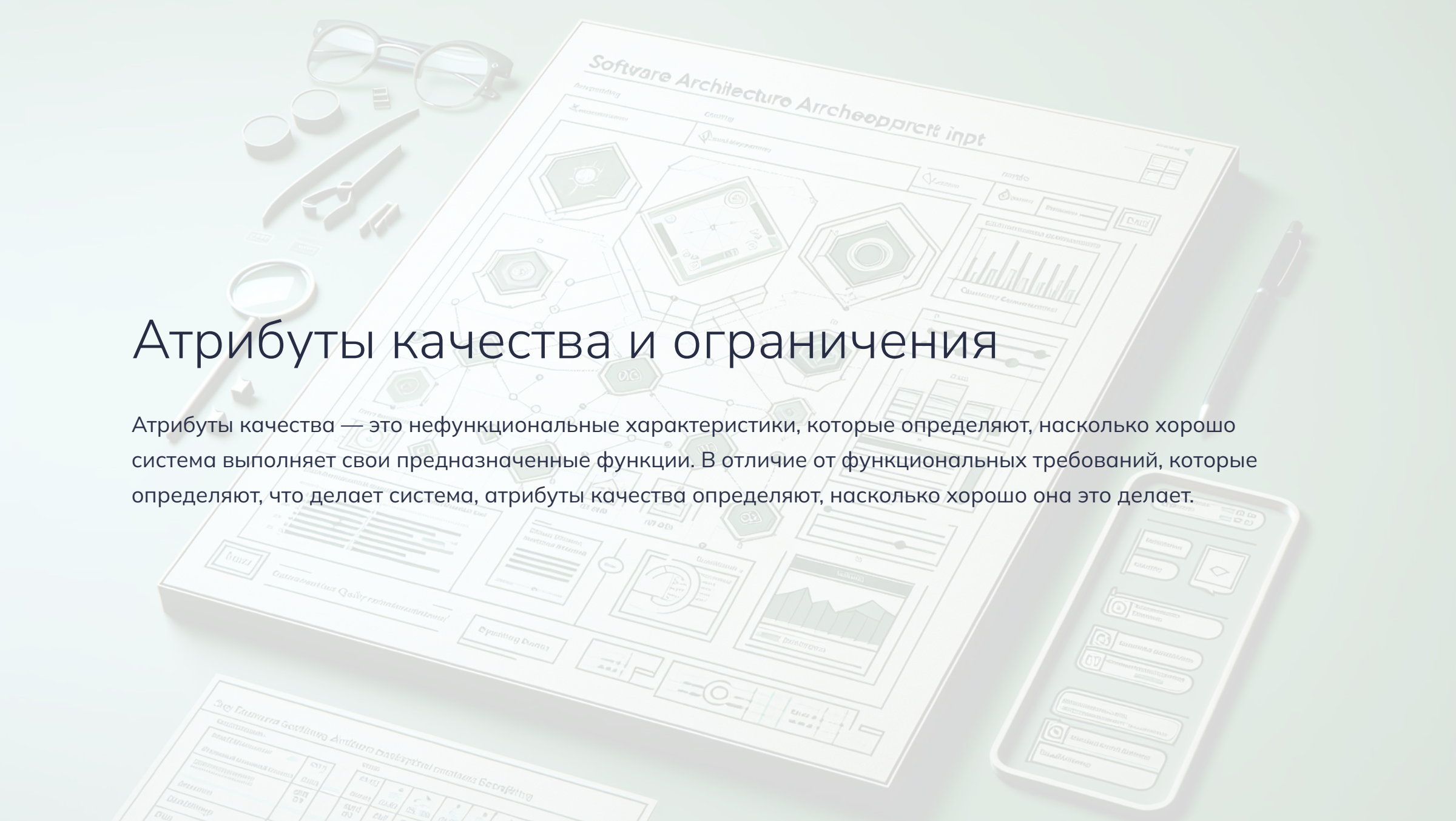


Атрибуты качества и ограничения

Атрибуты качества — это нефункциональные характеристики, которые определяют, насколько хорошо система выполняет свои предназначенные функции. В отличие от функциональных требований, которые определяют, что делает система, атрибуты качества определяют, насколько хорошо она это делает.

Фундаментальные принципы качества

Функциональные требования

Определяют **что** делает система

- Конкретные возможности
- Бизнес-логика
- Пользовательские сценарии

Атрибуты качества

Определяют **насколько хорошо** система работает

- Производительность
- Безопасность
- Масштабируемость

Архитектурные решения

Основаны на **компромиссах** между атрибутами

- Баланс приоритетов
- Оптимизация под цели
- Управление противоречиями

Производительность системы

Определение

Насколько эффективно система использует ресурсы для выполнения временных требований.

Три основных измерения

- **Задержка** — время обработки одного запроса
- **Пропускная способность** — количество запросов в единицу времени
- **Использование ресурсов** — эффективность использования ЦП, памяти и сети

Методы улучшения

- Кэширование часто используемых данных
- Асинхронная обработка длительных операций
- Пул подключений к базе данных
- Оптимизация запросов к БД

Доступность системы

99%	99.9%	99.99%
Базовый уровень	Бизнес-уровень	Критический уровень
3,65 дня простоя в год. Подходит для внутренних инструментов и некритичных систем.	8,76 часа простоя в год. Стандарт для большинства бизнес-приложений.	52,6 минуты простоя в год. Требуется для электронной коммерции и финансовых услуг.

Для достижения высокой доступности требуются избыточность, механизмы обнаружения неисправностей, стратегии плавного снижения производительности и надежные системы мониторинга.

Масштабируемость архитектуры

Горизонтальное масштабирование

Добавление большего количества машин для обработки нагрузки. Хорошо подходит для безсостоятельных сервисов.

- Распределение нагрузки
- Отказоустойчивость
- Гибкость ресурсов

Вертикальное масштабирование

Увеличение мощности существующих машин. Подходит для состоятельных сервисов с умеренным ростом нагрузки.

- Простота реализации
- Сохранение архитектуры
- Ограниченный потенциал роста

Проблемы масштабируемости

Согласованность данных

Поддержание целостности информации между несколькими узлами системы при одновременных операциях записи и чтения.

Управление сессиями

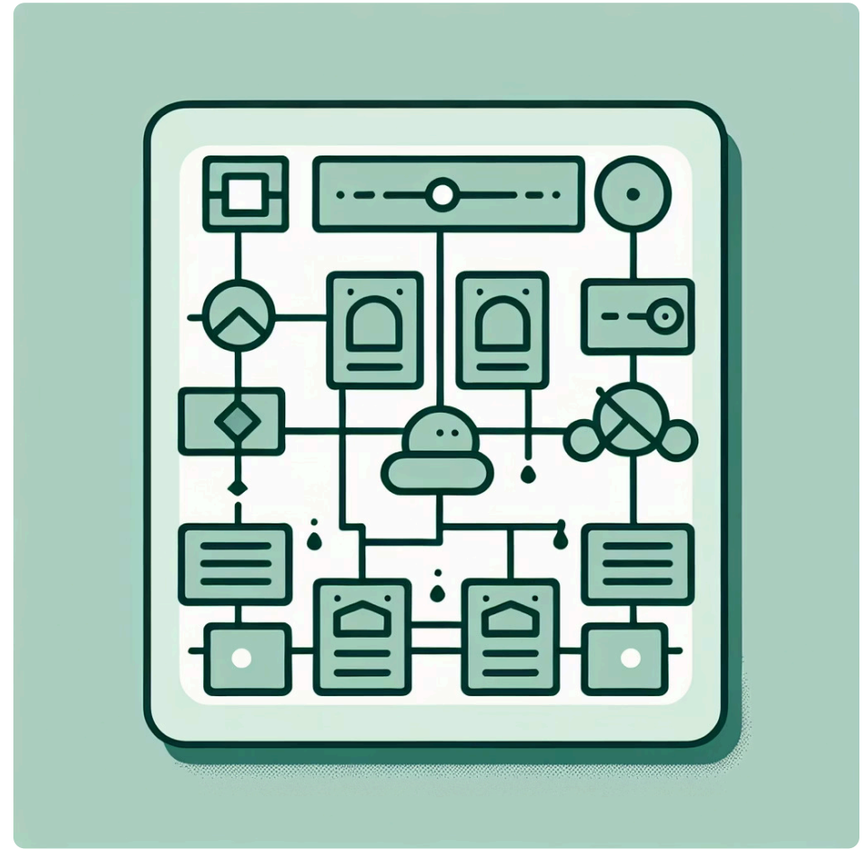
Обеспечение непрерывности пользовательского опыта в архитектурах без сохранения состояния при распределении запросов.

Сложность координации

Синхронизация работы множественных компонентов и сервисов для достижения согласованного поведения системы.

Ключевые стратегии решения

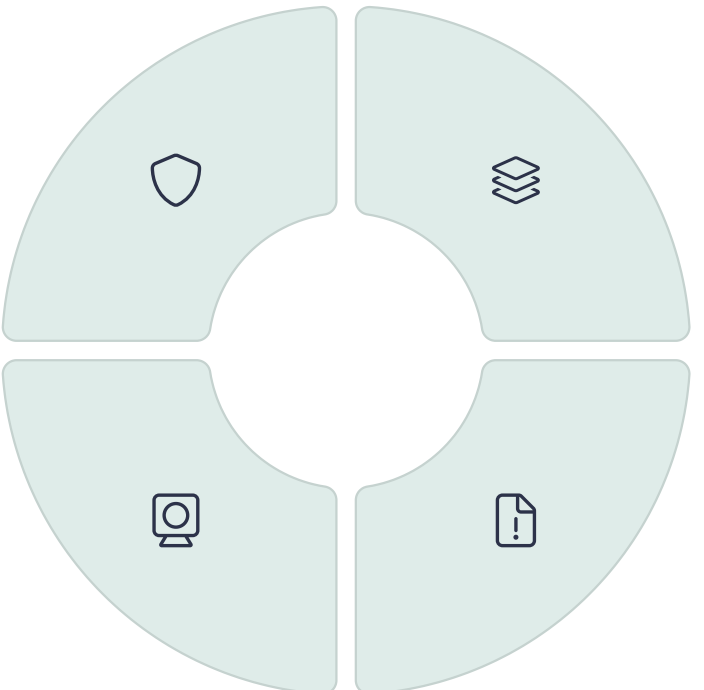
- Фрагментация баз данных
- Архитектуры микросервисов
- Событийно-ориентированная архитектура
- Тщательное управление состоянием



Безопасность системы

Минимальные привилегии

Предоставление минимально необходимых разрешений для выполнения задач



Многоуровневая защита

Несколько независимых уровней безопасности для комплексной защиты

Безопасность по дизайну

Встраивание безопасности в систему на этапе проектирования

Безопасность при сбое

Переход в безопасное состояние по умолчанию при возникновении ошибок

Меры безопасности

01

Аутентификация и авторизация

Системы проверки подлинности пользователей и контроля доступа к ресурсам на основе ролей и разрешений.

02

Проверка вводимых данных

Валидация и санитизация всех входящих данных для предотвращения инъекций и других атак.

03

Шифрование данных

Защита конфиденциальной информации при хранении и передаче с использованием современных алгоритмов.

04

Безопасные протоколы

Использование защищенных каналов связи и протоколов для всех сетевых взаимодействий.

05

Ведение журналов аудита

Комплексное логирование всех действий пользователей и системных событий для мониторинга и расследования.

Надежность системы

Определение

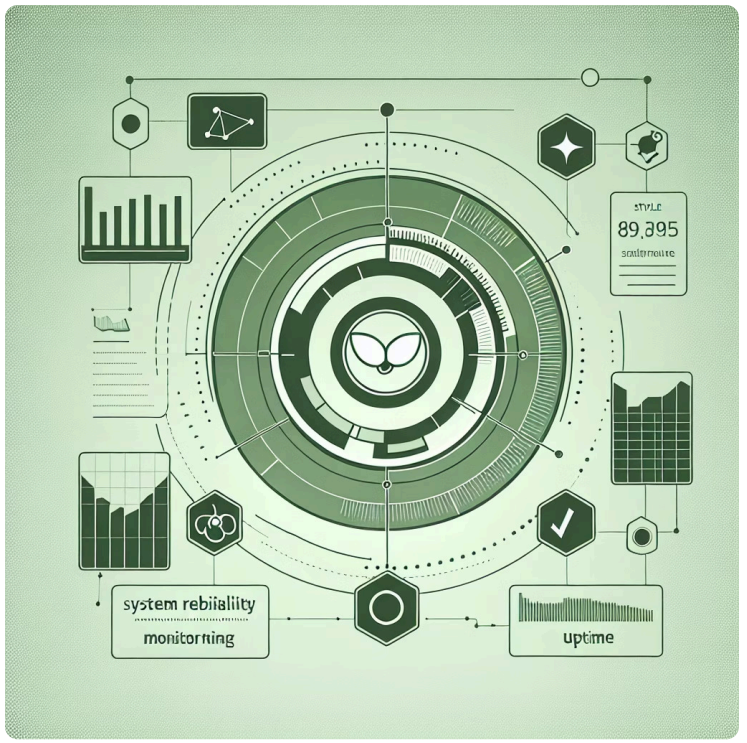
Вероятность того, что система будет работать правильно в течение определенного периода времени.

Ключевые показатели

- **MTBF** — среднее время между сбоями
- **MTTR** — среднее время восстановления
- **Доступность** — процент времени работоспособности

Характеристики надежных систем

- Корректная обработка ошибок
- Автоматическое восстановление после сбоев
- Сохранение целостности данных



Избыточность

Дублирование критических компонентов

Обнаружение ошибок

Мониторинг и раннее выявление проблем

Контрольные точки

Сохранение состояния для восстановления

Комплексное тестирование

Всесторонняя проверка функциональности

Модифицируемость архитектуры

Модифицируемость определяет легкость внесения изменений в систему и гибкость для адаптации к будущим требованиям. Системы с хорошей модифицируемостью имеют четко определенные границы модулей, минимальную зависимость между компонентами и четкие уровни абстракции.

Скрытие информации

Инкапсуляция внутренних деталей реализации за стабильными интерфейсами

Поддержание интерфейсов

Обеспечение стабильности контрактов между компонентами системы

Ограничение путей связи

Минимизация количества зависимостей между модулями

Использование посредников

Применение паттернов для уменьшения прямой связи компонентов

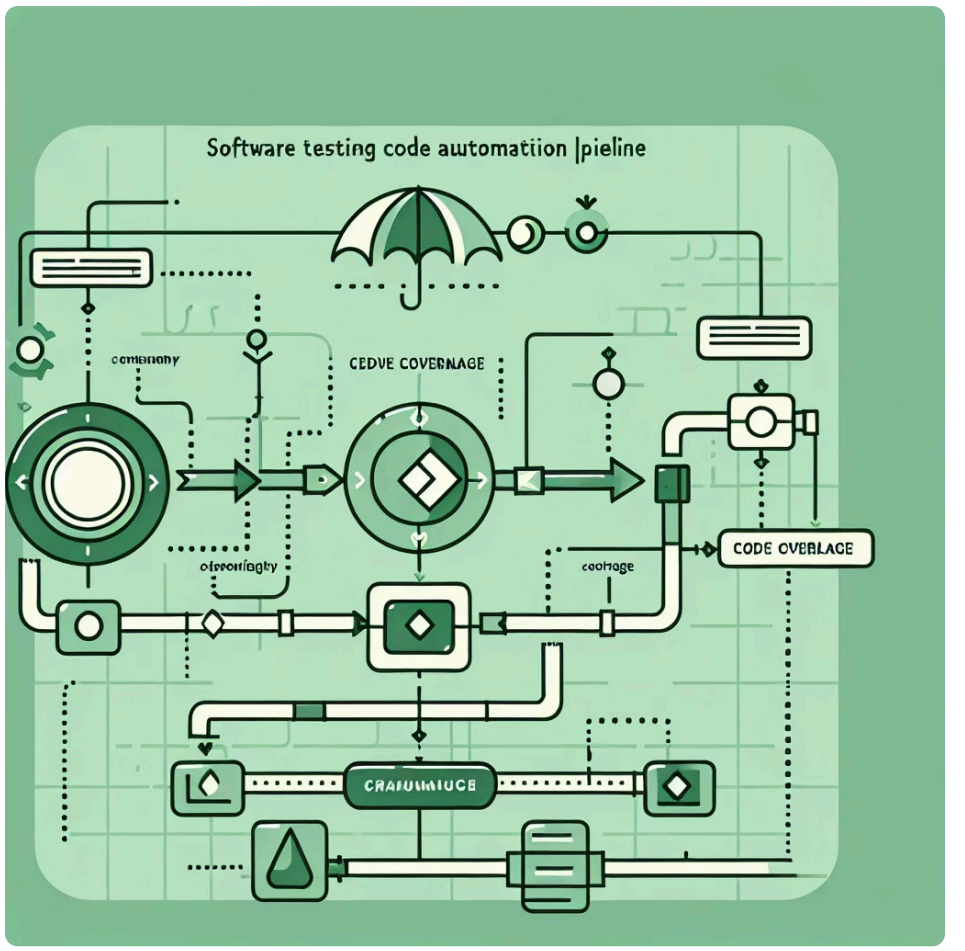
Тестируемость системы

Определение

Легкость, с которой программное обеспечение может продемонстрировать свои недостатки посредством тестирования.

Характеристики тестируемых систем

- Хорошая observability внутреннего состояния
- Контролируемые входы и выходы
- Минимальные сложные взаимозависимости



19

Внедрение зависимостей

Позволяет легко заменять компоненты моками для изолированного тестирования

Разделение задач

Четкое разграничение ответственности упрощает тестирование отдельных компонентов

Избегание глобального состояния

Предсказуемое поведение компонентов без скрытых зависимостей

Четкие интерфейсы

Хорошо определенные контракты облегчают создание тестовых сценариев

Удобство использования

Дизайн интерфейса

Интуитивно понятные элементы управления, логичная навигация и визуальная иерархия, которые помогают пользователям эффективно достигать своих целей.

Отзывчивость системы

Быстрое время отклика на действия пользователя, плавные переходы и немедленная обратная связь для создания комфортного опыта взаимодействия.

Обработка ошибок

Понятные сообщения об ошибках, предложения по исправлению и graceful degradation для поддержания работоспособности системы.

С архитектурной точки зрения, удобство использования влияет на решения, касающиеся отзывчивости системы, стратегий обработки ошибок, форматов представления данных и интеграции с внешними системами.

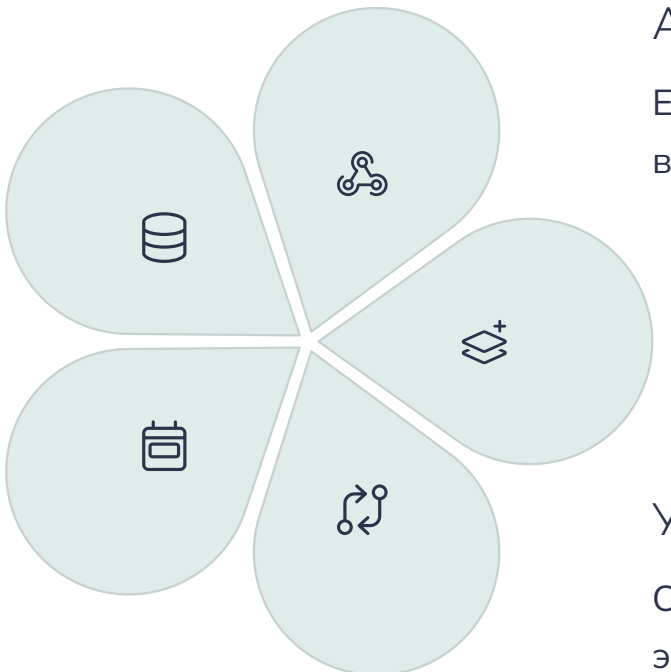
Взаимодействие систем

Обмен данными

Стандартизированные форматы и протоколы

Преобразование данных

Адаптация форматов между системами



API интеграция

Единообразные интерфейсы взаимодействия

Совместимость платформ

Работа в различных технологических средах

Управление версиями

Обратная совместимость и эволюция интерфейсов

Взаимодействие позволяет системам работать вместе, обмениваясь данными и функциональными возможностями между различными платформами, технологиями и организациями.

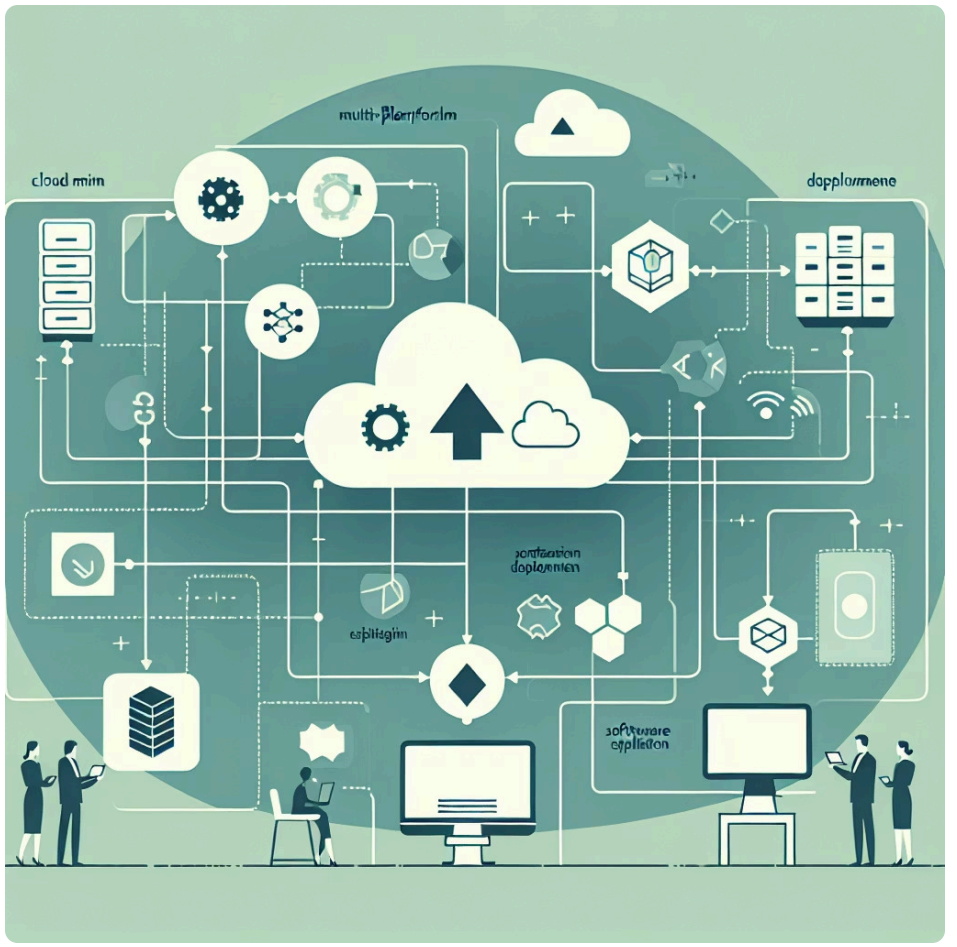
Переносимость архитектуры

Определение

Легкость, с которой система может быть перенесена из одной среды в другую.

Области применения

- Различные аппаратные платформы
- Разные операционные системы
- Альтернативные базы данных
- Облачные и гибридные среды



Стандартные API

Использование общепринятых интерфейсов вместо проприетарных решений

1

Уровни абстракции

Изоляция платформозависимого кода через архитектурные слои

2

Контейнеризация

Упаковка приложений с зависимостями для единообразного развертывания

3

Мультиоблачность

Поддержка развертывания в различных облачных провайдерах

4

Компромиссы между атрибутами

Безопасность ↔️ Производительность

Меры безопасности увеличивают задержки и снижают пропускную способность системы

Доступность ↔️ Сложность

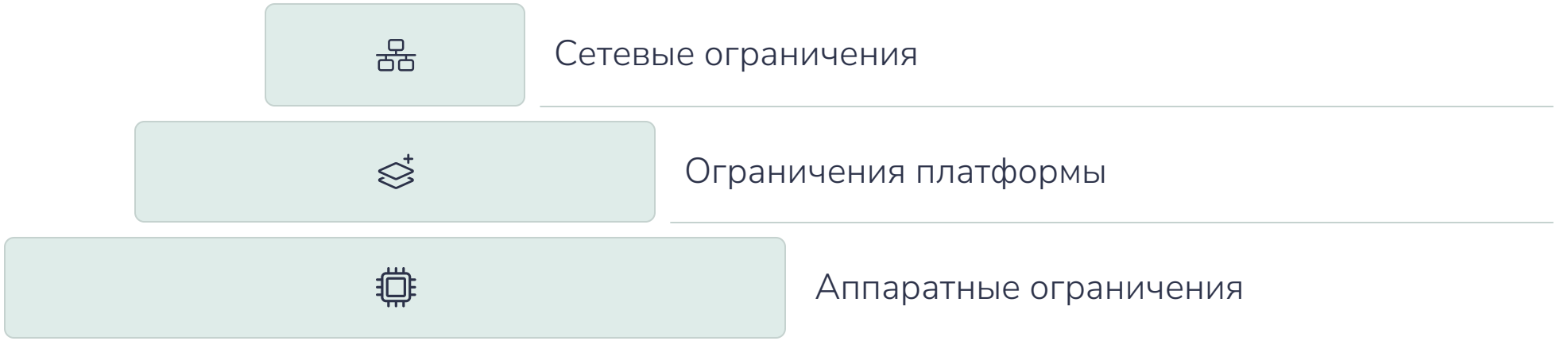
Высокая доступность требует избыточности, что увеличивает архитектурную сложность

Масштабируемость ↔️ Согласованность

Распределенные системы жертвуют строгой согласованностью ради доступности

> Ключ заключается в поиске правильного баланса с учетом приоритетов бизнеса, ожиданий пользователей и ограничений системы. Это требует четкого обсуждения приемлемых компромиссов, количественных критериев успеха и регулярной переоценки по мере изменения требований к системе.

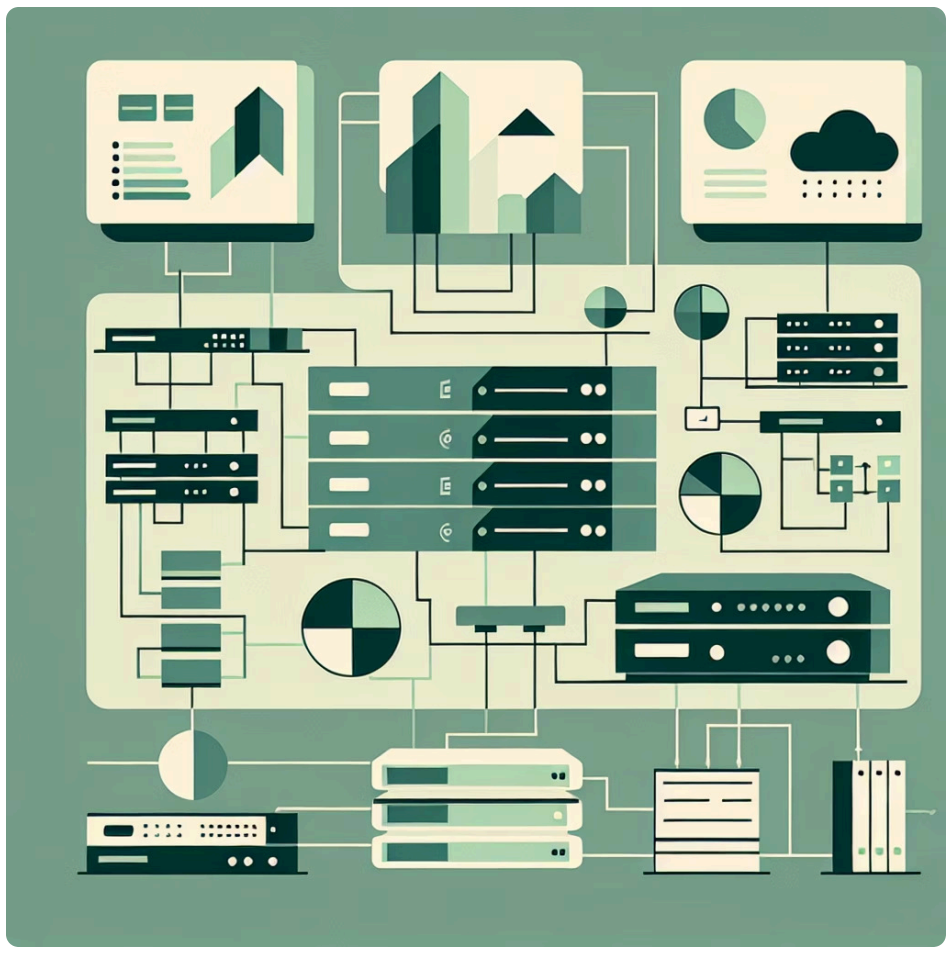
Ограничения инфраструктуры



Аппаратные ограничения

Мощность процессора, объем памяти и производительность хранилища напрямую влияют на подход к оптимизации производительности.

- Агрессивные стратегии кэширования
- Оптимизация использования ресурсов
- Выбор эффективных алгоритмов



Ограничения сети имеют глубокие архитектурные последствия. Ограниченная пропускная способность требует тщательного рассмотрения схем передачи данных — возможно, вам потребуется реализовать сжатие, оптимизировать размеры полезных нагрузок или изменить пакетные операции.

Нормативные ограничения

1

GDPR Соответствие

Требует создания возможностей удаления данных во всей системе — не просто пометки записей как удаленных, а фактического удаления персональных данных из резервных копий, журналов и производных наборов данных.

2

HIPAA Требования

В системах здравоохранения предписывают наличие определенных функций аутентификации и контроля. Это архитектурные основания, влияющие на обработку доступа к данным и ведение журналов.

3

SOX Регулирование

Финансовые нормативные требования требуют контроля изменений и разделения доступа, что влияет на процессы развертывания и границы системы.

Организационные ограничения

Временные ограничения

Давление заказчика вынуждает идти на технические компромиссы

Бюджетные ограничения

Влияют на выбор инфраструктуры и технологий

Навыки команды

Ограничивают выбор технологических решений

Структура команды

Определяет архитектуру через закон Конвея

Функциональные команды

Организованы по техническим специальностям (фронт-энд, бэк-энд, база данных). Создают многоуровневые архитектуры с четкими технологическими границами.

- Глубокие технические знания
- Замедление межфункциональных функций

Продуктовые команды

Организованы по бизнес-возможностям. Создают сервис-ориентированные архитектуры с полной ответственностью за функциональность.

- Быстрое внедрение функций
- Дублирование работы между командами

Работа с ограничениями

Принятие реальности

Наиболее успешные архитекторы работают в рамках ограничений, а не борются с ними, находя креативные решения.

Творческий подход

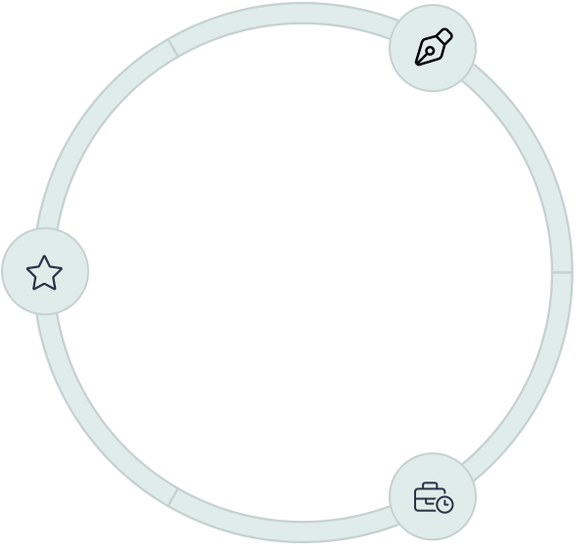
Ограничения стимулируют инновации и заставляют искать нестандартные решения, которые могут оказаться более эффективными.

Баланс целей

Достижение целей в области качества при уважении реальных ограничений, которые определяют каждую систему.

Важно помнить: Ограничения не являются препятствиями для хорошей архитектуры — они являются контекстом, в котором создается великая архитектура. Лучшие решения часто рождаются именно из необходимости работать в рамках жестких ограничений.

Заключение: Системный подход к качеству



Атрибуты качества

Определяют эффективность системы

Ограничения

Формируют границы решений

Бизнес-контекст

Определяет приемлемые компромиссы

Качественные характеристики и ограничения составляют основу всех архитектурных решений. Ключ к успешной архитектуре заключается в понимании того, что эти элементы работают вместе как система.

> Вместо поиска идеальных решений, эффективные архитекторы сосредотачиваются на принятии четких, осознанных решений, которые соответствуют приоритетам бизнеса и одновременно создают условия для будущего развития систем.

Помните, что как атрибуты качества, так и ограничения со временем меняются. Наиболее успешные архитекторы предвидят эти изменения и проектируют системы, которые могут адаптировать приоритеты атрибутов качества и творчески работать в условиях меняющихся ограничений по мере развития бизнеса.