

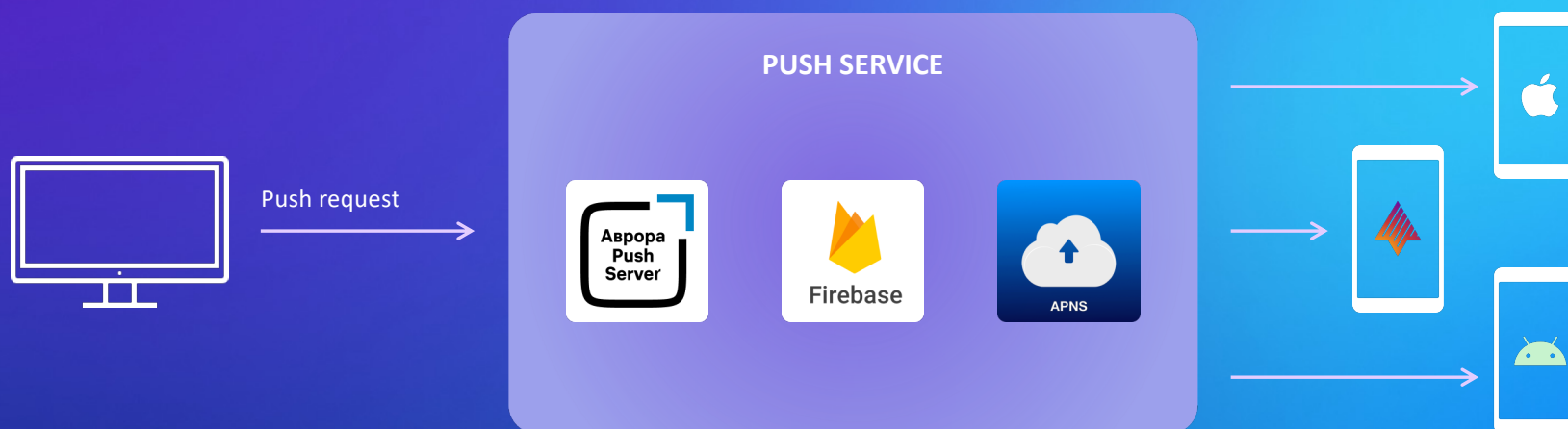
СЕРВИС УВЕДОМЛЕНИЙ АВРОРА

Дмитрий Шевченко, владелец продукта
Камиль Ахметов, инженер-разработчик
Евгений Дворников, инженер-разработчик

План доклада

- 1 О технологии Push
- 2 Процесс взаимодействия с сервисом
- 3 Реализация со стороны ОС Аврора
- 4 Реализация Push сервера
- 5 Ограничения
- 6 Заключение

Push Service в Android / iOS / ОС Аврора



Необходимые компоненты со стороны разработчика:

- › Приложение, поддерживающее Push-уведомления
- › Сервер приложения для общения с Push-сервером

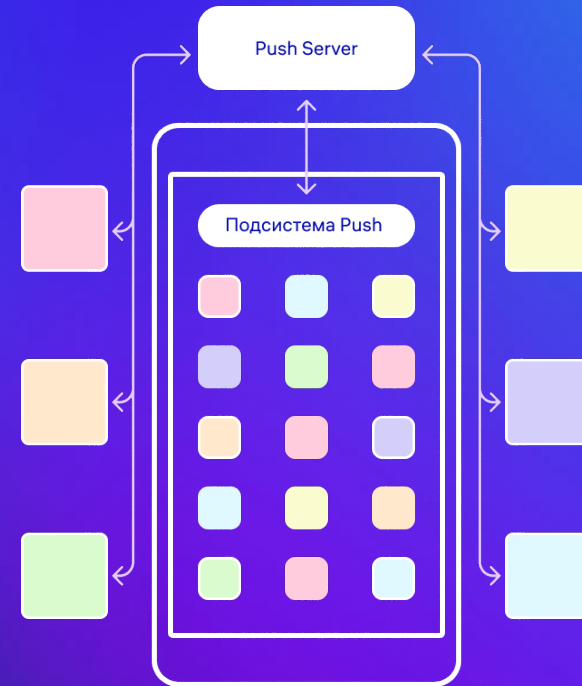
Необходимые компоненты со стороны Экосистемы ОС:

- › Push сервер
- › Подсистема Push-уведомлений в ОС

Преимущества технологии Push



Необходимо постоянное соединение для каждого приложения



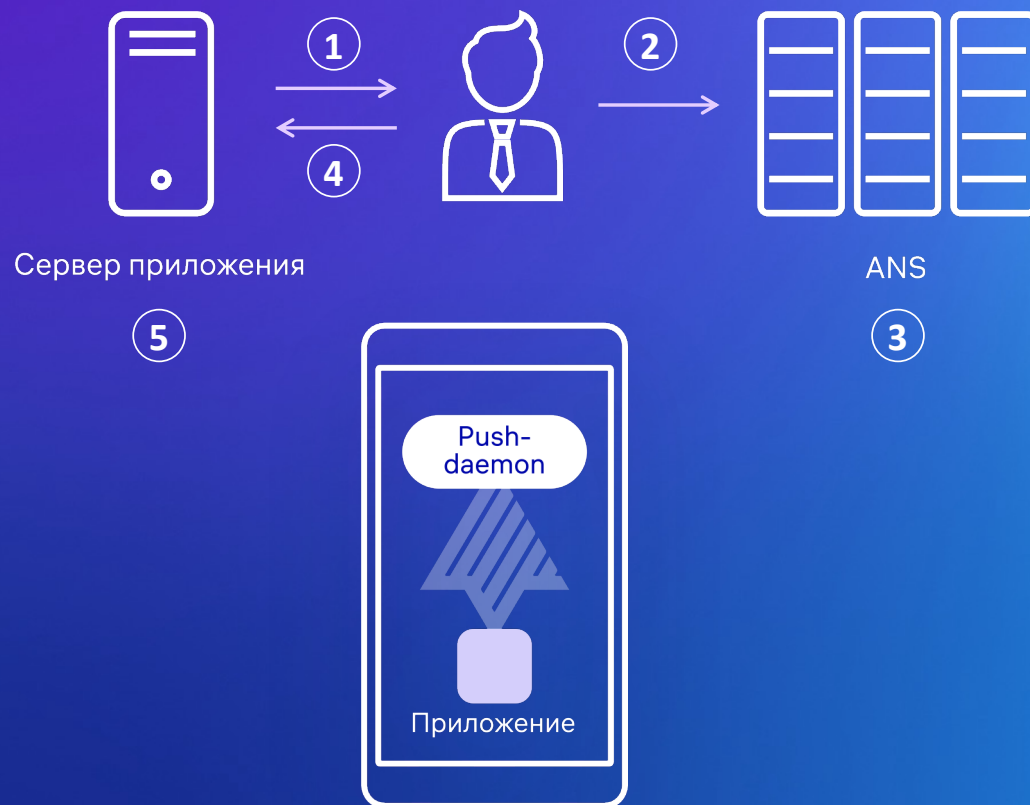
Необходимо только одно постоянное соединение между Push Server и подсистемой Push

Сервис уведомлений Аврора

Функциональные возможности

- › Регистрация разработчика и приложения
- › Установка и проверка соединения между устройством и Push Server'ом
- › Отправка push-уведомлений на активные/неактивные приложения
- › Подсчет количества отправленных push-уведомлений по отдельным проектам
- › Поддержка многопользовательского режима

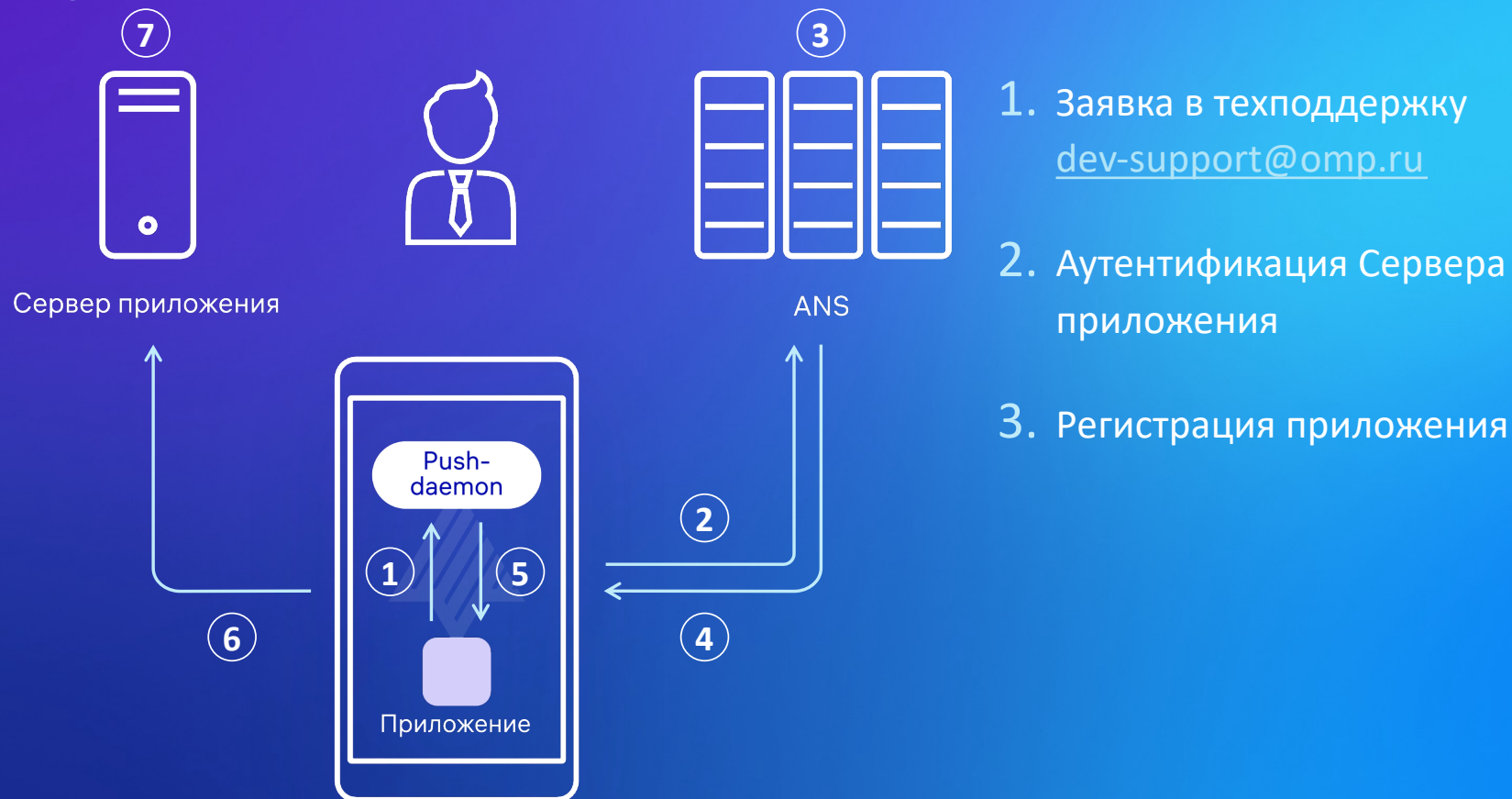
Процесс взаимодействия с ANS



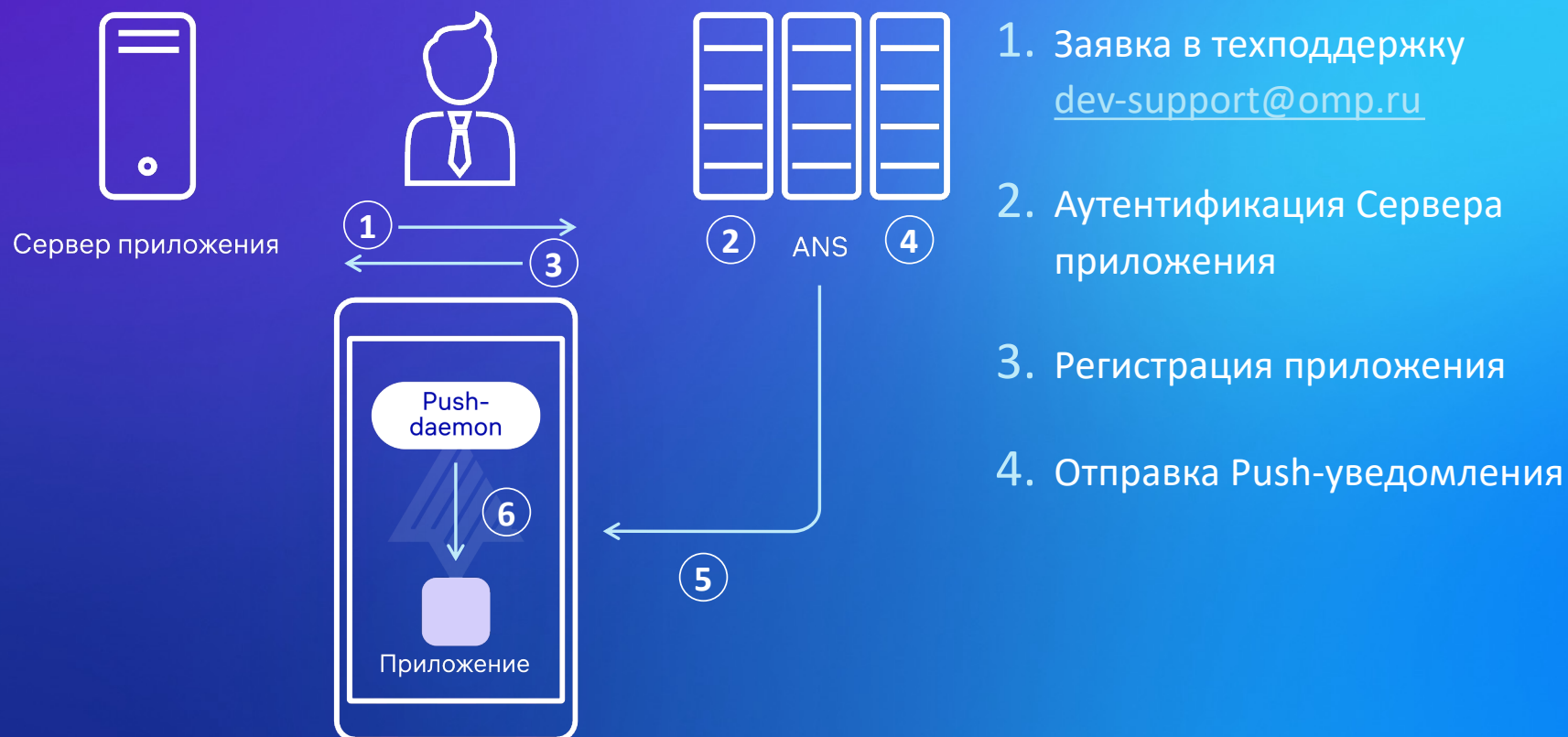
1. Заявка в техподдержку
dev-support@omp.ru

2. Аутентификация Сервера приложения

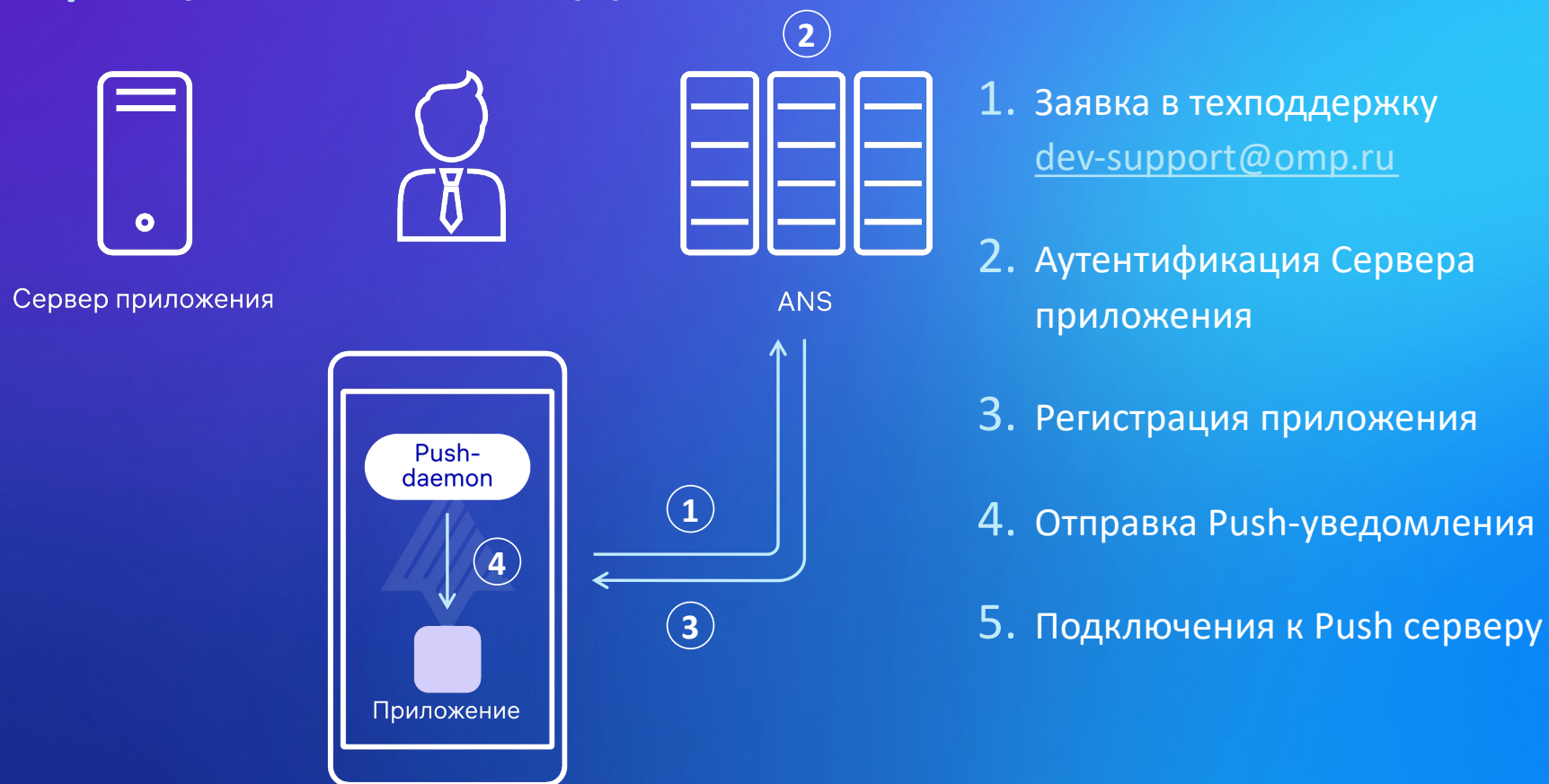
Процесс взаимодействия с ANS



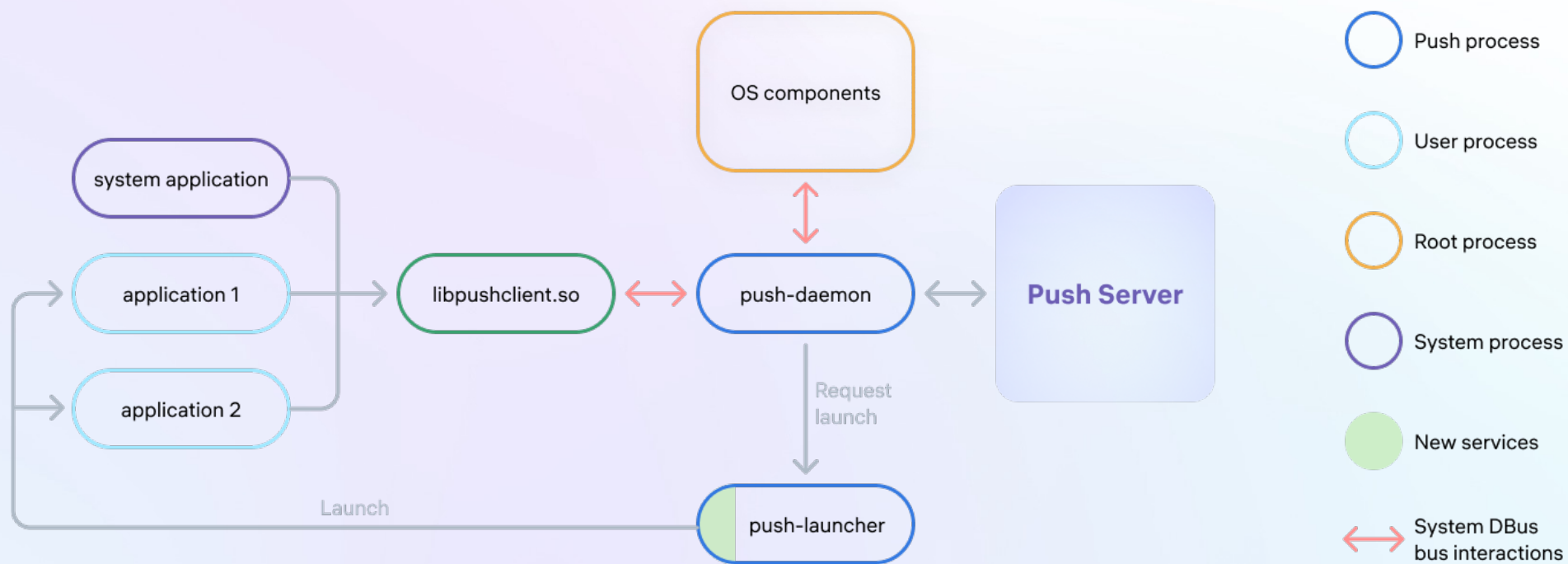
Процесс взаимодействия с ANS



Процесс взаимодействия с ANS



Подсистема Аврора Push

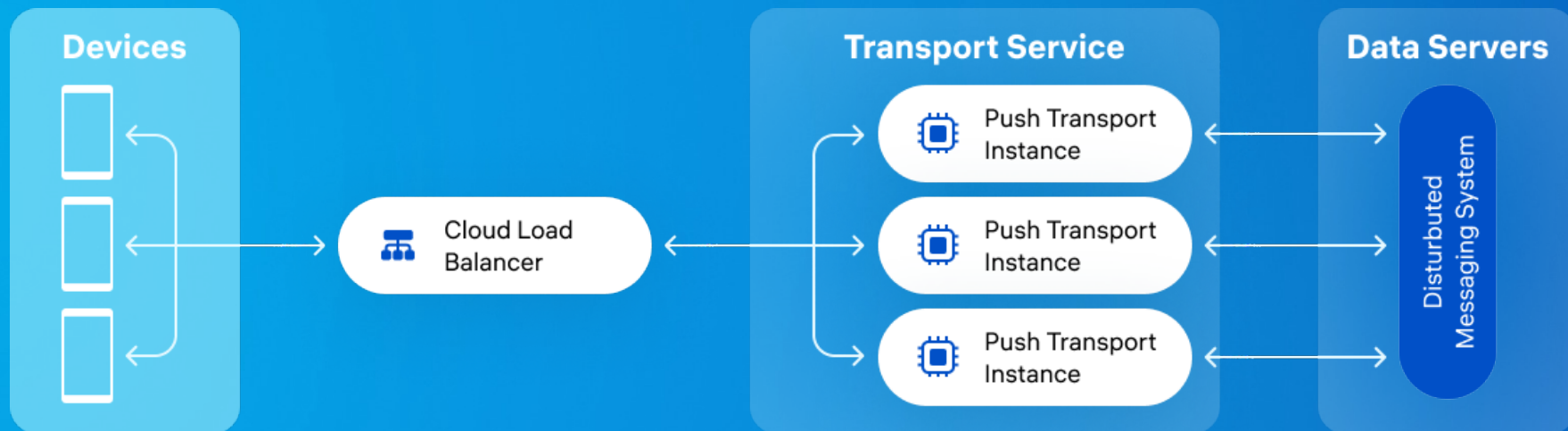


API подсистемы Аврора PUSH

Метод	Описание метода
<code>void setApplicationId(const QString &applicationId);</code>	Установка applicationId в приложении. До начала работы с пушами нужно вызвать эту функцию
<code>QString applicationId();</code>	Получение applicationId, установленного в клиенте.
<code>void registrate();</code>	Отправка запроса о регистрации приложения на Push Server. Функцию следует вызывать при каждом запуске приложения.
<code>bool isPushSystemReady ();</code>	Получение подтверждения о готовности Сервиса уведомлений Аврора к работе. (В версии 1 функция называется <code>getConnectionStatus</code>)
<code>int error()</code>	Получение кода последней ошибки (Новое в версии 2)
<code>QString errorMessage()</code>	Получение текстового описания последней ошибки (Новое в версии 2)
Сигнал	Описание сигнала
<code>void pushSystemReadinessChanged (bool status);</code>	Изменение готовности Сервиса уведомлений Аврора (В версии 1 сигнал называется <code>connectionStatusChanged</code>)
<code>void registrationId(const QString &registrationId);</code>	Получение registrationId
<code>void registrationError(ErrorType error);</code>	Ошибка регистрации
<code>void notifications(const PushList &pushList);</code>	Получение уведомления
<code>void clientInactive();</code>	Приложение длительное время не взаимодействовало с Push-демоном. Если приложение запущено в бэкграунд режиме, его стоит выключить для экономии заряда батареи устройства.

Push сервер

Подключение мобильных устройств

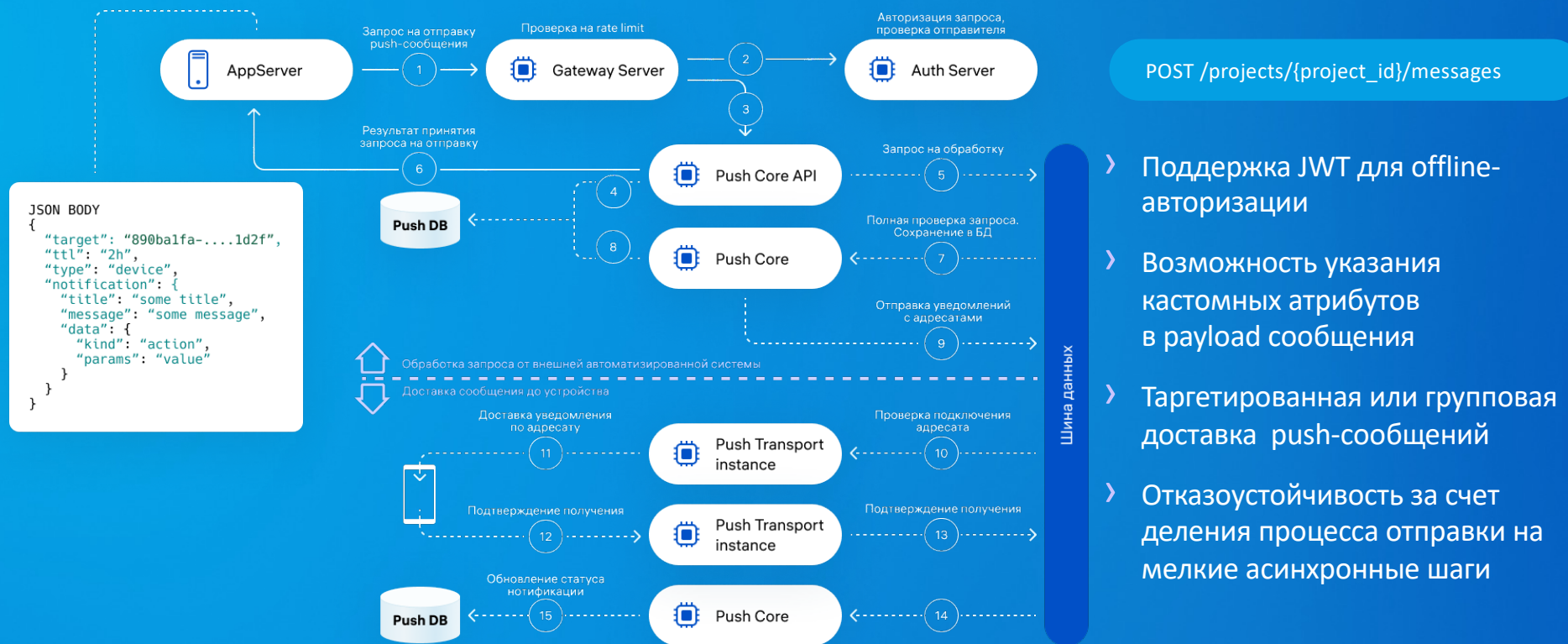


- › Изоляция Transport-сервисов
- › Бинарный протокол для связи с устройствами ради экономии трафика
- › До 500K соединений с каждым экземпляром transport'a

ON-PREMISE AVAILABLE

Push сервер

Отправка и доставка push-сообщения



Заключение

- › Сервис уведомлений Аврора (ANS) уже используется в Аврора Центре и с устройствами с ОС Аврора 3.2.x и 4.0.x
- › Push-стратегия — необходимый паттерн в проектировании современных мобильных приложений
- › Аврора Центр — одно из первых решений, подключивших Сервис уведомлений Аврора (ANS)
- › Для удобства интеграции нашего решения в ваши продукты мы предоставим все необходимые материалы и референсные приложения.

Спасибо!

omp.ru

auroraos.ru

info@omp.ru

