Applicability of Cyclomatic Complexity on WSDL

Sanjay Misra[1], and Adewole Adewumi[1]

[1] Department of Computer and Information Sciences, Covenant University, Ota, Nigeria
{sanjay.misra, wole.adewumi}@covenantuniversity.edu.ng

Abstract. Complexity metrics are used to predict the quality of the any software systems because they quantify the quality attributes. Web services, a new type of applications, have been providing a common standard mechanism for interoperable integration of disparate systems and gaining a great deal of acceptance by different types of parties that are connected to the internet for different purposes. In this respect, quality of the web-services should be quantified for easy maintenance and quality of services. Further, the Web Services Description Language (WSDL) forms the basis for Web Services. In this paper, we are evaluating the quality of the WSDL documents by applying the Cyclomatic complexity metric, a well known and effective complexity metric, which is not used to evaluate the quality of WSDL till date.

Keywords: WSDL, web-services, software metrics, cyclomatic complexity

1 Introduction

Software metrics have always been important for software engineers to assure software quality. However, absolute measures are uncommon in software engineering [9]. Instead, software engineers attempt to derive a set of indirect measures that lead to metrics that provide an indication of quality of some representation of software. Software engineers plan 'how' an information system should be developed in order to achieve its quality objectives. The quality objectives may be listed as performance, reliability, availability and maintainability [10], and are closely related to software complexity.

The researches on web services are in developing stage. The existing standards are under revisions and new technologies are evolving. Web services can be treated as a software component, whose data can be presented using XML Schema, and their interfaces through Web Service Definition Language (WSDL [1]). Web Services Description Language (WSDL) is an eXtensible Markup Language (XML) based document and also a language which continuously gaining popularity for web services. Since WSDL is a language, therefore its quality should be quantified.

Software metrics always play an important role in developing software. Further, their importance increases more when, the companies and organizations are adopting the new Technologies. In recent years, Web-services, a new type of application, was adopted by several companies, such as IBM, Microsoft, BEA, Oracle, Borland, etc. [3]. However, the quality of web services is not been evaluated through the applicability of very well known software metrics like cyclomatic complexity,

Halsted programming efforts. Cyclomatic complexity is one of the most popular and a well-accepted complexity metric, which is in use since last three decades.

Since the degree in complexity has effect on maintaining any kind of software project we studied in developing metrics to measure the complexity of a Web service. The complexity degree of a Web service can be measured by analyzing its WSDL [4], [5], [6], [7] document because WSDL provides the description of the Web service to the service requestor's software development [8].

Actually, our motivation for the applicability of cyclomatic complexity arises from the paper of Yijun et al. [3]. They argued that since there are no implementation details of WSDL, we cannot measure the metrics that are based on the complete source code, such as McCabe complexity [2] that measures the controls flow. However, in this paper we are showing that the cyclomatic complexity can be used to evaluate the WSDL.

2 Web Services and WSDL

The following figure shows how WSDL is used in web services.

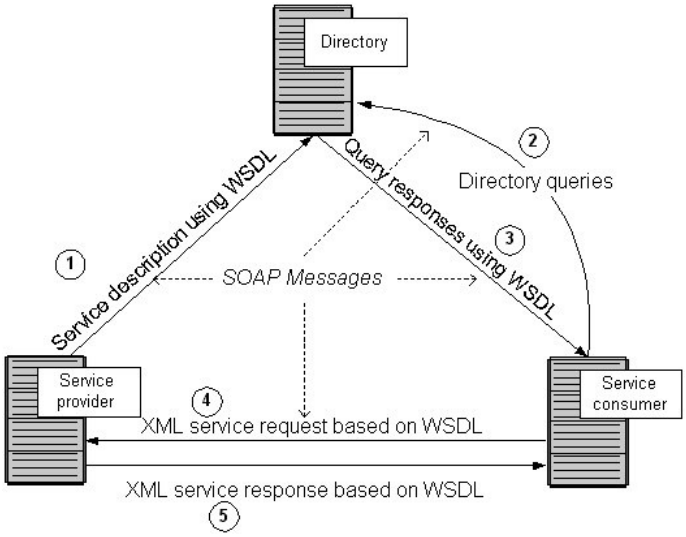


Fig. 1. This figure shows the use of WSDL in Web Services [11].

The Web Services Description Language (WSDL) is the foundation layer of the Web Services. We can easily observe from the figure that how at the two ends, service provider and service consumers are communicating through WSDL.

3 Cyclomatic Complexity

McCabe [2] developed a graph theoretic complexity measure for managing and controlling program complexity. The metric is independent of physical size and depends only on decision structure of a program and hence is calculated from its flow graph representation. Further, if a program has the number of modules then its complexity is calculated as the sum of cyclomatic complexity of modules calculated individually. This measure finds the number of linearly independent circuits that are required to provide information about all circuits and thus attempts, to determine the number of execution paths in a program. The number of fundamental circuit also acts as an index of program testing effort. This measure is most widely used one amongst the different available measure for control flow complexity.

The cyclomatic complexity of a part of code is the count of the number of linearly independent paths through the code. For example, if the code contained no decision points such as 'for' loops, the complexity would be only 1, as there is only one path through the code.

The cyclomatic *complexity* of a structured program is defined with reference to a directed graph containing the basic blocks of the program, with an edge between two basic blocks if control may pass from the first to the second (the *control flow graph* of the program). The complexity is then defined as:

$$CC = E - N + 2P \quad (1)$$

Where,
CC = Cyclomatic complexity
E = the number of edges of the graph
N = the number of nodes of the graph
P = the number of connected components.

4 Demonstration of CC for WSDL

4.1 Demonstration of Nodes in WSDL

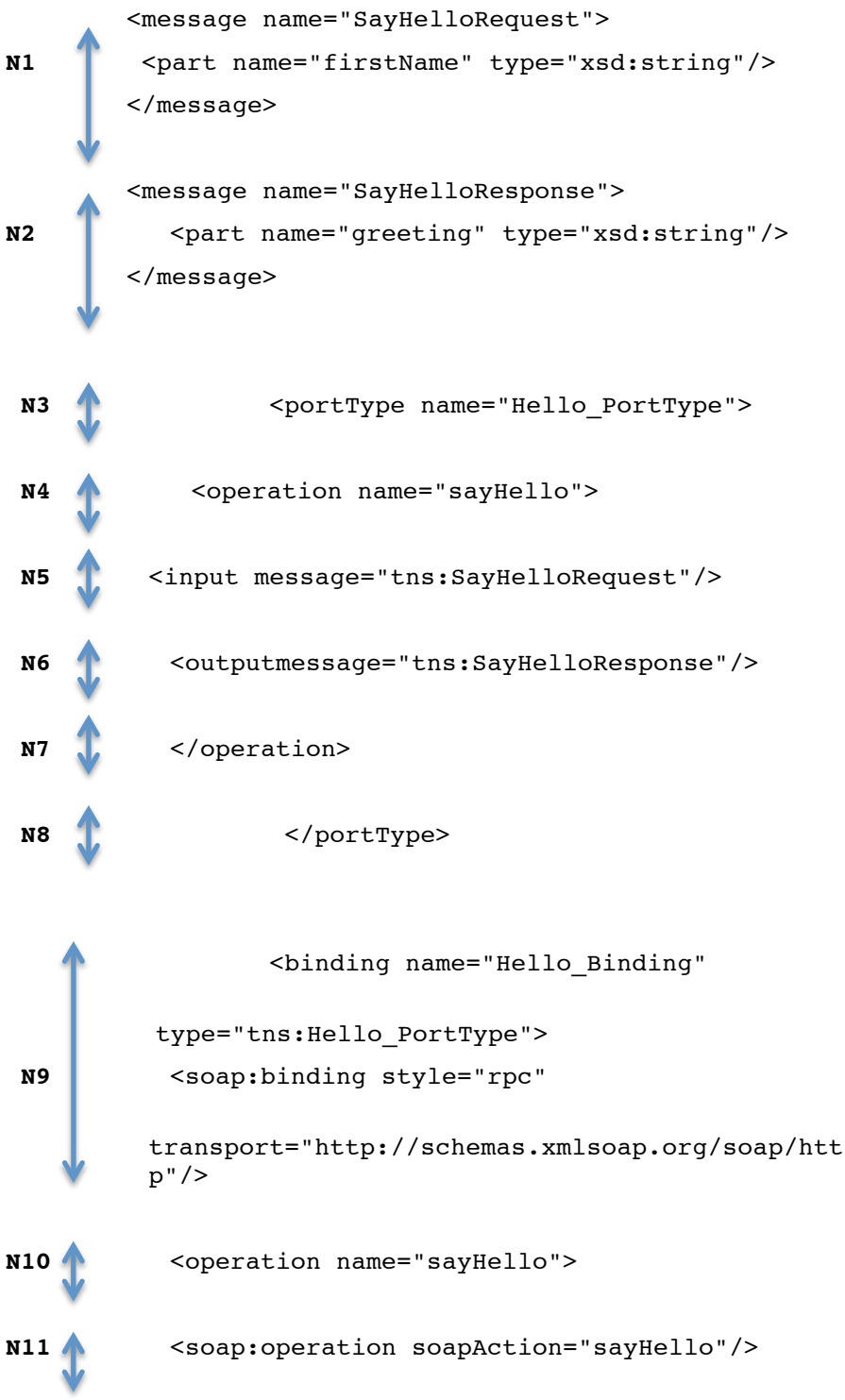
```
<?xml version="1.0" encoding="UTF-8"?>
```

```
  <definitions name="HelloService"
targetNamespace=http://www.ecerami.com/wsdl/HelloService.wsdl
xmlns=http://schemas.xmlsoap.org/wsdl/
```

NO

```
xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"
```

```
  xmlns:tns="http://www.ecerami.com/wsdl/HelloService.wsdl"
xmlns:xsd=http://www.w3.org/2001/XMLSchema >
```

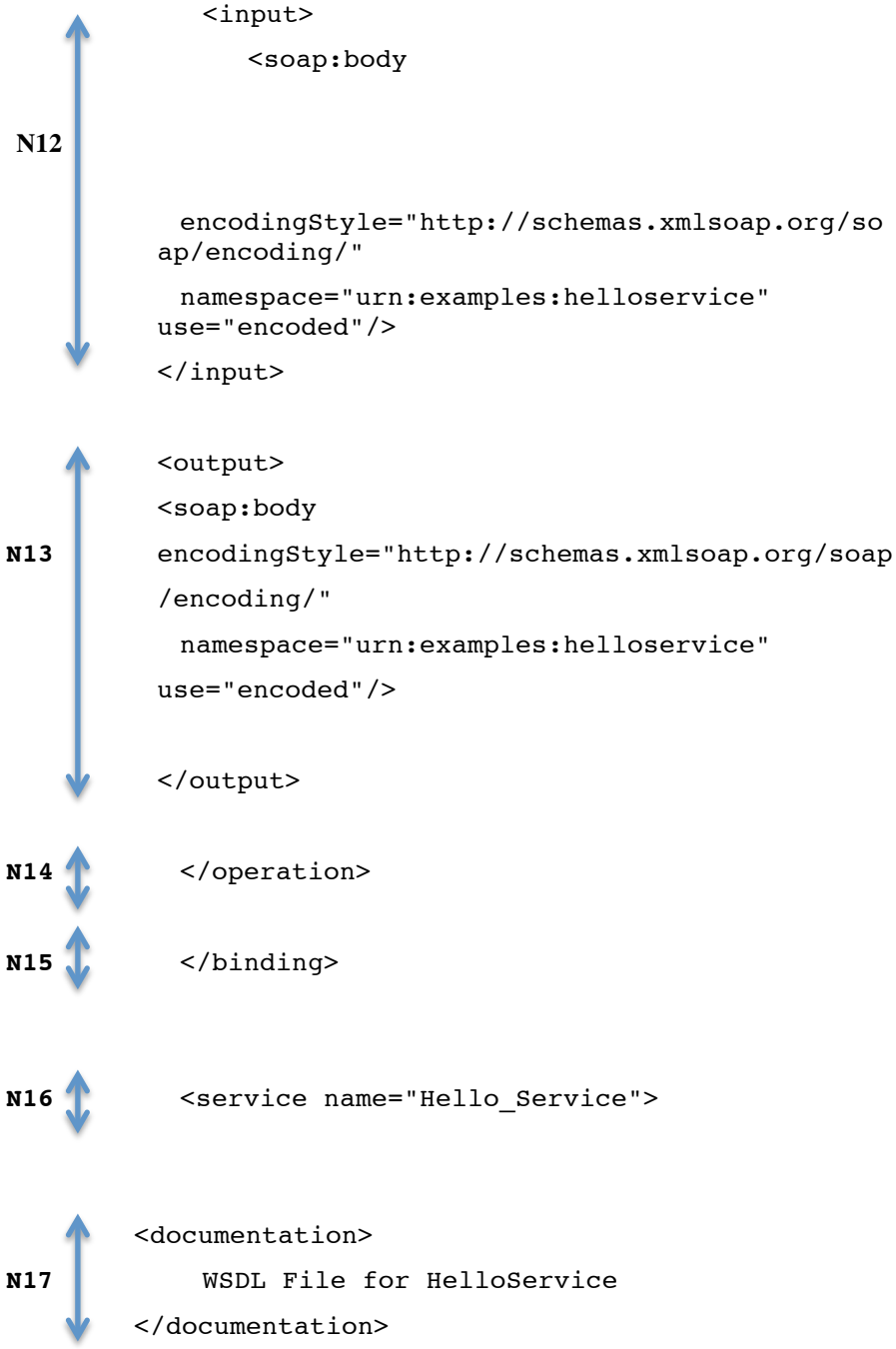



```
<message name="SayHelloRequest">
 <part name="firstName" type="xsd:string"/>
</message>

<message name="SayHelloResponse">
   <part name="greeting" type="xsd:string"/>
</message>

          <portType name="Hello_PortType">
    <operation name="sayHello">
  <input message="tns:SayHelloRequest"/>
   <outputmessage="tns:SayHelloResponse"/>
   </operation>
           </portType>

          <binding name="Hello_Binding"
 type="tns:Hello_PortType">
  <soap:binding style="rpc"
transport="http://schemas.xmlsoap.org/soap/htt
p"/>
   <operation name="sayHello">
   <soap:operation soapAction="sayHello"/>
```




```
    <input>
       <soap:body
 encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
 namespace="urn:examples:helloservice"
use="encoded"/>
</input>

<output>
<soap:body
encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
 namespace="urn:examples:helloservice"
use="encoded"/>

</output>

  </operation>

  </binding>

  <service name="Hello_Service">

<documentation>
    WSDL File for HelloService
</documentation>
```

```
N18   <port binding="tns:Hello_Binding"
        name="Hello_Port">
          <soap:address
        location="http://localhost:8080/soap/servlet/r
        pcrouter"/>
        </port>

N19       </service>

N20   </definitions>
```

4.2 The Graph Representation of the WSDL

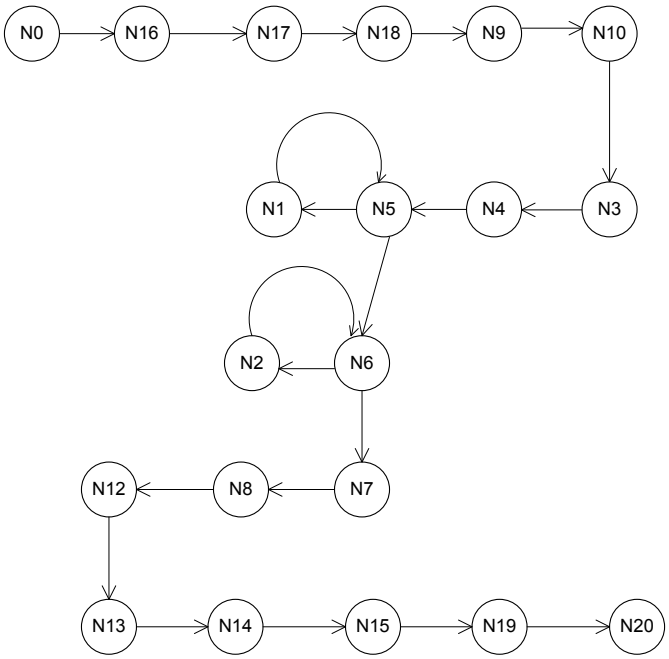


Fig. 1. This figure shows the control flow diagram for WSDL document.

The Cyclomatic Complexity of the given WSDL is given by:

$$CC = E - N + 2P$$

$$CC = 21 - 20 + 2*1$$

$$CC = 3$$

Table 1. Recommended thresholds for Cyclomatic complexity

Cyclomatic Complexity Values	Risk
1 - 10	Low Risk Program
11 - 20	Moderate Risk
21 - 50	High Risk
>50	Most complex and Highly Unstable method

5 Validation

Weyuker proposed nine properties to make judgment about any proposed software complexity measure [12]. These properties evaluate the weaknesses of a proposed measure in a practical way. By their help, one may obtain an idea about the validity of his/her own proposal. In fact, Weyuker herself evaluated the four complexity measures against these properties. Cyclomatic complexity was one of the complexity measures, which was evaluated by Weyuker's properties. Cyclomatic complexity was satisfied by six Weyuker's properties. Actually, the cyclomatic complexity, which was evaluated by Weyuker's properties, was basically for procedural languages. But all the arguments, which were given for the validation of cyclomatic complexity for procedural language, are also valid for the proof of CC for WSDL. In this respect, we can conclude that, our measure is also satisfied by six Weyuker's properties. Here it is worth mentioning that it is not required for any complexity measure to satisfy all Weyker's properties. It is because some of the Weyuker's properties are contradictory in its nature. Therefore, at most six or seven Weyuker's properties satisfy the complexity measures.

6 Conclusion

Evaluating quality of WSDL through software metrics is a new area of research. Since there has been little number of researches so far, adaptation of some existing software metrics to the WSDL is helpful for the evaluation of the quality of WSDL. It is suggested that Cyclomatic complexity metric can be considered in the design, development and operational phase of the web-services.

References

1. W3C, "Web Services Description Language (WSDL) 1.1," 2001

2. McCabe, T.: A Complexity Measure, IEEE Trans. Software Eng., vol. 2 (1976) 308-320
3. Yu, Y., Lu, J., Fernandez-Ramil, J., Yuan, P.: Comparing Web Services with other Software Components, Proc. IEEE International Conference on Web Services (2007) 388-397
4. Erl, T.: Service-Oriented Architecture: A Field Guide to Integrating XML and Web Services, Prentice Hall Publishers (2004)
5. Newcomer, E., Lomow, G.: Understanding SOA with Web Services, Addison Wesley Professional (2004)
6. Cerami, E.: Web Services Essentials, Distributed Applications with XML-RPC, SOAP, UDDI & WSDL, O'Reilly Publishers (2002)
7. Weerawarana, S., Curbera, F., Leymann, F., Storey, T., Ferguson, D. F.: Web Services Platform Architecture: SOAP, WSDL, WS-Policy, WS-Addressing, WS-BPEL, WS-Reliable Messaging, and More, Prentice Hall Publishers (2005).
8. http://www.w3.org/TR/2004/REC-xmlschema-1-20041028/
9. Pressman, R. S.: Software Engineering: A Practitioner's approach, Fifth Edition, McGraw Hill (2001)
10. Sommerville, I.: Software Engineering, Sixth Edition, Addison-Wesley (2001)
11. http://www.service-architecture.com/web-services/articles/web_services_explained.html
12. Weyuker, E.J.,: Evaluating software complexity measure". IEEE Transaction on Software Complexity Measure, vol. 14 (1988) 1357-1365