

Описание технической архитектуры
программного обеспечения
«Интеллектуальная корпоративная
система интеграционного менеджмента
информации "ИКСИМИ:АГЕНТ 6.4"»

Наименование программного обеспечения: «Интеллектуальная корпоративная система интеграционного менеджмента информации "ИКСИМИ:АГЕНТ 6.4"»

Данный документ описывает базовые принципы и подходы к реализации различных компонентов ПО «Интеллектуальная корпоративная система интеграционного менеджмента информации "ИКСИМИ:АГЕНТ 6.4"»

1. Общая характеристика архитектуры

Программное обеспечение «Интеллектуальная корпоративная система интеграционного менеджмента информации "ИКСИМИ:АГЕНТ 6.4"» представляет собой модульную клиент-серверную информационную систему, реализованную на основе архитектуры передачи запросов. Архитектура построена с использованием принципов модульного проектирования и ориентирована на простоту сопровождения, прозрачность обработки данных и возможность быстрого развертывания как в локальной, так и в защищенной корпоративной инфраструктуре.

При проектировании системы были приняты следующие архитектурные решения:

- Модульность инструментов: Каждый инструмент реализует единый абстрактный интерфейс BaseTool и может быть заменён или отключён через конфигурацию без изменения ядра системы.
- Stateless-инструменты: Инструменты не хранят состояние сессии как атрибут экземпляра, что обеспечивает потокобезопасность при параллельной работе нескольких пользователей.
- Изоляция сессий: Каждая сессия пользователя имеет собственное рабочее пространство и рабочую копию проекта, что предотвращает конфликты между пользователями.
- Approval Flow: Опасные операции (изменение кода, shell-команды, обработка медиа, создание документов в 1С) требуют подтверждения пользователя с сохранением полного состояния сессии для последующего возобновления цикла.
- Абстракция поисковых провайдеров: Модуль web_search реализован через интерфейс SearchProvider, что позволяет подключать разные источники (Yandex Search API, локальный полнотекстовый индекс) без изменения ядра.
- Абстракция интеграции с 1С и Битрикс 24: Продукт не содержит кода 1С\Битрикс 24 и взаимодействует через стандартные открытые интерфейсы (OData, HTTP-сервисы), что обеспечивает независимость от конкретной версии и конфигурации.
- Diff-ориентированный merge: Изменения вносятся в изолированную копию проекта, применяются к оригиналу только по команде пользователя с обязательным созданием бэкапа и возможностью отката.
- Минимизация внешних зависимостей: Отсутствие обязательной зависимости от внешних сервисов; опциональные модули (поиск) могут быть отключены без потери основной функциональности.

2. Структурные элементы системы и их интерфейсы

2.1. Уровень представления (Presentation Layer)

Уровень представления реализует взаимодействие пользователя с системой через специализированные интерфейсы.

Основные компоненты:

1. REST API (FastAPI)

Обеспечивает программный доступ к функциям системы.

Основные эндпоинты:

- POST /process — обработка запроса пользователя
- POST /approval — обработка подтверждений опасных операций
- GET /diff/{session_id} — просмотр изменений в рабочей копии
- POST /merge/{session_id} — применение изменений к оригиналу проекта
- GET /health — проверка работоспособности системы
- GET /tools — список доступных инструментов
- GET /projects — список доступных проектов
- GET /sessions — список активных сессий пользователя

Аутентификация: Bearer-токен (API-ключ пользователя).

2. MAX-бот

Диалоговый интерфейс для работы с системой через MAX.

Основные команды:

- /start — начало работы, выбор проекта
- /projects — список доступных проектов
- /merge — применение изменений к оригиналу
- /diff — просмотр изменений
- /reject — отмена изменений
- /status — статистика системы
- /whoami — информация о текущем пользователе

Компоненты: inline-кнопки для подтверждения approval, клавиатура для выбора проектов.

Стек реализации: Python, maxapi (MIT License).

2.2. Уровень приложения (Application Layer)

Уровень приложения реализует основную бизнес-логику системы

Основные компоненты:

1. Оркестратор (MultiAgentOrchestrator)

Центральный компонент управления запросами пользователей.

Функции: аутентификация, проверка прав, создание изолированных сессий, итеративный цикл Tool Calling, сохранение состояния при approval, финализация ответа.

2. Реестр инструментов (ToolRegistry)

Регистрация и выполнение инструментов.

Функции: проверка политики безопасности (Policy), создание approval-запросов, выполнение инструментов в контексте сессии.

3. Менеджер сессий (SessionManager)

Управление изолированными сессиями пользователей.

Функции: создание сессий, создание рабочих копий проектов, очистка устаревших сессий, блокировки.

4. Менеджер подтверждений (ApprovalManager)

Обработка запросов на подтверждение опасных операций.

Функции: создание approval, валидация (session_id, user_id, status, expiry), approve/reject, cleanup.

5. Менеджер проектов (ProjectManager)

Сканирование и кеширование доступных проектов.

Функции: обход каталогов, определение типа проекта, кеширование метаданных.

6. Менеджер пользователей (AuthManager)

Аутентификация и авторизация пользователей.

Функции: аутентификация по API-токену и MAX ID, управление ролями, проверка прав доступа.

7. MoE-эксперты (DomainExpertTool)

Маршрутизация запросов к специализированным LLM.

Функции: выбор модели и системной инструкции в зависимости от предметной области.

8. Инструменты работы с 1С и Битрикс 24

Чтение данных (контрагенты, организации, номенклатура, счета) и создание документов через стандартные интерфейсы.

9. Инструменты работы с проектами

code_search (AST-поиск), code_edit (patch-based), file_operations, run_cli (allowlist + approval).

10. Инструменты работы с сетью

web_fetch (получение данных по URL, SSRF-защита), опциональный web_search (сменные провайдеры).

2.3. Уровень данных (Data Layer)

Уровень данных обеспечивает хранение и управление информацией о состоянии системы.

Основные компоненты:

1. Конфигурационные данные

- Локальные файлы конфигурации (config.py, .env)
 - Хранятся на сервере приложения
 - Содержат настройки LLM, политики безопасности, пути, списки инструментов
2. Пользовательские данные
- Файл data/users.json
 - Хранит учётные записи, API-токены, роли, разрешения
 - Управляется через AuthManager
3. Данные сессий
- Директория data/sessions/
 - Каждая сессия — JSON-файл с состоянием
 - Хранит working_copy, project_name, историю, tool_calls, pending approvals
4. Рабочие копии проектов
- Директория data/workspace/<user>/<session>/project/
 - Изолированная копия проекта для каждой сессии
 - Оригинал проекта не изменяется до команды /merge
5. Бэкапы проектов
- Директория data/backups/
 - Создаются перед применением изменений через /merge
 - Полная копия оригинала для отката
6. Approval-запросы
- Директория data/approvals/
 - JSON-файлы с запросами на подтверждение (approval_id, session_id, tool_name, args)
7. Кеш проектов
- Файл data/projects_cache.json
 - Список доступных проектов с метаданными (name, path, type, files_count)
8. Память
- Файл data/memory.json
 - История запросов пользователей (опционально)

3. Взаимодействие компонентов системы

Взаимодействие между уровнями и компонентами осуществляется следующим образом:

Этапы взаимодействия

1. Пользователь → Система (REST API / MAX)

Пользователь отправляет запрос на естественном языке.

Система аутентифицирует пользователя и проверяет права доступа к проекту.

2. Система → LLM (Tool Calling)

Оркестратор формирует сообщения: системная инструкция, история диалога, запрос пользователя.

Передаёт список доступных инструментов (JSON Schema).

LLM возвращает ответ с вызовом инструмента или финальный текст.

3. Система → Инструменты (выполнение)

Для каждого вызова система проверяет политику безопасности (Policy).
Если требуется подтверждение — сохраняет состояние и запрашивает approval.
Выполняет инструмент в изолированном workspace сессии.

4. Система → Учётная система 1С (интеграция)
Через стандартные интерфейсы (OData, HTTP-сервисы) выполняется чтение или запись данных.
Операции записи требуют подтверждения пользователя.

5. Система → Внешние источники (web)
Через web_fetch загружаются данные по указанному URL.
Опциональный web_search маршрутизирует запрос к выбранному поисковому провайдеру.

6. Система → Пользователь (ответ)
Финальный ответ передаётся пользователю.
При наличии изменений — пользователь может просмотреть diff (/diff) и применить его (/merge).

7. Администрирование
Администратор управляет пользователями, правами доступа, проектами через API и конфигурацию.

4. Архитектурный стиль

Основу архитектуры составляют следующие архитектурные подходы:

Слои архитектуры (трёхуровневая архитектура):

- Уровень представления — REST API (FastAPI) и MAX-бот.
- Уровень приложения — оркестратор, реестр инструментов, менеджеры сессий, approvals, МоЕ-эксперты, инструменты работы с 1С, проектами и сетью.
- Уровень данных — конфигурации, файлы сессий, рабочие копии проектов, бэкапы, кеш, память.

Модули архитектуры (модульная организация):

- Каждый инструмент реализует абстрактный интерфейс BaseTool.
- Все компоненты (менеджеры, инструменты, эксперты) слабо связаны.
- Подключение новых инструментов не требует изменений ядра.

Пайплайны (конвейер обработки запросов):

- Обработка запроса построена как последовательный цикл: Request → Authentication → Session → Tool Calling → Approval (при необходимости) → Response.

Принципы проектирования:

- Stateless-инструменты: инструменты не хранят состояние, что обеспечивает потокобезопасность при параллельной работе.
- Изоляция сессий: каждый пользователь работает в своём рабочем пространстве.
- Approval-ориентированная безопасность: опасные операции требуют подтверждения с сохранением состояния цикла.
- Минимизация внешних зависимостей: продукт работает автономно без обязательных внешних сервисов.
- Расширяемость: подключение новых инструментов и провайдеров через конфигурацию без изменения ядра.
- Прозрачность: все действия логируются, изменения доступны для просмотра через

diff.

5. Основные технические решения

В системе приняты следующие технические решения.

- Асинхронный HTTP-клиент/сервер — aiohttp — Высокая производительность, поддержка async/await, HTTP/HTTPS.
- Веб-фреймворк — FastAPI — Асинхронная обработка запросов, автоматическая валидация, встроенная OpenAPI-документация.
- Валидация данных — pydantic — Строгая типизация, автоматическая проверка входных данных.
- Парсинг HTML — BeautifulSoup4 + lxml — Надёжный парсинг веб-страниц.
- Работа с LLM — Ollama API — Локальный inference без облачных зависимостей.
- Парсинг XML — lxml — Высокопроизводительный парсер (для Yandex Search API).
- Валидация JSON Schema — jsonschema — Проверка аргументов инструментов.
- Конфигурация — .env + python-dotenv — Управление настройками без изменения кода.
- Безопасность — ipaddress, socket — SSRF-защита, DNS-проверка.
- MAX — maxapi — Асинхронный фреймворк под MIT-лицензией.

Решение обеспечивает полную автономность, минимальное количество внешних зависимостей, потокобезопасность и расширяемость без изменения ядра системы.

Сторонние компоненты и лицензии

Список использованных сторонних компонентов с указанием их правообладателей и лицензий их распространения

Компонент	Автор/организация	Лицензия	Ссылка
aiohttp	aiohttp team	Apache License 2.0	https://www.apache.org/licenses/LICENSE-2.0
FastAPI	Sebastián Ramírez	MIT License	https://github.com/tiangolo/fastapi/blob/master/LICENSE
uvicorn	Encode OSS Ltd	BSD 3-Clause License	https://github.com/encode/uvicorn/blob/master/LICENSE.md
pydantic	Samuel Colvin	MIT License	https://github.com/pydantic/pydantic/blob/main/LICENSE

Компонент	Автор/организация	Лицензия	Ссылка
BeautifulSoup4	Leonard Richardson	MIT License	https://www.crummy.com/software/BeautifulSoup/bs4/doc/#license
lxml	lxml team	BSD 3-Clause License	https://github.com/lxml/lxml/blob/master/LICENSE.txt
jsonschema	Julian Berman	MIT License	https://github.com/python-jsonschema/jsonschema/blob/main/COPYING
python-dotenv	Saurabh Kumar	BSD 3-Clause License	https://github.com/theskumar/python-dotenv/blob/main/LICENSE
tqdm	Casper da Costa-Luis	MIT License / MPL-2.0	https://tqdm.github.io/licence/
python-multipart	Andrew Dunham	Apache License 2.0	https://github.com/andrew-d/python-multipart/blob/master/LICENSE
maxapi	love-apples	MIT License	https://github.com/love-apples/maxapi/blob/main/LICENSE
brotlieffi	python-hyper	MIT License	https://github.com/python-hyper/brotlieffi/blob/master/LICENSE
OpenAI *	OpenAI, LLC	MIT	https://github.com/openai/openai-python/blob/main/LICENSE
ollama	Ollama	MIT License	https://github.com/ollama/ollama-python/blob/main/LICENSE
json, asyncio, subprocess, platform, time, logging, threading, datetime, hashlib, socket, ipaddress, math, random, dataclasses, typing	Python Software Foundation	Python Software Foundation License	https://docs.python.org/3/license.html

*Примечание: Библиотека OpenAI Python Library применяется исключительно как клиентская прослойка для взаимодействия с локальным сервером Ollama по протоколу, совместимому с OpenAI API. Все вызовы направляются на локальный эндпоинт (<http://localhost:11434/v1>), передача данных во внешние облачные сервисы OpenAI не производится. Данное решение принято для обеспечения гибкости при смене бэкенда генерации без изменения кода приложения.

Используемые компоненты библиотеки, реализующие технологии искусственного интеллекта с указанием их правообладателей и лицензий их распространения

Компонент	Правообладатель	Лицензия	Ссылка на лицензию
Torch (Pytorch)	Linux Foundation	BSD-3-Clause	https://github.com/pytorch/pytorch/blob/main/LICENSE
torchvision	PyTorch Core Team Soumith Chintala	BSD-3-Clause	https://github.com/pytorch/vision/blob/main/LICENSE
torchaudio	Facebook Inc.	BSD-2-Clause	https://github.com/pytorch/audio/blob/main/LICENSE
transformers	The Hugging Face team	Apache-2.0	https://github.com/huggingface/transformers/blob/main/LICENSE
sentence-transformers	UKPLab	Apache-2.0	https://github.com/UKPLab/sentence-transformers/blob/master/LICENSE
FAISS	Meta Platforms, Inc.	MIT	https://github.com/facebookresearch/faiss/blob/main/LICENSE

Используемые модели машинного обучения с указанием их правообладателей и лицензий их распространения

Модель	Правообладатель	Лицензия	Ссылка на лицензию
multilingual-e5-base	Microsoft Research	MIT	https://huggingface.co/intfloat/multilingual-e5-base
Wrench-9B-Q4_K_M-GGUF	Clank Labs	Apache 2.0	https://huggingface.co/ClankLabs/Wrench-9B-Q4_K_M-GGUF

Системные зависимости с указанием их правообладателей и лицензий их распространения

Компонент	Правообладатель	Лицензия	Ссылка на лицензию
Python 3.11	Python Software Foundation	PSF License Agreement	https://docs.python.org/3.11/license.html
Ollama	Ollama	MIT License	https://github.com/ollama/ollama/blob/main/LICENSE
JavaScript	Microsoft	Apache-2.0	https://github.com/microsoft/TypeScript/blob/main/LICENSE.txt

6. Требования к аппаратному обеспечению (рекомендованные значения)

ЦП: x86_64 архитектура, от 4 ядер.

ОЗУ: от 8 ГБ (зависит от размера используемой языковой модели).

Дисковая подсистема: SSD от 50 ГБ (зависит от размера загруженной языковой модели и проектов).

GPU (опционально): от 8 ГБ VRAM для ускорения инференса LLM.

ОС: Рекомендуется Astra Linux, ОСнова.

Поддержка:

- Дистрибутив Linux с поддержкой Python 3.10+ (Debian, Ubuntu),

- Windows 10/11 с WSL2.

Дополнительно: Python 3.10+, Ollama или совместимый LLM-сервер,

1С:Предприятие 8.3+ (опционально, лицензия пользователя).

Архитектура программного обеспечения «Интеллектуальная корпоративная система интеграционного менеджмента информации "ИКСИМИ:АГЕНТ 6.4"» обеспечивает высокую производительность и эффективность при сохранении простоты сопровождения и прозрачности обработки данных. Реализованный подход соответствует современным принципам построения ПО и позволяет эффективно использовать программное обеспечение как в небольших организациях, так и в крупных корпоративных инфраструктурах с повышенными требованиями к безопасности и локальной обработки данных