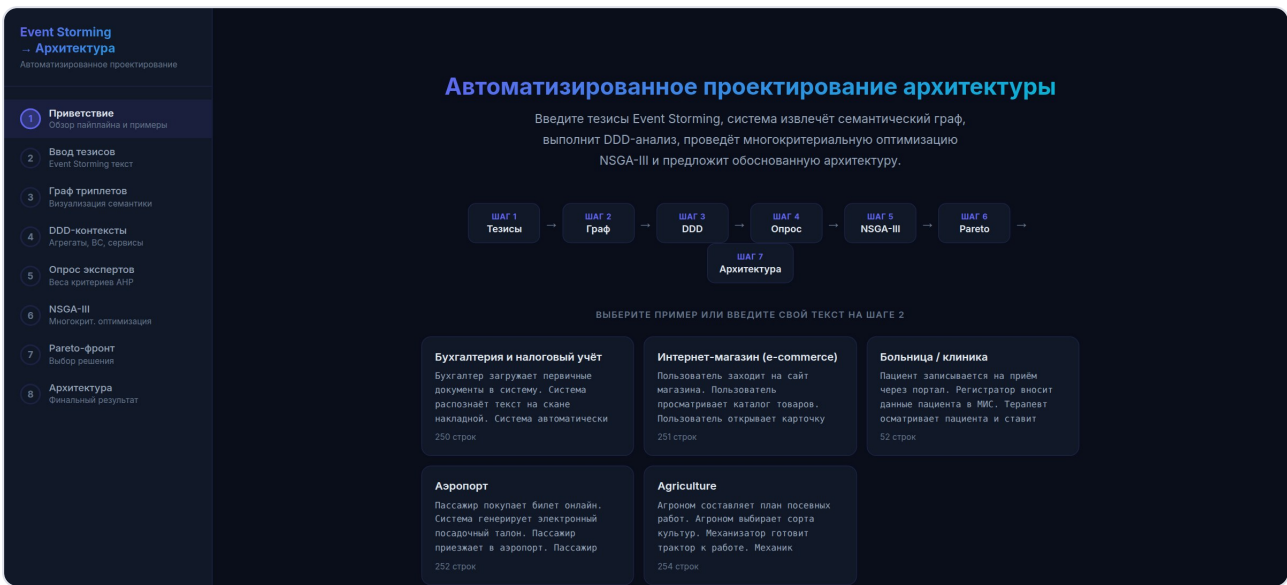


Event Storming → Архитектура

Программный комплекс автоматизированного проектирования архитектуры информационных систем: от текстовых тезисов Event Storming до обоснованной архитектуры с выбором стиля (монолит / SOA / микросервисы) для каждого ограниченного контекста.

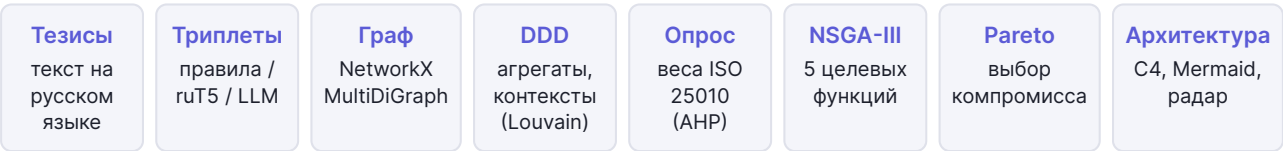


1 Назначение системы

Система принимает на вход описание бизнес-процессов на русском языке в виде тезисов Event Storming («Регистратор вносит данные пациента в МИС», «Система формирует направление на анализы» и т. п.) и выполняет полный цикл проектирования:

- извлекает из текста семантические триплеты «субъект — действие — объект» и строит граф предметной области;
- выполняет DDD-анализ: выделяет агрегаты, команды, события, акторов и ограниченные контексты (Bounded Contexts);
- получает веса критериев качества ISO/IEC 25010 от эксперта методом анализа иерархий (АНП);
- проводит многокритериальную оптимизацию генетическим алгоритмом NSGA-III и строит Парето-фронт архитектурных решений;
- рекомендует компромиссное решение и выдаёт итоговую архитектуру в виде графа контекстов, диаграмм PlantUML C4 и Mermaid, а также профиля качества.

Сквозной конвейер обработки



2 Архитектура системы

Комплекс состоит из пяти слоёв; каждый слой доступен как через веб-интерфейс, так и через REST API или командную строку.

2.1. Извлечение триплетов

Три взаимозаменяемых экстрактора выдают триплеты единого формата (source, relation, target, types). Типы сущностей: actor, system, document, aggregate, read_model, event; типы отношений: command, event.

Экстрактор	Реализация	Модуль
На основе правил (по умолчанию)	Natasha (морфология, синтаксис) + pymorphy2, словари акторов, систем и документов, разрешение анафоры. Быстрый и интерпретируемый.	rule_based/
Нейросетевой	Seq2Seq-модель на базе ruT5-base со скользящим окном в 2–3 предложения; загружается при первом обращении.	neural/
LLM	Внешняя языковая модель через API (Ollama / ChatGPT), используется для сравнения.	es_llm/

Перед извлечением текст нормализуется, сложносочинённые предложения разбиваются на простые; опционально назывные тезисы преобразуются в SVO-предложения (препроцессинг).

2.2. Граф и DDD-анализ

Триплеты собираются в ориентированный мультиграф `networkx.MultiDiGraph`. Модуль кластеризации выделяет сообщества алгоритмом Лувена, а DDD-анализатор (`rule_based/ddd_analyzer.py`) формирует агрегаты, ограниченные контексты, команды, события, запросы, потоки данных и API сервисов, а также присваивает контекстам предметные названия.

2.3. Формальная модель архитектурных решений

Для каждого ограниченного контекста выбирается архитектурный стиль, протокол коммуникации, стратегия хранения данных и схема развёртывания (`ArchStyle × CommProto × DataStrat × DeployType`), что даёт до 3^N вариантов по стилям. Модуль ограничений отсекает недопустимые сочетания (например, межмонолитное взаимодействие через брокер сообщений), библиотека образцов включает CQRS, Saga, Circuit Breaker, API Gateway и др. Для каждого варианта рассчитываются структурные метрики (связность, зацепление, задержка, стоимость и т. д.), которые отображаются на характеристики качества ISO/IEC 25010.

2.4. Многокритериальная оптимизация

Задача ставится в рутинно: одна переменная на контекст со значением в $[0, 2]$ (0 — монолит, 1 — SOA, 2 — микросервисы). Пять целевых функций — сопровождаемость, надёжность, производительность, безопасность (максимизация) и стоимость (минимизация). NSGA-III с опорными направлениями Das-Dennis строит Парето-фронт; прогресс передаётся в браузер потоком Server-Sent Events.

2.5. Веб-сервер и интерфейс

Flask-сервер (`server.py`, порт 5001) отдаёт одностраничный интерфейс `web/index.html` и REST API. Экстрактор на основе правил загружается при старте, нейросетевой — при первом запросе. Визуализация выполнена на `vis-network` (графы), `Plotly` (3D Парето-фронт), `Chart.js` (`hypervolume`, радар) и `Mermaid`.

3 Пользовательский интерфейс

Интерфейс построен как пошаговый мастер из восьми шагов. Слева — навигатор с отметками о завершённых шагах и кнопками «Назад / Далее»; результат каждого шага сохраняется в сессии браузера и используется на следующих шагах. Ниже работа мастера показана на примере «Больница / клиника» (52 тезиса).

1 Приветствие

Стартовый экран показывает схему конвейера и карточки готовых примеров из корпуса (бухгалтерия, интернет-магазин, больница, аэропорт и др.). Щелчок по карточке подставляет текст примера на шаг 2 (экран показан на обложке).

2 Ввод тезисов

Справа — поле для тезисов Event Storming (по одному на строку), слева — выбор экстрактора (Rule-Based / Neural), переключатель препроцессинга, статистика извлечения и список триплетов с цветовой разметкой: актор — красный, действие — оранжевый, объект — синий.

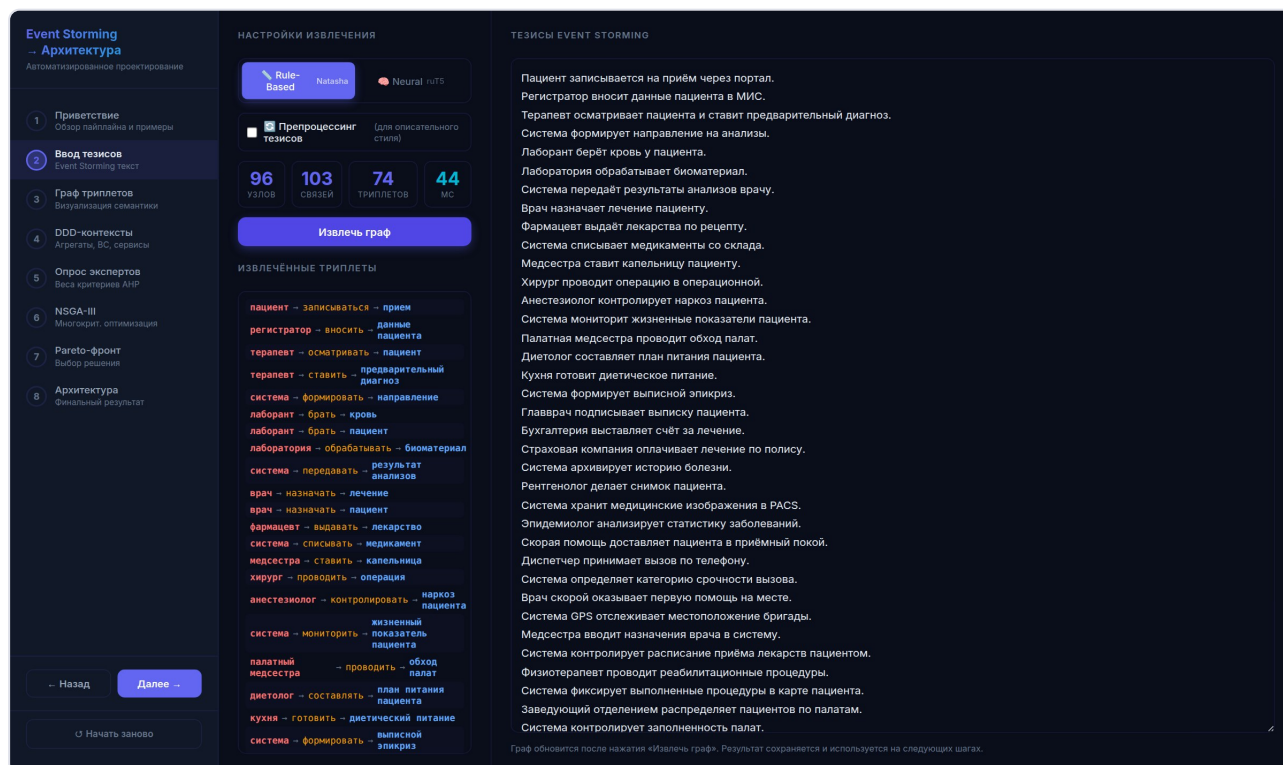


Рис. 1. Ввод тезисов и результат извлечения: 74 триплета, 96 узлов и 103 связи за 44 мс.

3 Граф триплетов

Интерактивный граф предметной области: форма и цвет узла соответствуют его типу (актор, система, сущность, событие, агрегат, read model), рёбра подписаны действиями. Граф можно масштабировать и перетаскивать. Для очень крупных графов (сотни узлов) интерфейс показывает только список триплетов, а визуализация переносится на уровень контекстов (шаг 4).

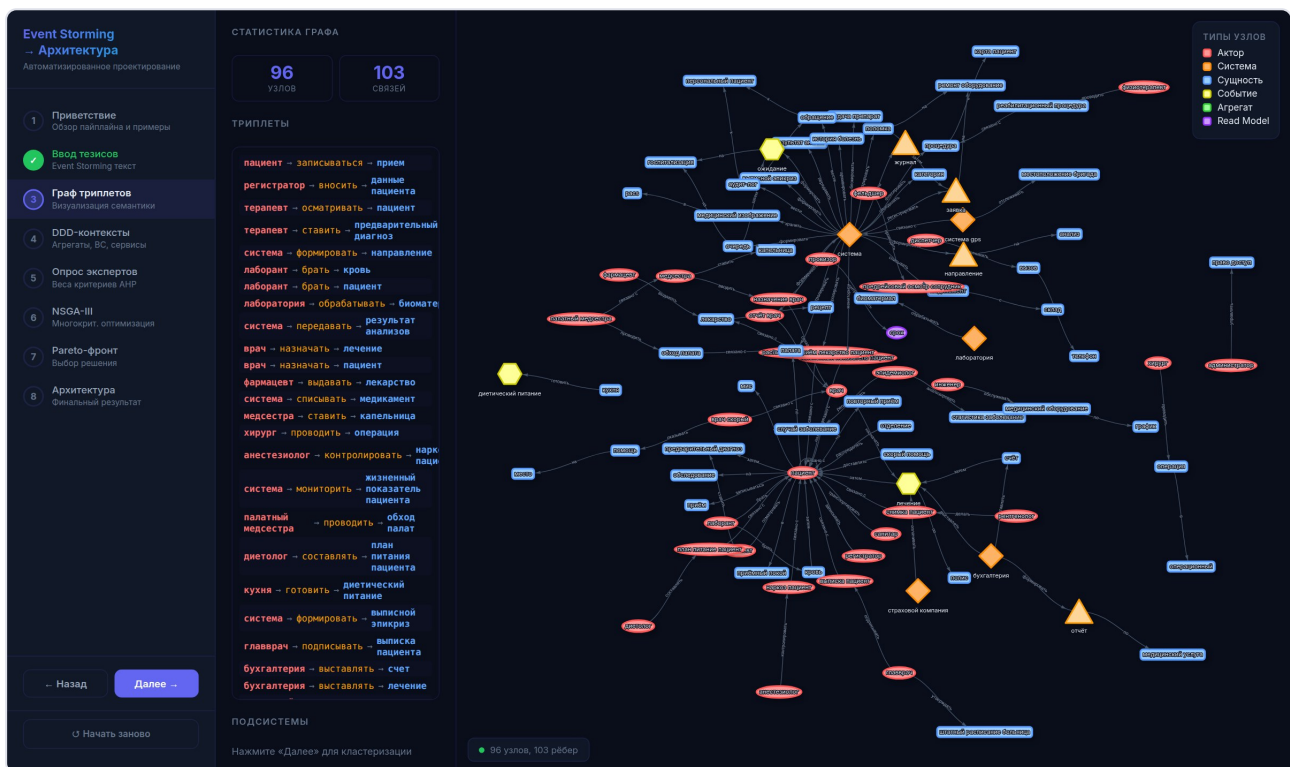


Рис. 2. Семантический граф предметной области с легендой типов узлов.

4 DDD-контексты

Левая панель перечисляет выделенные агрегаты с их командами и акторами, правая — граф ограниченных контекстов: узлы окрашены по контекстам, рёбра отражают команды и общие (shared) сущности между контекстами.

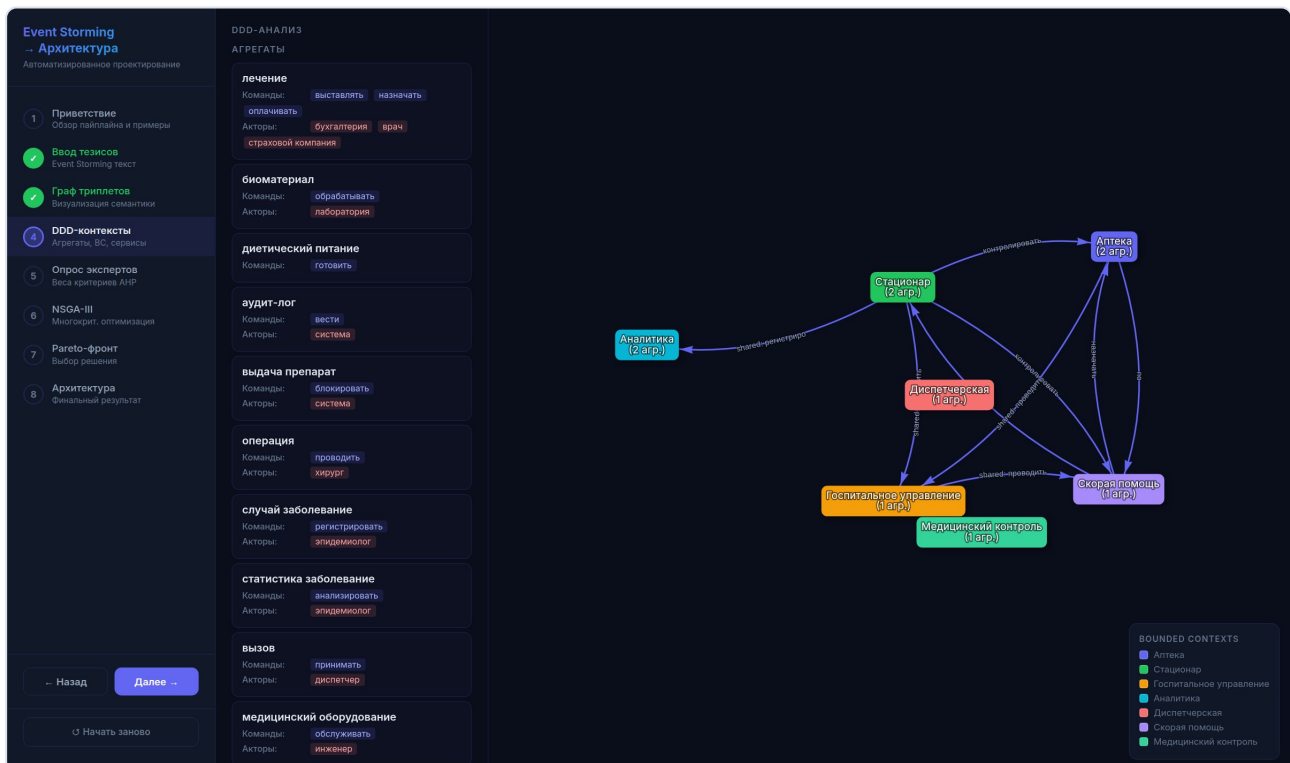


Рис. 3. Результат DDD-анализа: агрегаты и граф ограниченных контекстов.

5 Опрос экспертов (АНР)

Эксперт попарно сравнивает пять критериев качества (сопровожаемость, надёжность, производительность, безопасность, экономичность) по шкале Саати от 1/9 до 9. Система вычисляет веса и индекс согласованности CR; при $CR \leq 0,1$ матрица считается согласованной. Профили сохраняются и могут быть выбраны повторно.

Event Storming
-- Архитектура
Автоматизированное проектирование

1 Приветствие
Опрос экспертов и примеры

2 Ввод тезисов
Event Storming текст

3 Граф триплетов
Визуализация семантики

4 DDD-контексты
Агрегаты, BC, сервисы

5 **Опрос экспертов**
Веса критериев АНР

6 NSGA-III
Многокрит. оптимизация

7 Pareto-фронт
Выбор решения

8 Архитектура
Финальный результат

Назад Далее

Начать заново

Экспертный опрос — веса критериев качества

Укажите относительную важность критериев ISO 25010 для вашего класса системы. Результат используется при выборе решения на Pareto-фронте

СОХРАНЁННЫЕ ПРОФИЛИ

Сохранённых профилей нет — заполните опрос ниже

НОВЫЙ ОПРОС (МЕТОД АНР, МАТРИЦА 5×5)

Имя / роль эксперта
Архитектор ПО

Для каждой пары критериев задайте, во сколько раз первый важнее второго. 1 = одинаково важны, 9 = первый абсолютно важнее, 1/9 = второй абсолютно важнее.

Сопровожаемость	1/2	Надёжность
Сопровожаемость	1	Производительность
Сопровожаемость	1	Безопасность
Сопровожаемость	2	Экономичность
Надёжность	2	Производительность
Надёжность	2	Безопасность
Надёжность	4	Экономичность
Производительность	1	Безопасность
Производительность	2	Экономичность
Безопасность	2	Экономичность

Вычислить веса Сбросить

Рис. 4. Форма попарного сравнения критериев (матрица 5×5).



Рис. 5. Результат АНР: согласованная матрица (CR = 0) с приоритетом надёжности — 36,4 %.

6 Оптимизация NSGA-III

Пользователь задаёт число поколений и число разбиений опорных направлений и запускает оптимизацию. Во время работы в реальном времени обновляются 3D-проекция Парето-фронта (цвет точки — стоимость) и график hypervolume, отражающий сходимость.



Рис. 6. Завершённая оптимизация: 80 поколений, 7 контекстов, Парето-фронт и кривая hypervolume.

7 Выбор решения на Парето-фронте

Система отмечает рекомендованное решение — компромисс, взвешенный по весам эксперта (ромб). Щелчком по любой точке можно посмотреть её оценки по пяти критериям и стили всех контекстов, затем применить решение.

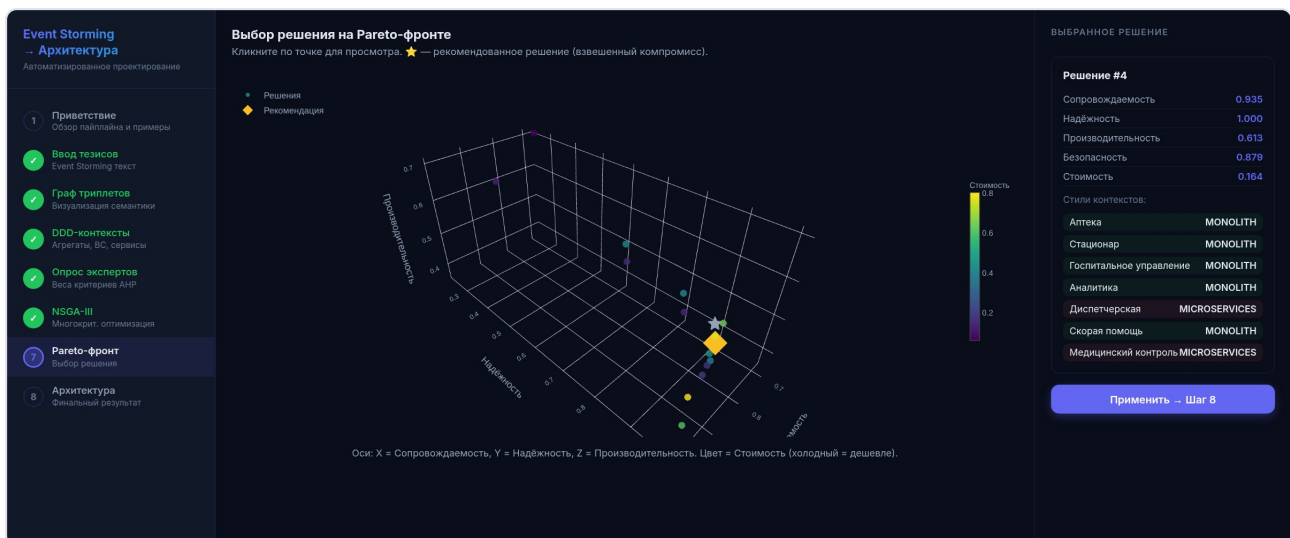


Рис. 7. Выбранное решение: монолит для пяти контекстов, микросервисы для «Диспетчерской» и «Медицинского контроля».

8 Итоговая архитектура

Финальный шаг содержит четыре вкладки. **Граф архитектуры** показывает контексты, окрашенные по стилю, с синхронными и асинхронными связями; справа — состав каждого контекста: протокол, стратегия данных, развёртывание, число образцов, агрегаты, акторы и команды.

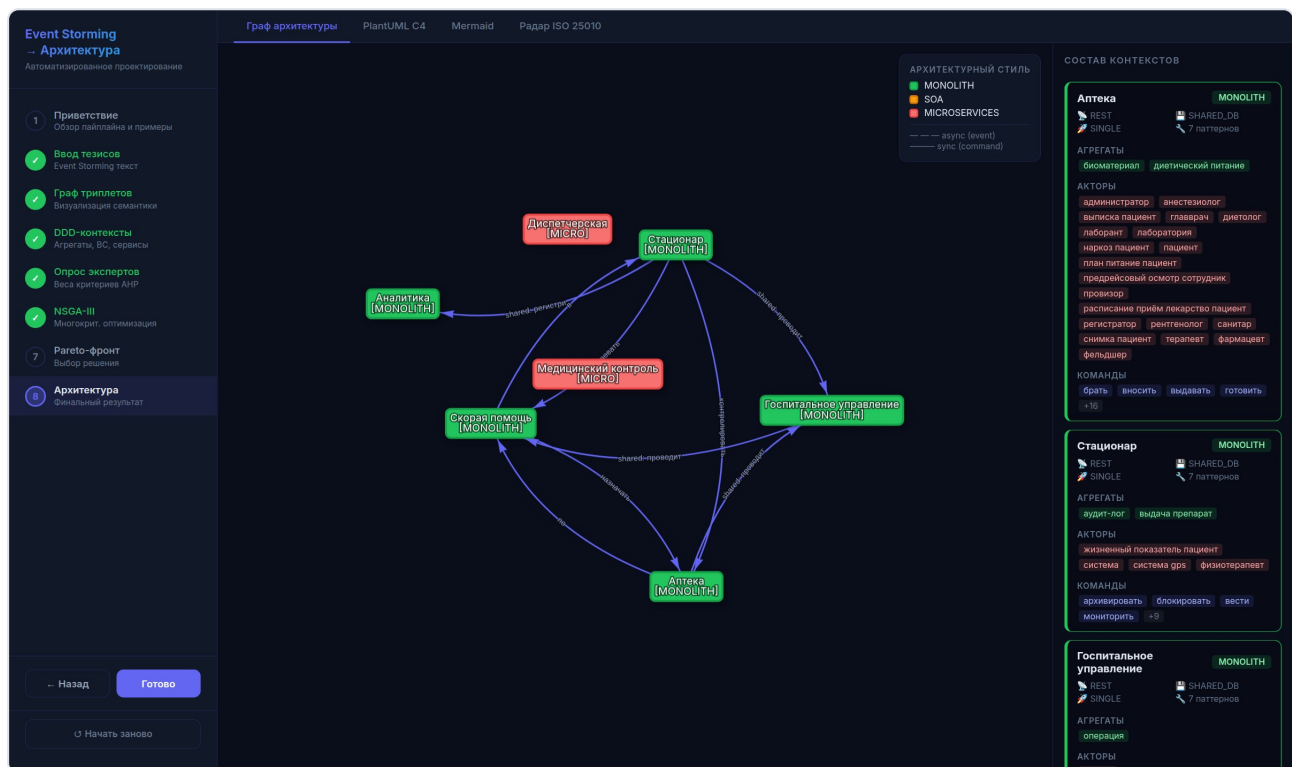


Рис. 8. Граф архитектуры и панель состава контекстов.

Вкладка **PlantUML C4** содержит контейнерную диаграмму C4 с возможностью копирования и скачивания .puml. Для отрисовки диаграммы в виде изображения нужен PlantUML-сервер (docker-compose up -d plantuml); без него, как на снимке, выводится исходный код.

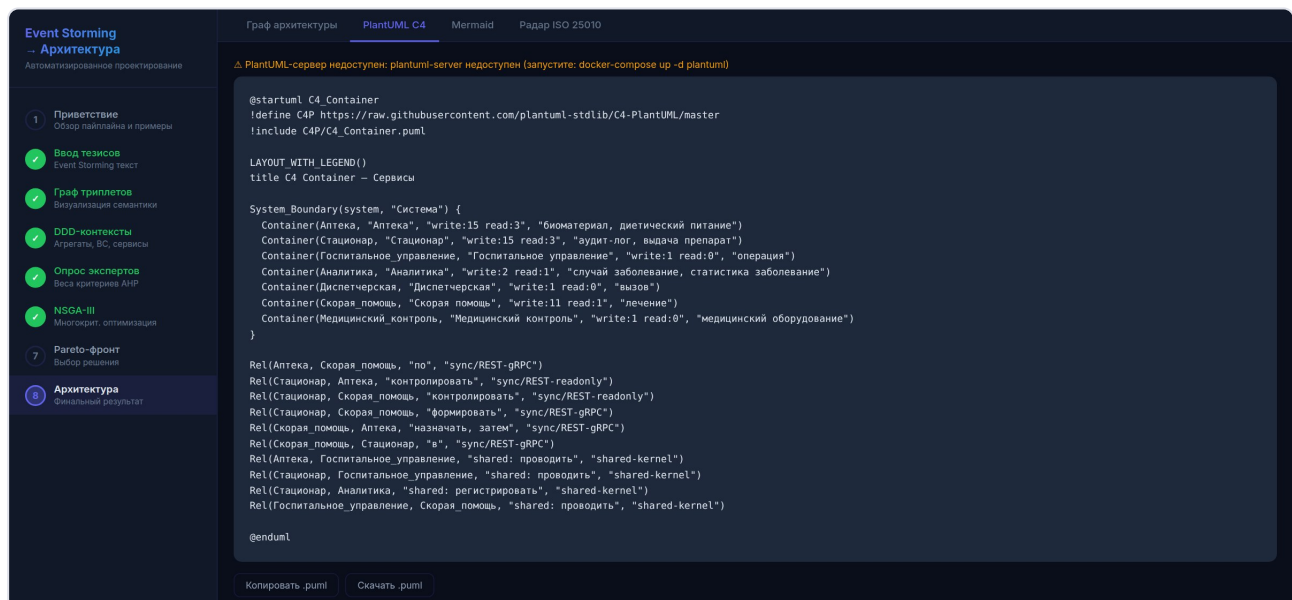


Рис. 9. Сгенерированная диаграмма PlantUML C4 Container.

Вкладка **Радар ISO 25010** сравнивает профиль качества выбранного решения с идеалом и с чисто монолитным вариантом по восьми характеристикам стандарта. Вкладка **Mermaid** содержит ту же модель в нотации Mermaid.

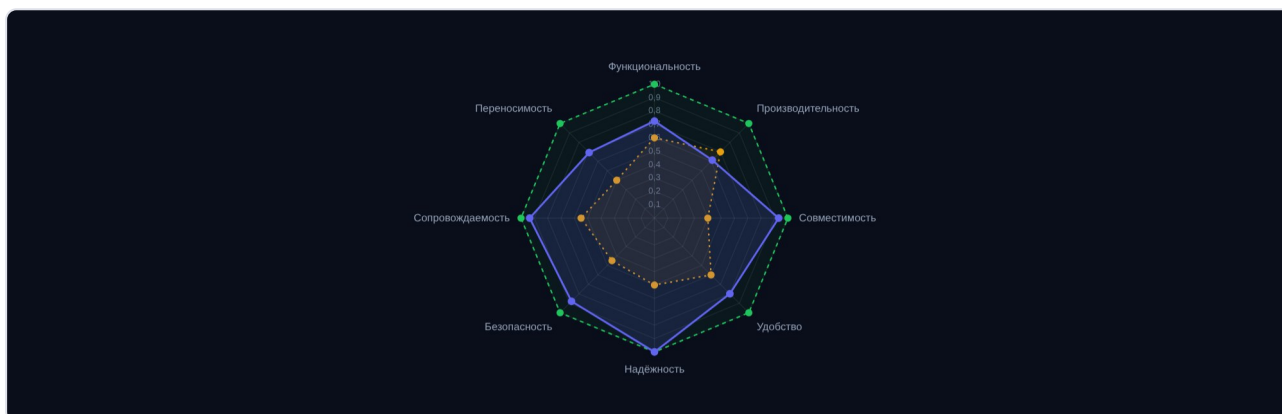


Рис. 10. Профиль качества: выбранное решение (синий), идеал (зелёный), монолит (оранжевый).

4 REST API

Каждый шаг мастера доступен программно. Все запросы и ответы передаются в формате JSON, кроме потоковой оптимизации.

Метод и путь	Назначение
POST /api/extract	Текст → триплеты и граф (режим <code>rule_based</code> / <code>neural</code> , препроцессинг)
POST /api/cluster	Граф → кластеры (подсистемы)
POST /api/analyze	Граф → DDD-структуры: агрегаты, контексты, потоки, API
POST /api/architecture	Граф → ранжированные архитектурные варианты с метриками
POST /api/architecture/decode	Вектор стилей → полное архитектурное решение
POST /api/diagram	DDD-модель → PlantUML C4 и Mermaid
POST /api/diagram/render	Отрисовка PlantUML через PlantUML-сервер
POST /api/survey/ahp	Матрица попарных сравнений → веса критериев и CR
GET /api/profiles, /api/profiles/<id>	Сохранённые профили экспертов
POST /api/optimize/nsga3	Запуск NSGA-III; поток Server-Sent Events с Парето-фронтом и hypervolume по поколениям
GET /api/examples, /api/examples/<id>	Список и тексты демонстрационных кейсов

5 Данные

- `data/` — корпус кейсов бизнес-процессов на русском языке (`case_*.txt`) из разных предметных областей;
- `data_holdout/` — отложенная выборка для независимой проверки;
- `rule_expected/` — эталонные результаты для оценки экстрактора на основе правил;

- `training_data/`, `checkpoints/` — обучающие выборки и веса нейросетевых моделей (не хранятся в репозитории).

6 Установка и запуск

```
cd eventstorming
python -m venv venv && source venv/bin/activate
pip install -r requirements.txt
python server.py — интерфейс откроется по адресу http://localhost:5001
```

Дополнительно:

- `python rule_based/main.py input.txt [--neural] [--preprocess]` — извлечение из файла в командной строке;
- `python scripts/evaluate.py` — оценка экстрактора на эталонах;
- `python -m pytest tests/test_formal_model.py -v` — тесты формальной модели;
- `docker-compose up -d plantuml` — PlantUML-сервер для отрисовки диаграмм C4.