границы и снимали ограничения на передвижение, это не привело к восстановлению прежнего уровня туристических потоков. Как отмечается в исследовании [3], вакцинация и медицинские препараты, к сожалению, «... дают небольшую надежду на восстановление экономики в краткосрочной перспективе, особенно что касается путешествий и туризма».

По нашему мнению, предметом дальнейших научных исследований могут выступать обоснования цифровой трансформации процессов функционирования сферы гостеприимства, как одного из направлений ее стабилизации в кризисных условиях и условиях пандемий.

Библиографические ссылки

1. Ворошилова Г. О. Розвиток міжнародного ринку туристичних послуг в умовах кризи // Науковий вісник Херсонського державного університету. Серія: Економічні науки. 2014. Вип. 7. С. 75–78.

2. Зануда А. Коронавірус: скільки втрачає туризм. URL: https://www.bbc.com/ukrainian/features-51870285.

3. Оболенцева Л. В., Галицька А. М. Оцінювання впливу туризму як економічного явища // Сучасні тенденції розвитку індустрії туризму та гостинності у конкурентному середовищі: матеріали міжнародної науково-практичної конференції. Харківський національний університет міськог господарства імені О. М. Бекетова. 2020. 232 с.

4. Пандемія COVID-19 та її наслідки у сфері туризму в Україні (Оновлення до документа «дорожня карта конкурентоспроможного розвитку сфери туризму в України»). URL: https://ntoukraine.org (дата обращения: 21.03.2021).

5. Світлична В. Ю. Туристична сфера: пошук шляхів подолання кризи в умовах глобальних карантинних обмежень // Комунальне господарство міст. 2020. Том 5. Вип. 158. С. 24–31.

УДК 004.031.4

ПРОБЛЕМЫ СОГЛАСОВАННОСТИ ДАННЫХ В МИКРОСЕРВИСНОЙ АРХИТЕКТУРЕ

Т. С. Лобанкова[1)], А. А. Лобанков[2)] (*научный руководитель*)

[1)] *магистр, Самарский университет, Самара, Российская Федерация, lobankova_tatiana@mail.ru*

[2)] *доцент, Самарский университет, Самара, Российская Федерация, mart1989@mail.ru*

В настоящее время все больше компаний стремятся внедрить микросервисную архитектуру в разрабатываемые приложения. Она дает много преимуществ: отказоустойчивость, единоличная ответственность, масштабируемость и другие. Однако, она же и создает проблемы: усложнение зависимостей между модулями, обслуживания и т. д. Но самой большой проблемой является обеспечение согласованности данных. В этой статье мы расскажем какие существуют подходы для обеспечения согласованности данных в микросервисной архитектуре, рассмотрим их достоинства и недостатки.

Ключевые слова: микросервисная архитектура; транзакция; двухфазная фиксация; согласованность и компенсация

PROBLEMS OF DATA CONSISTENCY IN MICROSERVICES ARCHITECTURE

T. S. Lobankova[1)], A. A. Lobankov[2)] (*supervisor*)

[1)] *Master, Samara University, Samara, Russian Federation, lobankova_tatiana@mail.ru*

[2)] *Associate Professor, Samara University, Samara, Russian Federation, mart1989@mail.ru*

Nowadays, many companies are striving to implement microservices architecture in their applications. Of course, it provides many benefits: resiliency, sole responsibility, scalability, and others. However, it also creates problems the complication of dependencies between modules, maintenance, etc. The biggest challenge is ensuring data consistency. In this article, we will describe what approaches to ensure data consistency in a microservices architecture, consider their advantages and disadvantages.

Keywords: microservices architecture; transaction; two-phase commit; consistency and compensation

Микросервисы – это архитектурный стиль в виде набора небольших сервисов, каждый из которых работает в отдельном процессе и взаимодействует с другими посредством некоего легковесного механизма. Обычно это http, брокеры сообщения, сокеты. Каждая служба соответствует какой-то отдельной части бизнес-логики приложения и может быть развернута независимо от других [1]. В реальных задачах существует минимальный уровень централизованного управления между этими службами.

Микросервисная архитектура – это противоположность монолитной архитектуре приложений, некая распределенная система. Проблема согласованности данных в распределенной системе всегда стояла очень остро. Согласованность данных – это их непротиворечивость и целостность. В реляционных базах данных согласованность достигается за счет транзакций. Транзакция должна обладать четырьмя основными свойствами ACID (Atomicity + Consistency + Isolation + Durability): атомарность, согласованность, изолированность и надежность. Под согласованностью данных в микросервисной архитектуре мы будем понимать реализацию транзакции в распределенной среде.

Почему же это настолько сложно реализовать? Микросервисная архитектура предполагает, что каждый отдельный микросервис будет использовать свою отдельную базу данных [1]. С одной стороны, это позволяет выбирать подходящую под нагрузку на сервис технологию хранения данных, что является одним из преимуществ использования микросервисной архитектуры. С другой, очень сильно усложняет согласованность данных. Ведь баз данных стало много, в каждой из них запускаются свои локальные транзакции. Таким образом, главная задача согласованности данных в микросервисной архитектуре – объединить эту кучу локальных транзакций в одну большую распределенную транзакцию.

Если в вашем микросервисном приложении необходимо реализовать распределенную транзакцию, первое что надо сделать – это еще раз взглянуть на архитектуру приложения, и подумать, нельзя ли переделать ее так, чтобы вообще обойтись без распределенных транзакций. Зачастую это будет намного проще. Однако, существуют сложные сценарии, реализующие бизнес-процесс в нескольких микросервисах, где необходимо реализовать согласованность данных между ними.

Даже если мы все-таки решили оставить микросервисную архитектуру, первое что приходит в голову – это оптимистичный сценарий, когда мы считаем, что данные всегда согласованы и ничего для этого делать не надо. Этот вариант имеет право на жизнь в некритичных для бизнеса задачах, особенно если учесть, сколько усилий и времени потребуется для обеспечения согласованности.

На данный момент существует два способа реализации распределенных транзакций в микросервисной архитектуре:

1. Двухфазная фиксация (2PC, Two-Phase-Commit).
2. Согласованность в конечном счете и компенсация (шаблон проектирования Сага).

Давайте рассмотрим подробнее каждый вариант.

Идея алгоритма 2PC довольно проста – существует некоторый транзакционный менеджер, который оркестрирует транзакцию в два этапа. Первый этап – этап подготовки. Транзакционный менеджер уведомляет все диспетчеры ресурсов (базы данных, отдельные микросервисы, компоненты) о начале транзакции. Диспетчеры ресурсов выполняют локальные транзакции, но не фиксируют их. Если транзакционный менеджер получает сообщение об ошибке на первом этапе от какого-то диспетчера ресурсов, или какой-то диспетчер ресурсов не отвечает за заданный промежуток времени – транзакционный менеджер отправляет сообщения для отката локальных транзакций. В противном случае – сообщение фиксации [2].

Преимуществом данного подхода является простота и полное соответствие данного подхода характеристикам ACID строгой транзакции.

Но недостатки здесь более значимы. Во-первых, нарушается один из главных принципов микросервисной архитектуры – устойчивость к отказам, более того транзакционный менеджер является единой точкой отказа. Во-вторых, многие современные инструменты попросту не поддерживают протокол двухфазной фиксации (Nosql, Kafka, RabbitMq и другие). Главная мысль об использовании 2PC алгоритма точно и кратко сформулирована Крисом Ричардсоном: «Двухфазный фиксация – не вариант» [2].

Сага – это асинхронный шаблон проектирования, набор локальных транзакций. Если какая-то из них завершается неудачей, Сага запускает компенсирующую транзакцию. То есть нам нужно как-то управлять списком локальных и компенсирующих транзакций. Существует два способа управления [3]:

1. Хореография (Choreography) – каждая транзакция публикует события, которые запускают транзакции в других сервисах.
2. Оркестровка (Orchestration) – оркестратор говорит участникам, какие транзакции должны быть запущены.

Как отмечалось выше – для каждой положительной транзакции необходимо создать обратную (компенсирующую). Если какой-то шаг Саги завершается неудачно не стоит торопиться и вызывать компенсирующую транзакцию. Быть может просто отвалилось сетевое соединение или возникла какая-то другая проблема. Для начала надо попробовать повторно запустить данный шаг, и только после нескольких повторных неудач стоит вызывать компенсирующие транзакции.

Очевидно, что цепочка вызова локальных транзакций может быть велика, и если мы сломались ближе к концу, то должна быть вызвана и соответствующая цепочка компенсирующих транзакций. И в этом кроется одна из главных проблем – чтение грязных данных. Так как каждая локальная транзакция фиксируется отдельно и неизвестно сколько времени пройдет до вызова финальной транзакции в цепочке, может случиться ситуация, когда мы прочитали данные, а потом они были изменены обратно компенсирующей транзакцией. Таким образом, сага позволяет добиться только ACD- модели в терминах ACID, т. е. мы потеряли изолированность. Существует ряд паттернов, позволяющих исправить это: повторное чтение значения, семантическая блокировка и другие. Но их использование приводит к очередному усложнению реализации согласованности данных.

К плюсам данной технологии относится децентрализация, уменьшение количества узких мест, благодаря одно ранговой связи между службами, возможность ручного управления транзакциями (долгие транзакции можно вынести в отдельный микросервис).

Таким образом мы проанализировали доступные способы реализации согласованности данных в микросервисной архитектуре. Откуда можно сделать вывод, что на данный момент не существует однозначно хорошего решения. Каждое решение имеет существенные недостатки, для обхода которых приходится жертвовать консистентностью данных в областях не важных для решения бизнес задач.

Библиографические ссылки

1. Microservices a definition of this new architectural term : [Электронный ресурс]. URL: https://martin-fowler.com/articles/microservices.html (дата доступа: 10.10.2021).
2. Richardson Ch. Microservices Patterns with examples on Java // Manning Publications. 2019. P. 520.
3. Dürr K., Lichtenthäler R., Wirtz G. An evaluation of saga pattern implementation technologies // CEUR Workshop Proceedings. 2021. V. 2839. P. 74–82.

УДК 008.2

ОБ АКТУАЛЬНОСТИ ЦИФРОВИЗАЦИИ КУЛЬТУРНОГО НАСЛЕДИЯ В МИРЕ И В КИТАЕ

Лю И[1], Б. Н. Паньшин[2] (*научный руководитель*)

[1] *аспирант, Белорусский государственный университет, Минск, Республика Беларусь, 1006537911@qq.com*

[2] *доктор технических наук, профессор, Белорусский государственный университет, Минск, Республика Беларусь, panshin@tut.by*

Рассмотрены вопросы экономической и социальной значимости цифровизации культуры и опыт Китая по цифровизации культурного наследия и совершенствования культурного контента в Сети. Отмечено влияние широко применения современных цифровых технологий культурно актуальность повышения уровня цифровой культуры для в создании и распространении культурного контента.

Ключевые слова: цифровизация культуры в Китае; цифровые технологии в создании культурного контента.

ON THE RELEVANCE OF THE DIGITALIZATION OF CULTURAL HERITAGE IN THE WORLD AND IN CHINA

Liu Yi[1], B. N. Panshin[2] (*supervisor*)

[1] *PhD Student, Belarusian State University, Minsk, Republic of Belarus, 1006537911@qq.com*

[2] *Doctor of Technical Sciences, Professor, Belarusian State University, Minsk, Republic of Belarus, panshin@tut.by*

The issues of the economic and social significance of the digitalization of culture and the experience of China in the digitalization of cultural heritage and the improvement of cultural content on the Web are considered. The influence of the widespread use of modern digital technologies on the cultural relevance of raising the level of digital culture for the creation and dissemination of cultural content is noted.

Keywords: digitalization of culture in China; digital technologies in the creation of cultural content.

В настоящее время творческий сектор всех стран вносит около 3 % в мировой ВВП, обеспечивая около 29 миллионов рабочих мест во всем мире. Поэтому цифровизация