ORIGIN
Product Lifecycle Management
Consulting Service

Teamcenter客户化开发（二）

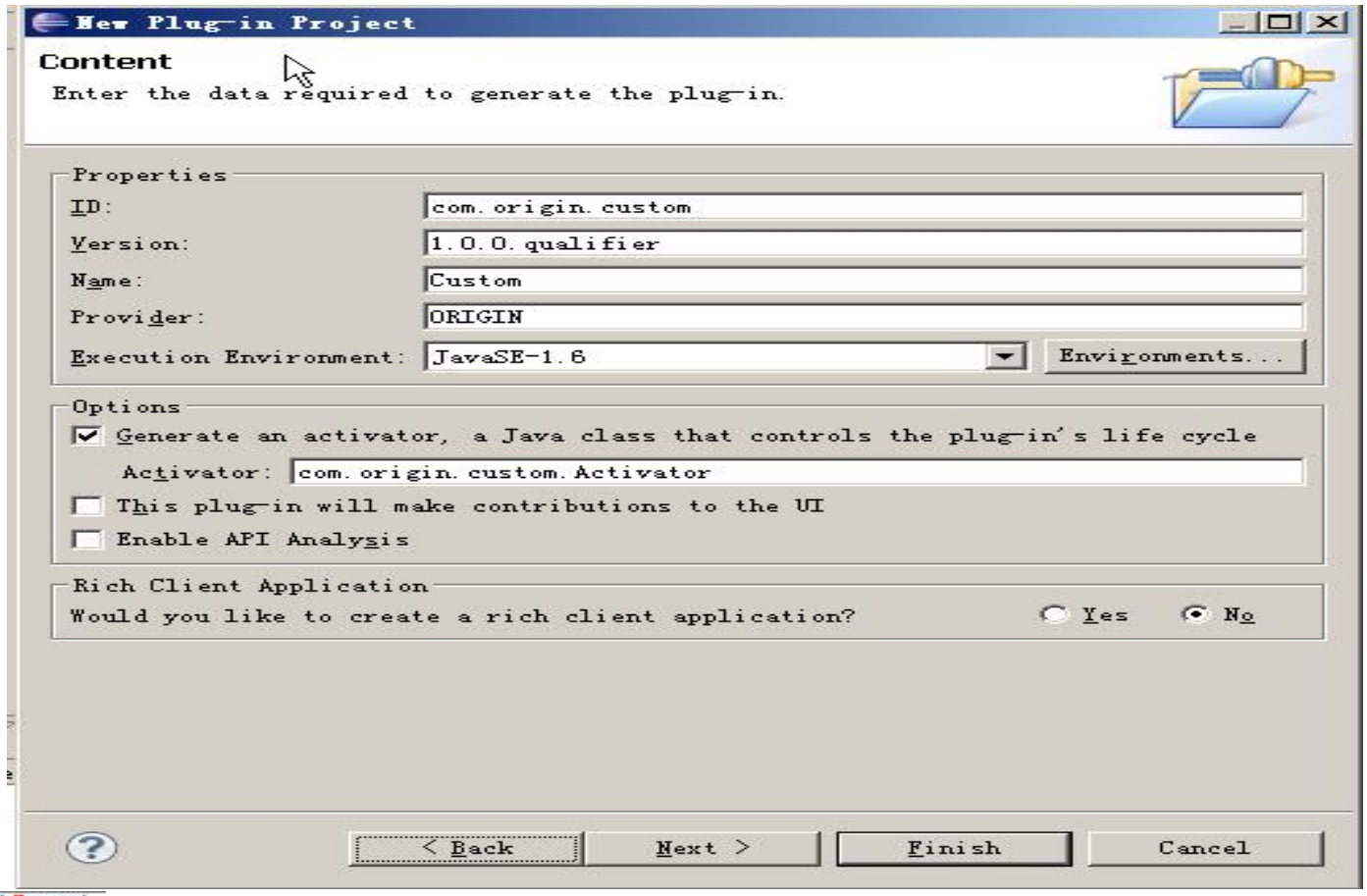
TC中对话框讲解，主要针对实例，在系统中添加新建Folder操作

综述，该章节主要是通过怎么在系统中客户化一个创建Folder对象的操作。并分别把该操作添加到菜单栏，工具栏及右键菜单，对前一章节的知识进行全面的实践。接下来，就从建立一个完整的客户化项目逐步进行讲解。

TC中对话框讲解，主要针对实例，在系统中添加新建Folder操作

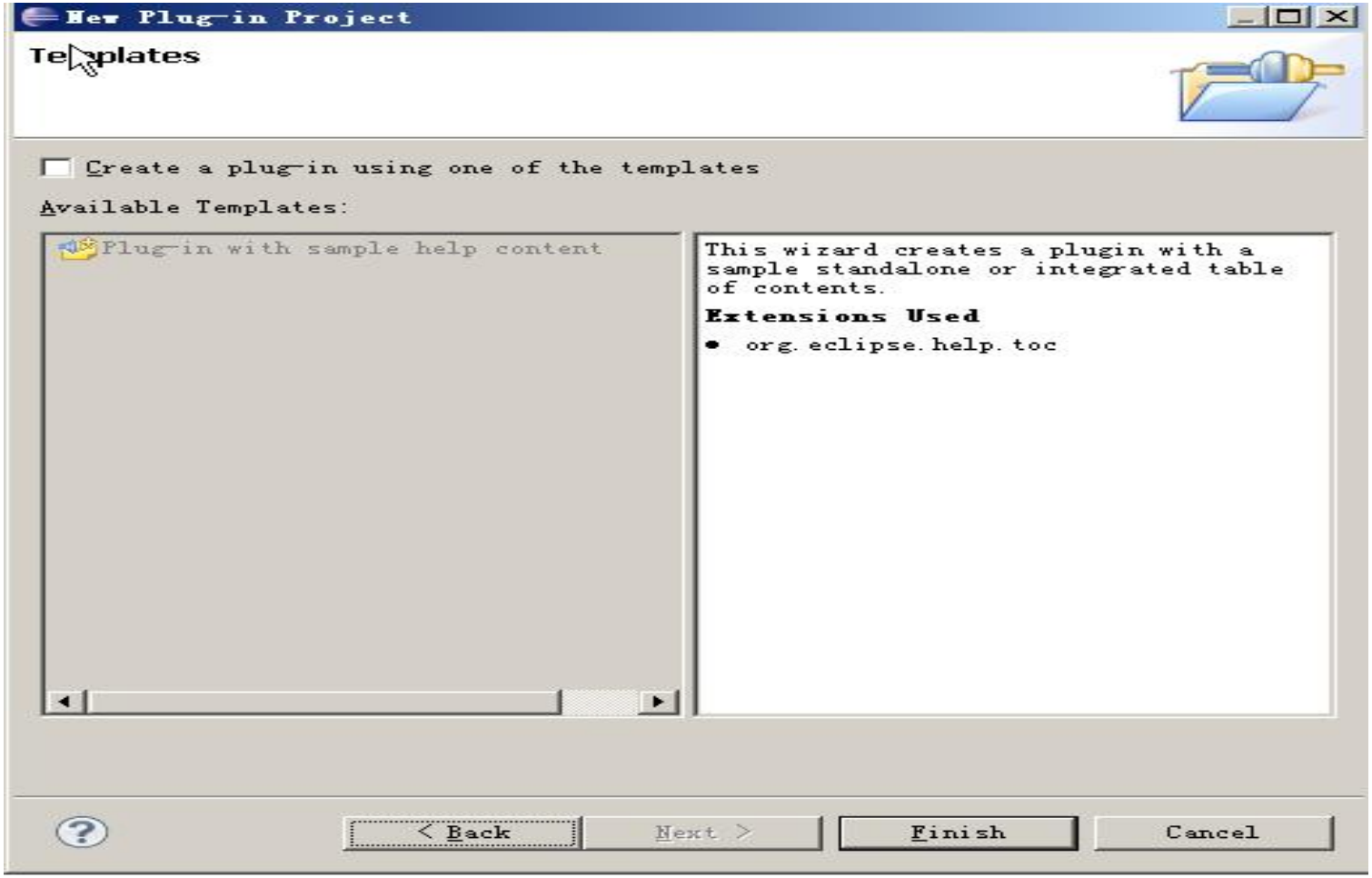
1.创建java插件工程com.origin.custom.

2.在New Plug-in Project 对话框 Content 面板, 取消This plug-in will make contributions to the UI的选择，然后Next



TC中对话框讲解，主要针对实例，在系统中添加新建Folder操作

3. 在Create a plug-in using one of these templates面板，确保Create a plug-in using one of these templates没有被选择。



4.在工程中新建plugin.xml文件。内容如下：

```
<?xml version="1.0"
encoding="UTF-8"?>
<?eclipse version="3.4"?>
<plugin>
</plugin>
```

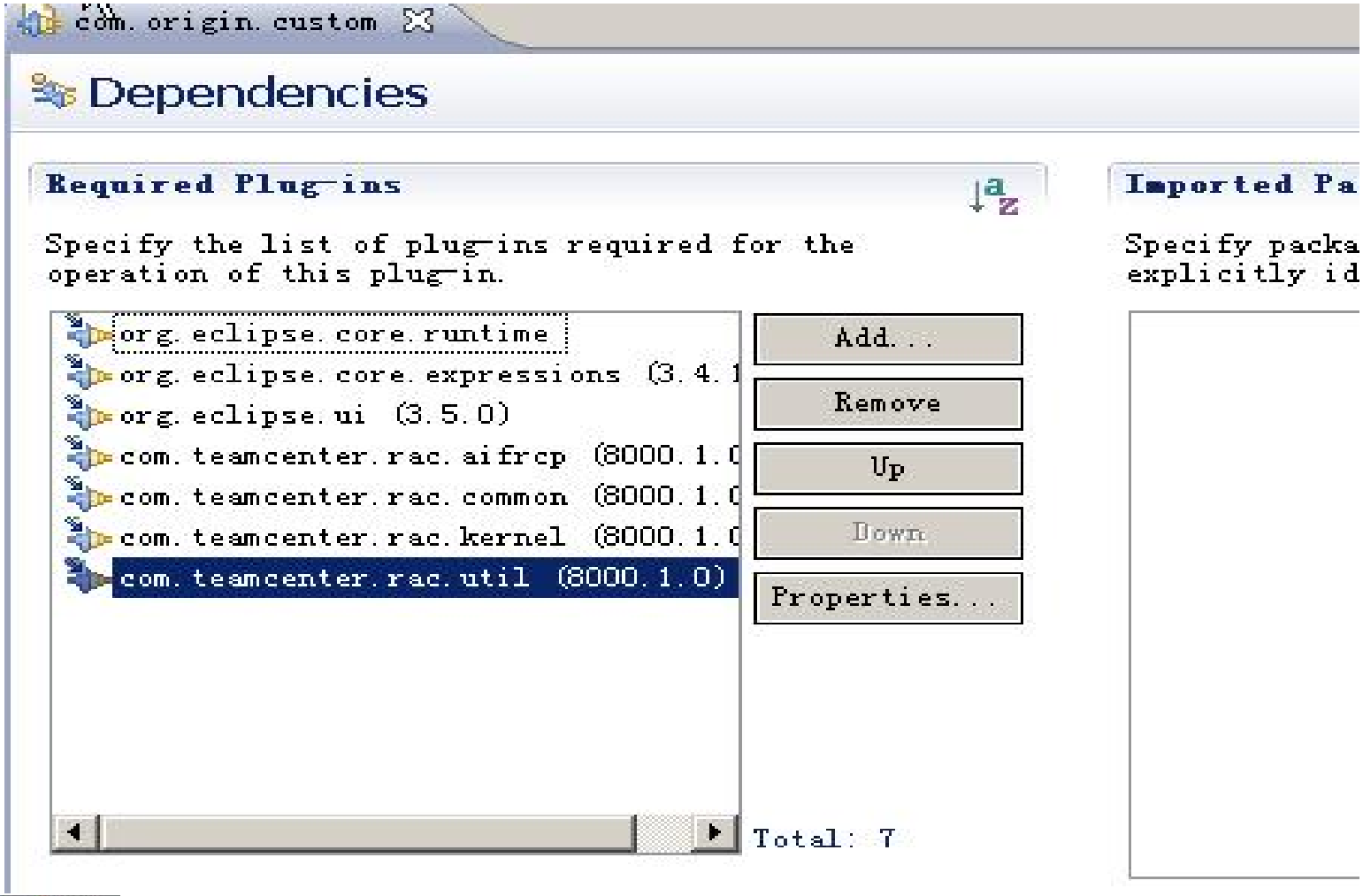
TC中对话框讲解，主要针对实例，在系统中添加新建Folder操作

5.在Eclipse中打开 **MANIFEST.MF** 文件,选择 **Dependencies**页， 并添加以下依赖关系：

- org.eclipse.core.runtime
- org.eclipse.core.expressions
- org.eclipse.ui
- com.teamcenter.rac.aifrcp
- com.teamcenter.rac.common
- com.teamcenter.rac.kernel
- com.teamcenter.rac.util

TC中对话框讲解，主要针对实例，在系统中添加新建Folder操作

如果还要依赖其他插件，添加方式一样，添加后的图如下所示：



TC中对话框讲解，主要针对实例，在系统中添加新建Folder操作

6.修改 **Activator** 类继承 **com.teamenter.rac.kernel.AbstractRACPlugin**。并进行方法的实现和重写。基本代码如下所示：

```
/**
 * The activator class controls the plug-in life cycle
 */
public class Activator extends AbstractRACPlugin {
		// The plug-in ID
		public static final String PLUGIN_ID = "com.origin.custom";
		// The shared instance
		private static Activator plugin;

		/**
		 * The constructor
		 */
		public Activator()
		{
			super();
			Activator.plugin = this;
		}
		/*
		 * (non-Javadoc)
		 * @see org.eclipse.core.runtime.Plugins#start(org.osgi.framework
		 * .BundleContext)
		 */
		public void start(BundleContext context) throws Exception {
			super.start(context);
			plugin = this;
		}
		/*
		 * (non-Javadoc)
		 * @see org.eclipse.core.runtime.Plugin#stop(org.osgi.framework
		 * .BundleContext)
		 */
		public void stop(BundleContext context) throws Exception {
			plugin = null;
			super.stop(context);
		}
}
```

TC中对话框讲解，主要针对实例，在系统中添加新建Folder操作

7. 新建com.origin.custom.handler包，并在该包中新建Handler类NewFolderHandler,该类继承于AbstractHandler。代码如下:

```
package com.origin.custom.handler;
import org.eclipse.core.commands.AbstractHandler;
import org.eclipse.core.commands.ExecutionEvent;
import org.eclipse.core.commands.ExecutionException;
public class NewFolderHandler extends AbstractHandler {
        @Override
        public Object execute(ExecutionEvent arg0) throws
ExecutionException {
                // TODO Auto-generated method stub
                return null;
        }
}
```

TC中对话框讲解，主要针对实例，在系统中添加新建Folder操作

8.分别进行菜单栏，工具栏，还有右键菜单的添加。

a) 扩展org.eclipse.ui.commands，代码如下：

```
<extension point="org.eclipse.ui.commands">
      <command
         name="新建文件夹"
         id="com.origin.custom.handler.newFolderHandler">
      </command>
</extension>
```

b)扩展org.eclipse.ui.handlers,代码如下：

```
<extension point="org.eclipse.ui.handlers">
      <handler
         commandId="com.origin.custom.handler.newFolderHandler"
         class="com.origin.custom.handler.NewFolderHandler">
      </handler>
 </extension>
```

TC中对话框讲解，主要针对实例，在系统中添加新建Folder操作

c) 通过扩展org.eclipse.ui.menus分别添加该操作到菜单栏，工具栏，以及右键菜单。

添加到菜单代码

```
<menuContribution locationURI="menu:org.eclipse.ui.main.menu?after=additions">
      <menu label="客户化菜单(M)" mnemonic="M" id="customMenus">
        <command
            commandId="com.origin.custom.handler.newFolderHandler"
            mnemonic="S"
            icon="icons/newfolder_16.png"
            id="customMenus">
            <visibleWhen>
               <reference
                   definitionId="com.teamcenter.rac.ui.inMainPerspective">
               </reference>
            </visibleWhen>
        </command>
      </menu>
</menuContribution>
```

TC中对话框讲解，主要针对实例，在系统中添加新建Folder操作

添加到工具栏代码

```
<menuContribution
      locationURI="toolbar:org.eclipse.ui.main.toolbar?after=additions">
         <toolbar id="customToolbar">
            <command
                  commandId="com.origin.custom.handler.newFolderHandler"
                  icon="icons/newfolder_16.png"
                  tooltip="新建文件夹"
                  id="customToolbar">
            </command>
         </toolbar>
</menuContribution>
```

TC中对话框讲解，主要针对实例，在系统中添加新建Folder操作

添加到右键菜单

```
<menuContribution
         locationURI="popup:org.eclipse.ui.popup.any?after=additions">
         <command
            commandId="com.origin.custom.handler.newFolderHandler"
            mnemonic="S"
            icon="icons/newfolder_16.png"
            id="customPopup">
            <visibleWhen>
               <reference
                     definitionId="com.teamcenter.rac.ui.inMainPerspective">
               </reference>
            </visibleWhen>
         </command>
 </menuContribution>
```

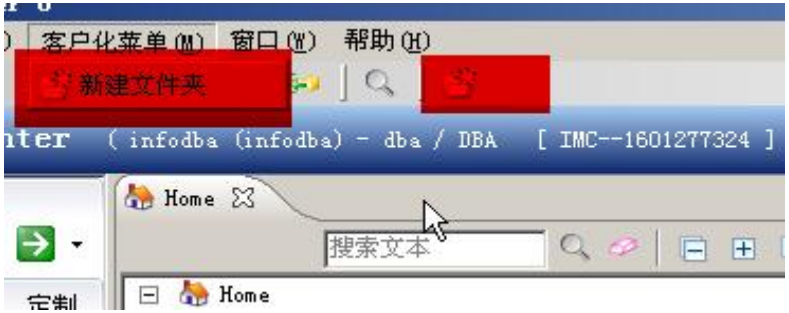
TC中对话框讲解，主要针对实例，在系统中添加新建Folder操作

添加到右键菜单

```
<menuContribution
         locationURI="popup:org.eclipse.ui.popup.any?after=additions">
         <command
            commandId="com.origin.custom.handler.newFolderHandler"
            mnemonic="S"
            icon="icons/newfolder_16.png"
            id="customPopup">
            <visibleWhen>
               <reference
                     definitionId="com.teamcenter.rac.ui.inMainPerspective">
               </reference>
            </visibleWhen>
         </command>
 </menuContribution>
```

TC中对话框讲解，主要针对实例，在系统中添加新建Folder操作

添加后的效果如下图所示：



TC中对话框讲解，主要针对实例，在系统中添加新建Folder操作

9.新建Folder逻辑代码的实现。

为了和系统的架构代码保持一致，我们通过分别通过 **NewFolderCustomAction，NewFolderCustomCommand，NewFolderCustomDialog, NewFolderCustomOperation**去实现。

a)新建com.origin.custom.handler.newfolder包，在该包中新建NewFolderCustomAction类，该类继承AbstractAIFAction类并对run方法进　行重写，代码如下：

```
public void run() {

    try{
        AbstractAIFCommand abstractaifcommand = new NewFolderCustomCommand(
                                                    parent, application
                                );

         abstractaifcommand.executeModal();

    }catch(Exception exception){
          MessageBox.post(parent, exception);
    }
}
```

TC中对话框讲解，主要针对实例，在系统中添加新建Folder操作

b) 在com.origin.custom.handler.newfolder包新建NewFolderCustomCommand类并继承AbstractAIFCommand类，主要代码如下:

```
public NewFolderCustomCommand(Frame frame, AbstractAIFApplication

                                                    abstractaifapplication){
    try{
        parentFrame = frame;
        application = abstractaifapplication;
        targetArray = application.getTargetComponent();
        session = (TCSession) abstractaifapplication.getSession();
        if(targetArray != null){
                AbstractAIFDialog abstracttccommanddialog = new
                      NewFolderCustomDialog(this,true);
            if(abstracttccommanddialog != null){
                 setRunnable(abstracttccommanddialog);
            }
        }else{
            MessageBox.post("请选择对象", "提示 ",MessageBox.WARNING);
        }
     }catch(Exception exception){
        MessageBox.post(frame, exception);
     }
}
```

TC中对话框讲解，主要针对实例，在系统中添加新建Folder操作

c) 在com.origin.custom.handler.newfolder包新建NewFolderCustomDialog类，并继承AbstractAIFDialog类，并实现InterfaceAIFOperationListener方法。其中核心代码如下：

```
public void initUI(){
    setTitle("创建文件夹对话框");
    Dimension dimension = new Dimension();
    dimension.setSize(300,70);
    setPreferredSize(dimension);
    JPanel parentPanel = new JPanel(new FlowLayout());
    final iTextField itext = new iTextField(20);
    JLabel label = new JLabel("文件夹名称:");
    JButton button = new JButton("确定");
    button.addActionListener(new ActionListener(){
        @Override
         public void actionPerformed(ActionEvent e) {
              //调用Operatio
         }
     });
    parentPanel.add(label);
    parentPanel.add(itext);
    parentPanel.add(button);
    getContentPane().add(parentPanel);
    pack();
    centerToScreen(1.0D, 0.75D);
}
```

TC中对话框讲解，主要针对实例，在系统中添加新建Folder操作

和Operation类的操作与调用定义在按钮事件中，具体的过程调用代码如下所示：

```
public void actionPerformed(ActionEvent e) {
      // TODO Auto-generated method stub
     String folderName = itext.getText();

     if(folderName != null && !"".equals(folderName)){

           NewFolderCustomOperation newFolderOperation = new

                 NewFolderCustomOperation(

                             tcsession,tccomponent,folderName);

           tcsession.queueOperation(newFolderOperation);

      }else{

           MessageBox.post("文件夹名称不能为空", "提示 ",MessageBox.ERROR);

      }

}
```

TC中对话框讲解，主要针对实例，在系统中添加新建Folder操作

c) 在com.origin.custom.handler.newfolder包新建NewFolderCustomDialog类，并继承AbstractAIFDialog类，并实现InterfaceAIFOperationListener方法。其中核心代码如下：

```
public void initUI(){
    setTitle("创建文件夹对话框");
    Dimension dimension = new Dimension();
    dimension.setSize(300,70);
    setPreferredSize(dimension);
    JPanel parentPanel = new JPanel(new FlowLayout());
    final iTextField itext = new iTextField(20);
    JLabel label = new JLabel("文件夹名称:");
    JButton button = new JButton("确定");
    button.addActionListener(new ActionListener(){
        @Override
         public void actionPerformed(ActionEvent e) {
              //调用Operatio
         }
     });
    parentPanel.add(label);
    parentPanel.add(itext);
    parentPanel.add(button);
    getContentPane().add(parentPanel);
    pack();
    centerToScreen(1.0D, 0.75D);
}
```

TC中对话框讲解，主要针对实例，在系统中添加新建Folder操作

d)在com.origin.custom.handler.newfolder包新建NewFolderCustomOperation类，并继承AbstractAIFOperation。重写executeOperation()方法。创建文件夹的逻辑操作都在该类中进行了实现，实现代码如下：

```
public class NewFolderCustomOperation extends AbstractAIFOperation {
    private TCComponent tccomponent = null;
    private String folderName = null;
    private TCSession session = null;

    public NewFolderCustomOperation(TCSession session,TCComponent
                        tccomponent,String folderName){
                    this.tccomponent = tccomponent;
                    this.folderName = folderName;
                    this.session = session;
    }

    @Override
    public void executeOperation() throws Exception {
       TCComponentFolderType t =

                        (TCComponentFolderType)session.getTypeComponent("Folder");

      TCComponentFolder f = t.create(folderName, "My Folder Description","Folder");
      tccomponent.add("contents", f);
    }
}
```

TC中对话框讲解，主要针对实例，在系统中添加新建Folder操作

10.在新建文件夹Handler类中添加代码，进行Action的调用，调用代码如下：

```
@Override
public Object execute(ExecutionEvent arg0) throws ExecutionException {

        AbstractAIFUIApplication abstractAIFUIApplication =

                                                AIFUtility.getCurrentApplication();
      NewFolderCustomAction newFolderAction = new

                              NewFolderCustomAction(abstractAIFUIApplication, null);
        new Thread(newFolderAction).start();


         return null;


}
```

TC中对话框讲解，主要针对实例，在系统中添加新建Folder操作

11.通过Eclipse启动TC，进行创建文件夹的测试，效果如下：