

УДК: 004.6+ 004.4

Современные промышленные приложения графов знаний

Апанович З.В (Институт систем информатики СО РАН)

Графы знаний проделали большой путь эволюции от простого множества RDF-триплет до систем получения новых знаний. Если в прежние годы основным приложением графов знаний считался семантический поиск, то на современном этапе графы знаний проникают во все области промышленного производства. Данная работа представляет обзор новых вариантов графов знаний, таких виртуальные графы знаний, динамические графы знаний и исполняемые графы знаний, применяемые в современном производстве, а также основную область их применения когнитивные цифровые двойники. Также в работе кратко рассмотрены способы генерации графов знаний при помощи больших языковых моделей и повышение качества работы больших языковых моделей за счет применения графов знаний.

Ключевые слова: *виртуальный граф знаний, динамический граф знаний, исполняемый граф знаний, когнитивный цифровой двойник, большие языковые модели*

1. Введение

Графы знаний (ГЗ) с момента их громкого появления под лозунгом «вещи, а не строки» (“things not strings”) прошли большой путь эволюции. Если первоначально граф знаний понимался просто как RDF-граф, то есть множество триплет в виде (субъект, предикат, объект), то к настоящему моменту графы знаний превратились в системы получения новых знаний [1]. Если в начальные годы основным приложением графов знаний считался семантический поиск, то сейчас в список приложений входят вопросно-ответные системы, рекомендательные системы, системы принятия решений и многие другие. В последние несколько лет графы знаний все чаще используются при разработке цифровых двойников и даже появился специальный термин «когнитивный цифровой двойник», означающий системы, совместно использующие методы семантического моделирования и глубокого обучения.

Одним из факторов такой эволюции графов знаний является то, что графы знаний унаследовали от *Открытых связанных данных* (Linked Open Data) стандарты, входящие в стек технологий Semantic Web [2].

1. Использование глобальной идентификации ресурсов на основе интернационализированных идентификаторов ресурсов (Internationalised Resource Identifier, IRI), обеспечивает глобальную однозначность представления каждого ресурса.
2. Использование Resource Description Framework (RDF) [3] позволяет стандартизировать представление данных в форме триплетов.
3. Использование онтологий для классификации ресурсов позволяет генерацию новых триплет как при помощи логического вывода, так и при помощи методов глубокого обучения на основе текстовых, визуальных и других данных.
4. Использование специализированных систем хранения, таких как RDF-хранилища и графовые базы данных, позволяет использовать наиболее эффективный способ хранения данных. 1
5. Язык запросов SPARQL позволяет не только осуществлять запросы, но и обновлять данные в хранилищах через конечную точку. Федеративные запросы предлагают возможность извлекать данные из нескольких источников данных и анализировать их одновременно, включая возможность объединения данных из общедоступных и частных источников данных, принадлежащих любому количеству различных сторон.
6. Доступ данным при помощи Виртуальных графов знаний, ранее известный как Доступ к данным на основе онтологий (Ontology-Based Data Access, OBDA) позволяет выполнять онтологические запросы над реляционными и NoSQL базами данных.

В настоящее время семантические модели внедряются в различных отраслях Промышленного Интернета вещей для моделирования и управления знаниями предметной области. Их приложения варьируются от управления следующим поколением производства до объяснимого транспорта и энергосбережения в зданиях, от использования семантической интеграции различных датчиков Интернета Вещей (IoT) до автоматизации аналитики созданных данных. Количество разновидностей графов знаний значительно расширяется за счет новых приложений. Ведущие производители такие как Siemens, Bosh, IBM предлагают и внедряют новые вариации ГЗ для промышленных приложений, такие как Динамические графы знаний, Исполняемые графы знаний, Виртуальные графы знаний и др. В данной

работе будет сделан обзор современных графов знаний и их приложений, обусловленных потребностями четвертой и пятой промышленных революций.

2. Виртуальные графы знаний для интеграции множественных источников данных

Тенденция цифровизации в обрабатывающей промышленности приводит к огромному росту объема и сложности данных, генерируемых устройствами, участвующими в производственных процессах. Эти данные становятся важным активом для улучшения эффективности производства. Однако раскрытие потенциала этих данных - серьезная проблема для многих организаций. Часто данные находятся в разрозненных хранилищах, которые не связаны между собой физически, но содержат *семантически связанные* данные, возможно с избыточной и противоречивой информацией. Поэтому эффективное использование данных требует *интеграции данных*, которая включает их очистку, дедупликацию и семантическую гомогенизацию. По оценке компании Bosch [4], усилия по интеграции данных составляют примерно 70-80 процентов по сравнению с 20-30 процентами, необходимыми для анализа данных. В последние годы для решения этой проблемы используются методы семантической интеграции данных на основе *виртуальных графов знаний* (ВГЗ) [5].

В подходе ВГЗ онтология моделирует предметную область и определяет общий словарь виртуального графа знаний, который скрывает от пользователя физическую структуру источников данных, а также обогащает данные из источников некоторыми общими знаниями. Онтология связана с конкретными источниками данных через декларативные спецификации данных в терминах отображений (mappings), которые связывают классы и свойства онтологии с конкретными представлениями данных в разрозненных источниках. Отображения указывают, каким образом превратить значения, хранящиеся в разрозненных реляционных таблицах в триплеты единого графа знаний.

Онтология вместе с отображениями представляет собой виртуальный граф знаний, к которому можно осуществлять запросы SPARQL. Запросы формулируются в терминах онтологии, описывающей предметную область и пользователю не требуется понимание конкретных источников данных, знание о взаимосвязях между этими источниками или способе кодирования данных в отдельных источниках. Ключевой технологией является технология переписывания запросов (query rewriting), которая позволяет избежать физической материализации источников в графе знаний. Запрос SPARQL транслируется в

серию SQL запросов к различным источникам данных, а затем результаты запросов собираются в единый ответ на исходный SPARQL запрос. Схема этого подхода показана на рисунке 1.

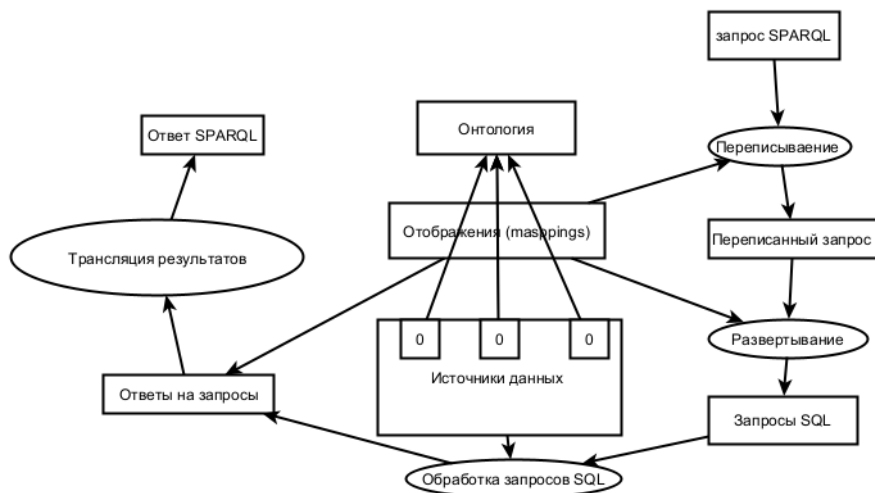


Рис. 1. Схема метода переписывания запросов для работы с виртуальным графом знаний.

Преимущество этого подхода заключается, с одной стороны, в знаниях о предметной области, закодированных в онтологии. Эти знания используются для обогащения ответов на запросы пользователей, описывающих задачи анализа продукта. Другим ключевым преимуществом является использование семантически насыщенных отображений ВГЗ. Они устраняют несоответствие между уровнем данных и уровнем онтологии, используя основанный на шаблонах механизм R2RML [6] для построения уникальных идентификаторов (IRI) графа знаний из значений базы данных. Более того, сопоставления обеспечивают решение задачи интеграции, поскольку семантически однородная информация, поступающая из разных файлов журналов, которые используют синтаксически различные представления, может быть согласована на уровне ГЗ. Это упрощает выполнение запросов ко всем данным интегрированным образом, используя семантику извлеченной информации.

В качестве примера рассмотрим использование виртуальных графов знаний для анализа качества поверхностного монтажа печатных плат на заводах BOSH [4]. Процесс поверхностного монтажа (surface mounting Process, SMT) включает четыре основных этапа (см. Рисунок 2):

(1) Нанесение паяльной пасты: этап состоит из нанесения паяльной пасты на печатные платы (ПП) и выполняется с помощью машины для печати паяльной пастой;

(2) Поверхностный монтаж: этап, где электронные компоненты фактически устанавливаются на печатные платы при поверхностном монтаже устройств (surface mount devices, SMD); по-русски говорят SMD компоненты, SMT технологи, и SMD-монтаж;

(3) Нагрев: для правильной пайки смонтированных компонентов на этом этапе. платы нагреваются в печи оплавления;

(4) Автоматизированная оптическая инспекция (automated optical inspection, AOI.

На заключительном этапе платы проверяются устройством AOI, чтобы выяснить, не произошел ли на предыдущих этапах какой-либо сбой, например, неправильная установка компонентов или неисправность пайки. Когда весь процесс завершен, система генерирует файлы, которые содержат два типа данных:

- журналы размещения, создаваемые SMD-устройством, содержат информацию о том, какой компонент установлен на какой плате;
- журналы неисправностей, генерируемые устройством AOI, показывают, где на плате и с какой компонентой случилась неисправность

Проблема состоит в том, что разные этапы этого процесса реализуются при помощи разных устройств, а эти устройства, как правило, поставляются разными производителями и имеют разные форматы и схемы для управления одними и теми же данными в процессе поверхностного монтажа.

Следовательно, необработанные, не интегрированные данные не дают целостного представления обо всем процессе SMT и затрудняют анализ выпускаемой продукции.

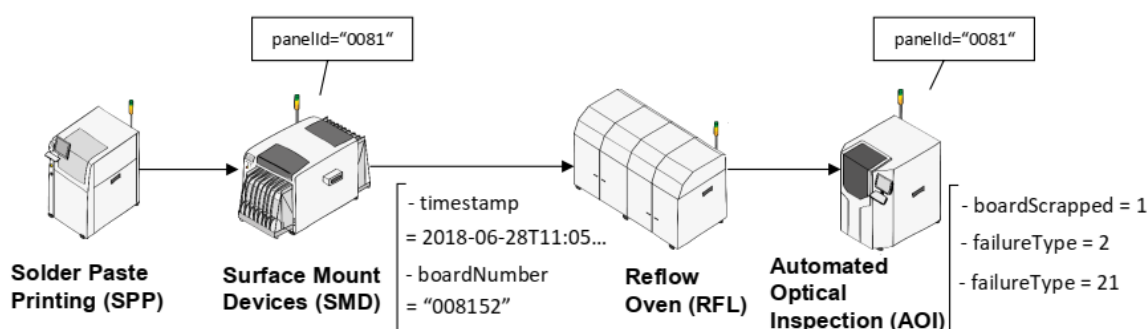


Рис. 2. Устройства, участвующие в процессе SMD-монтажа [4].

Каждое из устройств генерирует свои собственные данные (журналы), регистрирующие детали процесса обработки. Например, устройство оптического контроля, порождает таблицы такие как `aoi_event`, `aoi_location`, `aoi_panel` и `aoi_failures`. На рисунке 3 показаны фрагменты таблиц устройств. Таблица `aoi_failures` содержит столбец `failureCode`,

принимаящий различные целочисленные значения в зависимости от типа сбоя. Эти численные значения могут быть различными даже в рамках одного предприятия.

SMD Tables	smd_panel
smd_event	panelId boardNo machineName processedTS location
smd_location	p01 b01 SMD Machine 1 24-04-2020 mes01
smd_panel	
smd_components	smd_components
	panelId boardNo headId nozzleId turnNo pickSeqNo placeSeqNo
AOI Tables	p01 b01 h01 n01 2 1 3
aoi_event	aoi_failures
aoi_location	panelId boardNo refDesignator windowNo cPinNo failureCode
aoi_panel	p01 b01 rd01 w01 pn01 1
aoi_failures	

Рис. 3. Реляционные таблицы, используемые устройствами SMD и AOI [4].

На рисунке 3 показаны фрагменты трех таблиц. Таблицы smd-panel smd_components создаются SMD-устройством, а таблица aoi_failures - AOI устройством. Можно видеть, что SMD-устройство обрабатывает информацию о панели (столбец panelId), монтируемых на этой панели компонентах (столбец boardNo), и о моменте времени, в который данная панель была обработана (столбец processed TS), в то время как устройство AOI генерирует информацию о том, была ли панель списана или нет (столбец failureCode). Значение этой информации закодировано в числах: значение '1' соответствует случаю, когда панель списывается, 0 - в противном случае. Кроме того, устройство AOI содержит информацию о различных типах сбоев, например, число 2 соответствует ситуации «ложный вызов» (FalseCall), а число 21 соответствует случаю «Неправильно размещенный компонент» (MisplacedComponent). Смысл этой информации, закодированной в числах, имеется только во внутренних документах предприятия и в головах его инженеров.

Теперь предположим, что мы хотим получить ответ на следующий запрос. «Для панелей, обработанных в заданный период времени, получить количество отказов, связанных со списанными платами и сгруппированных по типам отказов». Чтобы получить эту информацию, необходимо семантически интегрировать данные, которые находятся на устройстве поверхностного монтажа SMD и на устройстве автоматического оптического контроля AOI.

На рисунке 4 показана часть онтологии SMT, которая используется в качестве модели предметной области для интеграции семантических данных и доступа к предметной области. Онтология разделена на три модуля:

- Онтология продуктов SMT (product ontology, префикс *psmt:*), описывающая продукты SMT;
- Онтология сбоев SMT (failure ontology, префикс *fsmt:*), моделирующая отказы, которые могут произойти во время процесса SMT; и
- Онтология устройств SMT (machine ontology, префикс *msmt:*), моделирующая устройства SMT. Общая онтология SMT включает 76 классов, 30 объектных свойств и 57 свойств типов данных.

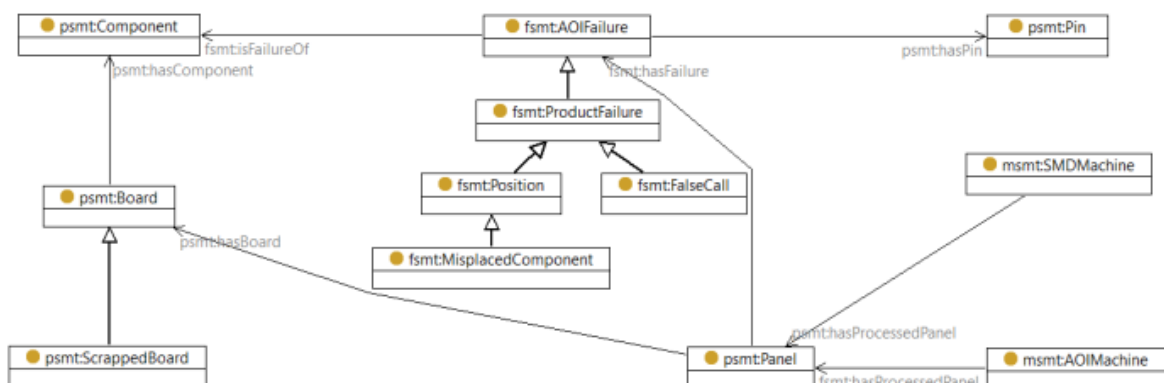


Рис. 4. Онтология SMT включает в качестве подмодулей онтологию продуктов, онтологию устройств и онтологию сбоев [4].

Как правило, способ кодирования отказов различается на разных предприятиях внутри одной и той же организации, что затрудняет идентификацию одних и тех же отказов, описанных в разных подсистемах. Чтобы решить эту проблему, предлагается вместо численных значений, принимающих различные значения в различных реляционных таблицах, использовать соответствующие классы в Онтологии сбоев. Например, для представления такого типа отказа как «неправильное размещение компонента» создается класс *fsmt:MisplacedComponent*. Этот класс является подклассом класса *fsmt:Position*, а также класса *fsmt:ProductFailure*, которые семантически группируют все отказы, попадающие в эти категории. Точно так же создается класс *psmt:ScrappedBoard* для семантического представления всех плат, которые списываются. Этот класс представлен как подкласс класса *psmt:Board*.

На рисунке 5 показаны два примера отображений. Одно из отображений называется «Ложный вызов» ("FalseCall"), а второе – «Неправильно размещенная компонента» ("MisplacedComponent"). Оба отображения определяют, какие триплеты графа знаний должны быть сгенерированы в зависимости от численного значения столбца *failureCode* в таблице *aoi_failures*.

```

mappingId FalseCall
target      fsmt:AOIFailure/{panelId}/{boardNo}/{refDesignator}/{windowNo}/{cPinNo}
            rdf:type fsmt:FalseCall ;
            fsmt:failureType "FalseCall" .
source      SELECT panelId, boardNo, refDesignator, windowNo, cPinNo
            FROM aoi_failures WHERE failureCode = 2

mappingId MisplacedComponent
target      fsmt:AOIFailure/{panelId}/{boardNo}/{refDesignator}/{windowNo}/{cPinNo}
            rdf:type fsmt:MisplacedComponent ;
            fsmt:failureType "MisplacedComponent" .
source      SELECT panelId, boardNo, refDesignator, windowNo, cPinNo
            FROM aoi_failures WHERE failureCode = 21

```

Рис. 5. Отображения FalseCall и MisplacedComponent [4].

Если значение failureCode равно 2, то в графе знаний должны быть сгенерированы две триплеты. Обе триплеты будут иметь в качестве субъекта объект, имеющий URI fsmt:AOIFailure/{panelId}/{boardNo}/{refDesignator}/{windowNo}/{PinNo}, где на месте местодержателей будут указаны конкретные идентификаторы панели, компоненты, контакта и т.д., например fsmt:AOIFailure/p01/b01/rd01/wx1/p01. Сгенерированные по этому отображению триплеты будут иметь вид:

fsmt:AOIFailure/p01/b01/rd01/wx1/p01 rdf:type fsmt:FalseCall.

fsmt:AOIFailure/p01/b01/rd01/wx1/p01 fsmt:failureType "FalseCall".

Аналогичным образом, отображение MisplacedComponent указывает, какие триплеты графа знаний должны быть сгенерированы, в случае, когда таблице aoi_failures код сбоя в столбце failureCode равен 21. Этому случаю в сгенерированном графе знаний также будет соответствовать две триплеты.

К результирующему графу знаний можно писать запросы SPARQL в терминах онтологии, описывающей предметную область, и пользователю не требуется понимание конкретных источников данных, знание о взаимосвязях между этими источниками или способе кодирования данных в отдельных источниках.

В работе [8] описано применение подхода ВГЗ в норвежской многонациональной нефтегазовой компании Equinor (ранее Statoil ASA). Одна из обычных задач геологов Equinor - найти новые пригодные для эксплуатации скопления нефти или газа в заданных областях, своевременно анализируя данные об этих областях. Однако сбор необходимых данных не является тривиальной задачей, так как данные хранятся в нескольких сложных и больших источниках данных, таких как EPDS, Recall, CoreDB, GeoChemDB, OpenWorks, Compass и NPD FactPages. Построение правильных запросов ко всем этим источникам невозможно для геологов Equinor, поэтому они должны сообщать свои потребности в информации ИТ-специалистам, которые затем превращают их в запросы SQL. Это резко влияет на

эффективность поиска правильных данных для поддержки принятия решений. Эта проблема была решена путем создания виртуального графа знаний, и создания каталога запросов SPARQL в терминах предметной области.

Еще одно применение подхода виртуальных графов знаний, реализованное в фирме Siemens, описано в работе [9]. Подразделение Siemens Energy управляет несколькими сервисными центрами, которые удаленно контролируют и выполняют диагностику нескольких тысяч устройств, таких как газовые и паровые турбины, генераторы и компрессоры, установленные на электростанциях. Для выполнения диагностики доступ к данным и интеграция как статических данных (например, конфигурация и структура турбин), так и динамических данных (например, данные от датчиков) особенно важны, но очень сложны. Опять же использование виртуальных графов знаний позволяет справиться с этой проблемой.

В работе [10] описано применение ВГЗ в аэрокосмической промышленности для извлечения и валидации информации о результатах тестирования электронных устройств, запускаемых в космос.

3. Исполняемые графы знаний для контроля качества точечной контактной сварки в BOSH

Точечная контактная сварка – это пример полностью автоматизированного и эффективного производственного процесса, широко применяемого в самолето-, судо- и автомобилестроительной промышленности, в сельскохозяйственном машиностроении и других отраслях промышленности. Для осуществления сварки два колпачка электродов сварочной горелки сжимают два или три металлических листа между электродами и пропускают ток высокого напряжения. Материал на небольшой площади между электродами плавится и образует точечный сварочный шов, соединяющий рабочие листы, известный как *литое ядро сварной точки* [11]. Качество операции сварки количественно оценивается как диаметр литого ядра сварной точки, как предписано в международных стандартах. Для точного определения диаметра литого ядра сварной точки в машиностроении обычной практикой является разрезание сваренного кузова автомобиля и измерение размера этого ядра, что разрушает сваренные автомобили и является чрезвычайно дорогой операцией. В настоящее время Bosch [12] разрабатывает методы оценки качества сварки, основанные на машинном обучении, чтобы уменьшить потребность в разрушенных кузовах автомобилей.

Сварка — это процесс, требующий больших объёмов данных. Каждая сварочная машина производит одну сварную точку за несколько секунд или минут, а кузов автомобиля может иметь до 6000 сварных точек. Для каждой точки генерируется несколько сотен характеристик, включая состояние сварки, показатели качества и показания датчиков, которые измеряют важные физические характеристики каждую миллисекунду, такие как ток, сопротивление, мощность.

Для оценки качества сварки применяется три важные направления машинного обучения такие как визуальная аналитика, статистическая аналитика и аналитика машинного обучения, основанная, например, на нейронных сетях.

Более того, у Bosch много источников данных для схожих производственных процессов, поэтому возможность повторного использования решений машинного обучения очень желательна. Эти решения можно перенести на аналогичные данные или вопросы машинного обучения.

В работе [12] предлагается закодировать решения машинного обучения при помощи графа знаний таким образом, чтобы графы знаний помогали в описании знаний о машинном обучении и решениях стандартизированным и прозрачным способом с помощью системы на основе графического интерфейса и визуализации графа знаний. Данный подход называется *исполняемые графы знаний*, потому что такие графы знаний могут быть преобразованы в скрипты машинного обучения, которые можно модифицировать и повторно использовать для решения похожих проблем.

Фреймворк для исполняемого графа знаний представляет возможность создания различных решений на базе методов машинного обучения (конвейеров) для решения проблем машинного обучения. Фреймворк поддерживает трансляцию исполняемых графов знаний в исполняемые скрипты машинного обучения.

Фреймворк исполняемого графа знаний определяет связи между такими понятиями как *Данные* (Data), *Метод* (Method) и *Задание* (Task). *Данные* — это множество элементов информации, организованные в разные структуры. *Метод* — это функция, реализованная в форме скрипта, написанного на заданном языке. *Метод* получает *Данные*, удовлетворяющие определенным ограничениям, и выдает в качестве результата другие *Данные*. *Задание* — это способ вызова *Метода*, подавая ему *Данные*, удовлетворяющие определенным ограничениям. Некоторые задания имеют метод для выполнения этого задания, а некоторые задания не могут быть выполнены при помощи одного метода и должны быть развернуты в последовательность заданий, где каждое задание является частью более сложного задания в

этом случае можно говорить о *Конвейере* (Pipeline) заданий. Связи между этими понятиями показаны на рисунке 6.

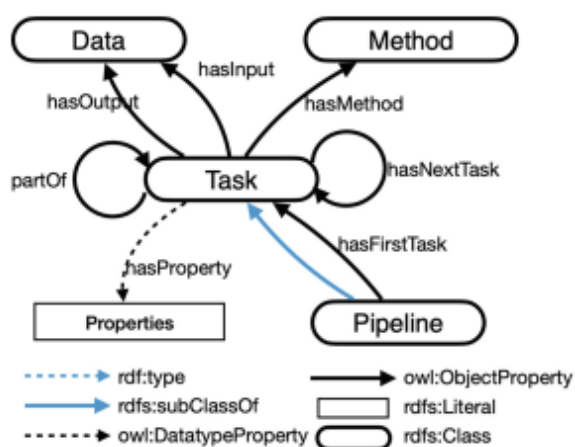


Рис. 6. Связи между основными классами фреймворка исполняемого графа знаний [12].

В зависимости от типа аналитического приложения могут вызываться различные методы, описанные в специализированных онтологиях, и генерироваться различные конвейеры заданий. Общий вид исполняемого графа знаний показан на рисунке 7.

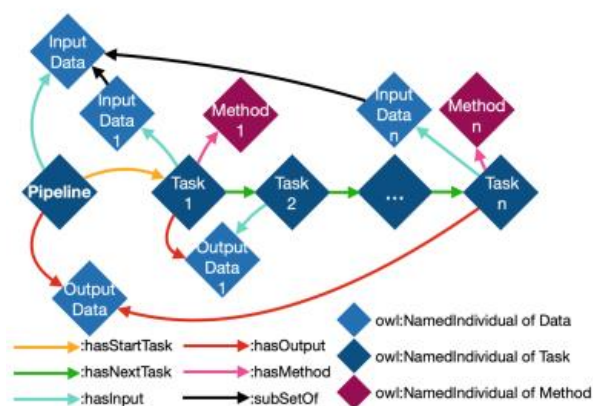


Рис. 7. Общая схема исполняемого графа знаний [12].

Архитектура системы для работы с исполняемыми графами знаний показана на рисунке 8. Система состоит из пяти слоев: слой данных (не граф знаний), уровень приложений, уровень базы данных графа знаний, уровень семантических модулей и уровень семантических артефактов. Модули одного уровня покрашены в один цвет. Например, самый нижний уровень данных содержит пять модулей, покрашенных в серый цвет. На этом уровне находятся входные и выходные данные системы. На вход Модуля Интеграции данных подаются необработанные данные сварки (нижний левый угол). Эти данные преобразуются Модулем Интеграции данных (с помощью онтологий предметной области) в графы знаний

машинного обучения для сварочных аппаратов, которые представляет собой тип графа знаний о данных сварки с некоторой аннотацией машинного обучения. Эти графы знаний используются четырьмя типами аналитических приложений на уровне приложений (показаны бежевым цветом).

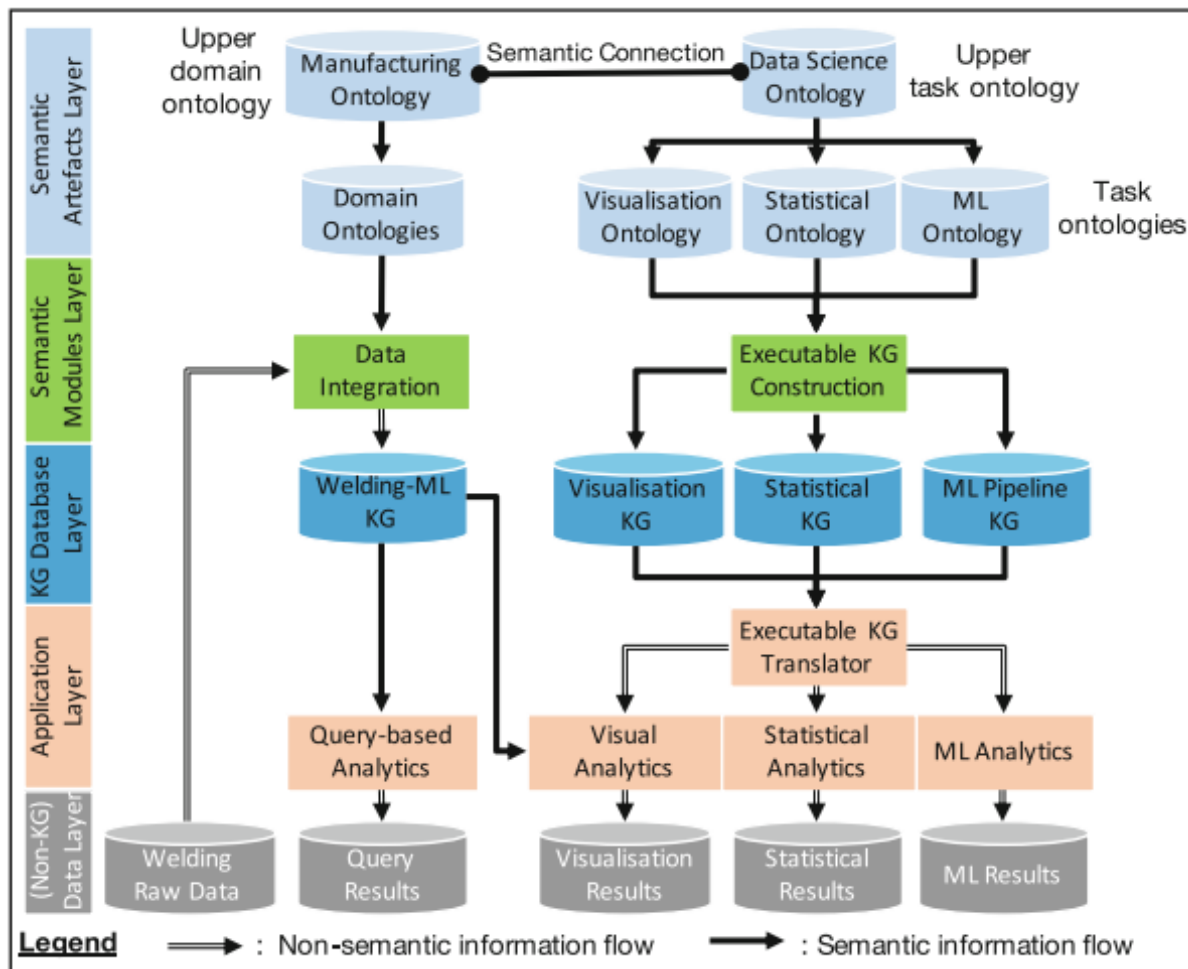


Рис. 8. Архитектура системы исполняемых графов знаний [12].

На самом верхнем уровне Семантических артефактов находятся две онтологии верхнего уровня. Это Онтология Производства (Manufacturing ontology) и Онтология интеллектуального анализа данных (data science ontology, O_{ds}). Специализацией онтологии производства являются онтологии предметных областей, такие как различные онтологии сварки, например, онтология точечной сварки. Эти онтологии создаются на основе онтологии производства верхнего уровня [13].

Онтология производства семантически связана с онтологией верхнего уровня Онтологии интеллектуального анализа данных (data science ontology, O_{ds}), таким образом, что атрибутные свойства (DataTypeProperty) онтологии производства аннотированы некоторыми классами онтологии интеллектуального анализа данных O_{ds} .

На рисунке 9 показаны Онтология интеллектуального анализа данных (Ods, Data Science Ontology) и три ее специализации Онтология статистики (StatisticalOntology), онтология Визуализации (Visualization Ontology) и Онтология машинного обучения (ML-Ontology). Три онтологии заданий StatisticalOntology, Visualization Ontology и ML-Ontology являются специализацией онтологии Ods, поскольку все классы этих онтологий являются подклассами онтологии Ods, а все свойства являются подсвойствами Ods.

Можно видеть, что онтология верхнего уровня содержит такие классы как Data (Данные), Method (Метод) и Task (Задание), в то время как онтологии-специализации содержат методы и задания, специфические для конкретного приложения. Например, методы, необходимые для выполнения статистической аналитики описаны в онтологии статистического анализа (Statistical_Ontology, Ostats), методы, необходимые для выполнения визуальной аналитики, описаны в онтологии визуализации (Visualization_Ontology, Ovisu), а методы, необходимые для выполнения аналитики, Машинного обучения, описаны в онтологии машинного обучения (ML_Ontology, Oml).

Так, например, в онтологии визуализации Visualisation_Ontology описаны специфические для визуализации методы такие как LineplotMethod (метод построения линейчатых диаграмм), ScatterplotMethod (метод построения диаграмм рассеяния BarplotMethod (метод построения гистограмм). Все эти методы являются экземплярами класса VisualMethod, являющегося подклассом класса Method.

Аналогичным образом методы, специфические для статистической аналитики, такие как MeanCalculationMethod и StandardDeviationMethod являются подклассами класса StatisticalMethod онтологии Statistical-Ontology.

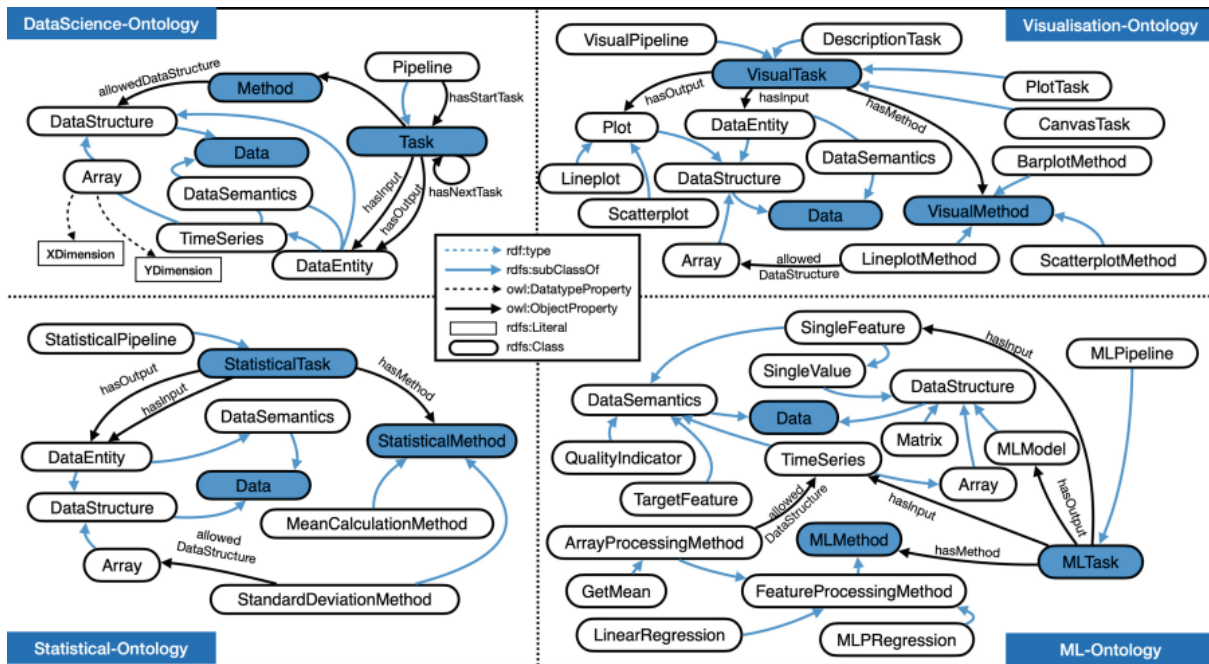


Рис. 9. Онтологии задач интеллектуального анализа данных [12].

Эти специализированные онтологии служат схемами для Модуля Построения исполняемого графа знаний, который кодирует различные конвейеры исполняемых графов знаний. Все исполняемые графы знаний принадлежат классу Pipeline (Конвейер) и соответствуют конкретным решаемым проблемам. На рисунке 10 показан один из возможных вариантов графа знаний для визуализации.

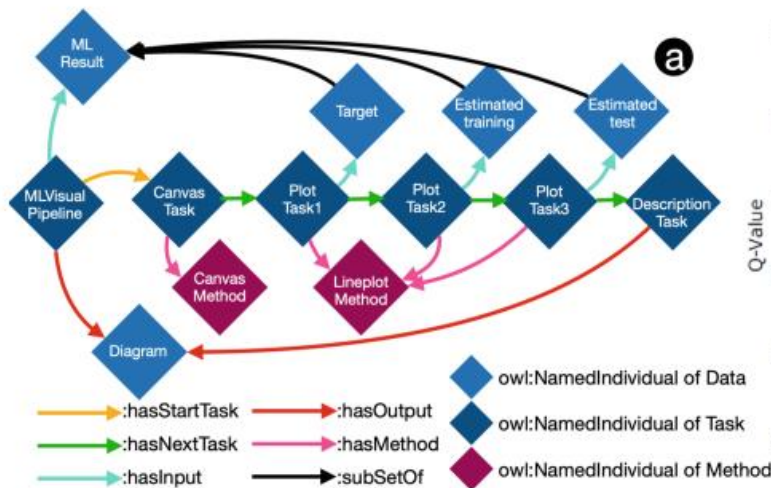


Рис. 10. Один из исполняемых графов знаний для визуализации [12].

Эти исполняемые графы могут быть переведены модулем Executable Knowledge Graph Translator в исполняемые скрипты, создавая тем самым приложения визуальной либо статистической аналитики или же приложения аналитики машинного обучения.

Возможно три способа построения новых исполняемых графов знаний: Создание графа знаний с нуля на основе библиотеки имеющихся модулей, Модификация ГЗ и интеграция

нескольких ГЗ. Например, можно интегрировать конвейер статистического анализа, который идентифицирует выбросы, с конвейером визуализации для того, чтобы визуализировать результаты идентификации выбросов.

4. Динамические графы знаний для цифровых двойников

Одной из областей применения семантических моделей являются цифровые двойники. Цифровой двойник (ЦД) — это цифровое представление системы IoT, способное непрерывно обучаться в течение всего жизненного цикла системы и прогнозировать поведение системы IoT. Цифровые двойники предназначены для постоянного использования на протяжении всего жизненного цикла системы - от предоставления рекомендаций при создании системы, до автоматизации ее производства и оптимизации ее работы путем диагностики аномалий или улучшения контроля и прогнозирования. В последние несколько лет наблюдается тенденция к объединению ЦД с передовыми технологиями семантического моделирования и методами глубокого обучения для обеспечения ЦД когнитивными возможностями.

Термин «когнитивные цифровые двойники» по отношению к промышленным приложениям впервые был предложен Адлом [14] в 2016 году, хотя и до этого в литературе рассматривались возможности улучшения когнитивных способностей цифровых двойников с помощью семантических технологий.

В ходе презентации на отраслевом семинаре в 2016 году Ахмед Эль Адл рассказал о когнитивной эволюции технологий Интернета вещей и предложил концепцию когнитивного цифрового двойника, а также её характеристики и категории. Она была определена как «цифровое представление, дополнение и интеллектуальный компаньон своего физического двойника в целом, включая его подсистемы на всех этапах жизненного цикла и эволюции».

Хотя в настоящее время нет широко распространенного консенсуса по определению КЦД, можно выделить некоторые их общие элементы и характеристики [15].

(1) Основаны на ЦД: КЦД является расширенной или дополненной версией ЦД. Он содержит как минимум три основных элемента ЦТ, включая физическую сущность (системы, подсистемы, компоненты и т. д.), цифровое (или виртуальное) представление, связи между виртуальным и физическим пространствами. Основное отличие заключается в том, что КЦД обычно содержит несколько моделей ЦД с определениями топологии и семантики. В частности, для сложной промышленной системы КЦД должен включать цифровые модели подсистем и компонентов, каждая из которых имеет разный статус на протяжении всего жизненного цикла. Для более сложных бизнес-сценариев ожидается подключение большого количества связанных цифровых моделей.

(2) Когнитивные способности: КЦД должен обладать определенными когнитивными способностями, то есть он позволяет выполнять интеллектуальные действия, подобные человеческим, такие как внимание, восприятие, понимание, память, рассуждение, прогнозирование, принятие решений, решение проблем, реакция и т. д.

(3) Управление полным жизненным циклом: КЦД должен состоять из цифровых моделей, охватывающих различные этапы всего жизненного цикла системы, включая начало жизненного цикла (например, проектирование, сборка, тестирование), середину жизненного цикла (например, эксплуатация, использование, техническое обслуживание) и конец жизненного цикла (например, разборка, переработка, восстановление). КЦД также должен быть способен интегрировать и анализировать все доступные данные, информацию и знания из различных этапов жизненного цикла.

(4) Способность к автономности: КЦД должен выполнять автономные действия без помощи человека или минимального уровня человеческого вмешательства. Эта способность частично перекрывается и подкрепляется когнитивными способностями КЦД. Например, основываясь на результатах восприятия и прогнозирования, КЦД может автономно принимать решения и адаптивно реагировать на проектирование, производство или эксплуатацию.

(5) Непрерывное развитие: КЦД должен иметь возможность развиваться вместе с реальной системой на протяжении всего жизненного цикла. Существует три уровня развития. Во-первых, для одной цифровой модели она обновляется в соответствии с изменением соответствующих данных, информации и знаний, полученных из реальной системы; во-вторых, благодаря взаимодействию между различными цифровыми моделями, содержащимися в одной и той же фазе жизненного цикла, каждая модель динамически развивается в соответствии с влиянием других моделей; в-третьих, благодаря обратной связи от других фаз жизненного цикла.

Одной из интересных моделей КЦД является World Avatar [16], использующий в качестве основного инструмента динамический граф знаний, который задуман как универсальный и всеобъемлющий. Универсальность Мирового Аватара основана на значительном количестве различных модульных онтологий предметных областей и экосистеме автономных агентов, способных модифицировать граф знаний.

Схематическое устройство World Avatar показано на рисунке 11. В основании конструкции находятся расширяемое множество модульных онтологий предметных областей. Это множество прямоугольников голубого цвета. Реальные экземпляры графа знаний, изображенные при помощи эллипсов, описываются при помощи этих онтологий

(зеленый слой). В зеленом слое также имеются красные эллипсы, изображающие экземпляры агентов. Агенты имеют разные типы (атомарный, составной и композитный) и являются частью графа знаний, управляемой Онтологией Агентов (OntoAgent) [17]. Агенты действуют автономно и непрерывно на графе знаний, постоянно обновляя его и, таким образом, заставляя его развиваться во времени.

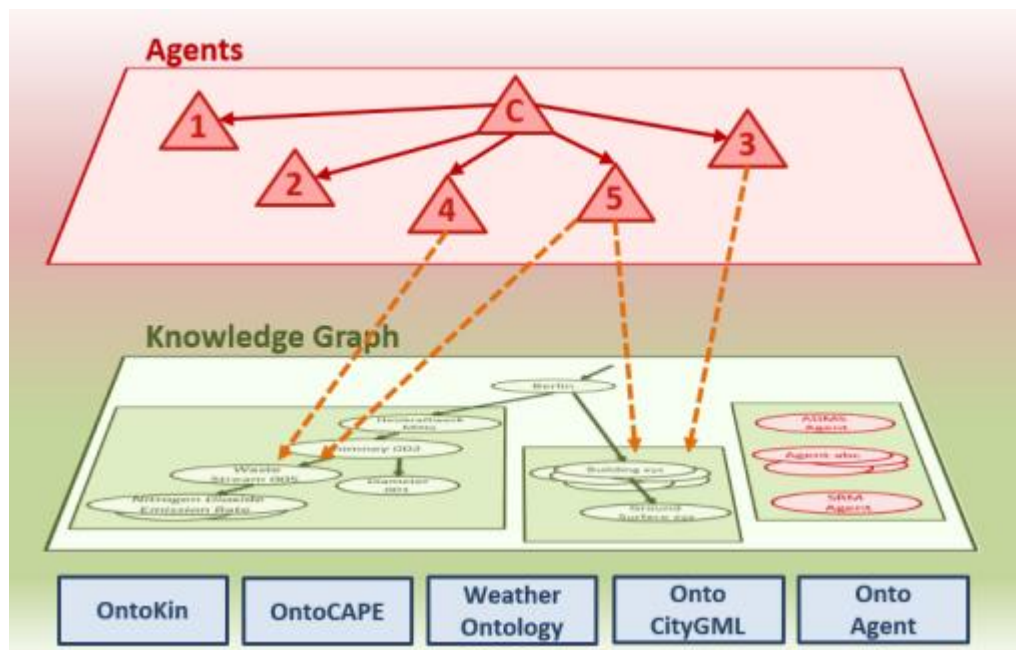


Рис. 11: Иллюстрация основных принципов Мирового Аватара (World Avatar) на основе динамического графа знаний [16].

В верхней части схемы World Avatar показаны активные агенты (красные треугольники), действующие над графом знаний и взаимодействующие друг с другом.

Понятие «Агент» может соответствовать программному обеспечению, методам, приложениям, сервисам и т. д., которые используют семантические веб-технологии и работают с графом знаний для чтения/записи, оценивания, моделирования, оптимизирования и/или запросов и т. д. для достижения конкретных целей.

Кроме того, чтобы облегчить использование агентов и упростить идентификацию агента, подходящего для конкретной задачи в среде с большим количеством агентов (где доступно множество сервисов), был создан рынок агентов на основе технологии блокчейн и смарт-контрактов [18].

World Avatar позволяет подключение к различным подграфам облака открытых связанных данных (LOD, lod-cloud.net), и, таким образом, может использовать богатство всех данных, доступных в Интернете. Благодаря своей универсальной конструкции, основанной на онтологиях и автономных агентах, World Avatar улучшает совместимость между

разнородными форматами данных, а также программным обеспечением и, таким образом, позволяет использовать междоменные приложения в более широком контексте.

Текущее онтологическое покрытие World Avatar включает:

- онтологию механизмов химических кинетических реакций OntoKin [19];
- технологических процессов OntoCAPE[20];
- погоды Weather Ontology [21];
- онтологию 3D моделей городов и ландшафтов OntoCityGML[22];
- систем электроснабжения OntoPowerSys, [23].

В области химии используются такие онтологии как

- онтология для квантовой химии OntoCompChem, [24];
- видов химических веществ OntoSpecies,[25];
- экспериментов по горению OntoChemExp [26];
- Онтологию выделяемых при сгорании веществ в зависимости от типа двигателя корабля OntoShip [27].

Далее будет показано два варианта использования принципиально разных аспектов фреймворка World Avatar – управление и проектирование.

Первый вариант демонстрирует, как цифровые двойники на основе динамического графа знаний могут сократить затраты и потребляемую энергию посредством интеллектуальных стратегий управления.

Второй вариант показывает, как можно использовать структуру параллельного мира для создания «живого» цифрового мира, т.е. сценария, который позволяет исследовать различные технологические альтернативы и эффект политики по переходу к энергосберегающим технологиям.

4.1 Кросс-доменное вычисление качества воздуха

В Сингапуре расположен один из самых загруженных портов в мире. Естественно, возникает вопрос о том, как выбросы углеводородов из такого трудно поддающегося сокращению сектора, такого как судоходство, влияют на такие факторы как качество воздуха в разных районах Сингапура.

Рисунок 12 иллюстрирует, каким образом интероперабельность World Avatar позволяет оценивать в режиме реального времени вклад выбросов от судоходства в качество воздуха в Сингапуре. Он демонстрирует, чего можно достичь с точки зрения совместимости, как между моделями и данными из разных предметных областей (погода, расположение зданий в

городе, корабли в гавани и топливо, которое они используют, выбросы вредных веществ и их рассеяние в атмосфере) и между различными масштабами размеров (от масштаба атомной длины в расчетах вычислительной химии до километровых масштабов моделирования атмосферной дисперсии).

Агенты, работающие над World Avatar, обновляют в графе знаний в режиме реального времени информацию о погоде и о кораблях, находящихся в окрестностях Сингапура. Агент по выбросам способен использовать информацию о судах для оценки выбросов несгоревших углеводородов, CO, NO₂, NO_x, O₃, SO₂, PM_{2,5} и PM₁₀ с каждого корабля. Агент Атмосферной дисперсии может использовать информацию о погоде, выбросах каждого корабля и расположении зданий Сингапура для моделирования рассеивания выбросов.

Агенты виртуальных датчиков сообщают результирующие оценки качества воздуха в разных локациях.

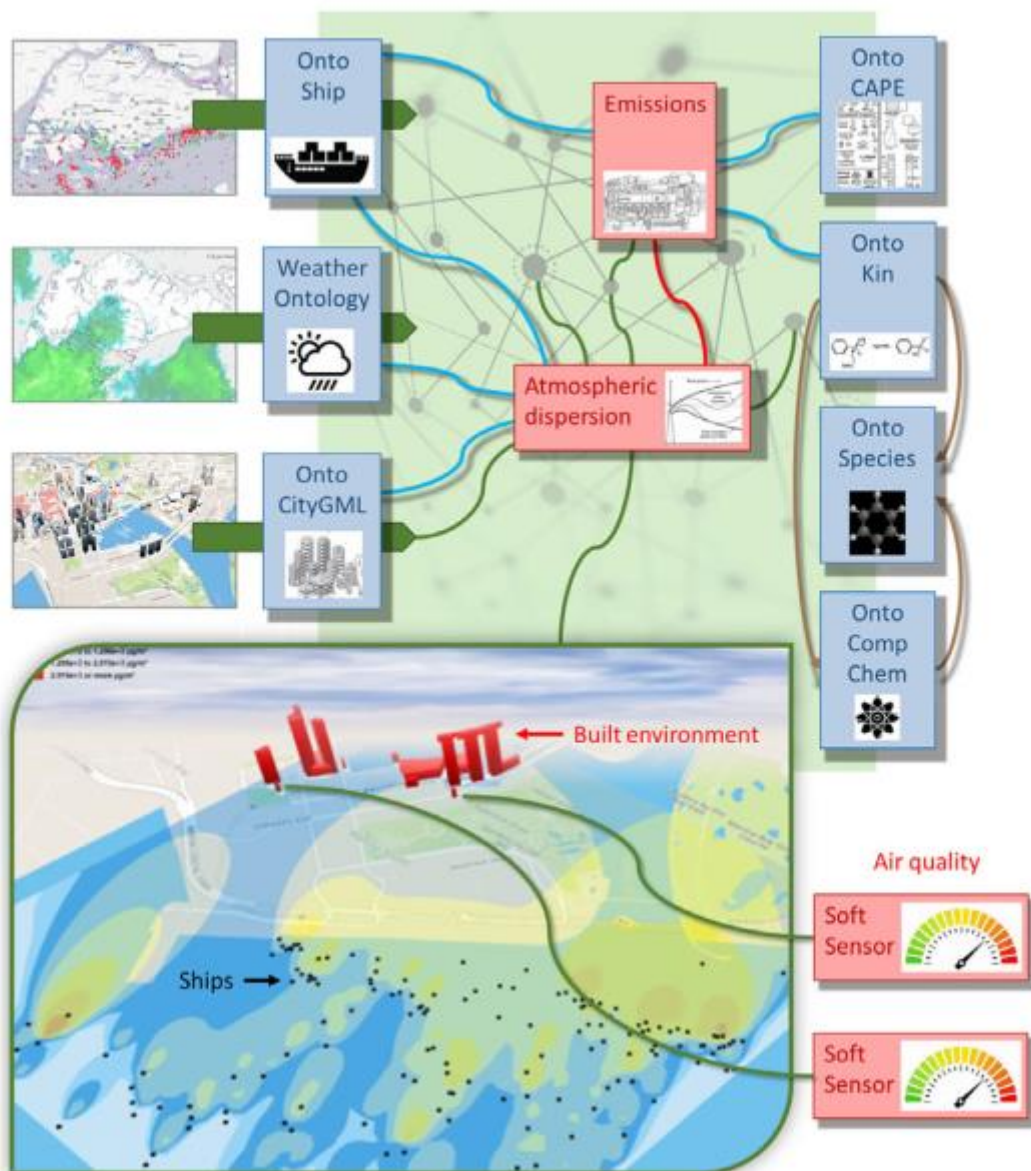


Рис. 12. Междоменная оценка в режиме реального времени вклада выбросов от судоходства в качество воздуха в Сингапуре [25].

4.2. Интеллектуальные стратегии проектирования систем электроснабжения

Этот вариант использования иллюстрирует, как цифровое дублирование и анализ сценариев «что, если» с использованием динамического графа знаний может помочь лицам, принимающим решения, понять взаимное влияние различных вариантов использования источников энергии и инструментов политики.

Возможности параллельных миров World Avatar поддерживают принятие решений в сложной среде, позволяя исследовать различные сценарии и связанные с ними результаты.

Параллельные миры используют структуру динамического графа знаний для использования агентов сценариев для группировки новых экземпляров любых сущностей, которые изменяются в сценарии, и сохраняют их в специфичной для сценария части графа знаний. Доступ, запросы и обновления экземпляров, специфичных для сценария, осуществляются через агентов сценария.

Специфическую для сценария информацию в параллельном мире можно рассматривать как наложение на базовый мир, поэтому неизменные сущности остаются связанными с базовым миром, и любые изменения в базовом мире отражаются в сценарии. И наоборот, любые изменения в параллельном мире остаются изолированными в рамках специфической для конкретного сценария части графа знаний и поэтому не мешают базовому миру. Основанные на идее, очень похожей на системы контроля версий, широко используемые среди разработчиков программного обеспечения, контейнеры параллельного мира хранят различия с базовым миром в именованных графах, где сценарий предоставляет контекст. Технические подробности можно найти в [28].

Рисунок 13 иллюстрирует использование концепции параллельного мира для исследования влияния введения налога на выбросы углерода в результате процессов производства электроэнергии, чтобы мотивировать переход от ископаемого топлива на экологически чистые энергетические технологии. Рассматриваются следующие вопросы:

- Какая сумма налога на выбросы углерода необходима для того, чтобы сделать переход на малые модульные ядерные реакторы (Small Modular Reactor, SMR) выгодным для заданного набора условий (например, срок службы проекта, нормы амортизации, профили электрической нагрузки и характеристики генератора)?
- Какой завод(ы) следует заменить, и где следует расположить новые SMR(ы)?



Рис. 13. Концепция параллельного мира для анализа сценариев «что, если» [28].

В верхней части рисунка показана электрическая сеть из реального мира (слева) и оптимизированная сеть, облагаемая налогом на выбросы углерода (справа). Сеть из реального мира описана в базовом мире. Изменения в сети в параллельном мире описаны в специфичной для сценария части графа знаний. Розовые треугольники обозначают генераторы на природном газе, которые присутствуют как в базовом мире, так и в параллельном мире. Синий квадрат обозначает масляный генератор, который присутствует только в базовом мире. Символ радиации обозначает небольшой модульный ядерный реактор (SMR), который присутствует только в параллельном мире.

Параллельный мир показывает, что нефтяные генераторы будут заменяться на SMR, как только налог на выбросы углерода будет увеличен с пяти до ста семидесяти сингапурских долларов за тонну. Типы имеющихся генераторов и соответствующие оценки выбросов CO₂ автоматически обновляются в параллельном мире, чтобы отразить эти изменения, и Агент оптимальной мощности потока (optimal power flow, OPF) вызывается для минимизации общих эксплуатационных затрат сценария в параллельном мире [28, 27].

В этом примере проблема, решаемая параллельным миром, достаточно проста, и ее можно было сформулировать как классическую задачу оптимизации с четко определенной целевой функцией. Однако многие сценарии будут слишком сложными для того, чтобы это

произошло. Как решить эту проблему, обсуждается в контексте цифрового двойника Великобритании на основе графа знаний [30].

The World Avatar можно рассматривать как пример новой парадигмы для информационных систем GeoWeb, выходящих за рамки Web 2.0. Основанный на базе Semantic 3D City Database [31], он переносит существующие стандарты ГИС в новую графовую базу данных и использует преимущества концепции открытого мира (Open World Assumption), которая отсутствует в аналогичных реляционных геопространственных базах данных. Сочетание её с системой интеллектуальных автономных агентов, основанных на когнитивной архитектуре, расширяет и масштабирует существующие инструменты преобразования геопространственных данных. Её агенты, как оказалось, способны автоматически создавать семантическую модель Берлина, состоящую в общей сложности из 419 909 661 утверждений типа «субъект–предикат–объект». Помимо традиционно трудоёмкого и подверженного ошибкам процесса создания модели, агенты также автономно создали представление модели, пригодное для взаимодействия с ней через веб-интерфейсы. Более того, агенты продемонстрировали способность отслеживать взаимодействие пользователя с моделью в сети, создавать новые знания и также автономно отображать их пользователю [31]. Данные работы положили начало большому множеству исследований по цифровым двойникам для моделирования умных городов, интегрирующим информацию о потреблении энергии, трафике, землепользовании, и др. [32, 33].

5. Взаимное обогащение ГЗ и больших языковых моделей

Появление Больших Языковых Моделей таких как Gemini от Google и серии GPT от OpenAI привело к повышению качества работы многих приложений, таких как автоматический перевод между разными языками, генерация контента и виртуальные помощники. Большие Языковые Модели используются в чатботах и сервисах клиентской поддержки, они прекрасно справляются с резюмированием текстов и анализом тональности документов для изучения общественного мнения. К сожалению, знание языковых моделей «замораживается» в их параметрах во время обучения, что приводит к определенным ограничениям. Прежде всего они способны генерировать неточную или бессмысленную информацию (галлюцинации), испытывают недостаток специфических знаний, а также нехватку знаний, появившихся после обучения, более того, не всегда их ответы возможно интерпретировать. Поэтому в последние несколько лет активно исследуются возможности повышения качества работы Больших Языковых моделей при помощи Графов Знаний. Недавние обзоры [34, 35] рассматривают возможность включения знаний из ГЗ в Большие

Языковые Модели. Наиболее простой способ включения знаний из ГЗ состоит генерации текстовых промптов на основе ГЗ, как это показано на рисунке 14 (вариант 1). Например, в работе [36] описан метод KAPING (Knowledge-Augmented language model PromptING, подсказки, расширенные знаниями для языковой модели), который извлекал факты из ГЗ и использовал их для создания подсказок LLM, чтобы получить быстрый ответ на вопросы.

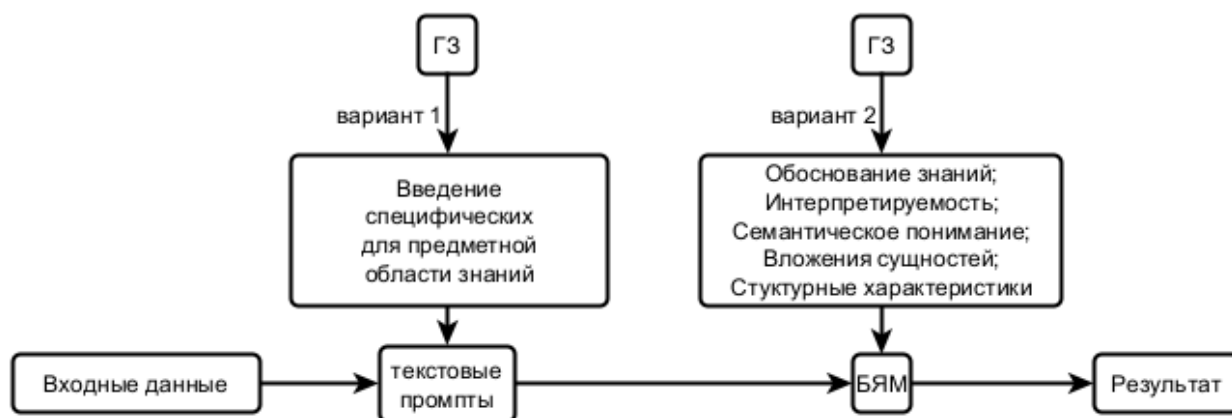


Рис. 14 Варианты введения знаний из графов знаний в большие языковые модели.

Возможны и другие варианты. Например, Knowledge Solver [37] вместо того, чтобы включать факты из ГЗ в промпты для БЯМ, обучает БЯМ многошаговому обходу ГЗ для поиска ответа на вопрос. Таким образом, ГЗ могут предоставлять факты, на основе которых БЯМ могут рассуждать, обосновывая эти факты. С другой стороны, ГЗ могут внести больший вклад в БЯМ, чем просто предоставляя факты для обоснования знаний. Для задачи ответа на вопросы QA-GNN (вопросно-ответная графовая нейронная сеть) [38] выполняла совместный вывод над векторными представлениями (embeddings) контекста вопроса, полученным при помощи БЯМ, и ГЗ для объединения двух представлений. Для лучшей интерпретируемости модели использовалась графовая нейронная сеть (GNN), вычислявшая веса между вершинами графа, предоставляя путь рассуждений, который модель прошла в ГЗ, чтобы получить ответ. Другим примером является LMExplainer [39], который использовал ГЗ и графовую нейронную сеть с механизмом внимания для понимания ключевых сигналов принятия решений в БЯМ, которые были преобразованы в объяснения на естественном языке для лучшей объяснимости. Таким образом, ГЗ также могут обеспечить лучшую интерпретируемость БЯМ и дать представление о процессах вывода БЯМ, что, в свою очередь, повышает доверие людей к БЯМ.

ГЗ также могут применяться для добавления семантического понимания или встраивания сущностей в БЯМ. Например, LUKE (Language Understanding with Knowledge-based Embeddings) [40], как расширение BERT, является механизмом само-внимания, осознающим

сущности, который может помочь модели обрабатывать слова и сущности в заданном тексте как независимые токены и выводить их контекстуализированные представления.

Что касается добавления семантического понимания, то недавняя методология, называемая «Правильное по правильным причинам» (**Right for Right Reasons, R3**) [41] для выполнения запросов на естественном языке к контенту графов знаний (KGQA) с использованием БЯМ, представляет проблему поиска ответов на такие запросы здравого смысла как древовидную структуру поиска для полного использования выявленных аксиом здравого смысла – ключевого свойства, делающего процедуру вывода проверяемой, так что семантическое понимание из ГЗ может быть добавлено в БЯМ. Эти более продвинутые возможности обогащения БЯМ при помощи ГЗ показаны на рисунке 14 (вариант 2).

Несмотря на вышеуказанные достоинства графов знаний, одной существенной проблемой остается весьма дорогостоящий, трудозатратный и затратный по времени процесс создания графов знаний. Он требует значительного количества промежуточных шагов, таких как извлечение сущностей и отношений в соответствии с заданной онтологией, разрешение ко-референций, слияние знаний. Более того, ГЗ специфичны для разных предметных областей, поэтому для разных приложений создаются разные графы знаний, а уже созданные графы знаний могут устаревать, если знание не обновляется.

Поэтому появление больших языковых моделей породило огромное количество исследований автоматической или полуавтоматической генерации графов знаний при помощи Больших Языковых Моделей. БЯМ обучаются на больших и разнообразных наборах данных и хранят эти знания неявно. В работе [42] описан полуавтоматический конвейер построения ГЗ при помощи ChatGPT-3.5, который побуждал эту модель генерировать высокоуровневые вопросы компетентности (competence questions) о данных. LLM было поручено извлекать сущности и отношения из этих вопросов для формирования онтологии, а затем отображать полученную информацию из документов в онтологию для построения ГЗ. Аналогичным примером является платформа AutoRD [43], недавно представленная для извлечения информации о редких заболеваниях и построения соответствующих графов знаний. Эта система может обрабатывать неструктурированный медицинский текст в качестве входных и выходных результатов извлечения, а также граф знаний, где БЯМ используется для извлечения сущностей и отношений из медицинских онтологий. Совсем недавно неконтролируемый фреймворк, называемый TKGCon (построение тематически-специфичного графа знаний) [44], использовал БЯМ для построения как онтологий, так и тематически-специфичных ГЗ, полагаясь на БЯМ для генерации и определения отношений между сущностями и построения ребер графа.

Эти методы свидетельствуют о том, что БЯМ способны на большее, чем извлечение знаний из неструктурированных данных. Они также могут обрабатывать и анализировать данные для построения и заполнения графов знаний. Кроме того, в работе [45] описаны другие методы, которые используют БЯМ для конкретных задач построения групп знаний, таких как сопоставление текста с онтологиями, извлечение сущностей и выравнивание онтологий. БЯМ также использовались для валидации групп знаний посредством проверки фактов и обнаружения несоответствий. Варианты введения информации из больших языковых моделей в графы знаний показаны на рисунке 15.

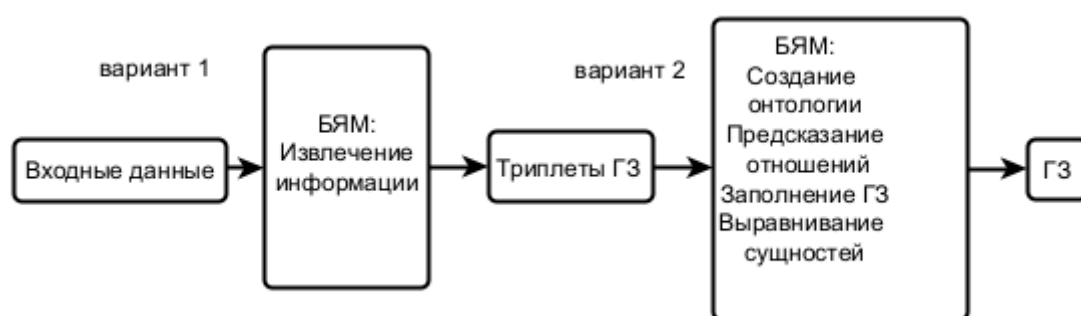


Рис. 15. Варианты введения информации из больших языковых моделей в графы знаний.

6. Заключение

В данной работе представлен обзор по современным приложениям различных вариаций графов знаний, таких как Виртуальные графы знаний, Исполняемые графы знаний и Динамические графы знаний. В работе сделан акцент именно на промышленных приложениях графов знаний, таких как цифровые двойники и их более современная версия когнитивные цифровые двойники. Также рассмотрено взаимное влияние графов знаний и больших языковых моделей, позволяющее создавать все более продвинутые приложения на основе ИИ. Приведенные примеры демонстрируют, что роль графов знаний при создании промышленный приложений в ближайшем будущем будет только возрастать.

Список литературы

1. Апанович З.В. Эволюция понятия и жизненного цикла графов знаний // Системная информатика. — 2020. — № 16. — С. 57-74
2. Berners-Lee T, Hendler J and Lassila O (2001) The semantic web// Scientific American 284(5), 28–37.

3. Klyne G and Carroll JJ (2004) Resource Description Framework (RDF): Concepts and Abstract Syntax. W3C Recommendation 10 February 2004. World Wide Web Consortium (W3C). [Электронный ресурс]. <http://www.w3.org/TR/2004/REC-rdf-concepts-20040210/> (дата обращения: 1.10.2025).
4. Kalaycı, E.G., Grangel Gonz'alez, I., L'osch, F., Xiao, G., Kharlamov, E., Calvanese, D., et al.: Semantic integration of Bosch manufacturing data using virtual knowledge graphs. // International Semantic Web Conference (ISWC). — 2020. — pp. 464–481.
5. Xiao, Guohui, Ding, Linfang, Cogrel, Benjamin, Calvanese, Diego, Virtual Knowledge Graphs: An Overview of Systems and Use Cases Data. //Intelligence 1(2019), 201-223.
6. Das, S., Sundara, S., Cyganiak, R.: R2RML: RDB to RDF mapping language. // W3C Recommendation, World Wide Web Consortium (September 2012)
7. Calvanese, D., et al.: Ontop: answering SPARQL queries over relational databases. //Semant. Web J. **8**(3), 471–487 (2017).
8. Kharlamov E., Hovland D., Skjæveland M.G., Bilidas D., Jiménez-Ruiz E., Xiao G., Soylu A., Lanti D., Rezk M., Zheleznyakov D., Giese M., Lie H., Ioannidis Y.E., Kotidis Y., Koubarakis M., Waaler A. Ontology based data access in Statoil. //Journal of Web Semantics. — 44:2017. — pp. 3–36.
9. Kharlamov E., Mailis T., Mehdi G., Neuenstadt C., Özçep Ö. L., Roshchin M., Solomakhina N., Soylu A., Svingos C., Brandt S., Giese M., Ioannidis Y.E., Lamparter S., Möller R., Kotidis Y., Waaler A. Semantic access to streaming and static data at Siemens// Journal of Web Semantics. — 44. — 2017. — pp. 54–74.
10. De Santis, A. *et al.* (2025). Integrating Large Language Models and Knowledge Graphs for Extraction and Validation of Textual Test Data. In: Demartini, G., *et al.* The Semantic Web – ISWC 2024. ISWC 2024. // Lecture Notes in Computer Science, vol 15233. Springer, Cham. https://doi.org/10.1007/978-3-031-77847-6_17.
11. [Электронный ресурс]. URL:https://commons.wikimedia.org/wiki/Category:Spot_welding. (дата обращения: 1.10.2025).
12. Zheng, Z. et al. Executable Knowledge Graphs for Machine Learning: A Bosch Case of Welding Monitoring // The Semantic Web – ISWC 2022. ISWC 2022. — Lecture Notes in Computer Science. — vol 13489. Springer, Cham. https://doi.org/10.1007/978-3-031-19433-7_45.
13. Svetashova, Y., et al.: Ontology-enhanced machine learning: a bosch use case of welding quality monitoring. // ISWC (2020).

14. Adl, Ahmed El. 2016. The Cognitive Digital Twins: Vision, Architecture Framework and Categories. Technical Report. [Электронный ресурс]. https://www.slideshare.net/slideshow/embed_code/key/JB60Xqcn. (дата обращения: 1.10.2025).
15. Xiaochen Zheng, Jinzhi Lu, Dimitris Kiritsis (2022) The emergence of cognitive digital twin: vision, challenges and opportunities// International Journal of Production Research, 60:24, 7610-7632, DOI: 10.1080/00207543.2021.2014591.
16. Mei Qi Lim, Xiaonan Wang, Oliver Inderwildi Markus Kraft August 3, 2021 The World Avatar – a world model for facilitating interoperability.
17. Zhou X, Eibeck A, Lim MQ, Krdzavac N and Kraft M (2019) An agent composition framework for the J-Park simulator – A knowledge graph for the process industry. // Computers and Chemical Engineering 130, 106577.
18. Zhou X, Lim MQ and Kraft M (2020a) A smart contract-based agent marketplace for the J-Park simulator – A knowledge graph for the process industry// Computers and Chemical Engineering 139, 106896.
19. Farazi F, Akroyd J, Mosbach S, Buerger P, Nurkowski D, Salamanca M and Kraft M (2020a) OntoKin: An ontology for chemical kinetic reaction mechanisms. //Journal of Chemical Information and Modeling 60(1), 108–120.
20. Marquardt W, Morbach J, Wiesner A and Yang A (2010) OntoCAPE – A re-Usable Ontology for // Chemical Process Engineering, 1st Edn. Berlin: Springer-Verlag.
21. Weather Ontology, Institute of Computer Engineering at Technical University of Vienna, [Электронный ресурс]. <https://www.auto.tuwien.ac.at/downloads/thinkhome/ontology/WeatherOntology.owl> (дата обращения: 8.08.2025).
22. Eibeck A, Lim MQ and Kraft M (2019) J-Park simulator: An ontology-based platform for cross-domain scenarios in process industry. // Computers and Chemical Engineering 131, 106586.
23. Devanand A, Kraft M and Karimi IA (2019) Optimal site selection for modular nuclear power plants. //Computers and Chemical Engineering 125, 339–350.
24. Krdzavac N, Mosbach S, Nurkowski D, Buerger P, Akroyd J, Martin J, Menon A and Kraft M (2019) An ontology and semantic web service for quantum chemistry calculations. // Journal of Chemical Information and Modeling 59(7), 3154–3165.
25. Farazi F, Krdzavac N, Akroyd J, Mosbach S, Menon A, Nurkowski D and Kraft M (2020b) Linking reaction mechanisms and quantum chemistry: An ontological approach. // Computers and Chemical Engineering 137, 106813.
26. Bai J, Geeson RM, Farazi F, Mosbach S, Akroyd J, Bringley EJ and Kraft M (2021) Automated calibration of a poly(oxymethylene) dimethyl ether oxidation mechanism using knowledge-graph technology. //Journal of Chemical Information and Modeling 61(4), 1701–1717.

27. Farazi F, Salamanca M, Mosbach S, Akroyd J, Eibeck A, Aditya LK, Chadzynski A, Pan K, Zhou X, Zhang S, Lim MQ and Kraft M Knowledge graph approach to combustion chemistry and interoperability. // ACS Omega 5(29), 18342–18348.
28. Eibeck A, Chadzynski A, Lim MQ, Aditya LK, Ong L, Devanand A, Karmakar G, MosbachS, Lau R, Karimi IA, Foo EYS and Kraft M (2020) A parallel world framework for scenario analysis in knowledge graphs. // Data-Centric Engineering 1, e6.
29. Devanand A, Kraft M and Karimi IA (2019) Optimal site selection for modular nuclear power plants. //Computers and Chemical Engineering 125, 339–350.
30. Akroyd J, Mosbach S, Bhawe A, Kraft M. Universal Digital Twin - A Dynamic Knowledge Graph. //Data-Centric Engineering. 2021;2:e14. doi:10.1017/dce.2021.10
31. Chadzynski A, Li S, Grišiūtė A, et al. Semantic 3D city interfaces—Intelligent interactions on dynamic geospatial knowledge graphs. //Data-Centric Engineering. 2023;4:e20. doi:10.1017/dce.2023.14
32. Quek HY, Sielker F, Akroyd J, et al. The conundrum in smart city governance: Interoperability and compatibility in an ever-growing ecosystem of digital twins. //Data & Policy. 2023;5:e6. doi:10.1017/dap.2023.1.
33. Hofmeister M, Bai J, Brownbridge G, et al. Semantic agent framework for automated flood assessment using dynamic knowledge graphs. //Data-Centric Engineering. 2024;5:e14. doi:10.1017/dce.2024.11.
34. Linyao Yang, Hongyang Chen, Zhao Li, Xiao Ding, and Xindong Wu. Give us the facts: Enhancing large language models with knowledge graphs for fact-aware language modeling. //IEEE Transactions on Knowledge and Data Engineering, pages 1–20, 2024. doi:10.1109/TKDE.2024.3360454.
35. Amanda Kau, Xuzeng He, Aishwarya Nambissan, Aland Astudillo, Hui Yin, Amir Aryani Combining knowledge graphs and large language models// arXiv:2407.06564.
36. Jinheon Baek, Alham Fikri Aji, Amir Saffari, Knowledge-Augmented Language Model Prompting for Zero-Shot Knowledge Graph Question Answering // arXiv:2306.04136.
37. Chao Feng, Xinyu Zhang, and Zichu Fei. Knowledge solver: Teaching llms to search for domain knowledge from knowledge graphs, 2023. //arXiv:2309.03118.
38. Michihiro Yasunaga, Hongyu Ren, Antoine Bosselut, Percy Liang, and Jure Leskovec. Qagmn: Reasoning with language models and knowledge graphs for question answering, 2022. arXiv:2104.06378.
39. Zichen Chen, Ambuj K Singh, and Misha Sra. LMExplainer: a knowledge-enhanced explainer for language models, 2023. //arXiv:2303.16537.
40. Ikuya Yamada, Akari Asai, Hiroyuki Shindo, Hideaki Takeda, and Yuji Matsumoto. Luke: Deep contextualized entity representations with entity-aware self-attention, 2020. //arXiv:2010.01057.
41. Armin Toroghi, Willis Guo, Mohammad Mahdi Abdollah Pour, and Scott Sanner. Right for right reasons: Large language models for verifiable commonsense knowledge graph question answering, 2024. //arXiv:2403.01390.

42. Vamsi Krishna Kommineni, Birgitta König-Ries, and Sheeba Samuel. From human expert to machines: An llm supported approach to ontology and knowledge graph construction, 2024. *//arXiv:2403.08345*.
43. Lang Cao, Jimeng Sun, and Adam Cross. Autord: An automatic and end-to-end system for rare disease knowledge graph construction based on ontologies-enhanced large language models, 2024. *//arXiv:2403.00953*
44. Linyi Ding, Sizhe Zhou, Jinfeng Xiao, and Jiawei Han. Automated construction of theme specific knowledge graphs. *//arXiv preprint arXiv:2404.19146*, 2024.
45. Hanieh Khorashadizadeh, Fatima Zahra Amara, Morteza Ezzabady, Frédéric Ieng, Sanju Tiwari, Nandana Mihindukulasooriya, Jinghua Groppe, Soror Sahri, Farah Benamara, and Sven Groppe. Research trends for the interplay between large language models and knowledge graphs, 2024. *arXiv:2406.08223*.