ИНТЕЛЛЕКТУАЛЬНАЯ СИСТЕМА УПРАВЛЕНИЯ
КОРПОРАТИВНЫМ КАЗНАЧЕЙСТВОМ FinAxio
МОДУЛЬ «ИСКУССТВЕННЫЙ ИНТЕЛЛЕКТ — СРЕДА АНАЛИТИКИ»

АРХИТЕКТУРА ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ

Назначение документа
Документ фиксирует логическую архитектуру аналитического модуля, его границы, состав компонентов, способы интеграции с принимающей системой TMS FinAxio, модель данных, требования к безопасности и ограничения ответственности.

Октябрь 2026

1 Общие сведения

Настоящий документ описывает архитектуру программного обеспечения интеллектуального модуля FinAxio «Искусственный интеллект — Среда аналитики». Модуль предназначен для диалогового анализа данных корпоративного казначейства и функционирует в контуре принимающей системы TMS FinAxio. Архитектура не предполагает повторной реализации казначейских операций или передачи модулю полномочий по акцепту платежей и изменению первичных финансовых объектов.

Параметр	Значение
Наименование программного продукта	Интеллектуальная система управления корпоративным казначейством FinAxio
Заявляемый функциональный модуль	«Искусственный интеллект — Среда аналитики»
Тип описываемого объекта	Программный модуль прикладной аналитики, интегрируемый в принимающую TMS-систему
Идентификатор документа	ARC-010-2026
Основание	Архитектурное описание объектной модели системы
Технологический состав	TypeScript/React-интерфейс; TypeScript-порты, адаптеры и прикладные сценарии; Python-процесс скоринга; SQL-миграции; manifest и реестр компонентов
Основной результат работы	Формальный разбор вопроса, разрешённая выборка, расчёт или прогноз, объяснение результата и запись результата для экспертной оценки

Граница заявляемого программного обеспечения
Первичные финансовые объекты — платежи, инструменты, счета, лимиты, справочники и договорные полномочия — принадлежат TMS FinAxio. Модуль получает необходимые проекции через порты и адаптеры, формирует аналитические результаты и сохраняет собственные сессии, следы анализа и рекомендации.

1.1 Объект и область архитектурного описания

В состав описания входят пользовательский интерфейс аналитической среды, прикладной контур исполнения формальных запросов и решений, порты и адаптеры к сервисам TMS, модельный процесс, аналитические таблицы PostgreSQL и контуры авторизации и аудита, предоставляемые принимающей системой. Сети, TLS, ingress, секреты, регистрация маршрутов и эксплуатационные политики рассматриваются как ресурсы принимающей инфраструктуры, а не как самостоятельные функции модуля.

2 Назначение и область применения

Модуль предоставляет казначею или иному уполномоченному пользователю средства диалоговой аналитики над данными TMS. Пользователь формулирует вопрос, после чего модуль формирует формальный запрос, получает разрешённую выборку с признаками полноты и пагинации, выполняет расчёт или обращается к модельному процессу, выдаёт объяснение и сохраняет результат для экспертной оценки. Аналитический журнал ведёт статусы open/done/skipped; акцепт и согласование платежей выполняются только в ядре TMS.

Основные сценарии применения охватывают анализ ликвидности и денежных потоков, оценку рисков и лимитов, анализ финансовых инструментов, подготовку вариантов размещения и прогнозирование по доступным датированным рядам. Для прогнозных сценариев учитываются дата среза as_of, доступность статьи денежных потоков и наличие совместимой истории.

Функциональная возможность	Входные данные	Результат
Диалоговая постановка задачи	Вопрос пользователя, UserContext, права доступа	FormalQuery и последовательность шагов обработки
Разрешённая выборка	Портовые операции TMS, организационный scope, дата среза	ToolResult с происхождением, полнотой и сведениями о пагинации
Расчёт и оценка	Снимок входных данных, параметры операции, модельный артефакт	Числовой или структурированный результат с идентификатором исполнения
Прогнозирование	Опубликованный код статьи ДДС, as_of, совместимый ряд не менее 366 точек	Прогноз либо мотивированный отказ при недостаточной истории или неизвестном коде
Объяснение результата	Результат расчёта, SHAP-пары и контекст выполнения	Проверяемое текстовое объяснение и набор факторов
Фиксация результата	Сессия, снимок, рекомендация, actor	Запись в аналитические таблицы с проверкой владельца и статуса

3 Архитектурные принципы и границы ответственности

Архитектура построена как интегрируемый аналитический контур с разделением представления, прикладных решений, интеграционных портов, модельного процесса и хранения аналитических результатов. Порт TreasuryPorts задаёт зависимость расчёта от возможностей принимающей системы; адаптеры позволяют заменить источник данных без переписывания прикладного сценария.

Системные границы распределены по владению данными. TMS владеет первичными финансовыми объектами, полномочиями и авторизацией; модуль владеет сессией, следом анализа, снимком входа и рекомендацией; принимающая инфраструктура владеет сетевыми и эксплуатационными механизмами. Разделение команд и запросов используется как способ описания границ и не означает декларацию универсальной платформы.

Принцип	Архитектурный смысл	Контроль реализации
Порт и адаптер	Источник данных можно заменить без переписывания сценария	Сверка LiveDeps и подписей методов портов
Граница host	Авторизация и первичные записи принадлежат TMS	Проверка маршрутов, actor и исходного UserContext
Снимок входа	Расчёт привязан к фактически использованным данным сессии	Проверка sessionId, completeness и происхождения
Отказ при неполноте	Неподтверждённая выборка не представляется как полный результат	Различение неполной страницы и подтверждённого результата
Изоляция по владельцу	Сессия, диалог и рекомендация доступны только владельцу и допустимому scope	Проверка user_id и организационных ограничений при записи и чтении

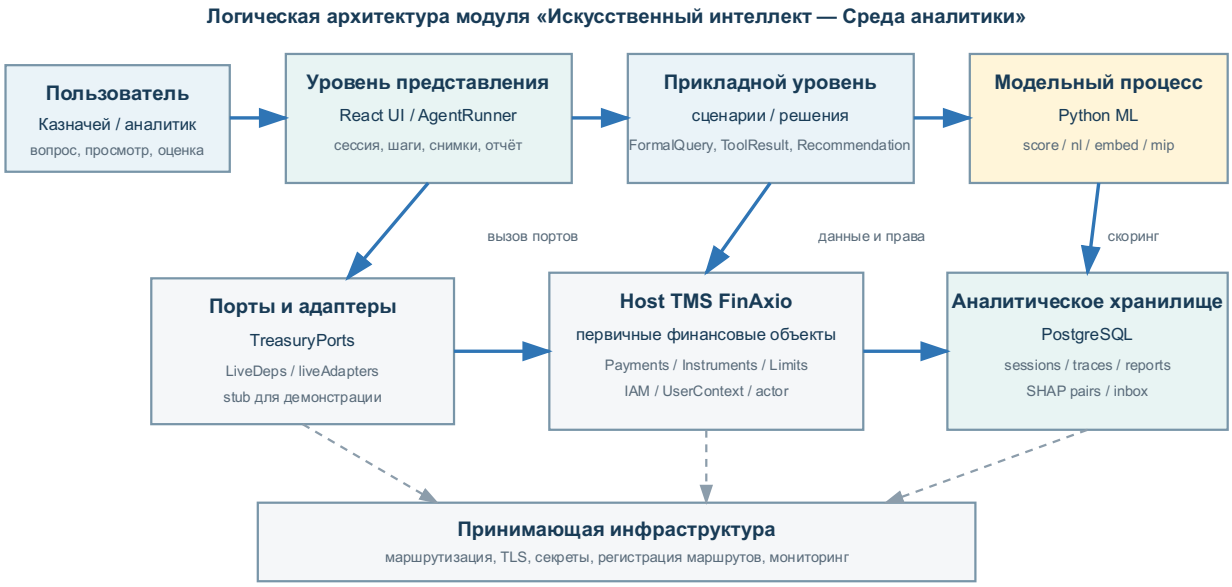


Рисунок 1 — Логическая архитектура модуля и его окружения

4 Состав программных компонентов

Компоненты распределены по слоям. Имена файлов и модулей ниже приведены как идентификаторы поставки и позволяют сопоставить архитектурное описание с исходным кодом и миграциями.

Слой / компонент	Состав и артефакты	Ответственность
Представление	ui/AnalyticsEnvironment.tsx; ui/AgentRunner.ts; ui/aiApi.ts; карточки вызовов; ui/toolSnapshot.ts	Ввод вопроса, управление сессией, отображение шагов и снимка; вызов /api/ai/tool-exec либо локального Python в браузере; отмена прежнего запуска той же сессии
Прикладной контур	decisions/runDecision.ts; типы FormalQuery, ToolResult, Recommendation	Формирование формального запроса, оркестрация расчёта, проверка инвариантов и подготовка рекомендации
Интеграционные порты	ports/types.ts; ports/registry.ts; ports/actor.ts	Контракт возможностей TMS, реестр реализаций, получение AiActor через runWithActor
Адаптеры TMS	adapters/liveAdapters.ts; adapters/coreEvents.ts	Вызов сервисов платежей, инструментов, лимитов и расчётов риска; узкая инвалидация по событию PaymentStatusChanged
Модельный процесс	ml/server.py	Обслуживание /score, /nl, /embed, /mip и /health; расчёт и модельные операции в отдельном процессе
Хранилище аналитического контура	adapters/store.ts; db/497_ai_analytics_contour.sql; db/498_ai_analytics_shap_pairs.sql; db/499_ai_analytics_inbox.sql	Сохранение сессий, следов, рекомендаций, SHAP-пар и inbox; подтверждение записи после ответа SQL
Контур поставки	manifest и реестр компонентов; метаданные моделей	Идентификация версии и модельного артефакта, контроль хеша и сопоставление состава выпуска

5 Доменная модель и ограниченные контексты

Доменная модель отделяет первичные финансовые объекты от аналитических сущностей. Финансовый инструмент, платёж, счёт, контрагент и лимит являются входными проекциями ядра, которое владеет их жизненным циклом. Аналитический контекст содержит формальный запрос, результат инструмента, решение, рекомендацию, снимок и след выполнения.

Объект	Владелец жизненного цикла	Инварианты и связь
Платёж, финансовый инструмент, счёт	TMS FinAxio	Сохраняются исходный идентификатор и организационный scope; модуль не копирует объект как самостоятельную первичную запись
Контрагент, лимит, справочник	TMS FinAxio	Доступны через опубликованные сервисы и права принимающего контура
FormalQuery	Контур анализа	Содержит формализованный вопрос, параметры операции, дату среза и контекст владельца
ToolResult	Контур анализа	Содержит результат вызова, происхождение, полноту и сведения о пагинации
SessionSnapshot	Контур анализа	Связан с sessionId, фиксирует input_data, completeness и происхождение входа
Решение	Контур анализа	Ссылается на фактически исполнившийся модельный артефакт и его хеш
Recommendation	Модуль	Содержит проверяемое объяснение, статус и ссылку на сессию; владелец равен actor

5.1 Связи и поток данных

Связь Payment—CashFlow определяется источником и временным срезом; операция не дублируется в отдельной таблице модуля. Синонимы и словарь статей денежных потоков разрешаются по опубликованной и версионированной схеме. Для прогноза адаптер формирует совместимый с обучением датированный ряд, содержащий не менее 366 точек и заканчивающийся днём as_of−1.
Если история недостаточна, код статьи неизвестен или полнота входа не подтверждена, модуль возвращает мотивированный отказ и сохраняет сведения, необходимые для проверки отказа. Это предотвращает выдачу числового ответа, который ошибочно воспринимается как полный или относящийся к иной одноимённой модели данных.

5.2 Жизненный цикл аналитической сессии

1. HostBridge передаёт интерфейсу токен, формат дат и механизм выгрузки через configureAnalyticsHost.
2. AgentRunner связывает запуск с активной сессией через runGate и не допускает продолжение прежнего запуска той же сессии после его отмены.
3. Прикладной контур строит FormalQuery, вызывает разрешённый порт, получает ToolResult и фиксирует снимок с completeness и происхождением.
4. Решение выполняется в принятом маршруте: через модельный процесс или иной зарегистрированный адаптер; результат связывается с модельным артефактом и его хешем.
5. Recommendation сохраняется только после проверки владельца сессии, снимка и ключа операции; изменение статуса выполняется после записи и подтверждения commitRecommendationStatus.

Каждое сообщение пользователя сначала превращается в формальный запрос: BFF по адресу /api/ai/formal-query определяет намерение пользователя и параметры — например, период, валюту и дату. Сначала работает внутренняя грамматика, если она не распознала намерение, подключаются дополнительные лингвистический и семантический разборы. Генеративная модель через настраиваемый согласно политикам заказчика привлекается на этом этапе только для нераспознанного намерения или недостающих обязательных параметров, причём уже найденные значения она не заменяет. Если обязательных данных всё ещё не хватает, экран просит пользователя уточнить запрос. После успешного разбора начинается отдельный диалоговый шаг: генеративная модель формулирует ответ и выбирает нужный инструмент, а сам инструмент читает данные или вызывает внутренние расчётные модели для прогноза, классификации и оценки риска.

6 Интерфейсы и интеграция с TMS FinAxio

Целевой контракт интеграции предполагает, что host подключает экран и маршруты модуля, передаёт IAM и UserContext, выдаёт портам сервисы ядра, применяет аналитические миграции на собственной PostgreSQL и адресует внутренний модельный процесс. Пользовательский REST-маршрут обслуживается принимающим приложением. gateway/openapi.ts описывает область модуля кратким объектом OpenAPI и не является поставляемым контроллером полного TMS API.

Интерфейс / сервис	Назначение	Условия использования
configureAnalyticsHost / HostBridge	Передача токена, формата дат и выгрузки в UI	Вызывается принимающей оболочкой; React-компоненты не импортируют сервисы TMS напрямую
/api/ai/chat-step	Один шаг диалогового сценария	Вызывается AgentRunner в рамках активной сессии и текущего runId
/api/ai/tool-exec	Исполнение разрешённого инструментального вызова	Результат должен иметь признаки полноты, пагинации и происхождения
/score, /nl, /embed, /mip	Модельные операции и обработка запроса	Обслуживаются Python-процессом; точный сетевой адрес задаётся инфраструктурой host
/health	Проверка доступности модельного процесса	Используется при эксплуатационной проверке маршрута
PaymentsService.list	Получение платежей	Вызов через live-адаптер при наличии подтверждённой зависимости
PaymentRegistryService.balancesSeries	Получение датированного ряда остатков	Используется для анализа и прогнозных сценариев
InstrumentsService.list	Получение финансовых инструментов	Организационный scope и предметные права проверяются в принимающей системе
BankLimitsService.overview	Получение сведений о лимитах	Данные не становятся самостоятельным справочником модуля
MarketRiskCalcService.calculate	Расчёт рыночного риска	Результат связывается с входным снимком и контекстом исполнения

7 Архитектура развёртывания

Развёртывание выполняется в составе принимающей инфраструктуры. Модуль не требует выделения самостоятельного универсального шлюза: host предоставляет прикладные маршруты, контекст пользователя, сервисы ядра, базу данных аналитического контура и сетевое взаимодействие с модельным процессом.

Узел / среда	Размещаемые компоненты	Роль в архитектуре
Клиентская среда host	React-компоненты аналитической среды, AgentRunner, aiApi, toolSnapshot	Ввод и отображение; управление сессией; вызов host-маршрутов
Сервер принимающего приложения	Прикладные сценарии, маршруты /api/ai/*, ports, liveAdapters, actor-контур	Авторизованный доступ к TMS, оркестрация, предметные проверки и сохранение результата
Модельный процесс	ml/server.py и зарегистрированные модельные артефакты	Изолированное выполнение score, nl, embed, mip; проверка доступности через health
PostgreSQL host	Аналитические таблицы миграций	Хранение сессий, следов, рекомендаций, SHAP-пар и inbox
TMS FinAxio	Платежи, инструменты, счета, лимиты, справочники, IAM и UserContext	Источник первичных объектов и полномочий; владелец транзакционных операций
Принимающая инфраструктура	TLS, ingress, секреты, регистрация маршрутов, мониторинг	Сетевое и эксплуатационное окружение; параметры определяются host

7.1 Условия эксплуатации и поставки

Для работы модуля требуется принимающее приложение, способное подключить React/TypeScript-интерфейс и обслужить маршруты аналитического контура; зарегистрированный IAM и UserContext; доступ к согласованным сервисам TMS; PostgreSQL для аналитических миграций; поддерживаемая среда выполнения Python для модельного процесса; а также сетевой маршрут между сервером host, модельным процессом и базой данных. Конкретные версии ОС, СУБД, интерпретатора и параметры ресурсов определяются manifest и эксплуатационной конфигурацией принимающей системы.
Целевая интеграция или обновление аналитического контура предполагает поставку компонентов и manifest, регистрацию компонентов и метаданных моделей, применение миграций, настройку адреса модельного процесса и выполнение сквозного испытания. Исполнение этих действий в конкретной инсталляции подтверждается отдельно. Модуль не изменяет первичные записи TMS за пределами предусмотренных host-сервисов.

8 Транзакционная модель и согласованность данных

Первичные операции являются атомарными в границах TMS. Аналитические сущности хранятся в отдельных таблицах и не образуют универсального журнала версий финансовых инструментов или outbox для отправки денег. adapters/store.ts подтверждает запись сессии или рекомендации только после ответа SQL и проверки user_id; при отсутствии подключённой базы результат явно отмечается как persisted:false.

Механизм	Реализация	Результат контроля
Сохранение сессии и рекомендации	SQL-операция через adapters/store.ts с проверкой user_id	Состояние persisted подтверждается только после ответа базы данных
Защита от повторной доставки	adapters/coreEvents.ts и уникальный event_id inbox	Повторное событие не создаёт дубликат обработки
Инвалидация	Узкая обработка PaymentStatusChanged и повышение generation	Обновляется только связанный аналитический контекст, а не все события TMS
Одновременное сохранение	Проверка владельца, снимка и operation key; объединение одинаковых запросов	Снижается риск дублирования рекомендаций и записи чужих данных
Согласованность входа	ToolSnapshot с sessionId и completeness	Нельзя подменить input_data при ссылке на существующий снимок

Механизм	Реализация	Результат контроля
Статус рекомендации	commitRecommendationStatus после записи	Статус не меняется до успешной фиксации результата

9 Безопасность и обработка данных

Аутентификация пользователя и проверка права доступа к аналитике выполняются IAM принимающей системы. Предметные права каждой операции проверяются в сервисах ядра. Роли «аналитик», «контролёр» и «администратор» описывают задачи в контуре, но не заменяют разрешение, выданное принимающей системой.

Контроль	Архитектурное решение
Контекст пользователя	AiActor содержит userId, orgScopes, roles, correlationId, traceId и raw UserContext; actor передаётся в прикладной и интеграционный контуры
Организационный scope	accessScope возвращает null при пустой области; wildcard допускается только по полномочиям host; прямые выборки каталогов фильтруются по организации там, где такой фильтр определён
Изоляция сессий	Чтение и запись диалога, сессии и рекомендации проверяют user_id и принадлежность к допустимому scope
Секреты и журнал	Служебный журнал маскирует секреты; полный prompt и персональные данные не раскрываются пользователям без соответствующего права
Минимизация данных	В модуль передаются только разрешённые проекции и данные, необходимые для конкретного аналитического сценария; первичные объекты не дублируются без необходимости
Политики хранения	Сроки хранения, требования защищённости и отраслевое соответствие задаются политиками host и подтверждаются отдельными актами, а не одной схемой таблиц

10 Надёжность, наблюдаемость и контроль выпуска

Надёжность аналитического результата обеспечивается сохранением входного снимка, фиксацией полноты, связыванием решения с модельным артефактом и отказом при неподтверждённых данных. Внешняя доступность модельного процесса проверяется маршрутом /health, а итоговая готовность эксплуатационного маршрута — сквозным испытанием в принимающей инфраструктуре.

Для диагностики используются correlationId, traceId, runId, event_id и сохранённый след анализа. Количественные показатели производительности, SLA и ёмкости в исходном архитектурном описании не заданы и не фиксируются настоящим документом; они устанавливаются отдельно для конкретной конфигурации внедрения.

Контрольная точка	Что проверяется
До запуска	Наличие зарегистрированных маршрутов, IAM, UserContext, сервисных зависимостей и применённых миграций
Во время запуска	Связь runId с активной сессией, права actor, полнота ToolResult и неизменность SessionSnapshot
При работе с моделью	Доступность /health, идентичность модельного артефакта, хеш и корректная привязка результата
При сохранении	Владелец сессии, operation key, ответ SQL, persisted и статус рекомендации
После выпуска	Сквозное испытание от пользовательского вопроса до проверяемой записи результата

11 Ограничения и исключения из ответственности модуля

Для корректного толкования архитектуры фиксируются следующие границы. Модуль не заменяет TMS FinAxio, не владеет жизненным циклом платежей и финансовых инструментов, не принимает и не согласует платежи, не выдаёт полномочия и не является самостоятельным IAM. Он не разворачивает полный TMS API, не предполагает отдельный универсальный API-шлюз.

12 Термины и сокращения

Термин	Определение
TMS	Принимающая система управления корпоративным казначейством FinAxio, владеющая первичными финансовыми объектами и транзакционными операциями
Host	Принимающее приложение и его инфраструктурный контур, в который интегрируется аналитический модуль
BFF	Бэкенд основной TMS
IAM	Система управления идентификацией и доступом принимающей системы
Actor / AiActor	Контекст уполномоченного пользователя и организации, передаваемый в операции аналитического контура
FormalQuery	Формализованное представление вопроса пользователя и параметров аналитической операции
ToolResult	Результат разрешённого инструментального вызова с признаками полноты, пагинации и происхождения
SessionSnapshot	Зафиксированный снимок входных данных, связанный с сессией и датой среза
as_of	Дата, на которую разрешено использовать входные данные и формировать расчёт или прогноз
Stub-адаптер	Демонстрационная реализация порта, не подтверждающая подключение к производственному TMS
Live-адаптер	Реализация порта, подключающая подтверждённые сервисы принимающей системы