OpenRTM-aist (Python) - 調査 #3735

実行されていないソースコードの実効行についての調査

2016/11/28 15:09 - n-miyamoto

ステータス:	終了	**開始日:**	2016/11/28
優先度:	通常	**期日:**	
担当者:	n-miyamoto	**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

テストで実行されていないソースコードの実効行を調べる。

履歴

#1 - 2016/11/28 17:21 - n-miyamoto

- 進捗率 を 0 から 10 に変更

- 関数にpassやreturnしか記述されておらず、実質的にテストする意味がない行

```
def __call__(self, value):
    pass
```

- 処理がOSで分岐しており、網羅が不可能な行

```
if platform.system() == "Windows":
```

- 処理がPythonのバージョンで分岐しており、網羅が不可能な行

```
if sys.version_info[0:3] >= (2, 4, 0):
```

- デストラクタは必ず呼び出される保証がないため、実行されていない行が存在する
  - デストラクタを排除して、exit関数やfinalize関数を実装する
- CORBA関連の例外
  - 発生させた場合に後のテストが続行できなくなる可能性がある
    - 以下の場合、例外を無理やり発生させるとORBの終了処理が正しく行われないため、連続してテストを実行することができなくなる

```
try:
    self._orb.shutdown(True)
    self._orb.destroy()
    self._rtcout.RTC_DEBUG("ORB was shutdown.")
    self._orb = CORBA.Object._nil
except CORBA.SystemException, ex:
    self._rtcout.RTC_ERROR("Caught CORBA::SystemException during ORB shutdown.")
    self._rtcout.RTC_ERROR(OpenRTM_aist.Logger.print_exception())
except:
    self._rtcout.RTC_ERROR("Caught unknown exception during ORB shutdown.")
    self._rtcout.RTC_ERROR(OpenRTM_aist.Logger.print_exception())
```

- 未実装の関数

```
def get_monitoring(self):
    self._rtcout.RTC_TRACE("get_monitoring()")
    raise SDOPackage.InterfaceNotImplemented("Exception: get_monitoring")
    return SDOPackage.Monitoring._nil
```

- RTObjectStateMachineの有限状態遷移型RTC、マルチモード型RTCに関する部分

```
def setFsmParticipantAction(self, comp):
    self._fsmVar = comp._narrow(RTC.FsmParticipantAction)
```

```
        if not CORBA.is_nil(self._fsmVar):
            self._fsm = True
        return

    def setMultiModeComponentAction(self, comp):
        self._modeVar = comp._narrow(RTC.MultiModeComponentAction)
        if not CORBA.is_nil(self._modeVar):
            self._mode = True
        return
```

- 例外の発生する条件が不明な行

```
    try:
        OpenRTM_aist.CORBA_SeqUtil.erase(self._memberList, index)
        return True
    except:
        self.__rtcout.RTC_ERROR("unknown exception")
        raise SDOPackage.InternalError("remove_member(): Not found.")
```

- 変数を返す、格納する等単純な処理だけしか実行しないにもかかわらず例外処理のコードがある行

```
    def get_sdo_type(self):
        self._rtcout.RTC_TRACE("get_sdo_type()")
        try:
            return self._profile.description
        except:
            self._rtcout.RTC_ERROR(OpenRTM_aist.Logger.print_exception())
            raise SDOPackage.InternalError("get_sdo_type()")
        return ""
```

#2 - 2016/11/28 18:16 - n-miyamoto

- 進捗率 を 10 から 20 に変更

以下のソースコードではデストラクタが一度も呼び出されていないクラスが存在するため修正を行う。

- ListenerHolder.py
- RTObjectStateMachine.py
- Manager.py
- PortConnectListener.py
- LocalServiceAdmin.py
- PeriodicECSharedComposite.py

以下のソースコードでは上記の問題以外で網羅できていない行が存在するためテストを追加する。

- CPUAffinity.py
- Process.py
- ManagerServant.py
- InPortCorbaCdrConsumer.py
- CorbaPort.py
- CorbaNaming.py
- NamingManager.py
- ManagerConfig.py
- OutPortConnector.py
- PortBase.py
- InPortBase.py
- PeriodicExecutionContext.py
- OutPortCorbaCdrConsumer.py
- OutPortBase.py
- ConfigurationListener.py
- SdoServiceAdmin.py
- ExecutionContextWorker.py
- ModuleManager.py
- SharedMemory.py
- RTObject.py
- CorbaConsumer.py
- SystemLogger.py
- OutPortPullConnector.py
- OutPortDirectConsumer.py
- InPort.py
- Properties.py

- PortAdmin.py
- OutPortSHMProvider.py
- ConfigAdmin.py
- OutPortPushConnector.py
- ExecutionContextBase.py
- InPortProvider.py
- OutPortCorbaCdrProvider.py
- OutPortProvider.py
- ExecutionContextProfile.py
- GlobalFactory.py
- OutPort.py
- RingBuffer.py
- StringUtil.py
- InPortPushConnector.py
- InPortCorbaCdrProvider.py
- StateMachine.py
- CORBA_RTCUtil.py
- InPortPullConnector.py
- Timer.py
- PublisherNew.py
- PeriodicTask.py
- PublisherPeriodic.py
- ExtTrigExecutionContext.py
- OpenHRPExecutionContext.py

#3 - 2017/02/01 09:18 - n-miyamoto

- ステータス を 新規 から 解決 に変更

- 進捗率 を 20 から 100 に変更

#4 - 2017/03/19 08:15 - n-ando

- ステータス を 解決 から 終了 に変更