Інформаційні технології

УДК 004

Будьонний Данило Юрійович
студент
Національного технічного університету України
«Київський Політехнічний Інститут імені Ігоря Сікорського»
Буденный Данил Юрьевич
студент
Национального технического университета Украины
«Киевский политехнический институт имени Игоря Сикорского»
Budonnyi Danylo
Student of the
National Technical University of Ukraine
«Igor Sikorsky Kyiv Polytechnic Institute»

АНАЛІЗ БРОКЕРІВ ПОВІДОМЛЕНЬ RABBITMQ ТА APACHE KAFKA
АНАЛИЗ БРОКЕРОВ СООБЩЕНИЙ RABBITMQ И APACHE KAFKA
ANALYSIS OF THE MESSAGE BROCKERS RABBITMQ AND APACHE KAFKA

Анотація. *У роботі було розглянуто найрозповсюдженіші брокери повідомлень, які використовуються при розробці систем з гарантованою доставкою повідомлень на базі сервіс-орієнтованої архітектури. Проаналізовано характеристики та методи реалізації даних підходів.*

Ключові слова: *брокери повідомлень, черга, повідомлення.*

Аннотация. *В работе были рассмотрены самые распространенные брокеры сообщений, которые используются при разработке систем с гарантированной доставкой сообщений на базе сервис-ориентированной*

архитектуры. Проанализированы характеристики и методы реализации данных подходов.

Ключевые слова: *брокеры сообщений, очередь, сообщения.*

Summary. *The paper considered the most common message brokers, which are used to develop systems with guaranteed delivery of messages based on a service-oriented architecture. The characteristics and methods of implementing these approaches are analyzed.*

Key words: *message brokers, queue, messages.*

Вступ. На сьогодні існує чимало реалізацій брокерів повідомлень в сервіс-орієнтованій архітектурі, тому перед розробниками постає важка задача з вибором найкращої реалізації для поставлених вимог. Тому у данній статті було розглянуто основні реазації брокерів повідомлень, такі як Apache Kafka та RabbitMQ.

Мета. Проаналізувати основні реалзівації брокерів повідомлень, таких як RabbitMQ та Apache Kafka.

Обмін повідомленнями – це спосіб обміну певними даними між процесами, додатками і серверами (як віртуальними, так і фізичними). Такі повідомлення, необхідні для вирішення деяких технічних потреб, вони можуть складатися з будь-яких даних, це може бути як прості текстові повідомлення, так і двійкові дані. Для цього необхідний інтерфейс, керований сторонньою програмою (його називають middleware, також проміжне або міжплатформене ПО). Даним інтерфейсом виступає брокер повідомлень (Message Broker) [1].

Брокери повідомлень – це, як правило, програма-посередник, яка перекладає мову системи з однієї глобальної мови на іншу за допомогою телекомунікаційного середовища. Брокери справляються з прийомом, створенням черги і передачею повідомлень набагато краще і швидше, ніж

будь-які непрофільні аналоги і прості обхідні варіанти (як бази дані, демон cron і тому подібне). Для роботи брокери повідомлень використовують буфери, в яких вони можуть зберігати повідомлення, створюючи чергу повідомлень, яка оброблюється автоматично або шляхом запиту [1].

Брокери повідомлень працюють як посередники для різних сервісів (наприклад, для веб-сервера-додатка). Вони можуть значно скоротити навантаження і терміни доставки повідомлень, оскільки завдання, обробка яких зазвичай займає зовсім трохи часу, передаються сторонній програмі, єдина робота якої полягає у виконанні цих завдань (тобто, робочих процесів). Вони також корисні, коли необхідно гарантувати збереженість передачі інформації з одного місця на інше.

Основна функціональність брокерів охоплює велику кількість галузей:

- Передача повідомлення декількох адресатів для обробки
- Відключені користувачі не втратять дані, оскільки можуть отримати їх пізніше в повному об'ємі
- Впровадження асинхронного режиму роботи з серверними системами
- Впорядкування і визначення пріоритетів завдань
- Балансування навантаження між робочими процесами
- Вища надійність і час безвідмовної роботи додатка

В основному, брокери повідомлення використовуються для:

- Гарантування вільного з'єднання між клієнтом і сервером
- Підтримування асинхронної та не блокуючої комунікації між клієнтом і сервером
- Випадків, коли сервером не може обробити повідомлення так швидко, як їх надсилає клієнт (повідомлення можуть бути додані в чергу)

На рисунку 1 зображений приклад базової архітектура брокера повідомлень. В даному прикладі сервер видає повідомлення до визначеного вузла в брокері і, якщо клієнт підписується на цей конкретний вузол, то клієнт отримує повідомлення, опубліковане сервером. Зворотний хід також можливий. Тіло повідомлення може мати інший формат, такий як JSON, XML, текст, бінарний тощо.

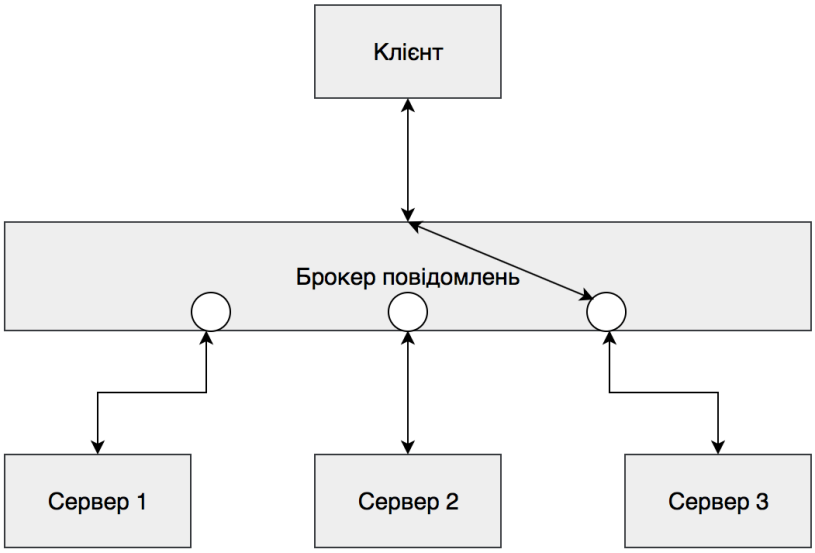

Рис. 1. Базова архітектура брокерів повідомлень

RabbitMQ - популярний брокер повідомлень, який має багато потужних функцій. RabbitMQ написаний на Erlang, не широка використовувану мову програмування, але добре пристосованому для таких завдань. Конфігурація RabbitMQ встановлюється у файлі rabbitmq.config і містить безліч параметрів, що налаштовуються. У термінах клієнтського API RabbitMQ підтримує довгий список мов і деякі стандартні протоколи, наприклад, STOMP, AMQP доступні за допомогою плагіна. Черги і теми можуть створюватися або через веб-інтерфейс, або через клієнтський API безпосередньо.

Існують наступні компоненти RabbitMQ:

- producer - клієнт, який створює повідомлення

- consumer - клієнт, який отримує повідомлення
- queue - необмежена за розміром чергу, яка зберігає повідомлення
- exchange - компонент, який дозволяє переправляти відправлені до нього повідомлення на різні черги

RabbitMQ простий у використанні, підтримує величезну кількість платформ для розробки. Він добре масштабується при додаванні більшого числа серверів.

RabbitMQ має такі властивості:

- Повідомлення, опубліковані в черзі (через обмінні пункти)
- Декілька споживачів можуть одночасно підключитися до черги
- Брокер повідомлень поширює повідомлення серед всіх доступних споживачів
- Повідомлення може бути переадресовано, якщо споживач не спрацює
- Запит на доставку гарантований для черг з одним споживачем (це неможливо, якщо чергу має кілька споживачів)

RabbitMQ підтимує наступних протоколів:

- HTTP, XMPP і STOMP
- Клієнтських бібліотек AMQP для Java і .NET Framework (підтримка інших мов програмування реалізована в ПО інших користувачів)
- Різних плагінів (таких як плагіни для моніторингу і управління через HTTP або веб-інтерфейс або плагін «Shovel» для передачі повідомлень між брокерами)

RabbitMQ дозволяє обробляти порядку 20000 повідомлень в секунду на одну машину [2].

Служба обміну повідомленнями Kafka володіє такими цінними якостями, як швидкість роботи, масштабованість, здатність секціонувати і безліч разів фіксувати одні й ті ж дані в пам'яті. Нижче було перелічено основні відмінності Kafka від традиційних систем обміну повідомленнями:

- Служба Kafka спочатку створювалася і позиціонується як розподілена програма, отже, вона пристосована до масштабування
- Система має відмінну продуктивність - як у випадку публікації повідомлень, так і в разі підписки на них
- Kafka зберігає повідомлення на диску і, таким чином, може використовуватися для пакетної передачі даних (наприклад, для ETL-процесів (Extract, Transform, Load - «витяг, трансформування, завантаження»)

Основними компонентами архітектури Apache Kafka є:

- Потік повідомлень (message) певного типу в термінах служби називається темою (topic). Повідомлення - це корисний для деякого процесу комплект даних, тоді як тема - це категорія, відповідно до якої публікується те або інше повідомлення
- Виробник (producer) - це будь-який процес, який публікує повідомлення у відповідній темі
- Опубліковані повідомлення потім відправляються на зберігання на кластер серверів, іменованих брокерами (brokers) або кластером Kafka
- Споживач (consumer) може підписатися на одну або кілька тем і використовувати повідомлення, забираючи дані від брокерів

На рисунку 2 зображений приклад архітектури Apache Kafka з двома виробниками та трьома споживачами.

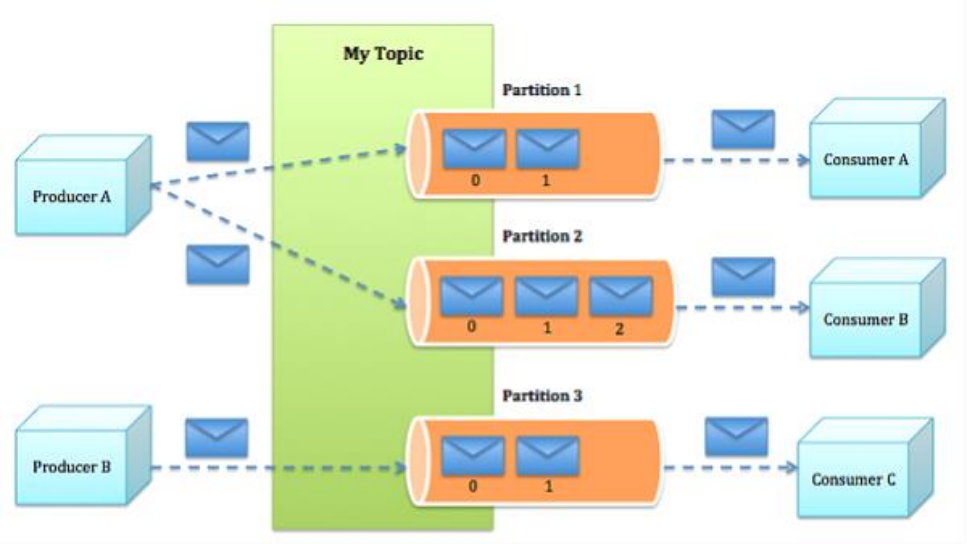

Рис. 2. Архітектура Apache Kafka [2]

Оскільки Kafka за своєю природою є розподіленою системою, кластер складається з декількох брокерів. Для зручності тема розбивається на секції, і кожен брокер відповідає за зберігання однієї або декількох секцій. Це дає можливість безлічі виробників і споживачів публікувати і використовувати повідомлення для своїх цілей одночасно [2].

Обсяг споживаних даних визначається не брокером, а споживачем. Брокер не володіє жодною інформацією щодо того, чи прийняв споживач повідомлення чи ні, повідомлення буде видалене автоматично, якщо воно затримується у брокера довше визначеного часу. При цьому споживач може в будь-який момент зробити «повторне замовлення» на ту чи іншу повідомлення.

По-друге, основною проблемою розподільних систем є неможливість визначити в будь-який момент часу, який сервер активний, а який ні. Тому з'являються такі проблеми як: безпека даних, відмови системи та інші «слабкі місця» розподілених систем. В Kafka повністю

інтегрована з ZooKeeper, який вирішує наступні проблеми: знажує ризки і дбає про безпеку та відновлення вісля відмов.

Kafka є інноваційною система для обробки великих обсягів даних. Її архітектура дозволяє споживачам самим регулювати швидкість, з якою вони будуть отримувати дані. При цьому, якщо виникне відмова системи або виняткова ситуація, споживач завжди має можливість отримати повідомлення повторно. Інтеграція з ZooKeeper дозволяє системі працювати не тільки швидко і злагоджено, але безпечно, що особливо важливо в разі великих даних - адже великі дані пов'язані з великими ризиками. Таким чином, Kafka дуже добре підходить для вимог, великий продуктивності, низького використання ресурсів [2].

В Таблиці 1 наведені порівняння основних брокерів повідомлень, таких як Apache Kafka та RabbitMQ.

Таблиця 1

Порівняння Apache Kafka та RabbitMQ

Характеристика	**Apache Kafka**	**RabbitMQ**
Пропускна здатність (повідомлення/секунду)	89000	20000
Маштабування	Горизонтальне	Горизонтальне
Впорядкування даних	Присутне	Присутне
Відказостійкість	Декілка кластерів	Декілка кластерів
Протоколи, які підтримуються	Свій власний протокол	STOMP, AMQP
Мови програмування	Python, .Net, Java, C++, PHP, Ruby	Python, Java, Go, Ruby, JavaScript, Swift, Elexir, C#, PHP, Objective-C

Висновки. В даній роботі було проаналізовано найпопулярніші брокери повідомлень для реалізації гарантованої доставки повідомлень в сервіс-орієнтованій архітектурі. Визначено їх переваги та недоліки. Та проведено аналіз за різними параметрами.

Література

1. Лінев Ф. А. Обзор систем обмена сообщениями / Лінев Ф. А // Молодий вчений. – 2017. – №19. – С. 29-32. – Режим доступу: https://moluch.ru/archive/153/43351/. – Дата доступу: 09.05.2018
2. Muhammad Ahsan. Message Broker Comparison – RabbitMQ, Kafka, ActiveMQ – 2017. – Режим доступа: http://teckhike.com/message-broker-comparison-rabbitmq-kafka-activemq-kestrel/. – Дата доступу: 12.05.2018
3. Офіційний сайт компанії RabbitMQ. – Режим доступу: https://www.rabbitmq.com. – Дата доступу: 18.05.2018
4. Офіційний сайт компанії Apache Kafka. – Режим доступу: https://kafka.apache.org. – Дата доступу: 18.05.2018