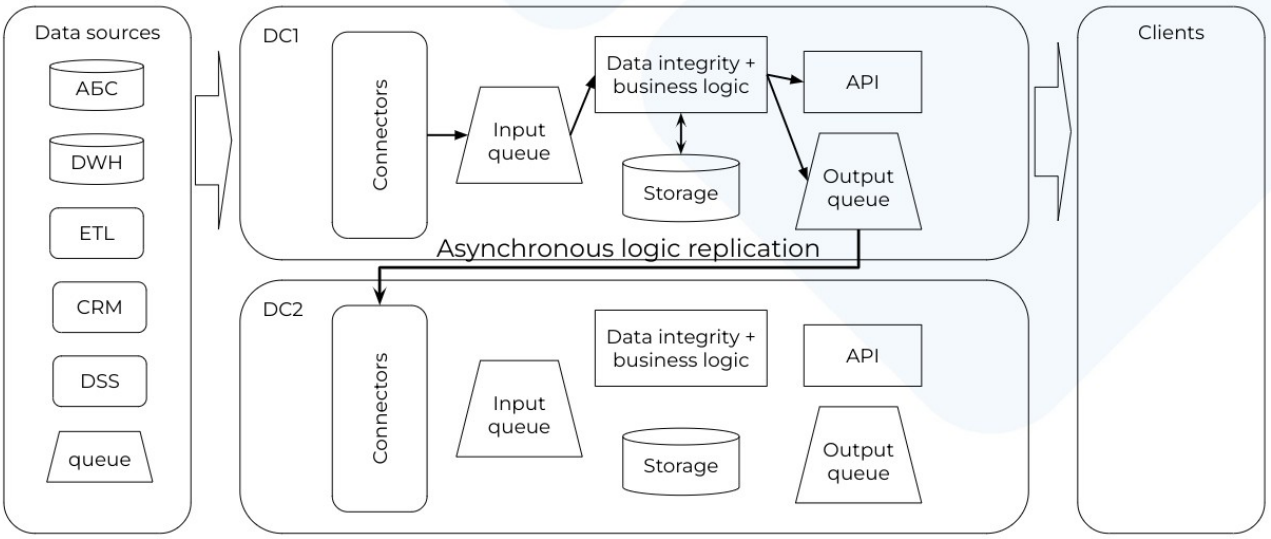
1. Функциональная и компонентная архитектура Picodata

Данный документ описывает основные понятия и компоненты программного обеспечения Picodata, а также их функциональное назначение.

1.1. Функциональная архитектура

Общая функциональная схема Picodata показана на рисунке ниже.



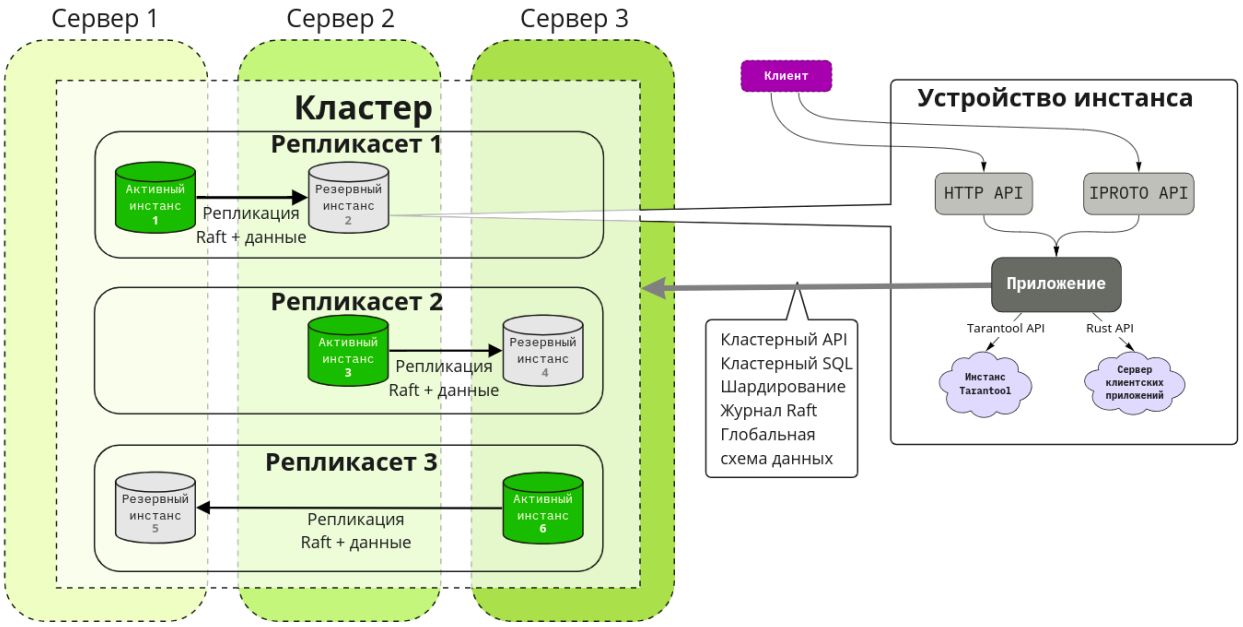
Архитектура решения состоит из следующих элементов:

- **Коннекторы.** Они позволяют осуществлять коммуникацию с внешними системами. Коммуникация может осуществляться как pull-запросами к каким-то API, так и реализацией API, ожидающей запросов извне. Помимо http-api коннекторы могут реализовывать и родные протоколы систем, например kafka. В поставке уже есть коннекторы для многих распространённых систем. Если необходимая система отсутствует в этом списке, то коннектор можно написать самостоятельно и подключить его через конфигурацию.
- **Входящая очередь.** Все сообщения из коннекторов попадают во входящую очередь. В очереди реализованы правила маршрутизации, таким образом, на каждое сообщение из коннекторов можно навесить ключи маршрутизации, и оно обработается согласно этим ключам.
- **Слой бизнес-логики**. Здесь проверяется соответствие схемы пришедших данных той модели, которая заданной в конфигурации. После этого данные приводятся к последней версии модели данных, проходят через функции-обработчики, согласно ключам маршрутизации, и сохраняются, если это предусмотрено конфигурацией, в хранилище.
- **Хранилище**. Здесь хранятся данные.
- **Выходящая очередь**. Сюда попадают задачи, которые должны быть отправлены push-методом в какие-то внешние системы.
- **API** – интерфейс для доступа к данным. Данные можно получить как через язык запросов graphql, так и через restful-сервисы. Также можно написать свои собственные сервисы с любым API и подключить их через документацию.

Через выходящую очередь можно построить асинхронную логическую репликацию и использовать её, в том числе, как репликацию между ЦОД. Логическая репликация позволяет сделать более гибкими схемы репликации данных, чем физическая репликация базы данных, тем самым её проще эксплуатировать и она может помочь сэкономить на избыточности данных.

1.2. Компонентная архитектура

Общая компонентная схема Picodata показана на рисунке ниже.



На схеме видно, что каждый из серверов содержит два инстанса Picodata, принадлежащих разным репликасетам. Одновременно с этим, все активные инстансы находятся на разных серверах. В правой части схему показано условное внутреннее устройство инстанса: на уровне каждого из них запускается отдельный экземпляр СУБД Tarantool. Для хранимых в нем данных поддерживается сервис клиентских (сторонних) приложений на языке Rust. Это позволяет создавать более сложную и оптимизированную бизнес-логику, при которых на инстансе Picodata возможно исполнение нужного кода (например, для обработки или перенаправления данных).
В условиях потери сетевой связности между компонентами системы Picodata ставит доступность конфигурации превыше консистентности, в терминах теоремы CAP — режим AP. Данная особенность является одним из ключевых преимуществ Picodata: возможности продолжать работать с кластером без потери данных даже при недоступности части узлов. Эта функция полагается на способности алгоритма Raft поддерживать кворум из доступных узлов по схеме n/2+1 с округлением в меньшую сторону. Пользователь имеет возможность управлять кластером пока выполняется условие доступности большинства реплик провайдера конфигурации (2 реплики из 3, или 3 из 5).