```
package other;
import java.io.File;
import java.io.FileNotFoundException;
import java.io.FileOutputStream;
import java.io.FileWriter;
import java.io.IOException;
import java.io.OutputStream;
import java.io.OutputStreamWriter;
import java.io.Writer;
import java.util.ArrayList;
import java.util.HashMap;
import java.util.Iterator;
import java.util.Map;
import java.util.Map.Entry;
import java.util.Set;
import org.dom4j.Attribute;
import org.dom4j.Document;
import org.dom4j.Element;
import org.dom4j.io.OutputFormat;
import org.dom4j.io.SAXReader;
import org.dom4j.io.XMLWriter;
import com.test.bpelbean.Message;
import com.test.bpelbean.Value;
import com.test.bean.Activity;
import com.test.bean.Atom;
import com.test.bean.Structure;
public class ParseBpel {
	public static Structure activity=new Structure("begin");
	public static ArrayList<String> If=new ArrayList<String>();
	public static HashMap<String,String> If_=new HashMap<String,String>();
	public static HashMap<String,String> If_Branch=new HashMap<String,String>();
	public static ArrayList<String> collection=new ArrayList<String>();
	public static void main(String[] args) throws FileNotFoundException, IOException{
		String filename="D:\\Workspace1\\DebugTest\\bpel\\TravelAgency.bpel";
		File file = new File(filename);
		Document document = openXMLFile(file.getAbsolutePath());
		Element root = document.getRootElement();
		for (Iterator iter = root.elementIterator(); iter.hasNext();){
			Element element = (Element) iter.next();
			String attr=element.attributeValue("name");
			System.out.println(attr);
			for(Iterator iter1 = element.elementIterator(); iter1.hasNext();){
				Element element1 = (Element) iter1.next();
				String attr1=element1.attributeValue("name");
				System.out.println("        "+attr1);
			}
		  System.out.println(attr1);
		}
	}
	public static Document openXMLFile(String filePath) {
        Document document = null;
        SAXReader reader = new SAXReader();
        try {
            File file = new File(filePath);
            document = reader.read(file);
        } catch (Exception e) {
            e.printStackTrace();
        }
        return document;
    }
    public static boolean readwsdlfile(String filepath) throws FileNotFoundException,
    IOException {
        try {
            File file = new File(filepath);
            if (!file.isDirectory()) {
                System.out.println("absolutepath=" + file.getAbsolutePath());
                System.out.println("name=" + file.getName());
                String absolutepath = file.getAbsolutePath();
                Document document = openXMLFile(absolutepath);
                parseWsdl(document);
            } else if (file.isDirectory()) {
                String[] filelist = file.list();
                for (int i = 0; i < filelist.length; i++) {
                    File readfile = new File(filepath + "\\" + filelist[i]);
                    if (!readfile.isDirectory()) {
                        if (filelist[i].contains(".wsdl")) {
                            String absolutepath = readfile.getAbsolutePath();
                            Document document = openXMLFile(absolutepath);
                            parseWsdl(document);
                        }
                    } else if (readfile.isDirectory()) {
                        readwsdlfile(filepath + "\\" + filelist[i]);
                    }
                }
            }
        } catch (FileNotFoundException e) {
            System.out.println("readfile()      Exception:" + e.getMessage());
        }
        if_if();
        return true;
    }
    public static void parseWsdl(Document document) {
        Element root = document.getRootElement();
        for (Iterator iter = root.elementIterator(); iter.hasNext();){
            Element element = (Element) iter.next();
            String attr=element.attributeValue("name");
            if((attr!=null)&&attr.equals("main")){
                parseMain(element,activity);
            }
        }
```

```
		for(Activity component : activity.components){
			if(component.getClass().getName().equals("com.test.bean.Structure")){
				If_.put(component.getChild(),component.getName());
			}
		}
	}
	public static void parseMain(Element node,Activity act){
		for (Iterator iter = node.elementIterator(); iter.hasNext();){
			Element element = (Element) iter.next();
			String type=element.getName();
			switch(type){
			case "assign":
			case "invoke":
			case "receive":
			case "reply":
				parseYuan(element,act);
				break;
			case "flow":
				parseFlow(element,act);
				break;
			case "sequence":
				parseSeq(element,act);
				break;
			case "if":
				parseIf(element,act);
				break;
		}
	}
	private static void parseSeq(Element node,Activity act){
		Structure seq=new Structure(node.getName());
		for (Iterator iter = node.elementIterator(); iter.hasNext();){
			Element element = (Element) iter.next();
			parseMain(element,act);
		}
	}
	private static void parseFlow(Element node,Activity act){
		String attr=node.attributeValue("name");
		System.out.println("	flow"+attr);
		if(attr!=null)
			collection.add(attr);
		for (Iterator iter = node.elementIterator(); iter.hasNext();){
			Element element = (Element) iter.next();
			parseMain(element,act);
		}
	}
	private static void parseYuan(Element node,Activity act) {
		String attr=node.attributeValue("name");
		String key=act.getName();
		if(If_Branch.containsKey(act.getName())){
			String value=If_Branch.get(key);
```

```
                If_Branch.put(act.getName(), value+attr+"#");
            }
            act.add(new Atom(attr));
            if(attr!=null)
                collection.add(attr);
        }
        private static void parseElse(Element node,Activity act){
            for (Iterator iter = node.elementIterator(); iter.hasNext();){
                Element element = (Element) iter.next();
                parseMain(element,act);
            }
        }
        private static void parseIf(Element node,Activity act) {
            String attr=node.attributeValue("name");
            String key=act.getName();
            Structure _if=new Structure(attr);
            activity.add(_if);
            if(If_Branch.containsKey(act.getName())){
                String value=If_Branch.get(key);
                If_Branch.put(act.getName(), value+attr+"#");
            }
            If_Branch.put(attr, "");
            If.add(attr);
            collection.add(attr);
            for (Iterator iter = node.elementIterator(); iter.hasNext();){
                Element element = (Element) iter.next();
                String type=element.getName();
                if(type.equals("condition"))
                    continue;
                if(type.equals("else"))
                    parseElse(element,_if);
                parseMain(element,_if);
            }
        }
        public static void if_if(){
            int n=ParseBpel.If_Branch.size();
            String[] key=new String[n];
            String[] value=new String[n];
            int i=0;
            for(Entry<String, String> entry: ParseBpel.If_Branch.entrySet()){
                key[i]=entry.getKey();
                value[i++]=entry.getValue();
            }
            for(i=0;i<n;i++){
                for(int j=0;j<n;j++){
                    if(i==j)
                        continue;
                    if(value[i].contains(key[j])){
                        ParseBpel.If_Branch.put(key[i],
        ParseBpel.If_Branch.get(key[i])+value[j]);
```

```
                    }
                }
            }
        }
}
package other;
import java.io.File;
import com.test.XMLHelper_Ran;
public class Name {
        public static void main(String args[]) throws Exception{
                String output="E:\\毕设实验\\Quote\\path.txt";
                File file = new File("C:\\Users\\Administrator\\Desktop\\实验\\quotemutant\\");
                File[] files = file.listFiles();
                for(int i = 0; i < files.length; i++){
                        System.out.println(files[i].getName());
                        XMLHelper_Ran.stringexport(files[i].getName()+"\n",output);
                }
        }
}
package other;
import java.io.BufferedReader;
import java.io.File;
import java.io.FileInputStream;
import java.io.InputStreamReader;
import java.util.regex.Pattern;
import com.test.XMLHelper_Ran;
public class Quote_Add {
        public static void main(String[] args) throws Exception{
                System.out.println("begin:");
                String output="E:\\毕设实验\\Quote\\path_new444.txt";
                String path1="E:\\毕设实验\\Quote\\path_Add.txt";
                String path2="E:\\毕设实验\\Quote\\path_new4.txt";
                File file1=new File(path1);
                InputStreamReader isr1=new InputStreamReader(new FileInputStream(file1));
                BufferedReader br1=new BufferedReader(isr1);
                File file2=new File(path2);
                InputStreamReader isr2=new InputStreamReader(new FileInputStream(file2));
                BufferedReader br2=new BufferedReader(isr2);
                String line1;
                while((line1=br1.readLine())!=null){
                        boolean b=Pattern.matches(".*bpel", line1);
                        if(b){
                                System.out.println(line1);
                                XMLHelper_Ran.stringexport(line1+"\n",output);
                                String line2;
                                while((line2=br2.readLine())!=null){
                                        if(line2.equals(line1)){
                                                System.out.println("        haha"+line2);
                                        }else{
                                                boolean b2=Pattern.matches(".*bpel", line2);
                                                if(b2){
                                                        break;
                                                }else{
                                                        System.out.println(line2);
                                                        XMLHelper_Ran.stringexport(line2+"\n",output);
                                                }
                                        }
                                }
                        }
                        else{
                                System.out.println(line1);
                                XMLHelper_Ran.stringexport(line1+"\n",output);
                        }
                }
        }
}
package other;
import java.io.BufferedReader;
import java.io.File;
import java.io.FileInputStream;
import java.io.InputStreamReader;
import java.util.ArrayList;
import java.util.regex.Pattern;
import com.test.XMLHelper_Ran;
public class Quote_PathAdd2 {
        public static void main(String[] args) throws Exception{
                System.out.println("begin:");
                String output="E:\\毕设实验\\Quote\\path_Add3.txt";
                String path1="E:\\毕设实验\\Quote\\path.txt";
                String path2="E:\\毕设实验\\Quote\\path_Quote_Mutant3.txt";
                int[] array={1,5,4,7,11};
                ArrayList list=new ArrayList();
                for(int num: array)
                        list.add(num);
                File file1=new File(path1);
                InputStreamReader isr1=new InputStreamReader(new FileInputStream(file1));
                BufferedReader br1=new BufferedReader(isr1);
                File file2=new File(path2);
                InputStreamReader isr2=new InputStreamReader(new FileInputStream(file2));
                BufferedReader br2=new BufferedReader(isr2);
                String line1;
                while((line1=br1.readLine())!=null){
                        boolean b=Pattern.matches(".*bpel", line1);
                        if(b){
                                System.out.println(line1);
                                XMLHelper_Ran.stringexport(line1+"\n",output);
                                String line2;
                                int count=0;
                                while((line2=br2.readLine())!=null){
                                        if(line2.equals(line1)){
```

```
                        count=0;
                    }else{
                        boolean b2=Pattern.matches(".*bpel", line2);
                        if(b2){
                            count=0;
                            break;
                        }else{
                            count++;
                            if(list.contains(count)){
                                System.out.println("add "+line2);
                                XMLHelper_Ran.stringexport(line2+"\n",output);
                            }
                        }
                    }
                }
            }
            else{
                System.out.println("1"+line1);
                XMLHelper_Ran.stringexport(line1+"\n",output);
            }
        }
    }
}
package other;
import java.io.BufferedReader;
import java.io.File;
import java.io.FileInputStream;
import java.io.InputStreamReader;
import java.util.ArrayList;
import java.util.regex.Pattern;
import com.test.XMLHelper_Ran;
public class Smart_PathAdd2 {
    public static void main(String[] args) throws Exception{
        System.out.println("begin:");
        String output="E:\\毕设实验\\SmartShelf\\path_temp3.txt";
        String path1="E:\\毕设实验\\SmartShelf\\path.txt";
        String path2="E:\\毕设实验\\SmartShelf\\path_old3.txt";
        int i=5;
        int[] array={21,22,23,24};
        ArrayList list=new ArrayList();
        for(int num: array)
            list.add(num);
        File file1=new File(path1);
        InputStreamReader isr1=new InputStreamReader(new FileInputStream(file1));
        BufferedReader br1=new BufferedReader(isr1);
        File file2=new File(path2);
        InputStreamReader isr2=new InputStreamReader(new FileInputStream(file2));
        BufferedReader br2=new BufferedReader(isr2);
        String line1;
        while((line1=br1.readLine())!=null){
```

```
				boolean b=Pattern.matches(".*bpel", line1);
				if(b){
					System.out.println(line1);
					XMLHelper_Ran.stringexport(line1+"\n",output);
					String line2;
					int count=0;
					while((line2=br2.readLine())!=null){
						if(line2.equals(line1)){
							count=0;
						}else{
							boolean b2=Pattern.matches(".*bpel", line2);
							if(b2){
								count=0;
								break;
							}else{
								count++;
								if(list.contains(count)){
									System.out.println("add "+line2);
									XMLHelper_Ran.stringexport(line2+"\n",output);
								}
							}
						}
					}
				}
				else{
					System.out.println("1"+line1);
					XMLHelper_Ran.stringexport(line1+"\n",output);
				}
			}
		}
}
package other;
import java.io.BufferedReader;
import java.io.File;
import java.io.FileInputStream;
import java.io.InputStreamReader;
import java.util.regex.Pattern;
import com.test.XMLHelper_Ran;
public class SmartShelf_Add {
	public static void main(String[] args) throws Exception{
		System.out.println("begin:");
		String output="E:\\毕设实验\\SmartShelf\\path_temp3333.txt";
		String path1="E:\\毕设实验\\SmartShelf\\path_temp333.txt";
		String path2="E:\\毕设实验\\SmartShelf\\path_Add.txt";
		File file1=new File(path1);
		InputStreamReader isr1=new InputStreamReader(new FileInputStream(file1));
		BufferedReader br1=new BufferedReader(isr1);
		File file2=new File(path2);
		InputStreamReader isr2=new InputStreamReader(new FileInputStream(file2));
		BufferedReader br2=new BufferedReader(isr2);
```

```
        String line1;
        while((line1=br1.readLine())!=null){
            boolean b=Pattern.matches(".*bpel", line1);
            if(b){
                System.out.println(line1);
                XMLHelper_Ran.stringexport(line1+"\n",output);
                String line2;
                while((line2=br2.readLine())!=null){
                    if(line2.equals(line1)){
                        System.out.println("        haha"+line2);
                    }else{
                        boolean b2=Pattern.matches(".*bpel", line2);
                        if(b2){
                            break;
                        }else{
                            System.out.println(line2);
                            XMLHelper_Ran.stringexport(line2+"\n",output);
                        }
                    }
                }
            }
            else{
                System.out.println(line1);
                XMLHelper_Ran.stringexport(line1+"\n",output);
            }
        }
    }
}
package other;
import java.io.BufferedReader;
import java.io.File;
import java.io.FileInputStream;
import java.io.InputStreamReader;
import java.util.regex.Pattern;
import com.test.XMLHelper_Ran;
public class Travel_Add {
    public static void main(String[] args) throws Exception{
        System.out.println("begin:");
        String output="E:\\毕设实验\\Travel\\path_new22222.txt";
        String path1="E:\\毕设实验\\Travel\\path_new2.txt";
        String path2="E:\\毕设实验\\Travel\\path_Add.txt";
        File file1=new File(path1);
        InputStreamReader isr1=new InputStreamReader(new FileInputStream(file1));
        BufferedReader br1=new BufferedReader(isr1);
        File file2=new File(path2);
        InputStreamReader isr2=new InputStreamReader(new FileInputStream(file2));
        BufferedReader br2=new BufferedReader(isr2);
        String line1;
        while((line1=br1.readLine())!=null){
            boolean b=Pattern.matches(".*bpel", line1);
```

```
                    if(b){
                        System.out.println(line1);
                        XMLHelper_Ran.stringexport(line1+"\n",output);
                        String line2;
                        while((line2=br2.readLine())!=null){
                            if(line2.equals(line1)){
                                System.out.println("       haha"+line2);
                            }else{
                                boolean b2=Pattern.matches(".*bpel", line2);
                                if(b2){
                                    break;
                                }else{
                                    System.out.println(line2);
                                    XMLHelper_Ran.stringexport(line2+"\n",output);
                                }
                            }
                        }
                    }
                    else{
                        System.out.println(line1);
                        XMLHelper_Ran.stringexport(line1+"\n",output);
                    }
                }
        }
}
package lizi;
import java.util.ArrayList;
import java.util.HashMap;
public class SusSetReduce {
        public HashMap<String,Integer> TruePath=new HashMap<String,Integer>();
        public HashMap<String,Integer> FalsePath=new HashMap<String,Integer>();
        public Parse parse;
        public static ArrayList S=new ArrayList();
        public void susipion(){
                S.add(null);
        }
        public   void predicate(){
                Node node;
        }
        public void atom(){
                Node node;
        }
}
package lizi;
import java.io.File;
import java.util.ArrayList;
public class RunTestCase {
        Parse parse;
        public File file=parse.BPELProgram;
        public boolean deploy(){
            return false;
        }
        public String sendMessage(ArrayList list){
            return "";
        }
        public boolean compareResult(String s1,String s2){
            return true;
        }
}
package lizi;
import java.util.ArrayList;
import java.util.HashMap;
import java.util.TreeMap;
public class Predicate_sort {
        SusSetReduce pd;
        public TreeMap<Double, String> map=new TreeMap<Double,String>();
        public TreeMap<Double, String> sober(HashMap TruePath,HashMap FalsePath){
            ArrayList list=SusSetReduce.S;
            return map;
        }
}
package lizi;
import java.io.File;
import java.io.FileNotFoundException;
import java.io.IOException;
import java.util.ArrayList;
import org.dom4j.Document;
public class Parse {
        public File BPELProgram;
        public ArrayList<Node> list;
        public File getFile(){
            return this.BPELProgram;
        }
        public boolean readwsdlfile(String filepath)  throws FileNotFoundException, IOException {
            return true;
        }
        public void parseWsdl(Document document){
            Node node;
        }
        public Node getChildren(Node node){
            return node;
        }
}
package lizi;
import java.util.ArrayList;
public class Output {
        private Predicate_sort s;
        private ArrayList list;
        public void result(){
}
```

```
}
package lizi;
import java.util.ArrayList;
public class Node {
	private int id;
	private String name;
	private String type;
	private int responseId;
	private int childNumber;
	private int beforeNumber;
	private ArrayList beforeNodes = new ArrayList();
	private ArrayList afterNodes = new ArrayList();
	private ArrayList chilidNodes = new ArrayList();
	Node(){
	}
	public int getBeforeNumber() {
		return beforeNumber;
	}
	public void setBeforeNumber(int beforeNumber) {
		this.beforeNumber = beforeNumber;
	}
	public ArrayList getChilidNodes() {
		return chilidNodes;
	}
	public void setChilidNodes(ArrayList chilidNodes) {
		this.chilidNodes = chilidNodes;
	}
	public int getId() {
		return id;
	}
	public void setId(int id) {
		this.id = id;
	}
	public String getName() {
		return name;
	}
	public void setName(String name) {
		this.name = name;
	}
	public String getType() {
		return type;
	}
	public void setType(String type) {
		this.type = type;
	}
	public int getResponseId() {
		return responseId;
	}
	public void setResponseId(int responseId) {
		this.responseId = responseId;
```

```
        }
        public int getChildNumber() {
              return childNumber;
        }
        public void setChildNumber(int childNumber) {
              this.childNumber = childNumber;
        }
        public ArrayList getBeforeNodes() {
              return beforeNodes;
        }
        public void setBeforeNodes(ArrayList beforeNodes) {
              this.beforeNodes = beforeNodes;
        }
        public ArrayList getAfterNodes() {
              return afterNodes;
        }
        public void setAfterNodes(ArrayList afterNodes) {
              this.afterNodes = afterNodes;
        }
        public String toString() {
              /*String ret = this.getId() + " " + this.getResponseId() + " "
                          + this.getAfterNodes().size() + " " + this.getType() + " "
                          + this.getName() + " "+this.getPredict();
              conditions = this.getConditions();
              return ret;
        }
        public int getbeforeNumber(Node node) {
              return node.beforeNodes.size();
        }
}
package com.test;
import java.io.File;
import java.io.FileNotFoundException;
import java.io.FileOutputStream;
import java.io.FileWriter;
import java.io.IOException;
import java.io.OutputStream;
import java.io.OutputStreamWriter;
import java.io.Writer;
import java.util.ArrayList;
import java.util.HashMap;
import java.util.Iterator;
import java.util.Map;
import java.util.Map.Entry;
import java.util.Set;
import org.dom4j.Attribute;
import org.dom4j.Document;
import org.dom4j.Element;
import org.dom4j.io.OutputFormat;
import org.dom4j.io.SAXReader;
import org.dom4j.io.XMLWriter;
import com.test.bpelbean.Message;
import com.test.bpelbean.Value;
public class XMLHelper_Ran {
        public static HashMap<String, ArrayList<String>> hashMap = new HashMap<String,
        ArrayList<String>>();
        public static HashMap<String, String> hashBpelHashMap = new HashMap<String,
        String>();
        public static HashMap<String, ArrayList<String>> finalHashMap = new HashMap<String,
        ArrayList<String>>();
        public static void main(String args[]) throws FileNotFoundException, IOException {
                  String bpelPath ="D:\\Workspace1\\DebugTest\\bpel\\SmartShelfProcess.bpel";
                  readwsdlfile(bpelPath);
                  Set set=hashMap.entrySet();
                  for(Object o:set){
                          Map.Entry entry=(Map.Entry)o;
                          String key=(String) entry.getKey();
                          ArrayList list=(ArrayList) entry.getValue();
                          System.out.println(key+"    "+list);
                }
          }
        public static Document openXMLFile(String filePath) {
                Document document = null;
                SAXReader reader = new SAXReader();
                try {
                        File file = new File(filePath);
                        document = reader.read(file);
                } catch (Exception e) {
                        e.printStackTrace();
                }
                return document;
        }
        public static boolean writeXMLFile(Document document, String filePath) {
                boolean flag = false;
                XMLWriter writer = null;
                OutputFormat format = OutputFormat.createPrettyPrint();
                format.setEncoding("UTF-8");
                try {
                        File file = new File(filePath);
                        writer = new XMLWriter(new FileWriter(file), format);
                        writer.write(document);
                        writer.flush();
                        writer.close();
                        flag = true;
                } catch (Exception e) {
                        flag = false;
                        e.printStackTrace();
                }
                return flag;
        }
```

```java
public static boolean readwsdlfile(String filepath)
        throws FileNotFoundException, IOException {
    try {
        File file = new File(filepath);
        if (!file.isDirectory()) {
            System.out.println("absolutepath=" + file.getAbsolutePath());
            System.out.println("name=" + file.getName());
            String absolutepath = file.getAbsolutePath();
            Document document = openXMLFile(absolutepath);
            parseWsdl(document);
        } else if (file.isDirectory()) {
            String[] filelist = file.list();
            for (int i = 0; i < filelist.length; i++) {
                File readfile = new File(filepath + "\\" + filelist[i]);
                if (!readfile.isDirectory()) {
                    if (filelist[i].contains(".wsdl")) {
                        String absolutepath = readfile.getAbsolutePath();
                        Document document = openXMLFile(absolutepath);
                        parseWsdl(document);
                    }
                } else if (readfile.isDirectory()) {
                    readwsdlfile(filepath + "\\" + filelist[i]);
                }
            }
        }
    } catch (FileNotFoundException e) {
        System.out.println("readfile()     Exception:" + e.getMessage());
    }
    return true;
}
public static void parseWsdl(Document document) {
    Element root = document.getRootElement();
    ArrayList<Message> arrayList = new ArrayList<Message>();
    ArrayList<Value> values = new ArrayList<Value>();
    for (Iterator iter = root.elementIterator(); iter.hasNext();) {
        Element element = (Element) iter.next();
        if (element.getName().equals("message")) {
            Message message = new Message();
            Attribute attribute = element.attribute("name");
            String key = attribute.getValue();
            Element partElement = element.element("part");
            Attribute elAttribute = partElement.attribute("element");
            String refString = elAttribute.getValue();
            String a[] = refString.split(":");
            message.setName(key);
            message.setRefsString(a[1]);
            arrayList.add(message);
        } else if (element.getName().equals("types")) {
            Element tyElement = element.element("schema");
            for (Iterator iterator = tyElement.elementIterator(); iterator
                    .hasNext();) {
                Value value = new Value();
                Element elementOut = (Element) iterator.next();
                String refName = elementOut.attribute("name").getValue();
                ArrayList<String> arrayList2 = new ArrayList<String>();
                Element elementSequence = elementOut.element("complexType")
                        .element("sequence");
                if (elementSequence != null) {
                    for (Iterator iterator2 = elementSequence
                            .elementIterator(); iterator2.hasNext();) {
                        Element elementInner = (Element) iterator2.next();
                        String value1 = elementInner.attribute("name")
                                .getValue();
                        arrayList2.add(value1);
                    }
                }
                value.setRefName(refName);
                value.setValue(arrayList2);
                values.add(value);
            }
        }
    }
    addtoHashmap(arrayList, values);
}
private static void addtoHashmap(ArrayList<Message> arrayList,
        ArrayList<Value> values) {
    for (int i = 0; i < arrayList.size(); i++) {
        String name = arrayList.get(i).getName();
        String refname = arrayList.get(i).getRefsString();
        for (int j = 0; j < values.size(); j++) {
            String reString = values.get(j).getRefName();
            if (refname.equals(reString)) {
                hashMap.put(name, values.get(j).getValue());
            }
        }
    }
}
public static void parseBpelVariable(String filePath) {
    Document document = openXMLFile(filePath);
    Element root = document.getRootElement();
    Element vsElement = root.element("variables");
    for (Iterator iterator = vsElement.elementIterator(); iterator
            .hasNext();) {
        Element vElement = (Element) iterator.next();
        String messageType = null;
        if (vElement.attribute("messageType") != null) {
            messageType = vElement.attribute("messageType").getValue();
            messageType = messageType.split(":")[1];
        }

```

```
                String name = vElement.attribute("name").getValue();
                hashBpelHashMap.put(messageType, name);
            }
            bpelWsdlParse();
        }
        public static void bpelWsdlParse() {
            for (Entry<String, ArrayList<String>> entry : hashMap.entrySet()) {
                String meString = entry.getKey();
                ArrayList<String> valueList = entry.getValue();
                for (Entry<String, String> entry2 : hashBpelHashMap.entrySet()) {
                    String meString2 = entry2.getKey();
                    String nameString = entry2.getValue();
                    if (meString.equals(meString2)) {
                        finalHashMap.put(nameString, valueList);
                        System.out.println(nameString+" "+valueList+"llllllllllllllllll");
                    }
                }
            }
        }
        public static boolean stringexport(String parameter1, String path)      throws Exception {
            boolean flag = false;
            OutputStream os;
            try {
                os = new FileOutputStream(new File(path), true);
                Writer fos = new OutputStreamWriter(os);
                fos.write(parameter1);
                fos.flush();
                fos.close();
                flag = true;
            } catch (FileNotFoundException e) {
                e.printStackTrace();
            }
            return flag;
        }
}
package com.test;
import java.util.Calendar;
import org.apache.axiom.om.OMAbstractFactory;
import org.apache.axiom.om.OMElement;
import org.apache.axiom.om.OMFactory;
import org.apache.axiom.om.OMNamespace;
import org.apache.axis2.AxisFault;
import org.apache.axis2.addressing.EndpointReference;
import org.apache.axis2.client.Options;
import org.apache.axis2.client.ServiceClient;
public class TravelClient
{
        public synchronized static OMElement sendmessage(String string, int amount) throws
        Exception
        {
                OMElement res=null;
                ServiceClient sc=null;
                  try {
                        sc = new ServiceClient();
                        Options opts = new Options();
                        opts.setTo(new EndpointReference(
                                    "http://localhost:8080/ode/processes/TravelAgency"));
                        opts.setAction("www.ustb.edu.cn/bpel/travelagency/process");
                        sc.setOptions(opts);
                        long startTime = Calendar.getInstance().getTimeInMillis();
                          res = sc.sendReceive(createPayLoad(string,amount));
                        long endTime = Calendar.getInstance().getTimeInMillis();
                        System.out.println(string+"&&"+amount);
                      System.out.println(res);

                  } catch (AxisFault e) {
                        e.printStackTrace();
                  }finally{
                        sc.cleanupTransport();
                  }
                return res;
        }
          public static OMElement createPayLoad(String string,int parameter2){
                      OMFactory fac = OMAbstractFactory.getOMFactory();
                      OMNamespace omNs =
        fac.createOMNamespace("www.ustb.edu.cn/bpel/travelagency", "ustb");
                      OMNamespace omNs1 =
        fac.createOMNamespace("www.ustb.edu.cn/bpel/travelagency", "ustb");
                      OMElement method = fac.createOMElement("TravelAgencyRequest",omNs);
                      OMElement value1 = fac.createOMElement("name",omNs1);
                      OMElement value2 = fac.createOMElement("amount",omNs1);
                      value1.setText(string+"");
                      value2.setText(parameter2+"");
                      method.addChild(value1) ;
                      method.addChild(value2) ;
                    System.out.println(method);
                      return method;
                } ;
}
package com.test;
import java.util.HashMap;
import java.util.Iterator;
import java.util.Map;
import org.dom4j.Document;
import org.dom4j.Element;
public class StringMapper {
        private static Map<String, String> stringMap = new HashMap<String, String>();
        static {
                Document document = XMLHelper_Ran.openXMLFile("string_en.xml");
                Element root = document.getRootElement();
```

```
		for (Iterator iter = root.elementIterator(); iter.hasNext();) {
			Element element = (Element) iter.next();
			String name = null;
			String value = null;
			for (Iterator ii = element.elementIterator(); ii.hasNext();) {
				Element e = (Element) ii.next();
				if (e.getName().equals("name")) {
					name = e.getText();
				} else if (e.getName().equals("value")) {
					value = e.getText();
				}
			}
			if (name != null && value != null) {
				stringMap.put(name, value);
			}
		}
	}
	public static String get(String name) {
		return stringMap.get(name);
	}
}
package com.test;
import com.test.update.InventoryUpdate;
import com.test.update.ProductUpdate;
import com.test.update.ShelfUpdate;
import com.test.update.WarehouseUpdate;
public class smartUpdate {
	public static void update(){
	InventoryUpdate inventory=new InventoryUpdate();
	inventory.inventoryupdate();
	ProductUpdate product=new ProductUpdate();
	product.productupdate();
	ShelfUpdate shelf=new ShelfUpdate();
	try {
		shelf.shelfupdate();
	} catch (Exception e) {
		e.printStackTrace();
	}
	WarehouseUpdate warehouse=new WarehouseUpdate();
	try {
		warehouse.warehouseupdate();
		Thread.sleep(1000);
	} catch (Exception e) {
		e.printStackTrace();
	}
	}
}
package com.test;
import java.util.Calendar;
import org.apache.axiom.om.OMAbstractFactory;
import org.apache.axiom.om.OMElement;
import org.apache.axiom.om.OMFactory;
import org.apache.axiom.om.OMNamespace;
import org.apache.axis2.AxisFault;
import org.apache.axis2.addressing.EndpointReference;
import org.apache.axis2.client.Options;
import org.apache.axis2.client.ServiceClient;
public class quoteprocessclient
{
    public synchronized static OMElement sendmessage(String name, int amount) throws
    Exception
    {
        OMElement res=null;
        ServiceClient sc =null;
        try {
            sc = new ServiceClient();
            Options opts = new Options();
            opts.setTo(new EndpointReference(
                    "http://localhost:8080/ode/processes/QuoteProcess"));
            opts.setAction("www.ustb.edu.cn/bpel/quoteprocess/process");
            sc.setOptions(opts);
            long startTime = Calendar.getInstance().getTimeInMillis();
            res = sc.sendReceive(createPayLoad(name,amount));
            long endTime = Calendar.getInstance().getTimeInMillis();
            System.out.println(name+"&&"+amount);
            System.out.println(res);
        } catch (AxisFault e) {
            e.printStackTrace();
        } finally{
            sc.cleanupTransport();
        }
        return res;
    }
    public static OMElement createPayLoad(String parameter1,int parameter2){
        OMFactory fac = OMAbstractFactory.getOMFactory();
        OMNamespace omNs =
    fac.createOMNamespace("www.ustb.edu.cn/bpel/quoteprocess", "ustb");
        OMNamespace omNs1 =
    fac.createOMNamespace("www.ustb.edu.cn/bpel/quoteprocess", "ustb");
        OMElement method = fac.createOMElement("QuoteProcessRequest",omNs);
        OMElement value1 = fac.createOMElement("Name",omNs1);
        OMElement value2 = fac.createOMElement("Amount",omNs1);
        value1.setText(parameter1);
        value2.setText(parameter2+"");
        method.addChild(value1) ;
        method.addChild(value2) ;
        System.out.println("���,���Щ:"+method);
        return method;
    }
}
```

```
package com.test;
import java.util.ArrayList;
import java.util.Iterator;
import java.util.List;
import org.apache.axiom.om.OMElement;
import org.apache.axiom.om.OMNode;
public class OmelementParse
{
	public static List<String> getresult(OMElement element)
	{
		if (element == null){
			return null;
		}
		Iterator iterator = element.getChildElements();
		List<String> list = new ArrayList<String>();
		while (iterator.hasNext())
		{
			OMNode omNode = (OMNode) iterator.next();
			if (omNode.getType() == OMNode.ELEMENT_NODE){
				OMElement omElement = (OMElement) omNode;
				String temp=omElement.getText().trim();
				list.add(temp);
			}
		}
		return list;
	}
}
package com.test;
import java.util.Calendar;
import org.apache.axiom.om.OMAbstractFactory;
import org.apache.axiom.om.OMElement;
import org.apache.axiom.om.OMFactory;
import org.apache.axiom.om.OMNamespace;
import org.apache.axis2.AxisFault;
import org.apache.axis2.addressing.EndpointReference;
import org.apache.axis2.client.Options;
import org.apache.axis2.client.ServiceClient;
public class MessageClient
{
	public synchronized static OMElement sendmessage(String name, int amount) throws
	Exception
	{
		OMElement res=null;
		ServiceClient sc=null;
		  try {
			    sc = new ServiceClient();
			  Options opts = new Options();
			  opts.setTo(new EndpointReference(
					"http://localhost:8080/ode/processes/SmartShelfProcess"));
			  opts.setAction("ustb.bpel.org/process");
```

```
                    sc.setOptions(opts);
                    long startTime = Calendar.getInstance().getTimeInMillis();
                      res = sc.sendReceive(createPayLoad(name,amount));
                    long endTime = Calendar.getInstance().getTimeInMillis();
                    System.out.println(name+"&&"+amount);
                  System.out.println("SmartShelf-----"+res);
            } catch (AxisFault e) {
                    e.printStackTrace();
            }finally{
                    sc.cleanupTransport ();
            }
          return res;
        }
         public static OMElement createPayLoad(String parameter1,int parameter2){
                  OMFactory fac = OMAbstractFactory.getOMFactory();
                  OMNamespace omNs = fac.createOMNamespace("ustb.bpel.org", "ustb");
                  OMNamespace omNs1 = fac.createOMNamespace("ustb.bpel.org", "ustb");
                  OMElement method = fac.createOMElement("SmartShelfProcessRequest",omNs);
                  OMElement value1 = fac.createOMElement("name",omNs1);
                  OMElement value2 = fac.createOMElement("amount",omNs1);
                  value1.setText(parameter1);
                  value2.setText(parameter2+"");
                  method.addChild(value1) ;
                  method.addChild(value2) ;
                System.out.println(method);
                  return method;
          } ;
}
package com.test;
import java.io.File;
public class getFiles {
        public static void main(String[] args) {
                File files = new File(
                                "C:\\Users\\Administrator\\Desktop\\实验\\travelmutant\\");
                File[] filess = files.listFiles();
                for (int i = 0; i < filess.length; i++){
                        System.out.println(filess[i]);
                }
                System.out.println(filess.length);
        }
}
package com.test;
import java.io.BufferedInputStream;
import java.io.BufferedOutputStream;
import java.io.BufferedReader;
import java.io.File;
import java.io.FileInputStream;
import java.io.FileNotFoundException;
import java.io.FileReader;
import java.io.IOException;
import java.io.PrintStream;
import java.io.RandomAccessFile;
import java.io.Reader;
import java.util.ArrayList;
import java.util.HashMap;
import javax.swing.JTextArea;
import com.test.bpelbean.TestcaseNode;
public class FileControl {
    public static String re=null;
    public static Boolean deletefile( String deployname, String abstractfilename) throws
    InterruptedException {
        boolean flag = false;
        String path = "D:/Programs/tomcat 6old/webapps/ode/WEB-INF/processes";
        File directory=new File(path);
        if (!directory.isDirectory()) {
            System.out.println("删除文件夹找不到");
            flag = false;
        }else{
            flag=parse(directory,deployname,abstractfilename);
        }
        return flag;
    }
    public static void readFileByLines(String fileName, JTextArea expectArea) {
        File file = new File(fileName);
        BufferedReader reader = null;
        try {
            System.out.println("以行为单位读取文件内容，一次读一整行：");
            reader = new BufferedReader(new FileReader(file));
            String tempString = null;
            int line = 1;
            while ((tempString = reader.readLine()) != null) {
                expectArea.append(tempString+"\r\n");
                System.out.println("line " + line + ": " + tempString);
                line++;
            }
            reader.close();
            }catch (IOException e) {
                e.printStackTrace();
            }finally {
                if (reader != null) {
                    try {
                        reader.close();
                    } catch (IOException e1) {
                }
            }
        }
    }
    public static void readFileByLines(String fileName,HashMap<String,Integer> Path,String
    flag,JTextArea expectArea) {
        File file = new File(fileName);
```

```
		BufferedReader reader = null;
		try {
			System.out.println("以行为单位读取文件内容，一次读一整行：");
			reader = new BufferedReader(new FileReader(file));
			String tempString = null;
			int line = 1;
			while ((tempString = reader.readLine()) != null) {
				if(tempString.contains(flag)){
					if(Path.containsKey(tempString)){
						Path.put(tempString, Path.get(tempString)+1);
					}else{
						Path.put(tempString, 1);
					}
					expectArea.append(tempString+"\r\n");
					System.out.println("line " + line + ": " + tempString);
					line++;
				}
			}
			reader.close();
			}catch (IOException e) {
				e.printStackTrace();
			}finally {
				if (reader != null) {
					try {
						reader.close();
					} catch (IOException e1) {
				}
			}
		}
	}
	public static ArrayList readFile(String fileName, JTextArea expectArea) {
		File file = new File(fileName);
		BufferedReader reader = null;
		ArrayList list=new ArrayList();
		try{
			System.out.println("以行为单位读取文件内容，一次读一整行：");
			reader = new BufferedReader(new FileReader(file));
			String tempString = null;
			int line = 1;
			tempString=reader.readLine();
			System.out.println(reader.readLine());
			System.out.println(tempString);
			String[] str=tempString.split("#");
			list.add("entry");
			list.add("receiveInput");
			expectArea.append("entry");
			expectArea.append("receiveInput");
			for(int i=0;i<str.length;i++){
				expectArea.append(str[i]+" ");
				list.add(str[i]);
```

```
                    if(i%5==0){
                        expectArea.append("\n");
                    }
                }
                expectArea.append("replyOutput");
                list.add("replyOutPut");
            }catch (IOException e) {
                e.printStackTrace();
            }finally {
                if (reader != null) {
                    try {
                        reader.close();
                    } catch (IOException e1) {
                        }
                    }
                }
                return list;
        }
        public static ArrayList readFileEndLine(String fileName) {
                File file = new File(fileName);
                BufferedReader reader = null;
                ArrayList list=new ArrayList();
                try {
                    System.out.println("以行为单位读取文件内容，一次读一整行：");
                    reader = new BufferedReader(new FileReader(file));
                    String tempString = null;
                    int line = 1;
                    String s=null;
                   while((s=reader.readLine())!=null){
                       s=tempString;
                   }
                    System.out.println(tempString);
                    String[] str=tempString.split("#");
                    list.add("entry");
                    list.add("receiveInput");
                    for(int i=0;i<str.length;i++){
                        list.add(str[i]);
                    }
                   list.add("replyOutPut");
                } catch (IOException e) {
                    e.printStackTrace();
                } finally {
                    if (reader != null) {
                        try {
                            reader.close();
                        } catch (IOException e1) {
                        }
                    }
                }
                return list;
```

```
	}
	public static void parseWsdlPath(String deployname,File directory){
		if(directory.isDirectory()){
			File[] files=directory.listFiles();
			for(int i=0;i<files.length;i++){
				if(files[i].isFile()){
					String curName=files[i].getName();
					if(curName.equals(deployname)){
						System.out.println(files[i].getParentFile().getAbsolutePath());
						re=files[i].getParentFile().getAbsolutePath();
					}
				}else{
					File directorynew=new File(directory+"/"+files[i].getName());
					parseWsdlPath(deployname,directorynew);
				}
			}
		}
	}
	private static boolean parse(File directory, String deletename,String abstractfilename) throws
	InterruptedException {
		boolean flag=false;
		if(directory.isDirectory()){
			File[] files = directory.listFiles();
			for (int i = 0; i < files.length; i++) {
				if (files[i].isFile()) {
					String oldname = files[i].getName();
					if (oldname.equals(deletename)) {
						files[i].delete();
						Thread.sleep(1000);
						try {
							copyfile(files[i].getAbsolutePath(),abstractfilename);
						} catch (Exception e) {
							e.printStackTrace();
						}
						System.out.println("删除 bpel 文件成功");
						String path = "D:/Programs/tomcat
	6old/webapps/ode/WEB-INF/processes";
						File odepath=new File(path);
						File[] outer=odepath.listFiles();
						for(int j=0;j<outer.length;j++){
							System.out.println(outer[j].getName());
							if(outer[j].isFile()){
								String
	deployname=files[i].getParentFile().getName()+".deployed";
								System.out.println(outer[j].getName()+" "+deployname);
								if(deployname.equals(outer[j].getName())){
									System.out.println("删除.deployed 文件成功");
									outer[j].delete();
									Thread.sleep(100);
									flag=true;
```

```
                                }
                            }
                        }
                    }
                }else {
                    File directorynew=new File(directory+"/"+files[i].getName());
                    parse(directorynew, deletename, abstractfilename);
                }
            }
        }
        return flag;
    }
    public static Boolean copyfile(String topath, String frompath) throws Exception
    {
        boolean flag=false;
        System.out.println(topath+"~~~~~~~~~~~~"+frompath);
        File oldfile=new File(topath);
        File newfile=new File(frompath);
        System.out.println(newfile.exists()+"^^^^^^^^^^^"+newfile.isFile());
        if(oldfile.exists()){
            oldfile.delete();
        }
        oldfile.createNewFile();
        if((!newfile.exists())||(!newfile.isFile()))
        {
            System.out.println("路径不是一个 directory 或者文件夹不存在");
            flag=false;
        }else{
            FileInputStream fin = new FileInputStream(newfile.getAbsolutePath());
            BufferedInputStream bin = new BufferedInputStream(fin);
            PrintStream pout = new PrintStream(oldfile);
            BufferedOutputStream bout = new BufferedOutputStream(pout);
            int total =bin.available();
            int count;
            while((count = bin.available())!= 0)
            {
                int c = bin.read();
                bout.write((char)c);
            }
            bout.close();
            pout.close();
            bin.close();
            fin.close();
            flag=true;
        }
        return flag;
    }
    public static String comparefile(String oldpath, String newpath) throws Exception
    {   XMLHelper_Ran export = new XMLHelper_Ran();
        String result="";
```

```
          boolean flag=false;
          File oldfile=new File(oldpath);
          File newfile=new File(newpath);
          if(!oldfile.isFile())
          {
                System.out.println("源文件不是一个文件");
                flag=false;}
          if(!newfile.isFile())
          {
                System.out.println("对比文件不是一个文件");
                flag=false;
          }
          try {
                Reader fin1 = new FileReader(oldfile);
                Reader fin2 = new FileReader(newfile);
                BufferedReader bin1 = new BufferedReader(fin1);
                BufferedReader bin2 = new BufferedReader(fin2);
                String s1=null;
                String s2=null;
                while(((s1=bin1.readLine())!=null))
                {
                      s2=bin2.readLine();
                      if(!s1.equals(s2)) {
                            XMLHelper_Ran.stringexport(newfile.getAbsolutePath()+"\n",
                                        "C:\\Users\\Administrator\\Desktop\\实验
    \\smarshelfSlicing\\smartcompare1.txt");
                            XMLHelper_Ran.stringexport("原始版本：  "+s1+"\n"+"故障版本：
    "+s2+"\n",
                                        "C:\\Users\\Administrator\\Desktop\\实验
    \\smarshelfSlicing\\smartcompare1.txt");
                            XMLHelper_Ran.stringexport("\n\n",
                                        "C:\\Users\\Administrator\\Desktop\\实验
    \\smarshelfSlicing\\smartcompare1.txt");
                            result="different";
                            break;
                      }else
                      {
                            result="same";
                      }
                 }
          } catch (FileNotFoundException e) {
                e.printStackTrace();
          }
          return result;
    public    TestcaseNode gettestcase(String testcasepath)
    {
          String line = "";
          TestcaseNode testcasenode = null;
          String[] temp1;
          int[] temp = null;
```

```
                @SuppressWarnings("rawtypes")
                ArrayList templist = new ArrayList();
                try {
                        RandomAccessFile rf = new RandomAccessFile(testcasepath, "rw");
                        try {
                                for (line = rf.readLine(); line != null; line = rf.readLine()) {
                                        temp1 = line.split("#");
                                        for(int i=0;i<temp1.length;i++)
                                                temp[i]=Integer.parseInt(temp1[i]);
                                        System.out.println("temp'length:" + temp.length);
                                        if(temp.length==2)
                                        {
                                        }
                                }
                        } catch (IOException e) {
                                e.printStackTrace();
                        }
                } catch (FileNotFoundException e) {
                        e.printStackTrace();
                }
                return testcasenode;
        }
}
package com.test;
import java.io.IOException;
public class EngineImpl    implements Engine {
        public    boolean deploy(String file1,String abstractfilename) throws InterruptedException {
                        boolean flag=false;
                System.out.println(abstractfilename.endsWith("."));
                flag=FileControl.deletefile(file1,abstractfilename);
                return flag;
        }
        public boolean start() {
                String line = "";
                boolean flag=false;
                Runtime runtime = Runtime.getRuntime();
                try {
                        Process process = runtime.exec("D://Programs//tomcat 6old//bin//startup.bat");
                        flag=true;
                } catch (IOException e) {
                        e.printStackTrace();
                }
                return flag;
        }
}
package com.test;
public interface Engine {
        public    boolean start();
        public boolean deploy(String file1, String abstractfilename) throws InterruptedException;
}
```

```
package com.test;
import java.util.Iterator;
import org.dom4j.Attribute;
import org.dom4j.Element;
public class ElementWraper {
	Element element;
	private String xPath;
	public ElementWraper() {
	}
	public ElementWraper(Element element) {
		this.element = element;
	}
	public Element getElement() {
		return element;
	}
	public void setElement(Element element) {
		this.element = element;
	}
	public String getXPath() {
		return this.xPath;
	}
	public void setXPath(String xPath) {
		this.xPath = xPath;
	}
	public String toString() {
		StringBuffer sb = new StringBuffer();
		sb.append(element.getName() + " ");
		for (Iterator iter = element.attributeIterator(); iter.hasNext();) {
			Attribute attribute = (Attribute) iter.next();
			sb.append(attribute.getName()).append("=")
					.append(attribute.getValue()).append(" ");
		}
		return sb.toString();
	}
}
package com.test;
import java.io.BufferedReader;
import java.io.BufferedWriter;
import java.io.File;
import java.io.FileNotFoundException;
import java.io.FileReader;
import java.io.FileWriter;
import java.io.IOException;
import java.io.Reader;
public class DeleteNull {
	public static void main(String[] args) {
	File oldfile = new File("C:\\Users\\Administrator\\Desktop\\实验
		try {
			Reader fin1 = new FileReader(oldfile);
			BufferedReader reader = new BufferedReader(fin1);
```