LAMPIRAN A

Isi *file* “MorphoService.java”

```
package org.me.morpho;

import java.util.Collection;
import java.util.ArrayList;
import javax.jws.WebMethod;
import javax.jws.WebParam;
import javax.jws.WebService;
import java.io.*;

/**
 *
 * @author Charles Christian 2008
 */
@WebService()
public class MorphoWS
{
    static ProcessBuilder pb;
    static Process p;

    /**
     * Web service operation
     */
    @WebMethod(operationName = "applyUp")
    public String applyUp(@WebParam(name = "arg0") String str)
    {
      return apply("up", false, str);
    }

    /**
     * Web service operation
     */
    @WebMethod(operationName = "applyDown")
    public String applyDown(@WebParam(name = "arg0") String str)
    {
      return apply("down", false, str);
    }

    /**
     * Web service operation
     */
    @WebMethod(operationName = "applyUpAll")
    public String applyUpAll(@WebParam(name = "arg0") String str)
    {
        return apply("up", true, str);
    }

    /**
```

```
     * Web service operation
     */
    @WebMethod(operationName = "applyDownAll")
    public String applyDownAll(@WebParam(name = "arg0") String str)
    {
        return apply("down", true, str);
    }

    public String apply( String command, boolean all, String input )
    {
      String result = "";

      try
        {
            pb = new ProcessBuilder( "cmd.exe", "/K", "D:" );
            p = pb.start();

            OutputStream os = p.getOutputStream();
            OutputStreamWriter osw = new OutputStreamWriter( os );
            BufferedWriter bw = new BufferedWriter( osw );

            InputStream is = p.getInputStream();
            InputStreamReader isr = new InputStreamReader( is );
            BufferedReader br = new BufferedReader( isr );

            bw.write( "cd xerox-windows\n" );
            bw.write( "xfst.exe\n" );
            bw.write( "load stack myfile.fst\n" );

            bw.write( "apply " + command + " " + input + "\n" );

            bw.close();

            br.skip(95);

            String temp;

            if(all)
                while( !(temp = br.readLine()).equals("bye.") )
                    result += temp + " ";
            else
                if( !(temp = br.readLine()).equals("bye.") )
                    result += temp + " ";

            br.close();
      }
        catch( Exception e )
        {
            result += "error - " + e;
            e.printStackTrace();
        }

      return result;
    }
```

```
/**
 * Web service operation
 */
@WebMethod(operationName = "doAnalyze")
public Collection<Morpheme> operation(@WebParam(name =
  "language")Language language, @WebParam(name = "text")
  String text)
{
    Collection<Morpheme> res = new ArrayList<Morpheme>();

    String word, lemma, temp;
    PartOfSpeech pos;

    String[] token = text.split( " " );

    for( int i = 0; i < token.length; i++ )
    {
        word = token[i];

        temp = apply("up", false, word);

        if(!temp.equals(""))
        {
            String temp2 = temp.replace("+", " ");

            String[] analyse = temp2.split( " " );
            lemma = analyse[0];

            if( analyse[1].contains( "Noun" ) )
            {
                pos = PartOfSpeech.noun;
            }
            else if( analyse[1].contains( "Verb" ) )
            {
                pos = PartOfSpeech.verb;
            }
            else if( analyse[1].contains( "Adjective" ) )
            {
                pos = PartOfSpeech.adjective;
            }
            else if( analyse[1].contains( "Etc" ) )
            {
                pos = PartOfSpeech.other;
            }
            else pos = PartOfSpeech.unknown;
        }
        else
        {
            lemma = "";
            pos = PartOfSpeech.unknown;
        }

        res.add( new Morpheme(word, lemma, pos) );
        }

        res.add( new Morpheme() );

        return res;
    }
}
```

Isi *file* “Language.java”

```
package org.me.morpho;

public class Language
{
    public Language()
    {

    }
    public Language(String code)
    {
        this.code = code;
    }

    public String getCode()
    {
        return code;
    }

    public void setCode( String code )
    {
        this.code = code;
    }

    private String code;
}
```

Isi *file* “Morpheme.java”

```
package org.me.morpho;

public class Morpheme
{
    public Morpheme()
    {

    }
    public Morpheme(String word, String lemma, PartOfSpeech
                    partOfSpeech)
    {
            this.word = word;
            this.lemma = lemma;
            this.partOfSpeech = partOfSpeech;
    }

    public String getWord()
    {
        return word;
    }

    public void setWord( String word )
    {
        this.word = word;
    }

    public String getLemma()
    {
        return lemma;
    }

    public void setLemma( String lemma )
    {
        this.lemma = lemma;
    }

    public PartOfSpeech getPartOfSpeech()
    {
        return partOfSpeech;
    }

    public void setPartOfSpeech( PartOfSpeech partOfSpeech )
    {
        this.partOfSpeech = partOfSpeech;
    }

    private String word;
    private String lemma;
    private PartOfSpeech partOfSpeech;
}
```

Isi *file* “PartOfSpeech.java”

```
package org.me.morpho;

/*
 * $Id:PartOfSpeech.java 3343 2006-10-13 14:06:48 +0900 nakaguchi $
 *
 * This is a program for Language Grid Core Node. This combines
 * multiple language resources and provides composite language
 * services.
 * Copyright (C) 2005-2008 NICT Language Grid Project.
 *
 * This program is free software: you can redistribute it and/or
 * modify it under the terms of the GNU Lesser General Public
 * License as published by the Free Software Foundation, either
 * version 2.1 of the License, or (at your option) any later \
 * version.
 *
 * This program is distributed in the hope that it will be useful,
 * but WITHOUT ANY WARRANTY; without even the implied warranty of
 * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU
 * Lesser General Public License for more details.
 *
 * You should have received a copy of the GNU Lesser General Public
 * License along with this program. If not, see
 * <http://www.gnu.org/licenses/>.
 */

import java.util.Collection;
import java.util.Collections;
import java.util.LinkedHashMap;
import java.util.Map;

/**
 * @author $Author:nakaguchi $
 * @version $Revision:3343 $
 */
public enum PartOfSpeech {
  noun_common("noun.common"),
  noun_proper("noun.proper"),
  noun_other("noun.other"),
  noun,
  verb,
  adjective,
  adverb,
  other,
  unknown,
  ;

  PartOfSpeech(){
    this.expression = null;
  }


  PartOfSpeech(String expression){
    this.expression = expression;
  }

  @Override
  public String toString(){
    return getExpression();
  }

  public String getExpression(){
    if(expression != null){
      return expression;
    } else{
      return name();
    }
  }
  public static PartOfSpeech valueOfExpression(String expression){
    return expressionToValue.get(expression);
  }


  public static Collection<PartOfSpeech> valuesCollection(){
    return values;
  }

  private final String expression;

  private static final Map<String, PartOfSpeech> expressionToValue;
  private static final Collection<PartOfSpeech> values;

  static{
    expressionToValue = new LinkedHashMap<String, PartOfSpeech>();
    for(PartOfSpeech i : values()){
      expressionToValue.put(i.getExpression(), i);
    }
    values = Collections.unmodifiableCollection(
      expressionToValue.values()
    );
  }
}
```

LAMPIRAN B

Isi *file* “MorphoWSService.wsdl”

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<!-- Generated by JAX-WS RI at http://jax-ws.dev.java.net. RI's
version is JAX-WS RI 2.1.2-b05-RC1. -->
<definitions targetNamespace="http://morpho.me.org/"
name="MorphoWSService" xmlns="http://schemas.xmlsoap.org/wsdl/"
xmlns:tns="http://morpho.me.org/"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/">

  <types>
    <xsd:schema>
      <xsd:import namespace="http://morpho.me.org/"
        schemaLocation="MorphoWSService_schema1.xsd"/>
    </xsd:schema>
  </types>

  <message name="applyUp">
    <part name="parameters" element="tns:applyUp"/>
  </message>
  <message name="applyUpResponse">
    <part name="parameters" element="tns:applyUpResponse"/>
  </message>
  <message name="applyDown">
    <part name="parameters" element="tns:applyDown"/>
  </message>
  <message name="applyDownResponse">
    <part name="parameters" element="tns:applyDownResponse"/>
  </message>
  <message name="applyUpAll">
    <part name="parameters" element="tns:applyUpAll"/>
  </message>
  <message name="applyUpAllResponse">
    <part name="parameters" element="tns:applyUpAllResponse"/>
  </message>
  <message name="applyDownAll">
    <part name="parameters" element="tns:applyDownAll"/>
  </message>
  <message name="applyDownAllResponse">
    <part name="parameters" element="tns:applyDownAllResponse"/>
  </message>
  <message name="doAnalyze">
    <part name="parameters" element="tns:doAnalyze"/>
  </message>
  <message name="doAnalyzeResponse">
    <part name="parameters" element="tns:doAnalyzeResponse"/>
  </message>
  <portType name="MorphoWS">
    <operation name="applyUp">
      <input message="tns:applyUp"/>
      <output message="tns:applyUpResponse"/>
    </operation>
    <operation name="applyDown">
      <input message="tns:applyDown"/>
      <output message="tns:applyDownResponse"/>
    </operation>
    <operation name="applyUpAll">
      <input message="tns:applyUpAll"/>
      <output message="tns:applyUpAllResponse"/>
    </operation>
    <operation name="applyDownAll">
      <input message="tns:applyDownAll"/>
      <output message="tns:applyDownAllResponse"/>
    </operation>
    <operation name="doAnalyze">
      <input message="tns:doAnalyze"/>
      <output message="tns:doAnalyzeResponse"/>
    </operation>
  </portType>
  <binding name="MorphoWSPortBinding" type="tns:MorphoWS">
    <soap:binding transport="http://schemas.xmlsoap.org/soap/http"
style="document"/>
    <operation name="applyUp">
      <soap:operation soapAction=""/>
      <input>
        <soap:body use="literal"/>
      </input>
      <output>
        <soap:body use="literal"/>
      </output>
    </operation>
    <operation name="applyDown">
      <soap:operation soapAction=""/>
      <input>
        <soap:body use="literal"/>
      </input>
      <output>
        <soap:body use="literal"/>
      </output>
    </operation>
    <operation name="applyUpAll">
      <soap:operation soapAction=""/>
      <input>
        <soap:body use="literal"/>
      </input>
      <output>
        <soap:body use="literal"/>
      </output>
    </operation>
    <operation name="applyDownAll">
      <soap:operation soapAction=""/>
      <input>
        <soap:body use="literal"/>
      </input>
      <output>
        <soap:body use="literal"/>
      </output>
    </operation>
    <operation name="doAnalyze">
      <soap:operation soapAction=""/>
      <input>
        <soap:body use="literal"/>
      </input>
      <output>
        <soap:body use="literal"/>
      </output>
    </operation>
  </binding>
  <service name="MorphoWSService">
    <port name="MorphoWSPort" binding="tns:MorphoWSPortBinding">
      <soap:address location="http://localhost:8080/
                              MorphoWSService/MorphoWS"/>
    </port>
  </service>
</definitions>
```

LAMPIRAN C

Isi *file* “MorphoWSService_schema1.xsd”

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<xs:schema version="1.0" targetNamespace="http://morpho.me.org/"
xmlns:tns="http://morpho.me.org/"
xmlns:xs="http://www.w3.org/2001/XMLSchema">

  <xs:element name="applyDown" type="tns:applyDown"/>

  <xs:element name="applyDownAll" type="tns:applyDownAll"/>

  <xs:element name="applyDownAllResponse"
type="tns:applyDownAllResponse"/>

  <xs:element name="applyDownResponse"
type="tns:applyDownResponse"/>

  <xs:element name="applyUp" type="tns:applyUp"/>

  <xs:element name="applyUpAll" type="tns:applyUpAll"/>

  <xs:element name="applyUpAllResponse"
type="tns:applyUpAllResponse"/>

  <xs:element name="applyUpResponse" type="tns:applyUpResponse"/>

  <xs:element name="doAnalyze" type="tns:doAnalyze"/>

  <xs:element name="doAnalyzeResponse"
type="tns:doAnalyzeResponse"/>

  <xs:complexType name="doAnalyze">
    <xs:sequence>
      <xs:element name="language" type="tns:language"
minOccurs="0"/>
      <xs:element name="text" type="xs:string" minOccurs="0"/>
    </xs:sequence>
  </xs:complexType>

  <xs:complexType name="language">
    <xs:sequence>
      <xs:element name="code" type="xs:string" minOccurs="0"/>
    </xs:sequence>
  </xs:complexType>

  <xs:complexType name="doAnalyzeResponse">
    <xs:sequence>
      <xs:element name="return" type="tns:morpheme" minOccurs="0"
maxOccurs="unbounded"/>
    </xs:sequence>
  </xs:complexType>
  <xs:complexType name="morpheme">
    <xs:sequence>
      <xs:element name="lemma" type="xs:string" minOccurs="0"/>
      <xs:element name="partOfSpeech" type="tns:partOfSpeech"
minOccurs="0"/>
      <xs:element name="word" type="xs:string" minOccurs="0"/>
    </xs:sequence>
  </xs:complexType>

  <xs:complexType name="applyDownAll">
    <xs:sequence>
      <xs:element name="arg0" type="xs:string" minOccurs="0"/>
    </xs:sequence>
  </xs:complexType>

  <xs:complexType name="applyDownAllResponse">
    <xs:sequence>
      <xs:element name="return" type="xs:string" minOccurs="0"/>
    </xs:sequence>
  </xs:complexType>

  <xs:complexType name="applyUp">
    <xs:sequence>
      <xs:element name="arg0" type="xs:string" minOccurs="0"/>
    </xs:sequence>
  </xs:complexType>

  <xs:complexType name="applyUpResponse">
    <xs:sequence>
      <xs:element name="return" type="xs:string" minOccurs="0"/>
    </xs:sequence>
  </xs:complexType>

  <xs:complexType name="applyUpAll">
    <xs:sequence>
      <xs:element name="arg0" type="xs:string" minOccurs="0"/>
    </xs:sequence>
  </xs:complexType>

  <xs:complexType name="applyUpAllResponse">
    <xs:sequence>
      <xs:element name="return" type="xs:string" minOccurs="0"/>
    </xs:sequence>
  </xs:complexType>

  <xs:complexType name="applyDown">
    <xs:sequence>
      <xs:element name="arg0" type="xs:string" minOccurs="0"/>
    </xs:sequence>
  </xs:complexType>

  <xs:complexType name="applyDownResponse">
    <xs:sequence>
```

```
      <xs:element name="return" type="xs:string" minOccurs="0"/>
    </xs:sequence>
  </xs:complexType>

  <xs:simpleType name="partOfSpeech">
    <xs:restriction base="xs:string">
      <xs:enumeration value="noun_common"/>
      <xs:enumeration value="noun_proper"/>
      <xs:enumeration value="noun_other"/>
      <xs:enumeration value="noun"/>
      <xs:enumeration value="verb"/>
      <xs:enumeration value="adjective"/>
      <xs:enumeration value="adverb"/>
      <xs:enumeration value="other"/>
      <xs:enumeration value="unknown"/>
    </xs:restriction>
  </xs:simpleType>
</xs:schema>
```

LAMPIRAN D

Isi *file* “client.php”

```
<html>

<head>
  <title>Morphological Analysis Service - Bahasa Indonesia</title>
  <script src="prototype.js" type=text/javascript></script>
  <script language="javascript" type=text/javascript>
  function process()
  {
    operationCall = document.getElementById("operation").value;
    inputTemp =
      document.getElementById("input").value.toLowerCase();
    inputString = "";

    i = 0;

    while( inputTemp.charAt(i) != "" )
    {
      if( (inputTemp.charCodeAt( i ) >= 97 && inputTemp.charCodeAt(
           i ) <= 123) || inputTemp.charCodeAt( i ) == 45 )
      {
        inputString += inputTemp.charAt( i );
      }
      else
      {
        inputString += " ";
      }

      i++;
    }

    wait();

    new Ajax.Request(
      "AjaxClient.php",
      {
        method: "get",
        parameters: {operationCall:operationCall,
                     inputString:inputString},
        evalJSON: true,
        onSuccess: showResult
      }
    );
  }

  function showResult(result)
  {
    var str = "";
    var myresult = result.responseJSON;
    str += "<h3>Results:</h3> ";

    if( operationCall == "doAnalyze" )
    {
      if( myresult.count <= 0 ) str += "<div style='font-" +
        "weight:bold; color:red'>No result matched...</div>";
      else
      {
        str += "<table border=1><tr><th>No</th>“<th>Word</th>" +
          "<th>Original</th><th>Part of Speech</th></tr>";

        for( i = 0; i < myresult.count; i++ )
        {
          str += "<tr><td>" + (i+1) + "</td>";
          str += "<td>" + myresult.result[i].word + "</td>";
          str += "<td>" + myresult.result[i].lemma + "</td>";
          str += "<td>" + myresult.result[i].partOfSpeech +
                 "</td></tr>";
        }

        str += "</table>";
      }

    }
    else
    {
      var str3 = myresult.result;
      var str2 = str3.split(" ");

      if( str2[0] == "" ) str += "<div style='font-weight:bold; "+
          "color:red'>No result matched...</div>";
      else
      {
        for(var i = 0; str2[i] != null; i++)
        {
          str += str2[i] + "<br />"
        }
      }
    }

    str += "<h3>Time:</h3> ";

    str += myresult.time + " sec";

    document.getElementById("Result").innerHTML = str;
  }

  function wait()
  {
    str = "<h3>Results:</h3>";
    str += "<strong><img border='0' src='images/wait.gif'" +
           "width='24' height='20'>&nbsp;Processing...</strong>";
    document.getElementById("Result").innerHTML = str;
  }
```

```
</script>
</head>

<body style="background-color:lightyellow; font-family:lucida sans
unicode">
  <h1>Morphological Analysis Service (Bahasa Indonesia)</h1>

  <div>
  <h3>Operations:</h3>
  <select id="operation">
    <option value="applyUp" id="applyUp">Apply Up</option>
    <option value="applyDown" id="applyDown">Apply Down</option>
    <option value="applyUpAll" id="applyUpAll">Apply Up All</option>
    <option value="applyDownAll" id="applyDownAll">Apply Down
      All</option>
    <option value="doAnalyze" id="doAnalyze">Do Analyze</option>
  </select>
  &nbsp;
  <input type="text" id="input" />
  &nbsp;
  <input type="button" id="process" value="Process"
    onClick="process()"/>
  </div>
  <div id="Result" style="position:relative; left:5px;"></div>
</body>

</html>
```

Isi *file* “AjaxClient.php”

```
<?php

header("Content-Type: application/json");

require_once('libs/nusoap.php');

$from = microtime(true);

$operationCall = $_GET['operationCall'];
$inputString = $_GET['inputString'];
$count = 0;

$wsdl = "http://localhost:8080/MorphoWSApplication/MorphoWS?wsdl";
$ns = "http://morpho.me.org/";

$client=new soapclientw($wsdl);

$param=array('arg0'=>$inputString);

if($operationCall == "applyUp")
{
      $outputString = $client->call('applyUp', $param, $ns );
}
else if($operationCall == "applyDown")
{
      $outputString = $client->call('applyDown', $param, $ns );
}
else if($operationCall == "applyUpAll")
{
      $outputString = $client->call('applyUpAll', $param, $ns );
}
else if($operationCall == "applyDownAll")
{
      $outputString = $client->call('applyDownAll', $param, $ns );
}
else if($operationCall == "doAnalyze")
{
      $param = array('language'=>array('code'=>'id'),
                     'text'=>$inputString);
      $outputArray = $client->call('doAnalyze', $param, $ns );
      $outputString = "";

      for($i = 0; $i < sizeof($outputArray)-1; $i++ )
      {
            $temp = $outputArray[$i];

            if( $temp['word'] != "" )
            {
                  $outputString[] = array('word'=>$temp['word'],
                    'lemma'=>$temp['lemma'],
                    'partOfSpeech'=>$temp['partOfSpeech']);
                  $count++;
            }

      }
}

$to = microtime(true);

$result = array("result"=> $outputString, "time" => $to - $from,
"count" => $count);
echo JSON_encode($result);

function microtime_float()
{
   list($usec, $sec) = explode(" ", microtime());
   return ((float)$usec + (float)$sec);
}

?>
```