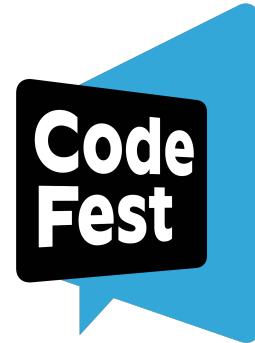


май, 2024



От самой большой нешардированной инсталляции **PostgreSQL** в Европе

к платформе управления десятками тысяч баз данных



Владимир Алёшин

Руководитель разработки
платформы DBaaS в Авито

О чём доклад?



3

1. Предпосылки создания платформы DBaaS.
2. Паттерны эксплуатации баз данных в k8s.
3. Паттерны эксплуатации баз данных в мультиклUSTERной среде.

DBaaS — это



DBaaS – это

- Базы данных как услуга.

DBaaS — это

- Базы данных как услуга.
- Полная автоматизация жизненного цикла базы данных.

DBaaS — это

- Базы данных как услуга.
- Полная автоматизация жизненного цикла базы данных.
- Снятие паразитной когнитивной нагрузки по работе с базами данных для продуктовых разработчиков.

DBaaS – это

Несколько
дата-центров

по 2-3 Kubernetes
кластера в каждом.

DBaaS – это

Несколько
дата-центров

10 различных
технологий
СУБД

по 2-3 Kubernetes
кластера в каждом

4 полностью
автоматизированные:
Redis, PostgreSQL,
Mongodb и
CockroachDB.

6 технологий с частичной
автоматизацией:
Clickhouse, Kafka,
Elasticsearch, Pulsar,
ZooKeeper, BookKeeper.

DBaaS – это

Несколько
дата-центров

10 различных
технологий
СУБД

> 4 тыс. баз
данных
разных типов

по 2-3 Kubernetes
кластера в каждом.

4 полностью
автоматизированные:
Redis, PostgreSQL,
MongoDB и
CockroachDB.

6 технологий с частичной
автоматизацией:
Clickhouse, Kafka,
Elasticsearch, Pulsar,
ZooKeeper, BookKeeper.

> 20 тыс. реплик.

DBaaS — это

Несколько
дата-центров

10 различных
технологий
СУБД

> 4 тыс. баз
данных
разных типов

> 100
операций
создания баз
данных в день

по 2-3 Kubernetes
кластера в каждом.

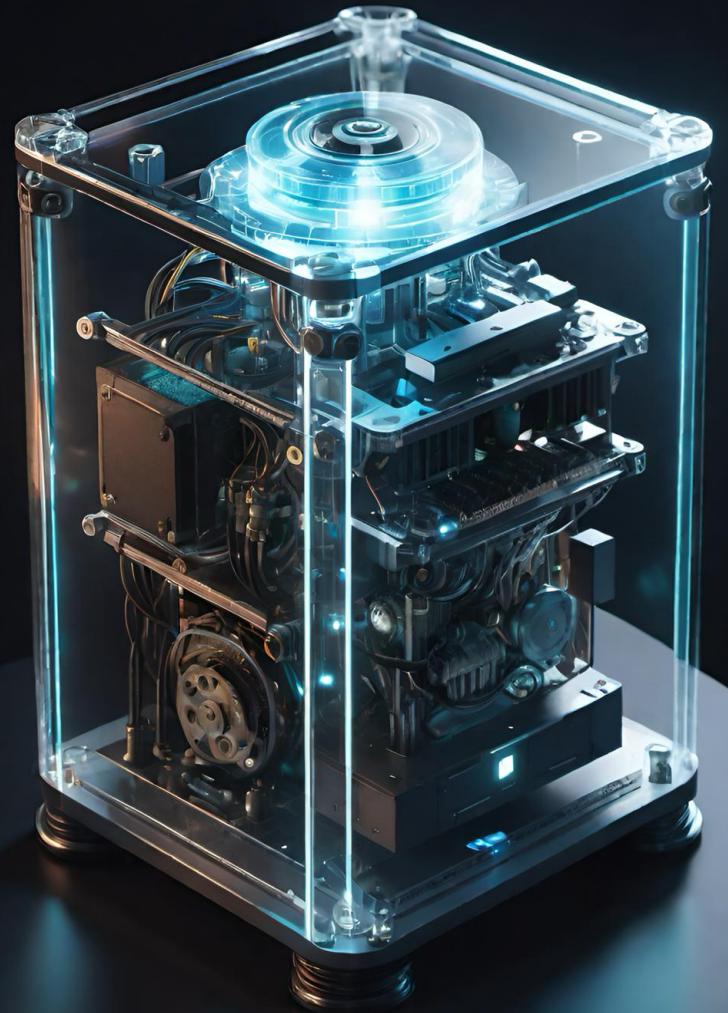
4 полностью
автоматизированные:
Redis, PostgreSQL,
Mongodb и
CockroachDB.

6 технологий с частичной
автоматизацией:
Clickhouse, Kafka,
Elasticsearch, Pulsar,
ZooKeeper, BookKeeper.

> 20 тыс. реплик.

централизованная
система обслуживает
окружения: staging,
performance и
production.

Не бьётся с общим
количеством, потому что
есть «временные» базы
данных.



Предпосылки создания платформы DBaaS

Переход Авито к микросервисной Архитектуре



Переход Авито к микросервисной Архитектуре

- В 2018 году Авито начал массовый переход от монолита к микросервисам.

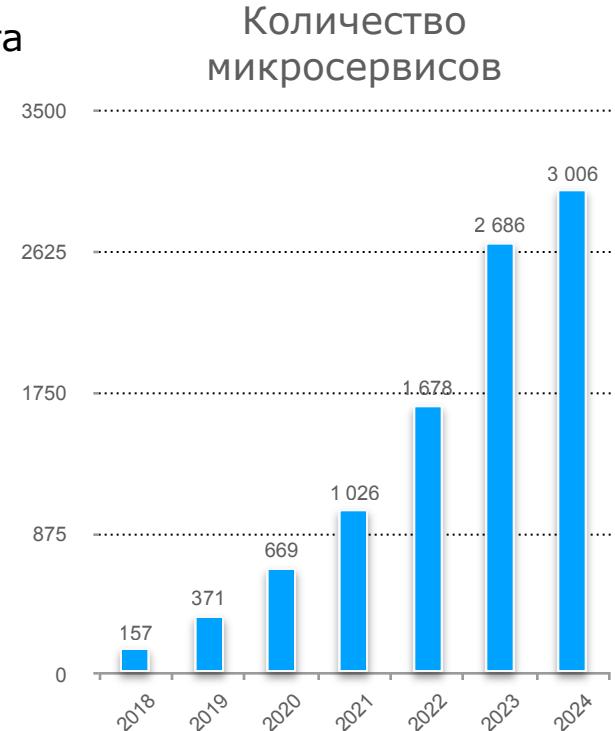


Переход Авто к микросервисной Архитектуре



15

- В 2018 году Авто начал массовый переход от монолита к микросервисам.
- Микросервисы не могут использовать общую базу данных (database per service).



Переход Авито к микросервисной Архитектуре



16

- В 2018 году Авито начал массовый переход от монолита к микросервисам.
- Микросервисы не могут использовать общую базу данных (database per service).
- Микросервисы могут использовать разные СУБД для решения разных задач (polyglot persistence).



Переход Авито к микросервисной Архитектуре

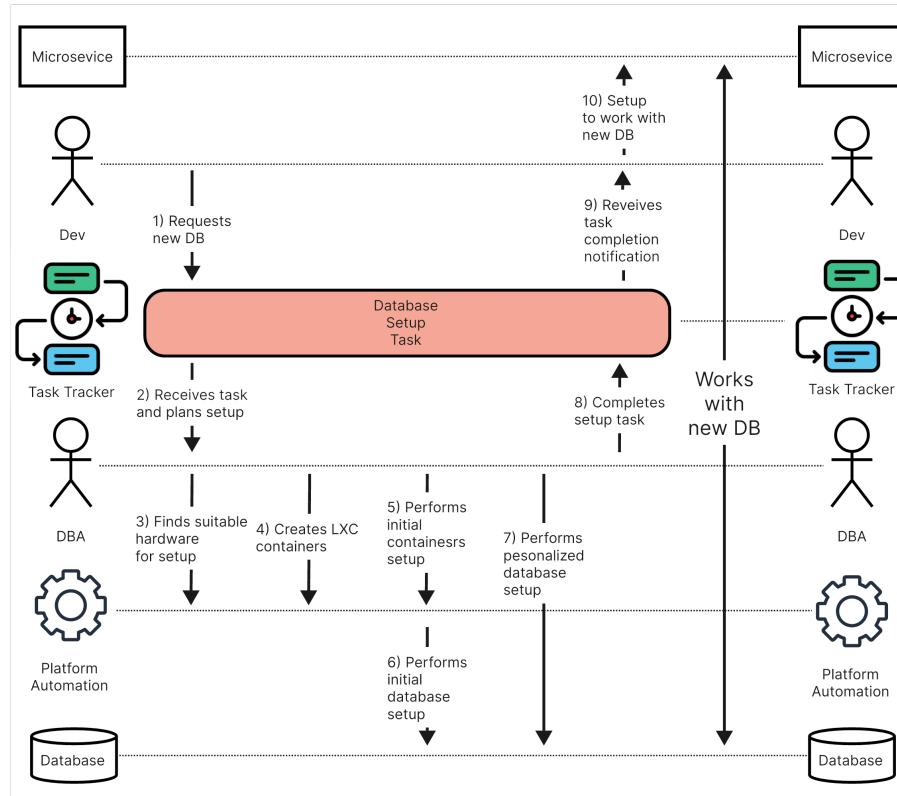


17

- В 2018 году Авито начал массовый переход от монолита к микросервисам.
- Микросервисы не могут использовать общую базу данных (database per service).
- Микросервисы могут использовать разные СУБД для решения разных задач (polyglot persistence).
- К 2021 году команда DBA перестала успевать обслуживать возросший рост спроса на развертывание новых баз данных и эксплуатацию уже развёрнутых.



Создание базы данных за 3-7 дней



Остальное, но не менее важное



Остальное, но не менее важное

- Отсутствие единого реестра баз данных.

Остальное, но не менее важное

- Отсутствие единого реестра баз данных.
- Полуавтоматическое развёртывание новых баз данных, необходимость внесения изменений в систему контроля версий.

Остальное, но не менее важное



22

- Отсутствие единого реестра баз данных.
- Полуавтоматическое развертывание новых баз данных, необходимость внесения изменений в систему контроля версий.
- Сложное и дорогое управление и планирование на уровне bare-metal.

Остальное, но не менее важное



23

- Отсутствие единого реестра баз данных.
- Полуавтоматическое развертывание новых баз данных, необходимость внесения изменений в систему контроля версий.
- Сложное и дорогое управление и планирование на уровне bare-metal.
- Гетерогенная инфраструктура в зависимости от окружения (паразитная когнитивная нагрузка).



Паттерны эксплуатации баз данных в k8s

Путь долг и тернист

Эстакада
Хуанцюэвань

*Самая сложная
развязка в мире*



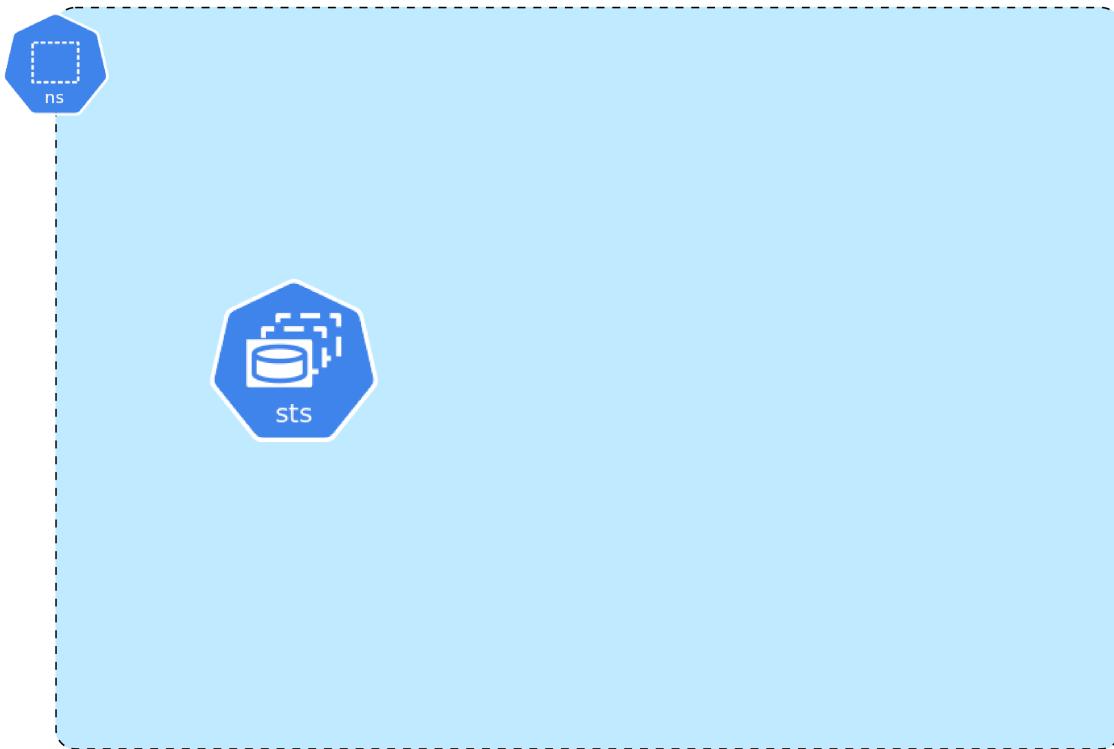
Проще некуда



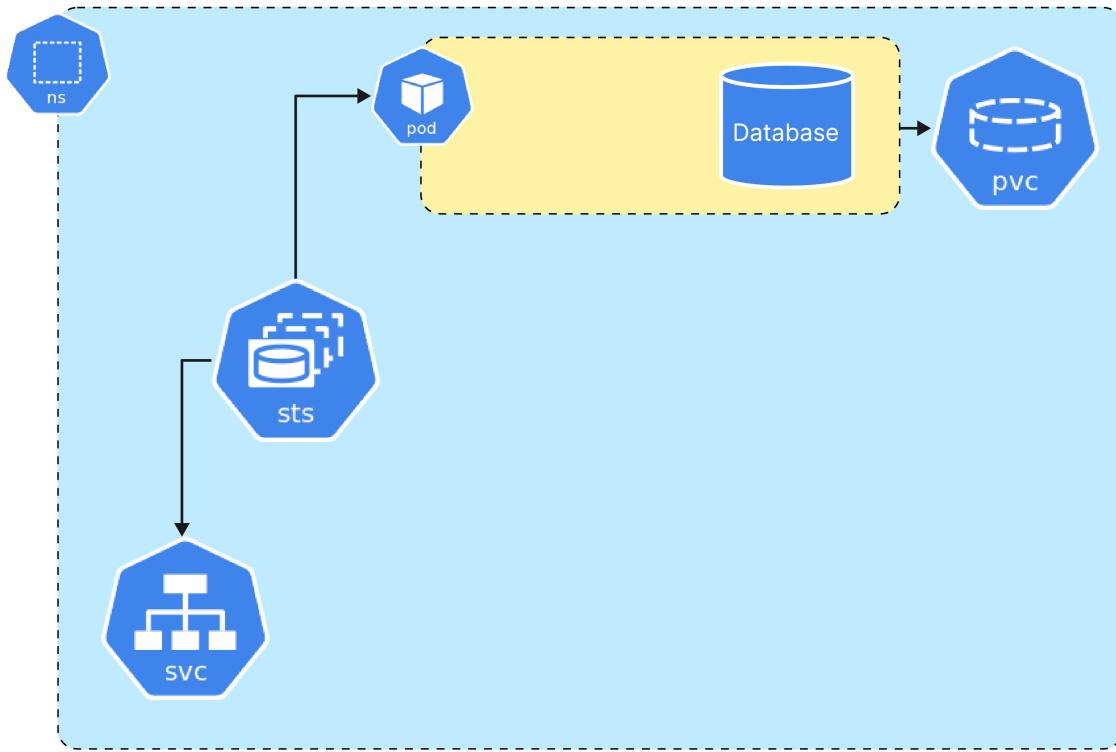
26



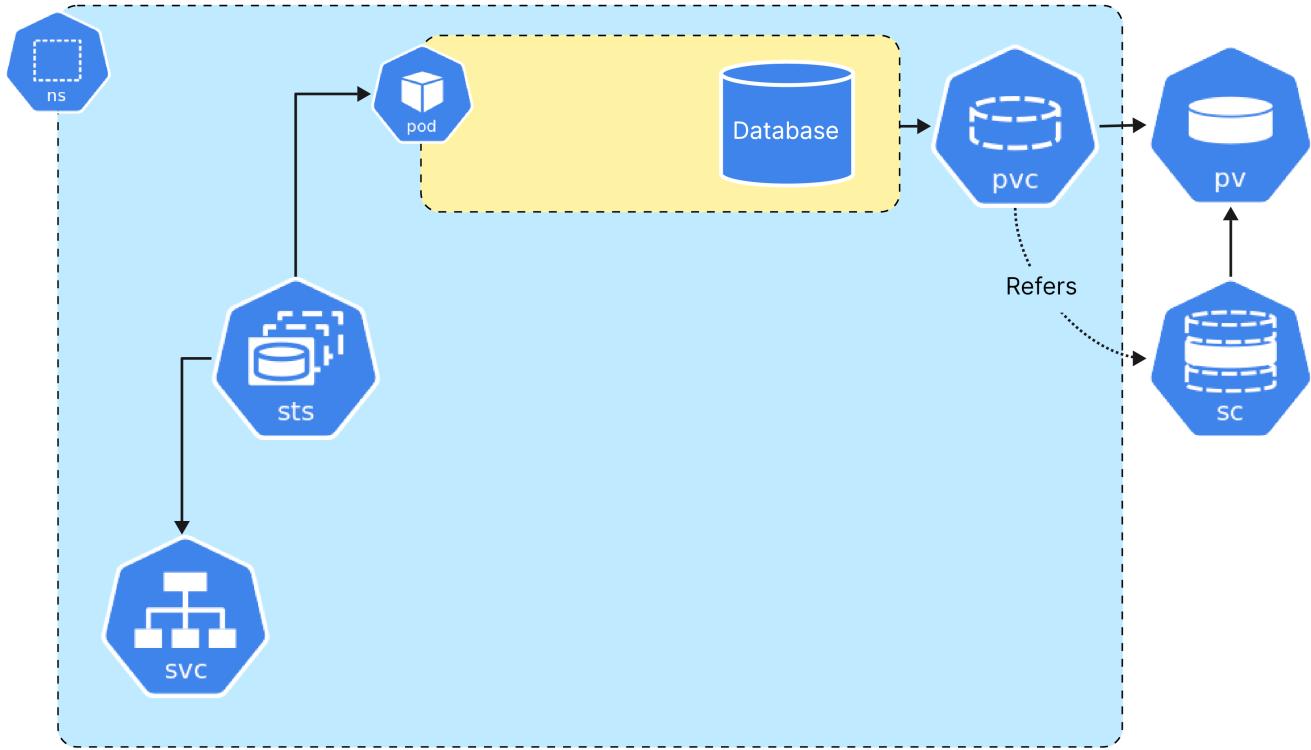
Проще некуда



Проще некуда



Проще некуда



С высокой доступностью

StatefulSet это не только лишь Pod + PersistentVolume, но и:

- Стабильные DNS-имена подов.
- Стабильный алгоритм RollingUpdate.
- Стабильный алгоритм (down)scale.

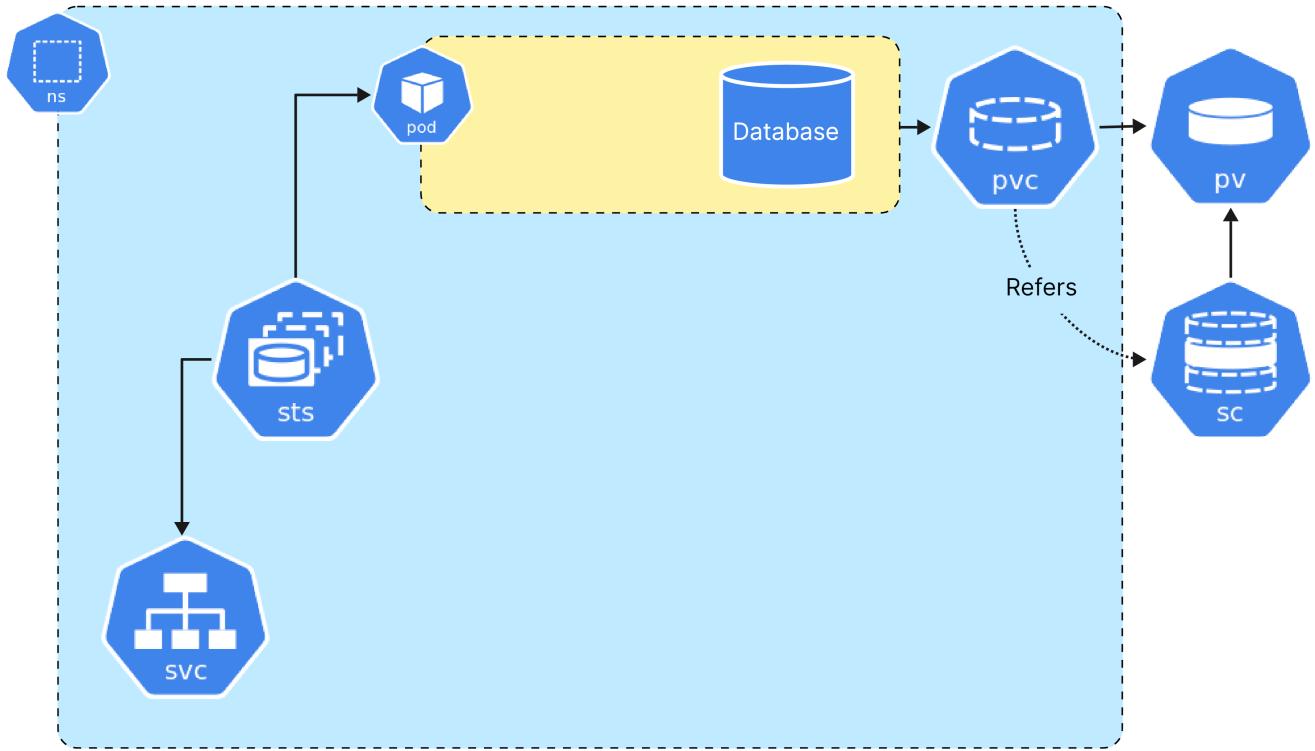
С высокой доступностью

StatefulSet это не только лишь Pod + PersistentVolume, но и:

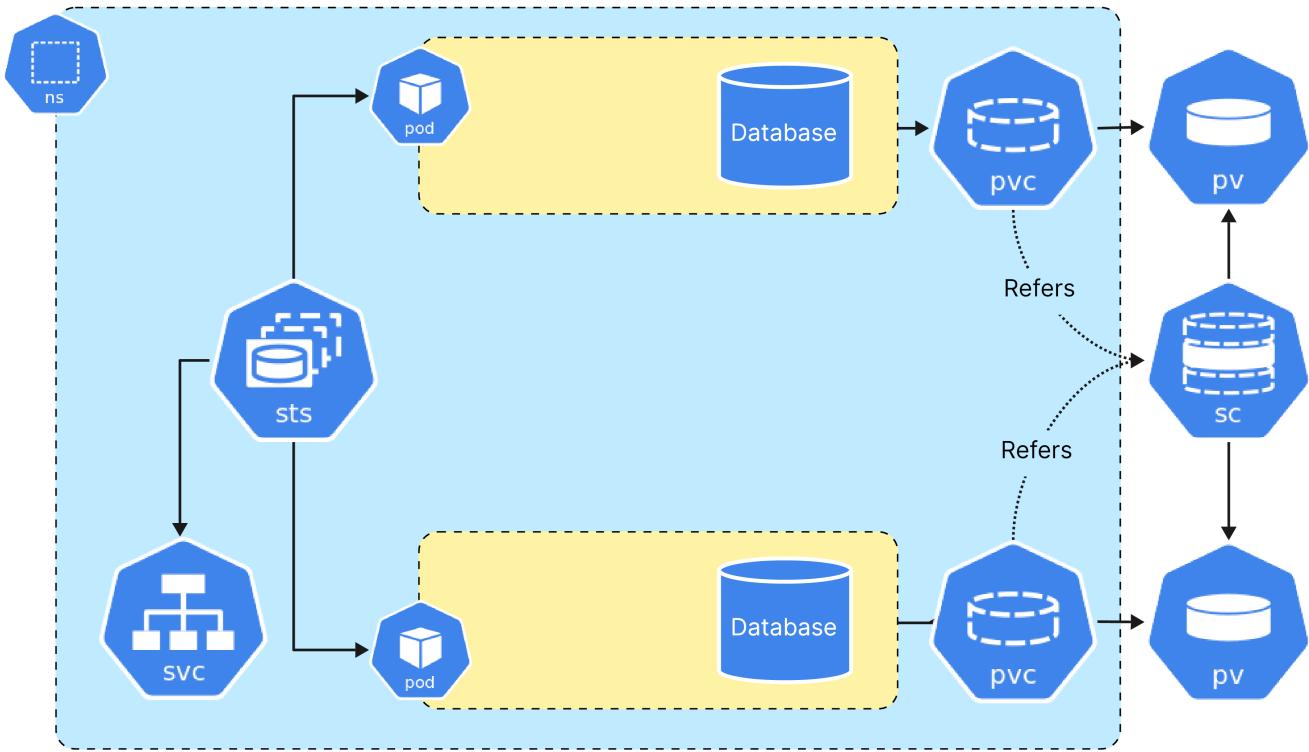
- Стабильные DNS-имена подов.
- Стабильный алгоритм RollingUpdate.
- Стабильный алгоритм (down)scale.

Базовый минимум для реализации репликации.

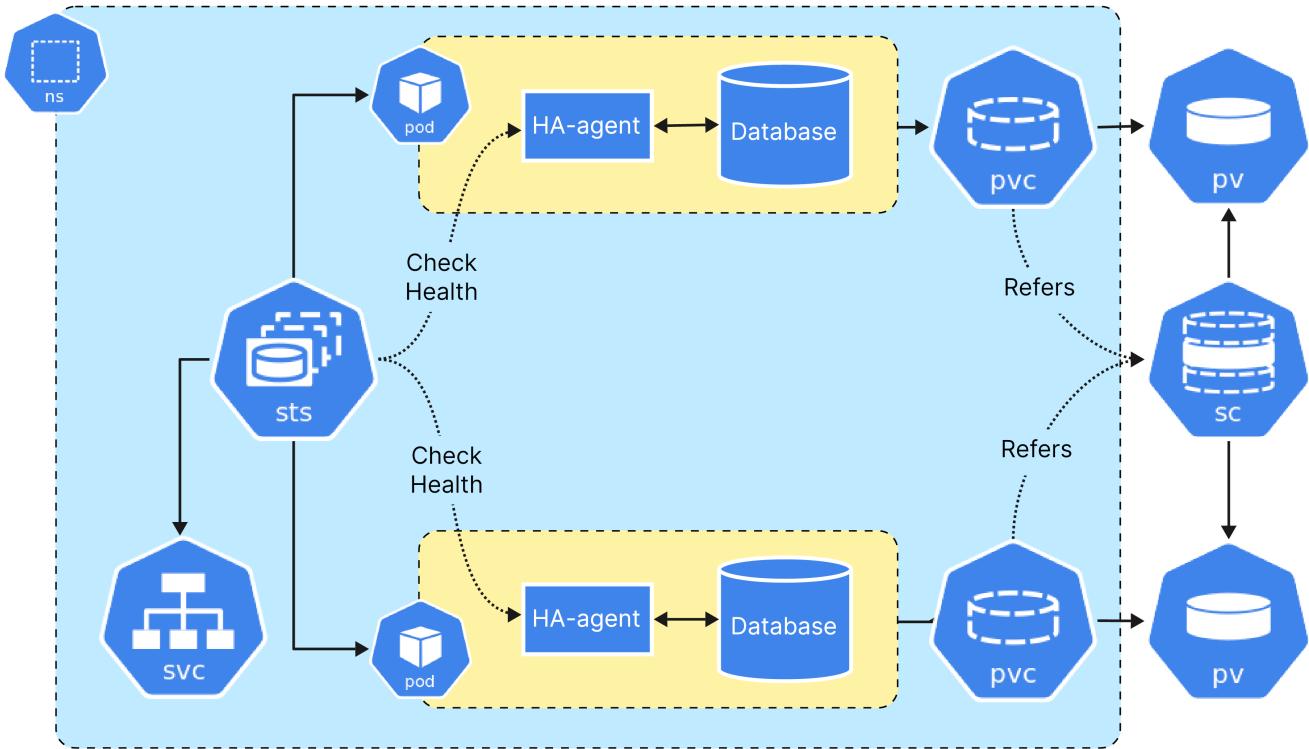
Включаем репликацию



Включаем репликацию



Делегируем НА отдельному компоненту



Паттерн #1

Используйте стандартные механизмы Kubernetes в связке с отдельным компонентом для обеспечения High Availability.

Изменения с минимальным простоем



Изменения с минимальным простоем



37

- В наивной реализации изменение любого параметра приводит к пересозданию pod (изменение манифестов).

Изменения с минимальным простоем



38

- В наивной реализации изменение любого параметра приводит к пересозданию pod (изменение манифестов).
- Современные СУБД умеют изменять часть своей конфигурации без остановки.

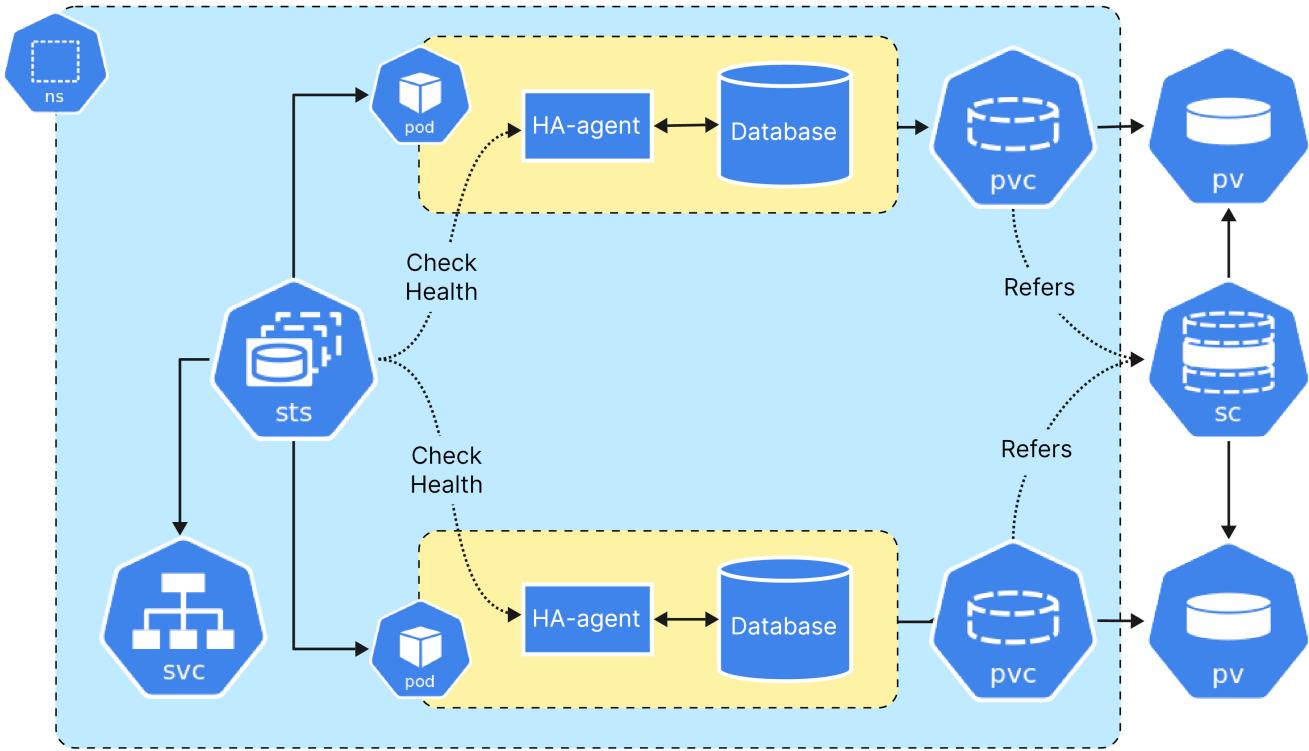
Изменения с минимальным простоем



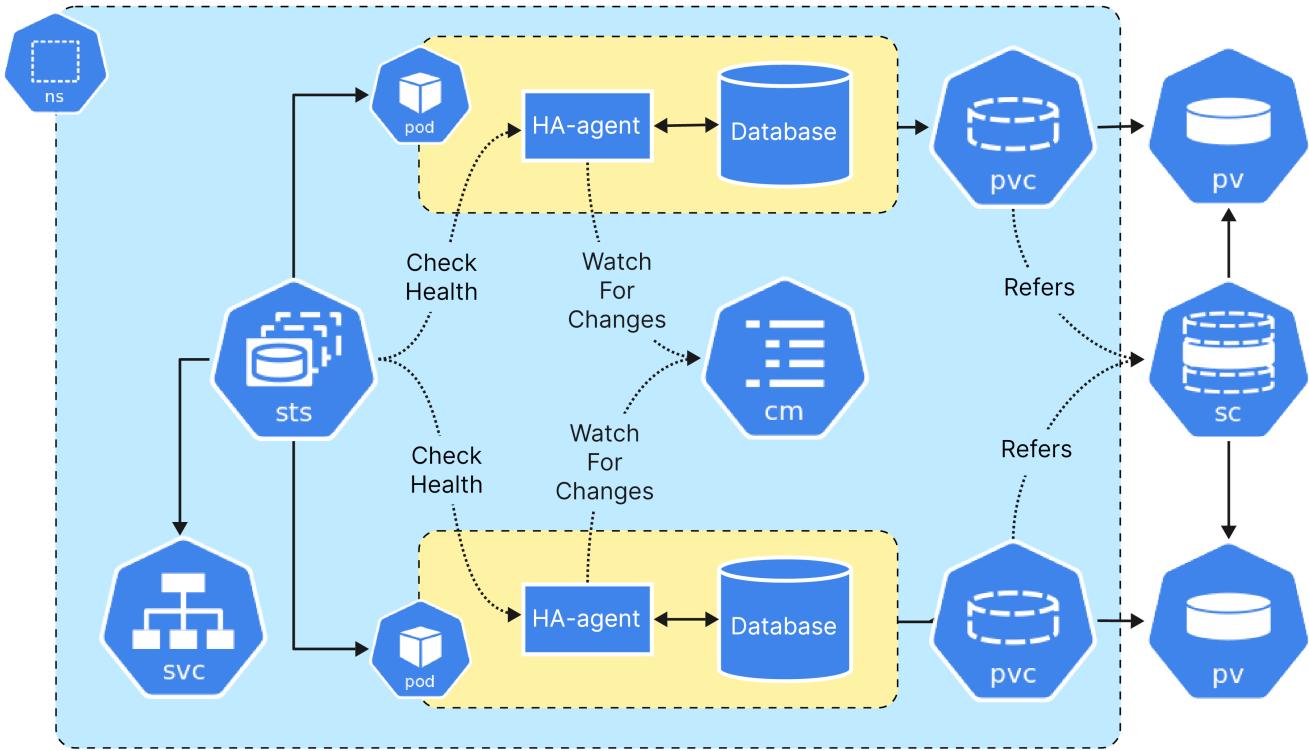
39

- В наивной реализации изменение любого параметра приводит к пересозданию pod (изменение манифестов).
- Современные СУБД умеют изменять часть своей конфигурации без остановки.
- Изменения содержимого ConfigMap не инициирует пересоздание pod.

И вновь HA-agent!



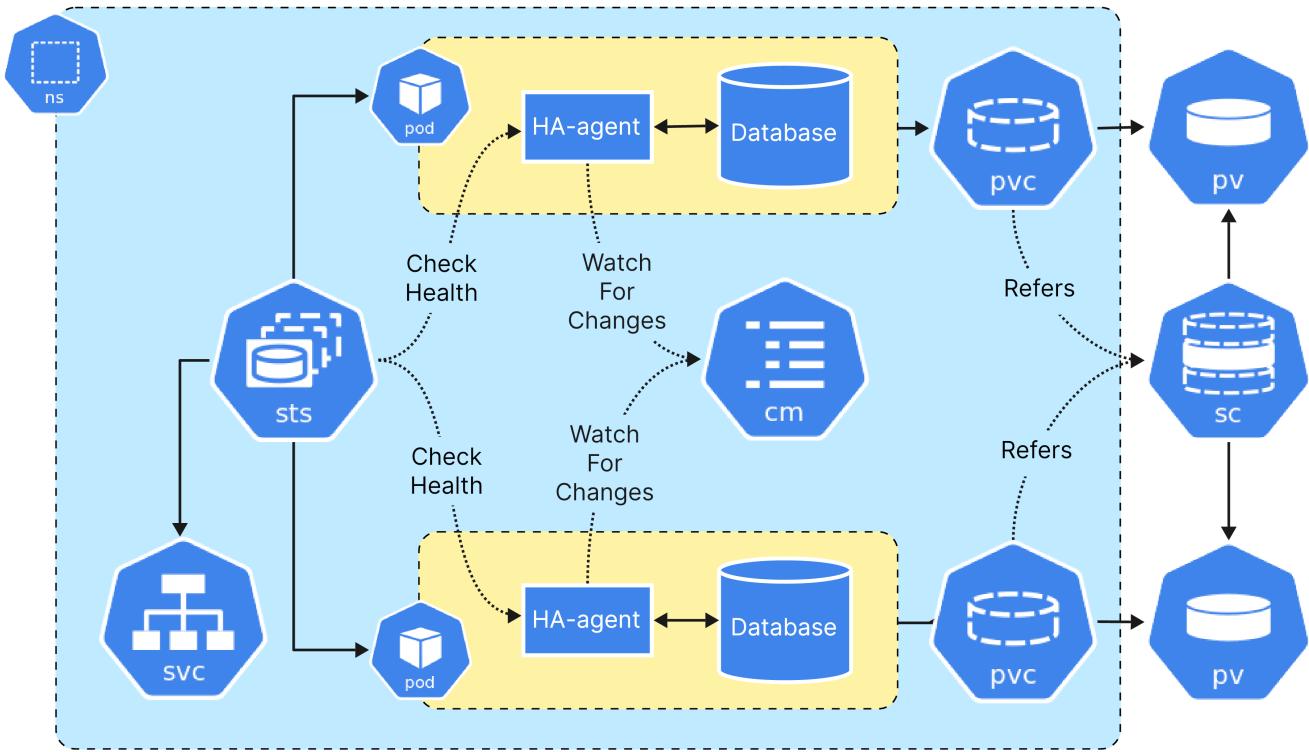
И вновь HA-agent!



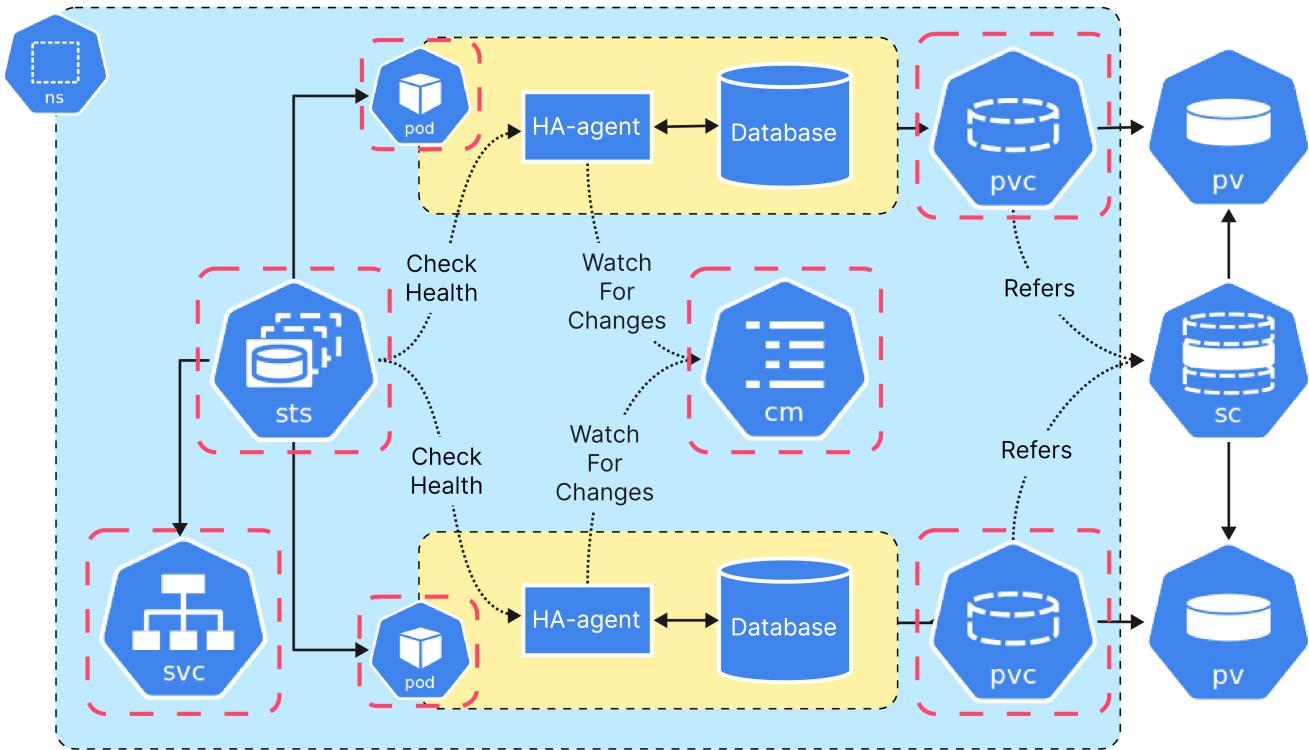
Паттерн #2

Минимизируйте простой в сценариях изменения конфигурации СУБД за счёт использования ConfigMap (в качестве шаблона конфигурации) и HA-agent (в качестве шаблонизатора и компонента, применяющего конфигурацию).

Зоопарк примитивов Kubernetes



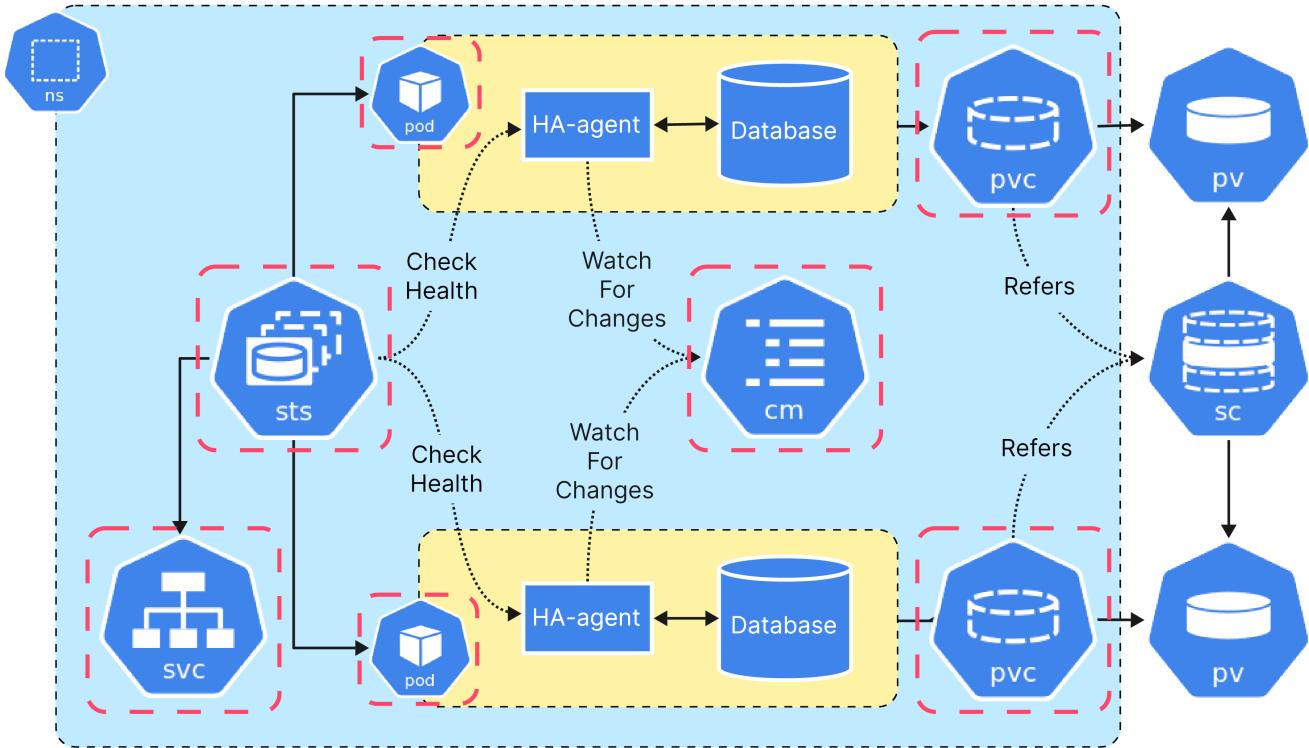
Зоопарк примитивов Kubernetes



Зоопарк примитивов Kubernetes



Очень внимательный
инженер DBaaS



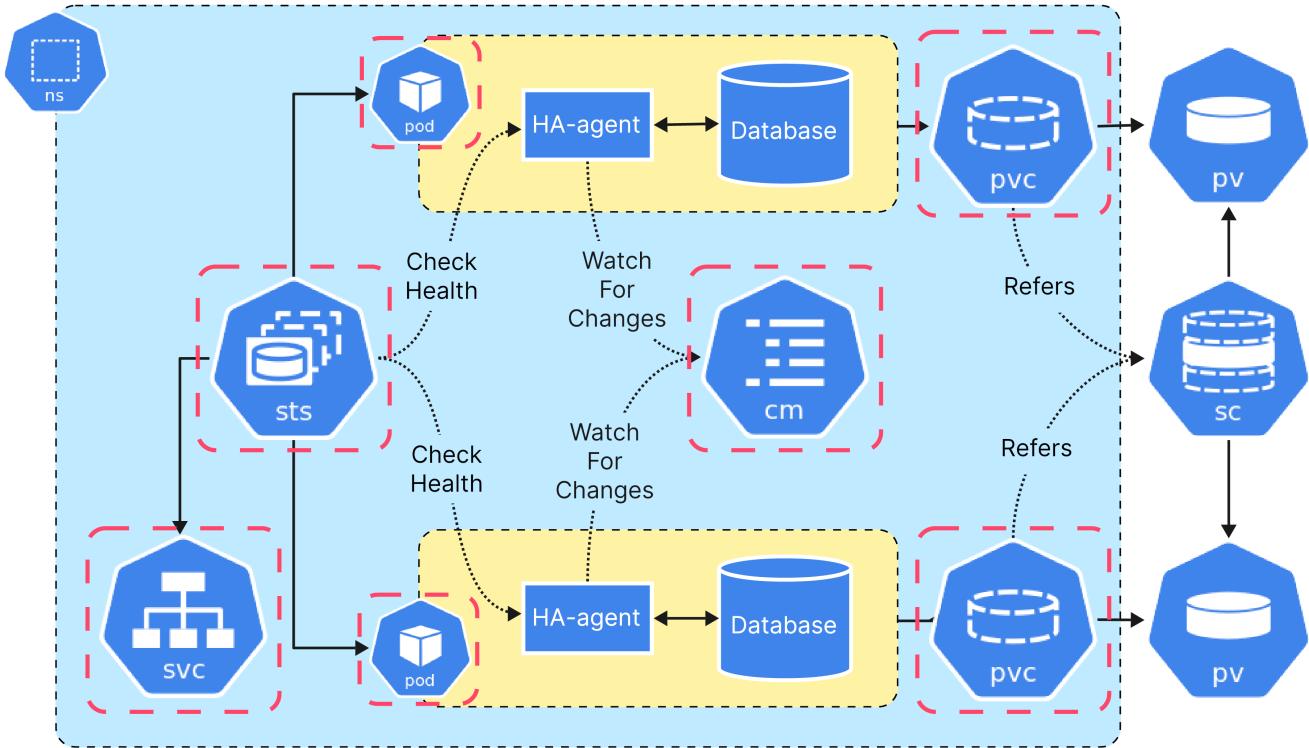
Нужно больше абстракций

Operators are software extensions to Kubernetes that make use of [custom resources](#) to manage applications and their components. Operators follow Kubernetes principles, notably the [control loop](#).

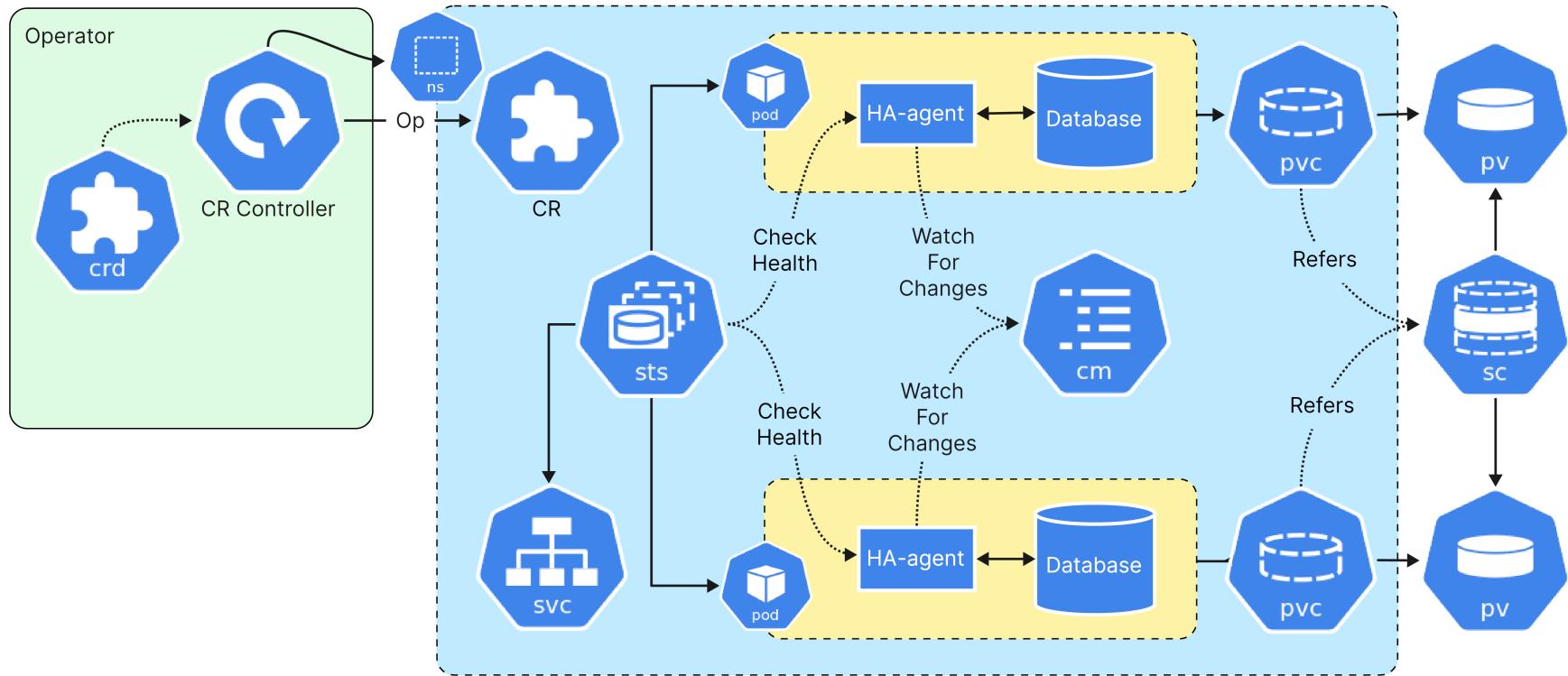
За одной абстракцией



Очень внимательный
инженер DBaaS



За одной абстракцией



Паттерн #3

49

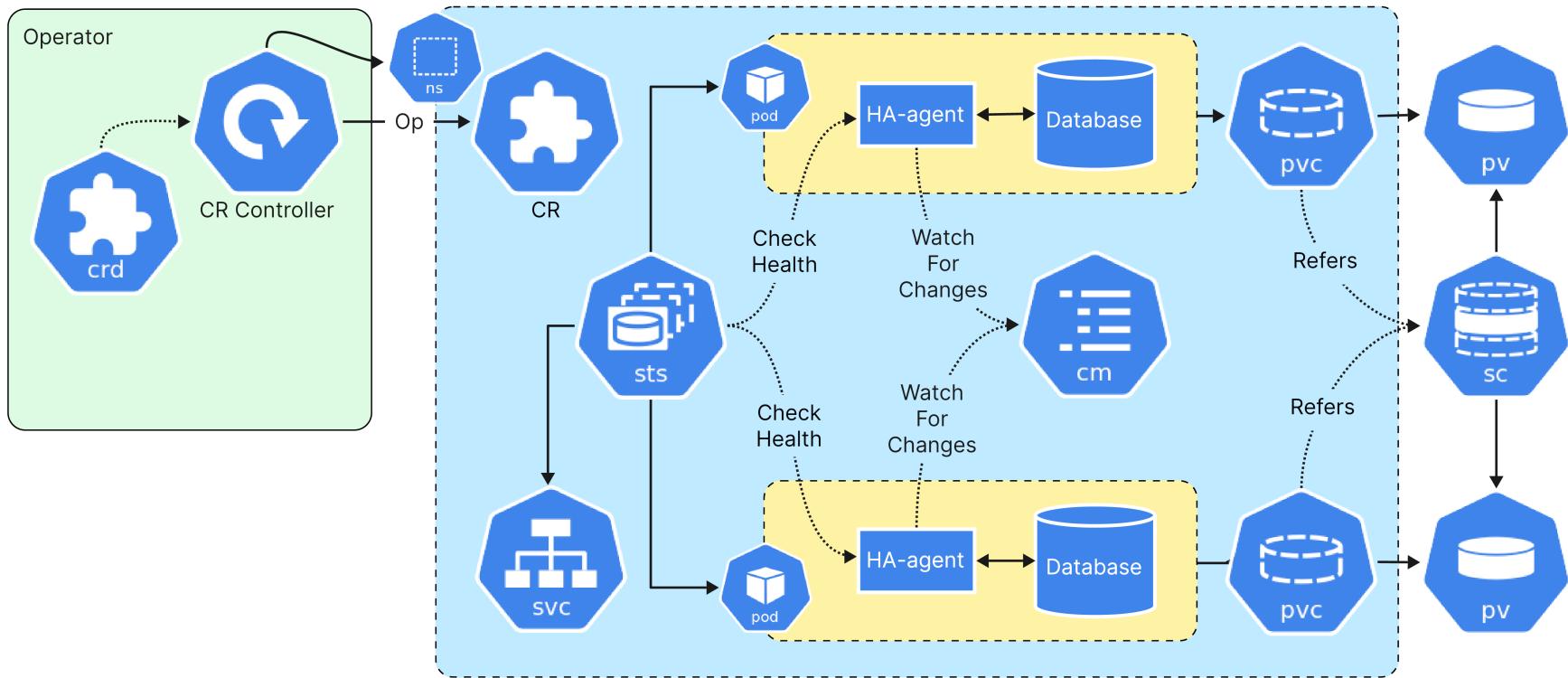
Используйте нативный для Kubernetes подход по группировке и полностью автоматическому управлению ресурсами — [Operator pattern](#).

Полная автоматизация

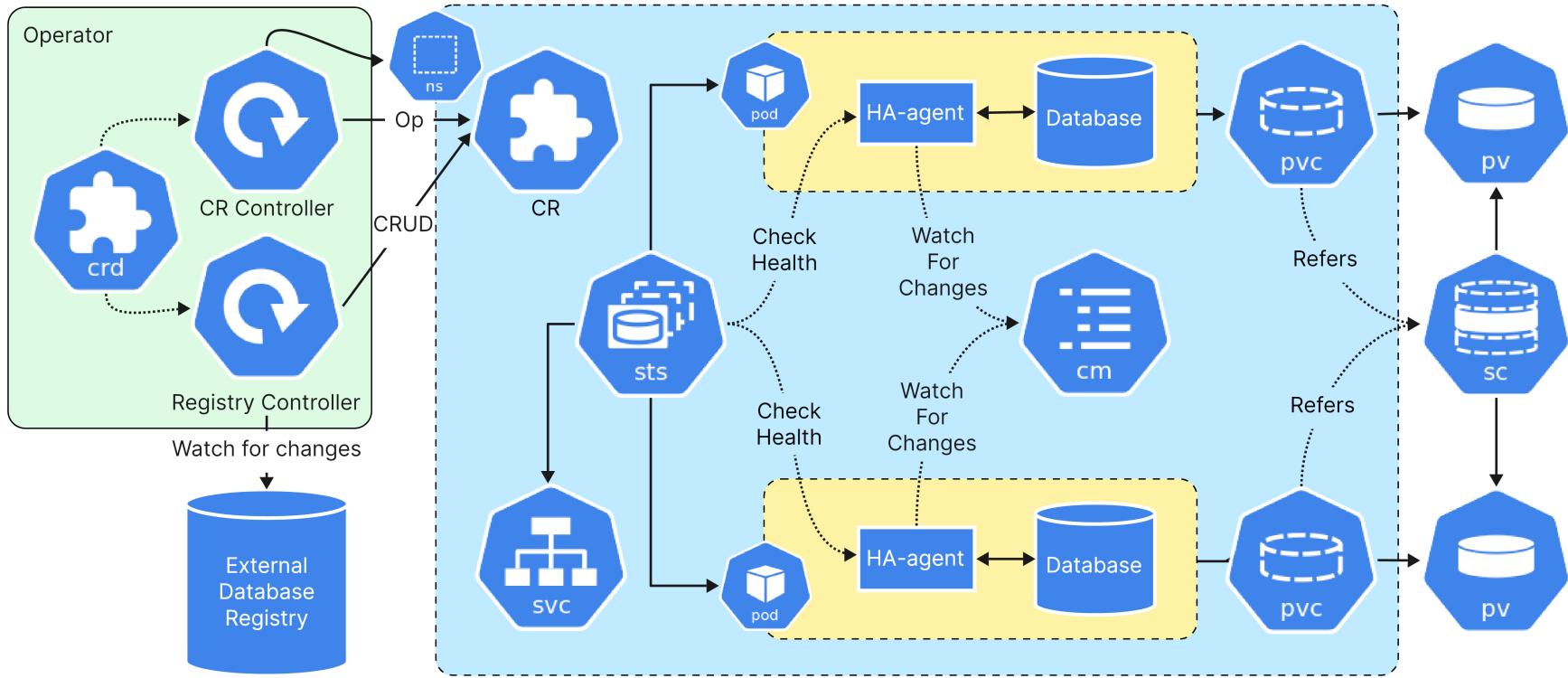
Для создания базы данных нужно создать только один CustomResource, а дальше operator сделает всё сам.

Но можно ли автоматизировать и это?

Разделяй и властвуй



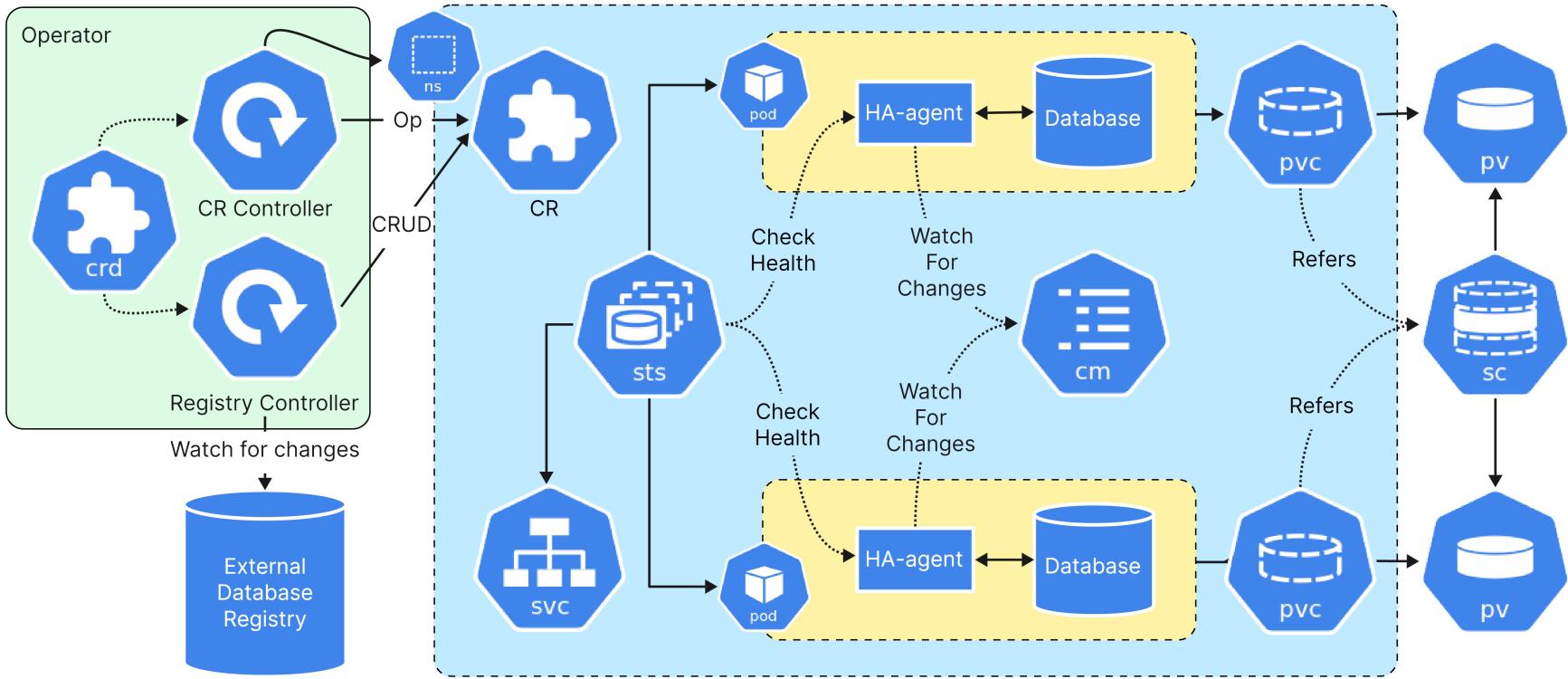
Разделяй и властвуй



Паттерн #4

Разделяйте operator на независимые контроллеры, каждый из которых будет выполнять собственную задачу по автоматизации.

Большая картина





Паттерны эксплуатации баз данных в мультиластерной среде

Переход Авито к multi-dc



Переход Авито к multi-dc

- Появилась необходимость управлять базами данных и обеспечивать их отказоустойчивость в рамках multi-dc (основное количество баз данных пришлось на Redis, PostgreSQL и MongoDB).

Переход Авито к multi-dc

- Появилась необходимость управлять базами данных и обеспечивать их отказоустойчивость в рамках multi-dc (основное количество баз данных пришлось на Redis, PostgreSQL и MongoDB).
- Появилась необходимость переносить базы данных между DC, а самое главное — делать это без привлечения конечных пользователей.

Переход Авито к multi-dc

Kubernetes-кластер «из коробки» плохо масштабируется на несколько дата-центров.

Переход Авито к multi-dc

Kubernetes-кластер «из коробки» плохо масштабируется на несколько дата-центров.

Наши варианты:

- ✗ Пуститься во все тяжкие и растянуть один большой кластер на несколько дата-центров.
- ✓ Адаптировать наши компоненты к работе в мультиклUSTERной среде.

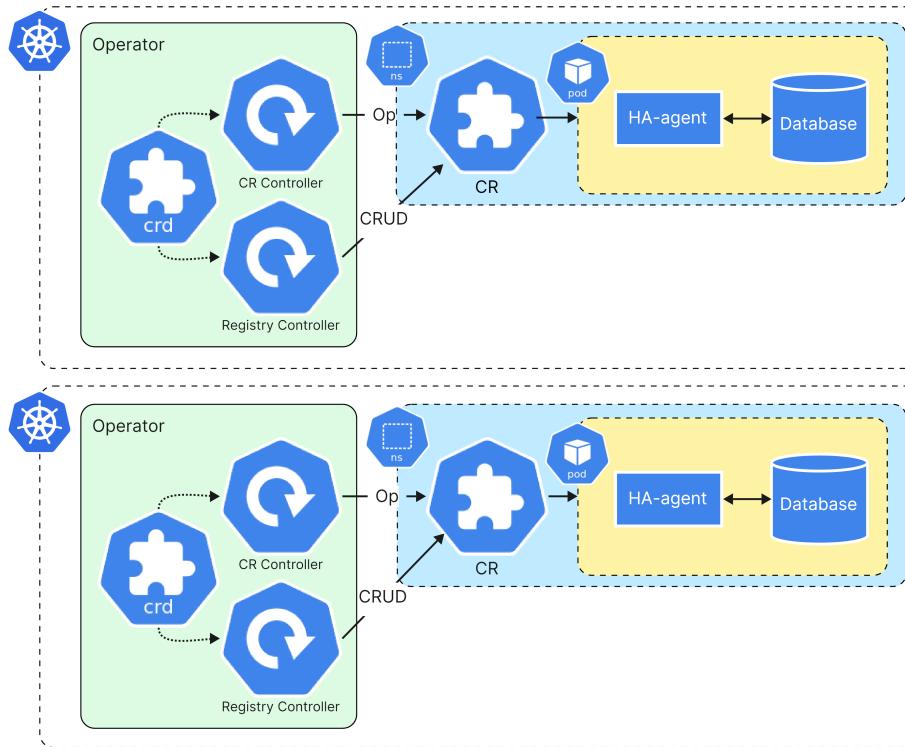
Кому верить?



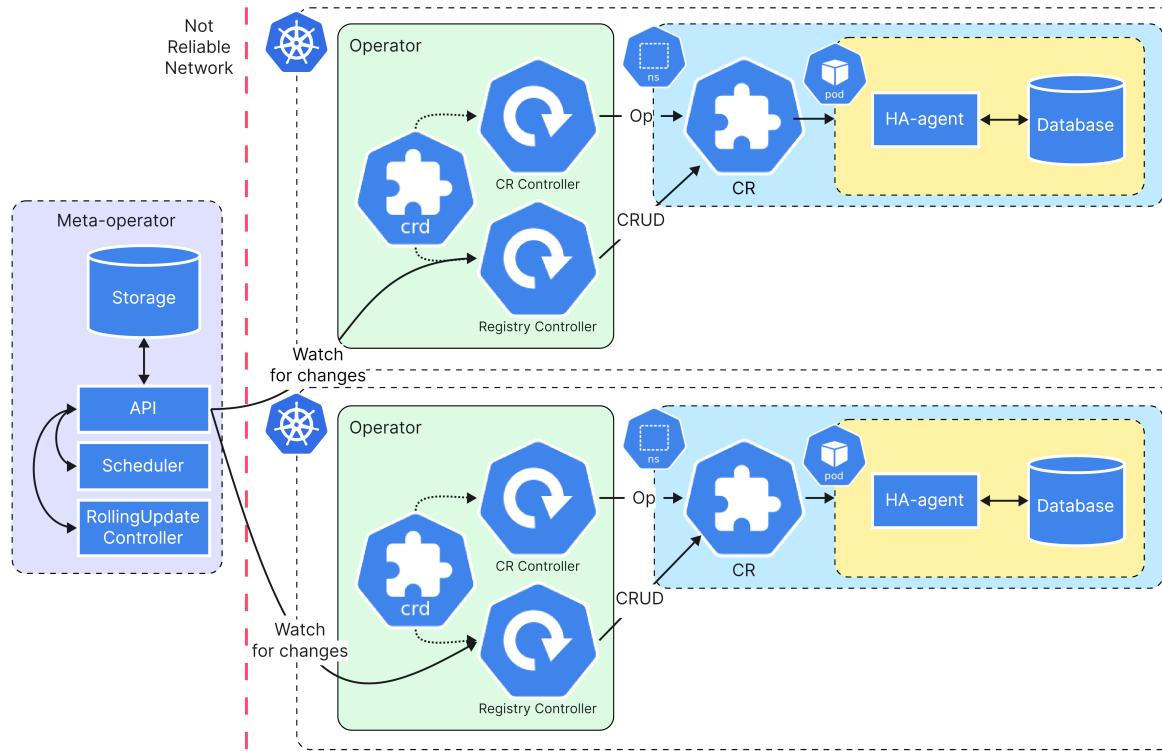
61



Кому верить?



Встречайте – meta-operator



Некоторые следствия



Некоторые следствия

Ненадёжная сеть разделяет
инфраструктуру на control plane
(meta-operator) и data plane (operator внутри k8s)

Некоторые следствия

Ненадёжная сеть разделяет инфраструктуру на control plane (meta-operator) и data plane (operator внутри k8s)

При отказе control plane отказ в data plane не происходит
базы данных продолжают обслуживать запросы пользователей

Некоторые следствия

Ненадёжная сеть разделяет инфраструктуру на control plane (meta-operator) и data plane (operator внутри k8s)

При отказе control plane отказ в data plane не происходит
базы данных продолжают обслуживать запросы пользователей

Scheduling становится двухуровневым
механизм scheduling по разным кластерам приходится реализовывать самостоятельно

Некоторые следствия

Ненадёжная сеть разделяет инфраструктуру на control plane (meta-operator) и data plane (operator внутри k8s)

При отказе control plane отказ в data plane не происходит
базы данных продолжают обслуживать запросы пользователей

Scheduling становится двухуровневым
механизм scheduling по разным кластерам приходится реализовывать самостоятельно

Механизм сохранения высокой доступности при RollingUpdate также становится двухуровневым
meta-operator должен применять изменения покластерно

Некоторые следствия

Ненадёжная сеть разделяет инфраструктуру на control plane (meta-operator) и data plane (operator внутри k8s)

При отказе control plane отказ в data plane не происходит
базы данных продолжают обслуживать запросы пользователей

Scheduling становится двухуровневым
механизм scheduling по разным кластерам приходится реализовывать самостоятельно

Механизм сохранения высокой доступности при RollingUpdate также становится двухуровневым
meta-operator должен применять изменения покластерно

Не привязаны к одному Kubernetes-кластеру

можем выводить из эксплуатации, вводить новые, работать на разных версиях и даже использовать managed кластеры в Cloud совместно с on premise

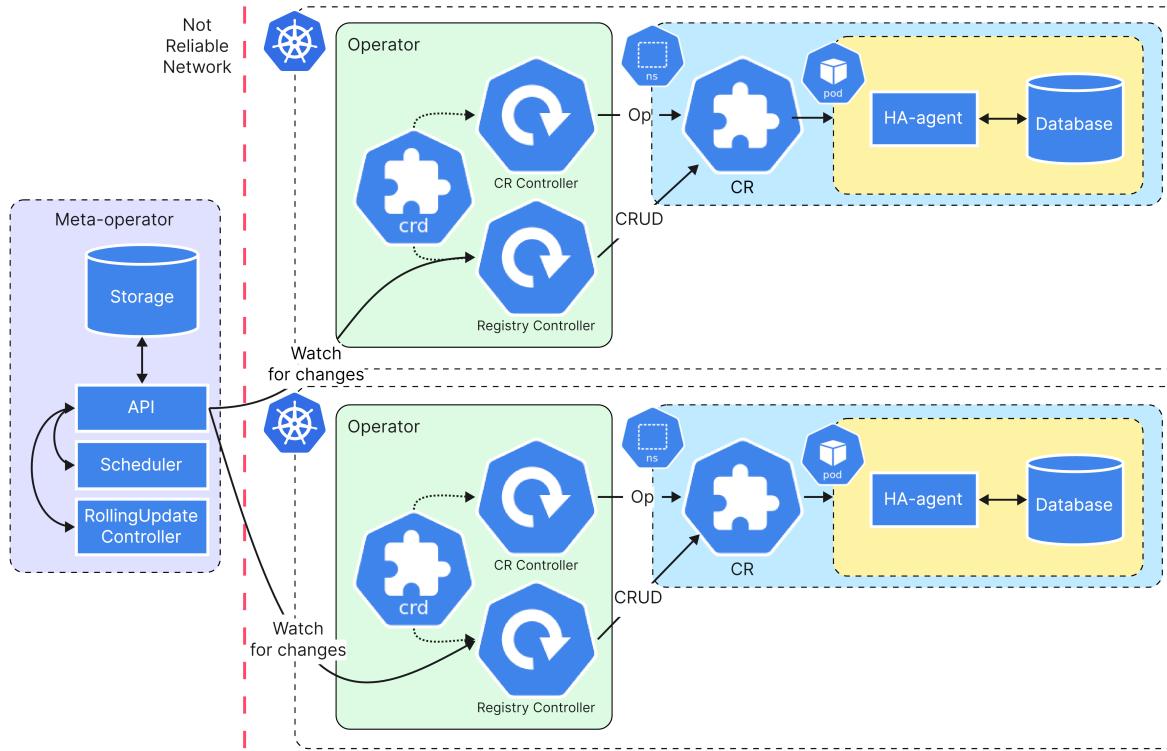
Паттерн #5

70

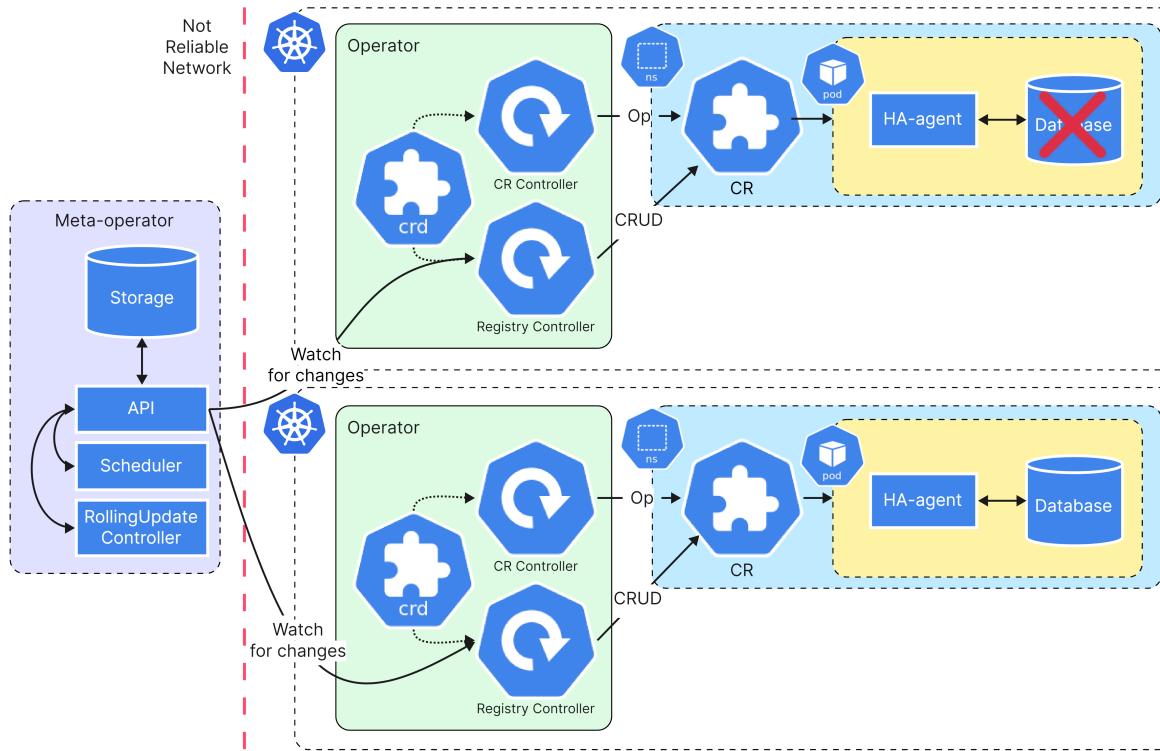
Разделяйте control plane и data plane таким образом, чтобы отказ control plane не приводил к отказу data plane

(в этом случае задача обеспечения отказоустойчивости control plane становится более простой и решать её можно отдельно)

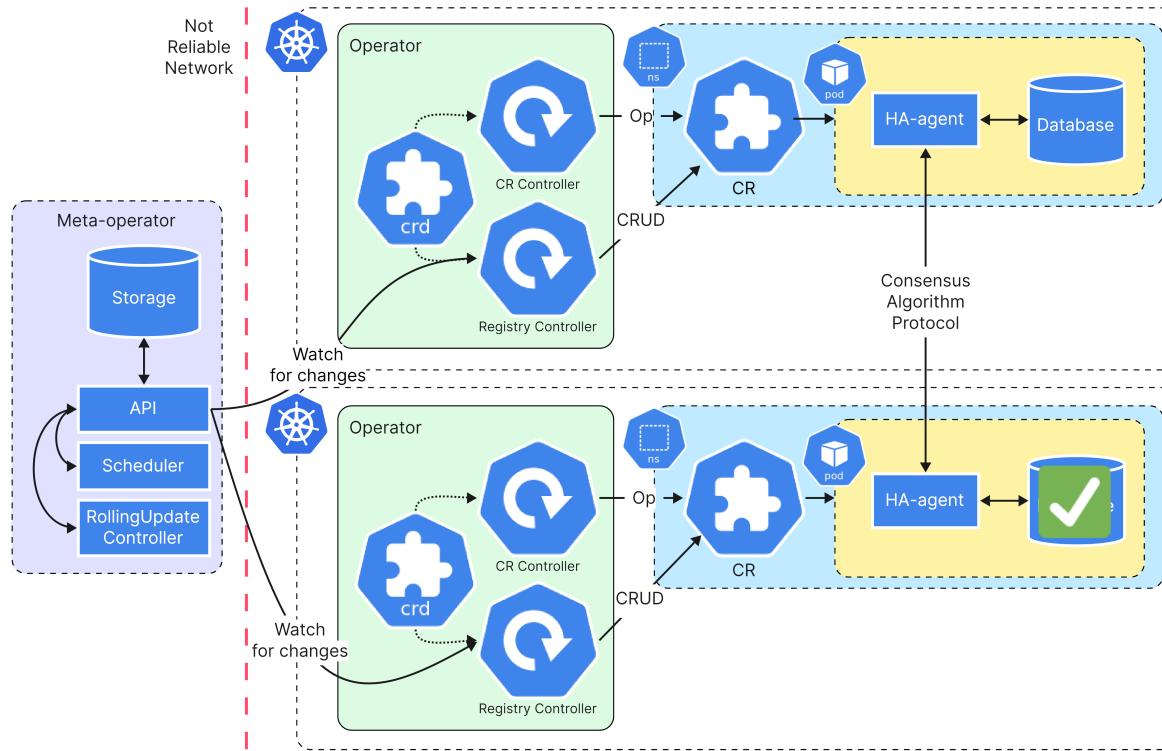
А если база отказалась?



А если база отказалась?



И опять HA-agent!

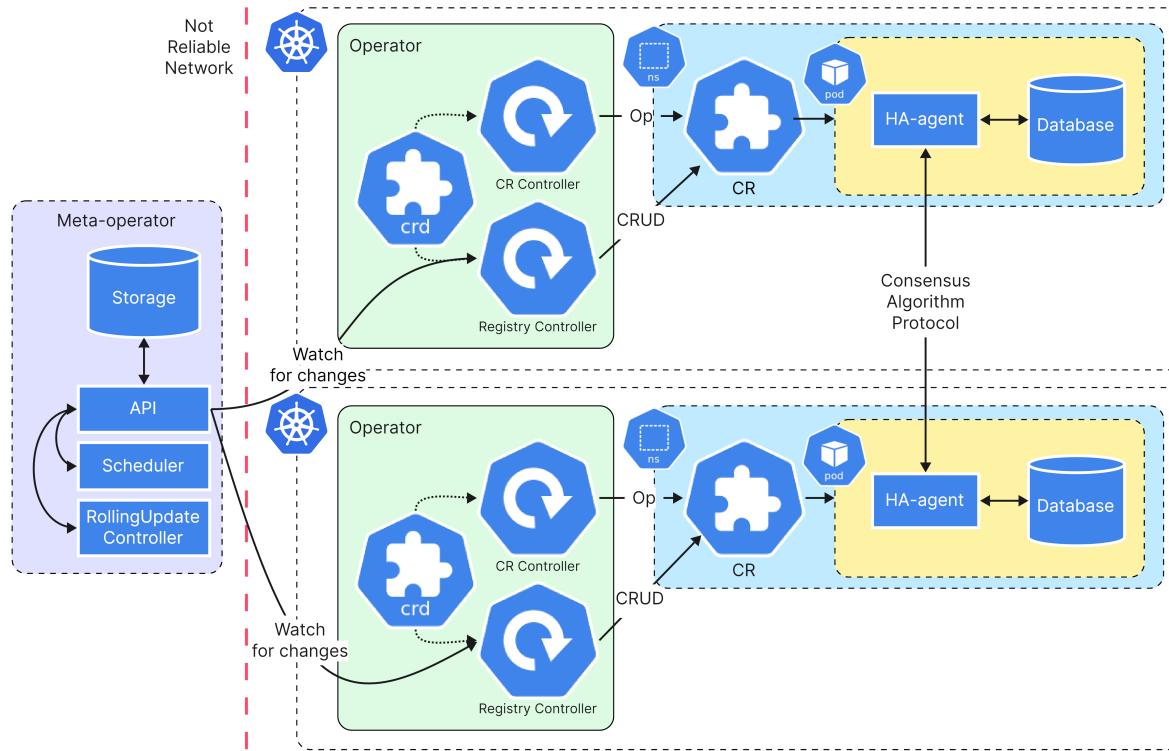


Паттерн #6

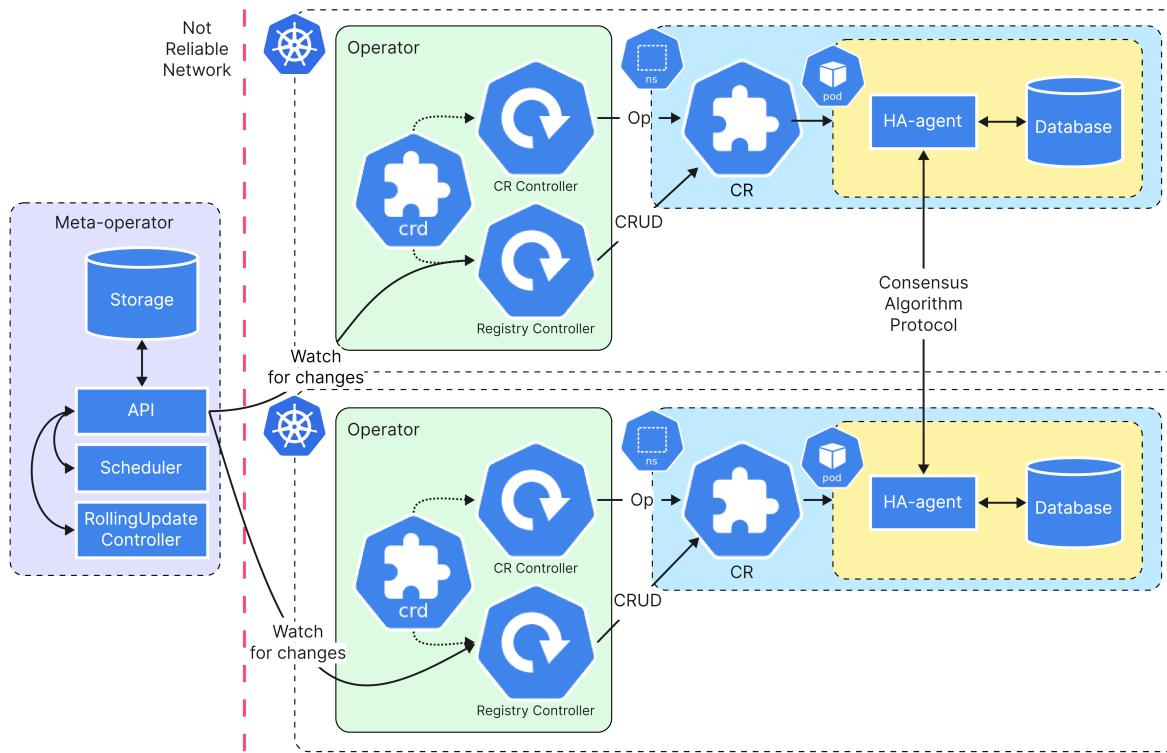
Решайте задачу leader election персонально для каждого кластера баз данных. Это позволит держать домен отказа маленьким и контролируемым, а также не завязываться на масштабируемость одного компонента.

(хорошая новость в том, что современные CloudNative СУБД умеют это «из коробки»)

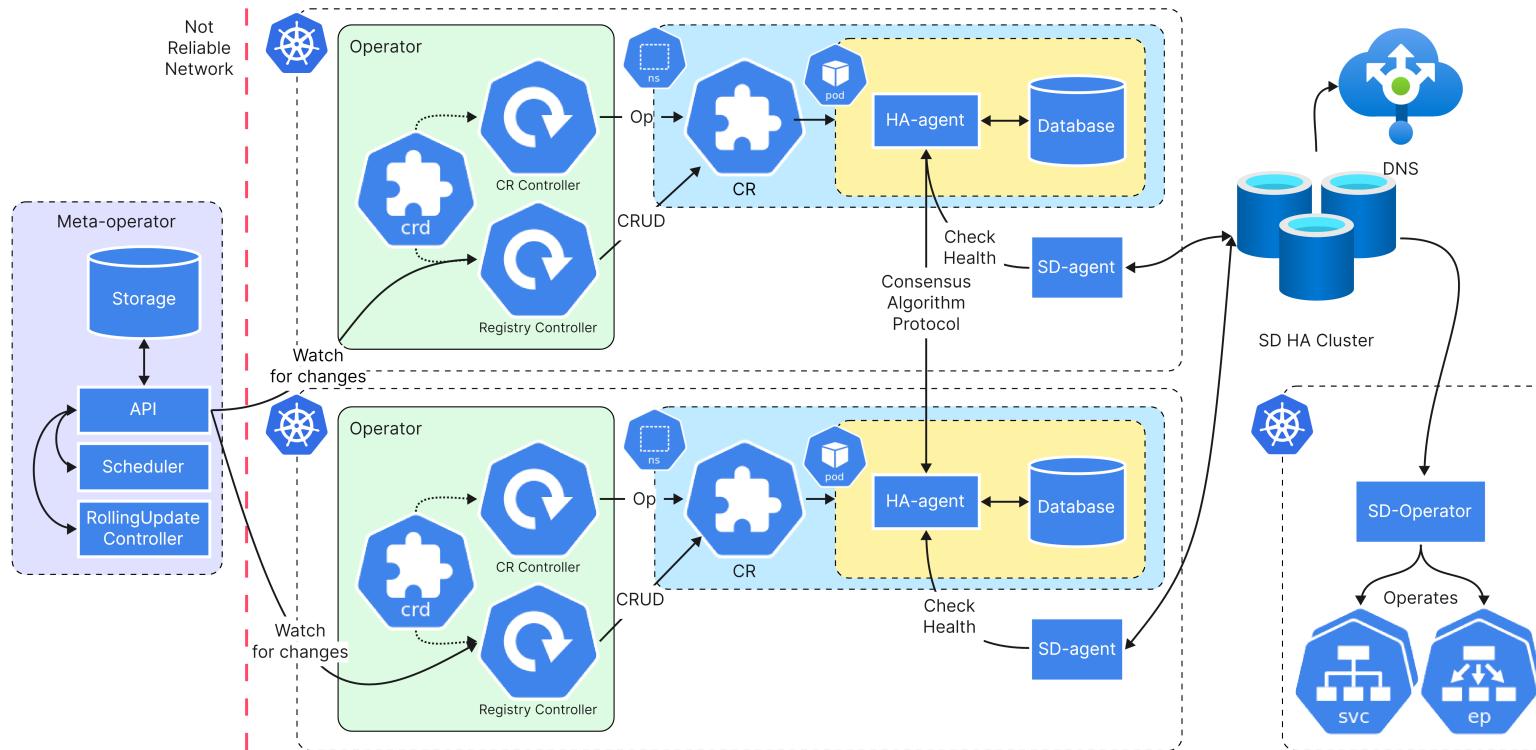
Помогите сервису найти его базу



Помогите сервису найти его базу



Возвращаем единую точку отказа?

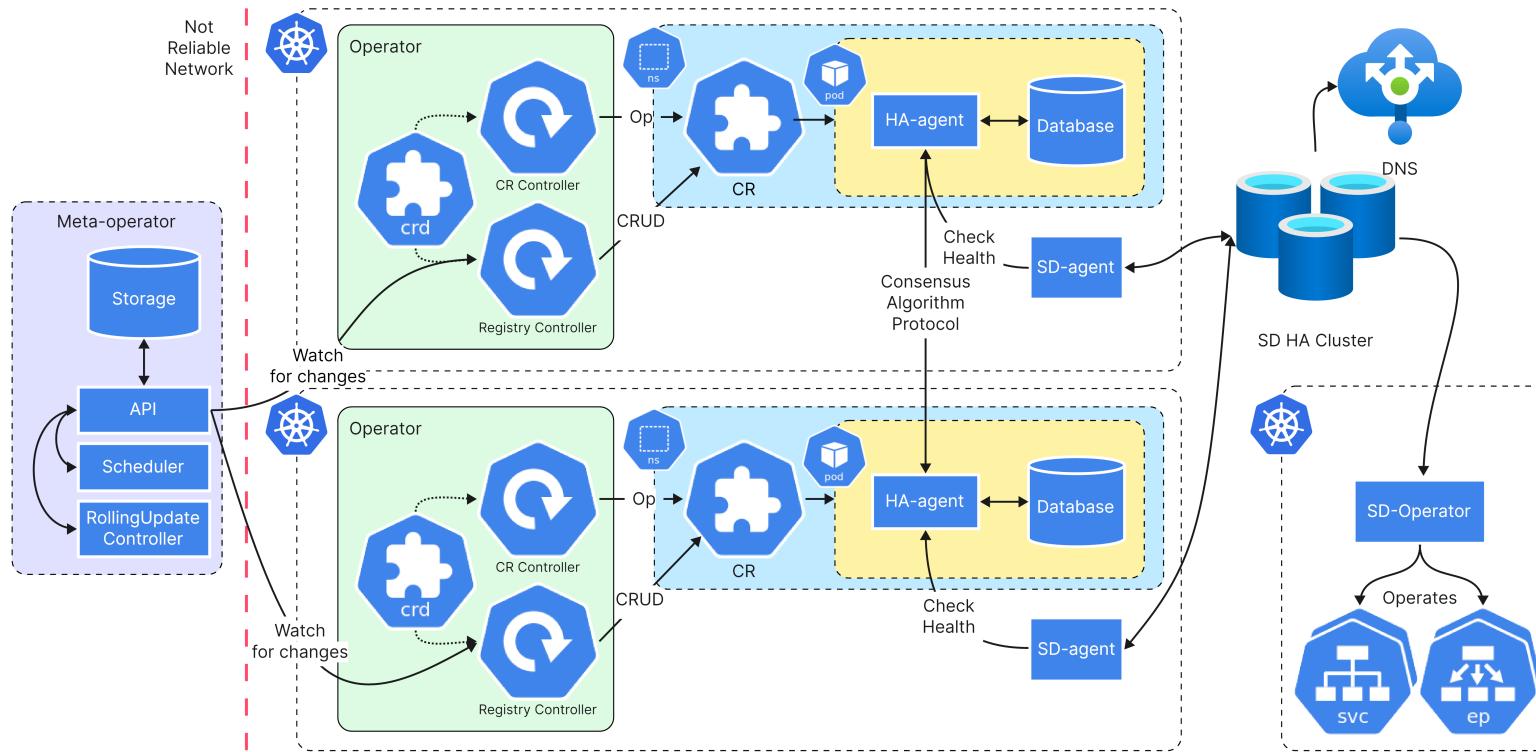


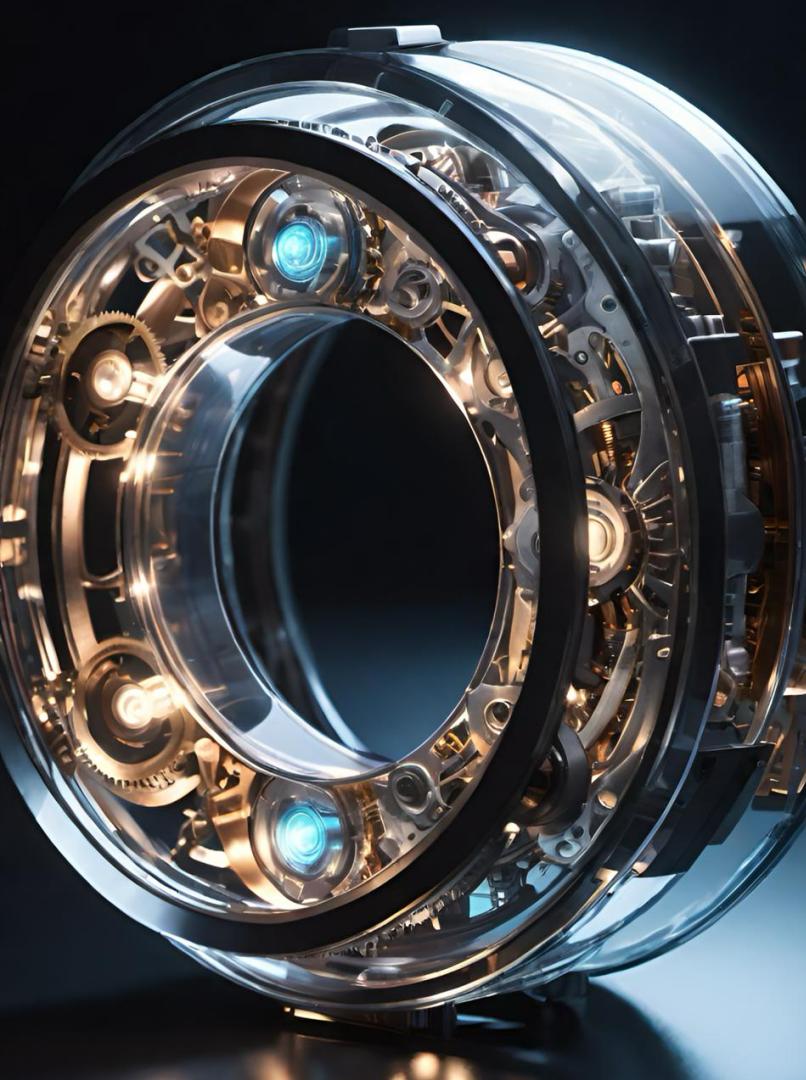
Паттерн #7

Для предоставления простого и согласованного способа обнаружения баз данных используйте централизованное отказоустойчивое решение.

Его кратковременный отказ должен происходить незаметно в штатных сценариях, а полная его утрата должна быть легко восполнима.

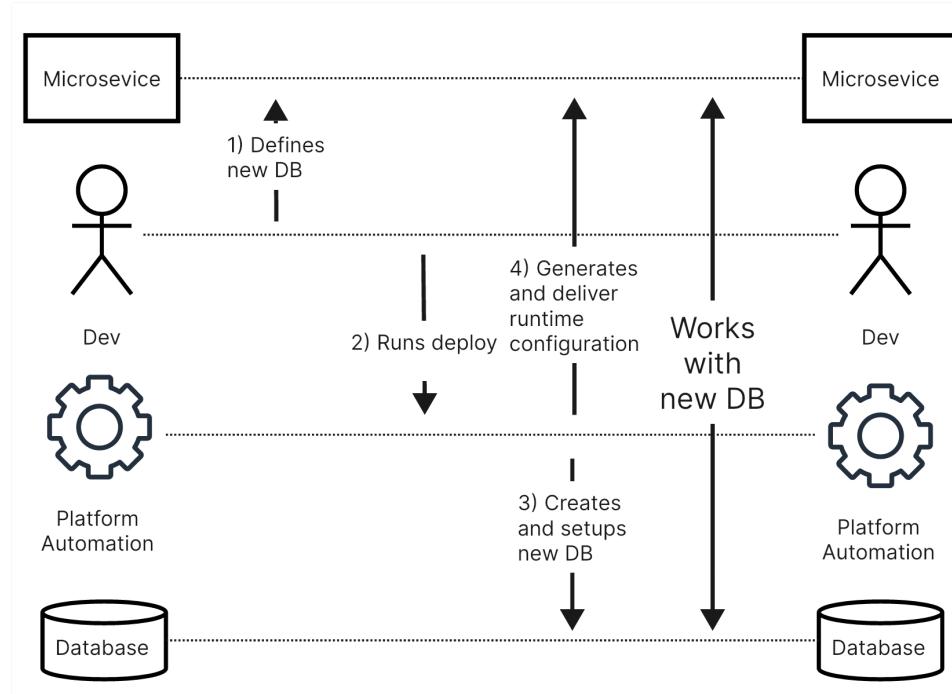
Большая картина





Выводы

Создание базы за 2-5 минут



Против 3-7 дней, которые были раньше

Избавление от рутины



* использована линейная экстраполяция по результатам первого квартала

Выводы



Выводы

- Инфраструктура не должна мешать бизнесу расти, а должна трансформироваться в платформу, автоматизация должна быть полной.

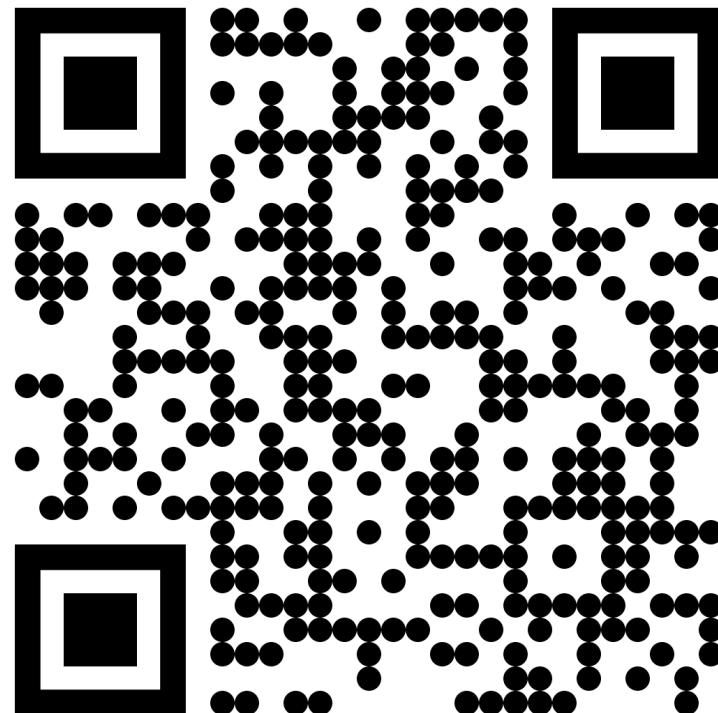
Выводы

- Инфраструктура не должна мешать бизнесу расти, а должна трансформироваться в платформу, автоматизация должна быть полной.
- Kubernetes — отличная и неплохо стандартизированная платформа, которая решает массу задач (эксплуатации).

Выводы

- Инфраструктура не должна мешать бизнесу расти, а должна трансформироваться в платформу, автоматизация должна быть полной.
- Kubernetes — отличная и неплохо стандартизированная платформа, которая решает массу задач (эксплуатации).
- Не нужно бояться использовать Kubernetes для баз данных: они прекрасно там себя чувствуют.

Материалы по теме





"If a human operator needs to touch your system during normal operations, you have a bug. The definition of normal changes as your systems grow".

Владимир Алёшин,
руководитель разработки платформы DBaaS



Авито

Оцените доклад, перейдя по QR-коду

**Спасибо за
внимание!**

