OpenRTM-aist (Python) - 機能 #3226

同一プロセスのコンポーネント間のデータポート接続で、データ転送をダイレクトに行うようにする

2015/06/17 00:42 - n-ando

ステータス:	終了	開始日:	2015/06/17
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

同一プロセスに存在する複数のコンポーネント間のデータ転送においては、

- マーシャリング
- バッファリング
- アンマーシャリング

が行われているが、実際には単にOutPortからInPortの変数にデータを書き込むことができれば効率的である。
複数のRTCが並列に動作していたり、異なる周期で動作する場合などはバッファリングも必要になるケースも考えられるが、例えばECを共有しそれらのコンポーネントの実行がシーケンシャルであることが保証される場合、コンポーネントの実行・データ転送効率を上げる意味でも、OutPort→InPortで直接変数に書き込む機能があればよいのでこれを実装する。

履歴

#1 - 2015/06/17 22:12 - n-ando

仕様

- ConnectorProfile.properties["dataport.outport.direct_dataput.disable"] != YES の時、または当該プロパティーが設定されていない場合にDirect data put モードにする
- Direct
  data-putモードでは、同一プロセス内のInPortとOutPortのpush接続時に、corba_cdrのインターフェースを通すことなく、InPortの変数にOutPortから直接データを書き込む
- Direct data putモードでは、データが変数に書き込まれた場合にInPort::isNew()=true,
  InPort::isEmpty()=falseとなり、InPort::read()がよばれた後は、変数およびバッファの状態によりInPort::isNew(),isEmpty()の状態が変化する

実装方法

- InPort
  - 変数書き込みフラグとミューテックスを追加
  - void write(const DataType&) 関数（OutPortConnectorからコール）を用意し、変数書き込み、フラグtrue代入をロック下で行う
  - 変数書き込みフラグの状態に応じてisNew(),isEmpty()を修正
  - read()関数内で変数書き込みフラグがtrueの場合のロジックを追加
  - InPort変数への参照をミューテックスで保護

- OutPortConnector
  - InPortのサーバントを与える setInPort() 関数を追加
  - write()関数内にInPortのwrite()関数を呼ぶ static_cast<InPort<DataType>*>(m_directInPort)->write(data); コードを追加

- OutPortBase
  - コネクタープロファイルのプロパティー (dataport.outport.)direct_dataput.disable
    を取得。disable=YESなら通常の処理、でなければ以下の処理を行う
  - Push接続時のコネクタ生成ロジック内で、InPortのローカルサーバントを取得する
  - 取得できない場合は、リモートオブジェクトなのでそのまま継続、取得できる場合は、コネクタにInPortのサーバントを渡す(setLocalInPort())
  - サーバント取得ロジック： InPortBase* inport = dynamic_cast<InPortBase*>(poa->reference_to_servant(obj));

#2 - 2017/12/26 17:16 - n-miyamoto

- **ステータス を 新規 から 終了 に変更**
- **担当者 を n-miyamoto にセット**
- **進捗率 を 0 から 100 に変更**