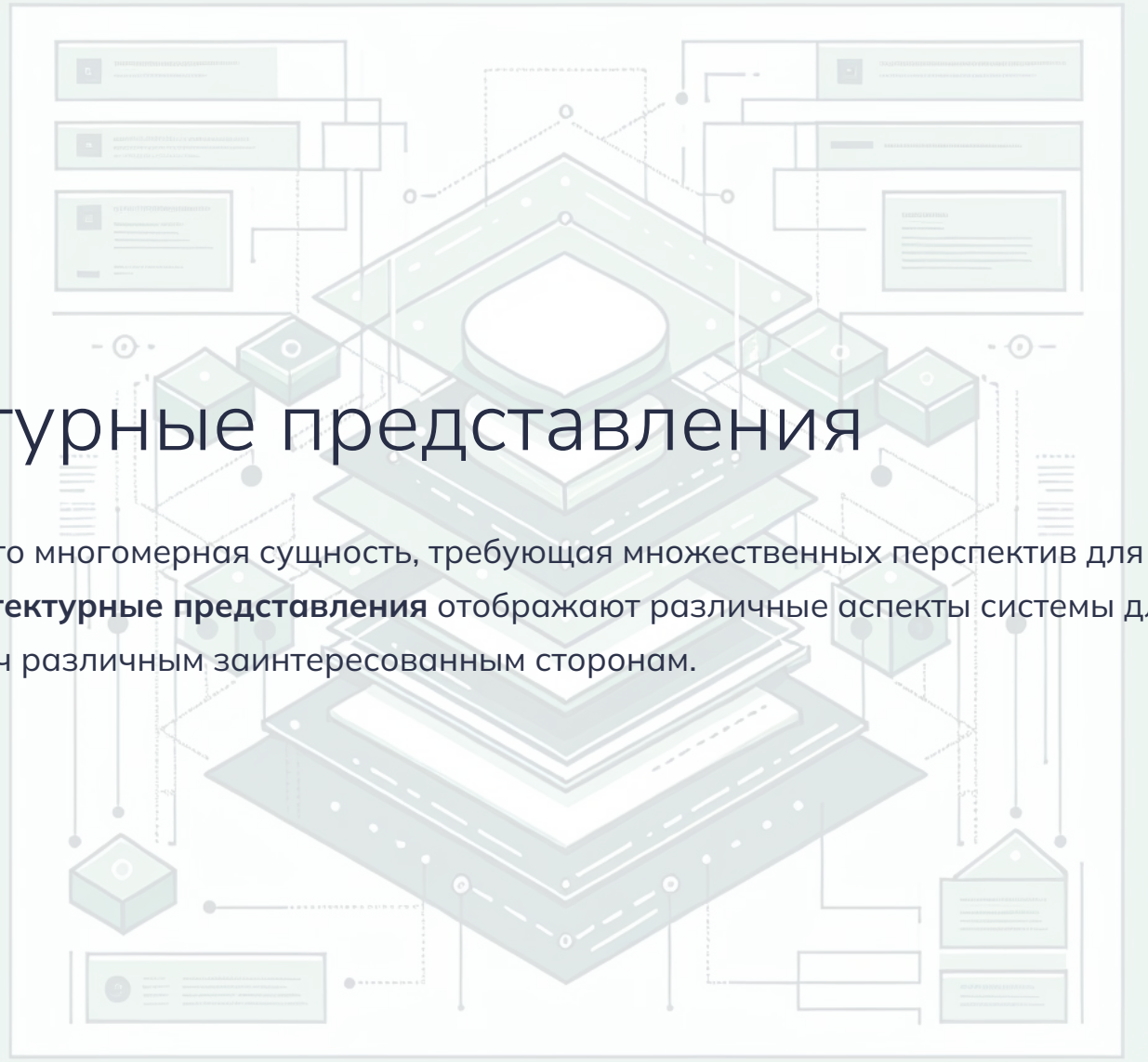


Архитектурные представления

Архитектура ПО — это многомерная сущность, требующая множественных перспектив для понимания и документации. **Архитектурные представления** отображают различные аспекты системы для передачи специфических задач различным заинтересованным сторонам.

Модель архитектурных представлений 4+1

Можно попробовать показывать разные аспекты ПО в одной диаграмме, но скорее всего получится запутанно и перегруженно.

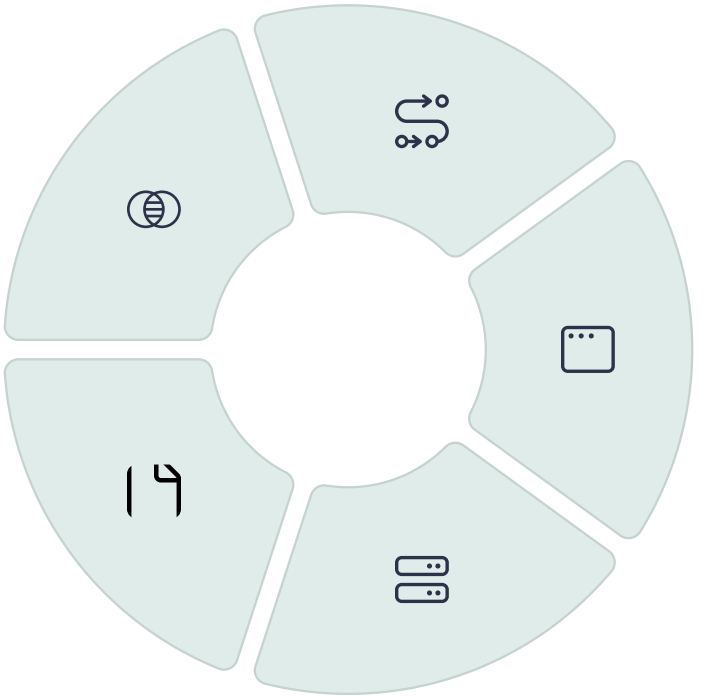
Модель архитектурных представлений 4+1, разработанная Филиппом Крухтеном, решает эту проблему. Она предоставляет четыре фундаментальных представления плюс сценарии для комплексного документирования архитектуры ПО.

Логическое представление

Функциональные требования и услуги пользователям

Сценарии

Объединяющие случаи использования



Представление процессов

Runtime-архитектура и взаимодействия

Представление разработки

Статическая организация модулей ПО

Физическое представление

Отображение ПО на аппаратное обеспечение

Логическое представление

Логическое представление описывает функциональные требования системы и то, как система предоставляет услуги конечным пользователям. Оно фокусируется на функциональности системы, показывая ключевые абстракции и объектные модели.

Доменные объекты

Основные бизнес-сущности и их отношения в системе

Функциональная декомпозиция

Разбиение системы на логические компоненты

Поведенческие паттерны

Ключевые алгоритмы и бизнес-правила

Это представление демонстрирует, что делает система и как различные функциональные элементы соотносятся друг с другом, используя диаграммы классов, диаграммы объектов и конечные автоматы. Далее мы будем разбирать эту область гораздо подробнее, когда погрузимся в DDD.

Пример на собеседовании: Можно задизайнить систему e-commerce исходя из логического деления: например, предложить создать Order Service, Payment Service и Catalog Service, каждый из которых будет отвечать за соответвующий функционал.

Представление процессов

Представление процессов показывает runtime-архитектуру системы, включая процессы, потоки и их взаимодействия. Как система работает в развернутом состоянии. Это представление затрагивает параллелелизм, распределение нагрузки и характеристики производительности.

Runtime-элементы

- Процессы и сервисы
- Базы данных
- Очереди сообщений
- Механизмы коммуникации

Ключевые паттерны

- Клиент-сервер взаимодействия
- Публикация-подписка
- Конвейерная обработка
- Распределенные системы

В отличие от логического представления, это показывает, что на самом деле происходит при выполнении системы, подчеркивая конкурентность, распределение и отказоустойчивость.

Пример на собеседовании: API Gateway маршрутизирует запросы в 10 реплик Order Service, который синхронно вызывает Inventory Service через REST, затем публикует событие OrderCreated в Kafka для асинхронной обработки Payment Service и Notification Service.

Представление разработки

Представление разработки описывает статическую организацию модулей ПО, библиотек, подсистем и инструментов разработки. Это представление затрагивает задачи управления ПО и показывает, как система организована с точки зрения разработчика.

01

Структура кода

Организация пакетов, модулей и библиотек в кодовой базе

02

Зависимости сборки

Управление зависимостями между компонентами системы

03

Организация команды

Распределение ответственности между командами разработки

04

Архитектурные ограничения

Обеспечение соблюдения границ модулей и принципов

Представление разработки более конкретно, чем традиционные модульные представления — оно показывает реальную организацию кода, а не абстрактную функциональную декомпозицию.

Пример на собеседовании: Монорепозиторий с модульной структурой `/services/order`, `/services/payment`, библиоткой `@company/auth-lib`, слоями (domain, application, infrastructure), где команда Checkout владеет Order и Payment модулями.

Физическое представление

Физическое представление описывает отображение ПО на аппаратное обеспечение и показывает физическое развертывание системы. Это представление показывает, как программные компоненты распределяются по инфраструктуре.

Топология развертывания

Физическое размещение серверов, сетевого оборудования и других компонентов инфраструктуры

Сетевые соединения

Конфигурация сетевых связей между различными узлами системы

Распределение ресурсов

Планирование мощностей и оптимизация использования аппаратных ресурсов

Это представление поддерживает планирование мощностей, оптимизацию производительности, планирование восстановления после сбоев и управление инфраструктурой.

Пример на собеседовании: Kubernetes в AWS EKS с автомасштабированием (3-20 подов), PostgreSQL RDS Multi-AZ с read репликами, Redis ElastiCache, CloudFront CDN для фронтэнда, всё распределено по трём availability zones с помощью Application Load Balancer.

Представления качества

Помимо структурных представлений, представления качества рассматривают специфические задачи заинтересованных сторон, фокусируясь на конкретных атрибутах качества или нефункциональных требованиях.

Что такое представления качества?

Архитектурные перспективы, которые подчеркивают специфические качества системы, такие как безопасность, производительность, надежность или масштабируемость. В отличие от структурных представлений, которые показывают "что" и "где", представления качества показывают "насколько хорошо".

Пересекающие аспекты

Представления качества накладывают дополнительные перспективы, которые пересекают множественные структурные элементы, обеспечивая целостный взгляд на качественные характеристики системы.

Рассмотрим это на примере двух аспектов: безопасности и производительности

Представление безопасности

Представление безопасности фокусируется на механизмах защиты и анализе поверхности атак. Показывает границы безопасности, зоны доверия, потоки аутентификации/авторизации и механизмы контроля доступа.

Файрволлы

Защита периметра сети

SSL/TLS терминация

Шифрование трафика

Аутентификация

OAuth, JWT токены

Управление секретами

Безопасное хранение ключей

Шифрование данных

Защита хранимой информации

Это представление позволяет командам безопасности идентифицировать уязвимости, планировать стратегии защиты и обеспечивать соответствие требованиям безопасности.

Представление производительности

Представление производительности подчеркивает отзывчивость системы, пропускную способность и характеристики масштабируемости. Показывает критичные для производительности компоненты и стратегии оптимизации.

Ключевые компоненты производительности

- Сеть доставки контента (CDN) для глобальной доставки
- Слои кэширования (кластеры Redis)
- Автомасштабируемые уровни приложений
- Оптимизированные конфигурации баз данных
- Реплики для чтения

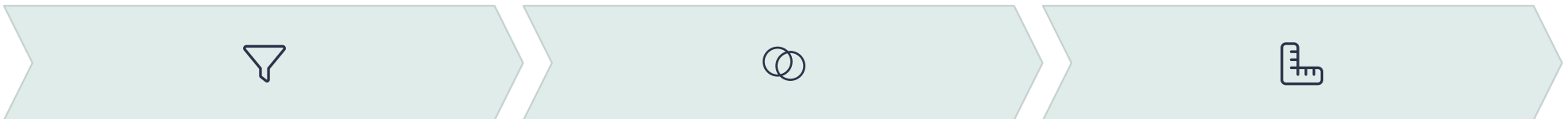
Метрики и мониторинг

- Время отклика
- Пропускная способность
- Использование ресурсов
- Идентификация узких мест

Это представление позволяет инженерам по производительности понимать поведение системы под нагрузкой и планировать улучшения мощности.

Комбинирование представлений

Мы начали статью с утверждения о том, что попытка показать все аспекты ПО на одной диаграмме ничего хорошего не принесет, и это правда. Но на практике есть смысл наложить и отобразить вместе несколько представлений, если у этого есть правильная аудитория или цель.



Отдельные представления

Каждое представление фокусируется на специфических аспектах архитектуры

Наложения

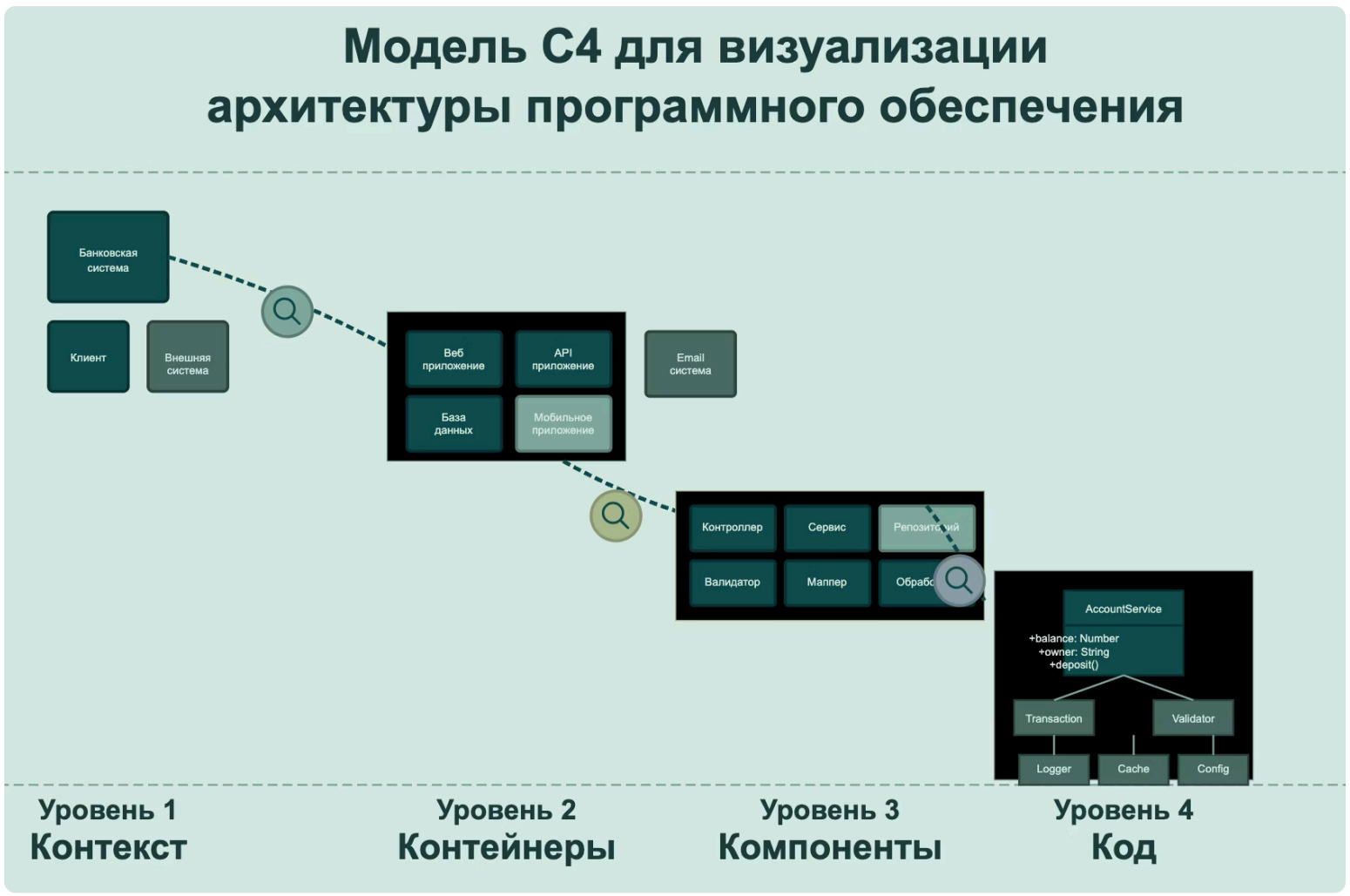
Комбинирование информации из двух представлений с сохранением оригинальных типов

Комбинированные представления

Эффективное отображение тесных ассоциаций между элементами

Важно: Избегайте перегрузки комбинированных представлений слишком большим количеством отображений. Это работает лучше всего, когда представления имеют тесные отношения и сильные ассоциации между элементами.

Модель C4: Современная архитектурная визуализация



Модель C4 представляет собой дополнительный метод документирования, который предоставляет стандартизированный, иерархический подход к визуализации архитектуры ПО на различных уровнях абстракции.

Системный контекст

Показывает программную систему и то, как она вписывается в мир с точки зрения пользователей и внешних систем. Отвечает на вопрос "Что это за система и как она вписывается в мир?" Фокусируется на людях и программных системах, а не на технических деталях

Диаграмма контейнеров

Увеличивает зум и показывает контейнеры. Показывает общую форму архитектуры и технологические решения. Контейнер представляет приложение или хранилище данных.

Диаграмма компонентов

Увеличивает отдельный контейнер, чтобы показать компоненты внутри него. Идентифицирует основные структурные строительные блоки и их взаимодействия в рамках конкретного контейнера.

Диаграмма кода

Увеличивает отдельный компонент, показывая элементы кода. Предоставляет детали реализации на уровне интерфейсов и классов, часто генерируется автоматически.

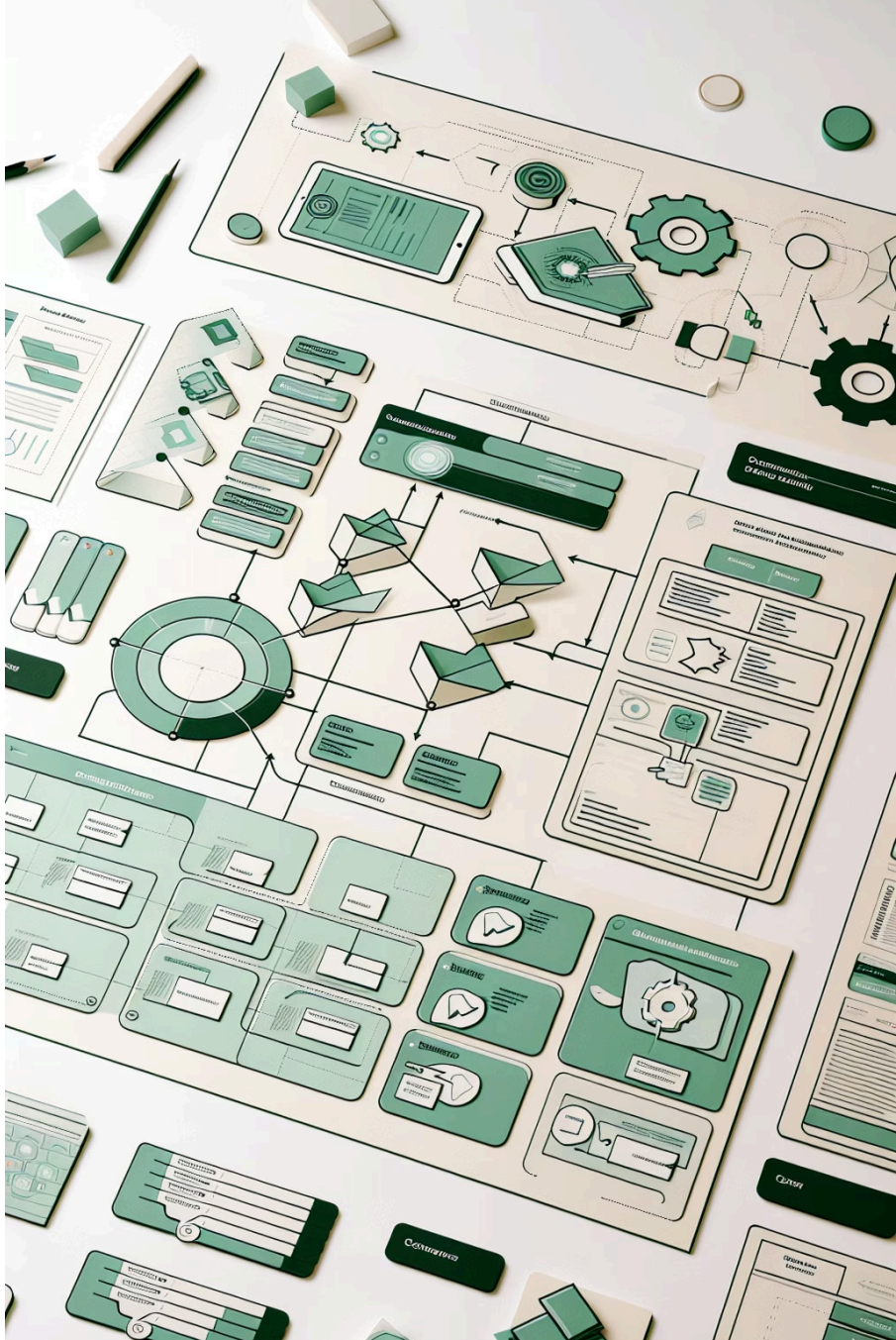
Она соединяет разрыв между высокоуровневым системным контекстом и детальной реализацией, предлагая практический framework для архитектурной документации.

Интеграция C4 с DDD и Event Storming

Интеграция C4 с Domain-Driven Design (DDD) работает особенно эффективно, создавая мощный подход к архитектурному моделированию.

C4 + DDD интеграция

- Системный контекст отображается на границы доменов
- Диаграммы контейнеров показывают bounded contexts
- Диаграммы компонентов раскрывают доменные агрегаты
- Сервисы в рамках каждого контекста



Enterprise Architecture Frameworks

Несколько enterprise architecture фреймворков предлагают комплексные, стандартизированные системы представлений из коробки, устраняя ad-hoc архитектурную документацию.

ArchiMate

Наиболее комплексный - структурированный язык моделирования с предопределенными слоями (Business, Application, Technology) и 25+ стандартных перспектив

TOGAF ADM

Architecture Development Method со стандартными архитектурными доменами: Business, Data, Application и Technology Architecture

Gartner Framework

Практический подход к корпоративной архитектуре, фокусирующийся на бизнес-результатах и связи между ИТ и бизнес-стратегией

ArchiMate предлагает наиболее зрелое решение "из коробки" для организаций, требующих комплексного архитектурного моделирования с минимальными накладными расходами на настройку.

Заключение

Архитектурные представления — это инструменты коммуникации, которые обеспечивают эффективное сотрудничество в организациях разработки. Модель 4+1, диаграммы C4 и методы документирования поведения создают архитектурную документацию, которая служит заинтересованным сторонам.

Потребности заинтересованных сторон

Эффективная архитектурная документация начинается с понимания потребностей различных заинтересованных сторон

Подходящие представления и нотации

Выбор правильных инструментов и методов документирования для каждой конкретной задачи

Живая документация

Поддержание документации как живого артефакта, который развивается вместе с системой

> Архитектурные представления превращают сложные технические системы в понятные, коммуницируемые структуры, обеспечивая успешную разработку и эволюцию программного обеспечения.