IBM PowerHA SystemMirror for AIX Standard Edition

バージョン **7.2.1**

PowerHA SystemMirror 用の Smart Assist アプリケーションの開発

IBM

IBM PowerHA SystemMirror for AIX Standard Edition

バージョン **7.2.1**

PowerHA SystemMirror 用の Smart Assist アプリケーションの開発

IBM

お願い

本書および本書で紹介する製品をご使用になる前に、 69 ページの『特記事項』に記載されている情報をお読みください。

本書は、IBM PowerHA SystemMirror 7.2.1 Standard Edition for AIX および新しい版で明記されていない限り、以降のすべてのリリースおよびモディフィケーションに適用されます。

お客様の環境によっては、資料中の円記号がバックスラッシュと表示されたり、バックスラッシュが円記号と表示されたりする場合があります。

原典： IBM PowerHA SystemMirror for AIX
Standard Edition
Version 7.2.1
Developing Smart Assist applications
for PowerHA SystemMirror

発行： 日本アイ・ビー・エム株式会社

担当： トランスレーション・サービス・センター

目次

本書について

本書では、PowerHA® SystemMirror® を構成することにより、インストール済みのアプリケーションを高可用性にする構成および管理ツールを開発する方法について説明します。

強調表示

本書では、以下の強調表示規則を使用します。

太字	システムによって名前が事前に定義されているコマンド、サブルーチン、キーワード、ファイル、構造、ディレクトリー、およびその他の項目を示します。また、ユーザーが選択するボタン、ラベル、アイコンなどのグラフィカル・オブジェクトも示します。
イタリック	実際の名前または値をユーザーが指定する必要があるパラメーターを示します。
モノスペース	特定のデータ値の例、画面に表示されるものと同様のテキスト例、プログラマーが作成するものと同様のプログラム・コード部分の例、システムからのメッセージ、実際に入力する必要がある情報などを示します。

AIX での大/小文字の区別

AIX® オペレーティング・システムは、すべてケース・センシティブとなっています。これは、英大文字と小文字が区別されるということです。例えば、**ls** コマンドを使用するとファイルをリスト表示できます。LS と入力した場合、そのようなコマンドは**ない**という応答がシステムから返ってきます。同様に、**FILEA**、**FiLea**、および **filea** は、同じディレクトリーにある場合でも、3 つの異なるファイル名です。予期しない処理が実行されないように、常に正しい大/小文字を使用するようにしてください。

ISO 9000

当製品の開発および製造には、ISO 9000 登録品質システムが使用されました。

関連情報

- PowerHA SystemMirror バージョン 7.2.1 の PDF 資料は、『PowerHA SystemMirror 7.2.1 PDFs』トピックで入手可能です。
- PowerHA SystemMirror バージョン 7.2.1 のリリース・ノートは、『PowerHA SystemMirror 7.2.1 release notes』トピックで入手可能です。

PowerHA SystemMirror 用の Smart Assist アプリケーションの開発

この情報を使用し、PowerHA SystemMirror を構成することにより、インストール済みのアプリケーションを高可用性にする構成および管理ツールを開発します。

このツールを開発するための前提条件として、以下の項目について十分に理解している必要があります。

- PowerHA SystemMirror の計画および管理 (既存の Smart Assist を含む)
- TCP/IP サブシステムなどの通信
- SMIT のコーディング
- Perl や KSH などのスクリプト言語
- ターゲット・アプリケーションの知識

PowerHA SystemMirror Smart Assist 開発概念

Smart Assist は、エンド・ユーザーがアプリケーションのインスタンス (DB2® や Oracle のインスタンス、または WebSphere® コンポーネントなど) を高可用性にするために PowerHA SystemMirror を構成できるツールです。

各 Smart Assist は、特定のアプリケーションをサポートするために必要な PowerHA SystemMirror コンポーネントの集合を管理します。ユーザーにはこの PowerHA SystemMirror コンポーネントの集合が 1 つのエンティティーとして示され、PowerHA SystemMirror ではこのエンティティーがアプリケーション名で示されます。Smart Assist では、個々のアプリケーションの追加、変更、および除去をサポートします。新しいアプリケーションに対して PowerHA SystemMirror を構成する際、Smart Assist はエンド・ユーザーに必要最小限の情報を要求し、次に、選択されたアプリケーション・インスタンスが使用するファイルシステム、ボリューム・グループ、サービス IP ラベル、およびその他のアプリケーション・リソースを自動検出します。その後、Smart Assist はアプリケーションを高可用性にするために、必要に応じて 1 つ以上の PowerHA SystemMirror リソース・グループ、アプリケーション・コントローラー、およびアプリケーション・モニターを構成します。これにより構成プロセスでのエンド・ユーザーの手順が省略され、PowerHA SystemMirror で基本アプリケーション・インスタンスを正しく構成することもできます。

Smart Assist は複数のコンポーネントで構成できます。各コンポーネントは、ターゲット・アプリケーション特有の側面をサポートします。例えば、Oracle Smart Assist には 3 つのコンポーネントがあります。1 つ目は Oracle RDBMS のサポート用、2 つ目は Oracle Application Server CFC のサポート用、3 つ目は Oracle Application Server AFC のサポート用です。これらの各コンポーネントには、異なる Smart Assist プロパティーと異なるユーザー・インターフェースが SMIT 内に存在する場合があります。

また、Smart Assist はクラスター検証と統合します。Smart Assist は、カスタムの PowerHA SystemMirror 検証方法を使用してクラスター検証を実行できます。ユーザーがターゲット・アプリケーションの構成または PowerHA SystemMirror の構成を変更する際、カスタム検証ルーチンによってアプリケーションが引き続き機能できるようになります。

Smart Assist 開発で使用される概念および用語

Smart Assist の実装では、いくつかの概念とそれに関連した用語が使用されます。

- アプリケーション。

  各 Smart Assist でアプリケーションがサポートされています。 DB2、WebSphere、および Oracle 用の Smart Assist があります。また、PowerHA SystemMirror には汎用アプリケーション Smart Assist (GASA) も組み込まれており、ユーザーが指定するアプリケーション始動スクリプトとアプリケーション停止スクリプトおよび関連ボリューム・グループを前提として汎用アプリケーションがサポートされます。
- アプリケーション・インスタンス。

  通常、Smart Assist は開始されると、ユーザーが以前に作成したアプリケーション・インスタンスを検出します。 DB2、WebSphere、および Oracle の Smart Assist にはすべて、それぞれの Smart Assist を使用する前にユーザーがセットアップする必要があるアプリケーション・インスタンスが事前に必要になります。
- アプリケーション、アプリケーション・コンポーネント、およびアプリケーション・サブフィーチャーのディスカバリー。

  ユーザーが PowerHA SystemMirror SMIT メニュー「**PowerHA SystemMirror** 構成へのアプリケーションの追加 **(Add an Application to the PowerHA SystemMirror Configuration)**」を選択すると、各 Smart Assist が提供するディスカバリー・スクリプトが、そのアプリケーションがインストールされているかどうかを検査します。アプリケーションがインストールされている場合、ディスカバリー・スクリプトはサポートされている特定のアプリケーションのコンポーネントまたはサブフィーチャーをディスカバーします。ディスカバリー・スクリプトは、アプリケーション・ディスカバリー・フレームワークに状況を報告し、その特定の Smart Assist コンポーネントを使用できるかどうかを示します。「ディスカバリー」という用語の使用は、PowerHA SystemMirror SMIT メニュー「拡張構成 **(Extended Configuration)**」のクラスター・ディスカバリー機能とは異なることに注意してください。
- *Smart Assist* アプリケーション・インスタンス。

  ユーザーが高可用性にするためのアプリケーション・インスタンスを選択した場合、Smart Assist は Smart Assist アプリケーション・インスタンスを作成しなければなりません。これにより、Smart Assist フレームワークが、基本アプリケーション・インスタンスを高可用性にするために必要な各種の PowerHA SystemMirror リソース・グループ、アプリケーション、およびアプリケーション・モニターを作成できるようになります。

Smart Assist の要件

Smart Assist にはいくつかの異なる要件があります。

Smart Assist は以下を実行します。

- アプリケーション、および必要に応じて現在構成されているリソース (サービス IP アドレス、ファイルシステム、ボリューム・グループなど) のインストール済み環境をディスカバーする
- ユーザーから構成情報 (新しいサービス IP アドレスなど) を取得するための SMIT インターフェースを提供する
- PowerHA SystemMirror およびアプリケーション構成情報を変更するための SMIT インターフェースを提供する
- アプリケーションを PowerHA SystemMirror に定義し、カスタムの始動および停止スクリプトを指定する

- アプリケーションのアプリケーション・モニターを指定する (適用可能な場合)
- 以下を含むリソース・グループを構成する
  - 1 次ノードおよびテークオーバー・ノード
  - アプリケーション
  - サービス IP アドレス
  - 共有ボリューム・グループ
- リソース・グループのさまざまな一時的な依存関係および場所の依存関係を構成する (アプリケーションのソリューションで必要な場合)
- PowerHA SystemMirror のファイル・コレクション機能を使用して同期する必要があるファイルを指定する
- 以前に構成されたアプリケーションを変更する
- 新しい検証方法を提供する
- 標準のクラスター・テスト・スイートが十分でない場合に、アプリケーションのクラスター構成をテストする方法を提供する (クラスター・テスト・ツールを使用)

本書で説明している Smart Assist フレームワークは、新しい Smart Assist を開発するための使いやすいフレームワークになるように意図されています。読者は本書を最初から最後まで読むのではなく、まず『Smart Assist のサンプル・プログラム』の例を参照し、その後に必要に応じて参照セクションに戻ると有効である場合があります。

関連概念:

53 ページの『Smart Assist のサンプル・プログラム』

以下のトピックには、汎用アプリケーション Smart Assist (GASA) に基づいた Smart Assist のサンプル・プログラムを記載します。

Smart Assist フレームワーク

Smart Assist フレームワークは、PowerHA SystemMirror が PowerHA SystemMirror Smart Assist の開発用に備えているインフラストラクチャーです。

以下の機能が含まれています。

- インストール済みの Smart Assist と関連アプリケーション (構成されていたり、構成されていない可能性があります) のリストをエンド・ユーザーに自動的に提供するディスカバリー機能。アプリケーションがインストールされていなかったり、Smart Assist がサポートするバージョンと異なる場合は、該当するメッセージが表示されます。
- SMIT パネルを使用してユーザーから特定の構成情報を取得する前に、必要に応じてソフトウェアのサイレント・インストールまたは構成を使用可能にする方法。
- Smart Assist がクラスター構成を変更し、クラスターの操作に影響を与えることができる Smart Assist API。これは PowerHA SystemMirror トポロジー、リソース・グループ、およびリソースの構成を作成するためのインターフェースです。
- Smart Assist を使用して構成されたアプリケーションに関する情報を保存および検索するための Smart Assist 登録 API。
- Smart Assist のデータを含むファイルセットが AIX でインストール済みまたは除去済みである場合に、その Smart Assist のデータをインストールまたは除去するためのユーティリティー。この Smart Assist を使用して構成されたアプリケーション、リソース・グループ、クラスターなどは引き続き残りますが、標準および拡張 PowerHA SystemMirror SMIT パスを使用して保持する必要があります。

- クラスター・テスト・ツールを使用してアプリケーションを含むリソース・グループをテストするメカニズム。
- クラスター検証ユーティリティーにカスタムの検証方法を追加する方法。
- クラスターのすべてのノードから情報を収集するセキュアな方法。

Smart Assist の全体のフロー

Smart Assist のインストール時、ファイルセットまたはインストール・スクリプトにより、フレームワークの API **claddsa** を使用して該当する Smart Assist レジストリー項目が PowerHA SystemMirror に ODM として追加され、その SMIT 画面が SMIT ODM に挿入されます。登録 ODM には、Smart Assist ID とコンポーネント ID を結合する名前/値のペアが含まれます。

注**:** IBM® 社内開発の Smart Assist を使用している場合は、ビルド・システムから項目がヘルプとともに自動的に追加されます (packages フォルダー内の packdep.mk ファイルが使用されます)。

Smart Assist SMIT メインメニューの「**PowerHA SystemMirror** 構成へのアプリケーションの追加 **(Add an Application to the PowerHA SystemMirror Configuration)**」から、以下の手順を実行します。

1. GASA Smart Assist の SMARTASSIST_VERSION という PowerHA SystemMirror ODM 項目の値を調べて、2 つの両方のノードで Smart Assist の最新のフレームワークが実行されているかどうか確認します。いずれかのノードで古いバージョンが実行されている場合は、ステップ 2 に進みます。これらのノードで現行バージョンが実行されている場合は、ステップ 3 に進みます。
2. PowerHA SystemMirror ODM の各 Smart Assist は、(ディスカバリー・スクリプトと呼ばれる) カスタム・スクリプトを実行します。このカスタム・スクリプトは、Smart Assist がサポートするアプリケーション・コードが、その時点でインストールされているかどうか検出します。その後、Smart Assist メニューで各 Smart Assist の状況がエンド・ユーザーに通知され (以下を参照)、ユーザーは基本アプリケーション・コードがインストールされている任意の Smart Assist を選択できます。Smart Assists のディスカバリー・スクリプトがターゲット・アプリケーションのインストール済みインスタンスを検出しない場合は、入力画面が表示されます。ただし、ユーザーはこれを選択すると、最初のディスカバリー画面より前にナビゲートしなければなりません。
3. 両方のノードにインストールされているすべての Smart Assist を検出してリストします。この時点ではディスカバリー・スクリプトは呼び出されず、開発者は Smart Assist を選択してさらに進みます (Smart Assist for SAP など)。
4. 構成のタイプに対して「自動ディスカバリーおよび構成 **(Automatic Discovery and Configuration)**」または「手動構成 **(Manual Configuration)**」を選択します。

   注**:** 「自動ディスカバリーおよび構成 **(Automatic Discovery and Configuration)**」を選択した場合は、前回選択した Smart Assist に使用可能なすべてのコンポーネントに対してディスカバリー・スクリプトが実行されます。
5. 処理するコンポーネントを選択します。選択したコンポーネントに構成する必要があるインスタンスが複数ある (例えば、特定の Oracle Database に複数の異なるデータベース・インスタンスがある) 場合は、その他のインスタンスを選択する必要があります。構成する必要があるインスタンスが 1 つしかない場合は、ダイアログ画面が表示されます。

注**:** ユーザーが既にクラスターとノードを構成している場合は、通信パスを要求する必要がありません。ユーザーは直接「アプリケーションを高可用性アプリケーションにする **(Make Applications Highly Available) > PowerHA SystemMirror** 構成へのアプリケーションの追加 **(Add an Application to the PowerHA SystemMirror Configuration)**」からセレクター画面に進みます。

```
************************************************
 Configure PowerHA SystemMirror Cluster and Nodes

 Enter Communication Path to Nodes <Entry Fields>

************************************************

 Select an Application from the List of Discovered...
 list of applications...

************************************************
```

注**:** Smart Assist の名前の後にノード名がリストされている Smart Assist は「アクティブ」です。これは、ディスカバリーによって検出されたアプリケーションのインスタンスが、そのアプリケーション・タイプの後にリストされているクラスター・ノードにインストールされているという意味です。上記の図では、DB2 または Oracle のインスタンスはディスカバーされていません。通常、Smart Assist は実行時にエンド・ユーザーから最小限の情報を取得し、その情報を使用して管理対象のアプリケーション・インスタンスに関する必要な情報 (ファイルシステムやボリューム・グループなど) を検出します。その後、フレームワークの構成 API を使用して、PowerHA SystemMirror クラスターと、アプリケーション・インスタンスに必要なリソース・グループ、アプリケーション、およびアプリケーション・モニターを構成します。
Smart Assist の開発者が使用する Smart Assist フレームワークは次のとおりです。

- 複数のコマンド (ユーザー・インターフェース API)。Smart Assist はこれらのコマンドを使用して、それ自体を SMIT と HACMPsa ODM 構造に追加します。
- Smart Assist SMIT メインメニュー (アプリケーションを高可用性アプリケーションにする (Make Applications Highly Available))。このメニューは、インストール済みの Smart Assist に接続するためのインターフェースです。
- ディスカバリー・スクリプトのサポート。このスクリプトは、Smart Assist によって提供され、SMIT メニュー「**PowerHA SystemMirror** 構成へのアプリケーションの追加 **(Add an Application to the PowerHA SystemMirror Configuration)**」から呼び出されます。これにより、特定のノードに基本アプリケーション・コードがインストールされているかどうか検出されます。
- 単一コマンド **clvt** (PowerHA SystemMirror 構成 API)。このコマンドは、基本のアプリケーション・インスタンスを高可用性にする必要がある PowerHA SystemMirror コンポーネントを表すオブジェクトの 11 個のクラスにアクセスします。Smart Assist はこれらのコマンドを使用して、基本のアプリケーション・インスタンスを高可用性にするために PowerHA SystemMirror を構成します。

Smart Assist の開発者はこのフレームワークを使用して、特定のアプリケーション用に Smart Assist を作成するための SMIT 画面を数多く開発する必要があります。Smart Assist フレームワークにはさらに項目がありますが、上記の項目は Smart Assist の開発者が考慮しなければならない基本的な機能を満たしています。

```
                    Make Applications Highly Available (Use Smart Assists)

Move cursor to desired item and press Enter

  Add an Application to the HACMP Configuration
  Change/Show an Application's HACMP Configuration
  Remove an Application from the HACMP Configuration

  Manage Your Applications
  Change/Show the Resources Associated with Your Application

  Test Your Application for Availability




F1=Help        F12=Refresh      F3=Cancel     F8=Image
F9=Shell       F10=Exit         Enter=Do
```

「アプリケーションを高可用性アプリケーションにする **(Make Applications Highly Available)**」SMIT メニューには、Smart Assist が提供する機能がリストされます。『Smart Assist Basics of Operation』。このセクションでは、Smart Assist の動作について詳しい情報を記載しています。また、このセクションでは、開発者が考慮すべき問題、および開発者独自の Smart Assist 用に作成する必要があるコードについて説明しています。

Smart Assist ID およびコンポーネント ID

Smart Assist ID およびコンポーネント ID は、Smart Assist フレームワーク内で特定の Smart Assist コンポーネントを一意的に識別します。

Smart Assist 開発の最初のステップとして、これらの ID を定義します。 Smart Assist ID は、DB2、Oracle、または WebSphere などのターゲット・アプリケーションを示します。例えば、DB2 の場合は「DB2_8.0」、Oracle Application Server および RDBMS の場合は「Oracle_10G」の Smart Assist ID となります。このような特定の場合、この ID はアプリケーションのバージョンも示します。バージョンの互換性を確保するには、ターゲット・アプリケーションの異なるバージョンごとに別々の Smart Assist を開発することが有効です。Smart Assist コンポーネント ID は、Smart Assist をアプリケーションのさまざまなサブコンポーネントに、および Smart Assist のサブフィーチャーに分類するために使用されます。DB2 と Oracle の場合は、データベース・インターフェースをさまざまな方法で構成できるため、データベースの構成方法ごとに 1 つずつ指定するための複数のコンポーネント ID が必要です。次の表では、DB2 コンポーネント ID の例を示します。

コンポーネント **ID**	**DB2** コンポーネント名 **(**ユーザーに表示される名前**)**
DB2_8.0_NON_DPF_SINGLE	DB2 単一インスタンス (DB2 Single Instance)
DB2_8.0_NON_DPF_MUTUAL	DB2 相互テークオーバー (DB2 Mutual Takeover)

WebSphere の場合、コンポーネント ID は複数の異なる機能するコンポーネント (Web サーバーやログ・サーバーなど) から構成されます。このため、各コンポーネントには固有のコンポーネント ID があります。次の表では、WebSphere コンポーネント ID の例を示します。

コンポーネント **ID**	**WebSphere** コンポーネント名 **(**ユーザーに表示される名前**)**
WAS_6.0_IHS_SERVER	IHS HTTP サーバー (IHS HTTP Server)
WAS_6.0_APP_SERVER	WebSphere アプリケーション・サーバー (スタンドアロン) (WebSphere Application Server (Standalone))
WAS_6.0_DEPLOYMENT_MANAGER	WebSphere デプロイメント・マネージャー (WebSphere Deployment Manager)
WAS_6.0_TRANSACTION_LOG_ RECOVERY	WebSphere クラスター・トランザクション・ログ (WebSphere Cluster Transaction Log)

パッケージ化およびインストール

ご使用の環境に適したツールを使用して Smart Assist をパッケージ化することができます。

LPP パッケージ管理システムは、AIX プラットフォームでファイルをパッケージ化するときに推奨される方法です。ただし、RPM または以下の操作を実行できる他のパッケージおよびファイル管理ツールも使用できます。

- インストール
- 除去
- 更新
- 新しい Smart Assist ファイルセットをインストールする前に、PowerHA SystemMirror ファイルセット **cluster.es.assist.common** をあらかじめインストールする必要があります。

インストール時の Smart Assist の追加

Smart Assist は、インストール時にその各コンポーネントに対して 2 つのタスクを実行します。

これらのタスクは次のとおりです。

- UI API ルーチン **claddsa** を呼び出します。このルーチンは、各コンポーネントの情報を PowerHA SystemMirror ODM に追加し、フレームワークが後で使用できるようにします。このときに指定される項目には、ディスカバリー・スクリプト・パスや SMIT の追加および変更メニュー・パスなどがあります。
- 各コンポーネントに指定される特定の SMIT の追加および変更画面を SMIT ODM に追加し、エンド・ユーザーが Smart Assist を起動したときに使用できるようにします。このときに他の SMIT 従属メニューも ODM に追加する必要があります。

claddsa 呼び出しの例を次に示します。

```
claddsa -s "Apache_2" -c "APACHE_2.0_HS_SSL" ¥
   COMPONENT_ID="APACHE_2.0_HS_SSL" ¥
   SMARTASSIST_VERSION="1.0" ¥
   SMIT_ADD="clsa_apache_add" ¥
   SMIT_MODIFY="clsa_apache_modify ¥
   SMIT_ADD_TYPE="d" ¥ SMIT_MODIFY_TYPE="d" ¥
   DISCOVERY_COMMAND="/usr/es/sbin/cluster/sa/apache/sbin/discover"
¥
   DEINSTALLATION_COMMAND="/usr/es/sbin/cluster/sa/apache/sbin/discover"
¥
   REGISTRATION_COMMAND="/usr/es/sbin/cluster/sa/apache/sbin/register"
¥
   DEREGISTRATION_COMMAND="/usr/es/sbin/cluster/sa/apache/sbin/deregister"
¥
   MIGRATION_COMMAND="/usr/es/sbin/cluster/sa/sample/sbin/migrate"
```

```
¥
   SA_ROOT="/usr/es/sbin/cluster/sa/sample/sbin/" ¥
   SA_NAME="Sample Smart Assist" ¥
   COMPONENT_NAME="Sample Smart Assist Component"
```

関連概念:

21 ページの『Smart Assist コマンド』

PowerHA SystemMirror のために Smart Assist を開発する際に使用するコマンドについては、これらのトピックを参照してください。それぞれのトピックでは、構文図と各コマンドを使用するための例を記載しています。

ディレクトリー構造

Smart Assist は、Smart Assist の名前の下にあるディレクトリー構造に従います。

```
/usr/es/sbin/cluster/sa/SmartAssistID
```

注**:** SmartAssistID は、**claddsa** コマンドで使用する ID と同じです。

ディレクトリー	内容
./sbin/	Smart Assist のために実行する必要がある実行可能ファイルおよびスクリプト。これには SMIT インターフェースを使用して実行されるスクリプトも含まれます
./appserver/	PowerHA SystemMirror アプリケーション・コントローラーの始動スクリプトおよび停止スクリプト
./monitor/	PowerHA SystemMirror カスタム・アプリケーション・モニター
./verification/	構成されたアプリケーションの妥当性検査を実行する PowerHA SystemMirror クラスター・カスタム検証ファイル
./smit	PowerHA SystemMirror クラスターに追加される一連の SMIT メニュー。Smart Assist の開発者がこれらの ODM スタンザを追加するためのメカニズムをインストール時に指定しない場合は、フレームワークがこのディレクトリーで SMIT ODM スタンザを検出できなければなりません。このディレクトリーには、「odm」のファイル拡張子を持つ SMIT ODM スタンザが 1 つ以上存在する可能性があります。

SMIT パネルの開発

開発する Smart Assist は、Smart Assist 用に設計された現在の PowerHA SystemMirror SMIT パネルに適合しなければなりません。

SMIT メニューは、Smart Assist を使用してインストール時にインストールされます。あるいは、Smart Assist で登録 API が ODM スタンザを追加するように要求することもできます。推奨される方法は、登録 API による ODM スタンザの追加を要求することです。これにより、Smart Assist のアンインストール・プロセスで ODM 項目を除去することができます。

PowerHA SystemMirror 登録 API では、sm_cmd_opt、sm_cmd_hdr、sm_menu_opt、sm_name_hdr の 4 つの SMIT スタンザ・タイプがサポートされています。Smart Assist フレームワークは、SMIT ODM スタンザに追加される項目を記録し、アンインストール時に単に登録解除コマンドを呼び出すことによって該当する項目を除去します。 Smart Assist が項目の追加操作を実行した場合、アンインストール時にその項目の除去も実行するとフレームワークでは見なされます。

Smart Assist フレームワークには、制御が Smart Assist フレームワークから新しい Smart Assist に移行するポイントが複数あります。セッション間で制御が適切に移行するように、SMIT ODM クラスに正しい Smart Assist ID を使用する必要があります。

関連情報:

System Management Interface Tool (SMIT)

PowerHA SystemMirror 構成へのアプリケーションの追加

「**Add an Application to the PowerHA SystemMirror Configuration** (PowerHA SystemMirror 構成へのアプリケーションの追加)」SMIT セッションは、多くの遷移を経てから最終的に Smart Assist に制御を渡します。

ユーザーは最初にクラスターのノードを指定する必要があります (ノードが存在する場合)。次に、使用可能なアプリケーションを示す選択画面が表示されます。ユーザーがアプリケーションを選択すると、Smart Assist の追加画面に制御が渡されます。

ユーザーがノードまたはサイトを指定した場合、その後次の名前値ペアが、SMIT 内のディスカバリー情報に表示されます。

```
#nodes: SmartAssistID_ComponentID
nodeA nodeB:<SmartAssistID_ComponentID>
```

アプリケーションの PowerHA SystemMirror 構成の変更または表示

「**PowerHA SystemMirror** 構成へのアプリケーションの追加 (Add An Application to the PowerHA SystemMirror Configuration)」SMIT 画面と同様、ユーザーは PowerHA SystemMirror に既に定義されている特定のアプリケーションを選択し、既存の値を変更します。

ユーザーが変更または表示するアプリケーションを選択すると、制御が Smart Assist に渡されます。

アプリケーションの管理

「アプリケーションの管理 **(Manage Your Applications)**」SMIT 画面は、追加および変更/表示 SMIT 画面とは異なります。この画面は、SMIT ダイアログではなく、Smart Assist で作成された SMIT メニュー・システムに制御を移行します。

「アプリケーションの管理 **(Manage Your Applications)**」SMIT 画面を選択すると、既に構成されていて、管理画面を持つアプリケーションのリストがユーザーに表示されます。ユーザーが特定の Smart Assist のコンポーネントを選択すると、Smart Assist メニューに制御が渡されます。Smart Assist の開発者は、clsa-manage <your_next_id> の ID を使用して sm_menu_opt SMIT ODM を指定する必要があります。

SMIT の開発に関する一般ガイドライン

各 Smart Assist には、SMIT の開発に少なくとも 2 つのディスプレイが必要です。1 つはアプリケーションを追加するためのディスプレイ、もう 1 つは既存の構成を変更するためのディスプレイです。SMIT の開発方法に関する説明をすることは本書の範囲を超えていますが、Smart Assist メニューから PowerHA SystemMirror 内でアプリケーションの構成を追加または変更するディスプレイに円滑に進むための要件をいくつか記載します。また、Smart Assist フレームワークにも開発者が使用できる有益な情報を記載します。

アプリケーションを追加するために SMIT ディスプレイにナビゲートする前に、追加のセレクターが必要になる場合があります。例えば、データベース・インスタンスを高可用性にする場合、Smart Assist ではユーザーが特定のデータベース・インスタンスをリストから選択する必要があります。この場合、エントリー・ポイントとしてディスプレイではなくセレクターを選択できます。SMIT メニューはオプションではありません (SMIT 要件のため)。追加および変更のためのエントリー・ポイントを登録するには、Smart Assist のインストール時に、以下のパラメーターを指定して **clquerysa** コマンドを実行する必要があります。

SMIT_ADD	**SMIT** スタンザの **ID** を含む文字列。
SMIT_ADD_TYPE	ディスプレイの場合は「d」、セレクターの場合は「n」の文字列。デフォルトは「d」です。
SMIT_MODIFY	SMIT スタンザの ID を含む文字列。
SMIT_MODIFY_TYPE	ディスプレイの場合は「d」、セレクターの場合は「n」の文字列。デフォルトは「d」です。

Smart Assist フレームワークによって、SMIT ディスプレイまたはセレクターで使用される以下のクックド・フィールド名の値が指定されます。

sa_id	**Smart Assist ID**
component_id	コンポーネント ID
cluster_name	クラスターの名前
nodes	コンポーネントがインストールされていたノードのスペース区切りリスト
application_id	(変更の場合のみ) アプリケーション ID

Smart Assist は、ユーザーから追加の情報を取得したり、ユーザーが追加、変更/表示、除去、およびテスト以外の操作を実行する方法を備えていなければならない場合があります。

開発者は、SMIT の「アプリケーションの管理 **(Manage Your Applications)**」メニュー・ヘッダーの下に SMIT メニュー、セレクター、およびダイアログをさらに追加することができます (clsa_manage)。最初の sm_menu_opt スタンザは、既存の Smart Assist シーケンス番号と競合しないシーケンス番号で開始しなければなりません。開発者の Assist のファイルセット・パッケージは、使用可能な SMIT ID を検査して次に使用可能な ID を使用するか、一連の ID を事前に割り当てて範囲が使用可能であるか検査します。ID が使用可能でない場合は、インストール・スクリプトを使用して SMIT sm_menu_opt ID を変更する必要があります。

注**:** この記述の時点では、clsa_manage 用に 100 以下および 900 以上のシーケンス番号が予約されています。

Smart Assist コンポーネント・ディスカバリー

Smart Assist のメイン画面「**PowerHA SystemMirror** 構成へのアプリケーションの追加 **(Add an Application to the PowerHA SystemMirror Configuration)**」を選択すると、追加されている各 Smart Assist コンポーネントによって提供されるディスカバリー・スクリプトが呼び出されます。

各スクリプトは、基本アプリケーション (DB2 や WebSphere など) がクラスターのノードにインストールされているかどうか検査し、次にディスカバリー・コマンドが実行されるローカル・ノードでサブコンポーネントまたはサブフィーチャーを使用できるかどうか検査します。機能またはアプリケーションがインストールされている場合は、メニューで Smart Assist が「使用可能」となり、その機能またはアプリケーションのサブコンポーネントがアクセス可能であるノードがその Smart Assist の横に表示されます。これにより、リアルタイムに構成されたメニューがユーザーに表示され、どのアプリケーションがインストールされているか、またどのクラスター・ノードでどの Smart Assist が使用可能であるかが示されます。アプリケーションがインストールされていない場合は、ユーザーがそのアプリケーションをインストールしたり、Smart Assist を再始動したりできます。

ディスカバリー・スクリプトは、Smart Assist がインストールされている各ノードで実行されます。定義されているすべてのクラスター・ノードに基本アプリケーションをインストールする必要があるかどうかは Smart Assist の開発者の判断によります。複数のノードまたはすべてのクラスター・ノードに Smart

Assist ファイルセットを含めるようにするには、パラメーター化された検証機能を使用することができます。この機能については、『パラメーター化された検証チェック・ファイル』セクションで説明しています。

関連資料:

16 ページの『パラメーター化された検証チェック・ファイル』
それぞれのパラメーター化された検証チェック・ファイルには、1 つ以上の異なるパラメーター化された検証チェックを含めることができます。

例: Smart Assist ディスカバリー・スクリプト

Smart Assist ディスカバリー・スクリプトの例を次に示します。

```
#!/bin/ksh93
#
# Apache 2.0 Discovery script
#
# This script will determine if Apache is installed, and if so, if the
#mod_ssl.so module is installed on the local node. If both conditions
#are true, the script will output the following string:
#
# Apache v2.0 Smart Assist:APACHE_2.0:Hot-Standby SSL Apache
#Server:APACHE_2.0_HS_SSL:1
#
# Otherwise, the last 2 characters will be replaced with :0 noting that
#the apache/ssl package is not properly installed on the local node.
#
dspmsg -s 1 apache.cat 1 "Apache v2.0 Smart Assist"
print -n ":APACHE_2.0:"
dspmsg -s 1 apache.cat 2
"Hot-Standby SSL Apache Server"
print -n ":APACHE_2.0_HS_SSL:"
# Simple example of scanning for the apache RPM on an AIX machine
result=$(rpm -qa 2>/dev/null | grep apache)
[[ -z $result ]] && {
 echo "0"
 exit 0
}
# If SSL is installed in this version of apache, this node has SSL
#available
rpm -ql apache 2>/dev/null | grep mod_ssl.so >/dev/null
 (( $? == 0 )) && {
 echo "1"
}
echo "0"
```

Smart Assist のコンポーネント追加メニューの起動

インストールされているアプリケーションとともに Smart Assist を選択する場合、フレームワークによってコンポーネント名のリストが表示されます。

これらの名前は、コンポーネントのディスカバリー・プロセスから生成されます。開発者はコンポーネントを選択する必要があり、Smart Assist が追加されたときに指定された SMIT 追加メニューが起動します。その後、Smart Assist の開発者が作成した SMIT メニューのコードが実行されます。

Smart Assist フレームワークは、**claddsa** に送られた 2 つの名前値ペア SMIT_ADD および SMIT_ADD_TYPE を使用します。SMIT_ADD_TYPE の値は「n」または「d」です。これらの値は、sm_name_hdr などの smitty ODM クラスで使用される値に対応します (特に next_type フィールド)。

「n」を指定すると、特定の Smart Assist コンポーネントを選択した後に、次の SMIT ID が sm_name_hdr 項目となります。多くの場合、sm_name_hdr SMIT 画面は、「追加」ダイアログに入る前

に特定のアプリケーション・インスタンス (DB2 インスタンス名など) またはオプションを選択するために使用されます。SMIT_ADD_TYPE に「d」を指定すると、次の SMIT ID が sm_cmd_hdr 項目 (ダイアログ) となります。 Smart Assist コンポーネントの SMIT_ADD フィールドは、次の ID の SMIT があることを指定します。

特定の Smart Assist SMIT スタンザが呼び出される場合、事前にアプリケーション・ディスカバリー・フレームワークによって以下のクックド名フィールドが作成されます。これらのフィールドは、SMIT で cmd_to_classify、cmd_to_discover、または cmd_to_exec 機能に対して使用できます。

SMIT 内で特定の Smart Assist に制御が渡されると、それ以降はアプリケーション・フレームワークの SMIT 画面は呼び出されません。開発者は必要に応じて自由に sm_name_hdr 項目と sm_cmd_hdr 項目を作成することができます。

SMIT でのクックド名	説明
sa_id	Smart Assist ID
component_id	コンポーネント ID
cluster_name	クラスターの名前
nodes	コンポーネントがインストールされていたノードのコンマ区切りリスト
application_id	(変更画面の場合のみ) アプリケーション ID

Smart Assist コンポーネントの標準的な追加メニュー機能**:**

通常、アプリケーション・インスタンスは既に構成されています。このタスクは、SMIT パネルを使用して、ユーザーの入力を取得し、アプリケーション・インスタンスで既に使用されているリソースを検出し、それらのリソースを高可用性になるように PowerHA SystemMirror が認識できるようにします。

インスタンス名の入力を要求されます。通常、この入力からボリューム・グループ名、ファイルシステム名、およびサービス・ラベルが決定されます。

次に、**claddsaapp** API ルーチンを使用して Smart Assist アプリケーション・インスタンスが作成されます。このルーチンは、インスタンス情報を HACMPsa_metadata ODM に格納します。Smart Assist アプリケーション・インスタンスの他の属性は、必要に応じて登録することができます。

```
/usr/es/sbin/cluster/sa/sbin/claddsaapp -a example_app ¥
 APPLICATION_NAME=" example_app" ¥
 RESOURCE_GROUP="example_app_group" ¥
 SMARTASSIST_ID="zzOther" ¥
 COMPONENT_ID="GASA"
```

その後、Smart Assist はクラスター構成 API (**clvt**) を使用して、アプリケーションに必要なリソース・グループとアプリケーション・モニターを構成します。

追加スクリプトが最初に作成する呼び出しは、アプリケーション・フレームワーク API **/usr/es/sbin/cluster/sa/sbin/clsapre** へのものです。このスクリプトには **-c** フラグが渡され、必要に応じて構成される特定のアプリケーションに適合するようにクラスターの名前が変更されます。

追加スクリプトが正常に完了した後、開発者が **/usr/es/sbin/sa/sbin/clsapost -v** スクリプトを呼び出します。**-v** フラグは、アプリケーション・インスタンスの追加のための最後のステップとして検証と同期を実行します。

追加スクリプトの例については、GASA スクリプト **/usr/es/sbin/cluster/sa/gasa/sbin/add** を参照してください。

関連概念:

28 ページの『clvt API』
これらのトピックでは、**clvt** (クラスター仮想化ツール) API について説明します。

PowerHA SystemMirror リソースを使用した既存の **Smart Assist** アプリケーション・インスタンスの変更または表示**:**

「アプリケーションの **PowerHA SystemMirror** 構成の変更/表示 **(Change/Show an Application's PowerHA SystemMirror Configuration)**」メニューには、特定の PowerHA SystemMirror 構成を細かく変更する方法があります。

「アプリケーションの **PowerHA SystemMirror** 構成の変更/表示 **(Change/Show an Application's PowerHA SystemMirror Configuration)**」を選択すると、変更可能な Smart Assist アプリケーション・インスタンスのリストが表示されます。 Smart Assist の開発者は、必要な変更を行うための画面を提供する必要があります。また、変更はクラスター構成 API **clvt** を使用して行われます。

```
                Make Applications Highly Available (Use Smart Assists)

Move cursor to desired item and press Enter

 Add an Application to the PowerHA SysteMirror Configuration
 Change/Show an Application's PowerHA SysteMirror Configuration
 Remove an Application from the PowerHA SysteMirror Configuration

 Manage Your Applications
 Change/Show the Resources Associated with Your Application

 Test Your Application for Availability
 ┌──────────────────────────────────────────────────────────────────────┐
 │   Select an Application from the List of Discovered Applications Below   │
 │  Move cursor to desired item and press Enter.                        │
 │   example_app                                                        │
 │  F1=Help        F12=Refresh      F3=Cancel     F8=Image              │
 │  F9=Shell       F10=Exit         Enter=Do                            │
 │  /=Find         n=Find Next                                          │
 └──────────────────────────────────────────────────────────────────────┘
```

変更の変更/表示スクリプトの例としては、GASA スクリプト **/usr/es/sbin/cluster/sa/gasa/sbin/modify** を参照してください。

特定のアプリケーション・インスタンス (上記の図の example_app) を選択すると、アプリケーション・フレームワークは HACMPsa ODM から SMIT_MODIFY_TYPE フィールドと SMIT_MODIFY フィールドを読み取ります。これらの項目は Smart Assist ファイルセット、すなわちパッケージがインストールされたときに定義され、インストーラーは **claddsa** Smart Assist コマンドを呼び出します。

```
                    General Applications Smart Assist

Type or select values in entry fields
Press Enter AFTER making all desired changes

                                       [Entry Fields]
 * Application Kane                    example_app
 * Primary Node                        [cujo]                    +
 * Takeover Nodes                      [cemetery]                +
 * Application Controller Start Script     [/exampleapp/start]
 * Application Controller Stop Script      [exampleapp/stop]
 * Service IP Label                    [service1]                +




F1=Help        F12=Refresh      F3=Cancel     F4=List
F5=Reset       F6=Command       F7=Edit       F8=Image
F9=Shell       F10=Exit         Enter=Do
```

Smart Assist 管理メニュー**:**

SMIT の制限として、追加メニューと変更/表示メニューはディスプレイとセレクターのみです。開発者は、SMIT の「アプリケーションの管理 **(Manage Your Applications)**」メニュー・ヘッダーの下に SMIT メニュー、セレクター、およびダイアログをさらに追加することができます (clsa_manage)。

最初の sm_menu_opt スタンザは、他のシーケンス番号と競合しないシーケンス番号で clsa_manage を指定する必要があります。インストール・スクリプト内で **odmadd** 呼び出しを使用して、インストール時に SMIT ODM スタンザをインストールする必要があります。前述されているクックド・フィールド名は SMIT では使用できません。

注**:** Smart Assist の開発者が開発する clsa_manage 画面用に 100 以下および 900 以上のシーケンス番号が予約されています。

Smart Assist アプリケーション・インスタンスの除去**:**

「**PowerHA SystemMirror** 構成からのアプリケーションの除去 **(Remove an Application from the PowerHA SystemMirror Configuration)**」メニューの選択項目には、Smart Assist アプリケーション・インスタンスのリストが表示されます。

Smart Assist アプリケーション・インスタンスを除去するには、削除するアプリケーション・インスタンスを選択します。**clrmsaapp** UI API ルーチンによって、そのインスタンス・データが HACMPsa_metadata から除去されます。一般的に、作成されたリソース・グループは、ユーザーが通常の PowerHA SystemMirror 機能を使用して管理できるように保持されます。

```
                Make Applications Highly Available (Use Smart Assists)

Move cursor to desired item and press Enter

 Add an Application to the PowerHA SystemMirror Configuration
 Change/Show an Application's PowerHA SystemMirror Configuration
 Remove an Application from the PowerHA SystemMirror Configuration

 Manage Your Applications
 Change/Show the Resources Associated with Your Application

 Test Your Application for Availability

  ┌──────────────────────────────────────────────────────────────────────┐
  │                                                                      │
  │ Select an Application from the List of Discovered Applications Below │
  │                                                                      │
  │ Move cursor to desired item and press Enter.                         │
  │                                                                      │
  │  example_app                                                         │
  │                                                                      │
  │ F1=Help        F12=Refresh      F3=Cancel     F8=Image               │
  │ F9=Shell       F10=Exit         Enter=Do                             │
  │ /=Find         n=Find Next                                           │
F1                                                                       │
F9└──────────────────────────────────────────────────────────────────────┘
```

Smart Assist によって作成された **claddsa** 呼び出しに指定されている DEREGISTRATION_COMMAND は、上記のアクションが実行される前に呼び出されます。

Smart Assist の除去**:**

Smart Assist の除去は、インストール時の **Smart Assist** の追加の操作と逆の操作です。

これはアンインストール・スクリプトによって実行されます。

- SMIT の追加、変更/表示、および管理メニューを SMIT ODM から除去する必要があります。
- Smart Assist ODM 内の項目は、**clrmsa** UI API ルーチンを使用して除去されます。

一般的に、作成されたリソース・グループは、ユーザーが通常の PowerHA SystemMirror 機能を使用して管理できるように保持されます。

カスタム検証およびパラメーター化された検証チェック

Smart Assist の追加または変更スクリプトを使用してクラスターを構成したら、最後に **clsapost -v** スクリプトを使用して検証と同期を実行します。

その際、必要なリソースが存在することを確認するために、オプションでクラスターのノードにおいて多くのチェックを行うことができます。

Smart Assist の開発者は、2 通りの方法を使用して、新しい検証チェックを PowerHA SystemMirror 製品に導入することができます。

1. カスタム検証方法。これは PowerHA SystemMirror を使用していないシステムとアプリケーション・コンポーネントのサード・パーティー検証を実行するために使用される既存の PowerHA SystemMirror メカニズムです。
2. パラメーター化された検証チェック (本章で説明)。

パラメーター化された検証チェックは、**/usr/es/sbin/cluster/etc/config/verify** ディレクトリー内にある記述ファイルです。

これらのファイルは、**.ver** の拡張子を持ち、通常、REGISTRATION_COMMAND を使用して Smart Assist インスタンスを作成するときに作成されます。これらは、インスタンスに必要なリソースが存在するか、またそのリソースが十分であるかどうか検査します。次のチェックを行うことができます。

- APAR がロードされている
- ディスク・スペースが使用可能である
- ファイルが存在する
- ファイルセットがインストールされている
- グループが存在する
- スワップ・スペースが存在する
- ユーザーが存在する

関連資料:

『パラメーター化された検証チェック・ファイル』
それぞれのパラメーター化された検証チェック・ファイルには、1 つ以上の異なるパラメーター化された検証チェックを含めることができます。

パラメーター化された検証チェック・ファイル

それぞれのパラメーター化された検証チェック・ファイルには、1 つ以上の異なるパラメーター化された検証チェックを含めることができます。

各ファイルには以下の属性があります。これらの属性は、ファイル内のすべてのパラメーター化された検証チェックにグローバルに適用されます。ファイルにコメントを追加するには、行の先頭にポンド記号 (#) を付けます。

```
Component.Name.DefaultName = "Default name as visible in error or
warning messages"
Component.MsgCat.ID = 1
Component.MsgCat.Set = 10
Component.MsgCat.Catalog = "myassist.cat"

Component.Nodes = "AlL"
または
Component.Nodes = "S=<smart assist ID>:A=<ApplicationID>"
または
Component.Nodes = "LOCALNODE"

Component.Name.DefaultName
```

Component.Name.DefaultName は、以下に記載されるパラメーター化された検証チェックによって生成される検証エラーまたは警告で使用される名前です。

Component.MsgCat

- ID - dspmsg で使用されるメッセージ・カタログ ID
- Set - dspmsg で使用されるメッセージ・カタログ・セット
- Catalog - dspmsg で使用されるメッセージ・カタログ名

Component.Nodes

- ALL - 使用可能なすべてのノードで検証する
- LOCALNODE - ローカル・ノードのみで検証する
- S=<SmartAssistID>:A=<ApplicationID> 特定のアプリケーション・インスタンスに関連したノードで検証する

APAR 検証

パラメーター化された APAR 検証チェックでは、指定された一連の APAR が *Component.Nodes* で定義されたノードにインストールされているかどうか検証します。

1 つ以上のノードに必要なファイルセットがない場合は、検証メッセージが出力され、メッセージの重大度 (エラーまたは警告) が *HAVerify.APAR.severity* の値によって決定されます。この値は次のように「ERROR」または「WARNING」のいずれかです。

```
HAVerify.APAR.severity = "ERROR | WARNING"
HAVerify.APAR.exists[0].apar = "IY7265H"
HAVerify.APAR.exists[1].apar = "IY72657"
.
.
.
HAVerify.APAR.exists[n].apar = "..."
```

ディスク・スペース検証

ディスク・スペースのパラメーター化された検証チェックでは、指定されたファイルシステムに十分なスペースが (指定どおりに) あるかどうか検証します。

ディスク・スペース検証チェックのフォーマットは次のとおりです。

```
HAVerify.DiskSpace.severity = "ERROR | WARNING"

#Validate /var filesystem has 200 MB of free space
HAVerify.DiskSpace.check[0].filesystem = "/var"
HAVerify.DiskSpace.check[0].minsize = "200MB"

# Validate /oradata has 1GB of free space
HAVerify.DiskSpace.check[1].filesystem = "/var"
HAVerify.DiskSpace.check[1].minsize = "200GB"

# Validate /orasoft has 4000Kb of free space
HAVerify.DiskSpace.check[2].filesystem = "/orasoft"
HAVerify.DiskSpace.check[2].minsize = "4000Kb"
...
```

minsize の修飾子は、# の後に [Kb - キロバイト | MB - メガバイト | GB - ギガバイト | B - バイト] を付けます。

ファイル検証

ファイル検証では、指定された一連のファイルが指定されたノードに存在するかどうかを判別します。

ファイルが存在しない場合は、検証の出力にエラーまたは警告メッセージが生成されます。ファイル・セクションのフォーマットは次のとおりです。

```
HAVerify.File.severity = "ERROR | WARNING"
HAVerify.File.exists[0].name = "/etc/hosts"
HAVerify.File.exists[1].name = "/orasoft/10g/admin/dbs/asdb/pfile/
warehouse_pfile.ora"
.
.
.
HAVerify.File.exists[n].name = "..."
```

ファイルセット検証

パラメーター化されたファイルセット検証チェックでは、一連の LPP ファイルセットがファイルのヘッダー情報にある一連のノードにインストールされているかどうか検証します。

ファイルセット・セクションのフォーマットは次のとおりです。

```
HAVerify.Fileset.severity = "ERROR | WARNING"

# Detect cluster.es.server.cfgast version installed
HAVerify.Fileset.exists[0].name = "cluster.es.server.cfgast"
HAVerify.Fileset.exists[0].version = "n.n"

# Detect cluster.es.server.testtool is installed (no version required)
HAVerify.Fileset.exists[1].name = "cluster.es.server.testtool"
```

グループ検証

パラメーター化されたグループ検証チェックでは、AIX グループがこのセクションに指定された ID を使用して、ヘッダーに指定された一連のノードに定義されているかどうか検証します。

1 つ以上のグループを検証できます。このセクションのフォーマットは次のとおりです。

```
HAVerify.Group.exists.severity = "ERROR | WARNING"
HAVerify.Group.exists[0].name = "dba"
HAVerify.Group.exists[0].GID = 100
```

ユーザー検証

パラメーター化されたユーザー検証チェックでは、AIX ユーザーがヘッダーに指定された一連のノードに定義されているかどうか検証します。

ユーザーにはセクションに指定された ID と同じ ID が必要です。 1 つの検証スクリプト内で 1 人以上のユーザーを検証することができます。このセクションのフォーマットは次のとおりです。

```
HAVerify.User.exists.severity. = "ERROR | WARNING"
HAVerify.User.exists[0].name = "oracle"
HAVerify.User.exists[0].UID = 100 ...
```

スワップ・スペース検証

パラメーター化されたスワップ・スペース検証チェックでは、ヘッダーに指定された一連のノードのスワップ・スペースが、空きスワップ・スペースと使用可能スワップ・スペースの合計に関する要件を満たしているかどうか検証します。

このセクションのフォーマットは次のとおりです。

```
HAVerify.SwapSpace.severity = "ERROR | WARNING"
# At a minimum 1024MB must be allocated to the swap space
HAVerify.SwapSpace.minsize = "1024MB"

# At a minimum 512MB of space must be available at the time verification
executes
HAVerify.SwapSpace.minfree = "512MB"
```

その他の概念および機能

このセクションでは、Smart Assist の開発者が使用できる追加の機能について説明します。

クラスターの自動定義

クラスターが定義されていない場合は、「**PowerHA SystemMirror** 構成へのアプリケーションの追加 **(Add an Application to the PowerHA SystemMirror Configuration)**」メニューの選択が行われたときに、Smart Assist フレームワークがクラスターのノードについてエンド・ユーザーに自動的に照会し、クラスターを作成します。このステップは、「**PowerHA SystemMirror** クラスターおよびノードの構成 **(Configure a PowerHA SystemMirror Cluster and Nodes)**」SMIT 機能に相当します。

API で使用されるコマンドの照会およびリスト

Smart Assist の開発者は、いくつかのユーティリティー・コマンドを使用できます。次のルーチンの詳細については、『Smart Assist コマンド』を参照してください。

- **cllsssaapp** - 構成されている Smart Assist アプリケーション・インスタンスをリストする
- **cllsserviceips** - 使用可能なサービス IP ラベルをリストする
- **clquerysa** - 指定された Smart Assist の属性を返す
- **clquerysaapp** - 指定された Smart Assist アプリケーション・インスタンスの属性を返す

clsapre ルーチンと **clsapost** ルーチンは、Smart Assist アプリケーション・インスタンスを作成する前と後に呼び出す必要があります。これらのルーチンは、このような操作の間にフックをフレームワークに返します。

```
clsapre [ -c cluster_name ]
-c cluster_name クラスターの名前を指定された値に変更します
clsapost [ -v ]
-v アプリケーション・インスタンスが追加された後にクラスター検証を実行します
```

登録および登録解除コマンド

Smart Assist の開発者は、**claddsa** 呼び出しに指定した REGISTRATION_COMMAND および DEREGISTRATTION_COMMAND を使用して、Smart Assist のインストール前および削除前のアクティビティーを実行することができます。

Smart Assist マイグレーション・コマンド

Smart Assist の開発者は、**claddsa** 呼び出しに指定した MIGRATION_COMMAND を使用して、古い Smart Assist を新しい Smart Assist に置き換える際にマイグレーション・スクリプトを指定することができます。**claddsa** 呼び出しが実行された際に SMARTASSIST_VERSION の値が一致しない場合は、新しい Smart Assist に指定された MIGRATION_COMMAND が呼び出されて、必要に応じて Smart Assist アプリケーション・インスタンスの既存のデータを新しいフォーマットにマイグレーションするか、既存のデータを変更します。

Smart Assist テスト・コマンド

claddsa 呼び出しに TEST_COMMAND が指定されている場合は、エンド・ユーザーが「高可用性のためのアプリケーションのテスト **(Test Your Application for Availability)**」メニュー項目を選択したときにこのスクリプトが呼び出されます。テスト・スクリプトには、最初の引数としてテストするアプリケーションの名前が指定されます。テスト・スクリプトは、クラスター・テスト・ツールのスクリプト形式に準拠した STDOUT 出力を生成しなければなりません。TEST_COMMAND が指定されていない場合は、ユーザーが「Smart Assist ディスカバリー・データベースの名前と値の計画」のテストを選択したときにデフォルトのテスト・スクリプトが実行されます。

関連概念:

21 ページの『Smart Assist コマンド』

PowerHA SystemMirror のために Smart Assist を開発する際に使用するコマンドについては、これらのトピックを参照してください。それぞれのトピックでは、構文図と各コマンドを使用するための例を記載しています。

Smart Assist ディスカバリー・データベースの名前と値の計画

Smart Assist が **installp** または同等の SMIT 機能を使用してインストールされる際、Smart Assist はそれ自体の情報をディスカバリー・データベース PowerHA SystemMirror に追加しなければなりません。これにより、ディスカバリー・プロセスで高可用性にすることができるアプリケーションが確実に検出されます。

ファイルセットに関連付けられたインストール・スクリプトでは、**claddsa** コマンドを使用して Smart Assist の各コンポーネントを登録する必要があります。各 Smart Assist の情報は ODM または同様の構造に格納され、クラスターの同期によってこの情報がクラスターの残りの部分に配布されるようにします。Smart Assist の各コンポーネントでは、以下の情報の一部またはすべてを指定します。この情報は、ディスカバリーおよび他のプロセスに必要です。

項目	タイプ	必須	デフォルト値	注記
Smart Assist ID	文字列 - 32 文字以下	はい		Smart Assist のすべてのコンポーネントに共通でなければなりません
コンポーネント ID	文字列 - 32 文字以下	はい		Smart Assist のコンポーネント間で固有でなければなりません
Smart Assist バージョン	数値	はい		旧バージョンより大きくなければなりません
ディスカバリー・コマンド	絶対ファイル・パスおよび引数	はい		ディスカバリー・スクリプトの説明および要件を参照してください。
アンインストール・コマンド	絶対ファイル・パスおよび引数	いいえ	なし	Smart Assist を完全にアンインストールするために特別なステップを実行する場合に使用されます。
登録コマンド	絶対ファイル・パスおよび引数	いいえ	なし	このオプションのコマンドは、SMIT の追加パスが呼び出される前に実行されます。
登録解除コマンド	絶対ファイル・パスおよび引数	いいえ	なし	このオプションのスクリプトは、リソース・グループとリソースが PowerHA SystemMirror から、その情報が Smart Assist のデータベースから除去される前に実行されます。
マイグレーション・スクリプト	絶対ファイル・パスおよび引数	いいえ	なし	このオプションのスクリプトは、前バージョンがインストールされていることを **claddsa** が検出した場合に実行されます。
SMIT 追加画面またはセレクター	SMIT パス			
SMIT 追加タイプ				
SMIT 変更画面またはセレクター	SMIT パス			
SMIT 変更タイプ				

項目	タイプ	必須	デフォルト値	注記
リソース・グループ名の生成スクリプト	絶対ファイル・パスおよび引数	はい		登録プロセスにより使用されます。
アプリケーション・コントローラー名の生成スクリプト	絶対ファイル・パスおよび引数	はい		登録プロセスにより使用されます。
カスタム・テスト・スクリプト	絶対ファイル・パス	いいえ	テスト計画を生成するためにアプリケーション・ベースのユーティリティーを使用します。	

Smart Assist コマンド

PowerHA SystemMirror のために Smart Assist を開発する際に使用するコマンドについては、これらのトピックを参照してください。それぞれのトピックでは、構文図と各コマンドを使用するための例を記載しています。

強調表示

このトピックでは、以下の強調表示規則を使用します。

項目	説明
太字	実際の名前がシステムによって事前定義されているコマンド・ワード、キーワード、ファイル、ディレクトリー、および他の項目を示します。
イタリック	実際の名前または値をユーザーが提供する必要があるパラメーターを示します。
モノスペース	特定のデータ値の例、表示される可能性があるテキストの例、プログラマーとして作成する可能性があるプログラム・コードの例、システムからのメッセージ、またはユーザーが実際に入力する必要がある情報を示します。

構文図の読み方

通常、コマンドは次の構文に従います。

項目	説明
[]	大括弧内の要素はオプションです。
{}	中括弧内の要素は必須です。
|	選択を表します。いずれか 1 つのオプションのみを選択できます。
...	省略符号の前に 1 つ以上の種類のパラメーターまたはオブジェクトを入力できます。

関連情報

コマンドの機能と制約事項の詳細については、オンライン・マニュアル・ページを参照してください。PowerHA SystemMirror for AIX のコマンドとユーティリティーに関するマニュアル・ページは、**/usr/share/man/cat1** ディレクトリーにインストールされています。マニュアル・ページの情報を表示するには、次の構文を使用します。ここで、*command-name* は、PowerHA SystemMirror コマンドまたはスクリプトの実際の名前です。

```
man command-name
```

例えば、PowerHA SystemMirror ユーザー・パスワード・コマンドに関する情報を取得するには、**man clpasswd** と入力します。

関連概念:

28 ページの『clvt API』
これらのトピックでは、**clvt** (クラスター仮想化ツール) API について説明します。

Smart Assist の登録および照会

以下のコマンドは、ディスカバリー・データベースと登録データベースに Smart Assist を追加したり、これらのデータベースから Smart Assist を除去したりします。

claddsa コマンド

アプリケーション・ディスカバリーで使用するために Smart Assist を登録します。

構文

```
claddsa -s SmartAssistID -c ComponentID [-C] name1= "value1"
name2=name1= "value1"..
```

パラメーター

-s *SmartAssistID*	**Smart Assist** の固有 **ID**
-c *ComponentID*	Smart Assist コンポーネントの固有 ID
`name1= "value1"...` `nameN= "valueN"`	データベース内に格納される名前/値のペアの名前。
-C	Smart Assist とコンポーネントをローカル・ノードで構成できる (ソフトウェアがインストールされている) かどうか検査します。

マイグレーション・スクリプトを指定した場合、**claddsa** は最初に以前のバージョンの Smart Assist がインストールされているかどうか検査します。コマンド行で指定された値でデータベースが更新されると、マイグレーション・スクリプトが呼び出され、CLSA_VER 環境変数を使用して以前の Smart Assist のバージョンが渡されます。他の値と異なり、MIGRATION_COMMAND は HACMPsa ODM には格納されないことに注意してください。以下のリストでは、Smart Assist フレームワークが認識する名前および値のペアを詳しく説明しています。Smart Assist 全体にかかわるには、これらすべてに Smart Assist ID とコンポーネント ID の両方が必要です。注: 必ず、使用する Smart Assist ID とコンポーネント ID が固有であることを確認してください。

名前	説明
SMARTASSIST_ID	Smart Assist の固有 ID
COMPONENT_ID	Smart Assist 内でコンポーネントを識別する固有の文字列
SMARTASSIST_VERSION	Smart Assist バージョンを識別する固有の文字列。これはマイグレーションが必要かどうか判別するために使用されます。
MIGRATION_COMMAND	同じ Smart Assist の古いバージョンが既にインストールされている場合、古いバージョンを削除する前に新しいマイグレーション・スクリプトが実行されます。このスクリプトは、PowerHA SystemMirror に定義されている既存のすべてのアプリケーション・インスタンスを新しい Smart Assist フォーマットにマイグレーションします。注: 他の値と異なり、この項目の値は HACMPsa ODM には格納されません。
SMIT_MODIFY_TYPE	sm_cmd_hdr 画面の場合は「d」、または sm_name_hdr 画面の場合は「n」
SMIT_MODIFY	特定の Smart Assist コンポーネントを変更する際に、名前セレクター画面、または cmd hdr ダイアログ画面にナビゲートするために使用する SMIT ID
DISCOVERY_ COMMAND	特定のアプリケーション・コンポーネントまたはアプリケーションのサブフィーチャーがローカル・クラスター・ノードでアクセス可能であるかどうかディスカバーするためのコマンド
DEINSTALLATION_ COMMAND	Smart Assist がローカル・ノードから除去される前に実行されるコマンド

名前	説明
REGISTRATION_ COMMAND	ユーザーがアプリケーション・インスタンスを PowerHA SystemMirror 構成に追加するときに実行されるコマンド
DEREGISTRATION_ COMMAND	Smart Assist フレームワークが PowerHA SystemMirror 構成からアプリケーションを除去する前に実行されるコマンド
SMIT_ADD_TYPE	sm_cmd_hdr 画面の場合は「d」、または sm_name_hdr 画面の場合は「n」
SMIT_ADD	ユーザーが選択した Smart Assist コンポーネントの Smart Assist 追加画面にナビゲートするために使用する SMIT ID。

必要に応じて、Smart Assist コンポーネントの名前値ペアをさらに追加できます。上記のいずれでもない名前および値のペアは、Smart Assist フレームワークでは解釈されません。**-C** が指定された場合、最後の手順は DISCOVER_COMMAND の実行です (指定されている場合)。このコマンドが情報を返す場合は、次の形式のメッセージを出力します。

```
The following components of <Smart Assist> can be configured on the
local node because the software is installed:
      <component 1>
      <component 2>
      ...
Please run "smit clsa" to start configuring them to make them highly
available with PowerHA SystemMirror.
```

claddsa の例

```
claddsa -s MYSMARTASSIST_8.0 -c FIRST_COMPONENT ¥
SMARTASSIST_ID="MYAPP_8.0" ¥
COMPONENT_ID="FIRST_COMPONENT" ¥
DISCOVERY_COMMAND="/usr/es/sbin/cluster/sa/MyApp/sbin/discovery"¥
DEINSTALLATION_COMMAND="/usr/es/sbin/cluster/sa/MyApp/install/uninstall"
¥
REGISTRATION_COMMAND="/usr/es/sbin/cluster/sa/MyApp/sbin/register"¥
DEREGISTRATION_COMMAND="/usr/es/sbin/cluster/sa/sbin/MyApp/deregister" ¥
SMARTASSIST_VERSION="1.0" ¥
SMIT_ADD_TYPE="n" ¥
SMIT_ADD="clsa_mysmartassist_add_selector"
SMIT_MODIFY_TYPE="d" ¥
SMIT_MODIFY="clsa_mysmartassist_modify_dialog"
```

結果として HACMPsa ODM 項目は次のようになります。

```
HACMPsa:
  sa_id ="MYSMARTASSIST_8.0"
  component_id ="FIRST_COMPONENT"
  name="SMARTASSIST_ID"
  value="MYSMARTASSIST_ID"
HACMPsa:
  sa_id ="MYSMARTASSIST_8.0"
  component_id ="FIRST_COMPONENT"
  name="COMPONENT_ID"
  value="FIRST_COMPONENT"
HACMPsa:
  sa_id ="MYSMARTASSIST_8.0"
  component_id ="FIRST_COMPONENT"
  name="DEREGISTRATION_COMMAND"
  value="/usr/es/sbin/cluster/sa/sbin/MyApp/deregister"
HACMPsa:
  sa_id ="MYSMARTASSIST_8.0"
  component_id ="FIRST_COMPONENT"
  name="SMARTASSIST_VERSION"
  value="1.0"
HACMPsa:
  sa_id ="MYSMARTASSIST_8.0"
```

```
  component_id ="FIRST_COMPONENT"
  name="SMIT_ADD_TYPE"
  value="n"
HACMPsa:
  sa_id ="MYSMARTASSIST_8.0"
  component_id ="FIRST_COMPONENT"
  name="SMIT_ADD"
  value=" clsa_mysmartassist_add_selector"
HACMPsa:
  sa_id ="MYSMARTASSIST_8.0"component_id ="FIRST_COMPONENT"
  name="SMIT_MODIFY_TYPE"
  value=" clsa_mysmartassist_add_selector"
HACMPsa:
  sa_id ="MYSMARTASSIST_8.0"
  component_id ="FIRST_COMPONENT"
  name="SMIT_MODIFY"
  value=" clsa_mysmartassist_add_selector"
HACMPsa:
  sa_id ="MYSMARTASSIST_8.0"
  component_id ="FIRST_COMPONENT"
  name="DISCOVERY_COMMAND"
  value="/usr/es/sbin/cluster/sa/MyApp/sbin/discovery"
HACMPsa:
  sa_id ="MYSMARTASSIST_8.0"
  component_id="FIRST_COMPONENT"
  name="REGISTRATION_COMMAND"
  value="/usr/es/sbin/cluster/sa/MyApp/sbin/register"
```

clquerysa コマンド

このコマンドは、使用可能な Smart Assist コンポーネントについてクラスターのノードを照会し、さらにコンポーネントがリモート・ノードで実行できるかどうか照会します。

-t discover : 使用可能な Smart Assist コンポーネントについてクラスターのノードを照会し、さらにコンポーネントがリモート・ノードで実行できるかどうか照会します。

-t filter : Smart Assist の選択用に SMIT で使用できるリストを生成するために Smart Assist [およびコンポーネントの両方または一方] のロー・リストをフィルタリングします。このロー・リストは、標準入力で渡す必要があります。

ディスカバリーの構文

```
clquerysa -t discover [-n nodename]
```

ディスカバリーのパラメーター

パラメーター	説明
-n *nodename*	**-n** が指定されている場合は、リモート・ノードからの情報のみが返されます。

注**:** `clquery -t discover` は、次のコマンドと同等です。

クラスター内の各ノードに対して

```
   clquerysa -t discover -n $node
```

ディスカバリーの出力

結果として生成される出力は、インストールされている各コンポーネントごとに次の情報を含む 1 行のテキストです。

項目	説明
Node	ディスカバリーによりコンポーネントが検出されたノードの名前。
SmartAssistName	Smart Assist の国際化された名前。
SmartAssistID	Smart Assist の ID。これはディスカバリー DB 内の ID と一致しなければなりません。
ComponentName	Smart Assist コンポーネントの国際化された名前。
ComponentID	コンポーネントの ID。これはディスカバリー DB 内の ID と一致しなければなりません。
[0|1]	コンポーネントがインストールされていない場合は 0、インストールされている場合は 1。

フィルタリングの構文

```
clquerysa -t filter
```

Smart Assist の選択用に SMIT で使用できるリストを生成するために Smart Assist およびコンポーネントのロー・リストをフィルタリングします。Smart Assist の名前と関連ノードのみが表示されます (ユーザーが「**PowerHA SystemMirror** 構成へのアプリケーションの追加 **(Add an Application to the PowerHA SystemMirror Configuration**)」を選択した後にセレクターが表示されます)。

```
DB2 UDB non-DPF Smart Assistant
 Oracle Smart Assist #
WebSphere Smart Assistant #
Other Applications # NodeA NodeB

clquerysa -t filter -s SmartAssistID
```

Smart Assist コンポーネントの選択用に SMIT で使用できるリストを生成するために Smart Assist およびコンポーネントのロー・リストをフィルタリングします。特定の Smart Assist の使用可能なコンポーネントのみが次のように示されます。

```
DB2_8.0
DB2 Hot Standby
DB2 Mutual Takeover

clquerysa -t filter -s SmartAssistID -c Component ID
```

Smart Assist のアプリケーション・コンポーネントのノードのリストのみがコンマ区切りリストとして表示されるように、Smart Assist およびコンポーネントのロー・リストをフィルタリングします。

clrmsa コマンド

Smart Assist のすべてのコンポーネントまたは指定されたコンポーネントを除去します。

構文

clrmsa -s *SmartAssistID -c ComponentID*

パラメーター

パラメーター	説明
-s *SmartAssistID*	除去する Smart Assist の固有 ID。
-c *componentID*	除去するコンポーネントの ID。

最初に、**clrmsa** はすべてのコンポーネント (1 つのコンポーネントを指定しない場合) に対してアンインストール・コマンドを実行します。次に、ディスカバリー・データベースおよび登録データベース HACMPsa および HACMPsa_metadata から Smart Assist に関するすべての情報を除去します。このコ

マンドは、PowerHA SystemMirror 構成からアプリケーションは除去せずに、関連した Smart Assist 構成のみを除去します。これにより、ユーザーは引き続き通常の SMIT PowerHA SystemMirror メニューを使用してアプリケーションを管理できます。

clrmsa の例

Smart Assist zzother を除去する場合: clrmsa -s zzOther

Smart Assist アプリケーションの登録および照会

以下のコマンドは、アプリケーションのディスカバリー・データベースと登録データベースに Smart Assist を追加したり、これらのデータベースから Smart Assist を除去したりします。

claddsaapp コマンド

アプリケーションに関する情報を **HACMPsa** データベースに登録します。

構文

```
claddsaapp -a [ApplicationID ] name1="value1" name2="value2" ...
nameN="valueN"
```

パラメーター

パラメーター	説明
-a	**-a** スイッチのみで **claddsaapp** コマンドを呼び出すと、次に使用可能なアプリケーション ID がリストされます。
-a *ApplicationID*	*name=value* ペアに含まれるアプリケーションの名前をリストします。
name=value のペア	アプリケーション・コンポーネントの名前。

このコマンドに必要な名前/値のペアは、SMARTASSIST_ID、COMPONENT_ID、および APPLICATION_NAME です。

項目	説明
SMARTASSIST_ID	この特定のアプリケーション・インスタンスが実装する Smart Assist ID。
COMPONENT_ID	Smart Assist コンポーネント ID。
APPLICATION_NAME	この値は **-a** フラグまたは ApplicationID に渡される値と同じです。

claddsaapp の例

```
claddsaapp -a MyApp ¥
    SMARTASSIST_ID="zzOther" ¥
    COMPONENT_ID="GASA" ¥
    APPLICATION_NAME="MyApp" ¥
    RESOURCE_GROUP="MyApp_group"
```

clquerysaapp コマンド

登録済みアプリケーションに関する情報を返します。

構文

```
clquerysaapp -a applicationID name1 ... nameN
```

パラメーター

パラメーター	説明
-a *ApplicationID*	*name=value* ペアに含まれるアプリケーションの名前をリストします。
name=value のペア	アプリケーション・コンポーネントの名前。

名前が指定されている場合は、その名前に関連付けられた値のみが *name =VALUE* の形式で返されます (複数の値がある場合は、コンマで区切られます)。複数の名前が指定されている場合は、*NAME1=VALUE1 ... NAMEN=VALUEN* と表示されます。名前を指定しないと、適用可能なすべての名前がコマンド行に指定されたかのようにすべての情報が返されます。ただし、定義された順序ではありません。

clquerysaapp の例

MyApp のすべての名前/値を照会する場合:

```
clquerysaapp -a MyApp
SMARTASSIST_ID="zzOther"
COMPONENT_ID="GASA"
RESOURCE_GROUP="MyApp_group"
APPLICATION_NAME="MyApp"
```

MyApp の Smart Assist の名前を照会する場合:

```
clquerysaapp -a MyApp SMARTASSIST_ID
SMARTASSIST_ID="zzOther"
```

clrmsaapp コマンド

登録済み Smart Assist に関する特定の情報またはすべての情報を除去します。

構文

```
clrmsaapp -a applicationID
clrmsaapp -a applicationID [name1|name1=VALUE1] ... [nameN|nameN=VALUEN]
clrmsaapp -s SmartAssistID
```

パラメーター

パラメーター	説明
-a *ApplicationID*	*name=value* ペアに含まれるアプリケーションの名前をリストします。
name=value のペア	アプリケーション・コンポーネントの名前。
-s	特定の Smart Assist ID で構成されているすべてのアプリケーションに関する情報を除去します。

コンポーネントを登録し、さらに構成すると、アプリケーションを検証して同期する必要があります。これにより、登録データベースがクラスターのすべてのノードと確実に同期されます。

clrmsaapp の例

MyApp に関するすべての情報を除去する場合:

```
Clrmsaapp -a MyAPP
```

cllssaapp コマンド

Smart Assist により構成されるアプリケーションのリストを返します。

構文

```
cllsaapp
```

出力例

```
DB2_UDB_db2inst1
DB2_UDB_db2inst2
OracleAFC_ias10g
OracleCFC_ias10g.
```

SMIT パネルのコンビニエンス・ルーチン

これらのルーチンは、Smart Assist アプリケーション・インスタンスを作成する前と後に呼び出す必要があります。これらのルーチンは、このような操作の間にフックをフレームワークに返します。

`clsapre [ -c cluster_name ]`

パラメーター	説明
-c *clustername*	クラスターの名前をこの値に変更します。

`clsapost [ -v ]`

パラメーター	説明
-v	アプリケーション・インスタンスが追加された後にクラスター検証を実行します。

---

clvt API

これらのトピックでは、**clvt** (クラスター仮想化ツール) API について説明します。

clvt コマンドは、PowerHA SystemMirror クラスターの操作を実行します。クラスターの操作には、クラスター・オブジェクトの追加、削除、および照会があります。さらにノードをオンラインまたはオフラインにしたり、リソース・グループとインターフェースを変更したりします。

clvt コマンドの一般的な構文は次のとおりです。

```
clvt action class object name [name=value]...
```

すべてのアクション・クラスの組み合わせに、オブジェクトまたは name=value ペアが必要であるわけではありません。

すべてのコマンドに出力があるわけではありません。出力がある場合は記載します。

ユーザーが正しくない値を入力した場合、戻りコードはゼロ以外となります。すべてのコマンドがエラー・メッセージを生成するわけではありません。

構文の一般規則

[] 大括弧内の要素はオプションです。

{} 中括弧内の要素は必須です。

イタリック (フラグなし) - オブジェクトの名前が必要です。

| 選択を表します。いずれか 1 つのオプションを使用します。

... 省略符号の前に 1 つ以上の種類のオブジェクトまたはパラメーターを入力できます。

クラスター・クラスの操作

clvt を使用してクラスターの操作を実行します。

add cluster

指定された名前を使用して PowerHA SystemMirror クラスターを作成します。クラスター名が既に存在する場合は、ゼロ以外を返します。名前を指定しなかった場合、クラスター名は cluster_*nodename* となります。ここで、*nodename* はコマンドを実行するノードです。

構文

```
clvt add cluster [cluster_name]
```

cluster_name クラスター名には、最大 32 文字の英数字と下線が含まれます。

例

ClusterA という名前のクラスターを作成する場合:

```
clvt add cluster clusterA
```

delete cluster

現在定義されているクラスターを削除します。指定したクラスターが存在しない場合、または既存のクラスターと一致しない場合は、ゼロ以外を返します。

構文

```
clvt delete cluster
```

例

クラスターを削除する場合:

```
clvt delete cluster
```

query cluster

クラスターに関する情報を返します。クラスターが存在しない場合は、ゼロ以外を返します。

構文

```
clvt query cluster
```

例

```
clvt query cluster

SECURITY="Standard"
CLUSTER_ID="1146018839"
STATE="ST_INIT"
CLUSTER_NAME="regaa11_cluster"
```

sync cluster

クラスターの検証および同期を実行します。検証中に検出されたエラーは自動的に修正されます。クラスターが存在しない場合は、ゼロ以外を返します。

構文

```
clvt sync cluster
```

例

クラスターを検証して同期する場合:

```
clvt sync cluster
```

discover cluster

指定されたクラスターに構成されているノードから PowerHA SystemMirror 関連情報のクラスター・ディスカバリーを実行します。クラスターが存在しない場合は、ゼロ以外を返します。

構文

```
clvt discover cluster
```

例

現行のクラスターをディスカバーする場合:

```
clvt discover cluster
```

ノード・クラスの操作

clvt を使用してクラスター・ノードの操作を実行します。

add node

指定された名前のノードをクラスターに追加します。

構文

```
clvt add node node_name COMMPATH =IPaddress | IPlabel
```

パラメーター

パラメーター	説明
node_name	定義されているクラスター・ノードの名前。
PATH	ノードへの通信パス。

必須オブジェクト

クラスター

エラー

- クラスターが定義されていない
- ノード名が既に存在する
- 最大数のノードが既に存在する

例

node1 という名前のノードをその IP アドレスを使用してクラスターに追加する場合:

```
clvt add node node1 COMMPATH=10.10.2.2
```

delete node

クラスターから指定されたノードを削除します。

構文

```
clvt delete node node_name
```

パラメーター

パラメーター	説明
node_name	既存のクラスター・ノードの名前。

必須オブジェクト

- クラスター
- ノード

エラー

- クラスターが定義されていない
- ノードが存在しない

例

クラスターから node2 を削除する場合:

```
clvt delete node node2
```

query node

ノード名が指定されている場合、このコマンドはディスカバーされたノード属性の情報を返します。ノード名が指定されていない場合、このコマンドはすべてのノードのリストを返します。

構文

```
clvt query node [node_name ]
```

パラメーター

パラメーター	説明
node_name	既存のクラスター・ノードの名前。

必須オブジェクト

クラスター

エラー

- クラスターが定義されていない
- ノードが存在しない

出力

ノード名が指定されている場合は、次の項目に関する情報が返されます。

- ノード名
- 通信パス
- ATTR = (パブリックまたはプライベート)
- IPADDRESS = ブート・アドレスおよびインターフェース・アドレス

- IPLABEL = ブート・ラベルおよびインターフェース・ラベル
- NETMASK=各パスごと
- NETTYPE = ネットワークのタイプ
- NETWORK = ネットワークの名前

例

```
clvt query node

  maple
  elm
  oak
clvt query node regaa11 | sort
ATTR1="public"
ATTR2="public"
ATTR3="public"
IPADDRESS1="192.168.210.11"
IPADDRESS2="192.168.220.11"
IPADDRESS3="10.70.33.11"
IPLABEL1="regaa11_base10"
IPLABEL2="regaa11_base20"
IPLABEL3="regaa11"
NAME="regaa11"
NETMASK1="255.255.255.0"
NETMASK2="255.255.255.0"
NETMASK3="255.255.0.0"
NETTYPE1="ether"
NETTYPE2="ether"
NETTYPE3="ether"
NETWORK1="net_ether_01"
NETWORK2="net_ether_01"
NETWORK3="net_ether_00"
```

online node

指定されたノードのクラスター・マネージャーを開始します。クラスター・マネージャーが既に開始している場合、コマンドは無視されます。

構文

```
clvt online node node_name
```

パラメーター

パラメーター	説明
node_name	オンラインにするノード。

必須オブジェクト

- クラスター
- ノード

エラー

- クラスターが定義されていない
- ノードが存在しない

例

node2 をオンラインにする場合:

```
clvt online node node2
```

offline node

指定されたノードのクラスター・マネージャーを停止します。クラスター・マネージャーが既に停止している場合、コマンドは無視されます。

構文

```
clvt offline node node_name
```

パラメーター

パラメーター	説明
node_name	既存のクラスター・ノードの名前。

必須オブジェクト

- クラスター
- ノード

エラー

- クラスターが定義されていない
- ノードが存在しない

例

node2 をオフラインにする場合:

```
Clvt offline node node2
```

インターフェース・クラスの操作

clvt を使用してインターフェースの操作を実行します。

modify interface

インターフェースが結合されるネットワークを変更します。

構文

```
clvt modify interface iplabel NETWORK=networkname
```

パラメーター

パラメーター	説明
iplabel	インターフェースの名前。
networkname	インターフェースが結合されるネットワーク。

必須オブジェクト

- クラスター
- node
- インターフェース

エラー

- インターフェースが存在しない
- ネットワークが存在しない

例

mylabel という名前の iplabel をネットワーク ether2 に結合する場合:

```
clvt modify interface mylabel NETWORKNAME=ether2
```

query interface

インターフェース・ラベルに関する情報を返します。インターフェース・ラベルが指定されていない場合は、すべてのインターフェース・ラベルのリストを返します。

構文

```
clvt query interface [interface_label ]
```

パラメーター

パラメーター	説明
interface_label	インターフェースの名前。

必須オブジェクト

クラスター

エラー

- クラスターが定義されていない
- ノードが存在しない

出力

インターフェース・ラベルが指定されていない場合は、インターフェース・ラベルのリストを表示します。インターフェース・ラベルが指定されている場合は、次の情報を返します。

- インターフェース・ラベル名
- インターフェース・タイプ
- ネットワーク
- ネットワーク・タイプ
- ネットワーク属性 (パブリックまたはプライベート)

- ノード
- IP アドレス
- ハードウェア・アドレス
- インターフェース名

例

すべてのインターフェースをリストする場合:

```
clvt query interface
  maple_base_10
  maple_base_20
  elm_base_10
  elm_base_20
```

インターフェース regaa11_base10 の属性をリストする場合:

```
clvt query interface regaa11_base10
  INTERFACENAME="en0"
  NETTYPE="ether"
  TYPE="boot"
  ATTR="public"
  IPADDR="192.168.210.11"
  NODE="regaa11"
  GLOBALNAME=""
  HADDR=""
  NETWORK="net_ether_01"
  NAME="regaa11_base10"
```

ネットワーク・クラスの操作

clvt を使用してクラスター・ネットワークの操作を実行します。

add network

ネットワークをクラスターに追加します。

構文

```
clvt add network networkname NETWORKTYPE=networktype [NETMASK=netmask]
```

パラメーター

パラメーター	説明
networkname	追加するネットワークの名前。
NETWORKTYPE	追加するネットワークのタイプ。選択項目は、「ether」、「XD_ip」、または「XD_data」です。
NETMASK	ネットワークのネットマスク。

必須オブジェクト

クラスター

エラー

- クラスターが存在しない
- 無効なネットワーク・タイプ
- 無効なネットマスク

例

以下のように、ネットマスクが 255.255.128 で、タイプ ether の ether_01 という名前のネットワークを追加します。

```
clvt add network ether_01 NETWORKTYPE=ether ¥
                          NETMASK=255.255.255.128 ¥
```

delete network

事前に定義されているネットワークを削除します。

構文

```
clvt delete network networkname
```

パラメーター

パラメーター	説明
networkname	削除するネットワークの名前。

必須オブジェクト

クラスター・ネットワーク

エラー

ネットワークが存在しない

例

クラスターから ether1 という名前のネットワークを削除する場合:

```
clvt delete network ether1
```

query network

指定されたネットワークに関する情報を返します。ネットワーク名が指定されていない場合は、すべてのネットワークのリストを返します。

構文

```
clvt query network [networkname ]
```

パラメーター

パラメーター	説明
networkname	照会するネットワークの名前。

必須オブジェクト

クラスター

エラー

- クラスターが定義されていない
- ネットワークが存在しない

出力

ネットワーク名が指定されていない場合は、すべてのクラスター・ネットワークをリストします。ネットワーク名が指定されている場合は、次のネットワーク属性をリストします。

```
ALIAS="true | false"
ALIAS_HB_ADDR= " "
ALIAS_HB_NETMASK=""
ATTR="public | private"
GLOBALNAME=""
MONITOR_METHOD="default | custom"
NAME="network_name"
NETMASK=""
NETWORK_ID=""
NIMNAME=""
POLLINTERVAL=""
```

例

クラスター・ネットワークをリストする場合:

```
clvt query network
  net_XD_data_01
  net_XD_ip_01
  net_ether_01
  token_01
```

net_ether_01 という名前のネットワークの属性をリストする場合:

```
clvt query network net_ether_01
  ALIAS="aliased"
  ATTR="public"
  POLLINTERVAL="0"
  NIMNAME="ether"
  NETWORK_ID="0"
  GLOBALNAME=""
  NAME="net_ether_01"
  ALIAS_HB_NETMASK=""
  NETMASK="255.255.255.0"
  ALIAS_HB_ADDR=""
  MONITOR_METHOD="default"
```

リソース・グループ・クラス

リソース・グループ・クラスは少し異なり、リソース・グループをクラスターに追加する場合に **clvt add** アクションが使用され、リソース・グループの属性とリソースを追加または変更する場合に **clvt modify** アクションが使用されます。 **clvt** を使用してリソース・グループの操作を実行します。

add resource_group

リソース・グループをクラスターに追加します。クラスターとノードが定義されていない場合、リソース・グループを追加するとこれらが定義され、クラスター・ディスカバリーが実行されます。クラスター名は、リソース・グループ名から生成されます。startup、fallover、または fallback ポリシーが定義されていない場合は、デフォルトが使用されます。

構文

```
clvt add resource_group resource_group_name PRIMARYNODES="primary node
list" [STARTUP="startup policy"] [FALLBACK="fall back policy"]
[FALLOVER="fall over policy"]
```

パラメーター

パラメーター	説明
Resource_group_name	32 文字以下の英数字の文字列。
PRIMARYNODES	リソース・グループのノード・リスト。ノード・リストは優先順位順です。
STARTUP	リソース・グループの始動ポリシー。指定可能な値は次のとおりです。 • OHN - ホーム・ノードでオンライン (デフォルト) • OFAN - 最初に使用可能なノードでオンライン • OAAN - 使用可能なすべてのノードでオンライン • OUDP - 配布ポリシーを使用してオンライン
FALLBACK	リソース・グループのフォールバック・ポリシー。指定可能な値は次のとおりです。 • NFB - フォールバックしない (STARTUP = OAAN の場合はデフォルト) • FBHPN - リスト中のより高い優先順位のノードにフォールバック (STARTUP!= OAAN の場合はデフォルト)
FALLOVER	リソース・グループのフォールオーバー・ポリシー。指定可能な値は次のとおりです。 • FNPN - リスト中の次の優先順位のノードにフォールオーバー (STARTUP != OAAN の場合はデフォルト) • FUDNP - 動的ノード優先順位を使用してフォールオーバー • BO - オフラインにする (エラー・ノードのみ) (STARTUP = OAAN の場合はデフォルト)

エラー

- ノードをクラスターに追加できなかった
- 無効な始動ポリシー
- 無効なフォールバック・ポリシー
- 無効なフォールオーバー・ポリシー

例

始動ポリシー OFAN、フォールバック・ポリシー NFB、およびフォールオーバー・ポリシー FNPN を使用して、Apple および Basket ノードで実行される myDBapp という名前のリソース・グループを追加する場合:

```
clvt add resource_group myDBapp NODE_LIST="Apple, Basket"
[STARTUP="OFAN"] [FALLBACK="NFB"] [FALLOVER="FNPN"]
```

delete resource_group

クラスターから指定されたリソース・グループを削除します。

構文

```
clvt delete resource_group resource_group_name
```

パラメーター

パラメーター	説明
resource_group_name	削除するリソース・グループ。

必須オブジェクト

- クラスター
- リソース・グループ

エラー

- クラスターが定義されていない
- リソース・グループが存在しない

例

リソース・グループ MyDBapp を削除する場合:

```
clvt delete resource_group MyDBapp
```

modify resource_group

リソース・グループに含まれる属性とリソースを追加または変更します。

構文

```
clvt modify resource_group <resource group name>
[ SERVICE_LABEL="service1 service2..." ]
[ APPLICATIONS="app1 app2..." ]
[ VOLUME_GROUP="vg1 vg2 ..." ]
[ FORCED_VARYON="true | false" ]
[ VG_AUTO_IMPORT="true | false" ]
[ FILESYSTEM="/fs1 /fs2 ..." ]
[ FS_BEFORE_IPADDR="true | false" ]
[ EXPORT_FILESYSTEM="/expfs1 /expfs2 ..." ]
[ MOUNT_FILESYSTEM="/nfs_fs1 /nfs_fs2 ..." ]
[ NFS_NETWORK="nfs_network" ][ DISK="hdisk1 hdisk2 ..." ]
```

パラメーター

パラメーター	説明
SERVICE_LABEL	定義されているサービス・ラベルとアドレスのコンマ区切りリスト
APPLICATIONS	定義されているアプリケーション・コントローラーのコンマ区切りリスト
VOLUME_GROUP	varyon (オンに変更) するボリューム・グループのコンマ区切りリスト
FORCED_VARYON	ボリューム・グループを varyon するために varyon の強制を使用する場合は TRUE
VG_AUTO_IMPORT	= "true | false"
FILESYSTEM	マウントするファイルシステム。(デフォルトは ALL)
FS_BEFORE_IPADDR	= "true | false"
EXPORT_FILESYSTEM	リソースが最初に取得されたときにリソース・チェーン内のすべてのノードにエクスポートされるファイルシステムとディレクトリーのマウント・ポイント。
NFS_NETWORK	NFS マウントの推奨ネットワーク
MOUNT_FILESYSTEM	現在リソースを保持していないリソース・チェーン内のすべてのノードが、これらのファイルシステムを NFS マウントしようとします。
DISK	= "hdisk1 hdisk2..."

必須オブジェクト

- クラスター
- リソース・グループ

エラー

- クラスターが定義されていない
- リソース・グループが存在しない
- サービス・ラベルが定義されていない
- アプリケーション・コントローラーが存在しない

例

- リソース・グループ MyDBapp で varyon するようにボリューム・グループを変更する場合:

  ```
  clvt modify resource_group MyDBGapp VOLUME_GROUP=vgmyDB1,vgmyDB3
  ```

- リソース・グループ MyDBapp の NFS ネットワークとアプリケーション・コントローラーを変更する場合:

  ```
  clvt modify resource_group myDBapp NFS_NETWORK=ether2
  APPLICATIONS=appserv1,appserv3
  ```

query resource_group

指定されたリソース・グループに関する情報を返します。返された情報には、追加アクションと変更アクションからの両方の項目が含まれます。リソース・グループ名が指定されていない場合、このコマンドはすべてのリソース・グループのリストを返します。

構文

```
clvt query resource_group [resource_group_name]
```

パラメーター

パラメーター	説明
resource_group_name	情報をリストするリソース・グループの名前。

必須オブジェクト

クラスター

エラー

- クラスターが定義されていない
- リソース・グループが存在しない

出力

未変更のリソース・グループの名前が指定されている場合は、そのリソース・グループの基本情報が出力されます。

- 名前
- ノード・リスト
- 始動、フォールオーバー、およびフォールバックのポリシー
- 状態

変更されたリソース・グループの名前が指定されている場合は、すべてのリソース・グループ属性に関する完全な情報を返します。

```
Disk
Volume Group
Concurrent Volume Group
Forced Varyon
Filesystem
Export Filesystem
Shared Tape Resources
Communication Links
Applications
Mount Filesystem
Service Label
VG Auto Import
Mount Filesystem before IP Address
NFS Network
Node Priority Policy
Mount All Fs
Fallback At
Relationship
Secondary Nodes
Primary Nodes
Startup Policy
Fallover Policy
Fallback Policy
Resource Group State
```

例

```
clvt query resource_group
  rg1
  rg2
  rg3

clvt query resource_group RG1
  DISK= "hdisk3"
  EXPORT_FILESYSTEM="/FSa1"
  FALLBACK="FBHPN"
  FALLOVER="FNPN"
  FILESYSTEM="/FSa1"
  FORCED_VARYON="false"
  FSCHECK_TOOL="fsck"
  FS_BEFORE_IPADDR="true"
  MOUNT_FILESYSTEM="/mnt1;/FSa1"
  NAME="RG1" NFS_NETWORK=""
  NODES="regaa11 regaa12"
  RECOVERY_METHOD="sequential"
  SERVICE_LABEL="alias_svc1"
  SSA_DISK_FENCING="false"
  STARTUP="OHN" STATE=""
  VG_AUTO_IMPORT="true"
  VOLUME_GROUP="vg1"
```

online resource_group

指定されたリソース・グループをオンラインにします。リソース・グループが既にオンラインである場合、コマンドは無視されます。

構文

```
clvt online resource_group resource_group_name
```

パラメーター

パラメーター	説明
resource_group_name	オンラインにするリソース・グループの名前。

必須オブジェクト

- クラスター
- リソース・グループ

エラー

- クラスターが定義されていない
- リソース・グループが存在しない

例

MyDBapp1 という名前のリソース・グループをオンラインにする場合:

```
clvt online resource group MyDBapp1
```

offline resource_group

リソース・グループをオフラインにします。リソース・グループが既にオフラインである場合、コマンドは無視されます。リソース・グループ名を指定する必要があります。

構文

```
clvt offline resource_group resource_group_name
```

パラメーター

パラメーター	説明
resource_group_name	オフラインにするリソース・グループの名前。

必須オブジェクト

- クラスター
- リソース・グループ

エラー

- クラスターが定義されていない
- リソース・グループが存在しない

例

MyDBapp1 という名前のリソース・グループをオフラインにする場合:

```
clvt offline resource group MyDBapp1
```

サービス IP クラス

clvt を使用してサービス IP の操作を実行します。

add service_ip

共有またはノード結合のサービス IP ラベルをクラスターに追加します。このラベルは後でリソース・グループに追加できます。

構文

```
clvt add service_ip ip_label_name NETWORKNAME=networkname SHARED="true |
false" [NODENAME=nodename]
```

パラメーター

パラメーター	説明
ip_label_name	IP ラベルの名前。
NETWORKNAME	IP ラベルおよびアドレスが接続されるネットワークの名前。
SHARED	サービス IP ラベルが複数のノードで構成可能である場合は「True」。サービス IP ラベルがノード結合である場合は「False」。
NODENAME	サービス IP ラベルが結合されるノード。

必須オブジェクト

- クラスター
- ネットワーク

エラー

- クラスターが存在しない
- 無効なネットワーク名
- 未定義のノード名
- 非共有サービス IP ラベルにノード名が必要

例

```
clvt add service_ip {LabelA} NETWORK={ServiceNetwork} ¥
                             SHARED=false ¥
                             NODENAME={NodeA}
```

delete service_ip

事前に定義されているサービス IP ラベルを削除します。

構文

```
clvt delete service_ip service_IP_label
```

パラメーター

パラメーター	説明
service_IP_label	削除するサービス IP ラベル。

必須オブジェクト

- クラスター
- サービス IP

例

LabelA という名前のサービス IP ラベルを削除する場合:

```
Clvt delete service_ip LabelA
```

query service_ip

指定されたサービス IP ラベルに関する情報を返します。サービス IP ラベルが指定されていない場合は、すべてのサービス IP ラベルのリストを返します。

構文

```
clvt query service_ip [service_IP_label]
```

パラメーター

```
service_IP_label
```

必須オブジェクト

クラスター

出力

サービス IP ラベルが指定されている場合は、次の情報が返されます。

- サービス IP ラベル名
- IP ラベル・タイプ
- ネットワーク
- ネットワーク・タイプ
- ネットワーク属性
- ノード
- IP アドレス
- ハードウェア・アドレス
- インターフェース名
- グローバル名

例

```
clvt query service_ip
  alias_svc1
  alias_svc2
  maple_base10
  elm_base10
  oak_base10

clvt query service_ip alias_svc1
  INTERFACENAME=""
  NETTYPE="ether"
  TYPE="service"
  ATTR="public"
  IPADDR="192.168.250.1"
  NODE=""
  GLOBALNAME="ignore"
  HADDR=""
  NETWORK="net_ether_01"
  NAME="alias_svc1"
```

アプリケーション・コントローラー・クラス

clvt を使用して、アプリケーション・コントローラーの操作を実行します。

add application

アプリケーション・コントローラーをクラスターに追加します。

これは後でリソース・グループに追加できます。

構文

clvt add application *application_controller_name* STARTPATH=start_script_path STOPPATH=stop_script_path

パラメーター

パラメーター	説明
Application_controller_name	32 文字以下の英数字の文字列。
STARTPATH	アプリケーション・コントローラーの始動スクリプトの絶対パス。
STOPPATH	アプリケーション・コントローラーの停止スクリプトの絶対パス。

必須オブジェクト

クラスター

エラー

- クラスターが存在しない
- アプリケーション・モニターが定義されていない

例

App1 という名前のアプリケーション・コントローラーを追加する場合: clvt add application App1 STARTSCRIPT=/full pathnameX ¥STOPSCRIPT=/full pathnameY

delete application

事前に定義されているアプリケーション・コントローラーを削除します。

構文

`clvt delete application` *`application_controller_name`*

パラメーター

パラメーター	説明
application_controller_name	削除するアプリケーション・コントローラー。

必須オブジェクト

- クラスター
- アプリケーション

エラー

アプリケーション・コントローラーが存在しない

例

MyApp という名前のアプリケーション・コントローラーを削除する場合:

```
clvt delete application MYApp
```

query application

指定されたアプリケーション・コントローラーに関する情報を返します。アプリケーション・コントローラーが指定されていない場合は、すべてのアプリケーション・コントローラーのリストを返します。

構文

```
clvt query application [application_controller_name]
```

パラメーター

```
application_controller_name
```

必須オブジェクト

クラスター

エラー

- クラスターが定義されていない
- アプリケーション・コントローラーが存在しない

出力

アプリケーション・コントローラーが指定されていない場合は、アプリケーション・コントローラーのリストが返されます。アプリケーション・コントローラーが指定されている場合は、次の情報が返されます。

```
Application Controller Name
Start Script
Stop Script
Application Monitors (if configured)
```

例

```
clvt query application
  app_srv_1
  db2
  websphere

clvt query application App1
  STARTSCRIPT="/xxxx"
  ASSOCIATEDMONITORS="AppMon"
  NAME="App1"
  STOPSCRIPT="/yyyy"
```

アプリケーション・モニター・クラス

clvt を使用してアプリケーション・モニターの操作を実行します。

add application_monitor

アプリケーション・モニターをクラスターに追加します。これは後でアプリケーション・コントローラーに追加できます。

構文

```
clvt add application_monitor <application monitor name>
         APPLICATIONMONITORTYPE="Process | Custom"
         APPLICATIONSERVERNAME="appserver1"
         MONITORMODE="longrunning | startup | longrunningandstartup"
         FAILUREACTION="notify | fallover"
         STABILIZATION="1 .. 3600"
         RESTARTCOUNT="0 .. 100"
        [ PROCESSMONITORS="pmon1 dbmon ..." ]
        [ INSTANCECOUNT="1 .. 1024" ]
        [ MONITORMETHOD="/script/to/monitor" ]
        [ MONITORINTERVAL="1 .. 1024" ]
        [ HUNGSIGNAL="1 .. 63" ]
        [ RESTARTINTERVAL="1 .. 3600" ]
```

パラメーター

パラメーター	説明
APPLICATIONMONITORTYPE	モニター・タイプ。オプションは、「Process」および「Custom」です。
MONITORMODE	モニター・モード。オプションは、「longrunning」、「startup」、または「both」です。
APPLICATIONSERVERNAME	モニター対象のアプリケーション・コントローラー名
PROCESSMONITORS	モニター対象のプロセス名
INSTANCECOUNT	モニター対象のプロセス数
MONITORMETHOD	カスタム・アプリケーション・モニター・スクリプト
MONITORINTERVAL	モニター・メソッドは、この間隔 (秒) で定期的に実行されます。
HUNGSIGNAL	モニター間隔の秒数以内に戻らない場合にモニター・メソッドを停止するために送信されるシグナル。デフォルトは SIGKILL(9) です。
STABILIZATION	モニターが安定するまで待機する時間
RESTARTCOUNT	アプリケーション・コントローラーを再始動する回数
RESTARTINTERVAL	RESTARTCOUNT を再設定する前にアプリケーションを安定して保持しなければならない時間
FAILUREACTION	失敗の検出時に実行するアクション。オプションは、「notify」または「fallover」です。

必須オブジェクト

クラスター

エラー

- クラスターが定義されていない
- アプリケーション・コントローラーが定義されていない

例

pmon1 という名前のプロセス・モニターを追加する場合:

```
clvt add application monitor pmon1
APPLICATIONSERVERNAME="appserver1"
 MONITORMODE="longrunning"
  FAILUREACTION="notify"
  PROCESSMONITORS="pmon1."
PROCESSOWNER="<username>"
 INSTANCECOUNT="5"
```

```
  STABILIZATION="50"
  RESTARTCOUNT="5"
RESTARTINTERVAL="50"
NOTIFYMETHOD="</script/name>"
```

cmon1 という名前のカスタム・モニターを追加する場合:

```
clvt add application monitor cmon1
APPLICATIONSERVERNAME="appserver1"
  MONITORMODE="longrunningandstartup"
  MONITORMETHOD="/script/to/monitor"
  MONITORINTERVAL="1024"
  HUNGSIGNAL="60"
  STABILIZATION="3600"
  RESTARTCOUNT="100"
  RESTARTINTERVAL="2400"
  FAILUREACTION="fallover"
```

delete application_monitor

事前に定義されているアプリケーション・モニターを削除します。

構文

```
clvt delete application_monitor application_monitor_name
```

パラメーター

パラメーター	説明
application_monitor_name	削除するアプリケーション・モニター。

必須オブジェクト

- クラスター
- アプリケーション・モニター

エラー

アプリケーション・モニターが存在しない

例

MyMonitor という名前のアプリケーション・モニターを削除する場合:

```
clvt delete application monitor MyMonitor
```

query application_monitor

指定されたアプリケーション・モニターに関する情報を返します。アプリケーション・モニターが指定されていない場合は、すべてのアプリケーション・モニターのリストを返します。

構文

```
clvt query application_monitor [application_monitor_name]
```

パラメーター

パラメーター	説明
app_monitor_name	削除するアプリケーション・モニター。

必須オブジェクト

- クラスター

エラー

- アプリケーション・モニターが存在しない
- クラスターが定義されていない

例

```
clvt query application_monitor
app_srv_1_mon
db2_mon
websphere_mon

clvt query application_monitor App1Mon1
MONITORINTERVAL="30"
CLEANUPMETHOD="/xxxx"
RESTARTCOUNT="3"
HUNGSIGNAL="9"
MONITORMODE="longrunning"
APPLICATIONSERVERNAME="App1"
RESTARTMETHOD="/yyyy"
FAILUREACTION="notify"
MONITORMETHOD="/xxxx"
APPLICATIONMONITORTYPE="user"
NAME="App1Mon1"
RESTARTINTERVAL="132"
STABILIZATION="10"  s
```

リソース・グループの一時的な依存関係クラス

clvt を使用して、リソース・グループの一時的な依存関係の構成を変更または照会します。

modify temporal_dependency

親と子のリソース・グループの依存関係を変更します。

構文

```
clvt modify temporal_dependency parent_resource_group
[RESOURCEGROUPCHILD=child_resource_groups]
```

パラメーター

パラメーター	説明
parent_resource_group	親リソース・グループ。
RESOURCEGROUPCHILDREN	子リソース・グループのスペース区切りリスト。何も指定されていない場合は、指定された親リソース・グループに対する子リソース・グループのすべての関連付けが除去されます。新しいリストを指定すると、現在のリストが置き換えられます。

必須オブジェクト

- クラスター
- リソース・グループ

エラー

- クラスターが定義されていない
- 親リソース・グループが存在しない
- 子リソース・グループが存在しない

例

子リソース・グループ RG2c2 を RG2 に追加して、以前は 1 つの子 (RG2c1) しかなかった一時的な依存関係を変更する場合:

```
clvt modify temporal_dependency RG2 RESOURCEGROUPCHILDREN="RG2c1 RG2c2"
```

query temporal dependency

指定されたリソース・グループの親と子の依存関係をリストします。

構文

```
clvt query temporal_dependency resource group MODE="children | parents"
```

パラメーター

パラメーター	説明
resource group	表示するリソース・グループ。
MODE	出力の表示方法: 親 (parents) または子 (children) のどちらによる表示。

必須オブジェクト

- クラスター
- リソース・グループ

出力

親と子の依存関係のリスト。

例

```
clvt query temporal_ dependency RG2c1 MODE=parents
  RG2

clvt query temporal_dependency RG2 MODE=children
  RG2c1
  RG2c2
```

リソース・グループの場所の依存関係クラス

clvt を使用して、リソース・グループの場所の依存関係の構成を変更または照会します。

modify location_dependency

リソース・グループ間の場所の依存関係を変更します。サポートされている場所の依存関係タイプは SAME_NODE のみです。

構文

```
clvt modify location_dependency SAME_NODE
RESOURCEGROUPLIST=resource_group_list
```

パラメーター

パラメーター	説明
SAME_NODE	場所の依存関係のタイプ。
RESOURCEGROUPLIST	リソース・グループのスペース区切りリスト。

必須オブジェクト

- クラスター
- リソース・グループ

エラー

- クラスターが定義されていない
- リソース・グループが存在しない

例

RG1 と RG2 が同じノードに配置されるように場所の依存関係をセットアップする場合:

```
clvt modify location_dependency SAME_NODE¥ RESOURCEGROUPLIST="RG1 RG2"
```

query location_dependency

常に同じノードに配置されるように構成された一連のリソース・グループのリストを返します。

構文

```
clvt query location_dependency [SAME_NODE]
```

パラメーター

パラメーター	説明
SAME_NODE	表示する場所の依存関係のタイプ。

必須オブジェクト

クラスター

例

ここでは、was_rg_1 と ihs_rg_1 が同じノードに配置され、was_rg_2 と ihs_rg_2 が同じノードに配置されるように定義されています。

```
clvt query location dependency SAME_NODE
  was_rg_1 ihs_rg_1
  was_rg_2 ihs_rg_2
```

ファイル・コレクション・クラス

clvt を使用して以下のファイル・コレクションの操作を実行します。

add file_collection

ファイル・コレクションをクラスターに追加します。

構文

```
clvt add file_collection file_collection_name
[ISPROPOGATEFILESDURINGSYNC="true | false"]
[ISPROPOGATEAUTOWHENDETECTED="true | fslse"] [FILES= "/path/to/file1
/path/to/file2..."]
```

パラメーター

パラメーター	説明
file_collection_name	32 文字以下の英数字の文字列。
ISPROPOGATEFILESDURINGSYNC	「true」または「false」。同期が発生したときに、その発生したクラスターからファイルを伝搬します。デフォルトは false です。
ISPROPOGATEAUTOWHENDETECTED	「true」または「false」。変更が行われたときにファイルを伝搬します。デフォルトは false です。
FILES	指定されたファイル・コレクションにより管理されるファイルの絶対パス名をスペースで区切ったリスト。

必須オブジェクト

クラスター

エラー

クラスターが定義されていない

例

同期中に検出されたときに自動的に伝搬される xxxx ファイルと yyyy ファイルを使用して、FC1 という名前のファイル・コレクションを追加する場合:

```
clvt add file_collection FC1 FILES="/xxxx /yyyy"¥
ISPROPOGATEDFILEDURINGSYNC=true ¥
ISPROPOGATEAUTOWHENDETECTED=true
```

delete file_collection

指定されたファイル・コレクションを削除します。

構文

```
clvt delete file_collection file_collection_name
```

パラメーター

パラメーター	説明
file_collection_name	削除するファイル・コレクション。

必須オブジェクト

- クラスター
- ファイル・コレクション

エラー

- クラスターが定義されていない
- ファイル・コレクションが存在しない

例

FC1 という名前のファイル・コレクションを削除する場合:

```
clvt delete file collection FC1
```

query file_collection

指定されたファイル・コレクションに関する情報を返します。ファイル・コレクションが指定されていない場合は、すべてのファイル・コレクションのリストを返します。

構文

```
clvt query file_collection [file_collection_name]
```

パラメーター

パラメーター	説明
file_collection_name	照会するファイル・コレクションの名前。

必須オブジェクト

クラスター

エラー

- クラスターが定義されていない
- ファイル・コレクションが存在しない

出力

ファイル・コレクションが指定されていない場合は、ファイル・コレクションのリストが出力されます。ファイル・コレクションが指定されている場合は、次の情報が返されます: ファイル・コレクション名、ファイル・コレクションの説明、同期中に伝搬するかどうか、変更されたときに伝搬するかどうか、ファイル (各ファイルはスペースで区切られます)

例

```
clvt query file_collection FC1
  ISPROPOGATEDFILEDURINGSYNC="false"
  FILES="/xxxx /yyyy"
  ISPROPOGATEAUTOWHENDETECTED="false"
  DESCRIPTION="File_Collection_FC1"
  NAME="FC1"
```

Smart Assist のサンプル・プログラム

以下のトピックには、汎用アプリケーション Smart Assist (GASA) に基づいた Smart Assist のサンプル・プログラムを記載します。

概説

汎用アプリケーション Smart Assist (GASA) は、PowerHA SystemMirror に付属したプリインストール済みの Smart Assist です。これは、まだターゲットの Smart Assist がない、始動および停止スクリプトを使用して簡単に管理できるアプリケーションを構成するためのものです。

汎用の Smart Assist は、ユーザーから最小限の情報を要求し、ターゲット・アプリケーションをサポートするために必要な PowerHA SystemMirror リソース・グループとアプリケーション・コントローラーを構成します。 GASA Smart Assist には 1 つの Smart Assist コンポーネントしかありません。より複雑な Smart Assist には複数のコンポーネントがあります。1 つのコンポーネントを含む Smart Assist を作成するための概念とタスクは、複数のコンポーネントを含む Smart Assist を作成する際に拡張して適用することができます。

サンプル・プログラムの機能

Smart Assist の追加画面では、ユーザーが選択した一連の参加ノードにリソース・グループを関連付けます。リソース・グループ内には、Smart Assist を使用して構成されたアプリケーション・コントローラーがあります。以降のセクションでは、汎用アプリケーション Smart Assist について、新しい Smart Assist の作成に必要なサブタスクまで詳しく説明します。

サンプル・プログラムのインストール

General Application Smart Assist は、**cluster.es.assist.common** ファイルセットを使用してインストールされます。複数のファイルがローカル・ファイルシステムにコピーされます。

これらのファイルは次のとおりです。

```
/usr/es/sbin/cluster/sa
                    /gasa
                           ./sbin
                                  add
                                  discovery
                                  modify
                                  smit.utils
```

スクリプト名	説明
add	GASA アプリケーションの新しいインスタンスを PowerHA SystemMirror クラスター構成に追加します
discovery	ローカル・クラスター・ノードでディスカバリーを実行し、Smart Assist コンポーネントを構成できるかどうか判別します。通常、スクリプトは次のように質問します: アプリケーションはローカル・ノードにインストールされていますか?
modify	既存の PowerHA SystemMirror アプリケーション・インスタンスを変更します
smit_util	cmd_to_list、cmd_to_discover、cmd_to_classify の操作を実行するための SMIT ユーティリティー

上記の各スクリプトは ksh93 に書かれています。GASA Smart Assist に対するすべてのソース・コードがアクセス可能です。すべてのスクリプトを同じディレクトリー階層でパッケージ化することをお勧めします。

```
/usr/es/sbin/cluster/sa/<Smart Assist name>/sbin/
```

この階層に従わなくても機能的な制限はありません。これは単に推奨される方法です。

ファイル権限

PowerHA SystemMirror では root ユーザーが PowerHA SystemMirror を構成しなければならないため、Smart Assist のすべてのスクリプトおよびバイナリーを実行するために root 権限が必要になります。通常、これらの権限は 500、所有者の root、およびグループ・システムに設定されます。この要件は、特に **clvt** API コマンドに当てはまります。すべての **clvt** API コマンドが ODM を操作するため、root 権限が必要になります。

Smart Assist の登録

ファイルがローカル・ノードにインストールされると、Smart Assist インストール処理は **claddsa** スクリプトを呼び出し、アプリケーション・フレームワークを使用して Smart Assist コンポーネントのさまざまな属性を登録します。GASA の場合、これは以下のコマンドを使用して実行されます。

```
/usr/es/sbin/cluster/sa/sbin/claddsa -a zz_Other -c GASA ¥
SMARTASSIST_VERSION="1.0" ¥
  COMPONENT_ID="GASA" ¥
  SMARTASSIST_ID="zzOther" ¥
  DISCOVERY_COMMAND="/usr/es/sbin/cluster/sa/gasa/sbin/discovery" ¥
  SMIT_ADD="clsa_gasa_add" ¥
  SMIT_ADD_TYPE="d" ¥
  SMIT_MODIFY="clsa_gasa_modify" ¥
  SMIT_MODIFY_TYPE="d"
```

GASA には 1 つの Smart Assist コンポーネントしかないことに注意してください。追加のコンポーネントが必要な場合は、各コンポーネントごとに上記のコマンドに対して複数の呼び出しが必要になります。

claddsa コマンドの呼び出しにより、一連の HACMPsa ODM 項目が生成されます。これらの項目は、次のコマンドを使用して表示できます。

```
odmget -q "sa_id=zzOther and component_id=GASA" HACMPsa
```

cluster.es.assist.common ファイルセットがインストールされているすべての PowerHA SystemMirror クラスター・ノードに上記の ODM 項目がインストールされます。

サンプル・プログラムのアンインストール

claddsa コマンドで追加された Smart Assist コンポーネント項目は、アンインストール時に除去する必要があります。これらの項目を除去するには、**clrmsa** コマンドを呼び出します。

GASA Smart Assist は、次のコマンドを呼び出して、HACMPsa からその項目を除去します。

```
/usr/es/sbin/cluster/sa/sbin/clrmsa -a zzOther
```

claddsa コマンドとは異なり、**clrmsa** コマンドは、Smart Assist 全体のすべての項目を除去したり (上記のとおり)、**-c** <*component id* > フラグを指定して Smart Assist の特定のコンポーネントのすべての項目を除去したりできます。

項目が除去されると、アンインストールされた Smart Assist とともに構成されていた既存のアプリケーション・インスタンスのアプリケーションへの参照が除去されます。ユーザーは Smart Assist UI 内からそのアプリケーションを変更することはできませんが、構成されているリソース・グループと関連リソースを変更または削除することはできます。

Smart Assist コンポーネントをディスカバーするコマンド

DISCOVERY_COMMAND は、コンポーネントごと、Smart Assist ごとに再呼び出しします。GASA Smart Assist の場合は、1 つのコンポーネントしかないため、実行するコマンドは 1 つのみです。

このコマンドは、ユーザーが次の SMIT メニュー・オプションを選択したときに実行されます。

PowerHA SystemMirror 構成へのアプリケーションの追加 (Add an Application to the PowerHA SystemMirror Configuration)

ユーザーがこの SMIT メニューを選択すると、フレームワークはすべての Smart Assist コンポーネントの DISCOVERY_COMMAND を、そのディスカバリー・コマンドが存在するクラスターで使用可能なす

べてのノード上で実行します。ディスカバーするためのコマンドにより、ターゲット・アプリケーションがインストールされているか、また Smart Assist での使用に適しているかどうかが判別されます。GASA Smart Assist は非常に一般的です。GASA Smart Assist では、すべての PowerHA SystemMirror クラスター・ノードがその Smart Assist で許可されるため、検証は実行されません。

ディスカバリー・コマンドの出力は、次の形式の情報を含む 1 行です。

```
Smart Assist Name in I8N format :
Smart Assist Identifier :
Component Name in I8N form :
Component Identifier :
 0 | 1
```

末尾の 0 または 1 の値により、Smart Assist をローカル・ノードで使用できるかどうかが判別されます。ディスカバリー・コマンドが 0 を出力した場合は、Smart Assist コンポーネントをローカル・ノードで使用できません。1 の値は、Smart Assist コンポーネントをローカル・ノードで使用できることを意味します。GASA Smart Assist の場合は、次のような行が出力されます。

```
Other Applications:zzOther:General Application Smart Assist:GASA:1
```

最初と 3 番目の列の値は、dspmsg とメッセージ・カタログを使用して国際化されています。

アプリケーション・インスタンスの追加機能

Smart Assist には、ユーザーがアプリケーションの新しいインスタンスを PowerHA SystemMirror 構成に追加できる 2 つの部分 (SMIT コンポーネントとスクリプト) があります。SMIT コンポーネントにはダイアログと smit フィールドがあり、スクリプトは追加操作を実行します。

これらの両方は、単一の Smart Assist コンポーネント上で作動します。これらの説明については、以降のセクションに示します。

SMIT の追加画面の作成

GASA Smart Assist の SMIT の追加画面の ID は、**claddsa** への呼び出しに定義されます。

```
SMIT_ADD="clsa_gasa_add"
```

および

```
SMIT_ADD_TYPE="d"
```

これらの項目は SMIT sm_cmd_hdr エレメントを指します。この子は、SMIT ダイアログ・フィールドを形成する sm_cmd_opt 項目です。これらのフィールドでは、ユーザーがアプリケーションに関連した情報 (名前、使用するサービス IP ラベル、Smart Assistant に必要な始動および停止スクリプトなど) を入力します。ダイアログを構成するために使用される項目は、『SMIT 汎用アプリケーション Smart Assist の追加スタンザ』セクションに示されています。

アプリケーション・フレームワークは SMIT コマンド・ヘッダー項目「clsa_gasa_add」を呼び出します。これにより、次の汎用アプリケーション Smart Assist の SMIT 画面が表示されます。

General Application Smart Assist - Add Screen

```
              General Application Smart Assist

 Type or select values in entry fields.


 Press Enter AFTER making all desired changes.

                                    [Entry Fields]
* Application Name                  []
* Primary Node                      [node name displays here] +
* Takeover Nodes                    [node names display here] +
* Application Controller Start Script  []
* Application Controller Stop Script   []
* Service IP Label                  [] +
```

sm_cmd_hdr へのすべての Smart Assist 遷移ではありませんが、場合によってはダイアログへの入力前に選択リストを表示するために sm_name_hdr が必要になります。このような場合、SMIT_ADD_TYPE は名前セレクターの「n」と同等となり、コンポーネントの SMIT_ADD フィールドは sm_name_hdr の SMIT ID を指します。

追加スクリプトの作成

上記に示されている各フィールド項目は、追加スクリプト: GASA Smart Assist 追加スクリプト: /usr/es/sbin/cluster/sa/gasa/add に渡される引数に対応しています:

フィールド名	引数	ディスカバリー・フィールド名 **(disc_name)**
アプリケーション名	-a name	N/A
1 次ノード (Primary Node)	-P node	Primary
テークオーバー・ノード (Takeover Nodes)	-T nodes	テークオーバー
始動スクリプト (Start Script)	-R script	N/A
停止スクリプト (Stop Script)	-O script	N/A
サービス IP ラベル	-S ip_label	N/A
	-C component_id	N/A
	-s smartassist_id	N/A

GASA 追加画面のフィールド名と引数

GASA 追加スクリプトは、開発者の Smart Assist で再使用できます。呼び出し元が Smart Assist ID とコンポーネント ID を指定するということに注意してください。Smart Assist ID -s とコンポーネント ID -C を変更することで、開発者の Smart Assist コンポーネントの追加スクリプトの基礎として既存の GASA 追加スクリプトを使用することができます。追加スクリプトは、以下の操作シーケンスを実行します。

1. **claddsaapp** API コマンドを使用して、重複した項目がないかユーザーが入力したアプリケーション名を検査します。
2. アプリケーション・コントローラーが既に存在しているかどうか判別します:

   ```
   clvt query application
   ```
3. リソース・グループが既に存在しているかどうか判別します:

   ```
   clvt query resource_group
   ```
4. **clsapre** を呼び出します。
5. アプリケーション・コントローラーを PowerHA SystemMirror 構成に追加します。

```
clvt add application $SERVERNAME ¥
   STARTSCRIPT=$STARTSCRIPT ¥
   STOPSCRIPT=$STOPSCRIPT
```

6. リソース・グループを PowerHA SystemMirror 構成に追加します。

```
clvt add resource_group "$RG_NAME" ¥
  PRIMARYNODES="$PRIMARY $TAKEOVER" ¥
  STARTUP="OHN" ¥
  FALLBACK="NFB" ¥
  FALLOVER="FNPN"
```

7. サービス IP ラベルを作成します (まだ定義されていない場合)。

```
clvt add service_ip $SERVICE_LABEL
```

8. 参加しているノード (1 次およびテークオーバー) 間でサービス IP ラベルと使用可能な任意の (共有) ボリューム・グループを関連付けます。

```
clvt modify resource_group "$RG_NAME" ¥
  APPLICATIONS="$NAME" ¥
  SERVICE_LABEL="$IP" ¥
  VOLUME_GROUP="$VGS" ¥
  VG_AUTO_IMPORT="true" ¥
  FORCED_VARYON="false" ¥
  FILESYSTEM=""
```

9. Smart Assist ID とコンポーネント ID を使用して、アプリケーション名のリソース・グループを登録します。これにより、ユーザーが特定のアプリケーション・インスタンスを選択して変更するときに、それに関連付けられているアプリケーションのタイプをフレームワークが確実に認識します。

```
claddsaapp -a $APPLICATION_NAME ¥
APPLICATION_NAME="$APPLICATION_NAME"
RESOURCE_GROUP="$RG_NAME" ¥
SMARTASSIST_ID="$SMARTASSIST_ID" ¥
COMPONENT_ID="$COMPONENT_ID"
```

   claddsaapp コマンドには、アプリケーション・フレームワークとは関連しない、その他の情報 (変数、およびアプリケーション・インスタンス固有の値 (例えば、アプリケーションのパスなど)、その他) が含まれる場合があります。

10. クラスター検証を実行する **clsapost -v** を呼び出します。

上記の手順がすべて完了すると、アプリケーション・インスタンスが Smart Assist コンポーネントに関連付けられ (登録され)、ユーザーはアプリケーションとそれに関連付けられているすべてのリソース全体で変更操作または削除操作を実行できます。

登録スクリプト

zzOther (GASA) Smart Assist は、GASA コンポーネントに REGISTRATION_COMMAND を使用しません。これを使用した場合、ユーザーが Smart Assist および **claddsaapp** という追加スクリプトを追加して、アプリケーション・インスタンス・プロパティーとアプリケーション名を関連付けたとき、HACMPsa ODM スタンザに指定されたとおりに登録コマンドが呼び出されます。

関連概念:

61 ページの『SMIT 汎用アプリケーション Smart Assist の追加スタンザ』
このセクションでは、GASA Smart Assist のコメント入り SMIT 追加スタンザを記載します。

アプリケーション・インスタンスの変更機能

変更機能は追加機能と類似しています。ユーザーはアプリケーション・インスタンスの Smart Assist ID とコンポーネント ID に基づいて、変更する特定のアプリケーション・インスタンスを選択します。

SMIT の変更画面の作成

GASA Smart Assist の SMIT の変更画面の ID は、**claddsa** への呼び出しに定義されます。

```
SMIT_MODIFY="clsa_gasa_modify"
```

および

```
SMIT_MODIFY_TYPE="d"
```

追加画面の方法と同様に、アプリケーション・フレームワークによって HACMPsa ODM 内の項目に基づいて clsa_gasa_modify ダイアログ、または名前セレクターが呼び出されます。

変更画面について異なるのは、フレームワークによってアプリケーションの Smart Assist ID sa_id と component_id と一緒に、クックド・フィールド名として application_id もダイアログまたは名前セレクターに渡されることです。

アプリケーション・フレームワークは SMIT コマンド・ヘッダー項目「clsa_gasa_modify」を呼び出します。これにより、「アプリケーションを高可用性アプリケーションにする **(Smart Assist** の使用**) (Make Applications Highly Available (Use Smart Assists))**」の SMIT 画面が表示されます。

General Application Smart Assist - Selector Screen

```
          Make Applications Highly Available (Use Smart Assists)

Move cursor to desired item and press Enter.

 Add an Application to the PowerHA SystemMirror Configuration
 Change/Show an Application's PowerHA SystemMirror Configuration
 Remove an Application from the PowerHA SystemMirror Configuration
 Manage Your Applications
 Change/Show the Resources Associated with Your Application
 Test Your Application for Availability

+"""""""""""""""""""""""""""""""""""-+
| Select an Application from the List of Discovered Applications Below
|
| Move cursor to desired item and press Enter.
|
| example_app
+"""""""""""""""""""""""""""""""""""""""""""""""""-+
```

以下のクックド・フィールド名がダイアログに渡されます。

項目	説明
application_id	=example_app
sa_id	=zzOther
component_id	=GASA

ユーザーの SMIT セッションは clsa_gasa_modify ダイアログに移行します。

General Application Smart Assist Modify Dialog

```
     General Application Smart Assist
Type or select values in entry fields.

Press Enter AFTER making all desired changes.

[Entry Fields]
* Application Name             example_app

* Primary Node                 [nodeA] +

* Takeover Nodes               [nodeB nodeC] +

* Application Controller Start     [/example_app/start]
Script

* Application Controller Stop      [/example_app/stop]
Script

* Service IP Label             [service1] +
```

通常 Smart Assist の変更画面の表示は追加画面と同じですが、一部のフィールドが変更不可である点が異なります。上記の図では、アプリケーション名が変更できないことに注意してください。ただし、ユーザーは SMIT 内でアプリケーションの構成の他の部分はすべて変更できます。ODM スタンザについては、『SMIT 汎用アプリケーション Smart Assist の変更スタンザ』を参照してください。

ユーザーが変更画面に入る際、フィールドは既に入力されていることに注意してください。これは sm_cmd_hdr ODM スタンザの cmd_to_discover メソッドを使用して実行されます。GASA Smart Assist の場合は、smit_util スクリプトが関連リソース・グループから参加ノードを収集し、clvt API を使用してサービス IP ラベルとアプリケーション始動/停止スクリプトを抽出します。同様の方法がすべての Smart Assist に対して実行されます。

変更スクリプトの作成

GASA Smart Assist の場合は、追加フェーズで構成された必要なリソースがすべて除去され、再度、追加スクリプトが呼び出されます。より複雑な Smart Assist では、変更スクリプトは、ユーザーが変更したアプリケーションのプロパティーを変更するだけです。 GASA は最初に **clsapre** スクリプトを呼び出し、次に **clvt** API を使用してクラスター構成から PowerHA SystemMirror コンポーネントを除去します。その後、追加スクリプトが呼び出され、アプリケーション・リソースが PowerHA SystemMirror 構成に追加されます。再度、追加スクリプトの終わりに **clsapost -v** コマンドが呼び出されて検証が実行されます。変更スクリプトがリソースの変更のみを行う場合は、**clsapost -v** が直接呼び出されます。

関連概念:

64 ページの『SMIT 汎用アプリケーション Smart Assist の変更スタンザ』
このセクションでは、GASA Smart Assist のコメント入り SMIT 変更スタンザを記載します。

アプリケーション・インスタンスの削除

クラスター構成からアプリケーション・インスタンスを除去するために必要な操作は、Smart Assist フレームワーク内に完備されています。単にインスタンスを除去する場合は、追加のコードは必要ありません。

登録解除スクリプト

GASA は DEREGISTRATION_COMMAND を使用しますが、開発者は必要に応じて **claddsa** 呼び出しに項目を追加するだけです。ユーザーがクラスター構成から除去するアプリケーション・インスタンスを選択すると、実際に PowerHA SystemMirror リソースを除去する前に DEREGISTRATION_COMMAND スクリプトが実行されます。これにより、Smart Assist の開発者は PowerHA SystemMirror のアプリケー

ションの制御を除去する前に、アプリケーションの構成設定を復元したり、その他のアプリケーションを復元したりできます。

SMIT 汎用アプリケーション Smart Assist の追加スタンザ

このセクションでは、GASA Smart Assist のコメント入り SMIT 追加スタンザを記載します。

```
# Cooked Field: nodes
#
# Note that the text "nodes" in the cmd_to_discover_postfix is part
# of #the cooked fields from the Smart Assist screens that precede this
# dialog. Users will #select a Smart Assist to instantiate, once
# selected a list of nodes where that smart #assist can be
# used will be provided to the SMIT screens that follow. The "nodes"
# cooked field contains a comma ',' delimited list of node names
sm_cmd_hdr:
  id = "clsa_gasa_add"
  option_id = "clsa_gasa_add_opts"
  has_name_select = ""
  name = "General Application Smart Assist"
  name_msg_file = "cluster.cat"
  name_msg_set = 51
  name_msg_id = 17
  cmd_to_exec = "/usr/es/sbin/cluster/sa/gasa/sbin/add"
  ask = ""
  exec_mode = ""
  ghost = ""
  cmd_to_discover = "/usr/es/sbin/cluster/sa/gasa/sbin/smit_util
add_discover"
  cmd_to_discover_postfix = "nodes"
  name_size = 0
  value_size = 0
  help_msg_id = "42,10"
  help_msg_loc = "cluster_hlp.cat"
  help_msg_base = ""
  help_msg_book = ""

# Application Name
#
# This is the name that PowerHA SystemMirror will use to uniquely identify the
# collection of PowerHA SystemMirror resource groups, application controller, and other
# related resources
# Users can modify / delete an existing application given its name
# All Smart Assists must contain an application name
#
sm_cmd_opt:
  id_seq_num = "10"
  id = "clsa_gasa_add_opts"
  disc_field_name = ""
  name = "Application Name"
  name_msg_file = "cluster.cat"
  name_msg_set = 43
  name_msg_id = 32
  op_type = ""
  entry_type = "t"
  entry_size = 24
  required = "+"
  prefix = "-a"
  cmd_to_list_mode = ""
  cmd_to_list = ""
  cmd_to_list_postfix = ""
  multi_select = ""
  value_index = 0
  disp_values = ""
  values_msg_file = ""
  values_msg_set = 0
```

```
  values_msg_id = 0
  aix_values = ""
  help_msg_id = "30,3"
  help_msg_loc = "cluster_hlp.cat"
  help_msg_base = ""
  help_msg_book = ""

# Primary Node:
#
# For the GASA Smart Assist there exists a separation of primary and
# takeover nodes.
#
# The primary node is where the application will be brought online
# initially and will be the first node in the participating nodes of the
# constructed PowerHA SystemMirror resource group.
#
sm_cmd_opt:
  id_seq_num = "20"
  id = "clsa_gasa_add_opts"
  disc_field_name = "primary"
  name = "Primary Node"
  name_msg_file = "cluster.cat"
  name_msg_set = 51
  name_msg_id = 15
  op_type = "l"
  entry_type = "t"
  entry_size = 33
  required = "+"
  prefix = "-P"
  cmd_to_list_mode = ""
  cmd_to_list = "/usr/es/sbin/cluster/sa/gasa/sbin/smit_util list"
  cmd_to_list_postfix = "nodes"
  multi_select = ""
  value_index = 0
  disp_values = ""
  values_msg_file = ""
  values_msg_set = 0
  values_msg_id = 0
  aix_values = ""
  help_msg_id = "42,8"
  help_msg_loc = "cluster_hlp.cat"
  help_msg_base = ""
  help_msg_book = ""

# Takeover Nodes:
#
# List of nodes that participate in the constructed resource group
#
# Note that the SMIT cmd_to_list command uses our internal smit_util
# command, which internally calls "clvt query node"
#
sm_cmd_opt:
  id_seq_num = "30"
  id = "clsa_gasa_add_opts"
  disc_field_name = "takeover"
  name = "Takeover Nodes"
  name_msg_file = "cluster.cat"
  name_msg_set = 51
  name_msg_id = 16
  op_type = "l"
  entry_type = "t"
  entry_size = 0
  required = "+"
  prefix = "-T"
  cmd_to_list_mode = ""
  cmd_to_list = "/usr/es/sbin/cluster/sa/gasa/sbin/smit_util list"
  cmd_to_list_postfix = "nodes"
```

```
  multi_select = ","
  value_index = 0
  disp_values = ""
  values_msg_file = ""
  values_msg_set = 0
  values_msg_id = 0
  aix_values = ""
  help_msg_id = "42,9"
  help_msg_loc = "cluster_hlp.cat"
  help_msg_base = ""
  help_msg_book = ""

# Application Controller Start Script
#
# Script name used to start the application
#
# No default value
# This property is required
#
sm_cmd_opt:
  id_seq_num = "40"
  id = "clsa_gasa_add_opts"
  disc_field_name = ""
  name = "Application Controller Start Script"
  name_msg_file = "cluster.cat"
  name_msg_set = 43
  name_msg_id = 4
  op_type = ""
  entry_type = "t"
  entry_size = 256
  required = "+"
  prefix = "-R"
  cmd_to_list_mode = ""
  cmd_to_list = ""
  cmd_to_list_postfix = ""
  multi_select = ""
  value_index = 0
  disp_values = ""
  values_msg_file = ""
  values_msg_set = 0
  values_msg_id = 0
  aix_values = ""
  help_msg_id = "30,4"
  help_msg_loc = "cluster_hlp.cat"
  help_msg_base = ""
  help_msg_book = ""

# Application Controller Stop Script
#
# Script name used to start the application
#
# No default value
# This property is required
#
sm_cmd_opt:
  id_seq_num = "50"
  id = "clsa_gasa_add_opts"
  disc_field_name = ""
  name = "Application Controller Stop Script"
  name_msg_file = "cluster.cat"
  name_msg_set = 43
  name_msg_id = 5
  op_type = ""
  entry_type = "t"
  entry_size = 256
  required = "+"
  prefix = "-O"
```

```
  cmd_to_list_mode = ""
  cmd_to_list = ""
  cmd_to_list_postfix = ""
  multi_select = ""
  value_index = 0
  disp_values = ""
  values_msg_file = ""
  values_msg_set = 0
  values_msg_id = 0
  aix_values = ""
  help_msg_id = "30,5"
  help_msg_loc = "cluster_hlp.cat"
  help_msg_base = ""
  help_msg_book = ""

# Service IP label
#
# The user must choose a service IP label to participate in the
# constructed HACM resource group
#
# The command to list cllsserviceips can be used by other Smart Assists
# to provide a list of pre-configured service IP labels, and available
# labels that can be defined to PowerHA SystemMirror.
#
sm_cmd_opt:
  id_seq_num = "60"
  id = "clsa_gasa_add_opts"
  disc_field_name = ""
  name = "Service IP Label"
  name_msg_file = "cluster.cat"
  name_msg_set = 43
  name_msg_id = 6
  op_type = "l"
  entry_type = "t"
  entry_size = 32
  required = "+"
  prefix = "-I"
  cmd_to_list_mode = "1"
  cmd_to_list = "/usr/es/sbin/cluster/sa/sbin/cllsserviceips"
  cmd_to_list_postfix = ""
  multi_select = "n"
  value_index = 0
  disp_values = ""
  values_msg_file = ""
  values_msg_set = 0
  values_msg_id = 0
  aix_values = ""
  help_msg_id = "30,6"
  help_msg_loc = "cluster_hlp.cat"
  help_msg_base = ""
  help_msg_book = ""
```

SMIT 汎用アプリケーション Smart Assist の変更スタンザ

このセクションでは、GASA Smart Assist のコメント入り SMIT 変更スタンザを記載します。

```
#
# Modify SMIT Dialog
#
# The smit_util modify_discover uses the application_id to determine
# what PowerHA SystemMirror components are associated with the application instance and
# produces cmd_to_discover output of the form:
#
# /usr/es/sbin/cluster/sa/gasa/sbin/smit_util modify_discover exampleapp
#
# #application:primary:takeover:start:stop:ip
# exampleapp:nodeA:nodeB
```

```
# nodeC:/exampleapp/start:/exampleapp/stop:service1
#
sm_cmd_hdr:
  id = "clsa_gasa_modify"
  option_id = "clsa_gasa_modify_opts"
  has_name_select = ""
  name = "General Application Smart Assist"
  name_msg_file = "cluster.cat"
  name_msg_set = 51
  name_msg_id = 17
  cmd_to_exec = "/usr/es/sbin/cluster/sa/gasa/sbin/modify"
  ask = ""
  exec_mode = ""
  ghost = ""
  cmd_to_discover = "/usr/es/sbin/cluster/sa/gasa/sbin/smit_util
modify_discover"
  cmd_to_discover_postfix = "application_id"
  name_size = 0
  value_size = 0
  help_msg_id = "42,11"
  help_msg_loc = "cluster_hlp.cat"
  help_msg_base = ""
  help_msg_book = ""

# Application Name
#
# This is the name that PowerHA SystemMirror will use to uniquely identify the
#collection of PowerHA SystemMirror resource groups, application controllers, and other
#related resources.
# Users can modify / delete an existing application given its name
# All Smart Assists must contain an application name
#
# The name cannot be modified in the modify dialog.
#
sm_cmd_opt:
  id_seq_num = "10"
  id = "clsa_gasa_modify_opts"
  disc_field_name = "application"
  name = "Application Name"
  name_msg_file = "cluster.cat"
  name_msg_set = 43
  name_msg_id = 32
  op_type = ""
  entry_type = "n"
  entry_size = 24
  required = "+"
  prefix = "-a"
  cmd_to_list_mode = ""
  cmd_to_list = ""
  cmd_to_list_postfix = ""
  multi_select = ""
  value_index = 0
  disp_values = ""
  values_msg_file = ""
  values_msg_set = 0
  values_msg_id = 0
  aix_values = ""
  help_msg_id = "30,3"
  help_msg_loc = "cluster_hlp.cat"
  help_msg_base = ""
  help_msg_book = ""

# Primary Node:
#
# For the GASA Smart Assist there exists a separation of primary and
# takeover nodes.
#
```

```
# The primary node is where the application will be brought online
# initially and will be the first node in the participating nodes of the
# constructed PowerHA SystemMirror resource group.
#
sm_cmd_opt:
  id_seq_num = "20"
  id = "clsa_gasa_modify_opts"
  disc_field_name = "primary"
  name = "Primary Node"
  name_msg_file = "cluster.cat"
  name_msg_set = 51
  name_msg_id = 15
  op_type = "l"
  entry_type = "t"
  entry_size = 33
  required = "+"
  prefix = "-P"
  cmd_to_list_mode = ""
  cmd_to_list = "cmd_to_list() {¥n¥
           /usr/es/sbin/cluster/utilities/clvt query node¥n¥
           return 0¥n¥
  }¥n¥
  cmd_to_list"
  cmd_to_list_postfix = ""
  multi_select = ""
  value_index = 0
  disp_values = ""
  values_msg_file = ""
  values_msg_set = 0
  values_msg_id = 0
  aix_values = ""
  help_msg_id = "42,8"
  help_msg_loc = "cluster_hlp.cat"
  help_msg_base = ""
  help_msg_book = ""

# Takeover Nodes:
#
# List of nodes that participate in the constructed resource group
#
# Note that the SMIT cmd_to_list command uses our internal smit_util
# command, which internally calls "clvt query node"
#
  sm_cmd_opt:
  id_seq_num = "30"
  id = "clsa_gasa_modify_opts"
  disc_field_name = "takeover"
  name = "Takeover Nodes"
  name_msg_file = "cluster.cat"
  name_msg_set = 51
  name_msg_id = 16
  op_type = "l"
  entry_type = "t"
  entry_size = 0
  required = "+"
  prefix = "-T"
  cmd_to_list_mode = ""
  cmd_to_list = "cmd_to_list() {¥n¥
           /usr/es/sbin/cluster/utilities/clvt query node¥n¥
           return 0¥n¥
  }¥n¥
  cmd_to_list"
  cmd_to_list_postfix = ""
  multi_select = ","
  value_index = 0
  disp_values = ""
  values_msg_file = ""
```

```
  values_msg_set = 0
  values_msg_id = 0
  aix_values = ""
  help_msg_id = "42,9"
  help_msg_loc = "cluster_hlp.cat"
  help_msg_base = ""
  help_msg_book = ""

# Application Controller Start Script
#
# Script name used to start the application
#
# No default value
# This property is required
#
sm_cmd_opt:
  id_seq_num = "40"
  id = "clsa_gasa_modify_opts"
  disc_field_name = "start"
  name = "Application Controller Start Script"
  name_msg_file = "cluster.cat"
  name_msg_set = 43
  name_msg_id = 4
  op_type = ""
  entry_type = "t"
  entry_size = 256
  required = "+"
  prefix = "-R"
  cmd_to_list_mode = ""
  cmd_to_list = ""
  cmd_to_list_postfix = ""
  multi_select = ""
  value_index = 0
  disp_values = ""
  values_msg_file = ""
  values_msg_set = 0
  values_msg_id = 0
  aix_values = ""
  help_msg_id = "30,4"
  help_msg_loc = "cluster_hlp.cat"
  help_msg_base = ""
  help_msg_book = ""

# Application Controller Stop Script
#
# Script name used to start the application
#
# No default value
# This property is required
#
sm_cmd_opt:
  id_seq_num = "50"
  id = "clsa_gasa_modify_opts"
  disc_field_name = "stop"
  name = "Application Controller Stop Script"
  name_msg_file = "cluster.cat"
  name_msg_set = 43
  name_msg_id = 5
  op_type = ""
  entry_type = "t"
  entry_size = 256
  required = "+"
  prefix = "-O"
  cmd_to_list_mode = ""
  cmd_to_list = ""
  cmd_to_list_postfix = ""
  multi_select = ""
```

```
  value_index = 0
  disp_values = ""
  values_msg_file = ""
  values_msg_set = 0
  values_msg_id = 0
  aix_values = ""
  help_msg_id = "30,5"
  help_msg_loc = "cluster_hlp.cat"
  help_msg_base = ""
  help_msg_book = ""
# Service IP label
#
# The user must choose a service IP label to participate in the
# constructed PowerHA SystemMirror resource group
#
# The command to list cllsserviceips can be used by other Smart Assists
# to provide a ist of pre-configured service IP labels, and available
# labels that can be defined to PowerHA SystemMirror.
#
sm_cmd_opt:
  id_seq_num = "60"
  id = "clsa_gasa_modify_opts"
  disc_field_name = "ip"
  name = "Service IP Label"
  name_msg_file = "cluster.cat"
  name_msg_set = 43
  name_msg_id = 6
  op_type = "l"
  entry_type = "t"
  entry_size = 32
  required = "+"
  prefix = "-I"
  cmd_to_list_mode = "1"
  cmd_to_list = "/usr/es/sbin/cluster/sa/sbin/cllsserviceips"
  cmd_to_list_postfix = ""
  multi_select = "n"
  value_index = 0
  disp_values = ""
  values_msg_file = ""
  values_msg_set = 0
  values_msg_id = 0
  aix_values = ""
  help_msg_id = "30,6"
  help_msg_loc = "cluster_hlp.cat"
  help_msg_base = ""
  help_msg_book = ""
```

特記事項

本書は米国 IBM が提供する製品およびサービスについて作成したものです。

本書に記載の製品、サービス、または機能が日本においては提供されていない場合があります。日本で利用可能な製品、サービス、および機能については、日本 IBM の営業担当員にお尋ねください。本書で IBM 製品、プログラム、またはサービスに言及していても、その IBM 製品、プログラム、またはサービスのみが使用可能であることを意味するものではありません。これらに代えて、IBM の知的所有権を侵害することのない、機能的に同等の製品、プログラム、またはサービスを使用することができます。ただし、IBM 以外の製品とプログラムの操作またはサービスの評価および検証は、お客様の責任で行っていただきます。

IBM は、本書に記載されている内容に関して特許権 (特許出願中のものを含む) を保有している場合があります。本書の提供は、お客様にこれらの特許権について実施権を許諾することを意味するものではありません。実施権についてのお問い合わせは、書面にて下記宛先にお送りください。

〒*103-8510*
東京都中央区日本橋箱崎町*19*番*21*号
日本アイ・ビー・エム株式会社
法務・知的財産
知的財産権ライセンス渉外

IBM およびその直接または間接の子会社は、本書を特定物として現存するままの状態で提供し、商品性の保証、特定目的適合性の保証および法律上の瑕疵担保責任を含むすべての明示もしくは黙示の保証責任を負わないものとします。国または地域によっては、法律の強行規定により、保証責任の制限が禁じられる場合、強行規定の制限を受けるものとします。

この情報には、技術的に不適切な記述や誤植を含む場合があります。本書は定期的に見直され、必要な変更は本書の次版に組み込まれます。 IBM は予告なしに、随時、この文書に記載されている製品またはプログラムに対して、改良または変更を行うことがあります。

本書において IBM 以外の Web サイトに言及している場合がありますが、便宜のため記載しただけであり、決してそれらの Web サイトを推奨するものではありません。それらの Web サイトにある資料は、この IBM 製品の資料の一部ではありません。それらの Web サイトは、お客様の責任でご使用ください。

IBM は、お客様が提供するいかなる情報も、お客様に対してなんら義務も負うことのない、自ら適切と信ずる方法で、使用もしくは配布することができるものとします。

本プログラムのライセンス保持者で、(i) 独自に作成したプログラムとその他のプログラム (本プログラムを含む) との間での情報交換、および (ii) 交換された情報の相互利用を可能にすることを目的として、本プログラムに関する情報を必要とする方は、下記に連絡してください。

IBM Director of Licensing
IBM Corporation
North Castle Drive, MD-NC119
Armonk, NY 10504-1785
US

本プログラムに関する上記の情報は、適切な使用条件の下で使用することができますが、有償の場合もあります。

本書で説明されているライセンス・プログラムまたはその他のライセンス資料は、IBM 所定のプログラム契約の契約条項、IBM プログラムのご使用条件、またはそれと同等の条項に基づいて、IBM より提供されます。

記載されている性能データとお客様事例は、例として示す目的でのみ提供されています。実際の結果は特定の構成や稼働条件によって異なります。

IBM 以外の製品に関する情報は、その製品の供給者、出版物、もしくはその他の公に利用可能なソースから入手したものです。 IBM は、それらの製品のテストは行っておりません。したがって、他社製品に関する実行性、互換性、またはその他の要求については確証できません。IBM 以外の製品の性能に関する質問は、それらの製品の供給者にお願いします。

IBM の将来の方向または意向に関する記述については、予告なしに変更または撤回される場合があり、単に目標を示しているものです。

表示されている IBM の価格は IBM が小売り価格として提示しているもので、現行価格であり、通知なしに変更されるものです。卸価格は、異なる場合があります。

本書はプランニング目的としてのみ記述されています。記述内容は製品が使用可能になる前に変更になる場合があります。

本書には、日常の業務処理で用いられるデータや報告書の例が含まれています。より具体性を与えるために、それらの例には、個人、企業、ブランド、あるいは製品などの名前が含まれている場合があります。これらの名称はすべて架空のものであり、名称や住所が類似する企業が実在しているとしても、それは偶然にすぎません。

著作権使用許諾:

本書には、様々なオペレーティング・プラットフォームでのプログラミング手法を例示するサンプル・アプリケーション・プログラムがソース言語で掲載されています。お客様は、サンプル・プログラムが書かれているオペレーティング・プラットフォームのアプリケーション・プログラミング・インターフェースに準拠したアプリケーション・プログラムの開発、使用、販売、配布を目的として、いかなる形式においても、IBM に対価を支払うことなくこれを複製し、改変し、配布することができます。このサンプル・プログラムは、あらゆる条件下における完全なテストを経ていません。従って IBM は、これらのサンプル・プログラムについて信頼性、利便性もしくは機能性があることをほのめかしたり、保証することはできません。これらのサンプル・プログラムは特定物として現存するままの状態で提供されるものであり、いかなる保証も提供されません。 IBM は、お客様の当該サンプル・プログラムの使用から生ずるいかなる損害に対しても一切の責任を負いません。

それぞれの複製物、サンプル・プログラムのいかなる部分、またはすべての派生した創作物には、次のように、著作権表示を入れていただく必要があります。

© (お客様の会社名) (西暦年).

このコードの一部は、IBM Corp. のサンプル・プログラムから取られています。

© Copyright IBM Corp. _年を入れる_.

プライバシー・ポリシーに関する考慮事項

サービス・ソリューションとしてのソフトウェアも含めた IBM ソフトウェア製品 (「ソフトウェア・オファリング」) では、製品の使用に関する情報の収集、エンド・ユーザーの使用感の向上、エンド・ユーザーとの対話またはその他の目的のために、Cookie はじめさまざまなテクノロジーを使用することがあります。多くの場合、ソフトウェア・オファリングにより個人情報が収集されることはありません。 IBM の「ソフトウェア・オファリング」の一部には、個人情報を収集できる機能を持つものがあります。ご使用の「ソフトウェア・オファリング」が、これらの Cookie およびそれに類するテクノロジーを通じてお客様による個人情報の収集を可能にする場合、以下の具体的事項を確認ください。

この「ソフトウェア・オファリング」は、Cookie もしくはその他のテクノロジーを使用して個人情報を収集することはありません。

この「ソフトウェア・オファリング」が Cookie およびさまざまなテクノロジーを使用してエンド・ユーザーから個人を特定できる情報を収集する機能を提供する場合、お客様は、このような情報を収集するにあたって適用される法律、ガイドライン等を遵守する必要があります。これには、エンドユーザーへの通知や同意の要求も含まれますがそれらには限られません。

このような目的での Cookie を含む様々なテクノロジーの使用の詳細については、IBM の『IBM オンラインでのプライバシー・ステートメント』(http://www.ibm.com/privacy/details/jp/ja/) の『クッキー、ウェブ・ビーコン、その他のテクノロジー』および『IBM Software Products and Software-as-a-Service Privacy Statement』(http://www.ibm.com/software/info/product-privacy) を参照してください。

商標

IBM、IBM ロゴおよび ibm.com は、世界の多くの国で登録された International Business Machines Corp. の商標です。 他の製品名およびサービス名等は、それぞれ IBM または各社の商標である場合があります。 現時点での IBM の商標リストについては、http://www.ibm.com/legal/copytrade.shtml をご覧ください。

索引

日本語, 数字, 英字, 特殊文字の順に配列されています。なお, 濁音と半濁音は清音と同等に扱われています。

[ア行]

[カ行]

[サ行]

[タ行]

[ナ行]

[ハ行]

［ヤ行］

［ラ行］

A

C

D

I

M

O

Q

S

Printed in Japan

日本アイ・ビー・エム株式会社
〒103-8510 東京都中央区日本橋箱崎町19-21