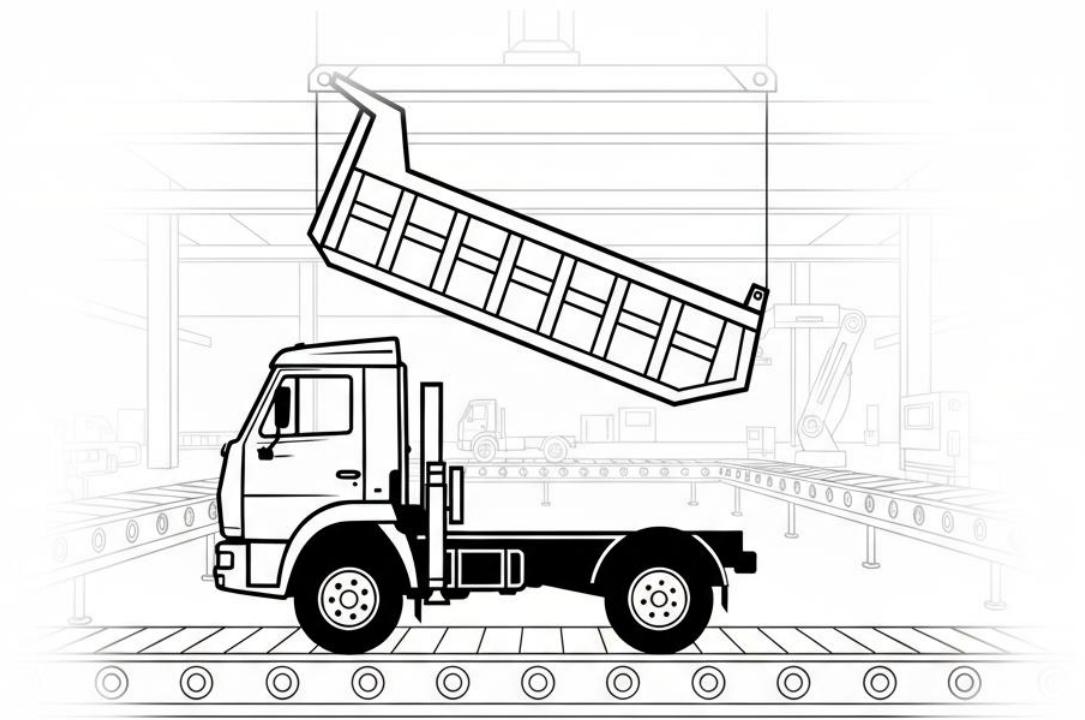
Путь к платформе
для коробочных продуктов

infotecs

ARCHDAYS
2025

Ценность
Вызовы
Уроки



Минко В.

О себе

Виталий Минко

Главный архитектор

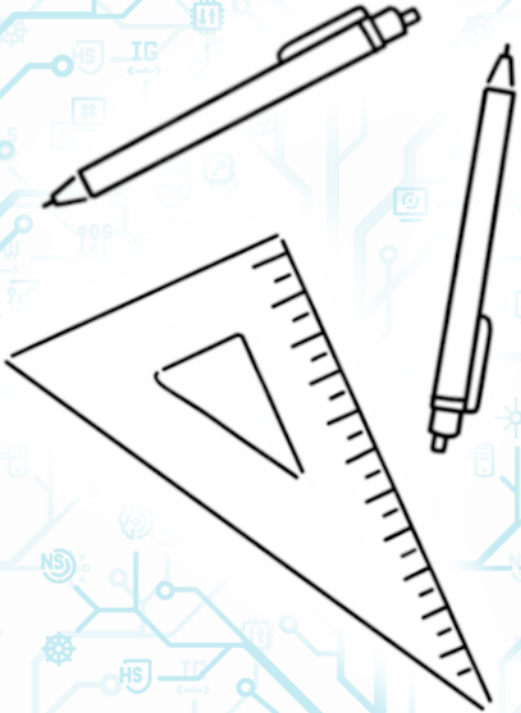
Центра разработки программных продуктов

infotecs

SA SAFe® 6

The Open Group Open Certified

TOGAF® 9 Certified

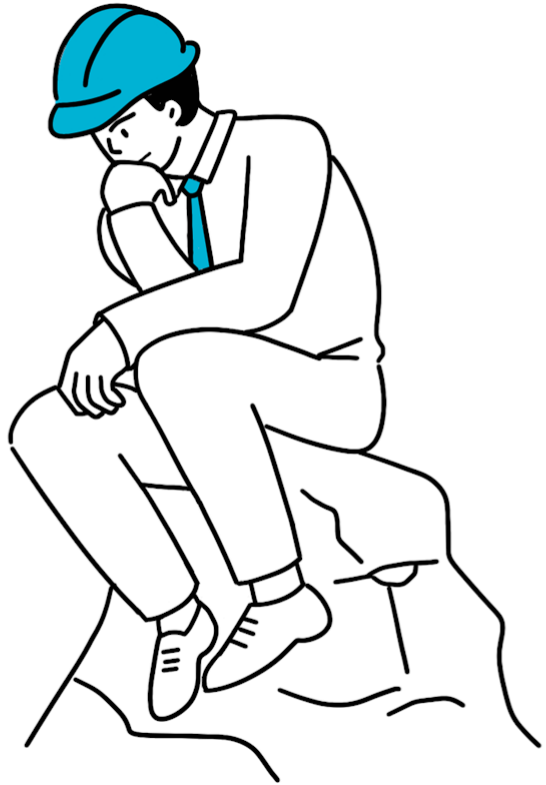


1. Проблематика
2. Что такое платформа?
3. Команда и организация
4. Реализация платформы
5. Заключение и выводы

Проблематика

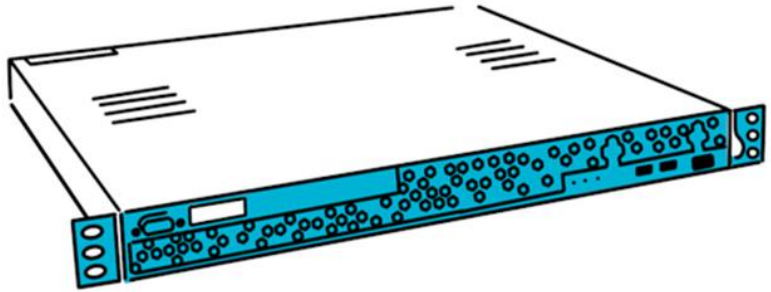
Проблема повышения эффективности

Как **повысить эффективность** разработки
и сопровождения продуктов,
снизить издержки?



Специфика infotecs

Специфика infotecs



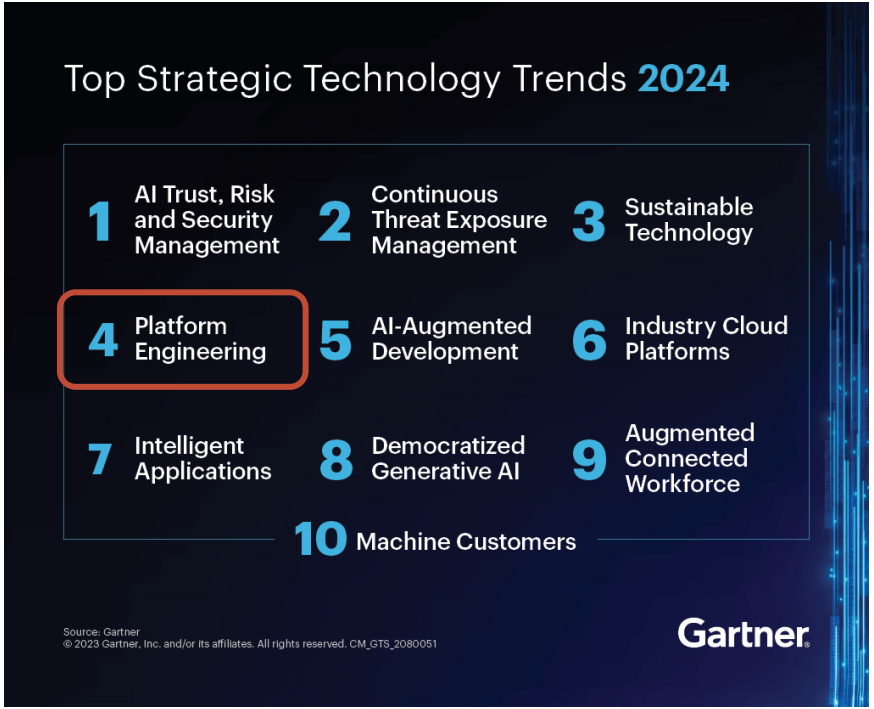
- Близкая технологическая база
- Широкий спектр функциональных требований
- Высокие нефункциональные требования
- Общие требования регуляторов

Проблематика

- Большие накладные расходы на адаптацию реализованной функции к смежным продуктам
- Высокая когнитивная нагрузка у продуктовых команд:
  - Погружение в проект – от 2-х месяцев
  - Количество технологий, которые должна освоить команда для разработки – 50

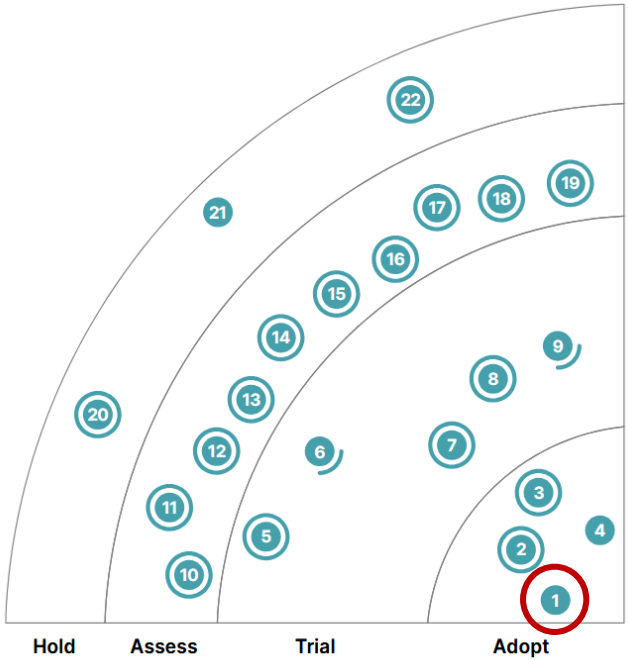
Platform engineering в трендах

- Топ-10 технологических трендов **Gartner** на 2024 г.



Platform engineering в трендах

- Топ-10 технологических трендов **Gartner** на 2024 г.
- Технологический радар **Thoughtworks** на апрель 2023 г.



Применение практик управления продуктом к внутренним платформам

Platform engineering в трендах

- Топ-10 технологических трендов **Gartner** на 2024 г.
- Технологический радар **Thoughtworks** на апрель 2023 г.
- **Forrester** - Содействие высококачественной разработке встроенного программного обеспечения, 2024 г.

> Инженерные платформы помогают командам встроенного ПО создавать **качественные** и **безопасные** продукты быстрее и с **меньшими затратами**, сочетая **эффективность** с соблюдением строгих стандартов отрасли. Более половины специалистов считают, что платформенная инженерия повышает качество разработки без снижения продуктивности.

Задача и ограничения

Перевести разработку продуктов на платформу:

- Темпы поставки бизнес-ценности снижать нельзя
- Существенные инвестиции для реализации стратегии не предусмотрены

Что такое платформа?

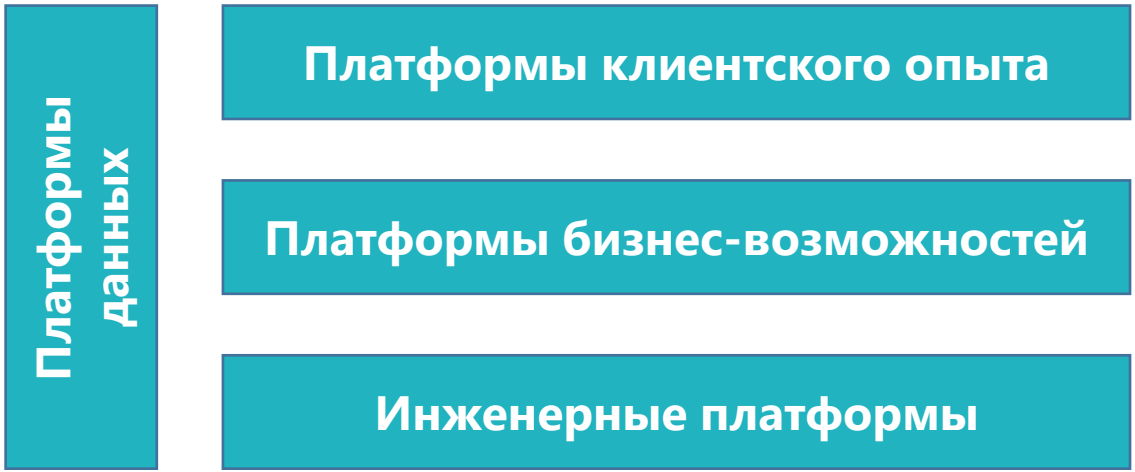
> **Внутренние платформы разработки** — это системы, создаваемые IT-отделами для переиспользования общих IT-сервисов, повышения производительности разработки и соблюдения операционных стандартов.

Gregor Hohpe, Platform Strategy

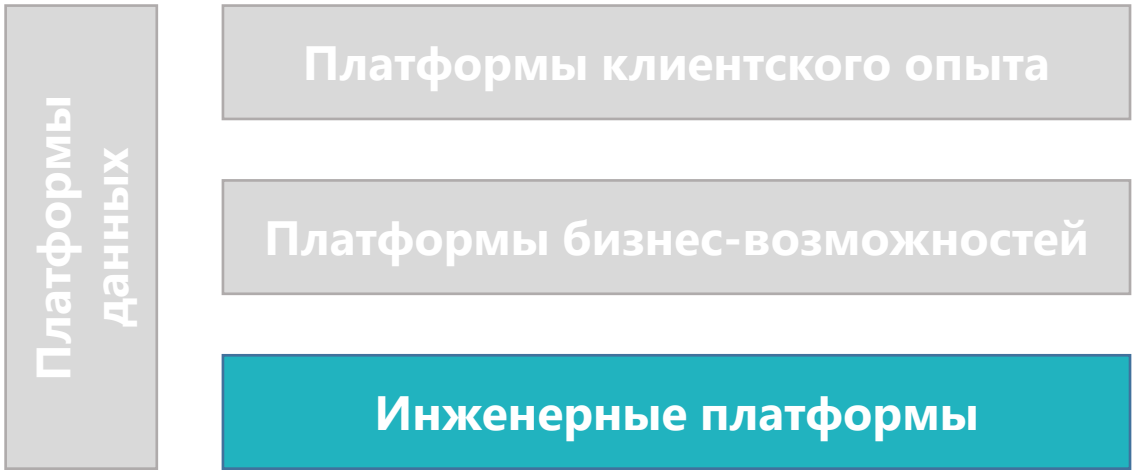
> **Цифровая платформа** — это основа из self-service API, сервисов, знаний и поддержки, которые объединены в привлекательный внутренний продукт.
> Автономные команды разработки могут использовать платформу для более быстрого выпуска новых функций с меньшими затратами на координацию.

https://martinfowler.com/articles/talk-about-platforms.html

Типы внутренних платформ



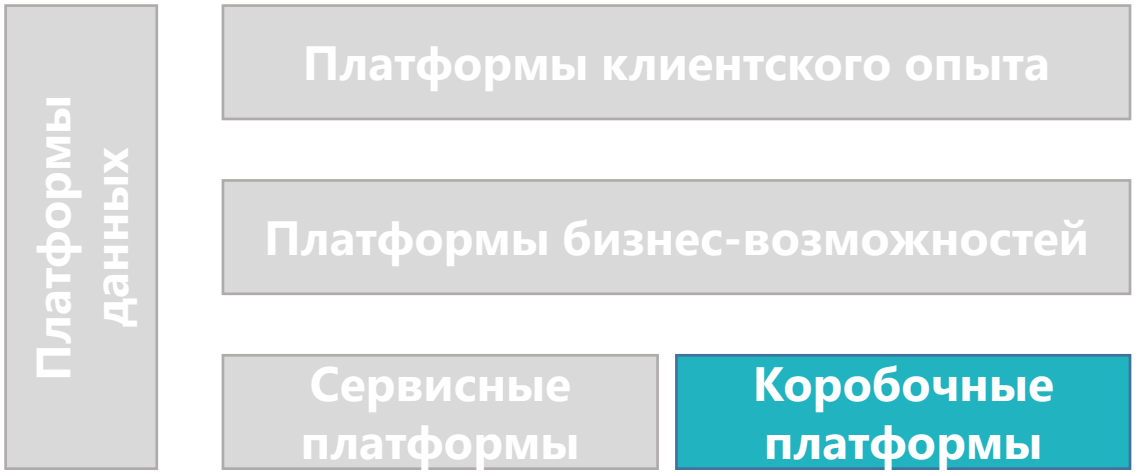
Типы внутренних платформ



Ключевые свойства
внутренних инженерных платформ

- Платформа не даёт непосредственную ценность бизнесу
- Платформа пригодна для самостоятельного использования продуктовыми командами
- Платформа позволяет создавать непредвиденные решения через продуманные точки расширения
- Платформа обладает свойством системности

Типы внутренних платформ



Специфика встраиваемых платформ ПАК

- Жёсткая связь с аппаратным обеспечением
- Наличие слоя абстракции от аппаратного обеспечения (HAL)
- Строгие требования к безопасности и сертификации
- Обновления ПО и поддержка без локального доступа к устройству
- Большая вариативность в стеке – от ядра ОС до Web UI

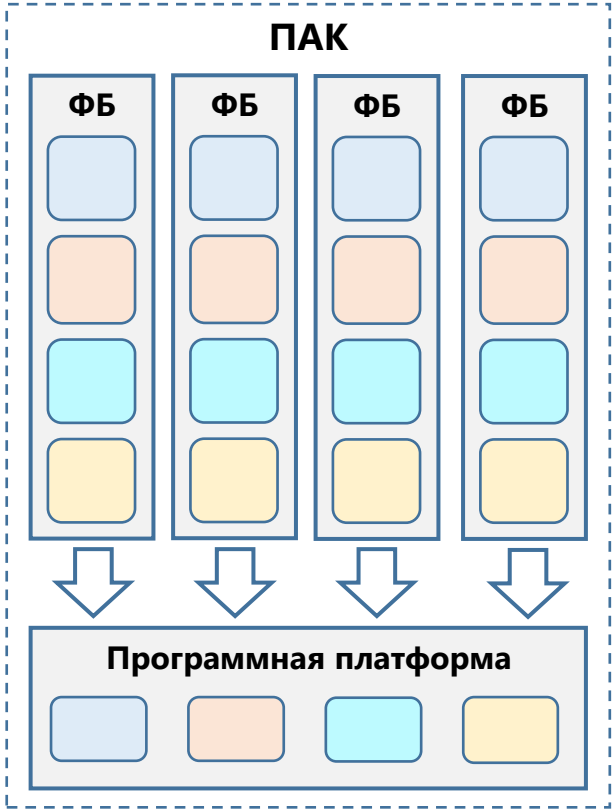
Концептуальная архитектура ПАК на основе платформы

Программная платформа (ПП) — программная основа, предоставляемая продуктовым командам для самостоятельного создания однотипных продуктов под конкретные бизнес-потребности.

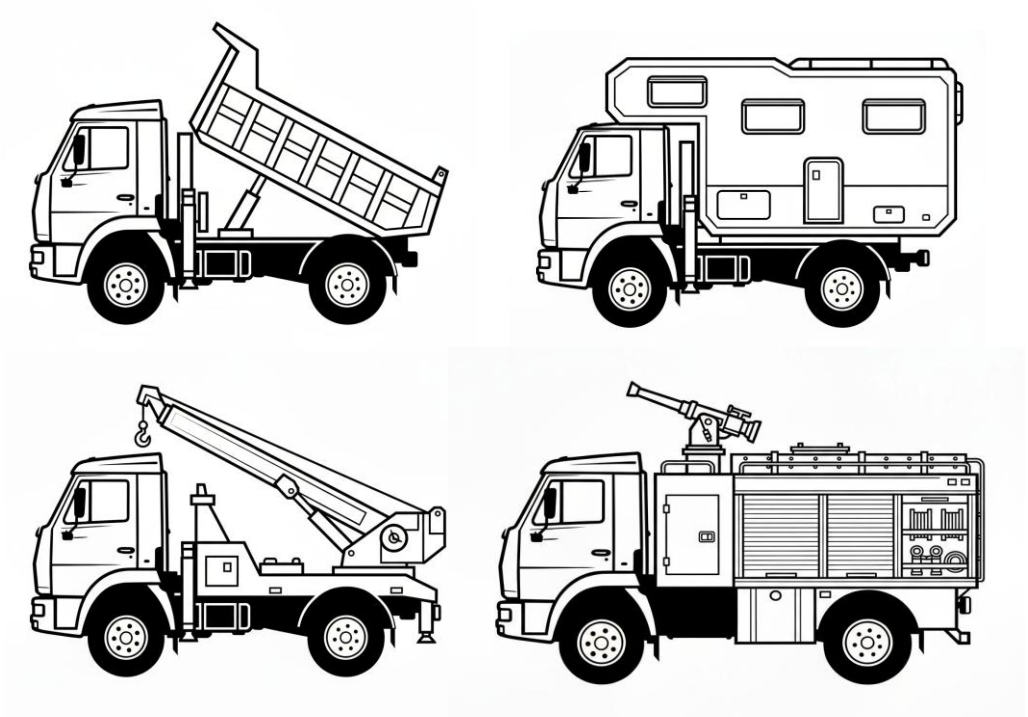
Реализует Обеспечивающие Функции продукта, обеспечивает возможность унификации схожих обеспечивающих функций и имеет модульную структуру.

Модуль ПП - Программная единица, часть Программной платформы, реализующая одну обеспечивающую функцию продукта.

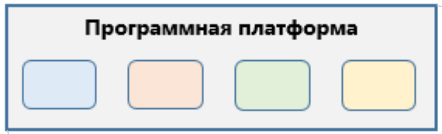
Функциональный блок (ФБ) — функционально законченный набор программ, библиотек и прочих файлов, реализующий одну целевую функцию продукта.

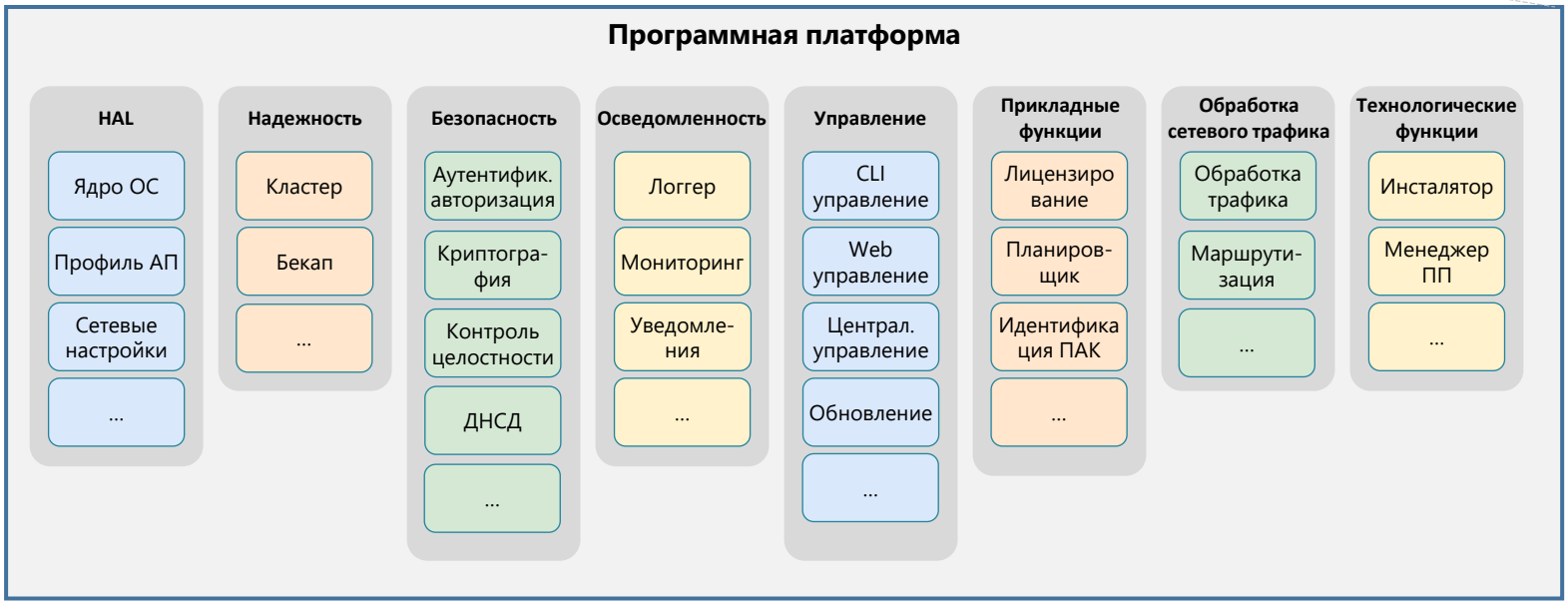


Платформа



Варианты продуктов на основе платформы (платформа + функциональный блок)


Программная платформа


Программная платформа
HAL
Ядро ОС
Профиль АП
Сетевые настройки
...
Надежность
Кластер
Бекап
...
Безопасность
Аутентифик. авторизация
Криптогра-фия
Контроль целостности
ДНСД
...
Осведомленность
Логгер
Мониторинг
Уведомле-ния
...
Управление
CLI управление
Web управление
Централ. управление
Обновление
...
Прикладные функции
Лицензиро вание
Планиров-щик
Идентифика ция ПАК
...
Обработка сетевого трафика
Обработка трафика
Маршрути-зация
...
Технологические функции
Инсталятор
Менеджер ПП
...

Пример: CLI


ПАК
ФБ IPS
ФБ МЭ
ФБ VPN
...
Команды настройки IPS
Команды настройки МЭ
Команды настройки VPN
Программная платформа
Парсер команд

Преимущества и недостатки

Бизнес

Ускорение выпуска новых версий продуктов
за счет переиспользования готовой основы

Сокращение сроков разработки интеграционных и корпоративных решений
за счет реализации решения в единой точке

Упрощение поставки новых функций в продукты
за счет перено
продуктов

Упрощение по сертификации
за счет единоо
реализации фу
документации к ним

Снижение скорости внесения частных изменений

Конкуренция работ при планировании

Отдельная реализация функции не подходит продукту

Увеличение сроков обработки по issue/bug

Снижение издержек
на доработки и тиражирование функций, устранение дефектов

Упрощение сопровождения и снижение рисков
за счет повышения качества архитектурных решений

Повышение экспертизы сотрудников
за счет специализации в конкретных предметных областях

Снижение когнитивной нагрузки сотрудников
за счет внедрения API платформы

Понимание недостатков

infotecs

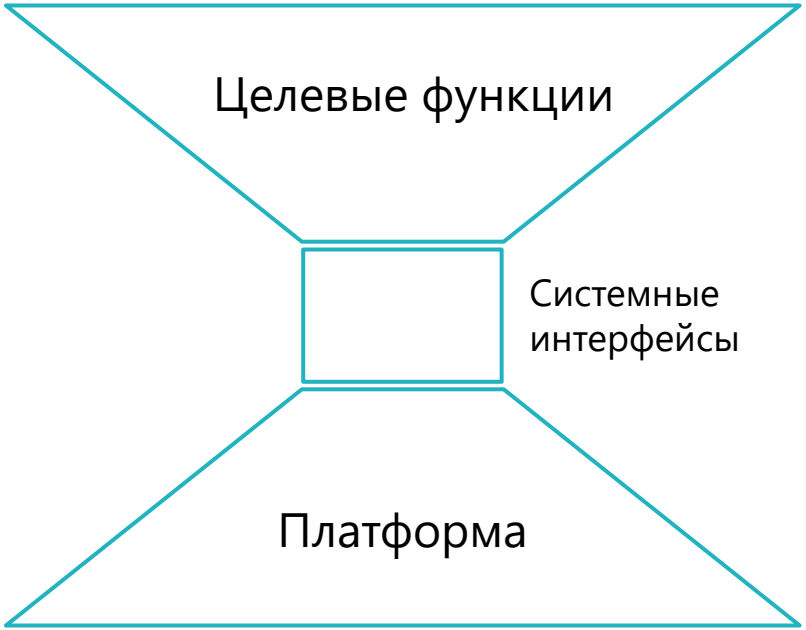
Снижение скорости внесения частных изменений

Конкуренция работ при планировании

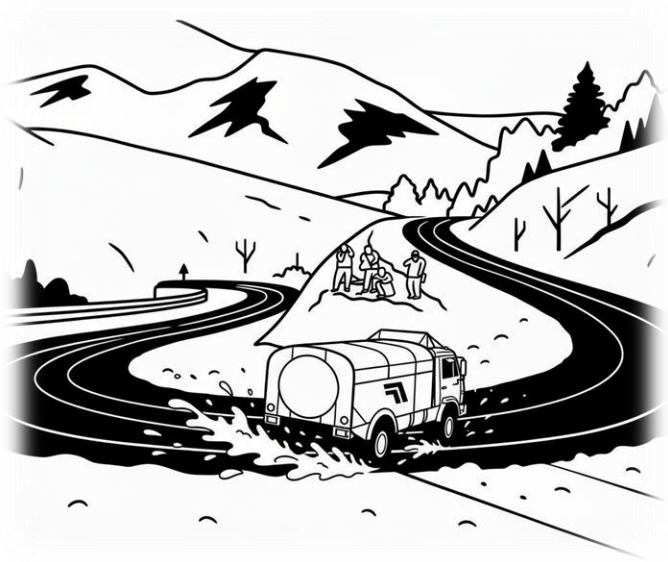
Отдельная реализация функции не подходит продукту

Увеличение сроков обработки по issue/bug

Снижение скорости внесения частных изменений



Снижение скорости внесения частных изменений



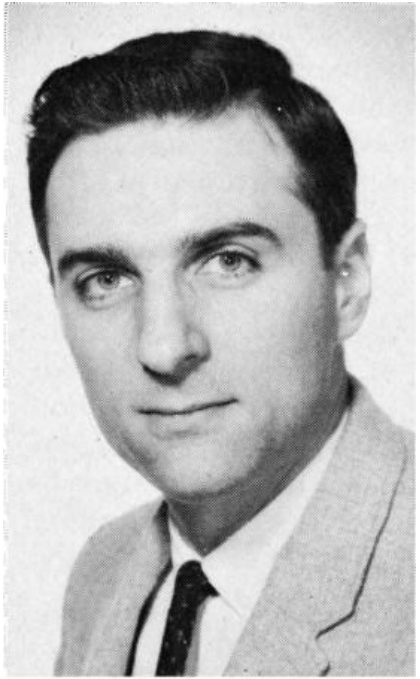
Платформа снижает гибкость для непредвиденных изменений

Платформа ускоряет движение по выбранному направлению

Команда и организация

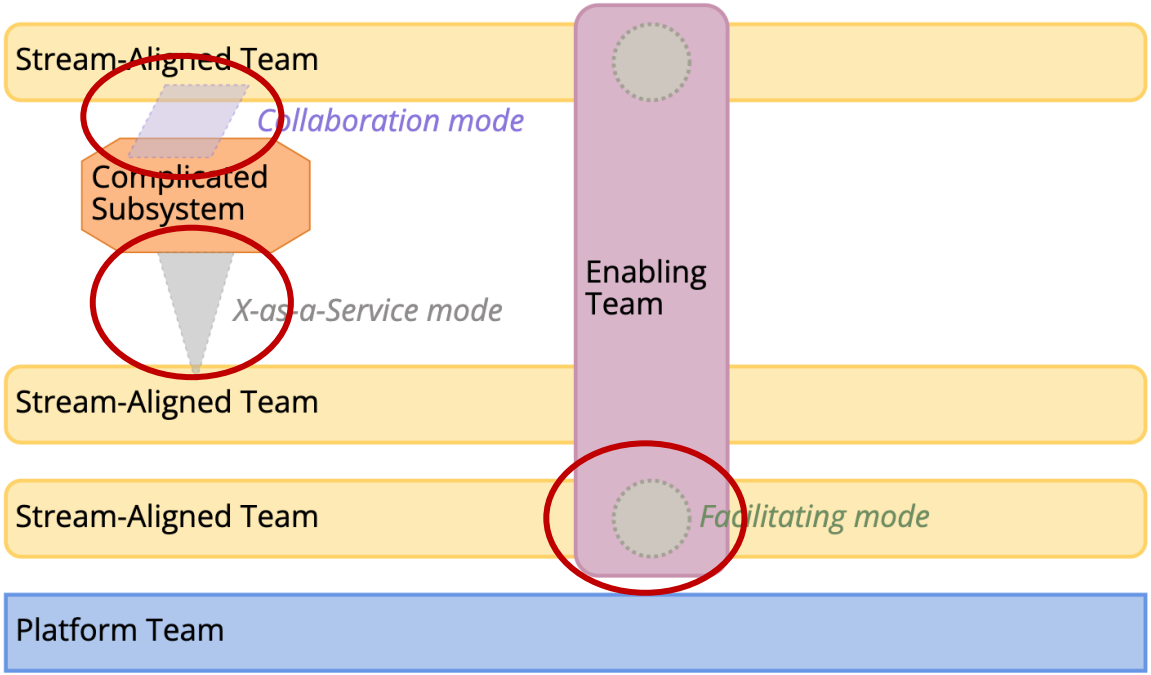
Закон Конвея

> Организация, разрабатывающая систему, неизбежно создаёт дизайн, который копирует структуру коммуникаций этой организации.



Мелвин Конвей

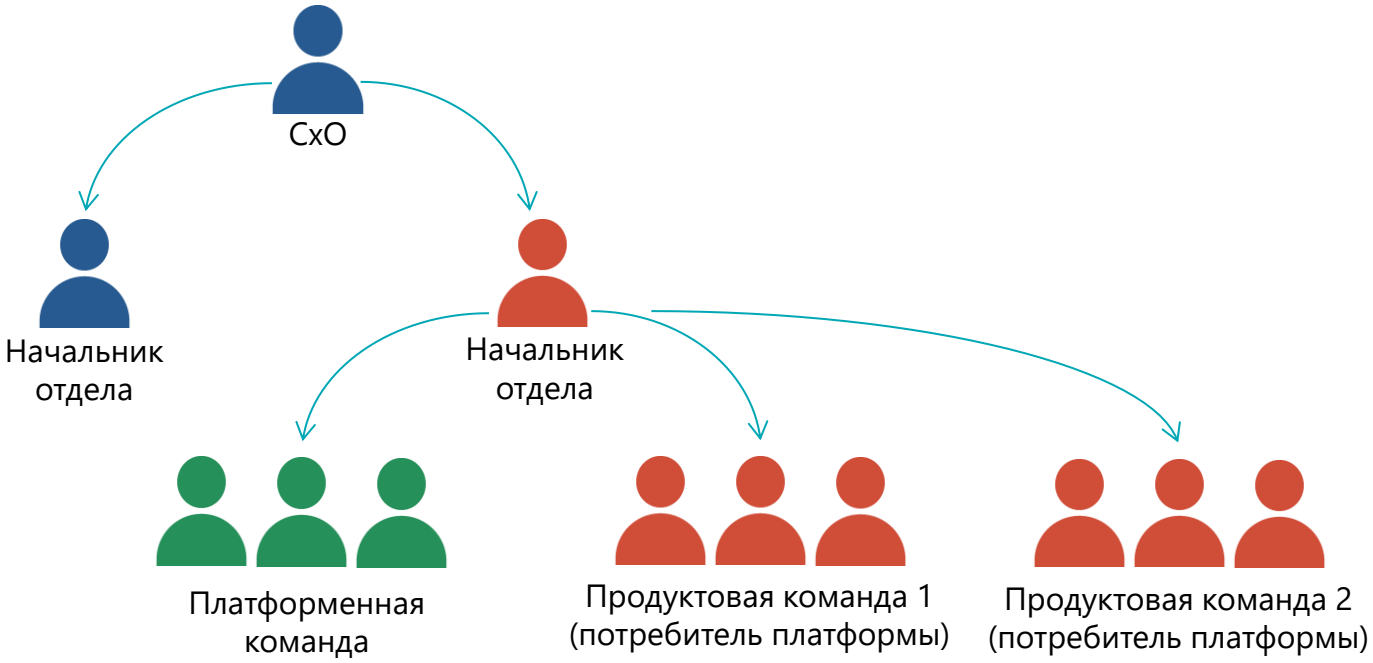
Платформенные команды в Team Topologies



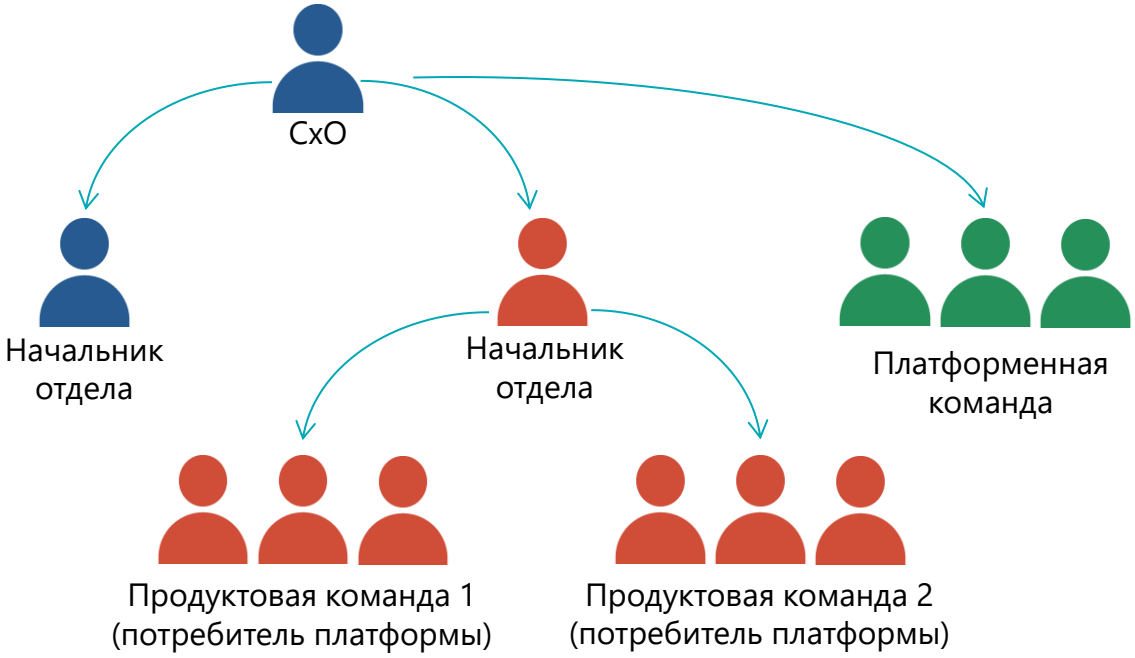
Выделенная команда разработки платформы

Ключевое – команда платформы должна быть обособленной от продуктовых команд

Контур управления
для платформенной команды



Контур управления
для платформенной команды



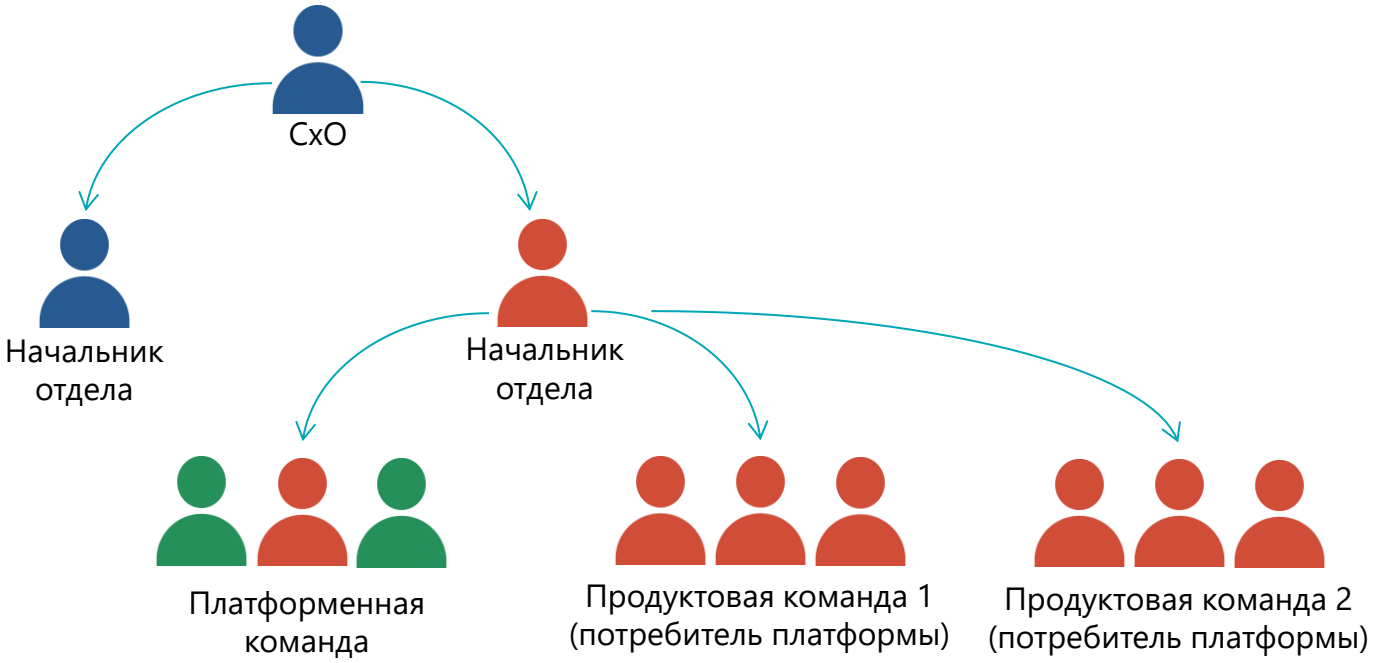
Антипаттерн
Build It and They Will Come

Платформа создаётся без реального запроса или вовлечения продуктовых команд, на основе предположения, что если её построить — команды сами начнут её использовать.

Ресурсы для команды платформы

- Не подходит продуктовая команда
  (у неё фокус на поставку бизнес-ценности)
- Не подходит сторонняя команда
  (недостаточно понимание потребностей и глубины знания предмета)
- Решение – комбинированный состав, где костяк составляют бывшие представители продуктовых команд, но усиленные дополнительными разработчиками.

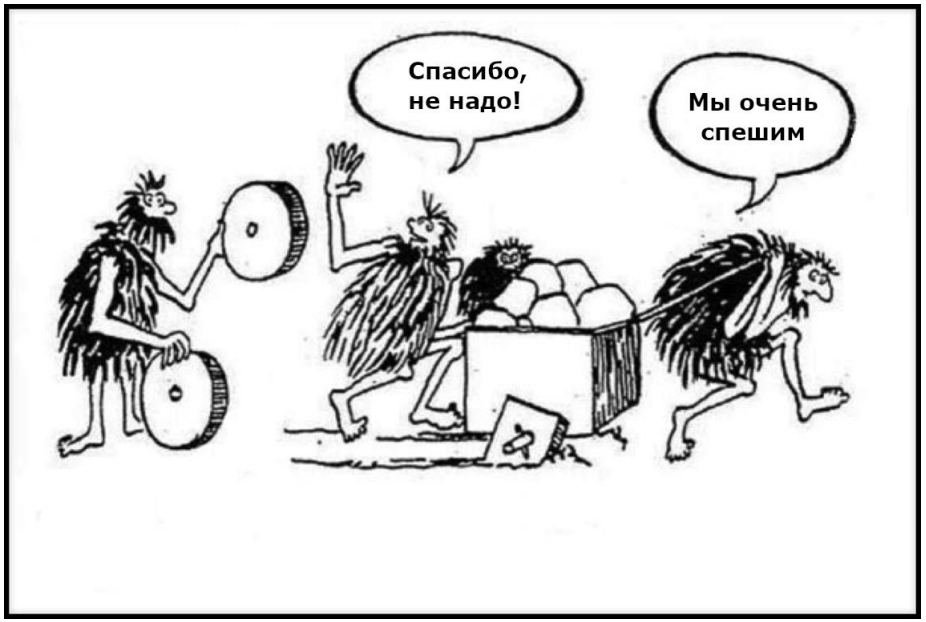
Контур управления
для платформенной команды



Ресурсы на реализацию платформенной стратегии

Вариант решения - выстроить итеративный процесс с окупаемостью на каждом шаге.

Ресурсы на реализацию платформенной стратегии



Практический пример:
миграция логирования в платформу

- Возникла бизнес-потребность реализовать ряд новых функций логирования
- Выделен модуль платформы для логирования
- Обеспечено обособленное развитие и контроль качества
- Реализованы новые функции в выделенном модуле платформы
- Функции поставлены сразу во все продукты на платформе

Реализация платформы

Как определить границы?

Два направления унификации:

1. **Распространения платформы (потребителей)**
2. Функционального состава платформы

Определение границ распространения

- Платформа для всех – платформа ни для кого:
  - Сильно затягивается разработка
  - Повышается сложность реализации
- Не брать сразу широко. Начать с инициативной группы – найти группу наиболее заинтересованных.

  Остальных потенциальных потребителей держать обозревателями.

Как определить границы?

Определение границ унификации:

1. Распространения платформы (потребителей)
2. **Функционального состава платформы**

Унификация функций ПАК

Результат анализа обеспечивающих функций

28 функций в 13 продуктах:

Определение функциональных границ

Точно
нецелесообразно
унифицировать

Точно
целесообразно
унифицировать

Определение функциональных границ

НА кластер
Журнал пакетов

Загрузчик
Ядро
Система инициализации

Функций посредине выбираем исходя из:

- Технологической связанности
- Новых бизнес-требований
- Допустимого уровня значения метрик

Определение функциональных границ

Бизнес

Разработка

- Снижение скорости внесения частных изменений
- Конкуренция работ при планировании
- Отдельная реализация функции не подходит продукту
- Увеличение сроков обработки по issue/bug

Бизнес-метрики

Название метрики DORA	Каноническое значение	Адаптация под платформенный подход	Ожидание от внедрения платформы
Deployment Frequency (Частота выкатов)	Как часто команда готова и успешно выкатывает изменения.	Количество поставляемых бизнесу функций в единицу времени.	Увеличится
Lead Time for Changes (Время от идеи до продакшна)	Измеряет, как быстро от простой правки доходит новая версия компонента/продукта до потребителя.	Время поставки отдельных изменений от взятия бизнес-запроса в работу до релиза.	Уменьшится до допустимого уровня
Change Failure Rate (Процент неудачных изменений)	Процент изменений, приведших к инцидентам, багам или откатам.	Количество дефектов/уязвимостей уровня Crit и High на релиз коробочного продукта.	Уменьшится
Mean Time to Restore (MTTR, Среднее время восстановления)	Время от обнаружения дефекта до полного восстановления работоспособности	Среднее время исправления дефектов/уязвимости в релизе. Среднее время исправления дефекта при стабилизации продукта.	Увеличится до допустимого уровня

Метрики разработки

- Время переноса крупной функции
- Время интеграции изменения
  (из компонента в платформу, из платформы в продукт)

DevEx-метрики:

- Количество команд и продуктов, использующих платформу
- Оценка качества и полноты документации
- Процент пользователей платформы, прошедших онбординг без прямой помощи платформенной команды
- Время подключения нового разработчика/команды до рабочего состояния на платформе

Контроль метрик

- Deployment Frequency (частота выкатов) – увеличилось на 30%
- Среднее время исправления дефекта при стабилизации – не увеличилось
- Время переноса крупной функции – снизилось до 1 недели
- Время интеграции изменений – увеличилось с 1 до 2-х недель
- Количество технологий, которые команда должна освоить для разработки – за год снизилось на 30%
- Количество продуктов, использующих платформу – за год увеличилось на 3
- Время погружения в проект – уменьшилось с 2 мес. до 1.5 мес.

Заключение и выводы

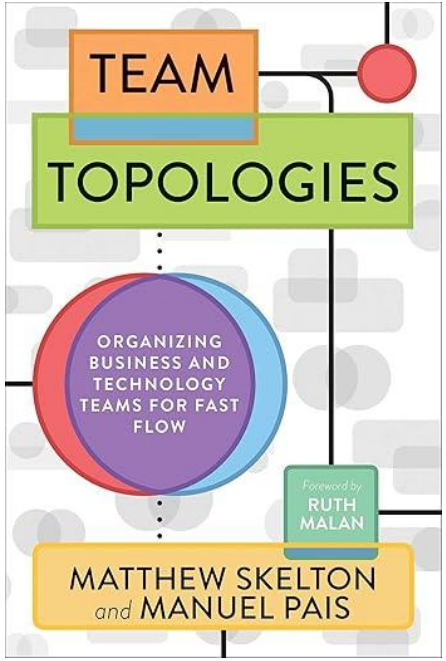
Распространённые тезисы
не подтверждённые практикой

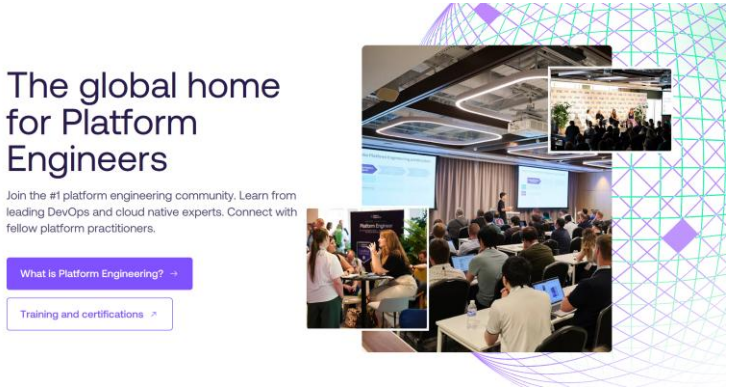
- Платформы всегда снижают time-to-market
- Платформенный подход всегда требует больших инвестиций
- Платформенную команду всегда стоит выносить из контура управления продуктовой разработки

Ключевые факторы успеха

1. Понимать цели и ограничения платформенного подхода.
2. Наличие обособленной платформенной команды, глубоко знающей предмет и потребности потребителей.
3. Итеративный процесс освоения подхода через непременную поставку бизнес-ценности на каждом шаге.
4. Непрерывный контроль ключевых показателей.

Дополнительные ресурсы





https://platformengineering.org/

ARCHDAYS 2025

infotecs

Ответы на вопросы

Подписывайтесь на наши соцсети

vk.com/infotecs_news

https://t.me/infotecs_official

rutube.ru/channel/24686363

ARCHDAYS 2025

infotecs

Спасибо за внимание!

Виталий Минко

главный архитектор

vitaly.minko@infotecs.ru +7 (915) 386-48-95

Подписывайтесь на наши соцсети

vk.com/infotecs_news

https://t.me/infotecs_official

rutube.ru/channel/24686363