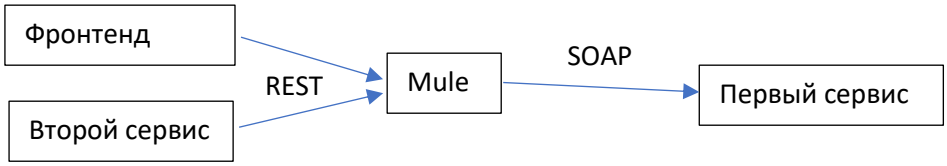
Туториал по 4 лабе СОА

Если у вас на SOAP **ВЫЗЫВАЕМЫЙ** сервис



1. Добавляете SOAP контроллер на вызываемый сервис (JAX-WS для Java EE, Spring WS для спринга), рестовый контроллер удалять пока не надо. По этому шагу есть куча примеров в интернете.

2. Скачиваете mule ESB, оформляете пробную версию на месяц

3. Заходите на https://anypoint.mulesoft.com/designcenter, регаетесь там, создаете новую спецификацию API, внимательно и полностью делаете спецификацию со всеми типами данных ошибками и так далее. Нужно делать очень внимательно и проверять, чтобы все типы данных соответствовали (например если это дробное число, нужно поставить тип number, а в advanced параметрах уточнить, что это float). Если на этом этапе сделать ошибки, mule дальше может не заработать.

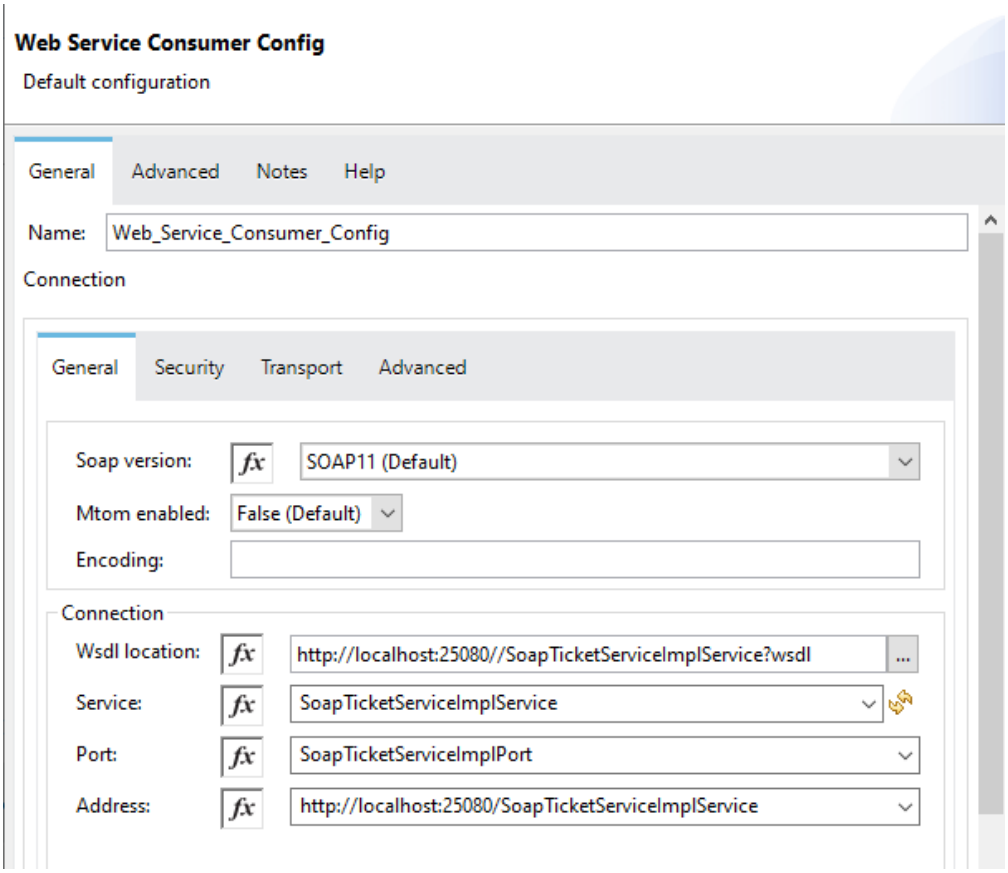
   Еще важно, чтобы на каждый URL был включен метод OPTIONS, это надо, чтобы CORS работал. Когда все сделали, публикуете спецификацию.

   Мой пример https://anypoint.mulesoft.com/exchange/7320a393-d3ee-413c-9a9a-eff41fe87eca/soa_lab4/minor/2.0/

4. Запускаете Mule, создаете новый проект. Во вкладке API Implementation выбираете Download RAML from Design Center. Вводите ссылку на свой API.

5. Mule создаст листенер и потоки на каждый запрос из спецификации. Заходите в конфиг листенера, ставите нужный порт, также можно настроить https.

6. В каждый поток добавляете web service consumer. Для него создаете новый конфиг, где указываете адрес первого сервиса и ссылку на его WSDL



После указания конфига ставите в каждом потоке нужную функцию

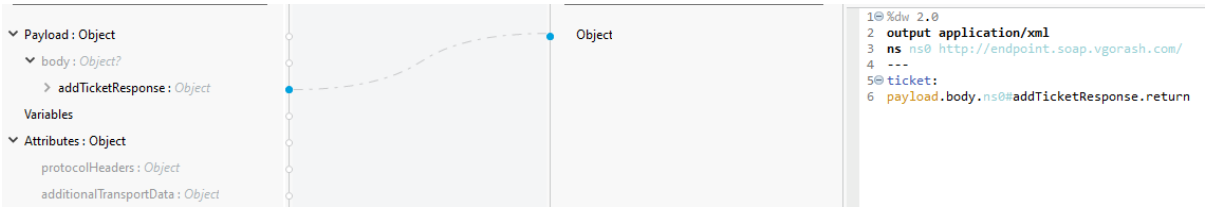
7. Перед каждым консьюмером ставите message transform. Настраиваете маппинг параметров на аргументы вызовов

ВАЖНО: если какие-то аргументы не обязательны, нужно дописать после output type **skipNullOn=”everywhere”**

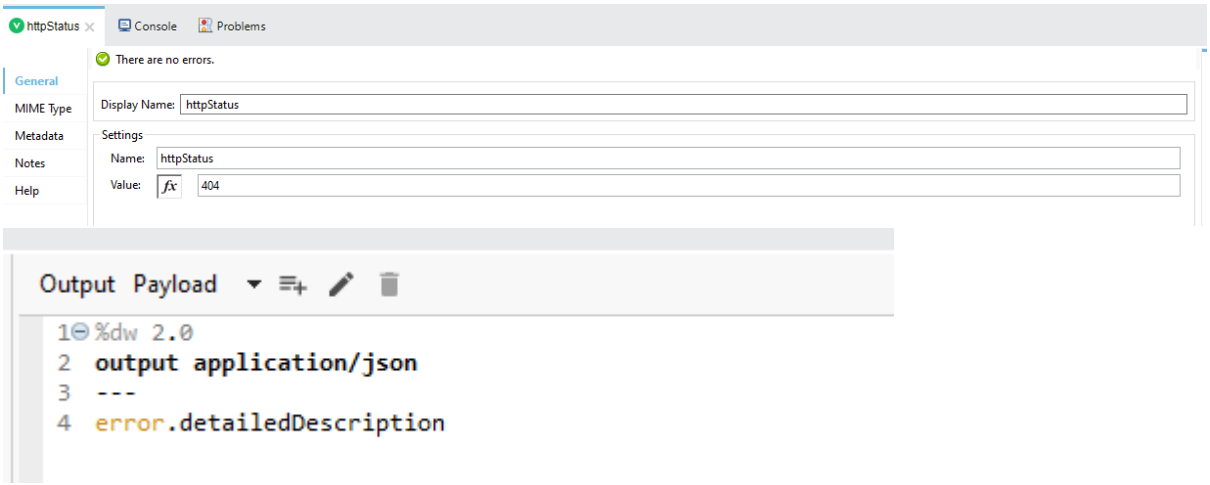
ЕЩЕ ВАЖНО: у меня был баг, когда я маппил параметр из request body, в аргумент передавалось пустое значение. Решение вот такое:

```
%dw 2.0
output application/xml
ns ns0 http://endpoint.soap.vgorash.com/
---
{
    ns0#addTicket: payload mapObject (v, k, i) -> {arg0: v}
}
```

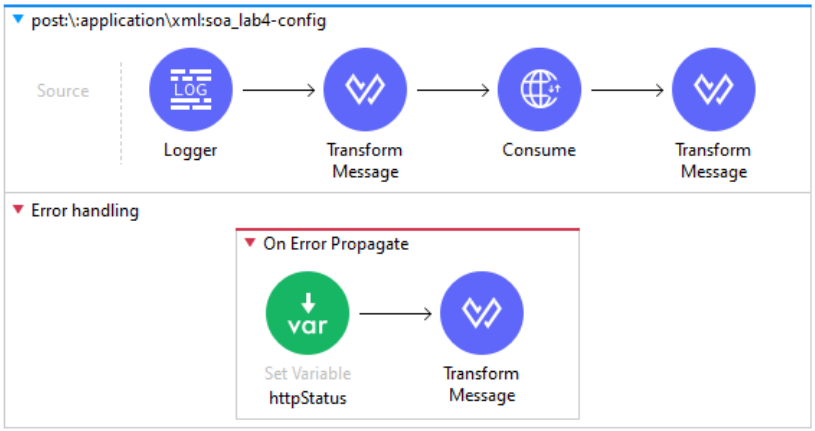
8. После консьюмера ставите transform message. Возвращаемый тип выбираете в зависимости от варианта.



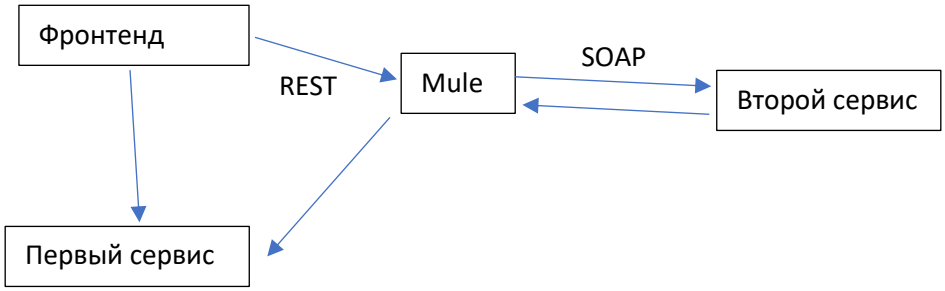
9. Добавляете обработку ошибок: нужно поставить соответствующий код возврата и передать сообщение



10. В итоге все должно выглядеть примерно так



Если у вас на SOAP **ВЫЗЫВАЮЩИЙ** сервис



Насколько я понимаю, это должно выглядеть примерно так. Но тут лучше спросить у препода. Если я прав, то работа состоит из двух частей.

1. Переписать API второго сервиса на SOAP. Сделать REST прослойку как я описал выше.
2. Заставить второй сервис обращаться к первому тоже через SOAP, добавить в Mule какой-то конвертер soap в рест. Как это сделать я к сожалению не знаю