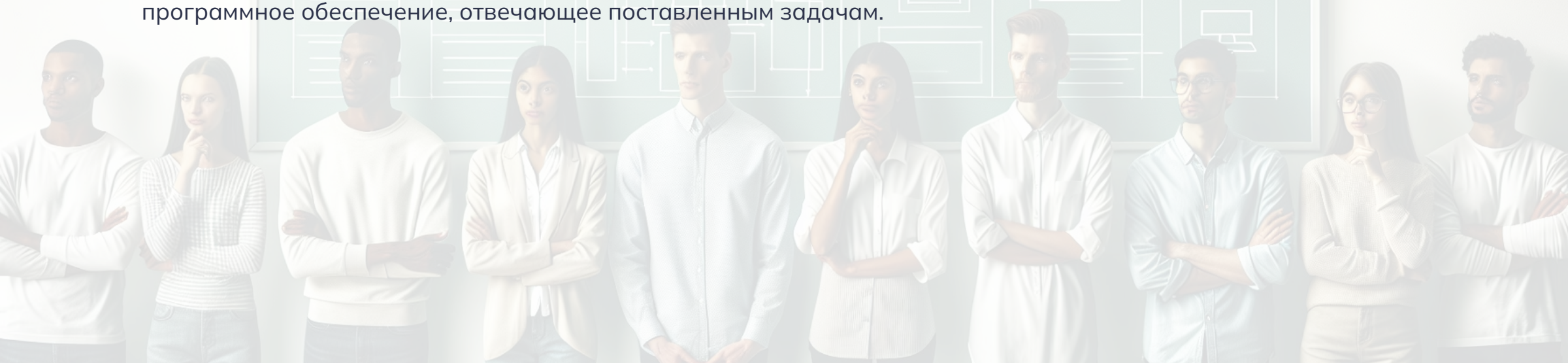
Инжиниринг требований

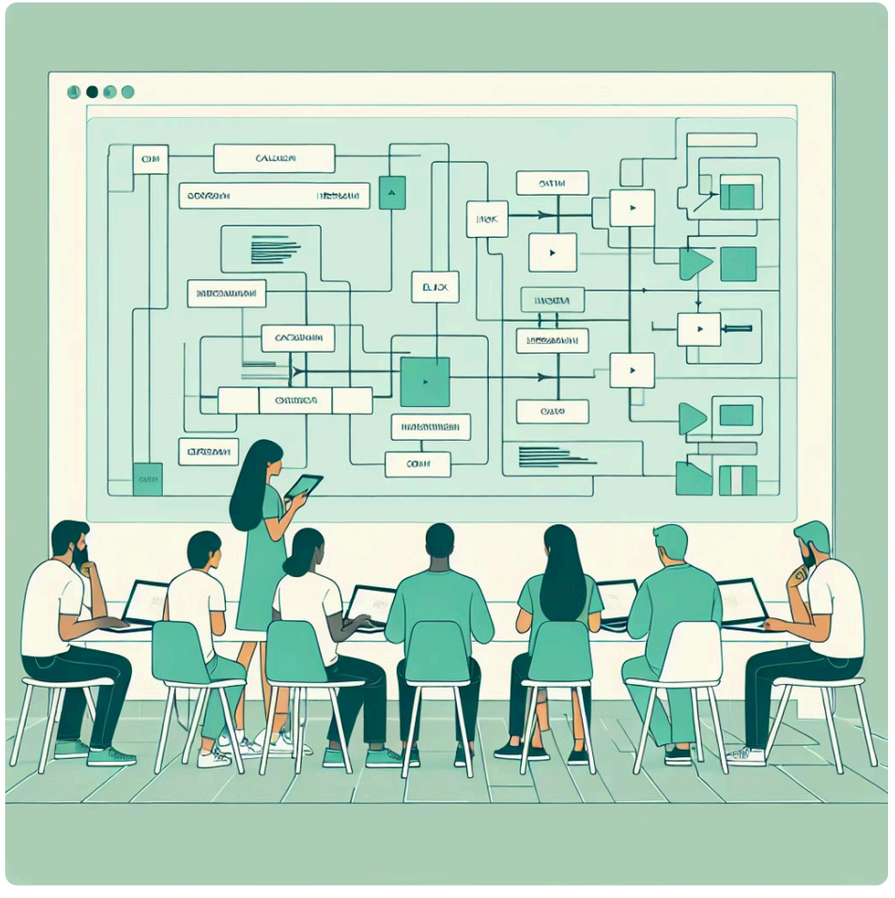
Понять, что нужно создать, часто бывает сложнее, чем создать это. Инжиниринг требований преобразует абстрактные бизнес-идеи в конкретные спецификации, которые помогают командам разработчиков создавать программное обеспечение, отвечающее поставленным задачам.



Вызовы современного сбора требований

Основная проблема

Сбор требований остается сложной задачей для многих команд. Необходимость приступать к разработке до полного понимания требований часто приводит к созданию систем, которые не соответствуют ожиданиям заинтересованных сторон или создают долгосрочные технические долги.



Функциональные требования

Что делает система

Качественные характеристики

Насколько хорошо она работает

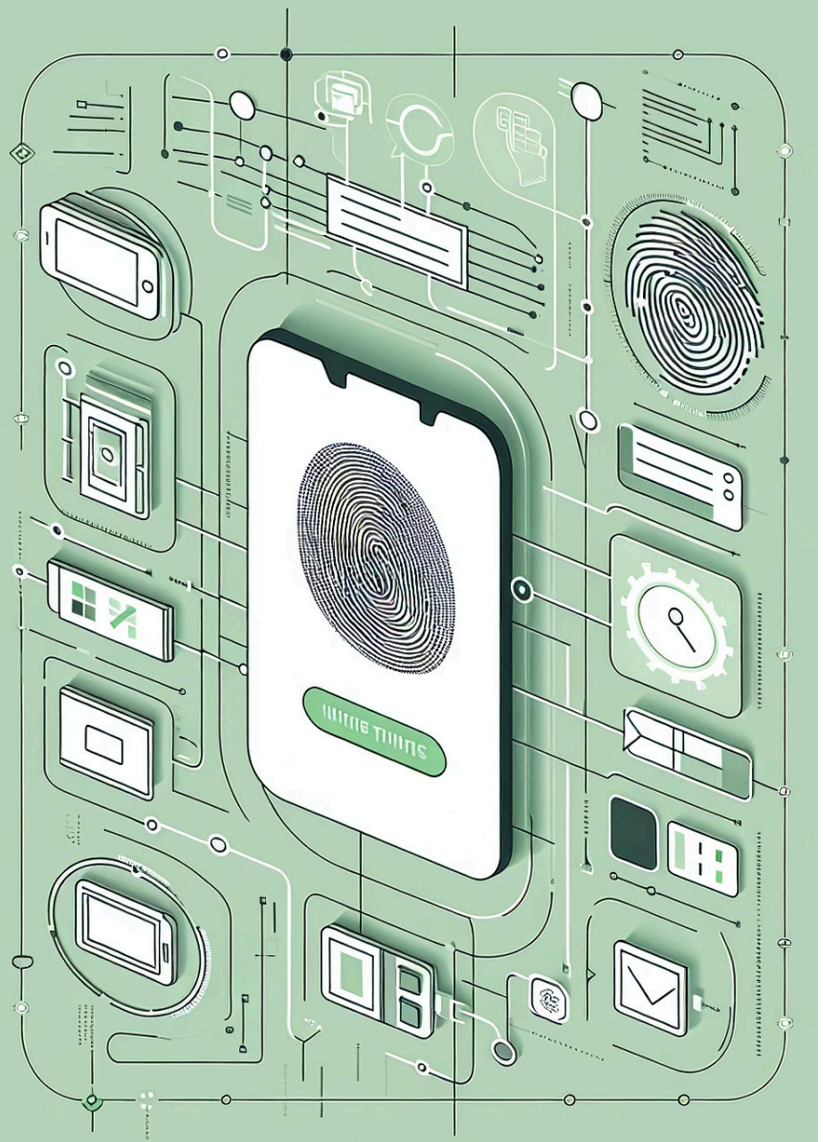
Ограничения

Что ограничивает варианты проектирования

Функциональные требования: что должна делать система?

Функциональные требования описывают конкретные действия, функции и возможности, которые обеспечивают ценность для бизнеса. Хотя они кажутся простыми — «система должна позволять пользователям искать продукты» — дьявол кроется в деталях.

> Эффективное выявление функциональных требований выходит за рамки сбора списка функций и включает понимание лежащих в основе бизнес-процессов, рабочих процессов пользователей и взаимодействий системы.



Структура FAB: функция-преимущество-выгода

Функции

Представляют собой то, что может делать система — неотъемлемые характеристики продукта

Преимущества

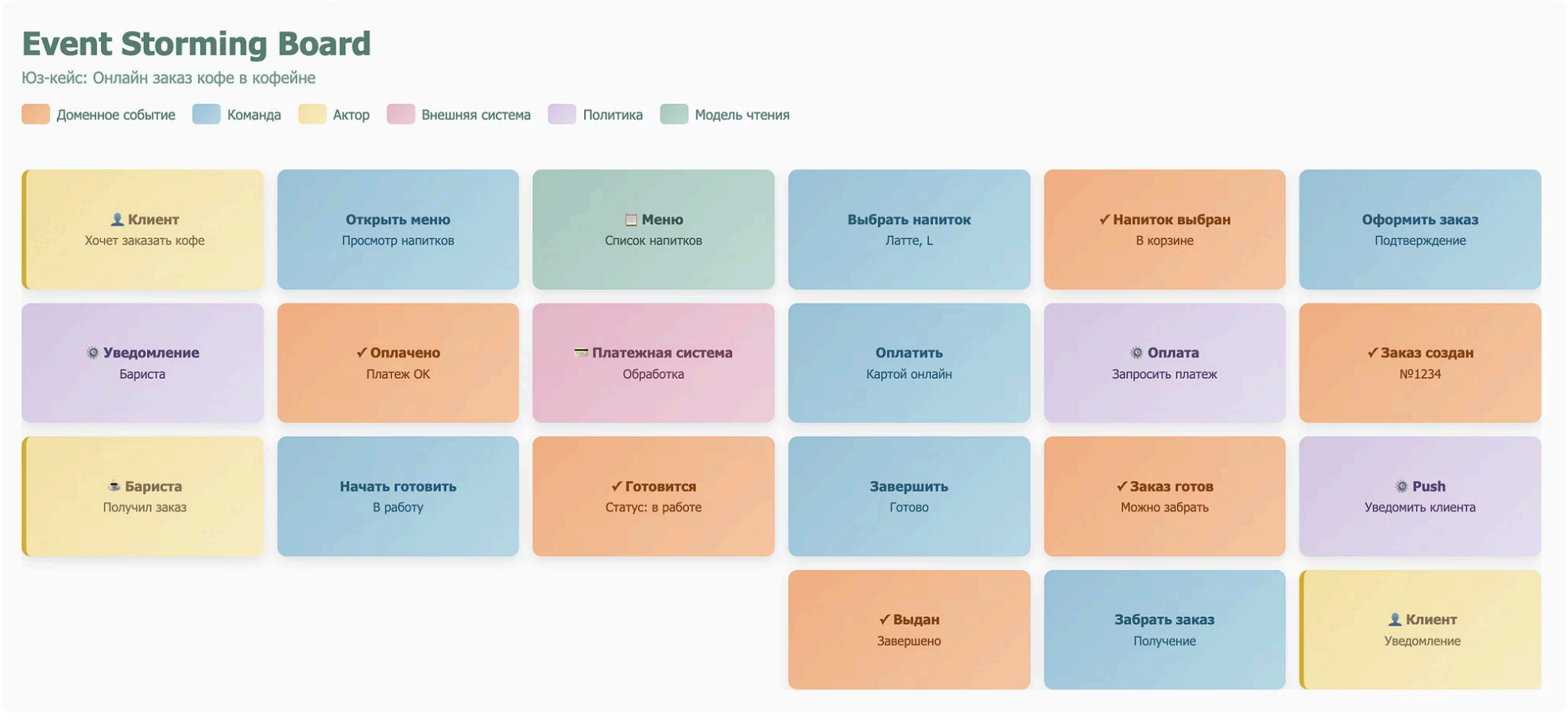
Описывают то, что пользователи получают при использовании этих особенностей

Выгоды

Объясняют, почему пользователи хотят иметь эти функции

Например, в системе мобильных платежей биометрическая аутентификация является функцией. Преимущество заключается в устранении необходимости ввода PIN-кода, а выгоды включают более быстрые транзакции и повышенную безопасность.

Event Storming для обнаружения процессов



Event Storming предоставляет гибкий интерактивный подход к обнаружению бизнес-процессов и функциональных требований, которые их поддерживают. Эта техника совместной работы объединяет различных заинтересованных участников для составления схемы сложных бизнес-процессов с помощью простых цветных стикеров.

Событие происходит

«Оплата завершена»

Анализ причин

Что вызывает события?

Определение участников

Кто их инициирует?

Следующие шаги

Что происходит дальше?

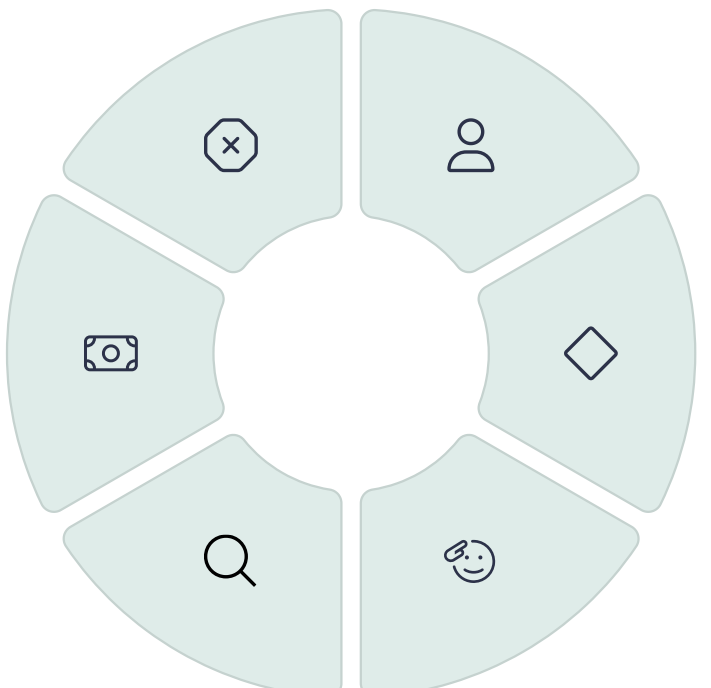
Преимущества Event Storming

Прелесть Event Storming заключается в его способности выявлять скрытые знания. Бизнес-эксперты часто интуитивно понимают процессы, но затрудняются полностью их сформулировать. Разработчики видят технические возможности, но могут упустить нюансы бизнеса.

- Создает общий язык между командами
- Выявляет скрытые функциональные требования
- Показывает взаимосвязи и зависимости
- Объединяет разные точки зрения

Lean Canvas для стратегического контекста

В то время как Event Storming превосходно подходит для функциональных требований на уровне процессов, Lean Canvas предоставляет важный стратегический контекст, который формирует приоритеты требований.



Влиятельные функциональные требования

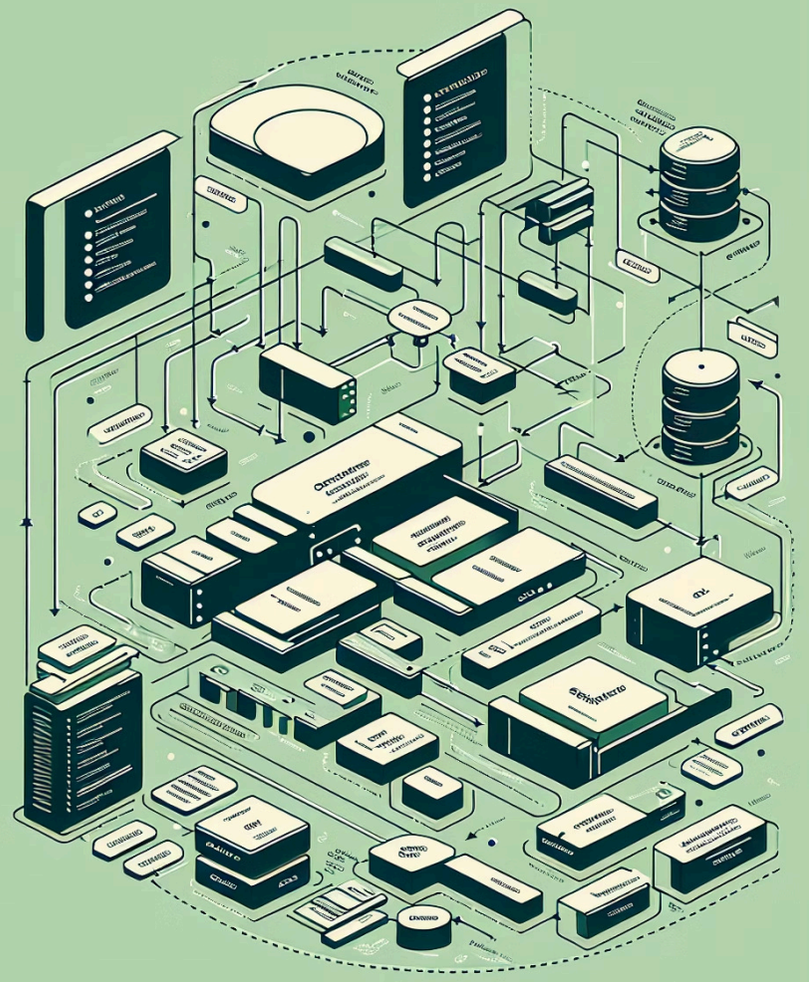
Архитектурная значимость

Не все функциональные требования одинаковы. Только некоторые из них имеют архитектурное значение и вынуждают принимать фундаментальные архитектурные решения, которые влияют на всю систему.

Скрытая сложность

Простая функция входа пользователя может быть понятной, но «пользователи могут просматривать историю своих действий, даже если они потеряли телефон» внезапно вводит множество архитектурных требований.

Важно: Одно, казалось бы, невинное функциональное требование может полностью изменить архитектуру системы.



Критические архитектурные требования

Интеграция

Требуют интеграции с внешними системами

Реальное время

Включают обработку в реальном или почти реальном времени

Безопасность

Обработка конфиденциальных данных или специальные меры безопасности

Мультиплатформенность

Должны работать на нескольких платформах или устройствах

Масштабирование

Требуют масштабирования до значительно более высоких объемов

Практические методы сбора требований

Составление карт пользовательских сценариев

Помогает командам понять функциональные требования в контексте пользовательских сценариев, выявляя недостающие функциональные требования и понимая связи между функциями.

Анкеты и структурированные интервью

Хорошо подходят для сбора подробных функциональных требований после понимания общего объема работ. Ключ — задавать открытые вопросы о целях и мотивации.

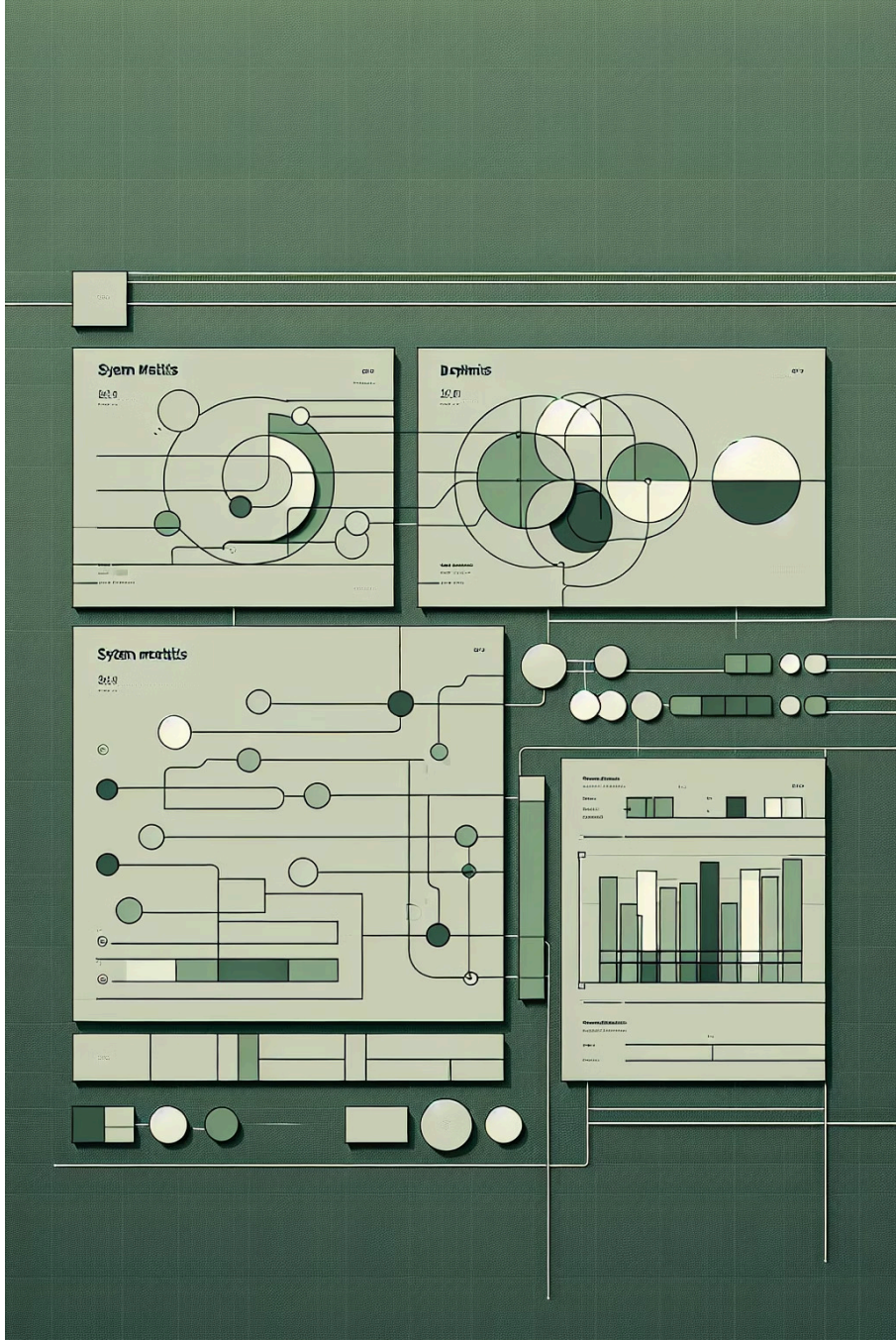
Макеты и прототипы

Делают абстрактные функциональные требования конкретными. Простой прототип может выявить десятки неявных функциональных требований.

Качественные характеристики: насколько хорошо это должно работать?

Нефункциональные требования, также называемые качественными характеристиками, определяют, насколько хорошо система должна выполнять свои функции. В отличие от функциональных требований, которые часто можно добавлять постепенно, нефункциональные требования обычно требуют архитектурных решений, которые должны быть приняты с самого начала.

> Они часто являются разницей между системой, которая работает в демо-версии, и системой, которая работает в производственной среде.



Проблема невидимых требований

Время отклика

Пользователи не задумываются о времени отклика, пока страницы не загружаются медленно

1

Безопасность

Не беспокоятся о безопасности, пока не происходит утечка данных

3

2

Доступность

Не думают о доступности, пока система не выходит из строя

Эта незаметность делает их легко упускаемыми из виду при сборе требований, что впоследствии приводит к болезненным последствиям.

Сценарии качественных атрибутов

Сценарии качественных атрибутов предоставляют структурированный способ сделать нефункциональные требования явными и поддающимися тестированию. «Масштабируемость» и «производительность» — это всего лишь слова, пока мы не определим конкретные, измеримые сценарии.

Источник

Кто или что инициирует сценарий

Стимул

Событие, требующее реакции системы

Артефакт

Тестируемый компонент системы

Среда

Контекст работы

Реакция

Ожидаемое поведение системы

Показатель

Конкретные, измеримые критерии успеха

Пример сценария качества

> Когда **1000 одновременных пользователей** (источник) отправляют **поисковые запросы** (стимул) во время **пиковой нагрузки** (среда), **служба поиска** (артефакт) возвращает результаты в течение **500 мс для 95% запросов** (реакция и показатель реакции).

Принцип: Хорошие сценарии являются конкретными, измеримыми и поддающимися тестированию. Если вы не можете написать тест для своего сценария, он не является достаточно конкретным.

Семинар по качественным атрибутам (QAW)

Семинар по качественным атрибутам предоставляет структурированный метод для выявления и приоритизации нефункциональных требований на ранних этапах разработки. В отличие от традиционного сбора требований, ориентированного на функциональные особенности, QAW специально нацелен на качественные атрибуты.

Презентация бизнеса

Устанавливает контекст и цели, помогая техническим командам понять важность атрибутов качества для бизнеса

Презентация архитектуры

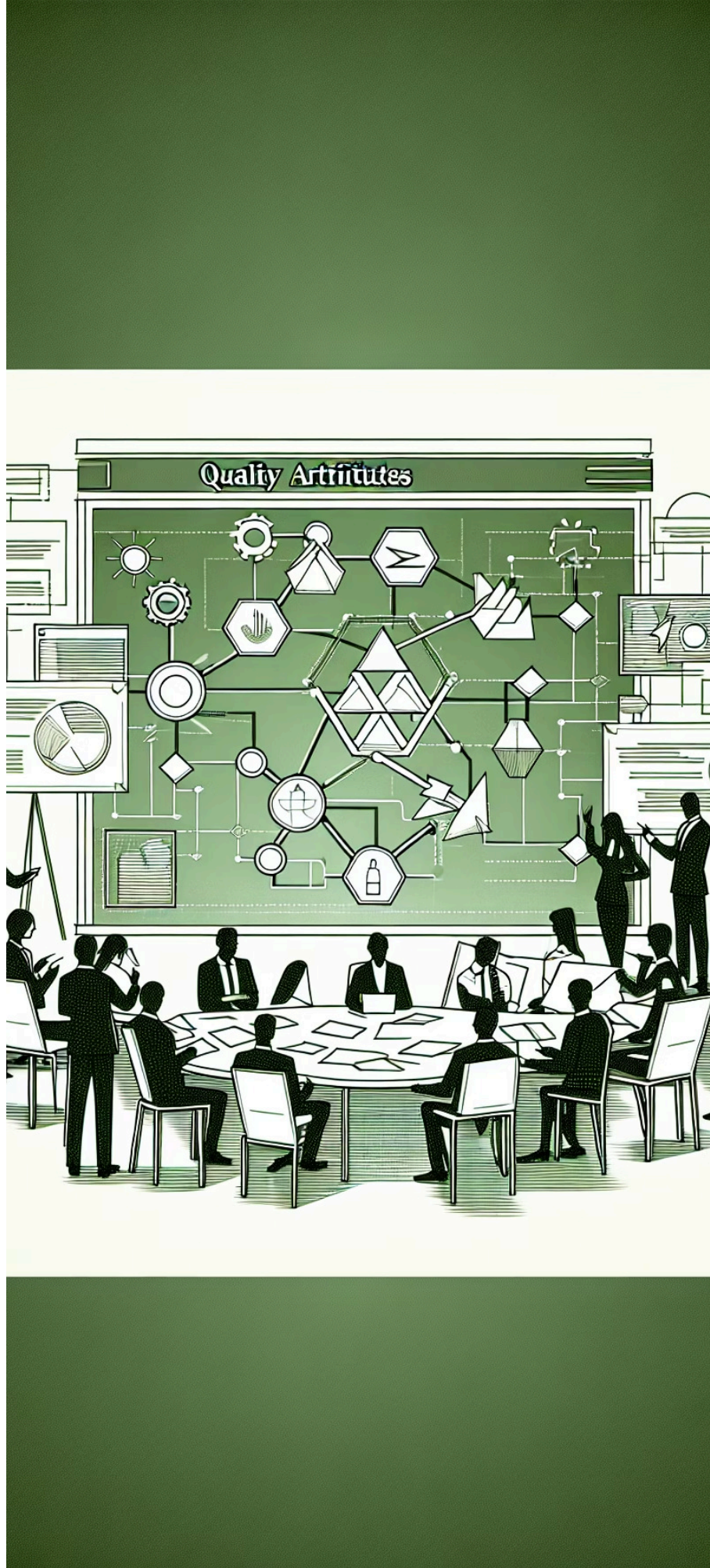
Дает заинтересованным сторонам контекст о текущем или предлагаемом дизайне системы

Мозговой штурм сценариев

Генерирует сценарии атрибутов качества от всех участников

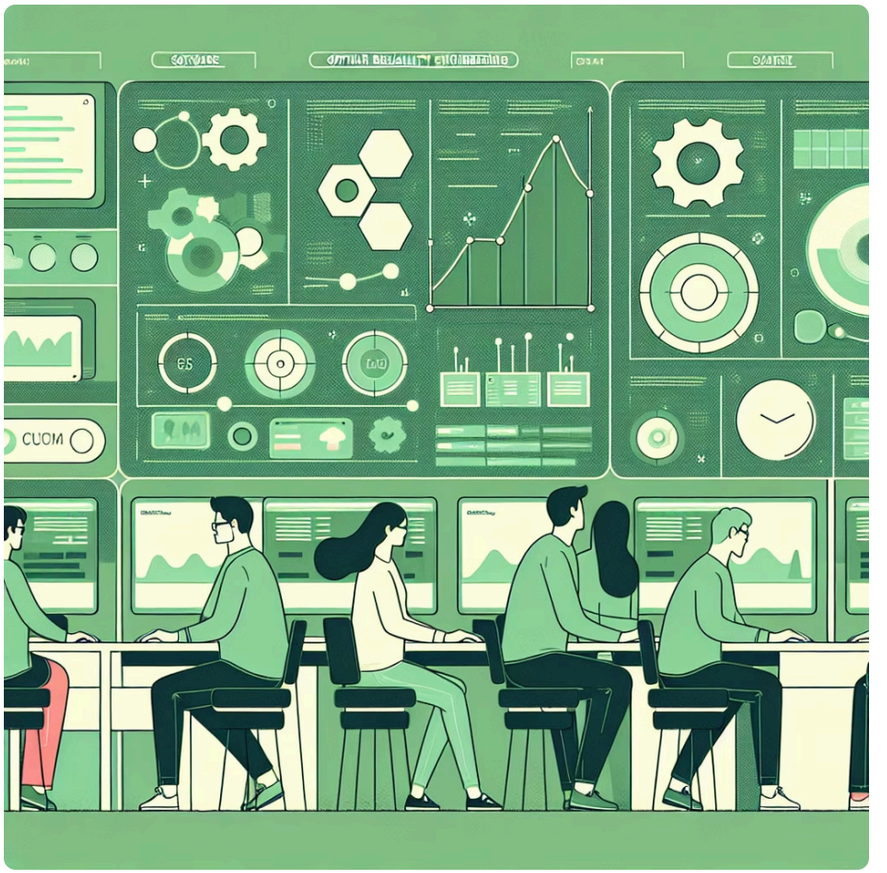
Консолидация и приоритизация

Устраняет дублирование и определяет наиболее важные атрибуты качества



Подходы инженерии надежности сайта (SRE)

Инженерия надежности сайта предоставляет практические рамки для измерения и управления атрибутами качества в производственных системах. Подход SRE соединяет разработку и эксплуатацию, применяя инженерные принципы к задачам обеспечения надежности.



SLI - Показатели уровня обслуживания

Конкретные измеримые метрики, отражающие состояние сервиса с точки зрения пользователя

SLO - Цели уровня обслуживания

Устанавливают конкретные целевые показатели SLI на определенные периоды времени

Бюджеты ошибок

Количественно определяют допустимую ненадежность в рамках ограничений SLO

Методы обнаружения качественных характеристик

Интервью с заинтересованными сторонами

Разные заинтересованные стороны заботятся о разных атрибутах качества: конечные пользователи фокусируются на производительности, операционные команды беспокоятся о надежности, руководители заботятся о стоимости.

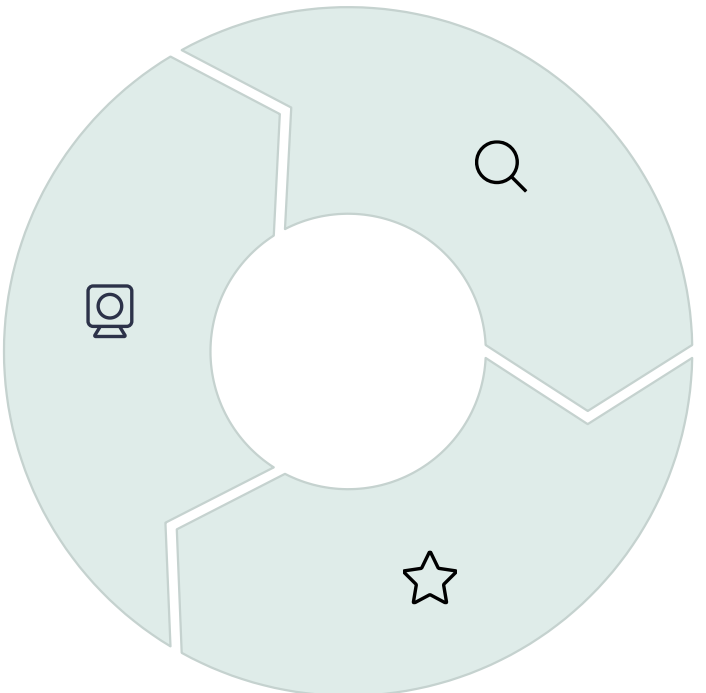
Анализ архитектурных рисков

Изучение предлагаемых или существующих архитектур на предмет рисков, связанных с показателями качества. Помогает выявить, где архитектурные решения могут повлиять на показатели качества.

Компромиссы между показателями качества

Безопасность

Увеличивает задержки



Высокая доступность

Требует избыточности и сложности

Масштабируемость

Может привести к потере согласованности

Показатели качества редко совпадают идеально. Наиболее успешные системы оптимизируют нужные качественные характеристики для конкретного контекста, а не пытаются максимизировать все.

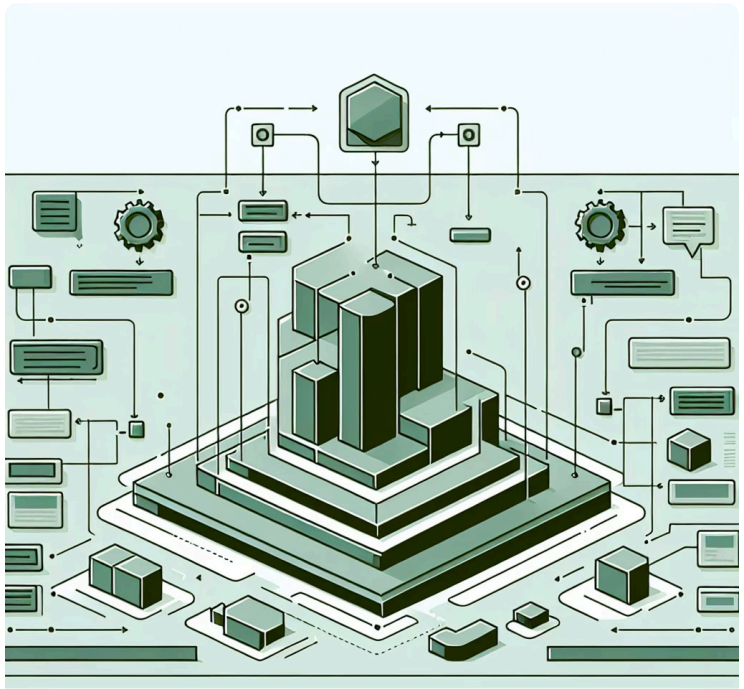
Ключевые принципы успешного инжиниринга требований

Каждый тип требований требует разных подходов к выявлению. Цель — не идеальные требования, а общее понимание между бизнес-заинтересованными сторонами и техническими специалистами.

Создание правильного продукта правильным способом

Инжиниринг требований — основа успешных программных систем. Три типа требований — функциональные, качественные и ограничения — взаимодействуют друг с другом, определяя, что нужно создать, насколько хорошо это должно работать и какие границы определяют решение.

Требования эволюционируют по мере развития систем и изменения бизнес-контекста. Наиболее успешные системы построены на процессах, которые учитывают эту эволюцию, сохраняя при этом фокус на предоставлении реальной ценности.



Предотвращение сбоев

Правильный инжиниринг требований предотвращает дорогостоящие ошибки и переделки

Основа для успеха

Создает прочную основу для исключительного успеха проекта

Реальная ценность

Обеспечивает фокус на предоставлении настоящей ценности пользователям

> При правильном подходе инжиниринг требований не только предотвращает сбои, но и создает основу для исключительного успеха.