Camunda

Подводные камни и особенности применения

Алексей Коняев

@alexeykonyaev

Разработчик в @tinkoff_bank

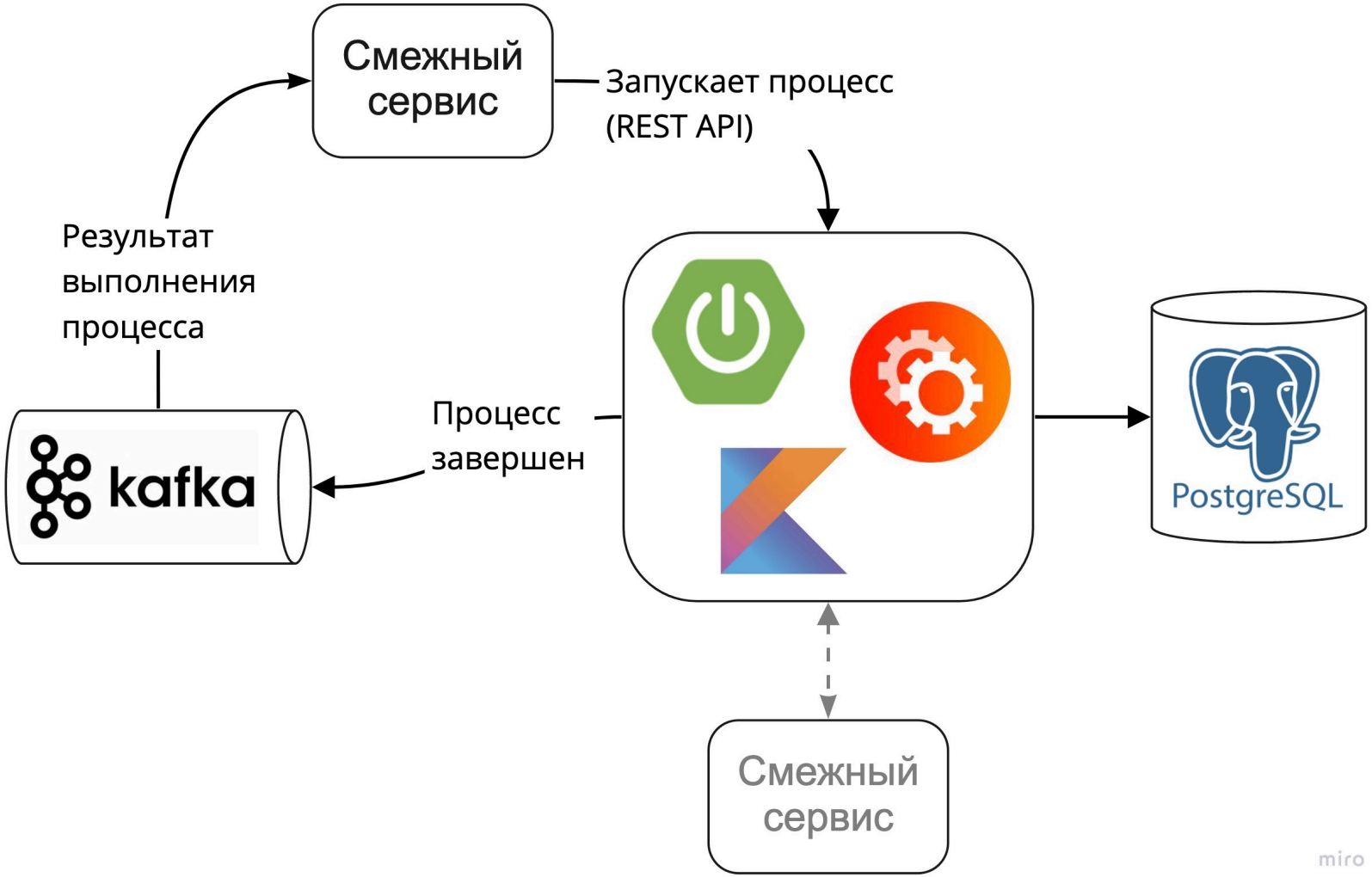
тинькофф

- Движок бизнес-процессов BPMN 2.0 + DMN
- JVM-библиотека + БД
- Интеграция со Spring-ом
- Богатый REST API
- Open Source (Activiti fork 2013)

Применение в Тинькофф

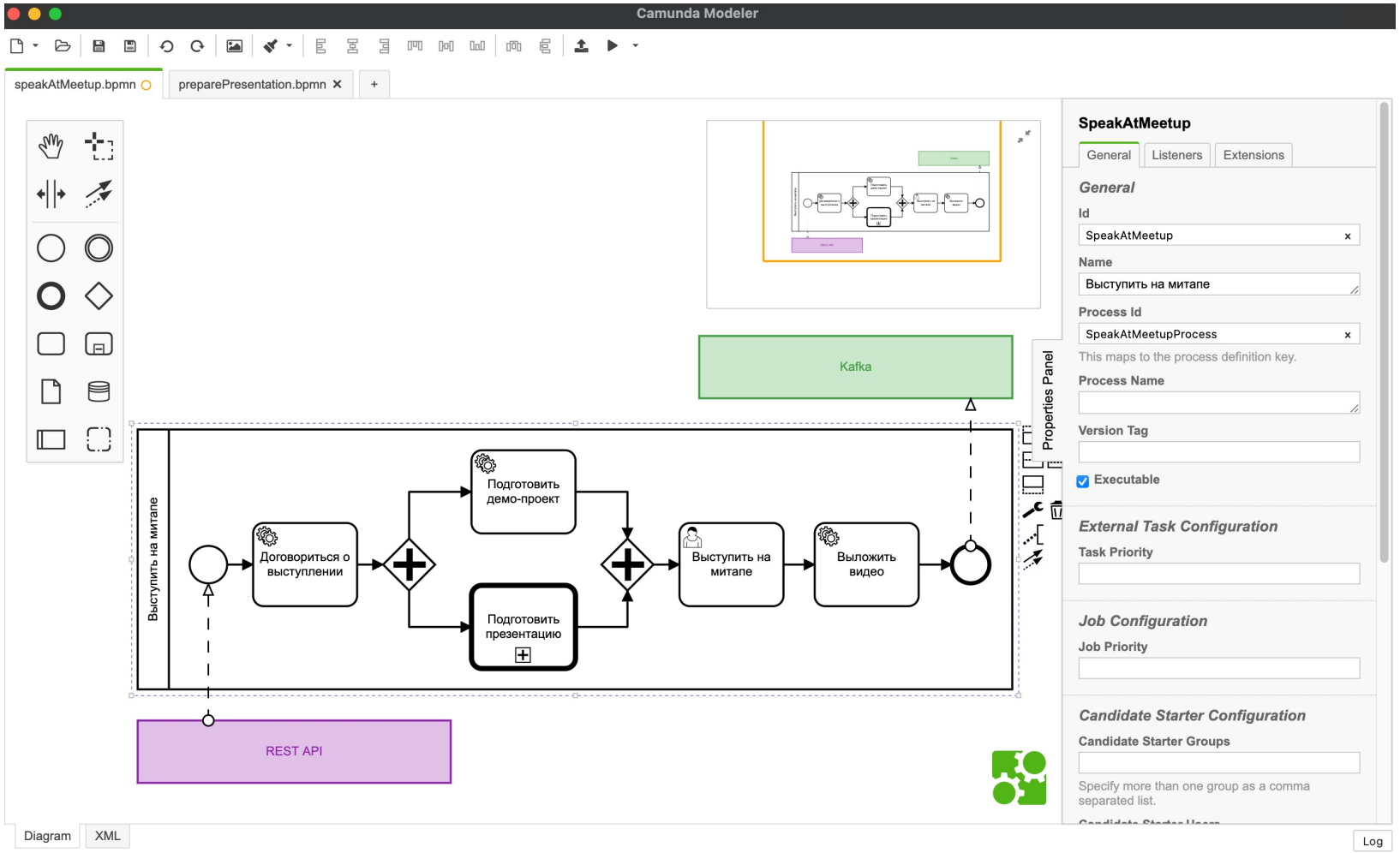
- Почти 3 года в эксплуатации
- Десятки сервисов
- Миллионы процессов
- Терабайты данных
- Пройдено достаточно граблей ☺

Архитектура сервиса



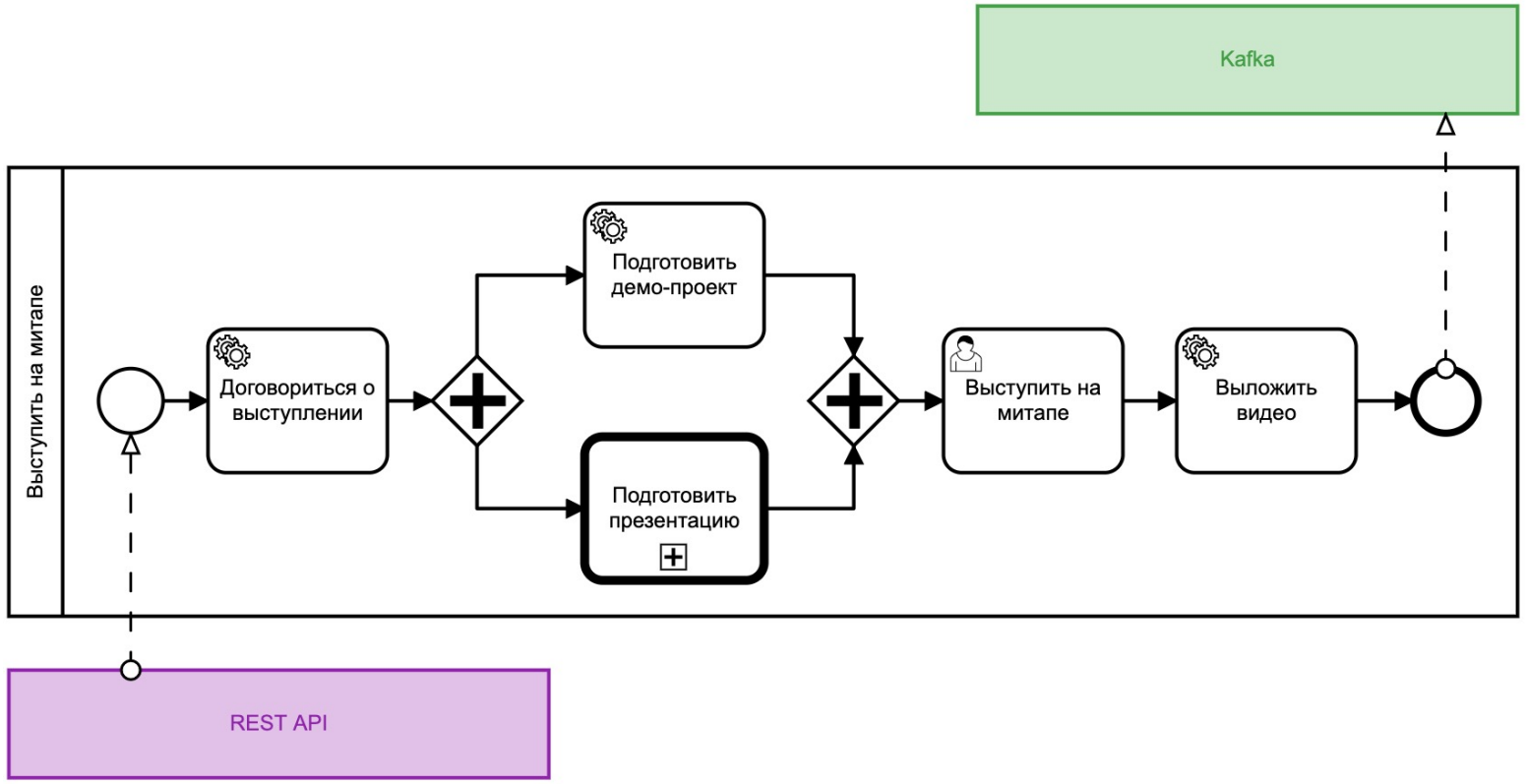
Подходы при разработке (схемы)

- Проектируем схемы в Camunda Modeler



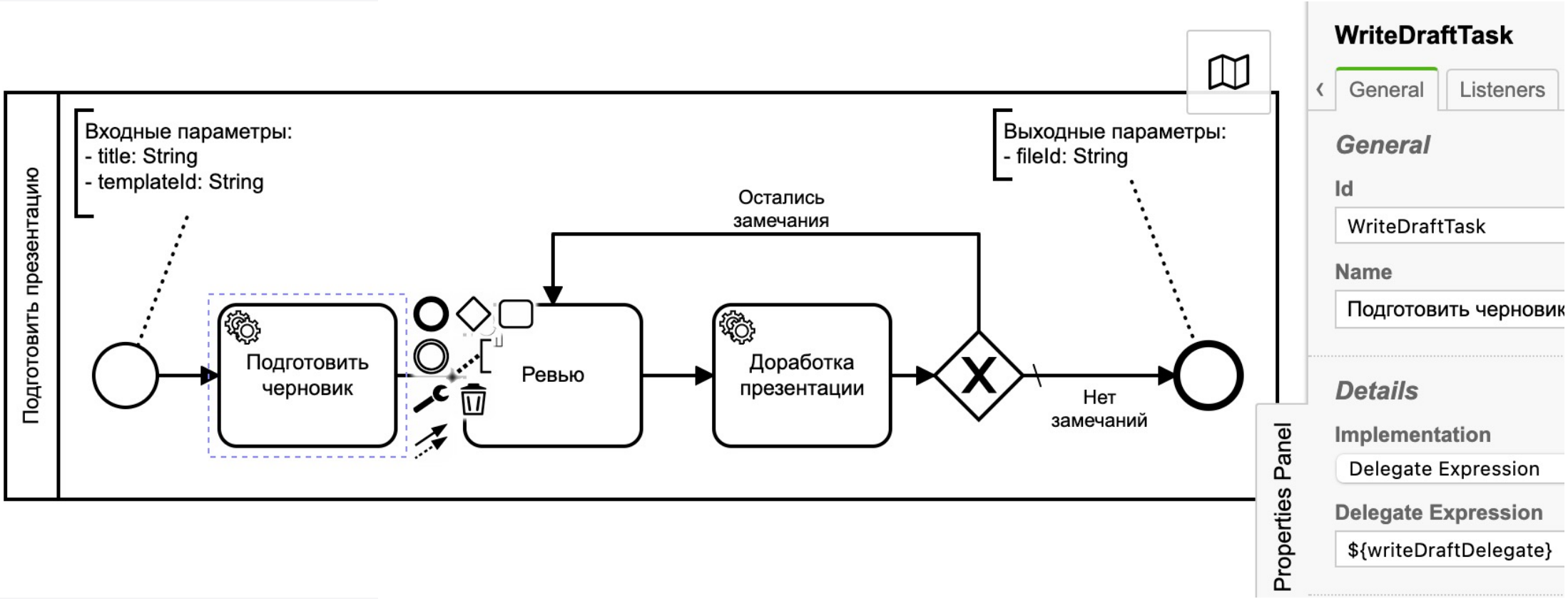
Подходы при разработке (схемы)

- Соблюдаем конвенцию:
  - Элементы внутри процесса – белые
  - Внешние элементы – цветные



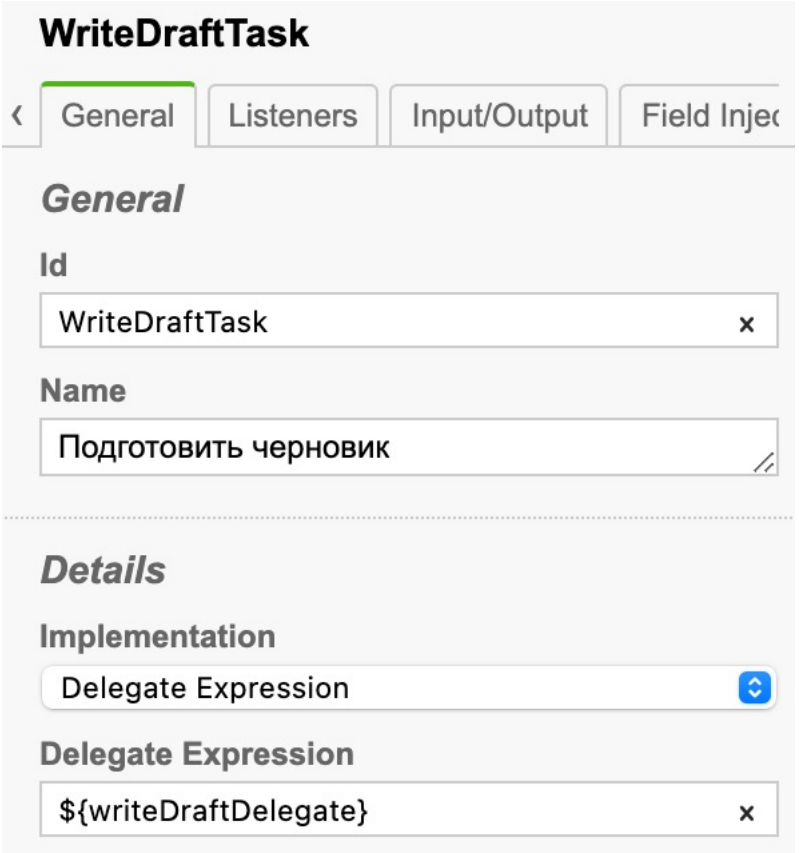
Подходы при разработке (схемы)

- Соблюдаем конвенцию:
  - Осмысленные ИД и имена
  - Комментарий для входных/выходных переменных
  - Ветвление только через Gateway-и



Подходы при разработке (схемы)

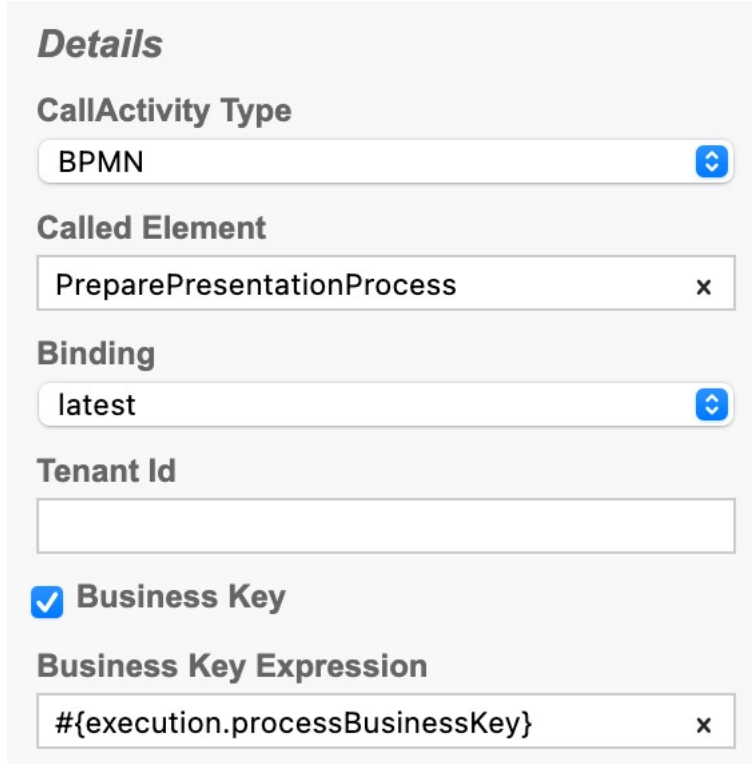
- Service Task - основной элемент
  - Delegate Expression = ссылка на Spring Bean



Подходы при разработке (схемы)

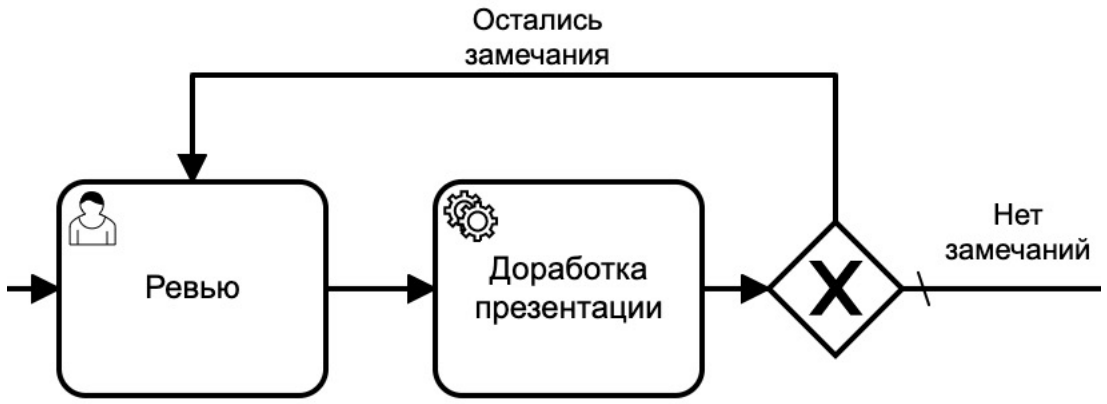
- Call Activity – вызов дочерних схем
  - Binding = latest
  - Business Key Expression =

    #{execution.processBusinessKey}



Подходы при разработке (схемы)

- Gateway – Condition type = Expression из переменных

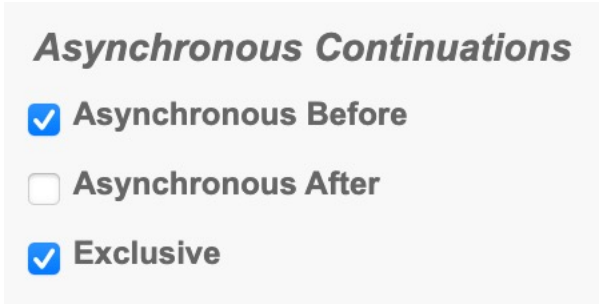


Подходы при разработке (схемы)

- Для Task-ок, перед выполнением которых важно закомитить состояние процесса

  Asynchronous Continuations:

  - Asynchronous Before = true
  - Exclusive = true

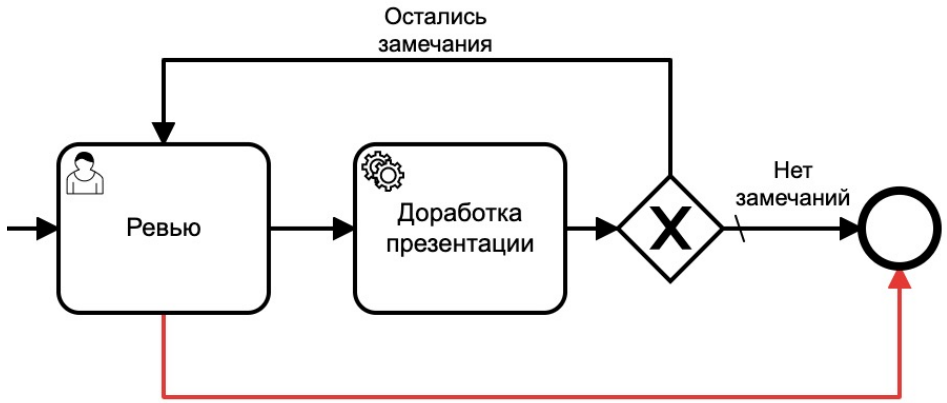


Подходы при разработке (код)

- Делегат – компонент, реализующий JavaDelegate
- Переменные контекста
  - Примитивные типы (по возможности)
  - Serializable + фиксированный serialVersionUID
- Исключения в делегатах
  - Ошибка бизнес-логики – BpmnError
  - Ошибка приложения - RuntimeException

Подходы при разработке (тестирование)

- Валидация схем – bpmn-io/bpmnlint




```
/Users/a.konyaev/others/camunda-demo/src/main/resources/bpmn/preparePresentation.bpmn
  DoReviewTask  warning  Incoming flows do not join  fake-join
  EndEvent      warning  Incoming flows do not join  fake-join
  DoReviewTask  error    Flow splits implicitly      no-implicit-split

✖ 3 problems (1 error, 2 warnings)
```

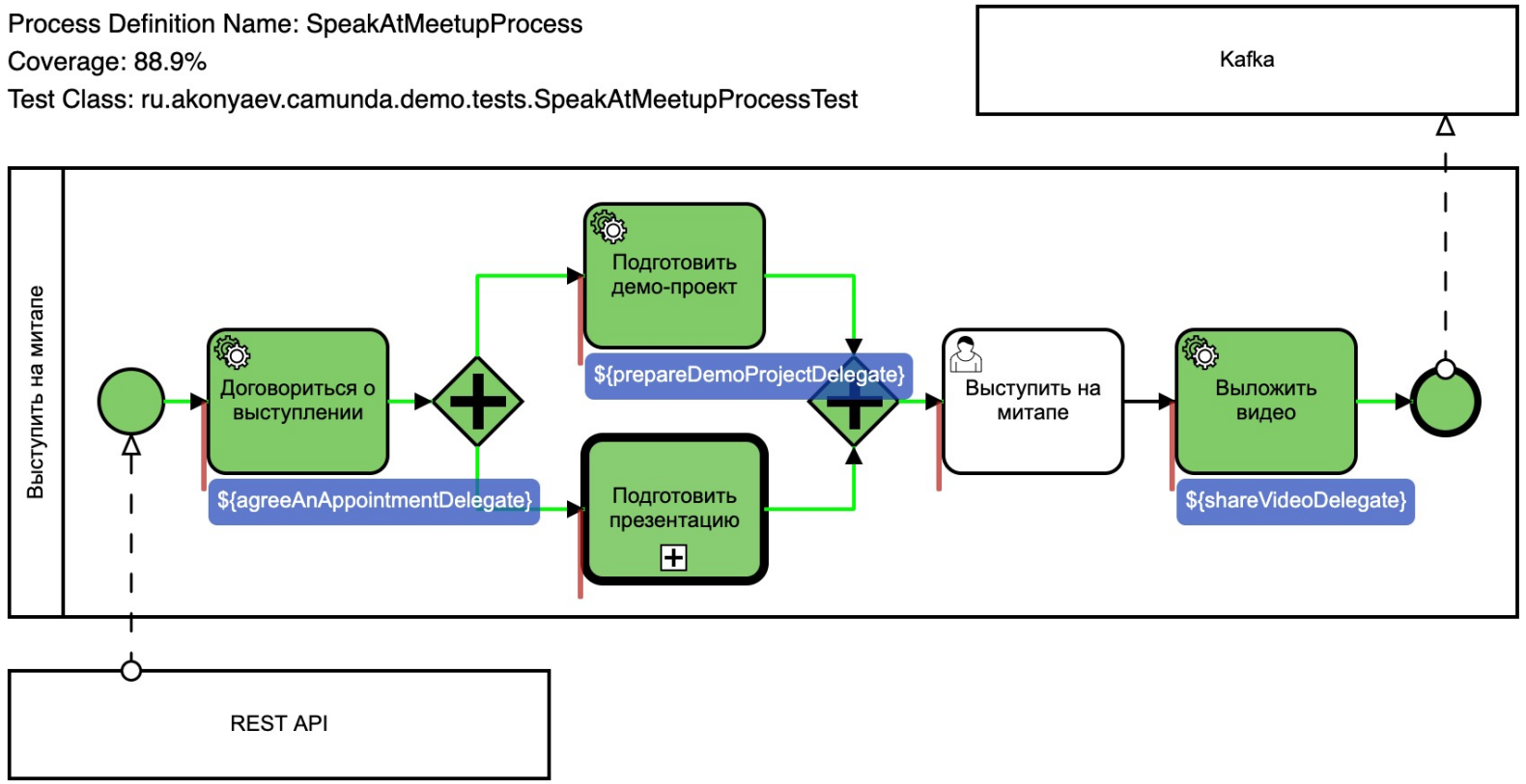
Подходы при разработке (тестирование)

- Юнит-тесты делегатов с DelegateExecutionFake
- Процессные тесты
  - Подключение схем через @Deployment
  - Проход по всем шагам и веткам процесса
  - Проверка наличия и значений переменных
  - Запуск вложенных процессов
  - Корреляция событий

Подходы при разработке (тестирование)

- Процессные тесты
  - Отчет о покрытии тестами

Process Definition Name: SpeakAtMeetupProcess
Coverage: 88.9%
Test Class: ru.akonyaev.camunda.demo.tests.SpeakAtMeetupProcessTest



Отладка и развертывание

- Запуск в Docker/IDE для отладки
- Развертывание в k8s для QA/Production
  - Если нужно – несколько реплик
- Метрики
- Мониторинг процессов
  - Встроенный Camunda Cockpit
  - EX-CAM-AD

Демо

Подводные камни и особенности

Длина и вложенность процессов

- Большие процессы не удобны в поддержке
- Большая вложенность процессов увеличивает сложность
- Трудно распараллелить разработку

- Кусок логики выделять в отдельный сервис со своей Camunda БД
- Размер процесса – не более 30 элементов
- Вложенность – не глубже 3

Внимание к переменным

- Неявно передаются из вышестоящей схемы
- Создаются при завершении вложенной схемы
- Создаются в результате выполнения DMN
- Создаются в делегатах
- Создаются при корреляции событий

- Документировать
- Явно передавать по одной, а не через ”all”
- Используем типизированные обвязки для переменных контекста
- Покрывать тестами

Учитывать старые экземпляры процессов

- Экземпляры предыдущих версий процессов все еще живы после обновления приложения!
- Старые схемы будут вызывать новые делегаты
- Вложенный процесс старой версии может не вернуть новых переменных

- Понимать тайминги выполнения процессов
- Новые переменные объявлять как Nullable
- На схеме проверять наличие переменной execution.hasVariable(“имя переменной”)

Сериализация переменных контекста

- Переменные примитивных типов хранятся нормально, все остальные – act_ge_bytearray
- Enum, OffsetDateTime, BigDecimal – и др. типы, похожие на примитивные, для Camunda сложные
- Сложные – сериализуются Java serialization-ом*
- Сложные типы (List, Map) – бывают необходимы

- Стараться использовать примитивные типы
- Если сложные – не забывать serialVersionUID
- Можно сделать «обертки», чтобы сохранять как String

Хранение истории

- История – хранит изменения метаданных, состояния процесса и его переменных
- История необходима для аналитики
- Сериализованные исторические переменные тоже хранятся в act_ge_bytearray!

- Если не нужна – отключить
- Сразу включить механизм очистки
- Удалять не нужные deployment-ы
- Секционирование исторических таблиц
- Хранить историю во внешней БД

Долгий старт приложения

- Причина – много данных в act_ge_bytearray
- При старте Camunda проверяет актуальность схем
- Запрос к таблице может сильно тормозить – десятки минут

- Меньше данных в act_ge_bytearray – лучше
- Тюнить БД

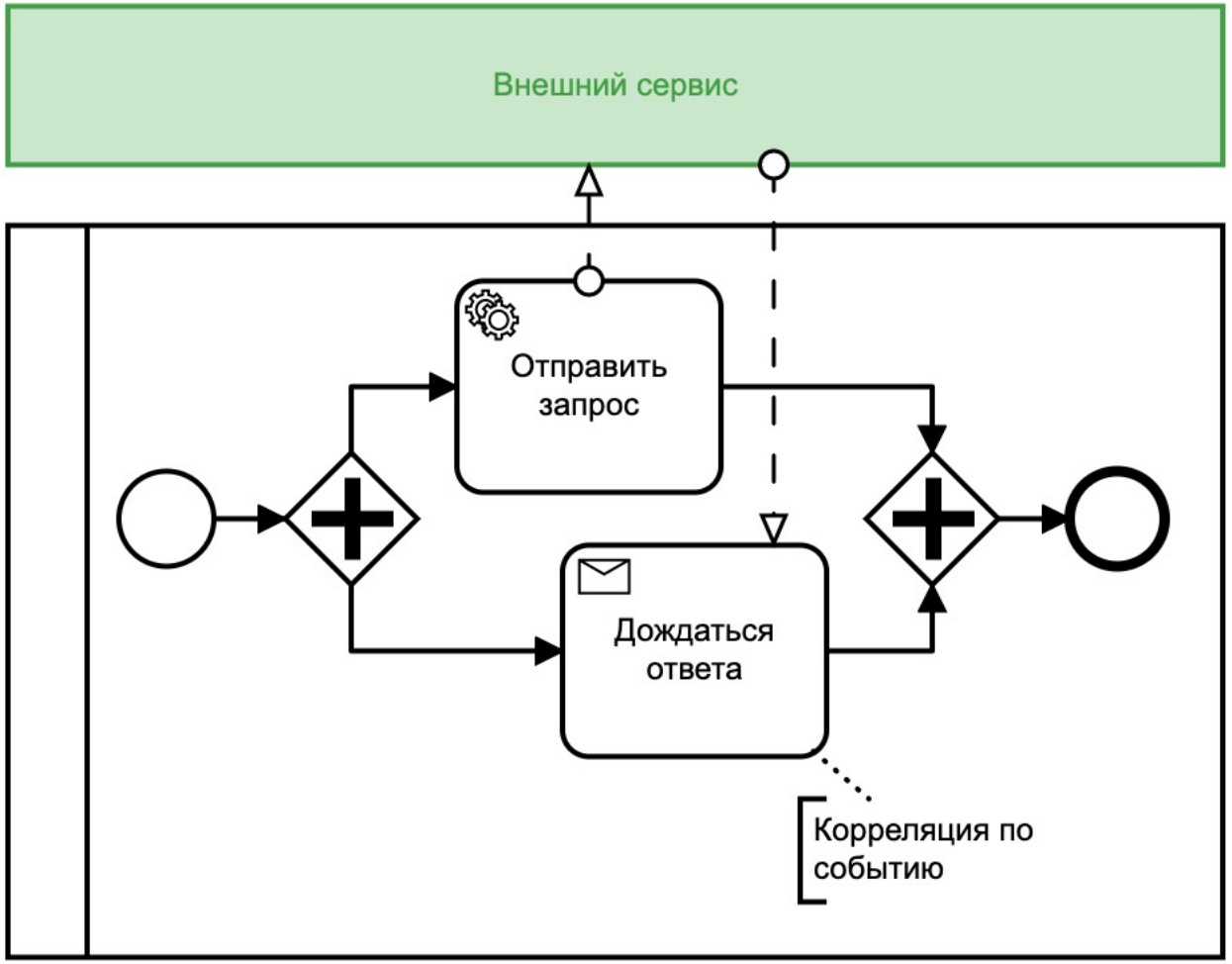
Корреляция событий

- Корреляция по переменным – может быть очень не эффективна
- Может выбросить OptimisticLockingException

- Коррелировать события по бизнес-ключу
- Добавлять индексы на act_ru_variables
- Коррелируем с ретраями

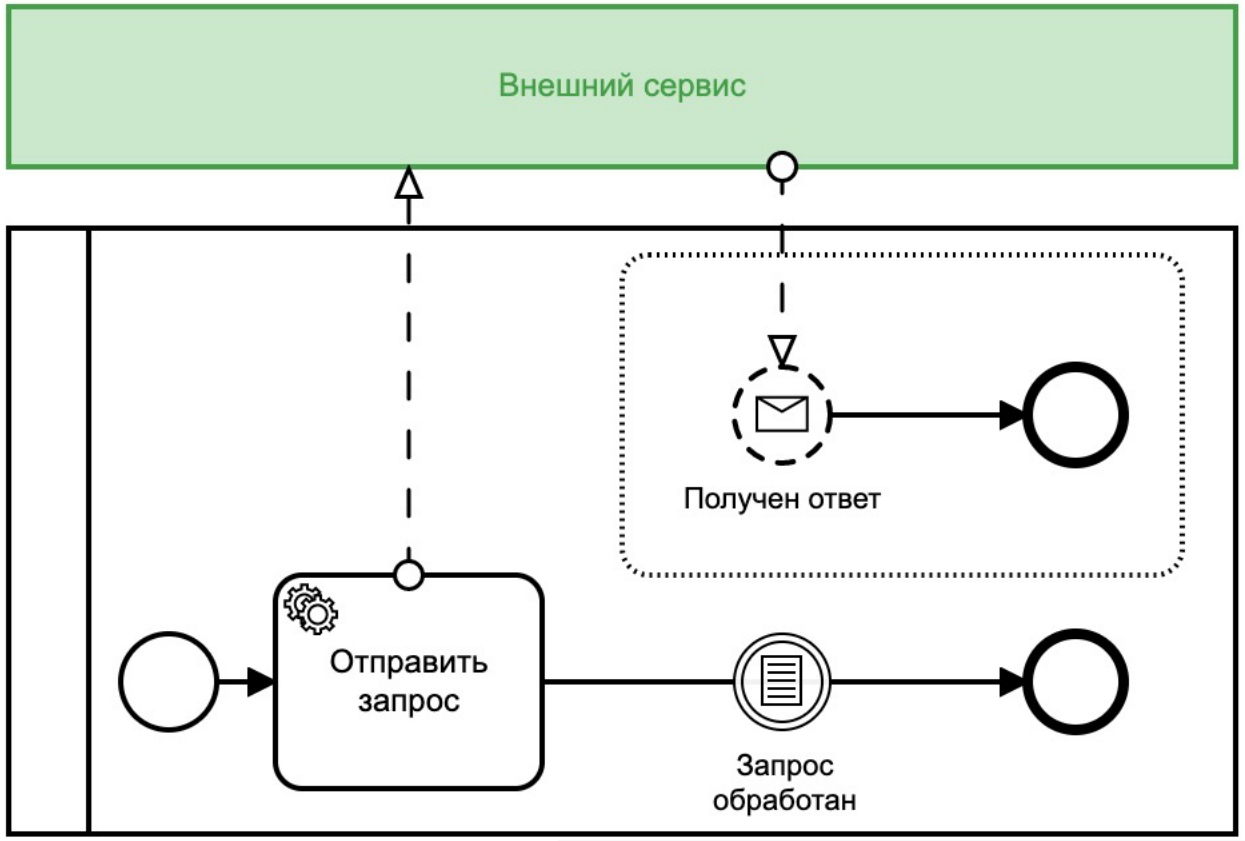
Асинхронное ожидание событий

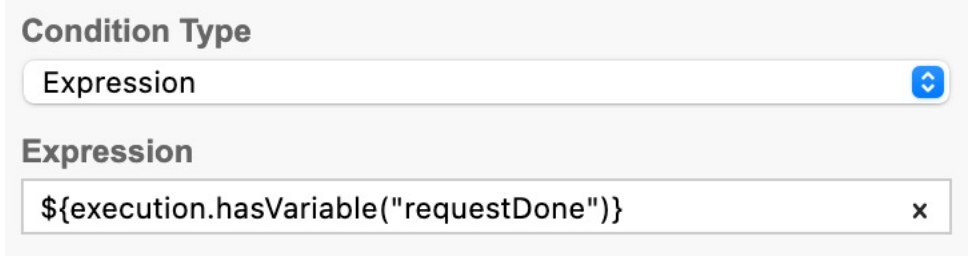
- Параллельный Gateway – не параллельный ☺



Асинхронное ожидание событий

- Асинхронный не прерывающий подпроцесс





Развертывание новой версии

- Нельзя обновляться, пока процесс выполняется

- Специальные схемы для постановки на паузу

- Camunda: вы старую схему поставили на паузу, но новую я все равно запущу!

- Выключить обновление не измененных схем
  - processEngineConfig.*isDeployChangedOnly* = true
  - <property name="isDeployChangedOnly">true</property>

Проблемы и альтернативы

- Сложно масштабировать – упираемся в БД
- Деградация со временем – если не удалять мусор

- Распределенная линейно-масштабируемая BPMN-платформа

Ссылки

- https://github.com/camunda
- https://github.com/camunda-cloud/zeebe
- https://github.com/KotskinKotskin/camunda-excamad
- https://bpmn2.ru/blog/top-25-oshibok-bpmn
- https://github.com/a-konyaev/camunda-demo