УДК 004.055

ИСПОЛЬЗОВАНИЕ БРОКЕРОВ СООБЩЕНИЙ В КЛИЕНТ-СЕРВЕРНЫХ ПРИЛОЖЕНИЯХ ОСНОВАННЫХ НА МИКРОСЕРВИСНОЙ АРХИТЕКТУРЕ

Хайба В.И.
д.т.н, профессор каф. ИС Антощук С.Г.
Государственный университет «Одесская политехника», УКРАИНА

АННОТАЦИЯ. В докладе рассмотрен процесс внедрения брокера сообщений для уменьшения связности компонент в клиент-серверных приложениях, реализованных на основе микро-сервисной архитектуры. Веб-приложение представляет собой систему автоматизации процесса продажи автобусных билетов.

Введение. В наше время ИТ компании всё чаще выбирают микро-сервисную архитектуру для реализации клиент-серверных приложений [1]. Основным преимуществом такой архитектуры, по сравнению с монолитной, является лёгкая горизонтальная масштабируемость. Она обеспечивает устойчивую производительность системы за счёт увеличения количества экземпляров отдельного микро-сервиса в момент повышения нагрузки на определенный узел системы. Традиционным подходом для общения между микро-сервисами является HTTP протокол. Однако он имеет недостаток – не позволяет реализовать асинхронную обработку данных, в отличие от брокеров сообщений. **Поэтому целью работы** стало описание процесса использования и внедрения брокеров сообщений в приложения, написанные с использованием микро-сервисной архитектуры, для внедрения асинхронной обработки и упрощения конфигурирования сложных связей между сервисами.

Основная часть работы. В качестве клиент-серверного приложения, которое будет оптимизировано с использованием брокера сообщений выбрана система заказа автобусных билетов. Приложение состоит из следующих сервисов: сервис бронирования и покупок билетов; сервис отправки уведомлений пользователю; сервис аналитики покупок. На рисунке 1 изображена система, сервисы которой общаются через REST протокол. Такой подход имеет следующие недостатки:

- сервис бронирования билетов отправляет сразу два запроса в разные сервисы при наступлении одного события - «Покупка билета». Данное решение сперва не выглядит проблемой, но предположим, что в будущем мы захотим добавить еще два сервиса которые будут связанны с покупкой билета. Нам вновь нужно будет строить связи;
- цепочка из сервисов бронирования, аналитики и отправки уведомлений может иметь негативные последствия из-за своей синхронности так как время выполнения задач в каждом сервисе разное. Это может приводить к простоям отдельных сервисов в определённые периоды времени.

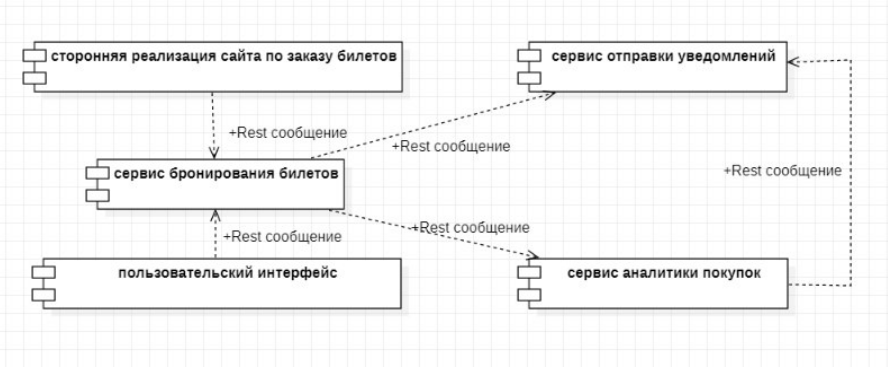

Рис. 1 – Система, построенная с использованием HTTP протокола

Рассмотрим пример той же системы, но с использованием брокера сообщений. Брокер сообщений формирует очередь из сообщений в топике, читатели получают полезную нагрузку

из топиков (них) по мере обработки сообщений, писатели отправляют в топики полезную нагрузку [2]. При помощи брокера легче настраивать и связывать потоки данных, поскольку для этого нужно лишь настроить новый топик и связать его с читателем и писателем, также большим плюсом является возможность подписки нового сервиса на топики без внесения изменений в сервис писатель. В случае же с REST технологией мы вынуждены описывать поток данных в каждом сервисе и связывать их напрямую между собой. Для увеличения горизонтальной масштабируемости можно разбивать топики на партиции – это нужно для масштабирования самого брокера сообщений. Увеличение количества партиций позволяет нескольким микро-сервисам читать не из одной очереди, а из нескольких – тем самым ускоряя процесс считывания данных.

Рис. 2 – Система, построенная с использованием брокера сообщений

Диаграмма на рисунке 2 имеет следующие топики: «отправить рекламу» «билет куплен» «билет в корзине» «билет просмотрен». Они отражают события в системе и в них содержится полезная нагрузка данных событий. Писатели, которые отправляют информацию помечены подписью «пишем в тему», читатели, помечены подписью «читаем из темы». Применение брокера в данном случае позволило уйти от синхронности обработки событий, уменьшить связность сервисов, улучшить возможность дальнейшего редактирования потока данных [3].

Минусами применения брокера сообщений являются:

- необходимость масштабировать, контролировать загруженность брокера и топиков для поддержки работоспособности системы. Масштабирование самих брокеров может стать проблемой для распределенных систем. Это станет еще одной вещью, которую нужно поддерживать вместе с микро-сервисами [4];
- более высокое время доставки сообщения от писателя к читателю;
- увеличение количества расходуемых ресурсов. Для работы брокерам необходимы ресурсы : CPU, память и хранилища. Эти ресурсы могли бы быть использованы для запуска других микро-сервисов.

Выводы. Применение брокера сообщений позволило упростить процесс связывания сервисов приложения. Также брокер позволил уйти от нагромождения прямых связей между группами сервисов при помощи подписки на топики. Кроме того, поток событий перешёл в асинхронное состояние что позволило уменьшить время простоя отдельных сервисов системы до 20%. Теперь топики брокера служат в качестве журнала для данных, проходящих через микро-сервисы, что помогает при расследовании различных инцидентов.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Graig Walls. Spring In Action / Walls Graig. – M.: Manning, 2018. – 498 с.
2. Apache-Kafka Потоковая обработка данных / O`REILLY, 2019 – 320 с.
3. Аналіз брокерів повідомлень RabbitMQ та Apache Kafka [Электронный ресурс]. – Режим доступа URL: https://nauka-online.com/ua/publications/informatsionnye-tehnologii/2018/6/analiz-brokerov-soobshhenij-rabbitmq-i-apache-kafka/
4. Понимание брокеров сообщений. Изучение механики обмена сообщениями посредством ActiveMQ и Kafka. Глава 1 [Электронный ресурс]. – Режим доступа URL: https://habr.com/ru/post/466385/