```
package ui;
import java.awt.GridLayout;
import java.awt.event.ActionEvent;
import java.awt.event.MouseAdapter;
import java.awt.event.MouseEvent;
import java.util.Map;
import javax.swing.AbstractAction;
import javax.swing.BorderFactory;
import javax.swing.ComboBoxModel;
import javax.swing.DefaultComboBoxModel;
import javax.swing.JButton;
import javax.swing.JComboBox;
import javax.swing.JLabel;
import javax.swing.JList;
import javax.swing.JPanel;
import javax.swing.JScrollPane;
import javax.swing.JTable;
import javax.swing.JTextArea;
import javax.swing.ListModel;
import javax.swing.ListSelectionModel;
import javax.swing.SwingUtilities;
import javax.swing.WindowConstants;
import javax.swing.border.BevelBorder;
import javax.swing.border.LineBorder;
import javax.swing.event.ListSelectionEvent;
import javax.swing.event.ListSelectionListener;
import javax.swing.table.DefaultTableModel;
import javax.swing.table.TableModel;
import cn.edu.ustb.webservicetester.model.OperationInfo;
import cn.edu.ustb.webservicetester.model.ParameterInfo;
public class PreferenceFrame    extends javax.swing.JFrame {
        private JPanel jPanel1;
        private JScrollPane jScrollPane1;
        private JScrollPane jScrollPane2;
        private AbstractAction nextAction;
        private JButton jButtonPre;
        private JButton jButtonNext;
        private JComboBox testStrategyJComboBox;
        private JLabel testStrategyJLabel;
        private DefaultTableModel paraInfoTableModel;
        private JTable paraInfoJTable;
        private JTextArea paraResJTextArea;
        private JLabel paraResJLabel;
        private JLabel paraInfoJLabel;
        private AbstractAction operationSelectionAction;
        private JComboBox operationComboBox;
        private JLabel opSelectionJLabel;
        private java.util.List<OperationInfo> opInfoList;
        private OperationInfo selectedOperationInfo;
        private java.util.List<ParameterInfo> paraInfoList;
private java.util.Map<String, String> setting = null;
public static void main(String[] args) {
	SwingUtilities.invokeLater(new Runnable() {
		public void run() {
			PreferenceFrame inst = new PreferenceFrame();
			inst.setLocationRelativeTo(null);
			inst.setVisible(true);
		}
	});
}
public PreferenceFrame() {
	super();
	initGUI();
}
public PreferenceFrame(Map<String, String> setting, java.util.List list) {
	super();
	this.setting = setting;
	initGUI(list);
}
private void initGUI() {
	initGUI(null);
}
private void initGUI(java.util.List<OperationInfo> oil) {
	this.opInfoList = oil;
	try {
		setTitle("操作选择");
		setResizable(false);
		GridLayout thisLayout = new GridLayout(1, 1);
		getContentPane().setLayout(thisLayout);
		setDefaultCloseOperation(WindowConstants.DISPOSE_ON_CLOSE);
		{
			jPanel1 = new JPanel();
			getContentPane().add(jPanel1);
			jPanel1.setLayout(null);
			jPanel1.setBackground(new java.awt.Color(228,236,243));
			jPanel1.setPreferredSize(new java.awt.Dimension(506, 349));
			{
				opSelectionJLabel = new JLabel();
				jPanel1.add(opSelectionJLabel);
opSelectionJLabel.setText("\u8bf7\u9009\u62e9\u5f85\u6d4b\u8bd5\u7684\u64cd\u4f5c");
				opSelectionJLabel.setBounds(12, 36, 128, 17);
			}
			{
				ComboBoxModel operationSelectionComboBoxModel;
				if (opInfoList == null)
					operationSelectionComboBoxModel =
						new DefaultComboBoxModel(
								new String[] { "Item One", "Item Two" });
				else {
					String[] opNames = new String[opInfoList.size()];
```

```
                        for (int i = 0; i < opNames.length; ++i) {
                            opNames[i] = opInfoList.get(i).getOpName();
                        }
                        operationSelectionComboBoxModel = new DefaultComboBoxModel(opNames);
                    }
                    operationComboBox = new JComboBox();
                    jPanel1.add(operationComboBox);
                    operationComboBox.setModel(operationSelectionComboBoxModel);
                    operationComboBox.setBounds(140, 32, 162, 24);
                    operationComboBox.setAction(getOperationSelectionAction());
                }
                {
                    jScrollPane1 = new JScrollPane();
                    jPanel1.add(jScrollPane1);
                    jScrollPane1.setBounds(24, 101, 183, 132);
                    jScrollPane1.setViewportView(getParaInfoJTable());
                }
                {
                    jScrollPane2 = new JScrollPane();
                    jPanel1.add(jScrollPane2);
                    jPanel1.add(getJLabelParameterInformation());
                    jPanel1.add(getParaResJLabel());
                    jPanel1.add(getTestStrategyJLabel());
                    jPanel1.add(getTestStrategyJComboBox());
                    jPanel1.add(getJButtonNext());
                    jPanel1.add(getJButtonPre());
                    jScrollPane2.setBounds(290, 101, 183, 132);
                    jScrollPane2.setViewportView(getParaResJTextArea());
                }
            }
            pack();
            this.setSize(500, 444);
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
    private AbstractAction getOperationSelectionAction() {
        if(operationSelectionAction == null) {
            operationSelectionAction = new AbstractAction("", null) {
                public void actionPerformed(ActionEvent evt) {
                    int opIndexSelected = operationComboBox.getSelectedIndex();
                    selectedOperationInfo = opInfoList.get(opIndexSelected);
                    paraInfoList = selectedOperationInfo.getParaList();
                    int oldPInfoCount = paraInfoTableModel.getRowCount();
                    while (oldPInfoCount-- > 0) {
                        paraInfoTableModel.removeRow(0);
                    }
                    for (ParameterInfo pi : paraInfoList) {
                        paraInfoTableModel.addRow(new String[] {pi.getName(),
pi.getClassInfo()});
                    }
                }
            };
        }
        return operationSelectionAction;
    }
    private JLabel getJLabelParameterInformation() {
        if(paraInfoJLabel == null) {
            paraInfoJLabel = new JLabel();
            paraInfoJLabel.setText("\u8f93\u5165\u53c2\u6570");
            paraInfoJLabel.setBounds(24, 73, 61, 17);
        }
        return paraInfoJLabel;
    }
    private JLabel getParaResJLabel() {
        if(paraResJLabel == null) {
            paraResJLabel = new JLabel();
            paraResJLabel.setText("\u53c2\u6570\u8bf4\u660e");
            paraResJLabel.setBounds(274, 73, 64, 17);
        }
        return paraResJLabel;
    }
    private JTextArea getParaResJTextArea() {
        if(paraResJTextArea == null) {
            paraResJTextArea = new JTextArea();
            paraResJTextArea.setEditable(false);
        }
        return paraResJTextArea;
    }
    private JTable getParaInfoJTable() {
        if (paraInfoJTable == null) {
            String[] name = new String[] {"参数名", "数据类型"};
            int opIndexSelected = operationComboBox.getSelectedIndex();
            OperationInfo opInfo = opInfoList.get(opIndexSelected);
            paraInfoList = opInfo.getParaList();
            paraInfoTableModel = new DefaultTableModel(null, name);
            for (ParameterInfo pi : paraInfoList) {
                paraInfoTableModel.addRow(new String[] {pi.getName(), pi.getClassInfo()});
            }
            paraInfoJTable = new JTable(paraInfoTableModel) {
                public boolean isCellEditable(int row, int column) {
                    return false;
                }
            };
            paraInfoJTable.setSelectionMode(ListSelectionModel.SINGLE_SELECTION);
            paraInfoJTable.addMouseListener(new MouseAdapter() {
                public void mouseClicked(MouseEvent evt) {
                    int selectedRow = paraInfoJTable.getSelectedRow();
                    if (paraResJTextArea != null && paraInfoList != null &&
selectedRow >= 0)
        paraResJTextArea.setText(paraInfoList.get(selectedRow).getRestriction());
                                }
                        });
                }
                return paraInfoJTable;
        }
        private JLabel getTestStrategyJLabel() {
                if(testStrategyJLabel == null) {
                        testStrategyJLabel = new JLabel();
                        testStrategyJLabel.setText("\u6d4b\u8bd5\u7b56\u7565");
                        testStrategyJLabel.setBounds(12, 266, 53, 17);
                }
                return testStrategyJLabel;
        }
        private JComboBox getTestStrategyJComboBox() {
                if(testStrategyJComboBox == null) {
                        ComboBoxModel testStrategyJComboBoxModel =
                                        new DefaultComboBoxModel(
                                                        new String[] { "Random Testing", "Dynamic Random
Testing" });
                        testStrategyJComboBox = new JComboBox();
                        testStrategyJComboBox.setModel(testStrategyJComboBoxModel);
                        testStrategyJComboBox.setBounds(24, 295, 116, 24);
                        testStrategyJComboBox.setFont(new java.awt.Font("Times New Roman", 0, 12));
                }
                return testStrategyJComboBox;
        }
        private JButton getJButtonNext() {
                if(jButtonNext == null) {
                        jButtonNext = new JButton();
                        jButtonNext.setText("\u4e0b\u4e00\u6b65");
                        jButtonNext.setBounds(392, 367, 80, 30);
                        jButtonNext.setAction(getNextAction());
                }
                return jButtonNext;
        }
        private JButton getJButtonPre() {
                if(jButtonPre == null) {
                        jButtonPre = new JButton();
                        jButtonPre.setText("\u4e0a\u4e00\u6b65");
                        jButtonPre.setBounds(238, 367, 80, 30);
                }
                return jButtonPre;
        }
        private AbstractAction getNextAction() {
                if(nextAction == null) {
                        nextAction = new AbstractAction("\u4e0b\u4e00\u6b65", null) {
                                public void actionPerformed(ActionEvent evt) {
                                        setting.put("Operation", (String)operationComboBox.getSelectedItem());
                        setting.put("Test Strategy", (String)testStrategyJComboBox.getSelectedItem());
                        SwingUtilities.invokeLater(new Runnable() {
                            public void run() {
                                TestSettingFrame inst = new TestSettingFrame(setting, selectedOperationInfo);
                                inst.setLocationRelativeTo(null);
                                inst.setVisible(true);
                            }
                        });
                    }
                };
            }
            return nextAction;
        }
}
package ui;
import java.awt.Component;
import java.awt.GridLayout;
import java.awt.event.ActionEvent;
import java.awt.event.KeyAdapter;
import java.awt.event.KeyEvent;
import java.awt.event.MouseAdapter;
import java.awt.event.MouseEvent;
import java.io.File;
import javax.swing.AbstractAction;
import javax.swing.ButtonGroup;
import javax.swing.ComboBoxModel;
import javax.swing.DefaultComboBoxModel;
import javax.swing.JButton;
import javax.swing.JComboBox;
import javax.swing.JFileChooser;
import javax.swing.JLabel;
import javax.swing.JPanel;
import javax.swing.JRadioButton;
import javax.swing.JScrollPane;
import javax.swing.JTable;
import javax.swing.JTextField;
import javax.swing.WindowConstants;
import javax.swing.event.ChangeEvent;
import javax.swing.event.ChangeListener;
import javax.swing.SwingUtilities;
import javax.swing.table.DefaultTableModel;
import javax.swing.table.TableColumn;
import cn.edu.ustb.webservicetester.model.OperationInfo;
import cn.edu.ustb.webservicetester.model.ParameterInfo;
import cn.edu.ustb.webservicetester.model.TestCase;
import cn.edu.ustb.webservicetester.model.TestSet;
import cn.edu.ustb.webservicetester.util.TestSetFileParser;
import test.TestJFileChooseer.MyFileFilter;
```

```
public class TestSettingFrame extends javax.swing.JFrame {
	private JPanel jPanel1;
	private JButton nextPageJButton;
	private JRadioButton addTestCaseAutoRadioButton;
	private AbstractAction importTestCaseAction;
	private JButton addTestCaseAutoAdvancedSettingJButton;
	private JButton addTestCaseAutoBeginJButton;
	private JLabel addTestCaseAutoLabel2;
	private JTextField addTestCaseAutoTextField;
	private JLabel addTestCaseAutoLabel1;
	private JButton importTestCaseAddButton;
	private JButton importTestCaseScanButton;
	private JTextField importTestCaseTextField;
	private ButtonGroup buttonGroup1;
	private JRadioButton addTestCaseImportJRadioButton;
	private JLabel jLabel1;
	private JButton prePageButton;
	private JComboBox selectPageComboBox;
	private JScrollPane testSetDemoJScrollPane;
	private JTable testSetDemoJTable;
	private DefaultTableModel testSetDemoTableModel;
	private java.util.List<ParameterInfo> paraInfoList = null;
	private AbstractAction abstractAction1;
	private TestSet testSet = null;
	private OperationInfo operationInfo;
	private java.util.Map<String, String> testSetting = null;
	public static void main(String[] args) {
		SwingUtilities.invokeLater(new Runnable() {
			public void run() {
				TestSettingFrame inst = new TestSettingFrame();
				inst.setLocationRelativeTo(null);
				inst.setVisible(true);
			}
		});
	}
	public TestSettingFrame() {
		super();
		initGUI();
	}
	public TestSettingFrame(java.util.Map<String, String> testSettingMap, OperationInfo
operInfo) {
		super();
		testSetting = testSettingMap;
		initGUI(operInfo);
	}
	private void initGUI() {
		initGUI(null);
	}
	private void initGUI(OperationInfo oi) {
		operationInfo = oi;
```

```
        paraInfoList = operationInfo.getParaList();
        try {
            setTitle("测试配置");
            setResizable(false);
            GridLayout thisLayout = new GridLayout(1, 1);
            getContentPane().setLayout(thisLayout);
            {
                jPanel1 = new JPanel();
                getContentPane().add(jPanel1);
                jPanel1.setLayout(null);
                jPanel1.setBackground(new java.awt.Color(228,236,243));
                {
                    testSetDemoJScrollPane = new JScrollPane();
                    jPanel1.add(testSetDemoJScrollPane);
                    jPanel1.add(getNextPageJButton());
                    jPanel1.add(getSelectPageComboBox());
                    jPanel1.add(getPrePageButton());
                    jPanel1.add(getJLabel1());
                    jPanel1.add(getAddTestCaseImportJRadioButton());
                    jPanel1.add(getAddTestCaseAutoRadioButton());
                    jPanel1.add(getImpotyTestCaseTextField());
                    jPanel1.add(getImportTestCaseScanButton());
                    jPanel1.add(getImportTestCaseAddButton());
                    jPanel1.add(getAddTestCaseAutoLabel1());
                    jPanel1.add(getAddTestCaseAutoTextField());
                    jPanel1.add(getAddTestCaseAutoLabel2());
                    jPanel1.add(getAddTestCaseAutoBeginJButton());
                    jPanel1.add(getAddTestCaseAutoAdvancedSettingJButton());
                    getButtonGroup1();
                    addTestCaseImportJRadioButton.setSelected(true);
                    testSetDemoJScrollPane.setBounds(12, 12, 470, 183);
                    testSetDemoJScrollPane.setViewportView(getTestSetDemoJTable());
                }
            }
            setDefaultCloseOperation(WindowConstants.DISPOSE_ON_CLOSE);
            pack();
            setSize(500, 444);
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
    private JTable getTestSetDemoJTable() {
        if (testSetDemoJTable == null) {
            java.util.List<String> nl = new java.util.LinkedList<String>();
            nl.add("id");
            for (ParameterInfo pi : paraInfoList) {
                nl.add(pi.getName());
            }
            nl.add("expected result");
            String[] name = new String[nl.size()];
            nl.toArray(name);
            String[] a = {""};
            String[][] data = {a};
            testSetDemoTableModel = new DefaultTableModel(data, name);
            testSetDemoJTable = new JTable(testSetDemoTableModel) {
                public boolean isCellEditable(int row, int column) {
                    return false;
                }
            };
            int w = testSetDemoJTable.getTableHeader().getWidth();
            testSetDemoJTable.getTableHeader().setSize(w, 17);
            testSetDemoJTable.setRowHeight(17);
        }
        return testSetDemoJTable;
    }
    private JButton getNextPageJButton() {
        if(nextPageJButton == null) {
            nextPageJButton = new JButton();
            nextPageJButton.setText("\u4e0b\u4e00\u9875");
            nextPageJButton.setBounds(393, 211, 90, 24);
            nextPageJButton.setAction(getAbstractAction1());
        }
        return nextPageJButton;
    }
    private JComboBox getSelectPageComboBox() {
        if(selectPageComboBox == null) {
            ComboBoxModel selectPageComboBoxModel =
                    new DefaultComboBoxModel(
                            new String[] { "第 1 页", "第 2 页" });
            selectPageComboBox = new JComboBox();
            selectPageComboBox.setModel(selectPageComboBoxModel);
            selectPageComboBox.setBounds(279, 211, 88, 24);
            selectPageComboBox.setEditable(true);
        }
        return selectPageComboBox;
    }
    private JButton getPrePageButton() {
        if(prePageButton == null) {
            prePageButton = new JButton();
            prePageButton.setText("\u4e0a\u4e00\u9875");
            prePageButton.setBounds(164, 211, 90, 24);
        }
        return prePageButton;
    }
    private JLabel getJLabel1() {
        if(jLabel1 == null) {
            jLabel1 = new JLabel();
            jLabel1.setText("\u5171 0 \u6761\u6d4b\u8bd5\u7528\u4f8b");
            jLabel1.setBounds(12, 205, 134, 21);
            jLabel1.setForeground(new java.awt.Color(255,0,0));
            }
            return jLabel1;
        }
        private JRadioButton getAddTestCaseImportJRadioButton() {
            if(addTestCaseImportJRadioButton == null) {
                addTestCaseImportJRadioButton = new JRadioButton();
                addTestCaseImportJRadioButton.setText("\u5bfc\u5165\u6587\u4ef6");
                addTestCaseImportJRadioButton.setBounds(12, 275, 80, 21);
                addTestCaseImportJRadioButton.setBackground(new
java.awt.Color(252,252,203));
                addTestCaseImportJRadioButton.addChangeListener(new ChangeListener() {
                    public void stateChanged(ChangeEvent evt) {
                        addTestCaseImportJRadioButtonStateChanged(evt);
                    }
                });
            }
            return addTestCaseImportJRadioButton;
        }
        private JRadioButton getAddTestCaseAutoRadioButton() {
            if(addTestCaseAutoRadioButton == null) {
                addTestCaseAutoRadioButton = new JRadioButton();
                addTestCaseAutoRadioButton.setText("\u81ea\u52a8\u6dfb\u52a0");
                addTestCaseAutoRadioButton.setBounds(12, 345, 80, 21);
                addTestCaseAutoRadioButton.setBackground(new java.awt.Color(252,252,203));
            }
            return addTestCaseAutoRadioButton;
        }
        private ButtonGroup getButtonGroup1() {
            if(buttonGroup1 == null) {
                buttonGroup1 = new ButtonGroup();
            }
            buttonGroup1.add(addTestCaseImportJRadioButton);
            buttonGroup1.add(addTestCaseAutoRadioButton);
            return buttonGroup1;
        }
        private JTextField getImpotyTestCaseTextField() {
            if(importTestCaseTextField == null) {
                importTestCaseTextField = new JTextField();
                importTestCaseTextField.setBounds(97, 274, 206, 24);
            }
            return importTestCaseTextField;
        }
        private JButton getImportTestCaseScanButton() {
            if(importTestCaseScanButton == null) {
                importTestCaseScanButton = new JButton();
                importTestCaseScanButton.setText("\u6d4f\u89c8");
                importTestCaseScanButton.setBounds(303, 274, 54, 24);
                importTestCaseScanButton.setFont(new java.awt.Font("微软雅黑",0,10));
                final javax.swing.JFrame parentFrame = this;
                importTestCaseScanButton.addMouseListener(new MouseAdapter() {
                    public void mouseClicked(MouseEvent evt) {
                        JFileChooser jf = new JFileChooser();
                        jf.setApproveButtonText("导入");
                        jf.setCurrentDirectory(new File("."));
                        jf.setMultiSelectionEnabled(false);
                        jf.setFileFilter(new MyFileFilter("xml"));
                        jf.setDialogTitle("请选择要导入的测试集文件");
                        jf.showOpenDialog(parentFrame);
                        File f = jf.getSelectedFile();
                        if (f != null) {
                            importTestCaseTextField.setText(f.getPath());
                        }
                    }
                });
            }
            return importTestCaseScanButton;
        }
        private JButton getImportTestCaseAddButton() {
            if(importTestCaseAddButton == null) {
                importTestCaseAddButton = new JButton();
                importTestCaseAddButton.setText("\u5bfc\u5165");
                importTestCaseAddButton.setBounds(360, 274, 54, 24);
                importTestCaseAddButton.setFont(new java.awt.Font("微软雅黑",0,10));
                importTestCaseAddButton.setAction(getImportTestCaseAction());
            }
            return importTestCaseAddButton;
        }
        private JLabel getAddTestCaseAutoLabel1() {
            if(addTestCaseAutoLabel1 == null) {
                addTestCaseAutoLabel1 = new JLabel();
                addTestCaseAutoLabel1.setText("\u6dfb\u52a0");
                addTestCaseAutoLabel1.setBounds(97, 347, 30, 17);
            }
            return addTestCaseAutoLabel1;
        }
        private JTextField getAddTestCaseAutoTextField() {
            if(addTestCaseAutoTextField == null) {
                addTestCaseAutoTextField = new JTextField();
                addTestCaseAutoTextField.setBounds(133, 344, 56, 24);
        addTestCaseAutoTextField.setToolTipText("\u81ea\u52a8\u751f\u6210\u6d4b\u8bd5\u7528\
u4f8b\u7684\u6570\u91cf");
                addTestCaseAutoTextField.setHorizontalAlignment(JTextField.RIGHT);
                addTestCaseAutoTextField.addKeyListener(new KeyAdapter() {
                    public void keyTyped(KeyEvent evt) {
                        char kc = evt.getKeyChar();
                        if (addTestCaseAutoTextField.getText().length() == 0 && kc >=
KeyEvent.VK_1 && kc <=  KeyEvent.VK_9) {
                            ;
                        } else if (addTestCaseAutoTextField.getText().length() > 0 &&
addTestCaseAutoTextField.getText().length() < 6 && kc >= KeyEvent.VK_0 && kc <=
KeyEvent.VK_9) {
					} else {
						evt.consume();
					}
				}
			});
			addTestCaseAutoTextField.setEnabled(false);
		}
		return addTestCaseAutoTextField;
	}
	private JLabel getAddTestCaseAutoLabel2() {
		if(addTestCaseAutoLabel2 == null) {
			addTestCaseAutoLabel2 = new JLabel();
			addTestCaseAutoLabel2.setText("\u6761\u6d4b\u8bd5\u7528\u4f8b");
			addTestCaseAutoLabel2.setBounds(197, 347, 70, 17);
		}
		return addTestCaseAutoLabel2;
	}
	private JButton getAddTestCaseAutoBeginJButton() {
		if(addTestCaseAutoBeginJButton == null) {
			addTestCaseAutoBeginJButton = new JButton();
			addTestCaseAutoBeginJButton.setText("\u5f00\u59cb");
			addTestCaseAutoBeginJButton.setBounds(290, 345, 90, 24);
			addTestCaseAutoBeginJButton.setEnabled(false);
		}
		return addTestCaseAutoBeginJButton;
	}
	private JButton getAddTestCaseAutoAdvancedSettingJButton() {
		if(addTestCaseAutoAdvancedSettingJButton == null) {
			addTestCaseAutoAdvancedSettingJButton = new JButton();
			addTestCaseAutoAdvancedSettingJButton.setText("\u9ad8\u7ea7\u8bbe\u7f6e");
			addTestCaseAutoAdvancedSettingJButton.setBounds(393, 345, 90, 24);
			addTestCaseAutoAdvancedSettingJButton.setFont(new java.awt.Font("微软雅黑
",0,10));
			addTestCaseAutoAdvancedSettingJButton.setEnabled(false);
		}
		return addTestCaseAutoAdvancedSettingJButton;
	}
	private AbstractAction getImportTestCaseAction() {
		if(importTestCaseAction == null) {
			importTestCaseAction = new AbstractAction("\u5bfc\u5165", null) {
				public void actionPerformed(ActionEvent evt) {
					File f = new File(importTestCaseTextField.getText());
					TestSetFileParser tsfp = new TestSetFileParser();
					tsfp.setParaInfoList(paraInfoList);
					testSet = tsfp.parseFile(importTestCaseTextField.getText());
					for (int i = 0; i < testSet.numOfTestCases(); ++i) {
						TestCase tc = testSet.get(i);
						java.util.List<String>		pValueList		=		new
java.util.ArrayList<String>(paraInfoList.size() + 2);
```

```
                        pValueList.add(new Integer(tc.getId()).toString());
                        for (ParameterInfo pi : paraInfoList) {
                            pValueList.add(tc.getValueOfParameter(pi));
                        }
                        pValueList.add(tc.getExpectedResult());
                        String[] pValueArray = new String[pValueList.size()];
                        pValueList.toArray(pValueArray);
                        for (String pv : pValueArray) {
                            System.out.println(pv);
                        }
                        testSetDemoTableModel.addRow(pValueArray);
                    }
                }
            };
        }
        return importTestCaseAction;
    }
    private AbstractAction getAbstractAction1() {
        if(abstractAction1 == null) {
            abstractAction1 = new AbstractAction("\u4e0b\u4e00\u9875", null) {
                public void actionPerformed(ActionEvent evt) {
                    if (testSet != null) {
                        String testStrategy = testSetting.get("Test Strategy");
                        testSetting.put("Test Set", importTestCaseTextField.getText());
                        System.out.println(testStrategy);
                        if ("Random Testing".equals(testStrategy)) {
                            ExecutionFrame ef = new ExecutionFrame(testSetting, operationInfo, testSet);
                            ef.setLocationRelativeTo(null);
                            ef.setVisible(true);
                        } else if ("Dynamic Random Testing".equals(testStrategy)){
                            System.out.println("??");
                            PartitionSettingFrame2 psf = new PartitionSettingFrame2(testSetting, operationInfo, testSet);
                            psf.setLocationRelativeTo(null);
                            psf.setVisible(true);
                        } else {
                            System.out.println("?");
                        }
                    } else {
                    }
                }
            };
        }
        return abstractAction1;
    }
    private void addTestCaseImportJRadioButtonStateChanged(ChangeEvent evt) {
        System.out.println("addTestCaseImportJRadioButton.stateChanged, event="+evt);
        if (addTestCaseImportJRadioButton.isSelected()) {
            importTestCaseTextField.setEnabled(true);
```

```
                importTestCaseScanButton.setEnabled(true);
                importTestCaseAddButton.setEnabled(true);
                addTestCaseAutoTextField.setEnabled(false);
                addTestCaseAutoBeginJButton.setEnabled(false);
                addTestCaseAutoAdvancedSettingJButton.setEnabled(false);
            } else {
                importTestCaseTextField.setEnabled(false);
                importTestCaseScanButton.setEnabled(false);
                importTestCaseAddButton.setEnabled(false);
                addTestCaseAutoTextField.setEnabled(true);
                addTestCaseAutoBeginJButton.setEnabled(true);
                addTestCaseAutoAdvancedSettingJButton.setEnabled(true);
            }
        }
}
package test;
import java.io.File;
import javax.swing.JFileChooser;
import javax.swing.filechooser.FileFilter;
public class TestJFileChooseer {
        public static void main(String[] args) {
            JFileChooser jf = new JFileChooser();
            jf.setApproveButtonText("导入");
            jf.setCurrentDirectory(new File("."));
            jf.setMultiSelectionEnabled(false);
            jf.setFileFilter(new MyFileFilter("xml"));
            jf.setDialogTitle("请选择要导入的测试集文件");
            jf.showOpenDialog(null);
            File f = jf.getSelectedFile();
            if (f != null)
                System.out.println(f.getPath());
        }
        public static class MyFileFilter extends FileFilter {
            private String ext;
            public MyFileFilter(String extString) {
                ext = extString;
            }
            public boolean accept(File f) {
                if (f.isDirectory()) {
                    return true;
                }
                String extension = getExtension(f);
                if (extension.toLowerCase().equals(this.ext.toLowerCase()))
                {
                    return true;
                }
                return false;
            }
              public String getDescription() {
                  return this.ext.toUpperCase();
              }
            private String getExtension(File f) {
                String name = f.getName();
                int index = name.lastIndexOf('.');
                if (index == -1)
                {
                    return "";
                }
                else
                {
                    return name.substring(index + 1).toLowerCase();
                }
            }
        }
}
package test;
import java.awt.BorderLayout;
import javax.swing.DefaultCellEditor;
import javax.swing.JPanel;
import javax.swing.JScrollPane;
import javax.swing.JTable;
import javax.swing.JTextField;
import javax.swing.WindowConstants;
import javax.swing.SwingUtilities;
public class TestJTableFrame extends javax.swing.JFrame {
        private JPanel jPanel1;
        private JScrollPane jScrollPane1;
        private JTable jTable1;
        public static void main(String[] args) {
            SwingUtilities.invokeLater(new Runnable() {
                public void run() {
                    TestJTableFrame inst = new TestJTableFrame();
                    inst.setLocationRelativeTo(null);
                    inst.setVisible(true);
                }
            });
        }
        public TestJTableFrame() {
            super();
            initGUI();
        }
        private void initGUI() {
            try {
                setDefaultCloseOperation(WindowConstants.DISPOSE_ON_CLOSE);
                {
                    jPanel1 = new JPanel();
                    jPanel1.setLayout(null);
                    getContentPane().add(jPanel1, BorderLayout.CENTER);
                    {
                        jScrollPane1 = new JScrollPane();
                        jPanel1.add(jScrollPane1);
                        jScrollPane1.setBounds(29, 36, 195, 136);
                        jScrollPane1.setViewportView(getJTable1());
                    }
                }
                pack();
                setSize(400, 300);
                jTable1.editCellAt(0, 0);
                jTable1.getEditorComponent().requestFocusInWindow();
            } catch (Exception e) {
                e.printStackTrace();
            }
        }
        private JTable getJTable1() {
            if (jTable1 == null) {
                String[] s1 = new String[] {"小王", "0001"};
                String[] s2 = new String[] {"小芳", "0002"};
                String[][] data = new String[][] {s1, s2};
                String[] name = new String[] {"姓名", "工号"};
                jTable1 = new JTable(data, name);
            }
            return jTable1;
        }
}
package test;
import java.io.File;
import org.dom4j.Document;
import org.dom4j.DocumentException;
import org.dom4j.Element;
public class XmlTest {
        public static void main(String[] args) {
            File f = new File("E:\\研究生\\毕业设计\\DRT 算法改进\\实验\\停车计费\\测试用例
集\\随机 1000 规模\\testcaseset4RT.xml");
            org.dom4j.io.SAXReader saxReader = new org.dom4j.io.SAXReader();
            Document document = null;
            Element rootElement = null;
            try {
                document = saxReader.read(f);
                rootElement = document.getRootElement();
                java.util.List<Element> testCaseList = rootElement.selectNodes("testCase");
                System.out.println(testCaseList.size());
            } catch (DocumentException e) {
                e.printStackTrace();
            }
        }
}
package test;
import cn.edu.ustb.mt4ws.tcg.WsdlOperationFormat;
import cn.edu.ustb.mt4ws.wsdl.parser.WsdlReader11;
import javax.wsdl.*;
import javax.wsdl.extensions.*;
import javax.wsdl.factory.*;
import javax.wsdl.xml.*;
import javax.xml.namespace.QName;
import java.io.*;
import java.util.*;
import org.dom4j.DocumentException;
import org.dom4j.io.SAXReader;
import org.w3c.dom.*;
public class ZzqTest {
    public static void main(String[] args) {
        String wsUrl = "http://localhost:8080/axis2/services/ATMService?wsdl";
        File f = new File("temp.xml");
        if (!f.exists()) {
        try {
            WSDLFactory factory = WSDLFactory.newInstance();
            WSDLReader reader = factory.newWSDLReader();
            reader.setFeature("javax.wsdl.verbose", true);
            reader.setFeature("javax.wsdl.importDocuments", true);
            Definition def = reader.readWSDL(wsUrl);
            Writer xmlWriter = new BufferedWriter(new OutputStreamWriter(new
FileOutputStream("temp.xml")));
            WSDLWriter wsdlWriter = factory.newWSDLWriter();
            wsdlWriter.writeWSDL(def, xmlWriter);
        } catch (FileNotFoundException e) {
            e.printStackTrace();
        } catch (WSDLException e) {
            e.printStackTrace();
        }
        }
        SAXReader dReader = new SAXReader();
        org.dom4j.Document document;
        try {
            document = dReader.read(new File("temp.xml"));
            org.dom4j.Element root = document.getRootElement();
            org.dom4j.Element typesElement = root.element("types");
            org.dom4j.Element schemaElement = typesElement.element("schema");
            @SuppressWarnings("unchecked")
            List<org.dom4j.Element>                elist                =
(List<org.dom4j.Element>)schemaElement.elements();
            for (org.dom4j.Element e : elist) {
                if (e.attribute("name").getValue().equals("accountNumber")) {
                    org.dom4j.Element rElement = e.element("restriction");
                    String nativeType = rElement.attribute("base").getValue();
                    System.out.println(nativeType);
                }
            }
        } catch (DocumentException e) {
            e.printStackTrace();
        }
```

```
	}
}
package test.thread;
import java.util.*;
import java.util.concurrent.TimeUnit;
public class ThreadTestMain {
	static class WriteString implements Runnable {
		private List<String> sl;
		public WriteString(List<String> ssl) {
			sl = ssl;
		}
		public void run() {
			int i = 0;
			String s;
			while (i++ < 20) {
				s = new Integer(i).toString();
				sl.add(s);
				try {
					TimeUnit.MILLISECONDS.sleep(new Random().nextInt(1500));
				} catch (InterruptedException e) {
					e.printStackTrace();
					System.out.println("size:" + sl.size());
					System.out.print(i + ": ");
					System.out.println(sl.get(i));

				}
			}
		}
	}
	static class ReadString implements Runnable {
		List<String> sl;
		public ReadString(List<String> ssl) {
			sl = ssl;
		}
		public void run() {
			int i = 0;
			while (i < 20) {
				if (i < sl.size()) {
					System.out.println("size:" + sl.size());
					System.out.print(i + ": ");
					System.out.println(sl.get(i));
					++i;
				} else {
					try {
						TimeUnit.MILLISECONDS.sleep(2000);
					} catch (InterruptedException e) {
						e.printStackTrace();
					}
				}
			}
```

```
        }
    }
    public static void main(String[] args) {
        List<String> sl = new ArrayList<String>();
        Thread th1 = new Thread(new WriteString(sl));
        Thread th2 = new Thread(new ReadString(sl));
        th1.start();
        th2.start();
    }
}
package cn.edu.ustb.webservicetester.controllor;
import javax.swing.SwingUtilities;
import ui.OperationSelectFrame;
import ui.OperationSelectionFrame2;
import ui.PreferenceFrame;
import cn.edu.ustb.webservicetester.util.WsdlParser;
public class ParseWsdlControllor {
    public void parseWsdlControl(String wsdlUrl) {
        WsdlParser parser = new WsdlParser();
        final java.util.List opInfoList = parser.parseWsdl(wsdlUrl);
        final java.util.Map<String, String> setting = new java.util.HashMap<String, String>();
        setting.put("Web Service", wsdlUrl.substring(0, wsdlUrl.lastIndexOf("?wsdl")));
        SwingUtilities.invokeLater(new Runnable() {
            public void run() {
                PreferenceFrame inst = new PreferenceFrame(setting, opInfoList);
                inst.setLocationRelativeTo(null);
                inst.setVisible(true);
            }
        });
    }
}
package cn.edu.ustb.webservicetester.model;
public class ParameterIntervalScenario {
    private String range;// 区间范围
    private int weight;// 权重
    public String getRange() {
        return range;
    }
    public void setRange(String range) {
        this.range = range;
    }
    public int getWeight() {
        return weight;
    }
    public void setWeight(int weight) {
        this.weight = weight;
    }
}
package cn.edu.ustb.webservicetester.model;
public class ParameterPartitionScenario {
	private ParameterInfo paraInfo;// 参数信息
	private java.util.List<ParameterIntervalScenario> partitionScenario;// 分区方案
	public ParameterPartitionScenario() {
		partitionScenario = new java.util.LinkedList<ParameterIntervalScenario>();
	}
	public ParameterInfo getParaInfo() {
		return paraInfo;
	}
	public void setParaInfo(ParameterInfo paraInfo) {
		this.paraInfo = paraInfo;
	}
	public ParameterIntervalScenario get(int index) {
		return partitionScenario.get(index);
	}
	public void add(ParameterIntervalScenario paraInter) {
		partitionScenario.add(paraInter);
	}
	public int numOfIntervals() {
		return partitionScenario.size();
	}
}
package cn.edu.ustb.webservicetester.model;
import java.util.Map.Entry;
public class Partition extends TestSet {
	private java.util.Map<ParameterInfo, String> rules;
	public Partition() {
		super();
		rules = new java.util.HashMap<ParameterInfo, String>();
	}
	public Partition(java.util.Map<ParameterInfo, String> rs) {
		super();
		rules = rs;
	}
	public void addRule(ParameterInfo pi, String rStr) {
		rules.put(pi, rStr);
	}
	public String getRule(ParameterInfo pi) {
		if (!rules.containsKey(pi))
			return null;
		return rules.get(pi);
	}
	public boolean couldAdd(TestCase tc) {
		java.util.List<ParameterInfo> paraInfoList = tc.getParameters();
		boolean ca = true;
		for (int i = 0; i < paraInfoList.size() && ca; ++i) {
			ParameterInfo pi = paraInfoList.get(i);
			if (pi.getType() == ParameterInfo.WHATEVER) {
				continue;
			}
			if (!rules.containsKey(pi)) {
```

```
                ca = false;
                continue;
            }
            String ci = pi.getClassInfo();
            if (pi.getType() == ParameterInfo.SCATTERED) {
                ca = tc.getValueOfParameter(pi).equals(rules.get(pi));
            } else{
                String rule = rules.get(pi);
                String ss[] = rule.split(",");
                String s1 = ss[0].substring(1).trim();
                String s2 = ss[1].substring(0, ss[1].length() - 1).trim();
                String s = tc.getValueOfParameter(pi);
                if ("byte".equals(ci)
                        || "short".equals(ci)
                        || "int".equals(ci)
                        || "long".equals(ci)) {
                    long l1 = new Long(s1);
                    long l2 = new Long(s2);
                    long l = new Long(s);
                    if (l < l2 && l > l1) {
                        ca = true;
                    } else if (rule.startsWith("[") && l == l1) {
                        ca = true;
                    } else if (rule.endsWith("]") && l == l2) {
                        ca = true;
                    } else {
                        ca = false;
                    }
                } else if ("float".equals(ci) || "double".equals(ci)) {
                    double d1 = new Double(s1);
                    double d2 = new Double(s2);
                    System.out.println(pi.getName());
                    System.out.println(s);
                    System.out.println(tc.containsParameter(pi));
                    System.out.println(i);
                    rsb.append(ent.getKey());
                    rsb.append(": ");
                    rsb.append(ent.getValue());
                    rsb.append("\n");
                    double d = new Double(s);
                    if (d < d2 && d > d1) {
                        ca = true;
                    } else if (rule.startsWith("[") && Math.abs(d - d1) < 1e-5) {
                        ca = true;
                    } else if (rule.endsWith("]") && Math.abs(d2 - d) < 1e-5) {
                        ca = true;
                    } else {
                        ca = false;
                    }
                } else if ("char".equals(ci)
                        && s1.length() == 1
                        && s2.length() == 1
                        && s.length() == 1) {
                    char c1 = s1.charAt(0);
                    char c2 = s2.charAt(0);
                    char c = s.charAt(0);
                    if (c < c2 && c > c1) {
                        ca = true;
                    } else if (rule.startsWith("[") && c == c1) {
                        ca = true;
                    } else if (rule.endsWith("]") && c == c2) {
                        ca = true;
                    } else {
                        ca = false;
                    }
                } else {
                }
            }
        }
        return ca;
    }
    public String toString() {
        StringBuffer rsb = new StringBuffer();
        for (Entry<ParameterInfo, String> ent : rules.entrySet()) {
            rsb.append(ent.getKey());
            rsb.append(": ");
            rsb.append(ent.getValue());
            rsb.append("\n");
        }
        rsb.append("[\n");
        java.util.Iterator<TestCase> tcIter = testCases.iterator();
        while (tcIter.hasNext()) {
            TestCase tc = tcIter.next();
            rsb.append(tc);
            rsb.append("\n");
            sb.append(p.getName());
            sb.append(" : ");
            sb.append(paraValues.get(p));
        }
        rsb.append("]");
        return rsb.toString();
    }
}
package cn.edu.ustb.webservicetester.model;
public class TestCase {
    private int id = -1;
    private java.util.Map<ParameterInfo, String> paraValues;
    private String expectedResult = null;
    public TestCase() {
        paraValues = new java.util.HashMap<ParameterInfo, String>();
```

```
	}
	public boolean containsParameter(ParameterInfo pi) {
		return paraValues.containsKey(pi);
	}
	public int getId() {
		return id;
	}
	public void setId(int id) {
		this.id = id;
	}
	public void setValueOfParameter(ParameterInfo pi, String value) {
		paraValues.put(pi, value);
	}
	public String getValueOfParameter(ParameterInfo pi) {
		return paraValues.get(pi);
	}
	public String getExpectedResult() {
		return expectedResult;
	}
	public void setExpectedResult(String expectedResult) {
		this.expectedResult = expectedResult;
	}
	public java.util.List<ParameterInfo> getParameters() {
		java.util.List<ParameterInfo>		paraInfoList		=		new
java.util.LinkedList<ParameterInfo>();
		java.util.Iterator<ParameterInfo> pii = paraValues.keySet().iterator();
		while (pii.hasNext()) {
			paraInfoList.add(pii.next());
			sb.append(p.getName());
			sb.append(" : ");
			sb.append(paraValues.get(p));
			sb.append(p.getName());
			sb.append(" : ");
			sb.append(paraValues.get(p));
			sb.append(", ");

		}
		return paraInfoList;
	}
	public boolean equals(Object o) {
		if (!(o instanceof TestCase)) {
			return false;
		}
		TestCase tc = (TestCase)o;
		if (!((tc.getParameters()).equals(getParameters()))) {
			return false;
		}
		java.util.Iterator<ParameterInfo> paraInfoIter = paraValues.keySet().iterator();
		boolean findDifference = false;
		while (paraInfoIter.hasNext() && !findDifference) {
```

```
                ParameterInfo pi = paraInfoIter.next();
                findDifference = !(paraValues.get(pi).equals(tc.getValueOfParameter(pi)));
                sb.append(p.getName());
                sb.append(" : ");
                sb.append(paraValues.get(p));
                sb.append(", ");
            }
            return !findDifference;
        }
        public int hashCode() {
            int result = 17;
            result = result * 37 + paraValues.hashCode();
            if (expectedResult != null)
                result = result * 37 + expectedResult.hashCode();
            return result;
        }
        public String toString() {
            StringBuffer sb = new StringBuffer();
            for (ParameterInfo p : getParameters()) {
                sb.append(p.getName());
                sb.append(" : ");
                sb.append(paraValues.get(p));
                sb.append(", ");
            }
            if (expectedResult != null) {
                sb.append("expected result : ");
                sb.append(expectedResult);
            } else {
                sb.delete(sb.length() - 2, sb.length());
            }
            return sb.toString();
        }
    }
    package cn.edu.ustb.webservicetester.model;
    public class TestSet {
        protected java.util.List<TestCase> testCases;
        public TestSet() {
            testCases = new java.util.LinkedList<TestCase>();
        }
        public void add(TestCase tc) {
            testCases.add(tc);
        }
        public TestCase get(int i) {
            return testCases.get(i);
        }
        public boolean contains(TestCase tc) {
            boolean result = false;
            java.util.Iterator<TestCase> tsIter = testCases.iterator();
            while (tsIter.hasNext() && !result) {
                result = tc.equals(tsIter.next());
```

```
		}
		return false;
	}
	public int numOfTestCases() {
		return testCases.size();
	}
}
package cn.edu.ustb.webservicetester.model;
public class TestSetWithPartition {
	private Partition[] partitions;
	private double[] profile;
	public TestSetWithPartition() {
		partitions = null;
		profile = null;
	}
	public TestSetWithPartition(int numOfPartitions) {
		partitions = new Partition[numOfPartitions];
		setProfileEvenly();
	}
	public TestSetWithPartition(Partition[] ps) {
		setPartitions(ps);
		setProfileEvenly();
	}
	public TestSetWithPartition(Partition[] ps, double[] prf) {
		setPartitions(ps);
		setProfile(prf);
	}
	public void setPartitions(Partition[] ps) {
		if (ps != null && ps.length > 0) {
			int i = 0;
			for (Partition p : ps) {
				if (p.numOfTestCases() > 0) {
					++i;
				}
			}
			partitions = new Partition[i];
			i = 0;
			for (int j = 0; j < ps.length; ++j) {
				if (ps[j].numOfTestCases() > 0) {
					partitions[i] = ps[j];
					++i;
				}
			}
		}
	}
	public void setProfile(double[] prf) {
		if (prf != null && prf.length > 0) {
			profile = new double[prf.length];
			for (int i = 0; i < prf.length; ++i) {
				profile[i] = prf[i];
				}
			}
		}
		public void setProfileEvenly() {
			if (partitions != null && partitions.length > 0) {
				profile = new double[partitions.length];
				for (int i = 0; i < profile.length; ++i) {
					profile[i] = 1.0 / profile.length;
				}
			}
		}
		public int numOfPartitions() {
			if (partitions == null)
				return 0;
			return partitions.length;
		}
		public int numOfTestCases() {
			int n = 0;
			if (partitions != null && partitions.length > 0) {
				for (Partition p : partitions) {
					n += p.numOfTestCases();
				}
			}
			return n;
		}
		public Partition[] getPartitions() {
			return partitions;
		}
		public Partition getPartition(int i) {
			return partitions[i];
		}
		public TestCase getTestCase(int pIndex, int tcIndex) {
			Partition p = partitions[pIndex];
			return p.get(tcIndex);
		}
		public double[] getProfile() {
			return profile;
		}
	}
	package cn.edu.ustb.webservicetester.model;
	public class Interval {
		private java.util.Map<ParameterInfo, ParameterIntervalScenario> range;
		private int weight;
		private java.util.List<TestCase> testCases;
		public Interval(java.util.Map<ParameterInfo, ParameterIntervalScenario> r, int w) {
			this.range = r;
			testCases = new java.util.LinkedList<TestCase>();
		}
		public int getWeight() {
			return weight;
	}
	public int numOfTestCases() {
		return testCases.size();
	}
	public void addTestCase(TestCase tc) {
		testCases.add(tc);
	}
	public TestCase getTestCase(int i) {
		return testCases.get(i);
	}
	public boolean includes(TestCase tc) {
		for (ParameterInfo pi : tc.getParameters()) {
			if (pi.getType() == ParameterInfo.SCATTERED) {
				if (excludeScatteredValue(tc, pi))
					return false;
			} else if (pi.getType() == ParameterInfo.CONTINUOUS){
				if (excludesContinuousValue(tc, pi))
					return false;
			}
		}
		return true;
	}
	private boolean excludesContinuousValue(TestCase tc, ParameterInfo pi) {
		String rangeOfParaInterval = range.get(pi).getRange();
		String valueOfTestCase = tc.getValueOfParameter(pi);
		double upperBound;
		double lowerBound;
		String[] ss = rangeOfParaInterval.split(", ");
		lowerBound = Double.valueOf(ss[0].substring(1));
		upperBound = Double.valueOf(ss[1].substring(0, ss[1].length() - 1));
		double votc = Double.valueOf(valueOfTestCase);
		if (rangeOfParaInterval.startsWith("[") && votc < lowerBound)
			return true;
		if (rangeOfParaInterval.startsWith("(") && votc <= lowerBound)
			return true;
		if (rangeOfParaInterval.endsWith("]") && votc > upperBound)
			return true;
		if (rangeOfParaInterval.endsWith(")") && votc >= upperBound)
			return true;
		return false;
	}
	private boolean excludeScatteredValue(TestCase tc, ParameterInfo pi) {
		return !tc.getValueOfParameter(pi).equals(range.get(pi.getName()).getRange());
	}
}
package cn.edu.ustb.webservicetester.model;
import java.util.List;
public class OperationInfo {
	private String opName;
	private List<ParameterInfo> paraList;
	public OperationInfo() {
	}
	public String getOpName() {
		return opName;
	}
	public void setOpName(String opName) {
		this.opName = opName;
	}
	public List<ParameterInfo> getParaList() {
		return paraList;
	}
	public void setParaList(List<ParameterInfo> paraList) {
		this.paraList = paraList;
	}
}
package cn.edu.ustb.webservicetester.model;
public class ParameterInfo {
	public static final int WHATEVER = 0;
	public static final int CONTINUOUS = 1;
	public static final int SCATTERED = 2;
	private String name;
	private String classInfo;
	private int type;
	private String restriction;
	public ParameterInfo() {
		type = WHATEVER;
	}
	public String getName() {
		return name;
	}
	public void setName(String id) {
		this.name = id;
	}
	public String getClassInfo() {
		return classInfo;
	}
	public void setClassInfo(String classInfo) {
		this.classInfo = classInfo;
	}
	public int getType() {
		return type;
	}
	public void setType(int type) {
		this.type = type;
	}
}
package cn.edu.ustb.mt4ws.configuration;
import java.util.Map;
import cn.edu.ustb.mt4ws.tcg.TestSuite;
public class Project {
	private Workspace workspace;
	private String projectName;
	private String wsdlUrl;
	protected Map<String, TestSuite> testSuites;
	public String getProjectName() {
		return projectName;
	}
	public String getWsdlUrl() {
		return wsdlUrl;
	}
	public void setWorkspace(Workspace workspace) {
		if (workspace == null){
			this.workspace = workspace;
		}else{
		this.workspace = workspace;
		}
	}
	public Workspace getWorkspace() {
		return workspace;
	}
	public Map<String, TestSuite> getTestSuiteMap() {
		return testSuites;
	}
	public void setProjectName(String projectName) {
		this.projectName = projectName;
	}
	public void setWSDLUri(String wsdlUri) {
		this.wsdlUrl = wsdlUri;
	}
	public void setTestSuites(Map<String, TestSuite> testsuites) {
		this.testSuites = testsuites;
	}
	public void print() {
		System.out.println("Project:" + projectName);
		System.out.println("WSDL URL=" + wsdlUrl);
		System.out.println("workspace=" + workspace.getPath());
	}
}
package cn.edu.ustb.mt4ws.configuration;
import java.util.List;
public class Configuration {
	private String operationName;
	private List<Integer> mrIds;
	private List<List<Integer>> testCases;
	public Configuration(){
	}
	public Configuration(String operationName, List<Integer> mrIds,
			List<List<Integer>> testCases) {
		this.operationName = operationName;
		this.mrIds = mrIds;
```

```
            this.testCases = testCases;
        }
        public void setOperationName(String operationName) {
            this.operationName = operationName;
        }
        public String getOperationName() {
            return operationName;
        }
        public void setMrIds(List<Integer> mrIds) {
            this.mrIds = mrIds;
        }
        public List<Integer> getMrIds() {
            return mrIds;
        }
        public void setTestCases(List<List<Integer>> testCases) {
            this.testCases = testCases;
        }
        public List<List<Integer>> getTestCases() {
            return testCases;
        }
}
package cn.edu.ustb.mt4ws.file;
import java.io.File;
import java.io.IOException;
import javax.xml.transform.Source;
import javax.xml.transform.stream.StreamSource;
import javax.xml.validation.Schema;
import javax.xml.validation.SchemaFactory;
import javax.xml.validation.Validator;
import org.xml.sax.SAXException;
public class ValidatorXML {
        public ValidatorXML(){
        }
        public boolean validateXml(String xsdpath, String xmlpath)
                    throws SAXException, IOException {
            SchemaFactory schemaFactory = SchemaFactory
                        .newInstance("http://www.w3.org/2001/XMLSchema");
            File schemaFile = new File(xsdpath);
            Schema schema = schemaFactory.newSchema(schemaFile);
            Validator validator = schema.newValidator();
            Source source = new StreamSource(xmlpath);
            try {
                validator.validate(source);
            } catch (Exception ex) {
                ex.printStackTrace();
                return false;
            }
            return true;
        }
}
```