ACE的安装与配置

 (2005-10-10 19:55:19)

[[http://simg.sinajs.cn/blog7style/images/common/sg_trans.gif](javascript:;)转载▼](javascript:;)

|  |  |
| --- | --- |
|  | 分类： [程序语言](http://blog.sina.com.cn/s/articlelist_1401158725_2_1.html) |

ACE(ADAPTIVE Communication Environment),中文的意思就是自适配通讯环  
境，ACE 是一个用于开发网络程序的优秀的C++的框架，在国外有很广泛的使  
用，在国内一些大的开发通讯产品的公司也有使用。我接触ACE 也有一段时间  
了，虽然时间不长，但我还是感觉到ACE 确实是一个好东西，对于丰富自己的  
知识面有很大的帮助。虽然我们项目目前是采用C 语言来开发，但是当接触ACE后，你会发现“喔，原来程序还可以这样”。例如：我觉得ACE 里面Reactor 框架就是一个非常的东西，我们在开发网络程序的时候，常常采用poll 来监视各种网络事件，但当采用该框架后，你现在只是需要关系你的业务逻辑，当发生特定的网络事件后，框架会回调你的业务逻辑。其实按照这个思路，我们完全可以用C 来实现类似的功能，当你完成这个后，你会发现你原来用C 语言写的过程风格的代码竟然有了OO 的味道。  
ACE 确实是好东西，但也不是能轻松的就能掌握的，我们还需要一步一步的来蚕食这个大象。  
万丈高楼平地起，首先我们还是了解一下ACE 的目录结构，从整体上对ACE  
有一个认识，为今后的进一步学习打下一个基础。  
解开ACE 的压缩包后，你会发现一个ACE\_wrappers 目录，这个目录也就是  
ACE 的HOME 目录，它下面还包含着一些子目录：  
ace：这个目录是ACE 中最重要的目录，它包含了ACE 的所有源码，但遗憾的  
是，ACE 的所有源文件和头文件全部杂乱的堆在这个目录里，这可能也是很多  
开源软件的缺点。其实ACE 的代码完全可以按照不同的功能进行不同目录的划  
分，例如：Reactor 框架和thread 框架代码完全可以划分开，我想一个代码组织  
良好的ACE，将会给大家的学习带来极大的好处，我将在后面的文章里给出ACE  
代码划分的方法；  
ACEXML：这个目录包含了用ACE 实现的一个XML 解析器；  
apps：这个目录包含了用ACE 来实现的一些较大的应用程序，例如：JAWS，  
一个WEB 服务器；  
ASNMP：基于ACE 的SNMP 协议实现；  
bin：包含里用例方便开发的perl 脚本程序，例如：在WIN32 上开发DLL 时候，  
需要导出DLL 的接口；  
docs：ACE 的一些帮助文档，其中ACE-subsets.html 文档，对我们划分ACE  
的代码有很大的帮助；  
examples：是用ACE 来编写的一些例子程序，方便更好的学习和理解ACE；  
include：也是ACE 中一个比较重要的目录，它包含了在不同的平台上编译时候  
的编译规则，库的编译规则等；  
netsvcs：一些基于ACE 的在分布式系统中常用的程序，例如：分布式系统日志  
系统，网络锁，时间同步等；  
TAO：基于ACE 的实时CORBA 实现，TAO 在分布式系统中使用相当广泛，也  
是一个不可多得的好资源；  
tests：用来对ACE 进行回归测试   
  
  
如果要阅读源码，最好结合那个用Doxygen生成的文档，从[url]ftp://ace.cs.wustl.edu/pub/ACE/[/url]下载即可。  
如果讨厌看E文，可以访问以下网址查看中译文：<http://www.flyingdonkey.com/ace/> ，顺便向作者致敬。  
一. Windows下for MSVC的安装与配置  
我们用的是ACE-5.4+TAO-1.4+CIAO-0.4.zip，Windows下解压缩该压缩包到目标路径（设为D:\ACE\_wrappers ），  
创建文件：$ACE\_ROOT/ace/config.h，  
增加一行：#include "ace/config-win32.h"  
如果是在Windows 9x/Me下，需要在$ACE\_ROOT/ace/config.h中#include语句前加入：#define ACE\_HAS\_WINNT4 0  
如果想使用standard C++ header，需要在$ACE\_ROOT/ace/config.h中#include语句前加入：  
#define ACE\_HAS\_STANDARD\_CPP\_LIBRARY 1  
如果想把MFC作为静态库链接到ACE（If you prefer to link MFC as a static library into ACE），则需要在$ACE\_ROOT/ace/config.h中#include语句前加入：  
#define ACE\_HAS\_MFC 1  
如果想生成ACE静态库或在工程中使用ACE静态库，都需要定义以下宏：ACE\_AS\_STATIC\_LIBS  
打开工程，D:\ACE\_wrappers\ace.dsw，编译生成以下库：  
ace.dll/ace.lib （DLL release）  
aced.dll/aced.lib（DLL debug）  
acemfc.dll/acemfc.lib （MFC DLL release）  
acemfcd.dll/acemfcd.lib（MFC DLL debug）  
aces.lib （Static library release）  
acesd.lib （Static library debug）  
这些库可分别实现了Debug/Release, MFC/Non-MFC, Static/Dynamic library.  
其中前四个的dll在D:\ACE\_WRAPPERS\bin目录下， 所有对应的.lib库文件在D:\ACE\_WRAPPERS\ace下。  
ACE是网络通讯中间件，如果机器没有装网卡，就仿真一个，比如可以从控制面板选装MS Loopback Adapter。  
接下来我们创建一个新的工程，如果机器上装了不同版本的ACE，需要针对某个版本配置，方法如下：  
1. 配置C/C++ tab  
(1) Code Generation category中应选择合适的选项：Multithreaded和Multithreaded DLL是for Release版的Debug Multithreaded和Debug Multithreaded DLL是for Debug版的  
  
(2) $(ACE\_ROOT)路径的配置：如要针对不同版本ACE灵活配置的话，则需设置Preprocessor category中"Additional include directories"这一项，指明特定版本ACE所在的路径，比如D:\ACE\_wrappers  
  
2. 链接到特定版本的ACE库。  
(1) 在Project/Setting/Link的Input category下 "Additional library path"中增加特定版本的ACE库文件路径，比如D:\ACE\_wrappers\ace，然后将D:\ACE\_wrappers\bin目录下对应的dll拷贝到工程文件所在路径下,或者在环境变量PATH中加入D:\ACE\_wrappers\bin。  
(2) 在Project/Setting/Link的Input category下"Object/library modules" 中指定需要包含的ACE库(\*.lib)  
如果只装了一个ACE，可采用对所有工程都生效的缺省配置，方法如下：  
1. 添加环境变量：ACE\_ROOT： D:\ACE\_wrappersPATH 中加入：D:\ACE\_wrappers\bin  
2. 设置VS的路径：Include files包含：D:\ACE\_wrapperslibrary files包含：D:\ACE\_wrappers\ace  
此后，就可以在工程中指定需要包含的ACE库(\*.lib)后正常使用了。  
  
ACE配置简介   
开发源代码的类库在配置上要比微软官方的当然复杂一点，好在所有这样的软件步骤都是类似的，所以只要熟悉一个其他的也就触类旁通了。而且借着这个机会还可以学习一点IDE背后细节的东西，何乐而不为呢？主要步骤如下：  
【1】下载源代码，解包。  
【2】打开对应的工程文件，在VC6中进行编译（整个过程大概需要十几分钟，由机器性能决定）。  
【3】把生成的dll文件拷贝到system32目录下（或者任何在你的环境变量中能够找到的路径）。  
【4】在VC6中Tools/Options/Directories中把ACE的目录加入include和lib路径下，并且移到最上面。  
【5】在每个使用ACE的项目中的link选项里加入ace.lib   
【6】使用多线程类库进行编译。  
  
在Windows和VC++6.0下的ACE安装手册- -  
  
  
1、解压ACE.zip，比如说解压到 C:\  
2、找到ACE\_wrappers目录所在位置，比如说是C:\ACE\_wrappers  
  
3、在系统中新建ACE\_ROOT环境变量，值设为 C:\ACE\_wrappers  
具体设置为：我的电脑 －属性 － 高级 － 环境变量 －新建  
4、打开C:\ACE\_wrappers\ace目录  
新建 config.h 文件  
在文件中 添加  
#include "ace/config-win32.h"  
5、如果使用C++标准库，比如 iostream之类的，  
在 config.h 中 #include 之前添加  
#define ACE\_HAS\_STANDARD\_CPP\_LIBRARY 1  
6、如果使用MFC  
在 config.h 中 #include 之前添加  
#define ACE\_HAS\_MFC 1  
7、打开VC6.0 IDE  
设置 Tools -> Option -> Directory ，新增 C:\ACE\_wrappers  
8、打开 C:\ACE\_wrappers\ace\ace.dsw 文件  
编译，等待大概 10分钟 左右，编译完成！  
9、这样，就可以在 以后的 MFC 工程文件的头文件中 #include "ace/OS.h"了  
  
  
二. Linux下的安装与配置1.安装 以用户aceuser为例 (1)下载ACE-5.4+TAO-1.4+CIAO-0.4.tar.gz(其它版本的也一样)  
(2)解压之   
#cd /home/aceuser/ace   
#tar -xzvf ACE-5.4+TAO-1.4+CIAO-0.4.tar.gz  
(3)   
#vi /home/aceuser/.bashrc 加入下面两行：   
export ACE\_ROOT=/home/aceuser/ace/ACE\_wrappers   
export LD\_LIBRARY\_PATH=$LD\_LIBRARY\_PATH:$ACE\_ROOT/ace  
  
并使这些环境变量生效（运行source 或者重新登陆）。  
(4)  
  
#cd /home/aceuser/ace/ACE\_wrappers/ace/   
#cp ./config-linux.h ./config.h   
#cd ../include/makeinclude/   
#ln -s ./platform\_linux.GNU ./platform\_macros.GNU  
(5)编译   
#cd /home/aceuser/ace/ACE\_wrappers/ace/   
#make 对于ACE-5.4+TAO-1.4+CIAO-0.4这个版本，不能ACE-5.4+TAO-1.4+CIAO-0.4.zip这个压缩包，否则会出现 如下错误：   
ln -s libACE.so.5.4.0 libACE.so chmod a+rx libACE.so.5.4.0 : command not foundpers/bin/ace\_components: line 10   
'/home/aceuser/ace/ACE\_wrappers/bin/ace\_components: line 31: syntax error near unexpected token' in '/home/aceuser/ace/ACE\_wrappers/bin/ace\_components: line 31: ' case $1 in make[1]: \*\*\*[ACE\_COMONENTS] Error 2 make[1]: Leaving directory '/home/aceuser/ace/ACE\_wrappers/ace' make: \*\*\*[all] Error 2   
如果换成ACE-5.4+TAO-1.4+CIAO-0.4.tar.gz就没有任何问题了  
编译时也可以根据自己的需要设置编译选项   
#make [options]   
下面是option的描述：  
Option   
  
Descriptiondebug=1|0 Enable or disable debugging in the built library or program. Default is enabled (1).  
optimize=1|0 Turn compiler optimization on or off. Default is off (0).  
  
buildbits=bits Explicitly select, for example, 32-bit or 64-bit build target. Default is the compiler's default for the build machine. This option works for AIX, Solaris, and HP-UX.  
  
exceptions=1|0 Enable or disable exception handling. Default is platform specific but usually enabled (1).  
  
inline=1|0 Enable or disable inlining of many of ACE's methods. Default is platform specific but usually enabled (1).  
  
templates=model Specify how templates are instantiated. Most common values for model are automatic, the default for compilers that support it well, and explicit, requiring source code directives to explicitly instantiate needed templates (see Section 1.6.1).  
  
static\_libs=1|0 Build and use static libraries. Default is to not build static libraries (0).  
2.使用下面介绍如何从零开始建立一个使用ACE的工程。并使之run起来  
(1)编写使用了ACE的代码 (2)编写Makefile  
BIN = hello\_ace FILES = Piece2 Piece3 SRC= $(addsuffix .cpp,$(FILES)) OBJ= $(addsuffix .o,$(FILES)) BUILD = $(VBIN) #--------------------------------------------------------- # Include macros and targets #--------------------------------------------------------- include $(ACE\_ROOT)/include/makeinclude/wrapper\_macros.GNU include $(ACE\_ROOT)/include/makeinclude/macros.GNU include $(ACE\_ROOT)/include/makeinclude/rules.common.GNU include $(ACE\_ROOT)/include/makeinclude/rules.nonested.GNU include $(ACE\_ROOT)/include/makeinclude/rules.bin.GNU include $(ACE\_ROOT)/include/makeinclude/rules.local.GNU  
以上的Makefile包含3个cpp文件hello\_ace.cpp、Piece2.cpp和Piece3.cpp  
(3)编译 #make 如果程序没有任何问题，现在程序应该就可以运行了。