OpenRTM-aist (C++) - 機能 #1055

複数NICを持つノードのCORBAオブジェクトのIORの問題

2010/01/10 00:07 - n-ando

ステータス:	終了	**開始日:**	2010/01/10
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

複数のネットワークインターフェースを持つノードが、ネームサーバにオブジェクトリファレンスを登録する際、通常ローカルループバックを除くインターフェースのうち、第1番目のインターフェースのアドレスのみを含むIORをネームサーバに登録する。
これにより、意図しないアドレスがIORに付与され、結果として通信のできないオブジェクトリファレンスとなる場合がある。
これを回避する方法として、corba.endpoint
オプションが利用できるが、指定できるアドレスはひとつのみとなり、特定のネットワークでしか使用できないIORとなる問題がある。
これを回避する方法を調査し、改良を行う。

関連するチケット:

関連している OpenRTM-aist (Java) - 機能 #1271: 複数NICを持つノードのCORBAオ...	**終了**	**2010/02/23**

履歴

#1 - 2010/01/10 00:19 - n-ando

IORには、オブジェクトのアドレス：ポートを含むプロファイルと呼ばれるデータを複数含むことができる。これを実現するには、omniORBでは、

- endPoint giop:tcp:<host>:port (コマンドラインオプションでは -ORBendPoint giop:tcp:<host>:port)
  を複数与えることで、与えたアドレスをIORに含めることができる。
- endPointPublishAllIFs = 1 (コマンドラインオプションでは -ORBendPointPublishAllIFs 1)
  を設定することで、IORにすべてのネットワークインターフェースのアドレスを含めることができる。
- omniORB4.1.* では、endPointPublish all(addr) もendPointPublishAllIFs = 1と同様の効果がある。

参考
http://omniorb.sourceforge.net/omni40/omniORB/omniORB008.html
http://omniorb.sourceforge.net/omni41/omniORB/omniORB008.html#toc42

1. rtc.conf の corba.args に -ORBendPointPublishAllIFs 1
   を指定した結果、RTSEでアクセスすると、応答が大変遅くなった。しばらく待つと、DnD、プロファイルの表示などができるが、遅すぎて実用にならない。また、Activate/Inactivateも同様に遅い上に、「サーバからの応答がありません」というエラーダイアログが表示される。
2. rtc.conf の corba.args に -ORBendPoint
   を利用するネットワークインターフェース分だけ(実験では2つ)指定したが、上記のケースよりは若干応答が速く、かろうじて実用になるレベルであった。しかし、 Activate/Inactivateでは同様に、「サーバからの応答がありません」というエラーダイアログが表示される。

http://java.sun.com/j2se/1.4.2/docs/guide/idl/jidlNaming.html
に以下の説明がある。

- Interoperable Object References (IOR)
  An IOR is an object reference that is understood by ORBs that can interoperate using the OMG-defined protocols GIOP and IIOP. A client can obtain an object reference using orb.object_to_string(objRef), as shown in the Browsing the Namespace example, or as a result of an invocation on another object reference.

Note: If an IOR contains multiple profiles, the J2SE v.1.4 ORB always uses the first one.

#2 - 2010/01/11 10:36 - n-ando

http://old.nabble.com/More-endPoint-questions-td6425783.html

ここに同様の問題に対する質問と解答がある。(本質的な解決策はない。)
この人は、ポータブルインターセプタのencodeIORを利用してIORのホストのエントリに手を加える方法をとっているらしい。

ただし、これを利用する場合でも、あるネームサーバに適切なアドレスを選択するには、OSルーティングテーブルに問い合わせて、ネームサーバのアドレスに対して適切なインターフェースアドレスを見つける必要がある。

#3 - 2010/01/11 23:57 - n-ando

- 進捗率 を 0 から 50 に変更

宛先アドレスから、使用されるエンドポイントを推定する方法について調査した。

- UNIX
  - 経路制御ソケット (AF_ROUTE を利用する方法) これはスーパーユーザ権限が必要
  - sysctl を利用して、経路情報を読み取る(AF_ROUTEを使用)方法
- Windows
  - GetBestInterface() APIを利用する方法
    などが存在する。

経路制御ソケットは、スーパーユーザ権限が必要なので、コンポーネントをrootユーザで実行するかsetuidするなどしなければいけないので現実的ではない。
また、sysctlについては、OS依存の可能性がある(らしい)ので、少々難しいかも知れない。

別の方法として、UNIX では route コマンド(Linuxではipコマンド)を利用する方法がある。

```
freebsd> route get 192.168.111.1
      route to: 192.168.111.1
destination: 192.168.111.1
    interface: le0
              flags: <UP,HOST,DONE,LLINFO,WASCLONED>
 recvpipe  sendpipe  ssthresh  rtt,msec    rttvar  hopcount      mtu     expire
       0         0         0         0         0         0      1500          1

linux$ ip route get 192.168.111.1
192.168.111.1 dev eth0  src 192.168.111.130
        cache  mtu 1500 advmss 1460 hoplimit 64
```

宛先アドレスを与えると、利用されるインターフェース名(le0)もしくはエンドポイントアドレスが得られる。

とりあえず、coil において、

```
    /*!
     * @if jp
     * @brief 宛先アドレスから利用されるエンドポイントアドレスを得る
     * @else
     * @brief Getting network interface name from destination address
     * @endif
     */
    bool dest_to_endpoint(std::string dest_addr, std::string& endpoint)
```

という関数を、posixではroute(もしくはip)コマンドを利用して、WindowsではGetBestInterface関数を利用して実装した。

#4 - 2010/01/12 00:12 - n-ando

- 進捗率 を 50 から 100 に変更

dest_to_endpoint を利用して、利用すべきエンドポイントアドレスを得ることができる。
さらに、ネームサーバに登録するオブジェクト参照のホストアドレス部分を、dest_to_endpoint
で得られたエンドポイントアドレスで書き換えることで、本問題を解決することができる。

IORの書き換えについては、omniORBのコードを参考に、CORBA_IORUtil::replaceEndpoint() 関数を実装した。
以下は、IORのホスト部分が書き換えられた際のログ出力である。

```
Jan 10 17:03:04 DEBUG: NamingOnCorba: Original IOR information:
  IOR information
    Type ID: "IDL:openrtm.aist.go.jp/OpenRTM/DataFlowComponent:1.0"
    Profiles:
        1. IIOP 1.2 192.168.111.128 2810
            POA(root)                Object Key: "...." = 0x00000000 (4 bytes)
            Object Key: "...IK..t:....." = 0xfeb889494b0000743a0000000000 (14 bytes)
            TAG_ORB_TYPE omniORB
            TAG_CODE_SETS char native code set: ISO-8859-1
                                  char conversion code set: UTF-8
                                  wchar native code set: UTF-16
                                  wchar conversion code set: UTF-16

Jan 10 17:03:04 TRACE: manager: Manager::getORB()
Jan 10 17:03:04 DEBUG: NamingOnCorba: Modified IOR information]
  IOR information
    Type ID: "IDL:openrtm.aist.go.jp/OpenRTM/DataFlowComponent:1.0"
    Profiles:
        1. IIOP 1.2 192.168.116.129 2810
            POA(root)                Object Key: "...." = 0x00000000 (4 bytes)
            Object Key: "...IK..t:....." = 0xfeb889494b0000743a0000000000 (14 bytes)
```

```
TAG_ORB_TYPE    omniORB
TAG_CODE_SETS   char   native   code   set:   ISO-8859-1
                                char   conversion   code   set:   UTF-8
                                wchar   native   code   set:   UTF-16
                                wchar   conversion   code   set:   UTF-16
```

192.168.111.128 から 192.168.116.129 に書き換えられていることがわかる。

#5 - 2010/01/12 14:19 - n-ando

エンドポイント書き換えをするかしないかを切り替えるオプションを導入する。

#6 - 2010/01/13 16:28 - n-ando

- ステータス を 新規 から 終了 に変更

エンドポイントを書き換えるオプションとして、corba.nameservice.replace_endpoint を導入した。(r1646)

ただし、ネームサーバに登録するオブジェクトリファレンスを書き換えても、RTObject等から取得するオブジェクトリファレンスのエンドポイントは書きわわっていないため、結局は複数のエンドポイントを有効にする必要がある。

omniORBどうしてあれば特に問題なくコンポーネント間の接続が可能であるが、Javaはアクセスが遅くなるか、タイムアウトになるなどして問題となる。

したがって、複数のエンドポイントを指定するためのオプションとして、corba.endpoints
オプションを導入した。これにより、複数のエンドポイントを指定することができる。

本件は、ポータブルインターセプタ等を利用してIORをすべてのケースにおいて書き換えるか、JavaのORBがmultiple
endpointに対応するのを待つしかないかもしれないので、ひとまずこのチケットは終了とする。