Министерство образования и науки Российской Федерации

Федеральное государственное автономное

образовательное учреждение высшего образования

«Московский физико-технический институт

(государственный университет)»

Факультет инноваций и высоких технологий
Кафедра алгоритмов и технологий программирования

Магистерская диссертация
по направлению 01.04.02 «Прикладная математика и информатика»

на тему «Онтологический подход к построению диалоговых систем в рамках ограниченной предметной области»

Выполнил: Емельянов А. А. _____

(подпись)

Научный руководитель: д.ф.м.н., проф. Ройтберг М. А. _____

(подпись)

Зав. кафедрой: д.ф.м.н., проф. Ройтберг М. А. _____

(подпись)

г. Москва 2017

АННОТАЦИЯ

В данной работе проведено исследование использования онтологий при разработке диалоговых систем. Сформулированы основные требования, применяемые к онтологиям при разработке диалоговых систем.

Создана математическая модель онтологии AN Ontology, имеющая в основе метаграф, что является ее отличающей чертой.

На базе AN Ontology предложена мета онтология Meta AN Ontology, которая используется для моделирования конкретных предметных областей. Она содержит описание основных объектов и отношений. Отличием данной модели от существующих является то, что для доменных объектов и отношений обязательными аргументами являются время и пространство, что увеличивает обобщающую способность модели и позволяет легко моделировать контекстно-зависимые предметные области. Также для моделирования времени введено понятие бесконечно малой точки, принадлежащей временному интервалу. Данная конструкция позволяет упростить моделирование одновременных событий, а также соотнесение событий по времени.

Создана модель части предметной области Home Domain System (умный дом). Она основана на уже существующих моделях умного дома, но описана в парадигме Meta AN Ontology.

Разработана программа для работы с онтологией и генерации корпуса. Он состоит из двух частей: онтологии и генератора. Физически представляет собой библиотеку на языке python 3.6 и не имеет какого-либо графического интерфейса.

С помощью программы на базе созданной онтологии предметной области сгенерирован корпус, где каждому выражению на естественном языке сопоставляется его логический эквивалент, что вносит в корпус структуру и является отличительной чертой созданного корпуса.

Структура и объём работы. Магистерская диссертация состоит из 3 разделов, содержит 93 страницы, 29 рисунков, 2 таблицы, 100 источников.

ДИАЛОГОВЫЕ СИСТЕМЫ, ОНТОЛОГИИ, КОРПУС ТЕКСТОВ, ПРЕДМЕТНАЯ ОБЛАСТЬ, ОБРАБОТКА ЕСТЕСТВЕННОГО ЯЗЫКА МЕТАГРАФЫ

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

Естественно-языковая диалоговая система – это компьютерная система, взаимодействующая с пользователями посредством связного направленного неограниченного диалога на естественном языке для достижения конкретной цели или целей. «Связный» означает возможность возвращения к предыдущим взаимодействиям, задавать вопросы. «Неограниченный» означает, что высказывания пользователя не ограничены заранее объявленным набором команд или слов.

Диалоговые системы должны быть способны выполнять несколько задач, таких как интерпретация запросов и генерирование ответов на естественном языке, обработка феноменов диалога (зависимость между высказываниями, эллипсис и анафора), а также доступ к информационным ресурсам. Большинство таких задач требуют знания о реальном мире и контексте диалога. Однако знания, используемые в конкретных диалогах ограниченны некоторой частью реального мира, называемой предметной областью или доменом (domain) системы.

Разные задачи имеют разные домены, что делает разработку диалоговых систем сложной и требует отделения знаний от диалоговой системы для дальнейшего ее пере-использования. Однако сами знания о реальном мире должны быть структурированы и подготовлены для работы с диалоговой системой.

Актуальность работы

На данный момент существует много диалоговых систем сфокусированных как на ограниченных предметных областях, так и пытающихся охватить более широкий спектр естественного языка. Однако у них есть некоторые недостатки.

Одним из таких недостатков является то, что большинство диалоговых систем являются коммерческими, для которых недоступен исходный код, их модель, способ представления и способ работы со знаниями.

Сейчас есть много способов создания диалоговых систем. Они подразделяются на три группы: rule-based – основанные на правилах, corpus-based – основанные на корпусе и смешанные.

Первый способ предполагает задание заранее всех правил и ответов, по которым система будет отвечать. Так как многообразие вариантов очень велико, то качество таких диалоговых систем низкое.

С развитиями алгоритмов машинного обучения и больших данных, а также компьютерными мощностями, стало возможным использовать большие корпуса текстов для создания и обучения алгоритмов работы диалоговых систем (corpus-based). Однако недостатком подхода, использующего корпус, является то, что сначала диалоговая система должна заработать и некоторое время взаимодействовать с пользователями, чтобы собрать корпус. Один из путей решения этой проблемы: заранее *синтезировать искусственный корпус*, который может быть использован в разработке диалоговой системы.

Имея корпус можно обучить диалоговую систему отвечать хотя бы на базовые потребности пользователя и начать собирать данные из реальных диалогов.

Сейчас есть некоторые корпуса текстов, ориентированные на конкретную задачу (например, babi корпус [89]), для диалоговых систем.

Однако они не структурированы и обычно представлены в виде пары «строка – строка». Перебор всех возможных естественно-языковых конструкций является комплексной задачей и имеет показательную сложность алгоритма.

Также перебор заранее заданных языковых конструкций может порождать высказывания, которые являются несогласованными в смысловом плане. Еще одной проблемой при построении корпуса, учитывая только естественно-языковые конструкции, выступает то, что одному смыслу (или намерению пользователя) может соответствовать несколько кардинально разных высказываний. Например, высказывания «увеличить температуру на кухне до 27 °C» и «изменить температуру системы отопления до 27 °C» соответствуют одному и тому же намерению пользователя в рамках предметной области, но в естественном языке имеют различные лексические конструкции. Учет таких различных вариантов вручную на уровне естественного языка даже в рамках ограниченной предметной области становится сложной и трудоемкой задачей.

В качестве решения описанных проблем в работе предлагается метод генерации корпуса, где каждому высказыванию на естественном языке соответствует его логический (семантический) эквивалент, что вносит в корпус некоторую структуру. Причем несколько высказываний на естественном языке могут соответствовать одному логическому выражению.

Данный корпус строится на основе введенной онтологии предметной области, с которой работает диалоговая система. Сначала генерируются все возможные логические выражения в соответствии с моделью предметной области, а после для каждого генерируется набор *псевдо* естественных языковых высказываний (все слова в высказывании стоят в начальной форме и опущены некоторые служебные слова).

Среда «умный дом», как и многие другие, является постоянно изменяющейся и контекстно-зависимой. Каждый объект или отношение зависит от времени и пространства, что должно быть отражено в онтологии.

Цель и задачи работы

Так как время и ресурсы ограничены, то в данной работе я решил сосредоточиться на первой части разработки диалоговой системы, что и является целью данной работы: создания общей онтологии для моделирования предметных областей и инструмента для генерации искусственных корпусов текстов, которые будут использоваться при разработке диалоговых систем.

Для достижения этих целей в работе необходимо решить следующие задачи:

1. Провести исследования диалоговых систем и онтологий.
2. Определить требования, предъявляемые к структуре онтологий для диалоговых систем.
3. Разработать базовую онтологию для моделирования онтологии предметной области.
4. Построить модель предметной области на базе введенной ранее онтологии.
5. Разработать инструмент, позволяющий работать с онтологией и генерировать по ней корпус.
6. Создать корпус для выбранной ограниченной предметной области.

Научная новизна и практическая значимость работы

Созданная онтология и программа имеет ряд особенностей по сравнению с уже имеющимися моделями:

- Считается необходимым использовать дискретизацию по времени и пространству при создании онтологии. Использование времени и пространства как необходимых аргументов легко адаптирует созданную онтологию для моделирования как предметных областей связанных с интернетом, так и непосредственно диалогом, и делает онтологию более общей.

- В основе математической модели онтологии лежит метаграф, что не было использовано в других моделях, проанализированных в ходе данной работы.
- Разработанная программа позволяет генерировать не просто текстовые корпуса текстов, а высказывания, которым соответствуют логические выражения.
- Программа предоставляет возможность визуализации полученных логических выражений для высказываний.
- Также отметим, что онтология является легко масштабируемой, т. е. добавление новых элементов не влечет за собой изменений в уже добавленных элементах онтологии. Также созданная модель обладает большой обобщающей способностью.

Личный вклад автора

Все результаты, описанные в работе, получены лично соискателем.

1. ОБЗОР ОНТОЛОГИЙ И ДИАЛОГОВЫХ СИСТЕМ

1.1. Понятие диалоговой системы и подходы к их созданию

Диалог (dialogue) – это разговор между двумя людьми (или больше) или машинами [1]. **Диалоговая система** – это компьютерная система, которая общается с человеком естественным образом. Диалоговые системы используются пользователями для достижения конкретных целей, таким образом, большей частью они ориентированы на задачу (task-oriented) [2].

Основные исследования сосредоточены на разговорных диалоговых системах, поддерживающих естественный язык. Начали создаваться мульти-модальные диалоговые системы, которые поддерживают мульти-модальную входную информацию: текст, аудио, видео и др. Согласно [1], типичная мульти-модальная диалоговая система содержит следующие модули: Input (текст, аудио, видео, рукописный текст, и др.), Fusion (извлечение, распознавание, интеграция), Dialogue Manager (диалоговый менеджер), Knowledge Base (база знаний), Fission (выбор модальностей, логические рассуждения, генератор выходных данных), Output (текст, аудио, видео, рукописный текст, и др.) Рисунок 1.1. иллюстрирует общую архитектуру диалоговой системы.

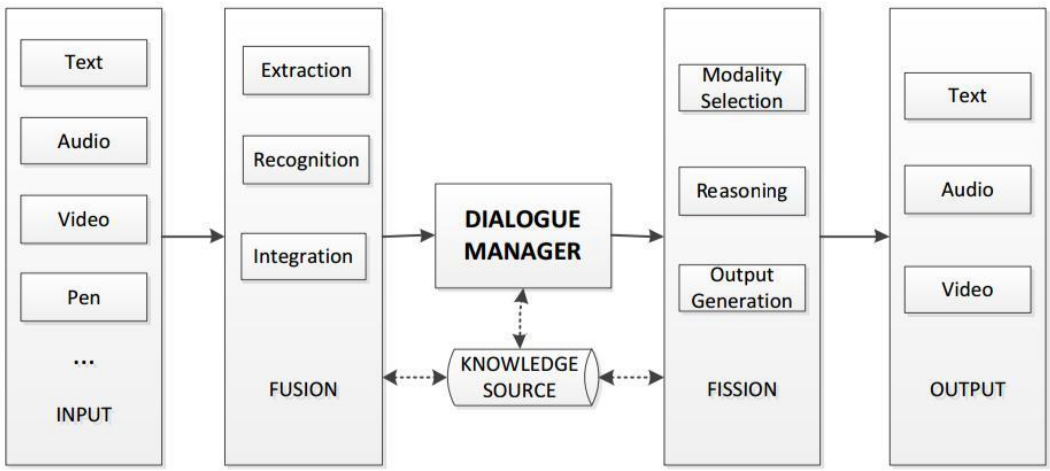

Рисунок 1.1. – общая архитектура диалоговой системы.

Диалоговый менеджер главный модуль управления для диалоговых систем. Он координирует работу компонентов системы. А также определяет намерения пользователя. Диалоговый менеджер контролирует поток данных через диалог, решая, как реагировать на сообщения пользователя.

В последние десятилетия было предложено несколько подходов для построения диалоговых систем. Каждый подход имеет свои достоинства и недостатки. В соответствии с построением и управлением диалоговые системы могут быть классифицированы на четыре категории [3]:

1. Подход, основанный на **конечных автоматах** (Finite state-based). Этот подход является наиболее распространенным и имеет наиболее простую стратегию управления диалогом. В его основе лежит некоторый конечный автомат, определяющий последовательность действий системы. Данный подход подходит для реализации простой диалоговой системы с хорошо структурированными задачами, но ему не хватает гибкости, естественности и применимости к другим предметным областям.
2. Подход, основанный на **фреймах** (Frame-based). Этот подход улучшает метод конечных автоматов для диалоговых систем. Система оперирует задачами по заполнению слотов (slot-filling task) для получения заранее предопределенного набора информации. Данный подход является подходом со смешанной инициативой. Он выглядит более гибким, эффективным и естественным по сравнению с предыдущим. Он способен обрабатывать ошибки распознавания и некоторый «недоразумения» во время общения с пользователями.
3. Подход, основанный на **информационных состояниях**, и **вероятностный подхо**д (Information state-based and

probabilistic). Information state-based подход оперирует состояниями, которые заранее предопределены и изменяются в соответствии с правилами и стратегиями. Их расширением является вероятностный подход, который использует вероятности для улучшения производительности. В данном подходе диалог моделируется как Марковский процесс [4] (MDP) или частично наблюдаемый Марковский процесс [5] (POMDP). Также применяют Байесовские [6] и нейронные сети для диалоговых систем.

4. Подход, основанный на использовании **агента** или **плана** (Plan-based or agent-based). Этот тип диалога основан на plan-based теориях речевого взаимодействия и диалога. С его помощью реализуются комплексные диалоговые задачи. Однако это влечет за собой высокий уровень расширения диалоговой системы.

Первые два типа подходов являются наиболее широко применяемыми методами в существующих системах диалога. Ряд практических систем был построен на их основе. Другие подходы, включая Information state-based, probabilistic-based, а также plan-based или agent-based относительно сложны и должны быть улучшены для практического использования.

1.2. Понятие онтологии

Согласно Gruber [7], онтология является точной формальной спецификацией некоторой общей концептуализации. Концептуализация – это абстрактная модель представления мира. Другими словами, онтология поддерживает общий словарь для представления знаний о задаче или предметной области. Она определяет понятия, свойства каждой концептуализации и ограничения на их использование в предметной области [2].

Люди могут иметь разные цели при создании онтологий, такие как [2]:

- Для совместного использования и понимания структурированной информации между людьми и программным обеспечением.
- Для обеспечения повторного использования знаний предметной области.
- Для создания явных предположений о предметной области.
- Чтобы отделить знания о предметной области от «технической» информации.
- Для анализа общих знаний в области [8].

Главной целью использования онтологий является поддержка общих знаний и их пере-использование. Это может быть применимо во многих областях, таких как человеческая коммуникация, системная совместимость и системная инженерия.

Согласно [8] есть несколько типичных шагов при построении онтологии:

Определение классов, представляющих понятия предметной области.

Определение таксономии классов или суперкласс-подкласс отношений.

Определение свойств классов и их экземпляров и ограничения на их значения.

Заполнить для конкретных экземпляров их свойства и их значения.

Valente и Breuker описали зависимость между онтологией, концепцией, представлением знаний и предметной областью (Рисунок 1.2.).

Концепции и предметные области – абстрактные сущности, в то время

как представления и онтологии конкретные «материальные» артефакты, представленные в точной форме [9].

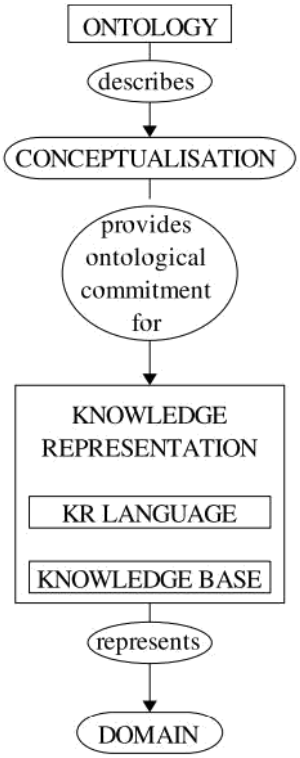

Рисунок 1.2. – зависимость между онтологией, концептуализацией, представлением знаний и предметной областью.

Онтология обычно состоит из следующих компонент [10]:

- Классы объектов.
- Классы могут иметь *атрибуты*, которые имеют *значения*.
- Классы могут существовать в *отношениях* между собой.
- Атрибуты и отношения могут изменяться во *времени*.
- События, которые происходят в разные *моменты времени* (*time instances*).
- *Процессы*, происходящие во времени, в которых участвуют объекты.
- Среда и объекты могут иметь различные *состояния*.
- События могут приводить к другим и состояниям, как *эффекты*.
- Классы могут моделировать объекты, состоящие из нескольких частей.

1.3. Языки онтологий

Онтология описывается с помощью языков представления онтологий.

Язык представления онтологий обычно вводит понятия и их свойства, отношения и дополнительные ограничения. Языки онтологий могут быть простыми, на основе фреймов (frame-based) или на основе логики (logic-based). Простые онтологии содержат только концепты и таксономии [12].

1. Frame-based онтологические языки, более структурированные по сравнению с простыми языками, и содержат как концепты, классы, так и их свойства. Классы (frame) являются основным семантическим примитивом данного типа.
2. Logic-based онтологические языки являются более сложными и выразительным. Они включают в себя два распространенных типа:
    a. Первый тип основан на логике первого порядка. Этот тип языка позволяет задавать общие предикаты.
    b. Другой тип, основанный на дескрипционной логике, которая описывает знания с точки зрения понятий и их отношений. Этот тип языков онтологий имеет возможность логического вывода. Другими словами, можно рассуждать о знании в основе логики, таких как умозаключения членства неявного класса, умозаключений неявного подкласса и эквивалентности классов, а также проверить согласованность. Вообще говоря, чем богаче язык, тем более неэффективно поддерживает рассуждения [2].

Далее приведем описание некоторых онтологических языков.

CycL (Common Logic (Common Logic Standard) [13], Cycl (the Syntax of Cycl) [11],) – формальный язык, синтаксис которого

происходит из исчисления предикатов первого порядка, и был разработан с целью создания онтологии с логическим выводом. Некоторые из основных понятий CycL: константы (словарь или «слова» языка как вещь, концепция и т. д.); переменные (обозначают константы или формулы); формулы (объединяют термины в значащие выражения); предикаты (выраженные отношения между терминами) и микротеориями. Микротеории – это множества формул, но они также могут участвовать в формулах.

Язык **Ontolingua** основан на KIF (Knowledge Interchange Format) [14] и Frame Ontology. Язык KIF имеет декларативную семантику и основан на исчислении предикатов первого порядка. Единицами языка являются: объект, функция, отношение и логические константы. KIF – это язык для обмена знаний и его сложно использовать для разработки онтологий. Таким образом, Frame Ontology построена поверх KIF и предоставляет определения для объектно-ориентированных и фреймовых терминов, например: класс, подкласс и экземпляр. Одним из достоинств этого подхода является то, что он интуитивно понятен. Но аксиомы не могут быть выражены в онтологии фреймов. Ontolingua позволяет разработчику решить, следует ли использовать полную выразительность KIF, где аксиомы могут быть выражены или быть более ограниченными во время спецификации с использованием только термов Frame Ontology. Онтология, разработанная с помощью Ontolingua, обычно определяется: отношениями, классами (определяются унарными отношениями), функциями (отношения), экземпляры (выделенные объекты) и аксиомами.

Язык **F-logic** (Frame Logic) – это интеграция объектно-ориентированной и фреймовой парадигм. Некоторые фундаментальные концепции объектно-ориентированных языков имеют прямое представление в F-logic, например класс, метод, типы и

наследование, а также другие вторичные аспекты, такие как полиморфизм. Одна из основных проблем объектно-ориентированного подхода – отсутствие логической семантики, которая решается с помощью логической основы F-logic.

Язык Web-онтологий (**OWL**), который предложен W3C, основан на дескрипционной логике [15]. Этот язык представляет собой веб-стандарт для обработки информации в Интернете. OWL написана в XML и предназначена для интерпретации компьютерами. Есть три подъязыка из OWL: OWL Lite, OWL DL и OWL Full. Среди них, OWL Full является наиболее выразительным, но ни одна из существующих машин логического вывода не поддерживает его [2]. OWL Lite и OWL DL разрешимы, но не столь выразительны. В дополнение к OWL DL и OWL Lite был предложен язык SWRL (Semantic Web Rule Language) [16].

1.4. Диалоговые системы и онтологии

В ранних диалоговых системах различие между знаниями предметной области, задачей и непосредственно знаниями, получаемыми в процессе диалога, было незначительно. Чтобы отвечать пользователю должным образом на его запрос система должна была обладать знаниями не только непосредственно о задаче, но и о предметной области. Таким образом, знание о задаче и предметной области часто интегрировались в диалоговую модель[17].

Рассмотрим некоторые примеры интегрирования знаний о предметной области и задачи в диалоговую систему. Типичной случаем является использование семантических фреймов как часть информационно-поддерживающих диалоговых систем [18], [19]. Иногда они формируют правила для интерпретации концептов и правил при заполнении форм.

Проект **SUNDIAL** (Speech UNderstanding in DIALogue) – это несколько диалоговых систем для обмена информацией по телефону,

созданных для разных языков [20]. В этой системе belief model [21] представлена как семантическая сеть, используемая для распознавания и интерпретации пользовательского ввода и управления диалогом. Модель содержит иерархию сематических классов, поддерживающую категоризацию и бинарные отношения между классами.

Еще одним примером выступает «нетрадиционная диалоговая система» **Verbmobil.** Особенность ее в том, что пользователи вели диалог не на родном языке. Задача системы состояла в мониторинге перевода пользователей, и при необходимости выдавала его [22]. Модель предметной области состояла из четырех частей: иерархия голосовых типов событий (speech-event types), иерархия концептов, тематические отношения и модель календарной информации [23].

Gatius and Gonzalez [24] представили диалоговую систему, которая использует онтологию для улучшения взаимодействия между системой и пользователем, обработки вмешательств пользователя и генерации сообщений системы. В данной статье основное внимание уделяется обработке естественного языка. Онтология используется в модулях обработки ввода и генерации вывода [2]. Она содержит понятия, связанные друг с другом с помощью основных отношений: is_a и instance_of. Понятия предметной области, используемые в диалоге, описаны в онтологии с помощью набора атрибутов. Эти атрибуты представляют информацию необходимую приложению от пользователя (или отдает пользователю) [2].

Sonntag и Romanelli [26] представили онтологическую синтаксическую структуру мульти-модальной вопросно-ответной системы. В этой статье онтология служит в качестве коммуникационного интерфейса между компонентами обработки естественного языка. Представленная онтология расширяет W3C

EMMA [27] описательный фреймворк (для синтаксиса) и MPEG-7 [28] для представления результатов в QA онтологии [26].

Bickmore и др. [29] предложили структуру на основе онтологий для диалоговых систем консультирования здоровья. В фреймворке реализованы несколько онтологий: Theory Model, Behavior Model, Protocol Model, User Model, External Data Model, Task Model. Представленные онтологии содержат в себе информацию о предметной области и поведении диалоговой системы.

Чтобы их представить, использовались стандартные формализмы OWL [15] и RDF [30], а также иерархическая модель задач [31]. Однако предложенная модель не делает управление диалоговой системой и знания предметной области независимыми.

Pardal и Mamede [33] расширили применение диалоговых систем для новых доменов знаний с помощью онтологий. Они создали диалоговую систему Cookcoach на основе фреймворка Университета Карнеги-Меллона

OLYMPUS [34], который включает в себя систему управления диалогом RAVENCLAW [35]. Онтология, **OntoChef** была разработана для предметной области, касающейся приготовления пищи. OntoChef содержит модули специально для приготовления пищи, включая действия, продукты питания, рецепты, столовые приборы. Cookcoach поддерживает различные рецепты. Однако модель поддерживает только ограниченную предметную область, связанную с приготовлением пищи.

Zhang [36] предложил модель для interactive requirement elicitation на основе онтологий. В этой модели, концепты, в том числе функции, качество, наряду с отношениями между этими понятиями и правилами устанавливаются особенно требование процессе. Едва ли применить модель для задач в других областях, таких как бронирование билетов поезда и т.д. [2]

Young [37] предложил модель скрытых состояний (Hidden Information State , **HIS**). Как онтологические правила в HIS принимаются модели задачи диалогового сценария и представляют собой группы belief states. Модель использует древесную структуру для представления цели пользователя.

Структура дерева в задаче специфична. Таким образом, правила в онтологии задаются на основе этой специфики [2].

Heinroth [38] создал адаптивный диалоговый менеджер **OwlSpeaker**, который поддерживает многозадачность в диалоговой системе. В OwlSpeaker, Spoken Dialogue Ontologies (SDOs) разработана для моделирования базы знаний для различных областей. Каждая SDO содержит речевой подкласс (Speech) и подкласс статуса (State), представленные в древовидной структуре. OwlSpeak позволяет переключаться между различными взаимосвязанными задачами. Тем не менее, цель работы заключается в оказании поддержки многозадачности в системе диалога, а не для создания доменно-независимой диалоговой систему [2].

Van Oijen [39] представил инициативную онтологию (ontology-driven), целе-ориентированную (goal-based) диалоговую систему для обучения агента. В системе онтологии используются для представления знаний в предметной области и знаний о роли агента. Они обеспечивают общие данные, установленные для понятий и их атрибутов, обмениваемых между агентами. Использование онтологий позволяет создать тесную связь между агентами, симуляции и предоставляет возможность повторного использования способностей агента [2].

В работе [40] объединены агенты и онтологии для поддержки task-oriented совместимости с NGAIEs [41]. Представленный фреймворк поддерживает взаимодействие в активных сферах (Activity Sphere).

Архитектура системы состоит из следующих частей: Sphere Manager (отвечает за создание, управление и удаление сфер), Ontology Manager (выполняет две функции: выравнивание онтологии и ответ на запросы в онтологии) и Interaction Agent (отвечает за взаимодействие с пользователем) [40]. Так как онтология разрабатывалась для совместимости с конкретным фреймворком, то она применима только для очень узкого круга задач.

Диалоговая система [42] имеет аналогичную архитектуру [40]. В ней также присутствуют компоненты Sphere Manager и Ontology Manager, но она тоже имеет ограниченную предметную область.

Далее более детально рассмотрены некоторые диалоговые системы, имеющие значимое практическое применение с использованием онтологий.

1.4.1. UK cancer referrals and Home control

В 2003 году Milward и Beveridge описали два приложения диалоговых систем основанных на онтологии. Эти системы основаны на трехуровневой архитектуре, которая отделяет знания предметной области и знания для конкретной задачи.

Особенности ответов в диалоговой системе

В диалоговых системах, которые обеспечивают относительно открытое общение, пользователи (U) могут предоставить ответы, которые имеют неточное соответствие с тем, что ожидает система (S). Примером частного случая может служить дача более информативного ответа, или же наоборот, какой-то его части.

В процессе диалога пользователь может давать ответы, которые не соответствуют какому-либо ответу из предметной

области. Примером такого ответа являются гипонимы [1] и гиперонимы[2].

В случае, когда ответом пользователя был гипероним, то его можно рассматривать в некоторых рамках предметной области, и система может задавать вопросы, чтобы прояснить позицию пользователя. В статье рассмотрен случай из предметной области рака.

S: У вас есть семейная история хронического заболевания?

[Система ожидает: рак легких, лейкоз, саркому, ...]

U: Да, рак

S: Какой тип рака?

U: рак легких.

В этом примере система ожидает, что ответом будет конкретное заболевание, но пользователь дает с более общий термин. Далее система формулирует более конкретный вопрос, чтобы получить ответ на ожидаемый уровень is-a иерархии.

Пример использования гипонима в качестве ответа приведен ниже [43].

S: У вас были какие-либо хронические заболевания?

[Система ожидает: рак, гипертония, сахарный диабет, ...]

U: Да, лейкемия

S: # Какой рак?

---

[1] Гипоним (греч. ὑπό — под, внизу + όνομα — имя) — понятие, выражающее частную сущность по отношению к другому, более общему понятию[44]. Является результатом логической операции ограничения.

[2] Гипероним (в лингвистике) — понятие в отношении к другому понятию, выражающее более общую сущность. В отношении некоторого множества объектов гиперонимом является понятие, отражающее надмножество к исходному [44]. Является результатом логической операции обобщения.

В этом примере система ожидает более общий ответ на вопрос (например, «рак»), но пользователь ответил более конкретным термином.

Система, однако, в состоянии понять, что был дан ответ на вопрос, на том основании, что лейкемия является раком. Система позволяет избежать повторного вопроса о типе рака, так как эта информация уже есть. Символом «#» обозначен возможный вопрос (подтекст).

Следующим важным аспектом в диалоге являются отношения «часть-целое» (part-whole relationships). В ходе диалога пользователь может дать ответ на сообщение, который не соответствует ни одному из ожидаемого диапазона ответов (как определено в прикладной области). А тем более он не является гипонимом и гиперонимом ожидаемого ответа, но связан с одним или несколькими ожидаемыми ответных действиями не с помощью is-a ссылки. Например, «мышцы» являются частью «колена», «руки» и т. д.

Еще одной важной особенность в диалоге является его согласованность (dialog coherence). В инициативных диалоговых системах вопросы группируются вместе во время разработки. Даже в системах, допускающих смешанную инициативу, порядок вопросов редко меняется.

Как это может быть достигнуто? Использование онтологической информации помогает решать данную проблему. Вопросы, уточняющие предыдущий вопрос, о конкретном атрибуте, или направленные на получение более конкретной информации (определяемой is-a или part-whole иерархией). В идеале сразу после это достигается при разработке хорошо структурированных текстов [Mann и Thompson, 1988].

Также онтологическая информация играет очень важную роль при распознавании (recognition), интерпретации (interpretation) и генерации (generation).

Для того чтобы обеспечить интеграцию с доменной областью, модель диалога делится на high-level и low-level представления. Представление низкого уровня определяет конечный автомат действий, записанный на языке VoiceXML. Представление высокого уровня выражает информацию о намерениях и информационных структурах, лежащих в основе диалога, наряду с его текущим состоянием сосредоточения внимания [Grosz и Sidner, 1986; Hobbs, 1996; Moore, 1995]. Информация на высоком уровне представляет основу знания о доменной области вместе со структурой намерений (intentional structure) и информационной структурой из онтологии предметной области. Используемая архитектура (аналогичная гибриду трех-уровневой архитектуры агента [Gat, 1998]) представлена на рисунке 1.3. [43].

Структура намерений пользователя в диалоге (intention model) может быть описана отношениями между сущностями в предметной области.

Пример приведен ниже [43].

I_1 = Intend(S, Intend(H, Know(S, clinical details)))

I_2 = Intend(S, Intend(H, Know-if(S, nipple discharge)))

I_3 = Intend(S, Intend(H, Know-if(S, bilateral nipple discharge)))

I_1 DOM I_2

I_1 DOM I_3

Здесь DOM значит преобладающее отношение [Grosz and Sidner, 1986].

Аналогичные конструкции использованы для задач генерации, интерпретации и распознавания.

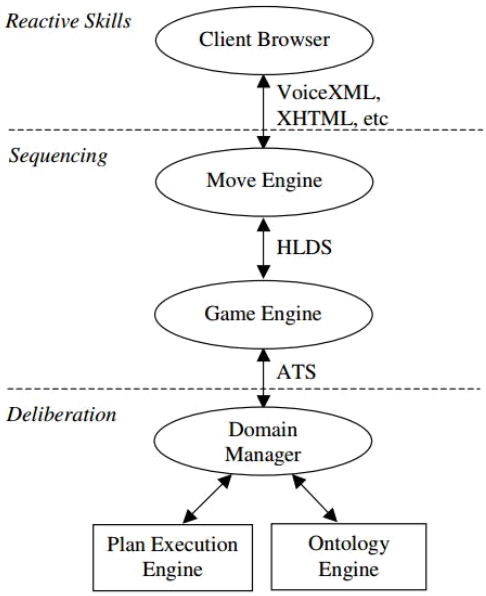

Рисунок 1.3. – архитектура диалоговой системы, предложенная Милвардом (Milward).

Стоит отметить, что модель покрывает большую часть естественного языка в рассмотренных предметных областях (Cancer Referrals и Home Device Querying and Control), также решает проблемы, связанные с гипонимами и гиперонимами, части-целого и в некоторой степени согласованности диалога.

Однако система является инициативной в сторону пользователя (пользователь только отвечает на вопросы и не задает действий).

Предложенная онтология в работе [45] это подтверждает. Эта работы связана только с is-a и part-whole отношениями [46], в то время как богатство языка этим не ограничивается.

Расширение системы для user-initiative диалоговой системы требует хорошей проработки онтологического концепта предметной области и доработки модели.

Положительные стороны, приведенные в работах [43], [45], [17], диалоговой системы состоят еще и в том, что новые компоненты могут быть добавлены динамически в систему; поддерживает асинхронный ввод и вывод; избегает неестественных скриптовых

диалогов; и гибкое мульти-модальное взаимодействие со всеми пользователями [46].

1.4.2. SmartKom

Это мульти-модальная мульти-доменная платформа диалоговой системы со смешанной инициативой. Она содержит несколько модулей и источников взаимодействия. Модули обработки могут быть сгруппированы следующим образом: обработка входных данных, анализ медиа данных, менеджер взаимодействия, менеджер приложения, генерация медиа и непосредственно выходных данных [17]. Такая организация повлекла за собой создание нескольких моделей: пользовательская модель, модель медиа, модель предметной области, модель предметной области, модель задачи и модель речи.

Одной из технических целей в рамках проекта SmartKom было решить следующие важные речевые явления, которые возникают в мультимодальных диалогах [47]:

- взаимная неоднозначность модальностей,
- разрешение и генерация кроссмодальных ссылок,
- разрешение и генерация мультимодальной анафоры,
- разрешение и генерация эллипсиса.

Сложность и глобальность модели делали онтологии зависимыми от языка. Она поддерживала общую верхне-уровневую онтологию и несколько специализированных доменных онтологий [48].

Онтология в SmartKom долго использовалась для NLP задач, таких как распознавание речи, управления диалогом и обмена информации [49]. В ее основе лежал онтологический язык OIL [50]

При автоматическом распознавании речи генерировалось несколько гипотез. Для выбора одной из них использовались

принципы семантической согласованности, которые базировались на онтологическом знании [49].

Пример приведен на рисунке 1.4.

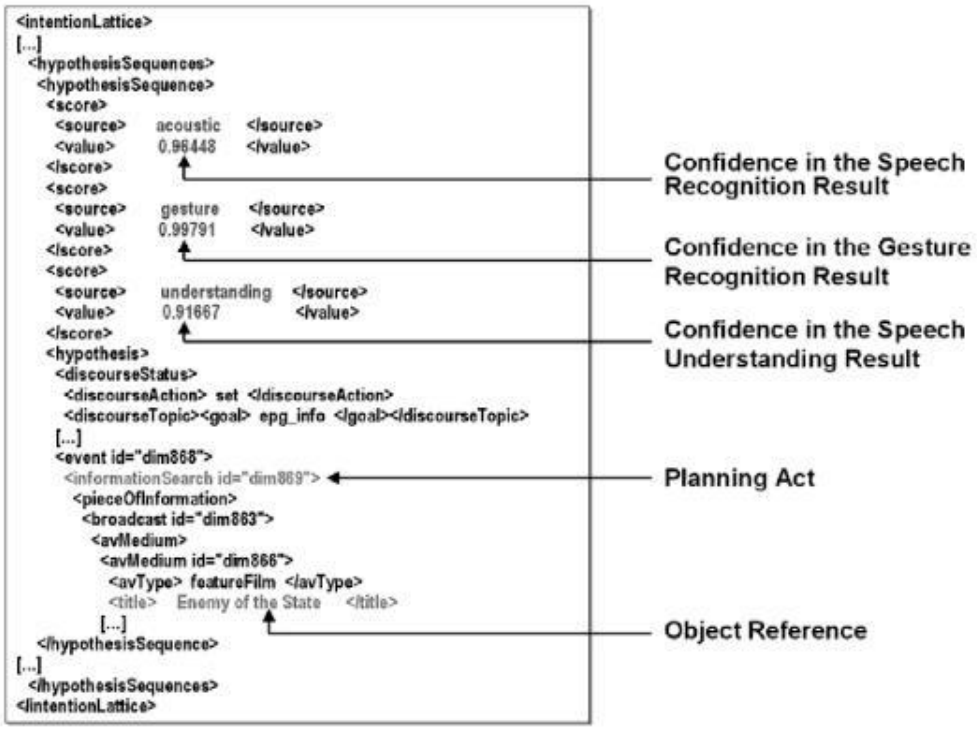

Рисунок 1.4. – пример гипотез при распознавании речи в SmartKom.

В основе анализа естественного языка в SmartKom был семантический парсер. Шаблон имел условную часть и часть действий. Действие выполнялось, если условия выполнялись с точки зрения наличия определенных слов или сущностей.

Диалоговый менеджер в SmartKom разрешал некоторые зависимости между высказываниями, используя речевую модель и модель доменной области из онтологии [48], [49].

Система SmartKom используется для решения 43 комплексных задач [51](рисунок 3):

- Помощник в коммуникации в телефоне, факсах, email и задачах аутентификации.
- Помощник в информационно-развлекательной среде: помощь в выборе медиа контента или TV (используя настольный компьютер, как мобильный клиент [48]).

- Помощник в поиске и извлечении информации.

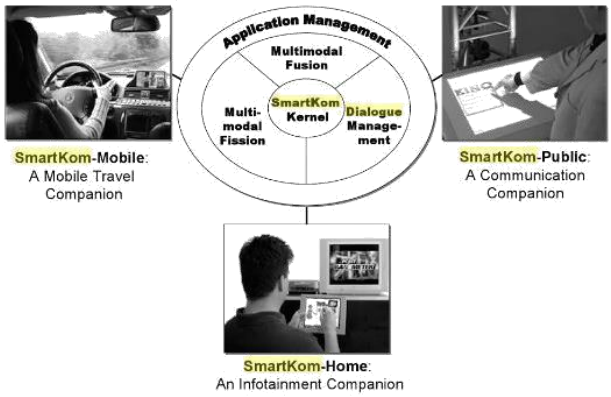

Рисунок 1.5. – применения SmartKom.

1.4.3. TRIPS

TRIPS – это фреймворк для мульти-доменной problem-solving диалоговой системы [33].

Архитектура диалоговой системы основана на нескольких агентах, каждый из которых выполняет лингвистическую задачу или задачу планирования, которая вносит свой вклад в общее диалоговое взаимодействие. Всего представлено четыре агента для интерпретации, генерации и поведенческий [54] аспект (специфика приложения). Рассмотрим интерпретатор, так как он основан на использовании онтологии.

```
(define-type LF_FOOD
 :parent LF_Physical-object
 :sem (phys-obj (origin natural)
                (object-function comestible)
                (mobility movable)))

(define-type LF_FILLING
 :parent LF_MOTION
 :sem (situation (cause agentive))
 :arguments (THEME (phys-obj (mobility moveable)))
            (AGENT (phys-obj (intentional +)))
            (GOAL (phys-obj (container +))))
```

Рисунок 1.6. – пример определений из языка онтологии понятий для пищевых объектов и заполнение событий в диалоговой системе TRIPS.

Интерпретация пользовательских высказываний может быть настроены для новых доменов за счет добавления знаний о предметной области специфического приложения в виде онтологии домена. Общая языковая онтология используется для лексики и в парсере для получения значения представления. Это затем может быть преобразовано в доменные представления путем сопоставления общих понятий в языковой онтологии предметной области [55].

Онтологии основаны на различных критериях разработки. Языковая онтология должна как можно более общей, иметь плоскую структуру, лингвистически мотивированных различия смыслов, имеют представление со структурой аргумента близко к языковой форме. Онтология предметной области должна покрывать концепты конкретной задачи и должна быть организована соответственно, иметь более глубокую структуру с сильными различиями и иметь представление, подходящее для диалога [46].

Сущности в онтологии имеют тип, которому соответствует некоторый набор свойств. Сущности также включают в себя структуру аргументов. В языковой онтологии это лингвистически ориентированные тематические структуры, например, с агентом, целью, тематическими ролями (рисунок 1).

Трансформация в онтологическое представление предметной области включает в себя применение синтаксиса и концептов. Структуры на рисунке 5 показывают семантическое представление в парсере, применение правил, и результат представления доменного знания высказывания «Load the oranges into the truck». Преобразование включает в себя замену определенных описаний, например, переменных ?v1 ?v2 с константой, в данном случае ORANGE2 и TRUCK3[46].

```
(SPEECHACT sa1 SA_REQUEST :content e123)
(F e123 LF_FILLING*load :agent pro1 :theme v1 :goal v2)
(IMPRO pro1 LF_PERSON :context-rel +YOU+)
(THE v1 LF_FOOD*orange)
(THE v2 LF_VEHICLE*truck)

(define-lf-to-kr-transform load-transform-trips
 :pattern ((?spec ?ev LF_FILLING :agent ?a :theme ?t
                                 :goal ?g)
           -> ((TYPE ?ev LOAD) (ACTOR ?ev ?a)
               (CARGO ?ev ?t) (CONTAINER ?ev ?g))))

(AND (TYPE e1 LOAD) (ACTOR e1 YOU123)
     (CARGO e1 oranges2) (VEHICLE e1 TRUCK))
```

Рисунок 1.7. – пошаговый интерпретация высказывания «Load the oranges into the truck», с примерами определений из онтологии языка правил отображения для перехода к представлению знаний предметной области в диалоговой системе TRIPS.

1.4.4. OWL for the In-Home Domain in Dialogue Manager

«In-Home Domain» диалоговая система – это фреймворк для управления домом. Особенностью данной системы является разделение управления внешними знаниями о предметной области (Knowledge manager) и, непосредственно, управления диалогом с пользователем. Общий вид системы показан на рисунке 1.8.

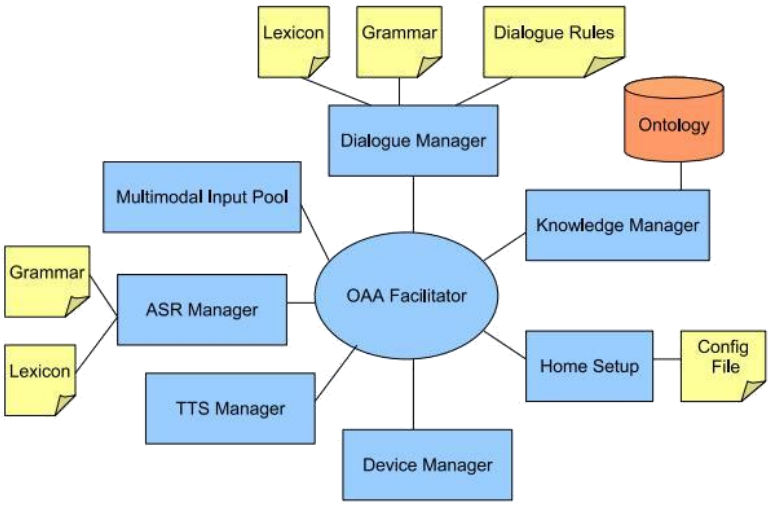

Рисунок 1.8 – общая архитектура диалоговой системы In-Home.

Ядром диалоговой системы является агент диалогового менеджера, который контролирует взаимодействие с пользователем, проверяя мульти-модальные входные данные [55]. Примерами входов могут быть: ASR агент (речевой агент) или Home Setup Agent (клики). Представление информации от диалогового менеджера для пользователя также могут передаваться в мульти-модальном режиме: голосом через агента TTS и графически через агента Home Setup. Для создания такого менеджера использовался подход описанный в работе [56], включая в себя систему Delfos [57] на уровне диалога и Episteme [58] для понимания естественного языка.

Диалог менеджер использует Knowledge manager для разрешения ссылок на конкретные устройства, которые могут быть переданы не только по названию, но и его расположение, типу и т.д. [55]. Подробное описание сценария и роли и услуг, предлагаемых каждым агентом можно найти в работе [59]. Для представления знаний о предметной области в Knowledge manager использовался язык OWL [21], [15].

OWL имеет следующие возможности для онтологий:

1. Масштабируемость для веб-потребностей,
2. Совместимость с веб-стандартами для доступности и интернационализация,
3. Возможность построения классов,
4. Ограничения по свойствам.

OWL был разделен на слои, чтобы отразить компромиссы между выразительностью и реализуемостью [55]:

1. OWL Lite поддерживает тех пользователей, в первую очередь нуждающихся в классовой иерархии объектов и простых ограничений.

2. OWL DL: максимальная выразительность при сохранении полноты вычислений (все заключения гарантировано будут вычислимы) и разрешимость (все вычисления завершатся в конечное время). OWL DL включает все языковые конструкции OWL при определенных ограничениях.
3. OWL Full: максимальная выразительность и синтаксическую свободу RDF [30] без каких-либо гарантий вычислительных.

Реализация базы данных в Knowledge manager разделена на два сервиса [55]:

1. DQ: Запросы о домене (объектах) некоторых свойствах.
    a. Параметры:
        i. P1: Класс.
        ii. P2: Список [nameProperty, value] пар; то есть свойств, которым субъект должен удовлетворять.
        iii. P3: Список [nameProperty, value] пар; то есть свойств, которым объект не должен удовлетворять.
    b. Решение: Объекты, принадлежащие к классу P1, совместные с ограничениями в графе, задаваемыми в P2 и P3.
2. RQ: Запросы о диапазоне (объектов) некоторых свойств, со следующими параметрами:
    3. Параметры:
        a. P1: Список объектов.
        b. P2: Список свойств.
    4. Решение: Все физические объекты, которые связаны с какими-либо объектами, указанными в P1 или в P2.

Основным элементом в OWL является триплет: субъект-предикат-объект. Субъекты и объекты обозначают классы и подклассы, в то время как предикаты, как правило, обозначают свойства.

Примерами классов в онтологии Knowledge manager является [55]:

System: специальный класс, используемый для описания роли диалоговой системы в онтологии, и как она взаимодействует с объектами из универсума.

Device: Описывает типы устройств, а также содержит подклассы.

Примером свойства в онтологии Knowledge manager выступает hasDeviceCommand. Это свойство содержит ряд подсвойств, каждое из которых соответствует одному типу функциональности устройств в доме[55].

1.4.5. GISE

Система GISE была разработана, чтобы обеспечить поддержку естественного языка в экспертных системах, хотя она также может быть адаптирована к другим типам приложениям, т.е. система отвечает должным образом запросы, связанные ограниченной предметной областью веб-сайтов.

GISE использует онтологии для моделирования приложения предметной области и производительности экспертной системы [60]. Задачи предметной области и системы представлены с использованием методологии спецификации задачи декларативно. Эта методика состоит из спецификации модели и спецификации процесса [61].

В GISE, концептуальной онтологией (CO) представлены понятия предметной области и коммуникационных задач, в то время как знания о естественном языке представлены лингвистической онтологией (LO). CO организована тремя независимыми таксономиями: понятия предметной области, атрибуты, описывающие понятия и операции над понятиями предметной области. Операции классифицируются как простые или сложные. Последний тип поддерживает возможность логического вывода для ответа на

сложные вопросы. Система использует обертки для получения информации из разнородных источников в Интернете. Извлеченные данные из веб-страниц, представляются в СО. Анализатор обрабатывает входные данные от пользователя, создавая набор возможных смысловых интерпретаций. Каждая интерпретация включает в себя эксплуатацию и концепции идентификаторов, а также рабочие параметры. Затем DM завершает семантическую интерпретацию, используя информацию, содержащуюся в СО, и историю диалога. Операция, которая выполняется совместно с его параметрами, находится в центре внимания (focus of attention). Для неоднозначных смысловых интерпретаций, DM рассматривает определение операции, а также контекст, и в центр внимания предыдущих взаимодействий. Когда все параметры операции определены, операция выполняется над экземплярами СО. Если удовлетворительного ответа не получено, то DM извлекает дополнительную информацию для приложения (на веб-страницах) [61].

1.4.6. HOPS

Диалоговая система HOPS [61] поддерживает взаимодействие с пользователями через текстовую и голосовую информацию на четырех языках. Архитектура диалоговой системы содержит следующие модули (рисунок 1.9):

- User Input,
- Natural Language Understanding,
- Dialogue Management,
- Database Access,
- Natural Language Generation.

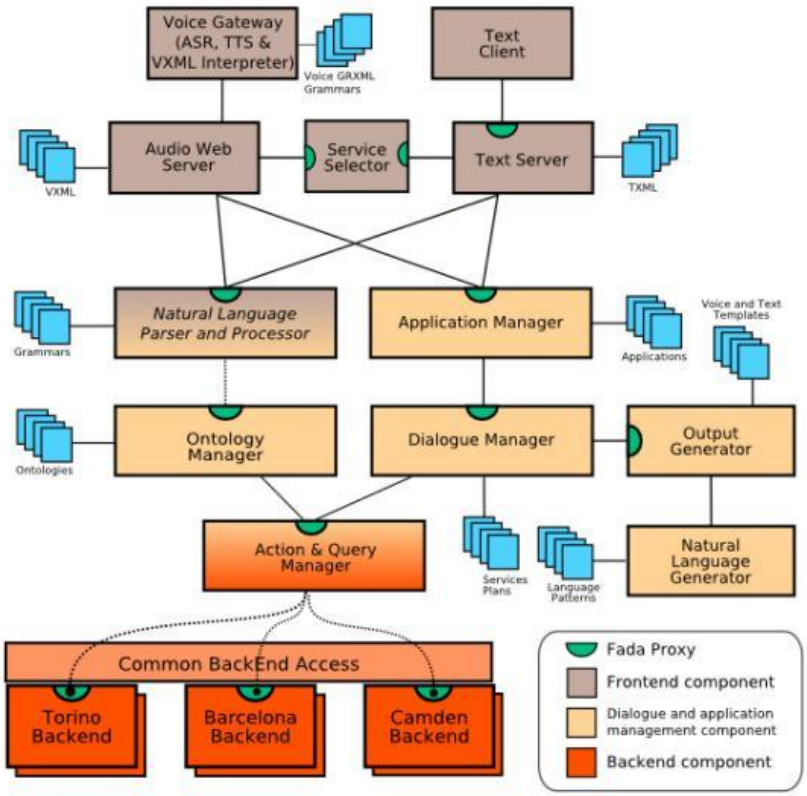

Рисунок 1.9 – архитектура диалоговой системы HOPS.

Сценарии поведения диалоговой системы представляют собой конечные автоматы. Текстовые входные данные обрабатываются синтаксическо-семантическим анализатором, описанным в [62]. Чтобы записывать онтологические данные использовался ABNF формат [63].

Диалоговая система HOPS использует онтологию для представления каждого сервиса и состояния в процессе диалога. Каждое состояние (State) может содержать следующую информацию:

- Атрибуты, которые пользователь должен предоставить в систему,
- Атрибуты системы, которые должны поддерживаться для конкретного пользователя (доступ к базе данных),
- Индикация состояния предварительных условий.

Особенности диалоговой системы HOPS [62]:

- Поддерживает mixed-initiative диалог и разрешение поддиалогов. Обнаруживаются намерения пользователя и учитываются при выборе следующего действия.
- Модуль генератора естественного языка (NLG) основан на коммуникационных операциях и атрибутах, определенных в модели диалога.
- Службы системы описаны с помощью онтологии, которая получает доступ обработчику и парсеру естественного языка, и используется для взаимодействия с пользователем.
- Онтология представляет сервисы, поддерживаемые системой. Эта онтология получает доступ естественного языка и синтаксического анализа процессора (NLPP) и двигателем обнаружения сервисов и языка, используемого в первом взаимодействии с пользователем.

Информационное состояние системы, поддерживаемое в HOPS, состоит из следующих компонент [62]:

- Описание информационного состояния.
- Набор действия диалога.
- Набор правил обновления информационного состояния системы.
- Контролирующая стратегия по выбору следующего действия.

Действия в диалоговой системе разделены по уровню доступа:

- Действия, доступные пользователю и диалоговой системе. Например, такие концепты, как answering, feedback, asking и др.
- Действия, доступные только диалоговой системе: exequery, reset и др.

Примером онтологических классов, которые участвуют в разборе пользовательских высказываний, выступают: title, date, price, location и др.

Генерация ответов на естественном языке использует две парадигмы для разных каналов: VoiceXML [64] для речи и TextXML [Hern´andez и др., 2006] для текстовых сообщений. Для поддержки четырех языков для каждого действия использовался сгенерированный полуавтоматически набор фраз [61] и шаблонов, представленных в XML [50].

Фразы, используемые HOPS построены с использованием правил.

Правила строятся из шаблонов (<имя модели>), концептов ([$ название концепта]), атрибуты ([& имя атрибута]), Значения атрибута ([@название_аттрибута]) и лексические конструкций ([lexical entry name]).

Основные понятия и атрибуты классов связаны с языком, а также с лексическими конструкциями. Значения атрибутов вставляются во фразу во время выполнения, так как зависят от контекста диалога. Шаблоны должны учитывать это замечание, строить как можно более жанра независимые фразы, насколько это возможно. Пример записи правила приведен на рисунке 1.10.

```
ask<event_type> -->
   Open      ([q-word] [verb_copulative_sg] [art] <event_type> (that [verb_wish_s])?)
   Open      (What<event_type> [verb_wish_q]?)
   Open      (What<event_type> [verb_wish_prep_q]?)
   Mid       (What <event_type> [verb_wish_q](: [music], [theatre], [cinema], [art]
             or [appointment])?)
   Mid       (What <event_type> [verb_wish_prep_q](: [music], [theatre], [cinema],
             [art] or [appointment])?)
   Directive ([verb_request] [art] <event_type> [verb_wish_s])
   Directive ([verb_request] what <event_type> [verb_wish_s])
   Directive ([verb_choose] [art] <event_type>: [music], [theatre], [cinema], [art] or [appointment])
```

Рисунок 1.10 – пример записи правила ask <event type>.

Для описания контента использовались следующие операторы: answ, ask; val place, search; Directive, Mid, Open; Directive, Mid, Open (указывает категорию шаблона); [name] (лексическая запись); < name > (класс шаблона); [$name] (концепт); [#name] (атрибут); [@name] (значение).

Диалоговая система HOPS позволяет осуществлять поиск информации в интернете. Однако с точки зрения организации понимания естественного языка решает только небольшие зависимости между высказываниями.

1.4.7. TaskOntology-based dialogue system

В данной работе создана диалоговая система, которая управляется с помощью онтологии и является предметно независимой. Концепты в модели этой онтологии показаны на рисунке 8 со схемой обозначений классов и подклассов [2].

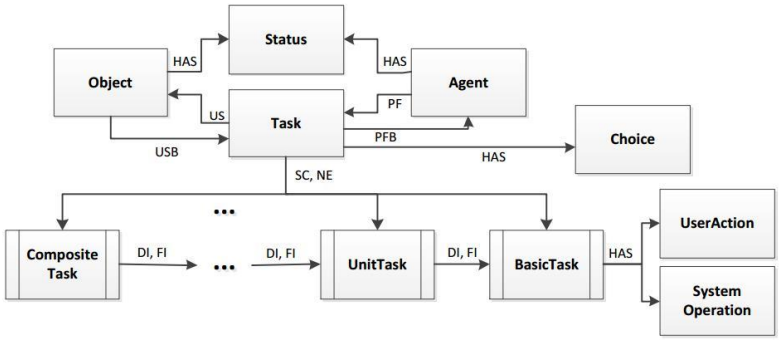

Рисунок 1.11 – модель task-oriented онтологии.

Объект и агент представляет контекст выполняемой задачи. Объект используется задачей; а агент выполняет задачу. Объект и агент имеют статус. Задачи – пользовательские мини-приложения, выполняющие их цели.

Есть три типа задач: CompositeTask, UnitTask и BasicTask. BasicTask имеет

UserAction и SystemOperation. Модели UserAction пользователи могут брать в задаче; в то время как SystemOperation является системная операция.

Описание компонентов можно найти в источнике [2], а здесь важно сказать, что данные объекты являются сущностями онтологии, между которыми могут устанавливаться отношения.

Примерами таких отношений выступают: USB: usedBy(x, y), x∈{Object}, y∈{Task}(объект x используется задаче y); HAS:

hasStatus(x, y), x∈{Object, Agent}, y∈{Status} (выражает статус агента или объекта, может быть как доступно, так и недоступно) и др.

Также в онтологии заданы логические правила асимметричности, нерефлексивности и др. для некоторых отношений.

Для представления онтологии используется OWL.

Данная диалоговая система является предметно-независимой. Однако нет функционала, позволяющего просто заполнять формы для пользовательской задачи, организовывать многоуровневые диалоги и анализировать естественный язык. Все это дополнительный функционал конкретной задачи, который реализует пользователь.

1.4.8. LINLIN framework

LINLIN фреймворк был адаптирован для двух различных предметных областей: автомобилей (CARS) и путешествий (TRAVELS).

LINLIN фреймворк имеет модульную архитектуру с интерпретатором (Interpreter), диалоговым менеджером (Dialogue Manager) и генератором (Generator). Архитектура представлена на рисунке 1.12.

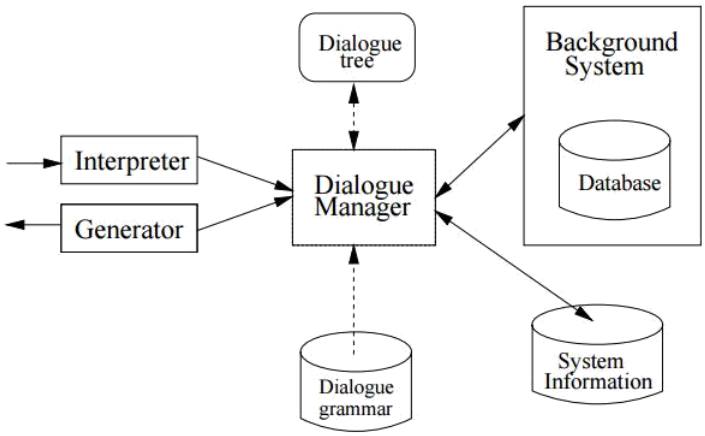

Рисунок 1.12 – архитектура LINLIN фреймворка для диалоговой системы.

Интерпретатор анализирует пользовательские сообщения и обеспечивает смысловое представление, передаваемое диалоговому менеджеру. Он основан на unication-based parser [65].

Основной задачей диалогового менеджера является поддержание диалога с пользователем и сохранение истории сообщений (если это необходимо). Диалог с пользователем, который поддерживается диалоговым менеджером, имеет смешанную инициативу, т. е. пользователю задаются уточняющие вопросы, до тех пор, пока все поля соответствующей формы не будут заполнены. Сам пользователь тоже может задавать уточняющие вопросы, которые обрабатывает диалоговый менеджер.

Генератор создает «презентацию» ответов и вопросов, приходящих от диалогового менеджера.

В LINLIN фреймворке управление диалогом в первую очередь сосредоточено на организации потока диалога и реакцию на высказывания пользователя. Для этого диалоговый менеджер использует модель диалога (dialogue model) и его историю. Каждое действие менеджера моделируется в грамматике диалога, зависящей от того, как сущности предметной области связаны отношениями с другими аналогичными объектами в домене и истории.

В онтологии определяются два важных концепта: объекты (Objects) и свойства (Properties), где Objects моделируют набор объектов в базе данных, а Properties обозначают сложный предикат, соответствующий этому набору.

Параметры Objects и Properties зависят от предметной области. Пример представления высказывания «Как быстра Volvo 850?» представлен на рисунке 1.13.

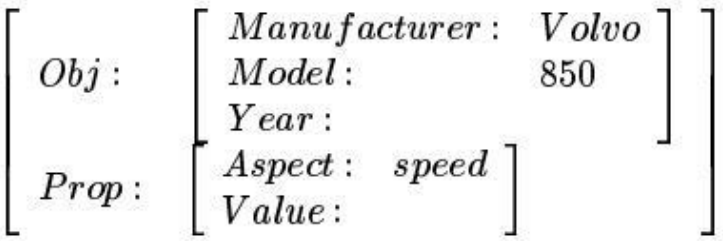

Рисунок 1.13 – представление высказывания «Как быстра Volvo 850?» в онтологии LINLIN фреймворка.

В рамках LINLIN фреймворка есть только одна история диалога, которая поддерживается диалоговым менеджером. Таким образом, остальные модули в системе не имеют памяти для предыдущих действий, так как это может привести к конфликтам. Запись в истории диалога содержит только информацию только о моменте в диалоге, в который она была сделана.

Таким образом, LINLIN фреймворк решает узкий круг задач, онтология имеет слабую выразительность для естественного языка. Его расширения – это Otraf фрейворк и MALIN [3] фреймворк. Они являются мульти-модальными в смысле обрабатываемой входной информации.

1.4.9. BirdQuest

BirdQuest – диалоговая система, поддерживающая информацию о птицах. Otraf и Malin фреймворки, основанные на LINLIN фреймворке использовали онтологии, чтобы отвечать на высказывания пользователя. В BirdQuest онтологии были использованы еще и для извлечения информации из неструктурированного текста и перевод этих данных в базу данных.

[3] Акроним MALIN расшифровывается как Multi-modal Application of LINLIN.

Структура онтологии в диалоговой системе BirdQuest представлена на рисунке 1.14. [66].

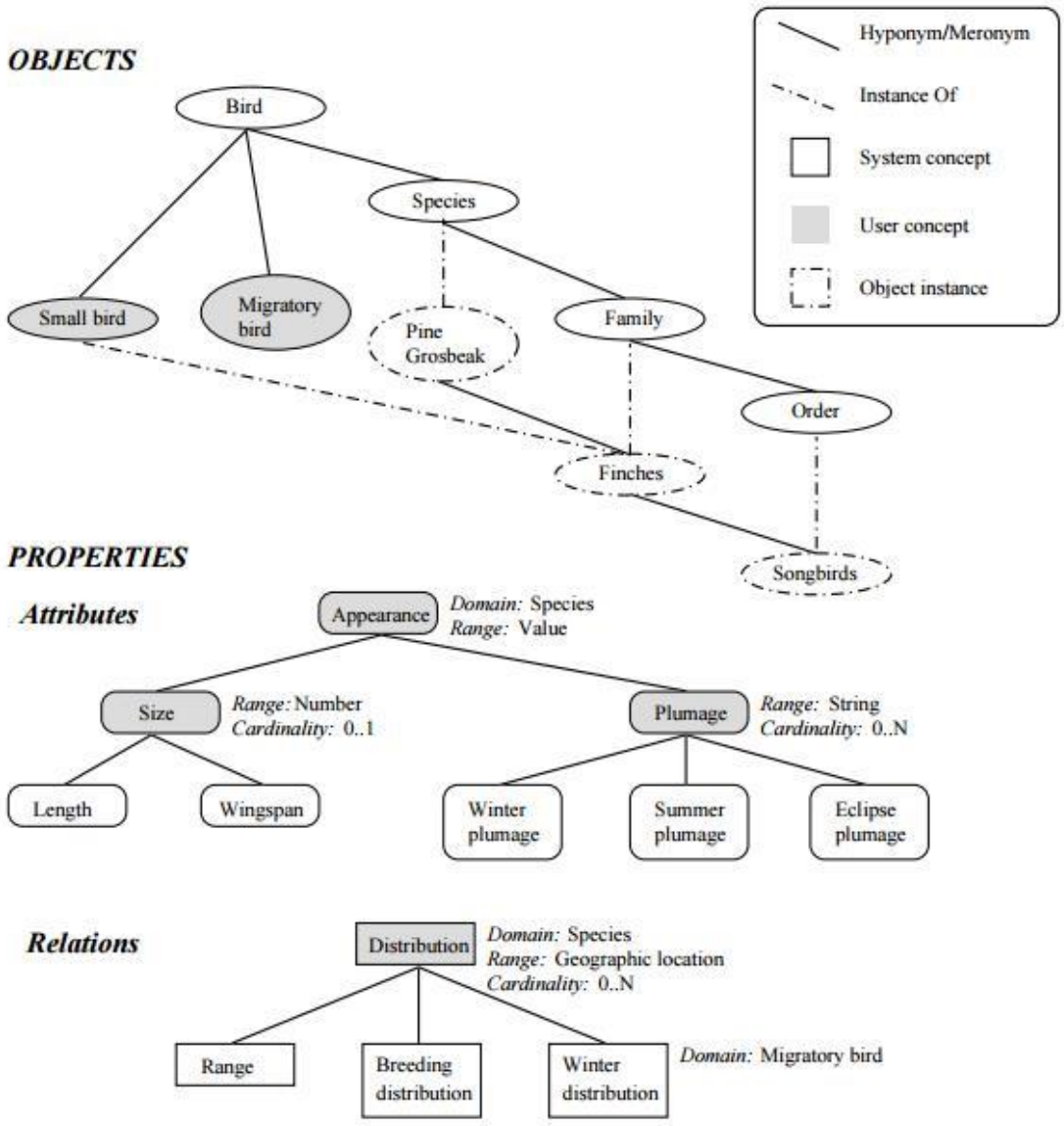

Рисунок 1.14 – часть онтологии в диалоговой системе BirdQuest.

В данной онтологии для некоторых объектов и свойств соответствовали понятия времени (TemporalRestrict) и пространства (SpatialRestrict). Например, вопрос: «Какие птицы я могу видеть в декабре в Швеции?» имеет интерпретацию в BirdQuest, представленную на рисунке 1.15.

В данном случае временное ограничение означает, что отношение Distribution с TemporalRestrict должно быть трансформировано в отношение WinterDistribution.

```
<O1, BirdSpecies: ALL>
<R1, WinterDistribution; O1:O2>
<O2, Country: Sweden>
```

Рисунок 1.15. – интерпретация предложения: «Какие птицы я могу видеть в декабре в Швеции?».

Таким образом, в BirdQuest поддерживается время и пространство, но только для некоторых элементов онтологии, а также заполнение базы данных из неразмеченного неструктурированного текста.

1.5. Обзор временных онтологий

В настоящее время процесс диалога, а также многие высказывания, описывающие события на естественном языке, включают в себя компонент времени. Контекстно-зависимые предметные области, например, Home Domain или Weather Domain, зависимы от времени непосредственно. Так пользователь может задать вопрос: «Какая погода завтра?». Очевидно, это высказывание зависит от времени. Чтобы уметь обрабатывать такие высказывания, требуется соответствующий логический аппарат.

В работе [67] рассматривается понятие выражения «exists» (∃x Dx) для выражения на естественном языке «There exist dinosaurs». Исследование данного высказывания приводят к необходимости введения временной логики в рассуждения.

Часто предполагается, что семантика временных выражений напрямую связана с понятием линейного времени.

В работе [68] рассматриваются английские времена и высказывания, содержащие процессные отношения. Например, «Harry reached the top». Здесь выделяется понятие *кульминация* (*culmination*) – событие, которое докладчик рассматривает как пунктуальное или мгновенное, а также сопровождается переходом в новое состояние мира. Это новое состояние следственное состояние (*consequent state*)

события. Высказывания были классифицированы в соответствии с происходящим действием (рисунок 1.16).

	EVENTS		STATES
	atomic	extended	
+conseq	**CULMINATION** recognize, spot, win the race	**CULMINATED PROCESS** build a house, eat a sandwhich	understand, love, know, resemble
–conseq	**POINT** hiccup, tap, wink	**PROCESS** run, swim, walk, play the piano	

Рисунок 1.16. – классификация высказываний в работе [68].

Приведенная классификация хорошо описывает разные времена и действия в высказываниях на естественном языке, однако сложно адаптируется к конкретной предметной области.

В TOWL онтологии [69] время выделяется как отдельный класс TemporalEntity, который имеет свойства, связанные с временными моментами, отвечающими за начало и конец. Он имеет два подкласса: Interval (интервал) и Instant (конкретное время). Подробная таксономия представлена на рисунке 1.17.

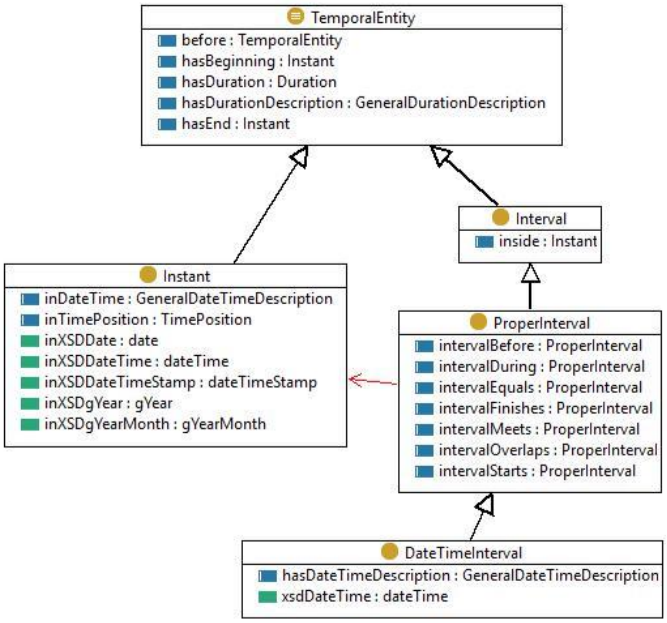

Рисунок 1.17. – описание времени в онтологии OWL.

Allen и Ferguson разработали правила вычисления бинарных отношений на интервалах, которые используются для представления временной информации и проблемы логического вывода с такой информацией. Отношения между интервалами, определенными в их исчислении, могут быть определены относительно прямолинейно в терминах: :before и идентифицировать в начальной и конечной точках [69].

Стандартное исчисление интервалов предполагает, что все интервалы правильные, начало и конец которых различны.

Для конкретных приложений в OWL созданы два типа данных, соответствующих времени: xsd:dateTime и xsd:dateTimeStamp [69]. Таким образом, время в онтологии OWL вводится как значение некоторого атрибута для описываемого объекта. Также OWL онтология разработана для структурированной информации в базе данных, и они главным образом сосредоточены на абсолютном времени и не покрывают многих аспектов естественного языка [70].

В онтологии CNTRO (A Semantic Web Ontology for Temporal Relation Inferencing in Clinical Narratives) [70] представлено расширение временной онтологии OWL, на которой она основана. Данная работа нацелена на представление и работу с клиническими записями. Чтобы охватить временные утверждения в естественно-языковых клинических описаниях в CNTRO вводятся новые временные классы: периодический временной интервал (Periodic Time Interval); отношение между двумя событиями (Relation between Two Event); относительное время (Relative Time); временная неопределенность (Uncertainty). Однако модель не покрывает всего языка, например не поддерживает отрицания («не позже чем», «не после»).

Еще одна модель, объединяющая в себе временную и пространственную размерность, Continuum [71] также основана на

OWL. В качестве основы пространственной модели использовался GeoSPARQL онтология [72], которая определяла пространственные объекты и отношения между ними. Пример объявления класса geo:SpatialObject приведен на рисунке 1.18.

```
geo:SpatialObject a rdfs:Class,
                    owl:Class;
  rdfs:isDefinedBy <http://www.opengis.net/spec/geosparql/1.0>;
  rdfs:label       "Spatial Object"@en;
  rdfs:comment     "The class Spatial Object represents everything that can
                    have a spatial representation. It is superclass of feature
                    and geometry"@en .
```

Рисунок 1.18 – объявление класса geo:SpatialObject в GeoSPARQL.

Структура модели Continuum использует hasGeometry и hasTime отношения для установления зависимости объектов от времени и пространства. На рисунке 1.19 представлено ядро данной модели.

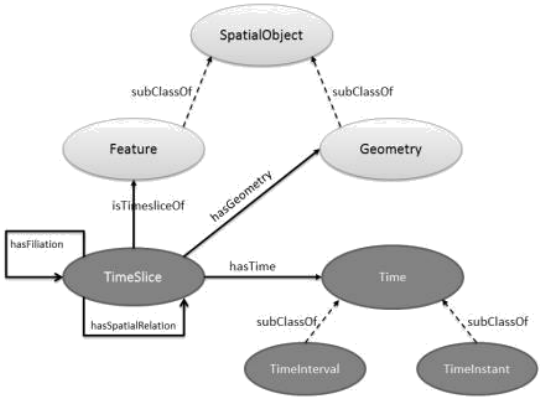

Рисунок 1.19 – ядро архитектуры модели Continuum.

Отношение hasSpatialRelation позволяет выразить пространственную топологию между объектами и должно быть установлено между двумя timeslices разных объектов, действительными в одном и том же интервале или моменте времени. Наконец, модель ориентирована на отношение hasFiliation. Это отношение устанавливается между двумя timeslices, оно позволяет отслеживать и определять эволюцию объекта в случае продолжения, а также отслеживать и определять преобразование одного объекта в

другой в случае вывода. Это отношение предназначено для перевода семантики перехода между двумя последовательными timeslices.

Эта взаимосвязь может быть определена более точно, объединяя identity-filiation и topological-filiation. Модель использует возможности специализации, присущие Semantic Web [71].

1.6. Выводы

В данном разделе описываются основные требования к онтологиям, предъявляемые для использования в диалоговых системах, и решения о структуры разрабатываемой нами онтологии. Основные требования будут касаться применимости онтологий к интерпретации, управлению диалогом и знаниями предметной области. Выводы, приведенные в этом разделе, основаны на исследовании диалоговых систем и онтологий, часть из которых приведена в обзоре.

1.6.1. Внутренняя структура

Что касается структуры, встает выбор между атомарными сущностями или имеющими некоторую внутреннюю структуру. Выбор между атомарными концептами (SUNDIAL, UK cancer referrals) и концептами с внутренней структурой (SmartKom) должен учитывать предметную область, тип синтаксического анализа и модель диалога. Атомарные конструкции проще и иногда поддерживают необходимую функциональность. Чаще бывает, когда выразительности атомарных конструкций недостаточно, например в task-model для разъясняющих вопросов, или когда необходимо включить в модель временную составляющую.

Решение: элементы онтологии имеют сложную структуру.

1.6.2. Типы свойств

Типы свойств (атрибуты) могут быть представлены как в системе или идентифицированы в словаре. Все свойства локальны

[10]. Они являются слотом во фрейме или связаны с концептом с помощью ограничений предметной области. Поскольку диалоговые системы работают с естественным языком, то свойства должны обладать полиморфизмом. Атрибуты могут принимать различные значения для разных экземпляров. Обычно много вопросов содержат значения атрибутов для экземпляров, например: «За какую дату показать письма?». При этом все свойства наследуются потомком. Таким образом, типы свойств должны быть локальны, наследуемы, обладать полиморфизмом и экземплярными [10].

Однако классы и родительские свойства могут быть опциональны и могут быть использованы в некоторых предметных областях в первую очередь для объявления и описания концептов.

Решение: для большей обобщенности и гибкости вводимой онтологии понятие атрибута класса не используется как *базовое* и вводится при моделировании предметной области при необходимости.

1.6.3. Ограничения атрибутов

Атрибуты могут иметь разные типы ограничений. Некоторые атрибуты могут становиться опциональными или необходимыми при анализе.

Ограничения на тип ограничивают концепт для атрибута, к которому он может быть применен для интерпретации или управления диалогом. Например, TRIPS использует это для интерпретации. Высказывание «мальчик улыбается» является валидным, в то время как высказывание «идея улыбается» – нет.

Жесткие ограничения не применяются во всех диалоговых системах, однако могут быть использованы для некоторых задач. Например, если используется частичный парсинг (partial parsing): комбинация идентифицированных объектов может быть основана на жестких ограничениях.

Иногда в некоторых задачах использую значения по умолчанию. Например, в задачах связанных со временем фразы «сейчас» и «сегодня» могут приниматься значениями свойств по умолчанию в диалоговой системе (TRIPS).

Решение: ограничения на тип необходимы, однако наложение жестких ограничений и значений по умолчанию являются опциональными.

1.6.4. Типы отношений

Бинарные отношения использую чаще, чем n-арные в общем, и это верно в случае диалоговых систем. Например, онтология SmartKom использует языки из семейства semantic web, которые поддерживают бинарные отношения. Однако в русском языке есть конструкции, которые не подходят под бинарное отношение, например: «расстояние от Москвы до Питера 700км.»

Решение: онтология должна иметь возможность использовать n-арные конструкции или механизм обхода таких отношений, выражением их через бинарные.

Также как будет показано дальше, все объекты онтологии – суть предикаты, имеющие обязательные аргументы.

1.6.5. Время и пространство

Использование времени и пространства в диалоговой системе зависит от предметной области. В UK cancer referrals и Home control диалоговых системах не использовалось время, так как не было необходимым. Однако для диалоговой системы SmartKom время являлось важным аргументом.

Например, begin_time для TV-program [10]. Использование понятия пространства необходимо для представления, например, рапортов о погоде.

Использование отдельно представления времени и пространства накладывает дополнительные обязанности на управление диалогом и сильно зависят от предметной области. Например, слово «завтра» должно быть заменено на соответствующую дату при дальнейшей обработке.

Однако многие сущности и значения их свойств зависят от пространства и времени. Для каждого момента времени объект может изменить свое местоположение и свои свойства. Что не учитывается во многих компьютерных системах, несмотря на важность данных параметров [73].

Решение: время и пространство – необходимые элементы онтологии.

1.6.6. Part-whole отношения

Bird Quest корпус определяет отношение PartOf необходимое для предметной области. Птицы описываются в терминах цветов, размеров частей тела, и для этого необходимо отношение PartOf. Также это может быть использовано для управлениями знаниями о предметной области или для обработки clarification, например, когда пользователь сказал: «Какого ворона цвета?» В базе данных должны быть цвета для всех частей тела птицы [10].

Решение: отношения данного типа опциональны для предметных областей.

1.6.7. Таксономическая организация категорий

Категории могут быть организованы структурно как деревья или как комбинация некоторых изменяющихся размерностей. Они могут быть объединены в одну таксономию или несколько распределенных структур. В мульти-доменных диалоговых системах, например, TRIPS и SmartKom несколько распределенных таксономий. В SmartKom это несколько онтологий верхнего уровня. Использование простых

древесных конструкций влечет за собой простое наследование свойств между концептами. Такая форма составляет основу OVERLAY оператора в SmartKom, а также образует онтологию в TRIPS, используемую при интерпретации [9], таким образом, давая возможность наследования многих свойств.

Использование древесных структур для диалоговых систем является естественным выбором, однако, выбор между одним или несколькими распределенными деревьями зависит от характера предметной области [9].

Однако несколько древесных структур и отделение таксономии от онтологии усложняет ее задание. Также таксономия может меняться во времени и пространстве.

Решение: таксономическое отношение выделено в онтологии как отдельный тип отношений и включено непосредственно в онтологию.

1.6.8. Отношения между категориями

Большинство систем использует is-a отношения для организации таксономии. Диалоговая система SmartKom основана на OIL, что позволяет некоторые разделения, но не четко ясно зачем и как это используется. Таким образом, is-a отношений достаточно для многих диалоговых систем [9], однако это может ограничить выразительность естественного языка. Также следует учитывать, что для некоторых групп свойственны разделенные категории.

Решение: *базовым* аппаратом для создания таксономий между категориями принимаем IsA отношение, которое относится к классу таксономических отношений и включено в онтологию.

1.6.9. Тип наследования

В большинстве диалоговых систем используется простое монотонное наследование. Однако, например в TRIPS оно не является

таковым. Вследствие этого монотонное наследование – некоторый baseline, однако множественное и не монотонное наследование применимо во многих случаях [9].

Решение: в онтологии применимо множественное наследование.

1.6.10. Разделение на уровни

Многое разделение на верхних уровнях онтологий очень специфично и зависит во многом от предметной области. Диалоговая система SmartKom использует более общую категоризацию верхнего уровня, что разделяет Roles и Types, которые потом делятся на Abstract и Instance, но с акцентом на процессах, что описывают поддерживаемые сервисы в различных приложениях [9].

С прагматичной точки зрения можно построить онтологию для диалоговой системы без общих разграничений. Но если в полученную онтологию потребуется интеграция новой онтологии, то разграничение на верхних уровнях будет играть значительную роль при интеграции и преобразовании концептов.

Таким образом, когда разрабатывается онтология для диалоговой системы, важно обращать внимание на возможность переиспользования и расширяемость.

Решение: для возможности интегрирования новых задач или предметной области разграничение на верхнем уровне должно быть как можно более общим в определенной степени [9].

1.6.11. Аксиомы

При анализе и корпусов наиболее важными аспектами является таксономия и то, как объявляются концепты, например, part-whole или non-monotonicity. Таким образом, аксиомы следует использовать только в слабой степени в данном случае [9].

Обратное отношение должно задаваться всякий раз, когда вводится новое отношение [9]. Тогда требуется использовать аксиомы из реляционной алгебры должны быть использованы.

Свойства должны быть отделены от типов сущностей, что значит необходимость наличия под-отношений, что может быть сделано в таксономии без использования аксиом [9]. Однако, такое разделение нарушает в некоторой степени общность онтологии.

Таким образом, использование аксиом не необходимо для диалоговых систем. В некоторых случаях их можно использовать для представления и ограничения отношений между свойствами концептов. Но тогда самим концептам следует иметь независимую организацию [9].

Решение: вводимая *базовая* онтология не использует аксиом.

1.6.11. Экземпляры

Здесь дается ответ на вопрос «Есть ли необходимость включения некоторых («обособленных») экземпляров (individuals) непосредственно в онтологию» (например некоторые обособленные факты).

В большинстве диалоговых систем отдельные экземпляры являются типами сущностей. В диалоговой системе Home control они включены в онтологию как листья в иерархии объектов. В TRIPS «обособленные» экземпляры введены преобразованием онтологических объектов в реальные экземпляры.

Решение: факты и «обособленные» экземпляры должны вводиться отдельно от онтологии в базе знаний.

2. ОПИСАНИЕ СТРУКТУРЫ ОНТОЛОГИЙ

В данном разделе описывается математическая модель вводимой онтологии, мета онтология и онтологическая модель предметной области.

2.1. Структура AN Ontology

Перед тем, как ввести понятие онтологии, определим необходимые математические определения.

Определение 1. [84] Гиперграфом $H = (V, E; R)$ называется пара множеств $V = \{v_i : i \in I\}$, $E = \{e_j : j \in J\}$ вместе с бинарным предикатом $R = R(v_i, e_j)$, определенным для любых $i \in I$, $j \in J$. Элементы $v_i \in V$ называются вершинами, элементы $e_j \in E$ гиперребрами, а предикат R инцидентором гиперграфа H.

Определение 2. Пусть дан гиперграф $H = (V, E; R)$. Метаграфом [74, 75, 76] $H^M = (V^M, E^M; \{R^M, R\}), M = 1$ называется пара множеств $V^M = \{v_i^M \in (E \cup V \cup \{v_{i^M}, i^M \in I^M\}), i \in I^M\}$, $E^M = \{e_j^M \in (E \cup \{e_j, j \in J^M\}), j \in J^M\}$ вместе с бинарными предикатами $R^M = R^M(v_i^M, e_j^M)$ определенным для любых $i \in I^M$, $j \in J^M$, и R. Элементы $v_i^M \in V^M$ называются вершинами, элементы $e_j^M \in E^M$ ребрами, а предикат R^M инцидентором метарграфа H^M. Множество $\{v_{i^M}, i^M \in I^M\}$ означает новые вершины, которые не присутствовали в гиперграфе H, но есть в метаграфе H^M.

Введенное определение метаграфа H^1 позволяет использовать в качестве вершин, как вершины гиперграфа H, так и его гиперребра. Отношение, связывающее вершины в метаграфе, называется метавершиной. Все элементы множества E^M – метавершины.

Определение 3. Пусть дан гиперграф H. Положим $H^0 = H$. Метаграфом $H^M = (V^M, E^M; R^M)$ порядка M называется пара множеств $V^M = \{v_i^M \in (E^{M-1} \cup V^{M-1} \cup \{v_{i^M}, i^M \in I^M\}), i \in I^M\}$,

$E^M = \{e_j^M \in (E^{M-1} \cup \{e_{j^M}, j^M \in J^M\}, j \in J^M)\}$ вместе с бинарными предикатами $R^M = \{R^m, m = \overline{0,M}\}$, где $R^m = R^m(v_i^m, e_j^m)$ определенными для любых $i \in I^m$, $j \in J^m$, где $m = \overline{0,M}$. Множества V^M и E^M определяются рекурсивно.

Определение 4. Подметаграфом H'^M метаграфа H^M (или частью метаграфа) будем называть метаграф $H'^M = (V'^M, E'^M; \{R'^m, m = \overline{0,M}\})$, если $V'^M \subseteq V^M$, $E'^M \subseteq E^M$ и R'^m индуцируется инцидентором R^m для $m = \overline{0,M}$.

Определение 5. Объединением метаграфов $H_1^{M_1}$ и $H_2^{M_2}$ называется метаграф

$$H = H_1^{M_1} \cup H_2^{M_2} = (V_1^{M_1} \cup V_2^{M_2}, E_1^{M_1} \cup E_2^{M_2}; R_1^{M_1} \cup R_2^{M_2})$$

Для краткости условимся, что функция $V(H^M) = V^M$, т. е. для метаграфа H^M она возвращает множество его вершин V^M. Аналогично $E(H^M) = E^M$.

Каждый элемент вводимой онтологии представляет собой предикат. Все доменные отношения в соответствии с идеями, вводимыми в работе BORO [67], зависят от времени и пространства. Само пространство также изменяется во времени. Таким образом, мы выделяем следующие базовые элементы онтологии, описываемые некоторыми метаграфами:

- Вспомогательные элементы онтологии, не зависящие от времени и пространства (некоторые абсолютные величины, например, меры измерения, типы данных; некоторые таксономические отношения и т. д.). Описываются метаграфом H^{M_H}:

$$H^{M_H} = (V_H^{M_H}, E_H^{M_H}; R_H^{M_H}) \quad (1)$$

- Элементы, описывающие временную составляющую онтологии (временные отношения, периоды, точные даты и т. д.). Описываются метаграфом T^{M_T}:

$$T^{M_T} = \left(V_T^{M_T} \cup V(H^{M_H}) \cup E(H^{M_H}), E_T^{M_T}; R_T^{M_T}\right) \quad (2)$$

- Пространственные элементы (локации и отношения между ними), зависящие от времени. Описываются метаграфом S^{M_S}

$$S^{M_S} = \left(V_S^{M_S} \cup V(T^{M_T}) \cup E(T^{M_T}), E_S^{M_S}; R_S^{M_S}\right) \quad (3)$$

- Объекты (Object) и отношения (Relation), зависящие от времени и пространства. Описываются метаграфом D^{M_D}

$$D^{M_D} = \left(V_D^{M_D} \cup V(S^{M_S}) \cup E(S^{M_S}), E_D^{M_D}; R_D^{M_D}\right) \quad (4)$$

Каждому из элементу из описанных метаграфов может соответствовать некоторый набор высказываний на естественном языке (в т. ч. пустой). Пусть он описывается графом L:

$$L = (V_L \cup V_E, E_L), \quad (5)$$

где

V_L – множество наборов слов на естественном языке, каждый такой набор может соответствовать одному или нескольким элементам метаграфа;

$V_E = V(D^{M_D}) \cup E(D^{M_D})$ – все элементы в ранее введенных метаграфах;

E_L – ребра графа L. Причем оно может быть проведено только между вершинами v_e и v_l, $v_l \in V_L$ и $v_e \in V_E$ для любых $l \leq |V_L|, e \leq |V_E|$.

Определение 6. AN Ontology – это метаграф O, получающийся путем объединения метаграфов, введенный ранее:

$$O = \langle D^{M_D}, S^{M_S}, T^{M_T}, H^{M_H}, L\rangle = D^{M_D} \cup S^{M_S} \cup T^{M_T} \cup H^{M_H} \cup L. \quad (6)$$

Введенную общую онтологическую модель можно представить в виде иерархической структуры (рисунок 2.1). Предикаты на каждом

уровне на рисунке в качестве аргументов могут иметь только элементы своего и предыдущего уровня.

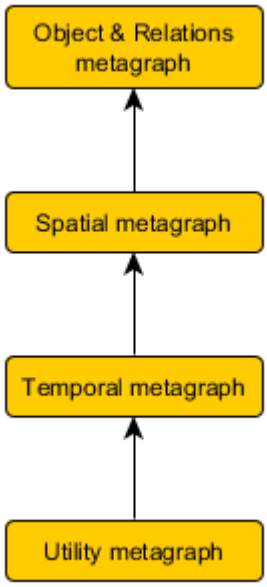

Рисунок 2.1 – иерархия метаграфов в AN Ontology.

Мы не ограничиваем порядки M_D , M_S , M_T , M_H вводимых метаграфов D^{M_D}, S^{M_S}, T^{M_T}, H^{M_H} соответственно, так как потенциально в языке могут быть использованы очень громоздкие большие вложенные конструкции. Как видно из определения метаграфа, для добавления новых метавершин достаточно описать их инцидентор R.

2.2. Структура Meta AN Ontology

Основываясь на определении AN Ontology, введем общую модель (Meta AN Ontology), с помощью которой будут описываться предметные области, высказывания на естественном языке и соответствующие им логические выражения. В совокупности каждый предикат и соответствующий ему набор на естественном языке представляют собой элемент онтологии (метавершину).

Пусть даны метаграфы (1)-(5) и определена онтология (6). На их основе построим модель Meta AN Ontology, с помощью которой далее будут описываться различные модели *реального мира*.

2.2.1. Моделирование вспомогательных элементов

В данном разделе производится моделирование вспомогательных предикатов Meta AN Ontology. Это самый «простой» уровень, так как предикаты зависят только от элементов, вводимых на этом уровне.

Taxonomy modeling

Для моделирования таксономий в метаграфе H^{M_H} введем бинарное отношение IsA:

$$\forall\, el_1, el_2 \in H^{M_H}\ IsA(el_1, el_2) \tag{7}$$

Далее запись вида $el_1, el_2 \in H^{M_H}$ перед формулой будет подразумеваться и опускаться для краткости.

Также здесь обобщим отношение IsA для более «высших» порядков онтологии:

$$\forall\, el_1, el_2 \in O\backslash L\ IsA(el_1, el_2) \tag{7'}$$

Types modeling

Аргументы некоторых предикатов могут иметь числовой, строковый или другой тип (не являться предикатом). Чтобы задать тип вводится предикат $Type$:

$$Type(el, type) = IsA(el, Type)\&(el = type) \tag{8}$$

Таксономия стандартных типов в Meta AN Ontology приведены на рисунке 2.2.

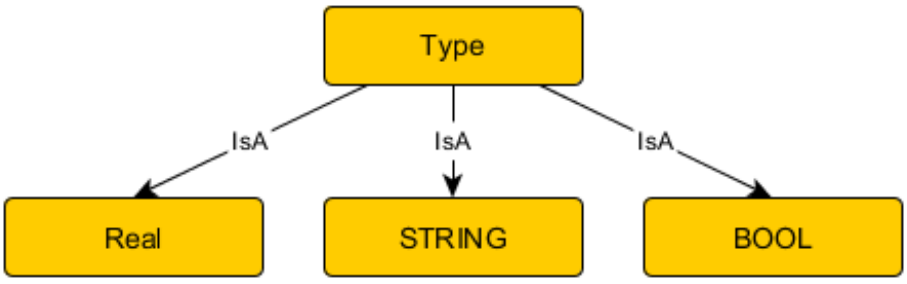

Рисунок 2.2. – таксономия типов в Meta AN Ontology.

Measure modeling

Здесь производится моделирование мер (шкал) измеримых величин, таких как температура, длина и т. д. Большое множество количественных данных используется и производится в научных экспериментах. Эти данные хранятся в формальных представлениях, так что ими можно манипулировать с помощью средств анализа и проектирования. Нашей целью в данной области не является улучшение существующих подходов, а их аккумуляция и интеграция в рамках Meta AN Ontology.

Существует несколько различных онтологий Unit of Measure, такие как EngMath (онтология для математического моделирования в технике), написанная в Ontolingua Gruber и Olsen [77]. UCUM созданная Schadow и др. [78] представляет собой систему кодов единиц и величин, на которые ссылаются, например, протоколы обмена электронными данными (EDI).

Другая онтология MUO (Measurement Units Ontology) в RDF [79], которая адоптировала юниты и измерения в UCUM и дала им URL. Модель OM (Ontology of units of Measure and related concepts) основана на OWL и включает опыт предыдущих работ.

Также есть и ряд других, более старых онтологий [80], [81], [82], [83], обзор которых приводится в работе [82].

Чтобы стандартизировать обработку измеримых данных в рамках Meta AN Ontology вводится предикат меры (Measure). Структура элемента представлена на рисунке 2.3. Ромбы на концах линий означают, что элементы являются аргументами верхнего предиката.

Мера Measure – это трехместный предикат, полностью характеризующий значение и единицу измерения.

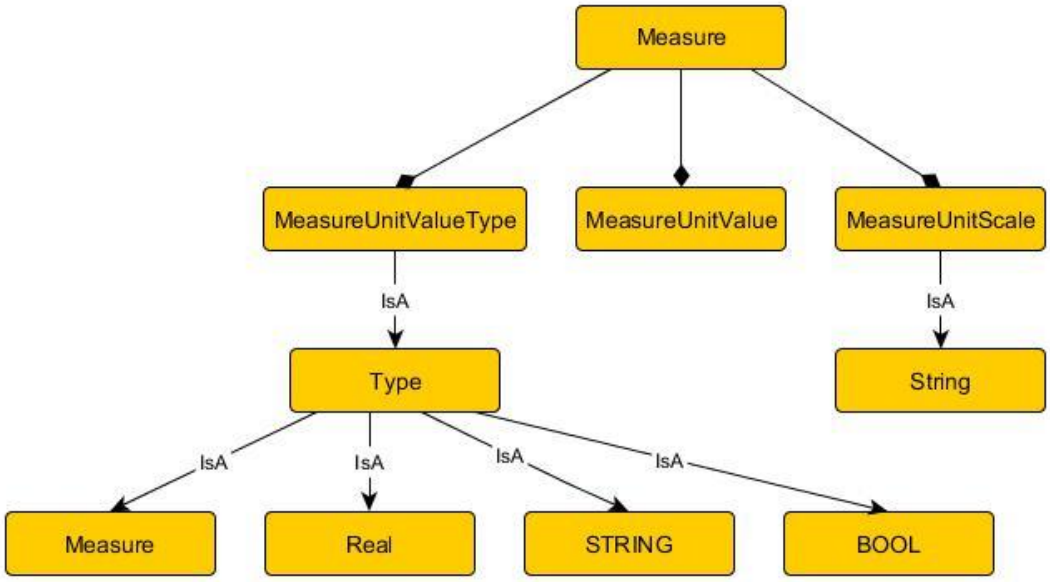

Рисунок 2.3 – структура предиката меры (Measure) в Meta AN Ontology.

MeasureUnitType – тип значения переменной. Он может быть как строковым, действительным или булевым (стандартные типы Meta AN Ontology), так и измеряться в более сложной шкале (MeasureUnitScale). Шкала обозначает единицы измерения меры, например, «Кельвин» (K) или «Градус Цельсия» (°C). MeasureUnitValue – непосредственное значение меры в единицах измерения MeasureUnitScale типа MeasureUnitType.

Логическая запись предиката приведена ниже:

$$Measure(muv, muvt, mus), \tag{9}$$

где

$MeasureUnitValue(muv, value) \equiv IsA(muv, MeasureUnitValue)\&muv = value,$

$MeasureUnitValueType(muvt, type) \equiv IsA(muv, MeasureUnitValueType)\&muvt = type,$

$MeasureUnitScale(mus, scale) \equiv IsA(muv, MeasureUnitScale)\&mus = scale.$

Пример высказывания на естественном языке, моделируемого предикатом (9): «23 °C», которое имеет следующее логическое представление:

$MeasureUnitValue(muv, 23)\&MeasureUnitValueType(muvt, REAL)\&$
$\backslash MeasureUnitScale(mus, "°C")\&Measure(muv, muvt, mus)$

«NoneObject» modeling

Иногда аргументы предикатов могут быть неопределенны. Часто это бывает при моделировании вопросов. Для обозначения неопределенного объекта (аргумента) вводится специальный предикат $NoneObject(el)$.

2.2.2. Моделирование времени

Предлагаемый подход для моделирования времени вбирает в себя идеи, предложенные TOWL [69] и Allen [87,88]. Было принято решение дополнить интервалы бесконечно малой точкой, принадлежащей данному интервалу, для обозначения абсолютного момента события. Вводимая точка в онтологии не обладает конкретным временным показателем, а только связывает с событие с интервалом. На рисунке 2.4 представлена структура временного элемента-объекта онтологии (часть метаграфа).

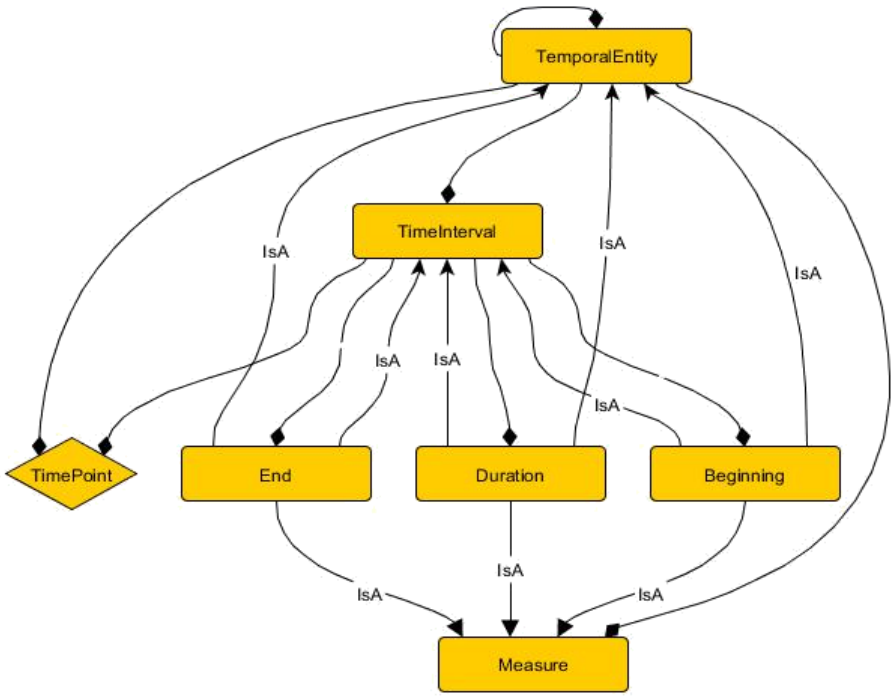

Рисунок 2.4 – структура предиката временной сущности в Meta AN Ontology.

Концепт TimePoint определяет бесконечно малую точку, принадлежащую интервалу (TimeInterval) или связывающуюся с временной сущностью (TemporalEntity). Они имеют в качестве аргументов начало (Beginning) и конец (End), а также продолжительность (Duration). Элемент TimePoint далее связывается с конкретным предикатом. Элемент TemporalEntity – это сложная временная сущность, которая в качестве аргументов может принимать не только временные интервалы и меры, но и другие временные сущности, заданные в онтологии.

Заметим, что TimePoint – служебный предикат, который не имеет выражения в естественном языке. Важность этого элемента состоит в том, что он помогает моделировать относительность событий друг с другом, а также выражать одновременность событий.

Также я не ввожу отдельно элемента для обозначения точного времени. Он представим в виде интервала с совпадающим началом и концом.

Аргументы Beginning, End и Duration могут быть как мерой (Measure), временным интервалом (TimeInterval), так и сложной временной сущностью (TemporalEntity).

Временной интервал в случае в простейшем случае, когда аргументы являются мерами, логически определяется следующим образом:

$$Begining(b)\&IsA(b,m_b)\&End(e)\&IsA(e,m_e)\&Duration(d)\&IsA(d,m_d)\&\backslash TimePoint(tp)\&TimeInterval(b,e,d,tp), \quad (10)$$

где m_b, m_e, m_d – это некоторые меры, моделирующие временные шкалы, необязательно различные.

Логическая запись TemporalEntity приведена ниже:

$$TimePoint(tp)\&TemporalEntity(\langle args\rangle, tp), \quad (11)$$

причем выполнено условие на аргументы: $\langle args\rangle \neq \emptyset, \forall a \in \langle args\rangle: IsA(a, TemporalEntity)\ or\ IsA(a, TimeInterval)$.

Помимо временных сущностей или интервалов в Meta AN Ontology вводится понятие временного отношения ($TemporalRelation$). Это отношение устанавливает зависимости между временными интервалами. В базовую онтологию вводятся следующие бинарные предикаты, наследующиеся от базового временного отношения $TemporalRelation$:

- $Before(t_1, t_2)$ – t_1 прежде t_2.
- $After(t_1, t_2)$ – t_1 после t_2.
- $Meets(t_1, t_2)$ – t_1 встречает t_2 (сразу после).
- $Metby(t_1, t_2)$ – t_1 встречено t_2 (сразу перед)
- $Contains(t_1, t_2)$ – t_1 содержится t_2.
- $Equal(t_1, t_2)$ – t_1 эквивалентен t_2.

2.2.3. Моделирование объектов и отношений

В данном разделе производится моделирование объектов и отношений в Meta AN Ontology, с помощью которых будет производиться описание почти всех объектов предметной области.

Концепты

Для обозначения минимальной единицы данного уровня введем понятие концепта ($Concept$):

$$Concept(< args >, loc(t_2), t_1), Concept \in D^{M_D}; loc \in S^{M_S}; t_1, t_2 \in T^{M_T}, \quad (12)$$

где

$< args >$ – набор аргументов предиката, в том числе пустой. Каждый такой аргумент может быть взят из метаграфа $O \backslash L$, т. е. быть любым элементом онтологии, кроме непосредственного его языкового представления.

loc – некоторая локация (пространственный предикат).

t_1, t_2 – временные элементы онтологии, описанные выше.

Базовая таксономическая структура этого уровня в Meta AN Ontology представлена на рисунке 2.5.

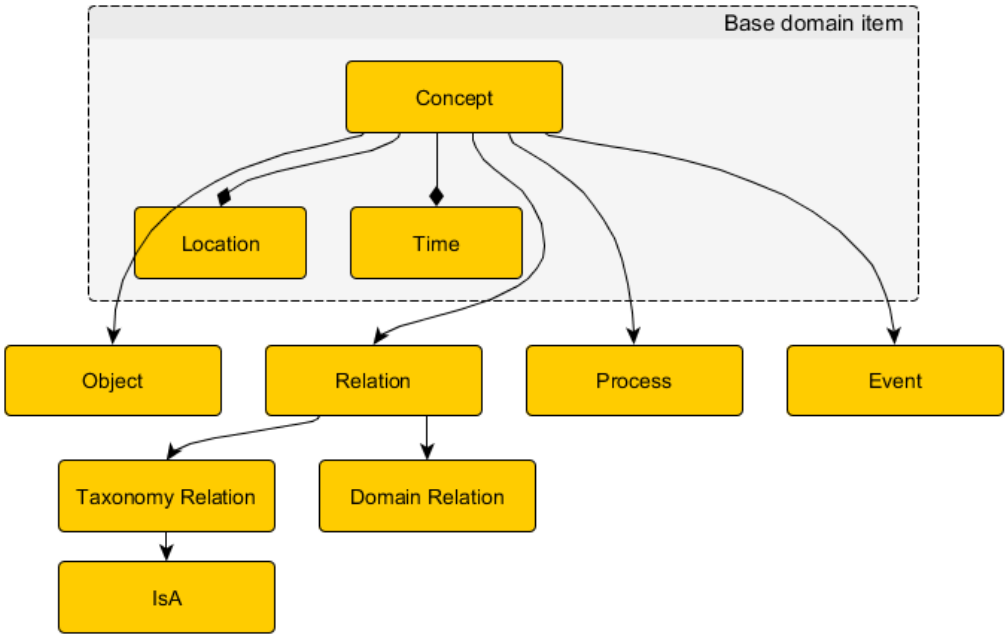

Рисунок 2.5 – структура концептов в Meta AN Ontology.

Каждый *простой* объект *реального мира* (например, слон) описывается (предполагается) с помощью двуместного предиката $Object$:

$$Object(loc(t_2), t_1), Object \in D^{M_D}; loc \in S^{M_S}; t_1, t_2 \in T^{M_T}. \quad (13)$$

Каждое отношение есть предикат, где множество $< args >$ непусто:

$$Relation(< args >, loc(t_2), t_1), Relation \in D^{M_D};$$
$$< args > \neq \emptyset \;\&\; \forall a \in < args >: a \in O \backslash L, m <; loc \in S^{M_S}; t_1, t_2 \in T^{M_T}. \quad (14)$$

Отношения делятся на таксономические (например, IsA) и доменные, которые связываются с конкретной предметной областью.

Отношения делятся на таксономические (Taxonomy Relations) и доменные (Domain Relations), описывающие конкретные предикаты из предметной области.

Отдельно выделены сложные временные элементы $Process$ и $Event$. Моделирование изменения объектов, перехода одного объекта (процессов) в другой описано ниже.

Изменение объектов во времени

Ограничиваясь для простоты лишь соотнесением мгновенных состояний одного и того же объекта, имеем для утверждений вида P(x, v_1, l, t_1) раньше, чем P(x, v_2, l, t_2) (например, «x был твердым раньше, чем жидким»), следующую логическую интерпретацию:

$$P(\mathrm{x}, v_1, l, t_1)\&P(\mathrm{x}, v_2, l, t_2)\&Before(t_1, t_2) \quad (15)$$

где P – некоторый предикат, v_1 и v_2 некоторые фиксированные значения аргументов состояния y соответствующих предикатов; l – пространственный предикат; t_1 и t_2 – временные предикаты, описываемые формулой Time. Также вводится предикат Before, который означает, что t_1 предшествует t_2. Здесь мы не приводи детализацию данного предиката, так как это выходит за рамки данной работы.

Существенно, что здесь описывается следование во времени значений одного и того же концепта. Отношение «одновременно» предполагает, конечно, сопоставление двух разных концептов, например «он был тогда студентом и много занимался спортом». Оно может быть представлено формулой:

$$P_1(\mathrm{x}, v_1, l, t)\&P_2(\mathrm{x}, v_2, l, t). \quad (15)$$

Чтобы моделировать высказывания «после футбола идет сериал», используется следующая конструкция:

$$P(\mathrm{x}, v_1, l, t_1)\&P(\mathrm{x}, v_2, l, t_2)\&After(t_1, t_2) \quad (16)$$

Здесь задачей стоит определить изменение объекта как предикат первого порядка. Так как признаки представляются в виде предикатов, то, например, «увеличение» или «уменьшение» некоторого признака («температура увеличилась с v_1 до v_2 градусов со времени t_1 к моменту t_2») может быть выражено предикатом $Change$:

$$Change(P, x, v_1, v_2, l, t_1, t_2) \equiv$$
$$P(x, v_1, l, t_1)\&P(x, v_2, l, t_2)\&comparison(v_1, v_2)\&Before(t_1, t_2), \quad (17)$$

где comparison – предикат сравнения значений v_1 и v_2.

Одновременные события

Здесь сосредоточимся на моделирование высказываний вида «х одновременно смотрит сериал и говорит по телефону». Таким образом, объект х в один и тот же момент времени выполняет несколько действий.

Данное высказывание имеет следующее логическое представление:

$$TimeInterval(b_1, e_1, d_1, tp)\&TimeInterval(b_2, e_2, d_2, tp)\&TalkPhone(x, l, tp)\&Watch(x, TV, l, tp)$$

где

$TalkPhone$ – отношение, означающее «говорит по телефону»,

$Watch$ – отношение, означающее «смотрит»,

x – объект, который выполняет действия,

l – локация,

b_1, b_2 – начала интервалов,

e_1, e_2 – концы интервалов,

d_1, d_1 – продолжительности интервалов,

tp – бесконечно малая точка.

Как видно, связывается сразу с двумя временными интервалами.

2.3. Home domain ontology

В качестве предметной области, для которой будет создаваться обучающий корпус, выбрана область Smart Home (Умный дом). Разработка интеллектуальных независимых и адаптивных к изменениям окружающей среды систем ведется последние десятилетия. Некоторые исследователи работали над интеллектуальными пространствами, такими как дом, офис,

университет, больница, гостиница, автомобили и другие частные или общественные места.

В качестве примера моделирования с помощью Meta AN Ontology, генерирования логических выражений и высказываний на естественном языке возьмем программу телепередач, возможные локации, различные домашние системы и т. д. В данной работе *не приводится* полная онтология предметной области, а лишь некоторые ее части, из-за ее комплексности.

Описанная онтология основана на работе [88] и адаптирована для Meta AN Ontology.

2.3.1. Моделирование локаций

Локация – это необходимый аргумент каждого предиката на доменном уровне онтологии и важная составляющая предметной области. Каждому оборудованию соответствует некоторое пространственное положение, которое может меняться во времени. На рисунке 2.6 представлена структура локация в HDS.

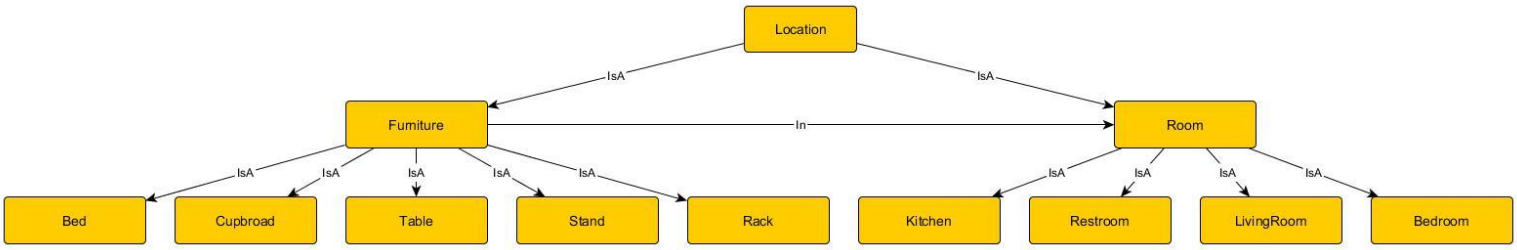

Рисунок 2.6 – структура локаций в HDS.

2.3.2. Моделирование телепрограммы и ее «составляющих»

Выделим сущности предметной области и их свойства для построения модели. Это непосредственно сама телепрограмма ($TVProgram$). Она имеет название ($Name$), и время ($TVProgrammTimeInterval$) и место ее показа (TV). Сам процесс ее показа описывается предикатом $ShowProgramm$.

- Название телепрограммы имеет строковое значение и моделируется с помощью меры Name, значение переменной которой имеет тип STRING:

$MeasureUnitValue(muv, name)\&MeasureUnitValueType(muvt, STRING)\&\backslash$
$MeasureUnitScale(mus, "")\&IsA(Name, Measure)\&Name(muv, muvt, mus),$
где

$name$ – это название программы.

- За время показа телепрограммы отвечает сложный временной предикат $TVProgrammTimeInterval$:

$TVProgrammTimeInterval(begining_{from}, end_{to}, duration),$

где

$begining_{from}$ – это временной интервал $TVProgrammTimeIntervalFrom$, который означает время начала показа программы. В качестве начала и конца он имеет предикаты $TemporalEntity$, которые моделируют непосредственно точное время. Аргументы $TemporalEntity$ – это временные интервалы, моделирующие часы ($HourTimeInterval$) и минуты ($MinuteTimeInterval$).

Например, предикат $HourTimeInterval$ в логическом виде записывается следующим образом:

$MeasureUnitType(bb)\&(bb = STRING)\&\backslash$
$HourOfDayValue(bc)\&IsA(bc, MeasureUnitValue)\&(bc = 3)\&HourScale(bd)\&\backslash$
$IsA(bd, MeasureUnitScale)\&(bd = Hour)\&HourMeasure(b)\&\backslash$
$HourMeasure(bb, bc, bd)\&IsA(b, Measure)\&\backslash$
$NoneObject(d)\& TimePoint(a)\&\backslash$
$IsA(b, Begining)\ \&\ IsA(b, End)\&IsA(d, Duration)\&\backslash$
$TimeInterval(e)\&IsA(e, TimeInterval)\&\backslash$
$HourTimeInterval(b, b, d, a).$

end_{to} – это временной интервал $TVProgrammTimeIntervalTo$, который означает время начала показа программы и имеет аналогичную структуру с $begining_{from}$.

$duration$ – это длительность телепрограммы, в нашем случае она не определена ($NoneObject$).

- Место показа телепрограммы (локация) имеет следующую логическую запись $TV(a)$, где указывает на $TimePoint$ из временного интервала $TVProgrammTimeInterval$.
- Предикат $TVProgramm$, моделирующий саму телепрограмму, имеет следующую логическую запись:

$$TVProgramm(Name, TV, a),$$

где указывает на $TimePoint$ из временного интервала $TVProgrammTimeInterval$;

TV – локация, введенная выше.

- Показ программы ($ShowProgramm$) имеет логическую запись:

$$ShowProgramm(TVProgramm, TV, a).$$

Приведенный предикат $ShowProgramm$ моделирует высказывания на естественном языке вида: идет телепрограмма «Наедине со всеми» по телевизору начиная со времени 4 часа 0 минут, заканчивая в 5 ч. 0 мин.

2.3.3. Моделирование систем оборудования и состояний

В предметной области Home Domain System (HDS) оборудование (Equipment) обычно имеет состояние, описываемое некоторым предикатом. На рисунке 2.7 представлена структура систем оборудования (Equipment) в HDS.

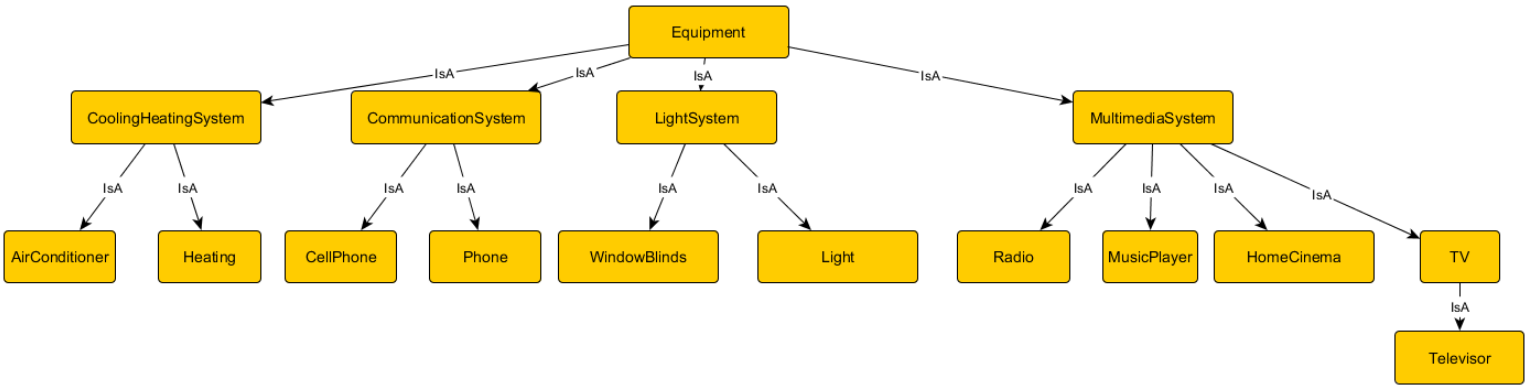

Рисунок 2.7 – структура систем оборудования в HDS.

На рисунке представлены следующие системы:

- MultimediaSystem – описывает мульти-медиа системы в доме, такие как радио (Radio), музыкальный плейер (MusicPlayer), домашний кинотеатр (HomeCinema) и телевизор (TV).
- CoolingHeatingSystem – система кондиционирования и отопления, которая включает в себя систему кондиционирования (AirConditioner) и систему обогрева (Heating).
- LightSystem – система освещения подразделяется на жалюзи (WindowBlinds) и ламповое освещение (Light).
- CommunicationSystem – система коммуникаций. Включает в себя домашний телефон (CellPhone) и мобильные телефоны (Phone).

Каждый элемент, являющийся Equipment, реализуется предикатом $Object$ в Meta AN Ontology и описывается некоторым состоянием.

Каждое состояние – это предикат отношения ($Relation$) следующего вида:

$$S = Relation(Equipment, Measure, Location, a),$$

где

$Equipment$ – некоторая система оборудования.

$Measure$ – некоторая мера, описывающая какой-то параметр оборудования.

$Location$ – локация, где находится оборудования.

a – элемент $TimePoint$, указывающий на некоторый временной интервал.

В HDS представлены следующие состояния:

- Noise – уровень звука. В качестве аргумента $Equipment$ может иметь MultimediaSystem и потомков. Значение, которое принимает предикат MeasureUnitValue, берется из множества {“средний”,”низкий”,”высокий”}.
- Temperature – температура. В качестве аргумента $Equipment$ может иметь CoolingHeatingSystem и потомков. Значение, которое принимает предикат MeasureUnitValue, берется из множества {“средний”,”низкий”,”высокий”} или задается в кельвинах или градусах цельсия.
- Illumination – уровень освещения. В качестве аргумента $Equipment$ может иметь LightSystemи потомков. Значение, которое принимает предикат MeasureUnitValue, берется из множества {”низкий”,”высокий”}.
- HaveIncomingCalls – моделирует наличие входящих звонков. В качестве аргумента $Equipment$ может иметь CommunicationSystem и потомков. Значение, которое принимает предикат MeasureUnitValue, берется из множества $\{True, False\}$.

Также было введено отношение SetTemperatureIntention, которое выражает потребность пользователя в установке необходимой температуры в локации. Данный предикат в качестве основных аргументов принимает предикат температуры (Temperature).

3. ПРОГРАММА ДЛЯ ГЕНЕРАЦИИ КОРПУСОВ

В данном разделе описывается структура программы для работы с онтологией Meta AN Ontology для генерации корпусов.

3.1. Физическая и логическая структура программы

В качестве языка программирования используется Python 3.6 со стандартными библиотеками и numpy версии 1.11.3. Для визуализации генерируется xml файл, который экспортируется в программу yEd Graph Editor.

Программа состоит из следующих основных частей: **генератор** или **пул онтологии** (собственно служит для генерации корпуса) и **онтология.**

Каждому *базовому* элементу онтологии Meta AN Ontology соответствует класс в Python, описанный в отдельном файле или файле утилит. Элемент предметной области – это экземпляр некоторого класса. В таблице 3.1 представлены соответствия реализации файлов и базовых объектов онтологии.

Все классы имеют поля, соответствующие описанию элементов онтологии, описанных в предыдущих разделах. Эти поля заполняются как именованные аргументы конструктов класса или передаются списком через параметр *arguments* (важен порядок). Каждый элемент в *arguments* – это список возможных значений, который может принимать данный элемент. Эти значения используются генератором при создании корпуса.

Помимо этого каждый экземпляр имеет следующие поля:

- *semantic_name* – семантическое имя предиката, описывающего экземпляр предметной области.
- *vocabulary* – словарь представлений онтологического объекта в естественном языке.

- natural_expression_config – показывает, какие аргументы имеют представление в естественном языке, а какие нет.
- natural_expression_mask – маска высказывания на естественном языке (задает порядок слов).

is_add_pool – добавлять ли элемент в генератор для последующей генерации.

Таблица 3.1 – соответствия базовых объектов онтологии и файлов.

Моделирование вспомогательных элементов	
$Type$	types.py
$Measure$	measure.py
$NoneObject$	utils_objects.py
Моделирование времени	
$TimeInterval$	time_interval.py
$TemporalEntity$	time_interval.py
$TemporalRelation$	temporal_relations.py
$TemporalRelations$	temporal_relations.py
Моделирование локаций	
$Location$	location.py
Моделирование объектов и отношений	
$Concept$	concept.py
$Object$	object.py
$Relation$	relation.py
$DomainRelation$	domain_relation.py
$TaxonomyRelation$	taxonomy_relation.py

После задания онтологии предметной области вызываются методы генератора, описанного в файле «ontology_pool.py», для создания корпуса.

Для работы с уже готовой онтологией в данный момент есть следующие возможности:

- Генерирование *случайной пары* «логическое выражение, высказывание на естественном языке».
- Генерирование n случайных аналогичных пар (пары могут повторяться).

- Генерирование по n случайных пар для всех элементов онтологии.
- Визуализация логического выражения для сгенерированной пары.

Каждая сгенерированная пара – это в свою очередь класс Python, хранящий необходимую информацию для корпуса. Т. е. такой объект является одной записью в корпусе.

Алгоритм генерации *случайной пары*:

1. Генерируем все возможные *полные* логические выражения для заданного элемента.
    a. Для всех аргументов из списка *arguments*
        i. Для каждого аргумента выбираем его случайное возможное значение.
    b. Формируем *набор* из логических элементов, от которых зависит родительский предикат.
2. Случайно выбираем естественно-языковые представления по одному для каждого из элементов, входящих в *набор*, сформированный для обрабатываемого логического выражения.
3. Генерируем *сокращенное* логическое выражение и соответствующее ему естественно-языковое высказывание.

Приведенный алгоритм имеет общее описание, показывающее суть его работы и результат. Его конкретная реализация использует рекурсивный подход.

Заметим, что на данный момент естественно-языковые высказывания таковыми не являются. На выходе генератора получаются *псевдо* высказывания, содержащие несогласованные слова и не все необходимые предлоги и другие служебные части речи.

Предложенный алгоритм позволяет не хранить в памяти все возможные варианты языковых выражений.

Также отметим, что разработанная логическая модель поддерживает *логический вывод* на уровне таксономий между элементами онтологии. Т. е. она обрабатывает отношений IsA между концептами и предоставляет возможность пользователю задавать отношение предметной области для родительского концепта, которое распространится на его последователей.

Так, например, телефон может находиться во всех комнатах дома.

Пользователь может использовать элемент онтологии при создании других доменных предикатов, а генератор сам подставит подходящие значения пространственного аргумента, определенные в таксономии с помощью отношения IsA.

3.2. Описание сгенерированного корпуса

В данном разделе описаны примеры работы программы и основные результаты, полученные в ходе работы.

Полученный корпус содержит логические выражения и их языковые соответствия для элементов предметной области HDS, описанных в разделе «Home domain ontology»: локации, системы оборудования, временные сущности, состояния.

Рассмотрим пример записи в корпусе для описанного ранее элемента предметной области. На рисунке 3.1. представлена строковая запись логического выражения и его *псевдо* естественно-языковой эквивалент. Данное высказывание соответствует высказыванию «идет передача с названием “наедине со всеми” по tv начиная со времени четыре часа ноль мин. по пять часов ноль мин». Визуализация визуализации этой метавершины (логического предиката предметной области) представлена на рисунке 3.2. Ромбами обозначены аргументы предикатов, стрелочками –

таксономические зависимости. Подразумевается, что отсылки к интервалам идут через TimePoint. Данное отношение было подробно разобрано в разделе «2.3.2. Моделирование телепрограммы и ее составляющих». Главный предикат отношения – ShowProgramm, имеет три аргумента: TVProgramm (какая программа показывается), TVProgrammTimeInterval (когда показывается) и TV (где показывается). Для временного интервала в языке заданы только начало и конец, о чем говорит наличие элемента NoneObject на позиции аргумента длительности (Duration). Сам же интервал является сложной временной сущностью (Clock) и объединяет в себе часы (HourTimeInterval) и минуты (MinuteTimeInterval). Заметим также, что на рисунке видна иерархичность системы: на верхнем уровне идут объекты и отношения (ShowProgramm и TVProgramm), далее локация (TV), ниже временные сущности и интервалы, и далее вспомогательные элементы (меры, моделирующие шкалы для интервалов). Данная иерархичность полностью соответствует описанной модели.

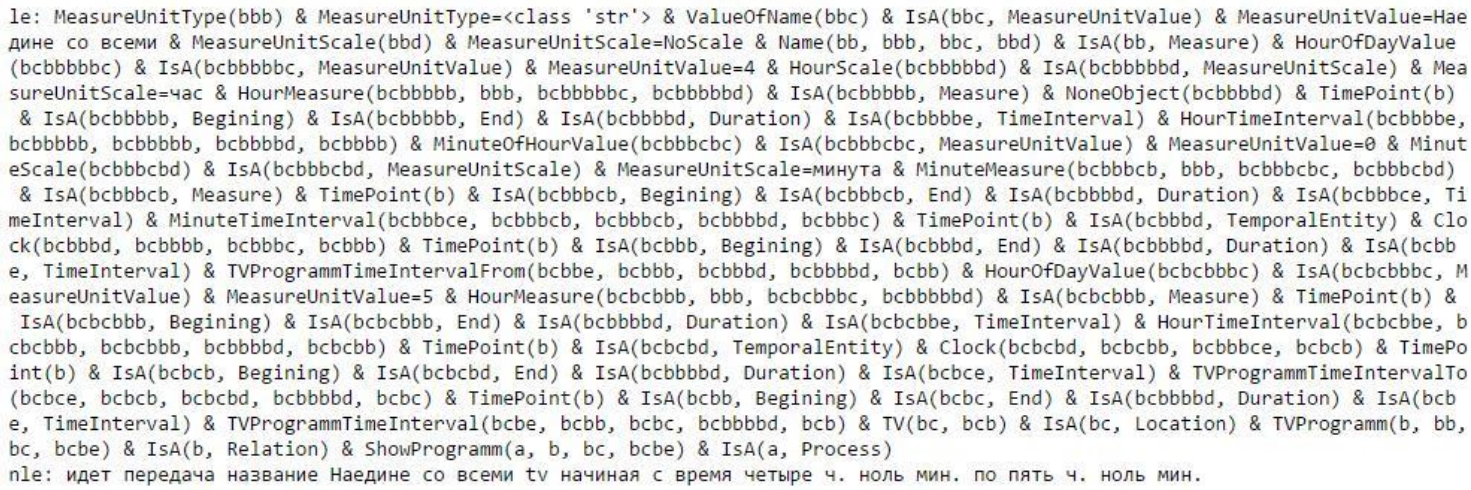

Рисунок 3.1 – *случайная пара* из корпуса для предиката.

В итоге сгенерировано 54043 логических выражения, описывающих предметную область, которым соответствует 950100 высказываний на *псевдо* естественном языке. Большую часть из них занимают высказывания, соответствующие описанию

временных выражений. Данный корпус может быть использован для обучения диалоговой системы распознавания намерений пользователя при управлении системами *умного дома*. Для применения корпуса достаточно нормализовать высказывания на естественном языке. Конечно, введенная онтологическая модель описывает только часть предметной области, но которая является одной из *важных* ее составляющих.

Фрагмент корпуса, содержащий высказывания на *псевдо* естественном языке и соответствующих им согласованных выражений приведен в таблице 3.2.

В дальнейшем для согласования высказываний на псевдо естественном языке будут использоваться нейронные сети.

Таблица 3.2 – сгенерированные высказывания из корпуса и соответствующие им естественно языковые эквиваленты.

Высказывание на *псевдо* естественном языке	**Высказывание на естественном языке**
идет программа имя Новости тв с время девять час 0 минута заканчивая в девять час двадцать минута	идет программа с именем «Новости» по тв со времени девять часов 0 минут заканчивая в девять часов двадцать минут
идет телепрограмма Контрольная закупка телевизор с девять час двадцать минута заканчивая в девять час пятьдесят минут	идет телепрограмма «Контрольная закупка» по телевизору с девяти часов двадцати заканчивая в девять часов пятьдесят минут
показывается программа Жить здорово! тв с девять ч. пятьдесят мин. по десять ч. пятьдесят пять мин.	показывается программа «Жить здорово!» по тв с девяти ч. пятидесяти мин. по десять пятьдесят пять мин.
пятьдесят пять мин.	пятьдесят пять мин.
начиная с 21 ч. 35 мин.	начиная с 21 ч. 35 мин.
заканчивая в время 23 ч. 40 мин.	заканчивая во время 23 ч. 40 мин.
с 21 ч. тридцать пять мин. заканчивая в двадцать три ч. 40 мин.	с 21 ч. тридцати пяти мин. заканчивая в двадцать три ч. 40 мин.
уровень обогрева система отопления зал наибольший время 18 час ноль минута	уровень обогрева системы отопления в зале наибольший во время 18 часов ноль минут
пятнадцать час пятнадцать минута	в пятнадцать часов пятнадцать минут
двадцать один час	двадцать один час
двадцать два час	двадцать два часа
гостиная время 5 ч. ноль мин.	гостиная во время 5 ч. ноль мин.
сделать температура отопление 24 градус цельсия туалет	сделать температуру отопления 24 градуса цельсия в туалете
установить температура двадцать один °C кухня	установить температуру двадцать один °C в кухне
установить температура система обогрева 25 градус цельсия кухня сейчас	установить температуру системы обогрева 25 градус цельсия на кухне сейчас

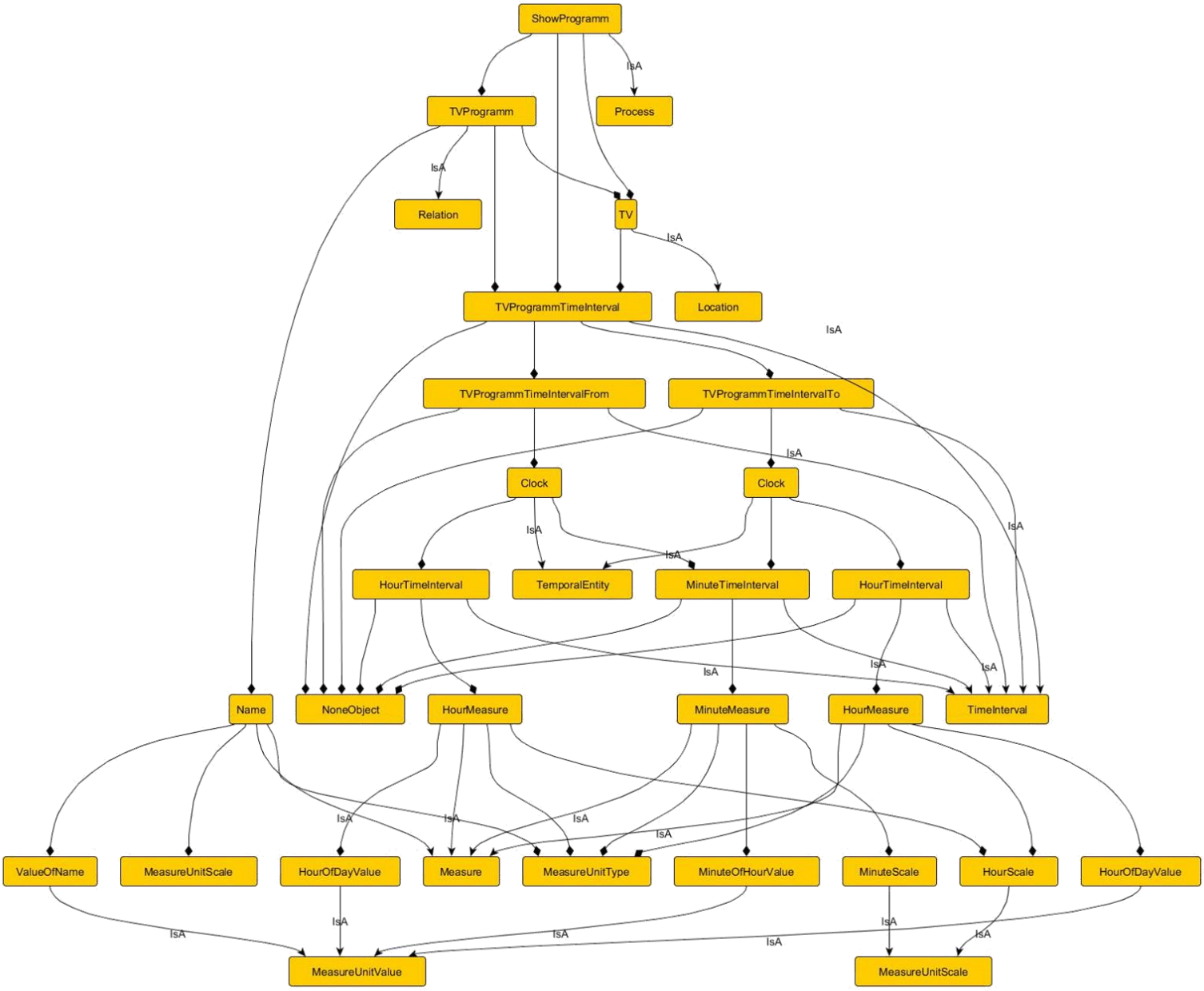

Рисунок 3.2 – визуализация логического выражения для предиката.

ЗАКЛЮЧЕНИЕ

В рамках данной работы получены следующие результаты.

1. Проведено исследование в области диалоговых систем и онтологий.

2. На основе исследования были сделаны выводы о структурных решениях, предъявляемые онтологиям, которые используются в диалоговых системах.

3. Построена математическая модель AN Ontology для моделирования структуры онтологий, на основе которых будут строиться модели предметных областей. Предложенная модель является многоуровневой и использует расширение понятия гиперграфа – метаграф, который лежит в ее основе.

4. Разработана онтология общая онтология Meta AN Ontology для моделирования различных предметных областей диалоговых систем. Предложенная онтология содержит базовые элементы (предикаты), спомощью которых можно описывать объекты из реального мира. Отличием онтологии от аналогов состоит в том, что доменные предикаты имеют обязательные аргументы времени и пространства. Также для моделирования времени введено понятие бесконечно малой точки, принадлежащей временному интервалу. Данная конструкция позволяет упростить моделирование одновременных событий, а также соотнесение событий по времени.

5. Создана программа, которая использует Meta AN Ontology и позволяет получать корпус пар «логическое выражение – текст» на базе созданной онтологии предметной области. Каждый элемент предметной области в программе физически задается с помощью объекта python класса. Программа поддерживает использование таксономий и предоставляет возможность визуализации семантической структуры элемента.

6. Реализована часть онтологии предметной области Home Domain System (HDS).

7. Сгенерирован корпус пар «логическое выражение – высказывание на *псевдо* естественном языке», который в дальнейшем будет использоваться для обучения алгоритмов машинного обучения и создания диалоговой системы.

Данная работа является началом для дальнейшей разработки диалоговых систем, а также содержит в себе новые подходы к моделированию предметных областей с помощью онтологий. Она будет развита по следующим направлениям:

- *Основным* и *первостепенным* пунктом является обучение алгоритмов машинного обучения для согласования полученных высказываний на *псевдо* естественном языке. Это позволит генерировать полноценные естественно языковые высказывания и упростит дальнейшее использование корпуса в разработке диалоговых систем.
- Следующее направление работы – это развитие введенной модели Meta AN Ontology. Важной частью этого пункта являет дополнение модели сложными предикатами, описывающими такие языковые явления как анафора, эллипсис и др.
- Разработка диалоговой системы на основе сгенерированного корпуса и оценка эффективности ее работы в конкретной задаче (task oriented). Сравнение с уже существующими аналогами корпусов для *task oriented* диалоговых систем (например, babi корпус [89]).
- Поддержка логического вывода в Meta AN Ontology.

СПИСОК ИСПОЛЬЗОВАННОЙ ЛИТЕРАТУРЫ

1. T. H. Bui. Multimodal Dialogue Management: State of the Art. // Centre for Telematics and Information Technology, University of Twente, Enschede, Netherlands, Tech. Rep. TR-CTIT- 06-01. – 2006.
2. Guoying Liu. A task ontology model for domain independent dialogue management. // IEEE International Conference on Virtual Environments Human-Computer Interfaces and Measurement Systems (VECIMS) Proceedings. – 2012.
3. M.F. McTear. Spoken Dialogue Technology: Enabling the Conversational User Interface. // ACM Computing Surveys. – 2002. vol. 34, no. 1, pp. 90– 169.
4. E. Levin, R. Pieraccini, and W. Eckert. // A stochastic model of human-machine interaction for learning dialogue strategies. IEEE Transactions on Speech and Audio Processing. – 2000. – vol. 8, no. 1, pp. 11-23.
5. J. D. Williams and S. Young. Partially observable Markov decision processes for spoken dialog systems. // Computer Speech and Language. – 2007. – vol. 21, pp. 393−422.
6. C.Wai, H. M. Meng, R. Pieraccini. Scalability and portability of a belief network based dialog model for different application domains. In Proceedings of the first international conference on Human language technology research. // Association for Computational Linguistics. – 2001.
7. T. R. Gruber. A Translation Approach to Portable Ontology Specifications. // Knowledge Acquisition. – 1993. – vol. 5, no. 2, pp. 199-220.
8. N. F. Noy and D. L. McGuinness. Ontology development 101: A guide to creating your first ontology. // Stanford Knowledge Systems Laboratory Technical Report KSL-01-05 and Stanford Medical Informatics Technical Report SMI-2001-0880. – 2001.

9. Annika Flycht-Eriksson. AnOntology: A Framework for Use of Ontologies in Dialogue Systems. // Department of Computer and Information Science. – 2007.

10. Chandrasekaran B., J. R. Josephson, V. R. Benjamins. What are ontologies, and why do we need them? // IEEE Intelligent Systems. – 1999. – vol. 14, no. 1, pp. 20-26.

11. Cycorp. The Syntax of Cycl [Электронный ресурс]. Режим доступа: \www/. // URL: http://www.cyc.com/syntax-cycl/. – 2009.

12. Enrico Franconi. Description Logics for Conceptual Design, Information Access, and Ontology Integration: Research Trends. // Networks journal of the philosophy of Artificial Intelligence and Cognitive Science, special issue on the Semantic Web. – 2003. – vol. 1, pp. 1–23.

13. Harry Delugach. Common Logic [Электронный ресурс]. Режим доступа: \www/. // ISO/IEC 24707. URL: http://cl.tamu.edu/. – 2003.

14. Michael R. Genesereth Richard E. Fikes. Knowledge Interchange Format Version 3.0 Reference Manual. // Computer Science Department Stanford University Stanford, California 94305. – 1992. – p. 68.

15. W3C. OWL Web Ontology Language [Электронный ресурс] режим доступа: \www/. // URL https://www.w3.org/TR/owl-features/. – 2009.

16. Annika Flycht-Eriksson, Arne Jonsson. Some empirical findings on dialogue management and domain ontologies in dialogue systems – Implications from an evaluation of BIRDQUEST. // SIGDIAL Workshop 2003. – 2003. vol. pp. 158-167.

17. Annika Flycht-Eriksson. Design and Use of Ontologies in Information-providing Dialogue Systems. // Dissertation, Linköping Studies in Science and Technology, Thesis No. 874, School of Engineering at Linköping University. – 2004. – 205 p.

18. Bennacef S., L. Devillers, S. Rosset, L. Lamel. Dialog in RAILTEL telephone-based system. // Processing of the International Conference on Spoken Dialogue Language Processing, ICSLP'96, Philadelphia, USA. – 1996 – vol. 1, pp. 550-553.

19. Carlson R., S. Hunnitcut. Generic and domain-specific aspects of the Waxholm NLP and dialog modules. ICSLP'96, Philadelphia, USA. –1996. –vol. 2, pp. 677-680.

20. Heisterkamp P., S. McGlashan, N. Yond. Dialogue semantics for a spoken dialogue system. // Processing of the International Conference on Spoken Dialogue Language Processing, ICSLP'92, Banff, Canada. – 1992.

21. Douglas Walton. // A dialogue model of belief, ISSN. – 2009. vol. 1, pp. 1946-2166.

22. Alexandersson, J., E. Maier, N. Reithinger. A robust and efficient three-layered dialogue component for speech-to-speech translation system. // e Proceedings of the EACL. Technical Report 50, DFKI GmbH. – 1994. – vol. 1, p. 17.

23. Quantz J. M. Gehrke, U. Kussner, B. Schmitz. The VERBMOBIL domain model version 1.0. // Technical Report 29, Technical University, Berlin. – 1994. – vol. 1, p. 78.

24. M. Gatius and M. Gonzalez. Using Ontologies for Improving the Communication Process in a Dialogue System. // Talp Research Center of Technical University of Catalonia, Barcelona. – 2005.

25. Guoying Liu. A task ontology model for domain independent dialogue management. // Thesis. Windsor, Ontario, Canada. – 2012. – p. 83.

26. D. Sonntag and M. Romanelli. A multimodal result ontology for integrated semantic web dialogue applications". // Proceedings of the 5th Conference on Language Resources and Evaluation (LREC). – 2006. – vol. 1, pp. 288-294.

27. W3C. EMMA: Extensible MultiModal Annotation markup language [Электронный ресурс]. Режим доступа: \www/. // URL: http://www.w3.org/TR/2009/REC-emma-20090210/. – 2009.

28. Neil Day, José M. Martínez. Introduction to MPEG-7 (v3.0). // INTERNATIONAL ORGANISATION FOR STANDARDISATION

29. ORGANISATION INTERNATIONALE DE NORMALISATION. – 2001. – vol. 1, p. 10.

30. T. W. Bickmore, D. Schulman, C. L. Sidner. A reusable framework for health counseling dialogue systems based on a behavioral medicine ontology. // Journal of Biomedical Informatics. – 2011. – vol. 44, no. 2, pp. 183-197.

31. W3C. RDF Schema [Электронный ресурс]. Режим доступа: \www/. // URL: http://www.w3.org/TR/2014/REC-rdf-schema-20140225/. – 2014.

32. J. P. Pardal and N. J. Mamede. Starting to Cook a Coaching Dialogue System in the Olympus framework. // Proceedings of the Paralinguistic Information and its Integration in Spoken Dialogue Systems Workshop. – 2011. – vol. 1, pp. 255-267.

33. W3C. ANSI/CEA-2018 Task Model Description standard [Электронный ресурс]. Режим доступа: \www/. // URL: https://www.w3.org/2005/Incubator/model-based-ui/wiki/ANSI/CEA-2018. – 2009.

34. Allen J., D. Byron, M. Dzikovska, G. Ferguson, L. Galescu, A. Stent. Toward conversational human-computer interaction. // AI Magazine. – 2001. – vol. 22, pp. 27- 38.

35. D. Bohus and I. Rudnicky. The RavenClaw Dialog Management Framework: Architecture and Systems. // Computer Speech and Language. – 2009. – vol. 23, pp. 332-361.

36. X. Zhang. An interactive approach of ontology-based requirement elicitation for software customization. // Master’s thesis, School of

Computer Science, University of Windsor, Windsor, Canada. – 2011. p. 87.

37. S. Young, M. Gasic, S. Keizer, F. Mairesse, J. Schatzmann, B. Thomson, K. Yu. The hidden information state model: a practical framework for POMDP-based spoken dialogue management. // Computer Speech and Language. – 2009. – vol. 24, no. 2, pp. 150–174.

38. T. Heinroth, D. Denich, and W. Minker. A Multitasking Approach to Adaptive Spoken Dialogue Management. // Universal Access in HCI, Part IV, HCII 2011, LNCS 6768, C. Stephanidis, Ed. – 2011. – pp. 42 – 51.

39. J. van Oijen, W. van Doesburg, F. Dignum. Goal-Based Communication Using BDI Agents as Virtual Humans in Training: An Ontology Driven Dialogue System. // Agents for Games and Simulations II, Frank Dignum, Ed. Berlin: Springer. – 2011. – pp. 38-52.

40. Gaëtan Pruvost, Achilles Kameas, Lambrini Seremeti, Tobias Heinroth, Wolfgang Minker. Combining agents and ontologies to support task-centred interoperability in Ambient Intelligent Environments. ISDA 2009. – 2009. – pp. 55-60.

41. Stefan Ultes,Florian Nothdurft,Tobias Heinroth,Wolfgang Minker. Next Generation Intelligent Environments: Ambient Adaptive Systems. – 2011. – vol. 2. p. 346.

42. Tobias Heinroth, Achilles Kameas, Christian Wagner, and Yacine Bellik . A Proposal for Realizing Adaptive Ambient Ecologies through an ontology-based Multi-Agent System. // Intelligent Agents, 2009. IA '09. IEEE Symposium on. – 2009. p. 9.

43. Meritxell Gonz`alez Berm´udez. Dialogue Management for multilingual communication through different channels. // Thesis. Department of Computer Languages and Systems Technical University of Catalonia. – 2007. – p. 67.

44. David Milward, Martin Beveridge. Ontology-Based Dialogue Systems. – 2003. – p. 8.

45. Гипоним и гипероним. standard [Электронный ресурс]. Режим доступа: \www/. // URL: https://ru.wikipedia.org/wiki/%D0%93%D0%B8%D0%BF%D0%BE%D0%BD%D0%B8%D0%BC_%D0%B8_%D0%B3%D0%B8%D0%BF%D0%B 5%D1%80%D0%BE%D0%BD%D0%B8%D0%BC. – 2017.

46. Martin Beveridge, David Milward. Combining Task Descriptions and Ontological Knowledge for Adaptive Dialogue.// EU-funded HOMEY project (Home Monitoring through an Intelligent Dialogue System. – 2003. – p. 8.

47. Joana Paulo Pardal. Dynamic Use of Ontologies in Dialogue Systems. // Department of Information Systems and Computer Engineering Instituto Superior Tecnico, Technical University of Lisbon ´ Lisbon, Portugal. – 2007. – p. 4.

48. Wolfgang Wahlster. Dialogue Systems Go Multimodal: The SmartKom Experience. // Wahlster, W. (ed.): SmartKom: Foundations of Multimodal Dialogue Systems. Cognitive Technologies Series, Heidelberg, Germany: Springer. – 2006. – vol. 1, pp. 3-27.

49. Gurevych I., R. Porzel, E. Slinko, N. Pfleger, J. Alexandersson, S. Merten. Less is more: Using a single knowledge representation in dialogue systems. // Processing of the HLT-NAACL'03 Workshop on Text Meaning. Edmonton, Canada. – 2003. – vol. 1, pp. 14-21.

50. R. Porzel, N. Pfleger, S. Merten, M. Loeckelt, I. Gurevych, R. Engel, Alexandersson. More on less: Further applications of ontologies in multi-modal dialogue systems. // Processing of the IJCAI'03 Workshop on Knowledge and Reasoning in Practical Dialogue Systems, Acapulco, Mexico. – 2003.

51. Horrocks, D. Fensel, J. Broekstra, S. Decker, M. Erdmann, C. Goble1 , F. van Harmelen, M. Klein, S. Staab, R. Studer, and E. Motta. The Ontology Inference Layer OIL. – 2000. – vol. 1, p. 54.

52. Marc Moens. Mark Steedman. TEMPORAL ONTOLOGY AND TEMPORAL REFERENCE. // Computational Linguistics. –1988. – vol.14, pp. 88-100.

53. Dirk Buhler, Wolfgang Minker, Jochen Hauvler, Sven Kruger. The SmartKom mobile multi-modal dialogue system. – 2006.

54. Blaylock N., J. Allen, G. Ferguson. Synchronization in an asynchronous agent-based architecture for dialogue systems. // Processing of the 3rd SIGdial Workshop on Discourse and Dialog, Philadelphia, USA. – 2002. – vol. 2, pp. 1-10.

55. Guillermo P´erez, Gabriel Amores, Pilar Manch´on, Fernando G´omez y Jes´us Gonz´alez. Integrating OWL Ontologies with a Dialogue Manager. Natural language processing. N. 37 – 2007. – pp. 153-160.

56. David Traum, Johan Bos, Robin Cooper Staffan Larsson Ian Lewin Colin Matheson Massimo Poesio. A model of dialogue moves and information state revision. // Trindi Project. Department of Linguistics Goteborg University. – 1999. – vol. 2.1, p. 86.

57. Gabriel Amores, Jos´e Francisco Quesada. Dialogue moves for natural command languages. // Procesamiento del Lenguaje Natural. – 2001. – vol. 27, pp. 89–96.

58. Gabriel Amores, Jos´e Francisco Quesada. Episteme. // Procesamiento del Lenguaje Natural. – 1997. – vol. 21, pp. 1–16.

59. Tilman Beckerm, Peter Poller, Nate Blaylock Staffan Larsson Oliver Lemon Guillermo P´erez Jan Schehl. Talk status report: Software infrastructure. // Talk Project. – 2006. – vol. 2.

60. Pilar Manch´on, Carmen del Solar, Gabriel Amores Guillermo P´erez.

61. Generating Multilingual Grammars from OWL Ontologies. – 2006. – p. 10.

62. M. Gatius. Using an ontology for guiding NL interaction with knowledge based systems. // PhD thesis, Polytechnic University of Catalonia. – 2001. – p. 224.

63. M. Gatius and H. Rodr´ıguez. Natural language guided dialogues for accessing the web. // Text, Speech and Dialogue Published. Natural Language Guided Dialogues for Accessing the Web. – 2002. – vol. 2448, pp. 373-380.

64. D. Crocker, Ed. Augmented BNF for Syntax Specifications: ABNF. // RFC Editor , United States. – 1997.

65. W3C. Voice Extensible Markup Language (VoiceXML) Version 2.0 [Электронный ресурс]. Режим доступа: \www/. // URL: https://www.w3.org/TR/voicexml20/. – 2004.

66. W3C. Extensible Markup Language (XML) [Электронный ресурс]. Режим доступа: \www/. // URL: https://www.w3.org/XML/. – 2006. Chris Partridge. Business Objects: Re-Engineering for Re-Use. // REV-ENG Consulting. – 2000. – vol. 2, p. 614.

67. Moeiz Miraoui, Sherif El-etriby, Chakib Tadj, Abdulbasit Zaid Abid. Ontology-Based Context Modeling for a Smart Living Room. // Proceedings of the World Congress on Engineering and Computer Science 2015, San Francisco, USA. – 2015. – vol. 1, p. 6.

68. Benjamin Harbelot. Helbert Arenas. Christophe Cruz. Continuum: A spatiotemporal data model to represent and qualify filiation relationships. // 4th ACM SIGSPATIAL International Workshop on GeoStreaming (IWGS) 2013, Orlando : United States. – 2013. – vol. 1, pp. 76-85.

69. James F. Allen. Towards a General Theory of Action and Time. // Artificial Intelligence. – 1984. – vol. 23, pp. 123–154.

70. W3C. Time Ontology in OWL [Электронный ресурс]. Режим доступа: \www/. // URL: https://www.w3.org/TR/owl-time/. – 2017.

71. Matthew Perry, John Herring. OGC GeoSPARQL - A Geographic Query Language for RDF Data. // Open Geospatial Consortium. – 2012. – vol. 1, p. 75.

72. Gruber, T. and Olsen, G. An ontology for engineering mathematics. // Fourth International Conference on Principles of Knowledge Representation and Reasoning. Morgan Kaufmann, Bonn, Germany. – 1994. – pp. 1-18.

73. Schadow, G., McDonald, C., Suico, J., Fohring, U., and Tolxdorff, T. Units of measure in clinical information systems. Journal of the American Medical Informatics Association. – 1999. – vol. 6, no, 2 pp. 151–162.

74. H. Chen, F. Perich, T.W. Finin, A. Joshi. Soupa: Standard Ontology for Ubiquitous and Pervasive Applications. // MobiQuitous, IEEE Computer Society. – 2004. – vol. 1. pp. 258-267.

75. D. Gaur, A. Shastri, R. Biswas and D. Seema Gaur Vague Metagraph. // International Journal of Computer Theory and Engineering. – 2009. – vol.1, no. 2, pp.1793–8201.

76. Глазунов Ю.Т. Графы и системы. Овеществление идеального // Вестник Балтийского федерального университета им. И. Канта. – 2010. – № 10. – с.132-150.

77. Астанин С. В., Жуковская Н. К. Управление бизнес-процессами на основе их моделирования нечеткими ситуационными сетями // УБС. – 2012. – №37. – с. 145-163.

78. W3C RDF Working Group. RDF Primer. W3C Recommendation [Электронный ресурс]. Режим доступа: \www/. // URL: http://www.w3.org/TR/rdf-primer/. – 2004.

79. Davenport, J. and Naylor, W. Units and dimensions in OpenMath. Technical report, OpenMath. [Электронный ресурс]. Режим доступа: \www/. // URL: http://www.openmath.org/documents/Units.pdf. 2003.

80. Leal, D. and Schröder, A. RDF vocabulary for physical properties, quantities and units. Technical report, ScadaOnWeb. URL: http://www.s-ten.eu/scadaonweb/NOTE-units/2002-08-05/NOTE-units.html. – 2002.

81. Raskin, R. and Pen, M. Knowledge representation in the Semantic Web for Earth and Environmental Terminology (SWEET). // Computers & Geosciences. – 2005. – vol. 31, pp. 1119–1125.

82. Pinto, H. and Martins, J. Revising and extending the Units of Measure "subontology". // Proceedings of the Workshop IEEE Standard Upper

83. Ontology, at IJCAI'2001. AAAI Press, Seattle, Washington. – 2001. Rijgersberg, H., Wigham, M., and Top, J. L. How semantics can improve engineering processes: A case of units of measure and quantities. // Advanced Engineering Informatics. – 2011. – vol. 25, no. 2, pp. 276–287.

84. А. А. Зыков. Гиперграфы. // УМН. том 29. – 1974. – № 6, с. 89–154.

85. Theodore Sider. Quantiers and Temporal Ontology. // Mind 115. – 2006. – pp. 75-97.

86. Cui Tao, PhD Wei-Qi Wei, MM Harold R. Solbrig, BS Guergana Savova, PhD Christopher G. Chute, MD, DrPH. CNTRO 2.0: A Semantic Web Ontology for Temporal Relation Inferencing. // Clinical Narratives. AMIA 2011 Symposium Proceedings. – 2011. – pp. 64–68.

87. S. Pantsar-Syvaniemi,K. Simula, E. Ovaska. Context-awareness in smart spaces. // IEEE Symposium on Computers and Communications (ISCC) Riccione. – 2010. – vol. 1, pp. 1023-1028.

88. Jason Weston, Antoine Bordes, Sumit Chopra, Alexander M. Rush, Bart van Merrienboer, Armand Joulin & Tomas Mikolov. TOWARDS AI-COMPLETE QUESTION ANSWERING : A SET OF PREREQUISITE TOY TASKS. // Under review as a conference paper at ICLR 2016. arXiv. – 2015. – vol. 10, pp. 1-14.

89. D. Bohus, A. Raux, T. K. Harris, M. Eskenazi and A. I. Rudnicky. Olympus: an opensource framework for conversational spoken language interface research. // NAACL-HLT '07: Proceedings of the Workshop on Bridging the Gap: Academic and Industrial Research in Dialog Technologies. – 2007. – vol. 1, pp. 32-39.

90. H. Chen, T. Finin, A. Joshi. An Ontology for Context-aware Pervasive Computing Environments. Knowledge Engineering Review, Springer Verlag. – 2003. – vol. 18, pp. 197-207.

91. T. Gu, X. H. Wang, H.K. Pung, D.Q. Zhang. An Ontology-based Context Model in Intelligent Environments". // Proceedings of Communication Networks and Distributed Systems Modeling and Simulation Conference, San Diego, California, January. – 2004.

92. Abdulrazak, B., Chikhaoui, B., Gouin-Vallerand, C. and Fraikin, B. A standard ontology for smart spaces. // Int. J. Web and Grid Services. –2010. – vol. 6, no. 3, pp.244–268.

93. M. Mohsin Saleemi, Natalia Díaz Rodríguez, Johan Lilius, Ivan Porres. 2011. A Framework for Context-Aware Applications for Smart Spaces. // Sergey Balandin, Yevgeni Koucheryavi, Honglin Hu (Eds.), Proceedings of The 4th conference on Smart Spaces ruSMART, Lecture Notes in Computer Science, Springer. – 2011. – pp. 14–25.

94. Ming Li. Ontology-Based Context Information Modeling for Smart Space. // 10th IEEE International Conference on Cognitive Informatics & Cognitive Computing (ICCI*CC). Banff, AB. – 2011. – pp. 278-283.

95. Haesung Lee and Joonhee Kwo. Ontology Model-based Situation and Socially-Aware Health Care Service in a Smart Home Environment International Journal of Smart Home. – 2013. – vol.7, no. 5, pp. 239-250.

96. Zachary Wemlinger, Lawrence Holder. The COSE Ontology: Bringing the Semantic Web to Smart Environments. // ICOST'11 Proceedings of the 9th international conference on Toward useful services for elderly and people with disabilities: smart homes and health telematics. – 2011. – pp. 205-209.

97. Moji Wei, Jianliang Xu, Hongyan Yun, Linlin Xu. Ontology-Based Home Service Model. COMPUTER SCIENCE AND INFORMATION SYSTEMS. – 2012. – vol. 9, no. 2, pp. 813-838.

98. Wan-rong Jih, Jane Yung-jen Hsu. Agent-Based Context-Aware Service in a Smart Space Agent-Based Ubiquitous Computing, Atlantis Ambient and Pervasive Intelligence. 2010. – vol. 1, pp 131-146.

99. Xinhua Zhu, Yaxin Yu, Yuming Ou, Dan Luo, Chengqi Zhang, Jiahang Chen. System Modeling of a Smart-Home Healthy Lifestyle Assistant // Agents and Data Mining Interaction Lecture Notes. Computer Science. – 2012. – vol. 7607, pp. 65-78.

100. C. Li, L. Sun, X. Hu. A context-aware lighting control system for smart meeting rooms. // Systems Engineering Procedia. Information Engineering and Complexity Science – Part II. – 2012. – vol. 4, pp. 314–323.