รายการอ้างอิง

[1] W3C Working Group, Web Services Glossary. Available from: http://www.w3.org/TR/ws-gloss, [Mar 2009]

[2] W3Schools, WSDL Tutorial. Available from: http://www.w3schools.com/WSDL/default.asp, [Mar 2009]

[3] W3C, Web Services Description Language (WSDL) 1.1. Available from: http://www.w3.org/TR/wsdl, [Mar 2009]

[4] Miller, J., Verma, K., Rajasekaran, P., Sheth, A., Aggarwal, R., and Sivashanmugam, K., WSDL-S: Adding Semantics to WSDL - White Paper, 2004.

[5] Tsai, W. T., Paul, R., Wang, Y., Fan, C., and Wang, D., Extending WSDL to facilitate Web services testing. the 7th IEEE International Symposium on High Assurance Systems Engineering, 2002.

[6] W3C Member Submission, Web Service Semantics - WSDL-S. Available from: http://www.w3.org/Submission/WSDL-S/, [Mar 2009]

[7] Bertolino, A., Gao, J., Marchetti, E., and Polini, A., Automatic Test Data Generation for XML Schema-based Partition Testing. the International Workshop on Automation of Software Test, 2007.

[8] W3Schools, XML Tutorial. Available from: http://www.w3schools.com/xml/default.asp, [Mar 2009]

[9] W3C Recommendation, Extensible Markup Language (XML) 1.0, 2004.

[10] W3Schools, XML Schema Tutorial. Available from: http://www.w3schools.com/Schema/default.asp, [Mar 2009]

[11] Protege, SWRL Language FAQ. Available from: http://protege.cim3.net/cgi-bin/wiki.pl?SWRLLanguageFAQ, [Mar 2009]

[12] W3C Member Submission, SWRL: A Semantic Web Rule Language Combining OWL and RuleML, 2004.

[13] Jorgensen., P. C., Software Testing A Craftsman's Approach. CRC PRESS, 2002

[14] Bai, X., Dong, W., Tsai, W.-T., and Chen, Y., WSDL-based automatic test case generation for Web services testing. the IEEE International Workshop on Service-Oriented System Engineering, 2005.

[15] Pancharat, S. and Suwannasart, T., An Approach for Generating Test Cases from Use Cases Based on A Decision Table. the National Computer Science and Engineering Conference, 2006.

[16] Hanna, S. and Munro, M., An Approach for Specification-based Test Case Generation for Web Services. the IEEE/ACS International Conference on Computer Systems and Applications, 2007.

ภาคผนวก

ภาคผนวก ก
รายละเอียดของเว็บเซอร์วิสที่นำมาทดลองเครื่องมือที่พัฒนาขึ้น

ก.1. เว็บเซอร์วิสที่ให้บริการตรวจสอบชนิดของรูปสามเหลี่ยม

```
<?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://tempuri.org/"
        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://tempuri.org/"
        xmlns:wssem="http://lsdis.cs.uga.edu/projects/meteor-s/wsdl-s/examples/WSSemantics.xsd"
        xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/">
 <wsdl:types>
  <s:schema elementFormDefault="qualified" targetNamespace="http://tempuri.org/">
   <s:element name="Triangle">
    <s:complexType>
     <s:sequence>
      <s:element minOccurs="1" maxOccurs="1" name="x" type="s:int" />
      <s:element minOccurs="1" maxOccurs="1" name="y" type="s:int" />
      <s:element minOccurs="1" maxOccurs="1" name="z" type="s:int" />
     </s:sequence>
    </s:complexType>
   </s:element>
   <s:element name="TriangleResponse">
    <s:complexType>
     <s:sequence>
      <s:element minOccurs="0" maxOccurs="1" name="TriangleResult" type="s:string" />
     </s:sequence>
    </s:complexType>
   </s:element>
  </s:schema>
 </wsdl:types>
 <wsdl:message name="TriangleSoapIn">
  <wsdl:part name="parameters" element="tns:Triangle" />
 </wsdl:message>
 <wsdl:message name="TriangleSoapOut">
  <wsdl:part name="parameters" element="tns:TriangleResponse" />
 </wsdl:message>
```

ข้อมูลนำเข้า

ผลลัพธ์

รูปที่ ก.1. รายละเอียดของเอกสารดับเบิลยูเอสดีแอล-เอส
สำหรับเว็บเซอร์วิสที่ให้บริการตรวจสอบชนิดของรูปสามเหลี่ยม

```
<wsdl:portType name="Service1Soap">
 <wsdl:operation name="TriType">
  <wsdl:input message="tns:TriangleSoapIn" />
  <wsdl:output message="tns:TriangleSoapOut" />
  <wssem:precondition name="TrianglePrecond"
        wssem:modelReference="http://localhost/example.swrlx#inputRule1"/>
  <wssem:effect name="ResultEffect"
        wssem:modelReference="http://localhost/example.swrlx#outputRule1
        http://localhost/example.swrlx#outputRule2
        http://localhost/example.swrlx#outputRule3
        http://localhost/example.swrlx#outputRule4
        http://localhost/example.swrlx#outputRule5
        http://localhost/example.swrlx#outputRule6"/>
 </wsdl:operation>
</wsdl:portType>
<wsdl:binding name="Service1Soap" type="tns:Service1Soap">
 <soap:binding transport="http://schemas.xmlsoap.org/soap/http" />
 <wsdl:operation name="Triangle">
  <soap:operation soapAction="http://tempuri.org/Triangle" 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="Service1Soap12" type="tns:Service1Soap">
 <soap12:binding transport="http://schemas.xmlsoap.org/soap/http" />
 <wsdl:operation name="Triangle">
  <soap12:operation soapAction="http://tempuri.org/Triangle" 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="Service1">
 <wsdl:port name="Service1Soap" binding="tns:Service1Soap">
  <soap:address location="http://localhost:3269/Service1.asmx" />
 </wsdl:port>
```

Pre-condition

Post-condition

รูปที่ ก.1. รายละเอียดของเอกสารดับเบิลยูเอสดีแอล-เอส
สำหรับเว็บเซอร์วิสที่ให้บริการตรวจสอบชนิดของรูปสามเหลี่ยม (ต่อ)

```
  <wsdl:port name="Service1Soap12" binding="tns:Service1Soap12">
   <soap12:address location="http://localhost:3269/Service1.asmx" />
  </wsdl:port>
 </wsdl:service>
</wsdl:definitions>
```

รูปที่ ก.1. รายละเอียดของเอกสารดับเบิลยูเอสดีแอล-เอส
สำหรับเว็บเซอร์วิสที่ให้บริการตรวจสอบชนิดของรูปสามเหลี่ยม (ต่อ)

```
<definitions xmlns:swrlx="http://www.w3.org/2003/11/swrlx"
        xmlns:ruleml="http://www.w3.org/2003/11/ruleml"
        xmlns:owlx="http://www.w3.org/2003/05/owl-xml"
        xmlns:xsd="http://www.w3.org/2001/XMLSchema"
        xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
        >
 <ruleml:imp rdf:ID="inputRule1">
  <ruleml:_rlab ruleml:href="#inputRule1"/>
  <ruleml:_body>
   <swrlx:builtinAtom swrlx:builtin="swrlb:greaterThan">
    <ruleml:var>x</ruleml:var>
    <owlx:DataValue owlx:datatype="xsd:int">0</owlx:DataValue>
   </swrlx:builtinAtom>                                              } x > 0
   <swrlx:builtinAtom swrlx:builtin="swrlb:greaterThan">
    <ruleml:var>y</ruleml:var>
    <owlx:DataValue owlx:datatype="xsd:int">0</owlx:DataValue>
   </swrlx:builtinAtom>                                              } y > 0
   <swrlx:builtinAtom swrlx:builtin="swrlb:greaterThan">
    <ruleml:var>z</ruleml:var>
    <owlx:DataValue owlx:datatype="xsd:int">0</owlx:DataValue>
   </swrlx:builtinAtom>                                              } z > 0
  </ruleml:_body>
 </ruleml:imp>
 <ruleml:imp rdf:ID="outputRule1">
  <ruleml:_rlab ruleml:href="#outputRule1"/>
  <ruleml:_body>
   <swrlx:builtinAtom swrlx:builtin="swrlb:add">
    <ruleml:var>sumOfY_Z</ruleml:var>
    <ruleml:var>y</ruleml:var>                                       } sumOfY_Z = y + z
    <ruleml:var>z</ruleml:var>
   </swrlx:builtinAtom>
```

รูปที่ ก.2. รายละเอียดของเอกสารเอสดับเบิลยูอาร์แอล
สำหรับเว็บเซอร์วิสที่ให้บริการตรวจสอบชนิดของรูปสามเหลี่ยม

```
<swrlx:builtinAtom swrlx:builtin="swrlb:add">
 <ruleml:var>sumOfX_Z</ruleml:var>
 <ruleml:var>x</ruleml:var>
 <ruleml:var>z</ruleml:var>
</swrlx:builtinAtom>
<swrlx:builtinAtom swrlx:builtin="swrlb:add">
 <ruleml:var>sumOfX_Y</ruleml:var>
 <ruleml:var>x</ruleml:var>
 <ruleml:var>y</ruleml:var>
</swrlx:builtinAtom>
<swrlx:builtinAtom swrlx:builtin="swrlb:lessThan">
 <ruleml:var>x</ruleml:var>
 <ruleml:var>sumOfY_Z</ruleml:var>
</swrlx:builtinAtom>
<swrlx:builtinAtom swrlx:builtin="swrlb:lessThan">
 <ruleml:var>y</ruleml:var>
 <ruleml:var>sumOfX_Z</ruleml:var>
</swrlx:builtinAtom>
<swrlx:builtinAtom swrlx:builtin="swrlb:lessThan">
 <ruleml:var>z</ruleml:var>
 <ruleml:var>sumOfX_Y</ruleml:var>
</swrlx:builtinAtom>
<swrlx:builtinAtom swrlx:builtin="swrlb:equal">
 <ruleml:var>x</ruleml:var>
 <ruleml:var>y</ruleml:var>
</swrlx:builtinAtom>
<swrlx:builtinAtom swrlx:builtin="swrlb:equal">
 <ruleml:var>y</ruleml:var>
 <ruleml:var>z</ruleml:var>
</swrlx:builtinAtom>
</ruleml:_body>
<ruleml:_head>
 <swrlx:datavaluedPropertyAtom swrlx:property="is">
  <ruleml:var>TriangleResult</ruleml:var>
  <owlx:DataValue owlx:datatype="xsd:string">Equilateral</owlx:DataValue>
 </swrlx:datavaluedPropertyAtom>
</ruleml:_head>
</ruleml:imp>
<ruleml:imp rdf:ID="outputRule2">
 <ruleml:_rlab ruleml:href="#outputRule1"/>
 <ruleml:_body>
```

sumOfX_Z = x + z

sumOfX_Y = x + y

x < sumOfY_Z

y < sumOfX_Z

z < sumOfX_Y

x == y

y == z

TriangleResult is Equilateral

รูปที่ ก.2. รายละเอียดของเอกสารเอสดับเบิลยูอาร์แอล
สำหรับเว็บเซอร์วิสที่ให้บริการตรวจสอบชนิดของรูปสามเหลี่ยม (ต่อ)

```
<swrlx:builtinAtom swrlx:builtin="swrlb:add">
 <ruleml:var>sumOfY_Z</ruleml:var>
 <ruleml:var>y</ruleml:var>
 <ruleml:var>z</ruleml:var>
</swrlx:builtinAtom>
<swrlx:builtinAtom swrlx:builtin="swrlb:add">
 <ruleml:var>sumOfX_Z</ruleml:var>
 <ruleml:var>x</ruleml:var>
 <ruleml:var>z</ruleml:var>
</swrlx:builtinAtom>
<swrlx:builtinAtom swrlx:builtin="swrlb:add">
 <ruleml:var>sumOfX_Y</ruleml:var>
 <ruleml:var>x</ruleml:var>
 <ruleml:var>y</ruleml:var>
</swrlx:builtinAtom>
<swrlx:builtinAtom swrlx:builtin="swrlb:lessThan">
 <ruleml:var>x</ruleml:var>
 <ruleml:var>sumOfY_Z</ruleml:var>
</swrlx:builtinAtom>
<swrlx:builtinAtom swrlx:builtin="swrlb:lessThan">
 <ruleml:var>y</ruleml:var>
 <ruleml:var>sumOfX_Z</ruleml:var>
</swrlx:builtinAtom>
<swrlx:builtinAtom swrlx:builtin="swrlb:lessThan">
 <ruleml:var>z</ruleml:var>
 <ruleml:var>sumOfX_Y</ruleml:var>
</swrlx:builtinAtom>
<swrlx:builtinAtom swrlx:builtin="swrlb:notEqual">
 <ruleml:var>x</ruleml:var>
 <ruleml:var>y</ruleml:var>
</swrlx:builtinAtom>
<swrlx:builtinAtom swrlx:builtin="swrlb:notEqual">
 <ruleml:var>x</ruleml:var>
 <ruleml:var>z</ruleml:var>
</swrlx:builtinAtom>
<swrlx:builtinAtom swrlx:builtin="swrlb:notEqual">
 <ruleml:var>y</ruleml:var>
 <ruleml:var>z</ruleml:var>
</swrlx:builtinAtom>
</ruleml:_body>
<ruleml:_head>
```

sumOfY_Z = y + z

sumOfX_Z = x + z

sumOfX_Y = x + y

x < sumOfY_Z

y < sumOfX_Z

z < sumOfX_Y

x <> y

x <> z

y <> z

รูปที่ ก.2. รายละเอียดของเอกสารเอสดับเบิลยูอาร์แอล
สำหรับเว็บเซอร์วิสที่ให้บริการตรวจสอบชนิดของรูปสามเหลี่ยม (ต่อ)

```
  <swrlx:datavaluedPropertyAtom  swrlx:property="is">
   <ruleml:var>TriangleResult</ruleml:var>
   <owlx:DataValue owlx:datatype="xsd:string">Scalene</owlx:DataValue>
  </swrlx:datavaluedPropertyAtom>
 </ruleml:_head>
</ruleml:imp>
<ruleml:imp rdf:ID="outputRule3">
 <ruleml:_rlab ruleml:href="#outputRule3"/>
 <ruleml:_body>
  <swrlx:builtinAtom swrlx:builtin="swrlb:add">
   <ruleml:var>sumOfY_Z</ruleml:var>
   <ruleml:var>y</ruleml:var>
   <ruleml:var>z</ruleml:var>
  </swrlx:builtinAtom>
  <swrlx:builtinAtom swrlx:builtin="swrlb:add">
   <ruleml:var>sumOfX_Z</ruleml:var>
   <ruleml:var>x</ruleml:var>
   <ruleml:var>z</ruleml:var>
  </swrlx:builtinAtom>
  <swrlx:builtinAtom swrlx:builtin="swrlb:add">
   <ruleml:var>sumOfX_Y</ruleml:var>
   <ruleml:var>x</ruleml:var>
   <ruleml:var>y</ruleml:var>
  </swrlx:builtinAtom>
  <swrlx:builtinAtom swrlx:builtin="swrlb:lessThan">
   <ruleml:var>x</ruleml:var>
   <ruleml:var>sumOfY_Z</ruleml:var>
  </swrlx:builtinAtom>
  <swrlx:builtinAtom swrlx:builtin="swrlb:lessThan">
   <ruleml:var>y</ruleml:var>
   <ruleml:var>sumOfX_Z</ruleml:var>
  </swrlx:builtinAtom>
  <swrlx:builtinAtom swrlx:builtin="swrlb:lessThan">
   <ruleml:var>z</ruleml:var>
   <ruleml:var>sumOfX_Y</ruleml:var>
  </swrlx:builtinAtom>
 </ruleml:_body>
 <ruleml:_head>
  <swrlx:datavaluedPropertyAtom  swrlx:property="is">
   <ruleml:var>TriangleResult</ruleml:var>
   <owlx:DataValue owlx:datatype="xsd:string">a Triangle</owlx:DataValue>
  </swrlx:datavaluedPropertyAtom>
```

TriangleResult is Scalene

sumOfY_Z = y + z

sumOfX_Z = x + z

sumOfX_Y = x + y

x < sumOfY_Z

y < sumOfX_Z

z < sumOfX_Y

TriangleResult is a Triangle

รูปที่ ก.2. รายละเอียดของเอกสารเอสดับเบิลยูอาร์แอล
สำหรับเว็บเซอร์วิสที่ให้บริการตรวจสอบชนิดของรูปสามเหลี่ยม (ต่อ)

```
 </ruleml:_head>
</ruleml:imp>
<ruleml:imp rdf:ID="outputRule4">
 <ruleml:_rlab ruleml:href="#outputRule4"/>
 <ruleml:_body>
  <swrlx:builtinAtom swrlx:builtin="swrlb:add">
   <ruleml:var>sumOfY_Z</ruleml:var>
   <ruleml:var>y</ruleml:var>
   <ruleml:var>z</ruleml:var>
  </swrlx:builtinAtom>
  <swrlx:builtinAtom swrlx:builtin="swrlb:add">
   <ruleml:var>sumOfX_Z</ruleml:var>
   <ruleml:var>x</ruleml:var>
   <ruleml:var>z</ruleml:var>
  </swrlx:builtinAtom>
  <swrlx:builtinAtom swrlx:builtin="swrlb:add">
   <ruleml:var>sumOfX_Y</ruleml:var>
   <ruleml:var>x</ruleml:var>
   <ruleml:var>y</ruleml:var>
  </swrlx:builtinAtom>
  <swrlx:builtinAtom swrlx:builtin="swrlb:lessThan">
   <ruleml:var>x</ruleml:var>
   <ruleml:var>sumOfY_Z</ruleml:var>
  </swrlx:builtinAtom>
  <swrlx:builtinAtom swrlx:builtin="swrlb:lessThan">
   <ruleml:var>y</ruleml:var>
   <ruleml:var>sumOfX_Z</ruleml:var>
  </swrlx:builtinAtom>
  <swrlx:builtinAtom swrlx:builtin="swrlb:lessThan">
   <ruleml:var>z</ruleml:var>
   <ruleml:var>sumOfX_Y</ruleml:var>
  </swrlx:builtinAtom>
  <swrlx:builtinAtom swrlx:builtin="swrlb:equal">
   <ruleml:var>x</ruleml:var>
   <ruleml:var>y</ruleml:var>
  </swrlx:builtinAtom>
  <swrlx:builtinAtom swrlx:builtin="swrlb:notEqual">
   <ruleml:var>x</ruleml:var>
   <ruleml:var>z</ruleml:var>
  </swrlx:builtinAtom>
 </ruleml:_body>
 <ruleml:_head>
```

sumOfY_Z = y + z

sumOfX_Z = x + z

sumOfX_Y = x + y

x < sumOfY_Z

y < sumOfX_Z

z < sumOfX_Y

x == y

x <> z

รูปที่ ก.2. รายละเอียดของเอกสารเอสดับเบิลยูอาร์แอล
สำหรับเว็บเซอร์วิสที่ให้บริการตรวจสอบชนิดของรูปสามเหลี่ยม (ต่อ)

```
  <swrlx:datavaluedPropertyAtom  swrlx:property="is">
   <ruleml:var>TriangleResult</ruleml:var>
   <owlx:DataValue owlx:datatype="xsd:string">Isosceles</owlx:DataValue>
  </swrlx:datavaluedPropertyAtom>
 </ruleml:_head>
</ruleml:imp>
<ruleml:imp rdf:ID="outputRule5">
 <ruleml:_rlab ruleml:href="#outputRule5"/>
 <ruleml:_body>
  <swrlx:builtinAtom swrlx:builtin="swrlb:add">
   <ruleml:var>sumOfY_Z</ruleml:var>
   <ruleml:var>y</ruleml:var>
   <ruleml:var>z</ruleml:var>
  </swrlx:builtinAtom>
  <swrlx:builtinAtom swrlx:builtin="swrlb:add">
   <ruleml:var>sumOfX_Z</ruleml:var>
   <ruleml:var>x</ruleml:var>
   <ruleml:var>z</ruleml:var>
  </swrlx:builtinAtom>
  <swrlx:builtinAtom swrlx:builtin="swrlb:add">
   <ruleml:var>sumOfX_Y</ruleml:var>
   <ruleml:var>x</ruleml:var>
   <ruleml:var>y</ruleml:var>
  </swrlx:builtinAtom>
  <swrlx:builtinAtom swrlx:builtin="swrlb:lessThan">
   <ruleml:var>x</ruleml:var>
   <ruleml:var>sumOfY_Z</ruleml:var>
  </swrlx:builtinAtom>
  <swrlx:builtinAtom swrlx:builtin="swrlb:lessThan">
   <ruleml:var>y</ruleml:var>
   <ruleml:var>sumOfX_Z</ruleml:var>
  </swrlx:builtinAtom>
  <swrlx:builtinAtom swrlx:builtin="swrlb:lessThan">
   <ruleml:var>z</ruleml:var>
   <ruleml:var>sumOfX_Y</ruleml:var>
  </swrlx:builtinAtom>
  <swrlx:builtinAtom swrlx:builtin="swrlb:equal">
   <ruleml:var>x</ruleml:var>
   <ruleml:var>z</ruleml:var>
  </swrlx:builtinAtom>
```

TriangleResult is Isosceles

sumOfY_Z = y + z

sumOfX_Z = x + z

sumOfX_Y = x + y

x < sumOfY_Z

y < sumOfX_Z

z < sumOfX_Y

x == z

รูปที่ ก.2. รายละเอียดของเอกสารเอสดับเบิลยูอาร์แอล
สำหรับเว็บเซอร์วิสที่ให้บริการตรวจสอบชนิดของรูปสามเหลี่ยม (ต่อ)

```
  <swrlx:builtinAtom swrlx:builtin="swrlb:notEqual">
   <ruleml:var>x</ruleml:var>
   <ruleml:var>y</ruleml:var>
  </swrlx:builtinAtom>
 </ruleml:_body>
 <ruleml:_head>
  <swrlx:datavaluedPropertyAtom  swrlx:property="is">
   <ruleml:var>TriangleResult</ruleml:var>
   <owlx:DataValue owlx:datatype="xsd:string">Isosceles</owlx:DataValue>
  </swrlx:datavaluedPropertyAtom>
 </ruleml:_head>
</ruleml:imp>
<ruleml:imp rdf:ID="outputRule6">
 <ruleml:_rlab ruleml:href="#outputRule6"/>
 <ruleml:_body>
  <swrlx:builtinAtom swrlx:builtin="swrlb:add">
   <ruleml:var>sumOfY_Z</ruleml:var>
   <ruleml:var>y</ruleml:var>
   <ruleml:var>z</ruleml:var>
  </swrlx:builtinAtom>
  <swrlx:builtinAtom swrlx:builtin="swrlb:add">
   <ruleml:var>sumOfX_Z</ruleml:var>
   <ruleml:var>x</ruleml:var>
   <ruleml:var>z</ruleml:var>
  </swrlx:builtinAtom>
  <swrlx:builtinAtom swrlx:builtin="swrlb:add">
   <ruleml:var>sumOfX_Y</ruleml:var>
   <ruleml:var>x</ruleml:var>
   <ruleml:var>y</ruleml:var>
  </swrlx:builtinAtom>
  <swrlx:builtinAtom swrlx:builtin="swrlb:lessThan">
   <ruleml:var>x</ruleml:var>
   <ruleml:var>sumOfY_Z</ruleml:var>
  </swrlx:builtinAtom>
  <swrlx:builtinAtom swrlx:builtin="swrlb:lessThan">
   <ruleml:var>y</ruleml:var>
   <ruleml:var>sumOfX_Z</ruleml:var>
  </swrlx:builtinAtom>
  <swrlx:builtinAtom swrlx:builtin="swrlb:lessThan">
   <ruleml:var>z</ruleml:var>
   <ruleml:var>sumOfX_Y</ruleml:var>
  </swrlx:builtinAtom>
```

x <> y

TriangleResult is Isosceles

sumOfY_Z = y + z

sumOfX_Z = x + z

sumOfX_Y = x + y

x < sumOfY_Z

y < sumOfX_Z

z < sumOfX_Y

รูปที่ ก.2. รายละเอียดของเอกสารเอสดับเบิลยูอาร์แอล
สำหรับเว็บเซอร์วิสที่ให้บริการตรวจสอบชนิดของรูปสามเหลี่ยม (ต่อ)

```
    <swrlx:builtinAtom swrlx:builtin="swrlb:equal">
     <ruleml:var>y</ruleml:var>
     <ruleml:var>z</ruleml:var>
    </swrlx:builtinAtom>
    <swrlx:builtinAtom swrlx:builtin="swrlb:notEqual">
     <ruleml:var>x</ruleml:var>
     <ruleml:var>z</ruleml:var>
    </swrlx:builtinAtom>
   </ruleml:_body>
   <ruleml:_head>
    <swrlx:datavaluedPropertyAtom  swrlx:property="is">
     <ruleml:var>TriangleResult</ruleml:var>
     <owlx:DataValue owlx:datatype="xsd:string">Isosceles</owlx:DataValue>
    </swrlx:datavaluedPropertyAtom>
   </ruleml:_head>
  </ruleml:imp>
</definitions>
```

รูปที่ ก.2. รายละเอียดของเอกสารเอสดับเบิลยูอาร์แอล
สำหรับเว็บเซอร์วิสที่ให้บริการตรวจสอบชนิดของรูปสามเหลี่ยม (ต่อ)

ก.2. เว็บเซอร์วิสที่ให้บริการตรวจสอบชนิดของรูปสี่เหลี่ยม

```
<?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://tempuri.org/"
        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://tempuri.org/"
        xmlns:wssem="http://lsdis.cs.uga.edu/projects/meteor-s/wsdl-s/examples/WSSemantics.xsd"
        xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/">
 <wsdl:types>
  <s:schema elementFormDefault="qualified" targetNamespace="http://tempuri.org/">
   <s:element name="Rectangle">
    <s:complexType>
```

รูปที่ ก.3. รายละเอียดของเอกสารดับเบิลยูเอสดีแอล-เอส
สำหรับเว็บเซอร์วิสที่ให้บริการตรวจสอบชนิดของรูปสี่เหลี่ยม

```
      <s:sequence>
       <s:element minOccurs="1" maxOccurs="1" name="x" type="s:int"/>
       <s:element minOccurs="1" maxOccurs="1" name="y" type="s:int"/>
       <s:element minOccurs="1" maxOccurs="1" name="w" type="s:int"/>
       <s:element minOccurs="1" maxOccurs="1" name="z" type="s:int"/>
       <s:element minOccurs="1" maxOccurs="1" name="u" type="s:int"/>
       <s:element minOccurs="1" maxOccurs="1" name="v" type="s:int"/>
      </s:sequence>
     </s:complexType>
    </s:element>
    <s:element name="RectangleResponse">
     <s:complexType>
      <s:sequence>
       <s:element minOccurs="0" maxOccurs="1" name="RectangleResult" type="s:string" />
      </s:sequence>
     </s:complexType>
    </s:element>
   </s:schema>
  </wsdl:types>
  <wsdl:message name="RectangleSoapIn">
   <wsdl:part name="parameters" element="tns:Rectangle" />
  </wsdl:message>
  <wsdl:message name="RectangleSoapOut">
   <wsdl:part name="parameters" element="tns:RectangleResponse" />
  </wsdl:message>
  <wsdl:portType name="Service1Soap">
   <wsdl:operation name="RectangleType">
    <wsdl:input message="tns:RectangleSoapIn" />
    <wsdl:output message="tns:RectangleSoapOut" />
    <wssem:precondition name="RectanglePrecond"
                wssem:modelReference="http://localhost/example.swrlx#inputRule1"/>
    <wssem:effect name="ResultEffect"
            wssem:modelReference="http://localhost/example.swrlx#outputRule1
            http://localhost/example.swrlx#outputRule2
            http://localhost/example.swrlx#outputRule3
            http://localhost/example.swrlx#outputRule4
            http://localhost/example.swrlx#outputRule5"/>
   </wsdl:operation>
  </wsdl:portType>
  <wsdl:binding name="Service1Soap" type="tns:Service1Soap">
   <soap:binding transport="http://schemas.xmlsoap.org/soap/http" />
   <wsdl:operation name="Rectangle">
```

รูปที่ ก.3. รายละเอียดของเอกสารดับเบิลยูเอสดีแอล-เอส
สำหรับเว็บเซอร์วิสที่ให้บริการตรวจสอบชนิดของรูปสี่เหลี่ยม (ต่อ)

```
    <soap:operation soapAction="http://tempuri.org/Triangle" 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="Service1Soap12" type="tns:Service1Soap">
  <soap12:binding transport="http://schemas.xmlsoap.org/soap/http" />
  <wsdl:operation name="Rectangle">
    <soap12:operation soapAction="http://tempuri.org/Triangle" 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="Service1">
  <wsdl:port name="Service1Soap" binding="tns:Service1Soap">
    <soap:address location="http://localhost:3269/Service1.asmx" />
  </wsdl:port>
  <wsdl:port name="Service1Soap12" binding="tns:Service1Soap12">
    <soap12:address location="http://localhost:3269/Service1.asmx" />
  </wsdl:port>
</wsdl:service>
</wsdl:definitions>
```

รูปที่ ก.3. รายละเอียดของเอกสารดับเบิลยูเอสดีแอล-เอส
สำหรับเว็บเซอร์วิสที่ให้บริการตรวจสอบชนิดของรูปสี่เหลี่ยม (ต่อ)

```
<definitions xmlns:swrlx="http://www.w3.org/2003/11/swrlx"
        xmlns:ruleml="http://www.w3.org/2003/11/ruleml"
        xmlns:owlx="http://www.w3.org/2003/05/owl-xml"
        xmlns:xsd="http://www.w3.org/2001/XMLSchema"
        xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#">
 <ruleml:imp rdf:ID="inputRule1">
  <ruleml:_rlab ruleml:href="#inputRule1"/>
  <ruleml:_body>
   <swrlx:builtinAtom swrlx:builtin="swrlb:greaterThan">
    <ruleml:var>x</ruleml:var>
    <owlx:DataValue owlx:datatype="xsd:int">0</owlx:DataValue>
   </swrlx:builtinAtom>
   <swrlx:builtinAtom swrlx:builtin="swrlb:greaterThan">
    <ruleml:var>y</ruleml:var>
    <owlx:DataValue owlx:datatype="xsd:int">0</owlx:DataValue>
   </swrlx:builtinAtom>
   <swrlx:builtinAtom swrlx:builtin="swrlb:greaterThan">
    <ruleml:var>w</ruleml:var>
    <owlx:DataValue owlx:datatype="xsd:int">0</owlx:DataValue>
   </swrlx:builtinAtom>
   <swrlx:builtinAtom swrlx:builtin="swrlb:greaterThan">
    <ruleml:var>z</ruleml:var>
    <owlx:DataValue owlx:datatype="xsd:int">0</owlx:DataValue>
   </swrlx:builtinAtom>
   <swrlx:builtinAtom swrlx:builtin="swrlb:greaterThan">
    <ruleml:var>u</ruleml:var>
    <owlx:DataValue owlx:datatype="xsd:int">0</owlx:DataValue>
   </swrlx:builtinAtom>
   <swrlx:builtinAtom swrlx:builtin="swrlb:greaterThan">
    <ruleml:var>v</ruleml:var>
    <owlx:DataValue owlx:datatype="xsd:int">0</owlx:DataValue>
   </swrlx:builtinAtom>
  </ruleml:_body>
 </ruleml:imp>
 <ruleml:imp rdf:ID="outputRule1">
  <ruleml:_rlab ruleml:href="#outputRule1"/>
  <ruleml:_body>
   <swrlx:builtinAtom swrlx:builtin="swrlb:equal">
    <ruleml:var>x</ruleml:var>
    <ruleml:var>y</ruleml:var>
   </swrlx:builtinAtom>
```

x > 0

y > 0

w > 0

z > 0

u > 0

v > 0

x == y

รูปที่ ก.4. รายละเอียดของเอกสารเอสดับเบิลยูอาร์แอล
สำหรับเว็บเซอร์วิสที่ให้บริการตรวจสอบชนิดของรูปสี่เหลี่ยม

```
  <swrlx:builtinAtom swrlx:builtin="swrlb:equal">
   <ruleml:var>y</ruleml:var>
   <ruleml:var>z</ruleml:var>
  </swrlx:builtinAtom>
  <swrlx:builtinAtom swrlx:builtin="swrlb:equal">
   <ruleml:var>w</ruleml:var>
   <ruleml:var>z</ruleml:var>
  </swrlx:builtinAtom>
  <swrlx:builtinAtom swrlx:builtin="swrlb:equal">
   <ruleml:var>u</ruleml:var>
   <ruleml:var>v</ruleml:var>
  </swrlx:builtinAtom>
 </ruleml:_body>
 <ruleml:_head>
  <swrlx:datavaluedPropertyAtom  swrlx:property="is">
   <ruleml:var>RectangleResult</ruleml:var>
   <owlx:DataValue owlx:datatype="xsd:string">Square</owlx:DataValue>
  </swrlx:datavaluedPropertyAtom>
 </ruleml:_head>
</ruleml:imp>
<ruleml:imp rdf:ID="outputRule2">
 <ruleml:_rlab ruleml:href="#outputRule2"/>
 <ruleml:_body>
  <swrlx:builtinAtom swrlx:builtin="swrlb:equal">
   <ruleml:var>x</ruleml:var>
   <ruleml:var>y</ruleml:var>
  </swrlx:builtinAtom>
  <swrlx:builtinAtom swrlx:builtin="swrlb:equal">
   <ruleml:var>y</ruleml:var>
   <ruleml:var>z</ruleml:var>
  </swrlx:builtinAtom>
  <swrlx:builtinAtom swrlx:builtin="swrlb:equal">
   <ruleml:var>w</ruleml:var>
   <ruleml:var>z</ruleml:var>
  </swrlx:builtinAtom>
 </ruleml:_body>
 <ruleml:_head>
  <swrlx:datavaluedPropertyAtom  swrlx:property="is">
   <ruleml:var>RectangleResult</ruleml:var>
   <owlx:DataValue owlx:datatype="xsd:string">Diamond</owlx:DataValue>
  </swrlx:datavaluedPropertyAtom>
 </ruleml:_head>
```

y == z

w == z

u == v

RectangleResult is Square

x == y

y == z

w == z

RectangleResult is Diamond

รูปที่ ก.4. รายละเอียดของเอกสารเอสดับเบิลยูอาร์แอล
สำหรับเว็บเซอร์วิสที่ให้บริการตรวจสอบชนิดของรูปสี่เหลี่ยม (ต่อ)

```
</ruleml:imp>
<ruleml:imp rdf:ID="outputRule3">
 <ruleml:_rlab ruleml:href="#outputRule3"/>
 <ruleml:_body>
  <swrlx:builtinAtom swrlx:builtin="swrlb:equal">
   <ruleml:var>x</ruleml:var>
   <ruleml:var>y</ruleml:var>
  </swrlx:builtinAtom>
  <swrlx:builtinAtom swrlx:builtin="swrlb:notEqual">
   <ruleml:var>y</ruleml:var>
   <ruleml:var>z</ruleml:var>
  </swrlx:builtinAtom>
  <swrlx:builtinAtom swrlx:builtin="swrlb:equal">
   <ruleml:var>w</ruleml:var>
   <ruleml:var>z</ruleml:var>
  </swrlx:builtinAtom>
  <swrlx:builtinAtom swrlx:builtin="swrlb:equal">
   <ruleml:var>u</ruleml:var>
   <ruleml:var>v</ruleml:var>
  </swrlx:builtinAtom>
 </ruleml:_body>
 <ruleml:_head>
  <swrlx:datavaluedPropertyAtom  swrlx:property="is">
   <ruleml:var>RectangleResult</ruleml:var>
   <owlx:DataValue owlx:datatype="xsd:string">Rectangle</owlx:DataValue>
  </swrlx:datavaluedPropertyAtom>
 </ruleml:_head>
</ruleml:imp>
<ruleml:imp rdf:ID="outputRule4">
 <ruleml:_rlab ruleml:href="#outputRule4"/>
 <ruleml:_body>
  <swrlx:builtinAtom swrlx:builtin="swrlb:equal">
   <ruleml:var>x</ruleml:var>
   <ruleml:var>y</ruleml:var>
  </swrlx:builtinAtom>
  <swrlx:builtinAtom swrlx:builtin="swrlb:notEqual">
   <ruleml:var>y</ruleml:var>
   <ruleml:var>z</ruleml:var>
  </swrlx:builtinAtom>
```

x == y

y == z

w == z

u == v

RectangleResult is Rectangle

x == y

y <> z

รูปที่ ก.4. รายละเอียดของเอกสารเอสดับเบิลยูอาร์แอล
สำหรับเว็บเซอร์วิสที่ให้บริการตรวจสอบชนิดของรูปสี่เหลี่ยม (ต่อ)

```
    <swrlx:builtinAtom swrlx:builtin="swrlb:equal">
     <ruleml:var>w</ruleml:var>
     <ruleml:var>z</ruleml:var>
    </swrlx:builtinAtom>
    <swrlx:builtinAtom swrlx:builtin="swrlb:notEqual">
     <ruleml:var>u</ruleml:var>
     <ruleml:var>v</ruleml:var>
    </swrlx:builtinAtom>
   </ruleml:_body>
   <ruleml:_head>
    <swrlx:datavaluedPropertyAtom swrlx:property="is">
     <ruleml:var>RectangleResult</ruleml:var>
     <owlx:DataValue owlx:datatype="xsd:string">Parallelogram</owlx:DataValue>
    </swrlx:datavaluedPropertyAtom>
   </ruleml:_head>
  </ruleml:imp>
  <ruleml:imp rdf:ID="outputRule5">
   <ruleml:_rlab ruleml:href="#outputRule5"/>
   <ruleml:_body>
    <swrlx:builtinAtom swrlx:builtin="swrlb:notEqual">
     <ruleml:var>x</ruleml:var>
     <ruleml:var>y</ruleml:var>
    </swrlx:builtinAtom>
    <swrlx:builtinAtom swrlx:builtin="swrlb:notEqual">
     <ruleml:var>y</ruleml:var>
     <ruleml:var>z</ruleml:var>
    </swrlx:builtinAtom>
    <swrlx:builtinAtom swrlx:builtin="swrlb:notEqual">
     <ruleml:var>w</ruleml:var>
     <ruleml:var>z</ruleml:var>
    </swrlx:builtinAtom>
   </ruleml:_body>
   <ruleml:_head>
    <swrlx:datavaluedPropertyAtom swrlx:property="is">
     <ruleml:var>RectangleResult</ruleml:var>
     <owlx:DataValue owlx:datatype="xsd:string">Trapezium</owlx:DataValue>
    </swrlx:datavaluedPropertyAtom>
   </ruleml:_head>
  </ruleml:imp>
</definitions>
```

w == z

u <> v

RectangleResult is Parallelogram

x <> y

y <> z

w <> z

RectangleResult is Trapezium

รูปที่ ก.4. รายละเอียดของเอกสารเอสดับเบิลยูอาร์แอล
สำหรับเว็บเซอร์วิสที่ให้บริการตรวจสอบชนิดของรูปสี่เหลี่ยม (ต่อ)

ก.3. เว็บเซอร์วิสที่ให้บริการการหาค่ากลาง

```
<?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://tempuri.org/"
        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://tempuri.org/"
        xmlns:wssem="http://lsdis.cs.uga.edu/projects/meteor-s/wsdl-s/examples/WSSemantics.xsd"
        xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/">
 <wsdl:types>
  <s:schema elementFormDefault="qualified" targetNamespace="http://tempuri.org/">
   <s:element name="Middle">
    <s:complexType>
     <s:sequence>
      <s:element minOccurs="1" maxOccurs="1" name="x" type="s:int" />
      <s:element minOccurs="1" maxOccurs="1" name="y" type="s:int" />
      <s:element minOccurs="1" maxOccurs="1" name="z" type="s:int" />
     </s:sequence>
    </s:complexType>
   </s:element>
   <s:element name="MiddleResponse">
    <s:complexType>
     <s:sequence>
      <s:element minOccurs="0" maxOccurs="1" name="MiddleResult" type="s:string" />
     </s:sequence>
    </s:complexType>
   </s:element>
  </s:schema>
 </wsdl:types>
 <wsdl:message name="MiddleSoapIn">
  <wsdl:part name="parameters" element="tns:Middle" />
 </wsdl:message>
 <wsdl:message name="MiddleSoapOut">
  <wsdl:part name="parameters" element="tns:MiddleResponse" />
 </wsdl:message>
```

ข้อมูลนำเข้า

ผลลัพธ์

รูปที่ ก.5. รายละเอียดของเอกสารดับเบิลยูเอสดีแอล-เอส
สำหรับเว็บเซอร์วิสที่ให้บริการการหาค่ากลาง

```
<wsdl:portType name="Service1Soap">
 <wsdl:operation name="MiddleValue">
  <wsdl:input message="tns:MiddleSoapIn" />
  <wsdl:output message="tns:MiddleSoapOut" />
  <wssem:precondition name="MiddlePrecond"
                wssem:modelReference="http://localhost/example.swrlx#inputRule1
                "/>                                                          } Pre-condition
  <wssem:effect name="ResultEffect"
            wssem:modelReference="http://localhost/example.swrlx#outputRule1
            http://localhost/example.swrlx#outputRule2
            http://localhost/example.swrlx#outputRule3
            http://localhost/example.swrlx#outputRule4                       } Post-condition
            http://localhost/example.swrlx#outputRule5
            http://localhost/example.swrlx#outputRule6
            "/>
 </wsdl:operation>
</wsdl:portType>
<wsdl:binding name="Service1Soap" type="tns:Service1Soap">
 <soap:binding transport="http://schemas.xmlsoap.org/soap/http" />
 <wsdl:operation name="Middle">
  <soap:operation soapAction="http://tempuri.org/Middle" 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="Service1Soap12" type="tns:Service1Soap">
 <soap12:binding transport="http://schemas.xmlsoap.org/soap/http" />
 <wsdl:operation name="Middle">
  <soap12:operation soapAction="http://tempuri.org/Middle" style="document" />
  <wsdl:input>
   <soap12:body use="literal" />
  </wsdl:input>
  <wsdl:output>
   <soap12:body use="literal" />
  </wsdl:output>
 </wsdl:operation>
</wsdl:binding>
```

รูปที่ ก.5. รายละเอียดของเอกสารดับเบิลยูเอสดีแอล-เอส
สำหรับเว็บเซอร์วิสที่ให้บริการการหาค่ากลาง (ต่อ)

```
<wsdl:service name="Service1">
  <wsdl:port name="Service1Soap" binding="tns:Service1Soap">
    <soap:address location="http://localhost:3269/Service1.asmx" />
  </wsdl:port>
  <wsdl:port name="Service1Soap12" binding="tns:Service1Soap12">
    <soap12:address location="http://localhost:3269/Service1.asmx" />
  </wsdl:port>
 </wsdl:service>
</wsdl:definitions>
```

รูปที่ ก.5. รายละเอียดของเอกสารดับเบิลยูเอสดีแอล-เอส
สำหรับเว็บเซอร์วิสที่ให้บริการการหาค่ากลาง (ต่อ)

```
<definitions xmlns:swrlx="http://www.w3.org/2003/11/swrlx"
        xmlns:ruleml="http://www.w3.org/2003/11/ruleml"
        xmlns:owlx="http://www.w3.org/2003/05/owl-xml"
        xmlns:xsd="http://www.w3.org/2001/XMLSchema"
        xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
        >
 <ruleml:imp rdf:ID="inputRule1">
  <ruleml:_rlab ruleml:href="#inputRule1"/>
  <ruleml:_body>
   <swrlx:builtinAtom swrlx:builtin="swrlb:greaterThan">
    <ruleml:var>x</ruleml:var>
    <owlx:DataValue owlx:datatype="xsd:int">0</owlx:DataValue>
   </swrlx:builtinAtom>                                              } x > 0
   <swrlx:builtinAtom swrlx:builtin="swrlb:greaterThan">
    <ruleml:var>y</ruleml:var>
    <owlx:DataValue owlx:datatype="xsd:int">0</owlx:DataValue>
   </swrlx:builtinAtom>                                              } y > 0
   <swrlx:builtinAtom swrlx:builtin="swrlb:greaterThan">
    <ruleml:var>z</ruleml:var>
    <owlx:DataValue owlx:datatype="xsd:int">0</owlx:DataValue>
   </swrlx:builtinAtom>                                              } z > 0
   <swrlx:builtinAtom swrlx:builtin="swrlb:lessThan">
    <ruleml:var>x</ruleml:var>
    <owlx:DataValue owlx:datatype="xsd:int">100</owlx:DataValue>
   </swrlx:builtinAtom>                                              } x < 100
```

รูปที่ ก.6. รายละเอียดของเอกสารเอสดับเบิลยูอาร์แอล
สำหรับเว็บเซอร์วิสที่ให้บริการการหาค่ากลาง

```
    <swrlx:builtinAtom swrlx:builtin="swrlb:lessThan">
      <ruleml:var>y</ruleml:var>
      <owlx:DataValue owlx:datatype="xsd:int">100</owlx:DataValue>
    </swrlx:builtinAtom>
    <swrlx:builtinAtom swrlx:builtin="swrlb:lessThan">
      <ruleml:var>z</ruleml:var>
      <owlx:DataValue owlx:datatype="xsd:int">100</owlx:DataValue>
    </swrlx:builtinAtom>
  </ruleml:_body>
</ruleml:imp>
<ruleml:imp rdf:ID="outputRule1">
  <ruleml:_rlab ruleml:href="#outputRule1"/>
  <ruleml:_body>
    <swrlx:builtinAtom swrlx:builtin="swrlb:lessThan">
      <ruleml:var>y</ruleml:var>
      <ruleml:var>z</ruleml:var>
    </swrlx:builtinAtom>
    <swrlx:builtinAtom swrlx:builtin="swrlb:lessThan">
      <ruleml:var>x</ruleml:var>
      <ruleml:var>y</ruleml:var>
    </swrlx:builtinAtom>
  </ruleml:_body>
  <ruleml:_head>
    <swrlx:datavaluedPropertyAtom swrlx:property="is">
      <ruleml:var>MiddleResult</ruleml:var>
      <owlx:DataValue owlx:datatype="xsd:string">y</owlx:DataValue>
    </swrlx:datavaluedPropertyAtom>
  </ruleml:_head>
</ruleml:imp>
<ruleml:imp rdf:ID="outputRule2">
  <ruleml:_rlab ruleml:href="#outputRule1"/>
  <ruleml:_body>
    <swrlx:builtinAtom swrlx:builtin="swrlb:lessThan">
      <ruleml:var>y</ruleml:var>
      <ruleml:var>z</ruleml:var>
    </swrlx:builtinAtom>
    <swrlx:builtinAtom swrlx:builtin="swrlb:lessThan">
      <ruleml:var>x</ruleml:var>
      <ruleml:var>z</ruleml:var>
    </swrlx:builtinAtom>
```

y < 100

z < 100

y < z

x < y

MiddleResult is y

y < z

x < z

รูปที่ ก.6. รายละเอียดของเอกสารเอสดับเบิลยูอาร์แอล
สำหรับเว็บเซอร์วิสที่ให้บริการการหาค่ากลาง (ต่อ)

```
    <swrlx:builtinAtom swrlx:builtin="swrlb:greaterThanOrEqual">
     <ruleml:var>x</ruleml:var>
     <ruleml:var>y</ruleml:var>
    </swrlx:builtinAtom>
   </ruleml:_body>
   <ruleml:_head>
    <swrlx:datavaluedPropertyAtom  swrlx:property="is">
     <ruleml:var>MiddleResult</ruleml:var>
     <owlx:DataValue owlx:datatype="xsd:string">x</owlx:DataValue>
    </swrlx:datavaluedPropertyAtom>
   </ruleml:_head>
  </ruleml:imp>
  <ruleml:imp rdf:ID="outputRule3">
   <ruleml:_rlab ruleml:href="#outputRule3"/>
   <ruleml:_body>
    <swrlx:builtinAtom swrlx:builtin="swrlb:lessThan">
     <ruleml:var>y</ruleml:var>
     <ruleml:var>z</ruleml:var>
    </swrlx:builtinAtom>
    <swrlx:builtinAtom swrlx:builtin="swrlb:greaterThanOrEqual">
     <ruleml:var>x</ruleml:var>
     <ruleml:var>z</ruleml:var>
    </swrlx:builtinAtom>
    <swrlx:builtinAtom swrlx:builtin="swrlb:greaterThanOrEqual">
     <ruleml:var>x</ruleml:var>
     <ruleml:var>y</ruleml:var>
    </swrlx:builtinAtom>
   </ruleml:_body>
   <ruleml:_head>
    <swrlx:datavaluedPropertyAtom  swrlx:property="is">
     <ruleml:var>MiddleResult</ruleml:var>
     <owlx:DataValue owlx:datatype="xsd:string">z</owlx:DataValue>
    </swrlx:datavaluedPropertyAtom>
   </ruleml:_head>
  </ruleml:imp>
  <ruleml:imp rdf:ID="outputRule4">
   <ruleml:_rlab ruleml:href="#outputRule4"/>
   <ruleml:_body>
    <swrlx:builtinAtom swrlx:builtin="swrlb:greaterThanOrEqual">
     <ruleml:var>y</ruleml:var>
     <ruleml:var>z</ruleml:var>
    </swrlx:builtinAtom>
```

x < y

MiddleResult is x

y < z

x >= z

x >= y

MiddleResult is z

y >= z

รูปที่ ก.6. รายละเอียดของเอกสารเอสดับเบิลยูอาร์แอล
สำหรับเว็บเซอร์วิสที่ให้บริการการหาค่ากลาง (ต่อ)

```
    <swrlx:builtinAtom swrlx:builtin="swrlb:greaterThan">
     <ruleml:var>x</ruleml:var>
     <ruleml:var>y</ruleml:var>
    </swrlx:builtinAtom>                                          } x >= y
   </ruleml:_body>
   <ruleml:_head>
    <swrlx:datavaluedPropertyAtom  swrlx:property="is">
     <ruleml:var>MiddleResult</ruleml:var>
     <owlx:DataValue owlx:datatype="xsd:string">y</owlx:DataValue>
    </swrlx:datavaluedPropertyAtom>                               } MiddleResult is y
   </ruleml:_head>
  </ruleml:imp>
  <ruleml:imp rdf:ID="outputRule5">
   <ruleml:_rlab ruleml:href="#outputRule5"/>
   <ruleml:_body>
    <swrlx:builtinAtom swrlx:builtin="swrlb:greaterThanOrEqual">
     <ruleml:var>y</ruleml:var>
     <ruleml:var>z</ruleml:var>
    </swrlx:builtinAtom>                                          } y >= z
    <swrlx:builtinAtom swrlx:builtin="swrlb:greaterThan">
     <ruleml:var>x</ruleml:var>
     <ruleml:var>z</ruleml:var>
    </swrlx:builtinAtom>                                          } x > z
    <swrlx:builtinAtom swrlx:builtin="swrlb:lessThanOrEqual">
     <ruleml:var>x</ruleml:var>
     <ruleml:var>y</ruleml:var>
    </swrlx:builtinAtom>                                          } x <= y
   </ruleml:_body>
   <ruleml:_head>
    <swrlx:datavaluedPropertyAtom  swrlx:property="is">
     <ruleml:var>MiddleResult</ruleml:var>
     <owlx:DataValue owlx:datatype="xsd:string">x</owlx:DataValue>
    </swrlx:datavaluedPropertyAtom>                               } MiddleResult is x
   </ruleml:_head>
  </ruleml:imp>
  <ruleml:imp rdf:ID="outputRule6">
   <ruleml:_rlab ruleml:href="#outputRule6"/>
   <ruleml:_body>
    <swrlx:builtinAtom swrlx:builtin="swrlb:greaterThanOrEqual">
     <ruleml:var>y</ruleml:var>
     <ruleml:var>z</ruleml:var>
    </swrlx:builtinAtom>                                          } y >= z
```

รูปที่ ก.6. รายละเอียดของเอกสารเอสดับเบิลยูอาร์แอล
สำหรับเว็บเซอร์วิสที่ให้บริการการหาค่ากลาง (ต่อ)

```
    <swrlx:builtinAtom swrlx:builtin="swrlb:lessThanOrEqual">
      <ruleml:var>x</ruleml:var>
      <ruleml:var>z</ruleml:var>
    </swrlx:builtinAtom>
    <swrlx:builtinAtom swrlx:builtin="swrlb:lessThanOrEqual">
      <ruleml:var>x</ruleml:var>
      <ruleml:var>y</ruleml:var>
    </swrlx:builtinAtom>
  </ruleml:_body>
  <ruleml:_head>
    <swrlx:datavaluedPropertyAtom swrlx:property="is">
      <ruleml:var>MiddleResult</ruleml:var>
      <owlx:DataValue owlx:datatype="xsd:string">z</owlx:DataValue>
    </swrlx:datavaluedPropertyAtom>
  </ruleml:_head>
 </ruleml:imp>
</definitions>
```

x <= z

x <= y

MiddleResult is z

รูปที่ ก.6. รายละเอียดของเอกสารเอสดับเบิลยูอาร์แอล
สำหรับเว็บเซอร์วิสที่ให้บริการการหาค่ากลาง (ต่อ)

ภาคผนวก ข
ตัวอย่างตารางตัดสินใจ

ข.1. เว็บเซอร์วิสที่ให้บริการตรวจสอบชนิดของรูปสามเหลี่ยม

ตารางที่ ข.1. ตารางตัดสินใจสำหรับเว็บเซอร์วิสที่ให้บริการตรวจสอบชนิดของรูปสามเหลี่ยม

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19
(x > 0)	T	F	T	T	T	T	T	T	T	T	T	T	T	T	T	T	T	T	T
(y > 0)	T	-	F	T	T	T	T	T	T	T	T	T	T	T	T	T	T	T	T
(z > 0)	T	-	-	F	T	T	T	T	T	T	T	T	T	T	T	T	T	T	T
(x < (y + z))	T	-	-	-	F	T	T	T	T	T	F	F	T	F	F	T	F	T	F
(y < (x + z))	T	-	-	-	T	F	T	T	T	T	F	T	F	F	T	F	F	T	T
(z < (x + y))	T	-	-	-	T	T	F	T	T	T	T	F	F	F	T	T	T	F	F
(x == y)	T	-	-	-	T	T	T	F	T	T	T	T	T	T	F	F	F	F	F
(y == z)	T	-	-	-	T	T	T	T	F	T	T	T	T	T	T	T	T	T	T
(x == z)	T	-	-	-	T	T	T	T	T	F	T	T	T	T	T	T	T	T	T
Impossible					X	X	X	X	X	X	X	X	X	X	X	X	X	X	X
Invoking Web service	X																		
Not invoking Web service		X	X	X															
Other actions																			
($TriangleResult$ is Equilateral)	X																		
($TriangleResult$ is Scalene)																			
($TriangleResult$ is a Triangle)	X																		
($TriangleResult$ is Isosceles)																			

ตารางที่ ข.1. ตารางตัดสินใจสำหรับเว็บเซอร์วิสที่ให้บริการตรวจสอบชนิดของรูปสามเหลี่ยม (ต่อ)

	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36
(x > 0)	T	T	T	T	T	T	T	T	T	T	T	T	T	T	T	T	T
(y > 0)	T	T	T	T	T	T	T	T	T	T	T	T	T	T	T	T	T
(z > 0)	T	T	T	T	T	T	T	T	T	T	T	T	T	T	T	T	T
(x < (y + z))	T	F	F	T	F	T	F	T	F	T	F	T	F	T	F	T	F
(y < (x + z))	F	F	T	F	F	T	T	F	F	T	T	F	F	T	T	F	F
(z < (x + y))	F	F	T	T	T	F	F	F	F	T	T	T	T	F	F	F	F
(x == y)	F	F	T	T	T	T	T	T	T	F	F	F	F	F	F	F	F
(y == z)	T	T	F	F	F	F	F	F	F	F	F	F	F	F	F	F	F
(x == z)	T	T	T	T	T	T	T	T	T	T	T	T	T	T	T	T	T
Impossible	X	X	X	X	X	X	X	X	X		X		X	X	X	X	X
Invoking Web service										X		X					
Not invoking Web service																	
Other actions												X					
($TriangleResult$ is Equilateral)																	
($TriangleResult$ is Scalene)																	
($TriangleResult$ is a Triangle)										X							
($TriangleResult$ is Isosceles)										X							

ตารางที่ ข.1. ตารางตัดสินใจสำหรับเว็บเซอร์วิสที่ให้บริการตรวจสอบชนิดของรูปสามเหลี่ยม (ต่อ)

	37	38	39	40	41	42	43	44	45	46	47	48	49	50	51	52	53	54
(x > 0)	T	T	T	T	T	T	T	T	T	T	T	T	T	T	T	T	T	T
(y > 0)	T	T	T	T	T	T	T	T	T	T	T	T	T	T	T	T	T	T
(z > 0)	T	T	T	T	T	T	T	T	T	T	T	T	T	T	T	T	T	T
(x < (y + z))	F	T	F	T	F	T	F	T	F	T	F	T	F	T	F	T	F	T
(y < (x + z))	T	F	F	T	T	F	F	T	T	F	F	T	T	F	F	T	T	F
(z < (x + y))	T	T	T	F	F	F	F	T	T	T	T	F	F	F	F	T	T	T
(x == y)	T	T	T	T	T	T	T	F	F	F	F	F	F	F	F	T	T	T
(y == z)	T	T	T	T	T	T	T	T	T	T	T	T	T	T	T	F	F	F
(x == z)	F	F	F	F	F	F	F	F	F	F	F	F	F	F	F	F	F	F
Impossible	X	X	X	X	X	X	X			X	X	X	X	X	X		X	X
Invoking Web service								X	X							X		
Not invoking Web service																		
Other actions									X									
($TriangleResult$ is Equilateral)																		
($TriangleResult$ is Scalene)																		
($TriangleResult$ is a Triangle)								X								X		
($TriangleResult$ is Isosceles)								X								X		

ตารางที่ ข.1. ตารางตัดสินใจสำหรับเว็บเซอร์วิสที่ให้บริการตรวจสอบชนิดของรูปสามเหลี่ยม (ต่อ)

	55	56	57	58	59	60
(x > 0)	T	T	T	T	T	T
(y > 0)	T	T	T	T	T	T
(z > 0)	T	T	T	T	T	T
(x < (y + z))	F	T	F	T	F	T
(y < (x + z))	F	T	T	F	F	T
(z < (x + y))	T	F	F	F	F	T
(x == y)	T	T	T	T	T	F
(y == z)	F	F	F	F	F	F
(x == z)	F	F	F	F	F	F
Impossible	X		X	X	X	
Invoking Web service		X				X
Not invoking Web service						
Other actions		X				
($TriangleResult$ is Equilateral)						
($TriangleResult$ is Scalene)						X
($TriangleResult$ is a Triangle)						X
($TriangleResult$ is Isosceles)						

ข.2. เว็บเซอร์วิสที่ให้บริการตรวจสอบชนิดของรูปสี่เหลี่ยม

ตารางที่ ข.2. ตารางตัดสินใจสำหรับเว็บเซอร์วิสที่ให้บริการตรวจสอบชนิดของรูปสี่เหลี่ยม

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17
(x > 0)	T	F	T	T	T	T	T	T	T	T	T	T	T	T	T	T	T
(y > 0)	T	-	F	T	T	T	T	T	T	T	T	T	T	T	T	T	T
(w > 0)	T	-	-	F	T	T	T	T	T	T	T	T	T	T	T	T	T
(z > 0)	T	-	-	-	F	T	T	T	T	T	T	T	T	T	T	T	T
(u > 0)	T	-	-	-	-	F	T	T	T	T	T	T	T	T	T	T	T
(v > 0)	T	-	-	-	-	-	F	T	T	T	T	T	T	T	T	T	T
(x == y)	T	-	-	-	-	-	-	F	T	T	T	F	F	T	F	F	T
(y == z)	T	-	-	-	-	-	-	T	F	T	T	F	T	F	F	T	F
(w == z)	T	-	-	-	-	-	-	T	T	F	T	T	F	F	F	T	T
(u == v)	T	-	-	-	-	-	-	T	T	T	F	T	T	T	T	F	F
Impossible																	
Invoking Web service	X							X	X	X	X	X	X	X	X	X	X
Not invoking Web service		X	X	X	X	X	X										
Other actions								X		X		X	X	X		X	
($RectangleResult$ is Square)	X																
($RectangleResult$ is Diamond)											X						
($RectangleResult$ is Rectangle)									X								
($RectangleResult$ is Parallelogram)																	X
($RectangleResult$ is Trapezium)															X		

ข.3. เว็บเซอร์วิสที่ให้บริการการหาค่ากลาง

ตารางที่ ข.3. ตารางตัดสินใจสำหรับเว็บเซอร์วิสที่ให้บริการการหาค่ากลาง

	1	2	3	4	5	6	7
(x > 0)	T	F	T	T	T	T	T
(y > 0)	T	-	F	T	T	T	T
(z > 0)	T	-	-	F	T	T	T
(y < z)	T	-	-	-	F	T	T
(x < y)	T	-	-	-	T	F	T
(x < z)	T	-	-	-	T	T	F
Impossible							X
Invoking Web service	X				X	X	
Not invoking Web service		X	X	X			
Other actions							
($MiddleResult$ is y)	X						
($MiddleResult$ is x)						X	
($MiddleResult$ is z)					X		

ภาคผนวก ค
ตัวอย่างกรณีทดสอบ

ค.1. เว็บเซอร์วิสที่ให้บริการตรวจสอบชนิดของรูปสามเหลี่ยม

Test Case ID : 1

Operation : TriType

Input

Name	Value
x	1
y	1
z	1
Expected Results	Invoking Web service ($TriangleResult$ is Equilateral) ($TriangleResult$ is a Triangle)

รูปที่ ค.1. กรณีทดสอบหมายเลข 1 ของเว็บเซอร์วิสที่ให้บริการตรวจสอบชนิดของรูปสามเหลี่ยม

Test Case ID : 2

Operation : TriType

Input

Name	Value
x	-999999999
Expected Results	Not invoking Web service

รูปที่ ค.2. กรณีทดสอบหมายเลข 2 ของเว็บเซอร์วิสที่ให้บริการตรวจสอบชนิดของรูปสามเหลี่ยม

Test Case ID : 3

Operation : TriType

Input

Name	Value
x	1
y	-999999999
Expected Results	Not invoking Web service

รูปที่ ค.3. กรณีทดสอบหมายเลข 3 ของเว็บเซอร์วิสที่ให้บริการตรวจสอบชนิดของรูปสามเหลี่ยม

Test Case ID : 4

Operation : TriType

Input

Name	Value
x	1
y	1
z	-999999999
Expected Results	Not invoking Web service

รูปที่ ค.4. กรณีทดสอบหมายเลข 4 ของเว็บเซอร์วิสที่ให้บริการตรวจสอบชนิดของรูปสามเหลี่ยม

Test Case ID : 5

Operation : TriType

Input

Name	Value
x	2.0000
y	1.0000
z	2.0000
Expected Results	Invoking Web service ($TriangleResult$ is a Triangle) ($TriangleResult$ is Isosceles)

รูปที่ ค.5. กรณีทดสอบหมายเลข 5 ของเว็บเซอร์วิสที่ให้บริการตรวจสอบชนิดของรูปสามเหลี่ยม

Test Case ID : 6

Operation : TriType

Input

Name	Value
x	1.0000
y	2.0000
z	1.0000
Expected Results	Invoking Web service Other actions

รูปที่ ค.6. กรณีทดสอบหมายเลข 6 ของเว็บเซอร์วิสที่ให้บริการตรวจสอบชนิดของรูปสามเหลี่ยม

Test Case ID : 7

Operation : TriType

Input

Name	Value
x	3.0000
y	2.0000
z	2.0000
Expected Results	Invoking Web service ($TriangleResult$ is a Triangle) ($TriangleResult$ is Isosceles)

รูปที่ ค.7. กรณีทดสอบหมายเลข 7 ของเว็บเซอร์วิสที่ให้บริการตรวจสอบชนิดของรูปสามเหลี่ยม

Test Case ID : 8

Operation : TriType

Input

Name	Value
x	2.0000
y	1.0000
z	1.0000
Expected Results	Invoking Web service Other actions

รูปที่ ค.8. กรณีทดสอบหมายเลข 8 ของเว็บเซอร์วิสที่ให้บริการตรวจสอบชนิดของรูปสามเหลี่ยม

Test Case ID : 9

Operation : TriType

Input

Name	Value
x	2.0000
y	2.0000
z	1.0000
Expected Results	Invoking Web service ($TriangleResult$ is a Triangle) ($TriangleResult$ is Isosceles)

รูปที่ ค.9. กรณีทดสอบหมายเลข 9 ของเว็บเซอร์วิสที่ให้บริการตรวจสอบชนิดของรูปสามเหลี่ยม

Test Case ID : 10

Operation : TriType

Input

Name	Value
x	1.0000
y	1.0000
z	2.0000
Expected Results	Invoking Web service Other actions

รูปที่ ค.10. กรณีทดสอบหมายเลข 10 ของเว็บเซอร์วิสที่ให้บริการตรวจสอบชนิดของรูปสามเหลี่ยม

Test Case ID : 11

Operation : TriType

Input

Name	Value
x	4
y	3
z	2
Expected Results	Invoking Web service ($TriangleResult$ is Scalene) ($TriangleResult$ is a Triangle)

รูปที่ ค.11. กรณีทดสอบหมายเลข 11 ของเว็บเซอร์วิสที่ให้บริการตรวจสอบชนิดของรูปสามเหลี่ยม

ค.2. เว็บเซอร์วิสที่ให้บริการตรวจสอบชนิดของรูปสี่เหลี่ยม

Test Case ID : 1

Operation : RectangleType

Input

Name	Value
x	1
y	1
w	1
z	1
u	1
v	1
Expected Results	Invoking Web service ($RectangleResult$ is Square)

รูปที่ ค.12. กรณีทดสอบหมายเลข 1 ของเว็บเซอร์วิสที่ให้บริการตรวจสอบชนิดของรูปสี่เหลี่ยม

Test Case ID : 2

Operation : RectangleType

Input

Name	Value
x	-999999999
Expected Results	Not invoking Web service

รูปที่ ค.13. กรณีทดสอบหมายเลข 2 ของเว็บเซอร์วิสที่ให้บริการตรวจสอบชนิดของรูปสี่เหลี่ยม

Test Case ID : 3

Operation : RectangleType

Input

Name	Value
x	1
y	-999999999
Expected Results	Not invoking Web service

รูปที่ ค.14. กรณีทดสอบหมายเลข 3 ของเว็บเซอร์วิสที่ให้บริการตรวจสอบชนิดของรูปสี่เหลี่ยม

Test Case ID : 4

Operation : RectangleType

Input

Name	Value
x	1
y	1
w	-999999999
Expected Results	Not invoking Web service

รูปที่ ค.15. กรณีทดสอบหมายเลข 4 ของเว็บเซอร์วิสที่ให้บริการตรวจสอบชนิดของรูปสี่เหลี่ยม

Test Case ID : 5

Operation : RectangleType

Input

Name	Value
x	1
y	1
w	1
z	-999999999
Expected Results	Not invoking Web service

รูปที่ ค.16. กรณีทดสอบหมายเลข 5 ของเว็บเซอร์วิสที่ให้บริการตรวจสอบชนิดของรูปสี่เหลี่ยม

Test Case ID : 6

Operation : RectangleType

Input

Name	Value
x	1
y	1
w	1
z	1
u	-999999999
Expected Results	Not invoking Web service

รูปที่ ค.17. กรณีทดสอบหมายเลข 6 ของเว็บเซอร์วิสที่ให้บริการตรวจสอบชนิดของรูปสี่เหลี่ยม

Test Case ID : 7

Operation : RectangleType

Input

Name	Value
x	1
y	1
w	1
z	1
u	1
v	-999999999
Expected Results	Not invoking Web service

รูปที่ ค.18. กรณีทดสอบหมายเลข 7 ของเว็บเซอร์วิสที่ให้บริการตรวจสอบชนิดของรูปสี่เหลี่ยม

Test Case ID : 8

Operation : RectangleType

Input

Name	Value
x	2
y	1
w	1
z	1
u	1
v	1
Expected Results	Invoking Web service Other actions

รูปที่ ค.19. กรณีทดสอบหมายเลข 8 ของเว็บเซอร์วิสที่ให้บริการตรวจสอบชนิดของรูปสี่เหลี่ยม

Test Case ID : 9

Operation : RectangleType

Input

Name	Value
x	2.0000
y	2.0000
w	1.0000
z	1.0000
u	1.0000
v	1.0000
Expected Results	Invoking Web service ($RectangleResult$ is Rectangle)

รูปที่ ค.20. กรณีทดสอบหมายเลข 9 ของเว็บเซอร์วิสที่ให้บริการตรวจสอบชนิดของรูปสี่เหลี่ยม

Test Case ID : 10

Operation : RectangleType

Input

Name	Value
x	1
y	1
w	2
z	1
u	1
v	1
Expected Results	Invoking Web service Other actions

รูปที่ ค.21. กรณีทดสอบหมายเลข 10 ของเว็บเซอร์วิสที่ให้บริการตรวจสอบชนิดของรูปสี่เหลี่ยม

Test Case ID : 11

Operation : RectangleType

Input

Name	Value
x	1.0000
y	1.0000
w	1.0000
z	1.0000
u	2.0000
v	1.0000
Expected Results	Invoking Web service ($RectangleResult$ is Diamond)

รูปที่ ค.22. กรณีทดสอบหมายเลข 11 ของเว็บเซอร์วิสที่ให้บริการตรวจสอบชนิดของรูปสี่เหลี่ยม

Test Case ID : 12

Operation : RectangleType

Input

Name	Value
x	3.0000
y	2.0000
w	1.0000
z	1.0000
u	1.0000
v	1.0000
Expected Results	Invoking Web service Other actions

รูปที่ ค.23. กรณีทดสอบหมายเลข 12 ของเว็บเซอร์วิสที่ให้บริการตรวจสอบชนิดของรูปสี่เหลี่ยม

Test Case ID : 13

Operation : RectangleType

Input

Name	Value
x	2.0000
y	1.0000
w	2.0000
z	1.0000
u	1.0000
v	1.0000
Expected Results	Invoking Web service Other actions

รูปที่ ค.24. กรณีทดสอบหมายเลข 13 ของเว็บเซอร์วิสที่ให้บริการตรวจสอบชนิดของรูปสี่เหลี่ยม

Test Case ID : 14

Operation : RectangleType

Input

Name	Value
x	2.0000
y	2.0000
w	2.0000
z	1.0000
u	1.0000
v	1.0000
Expected Results	Invoking Web service Other actions

รูปที่ ค.25. กรณีทดสอบหมายเลข 14 ของเว็บเซอร์วิสที่ให้บริการตรวจสอบชนิดของรูปสี่เหลี่ยม

Test Case ID : 15

Operation : RectangleType

Input

Name	Value
x	3.0000
y	2.0000
w	2.0000
z	1.0000
u	1.0000
v	1.0000
Expected Results	Invoking Web service ($RectangleResult$ is Trapezium)

รูปที่ ค.26. กรณีทดสอบหมายเลข 15 ของเว็บเซอร์วิสที่ให้บริการตรวจสอบชนิดของรูปสี่เหลี่ยม

Test Case ID : 16

Operation : RectangleType

Input

Name	Value
x	2.0000
y	1.0000
w	1.0000
z	1.0000
u	2.0000
v	1.0000
Expected Results	Invoking Web service Other actions

รูปที่ ค.27. กรณีทดสอบหมายเลข 16 ของเว็บเซอร์วิสที่ให้บริการตรวจสอบชนิดของรูปสี่เหลี่ยม

Test Case ID : 17

Operation : RectangleType

Input

Name	Value
x	2.0000
y	2.0000
w	1.0000
z	1.0000
u	2.0000
v	1.0000
Expected Results	Invoking Web service ($RectangleResult$ is Parallelogram)

รูปที่ ค.28. กรณีทดสอบหมายเลข 17 ของเว็บเซอร์วิสที่ให้บริการตรวจสอบชนิดของรูปสี่เหลี่ยม

ค.3. เว็บเซอร์วิสที่ให้บริการการหาค่ากลาง

Test Case ID : 1

Operation : MiddleValue

Input

Name	Value
x	1.0000
y	2.0000
z	3.0000
Expected Results	Invoking Web service ($MiddleResult$ is y)

รูปที่ ค.29. กรณีทดสอบหมายเลข 1 ของเว็บเซอร์วิสที่ให้บริการการหาค่ากลาง

Test Case ID : 2

Operation : MiddleValue

Input

Name	Value
x	-999999999
Expected Results	Not invoking Web service

รูปที่ ค.30. กรณีทดสอบหมายเลข 2 ของเว็บเซอร์วิสที่ให้บริการการหาค่ากลาง

Test Case ID : 3

Operation : MiddleValue

Input

Name	Value
x	1
y	-999999999
Expected Results	Not invoking Web service

รูปที่ ค.31. กรณีทดสอบหมายเลข 3 ของเว็บเซอร์วิสที่ให้บริการการหาค่ากลาง

Test Case ID : 4

Operation : MiddleValue

Input

Name	Value
x	1
y	1
z	-999999999
Expected Results	Not invoking Web service

รูปที่ ค.32. กรณีทดสอบหมายเลข 4 ของเว็บเซอร์วิสที่ให้บริการการหาค่ากลาง

Test Case ID : 5

Operation : MiddleValue

Input

Name	Value
x	1.0000
y	2.0000
z	2.0000
Expected Results	Invoking Web service ($MiddleResult$ is z)

รูปที่ ค.33. กรณีทดสอบหมายเลข 5 ของเว็บเซอร์วิสที่ให้บริการการหาค่ากลาง

Test Case ID : 6

Operation : MiddleValue

Input

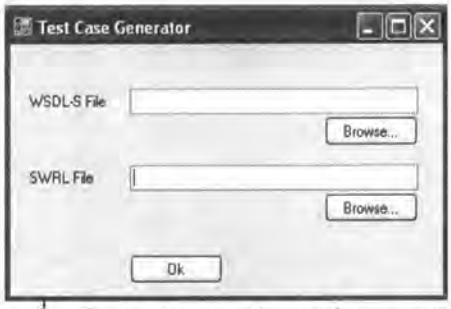
Name	Value
x	1.0000
y	1.0000
z	2.0000
Expected Results	Invoking Web service ($MiddleResult$ is x)

รูปที่ ค.34. กรณีทดสอบหมายเลข 6 ของเว็บเซอร์วิสที่ให้บริการการหาค่ากลาง

ภาคผนวก ง
คู่มือการใช้งาน

รายละเอียดขั้นตอนการใช้งานเครื่องมือสำหรับการสร้างกรณีทดสอบสำหรับเว็บเซอร์วิสจากดับเบิลยูเอสดีแอล-เอส และเอสดับเบิลยูอาร์แอลด้วยตารางตัดสินใจที่นำเสนอในงานวิจัยนี้ มีดังนี้

1. เปิดเครื่องมือสำหรับการสร้างกรณีทดสอบสำหรับเว็บเซอร์วิสจากดับเบิลยูเอสดีแอล-เอส และเอสดับเบิลยูอาร์แอลด้วยตารางตัดสินใจ จะพบหน้าจอดังรูปที่ ง.1



รูปที่ ง.1. หน้าจอเครื่องมือสำหรับการสร้างกรณีทดสอบสำหรับเว็บเซอร์วิส

2. ระบุตำแหน่งของไฟล์เอกสารดับเบิลยูเอสดีแอล-เอส และเอกสารเอสดับเบิลยูอาร์แอล จากตั้นกดปุ่ม OK

3. เครื่องมือจะทำการวิเคราะห์ข้อมูล และสร้างกรณีทดสอบสำหรับเว็บเซอร์วิส

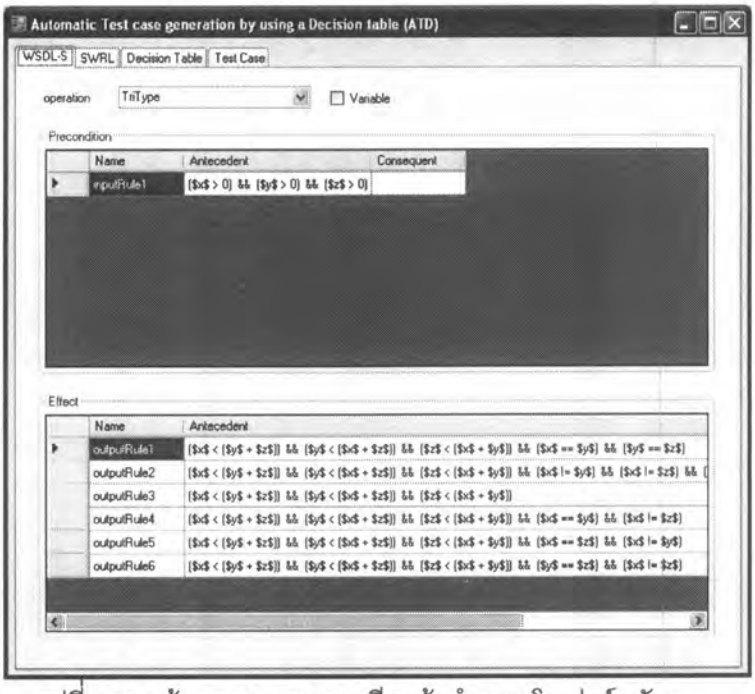
4. เมื่อเครื่องมือทำการเครื่องมือจะทำการวิเคราะห์ข้อมูล และสร้างกรณีทดสอบสำหรับเว็บเซอร์วิสเสร็จเรียบร้อย เครื่องมือจะแสดงหน้าจอดังรูปที่ ง.2 ซึ่งผู้ใช้งานสามารถเรียกดูรายละเอียดต่างๆ โดยการเลือกแท็บเมนู ดังนี้

4.1. WSDL-S เป็นแท็บเมนูที่จะแสดงรายละเอียดข้อกำหนดโอเปอร์เรชันต่างๆ ในเว็บเซอร์วิส

4.2. SWRL เป็นแท็บเมนูที่จะแสดงรายละเอียดกฎที่นิยามไว้ในเอกสารเอสดับเบิลยูอาร์แอล

4.3. Decision Table เป็นแท็บเมนูที่จะแสดงรายละเอียดของตารางตัดสินใจที่สร้างขึ้น เพื่อใช้เป็นข้อมูลนำเข้าสำหรับการสร้างกรณีทดสอบ

4.4. Test Case เป็นแท็บเมนูที่จะแสดงรายละเอียดของกรณีทดสอบที่ถูกสร้างขึ้นด้วยเครื่องมือการสร้างกรณีทดสอบสำหรับเว็บเซอร์วิส



รูปที่ ง.2. หน้าจอแสดงรายละเอียดข้อกำหนดโอเปอร์เรชัน

ภาคผนวก จ
บทความวิจัย

The Eighth International Conference on Quality Software

Web Service Test Case Generation Based on Decision Table

Siripol Noikajana and Taratip Suwannasart
Software Engineering Laboratory
Department of Computer Engineering, Faculty of Engineering
Chulalongkorn University, Bangkok 10330 Thailand
Siripol.N@student.chula.ac.th, Taratip.S@chula.ac.th

Abstract

Software testing [1] consumes at least 50% of labor and its cost approximates 50-70% of software cost which it may cause software delay. To date, web service is a popular way of implementing a Service-Oriented Architecture (SOA), which has gained rapid adoption and support from leading companies in industry [2]. Most of researches, test case is generated from web service description only based on black box testing which its limitation is a large number of generated test cases. This paper proposes a methodology to generate web service test case based on decision table. The generated test case corresponds to web service's requirement and web service description which are defined in WSDL-S and SWRL.

Keywords: Software testing, Decision table, WSDL-S, SWRL.

1. Introduction

Software testing [3] is the process of analyzing a software item to detect the differences between existing and required conditions and to evaluate the features of the software item. Unfortunately, it is impossible to completely test a nontrivial system. Whittaker [4] proposed to approach testing in four phases: modeling the software's environment, selecting test scenarios, running and evaluating test scenarios and measuring testing progress. This paper proposes a methodology to automatically generate web service test case in selecting test scenarios phase which helps to decrease time and effort for testing software.

Definition of web services are described by using Web Services Description Language (WSDL) [5-8] that is a XML document. WSDL contains all the information about service capabilities and invocation mechanisms. The capabilities are described in terms of the operations of the service and the input and output messages for each operation. Unfortunately, the current WSDL standard [9, 10] operates at the syntactic level and lacks the semantic expressivity needed to represent the requirements and capabilities of web services. Then, W3C proposed Web Service Semantics (WSDL-S) in order to improve web service description and identify semantic annotation for web service such as precondition, post-condition, etc.

The remainder of this paper is organized as follows: Section 2, we introduce background related this paper: decision table, SWRL, and WSDL-S. Section 3, we explain our methodology to generate web service test case from SWRL and WSDL-S based on limited entry decision table and demonstrate *TAD (Testing by Automatically generate Decision table)* to generate test case for Rectangle web service. Section 4, we introduce shortly related work. Finally, conclusion and future work are drawn in Section 5.

2. Background

2.1. Semantic Web Rule Language

The semantic web rule language (SWRL) [11, 12] represents rule language for semantic web which consists antecedent (body) and consequent (head), such as a form

$$antecedent \rightarrow consequent \quad (1)$$

The meaning of equation (1) is the conditions specified in the antecedent hold, and then the conditions specified in the consequent must also hold. It can be written as an logic expression with positive conjunctions only [12]. The XML concrete syntax [11] is a combination of the OWL Web Ontology Language XML Presentation Syntax with the RuleML XML Syntax that is XML document according to the definition in [11]. This paper expresses conditions for precondition and effect according to the XML concrete

1550-6002/08 $25.00 © 2008 IEEE
DOI 10.1109/QSIC.2008.7

syntax pattern which is referred by using modelReference attribute [10] in WSDL-S.

2.2. Web Service Semantics

The Web Service Semantics (WSDL-S) [10, 13-15] aims to add semantic annotation to web service description by extending WSDL that is relatively easy to update the existing tooling around WSDL. The WSDL extension consists of two elements, precondition and effect, and three attributes, modelReference, schemaMapping and category. This paper expresses operation based on WSDL-S which consists of input, output, precondition, and post-condition (effect).

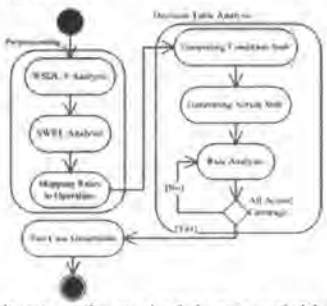


Figure 1. The methodology's activities

2.3. Decision Table [1]

A decision table has been used to represent and analyze complex logical relationships. It is ideal for describing situations in which a number of combinations of actions are taken under varying sets of conditions.

A decision table has four portions (Table 1): the part to the left of the bold vertical line is the stub portion; to the right is the entry portion. The part above the bold line is the condition portion, and below is the action portion. Thus, it can be refer to the condition stub, the condition entries, the action stub, and the action entries. A column in the entry portion is a rule. Rules indicate which actions are taken for the conditional circumstances indicated in the condition portion of the rule.

Decision table represents two perceptions, a limited entry decision tables and an extended entry decision tables. All conditions of limited entry decision tables have binary values which are true (T) or false (F), whereas all conditions of extended entry decision tables contain several values.

To identify test cases with decision tables, it should be interpreted conditions as inputs and actions as outputs. Sometimes conditions end up referring to equivalence classes of inputs, actions refer to major functional processing portions of the item tested and rules refer to each column in entries. The rules are then interpreted as test cases. The equivalence classes form a partition of a set, where partition refers to a collection of mutually disjoint subsets when the union is the entire set. Thus, if we pick up some value in a partition that detects defect then other values in same partition can be detected alike defect. Thus, equivalence classes help to decrease a number of test cases by selecting one value in each partition.

Table 1. Decision table structure

Condition stub	Condition entries				
Action stub	Action entries				

3. Methodology

This paper contains a methodology to automatically generate web service test case from WSDL-S and SWRL based on limited decision table that decision table identifies test case. Conditions should be interpreted with both antecedents of precondition and post-condition. Actions represent both consequents of precondition and post-condition. These conditions must be impendence. Figure 1 shows methodology's activities which consist of preprocessing, decision table analysis, and test case generation.

This paper presents *TAD (Testing by Automatically generate Decision table)* that is a prototype tool implemented using by C#. TAD examines Rectangle web service to generate web service test case. The definition of Rectangle web service via WSDL-S that consists of RectangleType operation. It serves examining type of rectangle: square, diamond, oblong, parallelogram, and trapezium. This operation requires four parameters: x, y, w, and z. These parameters are side of rectangle that is integer. An x side is an opposite side of y side, and w side is an opposite side of z side.

The methodology's activities are as follows:

3.1. Preprocessing

Preprocessing prepares data to generate web service test case that requires two input documents, SWRL, and WSDL-S. Preprocessing consists of three sub-activities: WSDL-S analysis, SWRL analysis, and mapping rules to operation. The result of WSDL-S analysis is operation definition that consists of operation name, data types, precondition, and post-

condition. The result of SWRL analysis is logic expressions which may be referred from both precondition and post-condition of operation. Therefore, both precondition and post-condition consist of logic expression by mapping rules to operation in latest sub-activity.

3.1.1. WSDL-S analysis. An input WSDL-S document describes web service description which is a semantic annotation. This paper discusses these semantic annotations which are two analysis prescriptions: data type definition and operation definition. First, TAD extracts data type definition and operation definition from WSDL-S. Next, TAD redefines data type definition into generic data type definition as variable, class, or enumeration. Finally, TAD binds data types to operation by using message type.

The basic algorithm of operation definition analysis is as follows:

1. Analyze the hierarchical tree structure of operation element node.
2. Traverse the tree, and at each tree node:
   a. If it is an input element node, generate data type definition for input parameters.
   b. If it is an output element node, generate data type definition for output parameters.
   c. If it is a precondition element node, extract rule's URIs of precondition from modelReference attribute.
   d. If it is an effect element node, extract referred rule's URIs of post-condition from modelReference attribute.
3. Keep operation definition into database.

In step 2.a and 2.b, the process is generated data type definition which consequence from binding message type to operation.

3.1.2. SWRL analysis. An input SWRL document defines rules which are referred by modelReference attribute in WSDL-S document. The rule definition consists of two parts, antecedent and consequent, each of which consists of either a set of atoms or an empty set of atom. A set of atoms can be extracted to a logic expression for defining precondition and post-condition which both conditions relate to input and output of operation.

TAD retrieves input SWRL documents with specific URIs are defined in either precondition or post-condition for each operation before perform to extract rule.

The basic algorithm of SWRL analysis is as follows:

1. Find ruleml:imp element node.
2. Retrieve rule name from rdf:ID attribute.
3. Analyze the hierarchical tree structure of ruleml:imp element node.
4. Traverse the tree, and at each tree node:
   a. If it is a ruleml:_body element node, generate a logic expression of antecedent part.
   b. If it is a ruleml:_head element node, generate a logic expression of consequent part.
5. Keep rule name and logic expression of antecedent part and consequent part into database.
6. Find next ruleml:imp element node and go to step 2. If next ruleml:imp element node is not exist, exit analysis.

In step 4.a and 4.b, the process is generated logic expression of antecedent part and consequent part by analyzing a set of atoms and interpreting operator according SWRL built-ins. After that, it normalizes logic expression to generic logic expression.

This paper demonstrates example to analyze SWRL rule that consists a non-generic logic expression shown in figure 2. Figure 2 presents outputRule1 definition that is constraint of a square rectangle. TAD must extracts generic logic expression from the rule as follows

$$(x == y) \text{ and } (y == z) \text{ and } (w == z) \text{ and } (x^2 + z^2 == w^2 + y^2) \quad (2)$$

3.1.3. Mapping rules to operation. This activity retrieves information of two previous activities from database. After that, rules are mapped to each operation definition.

The basic algorithm of mapping rules to operation is as follows:

1. Retrieve information that consists of SWRL analysis and WSDL-S analysis.
2. Analyze the rule's URIs list of precondition.
3. Traverse the list, and at each item by mapping rules from SWRL analysis to item.
4. Analyze the rule's URIs list of post-condition and repeat step 3.

The result of Rectangle web service preprocessing that shows as table 2.

3.2. Decision Table Analysis

In this activity, it consists of three sub-activates: generating condition stub, generating action stub, and rule analysis. For generating condition stub, TAD generates conditions from operation definition which possibly finds a redundant condition. Then, TAD generates conditions by referring to equivalence class. For action stub, it always includes two actions, "Impossible" and "Invoking web service". Impossible action is taken when TAD cannot define any actions

according to generated conditions or some conditions value conflicts with other conditions. Invoking web service action is taken when all conditions, defined in precondition, are true.

The basic algorithm of condition and action stub generation is as follows:

1. Retrieve operation definition.
2. Analyze the rules of precondition.
3. Traverse the rules, and at each atom of logic expression:
   a. If atom negates other atom in condition stub, ignore it.
   b. If atom does not negate other atom and atom has not a transitive property, insert it into condition stub.
4. Insert impossible action and invoking web service action into action stub.
5. Analyze the rule of post-condition.
6. Traverse the rule and insert each rule into action stub.

```
<ruleml:imp rdf:ID="outputRule1">
 <ruleml:_rlab ruleml:href="#outputRule1"/>
 <ruleml:_body>
  <swrlx:builtinAtom swrlx:builtin="swrlb:pow">
   <ruleml:var>powerOfX</ruleml:var>
   <ruleml:var>x</ruleml:var>
   <ruleml:var>2</ruleml:var>
  </swrlx:builtinAtom>
  <swrlx:builtinAtom swrlx:builtin="swrlb:pow">
   <ruleml:var>powerOfZ</ruleml:var>
   <ruleml:var>z</ruleml:var>
   <ruleml:var>2</ruleml:var>
  </swrlx:builtinAtom>
  <swrlx:builtinAtom swrlx:builtin="swrlb:pow">
   <ruleml:var>powerOfW</ruleml:var>
   <ruleml:var>w</ruleml:var>
   <ruleml:var>2</ruleml:var>
  </swrlx:builtinAtom>
  <swrlx:builtinAtom swrlx:builtin="swrlb:pow">
   <ruleml:var>powerOfY</ruleml:var>
   <ruleml:var>y</ruleml:var>
   <ruleml:var>2</ruleml:var>
  </swrlx:builtinAtom>
  <swrlx:builtinAtom swrlx:builtin="swrlb:add">
   <ruleml:var>sumOfX_Z</ruleml:var>
   <ruleml:var>powerOfX</ruleml:var>
   <ruleml:var>powerOfZ</ruleml:var>
  </swrlx:builtinAtom>
  <swrlx:builtinAtom swrlx:builtin="swrlb:add">
   <ruleml:var>sumOfW_Y</ruleml:var>
   <ruleml:var>powerOfW</ruleml:var>
   <ruleml:var>powerOfY</ruleml:var>
  </swrlx:builtinAtom>
  <swrlx:builtinAtom swrlx:builtin="swrlb:equal">
   <ruleml:var>sumOfX_Z</ruleml:var>
   <ruleml:var>sumOfW_Y</ruleml:var>
  </swrlx:builtinAtom>
  <swrlx:builtinAtom swrlx:builtin="swrlb:equal">
   <ruleml:var>x</ruleml:var>
   <ruleml:var>y</ruleml:var>
  </swrlx:builtinAtom>
  <swrlx:builtinAtom swrlx:builtin="swrlb:equal">
   <ruleml:var>y</ruleml:var>
   <ruleml:var>z</ruleml:var>
  </swrlx:builtinAtom>
  <swrlx:builtinAtom swrlx:builtin="swrlb:equal">
   <ruleml:var>w</ruleml:var>
   <ruleml:var>z</ruleml:var>
  </swrlx:builtinAtom>
 </ruleml:_body>
 <ruleml:_head>
  <swrlx:datavaluedPropertyAtom swrlx:property="is">
   <ruleml:var>RectangleResult</ruleml:var>
   <owlx:DataValue owlx:datatype="xsd:string">
    a Square
   </owlx:DataValue>
  </swrlx:datavaluedPropertyAtom>
 </ruleml:_head>
</ruleml:imp>
```

Figure 2. An outputRule1 Definition

Table 2. Mapping rules to RectangleType operation definition

Operation	RectangleType		
Input data	Variable	Data type	
	X	Integer	
	Y	Integer	
	W	Integer	
	Z	Integer	
Output data	Variable	Data type	
	RectangleResult	String	
Precondition	Rule	Antecedent	Consequent
	inputRule1	(x > 0) and (y > 0) and (w > 0) and (z > 0)	
Post-condition	Rule	Antecedent	Consequent
	outputRule1	(x == y) and (y == z) and (w == z) and ($x^2 + z^2 == w^2 + y^2$)	RectangleResult is a Square
	outputRule2	(x == y) and (y == z) and (w == z) and ($x^2 + z^2 <> w^2 + y^2$)	RectangleResult is a Diamond
	outputRule3	(x == y) and (y <> z) and (w == z) and ($x^2 + z^2 == w^2 + y^2$)	RectangleResult is an Oblong
	outputRule4	(x == y) and (y <> z) and (w == z) and ($x^2 + z^2 <> w^2 + y^2$)	RectangleResult is a Parallelogram
	outputRule5	(x != y) and (y != z) and (w != z)	RectangleResult is a Trapezium

Subsequent to generate condition and action stub, TAD analyzes rules of decision table. For limited entry decision tables [1], if n conditions exist, there must be 2^n rules. These rules are applied to test case that is a large number of generated test cases. Then, we cannot execute completely test cases. This paper can be discussed how to generate a minimum number of rules.

The basic algorithm of rule analysis is as follows:

1. Define all conditions to be true.
2. Specify the first rule based on actions corresponding to the conditions in step 1.
3. For subsequent rules, change each condition to be false one by one and specify actions corresponding to the conditions.
4. If all actions are not preformed, add more rules until they are completed.

The result of Rectangle web service preprocessing displays in table 3.

3.3. Test Case Generation

Figure 3 represents the generated web service test case definition according to XML Schema, called Web service Test Case Specification (WTCS). WTCS can be

easily represented several formats upon application and inputted into test driver. This section discusses how to generate web service test case.

The basic algorithm of test case generation is as follows:

1. Retrieve information that consists of operation definition and decision table corresponding to operation.
2. Analysis the rule of decision table.
3. Generate test case for each rule if its action is not impossible. Data values must be corresponding to all conditions, defined in condition stub, and operation definition.

Table 3. Decision Table for RectangleType operation

	1	2	3	...	14	15
x > 0	T	F	T	...	T	T
y > 0	T	T	F	...	T	T
z > 0	T	T	T	...	T	T
x == y	T	T	T	...	T	T
y == z	T	T	T	...	F	F
w == z	T	T	T	...	F	T
$x^2 + z^2 == w^2 + y^2$	T	T	T	...	T	F
Impossible		x	x		x	
Invoking web service	x					x
RectangleResult is a Square	x					
RectangleResult is a Diamond						
RectangleResult is an Oblong						
RectangleResult is a Parallelogram						x
RectangleResult is a Trapezium						

```
<schema xmlns="http://www.w3.org/2001/XMLSchema">
  <element name="items">
    <complexType>
      <sequence>
        <element name="item" type="item" minOccurs="0"
          maxOccurs="unbounded" />
      </sequence>
    </complexType>
  </element>
  <complexType name="item">
    <sequence>
      <element name="input" type="dataType" minOccurs="0"
        maxOccurs="unbounded" />
      <element name="expected" type="string" minOccurs="0"
        maxOccurs="unbounded" />
    </sequence>
    <attribute name="operation" type="string" />
  </complexType>
  <complexType name="dataType">
    <sequence>
      <element name="variable" type="string" />
      <element name="value" type="string" />
    </sequence>
  </complexType>
</schema>
```

Figure 3. Web service Test Case Specification

4. Related Work

Many researches proposed to generate test case with decision table [16-20]. Pancharat *et al* [16] proposed to generate test case based on limited entry decision table for use case scenario. This generated test case assumption is a single fault that its results cover every expected result. Fangmei *et al* [17] proposed to test railway signaling computer control software based on dynamic decision table. The dynamic decision table generation system is a sequential apparatus, which provides the comparison information used to judge the correctness of the software under test according to the state of station signals and switches as well as trains' motions during the testing process. It helps to decrease the effort and increase efficiency. Vanthienen *et al* [18] proposed to apply decision table for verification and validation software. Nowadays, web service is a popular implementation. Its quality is an important key for serving. Thus, we need to test web service that consumes at least effort and time. To date, many researches proposed to generate web service test case[21-24]. This generated test case is inspired black box testing such as random testing and boundary testing. Dalal *et al* [23] presented AETG^{SM} web which employs a web based user interface to model the functional requirements of the System Under Test and automatically generates test case for the system. The AETG generation algorithms ensure that each level of each factor is tested at least once with every other level of every other factor, which is called pair-wise coverage of the input domain. Bai *et al* [24] proposed to generate web service test case from WSDL which consists of test data and operation flows for testing. Operation flows are analyzed dependency between operation such as input, input-output and output. This generated test case is a XML document which easy to reuse and represent upon application. These researches do not consider both precondition and post-condition of operation for generating test case. This paper proposes to generate web service test case which meets web service requirement, such as precondition and post-condition. This generated test case is inspired decision table because a number of test cases are few test cases which help to decrease the effort for testing web service.

5. Conclusion

This paper has presented a method to generate web service test case from web service's requirement and web service description, defined in WSDL-S and SWRL, based on limited entry decision table. Result of generated test cases shows that number of test cases has been decreased and still cover all expected outputs (action) according to web service's requirement. TAD helps to decrease the effort of generating test cases and a number of test cases.

For future work we plan to define precondition and post-condition which their logic expression supports negated expression and disjunction by using OCL.

6. Acknowledgement

This research is part of the Engineering New Paradigm Software for Enterprises with Service-Oriented Architecture Project, supported by Thailand's Software Industry Promotion Agency (Public Organization).

The authors are grateful to Dr. Mala Supongpan for her kindly making through useful comments and also extend their grateful to the Royal Bangkok Sports Club (RBSC) scholarship for partly funded for the project. Finally, thanks to Dr. Xiaoying Bai.

7. References

[1] P. Jorgensen, *Software Testing A Craftsman's Approach* 2ed: CRC PRESS, 2002.

[2] E. Martin, S. Basu, and T. Xie, "Automated Testing and Response Analysis of Web Services," presented at Web Services, 2007. ICWS 2007. IEEE International Conference on, 2007.

[3] "IEEE standard for software test documentation," *IEEE Std 829-1998*, 1998.

[4] J. A. Whittaker, "What is software testing? And why is it so hard?," *Software, IEEE*, vol. 17, pp. 70-79, 2000.

[5] W3Schools, "WSDL Tutorial."

[6] W3C Working Group, "Web Services Glossary," 2004.

[7] W3C Note, "Web Services Description Language (WSDL) 1.1," 2001.

[8] D. A. Andrea, "A Model-driven WSDL Extension for Describing the QoS ofWeb Services," presented at Web Services, 2006. ICWS '06. International Conference on, 2006.

[9] J. Miller, K. Verma, P. Rajasekaran, A. Sheth, R. Aggarwal, and K. Sivashanmugam, "WSDL-S: Adding Semantics to WSDL - White Paper."

[10] W3C Member Submission, "Web Service Semantics - WSDL-S," 2005.

[11] W3C Member Submission, "SWRL: A Semantic Web Rule Language Combining OWL and RuleML."

[12] Protege, "SWRL Language FAQ," 2007.

[13] P. Massimo and W. Matthias, "Grounding OWL-S in WSDL-S," presented at Web Services, 2006. ICWS '06. International Conference on, 2006.

[14] R. Akkiraju, J. Farrell, J.Miller, M. Nagarajan, M. Schmidt, A. Sheth, and K. Verma, "Web Service Semantics - WSDL-S."

[15] I. M. Diaconescu, "Methodologies and Tools for Modeling Rule-based Web Services," presented at Digital EcoSystems and Technologies Conference, 2007. DEST '07. Inaugural IEEE-IES, 2007.

[16] S. Pancharat and T. Suwannasart, "An Approach for Generating Test Cases from Use Cases Based on A Decision Table," presented at the 2006 National Computer Science and Engineering Conference (NCSEC'06), 2006.

[17] W. Fangmei and L. Meng, "Railway signaling safety-critical software testing based on dynamic decision table," presented at Test Symposium, 1999. (ATS '99) Proceedings. Eighth Asian, 1999.

[18] J. Vanthienen and E. Dries, "Illustration of a decision table tool for specifying and implementing knowledge based systems," presented at Tools with Artificial Intelligence, 1993. TAI '93. Proceedings., Fifth International Conference on, 1993.

[19] G. W. Oerter, "A New Implementation of Decision Tables for a Process Control Language," *Industrial Electronics and Control Instrumentation, IEEE Transactions on*, vol. IECI-15, pp. 57-61, 1968.

[20] R. Halverson, Jr., "An empirical investigation comparing IF-THEN rules and decision tables for programming rule-based expert systems," presented at System Sciences, 1993, Proceeding of the Twenty-Sixth Hawaii International Conference on, 1993.

[21] R. Siblini and N. Mansour, "Testing Web services," presented at Computer Systems and Applications, 2005. The 3rd ACS/IEEE International Conference on, 2005.

[22] W. T. Tsai, R. Paul, Y. Wang, C. Fan, and D. Wang, "Extending WSDL to facilitate Web services testing," presented at the 7th IEEE International Symposium on High Assurance Systems Engineering (HASE'02), 2002.

[23] S. R. Dalal, A. Jain, G. Patton, M. Rathi, and P. Seymour, "AETG^{SM} Web: a Web based service for automatic efficient test generation from functional requirements," presented at Industrial Strength Formal Specification Techniques, 1998 Proceedings. 2nd IEEE Workshop on, 1998.

[24] X. Bai, W. Dong, W.-T. Tsai, and Y. Chen, "WSDL-based automatic test case generation for Web services testing," presented at the 2005 IEEE International Workshop on Service-Oriented System Engineering (SOSE'05), 2005.

An Approach for Web Service Test Case Generation Based on Web Service Semantics

Siripol Noikajana and Taratip Suwannasart
Software Engineering Laboratory
Department of Computer Engineering, Faculty of Engineering
Chulalongkorn University, Bangkok 10330 Thailand
Siripol.n@student.chula.ac.th, Taratip.S@chula.ac.th

Abstract - *Software testing is an important step in software development process. Testers must create test case in order to test their software and this test case must at least contain input and expected result. Most of researches generate web service test case from WSDL (Web Service Description Language) by using black box testing technique, this test case does not consider precondition and post-condition because both precondition and post-condition have not been defined yet in WSDL standard. This paper proposes an approach to generate web service test case from SWRL (Semantic Web Rule Language) and WSDL-S (Web Service Semantics) by using random testing technique which generates test case from both precondition and post-condition, defined in web service.*

Keywords: WSDL-S, SWRL, Random testing.

1 Introduction

Web service [1] is a software which its designing aims to support interaction between mechanisms on network and able to reuse services. Web services are described by using WSDL (Web Service Description Language) [2, 3] which is an XML [4] format for describing network services as a set of end points operating on messages containing either document-oriented or procedure-oriented information. However, WSDL is extensible to allow description of endpoints and their messages regardless of what message formats or network protocols are used to communicate. But this standard [5] operates at the syntactic level and lacks the semantic expressivity needed to represent the requirements and capabilities of web services. To date, W3C member submission proposes WSDL-S (Web Service Semantics) [5, 6] that extends from WSDL standard and helps to define a mechanism to associate semantic annotations with web services so requestor meets easy web service's requirement and accesses web service correspond to web service's constraints. These semantic annotations include definition of the precondition, input, output and effect of web service operations. Both definition of the precondition and effect are able to describe by either SWRL (Semantic Web Rule Language) [7] or OCL (Object Constraint Language).

The quality of web service [8] is very important and must guarantees what web service works and what results the web service is returning. Hence, web service needs to be tested in all its operations. Most of researches, test case generation is inspired via black box testing technique based on WSDL but its information is insufficient for generating test case. Tsai et al [9] proposed to extend WSDL to support information description for generating test case. These extensions include input-output dependency, invocation sequences, hierarchical functional description and concurrent sequence specifications. By using these extensions, they help to reduce the effort and able to generate efficiently test case. Bai et al [8] proposed to generate web service test case automatically based on WSDL which this test case is generated from two perspectives. First, test data are generated by analyzing both message and data types according to XML schema syntax [10]. Finally, operation flows are generated based on the operation dependency analysis which includes input dependency, output dependency and input/output dependency. They help to reduce the number of test cases for regression testing [8, 9]. The generated test case represents XML-based document, it calls Service Test Specification (STS), which takes the basic elements from WSDL standard including operations, input, output, message and part. Then, STS examines the coverage at four levels: part coverage, message coverage, operation coverage and operation flow coverage. Bertolino et al [11] presents TAXI (Testing by Automatically generated XML Instances) tool based on the XML-based Partition Testing (XPT) approach for the automatic generation of XML instances from XML schema. The generated XML instance can be used for inter-operability testing of applications which expect in input conforming XML instances. A TAXI limitation is a large number of generated test cases. Then, Bertolino et al [11] resolves this limitation by using practical strategies for handling element weights and type values. These strategies have applying TAXI with a fixed functional coverage, applying TAXI with a fixed number of instances and applying TAXI with both.

This paper has provided background introduction about Semantic web rule language (Section 2.1), Web service semantics (Section 2.2) and Random testing technique (Section 2.3).

This paper proposes an approach to generate web service test case from SWRL and WSDL-S by using random testing technique. The method extends WSDL elements by using WSDL-S. These additional elements include precondition and effect which are defined by using XML Concrete syntax according to SWRL.

2 Background

2.1 Semantic Web Rule Language

The semantic web rule language (SWRL) [7, 12] is combination of the OWL DL and OWL Lite sublanguages of the OWL Web Ontology Language with the Unary/Binary Datalog RuleML sublanguages of the Rule Markup Language that extends the set of OWL axioms to include Horn-like rules. SWRL includes two important parts, the antecedent (body) and the consequent (head), which has such a form

$$antecedent \rightarrow consequent \qquad (1)$$

The meaning of equation (1) is the conditions specified in the antecedent hold, then the conditions specified in the consequent must also hold. Both the antecedent and the consequent consist of positive conjunctions of atoms only [12]. We can write SWRL several patterns, e.g., Abstract syntax, Human readable syntax, XML concrete syntax etc. Here we consider XML concrete syntax only.

The XML concrete syntax [7] is combination of the OWL Web Ontology Language XML Presentation Syntax, we call OWL XML Presentation Syntax [13], with the RuleML XML syntax. The root element is ruleml:imp. The main child element of ruleml:imp has ruleml:_rlab, ruleml:_body and ruleml:_head. In Figure 1, SWRL represents condition of an equilateral triangle that consists of variable x, y and z which are side of triangle.

2.2 Web Service Semantics

The Web Service Semantics (WSDL-S) [5] aims to add semantic to web service description by extending WSDL that is relatively easy to update the existing tooling around WSDL. The extensibility elements and attributes are provided:

2.2.1 modelReference

A modelReference [5] is an extension attribute which specify the association between a WSDL entity and a concept in some semantic model.

2.2.2 schemaMapping

A schemaMapping [5] is an extension attribute which is handling structural differences between the schema elements of web service and their corresponding semantic model concepts.

2.2.3 precondition and effect

Both precondition and effect [5] are new elements which are specified as child elements of the operation element. Each operation may have at most one precondition and one effect. A precondition defines a set of assertions that must be met before web service operation can be invoked. An effect defines the result of invoking an operation. In this paper, we define logic expression for precondition and effect by using modelReference to refer rule names in SWRL.

2.2.4 category

A category [5] is an extension attribute on the interface element which consists of service categorization information that could be used when publishing a service in web services registry such as UDDI.

```
<ruleml:imp rdf:ID="outputRule1">
 <ruleml:_rlab ruleml:href="#outputRule1"/>
 <ruleml:_body>
  <swrlx:builtinAtom swrlx:builtin="swrlb:equal">
   <ruleml:var>x</ruleml:var>
   <ruleml:var>y</ruleml:var>
  </swrlx:builtinAtom>
  <swrlx:builtinAtom swrlx:builtin="swrlb:equal">
   <ruleml:var>y</ruleml:var>
   <ruleml:var>z</ruleml:var>
  </swrlx:builtinAtom>
 </ruleml:_body>
 <ruleml:_head>
  <swrlx:datavaluedPropertyAtom swrlx:property="is">
   <ruleml:var>TriangleResult</ruleml:var>
   <owlx:DataValue owlx:datatype="xsd:string">
    an Equilateral Triangle
   </owlx:DataValue>
  </swrlx:datavaluedPropertyAtom>
 </ruleml:_head>
</ruleml:imp>
```

Figure 1. An outputRule1 definition via XML concrete syntax.

As in Figure 2, that shows definition of Triangle web service via WSDL-S which consists of TriangleType operation only that its result is a triangle type. This

operation has two precondition rules five effect rules which their logic expressions are represented by using SWRL.

```
<types>
  <xsd:schema elementFormDefault="qualified"
    targetNamespace="http://tempuri.org/">
    <xsd:element name="Triangle">
      <xsd:complexType>
        <xsd:sequence>
          <xsd:element name="x">
            <xsd:simpleType>
            <xsd:restriction base="xsd:int ">
              <xsd:maxExclusive value="0"/>
              <xsd:maxInclusive value="100"/>
            </xsd:restriction>
            </xsd:simpleType>
          </xsd:element>
          <xsd:element name="y"/>
            <xsd:simpleType>
            <xsd:restriction base="xsd:int ">
              <xsd:maxExclusive value="0"/>
              <xsd:maxInclusive value="100"/>
            </xsd:restriction>
            </xsd:simpleType>
          </xsd:element>
          <xsd:element name="z">
            <xsd:simpleType>
            <xsd:restriction base="xsd:int ">
              <xsd:maxExclusive value="0"/>
              <xsd:maxInclusive value="100"/>
            </xsd:restriction>
            </xsd:simpleType>
          </xsd:element>
        </xsd:sequence>
      </xsd:complexType>
    </xsd:element>
    <xsd:element name="TriangleResponse">
      <xsd:complexType>
        <xsd:sequence>
          <xsd:element name="TriangleResult"
            type="xsd:string" />
        </xsd:sequence>
      </xsd:complexType>
    </xsd:element>
  </xsd:schema>
</types>
<message name="TriangleSoapIn">
  <part name="parameters" element="tns:Triangle" />
</message>
<message name="TriangleSoapOut">
  <part name="parameters" element="tns:TriangleResponse" />
</message>
<portType>
  <operation name="TriangleType">
    <input message="tns:TriangleSoapIn" />
    <output message="tns:TriangleSoapOut" />
    <wssem:precondition name="TrianglePrecond"
      wssem:modelReference=
      "http://localhost/example.swrlx#inputRule1
      http://localhost/example.swrlx#inputRule2"/>
    <wssem:effect name="ResultEffect" wssem:modelReference=
      "http://localhost/example.swrlx#outputRule1
      http://localhost/example.swrlx#outputRule2
      http://localhost/example.swrlx#outputRule3
      http://localhost/example.swrlx#outputRule4
      http://localhost/example.swrlx#outputRule5"/>
  </operation>
</portType>
```

Figure 2. Definition of Triangle web service in WSDL-S.

2.3 Random Testing Technique

The random testing technique [14, 15] is a strategy that picks test case values from the entire input domain. For random testing technique, values of each test case are generated randomly, but very often the overall distribution of the test case has to conform to the distribution of the input domain, or an operational profile. The advantage of random testing technique is efficiency and able to avoid a form of bias in testing, e.g., boundary testing technique which always choose the min, min+, nom, max- and max+ values of a bounded variables.

Many researches proposed to generate test case by using random testing technique that has been successfully applied in many real life applications [16]. Therefore, we use random testing technique to generate test case that helps to reduce the effort and avoid complex analyses or program specifications or structures [14-16]

3 Approach

This paper contains an approach to automatically web service test case generate from WSDL-S and SWRL based on random testing. The implementation of this approach is a prototype tool called TAG-WS (Testing by Automatically Generated Web Service semantic). Figure 3 shows TAG-WS activities. TAG-WS first preprocesses data in to generate web service test case. The preprocessing step consists of SWRL analysis, WSDL-S analysis and mapping rules to operation. Next, TAG-WS picks up randomly test case values. After that, TAG-WS generates test case by using test case values.

3.1 Preprocessing

In this activity, TAG-WS requires at least two input documents, SWRL and WSDL-S. TAG-WS uses both documents in order to prepare data for generating web service test case. These data include operation name, operation's data values, web service operation's precondition and web service operation's post-condition. Both precondition and post-condition consist of logic expression by mapping rules from SWRL document to operation definition.

3.1.1 SWRL analysis

A SWRL [5] document defines rules which are referred by modelReference element in WSDL-S document. The rule definition consists of two parts, antecedent and consequent, each of which consists of either a set of atoms or an empty set of atom. A set of atoms can be extracted to a logic expression for defining web service operation's precondition and web service operation's post-condition. This paper uses SWRL built-ins [5] to define interoperation for logic expression. e.g., swrlb:greaterThan is used to

compare whether first argument's value greater than second argument's value.

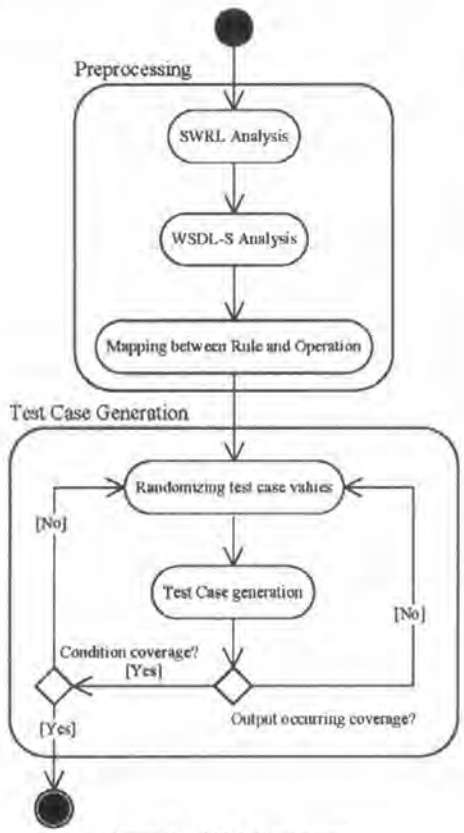


Figure 3. TAG-WS activities.

The basic algorithm of SWRL analysis is as follows:

1. Find ruleml:imp element node.
2. Retrieve rule name from rdf:ID attribute.
3. Analyze the hierarchical tree structure of ruleml:imp element node.
4. Traverse the tree, and at each tree node:
   a. If it is a ruleml:_body element node, generate a logic expression of antecedent part.
   b. If it is a ruleml:_head element node, generate a logic expression of consequent part.
5. Keep rule name and logic expression of antecedent part and consequent part into database.
6. Find next ruleml:imp element node and go to step 2. If next ruleml:imp element node is not exist, exit analysis.

In step 4.a and 4.b, the process is generated logic expression of antecedent part and consequent part by analyzing a set of atoms and interpreting operator according SWRL built-ins. After that, it normalizes logic expression to generic logic expression.

3.1.2 WSDL-S analysis

A WSDL-S [5] document describes web service's specification which is a semantic annotation. Data type definition for each operation can be defined based on XML Schema document. This paper discusses how to extract operation definition from WSDL-S.

The basic algorithm of WSDL-S analysis is as follows:

1. Analyze the hierarchical tree structure of operation element node.
2. Traverse the tree, and at each tree node:
   a. If it is an input element node, generate data type definition for input parameter.
   b. If it is an output element node, generate data type definition for output parameter.
   c. If it is a precondition element node, extract rule's URIs of precondition from modelReference attribute.
   d. If it is an effect element node, extract referred rule's URIs of post-condition from modelReference attribute.
3. Keep operation definition into database.

In step 2.a and 2.b, the process is generated data type definition which consequence from binding message type to operation. Data type definition consists of parameter name, data type, bound value and number of data value.

3.1.3 Mapping rules to operation

This activity retrieves information of two previous activities from database. After that, all rules are mapped to each operation definition.

The basic algorithm of mapping rules to operation is as follows:

1. Retrieve information that consists of SWRL analysis and WSDL-S analysis.
2. Analyze the rule's URIs list of precondition.
3. Traverse the list, and at each item by mapping rules from SWRL analysis to item.
4. Analyze the rule's URIs list of post-condition and repeat step 3.

3.2 Test Case Generation

In this activity, consists of randomizing test case values and generating test case. As in Figure 4, represents the generated web service test case definition according XML Schema, called Web service Test Case Specification

(WTCS). WTCS can be easily represented several formats upon application.

```
<schema xmlns="http://www.w3.org/2001/XMLSchema">
 <element name="items">
  <complexType>
   <sequence>
    <element name="item" type="item" minOccurs="0"
     maxOccurs="unbounded"/>
   </sequence>
  </complexType>
 </element>
 <complexType name="item">
  <sequence>
   <element name="input" type="dataType" minOccurs="0"
    maxOccurs="unbounded" />
   <element name="expected" type="string" minOccurs="0"
    maxOccurs="unbounded" />
  </sequence>
  <attribute name="operation" type="string" />
 </complexType>
 <complexType name="dataType">
  <sequence>
   <element name="variable" type="string" />
   <element name="value" type="string" />
  </sequence>
 </complexType>
</schema>
```

Figure 4. Web service Test Case Specification.

The basic algorithm of test case generation is as follows:

1. Analyze operation definition.
2. Pick up randomly test data values according operation definition.
3. Analyze expected outputs.
4. Generate web service test case according WTCS.
5. Repeat step 2 if
   a. All web service test cases are not output occurring coverage.
   b. All web service test cases are not 100% condition coverage.

4 Results

We implement TAG-WS the proposed approach by C#. TAG-WS examines to generate web service test case. The web service case study serves to examine type of triangle. Figure 2 shows definition of Triangle web service via WSDL-S standard that consists of three integer variables, x, y and z.

TAG-WS generates web service test case of case study as follows.

4.1 SWRL analysis

TAG-WS starts reading an input SWRL document. Next, TAG-WS extracts rule from an input SWRL. After that, TAG-WS analyzes each rule by its normalizing logic expression to generic logic expression and keeping it into database.

As in Figure 1, shows rule definition of outputRule1 that TAG-WS extracts logic expression as in

$$(x = y) and (y = z) \rightarrow \text{TriangleResult is an Equilateral Triangle} \quad (2)$$

Figure 5 shows rule definition of inputRule2 that TAG-WS extracts logic expression as in

$$\begin{aligned} &(sumOfY\_Z = y + z) and (x < sumOfY\_Z) \\ &(sumOfX\_Z = x + z) and (y < sumOfX\_Z) \\ &(sumOfX\_Y = x + y) and (z < sumOfX\_Y) \\ &\rightarrow \text{TriangleResult is a Triangle} \end{aligned} \quad (3)$$

Anyway, equation (3) is not generic equation, then, TAG-WS normalizes equation (3) as in

$$(x < y + z) and (y < x + z) and (z < x + y) \rightarrow \text{TriangleResult is a Triangle} \quad (4)$$

TAG-WS analyzes an input SWRL document of case study; results of SWRL analysis are as Table 1.

Table 1. Definition of logic expression for each rule.

Rule	*Antecedent*	*Consequent*
inputRule1	(x > 0) and (y > 0) and (z > 0)	
inputRule2	(x < (y + z)) and (y < (x + z)) and (z < (x + y))	TriangleResult is a Triangle
outputRule1	(x == y) and (y == z)	TriangleResult is an Equilateral Triangle
outputRule2	(x <> y) and (x <> z) and (y <> z)	TriangleResult is a Scalene Triangle
outputRule3	(x == y) and (x <> z)	TriangleResult is an Isosceles Triangle
outputRule4	(x == z) and (x <> y)	TriangleResult is an Isosceles Triangle
outputRule5	(y == z) and (x <> z)	TriangleResult is an Isosceles Triangle

```
<ruleml:imp rdf:ID="inputRule2">
 <ruleml:_rlab ruleml:href="#inputRule2"/>
 <ruleml:_body>
  <swrlx:builtinAtom swrlx:builtin="swrlb:add">
   <ruleml:var>sumOfY_Z</ruleml:var>
   <ruleml:var>y</ruleml:var>
   <ruleml:var>z</ruleml:var>
  </swrlx:builtinAtom>
  <swrlx:builtinAtom swrlx:builtin="swrlb:add">
   <ruleml:var>sumOfX_Z</ruleml:var>
   <ruleml:var>x</ruleml:var>
   <ruleml:var>z</ruleml:var>
  </swrlx:builtinAtom>
  <swrlx:builtinAtom swrlx:builtin="swrlb:add">
   <ruleml:var>sumOfX_Y</ruleml:var>
   <ruleml:var>x</ruleml:var>
   <ruleml:var>y</ruleml:var>
  </swrlx:builtinAtom>
  <swrlx:builtinAtom swrlx:builtin="swrlb:lessThan">
   <ruleml:var>x</ruleml:var>
   <ruleml:var>sumOfY_Z</ruleml:var>
  </swrlx:builtinAtom>
  <swrlx:builtinAtom swrlx:builtin="swrlb:lessThan">
   <ruleml:var>y</ruleml:var>
   <ruleml:var>sumOfX_Z</ruleml:var>
  </swrlx:builtinAtom>
  <swrlx:builtinAtom swrlx:builtin="swrlb:lessThan">
   <ruleml:var>z</ruleml:var>
   <ruleml:var>sumOfX_Y</ruleml:var>
  </swrlx:builtinAtom>
 </ruleml:_body>
 <ruleml:_head>
  <swrlx:datavaluedPropertyAtom swrlx:property="is">
   <ruleml:var>TriangleResult</ruleml:var>
   <owlx:DataValue owlx:datatype="xsd:string">
    a Triangle
   </owlx:DataValue>
  </swrlx:datavaluedPropertyAtom>
 </ruleml:_head>
</ruleml:imp>
```

Figure 5. An inputRule2 definition.

4.2 WSDL-S analysis

TAG-WS starts reading an input WSDL-S document of case study and analyzing an input WSDL-S document that result of WSDL-S analysis as Table 2.

4.3 Mapping rules to operation

After TAG-WS analyzes finished WSDL-S and SWRL that, TAG-WS maps rules to operation definition. For case study, TriangleType operation includes two precondition rules and five post-condition rules. TAG-WS retrieves rule definition according rule of precondition and post-condition then, TAG-WS maps rules to operation.

4.4 Test Case Generation

TAG-WS picks up test data values in order to generate web service test case. The web service test cases are 100% condition coverage (specified in WSDL-S) and each occurred output at least consists in any web service test case. Table 3 shows results of generated web service test case for Triangle web service and Table 4 shows example of WTCS.

Table 2. Result of WSDL-S analysis for TriangleType operation definition

Operation	TriangleType	
Input data	*Variable*	*Data type*
	X	Integer
	Y	Integer
	Z	Integer
Output data	*Variable*	*Data type*
	TriangleResult	string
Precondition	*Rule*	*URI*
	inputRule1	http://localhost/example.swrlx#inputRule1
	inputRule2	http://localhost/example.swrlx#inputRule2
Post-condition	*Rule*	*URI*
	outputRule1	http://localhost/example.swrlx#outputRule1
	outputRule2	http://localhost/example.swrlx#outputRule2
	outputRule3	http://localhost/example.swrlx#outputRule3
	outputRule4	http://localhost/example.swrlx#outputRule4
	outputRule5	http://localhost/example.swrlx#outputRule5

Table 3. Web service test case for Triangle web service.

Test Case	*Non-triangle*	*Equilateral*	*Scalene*	*Isosceles*
11509	8082	1	7113	313
7706	4019	1	3528	158
11499	5950	1	5302	246
6217	3248	2	2837	130
11015	5802	1	4971	241

5 Conclusion

This paper has presented web service test case generation based on web service semantics. The generated test case meets web service's requirement and web service description because this test case is the consequence of analyzing also precondition and post-condition. The web service test case generation based on random testing technique which results in many test cases. The web service

test cases are 100% condition coverage and output occurring coverage.

For future work, it is planned to reduce amount of web service test case and improve test case generation by using other testing techniques, such as boundary testing, equivalence class testing, etc.

Table 4. Web service Test Case Specification for TriangleType operation.

Test Case No	1	
Operation	TriangleType	
Test Data	*Variable*	*Value*
	X	20
	y	10
	z	20
Expected Result	Invoke web service TriangleResult is a Triangle TriangleResult is an Isosceles Triangle	

6 References

[1] W3C Working Group, "Web Services Glossary," 2004.

[2] W3C, "Web Services Description Language (WSDL) 1.1," 2001.

[3] W3Schools, "WSDL Tutorial."

[4] W3C Recommendation, "Extensible Markup Language (XML) 1.0," 2004.

[5] W3C Member Submission, "Web Service Semantics - WSDL-S," 2005.

[6] J. Miller, K. Verma, P. Rajasekaran, A. Sheth, R. Aggarwal, and K. Sivashanmugam, "WSDL-S: Adding Semantics to WSDL - White Paper."

[7] W3C Member Submission, "SWRL: A Semantic Web Rule Language Combining OWL and RuleML."

[8] X. Bai, W. Dong, W.-T. Tsai, and Y. Chen, "WSDL-based automatic test case generation for Web services testing," presented at the 2005 IEEE International Workshop on Service-Oriented System Engineering (SOSE'05), 2005.

[9] W. T. Tsai, R. Paul, Y. Wang, C. Fan, and D. Wang, "Extending WSDL to facilitate Web services testing," presented at the 7th IEEE International Symposium on High Assurance Systems Engineering (HASE'02), 2002.

[10] W3Schools, "XML Schema Tutorial."

[11] A. Bertolino, J. Gao, E. Marchetti, and A. Polini, "Automatic Test Data Generation for XML Schema-based Partition Testing," presented at Second International Workshop on Automation of Software Test (AST'07), 2007.

[12] Protege, "SWRL Language FAQ," 2007.

[13] M. Hori, J. Euzenat, and P. F. Patel-Schneider, "OWL Web Ontology Language XML Presentation Syntax."

[14] P. Jorgensen, Software Testing A Craftsman's Approach 2ed: CRC PRESS.

[15] J. Z. Gao, H.-S. J. Tsao, and Y. Wu, Testing and Quality Assurance for Component-Based Software: Artech House computing library, 2003.

[16] T. Y. Chen, H. De Hao, T. H. Tse, and Z. Yang, "An Innovative Approach to Tackling the Boundary Effect in Adaptive Random Testing," presented at System Sciences, 2007. HICSS 2007. 40th Annual Hawaii International Conference on, 2007.

[17] W3C Recommendation, "XML Schema Part 2: Datatypes."

7 Acknowledgement

This research is part of the Engineering New Paradigm Software for Enterprises with Service-Oriented Architecture Project, supported by Thailand's Software Industry Promotion Agency (Public Organization).

The authors are grateful to Dr. Mala Supongpan for her kindly making through useful comments and also extend their gratefuls to the Royal Bangkok Sports Club (RBSC) scholarship for partly funded for the project. Thanks to Miss Parat Supongpan and Dr. Xiaoying Bai. Finally, the authors are appreciated to Mr. Pedro Ramirez Suárez for his English edited.

ประวัติผู้เขียนวิทยานิพนธ์

นายศิริพล น้อยกาญจนะ เกิดวันที่ 12 มกราคม พ.ศ. 2527 สำเร็จการศึกษาระดับมัธยมศึกษาตอนปลายจากโรงเรียนมหิดลวิทยานุสรณ์ จังหวัดนครปฐม เมื่อปีการศึกษา 2544 และสำเร็จการศึกษาในหลักสูตรวิทยาศาสตรบัณฑิต สาขาวิชาเอกคณิศศาสตร์ สาขาวิชาโทคณิตศาสตร์ประกันภัยและคอมพิวเตอร์ ภาควิชาคณิตศาสตร์และสถิติ คณะวิทยาศาสตร์และเทคโนโลยี มหาวิทยาลัยธรรมศาสตร์ จังหวัดปทุมธานี เมื่อปีการศึกษา 2548 และเข้าศึกษาต่อในหลักสูตรวิทยาศาสตรมหาบัณฑิต สาขาวิทยาศาสตร์คอมพิวเตอร์ ภาควิชาวิศวกรรมคอมพิวเตอร์ คณะวิศวกรรมศาสตร์ จุฬาลงกรณ์มหาวิทยาลัย ในปีการศึกษา 2549 ที่อยู่ปัจจุบันที่สามารถติดต่อได้คือ 212/3 ซอยอำนวยสุข 2 ถนนเพชรเกษม 16 แขวงท่าพระ เขตบางกอกใหญ่ กรุงเทพมหานคร 10600 หมายเลขโทรศัพท์ 086-662-1250 อีเมล์ siripol.n@gmail.com