Creating a PGSuper Agent

Written By:	RAB	Creation Date:	10/17/98
Reviewed By:		Last Revised:	5/20/2009
Project:	PGSuper	WI #	<<#>>
		Version	1.0 - Draft

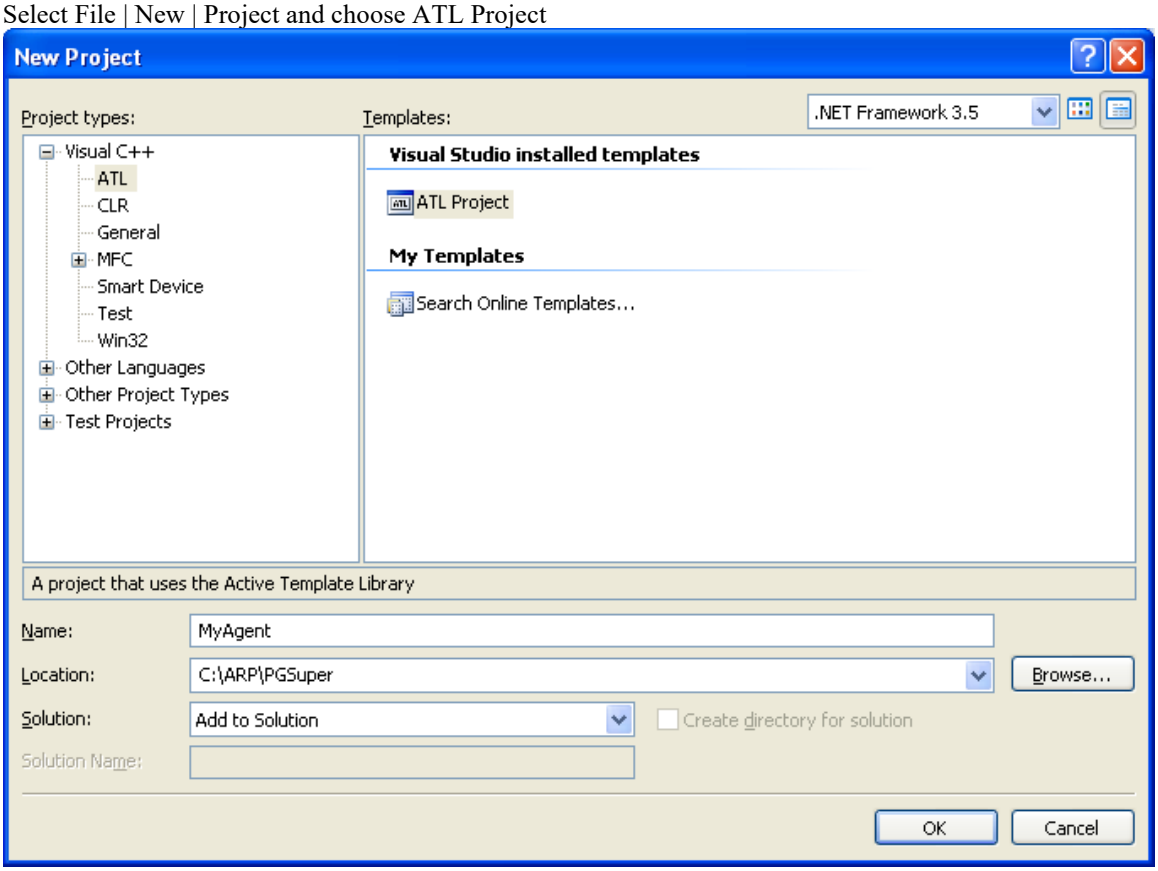
Purpose

The purpose of this work instruction is to describe how to create a new agent for the PGSuper application.

Procedure

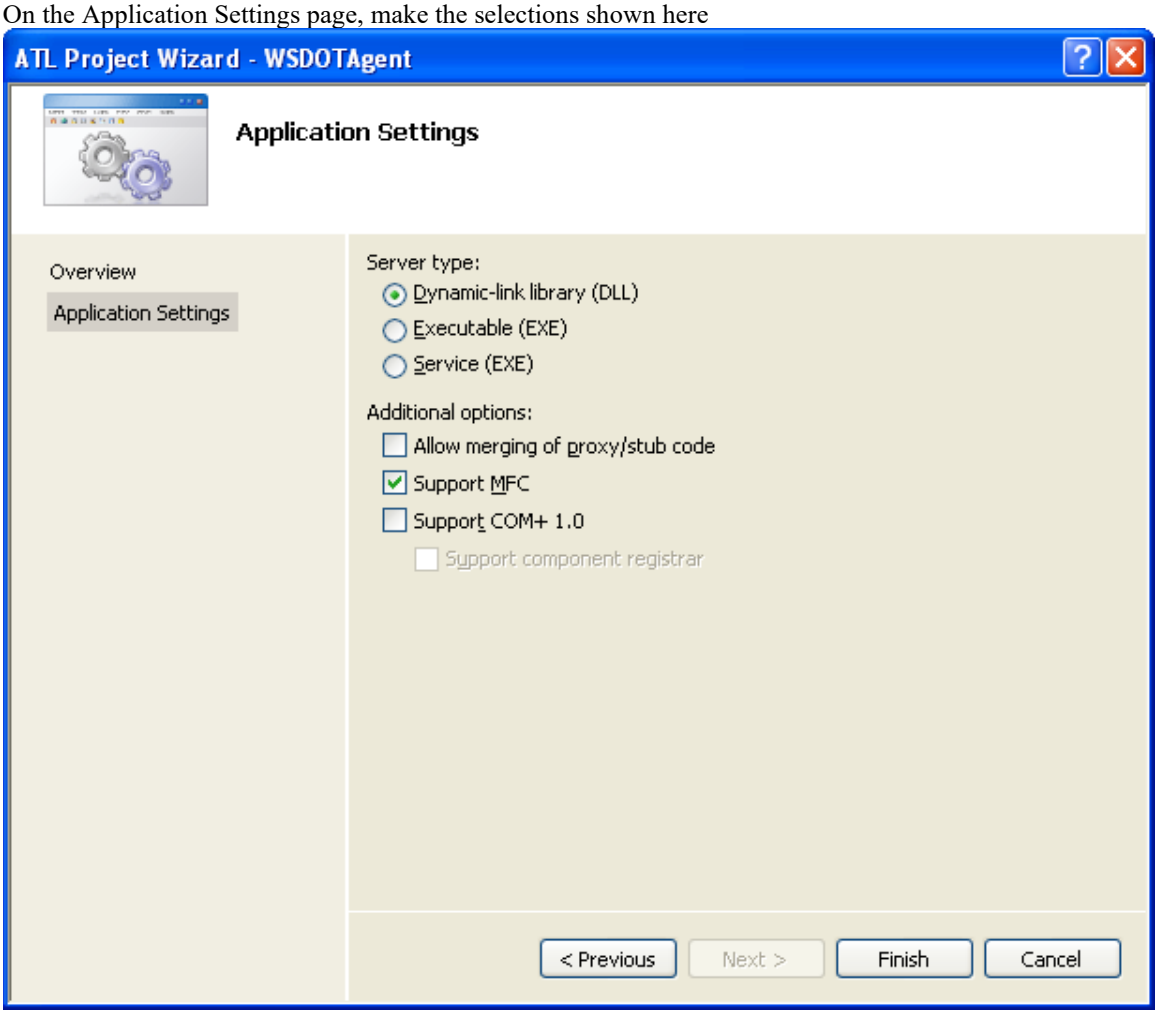
Setting up COM server

1. Open the PGSuper solution in Visual Studio
2. Select File | New | Project and choose ATL Project



3. Enter the agent's name in the Name box and set the path in the Location box. The path is usually C:\ARP\PGSuper\<<AgentName>>
4. Select the OK to start the ATL Project Wizard

5. On the Application Settings page, make the selections shown here

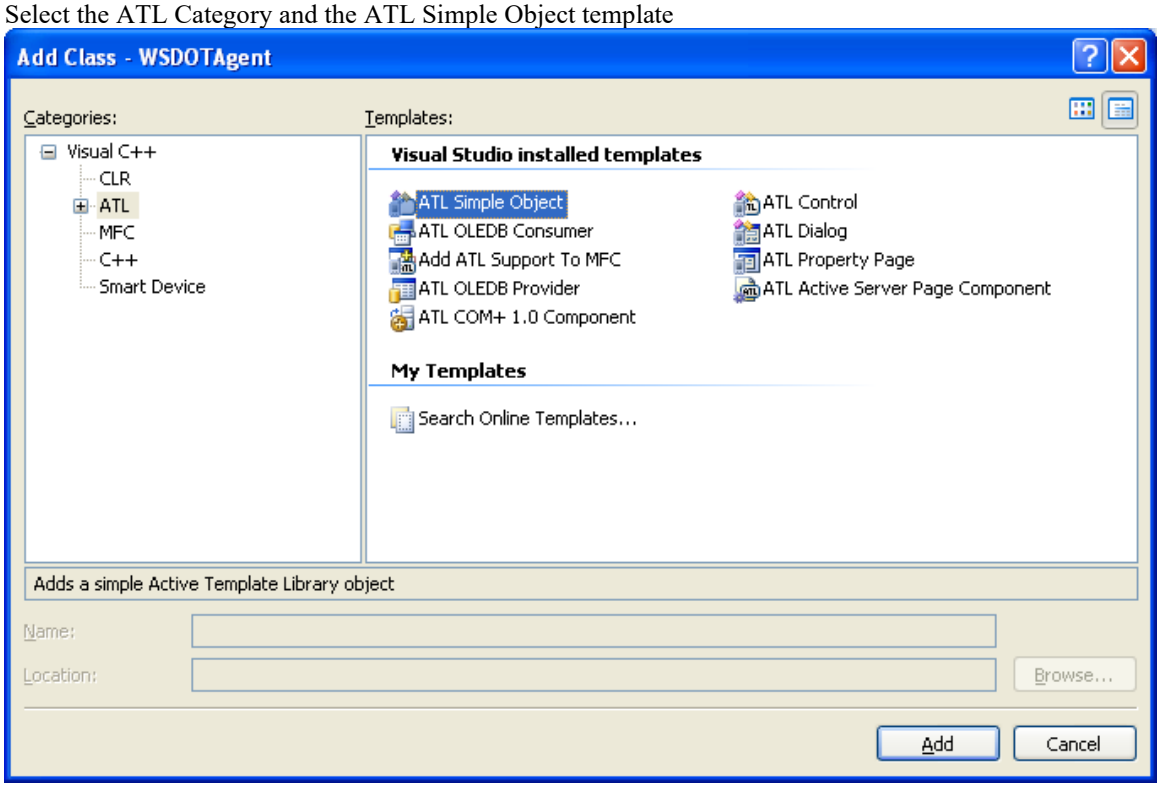


6. After the new project is created and added to the Solution, delete the PS (proxy/stub) project. You don't need it.

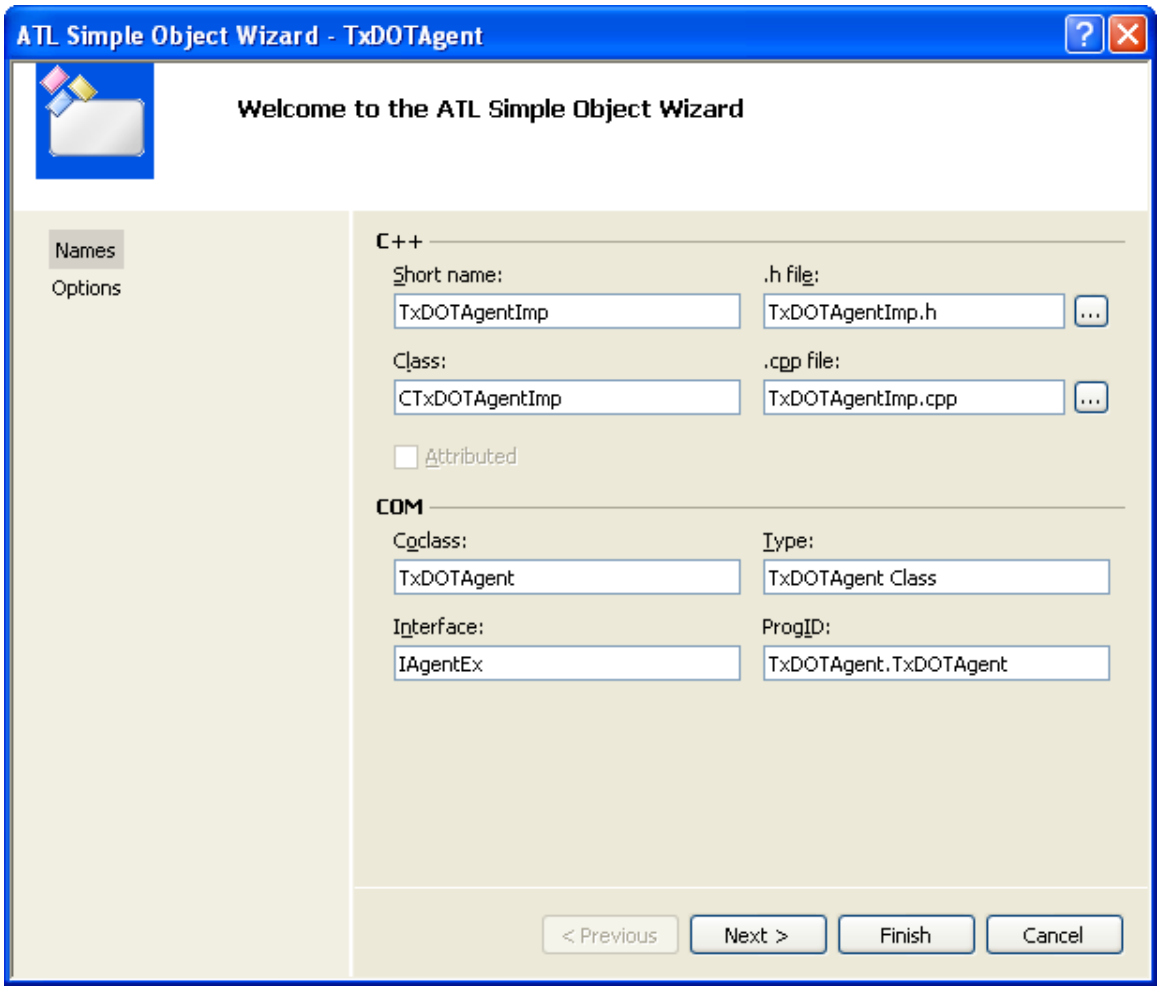
Add support for the IAgentEx interface.

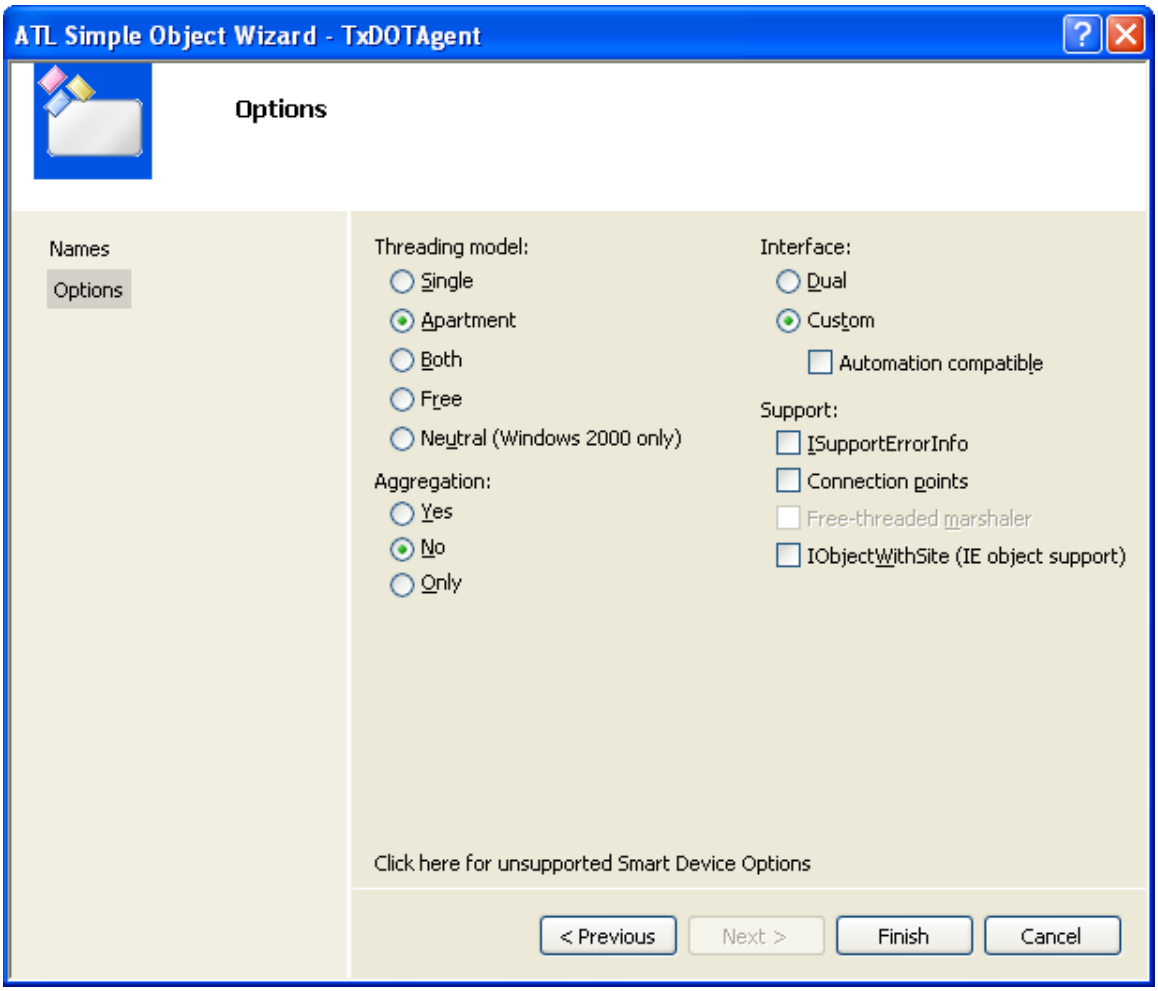
1. In the Solution Explorer window, right click on your agent project and select Add | Class.

2. Select the ATL Category and the ATL Simple Object template



3. Setup the object properties as shown in the two figures below. Substitute your agent's name where ever you see TxDOTAgent. You can optionally add support for ISupportErrorInfo and Connection Points if they apply to your agent. **Carefully review all the settings before you press the OK button. If you mess this up you might as well start over.**


ATL Simple Object Wizard - TxDOTAgent
Welcome to the ATL Simple Object Wizard
Names
Options
C++
Short name:
TxDOTAgentImp
.h file:
TxDOTAgentImp.h
Class:
CTxDOTAgentImp
.cpp file:
TxDOTAgentImp.cpp
Attributed
COM
Coclass:
TxDOTAgent
Type:
TxDOTAgent Class
Interface:
IAgentEx
ProgID:
TxDOTAgent.TxDOTAgent
< Previous
Next >
Finish
Cancel



4. Modify the IDL file
    5. Open the IDL file and remove the interface declared for the object you just created.
    6. Add `import "WBFLCore.idl";`
    7. Delete the IAgentEx interface the Visual Studio created for you
8. Modify the class definition
    9. Replace all occurrences of I*YourAgent* with IAgentEx in *YourAgent*Imp.h and *YourAgent*Imp.cpp. Add the following code to the C*YourAgent*Imp class declaration

```
// IAgentEx
public:
    STDMETHOD(SetBroker)(IBroker* pBroker);
    STDMETHOD(RegInterfaces)();
    STDMETHOD(Init)();
    STDMETHOD(Reset)();
    STDMETHOD(ShutDown)();
    STDMETHOD(Init2)();
    STDMETHOD(GetClassID)(CLSID* pCLSID);
```

6. Add `DECLARE_AGENT_DATA;` to the private part of the class declaration
7. Add the following code to *YourAgent*Imp.cpp (Substitute your class name for CAnalysisAgentImp).

```
////////////////////////////////////////////////////////////////////////
//////
// IAgent
//
```

```
STDMETHODIMP CAnalysisAgentImp::SetBroker(IBroker* pBroker)
{
   AGENT_SET_BROKER(pBroker);
   return S_OK;
}

STDMETHODIMP CAnalysisAgentImp::RegInterfaces()
{
   CComQIPtr<IBrokerInitEx2,&IID_IBrokerInitEx2>
pBrokerInit(m_pBroker);
   // Register interfaces here
   // pBrokerInit->RegInterface( IID_IinterfaceIimpelemnt, this );

    return S_OK;
}

STDMETHODIMP CAnalysisAgentImp::Init()
{
    CREATE_LOGFILE("AnalysisAgent");

    AGENT_INIT;

    return S_OK;
}

STDMETHODIMP CAnalysisAgentImp::Init2()
{
   return S_OK;
}

STDMETHODIMP CAnalysisAgentImp::GetClassID(CLSID* pCLSID)
{
   *pCLSID = CLSID_AnalysisAgent;
   return S_OK;
}

STDMETHODIMP CAnalysisAgentImp::Reset()
{
   return S_OK;
}

STDMETHODIMP CAnalysisAgentImp::ShutDown()
{
// Put all other shutdown code before this line.
// This macro sets the broker to NULL.
   AGENT_CLEAR_INTERFACE_CACHE;
   return S_OK;
}
```

8. By default, the agent will use a local interface caching scheme. If you do not want to use local interface caching define the `NO_INTERFACE_CACHE` macro in the project C++ Preprocessor settings

PGSuper Agent component category registration

1. Your agent must register itself as a member of the PGSuperAgent component category. In the Agent.cpp file, replace the generated DllRegisterServer and DllUnregisterServer functions with the following:

```
////////////////////////////////////////////////////////////////////////////
// DllRegisterServer - Adds entries to the system registry
STDAPI DllRegisterServer(void)
{
    // registers object, typelib and all interfaces in typelib
    HRESULT hr = _Module.RegisterServer(FALSE);
   if ( FAILED(hr) )
      return hr;
   return sysComCatMgr::RegWithCategory(CLSID_AnalysisAgent,CATID_PGSuperAgent,true);
}
////////////////////////////////////////////////////////////////////////////
// DllUnregisterServer - Removes entries from the system registry
STDAPI DllUnregisterServer(void)
{
   sysComCatMgr::RegWithCategory(CLSID_AnalysisAgent,CATID_PGSuperAgent,false);
    _Module.UnregisterServer();
    return S_OK;
}
```

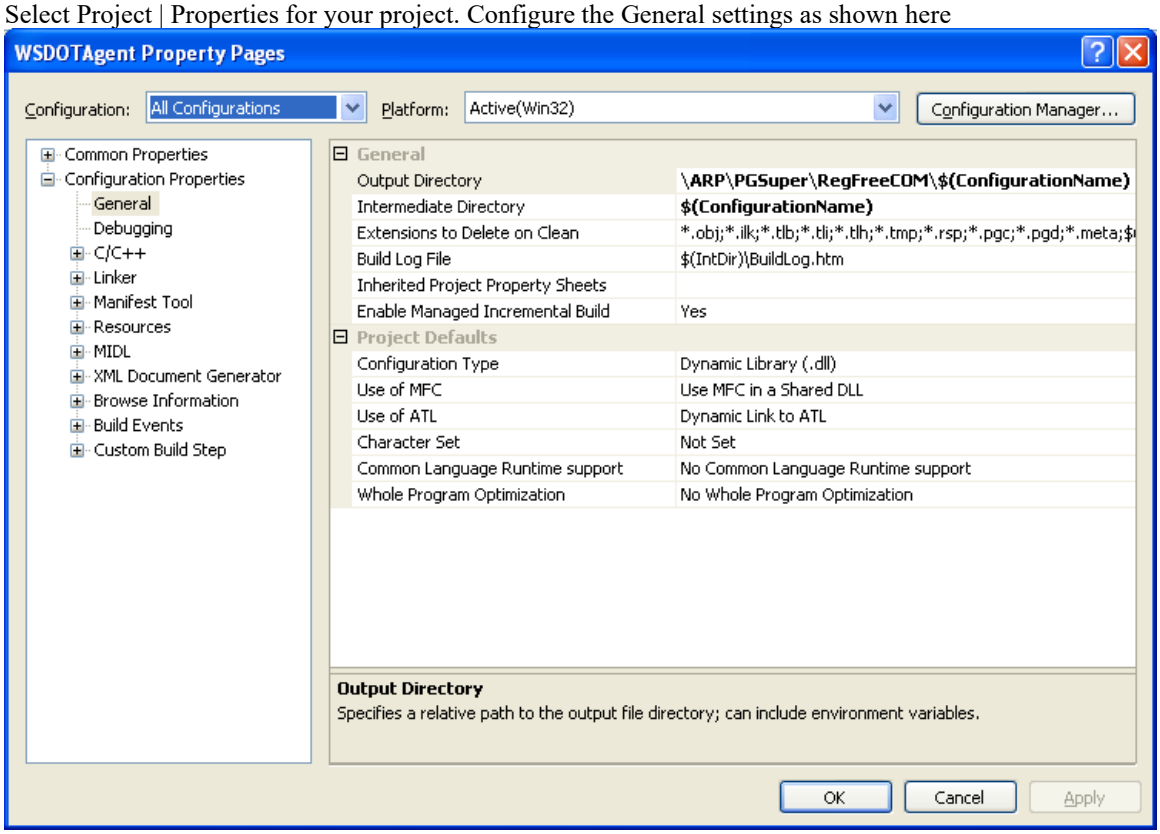
2. Open the resource file as text and remove the following lines. The tlb file name will be named after your agent.

```
////////////////////////////////////////////////////////////////////////////
//
// Generated from the TEXTINCLUDE 3 resource.
//
1 TYPELIB "AnalysisAgent.tlb"
```

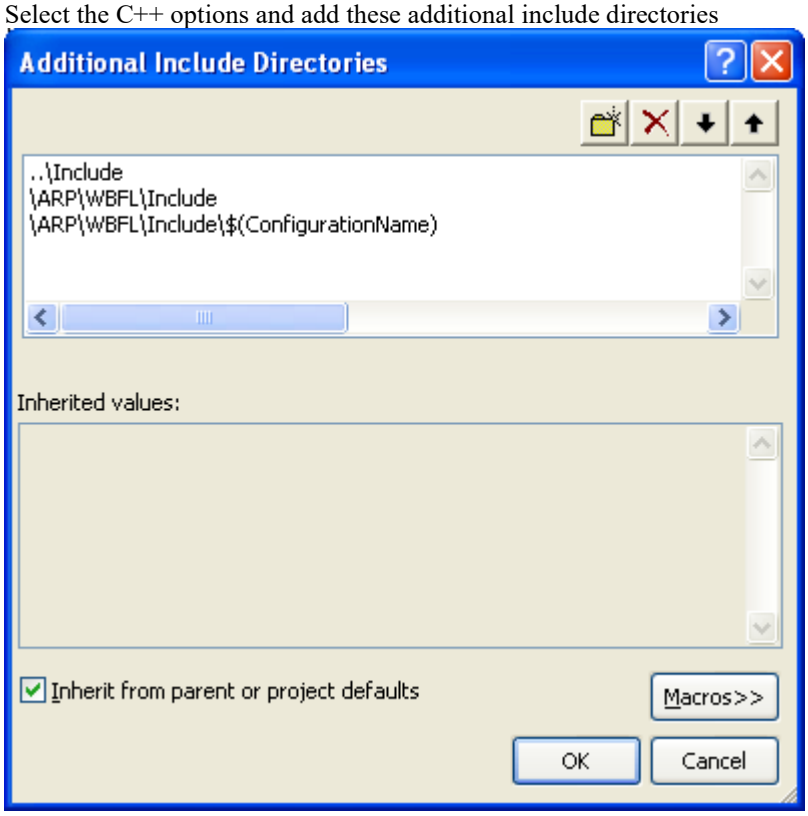
3. Remove the following line from *YourAgent*.cpp
   #include "AnalysisAgent_i.c"

Configure Project Settings

1. Select Project | Properties for your project. Configure the General settings as shown here



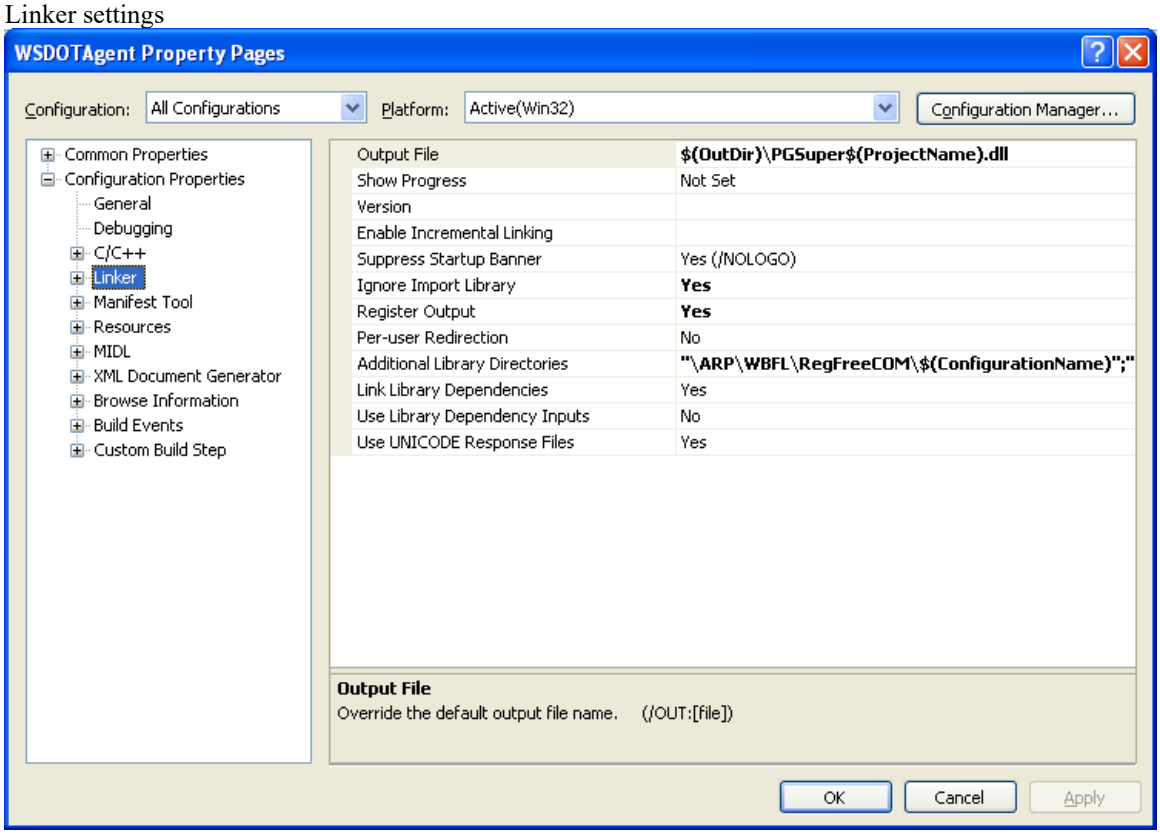
2. Select the C++ options and add these additional include directories



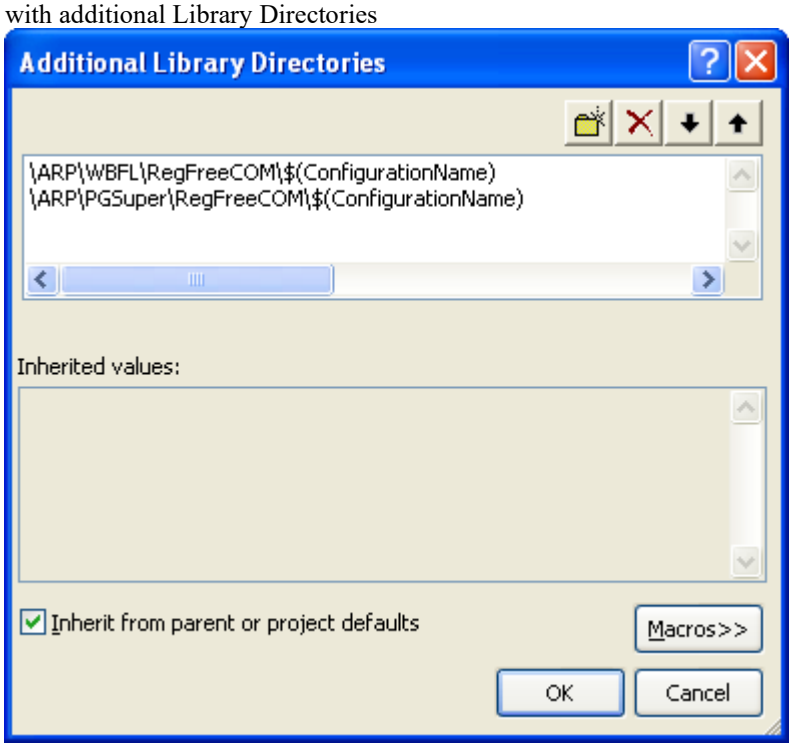
and add the following pre-processor definitions

```
ENABLE_LOGGING
_HAS_ITERATOR_DEBUGGING=0
_SECURE_SCL=0
```

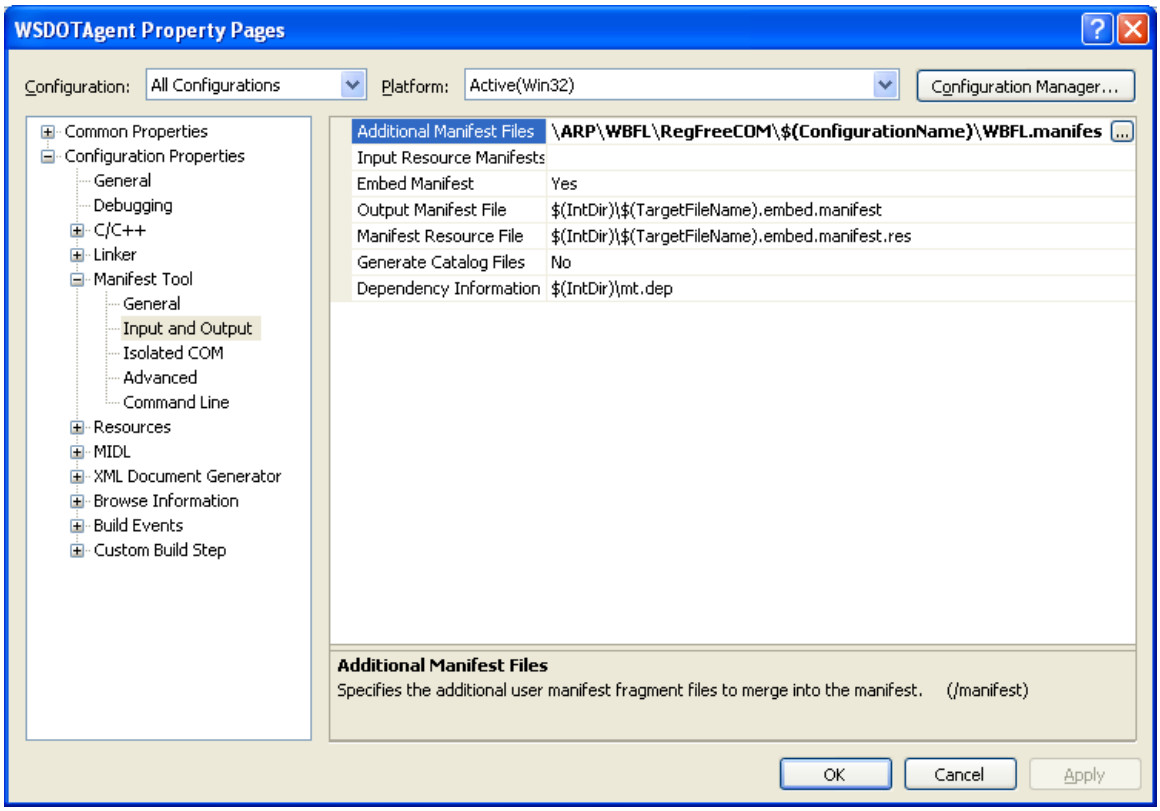
3. Linker settings



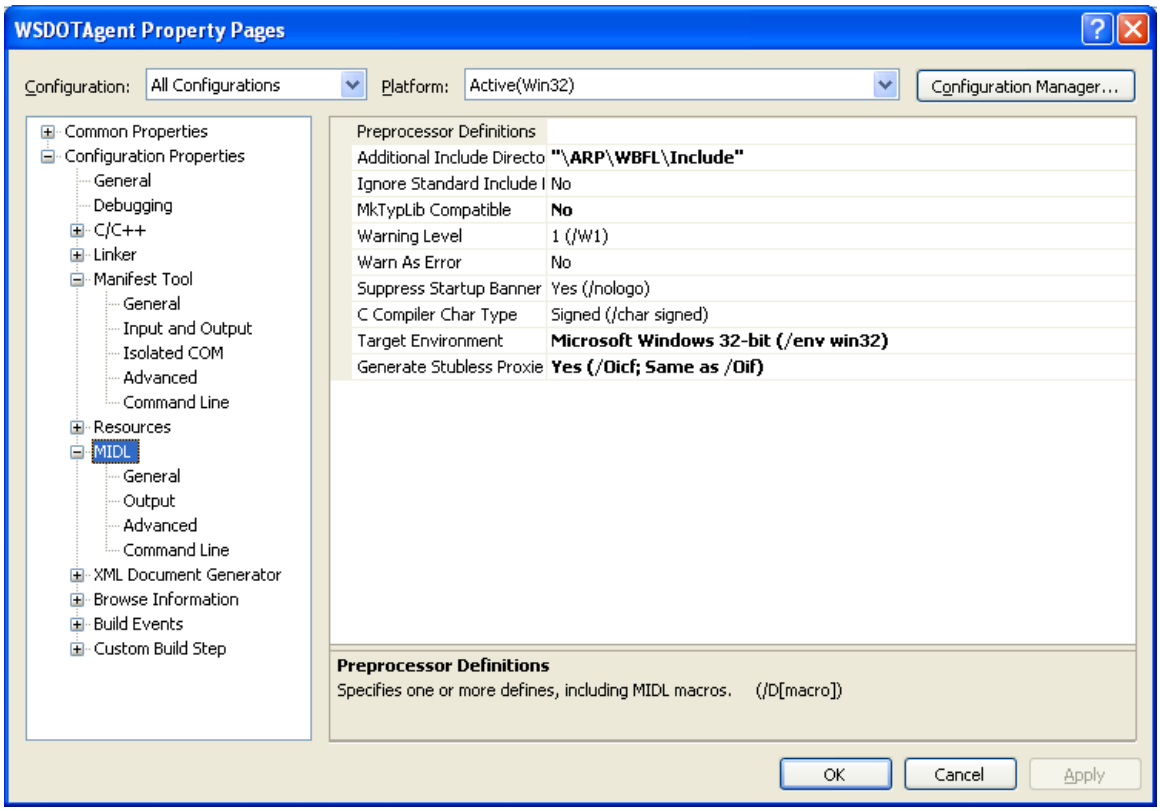
with additional Library Directories



4. Manifest Tool



5. MIDL



Your project should be ready to build.

If the linker cannot resolve IID_IStatusCenter, add
```
#include <IFace\StatusCenter.h>
```
To agent.cpp file below `#include <initguid.h>`