Durov in their “White Paper” for the TON project [2]. It turned out that it is at various levels of multi-blockchains that the necessary indicators for scalability, security and decentralization can be achieved. AI algorithms themselves can be considered as smart contracts that are executed on the blockchain substructures of a multi-blockchain, which ensures the security, correctness and transparency of all operations, training samples and decision making.

In addition to the problems of the “black box” and centralization for modern AI, the problem of AI auditing is becoming increasingly relevant, which is that even with access to AI algorithms, we cannot always check how an AI system based on training on large arrays makes decisions data. It is clear that competent analysis of such software solutions will require highly professional specialists, who, as a rule, are in short supply and who must be specially trained.

Due to the above problems, there is a danger that the AI system may make decisions that are either incomprehensible to the user or simply not justified. It is for this reason that eXplanable artificial intelligence (XAI) arose, requiring AI systems to be transparent and explainable in all the results they produce. Many countries are currently trying to put this requirement into legal form. For example, in September 2021, the US National Institute of Standards and Technology approved 4 principles of explanatory artificial intelligence [3]:

● Explanation: AI systems should provide explanations that are backed by evidence.

● Meaningful: Explanations should be meaningful in a way that can be understood by users of the AI system

● Accuracy: Explanations should be accurate in describing the AI system's process

● Boundaries: AI systems should operate within the limits that they were designed for

These principles were taken into account by the Chinese government, and already in April 2023, Beijing set a course for XAI [4]. China has automatically imposed restrictions on the AI development of all its AI companies. Now developer companies will be responsible for their AI solutions (including the actions of chatbots) if:

● original data source are illegitimate

● for all the results of issuing chatbots

● for content that would violate the basic principles of the Socialist Party

● for calls to overthrow the regime or actions that discredit the party.

In Russia, back in 2021, the national standard of the Russian Federation for trustworthy artificial intelligence was adopted (GOST R 59276-2020) [5]: "Artificial intelligence systems: ways to ensure trust," which provides a classification of factors affecting the quality and ability of AI systems to inspire trust at life cycle stages. And one of the main points of this document is the explainability of artificial intelligence. This standard applies only to AI systems that provide solutions only to specific practical problems and cannot be used for "strong" or "general" AI systems.

Considering such a huge interest in the problem of trust and explainability of AI, the authors of this work present to the public their original concept of constructing Trustworthy Artificial Intelligence (TwAI), which is based on a task approach to artificial intelligence [7-12], which to a certain extent generalizes such well-known and popular approaches, like agent-based approach and artificial general intelligence (AGI).

Note that the success and popularity of the agent approach, set out in the monograph [13], owes much to a well-chosen system of concepts, including various types of "agents" and "external environment". Various problems of artificial intelligence are considered in this approach as tasks of interaction between the "agent" and the "environment", which allowed its authors to carry out a detailed classification of agents, environments and, what is fundamentally important, the *tasks* they solve. Thus, the agent approach actually follows the task approach, but without explicitly defining the tasks solved by agents and indicating all the components inherent in the concept of "task".

Artificial General Intelligence (AGI), reviewed in [14], showed that modeling human cognitive processes is not a prerequisite for solving intellectual tasks.

Therefore, AGI reflects the fact that artificial intelligence, to one degree or another, can be possessed by both a person or a living organism with a highly developed central nervous system, and an abstract robotic system. Leading developers of AGI (Ben Goertzel, Shane Legge, Pei Wang) give the following definition of AGI: "it is the ability to solve cognitive problems in general, acting purposefully, adapting to environmental conditions through learning, minimizing risks and optimizing losses to achieve one's goals" [14] .

The purpose of this work is to show that the task approach to AI, generalizing and developing the approaches mentioned above, and at the same time being anthropomorphic in nature, since, based on the Theory of Functional Systems (TST) of brain function [15-20], allows adequate modeling of cognitive purposeful human activity, can become the conceptual basis of Trusted AI and the corresponding tools.

2. *Task approach and its mathematical theory*

2.1. Task approach

The first attempts to give the concept of "task" a precise meaning were made by academician A.N. Kolmogorov back in the 30s of the last century, who used the concept of task to describe the mathematical semantics of intuitionistic propositional calculus [21]. The empirical Theory of Inventive Task Solving (TRIZ) also made a great contribution to the development of the task approach in relation to solving tasks related to the creation and development of technical systems [22]. However, the most substantiated, systematic and detailed description of the concept of the task approach was carried out by Academician Yu.L. Ershov and Doctor of Philology. K.F. Samokhvalov in relation to the problems of the foundations of mathematics [7-8].

2.1.1. *A task approach to the foundations of mathematics and AI*

The analysis of the concept of a task in relation to the foundations of mathematics begins with the following simple arguments formulated by K.F. Samokhvalov: "I'm thirsty" - what does that mean? There is, of course, no mistake in believing that the words "I'm thirsty" simply mean this, where this is a certain state of consciousness that I am experiencing now and which I call thirst. But then a new question arises: how is the feeling of thirst (wanting) related to the actual drinking (satisfaction of wanting)? How do I know that drinking can satisfy my thirst? Does the very experience of thirst contain the consciousness of how this thirst can be satisfied? … To know desire does not mean to know what is desired, but to be able to know what is desired," i.e. have a criterion for satisfying a desire.

Thus, the task is defined if and only if we have a criterion for solving the task - a criterion for checking whether the presented solution is really a solution to the task. In mathematical theories, such a criterion is considered to be the presence of a proof of the solution to the task. But this criterion is applicable only when, within the framework of the formal system itself, we have both a proof of the solution to the

task and the opportunity to verify by means of the system itself that this proof is indeed a solution to the task. In [7-8] it was proven that only in “weak” formal systems (for which Gödel’s theorem does not apply) can we use the formal system itself to determine whether some text is a proof of the solution to the task or not.

As a result, Hilbert's program for the justification of mathematics should be reformulated as follows: it is not necessary for all mathematics to prove its consistency, as was proclaimed in Hilbert's original program - this is impossible and unnecessary. It is necessary to formulate and solve tasks within the framework and means of appropriate weak formal systems.

2.1.2. ***The concept of “task” in artificial intelligence***

Let us now define the concept of “task”, aiming to consider the widest possible range of tasks, which is typical for such a direction as artificial intelligence. Let us first give an informal definition.

A task is defined if and only if its formulation contains:

- indication of the subject area, recorded in the form of a certain formal model, including a description of the signature and structure of the language (ontology), as well as knowledge about the subject area, including source data, facts and hypotheses;
- what request (question) formulated in the task for the subject area should we receive an answer to (solution to the task);
- criterion for satisfying a request - in which case it can be considered that an answer (solution) to a request (question) has been received;
- in what context should we look for an answer (solution) to a request (question) - what goal do we pursue when solving a task, i.e. what do we expect from the result obtained and what are its consequences and what to do if the answer turns out to be negative.

The proposed task approach assumes that the true purpose of artificial intelligence is to automate problem solving, assuming that tasks are formulated in terms of executable (declarative) specifications, discussed below. In addition, the concept of task we introduced, in contrast to AGI, as we will show below, adequately models the purposeful activity of humans and animals in accordance with TFS.

2.1.3. ***The concept of "task" in cognitive sciences***. A generalization of the task concept in cognitive sciences is the concept of a ***Goal*** [15-16,20]. The goal cannot be achieved without having a criterion for its achievement, otherwise it can always be assumed that it has already been achieved. Therefore, the formulation of the Goal should always include a criterion for the goal achievement. Achieving a Goal gives a certain ***Result***.

The only physiological theory in which the Goal achievement and obtaining a Result is considered as the brain's solution to the problem of satisfying a certain need is the ***Theory of Functional Systems*** by P.K. Anokhin [15-20]. This theory also reveals the physiological mechanisms of achieving the goal and solving this problem

by the brain. Anokhin wrote: "Perhaps one of the most dramatic moments in the history of studying the brain as an integrative education is the fixation of attention on the action itself, and not on its results ... we can assume that the result of the "grasping reflex" will not be grasping itself as an action, but that set of afferent stimuli that corresponds to the signs of the "grasped" object." The "totality of afferent stimuli" is the criterion for achieving the goal in the TFS. The goal in TFS is to satisfy some need: "Every need, even with a slight deviation of vital function from the optimal level for metabolism (which is what the need is), is immediately perceived by special receptor devices" (the criterion for achieving the goal).

According to P.K. Anokhin, the central mechanisms of functional systems providing purposeful behavioral acts have the same architecture.

2.1.3.1. ***Afferent synthesis***. The initial stage of a behavioral act of any degree of complexity is afferent synthesis, which includes the synthesis of motivational excitation, memory, situational and trigger afferentation.

Motivational excitement. Goal setting is carried out by the need that has arisen, which is transformed into motivational excitement.

Memory. Motivational excitement "extracts from memory" all possible ways to achieve the goal, as well as the entire sequence and hierarchy of results that must be obtained in order to achieve the goal in some specific way.

Situational afferentation. When the goal is achieved, the situation in which the result was obtained is also recorded. This situation is fixed as the necessary conditions along with the motivation required to achieve the result. Therefore, motivational excitation in this situation "extracts from memory" only those ways to achieve the goal that are possible in this situation. Thus, situational afferentation, when interacting with the experience extracted from memory, determines what and how can be done in this situation to achieve the goal.

Trigger afferentation. Trigger afferentation is also a situational afferentation, only related to the time and place of achieving the result. Trigger afferentation answers the question of where and when the result can be achieved.

2.1.3.2. ***Decision-making***. At the stage of afferent synthesis, by the motivational excitation can be extracted from memory (in a given setting) several ways to achieve the goal. At the decision–making stage, only one of these methods is selected - a specific action plan. By "pulling" all the accumulated experience from memory, motivational excitement is transformed into a ***specific goal*** that determines the way to achieve itself. A specific goal is called a "higher motivation" in the TFS.

2.1.3.3. ***The acceptor of the actions results***. Motivational excitement also "extracts from memory" the entire sequence and hierarchy of results that must be obtained for the implementation of the action plan. This sequence is called in the TFS the acceptor of the actions results. It is the dominant need (Goal) of the organism, transformed in the form of the anticipatory excitation of the brain, into a complex receptor of future reinforcement, which is the criterion for achieving a specific goal.

2.1.3.4. ***Reinforcement***. The sanctioning stage. If, as a result of the implementation of a specific action plan, the goal is achieved (the need is satisfied) and all the results of the action results acceptor are obtained, then the last sanctioned stage occurs in which the need is satisfied and the completed specific action plan is stored in memory.

2.1.3.5. ***Formal model of functional system*.** Based on the works [16-17,20,23] on the formalization of functional systems, a formal model was developed summarizing the basic properties of functional systems that:

1. uses a formal neuron model that detects causal relationships and is based on semantic probabilistic inference;

2. sets goals in purposeful behavior, forms a functional system and an acceptor of the results of an action, which continuously compare the results achieved with the results expected in the acceptor of the actions;

3. automatically generates sub-goals and reinforces the achievement of sub-goals if their achievement increases the likelihood of achieving the final goal;

4. selects an action in real time, taking into account the current situation and the afferentation received;

6. plans the achievement of the goal in accordance with the sequence and hierarchy of functional systems to achieve the final goal;

7. decides by decision-making on a certain way to achieve the goal.

The experiments conducted show the "naturalness" and effectiveness of the behavior of the received animates.

This model has been successfully used to model animates [20,23-25].

2.2. *Mathematical theory of task approach - semantic modeling*

Let us now move on to formal definitions. Let's define a language for task specifications based on the logical-mathematical theory of the task approach - semantic modeling. As the concept of the basic model of calculations, it is proposed to take the concept of ∑-definability of calculations [26], supplementing it with a procedure for checking the truth of ∑-formulas on a multi-sorted constructive model M, considered together with its list superstructure HW(M), and the concept of relative computability, allowing the use of oracles [27-32].

Let M be some set. Let us define by induction the set HW(M) of hereditarily finite lists over M:

$HW0(M) = \{\varnothing\}$;

$HWn+1(M) = HWn(M) \cup Lisp(HWn(M) \cup M))$;

$HW(M) = \cup\omega n=0\ HWn(M)$,

where Lispw(X) is the set of all finite lists consisting of elements of the set X.

Let $\mathcal{M}$ be a multi-sorted constructive model of signature σ, where M is the basis set of this model and let (HW(M)∪M) be the basis set of the extended model HW($\mathcal{M}$) of signature $\sigma^* = \sigma \cup \{nil, head, tail, cons, conc , =, \epsilon, \leq, U\}$, where the new

functional symbols have a natural interpretation of the list-operations, and the predicate symbols are interpreted as list-relations, where = is the equality relation of lists, ϵ is the relation "to be an element of a list," $\leq$ is the relation "to be the beginning list" and where U(HW(M)) = M. We will call the model HW($\mathcal{M}$) of extended signature σ^* a superstructure of hereditarily finite lists for the model $\mathcal{M}$ or simply a list superstructure over $\mathcal{M}$. Next, we will consider the multi-sorted constructive model $\mathcal{M}$ together with its list superstructure HW($\mathcal{M}$) as a certain initial basic calculator, in which all signature entities are computable constructions. In addition, we also allow the possibility of enriching the signature σ^* with additional predicate and functional symbols, some of which will be used as predicate and functional variables, and for the rest, certain external computable constructions, which we will further call oracles, act as their interpretations.

Terms and atomic formulas of signature σ^* are defined in a natural way in our language. The class of $\Delta 0$-formulas of signature σ^* is defined as the smallest class of formulas containing all atomic formulas and closed under $\neg$, $\wedge$,$\vee$, $\rightarrow$, $\forall x \in a$, $\exists x \in a$, $\forall x \leq a$, $\exists x \leq a$, where a is a term. Recall that here "$\in$" means "to be an element of the list", and "$\leq$" means "to be the initial sublist". We will further call expressions of the form $\forall x \in a$, $\exists x \in a$, $\forall x \leq a$, $\exists x \leq a$ restricted quantifiers.

Having the class of $\Delta 0$-formulas, we now define the class of $\sum$-formulas of signature σ^* as the smallest class of formulas containing all $\Delta 0$-formulas and closed under $\wedge$, $\vee$, $\forall x \in a$, $\exists x \in a$, $\forall x \leq a$, $\exists x \leq a$ and the unbounded existence quantifier $\exists x$. Note that in our language we can quite correctly write formulas that have added limited quantifiers of the form "$\forall x=t$" and "$\exists x=t$", where t is a term. It is easy to see that such expressions are logically equivalent in meaning to the assignment operator "x:=t" in traditional programming.

We will call a certain relation in HW($\mathcal{M}$) a $\sum$-relation if it is $\sum$-definable, and a Δ_0-relation if it is Δ_0-definable. A partially defined function is a $\sum$-function (Δ_0-function) in HW($\mathcal{M}$) if its graph is $\sum$-definable (Δ_0-definable).

Let us now introduce the concept of formula definability, which is very important for further presentation. Let some model $\Re$ = (R; $P^{n_0}_{m_0}, \ldots, P^{n_k}_{m_k}$) be given. We will say that a model $\Re$ is Σ-definable over HW($\mathcal{M}$) if there exists a Σ-definable subset S $\subseteq$ HW($\mathcal{M}$) and a map μ : S $\rightarrow$ R such that μ-prototypes of the images of the predicates $P^{n_0}_{m_0}, \ldots, P^{n_k}_{m_k}$ and =, as well as μ-prototypes of the images their complements are Σ–relations in HW($\mathcal{M}$) (usually with parameters). This definition can be correctly generalized to the case of models whose signature, in addition to predicate symbols, also includes functional symbols. The importance of this definition lies in the fact that it allows us to formally define a model of a certain subject area as a set of $\sum$-definitions of our language, i.e. as a set of $\sum$-formulas and $\sum$-terms. In this case, recursive schemes of $\sum$-definitions are allowed with some restrictions on the occurrence of defined predicates and terms in them. We will also define a query to a domain model as a $\sum$-formula, the recording of which can use

both signature constructions of the basic constructive model, as well as defined predicates and domain terms.

Let us present some facts that characterize the exceptional expressive and computational power of the concepts and constructions we introduced [28-31,33-41]:

● If the original model $\mathcal{M}$ is computable (polynomial computable), then its list superstructure HW($\mathcal{M}$) is also computable (polynomial computable).

● If the model $\mathcal{M}$ is polynomial computable, then in its list superstructure HW($\mathcal{M}$) the functions defined by the terms are also polynomial computable.

● If the original model $\mathcal{M}$ is computable, then in its list superstructure HW($\mathcal{M}$) $\sum$-relations are recursively enumerable, and $\sum$-functions are computable. This fact explains the content of the term "formula definability of computability".

● If model $\mathcal{M}$ is polynomial computable, then in its list superstructure HW($\mathcal{M}$) the truth of each Δ_0-formula $\varphi(x_1, \ldots, x_n)$ is polynomial computable.

● For a decidable model $\mathcal{M}$ and its computable list superstructure HW($\mathcal{M}$), their theories are unfortunately undecidable.

● In the HW($\mathcal{M}$) model, for n-ary $\sum$–relations there is a universal (n+1)-ary $\sum$–relation $P^{(n+1)}(x; x_1,\ldots, x_n)$ such that for each $\sum$–relation $Q^{(n)}(x_1,\ldots,x_n)$ there is an element a such that $Q(x_1,\ldots,x_n) \leftrightarrow P(\mathbf{a}; x_1,\ldots,x_n)$. At the same time, there are no universal computable functions in the HW($\mathcal{M}$) model.

● Let $\varphi(x_1 , \ldots, x_n; Q+)$ be a $\sum$-formula with a positive occurrence of the predicate $Q(x_1,\ldots,x_n)$ and let the predicate Q be an n-ary relation in the list suspension HW($\mathcal{M}$). Let us define the operator G on HW($\mathcal{M}$):

$G(Q) = \{(a_1, \ldots, a_n)|\ HW(\mathcal{M}) \models \varphi(a_1, \ldots,a_n; Q+)\}$.

then the operator G is monotone and has a smallest fixed point that is $\sum$-definable. Let us note that it is this extremely important, the Gandy fixed point theorem that allows the use of recursive constructions in the formal definition of subject areas. At the same time, it is often advisable to look at the predicate symbol Q+, which is positively included in the $\sum$-formula $\varphi(x_1 , \ldots, x_n; Q+)$, either as a recursively defined predicate, or as a call to an oracle, thereby allowing in our concept of semantic computability the so-called relative computability is an exceptionally powerful tool for modeling, particularly complex hierarchical structures.

In practice, it is proposed to limit ourselves to the class of Δ_0-formulas. Note that with the help of Δ_0-definitions, we can significantly increase the expressive power of our source language by introducing the so-called conditional and recursive Δ_0-terms [34-35]. So, for example, if φ is a Δ_0-formula, and t_1 and t_2 are terms, then the triple (φ,t_1,t_2) is a conditional Δ_0-term, the interpretation of which is the following computational scheme: if φ is true, then the term t_1 is executed , otherwise – term t_2. Using various iterative and recursive computation schemes, recursive terms are defined in a similar way [35]. The language of Δ_0-definitions, enriched with such constructions, appears as a very expressive logical-functional modeling language.

And what is very important is that such enrichment preserves the original polynomial computability. Moreover, it has been shown that under certain conditions the converse is also true - every polynomial computable function is Δ_0-definable [39]. To do this, we will use a more practically acceptable language for specifying problems in the class of Δ_0-formulas. We will say that problem P can be represented formulaically in HW($\mathcal{M}$) if:

- the subject area can be specified in the form of a Δ_0-model P of the signature of the list superstructure HW($\mathcal{M}$), i.e. as a set of Δ_0-definitions (recall once again that recursive Δ_0-definitions are allowed);

- a request to the Δ_0-model P is specified either in the form of a Δ_0-formula φ, or in the form of a Δ_0-term t.

- as a response to a request (i.e. a solution to problem P):

- if the query to the Δ_0-model P has the form of a Δ_0-formula φ, then specifications of free variables φ are sought for which the formula φ is true,

- if the request has the form Δ_0-term t, then the value of Δ_0-term t is calculated.

Here it is important to emphasize that the truth of the Δ_0-query formula obtained by substituting constants instead of variables is the criterion for solving Problem R. Note that there may be several sets of constants that make the Δ_0-query formula true and therefore it is possible to choose one that is, in the sense of the “best” solution, taking into account the context of solving the problem. In this case, the criterion for solving problem P must also contain a criterion for choosing the best answer to the query. This approach to formulating and solving problems is called semantic modeling. It is important to note that the formalism described above can be naturally extended by introducing probabilistic and fuzzy constructions.

2.3. *Inductive knowledge inference on the domain model*

It was defined in [41-46] that the subject domain is empirical system $\Im = \langle A, \Omega \rangle$, where A is the objects of the Subject Domain (SD), and Ω is ontology, represented by the set of relations and operations interpretable in the notion system of SD.

In [47] it noted that "Knowledge is perceived, realized and personally significant information". Therefore, for the inductive inference of knowledge, it is necessary for a human to understand and interpret the SD ontology.

Inductive inference of knowledge on SD that carried out by any machine learning method, must correctly to process objects properties and attributes in order to obtain knowledge in terms of ontology. It should be noted that the numerical values of the quantities themselves do not contain knowledge and information, because the meaning of the quantities is indicated in their interpretation, for example, 5 meters, 5 liters, 5 kilograms, etc. Interpretation, in particular, determines which mathematical operations can be meaningfully carried out with them. It is worth noting that physical quantities measured in a subject domain, other than physics, lose their original physical interpretation. Consider, for example, such a physical quantity

as temperature. Temperature scales in non–physical areas, for example, when measuring the patient's body temperature in medicine, soil temperature in agriculture or air temperature in the oven in cooking, etc., should be different, although they can be measured with the same device – a thermometer. A scale of some magnitude, from the point of view of measurement theory, is a set of relations and operations that it makes sense to perform with numerical values of quantities in a given subject domain, i.e. these are relations and operations that are interpreted in SD ontology.

It may be objected that a thermometer can measure nothing but temperature. He really measures physical temperature in all cases. But temperature, like any other device, is needed to obtain conclusions (knowledge) in the SD ontology to which it belongs. The patient's physical temperature is an indirect measurement of a medical quantity – the level of metabolism. For soils, temperature is interpreted in terms of plant physiology and microbial activity. Physical temperature in other SD is an indirect measurement of some not physical quantity interpreted in the SD ontology. What relations and operations on numerical temperature values that make sense for these quantities are already determined by the ontologies of the SD. Thus, in order to extract information from attributes and quantities of SD, it is necessary to define a set of mathematical relations and operations interpreted in the ontology Ω and include them in the ontology Ω of the subject domain $\Im = \langle A, \Omega \rangle$.

Let us consider the problem of the theory $Th(\Im)$ discovery. We will assume that the theory $Th(\Im)$ is a set of universal formulas (a more general case is considered in [47]). It is known that a set of universal formulas is logically equivalent to a set of rules of the form:

$$\forall x1,x2,\ldots(A1\&...\&Ak \Rightarrow A0),\ k \geq 0, \qquad (1)$$

where A0,A1,...,Ak are literals. Therefore, we can assume that the theory of $Th(\Im)$ is a set of rules of the form (1).

It is not difficult to see that the rule $C = (A1\&...\&Ak \Rightarrow A0)$ logically follows from any sub-rule of the form: $(Ai1\&...\&Ain \Rightarrow A0)$, where $\{Ai1,...,Ain\} \subseteq \{A1,...,Ak\}$, $0 \leq h < k$, that is: $(Ai1\&...\&Ain \Rightarrow A0) \vdash (A1\&...\&Ak \Rightarrow A0)$. Then the theory $Th(\Im)$ can be simplified. By the *law* of the empirical system $\Im = \langle A, \Omega \rangle$ we will call a true rule C of the form (1), for which each of its sub-rules is no longer true on $\Im$. Then it is true that $L \vdash Th(M)$ [16,48]. Then the theory of $Th(\Im)$ can be considered as a set of laws of an empirical system.

Let's define the probability η on the empirical system $\Im = \langle A, \Omega \rangle$ as on the model [49]. The probabilistic law on $\Im$ will be called the rule $(A1\&...\&Ak \Rightarrow A0)$ the conditional probability $\eta(A_0\&A_1\&...\&A_k)/\eta(A_1\&...\&A_k)$ of which is defined $(\eta(A_1\&...\&A_k) > 0)$ and strictly greater than the conditional probabilities of each of its sub-rules. The *Strongest Probabilistic Law* (SPL) will be called the probabilistic law C, which is not a sub-rule of any other probabilistic law.

Inductive inference of probabilistic knowledge on the SD $\Im = \langle A, \Omega \rangle$, is carried out by the following semantic probabilistic inference.

The Semantic Probabilistic Inference (SPI) of some strongest probabilistic law Cn predicting some fact G, we will call the sequence C1 ⊏ C2 ⊏.. ⊏ Cn, Ci = (Ai1&…&Aiki ⇒ G) of probabilistic laws, for which the Ci is a sub-rule of Ci+1 and $\eta(Ci) < \eta(Ci+1)$, i = 1,2,...n-1.

We need knowledge for predictions. Let us define SPL that solves the problem of statistical ambiguity [48,50] and predict without contradictions.

Let us consider the set of all SPLs that predict some fact G. This set can be viewed as semantic probabilistic inference tree of G.

By the *Most Specific Probabilistic Law* (MSWZ(G)) of the infeernce of some fact G we will call the SPL, belonging to some semantic probabilistic inference tree of G, having the maximum conditional probability value among all the rules of that tree. The set of all MSWZ(G) for all literals G ∈ Ω we denote as MSZ.

It was proved that L ⊂ MSZ [41,47-48] and therefore the set MSZ of laws generalizes the theory Th(ℑ). In addition MSZ is logically consistent [47-48,50] and therefore it is in exact sense a probabilistic theory of the domain ℑ = ⟨A, Ω⟩. It can be proved [47-48,50] that predictions obtained by the inductive statistical inference according to the laws from the MSZ are consistent.

This theory of the inductive inference of knowledge on the SD ℑ = ⟨A, Ω⟩ is implemented as a platform and software system "Discovery", described below. By this platform a lot of practical tasks were solved (see http://old.math.nsc.ru/AP/ScientificDiscovery/index_rus.html).

3. *Platform solutions for task approach*

Currently, semantic modeling, as one of the concepts for automatically solving intellectual problems, is based not only on the methodology and theory of the task approach, but also has at its disposal a developed toolkit aimed at supporting and maintaining the following technological scheme for solving intellectual problems:

- STEP 1. The emerging need is clarified, the associated contradiction associated with the lack of a “template” way to satisfy this need is identified and studied. Let us note that this contradiction, in its essence, is the true cause and content of the problem being solved. Next, a criterion for successfully overcoming the identified contradiction is formulated in natural language (a prototype criterion for solving a problem), if necessary, a decomposition of the contradiction/task is carried out and the context of the problem is determined (why, why, the nearest supertask, subtasks, purpose and consequences of solving/not solving the problem, etc. .).
- STEP 2. The general knowledge relevant to the problem is described in natural language, an ontological model of the problem domain is constructed (concepts, facts, rules, relationships, ...), and a class of queries to the problem domain is formulated in general and ontological terms.
- STEP 3. In logical-probabilistic terms of semantic modeling, a formal model

of the problem domain of the problem is constructed and the query and solution criterion are formulated in the same formal terms.

● STEP 4. Within the framework and means of the corresponding semantic modeling tool platform, a computer model of the problem is built.

● STEP 5. Using a computer model, answers to queries are found. The feasibility of the decision criterion, the validity and/or explainability of the answers and their interpretability are checked.

As for the technological tools for semantic modeling, several platform solutions have been created and are actively being developed.

3.1. *Platform D0SL*

An example of the successful application of the task approach and its logical-mathematical theory – semantic modeling is the semantic platform **D0SL**, which represents the next generation of the so-called Business Rules Engines (BRE) [www.d0sl.org]. D0SL, just like the usual BRE platform, allows you to implement the business logic of the system using the language of declarative logical specifications. The project of creating the D0SL platform was carried out under the leadership of V.S.Gumirov.

The d0sl platform allows you to control the logic of the behavior of complex systems using the d0sl language, understandable to a specialist in the subject area. The platform has a wide range of applications – from enterprise business processes to project management or the behavior of autonomous systems, including artificial intelligence systems and the Internet of Things. Note that the d0sl language is easily extended by functions and objects specific to a particular subject area. Moreover, d0sl itself is a domain-specific language (DSL). The existence of an extension mechanism at the d0sl language level allows you to build a hierarchy of DSL languages that can be converted to d0sl and, therefore, can be executed by the d0sl kernel.

As for applications, the d0sl language is actively used in processing transactions and system events in telecommunications, fintech and banking, which makes it easy to change and configure the business logic of the applications being created. The use of the d0sl platform has proved particularly effective in the digital transformation of enterprises, allowing for the rapid and high-quality development of autonomous decision support systems, predictive analytics systems, modeling and digital counterparts of various objects, including the organizations themselves.

It should be noted that the d0sl language is one of the main technological tools of the Eyeline Mobilizer open platform, designed for the development and operation of services with support for mobile devices on all types of platforms, including USSD/SMS/SS7, smartphones (Android, iOS), mobile Internet [www.eyeline.ru]. Using the Eyeline Mobilizer platform, solutions have been developed and projects have been implemented in the field of telecommunications, mobile marketing and advertising, finance, banking services, municipal management and mobile payments.

Currently, solutions and services based on the Eyeline Mobilizer platform serve tens of millions of mobile device users. The platform allows you to significantly reduce the cost of developing and operating mobile solutions for your business. Examples of successful implementations are: USSD services for MTS (*100#, *111#, etc.) and the USSD portal Alfa-Click *142# Alfa-Bank (Russia), InnomaNet/T-mobile (Russia), WATAGO Africa (all the largest mobile payment systems in Nigeria), the project "Parking" of the Government of Moscow (Russia), missed call notification system, Yellow Pages/SMART in the Philippines, cloud installation for customer service GlobalUSSD.com , Beeline Russia (direct money). transfers) and much more.

3.2. *The bSystem Platform*

bSystem is a platform for building digital twins of organizations and processes. It has been developed by the research group of A.V. Mantsivoda for several years. The platform is focused on creating intelligent management systems for large business ecosystems, digital transformation of enterprises and other integrated solutions. Thanks to a unified semantic modeling environment, bSystem integrates the basic level of ontologies with the inductive logical-probabilistic knowledge inference and the semantic probabilistic inference of predictive rules/decisions suggested by E.E. Vityaev, on the one hand, and with classical programming, on the other.

So, the architecture of bSystem is based on the integration of three levels
logical-probabilistic
analytical/ontological
operational/transactional.

The ontological level [51-52] controls the main parameters of the simulated organization, reflects the current state of its components and connections between them, provides analytical tools for making decisions at the operational level (e.g., business intelligence tasks). This level forms the core of the platform; here ontologies are managed – the basic domain knowledge systems around which all activities are built.

The level of logical-probabilistic inference (LPI) is built over this core level. It implements the functionality of explainable artificial intelligence. Within the platform, LPI is responsible for strategic issues of forecasting, decision-making and other AI tasks. Its value for bSystem lies in the fact that, unlike neural networks, which (within the learning process) “accumulate” numerical coefficients in neurons that do not have a semantic interpretation, LPI – while learning – generates probabilistic logical theories (sets of patterns) that semantically describe the domain of discourse. These theories secure the ability of the LPI to explain its decisions. Potentially, this also provides “self-control”, that is, the ability to evaluate the advantages and disadvantages of generated knowledge with logical meta-tools. Such capabilities are unattainable for neural networks.

The lower operational-transactional level is the level of procedures and methods that change the logical environment over time. At this level, bSystem runs Libretto, an object-oriented language. From the task-approach viewpoint, the operational-transactional level forms the environment for the task-life-cycle management. Any business process controlled at this level is nothing but the process of solving a task in real time (in practice, an instance of some generalized task). Usually, a digital twin unfolds over time through a bunch of actively interacting tasks, the generation of new instances of tasks and the completion of tasks that satisfy the solution condition. And while the declarative description of a task and the criterion for its solution are controlled at a higher ontological level, the solution process itself is carried out at the operational level as a chain of transactions. At the same time, the business processes themselves are also represented as ontology objects, and this allows the platform to control the solution process at a declarative level. Thus, bSystem, through the interaction of the ontological and operational levels, provides a full implementation of the task approach.

The key feature of bSystem is that all its three levels are embedded in a single semantic space, they interact with each other and provide vertical scaling. This integration brings a powerful synergistic effect, not only helping to improve quantitative indicators (due to seamless interaction where a zoo of various information technologies is usually used), but also allowing for a qualitative change. For example, such integration makes it possible to implement a fundamentally new low-code technology, so-called object low-code, based on automatic code synthesis from ontology data [53]. This fundamentally distinguishes it from traditional procedural low-code, based on diagrams - variations of the BPMN specification [54].

Ontologies are the core tool that ensures the integrity of the bSystem digital environment. Actually, a digital twin is an ontology or a network of ontologies that plays the role of the semantic modeling basis, which defines the described domain, as well as the tasks to be solved.

An ontology also can be considered as a hub that ensures the interaction of various components of the digital twin. At the logical-probabilistic level, ontologies play the role of empirical systems, which contain *a priori* knowledge, and from which information for learning is taken. At the core – analytical-ontological level – ontology is the semantic basis for the digital twin, a source of information for analytical manipulations. Finally, at the procedural level, the ontology serves as a data type system for the platform programming language - Libretto.

Finding a suitable architecture for ontologies appeared to be a hard problem. Suffice it to say that bSystem was the sixth (and only the first quite efficient) attempt to build an ontology management platform of this kind. To understand better what kind of ontologies we really need, we had to develop a system of requirements that an ontology must satisfy. Below we list some of them.

First, as noted above, ontologies are responsible for ensuring vertical

connectivity and a unified semantic space of the platform. Knowledge from the ontology must have an efficient interpretation at all levels of bSystem: in sophisticated logical/probabilistic inference, in analytical modules (e.g., in business intelligence), as well as in software that solves traditional algorithmic problems.

Secondly, users should be able to work with ontologies even if they lack an understanding of mathematical logic (difficulties in understanding were one of the key barriers responsible for the failures of the well-known Semantic Web concept [55]). Since an ontology provides the basic knowledge about a domain during the interaction between people and the platform, if people do not understand the ontology, everything else becomes meaningless. Therefore, constructing an ontology that provides an intuitive understanding, which is consistent with the user's everyday experience, is also a must.

Further, the ontology must provide a standardized mechanism for interaction with external ‘oracles’ (information systems, neural networks, the Internet of things). For example, collecting data from controllers installed in a residential complex is inevitable for the proper functioning of its digital twin. For interacting with the outer world, an ontology must be equipped with logical analogues of well-known program interfaces (APIs).

Finally, if we want our ontologies to be practically useful, they must be efficient. This means that high-level ontologies must have implementations comparable in productivity to relational databases. An ontology must be able to manage millions of entities and provide answers to queries of practical complexity in real time.

Combining all these requirements within a single formalism turned out to be quite a difficult task (for example, ontologies based on the well-known OWL language [56] do not comply with several of them). While experimenting, we found that ontologies that are based on the ideas of object modeling are most suitable for these requirements.

The basic feature of object models is that they clearly have two layers - declarative and procedural. The first one consists of classes, objects and fields (properties) of objects. The second provides methods that are defined at the class level and acting on objects. Since ontologies are formal descriptions of the basic concepts, facts and relationships of a domain, and object models deal with similar tasks in programming, the declarative layer of object models can serve as an adequate background for an ontology. Moreover, the essence of object models is very intuitive, natural and familiar to people. This is proven by the widespread use of object-oriented languages like Java. Therefore, it is quite possible to take the declarative layer of object models and, by giving it strict logical semantics, turn it into a variant of ontology.

As for the procedural layer, object-oriented languages are usually Turing-complete and this makes the situation far less controllable. On the other hand, the presence of a procedural layer makes it possible to implement the idea of developing

an ontology over time and managing its life cycle. From this point of view, the application of each method or group of methods can be considered as a transaction that transfers the ontology from one state to another [57]. This capability is especially useful when solving management problems, since such an interpretation naturally defines business processes as transaction chains in the life cycle of ontologies [58].

So, to turn object models into ontologies we need to perform the following steps:

1. Define a logical formalism corresponding to the declarative layer of the object model. To do this, we use the mechanisms of Δ0-definability of semantic modeling.

2. Traditional object models are short-lived; they are usually stored in RAM. We need to make them persistent because ontologies are long-term repositories of data and knowledge. This provides them with object database functionality.

3. To work with an ontology, we need a query language that combines logical expressiveness with efficiency comparable to that of SQL in relational databases. In bSystem, the declarative core of Libretto, which has strict logical semantics, is used as such a query language.

4. To provide mechanisms for changing and developing the ontology over time, we reformat the procedural level of the object model into a transactional model.

5. Practice has shown that object-life-cycle management tools significantly enhance functionality compared to traditional object models, which is especially useful for business process management. By using transactional mechanisms, we get more control over the life cycle of objects.

6. The final step is to achieve ontology autonomy. An ontology is folded into a ‘cocoon’, isolated from the outside world, with which it now communicates only through a strictly regulated interface (analogous to a programmer’s API).

From the viewpoint of the task approach, step 5 implements two-level management of the task's life cycle, which is executed at the lower operational level and semantically controlled at the ontological level.

The final step implements the concept of local simplicity [52] as the construction of arbitrarily complex systems as networks of relatively simple ontologies with logically based interaction mechanisms. This allows for free horizontal scaling when building digital twins.

From an IT point of view, the second and last steps transform an object ontology into a microservice [59], since they provide two main properties of a microservice - the data storage autonomy (this is our ontology) and strict control over interaction channels with the outside world. This also determines the general architecture of digital twins, which are built on the top of bSystem.

bSystem provides us with a brand-new development methodology [60]. Due to its openness, modeling fundamentally changes the mechanisms of interaction between participants at the development and operation stages. The ontology is directly available for all participating competencies. For example, the customer

receives full control over the development process and can provide early feedback with immediate detection of problems. The high-level model, due to its strict logical semantics, simultaneously serves as a specification and executable documentation. This fundamentally differs from the black box of software code.

3.3. *DISCOVERY Platform - an ontological approach to inductive knowledge inference*

As it was mentioned, E.E. Vityaev developed an ontological [6,42,50-51] approach to knowledge discovery methods and the corresponding program system "Discovery". This approach is designed to discover knowledge from the information extracted from data and presented as an empirical system SD $\Im = \langle A, \Omega \rangle$. The Discovery system has the following advantages over other machine learning methods:

1) removes restrictions from the data types by using information extracted from the data in terms of ontology Ω;

2) removes restrictions from a priori knowledge by representing it in Ω;

3) removes restrictions on the tested hypotheses classes by specifying them by the knowledge classes (Rule Type) defined in Ω;

4) the Discovery system detects the following types of patterns:

• the set of laws L and the theory Th($\Im$) of the empirical system $\Im$;

• the set of probabilistic and strongest probabilistic laws;

• the set of maximally specific laws predicting without contradictions [48,52];

In the ontological approach, discovered knowledge is complete in two senses: (1) in the sense of completeness of extracted information from data by the use of ontology and measurement theory; (2) in the sense of completeness of discovered knowledge in the form of sets of rules a)-c);

Existing machine learning methods are not able to support data exploration mode when the detected pattern is unknown in advance. Each machine learning method reveals its own quite specific class of hypotheses corresponding to its ontology. The "Discovery" system can discover an arbitrary class of hypotheses that an expert can set, and thus is able to support the data research mode when the hypothesis class may change during the research process.

The Discovery system discovers knowledge in terms of SD ontology. Interpretability of patterns can be very important when making responsible decisions in areas such as medicine, finance or military applications.

The Discovery system was used to solve many problems in financial forecasting [6,42], bioinformatics [64], medicine [65], fraud analysis [66] and many others.

3.4. *Delta-platform*

As mentioned in the Introduction, one of the pressing problems of modern AI, in addition to the well-known ***"black box" problem***, is the so-called ***centralization***

problem, which means the emergence of a tendency to concentrate power and control over AI in the hands of a small number of legal entities. To solve this problem, in the Introduction it was proposed to use the experience of using decentralized solutions from the field of cryptocurrencies. At the same time, it was proposed to consider the AI algorithms themselves as smart contracts that are executed on the blockchain substructures of a multi-blockchain, which would ensure transparency of all operations, transparency of the training sample and transparency of decision-making. And finally, the requirement was put forward for polynomial computability of AI algorithms, which has obviously practical significance. In general, the task was to create a polynomial computable (in all senses) system of trust semantic modeling. But it took several years of serious research to begin to solve it practically.

Conditional terms introduced in 2017 by S.S. Goncharov [34] made it possible to understand that terms can be not only standard terms of first-order predicate logic, but also other, correctly defined syntactic constructions. This allowed S.S. Goncharov and D.I. Sviridenko in 2019 to introduce such termal extensions as bounded recursive terms, B-while terms, iterative terms, etc. [33] and show that all these termal extensions do not go beyond polynomiality. But the problem of p-completeness of these extensions was never solved by them. However, already in 2022, S.S. Goncharov and A.V. Nechesov found the final syntactic construction, which completed the construction of the p-complete semantic modeling language L [39]. A little earlier, in 2021, they also proved a polynomial analogue of Gandy fixed point theorem [36], which said that within the framework of a certain Δ_0-system it is possible to find such conditions on formulas that the classical result of Gandy fixed point theorem becomes valid and for this system. Moreover, which is very important, in this system the smallest fixed point of the constructed operator will be p-computable. This gave an even greater impetus to the further development of the semantic programming language L and to the development of the formal Δ_0-system.

Note that the polynomial analogue of Gandy fixed point theorem allows us to immerse any inductively specified syntactic constructions inside a formal system, offering both encoding and means of assessing the complexity of such storage. This allowed S.S. Goncharov, D.I. Sviridenko and A.V. Nechesov to describe an object-oriented version of the p-complete semantic programming language L* [40] and begin to develop a special framework - the Delta platform, which incorporated, including all the necessary functionality for solving the above mentioned ideas: multi-blockchains, smart contracts and the language L* based on the p-complete logical language L. At the same time, at different levels of multi-blockchains, if necessary, you can set different levels of user access, set the levels of transparency of both the blockchain itself and the smart contracts executed on them. Due to the fact that smart contracts are programs of a special type in the p-complete language L*, this significantly reduces the level of labor costs when auditing them. In the language L* itself, it is possible to describe the operation of both neural networks [61] and

blockchains [62], which is undoubtedly an advantage, since it does not require any external software for their implementation.

From the point of view of the problem approach in the p-complete language L*, we can not only describe the problem itself, but also solve it within the framework and means of the same language. Due to the fact that our base is blockchain, decentralized solutions are built quite simply. The most secure and decentralized one, for example, the Bitcoin or Ethereum blockchain, can be chosen as the top-level blockchain. Blockchains at lower levels may not have decentralization and can be used as a regular database, including on the home computer of one of the users. And since there will be no need to reach any consensus with other blockchains, this will entail a high write speed.

We would like to note that the creation of the Delta platform in integration with predictive systems such as Discovery can give a powerful impetus to the construction of AI algorithms, which by their nature are close to artificial general intelligence (AGI) algorithms, which ensures both high self-learning of such systems, and the high level of trust that ordinary users have in them. In this combination of the above-mentioned possibilities, we see the implementation of the main provisions of the task approach, when AI systems can solve both the tasks assigned to them and independently formulate new tasks that were not explicitly stated in the AI algorithms. All this allows us to talk about the possibility of building a trusted "general" AI within the framework of a task approach.

4. *Conclution*

This article can be considered as a kind of report on the results of research in the field of the task approach and its practical applications carried out by the team of authors of the article, their students and colleagues, which were started back in the 70-80-ies of the last century. Over the years, a lot of work has been done on the development and practical application of the provisions of the task approach in relation not only to various problems of the foundations of mathematics and AI, but also to such areas as cognitive sciences, neurophysiology, medicine, digital transformation of enterprises, automation of the design of complex systems, digital twins, etc. As can be seen from the above, it was possible not only to significantly develop the methodological and mathematical provisions of the task approach, but also to create its effective instrumental and technological base, which made it possible to successfully solve a wide range of tasks from various fields: telecommunications, business, retail, fintech, genetics, geology, cybersecurity, robotics, medicine, etc. On the agenda is the creation of a trust–based "common" AI. And as the experience of previous years shows, there is every reason that such an AI will be created in the near future. Including within the framework and by means of the task approach.

摘要：本文介绍基于任务方法的可解释、可信人工智能的基本概念，目的是阐述人工智能的任务方法可以概括和开发其它著名的诸如基于智能体智能和通用人工智能方法。此外，不同于通用人工智能(AGI)，任务方法是拟人化的方法，因为它将脑功能的生理功能系统理论形式化[15-20]，用于描述目的性人类活动。

为了在实践中有效实施任务方法，本文介绍几种可执行的任务描述语言，通过这些语言对任务的描述和解释，可以生成解决问题的可执行代码。任务的描述与特定的应用领域有关，完成这些任务需要先产生面向应用领域的查询，再执行查询搜索答案，最后得到问题的解。这些操作在本项目研发的一个语义建模框架内完成。

本文介绍用不同任务描述语言开发的解决应用问题的几个平台和解决问题的实例，并给出本项工作的结论。

关键词：人工智能，任务方法，可信问题

1. 绪论

ChatGPT 革命关联的事件开辟了人工智能的一个新世界，不仅充满了多种可能性和新的辉煌前景，也给人工智能应用带来新的风险和危害，产生不可控的结果。人工智能的题目已经变得极其时髦和流行，不仅在专家和专业人员之间热烈地讨论，也在包括政府部门在内的很多圈子中热议。人们讨论人工智能飞速发展的一个中心问题是人工智能系统的可信问题。这一问题实际反应了社会对人工智能现状的认知，即：当代人工智能系统不能保证它给出的问题解的正确性。人们已经认识到人工智能系统的决策可能不真实、不对、不正确、不精确、故意误导等。

人工智能可信的问题是一个复杂问题，众多问题的纠结是人们是否相信人工智能算法。本文只触及其中的一些问题，也就是黑箱问题、中心化问题和审计问题。根据作者的理解，这些问题的解决可以显著提高人们对人工智能系统的信心。

让我们回顾一下，黑箱问题指的是某些人工智能系统对某一问题生成的解不能给出任何解释。更严重的是，终端用户对这种人工智能系统产生的结果通常在任何条件下都不能进行确认，或者对结果的确认非常复杂，需要大量的资源和人力成本。要完全解决所关心的问题，只让人工智能系统产生结果的机理透明是不够的。同时，还需要提高这些系统的用户能力。毕竟，人工智能系统服务的高需求量是用户支撑的，而这些用户不关心黑箱问题。此外，即使同一神经网络的算法是透明的，形成的结果也需要专门程序给予解释，给出的正当理由也不足以使普通的不熟练用户相信和理解。这种现象在具有百亿参数、

大量权重系数和激活函数的人工神经网络特别突出。在这种情况下，似乎有必要使用逻辑方法和逻辑规则检查结果，不仅使结果可信，也可以给用户解释生成结果是怎样考虑的。

现代人工智能的另一个重要问题是所谓的中心化问题，指的是当出现紧急状况时，人工智能系统的控制权集中在少数法人手中。这一状况对人工智能领域的发展和创新有极大地负面影响。此外，集中控制的系统，如 ChatGPT，可能产生错误的系统设置，进而引起严重的后果，伤害系统最终用户和社会总体。解决这一问题，建议采用非中心化解决方案的优点。一个案例是加密货币采用的区块链技术。当今世界最稳定的计算机网络是比特币区块链网络，从 2009 年 1 月 3 日启动以来一直稳定运行。原理上讲，如数据库适合存储信息相同，区块链可能更适合人工智能解决方案。同时，可以采用多区块链来避免区块链著名的三重困境[1]。让我们回忆一下，最早的多区块链概念是 2017 年由 Nikolai 和 Pavel Durov 兄弟在他们的 TON 项目白皮书中提出[2]。事实证明，采用不同层次的多区块链，可以实现可扩展性、安全性和去中心化多种必要的指标。人工智能算法本身可以作为智能合约在多区块链的区块链子架构上运行以确保所有操作、训练数据集和决策的安全、正确和透明。

除了黑箱和中心化问题外，当代人工智能的另一重要问题是审计问题，逐渐变得越来越相关。审计问题是：尽管可以访问人工智能算法，我们不能核查一个基于大数据训练的人工智能系统是怎样做决策的。很清楚，对于这种软件解决方案的能力分析需要专业能力强的专家。通常，这样的专家奇缺，必须专业培训。

由于上述问题，人工智能系统做出的决策存在一种风险--用户无法理解，或者用户无法判断。由于这个原因，出现了可解释人工智能（XAI），要求人工智能系统产生的结果透明并可解释。当前，许多国家正在试图把这一要求立法。例如，2021 年 9 月，美国国家标准与技术研究所批准了可解释人工智能的 4 项原则[3]：

- 可解释性：人工智能系统需要提供有证据支持的解释。
- 意思明确：提供的解释必须意思明确，可以被人工智能系统的用户理解。
- 解释精准：提供的解释必须精准地阐述人工智能系统的决策过程。
- 界限清晰：人工智能系统必须在它的设计范围内工作。

这些原则已经被中国政府考虑，放入 2023 年 4 月发布的可解释人工智能北京路线图。中国已经对所有人工智能公司开发人工智能施加了限制。开发公司必须对他们开发的人工智能应用承担如下责任（包括开发聊天机器人）：

- 非法获取原数据
- 聊天机器人谈的所有话题内容

- 内容违反社会的基本道德准则
- 号召颠覆政府或污蔑党的行动

俄罗斯在2021年发布了俄罗斯联邦可信人工智能国家标准(GOST R 59276-2020) [5]。标准限定：“人工智能系统要有确保可信的方法”。标准对影响人工智能系统的能力和品质的因素做了分类，鼓励在生命周期不同阶段保证可信。这一文件的重要一点是人工智能的可解释性。这一标准只限于对具体的应用问题提供解决方案的人工智能系统，不能用于强人工智能系统或通用人工智能系统。

考虑到对于人工智能可信和可解释问题的极大兴趣，本文作者在此公开阐述他们对构建可信人工智能(TwAI)的最初概念，这些概念基于人工智能任务方法[7-12]，在一定程度上，这一方法是对其它著名流行方法的概括，如：基于智能体方法和通用人工智能（AGI）。

注意。专著[13]提出的智能体方法的成功和流行，很大程度上归功于系统概念的选定，包括各种类型的智能体和外部环境。在这一方法中，人工智能的各种问题被认定为智能体与环境的交互任务，允许作者对智能体、环境和要完成的任务做详细地分类。其中，对任务的分类尤其重要。因此，智能体方法实际上是遵循了任务方法，但是没有明确定义智能体要完成的任务，也没有指定内涵任务概念的所有组成部分。

文献[14]综述的通用人工智能指出，对人类认知过程建模不是解决智力任务的前提。因此，通用人工智能只反映这样的事实，在某种程度上，人类或具有高度发展的中央神经系统的其它生命体，以及抽象的机器人系统，都能够具备人工智能。通用人工智能领域的领军开发者（Ben Goertzel, Shane Legge, Pei Wang）对通用人工智能给出如下定义：一般来讲，通用人工智能具有解决认知问题的能力，能够有目的地响应，通过学习自适应环境条件，能够使风险最小化和损失最优化，并达到最终目的[14]。

本文的目的是介绍人工智能的任务方法，它可以用于概括和开发前面提出的其它方法，同时具有拟人化的特质。基于脑功能的功能系统理论[15-20]，人工智能的任务方法可以对人类目的性的认知活动建立充足的模型。因此，任务方法可以作为可信人工智能和相应工具的概念基础。

2. 任务方法及其数学理论

c. 任务方法

最早给出任务精准概念的人是A.N. Kolmogorov院士。在上个世纪30年代，Kolmogorov院士就用任务概念来阐述直觉命题演算的数学语义 [21]。伴随着求解任务技术系统的产生和发展[22]，创意性任务求解的经验理论对任务方法的发展贡献极大。但是，对任务方法给出最具体、系统和详细描述的人是

Yu.L. Ershov 院士和文献学博士 K.F. Samokhvalov，他们在研究数学基本原理的问题时用了任务方法概念[7-8]。

2.1.1 数学基本原理和人工智能中的任务方法

数学基本原理中任务概念的分析起始于 K.F. Samokhvalov 对下面简单论点的形式化定义：“我口渴了” 这句话是什么意思？当然，相信“我口渴了”这句话的确切意思是没有错误的，这是我们正在经历的我们叫做“口渴”的一种确定的共识状态。不过，一个新的问题出现了：对比你口渴了（有想喝水的愿望）和你喝了水（愿望满足），口渴的感觉是怎样的？怎么知道喝水可以满足口渴的愿望？口渴的经历包括怎样满足口渴时愿望的共识吗？…。知道有愿望不等于知道具体愿望是什么，但是有能力知道具体愿望是什么，也就是说，具有满足愿望的准则。

因此，只有有了求解任务的准则，用这一准则检查给出的解是否真的是任务的解，才能定义（理解）任务。在数学理论中，这样的准则等同于任务解的证明。但是，在形式系统自身的框架内，这一准则只能用在如下情况，我们具有任务解的证明，并有机会通过系统本身确认，解的证明确实是任务的解。文献[7-8]证明了只有在“弱”形式系统（Gödel 定理不成立）可以用形式系统本身确定一些文本描述是否是任务解的证明。

因此，数学的正当性的希尔伯特计划可以形式化表示如下：没必要证明所有数学都是一致的，正如希尔伯特最初计划中声称，这是不可能的，也是没必要的。有必要在恰当的弱形式系统框架和工具内对任务作形式化并求解。

2.1.2. 人工智能中的任务概念

让我们定义任务概念，目标是在最广范围内考虑各种任务，这也是人工智能方向的特有方法。先给一个非正式定义。

定义一个任务必须包含如下形式化内容：

- 标示主题领域，记录在特定的形式模型中，包括语言（本体）的结构和签名的描述，以及主题领域知识，如：原始数据、事实和假设；
- 主题领域任务中形式化的什么要求（问题）应该得到任务解的答案；
- 满足要求的准则：可以认为，准则满足后，表示得到了要求（问题）的答案（解）；
- 在何种条件下寻找要求（问题）的答案（解）；
- 求解任务解的目标：从结果中期望得到什么，后果是什么，对负面答案采取何种措施。

本文建议的任务方法假设人工智能的真实目的是使问题求解自动化，假定采用可执行的（陈述性的）描述对任务形式化，具体描述下面讨论。除此之外介绍的任务概念比照通用人工智能，正如我们将在后续展示的，根据功能系统理论，任务概念对人类和动物的目的性活动恰当地建模。

2.1.3. 认知科学中的任务概念

认知科学中把任务概念概括成目标概念。没有成功准则，目标是不能实现的。否则，总可以假设目标已经实现了。所以，目标的形式化总要包括目标成功的准则。实现一个目标要给出一个确定结果。

唯一的生理学理论是 P.K. Anokhin 的功能系统理论。这一理论认为，目标实现和得到结果是大脑对满足某种需要的问题求解 [15-20]。这一理论同时揭示由大脑实现目标和解决问题的生理学机理。Anokhin 写道：“作为一体化教育来研究大脑的历史过程中，最引人注目的一个时刻是注意力固定在动作的本身，而不是结果…我们可以假设抓住反射信号的结果不是抓住这个动作本身，而是一组响应被抓对象踪迹的传入刺激”。传入刺激的全部就是功能系统理论中的实现目标的准则。功能系统理论中的目标是满足某种需求：“每个需求，即使是新陈代谢最优水平的生活机能微小变化，专用的接收装置也马上收到”（实现目标准则）。

P.K. Anokhin 指出，提供有目的行为动作的功能系统中心机理具有相同结构。

2.1.3.1. 传入合成

任何复杂度的行为动作的起始阶段是传入合成，包括动机兴奋、记忆、情景和触发传入。

动机兴奋。根据出现的需求设定目标，然后转换成动机兴奋。

记忆。动机兴奋从记忆中提取所有可能实现目标的方式，以及用某种具体方式实现目标必须获得结果的整个序列和层次架构。

情景传入。当目标实现时，获得结果的情景也记录下来。这一情景固定下来，与动机一起作为实现目标需要的必要条件。因此，这一情景中的动机兴奋只从记忆中提取在这一情景中实现目标的可能方式。所以，当与从记忆中提取的经验交互时，情景传入决定在这一情景中实现目标能做的动作和怎样做。

触发传入。触发传入是唯一与实现目标的时空相关的情景传入。触发传入回答何时和在哪里获得结果的问题。

2.1.3.2. 决策

在传入合成阶段，动机兴奋可能从记忆中（在给定的设置中）提取多种方式实现目标。在决策阶段，只能选择一种方式——一个具体的行动计划。通过从记忆中提出累积的全部经验，动机兴奋转化成具体目标，决定实现目标本身的方式。一个具体目标在功能系统理论中称作更高动机。

2.1.3.3. 动作结果的接收器

动机兴奋也从记忆中提取实施行动计划必须获得的结果的整个序列和层次结构。在功能系统理论中，这一序列称作动作结果接收器，是生物有机体主导需求（目标），从大脑预期激励形式转换成复杂的进一步强化的接收器，作为

实现具体目标的准则。

2.1.3.4. 强化阶段

强化阶段即准许阶段。作为具体行动计划的实施结果，如果目标实现（需求满足），并获得动作结果接收器的全部结果后，最后的准许阶段开始。在这一阶段，需求满足了，完整的具体行动计划存储在记忆中。

2.1.3.5. 功能系统形式模型

基于文献[16-17,20,23]功能系统形式化的工作，开发了一个形式模型，总结了功能系统的基本特征：

1. 采用形式化神经模型检测因果关系，模型基于语义概率推理；
2. 采用目的性行为设置目标，形成一个功能系统和动作结果接收器，不间断地比较获得的结果和动作接收器中的期望结果。
3. 自动产生子目标，如果子目标的结果提高了实现最终目标的可能性，强化子目标的结果。
4. 实时选择动作，考虑当前情景和收到的传入。
5. 根据实现最终目标的功能系统序列和层次结构规划目标的实现步骤；
6. 通过决策确定实现目标的某种方式。

实验结果展示，获得的仿真体的行为自然而有效。这一模型被成功用于仿真体建模[20,23-25]。

d. 任务方法语义建模的数学理论

本节介绍形式化定义。我们可以基于任务方法的逻辑数学理论--语义建模，定义一种任务描述语言。作为基础的计算模型概念，语义建模采用 Σ-可定义性计算概念[26]，并补充了多重排序的构造式模型 M 中 Σ-函数真值的检查程序，以及模型的超结构 HW(M)和相对可计算性概念，允许使用一些黑箱操作[27-32]。

设 M 是某个集合。M 上的超结构集合 HW(M)中继承关系的有限列表用归纳形式定义如下：

1. $HW_0(M) = \{\varnothing\}$;
2. $HW_{n+1}(M) = HW_n(M) \cup Lisp(HW_n(M) \cup M))$;
3. $HW(M) = \cup^{\omega}_{n=0} HW_n(M)$。

这里，$Lisp_w(X)$是由集合 X 元素组成的全部有限列表的集合。

设$\mathcal{M}$是的多重排序构造模型 M 的签名 σ，M 是模型的基集，(HW(M)∪M)是扩展模型 HW($\mathcal{M}$)的基集，HW($\mathcal{M}$)的签名 σ* = σ ∪ {nil, head, tail, cons, conc , =, ϵ, ≤, U}。这里，新的函数符号自然解释为列表操作，谓词符号解释为列表关系，符号“=”表示列表的等价关系，ϵ 表示列表元素的从属关系，≤ 表示起始列表，U(HW(M)) = M。扩展签名 σ*的模型 HW($\mathcal{M}$)称作模型$\mathcal{M}$中有继承关系的有限列表的超结构，简单称为$\mathcal{M}$上的列表超结构。下一步，将多重排序构造模

型$\mathcal{M}$与它的列表超结构一起考虑，作为某种原始的基础计算器，其中，所有签名实体是可计算的结构构件。除此之外，允许通过增加谓词符号和函数符号丰富签名 σ*。一部分新的符号将用于表示谓词和函数变量，剩余部分表示某些外部可计算结构构件，称作黑箱操作，用来解释变量。

以自然方法用语言定义签名 σ*的项和原子函数。签名 σ*的Δ0-函数类定义为最小函数类，包括全部原子函数，在（¬, ∧,∨, →, ∀x∈a, ∃x∈a, ∀x≤a, ∃x≤a）操作下封闭。其中，a 表示一个项。请注意，“∈”表示“列表原素的所属关系”，“≤”指的是“起始子列表”。可以进一步称∀x∈a, ∃x∈a, ∀x≤a, ∃x≤a 表达方式为限制量。

有了Δ0-函数类，可以将签名 σ*的Σ-函数类定义为包含全部Δ0-函数，并在∧, ∨, ∀x∈a, ∃x∈a, ∀x≤a, ∃x ≤a 操作下封闭的最小类，存在无限制的约束函数∃x。注意，在我们的语言中可以正确地写出带有形式为“∀x=t”和 “∃x=t”的限制量。这里，t 表示项。可以很容易看出这样的表达式逻辑上等价于传统编程语言中的赋值操作“x:=t”。

如果 HW($\mathcal{M}$)的某个关系是Σ-可定义，称作Σ-关系，如果是Δ_0-可定义，称作Δ_0-关系。如果一个部分定义的函数的图是Σ-可定义（Δ_0-可定义），称作 HW($\mathcal{M}$)中的Σ-函数（Δ_0-函数）。

现在介绍函数可定性概念，这一概念在后续的表述中很重要。设某个给定模型$\mathfrak{R}$ = (R; $P^{n_0}_{m_0},\dots,P^{n_k}_{m_k}$)，如果下述条件满足，称模型$\mathfrak{R}$在 HW($\mathcal{M}$)上 Σ-可定义：如果存在一个 Σ-可定义的子集 S ⊆HW($\mathcal{M}$)和一个映射 μ : S → R，而映射的谓词 $P^{n_0}_{m_0},\dots,P^{n_k}_{m_k}$ 和 = 的投影 μ-原型，以及投影 μ-原型的补集，是 HW($\mathcal{M}$)中的Σ-关系（通常带有参数）。这一定义可以准确地概括这些模型，它们的签名即包含谓词符号，也包含函数符号。这一定义的重要性在于它允许将某个主题领域的模型形式化定义成语言中的Σ-定义集合，即：Σ-项集和Σ-函数集的集合。在这种情况下，Σ-定义允许采用递归策略，可以在Σ-定义中定义的项和谓词上施加约束。也可以将一个查询定义为一个Σ-函数的主题模型，函数的记录可以用于构建基本构造模型的签名构件，以及定义谓词和领域的术语项。

下面几种情况可以阐述我们介绍的概念和构件的卓越表达和计算能力[28-31,33-41]：

- 如果原始模型$\mathcal{M}$可计算（多项式可计算），它的列表超结构 HW($\mathcal{M}$)也可计算（多项式可计算）。
- 如果模型$\mathcal{M}$多项式可计算，它的列表超结构 HW($\mathcal{M}$)中由项定义的函数也是多项式可计算。
- 如果原始模型$\mathcal{M}$可计算，它的列表超结构 HW($\mathcal{M}$)中的Σ-关系可递归枚举，Σ-函数可计算。这一事实解释了术语“可计算性的可定义性

函数”的内涵。

- 如果模型$\mathcal{M}$多项式可计算，它的列表超结构 HW($\mathcal{M}$)中每个Δ_0-函数 $\varphi(x_1, \ldots, x_n)$的真值计算是多项式可计算。
- 对于可确定的模型$\mathcal{M}$和它的可计算的列表超结构 HW($\mathcal{M}$)，很不幸，它们的理论不确定。
- 在 HW($\mathcal{M}$)模型中，对于 n 元Σ－关系，普遍有一个(n+1)-元Σ－关系 $P^{(n+1)}(x; x_1,\ldots, x_n)$。在这个关系中，每个Σ－关系 $Q^{(n)}(x_1,\ldots,x_n)$都有一个元素 **a**，满足 $Q(x_1,\ldots,x_n) \leftrightarrow P(\mathbf{a}; x_1,\ldots,x_n)$。同时，在 HW($\mathcal{M}$)模型中没有普遍可计算的函数。
- 设 $\varphi(x_1 , \ldots, x_n; Q+)$是一定包含谓词 $Q(x_1,\ldots,x_n)$的Σ-函数，设谓词 Q 是列表超结构 HW($\mathcal{M}$)中的一个 n 元关系。定义 G 是 HW($\mathcal{M}$)上的一个操作：$G(Q) = \{(a_1, \ldots, a_n)|\ HW(\mathcal{M}) \models \varphi(a_1, \ldots,a_n; Q+)\}$。

则操作 G 单调，存在Σ-可定义的最小的固定点。注意，这一点非常重要，Gandy 固定点定理允许在主题领域的形式定义中使用递归结构构件。同时，通常建议看谓词符号 Q+，它一定在Σ-函数 $\varphi(x_1 , \ldots, x_n; Q+)$中，或者作为一个递归定义的谓词，或者是调用一个黑箱操作。因此，允许在语义可计算的概念中加入所谓的相对可计算性，这是用于建模的一种非常强的工具，特别是复杂层次结构的模型。

建议在实际应用中限制在Δ_0-函数类。借助于Δ_0-定义，通过引进所谓条件和递归Δ_0-项，可以显著提高原来语言的表达力[34-35]。例如，假设 φ 是一个Δ_0-函数，t_1和 t_2 是两个项，三元组(φ,t_1,t_2)是一个条件Δ_0-项，用如下可计算的程序解释：如果 φ 为真，执行 t_1 项；否则，执行 t_2 项。使用各种迭代和递归计算程序，递归项用类似的方法定义[35]，Δ_0-定义的语言通过这些构件的扩充，成为一种有非凡表达力的逻辑函数建模语言。更重要的一点是这种扩充保留了原有的多项式可计算性。此外，已经证明在一定条件下，反向操作也正确，即每个多项式可计算函数是Δ_0-可定义[39]。做到这一点，可以使用实践中更可接受的语言来描述Δ_0-函数类的问题。如果下面条件满足，我们说问题 P 可以形式化表达成 HW($\mathcal{M}$)：

- 主题领域可以用列表超结构 HW($\mathcal{M}$)的签名 Δ_0-模型 P 来定义。也就是说，用Δ_0-定义集的集合（再次回顾一下，递归Δ_0-定义是允许的）。
- 对Δ_0-模型 P 的请求可以用Δ_0-函数 φ，或者Δ_0-项 t 来表达。
- 作为对请求的答复 （即问题的解）：
- 如果对Δ_0-模型 P 的查询是Δ_0-函数 φ，则自由变量 φ 的确定需要函数 φ 为真。
- 如果请求的形式是Δ_0-项 t，则计算Δ_0-项 t 的值。

这里，我们要强调，求解问题 R 的准则是通过常量替代变量获得Δ_0-查询函

数的真值。注意，使Δ_0-查询函数为真的常量集合有多个。因此，从中挑选最优解是可能的。这一情况下，求解问题 P 的准则必须有选择查询最优答案的准则。形式化和求解问题的之一方法称作语义建模。需要注意上述阐述的形式化系统可以通过引进概率和模糊构件自然扩充。

e. 领域模型的知识归纳推理

文献[41-46]中定义主题领域是一个经验系统$\Im = \langle A, \Omega \rangle$。其中，A 表示主题领域（SD）的对象，$\Omega$表示本体。本体表示成 SD 概念系统中可解释的关系及操作集合。

文献[47]指出，知识是可以感知、认识和人性化的重要信息。因此，为了知识的归纳推理，人对 SD 本体的理解和解释很必要。

用机器学习方法对 SD 的知识做归纳推理，必须对对象的特性和属性做正确处理，以便获得由本体表达的知识。应注意，量的数值本身并不包含知识和信息，量的意思由它的具体单位的说明确定，例如，5 米、5 升、5 公斤等。单位的说明决定可能对量做哪些有意义的数学运算。值得注意的是，除了物理学其它主题领域测量的物理量失去了原有的物理解释。以温度的物理量为例，温度在非物理的范围变化。例如，医学测量病人体温、农业测量土壤温度、做饭烤炉测量空气温度等，尽管可能用相同温度计，但测量对象和目的应该有所不同。根据测量理论，测量大小的刻度是关系和操作的集合，只有用在指定的主题领域中的量的数值才有意义，也就是说，这些关系和操作需要用 SD 的本体解释。

温度计只能测量温度的说法是不对的，它实际测量的是所有情况下的物理温度。与其它度量手段一样，温度需要有它所属的 SD 本体的结论（知识）。病人体温是一种医学量的间接测量，即一种生命体征。对于土壤，温度用植物生理学和微生物的活动解释。其它主题领域中的物理温度是用其本体解释的某种非物理量的间接测量。这些主题领域的本体决定对这些量有意义的温度值的关系和操作。因此，为了提取主题领域的属性和量的信息，需要定义一组在本体Ω中阐述的数学关系和操作，并将它们包含在主题领域$\Im = \langle A, \Omega \rangle$的本体$\Omega$中。

考虑理论 Th($\Im$)的发现问题。设理论 Th($\Im$)是一组通用函数（文献[47]考虑了一种更通用的情况）。已知一组通用函数的逻辑等价于一组规则，表示为式（1）

$$\forall\forall x1,x2,\ldots(A1\&\ldots\&Ak \Rightarrow A0), k \geq 0, \qquad (1)$$

其中，A0,A1,...,Ak 是常量。所以，我们可以假设理论 Th($\Im$)是一组式（1）形式的规则。

不难看出，规则 $C = (A1\&\ldots\&Ak \Rightarrow A0)$在逻辑上遵循形式为$(Ai1\&\ldots\&Ain \Rightarrow A0)$的任意子规则，其中，$\{Ai1,\ldots,Ain\} \subset \{A1,\ldots,Ak\}, 0 \leq h < k$，也就是说，$(Ai1\&\ldots\&Ain \Rightarrow A0) \vdash (A1\&\ldots\&Ak \Rightarrow A0)$。因此，理论 Th($\Im$)可以化简。根据经验

系统$\Im = \langle A, \Omega \rangle$的定律，我们称式（1）为求真规则 C。因为，它的每个子规则在$\Im$上不再为真。然而，$L \vdash Th(M)$为真[16, 48]。因此，可以认为理论 $Th(\Im)$是一个经验系统的一组定律。

让我们定义 η 为经验系统$\Im = \langle A, \Omega \rangle$的概率，类似文献[49]中模型的概率。$\Im$的概率定律称作规则$(A1\&...\&Ak \Rightarrow A0)$，它的条件概率定义为$\eta(A0\&A1\&...\&Ak)/\eta(A1\&...\&Ak)$。其中，$(\eta(A1\&...\&Ak) > 0)$并严格地大于每个子规则的条件概率。强概率定律(SPL)称为概率定律 C, 不属于任何其它概率定律的子规则。

主题领域$\Im = \langle A, \Omega \rangle$的概率知识归纳推理遵循下列语义概率推理。

某种最强概率定律 Cn 的语义概率推理(SPI)用于预测某个事件 G，我们称序列 $C1 \sqsubset C2 \sqsubset ... \sqsubset Cn$, $Ci = (Ai1\&\ldots\&Aiki \Rightarrow G)$ 为概率定律，这里，Ci 是 Ci+1 的子规则，$\eta(Ci) < \eta(Ci+1)$, i = 1,2,...n-1。

我们需要预测的知识。定义 SPL 来解决统计二义性的问题[48, 50]，可以做无歧义的预测。

考虑用于预测每个事件 G 的所有 SPL 的集合，这个集合可以作为 G 的语义概率推理树。

用于推理某种事件 G 的最具体的概率定律(MSWZ(G))也称为 SPL，属于 G 的某种语义概率推理树，树的全部规则中存在最大条件概率值。所有常数 $G \in \Omega$ 的所有 MSWZ(G)定律集合写成 MSZ。

已经证明 $L \subset MSZ$ [41,47-48]，因此，定律集合 MSZ 概述成理论 $Th(\Im)$。另外，MSZ 逻辑一致[47-48,50]。所以，确切地说，它是领域$\Im = \langle A, \Omega \rangle$的概率理论。可以证明[47-48,50]，根据 MSZ 中定律通过归纳统计推理得到的预测结果是一致的。

主题领域$\Im = \langle A, \Omega \rangle$的知识归纳推理理论已经在下节讨论的"Discovery"软件系统平台上实现，许多实际应用问题可以通过这个平台解决（见 http://old.math.nsc.ru/AP/ScientificDiscovery/index_rus.html）。

6. 任务方法的平台解决方案

当前，语义建模作为一种自动求解智能问题的概念，不仅以任务方法的理论和方法论为基础，同时也已经开发出工具箱作可以使用，支撑和维护下面的智能问题求解的技术路线：

步骤 1. 阐明新需求，判别和研究由于缺少满足新需求的模板方法而产生的矛盾。这一矛盾的本质是要解决的问题的真正原因和内容。下一步，用自然语言形式化描述要克服所识别的矛盾的成功准则（解决问题的原型准则）。必要的话，将问题/任务分解，确定问题的上下文（为什么、最近的超任务、子任务、求解/不求解问题的目的和后果，等等）

步骤 2. 用自然语言描述与问题相关的一般知识，构建问题领域的本体模型（概念、事实、规则、关系、…），用常用的本体术语形式化问题关联的查询的类。

步骤 3. 用语义建模的逻辑概率术语构建问题的形式模型，用相同的形式术语定义查询和解的准则。

步骤 4. 用相关的语义建模工具平台构建问题的计算模型。

步骤 5. 用计算模型求解问题查询的答案。检查决策准则的可行性、答案的有效性和/或可理解性，以及答案可解释性。

至于语义建模的技术工具，已经开发了和正在积极开发许多平台解决方案。

3.1. DOSL 平台

任务方法和它的逻辑数学理论-语义建模-的一个成功应用案例是 DOSL 语义平台[www.dOsl.org]，是所谓的下一代业务规则引擎(BRE)的代表。如通常的 BRE 引擎一样，DOSL 允许使用陈述性逻辑规范语言部署业务逻辑。DOSL 平台的开发项目由 V.S.Gumirov 领导。

DOSL 平台采用主题领域专家易懂的 DOSL 语言控制复杂系统的行为逻辑。平台的应用范围广泛，从企业业务流程系统，到项目管理，或行为控制自动化系统，包括人工智能系统和物联网。DOSL 语言很容易通过增加特定主题领域的函数和多项进行扩展。此外，DOSL 本身是一种领域相关语言（DSL）。DOSL 提供语言层面的扩展机制，允许创建 DSL 语言层次架构，并转换成 DOSL 一部分，因此，可以在 DOSL 内核执行。

在应用层面，DOSL 语言被积极地用在电信、金融科技和银行系统中处理交易和系统事件，因为用 DOSL 语言可以很容易修改和重构应用系统中的业务逻辑。已经证明，使用 DOSL 平台进行企业数字化转型特别有效，允许快速开发高质量的自动决策支持系统、预测分析系统，以及各种业务的数字化模型，包括企业组织本身。

应该指出的是，DOSL 语言是 Eyeline Mobilizer 开放平台上的主要技术工具之一，支持各种平台上移动设备的服务开发和运行，包括 USSD/SMS/SS7、智能手机(Android, iOS)和移动互联网[www.eyeline.ru]。在 Eyeline Mobilizer 平台上开发的解决方案和实施的项目有电信、移动营销、金融、银行服务、市政管理和移动支付。目前，Eyeline Mobilizer 平台为几千万移动设备用户提供解决方案和服务。平台使用户显著地降低了开发和运营移动业务解决方案的费用。成功实施的案例有：MTS (*100#, *111#, etc.)的 USSD 服务和门户网站 USSD portal Alfa-Click *142# Alfa-Bank (Russia)、InnomaNet/T-mobile (Russia)、WATAGO Africa (尼日利亚最大的移动支付系统)、莫斯科市政府停车系统项目、未接电话通知系统、菲律宾

Yellow Pages/SMART、云客户服务系统GlobalUSSD.com、Beeline Russia（直接支付和转账），等等。

3.2 bSystem 平台

bSystem是构建机构组织和业务流程数字孪生系统的平台，由A.V. Mantsivoda研究组用了多年时间开发。平台聚焦大型商业环境的智能管理系统开发、企业数字化转型和其它集成解决方案。根据E.E. Vityaev的建议采用了统一的语义建模环境。一方面，bSystem将基本层次的本体与归纳逻辑概率知识推理和预测规则/决策的语义逻辑推理做了集成；另一方面，集成了经典程序设计。

因此，bSystem的架构基于三个层次的集成：逻辑概率层、分析/本体层和操作/业务层。

本体层控制仿真机构的主要参数[51-52]，反应各组成部分和它们之间连接的当前状态，提供操作层决策的分析工具（例如，商业智能任务）。这一层是平台的内核，管理所有本体，也就是基本的领域知识系统，围绕基本的领域知识系统开发所有应用系统。

逻辑概率推理层（LPI）建立在内核层之上，实现可解释人工智能的功能。在平台上，LPI负责解决预测、决策和其它AI任务的战略问题。它在bSystem中的价值是：LPI在学习时生成概率逻辑理论（模式集合），以语义形式描述应用领域范畴。这一点与神经网络不同，神经网络积累神经元的数值参数，这些参数不提供语义解释。LPI生成的理论确保了解释决策的能力。这也潜在地提供了自控能力，即评估用逻辑的元工具生成的知识的优缺点。这种能力神经网络难以达到。

底端的操作事务层是方法和程序层，随着时间改变逻辑环境。在这一层，bSystem运行面向对象语言Libretto。以任务方法观点，操作事务层构成任务生命周期的管理环境。在这一层控制的任何业务流程不过是实时完成任务的流程（实践中，某个归纳的任务实例）。通常，一个数字孪生系统是通过一组主动地交互任务随着时间展开的，伴随着满足解的条件的新的任务实例的生成和结束。任务的说明描述和解的准则由较高的本体层控制，求解过程本身作为作业链在操作层执行。同时，业务流程本身表示成本体对象，允许平台在说明层控制求解流程。因此，bSystem通过本体层和操作层的交互提供任务方法的完整实现。

bSystem的关键属性是三个层次都嵌在单一语义空间，它们相互交互并提供垂直扩展功能。这种交互产生强有力的协作效果，不仅有助于增强量的标志（在常用的各种信息技术范围内无缝交互），而且能产生质变。例如，这种集成能够从本质上实现一种低代码技术，这种面向对象的低代码技术基于本体数据的自动代码合成[53]。这一技术与传统的程序低代码技术有本质区别，程序

低代码基于简图- BPMN 规范的变体[54]。

各类本体是确保 bSystem 数字环境完整性的核心工具。实际上，一个孪生系统是一个本体或本体网络，扮演语义建模准则的角色，用于定义所表示的领域，以及要解决的任务。

一个本体也可以被认为是一个确保数字孪生系统各个部分相互交互的中心在逻辑概率层，各类本体扮演经验系统的角色，包括先验知识，根据先验知识决定学习需要的信息。在分析本体层的核，本体是数字孪生系统的语义基础和解析运算的信息源。最后，在程序层，本体作为平台编程语言 Libretto 的数据类型系统。

发现一个合适的本体架构是一个很难的问题。足可以说，bSystem 是构建这类本体管理平台的第六次尝试（唯一的相当有效的尝试）。为了更好的理解我们真正需要什么类型的本体，需要先开发一个本体必须满足的需求系统。下面列出一部分需求。

首先，正如前面指出，本体要确保平台系统的垂直连接和完整的语义空间本体的知识必须在 bSystem 所有层次中能有效解析：包括复杂的逻辑/概率推理机、分析模块（例如，商务智能）和求解传统算法问题的软件。

其次，用户要能用本体，即使他们缺少对数学逻辑的理解（理解的困难是著名语义网概念失败的关键障碍之一[55]）。由于本体提供人与平台互动期间相关领域的基础知识，如果人不理解本体，做任何事都变得无意义。所以，构建能够容易理解、与用户日常经验一致的本体也是必须做的工作。

此外，本体必须提供与外部系统交互的标准机制（信息系统、神经网络和物联网）。例如，从安装在住宅综合楼的控制器上采集数据是住宅综合楼数字孪生系统不可或缺的正常功能。为了与外界交互，本体必须备有众所周知的程序接口（API）的逻辑仿真装置。

最后，如果让本体可以实际应用，它们必须高效。意思是说，高层次本体的实施必须与关系数据库的生产力有可比性。本体必须能够管理百万级实体，实时提供复杂查询的答案。

将所有这些需求与一个形式化系统结合起来是一个相当困难的任务（例如基于众所周知的 OWL 语言的本体与上述的一些需求就不相符[56]）。在实践过程中，我们发现基于对象建模想法的本体与这些需求最相符。

对象模型的基本特征是这些模型有两个清晰的层次，说明层和程序层。第一个由类、对象和多项的属性组成。第二个层提供在类中定义的方法，这些方法在对象上操作。由于本体是领域中的基本概念、事实和关系的形式表示，对象模型可以处理程序中的类似任务，对象模型的说明层可以作为本体的充足的背景。此外，对象模型的本质非常简易、自然，为人类所熟悉。类似 Java 的面向对象语言的广泛使用证明了这一点。所以，相当可能做到给对象模型的说明

层定义严格的逻辑语义，将它转成本体的变体。

至于程序层，面向对象的语言通常是图灵完备，这一特点使情况的可控性差得多。另一方面，程序层的出现，使随着时间开发本体和管理本体的生命周期的想法成为可行。根据这一点，每个方法或每组方法的应用可以当作一个业务，将本体从一种状态转换到另一种状态[57]。这一能力在解决管理问题时特别有用，因为这种解释可以将业务流程定义成本体生命周期的业务链[58]。

所以，将对象模型转成本体需要下面步骤：

1. 定义与对象模型的说明层相关的逻辑形式系统。用语义建模的 Δ0-可定性机制完成这一步。
2. 传统的对象模型生存很短，通常存在 RAM 中。因为本体是数据和知识的长期存储，需要将本体持久化。这一步要为本体提供对象数据库功能。
3. 使用本体需要一个查询语言，这个语言要将逻辑表达能力与效率相结合，使它与关系数据库的 SQL 有可比性。在 bSystem 系统，Libretto 的陈述核部分有严格的逻辑语义，用来作为查询语言。
4. 为了提供随时间修改和开发本体的机制，将对象模型的程序层重新格式化成一个业务模型。
5. 实践中发现，对象生命周期管理工具显著增强了比传统对象模型更强的功能，对业务流程管理特别有用。使用业务机制，我们对对象生命周期的控制更强。
6. 最后一步是实现本体的自主性。一个本体被裹在一个保护层中，与外部世界隔离，只能通过严格规范的接口才能与保护层通信（类似于程序的 API）。

从任务方法看，第五步实现了任务生命周期的两层管理，即：底层操作的执行和本体层的语义控制。

最后一步实现了局部约简的概念，将任意复杂系统构建成基于逻辑交互机制、相对简单的本体网络，可以在开发数字孪生系统时自由地做水平扩展。

从 IT 的观点看，第二步和最后一步将一个对象本体转换成一个微服务[59]，因为这些步骤提供了微服务的两个主要特征--数据存储自治（体现在本体）和与外部世界交互通道的严格控制。这点也决定了在 bSystem 上开发的数字孪生系统的总架构。

bSystem 为我们提供了全新的系统开发方法论[60]。他的开放性建模方法从跟本上改变了参与者之间在开发和运行阶段的交互机制。本体对所有不同角色参与者直接可用。例如，客户完全控制开发过程，可以对开发中发现的问题及早提供反馈。由于严格的逻辑语义，高层模型可以同时当作系统的规范和执行文档，与软件代码的黑箱方式有本质的不同。

3.3. Discovery 平台—归纳知识推理的本体方法

前面已经提到，E.E. Vityaev 研发了知识发现的本体方法和软件系统 Discovery [6,42,50-51]。这一方法用于从数据提取的信息中发现知识，表达成一个经验系统 $SD\ \Im = \langle A, \Omega \rangle$。Discovery 系统比其它机器学习方法有如下优点：

1. 采用从本体Ω的数据提取的信息，消除了数据类型的限制；
2. 用本体Ω定义先验知识，消除了先验知识表达类型的限制；
3. 用本体Ω定义的知识类（规则形式）表达测试的假设类，消除了对测试假设类的表达类型的限制；
4. Discovery 系统能够识别下列模式类型：
   a) 经验系统$\Im$的定律集和理论 $Th(\Im)$，
   b) 概率定律集和最强概率定律集，
   c) 用于无歧义预测的最具体的定律集[48,52]。

在本体方法中，发现的知识的完备性有两个意义：（1）使用本体和度量理论从数据中提取的信息的完备性；（2）用规则 a)-c)表达发现的知识的完备性。

当检测的模式预先不知道时，已有的机器学习方法不支持数据探索。每个机器学习方法只能揭示与其本体相关、相当具体的假设的类。Discovery 系统能够发现专家可指定的任意类型的假设，当研究过程中假设类型改变时，能够支持数据探索模式。

Discovery 系统发现的知识用 SD 本体表示。当涉及医学、金融或军事应用领域的决策时，模式的互操作可能很重要。

Discovery 系统被用于解决许多问题，包括金融预测[6,42]、生物信息学[64]、医学[65]、欺诈分析[66]和其它领域。

3.4. Delta 平台

在引言中提到所谓的中心化问题，这是当代人工智能迫切需要解决的一个问题，另一个问题是众所周知的黑箱问题，意味着一种趋势，即：人工智能的能力和控制被集中在少数法律实体的人的手中。解决这一问题，引言中建议采用虚拟货币采用的去中心化经验。同时，建议将 AI 算法本身当作多链区块链子结构上运行的智能合约，能够确保所有操作的透明、训练样本的透明和决策透明。最后提出了 AI 算法多项式可计算的复杂度要求，有明显的实践意义。总的来讲，最终任务是开发任何意义上多项式可计算的可信语义建模系统。

2017 年，S.S. Goncharov[34]提出了条件项可以理解成条款不仅满足第一谓词逻辑的标准项，也满足正确定义的句法结构的其它项。根据这一结果，S.S. Goncharov and D.I. Sviridenko 在 2019 年提出将这些扩展项分为有界递归项、B-while 项、迭代项等[33]，他们指出所有这些扩展项都没有超出多项式的计算复杂度。但是，他们没有解决这些扩展项的 P-完全问题。然而，在

2022年，S.S. Goncharov and A.V. Nechesov发现了最早的句法结构，完成了P-完全语义建模语言L的构建[39]。在这之前的2021年，他们还证明了Gandy固定点定理的多项式模拟结果[36]，可以说，在特定Δ_0-系统框架内能够找到这样的函数条件，使Gandy固定点定理的经典结果在这一系统有效。此外，重要的一点是，这一系统中构建的操作的最小固定点是多项式可计算。这一结果对进一步发展语义编程语言L和形式化Δ_0-系统本身提供了更大的动力。

注意，Gandy固定点定理的多项式模拟结果可以让我们沉浸在形式系统任意归纳定义的句法结构中，提出评估这种存储复杂度的编码和手段。根据这一点，S.S. Goncharov, D.I. Sviridenko and A.V. Nechesov阐述了P-完全语义编程语言L*的面向对象版本[40]，并开始开发它的专用框架-Delta平台。这个平台已经产业化，包括解决上述提到的多区块链和智能合约问题所需的必要功能。L*语言基于P-完全逻辑语言L开发。同时，如果有必要，可以在多链区块链的不同层次设置不同的用户访问层，为区块链本身和在区块链上运行的智能合约设置不同的透明等级。由于智能合约写成P-完全语言L*的特定类型程序，显著地降低了合约审计的人工成本。用L*语言本身可以描述神经网络[61]和区块链[62]操作，这是一个不用质疑的优点，因为实现这些操作不再需要其它外部软件。

采用P-完全语言L*解决问题的方法，我们不仅描述问题的本身，也用同一语言的框架和工具解决问题。由于我们的以区块链为基础，可以很简单地搭建去中心化的解决方案。例如，比特币或Ethereum区块链是最安全的去中心化区块链，可以作为多区块链的最顶层。下层的区块链不必去中心化，可以当作正常的数据库，放在每个用户的电脑上。因为没必要与其它区块链达成任何共识，可以确保数据高速读写。

需要引起读者注意，Delta平台的开发同Discovery平台一样集成了其它预测系统，为构建AI算法提供了强大动力。这些算法的本质非常接近通用人工智能（AGI）算法，确保了智能系统的高度自学习能力和普通用户对系统的高可信度。有了上述组合的可能性，可以看出，任务方法是在下面情况下实现系统的主要配备：AI系统能够解决赋予它们的任务并独立形式化新的任务，而这些新任务又不能在AI算法中明显陈述。有了这些就允许我们谈论在任务方法的框架内构建可信的通用AI的可能性。

7. 结论

本文可以作为任务方法领域的一份研究报告。任务方法的实际应用是本文作者、他们的学生和同事完成的。这些应用起始于上个世纪70-80年代，随着多年的发展，任务方法的发展和应用产生了许多成果。这些工作不仅与数学和人工智能的各种问题有关，也关联其它领域，包括认知科学、神经生理学、医学、企业数字化转型、复杂系统设计的自动化、数字孪生，等等。从前面的讨

论可以看到，不仅可以开发任务方法的方法论和数学工具，也可以建立有效而重要的技术储备，这些技术储备可以用于解决各领域范围广泛的任务，如：电讯、商业、零售、金融科技、遗传学、地质学、网络安全、机器人学、医学等基于可信通用人工智能的研发也进入日程。正如早年的经验展示，有各种理由认为，这样的人工智能在不远的将来会出现，而在通用人工智能框架内将包括任务方法。

Литература ***References*** ***參考書目***

1. Blockchain trilemma https://coinmarketcap.com/alexandria/glossary/blockchain-trilemma

2. Durov, N., Durov, P. White Paper Ton-Blockchain, 2017 https://www.docdroid.net/NFTQHzI/ton-pdf

3. Phillips, P. , Hahn, C. , Fontana, P. , Yates, A. , Greene, K. , Broniatowski, D. and Przybocki, M. (2021), Four Principles of Explainable Artificial Intelligence, NIST Interagency/Internal Report (NISTIR), National Institute of Standards and Technology, Gaithersburg, MD, [online], https://doi.org/10.6028/NIST.IR.8312,

4. Notice of the Cyberspace Administration of China on Public Comments on the "Administrative Measures for Generative Artificial Intelligence Services (Draft for Comment)" http://www.cac.gov.cn/2023-04/11/c_1682854275475410.htm

5. National standard of the Russian Federation, Artificial intelligence systems, ways to ensure trust, GOST R 59276-2020 https://docs.cntd.ru/document/ 1200177291

6. Notice of the Cyberspace Administration of China on Public Comments on the "Administrative Measures for Generative Artificial Intelligence Services (Draft for Comment)" http://www.cac.gov.cn/2023-04/11/c_1682854275475410.htm

7. Ershov Yu. L., Samokhvalov K. F. On a new approach to the philosophy of mathematics // Structural analysis of symbolic sequences. Novosibirsk, 1984. Issue 101. pp. 141-148. (in Russian)

8. Ershov Yu. L., Samokhvalov K. F. Modern philosophy of mathematics: ailments and treatment. Novosibirsk: Parallel, 2007. 142 p. (in Russian)

9. Sergei S. Goncharov, Dmitrii I. Sviridenko, Evgenii E. Vityaev. Task Approach to Artificial Intelligence // Proceedings of the Workshop on Applied Mathematics and Fundamental Computer Science 2020, Omsk, Russia, April 23-30, 2020. CEUR Workshop Proceeding. Vol. 2642.

10. S.S.Goncharov, E.E. Vityaev, D.I. Sviridenko. Problem-solving approach in artificial intelligence // Applied Mathematics and Fundamental Informatics. - 2020. - T. 7. - No. 2. - P. 4-9

11. Sviridenko D.I. Semantic Smart Wallets // 2019 International Multi-Conference on Engineering, Computer and Information Sciences (SIBIRCON), Novosibirsk, Russia, 2019, pp. 991-994.

12. D.I.Sviridenko, E.E.Vityaev. A task-based approach to artificial intelligence and its theoretical and technological base // 18 National conference on artificial intelligence with international participation (CII-2020), (October 10-16, 2020, Moscow, Russia). Conference proceedings / Ed. V.V. Borisova, O. P. Kuznetsova. - M: MFTI, 2020. (326 p.). S. 36-44.

13. S.Russell, P.Norwig. Artificial Intelligence: A Modern Approach. Kiev: Williams, 2006.

14. STRONG ARTIFICIAL INTELLIGENCE. On the approaches to the supermind // Alexander Vedyakhin [et al.]. — M.: Intellectual Literature, 2021. — 232 p. (in Russian)

15. Anokhin P. K. Fundamental questions of the theory of functional systems // Philosophical aspects of the theory of functional systems. Moscow: Nauka, 1978. pp. 49-106. (in Russian)

16. Vityaev E.E. The logic of the brain // Approaches to modeling thinking. ed. by V.G. Redko. URSS Editorial, Moscow, 2014, pp. 120-153. (in Russian)

17. Vityaev E.E. Mathematical probabilistic model of cognitome and functional systems // Artificial intelligence, algebraic biology and theory of systems in mathematical research of Russian scientists. Volume 1. Proceedings of the Meeting "Interdisciplinary interaction of algebraic biology, systems theory and artificial intelligence". Special Issue of the journal Biomash Systems, Volume 5, No. 1, January-March, 2021, pp. 238-292. (in Russian)

18. Anokhin, P.K.: Biology and neurophysiology of the conditioned reflex and its role in adaptive behaviour, Oxford etc.: Pergamon press, pp. 574. (1974).

19. Sudakov K.V.: The general theory of functional systems. Moscow: Medicine, p.222. (1984) (in Russian)

20. Evgenii E. Vityaev Purposefulness as a Principle of Brain Activity // Anticipation: Learning from the Past, (ed.) M. Nadin. Cognitive Systems Monographs, V.25, Ch. 13. Springer, 2015, pp. 231-254.

21. Kolmogorov A. N. Grundbegriffe der Wahrscheinlichkeitrechnung, in Ergebnisse der Mathematik. — Berlin, 1933.

22. Altshuller G. S. Finding an idea: An Introduction to the TRIZ – Theory of solving inventive problems, 3rd ed. — Moscow: Alpina Publisher, 2010. p. 392. (in Russian)

23. Mukhortov V.V., Khlebnikov S.V., Vityaev E.E. Improved algorithm of semantic probabilistic inference in the problem of 2-dimensional animate, Neuroinformatics. 2012. Vol.6, No. 1, pp. 50-62. (in Russian)

24. Demin A.V., Vityaev E.E. Learning in a virtual model of the C. elegans nematode for locomotion and chemotaxis, Biologically Inspired Cognitive Architectures. 2014, v.7, pp.9–14.

25. Vityaev, E.E., Demin, A.V., Kolonin, Y.A. (2020). Logical Probabilistic Biologically Inspired Cognitive Architecture // Goertzel, B., Panov, A., Potapov, A., Yampolskiy, R. (eds) Artificial General Intelligence. AGI 2020. LNCS, v. 12177. Springer, Cham.

26. Ershov Yu. L. Definability and computability. Novosibirsk, 2000. (in Russian)

27. Goncharov S. S., Sviridenko D. I. Semantic modeling and artificial intelligence // Siberian Philosophical Journal. 2018. vol. 16, No. 4. pp. 5-25. (in Russian)

28. Goncharov S. S., Sviridenko D. I. Theoretical aspects of Σ-programming. Lecture Notes in Computer Science, 1986, vol. 215, p. 169–179.

29. Goncharov S.S., Ershov Yu.L., Sviridenko D.I. Semantic programming. In: Proc.IFIP 10-th World Comput. Congress. Dublin. Vol. 10. 1986. pp 1093–1100.

30. Goncharov S.S., Ershov Yu.L., Sviridenko D.I. Semantic foundations of programming, // LNCS. Vol. 278. 1987. pp. 116–122.2.

31. Goncharov S.S., Sviridenko D.I. Σ-programming // Transl., II.Ser., Am. Math. Soc., 1989, vol. 142, p. 101–121.

32. Goncharov S., Sviridenko D. Semantic Modeling and Hybrid Models // 2019 International Multi-Conference on Engineering, Computer and Information Sciences (SIBIRCON), Novosibirsk, Russia, 2019, pp. 987-990.

33. Goncharov S. S., Sviridenko D. I. Logical language for describing polynomial computability // Reports of the RASciences, 2018. Vol. 485, No. 1. pp. 11-14. (in Russian)

34. Goncharov S.S. Conditional terms in semantic programming // Siberian Mathematical Journal. 2017. Т. 58. № 5. С. 794-800.

35. Goncharov, S.S. Sviridenko, D.I. Recursive Terms in Semantic Programming (Article) // Siberian Mathematical Journal. - Volume 59, Issue 6, 1 November 2018, Pages 1014-1023.

36. Goncharov, S., Nechesov, A. Polynomial analogue of Gandy's fixed point theorem. Mathematics, 2021, 9(17), 2102.

37. Goncharov S., Ospichev S., Ponomaryov D., Sviridenko D. The expressiveness of looping terms in the Semantic Programming. Siberian Electronic Mathematical News. 2020, 380-394.

38. Nechesov, A. Semantic Programming and Polynomially Computable Representations. Siberian Advances in Mathematics. 2023. V.33. N1. P.66-85.

39. Goncharov, S., Nechesov, A. Solution of the Problem P = L. Mathematics, 2022, 10(1), 113.

40. Goncharov, S.S.; Nechesov, A.V. Semantic programming for AI and Robotics. IEEE International Multi-Conference on Engineering, Computer and Information Sciences (SIBIRCON)}, Yekaterinburg, Russian Federation, 2022, pp. 810-815.

41. Vityaev E. E. Extracting knowledge from data. Computer cognition. Models of cognitive processes. Novosibirsk, 2006. 293 p. (in Russian)

42. Pfanzagl I. Theory of measurements. M.: Mir, 1976. 248 p.

43. B. Kovalerchuk and E. Vityaev. Data Mining in Finance: Advances in Relational and Hybrid Methods, Kluwer Acad. Pub., 2000.

44. Krantz D.H., Luce R.D., Suppes P., Tversky A. Foundations of Measurement. Acad. Press, N.Y.; L. 1971; 1989; 1990. Vol. 1–3.

45. E. Vityaev, B.Y. Kovalerchuk, Relational Methodology for Data Mining and Knowledge Discovery. Intelligent Data Analysis. Special issue on “Philosophies and Methodologies for Knowledge Discovery and Intelligent Data Analysis” eds. Keith Rennolls, Evgenii Vityaev. v.12(2), IOS Press, 2008, pp. 189-210.

46. Evgenii Vityaev, Boris Kovalerchuk. Ontological Data Mining. Uncertainty Modeling: Dedicated to Professor Boris Kovalerchuk on his Anniversary. Studies in Computational Intelligence 683, V. Kreinovich (ed.), Springer, 2017, pp. 277-292.

47. Vityaev E.E. Semantic probabilistic inference of predictions // Izvestia of Irkutsk State University, Series "Mathematics". 2017. Volume 21. pp. 33-50. (in Russian)

48. Evgenii Vityaev The logic of prediction. In: Proceedings of the 9th Asian Logic Conference (August 16-19, Novosibirsk, Russia), World Scientific Publishers, 2006 pp.263-276.

49. Halpern, J.Y. An analysis of first-order logic of probability. Artificial Intelligence. v. 46, 1990. – P. 311-350.

50. Vityaev, E., Odintsov, S. How to predict consistently? Trends in Mathematics and Computational Intelligence In: Studies in Computational Intelligence, 796, Mar?a Eugenia Cornejo (ed), 2019, 35-41.

51. Malykh A.A., Mantsivoda A.V. Object theories over list superstructures // Izvestiya Irkutsk State University. Mathematics series. 2012. Vol. 4. pp. 27-44. (in Russian)

52. Malykh A.A., Mantsivoda A.V. Documentary modeling // Izvestiya Irkutsk State University. Mathematics series. 2017. Vol. 21. pp. 89-107. (in Russian)

53. Gavrilina D. E., Mantsivoda A.V. Low-code and object spreadsheets // Izvestiya Irkutsk State University. Mathematics series. 2022. Vol. 40. C. 93-103. (in Russian)

54. Business Process Model and Notation (BPMN). Version 2.0. URL: https://www.omg.org/spec/BPMN/2.0/PDF (accessed 1 April 2022).

55. T.Berners-Lee, J.Hendler, O.Lassila. The Semantic Web. Scientific American 284 (5), 34-43, 2001.

56. OWL2 Web Ontology Language. Structural Specification and Functional-Style Syntax (Second Edition). W3C Recommendation 11 December 2012. URL: https://www.w3.org/TR/owl2-syntax/ (accessed 1 April 2022).

57. Mantsivoda A.V., Ponomaryov D.K. Towards Semantic Document Modelling of Business Processes // News of Irkutsk State University. Mathematics series. 2019. Vol. 29. pp. 52-67.

58. Mantsivoda A.V., Ponomaryov D.K. On Termination of Transactions over Semantic

Document Models // News of Irkutsk State University. Math. series. 2020. Vol. 31. pp. 111-131.

59. Gavrilin D. N., Kustova I. A., Mantsivoda A.V. Object models as microservices: query language // Izvestiya Irkutsk State University. Math. series. 2022. Vol. 42. C. 121-137. (in Russian)

60. Kazakov I.A., Kustova I.A., Mantsivoda A.V. Documentary modeling: methodology and applications // News of Irkutsk State University. Math. series. 2020. Vol. 32. pp. 79-93. (in Russian)

61. Goncharov, S.; Nechesov, A. Polynomial-Computable Representation of Neural Networks in Semantic Programming. J 2023, 6, 48-57.

62. Goncharov, S.; Nechesov, A. Axiomatization of Blockchain Theory. Mathematics 2023, 11, 2966.