Active Template Library

Library support for building COM components

CSE775 - Distributed Objects, Spring 2012

Jim Fawcett

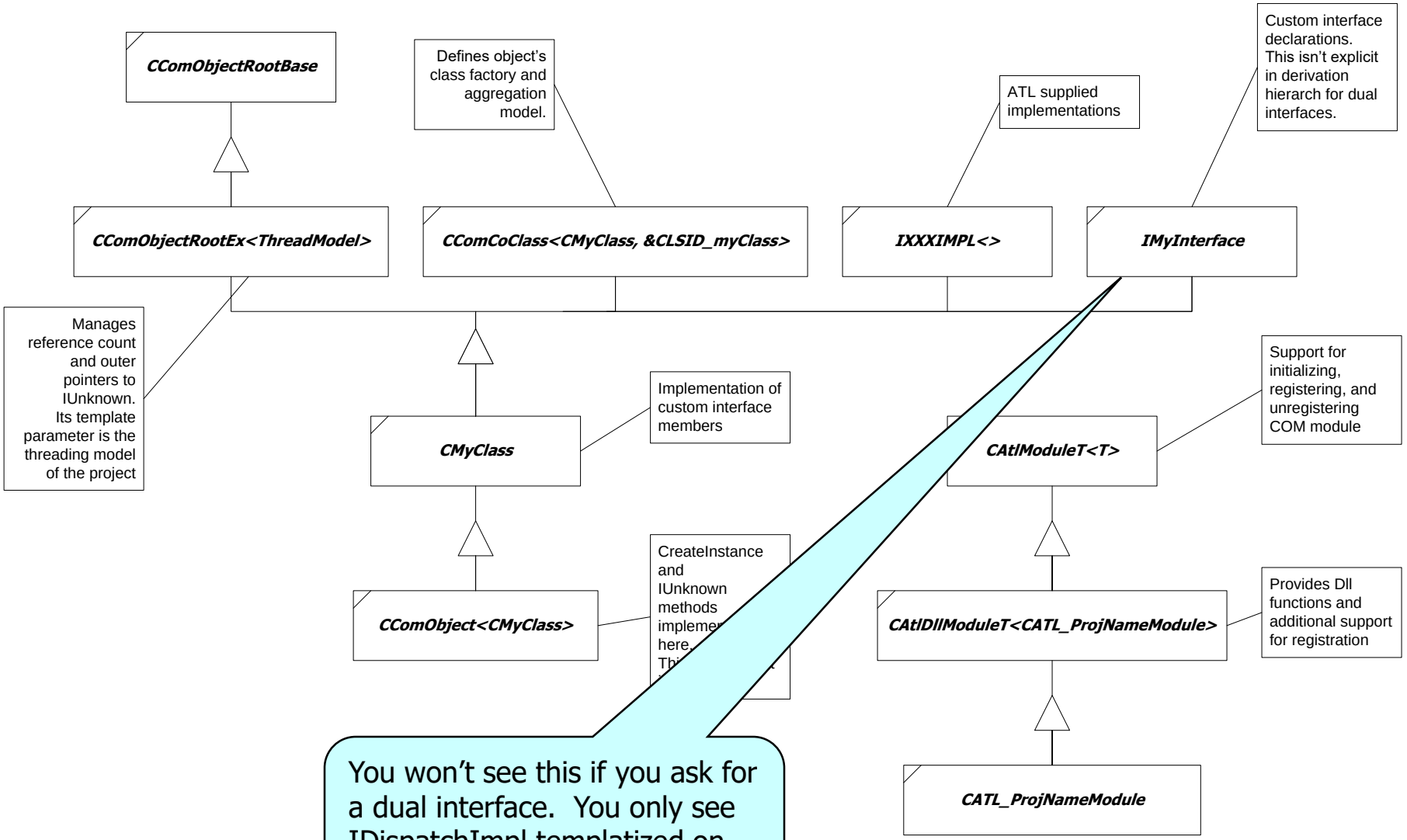
Library Support for COM

- One of the difficulties with designing COM components with no support other than C++ is that COM does not support reuse through inheritance.
  - This means that either you implement all the interfaces you need yourself, even the standard COM interfaces, or:
  - You make your server a client of the standard COM component and use aggregation to provide access to the standard interface through COM aggregation – a very messy process.
- COM has addressed this problem with two types of support:
  - The Microsoft Foundation Classes (MFC) provide extensive support for creating COM clients and servers.
  - The Active Template Library (ATL) also provides a lot of support for creating COM servers and is generally preferred over MFC.
- Both libraries make it relatively easy to uses standard COM components without re-implementing them yourself.

Active Template Library (ATL)

- A diagram showing the structure of an ATL generated server is shown on the next page.
  - The ATL CComObjectRootEx and CComObject classes, using templatized arguments you provide, implements the IUnknown interface and provides COM aggregation where needed.
  - CComCoClass implements the server's class factory.
  - Classes with the name I…Impl implement standard COM components for you.
  - Your code only needs to:
    - provide template arguments for these classes
    - Create a class that implements your server's functionality, shown as CMyClass, and provide declarations for its interfaces, shown as IMyInterface
  - Note that the ATL wizard will lead you through all this, so the process becomes quick and easy, if you understand how ATL works.

ATL Class Hierachy



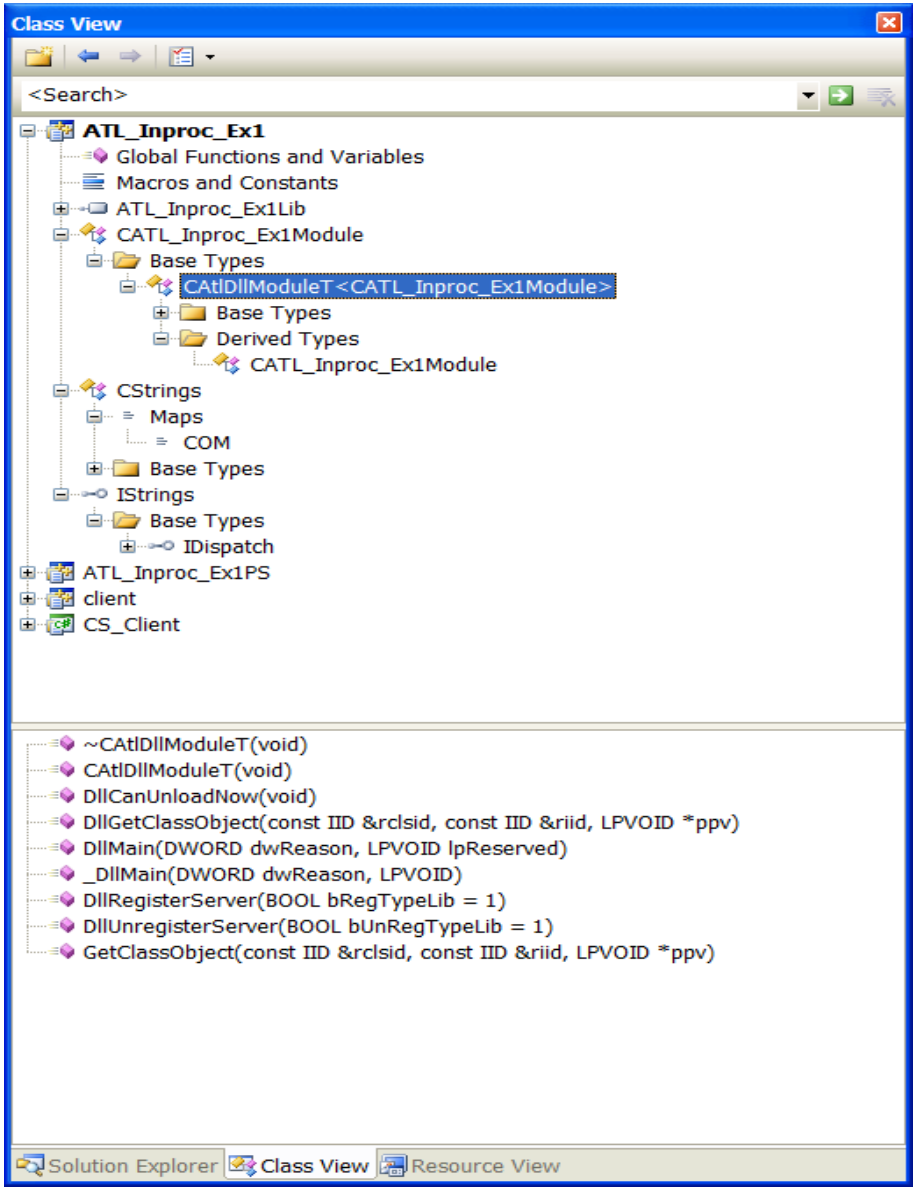
ATL Support for QueryInterface

- QueryInterface is implemented with map macros, much like the macros used in MFC programming:
  - BEGIN_COM_MAP(myclass)
    COM_INTERFACE_ENTRY(IMyInterface)
    COM_INTERFACE_ENTRY(IDispatch) // here for dual interface
    END_COM_MAP()
- If you saw the expansion of these macros after preprocessing you would see a table-based process that supports QueryInterface via enumeration through table elements, one element for each interface.

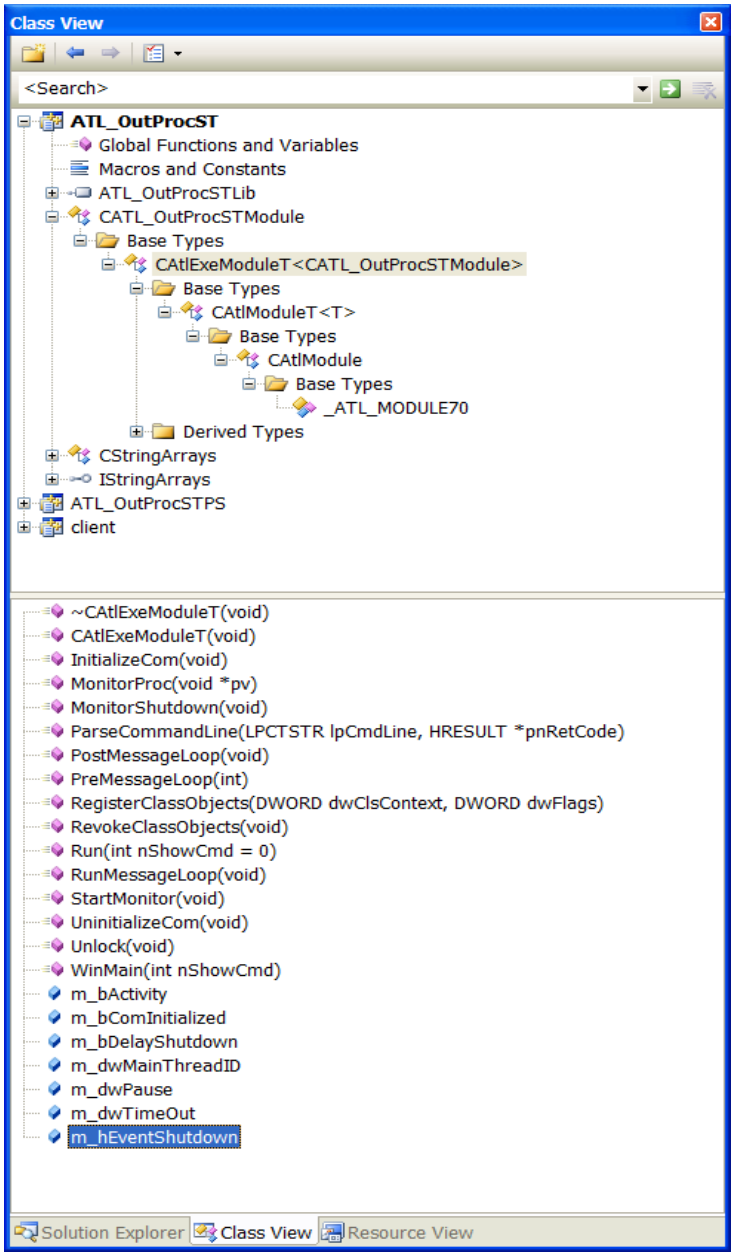
ATL Support for COM Servers

- COM servers provide the following services:
  - Register and unregister all classes in the server and the server and its type library.
  - Provide the Service Control Manager (SCM) access to the class factories hosted by the server.
  - Manage server lifetime.

CAtlDllModuleT



CAtlExeModuleT



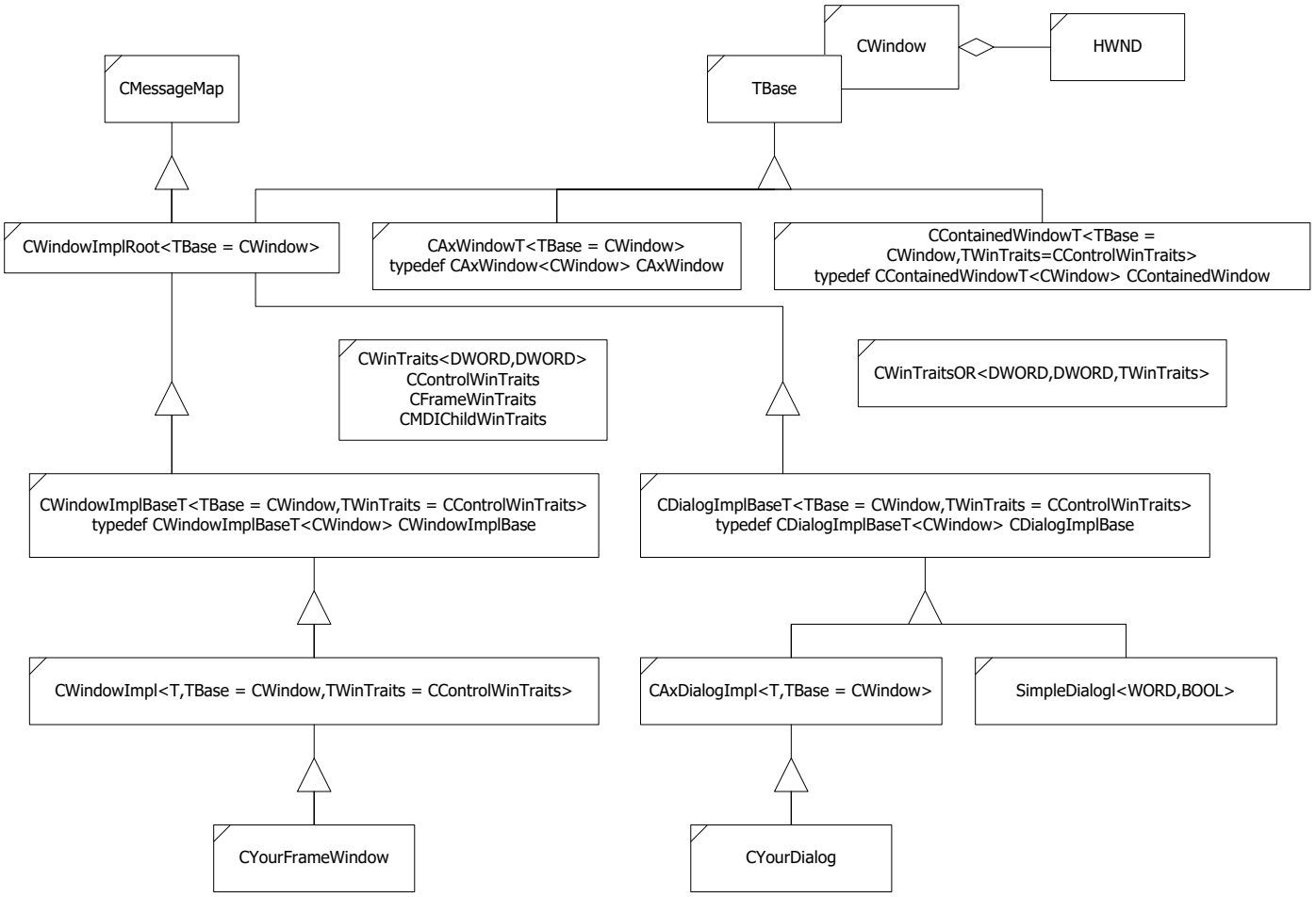
ATL Support for Building Clients

- **CComQIPtr**<class T, const IID* piid = &__uuidof(T)>()
  - Provides instance creation, lifetime management, query for interfaces
- **CComBSTR**(LPCSTR pStr)
  - Wraps BSTRs, providing string manipulation functions, and memory management
- **CComVariant**(Type, VARTYPE)
  - Wraps variants, used in Idispatch and for automation
- **CComSafeArray**<class T, VARTYPE>(count)
  - Wraps arrays of variants

ATL Support for Windows Interfaces

- ATL provides, natively, support for creating both Frame and Dialog windows interfaces.
  - When augmented with the wrappers in the (undocumented and unsupported) atlctrls.h, they provide a very useful framework for creating user interfaces.
  - You have to know some Win32 windows programming, but they provide a lot of help.
- The classes used are shown in the diagram on the next page.
  - If you want to create a highly functional, complex interface, then using WinForms or the MFC framework are good alternatives.
  - However, ATL now provides additional support in the form of an add-on library call the Windows Template Library (WTL), available from sourceforge.
- We may discuss the WTL in more detail later in the semester.

ATL Windows


CWindow
HWND
CMessageMap
TBase
CWindowImplRoot<TBase = CWindow>
CAxWindowT<TBase = CWindow>
typedef CAxWindow<CWindow> CAxWindow
CContainedWindowT<TBase =
CWindow,TWinTraits=CControlWinTraits>
typedef CContainedWindowT<CWindow> CContainedWindow
CWinTraits<DWORD,DWORD>
CControlWinTraits
CFrameWinTraits
CMDIChildWinTraits
CWinTraitsOR<DWORD,DWORD,TWinTraits>
CWindowImplBaseT<TBase = CWindow,TWinTraits = CControlWinTraits>
typedef CWindowImplBaseT<CWindow> CWindowImplBase
CDialogImplBaseT<TBase = CWindow,TWinTraits = CControlWinTraits>
typedef CDialogImplBaseT<CWindow> CDialogImplBase
CWindowImpl<T,TBase = CWindow,TWinTraits = CControlWinTraits>
CAxDialogImpl<T,TBase = CWindow>
SimpleDialogI<WORD,BOOL>
CYourFrameWindow
CYourDialog

Active Template Library

- Microsoft has developed the Active Template Library (ATL) to support reuse of existing designs for many of the standard COM interfaces.
  - IUnknown, IClassFactory, IDispatch, IMarshal
  - ActiveX Controls interfaces
    - event notification
    - properties and property pages

- ATL provides source composition, not binary composition. Once built, however, ATL components, like any other, can be composed as binary objects.

  (we can add a binary control to a window, for example)

Some of the ATL Files

AtlBase.h	Low level type and class definitions
AtlCom.h	COM object management
AtlConv.h	Convert strings to/from unicode
AtlCtl.h, AtlCtl.cpp	ActiveX control support
AtlComCli.h	Smart pointers and BSTR wrapper
AtlSafe.h	SafeArray wrapper
AtlSync.h	Defines classes for locks
Atlimpl.cpp	Implementation of pieces too big to inline
AtlWin.h, AtlWin.cpp	Support for frame and dialog windows
Statreg.h, Statreg.cpp	Support for registry code

End of ATL Presentation