ARM9-based implementation of Using EVC to Access Remote WEBSERVICE Interface

HeFu Liu

Informatized Office, Huazhong Normal University, Wuhan 430070, China
liuhf@mail.ccnu.edu.cn

Keywords: WebService interface; ARM9; SOAP; XML; WSDL; Client-Side Data Flow

Abstract: The WebService interface can realize communication between different applications and development in different system platform application. EVC can access remote WebService interface on other services and easily Implement the interface on ARM9. In this way, the WINDOW CE.NET system running on the ARM9 will achieve WebService interface where a remote server provides the server function. To achieve the above functions, we can use the following SOAP object: ISoapConnector, ISoapReade and ISoapSerializer, and can achieve satisfactory results.

Introduction

The WebService interface can realize communication between different applications and development in different system platform application. WebService uses SOAP[1] as the basic communication protocol so as to avoid the complex protocol conversion. Any device which can support for HTTP and XML, can have the way of accessing to the WebService interface. Microsoft® eMbedded Visual C++® 4.0(EVC)[2] provides an integrated development environment (IDE) and tools for developers who want to develop WebService applications for Windows CE .NET. Obviously, EVC can access remote WebService interface on other services and easily Implement the interface on ARM9[2]. This article walks you through the process of creating a client SOAP-based WebService server by means of SOAP on EVC .

Implementation Process of SOAP, XML and WDSL Protocol

A. Implementation of SOAP

SOAP[3], originally an acronym for Simple Object Access protocol, is a protocol specification for exchanging structured information in the implementation of web services in computer networks. SOAP can form the foundation layer of a web services protocol stack, providing a basic messaging framework for web services. The client-side SOAP of WebService allows an application to invoke Web service operations, while the server-side functionality maps invoked Web service operations to Component Object Model object method calls. ARM9 embedded system can support SOAP with two widely used protocols: HTTP[4] and XML[6]. HTTP is used to transfer the SOAP RPC style, code information and parameters that can be set to SOAP by XML. The SOAP communication protocol uses HTTP to send the information in XML format. In this way, the WINDOW CE.NET system running on the ARM9 will achieve WebService interface, where a remote server provides the server function. To achieve the above functions, we can use the following SOAP object: ISoapConnector[5], ISoapReader[5] and ISoapSerializer[5] , and can achieve satisfactory results.

Programming element	Description
ISoapConnector	This interface implements the transport protocol used by objects that send and receive SOAP messages.
ISoapReader	This interface reads SOAP messages.
ISoapSerializer	The interface builds a SOAP message.

The following table shows the SOAP objects and interfaces with a description of the purpose of each.

B. Implementation of XML and WSDL

Soap XML[6] message is from Web services Definition Language(WSDL)[6] . Web services uses XML encoding and decoding the data, and use SOAP to transmit data. Throug these XML documents, We can obtain many WEBSERVICE interface information, includes: access address, parameters etc.. A SOAP message is an ordinary XML document containing the following elements: An Envelope element that identifies the XML document as a SOAP message....

The Web Services Description Language (WSDL)[4] is an XML message format for describing the network services offered by the server. You use WSDL to create a file that identifies the services provided by the server and the set of operations within each service that the server supports. For operation, the WSDL file also describes the format that the client must follow in requesting an operation. Because the WSDL file sets up requirements for both server and client, this file is like a contract: The server agrees to provide certain services if the client sends a properly formatted SOAP request.

Suppose a WSDL file defines a service called AddNum. This service describes operations such as the input number of A, B and Result output. You place this file on the Internet Information Server (IIS) and will get the XML message, which can provide all kinds of service interface and message.

Executing Process of the Client-data flow

A. Client-Side Data Flow

The user application of WebService sends a message to the client Soap object using the WebService client Soap interface. These interfaces include ISoapConnector, ISoapSerializer and ISoapReader . We interact with the client Soap object solely through the interfaces. A client who wants to send a SOAP request to the server first obtains a copy of this WSDL file from the server. The client then uses the information in this file to format a SOAP request. The client sends this request to the server. The server executes the requested operation and sends the resulting messages back to the client as a SOAP response.

The WebService message sent by the user is an operation; in this example, the operation is adding two numbers, as follows:

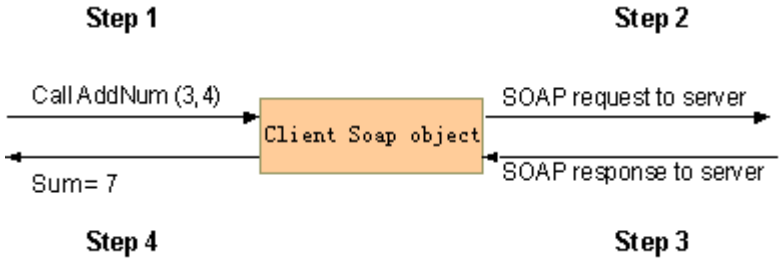
1. The client Soap object processes this function request and formulates a SOAP request to the WebService server.

2. The server receives the request and performs the requested operation.

3. The server sends the result of the operation back to the client in the form of a SOAP response XML document.

4. The client Soap object processes this SOAP response and sends a message that contains the function result of the operation back to the user application.

The following diagram illustrates how SOAP handles data flow between the client and the server.



B. Setting Up the Client Soap Object of WebService

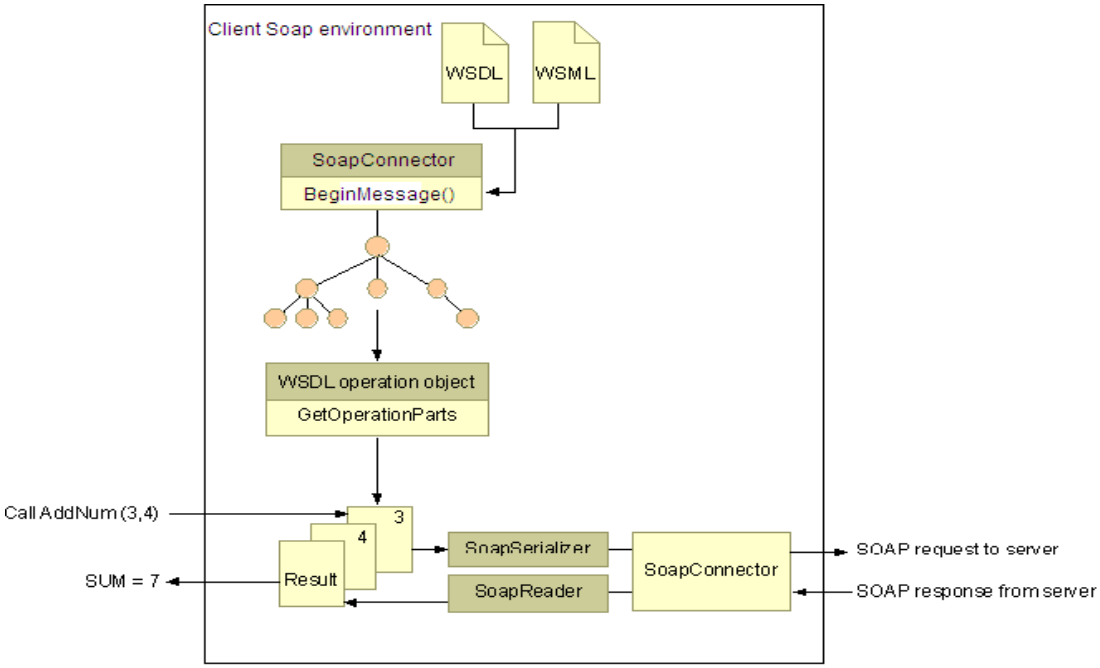
Before sending different operation requests using the Client Soap interface, the user application must first instantiate the Soap object that implements this interface. The Soap object is implemented in MSSOAP1.dll[3].This function CoCreateInstance[2] can create on the local system a single uninitialized Soap object of the class associated with a specified Soap class identifier. As such, it encapsulates the following functionality:

●HRESULT hr1=CoCreateInstance(CLSID_HttpConnector, NULL, CLSCTX_INPROC_SERVER,IID_ISoapConnector, (void **)&pConnector)
●HRESULT hr2=CoCreateInstance(CLSID_SoapSerializer, NULL,CLSCTX_INPROC_SERVER, IID_ISoapSerializer,(void **)&pSerializer)
●HRESULT hr3=CoCreateInstance(CLSID_SoapReader,NULL, CLSCTX_INPROC_SERVER,IID_ISoapReader,(void **)&pReader)

C. Processing Inside the Client Soap Object

In your application, you use the Client Soap interface to send SOAP requests; then the Client Soap object does several things internally to send the request to the server.

The following diagram illustrates how the Soap Client object processes the operation request from the user application, formulates a SOAP request message, and processes a SOAP response message.



Here is what happens:

1. According to the WSDL address of the XML file, you can get to the parameters required by the WebService interface, including: MethodName[6], EndPoinUrl[6],, Namespace[6],, Soapaction[6],, and the other required parameters.
2. The SoapConnector object set the property. This property sets or gets the value of a property specific to an SoapConnector implementation from the WSDL and Web Services Meta Language (WSML) files .
3. Based on this analysis, the SoapConnector object gets URI of a Soap action for the requested operation, including the requested function (for example, AddNum(3, 4)).
4. The SoapConnector object calls the BeginMessage[5] method, which signals the start of a SOAP message being sent to the server and calls the get_InputStream[5] method, which gets the input stream of the process from XML packet.
5. The SoapSerializer object (accessed through the ISoapSerializer[7] interface) builds the SOAP request message and sends it to the server. The SoapSerializer object calls startElement[5] method,endElement[5] and etc. These methods begin and end an entry (child element) in the <Body> element of a SOAP message[7].
6. The server processes the SOAP request and returns a SOAP response to the client. From this response, the SoapReader object loads the result of the operation into the output stream data of SoapConnector object and returns the result to the user application.

Internally, all methods and Parameters described in the WSDL file are bound dynamically to the Client Soap interface during initialization. This dynamic binding allows you to call any method described in the WSDL file. The above idea is of course easy to use EVC to access remote WebService Interface. The WebService interface to realize the idea can bring great convenience to our work on ARM9.

References

[1] http://spring.io/guides/gs/producing-web-service/

[2] Wangbing, Licehnbin, Embedded Visual C++ Embedded Programming,. Waterpub
[3] W3C.SOAP specifications [EB/OL].[2011-012-13] .
http://www.w3.org/TR/2011/REC-ws-soap-assertions-20111213.
[4] W3C.Hypertext transfer protlcol-HTTP/1.1.rfc .
http://www.w3.org/Protocols/HTTP/1.1/rfc2616bis/draft-lafon-rfc2616bis-02.html
[5] https://msdn.microsoft.com/en-us/library/ms123401.aspx
[6] http://www.360doc.com/content/09/1210/13/495229_10785010.shtml
[7] http://en.wikipedia.org/wiki/SOAP