OpenRTM-aist (Python) - 機能 #3412

⑬　名前付けポリシー機能の拡張

2015/12/22 09:46 - n-ando

ステータス:	終了	**開始日:**	2015/12/22
優先度:	通常	**期日:**	2016/03/25
担当者:	miyamoto	**進捗率:**	100%
カテゴリ:		**予定工数:**	30.00時間
対象バージョン:	RELEASE_1_2_0		

説明

「名前ベースのコンポーネント」機能を実現するために、異なるプロセスで起動した同一コンポーネントでも、マスターの配下にある他のコンポーネントの名前を見て、インスタンス番号を付ける機能を実装する。また、コンポーネント自身もINS機能を利用して、カテゴリ名、インスタンスIDをObjectIdとして起動するように、新たなPOA上でのアクティブ化、名前付けなどを実装すること。

関係しているリビジョン

リビジョン 658 - 2016/02/25 14:27 - miyamoto

[incompat,new_func,new_file,->RELENG_1_2] add NumberingPolicyBase.py, NodeNumberingPolicy.py, and NamingServiceNumberingPolicy.py. refs #3412

リビジョン 693 - 2016/03/12 04:16 - miyamoto

[compat,->RELENG_1_2] refs #3412

履歴

#1 - 2016/01/14 16:22 - miyamoto

- 期日 を 2016/03/25 にセット

- 担当者 を miyamoto にセット

- 対象バージョン を RELEASE_1_2_0 にセット

- 予定工数 を 30.00時間 にセット

#2 - 2016/02/18 15:11 - n-ando

- NamingPolicyBase基底クラスを用意する
- Policyのファクトリを用意する
  typedef ::coil::GlobalFactory<NamingPolicyBase> NamingPolicyFactory;

- いくつかのNamingPolicyを作る
  - プロセス間でインスタンス番号をシーケンシャルに振るポリシー
  - ネームサービス上でユニークになるようにシーケンシャルに番号を振るポリシー

- rtc.conf に manager.components.naming_policy:
  - ex: process_unique: プロセス内でユニークになるようにインスタンス名にシーケンシャルに番号をふる （現在のデフォルト）
  - ex: node_unique: ノード内でユニークになるようにインスタンス名にシーケンシャルに番号をふる
  - ex: ns_unique: ネーミングサービス上でユニークになるようにry

- Manager:registerFactory()
  内で、コンポーネントのファクトリを登録する際に、rtc.confで指定されたポリシーをインスタンス化し、FactoryCXX に渡す。

#3 - 2016/02/25 14:25 - miyamoto

- ファイル test_NumberingPolicy_node.py を追加

- ファイル test_NumberingPolicy_ns.py を追加

- 進捗率 を 0 から 30 に変更

NumberingPolicyBase.py、NodeNumberingPolicy.py、NamingServiceNumberingPolicy.pyを追加した。

NumberingPolicyBase.pyにはNumberingPolicyBaseクラス、NumberingPolicyFactoryクラスを定義している。
これに伴いNumberingPolicyクラスはNumberingPolicyBaseクラスを継承するように変更した。

Managerクラスのregister Factory関数でFactoryPythonクラスの初期化時にmanager.components.naming_policyで指定したポリシーを渡すように変更した。

manager.components.naming_policyはprocess_unique、node_unique、ns_uniqueから選択可能であり、デフォルトではprocess_uniqueを設定してある。

NodeNumberingPolicyクラスではmanager.components.naming_policyをnode_uniqueに指定したときのポリシーを定義してある。
マスターマネージャ、スレーブマネージャからRTCを検索して、名前がユニークになるように番号を振る。
起動したマネージャがスレーブマネージャの場合は登録されているマスターマネージャからスレーブマネージャのリストを取得してRTCの検索を行う。登録されているマスターマネージャが存在しない場合はプロセス内で起動したマネージャでのみ検索を行う。
RTCの存在確認には#3411で実装したManager.idlのget_components_by_nameを使用する。

以下のコードでテストを行った。
このテスト用コードでは2つのプロセスでRTCを起動し、それぞれマスターマネージャ、スレーブマネージャが起動するようにしている。

```
self.queue = multiprocessing.Queue()
self.outport_process = multiprocessing.Process(target=runTestComp2, args=(self.queue,))
self.outport_process.start()
time.sleep(1)
sys.argv.extend(['-o','manager.components.naming_policy:node_unique'])
self.manager = OpenRTM_aist.Manager.init(sys.argv)
self.manager.setModuleInitProc(TestComp1ModuleInit)
self.manager.activateManager()

comp = self.manager.getComponent("TestComp11")
self.assertTrue(comp is not None)
```

NamingServiceNumberingPolicyクラスではmanager.components.naming_policyをns_uniqueに指定したときのポリシーを定義してある。
ネーミングサービスから一致するインスタンス名のRTCを検索して、ユニークな名前を付けるように番号を振る。

以下のコードでテストを行った。

```
self.queue = multiprocessing.Queue()
self.outport_process = multiprocessing.Process(target=runTestComp2, args=(self.queue,))
self.outport_process.start()
time.sleep(1)
sys.argv.extend(['-o','manager.components.naming_policy:ns_unique'])
self.manager = OpenRTM_aist.Manager.init(sys.argv)
self.manager.setModuleInitProc(TestComp1ModuleInit)
self.manager.activateManager()

comp = self.manager.getComponent("TestComp11")
self.assertTrue(comp is not None)
```

#4 - 2016/03/17 11:03 - miyamoto

- 進捗率 を 30 から 100 に変更

#5 - 2017/04/10 08:58 - n-ando

- 進捗率 を 100 から 90 に変更

これちゃんと動いてる？＞NodeUniquePolicy

#6 - 2018/01/19 18:33 - n-miyamoto

- ステータス を 新規 から 解決 に変更

- 進捗率 を 90 から 100 に変更

#7 - 2018/09/13 09:21 - n-miyamoto

- ステータス を 解決 から 終了 に変更

ファイル

test_NumberingPolicy_node.py	2.58 KB	2016/02/25	miyamoto
test_NumberingPolicy_ns.py	2.57 KB	2016/02/25	miyamoto