Skillbox

Service Discovery & Configuration

Обнаружение и регистрация сервисов

Андрей Гордиенков

Solution Architect

ABAX

В прошлом модуле

- Интеграция сервисов, какие способы есть
- Протоколы интеграции
- Шаблоны обмена сообщениями

В этом модуле

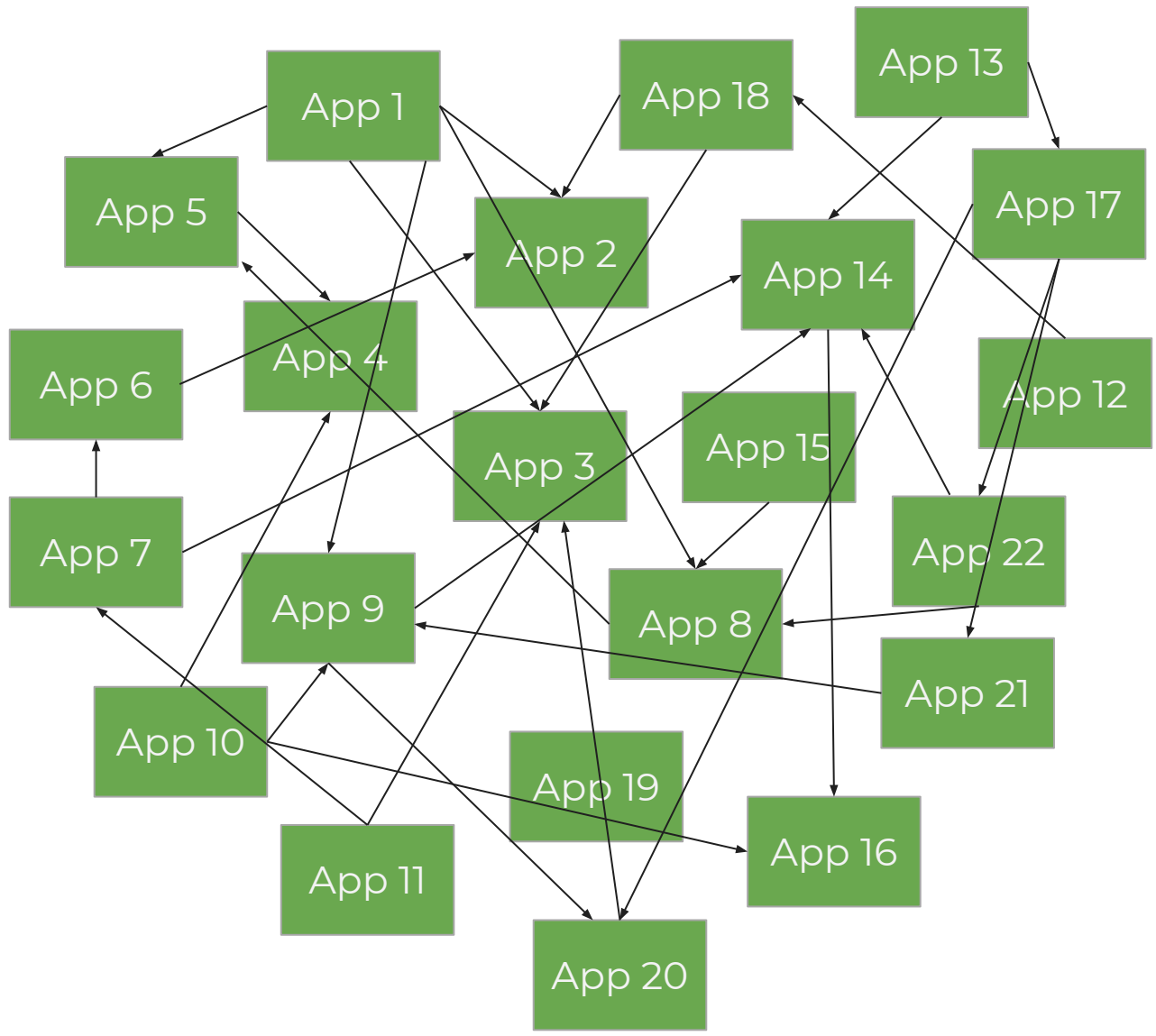
- Управлять добавлением и отключением сервисов в системе
- Конфигурация сервисов

Skillbox

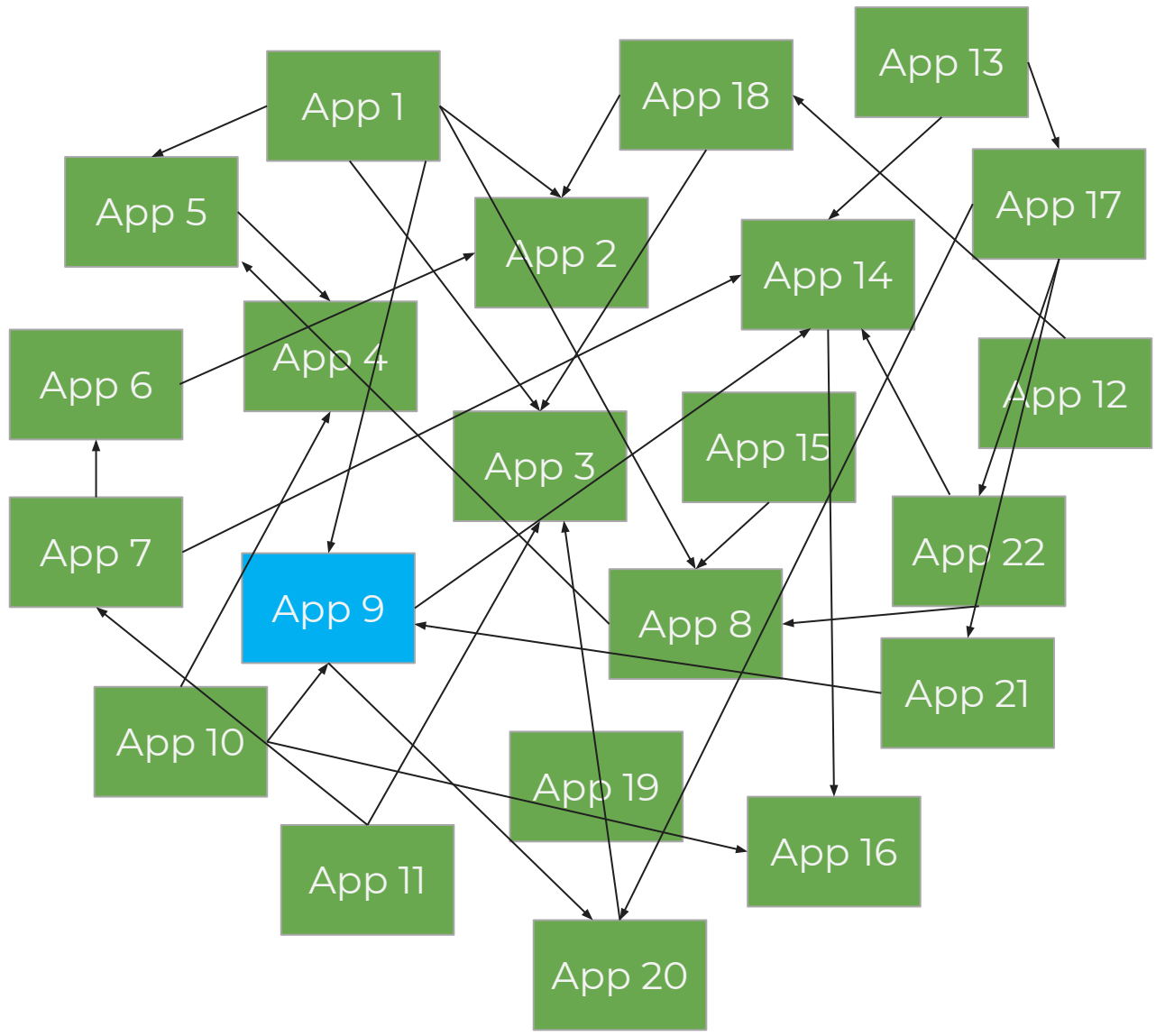
На этом уроке

- Проблемы введения сервисов в эксплуатацию
- Управление регистрацией

Постановка проблемы

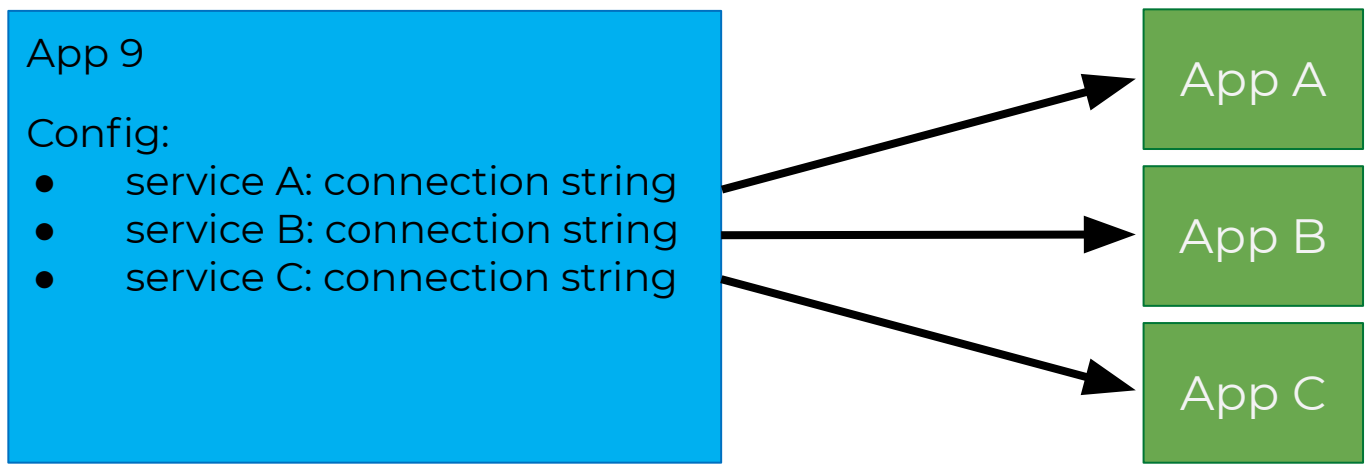


Постановка проблемы



Постановка проблемы

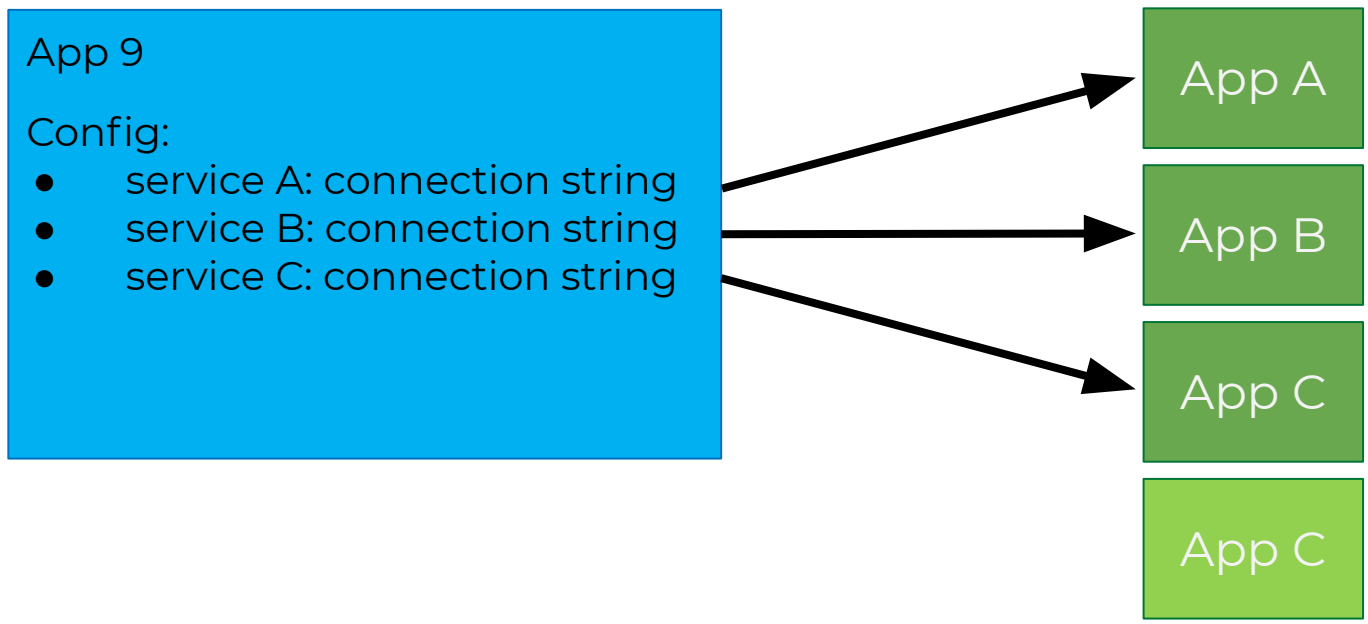
Как сервис узнает, где находятся другие сервисы, к кому подсоединяться?



Конфигурация сервиса содержит заранее известные адреса сервисов, с которыми идёт коммуникация.

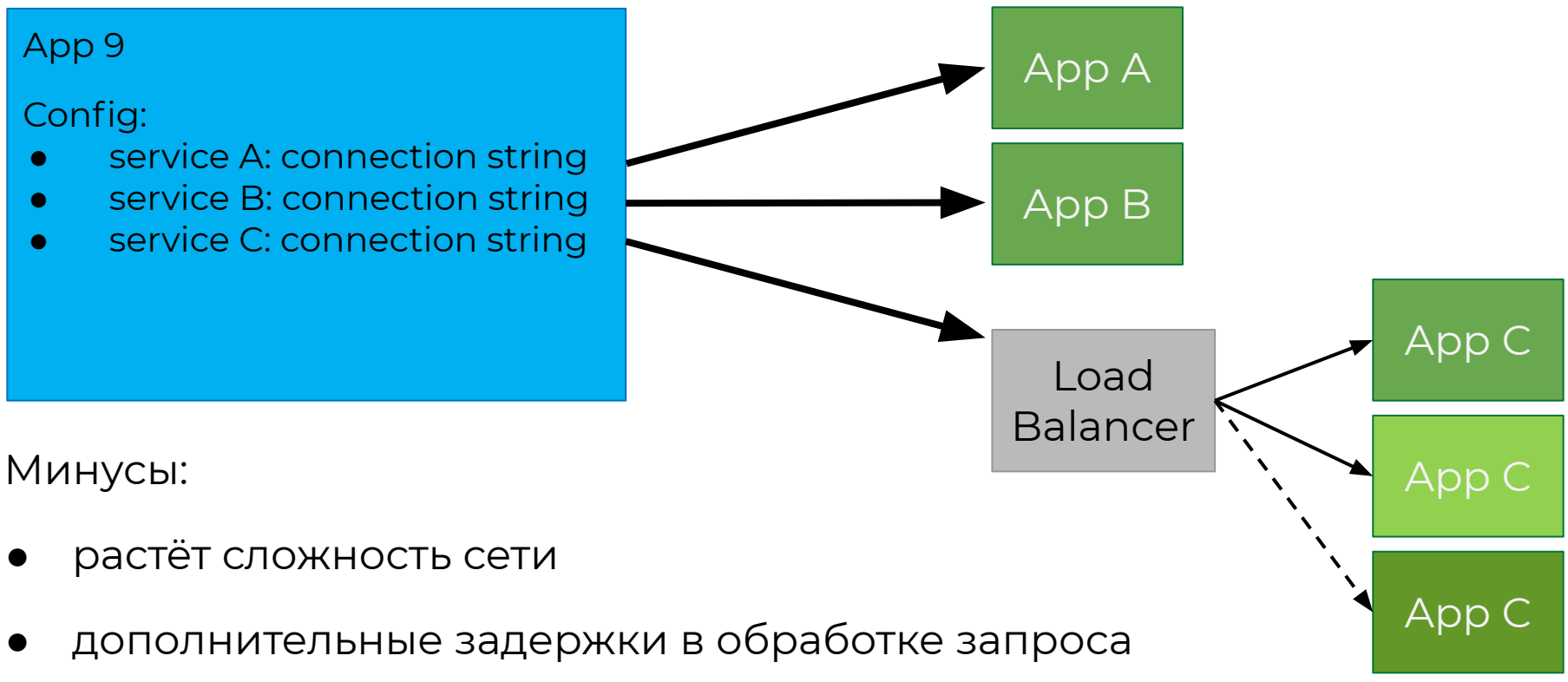
Статическая адресация

Как сказать нашему сервису, что он может использовать
ещё и копию сервиса C?



Статическая адресация

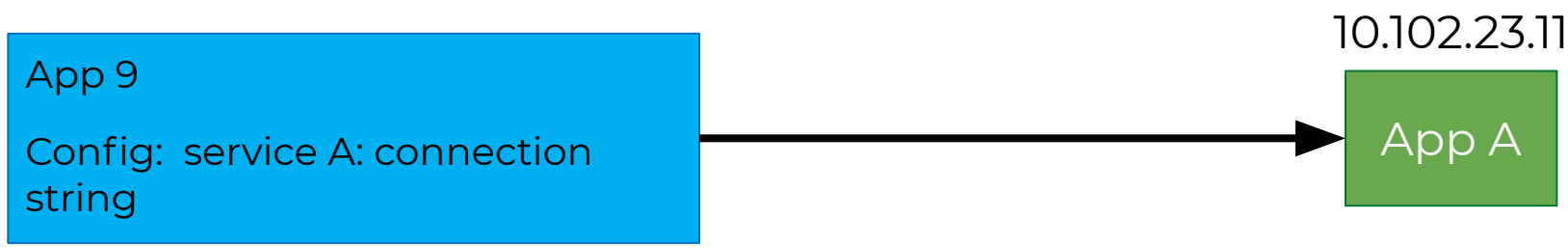
Как сказать нашему сервису, что он может использовать
ещё и копию сервиса C?



Минусы:

- растёт сложность сети
- дополнительные задержки в обработке запроса

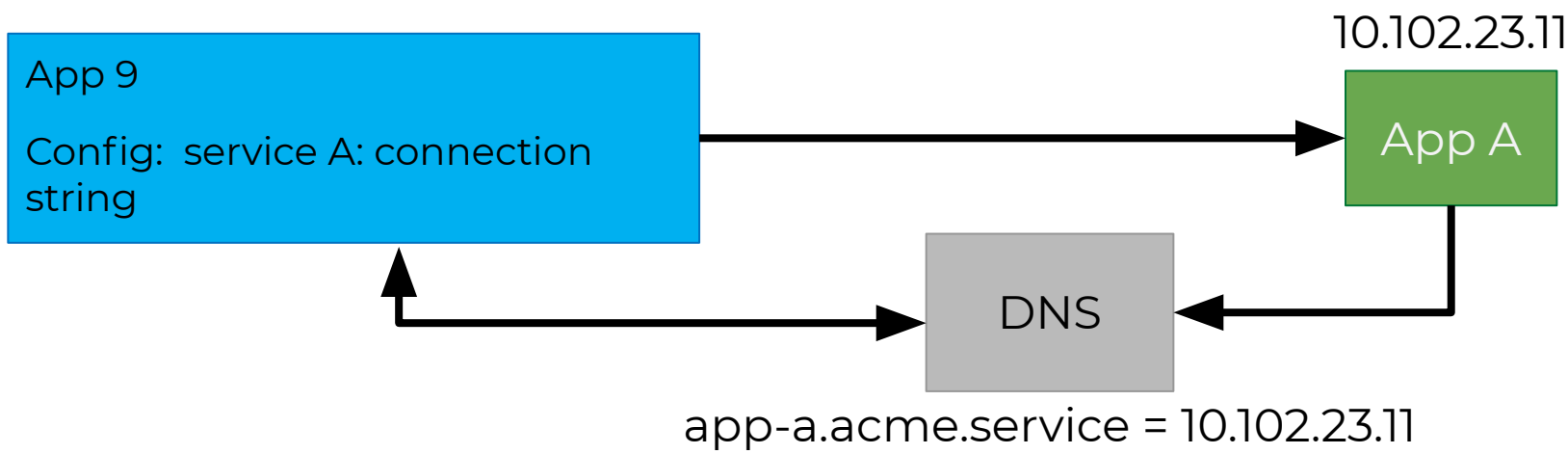
Статическая адресация



Требования:

- знание адреса

Статическая адресация



Требования:

- знание адреса
- существование DNS (Domain Name Service)

Минусы:

- растёт физическая сложность сети
- дополнительные задержки в обработке запроса
- ручная конфигурация

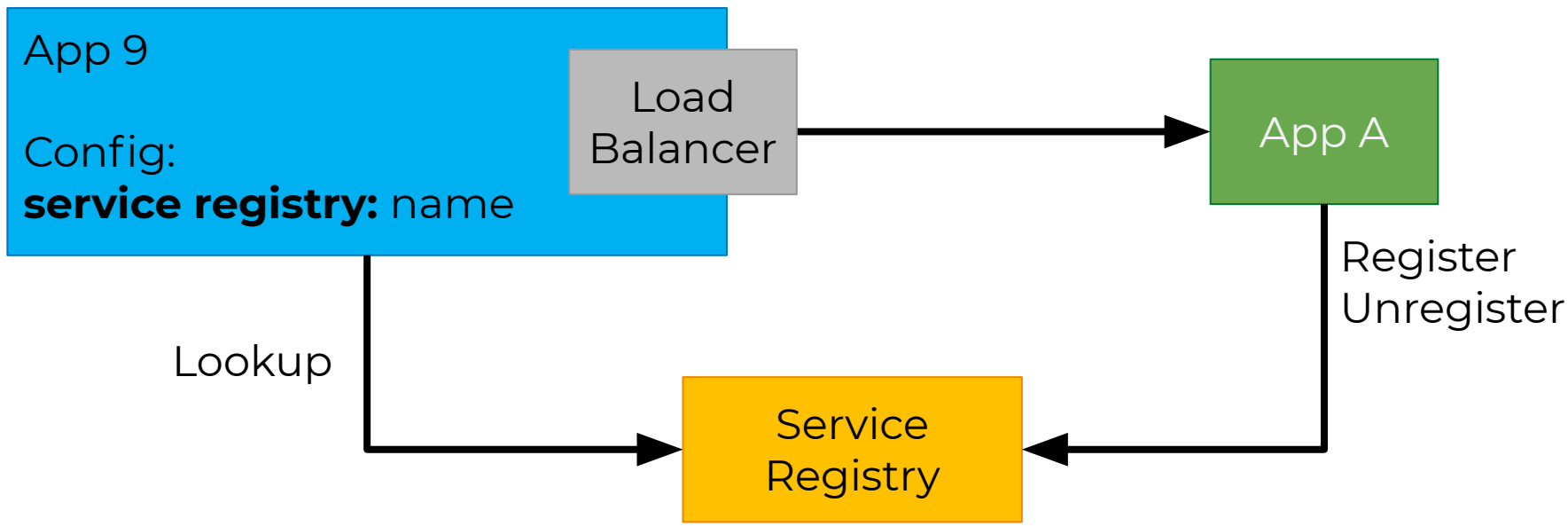
Основные проблемы

- Общая изменчивость конфигурации сети:
  - динамические адреса сервисов
  - динамическое количество сервисов
- Сложно оценить общую сложность коммуникации
- Трудно находить потребителей

Решение: использование регистратора сервисов:

- Client-side Discovery Pattern
- Server-side Discovery Pattern

Client-side Discovery Pattern



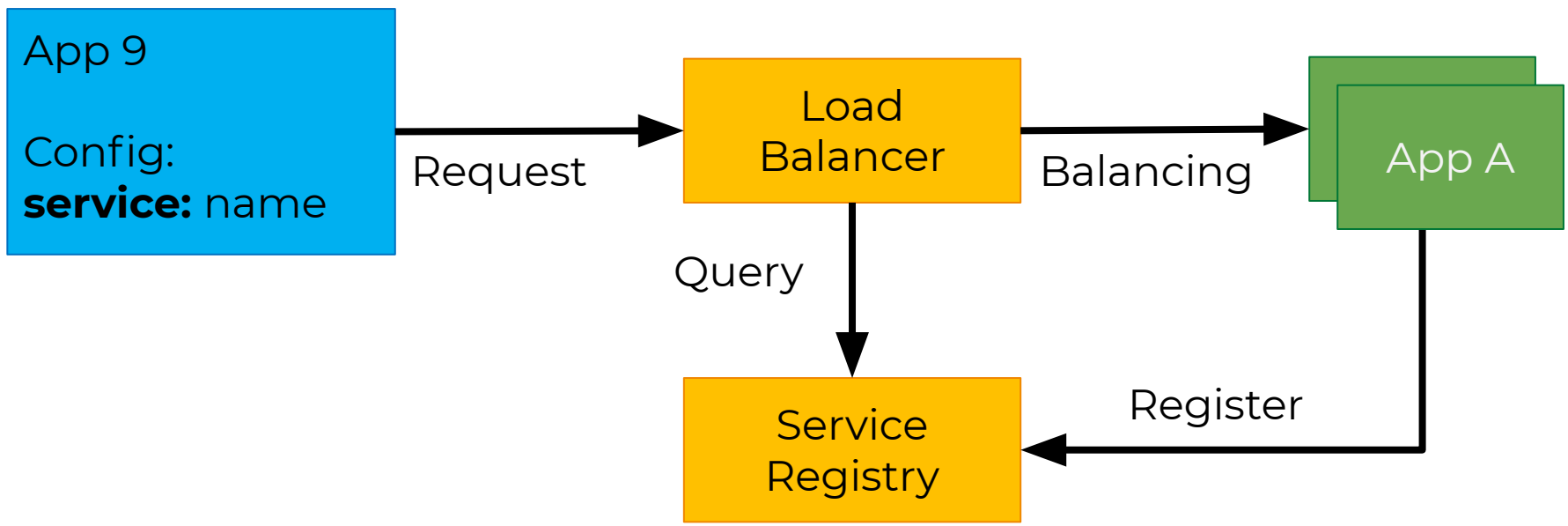
Новые сервисы самостоятельно сообщают о том, что они готовы к работе.

Клиент (App9) запрашивает информацию о доступных сервисах и принимает решение, к кому обратиться.

Плюсы: простота реализации.

Минусы: клиент жёстко связывается с Service Registry.

Server-side Discovery Pattern



Ответственность по нахождению лучшего сервиса для обработки запроса переходит балансировщику нагрузки.

Плюсы: клиент не связан с Service Registry.

Минусы: балансировщик нагрузки становится узким местом, необходимость обеспечения высокой надёжности.

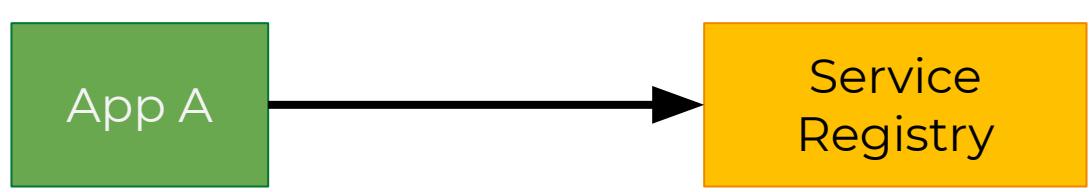
Server-side Discovery Pattern

Поддерживается многими продуктами по умолчанию:

- AWS Elastic Load Balancer
- Azure Load Balancer
- NGINX
- Kubernetes
- Traefik и др.

Регистрация сервисов

Самостоятельная регистрация:



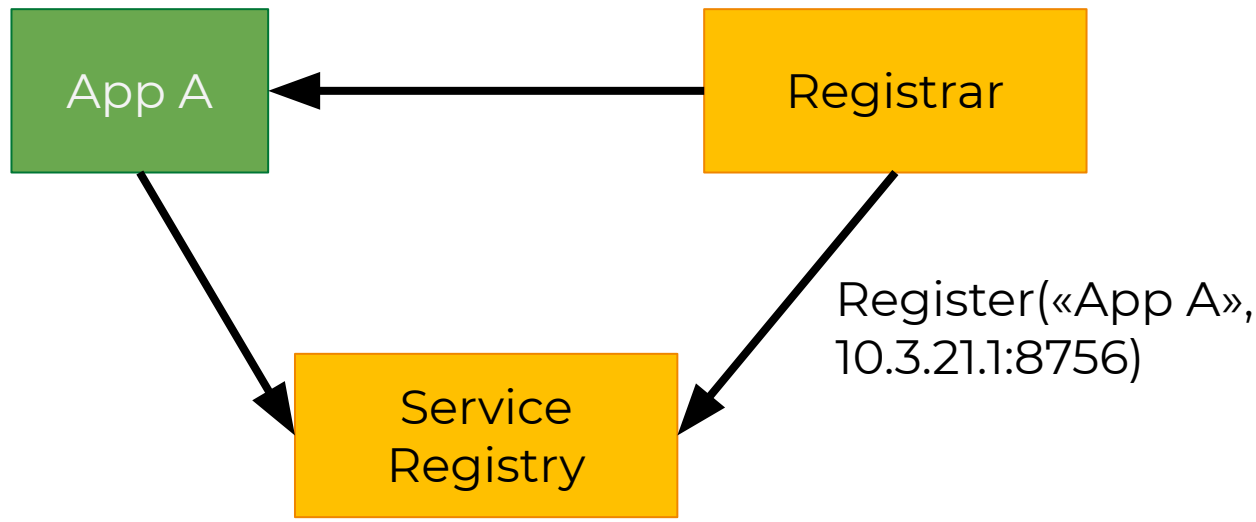
Register(«App A», 10.3.21.1:8756)

Плюсы: простота реализации.

Минусы: жёсткая привязка к Service Registry.

Регистрация сервисов

Регистрация специальным компонентом:



Регистратор постоянно опрашивает сеть на предмет новых сервисов или подписывается на события контроллера сервисов.

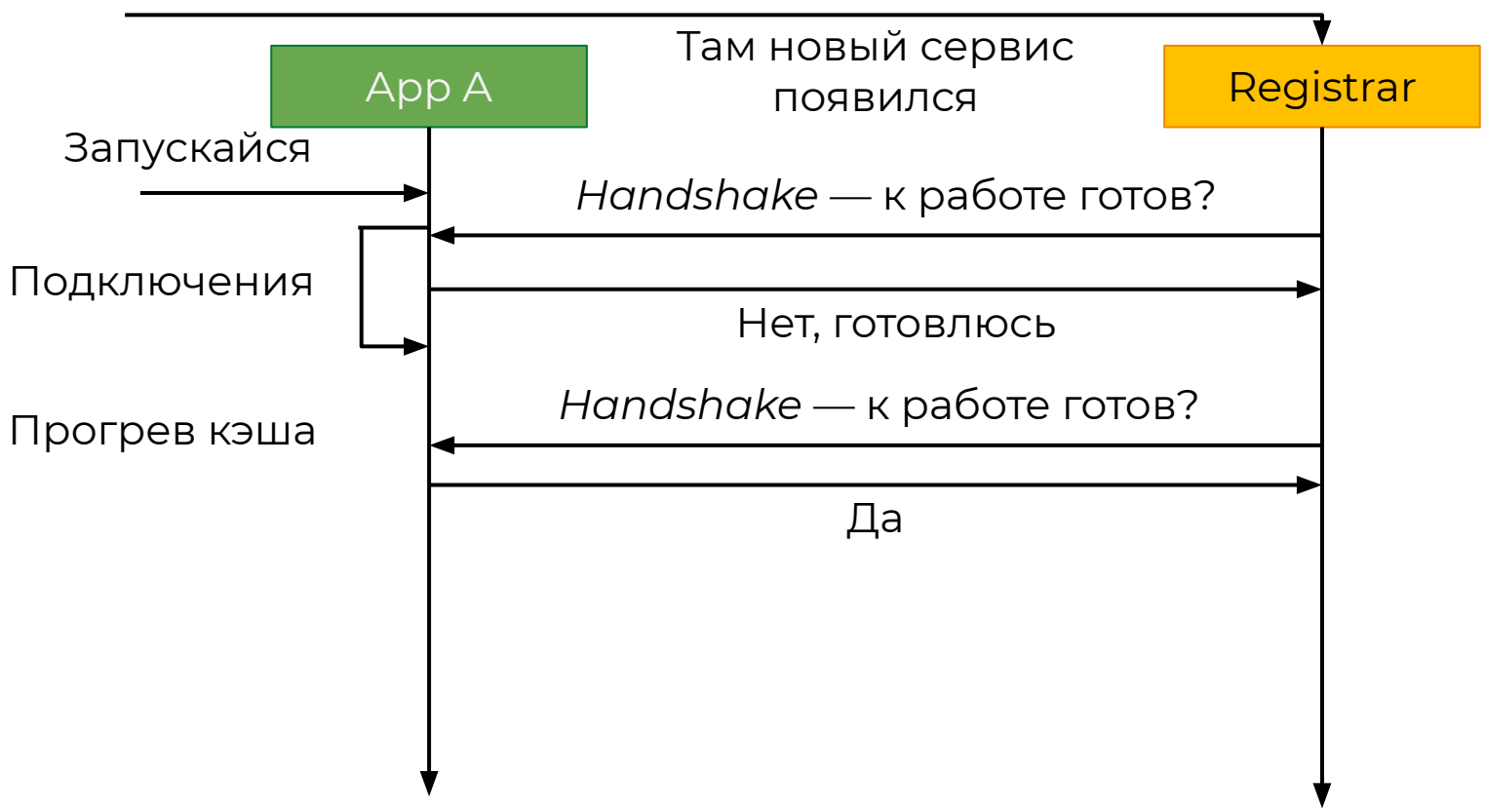
Плюсы: клиент не привязан к Service Registry.

Минусы: дополнительный компонент для поддержки.

Использование Service Discovery

Какие требования предъявляются к сервисам, чтобы можно было использовать Service Registry?

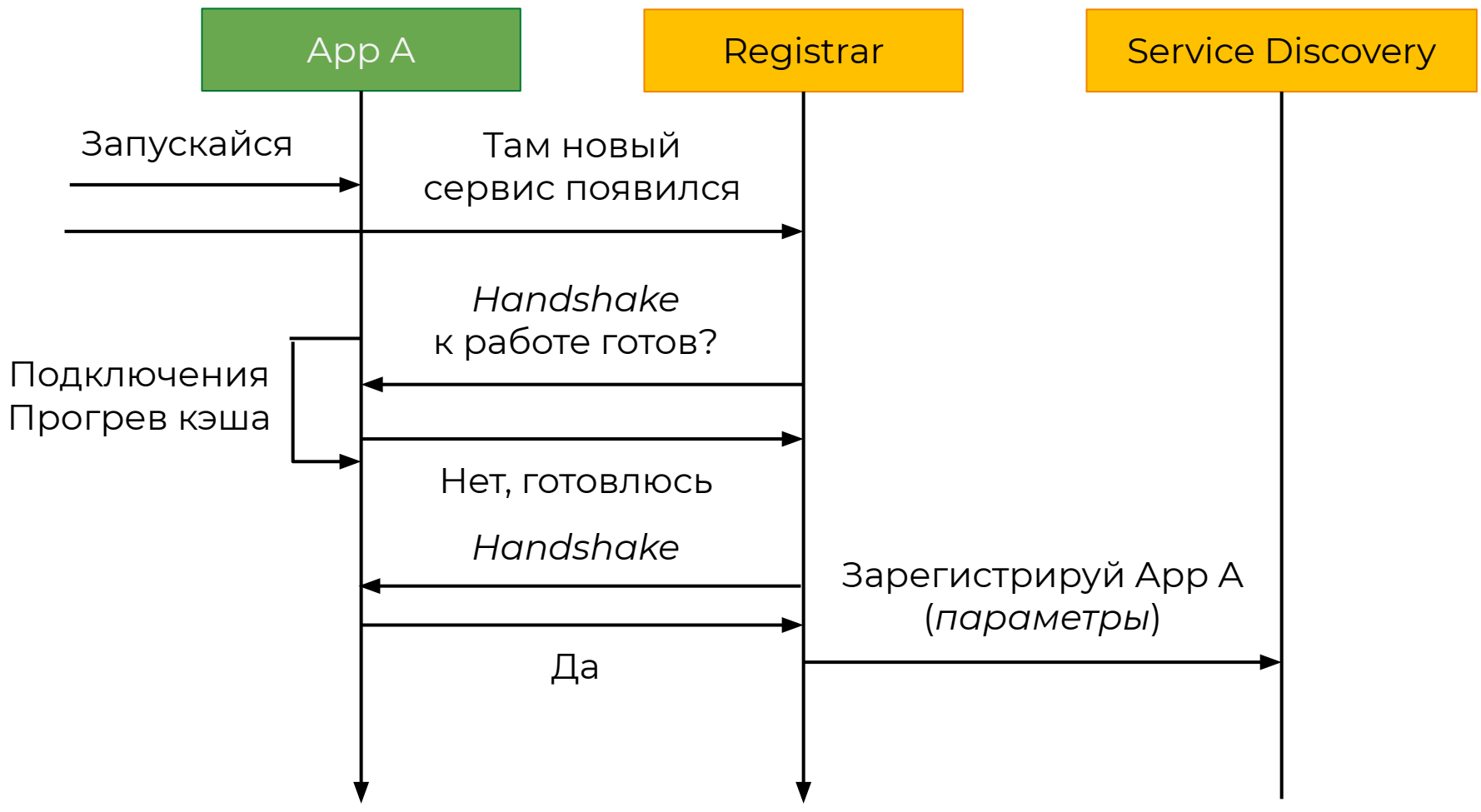
Сервис должен уметь сообщать о своей работоспособности и готовности к работе:



Skillbox

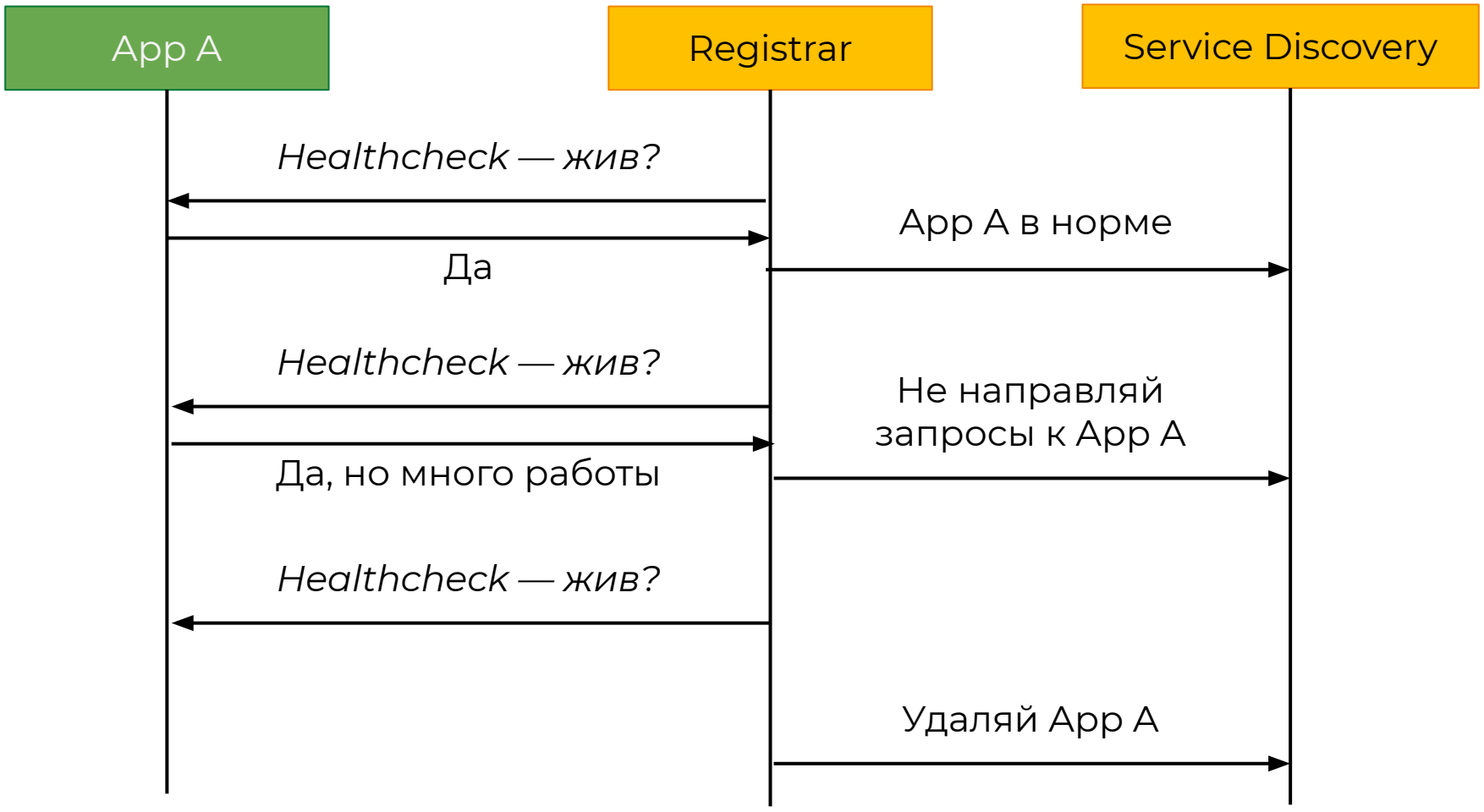
Handshake/Readiness

Позволяет ограничить бизнес-запросы к сервису, пока он стартует:



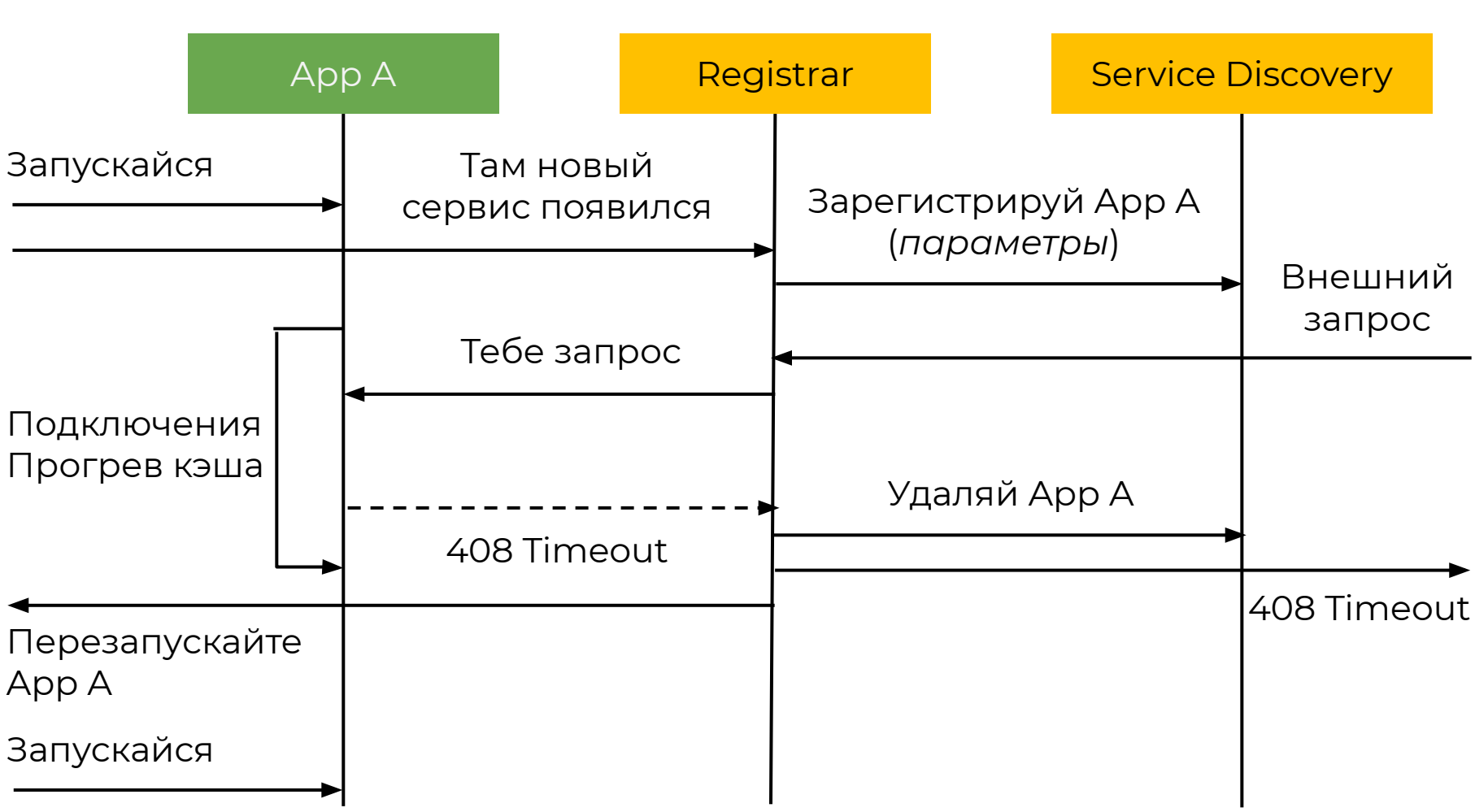
Healthcheck/Liveness

- Маркер, что сервис может отвечать на запросы
- Дополнительная информация об общем состоянии



Проблема инициализации

Сервис может уйти в постоянную перезагрузку, если время инициализации продолжительное:



Выводы

- Использование Service Discovery позволяет динамично добавлять и удалять сервисы из сети автоматически
- Сервисы необходимо подготовить к использованию Service Discovery
- Существует несколько подходов к организации Service Discovery, решение необходимо принимать на основе требований к проекту
- Сервисы должны как минимум отвечать на Healthcheck-запросы. Желательно поддерживать свой вариант Handshake

Skillbox

Спасибо за внимание!