Когда всё пошло по Кафке

Kafka, грабли, DevOps

Григорий Кошелев
СКБ Контур

JPoint, Москва, 2019

План

1. Зачем нам Apache Kafka
2. Введение в Кафку
3. Архитектура
4. Неочевидности
5. Выводы

Зачем нам Apache Kafka

Зачем нам Apache Kafka

- Vostok Hercules

https://github.com/vostok

Зачем нам Apache Kafka

- Vostok Hercules
- Логи

https://github.com/vostok

Зачем нам Apache Kafka

- Vostok Hercules

- Логи
- Метрики

https://github.com/vostok

Зачем нам Apache Kafka

- Vostok Hercules

- Логи
- Метрики
- Трассировки

https://github.com/vostok

Зачем нам Apache Kafka

- Vostok Hercules

- Логи
- Метрики
- Трассировки
- Бизнес-события

https://github.com/vostok

Зачем нам Apache Kafka

- Vostok Hercules
- Search & Recommendation Systems (SRS)

Зачем нам Apache Kafka

- Vostok Hercules [Kafka 2.0+]
- Search & Recommendation Systems (SRS)

Зачем нам Apache Kafka

- Vostok Hercules [Kafka 2.0+]
- Search & Recommendation Systems (SRS) [Kafka 0.11.x]

Блиц-опрос

Блиц-опрос

Кто использует Apache Kafka?

Блиц-опрос

Кто использует Apache Kafka?

Версия…

< 0.11?

Блиц-опрос

Кто использует Apache Kafka?

Версия…

< 0.11?

0.11.x?

Блиц-опрос

Кто использует Apache Kafka?

Версия…

< 0.11?

0.11.x?

1.x.x?

Блиц-опрос

Кто использует Apache Kafka?

Версия…

< 0.11?

0.11.x?

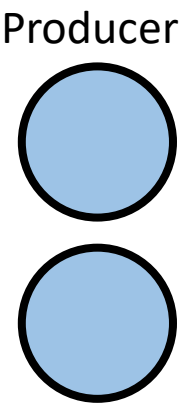
1.x.x?

2.x.x?

Введение в Apache Kafka

Введение в Apache Kafka

Kafka Producer

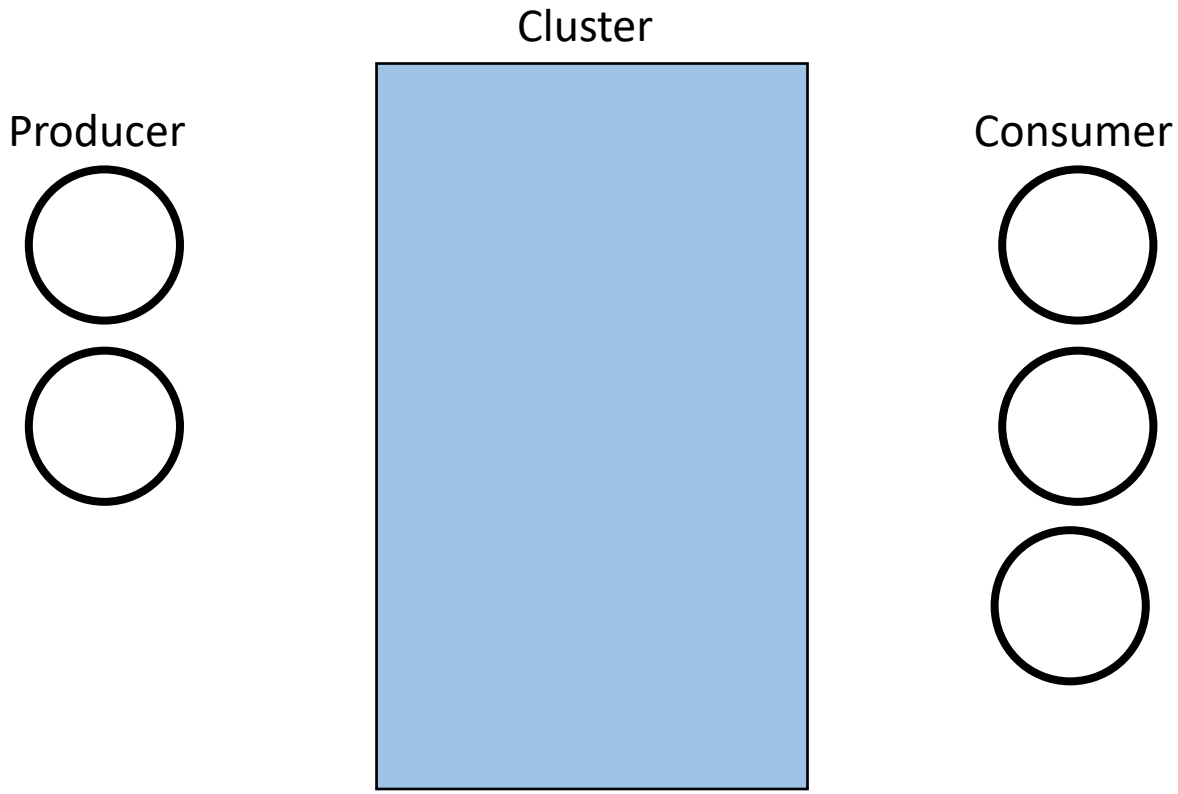


Введение в Apache Kafka

Kafka Consumer

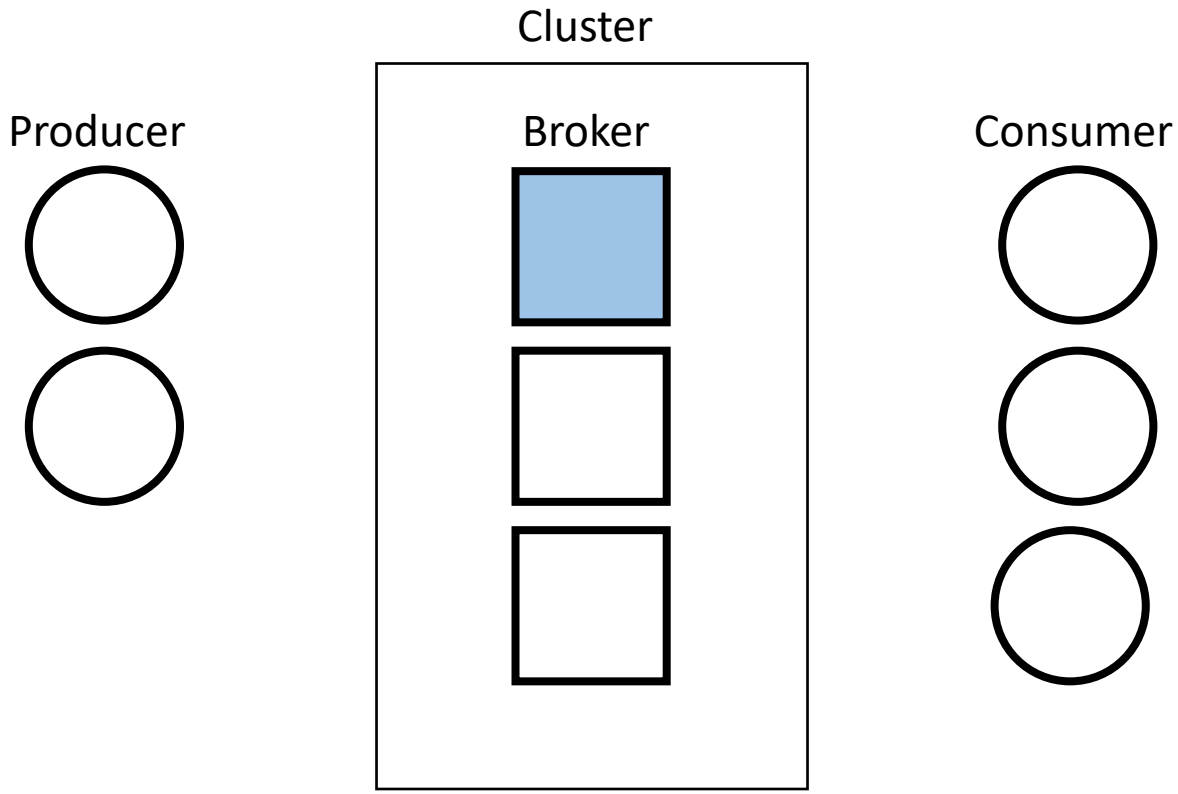
Введение в Apache Kafka

Kafka Cluster

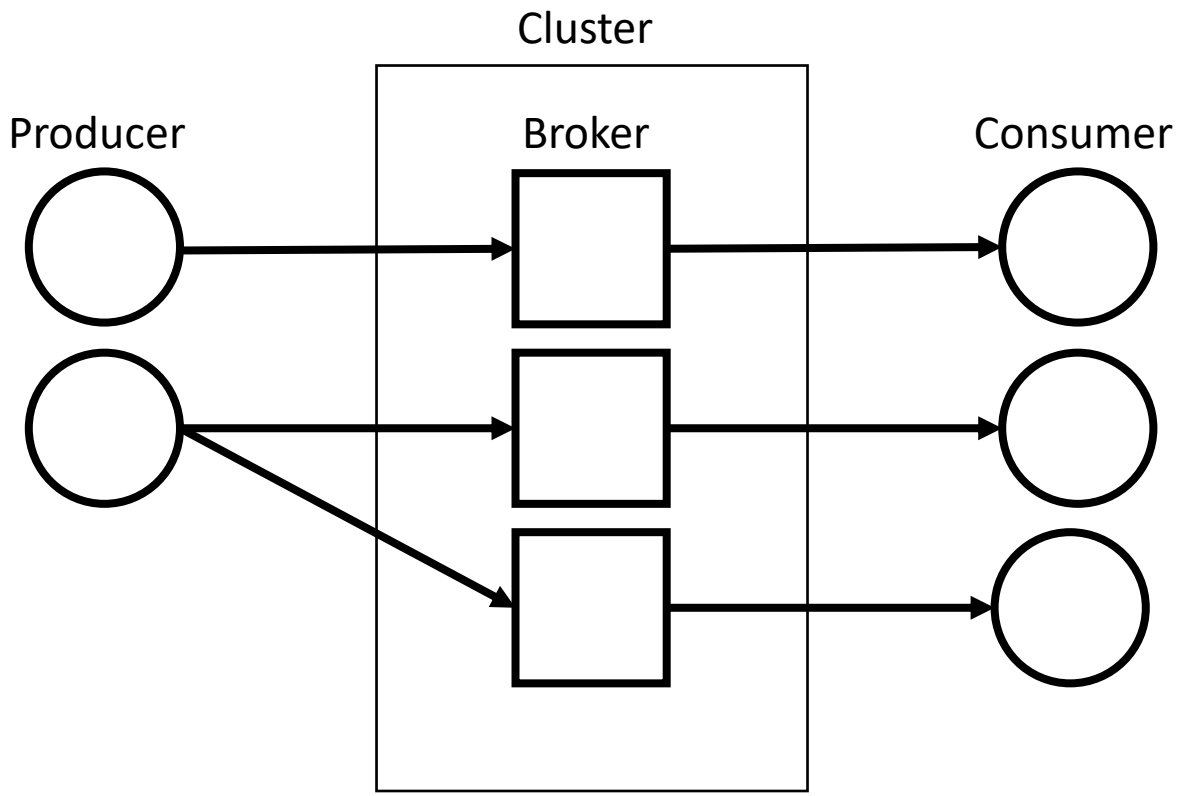


Введение в Apache Kafka

Kafka Broker

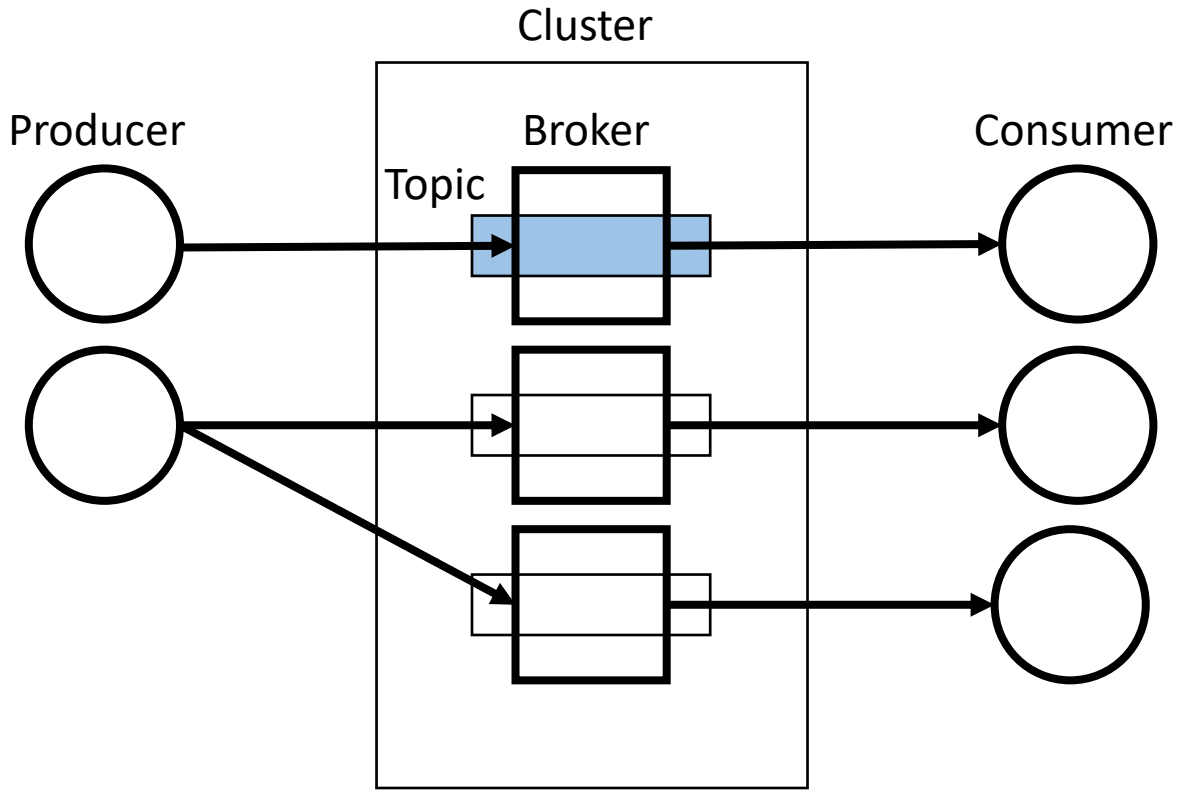


Введение в Apache Kafka



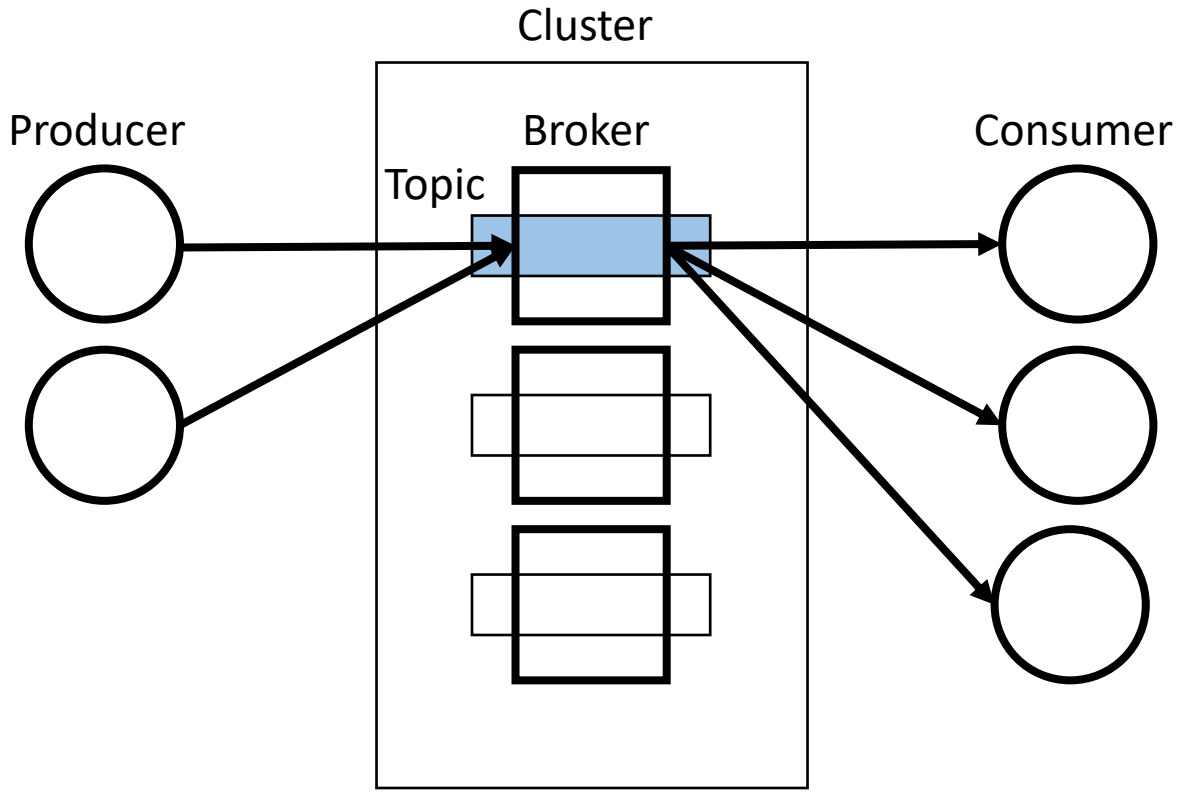
Введение в Apache Kafka

Kafka Topic



Введение в Apache Kafka

Pub-Sub с poll-механикой чтения

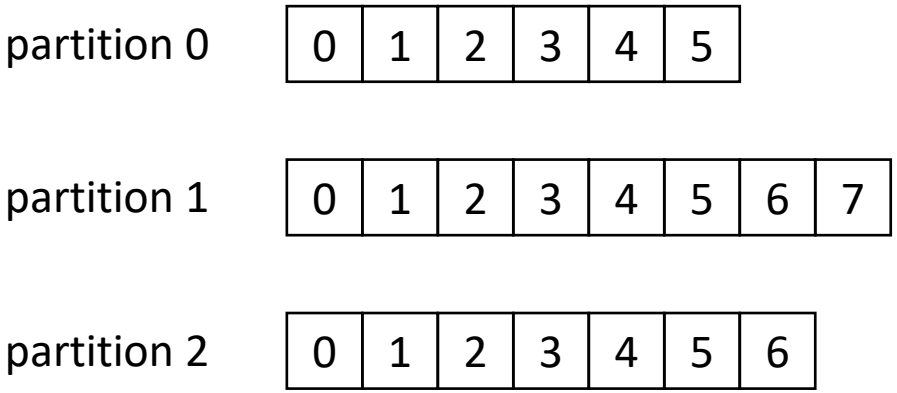


Архитектура Apache Kafka

- Topic
- Broker
- Producer
- Consumer

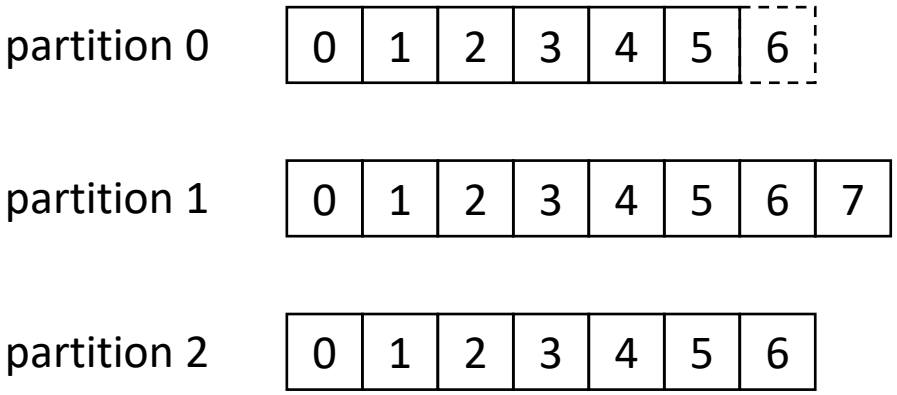
Архитектура Kafka Topic

topic = {partition}



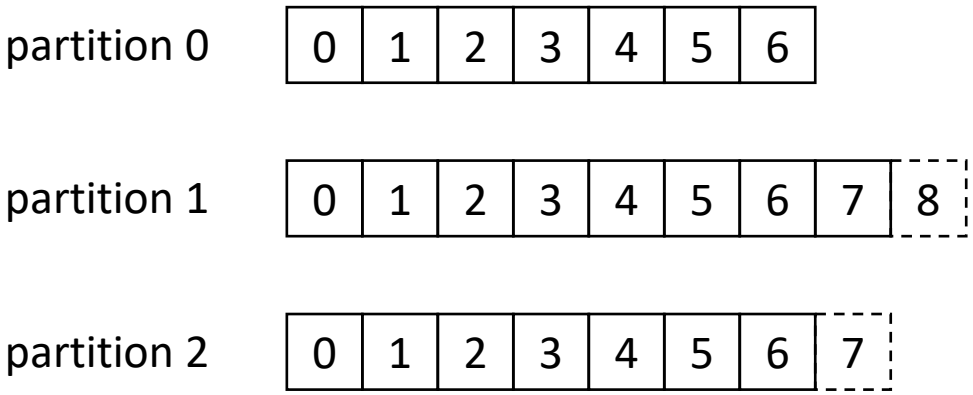
Архитектура Kafka Topic

topic = {partition}



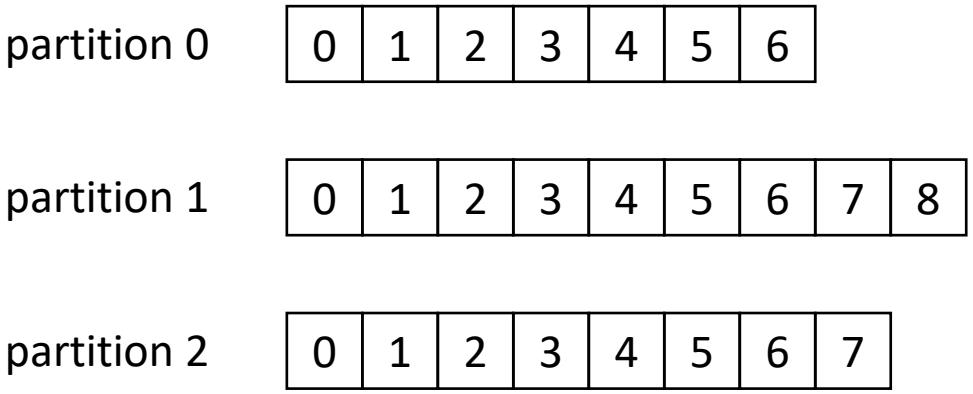
Архитектура Kafka Topic

topic = {partition}



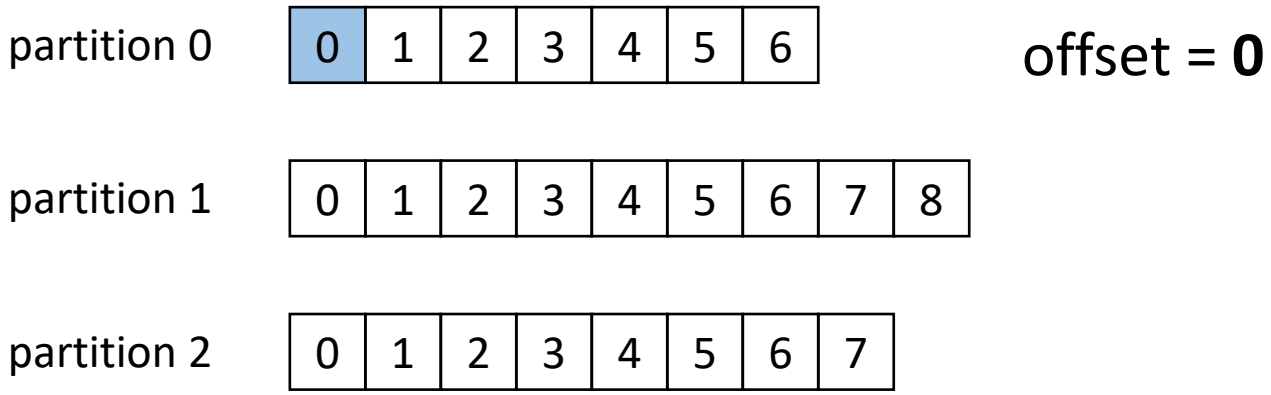
Архитектура Kafka Topic

topic = {partition}



Архитектура Kafka Topic

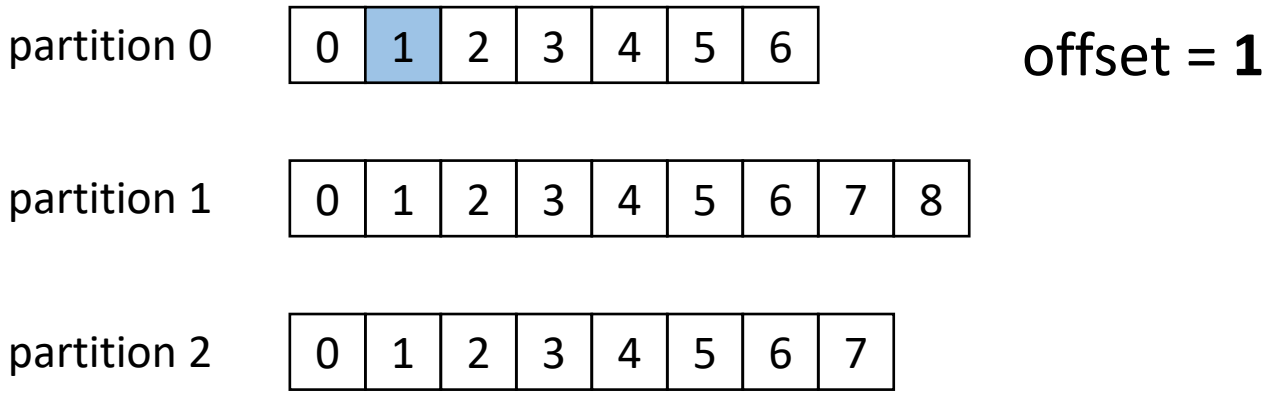
topic = {partition}



offset = **0**

Архитектура Kafka Topic

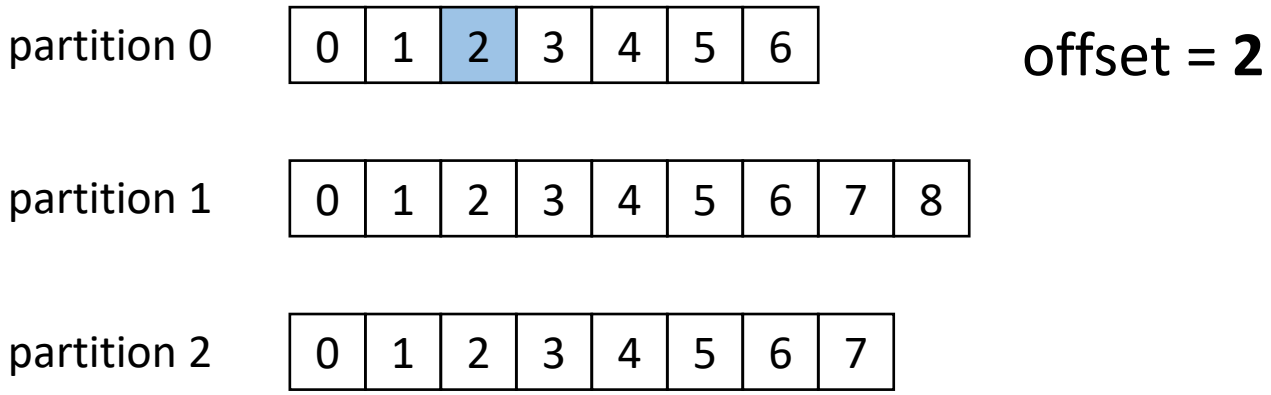
topic = {partition}



offset = **1**

Архитектура Kafka Topic

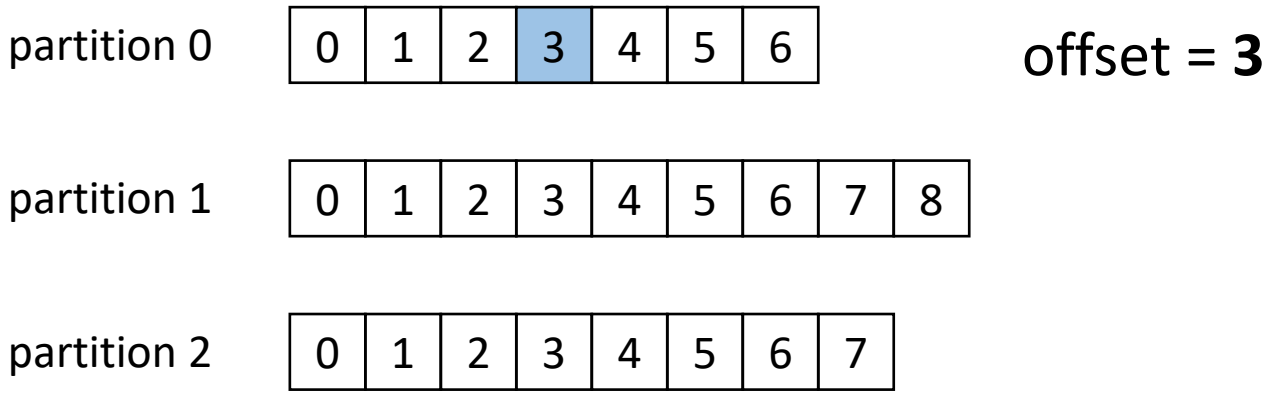
topic = {partition}



offset = **2**

Архитектура Kafka Topic

topic = {partition}



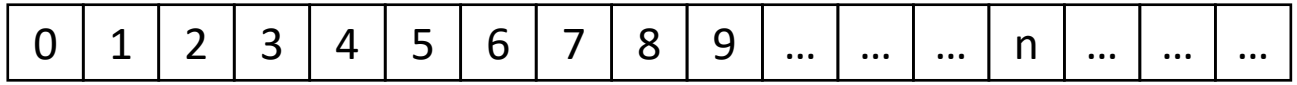
offset = **3**

Архитектура Kafka Topic

partition = {segment}

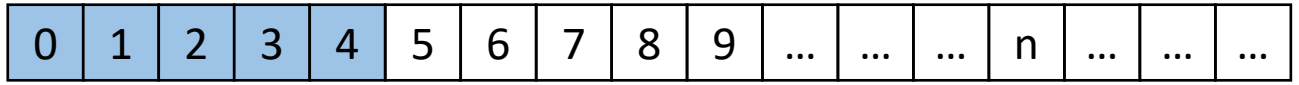
Архитектура Kafka Topic

partition = {segment}



Архитектура Kafka Topic

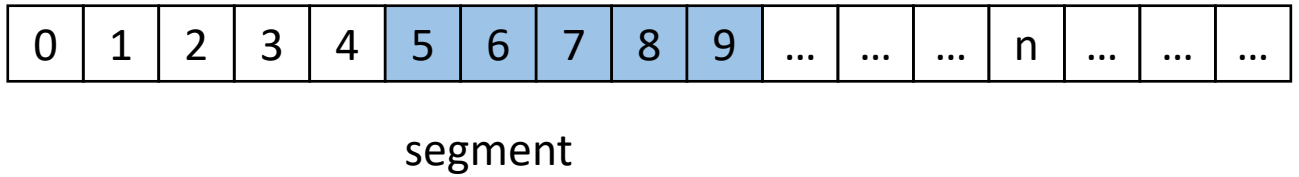
partition = {segment}



segment

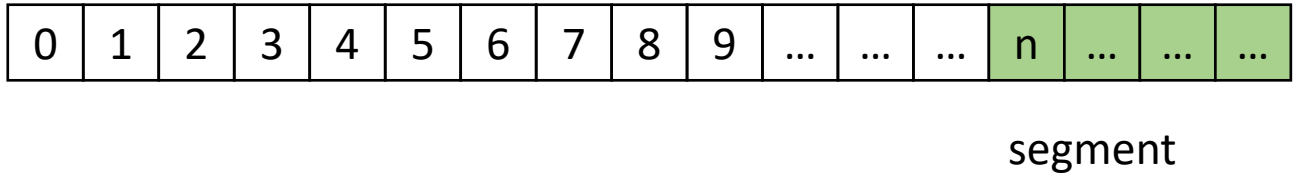
Архитектура Kafka Topic

partition = {segment}



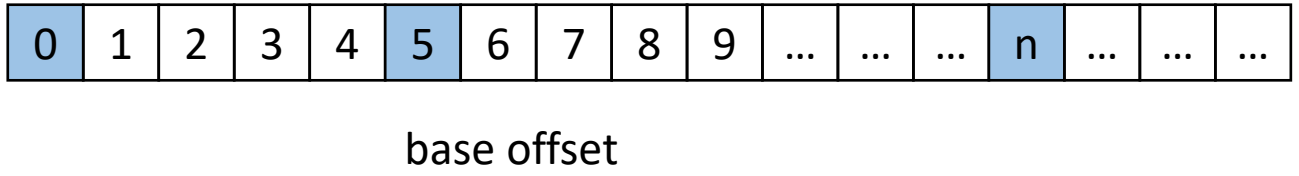
Архитектура Kafka Topic

partition = {segment}



Архитектура Kafka Topic

partition = {segment}



Архитектура Kafka Topic

segment = (base_offset, data, index, timeindex)

```
00000000001234567890.log
00000000001234567890.index
00000000001234567890.timeindex
```

Архитектура Kafka Topic

segment = (**base_offset**, data, index, timeindex)

000000000**1234567890**.log
000000000**1234567890**.index
000000000**1234567890**.timeindex

Архитектура Kafka Topic

segment = (base_offset, **data**, index, timeindex)

00000000001234567890.log
00000000001234567890.index
00000000001234567890.timeindex

Архитектура Kafka Topic

segment = (base_offset, **data**, index, timeindex)

00000000001234567890.log
00000000001234567890.index
00000000001234567890.timeindex

log

Архитектура Kafka Topic

segment = (base_offset, data, **index**, timeindex)

```
00000000001234567890.log
00000000001234567890.index
00000000001234567890.timeindex
```

log

Index record = (relative offset, position)

Архитектура Kafka Topic

segment = (base_offset, data, **index**, timeindex)

```
00000000001234567890.log
00000000001234567890.index
00000000001234567890.timeindex
```

log

offset = 1234567890 relative offset = **0**

size = 100 position = **0**

Index record = (relative offset, position)

Архитектура Kafka Topic

segment = (base_offset, data, **index**, timeindex)

```
00000000001234567890.log
00000000001234567890.index
00000000001234567890.timeindex
```

log

Index record = (relative offset, position)

offset = 1234567891 relative offset = **1**

size = 100 position = **100**

Архитектура Kafka Topic

segment = (base_offset, data, **index**, timeindex)

```
00000000001234567890.log
00000000001234567890.index
00000000001234567890.timeindex
```

log

Index record = (relative offset, position)

offset = 1234567892 relative offset = **2**

size = 50 position = **200**

Архитектура Kafka Topic

segment = (base_offset, data, **index**, timeindex)

```
00000000001234567890.log
00000000001234567890.index
00000000001234567890.timeindex
```

log

Index record = (relative offset, position)

offset = 1234567893 relative offset = **3**

size = 150 position = **250**

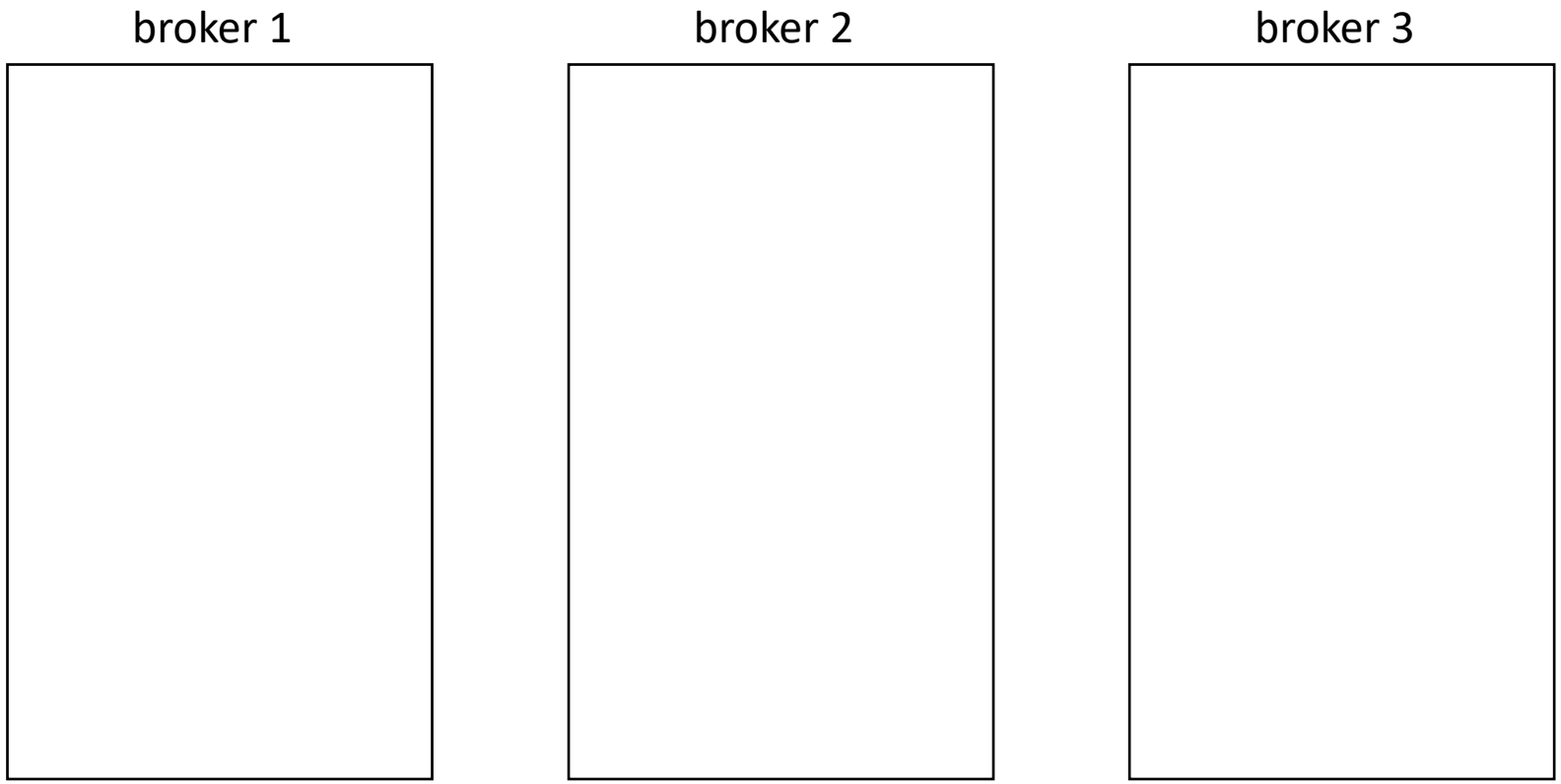
Архитектура Kafka Topic

segment = (base_offset, data, index, **timeindex**)

```
00000000001234567890.log
00000000001234567890.index
00000000001234567890.timeindex
```

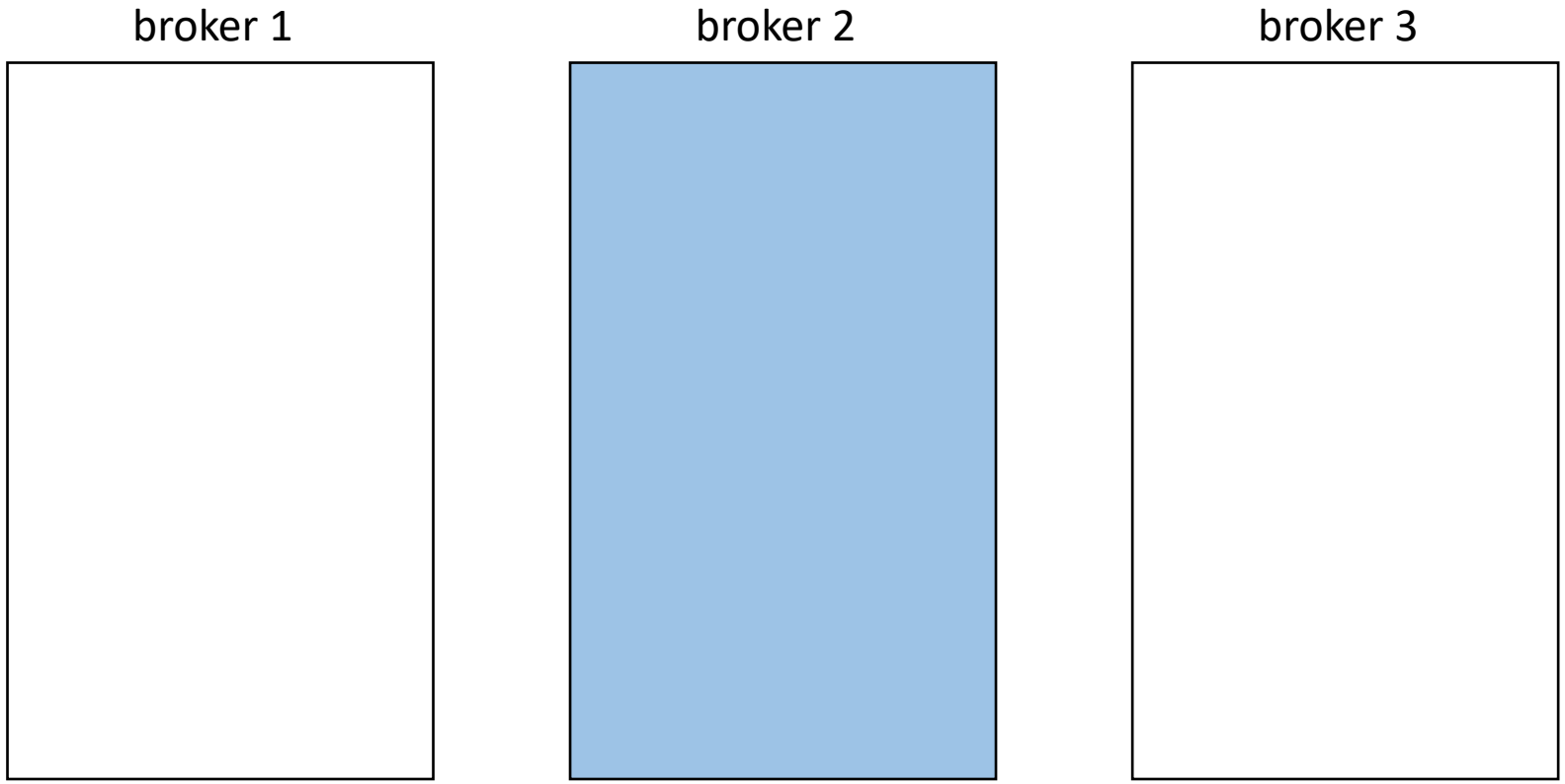
Архитектура Kafka Broker

cluster = {broker}



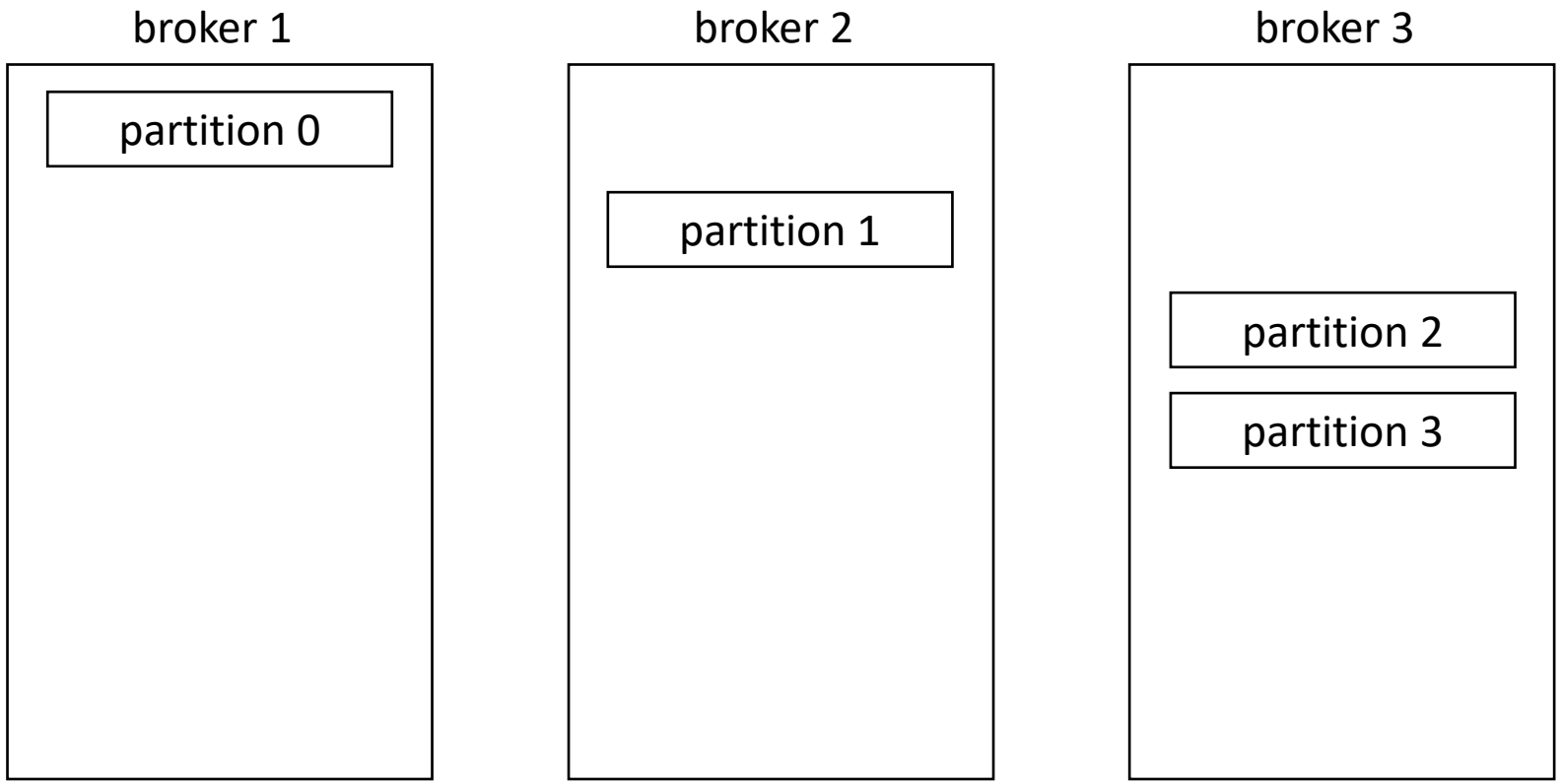
Архитектура Kafka Broker

Controller – координирует работу кластера



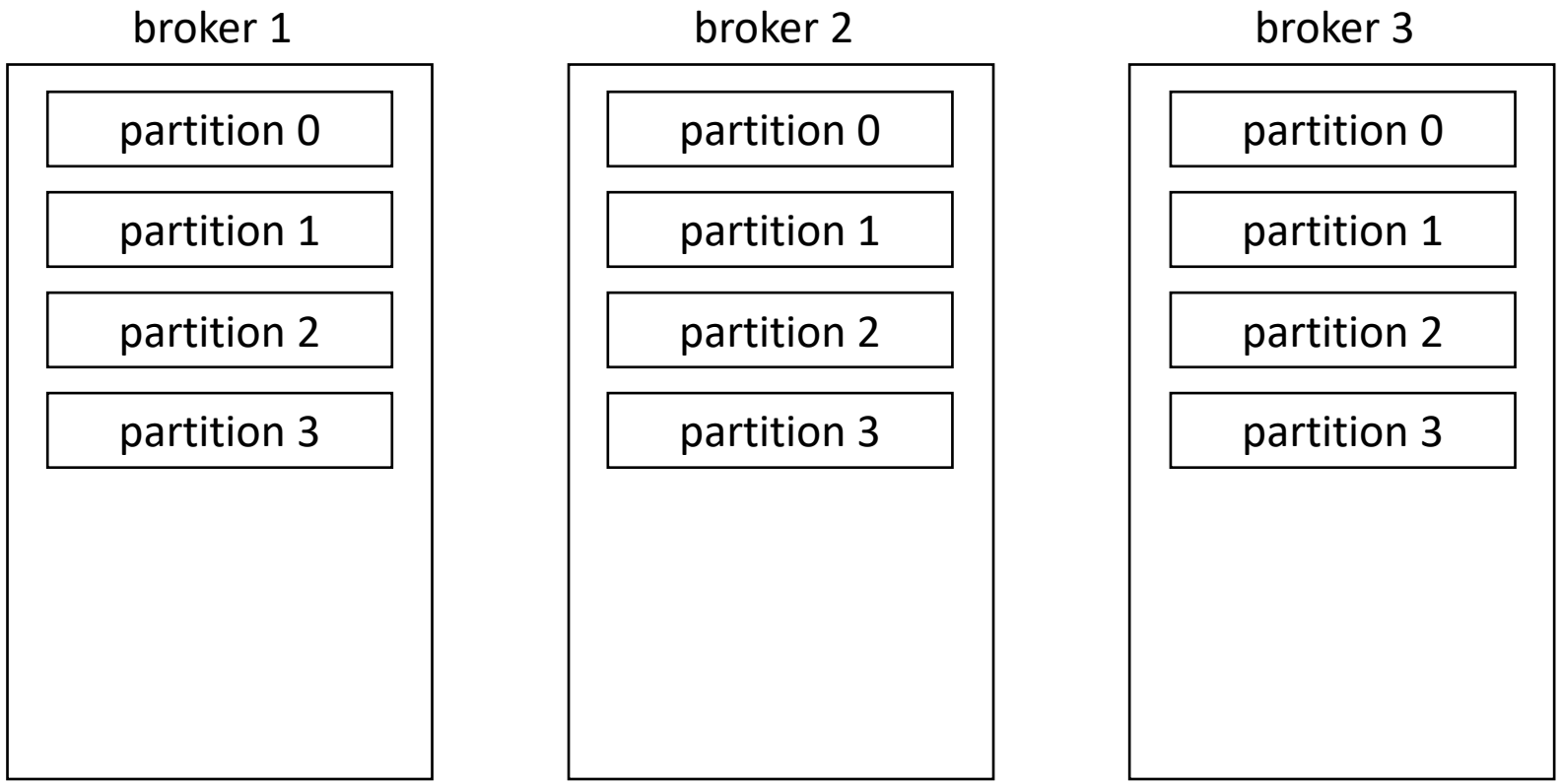
Архитектура Kafka Broker

topic = {partition}



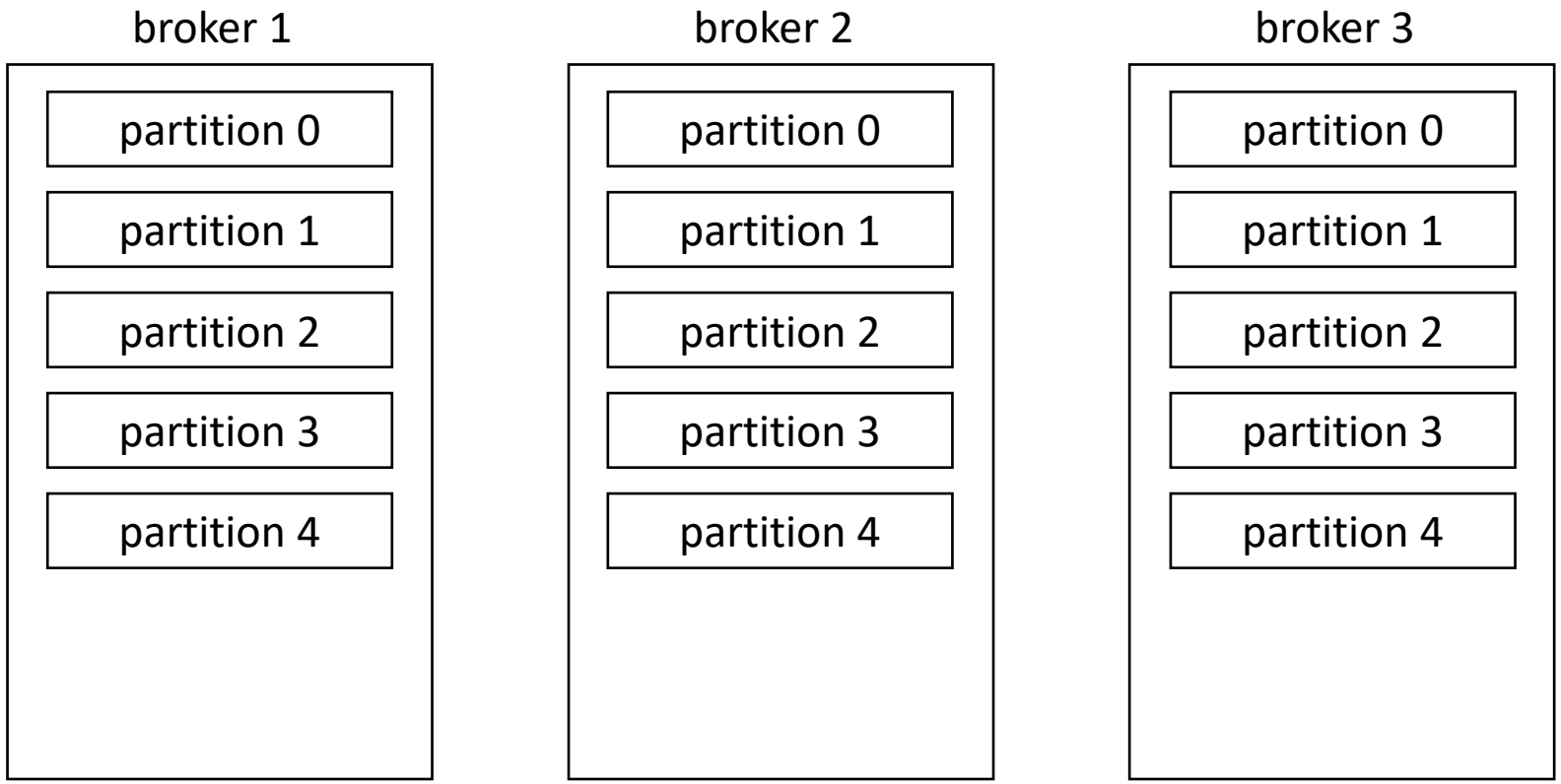
Архитектура Kafka Broker

replication factor = 3



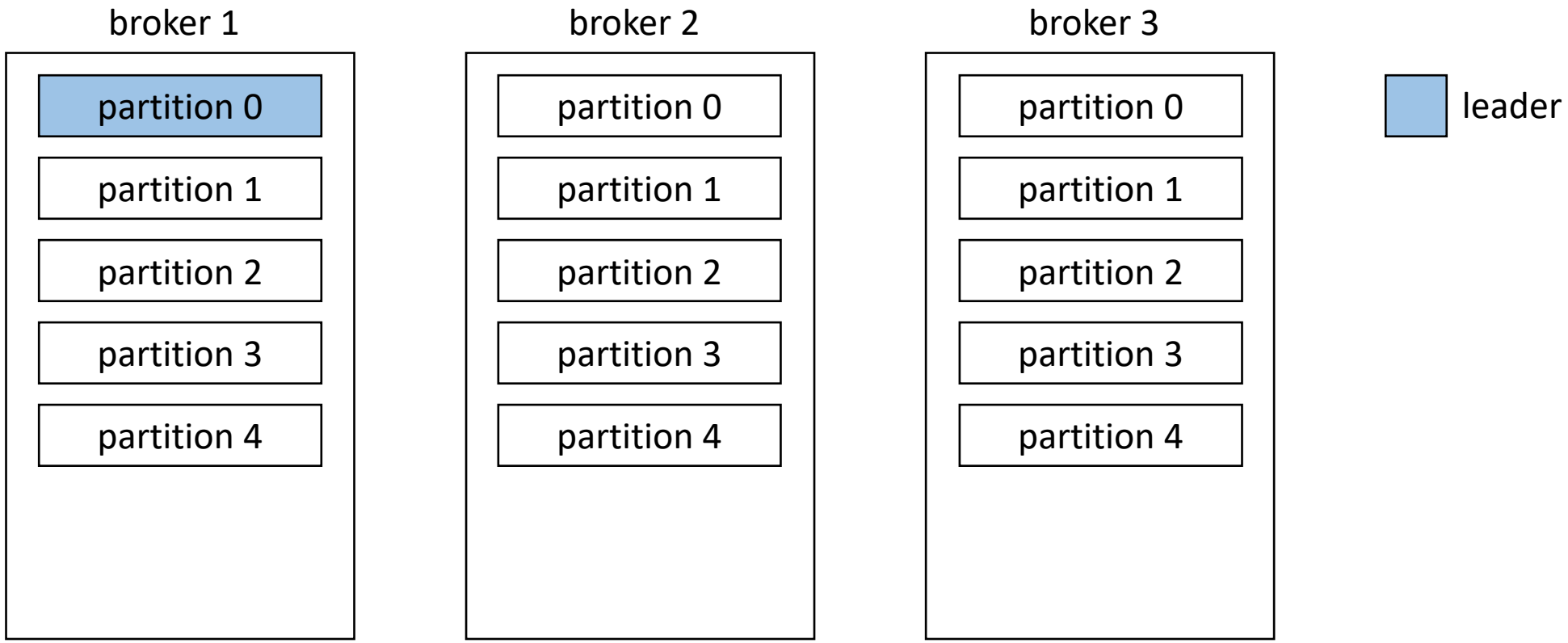
Архитектура Kafka Broker

Добавление partition



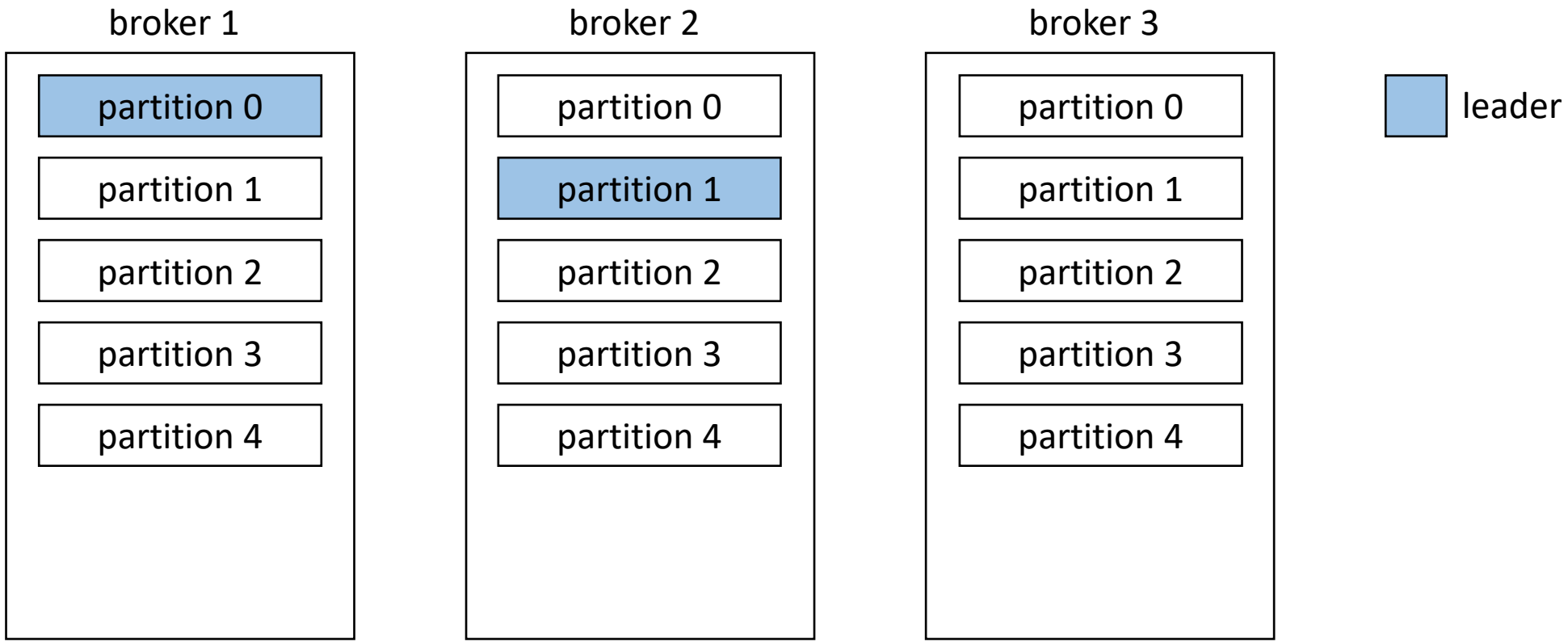
Архитектура Kafka Broker

broker 1 – leader для partition 0.



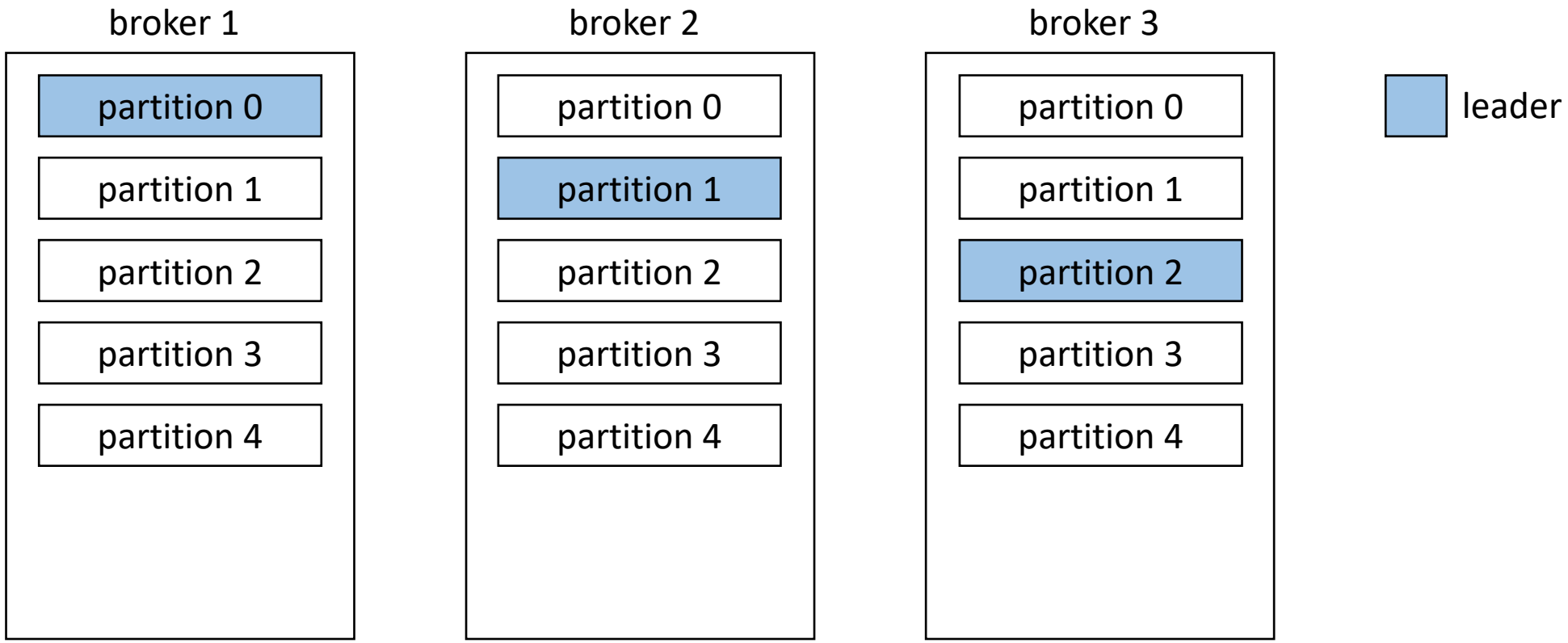
Архитектура Kafka Broker

broker 2 – leader для partition 1



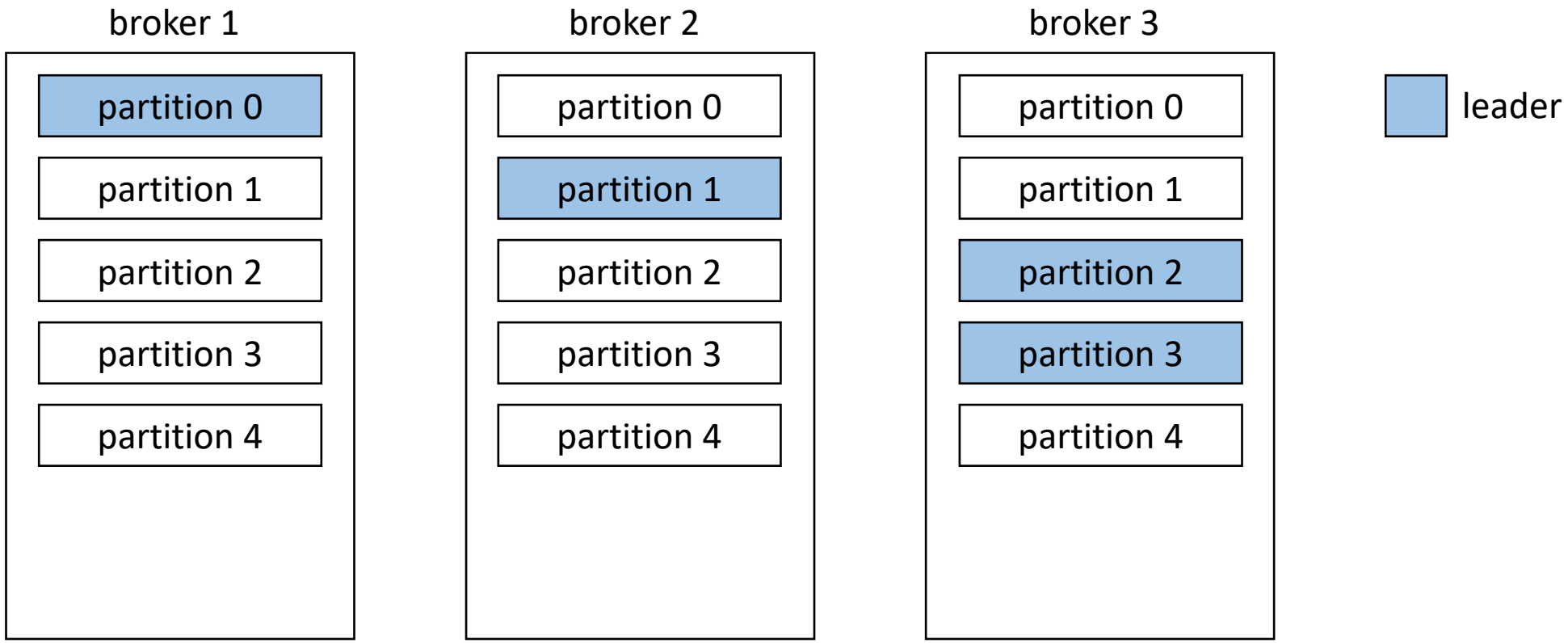
Архитектура Kafka Broker

broker 3 – leader для partition 2



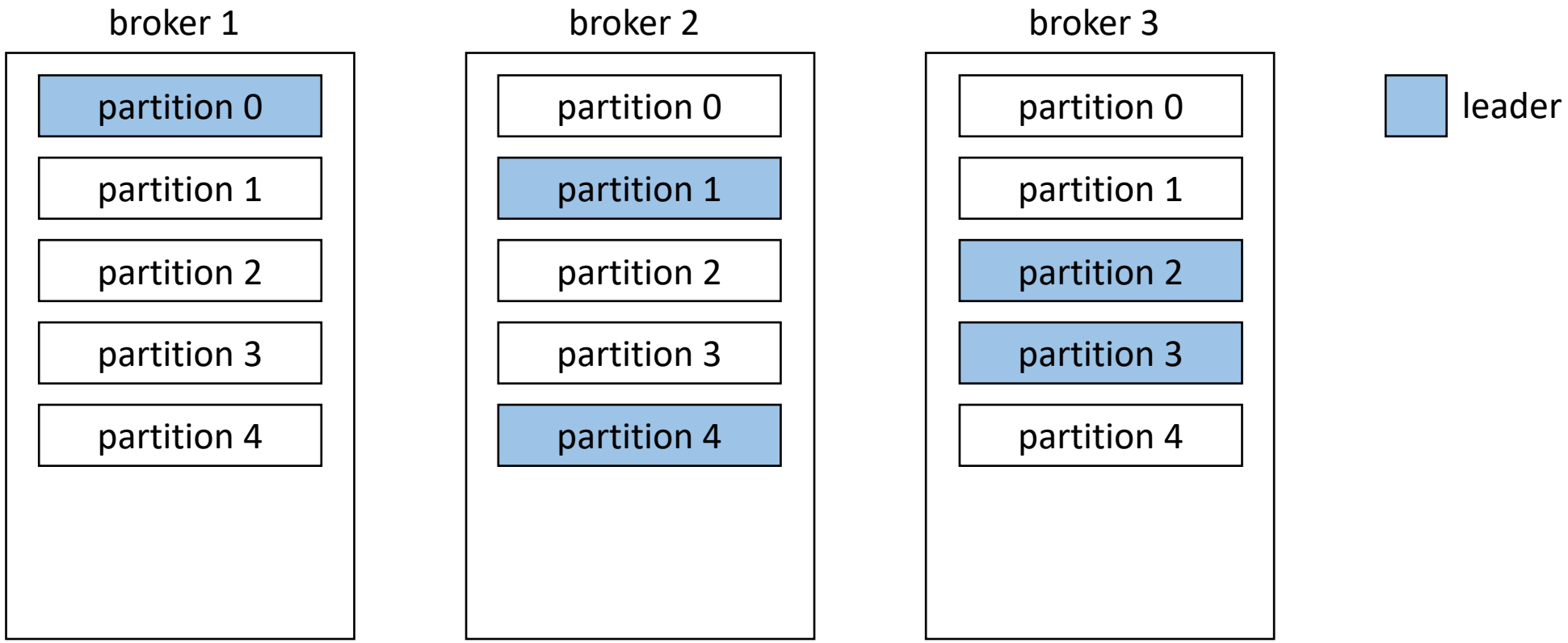
Архитектура Kafka Broker

broker 3 – leader для partition 3



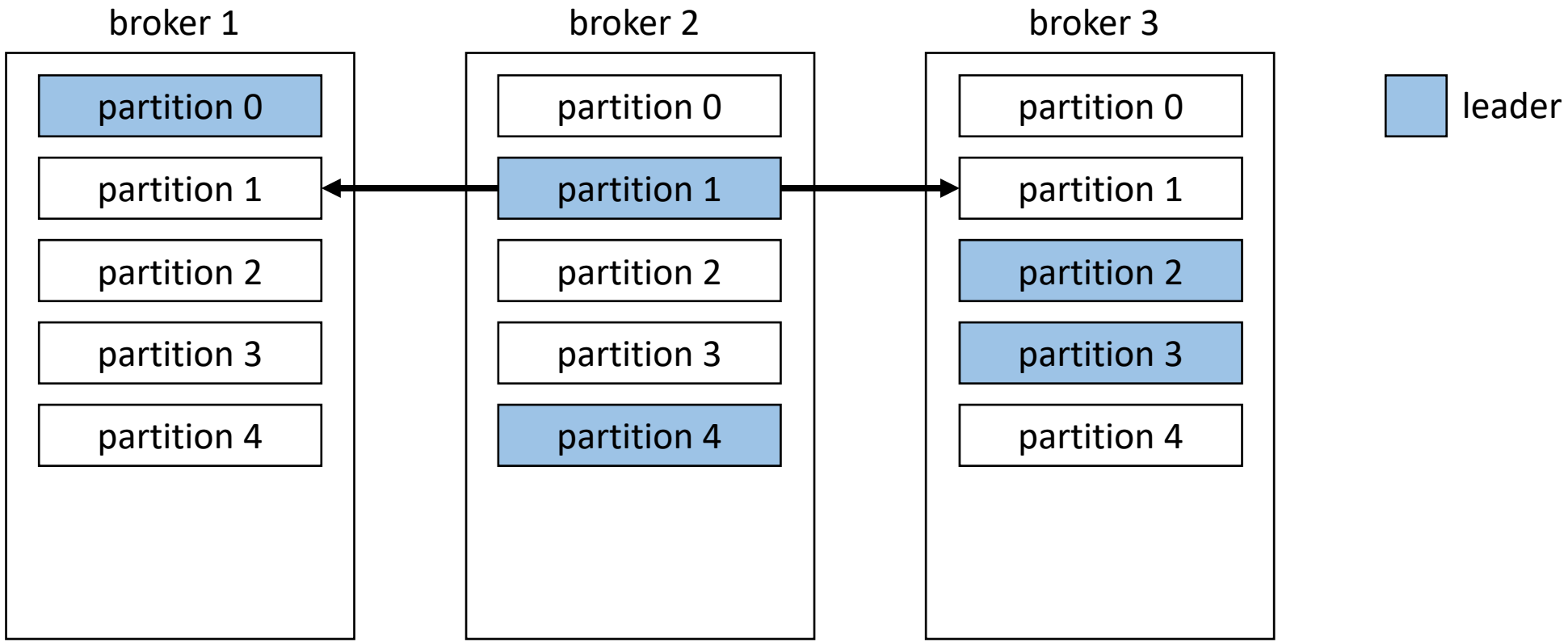
Архитектура Kafka Broker

broker 2 – leader для partition 4



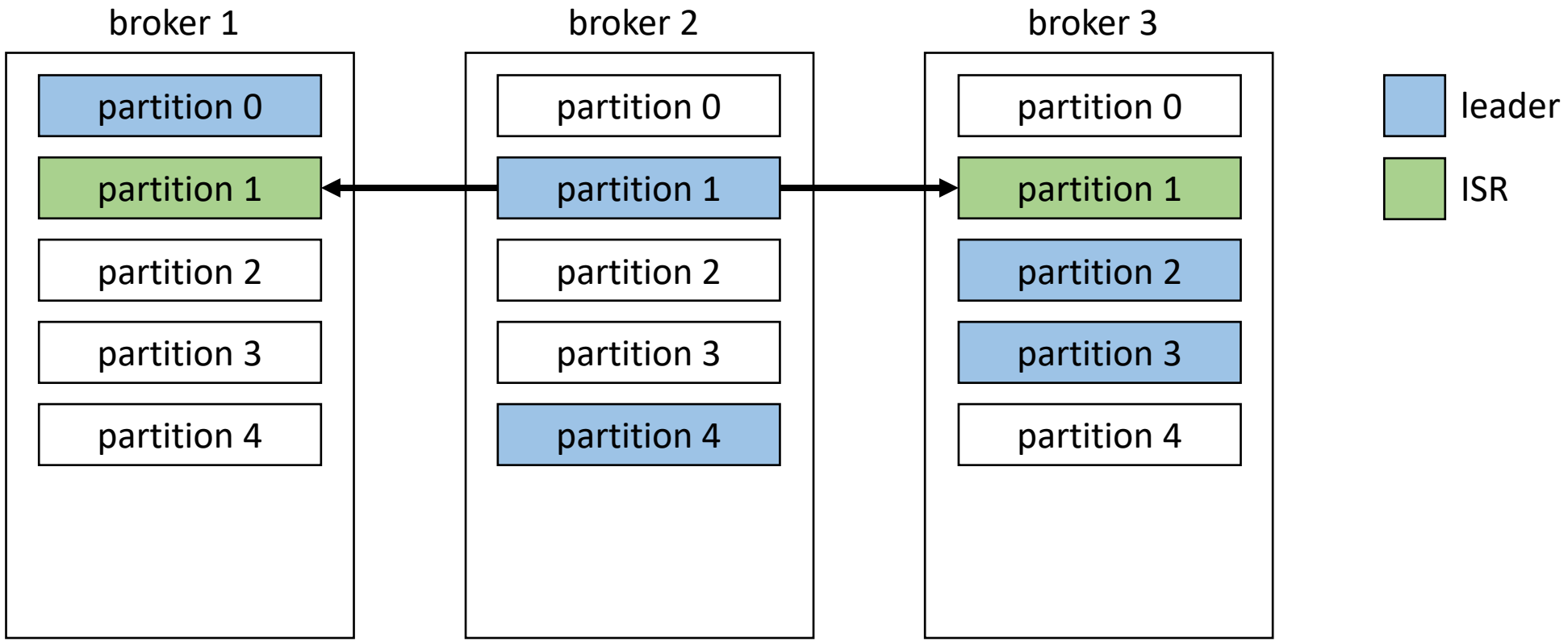
Архитектура Kafka Broker

Репликация с *лидера* на другие *брокеры* (*фолловеры*)



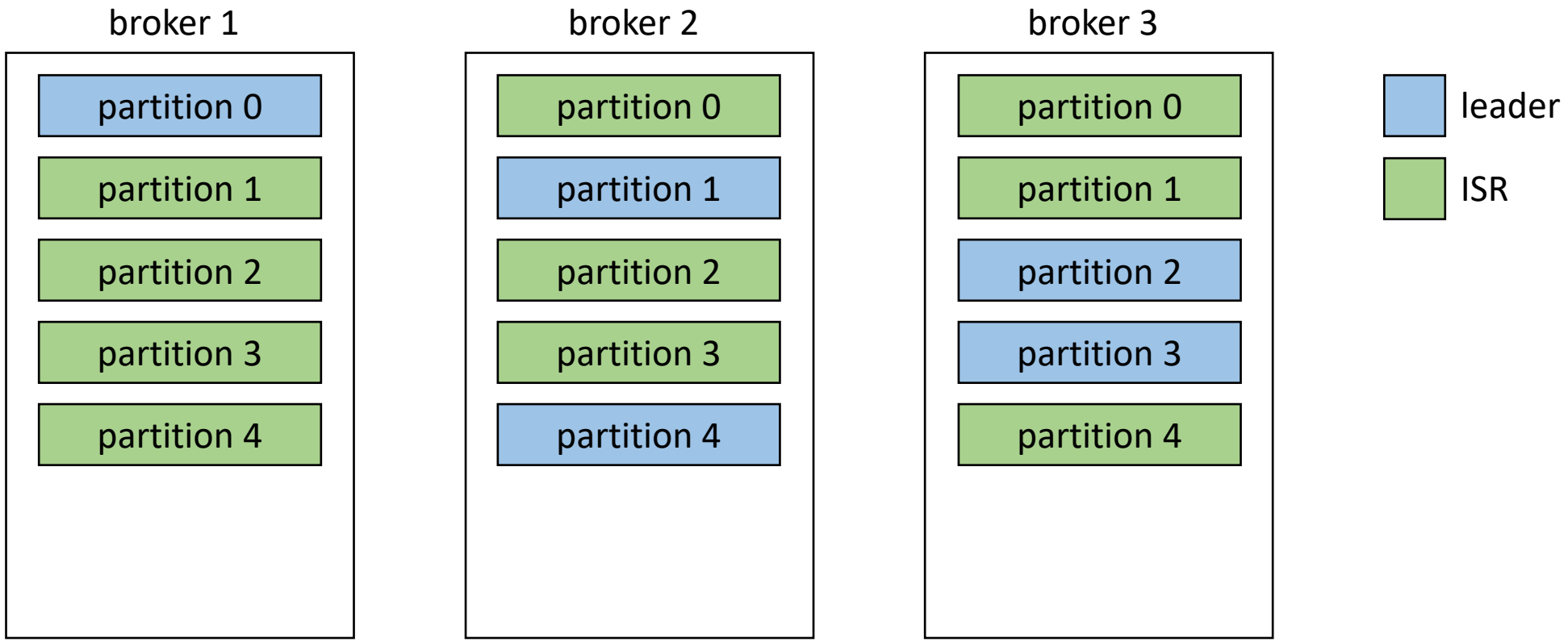
Архитектура Kafka Broker

ISR (in sync replica) – *реплика*, синхронизированная с *лидером*



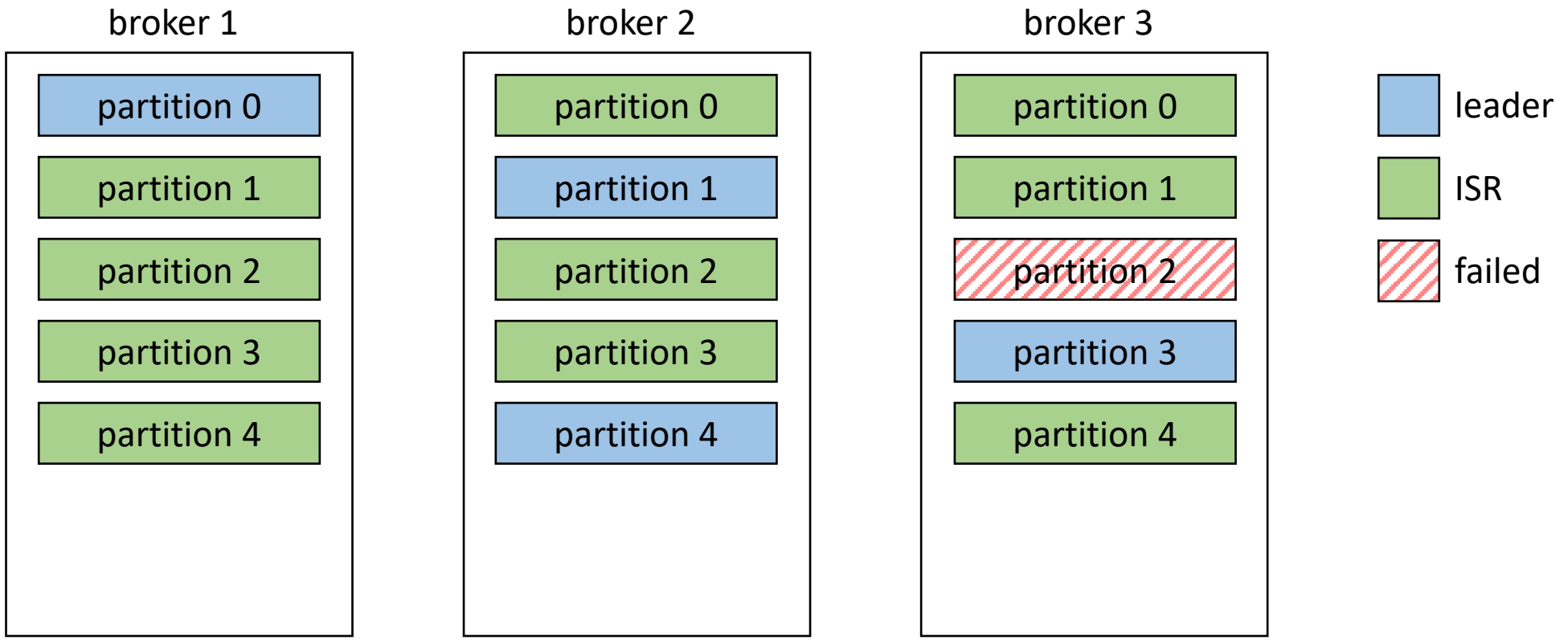
Архитектура Kafka Broker

Все реплики синхронизированы



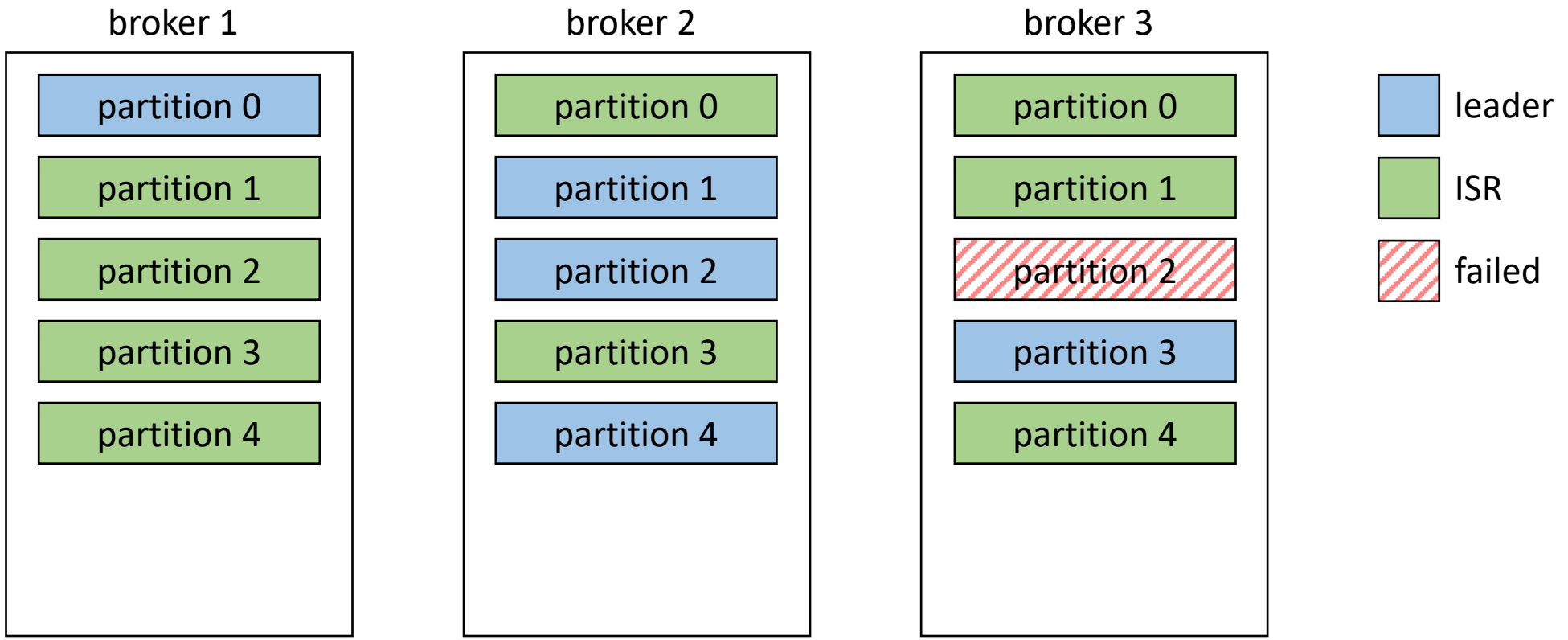
Архитектура Kafka Broker

Недоступность *лидера* у partition 2



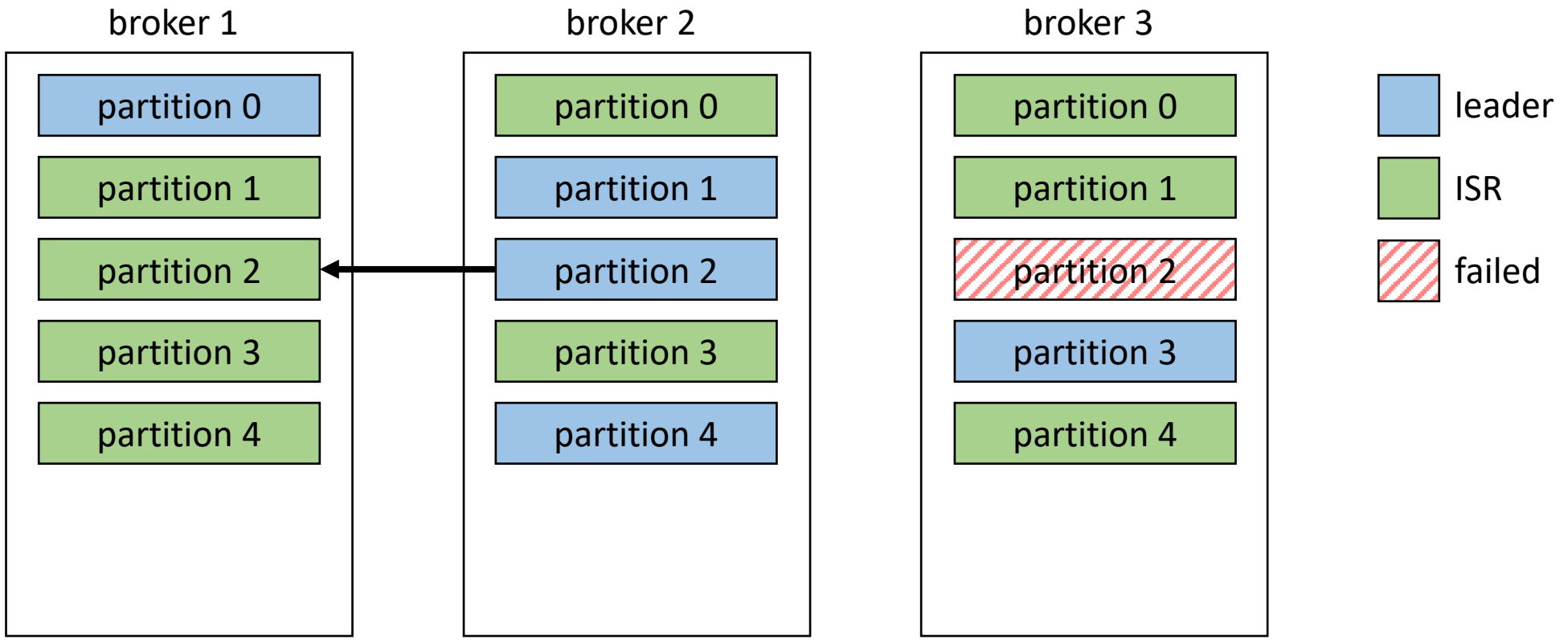
Архитектура Kafka Broker

Выбор нового *лидера* в случае недоступности

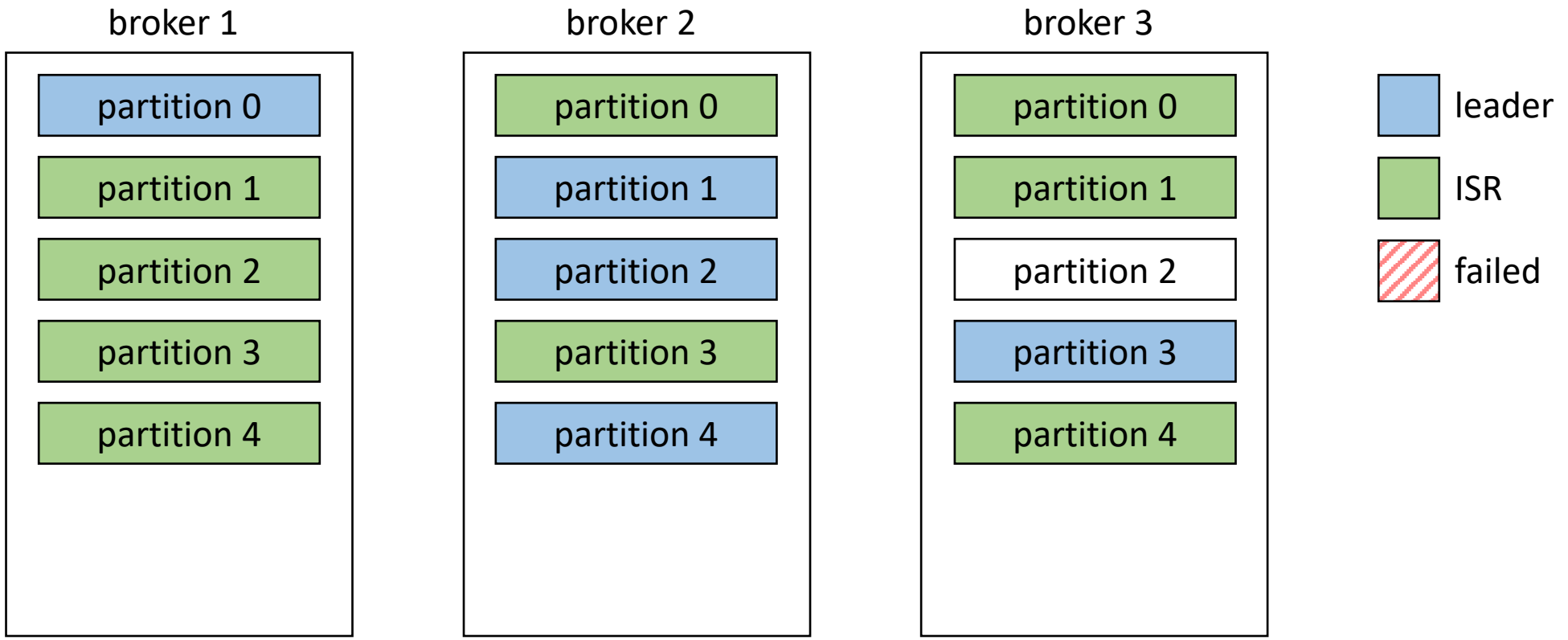


Архитектура Kafka Broker

Репликация с нового *лидера*

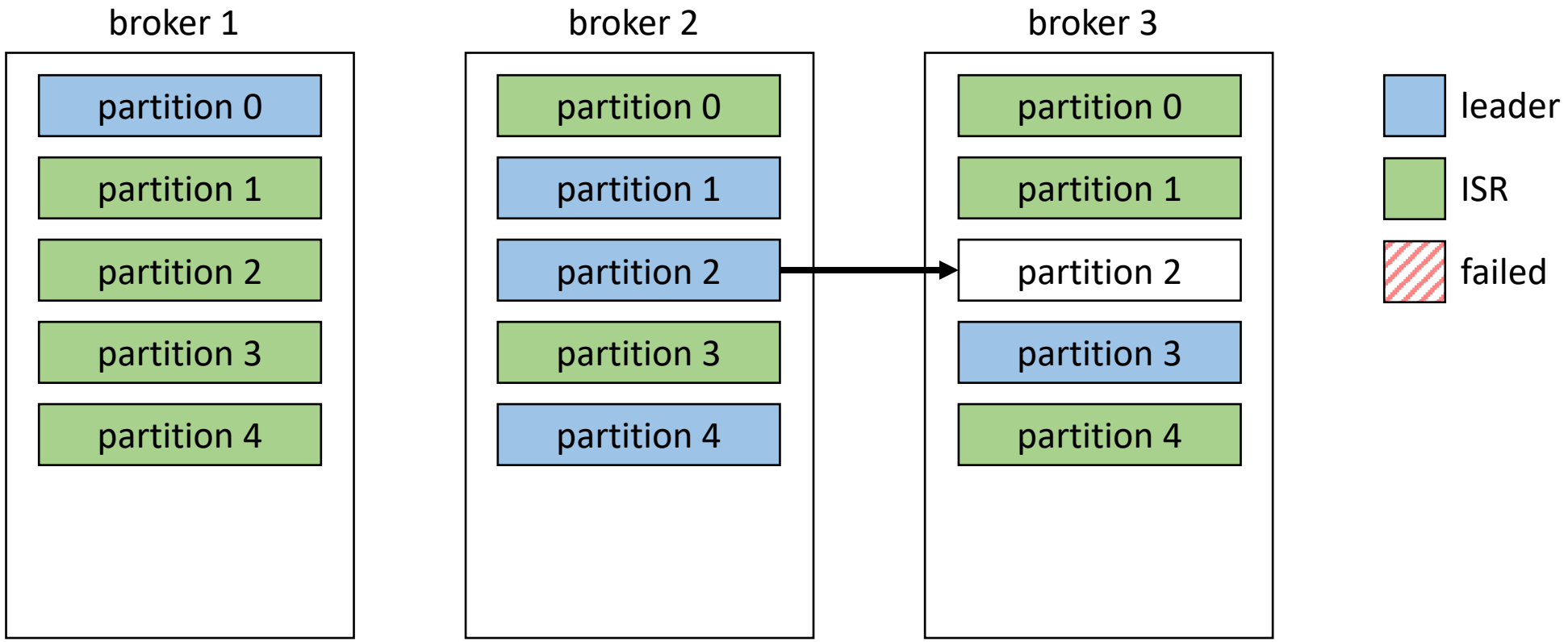


Архитектура Kafka Broker

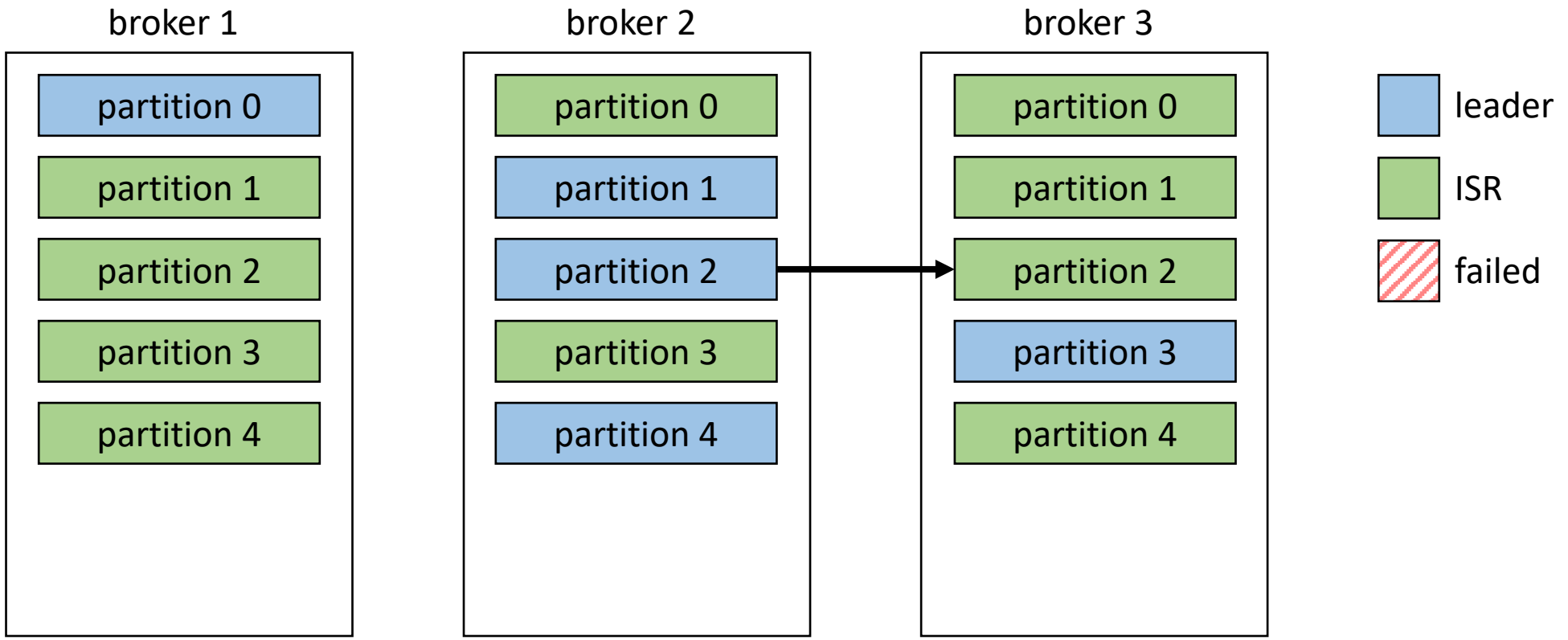


Архитектура Kafka Broker

Синхронизация реплики с *лидером* после восстановления

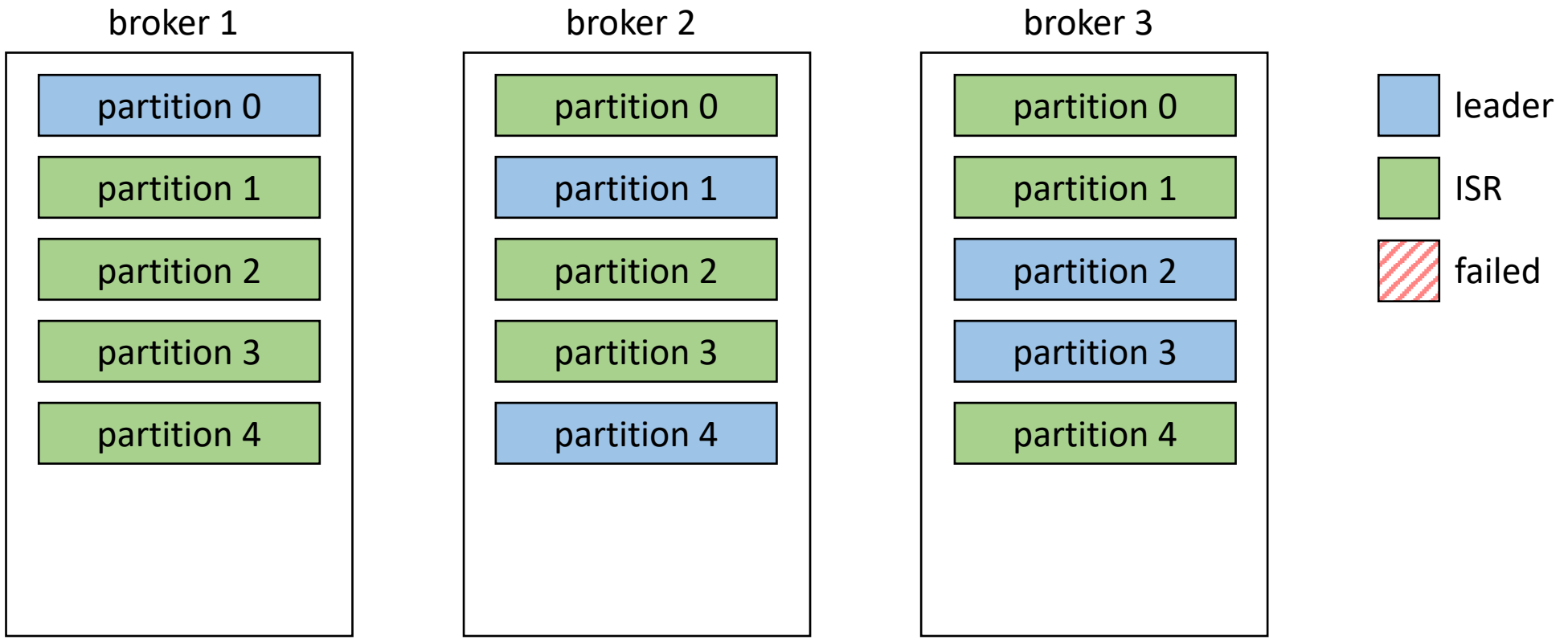


Архитектура Kafka Broker



Архитектура Kafka Broker

Перебалансировка *лидеров*



Архитектура Kafka Producer

Архитектура Kafka Producer

message = (key, value)

Архитектура Kafka Producer

message = (**key**, value)

Архитектура Kafka Producer

message = (**key**, value)

partition = murmur2(key) % partitions

Архитектура Kafka Producer

message = (**key**, value)

partition = murmur2(key) % partitions // key != null

Архитектура Kafka Producer

message = (**key**, value)

partition = murmur2(key) % partitions // key != null

https://ru.wikipedia.org/wiki/MurmurHash2

Архитектура Kafka Producer

message = (**key**, value)

partition = murmur2(key) % partitions // key != null

partition = round_robin(partitions)

Архитектура Kafka Producer

message = (**key**, value)

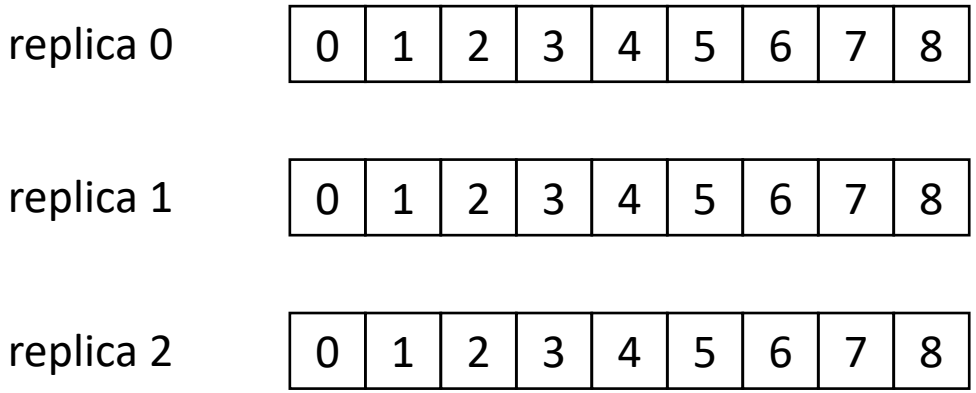
partition = murmur2(key) % partitions // key != null

partition = round_robin(partitions) // key == null

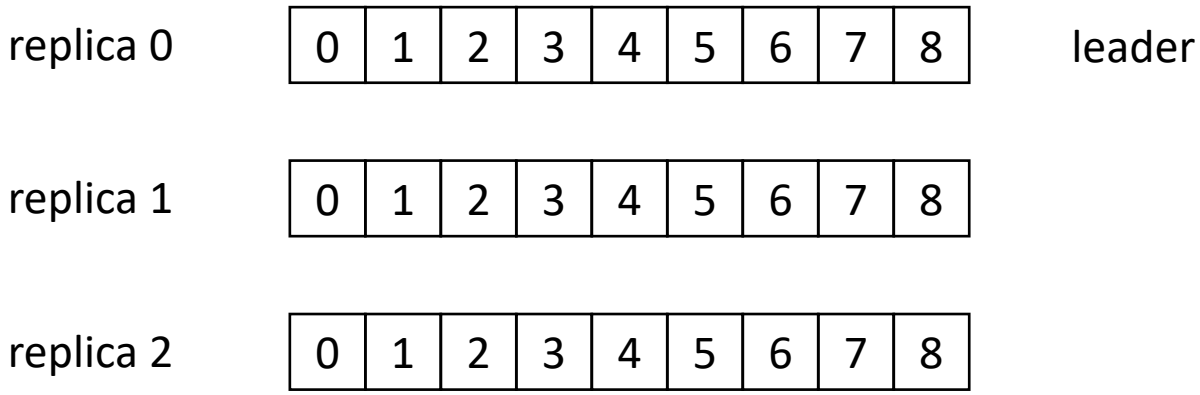
Архитектура Kafka Producer

message = (key, **value**)

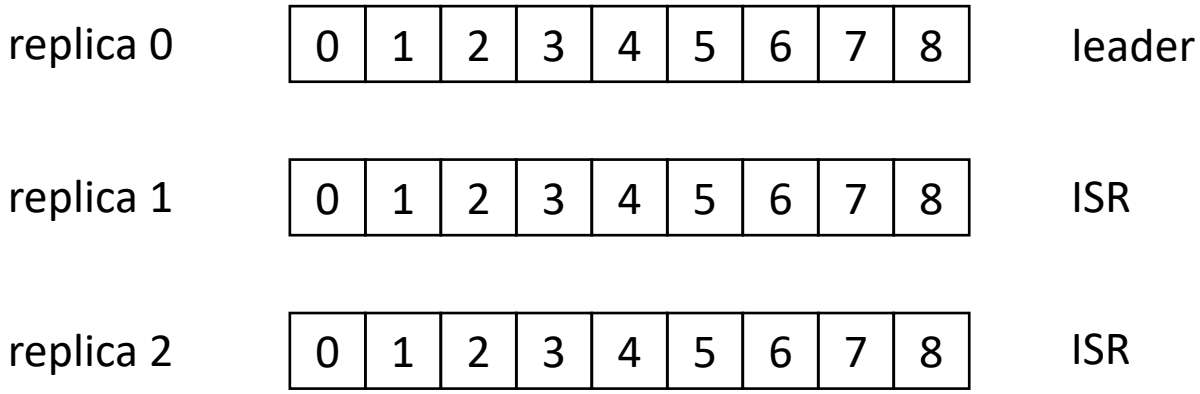
Архитектура Kafka Producer



Архитектура Kafka Producer

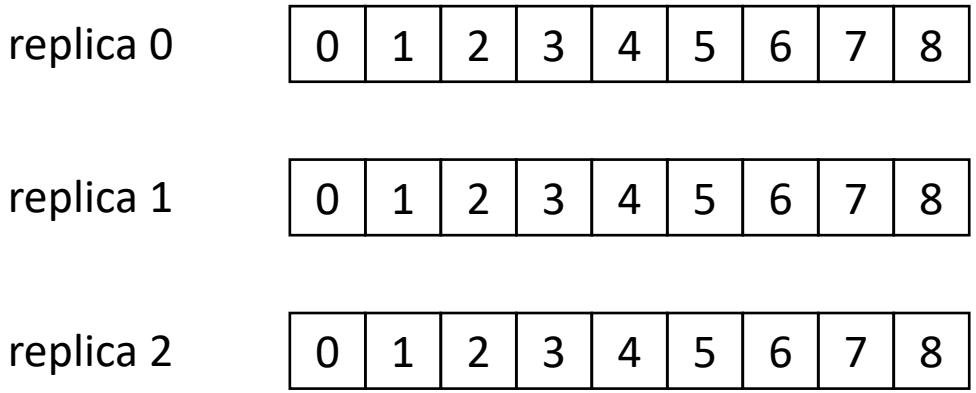


Архитектура Kafka Producer



Архитектура Kafka Producer

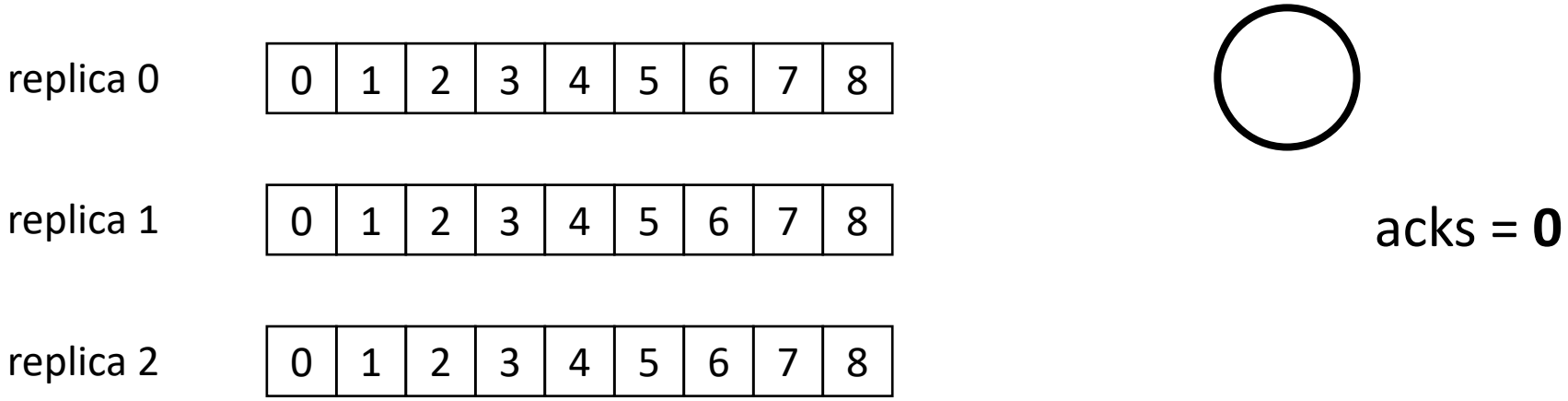
Acknowledgement (ack) – подтверждение записи



acks = **0**

Архитектура Kafka Producer

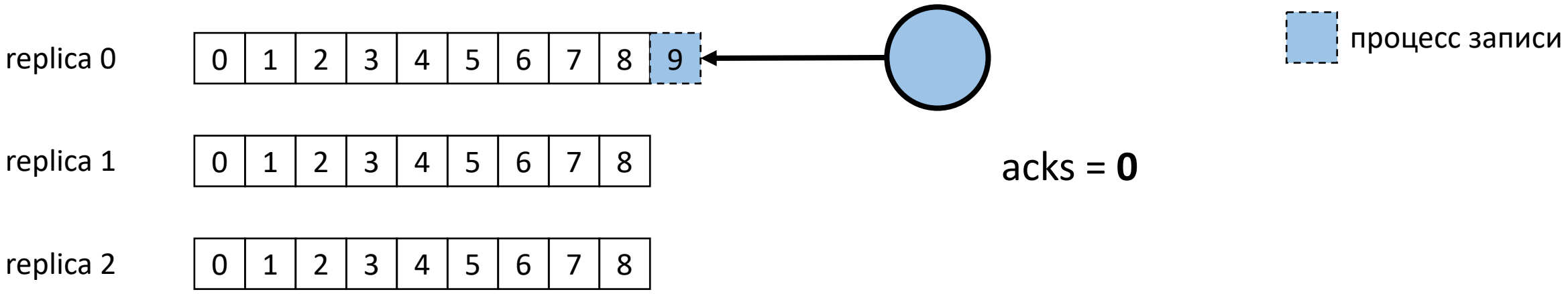
Acknowledgement (ack) – подтверждение записи



acks = **0**

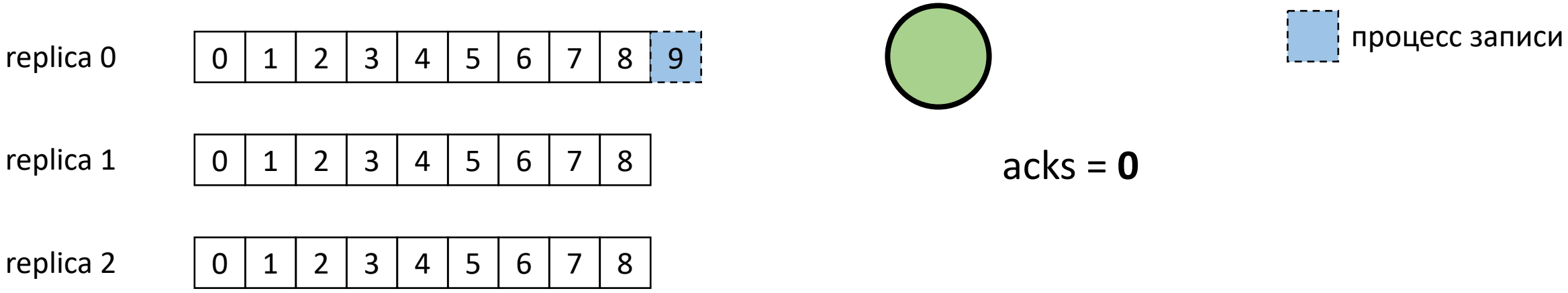
Архитектура Kafka Producer

Acknowledgement (ack) – подтверждение записи



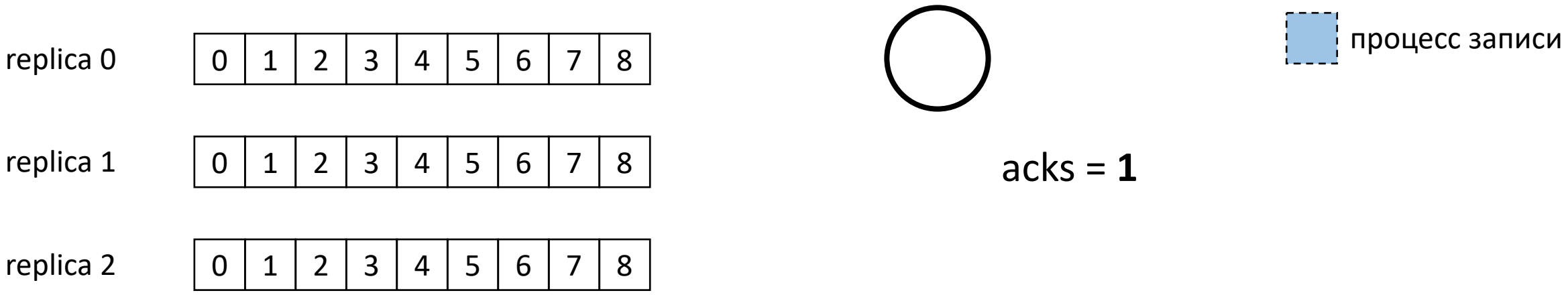
Архитектура Kafka Producer

Acknowledgement (ack) – подтверждение записи



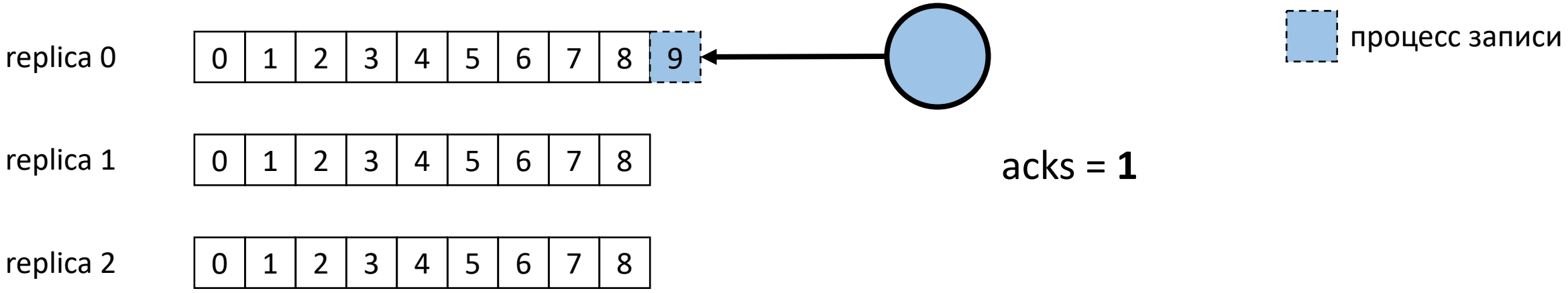
Архитектура Kafka Producer

Acknowledgement (ack) – подтверждение записи



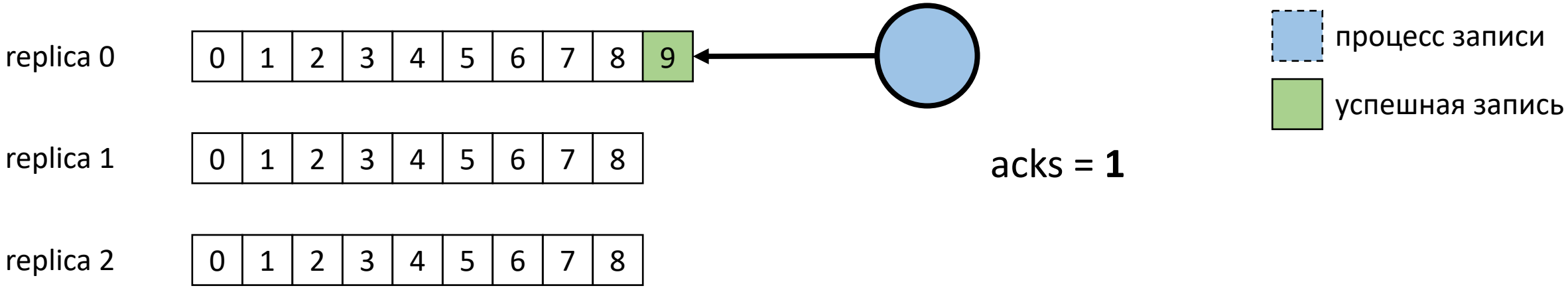
Архитектура Kafka Producer

Acknowledgement (ack) – подтверждение записи



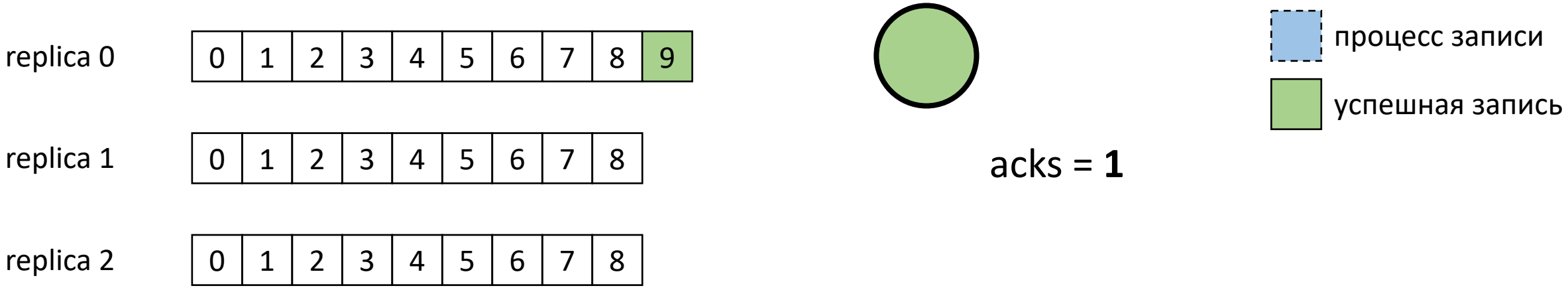
Архитектура Kafka Producer

Acknowledgement (ack) – подтверждение записи



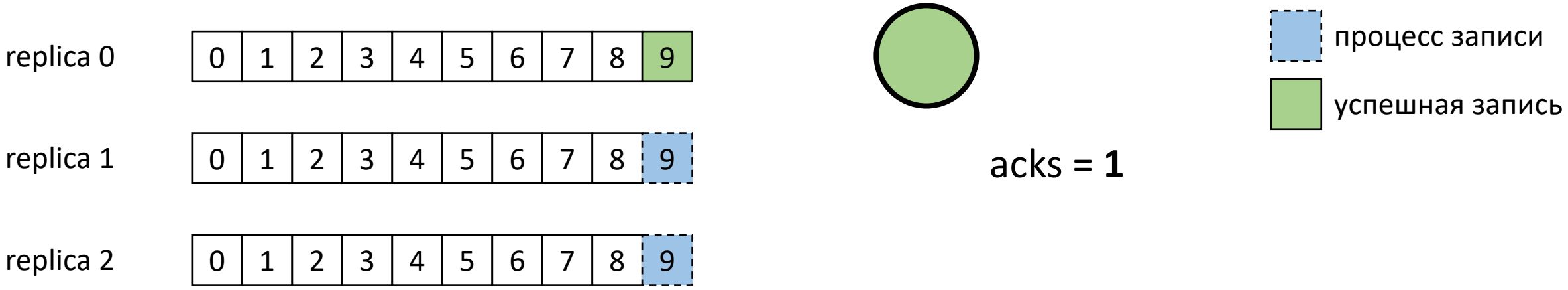
Архитектура Kafka Producer

Acknowledgement (ack) – подтверждение записи



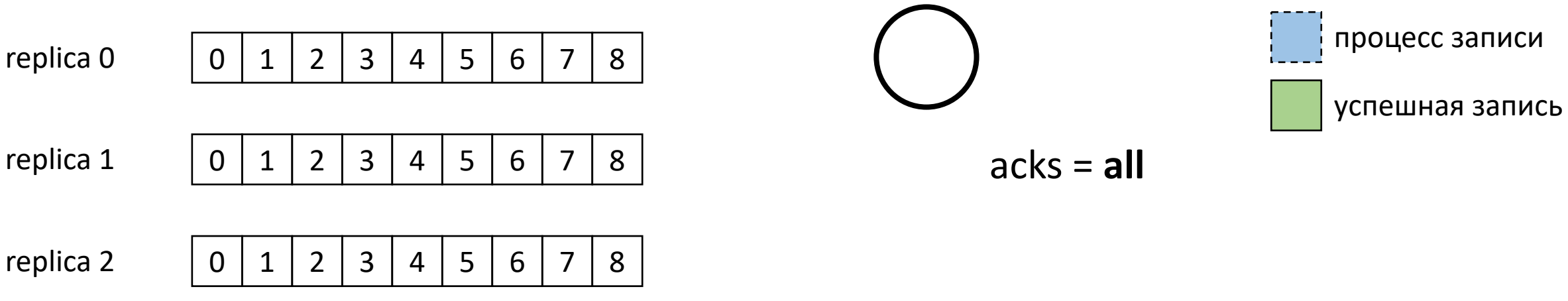
Архитектура Kafka Producer

Acknowledgement (ack) – подтверждение записи



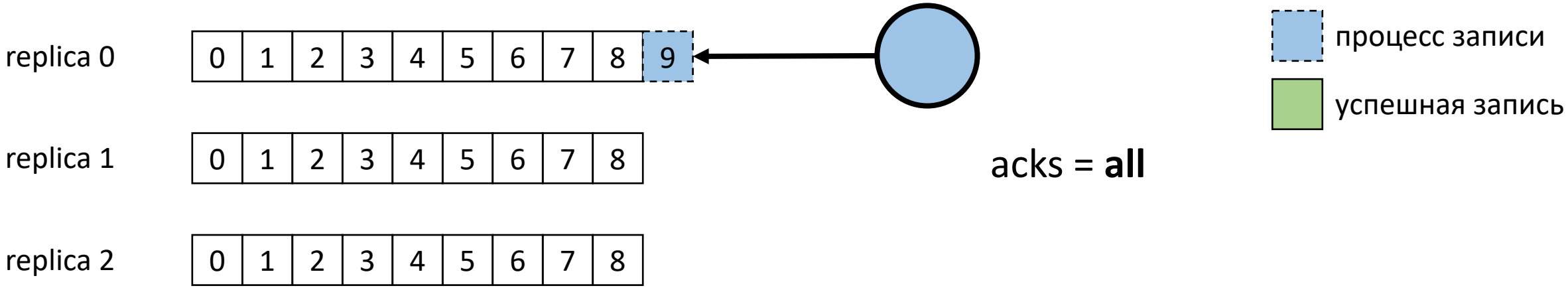
Архитектура Kafka Producer

Acknowledgement (ack) – подтверждение записи



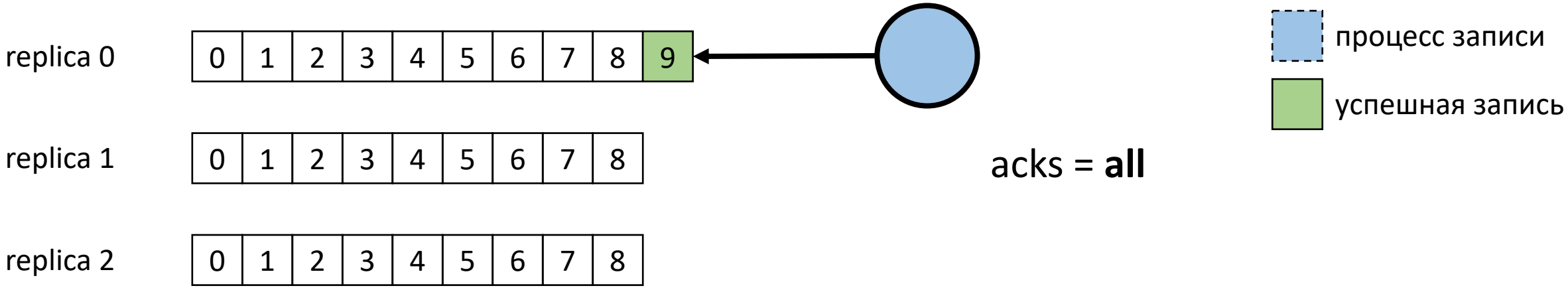
Архитектура Kafka Producer

Acknowledgement (ack) – подтверждение записи



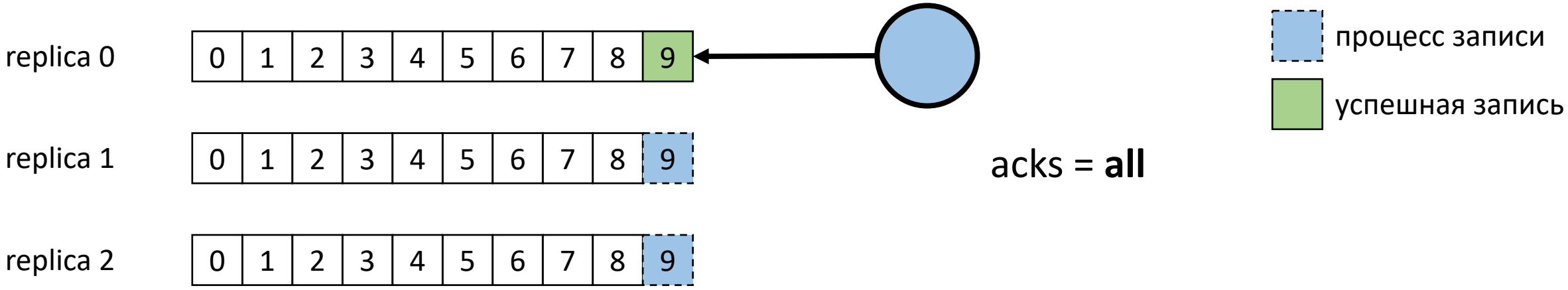
Архитектура Kafka Producer

Acknowledgement (ack) – подтверждение записи



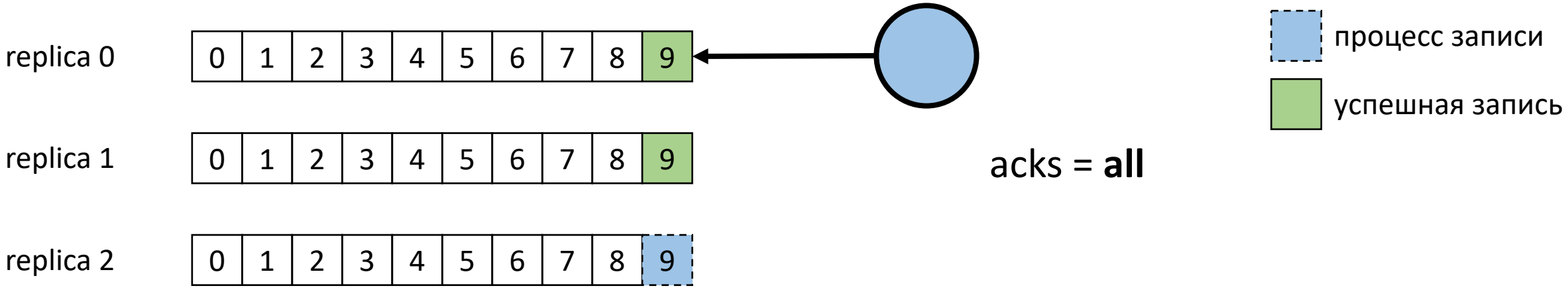
Архитектура Kafka Producer

Acknowledgement (ack) – подтверждение записи



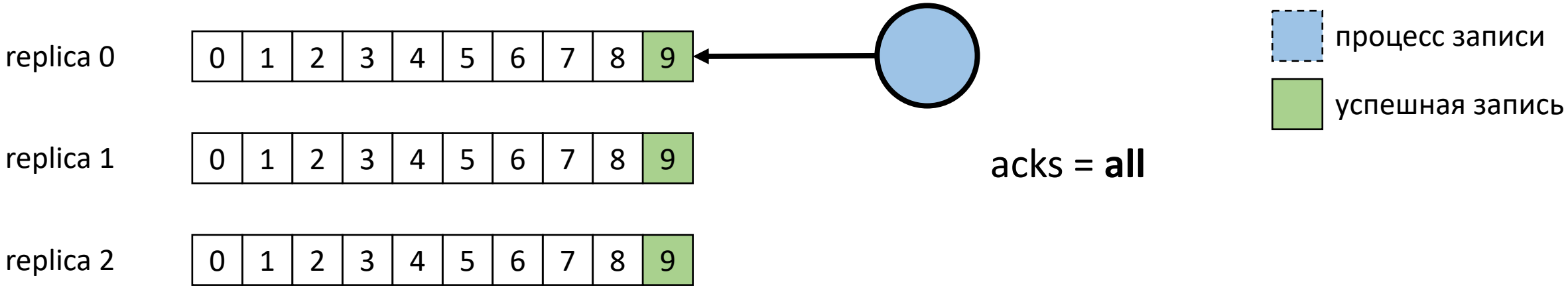
Архитектура Kafka Producer

Acknowledgement (ack) – подтверждение записи



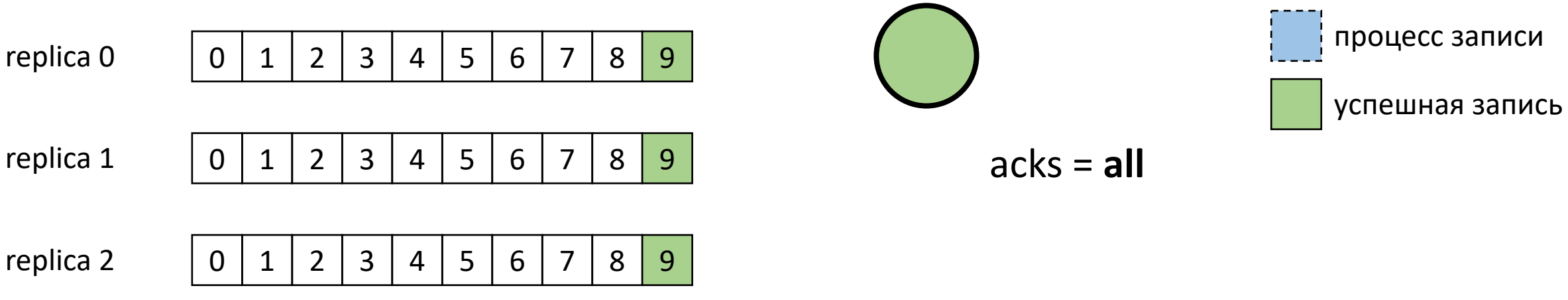
Архитектура Kafka Producer

Acknowledgement (ack) – подтверждение записи



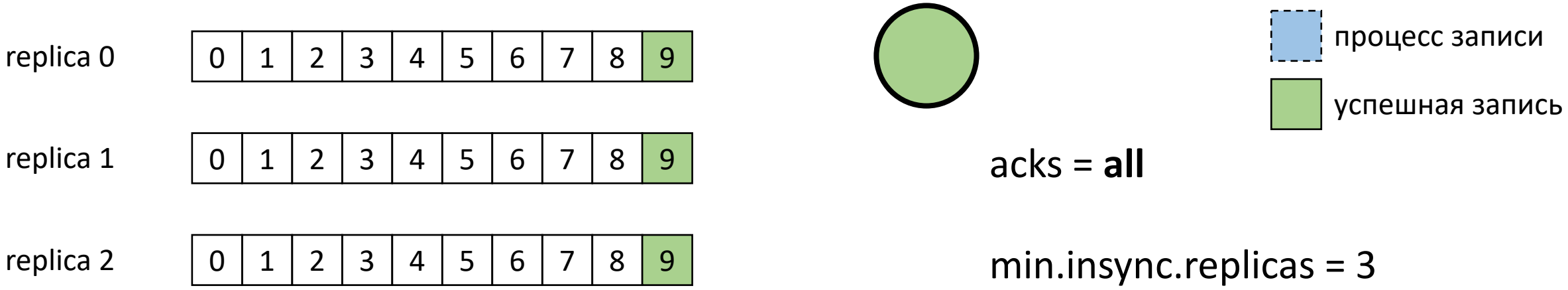
Архитектура Kafka Producer

Acknowledgement (ack) – подтверждение записи



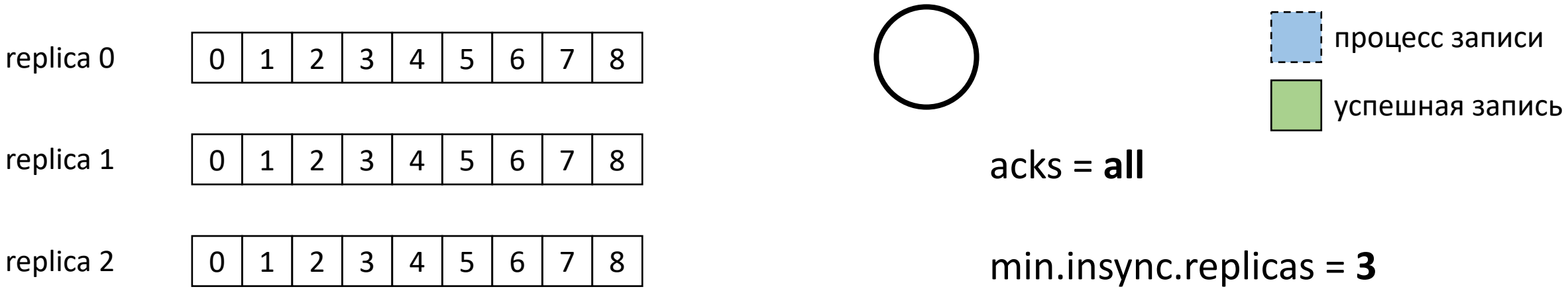
Архитектура Kafka Producer

Acknowledgement (ack) – подтверждение записи



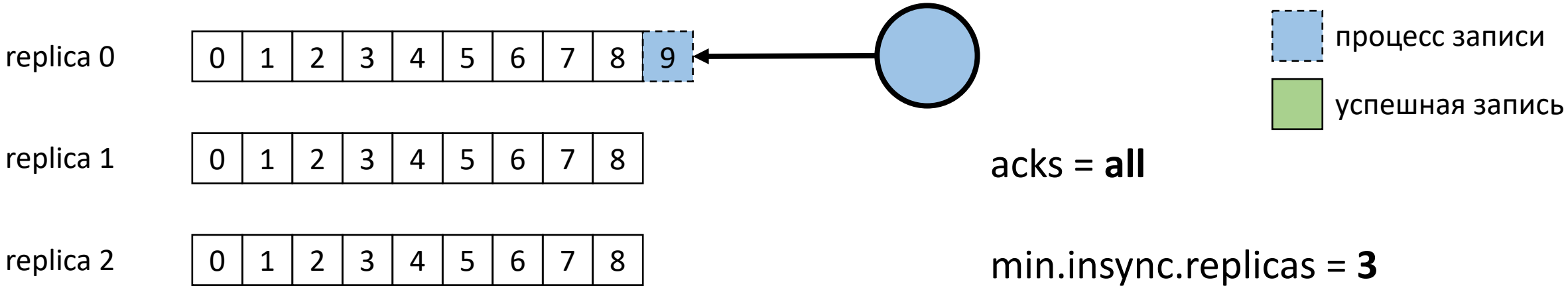
Архитектура Kafka Producer

Acknowledgement (ack) – подтверждение записи



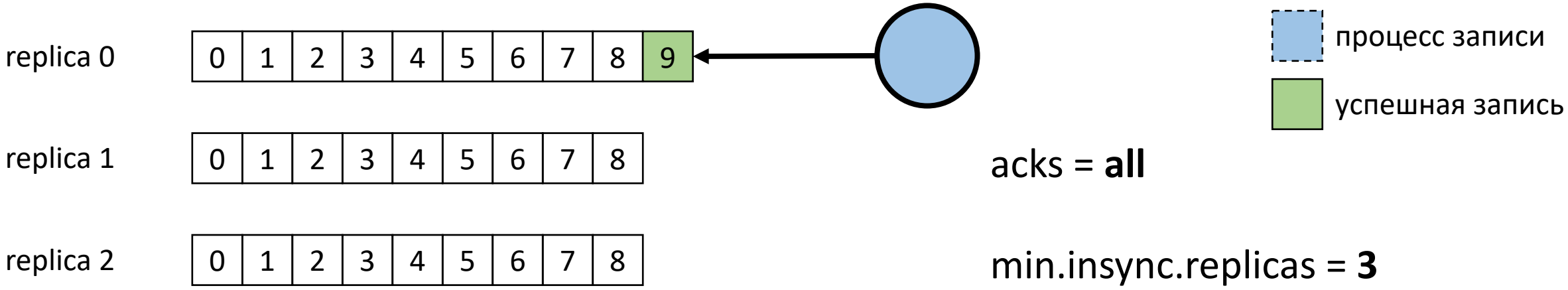
Архитектура Kafka Producer

Acknowledgement (ack) – подтверждение записи



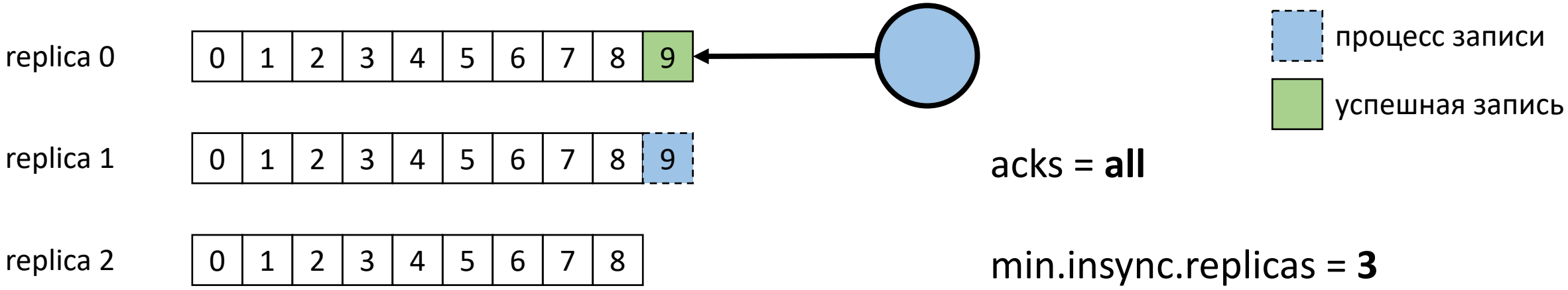
Архитектура Kafka Producer

Acknowledgement (ack) – подтверждение записи



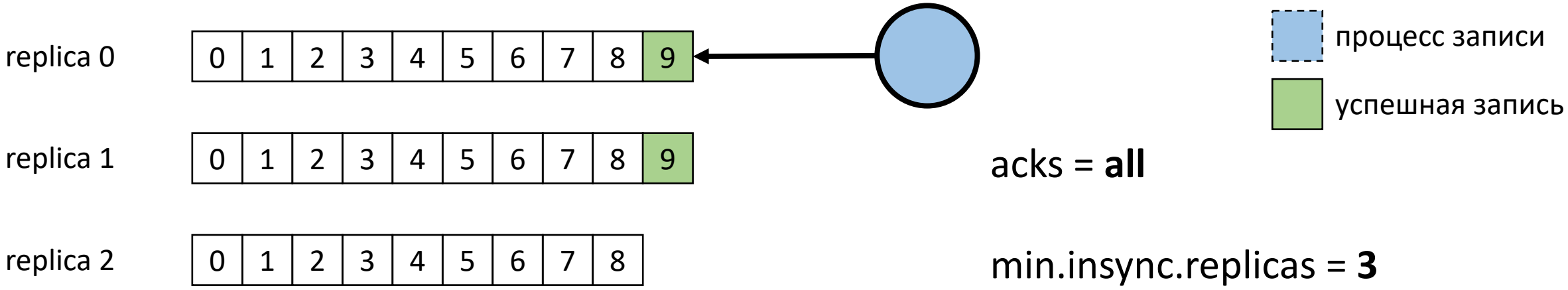
Архитектура Kafka Producer

Acknowledgement (ack) – подтверждение записи



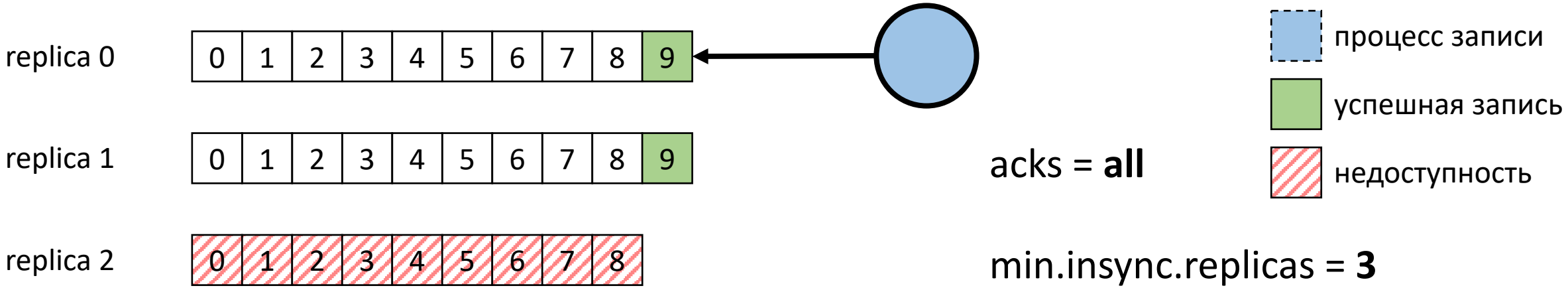
Архитектура Kafka Producer

Acknowledgement (ack) – подтверждение записи



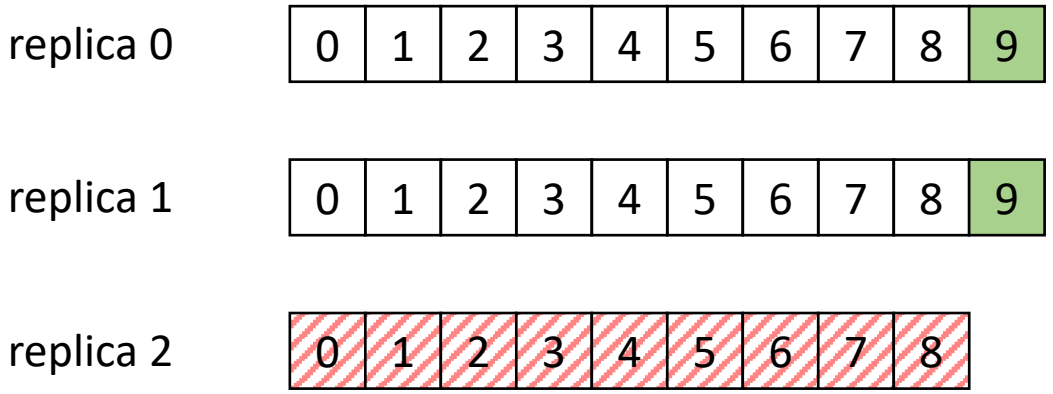
Архитектура Kafka Producer

Acknowledgement (ack) – подтверждение записи



Архитектура Kafka Producer

Acknowledgement (ack) – подтверждение записи



процесс записи

успешная запись

недоступность

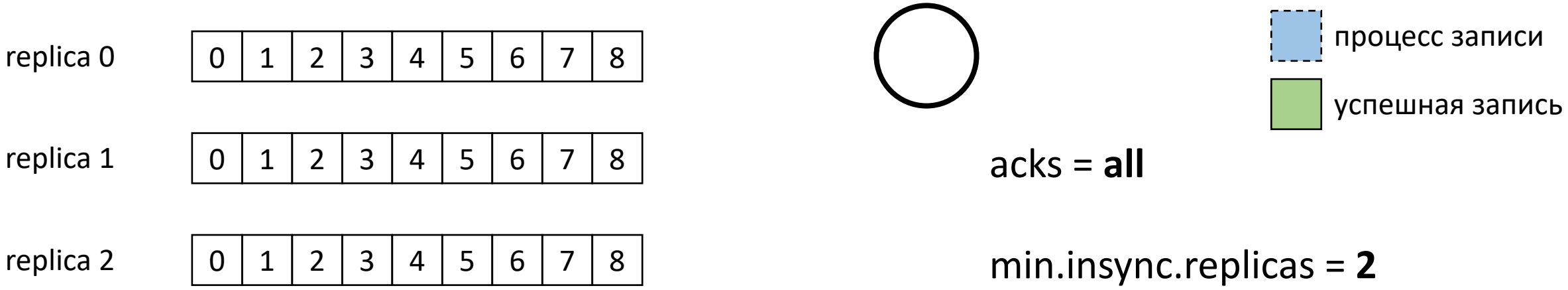
acks = **all**

min.insync.replicas = **3**

request.timeout.ms = 30_000

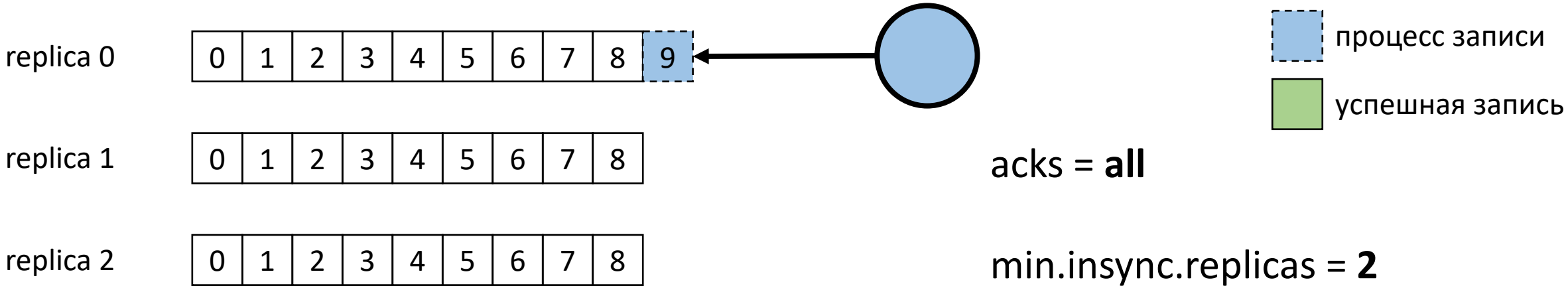
Архитектура Kafka Producer

Acknowledgement (ack) – подтверждение записи



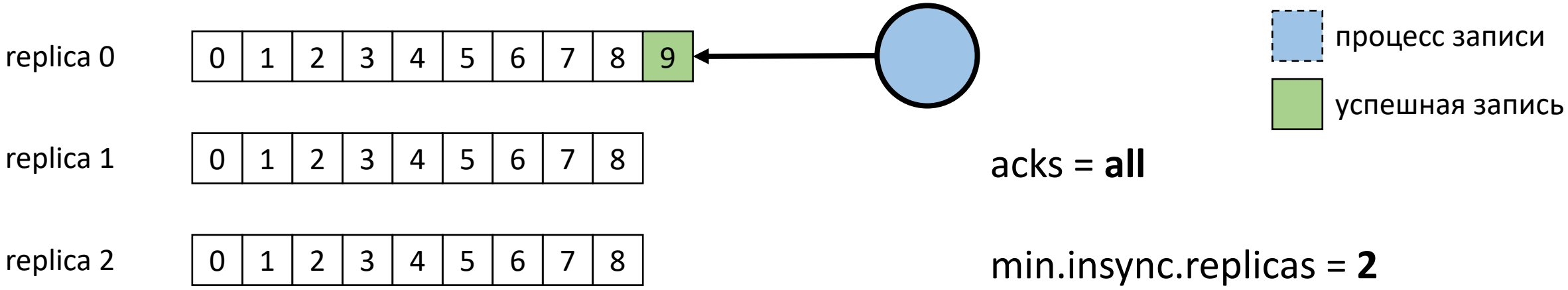
Архитектура Kafka Producer

Acknowledgement (ack) – подтверждение записи



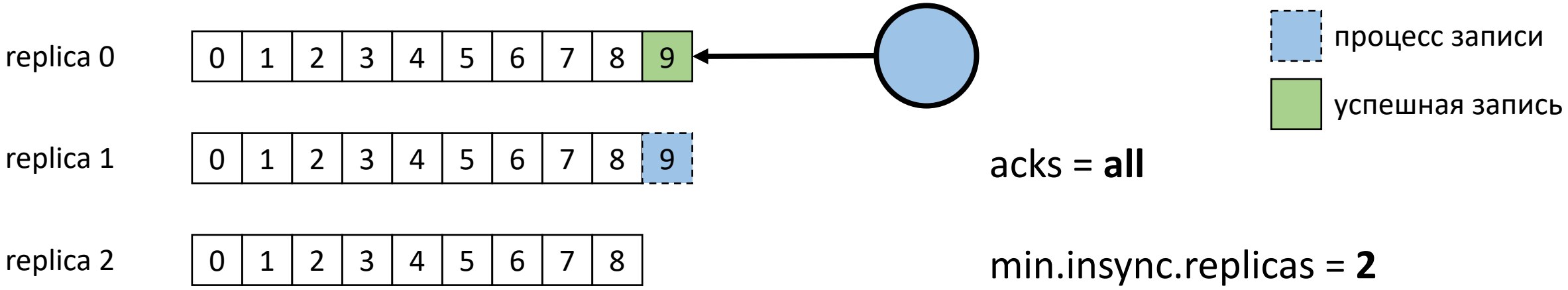
Архитектура Kafka Producer

Acknowledgement (ack) – подтверждение записи



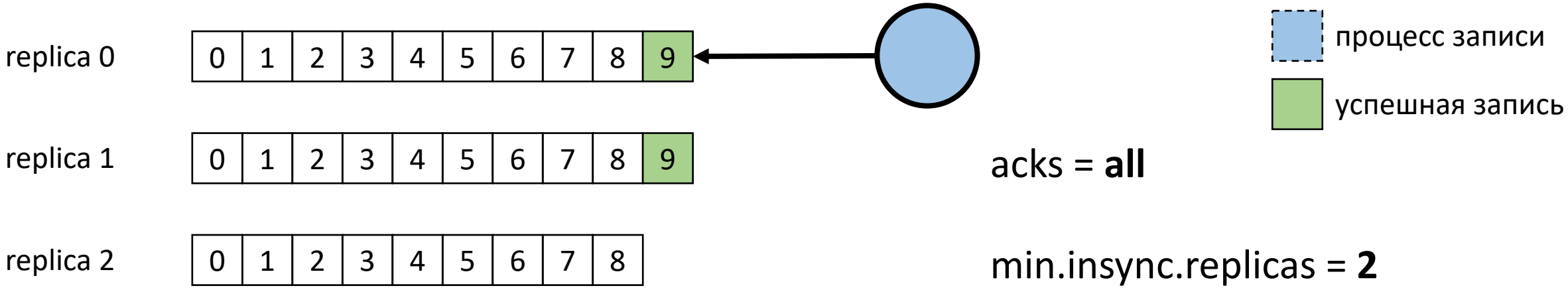
Архитектура Kafka Producer

Acknowledgement (ack) – подтверждение записи



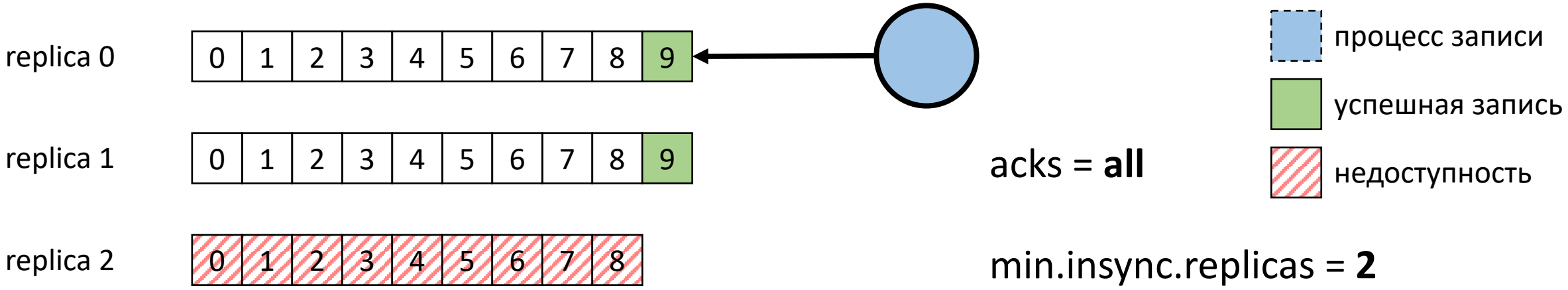
Архитектура Kafka Producer

Acknowledgement (ack) – подтверждение записи



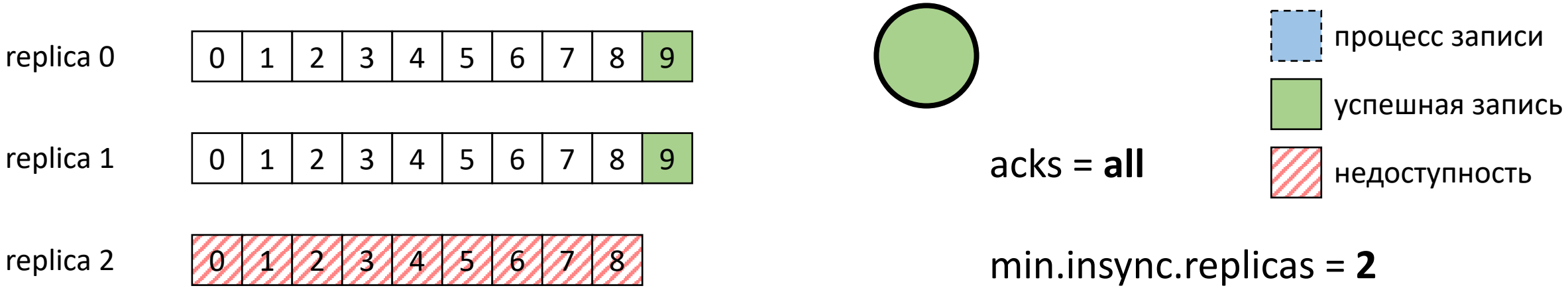
Архитектура Kafka Producer

Acknowledgement (ack) – подтверждение записи



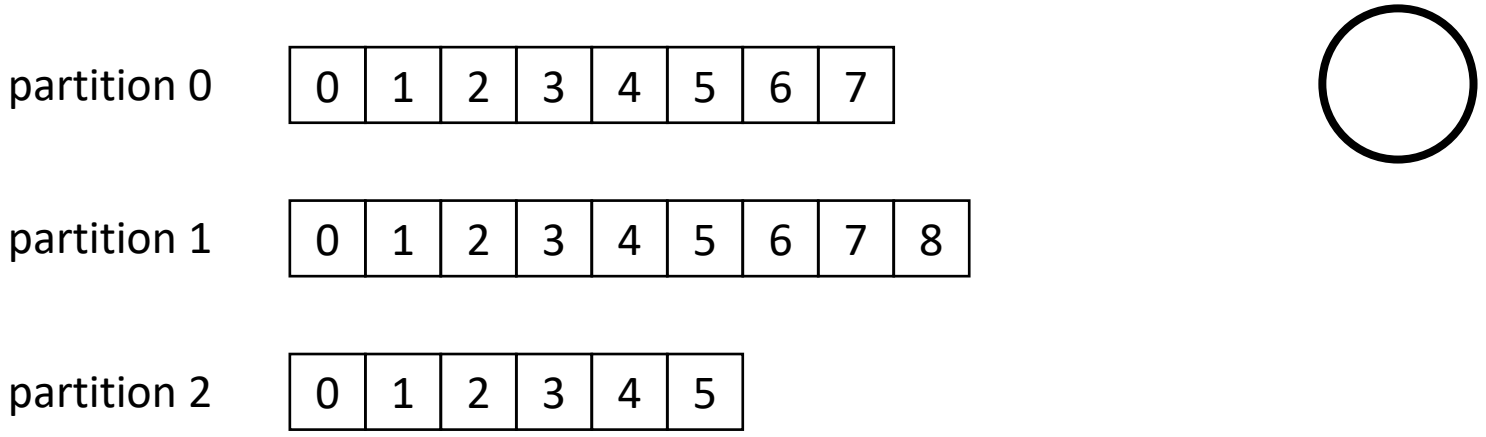
Архитектура Kafka Producer

Acknowledgement (ack) – подтверждение записи

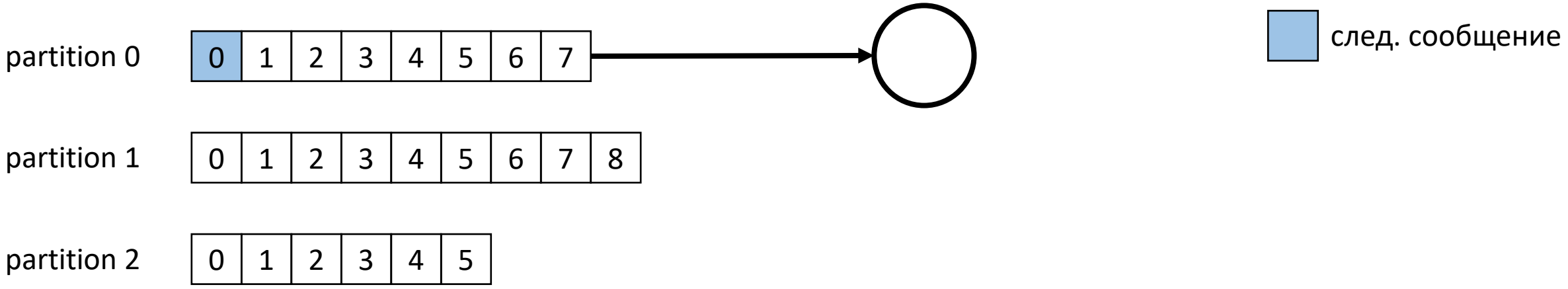


Архитектура Kafka Consumer

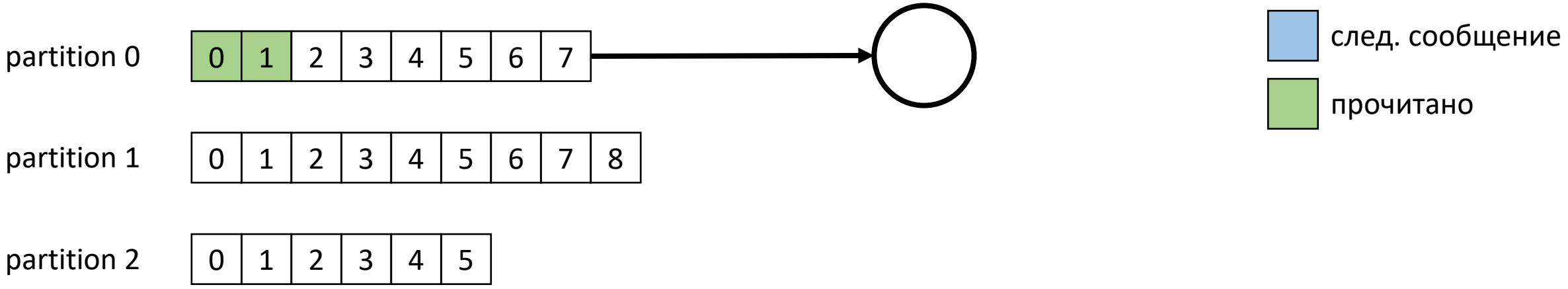
Архитектура Kafka Consumer



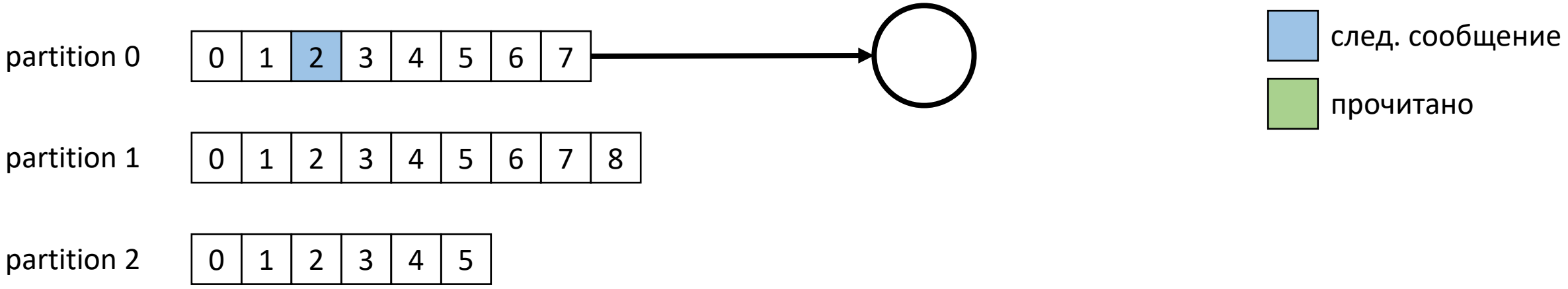
Архитектура Kafka Consumer



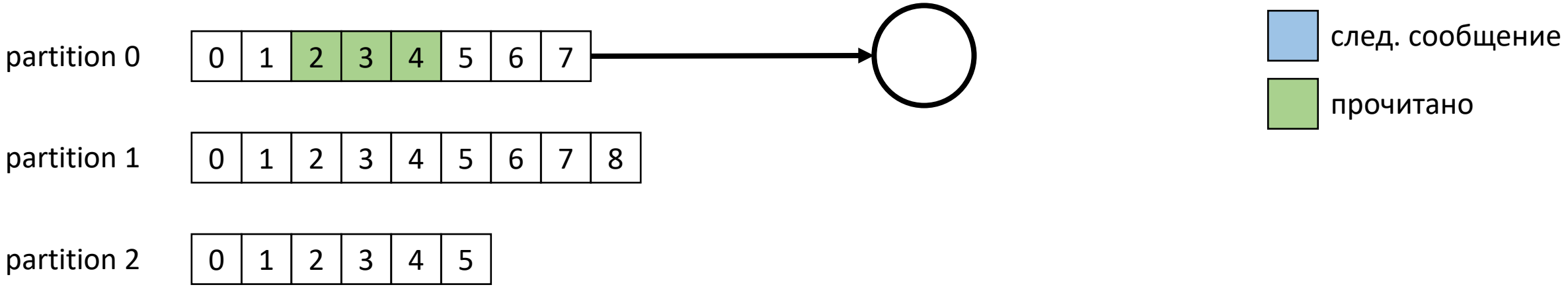
Архитектура Kafka Consumer



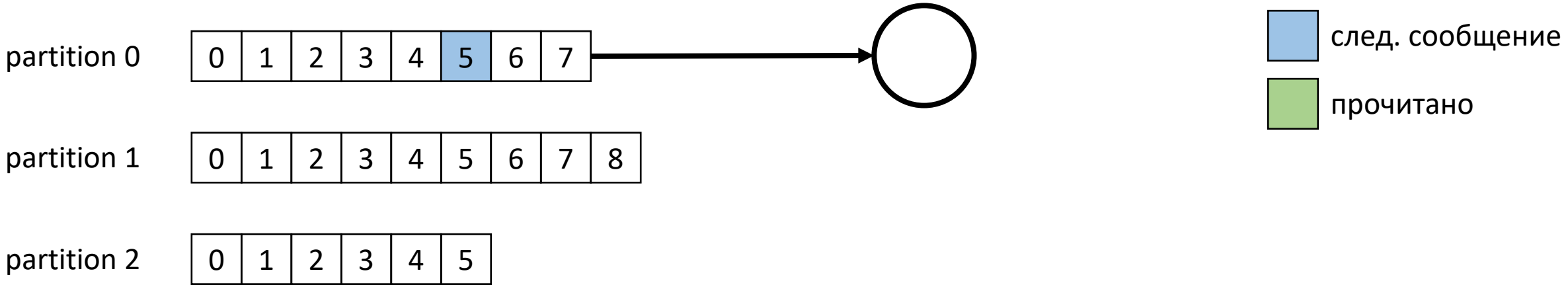
Архитектура Kafka Consumer



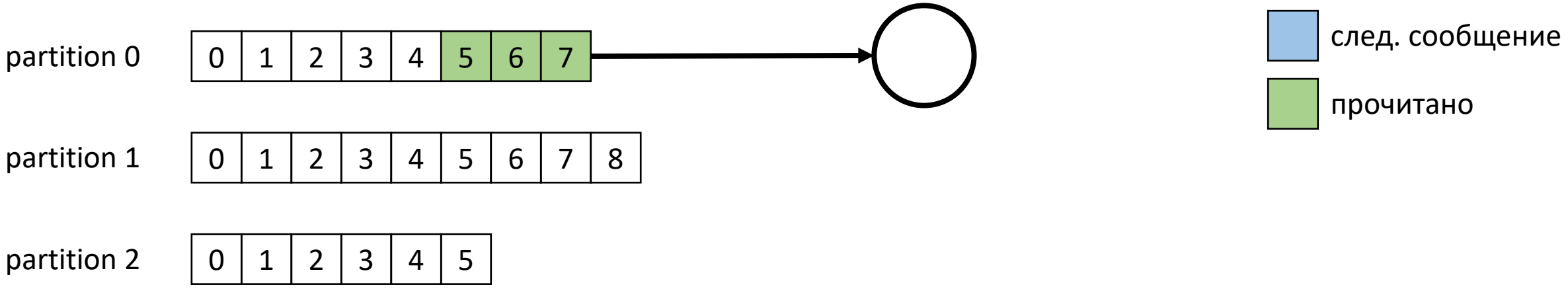
Архитектура Kafka Consumer



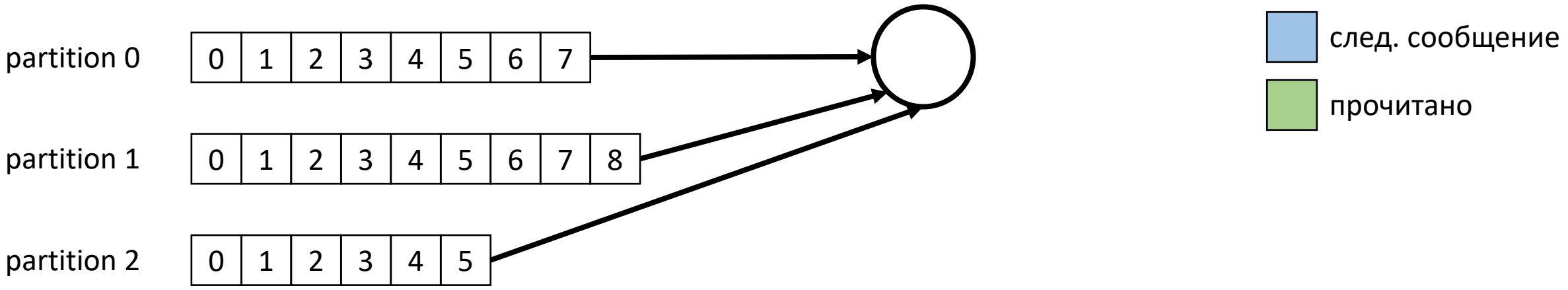
Архитектура Kafka Consumer



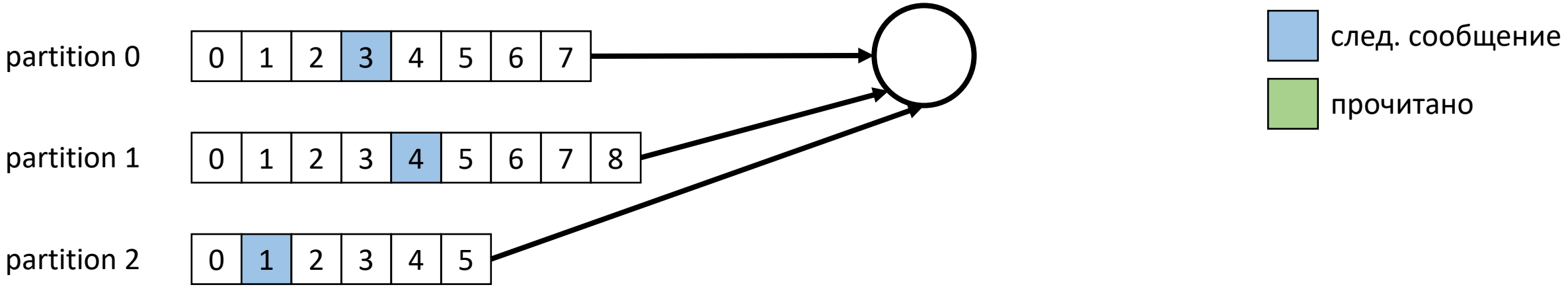
Архитектура Kafka Consumer



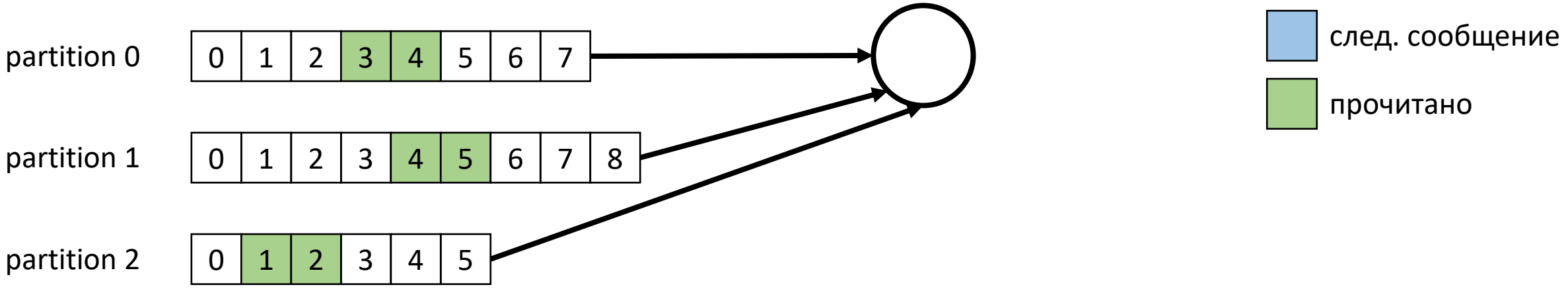
Архитектура Kafka Consumer



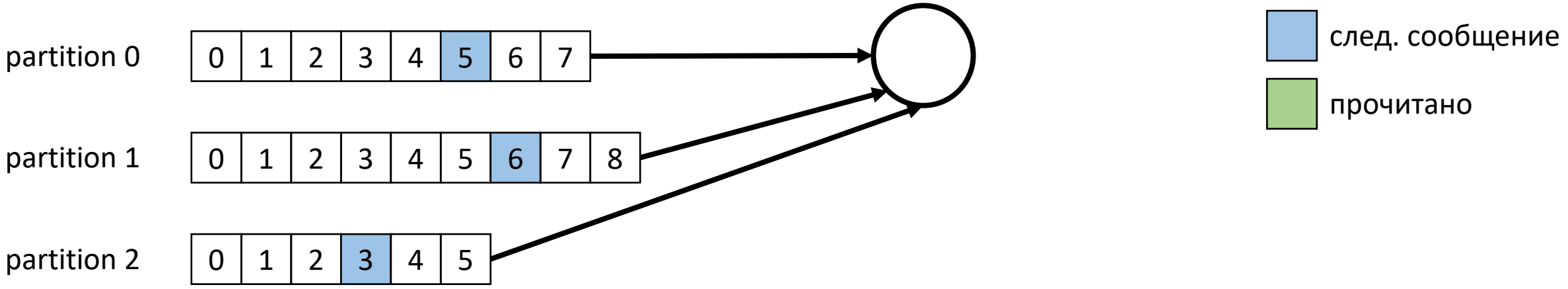
Архитектура Kafka Consumer



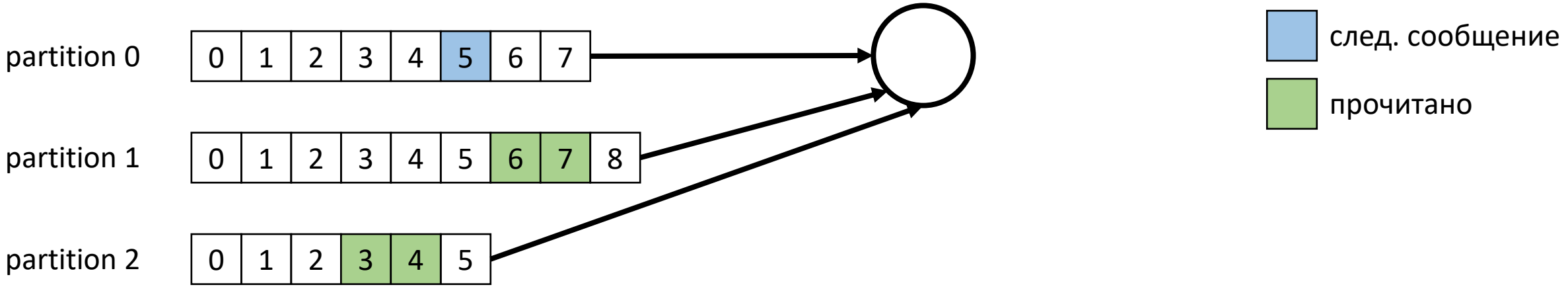
Архитектура Kafka Consumer



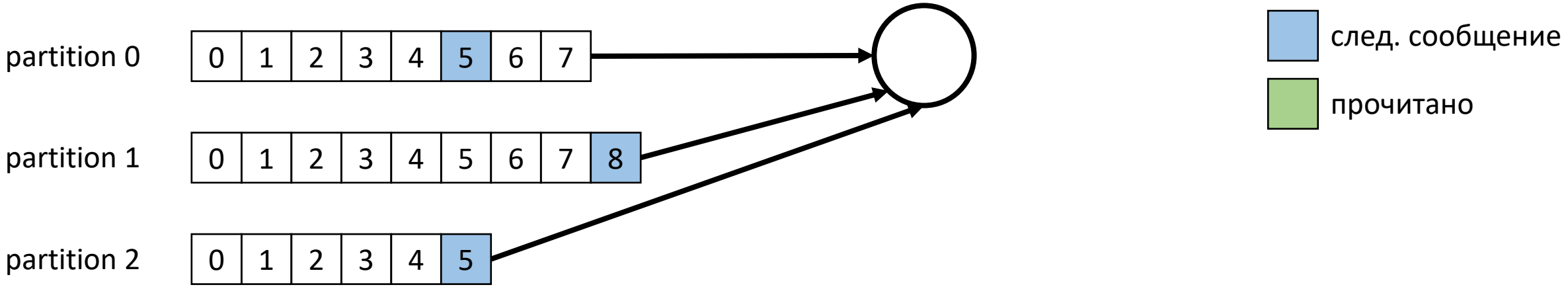
Архитектура Kafka Consumer



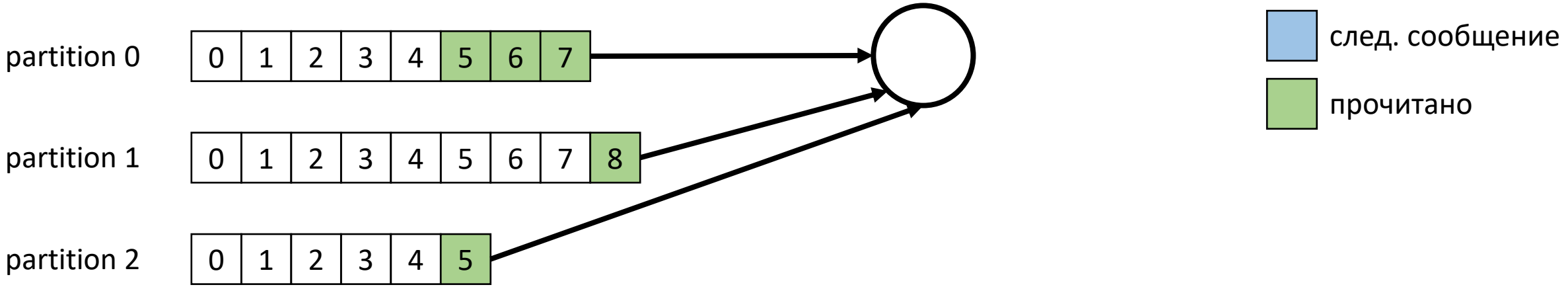
Архитектура Kafka Consumer



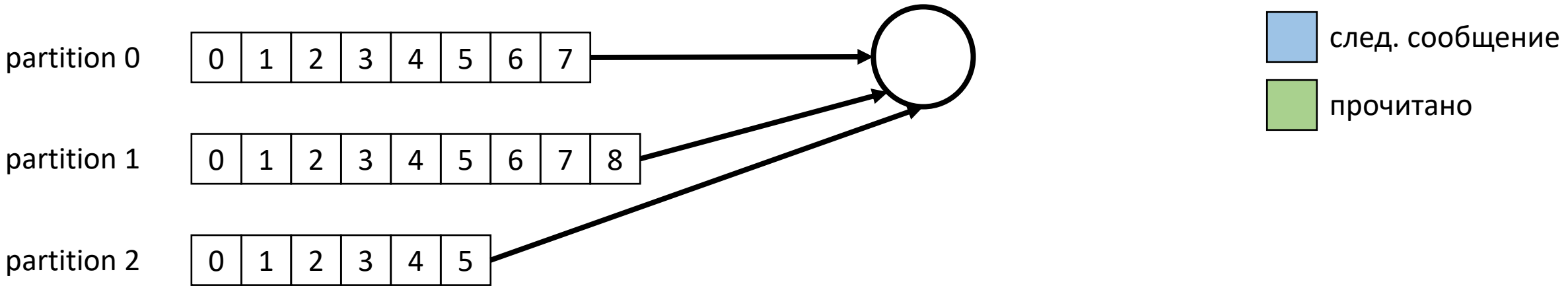
Архитектура Kafka Consumer



Архитектура Kafka Consumer



Архитектура Kafka Consumer

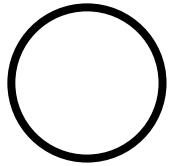


Архитектура Kafka Consumer

Commit offset

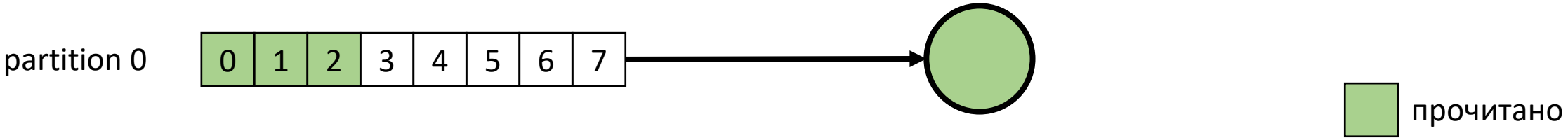
partition 0

0	1	2	3	4	5	6	7



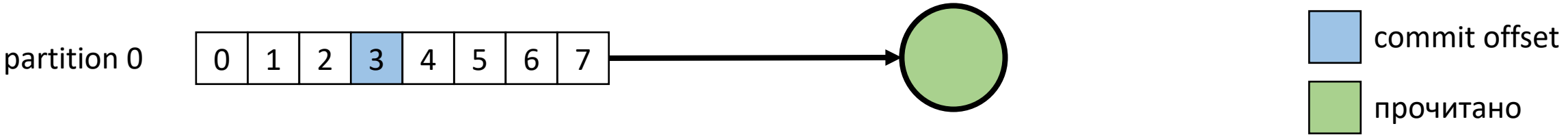
Архитектура Kafka Consumer

Commit offset



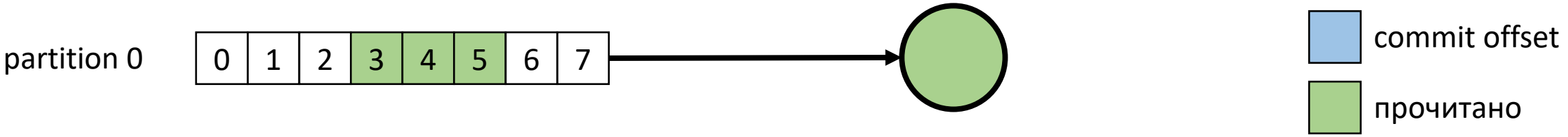
Архитектура Kafka Consumer

Commit offset



Архитектура Kafka Consumer

Commit offset



Архитектура Kafka Consumer

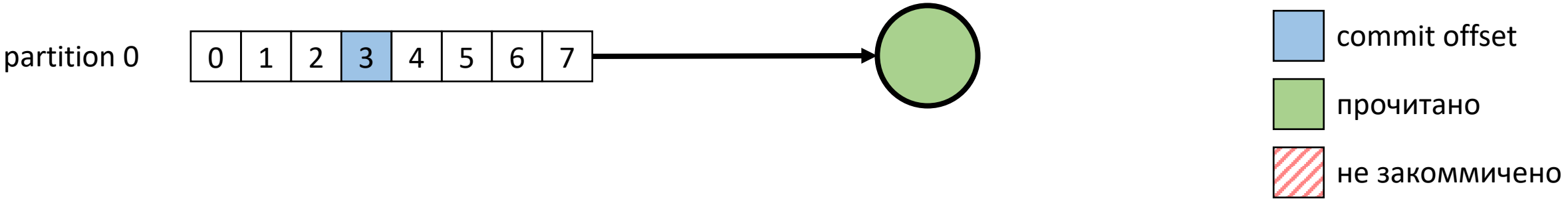
Commit offset

Архитектура Kafka Consumer

Commit offset

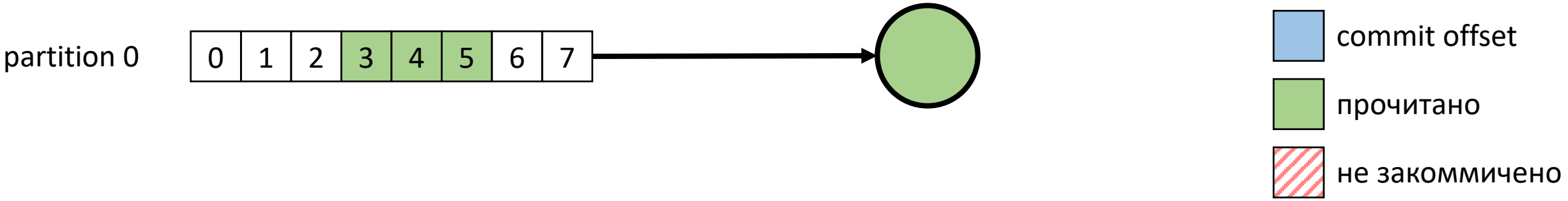
Архитектура Kafka Consumer

Commit offset



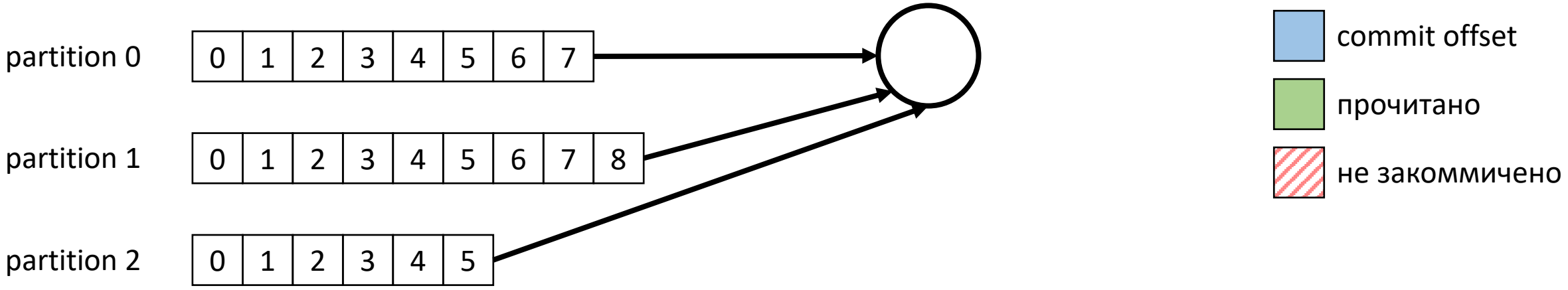
Архитектура Kafka Consumer

Commit offset



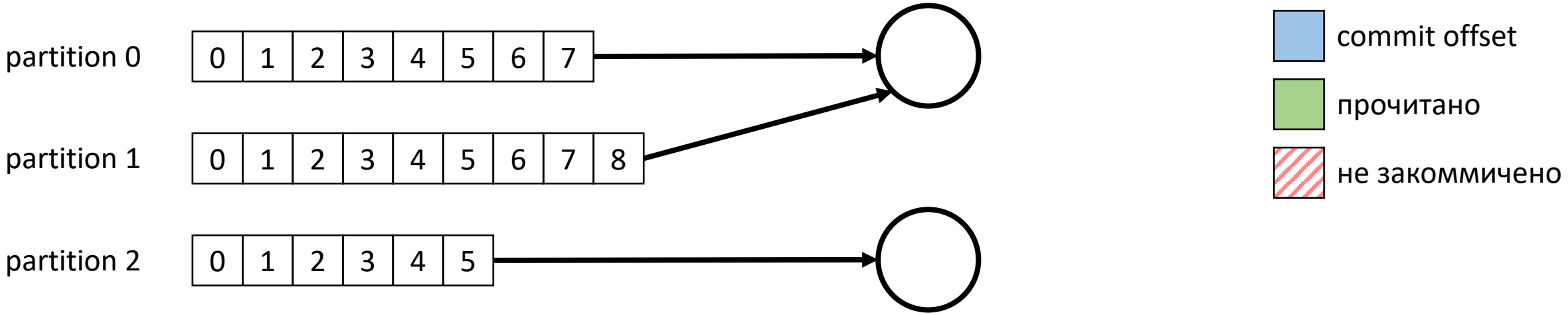
Архитектура Kafka Consumer

Consumer Group



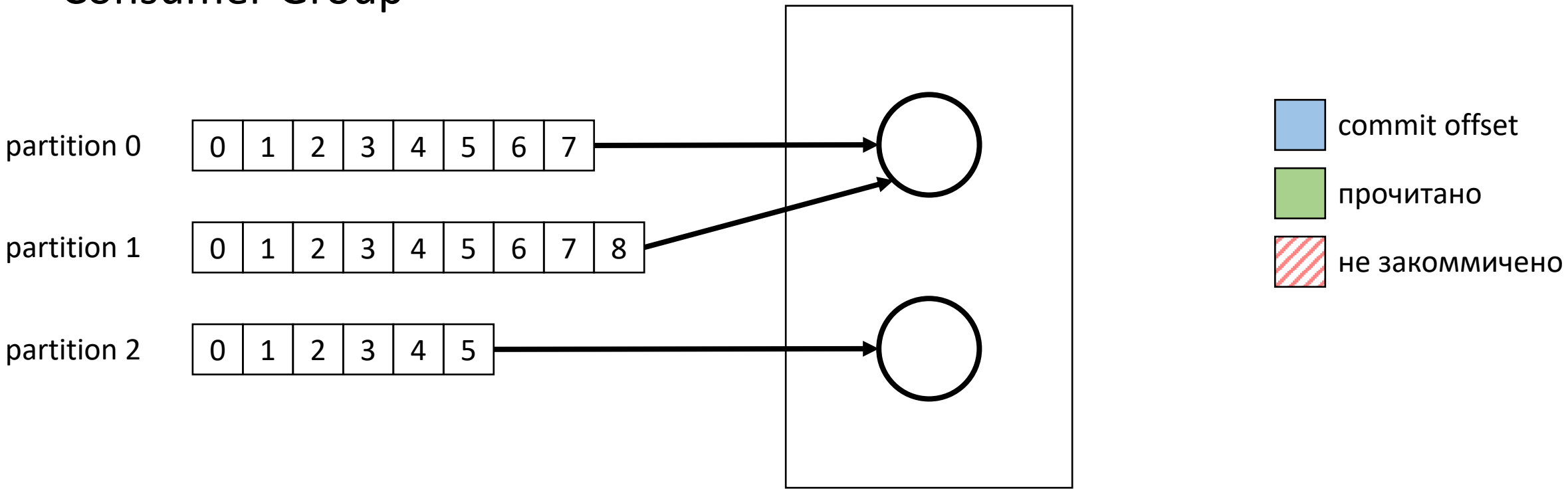
Архитектура Kafka Consumer

Consumer Group



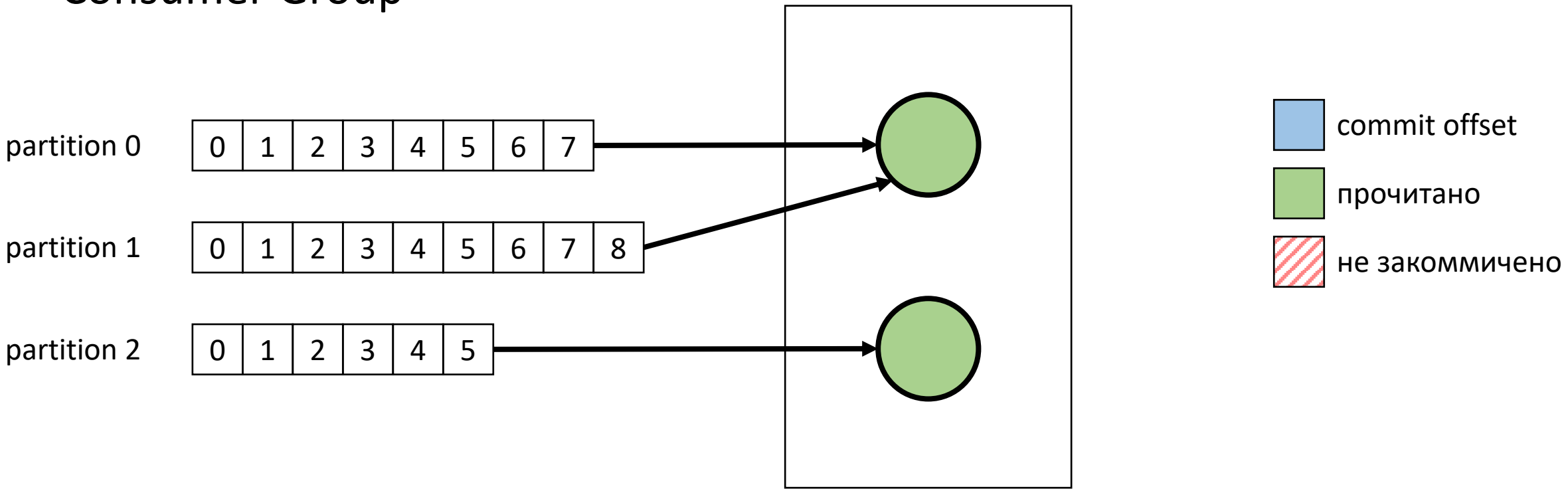
Архитектура Kafka Consumer

Consumer Group

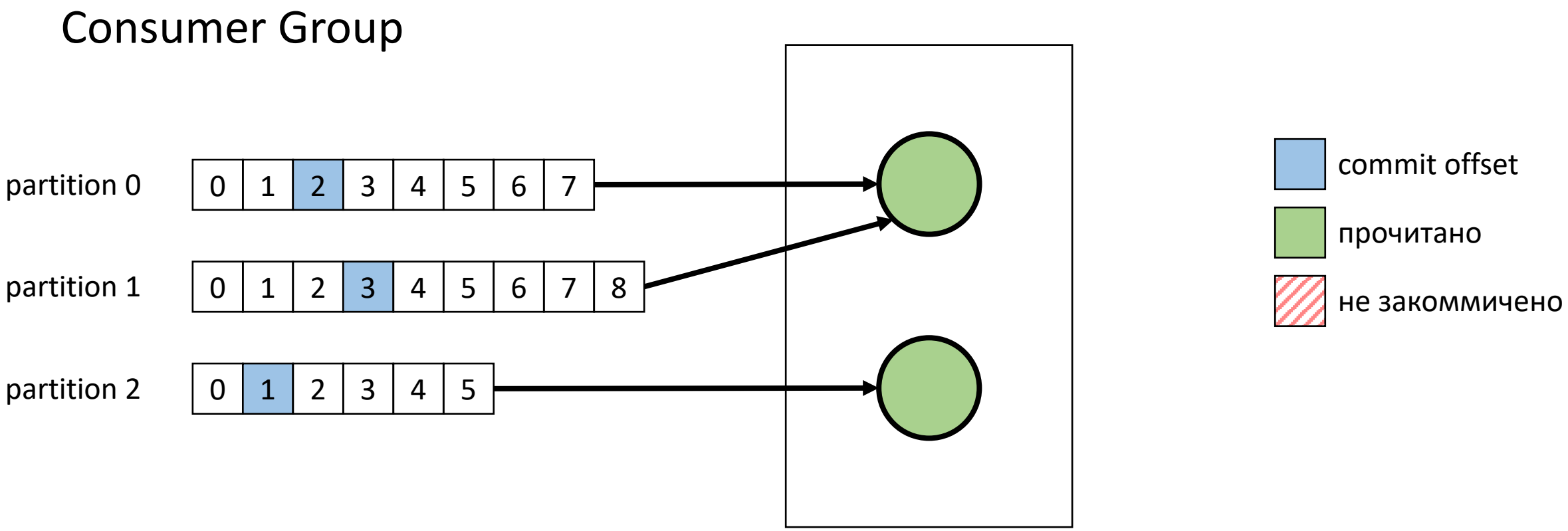


Архитектура Kafka Consumer

Consumer Group

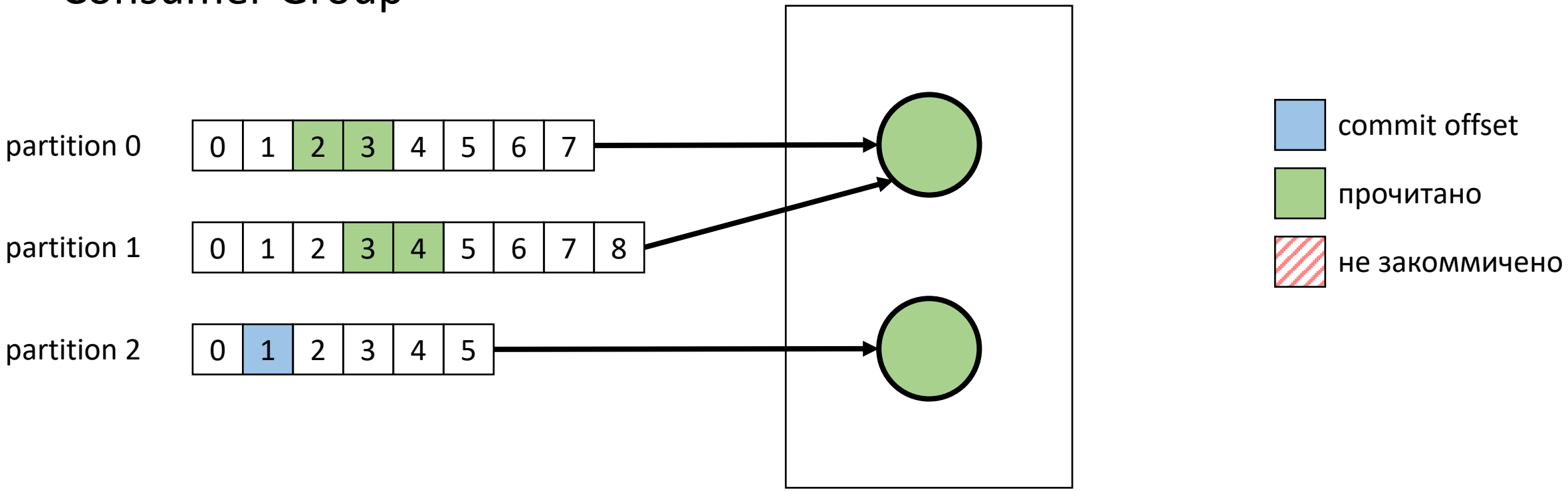


Архитектура Kafka Consumer



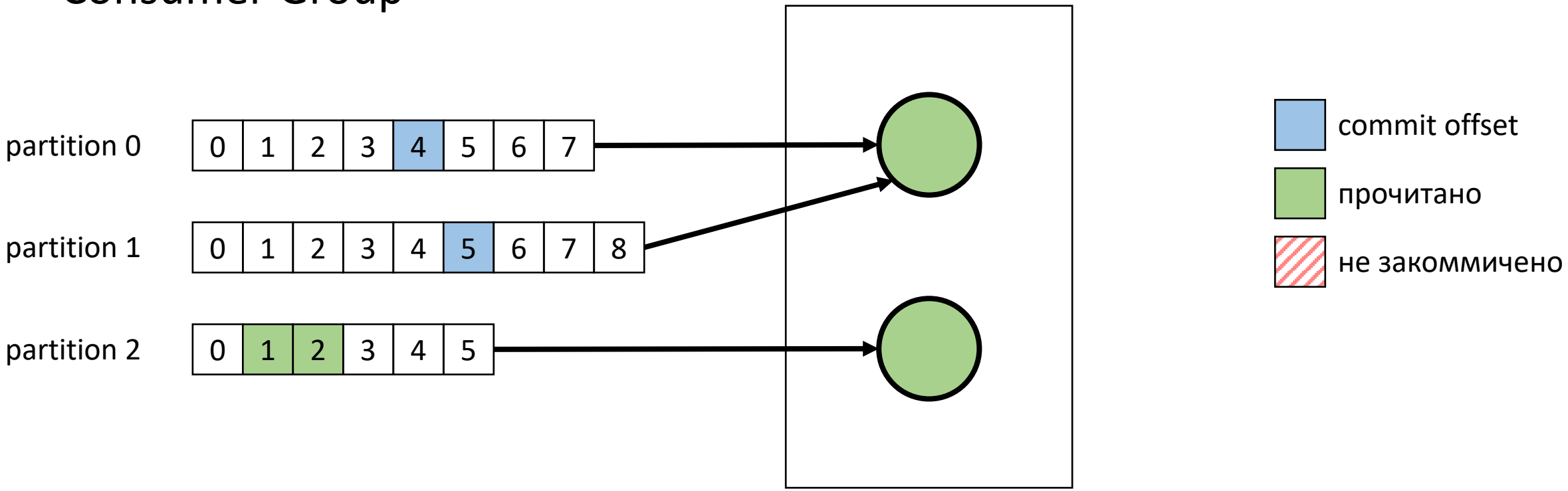
Архитектура Kafka Consumer

Consumer Group



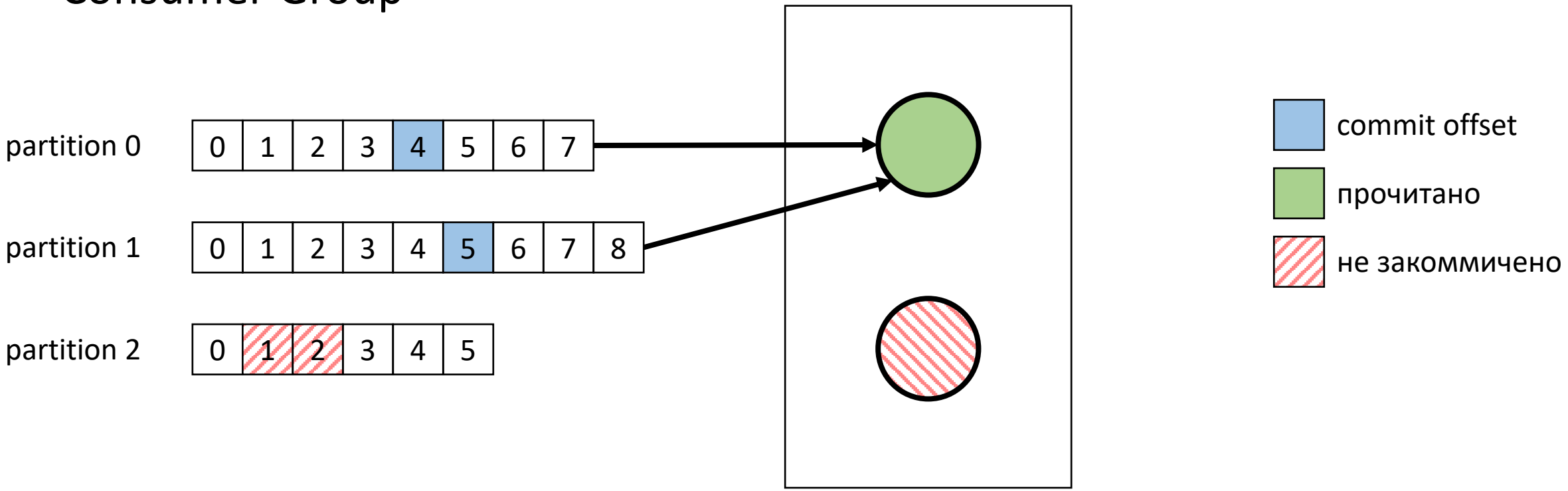
Архитектура Kafka Consumer

Consumer Group



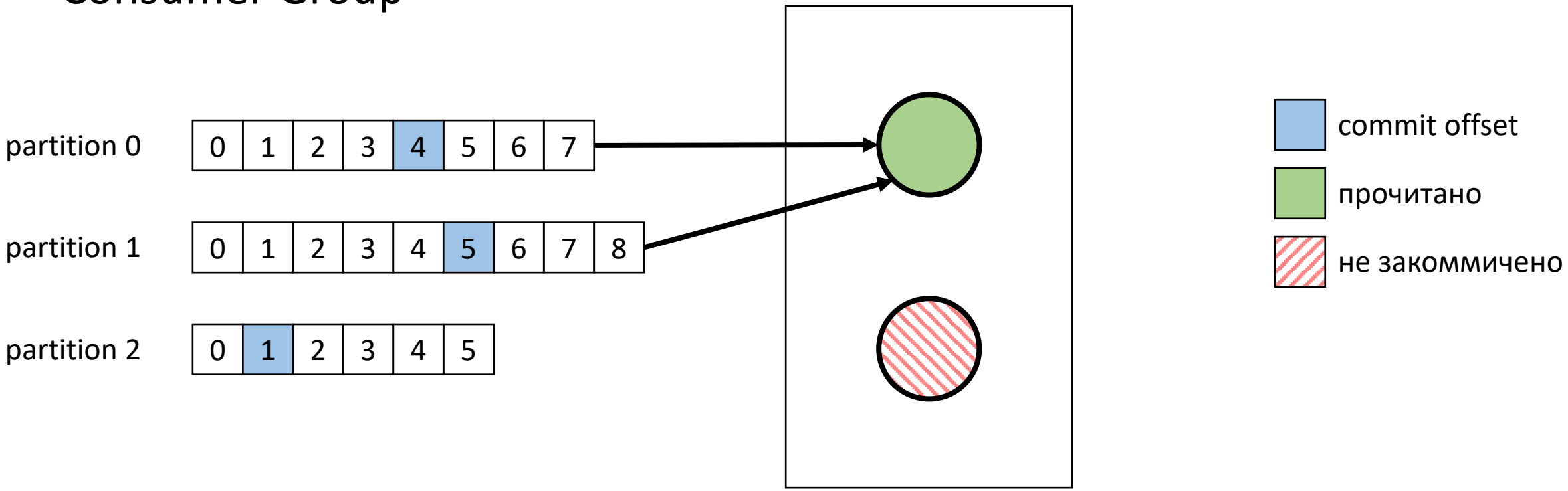
Архитектура Kafka Consumer

Consumer Group



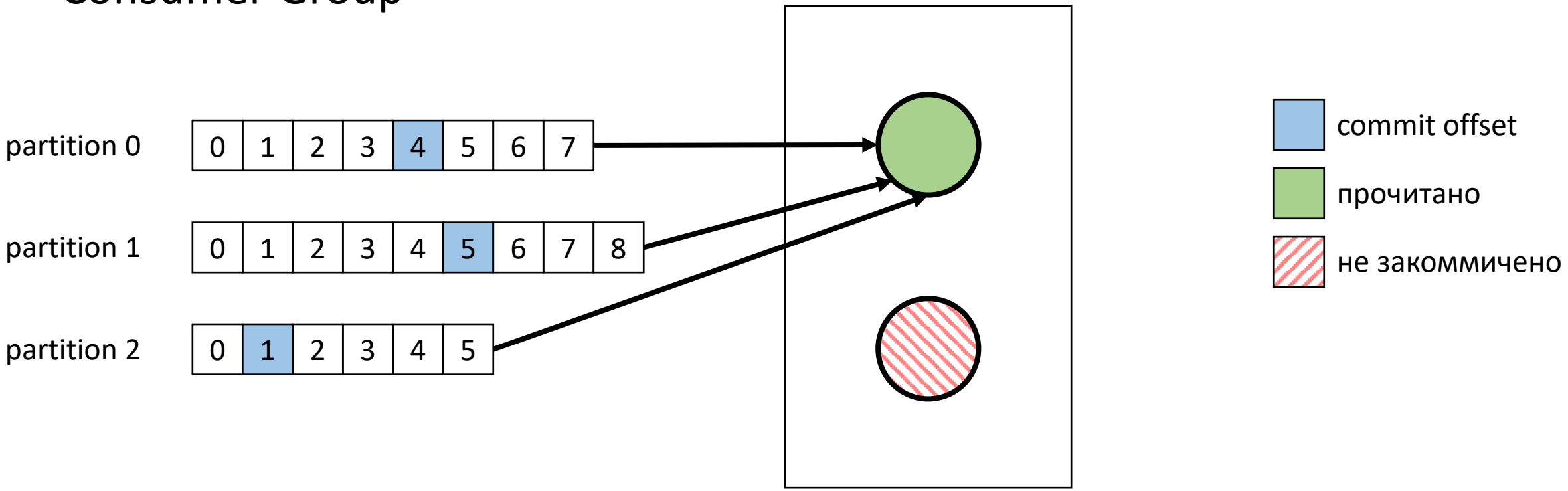
Архитектура Kafka Consumer

Consumer Group



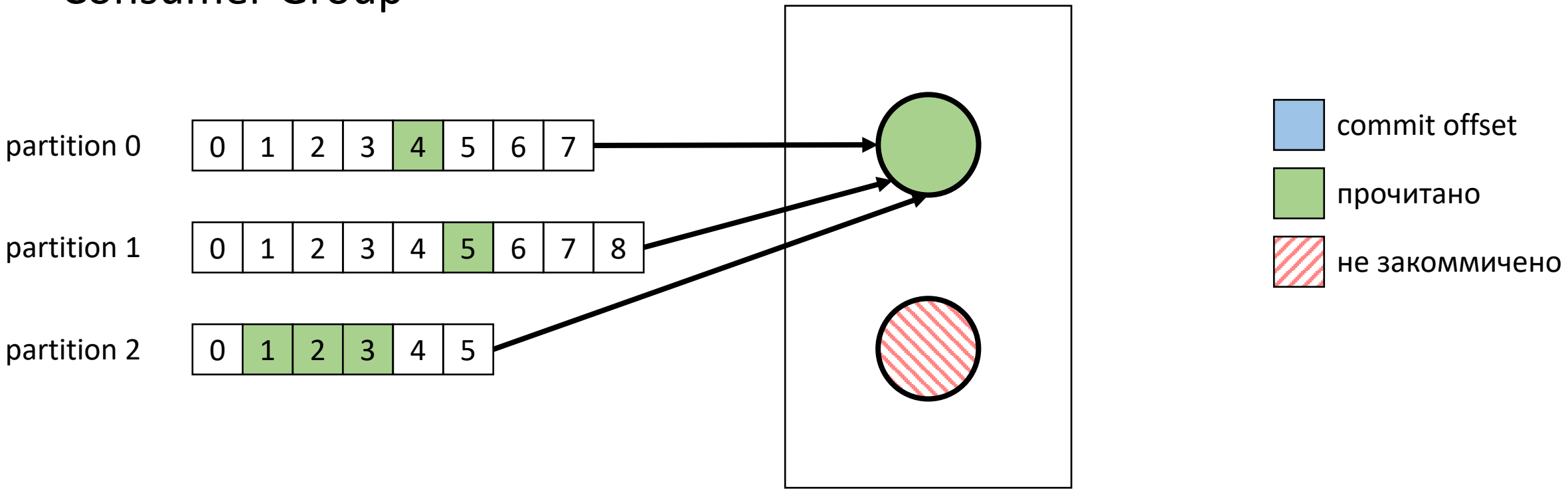
Архитектура Kafka Consumer

Consumer Group



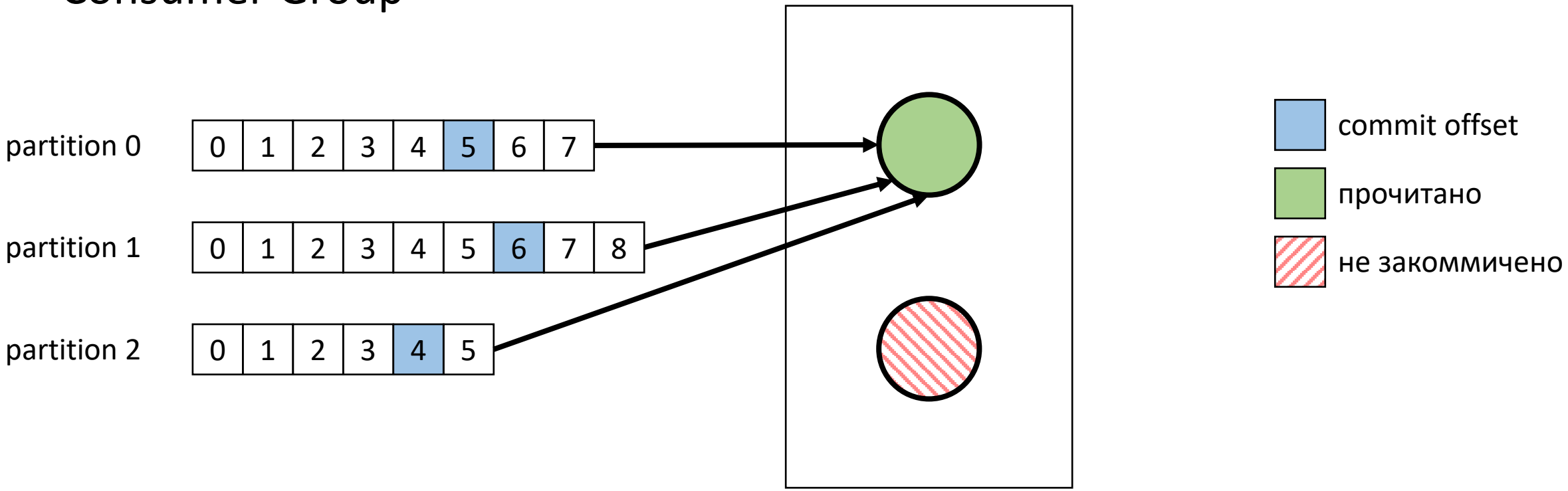
Архитектура Kafka Consumer

Consumer Group



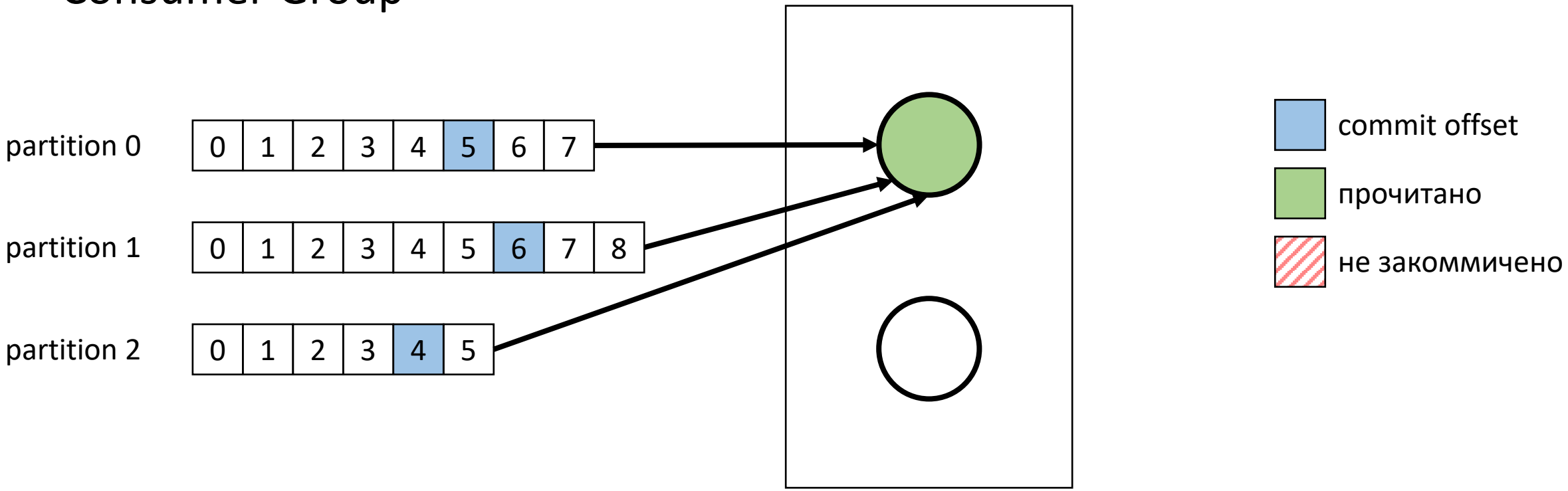
Архитектура Kafka Consumer

Consumer Group



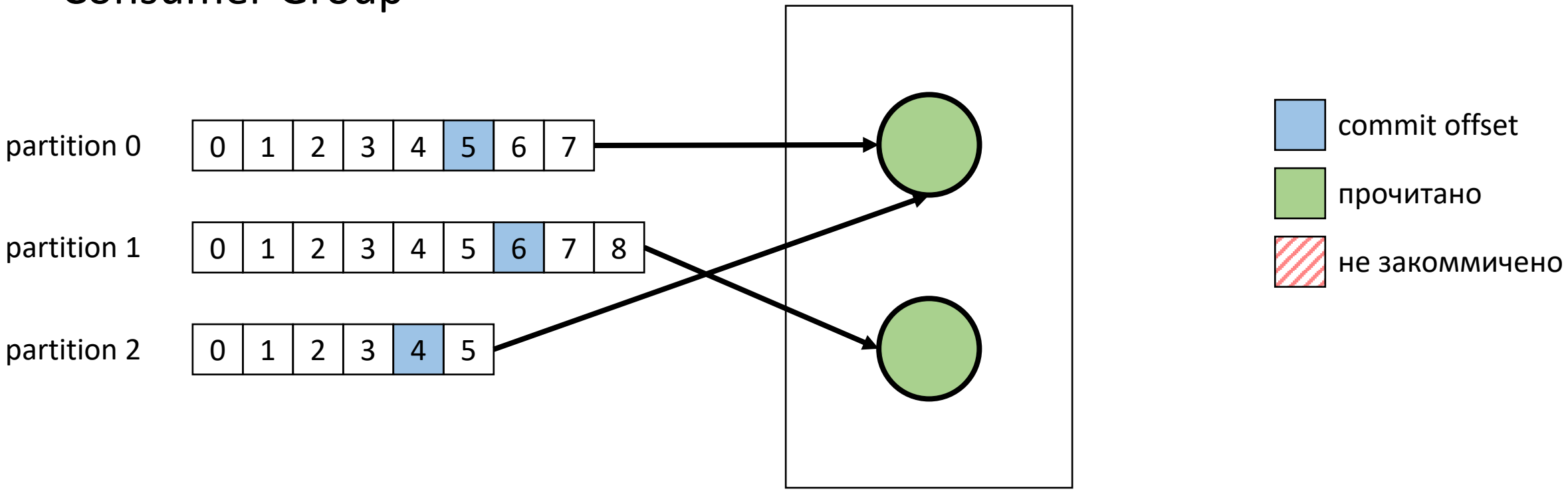
Архитектура Kafka Consumer

Consumer Group



Архитектура Kafka Consumer

Consumer Group



ZooKeeper

ZooKeeper

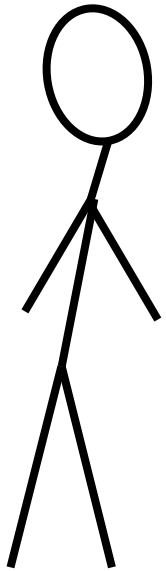
- Controller election

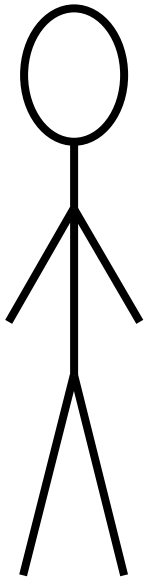
ZooKeeper

- Controller election
- Topic (config, partitions, replicas)

ZooKeeper

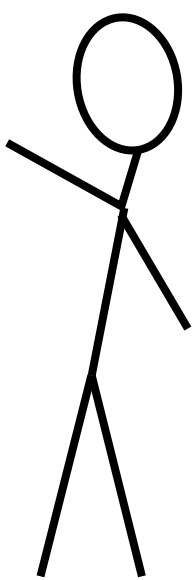
- Controller election
- Topic (config, partitions, replicas)
- Cluster state (online brokers)

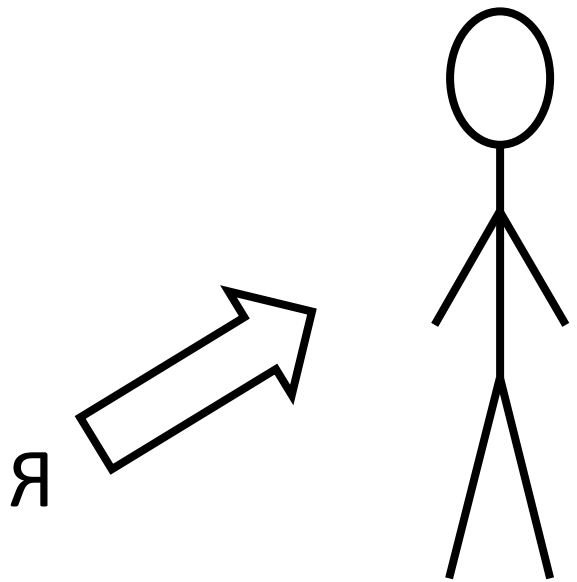




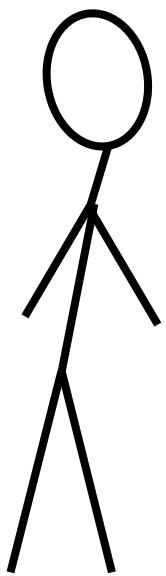

я

МЫ ХОТИМ ВЫБРАТЬ
POSTGRESQL ДЛЯ
НОВОГО ПРОЕКТА…

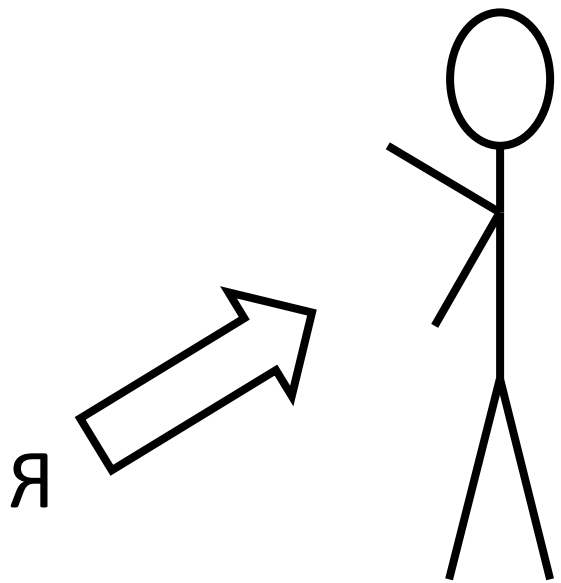




МЫ ХОТИМ ВЫБРАТЬ POSTGRESQL ДЛЯ НОВОГО ПРОЕКТА…

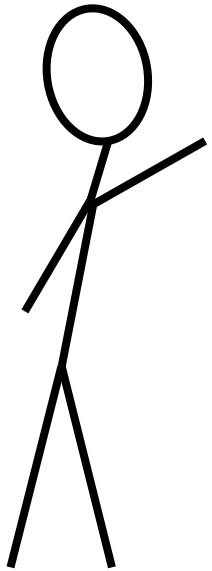


НЕТ!
УЖЕ ЕСТЬ KAFKA!!!

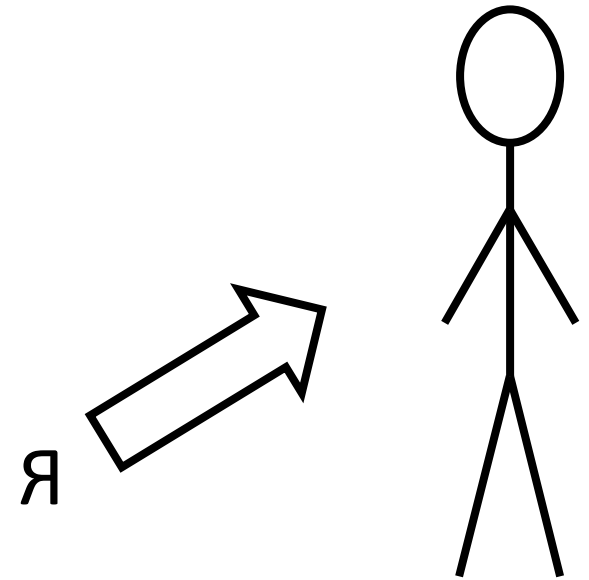


МЫ ХОТИМ ВЫБРАТЬ POSTGRESQL ДЛЯ НОВОГО ПРОЕКТА…

НО У НАС ВСЕГО 100-500 RPS…

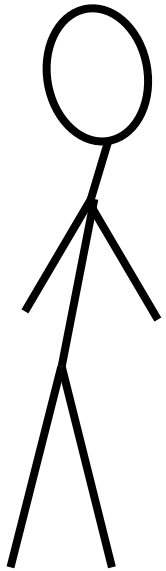


НЕТ!
УЖЕ ЕСТЬ KAFKA!!!



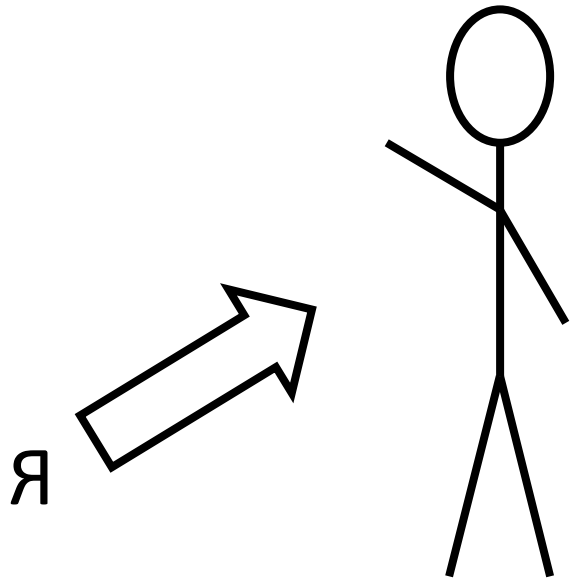
МЫ ХОТИМ ВЫБРАТЬ POSTGRESQL ДЛЯ НОВОГО ПРОЕКТА…

НО У НАС ВСЕГО 100-500 RPS…



НЕТ!
УЖЕ ЕСТЬ KAFKA!!!

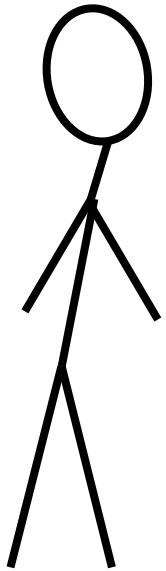
ТОЛЬКО KAFKA!!!
И НЕЧЕГО ДУМАТЬ!!



МЫ ХОТИМ ВЫБРАТЬ POSTGRESQL ДЛЯ НОВОГО ПРОЕКТА…

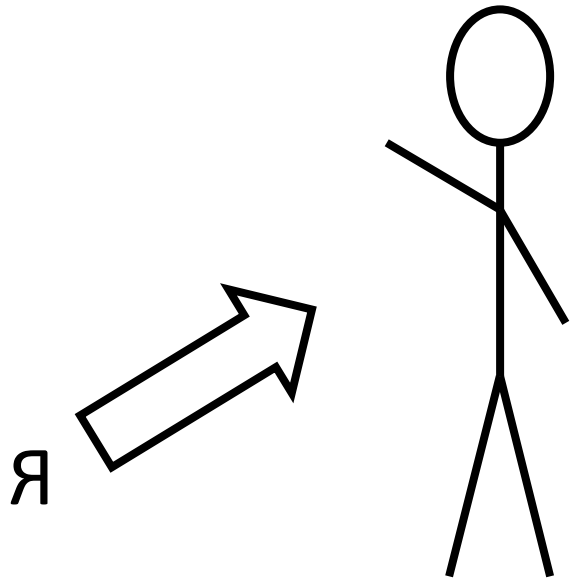
НО У НАС ВСЕГО 100-500 RPS…

… И ДАННЫЕ В ОБЩЕМ-ТО РЕЛЯЦИОННЫЕ…



НЕТ!
УЖЕ ЕСТЬ KAFKA!!!

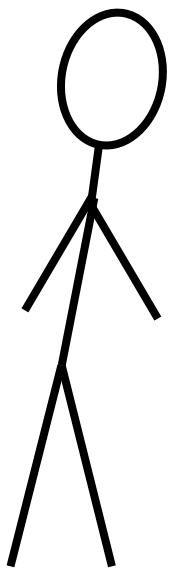
ТОЛЬКО KAFKA!!!
И НЕЧЕГО ДУМАТЬ!!



МЫ ХОТИМ ВЫБРАТЬ POSTGRESQL ДЛЯ НОВОГО ПРОЕКТА…

НО У НАС ВСЕГО 100-500 RPS…

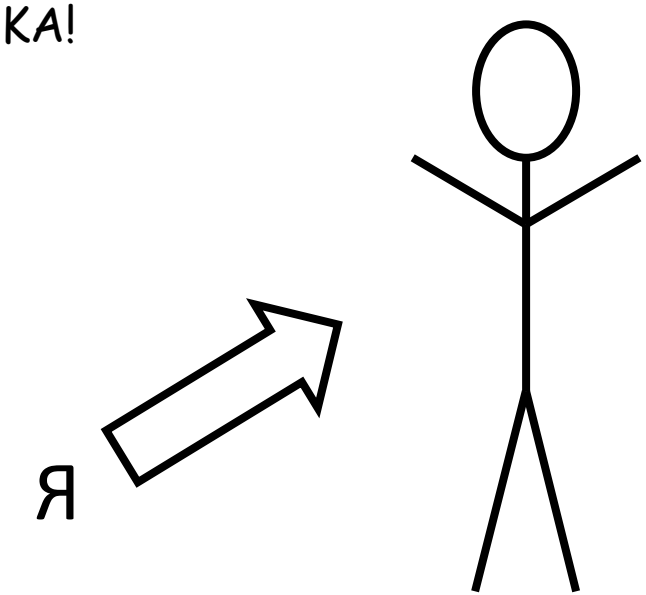
… И ДАННЫЕ В ОБЩЕМ-ТО РЕЛЯЦИОННЫЕ…



НЕТ!
УЖЕ ЕСТЬ KAFKA!!!

ТОЛЬКО KAFKA!!!
И НЕЧЕГО ДУМАТЬ!!

KAFKA! KAFKA!
KAFKA!



Неочевидности в Kafka

Неочевидности в Kafka

… или что мы пережили за год эксплуатации

Неочевидности в Kafka

… или что мы пережили за год эксплуатации

(в кратком изложении)

Неочевидности в Kafka

… или что мы пережили за год эксплуатации

(в кратком изложении)

- Внимательное отношение к **настройкам**

Неочевидности в Kafka

… или что мы пережили за год эксплуатации

(в кратком изложении)

- Внимательное отношение к **настройкам**
- Особенности (недоработки?) клиентского **API**

Неочевидности в Kafka

… или что мы пережили за год эксплуатации

(в кратком изложении)

- Внимательное отношение к **настройкам**
- Особенности (недоработки?) клиентского **API**
- Большое количество **рутины**

Неочевидности в Kafka

… или что мы пережили за год эксплуатации
(в кратком изложении)

- Внимательное отношение к **настройкам**
- Особенности (недоработки?) клиентского **API**
- Большое количество **рутины**

- Документация о многом умалчивает

Неочевидности в Kafka

… или что мы пережили за год эксплуатации

(в кратком изложении)

- Внимательное отношение к **настройкам**
- Особенности (недоработки?) клиентского **API**
- Большое количество **рутины**

- Документация о многом умалчивает (In KIP We Trust)

Как потерять данные

Как потерять данные

- ZK split brain

Как потерять данные

- ZK split brain

2 ZK master

Как потерять данные

- ZK split brain

2 ZK master => 2 Kafka Controller

Как потерять данные

- ZK split brain

2 ZK master => 2 Kafka Controller => 2 leader per partition

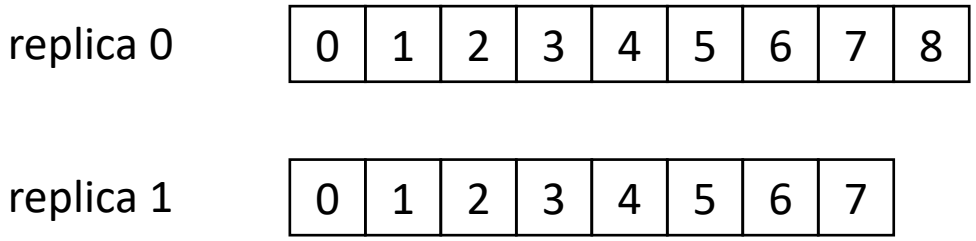
Настройки – Как разломать кластер

Настройки – Как разломать кластер

log.dirs

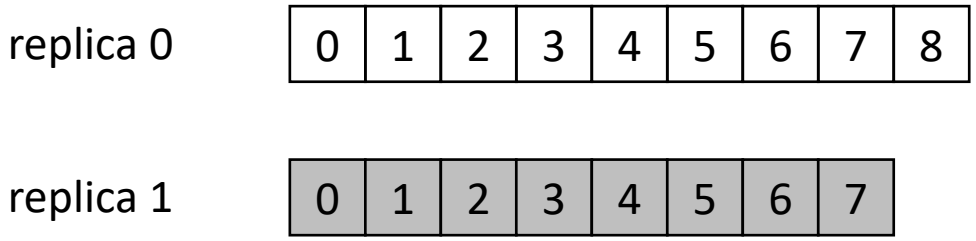
Настройки – Как разломать кластер

log.dirs



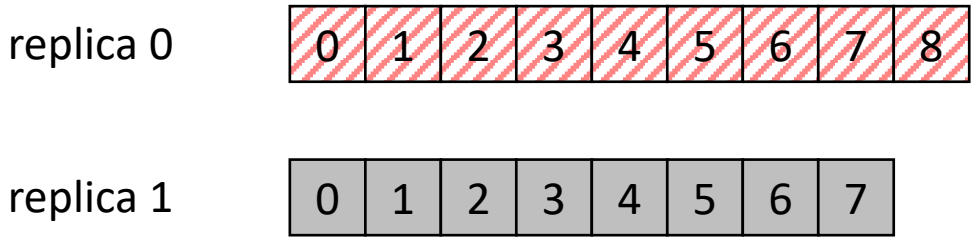
Настройки – Как разломать кластер

log.dirs



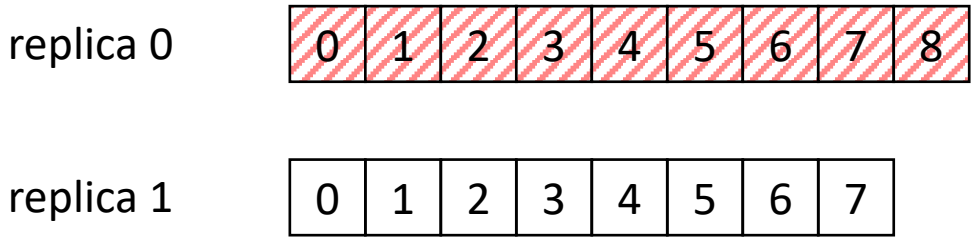
Настройки – Как разломать кластер

log.dirs



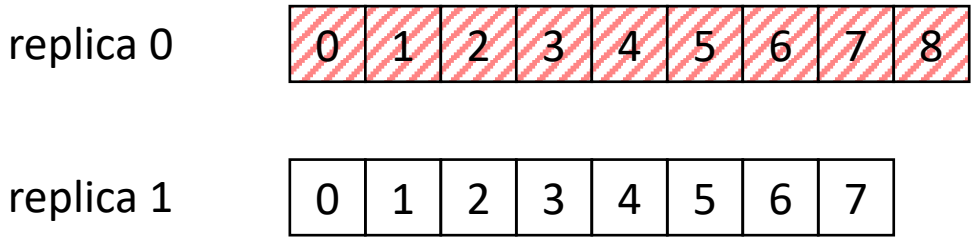
Настройки – Как разломать кластер

log.dirs



Настройки – Как разломать кластер

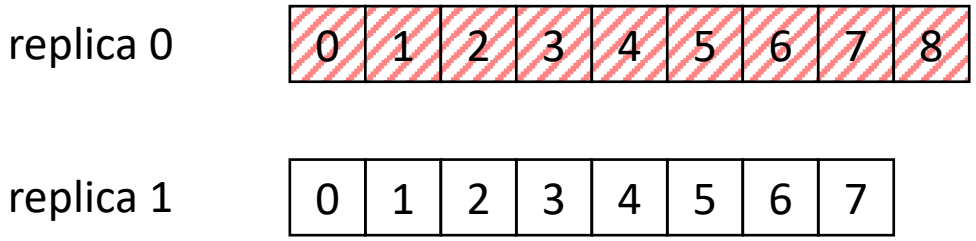
log.dirs



unclean.leader.election.enable=false

Настройки – Как разломать кластер

log.dirs

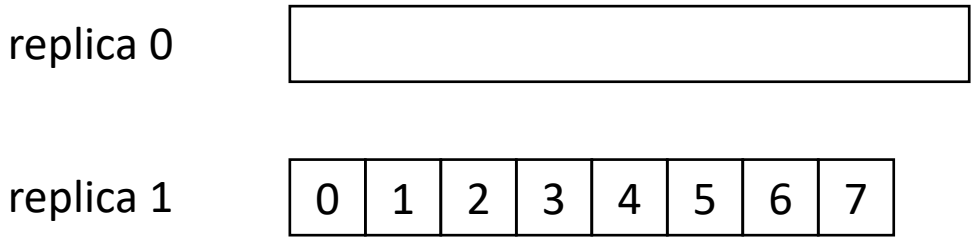


unclean.leader.election.enable=false

KIP-106 - Change Default unclean.leader.election.enabled from True to False (0.11)

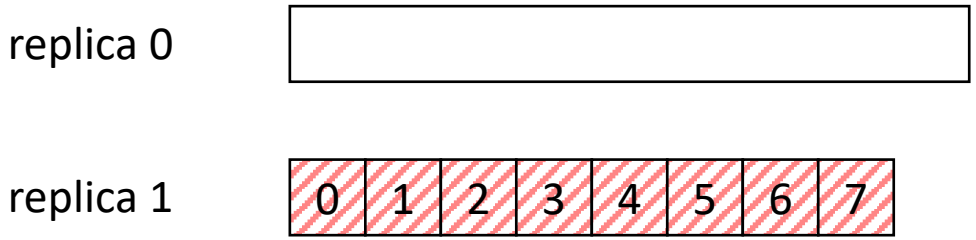
Настройки – Как разломать кластер

log.dirs



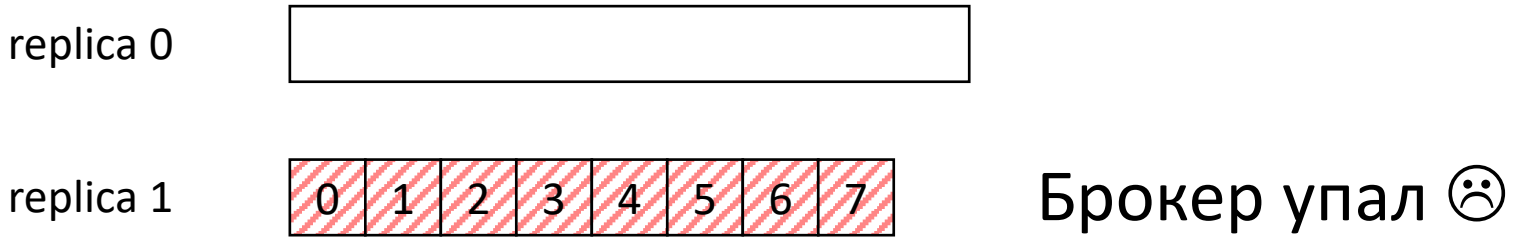
Настройки – Как разломать кластер

log.dirs



Настройки – Как разломать кластер

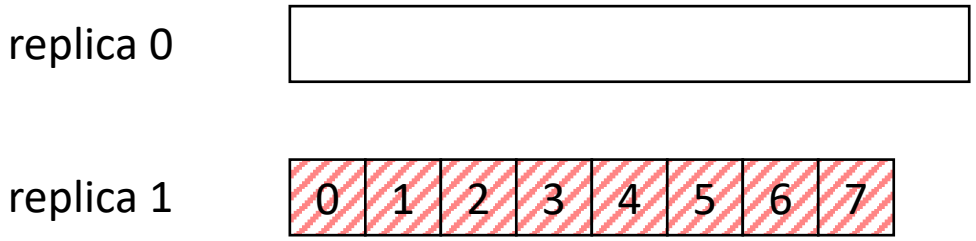
log.dirs



Брокер упал ☹

Настройки – Как разломать кластер

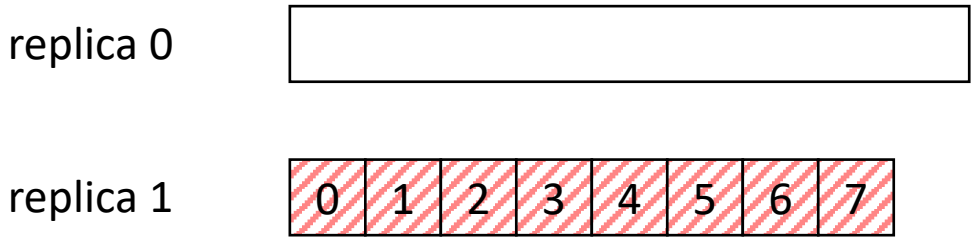
log.dirs



https://issues.apache.org/jira/browse/KAFKA-3410

Настройки – Как разломать кластер

log.dirs



(исправлено* в 1.1)

https://issues.apache.org/jira/browse/KAFKA-3410

Настройки – Как разломать кластер

log.dirs

(исправлено* в 1.1)

https://issues.apache.org/jira/browse/KAFKA-3410

Настройки – Настройки по умолчанию (1)

Настройки – Настройки по умолчанию (1)

- default.replication.factor = 1

Настройки – Настройки по умолчанию (1)

- default.replication.factor = 1
- auto.create.topics.enable = true

Настройки – Настройки по умолчанию (2)

Настройки – Настройки по умолчанию (2)

- Настройки Broker, Consumer и Producer должны быть

Настройки – Настройки по умолчанию (2)

- Настройки Broker, Consumer и Producer должны быть согласованы

Настройки – Настройки по умолчанию (2)

- Настройки Broker, Consumer и Producer должны быть согласованы

- message.max.bytes

Настройки – Настройки по умолчанию (2)

- Настройки Broker, Consumer и Producer должны быть согласованы

- message.max.bytes (Broker, 1_000_012)

Настройки – Настройки по умолчанию (2)

- Настройки Broker, Consumer и Producer должны быть согласованы

- message.max.bytes (Broker, 1_000_012)
- max.request.size

Настройки – Настройки по умолчанию (2)

- Настройки Broker, Consumer и Producer должны быть согласованы

- message.max.bytes (Broker, 1_000_012)
- max.request.size (Producer, 1_048_576)

Настройки – Настройки по умолчанию (2)

- Настройки Broker, Consumer и Producer должны быть согласованы

- message.max.bytes (Broker, 1_000_012)
- max.request.size (Producer, 1_048_576)
- max.partition.fetch.bytes (Consumer, 1_048_576)

Настройки – Дели / Умножай

- message.max.bytes (Broker, 1_000_012)
- max.request.size (Producer, 1_048_576)
- max.partition.fetch.bytes (Consumer, 1_048_576)

Настройки – Дели / Умножай

- message.max.bytes (Broker, 1_000_012)
- max.request.size (Producer, 1_048_576)
- max.partition.fetch.bytes (Consumer, 1_048_576)

- batch.size (Producer, 16_384)

Настройки – Дели / Умножай

- message.max.bytes (Broker, 1_000_012)
- max.request.size (Producer, 1_048_576)
- max.partition.fetch.bytes (Consumer, 1_048_576)

- batch.size (Producer, 16_384)

- KIP-126 - Allow KafkaProducer to split and resend oversized batches (0.11)

API – Блокирующий send

API – Блокирующий send

- Если мета-данные не доступны – producer.send() блокируется

API – Блокирующий send

- Если мета-данные не доступны – producer.send() блокируется
- max.block.ms = 60_000

API – Блокирующий send

- Если мета-данные не доступны – producer.send() блокируется
- max.block.ms = 60_000

- KIP-286: producer.send() should not block on metadata update (discuss)

API – Бесконечная десериализация

API – Бесконечная десериализация

```
while(true) {
    ConsumerRecords<Key, Event> records =
            consumer.poll(1_000);
    for (var record : records) {
        /* do something */
    }
}
```

API – Бесконечная десериализация

```
try {
    /* parsing */
} catch (RuntimeException e) {
    throw new SerializationException(
"Error deserializing key/value for partition " + partition +
" at offset " + record.offset() +
". If needed, please seek past the record to continue consumption.", e);
}
```

API – Бесконечная десериализация

```
try {
    /* parsing */
} catch (RuntimeException e) {
    throw new SerializationException(
"Error deserializing key/value for partition " + partition +
" at offset " + record.offset() +
". If needed, please seek past the record to continue consumption.", e);
}
```

```
consumer.seek(partition, offset);
```

API – Бесконечная десериализация

```
try {
    /* parsing */
} catch (RuntimeException e) {
    throw new SerializationException(
"Error deserializing key/value for partition " + partition +
" at offset " + record.offset() +
". If needed, please seek past the record to continue consumption.", e);
}
```


```
consumer.seek(partition, offset);
```

API – Бесконечная десериализация

```
try {
    /* parsing */
} catch (RuntimeException e) {
    throw new SerializationException(
"Error deserializing key/value for partition " + partition +
" at offset " + record.offset() +
". If needed, please seek past the record to continue consumption.", e);
}
```


```
consumer.seek(partition, offset);
```

API – Бесконечная десериализация

```
try {
    /* parsing */
} catch (RuntimeException e) {
    throw new SerializationException(
"Error deserializing key/value for partition " + partition +
" at offset " + record.offset() +
". If needed, please seek past the record to continue consumption.", e);
}
```


```
consumer.seek(partition, offset);
```

API – Бесконечная десериализация

```
try {
    /* parsing */
} catch (RuntimeException e) {
    throw new SerializationException(
"Error deserializing key/value for partition " + partition +
" at offset " + record.offset() +
". If needed, please seek past the record to continue consumption.", e);
}
```


```
consumer.seek(partition, offset);
```

API – Бесконечная десериализация

```
try {
    /* parsing */
} catch (RuntimeException e) {
    throw new SerializationException(
"Error deserializing key/value for partition " + partition +
" at offset " + record.offset() +
". If needed, please seek past the record to continue consumption.", e);
}
```


```
consumer.seek(partition, offset);
```

API – Бесконечная десериализация

```
try {
    /* parsing */
} catch (RuntimeException e) {
    throw new SerializationException(
"Error deserializing key/value for partition " + partition +
" at offset " + record.offset() +
". If needed, please seek past the record to continue consumption.", e);
}
```


```
consumer.seek(partition, offset);
```

API – Бесконечная десериализация

Наш выбор: кастомный десериализатор, который вернёт null
в случае ошибки

API – Бесконечная десериализация

Наш выбор: кастомный десериализатор, который вернёт null
в случае ошибки

```
while(true) {
    ConsumerRecords<Key, Event> records =
            consumer.poll(1_000);
    for (var record : records) {
        if (record.getValue() == null) { continue; }
        /* do something */
    }
}
```

API – Бесконечная десериализация

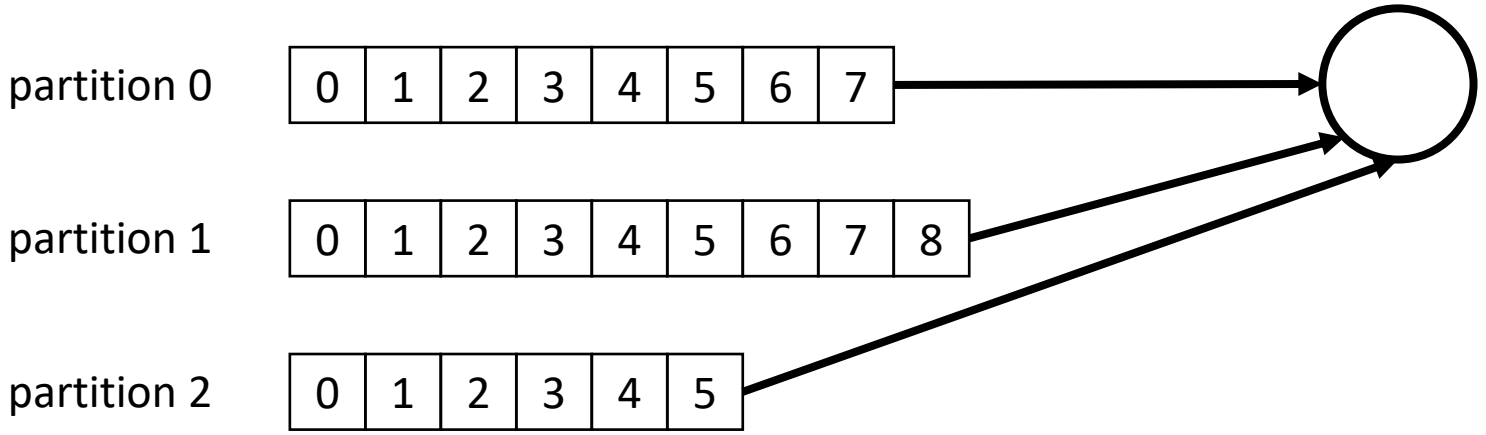
Наш выбор: кастомный десериализатор, который вернёт null в случае ошибки

```
while(true) {
    ConsumerRecords<Key, Event> records =
            consumer.poll(1_000);
    for (var record : records) {
        if (record.getValue() == null) { continue; }
        /* do something */
    }
}
```

API – НЕчестное чтение

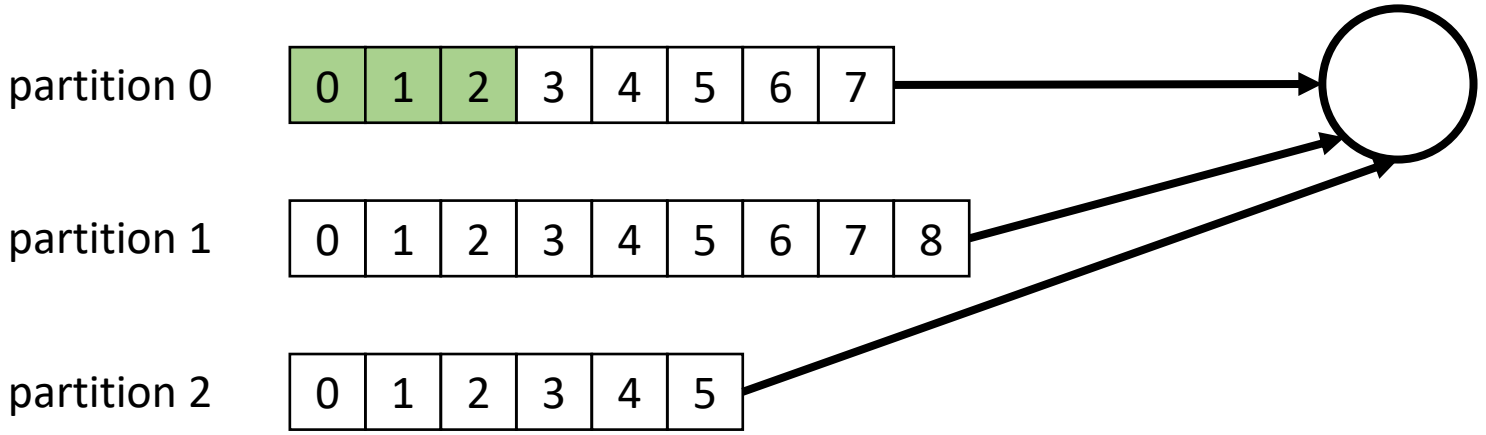
API – НЕчестное чтение

Чтение по 3 сообщения



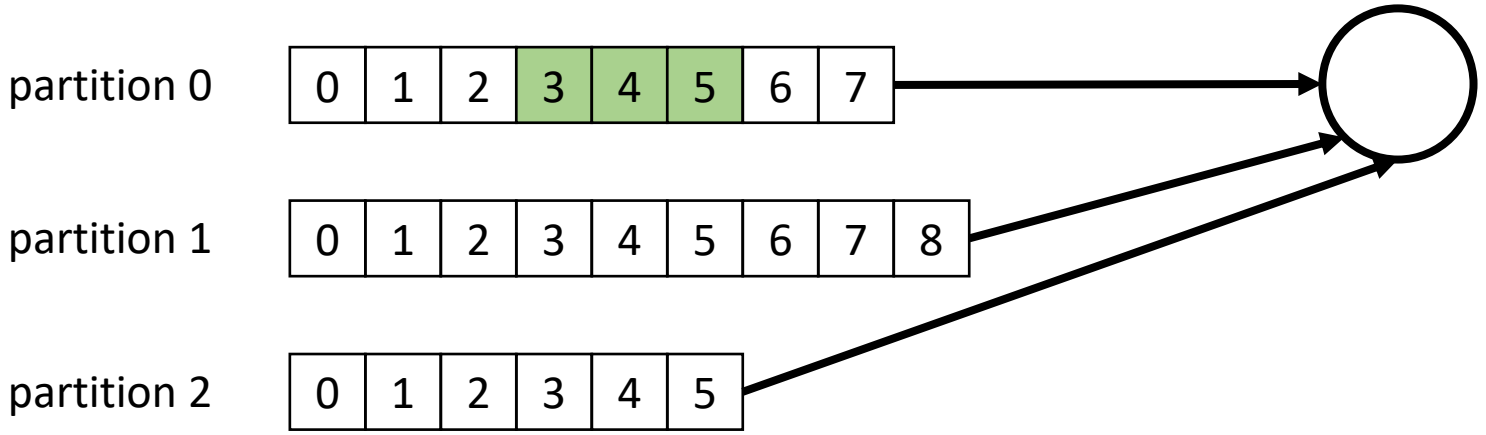
API – НЕчестное чтение

Чтение по 3 сообщения



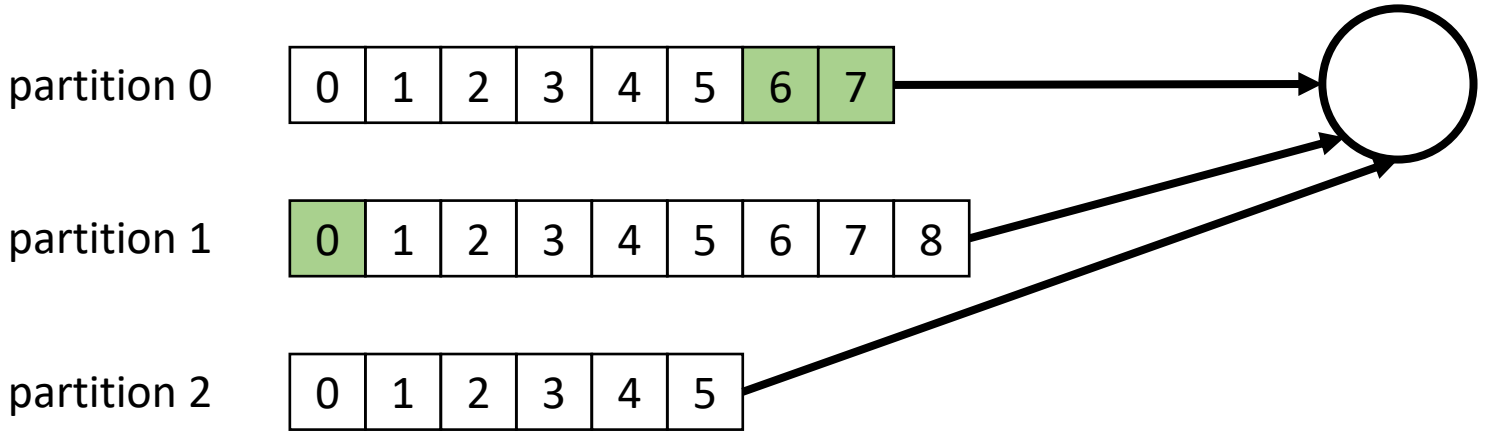
API – НЕчестное чтение

Чтение по 3 сообщения



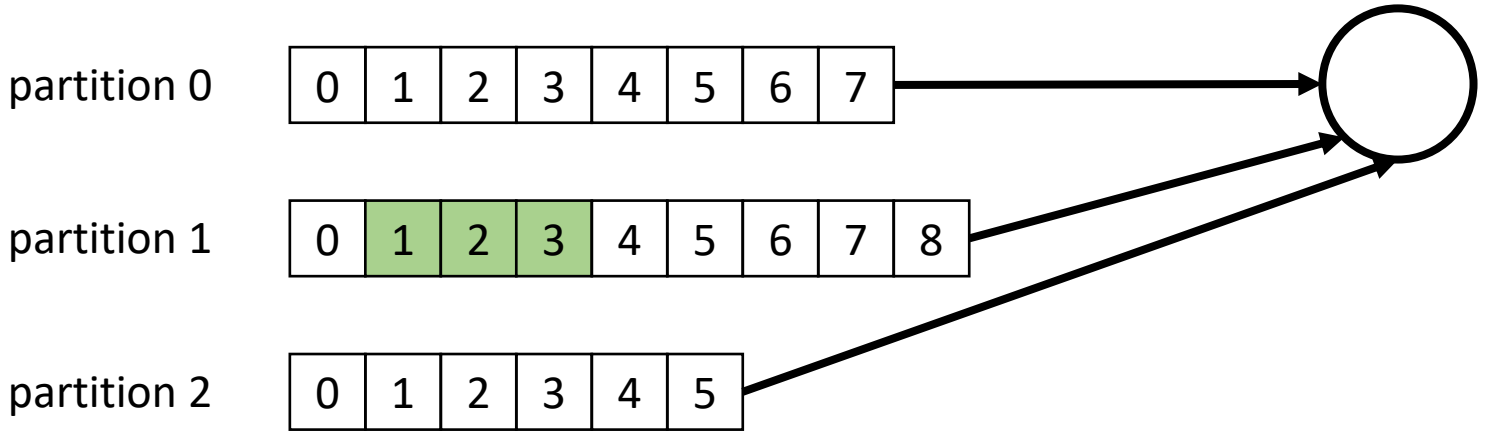
API – НЕчестное чтение

Чтение по 3 сообщения



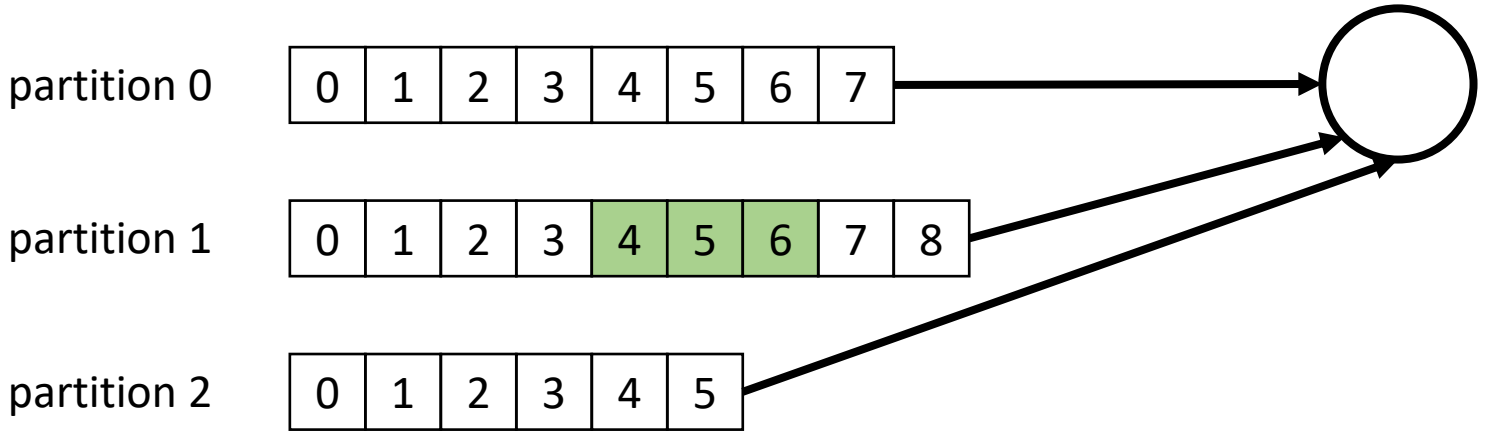
API – НЕчестное чтение

Чтение по 3 сообщения



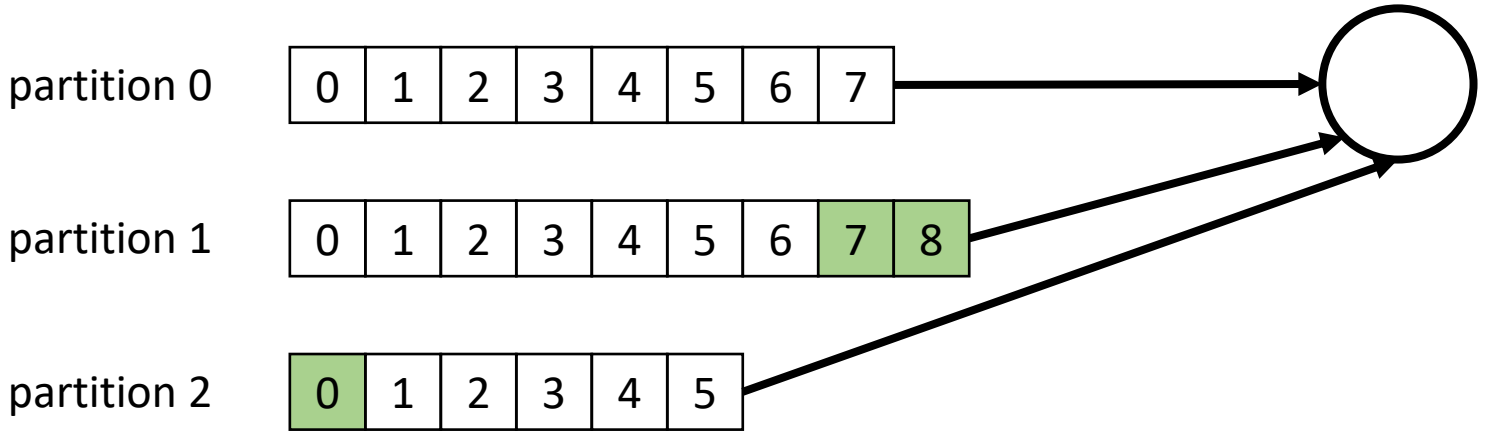
API – НЕчестное чтение

Чтение по 3 сообщения



API – НЕчестное чтение

Чтение по 3 сообщения



API – НЕчестное чтение

- KIP-41: KafkaConsumer Max Records (0.10)

API – НЕчестное чтение

- KIP-41: KafkaConsumer Max Records (0.10)
- Жадный round-robin

API – НЕчестное чтение

- KIP-41: KafkaConsumer Max Records (0.10)
- Жадный round-robin

- KIP-387: Fair Message Consumption Across Partitions in KafkaConsumer (discuss)

Рутина – Место на диске

Рутина – Место на диске

- Нет автораспределения партиций по новым дискам

Рутина – Место на диске

- Нет автораспределения партиций по новым дискам
- KIP-113: Support replicas movement between log directories (1.1)

Рутина – Место на диске

- Нет авторaспределения партиций по новым дискам
- KIP-113: Support replicas movement between log directories (1.1)

- Равномерное распределение партиций **по количеству**

Рутина – Место на диске

- Нет авторaспределения партиций по новым дискам
- KIP-113: Support replicas movement between log directories (1.1)

- Равномерное распределение партиций **по количеству**
- KIP-178: Size-based log directory selection strategy (discuss)

Рутина – Новый Брокер

Рутина – Новый Брокер

- Нет автораспределения партиций на нового Брокера

Рутина – Новый Брокер

- Нет автораспределения партиций на нового Брокера
- Руками делать partition reassignment

Рутина – Новый Брокер

- Нет автораспределения партиций на нового Брокера
- Руками делать partition reassignment

```
{ "version": 1,
  "partitions": [
    { "topic": "topic", "partition" : partition,
      "replicas": [brokerIds]
    },
    ...
  ]
}
```

Рутина – Новый Брокер

- Нет автораспределения партиций на нового Брокера
- Руками делать partition reassignment

```
{ "version": 1,
  "partitions": [
    { "topic": "topic", "partition" : partition,
      "replicas": [brokerIds]
    },
    ...
  ]
}
```

Рутина – Новый Брокер

- Нет автораспределения партиций на нового Брокера
- Руками делать partition reassignment

```
{ "version": 1,
  "partitions": [
    { "topic": "hg2tg", "partition" : partition,
      "replicas": [brokerIds]
    },
    ...
  ]
}
```

Рутина – Новый Брокер

- Нет автораспределения партиций на нового Брокера
- Руками делать partition reassignment

```
{ "version": 1,
  "partitions": [
    { "topic": "hg2tg", "partition" : 42,
      "replicas": [brokerIds]
    },
    ...
  ]
}
```

Рутина – Новый Брокер

```
{ "version": 1,
  "partitions": [
    { "topic": "hg2tg", "partition" : 42,
      "replicas": [1, 2, 3]
    },
    ...
  ]
}
```

Рутина – Новый Брокер

- Preferred leader – первый брокер в списке реплик

```
{ "version": 1,
  "partitions": [
    { "topic": "hg2tg", "partition" : 42,
      "replicas": [1, 2, 3]
    },
    ...
  ]
}
```

Рутина – Ограничение на размер топика

Рутина – Ограничение на размер топика

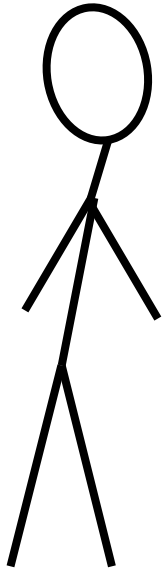
- log.retention.bytes (Broker, unlimited)

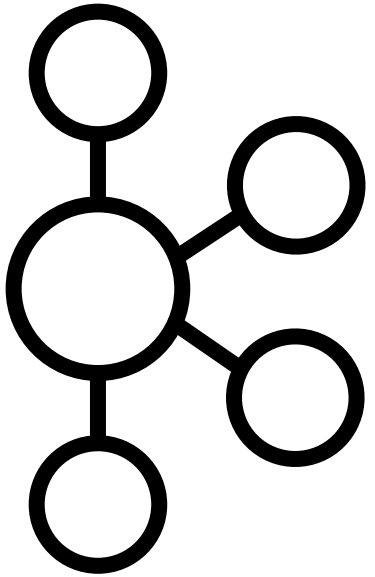
Рутина – Ограничение на размер топика

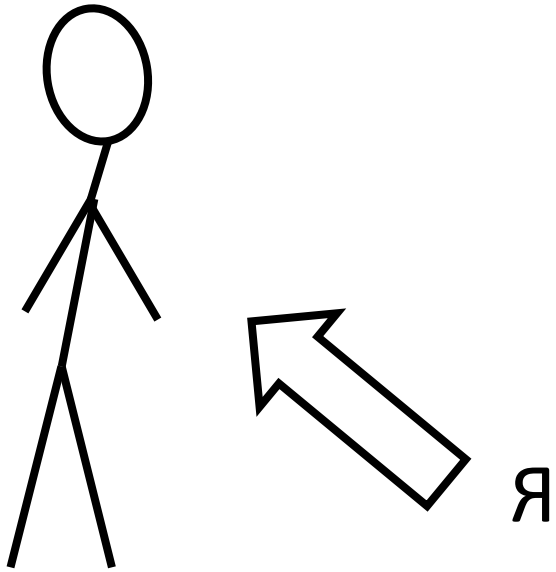
- log.retention.bytes (Broker, unlimited)
- retention.bytes (Topic)

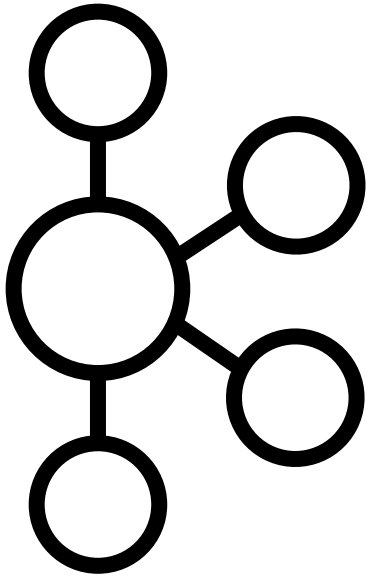
Рутина – Ограничение на размер топика

- log.retention.bytes (Broker, unlimited)
- retention.bytes (Topic) – per partition

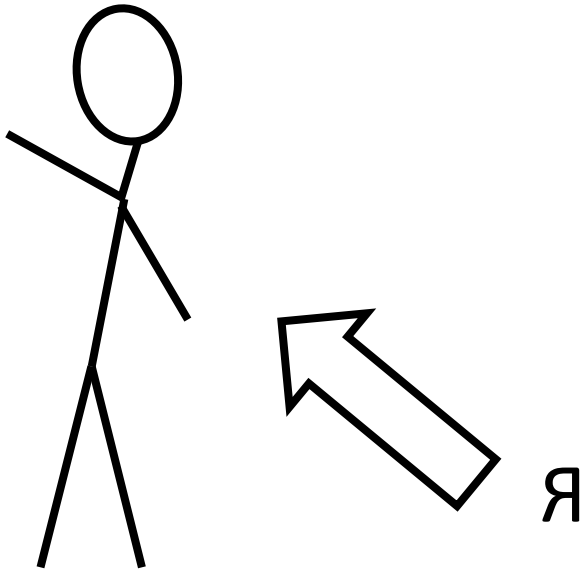


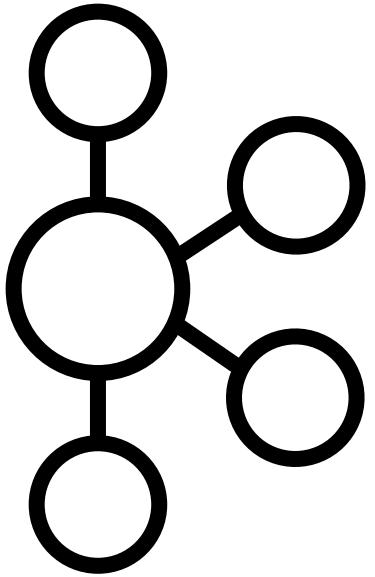



я

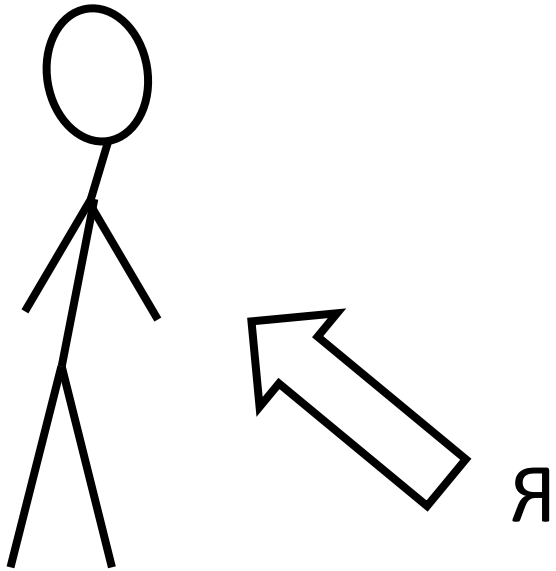


KAFKA, МЫ ДОЛЖНЫ НАЧАТЬ ЭКОНОМИТЬ МЕСТО.

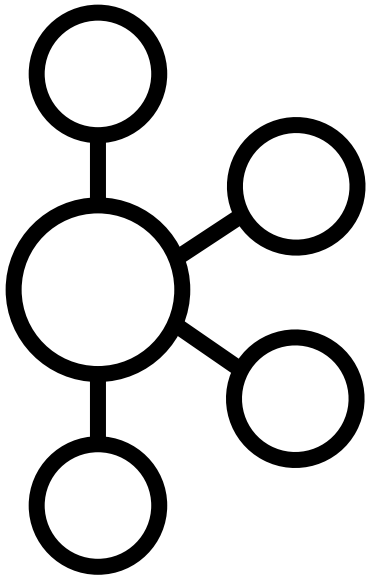




KAFKA, МЫ ДОЛЖНЫ НАЧАТЬ ЭКОНОМИТЬ МЕСТО.

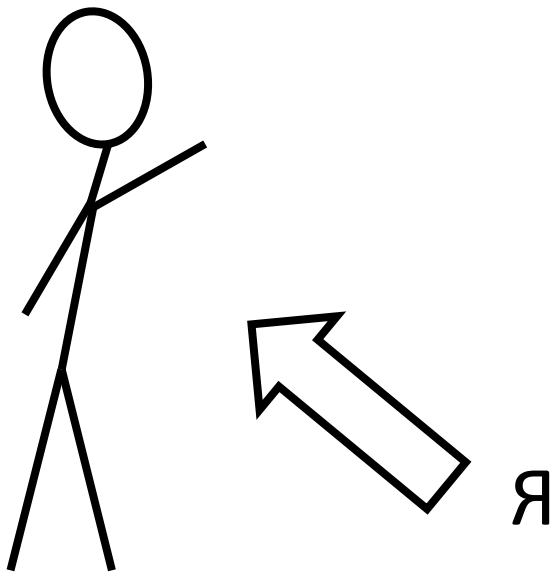


ТЫ ЖАДНЫЙ!
НЕ БУДЬ ТАКИМ!

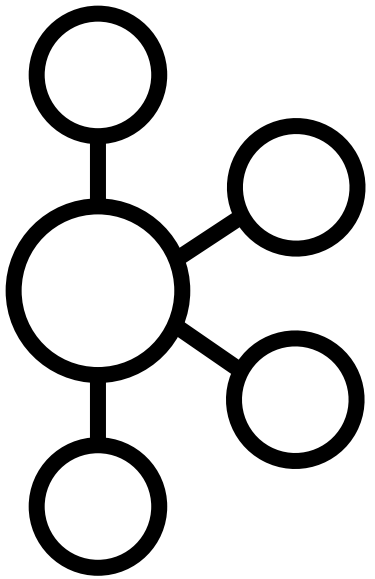


KAFKA, МЫ ДОЛЖНЫ НАЧАТЬ ЭКОНОМИТЬ МЕСТО.

МЫ МОЖЕМ ПОПРОБОВАТЬ ОГРАНИЧИТЬ ТОПИК?..



ТЫ ЖАДНЫЙ!
НЕ БУДЬ ТАКИМ!

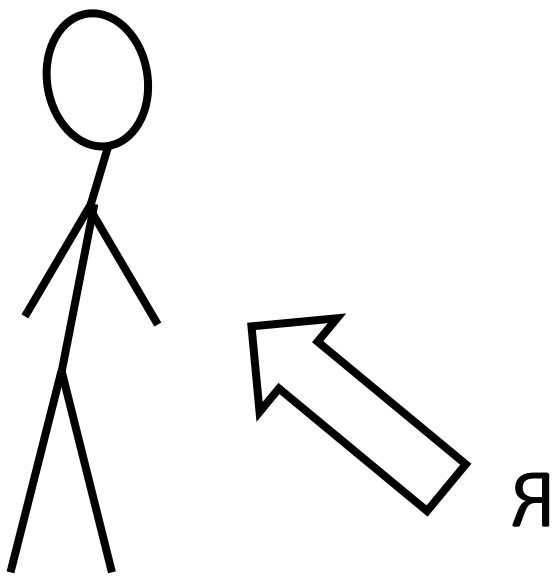


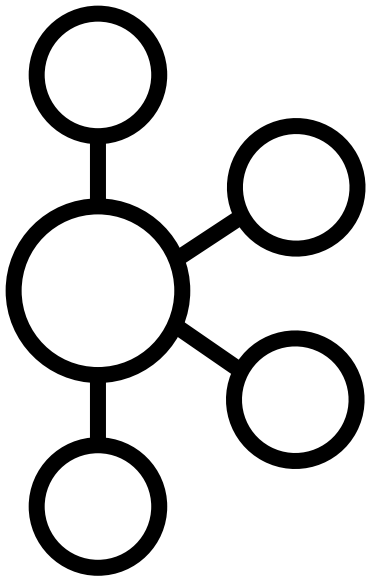
KAFKA, МЫ ДОЛЖНЫ НАЧАТЬ ЭКОНОМИТЬ МЕСТО.

МЫ МОЖЕМ ПОПРОБОВАТЬ ОГРАНИЧИТЬ ТОПИК?..

ТЫ ЖАДНЫЙ!
НЕ БУДЬ ТАКИМ!

ИЗВЕНИ,
НО НЕТ!

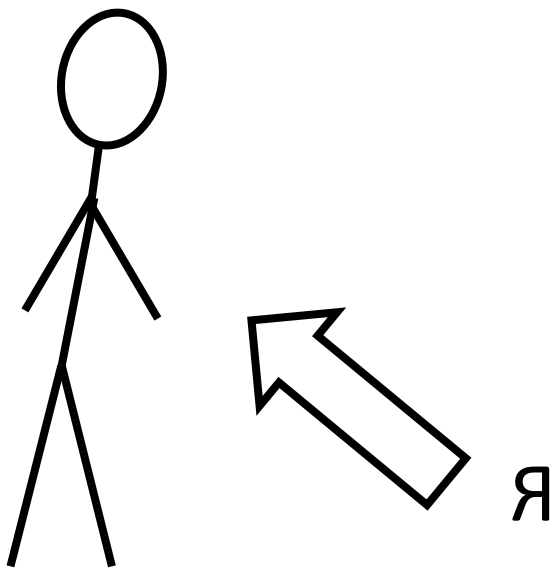




KAFKA, МЫ ДОЛЖНЫ НАЧАТЬ ЭКОНОМИТЬ МЕСТО.

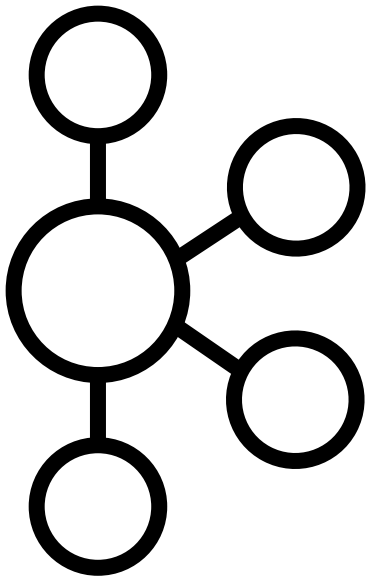
МЫ МОЖЕМ ПОПРОБОВАТЬ ОГРАНИЧИТЬ ТОПИК?..

... Я... Я РАЗОЧАРОВАН!



ТЫ ЖАДНЫЙ!
НЕ БУДЬ ТАКИМ!

ИЗВЕНИ,
НО НЕТ!



KAFKA, МЫ ДОЛЖНЫ НАЧАТЬ ЭКОНОМИТЬ МЕСТО.

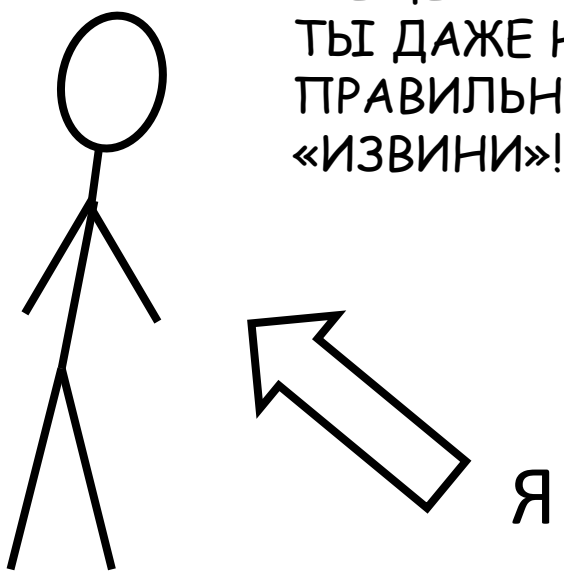
МЫ МОЖЕМ ПОПРОБОВАТЬ ОГРАНИЧИТЬ ТОПИК?..

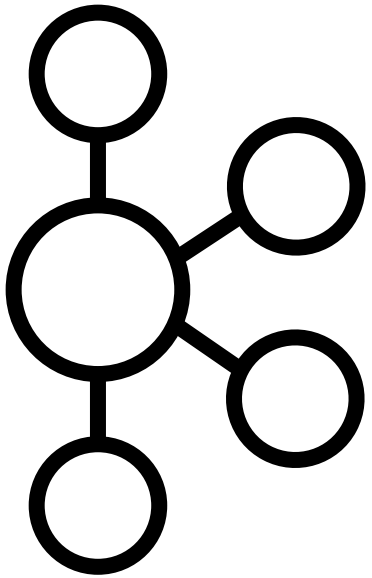
ТЫ ЖАДНЫЙ!
НЕ БУДЬ ТАКИМ!

ИЗВЕНИ,
НО НЕТ!

… Я… Я РАЗОЧАРОВАН!

И ЕЩЁ 100500 ЧЕЛОВЕК:
ТЫ ДАЖЕ НЕ СМОГЛА ПРАВИЛЬНО НАПИСАТЬ «ИЗВИНИ»! ☹





Архитектурные байки

- Zero-copy
- Time is index
- Consumer Offset

Zero-copy – magic

Zero-copy – magic

- Единый формат для хранения и передачи

Zero-copy – magic

- Единый формат для хранения и передачи
- Чтение и передача непрерывного куска лога

Zero-copy – magic

- Единый формат для хранения и передачи
- Чтение и передача непрерывного куска лога

- No 0-copy: disk -> pagecache -> app -> socket buff ->NIC buff

Zero-copy – magic

- Единый формат для хранения и передачи
- Чтение и передача непрерывного куска лога

- No 0-copy: disk -> pagecache -> app -> socket buff ->NIC buff
- 0-copy: disk -> pagecache -> socket buff -> NIC buff

Zero-copy – magic

- Единый формат для хранения и передачи
- Чтение и передача непрерывного куска лога

- No 0-copy: disk -> pagecache -> app -> socket buff ->NIC buff
- 0-copy: disk -> pagecache -> socket buff -> NIC buff

- https://developer.ibm.com/articles/j-zerocopy/

Zero-copy – Compression

Zero-copy – Compression

- Producer Compression

Zero-copy – Compression

- Producer Compression
- Consumer Decompression

Zero-copy – Compression

- Producer Compression
- Consumer Decompression

- Consumer получит пачку целиком (zero-copy!)

Zero-copy – Compression

- Producer Compression
- Consumer Decompression

- Consumer получит пачку целиком (zero-copy!)
- В Kafka сжатая пачка хранится в виде Wrapper message

Zero-copy – Быстрый Consumer

Zero-copy – Быстрый Consumer

- Поиск **в** Kafka

Zero-copy – Быстрый Consumer

- Поиск **поверх** Kafka

Zero-copy – Быстрый Consumer

- Поиск **поверх** Kafka

- Быстрый Consumer выжирает всю сеть

Zero-copy – Быстрый Consumer

- Поиск **поверх** Kafka

- Быстрый Consumer выжирает всю сеть
- Quotas

Zero-copy – Быстрый Consumer

- Поиск **поверх** Kafka

- Быстрый Consumer выжирает всю сеть
- Quotas

- KIP-13: Quotas

Zero-copy – Различные версии

Zero-copy – Различные версии

- https://kafka.apache.org/documentation/#upgrade

Zero-copy – Различные версии

- https://kafka.apache.org/documentation/#upgrade
- Down-conversion на стороне Broker для старых Consumer

Zero-copy – Различные версии

- https://kafka.apache.org/documentation/#upgrade
- Down-conversion на стороне Broker для старых Consumer

- KIP-283: Efficient Memory Usage for Down-Conversion (2.0)

Time is index – Time index

Time is index – Time index

- Если события слабо упорядочены по timestamp (большие лаги), то поиск по timestamp в общем-то бесполезен

Time is index – Time index

- Если события слабо упорядочены по timestamp (большие лаги), то поиск по timestamp в общем-то бесполезен
- KIP-33 - Add a time based log index (0.10)

Time is index – Time index

- Если события слабо упорядочены по timestamp (большие лаги), то поиск по timestamp в общем-то бесполезен
- KIP-33 - Add a time based log index (0.10)

timeindex record = (relative_offset_next, max_timestamp)

Time is index – Time index

- Если события слабо упорядочены по timestamp (большие лаги), то поиск по timestamp в общем-то бесполезен
- KIP-33 - Add a time based log index (0.10)

timeindex record = (relative_offset_next, max_timestamp)

relative_offset_next – относительный оффсет следующего события

Time is index – Time index

- Если события слабо упорядочены по timestamp (большие лаги), то поиск по timestamp в общем-то бесполезен
- KIP-33 - Add a time based log index (0.10)

timeindex record = (relative_offset_next, max_timestamp)

relative_offset_next – относительный оффсет следующего события

max_timestamp – максимальный таймстемп с начала сегмента

Time is index – Timestamp и Retention

Time is index – Timestamp и Retention

- Timestamp записи (Producer или Broker)

Time is index – Timestamp и Retention

- Timestamp записи (Producer или Broker)
- Retention (по времени) – удаление протухших сегментов

Time is index – Timestamp и Retention

- Timestamp записи (Producer или Broker)
- Retention (по времени) – удаление протухших сегментов
- «Свежесть» сегмента определяется по max(timestamp) среди всех событий в сегменте, а не моменту последней записи

Time is index – Timestamp и Retention

- Timestamp записи (Producer или Broker)
- Retention (по времени) – удаление протухших сегментов
- «Свежесть» сегмента определяется по max(timestamp) среди всех событий в сегменте, а не моменту последней записи

- Одна запись из будущего блокирует удаление сегмента

Consumer Offset – Puzzle

Consumer Offset – Puzzle

1 Producer, 1 Consumer, 1 Partition

Consumer Offset – Puzzle

1 Producer, 1 Consumer, 1 Partition

Consumer Offset – Puzzle

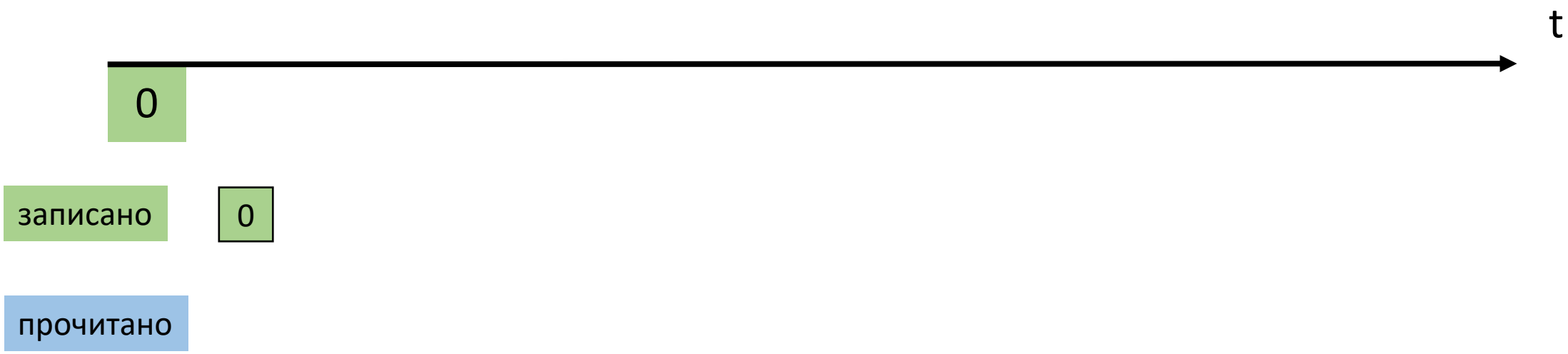
1 Producer, 1 Consumer, 1 Partition

записано

прочитано

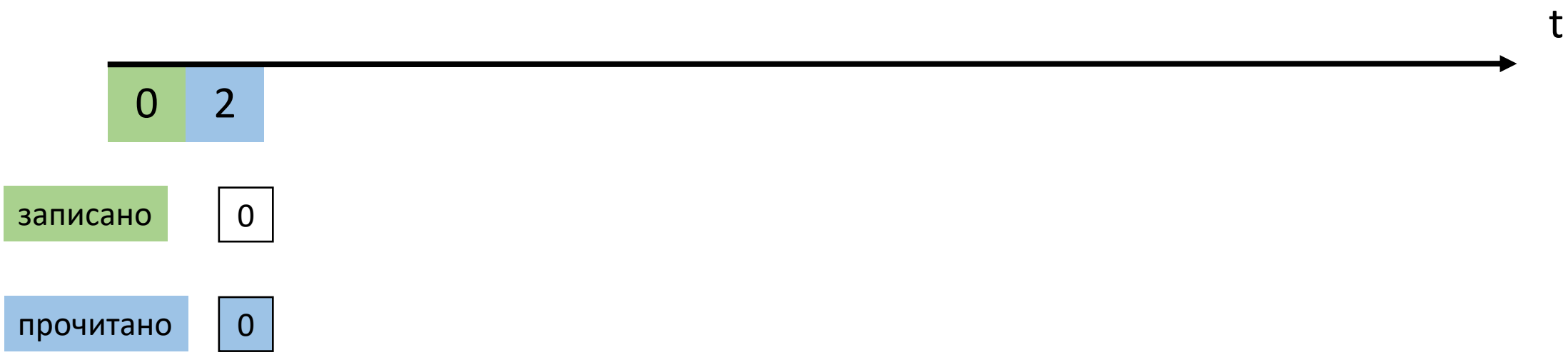
Consumer Offset – Puzzle

Producer *успешно* отправил 1 сообщение в момент ***t = 0 часов***



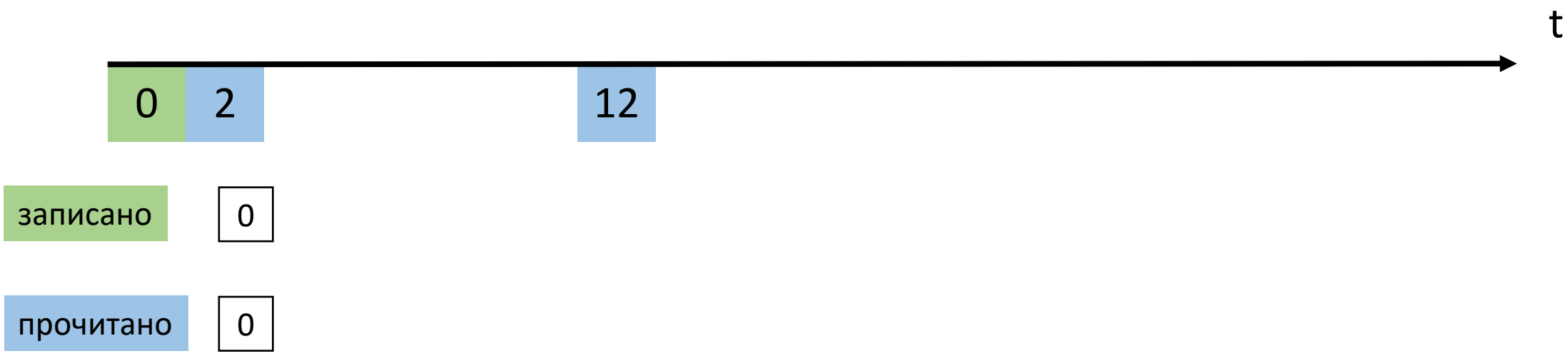
Consumer Offset – Puzzle

Consumer *успешно* прочитал 1 сообщение в момент ***t = 2 часа***



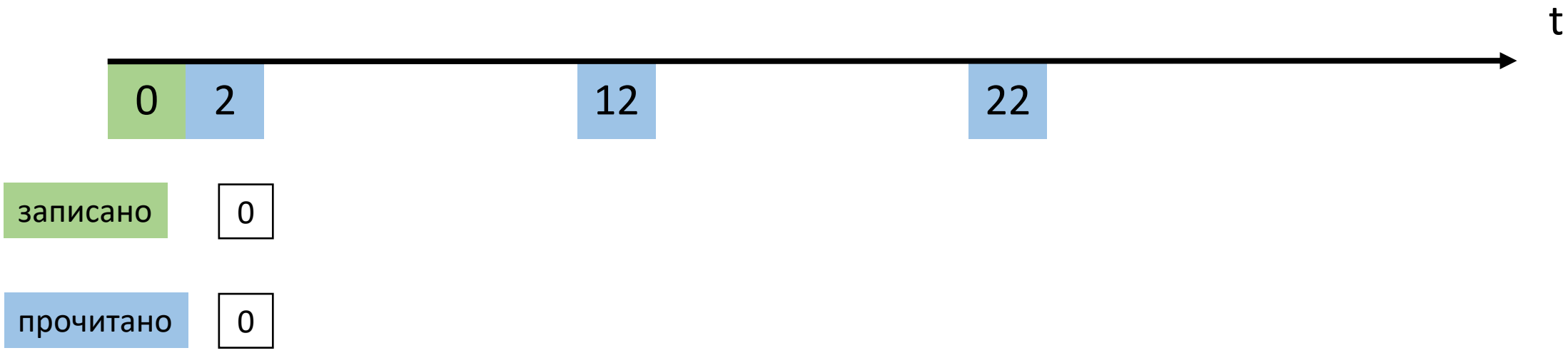
Consumer Offset – Puzzle

Consumer продолжил читать каждые ***10 часов***



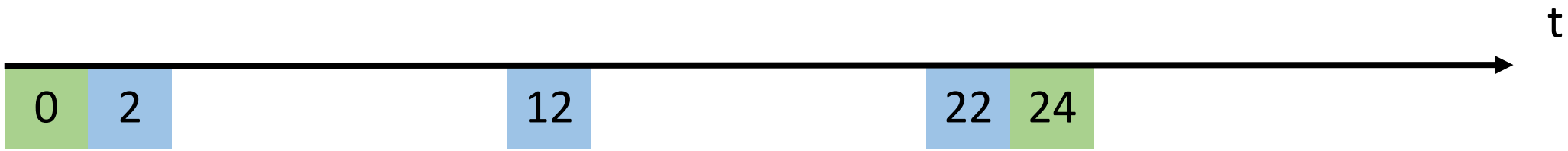
Consumer Offset – Puzzle

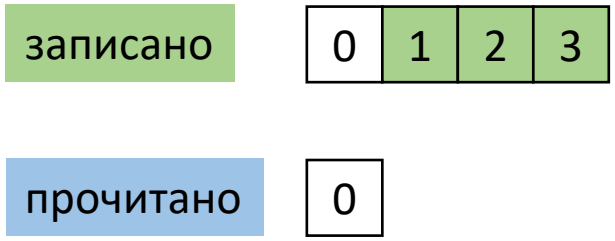
Consumer продолжил читать каждые ***10 часов***



Consumer Offset – Puzzle

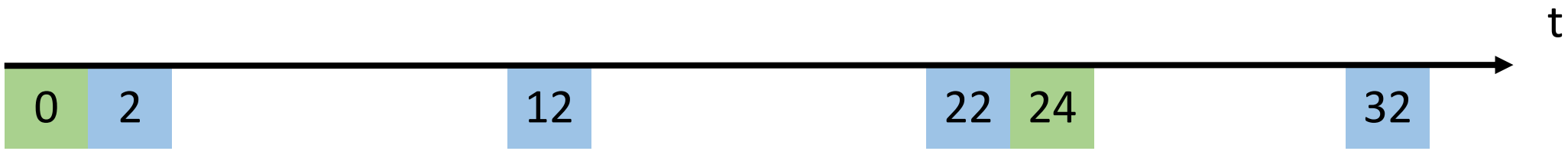
Producer *успешно* записал сразу 3 сообщения в момент ***t = 24 часа***

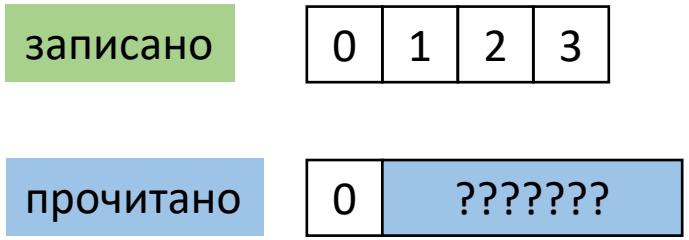




Consumer Offset – Puzzle

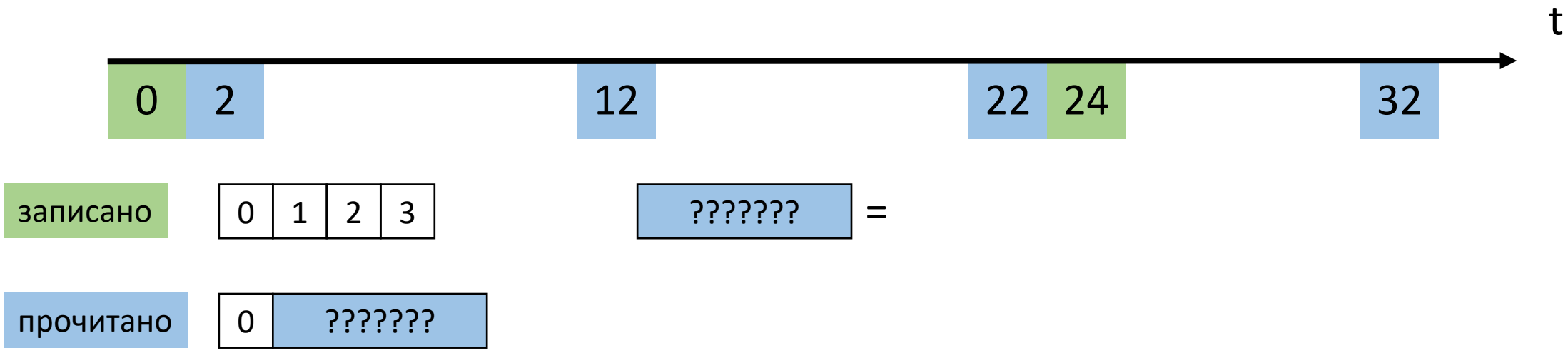
Что прочитает Consumer в момент ***t = 32 часа***?





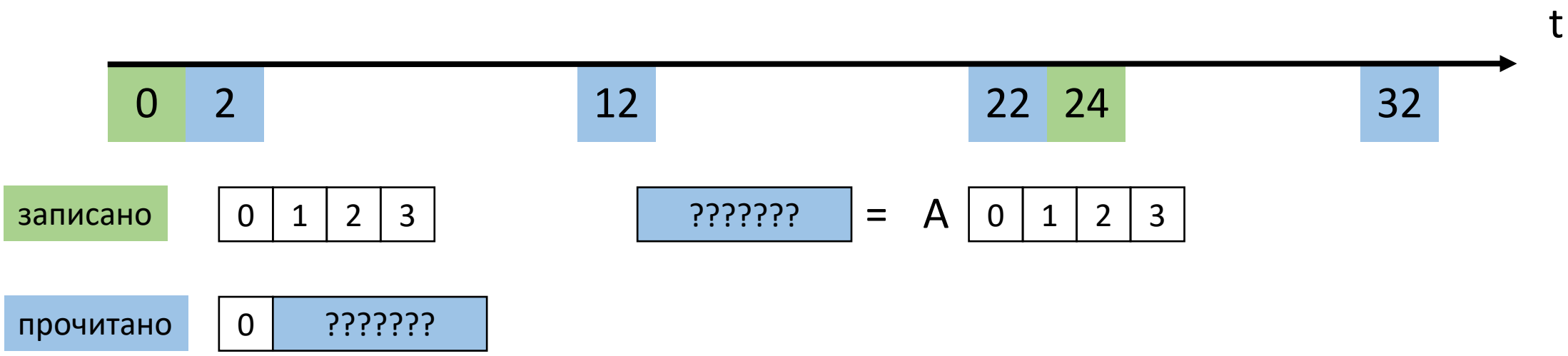
Consumer Offset – Puzzle

Что прочитает Consumer в момент ***t = 32 часа***?



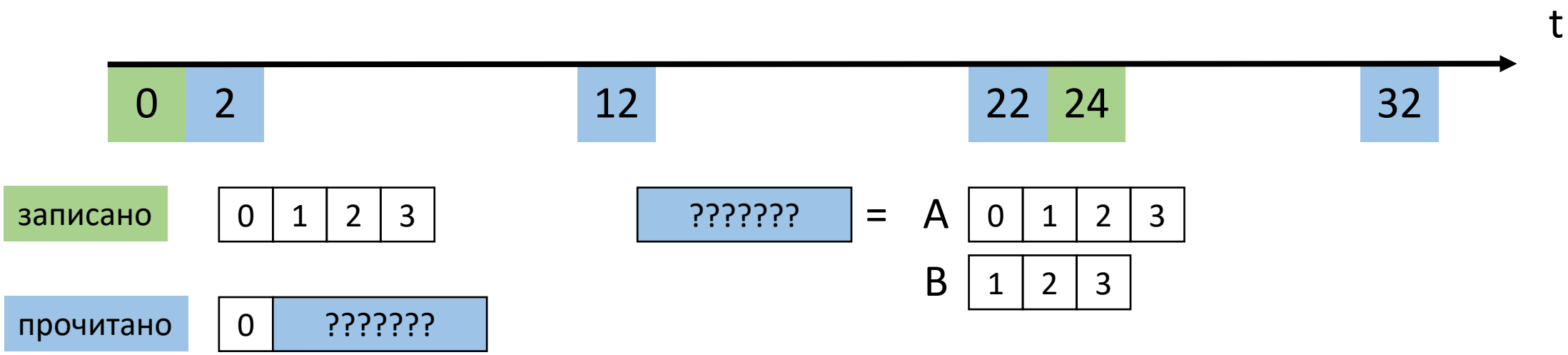
Consumer Offset – Puzzle

Что прочитает Consumer в момент ***t = 32 часа***?



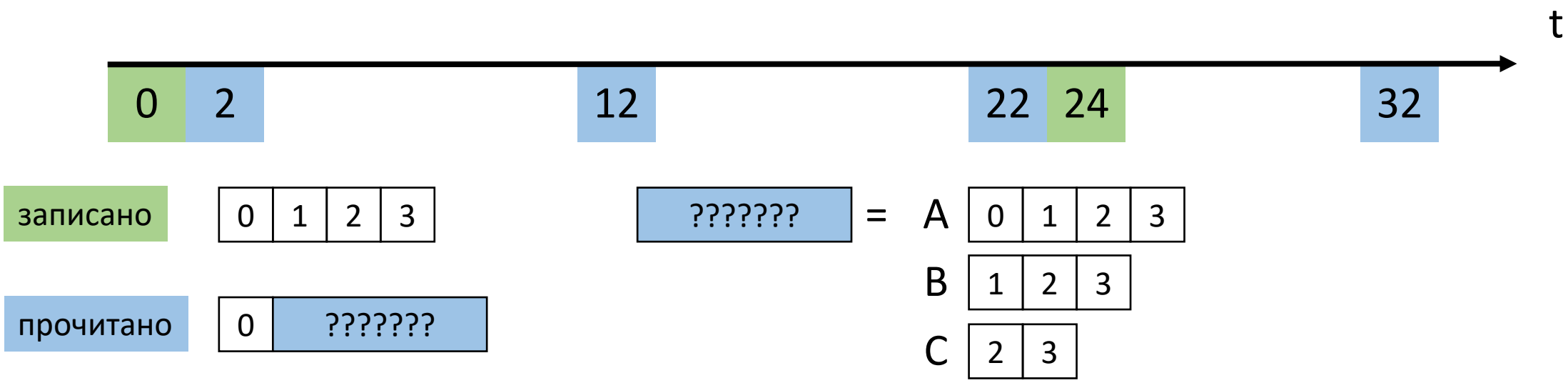
Consumer Offset – Puzzle

Что прочитает Consumer в момент ***t = 32 часа***?



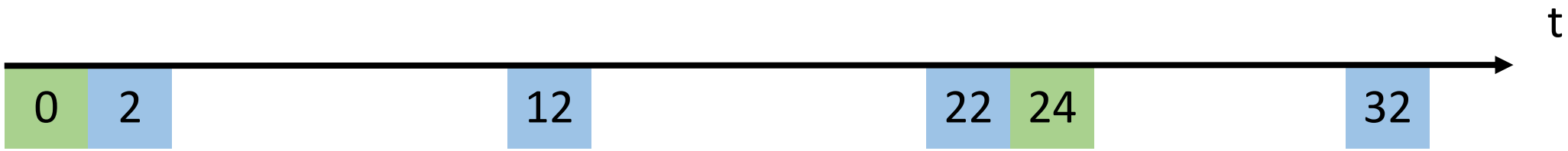
Consumer Offset – Puzzle

Что прочитает Consumer в момент ***t = 32 часа***?

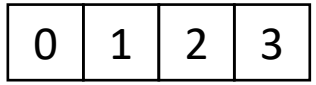


Consumer Offset – Puzzle

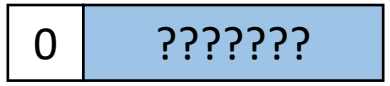
Что прочитает Consumer в момент ***t = 32 часа***?

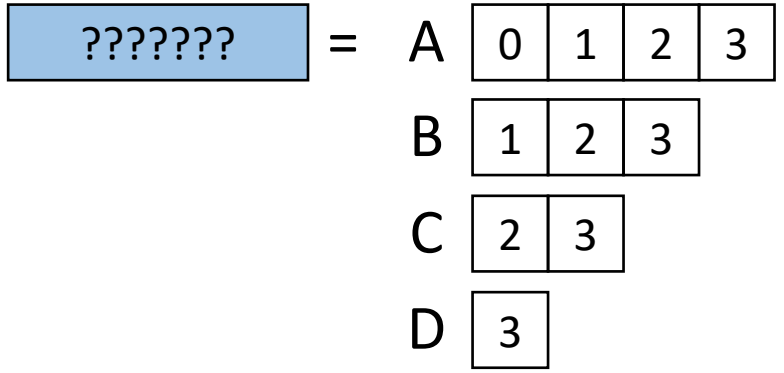


записано



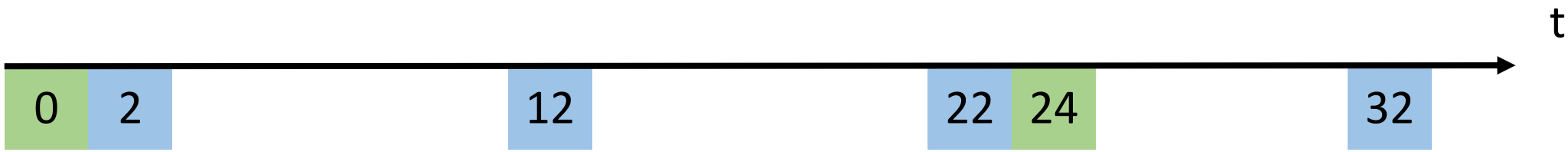
прочитано





Consumer Offset – Puzzle

Что прочитает Consumer в момент ***t = 32 часа***?



записано | 0 | 1 | 2 | 3 |

прочитано | 0 | ??????? |

??????? =

A | 0 | 1 | 2 | 3 |

B | 1 | 2 | 3 |

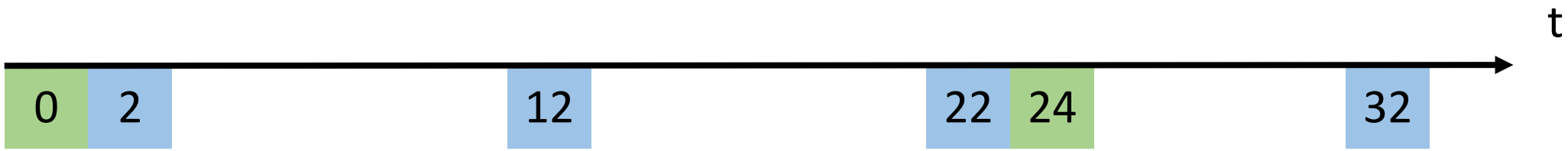
C | 2 | 3 |

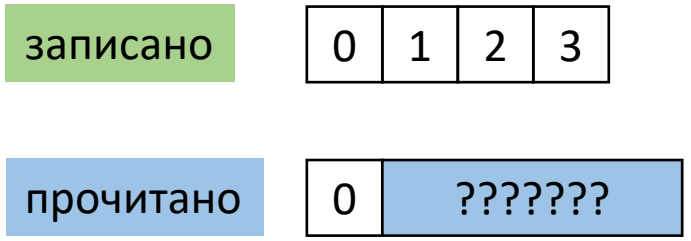
D | 3 |

E *ничего*

Consumer Offset – Puzzle

Что прочитает Consumer в момент ***t = 32 часа***?





??????? =

A 0 1 2 3

B 1 2 3

C 2 3

D 3

E *ничего*

F *будет ошибка*

Consumer Offset – Хранение

- Consumer Offset хранятся сутки по умолчанию (7 дней с 2.0)

Consumer Offset – Хранение

- Consumer Offset хранятся сутки по умолчанию (7 дней с 2.0)
- KIP-186: Increase offsets retention default to 7 days (2.0)

Consumer Offset – Хранение

- Consumer Offset хранятся сутки по умолчанию (7 дней с 2.0)
- KIP-186: Increase offsets retention default to 7 days (2.0)
- Если в топик редко пишут **или** читают, то Consumer всё «забудет»

Consumer Offset – Хранение

- Consumer Offset хранятся сутки по умолчанию (7 дней с 2.0)
- KIP-186: Increase offsets retention default to 7 days (2.0)
- Если в топик редко пишут **или** читают, то Consumer всё «забудет»
- KIP-211: Revise Expiration Semantics of Consumer Group Offsets (2.1)

Consumer Offset – auto reset

Consumer Offset – auto reset

- auto.offset.reset = latest

Consumer Offset – auto reset

- auto.offset.reset = **latest**
- latest – сбрасывается на самый «свежий» offset

Consumer Offset – auto reset

- auto.offset.reset = latest
- latest – сбрасывается на самый «свежий» offset
- earliest – сбрасывается на самый ранний offset

Consumer Offset – auto reset

- auto.offset.reset = latest
- latest – сбрасывается на самый «свежий» offset
- earliest – сбрасывается на самый ранний offset
- none – кидает ошибку («У меня лапки»)

Выводы

Выводы

- Внимательное отношение к **настройкам**
- Особенности (недоработки?) клиентского **API**
- Большое количество **рутины**

- Документация о многом умалчивает

Выводы

- Внимательное отношение к **настройкам**
- Особенности (недоработки?) клиентского **API**
- Большое количество **рутины**

- Документация о многом умалчивает

- Kafka – лучшее, что есть…

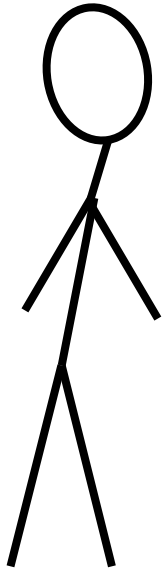
Выводы

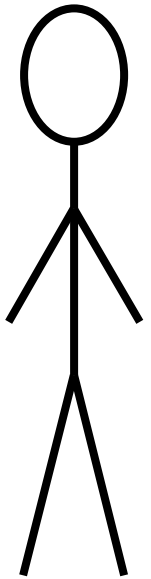
- Внимательное отношение к **настройкам**
- Особенности (недоработки?) клиентского **API**
- Большое количество **рутины**

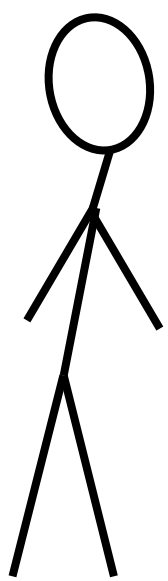
- Документация о многом умалчивает

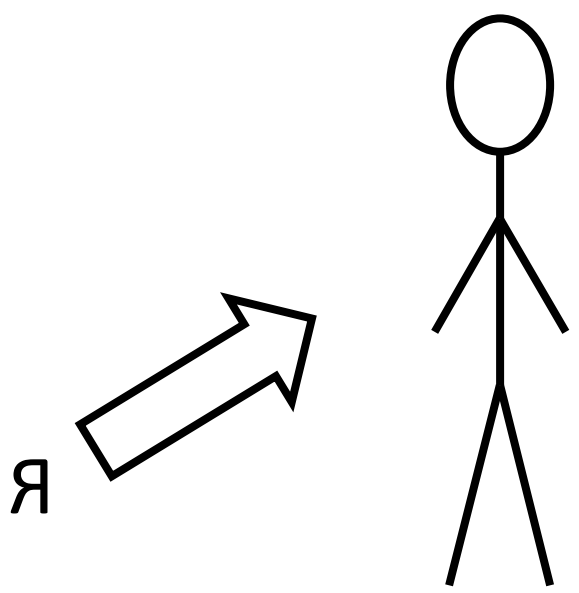
- Kafka – лучшее, что есть… И она классная

* В СКОРОМ ВРЕМЕНИ *

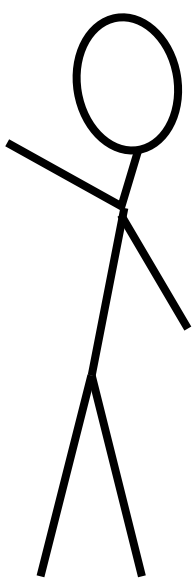


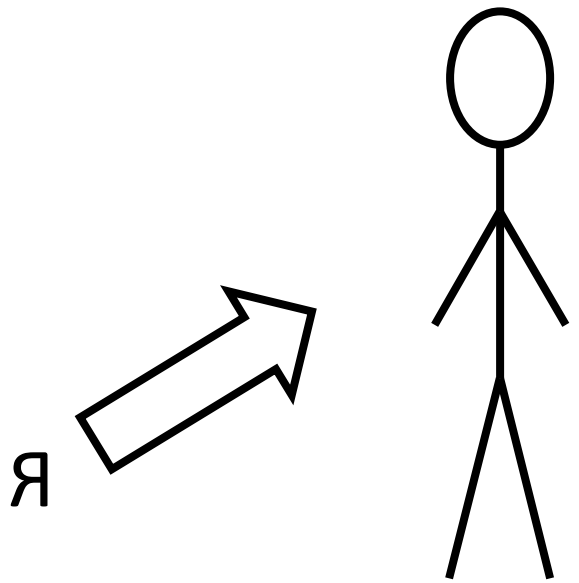




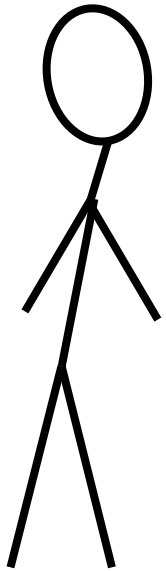

Я

НОВЫЙ ПРОЕКТ НАМЕЧАЕТСЯ,
ДУМАЕМ НАД ВЫБОРОМ
ТЕХНОЛОГИЙ…

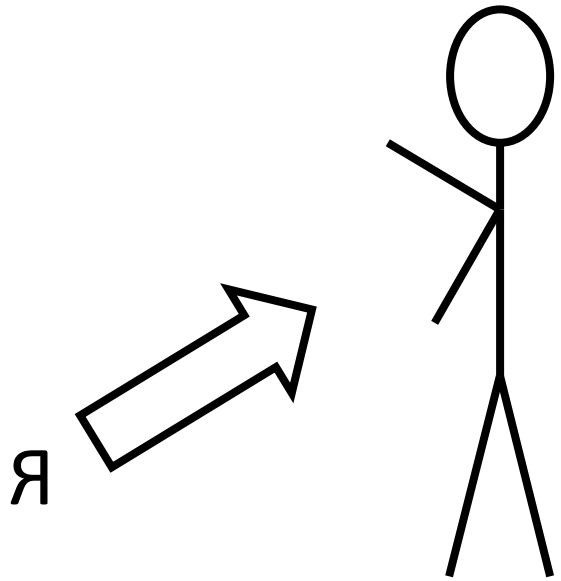




НОВЫЙ ПРОЕКТ НАМЕЧАЕТСЯ,
ДУМАЕМ НАД ВЫБОРОМ
ТЕХНОЛОГИЙ…

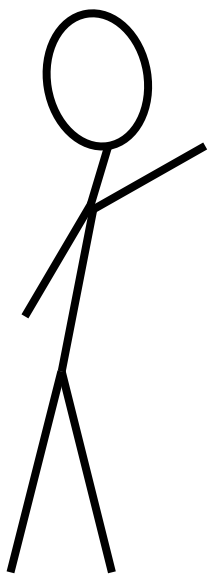


МНОГО ЖЕ ВСЕГО!
ДАВАЙ ДЕТАЛИ!

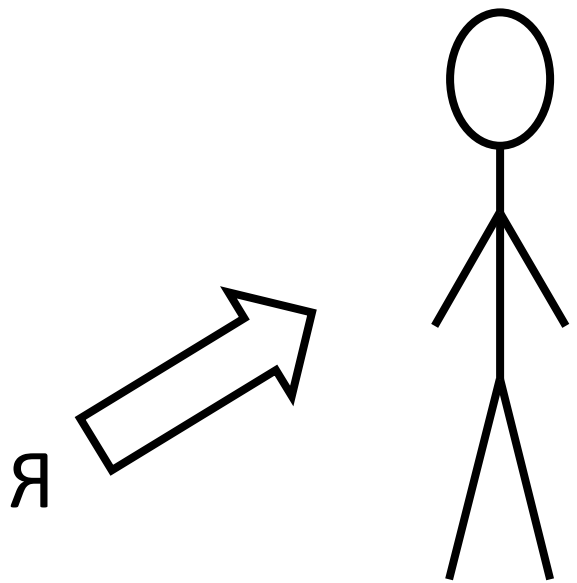


НОВЫЙ ПРОЕКТ НАМЕЧАЕТСЯ,
ДУМАЕМ НАД ВЫБОРОМ
ТЕХНОЛОГИЙ…

ОБЕЩАЮТ ХАЙЛОАД,
БИГ-ДАТУ,
ВОТ ЭТО ВСЁ!

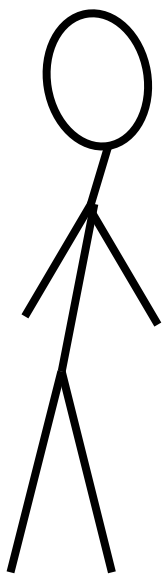


МНОГО ЖЕ ВСЕГО!
ДАВАЙ ДЕТАЛИ!



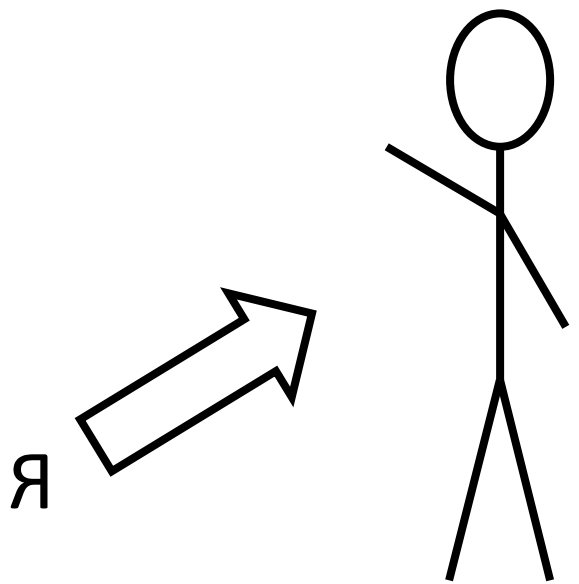
НОВЫЙ ПРОЕКТ НАМЕЧАЕТСЯ,
ДУМАЕМ НАД ВЫБОРОМ
ТЕХНОЛОГИЙ…

ОБЕЩАЮТ ХАЙЛОАД,
БИГ-ДАТУ,
ВОТ ЭТО ВСЁ!



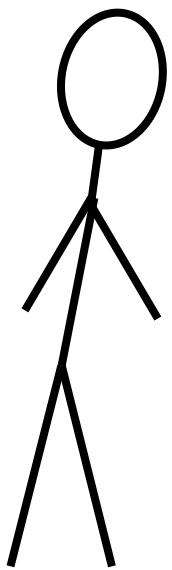
МНОГО ЖЕ ВСЕГО!
ДАВАЙ ДЕТАЛИ!

А-А-А!
К ЧЁРТУ ВСЁ!!!



НОВЫЙ ПРОЕКТ НАМЕЧАЕТСЯ,
ДУМАЕМ НАД ВЫБОРОМ
ТЕХНОЛОГИЙ…

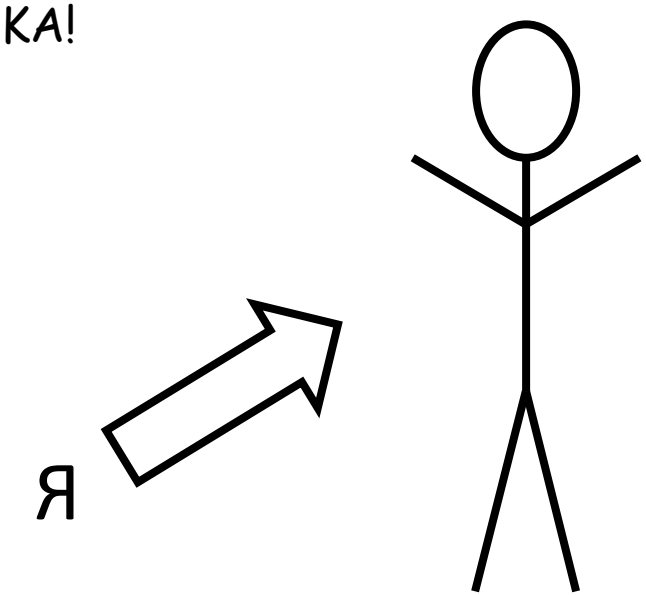
ОБЕЩАЮТ ХАЙЛОАД,
БИГ-ДАТУ,
ВОТ ЭТО ВСЁ!



МНОГО ЖЕ ВСЕГО!
ДАВАЙ ДЕТАЛИ!

А-А-А!
К ЧЁРТУ ВСЁ!!!

KAFKA! KAFKA!
KAFKA!



GregoryKoshelev

K_Gregory

gnkoshelev

tech.kontur.ru