MQ

Что такое MQ?

Базовая архитектура очереди сообщений (MQ - Messages queue) проста: есть клиентские приложения, называемые производителями (**Producer)**, которые создают сообщения и доставляют их в очередь. Другое приложение, называемое потребителем (**Consumer**), подключается и обрабатывает сообщения. Уведомления, помещенные в очередь, сохраняются до тех пор, пока потребитель не получит их. Messages queue обеспечивает асинхронный протокол связи. Система, которая помещает сообщение в очередь, не требует немедленного ответа на продолжающуюся обработку.

Принципы работы очередей сообщений

Очереди предоставляют буфер для временного хранения сообщений и конечные точки, которые позволяют подключаться к очереди для отправки и получения сообщений в асинхронном режиме.

В сообщениях могут содержаться запросы, ответы, ошибки и иные данные, передаваемые между программными компонентами. Компонент, называемый производителем (Producer), добавляет сообщение в очередь, где оно будет храниться, пока другой компонент, называемый потребителем (Consumer), не извлечет сообщение и не выполнит с ним необходимую операцию.

Брокер сообщений

Асинхронную обработку обычно делают с помощью специальных сервисов — брокеров сообщений. Брокер сообщений представляет собой тип построения архитектуры, при котором элементы системы «общаются» друг с другом с помощью посредника. Благодаря его работе происходит снятие нагрузки с веб-сервисов, так как им не приходится заниматься пересылкой сообщений: всю сопутствующую этому процессу работу он берёт на себя. Сегодня существует множество различных брокеров сообщений и каждый из них хорошо подходит под свои задачи. Концептуально работу брокера сообщений можно описать так:

1. Back-end сервис кладет сообщения в брокера

2. С другой стороны сервис-обработчик (worker) читает сообщения из брокера и как-то их обрабатывает
3. Сообщения внутри брокера буферизируются и формируется очередь. Если в один момент времени нужно записать 10 тыс. сообщений, они запишутся без задержки, а потом обработаются за какой-то промежуток времени

Очереди поддерживают получение сообщений как методом Push, так и методом Pull:

- метод Pull подразумевает периодический опрос очереди получателем по поводу наличия новых сообщений.
- метод Push — отправку уведомления получателю в момент прихода сообщения. Второй метод реализует модель «Издатель/Подписчик» (Publisher/Subscriber). Так как очереди могут использоваться несколькими производителями и потребителями одновременно, обычно их реализуют с помощью дополнительной системы, называемой брокером. Брокер сообщений (Message Broker) занимается сбором и маршрутизацией сообщений на основе предопределенной логики. Сообщения могут передаваться с некоторым ключом — по этому ключу брокер понимает, в какую из очередей (одну или несколько) должно попасть сообщение.

Польза и преимущества очередей сообщений в микросервисной архитектуре

Используя очереди сообщений в качестве основного средства взаимодействия микросервисов (Microservices Communication), можно добиться следующих преимуществ:

1. **Отделение логически независимых компонентов друг от друга (Decoupling)**. Отличительная черта микросервисов — их автономность. И очереди во многом помогают уменьшить зависимости между ними. Каждое

сообщение, передаваемое в очереди, — это всего лишь массив байтов с некоторыми метаданными. Метаданные нужны для направления в конкретную очередь, а информация, содержащаяся в основной части (теле) сообщения, может быть практически любой. Брокер не анализирует данные, он выступает лишь в качестве маршрутизатора. Это позволяет настроить взаимодействие между компонентами, работающими даже на разных языках и платформах.

2. **Улучшение масштабируемости**. Очереди сообщений упрощают независимое масштабирование микросервисов. Наблюдая за состоянием очередей, можно масштабировать те сервисы, на которые приходится большая часть нагрузки. Кроме этого, очереди легко позволяют не только увеличивать число экземпляров существующих сервисов, но и добавлять новые с минимальным временем простоя. Все, что для этого требуется, — добавить нового потребителя, прослушивающего события в очереди.

Однако сами очереди также необходимо масштабировать, и это может создать дополнительные сложности.

1. **Балансировка нагрузки**. Если один из сервисов не справляется с нагрузкой, требуется возможность запускать больше его экземпляров быстро и без дополнительных настроек. Обычно для этих целей используют балансировщик нагрузки, интегрированный с сервером обнаружения служб и предназначенный для распределения трафика. При использовании очередей сообщений сам брокер по умолчанию является балансировщиком нагрузки. Если несколько потребителей слушают очередь одновременно, сообщения будут распределяться между ними в соответствии с настроенной стратегией.
2. **Повышение надежности**. Выход из строя одного из компонентов не сказывается на работе всей системы: при восстановлении он обработает сообщение, находящееся в очереди. Ваш веб-сайт по-прежнему может работать, даже если задерживается часть обработки заказа, например, из-за проблем с сервером БД или системой электронной почты. Правда, при этом очередь сама приобретает статус SPoF (Single Point Of Failure), поэтому необходимо заранее предусмотреть действия на случай ее аварийного отключения.
3. **Безопасность.** Большинство брокеров выполняют аутентификацию приложений, которые пытаются получить доступ к очереди, и позволяют использовать шифрование сообщений как при их передаче по сети, так и при

хранении в самой очереди. Таким образом, очередь снимает с ваших сервисов бремя организации авторизации запросов.

Варианты использования очередей сообщений

Очереди сообщений полезны в тех случаях, где возможна асинхронная обработка. Рассмотрим наиболее частые сценарии использования очередей сообщений (Message Queue use Cases):

1. **Фоновая обработка долгосрочных задач на веб-сайтах.** Сюда можно отнести задачи, которые не связаны напрямую с основным действием пользователя сайта и могут быть выполнены в фоновом режиме без необходимости ожидания с его стороны. Это обработка изображений, преобразование видео в различные форматы, создание отзывов, индексирование в поисковых системах после изменения данных, отправка электронной почты, формирование файлов и так далее.
2. **Буферизация при пакетной обработке данных.** Очереди можно использовать в качестве буфера для некоторой массовой обработки, например пакетной вставки данных в БД или HDFS. Очевидно, что гораздо эффективнее добавлять сто записей за раз, чем по одной сто раз, так как сокращаются накладные расходы на инициализацию и завершение каждой операции. Но для стандартной архитектуры может стать проблемой генерация данных клиентской службой быстрее, чем их может обработать получатель. Очередь же предоставляет временное хранилище для пакетов с данными, где они будут храниться до завершения обработки принимающей стороной.
3. **Отложенные задачи.** Многие системы очередей позволяют производителю указать, что доставка сообщений должна быть отложена. Это может быть полезно при реализации льготных периодов. Например, вы разрешаете покупателю отказаться от размещения заказа в течение определенного времени и ставите отложенное задание в очередь. Если покупатель отменит операцию в указанный срок, сообщение можно удалить из очереди.
4. **Сглаживание пиковых нагрузок.** Помещая данные в очередь, вы можете быть уверены, что данные будут сохранены и в конечном итоге обработаны, даже если это займет немного больше времени, чем обычно, из-за большого скачка трафика. Увеличить скорость обработки в таких случаях также возможно — за счет масштабирования нужных обработчиков.

5. **Гарантированная доставка при нестабильной инфраструктуре.** Нестабильная сеть в сочетании с очередью сообщений создает надежный системный ландшафт: каждое сообщение будет отправлено, как только это будет технически возможно.
6. **Упорядочение транзакций.** Многие брокеры поддерживают очереди FIFO, полезные в системах, где важно сохранить порядок транзакций. Если 1000 человек размещают заказ на вашем веб-сайте одновременно, это может создать некоторые проблемы с параллелизмом и не будет гарантировать, что первый заказ будет выполнен первым. С помощью очереди можно определить порядок их обработки.
7. **Сбор аналитической информации.** Очереди часто применяют для сбора некоторой статистики, например использования определенной системы и ее функций. Как правило, моментальная обработка такой информации не требуется. Когда сообщения поступают в веб-службу, они помещаются в очередь, а затем при помощи дополнительных серверов приложений обрабатываются и отправляются в базу данных.
8. **Разбиение трудоемких задач на множество маленьких частей.** Если у вас есть некоторая задача для группы серверов, то вам необходимо выполнить ее на каждом сервере. Например, при редактировании шаблона мониторинга потребуется обновить мониторы на каждом сервере, использующем этот шаблон. Вы можете поставить сообщение в очередь для каждого сервера и выполнять их одновременно в виде небольших операций.
9. **Прочие сценарии, требующие гарантированной доставки информации и высокого уровня отказоустойчивости.** Это обработка финансовых транзакций, бронирование авиабилетов, обновление записей о пациентах в сфере здравоохранения и так далее.

Сложности использования и недостатки очередей сообщений: как с ними справляться

Несмотря на многочисленные преимущества очередей сообщений, самостоятельное их внедрение может оказаться довольно сложной задачей по нескольким причинам:

- По сути, это еще одна система, которую необходимо купить/установить, правильно сконфигурировать и поддерживать. Также потребуются

дополнительные мощности.

- Если брокер когда-либо выйдет из строя, это может остановить работу многих систем, взаимодействующих с ним. Как минимум необходимо позаботиться о резервном копировании данных.
- С ростом числа очередей усложняется и отладка. При синхронной обработке сразу очевидно, какой запрос вызвал сбой, например, благодаря иерархии вызовов в IDE. В очередях потребуется позаботиться о системе трассировки, чтобы быстро связать несколько этапов обработки одного запроса для обнаружения причины ошибки.
- При использовании очередей вы неизбежно столкнетесь с выбором стратегии доставки сообщений. В идеале сообщения должны обрабатываться каждым потребителем однократно. Но на практике это сложно реализовать из-за несовершенства сетей и прочей инфраструктуры. Большинство брокеров поддерживают две стратегии: доставка хотя бы раз (At-least-once) или максимум раз (At-most-once). Первая может привести к дубликатам, вторая — к потере сообщений. Обе требуют тщательного мониторинга. Некоторые брокеры также гарантируют строго однократную доставку (Exactly-once) с использованием порядковых номеров пакетов данных, но даже в этом случае требуется дополнительная проверка на стороне получателя.

В каких случаях очереди неэффективны

Конечно, очереди не являются универсальным средством для любых приложений. Рассмотрим варианты, когда очереди не будут самым эффективным решением:

- У вашего приложения простая архитектура и функции, и вы не ожидаете его роста. Важно понимать, что очереди сообщений — это дополнительная сложность. Эту систему также необходимо настраивать, поддерживать, осуществлять мониторинг ее работы и так далее. Да, можно использовать Managed-решение, но вряд ли это будет оправдано для небольших приложений. Добавление очередей должно упрощать архитектуру, а не усложнять ее.
- Вы используете монолитное программное обеспечение, в котором развязка (Decoupling) невозможна или не приоритетна. Если вы не планируете разбивать монолит на микросервисы, но вам требуется асинхронность — для ее реализации обычно достаточно стандартной многопоточной модели.

Очереди могут оказаться избыточным решением до тех пор, пока не возникнет явная необходимость в разделении приложения на автономные компоненты, способные независимо выполнять задачи.

Типы запросов и ответов брокеру сообщений

Для взаимодействия между собой устройства используют различные промышленные протоколы, одним из популярных протоколов для этой цели является **MQTT. MQTT** или Message Queue Telemetry Transport – это легкий, компактный и открытый протокол обмена данными созданный для передачи данных на удалённых локациях, где требуется небольшой размер кода и есть ограничения по пропускной способности канала.

Обмен сообщениями в протоколе MQTT осуществляется между клиентом (client), который может быть издателем или подписчиком (publisher/subscriber) сообщений, и брокером (broker) сообщений (например, Mosquitto MQTT).

Издатель отправляет данные на MQTT брокер, указывая в сообщении определенную тему, топик (topic). Подписчики могут получать разные данные от множества издателей в зависимости от подписки на соответствующие топики.

Всего в протоколе MQTT существует 15 типов сообщений:

Тип сообщения	Значение	Направление передачи	Описание
Reserved	0000 (0)	нет	Зарезервирован
CONNECT	0001 (1)	K* -> C**	Запрос клиента на подключение к серверу
CONNACK	0010 (2)	K <- C	Подтверждение успешного подключения
PUBLISH	0011 (3)	K <- C, K -> C	Публикация сообщения
PUBACK	0100 (4)	K <- C, K -> C	Подтверждение публикации
PUBREC	0101 (5)	K <- C, K -> C	Публикация получена
PUBREL	0110 (6)	K <- C, K -> C	Разрешение на удаление сообщения
PUBCOMP	0111 (7)	K <- C, K -> C	Публикация завершена
SUBSCRIBE	1000 (8)	K -> C	Запрос на подписку
SUBACK	1001 (9)	K <- C	Запрос на подписку принят
UNSUBSCRIBE	1010 (10)	K -> C	Запрос на отписку
UNSUBACK	1011 (11)	K <- C	Запрос на отписку принят
PINGREQ	1100 (12)	K -> C	PING запрос
PINGRESP	1101 (13)	K <- C	PING ответ
DISCONNECT	1110 (14)	K -> C	Сообщение об отключении от сервера
Reserved	1111 (15)		Зарезервирован

Возвращенное значение от брокера может быть следующим:

Значение	Возвращенное значение	Описание
0	0x00 Connection Accepted	Подключение принято
1	0x01 Connection Refused, unacceptable protocol version	Брокер не поддерживает версию протокола, используемую клиентом
2	0x02 Connection Refused, identifier rejected	Client ID подключаемого клиента нет в списке разрешенных
3	0x03 Connection Refused, Server unavailable	Соединение установлено, но MQTT сервис не доступен
4	0x04 Connection Refused, bad user name or password	Не правильный логин или пароль
5	0x05 Connection Refused, not authorized	Доступ к подключению запрещен
6-255		зарезервировано