**Основной сценарий работы с докладчиком**

//www.plantuml.com/plantuml/png/lPLFRjD05CRtSugvG2zGAL45I2nG1H9Sm0ekYd3ZaDZxn4KKPAGAX8J48eKRf8rDJLmxLtXp8xxdYTMEfnRs1YHJsVFztlytEPe7pYm8JpqHZ0FFbRHIIogefElg\_pzIBTK7Il6oLWlAy05\_rdH5fOfKZ5D92NHRol4rKUUGRF5-AvnHuC-a\_aShG1i82XpVm584V26TkNm-UoF4516CH-EfCmaa\_G1TgFUK0e5RECyeO-bZVt8YXNWNkg6xj\_lCENdho8Q1Bm1oF0Lj67iqzZnJiA8TvmSFdwh4CdN1mcDdx4b3-VBurGjX9lAjcIkJJ\_oHgRg4gaGTKoIQuKafLvzP-JHqvu4GOZ2WNz3SKB8lIYff-r3YWu50YUJ1KAT\_AEad3ZUmkbGHTm8onDz22uOOreK2VSbD19-3\_5XdB\_JnmS5mM5MXLSU7R8uBqof2DLhYvH7IyHf3G-igJksVipqqwnopEcVe6kzeBIFx13tVd-f0T4SNeDQcTHm8nmkCZv9sLUo5kb1Hiy9VJIzsjAutToMbd1BQoWDDjtWzvrwolcdmUrCWUGBDnJapUzMLibhNaJDQsHbYiVR5JUwC\_TU9S2Ul-Nbmnwg5MTAEdTSHsnl6IMqHLwoYcZ9Npjoj6eCpdHA2oa6qkxIJiqVEvEovwavPWxMYAvoK\_mpJdgtkg35UtJbxmlgJ-fBjW\_AeeEdw0k6RhEULt9srUj4TlTyD\_ath1pC1ltTLtV4gyRHYXrHSZVOPVwolSKqOFnqmd82eFD9lIkvybhKv5TyFavFjGHvXRSDJxoy0

@startuml

title Основной сценарий работы с докладчиком

actor Организатор as Org

participant Конференции as Conf

queue Kafka

participant Билинг as Bill

participant Оповещения as Mail

participant SMTP

actor Докладчик as Doc

actor Поосетитель as Guest

==Создание конференции==

Org -> Conf : Регистрация конференции

Conf -> Kafka : Приглашения

Kafka -->> Mail: Приглашения

Mail -> SMTP: Приглашения

Doc -> Conf : Регистарция доклада

Conf -->> Org : Доклады для валидации

loop Проверка докладов

alt доклад принят

Org -> Conf : Валидация доклада

else есть замечания

Org -> Conf : Замечания по докладу

Conf -> Kafka: Замечания по докладу

Kafka -->> Mail: Замечания по докладу

Mail -> SMTP: Замечания по докладу

SMTP -->> Doc : Замечания по докладу

Doc -> Conf : Регистарция доклада

Conf -->> Org : Доклады для валидации

end

end

==Создание расписания==

Org -> Conf : Закрытие регистрации

Conf -> Conf : Составление расписания

Conf -->> Org : Валидация расписания

Org -> Conf : Валидация расписания

Conf -> Kafka: Обновление расписания

Kafka -->> Mail: Обновление расписания

Mail -> SMTP: Обновление расписания

SMTP -->> Doc : Обновление расписания

SMTP -->> Guest: Обновление расписания

==Завершение конференции==

Org -> Conf : Завершение конференции

Conf -> Kafka: Завершение конференции

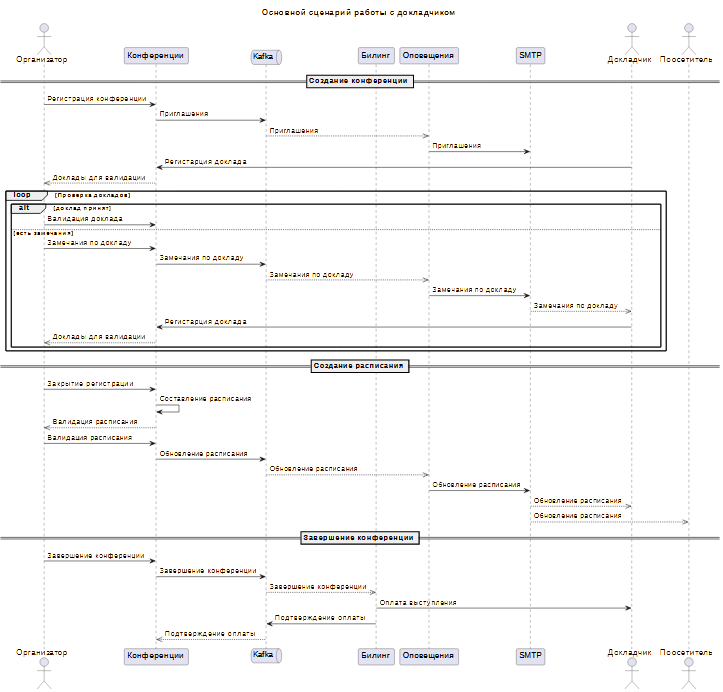
Kafka -->> Bill: Завершение конференции

Bill -> Doc : Оплата выступления

Bill -> Kafka : Подтверждение оплаты

Kafka -->> Conf : Подтверждение оплаты

@enduml

 **Оценка атрибутов качества**

В данном решении основная логика сосредоточена в сервисе «Конференции». Это statefull сервис.

По требованиям нагрузка на данный сервис может вырасти в разы. Соответственно нагрзука будет расти как на бэкенд часть так и на БД.

Исходно мы приняли решение о размещении наших сервисов в облачной инфрастурктуре. Поставщик обеспечивает надежность на уровне Tier2. Данный уровень вполне устраивает наше решение.

Остается обеспечить безотказную работу сервисов.

**Масштабируемость**

* Развертывание среды контейнеризации OKD (решение принято ранее) обеспечивает горизонтальную масштабируемость.
* Для обеспечения масштабируемости необходимо подготовить бэкенд сервис для работы в нескольких подах сервиса с одной БД. Необходим autoscaling в deployment для нивелирования взрывного кратковременного роста нагрузки. Минимальное кол-во подов каждого сервиса 2 шт.
* Для распределения нагрузки между подами необходимо использовать балансировщик запросов active-active c healthcheck’ами.
* Также необходимо использовать распределенный кэш данных между подами.
* Необходимо создать отдельные БД для архивирования информации о завершенных конференциях.
* При дальнейшем росте нагрузки возможно шардирование БД для выделения крупных организаторов в отдельные инстансы.

**Отказоустойчивость**

Вероятность сбоя каждого отдельного сервиса достаточно высока. Для обеспечения отказоустойчивости, помимо автоматического горизонтального масштабирования каждого сервиса и балансировки active-active описанного разделе «масштабируемость» необходимо применить паттерны:

* Circuit Breaker -позволит сервису восстанавливать свою работоспособность при получении большого числа запросов.
* Throttling (rate-limit / quote) — не даст привести сервис в неработоспособное состояние.
* Load Shedding — ограничит нагрузку на каждый инстанс. Важно реализовать readiness liveness пробы.

