МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

ФГБОУ ВО

«Пермский государственный национальный исследовательский университет»

Механико-математический факультет

Кафедра информационной безопасности и систем связи

**Лабораторная работа №3 по дисциплине «Технологии разработки распределенных приложений»**

Специальность 02.03.02 Фундаментальная информатика и информационные технологии

|  |  |
| --- | --- |
| Выполнил:  Бродников Вадим Сергеевич, студент механико-математического факультета, группы ММ/З-2016НБ | Проверил:  Деменев Алексей Геннадьевич, доцент кафедры прикладной математики и информатики  механико-математического факультета, к.ф-м.н., доц. |

Пермь, 2021 г

**Введение**

**Цель:** Изучение возможностей технологии веб-служб для создания распределенных приложений.

**Задача:** реализовать веб-сервис и приложение, его использующее, при этом:

1. Веб-сервис и приложение, его использующее, должны быть разработаны на разных объектно-ориентированных языках программирования.
2. Веб-сервис должен принимать параметры и передавать приложению результат.
3. В качестве параметров должны передаваться объекты классов, написанных самостоятельно, т.е. не должны передаваться строки, числа или другие простейшие типы

**Постановка задачи**

Обеспечить проектирование, реализацию, проверку созданного приложения требованиям в таблице:

***Распределение баллов за выполнение работы (***Максимальное количество баллов — 6)***:***

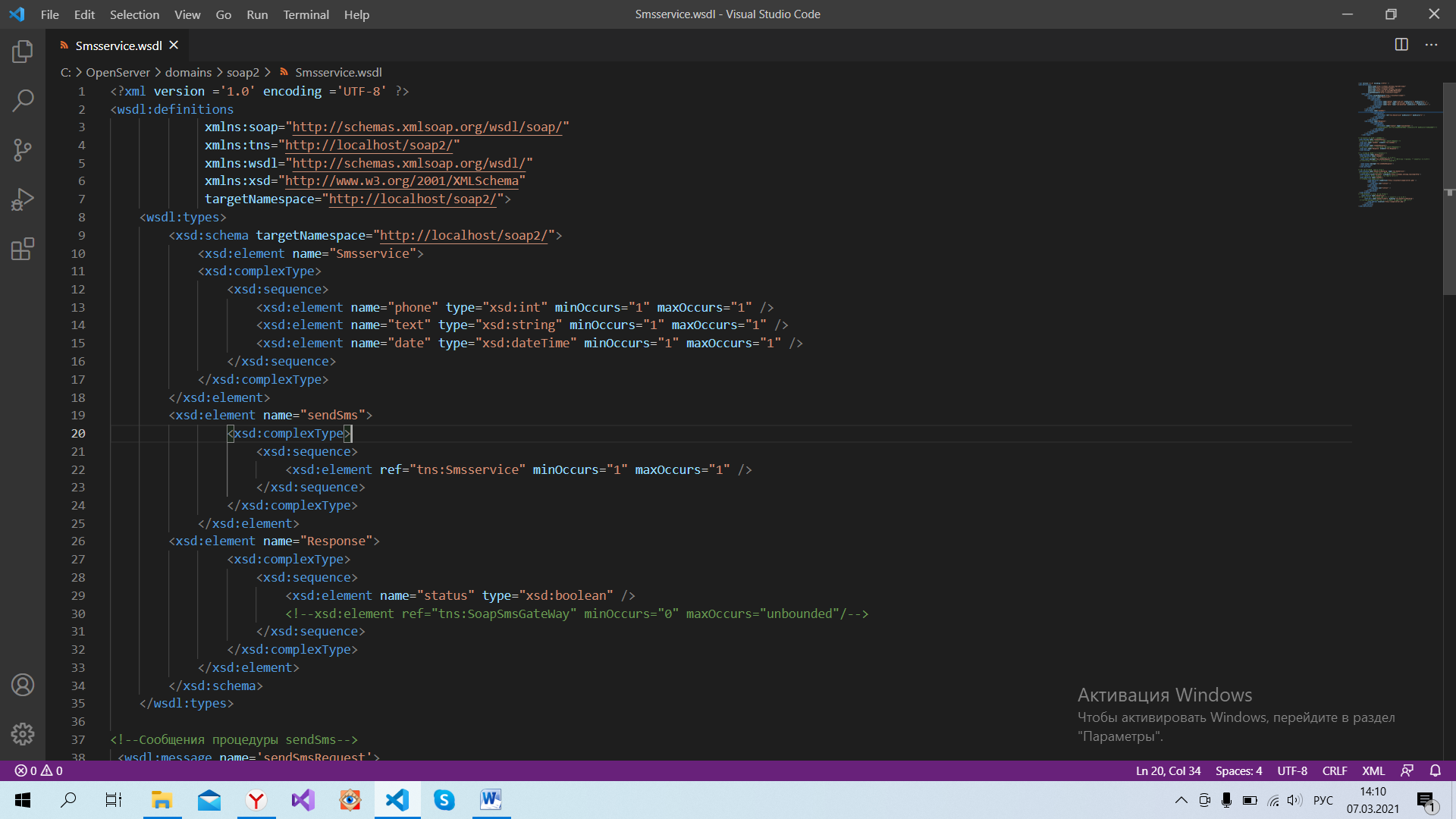
|  |  |
| --- | --- |
| **Требование к заданию** | **Максимальное количество баллов** |
| Приложение, написанное студентом, работает в сети Интернет без сбоев. | 2 |
| Веб-сервис и приложение, его использующее, написаны на разных языках программирования. | 1 |
| Веб-сервис принимает от приложения исходные данные и возвращает результат. | 1 |
| В качестве параметров веб-сервис принимает объекты классов. | 2 |

**Проектирование**

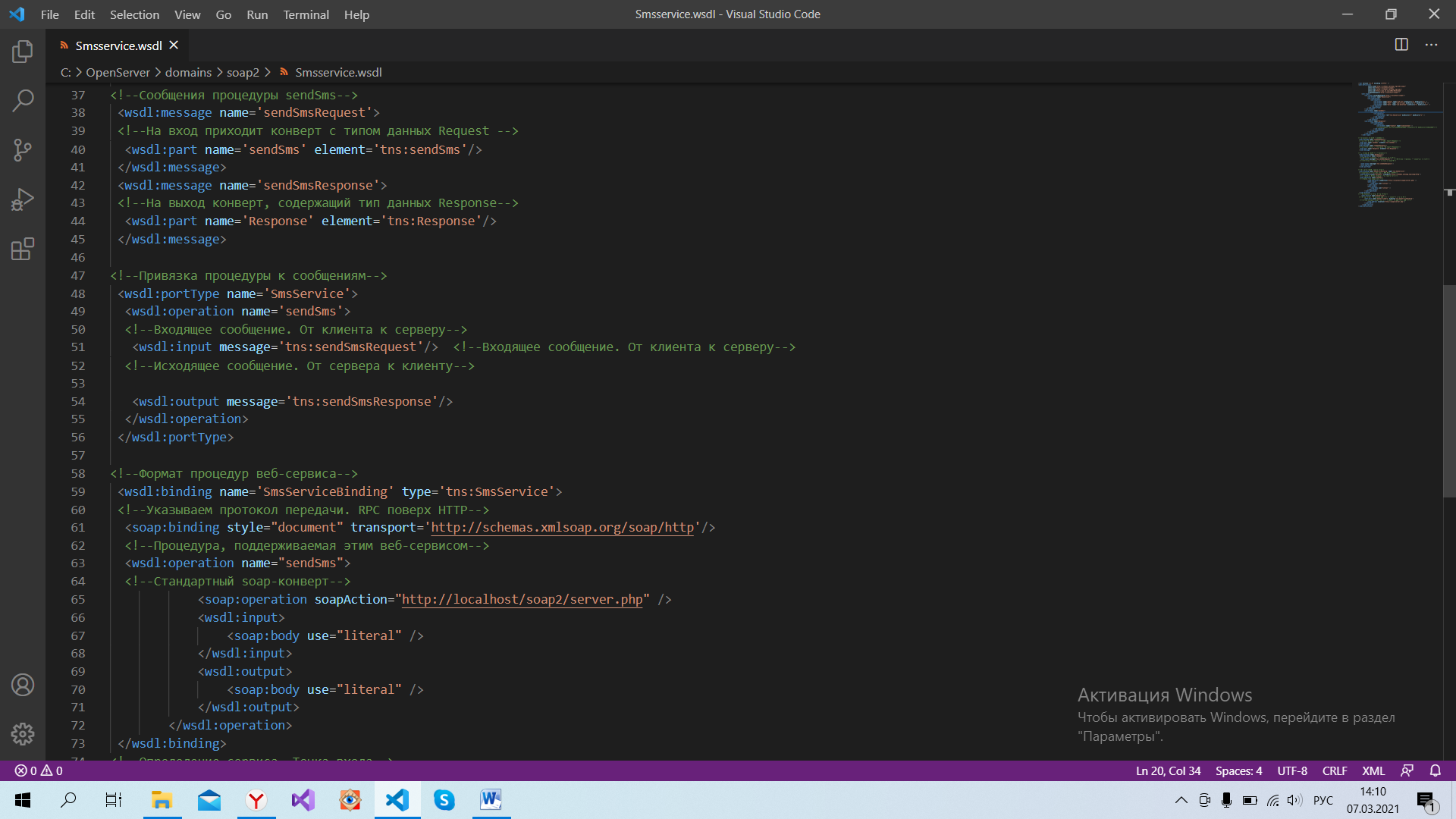
Веб-сервис осуществляет прием данных, отправленных от клиента, и дальнейшая их запись в базу данных.

Веб-сервис реализуется с помощью парадигмы RPC [15] (Remote Procedure Call, удалённый вызов процедур). Подход RPC позволяет использовать небольшое количество сетевых ресурсов с большим количеством методов и сложным протоколом. Т.е., это означает, что на сайте в случае RPC подхода будет всегда один вход (ссылка) на сервис и какую процедуру вызывать для обработки поступающих данных мы передаем вместе с данными. SOAP [1] (от англ. Simple Object Access Protocol — простой протокол доступа к объектам) – протокол используемый для реализации удаленного вызова процедур и в качестве транспорта он использует XML [15]. SOAP работает с WSDL [2] – язык описания веб-сервисов и доступа к ним, основанный на языке XML. WSDL по сути является веб-сервисом. В нашем случае он должен принимать объект типа stdClass, состоящий из int значения «номер телефона» string значения «текст сообщения» и date значения даты отправки сообщения. Потом отправлять этот объект на файл-обработчик и возвращать результат в виде статуса, то есть подтверждение, что файл-обработчик получил объект с клиента.

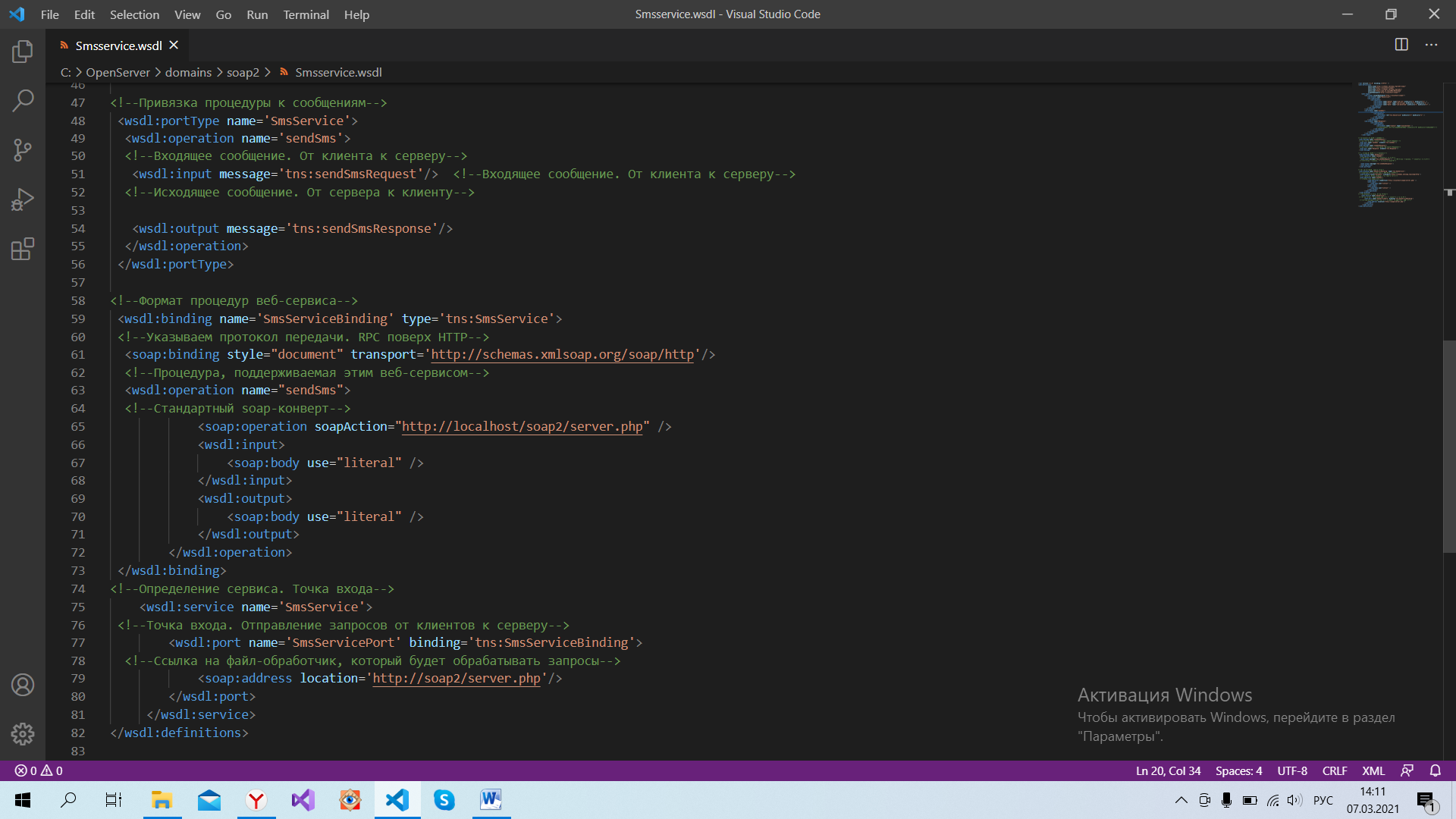
Был сформирован следующий WSDL файл:



**Рис. 1.** Содержимое файла Smsservice.wsdl



**Рис. 2.** Содержимое файла Smsservice.wsdl



**Рис. 3.** Содержимое файла Smsservice.wsdl

Таким образом, были спроектирован файл типа XML.

Он состоит из 5 блоков:

Самый первый блок – определение веб-сервиса или другими словами – точки входа.

|  |
| --- |
| <?xml version="1.0" encoding="utf-8"?>  <wsdl:definitions>  <!—Определение сервиса -->  <wsdl:service name="SmsService">  <wsdl:port name="SmsServicePort" binding="tns:SmsServiceBinding">  <soap:address location="http://soap2/server.php" />  </wsdl:port>  </wsdl:service>  </wsdl:definitions> |

Здесь написано, что у нас есть сервис, который называется – «SmsService».

После этого мы объявляем о том, что в нашем веб-сервисе «SmsService» есть точка входа («port»), которая называется «SmsServicePort». Именно в эту точку входа и будут отправляться все запросы от клиентов к серверу. И указываем в элементе «address» ссылку на файл-обработчик, который будет принимать запросы.

|  |
| --- |
| <!—Формат процедур веб-сервиса -->  <wsdl:binding name="SmsServiceBinding" type="tns:SmsService">  <soap:binding style=”document” transport="http://schemas.xmlsoap.org/soap/http" />  <wsdl:operation name="sendSms">  <soap:operation soapAction="http://localhost/soap2/server.php" />  <wsdl:input>  <soap:body use="literal" />  </wsdl:input>  <wsdl:output>  <soap:body use="literal" />  </wsdl:output>  </wsdl:operation>  </wsdl:binding> |

Для этого перечисляется какие операции и в каком виде у будут вызываться. Т.е. для порта «SmsServicePort» определена привязка под именем «SmsServiceBinding», которая качестве протокола передачи (транспорта) используется HTTP. После этого мы описываем какие процедуры (operation) поддерживаются в веб-сервисе. Мы будем поддерживать всего одну процедуру – «sendSms». Через эту процедуру будут отправляться на сервер наши сообщения! После того, как была объявлена процедура, необходимо указать в каком виде будут передаваться данные. В данном случае указано, что будут использоваться стандартные SOAP-конверты.

После этого нам необходимо привязать процедуру к сообщениям:

|  |
| --- |
| <!— Привязка процедуры к сообщениям -->  <wsdl:portType name="SmsService">  <wsdl:operation name="sendSms">  <wsdl:input message="tns:sendSmsRequest" />  <wsdl:output message="tns:sendSmsResponse" />  </wsdl:operation>  </wsdl:portType> |

Для этого мы указываем, что наша привязка («binding») имеет тип «SmsService» и в элементе «portType» с одноименным типу именем указываем привязку процедур к сообщениям. И так, входящее сообщение (от клиента к серверу) будет называться «sendSmsRequest», а исходящее (от сервера к клиенту) «sendSmsResponse».

Теперь нам необходимо описать сами сообщения, т.е. входящие и исходящие:

|  |
| --- |
| <!-- Сообщения процедуры sendSms -->  <wsdl:message name="sendSmsRequest">  <wsdl:part name="sendSms" element="tns:sendSms" />  </wsdl:message>  <wsdl:message name="sendSmsResponse">  <wsdl:part name="Response" element="tns:Response" />  </wsdl:message> |

Для этого мы добавляем элементы «message» с именами «sendSmsRequest» и «sendSmsResponse» соответственно. В них мы указываем, что на вход должен прийти конверт, структура которого соответствует типу данных «sendSms». После чего с сервера возвращается конверт содержащий тип данных – «Response».

Теперь надо добавить описание данных типов в наш WSDL-файл. входящие и исходящие данные при помощи XML-схем:

|  |
| --- |
| <wsdl:definitions  xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"  xmlns:tns="http://localhost/soap2/"  xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/"  xmlns:xsd="http://www.w3.org/2001/XMLSchema"  targetNamespace="http://localhost/soap2/">  <wsdl:types>  <xsd:schema targetNamespace="http://localhost/soap2/">  <xsd:element name="Smsservice">  <xsd:complexType>  <xsd:sequence>  <xsd:element name="phone" type="xsd:int" minOccurs="1" maxOccurs="1" />  <xsd:element name="text" type="xsd:string" minOccurs="1" maxOccurs="1" />  <xsd:element name="date" type="xsd:dateTime" minOccurs="1" maxOccurs="1" />  </xsd:sequence>  </xsd:complexType>  </xsd:element>  <xsd:element name="sendSms">  <xsd:complexType>  <xsd:sequence>  <xsd:element ref="tns:Smsservice" minOccurs="1" maxOccurs="1" />  </xsd:sequence>  </xsd:complexType>  </xsd:element>  <xsd:element name="Response">  <xsd:complexType>  <xsd:sequence>  <xsd:element name="status" type="xsd:boolean" />  <!--xsd:element ref="tns:SoapSmsGateWay" minOccurs="0" maxOccurs="unbounded"/-->  </xsd:sequence>  </xsd:complexType>  </xsd:element>  </xsd:schema>  </wsdl:types> |

То есть мы отправляем объект Smsservice с 3 данными: число, строка и дата. А обратно Response булево данное, уведомляющее об успешности передачи объекта.

То есть описан сервис «SmsService», содержащий единственный метод «sendSms», принимающий на вход объект типа «SendSms» и возвращающий в качестве результата объект типа «Response».

Для разработки клиентского приложения был выбран объектно-ориентированный язык программирования Python3 3.9.1 [5,6]. В качестве среды разработки была выбрана среда Python IDLE [6].

Для разработки серверного приложения был выбран объектно-ориентированный язык программирования PHP 7.4.6 [7]. В качестве среды разработки был выбрана среды Visual Studio Code 1.54 [3,4].

**Сценарии использования программы-клиента**

Сценарий использования №1 – отправка данных

Предусловия: Сервер запущен

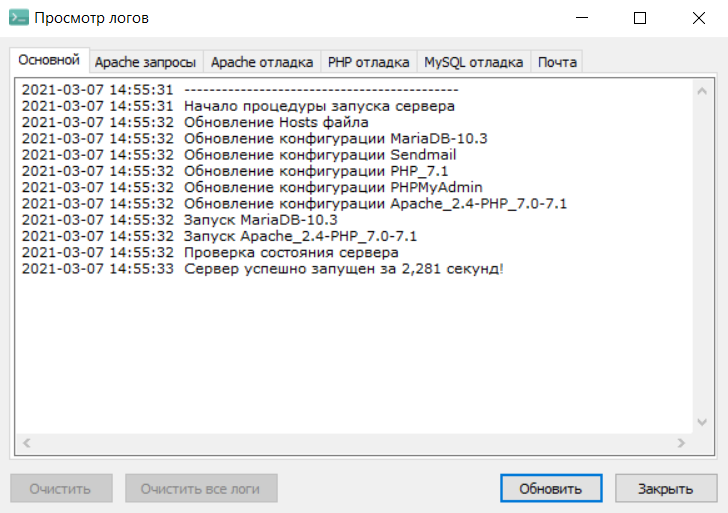
1. Ввод номера телефона, который собираетесь отправить
2. Ввод сообщения, который собираетесь отправить
3. Ожидание получения ответа

Ожидаемый результат: от сервера получен ответ в виде True, соответствующий подтверждение отправки данных

**Тестирование**

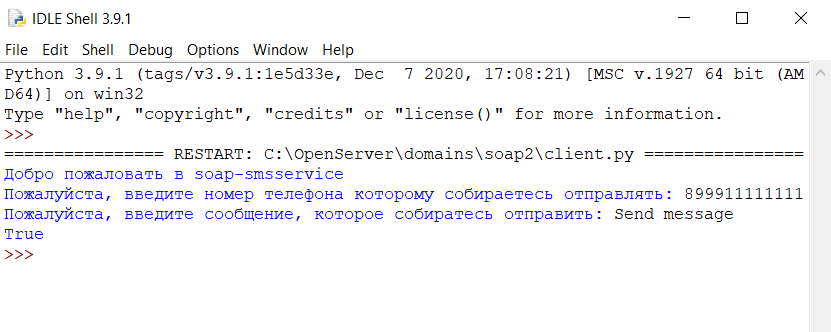
В результате разработки были созданы client.py – клиентское приложение, server.php – серверное приложение, Smsservice.wsdl – описание веб-сервиса и test.db в качестве базы данных

Перед началом тестирования выполним предусловие для сценария – запустим сервер. Для этого воспользуемся панелью Open Server Panel 5.3.8 [10,11], она включает Apache 2.4 [14] и PHP 7.0.-7.1 [7]:



**Рис. 4.** Запуск сервера

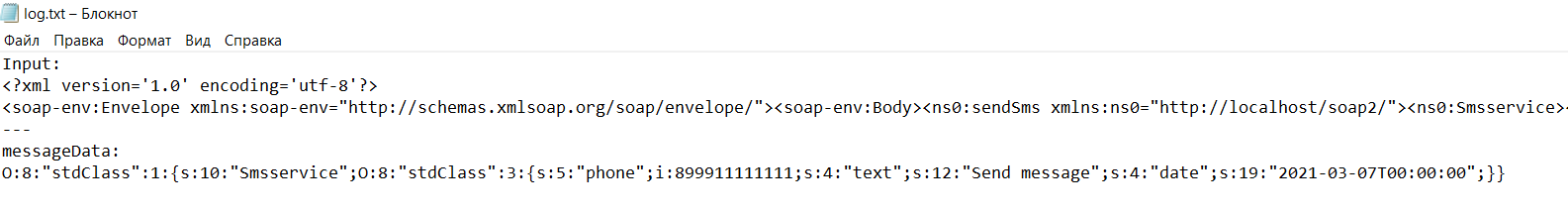
Сценарий №1

****

**Рис. 5.** Работа клиента

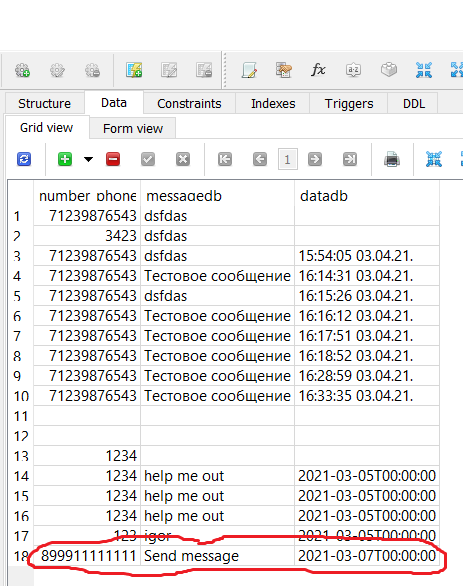
Результат: От сервера получен ответ, соответствующий подтверждению получения данных от клиента.

Сервер создает текстовый файл log.txt показав какую информацию получает от клиента. StdClass. Значит, объект.



**Рис. 6.** Содержимое log.txt

Для проверки, что в базу данных записался объект. Воспользуемся приложением SQLiteStudio v.3.2.1 [12,13]



**Рис. 7**. Демонстрация БД “test.db” с таблицей “smsservice” в программе SQLiteStudio 3.2.1

Результат: в БД записался объект. Тест выполнен успешно.

**Заключение**

В результате работы была достигнута цель и выполнены следующие требования:

* Требование «Приложение, написанное студентом, работает в сети Интернет без сбоев» выполнено полностью. Ожидаемое количество баллов: 2.
* Требование «Веб-сервис и приложение, его использующее, написаны на разных языках программирования» выполнено полностью(Python3 и PHP). Ожидаемое количество баллов: 1.
* Требование «Веб-сервис принимает от приложения исходные данные и возвращает результат» выполнено полностью. Ожидаемое количество баллов: 1.
* Требование «В качестве параметров веб-сервис принимает объекты классов» выполнено полностью. Ожидаемое количество баллов: 2.

**Перечень использованных источников**

1. Стандарт SOAP – протокол взаимодействия сервисов. Лекция по SOAP (дата обращения: 07.03.2021): <http://khpi-iip.mipk.kharkiv.edu/library/sotii/lectures/Lecture5.pdf>
2. WSDL — Краткое руководство (дата обращения 07.03.2021): <https://coderlessons.com/tutorials/xml-tekhnologii/uznaite-wsdl/wsdl-kratkoe-rukovodstvo>
3. Ссылка на скачивание Visual Studio Code v.1.54 (дата обращения 07.03.2021): <https://code.visualstudio.com/download>
4. MIT Лицензия Visual Studio Code (дата обращения 07.03.2021): <https://github.com/microsoft/vscode/blob/main/LICENSE.txt>
5. Ссылка на скачивание Python3 3.9.1[Электронный ресурс]: URL: <https://www.python.org/downloads/> (дата обращения: 07.03.2021)
6. Спецификация Python IDLE 3.9.1 (дата обращения: 07.03.2021): <https://docs.python.org/3.9/library/idle.html>
7. Лицензия языка PHP. URL: <https://www.php.net/license/> (дата обращения: 06.03.21)
8. Документация библиотеки Zeep: Python SOAP Client (дата обращения 07.03.2021): <https://docs.python-zeep.org/en/master/index.html>
9. Лицензия на Zeep (дата обращения 07.03.2021): <https://github.com/mvantellingen/python-zeep/blob/master/LICENSE>
10. Ссылка на скачивание Open Server Panel 5.3.8 (дата обращения 07.03.2021): <https://ospanel.io/download/>
11. Лицензия Open Server Panel 5.3.8 (дата обращения 07.03.2021): <https://ospanel.io/license/>
12. Ссылка на скачивание SQLiteStudio v.3.2.1 (дата обращения 07.03.2021): <https://sqlitestudio.pl/>
13. Лицензия SQLiteStudio v.3.2.1 (дата обращения 07.03.2021): <https://github.com/pawelsalawa/sqlitestudio/blob/master/LICENSE>
14. Ссылка на лицензию Apache 2.0 (дата обращения: 07.02.2021): <https://www.apache.org/licenses/LICENSE-2.0.txt>
15. Пишем SOAP клиент-серверное приложение на PHP. Также об технологии RPC, XML-Schema (дата обращения 07.03.2021):

<https://habr.com/ru/post/187390/>