Development of distributed services - Project III

Jan Magne Tjensvold

November 11, 2007

Chapter 1

Project III

1.1 Introduction

In this project we were going to make a web service from one of the assignments from last project. I choose to base my web service on the bubble sort algorithm assignment (assignment 2). By foresight I have already written all the business logic for the project II assignments as class libraries (compiles to .dll files). The only remaining thing was to write a web service as a wrapper around the class library, and do some slight modifications to the original client so that it uses the web service instead.

1.2 Implementation

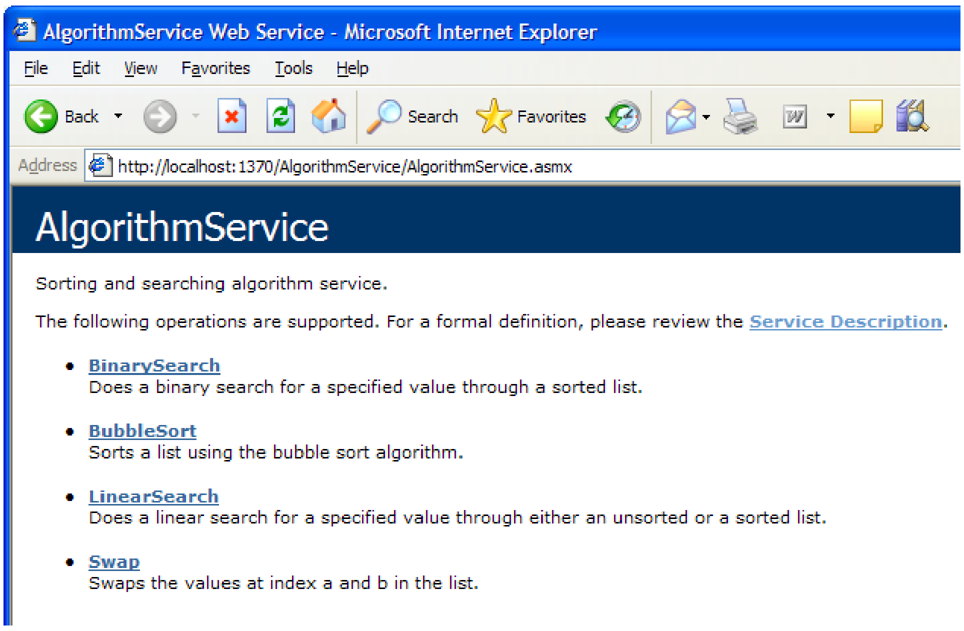
This project focused on implementing an algorithm web service using C# together with ASP.NET. The algorithm web service should provide an interface to the bubble sort, linear search and binary search algorithms implemented in project II, assignment 2. For convenience it also includes an interface to the swap method which accepts an integer array and two index values. This method returns the array where the contents of the two elements has been swapped.

1.2.1 Web service

The web service was implemented in the AlgorithmService.asmx file. By adding a reference to the algorithm class library the sorting and searching methods could be used directly by the web service. No modifications had to be made to the original algorithm class library. Instead some adjustments had to be made in the web service because of the way the results were returned by the algorithm library. The sort and swap methods in the library returns the results through their input parameters instead of using the `return` statement. A web service, on the other hand, needs to return the value by using the `return` statement.

An advantage of the algorithm library is that it is entirely stateless. This means that it does not have to implement a session mechanism which stores session information or state data. This results in very compact code and a highly performance efficient web service. Stateful web services often require that additional calls are made to register and/or deregister. These calls usually return a session ID of some sort which needs to be provided in subsequent calls. A stateless web service, on the other hand, does not suffer the overhead of these calls and the additional session ID data.

In the appendix at the end of the report you can see the program code for AlgorithmService.asmx and also the Web Services Description Language (WSDL) file the web service automatically generates. The algorithm library code from project II has been reproduced in the appendix as well.


AlgorithmService Web Service - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back
Search
Favorites
Address http://localhost:1370/AlgorithmService/AlgorithmService.asmx

AlgorithmService

Sorting and searching algorithm service.

The following operations are supported. For a formal definition, please review the Service Description.

- BinarySearch
  Does a binary search for a specified value through a sorted list.
- BubbleSort
  Sorts a list using the bubble sort algorithm.
- LinearSearch
  Does a linear search for a specified value through either an unsorted or a sorted list.
- Swap
  Swaps the values at index a and b in the list.

Figure 1.1: ASP.NET web page for the algorithm web service.

In Figure 1.1 you can see the ASP.NET web page for the web service running on localhost. It uses Visual Studio 2005's ASP.NET Development Server to serve the web service for testing and debugging purposes. To deploy the web service in a production environment AlgorithmService.asmx should be placed in a virtual folder on a web server along with Algorithms.dll which should be placed in a subdirectory called "bin".

1.2.2 Web service client

The web service was derived from the original algorithm console application from project II. Due to the way that web services return their results a few modifications had to be made. However, the amount of modifications required was very low and the client program remained largely the same.

The only modifications required was adding the following code on the lines 20-21 in Program.cs to create the web service

```
AlgorithmService.AlgorithmService algo =
    new AlgorithmService.AlgorithmService();
```

and changing the method calls to use the `algo` object instead of the static methods on the `Algorithm` class. A web reference to the web service project was also added so that the proxy code for the web service was automatically generated (instead of running wdsl.exe each time).

Below you can see the output from the web service client:

```
BubbleSort
==========

Sorting an array of integers:
Before: {8, 4, 2, 6, 1}
After:  {1, 2, 4, 6, 8}

Doing a linear search with an array of integers:
Array: {3, 2, 1, 9, 2, 4, 0, 5, 3, 0}
The first occurrence of 2 was found at index 1 in the search array

Sorting another array of integers:
Before: {3, 2, 1, 9, 2, 4, 0, 5, 3, 0}
After:  {0, 0, 1, 2, 2, 3, 3, 4, 5, 9}

Doing a binary search with a sorted array of integers:
Array: {0, 0, 1, 2, 2, 3, 3, 4, 5, 9}
2 was found at index 4 in the search array
```

The output has the exact same format as the original algorithm client developed in project II. From the user's perspective the only difference is a slight delay each time the web service is queried.

Appendix A

Program code

```
using System;
using System.Collections.Generic;
using System.Text;

namespace Algorithms
{
    /**
     * A small collection of sorting and searching algorithms.
     */
    public class Algorithms
    {
        /**
         * Use the Bubble sort algorithm to sort a list of integers in
         * ascending order.
         */
        public static void BubbleSort(int[] list)
        {
            for (int i = list.Length - 1; i > 0; i--)
                for (int j = 0; j < i; j++)
                    if (list[j] > list[j + 1]) Swap(list, j, j + 1);
        }

        /**
         * Swap the elements at index a and b in the list.
         */
        public static void Swap(int[] list, int a, int b)
        {
            int temp = list[a];
            list[a] = list[b];
            list[b] = temp;
        }

        /**
         * Does a linear search through an array of integers and returns the
         * index of the first occurrence of the specified value. If the value
         * cannot be found it returns -1.
         */
        public static int LinearSearch(int[] list, int value)
        {
            for (int i = 0; i < list.Length; i++)
                if (list[i] == value) return i;
            return -1;
        }

        /**
         * Does a binary search through a sorted array of integers and returns
         * the index of the specified value. If the value being searched for
         * occurs multiple times (in sequence) this algorithm will may not
         * return the index of the first occurrence. If the value cannot be
         * found it returns -1.
         */
        public static int BinarySearch(int[] list, int value)
        {
            int low = 0;
            int high = list.Length - 1;
            int mid;

            while (low <= high)
            {
                mid = (low + high) / 2;
                if (list[mid] > value)
                    high = mid - 1;
                else if (list[mid] < value)
                    low = mid + 1;
                else
                    return mid;
            }

            return -1;
        }
    }
}
```

```
<%@ WebService Language="C#" Class="AlgorithmService" %>

using System;
using System.Web;
using System.Web.Services;
using System.Web.Services.Protocols;

[WebService (Namespace = "http://www.uis.no/ws/",
    Description = "Sorting and searching algorithm service.")]
[WebServiceBinding(ConformsTo = WsiProfiles.BasicProfile1_1)]
public class AlgorithmService : WebService
{
    [WebMethod (Description = "Sorts a list using the bubble sort algorithm.")]
    public int[] BubbleSort(int[] list)
    {
        Algorithms.Algorithms.BubbleSort(list);
        return list;
    }

    [WebMethod (Description = "Swaps the values at index a and b in the "
        + "list.")]
    public int[] Swap(int[] list, int a, int b)
    {
        Algorithms.Algorithms.Swap(list, a, b);
        return list;
    }

    [WebMethod (Description = "Does a linear search for a specified value " +
        "through either an unsorted or a sorted list.")]
    public int LinearSearch(int[] list, int value)
    {
        return Algorithms.Algorithms.LinearSearch(list, value);
    }

    [WebMethod(Description = "Does a binary search for a specified value " +
        "through a sorted list.")]
    public int BinarySearch(int[] list, int value)
    {
        return Algorithms.Algorithms.BinarySearch(list, value);
    }
}
```

```
using System;
using System.Collections.Generic;
using System.Text;

namespace AlgorithmClient
{

    /**
     * A program to show how the sorting and searching algorithms are used
     * together with a web service.
     */
    class Program
    {
        static void Main(string[] args)
        {
            Console.WriteLine();
            Console.WriteLine("BubbleSort");
            Console.WriteLine("==========");

            AlgorithmService.AlgorithmService algo =
                new AlgorithmService.AlgorithmService();

            Console.WriteLine();
            Console.WriteLine("Sorting an array of integers:");
            int[] sort = {8, 4, 2, 6, 1};
            Console.WriteLine("Before: " + ArrayToString(sort));

            sort = algo.BubbleSort(sort);
            Console.WriteLine("After:  " + ArrayToString(sort));

            Console.WriteLine();
            Console.WriteLine(
                "Doing a linear search with an array of integers:");
            Random ran = new Random();
            int[] search = RandomArray(10, 0, 9, ran);
            Console.WriteLine("Array: " + ArrayToString(search));
            int value = ran.Next(0, 10);
            int result = algo.LinearSearch(search, value);
            if (result < 0)
                Console.WriteLine("No occurrence of " + value
                    + " was not found in the search array");
            else
                Console.WriteLine("The first occurrence of " + value
                    + " was found at index " + result + " in the search array");

            Console.WriteLine();
            Console.WriteLine("Sorting another array of integers:");
            Console.WriteLine("Before: " + ArrayToString(search));
            search = algo.BubbleSort(search);
            Console.WriteLine("After:  " + ArrayToString(search));

            Console.WriteLine();
            Console.WriteLine(
                "Doing a binary search with a sorted array of integers:");
            Console.WriteLine("Array: " + ArrayToString(search));
            result = algo.BinarySearch(search, value);
            if (result < 0)
                Console.WriteLine("No occurrence of " + value
                    + " was not found in the search array");
            else
                Console.WriteLine(value + " was found at index " + result
                    + " in the search array");
        }

        /**
         * Create an array of the given size with random values.
         */
        static int[] RandomArray(uint size, int min, int max, Random random)
        {
            int[] array = new int[size];
            for (uint i = 0; i < size; i++) array[i] = random.Next(min, max + 1);
            return array;
        }

        /**
         * Convert an array of integers to a string.
         */
        static String ArrayToString(int[] list)
        {
            StringBuilder sb = new StringBuilder("{" + list[0]);
            for (uint i = 1; i < list.Length; i++) sb.Append(", " + list[i]);
            sb.Append("}");
            return sb.ToString();
        }
    }
}
```

```xml
<?xml version="1.0"?>
<!--
    Note: As an alternative to hand editing this file you can use the
    web admin tool to configure settings for your application. Use
    the Website->Asp.Net Configuration option in Visual Studio.
    A full list of settings and comments can be found in
    machine.config.comments usually located in
    \Windows\Microsoft.Net\Framework\v2.x\Config
-->
<configuration>
  <appSettings/>
  <connectionStrings/>
  <system.web>
    <!--
            Set compilation debug="true" to insert debugging
            symbols into the compiled page. Because this
            affects performance, set this value to true only
            during development.
        -->
    <compilation debug="true"/>
    <!--
            The <authentication> section enables configuration
            of the security authentication mode used by
            ASP.NET to identify an incoming user.
        -->
    <authentication mode="Windows"/>
    <!--
            The <customErrors> section enables configuration
            of what to do if/when an unhandled error occurs
            during the execution of a request. Specifically,
            it enables developers to configure html error pages
            to be displayed in place of a error stack trace.

        <customErrors mode="RemoteOnly" defaultRedirect="GenericErrorPage.htm">
            <error statusCode="403" redirect="NoAccess.htm" />
            <error statusCode="404" redirect="FileNotFound.htm" />
        </customErrors>
        -->
  </system.web>
</configuration>

```

```xml
<?xml version="1.0" encoding="utf-8"?>
<wsdl:definitions xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"
                  xmlns:tm="http://microsoft.com/wsdl/mime/textMatching/"
                  xmlns:soapenc="http://schemas.xmlsoap.org/soap/encoding/"
                  xmlns:mime="http://schemas.xmlsoap.org/wsdl/mime/"
                  xmlns:tns="http://www.uis.no/ws/"
                  xmlns:s="http://www.w3.org/2001/XMLSchema"
                  xmlns:soap12="http://schemas.xmlsoap.org/wsdl/soap12/"
                  xmlns:http="http://schemas.xmlsoap.org/wsdl/http/"
                  targetNamespace="http://www.uis.no/ws/"
                  xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/">
  <wsdl:documentation xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/">
    Sorting and searching algorithm service.</wsdl:documentation>
  <wsdl:types>
    <s:schema elementFormDefault="qualified" targetNamespace="http://www.uis.no/ws/">
      <s:element name="BubbleSort">
        <s:complexType>
          <s:sequence>
            <s:element minOccurs="0" maxOccurs="1" name="list" type="tns:ArrayOfInt" />
          </s:sequence>
        </s:complexType>
      </s:element>
      <s:complexType name="ArrayOfInt">
        <s:sequence>
          <s:element minOccurs="0" maxOccurs="unbounded" name="int" type="s:int" />
        </s:sequence>
      </s:complexType>
      <s:element name="BubbleSortResponse">
        <s:complexType>
          <s:sequence>
            <s:element minOccurs="0" maxOccurs="1" name="BubbleSortResult"
                       type="tns:ArrayOfInt" />
          </s:sequence>
        </s:complexType>
      </s:element>
      <s:element name="Swap">
        <s:complexType>
          <s:sequence>
            <s:element minOccurs="0" maxOccurs="1" name="list" type="tns:ArrayOfInt" />
            <s:element minOccurs="1" maxOccurs="1" name="a" type="s:int" />
            <s:element minOccurs="1" maxOccurs="1" name="b" type="s:int" />
          </s:sequence>
        </s:complexType>
      </s:element>
      <s:element name="SwapResponse">
        <s:complexType>
          <s:sequence>
            <s:element minOccurs="0" maxOccurs="1" name="SwapResult"
                       type="tns:ArrayOfInt" />
          </s:sequence>
        </s:complexType>
      </s:element>
      <s:element name="LinearSearch">
        <s:complexType>
          <s:sequence>
            <s:element minOccurs="0" maxOccurs="1" name="list" type="tns:ArrayOfInt" />
            <s:element minOccurs="1" maxOccurs="1" name="value" type="s:int" />
          </s:sequence>
        </s:complexType>
      </s:element>
      <s:element name="LinearSearchResponse">
        <s:complexType>
          <s:sequence>
            <s:element minOccurs="1" maxOccurs="1" name="LinearSearchResult"
                       type="s:int" />
          </s:sequence>
        </s:complexType>
      </s:element>
      <s:element name="BinarySearch">
        <s:complexType>
          <s:sequence>
            <s:element minOccurs="0" maxOccurs="1" name="list" type="tns:ArrayOfInt" />
            <s:element minOccurs="1" maxOccurs="1" name="value" type="s:int" />
          </s:sequence>
        </s:complexType>
      </s:element>
      <s:element name="BinarySearchResponse">
        <s:complexType>
          <s:sequence>
            <s:element minOccurs="1" maxOccurs="1" name="BinarySearchResult"
                       type="s:int" />
          </s:sequence>
        </s:complexType>
      </s:element>
    </s:schema>
  </wsdl:types>
  <wsdl:message name="BubbleSortSoapIn">
    <wsdl:part name="parameters" element="tns:BubbleSort" />
  </wsdl:message>
  <wsdl:message name="BubbleSortSoapOut">
    <wsdl:part name="parameters" element="tns:BubbleSortResponse" />
  </wsdl:message>
  <wsdl:message name="SwapSoapIn">
    <wsdl:part name="parameters" element="tns:Swap" />
  </wsdl:message>
  <wsdl:message name="SwapSoapOut">
    <wsdl:part name="parameters" element="tns:SwapResponse" />
  </wsdl:message>
  <wsdl:message name="LinearSearchSoapIn">
    <wsdl:part name="parameters" element="tns:LinearSearch" />
  </wsdl:message>
  <wsdl:message name="LinearSearchSoapOut">
    <wsdl:part name="parameters" element="tns:LinearSearchResponse" />
  </wsdl:message>
  <wsdl:message name="BinarySearchSoapIn">
    <wsdl:part name="parameters" element="tns:BinarySearch" />
  </wsdl:message>
  <wsdl:message name="BinarySearchSoapOut">
    <wsdl:part name="parameters" element="tns:BinarySearchResponse" />
  </wsdl:message>
  <wsdl:portType name="AlgorithmServiceSoap">
    <wsdl:operation name="BubbleSort">
      <wsdl:documentation xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/">
        Sorts a list using the bubble sort algorithm.
      </wsdl:documentation>
      <wsdl:input message="tns:BubbleSortSoapIn" />
      <wsdl:output message="tns:BubbleSortSoapOut" />
    </wsdl:operation>
    <wsdl:operation name="Swap">
      <wsdl:documentation xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/">
        Swaps the values at index a and b in the list.
      </wsdl:documentation>
      <wsdl:input message="tns:SwapSoapIn" />
      <wsdl:output message="tns:SwapSoapOut" />
    </wsdl:operation>
    <wsdl:operation name="LinearSearch">
      <wsdl:documentation xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/">
        Does a linear search for a specified value through either an unsorted or a
        sorted list.
      </wsdl:documentation>
      <wsdl:input message="tns:LinearSearchSoapIn" />
      <wsdl:output message="tns:LinearSearchSoapOut" />
    </wsdl:operation>
    <wsdl:operation name="BinarySearch">
      <wsdl:documentation xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/">
        Does a binary search for a specified value through a sorted list.
      </wsdl:documentation>
      <wsdl:input message="tns:BinarySearchSoapIn" />
      <wsdl:output message="tns:BinarySearchSoapOut" />
    </wsdl:operation>
  </wsdl:portType>
  <wsdl:binding name="AlgorithmServiceSoap" type="tns:AlgorithmServiceSoap">
    <soap:binding transport="http://schemas.xmlsoap.org/soap/http" />
    <wsdl:operation name="BubbleSort">
      <soap:operation soapAction="http://www.uis.no/ws/BubbleSort"
                      style="document" />
      <wsdl:input>
        <soap:body use="literal" />
      </wsdl:input>
      <wsdl:output>
        <soap:body use="literal" />
      </wsdl:output>
    </wsdl:operation>
    <wsdl:operation name="Swap">
      <soap:operation soapAction="http://www.uis.no/ws/Swap"
                      style="document" />
      <wsdl:input>
        <soap:body use="literal" />
      </wsdl:input>
      <wsdl:output>
        <soap:body use="literal" />
      </wsdl:output>
    </wsdl:operation>
    <wsdl:operation name="LinearSearch">
      <soap:operation soapAction="http://www.uis.no/ws/LinearSearch"
                      style="document" />
      <wsdl:input>
        <soap:body use="literal" />
      </wsdl:input>
      <wsdl:output>
        <soap:body use="literal" />
      </wsdl:output>
    </wsdl:operation>
    <wsdl:operation name="BinarySearch">
      <soap:operation soapAction="http://www.uis.no/ws/BinarySearch"
                      style="document" />
      <wsdl:input>
        <soap:body use="literal" />
      </wsdl:input>
      <wsdl:output>
        <soap:body use="literal" />
      </wsdl:output>
    </wsdl:operation>
  </wsdl:binding>
  <wsdl:binding name="AlgorithmServiceSoap12" type="tns:AlgorithmServiceSoap">
    <soap12:binding transport="http://schemas.xmlsoap.org/soap/http" />
    <wsdl:operation name="BubbleSort">
      <soap12:operation soapAction="http://www.uis.no/ws/BubbleSort"
                        style="document" />
      <wsdl:input>
        <soap12:body use="literal" />
      </wsdl:input>
      <wsdl:output>
        <soap12:body use="literal" />
      </wsdl:output>
    </wsdl:operation>
    <wsdl:operation name="Swap">
      <soap12:operation soapAction="http://www.uis.no/ws/Swap"
                        style="document" />
      <wsdl:input>
        <soap12:body use="literal" />
      </wsdl:input>
      <wsdl:output>
        <soap12:body use="literal" />
      </wsdl:output>
    </wsdl:operation>
    <wsdl:operation name="LinearSearch">
      <soap12:operation soapAction="http://www.uis.no/ws/LinearSearch"
                        style="document" />
      <wsdl:input>
        <soap12:body use="literal" />
      </wsdl:input>
      <wsdl:output>
        <soap12:body use="literal" />
      </wsdl:output>
    </wsdl:operation>
    <wsdl:operation name="BinarySearch">
      <soap12:operation soapAction="http://www.uis.no/ws/BinarySearch"
                        style="document" />
      <wsdl:input>
        <soap12:body use="literal" />
      </wsdl:input>
      <wsdl:output>
        <soap12:body use="literal" />
      </wsdl:output>
    </wsdl:operation>
  </wsdl:binding>
  <wsdl:service name="AlgorithmService">
    <wsdl:documentation xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/">
      Sorting and searching algorithm service.</wsdl:documentation>
    <wsdl:port name="AlgorithmServiceSoap" binding="tns:AlgorithmServiceSoap">
      <soap:address
        location="http://localhost:1370/AlgorithmService/AlgorithmService.asmx" />
    </wsdl:port>
    <wsdl:port name="AlgorithmServiceSoap12" binding="tns:AlgorithmServiceSoap12">
      <soap12:address
        location="http://localhost:1370/AlgorithmService/AlgorithmService.asmx" />
    </wsdl:port>
  </wsdl:service>
</wsdl:definitions>
```