- Amazon Dynamo -

высокодоступное хранилище данных ключ-значение

~ Целевая аудитория ~

… сервисы, которым необходимы:

~ Целевая аудитория ~

… сервисы, которым необходимы:

- высокая доступность

~ Целевая аудитория ~

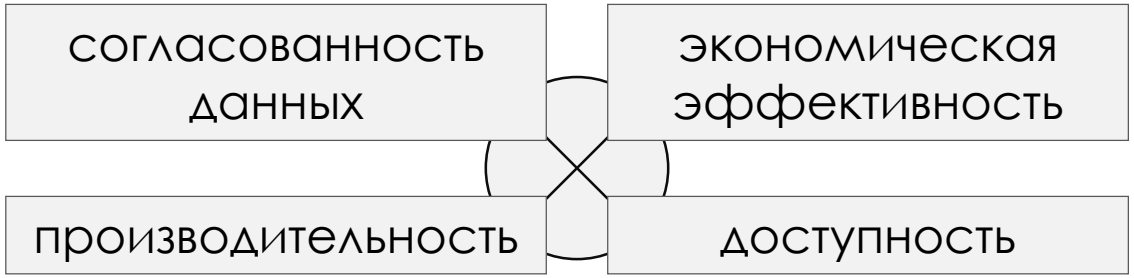
… сервисы, которым необходимы:

- высокая доступность
- формат данных: ключ-значение

~ Целевая аудитория ~

… сервисы, которым необходимы:

- высокая доступность
- формат данных: ключ-значение
- контроль над балансом:



~ Системные характеристики ~

~ Системные характеристики ~

Параметры ACID
(атомарность, непротиворечивость, ~~изоляция~~, долговечность)

Нет гарантий по изоляции
Изменение только 1 записи за раз

~ Системные характеристики ~

Параметры ACID (атомарность, непротиворечивость, ~~изоляция~~, долговечность)

Нет гарантий по изоляции
Изменение только 1 записи за раз

Экономическая эффективность

Гарантированное время отклика

~ Системные характеристики ~

Параметры ACID (атомарность, непротиворечивость, ~~изоляция~~, долговечность)

Нет гарантий по изоляции
Изменение только 1 записи за раз

Модель запросов

Простые чтение и запись

Экономическая эффективность

Гарантированное время отклика

~ Системные характеристики ~

Параметры ACID (атомарность, непротиворечивость, ~~изоляция~~, долговечность)

Нет гарантий по изоляции
Изменение только 1 записи за раз

Экономическая эффективность

Гарантированное время отклика

Модель запросов

Простые чтение и запись

Дополнительное допущение

«Не враждебное» рабочее окружение

~ Архитектурные решения ~

~ Архитектурные решения ~

Разрешение конфликтов

Конфликт разрешается при чтении

~ Архитектурные решения ~

Разрешение конфликтов

Конфликт разрешается при чтении

Симметричность

У узлов один набор функций

~ Архитектурные решения ~

Разрешение конфликтов

Конфликт разрешается при чтении

Симметричность

У узлов один набор функций

Гетерогенность

Честное распределение нагрузки

~ Архитектурные решения ~

Разрешение конфликтов

Конфликт разрешается при чтении

Симметричность

У узлов один набор функций

Гетерогенность

Честное распределение нагрузки

Масштабируемость

Добавление и удаление узлов

~ Архитектурные решения ~

Разрешение конфликтов

Конфликт разрешается при чтении

Симметричность

У узлов один набор функций

Гетерогенность

Честное распределение нагрузки

Масштабируемость

Добавление и удаление узлов

Децентрализация

Управление через децентрализованные P2P технологии

- Архитектура системы -

смотрим под капот, но железо пока не трогаем

~ Решаемые задачи ~

~ Решаемые задачи ~

Сегментирование

Распределение данных по узлам

~ Решаемые задачи ~

Сегментирование

Распределение данных по узлам

Доступность записи

Запись должна срабатывать всегда

~ Решаемые задачи ~

Сегментирование

Распределение данных по узлам

Поддержка временных неисправностей

Доступность записи

Запись должна срабатывать всегда

~ Решаемые задачи ~

Сегментирование

Распределение данных по узлам

Доступность записи

Запись должна срабатывать всегда

Поддержка временных неисправностей

Восстановление после долговременного отключения

~ Решаемые задачи ~

Сегментирование

Распределение данных по узлам

Доступность записи

Запись должна срабатывать всегда

Поддержка временных неисправностей

Восстановление после долговременного отключения

Членство в кластере и выявление неисправностей

~ Системный интерфейс ~

get (key)

Возвращает объект с контекстом

Если есть конфликты:
Возвращает список объектов с их контекстами

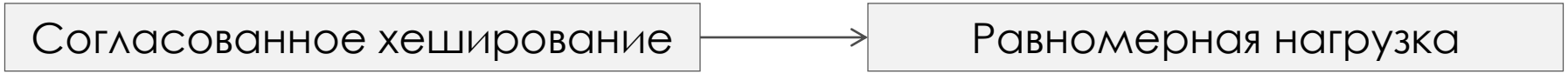
put (key, context, object)

Записывает объект на диск

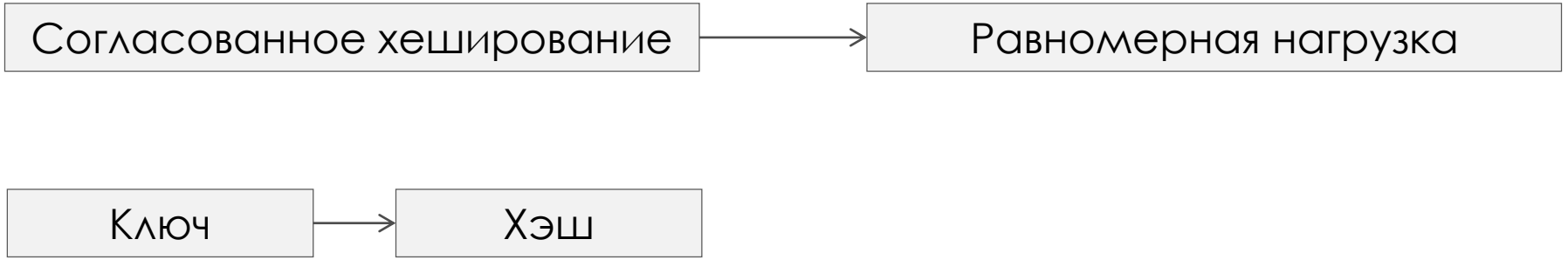
Контекст:
Системные метаданные
(например, версия объекта)

Хранится вместе с объектом
(новый может быть проверен)

~ Сегментирование ~

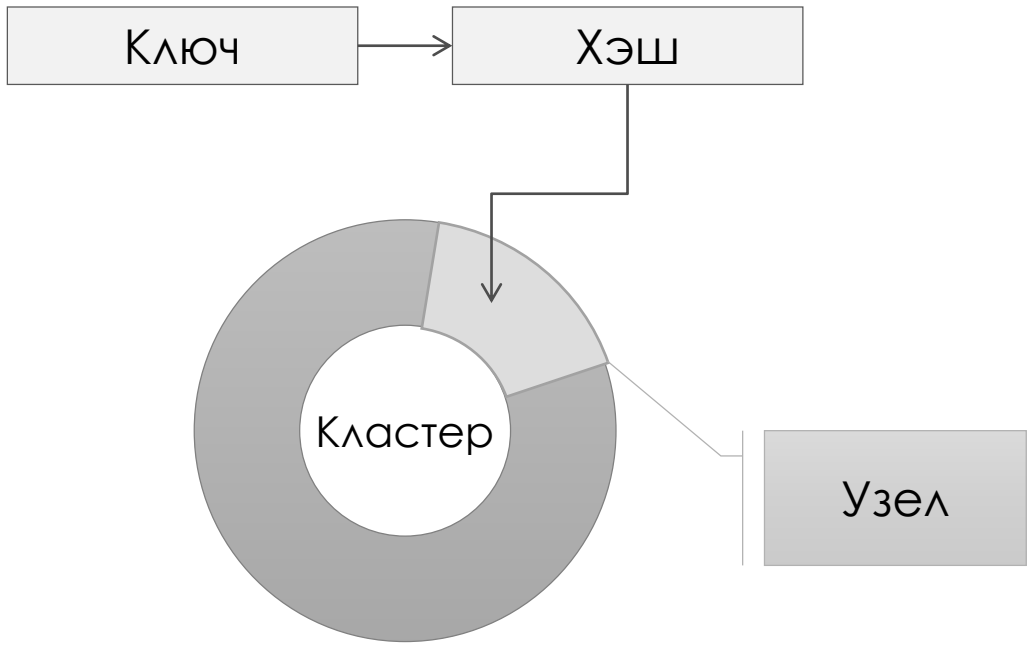


~ Сегментирование ~

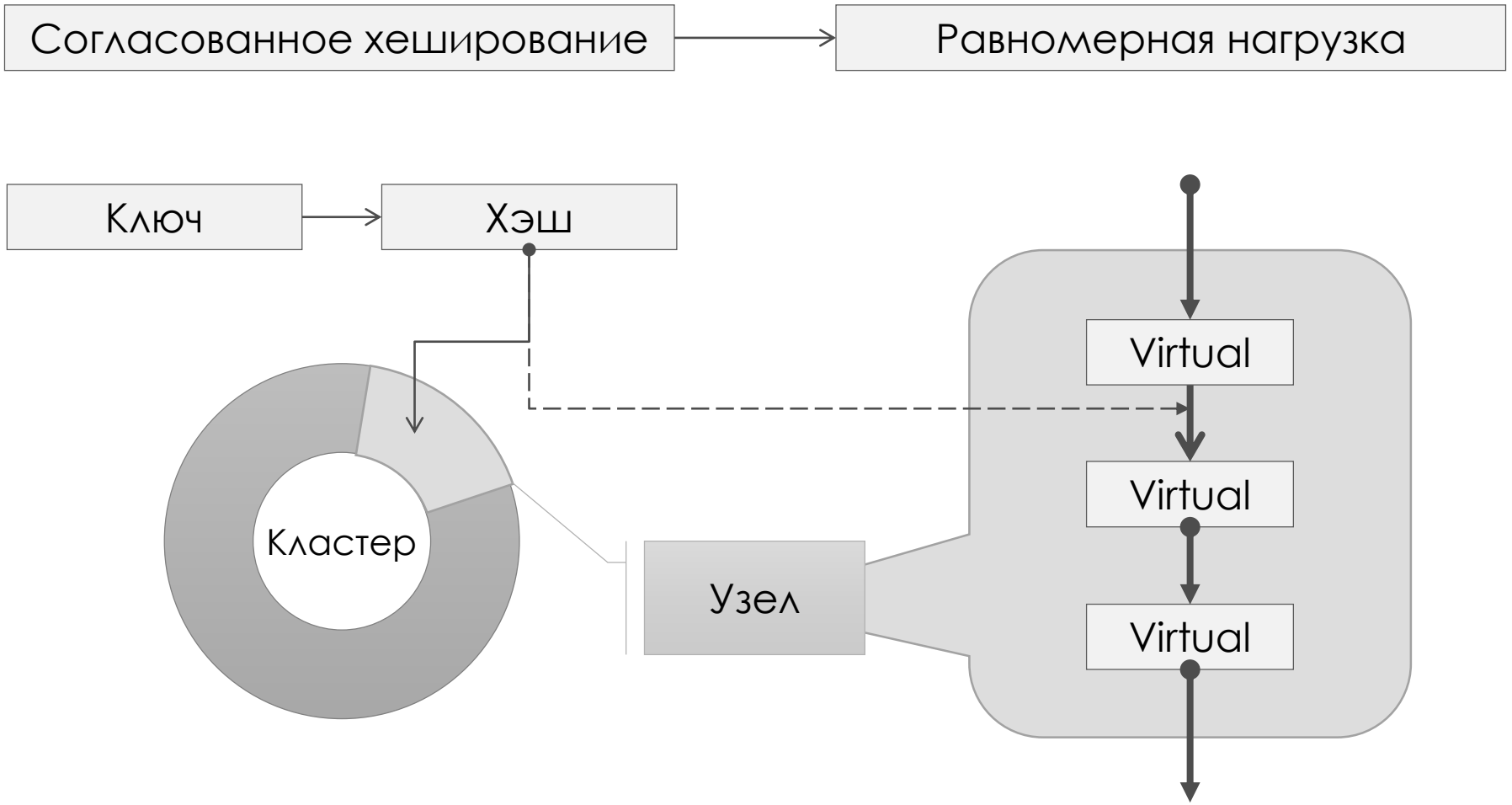


~ Сегментирование ~

Согласованное хеширование → Равномерная нагрузка



~ Сегментирование ~



~~ Преимущества virtual узлов ~~

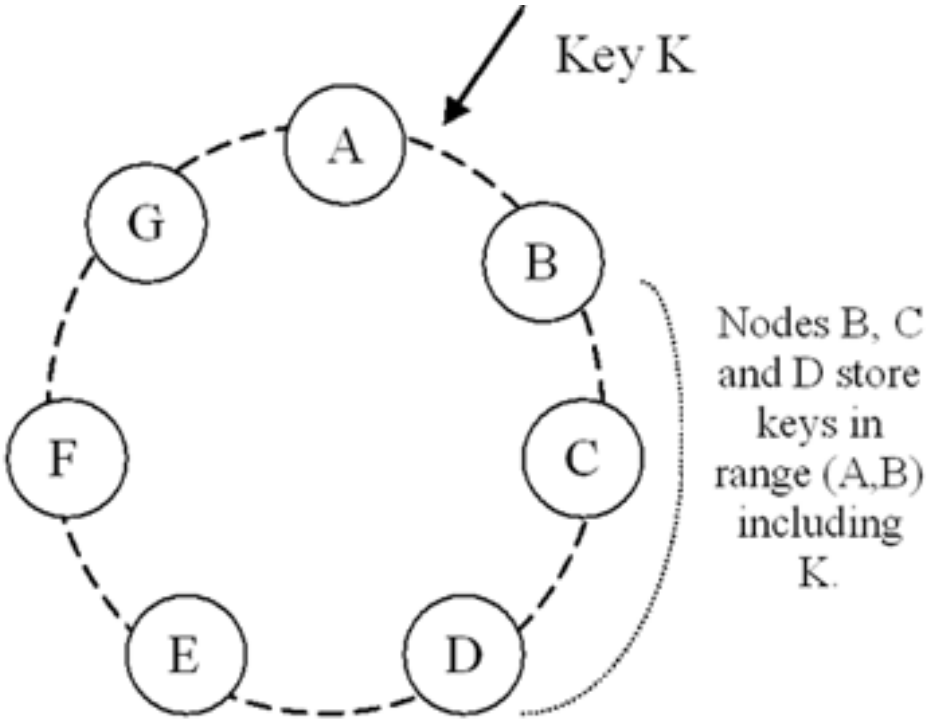
Масштабируемость

Количество виртуальных узлов зависит от мощности физического узла

Устойчивость системы к неполадкам в узлах

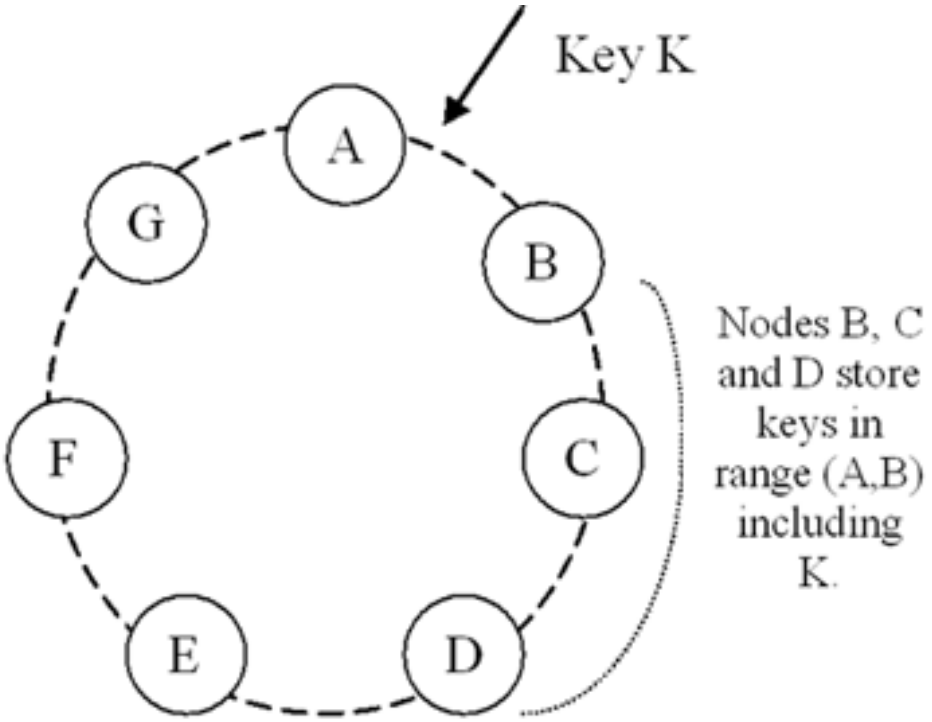
Распределение нагрузки на все доступные узлы

~ Репликация данных ~



N - параметр системы:
число копий каждого узла

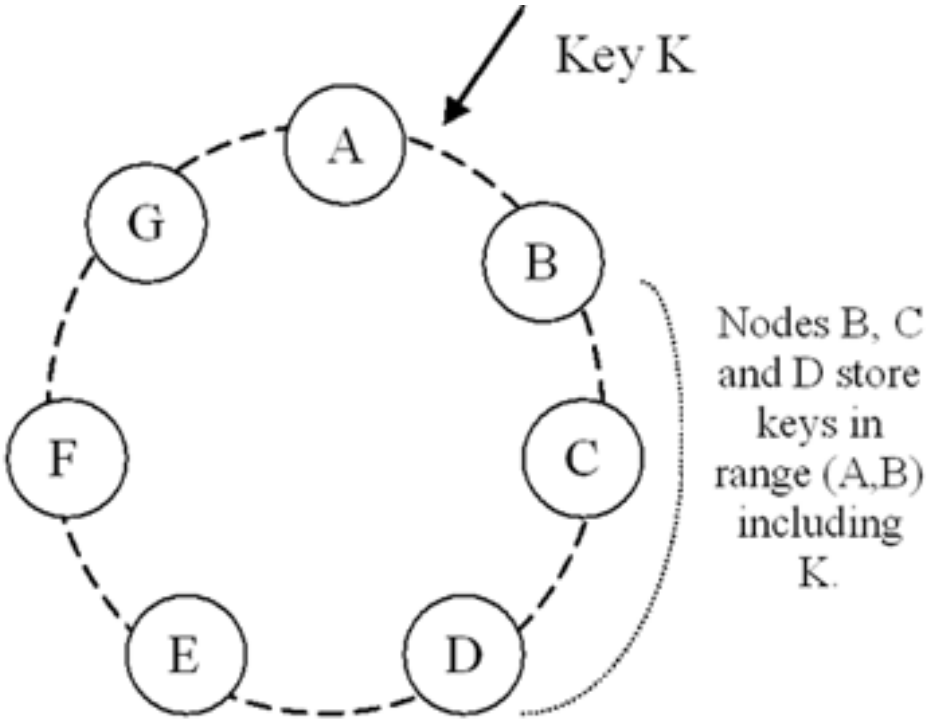
~ Репликация данных ~



N - параметр системы:
число копий каждого узла

Узел B

~ Репликация данных ~

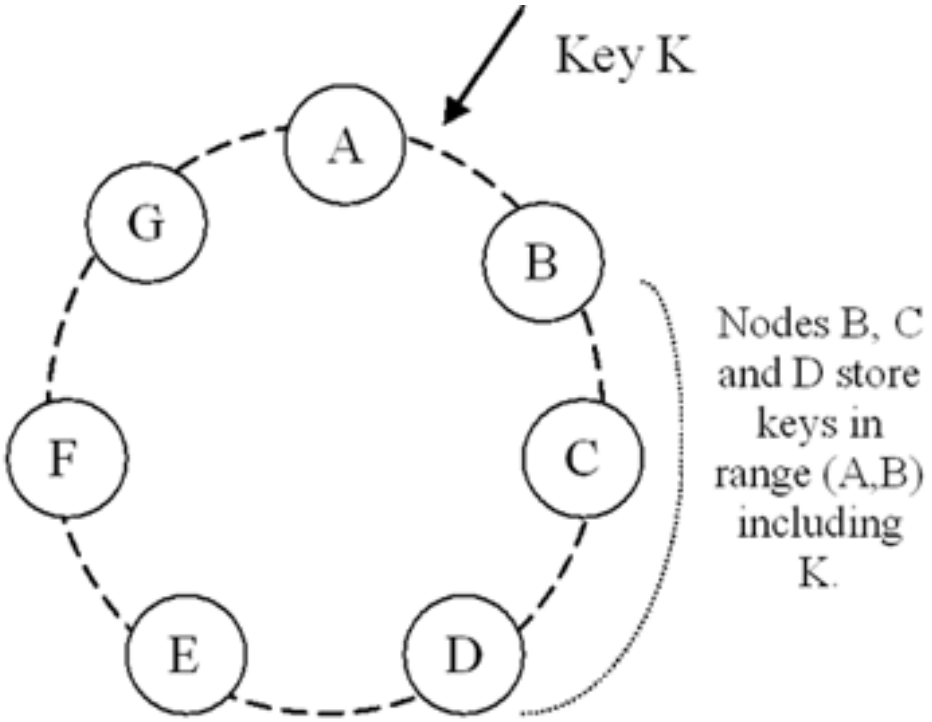


N - параметр системы:
число копий каждого узла

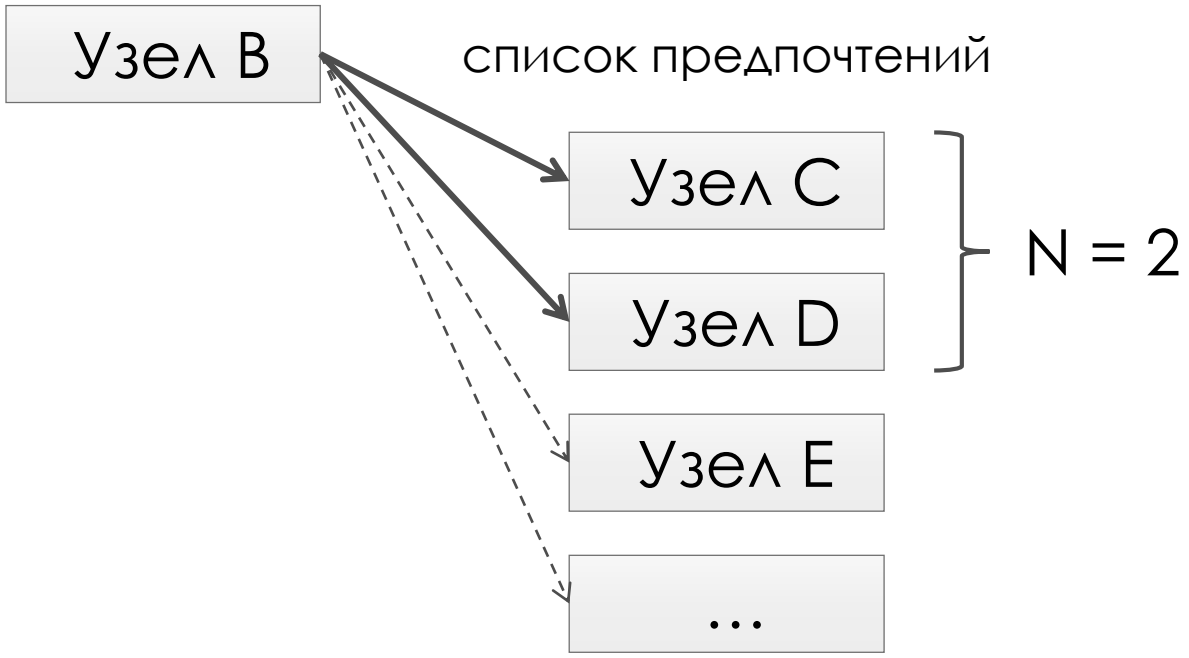
Узел B — список предпочтений

- Узел C
- Узел D
- Узел E
- …

~ Репликация данных ~



N - параметр системы:
число копий каждого узла



~ Версии данных ~

~ Версии данных ~

Каждая правка объекта - **неизменяемая** версия данных

~ Версии данных ~

Каждая правка объекта - **неизменяемая** версия данных

Вместе с каждым объектом хранятся **векторные** часы

~ Версии данных ~

Каждая правка объекта - **неизменяемая** версия данных

Вместе с каждым объектом хранятся **векторные** часы

Размер вектора не более 10 для каждого объекта

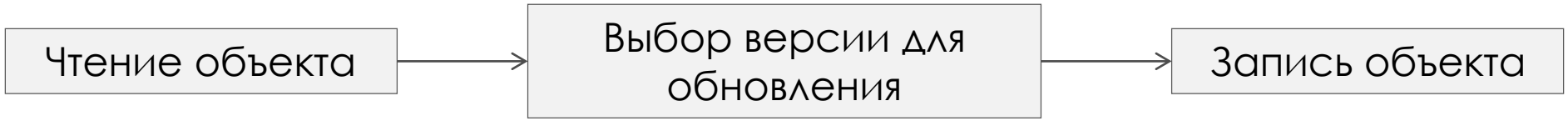
~ Версии данных ~

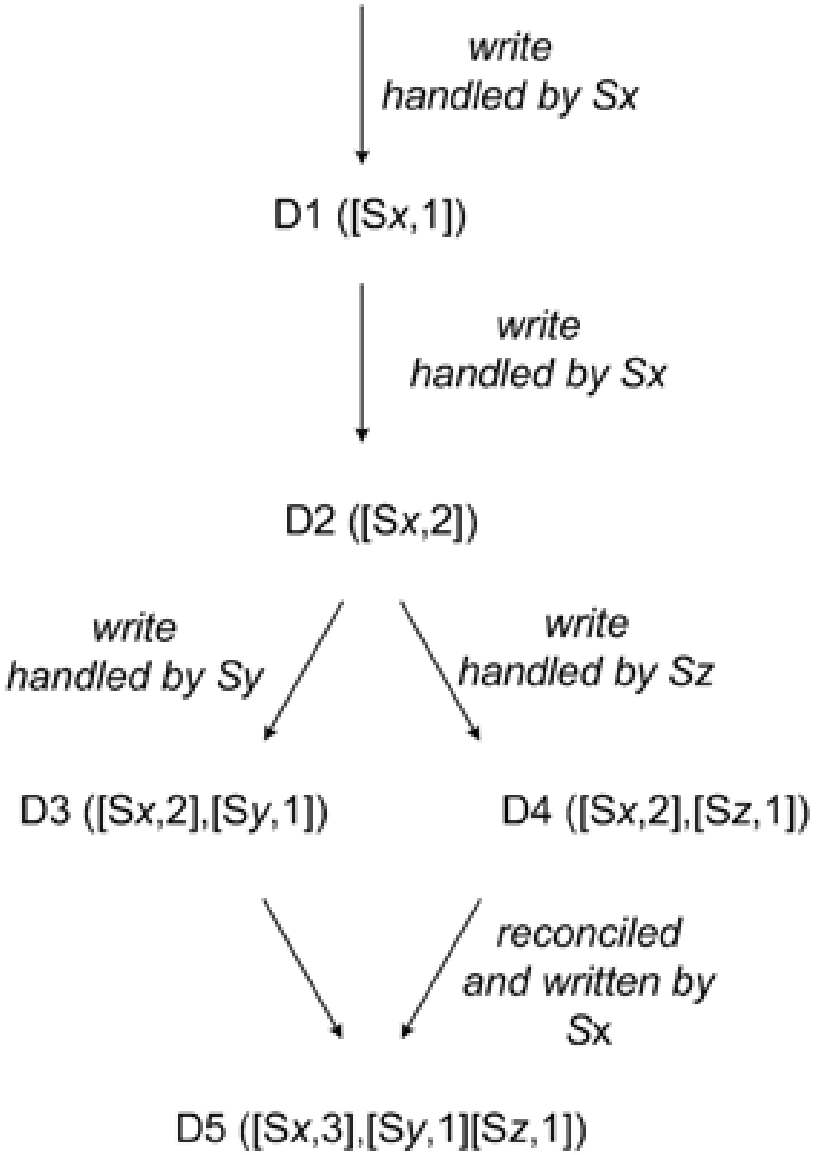
Каждая правка объекта - **неизменяемая** версия данных

Вместе с каждым объектом хранятся **векторные** часы

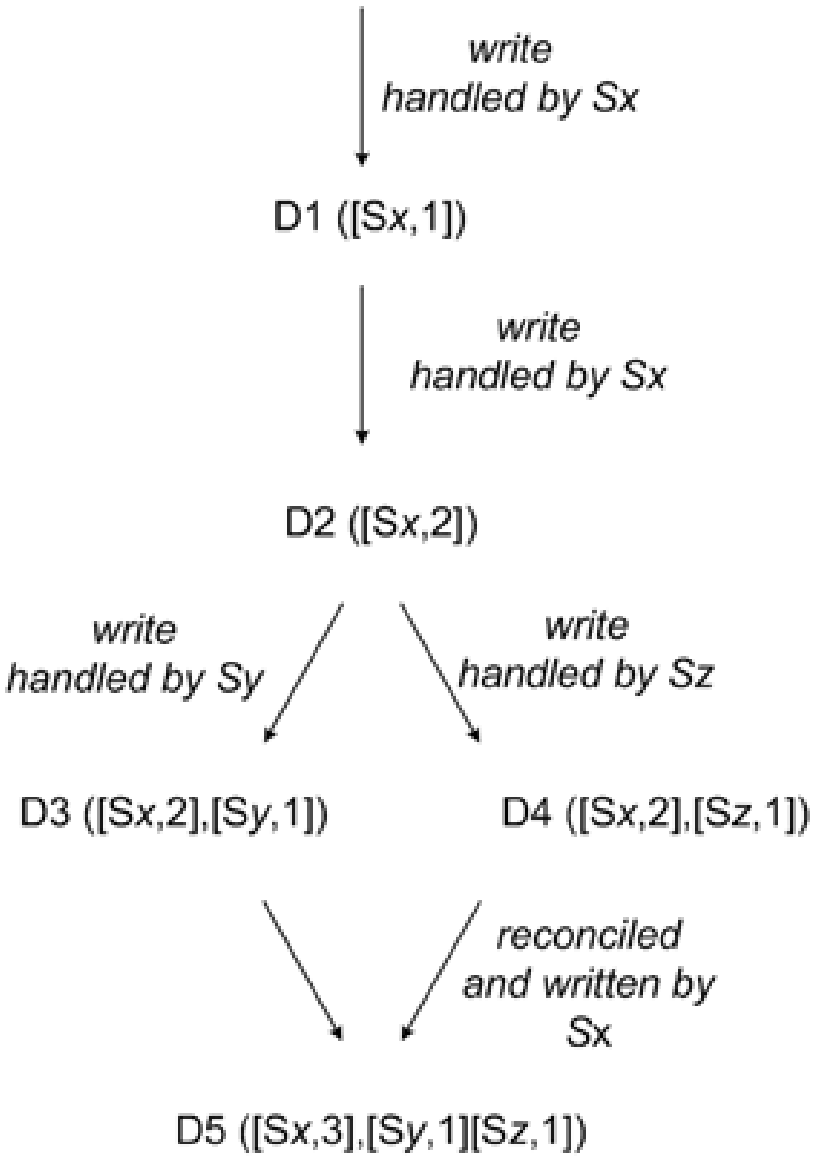
Размер вектора не более 10 для каждого объекта

Операция обновления данных:



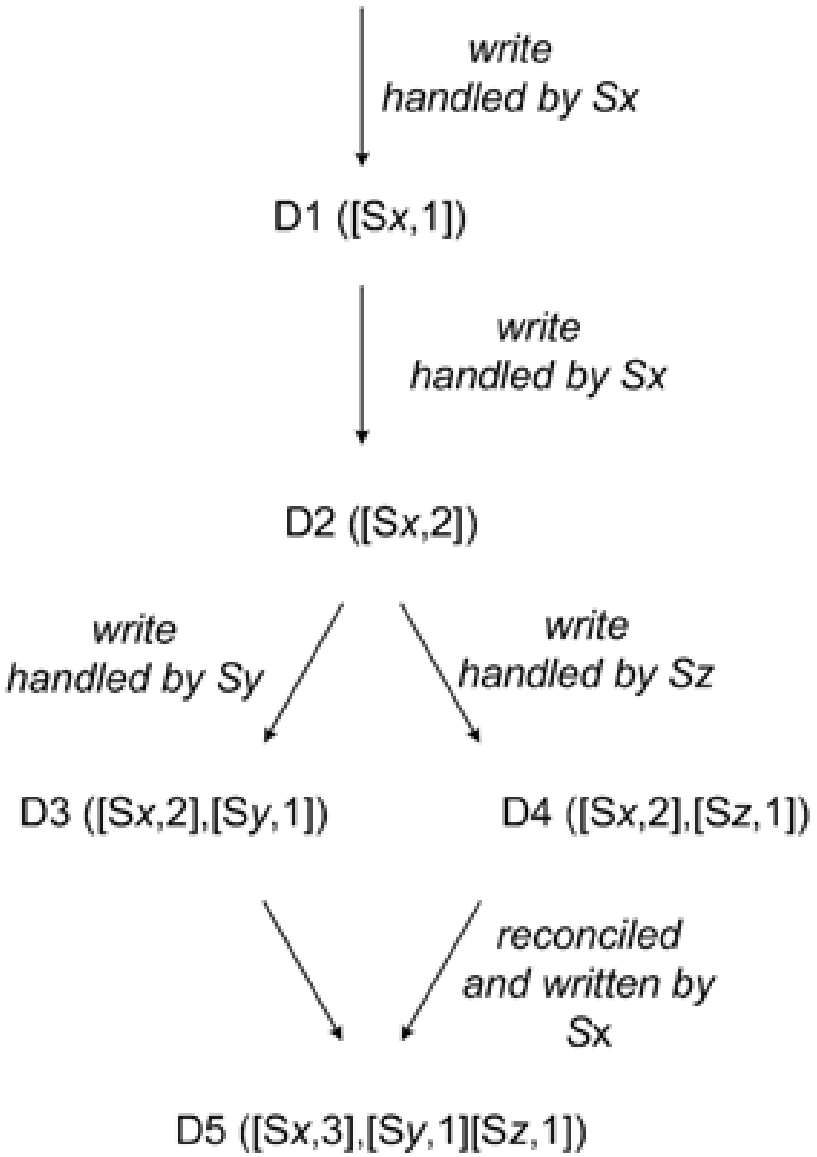


- Запись объекта была обработана узлом Sx



- Запись объекта была обработана узлом Sx

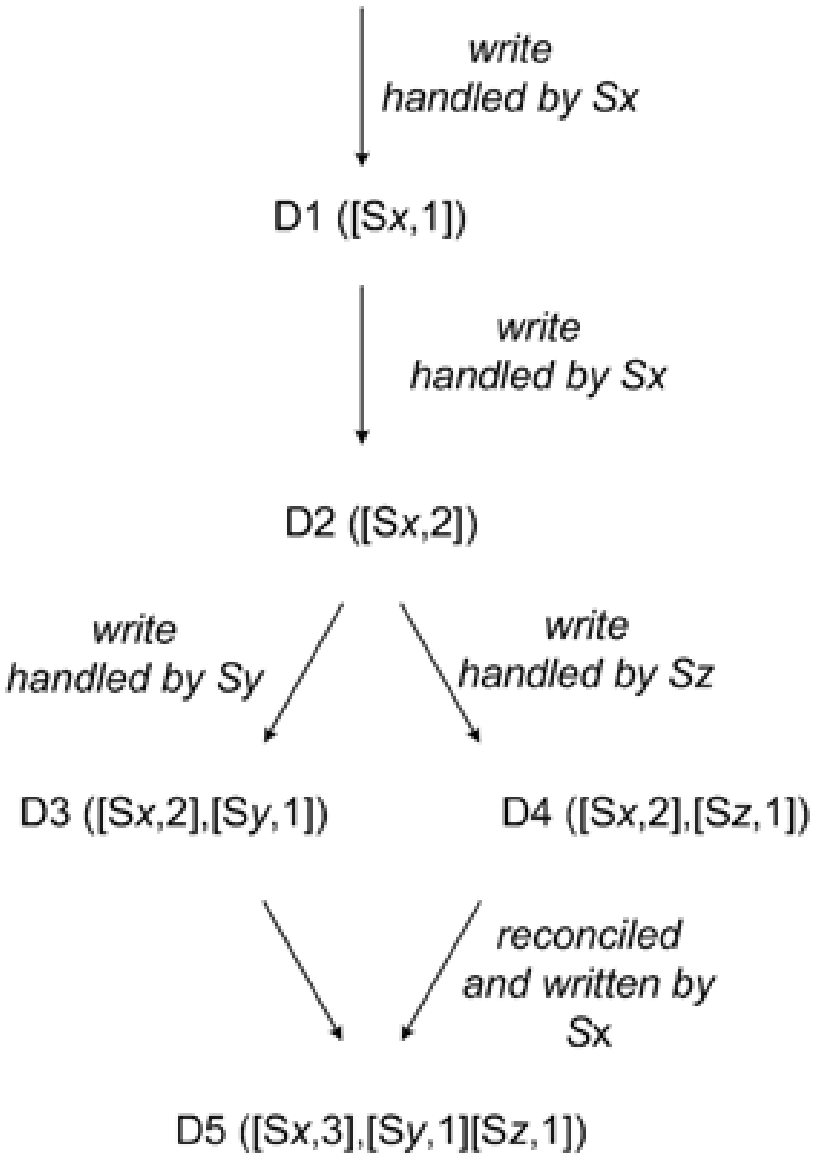
- Версия объекта, со связанными часами



- Запись объекта была обработана узлом Sx

- Версия объекта, со связанными часами

- Перезапись объекта, обработанная узлом Sx

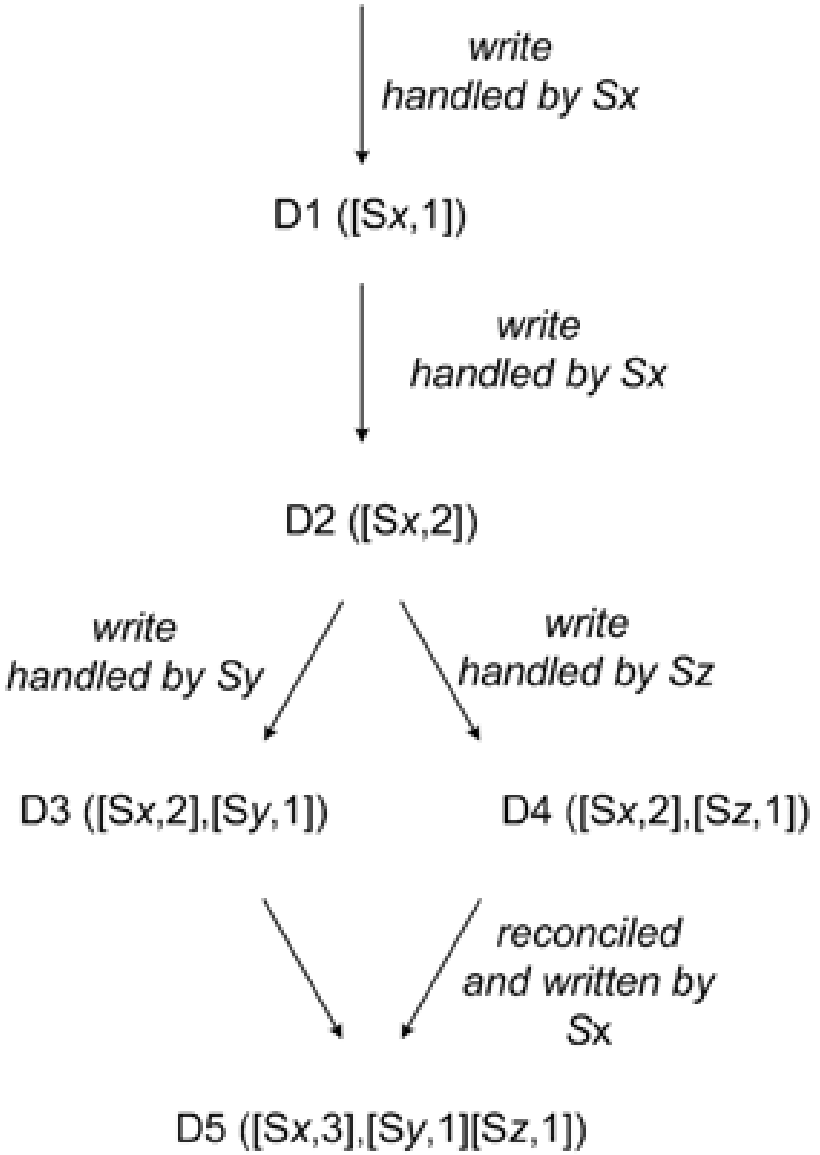


- Запись объекта была обработана узлом Sx

- Версия объекта, со связанными часами

- Перезапись объекта, обработанная узлом Sx

- Версия объекта, с обновленными часами



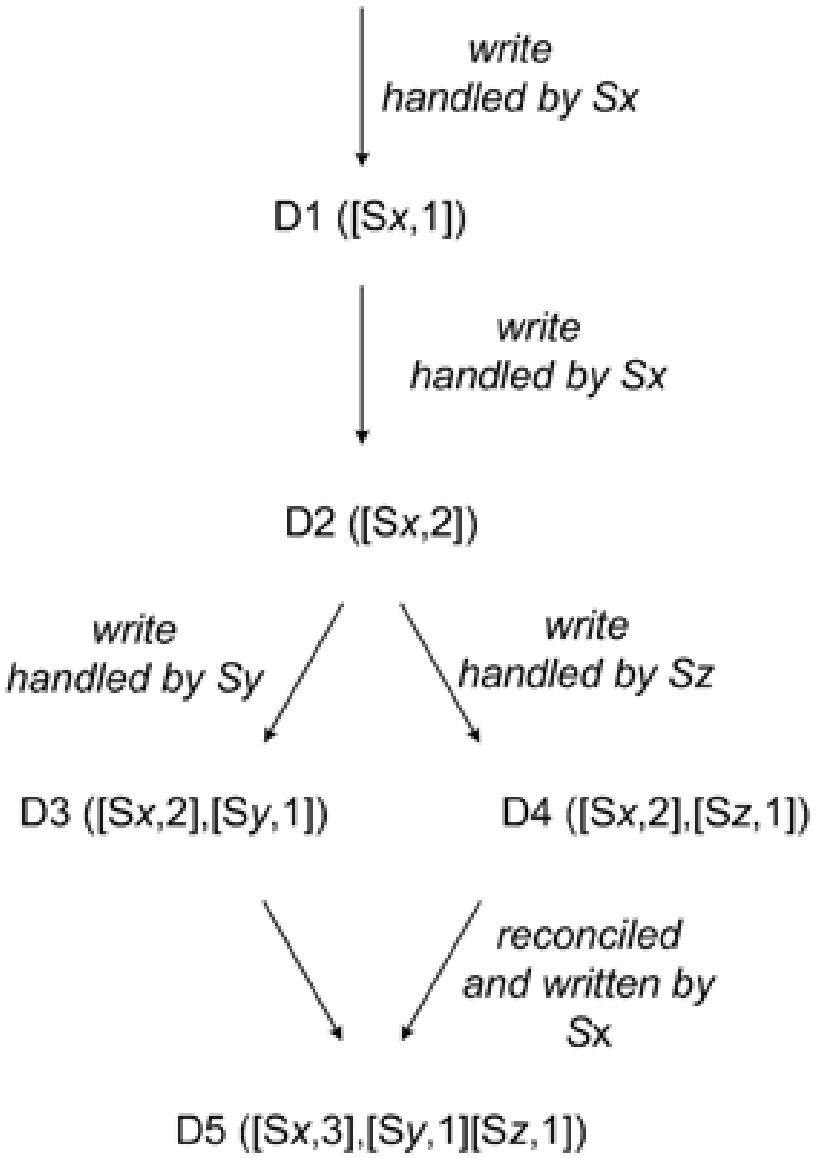
- Запись объекта была обработана узлом Sx

- Версия объекта, со связанными часами

- Перезапись объекта, обработанная узлом Sx

- Версия объекта, с обновленными часами

- Два клиента одновременно делают перезапись
Запросы обрабатываются, соответственно, узлами Sy и Sz



- Запись объекта была обработана узлом Sx

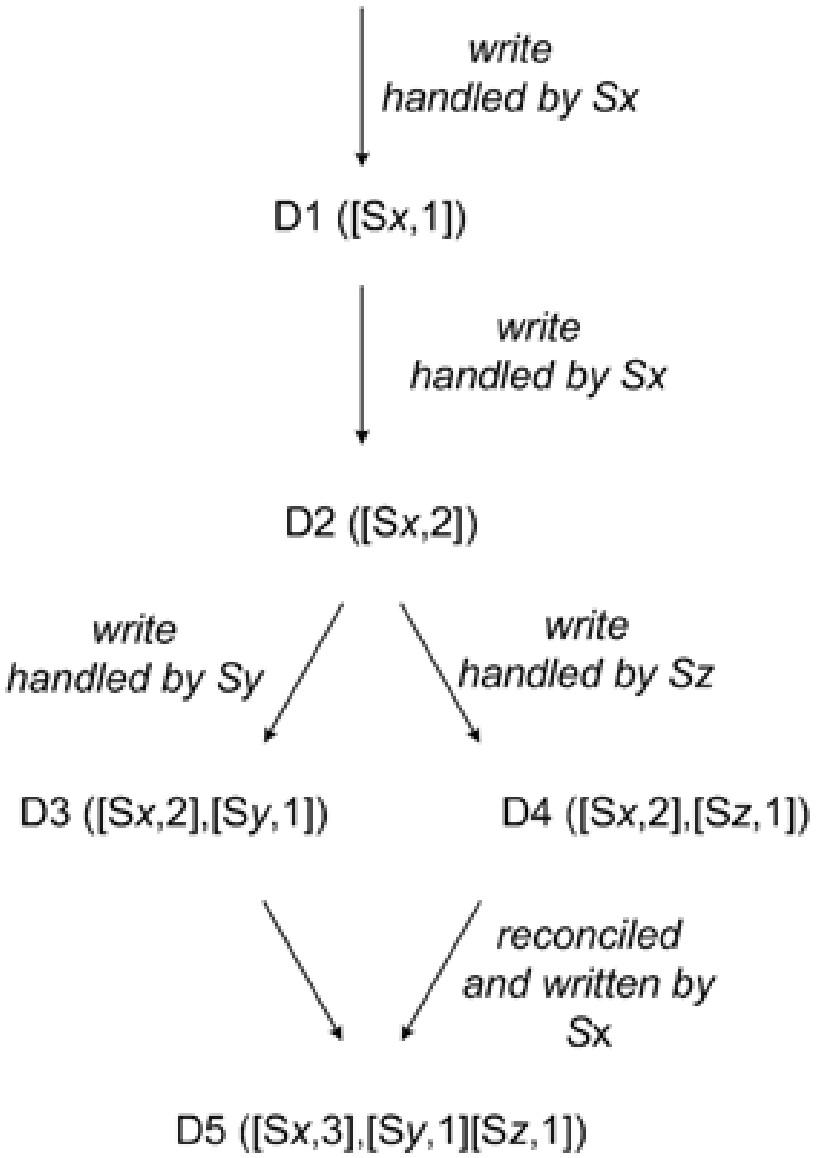
- Версия объекта, со связанными часами

- Перезапись объекта, обработанная узлом Sx

- Версия объекта, с обновленными часами

- Два клиента одновременно делают перезапись
Запросы обрабатываются, соответственно, узлами Sy и Sz

- Две версии объекта, хранящиеся на 2 узлах (Sy и Sz)
Для каждого узла свои часы, но с общей частью [Sx, 2]



- Запись объекта была обработана узлом Sx

- Версия объекта, со связанными часами

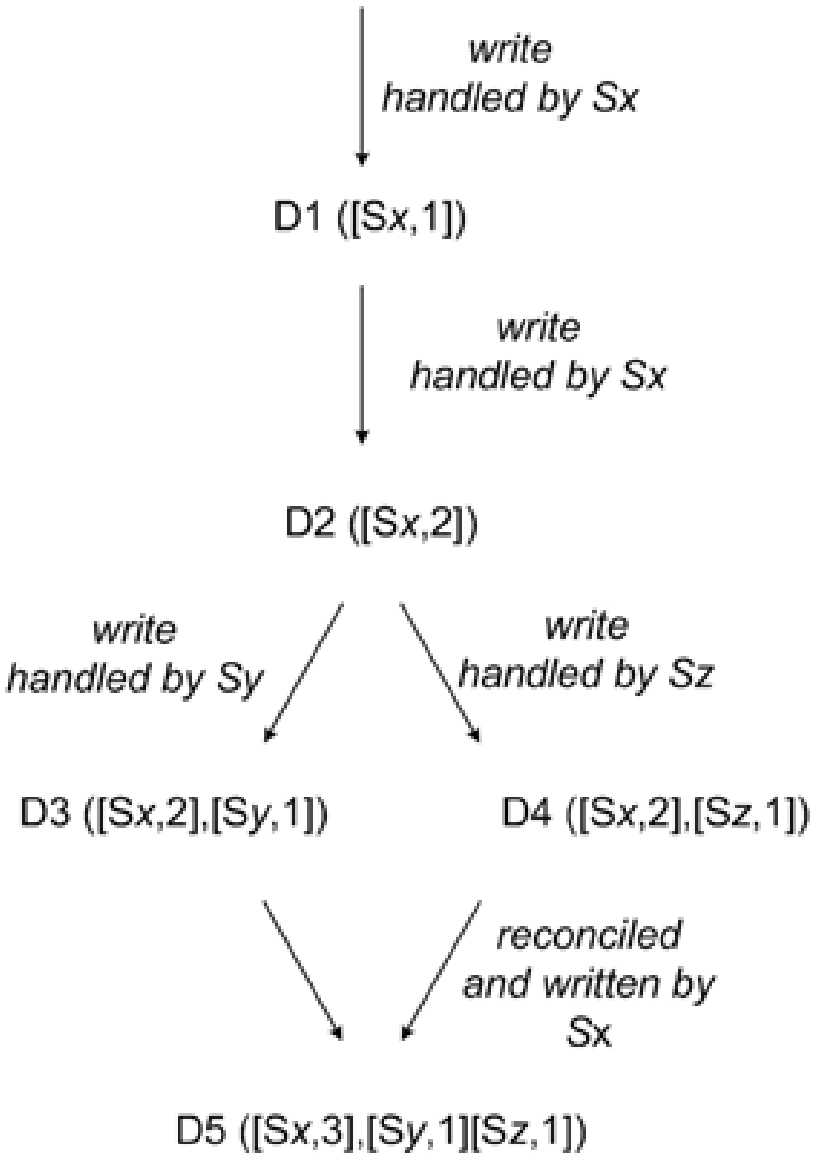
- Перезапись объекта, обработанная узлом Sx

- Версия объекта, с обновленными часами

- Два клиента одновременно делают перезапись
Запросы обрабатываются, соответственно, узлами Sy и Sz

- Две версии объекта, хранящиеся на 2 узлах (Sy и Sz)
Для каждого узла свои часы, но с общей частью [Sx, 2]

- Sx обрабатывает новый запрос на перезапись с суммарными часами для D3 и D4



- Запись объекта была обработана узлом Sx

- Версия объекта, со связанными часами

- Перезапись объекта, обработанная узлом Sx

- Версия объекта, с обновленными часами

- Два клиента одновременно делают перезапись
Запросы обрабатываются, соответственно, узлами Sy и Sz

- Две версии объекта, хранящиеся на 2 узлах (Sy и Sz)
Для каждого узла свои часы, но с общей частью [Sx, 2]

- Sx обрабатывает новый запрос на перезапись с суммарными часами для D3 и D4

- Сохранение новой версии, с согласованными часами

~ Выполнение операций ~

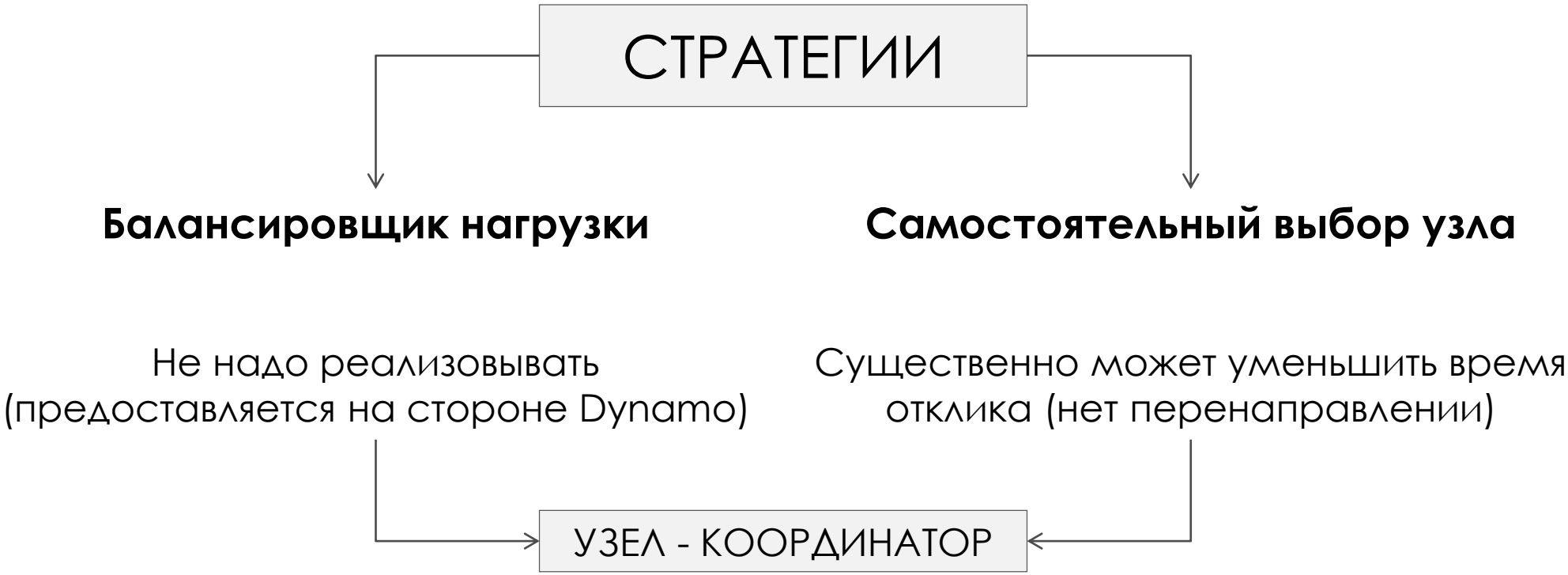
Балансировщик нагрузки

Не надо реализовывать
(предоставляется на стороне Dynamo)

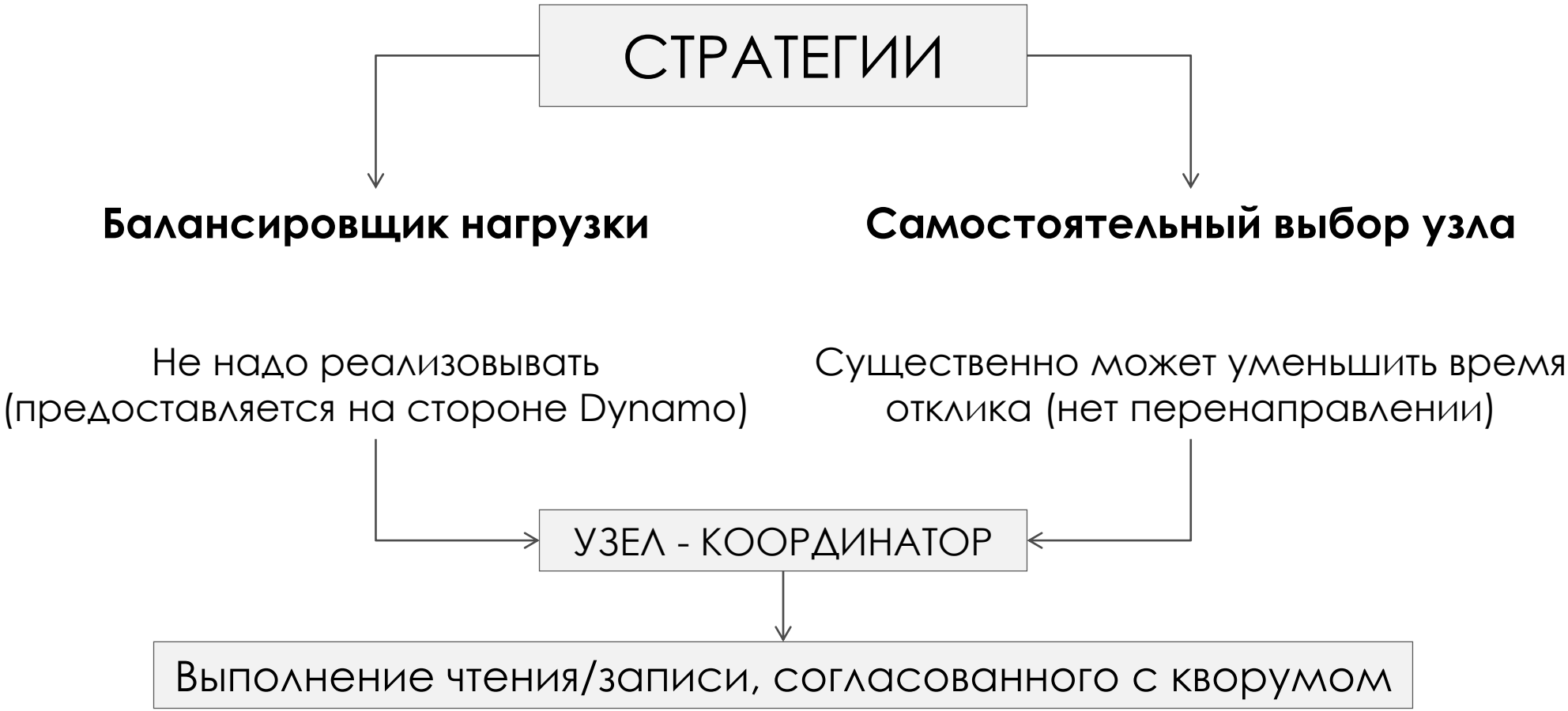
Самостоятельный выбор узла

Существенно может уменьшить время
отклика (нет перенаправлении)

~ Выполнение операций ~



~ Выполнение операций ~



~ Кворум и “*небрежный*” кворум ~

R - параметр системы:
число узлов для кворума чтения

W - параметр системы:
число узлов для кворума записи

~ Кворум и “*небрежный*” кворум ~

R - параметр системы: **число узлов для кворума чтения**

W - параметр системы: **число узлов для кворума записи**

“Небрежный” кворум – вместо опроса всех узлов, опросить только первые **N** доступных узлов из списка предпочтения

Требуемая установка: **W + R > N**

~ Кворум и “*небрежный*” кворум ~

R - параметр системы: **число узлов для кворума чтения**

W - параметр системы: **число узлов для кворума записи**

“Небрежный” кворум – вместо опроса всех узлов, опросить только первые **N** доступных узлов из списка предпочтения

Требуемая установка: **W + R > N**

Действия координатора:

Запрос версий у **N** узлов

Если не **R-1** ответ, то завершение

Возврат всех несвязанных версий

~ Кворум и “*небрежный*” кворум ~

R - параметр системы: **число узлов для кворума чтения**

W - параметр системы: **число узлов для кворума записи**

“Небрежный” кворум – вместо опроса всех узлов, опросить только первые **N** доступных узлов из списка предпочтения

Требуемая установка: **W + R > N**

Действия координатора:

Запрос версий у **N** узлов

Если не **R-1** ответ, то завершение

Возврат всех несвязанных версий

Действия координатора:

Формирование векторных часов

Сохранение локальной копии

Оповещение **N** узлов о перезаписи

Если хотя бы **W-1** ответят, то успех

~ Временная передача ответственности ~

Отключение узла **ND1**

Все запросы отправляются на **N +1** узел в списке предпочтений (**ND2**)

ND2 будет сохранять запрос со специальной пометкой (*hint **ND1***)

~ Временная передача ответственности ~

Отключение узла **ND1**

Все запросы отправляются на **N +1** узел в списке предпочтений (**ND2**)

ND2 будет сохранять запрос со специальной пометкой (*hint **ND1***)

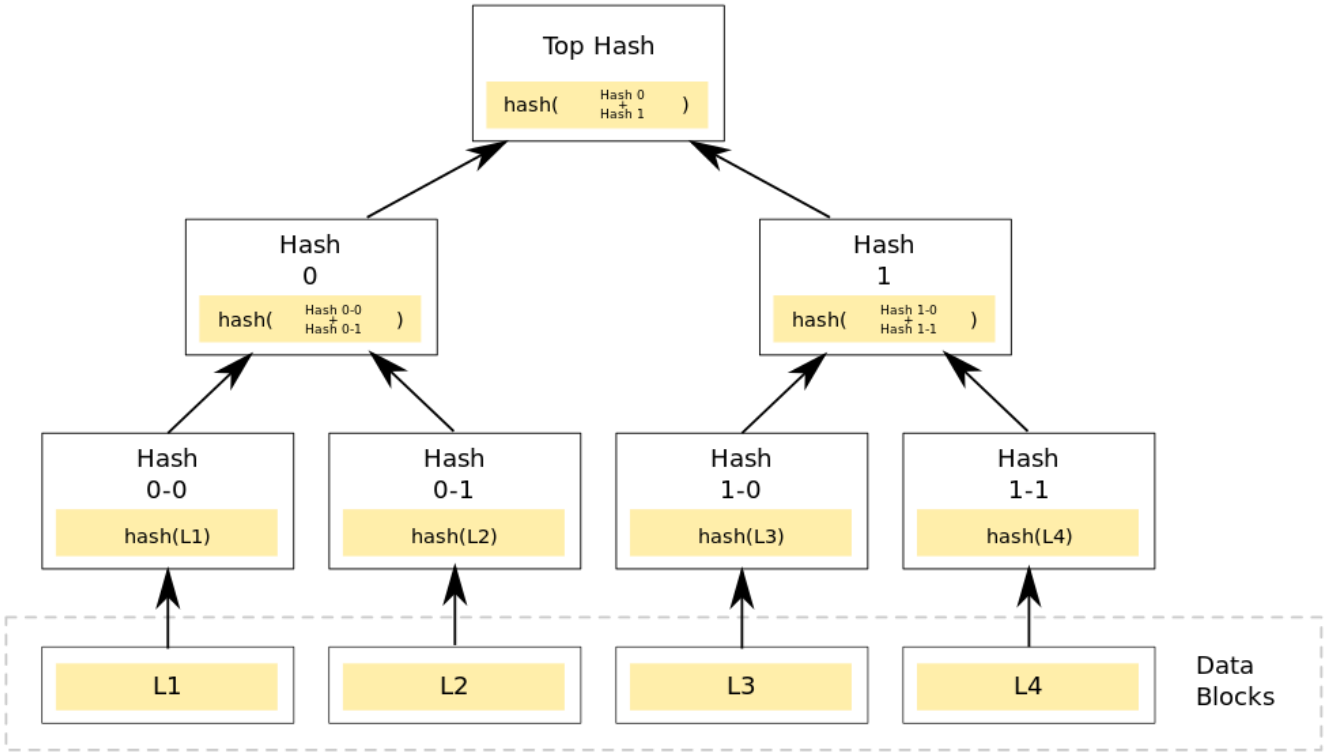
Восстановление узла **ND1**

ND2 будет проверять наличие записей с пометкой “*hint **ND1***”

При наличии записей отправит на **ND1** и удалит из локального хранилища

~ Синхронизация реплик ~

~~ Дерево Меркла (*Merkle tree*) ~~



Двоичное дерево поиска:

Лист – хэш данных
Родитель – хэш от конкатенации хэшей детей

Проверка существования:

O($\log_2$ K), где K – количество блоков данных

~ Синхронизация реплик ~

Каждый **физический** узел содержит дерево Меркла для каждого **виртуального** узла

~ Синхронизация реплик ~

Каждый **физический** узел содержит дерево Меркла для каждого **виртуального** узла

При необходимости синхронизации узлов сравниваются деревья

~ Синхронизация реплик ~

Каждый **физический** узел содержит дерево Меркла для каждого **виртуального** узла

При необходимости синхронизации узлов сравниваются деревья

Вычисляются данные, требующие синхронизации

~ Синхронизация реплик ~

Каждый **физический** узел содержит дерево Меркла для каждого **виртуального** узла

При необходимости синхронизации узлов сравниваются деревья

Вычисляются данные, требующие синхронизации

На основе векторных часов вычисляются актуальные данные и пересылаются с узла на узел

~ Членство в кольце ~

Узел отправлен на техобслуживание **!=** выведен навсегда

~ Членство в кольце ~

Узел отправлен на техобслуживание **!=** выведен навсегда

Явно обозначена процедура добавления и удаления
узлов из кольца (выполняется администратором)

~ Членство в кольце ~

Узел отправлен на техобслуживание **!=** выведен навсегда

Явно обозначена процедура добавления и удаления узлов из кольца (выполняется администратором)

Логирование событий введения/выведения узлов

~ Членство в кольце ~

Узел отправлен на техобслуживание **!=** выведен навсегда

Явно обозначена процедура добавления и удаления узлов из кольца (выполняется администратором)

Логирование событий введения/выведения узлов

Распространение информации “о членстве” методом “слухов” и согласуется в “конечном итоге”

~ Членство в кольце ~

Узел отправлен на техобслуживание **!=** выведен навсегда

Явно обозначена процедура добавления и удаления узлов из кольца (выполняется администратором)

Логирование событий введения/выведения узлов

Распространение информации “о членстве” методом “слухов” и согласуется в “конечном итоге”

Раз в секунду узел связывается со случайным соседом и согласует сохранённую историю изменений

~ Членство в кольце ~

Если узел первый раз вводится в эксплуатацию, то он генерирует токены, отвечающие за диапазоны сегментирования

~ Членство в кольце ~

Если узел первый раз вводится в эксплуатацию, то он генерирует токены, отвечающие за диапазоны сегментирования

Токены вместе с синхронизацией изменений в членстве переносятся между узлами методом “слухов”

~ Членство в кольце ~

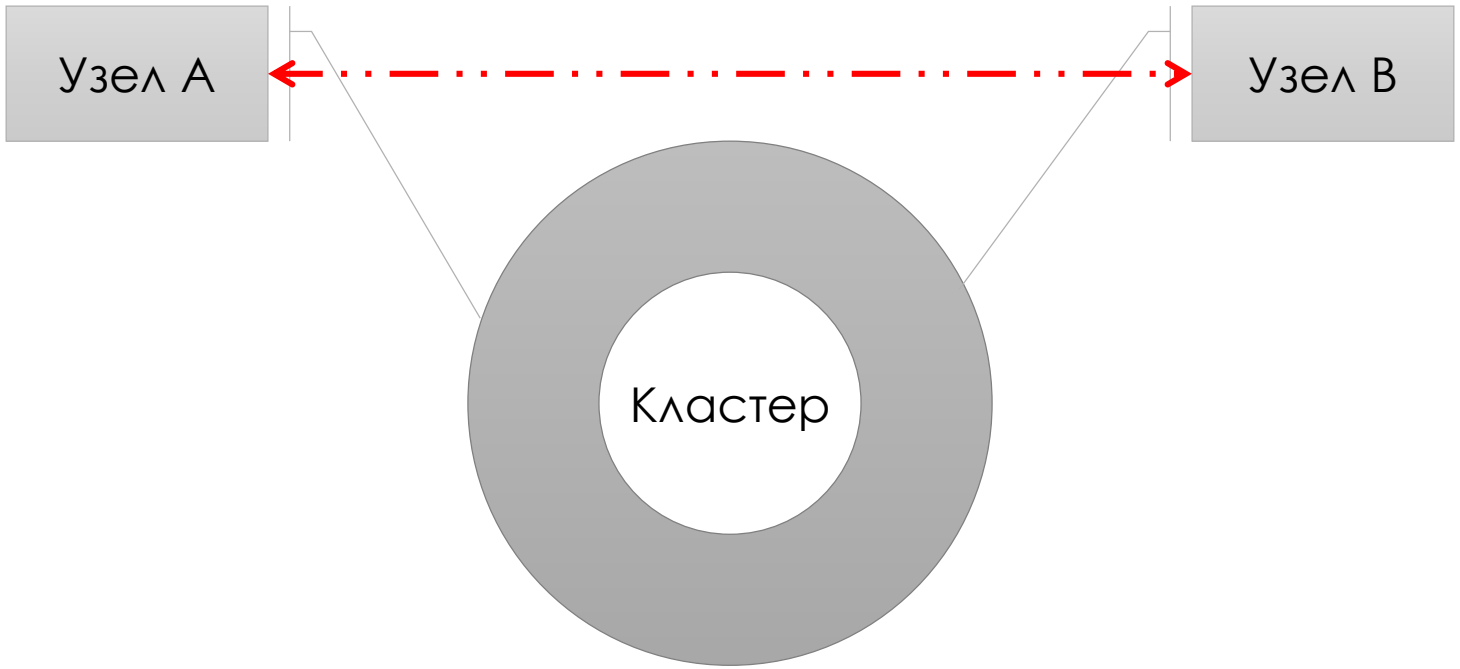
Если узел первый раз вводится в эксплуатацию, то он генерирует токены, отвечающие за диапазоны сегментирования

Токены вместе с синхронизацией изменений в членстве переносятся между узлами методом “слухов”

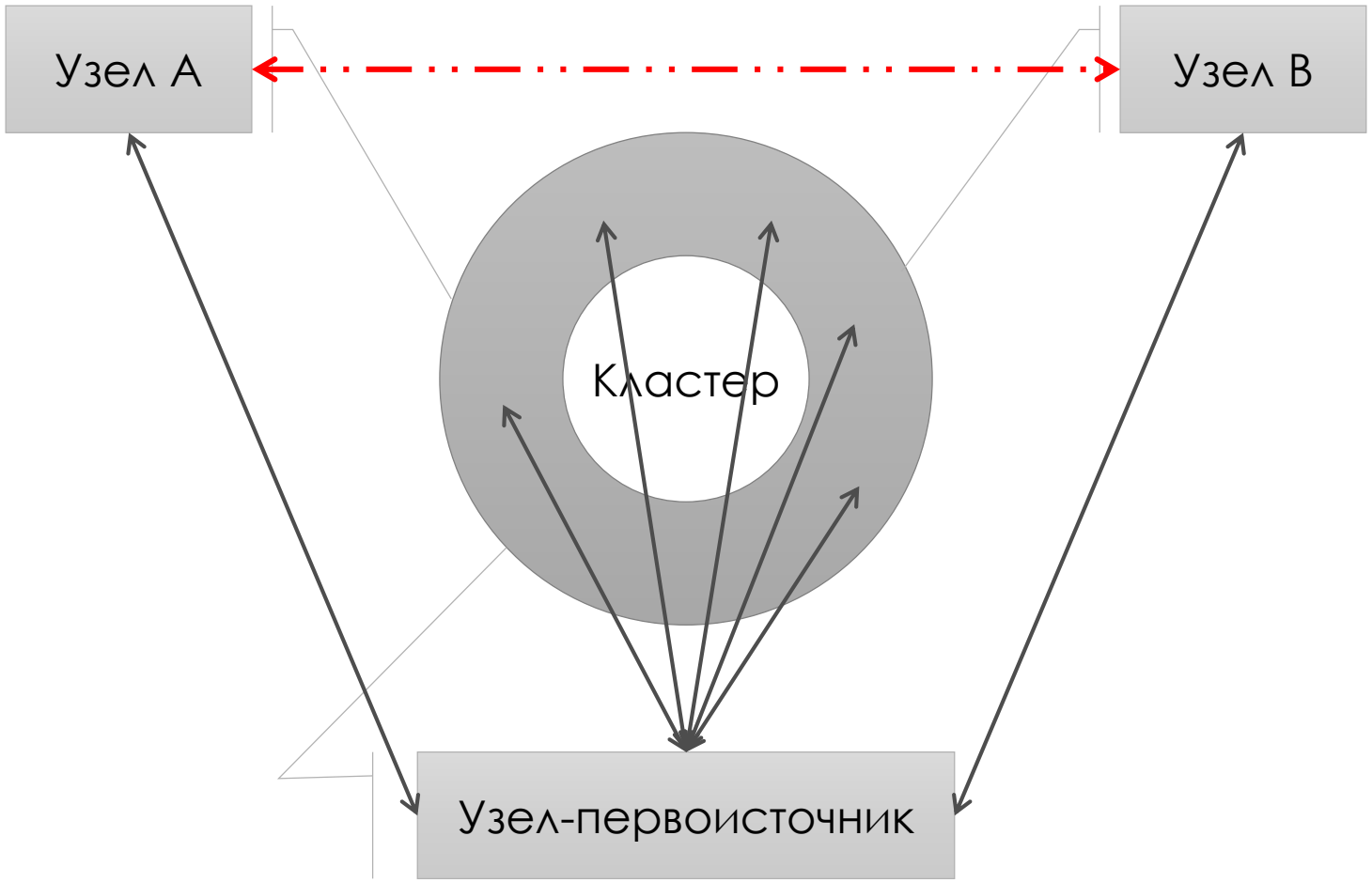
Каждый узел знает о каждом узле хранимые диапазоны

~ Внешнее обнаружение ~

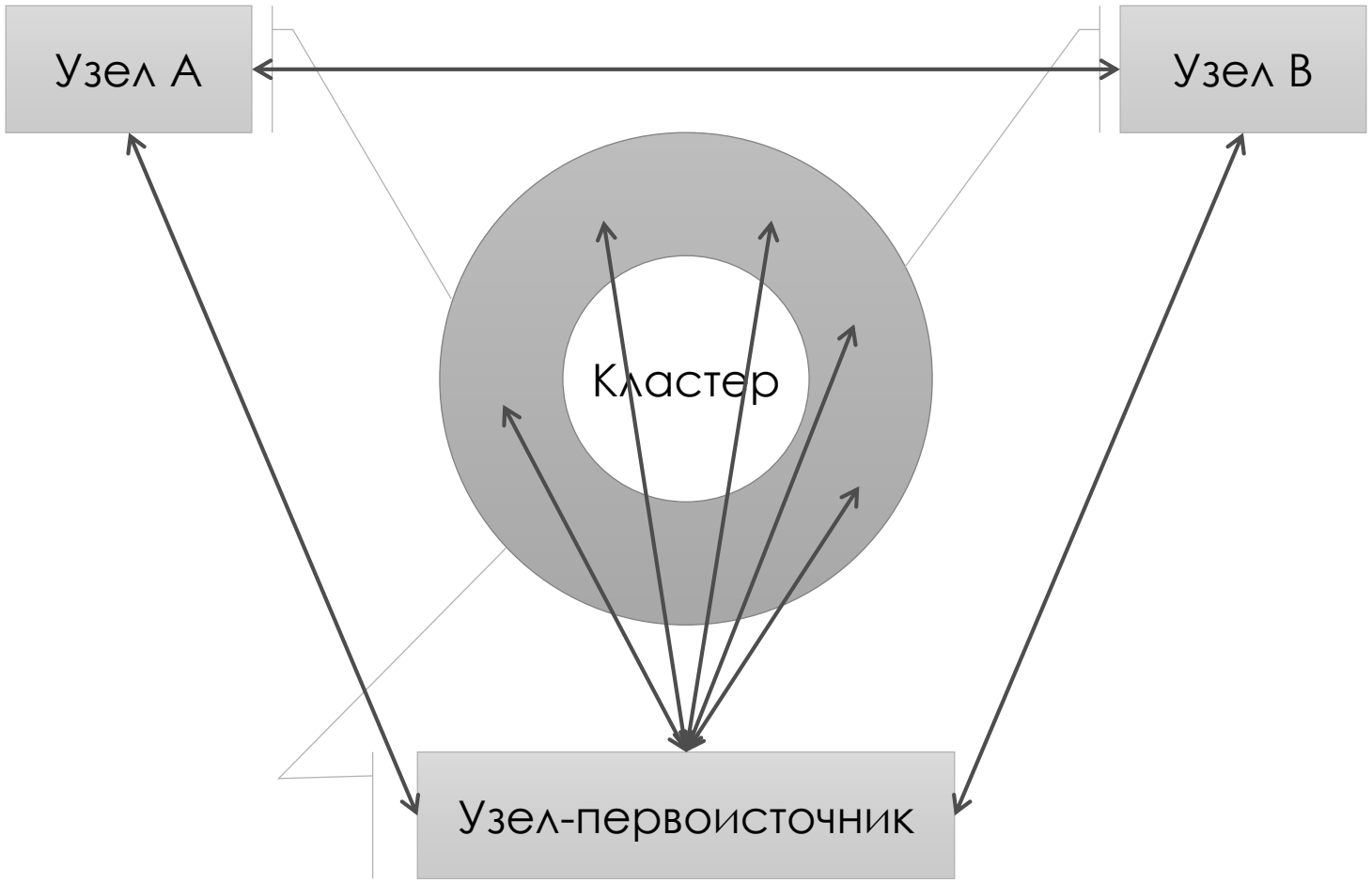
~ Внешнее обнаружение ~



~ Внешнее обнаружение ~



~ Внешнее обнаружение ~



~ Обнаружение неисправностей ~

Узел **B** неисправен для **A** -- **B** не отвечает на запросы **A**

~ Обнаружение неисправностей ~

Узел **B** неисправен для **A** -- **B** не отвечает на запросы **A**

Пока **B** неисправен для **A**, узел **A** общается с узлами из списка приоритетов, для данного сегмента

Если нет клиентских запросов, то взаимодействия между узлами нет

Если запросы есть, то **A** периодически обновляет состояние **B** для себя

~ Обнаружение неисправностей ~

Узел **B** неисправен для **A** -- **B** не отвечает на запросы **A**

Пока **B** неисправен для **A**, узел **A** общается с узлами из списка приоритетов, для данного сегмента

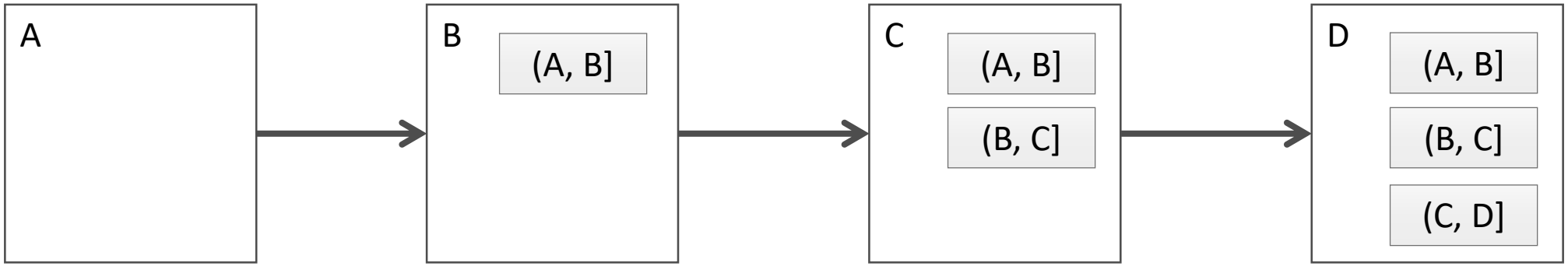
Если нет клиентских запросов, то взаимодействия между узлами нет

Если запросы есть, то **A** периодически обновляет состояние **B** для себя

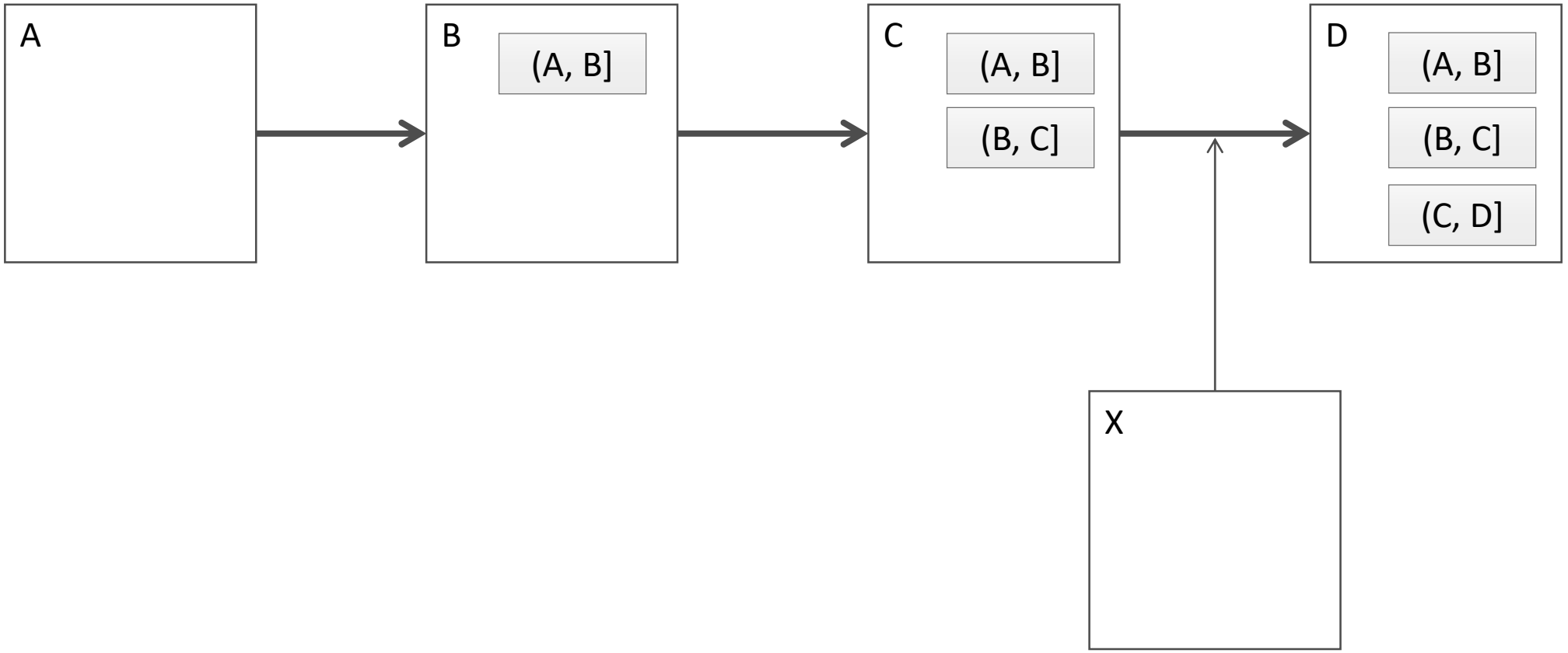
↓

Каждый узел знает о присоединении и выходе других

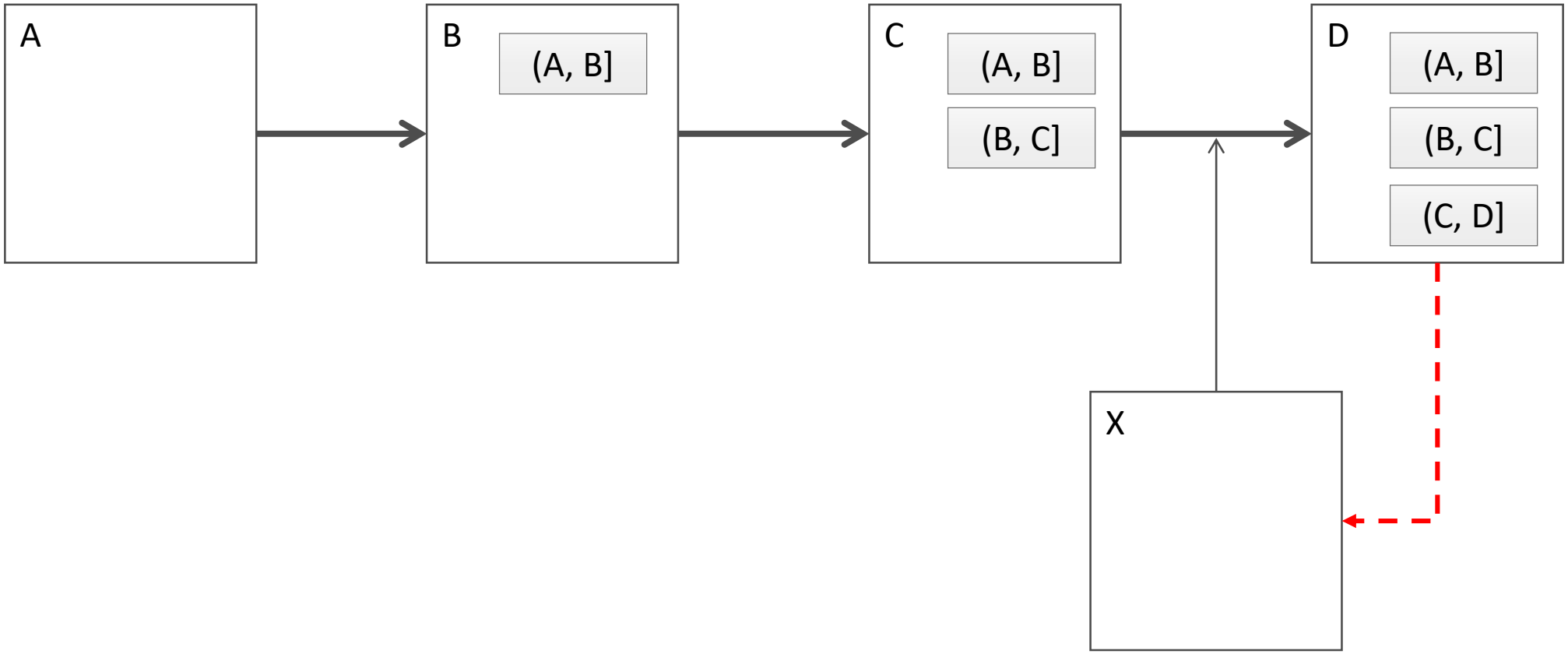
~ Добавление узла хранилища ~



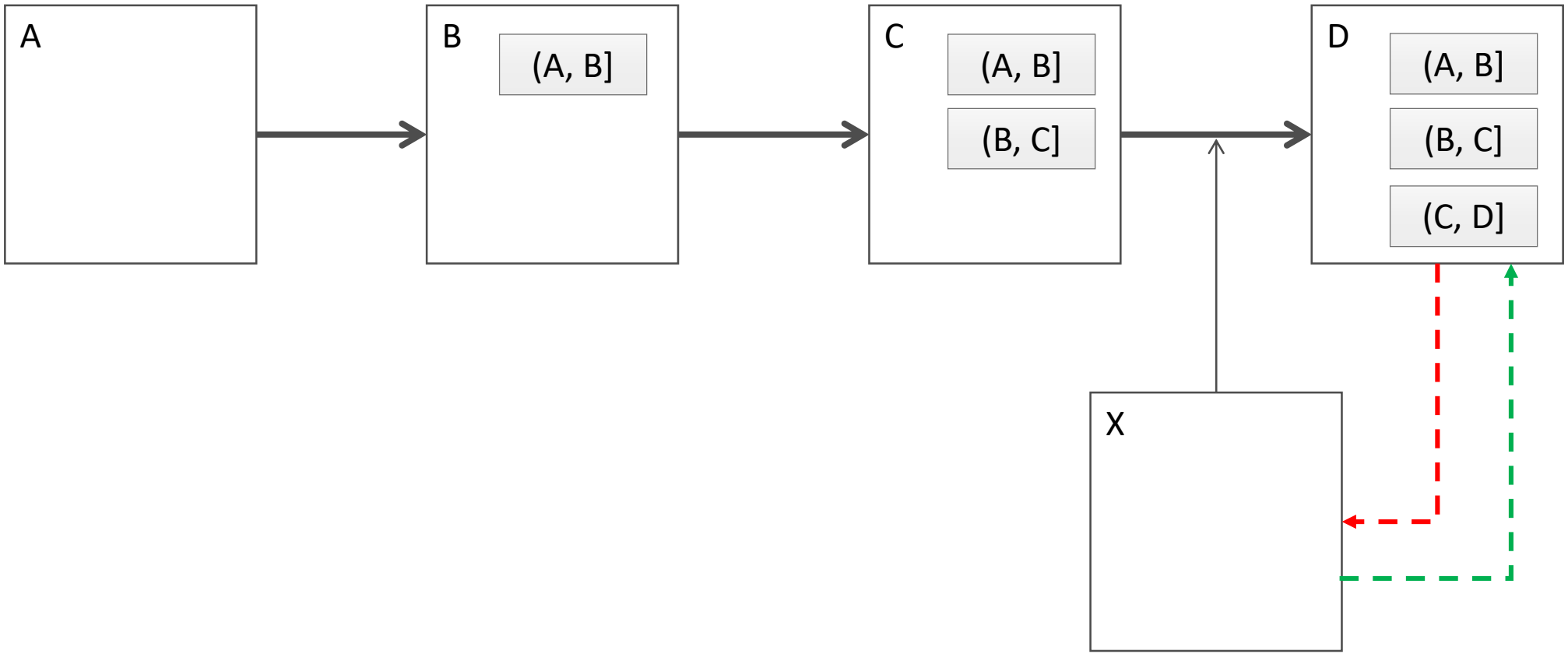
~ Добавление узла хранилища ~



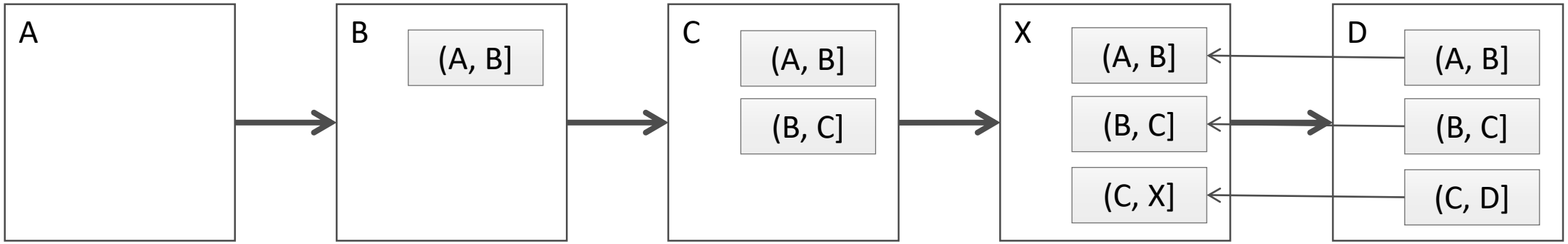
~ Добавление узла хранилища ~



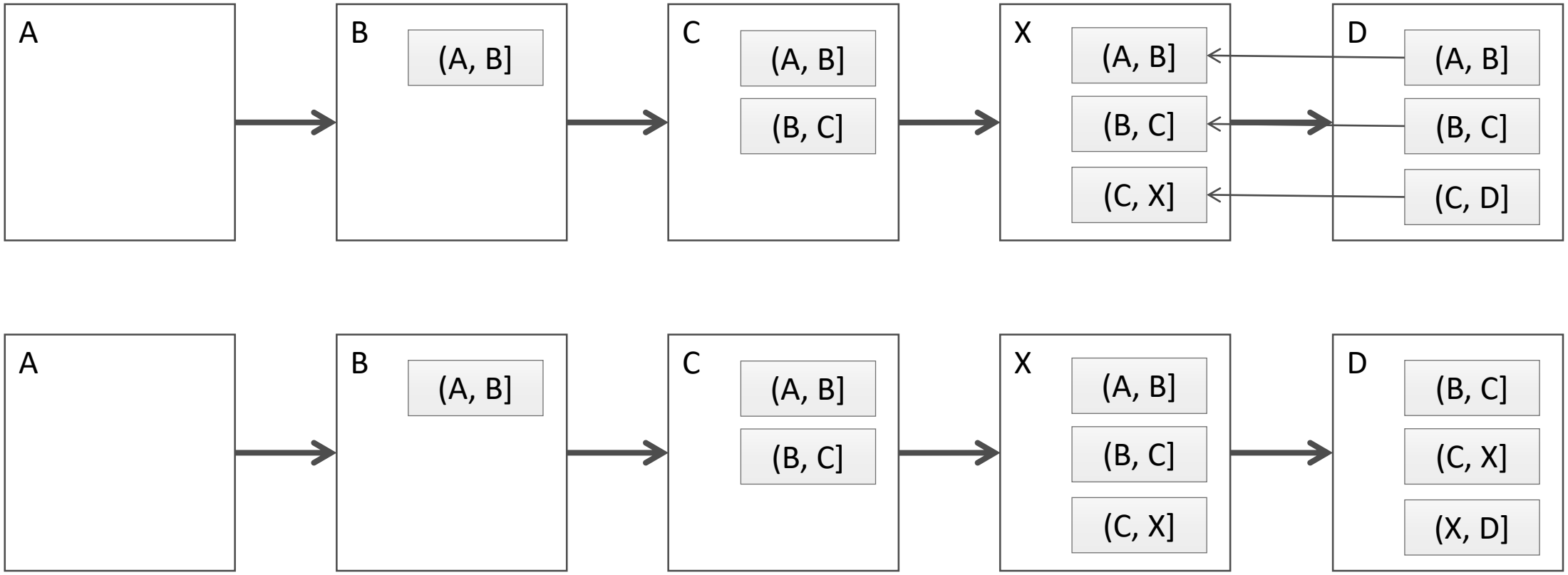
~ Добавление узла хранилища ~



~ Добавление узла хранилища ~



~ Добавление узла хранилища ~



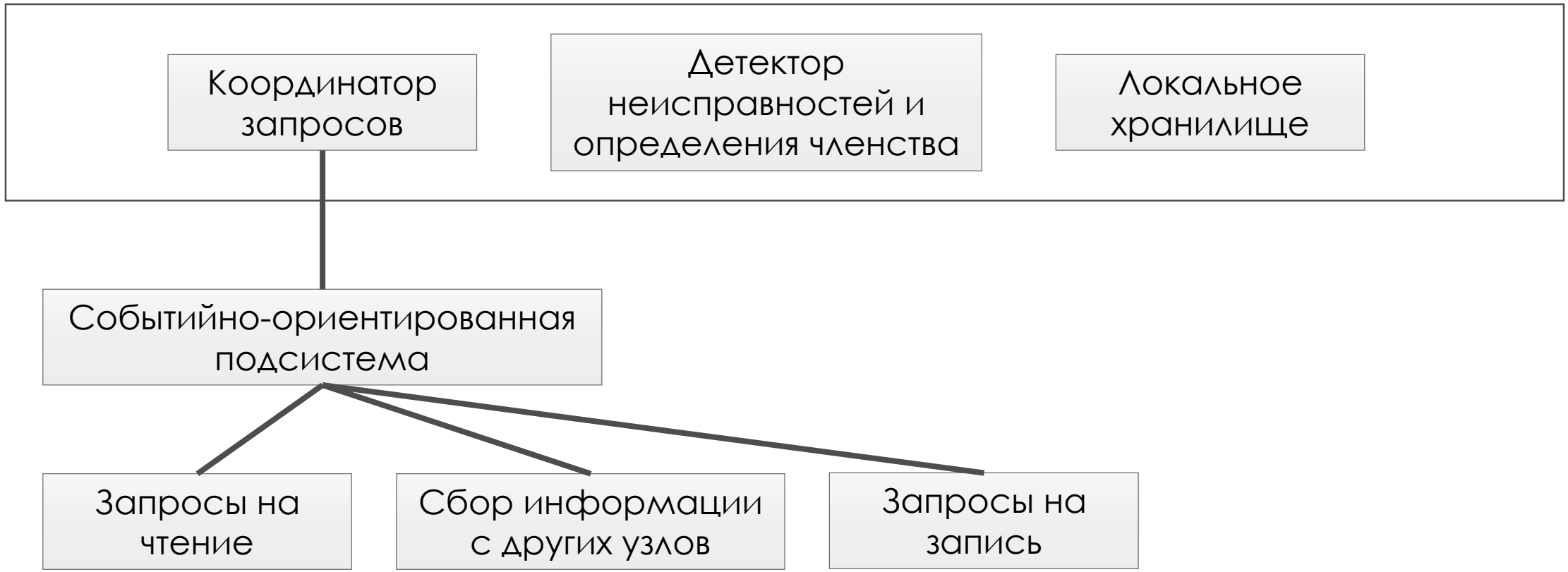
- Реализация -

наконец-то добрались до пирога с котятами

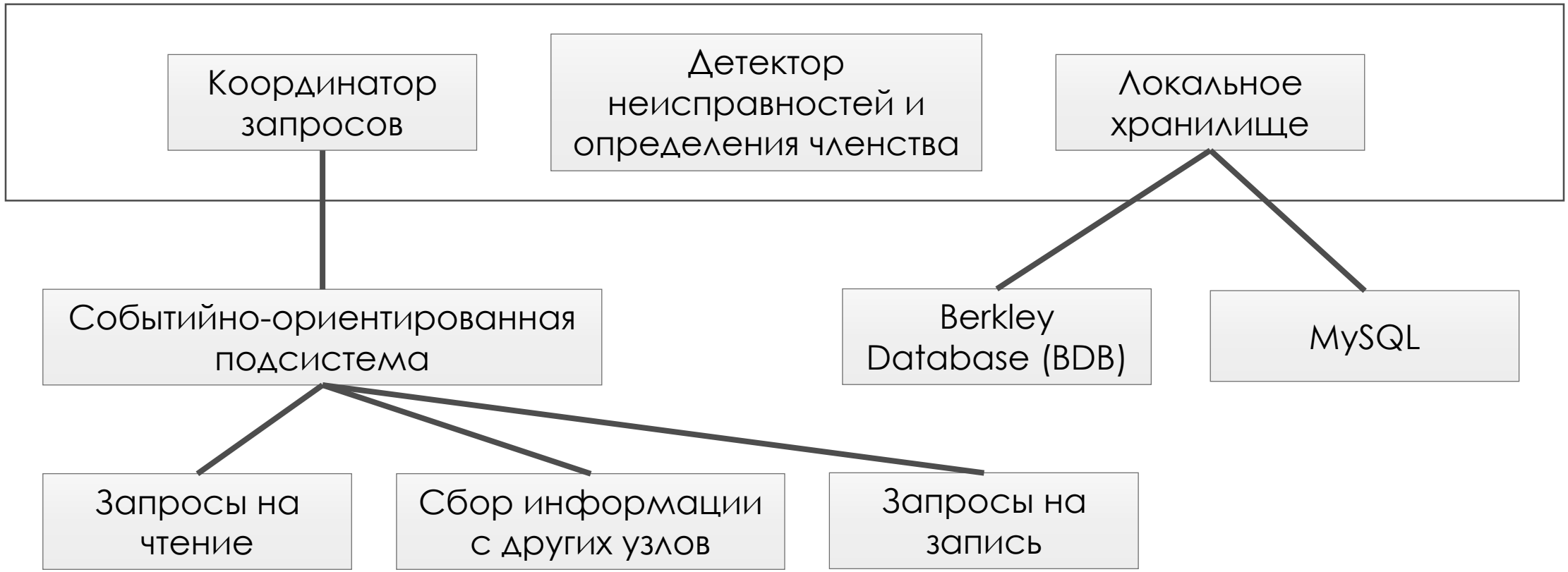
~ Устройство узла ~

~ Устройство узла ~

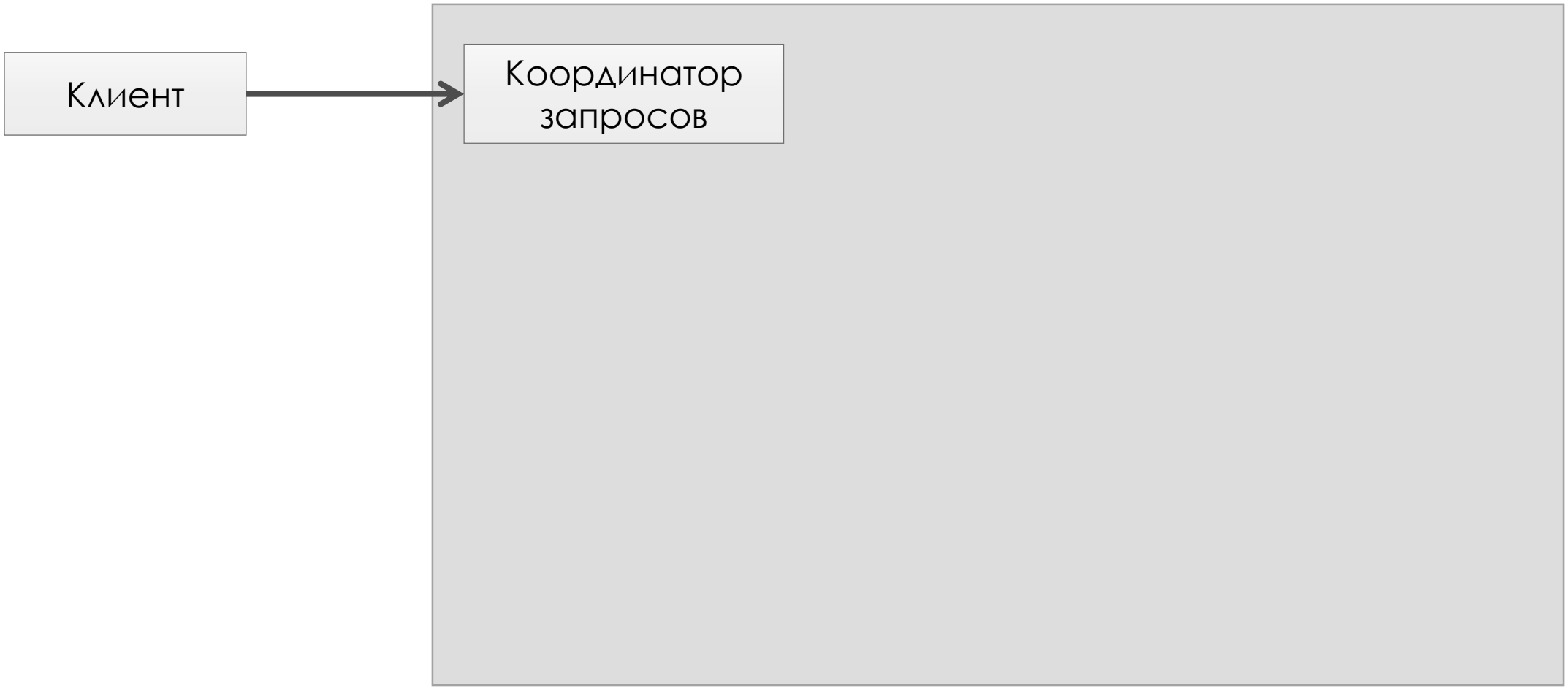
~ Устройство узла ~



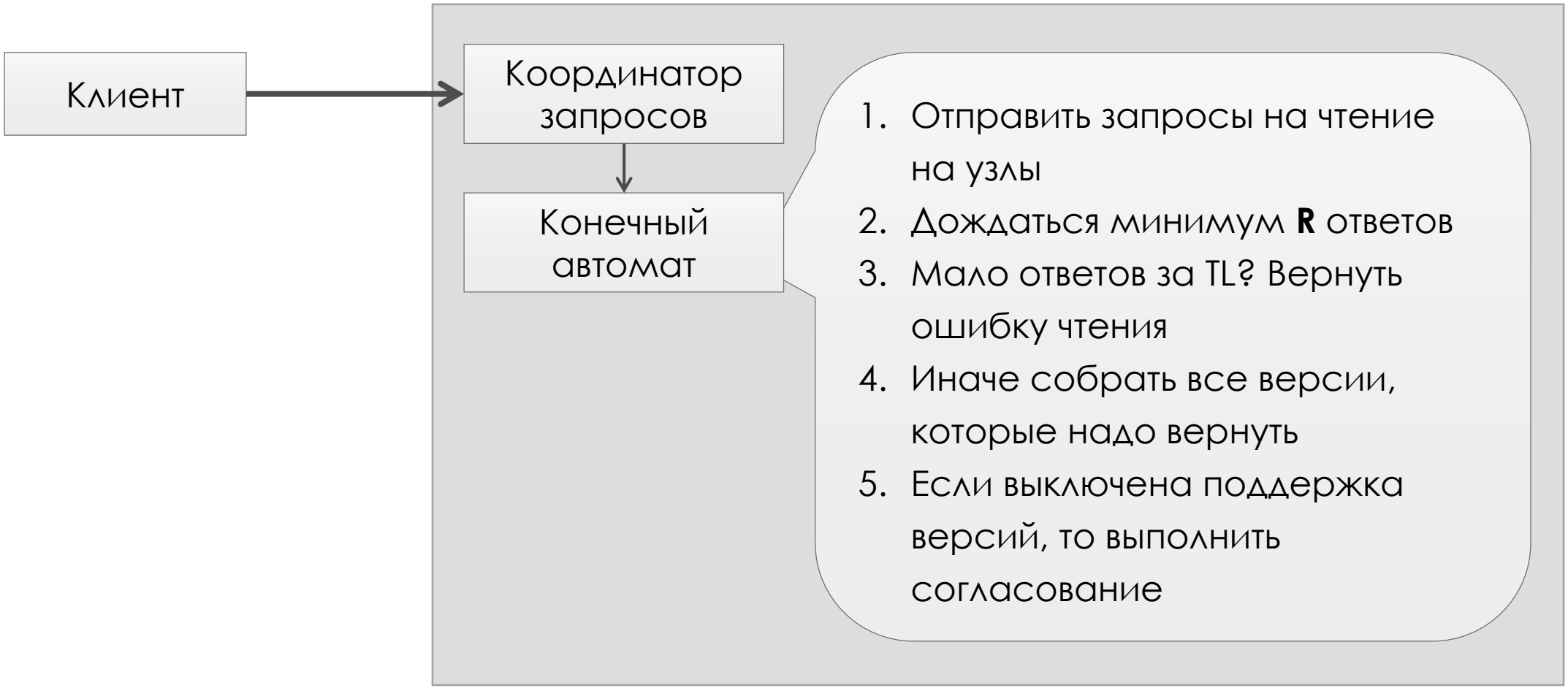
~ Устройство узла ~



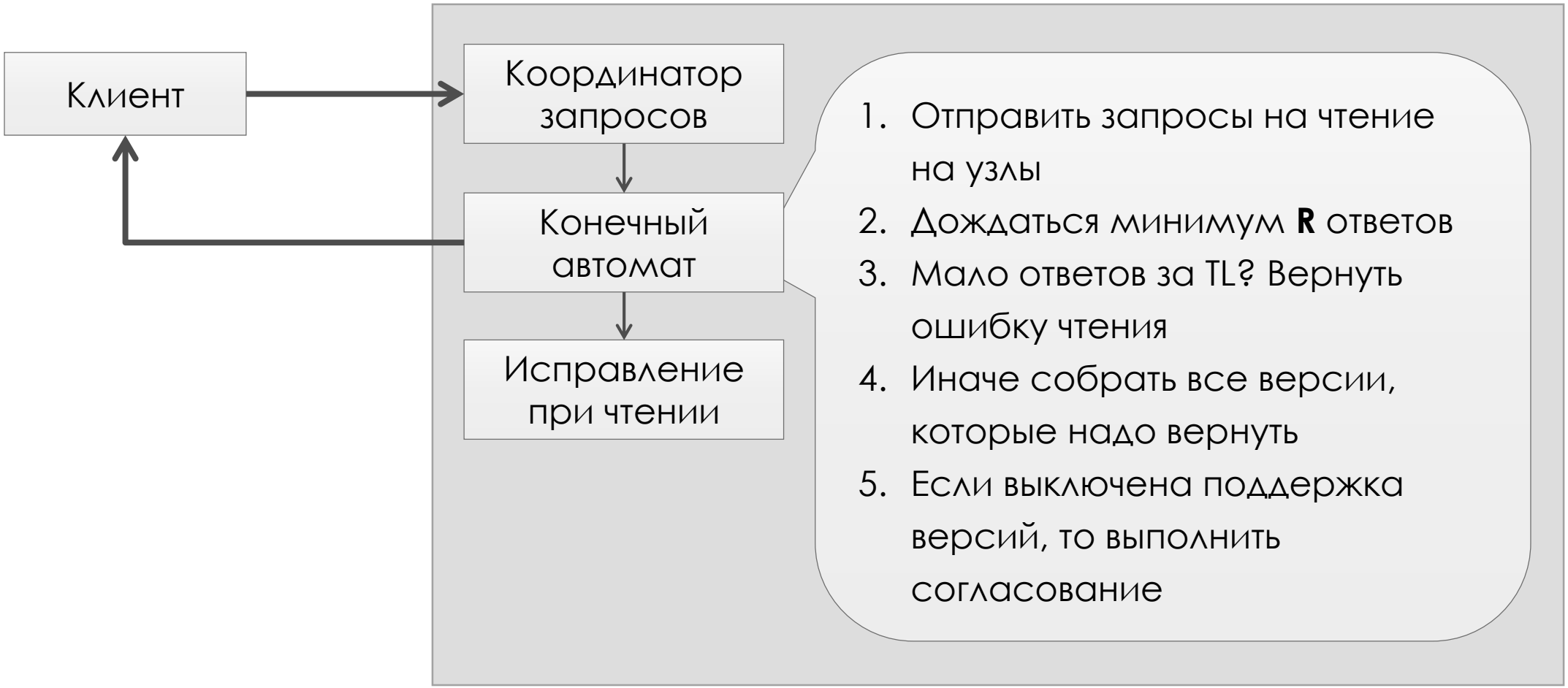
~ Взаимодействие с клиентом ~



~ Взаимодействие с клиентом ~



~ Взаимодействие с клиентом ~



- Опыт эксплуатации -

“Ой! да ну его в … продакшн”

~ Типичные варианты конфигурации ~

Согласование в бизнес логике

Объект реплицируется между несколькими узлами

Логика согласования реализуется в клиентском приложении

(Пример: корзина в онлайн-магазине)

Согласование по времени

Объект реплицируется между несколькими узлами

Согласование на стороне Dynamo по принципу “побеждает последний”

(Пример: приложение, отвечающее за клиентские сессии)

Система с быстрым чтением

Нетипичные параметры:
R = 1, **W** = **N**

Объект реплицируется между несколькими узлами

(Пример: каталог товаров)

~ Типичные значения (**N**, **R**, **W**) ~

~ Типичные значения (**N**, **R**, **W**) ~

N	**R**	**W**
3	**2**	**2**

- Широко используемая на практике конфигурация *(получена опытным путём)*

~ Типичные значения (**N**, **R**, **W**) ~

N	**R**	**W**
3	2	2
C	C	1

- Широко используемая на практике конфигурация *(получена опытным путём)*
- Гарантированно записываем изменения в ущерб согласованию версии

~ Типичные значения (**N**, **R**, **W**) ~

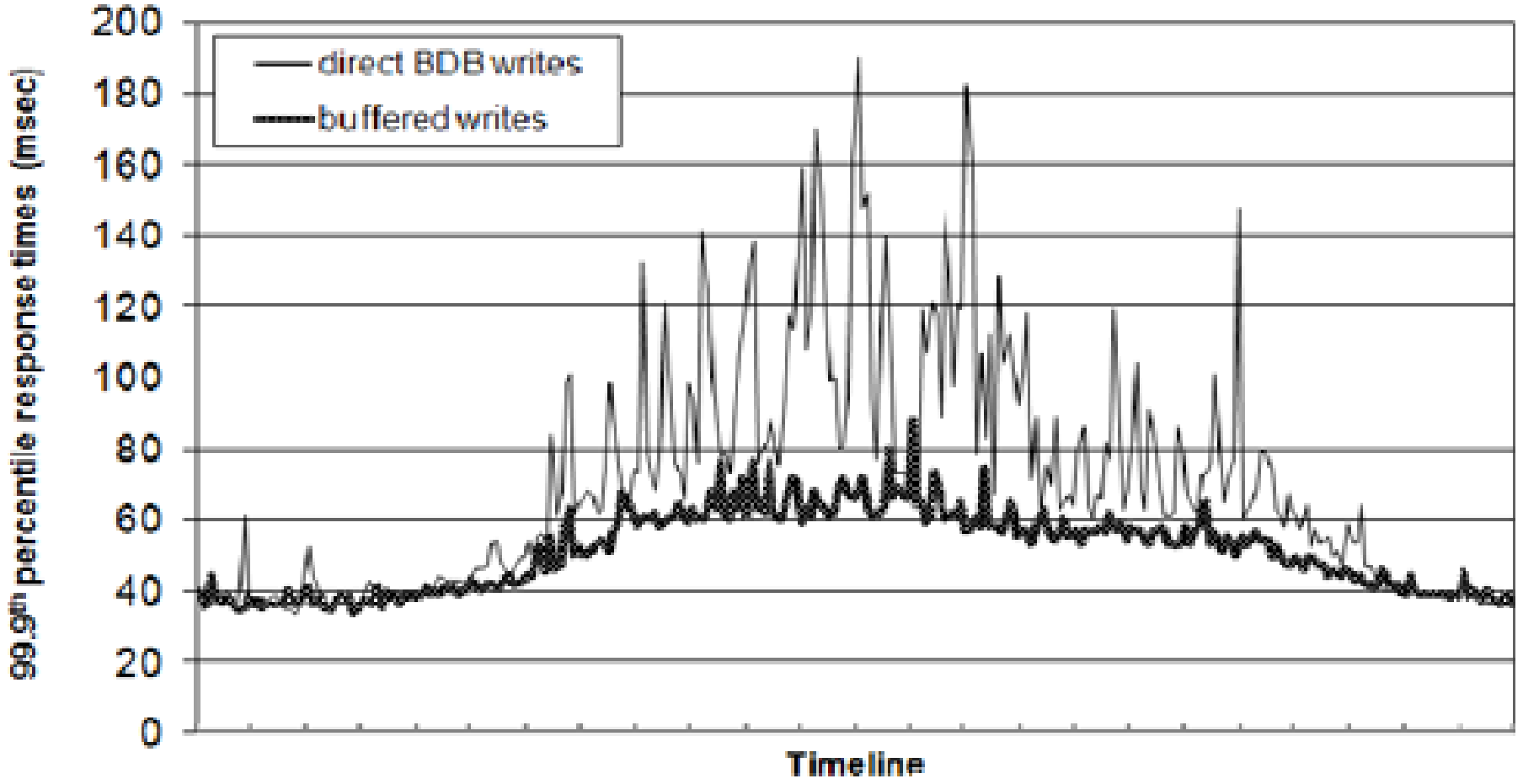
N	R	W
3	2	2
C	C	1
C	1	C

- Широко используемая на практике конфигурация *(получена опытным путём)*
- Гарантированно записываем изменения в ущерб согласованию версии
- Быстрое чтение, но жертвуем высокой доступностью

~ Буфер записи и кэш ~

(Поддерживается всеми узлами, но используется при необходимости)

Производительность	⟷	**Надёжность**
Буфер в оперативной памяти, через который происходят запросы на запись/чтение		При незапланированном выключении узла данные могут быть потеряны, т.к. не были записаны на диск
Периодически сбрасывается на диск фоновым записывающим потоком		Для уменьшения рисков используют “надёжную запись”
Снижение времени отклика **x5**		Не влияет на время отклика



Сравнение времени отклика при буферизированной записи и прямой записи для 99.9-й процентили выборки в течение 24 часов.

~ Равномерная нагрузка ~

Равномерное **распределение ключей**, но неравномерное **распределение запросов** по определённым ключам

~ Равномерная нагрузка ~

Равномерное **распределение ключей**, но неравномерное **распределение запросов** по определённым ключам

Сбалансированный узел – количество запросов на данный узел отклонялось от среднего значения на величину меньше определённого порога (здесь на – 15%)

~ Равномерная нагрузка ~

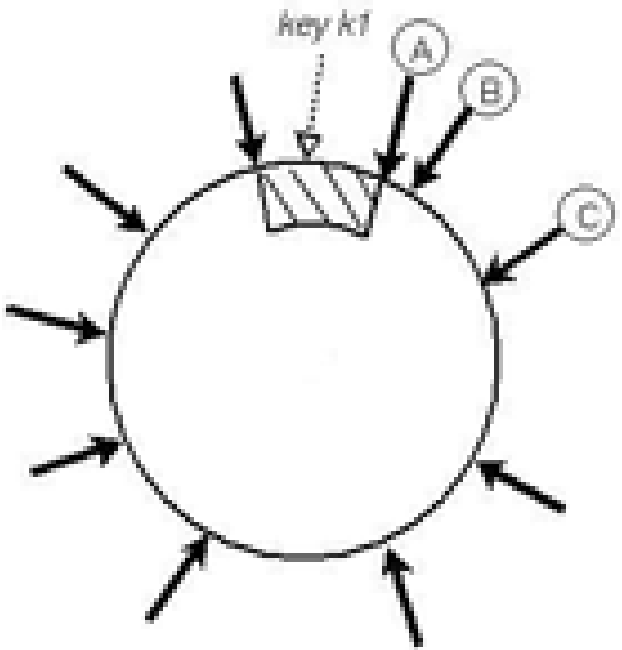
Равномерное **распределение ключей**, но неравномерное **распределение запросов** по определённым ключам

Сбалансированный узел – количество запросов на данный узел отклонялось от среднего значения на величину меньше определённого порога (здесь на – 15%)

Способ уменьшения “несбалансированных” узлов - сегментация

~ Стратегия 1 ~

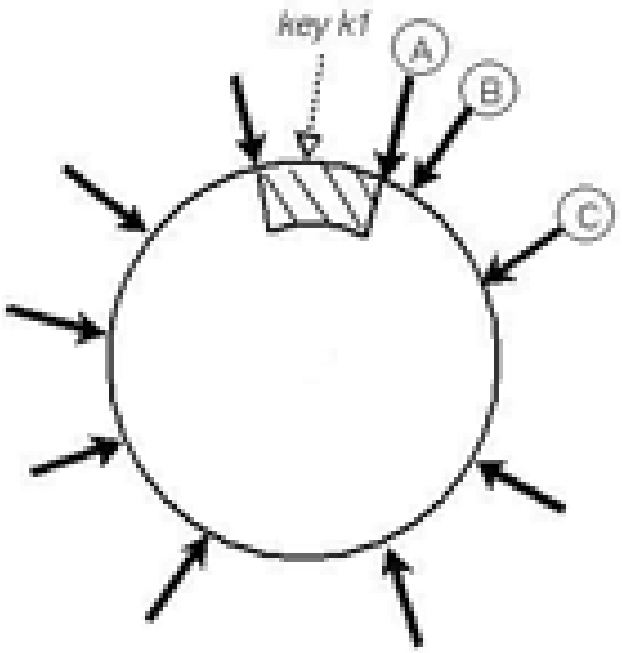
~~ T случайных токенов и разбиение по значению ~~



Strategy 1

~ Стратегия 1 ~

~~ T случайных токенов и разбиение по значению ~~

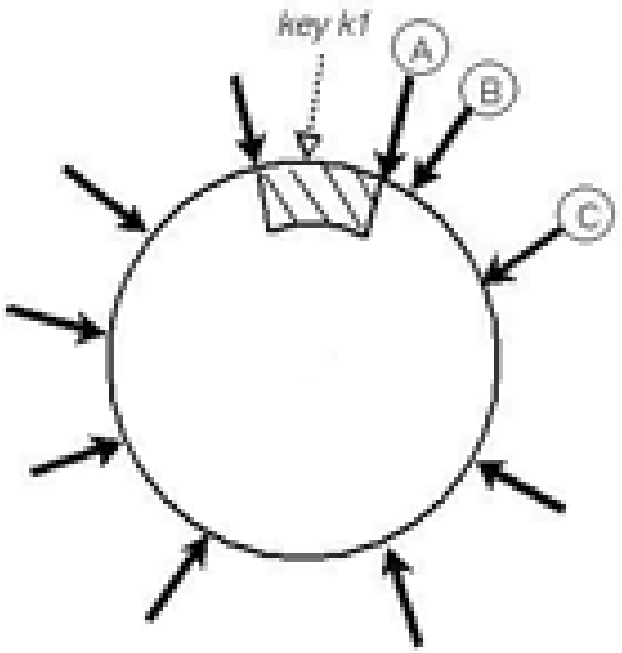


Strategy 1

Токены всех узлов упорядочены по значению

~ Стратегия 1 ~

~~ T случайных токенов и разбиение по значению ~~



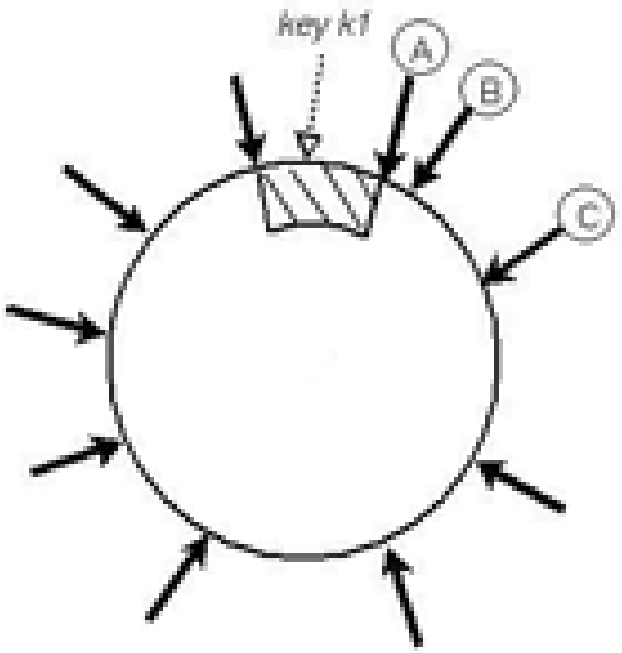
Strategy 1

Токены всех узлов упорядочены по значению

Разные размеры диапазонов из-за случайного выбора токенов

~ Стратегия 1 ~

~~ T случайных токенов и разбиение по значению ~~



Strategy 1

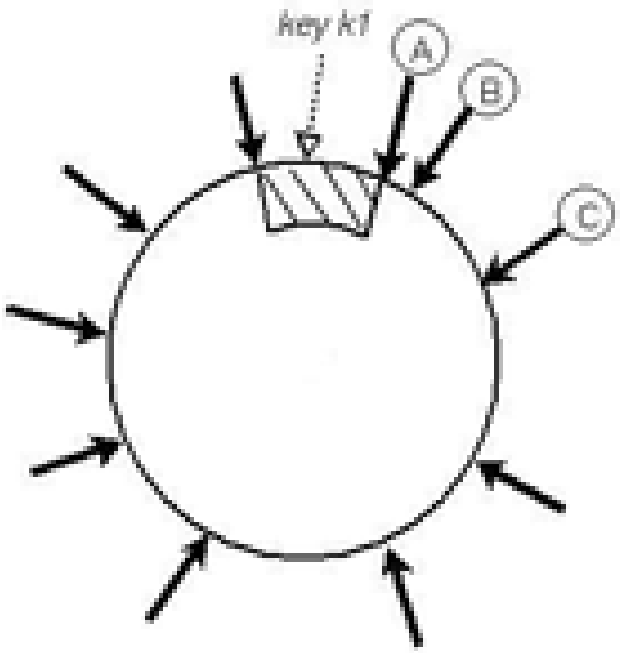
Токены всех узлов упорядочены по значению

Разные размеры диапазонов из-за случайного выбора токенов

Множество токенов и размеры диапазонов меняются при подключении/отключении нового узла

~ Стратегия 1 ~

~~ T случайных токенов и разбиение по значению ~~



Strategy 1

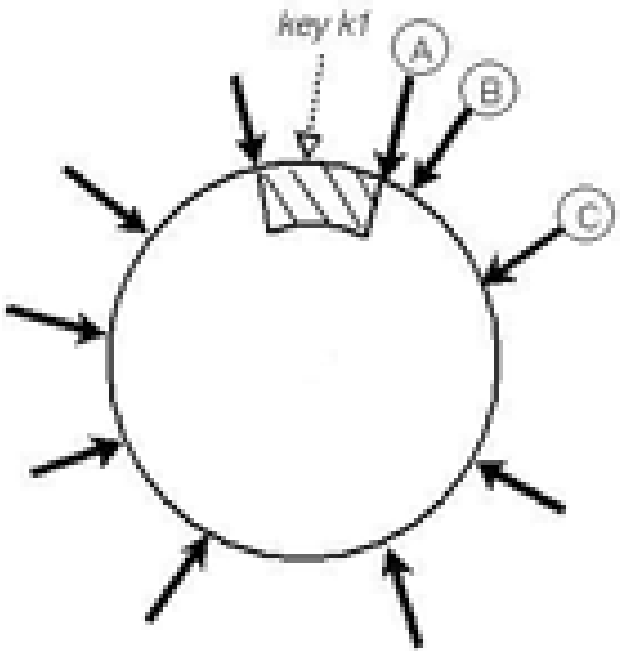
Токены всех узлов упорядочены по значению

Разные размеры диапазонов из-за случайного выбора токенов

Множество токенов и размеры диапазонов меняются при подключении/отключении нового узла

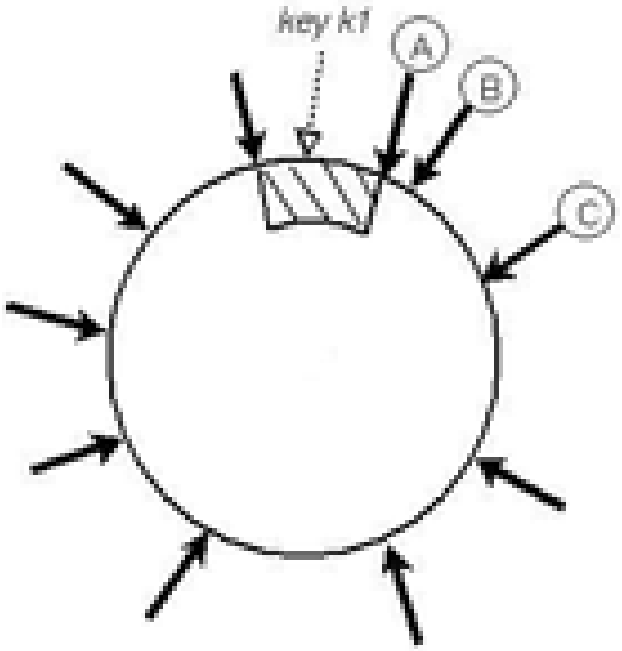
Линейный рост данных с информацией о членстве на каждом узле

~ Проблемы стратегии 1 ~



Strategy 1

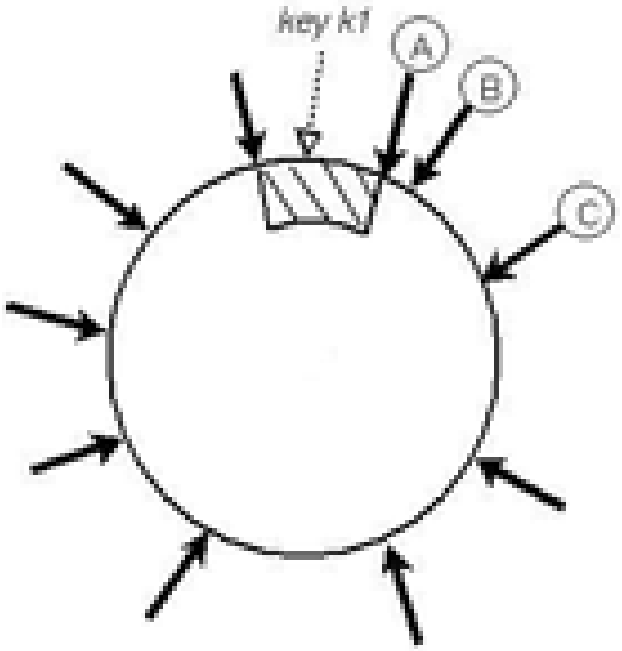
~ Проблемы стратегии 1 ~



Strategy 1

Для передачи диапазонов ключей узлу необходимо было выполнить сканирование хранилища

~ Проблемы стратегии 1 ~

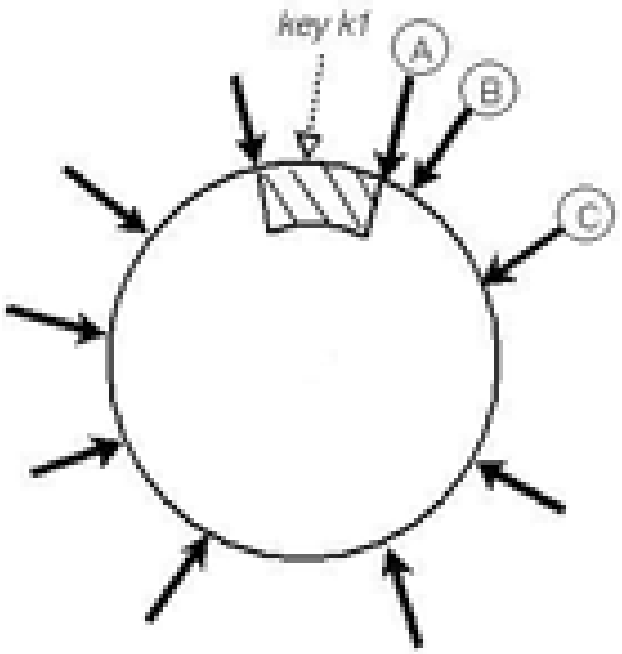


Strategy 1

Для передачи диапазонов ключей узлу необходимо было выполнить сканирование хранилища

Для новых диапазонов необходимо пересчитать деревья Меркле

~ Проблемы стратегии 1 ~



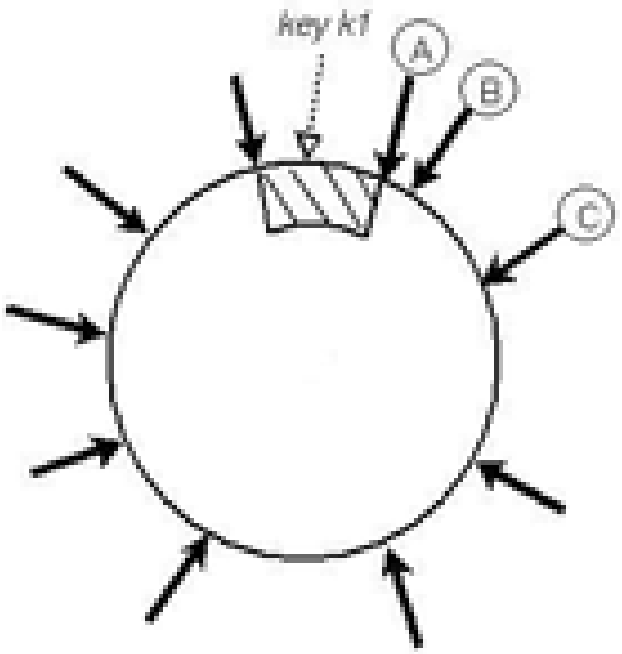
Strategy 1

Для передачи диапазонов ключей узлу необходимо было выполнить сканирование хранилища

Для новых диапазонов необходимо пересчитать деревья Меркле

Нет простого способа сделать snapshot хранилища

~ Проблемы стратегии 1 ~



Strategy 1

Для передачи диапазонов ключей узлу необходимо было выполнить сканирование хранилища

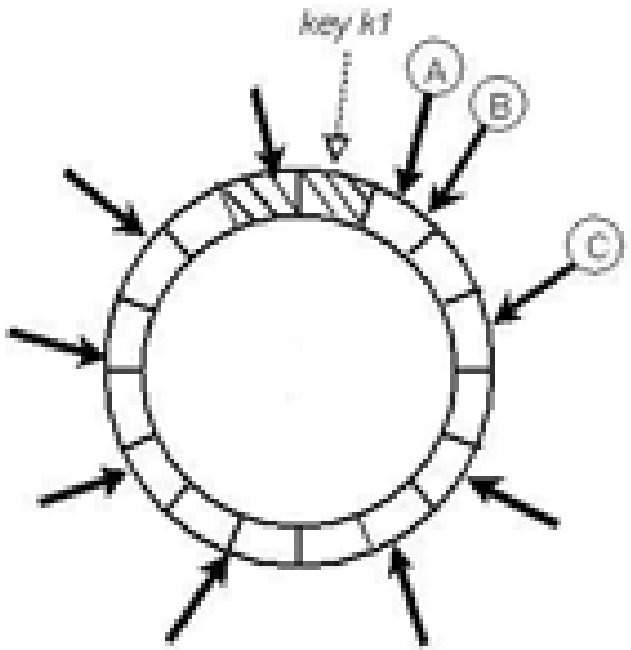
Для новых диапазонов необходимо пересчитать деревья Меркле

Нет простого способа сделать snapshot хранилища

Фундаментальная проблема: взаимосвязь схем сегментирования и физического размещения

~ Стратегия 2 ~

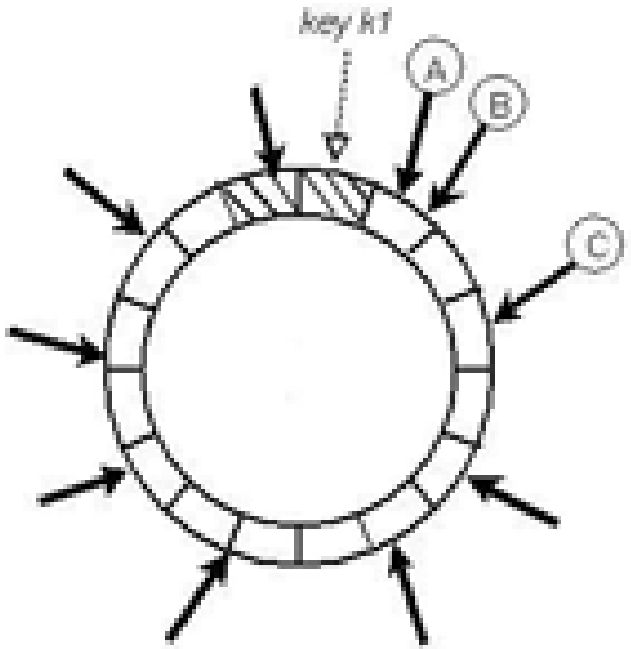
~~ T случайных токенов и равные разбиения ~~



Strategy 2

~ Стратегия 2 ~

~~ T случайных токенов и равные разбиения ~~



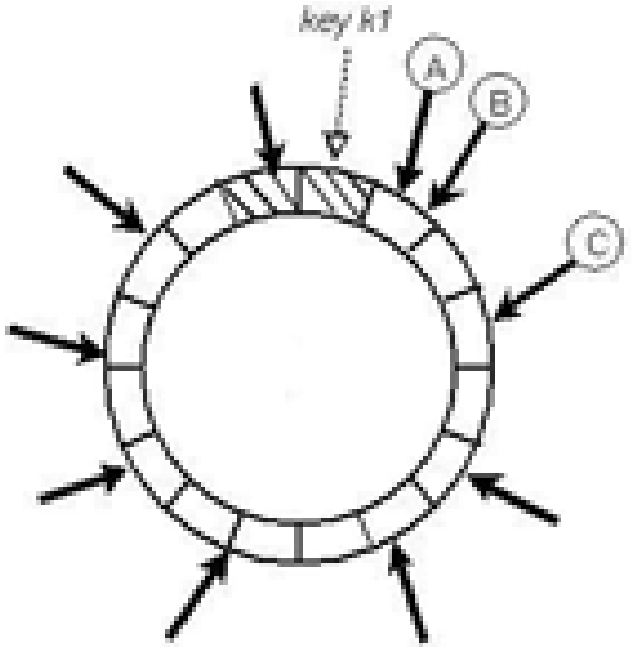
Strategy 2

Хэш-пространство разделено на Q разделов и каждому узлу присваивается T токенов

Требования: **Q** >> **N**, **Q** >> **S * T** (S – число узлов)

~ Стратегия 2 ~

~~ T случайных токенов и равные разбиения ~~



Strategy 2

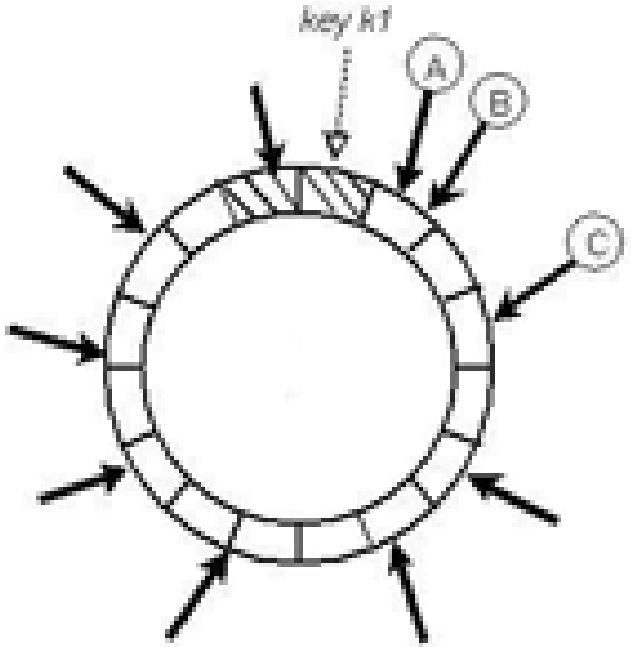
Хэш-пространство разделено на Q разделов и каждому узлу присваивается T токенов

Требования: **Q** >> **N**, **Q** >> **S * T** (S – число узлов)

Токены не влияют на сегментирование, а только задают функцию отображения хэша на кольцо

~ Стратегия 2 ~

~~ T случайных токенов и равные разбиения ~~



Strategy 2

Хэш-пространство разделено на Q разделов и каждому узлу присваивается T токенов

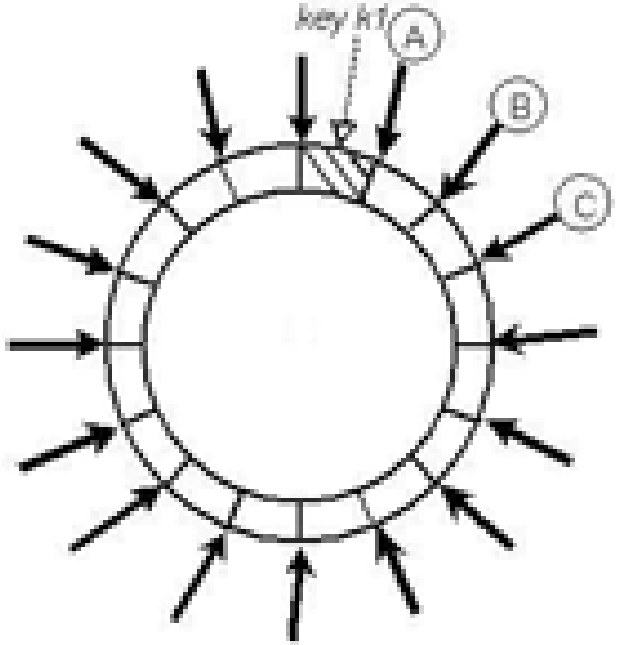
Требования: **Q** >> **N**, **Q** >> **S * T** (S – число узлов)

Токены не влияют на сегментирование, а только задают функцию отображения хэша на кольцо

Раздел помещается на **N** первых узлов

~ Стратегия 3 ~

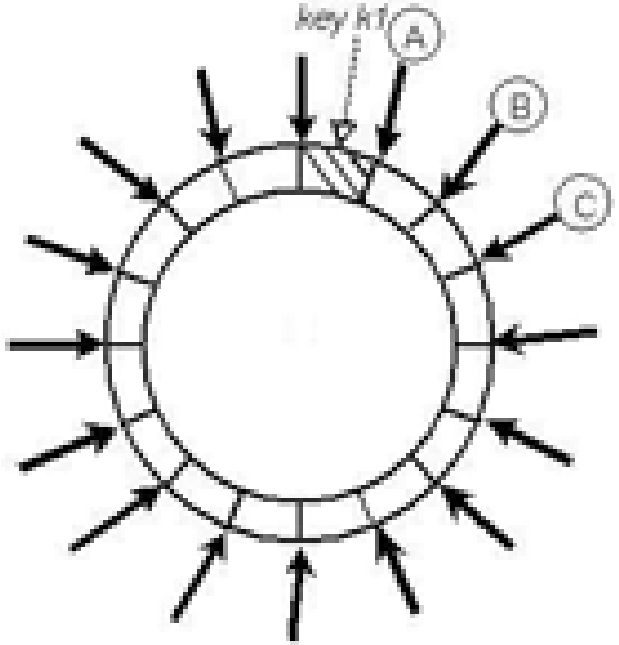
~~ Q/S токенов и равные разбиения ~~



Strategy 3

~ Стратегия 3 ~

~~ Q/S токенов и равные разбиения ~~

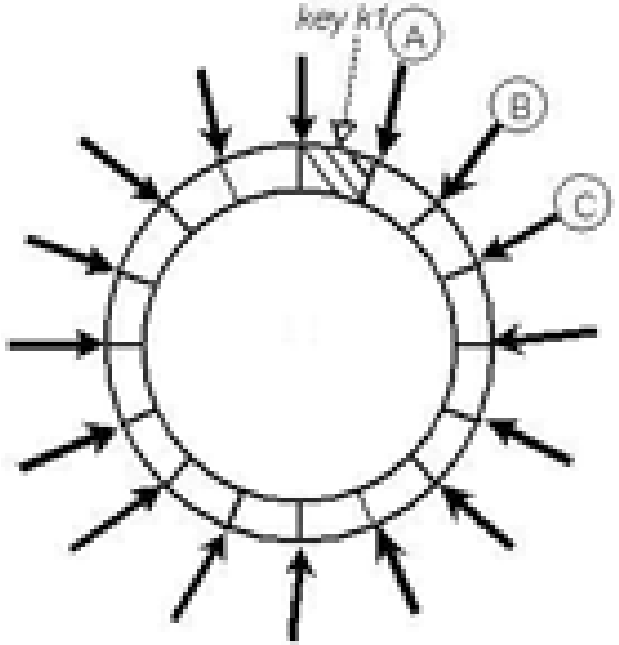


Strategy 3

С каждым узлом сопоставлено **Q/S** токенов

~ Стратегия 3 ~

~~ Q/S токенов и равные разбиения ~~



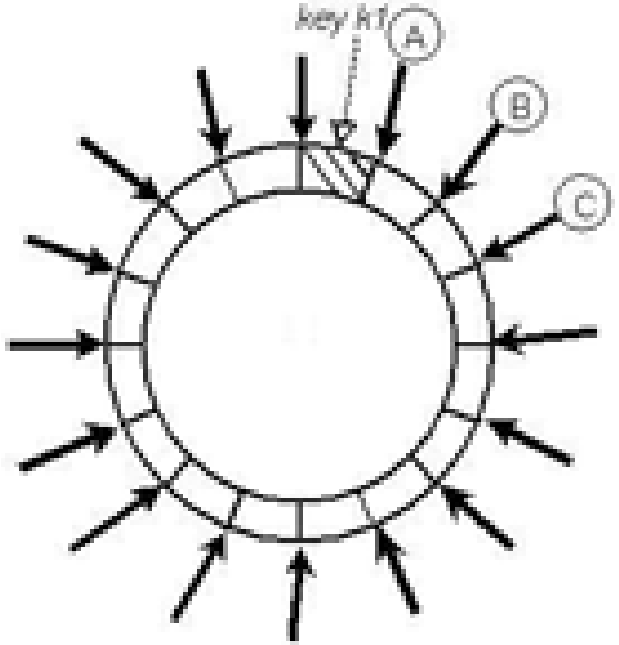
Strategy 3

С каждым узлом сопоставлено **Q/S** токенов

Токены выбывшего узла распределяются между остальными (аналогично при присоединении)

~ Стратегия 3 ~

~~ Q/S токенов и равные разбиения ~~



Strategy 3

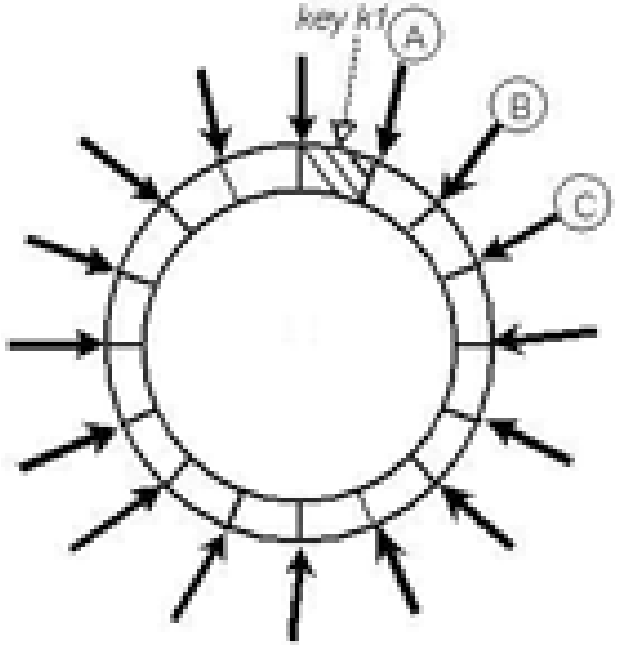
С каждым узлом сопоставлено **Q/S** токенов

Токены выбывшего узла распределяются между остальными (аналогично при присоединении)

Быстрая инициализация и восстановление

~ Стратегия 3 ~

~~ Q/S токенов и равные разбиения ~~



Strategy 3

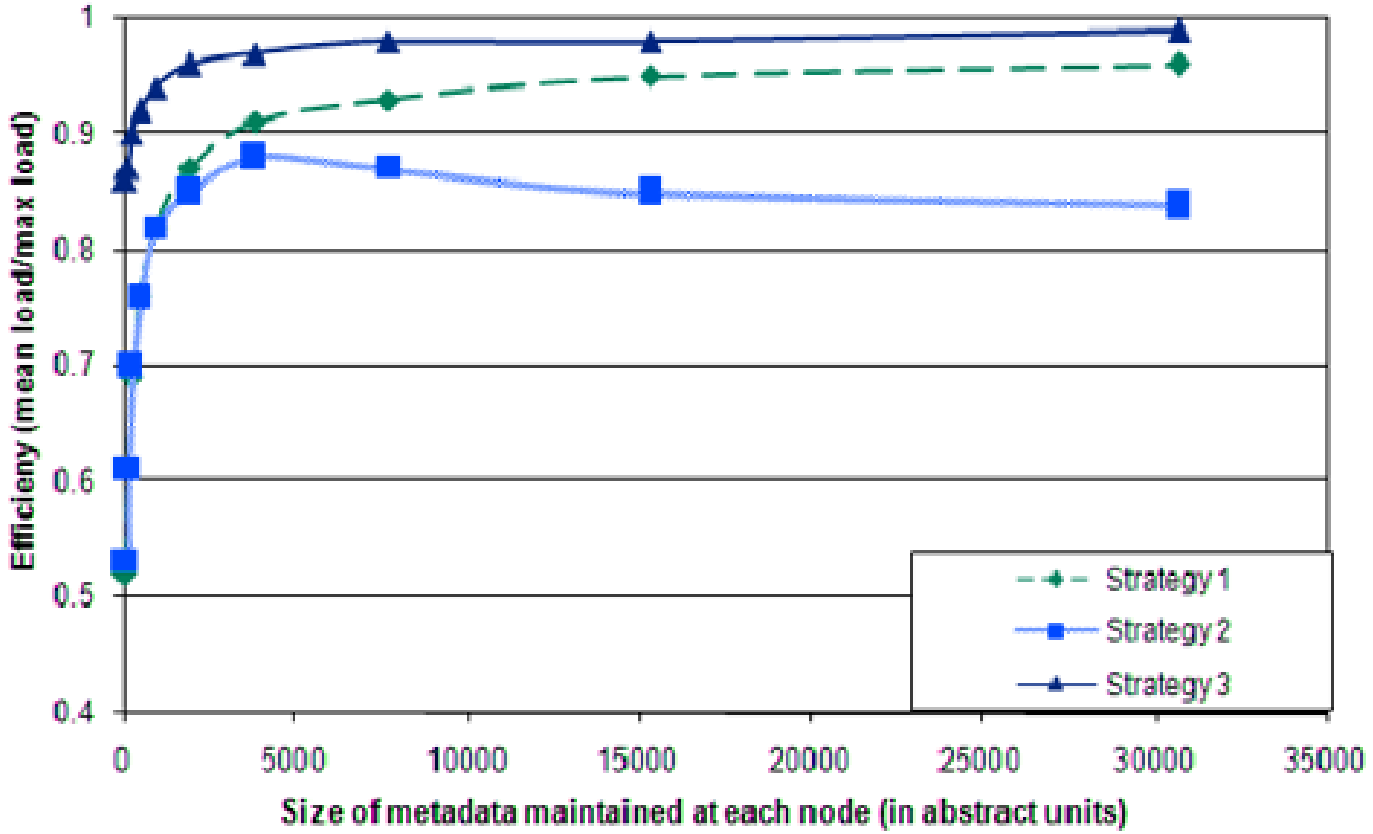
С каждым узлом сопоставлено **Q/S** токенов

Токены выбывшего узла распределяются между остальными (аналогично при присоединении)

Быстрая инициализация и восстановление

Простота архивирования хранилища

~ Сравнение стратегий ~



Сравнение эффективности распределения нагрузки при различных стратегиях для системы с 30 узлами и **N** = 3 с одинаковым объемом метаданных

- Заключение -

сейчас будет хоть что-то понятно

~ Amazon Dynamo ~

Высокая доступность

Успешная обработка сбоев серверов,
дата-центров и нарушение связности сети

~ Amazon Dynamo ~

Высокая доступность

Успешная обработка сбоев серверов, дата-центров и нарушение связности сети

Масштабируемость

Увеличение числа узлов для увеличения производительности в целом

~ Amazon Dynamo ~

Высокая доступность

Успешная обработка сбоев серверов, дата-центров и нарушение связности сети

Масштабируемость

Увеличение числа узлов для увеличения производительности в целом

Кастомизация под требования сервиса

Всего 3 параметра (**N**, **R**, **W**), которые позволяют полностью изменять концепцию использования хранилища

~ Amazon Dynamo ~

Высокая доступность

Успешная обработка сбоев серверов, дата-центров и нарушение связности сети

Масштабируемость

Увеличение числа узлов для увеличения производительности в целом

Кастомизация под требования сервиса

Всего 3 параметра (**N**, **R**, **W**), которые позволяют полностью изменять концепцию использования хранилища

Децентрализованные системы могут быть использованы для построения высокодоступной системы


amazon
DynamoDB