

RTM::Manager インタフェース

```
import "Manager.idl";
```

公開メンバ関数

RTC::ReturnCode_t	load_module (in string pathname, in string initfunc) モジュールをロードする [詳解]
RTC::ReturnCode_t	unload_module (in string pathname) モジュールをアンロードする [詳解]
ModuleProfileList	get_loadable_modules () ロード可能なモジュールのプロファイルを取得する [詳解]
ModuleProfileList	get_loaded_modules () ロード済みのモジュールのプロファイルを取得する [詳解]
ModuleProfileList	get_factory_profiles () コンポーネントファクトリのプロファイルを取得する [詳解]
RTC::RTObject	create_component (in string module_name) コンポーネントを生成する [詳解]
RTC::ReturnCode_t	delete_component (in string instance_name) コンポーネントを削除する [詳解]
RTC::RTCList	get_components () 起動中のコンポーネントのリストを取得する [詳解]
RTC::ComponentProfileList	get_component_profiles () 起動中のコンポーネントプロファイルのリストを取得する [詳解]
RTC::RTCList	get_components_by_name (in string name) 指定名のRTCオブジェクトリファレンスを取得 [詳解]
ManagerProfile	get_profile () マネージャのプロファイルを取得する [詳解]
NVList	get_configuration () マネージャのコンフィギュレーションを取得する [詳解]
RTC::ReturnCode_t	set_configuration (in string name, in string value) マネージャのコンフィギュレーションを設定する [詳解]
boolean	is_master () マネージャがマスターかどうか [詳解]
ManagerList	get_master_managers () マスターマネージャの取得 [詳解]
RTC::ReturnCode_t	add_master_manager (in Manager mgr) マスターマネージャの追加 [詳解]
RTC::ReturnCode_t	remove_master_manager (in Manager mgr) マスターマネージャの削除 [詳解]

ManagerList **get_slave_managers** ()

スレーブマネージャの取得 [\[詳解\]](#)

RTC::ReturnCode_t **add_slave_manager** (in **Manager** mgr)

スレーブマネージャの追加 [\[詳解\]](#)

RTC::ReturnCode_t **remove_slave_manager** (in **Manager** mgr)

スレーブマネージャの削除 [\[詳解\]](#)

RTC::ReturnCode_t **fork** ()

マネージャプロセスをforkする [\[詳解\]](#)

RTC::ReturnCode_t **shutdown** ()

マネージャプロセスをshutdownする [\[詳解\]](#)

RTC::ReturnCode_t **restart** ()

マネージャプロセスを再起動する [\[詳解\]](#)

Object **get_service** (in string name)

サービスのオブジェクト参照を取得する [\[詳解\]](#)

詳解

RTコンポーネントのライフサイクルの管理などを行うManagerへの外部インターフェース。主たる機能としては、以下のものがある。

- RTCのロードブルモジュールの操作
 - モジュールのロード
 - モジュールのアンロード
 - ロード可能なモジュール一覧の取得
 - ロード済みモジュール一覧の取得
 - ロード済みモジュールのプロファイル一覧の取得
- RTCに関する操作
 - コンポーネントの生成
 - コンポーネントの削除
 - インスタンス化済みのRTC一覧の取得
 - インスタンス化済みのRTCのプロファイル一覧の取得
 - 名前によるコンポーネントの取得
- マネージャに関する操作
 - マネージャのプロファイル情報の取得
 - マネージャのコンフィギュレーション情報の取得
 - マネージャのコンフィギュレーション情報の設定
 - マネージャがマスターかどうか
 - マスターマネージャの取得
 - マスターマネージャの追加
 - マスターマネージャの削除
 - スレーブマネージャの取得
 - スレーブマネージャの追加
 - スレーブマネージャの削除
 - サービスの取得

- プロセス操作
 - フォーク
 - シャットダウン
 - リスタート

マネージャには、同一ノード内で原則1つしか存在しないマスターマネージャと、マスターマネージャの管理下に0個以上存在するスレーブマネージャが存在する。

マスターマネージャは通常、デーモン・サービスなどとして常駐し、ノードのOS稼働中は原則として常に動作し続ける。また、マスターマネージャは自身の内部にRTCを生成・ホストせず、RTCの生成はスレーブマネージャに依頼する。依頼するスレーブマネージャは、すでに起動しているものでも、マスターマネージャが新規に起動してもよい。どちらの方法でRTCを生成するかは、`create_component()` の引数にて指定する。なお、スレーブマネージャは通常、1つ以上のRTCをホストする。デフォルトの設定では、RTCをホストしていないスレーブマネージャは自動終了する。設定項目：

- `manager.shutdown_on_nortcs`: YES
- `manager.shutdown_auto`: YES

```

          1          0..*          1          1..*
[ master ]<>-----[ slave ]<>-----[ RTC ]

```

関数詳解

RTC::ReturnCode_t RTM::Manager::add_master_manager (in Manager mgr)

マスターマネージャの追加

このマネージャのマスタとしてマネージャを一つ追加する。

マスターマネージャとスレーブマネージャの関係は、以下の図のように、一つのノード (node: 1つの独立したコンピュータでありホスト) に対して、1つのマスター (master) マネージャが対応する。1つのマスター マネージャの下には0個以上のスレーブ (slave) マネージャが存在し、RTCは1つのスレーブマネージャに属する。なお、スレーブマネージャは 通常、1つ以上のRTCをホストする。デフォルトの設定では、RTCをホス トしていないスレーブマネージャは自動終了する。

```

      1      1      1    0..*      1    1..*
[ node ]<-----[ master ]<-----[ slave ]<-----[ RTC ]

```

マスターとスレーブの関係は、add/remove_master_manager(), add_remove_slave_manager() オペレーションによって構成される。これらのオペレーションは原則として、すべてスレーブマネージャからアクションを開始する。マスターマネージャは原則として、corbaloc://localhost:2810/manager によりオブジェクト参照を取得できる一方、スレーブマネージャの参照を知る一般的な方法はない。スレーブマネージャは、起動時にローカルのマスターマネージャを探して、内部的にこのオペレーションを呼び出し、マスターマネージャを追加するとともに、マスターマネージャに対して、add_slabe_manager() を呼び出して、マスターマネージャに対して、スレーブマネージャが配下に入ったことを知らせる。

```

[ master ]                [ slave ]
|                          |
|                          |--, find master
|                          | | by corbaloc:
|                          |<-'
|                          |
|                          |--,
|                          | | add_master_manager ()
| add_slave_manager (own_ref) |<-'
|<-----|
|                          |

```

引数

mgr マスターマネージャ

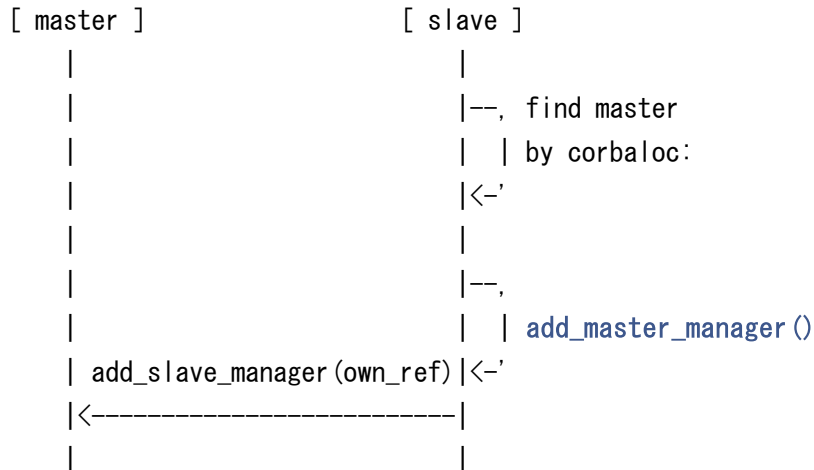
戻り値

ReturnCode_t

RTC::ReturnCode_t RTM::Manager::add_slave_manager (in Manager mgr)

スレーブマネージャの追加

このマネージャのマスタとしてマネージャを一つ追加する。通常、スレーブマネージャは、起動時にローカルのマスターマネージャを探して、内部的に、`add_master_manager()` を呼び出し、マスターマネージャを追加するとともに、マスターマネージャに対して、この **`add_slave_manager()`** を呼び出して、マスターマネージャに対して、スレーブマネージャが配下に入ったことを知らせる。

**引数**

mgr スレーブマネージャ

戻り値

ReturnCode_t

RTC::RObject RTM::Manager::create_component (in string module_name)

コンポーネントを生成する

引数に指定されたコンポーネントを生成する。マスターマネージャに対してこのオペレーションが呼び出された場合には、スレーブマネージャに対してコンポーネントの生成を依頼する。依頼対象となるスレーブマネージャは、create_component() の引数内パラメータ manager_name により指定されたマネージャが配下にある場合は、そのマネージャに対して、create_component() を呼び出すことで生成、manager_name で指定されたマネージャがない場合は、新規にスレーブマネージャを起動してからコンポーネントを生成する。

- manager_nameで指定されたスレーブが存在する場合

```

                                (manager_name = servo)
[ app ]           [ master ]           [ slave ]           [ RTC ]
| create_comp() |                       |                       .
|----->|                               |                       .
| manager_name = |--, search            |                       .
|           servo | | slaves            |                       .
|                       |<-'           |                       .
|                       | create_comp() |                       .
|                       |----->|       |                       .
|                       |               |--,                       .
|                       |               |createComp() .
|                       |               |<-'                       .
|                       |               |.....create...> .
|                       |               |                       |
|                       |               |                       |

```

- manager_nameで指定されたスレーブが存在しない場合

```

[ app ]           [ master ]           [ slave ]           [ RTC ]
| create_comp() |                       .                       .
|----->|                               .                       .
| manager_name = |--, search            .                       .
| controller    | | slaves            .                       .
|               |<-'                   .                       .
|               | launch proc          .                       .
|               |----->|             .                       .
|               | create_comp()        |                       .
|               |----->|             .                       .
|               |                       |--,                       .
|               |                       |createComp() .
|               |                       |<-'                       .
|               |                       |.....create...> .
|               |                       |                       |
|               |                       |                       |

```

指定パラメータ

- `manager_name`: コンポーネントを起動するスレーブマネージャ名
 - 名前指定: `<manager_name>`
 - アドレス指定: `<hostname>:<port>`
- `language`: 起動対象のコンポーネントの実装言語

引数

`module_name` 生成対象RTコンポーネントIDおよびコンフィギュレーション引数。フォーマットは大きく分けて "id" と "configuration" 部分が存在する。

`comp_args: [id]?[configuration]` id は必須、configurationはオプション id: **`RTC`**:`[vendor]:[category]:[implementation_id]:[version]` **`RTC`** は固定かつ必須 vendor, category, version はオプション implementation_id は必須 オプションを省略する場合でも ":" は省略不可 configuration: `[key0]=[value0]&[key1]=[value1]&[key2]=[value2].....` RTCが持つPropertiesの値をすべて上書きすることができる。key=value の形式で記述し、"&" で区切る

戻り値

生成されたRTコンポーネント

`RTC::ReturnCode_t` **`RTM::Manager::delete_component (in string instance_name)`**

コンポーネントを削除する

引数に指定されたコンポーネントを削除する。マスターマネージャに対してこのオペレーションが呼び出された場合、配下のスレーブコンポーネント上の当該コンポーネントを削除する。ただし、通常は対象コンポーネントに対して `exit()` オペレーションを呼び出すことで **`RTC`** を終了させる方法を推奨する。

戻り値

リターンコード

`RTC::ReturnCode_t` **`RTM::Manager::fork ()`**

マネージャプロセスをforkする

戻り値

Return Code

RTC::ComponentProfileList RTM::Manager::get_component_profiles ()

起動中のコンポーネントプロファイルのリストを取得する

現在当該マネージャ上で起動中のコンポーネントのプロファイルのリストを返す。マスターマネージャに対してこのオペレーションが呼び出された場合、配下のスレーブコンポーネント上で実行中のRTCのプロファイルリストを収集して返す。スレーブマネージャに対して呼び出された場合には、自身が管理する起動中のRTCのプロファイルリストのみを返す。

戻り値

RTコンポーネントプロファイルのリスト

RTC::RTCList RTM::Manager::get_components ()

起動中のコンポーネントのリストを取得する

現在当該マネージャ上で起動中のコンポーネントのリストを返す。マスターマネージャに対してこのオペレーションが呼び出された場合、配下のスレーブコンポーネント上で実行中のRTCのリストを収集して返す。スレーブマネージャに対して呼び出された場合には、自身が管理する起動中のRTCのリストのみを返す。

戻り値

RTコンポーネントのリスト

RTC::RTCList RTM::Manager::get_components_by_name (in string **name)**

指定名のRTCオブジェクトリファレンスを取得

引数

name RTC名

戻り値

RTCリスト

NVList RTM::Manager::get_configuration ()

マネージャのコンフィギュレーションを取得する

現在当該マネージャのコンフィギュレーションを取得する。

戻り値

マネージャコンフィギュレーション

ModuleProfileList RTM::Manager::get_factory_profiles ()

コンポーネントファクトリのプロファイルを取得する

ロード済みのモジュールのうち、RTコンポーネントのモジュールが持つ ファクトリのプロファイルのリストを取得する。このオペレーションでは、スレーブマネージャに対しても同じオペレーションが再帰的に呼び出され、配下にあるすべてのマネージャが保有するRTコンポーネントのプロファイルを返す。

戻り値

コンポーネントファクトリのプロファイルリスト

ModuleProfileList RTM::Manager::get_loadable_modules ()

ロード可能なモジュールのプロファイルを取得する

ロード可能なモジュールのプロファイルを取得する。ロード可能なRTコンポーネントのモジュールのプロファイルは、あらかじめ設定されたモジュールロードパスに対して、プロファイル取得コマンドを別プロセスで実行することで取得する。プロファイル取得対象となる言語の種類は、

- manager.supported_language: C++, Python, Java

で設定されている。ここで指定された言語に対して、以下のプロパティで言語毎のモジュールロードパス、プロファイル取得コマンド、モジュール拡張子が指定される。

- manager.modules.< 言語>="">.load_paths: ./ (モジュールロードパス)
- manager.modules.< 言語>="">.profile_cmd: rtcprof等コマンド名
- manager.modules.< 言語>="">.suffixes: モジュール拡張子

例えば、C++ 言語の場合は、以下のプロパティがデフォルトで設定されている。

- manager.modules.C++.load_paths: ./ (モジュールロードパス)
- manager.modules.C++.profile_cmd: rtcprof
- manager.modules.C++.suffixes: so (Linuxの場合), dll (Windowsの場合)

マスターマネージャは、上記で設定されたすべての言語 (WindowsではすべてのVCのバージョンも含む) のモジュールロードパスがあらかじめ設定されている必要がある。この作業は、このオペレーションが呼び出されたマネージャ (通常はマスターマネージャ) 上でのみ実行され、配下にあるスレーブマネージャ等に再帰的に呼び出されることはない。

戻り値

モジュールプロファイル。なお、モジュールプロファイルは以下のような内容を持つKey-Value形式のリストで返される。

```
implementation_id: ConfigSample
type_name: ConfigSample
description: Configuration example component
version: 1.0
vendor: Noriaki Ando, AIST
category: example
activity_type: DataFlowComponent
max_instance: 10
language: C++
lang_type: compile
conf.default.int_param0: 0
conf.default.int_param1: 1
conf.default.double_param0: 0.11
conf.default.double_param1: 9.9
conf.default.str_param0: hoge
conf.default.str_param1: dara
conf.default.vector_param0: 0.0, 1.0, 2.0, 3.0, 4.0
```

ModuleProfileList RTM::Manager::get_loaded_modules ()

ロード済みのモジュールのプロファイルを取得する

ロード済みのモジュールのプロファイルを取得する。このオペレーションでは、スレーブマネージャに対しても同じオペレーションが再帰的に呼び出され、配下にあるすべてのマネージャがロードしているすべてのモジュールのプロファイルを返す。

戻り値

モジュールプロファイル

ManagerList RTM::Manager::get_master_managers ()

マスターマネージャの取得

このマネージャがスレーブマネージャの場合、マスターとなっているマネージャのリストを返す。このマネージャがマスターの場合、空のリストが返る。原則としてスレーブマネージャは1つのマスターマネージャの配下となるが、実装上はマスターマネージャは唯一となるような制約はなく、add_master_manager() オペレーションにより複数のマスターマネージャを登録することが可能である。

戻り値

マスターマネージャのリスト

ManagerProfile RTM::Manager::get_profile ()

マネージャのプロファイルを取得する

現在当該マネージャのプロファイルを取得する。

戻り値

マネージャプロファイル

Object RTM::Manager::get_service (in string name)

サービスのオブジェクト参照を取得する

仕様未確定

戻り値

Return Code

ManagerList RTM::Manager::get_slave_managers ()

スレーブマネージャの取得

このマネージャがスレーブマネージャの場合、スレーブとなっているマネージャのリストを返す。このマネージャがスレーブの場合、空のリストが返る。

戻り値

スレーブマネージャのリスト

boolean RTM::Manager::is_master ()

マネージャがマスターかどうか

この関数はマネージャがマスターかどうかを返す。Trueならば、当該マネージャはマスターであり、それ以外は False を返す。

戻り値

マスターマネージャかどうかのbool値

RTC::ReturnCode_t RTM::Manager::load_module (in string **pathname,
in string **initfunc**
)**

モジュールをロードする

当該マネージャに指定されたモジュールをロードし、指定された初期化関数で初期化を行う。任意の共有オブジェクトファイル (.so, .dll) をロード可能であり、コンポーネントの共有オブジェクトファイルに限らない。通常、RTCの生成時は自動的にロードパス上を共有オブジェクトファイルを探索するため、この操作を呼び出す必要はない。

引数

pathname モジュールへのパス

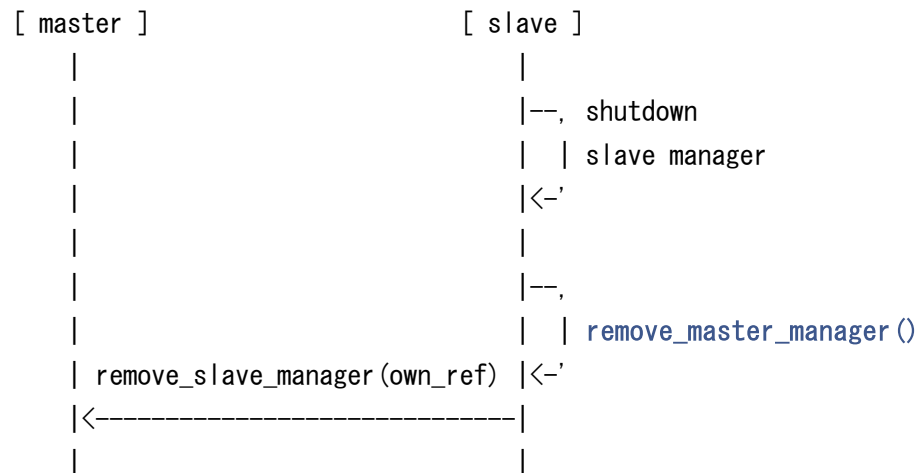
initfunc モジュールの初期化関数

戻り値

リターンコード

RTC::ReturnCode_t RTM::Manager::remove_master_manager (in Manager mgr)**マスターマネージャの削除**

このマネージャが保持するマスタのうち、指定されたものを削除する。通常このオペレーションは、スレーブマネージャが終了する際に、自身 に対して呼び出される。また、スレーブマネージャ自身がマスターマネージャの配下から外れることを **remove_slave_manager()** により、マスター マネージャに対して通知する。

**引数**

mgr マスターマネージャ

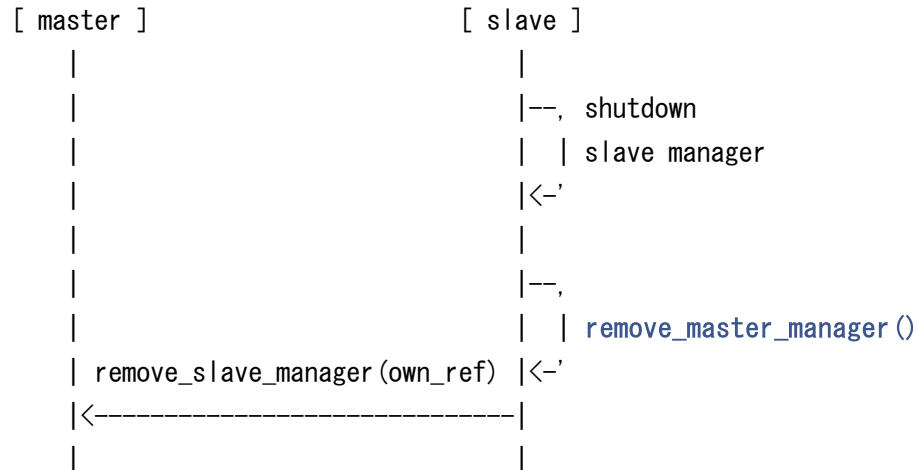
戻り値

ReturnCode_t

RTC::ReturnCode_t RTM::Manager::remove_slave_manager (in Manager mgr)

スレーブマネージャの削除

このマネージャが保持するマスタのうち、指定されたものを削除する。通常このオペレーションは、スレーブマネージャが終了する際に、自身 に対して **remove_master_manager()** を呼び出した後、スレーブマネージャ自身がマスターマネージャの配下から外れることを知らせるための、この **remove_slave_manager()** をマスターマネージャに対して呼ぶこと で通知する。



引数

mgr スレーブマネージャ

戻り値

ReturnCode_t

RTC::ReturnCode_t RTM::Manager::restart ()

マネージャプロセスを再起動する

戻り値

Return Code

```
RTC::ReturnCode_t RTM::Manager::set_configuration ( in string  name,
                                                    in string  value
                                                    )
```

マネージャのコンフィギュレーションを設定する

現在当該マネージャのコンフィギュレーションを設定する。

引数

name セットするコンフィギュレーションのキー名

value セットするコンフィギュレーションの値

戻り値

リターンコード

```
RTC::ReturnCode_t RTM::Manager::shutdown ( )
```

マネージャプロセスをshutdownする

戻り値

Return Code

```
RTC::ReturnCode_t RTM::Manager::unload_module ( in string  pathname )
```

モジュールをアンロードする

当該マネージャに指定されたモジュールをアンロードする。現在のところ、OpenRTM-aistのモジュールロード機能は、ロードされたモジュールにより生成されたオブジェクトのリファレンスカウント管理等は行われていないので、安易にモジュールをアンロードすると、メモリアクセス違反を引き起こす恐れがある。ただ、多くの動的ライブラリのアンロードの実装では、実際にメモリ上からアンロードされないケースが多く、したがってこのオペレーションは呼び出すことを推奨しない。

引数

pathname モジュールへのパス

戻り値

リターンコード

このインタフェース詳解は次のファイルから抽出されました:

- [Manager.idl](#)