Разработка генератора платформенно-зависимой модели для системы проектирования виртуальных помощников

Никита Допиро[1]

[1] *Иркутский национальный исследовательский технический университет, ул. Лермонтова 83, г. Иркутск, 664033, Россия*

Аннотация

В данной статье будет рассмотрена разработка генератора платформенно-зависимой модели для системы проектирования виртуальных помощников на базе JUST AI с использованием платформы JAICP с помощью MDA подхода.

Ключевые слова

MDA, виртуальный помощник, чат-бот, JAICP, искусственный интеллект

1. Введение

Искусственный интеллект уже давно стал неотъемлемой частью нашей повседневной жизни. Одним из наиболее популярных применений ИИ являются виртуальные помощники – это сервис, способный взаимодействовать с людьми на основе уже имеющегося сценария, выполнять различные задачи и предоставлять необходимую информацию. В большинстве современных устройствах уже есть голосовой помощник, способный предоставить информацию о погоде, вызвать такси, заказать еду.

Алгоритм работы виртуального помощника [1], [2] включает в себя следующие этапы: принятие и распознавание голосового ввода; преобразование его в текстовый формат; обработка уже имеющегося текста на естественном языке; определение намерений пользователя; выявление задачи, ее параметров и определение дальнейших шагов решения; выполнение этапов решения выявленной задачи; создание ответа в виде диалога и представление результата пользователю.

Для реализации выше указанного алгоритма современные виртуальные помощники должны обладать определенным набором необходимых навыков: распознавать и понимать естественный язык (NLP), обеспечивающие распознавание и генерацию речи (голоса и текста); поддерживать многоканальную интеграцию; сохранять и использовать контекст диалога с пользователем; диалоговые элементы управления, относящиеся к системе управления диалогами; иметь доступ к БД и брать оттуда необходимую информацию для пользователя; компоненты для решения задач и принятия решений.

Актуальность проведения научных исследований в данном направлении подтверждается растущим спросом на использование виртуальных помощников (Сбер “Салют”, “Amazon Alexa”, Microsoft “Cortana”), а также увеличением разнообразия программных средств для их разработки. Применение виртуальных помощников широко распространено в компаниях и предприятиях для автоматизации клиентского обслуживания, и не только, которые в свою очередь направленны на оптимизацию производственной деятельности пользователя.

Данная работа вносит вклад в направления автоматизации разработки BA в данной статье используется MDA подход. Model Driven Architecture (MDA) — модельно-ориентированный

6th International Workshop on Information, Computation, and Control Systems for Distributed Environments (ICCS-DE 2024), July 01–05, 2024, Irkutsk, Russia
EMAIL: nikitadopiro@gmail.com
ORCID:0009-0007-2621-9069

ICCS-DE 2024 Workshop Proceedings
DOI: 10.47350/ICCS-DE.2024.12

подход к разработке программного обеспечения. Принцип работы этого подхода состоит в создание абстрактной метамодели управления и обмена метаданными, то есть моделями. И в последующем задании способов ее преобразования в поддерживаемые технологии программирования, такие как C++, Java и другие. На примере [3], были сформулированы модели MDA и их трансформации (рис. 1).

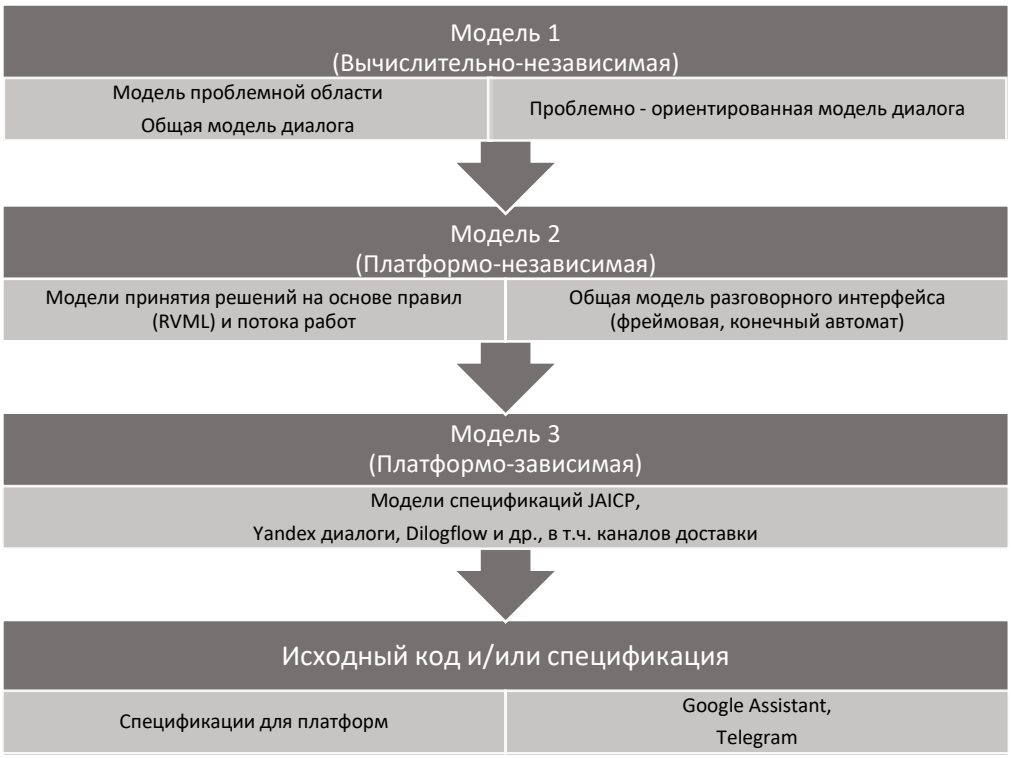

Рисунок 1: Концепция средства: модели MDA и их трансформации

2. Платформенно-зависимая модель JAICP

В качестве платформы разработки BA выбрана JAICP.Just AI Conversational Platform (JAICP Ultimate) — это комплексное программное решение для разработки, внедрения, эксплуатации и поддержки интеллектуальных чат-ботов и голосовых помощников. Платформа использует передовые алгоритмы анализа естественного языка и эффективно сочетает подходы на основе правил и машинного обучения.

Согласно используемой MDA методологии была разработана платформенно-зависимые модели JAICP на языке Java.Далее будут рассмотрены основные классы и их взаимосвязи.

Диаграмма классов (рис. 2) описывает работу платформы JAICP, а также наглядно показывает его составляющие компоненты. Таким образом созданный BA на данной платформе поддерживает многоканальность (Phone channels, Customer engagement platforms, Inbound channels), обработку естественного языка, благодаря встроенному NLU API (Caila, Dialogflow), что обеспечивает обработку сущностей, на базе платформы имеется два вида сущностей-пользовательские сущности (User entity) и системные сущности (System entity), а также поддерживает работу с переходами между состояниями, по принципу конечного автомата.

Состояние (рис. 3) может хранить контекст, структуру, которая представляет текущий контекст обработки запроса. JAICP также позволяет сохранять произвольные данные о текущей сессии. После выполнения реакций бота содержимое сессии сохраняется во внутренней базе данных и хранится между запросами пользователя. При завершении сессии эти данные очищаются. Переход между состояниями осуществляется благодаря тегам активации (н-р теги q или q!, intent или intent!) и реакций (н-р теги go или go!).JAICP позволяет гибко настроить выполнения тегов активации, разметить область действия (н-р параметр onlyThisState),или указать определённый переход в определённое состояние (н-р параметр ToState). Также в JAICP

имеется гибкая настройка обработки намерений, благодаря встроенному NLU API, при активации тегов intent или intent!

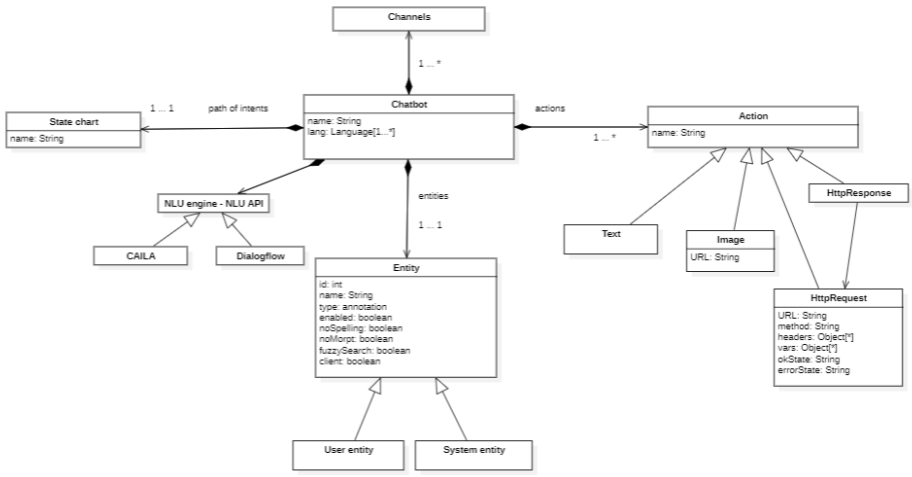

Рисунок 2: Диаграмма классов основных элементов JAICP

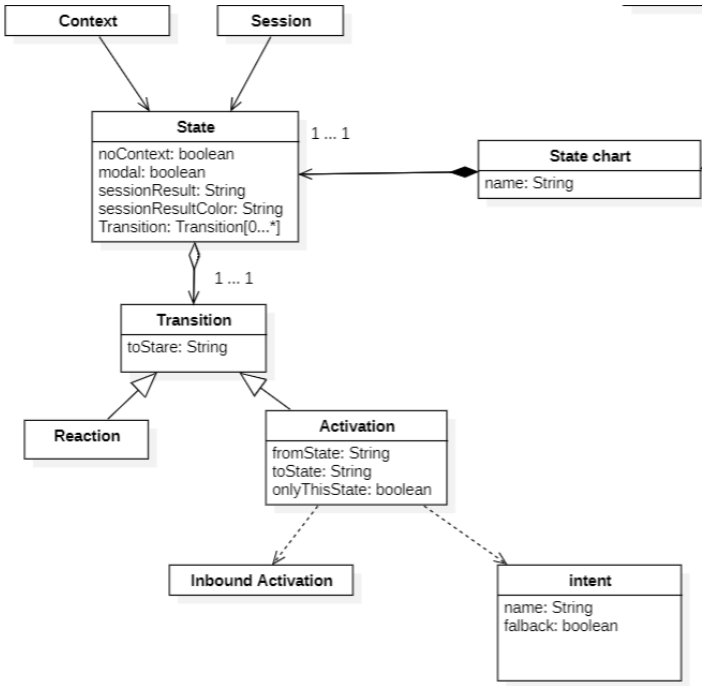

Рисунок 3: Диаграмма классов, описывающая взаимосвязь переходов и состояний

Диаграмма классов (рис. 4) описывает то, как обрабатываются намерения и сущности с заполнением слотов (слот-филлинг), и их компоненты. Обработка происходит следующем образом: фраза, которая была распознанная из общего контекста, благодаря заполненным паттернам и/или тренировочным фразам, представляет собой активацию для перехода в другое состояние, также в этом же контексте, где была распознана фраза, обрабатываются сущности, при распознавании которых заполняются определённые слоты. Заполненные слоты имеют следующие параметры: название — имя слота; сущность — системная, либо пользовательская сущность, задающая тип данных, которые попадут в слот; required — параметр активен, если слот обязателен для заполнения, чтобы обработать интент; массив — если параметр активен, то в слот помещаются все сущности данного типа, оформленные как массив; questions — указываются вопросы, которые задаются в процессе уточнения незаполненных слотов.

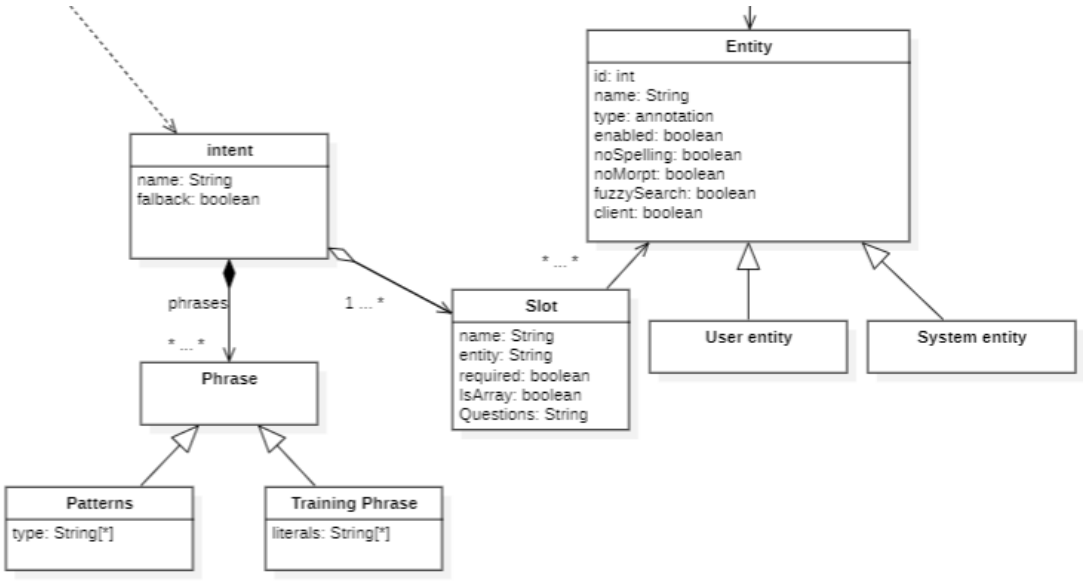

Рисунок 4.Классы для хранения информации о намерениях и сущностей с заполнением слотов, и их компонентов

3. Иллюстративный пример

В качестве примера был разработан чат-бот для поиска и ремонта неисправностей стиральной машины. Для формирования множества возможных неисправностей и соответствующего множества рекомендаций по их устранению было использовано руководство по ремонту неисправностей стиральных машин “Samsung” [4]. Разработанный чат-бот функционирует путем анализа описания обнаруженной неисправности в стиральной машине со стороны пользователя. В зависимости от характера и описания выявленной проблемы, бот переходит в различные состояния диалога. С учетом последующих ответов пользователя, чат-бот принимает решения относительно методов устранения указанной неисправности.

Чат-бот включает следующие состояния: старт, начальное состояние бота; меню; стиральная машинка не функционирует; в резервуар не поступает вода; не запрограммированная остановка во время стирки; не работает таймер; барабан не крутится; отключения бота; обработка исключений; и следующие сущности: стиральная машина; резервуар; барабан; таймер.

Сущности и намерения были описаны в терминах общей модели разговорного интерфейса [1]. Состояния и переходы между ними описаны в разрабатываемой в настоящие время проблемно-ориентированной модели диалога для диагностика неисправности бытовых приборов.

Задачей генератора является формирование спецификаций языкового интерфейса и кода управления диалогом на языке сценария JAICP DSL на основе разработанной и представленной выше платформенно-зависимой моделей JAICP:

Описание алгоритма работы генератора:

1. Заполняются объекты представленной информационной модели (интерфейса генератора) (фрагмент представлен на рис. 5).
2. Осуществляется генерация кода управления диалогом в форме языка сценария JAICP DSL на основе информации из объектов (фрагмент представлен на рис. 5).

Рисунок 5: Фрагмент входных и выходных данных работы генератора

4. Заключение

В представленной статье был проведен анализ и разработка генератора платформенно-зависимой модели для системы проектирования виртуальных помощников, основанных на технологии JUST AI с использованием платформы JAICP посредством Model-Driven Architecture (MDA) подхода. Основной целью исследования являлось создание эффективного инструмента для автоматизации процесса создания виртуальных помощников на данной платформе.

Были рассмотрены модели MDA и их трансформации, что позволило выявить основные преимущества данного подхода при разработке программного обеспечения.

Особое внимание было уделено объектным платформенно-зависимым моделям, используемым в рамках платформы JAICP. Подробно была описана структура этих моделей, их взаимосвязи и функциональные возможности, что способствовало более полному пониманию принципов работы данной системы.

В заключение статьи был представлен иллюстративный пример работы генератора платформенно-зависимой модели, с практическим применением разработанных концепций. Данный пример продемонстрировал эффективность и функциональность генератора по созданию виртуальных помощников на основе JUST AI с использованием платформы JAICP.

5. Список литературы

[1] А.Б. Столбов, Н.О. Дородных, О.А. Николайчук, А.Ю. Юрин, Н.В. Допиро. Генерация и трансформация специализированных моделей для проблемно-ориентированного интеллектуального помощника //Материалы конференции «Ляпуновские чтения 2023». – Иркутск: ИДСТУ СО РАН, 2023. – 184 с.

[2] A. Yurin, O. Nikolaychuk, N. Dorodnykh, A. Stolbov Towards knowledge-based virtual assistant development with the aid of ontology // International Multi-Conference on Engineering, Computer and Information Sciences (SIBIRCON). – 2022. – 830-834 pp.

[3] Pérez-Soler S., Guerra E., de Lara J. Model-Driven Chatbot Development.Lecture Notes in Computer Science. – 2020. Vol. 12400, pp. 207–222.

[4] Руководство по работе со стиральными машинами Samsung, 2024. URL: https://www.samsung.com/ru/support/category/home-appliances/laundry/