Что такое очереди сообщений?

Запросы покупателей или клиентов любых онлайн-сервисов обрабатываются на серверах, где этот сервис размещен. Эти запросы должны как можно быстрее дойти до сервера и обратно, чтобы клиент получил тот ответ, который ожидает: «Оплата прошла», «Заказ оформлен» или что-то еще.

Очереди сообщений хранят у себя запросы, чтобы они не потерялись, и передают их серверам постепенно. Например, клиент оформляет заказ в интернет-магазине, и заказ уходит не сразу на сервер, а в очередь. Если сервер откажет, очередь сохранит заказ и отправит его позже, ничего не потеряется.

Таким образом, использование очередей сообщений решает сразу две задачи: сокращает время ожидания пользователя за счет асинхронной обработки и предотвращает потерю информации при сбоях. Но не следует рассматривать очереди как универсальное средство для любого вида приложений: как и у любого инструмента, у них есть свои преимущества и недостатки.

Обработка запросов пользователей, когда их слишком много

Ситуация. Нагрузка на серверы колеблется: в одни часы или дни она выше, в другие ниже. **Что происходит без очередей**. В моменты повышенной нагрузки серверы не справляются. И вместо того чтобы поставить излишек запросов в ожидание, они выдают клиентам отказы: пишут что-то вроде «Невозможно обработать заказ» или совсем отключают нужные функции.

Чтобы этого избежать, можно заранее докупить мощности про запас. Но часто это избыточно: в будущем ресурсы могут не понадобиться, а запросы вполне можно обработать на текущих мощностях, но позже.

Как помогут очереди. Все запросы от клиентов не идут напрямую на сервер, а кладутся в очередь. И очередь отправляет клиенту ответ, например: «Заказ принят». Сервер же берет из очереди запросы по мере того, как освобождается. В итоге обработка займет чуть больше времени, но клиент сразу видит, что его действие выполнено, ни один запрос не пропадает, а сайт и все формы работают без падений и ошибок.

Если ваши серверы находятся в облаке, к ним можно докупить дополнительные мощности временно, на период повышенной нагрузки. Это поможет избежать сбоев, но повысит расходы. С очередями можно ничего не докупать и не подключать, а обойтись обычными мощностями.

Учтите: очереди не помогут, если на запрос нужно выдавать мгновенный ответ. Например, если пользователь ждет, что калькулятор посчитает ему стоимость заказа. Или что заказ сразу уйдет в оплату.

Снижение риска сбоя в системе

Ситуация. В компании может быть много разных серверов: слабее и мощнее, более старые и более новые, уже загруженные работой и практически свободные.
Что происходит без очередей. Запросы просто идут к любому свободному серверу, не учитывая его загруженность и способность выполнить задачу. В случае неравномерного распределения запросов какой-то сервер может простаивать, а какой-то — работать на последнем издыхании и решать задачи гораздо медленнее допустимого.

А может случиться так, что какой-то сервер рухнет или уйдет в перезагрузку. Если при этом на него пойдет запрос, он просто потеряется.

Как помогут очереди. Запросы можно не отправлять напрямую серверам, а класть в очередь. А сервер, когда освободится от прошлой задачи, сам «придет» к очереди и заберет оттуда новый запрос. В итоге тот сервер, что слабее, будет просто обращаться к очереди реже и работать в комфортном темпе. А более мощный — быстрее справляться с задачами и брать новые запросы чаще. Отключившийся или сломавшийся сервер задачу вообще не получит — он просто не сможет обратиться к очереди, пока снова не начнет работать. В итоге ни один запрос не пропадет.

Обработка важных, но не срочных задач в фоновом режиме

Ситуация. В работе IT-системы может быть много дополнительных задач: отправка электронной почты, обработка изображений, формирование отчетов, конвертация файлов в другие форматы. Все эти задачи не требуют мгновенного выполнения, а пользователь готов подождать, если уведомить его, что запрос принят и все в порядке.
Что происходит без очередей. Несрочные задачи уходят на выполнение на сервере сразу. И в итоге могут помешать сделать что-то важное и срочное, например отработать заявку клиента.
Как помогут очереди. Все фоновые запросы не идут на сервер сразу, а кладутся в очередь. Когда сервер освобождается от нагрузки, он сам забирает из очереди то, что может сделать.

В итоге это очень выгодно: в моменты, когда нет срочных запросов, сервер не простаивает, а выполняет несрочные. Он равномерно нагружен и полностью отрабатывает потраченные на него деньги. И тормозов в моменты высокой нагрузки тоже нет, так как на несрочные запросы ресурсы просто не тратятся.

Создание надежной связи между филиалами

Ситуация. Серверы компании могут быть сильно распределены по географии. Например, если головной офис в Москве, а филиалы в Новосибирске и Владивостоке. И между этими серверами нужно обмениваться сообщениями — например, отправлять данные от клиентов из региональной базы в общую.
Что происходит без очередей. Из-за удаленности всегда есть риск, что сообщение не дойдет: канал связи оборвется, передача будет идти слишком долго, что-то

пойдет не так. В итоге можно потерять важные данные. Или сохранить их на региональном сервере, но не обновить на главном.

Иногда этот вопрос решают без очередей. Например, добавляют дополнительную проверку передачи. Но это создает серьезную нагрузку на серверы, так как сообщения становятся тяжелее, а их обработка занимает больше времени.

Как помогут очереди. На все серверы, участвующие в обмене сообщениями, устанавливаются системы очередей. Когда один сервер хочет что-то отправить, то не шлет запрос другому серверу, а кладет его в свою личную очередь. И после этого считает, что запрос отправлен. И уже очередь «стучится» к другому серверу и отправляет сообщение до тех пор, пока отправка не увенчается успехом.

В итоге мощности сервера не заняты постоянными отправками и проверками — он работает в нормальном режиме, считая, что уже отправил сообщение. А персональная очередь будет держать у себя сообщение до успешной отправки и точно ничего не потеряет.

Рассылка уведомлений клиентам

Ситуация. Бывает, что пользователям или клиентам нужно рассылать уведомления.
Что происходит без очередей. Уведомление по умолчанию уходит сразу после формирования. Но это не всегда удобно: иногда формирование может произойти ночью или в нерабочее время. В итоге есть риск, что пользователь пропустит уведомление либо оно вызовет у него раздражение.
Как помогут очереди. После формирования на сервере сообщение никуда не уходит, а кладется в очередь. И ждет там, пока пользователь выйдет в онлайн, у него наступит утро или начнется рабочее время.
Например, так делает социальная сеть «Одноклассники». В 00:00 нового дня они собирают базу всех, у кого сегодня день рождения. Потом отбирают список лучших друзей и ставят уведомление о дне рождения в очередь. Но отправляют его только в тот момент, когда у человека зафиксирована максимальная активность в профиле — чтобы он точно ничего не пропустил и при этом не отвлекался на уведомление в другое время.

Как мы уже видели, сервисы очередей помогают решить множество проблем, связанных с обработкой запросов пользователей. Однако самостоятельно разворачивать, настраивать и администрировать такие сервисы — нетривиальная задача. В облаке сервис очередей можно подключить в пару кликов: Cloud Queues на платформе Mail.ru Cloud Solutions позволяет принимать, обрабатывать и хранить любой объем сообщений.

Кратко о самом важном: зачем компаниям очереди?

- Обрабатывать запросы пользователей в пиковые нагрузки. Не перегружать серверы, избегать ошибок и отказов.

- Снизить риск сбоев и потери данных из-за отказа или медленной работы серверов.
- Обрабатывать важные, но не срочные запросы тогда, когда серверы загружены меньше всего.
- Создать надежную связь между серверами в разных филиалах. Не терять данные, даже если канал связи между филиалами неустойчивый.
- Рассылать пользователям уведомления не в момент формирования, а в удобное время.

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.

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—приложение, которое преобразует сообщение по одному протоколу от приложения-источника в сообщение протокола приложения-приёмника, тем самым выступая между ними посредником. Кроме преобразования сообщений из одного формата в другой, в задачи брокера сообщений также входит:

- проверка сообщения на ошибки;
- маршрутизация конкретному приемнику(ам);
- разбиение сообщения на несколько маленьких, а затем агрегирование ответов приёмников и отправка результата источнику;
- сохранение сообщений в базе данных;
- вызов веб-сервисов;
- распространение сообщений подписчикам, если используются шаблоны типа издатель-подписчик.

Использование брокеров сообщений позволяет разгрузить веб-сервисы в распределённой системе, так как при отправке сообщений им не нужно тратить время на некоторые ресурсоёмкие операции типа маршрутизации и поиска приёмников. Кроме того, брокер сообщений для повышения эффективности может реализовывать стратегии упорядоченной рассылки и определение приоритетности, балансировать нагрузку и прочее.