

Н. С. Федотов¹✉, П. Ю. Бугаков¹

Концептуальная архитектура интеллектуального ассистента для работы с распределенными источниками данных

¹Сибирский государственный университет геосистем и технологий,
г. Новосибирск, Российская Федерация
e-mail: Fedotov.N.S@yandex.ru

Аннотация. В статье предложена концептуальная архитектура персонального интеллектуального ассистента, предназначенного для поиска и обработки информации в распределенной пользовательской среде. Актуальность исследования состоит в том, что сообщения, письма, вложения и документы пользователя хранятся в разных цифровых каналах, тогда как единый поисковый интерфейс доступа к ним, как правило, отсутствует. Цель работы состоит в разработке архитектуры, которая способна интерпретировать запрос, выделять его параметры, выбирать релевантные источники данных, обращаться к внешним API и возвращать результат в структурированном виде. Методологической основой стали публикации по мультиагентным системам и смежным темам, а также официальная документация API сервисов, применимых в роли источников данных и интеллектуальных компонентов. В результате предложена трехагентная архитектура. Практическая ценность работы состоит в пригодности данной схемы для последующего создания программного прототипа персонального интеллектуального ассистента.

Ключевые слова: мультиагентная система, интеллектуальный ассистент, обработка естественного языка, информационный поиск, распределенные источники данных

N. S. Fedotov¹✉, P. Yu. Bugakov¹

Conceptual architecture of an intelligent assistant for distributed data source integration

¹Siberian State University of Geosystems and Technologies, Novosibirsk, Russian Federation
e-mail: Fedotov.N.S@yandex.ru

Abstract. The article proposes a conceptual architecture for a personal intelligent assistant designed to search for and process information in a distributed user environment. The relevance of the research lies in the fact that user messages, emails, attachments, and documents are stored in various digital channels, while a unified search interface for accessing them is typically absent. The goal of the work is to develop an architecture that can interpret a query, extract its parameters, select relevant data sources, access external APIs, and return the result in a structured format. The methodology is based on publications on multi-agent systems and related topics, as well as the official documentation of APIs for services that can be used as data sources and intelligent components. As a result, a three-agent architecture has been proposed. The practical value of the work lies in the suitability of this framework for the subsequent development of a software prototype for a personal intelligent assistant.

Keywords: multi-agent system, intelligent assistant, natural language processing, information retrieval, distributed data sources

Введение

Пользователь при использовании различных сервисов все чаще сталкивается с тем, что история общения распределена между чатами, письма и вложения находятся в почтовых сервисах, а документы и рабочие материалы – в файловых хранилищах. Даже простой запрос на поиск нужного сообщения или вложения в такой ситуации превращается в последовательность ручных переходов между интерфейсами. В исследованиях по интеллектуальным системам это противоречие обычно снимается за счет агентной декомпозиции, когда сложная задача разделяется между специализированными ролями, а координация между ними становится самостоятельной функцией системы [1–3]. Однако одной поисковой подсистемы для такого класса задач недостаточно. Чтобы ассистент действительно работал с персональными данными, ему необходимо понимать намерение пользователя, удерживать контекст обращения, уточнять недостающие параметры, а затем выбирать корректный сценарий исполнения. Поэтому архитектурная база подобного решения естественным образом опирается на исследования по персональным когнитивным ассистентам, когнитивным диалоговым системам и современным архитектурам диалоговых агентов [4–7].

Не менее важен языковой слой. Для персонального поиска недостаточно буквально сопоставлять слова запроса и текста документов. Необходимо выделять сущности, учитывать временные ограничения, строить более содержательное представление пользовательского намерения и в ряде случаев применять вопросно-ответные механизмы над найденными данными [7–10].

Отдельного внимания заслуживает голосовой режим и мессенджерный контур. Публикации по речевым интерфейсам и прикладным вопросно-ответным системам в мессенджерах показывают, что итоговое качество определяется не только корректностью распознавания речи, но и тем, насколько удачно распознанный текст включается в дальнейшую логику поиска и исполнения [11, 12].

Целью работы является разработка концептуальной архитектуры персонального интеллектуального ассистента для работы с распределенными источниками данных по текстовым и голосовым запросам. В статье рассматривается не чат-бот в узком смысле, а архитектурная схема, позволяющая объединить личную историю сообщений, электронную почту и файловые хранилища в едином контуре поиска и обработки.

Методы и материалы

Работа носит концептуальный характер. Сначала выделяются пользовательские сценарии, затем определяются необходимые данные, после чего функции системы раскладываются по специализированным ролям. Такой подход удобен тем, что позволяет не смешивать понимание запроса, маршрутизацию сценария и собственно вызов внешних сервисов.

Для доступа к личной истории сообщений в Telegram в предлагаемом варианте используется клиентский контур Telegram API/TDLib. Это удобное решение, так как архитектура ориентирована именно на персональный поиск по сооб-

щениям пользователя. В документации TDLib явно поддерживается поиск сообщений по всем чатам и внутри отдельного чата, а значит, позволяет строить сценарии междиалогового и контекстного поиска поверх пользовательской истории [13].

Почтовый и основные файловые контуры в статье опираются на сервисы Google. Gmail API предоставляет REST-доступ к почтовым ящикам, использует модель OAuth-scopes и поддерживает push-уведомления об изменениях ящика. Google Drive API, в свою очередь, поддерживает поиск файлов и папок, получение содержимого и поэтапную обработку изменений через changes.list. В совокупности это делает почту и файловое хранилище пригодными для событийно-ориентированного персонального ассистента [14, 15].

В качестве дополнительного файлового источника и альтернативного облачного контура может использоваться REST API Яндекс Диска, а для голосового режима – SpeechKit STT. Такой выбор важен не только с практической точки зрения, но и архитектурно: в системе появляется отдельный взаимозаменяемый слой доступа к файлам и отдельный взаимозаменяемый слой распознавания речи, что снижает жесткую зависимость от одного провайдера [16, 17].

Для интерпретации запроса, структурирования параметров и подключения инструментов могут использоваться модели OpenAI и Сбер GigaChat. Официальная документация обеих платформ поддерживает функциональные вызовы и подключение внешних инструментов, а OpenAI дополнительно предоставляет отдельный контур speech-to-text. Для концептуальной архитектуры это означает, что языковая интерпретация и выполнение внешних действий могут оставаться модульными и заменяемыми [18, 19].

В качестве промежуточного хранилища в работе предполагается реляционная база данных с JSON-полями и полнотекстовой индексацией. Полная копия пользовательских данных системе не требуется. Однако ей необходимы служебные таблицы для хранения нормализованных метаданных, найденных сущностей, идентификаторов объектов, статусов фоновых задач и журналов исполнения.

Результаты

В результате проведенного исследования сформирована архитектура мультиагентного интеллектуального ассистента, предназначенного для поиска и обработки информации из распределенных источников данных. Система включает три основных компонента: диалоговый агент, агент-оркестратор и агент исполнения. Общая схема взаимодействия компонентов представлена на рис. 1.



Рис. 1. Общая схема архитектуры интеллектуального ассистента для работы с распределенными источниками данных

Представленная схема показывает общий порядок прохождения пользовательского запроса через систему. На первом этапе запрос поступает через текстовый или голосовой интерфейс, после чего преобразуется в форму, пригодную для дальнейшей обработки. Далее запрос передается диалоговому агенту, который выделяет намерение пользователя и основные параметры задачи. Если данных достаточно, сформированная структура направляется агенту-оркестратору; если параметры неполные, система возвращает уточняющий вопрос.

Диалоговый агент реализует первичное взаимодействие с пользователем и выполняет преобразование естественно-языкового запроса в структурированное представление. Его схема приведена на рис. 2.

Диалоговый агент принимает текстовый запрос или результат распознавания голосовой команды. Затем он выполняет нормализацию текста, определяет намерение пользователя, извлекает сущности и фиксирует дополнительные параметры: источник данных, временной диапазон, тип объекта и ожидаемое действие. Например, запрос «найди вложение с договором из переписки за март» должен быть преобразован в структуру, где действие связано с поиском вложения, объектом является договор, а временным ограничением выступает март. Такой подход позволяет отделить пользовательскую формулировку от внутреннего формата задачи.

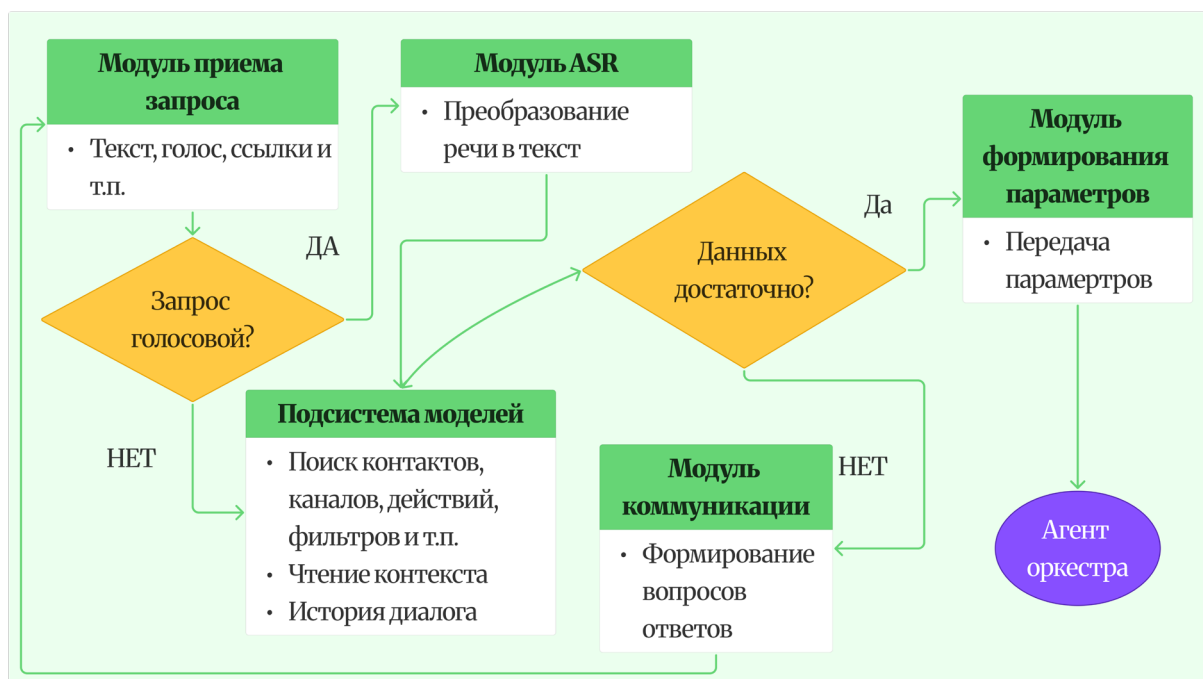


Рис. 2. Структура диалогового агента

Центральным компонентом архитектуры является агент-оркестратор, схема которого представлена на рис. 3.

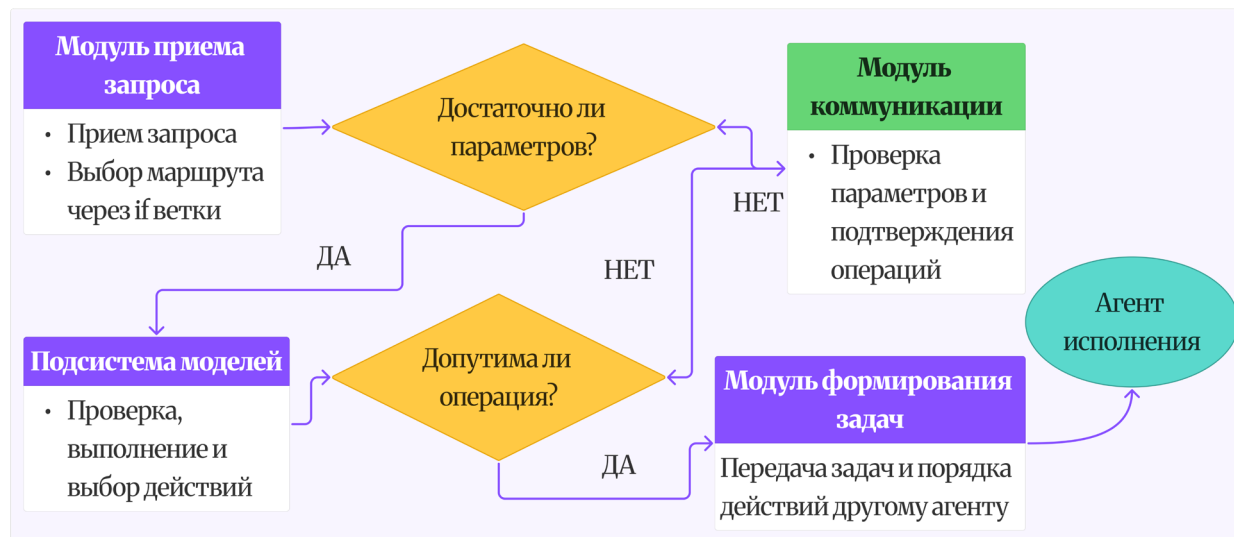


Рис. 3. Схема агента-оркестратора

Агент-оркестратор выполняет функции принятия решений и управления сценарием обработки. Он получает от диалогового агента формализованный запрос, проверяет полноту параметров, определяет допустимость операции и выбирает дальнейший маршрут выполнения. В зависимости от содержания запроса оркестратор может вернуть уточнение пользователю, обратиться к промежуточ-

ному хранилищу, передать задачу агенту исполнения или запустить фоновую обработку. Благодаря этому в системе разделяются понимание запроса и непосредственное выполнение действий.

Агент исполнения отвечает за обращение к внешним источникам данных и выполнение операций, выбранных оркестратором. Его структура показана на рис. 4.

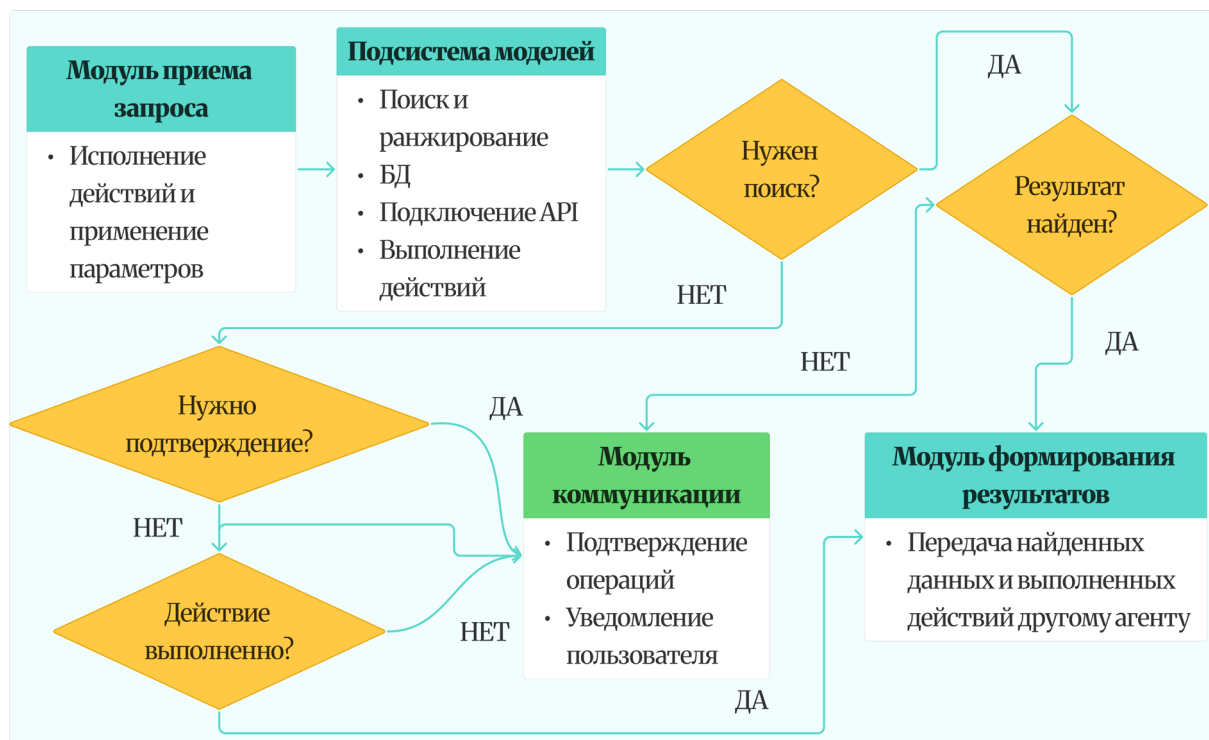


Рис. 4. Схема работы агента исполнения

Агент исполнения работает с почтовыми сервисами, мессенджерами, файловыми хранилищами, календарными сервисами и промежуточной базой данных. При поисковом сценарии он извлекает сообщения, письма, вложения или документы, нормализует полученные данные и возвращает результат в систему. При исполнительном сценарии агент может инициировать действие во внешнем сервисе, например создать событие в календаре, подготовить черновик документа или сохранить найденные сведения в хранилище. Для повышения устойчивости результаты операций и служебные метаданные могут сохраняться в базе данных.

В ходе проектирования были выделены основные пользовательские сценарии, которые отражены на диаграмме прецедентов (рис. 5).



Рис. 5. Диаграмма прецедентов интеллектуального ассистента

К основным сценариям относятся поиск сообщений по смысловому описанию, поиск вложений по контексту, поиск файлов в подключенных хранилищах, создание события в календаре, подготовка документа по образцу и запуск фоновой обработки. Отдельно учитывается сценарий уточнения запроса, поскольку пользователь не всегда указывает все параметры сразу. В этом случае система не завершает выполнение с ошибкой, а возвращает уточняющий вопрос через диалоговый агент.

Таким образом, предложенная архитектура объединяет пользовательский интерфейс, интеллектуальную обработку запроса, оркестрацию действий и доступ к распределенным источникам данных. Разделение функций между агентами делает систему расширяемой: при добавлении нового сервиса требуется изменить только слой исполнения и правила маршрутизации, не перестраивая всю архитектуру.

Обсуждение

Предложенная архитектура обладает несколькими важными достоинствами. Прежде всего, она уменьшает связанность между смысловой интерпретацией запроса и вызовом внешних сервисов. Это особенно важно в персональной цифровой среде, где почта, сообщения и файлы имеют разные модели доступа и разные механизмы синхронизации. Мультиагентная декомпозиция делает эту неоднородность не препятствием, а нормальным проектным условием.

Не менее важна модульность. Если в дальнейшем потребуется заменить речевой сервис, языковую модель или один из файловых коннекторов, такая замена не затронет всю систему. Благодаря этому архитектура хорошо подходит не только для быстрого прототипирования, но и для постепенного перехода от об-

лачных компонентов к локальным. Это особенно полезно в сценариях, где чувствительность данных ограничивает возможность внешней обработки.

При этом концепт имеет и ограничения. Во-первых, реальная реализация потребует аккуратной работы с правами доступа, токенами, журналами исполнения и политиками хранения данных. Во-вторых, качество ответа будет зависеть не только от языковой модели, но и от того, насколько корректно внутри системы описаны интенды, сущности, временные фильтры и условия полноты запроса. Поэтому основным направлением дальнейшей работы должна стать формализация межагентного контракта и экспериментальная проверка его устойчивости на реальных пользовательских сценариях.

Заключение

В статье разработана концептуальная архитектура персонального интеллектуального ассистента для работы с распределенными источниками данных по текстовым и голосовым запросам. Архитектура построена вокруг трех ролей – диалогового агента, агента-оркестратора и агента исполнения – и дополнена промежуточным хранилищем и фоновым контуром обработки.

Показано, что такой подход позволяет объединить в одном естественно-языковом интерфейсе поиск по личной истории сообщений, электронной почте и файловым хранилищам, а также поддерживать последующую интеллектуальную обработку результатов. Практическая значимость работы состоит в пригодности предложенной архитектуры для создания программного прототипа персонального ассистента. Дальнейшее развитие исследования целесообразно связать с реализацией прототипа, формализацией межагентного обмена и сравнительной оценкой облачных и локальных моделей в задачах персонального поиска.

БИБЛИОГРАФИЧЕСКИЙ СПИСОК

1. Прикладные многоагентные системы группового управления / В. И. Городецкий, О. В. Карсаев, В. В. Самойлов, С. В. Серебряков // Искусственный интеллект и принятие решений. – 2009. – № 2. – С. 3-24. – EDN KWTRFV.
2. Рыбина, Г. В. Моделирование процессов взаимодействия интеллектуальных агентов в многоагентных системах / Г. В. Рыбина, С. С. Паронджанов // Искусственный интеллект и принятие решений. – 2008. – № 3. – С. 3-15. – EDN JYAFUL.
3. Леденева, Т. М. Многоагентные системы: модели агентов и постановки задач, обеспечивающие их характерные свойства (часть 1) / Т. М. Леденева, М. А. Принев // Вестник Воронежского государственного университета. Серия: Системный анализ и информационные технологии. – 2024. – № 3. – С. 50-73. – DOI 10.17308/sait/1995-5499/2024/3/50-73. – EDN GHJMUH.
4. Персональный когнитивный ассистент: концепция и принципы работы / И. В. Смирнов, А. И. Панов, А. А. Скрынник, Е. В. Чистова // Информатика и ее применения. – 2019. – Т. 13, № 3. – С. 105-113. – DOI 10.14357/19922264190315. – EDN QQSTUK.
5. Станкевич, Л. А. Когнитивные диалоговые системы / Л. А. Станкевич // XXI международная научно-техническая конференция "НЕЙРОИНФОРМАТИКА-2019": Сборник научных трудов, Москва, 07–11 октября 2019 года. Том Часть 1. – Москва: Московский физико-технический институт (национальный исследовательский университет), 2019. – С. 9-62. – EDN IRTDJD.

6. Новая архитектура диалогового агента для прикладной генерации диалогов / В. А. Рындюк, Я. С. Варакин, Е. А. Писаренко, Е. Н. Пронченко // Волновая электроника и информационные системы : Сборник статей XXIV Международной научной конференции, Санкт-Петербург, 31 мая – 04 2021 года. Том Часть 3. – Санкт-Петербург: Санкт-Петербургский государственный университет аэрокосмического приборостроения, 2021. – С. 169-174. – EDN OYUUPS.
7. Смирнов, И. В. Разноуровневая обработка естественного языка для интеллектуального поиска и анализа текстов / И. В. Смирнов // Искусственный интеллект и принятие решений. – 2023. – № 1. – С. 90-99. – DOI 10.14357/20718594230109. – EDN ALPUWY.
8. Максименко, О. А. Технологии семантического поиска информационных ресурсов / О. А. Максименко, А. П. Савченко // Общество и экономика знаний, управление капиталами: цифровая экономика знаний : Материалы XII Международная научно-практическая конференция, Краснодар, 27–28 мая 2022 года. – Краснодар: Кубанский государственный университет, 2022. – С. 510-515. – EDN QGRDXD.
9. Алексеев, А. А. Автоматическое извлечение сущностей на основе структуры новостного кластера / А. А. Алексеев, Н. В. Лукашевич // Искусственный интеллект и принятие решений. – 2011. – № 4. – С. 51-59. – EDN PXGUUZ.
10. Открытое извлечение информации из текстов. Часть III. Система вопросно-ответного поиска / Е. В. Чистова, Д. С. Ларионов, А. О. Шелманов [и др.] // Искусственный интеллект и принятие решений. – 2021. – № 4. – С. 35-49. – DOI 10.14357/20718594210204. – EDN LUQGSK.
11. Повышение эффективности речевого интерфейса с применением когнитивных и лингвистических знаний / М. П. Фархадов, Н. В. Петухова, С. В. Васьковский, М. Э. Фархадова // Управление большими системами: сборник трудов. – 2019. – № 81. – С. 90-112. – DOI 10.25728/ubs.2019.81.4. – EDN FVONJA.
12. Вопросно-ответная система для поддержки абитуриентов с использованием современных мессенджеров / Д. Р. Филонов, Д. Ю. Чалый, Д. М. Мурин [и др.] // Моделирование и анализ информационных систем. – 2018. – Т. 25, № 4(76). – С. 411-420. – DOI 10.18255/1818-1015-2018-4-411-420. – EDN XWQZHF.
13. Telegram APIs. URL: <https://core.telegram.org/> (дата обращения: 27.04.2026).
14. Gmail API overview: <https://developers.google.com/workspace/gmail/api/guides> (дата обращения: 27.04.2026).
15. Search for files and folders. URL: <https://developers.google.com/workspace/drive/api/guides/search-files> (дата обращения: 27.04.2026).
16. REST API Диска – Технологии Яндекса. URL: <https://yandex.ru/dev/disk/rest/> (дата обращения: 27.04.2026).
17. Speech recognition | AI Studio documentation. URL: <https://aistudio.yandex.ru/docs/en/speechkit/stt/> (дата обращения: 27.04.2026).
18. OpenAI API: Responses Overview. URL: <https://developers.openai.com/api/reference/responses/overview/> (дата обращения: 27.04.2026).
19. GigaChat API: Обзор. URL: <https://developers.sber.ru/docs/ru/gigachat/api/overview> (дата обращения: 27.04.2026).

© Н. С. Федотов, П. Ю. Бугаков, 2026