Сервис Аудита

Open API

Exported on 07/26/2022

Table of Contents

1 1. Требования к ФП....3
1.1 Исходное состояние / Описание проблемы....3
1.2 Цель....3
1.3 Задачи....3
1.4 Функциональные требования....3
1.5 Нефункциональные требования....4
2 2. Предметная область....5
2.1 Смысловое ядро....5
2.2 Сущности....5
2.3 Объекты-значение....6
2.4 Сервисы домена....6
3 Архитектура решения....7

1 1. Требования к ФП

1.1 Исходное состояние / Описание проблемы

Существует большое кол-во компонентов некой системы которые взаимодействуют между собой по http. Так же существует система аудита в которую каждый компонент отдельно уже отсылает или должен отсылать события аудита. Логика формирования операции аудита децентрализована и в случае изменения требует единовременной доработки всех компонентов что дорого, долго и может приводить к ошибкам. Предлагается вынести логику формирования событий в отдельную функциональную подсистему которая бы на основе логов с обязательным набором данных от каждого компонента формировала бы операцию аудита для заранее известных последовательностей вызовов в системе, представленных в виде карты событий аудита.

1.2 Цель

Целью разрабатываемой ФП является вынесение из основных компонентов системы логики обработки событий аудита и формирования операции аудита в формате целевой системы аудита с последующей отправкой в последнюю и, как следствие, снижение структурной, логической и вычислительной сложности основных компонентов системы, а так же сосредоточение логики формирования операций аудита в целевом виде в одной ФП.

1.3 Задачи

Разрабатываемая система должна **принимать** журнальные сообщения основных компонентов системы в их нативном формате, **валидировать** на наличие обязательного набора данных , **балансировать** входящие сообщения по уникальному идентификатору запроса (rquid), определять **цепочки вызовов** на основе **карты событий аудита**, их соответствие **критериям завершенности**, формировать на основе транзакции **операцию аудита** и отправлять в целевую систему аудита.

1.4 Функциональные требования

Дано сервис аудита поднимается
И загружает в память актуальную карту аудита
И начинает слушать события из брокера сообщений
А также запускает планировщик (обходчик) для периодического определения завершенных транзакций

Когда сервис аудита получает поток событий из брокера сообщений
Тогда он фильтрует события согласно вхождению в карту аудита
И трансформирует их к общему виду для последующей единообразной обработки
И группирует их по уникальному идентификатору запроса (rquid) в цепочки событий
А планировщик периодически обходит полученные цепочки и определяет их завершенность по заранее известным критериям

Когда цепочка завершена
Тогда формируется операция аудита и отправляется в брокер сообщений системы аудита

1.5 Нефункциональные требования

Сервис должен обрабатывать поток событий максимально быстро. Для достижения этого предлагается хранить цепочки событий в оперативной памяти распараллелив их обработку путем разбиения их на небольшие группы.

Для достижения этого предлагается разбить сервис аудита на два микро сервиса (см. дальнейшее описание и архитектурную схему ниже)

1. Поддерживать получение сообщений из Kafka
2. Поддерживать отправку сообщений в Kafka
3. alf-audit-filtering-distributor хранит в памяти карту событий аудита
4. alf-audit-filtering-distributor трансформирует полученное сообщение в форму пригодную для фильтрации
5. alf-audit-filtering-distributor фильтрует сообщения полученные из топиков под компоненты системы согласно карте событий аудита
6. alf-audit-filtering-distributor приводит (трансформирует) отфильтрованное сообщение от компонента системы к событию аудита. В терминах предметной области производить маппинг Event к AuditEvent
7. alf-audit-filtering-distributor балансирует сообщения по партициям топика rquid-bucket на основе остатка от деления последнего октета rquid преобразованного в число.
8. alf-audit-processor хранит в памяти карту событий аудита
9. alf-audit-processor формирует и хранит маппинг rquid (**цепочки событий**) на события полученные из топика rquid-bucket
10. alf-audit-processor производит периодический обход цепочек событий с целью определения критерия завершенности каждой транзакции.
11. При завершении транзакции alf-audit-processor формирует операцию аудита из данных транзакции с последующей отправкой в целевую систему аудита.

2 2. Предметная область

2.1 Смысловое ядро

Событие (Event) - некоторое изменение состояния какого-либо компонента системы произошедшее в рамках или согласно принятому бизнес-процессу.

Сообщение (Message) - предназначенная для обработки единица информации о событии поступающая из источника в разрабатываемую ФП по сети и отправляемая по сети в приёмник.

Источник (Source) - отправитель сообщений содержащих информацию о событии по каналу связи и предназначенных для получения разрабатываемой ФП.

Приёмник (Receiver) - получатель преобразованных сообщений содержащих информацию о событии по каналу связи от разрабатываемой ФП.

Цепочка событий (Event chain) - последовательность логически связанных сообщений о событиях.

RqUID, TID - уникальный идентификатор события.

Событие аудита - действие над бизнес-сущностью произощедшее в источнике ответственным за событие.

Корневое событие - событие произошедшее в компоненте инициализирующем некий бизнес-процесс

Терминирующее событие - ответ на корневое событие в компоненте инициализирующем некий бизнес-процесс

Карта событий аудита - таблица соответствия наименования события (действия), источника возникновения, HTTP-ресурса и метода корневого события в транзакции.

Критерии завершенности транзакции - оценочная мера успеха или неудачи транзакции

- успешная транзакция (логическое И) -
  - в цепочке найдено корневое событие соответствующее записи из карты событий аудита
  - в цепочке найдено терминирующее событие со статусом RESPONSE и кодом 2xx
  - не завершился таймаут жизни цепочки в рантайме
- неуспешная транзакция
  - завершился таймаут жизни цепочки в рантайме
  - в цепочке найдено терминирующее событие со статусом RESPONSE и кодом отличным от 2xx

Операция аудита - технически это объект заданной структуры который нужно заполнять правильными данными и отправлять в целевую систему аудита

2.2 Сущности

AuditEvent - событие аудита

AuditEventMap - карта событий аудита

AuditOperation - операция аудита

2.3 Объекты-значение

SourceMessage -

EventChain - цепочка событий разложенных по rquid

2.4 Сервисы домена

PreFilterMapper - маппер сообщений в форму пригодную для фильтрации

EventsFilter - фильтр сообщений полученных из кафки

EventToAuditMapper - маппер сообщений от компонента системы к событию аудита

BucketBalancer - балансировщик событий по топикам

ChainUpdater - обновитель цепочек

ChainCrawler - обходчик цепочек

AuditOperationFactory - фабрика для создания операций аудита

3 Архитектура решения