1、线程哪里我有一个问题，你为什么要加载kenel32.dll,再调用QueueUserWorkItem，这个库不是自动加载，直接可以调用QueueUserWorkItem的吗？

高版本VS可以直接调用，低版本VS这个接口没对外开放，需加载这个dll

写代码尽量考虑到用户的运行环境，虽然目前我们的后台问题不大，如果以后我们做平台，软件直接卖给客户，那要考虑到用户可能的运行环境，别人就不一定是win7了，可能是xp，更有可能是win2000或是wince。如果我们代码在低版本的操作系统中没法跑，说明我们兼容没做好。因为VC 6编辑出来的东西能在win2000,wince上跑

2、我上一家公司的时候，很多客户的电脑配置都很低，他们不可能升级系统，那成本得多大？基本都是xp,有些更古董的wince ,还有linux系统，因为那些操作系统是免费的，客户每年要卖几千套设备，如果每套设备都配个win7 ,又不可能装盗版软件，你想想得花多少成本？

如果要用linux，那我们的都不能跑，java可以跨平台，跨平台有很大风险，比如window下一些功能，跨平台不一定能实现，或是很难实现。

3、朱工，系统有archive类实现串行化，为什么还要自己实现一个库呢？

MFC的那个没有用内存映射的方式，速度没法和我这个相比，Archive只是把数据放到一个连续的数据块中，真正写与读的时候是用内存映射。

4、重复释放困惑

你写一个函数，参数是一个指针，因为你不知道别人会怎么用，那你肯定要在函数中判空处理

Test(C\*\*\* pObj)

{

if(pObj) ...

}

你调用别人写的函数，因为你不知道他函数里有没有对参数判空，所以你就必须先判空然后才调用他的函数

if(pObj)

{

Test(pObj);

}

多层防护，每一层释放都首先在链表中删除，这样如果已经释放了，就不会再次释放，如果只判空，可以出现野指针。

5、工作态度

要把任何一个开发任务当做产品来对待，不能把功能做完了就完事了，有时候要想一想有没有漏洞，合不合理，站用户角度想问题

比如很简单的例子，复制组，然后粘贴时，在组节点上点粘贴是无效，但没任何提示，用户不知道，就只知道怎么粘贴不出来?对不对？

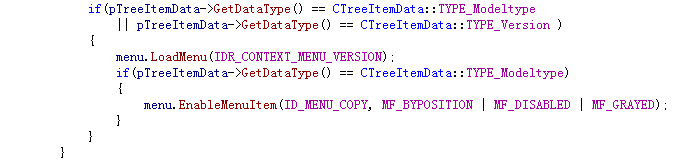
还有就是轴的复制时，新轴的那个移地址应该怎么填，让用户最省心？这都是我们要考虑的事情，

功能码复制，重名的处理？

6、

这个m\_Tree是一个对象，并不是指针，这句代码会有问题，可以去掉。

7、使用menu进行设置按钮状态没有效果

要通过添加消息on\_commond\_UI(x,x1)，然后使用void CParamWorkSpacePanelDialog::OnUpdateRemoveVersion(CCmdUI\* pCmdUI)这个来设置, 你那个只是初始化，后面会通过CCmdUI\* pCmdUI这种消息把它还原了，就看不到效果了。

8、指令处理线程通信问题

对于异步函数读写就不关心回应帧了，如果是同步读写，会置一个同步指令的标志，同步函数将指令放入队列中后然后会等待，收到数据后将数据放入数据缓存中会置一个事件，通知同步接口从缓存中去获取数据。