

Брокеры сообщений в системах обработки данных

Селезнёв Александр Игоревич, ассистент;

Селезнёв Игорь Львович, кандидат технических наук, доцент

Белорусский государственный университет информатики и радиоэлектроники (г. Минск, Беларусь)

В статье рассматриваются особенности работы брокеров сообщений в системах обработки данных. Проведен анализ основных брокеров сообщений, а также их достоинства и недостатки.

Ключевые слова: брокер сообщений, система обработки данных, асинхронная передача данных.

В современных системах обработки данных (СОД) для полноценной работы важна не только работа каждой отдельной части внутренней системы обработки [1], но и производительность работы интерфейсов взаимодействия отдельных частей между собой. Такое взаимодействие может быть как синхронным, так и асинхронным. При синхронном взаимодействии передающая часть системы взаимодействуя с другой частью системы ждёт ответа, и только после его получения продолжает свою работу. При асинхронном взаимодействии передача информации другой части системы не прерывает работу. Таким образом, чем более сложную внутреннюю структуру и логику взаимодействия имеет СОД, тем большей проблемой становится синхронное взаимодействие её элементов за счёт многочисленных простоев. В этом случае используют асинхронное взаимодействие, которое позволяют реализовать брокеры сообщений.

Брокер сообщений — это программный компонент, служащий посредником в коммуникации между различными частями распределенной системы обработки информации. В этом случае брокер сообщений реализуется как часть общей архитектуры системы, либо как отдельный сервис [2]. Существуют два основных типа брокеров сообщений — с организацией взаимодействия производитель-потребитель (producer-consumer) и с организацией взаимодействия издатель-подписчик (publisher-subscriber):

1. Брокер сообщений с организацией взаимодействия производитель-потребитель позволяет организовывать прямую передачу информации конечному потребителю

и сам занимается процессом передачи. Структура типового брокера с данным типом взаимодействия изображена на рисунке 1.

Из рисунка 1 видно, что «Производитель» записывает информацию в две очереди сообщений, затем выбирает «Сообщение» из второй очереди и посылает его получателю «Потребитель». Такая структура брокера сообщений характера для брокеров, построенных на использовании очередей (queuebased).

2. Брокер сообщений с организацией взаимодействия издатель-подписчик позволяет организовывать такое взаимодействие, при котором активными элементами системы выступают подписчики, которые производят чтение информации из тем (topics), на которые они подписаны (рис. 2).

Из рисунка 2 видно, что «Издатель» производит публикацию сообщений с помощью двух тем, при этом в системе имеется потребитель «Подписчик», который «читает» вторую тему, получая возможность просматривать «Сообщение 1» и «Сообщение 2» с помощью настроенных прав доступа для него.

В этом случае у потребителя «Подписчик» нет доступа к первой теме. Такая структура брокера сообщений характерна для потоковых (streaming) брокеров сообщений.

При использовании брокеров сообщений следует учитывать их преимущества и недостатки. Преимущества брокеров сообщений:

1. Отказоустойчивость и масштабируемость. Применение брокеров сообщений повышает отказоустойчивость за счет использования типовых методов взаимодей-

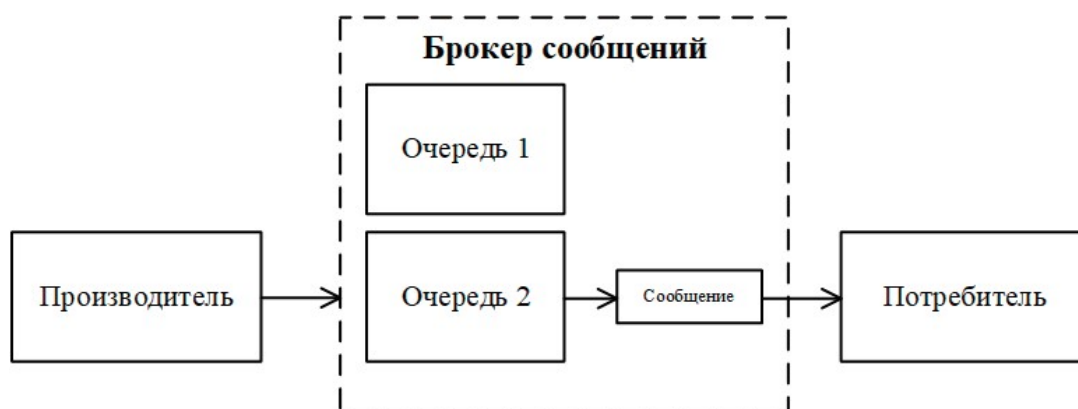


Рис. 1. Структура брокера сообщений с организацией взаимодействия производитель-потребитель

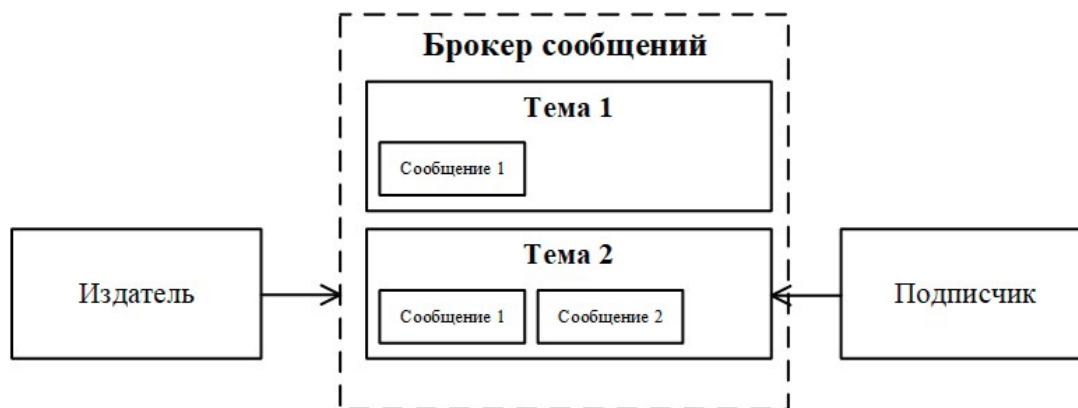


Рис. 2. Структура брокера сообщений с организацией взаимодействия издатель-подписчик

ствия всех элементов системы между собой и возможность реализовать кластеры брокеров.

2. Разделение слоев исполнения и интерфейсов взаимодействия. Такое разделение слоев приложения уменьшает связность между ними, предоставляя возможность сделать приложение более модульным, гибким и удобным в разработке.

3. Гарантии доставки сообщений. Если получатель не готов обработать полученную информацию в данный момент, то она сохраняется в очереди с возможностью отложенной обработки.

4. Повышение производительности. Использование брокеров повышает производительность благодаря асинхронной обработке сообщений.

5. Безопасность. Брокеры сообщений предоставляют механизмы шифрования и аутентификации, обеспечивая безопасность обмена данными.

Недостатки брокеров сообщений:

1. Усложнение системы. Использование брокеров увеличивает сложность системы за счет добавления нового элемента, что приводит к усложнению отладки, обслуживания и администрирования.

2. Дополнительная точка отказа. Как правило, отказ брокера сообщений является критическим событием, при котором большинство систем не смогут функционировать дальше.

3. Дополнительная нагрузка на сеть. Использование брокеров сообщений может повлечь за собой дополнительную нагрузку на сеть ввиду увеличения объема передаваемых данных.

4. Возможность конфликтов версий: если участники системы используют разные версии протоколов или разные реализации брокеров сообщений, могут возникнуть проблемы совместимости, что может привести к неправильной обработке или потере сообщений.

В настоящее время наиболее популярными являются такие брокеры сообщений как RabbitMQ, Kafka, AWS SNS/SQS [3].

Брокер сообщений RabbitMQ. RabbitMQ использует улучшенный протокол очередей сообщений (Advanced Message Queuing Protocol, AMQP), представляющий собой

открытый протокол передачи сообщений. RabbitMQ базируется на структуре брокера сообщений с организацией взаимодействия производитель-потребитель (рис. 1), в которой в брокер дополнительно включены блоки «обменника» (exchange) и маршрутизации и хранения правил доступа для «обменника» (routing-binding). Типовой путь следования информации в этом брокере сообщений следующий — производитель отправляет сообщения в брокер, сообщения поступают в «обменник», затем эти сообщения попадают в блок маршрутизации и согласно установленным правилам доступа передаются в нужные очереди сообщений, откуда они уже направляются конечному потребителю. «Обменник» может быть сконфигурирован следующим образом:

1. «Маршрутизация без маршрутизации» (Fanout exchange) — отправка сообщений происходит во все очереди.

2. Маршрутизация по ключу (Direct exchange). В RabbitMQ существует ключ маршрутизации (routing-key), который выступает в качестве категории сообщения. В зависимости от ключа маршрутизации происходит распределение сообщений по очередям.

3. Маршрутизация по шаблону (Topic exchange) — маршрутизация происходит в зависимости от шаблонов, заданных в ключе маршрутизации.

Достоинства RabbitMQ [4]:

1. Гибкая маршрутизация, позволяющая доставлять определенные сообщения в требуемые очереди и конкретным потребителям.

2. Несколько типов систем обмена сообщениями, позволяющих по-разному направлять сообщения потребителям.

3. Простота развертывания на веб-серверах и в общедоступных облаках благодаря относительно небольшому объему программного кода.

Недостатки RabbitMQ:

1. Невозможность просмотра очередей и сообщений в очереди.

2. Замедление работы при высокой нагрузке. У RabbitMQ есть трудности при горизонтальном масштабировании в кластере, которые заключаются в добавлении дополнительных сложных настроек кластеризации, что замедляет работу системы.

Брокер сообщений Kafka. Ядром функциональности брокера сообщений Kafka является исполнение трех действий над данными — запись, хранение в течение заданного времени и выдача по запросу. Структурно Kafka основан на брокере сообщений с организацией взаимодействия издатель-подписчик (рис. 2). В Kafka данные физически хранятся на диске в виде партиций (частей, partitions). Новые сообщения добавляются в журнал фиксаций (commit log) брокера. Они помещаются строго в конец этого журнала, и их порядок после этого не меняется, благодаря чему в каждой отдельной партиции сообщения всегда расположены в порядке их добавления. Функцию очереди в Kafka выполняет тема (topic), которая нужна для объединения нескольких партиций в общий поток. Таким образом сообщение, относящиеся к одной теме, может храниться в нескольких разных партициях, из которых подписчик извлекает их по запросу. Используется pull-механизм, а не push-механизм — в отличие от RabbitMQ, где сообщения загружаются в потребителей — в Kafka подписчики сами извлекают нужные им сообщения (согласно установленным правам доступа). Чтобы начать читать сообщения с произвольного места в очереди, подписчик сообщает брокеру Kafka номер партиции и номер сообщения. Данные в партиции постоянны и могут быть повторно использованы, что повышает пропускную способность сети. В брокере также реализована возможность создания групп подписчиков и тонкая настройка их взаимодействия с темами в брокере сообщений.

Достоинства Kafka:

1. Простота использования. Сам брокер делает лишь две вещи — записывает и отдает сообщения.
2. Высокая пропускная способность.
3. Позволяет читать сообщения в произвольном порядке.
4. Позволяет читать группы сообщений одновременно. К примеру, можно запросить сразу 1000 сообщений, что снижает нагрузку на сеть.

Недостатки Kafka:

1. Сложности с обработкой «испорченных» сообщений, для которых требуется выделять отдельное место хранения с помощью организации очереди мертвых писем (dead letter queue).
2. Необходимость в учёте последнего прочитанного сообщения для каждого пользователя, ввиду того что они читают сообщения, а не сообщения доставляются пользователям.

Брокер сообщений Amazon Simple Queue Service & Simple Notification Service. Простой сервис уведомлений (Simple Notification Service, SNS) — брокер сообщений, принцип работы которого основан на организации взаимодействия производитель-потребитель (рис. 1). Простой сервис очередей (Simple Queue Service, SQS) — брокер сообщений, основанный на организации взаимодействия издатель-подписчик (рис. 2), при этом вместо тем используется только одна очередь со схожей структурой.

SQS можно использовать совместно с SNS — в этом случае становится возможным организация полноценного взаимодействия издатель-подписчик с различным числом подписчиков, обрабатывающих настроенные именно для них очереди.

Amazon SQS предлагает два типа очередей сообщений:

1. Стандартные очереди — обеспечивают максимальную пропускную способность и доставку сообщений по принципу «хотя бы один раз». Стандартная очередь старается соблюдать порядок сообщений, но он может быть нарушен.

2. Очереди FIFO с ограниченной пропускной способностью — гарантируют, что сообщения будут обрабатываться строго однократно и исключительно в порядке отправления. Они предназначены для улучшения обмена сообщениями между приложениями, когда порядок операций и событий имеет решающее значение или когда дублирование недопустимо.

Amazon SNS предназначен для обмена сообщениями в крупных распределённых системах: например, для параллельной обработки данных на большом количестве агентов, отправки пользователям уведомлений, обновления записей в базах данных.

Пропускная способность SNS практически не ограничена. SNS не регулирует частоту доставки получателю. Если конечный сервис недоступен, SNS повторяет отправку в соответствии с установленными правилами. Это значит, что, если получатель какое-то время был недоступен, впоследствии он может быть завален повторными сообщениями.

Так как компания Amazon полностью владеет всей экосистемой Amazon Web Services (AWS), в том числе SQS и SNS, то в этих брокерах присутствует ряд ограничений присущий проприетарным продуктам [6].

Достоинства SQS:

1. Популярность во всем мире. Многие крупные компании используют инфраструктуру Amazon, поэтому брокер актуален.
 2. Безопасность. Шифрование всех сообщений, которые обрабатывает брокер.
 3. Автоматическое резервное копирование данных.
 4. Интеграция с другими сервисами компании Amazon.
- Недостаток SQS — полная зависимость от AWS и очень сложный перенос данных на другую платформу.

Выводы

Использование асинхронного взаимодействия между частями системы и, как следствие, использование брокеров сообщений в современных системах обработки данных является обязательным ввиду многочисленных преимуществ присущих им.

Выбор конкретного брокера сообщений зависит от характера проектируемой системы обработки данных и выбранных критериев разработки — небольшие СОД могут использовать RabbitMQ, при большем масштабировании предпочтительней использовать Kafka. Брокеры сооб-

щений Amazon SQS являются хорошим решением в том случае, если остальные части СОД находятся в экосистеме AWS.

Правильно выбранный брокер сообщений, отвечающий требованиям конкретной системы обработки

данных, упрощает её масштабирование, однотипно структурирует внутреннюю логику путем разграничения исполняемой части и интерфейсов взаимодействия, увеличивает производительность всей системы в целом.

Литература:

1. Селезнёв, А. И. Обобщенная модель построения системы обработки данных / А. И. Селезнёв, И. Л. Селезнёв. — Текст: непосредственный // Молодой учёный. — 2024. — № 29. — С. 22–25.
2. Брокеры сообщений — что это, из чего состоят, плюсы и минусы: сравниваем Apache Kafka, Redis и RabbitMQ // Mediasoft: [сайт]. — URL: <https://academy.mediasoft.team/article/brokery-soobshenii-chto-eto-iz-chego-sostoyat-plyusy-i-minusy-sravnivaem-apache-kafka-redis-i-rabbitmq/> (дата обращения: 02.11.2025)
3. Битва брокеров сообщений: RabbitMQ, Kafka, AWS SNS/SQS // Habr: [сайт]. — URL: https://habr.com/ru/companies/yandex_praktikum/articles/700608/ (дата обращения: 02.11.2025)
4. What is RabbitMQ? // Iron: [сайт]. — URL: <https://blog.iron.io/what-is-rabbitmq/> (дата обращения: 02.11.2025)
5. AWS SNS vs. SQS — What Are the Main Differences? // Awsfundamentals: [сайт]. — URL: <https://awsfundamentals.com/blog/aws-sns-vs-sqs-what-are-the-main-differences> (дата обращения: 02.11.2025)
6. Amazon SNS (from AWS) // Serverless: [сайт]. — URL: <https://www.serverless.com/guides/amazon-sns/> (дата обращения: 02.11.2025)