Apache Kafka
Spark Streaming

KARPOV.COURSES

Message broker

Архитектурный паттерн в распределённых системах; приложение, которое преобразует сообщение по одному протоколу от приложения-источника в сообщение протокола приложения-приёмника, тем самым выступая между ними посредником (Wikipedia)

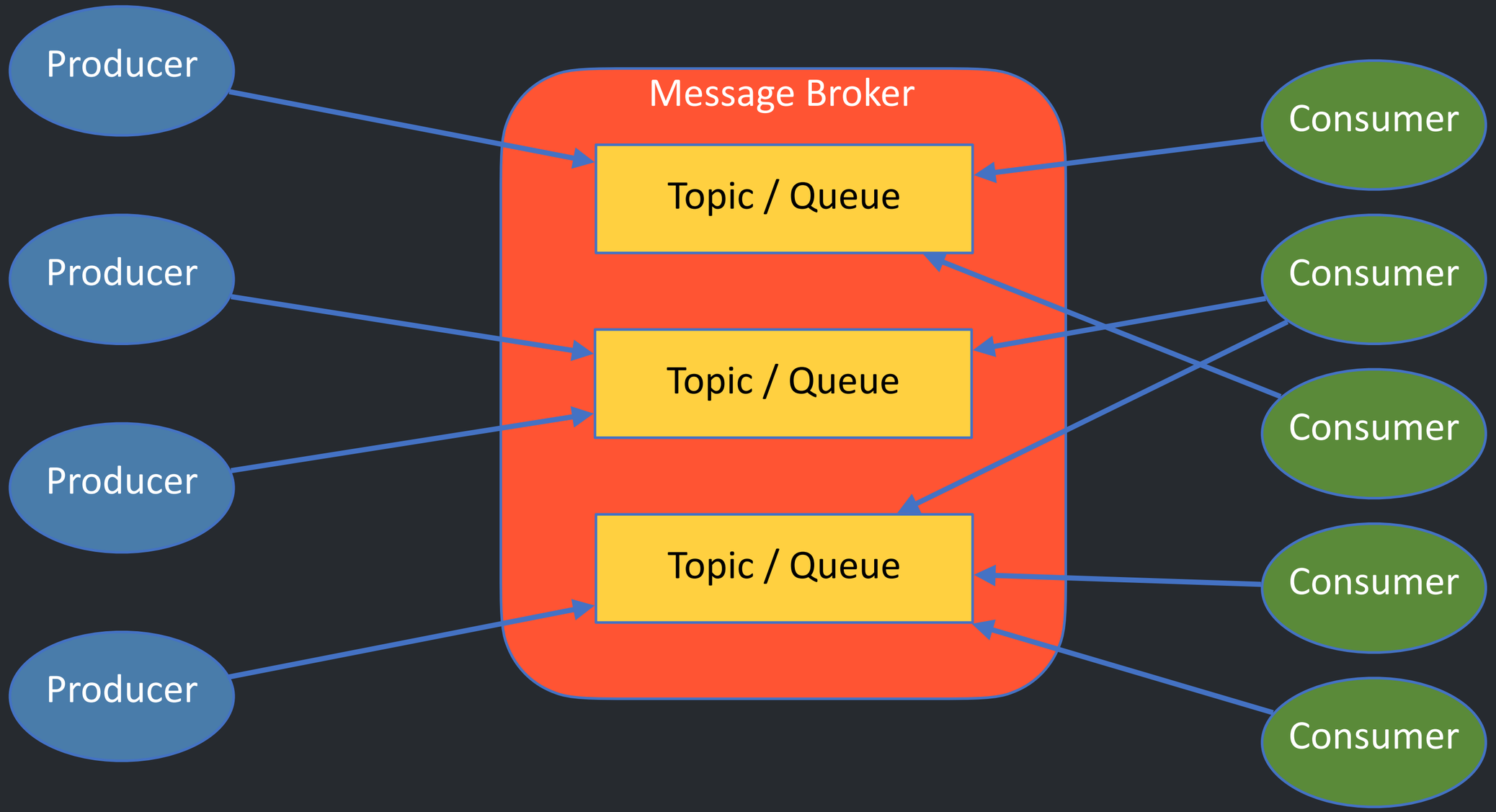
Типы:

- point-to-point
- publish / subscribe

Message broker назначение

- интеграция систем с разными протоколами
- роутинг сообщений
- надежное хранение
- гарантированная доставка
- масштабирование (как источников, так и потребителей)
- преобразование сообщений
- интерграция с внешними системами

Message broker назначение

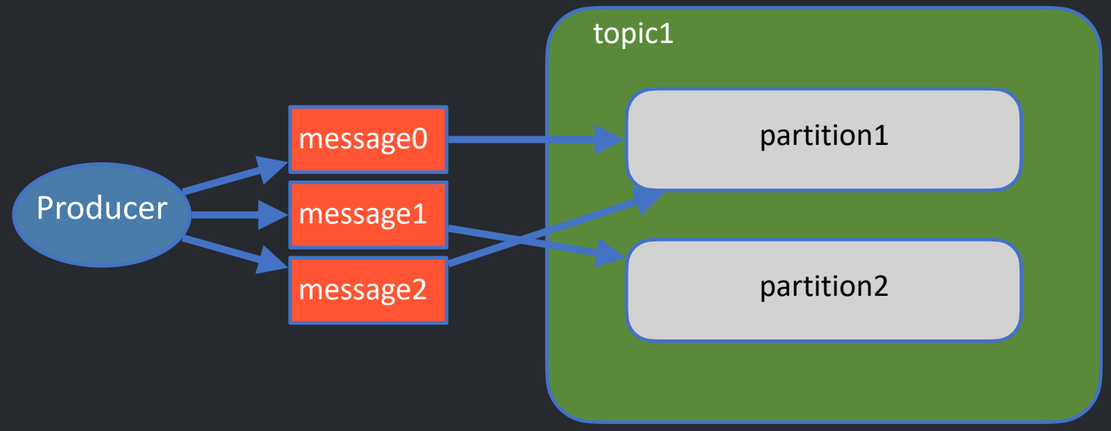


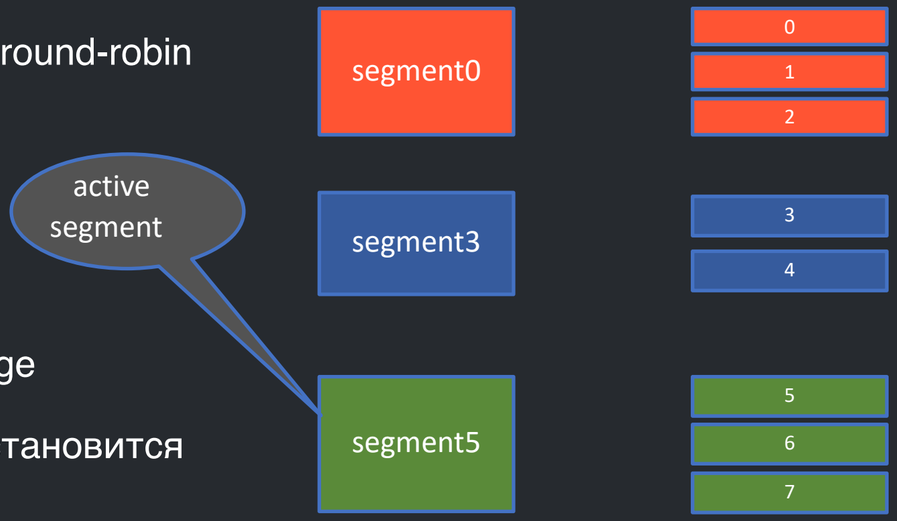
Apache Kafka

- изначально внутренняя разработка LinkedIn (open sourced 2011)
- названа в честь Франца Кафки (Franz Kafka) потому что “система оптимизирована для записи” и Jay Kreps нравились произведения автора
- работает по модель publish / subscribe
- гибко масштабируется
- репликация
- данные хранятся в topic
- сообщения - key - value + служебный заголовок (header)
- навигация на основе offset (смещения)
- данные хранятся согласно log retention и cleanup policy

Внутренняя иерархия

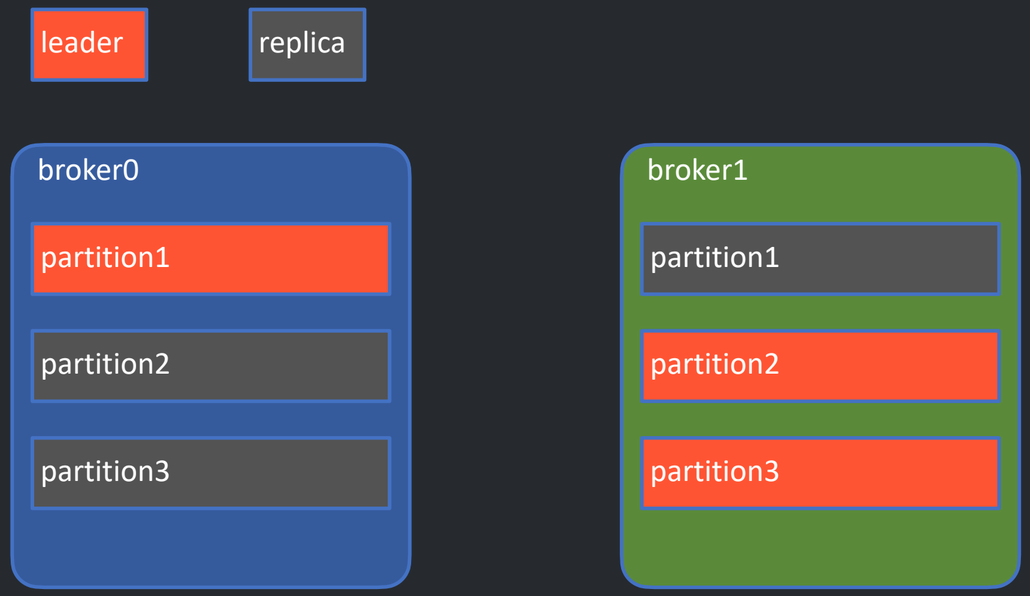
- topic
  - логическая append-only очередь сообщений (message)
  - состоит из 1+ партиций (partition)
- partition
  - физическая единица хранения данных topic
  - можно писать, нельзя удалять
  - **привязана** к broker
  - message -> partition на основаннии key (hash % partition num или round-robin если key не заполнен)
- segment
  - набор записей внутри partition
  - всегда есть active segment в который идет запись новых message
  - при превышении размера сегмента создается новый, который становится активным





Внутренняя иерархия

- message
  - Заголовок
    - topic + partition + offset +timestamp
    - additional headers (Map[String, String])
  - body
    - key - byte[]
    - value - byte[]
- broker
  - обслуживает topic partitions
  - возможна репликация (leader + followers)



Log retention

- time based
  - по истечении срока маркирует данные
    - log.retention.ms
    - log.retention.minutes
    - log.retention.hours
- size based
  - log.retention.size

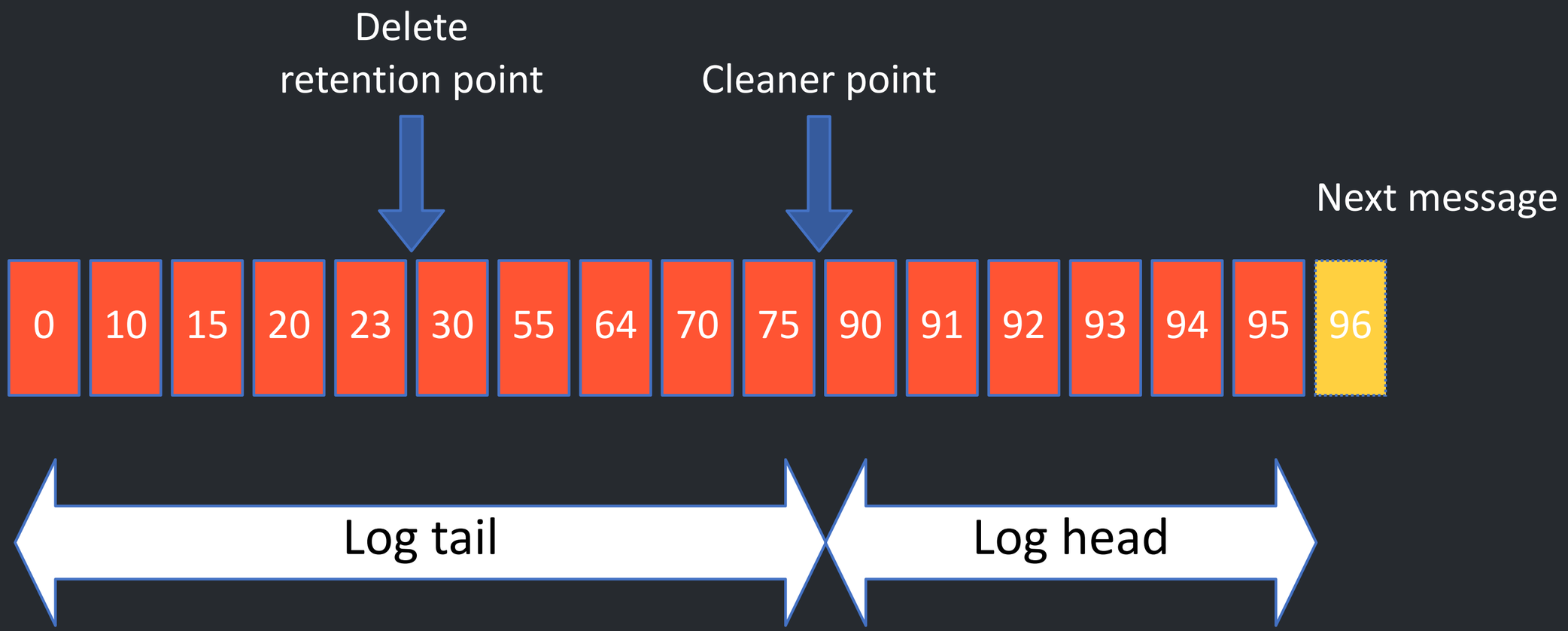
Log retention

- time based
  - по истечении срока маркирует данные
    - log.retention.ms
    - log.retention.minutes
    - log.retention.hours
- size based
  - при превышении размера segment
    - log.retention.size

Cleanup policy

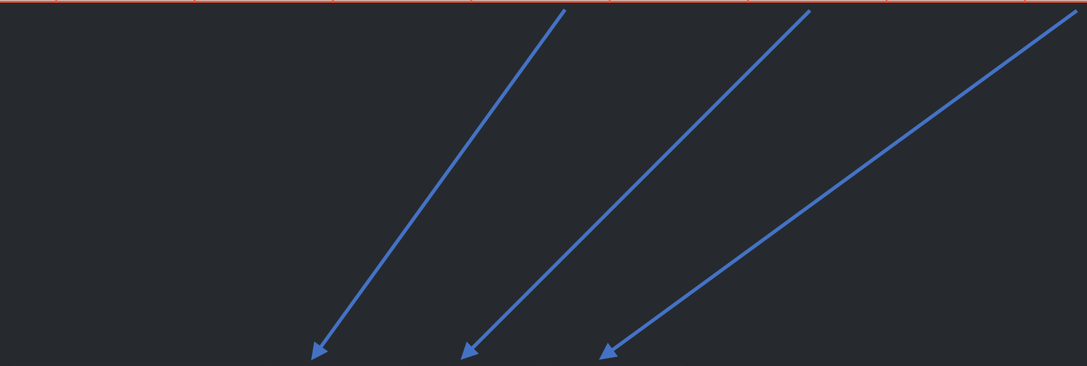
- DELETE (по умолчанию)
  - помечает на удаление segment при устаревании / превышении размера
- COMPACT
  - оставляет только последние сообщения (message) для каждого ключа (message key)
- DELETE and COMPACT
  - производится compaction
  - удаление согласно retention policy

Cleanup policy compaction



Cleanup policy compaction

Offset	0	1	2	3	4	5	6	7	8
Key	K1	K2	K3	K2	K2	K3	K3	K1	K1
Value	V1	V2	V3	V4	V5	V6	V7	V8	V9



Offset	4	6	8
Key	K2	K3	K1
Value	V5	V7	V9

- сбор статистики по каждому ключу (last offset в log head)
- log recopy с удалением старых записей с одинаковым ключом (требуется дополнительное место на диске)
- swap нового и старого лога

Spark streaming

- Оперирует понятием micro-batch
- Discretized Streams (DStreams)
  - появился в Spark 0.7.0
  - RDD-based batch
  - считается устаревшим
- Structure streaming
  - появился в Spark 2.0 (stable в 2.2)
  - DataFrame API
  - catalyst
  - постоянно развивается

Structure streaming

- Концепция “бесконечной” таблицы
- Поддержка watermarks (использование данных прошлых micro-batch)
- DataFrame API (streaming join static)
- fault tolerance
  - checkpoint (информация о последних обработанных offset per partiton per topic)
  - несколько политик для trigger
  - динамические метрики

Structure streaming source

- File
  - text
  - CSV
  - JSON
  - ORC
  - Parquet
- Kafka
- Socket - чтение данных из сокета (обычно для отладки)
- Rate - для генерации данных (тестирование + benchmarking)

Structure streaming sink

- File
- Kafka
- Foreach (foreach & foreachBatch) - позволяет применять операции каждому row / batch. Также позволяет несколько раз переиспользовать данные и писать в несколько output (что нельзя делать для других типов sink)
- Console - для отладки и тестирования
- Memory - хранит данные в памяти (table_name == query_id) - обычно для отладки и демо

Structure streaming triggers

- unspecified (default) - запуск micro-batch по готовности
  - spark.streaming.receiver.maxRate
  - spark.streaming.kafka.maxRatePerPartition
- Fixed interval - запуск через равные промежутки времени
  - предыдущий успел выполниться - ждет конца интервала
  - предыдущий не успел выполниться - ждем окончания и сразу запускается
  - нет данных - не запускается
- One-time - получает все доступные данные и обрабатывает, после чего останавливается
- Continuous with fixed checkpoint interval - *экспериментальный* - позволяет обрабатывать данные с малой задержкой (~1 ms)

СПАСИБО

АНТОН ПИЛИПЕНКО