Sun Desktop Manager 1.0 インストールガイド

Sun Microsystems, Inc.
4150 Network Circle
Santa Clara, CA 95054
U.S.A.

Part No: 819–6093–10
2006 年 1 月

原典: Sun Desktop Manager 1.0 Installation Guide

Part No: 819-2725

Revision A

060403@14558

目次

はじめに

このマニュアルでは、Sun™ Desktop Manager 1.0 の配備に必要なインストールと構成の手順について説明します。

概要

Sun Desktop Manager は、デスクトップホストの一元構成を提供することを目的としています。組織またはドメイン構造のさまざまな要素に合わせて設定を割り当てることができるため、管理者はユーザーまたはホストのグループを効率よく管理できます。

マニュアルの構成

第 1 章では、Sun Desktop Manager について簡単に説明します。

第 2 章では、サーバー側での Sun Desktop Manager のインストールについて説明します。

第 3 章では、Java Desktop System Configuration Agent のインストールに関する情報を提供します。

第 4 章では、Java Web Console に関するインストール情報を提供します。

付録 A には、構成パラメータ情報が含まれています。

付録 B では、Desktop Manager による OpenLDAP と Active Directory の使用について説明します。

付録 C では、組織のマッピングに関する情報を提供します。

関連情報

- 『Sun Desktop Manager 1.0 管理ガイド』
- 『Sun Desktop Manager 1.0 開発者ガイド』

マニュアル、サポート、およびトレーニング

Sun のサービス	URL	内容
マニュアル	`http://jp.sun.com/documentation/`	PDF 文書および HTML 文書をダウンロードできます。
サポートおよびトレーニング	`http://jp.sun.com/supportraining/`	技術サポート、パッチのダウンロード、および Sun のトレーニングコース情報を提供します。

表記上の規則

このマニュアルでは、次のような字体や記号を特別な意味を持つものとして使用します。

表 P–1 表記上の規則

字体または記号	意味	例
`AaBbCc123`	コマンド名、ファイル名、ディレクトリ名、画面上のコンピュータ出力、コード例を示します。	`.login` ファイルを編集します。 `ls -a` を使用してすべてのファイルを表示します。 `system%`
`AaBbCc123`	ユーザーが入力する文字を、画面上のコンピュータ出力と区別して示します。	`system%` **`su`** `password:`
`AaBbCc123`	変数を示します。実際に使用する特定の名前または値で置き換えます。	ファイルを削除するには、`rm` *`filename`* と入力します。
『 』	参照する書名を示します。	『コードマネージャ・ユーザーズガイド』を参照してください。

表 P–1 表記上の規則　(続き)

字体または記号	意味	例
「」	参照する章、節、ボタンやメニュー名、強調する単語を示します。	第 5 章「衝突の回避」を参照してください。 この操作ができるのは、「スーパーユーザー」だけです。
\	枠で囲まれたコード例で、テキストがページ行幅を超える場合に、継続を示します。	sun% **grep '^#define ** **XV_VERSION_STRING'**

コード例は次のように表示されます。

- C シェル

  machine_name% **command y|n** [*filename*]

- C シェルのスーパーユーザー

  machine_name# **command y|n** [*filename*]

- Bourne シェルおよび Korn シェル

  $ **command y|n** [*filename*]

- Bourne シェルおよび Korn シェルのスーパーユーザー

  # **command y|n** [*filename*]

[ ] は省略可能な項目を示します。上記の例は、*filename* は省略してもよいことを示しています。

| は区切り文字 (セパレータ) です。この文字で分割されている引数のうち 1 つだけを指定します。

キーボードのキー名は英文で、頭文字を大文字で示します (例: Shift キーを押します)。ただし、キーボードによっては Enter キーが Return キーの動作をします。

ダッシュ (-) は 2 つのキーを同時に押すことを示します。たとえば、Ctrl-D は Control キーを押したまま D キーを押すことを意味します。

◆ ◆ ◆ 第 1 章

概要と概念

このマニュアルでは、Sun™ Desktop Manager 1.0 の配備に必要なインストールと構成の手順について説明します。Sun Desktop Manager のより広範な概要については、『Sun Desktop Manager 1.0 管理ガイド』 を参照してください。

Sun Desktop Manager の概要

Sun Desktop Manager は、デスクトップホストの中央構成を提供します。組織またはドメイン構造のさまざまな要素に合わせて設定を割り当てることができるため、管理者はユーザーまたはホストのグループを効率よく管理できます。

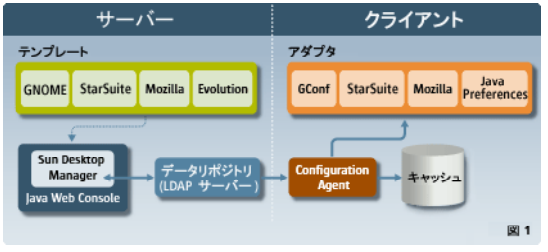

図 1–1 Desktop Manager のアーキテクチャー

Desktop Manager の主なコンポーネントは次のとおりです。

- 構成リポジトリ
- 管理ツール
- Desktop Manager テンプレート
- Configuration Agent
- 構成アダプタ

構成データは、構成リポジトリに一元的に格納されます。構成データは、Web ベースの Desktop Manager グラフィカルユーザーインタフェースとコマンド行インタフェースで構

成される管理ツールを使用して管理 (作成 / 削除 / 変更 / 割り当て / 割り当て解除) されます。テンプレートは、Web ブラウザに構成データを描画するために Web ベースの管理ツールにより使用されます。

Configuration Agent は、ユーザーアプリケーションに代わって、構成リポジトリから構成データを取り出します。エージェントは、中央の構成リポジトリから取り出した情報をキャッシュに書き込みます。

管理ツールはエージェントとは完全に分離されます。つまり、構成リポジトリに対してのみ作用します。

構成アダプタを使用するユーザーアプリケーションは、Configuration Agent を介して構成データを照会します。

製品は、次の構成システムの設定の検索とアプリケーションを直接サポートします。

- GConf. GNOME 構成フレームワーク
- StarSuite Registry
- Mozilla Preferences
- Java Preferences

◆ ◆ ◆ 第 2 章

2 管理アプリケーションのインストール

この章では、Sun Desktop Manager のサーバー側コンポーネントのインストール方法について説明します。

Sun Desktop Manager

Desktop Manager は、Java Web Console 上で動作する Web ベースの管理ツールを提供します。このユーザーインタフェースにより、管理者が組織の階層をナビゲートしてデスクトップアプリケーションのポリシーを指定できます。このポリシーは、組織、役割、ユーザー、ドメイン、ホストなど、階層内の各項目に対して指定できます。Desktop Manager はいくつかの構成テンプレートを使用して、GNOME、Mozilla、StarSuite、Evolution などのさまざまなデスクトップアプリケーションに固有な設定を表示します。

▼ インストール

始める前に Desktop Manager では、Java Web Console バージョン 2.2.5 以上がインストールされている必要があります。システムに有効なバージョンがインストールされていることを確認してください。有効なバージョンがインストールされているかどうかを確認するには、スーパーユーザー (root) になって、次のコマンドを実行します。

```
# smcwebserver status
```

注 – Java Web Console 2.2.4 は Solaris™ 10 オペレーティングシステムに含まれますが、Desktop Manager にはバージョン 2.2.5 以上が必要です。バージョン 2.2.5 のコピーは、server/console ディレクトリ内の Desktop Manager アーカイブにあります。そのディレクトリ内にある `./setup` を実行するとインストールできます。

Java Web Console がシステムにインストールされていない、またはインストールされているバージョンが Desktop Manager に有効でない場合は、第 4 章 の手順を参照して、Java

Web Console を最初にインストールまたは更新してください。そのあとで、この章に戻り、Desktop Manager のインストールを続行してください。

1. **Desktop Manager** の **zip** アーカイブをダウンロードし、一時ディレクトリに内容を取り出します。

   ```
   # unzip SunDesktopMgr-1.0.zip
   ```

2. スーパーユーザー **(root)** になって、次のように設定スクリプトを実行します。

   ```
   # cd SunDesktopMgr-1.0/<platform>/server/manager
   # ./setup
   ```

3. 設定スクリプトの出力でエラーを確認します。

   インストールが正常に実行された場合、設定スクリプトは Java Web Console を自動的に再起動し、Web ブラウザで Desktop Manager にアクセスできるようになります。

▼ 操作

1. 次の **URL** をブラウザに入力します。

   ```
   https://<hostname>.<domainname>:6789
   ```

2. ログイン画面で、既存の **Unix** ユーザーのユーザー名とパスワードを入力します。

   Java Web Console が開きます。

3. コンソールアプリケーションの起動ページで、**Desktop Manager** のリンクをクリックします。

   - コンソールアプリケーションの起動ページをスキップして、**Desktop Manager** に直接アクセスするには、次の **URL** をブラウザに入力します。

     ```
     https://<hostname>.<domainname>:6789/apoc
     ```

▼ Desktop Manager の削除

◗ **Desktop Manager** を **Java Web Console** から削除するには、インストール用に作成した一時ディレクトリに移動して、スーパーユーザー **(root)** になり、次のコマンドを実行します。

```
# cd server/manager
# ./setup -u
```

移行の問題

Desktop Manager は、以前のバージョンの Java Desktop System Configuration Manager (リリース 1.0 と 1.1) と互換性があります。ただし、注意する必要がある相違点がいくつかあります。

以前のバージョンの Configuration Manager では、プロファイルデータはすべて、指定した 1 台の LDAP サーバーに格納されました。この LDAP サーバーは、Configuration Manager インストール手順全体の一部として構成されました。これには、LDAP サーバーに対する認証をカプセル化する LDAP ログインモジュールの構成も含まれていました。

Desktop Manager では、必要な構成手順はすべてウィザードベースとなり、インストール中に何らかの構成を行う必要がなくなりました。Desktop Manager は、複数の構成リポジトリもサポートするようになりました。したがって、いくつもの異なる LDAP サーバーやファイルベースのリポジトリなどに格納されたポリシーデータを管理できます。特定の LDAP ログインモジュールを構成する必要はなくなりました。

異なるバージョン間での LDAP スキーマに変更はありません。以前のバージョンの Configuration Manager 用に LDAP サーバーをすでに構成している場合、Desktop Manager に切り換える際に必要な変更はありません。したがって、クライアント (Java Desktop System Configuration Manager 1.1. Agent) または LDAP 側を更新しないで、Desktop Manager を利用することができます。

注 – Desktop Manager をインストールする前に、まず以前にシステムにインストールした Configuration Manager または Desktop Manager を削除してください。以前のインストールを削除するには、(スーパーユーザーとして) 次のコマンドを実行します。

```
# cd server/manager
# ./setup -u
```

Desktop Manager をインストールしたあとで、既存の LDAP サーバーを指す構成リポジトリを作成できます。

▼ 構成リポジトリの作成

1 次の **URL** をブラウザに入力します。

```
https://<hostname>.<domainname>:6789
```

2 ログイン画面で、既存の **Unix** ユーザーのユーザー名とパスワードを入力します。

Java Web Console が開きます。

3 コンソールアプリケーションの起動ページで、**Sun Desktop Manager 1.0** のリンクをクリックします。

4 「新規作成」ボタンをクリックして構成リポジトリウィザードを起動します。
ウィザードに従って、LDAP ベースの構成リポジトリの構成に必要な手順を実行します。

注意 – ウィザードにより、既存のポリシーデータを新しい 2.0 形式に自動で移行できます。この移行は省略可能で、主に新しい Sun Desktop Manager 1.0 エージェントの性能を向上するために使用できます。使用する環境において Java Desktop System Configuration Manager 1.1 エージェントをサポートする必要があるかぎり、この移行を実行しないでください。

Desktop Manager のトラブルシューティング

インストールできない

症状: Java Web Console のインストールの最後に、登録されたアプリケーションがないためアプリケーションを起動できないことを示すメッセージが表示されます。

考えられる原因: Desktop Manager を含め、アプリケーションがインストールされていません。

解決方法: Desktop Manager をインストールしてから、Java Web Console を起動してください。

接続が拒否される

症状: `http://`*<hostname>*`.`*<domainname>*`:6789` などの適切な URL を開こうとしたが、接続が拒否されたことを示すメッセージを受け取りました。

考えられる原因: サーバー上で Java Web Console が実行されていません。

解決方法: Java Web Console を起動するには、スーパーユーザーになって、次のコマンドを実行します。

```
#smcwebserver status
#smcwebserver start
```

ログインできない

症状: Java Web Console のログインページで、ユーザーとパスワードの組み合わせが拒否されます。

考えられる原因: 対応する UNIX ユーザーアカウントが存在しません。

解決方法: システム上で対応する UNIX ユーザー名とパスワードが構成されていることを確認してください。必要に応じて、テスト用にローカルの UNIX ユーザーアカウントを作成します。

Desktop Manager のリンクがない

症状: Java Web Console アプリケーションリストのページに、 Sun Desktop Manager のリンクが表示されません。

考えられる原因: Desktop Manager モジュールがインストールされていません。

解決方法: Desktop Manager が Java Web Console にインストールされているかどうかを確認するには、スーパーユーザーになって、次のコマンドを実行します。

```
# smreg list -a
```

リストに `com.sun.apoc.manager_<version>` アプリケーションが含まれていない場合は、Desktop Manager をインストールする必要があります。

Null ポインタ例外、Tomcat/Java エラー、または空白ページ

症状: Desktop Manager を起動したが、空白ページまたはエラーメッセージのみが表示されます。

考えられる原因: エラーが `NoClassDefFoundError:sun/tools/javac/Main` と表示される場合は、Java Web Console が間違ったバージョンの Java を使用しています。

解決方法: 現在の Java Web Console の Java 環境は、`# smreg list -p` コマンドを実行して、`java.home` プロパティーを調べることで確認できます。このプロパティーが有効な Java ホームを指している必要があり、そのホームは JDK である必要があります。この値の設定が正しくない場合は、次のコマンドを実行する必要があります。

```
# smreg add -p java.home=<JAVA_HOME>
```

注 – *<JAVA_HOME>* は、`bin` サブディレクトリに見られる `javac` など、有効なインストールを指す必要があります。

そのあとで、次のコマンドで Java Web Console を再起動する必要があります。

```
# smcwebserver restart
```

SSL LDAP サーバーに接続できない

症状: リポジトリ作成ウィザードで、「SSL の使用」チェックボックスを選択するなど、LDAP サーバーの詳細を入力したあとで、「次へ」を押すと、サーバーに接続できないことを示すメッセージが表示されます。

考えられる原因: 不正なポート番号が入力された、LDAP サーバーがそのポート上で SSL を使用する接続を待機するように構成されていない、または適切な証明書が Java Web Console キーストアにありません。

解決方法: 最初に、LDAP サーバーが、ウィザードで指定したポート上の SSL 接続要求を待機するように構成されていることを確認します。これが正しい場合、認証局または LDAP サーバー証明書のいずれかが、`/etc/opt/webconsole/keystore` に置かれている Java

Web Console キーストアにあることを確認します。証明書は、`keytool -import -file <certificate file> -keystore /etc/opt/webconsole/keystore` コマンドで追加することができます。そのキーストアのデフォルトのパスワードは **`changeit`** です。その変更が Desktop Manager で表示されるようにするためには、`smcwebserver restart` コマンドを使用して、Java Web Console を再起動する必要があります。

ディレクトリに書き込めない

症状: ファイルベースまたはハイブリットのバックエンドの作成中に、「ディレクトリに書き込めません。」エラーが表示されます。

考えられる原因: noaccess ユーザーには、正しいアクセス権がありません。

解決方法: noaccess ユーザーに書き込み権を割り当てます。

◆ ◆ ◆ 第 3 章

3 クライアントコンポーネント

Desktop Manager から構成データにアクセスするには、デスクトップクライアントに Java Desktop System Configuration Agent が必要になります。Configuration Agent は、リモートの構成データリポジトリおよびアダプタと通信すると同時に、データを特定の構成システムに統合します。現在サポートされている構成システムは、GConf、Java Preferences、Mozilla Preferences、および StarSuite Registry です。

あるバージョンの Configuration Agent が Solaris 10 オペレーティングシステムに付属しています。しかし、Desktop Manager では、そのツールのより新しいバージョンが必要になります。その新しいバージョンは、Desktop Manager クライアントコンポーネントおよび関連するパッチの設定の一部としてインストールされます。

Desktop Manager クライアントコンポーネントをインストールするには

1. Desktop Manager zip アーカイブをダウンロードし、その内容を一時ディレクトリに取り出します。

   ```
   # unzip SunDesktopMgr-1.0.zip
   ```

2. 推奨パッチをインストールします。

   これらのパッチは、`SunDesktopMgr-1.0/<platform>/client/Patches` ディレクトリにあります。パッチごとに指定されるインストール手順に従います。

3. スーパーユーザー (root) になって、次のように設定スクリプトを実行します。

   ```
   # cd SunDesktopMgr-1.0/<platform>/client
   # ./setup
   ```

Configuration Agent

Configuration Agent は、次の表に示す各種のパッケージに含まれています。

Solaris のパッケージ名	説明
SUNWapbas	構成共有ライブラリ
SUNWapmsc	Configuration Agent に関するその他のファイル
SUNWapoc	Configuration Agent
SUNWapdc	Configuration Agent ウィザード

上記のパッケージをインストールすると、この API に必要なファイルもインストールされます。このパッケージは手作業でも Java Desktop System インストール経由でもインストールできます。インストール後は、自分のシステム上で Configuration Agent を構成および有効にする必要があります。

注 – Configuration Agent パッケージは、Java Desktop System インストールで Solaris の一部としてインストールされますが、Desktop Manager は、適切なレベルの機能を提供するために、インストール中にこれらのファイルにパッチをあてます。

リモートの構成データにアクセスするには、Configuration Agent に最小限のブートストラップ情報 (LDAP サーバーのホスト名やポートなど) が必要です。この情報は、`policymgr.properties`、`apocd.properties`、`os.properties` などのプロパティーファイルの集合として保守されます。これらの属性ファイルはローカルの `/etc/apoc` ディレクトリに格納されます。これらのプロパティーファイル (付録 A 参照) は手動でも、Configuration Agent の構成ウィザードでも編集できます。

この構成ウィザードは、グラフィカルユーザーインタフェースを提供し、これに従って Configuration Agent に必要な設定ができます。このウィザードのすべてのページで、対応するヘルプ画面が利用できます。このウィザードを起動するには、スーパーユーザー (root) として `/usr/bin/apoc-config` スクリプトを実行します。

注 – ウィザードはグラフィカルインタフェースを起動しなくても起動できます。たとえば、`/usr/bin/apoc-config -nodisplay` を実行すると、ウィザードはコンソールモードで起動します。

ブートストラップ情報

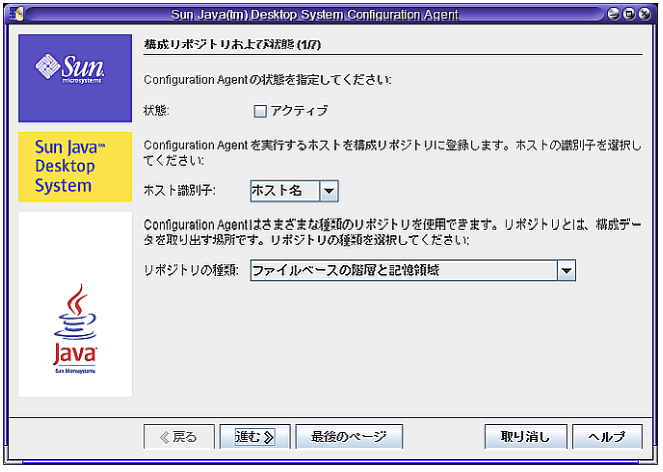

図 3–1 Configuration Agent、構成リポジトリ

注 – 対応するプロパティーファイルキーを括弧内に示しています (適切な場合)。

- 状態: Configuration Agent の状態。このチェックボックスを使用すると、Configuration Agent をアクティブまたは非アクティブにできます。構成リポジトリを使用するには、Configuration Agent をアクティブにしておく必要があります。アクティブにすると、Solaris 上のサービス管理機能 (smf(5)) への必要な登録も自動的に行われます。
- ホスト識別子 (HostIdentifierType): 「ホスト名」または「IP アドレス」のいずれかを指定できます。ホスト固有のポリシーデータを検索するとき、Configuration Agent は、ホスト名または IP アドレスのいずれかによって現在のホストを識別します。選択した「リポジトリタイプ」でホストを識別する方法に基づいて正しい値を選択します。
- リポジトリタイプ: この設定を使用して、組織階層と構成データが LDAP またはファイルベースの記憶領域、あるいはその両方に定義されているかどうかを Configuration Agent に示します。

注 – 手動で Configuration Agent を有効または無効にするには、**root** としてログインして、有効にする場合は `/usr/lib/apoc/apocd enable` コマンドを入力し、無効にする場合は `/usr/lib/apoc/apocd disable` コマンドを入力します。

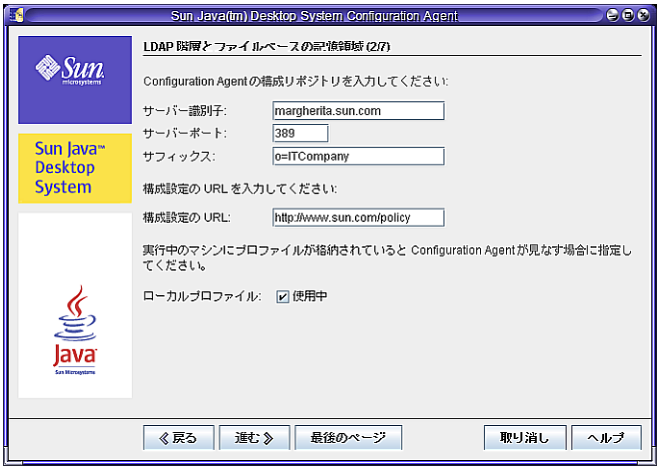

図 3–2 Configuration Agent、LDAP 階層とファイルベースの記憶領域

注 – 図 3–2 の画面は、前の画面で選択した「リポジトリタイプ」により異なります。「LDAP」または「ハイブリッド」リポジトリタイプを選択した場合、「サーバー識別子」、「サーバーポート」、および「サフィックス」が必須です。「ファイルベース」または「ハイブリッド」リポジトリタイプを選択した場合、「構成設定 URL」が必須です。

- サーバー識別子: LDAP サーバーのホスト名。
- サーバーポート: LDAP サーバーのポート番号。
- サフィックス: LDAP リポジトリのベース DN。
- 構成設定 URL: ファイルベースのリポジトリの場所を指定する URL。

  URL のリストを使用して、最初の URL への接続が成功しなかった場合の代替リポジトリを指定できます。リストは、`file://<filepath>`、`http://<host>:<port>/<filepath>`、または `https://<host>:<port>/<filepath>` の形式の、空白文字で区切られた 1 つ以上の URL で構成できます。詳細は、付録 A を参照してください。

注 – エージェントは、最初に SSL 接続を使用して、LDAP サーバーにアクセスしようとします。接続に失敗すると、エージェントはプレーン SSL 接続を試みます。

SSL 接続に成功するためには、適切な証明書が Java 実行環境のキーストアにある必要があります。キーストアは、標準 JRE 向けは `<installation directory>/lib/security/cacerts` に置かれ、標準 JDK 向けは `<installation directory>/jre/lib/security/cacerts` に置かれています。認証局または LDAP サーバー証明書のいずれかが、`keytool -import -file <certificate file> -keystore <cacerts file location>` コマンドを使用して、そのストアに追加される必要があります。そのキーストアのデフォルトのパスワードは **`changeit`** です。

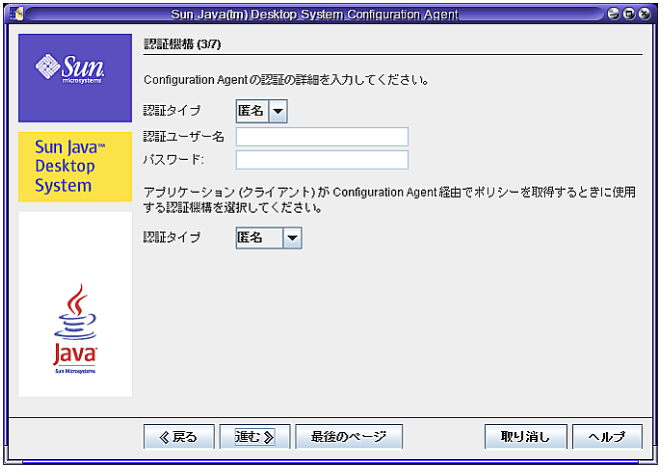

図 3–3 Configuration Agent、認証機構

- Configuration Agent の認証タイプ: 「匿名」または「簡易」のいずれかを指定できます。「匿名」を選択した場合、「認証ユーザー名」と「パスワード」フィールドは自動的に無効になります。
- 認証ユーザー名 (AuthDn): リポジトリに対する読み取り権および検索権を持つユーザーの完全な DN。
- パスワード (Password): 登録された LDAP ユーザーのパスワード。

注 – ディレクトリ内で匿名アクセスが有効である場合、「認証ユーザー名」と「パスワード」の設定は空白のままで構いません。

- アプリケーションの認証タイプ (AuthType): LDAP サーバーによるユーザーの認証方法に応じて、「匿名」または「GSSAPI」のいずれかを指定できます。

注 – 詳細については、27 ページの「データアクセス/ユーザー認証」を参照してください。

ポートの設定

Configuration Agent は、次の 2 つのポートを使用します。

- エージェントポート (DaemonPort): エージェントがクライアントアプリケーションと通信するときに使用します。デフォルトは **38900** です。
- 管理ポート (DaemonAdminPort): エージェントコントローラプログラム apocd が、エージェントと通信するときに使用します。デフォルトは **38901** です。

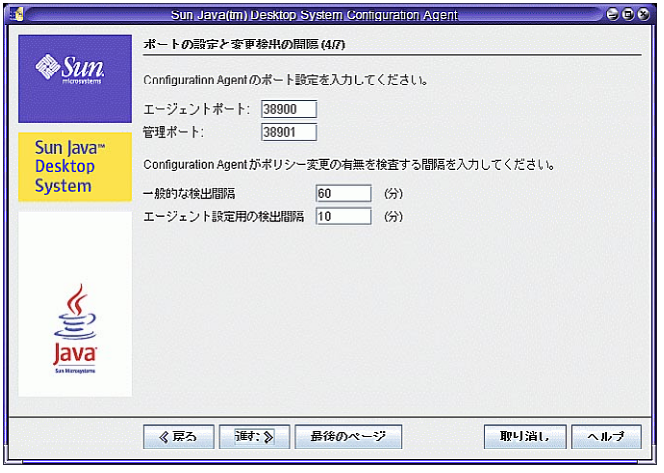

図 3–4 Configuration Agent、ポートの設定

変更検出間隔

Configuration Agent は、次の 2 つの間隔を使用して、構成データに変更があるかどうかを定期的にチェックします。

- 一般的な検出間隔 (ChangeDetectionInterval): デスクトップアプリケーション (クライアント) の構成データの変更検出サイクルの間隔 (分)。

注 – **-1** を指定すると、変更検出がオフになります。

- エージェント設定用の検出間隔 (DaemonChangeDetectionInterval): エージェントに固有な構成設定の変更検出サイクルの間隔 (分)。

注 – **--1** を指定すると、変更検出がオフになります。

汎用の検出間隔を使用すると、リモートの構成データの変更をクライアント側のアプリケーションに伝播する間隔を調整できます。この設定で指定する値は、リモートに加えられた変更の内容がクライアントアプリケーションに反映されるまでの最大期間（分）です。

この値が小さくなるほど、Configuration Agent と LDAP サーバーの活動が増えます。したがって、この設定値を調整する場合は注意が必要です。たとえば、最初の配備段階でこの値を 1 分に設定すれば、クライアントアプリケーションに対するリモート構成の影響を簡単にテストできます。テストが完了したら、この設定を初期値に戻します。

操作設定

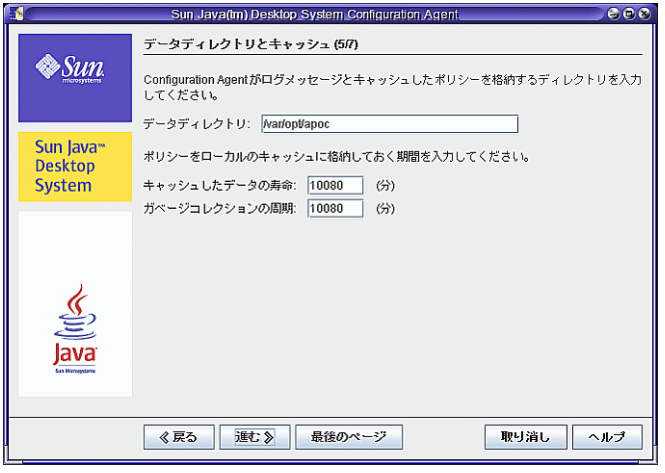

図 3–5 Configuration Agent、データディレクトリ

次の設定が構成できます。

- データディレクトリ (DataDir): 実行時データを格納するために使用されるディレクトリ。デフォルトは **/var/opt/apoc** です。
- キャッシュしたデータの寿命 (TimeToLive): ローカルデータベース内にオフラインでない構成データが留まる間隔 (分)。

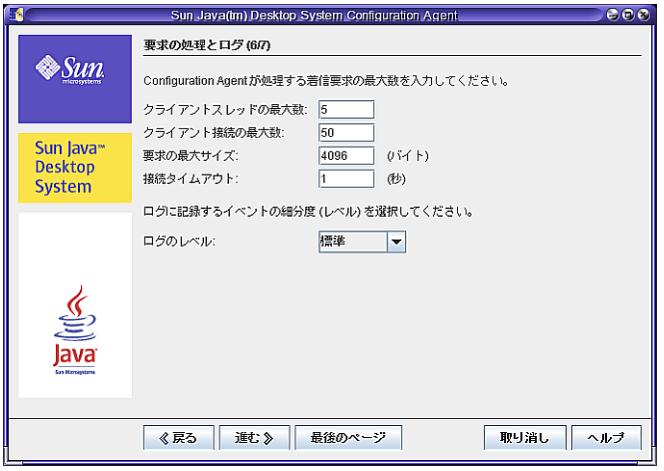

図 3–6 Configuration Agent、要求の処理とロギング

- ガベージコレクションのサイクル (GarbageCollectionInterval): ローカル構成データベースのガベージコレクションサイクルの間隔 (分)。
- クライアントスレッドの最大数 (MaxClientThreads): 同時に処理できるクライアント要求の最大数。
- クライアント接続の最大数 (MaxClientConnections): クライアント接続の最大数。
- 要求の最大サイズ (MaxRequestSize): クライアント要求の最大サイズ。
- 接続タイムアウト (ConnectTimeout): LDAP サーバーが接続要求へ応答するのに許可される間隔 (秒)。デフォルトは 1 秒です。
- ログのレベル (LogLevel): エージェントのログファイルの詳細レベル。ロギングレベルは Java Logger のレベルに一致します。次に、これらのレベルを重要度の降順に示します。
  - 簡潔
  - 警告
  - 標準
  - 構成
  - 詳細
  - より詳細
  - かなり詳細

注 – ほとんどの操作設定 (「データディレクトリ」と「接続タイムアウト」の設定を除く) は、LDAP サーバーに格納された対応するポリシー経由で集中的に保守できます。この機能を使用する場合は、対応する設定をウィザードで変更しないでください。代わりに、Desktop Manager 内の Configuration Agent ポリシーを使用して、操作設定を集中的に指定します。

エージェント設定の適用

Desktop Manager を使用して LDAP サーバーに格納した操作設定 (「データディレクトリ」と「接続タイムアウト」の設定を除く) は、エージェント構成の次回の変更検出サイクルで自動的に適用されます (DaemonChangeDetectionInterval を参照)。

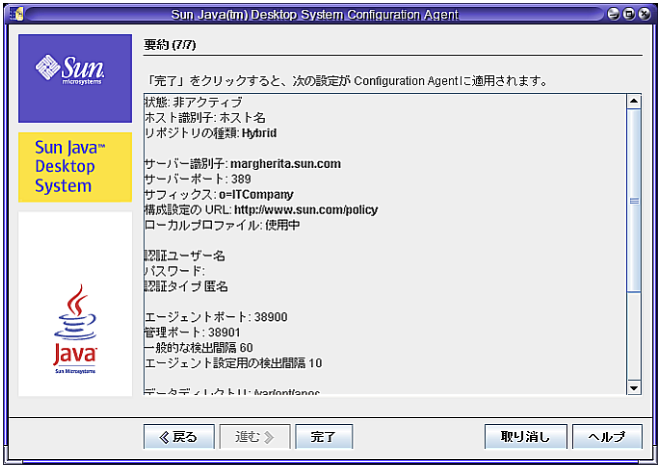

図 3–7 Configuration Agent、設定の要約ページ

ローカルで変更したその他の設定については、Configuration Agent を再読み込みまたは再起動する必要があります。構成ウィザードを使用する場合、再読み込みまたは再起動は自動的に実行されます。

注 – Configuration Agent を手作業で再起動するには、関連するクライアントアプリケーションが動作していないことを確認し、root としてログインして、コマンド `/usr/lib/apoc/apocd restart` を入力します。

エージェントの追加設定

注 – 次の設定は、構成ウィザードでは設定できません。

- ローカルポリシーの適用 (ApplyLocalPolicy): この設定を使用して、ローカルホスト上で使用可能なポリシーデータをクライアントアプリケーションで使用可能にする必要があるかどうかを指定します。「true」の値は、ローカルのポリシーデータを使用可能にする必要があることを示しています。「false」の値は、ローカルのポリシーデー

タを使用可能にするべきではないことを示しています。詳細は、26 ページの「ローカルポリシーの使用」 を参照してください。

ローカルポリシーの使用

Configuration Agent を構成して、全体的なポリシーに加えてまたは代替として、ローカルで配備されたポリシーから構成設定を適用することができます。次の手順に従って、このようなローカルポリシーを配置します。

▼ ローカルポリシーを配置する

1. **Desktop Manager** を使用して、必要なポリシー設定でプロファイルを作成します。
2. **Desktop Manager** を使用して、**zip** ファイルにプロファイルをエクスポートします。
3. クライアントホスト上で、`${DataDir}/Policies/profiles/PROFILE_REPOSITORY_default` ディレクトリをまだ存在していない場合は、作成します。

   `${DataDir}` は、デフォルトで `/var/opt/apoc` になる Configuration Agent のデータディレクトリの値を表します。
4. 前にエクスポートした **zip** ファイルを `${DataDir}/Policies/profiles/PROFILE_REPOSITORY_default` にコピーします。
5. 使用可能なローカルポリシーを適用するように **Configuration Agent** が構成されていることを確認します。詳細は、**25** ページの「エージェントの追加設定」 を参照してください。

注 – Configuration Agent の「ApplyLocalPolicy」設定を変更した場合、`root` としてログインし、`/usr/lib/apoc/apocd reload` コマンドを入力して、Configuration Agent を再起動する必要があります。

この方法で配置されたローカルポリシーは、Configuration Agent による次の変更検出サイクル中にクライアントが使用できるようになります。

Configuration Agent の自動起動

障害の発生時に、Configuration Agent は自動的に再起動されます。サービス管理機能 (smf(5)) により、この決定がなされます。すでに数多くの障害が発生している場合など、再起動が不適切であるとサービス管理機能が判断すると、Configuration Agent は保守モードになります。

Configuration Agent が再起動しない場合は、root としてログインし /usr/lib/apoc/apocd disable コマンドを実行してエージェントを一時的に無効にして、エージェントに障害を発生させた問題を修正し、/usr/lib/apoc/apocd enable コマンドを実行してエージェントを再度有効にしてください。

データアクセス/ユーザー認証

Configuration Agent は、デスクトップユーザーのログイン ID に基づいて LDAP サーバーから情報を取得します。組織のマッピングファイルの User/UniqueIdAttribute 設定により、ログイン ID と、LDAP サーバー内のユーザー要素がマッピングされます。Configuration Agent はまた、ホストについての情報 (ホストの名前と IP アドレスなど) を取得します。この情報は、組織のマッピングファイルの Host/UniqueIdAttribute 設定により、LDAP サーバー内のホスト要素にマッピングされます。組織のマッピングの詳細は、付録 C を参照してください。

LDAP サーバーにアクセスする場合、匿名または GSSAPI の 2 つの方法があります。匿名アクセスでは、デスクトップ上での処理は必要ありません。GSSAPI 方式では、デスクトップ上で Kerberos 資格情報を取得する必要があります。Kerberos 資格情報とユーザーログインの取得を統合するには、pam_krb5 モジュールを Java Desktop System ホストにインストールおよび構成する必要があります。

gdm を使用して Kerberos とユーザーログインを統合できます。たとえば、次のように /etc/pam.d/gdm ファイルを使用します。

```
#%PAM-1.0
auth    required     pam_unix2.so  nullok #set_secrpc
auth    optional   pam_krb5.so use_first_pass missing_keytab_ok ccache=SAFE putenv_direct
account required     pam_unix2.so
password required     pam_unix2.so  #strict=false
session required     pam_unix2.so  # trace or none
session required     pam_devperm.so
session optional     pam_console.so
```

このように Kerberos をユーザーログインに統合する場合は、スクリーンセーバーの Kerberos サポートを有効にするとよいでしょう。たとえば、次のように /etc/pam.d/xscreensaver ファイルを使用します。

```
auth required pamkrb5.so use_first_pass missing_keytab_ok
ccache=SAFE putenv_direct
```

アダプタ

アプリケーションアダプタは、Desktop Manager がサポートする構成システムの拡張機能です。アダプタにより、各種アプリケーションは、構成システムに応じて、中央の構成データを取り入れることができます。次の構成システムがサポートされます。

- GConf: デスクトップと、Evolution などのほとんどの GNOME アプリケーションに使用される GNOME 構成システム
- StarOfficeRegistry: StarSuite と OpenOffice.org が使用する構成システム
- Mozilla Preferences: Mozilla が使用する構成システム
- Java Preferences: Java アプリケーションに提供される構成 API

ユーザーのデスクトップにデスクトップランチャー、メニュー項目、および起動プログラムを追加する、デスクトップ定義アダプタも提供されます。

GConf アダプタ

GConf アダプタは、Solaris 用 `SUNWapoc-adapter-gconf` パッケージに含まれます。該当する packageAdapter からアダプタをインストールすると、`/etc/gconf/2/path` にある GConf データソースパスは Desktop Manager ソースを含むように更新されます。アダプタから提供される 2 つのデータソースは、次のとおりです。

- "apoc:readonly:": ポリシーの非保護設定へのアクセスを可能にします。ユーザー設定のあと、ローカルのデフォルト設定の前にこのデータソースを挿入します。
- "apoc:readonly:mandatory@": ポリシーの保護設定へのアクセスを可能にします。ローカルの必須設定のあと、ユーザー設定の前にこのデータソースを挿入します。

GConf アダプタの構成

GConf アダプタはインストールの一部として構成されますが、その操作は、必須の中央設定とデフォルト設定を表す 2 つのデータソースが GConf パスファイル (`/etc/gconf/2/path`) に存在するかどうかによって異なります。このパスファイルには、システムのインストール後に GConf が予期したとおりに中央設定を取り入れるための適切な情報が含まれていますが、管理者は、そのパスを変更して追加のカスタムソースを含める必要がある場合は、「apoc」の接頭辞が付けられたデータソースがファイル内にまだ存在していることを確認します。また、データソースが、必須の中央設定を表すデータソースに対してローカルの必須設定とユーザー設定の間に配置され、デフォルトの中央設定を表すデータソースに対してユーザー設定とローカルのデフォルト設定の間に配置されていることも確認する必要があります。

Java Preferences アダプタ

Java Preferences アダプタは、Solaris 用 `SUNWapcj` パッケージに含まれます。

Java Preferences アダプタの構成

Java Preferences アダプタは、JRE が提供するデフォルトのファイルベースシステムなどほかの既存の実装に対してラッパーとして使用される必要がある Preferences API の実装として提供されます。Preferences API を利用する Java アプリケーションで中央構成を使用できるようにするには、ヘルパーとして `/usr/lib/apoc/apocjlaunch` スクリプトを使用して、そのアプリケーションの起動スクリプトを作成する必要があります。このスクリプトには、2、3 の環境変数を指定して、必要な環境で Java アプリケーションを起動する `apocjlaunch` スクリプトを最後に含める必要があります。次の環境変数を設定する必要があります。

- `JAVA`: Java 実行時実行可能ファイルへのパスを含みます。
- `APPLICATION`: そのアプリケーションに対する定期的な Java 実行時起動の後続部分を含みます。たとえば、シングルクラス起動の場合は *classname [arguments]*、jar アーカイブ起動の場合は `-jar` *jarname [arguments]* となります。

次の省略可能な追加の環境変数を設定できます。

- `CLASSPATH`: アプリケーションクラスパスに含める必要がある、コロンで区切られた jar またはクラスファイルのリスト
- `DEFINES`: アプリケーション起動に含める必要がある define 文を含む文字列
- `PREFFACTORY`: アプリケーションが使用する必要がある基本的な Preferences API 実装でのファクトリのクラス名

Mozilla アダプタ

Mozilla アダプタは、Solaris 用 `SUNWmozapoc-adapter` パッケージに含まれます。

Mozilla アダプタの構成

Mozilla アダプタは、この製品のインストールの一部として設定され、追加で構成する必要はありません。

StarSuite アダプタ

StarSuite アダプタは標準の StarSuite インストールに含まれています。これにより、特殊な変更を加えることなく、プロファイル構成データにアクセスできます。

StarSuite アダプタの構成

StarSuite アダプタは、この製品のインストールの一部として設定され、追加で構成する必要はありません。

Desktop Definition アダプタ

Desktop Definition アダプタは、次のパッケージで構成されます。

パッケージ名	説明
SUNWapleg	構成アクセスバイナリ
SUNWardsa	デスクトップ定義アダプタ
SUNWardsa-misc	アダプタ用システム統合

これらのパッケージは、Desktop Manager クライアントコンポーネントのインストール時にインストールされ、追加で設定する必要はありません。

Desktop Definition アダプタの構成

Desktop Definition アダプタは、ユーザーがログインするたびに使用される設定処理によって構成され、追加で設定する必要はありません。

アダプタの削除

Mozilla アダプタと StarSuite アダプタは、これらの製品が削除されると削除されます。GConf、Java Preferences、および Desktop Definition アダプタは、適切なパッケージ管理システムツールを使用して、インストールのセクションで説明したパッケージを削除することで削除できます。

Java Preferences アダプタを削除した場合、Preferences API を使用する Java アプリケーションを起動するために作成された起動スクリプトは使用しないでください。いくつかの必要なクラスが使用できなくなるため、起動スクリプト内の Java の起動が失敗します。

アダプタのトラブルシューティング

対応するアプリケーションで中央の構成データが表示されない原因となる問題の大部分は、Configuration Agent に起因する可能性があります。これは、データを取得するためにすべてのアダプタが使用する共通のメカニズムであるためです。

中央構成の変更が特定の設定 (またはそのグループ) に対して効果がない場合、ユーザーが、通常は製品の「オプション」ダイアログや「設定」ダイアログを使用して、アプリケーション内のその設定に対して明示的に値を設定していることが考えられます。この場合、中央設定をプロテクトとして、つまり管理者が設定してユーザーは変更できないように定義されないかぎり、ユーザー設定は、Desktop Manager を使用して設定された値よりも優先度が高くなります。

Configuration Agent のトラブルシューティング

このセクションでは、Configuration Agent の性質と動作に関してユーザーが抱く可能性がある疑問と、エージェントの問題のトラブルシューティングを行うためのいくつかの注意事項について説明します。

質問と回答

Configuration Agent とは何ですか、またその動作はどのようなものですか

Configuration Agent は、ポリシーをキャッシュし配信するアプリケーションです。アプリケーションとそれが実行されるホストの性能に重大な影響を与えずに、デスクトップクライアントアプリケーションを確実に一元的に構成できるように設計され構築されています。この機能は、次のことにより達成されます。

- クライアントが再利用できるように、任意のダウンロードされたポリシーをローカルで使用可能なキャッシュに書き込む
- ポリシーがホストされている LDAP サーバーへの接続などの、共有可能でかつ共有すべき、任意の高価なリソースを共有する

クライアントアプリケーションと Configuration Agent との間で対話が発生する典型的なシナリオは、非常に単純で、次のように説明できます。

1. ユーザーが、gconfd、Mozilla、StarSuite など、関連するデスクトップクライアントアプリケーションの 1 つを起動します。
2. クライアントアプリケーションが Configuration Agent に接続します。
3. クライアントアプリケーションは、Configuration Agent から必要とするポリシーデータを要求します。
4. Configuration Agent は、要求されたポリシーデータを探してキャッシュを検索します。
5. ポリシーデータがキャッシュ内に見つからなかった場合、Configuration Agent は、事前に構成されたポリシーリポジトリから要求されたデータをダウンロードして、キャッシュに格納します。
6. ポリシーデータは、要求を行なったクライアントアプリケーションに送信されます。
7. Configuration Agent は、ポリシーデータに対する変更についてポリシーリポジトリを監視します。
8. 変更が検出されると、Configuration Agent はキャッシュをリフレッシュして、最新の状態にし、クライアントアプリケーションに変更があることを知らせます。

Configuration Agent を入手してインストールするにはどのようにするのですか

Configuration Agent は、Solaris 10 にデフォルトで付属し、インストールされています。

Solaris 10 をインストールしたあとで、何をすべきですか

Configuration Agent は、デフォルトでは無効になっており、構成されていません。Configuration Agent を使用するためには、少なくとも最小限に構成を行い、有効にする必要があります。これらの手順を完了すると、次回の起動時に、デスクトップクライアントアプリケーションが提供されたポリシーを自動的に使用し始めます。

Configuration Agent を構成する方法について知りたいのですが

Configuration Agent を正しく構成するには、Configuration Agent ウィザードを使用します。スーパーユーザーとして `/usr/bin/apoc-config` コマンドを実行すると、ウィザードを起動できます。ウィザードに従って、エージェントを正しく構成するために必要な手順を実行します。ほとんどの場合では、ウィザードを完了するために絶対必要な情報は、ポリシーリポジトリの場所だけです。

また、構成ファイルを手動で編集して、Configuration Agent を構成することも可能です。この場合、エージェントは間違って構成されやすいため、この方法はお勧めしません。さらに、Configuration Agent ウィザードには、特定の構成変更がエージェントの再起動または再ロードを必要とするかどうかを判断するロジックも追加されています。

Configuration Agent を有効にするにはどうすればよいのでしょうか

次の 3 つのメカニズムのいずれかを使用して、エージェントを有効にすることができます。

1. Configuration Agent ウィザード (`/usr/bin/apoc-config`) を使用して、「状態」を「アクティブ」に設定します。
2. Configuration Agent コントローラプログラム (`/usr/lib/apoc/apocd`) を使用して、スーパーユーザーとして次のコマンドを実行します。

   ```
   /usr/lib/apoc/apocd enable
   ```

3. smf(5) を使用して、スーパーユーザーとして次のコマンドを実行します。

   ```
   /usr/sbin/svcadm enable svc:/network/apocd/udp
   ```

Configuration Agent を構成し有効にしたあとで、正常に動作していることを確認するにはどうすればよいのでしょうか

Configuration Agent が正しく構成され動作していることを確認するもっとも簡単な方法は、Desktop Manager でポリシーを作成し、そのポリシーをユーザーに割り当てることです。そのユーザーとしてデスクトップマシンにログインして、作成したポリシー設定が

使用されていることを確認します。背景やテーマなど、デスクトップセッションで容易に検出できる多くのポリシー設定があります。

Configuration Agent を有効にすることにはどのような意味があるのでしょうか

Configuration Agent は、smf(5) 準拠のサービスであり、したがって、エージェントを有効にするという意図は smf(5) に起因します。エージェントを有効にすると、エージェントはサービスを提供する準備が整います。エージェントを有効にしたとき、次の処理が発生します。

- エージェントが起動する
- エージェントが有効にされたあとで起動されたデスクトップクライアントアプリケーションが、ポリシーデータを取得できる
- システムのブート中に、エージェントは自動的に再起動する

Configuration Agent が有効かどうかを確認する方法を教えてください

次の方法のいずれか 1 つを使用して、Configuration Agent が有効かどうかを判断できます。

- Configuration Agent のコントローラプログラムを使用します。スーパーユーザーになって、次のコマンドを入力します。

  ```
  /usr/lib/apoc/apocd is-enabled
  ```

  エージェントが有効の場合、コントローラプログラムは次のメッセージを返します。

  ```
  Checking Configuration Agent enabled status ... Enabled
  ```

  エージェントが無効の場合、コントローラプログラムは次のメッセージを返します。

  ```
  Checking Configuration Agent enabled status ... Not enabled
  ```

- smf(5) を使用して、次のコマンドを実行します。

  ```
  /usr/bin/svcs svc:/network/apocd/udp:default
  ```

  エージェントが有効の場合、svcs は次のメッセージを返します。

  ```
  STATE          STIME    FMRI
  online          8:36:04 svc:/network/apocd/udp:default
  ```

  エージェントが無効の場合、svcs は次のメッセージを返します。

  ```
  STATE          STIME    FMRI
  disabled       15:58:34 svc:/network/apocd/udp:default
  ```

  エージェントが保守モードの場合、svcs は次のメッセージを返します。

  ```
  STATE          STIME    FMRI
  maintenance     8:38:42 svc:/network/apocd/udp:default
  ```

Configuration Agent が稼動しているかどうかを確認するにはどのようにすればよいのでしょうか

次の方法のいずれか 1 つを使用して、Configuration Agent が稼動しているかどうかを判断します。

- スーパーユーザーになって、Configuration Agent コントローラプログラムを実行します。

  ```
  /usr/lib/apoc/apocd status
  ```

  エージェントが有効の場合、コントローラプログラムは次のメッセージを返します。

  ```
  Checking Configuration Agent status ... Running
  ```

  エージェントが無効の場合、コントローラプログラムは次のメッセージを返します。

  ```
  Checking Configuration Agent status ... Not running
  ```

- 次のコマンドを実行します。

  ```
  /usr/bin/svcs svc:/network/apocd/udp:default
  ```

  エージェントが稼動している場合、svcs は次のメッセージを返します。

  ```
  STATE          STIME    FMRI
  online          8:36:04 svc:/network/apocd/udp:default
  ```

  エージェントが稼動していない場合、svcs は次のメッセージを返します。

  ```
  STATE          STIME    FMRI
  disabled       15:58:34 svc:/network/apocd/udp:default
  ```

  エージェントが保守モードの場合、svcs は次のメッセージを返します。

  ```
  STATE          STIME    FMRI
  maintenance     8:38:42 svc:/network/apocd/udp:default
  ```

- 次のコマンドを実行します。

  ```
  ps -ef | grep apoc
  ```

  Configuration Agent が稼動している場合、ps 出力に、次の関連付けされた Java 処理が表示されます。

  ```
   daemon 29295 29294  0 13:05:22?  0:03 java -Djava.library.path=/usr/lib/apoc
   -cp /usr/share/lib/apoc/apocd.jar:/usr/s
   daemon 29294   1   0 13:05:22?  0:00 sh -c java
  -Djava.library.path=/usr/lib/apoc -cp /usr/share/lib/apoc/apocd.jar:
   root 29345 28134  0 13:08:59 pts/1   0:00 grep apoc
  ```

ログファイルの場所はどこですか

Configuration Agent の問題のトラブルシューティングを行う必要があるときは、次のログファイルを調べることができます。

- smf(5) ログファイル:
  - `/var/svc/log/network-apocd-udp:default.log` ファイルには、Configuration Agent の特定のインスタンスを開始または停止しようとする試みに関連するイベントが記録されます。また、ファイルには、Configuration Agent コントローラプログラム、`/usr/lib/apoc/apocd` が標準出力に書き込みを行なったというメッセージと、JVM または Configuration Agent の出力メッセージも含まれます。
  - `/var/svc/log/svc.startd.log` ログファイルは、高レベルの smf(5) イベントの記録を保持します。たとえば、Configuration Agent を起動しようとする試みが短時間で連続して複数回発生した場合、 smf(5) は、Configuration Agent が起動できないと判断することがあります。これが発生した場合、smf(5) は Configuration Agent を保守モードにして、その結果にログエントリを書き込みます。

  一般に、Configuration Agent 起動の問題が発生したとき、これらのログファイルの両方が役立ちます。
- Configuration Agent ログ:

  Configuration Agent は、デフォルトのログディレクトリ `/var/opt/apoc/Logs` にあるログファイルにログメッセージを書き込みます。Configuration Agent の「データディレクトリ」は `/var/opt/apoc` です。Configuration Agent ウィザード (`/usr/bin/apoc-config`) アプリケーションを使用して、このディレクトリの場所を変更することができます。Configuration Agent ウィザードで「ログのレベル」を変更して、ログメッセージの詳細度を変更することができます。Configuration Agent が正しく構成されていない場合、または何らかのエージェントの傷害が発生した場合は、Configuration Agent ウィザードを使用して、ログのレベルを「Finest」に設定してから、エージェントのログファイルを調べてください。この手順により、利用可能な最大量のログ情報を確実に取得できます。
- システムログ:

  `/var/adm/messages` ログファイル、または SunRay マシンにある `/var/opt/SUNWut/log/messages` ログファイルをチェックして、Configuration Agent の問題を診断することもできます。

エージェントのログ記録メカニズムの詳細度を高めるにはどのようにするのですか

34 ページの「ログファイルの場所はどこですか」を参照してください。

保守モードとはどのようなものですか

smf(5) は、エージェントの起動または再起動で問題を検出すると、Configuration Agent を保守モードにします。smf(5) はエージェントの起動に失敗すると、エージェントの起動が成功するまで繰り返し再起動を試みるか、または smf(5) はエージェントが起動できないと判断します。後者の場合、smf(5) はエージェントを保守モードにして、発生した問題に対処する必要があることを知らせます。問題に対処したあとで、エージェントの smf(5) 状態をクリアして、通常の操作に戻ることができます。

保守モードから抜け出す、つまり **smf(5)** 状態をクリアするにはどうするのですか

スーパーユーザーになって、`/usr/sbin/svcadm clear svc:/network/apocd/udp` コマンドを実行します。

Configuration Agent が不意に実行を停止した場合、何が起こりますか

smf(5) は、エージェントが停止したことを検出し、エージェントの再起動を試みます。何らかの理由により再起動の試みが連続して失敗した場合、smf(5) はエージェントを保守モードにします。エージェントが正常に再起動された場合、デスクトップクライアントアプリケーションの実行は影響を受けません。このようなデスクトップクライアントアプリケーションは、再起動時にエージェントに自動で再接続します。

エージェントを有効または起動した場合、デスクトップクライアントアプリケーションを再起動する必要がありますか

実際に行う処置は、特定のデスクトップクライアントアプリケーションの起動時にエージェントが有効にされ稼動していたかどうかによって異なります。エージェントが有効にされ稼動していた場合、 クライアントアプリケーションはエージェントへの接続を確立しており、接続が切断されると再接続を試みます。つまり、エージェントを起動、有効、または無効にするたびに、クライアントアプリケーションは常に、エージェントが実行していれば再接続を試みます。クライアントアプリケーションの起動時にエージェントが有効にされておらず、稼動していなかった場合、クライアントアプリケーションは Configuration Agent を使用せず、エージェントの起動時にも接続を試みません。実際に、これは次のことを意味します。

- エージェントが有効にされ稼動している間に起動したデスクトップクライアントアプリケーションは、再起動する必要がありません。
- エージェントが有効にされておらず稼動していないときに起動したデスクトップクライアントアプリケーションは、再起動する必要があります。

デスクトップクライアントアプリケーションが構成済みポリシーを使用していない場合はどのようにすればよいでしょうか

Configuration Agent に関連するもっとも一般的な問題は、構成済みポリシーの効果がデスクトップクライアントアプリケーションに表れないことです。この問題のもっとも一般的な原因は、エージェントが間違って構成されている、ポリシーリポジトリが間違って構成されている、またはポリシーリポジトリが使用できないということです。次のガイドラインに従って、このような問題を検出し取り除くことができます。

- エージェントが構成されていることを確認します。
- エージェントが有効にされ稼動していることを確認します。エージェントを起動する必要がある場合、現在開いているデスクトップクライアントアプリケーションも再起動する必要があります。

- 問題が解決しない場合は、エージェントログ記録メカニズムの詳細度を一時的に上げ、可能であればエージェントを再起動して、エージェントの起動時からの完全で詳細なログメッセージを得られるようにします。
- エージェントが正しく起動できない場合、37 ページの「Configuration Agent を起動するときの問題」 のセクションを調べてください。
- エージェントが正しく起動したが、デスクトップクライアントアプリケーションが使用可能なポリシーを使用しない場合は、「実行中の Configuration Agent からポリシーを取得するときの問題」のセクションを調べてください。
- それでも問題を解決できない場合は、技術サポートにお問い合わせください。

Configuration Agent を起動するときの問題

Configuration Agent が起動できず、Configuration Agent を構成し有効にしている場合は、ログファイルを調べる必要があります。次のセクションでは、この問題のもっとも一般的なエラーについて説明します。

エージェントデータディレクトリが無効またはアクセス不可

ログファイル、ポリシーキャッシュなどを格納するために、エージェントは Configuration Agent データディレクトリを作成および使用します。このディレクトリは、デフォルトで `/var/opt/apoc` に置かれます。

データディレクトリがアクセスできない場所、つまり `/dev/null/cant/write/here` に設定されると、Configuration Agent は smf(5) ログに次のエラーメッセージを生成します。この問題を解決するには、Configuration Agent ウィザード (`/usr/bin/apoc-config`) を使用して、データディレクトリをアクセス可能な場所に指定します。

```
[ Nov 17 14:35:38 Executing start method ("/usr/lib/apoc/apocd svcStart") ]
Starting Configuration Agent ... Warning: Cannot create Log directory
   '/dev/null/cant/write/here/Logs'
Warning:Failed to create log file handler
Nov 17, 2005 2:35:39 PM com.sun.apoc.daemon.misc.APOCLogger config
CONFIG: Daemon configuration:
 MaxRequestSize = 4096
 DaemonAdminPort = 38901
 ThreadTimeToLive = 5
 DaemonChangeDetectionInterval = 10
 IdleThreadDetectionInterval = 15
 PROVIDER_URL =
 DataDir = /dev/null/cant/write/here
 ApplyLocalPolicy = true
 ChangeDetectionInterval = 60
 MaxClientConnections = 50
 GarbageCollectionInterval = 10080
 InitialChangeDetectionDelay = 10
 TimeToLive = 10080
 ConnectionReadTimeout = 5000
```

```
 DaemonPort = 38900
 LogLevel = FINEST
 MaxClientThreads = 5


Nov 17, 2005 2:35:39 PM Daemon main
FINER: THROW
com.sun.apoc.daemon.misc.APOCException
    at com.sun.apoc.daemon.apocd.Daemon.initAuthDir(Unknown Source)
    at com.sun.apoc.daemon.apocd.Daemon.init(Unknown Source)
    at com.sun.apoc.daemon.apocd.Daemon.<init>(Unknown Source)
    at com.sun.apoc.daemon.apocd.Daemon.main(Unknown Source)
[ Nov 17 14:36:08 Method or service exit timed out.  Killing contract 980 ]
[ Nov 17 14:36:08 Method "start" failed due to signal KILL ]
```

ビジー状態にあるクライアント要求ポートの使用

Configuration Agent は TCP/IP ソケット接続を使用して、デスクトップクライアントアプリケーションと通信します。デフォルトで、これらの接続はポート 38900 経由で行われます。

すでにほかのサービスが使用しているポート 1234 を使用するように Configuration Agent が構成されている場合、次のエラーメッセージが生成されます。エラーメッセージは、Configuration Agent ログに記録されます。この問題を解決するには、Configuration Agent ウィザード (`/usr/bin/apoc-config`) を使用して、「エージェントポート」の設定を未使用のポート番号に変更します。

```
Nov 17, 2005 2:50:59 PM com.sun.apoc.daemon.misc.APOCLogger config
CONFIG: Daemon configuration:
 MaxRequestSize = 4096
 DaemonAdminPort = 38901
 ThreadTimeToLive = 5
 DaemonChangeDetectionInterval = 10
 IdleThreadDetectionInterval = 15
 PROVIDER_URL =
 DataDir = /var/opt/apoc
 ApplyLocalPolicy = true
 ChangeDetectionInterval = 60
 MaxClientConnections = 50
 GarbageCollectionInterval = 10080
 InitialChangeDetectionDelay = 10
 TimeToLive = 10080
 ConnectionReadTimeout = 5000
 DaemonPort = 1234
 LogLevel = FINEST
 MaxClientThreads = 5
```

```
Nov 17, 2005 2:50:59 PM com.sun.apoc.daemon.misc.APOCLogger info
INFO: Daemon starting
Nov 17, 2005 2:50:59 PM com.sun.apoc.daemon.misc.APOCLogger fine
FINE: Garbage collection scheduled ( interval = 10080 minutes )
Nov 17, 2005 2:50:59 PM Daemon main
FINER: THROW
com.sun.apoc.daemon.misc.APOCException: java.net.BindException: Address already in use
    at com.sun.apoc.daemon.transport.ChannelManager.<init>(Unknown Source)
    at com.sun.apoc.daemon.apocd.Daemon.run(Unknown Source)
    at com.sun.apoc.daemon.apocd.Daemon.main(Unknown Source)
Caused by: java.net.BindException: Address already in use
    at sun.nio.ch.Net.bind(Native Method)
    at sun.nio.ch.ServerSocketChannelImpl.bind(ServerSocketChannelImpl.java:119)
    at sun.nio.ch.ServerSocketAdaptor.bind(ServerSocketAdaptor.java:59)
    at sun.nio.ch.ServerSocketAdaptor.bind(ServerSocketAdaptor.java:52)
```

ビジー状態にある管理ポートの使用

Configuration Agent は TCP/IP ソケット接続を使用して、Configuration Agent コントローラプログラム (`/usr/lib/apoc/apocd`) と通信します。デフォルトで、これらの接続はポート 38901 経由で行われます。

すでにほかのサービスが使用しているポート 1234 を使用するように Configuration Agent が構成されている場合、次のエラーメッセージが Configuration Agent ログに生成されます。この問題を解決するには、Configuration Agent ウィザード (`/usr/bin/apoc-config`) を使用して、「管理ポート」の設定を未使用のポート番号に変更します。

```
ONFIG: Daemon configuration:
 MaxRequestSize = 4096
 DaemonAdminPort = 1234
 ThreadTimeToLive = 5
 DaemonChangeDetectionInterval = 10
 IdleThreadDetectionInterval = 15
 PROVIDER_URL =
 DataDir = /var/opt/apoc
 ApplyLocalPolicy = true
 ChangeDetectionInterval = 60
 MaxClientConnections = 50
 GarbageCollectionInterval = 10080
 InitialChangeDetectionDelay = 10
 TimeToLive = 10080
 ConnectionReadTimeout = 5000
 DaemonPort = 38900
 LogLevel = FINEST
 MaxClientThreads = 5


Nov 17, 2005 2:55:11 PM com.sun.apoc.daemon.misc.APOCLogger info
```

```
INFO: Daemon starting
Nov 17, 2005 2:55:11 PM com.sun.apoc.daemon.misc.APOCLogger fine
FINE: Garbage collection scheduled ( interval = 10080 minutes )
Nov 17, 2005 2:55:11 PM com.sun.apoc.daemon.misc.APOCLogger fine
FINE: Client manager started
Nov 17, 2005 2:55:11 PM com.sun.apoc.daemon.misc.APOCLogger fine
FINE: Channel manager started
Nov 17, 2005 2:55:11 PM Daemon main
FINER: THROW
com.sun.apoc.daemon.misc.APOCException: java.net.BindException: Address already in use
    at com.sun.apoc.daemon.admin.AdminManager.initChannel(Unknown Source)
    at com.sun.apoc.daemon.admin.AdminManager.<init>(Unknown Source)
    at com.sun.apoc.daemon.apocd.Daemon.run(Unknown Source)
    at com.sun.apoc.daemon.apocd.Daemon.main(Unknown Source)
Caused by: java.net.BindException: Address already in use
    at sun.nio.ch.Net.bind(Native Method)
    at sun.nio.ch.ServerSocketChannelImpl.bind(ServerSocketChannelImpl.java:119)
    at sun.nio.ch.ServerSocketAdaptor.bind(ServerSocketAdaptor.java:59)
    at sun.nio.ch.ServerSocketAdaptor.bind(ServerSocketAdaptor.java:52)
    ... 4 more
```

実行中の **Configuration Agent** からポリシーを取得するときの問題

構成リポジトリの指定が欠落しているか無効である

Configuration Agent は、ポリシー情報をダウンロードまたはキャッシュに書き込むために有効な構成リポジトリに接続する必要があります。エージェントの構成で構成リポジトリを正しく特定していない場合、無効な形式の使用、リポジトリの未指定などにより、デスクトップクライアントアプリケーションの起動時に、次のようなエラーが Configuration Agent ログに書き込まれます。この問題を解決するには、Configuration Agent ウィザード (`/usr/bin/apoc-config`) を使用して、使用する構成リポジトリを特定します。

```
FINER: New client added
Nov 18, 2005 1:59:22 PM com.sun.apoc.daemon.misc.APOCLogger finer
FINER: CreateSession transaction started
Nov 18, 2005 1:59:22 PM com.sun.apoc.daemon.misc.APOCLogger finer
FINER: Creating new client session
Nov 18, 2005 1:59:22 PM com.sun.apoc.daemon.misc.APOCLogger finest
FINEST: Authenticating user geoffh
Nov 18, 2005 1:59:22 PM com.sun.apoc.daemon.misc.APOCLogger finest
FINEST: Authentication successful
Nov 18, 2005 1:59:23 PM PolicyBackend openPolicyBackend
FINER: THROW
com.sun.apoc.daemon.misc.APOCException: com.sun.apoc.daemon.misc.APOCException:
com.sun.apoc.spi.environment.InvalidParameterException: The parameter organisation
PROVIDER_URL#protocol (null) is not valid, the value must be comprised in
{ldaps,ldap,https,http,file}.
```

```
    at com.sun.apoc.daemon.apocd.PolicyBackend.<init>(Unknown Source)
    at com.sun.apoc.daemon.apocd.HostPolicyBackend.<init>(Unknown Source)
    at com.sun.apoc.daemon.apocd.PolicyBackendFactory.openPolicyBackend(Unknown Source)
    at com.sun.apoc.daemon.apocd.Cache$DataSource.openPolicyBackend(Unknown Source)
    at com.sun.apoc.daemon.apocd.Cache$DataSource.open(Unknown Source)
    at com.sun.apoc.daemon.apocd.Cache.createDataSources(Unknown Source)
    at com.sun.apoc.daemon.apocd.Cache.<init>(Unknown Source)
    at com.sun.apoc.daemon.apocd.CacheFactory.createNewCache(Unknown Source)
    at com.sun.apoc.daemon.apocd.CacheFactory.openCache(Unknown Source)
    at com.sun.apoc.daemon.apocd.Session.<init>(Unknown Source)
    at com.sun.apoc.daemon.transaction.CreateSessionTransaction.executeTransaction
(Unknown Source)
    at com.sun.apoc.daemon.transaction.Transaction.execute(Unknown Source)
    at com.sun.apoc.daemon.apocd.ClientEventHandler.handleEvent(Unknown Source)
    at com.sun.apoc.daemon.apocd.EventWorkerThread.run(Unknown Source)
Caused by: com.sun.apoc.daemon.misc.APOCException:
 com.sun.apoc.spi.environment.InvalidParameterException:
 The parameter organisation PROVIDER_URL#protocol (null) is not valid,
 the value must be comprised in {ldaps,ldap,https,http,file}.
    at com.sun.apoc.daemon.apocd.PolicyBackendFactory.openPolicyMgr(Unknown Source)
    ... 14 more
Caused by: com.sun.apoc.spi.environment.InvalidParameterException: The parameter
organisation PROVIDER_URL#protocol (null) is not valid, the value must be comprised in
{ldaps,ldap,https,http,file}.
    at com.sun.apoc.spi.PolicyMgrFactoryImpl.createPolicyMgr(Unknown Source)
    ... 15 more
Nov 18, 2005 1:59:23 PM PolicyBackend openPolicyBackend
```

ポリシーリポジトリに接続できない

Configuration Agent は、ポリシー情報をダウンロードまたはキャッシュに書き込むために有効な構成リポジトリに接続する必要があります。接続を確立できない場合、デスクトップクライアントアプリケーションの起動時に、次のようなエラーが Configuration Agent ログに記録されます。次のケースでは、ホスト sobuild が存在していない、接続できない、またはポート 389 経由で LDAP サーバーにアクセスできません。この問題を解決するには、Agent Configuration ウィザード (`/usr/bin/apoc-config`) を使用して、ポリシーリポジトリが正しく特定されていることを確認します。正しく特定されている場合は、ポリシーリポジトリへのアクセスが使用可能であることを確認します。たとえば、LDAP リポジトリの場合、LDAP サーバーが稼動していること、LDAP サーバーをホストしているマシンがネットワーク上で使用可能であること、および指定したポートが LDAP サーバーが使用しているポートであることを確認する必要があります。

SSL 接続を使用する LDAP サーバーに接続する場合は、Configuration Agent を実行するために必要な Java 実行環境に関連付けされたキーストア内で、適切な証明書が使用可能であることを確認します。`apoc-config` についての詳細については、18 ページの「Configuration Agent」 のセクションを参照してください。

```
FINER: New client added
Nov 18, 2005 2:17:43 PM com.sun.apoc.daemon.misc.APOCLogger finer
```

```
FINER: CreateSession transaction started
Nov 18, 2005 2:17:43 PM com.sun.apoc.daemon.misc.APOCLogger finer
FINER: Creating new client session
Nov 18, 2005 2:17:43 PM com.sun.apoc.daemon.misc.APOCLogger finest
FINEST: Authenticating user geoffh
Nov 18, 2005 2:17:43 PM com.sun.apoc.daemon.misc.APOCLogger finest
FINEST: Authentication successful
Nov 18, 2005 2:17:43 PM PolicyBackend openPolicyBackend
FINER: THROW
com.sun.apoc.daemon.misc.APOCException: com.sun.apoc.daemon.misc.APOCException:
com.sun.apoc.spi.OpenConnectionException: An error occured while connecting to
ldap://sobuild:389.
    at com.sun.apoc.daemon.apocd.PolicyBackend.<init>(Unknown Source)
    at com.sun.apoc.daemon.apocd.HostPolicyBackend.<init>(Unknown Source)
    at com.sun.apoc.daemon.apocd.PolicyBackendFactory.openPolicyBackend(Unknown Source)
    at com.sun.apoc.daemon.apocd.Cache$DataSource.openPolicyBackend(Unknown Source)
    at com.sun.apoc.daemon.apocd.Cache$DataSource.open(Unknown Source)
    at com.sun.apoc.daemon.apocd.Cache.createDataSources(Unknown Source)
    at com.sun.apoc.daemon.apocd.Cache.<init>(Unknown Source)
    at com.sun.apoc.daemon.apocd.CacheFactory.createNewCache(Unknown Source)
    at com.sun.apoc.daemon.apocd.CacheFactory.openCache(Unknown Source)
    at com.sun.apoc.daemon.apocd.Session.<init>(Unknown Source)
    at com.sun.apoc.daemon.transaction.CreateSessionTransaction.executeTransaction
(Unknown Source)
    at com.sun.apoc.daemon.transaction.Transaction.execute(Unknown Source)
    at com.sun.apoc.daemon.apocd.ClientEventHandler.handleEvent(Unknown Source)
    at com.sun.apoc.daemon.apocd.EventWorkerThread.run(Unknown Source)
Caused by: com.sun.apoc.daemon.misc.APOCException:
com.sun.apoc.spi.OpenConnectionException: An error occured while
connecting to ldap://sobuild:389. at
com.sun.apoc.daemon.apocd.PolicyBackendFactory.openPolicyMgr(Unknown Source)
    ... 14 more
Caused by: com.sun.apoc.spi.OpenConnectionException: An error occured while
connecting to ldap://noSuchHost:389.
    at com.sun.apoc.spi.ldap.LdapClientContext.prepareConnection(Unknown Source)
    at com.sun.apoc.spi.ldap.LdapClientContext.connect(Unknown Source)
    at com.sun.apoc.spi.ldap.LdapConnectionHandler.openAuthorizedContext(Unknown Source)
    at com.sun.apoc.spi.ldap.LdapConnectionHandler.connect(Unknown Source)
    at com.sun.apoc.spi.ldap.entities.LdapOrganizationProvider.open(Unknown Source)
    at com.sun.apoc.spi.PolicyMgrFactoryImpl.createPolicyMgr(Unknown Source)
    ... 15 more
Caused by: netscape.ldap.LDAPException: failed to connect to server sobuild:389 (91);
Cannot connect to the LDAP server
    at netscape.ldap.LDAPConnSetupMgr.connectServer(LDAPConnSetupMgr.java:422)
    at netscape.ldap.LDAPConnSetupMgr.openSerial(LDAPConnSetupMgr.java:350)
    at netscape.ldap.LDAPConnSetupMgr.connect(LDAPConnSetupMgr.java:244)
    at netscape.ldap.LDAPConnSetupMgr.access$0(LDAPConnSetupMgr.java:241)
    at netscape.ldap.LDAPConnSetupMgr$1.run(LDAPConnSetupMgr.java:179)
```

```
    at java.lang.Thread.run(Thread.java:595)
Nov 18, 2005 2:17:44 PM PolicyBackend openPolicyBackend
```

構成されていないポリシーリポジトリへの接続

Configuration Agent がポリシーリポジトリ内のポリシーデータを検索できるように、ポリシーリポジトリを正しく構成する必要があります。構成されていない、または間違って構成されたポリシーリポジトリを指定すると、デスクトップクライアントアプリケーションの起動時に、次のようなエラーが Configuration Agent ログに記録されます。

```
FINER: New client added
Nov 18, 2005 2:36:55 PM com.sun.apoc.daemon.misc.APOCLogger finer
FINER: CreateSession transaction started
Nov 18, 2005 2:36:55 PM com.sun.apoc.daemon.misc.APOCLogger finer
FINER: Creating new client session
Nov 18, 2005 2:36:55 PM com.sun.apoc.daemon.misc.APOCLogger finest
FINEST: Authenticating user geoffh
Nov 18, 2005 2:36:55 PM com.sun.apoc.daemon.misc.APOCLogger finest
FINEST: Authentication successful
Nov 18, 2005 2:36:55 PM PolicyBackend openPolicyBackend
FINER: THROW
com.sun.apoc.daemon.misc.APOCException: com.sun.apoc.daemon.misc.APOCException:
com.sun.apoc.spi.environment.RemoteEnvironmentException: Error on reading the
configuration data on LDAP server ldap://sobuild:389.
    at com.sun.apoc.daemon.apocd.PolicyBackend.<init>(Unknown Source)
    at com.sun.apoc.daemon.apocd.HostPolicyBackend.<init>(Unknown Source)
    at com.sun.apoc.daemon.apocd.PolicyBackendFactory.openPolicyBackend(Unknown Source)
    at com.sun.apoc.daemon.apocd.Cache$DataSource.openPolicyBackend(Unknown Source)
    at com.sun.apoc.daemon.apocd.Cache$DataSource.open(Unknown Source)
    at com.sun.apoc.daemon.apocd.Cache.createDataSources(Unknown Source)
    at com.sun.apoc.daemon.apocd.Cache.<init>(Unknown Source)
    at com.sun.apoc.daemon.apocd.CacheFactory.createNewCache(Unknown Source)
    at com.sun.apoc.daemon.apocd.CacheFactory.openCache(Unknown Source)
    at com.sun.apoc.daemon.apocd.Session.<init>(Unknown Source)
    at com.sun.apoc.daemon.transaction.CreateSessionTransaction.executeTransaction
(Unknown Source)
    at com.sun.apoc.daemon.transaction.Transaction.execute(Unknown Source)
    at com.sun.apoc.daemon.apocd.ClientEventHandler.handleEvent(Unknown Source)
    at com.sun.apoc.daemon.apocd.EventWorkerThread.run(Unknown Source)
```

「クライアントの最大接続数」に関する **Configuration Agent** ログに見られるメッセージは何を意味しているのですか

Configuration Agent によって有効にされる各デスクトップクライアントアプリケーション (gconfd、Mozilla、StarSuite) は、起動時に Configuration Agent へ接続します。この接続制限は、エージェントの構成で指定されます。デフォルトの接続制限は、50 です。複数ユーザーのマシンでは、Configuration Agent ウィザード (`/usr/bin/apoc-config`) で「クラ

イアントの最大接続数」設定を変更して、この制限を増やす必要がある場合があります。Configuration Agent が最大接続数に達すると、次のようなエラーメッセージが Configuration Agent ログに記録されます。

```
Nov 18, 2005 3:20:55 PM com.sun.apoc.daemon.misc.APOCLogger warning
WARNING: The maximum number of client connections ( 50 ) has been reached.
No new client connections can be established at this time.
```

Desktop Manager を使用していくつかのポリシーを変更したが、変更内容が使用しているクライアントマシンに表れない

Configuration Agent は、Desktop Manager が作成したポリシーデータは比較的静的であり、頻繁に変更されることはないという仮定の基に設計されました。この仮定により、変更が発生したかどうかを確認するために、エージェントが断続的にポリシーリポジトリを調べるという手法が採られました。デフォルトで、エージェントは、実行中のすべてのデスクトップアプリケーションのリポジトリを、1 時間に 1 度調べます。結果として、Desktop Manager で変更を行なったときは、実行中のデスクトップアプリケーションに変更が通知されるまで、最大で 1 時間待つ必要があります。必要であれば、リポジトリチェックの頻度を上げるために、Agent Configuration ウィザード (`/usr/bin/apoc-config`) を使用して、「一般的な検出間隔」の値を変更できます。あるいは、スーパーユーザーになって、`/usr/lib/apoc/apocd change-detect` コマンドを実行することにより、Configuration Agent が接続されたすべてのアプリケーションのポリシーデータを強制的にリフレッシュするようにできます。

◆ ◆ ◆ 第 4 章

Java Web Console

Java Web Console は、Sun Microsystems 製品用に共通の Web ベースのシステム管理ソリューションを提供する目的で設計されました。これは、ユーザーがシステム管理アプリケーションにアクセス可能な単一の場所として役立ちます。これらのシステム管理アプリケーションは、そのすべてが一貫したユーザーインタフェースを提供します。

このコンソールは、さまざまな理由から Web モデルに基づいています。しかし、その第一の理由はシステム管理者が Web ブラウザを使ってシステム管理アプリケーションにアクセスできるようにすることです。

Java Web Console が提供する機能は次のとおりです。

- 共通の認証と承認
- 共通のログ
- すべてのシステム管理アプリケーションに対して、同じ HTTPS ベースのポートを経由する単一のエントリポイント
- 共通の外観と使用感

このコンソールの利点は、管理者が 1 回ログインしたあと、コンソール内でどのアプリケーションでも使用可能であることです。

インストール

システム要件

Java Web Console は、複数のクライアントサーバー型オペレーティングシステムといくつかのブラウザをサポートします。

クライアント

- Solaris 10 で動作する Netscape™ 6.2x および 7.x
- Windows 98、98 SE、ME、2000、および XP で動作する Netscape 6.2x および 7.x
- Windows 98、98 SE、ME、2000、および XP で動作する Internet Explorer 5.5x および 6.x
- Solaris で動作する Mozilla 1.4x
- Solaris で動作する Firefox 1.0

サーバー

- Solaris 10
- Red Hat Application Server 2.1 および 3.0
- SuSE Linux 8.0 以降
- J2SE バージョン 1.4.1_03 以降

  サーバー上で J2SE 1.4.1 以前のバージョンが検出された場合、セットアッププログラムにより Java Desktop System Management Tools CD に収録されている J2SE バージョンを使ってインストールをアップグレードするように求められます。
- Tomcat: 4.0.3 以降

  Tomcat は Java Desktop System Management Tools CD に収録されています。

Java Web Console のインストール

Java Web Console 2.2.4 は、Solaris 10 オペレーティングシステムに含まれていますが、Desktop Manager にはバージョン 2.2.5 が必要になります。バージョン 2.2.5 のコピーは、`server/console` ディレクトリ内の Desktop Manager アーカイブにあります。そのディレクトリ内で `./setup` を実行するとインストールできます。

Java Web Console 3.0 がインストールされている場合は、バージョン 3.0 をアンインストールして、先に述べた `server/console` ディレクトリから Java Web Console 2.2.5 をインストールする必要があります。

コンソールの実行

通常、Java Web Console サーバーを停止して再起動するだけで、新しいアプリケーションを登録できます。

注意 – 初めて Java Web Console を起動する前に、Desktop Manager のインストールが完了していることを確認してください。Java Web Console は、コンソールに最低 1 つのアプリケーションを配備するまでは正常に実行されません。

- Java Web Console を起動するには、`smcwebserver start` を入力します。
- Java Web Console を停止するには、`smcwebserver stop` を入力します。
- Java Web Console を再起動するには、`smcwebserver restart` を入力します。
- Java Web Console にアクセスするには、ブラウザに次の URL を入力します。
  `https://`*<hostname>*`.`*<domainname>*`:6789`

Java Web Console は簡単に導入でき、Unix ベースの認証と役割ベースのアクセス制御 (RBAC) をサポートします。ただし、LDAP 認証など、ほかの認証機構も構成できます。

注 – デフォルトのセッションタイムアウトは 15 分です。`smreg` コマンドを使用すれば、このタイムアウトを構成できます。たとえば、このタイムアウトを 5 分に設定するには、`smreg add -p -c session.timeout.value=5` を入力します。

Java Web Console のコマンドの詳細については、smcwebserver と smreg のマニュアルページを参照してください。

Java Web Console の削除

注意 – Java Web Console はオペレーティングシステムの一部であるため、Solaris を使用している間は削除できません。

Java Web Console のトラブルシューティング

Java Web Console をインストールできません

症状: インストールの最後に、登録されたアプリケーションがないため Java Web Console が起動できないことを示すメッセージが表示されます。

考えられる原因: Desktop Manager モジュールはインストールされると、Java Web Console を起動します。

接続が拒否されました

症状: `https://<your.server>:6789` などの適切な URL を開こうとしたが、接続が拒否されました。

考えられる原因: サーバー上で Java Web Console が実行されていません。

ログインできません

注 – LDAP ログインモジュールは、デフォルトではインストールされません。その結果、ログイン資格が LDAP サーバー内に格納されたログイン資格と比較されず、通常のシステムログイン資格のみが要求されます。このトラブルシューティングのセクションは、手動で LDAP ログインモジュールをインストールした場合にのみ適用されます。

症状: Web Console のログインページが開かれましたが、ユーザー名とパスワードの組み合わせが拒否されます。

考えられる原因:

- LDAP サーバーが稼動していません。
- Web Console LDAP 認証モジュールが正しく構成されていません。
- ユーザーが LDAP サーバー上に存在しません。
- ユーザーのパスワードが、LDAP サーバーのパスワードと異なっています。

Desktop Manager のリンクがありません

症状: Web Console にログインしましたが、アプリケーションリストのページに Desktop Manager が含まれていません。

考えられる原因:

- Desktop Manager モジュールがインストールされていません。

Null ポインタ例外、**Tomcat/Java** エラー、または空白

症状: Desktop Manager を開きましたが、値が表示されず、空白ページまたはエラーが表示されます。

考えられる原因: エラーが `NoClassDefFoundError:sun/tools/javac/Main` と表示される場合は、Java Web Console が間違った Java インストールを使用しています。

その他の問題

Web サーバーが正常に稼動しない場合、ログファイルで情報が提供される場合があります。ログファイルは、/var/log/webconsole/ に置かれています。次のように smreg を使用して、ログの詳細レベルを上げることができます。

```
smreg add -p debug.trace.level=3
smreg add -p debug.trace.options=tmp
```

次のコマンドを実行すると、元の設定を復元することができます。

```
smreg add -p debug.trace.level=0
smreg add -p debug.trace.options=m
```

次のコマンドを実行すると、構成データベースのフルダンプが実行されます。

```
smreg list
```

ポートを使用したままで、Desktop Manager をホストする Web サーバーが正しく停止しないことがあります。これによって、新しく起動された Web サーバーがまったく起動しなくなります。smcwebserver start/restart コマンドでエラーメッセージが表示される場合、smcwebserver stop を実行したあとでもまだ Desktop Manager にアクセス可能な場合、または新しく起動したサーバーがまだ古いインスタンスのように動作する場合、ポート 6789 がまだ使用中であるかどうかを調査 (netstat -a | grep 6789) するか、または Web サーバーがまだ稼動しているかどうかを調査 (ps -ef | grep java) してください。いずれかが当てはまる場合、一致するプロセスを終了して、ポート 6789 が使用されていないようにする必要があります。

◆ ◆ ◆ 付 録 A

構成パラメータ

ここでのパラメータは、次の Desktop Manager コンポーネントに対して指定できます。

- `/etc/opt/SUNWapcmg/` に置かれた構成リポジトリを定義するファイルにある Desktop Manager。
- `/etc/apoc/policymgr.properties` ファイルにある Configuration Agent。
- `$HOME/pgtool.properties` ファイルにある Desktop Manager CLI。CLI は純粋な LDAP リポジトリのみをサポートするという制限があります。

パラメータには、適用するリポジトリプロバイダを示す接頭辞を付けることができます。プロバイダごとに、接頭辞が付けられたパラメータが最初に考慮されます。このようなパラメータが指定されていない場合は、接頭辞の付いていないパラメータが使用されます。

表 A–1 接頭辞

接頭辞の値	リポジトリプロバイダ
ORGANIZATION_	組織ツリー
DOMAIN_	ドメインツリー
PROFILE_	プロファイル
ASSIGNMENT_	割り当て
LDAP_META_CONF_	LDAP リポジトリの場合のマッピングデータ

表 A–2 パラメータ

名前	説明	可能な値	デフォルト値

PROVIDER_URL	リポジトリへの接続を指定する URL。URL のリストを使用して、最初の URL への接続が成功しなかった場合の代替リポジトリを指定することができます。	空白文字で区切られた 1 つ以上の URL のリスト。各 URL は、次の形式のいずれか 1 つになります。
		`ldap://<host>:<port>/<baseDN>`
		`ldaps://<host>:<port>/<baseDN>` `file://<filepath>`
		`http://<host>:<port>/<filepath>`
		`https://<host>:<port>/<filepath>`
			なし、必須パラメータ

SECURITY_PRINCIPAL	リポジトリに接続するためのユーザー名。	リポジトリの読み取り権または検索アクセス権を持つユーザーの名前。匿名による接続の場合は値なし。	なし、匿名接続
SECURITY_CREDENTIALS	SECURITY_PRINCIPAL で指定されたユーザーのパスワード。	スクランブルがかけられた、または明確なテキストによるパスワード。	なし
SECURITY_CREDENTIALS_ENCRYPTED	SECURITY_PRINCIPAL で指定されたパスワードにスクランブルがかけられているかどうかを示します。警告: パスワードのスクランブルは、パスワードにマスクをかけるだけであり、どのような安全な暗号化も行われません。	「scramble」。パスワードにスクランブルがかけられている場合 (構成データの生成時にウィザードによって自動的にかけられます)。 「none」。パスワードが明確なテキストで表示される場合。パスワードを編集する場合はこの値を使用します。	「none」
MAX_SEARCH_RESULT	リポジトリ内での検索により得られた結果の最大数。注意: 接頭辞のスキーマは、このパラメータには適用されません。	正の数、0 は制限なし。	100

次のパラメータは、LDAP リポジトリにのみ適用されます。

表 A–3 LDAP固有のパラメータ

名前	説明	可能な値	デフォルト値
AuthDn	SECURITY_PRINCIPALで指定されたユーザーを取得するため、LDAPリポジトリに最初にアクセスするために使用されるユーザーの完全修飾DN。	リポジトリの読み取り権または検索アクセス権を持つユーザーの名前。匿名による接続の場合は値なし。	なし、匿名アクセス
Password	AuthDNのパスワード。	スクランブルがかけられた、または明確なテキストパスワード。	なし
Password_ENCODING	Passwordで指定されたパスワードにスクランブルがかけられているかどうかを示します。警告: パスワードのスクランブルは、パスワードにマスクをかけるだけであり、どのような安全な暗号化も行われません。	「scramble」。パスワードにスクランブルがかけられている場合(構成データの生成時にウィザードによって自動的にかけられます)。「none」。パスワードが明確なテキストで表示される場合。パスワードを編集する場合はこの値を使用します。	「none」
Connect Timeout	接続作成のタイムアウト(秒単位)。	正の数、0は時間無制限。	1

例 A–1 ハイブリッドバックエンドの例

プロファイルとその割り当てがファイルシステムに格納される一方で、ホストとユーザーに関する情報が既存のLDAPリポジトリから取得される、ハイブリットバックエンドの例。

```
#Organization, Domain, MetaConf
PROVIDER_URL = ldap://server1.sun.com:389/o=apoc ldap://server2.sun.com:389/o=apoc
SECURITY_PRINCIPAL = jmonroe
SECURITY_CREDENTIALS = JmonroE
SECURITY_CREDENTIALS_ENCODING = none
AuthDn = cn=reader,ou=special users,o=apoc
Password = lakjflajf
Password_ENCODING = scramble
ConnectTimeout = 5

#Profile
PROFILE_PROVIDER_URL = file:///path/to/repository
```

例 A–1 ハイブリッドバックエンドの例　(続き)

```
#Assignment
ASSIGNMENT_PROVIDER_URL = file:///path/to/repository
```

◆ ◆ ◆ 付 録 B

Desktop Manager による OpenLDAP と Active Directory の使用

Desktop Manager による OpenLDAP サーバーの使用

OpenLDAP サーバーを Desktop Manager データのリポジトリとして使用するには、構成データを格納するために使用するオブジェクトクラスと属性を提供するようにサーバーのスキーマを拡張する必要があります。`/usr/share/webconsole/apoc/deploy` ディレクトリには、`apoc.schema` という名前のカスタムスキーマファイルがあります。

このファイルは、OpenLDAP 構成ディレクトリ (`/etc/openldap`) の `schema` サブディレクトリ内にコピーし、そのディレクトリにある `slapd.conf` ファイルに含めることにより、OpenLDAP に追加する必要があります。このためには、ファイル内に存在する一連のスキーマの include 行の末尾に `include /etc/openldap/schema/apoc.schema` という行を挿入します。OpenLDAP サーバーのスキーマの拡張について詳細は、サーバーのマニュアルを参照してください。

OpenLDAP サーバーのスキーマを拡張した場合、Desktop Manager の Add Configuration Repository ウィザードを使用して、残りの構成を完了することができます。

注 – Desktop Manager エージェントは、データが必要なユーザーの DN だけを提供して、パスワードを提供せずに、OpenLDAP サーバーに匿名で接続しようとします。OpenLDAP サーバーの一部のリリースでは、デフォルトでこのモードの匿名認証が無効になる場合があります。この場合、OpenLDAP 構成ディレクトリ (`/etc/openldap`) 内のファイル `slapd.conf` に定義される共通のサーバーパラメータに `allow bind_anon_cred` という行を追加することにより、このモードを有効にする必要があります。パラメータの詳細については、サーバーのマニュアルを参照してください。

Desktop Manager による Active Directory サーバーの使用

Active Directory サーバーを Desktop Manager データのリポジトリとして使用するには、構成データを格納するために使用するオブジェクトクラスと属性を提供するようにサーバーのスキーマを拡張する必要があります。 `/usr/share/webconsole/apoc/deploy` ディレクトリには、`apoc-ad.ldf` という名前のスキーマ拡張ファイルがあります。

次の手順に従って、`apoc-ad.ldf` ファイルを Active Directory スキーマにインポートする必要があります。

1. スキーマの拡張を有効にします。操作方法についての詳細は、Active Directory のマニュアルを参照してください。
2. コマンドプロンプトから**`ldifde -i -c "DC=Sun,DC=COM" <`*BaseDN*`> -f apoc-ad-registry.ldf`** を実行します。

注 – <*BaseDN*> は Active Directory のベース DN で置き換えます。

Active Directory サーバーのスキーマを拡張した場合、Desktop Manager の Add Configuration Repository ウィザードを使用して、残りの構成を完了することができます。

Add Configuration Repository ウィザードで LDAP 資格の入力を求められたら、ツリーへの読み取り権を持つユーザーの完全 DN とパスワードを入力します。このユーザーとして、ほかの目的で Active Directory を使用できないユーザーを選択できます。このようなユーザーを設定する方法についての詳細情報は、Active Directory のマニュアルを参照してください。さらに、Active Directory のドメイン名は Desktop Manager が動作しているマシンによって認識されている必要があります。このためには、Active Directory サーバーの IP アドレスをそのドメイン名にマッピングする行をコンピュータの `/etc/hosts` ファイルに追加します。

デスクトップのホストから構成データを取得するには、そのホストでも Active Directory のドメイン名が認識されていることが必要です。デスクトップユーザーの認証は、匿名および GSSAPI の 2 通りの方法で行うことができます。

- 匿名接続を使用して認証するには、すべてのユーザーに読み取り権を許可するように Active Directory サーバーを構成する必要があります。操作方法について詳細は、Active Directory のマニュアルを参照してください。
- GSSAPI を使用して認証するには、ユーザーは Active Directory に対して認証され、ユーザーの資格がシステム上で使用可能である必要があります。これは、システムの Kerberos 認証を構成し、ログイン時にこれらの資格を生成することで達成できます。これを行う方法については、使用するシステムの管理マニュアルを参照してください。

◆ ◆ ◆ 付 録 C

組織のマッピング

組織のマッピング

LDAP エントリと Desktop Manager 要素間のマッピングを指定するには、`Organization` マッピングファイルを編集する必要があります。さまざまなキーに、LDAP リポジトリのレイアウトに一致する値を指定する必要があります。

ユーザー要素は、すべての要素で使用されるオブジェクトクラスと、リポジトリ全体で固有の値を持つ必要がある属性によって識別されます。管理アプリケーション内でのユーザーの表示方法に影響する名前の形式を指定できます。また、組織内のユーザーエントリでコンテナエントリを使用する場合に、オプションでそのコンテナエントリを指定できます。次に、キーの名前とそのデフォルト値を示します。

```
# すべてのユーザーエントリが使用するオブジェクトクラス
User/ObjectClass=inetorgperson
# ユーザーエントリの値がリポジトリ内で一意である属性
User/UniqueIdAttribute=uid
# ユーザーエントリの組織エントリ内にあるオプションのコンテナ
# 使用しない場合は、この行を削除すること
User/Container=ou=People
# 管理アプリケーションにおける表示名の形式
User/DisplayNameFormat=sn, givenname
```

役割の要素は、それが使用する可能性のあるオブジェクトクラスのリストと、対応する名前属性によって識別されます。これらのリストは `<item1>,<item2>,...,<itemN>` の形式を使用して、整列する必要があります。つまり、リストでは命名属性と同数の項目を含め、n 番目の命名属性で n 番目のオブジェクトクラスを使用する必要があります。2 つのキーにより、役割とユーザーの関係、および役割とホストの関係が指定されます。*VirtualMemberAttribute* キーで指定する属性の値には、ユーザーまたはホストのエントリからクエリーできることが必要です。またこのキーには、エントリが属する役割の完全な DN を含める必要があります。*MemberAttribute* キーには、検索フィルタに適用するユーザーまたはホストのエントリの属性を指定する必要があります。またこのキーには、ユーザーまたはホストが属する役割の完全な DN を含める必要があります。*VirtualMemberAttribute* キーには Class Of Service 仮想属性を指定できるのに対

し、*MemberAttribute* キーにはフィルタ内で使用可能な物理属性を指定する必要があります。次に、キーの名前とそのデフォルト値を示します。

```
# 役割のオブジェクトクラスのリスト
Role/ObjectClass=nsRoleDefinition
# 対応する命名属性の整列済みリスト
Role/NamingAttribute=cn
# ユーザーまたはホストの役割 DN を含む
# 物理的な属性 (フィルタで使用可能)
Role/MemberAttribute=nsRoleDN
# ユーザーまたはホストへのクエリーによって
# 自分が属する役割の IN が返される属性
Role/VirtualMemberAttribute=nsRole
```

組織の要素は、オブジェクトクラスから成る 2 つの整列済みリストと対応する命名属性により、役割に似た方法で識別されます。次に、キーの名前とそのデフォルト値を示します。

```
# 組織のオブジェクトクラスのリスト
Organization/ObjectClass=organization
# 対応する命名属性の整列済みリスト
Organization/NamingAttribute=o
```

ドメインの要素は、組織の要素と同様の方法で識別されます。次に、キーの名前とそのデフォルト値を示します。

```
# ドメインのオブジェクトクラスのリスト
Domain/ObjectClass=ipNetwork
# 対応する命名属性の整列済みリスト
Domain/NamingAttribute=cn
```

ホストの要素は、ユーザーの要素と同様の方法で識別されます。次に、キーの名前とそのデフォルト値を示します。

```
# すべてのホストエントリが使用するオブジェクトクラス
Host/ObjectClass=ipHost
# ホストエントリの値がリポジトリ内で一意である属性
Host/UniqueIdAttribute=cn
# ホストエントリのドメインエントリ内にあるオプションのコンテナ
# 使用しない場合は、この行を削除すること
Host/Container=ou=Hosts
```