**【方案背景】**

EOS Studio是建立在Eclipse基础上的，Eclipse内置了CVS client，EOS的团队开发支持用的就是这个CVS client。

**【需求来源】**

在BPM项目中，所有的公共文件放置在WEB-INF下的config 目录下。目前是相关开发人员在固定的时间段将公共文件进行打包并上传到CVS受控库，再由测试人员进行下载部署和测试。

这一过程很麻烦，同时，经常遇到由于各种问题忘记打包导致部署的公共文件仍然是旧版本，增加了沟通、人力、时间等各种成本。

**【需求分析】**

深登自动化测试项目中，持续集成实现了通过脚本自动获取最新代码并进行编译，打包，上传ftp，部署等等一系列操作。

所以，针对上述需求，也完全可以以前持续集成的经验通过脚本将这一人工操作自动化。

**【方案设计】**

**先来看一下是如何搭建CVS服务器和客户端的操作吧。**

### 安装文件

**服务端：cvsnt-2.5.03.2382.msi**

**客户端：WinCvs 1.3 wincvs.exe**

### 安装过程

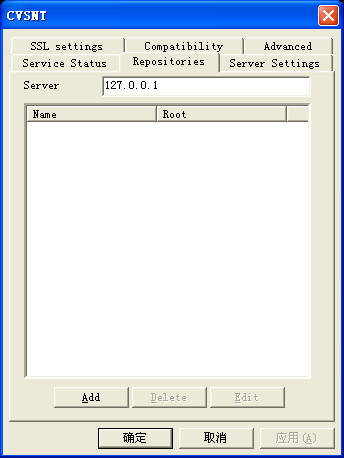
安装顺序本没有规定，但尽量先安装CVSNT，然后再安装WinCvs。 双击安装文件一路“下一步”就可以。记得安装路径~~~装完CVSNT后会提示重启机器，立即重启或稍后重启都行。

### 配置CVSNT

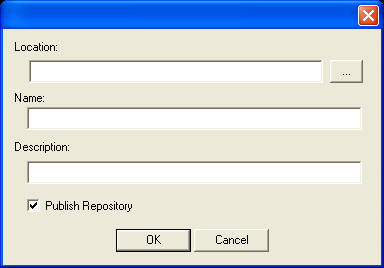
装完CVSNT后点开始菜单－>所有程序－>CVSNT－>CVSNT Control Panel，打开如下的配置界面：



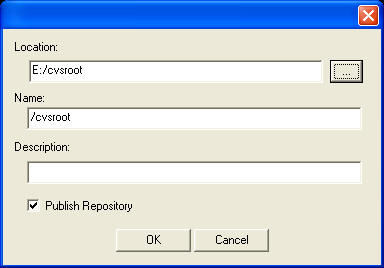
在这个界面上可以看到服务是否已经启动及CVSNT的版本信息。点Repositories标签，打开如下的界面：



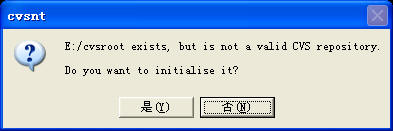
这里将列出所有的repository，第一次安装的话列表应该是空的，点Add按钮，将打开如下的对话框



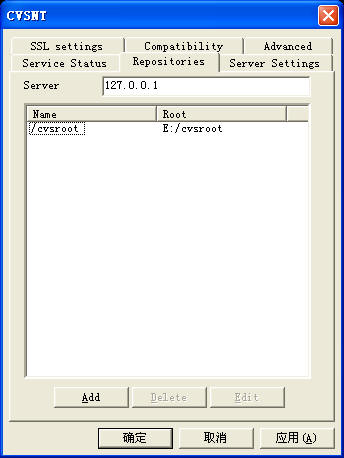
在这个对话框输入如下信息（输入这些信息前，我在E:\下新建了一个空的cvsroot目录，这个目录就是Repositories在服务器上的存放地址）：



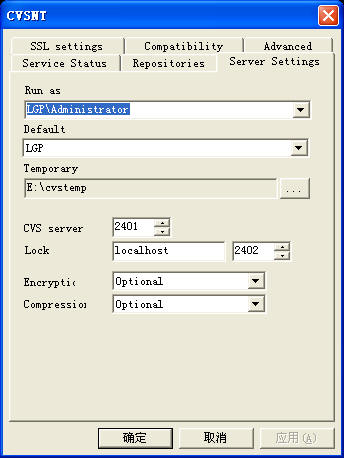
点OK按钮，将弹出如下确认对话框：



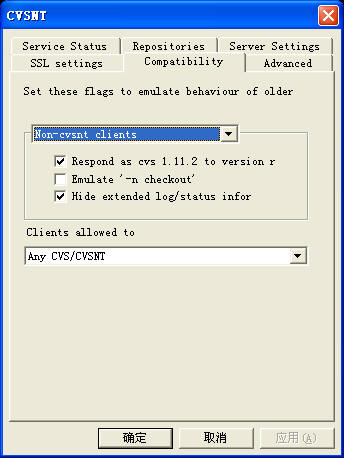
这是提示你是否初始化repository，点是，将看到如下界面：



这说明一个repository就建立好了。再点Server Settings标签，设置如下：



Run as栏用administrator用户，Temporaty我指定的是E:\cvstemp，然后再点Compatibility表签，打开如下界面：



注意要勾选Respond as cvs 1.11.2 to version…，要不然就会出问题，这个好像是为了保证兼容性。

到现在，CVSNT的配置就基本完毕了。以后的操作都是在客户机上进行。如果只是在服务器上配CVSNT，那做到这一步就可以换机器了。

### 简单的权限分配

#### 安装配置WinCvs

安装完WinCvs后，在客户机上的桌面上应该有WinCvs这个图标（一个黄色的小鱼），双击，将打开WinCvs的界面（为避免CVS的版本问题，一定要在做以下操作前拷贝C:\Program Files\CVSNT（CVSNT的安装目录）目录下的所有文件，然后用这些文件覆盖C:\Program Files\GNU\WinCvs 1.3\CVSNT（WinCvs的安装目录）目录下的所有文件）。

#### 创建系统用户

现在要实现的权限分配是：

建立3个用户：test1、test2、test3，密码与用户名一致。

建立一个组：test，用户test1、test2、test3均在此组中。

我们的目标是用户test1能且仅能编辑修改pkg1下的内容，用户test2能且仅能编辑修改pkg2下的内容，用户test3能且仅能编辑修改pkg3下的内容。

#### 建立客户端本地目录

为方便讲解，我在F:\下建立了一个目录clientadmin/project，其目录结构如下

project

|--pkg1

|--test2.txt

|--pkg2

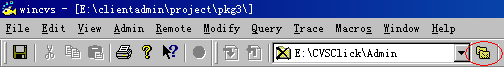
|--test3.txt

|--pkg3

|--test4.txt

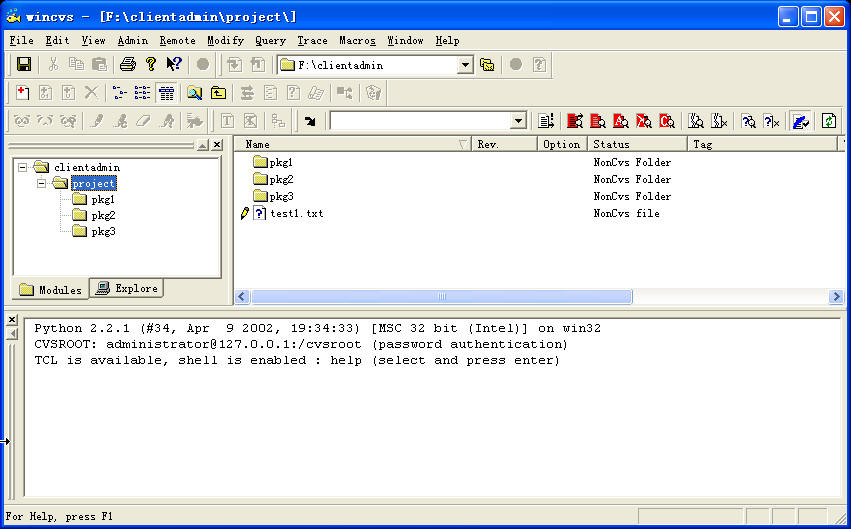
#### 导入本地目录

选择本地目录，如图点击文件夹按钮：



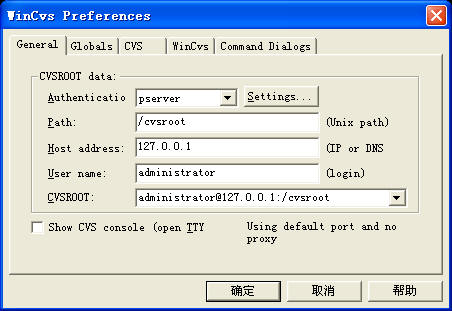
在浏览文件夹中找到并选择F:\clientadmin\project

然后会如下图所示画面：



#### 进行WinCvs设置

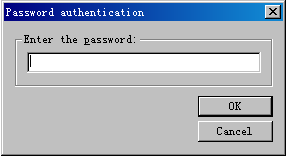
点菜单栏的Admin－>Preferences…，打开如下对话框：



设置如图上所示，然后点确定。

#### 用管理员用户登录

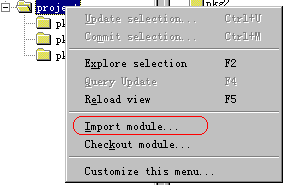
然后再点菜单栏的Admin－>Login…，打开一个对话框，点确定，然后会提示输入密码，如下图：



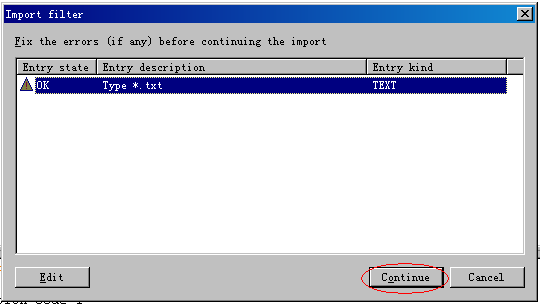
输入操作系统的administrator的密码后点确定，将登录成功。

#### 导入工程到CVS端

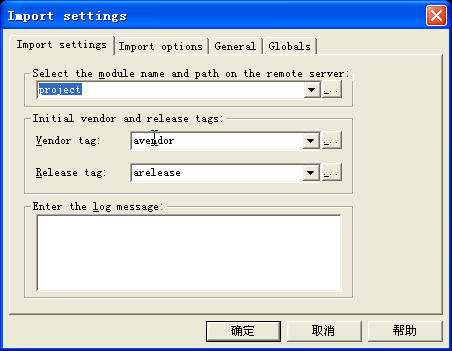
右键点击左侧目录树中的project，在弹出的右键菜单中点Import module…



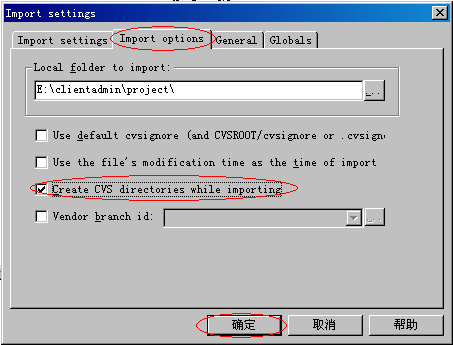
将弹出Import filter对话框，点Continue按钮



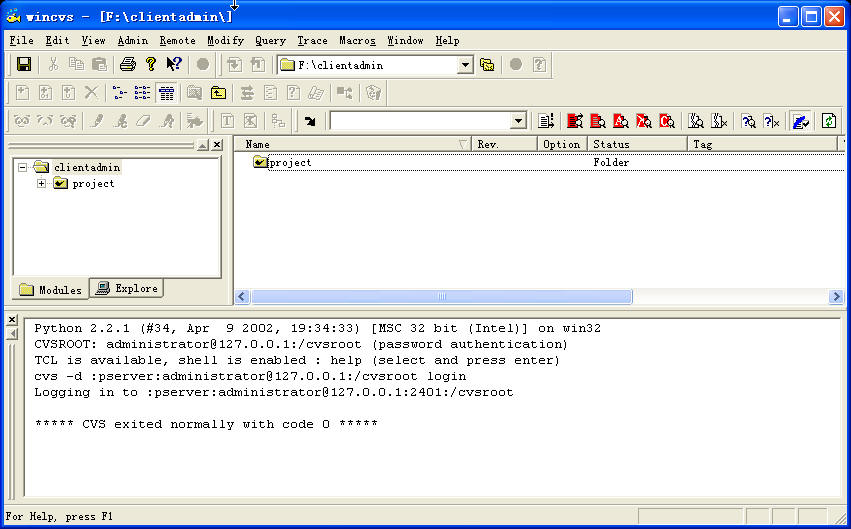
将打开Import settings对话框，如下图所示：



输入projet，然后点Import options标签，在多选框列表中选中Create CVS directories while importing，然后点确定。



这个操作将在CVS服务器的根目录下增加一个目录E:\cvsroot\project，这就是一个module了，如下图所示：



#### 创建CVS用户

接下来开始建立用户，点菜单栏的Admin－>Command Line…或者按快捷键Ctrl+L，将打开如下的命令输入框：



输入如图中所示的命令，然后点确定或按回车键，然后输入两次密码（均为test1），就加好一个用户了。其它两个用户如法炮制。加用户的命令的详细帮助可以参考C:\Program Files\CVSNT\cvs.chm，这个就是CVSNT Manual，以后的其它命令的详细帮助都可以在这个文件中找到。

#### 创建组

接下来创建组，创建组的操作比较简单，就是编辑服务器上的文件的内容，文件所在目录为E:\cvsroot\CVSROOT，默认group文件并不存在，需新建。这个文件中的内容就是简简单单一行，如下：

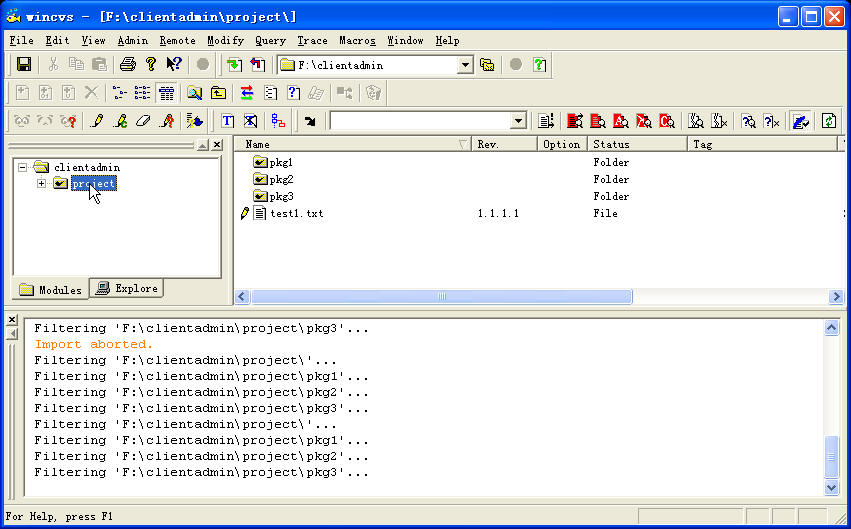
test:test1 test2 test3

这个表示有一个test组，test1、test2、test3均是改组的成员。

#### 分配权限

分配权限的命令为cvs chacl …。

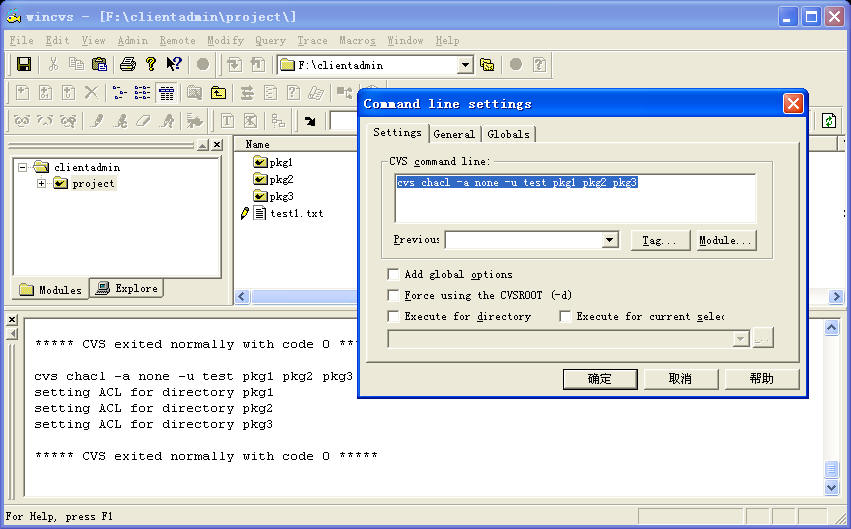
在WinCvs中选中project，如下图：



然后按快捷键Ctrl+L，打开命令输入框，按顺序输入以下命令：

cvs chacl –a none –u test pkg1 pkg2 pkg3

按回车键，如下图所示：



将出现上图左下角所示的命令执行成功信息。上面的命令达到的效果是test用户对pkg1、pkg2、pkg3均无权限，test用户其实就是我们前面建立的组（group），它包括了test1、test2、test3三个用户。

接下来按顺序执行如下三个命令：

cvs chacl –a read,write,create –u test1 pkg1

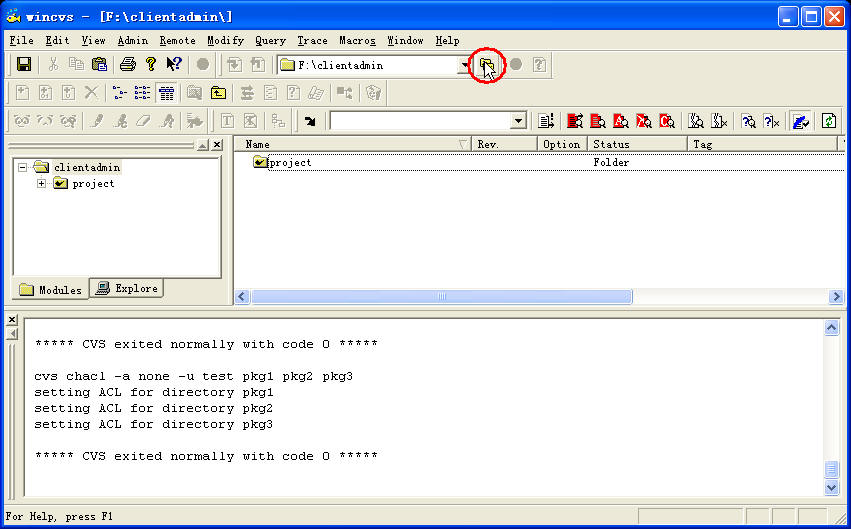
cvs chacl –a read,write,create –u test2 pkg2

cvs chacl –a read,write,create –u test3 pkg3

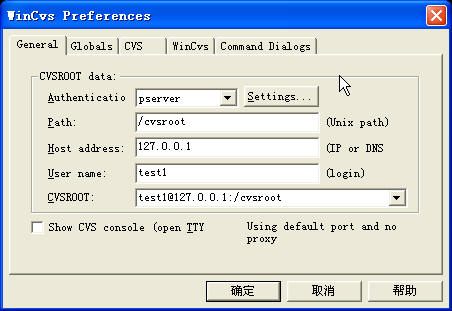
这三个命令成功执行完毕就达到我们的目的。

#### 测试效果

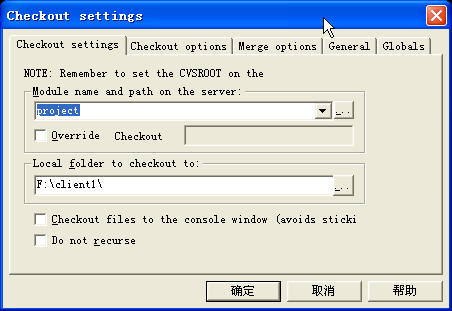
在F:\新建三个目录client1，client2，client3，然后点如下图所示的change location按钮，修改目录，选择test1的工作目录client1，做完这一步后，不要着急，先关掉WinCvs：



再打开WinCvs，修改Preferences，如下图：



然后登录，登录成功后，在client1上点右键－>Checkout module…，打开如下界面：



点确定，就可以看到权限控制后的效果了，因为project下pkg2和pkg3没有checkout下来。依样画葫芦，换用test2，test3看看。

**接下来仔细看一下ant 脚本是如何将人工操作自动化的。**

第一步： 分别checkout 主干和分支脚本。通过cvs的 tag 标签来获取。 package 为需要checkout的moudle 包， dest 为保存的路径。

|  |
| --- |
| <target name="checkOutBuild" >  <cvs cvsRoot="\:pserver\:admin\:admin@126.0.0.14\:/bpmproject" command="checkout" package="${cvspackageBpmConfig}" dest="${bpm\_pre\_env\_dir}" tag="${release\_name\_BPM\_head}" passfile="ant-cvs.cvspass" reallyquiet="false"/>  <cvs cvsRoot="\:pserver\:admin\:admin@126.0.0.14\:/bpmproject" command="checkout" package="${cvspackageBpmConfig}" dest="${bpm\_fenzhi\_pre\_env\_dir}" tag="${release\_name\_bpm\_fenzhi}" passfile="ant-cvs.cvspass" reallyquiet="false"/>  </target> |

第二步： 打包成ecd。需要以下两个步骤：

|  |
| --- |
| <!--target name="build.contribution.jar" >  <jar destfile="D:\bpmConfigFiles/bpm\_main\_jar/bpm\_main.jar" manifestencoding="UTF-8">  <fileset dir="D:\bpmConfigFiles\bpm\bpmproject\bpm二期\src\bpm"/>  </jar>  <jar destfile="D:\bpmConfigFiles/bpm\_fenzhi\_jar/bpm\_fenzhi.jar" manifestencoding="UTF-8">  <fileset dir="D:\bpmConfigFiles\bpm\_fenzhi\bpmproject\bpm二期\src\bpm"/>  </jar>  </target--> |

<target name="build.contribution.ecd" >

<jar destfile="D:/bpmConfigFiles/bpm/bpmproject/受控库/上线补丁/公共文件补丁/BPM\_主干公共文件.ecd">

<jar destfile="D:/bpmConfigFiles/bpm/bpmproject/受控库/上线补丁/公共文件补丁/BPM\_分支公共文件.ecd">

</target-->

第三步： 将打包好的ecd 文件重新提交到CVS 目录。

<target name="commitPackage" description="check the modified,registarted files into the cvs" >

<cvs failonerror="false" cvsRoot="${CVSROOT\_A}" dest="D:/bpmConfigFiles/bpm/bpmproject/受控库/上线补丁/公共文件补丁" command="update -R" />

<cvs failonerror="false" cvsRoot="${CVSROOT\_A}" dest="D:/bpmConfigFiles/bpm/bpmproject/受控库/上线补丁/公共文件补丁" command="add -kb \*.ecd" />

<cvs failonerror="false" cvsRoot="${CVSROOT\_A}" dest="D:/bpmConfigFiles/bpm/bpmproject/受控库/上线补丁/公共文件补丁" command="commit -m ${comments}" />

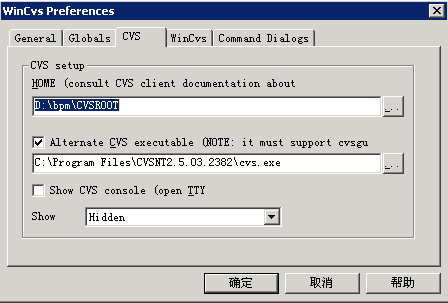
</target>

其中： dest 表示CVS 上需要更新和提交的目录。 在做CVS commit 操作时，必须对该目录先进行update 操作，然后add，最后commit。 否则会报异常“cvs [add aborted]: there is no version here; do ‘cvs checkout’ first”。

Cvs用户权限的问题也有可能导致提交不成功，所以必须保证用户拥有“create,write,read”权限。例如： 要给用户administrator 添加这三个权限， 可以通过命令“cvs chacl administrator:cwr”进行操作。

在这个过程中，也报了一个错误“cvs [chacl aborted]:server does not support chacl”。这个问题是由于CVS 版本过低导致的，可以安装更高版本的cvs。

另外，可以在winCvs 首选项中设置CVS 的可执行程序。 默认的目录是wincvs 目录下的cvsnt， 这里要把它改成安装的cvsnt 本身，如下图： 这样就可以解决了。



提交以后，CVS 内容有可能会出现文件被破坏的情况，这是因为CVS 默认是以 .txt 的文本文件导入的，而.excel、.doc以及其他后缀类型的则是二进制文件，二进制的文件绝对不能使用文本文件导入，因为这样的操作会将数据类型损坏。 所以，在add 的时候必须添加 –kb 。

第四步： 清理。 将对应目录的文件删除。

<target name="deletePackage" >

<delete >

<fileset dir="D:\bpmConfigFiles/bpm/bpmproject/受控库/上线补丁/公共文件补丁" includes="\*.ecd" />

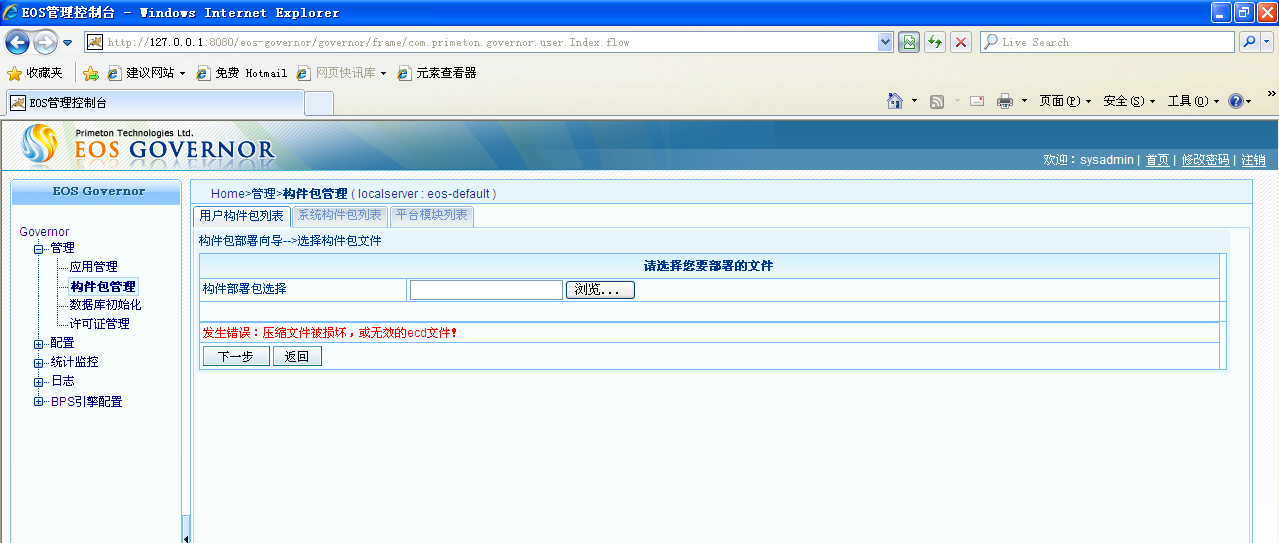
</delete>

</target>

**【备注】** 这里的绝对路径可以通过属性文件.properties 来进行设置，更便于维护和修改。

那么到这里，基本上整个过程就完成了。 最后还差一个验证部署的过程。

打开gorvernor 登陆，进入管理---构件包管理---eos-default----部署。 之后会提示如下信息： 压缩文件损坏，或无效的ecd 文件，导致部署不成功。



这里可以看到压缩文件并没有损坏，所以可以证明是ecd 的文件。 通过仔细对比后发现，构件包导出ecd 后是由jar 文件组成的，而eos-web下的WEB-INF 目录导出ecd 则不需要先打成jar 包。 所以，在打包的过程中只需要打ecd 一个步骤即可。

注释掉前面打jar 包的脚本，重新运行后部署，成功了！！！