CHAPTER 4

Collection Manager の管理

この章では、ユーティリティ スクリプトを使用して Collection Manager（CM）パラメータを表示およびアップデートする方法について説明します。

Telnet や SHH など CM を経由して接続するマシンは、CM をモニタし、管理するためのユーティリティ スクリプトを使用することができます。ユーティリティ スクリプトは CM のインストール ディレクトリにあります。

データベースおよび CSV リポジトリの管理については、「データベースおよび CSV リポジトリの管理」（p.5-1）を参照してください。

ユーティリティ スクリプトの使用法

ここでは、ユーティリティ スクリプトの一般的な使用手順を示します。

- スクリプトを呼び出すには、特に明記している場合を除き、scmscm ユーザとしてログインします。これらのスクリプトを root ユーザとして実行しようとすると、エラーが発生します。
- スクリプトの説明、およびすべてのフラグとパラメータの説明を表示するには、help フラグを指定してスクリプトを起動します。

（注）　help フラグはスクリプトごとに多少異なっています。CM の管理用スクリプトは **--help** を使用し、データベースの管理用スクリプトは **-h** を使用します。各スクリプトの定義を参照してください。

次に、**dbperiodic.py** スクリプトの説明を表示する例を示します。

```
>~scmscm/scripts/dbperiodic.py --helpUsage:
~scmscm/scripts/dbperiodic.py --load
load configuration from
/export/home/scmscm/db_maint/dbperiodic.conf
~scmscm/scripts/dbperiodic.py --loadfile=FILE
load configuration from FILE
~scmscm/scripts/dbperiodic.py --dump
print the current configuration in INI format to standard output
~scmscm/scripts/dbperiodic.py --help
print this help message
```

（注）　データ収集用ソフトウェアを制御およびモニタするためのスクリプトには、Python スクリプト言語を使用するものがあります。Python の詳細については、http://www.python.org を参照してください。

CM の設定

ユーティリティ スクリプトは以下の処理に使用します。

- 起動時にアクティブになるサーバの指定
- データベースの起動または終了
- アダプタの起動または終了
- Service Control Engine（SCE）接続の切断

CM の設定に使用するスクリプトは、次のとおりです。

- `~scmscm/setup/on-boot.py`
- `~scmscm/scripts/adapterconf.py`
- `~scmscm/scripts/dbconf.sh`
- `~scmscm/scripts/sceconf.py`

データベースおよび CSV リポジトリの管理のスクリプトについては、「データベースおよび CSV リポジトリの管理」（p.5-1）を参照してください。

以下のファイルも、CM の設定に使用されます。

- **cm.conf —** CM の一般設定、CM 起動時にオンになるアダプタが含まれています。「アダプタのイネーブル化」（p.4-4）を参照してください。
- **queue.conf —** アダプタ キューの設定。特定のアダプタと関連づけられている RDR タグが含まれています。「カテゴライザの設定」（p.4-6）を参照してください。

サーバのアクティブ化

起動時にサーバ（CM または Sybase）をアクティブにするように設定するには、**on-boot.py** スクリプトを使用します。

`~scmscm/setup/on-boot.py--cm=`*`flag`***`--sybase=`***`flag`**

次のシステム再起動時に変更が有効になります。

（注）　スクリプトをパラメータなしで実行して、各コンポーネントの現在の起動ステータスを確認します。

scmscm として次のスクリプトを実行し、CM を再起動します。

`~scmscm/cm/bin/cm restart`

表 4-1　on-boot.py オプション

`--cm=``{ on | off}`	起動時に CM をアクティブまたは非アクティブにします。
`--sybase=``{ on | off }`	起動時に Sybase サーバをアクティブまたは非アクティブにします。

次に、起動時に CM および Sybase がアクティブになるように設定する例を示します（これがスクリプトのデフォルト設定です）。

```
>~scmscm/setup/on-boot.py --cm=on --sybase=on
```

アダプタの制御

設定されたアダプタをシャットダウンまたは起動したり、現在稼働中の CM アダプタを表示するには、**adapterconf.py** スクリプトを使用します。

~scmscm/scripts/adapterconf.py--op=*action*[**--adapter**=*adapter name*]

表 4-2　adapterconf.py オプション

--op=*start*	**adapter** パラメータで指定されたアダプタを起動します。
--op=*stop*	**adapter** パラメータで指定されたアダプタをシャットダウンします。
--op=*list*	現在動作している CM アダプタを表示します。
adapter=*adapter name*	動作しているアダプタを識別します。**start** および **stop** アクションの場合だけ使用します。
--help	このスクリプトのオプションを表示します。

アダプタをシャット ダウンするには、scmscm ユーザとして次のスクリプトを実行します。

~scmscm/scripts/adapterconf.py--op=*stop***--adapter**=*adapter name*

アダプタを起動するには、scmscm ユーザとして次のスクリプトを実行します。

~scmscm/scripts/adapterconf.py--op=*start***--adapter**=*adapter name*

アダプタのイネーブル化

cm.conf ファイルの該当行の先頭にあるコメント記号を削除することで、CM 起動時にアダプタをオンに定義することができます。

次に、CM 起動時に RAG アダプタをオンにするように定義する例を示します。

```
adapter.4 = com.cisco.scmscm.adapters.rag.RAGAdapter
```

次に、CM 起動時に CSV アダプタをオフにしたままにするように定義する例を示します。

```
#adapter.2 = com.cisco.scmscm.adapters.CSVAdapter
```

（注）　**adapter.<number>** の値は、対応するアダプタの queue.conf ファイルに定義されている **adapter_id** パラメータ値と一致していなければいけません。

データベースの制御

CM データベースをシャットダウンまたは起動する場合、またはデータベースの動作ステータスを表示するには、**dbconf.sh** スクリプトを使用します。

~scmscm/scripts/dbconf.sh--op=*action*

このスクリプトは、バンドルされているデータベースでのみ使用することができます。

（注）　このスクリプトが動作するのは、sudo パッケージがインストールされている場合だけです。sudo をインストールしない場合に、Sybase を起動または終了するには、root ユーザとしてログインし、**/etc/init.d/sybase** スクリプトを実行する必要があります。

表 4-3　dbconf.sh オプション

--op=*start*	CM データベースを起動します。
--op=*stop*	CM データベースをシャットダウンします。
--op=*status*	データベースの現在の動作ステータスを表示します。

CM データベースをシャット ダウンするには、scmscm ユーザとして次のコマンドを実行します。

~scmscm/scripts/dbconf.sh --op=*stop*

CM データベースを起動するには、scmscm ユーザとして次のコマンドを実行します。

~scmscm/scripts/dbconf.sh --op=*start*

SCE 接続の切断

特定の SCE で接続を切断するには、**sceconf.py** スクリプトを使用します。

~scmscm/scripts/sceconf.py --op=*drop* **--ip=***IP address*

CM の HTTP アダプタが動作中の場合のみ、このスクリプトを使用することができます。

このスクリプトは、SCE 接続に関する情報を表示する場合にも使用されます（SCE 接続の確認 [p.4-12] を参照）。

表 4-4　sceconf.py オプション

Adapter=*IP address*	指定された IP アドレスの接続を切断します。
--help	このスクリプトのオプションを表示します。

SCE 接続を切断するには、scmscm ユーザとして次のコマンドを実行します。

~scmscm/scripts/sceconf.py --op=*drop* **--ip=***IP address*

カテゴライザの設定

カテゴライザでは、RDR タグに従って各 RDR を分類します。アダプタの RDR タグを tags パラメータ（カンマで区切られた RDR タグの一覧）に追加することで、RDR が特定のアダプタにルーティングされます。この設定は、**queue.conf** ファイルに含まれています。

次に、RDR タグ **4042321920** および **4042321922** を Topper/Aggregator アダプタに送信するように設定する例を示します。

```
# Topper/Aggregator Adapter
[topper-hi]
adapter_id=3
priority=3
warning_size=40000
maximum_size=50000
tags=4042321920,4042321922
```

（注）　**adapter_id** パラメータの値は、対応するアダプタの **cm.conf** ファイルに定義されている **adapter.<number>** と一致していなければいけません。

システム状態のモニタ

CM には、システムをモニタして、事前定義された、潜在的に問題を含む状態に対してアラートを発行する、小さくて拡張可能なフレームワークが含まれています。

CM のモニタに使用するスクリプトは、次のとおりです。

- `~scmscm/setup/monitor/setup-monitor.sh`
- `~scmscm/setup/monitor/monitor.sh`
- 定期チェッカのインストール（p.4-7）
- 定期チェッカ スクリプト（p.4-7）

定期チェッカのインストール

cron（定期スケジューラ）サブシステム内で、定期チェッカ スクリプト **monitor.sh** のエントリを作成（または削除）するには、**setup-monitor.sh** スクリプトを使用します。

```
~scmscm/setup/monitor/setup-monitor.sh-a flag[-i flag]
```

表 4-5　setup-monitor.sh オプション

`-a{ install| uninstall}`	cron 内で **monitor.sh** のエントリを作成 / 削除します。
`-i{ 30m| 1h| 12h| 24h}`	**monitor.sh** を 30 分、1 時間、12 時間、または 24 時間毎に実行します。

次に、30 分ごとに実行するように **monitor.sh** をインストールする例を示します。

```
$ ./setup-monitor.sh -a install -i 30m
```

次に、monitor.sh をアンインストールする例を示します。

```
$ ./setup-monitor.sh -a uninstall
```

定期チェッカ スクリプト

- 定期チェッカ スクリプト（p.4-7）
- テスト（p.4-8）

定期チェッカ スクリプト

定期チェッカ スクリプト **monitor.sh** は、実行中のシステムをさまざまな角度からモニタする一連のサブスクリプトを呼び出します。

```
~scmscm/setup/monitor/monitor.sh{ -a| TEST NAME} [ -v] [ -d]
```

このスクリプトは、可能であるもののコマンド ラインで実行することを想定していません。テスト結果は、syslog サブシステムに送信され、**/var/log/messages** ファイルに記録されます。

表 4-6　monitor.sh オプション

-a	すべてのテストを実行します。
TEST NAME	1 つ以上のテストの名前。テスト名はテスト ファイル名から先頭の桁と後ろの .sh を取り除いたものです。
-v	詳細モードの結果を出力します（成功したテストを記録します）。
-d	結果を画面に出力します（デフォルトで、結果は syslog に送信されます）。

実行されるテストは、以下のフォーマットで結果を返します。

```
STATUS: Message
```

- STATUS — PASS または FAIL
- Message — 短い通知ステータス メッセージ

たとえば、**FAIL:db "apricot" has only 1523 free blocks**

次に、使用可能なすべてのテストを実行して、システム出力を画面に出力する例を示します。

```
$ ./monitor.sh -d -aTest: 01free_db.sh. Status: PASS. Message: db apricot has 1532
free blocks
Test: 02cm_is_up.sh. Status: FAIL. Message: cm process is not running
```

次に、インストールされたデータベースに十分な空き容量があることをチェックするための単一テストを実行する例を示します。

```
$ ./monitor.sh -d free_dbTest: 01free_db.sh. Status: PASS. Message: db apricot has
1532 free blocks
```

テスト

以下のテストは、**monitor.sh** を使用して実行することができます。

- **db_up** — バンドルされている Sybase データベースが実行中かどうかをチェックします。
- **cm_up** — CM アプリケーションが実行中かどうかをチェックします。
- **free_db** — データベースに 10 % 以上の空き容量があるかどうかチェックします。
- **free_log** — データベース トランザクション ログに 70 % 以上の空き容量があるかどうかチェックします。
- **cm_persistent_buffers** — 各 CM アダプタの永続的バッファに含まれているファイル数が 500 ファイル未満かどうかをチェックします。

これらの全テストのスクリプトは、**~/setup/monitor/tests** ディレクトリにあります。

test_name というテストを呼び出す際に、スクリプトは **NNtest_name.sh** というファイルが検出されることを予想します。NN はスクリプト全体のプライオリティを指定したものです。たとえば、テスト **free_db** は、ファイル **01free_db.sh** にマッピングされます。

ユーザ管理

CM は、**p3rpc** ユーティリティを使用して、認証済の RPC コールについてユーザを管理します。

コマンドのフォーマットは、**p3rpc OPERATION [OPTIONS]** です。

次の表は、**p3rpc** の動作とオプションについてまとめたものです。

表 4-7　p3rpc の動作

動作	説明
--set-user--username=*username***--password**=*password*	ユーザ名とパスワードを追加しアップデートします。
--validate-password--username=*username***--password**=*password*	ユーザ名とパスワードを確認します。
--delete-user--username=*username*	ユーザ設定を削除します。
--show-users	すべての設定済ユーザを表示します。

仮想リンクの管理

スクリプトが CM ディストリビューションに含まれるため、特定の SCE に設定される仮想リンクおよびインデックスを管理できます。

仮想リンクを表示または設定するには、**update_vlinks.sh** スクリプトを使用します。

~scmscm/cm/bin/update_vlinks.sh--sce=*SCE IP address*[**--file**=*file*| **--show**]

表 4-8　update_vlinks.sh オプション

--sce=*SCE IP***--file**=*file*	指定された SCE の供給された CSV 形式ファイルにデータを持つ VLINK_INI テーブルをアップデートします。
--sce=*SCE IP***--show**	指定された SCE のエントリに関して、VLINK_INI テーブルにクエリーを実行します。
--help	このスクリプトのオプションを表示します。

仮想リンクの詳細を設定するには、scmscm ユーザとして次のコマンドを実行します。

~scmscm/cm/bin/update_vlinks.sh--sce=*SCE IP address***--file**=*file*

CSV ファイル形式は、link id（正の整数）、link direction（0 = アップストリーム、1 = ダウンストリーム）、name（文字列）です。

次の確認手順がファイル上で実行されます。

- ファイルが存在する。
- 各方向で仮想リンク ID が重複していない。
- 仮想リンク ID が 0 ～ 1024 の正の整数である。
- 方向は、0（アップストリーム）または 1（ダウンストリーム）である。
- 各方向で仮想リンク名が重複していない、または名前が空白ではない。
- 仮想リンク名が 256 文字以下で構成されている。34、39（'）、および 96（`）を除く、32 ～ 126（含める）間の ASCII コードによるすべてのプリント可能文字が使用されている。

ファイルが正常に確認されたあと、スクリプトは次の動作を実行します。

1. SCE IP フィールド内の SCE IP アドレスを含むすべてのエントリが、VLINK_INI テーブルから削除されます。
2. 次のフォーマットの 2 つのエントリが VLINK_INI テーブルに追加されます。
   - Timestamp, sce ip, 0, 0, "Default Virtual Link Up"
   - Timestamp, sce ip, 0, 1, "Default Virtual Link Down"
3. CSV ファイルは解析され、CSV ファイル内の各行は VLINK_INI テーブルの行エントリとして入力されます。

仮想リンクの詳細を表示するには、scmscm ユーザとして次のコマンドを実行します。

~scmscm/cm/bin/update_vlinks.sh--sce=*SCE IP address***--show**

CM のモニタ

スクリプトを使用すると、CM に関連する、次のようなシステム統計情報をモニタできます。

- データベースの空き容量の割合
- CM に入力される RDR レート
- SCE プラットフォームの接続データ

CM のモニタに使用するスクリプトは、次のとおりです。

- `~scmscm/scripts/dbfree.sh`
- `~scmscm/scripts/rdr-rate.py`
- `~scmscm/scripts/sceconf.py`
- `~scmscm/setup/alive.sh`

次のスクリプトは CM を設定する場合に使用しますが（「CM の設定」[p.4-3] を参照）、関連設定を表示する場合にも使用できます。

- `~scmscm/setup/on-boot.py`
- `~scmscm/scripts/adapterconf.py`
- `~scmscm/scripts/dbconf.sh`

データベース容量の確認

データベース レポート テーブルおよび関連するトランザクション ログ内の空き容量の割合を表示するには、**dbfree.sh** スクリプトを使用します。

`~scmscm/scripts/dbfree.sh`

このスクリプトは、バンドルされているデータベースでのみ使用することができます。

詳細手順

ステップ 1　scmscm ユーザとして、**dbfree.sh** スクリプトを実行します。

RDR レートの確認

CM に入力されるレポートの瞬間的な合計レートを表示するには、**rdr-rate.py** スクリプトを使用します。

`~scmscm/scripts/rdr-rate.py`

出力は、直前の 5 秒間に CM に着信した（すべての送信元からの）RDR の 1 秒間の合計レートを、単精度浮動小数点で表したものです。

CM の HTTP アダプタが動作中の場合のみこのスクリプトを使用することができます。

詳細手順

ステップ 1　scmscm ユーザとして、**rdr-rate.py** スクリプトを実行します。

SCE 接続の確認

SCE 接続に関する情報を表示するには、**sceconf.py** スクリプトを使用します。

```
~scmscm/scripts/sceconf.py --op=list
```

CM の HTTP アダプタが動作中の場合のみ、このスクリプトを使用することができます。

このスクリプトは、特定の SCE との接続を切断する場合にも使用します「SCE 接続の切断」（p.4-5）を参照してください。

詳細手順

ステップ 1　scmscm ユーザとして、**sceconf.py** スクリプトを実行します。

```
~scmscm/scripts/sceconf.py --op=list
```

例：

次に、SCE 接続の出力例を示します。

```
>~scmscm/scripts/sceconf.py --op=listIP                  Rate                  Peak
-------                 -------               -------
10.1.6.93               0.71798986            0.718
10.1.9.36               0.14420895            0.1442139
10.1.9.35               0.0                   0.027929332
10.1.12.11              0.0                   0.0
```

サーバが動作可能であることの確認

サーバが適切に機能していることを確認するには、**alive.sh** スクリプトを使用します。

```
~scmscm/setup/alive.sh
```

スクリプトで、次のコンポーネントの動作が確認されます。

- CM
- データベース（バンドルされているデータベースの場合）
- レポート テーブル（バンドルされているデータベースの場合）

停止しているコンポーネントがある場合、スクリプトはエラー メッセージを発行します。

詳細手順

ステップ 1　scmscm ユーザとして、**alive.sh** スクリプトを実行します。

（注）　起動後、コンポーネントの初期化には時間がかかります。再起動後、5 分待機してこのスクリプトを実行してください。