WEB SERVICE ENCAPSULATION OF FORTRAN-BASED GEOGRAPHICAL MODEL

Xiaolin Wang, Haibo Wang, Hao Deng, Yingwei Luo
Dept. of Computer Science and Technology, Peking University, Beijing, China, 100871
E-mail: lyw@pku.edu.cn

1. INTRODUCTION

Geographical modeling work is a long term and basic research work. The geologists who come from different fields have established a large number of geographical models. However, these model resources are scattered in different agencies and different fields of scientists, and so far, there isn't an effective way to share and distributed integrated use of these resources in the Internet environment. This has result to the isolation of geographical models problem. In order to solve this problem and realize the reuse, share, and integration of models in the Web, it is necessary to establish a distributed geographical modeling environment. This environment is mainly based on the Internet, and makes use of distributed computing technology. To achieve such an environment, we should study the methods of interpreting and reusing of geographical model resources on the cyberspace, explore the Web service encapsulation mechanism of geographical models, and finally, realize a multi-source and heterogeneous environment which can effectively share geographical model resources. By making use of the Service-Oriented Architecture (SOA), we can decouple a tightly coupled closed system to a loosely coupled open system in the network space, and making it possible to get a unified model of the geographical models of expressing and sharing. As a kind of architecture, SOA is independent of any particular technology, while its implementation is mostly based on the web service currently.

SWAT is the acronym for Soil and Water Assessment Tool, which is a river basin, or watershed, scale model developed by Agricultural Research Service (ARS) of USDA (United States Department of Agriculture). SWAT was developed to predict the impact of land management practices on water, sediment and agricultural chemical yields in large complex watersheds with varying soils, land use and management conditions over long periods of time. The SWAT is a typical structural program written by Fortran language. Different modules in SWAT share common data by global variables, and there is a tightly coupled relationship between modules.

In order to establish a distributed environment of geographical model, first of all we need to build a model library. A important source of models in the library is from the deconstruction, split, and encapsulation of the already existed models. Once the model library is established, we can get a new model by replacing some of the sub-modules in an original model by using models in the library. This paper discusses how to split the climate simulator out of the SWAT module, encapsulate it into a web service, and to serve as a model in the model library.

2. FORTRAN PROGRAM ENCAPSULATION TECHNOLOGY

SWAT is written by Fortran language, if we want to encapsulate some of its modules into Web Services, we can use the following method: First, using C language to encapsulate the

Fortran source code of one module, then use Java to encapsulate the C program, last we can encapsulate the Java program into a Web Service (shown in Figure 1).

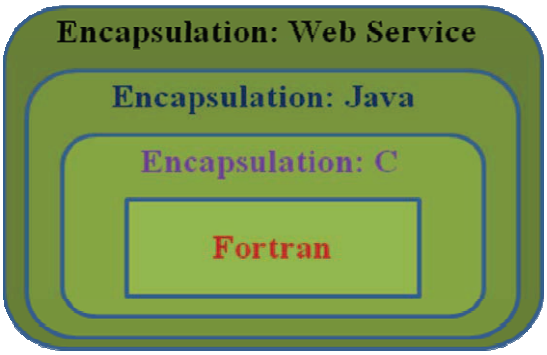

Figure 1 Encapsulating a Fortran program into a Web Service

2.1 C And Fortran Mixed Programming

The process of using C encapsulate the Fortran source code involves C and Fortran mixed programming. Fortran language and C language are both compiled languages, their syntax rules, data structures, etc, are mostly the same like. When we use these two languages to mixed programming, pay special attention to the following:

1) Fortran stores multi-dimensional arrays as a continues linear sequence of elements. It is important to know that multi-dimensional arrays are stored by column. That means the first column will stored first, and then follows the second column, the third column, and so on. While C language stores multi-dimensional arrays by row. When Fortran and C communicate with multi-dimensional arrays as parameters in mixed programming, it is necessary to reverse the suffix, such as in Fortran array element a(i, j) should be converted to be a(j, i) in C.

2) The string in Fortran and C is totally different. In Fortran, a string has no ending tag, while in C, character '\0' means the end of a string. So, when communicate with a string as parameter, it is necessary to convert the string format.

3) In Fortran, the passing parameters in procedure or function is Call-by-reference, while in C there are both Call-by-reference and Call-by-value, so pay attention to the difference when passing parameters in the mixed programming.

2.2 Java And C Mixed Programming

Because of the Java Native Interface (JNI), Java and C can communicate in the mixed programming. Since Java1.1 version, JNI had been part of the standard Java platform, which makes the communication between Java and the other language possible. However, make Java program communicating with the native code, will lost the independent of platform which is a very important feature that Java is better than other languages. But, in some situations, this is acceptable, or has no alternative. For example, making use of some old library, interacting with the hardware or operation system, improving the performance of procedures, and encapsulating as in this paper. JNI standard can at least assure that the native code can work in any Java Virtual Machine (JVM) effectively.

3. SWAT MODULE AND ITS ENCAPSULATION EXPERIMENT

SWAT contains 8 modules, they are: climate simulation module, hydrology module, nutrients/pesticides module, erosion module, land cover/plant module, management practices

module, main channel processes module, and water body module.

In SWAT, the climate simulation module is relatively independent which has a low coupled with other modules, in other words, there is no function calls with other modules. As a result, this module is suitable for being put out of the origin model and encapsulating it into a Web Service. In the distributed geographical model environment, we can use another climate simulator to replace this one in SWAT, thus get a new kind of SWAT model.

The entire SWAT program contains more than 1000 global variables, and every module contains more than 100 global variables in average[1]. In order to encapsulate a module into a Web Service, two problems are concerned:

1) As we put it, there are more than 100 global variables in a module, with so many variables, how to pass their values?

2) In a module, there are a number of places involves reading file operation, and different time read different part of the file, how we can assure this can work properly after we encapsulate this module into a Web Service?

For the first problem, it is infeasible to use the function parameters approach to pass variables' value, because for every function call, we must write a lot of parameters and must ensure that these parameters are all in the corresponding position, this is cumbersome and error-prone. Besides, the language can only accept limited number of parameters. Except the function parameters approach, there are two other ways to solve this problem. One is make use of the common variables between Fortran and C, the second is by writing all the variables and their values into a file according to some predefined format. Here we use the second method, and predefined the data storage format, an adapter will generate or interpret data strictly according the format.

For the second problem, we can pass file pointer parameters to solve it. For the invoker, every time it calls the service, it will pass the corresponding file name and the position information as parameters to the service, so as to realize reading data correctly. For some metadata in an input file which just relates with the module service, we can extract these data out of the input file encapsulate them with the module, this will improve the efficiency of data passing.

We take an experiment to encapsulate the *climate simulation module* in SWAT into a Web Service. First we give an overview of the encapsulation process, and then discuss the data format which is a key part in this experiment.

To encapsulate *climate simulation module* into a Web Service, we use the file to pass a large number of global variable data, the concrete steps are as follows:

1) Use Fortran language writes a adapter, which in charge of interpreting all the global variable parameters' value from the input file and generate an output file of global variable data. The input file and output file's formats are predefined strictly.

2) Use C language write a program, which receives the input data file name, calls the above adapter, and returns the output data file name.

3) Use Java language write a program, which receives the input data file name, calls the above C program, and returns the output data file name.

4) Encapsulate the above Java program into a Web Service.

The data passed in the process of encapsulation includes global variable data and file pointer parameters, we use XML file to store and transport them (shown in Figure 2).

[1] We've built a tool which could extract all the variables in a module automatically.

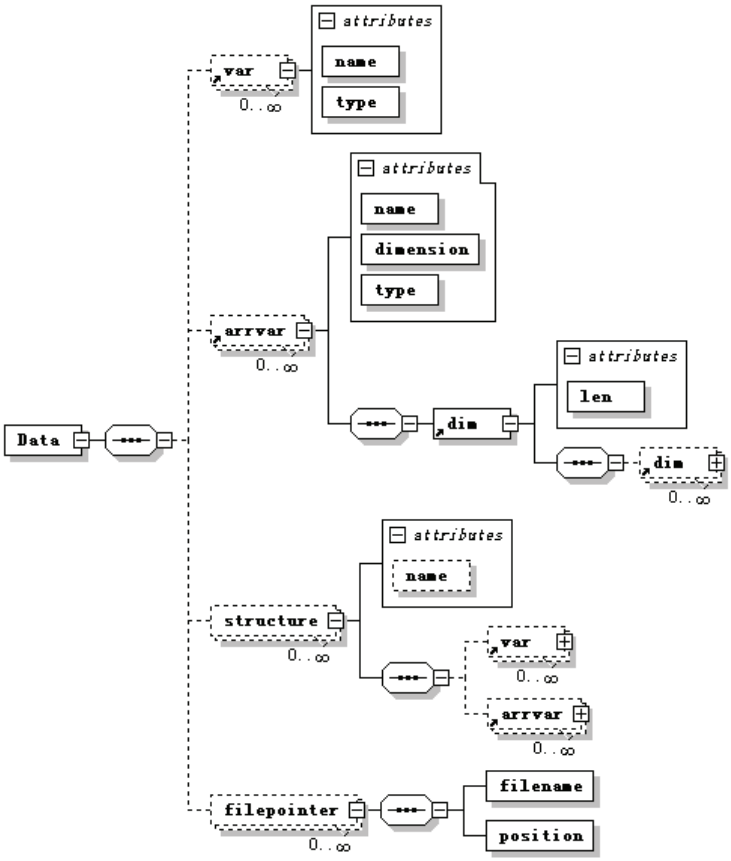

Figure 2 XML Data Format Schema

According to the above XML data format, the adapter is in charge of generating the output data file and interpreting the input data file, so as to achieve the purpose of transferring data.

4. CONCLUSION

This paper encapsulates the climate simulation module in SWAT into a Web Service, which provides a module element for the distributed geographical model environment, and this is the basis of construct a new geographical model. The encapsulation method is suitable for modules which have a great many parameters and internal file reading.

5. REFERENCES

[1] S.L. Neitsch, J.G. Arnold, etc, Soil and Water Assessment Tool theoretical documentation, version 2005, http://www.brc.tamus.edu/swat/doc.html, 2009-4-14

[2] S. Comella-Dorda, K. Wallnau, R.C. Seacord, J. Robert, A survey of black-box modernization approaches for information systems, Proceedings of the International Conference on Software Maintenance, IEEE CS Press, Los Alamitos, CA, 2000, pp.173-183.

[3] G. Canfora, A.R. Fasolino, etc., Migrating Interactive Legacy Systems To Web Services, Proceedings of the 10th European Conference Software Maintenance and Reengineering (CSMR 2006), 2006.

[4] H. M. Sneed, S.H. Sneed, Creating Web services from legacy host programs, Proceedings of the 5th IEEE International Workshop on Web Site Evolution, WSE 2003, IEEE CS Press, Los Alamitos, CA, 2003, pp.59-65.