Министерство образования и науки Российской Федерации

Федеральное государственное бюджетное образовательное учреждение

высшего профессионального образования

«Владимирский государственный университет

имени Александра Григорьевича и Николая Григорьевича Столетовых»

(ВлГУ)

Кафедра информационных систем и программной инженерии

Лабораторная работа №2

по дисциплине "Тестирование программного обеспечения"

ТЕМА РАБОТЫ:

SoapUI как инструмент эмуляции сервисов.

Выполнил:

студент гр. ПРИ-120

Парахин К.В.

Приняла:

Ассистент кафедры ИСПИ

Породникова П.А.

Владимир 2022 г.

Цель работы:

Изучить принципы работы со средством эмуляции работы веб-сервисов SoapUI.

**Веб-сервис** - идентифицируемая веб-адресом программная система со стандартизированными интерфейсами.

**Драйвер –** компонент программного обеспечения или средство тестирования, которое заменяет компонент, обеспечивающий управление и/или вызов компонента или системы.

**Заглушка** – минимальная или специализированная реализация программного компонента. Использующаяся для подмены компонента, от которого зависит разработка или тестование другого компонента системы.

Выполнение работы:

Сначала на компьютер была установлена программа SoapUI.

При входе были введены данные пользователя:

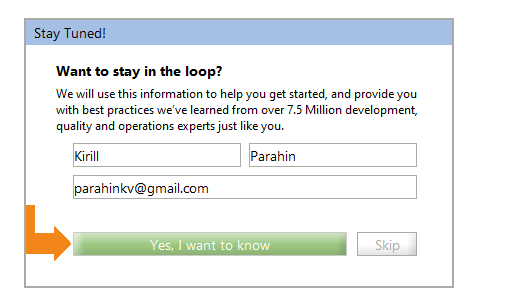


Рисунок 1. Ввод данных

К сожалению, в тех версиях программы SoapUI, которые я пробовал запускать, отсутствует функция создания проекта SoapUI (я пробовал версии 5.5.0 – 5.7.0)

Поэтому буду создавать проект типа SOAP Project

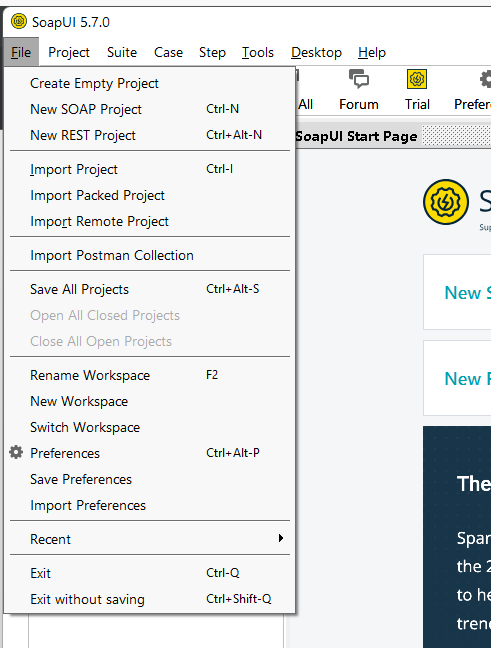


Рисунок 2. Выбор типа создаваемого проекта

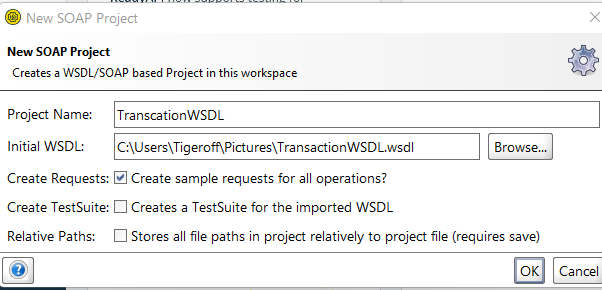


Рисунок 3. Настройка шаблона создаваемого SOAP Project

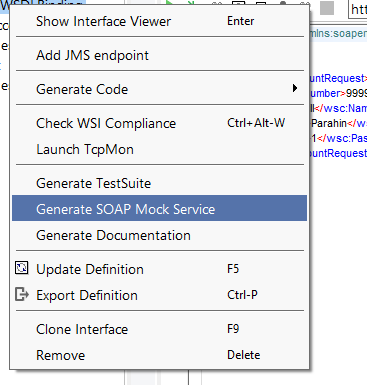


Рисунок 4. Добавление ноовго SOAP MockService

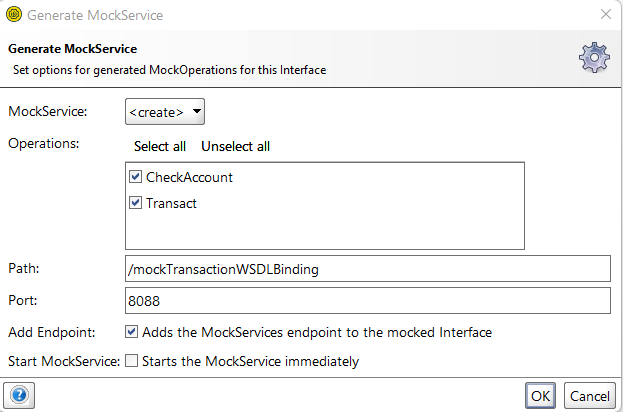


Рисунок 5. Генерация настроек создаваемого MockService

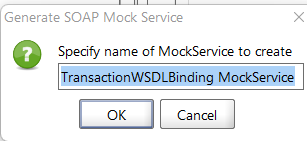


Рисунок 6. Выбор имени создаваемого MockService

После проделанных действий слева в окне навигатора можно увидеть сервис с операциями CheckAccount и Transact, а также вложенными в них запросами (Request 1) и фейковый сервис MockService с теми же операциями CheckAccount и Account, а также вложенными в них ответами на запросы от реального сервиса (Response 1).

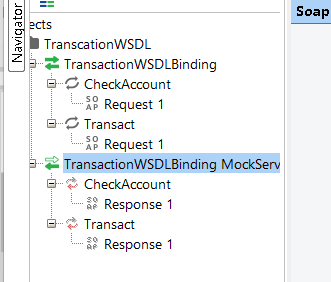


Рисунок 7. Меню сервиса и MockService

Индивидуальное задание:

Поработаем с операцией CheckAccount, заключающаяся в проверке счета на аккаунте.

Операция CheckAccount:

Входные параметры:

* AccountNumber – номер счета;
* Name – имя клиента;
* Surname – фамилия клиента;
* Passport – номер паспорта.

Параметры ответа:

* AccountNumber – номер счета;
* Status – статус (может принимать значения: Close, Open, Blocked).

Операция Transact, входные параметры:

* AccountNumber
* Sum
* Target

Параметры ответа:

* TransactionId – идентификационный номер транзакции;
* TransactionStatus – статус (Success).

Каждый сервис может вернуть ошибку состоящую из:

* ErrorCode – код ошибки;
* ErrorString – описание ошибки.

Задание заключается в следующем:

Для операции ChechAccount реализовать:

- В случае, если AccountNumber = 999999999, возвращать Close;

- В случае, если AccountNumber = 777777777, возвращать Blocked;

- В случае, если AccountNumber = 666666666, возвращать ошибочный ответ.

- Для других значений AccountNumber возвращать Open;

- В AccountNumber всегда возвращать входное значение.

Далее рассмотрим код, содержащийся в запросе Request1 операции CheckAccount на сервисе и в ответе Response1 этой же операции на MockService:

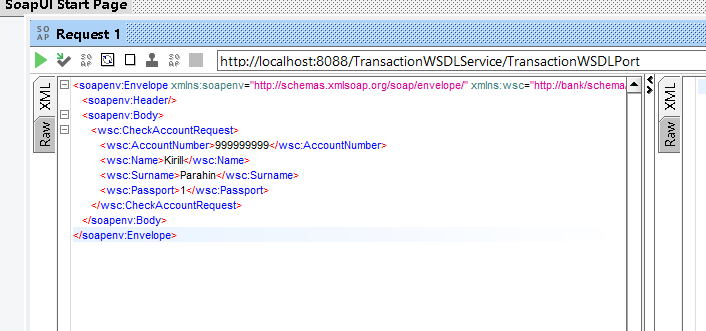


Рисунок 9. Request1

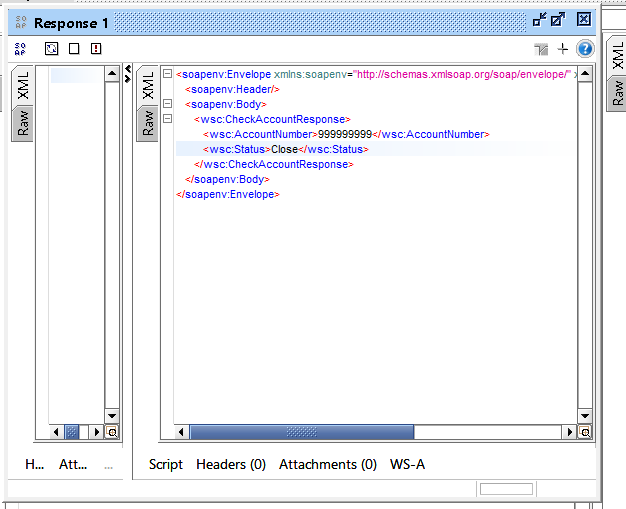


Рисунок 10. Response 1

Далее запускаем MockService и нажимаем зеленую кнопку запуска запроса от основного сервиса. В итоге справа от окна Response получаем следующий ответ:

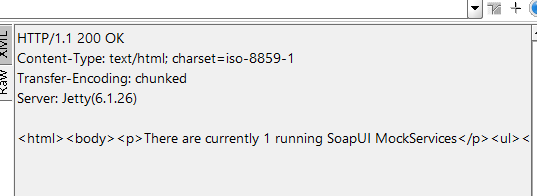


Рисунок 11. Код ответа (OK)

Далее проведем редактирование операции CheckAccount сервиса TrasnactionWSDLBinding и изменим там значение для опции Dispatch, которая отвечает за способ получения ответа.

В тестировании есть необходимость, что бы на определенные запросы приходили определенные ответы. В SoapUI есть возможность с использованием языка Groovy обрабатывать те или иные запросы, и возвращать те или иные ответы.

Укажем способ «Script», то есть запрос Request 1 будет обрабатываться скриптом.

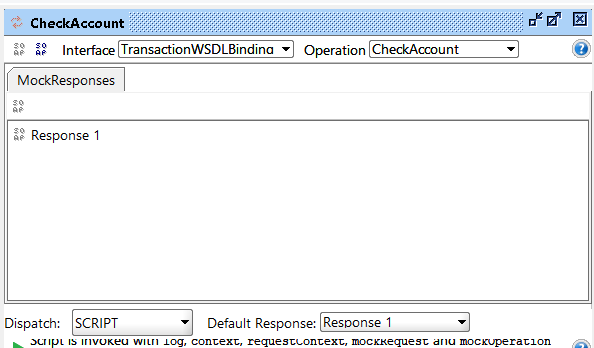


Рисунок 12. Выбор опции Dispatch

Далее в окно ниже вводим скрипт на языке Groovy, который отвечает за настройку ответов на запросы с определенным параметром (а именно - в зависимости от значения входного параметра AccountNumber на выходе будем получать разные варианты ответов согласно коду ниже) :

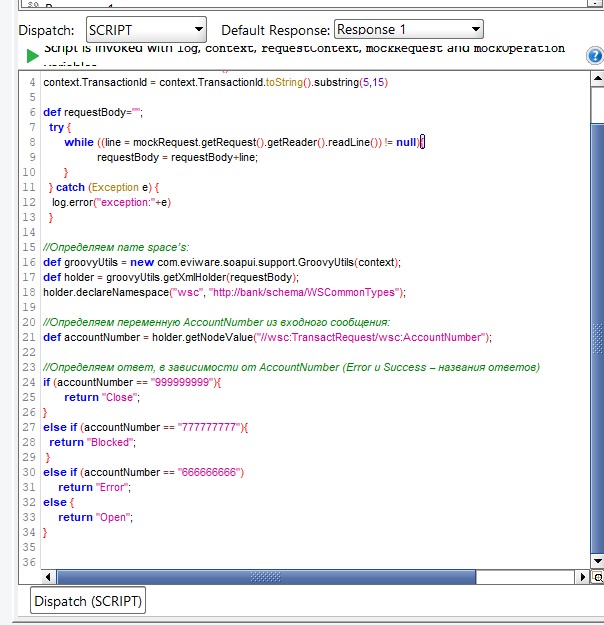


Рисунок 13. Скрипт для ответа Response 1 операции CheckAccount

Далее проверяем корректность указанной кодировки в использующихся XML – файлах и проверяем работу запросов и полученных ответов от MockService:

Основные тестовые сценарии:

1. Запрос содержит AccountNumber равный 999999999, в этом случае на выходе получим ответ «Close»

Пример:

Запрос: <wsc:AccountNumber>999999999</wsc:AccountNumber>Ожидаемый ответ: Close

1. Запрос содержит AccountNumber равный 777777777, в этом случае на выходе получим ответ «Blocked»

Пример:

Запрос: <wsc:AccountNumber>777777777</wsc:AccountNumber>

Ожидаемый ответ: Blocked

1. Запрос содержит AccountNumber равный 666666666, в этом случае на выходе получим ответ «Error» (ошибочный ответ)

Пример:

Запрос: <wsc:AccountNumber>666666666</wsc:AccountNumber>

Ожидаемый ответ: Error

1. Запрос содержит AccountNumber не равный ни 999999999, ни 777777777, ни 666666666 (то есть любое другое значение). В этом случае на выходе получим ответ «Open»

Пример:

Запрос: <wsc:AccountNumber>11111111111</wsc:AccountNumber>

Ожидаемый ответ: Open

При этом полученные ответы никак не зависят от значений входных параметров Name, Surname и Passport, которые присутствуют в коде запроса Request 1 на ряду с AccountNumber.

Проверка тестовых сценариев:

1. Запрос с AccountNumber = 99999999

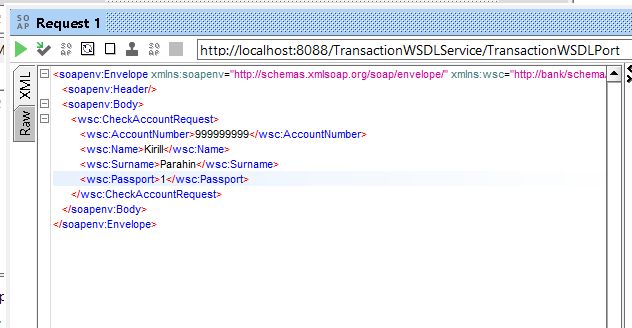


Рисунок 14.1. XML – код запроса

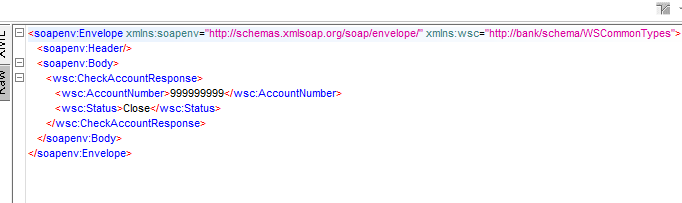


Рисунок 14.2. XML – код полученного от MockService ответа (Close)

1. Запрос с AccountNumber = 7777777777

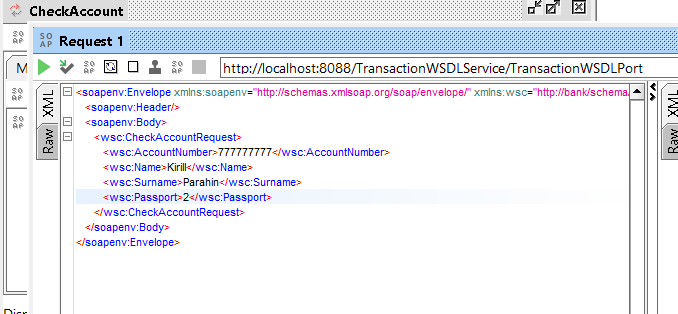


Рисунок 15.1. XML – код запроса

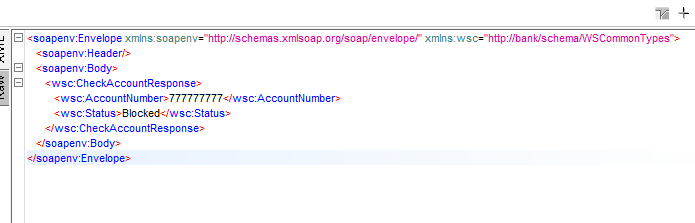


Рисунок 15.2. XML – код полученного от MockService ответа (Blocked)

1. Запрос с AccountNumber = 66666666

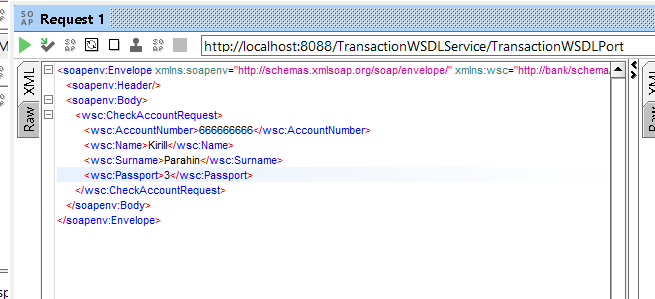


Рисунок 16.1. XML – код запроса

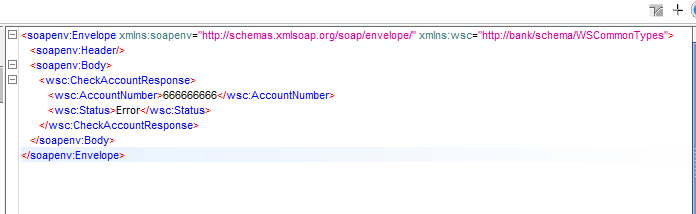


Рисунок 16.2. XML – код полученного от MockService ответа (Error)

1. Запрос с AccountNumber = 1111111111 (то есть любой не обозначенный в условиях номер)

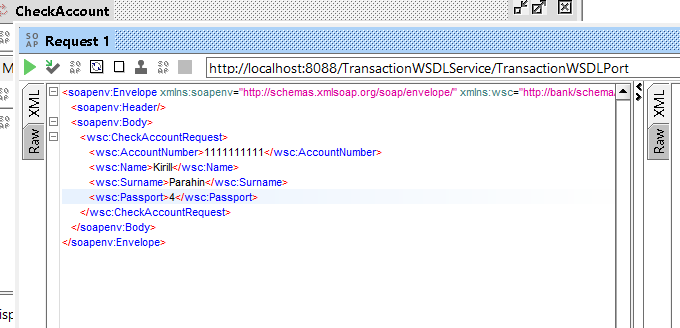


Рисунок 17.1. XML – код запроса

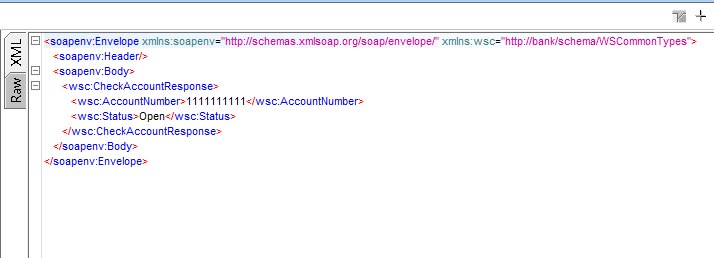


Рисунок 17.2. XML – код полученного от MockService ответа (Open)

Вывод

В результате выполнения работы, я изучил принципы работы со средством эмуляции работы веб-сервисов SoapUI.