OpenRTM-aist (C++) - バグ #2138

Windows版でComponentObserverが動作しない

2011/05/25 18:53 - n-ando

ステータス:	終了	**開始日:**	2011/05/25
優先度:	通常	**期日:**	
担当者:	n-ando	**進捗率:**	100%
カテゴリ:		**予定工数:**	0.00時間
対象バージョン:			

説明

Windows版でComponentObserverが動作しない

関係しているリビジョン

リビジョン 2142 - 2011/05/26 23:23 - n-ando

The problem that ComponentObserver cannot be used in Windows envrionment
has been fixed. IDL compiler option and definition has been changed.
SdoServiceConsumerBase's virtual dtor has been added. refs #2138

リビジョン 2142 - 2011/05/26 23:23 - n-ando

The problem that ComponentObserver cannot be used in Windows envrionment
has been fixed. IDL compiler option and definition has been changed.
SdoServiceConsumerBase's virtual dtor has been added. refs #2138

リビジョン 2142 - 2011/05/26 23:23 - n-ando

The problem that ComponentObserver cannot be used in Windows envrionment
has been fixed. IDL compiler option and definition has been changed.
SdoServiceConsumerBase's virtual dtor has been added. refs #2138

履歴

#1 - 2011/05/25 18:57 - n-ando

- 進捗率 を 0 から 30 に変更

- omniORBのスタブ生成、コンパイル時に、シンボルの扱いに気を付ける必要がある。

http://omniorb.sourceforge.net/omni41/omniORB/omniORB012.html

ComponentObserver.idl はSDOPackage.idl
をインクルードしているので、SDOPackageのシンボルに対してはUSE_stub_in_nt_dllを有効にしてシンボル等をimportする必要がある。一方Comp
onentObserver.idlで定義されているシンボルはUSE_stub_in_nt_dllを無効にする必要がある。

- omniidl のオプションに -Wbdll_includes を付ける。
- INCLUDED_stub_in_nt_dll をプリプロセッサで定義する

これにより、シンボルの参照が正しく行われるようになった。

#2 - 2011/05/25 18:59 - n-ando

SdoServiceConsumerFactory
はシングルトンのはずなのに、ConmponentObserverConsumer.dll内とRTC110d.dll内で異なるインスタンスを返しているようだ。

#3 - 2011/05/25 21:31 - n-ando

- coil::Factory, coil::Singleton にDLL_PLUGINマクロ(dllexport/dllimpoerを切り替える)を追加
- explicit instantiation をする以下の1行をSdoServiceAdminに追加

```
template class DLL_PLUGIN ::coil::GlobalFactory< ::RTC::SdoServiceConsumerBase >;
```

効果なし。

dumpbin /EXPORTS rtc110d.dll ¦grep addFactory¦grep SdoService

とやっても何もなし。Factory関係の関数がexportされていない。

#4 - 2011/05/25 21:35 - n-ando

Factoryの実体化のために実際にSdoServiceConsumerのファクトリを使用するダミーコードを追加。

```
class DummySdoService
        : public RTC::SdoServiceConsumerBase
{
public:
        virtual bool init(RTC::RTObject_impl& rtobj,
                  const SDOPackage::ServiceProfile& profile){ return true;}
        virtual bool reinit(const SDOPackage::ServiceProfile& profile) {return true;}
        virtual const SDOPackage::ServiceProfile& getProfile() const { return SDOPackage::ServiceProfile(); }
        virtual void finalize() {}
};
void DummySdoServiceInit()
{
        RTC::SdoServiceConsumerFactory& factory
            = RTC::SdoServiceConsumerFactory::instance();
        factory.addFactory("dummy",
                                              ::coil::Creator< ::RTC::SdoServiceConsumerBase,
                                              DummySdoService>,
                                              ::coil::Destructor< ::RTC::SdoServiceConsumerBase,
                                              DummySdoService>);
        factory.removeFactory("dummy");
        std::cout << "DummySdoServiceInit()" << std::endl;
}
```

とりあえず、関数はexportsされている。

```
H:\work\trunk\OpenRTM-aist\win32\OpenRTM-aist\bin>dumpbin /EXPORTS RTC110d.dll ¦
  grep addFactory¦grep SdoService
              8489  2128  0016FA67  ?addFactory@?$Factory@VSdoServiceConsumerBase@RTC@@V?$
basic_string@DU?$char_traits@D@std@@V?$allocator@D@2@@std@@U?$less@V?$basic_stri
ng@DU?$char_traits@D@std@@V?$allocator@D@2@@std@@@4@P6APAV12@XZP6AXAAPAV12@@Z@co
il@@QAE?AW4ReturnCode@12@ABV?$basic_string@DU?$char_traits@D@std@@V?$allocator@D
@2@@std@@P6APAVSdoServiceConsumerBase@RTC@@XZP6AXAAPAV67@@Z@Z = @ILT+31330(?addF
actory@?$Factory@VSdoServiceConsumerBase@RTC@@V?$basic_string@DU?$char_traits@D@
std@@V?$allocator@D@2@@std@@U?$less@V?$basic_string@DU?$char_traits@D@std@@V?$al
locator@D@2@@std@@@4@P6APAV12@XZP6AXAAPAV12@@Z@coil@@QAE?AW4ReturnCode@12@ABV?$b
asic_string@DU?$char_traits@D@std@@V?$allocator@D@2@@std@@P6APAVSdoServiceConsum
erBase@RTC@@XZP6AXAAPAV67@@Z@Z)
```

#5 - 2011/05/25 23:16 - n-ando

- 進捗率 を 30 から 50 に変更

実験

```
void DummySdoServiceInit()
    {
        RTC::SdoServiceConsumerFactory& factory
            = RTC::SdoServiceConsumerFactory::instance();
//  ここから
        factory.addFactory("dummy",
                                              ::coil::Creator< ::RTC::SdoServiceConsumerBase,
                                              DummySdoService>,
                                              ::coil::Destructor< ::RTC::SdoServiceConsumerBase,
                                              DummySdoService>);
        factory.removeFactory("dummy");
        std::cout << "DummySdoServiceInit()" << std::endl;
//  ここまで
}
```

- ここから・ここまでをコメントイン: addfactoryのExportシンボルあり
- ここから・ここまでをコメントアウト: addfactoryのExportシンボルなし

使用した関数のみ実体化されそれらがexportされる。

また、template class DLL_PLUGIN ::coil::GlobalFactory< ::RTC::SdoServiceConsumerBase >;
を削除したが、関数はExportされる。RTC.DLLはdumpbinでシンボルを引っ張り出して

すべてのシンボルをexportしているので関係ないみたい。

#6 - 2011/05/26 23:26 - n-ando

- ステータス を 新規 から 終了 に変更

- 進捗率 を 50 から 100 に変更

- DummySdoServiceInit() は不要なことが判明
- INCLUDE_stub_nt_dll と -Wbdll_include を追加
- SdoServiceConsumerBaseのvirtualデストラクタが抜けていたため、オブジェクトが解体されずタイマー等が動き続け落ちていた(Observer削除時)
- explicit instantiation は必要、EXTERNも追加しヘッダに記載

```
EXTERN template class DLL_PLUGIN ::coil::GlobalFactory< ::RTC::SdoServiceConsumerBase >;
```

以上でほぼ落ちなくなったので解決とする。

#7 - 2015/07/10 23:34 - n-ando

- 対象バージョン を削除 (RELEASE_1_1_0)