Міністерство освіти і науки України

Національний університет «Львівська політехніка»

Інститут комп’ютерних наук та інформаційних технологій

Кафедра програмного забезпечення



**ЗВІТ**

**Про виконання лабораторної роботи № 4**

**«Тестування Soap Web-service»**

**з дисципліни «Управління якістю програмного забезпечення»**

**Лектор:**

асист. кафедри ПЗ

Ваврук І.Є.

**Виконав:**

студ. групи ПЗМ-11

Павленчик М. М.

**Прийняв:**

асист. кафедри ПЗ

Ваврук І.Є.

«\_\_\_» \_\_\_\_\_\_ 2019 р.

∑ = \_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_

Львів – 2019

**ТЕМА РОБОТИ**: Тестування Soap Web-service.

**МЕТА РОБОТИ**: Освоїти основні принципи тестування Soap Web-service.

### ЛАБОРАТОРНЕ ЗАВДАННЯ

1. Завантажити WSDL-файл, використовуючи посилання: http://www.dneonline.com/calculator.asmx?wsdl
2. Написати SOAP-клієнт для тестування сервісу.
3. Протестувати всі методи, наявні в WSDL, використовуючи дата провайдер.
4. Здійснити логування усіх тестів в текстовий документ.
5. Сформувати звіт.

### РЕЗУЛЬТАТИ

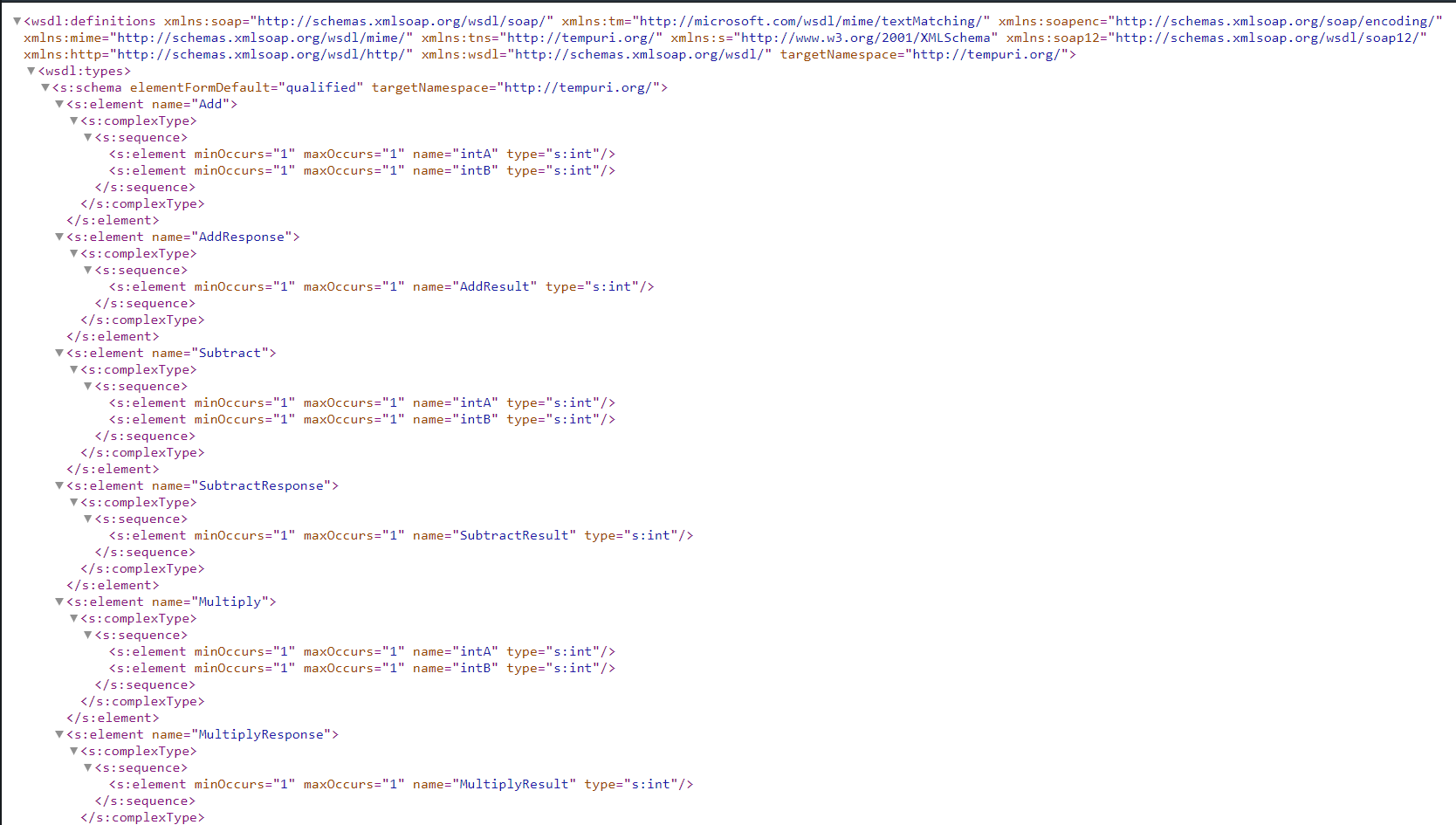


Рис. 1. Завантажений WSDL-файл

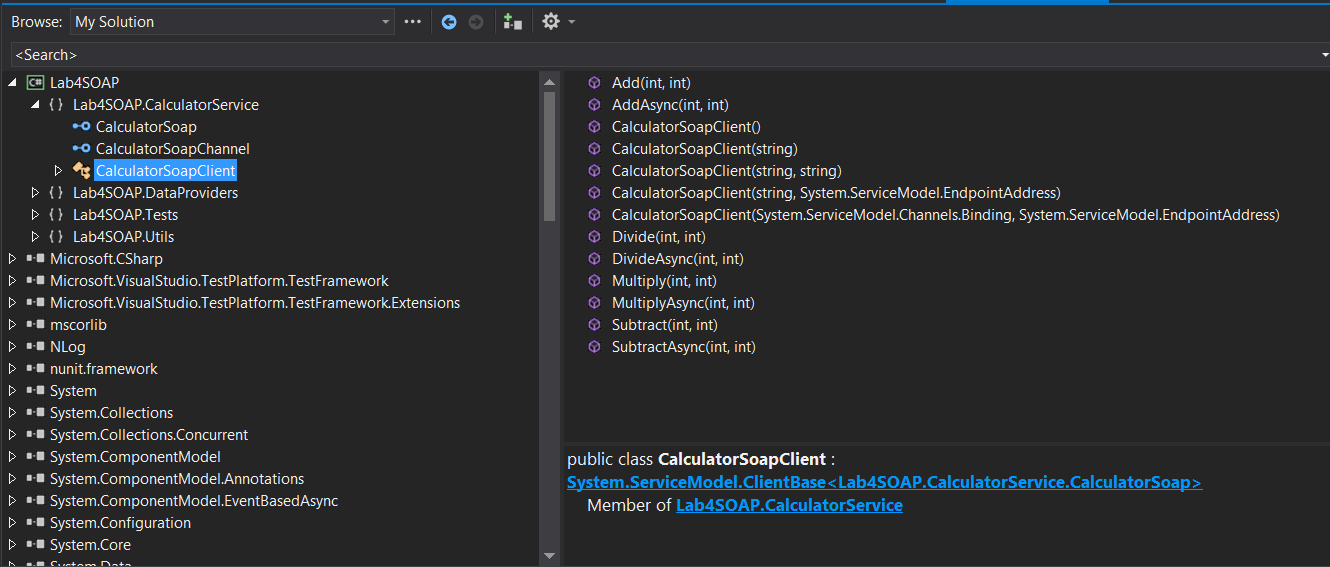


Рис. 2. Реалізований SOAP-клієнт для тестування сервісу

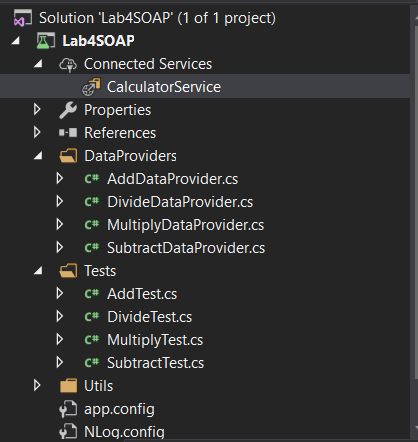


Рис. 3. Структура проекту із створеними тестами та DataProvider класами

**Код тестів для методів додавання:**

[TestFixture]

public class AddTest

{

private static readonly Logger Logger = LogManager.GetCurrentClassLogger();

[TestCaseSource(typeof(AddDataProvider), "NormalTestData")]

public void Add\_CorrectData\_CorrectResult(int intA, int intB, int result)

{

//arrange

Logger.Info($"intA: {intA}; intB: {intB}");

var client = SOAPClientFactory.CreateClient();

//act

int actualResult = client.Add(intA, intB);

Logger.Info($"Result: {actualResult}");

//assert

Assert.AreEqual(result, actualResult);

}

[TestCaseSource(typeof(AddDataProvider), "OverflowTestData")]

public void Add\_OwerflowData\_ThrowsOverflowException(int intA, int intB)

{

//arrange

Logger.Info($"intA: {intA}; intB: {intB}");

var client = SOAPClientFactory.CreateClient();

//act

var ex = Assert.Throws<FaultException>(() => client.Add(intA, intB));

//assert

Assert.That(ex.Message.Contains("OverflowException"));

}

[TestCaseSource(typeof(AddDataProvider), "NormalTestData")]

public void AddAsync\_CorrectData\_CorrectResult(int intA, int intB, int result)

{

//arrange

Logger.Info($"intA: {intA}; intB: {intB}");

var client = SOAPClientFactory.CreateClient();

//act

var taskResult = client.AddAsync(intA, intB);

Task.WaitAny(taskResult);

var actualResult = taskResult.Result;

Logger.Info($"Result: {actualResult}");

//assert

Assert.AreEqual(result, actualResult);

}

[TestCaseSource(typeof(AddDataProvider), "OverflowTestData")]

public void AddAsync\_OwerflowData\_ThrowsOverflowException(int intA, int intB)

{

//arrange

Logger.Info($"intA: {intA}; intB: {intB}");

var client = SOAPClientFactory.CreateClient();

//act

var ex = Assert.Throws<AggregateException>(() =>

{

var taskResult = client.AddAsync(intA, intB);

Task.WaitAny(taskResult);

var actualResult = taskResult.Result;

});

//assert

Assert.That(ex.InnerException.Message.Contains("OverflowException"));

}

}

**DataProvider для методів додавання:**

class AddDataProvider

{

static object[] NormalTestData =

{

new object[] { 1, 5, 6 },

new object[] { 10, 526, 536 },

new object[] { -5, 5, 0 },

new object[] { -1485, -7789, -9274 }

};

static object[] OverflowTestData =

{

new object[] { int.MaxValue, 5 },

new object[] { int.MinValue, -5 }

};

}

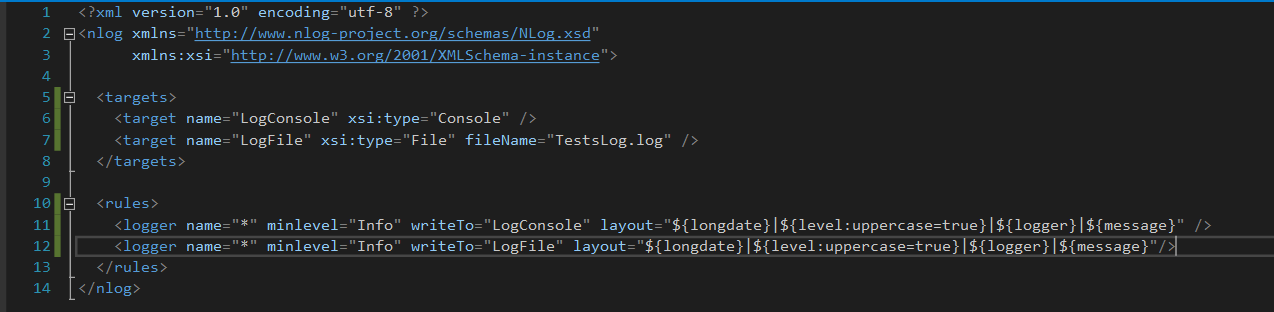


Рис. 4. Файл конфігурації для логування за допомогою бібліотеки NLog

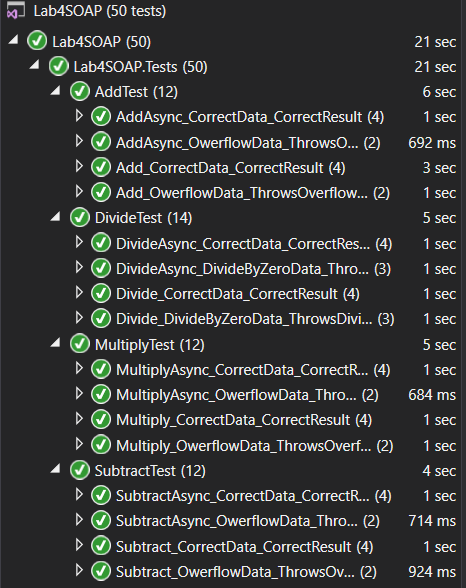


Рис. 5. Результати запуску тестів

### ВИСНОВКИ

На даній лабораторній роботі я освоїв основні принципи тестування Soap Web-service.