СБЕР

Синхронизация производства

Скорость, надежность и простота артерии DevOps

Владимир Мединский
SberWorks

Чем будет полезен доклад?

Поговорим о сложности тривиальных задач в кровавом Enterprise

Расскажем про архитектуру решений, highload в DevOps и собственный опыт

Объясним ценности служебных сервисов DevOps и важности их качества

Коротко об окружении

1

Периметр сетей Банка поделен на изолированные сегменты

2

Сегменты поделены на сетевые зоны

3

Сетевые зоны принадлежат соответствующему этапу разработки ПО

4

В каждом этапе есть инструмент хранения, будь то код, или артефакт

Какую проблему решаем

1

Из-за изоляции и требований к сетевым сегментам системы хранения артефактов и кода дублируются

2

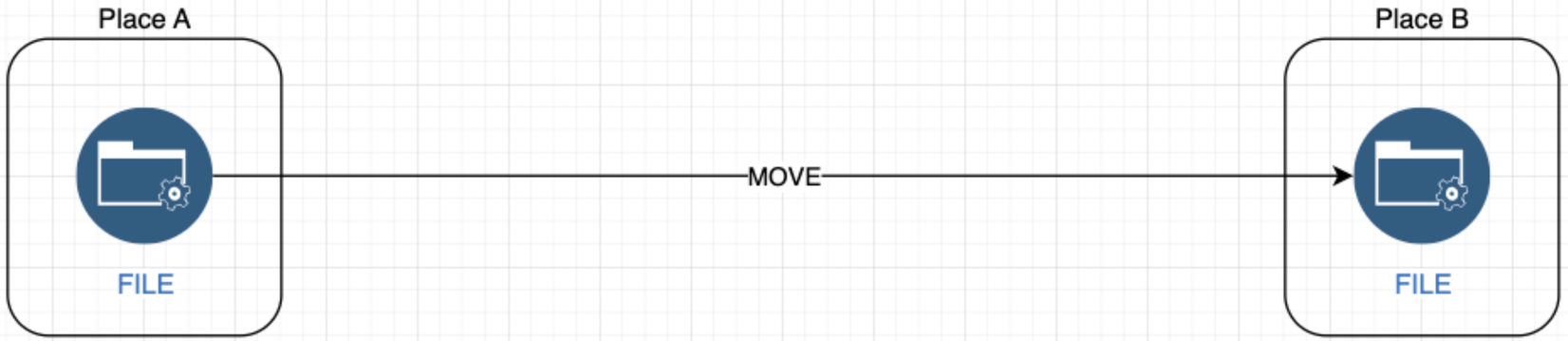
Среда разработки изолирована от среды эксплуатации

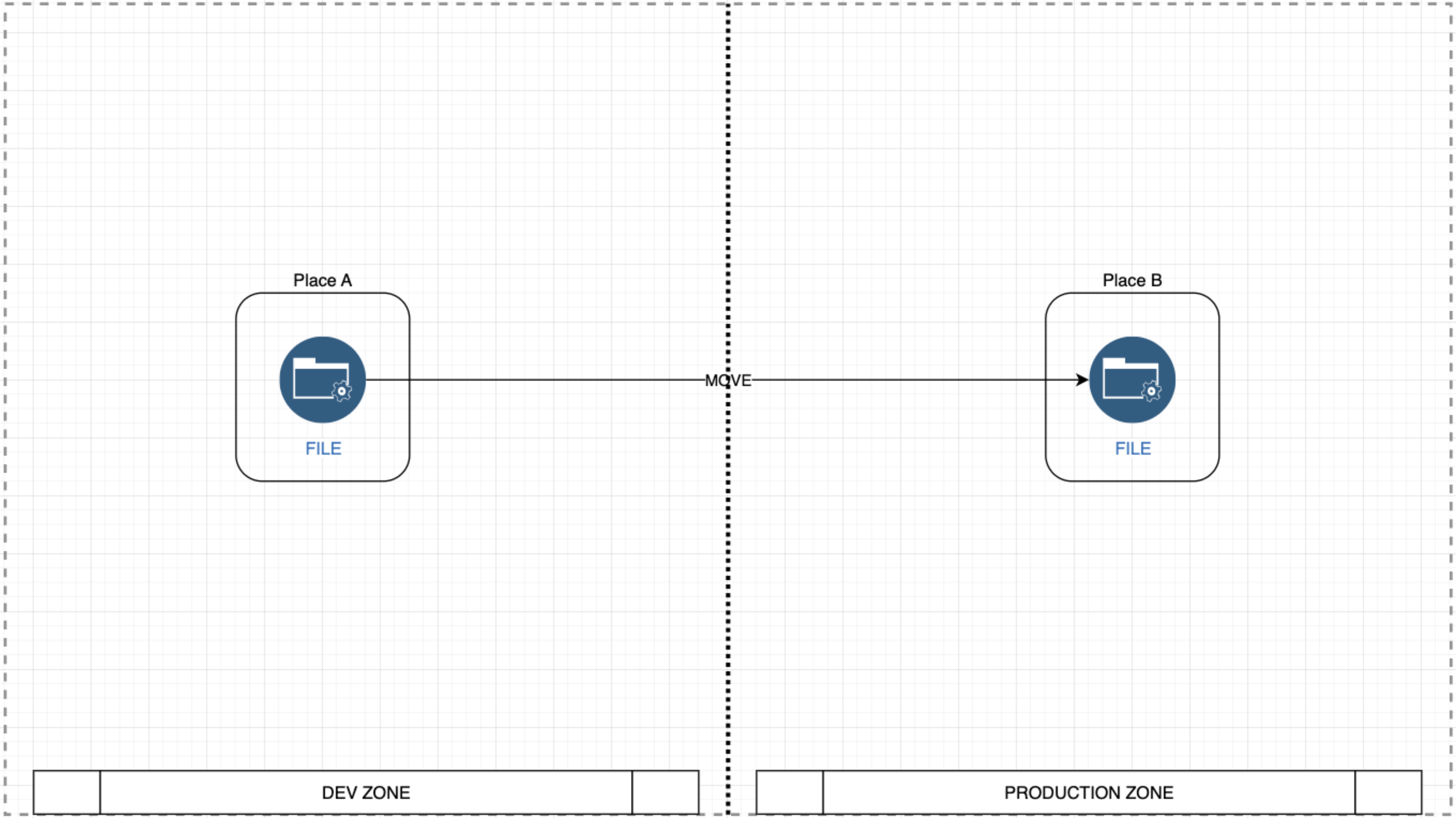
3

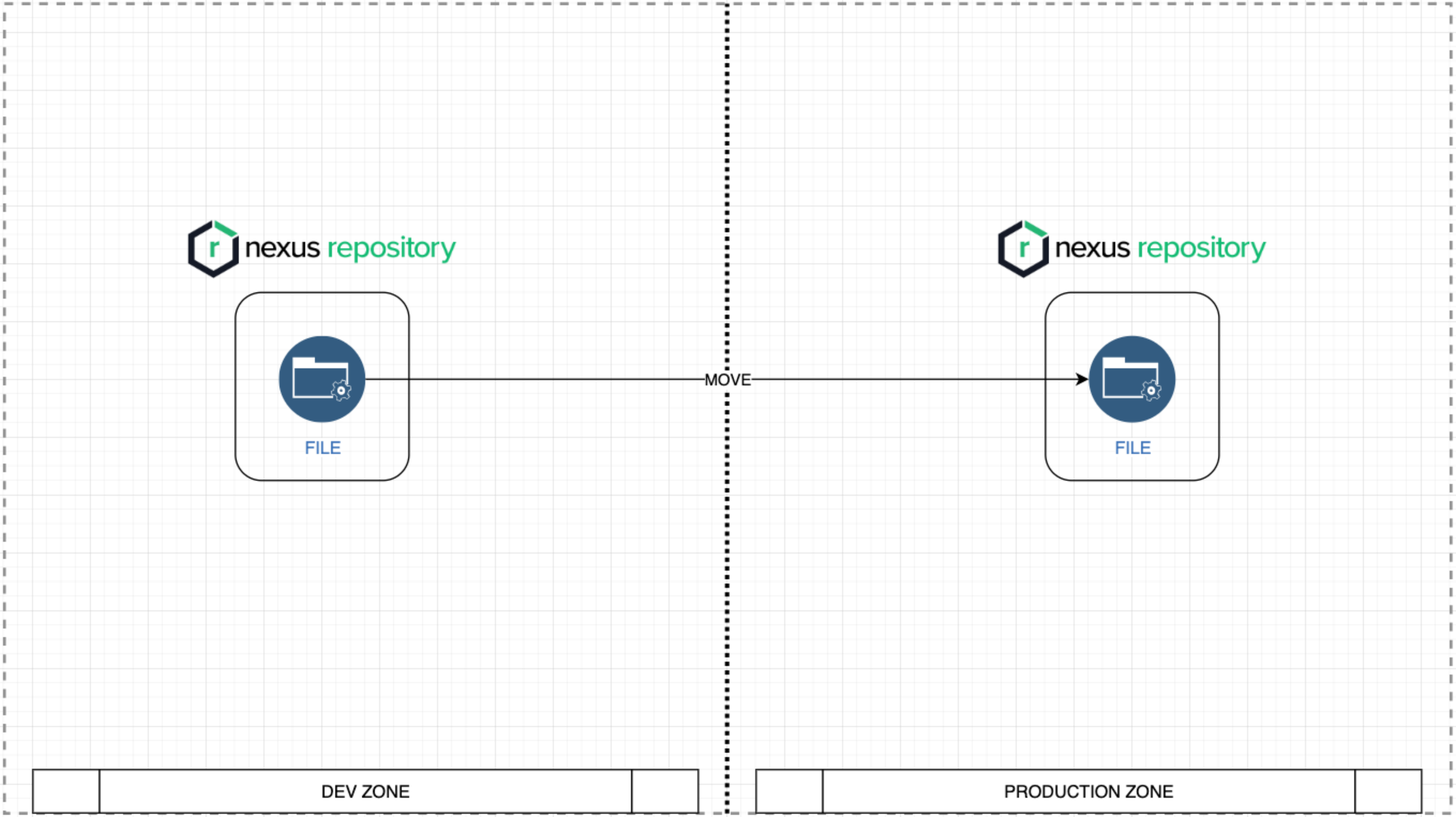
Цикл производства – Последовательная цепочка нескольких систем хранения (Nexus, Bitbucket)

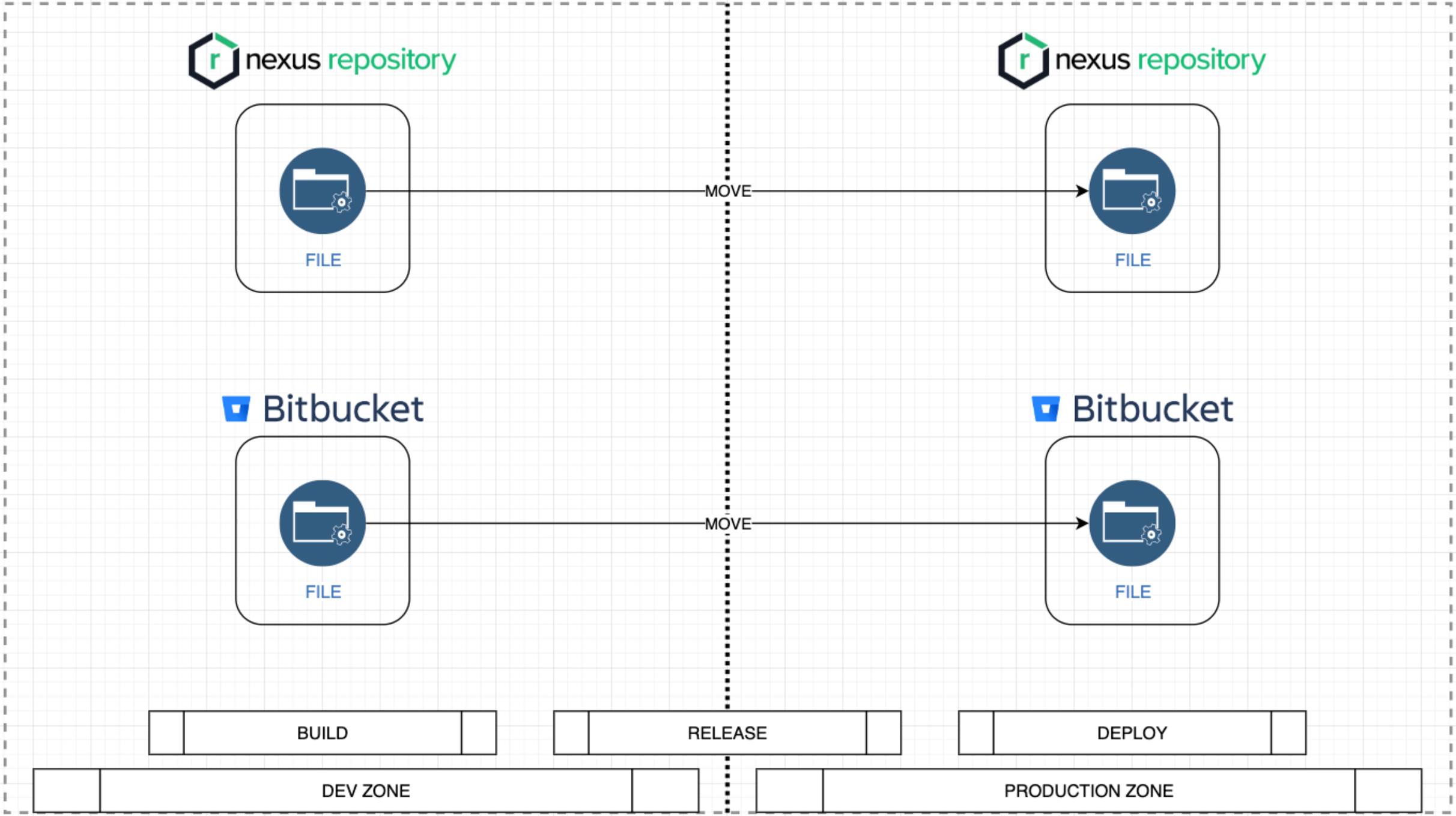
4

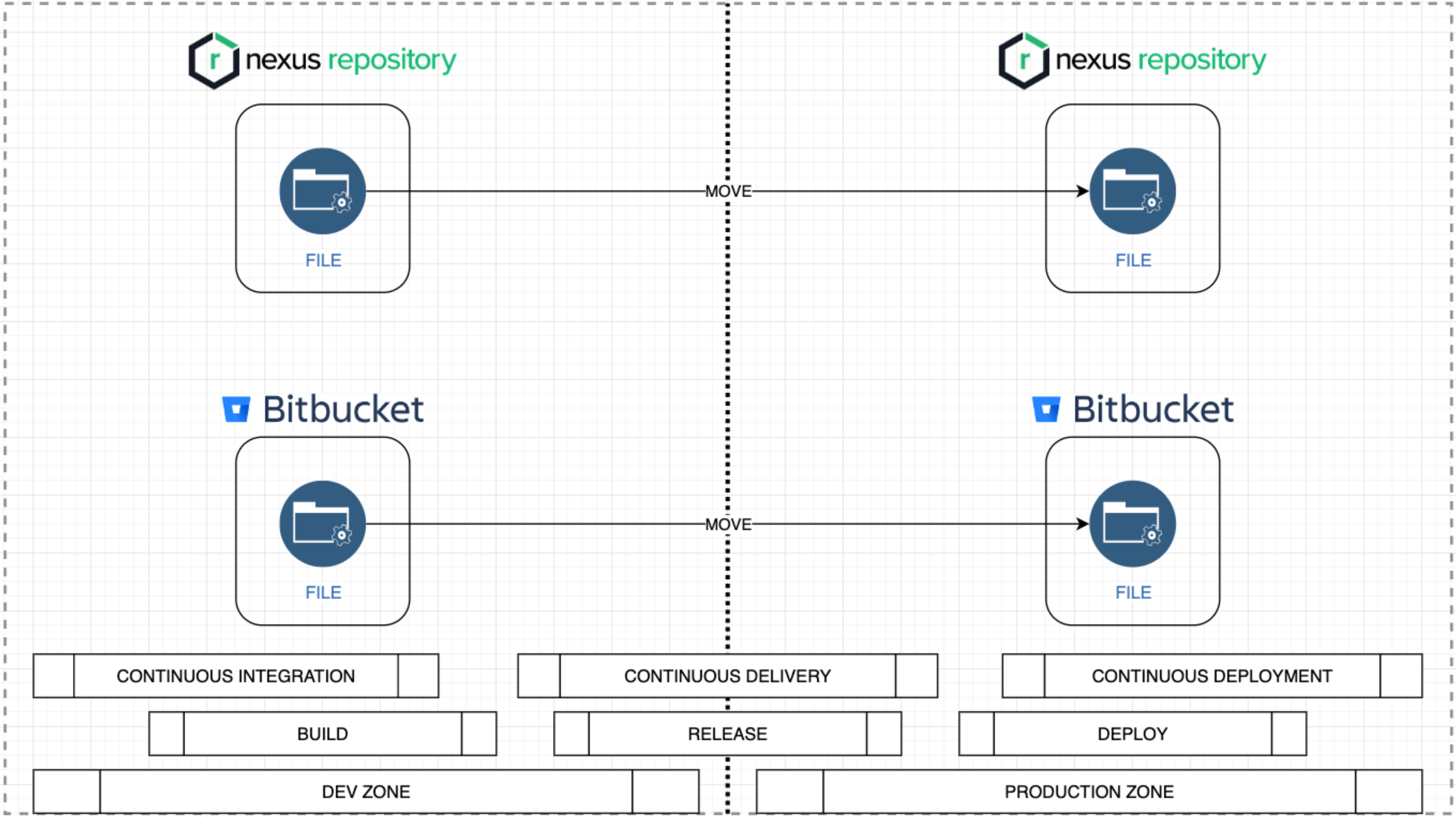
Отсутствует автоматическая последовательная передача артефактов от этапа к этапу

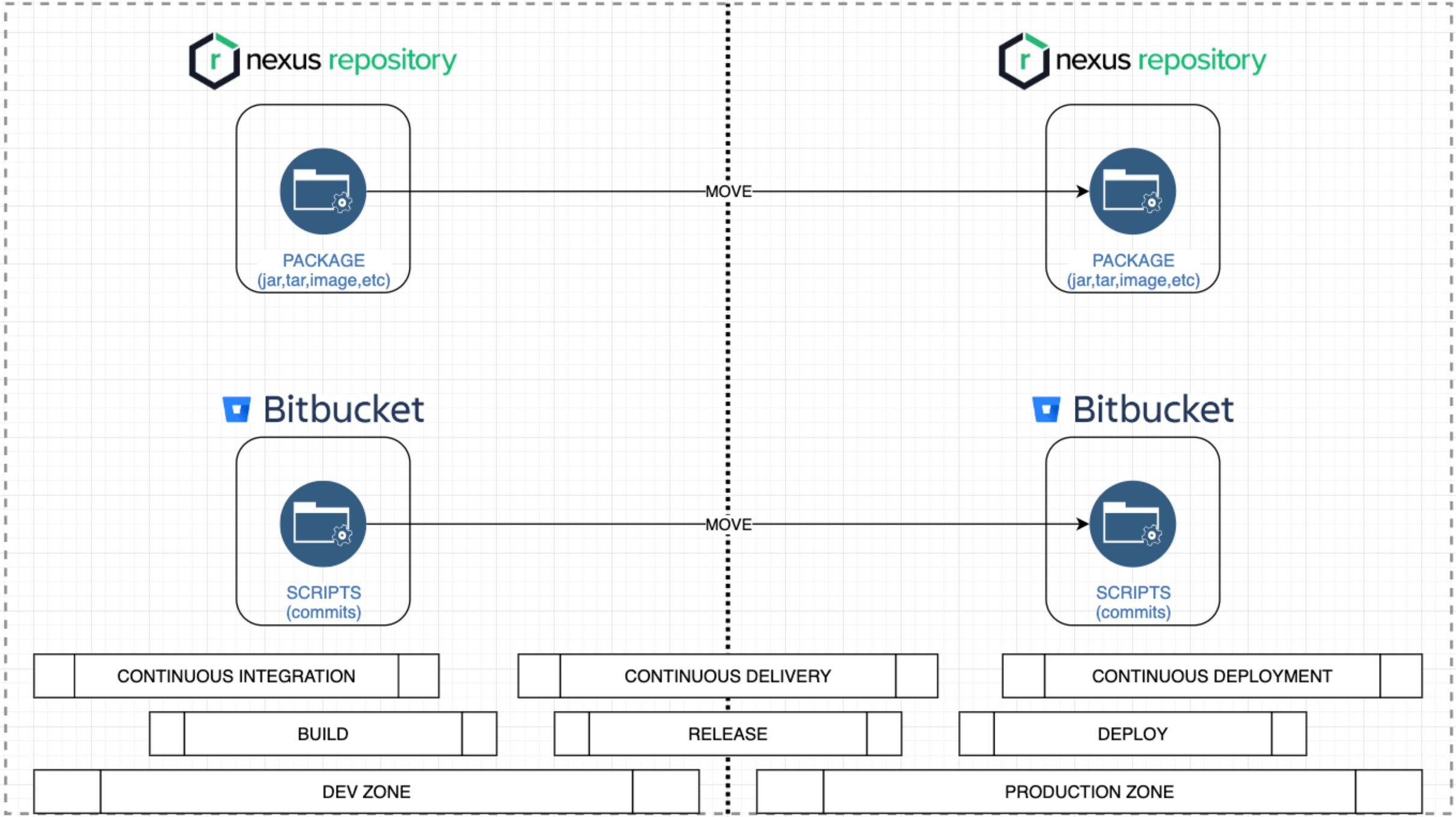

Place A
FILE
MOVE
Place B
FILE

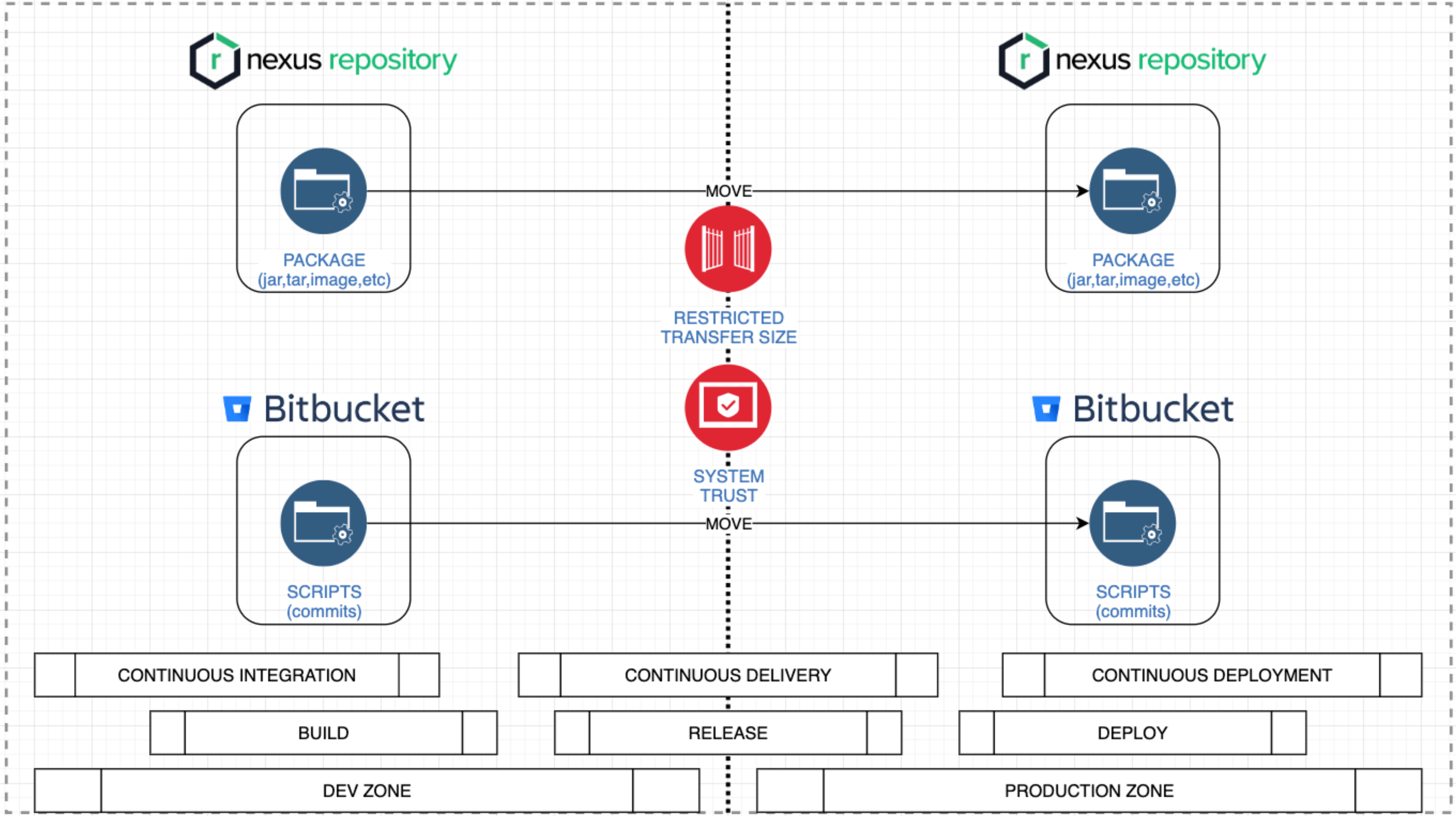

Place A
FILE
MOVE
Place B
FILE
DEV ZONE
PRODUCTION ZONE

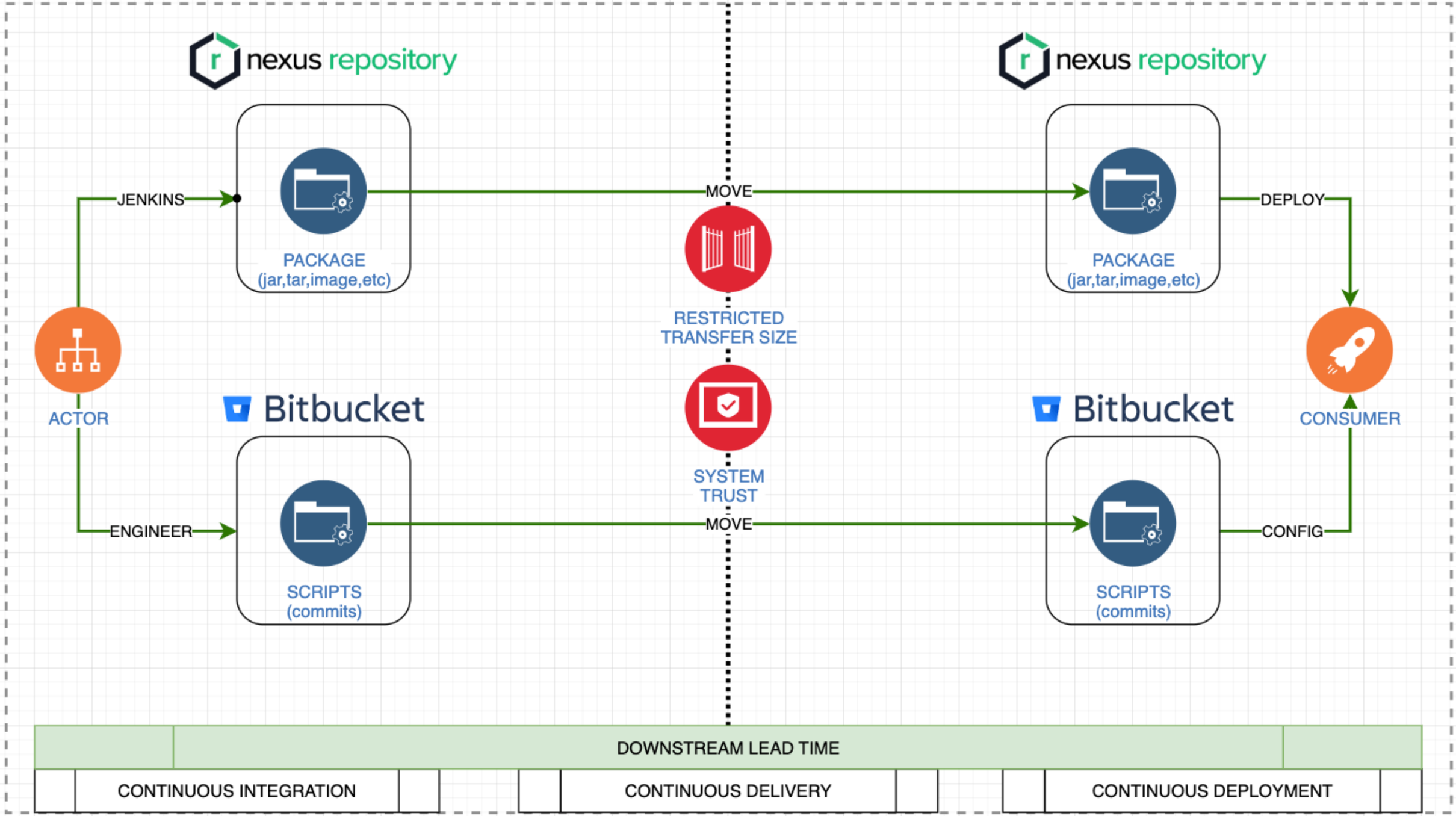

nexus repository
FILE
MOVE
nexus repository
FILE
DEV ZONE
PRODUCTION ZONE


nexus repository
FILE
MOVE
nexus repository
FILE
Bitbucket
FILE
MOVE
Bitbucket
FILE
BUILD
RELEASE
DEPLOY
DEV ZONE
PRODUCTION ZONE


nexus repository
nexus repository
FILE
MOVE
FILE
Bitbucket
Bitbucket
FILE
MOVE
FILE
CONTINUOUS INTEGRATION
CONTINUOUS DELIVERY
CONTINUOUS DEPLOYMENT
BUILD
RELEASE
DEPLOY
DEV ZONE
PRODUCTION ZONE


nexus repository
PACKAGE
(jar,tar,image,etc)
MOVE
nexus repository
PACKAGE
(jar,tar,image,etc)
Bitbucket
SCRIPTS
(commits)
MOVE
Bitbucket
SCRIPTS
(commits)
CONTINUOUS INTEGRATION
CONTINUOUS DELIVERY
CONTINUOUS DEPLOYMENT
BUILD
RELEASE
DEPLOY
DEV ZONE
PRODUCTION ZONE


nexus repository
nexus repository
PACKAGE
(jar,tar,image,etc)
MOVE
PACKAGE
(jar,tar,image,etc)
RESTRICTED
TRANSFER SIZE
Bitbucket
Bitbucket
SYSTEM
TRUST
SCRIPTS
(commits)
MOVE
SCRIPTS
(commits)
CONTINUOUS INTEGRATION
CONTINUOUS DELIVERY
CONTINUOUS DEPLOYMENT
BUILD
RELEASE
DEPLOY
DEV ZONE
PRODUCTION ZONE


nexus repository
nexus repository
JENKINS
PACKAGE
(jar,tar,image,etc)
MOVE
RESTRICTED
TRANSFER SIZE
PACKAGE
(jar,tar,image,etc)
DEPLOY
ACTOR
Bitbucket
SYSTEM
TRUST
Bitbucket
CONSUMER
ENGINEER
SCRIPTS
(commits)
MOVE
SCRIPTS
(commits)
CONFIG
DOWNSTREAM LEAD TIME
CONTINUOUS INTEGRATION
CONTINUOUS DELIVERY
CONTINUOUS DEPLOYMENT

Как будем решать

1

Разработать систему синхронизации

2

Обеспечить систему процессами безопасности

3

Достичь максимально быстрой работы

4

Сроки - вчера

Требования

Функциональные

Контракт качества

- Установленный и соблюдаемый SLA
- Автономность в сопровождении
- Прозрачность работы системы для всех

Интеграция

- Поддержка синхронизации Bitbucket
- Поддержка синхронизации Nexus
- Поддержка синхронизации Registry

Контроль целостности

- Отслеживание контрольных сумм
- Процессы безопасности
- Контроль загружаемых объектов

Требования

Нефункциональные

Скорость

Максимально быстрое выполнение бизнес-функции

Отсутствие зависимостей от смежных систем черных ящиков

Наличие разных по скорости каналов связи

Надежность

Stateless

Отсутствие неконтролируемого влияния при потере 50% мощности

Отсутствие PoF

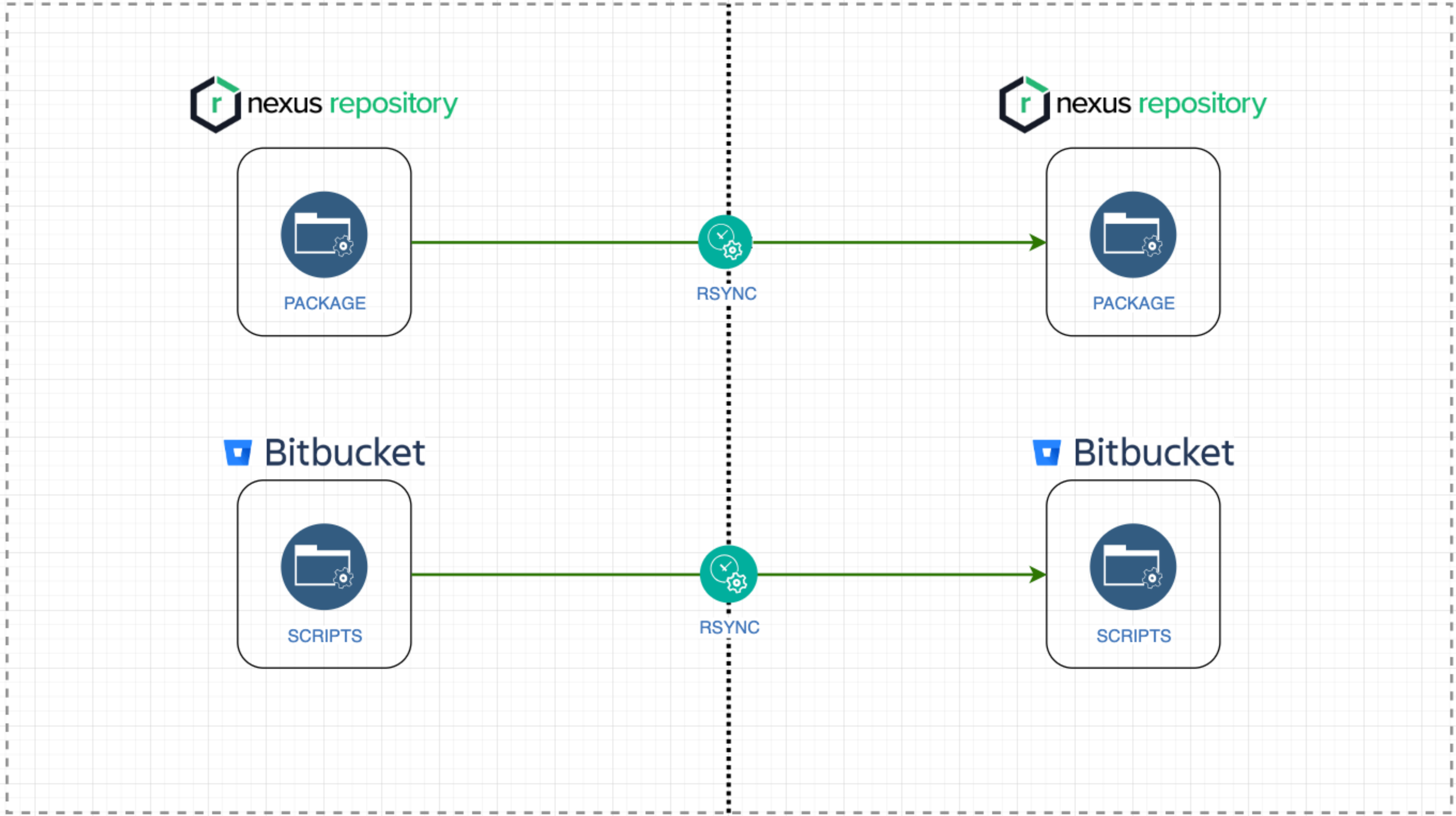
Готовность к отказу интегрированных систем хранения

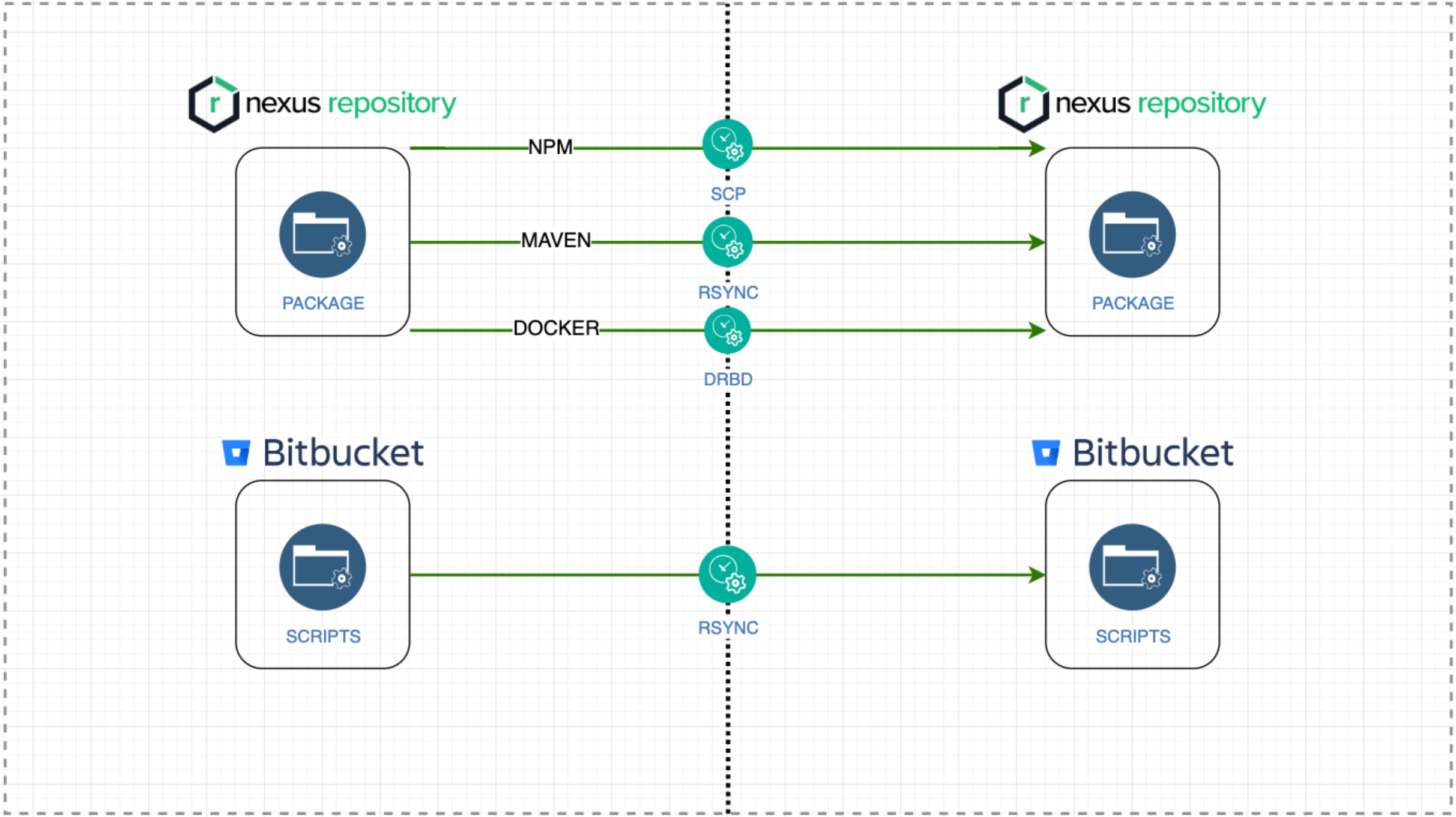
Эластичность

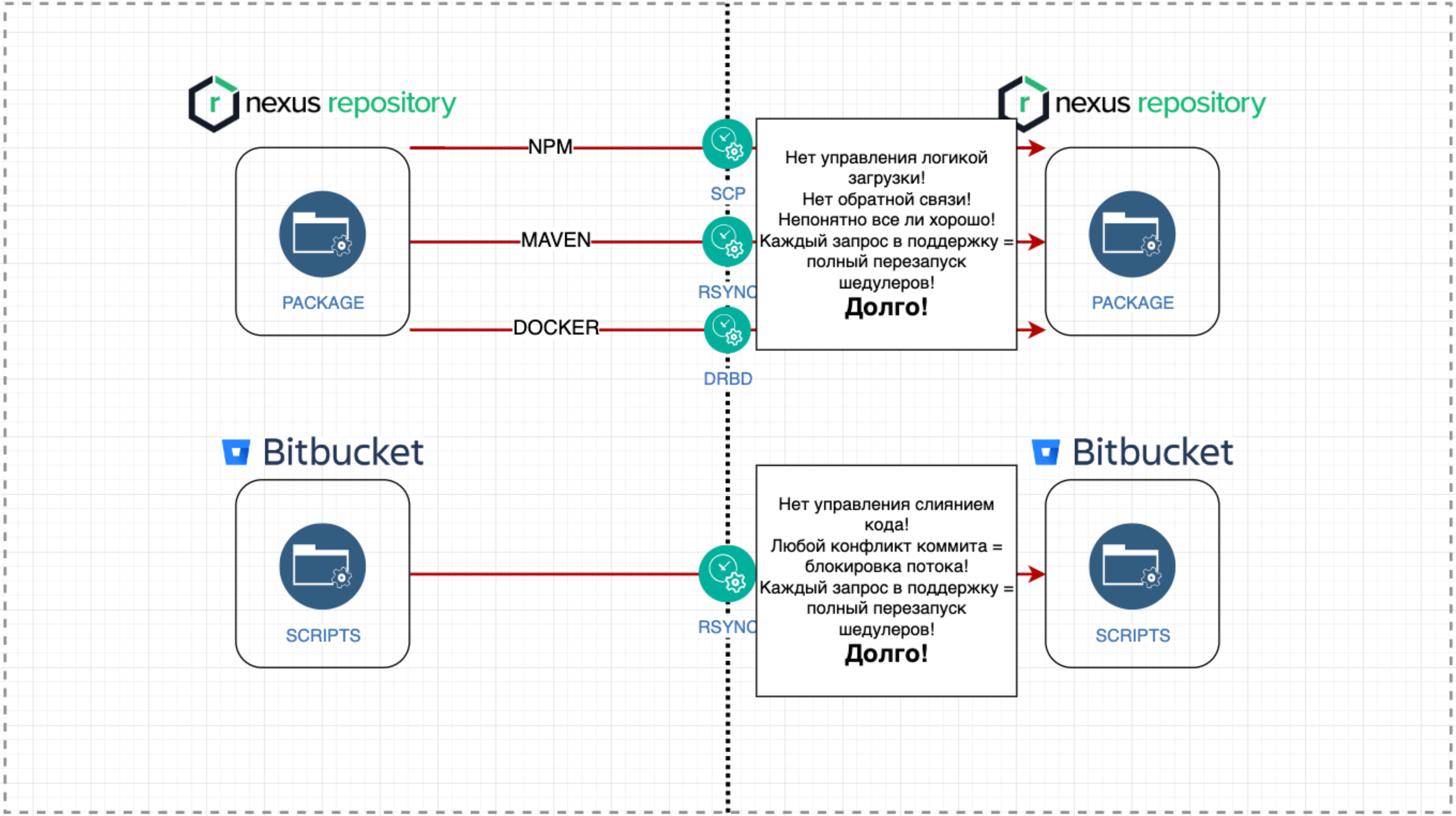
Горизонтальное масштабирование

Обработка большого объема данных без ущерба трафика малого объема

Управление типами потоков и качеством/количеством их обработчиков


nexus repository
PACKAGE
RSYNC
nexus repository
PACKAGE
Bitbucket
SCRIPTS
RSYNC
Bitbucket
SCRIPTS


nexus repository
nexus repository
NPM
SCP
MAVEN
RSYNC
DOCKER
DRBD
PACKAGE
PACKAGE
Bitbucket
Bitbucket
SCRIPTS
RSYNC
SCRIPTS

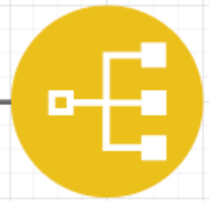

nexus repository
NPM
MAVEN
DOCKER
PACKAGE
SCP
RSYNC
DRBD
Нет управления логикой загрузки!
Нет обратной связи!
Непонятно все ли хорошо!
Каждый запрос в поддержку = полный перезапуск шедулеров!
Долго!
nexus repository
PACKAGE
Bitbucket
SCRIPTS
RSYNC
Нет управления слиянием кода!
Любой конфликт коммита = блокировка потока!
Каждый запрос в поддержку = полный перезапуск шедулеров!
Долго!
Bitbucket
SCRIPTS

NEXUS A

EVENT

API-GATE

EVENT



BITBUCKET A

NEXUS B

BITBUCKET B

API-GATE

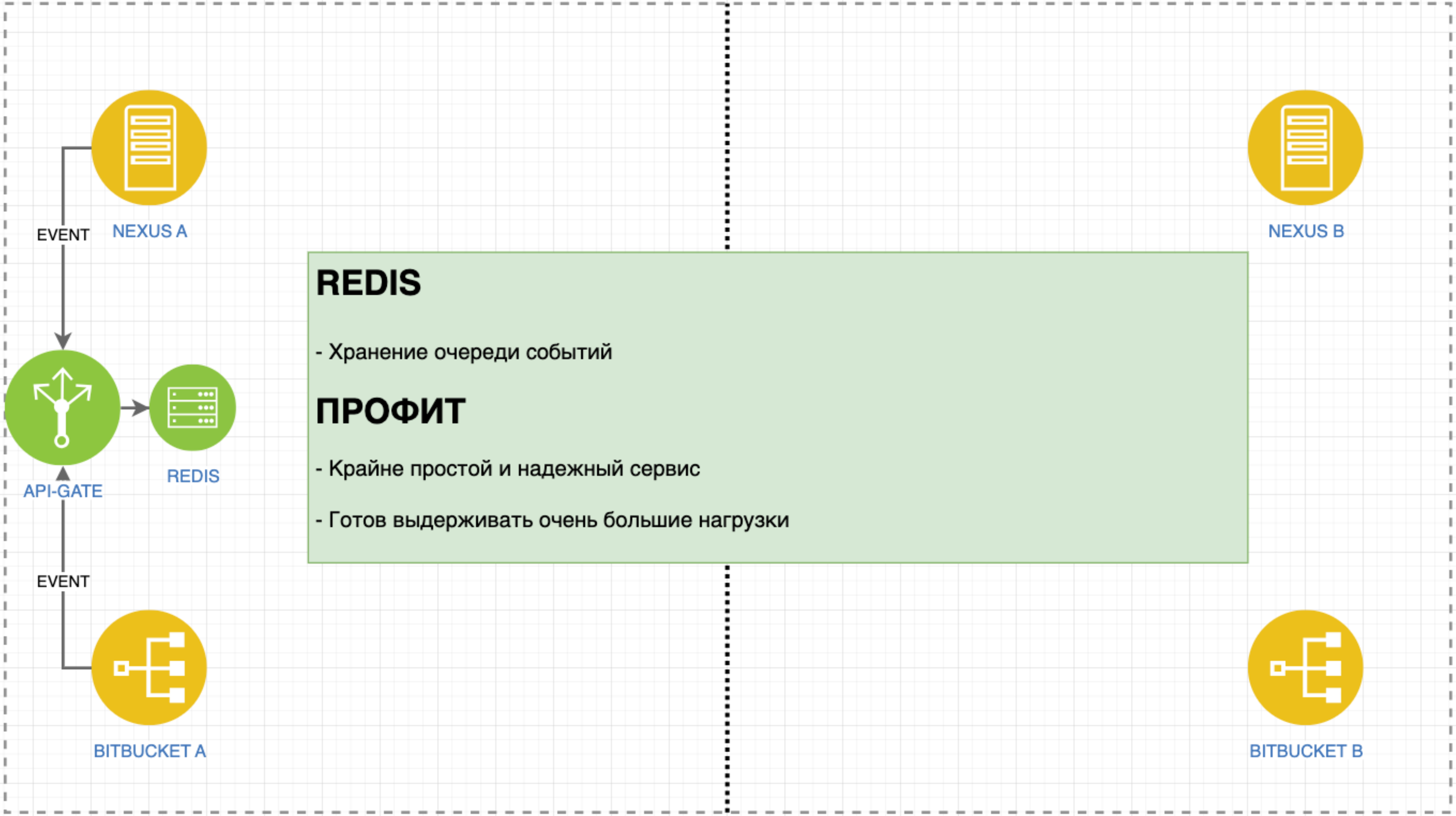
- Прием событий загрузки изменений

- Загрузка событий в очередь

ПРОФИТ

- Крайне простой и надежный сервис

- Готов выдерживать очень большие нагрузки



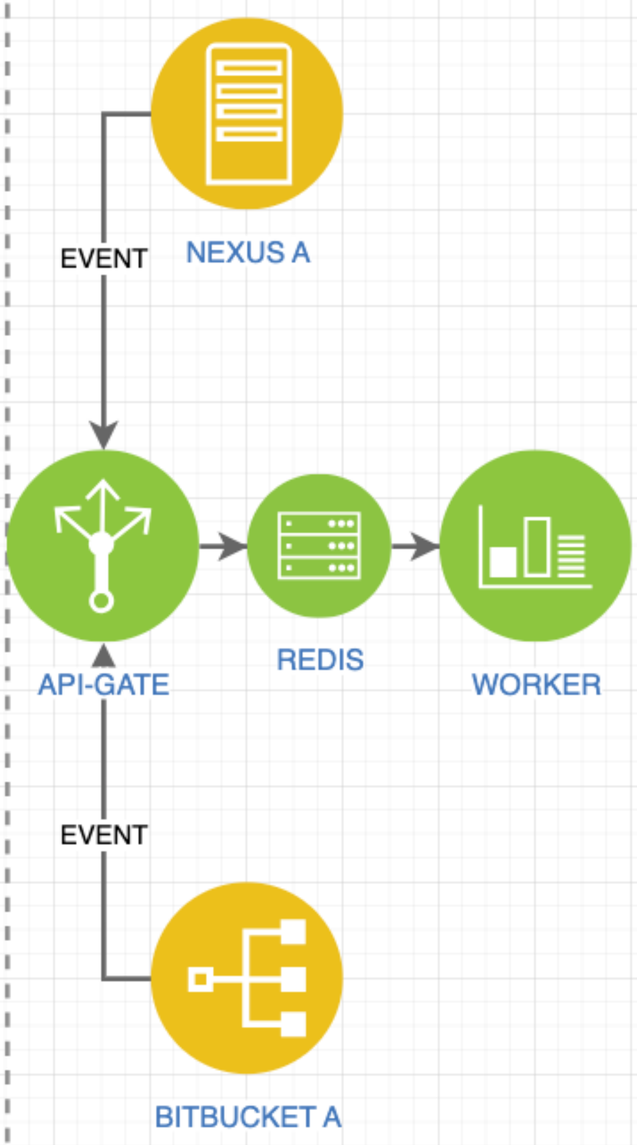
REDIS

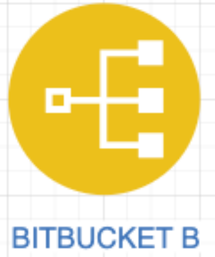
- Хранение очереди событий

ПРОФИТ

- Крайне простой и надежный сервис

- Готов выдерживать очень большие нагрузки





WORKER

- Обработка и сериализация очереди событий

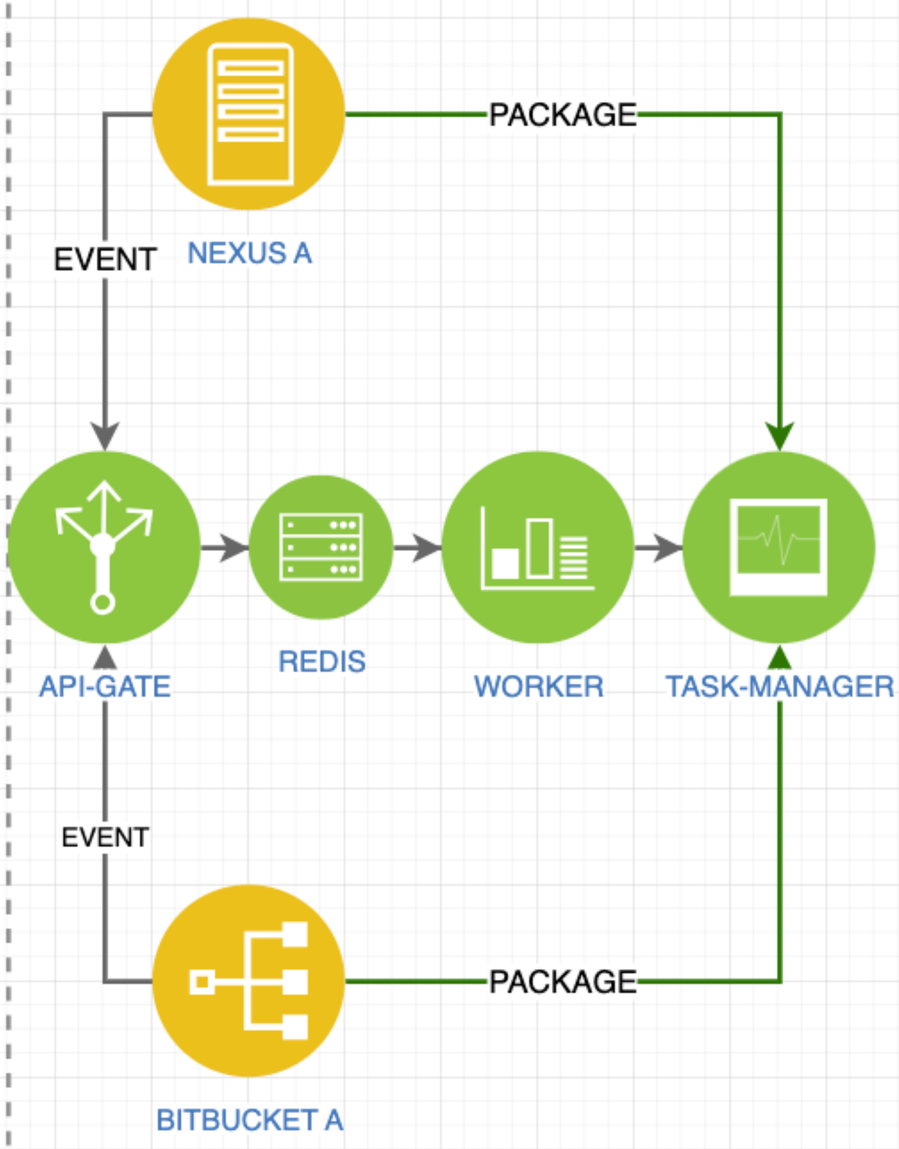
- Загрузка событий в персистентную БД

ПРОФИТ

- Крайне простой и надежный сервис

- Готов выдерживать очень большие нагрузки

- Первично управляет приоритетом обработки



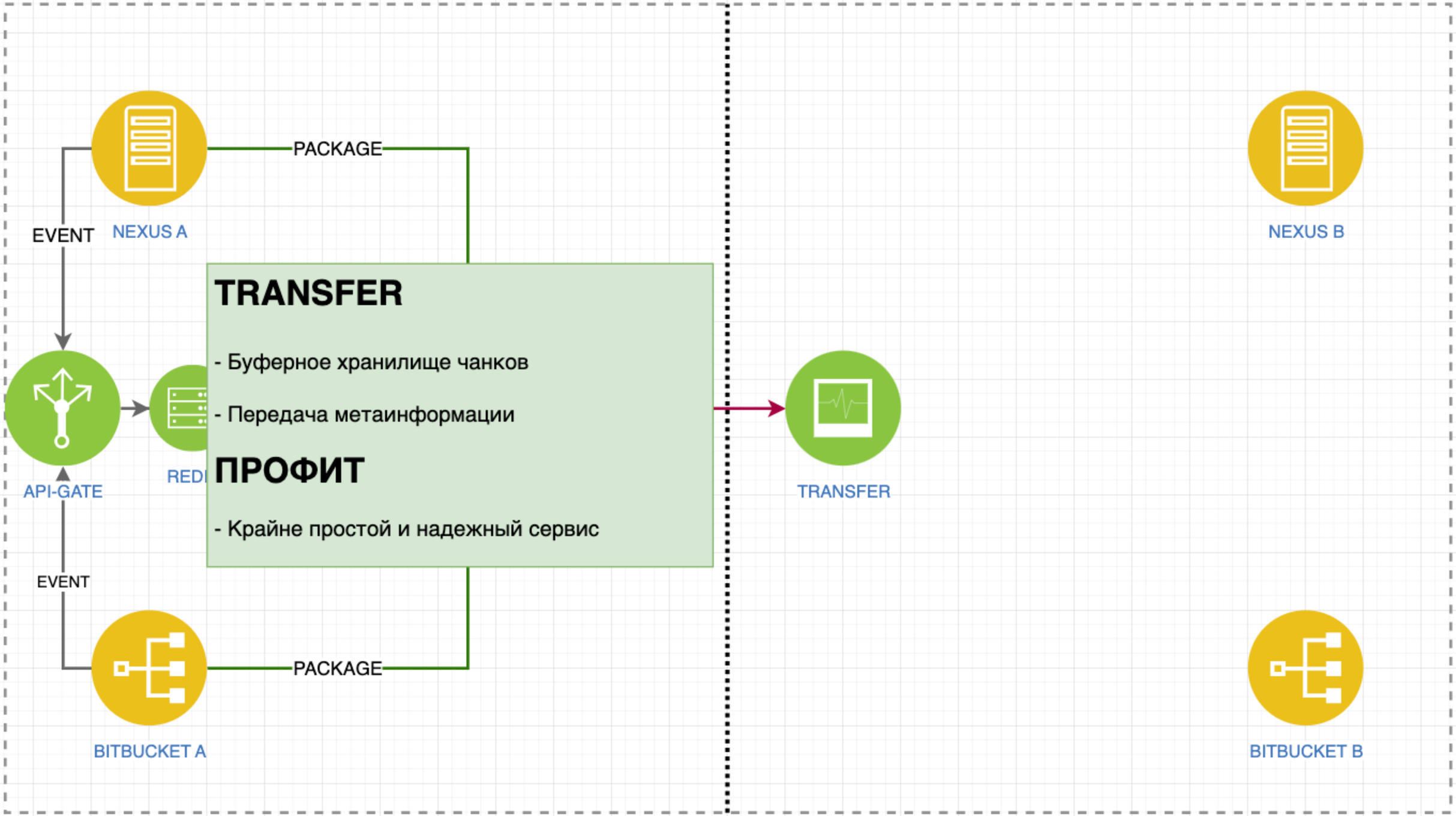
TASK-MANAGER

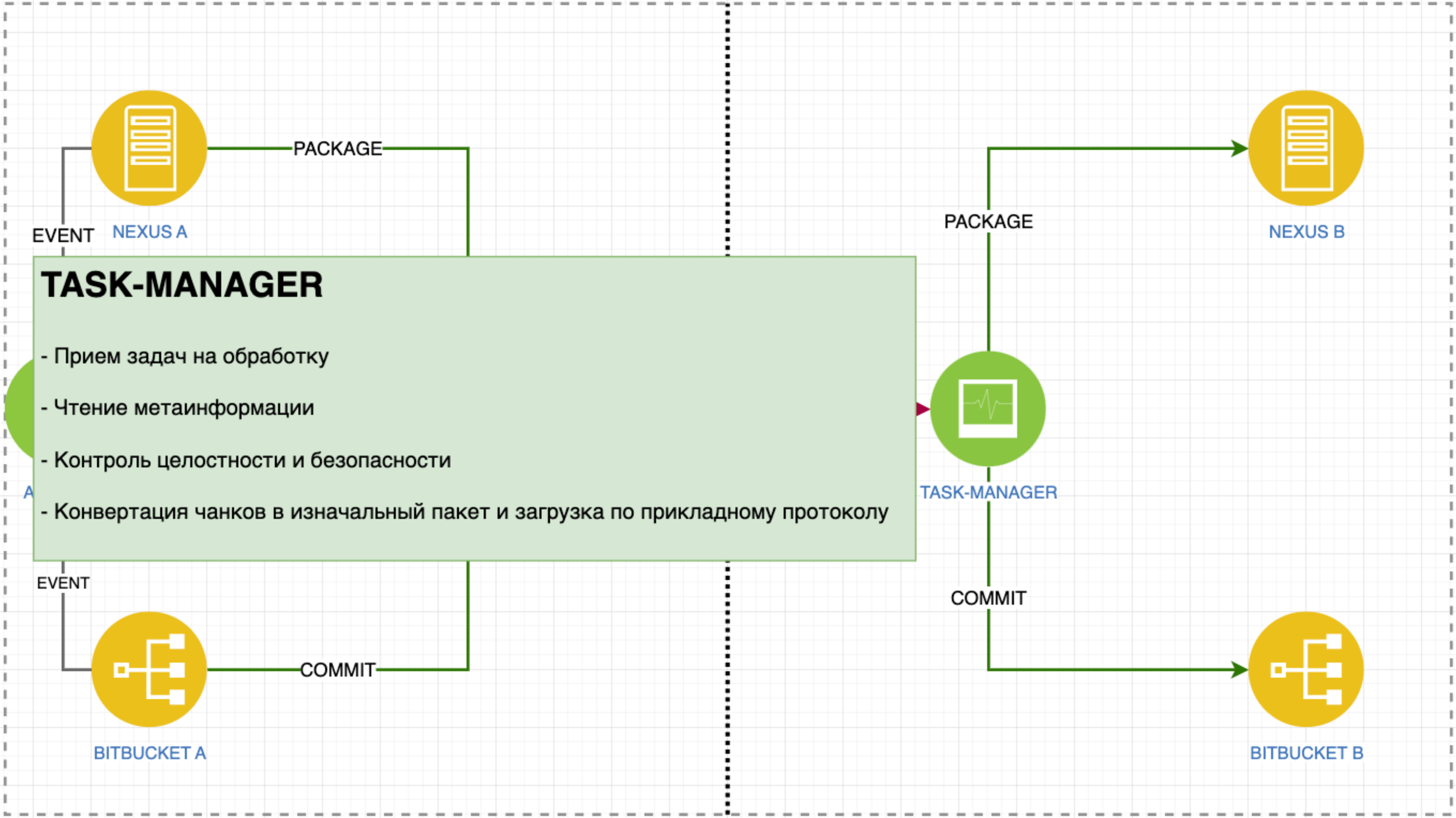
- Обработка событий и создание задач
- Скачивание объектов из систем хранения по прикладному протоколу
- Контроль целостности и безопасности
- Формирование чанков и отправка в буфер по системному протоколу

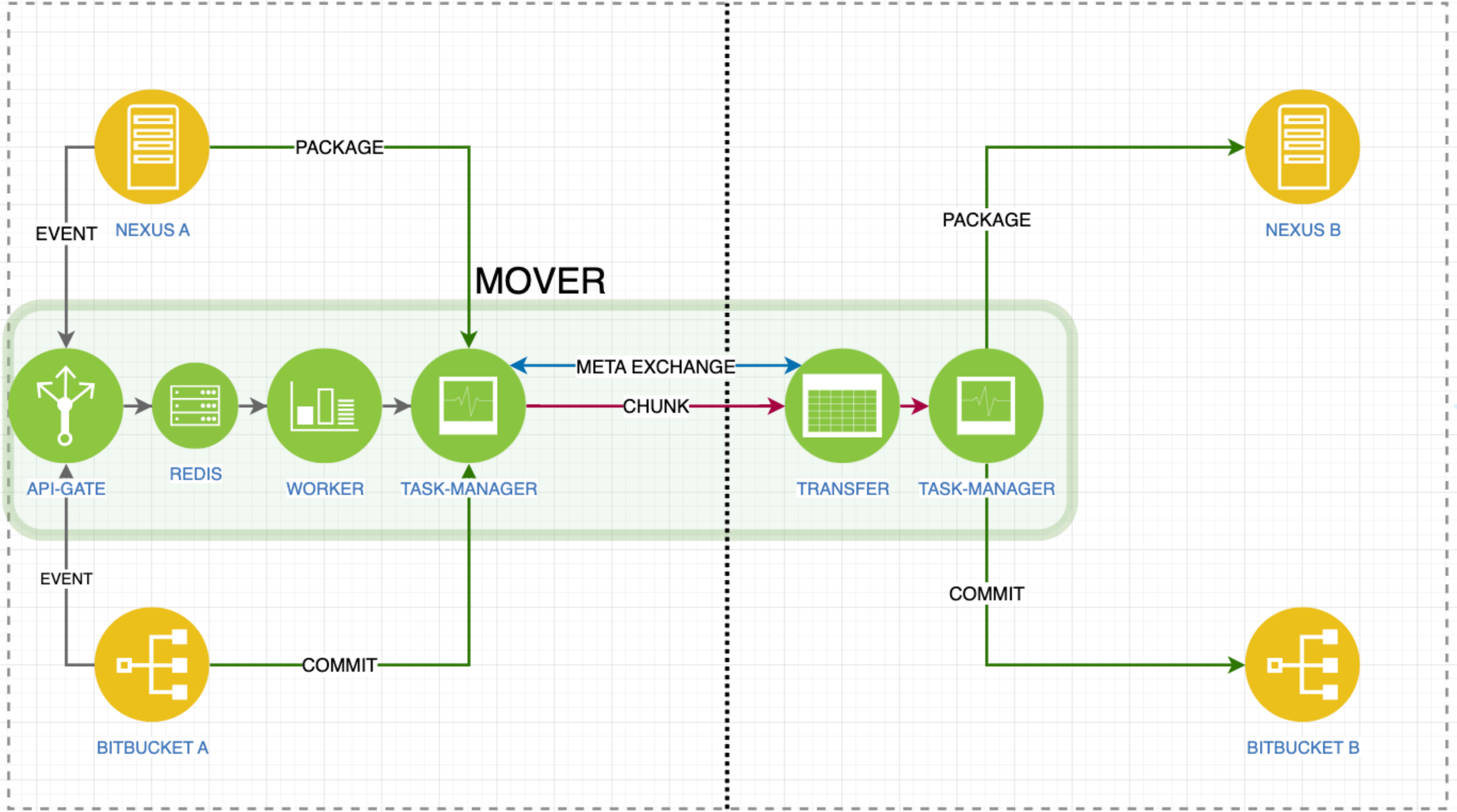
ПРОФИТ

- Контроль маршрутов синхронизации
- Автоматическое определение очередности выполнения

BITBUCKET B

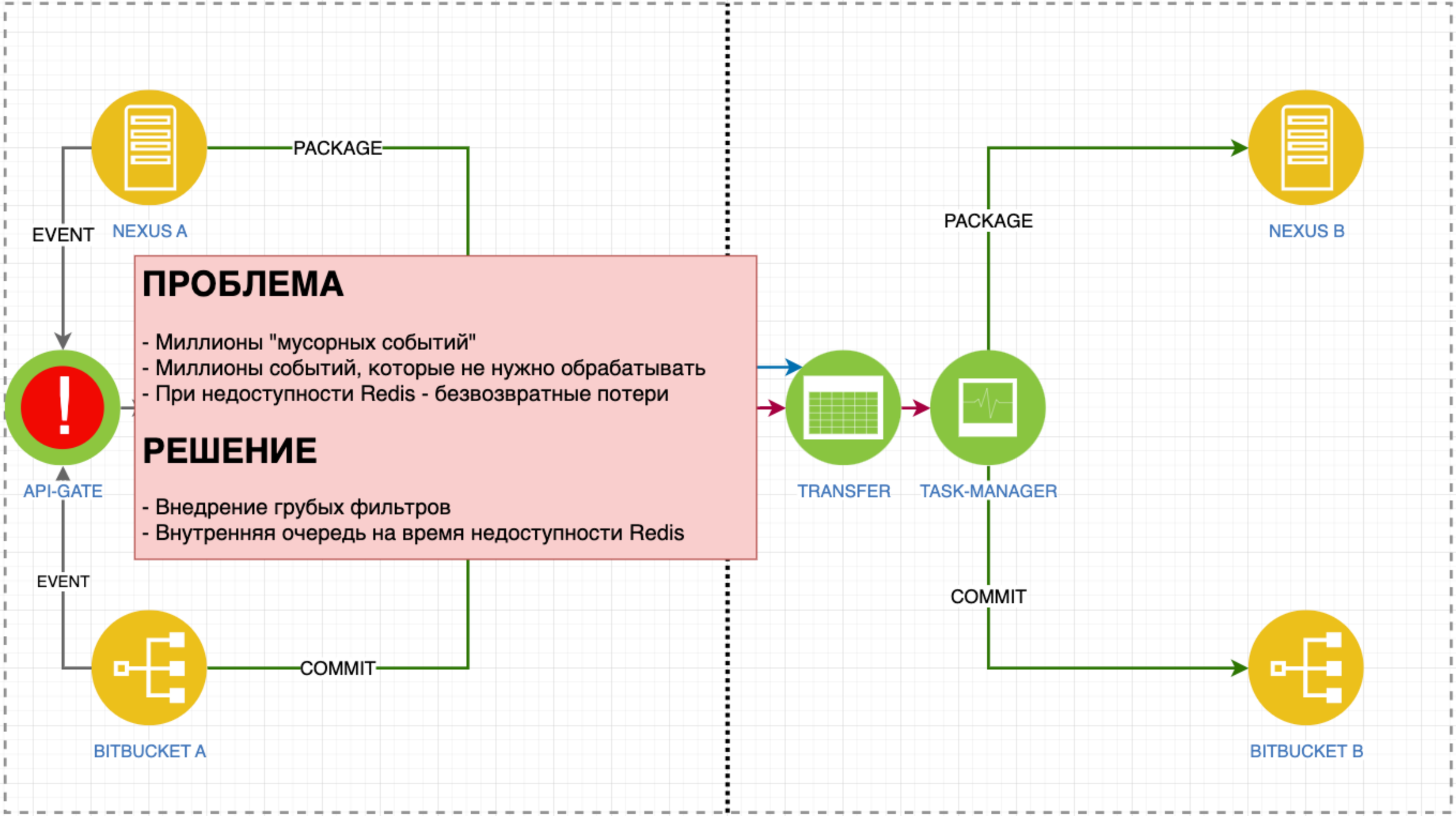

PACKAGE
EVENT
NEXUS A
NEXUS B
TRANSFER
- Буферное хранилище чанков
- Передача метаинформации
ПРОФИТ
- Крайне простой и надежный сервис
API-GATE
TRANSFER
EVENT
PACKAGE
BITBUCKET A
BITBUCKET B

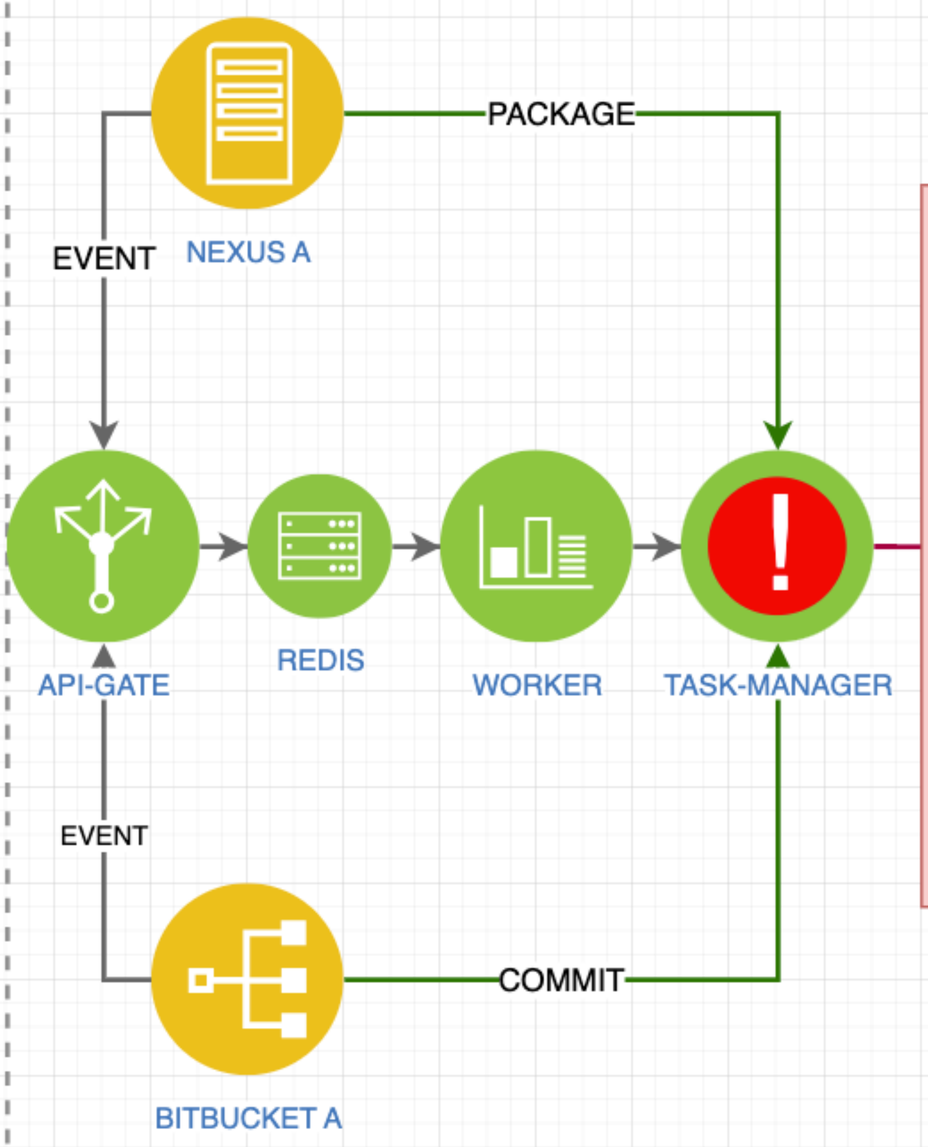

PACKAGE
EVENT
NEXUS A
PACKAGE
NEXUS B
TASK-MANAGER
- Прием задач на обработку
- Чтение метаинформации
- Контроль целостности и безопасности
- Конвертация чанков в изначальный пакет и загрузка по прикладному протоколу
TASK-MANAGER
EVENT
COMMIT
COMMIT
BITBUCKET A
BITBUCKET B

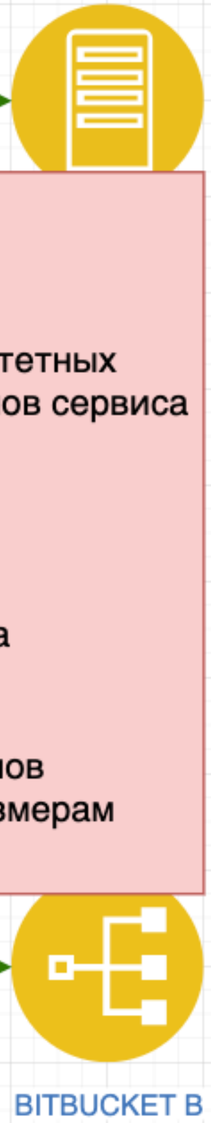

NEXUS A
PACKAGE
EVENT
MOVER
NEXUS B
PACKAGE
META EXCHANGE
CHUNK
API-GATE
REDIS
WORKER
TASK-MANAGER
TRANSFER
TASK-MANAGER
EVENT
COMMIT
COMMIT
BITBUCKET A
BITBUCKET B

Оно заработало! Но…

Мы конечно же ошибались


PACKAGE
EVENT
NEXUS A
PACKAGE
NEXUS B
ПРОБЛЕМА
- Миллионы "мусорных событий"
- Миллионы событий, которые не нужно обрабатывать
- При недоступности Redis - безвозвратные потери
РЕШЕНИЕ
- Внедрение грубых фильтров
- Внутренняя очередь на время недоступности Redis
API-GATE
TRANSFER
TASK-MANAGER
EVENT
COMMIT
COMMIT
BITBUCKET A
BITBUCKET B



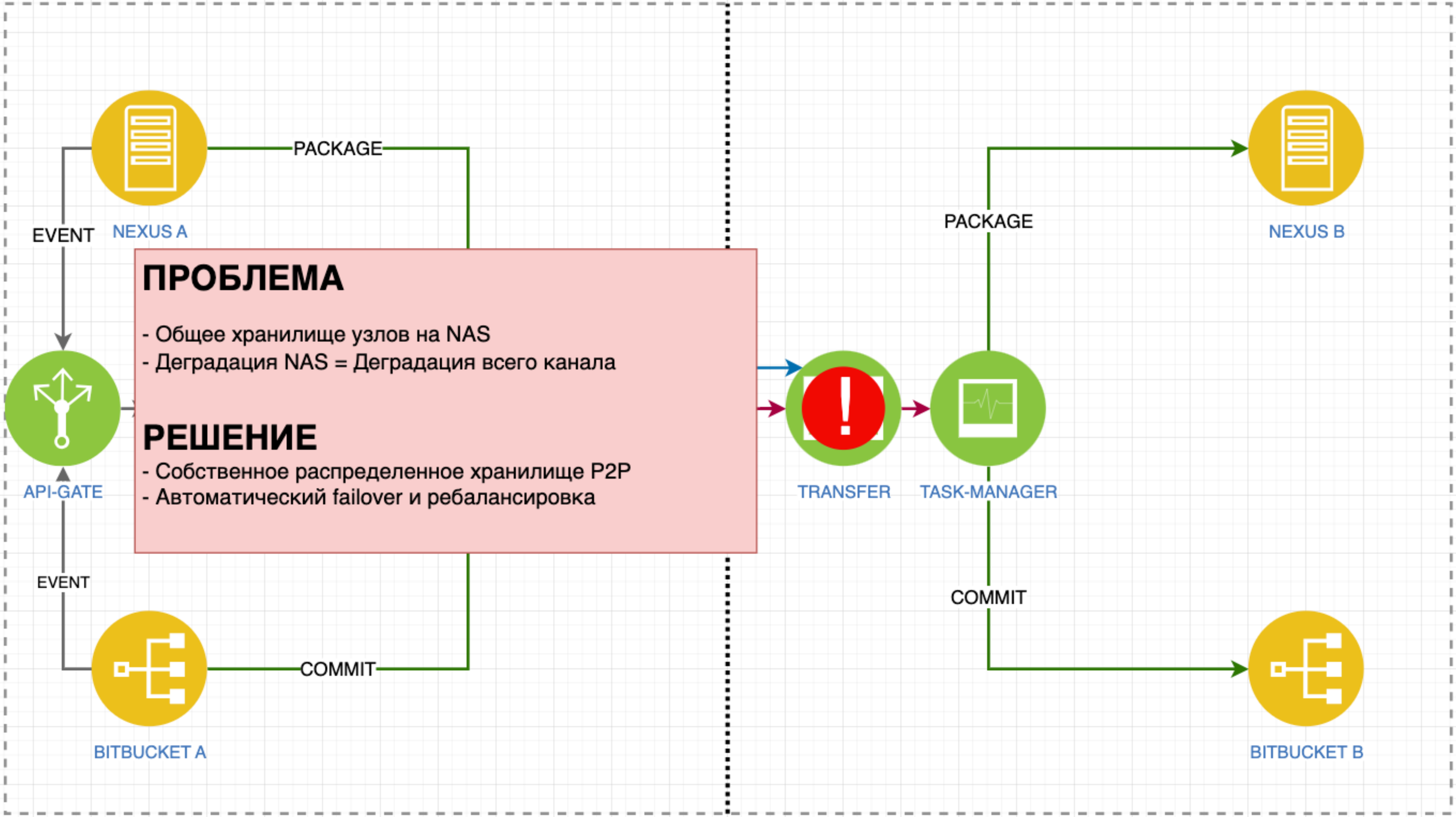


ПРОБЛЕМА

- Разная сложность задач блокирует всю очередь
- Задачи высокого приоритета теряются в очереди менее приоритетных
- Большие размеры объектов могут выводить из строя группы узлов сервиса
- Бесконечные петли событие-действие-событие

РЕШЕНИЕ

- Внедрение системы приоритетов и RQUID сквозная трассировка
- Разделение потоков задач на типы систем хранения
- Разделение потоков задач на типы обрабатываемых объектов
- Балансировка в зависимости от сложности задач и нагрузки узлов
- Внедрение тонкой настройки синхронных потоков, по типам, размерам батча, требуемой операции


PACKAGE
EVENT
NEXUS A
PACKAGE
NEXUS B
ПРОБЛЕМА
- Общее хранилище узлов на NAS
- Деградация NAS = Деградация всего канала
РЕШЕНИЕ
- Собственное распределенное хранилище P2P
- Автоматический failover и ребалансировка
API-GATE
TRANSFER
TASK-MANAGER
EVENT
COMMIT
COMMIT
BITBUCKET A
BITBUCKET B

Цифры, которые ~~пугают~~ пугали

и продолжают расти

~2,5M
Артефактов в сутки

~1,5M
Коммитов в сутки

Nexus

24
Инсталляции

7
Типов репозиториев

Bitbucket

12
Инсталляций

2
Типа репозиториев

Полученный результат

x60

Сократили Lead Time

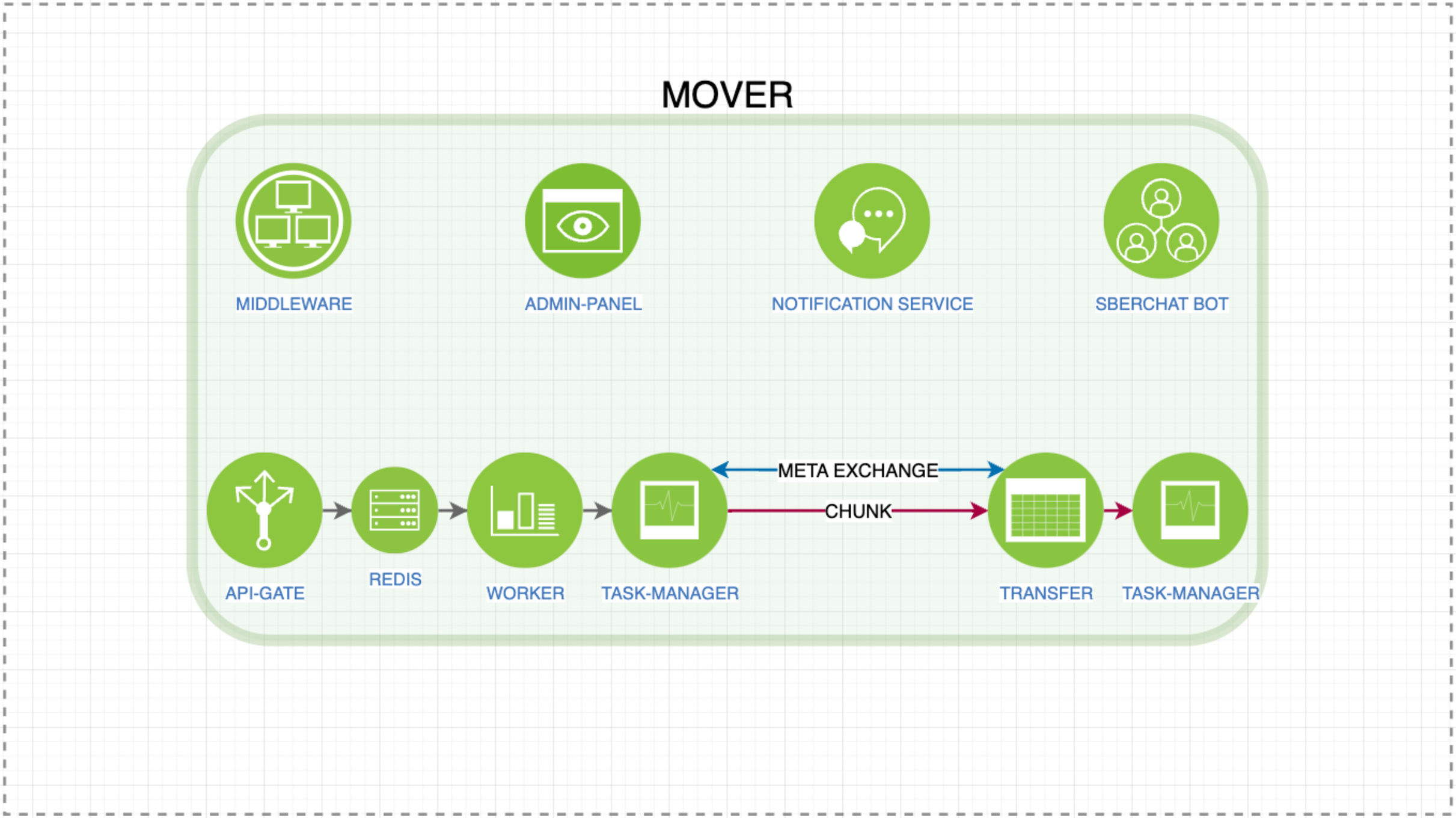
Что привело к успеху

Желание глобально изменить ситуацию

Постоянное использование собственного сервиса

Свобода действий в принятии решений

Правильный DevOps в команде разработки


MOVER
MIDDLEWARE
ADMIN-PANEL
NOTIFICATION SERVICE
SBERCHAT BOT
META EXCHANGE
CHUNK
API-GATE
REDIS
WORKER
TASK-MANAGER
TRANSFER
TASK-MANAGER

MOVER

MIDDLEWARE

ADMIN-PANEL

NOTIFICATION SERVICE

SBERCHAT BOT

API-GATE

MIDDLEWARE - REST для системных и внутренних интеграций

ADMIN PANEL - WEB-UI для линий сопровождения системы

NOTIFICATION SERVICE - сервис оповещений

SBERCHAT BOT - интерактивный помощник с ИИ

TASK-MANAGER

Заключение и напутствие

Не все инструменты DevOps можно найти в Github и Docker

Там где не работает практика – работает фантазия

Лучший сервис делает конечный пользователь


Спасибо