IBM Security Directory Integrator
バージョン 7.2.0.1

インストールおよび管理者ガイド

SC88-8415-03
(英文原典：SC27-2705-03)

IBM Security Directory Integrator
バージョン 7.2.0.1

インストールおよび管理者ガイド

SC88-8415-03
(英文原典：SC27-2705-03)

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、 431 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。

注: 本書は、*IBM Security Directory Integrator* バージョン 7.2.0.1 ライセンス・プログラム (5724-K74)、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： SC27-2705-03
IBM Security Directory Integrator
Version 7.2.0.1
Installation and Administrator Guide

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

この資料には、IBM® Security Directory Integrator に含まれるコンポーネントを使用したソリューション開発に必要な情報が記載されています。

IBM Security Directory Integrator の各コンポーネントは、ユーザー・ディレクトリーおよびその他のリソースの管理を担当するネットワーク管理者用に設計されています。IBM Security Directory Integrator と IBM Security Directory Server の両方のインストールおよび使用に関する実務経験を持っていることを想定しています。

本書は、IBM Security Directory Integrator を使用したソリューションの開発、インストール、および管理を担当するユーザーも対象としています。読者は、開発したソリューションの接続先となるシステムの概念や管理方法について習熟している必要があります。そのようなシステムには、ソリューションに応じて、以下の製品、システム、概念のうち 1 つ以上が含まれます (これらに限定されてはいません)。

- IBM Security Directory Server
- IBM Security Identity Manager
- IBM Java™ ランタイム環境 (JRE) または Oracle Java ランタイム環境
- Microsoft Active Directory
- Windows および UNIX オペレーティング・システム
- セキュリティー管理
- Hypertext Transfer Protocol (HTTP)、HyperText Transfer Protocol Secure (HTTPS)、および Transmission Control Protocol/Internet Protocol (TCP/IP) を含むインターネット・プロトコル (IP)
- Lightweight Directory Access Protocol (LDAP) およびディレクトリー・サービス
- サポートされるユーザー・レジストリー
- 認証と許可の概念
- SAP ABAP アプリケーション・サーバー

資料および用語集へのアクセス

以下は英語のみの対応となります。オンラインでアクセス可能な IBM Security Directory Integrator バージョン 7.2.0.1 ライブラリーおよび関連資料の説明をお読みください。

このセクションには、以下が含まれています。

- 『IBM Security Directory Integratorライブラリー』にある資料のリスト。
- xi ページの『オンライン資料』へのリンク。
- xi ページの『IBM Terminology Web サイト』へのリンク

IBM Security Directory Integratorライブラリー

IBM Security Directory Integrator ライブラリーでは以下の資料を入手できます。

- *IBM Security Directory Integrator バージョン 7.2.0.1 Federated Directory Server 管理ガイド*

  Federated Directory Server コンソールを使用して、データ統合ソリューションを設計、実装、および管理する方法が記載されています。また、ID 管理のために System for Cross-Domain Identity Management (SCIM) プロトコルおよびインターフェースを使用する方法も記載されています。
- *IBM Security Directory Integrator バージョン 7.2.0.1 スタートアップ・ガイド*

  IBM Security Directory Integrator の解説および概要です。対話の作成の例と、IBM Security Directory Integrator の実践学習を含んでいます。
- *IBM Security Directory Integrator バージョン 7.2.0.1 ユーザーズ・ガイド*

  IBM Security Directory Integrator の使用方法が記載されています。Security Directory Integrator デザイナー・ツール (構成エディター) を使用したソリューションの設計や、コマンド行からの既製ソリューションの実行について説明しています。また、インターフェース、概念、および AssemblyLine の作成に関する情報も記載されています。
- *IBM Security Directory Integrator バージョン 7.2.0.1 インストールおよび管理者ガイド*

  インストール、旧バージョンからのマイグレーション、ロギング機能の構成、および IBM Security Directory Integrator のリモート・サーバー API の基礎となるセキュリティー・モデルについて記載されています。ソリューションのデプロイおよび管理方法が含まれています。
- *IBM Security Directory Integrator バージョン 7.2.0.1 リファレンス・ガイド*

  IBM Security Directory Integrator の個々のコンポーネント (コネクター、関数コンポーネント、パーサー、オブジェクトなど) に関する詳細情報が記載されています。
- *IBM Security Directory Integrator バージョン 7.2.0.1 Problem Determination Guide*

  問題の識別および解決を支援する IBM Security Directory Integrator のツール、リソース、および技法に関する情報が記載されています。
- *IBM Security Directory Integrator バージョン 7.2.0.1 メッセージ・ガイド*

  IBM Security Directory Integrator に関連付けられたすべての情報、警告、およびエラー・メッセージのリストが記載されています。
- *IBM Security Directory Integrator バージョン 7.2.0.1 パスワード同期プラグイン*

  5 つの IBM Password Synchronization Plug-ins (Windows 用 Password Synchronizer、Sun Directory Server 用 Password Synchronizer、IBM Security Directory Server 用 Password Synchronizer、Domino® 用 Password Synchronizer、UNIX および Linux 用 Password Synchronizer) それぞれのインストールおよび構成について詳細に説明されています。また、LDAP パスワード・ストアと JMS パスワード・ストアの構成手順についても説明します。
- *IBM Security Directory Integrator バージョン 7.2.0.1 リリース・ノート*

資料に記載されていない IBM Security Directory Integrator の新機能および最新情報を記載しています。

オンライン資料

IBM では、製品のリリース時および資料の更新時に、以下の場所に製品資料を掲載しています。

IBM Security Directory Integrator ライブラリー
製品資料サイト (http://www-01.ibm.com/support/knowledgecenter/SSCQGF/welcome) には、ライブラリーのウェルカム・ページとナビゲーションが表示されます。

IBM Security Systems Documentation Central
IBM Security Systems Documentation Central には、すべての IBM Security Systems 製品ライブラリーのアルファベット順のリストと、各製品のそれぞれのバージョンのオンライン資料へのリンクが掲載されています。

IBM Publications Center
IBM Publications Center サイト (http://www-05.ibm.com/e-business/linkweb/publications/servlet/pbi.wss) には、必要なすべての IBM 資料を見つけるのに役立つカスタマイズ検索機能が用意されています。

関連情報

IBM Security Directory Integrator の関連情報は以下の場所で入手できます。

- IBM Security Directory Integrator では、Oracle の JNDI クライアントを使用しています。JNDI クライアントについては、「*Java Naming and Directory Interface™ Specification*」(http://download.oracle.com/javase/7/docs/technotes/guides/jndi/index.html) を参照してください。
- IBM Security Directory Integrator に関する疑問点を解決するために有用な情報が https://www-947.ibm.com/support/entry/myportal/over-accesspubsview/software/security_systems/tivoli_directory_integratorに記載されています。

IBM Terminology Web サイト

IBM Terminology Web サイトは、製品ライブラリーの用語を 1 つの場所にまとめたものです。Terminology Web サイトには、http://www.ibm.com/software/globalization/terminology からアクセスできます。

アクセシビリティー

アクセシビリティー機能は、運動障害または視覚障害など身体に障害を持つユーザーがソフトウェア・プロダクトを快適に使用できるようにサポートします。この製品では、インターフェースの読み上げおよびナビゲートを行うための支援技術を使用できます。また、マウスの代わりにキーボードを使用して、グラフィカル・ユーザー・インターフェースのすべての機能を操作できます。

詳しくは、「*Directory Integrator の構成*」のアクセシビリティーに関する付録を参照してください。

技術研修

以下は英語のみの対応となります。技術研修の情報については、IBM Education Web サイト (http://www.ibm.com/software/tivoli/education) を参照してください。

サポート情報

IBM サポートは、コード関連の問題、およびインストールまたは使用方法に関する短時間の定型質問に対する支援を提供します。IBM ソフトウェア・サポート・サイトには、http://www.ibm.com/software/support/probsub.html から直接アクセスできます。

トラブルシューティング では、以下が詳細に説明されています。

- IBM サポートに連絡する前に収集する情報。
- IBM サポートに連絡するためのさまざまな方法。
- IBM Support Assistant の使用方法。
- 問題を自分自身で特定して解決するための手順と問題判別のためのリソース。

適切なセキュリティーの実践に関する注意事項

IT システムのセキュリティーでは、企業内および企業外からの不適切なアクセスの防止、検出、およびそれらのアクセスへの対応により、システムおよび情報を保護する必要があります。不適切なアクセスにより、情報が改ざん、破壊、盗用、または悪用されたり、ご使用のシステムの損傷または他のシステムへの攻撃のための利用を含む悪用につながる可能性があります。完全に安全と見なすことができる IT システムまたは IT 製品は存在せず、また単一の製品、サービス、またはセキュリティー対策が、不適切な使用またはアクセスを防止する上で、完全に有効となることもありません。 IBM のシステム、製品およびサービスは、包括的なセキュリティーの取り組みの一部となるように設計されており、これらには必ず追加の運用手順が伴います。また、最高の効果を得るために、他のシステム、製品、またはサービスを必要とする場合があります。IBM は、システム、製品、またはサービスが、悪意のある行為または不正な行為から影響を受けないこと、またはこれらの行為がお客様の企業に影響を与えないことを保証しません。

第 1 章 概要

インストールおよび管理のタスクを開始する前に、IBM Security Directory Integrator の一般的な概念についてお読みください。

IBM Security Directory Integrator の一般的な概念の概要については、「*Directory Integrator の構成*」の『IBM Security Directory Integrator の概念』を参照してください。

IBM Security Directory Integrator の概念に関するさらに詳細な情報は、「リファレンス」を参照してください。

IBM Security Directory Integrator エディション

ここでは、製品のさまざまなエディションについて説明します。

IBM Security Directory Integrator バージョン 7.2 には 2 つの異なるエディションがあります (それぞれに異なるご使用条件が適用されます)。

- 汎用エディション: このエディションのライセンス交付は、プロセッサーごとに行われます。
- ID エディション: ライセンス交付はユーザーごとに行われます。

旧バージョンの IBM Security Directory Integrator とは異なり、バージョン 7.2 では、汎用エディションも ID エディションも内容、機能、および能力については同じです。ご使用条件のみが異なります。

第 2 章 IBM Security Directory Integrator のインストールの説明

IBM Security Directory Integrator のインストールは、要件の事前確認、ソフトウェアのインストールおよびソフトウェアを最終的に機能させるためのいくつかのタスクの実行で構成されます。

インストールの前に

IBM Security Directory Integrator のインストールを開始する前に、ご使用のシステムが最小要件を満たしているようにする必要があります。

IBM Security Directory Integrator インストーラーでは InstallAnywhere 2012 SP1 テクノロジーを使用します。

ディスク・スペース要件

以下のリンクを参照して、ディスク・スペースの要件を確認してください。

IBM Security Directory Integrator 資料のソフトウェア要件を参照してください。

メモリー要件

以下のリンクを参照して、メモリーの要件を確認してください。

IBM Security Directory Integrator 資料のソフトウェア要件を参照してください。

プラットフォーム要件

以下のリンクを参照して、プラットフォームの要件を確認してください。

IBM Security Directory Integrator 資料のソフトウェア要件を参照してください。

IBM Security Directory Integrator のコンポーネント

使用可能なコンポーネントについては、以下の情報を参照してください。

いくつかの例外はありますが、IBM Security Directory Integrator の一部としてインストールするために、以下のコンポーネントが選択可能です。

ランタイム・サーバー

IBM Security Directory Integrator 統合ソリューションをデプロイおよび実行するために使用されるルール・エンジン。

構成エディター

IBM Security Directory Integrator 統合ソリューションを作成、デバッグ、および拡張するための開発環境。

注: IBM Security Directory Integrator は、以下のオペレーティング・システムでは構成エディター (CE) をサポートしません。

- Linux PPC
- Linux 390
- AIX® PPC 64

ローカルの構成エディターを使用せずにソリューションを開発する方法について詳しくは、163 ページの『リモート構成エディターの使用』を参照してください。

構成エディター更新サイト (CE の Eclipse 更新サイト)

CE 更新サイト・フォルダーを使用して、既存の Eclipse インストール済み環境に IBM Security Directory Integrator 構成エディターをインストールします。Eclipse ソフトウェア更新ツールを使用し、このフォルダーをローカル更新サイトとして使用します。CE 更新サイトがサポートされるのは、Eclipse 3.5.1 以降へのデプロイメントのみです。

注: IBM Security Directory Integrator は、以下のオペレーティング・システムでは構成エディター更新サイトをサポートしません。

- Linux PPC
- Linux 390
- AIX PPC 64

ローカルの構成エディターを使用せずにソリューションを開発する方法について詳しくは、163 ページの『リモート構成エディターの使用』を参照してください。

Java API 文書

IBM Security Directory Integrator 内部の完全な HTML 資料。 カスタム・コンポーネントの開発や、ソリューションにおけるスクリプト記述のために不可欠な参照資料。

例　特定の IBM Security Directory Integrator の機能またはコンポーネントを中心に説明する、一連の具体的な短い構成の例。

ヘルプ・システム (ローカルなホスト IBM Security Directory Integrator ヘルプ。デフォルトはオンラインです。)

注: この機能は推奨されません。IBM Security Directory Integrator の将来のバージョンでは削除されます。

グローバル・オンライン・ヘルプ・サービスを使用する代わりに、ローカルに Eclipse を採用した IBM IBM ユーザー・インターフェース・ヘルプ・システム (以前の IBM Eclipse ヘルプ・システム、または IEHS) をインストールすることができます。このオプションを使用するには、インストール後に IBM Security Directory Integrator ヘルプ・ファイルを手動でダウンロードし、デプロイする必要があります。

ご使用のプラットフォームがこれらのシステム要件を満たしている場合、50 ページの『ローカル・ヘルプ・ファイルのインストール』に記載されているダウンロードおよびインストールの説明に進むことができます。

組み込み Web プラットフォーム v8.1.0.3 (Integrated Solutions Console SE を含む)　IBM Security Directory Integrator には、「LWI」としても知られる、軽量の組み込み Web サーバー・プラットフォームが含まれています。このサーバ

ー・プラットフォームは、Eclipse および Open Services Gateway Initiative (OSGI) アーキテクチャーをベースにしており、Web アプリケーションと Web サービスの実行をサポートします。ランタイムにより、小規模なフットプリントと最小構成でセキュア・インフラストラクチャーが提供されます。 組み込み Web プラットフォームには Integrated Solution Console SE が含まれており、これは、既存の ISC インストール済み環境に AMC をデプロイするためのデフォルトの代替として使用されます。組み込み Web プラットフォームには、次のような特性を持つ、Web アプリケーションと Web サービスをホスティングする OSGI ベースの軽量のインフラストラクチャーが備わっています。

- 最小のフットプリント
- 最小の構成
- OSGI ベースの ISC との互換性

注: AMC 機能および組み込み Web プラットフォームは推奨されません。IBM Security Directory Integrator の将来のバージョンでは削除されます。

AMC: 管理およびモニター・コンソール

IBM Security Directory Integrator サーバーの稼働をモニターおよび管理するためのブラウザー・ベース・アプリケーション。 AMC は、Integrated Solutions Console (ISC) 内で動作します。以前のリリースでは、AMC は IBM WebSphere Application Server の組み込みインスタンスあるいは既存のインスタンスにデプロイされるサーブレット・アプリケーションでした。

注: AMC 機能は推奨されません。IBM Security Directory Integrator の将来のバージョンでは除去されます。
IBM WebSphere Application Server では ISC SE 7.2.0.2 および IBM Dashboard Application Services Hub バージョン 3.1 がサポートされます。

Password Synchronization Plug-ins

IBM Security Directory Integrator により構築されるソリューションでは、多数のシステムでのパスワードの変更をインターセプトできます。

選択不可能で自動的にインストールされる追加コンポーネント:

Java ランタイム環境 (JRE) 7.0.4

標準 Java プラットフォームを構成するコアの実行可能ファイルおよびその他のファイルを含む、Java Development Kit (JDK) のサブセットです。JRE には、 Java 仮想マシン (JVM)、コア・クラス、およびそれをサポートするファイルが組み込まれています。

注: インストール済み IBM Security Directory Integrator パッケージのいずれかに使用される JRE は、システムにインストールした可能性のあるシステム規模の JRE または JDK とは独立しています。この JRE は、どの機能を選択したかに関係なくインストールされます。アンインストーラーで JRE が必要なため、JRE は常にインストールされます。

各種設定

ライセンス・パッケージ、アンインストーラー、および更新インストーラーを含みます。

IBM Security Directory Integrator のライセンス・パッケージには、IBM Security Directory Integrator のライセンス・ファイルが含まれています。

その他の要件

ご使用のシステムが、ここで説明するその他の要件を満たしていることを確認してください。

ルート特権または管理者特権

自身がユーザーとして、ルート特権または管理者特権に関する要件を満たしていることを確認してください。

IBM Security Directory Integrator を管理者特権でインストールする場合と非管理者特権でインストールする場合とでは、次の違いがあることに注意してください。

- IBM Security Directory Integrator を指定したインストール・ロケーションにインストールする場合は、すべてのユーザーに書き込み特権が必要です。
- 非管理者ユーザーの構成エディターへのショートカットは、管理者ユーザーとは異なります。
- 管理者特権を持たないユーザーが IBM Security Directory Integrator をインストールする時、「AMC をサービスとして登録する」ウィンドウおよび「サーバーをシステム・サービスとして登録する」ウィンドウは表示されません。
- 特定の非 root ユーザー ID を使用して IBM Security Directory Integrator をインストールした場合、アンインストールや新バージョンへのマイグレーションなど、インストール済み環境に対する以後のメンテナンスでは、同じユーザー ID を使用する必要があります。

Security Enhanced (SELinux)

以下の手順に従って、設定を変更して、SELinux を実行できます。これは、エラー・フリーで SELinux を実行するのに役立ちます。

RedHat Linux (RHEL) には、Security Enhanced Linux あるいは SELinux として知られるセキュリティー機能があります。SELinux は、ホストを特定の悪意ある攻撃から保護するセキュリティーを提供します。 RHEL バージョン 5.0 では、デフォルトで SELinux が使用可能になります。RHEL 5.0 SELinux のデフォルトの設定では、Java の正常動作が妨げられることが知られています。 RHEL 5.0 IBM Security Directory Integrator インストーラーの実行を試みると、以下の出力に似たエラーが表示される可能性があります。

```
# ./install_sdiv72_linux_x86_64.bin

Initializing Wizard........
Verifying JVM...

No Java Runtime Environment (JRE) was found on this system.
```

このエラーの原因は、InstallAnywhere 2012 SP1 によって `/tmp` ディレクトリーに抽出された Java ランタイム環境 (JRE) が実行に必要な権限を持っていないことです。このエラーを回避するには、以下の操作を行います。

1. SELinux を使用不可にします: `setenforce 0`
2. IBM Security Directory Integrator インストーラーを実行します。
3. 再び SELinux を使用可能にします: `setenforce 1`

構成ファイル /etc/selinux/config を編集して、SELinux を使用可能または使用不可にすることもできます。/etc/selinux/config ファイルのデフォルトの設定は、次のような行で構成されています。

```
# This file controls the state of SELinux on the system.
# SELINUX= can take one of these three values:
# enforcing - SELinux security policy is enforced.
# permissive - SELinux prints warnings instead of enforcing.
# disabled - SELinux is fully disabled.
SELINUX=enforcing
# SELINUXTYPE= type of policy in use. 値は以下のとおりです。
# targeted - Only targeted network daemons are protected.
# strict - Full SELinux protection.
SELINUXTYPE=targeted
```

SELINUX を SELINUX=permissive または SELINUX=disabled のいずれかに変更すると、IBM Security Directory Integrator インストーラーを実行することができます。ただし、SELINUX プロパティーを SELINUX=permissive または SELINUX=disabled のいずれに変更しても、ホストのセキュリティー・レベルに影響を与えます。

IBM Security Directory Integrator インストーラーは、*install_dir*/jvm にある JRE を使用しますが、これは SELinux のデフォルトの設定では実行することができません。インストーラーは、SELinux のデフォルトの設定が原因のこの問題を回避するための最善策として、インストーラーをブロックしている IBM Security Directory Integrator JRE のセキュリティー権限を変更しようとします。IBM Security Directory Integrator インストーラーは、JRE を実行できるようにするため、IBM Security Directory Integrator JRE のセキュリティー権限を変更するコマンドを発行します。IBM Security Directory Integrator インストーラーは次のコマンドを発行します。

```
chcon -R -t textrel_shlib_t install_dir/jvm/jre
```

注: インストーラーが chcon コマンドを発行できない場合、あるいはコマンドを発行したときにエラーが発生した場合は、権限を手動で編集する必要があります。以下の出力に似たエラーは、chcon コマンドが機能しなかったことを示します。

```
[root@dyn9-37-225-164 V7.2]# ./ibmdisrv
Failed to find VM — aborting

[root@dyn9-37-225-164 V7.2]# ./ibmditk
Failed to find VM — aborting

[root@dyn9-37-225-164 V7.2]# bin/amc/start_tdiamc.sh
Failed to find VM - aborting
```

Unix/Linux での AMC の認証

AMC で作業する場合、非 root ユーザーに対する制約がいくつかあります。それらと使用可能な回避策の詳細については、以下の情報を参照してください。

一部の UNIX プラットフォームでは、ISE SE の管理およびモニター・コンソール (AMC) は、正しい資格情報を指定されていても、ユーザーの認証に失敗します。このような動作は、AMC が非 root ユーザーとして実行されており、オペレーティング・システムがパスワード・データベース (例えば、/etc/shadow ファイル) を使用している場合に発生します。この問題の詳細およびその回避策については、「トラブルシューティング」の『Authentication failure on UNIX when LWI runs as non-root user』を参照してください。

UNIX システム用のグラフィックス・パッケージ

以下の手順に従って、必要なグラフィックス・パッケージがない状態で CE を実行しているときに発生したエラーを解決します。

UNIX システム上に必要なグラフィックス・パッケージがインストールされていない場合、ibmditk コマンドを実行して構成エディターを開始すると以下のエラーが発生する可能性があります。

```
No fonts found; this probably means that the fontconfig library is not correctly
configured. You may need to edit the fonts.conf configuration file.
More information about fontconfig can be found in the fontconfig(3) manual page
and on http://fontconfig.org
```

このようなエラーを回避するには、以下のステップを実行します。

1. 以下のグラフィックス・パッケージが UNIX システム上にインストールされていることを確認します。
   - libgtk-x11-2.0.so.0
   - libgthread-2.0.so.0
2. 次のコマンドを実行します。

   ```
   export LD_LIBRARY_PATH=/usr/sfw/lib/:/usr/lib:/lib
   ```
3. 以下のファイルをインストールするか、インストールされていることを確認します。

   ```
   /etc/fonts/fonts.conf
   ```
4. 次のコマンドを実行します。

   ```
   export FONTCONFIG_PATH=/etc/fonts
   ```

AIX オペレーティング・システムでの CE の前提条件

以下の手順に従って、AIX に RPM をインストールします。

CE は、AIX オペレーティング・システム上の Eclipse にプラグインとしてインストールしても起動せず、ログ・ファイルが作成されます。

CE を使用するには、AIX で gtk+ RPM および依存コンポーネントが使用可能でなければなりません。AIX に以下の RPM をインストールしてください。

```
atk-1.12.3-2.aix5.2.ppc.rpm
cairo-1.8.8-1.aix5.2.ppc.rpm
expat-2.0.1-1.aix5.2.ppc.rpm
fontconfig-2.4.2-1.aix5.2.ppc.rpm
freetype2-2.3.9-1.aix5.2.ppc.rpm
gettext-0.10.40-6.aix5.1.ppc.rpm
glib2-2.12.4-2.aix5.2.ppc.rpm
gtk2-2.10.6-4.aix5.2.ppc.rpm
libjpeg-6b-6.aix5.1.ppc.rpm
libpng-1.2.32-2.aix5.2.ppc.rpm
libtiff-3.8.2-1.aix5.2.ppc.rpm
pango-1.14.5-4.aix5.2.ppc.rpm
pixman-0.12.0-3.aix5.2.ppc.rpm
xcursor-1.1.7-3.aix5.2.ppc.rpm
xft-2.1.6-5.aix5.1.ppc.rpm
xrender-0.9.1-3.aix5.2.ppc.rpm
zlib-1.2.3-3.aix5.1.ppc.rpm
```

注: 旧バージョンや新バージョンは互換性がない可能性があるため、インストールする RPM のバージョンは、ここに記載したとおりでなければなりません。

これらの RPM バージョンをインストールするには、以下の手順を実行します。

1. RPM を新しいディレクトリーにダウンロードします。RPM は ftp://ftp.software.ibm.com/aix/freeSoftware/aixtoolbox/RPMS/ppc/ にあります。
2. ダウンロードした RPM は、次のコマンドを使用してインストールします。既存のバージョンの RPM が既にインストールされている場合、このコマンドは、ダウンロードされたバージョンにアップグレードまたはダウングレードします。

   ```
   rpm -U *.rpm --force
   ```

3. ライブラリーのクロージャーのパスが環境変数 LIBPATH に含まれていることを確認します。例: LIBPATH=/opt/freeware/64/lib/。

Windows 2012 オペレーティング・システムで V7.1.1 から V7.2 にアップグレードする場合の前提条件

バージョンをアップグレードする場合の前提条件の詳細については、以下の情報を参照してください。

Windows 2012 オペレーティング・システムで IBM Security Directory Integrator バージョン 7.1.1 からバージョン 7.2 にアップグレードする予定の場合は、バージョン 7.1.1 の uninstaller.exe で **Windows 7 互換モード**が有効になっていることを確認してからバージョン 7.2 のインストーラーを開始してください。詳しくは、http://www-01.ibm.com/support/docview.wss?uid=swg21634336 に記載されている技術情報を参照してください。

IBM Security Directory Integrator のインストール

インストーラーを使用して、IBM Security Directory Integrator 全体、または必要な IBM Security Directory Integrator コンポーネントのみをインストールできます。また、以前のバージョン (バージョン 7.0、7.1、または 7.1.1) をアップグレードしたり、既存の IBM Security Directory Integrator インストール済み環境に機能を追加したりすることができます。

注:

- IBM Security Directory Integrator をバージョン 6.x 以前からバージョン 7.2 に直接アップグレードすることはサポートされていません。まずバージョン 6.x をバージョン 7.1.1 にアップグレードしてから、バージョン 7.1.1 をバージョン 7.2 にアップグレードする必要があります。
- IBM Security Directory Integrator は、以下のオペレーティング・システムでは構成エディター (CE) をサポートしません。
  - Linux PPC
  - Linux 390
  - AIX 64 ビット

  構成エディターをローカルにインストールせずに製品を使用する方法について詳しくは、163 ページの『リモート構成エディターの使用』を参照してください。

旧バージョンからのアップグレードを選択すると、IBM Security Directory Integrator によって旧バージョンがアンインストールされます。アンインストール後も、ユーザーが作成したファイルは削除されません。ユーザーが作成したファイルは、新しいインストールが完了した後でも使用可能です。`global.properties` および

am_config.properties などの構成ファイルは、行ったカスタム構成変更が保持されたまま、IBM Security Directory Integrator バージョン 7.2 にマイグレーションされます。

注: IBM Security Directory Integrator インストーラーによって、事前定義の構成ファイルおよびプロパティー・ファイルの一部はバックアップおよびリストアが行われますが、重要なデータが含まれるファイルおよびデータベースについては、インストールを開始する前に手動でもバックアップすることをお勧めします。

IBM Security Directory Integrator バージョン 7.2 のインストール済み環境には、IBM Security Directory Integrator の以前のバージョンで使用可能であった以下の機能が、引き続き含まれます。

- 管理およびモニター・コンソール (AMC)。
- 構成エディター (CE)
- 例
- Eclipse を採用した IBM ユーザー・インターフェース・ヘルプ・システム
- Java API 文書
- ランタイム・サーバー

注: この「*インストールと管理*」では以降、変数 *TDI_install_dir* は、インストール時にユーザーが「**宛先パネル**」で選択したインストール・ディレクトリーの場所を指します。一般に IBM Security Directory Integrator がインストールされる場所については、57 ページの『デフォルトのインストール・ロケーション』を参照してください。

適切なインストーラーの起動

インストーラーは、ランチパッドを使用するか、直接インストールして起動できます。詳しい起動手順については、以下の情報を参照してください。

以下のいずれかの方法により、IBM Security Directory Integrator インストーラーを起動できます。

ランチパッドからインストーラーを起動する

IBM Security Directory Integrator ランチパッドにより、インストールに関する重要な入門情報、およびさまざまなインストール、マイグレーション、インストール後のトピックに関するより詳細な情報へのリンクが提供されています。またランチパッドを使用すると、IBM Security Directory Integrator インストーラーを起動することもできます。

注: ランチパッドを使用するには、サポートされている Web ブラウザーがインストールされており、構成済みである必要があります。そうしておかないと、ランチパッドは使用できません。ただし、その場合でも、プラットフォーム固有のインストーラーは直接使用することができます。IBM Security Directory Integrator インストーラーの使用法の説明については、13 ページの『プラットフォーム固有の IBM Security Directory Integrator インストーラーの使用』を参照してください。

1. コマンド・プロンプトで以下のコマンドを入力して、IBM Security Directory Integrator ランチパッドをオープンします。

- Windows プラットフォームの場合、以下のように入力します。

  ```
  Launchpad.bat
  ```

- 他のすべてのプラットフォームの場合、以下のように入力します。

  ```
  Launchpad.sh
  ```

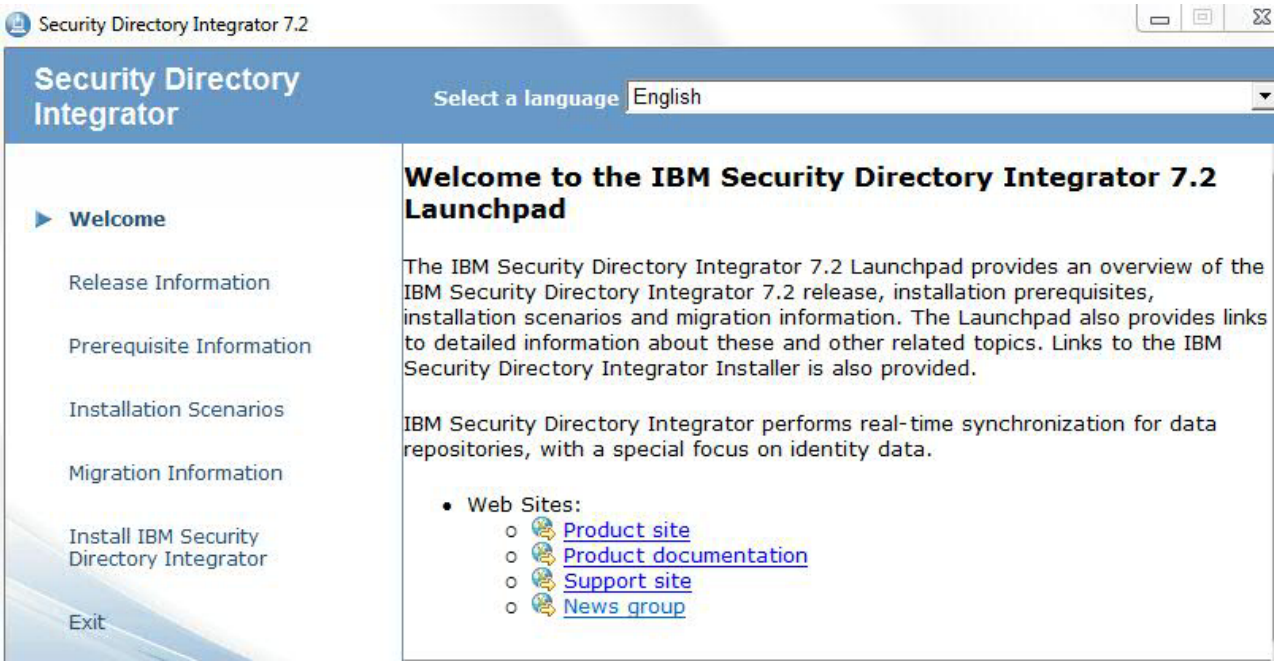
ランチパッドの左側のメニューを使用して、各ランチパッド・ウィンドウにナビゲートできます。メニュー項目をクリックして、関連情報を表示します。以下のメニュー項目が選択可能です。

ようこそ

インストールの「ようこそ」ウィンドウには以下の場所へのリンクが含まれています。

- IBM Security Directory Integrator Web サイト
- IBM Security Directory Integrator 資料
- サポート Web サイト
- IBM Security Directory Integrator ニュース・グループ

画面キャプチャー名: Adminst-1



IBM Security Directory Integrator ランチパッド・ウィンドウの左側には、以下のオプションがあります。

リリース情報

ウィンドウには、このリリースに関する文書へのリンクと、このリリースで使用可能な特定の新規および改良機能のリストが含まれます。

前提条件情報

このウィンドウには、プラットフォームのサポートおよびハードウェア要件に関する情報へのリンクが含まれます。

インストール・シナリオ

このウィンドウには、インストール可能な IBM Security Directory Integrator コンポーネントについての説明が含まれます。インストール時にこれらのコンポーネントの一部またはすべてをインストールできます。このウィンドウには、インストール可能な Password Synchronization Plug-ins コンポーネントについての説明も含まれています。

マイグレーション情報

このウィンドウには、IBM Security Directory Integrator 7.0、7.1、または 7.1.1 からバージョン 7.2 へのマイグレーションに関する情報へのリンクが含まれます。また、Derby システム・ストアのマイグレーションに関する情報も含まれています。

注: IBM Security Directory Integrator をバージョン 6.x 以前からバージョン 7.2 に直接アップグレードすることはサポートされていません。まずバージョン 6.x をバージョン 7.1.1 にアップグレードしてから、バージョン 7.1.1 をバージョン 7.2 にアップグレードする必要があります。

IBM Security Directory Integrator のインストール

このウィンドウには、インストール、マイグレーション、およびサポートされるプラットフォームに関する文書へのリンクと、IBM Security Directory Integrator インストーラーへのリンクが含まれます。IBM Security Directory Integrator インストーラーの使用方法については、13 ページの『プラットフォーム固有の IBM Security Directory Integrator インストーラーの使用』を参照してください。

IBM Security Directory Integrator Password Synchronization Plug-ins のインストール (Install IBM Security Directory Integrator Password Synchronization Plug-ins)

このウィンドウには、インストールおよびサポートされるプラットフォームに関する文書へのリンクと、IBM Security Directory Integrator Password Synchronizer Plug-ins インストーラーへのリンクが含まれます。

注: このウィンドウは、Linux PPC および Linux 390 プラットフォームでは使用できません。

終了 何もインストールしないでランチパッドを終了します。

2. インストール・ウィンドウで、「IBM Security Directory Integrator **インストーラー**」をクリックします。これにより、インストーラーが起動されます。インストーラーの使用方法については、13 ページの『プラットフォーム固有の IBM Security Directory Integrator インストーラーの使用』を参照してください。

インストーラーを直接起動する

インストール実行可能ファイルを使用して、インストーラーを直接起動できます。

1. 製品 CD の tdi_installer ディレクトリーで、ご使用のプラットフォームに応じたインストール実行可能ファイルを見つけます。

   Windows Intel
   install_sdiv72_win_x86.exe

   Windows 64 ビット
   install_sdiv72_win_x86_64.exe

   AIX install_sdiv72_aix_ppc.bin

   AIX 64 ビット
   install_sdiv72_aix_ppc_64.bin

   Linux 64 ビット
   install_sdiv72_linux_x86_64.bin

   Power® PC Linux
   install_sdiv72_ppclinux.bin

   Solaris Sparc
   install_sdiv72_solaris_sparc.bin

   Solaris (Intel)
   install_sdiv72_solaris_x86_64.bin

2. 実行可能ファイルをダブルクリックするか、コマンド・プロンプトで実行可能ファイル名を入力します。これにより、インストーラーが起動されます。インストーラーの使用方法については、『プラットフォーム固有の IBM Security Directory Integrator インストーラーの使用』を参照してください。

インストーラーを起動すると、『プラットフォーム固有の IBM Security Directory Integrator インストーラーの使用』の処理を開始する準備が完了します。

プラットフォーム固有の IBM Security Directory Integrator インストーラーの使用

以下の手順に従って、プラットフォーム固有の IBM Security Directory Integrator をインストールします。

プラットフォーム固有の IBM Security Directory Integrator インストーラーは、ランチパッドまたはコマンド行から起動します。IBM Security Directory Integrator インストーラーは、IBM Security Directory Integrator の新規コピーのインストール、既存の IBM Security Directory Integrator インスタンスへの機能の追加、および IBM Security Directory Integrator の以前のバージョンのアップグレードに使用することができます。ご使用のコンピューターで IBM Security Directory Integrator がインストールされるデフォルトの場所は、プラットフォームによって異なります。

インストール時に、インストーラーはそのアクションを、システムの一時ファイル・ディレクトリー (UNIX プラットフォームでは通常 /tmp または /var/tmp) 内のファイル (sdiv72install.log および sdiv72debug.log) にログとして記録します。

グラフィカル・インストーラーを使用したインストール

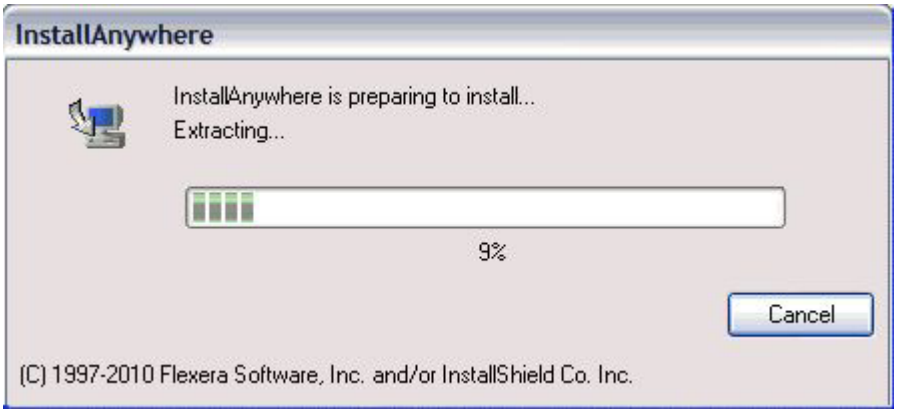
以下の手順に従って、グラフィカル・インストーラーを使用して IBM Security Directory Integrator をインストールします。

インストール・パネル・フロー

以下の手順に従って、パネル・フローをインストールします。

事前初期設定パネル

インストーラーの実行可能ファイルは、コマンド行から、または実行可能ファイルをダブルクリックして起動できます (Windows のみ)。最初は、このパネルの後にスプラッシュ画面が表示されます。

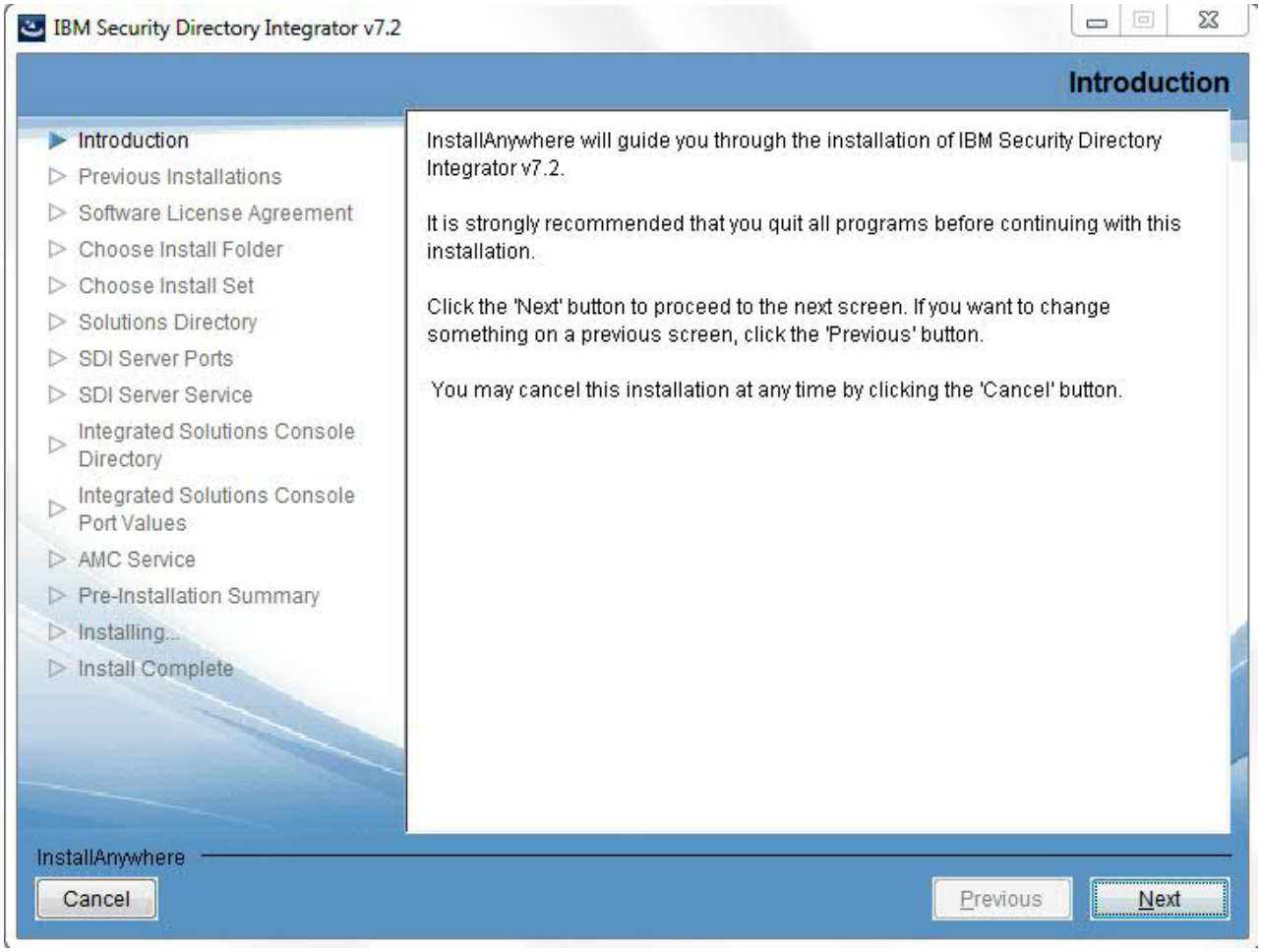


注: 基礎となるシステムが複数の言語をサポートしている場合、スプラッシュ画面には、言語選択のドロップダウン・リストも表示されます (デフォルトは英語)。


IBM Security Directory Integrator v7.2
IBM
Security Directory Integrator
7.2
Licensed Materials - Property of IBM Corp. © Copyright IBM Corporation and its licensors 2002, 2013. All Rights Reserved. IBM, the IBM logo, and Tivoli are trademarks of IBM Corporation in the United States, other countries, or both. Built on Eclipse is a trademark of Eclipse Foundation, Inc. Java and all Java-based trademarks and logos are trademarks or registered trademarks of Oracle and/or its affiliates. Other company, product or service names may be trademarks or service marks of others
BUILT ON
eclipse
Java
7.2
English
OK

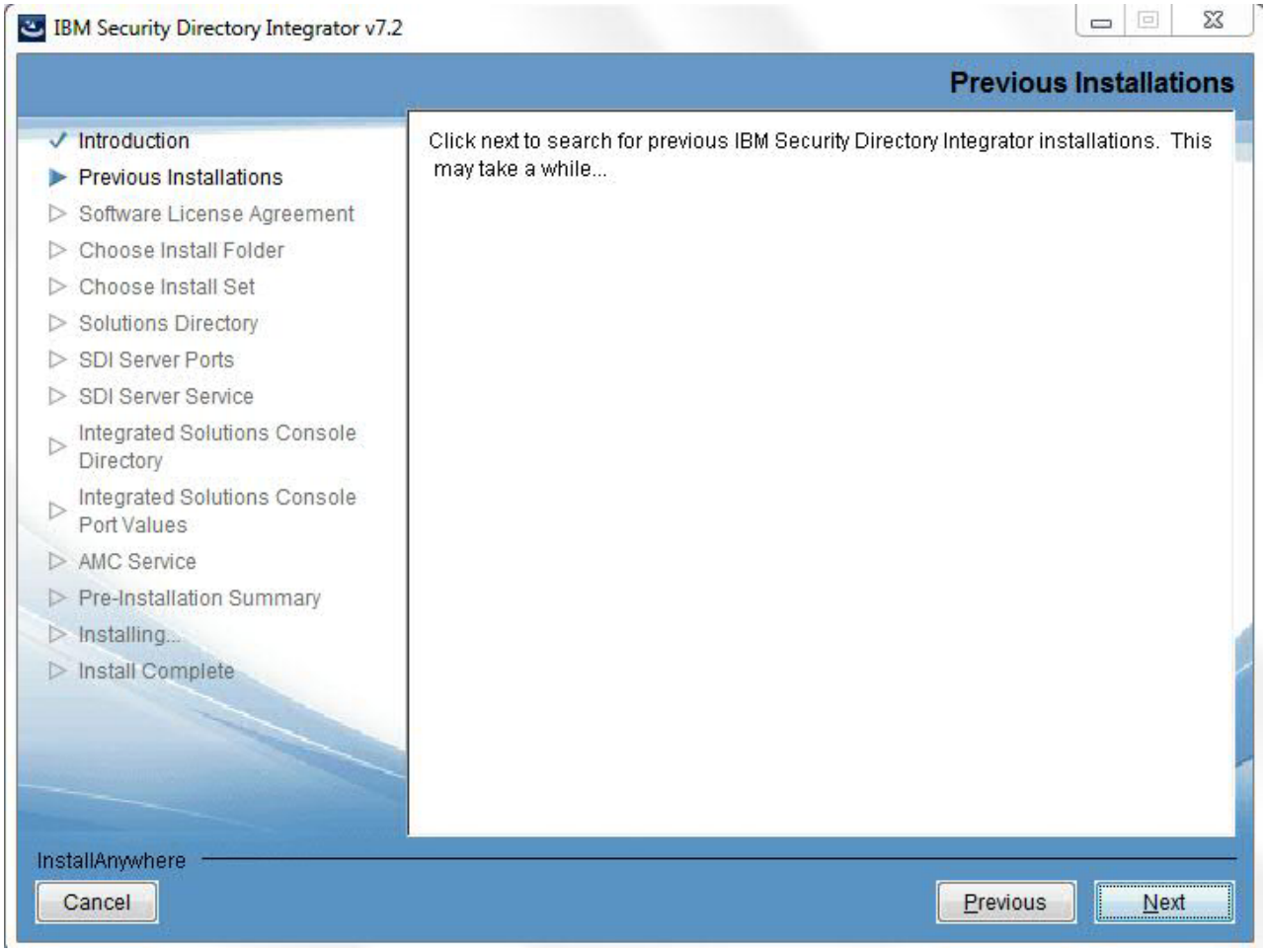
「概要」パネル

これはインストーラーの「ようこそ」パネルです。これは InstallAnywhere インストーラーで提供されているデフォルトのパネルです。「**次へ**」ボタンをクリックして先に進むか、または「**キャンセル**」をクリックしてインストーラーをキャンセルして終了することができます。



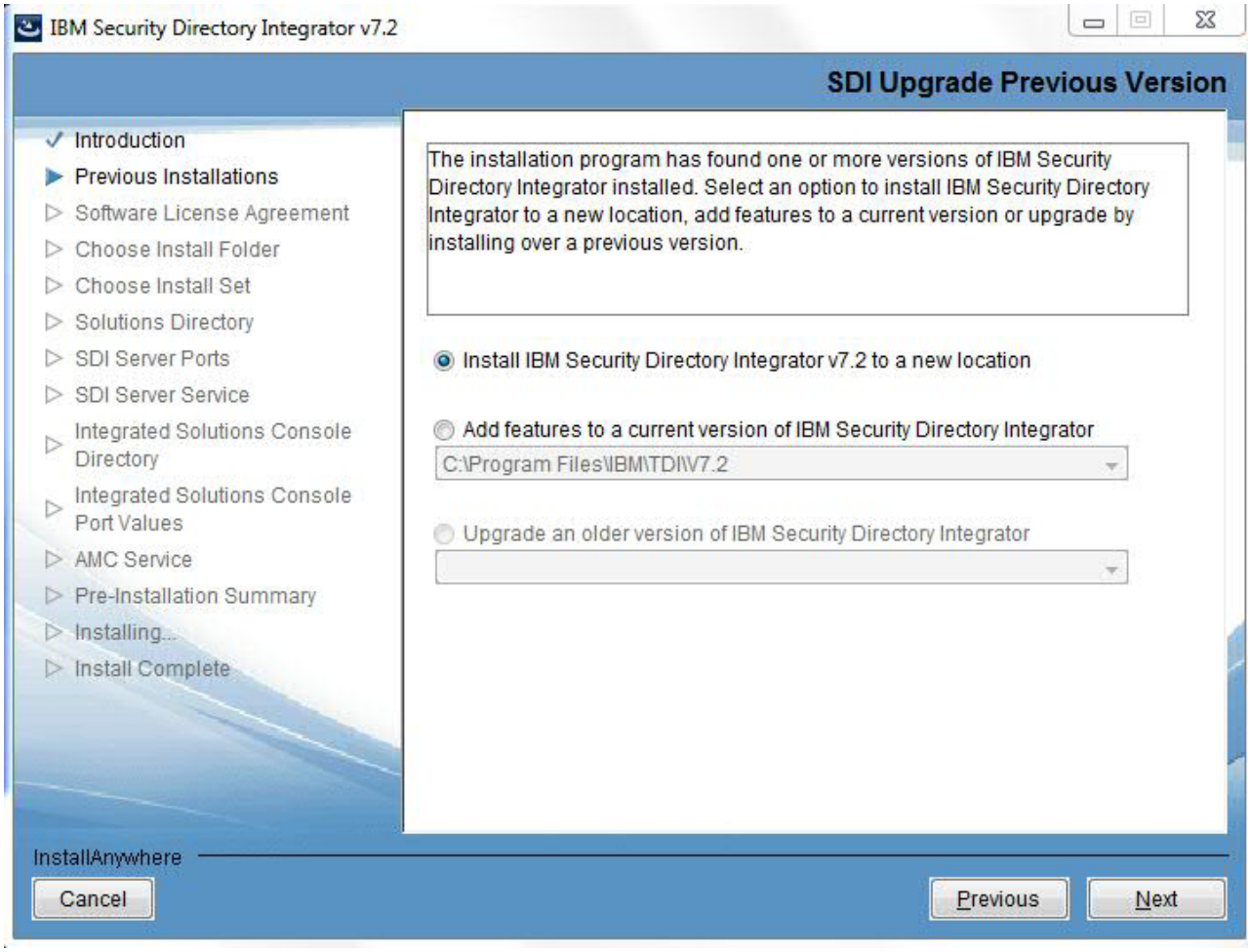
「以前のインストール済み環境」パネル

このパネルでは、以前のバージョンの IBM Security Directory Integrator の検出にはしばらく時間がかかる場合があることを通知します。



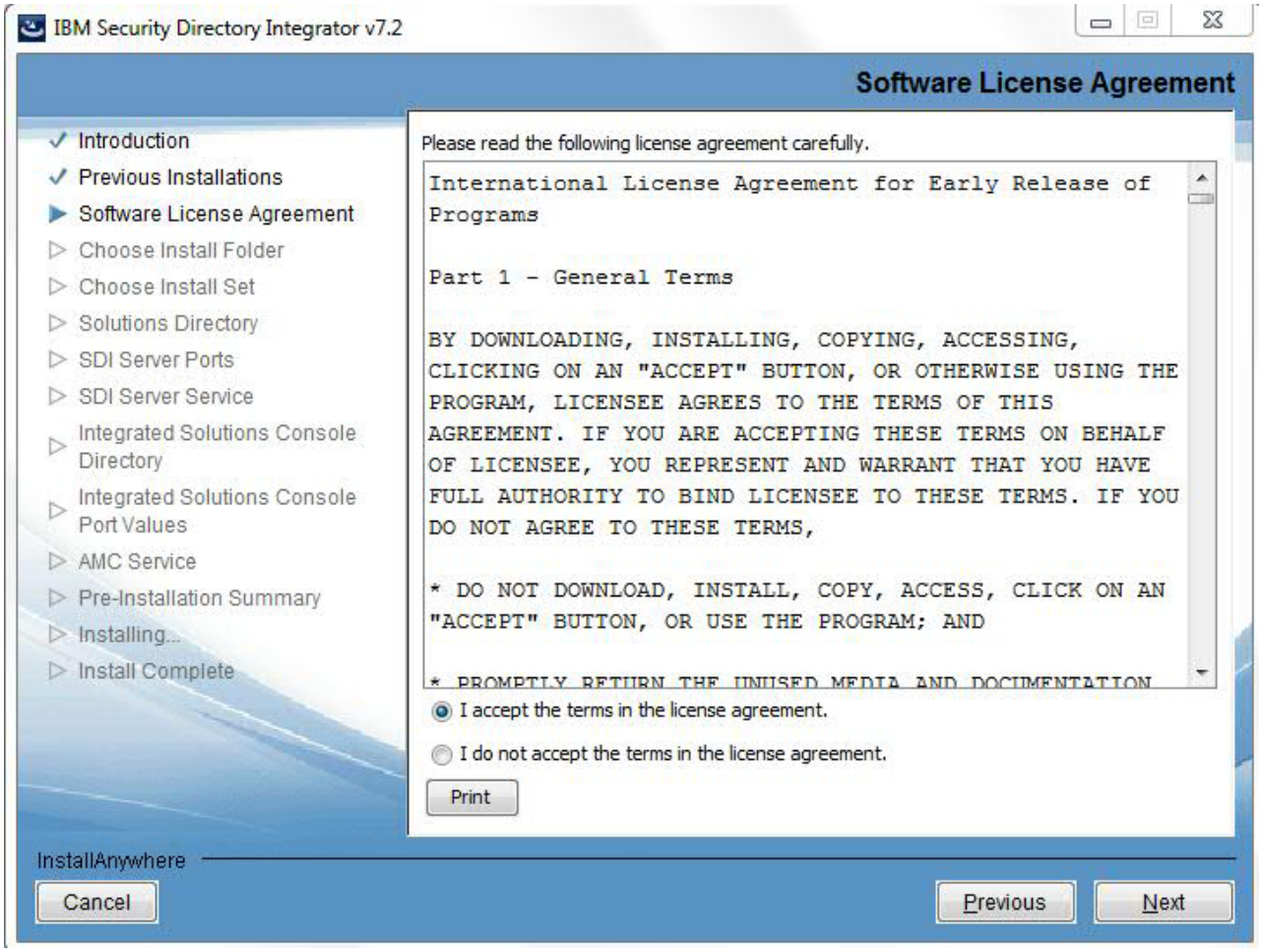
以前のバージョンを検出すると、いくつかのアップグレード・オプションが示されます。

注: IBM Security Directory Integrator をバージョン 6.x 以前からバージョン 7.2 に直接アップグレードすることはサポートされていません。まずバージョン 6.x をバージョン 7.1.1 にアップグレードしてから、バージョン 7.1.1 をバージョン 7.2 にアップグレードする必要があります。



「ソフトウェアのご使用条件」パネル

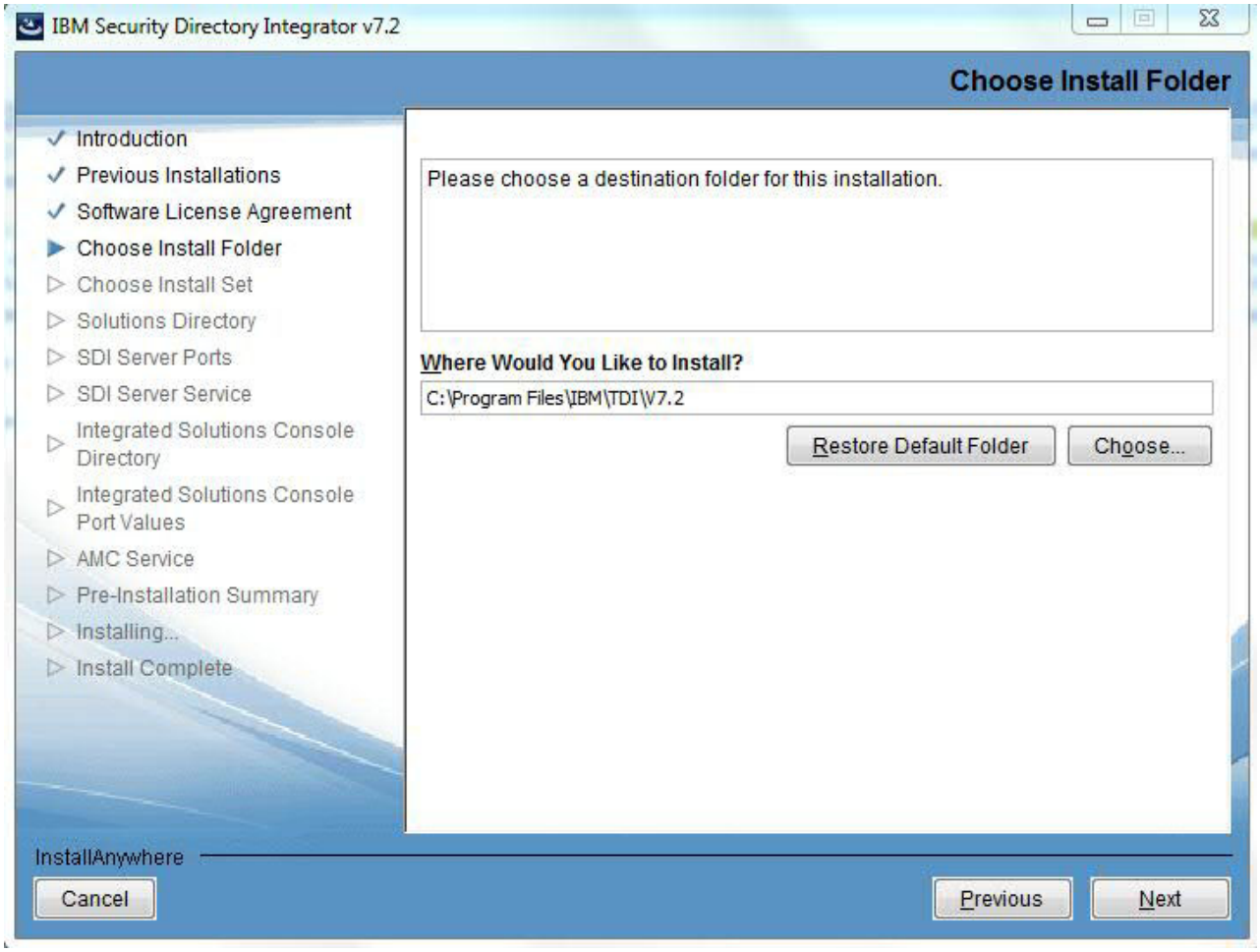
ライセンス・パネルは IBM ライセンス・ツールによって提供されます。このパネルは、「**Security Directory Integrator v72 の新規インストール**」および「**以前のバージョンの Security Directory Integrator のアップグレード**」で表示されます。



「インストール・フォルダーの選択」パネル

注:

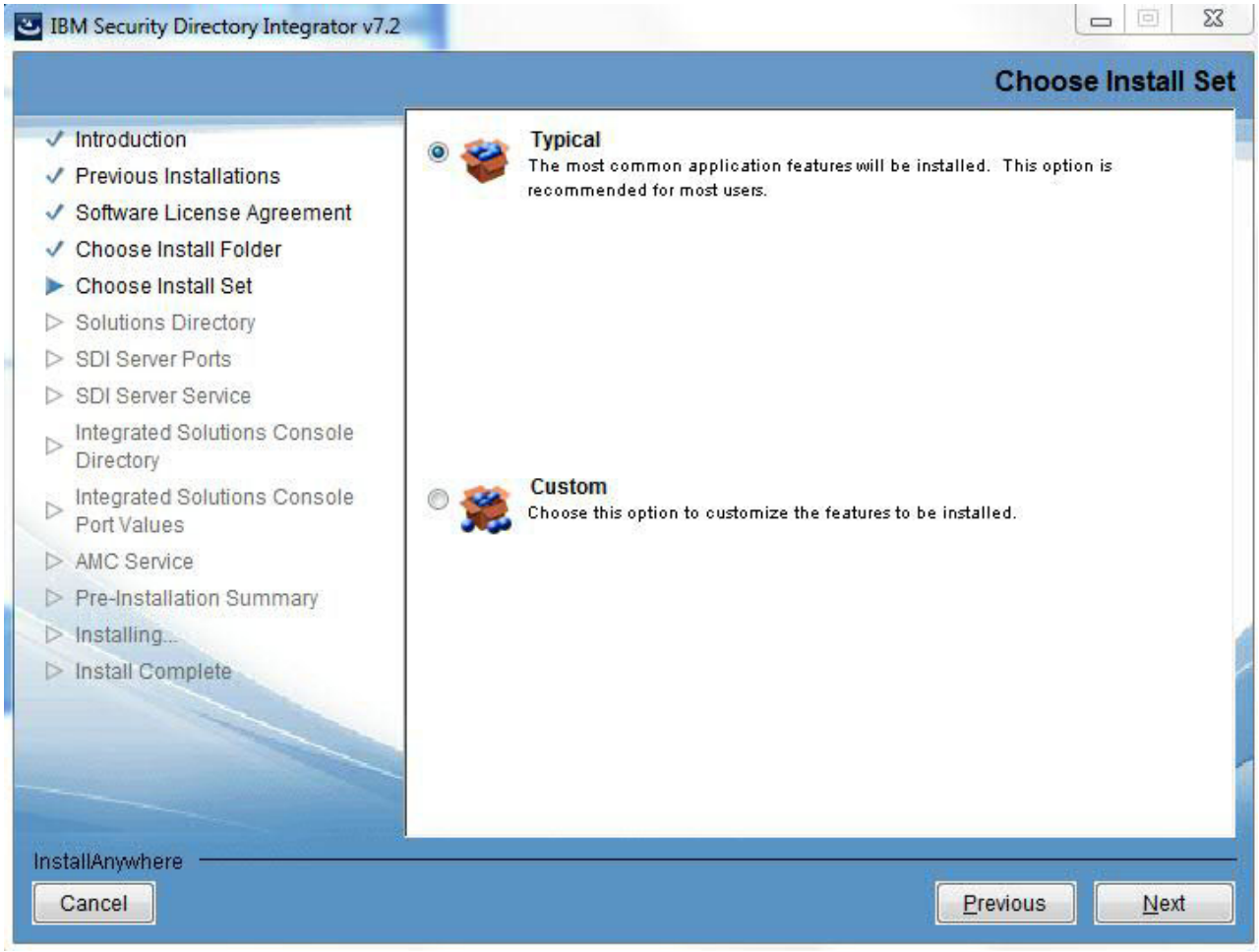
1. このパネルは、IBM Security Directory Integrator バージョン 7.0、7.1、または 7.1.1 からのアップグレードを選択した場合は表示されません。また既存の IBM Security Directory Integrator バージョン 7.2 インスタンスに機能を追加する場合も表示されません。
2. ウィザード内から他のパネルに移動し、その後、戻ってきた場合、宛先パネルには最後に入力した値が表示されます。
3. 非 ASCII 文字および `";|*?!#&$',=^@%+` の各文字は、インストール・パスでは使用できません。



「インストール・セットの選択」パネル

標準インストールには、ランタイム・サーバー、構成エディター (CE)、Javadoc、例、および AMC が含まれます。構成エディター更新サイト、Eclipse を採用した IBM ユーザー・インターフェース・ヘルプ・システム、あるいはパスワード同期プラグインは含まれません。

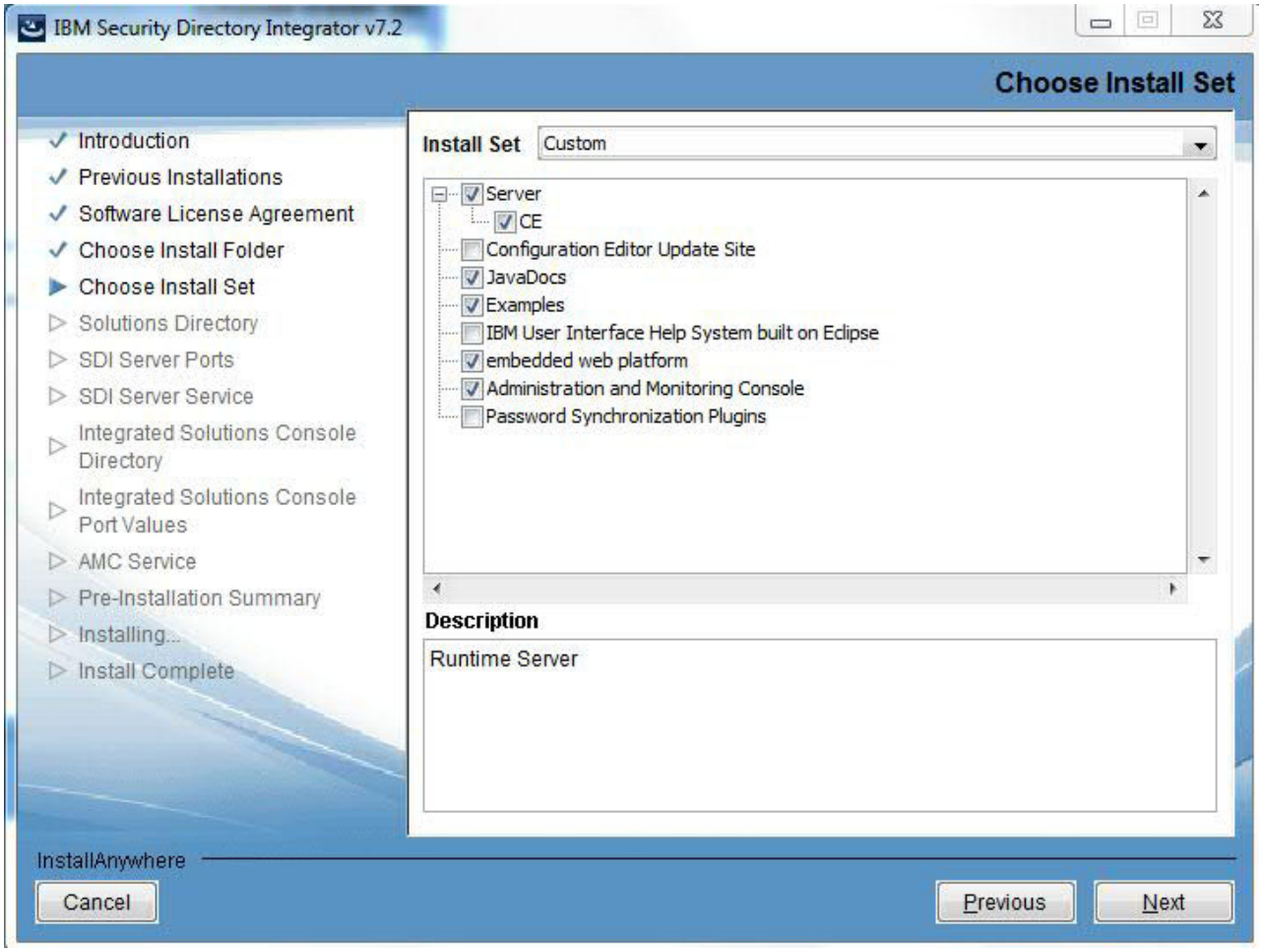
「**標準**」を選択すると、機能選択パネルはスキップされます。また、バンドルされている組み込み Web プラットフォーム/ISC パッケージは、自動的にインストールされます。ISC ディレクトリー・パネルはスキップされます。



機能選択パネル

このパネルを使用すると、インストールする機能を指定できます。すべての機能は必要に応じ、個別にインストールできます。唯一の例外は、構成エディターを選択した場合、サーバーも同時に選択されることです。これは構成エディターがサーバーのサブフィーチャーであることが理由です。

プラットフォームでサポートされていない機能は、機能選択パネルには表示されません。



以下のリストは、各機能についての要約です。

ランタイム・サーバー

IBM Security Directory Integrator 統合ソリューションをデプロイおよび実行するために使用されるルール・エンジン。

構成エディター

IBM Security Directory Integrator 統合ソリューションを作成、デバッグ、および拡張するための開発環境。ランタイム・サーバーをインストールしないで、この機能をインストールすることはできません。

構成エディター更新サイト

Eclipse 更新サイトに合わせて作成されています。既存の Eclipse に構成エディターをインストールするのに必要なファイルを含んでいます。これはメンテナンス用にも使用されます。(zLinux または Linux PPC では使用できません。)

Javadoc

IBM Security Directory Integrator 内部の完全な HTML 資料。 カスタム・コンポーネントの開発や、ソリューションにおけるスクリプト記述のために不可欠な参照資料。

例　特定の IBM Security Directory Integrator の機能またはコンポーネントを中心に説明する、一連の具体的な短い構成の例。

Eclipse を採用した IBM ユーザー・インターフェース・ヘルプ・システム (ローカル・ヘルプ)
Eclipse を採用した IBM ユーザー・インターフェース・ヘルプ・システム (旧称 IEHS) で、グローバル・オンライン・ヘルプ・サービスを使用する代替としてローカルにインストールできます。このオプションを使用するには、インストール後に IBM Security Directory Integrator ヘルプ・ファイルを手動でダウンロードし、デプロイする必要があります。

組み込み Web プラットフォーム
組み込み Web プラットフォーム・パッケージ。ISC SE が含まれています。

管理およびモニター・コンソール
IBM Security Directory Integrator サーバーの実行をモニターおよび管理するためのブラウザー・ベース・アプリケーション。

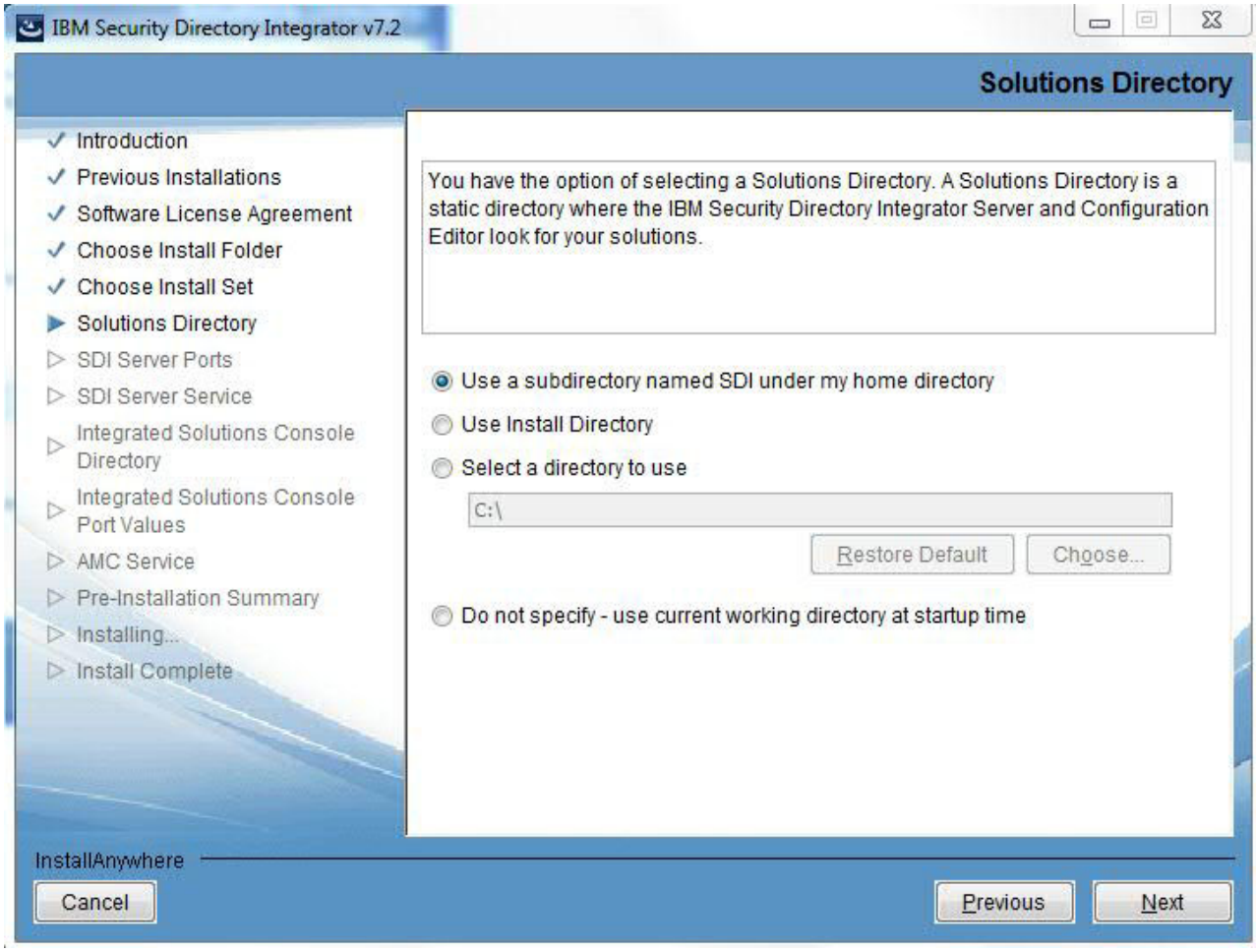
Password Synchronization Plug-ins
IBM Security Directory Integrator の Password Synchronization Plug-ins。

IBM Security Directory Integrator の「ソリューション・ディレクトリー」パネル
このパネルが表示されるのは、サーバー機能を選択した場合のみです。 サーバーのデフォルトのソリューション・ディレクトリーを選択できます。ソリューション・ディレクトリーは、ユーザーが将来の実行のために作成したソリューションを含む静的なディレクトリーです。このパネルは、デフォルトでは、ソリューション・ディレクトリーをユーザーのホーム・ディレクトリーに設定します。

「使用するディレクトリーを選択します」ラジオ・ボタンを選択する場合は、有効な「ソリューション・ディレクトリー」を指定する必要があります。インストール時に「ソリューション・ディレクトリー」では、汎用命名規則 (UNC) パスがサポートされます。

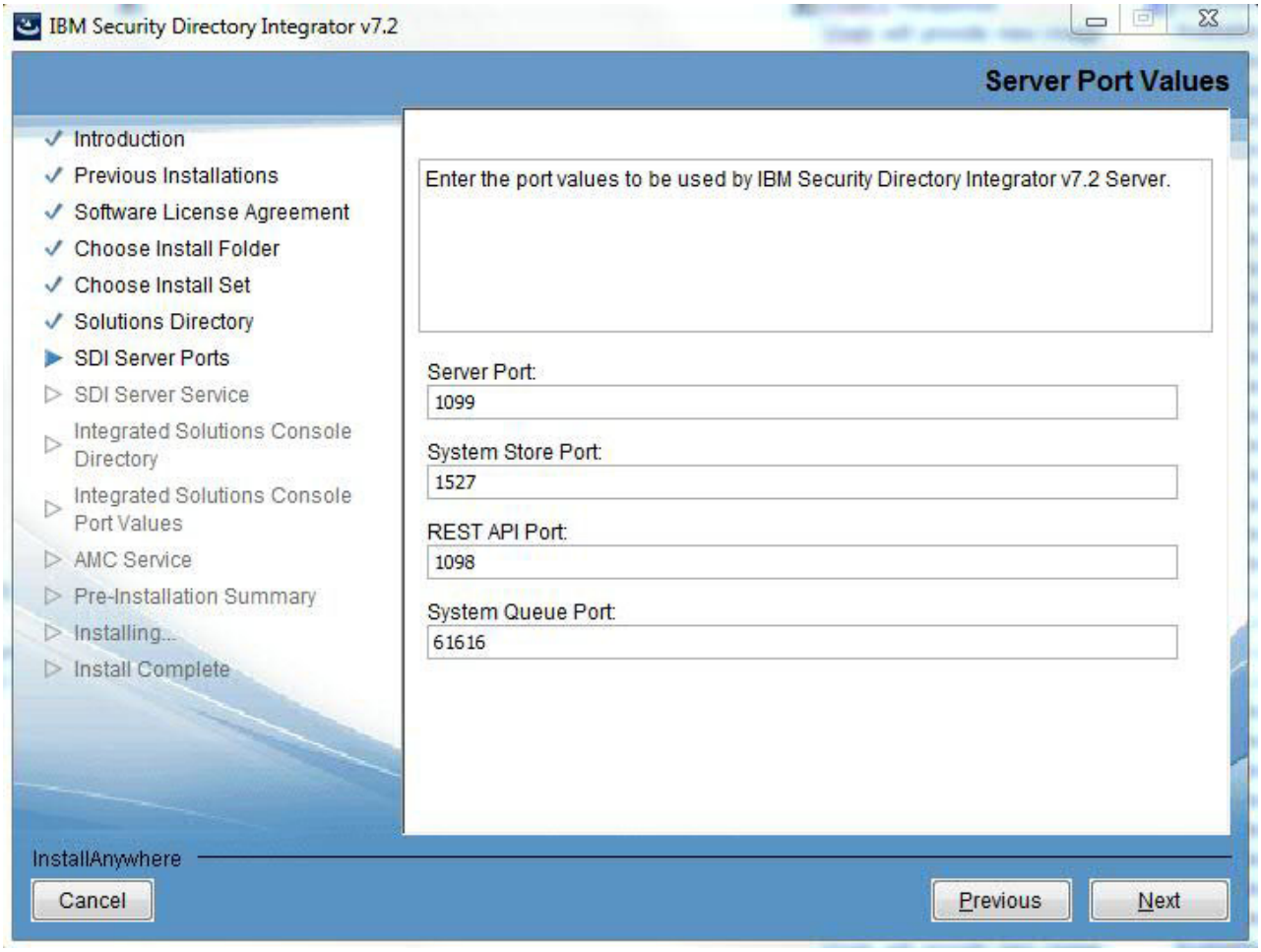
注: このパネルは、IBM Security Directory Integrator バージョン 7.0、7.1、または 7.1.1 からのアップグレードを選択した場合は表示されません。

機能を追加しているときに、サーバー機能が既にインストール済みの場合は、このパネルは表示されません。



サーバー・ポート構成パネル

4 つのサーバー・ポート番号を入力するよう要求されます。これらのポートにはそれぞれデフォルトの値があります。インストーラーはユーザーが有効で使用可能なポート番号を入力したか確認します (『サーバー・ポートの構成』を参照)。



サーバーをシステム・サービスとして登録するパネル

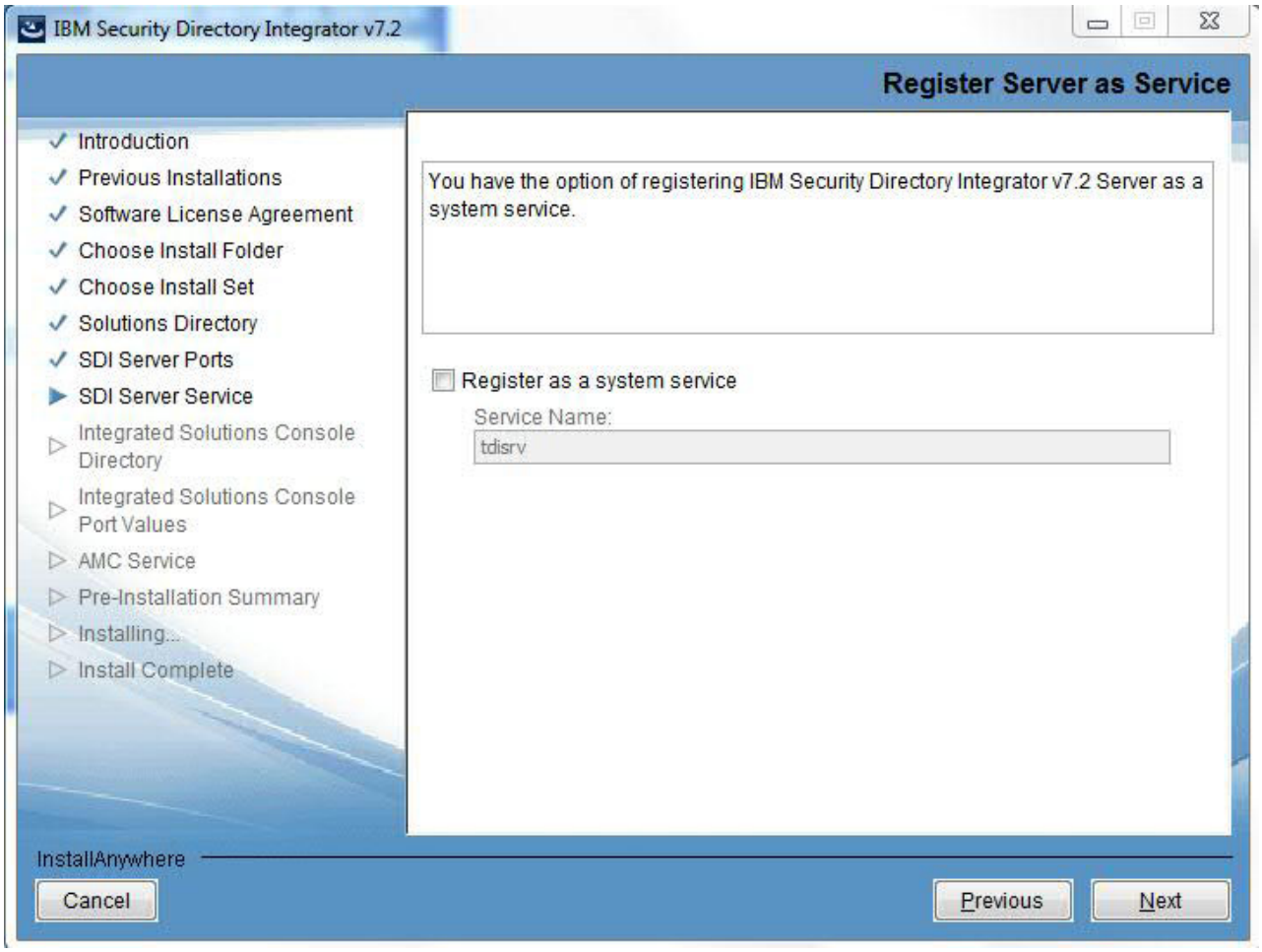
このパネルが表示されるのは、IBM Security Directory Integrator の新しいインスタンスをインストール中であり、インストールする機能としてサーバーを選択している場合か、またはアップグレード・インストールである場合のみです。また、このパネルが表示されるのは、ユーザーが管理者特権を持っている場合のみです。

チェック・ボックスにチェックを付けると、サーバーのみがその OS のサービスとして登録されます。

デフォルトでは、チェック・ボックスのチェックは外されています。 2 つのテキスト・ボックスが使用可能になるのは、 チェック・ボックスにチェックを付けた場合のみです。最初のテキスト・ボックスにはサービス名を入力します。2 番目のテキスト・ボックスには、システム・サービスとしてのサーバーが実行時に使用するポート番号を入力します。

インストーラーは、このサービス名に有効なデフォルト値を与えるために最大限の努力を払います (このプロセスについての詳細は、『サーバーを Windows サービスまたは Unix プロセスとして登録する』を参照してください)。インストーラーが有効なサービス名を判別できない場合、このフィールドはブランクになります。有効なサービス名を入力するまで、先に進めません。

注: UNIX システムにインストールする場合は、サービス名が最大長の 4 文字を超えないようにしてください。4 文字を超える場合、この制限がエラーとなり、インストールを続行できません。



IBM Security Directory Integrator の AMC デプロイメント・パネル

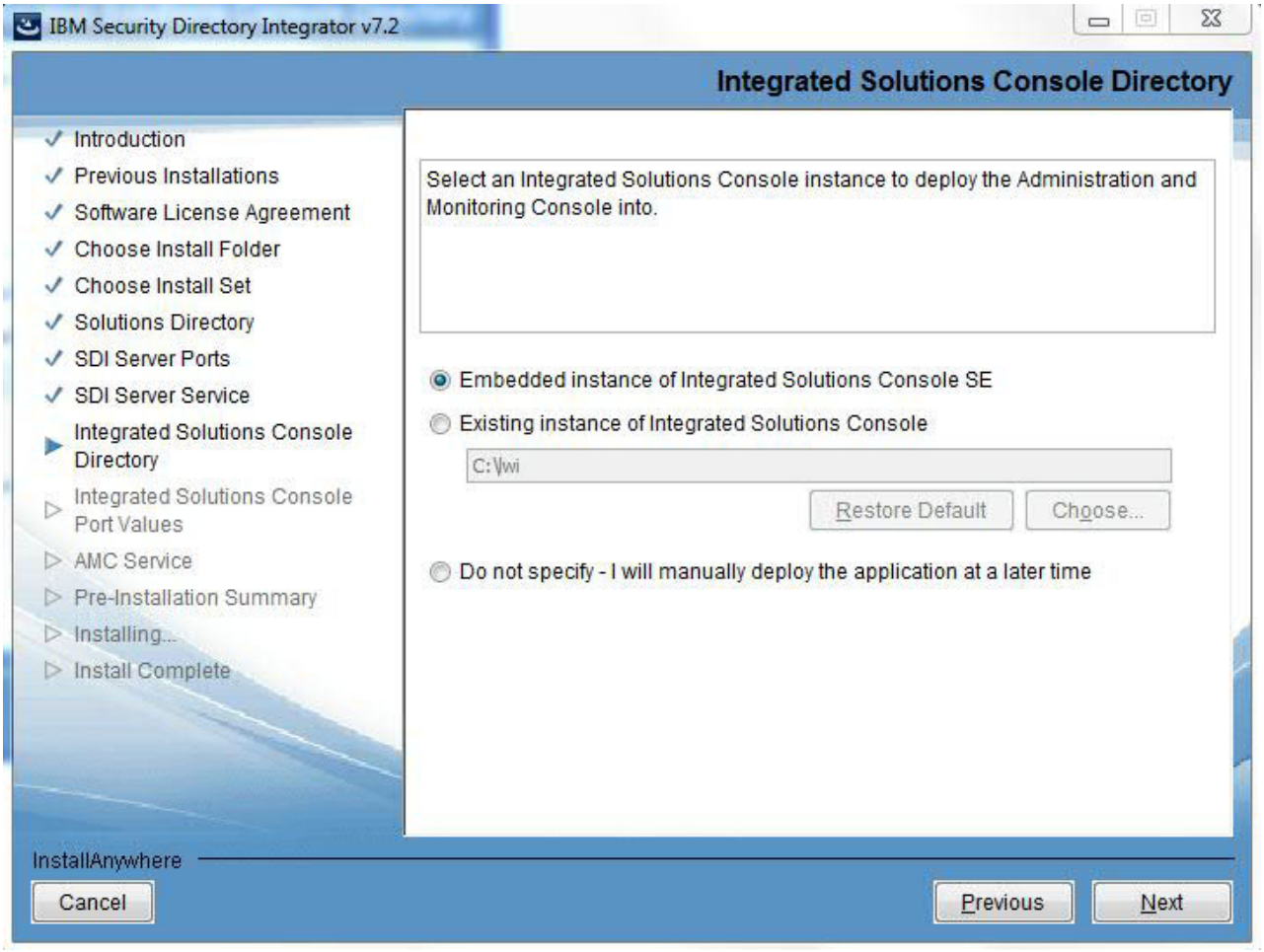
このパネルが表示されるのは、カスタム・インストール・セットを選択し、それと同時に AMC 機能のインストールも選択した場合のみです。AMC がデプロイされる ISC インスタンスを選択する必要があります。IBM Security Directory Integrator に付属しているバンドル ISC、または既にターゲット・マシンにインストール済みの ISC に AMC をデプロイするよう選択、または AMC を後でデプロイするよう選択することもできます。既にインストール済みの ISC を選択した場合、ユーザーは組み込み Web プラットフォーム (LWI) または IBM WebSphere Application Server を含むディレクトリー (例えば、`C:¥Program Files¥IBM¥WebSphere¥AppServer` または `C:¥dev¥IBM¥TDI¥lwi`) を選択する必要があります。

組み込み Web プラットフォーム機能のインストールを選択していなかった場合、選択はグレイ表示になります。

注:

1. 機能を追加しているときに、AMC 機能が既にインストール済みの場合、このパネルはスキップされます。

2. AMC を IBM WebSphere Application Server にデプロイする場合は、組み込み Web プラットフォームにデプロイする場合とは異なって、Security Directory Integrator AMC Admin 役割は自動的には割り当てられません。この役割は ISC コンソールの管理者が手動で割り当てる必要があります。



ISC ポート・パネル

このパネルが表示されるのは、標準インストールまたはカスタム・インストールのいずれかを行っているときに、ISC の組み込みインスタンスに AMC をデプロイすることを選択した場合です。 ISC インスタンスは、IBM Security Directory Integrator に付属している組み込み ISC、またはターゲット・システムに既に常駐している ISC を指定できます。

AMC をカスタム SE にデプロイしている場合、HTTP ポートおよび HTTPS ポートに使用されるデフォルト値は、次のような方法で見つけることができます。

TDI_Selected_ISC/conf/overrides/*.properties ファイルで最初に com.ibm.pvc.webcontainer.port プロパティーと com.ibm.pvc.webcontainer.port.secure プロパティーが出現する場所を調べます。見つかればそれに割り当てられている値が求めているデフォルト値です。そのディレクトリー内のどの .properties ファイルでもこれらのプロ

パティーのいずれかが未定義だった場合は、*TDI_Selected_ISC*/conf/config.properties でプロパティーを探します。HTTP ポートが見つからなかった場合、デフォルトはポート 80 です。HTTPS ポートが見つからなかった場合、デフォルトはポート 443 です。 ヘルプ・ポートは、HTTP ポートと同じ値になります。

AMC をカスタム AE にデプロイしている場合、HTTP ポートおよび HTTPS ポートに使用されるデフォルト値は、次のような方法で見つけることができます。:

以下のようにディレクトリーを指定して、serverindex.xml という名前のファイルを探します。

TDI_Selected_ISC¥profiles¥AppSrv01¥config¥cells¥*¥nodes¥*

これらのファイル内で、次に類似した HTTP ポートの XML ブロックがないか調べます。

```
<specialEndpoints xmi:id="NamedEndPoint_1200476459036"
     endPointName="WC_adminhost">
 <endPoint xmi:id="EndPoint_1200476459036" host="*" port="9060"/>
</specialEndpoints>
```

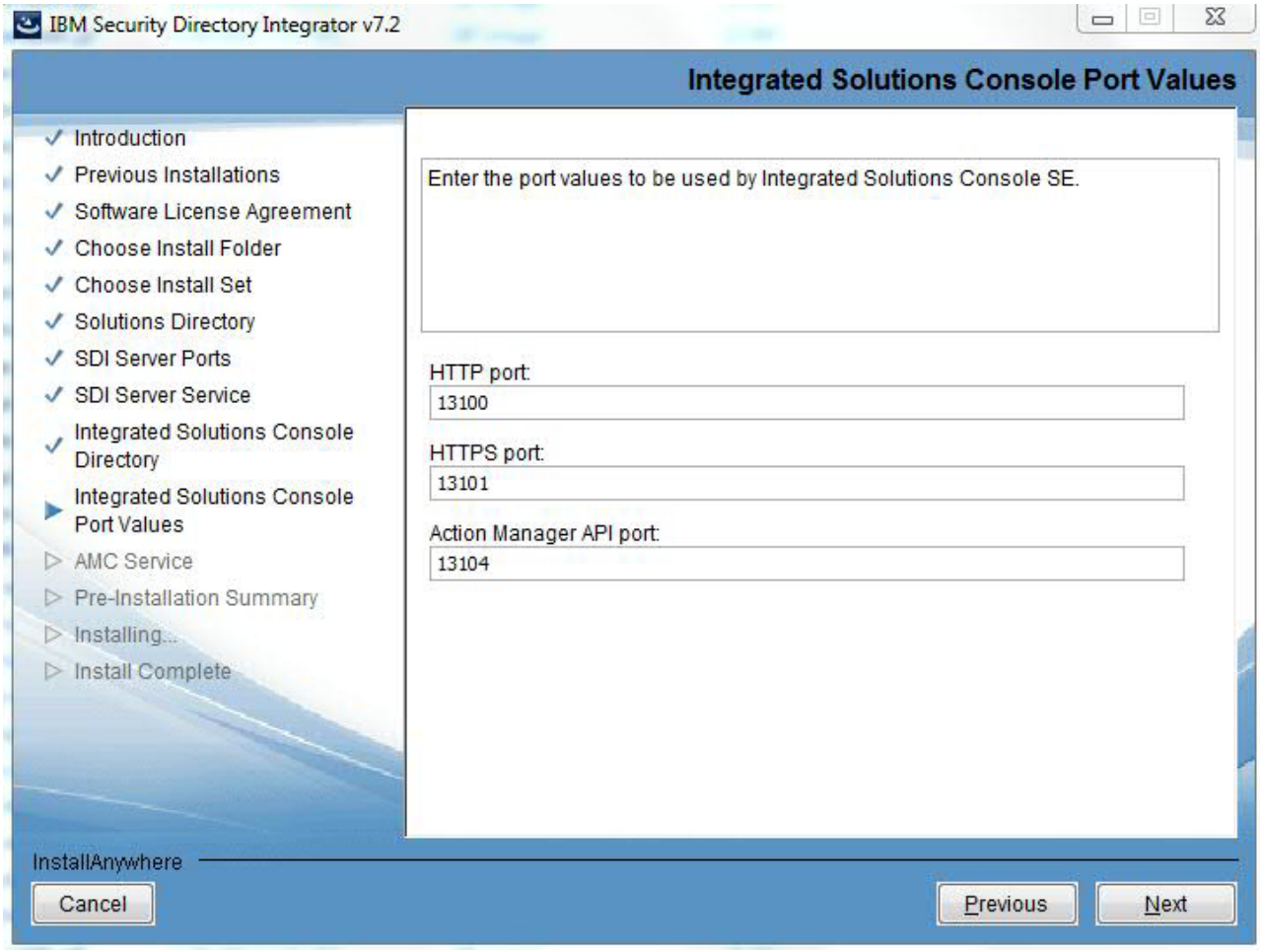
HTTPS ポートについても以下同様に調べます。

```
<specialEndpoints xmi:id="NamedEndPoint_1200476459039"
     endPointName="WC_adminhost_secure">
 <endPoint xmi:id="EndPoint_1200476459039" host="*" port="9043"/>
</specialEndpoints>
```

インストーラーは、endPointName に **WC_adminhost** または **WC_adminhost_secure** を持つ specialEnpoints タグを検索し、組み込み endPoint タグに関連付けられたポート値をデフォルト値として使用します。この方法で HTTP ポートを検出できなかった場合、デフォルトは 9060 です。またこの方法で HTTPS ポートを検出できなかった場合、デフォルトは 9043 です。ヘルプ・ポートには、HTTP ポートの値が設定されます。

表示される値は、組み込み SE のデフォルトです。

既に使用されているポートをパネルに入力することはできません。別のポート値の選択を要求する警告メッセージが表示されます。

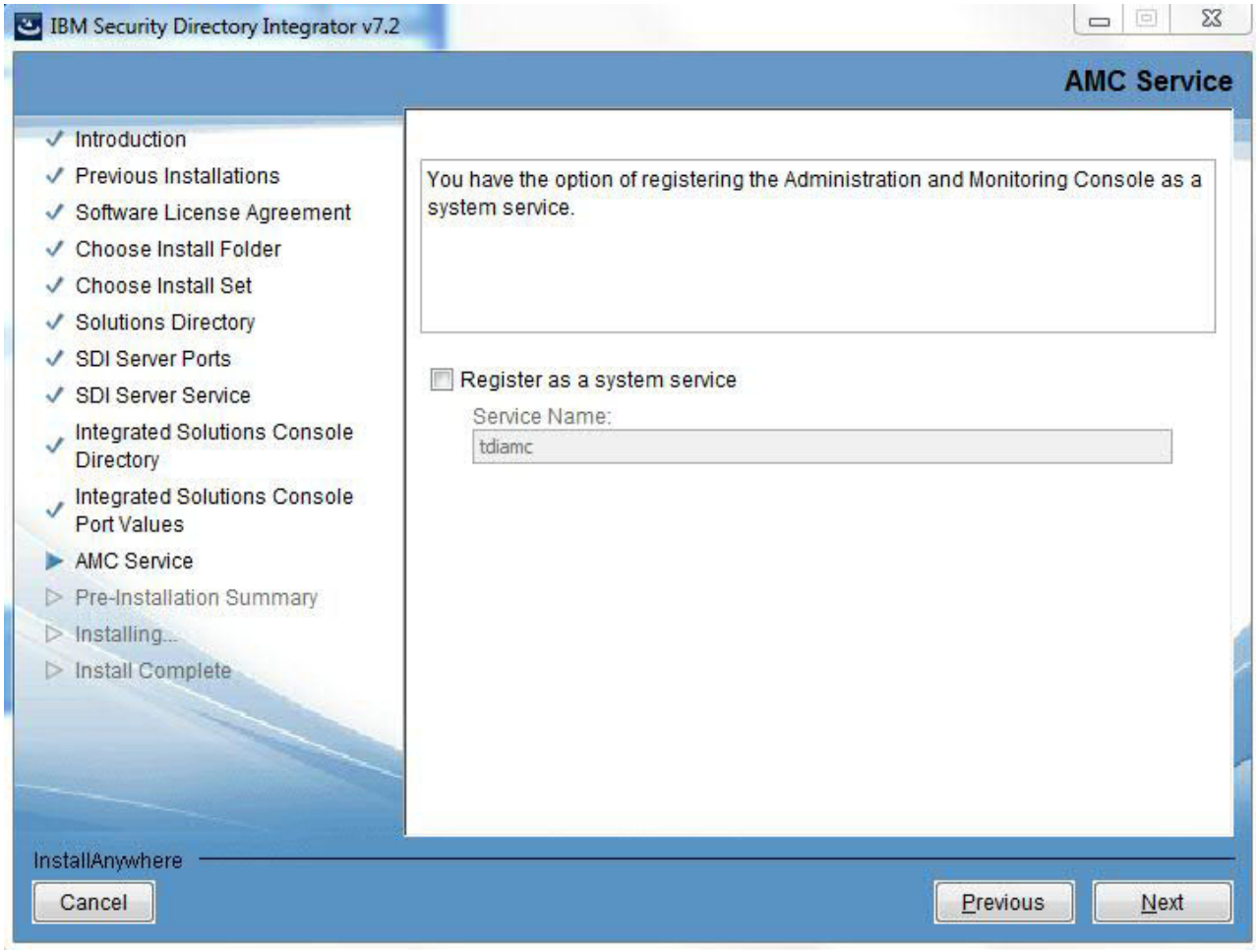


AMC をサービスとして登録するパネル

チェック・ボックスにチェックを付けると、AMC はその OS のサービスとして登録されます。

デフォルトでは、チェック・ボックスのチェックは外されています。

このパネルが表示されるのは、組み込み Web プラットフォームと AMC 機能が選択されていて、ユーザーが管理者特権を持っている場合のみです。

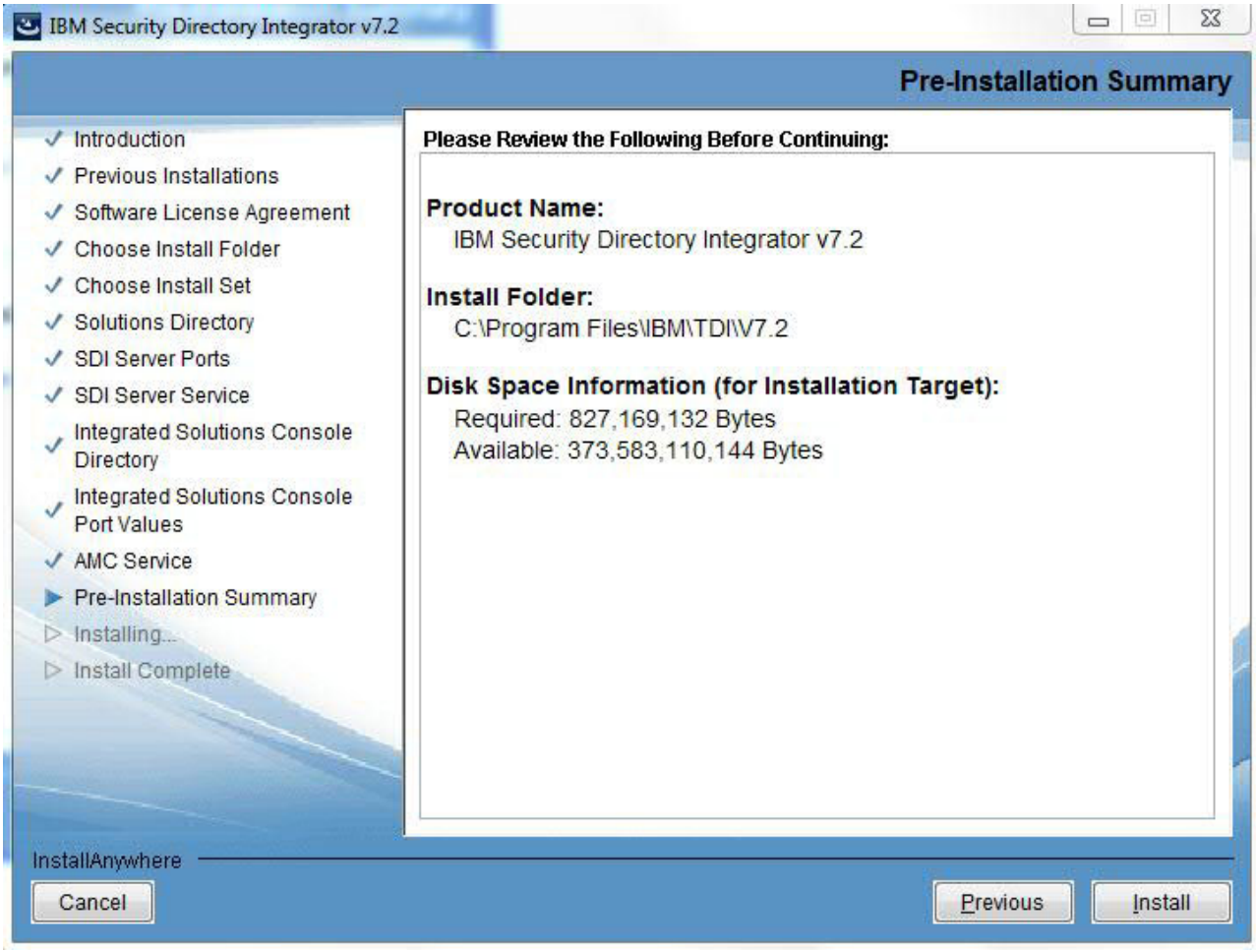


インストーラーは、このサービス名に有効なデフォルト値を与えるために最大限の努力を払います (このプロセスについての詳細は、『AMC を Windows サービスまたは Unix プロセスとして登録する』を参照してください)。インストーラーが有効なサービス名を判別できない場合、このフィールドはブランクになります。有効なサービス名を入力するまで、先に進めません。

注: UNIX システムにインストールする場合は、サービス名が最大長の 4 文字を超えないようにしてください。4 文字を超える場合、この制限がエラーとなり、インストールを続行できません。
AMC 機能は推奨されません。IBM Security Directory Integrator の将来のバージョンでは除去されます。

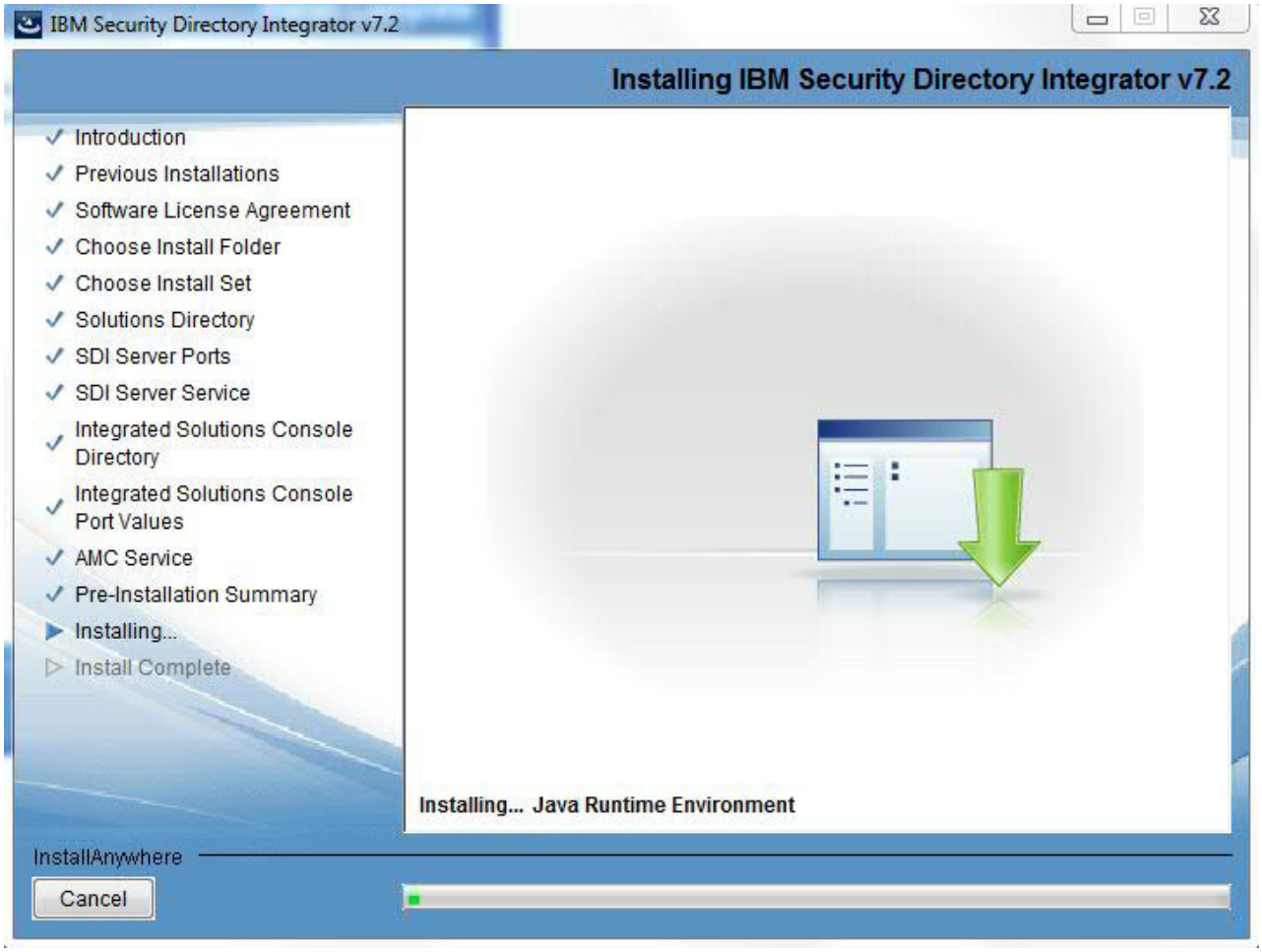
プリインストール・サマリー・パネル

この要約パネルには、インストールされる機能とそれらがインストールされる場所についての要約が表示されます。



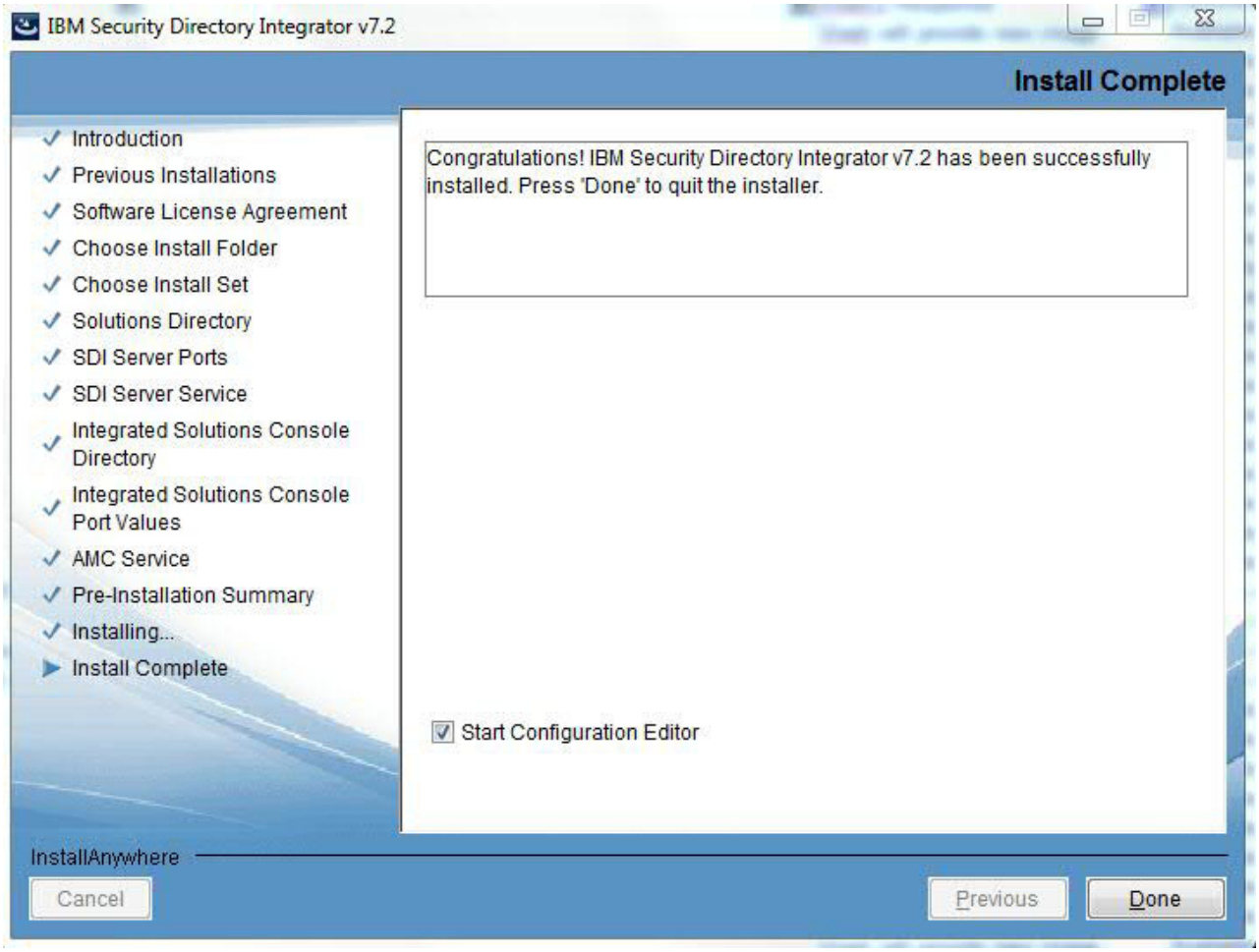
インストール進行状況パネル

このパネルは、実際にインストールが進行している場合に表示されます。このパネルは、InstallAnywhere で提供されている進行状況パネルです。すべての機能は、このパネルが表示されているときにインストールされます。



インストール完了パネル

このパネルはインストールが正常に完了したことを示します。「完了」ボタンを押すと、インストールは完了します。「**構成エディターの開始**」にはデフォルトでチェックが付いています。

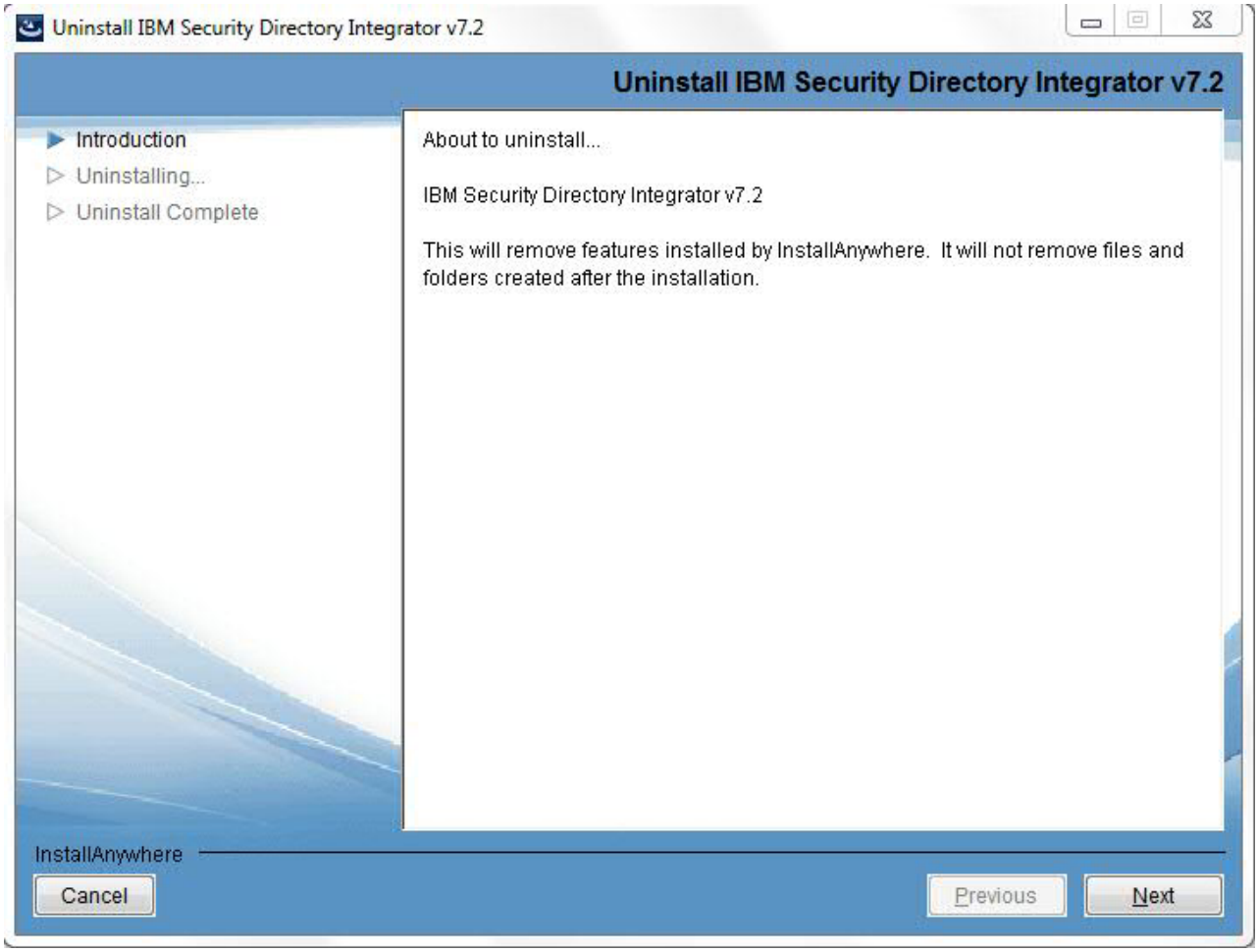


アンインストール・パネル・フロー

ここで提供する説明を参照して、パネル・フローをアンインストールできます。

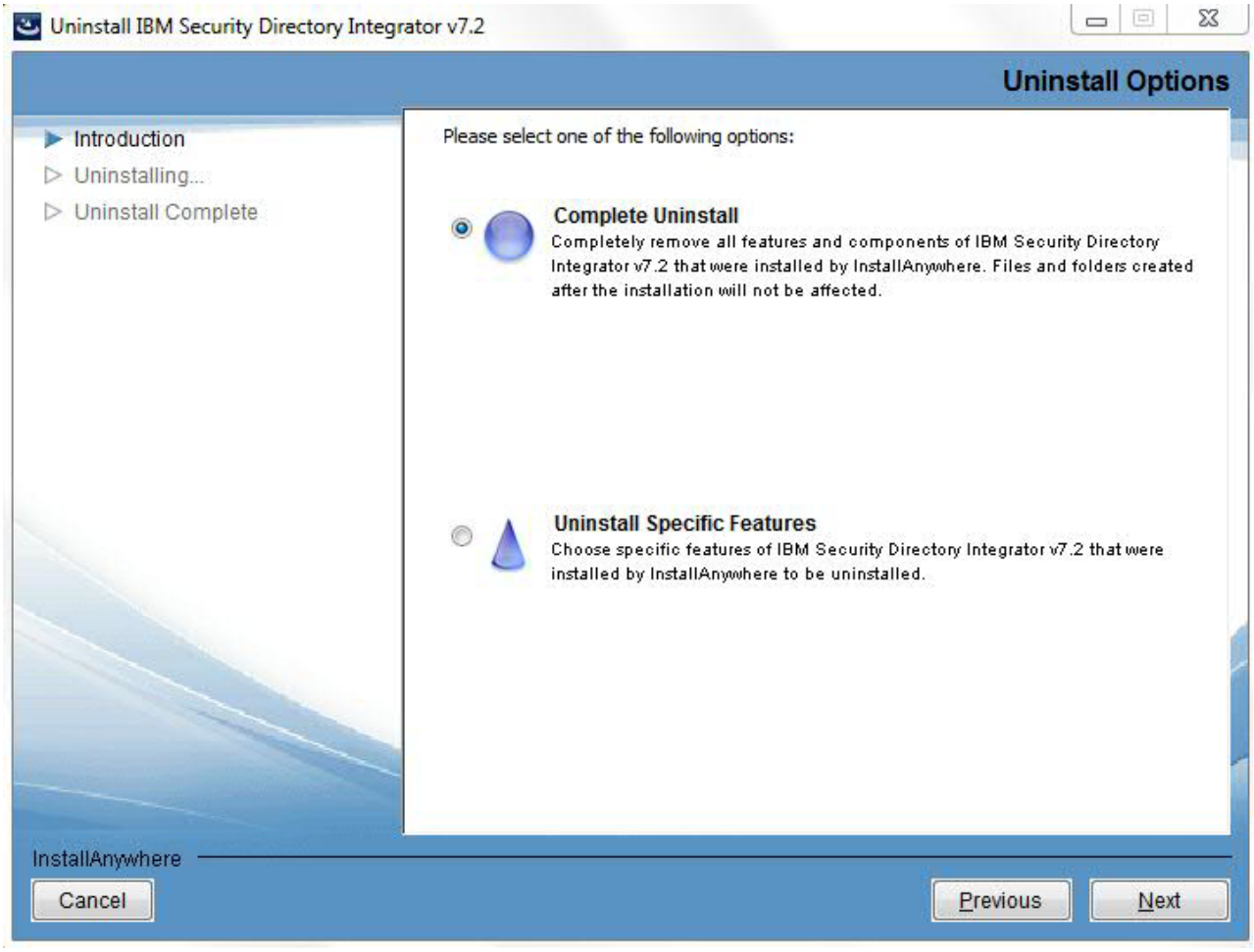
アンインストールの「ようこそ」パネル

これは標準的な内容の、InstallAnywhere のパネルです。

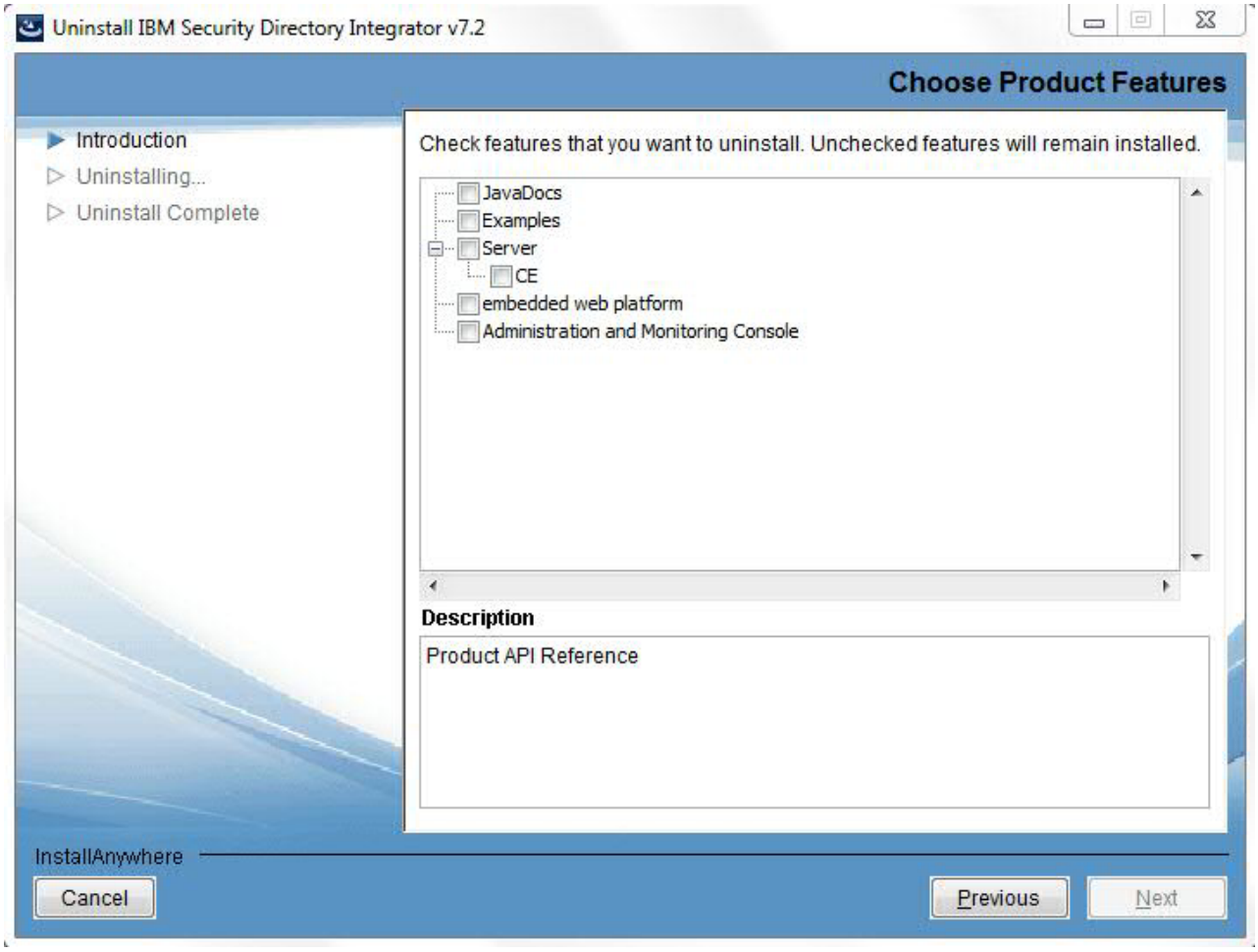


製品の機能の選択パネル

このパネルを使用して、製品すべて、または特定の機能のみのアンインストールを選択できます。

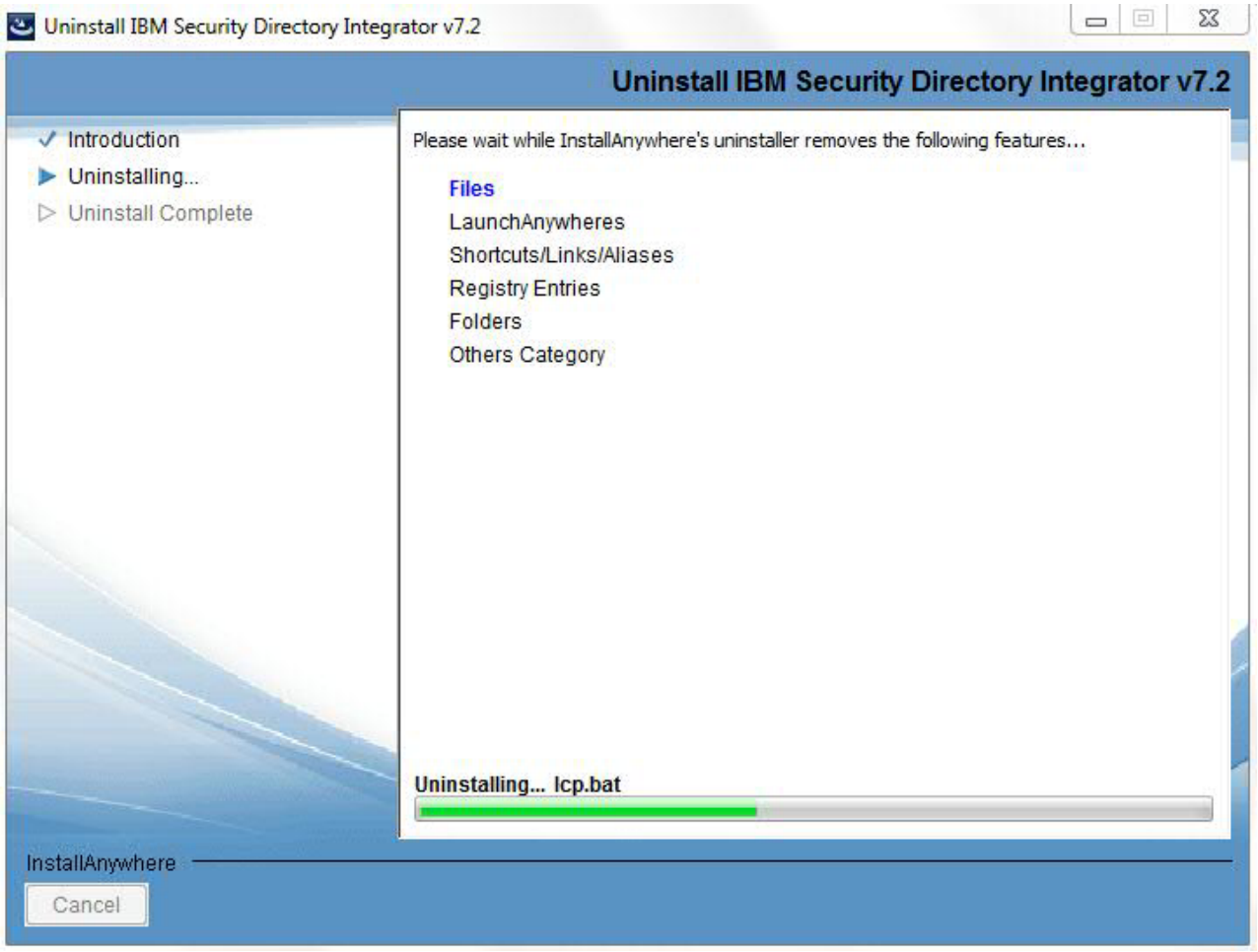


「**特定の機能のアンインストール**」を選択すると、次のパネルも表示されます。



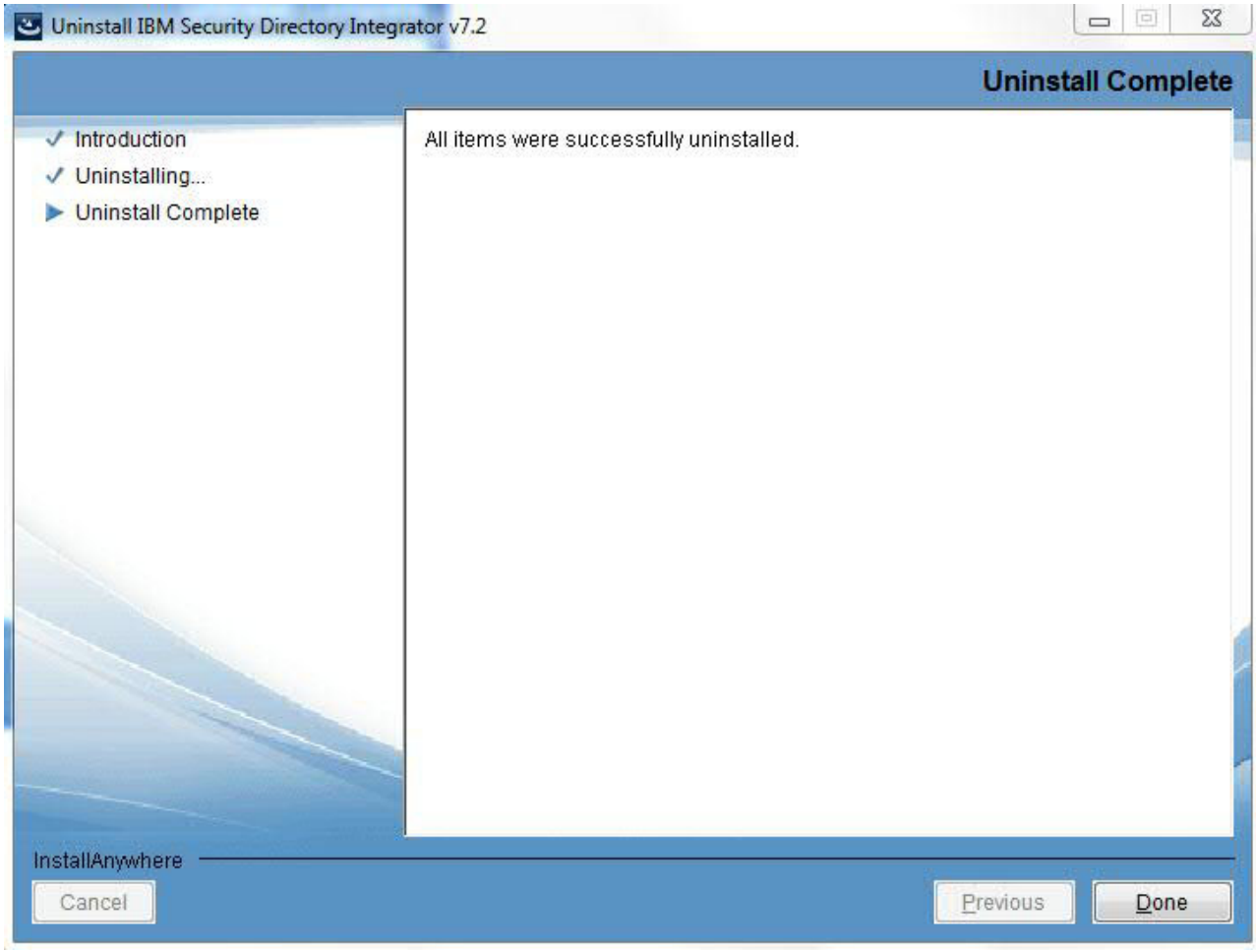
アンインストール進行状況パネル

このパネルはアンインストール時に表示されます。



アンインストール終了パネル

このパネルはアンインストールが正常に完了したことを示します。「完了」ボタンを押すと、アンインストーラーは終了します。



機能追加パネル・フロー

以下の情報を参照することで、機能追加のフローについて詳しく知ることができます。

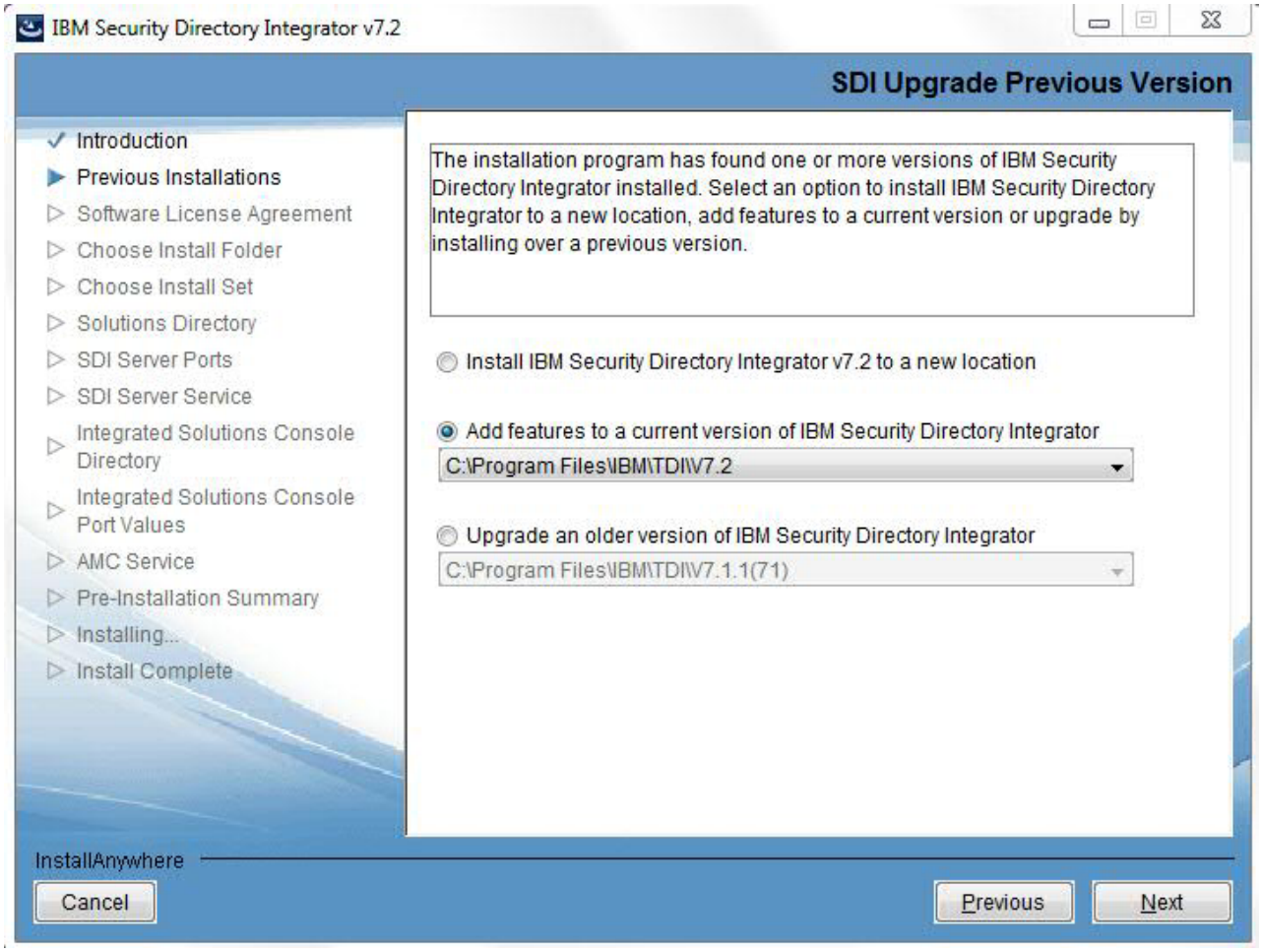
機能追加のフローは、新規インストールのフローに似ています。ここでは固有のパネルのみを示します。

事前初期設定パネル

「ようこそ」パネル

アップグレード・パネル

ボックスに既にインストールされている IBM Security Directory Integrator のインスタンスが存在する場合、「ようこそ」パネルと以前の IBM Security Directory Integrator 情報パネルの後に、このパネルが表示されます。



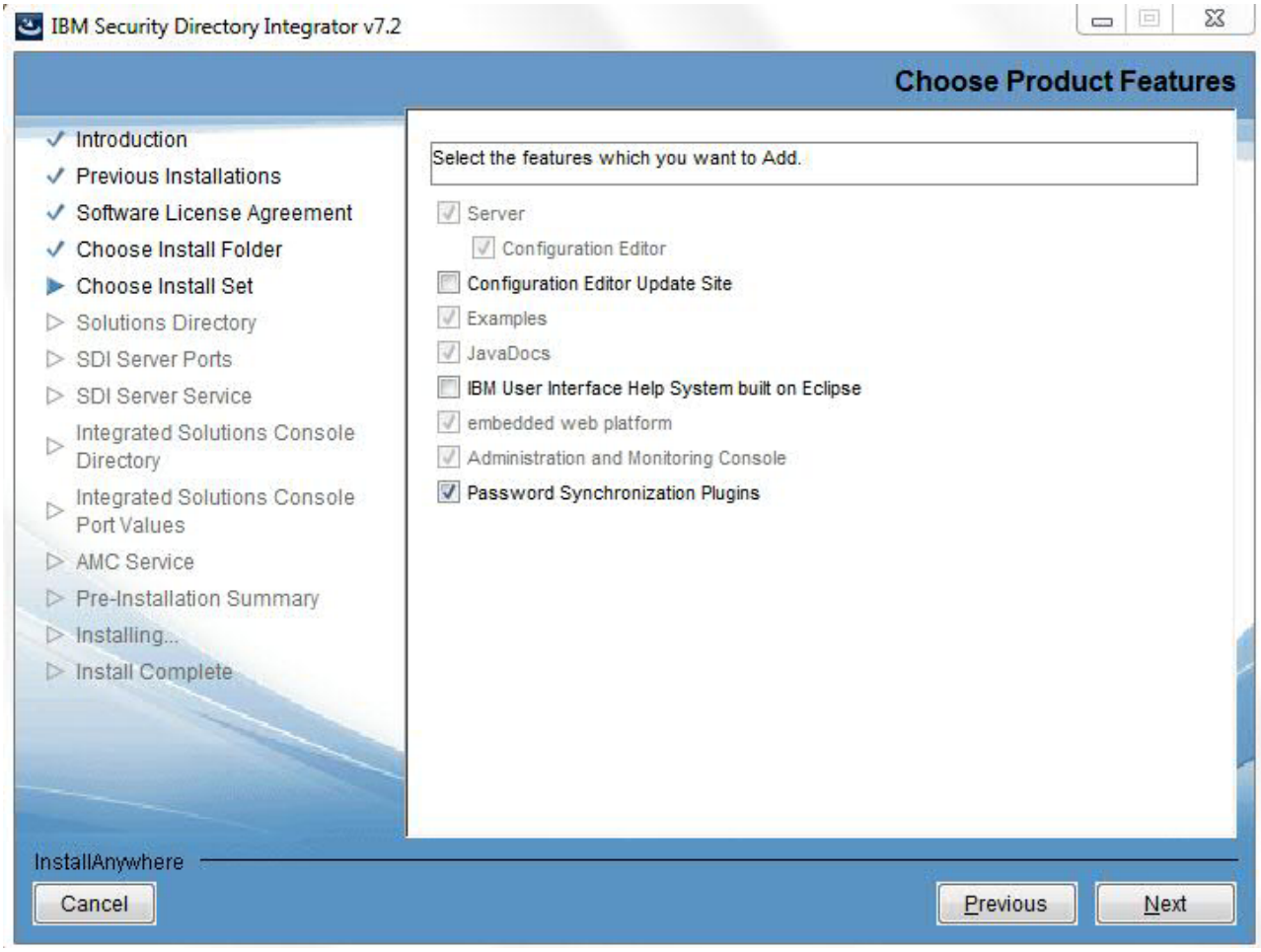
使用可能な IBM Security Directory Integrator バージョン 7.2 インスタンスが存在しない場合、「**機能追加**」ボタンは選択できません。

使用可能な以前のバージョンの IBM Security Directory Integrator が存在しない場合、「**アップグレード**」ボタンは選択できません。

「**機能追加**」ボタンを選択すると、IBM Security Directory Integrator ドロップダウンが使用可能になります。

機能選択パネル

一連の機能追加手順の中で次のパネルは、「機能選択」パネルです。既にインストール済みの機能は選択済みでグレイ表示になっています。



この時点で、さらに選択した機能を追加できます。

機能を削除することはできません。

この時点から、パネル・フローは新規のインストールのフローに一致します。ただし、既にインストール済みの機能に関連したパネルはスキップされます。

構成エディターを選択する場合、サーバーも自動的に選択されます。また両方の機能を選択し、サーバーの選択を解除した場合は、構成エディターも選択解除されます。

Security Directory Integrator の「ソリューション・ディレクトリー」パネル

サーバーをシステム・サービスとして登録するパネル

Security Directory Integrator の AMC デプロイメント・パネル

AMC をサービスとして登録するパネル

プリインストール・サマリー・パネル

インストール進行状況パネル

インストール完了パネル

マイグレーション・パネル・フロー

以下の情報を参照することで、マイグレーション・パネル・フローについて詳しく知ることができます。

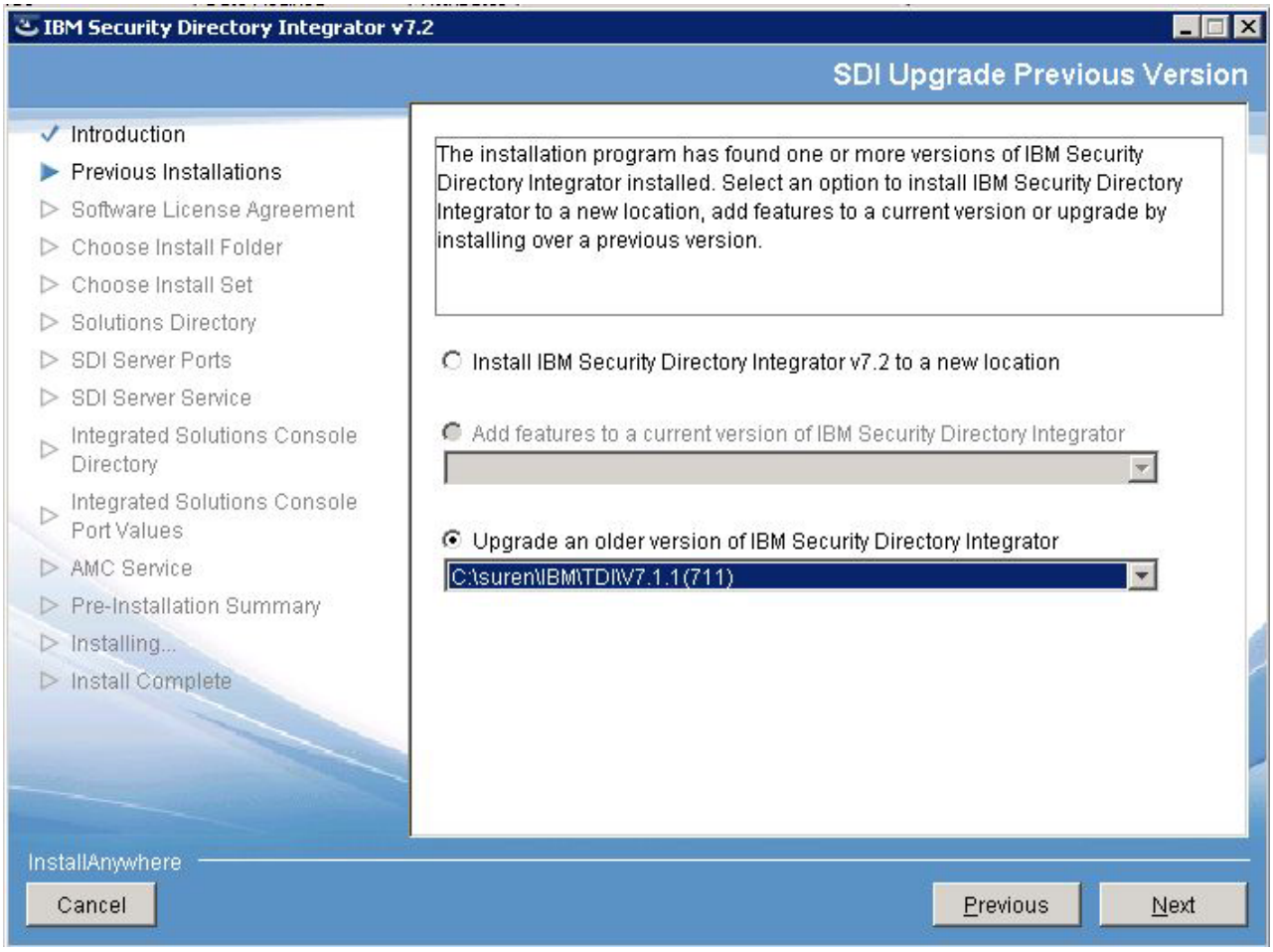
マイグレーションのフローは、新規インストールのフローに似ています。ここでは固有のパネルのみを示します。

事前初期設定パネル

「ようこそ」パネル

アップグレード・パネル

注: Windows 2012 オペレーティング・システムで IBM Security Directory Integrator バージョン 7.1.1 からバージョン 7.2 にアップグレードする予定の場合は、バージョン 7.1.1 の uninstaller.exe で **Windows 7 互換モード**が有効になっていることを確認してからバージョン 7.2 のインストーラーを開始してください。詳しくは、http://www-01.ibm.com/support/docview.wss?uid=swg21634336 に記載されている技術情報を参照してください。
IBM Security Directory Integrator のインスタンスがボックスに既にインストールされている場合、「ようこそ」パネルと旧 SDI 情報パネルの後に、このパネルが表示されます。



使用可能な IBM Security Directory Integrator バージョン 7.2 インスタンスが存在しない場合、「機能追加」ボタンは選択できません。

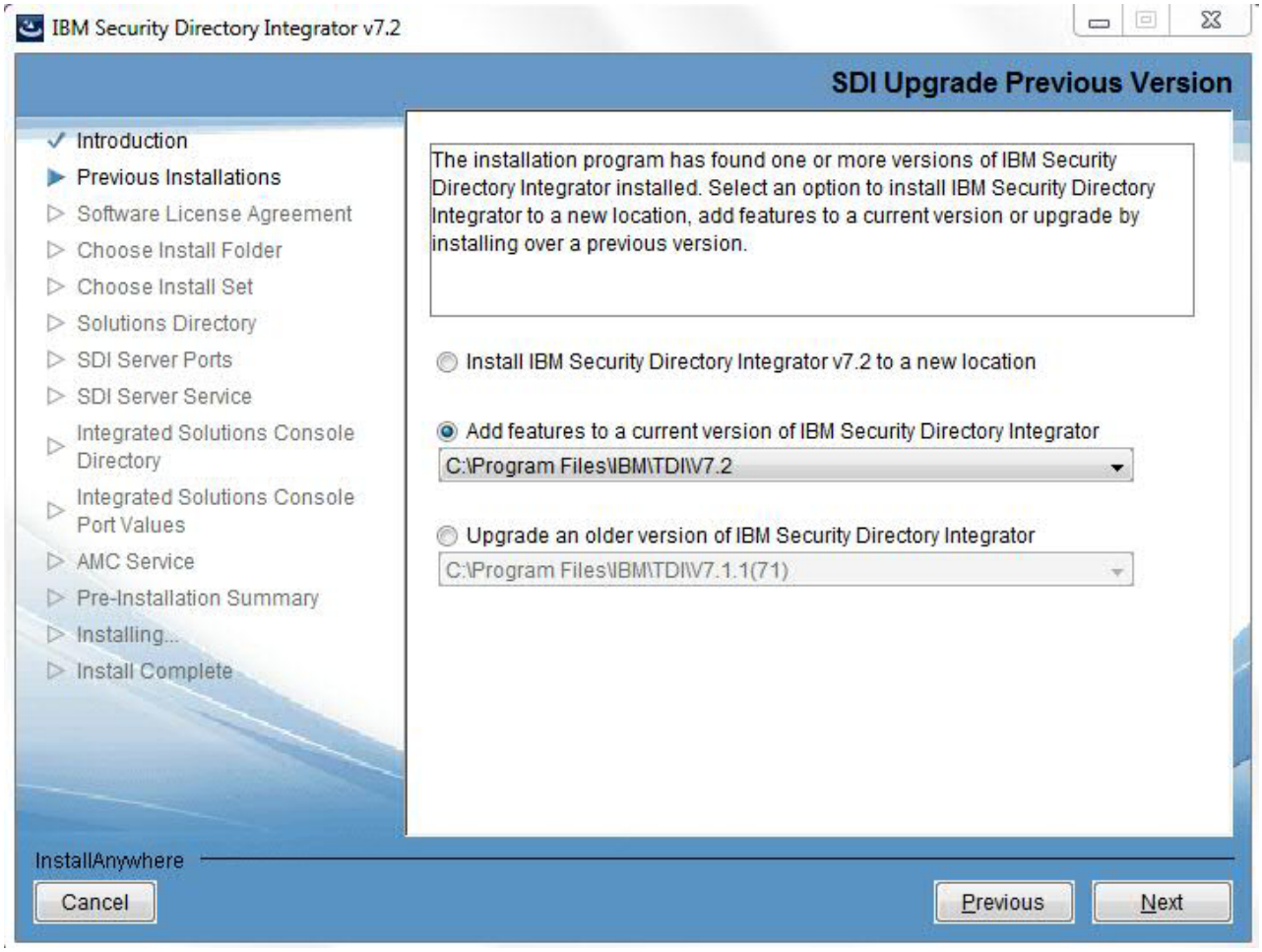
使用可能な以前のバージョンの IBM Security Directory Integrator が存在しない場合、「アップグレード」ボタンは選択できません。

「アップグレード」ボタンを選択すると、以前の IBM Security Directory Integrator バージョンのドロップダウンが使用可能になります。

IBM Security Directory Integrator 6.1.x、7.0、7.1.x のアップグレード・フロー

注: IBM Security Directory Integrator をバージョン 6.x 以前からバージョン 7.2 に直接アップグレードすることはサポートされていません。まずバージョン 6.x をバージョン 7.1.1 にアップグレードしてから、バージョン 7.1.1 をバージョン 7.2 にアップグレードする必要があります。

IBM Security Directory Integrator バージョン 7.1.1 からのアップグレードを選択すると、次に表示されるパネルはライセンス・パネルとなり、ライセンスを受け入れた後は、機能選択パネルになります。



このパネルでは、以前のバージョンでインストールされていたこれらの機能が表示され(選択済み)、新しい機能の追加が可能です。以前からインストールされていた機能を使用不可にすることは許されていません。

サーバーが既にインストールされている場合、SDI ソリューション・ディレクトリー・パネルはスキップされます。インストーラーは以前のインストールの値を使用します。

IBM Security Directory Integrator 6.x.x マイグレーションでは、AMC が前にインストールされている場合、ISC の選択パネルが引き続き表示されます。

残りのパネル・フローは、新規インストールの場合と同じです。IBM Security Directory Integrator 7.0 マイグレーションでは、1 つの新しい機能があります。この「**サーバーをシステム・サービスとして登録する**」パネルは、機能選択パネルの後でのみ表示されます。

構成エディターを選択する場合、サーバーも自動的に選択されます。また両方の機能を選択し、サーバーの選択を解除した場合は、構成エディターも選択解除されます。

コマンド行を使用したインストール

以下に、インストール中に使用できるコマンドのリストを示します。

IBM Security Directory Integrator インストーラーでは、以下のコマンド行オプションがサポートされています。

-i インストーラーのインターフェース・モード (silent、console、または gui) を設定します。

```
install_sdiv72_win_x86.exe -f Response File Name -i silent

install_sdiv72_win_x86.exe –i console
```

-f インストーラーが使用する応答ファイル (installer.properties ファイル) の場所を設定します。

```
install_sdiv72_win_x86.exe –f installer.properties
```

このパスは、絶対パスまたは相対パスのどちらでも指定できます。(相対パスは、インストーラーの場所からの相対関係を示します。)

-r 応答ファイルを作成します。

```
install_sdiv72_win_x86.exe –r myinstaller.properties
```

注: *-r* オプションを指定しない場合でも、IBM Security Directory Integrator インストーラーによって、システムの一時ファイル・ディレクトリーに tdi_respfile72.txt 応答ファイルが作成されます。例を示します。

- Windows プラットフォームでは、応答ファイルは C:¥DOCUME~1¥ADMINI~1¥LOCALS~1¥Temp ディレクトリーに作成されます。
- Windows 以外のプラットフォームでは、応答ファイルは /tmp ディレクトリーに作成されます。

TDI_install_dir/examples/install ディレクトリーには、さまざまなインストールとアンインストールのシナリオに対応したサンプルの応答ファイルがあります。

-D カスタムのコマンド行引数を渡します。

```
install_sdiv72_win_x86.exe -Dmyvar=myvalue
```

-l 指定した言語コード (およびオプションで国別コード) を使用して、InstallAnywhere インストーラーのロケールを設定します。

```
install_sdiv72_win_x86.exe -l en

install_sdiv72_win_x86.exe -l pt_BR
```

必要な言語コードは、ISO-639 標準で定義された 2 文字 (通常、小文字) のコードです。InstallAnywhere は、古い言語コード (iw、ji と in)、および新しい言語コード (he、yi と id) の両方を受け入れます。

オプションの国別コードは、ISO-3166 標準で定義された 2 文字 (通常、大文字) のコードです。

ロケール・オプションが影響するのは、インストーラーにユーザーが指定したロケールのローカリゼーションが含まれる場合のみです。

-? InstallAnywhere インストーラーのヘルプを表示します。

Windows では、-help はコンソール・ランチャーからのみ機能します。LaunchAnywhere には、「**プロジェクト**」>「**プラットフォーム**」サブタスクの「Windows」タブで、必ず「**コンソール**」を設定してください。(インス

トール済みの LaunchAnywhere でこの情報を得るには、アクションで明示的に LaunchAnywhere をコンソール・ランチャーに設定する必要があります。)

以下のコマンド行オプションは、IBM Security Directory Integrator のインストール・ウィザードに固有です。

LAX_VM

LAX_VM パラメーターは、システムにインストールされている Java 仮想マシンからインストーラーをブートするために使用します。

Java の bin ディレクトリーにある Java 実行可能ファイルの絶対パスを指定する必要があります。以下に例を示します。

```
install_sdiv72_win_x86.exe LAX_VM "Java_DIR/jre/bin/java.exe"
```

各引数の間にはスペース文字のみを使用します。

注: パラメーター値として IBM JRE 7.0.4 以降の絶対パスを必ず使用してください。その他の JRE では、IBM Security Directory Integrator インストーラーが正しく動作しない場合があります。

-DTDI_BACKUP="true"

このパラメーターは、アンインストール時にのみ指定します。このパラメーターは、将来のマイグレーションを考慮して提供されています。例えば以下のようにします。

```
TDI_install_dir¥_uninst¥uninstaller.exe –D$TDI_BACKUP$="true"
```

これはアンインストーラーに対して、*TDI_install_dir*/bin/tdiBackup.bat (.sh) スクリプトを実行することを指示しています。その結果、ディレクトリー *TDI_install_dir*/backup_tdi が作成されることになります。グローバル・プロパティー・ファイル、グローバル証明書など、インストールに関係したいくつかのファイルのバックアップは、このディレクトリーに保管されます。

注: Windows 以外のシステムでは、$ (ドル) を ¥ (円記号) でエスケープする必要があります。例を示します。

```
TDI_install_dir¥_uninst¥uninstaller -D¥$TDI_BACKUP¥$="true"
```

-D$TDI_SKIP_VERSION_CHECK$=「true」

このパラメーターは、インストーラーに対して、以前のバージョンのチェックをスキップするよう指示します。それにより、以前のリリースからのマイグレーションは不可能になります。

サイレント・インストールの場合、このスキップ・オプションを選択し、インストール・ディレクトリーが IBM Security Directory Integrator の以前のインストールと同じだった場合は、インストーラーが停止する原因になります。

注: Windows 以外のシステムでは、$ (ドル) 記号を ¥ (円記号) でエスケープする必要があります。例を示します。

```
./install_sdiv72_linux_x86_64.bin -D¥$TDI_SKIP_VERSION_CHECK¥$="true"
```

-D$TDI_NOSHORTCUTS$="true"

このパラメーターを使用して、インストーラーがアンインストーラー、CE、または AMC へのショートカットを作成しないようにします。

注: Windows 以外のシステムでは、$ (ドル) を ¥ (円記号) でエスケープする必要があります。例を示します。

```
./install_sdiv72_linux_x86_64.bin -D¥$TDI_NOSHORTCUTS¥$="true"
```

インストール時の一時ファイル・スペースの使用量

ここで説明する手順に従って、一時ファイル・ストレージを活用します。

インストール時には、インストーラーは、ファイルを実行するため、大量の一時ファイル・スペースを使用します。ご使用のシステムでファイル・スペースに制限がある場合は、インストール時にエラーが発生することがあります。

UNIX/Linux システムでは、通常、`/tmp` または `/var/tmp` を一時ファイル・ストレージとして使用します。一方、 Windows では、一時ファイル・ストレージ領域は、環境変数 TEMP で指示される場所にあります。

InstallAnywhere インストーラーを始動する前に環境変数 IATEMPDIR を設定すると、インストーラーに対して使用する一時ファイルをリダイレクトするように指示することができます。例えば、UNIX では次のコマンドを使用します。

```
export IATEMPDIR=/opt/IBM/TDI/temp
```

その後、IATEMPDIR 変数を設定したセッションから、コンソール・モードのインストーラーを始動します。

サイレント・インストールの実行

以下の説明に従って、サイレント・インストールを実行できます。

サイレント・インストールを実行するには、まず応答ファイルを生成する必要があります。 このファイルを生成するには、次のように `-r` オプションを指定して、非サイレント・インストールを実行します。

```
install_sdiv72_win_x86.exe -r Response File Name
```

インストール時に指定するディレクトリーに応答ファイルが作成されます。

注: *TDI_install_dir*`/examples/install` ディレクトリーには、さまざまなインストールとアンインストールのシナリオに対応した多数のサンプル応答ファイルがあります。
応答ファイルが作成されると、以下のコマンドを使用して、サイレント・モードでインストールできます。

```
install_sdiv72_win_x86.exe –i silent –f Response File Name
```

注: 本書の例では、Windows プラットフォームのインストールの実行可能ファイルを使用しています。サポートされる各プラットフォームの実行可能ファイル名のリストについては、 10 ページの『適切なインストーラーの起動』を参照してください。

UNIX システムでのサービス名の制限

UNIX システムでサービスを命名する際の制限事項について検討します。

UNIX システムで IBM Security Directory Integrator のサイレント・インストールを行う場合は、IBM Security Directory Integrator のサービス名および AMC が最大長の 4 文字を超えないようにしてください。この値は、応答ファイル examples¥install¥TDICustomInstallRsp_Unix.txt に指定します。

以下のような 4 文字以上の名前を指定すると、サイレント・インストールが失敗します。

```
TDI_SERVER_SERVICENAME=tdisrv_silent
TDI_AMC_SERVICENAME=tdiamc_silent
```

ログ・ファイル /tmp/sdiv72install.log および /tmp/sdiv72debug.log は、以下のエラーをレポートします。

```
tdisrv_silent は 4 文字以下とする必要があります。
その名前を持つサービスがすでに存在しているか、または名前が無効です。
UNIX ベースのプラットフォーム、文字長が最大 4 文字の名前は有効です。
```

このエラーが発生した場合、応答ファイルを編集して、TDI_SERVER_SERVICENAME プロパティーと TDI_AMC_SERVICENAME プロパティーの値を 4 文字以下に変更する必要があります。

インストール後の手順

インストールが完了したら、ここにリストされている手順を実行します。

CE 更新サイト

以下の情報を使用して、 Eclipse を手動でデプロイします。

CE 更新サイトがインストールされた場合、この段階で、手動で Eclipse にデプロイする必要があります。詳細については、53 ページの『Eclipse 更新マネージャーを使用したインストールと更新』セクションを参照してください。

プラグイン

パスワード同期プラグインに関する資料にアクセスする場合は、以下の情報を参照してください。

パスワード同期プラグインのいずれかがインストールされている場合は、プラグイン・コードのデプロイ方法について、IBM Security Directory Integrator の IBM Knowledge Centerの『*パスワード同期プラグイン*』セクションを参照してください。

管理およびモニター・コンソール (Administration and Monitoring Console) (AMC)

ここでは、管理およびモニター・コンソールの一般情報、Web プラットフォームのデプロイメントおよび遅延デプロイメントに関する情報の詳細を説明します。

一般情報

- AMC について詳しくは、 48 ページの『管理およびモニター・コンソール (Administration and Monitoring Console) (AMC)』を参照してください。
- コンソールにログインする準備ができている場合は、http://hostname:port/ibm/console を参照してください。詳しくは、セクション 293 ページの『コンソールへのログインとログアウト』を参照してください。
- ユーザーおよびユーザーの役割の追加について詳しくは、 277 ページの『Integrated Solution Console の AMC』の『『コンソールのユーザー権限』』セクションを参照してください。

バンドルされた組み込み Web プラットフォームのデプロイメント

- バンドルされた組み込み Web プラットフォームに AMC をインストールし、AMC を使用する準備ができたら、ISC コンソールにログインする前に AMC と Action Manager (AM) を始動するため、複数のコマンドを実行する必要があります。詳しくは、セクション 273 ページの『AMC および Action Manager の開始とログイン』を参照してください。

  注: Windows では、スタート・メニューの「すべてのプログラム」に launchAMC.html ファイルへのショートカットが作成されます。
- デフォルトでは、IBM Security Directory Integrator をインストールしたユーザーが、コンソールにログイン・アクセスできる唯一のユーザーになります。

カスタムまたは遅延デプロイメント

- AMC のデプロイ先としてカスタムを選択して、デプロイする準備ができている場合は、 52 ページの『カスタム ISC SE または IBM Dashboard Application Services Hub への AMC のデプロイ』を参照してください。この方法で AMC をデプロイすると、インストーラーは現在のユーザーに SDI AMC Admin 役割を自動的には割り当てません。この権限は ISC コンソールの管理者が手動で許可する必要があります。これは通常、IBM Dashboard Application Services Hub コンソールの「**ユーザーおよびグループ**」->「**管理者ユーザーのロール**」パネルを使用して行います。あるいは、この役割は `setAMCRoles` コマンドを使用して割り当てることもできます。
- AMC を ISC に遅延デプロイメントをすることを選択しているとき、その準備が完了した場合は、 52 ページの『カスタム ISC SE または IBM Dashboard Application Services Hub への AMC のデプロイ』を参照してください。

  注: カスタムの ISC SE/AE デプロイメントを完了したら、AMC をインストールした ISC SE/AE を始動した後、少なくとも、AM が始動されたことを確認する必要があります。

資料

文書にオンラインでアクセスしたり、文書を手動でデプロイするよう選択できます。詳しくは、以下の情報を参照してください。

IBM Security Directory Integrator で使用されている文書システムは IBM Knowledge Center です。つまり、デフォルトのインストールが完了すると、IBM Security Directory Integrator 文書は IBM がホストする Web 上でオンラインで使用可能になります。ただし、文書はローカルにデプロイすることも選択できます。詳しくは、50 ページの『ローカル・ヘルプ・ファイルのインストール』を参照してください。

初めて IBM Security Directory Integrator を使用する場合は、使用されている概念に慣れるため、「*始めに*」を一通り読むことをお勧めします。

以前のバージョンの IBM Security Directory Integrator を使用していた場合は、新しい IDE フレームワークとレイアウトを理解する上で、「*Directory Integrator の構成*」のセクション 3 が非常に役立ちます。ここには、既存の構成をインポートして開く方法や、サーバーが従来同様、ランタイム時に構成モデルを使用する方法なども説明されています。

マイグレーション

マイグレーションの詳細については、以下のリンクを参照してください。

以前のバージョンの IBM Security Directory Integrator をインストールしていた場合、通常は以前のデプロイメントから特定の側面をマイグレーションする必要に迫られます。この作業についての詳細は、67 ページの『第 5 章 マイグレーション』を参照してください。

ローカル・ヘルプ・ファイルのインストール

以下の手順に従って、文書をローカルにインストールします。

注: この機能は推奨されません。IBM Security Directory Integrator の将来のバージョンでは削除されます。

IBM Security Directory Integrator インストーラーには、Java API 文書以外のユーザー向け資料は含まれていません。これは構成エディターの「**ヘルプ**」->「**ようこそ**」画面、「**JavaDocs**」リンクを選択することで表示できます。IBM は、ユーザー向け文書をIBM Security Directory Integrator の IBM Knowledge Centerでオンライン形式で提供します。

IBM Security Directory Integrator には、構成エディター (CE) からコンテキスト・オンライン・ヘルプを起動するためのコード[1]が備えられています。このコードは、デフォルトでは、上述の説明のようにオンライン製品資料の文書を処理します。ただし、ヘルプを参照する際にインターネットに依存せずに済むよう、文書をローカルにインストールすることもできます。

文書をローカルにインストールするには、次のステップを実行する必要があります。

- 文書ファイルを取り扱うためのコード (Eclipse を採用した IBM ユーザー・インターフェース・ヘルプ・システム) は、デフォルトではインストールされません。ヘルプ・システムをインストールするには、カスタム・インストールを使用して、既存の IBM Security Directory Integrator インストール済み環境にヘルプ・システム機能をインストールする必要があります。
- すべてのマニュアルは、1 つの圧縮ディレクトリーにまとめて保管されています。解凍したとき、このディレクトリーには Eclipse *Document plug-in* が含まれます。

1. このヘルプ・システムは Eclipse™ テクノロジーを利用して機能します (http://www.eclipse.org)。

- すべてのマニュアルは、IBM Security Directory Integrator の IBM Knowledge Centerから圧縮形式でダウンロードできます。現行リリースのウェルカム・ページで、「*詳細情報*」列にある「インフォメーション・センター・プラグイン」をクリックします。
- 文書のフルパッケージ (*di_plug-ins-7.2.0.1.zip*) は、正規の場所 (*TDI_install_dir*/ibm_help/eclipse/plugins フォルダー) に解凍する必要があります (あるいは、どこか別の場所で解凍し、正規の場所に移動します)。このパッケージでは、com.ibm.IBMDI.doc_7.2.0.1 に、実際の IBM Security Directory Integrator 文書が含まれているとともに、名前が .doc で終わるその他多数のディレクトリーも含まれています。これらのディレクトリーはすべて、前述の同一の *plug-ins* レベルにある必要があります。
- CE がアクセスする文書の位置は、global.properties ファイル (IBM Security Directory Integrator のインストール・ディレクトリーの etc フォルダーにあります)、または solutions.properties ファイル (ソリューション・ディレクトリーにあります) で設定します。デフォルトではオンラインの製品資料を指しています。以下の行をここに示すように変更すると、次に CE を実行してヘルプを起動したときにローカル・ヘルプ・システムが使用されます。

  ```
  com.ibm.di.helpHost=publib.boulder.ibm.com
  com.ibm.di.helpPort=80
  ```

  変更後:

  ```
  com.ibm.di.helpHost=localhost
  com.ibm.di.helpPort=9999
  ```

- AMC がアクセスを試みる文書サーバーの場所は、web.xml ファイルで設定されます。tdiamc webapp の WEB-INF フォルダーにある web.xml ファイルを開いて、*InfocenterHostName* 属性と *InfocenterPort* 属性のヘルプ・サーバーの IP アドレス (またはホスト名) とポートをリストします。

上記の説明に従って文書を *plug-ins* ディレクトリーにインストールした後は、そのマシン上の文書を、IBM Security Directory Integrator がインストールされているご使用の環境の他のコンピューターから使用できるように設定することもできます。それには、*TDI_install_dir*/ibm_help ディレクトリーにある一部の .bat ファイル (Windows) または .sh ファイル (Unix/Linux) を使用します。

ローカル資料タスクの開始および停止

資料を提供するには、最初にローカル・タスクを開始する必要があります。

IC_start.bat または IC_start.sh

このスクリプトを実行すると、このスクリプトによって、インフォメーション・センターが http://*your_IP_address*:9999 で始動されます。

このファイルを編集して、ポート番号をデフォルトの 9999 から変更できます。例えばこれを 80 に変更する場合は、「-port 9999」を「-port 80」に変更します。このインフォメーション・センターにアクセスするクライアントでは、global.properties または solution.properties ファイル内の com.ibm.di.helpPort – プロパティーで指定する番号がそのポート番号と一致している必要があります (最初はデフォルトとして 80 に設定されています)。また、com.ibm.di.helpHost プロパティーの値を、*infocenter_IP_address* のように指定してください (*infocenter_IP_address* は使用するローカル・インフォメーション・センターのアドレスです)。さ

らに、AMC がこのインフォメーション・センターを見つけられるようにするために、AMC の構成ファイル `web.xml` 内の *InfoCenterHostname* および *InfoCenterPort* 属性のパラメーターを、上記の値に一致するよう更新する必要があります。

IC_stop.bat または IC_stop.sh
ローカル・インフォメーション・センターのサービスを提供するヘルプ・システム (Java プログラム) を停止します。

help_start.bat または help_start.sh
使用するポートがランダムである点を除いて、IC_start と似ています。また、開始ページを表示するローカル・ブラウザーを起動することもできます。ポートがランダムであるため、ローカル・コンピューター以外で使用するには不適当です。

help_stop.bat または help_stop.sh
help_start によって開始されたローカル Java タスクを停止します。

カスタム ISC SE または IBM Dashboard Application Services Hub への AMC のデプロイ

AMC の遅延デプロイメントを行う場合は、以下に示されている手順を使用します。

AMC の ISC への遅延デプロイメントを選択した場合、デプロイの準備が完了したら、以下の手順を実行します。

- 以下のスクリプトを実行します。

  TDI_install_dir/bin/setISCHome.bat(sh) *ISC ロケーション*
  TDI_install_dir/bin/amc/install.bat(sh)
  TDI_install_dir/bin/amc/setAMCRoles.bat(sh) *ユーザー名*

  注:

  1. SE と AE のいずれの場合でも、`setAMCRoles` スクリプトの呼び出しは任意です。スクリプトを実行する場合、ユーザー名 は、ISC/IBM WebSphere Application Server 環境に既に存在するユーザー名である必要があります。あるいは、ISC コンソール (具体的には、「コンソール・ユーザーの権限」パネル) を使用して、AMC に付属している役割 (「SDI AMC Admin」および「SDI AMC User」) の 1 つを手動でユーザーに割り当てることもできます。
  2. AMC の役割についての詳細は、277 ページの『Integrated Solution Console の AMC』を参照してください。

- `amc.properties` ファイルを変更し、am.api.port と amc.help.port を指定している行に適切なポート値を割り当てます。このファイルは、ISC SE の場合は *ISC location*/runtime/isc/eclipse/plugins/AMC_7.2.0.0/ にあり、IBM Dashboard Application Services Hub の場合は *ISC location*/systemApps/isclite.ear/tdiamc.war にあります。

AMC のデプロイ先としてカスタム IBM Dashboard Application Services Hub を選択し、デプロイする準備ができている場合は、以下の手順に従ってください。

- 以下のスクリプトを実行します。

TDI_install_dir/bin/amc/setAMCRoles.bat(sh) ユーザー名

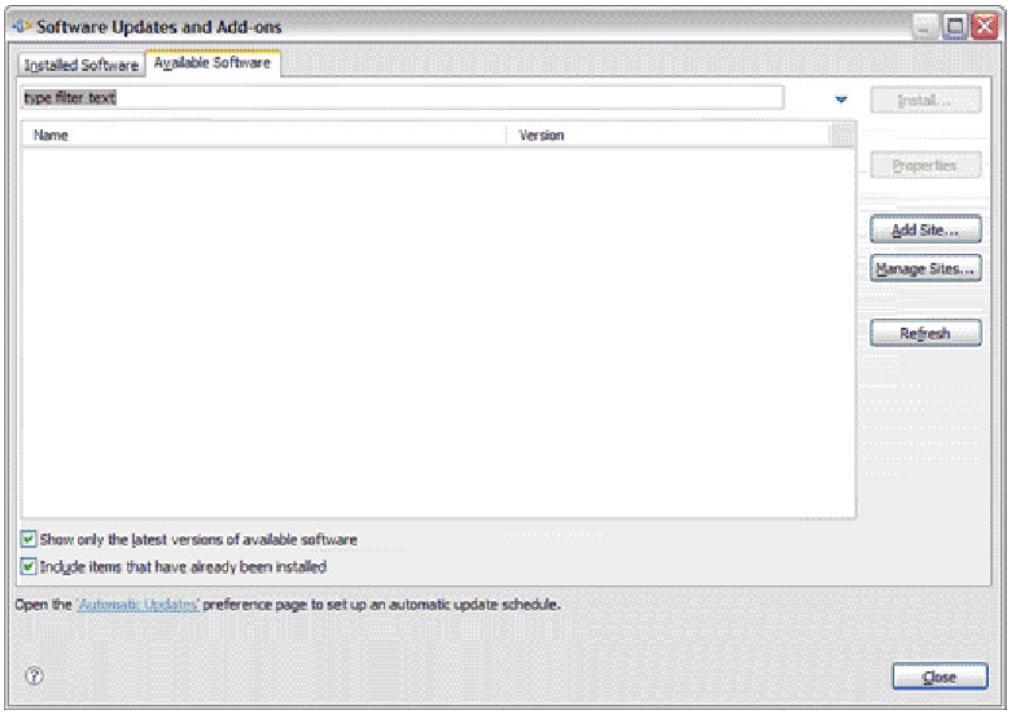
注:

1. SE と AE のいずれの場合でも、`setAMCRoles` スクリプトの呼び出しは任意です。スクリプトを実行する場合、ユーザー名 は、ISC/IBM WebSphere Application Server 環境に既に存在するユーザー名である必要があります。あるいは、ISC コンソール (具体的には、「コンソール・ユーザーの権限」パネル) を使用して、AMC に付属している役割 (「SDI AMC Admin」および「SDI AMC User」) の 1 つを手動でユーザーに割り当てることもできます。
2. AMC の役割についての詳細は、277 ページの『Integrated Solution Console の AMC』を参照してください。

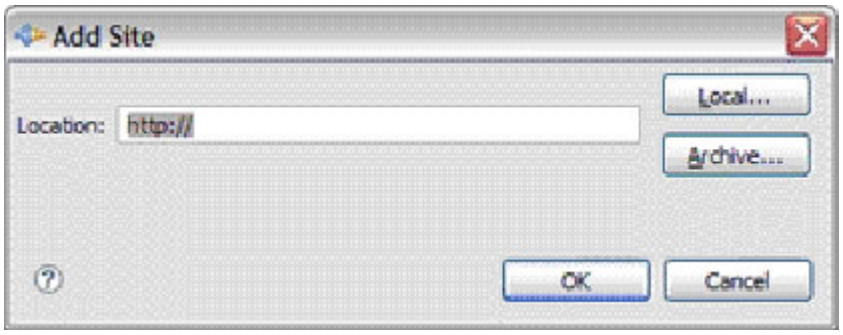
Eclipse 更新マネージャーを使用したインストールと更新

以下の手順に従って、Eclipse 更新マネージャーを使用することができます。

IBM Security Directory Integrator リッチ・クライアント・プラットフォームには、IBM Security Directory Integrator CE を実行するための完全なランタイム環境が含まれています。ただし、既存の Eclipse インストール済み環境に IBM Security Directory Integrator Eclipse プラグインをインストールすることもできます。これは Eclipse 更新マネージャーを使用して行います。Eclipse で、「**ヘルプ**」メニューから「Eclipse 更新マネージャー」を開きます。



IBM Security Directory Integrator プラグインをインストールする前に、新しい更新サイトを追加することができます。「**サイトの追加...**」ボタンを選択して、更新サイトの場所を指定します。



更新サイトの場所に応じて、適切なアクションを選択します。この例では、ローカル・ファイル・システムのディレクトリーを選択することにします。「**ローカル**」ボタンを使用すると、場所入力フィールドに入力するディレクトリーを選択するよう求めるプロンプトが表示されます。「**OK**」を押すと、新しい更新サイトと更新が使用可能になります。

インストールするプラグインにチェックを付けて、「**インストール**」を押します。ソフトウェア更新マネージャーはインストール済み環境を更新する場合、インストールを確認するプロンプトを表示します。インストールが終了したら、ワークベンチをリスタートすることをお勧めします。インストールが完了すると、「**インストール済みのソフトウェア**」タブに IBM Security Directory Integrator が表示されるようになります。

インストール後の手順

以下の説明に従って、インストール後の作業をいくつか実行します。

上述の手順で示したように、他の Eclipse インストール済み環境で CE をプラグインとしてインストールした場合、いくつかの特定のプロパティーを設定することによって、*SDI ローダー* を追加する必要があります。 IBM Security Directory Integrator ローダーは、CE に対してクラスをロードする機能を提供する `org.eclipse.osgi` フラグメントです。

```
# TDI class loader
osgi.framework.extensions=com.ibm.tdi.loader
osgi.hook.configurators.include=com.ibm.tdi.loader.TDIClassLoaderHook
TDI_HOME_DIR=c¥:/Program Files/IBM/TDI/V7.2
```

プロパティー TDI_HOME_DIR は、既存の IBM Security Directory Integrator サーバーをインストールした場所を指す必要があることに注意してください。これは、CE が、正しく機能するために、多数の IBM Security Directory Integrator コンポーネント Java クラスを照会する必要があるからです。このインストール済み環境は、CE が使用するローカル開発サーバーを作成する目的でも使用されます。上のフラグメントは、Windows のデフォルトのインストール場所を示しています。ご使用の環境に応じて更新してください。

これらのプロパティーを設定するには、いくつかの方法があります。1 つの方法は、Eclipse インストール済み環境の `configuration/config.ini` ファイルを更新することです。

注: CE を Eclipse にインストールして構成すると、依存関係の問題が発生することがあります。このような問題を取り扱っている技術情報を参照すると、問題の解決に役立つ場合があります。

アンインストール

IBM Security Directory Integrator 全体をアンインストールするか、または特定のコンポーネントのみをアンインストールすることができます。

アンインストーラーの起動

アンインストーラーを起動するには、いくつかの手順を実行する必要があります。以下の情報を参照して、それを行ってください。

このタスクについて

IBM Security Directory Integrator をアンインストールするには、まずアンインストーラーを起動する必要があります。

注: アンインストールする前に、IBM Security Directory Integrator ランタイムのインスタンス、動作中の AMC サービス、あるいはパスワード同期プラグインなど、削除するコンポーネントをすべて停止します。動作中のコンポーネントを停止しないと、削除されない (アンインストール後も残る) ファイルが生ずる原因になります。Windows ではリスタートが必要です。またサービス・リストに IBM Security Directory Integrator Web Admin (AMC) サービスが残っていれば、手動で削除する必要があります。

手順

1. 例えば次のようにして IBM Security Directory Integrator _uninst ディレクトリーにナビゲートします。*install_path*/_uninst
2. アンインストール実行可能ファイルを実行して、アンインストーラーを起動します。

   Windows プラットフォームでは、アンインストール実行可能ファイル名は uninstaller.exe です。その他すべてのプラットフォームの場合、アンインストール実行可能ファイル名は uninstaller.bin です。
3. ここで、 34 ページの『アンインストール・パネル・フロー』の手順を実行します。

タスクの結果

重要: アンインストール時にいくつかのコンピューターのディレクトリーが空になったり、削除されることがあります。これには、以下のものがあります。

- *TDI_install_dir*/lwi - ここにいくつかのファイルが残っている可能性があります。また組み込み Web プラットフォームによって、インストーラーが配置していないファイルが作成されていることもあります。このディレクトリーは、アンインストールで削除されます。
- *TDI_install_dir*/ce/eclipsece/features/com.ibm.tdi.*.jar
- *TDI_install_dir*/ce/eclipsece/plugins/com.ibm.tdi.*.jar
- *TDI_install_dir*/ce/eclipsece/configuration
- *TDI_install_dir*/ce/update_site/features/com.ibm.tdi.*.jar - このワイルドカードに一致する機能が追加されていた場合、それらは削除されます。
- *TDI_install_dir*/ce/update_site/plugins/com.ibm.tdi.*.jar - 上と同じです。
- *TDI_install_dir*/maintenance/BACKUP - 更新インストーラーによって、このディレクトリーが作成されることがあります。
- *TDI_install_dir*/_uninst/*、*TDI_install_dir*/amc/*、*TDI_install_dir*/osgi/*、*TDI_install_dir*/SCIM/*、および *TDI_install_dir*/LDAPSync/* - これらのディレクトリーは、ユーザーによる変更に関係なく削除されます。

ユーザーがこれらのディレクトリーに配置したファイルでも、以上の基準に一致すれば、アンインストール時には同様に削除されます。

サイレント・アンインストールの実行

ここで説明する手順を使用して、サイレント・アンインストールを実行することができます。

IBM Security Directory Integrator のサイレント・アンインストールを実行するには、まず応答ファイルを生成する必要があります。このファイルを生成するには、GUI でフル・アンインストールを行うか、または -options-record オプションを指定してコンソールでアンインストールを行う必要があります (例えば、TDI_install_dir/_uninst/uninstaller.exe -r UninstallResponseFileName)。アンインストール時に指定するディレクトリーに応答ファイルが作成されます。

注: TDI_install_dir/examples/install ディレクトリーには、さまざまなインストールまたはアンインストールのシナリオに対応した多数のサンプル応答ファイルがあります。
応答ファイルが作成されたら、次のコマンドを使用して、サイレント・モードでアンインストールできます。TDI_install_dir/_uninst/uninstaller.exe -f UninstallResponseFileName

デフォルトのインストール・ロケーション

IBM Security Directory Integrator は、以下のデフォルト・ロケーションにインストールされます。

Windows プラットフォーム
C:¥Program Files¥IBM¥TDI¥V7.2

Linux および UNIX プラットフォーム
/opt/IBM/TDI/V7.2

デフォルトのソリューション・ディレクトリー

tdi_install_dir¥bin¥setDefaultSolDir.bat(sh) を使用して、デフォルトのソリューション・ディレクトリーを新規で設定します。

例えば、以下のコマンドを実行するとします。

```
setDefaultSolDir.bat C:¥mysoldir
```

TDI_SOLDIR=C:¥mysoldir が、 defaultSoldir.bat(sh) ファイルに設定されます。

defaultSoldir.bat(sh) のこの値は、 IBM Security Directory Integrator のインストールでソリューション・ディレクトリーを選択する際に、デフォルトのディレクトリーを設定するのに使用されます。

defaultSoldir.bat(sh) スクリプトのソリューション・ディレクトリー値は、 TDI インストーラーによって設定されます。

このデフォルト値は、インストーラーの Solutions Directory ページで選択するオプションに基づいて変更されます。

defaultSoldir.bat(sh) ファイルは、 IBM Security Directory Integrator サーバーがデフォルトのソリューション・ディレクトリーの場所を取得するのに使用します。

第 3 章 更新インストーラー

IBM Security Directory Integrator 更新インストーラー **applyUpdates.bat(sh)** を使用して、既存の IBM Security Directory Integrator インストール済み環境にフィックスパックをインストールします。

通常のインストーラーは、インストール・ディレクトリーの中に .registry という、インストール済みコンポーネントの現在のレベルを表すファイルを配置しています。tdiSetBackupDir.bat または tdiSetBackupDir.sh というスクリプトが、バックアップ・ディレクトリーの場所を設定するインストール済み環境の bin ディレクトリーに作成されます。このディレクトリーは、デフォルトでは BACKUP の名前でメンテナンス・ディレクトリーにあります。バックアップの場所は、tdiSetBackupDir スクリプトを実行して変更することができます。例えば、このシナリオでは、フィックスが「ifix1」という名前であり、バックアップ・ファイルおよびバックアップ・ディレクトリー が install dir/maintenance/BACKUP/ifix1 の下にあるとします。更新インストーラーはメンテナンス中に、バックアップ・ディレクトリーの名前を取得します。 IBM Security Directory Integrator でメンテナンス手順を実行するユーザーには、インストール・ディレクトリーとバックアップ・ディレクトリーへの書き込み許可が付与されている必要があります。また、完全なアンインストールの際に、アンインストーラーはデフォルトのバックアップ・ディレクトリーの削除を試みる、ということを認識しておく必要があります。

通常のインストーラーは、アンインストール時と機能の追加時に、.registry ファイルのメンテナンスも行います。

- フル・アンインストール時には、他のファイルと共に .registry ファイルも削除されます。
- 部分アンインストール時に .registry ファイルから削除されるのは、アンインストールされるコンポーネントのみです。
- 機能が追加されると、.registry ファイルが更新され、新たにインストールされた機能を取り込みます。

機能を追加したら、現在適用されているすべてのフィックスを直ちにインストールする必要があります。

更新インストーラーには、数件の Java ファイルが含まれています。Java 実行可能ファイルを指定せずに済むよう、 bin ディレクトリーには **applyUpdates.bat(sh)** というラッパー・スクリプトが作成されます。このスクリプトは既存のスクリプトを使用して、使用する該当 JRE を検出し、基になるコードを呼び出します。スクリプトの使用量は以下のように表示されます。

```
applyUpdates -update fix_file.zip [-clean [-silent]]
applyUpdates -rollback
applyUpdates -queryreg
applyUpdates -queryfix fix_file.zip
applyUpdate -enroll license_file.zip
applyUpdates -?
```

オプションは次のとおりです。

-update
このオプションはフィックスパックを適用するために使用します。

フィックスパックを含む圧縮ファイルの名前は、fix_file.zip です。これは、相対パスの場合も絶対パスの場合もあります。

-clean オプションはフィックスパックにのみ使用可能であり、現在のフィックスパックが適用される前にバックアップされていたすべてのファイルを削除します。古いデータの削除を確認するプロンプトが出されます。**-silent** オプションは、確認プロンプトが表示されるのを抑止します。

フィックスパックが再適用される際、新機能でフィックスパックが追加されていることが必要となっている場合などには、**-clean** オプションは無視されます。

-clean オプションを使用してバックアップ・ディレクトリーを空にする場合、ロールバックの機能は単一レベルに制限されます。

-rollback
このオプションは、直前のフィックスを適用する前の状態に IBM Security Directory Integrator をロールバックするために使用します。このデータは、デフォルトでは *tdi_install_dir*/maintenance/BACKUP/FP## に格納されています。

-queryreg
このオプションは、現在のインストール済み環境にある機能と、適用されているすべてのフィックスを表示します。

出力の例を以下に示します。

```
Information from .registry file in: C:¥Program Files¥IBM¥TDI¥V7.2
Edition: Identity
Level: 7.2.0.1

Fixes Applied
=-=-=-=-=-=-=
SDI-7.2-FP0001(7.2.0.0)


Components Installed
=-=-=-=-=-=-=-=-=-=-=
BASE
SERVER
CE
CE UPDATE
JAVADOCS
EXAMPLES
IEHS
EMBEDDED WEB PLATFORM
AMC
   Deferred: false
PLUGINS
```

-queryfix
このオプションは、fix_file.zip に含まれているフィックスに関する情報を表示します。

出力の例を以下に示します。

```
Information from fix file: C:¥fixes¥SDI-7.2-FP0001.zip
Name: fixpack1

Minimum level required to apply fix: 7.2.0
Maximum level allowed to apply fix: 7.2.0.1

Prereq
=-=-=-
None

Components Affected
```

```
=----------------=
BASE
CE
EXAMPLES
```

フィックスを含む圧縮ファイル fix_file.zip には、フィックスの適用に関する情報を含むマニフェスト・ファイル .manifest があります。

-enroll
このオプションは、空のライセンス、試用ライセンス、あるいはフル・ライセンスを登録するために使用します。このオプションは、製品を試用バージョンからフル・バージョンにアップグレードするためにも使用できます。ただし、すべてのインストーラーはこのオプションを使用します。登録されるライセンスは圧縮されたファイルに含まれていて、引数として渡されます。

製品を試用バージョンからフル・バージョンにアップグレードするためにライセンスを更新するには、以下のコマンドを実行します。

```
applyupdates -enroll license_file.zip
```

注: *license_file.zip* ファイルは、 IBM の営業チームまたはサポート・チームから入手できます。

IBM Security Directory Integrator サーバーが、既に登録されているライセンスまたはハードウェア障害のために始動しない場合は、以下の手順を実行します。

1. *tdi-home*.registry ファイルのバックアップをとります。
2. .registry ファイル内で、ライセンス・タグに定義された値を削除します。例えば、<LICENSE>　　Full　　</LICENSE>
3. tdi-home¥license ディレクトリーから既存のノードロック・ファイルを削除します。
4. **applyUpdates** コマンドを実行します。

   ```
   tdi-home¥bin¥applyUpdates -enroll lumfile.zip
   ```

   圧縮ファイル内に含まれたライセンスの数に応じて、生成されるメッセージには、次の例に示すように、適用されたライセンスが示されます。

   ```
   ./applyUpdates.sh -enroll /tmp/TDI_LUM_FULL.zip
   CTGDKO059I 試用ライセンスが正しく登録されました。
   CTGDKO062I 完全ライセンスが正しく登録されました。
   ```

   注: lum_file.zip ファイルは、コマンドの実行後に削除されます。

-? このオプションは、使用法に関する情報を表示します。

.registry ファイル

.registry ファイルによって、システムの特定のインストール・ディレクトリーに現在インストールされているすべての IBM Security Directory Integrator コンポーネントのレベルを知ることができます。

.registry ファイルはインストール・ディレクトリー内にあります。このファイルは最初にインストーラーによって、インストール時に選択したオプションに基づいて作成されます。

フィックスをインストールすると、バックアップ・ディレクトリー内でそのフィックスの名前を持つディレクトリーにバックアップ・ファイルが格納されます。フィ

ックスパックが正常にインストールされると、.registry ファイルに項目が追加されます。これは、フィックスによってコンポーネントに対してなされた変更を表します。.registry ファイルには、適用されたフィックスを表す FIXES セクションがあり、各コンポーネントには、そのコンポーネントを変更した適用済みのフィックスを表す項目があります。ただし、フィックスパックのインストールが失敗すると、.registry ファイルは更新されず、フィックスパックの適用前に存在したのと同じ項目が含まれたままになります。

更新インストーラーは以下のコンポーネントを認識します。

- BASE
- SERVER
- 構成エディター (CE)
- CE_UPDATE
- JAVADOCS
- EXAMPLES
- IEHS
- 組み込み Web プラットフォーム
- 管理およびモニター・コンソール (Administration and Monitoring Console) (AMC)
- PLUGINS: プラグイン・コンポーネントに対して必要ないくつかの手順は、更新インストーラーでは実行できないため、手動で実行することが必要な場合があります。いずれかの pwsync.props ファイルに対して何らかの変更が必要な場合は、その変更を手動で行う必要があります。フィックスパックの manual_readme.txt ファイルに記載されているステップに従ってください。これらのステップはフィックスパックのインストール後に実行する必要がありますが、以下のセクションにリストしたインストール後ステップのどれよりも前に行わなければなりません。この readme ファイルでは、更新インストーラーによって更新されるのは、インストーラーによってインストールされたファイルのみであることを警告しています。ユーザーがコピーしたファイルは、以下のインストール後ステップで説明しているように、手動で更新する必要があります。これらのステップを、指定されているとおりにフィックスパックのインストールの前後に実行する必要があります。以下のステップが必要なのは、対応する Password Synchronizer をターゲット・システムに登録した場合のみです。

  Windows Password Synchronizer

  インストール前

  None

  インストール後

  以下のステップが必要なのは、フィックスパックに Password Synchronizer の DLL の更新が含まれている場合のみです。

  1. 以下のレジストリー・キーから、Password Synchronizer の DLL の名前を削除します。

     HKEY_LOCAL_MACHINE¥SYSTEM¥CurrentControlSet¥Control¥LSA¥Notification Packages

     DLL ファイルの名前は、32 ビット Windows では tdipwflt、64 ビット Windows では tdipwflt_64 です。

2. ローカル・セキュリティー権限 (LSA) プロセスで Password Synchronizer の DLL をアンロードできるようにするため、Windows をリブートします。
3. Windows の system32 フォルダー内の DLL を、Password Synchronizer インストール済み環境の DLL で置き換えます。インストール後の DLL パスは、Windows のバージョンに応じて、*install_dir*/pwd_plugins/windows/tdipwflt.dll か *install_dir*/pwd_plugins/windows/tdipwflt_64.dll のいずれかです。
4. 以下のレジストリー・キーに DLL の名前 (.dll 拡張子は外します) を追加します。

   HKEY_LOCAL_MACHINE¥SYSTEM¥CurrentControlSet¥Control¥LSA¥Notification Packages
5. LSA に新しい DLL をロードできるようにするため、Windows を再びリブートします。

後に再始動したときに、Password Synchronizer は更新されたファイルを使用して正常に実行されるはずです。

IBM Security Directory Server Password Synchronizer

インストール前

1. Directory Server を停止します。
2. **stopProxy** コマンド行ユーティリティーを使用して、Password Synchronizer のプロキシー・プロセスを停止します。この手順が必要なのは、IBM Security Directory Server Password Synchronizer が、終了時に自らのプロキシーを自動的に停止しないためです。

インストール後

None

Sun Directory Server の Password Synchronizer

インストール前

Directory Server を停止します。

インストール後

None

PAM Password Synchronizer

インストール前

更新時にパスワードを変更することは、できればやめてください。別の方法として、PAM 構成ファイルから Password Synchronizer の登録を抹消してください。

インストール後

更新前に Password Synchronizer を登録抹消した場合は、再度登録します。

詳しくは、IBM Security Directory Integrator の資料の『*Password Synchronization Plug-ins*』セクションを参照してください。

IBM Domino Password Synchronizer

インストール前

インストール後

IBM Security Directory Integrator の資料の『*Password Synchronization Plug-ins*』セクションの『*IBM Domino HTTP Password Synchronizer*』に関するセクションに記載されているインストール後手順を実行します。

次に、『*単一 IBM Domino サーバーでのデプロイメント*』セクションの説明に従って、IBM Domino プラグインの新規セットアップを行います。

フィックスパックのインストール

フィックスパックをインストールするには、フィックスパックに付属の `readme` ファイルに記載されている説明に従います。

フィックス・ファイルにコンポーネントのフィックスが含まれていて、しかもシステムにそのコンポーネントがインストールされている場合は、個々のコンポーネントに対して、いくつかのプログラム化されたアクションが実行されます。

更新インストーラーではなく手動で手順を実行することが必要な場合は、その指示がフィックスの `readme` ファイルに含まれています。

注: フィックスパックによる更新を行う前に、IBM Security Directory Integrator のすべてのプロセスをシャットダウンする必要があります。

インストール後: Federated Directory Server を含む以前のインストール済み環境が存在していた場合は、`*.xml` ファイルを *SDI_solution_dir*`/LDAPSync` から *SDI_solution_dir*`/configs` に手動でコピーする必要があります。

ロールバック

ロールバック時に、更新インストーラーは、フィックスのインストール中に記録した情報とバックアップされたファイルを使用して、以前の状態を復元します。

注: ロールバックを行う前に、IBM Security Directory Integrator のすべてのプロセスをシャットダウンする必要があります。

ロールバック操作では、フィックスパックのインストール中に手動でアクションを行ったファイルはロールバックされません。

トラブルシューティング

更新インストーラーのログを使用して、更新のインストールに関連したエラーのトラブルシューティングを行います。

更新インストーラーは、`install_dir/logs` ディレクトリーに `updateinstaller.log` という名前のログ・ファイルを作成します。デフォルトでは、INFO レベルのメッセージがログに記録されます。これを変更するには、 DEBUG メッセージもログに記録されるよう `install_dir/logsinstall_dir/etc/updateinstaller-log4j.properties` ファイルを変更します。

第 4 章 サポートされるプラットフォーム

サポートされるオペレーティング・システム、Web ブラウザー、および仮想化のサポートの詳細については、 ここに示したリンクを参照してください。

http://pic.dhe.ibm.com/infocenter/tivihelp/v2r1/topic/com.ibm.IBMDI.doc_7.2/sysreqs.html にある IBM Security Directory Integrator の資料に記載されている『ソフトウェア要件』のセクションを参照してください。

第 5 章 マイグレーション

以下の情報を参照して、マイグレーション、そのタイプ、シナリオ、およびマイグレーションできる各種コンポーネントについて学習できます。

IBM Security Directory Integrator のコンテキストでは、「マイグレーション」に以下の複数の意味がある場合があります。

- 同じマシンや別のマシン上の新規の場所で使用する、関連ファイル (およびそのコンテンツ) を作成する。
- 製品の新規バージョンで使用する関連ファイルを作成する。

マイグレーションのシナリオの要約を次の表に示します。

表 1. マイグレーションのシナリオ

ソースと宛先のバージョンが同一か?	ソースと宛先のインストール・パスは同一か?	シナリオの説明
いいえ	いいえ	異なる場所にインストールされた新しいバージョンへのファイルのマイグレーション
いいえ	はい	同じ場所にインストールされた新しいバージョンへのファイルのマイグレーション
はい	いいえ	同一バージョンの異なるインストール済み環境へのファイルのマイグレーション
はい	はい	バックアップ・ファイルの元の場所へのファイルのリストア

新しいバージョンへのマイグレーションおよび新しい場所へのマイグレーションの両方を行わなければならない場合は、最初にバージョンをアップグレードしてください。ここでは、現行リリースでの場所のマイグレーションのみを説明します。

IBM Security Directory Integrator のインストーラーは、IBM Security Directory Integrator 7.0、7.1、および 7.1.1 から IBM Security Directory Integrator バージョン 7.2 へのマイグレーションを支援できます。

注: IBM Security Directory Integrator をバージョン 6.x 以前からバージョン 7.2 に直接アップグレードすることはサポートされていません。まずバージョン 6.x をバージョン 7.1.1 にアップグレードしてから、バージョン 7.1.1 をバージョン 7.2 にアップグレードする必要があります。

異なる場所へのファイルのマイグレーション

マイグレーションするコンポーネントやファイルを選択する前に、特定の問題を解決することができます。

このセクションでは、IBM Security Directory Integrator についてのみ説明します。

別の場所で使用するファイルで、変更の必要がないファイル

変更の必要がないファイル・タイプのリストを参照できます。

- ユーザー構成、データ・ファイル (.xml、.xsd、.xsl、.txt ...)、鍵ストア・ファイル (.jks、...)、証明書ファイル (.der、...) など。

次のセクションも参照してください。 208 ページの『暗号化の成果物 (鍵、証明書、鍵ストア、暗号化ファイル) の管理』

- スクリプト (例外は 69 ページの『通常の環境では別の場所で使用されないファイル』を参照してください)。
- .bat、.sh および .vbs ファイル
- JAR ファイル
- ネイティブ・バイナリー (.exe、.dll、.so など)。
- サーバー API のレジストリー・ファイル
- サーバー Stash ファイル
- Derby データベース

  例えば、デフォルトのシステム・ストア・データベース「TDISysStore」および、デフォルトの AMC データベース「tdiamcdb」など。

  データベースはひとまとめ (フォルダー全体)として、移動しなければならないのでご注意ください。2 つのデータベースのファイルをマージしないでください。

  もっと複雑なシナリオの場合は、JDBC コネクターを使用してデータベース間でデータを転送します。
- Action Manager のプロパティー・ファイル (*TDI_install_dir*/bin/amc/ActionManager フォルダーにあります)。
- 以下を除く、「etc」フォルダーにある構成ファイル。
  - build.properties
  - global.properties
  - updateinstaller-log4j.properties
  - tdisrvctl-log4j.properties
- AMC 構成ファイル:
  - amc.properties
  - amcdbhandler.properties
  - amcdbschema.xml
  - idiamc.sth
  - AMC 機能は推奨されません。IBM Security Directory Integrator の将来のバージョンでは除去されます。

別の場所で使用する前に変更が必要なファイル

変更が必要なファイル・タイプのリストを参照できます。

一般に、場所に依存するファイルには、1 つ以上の場所にインストール・フォルダーの絶対パスが含まれています。これらが現れる場所は、新しいロケーション・パスで置き換え、ファイルが新しい場所に関連付けられるようにする必要があります。

次に、マイグレーションが必要なファイルと、更新する必要があるフィールドについてのヒントをリストします。これらのヒントは、各ファイルのデフォルトの内容

を基にしています。ファイルを変更した場合は、他にもロケーション固有で、更新が必要なフィールドがあるかもしれません。

- bin/amc/amcwinservice.ini:

  これは、AMC が Windows サービスとして登録されるときに使用される構成ファイルです。

  「WorkingDirectory」、「StartCommand」および「StopCommand」プロパティーを更新します。

- global.properties/solution.properties:

  「com.ibm.di.store.database」プロパティーを更新します。

  次のプロパティーも確認してください。
  「api.config.folder」、「systemqueue.jmsdriver.param.mqe.file.ini」および「com.ibm.di.loader.userjars」

  ファイルを、異なる暗号鍵を使用するインストール済み環境にマイグレーションする場合は、208 ページの『暗号化の成果物 (鍵、証明書、鍵ストア、暗号化ファイル) の管理』セクションを参照してください。

- etc/updateinstaller-log4j.properties:

  「log4j.appender.Default.file」プロパティーを更新します。

- etc/tdisrvctl-log4j.properties:

  「log4j.appender.Default.file」プロパティーを更新します。

- ibmdiservice.props:

  これは、サーバーが Windows サービスとして登録されるときに使用される構成ファイルです。

  「path」、「ibmdiroot」および「jvmRoot」プロパティーを更新します。

- pwsync.props:

  これらは、パスワード・シンクロナイザーの構成ファイルです。

  「proxyStartExe」、「logFile」、「javaLogFile」および「mqe.file.ini」プロパティーを更新します。

通常の環境では別の場所で使用されないファイル

マイグレーション後に使用されないファイル・タイプのリストを参照できます。

- 特定のスクリプト

  これらのファイルが存在する目的は、ロケーション固有のデータを送ることだけです。

  実質的には他に何もないので、別の場所に置いてもほとんど価値がありません。

  TDI_install_dir/bin フォルダーから次のスクリプトを確認してください。
  「javaHome」、「defaultSolDir」、「backupDir」、「tdiISCHome」。

- .reg ファイル

  これらは、Windows Password Synchronizer で使用されます。
- IBM WebSphere MQ Everyplace キュー・マネージャー・ファイル

  IBM WebSphere MQ Everyplace ファイルを別の場所に簡単にマイグレーションすることはできませんが、JMS コネクターを IBM WebSphere MQ Everyplace JMS ドライバーと一緒に使用して、データをある IBM WebSphere MQ Everyplace キューから別のキューに転送することはできます。
- CE ワークスペース

  Directory Integrator プロジェクトを構成エディターのワークスペースから再利用するために、それらを Directory Integrator 構成としてエクスポートし、それを新規のワークスペースにインポートします。
- etc/build.properties

  このファイルには、製品のリリースについての日時およびバージョン情報が含まれます。

暗号化データが含まれたファイルのマイグレーション

暗号化データが含まれたファイルのマイグレーションについて詳しくは、以下のリンクを参照してください。

208 ページの『暗号化の成果物 (鍵、証明書、鍵ストア、暗号化ファイル) の管理』を参照してください。

新しいバージョンへのファイルのマイグレーション

新しいバージョンへのファイルのマイグレーションは 3 つの方法で行うことができます。詳しくは、以下の情報を参照してください。

インストーラーが支援するマイグレーション

以下の情報を参照することで、インストーラーがどのように動作して、すべてのファイルが自動および手動でマイグレーションされるかについて詳しく知ることができます。

インストーラーは、新しいバージョンへのアップグレード中に、特定のファイルを自動的にマイグレーションします。インストーラーは、Directory Integrator のインストール・フォルダーのみを確認することにご注意ください。

インストール・フォルダーとは異なるすべてのソリューション・フォルダーは、手動で、(またはセクション 72 ページの『ツールが支援するマイグレーション』で説明されているツールを使用して) マイグレーションしなければなりません。

インストーラーが自動的にマイグレーションするファイル

- 6.0 から 7.1
  - global.properties

  - Cloudscape データベース (システム・ストアで使用されている場合) は Derby バージョン 10.5.3 にアップグレードされます。 100 ページの『Cloudscape データベースの Derby へのマイグレーション』を参照してください。
  - pwsync.props (インストールされているパスワード・プラグインごと)
- 6.1.x から 7.1
  - global.properties
  - AMC データベース
  - amc.properties
  - am_config.properties
  - pwsync.props (インストールされているパスワード・プラグインごと)
- 7.0 から 7.1
  - global.properties
  - pwsync.props (インストールされているパスワード・プラグインごと)
- 7.1 から 7.1.1
  - global.properties
  - solution.properties (デフォルトのソリューション・ディレクトリーに存在する場合)
  - pwsync.props (インストールされているパスワード・プラグインごと)
- 7.1.1 から 7.2
  - etc¥reconnect.rules
  - etc¥derby.properties
  - etc¥jlog.properties
  - etc¥log4j.properties
  - etc¥tdisrvctl-log4j.properties
  - etc¥tdimiggbl-log4j.properties
  - etc¥updateinstaller-log4j.properties
  - etc¥it_registry.properties
  - etc¥tp.xml
  - etc¥activemq.xml
  - etc¥global.properties
  - solution.properties (デフォルトのソリューション・ディレクトリーに存在する場合)
  - pwsync.props (インストールされているパスワード・プラグインごと)
  - AMC データベース (bin/backupam.bat、bin/backupamc.bat、および bin/backupamcdb.bat のツールを使用して AMC をバックアップしてください)
  - AMC_7.2.0.0¥amc.properties
  - AMC_7.2.0.0¥conf¥amcdbhandler.properties
  - AMC_7.2.0.0¥conf¥logging.properties
  - bin¥amc¥ActionManager¥am_config.properties
  - bin¥amc¥ActionManager¥am_logging.properties

手動でマイグレーションする必要のあるファイル

73 ページの『手動マイグレーション』セクションに示したすべて (ただし、最初の『プロパティー・ファイル』サブセクションに示したものは除く)。

ツールが支援するマイグレーション

インストーラーが支援するマイグレーションで使用されるツールのリストを確認します。

次のツールは、インストーラーが支援するマイグレーションでインストーラーが使用するものです。これらを手動のマイグレーションで使用することができます。

プロパティー・ファイルのマイグレーション

- global.properties:

  TDI_install_dir/bin にある「tdimiggbl」ツールを使用します。 101 ページの『マイグレーション・ツールを使用したグローバル・プロパティー・ファイルおよびソリューション・プロパティー・ファイルのマイグレーション』セクションを参照してください。

- amc.properties:

  TDI_install_dir/bin/amcにある「tdimigamc」ツールを使用します。 325 ページの『AMC および AM コマンド行ユーティリティー』を参照してください。 AMC 機能は推奨されません。IBM Security Directory Integrator の将来のバージョンでは除去されます。

- am_config.properties:

  TDI_install_dir/bin/amcにある「tdimigam」ツールを使用します。 325 ページの『AMC および AM コマンド行ユーティリティー』を参照してください。

- pwsync.props (インストールされているパスワード・プラグインごと)

  TDI_install_dir/pwd_plugins/binにある「migpwsync」ツールを使用します。 103 ページの『マイグレーション・ツールを使用したパスワード・プラグイン・プロパティー・ファイルのマイグレーション』を参照してください。

AMC データベースのマイグレーション

TDI_install_dir/bin/amc にある「backupamcdb」/「restoreamcdb」ツールを使用します。 325 ページの『AMC および AM コマンド行ユーティリティー』を参照してください。 AMC 機能は推奨されません。IBM Security Directory Integrator の将来のバージョンでは除去されます。

Cloudscape システム・ストア・マイグレーション (6.0 のみ)

詳細は、 100 ページの『Cloudscape データベースの Derby へのマイグレーション』セクションを参照してください。

手動マイグレーション

以下の情報を参照して、手動マイグレーションを開始できます。また、各 IBM Security Directory Integrator バージョンで変更、追加および削除されたプロパティーの広範なリストを参照できます。

構成ファイルおよびその他のカスタム・ファイル (Derby データベースを含む) を、前のインストール・ディレクトリーから新しいインストール・ディレクトリーにコピーします。IBM Security Directory Integrator ではソリューション・ディレクトリーがサポートされるため、構成ファイル、プロパティー・ファイル、Derby データベースなどを、IBM Security Directory Integrator バージョンのインストール・ディレクトリーではなく、このようなソリューション・ディレクトリーにコピーすることをお勧めします。

上記で参照したオブジェクトを新しい場所にコピーしたら、その内容の手動によるマイグレーションを開始することができます。次のセクションで説明されているように、オブジェクトは IBM Security Directory Integrator で使用できるように変更します。

1. プロパティー・ファイル
2. 構成
3. カスタマイズ・スクリプト
4. 追加または置き換えられたインストール済み環境の JAR ファイル
5. パスワード・シンクロナイザーの構成

注: Sandbox のデータはバージョンごとに固有です。以前のバージョンで記録したデータをバージョン 7.2 で再生することはできません。

プロパティー・ファイル

- global.properties:

  以下の表に、IBM Security Directory Integrator の各種バージョンで削除、変更、または追加されたプロパティーをリストします。

表 2. 削除および変更されたプロパティー

Global/Solution.properties 内のプロパティー	変更、削除、または追加	注釈
web.server.ssl.on	*変更*	このプロパティーについて詳しくは、『*v7.1.1 での新規プロパティー*』という表を参照してください。 IBM Security Directory Integrator バージョン 7.2 以降では、このプロパティーのデフォルト値は true です。 例: web.server.ssl.on=true

表 2. 削除および変更されたプロパティー (続き)

Global/Solution.properties 内のプロパティー	変更、削除、または追加	注釈
{protect}-dashboard.auth.user.admin	*追加*	このプロパティーは、IBM Security Directory Integrator バージョン 7.2 で追加されました。これは Federated Directory Server のユーザー名およびパスワードを指定するために使用します。 このプロパティーのデフォルト値は admin です。 例: {protect}-dashboard.auth.user.admin=admin 複数の Federated Directory Server ユーザー・ログイン・アカウントを指定するには、以下の例を参照してください。 {protect}-dashboard.auth.user.admin=admin {protect}-dashboard.auth.user.user1=user1passwd {protect}-dashboard.auth.user.user2=user2passwd
dashboard.auth.localhost	*変更*	IBM Security Directory Integrator バージョン 7.2 以降では、このプロパティーのデフォルト値は properties です。
dashboard.auth.remote	*変更*	IBM Security Directory Integrator バージョン 7.2 以降では、このプロパティーのデフォルト値は properties です。
com.ibm.di.server.NIST.on	*追加*	IBM Security Directory Integrator バージョン 7.2 以降では、このプロパティーは IBM Security Directory Integrator の NIST モードに切り替えるために使用します。 このプロパティーを true に設定すると、IBM Security Directory Integrator は NIST 準拠モードでの実行が適用されます。 デフォルト値は false です。つまり、デフォルトでは NIST モードで実行されません。

表 3. v7.1.1 で削除および変更されたプロパティー

前のプロパティー (v7.0 より前)	新規プロパティー	注釈
## Active Correlation Technology engine settings # act.engine.rule.set.file =myrules.acts	*削除*	ACT エンジンおよび ACT コネクターは削除された
# Location of directory where the JRE that SDI will use is installed com.ibm.di.jvmdir=$jvmRoot$	*削除*	指定不可。
com.ibm.di.scriptengine .precompile=true	*削除*	指定不可。現在のスクリプト・エンジンにこの機能はない。
com.ibm.di.scriptengine .regex=java	*削除*	指定不可。Java 構文が常に続く。
ibmjs.options=com.ibm.di.script .ScriptEngineOptions	*削除*	前のプロパティーに関連。このオプションは無効。
com.ibm.di.store.create .checkpoint.store =<multiple statements>	*削除*	チェックポイント・リスタート機能は削除された。これに関連するシステム・ストアのテーブル作成ステートメントも削除された。
com.ibm.di.admin.library.dir=	*削除*	現在の構成エディターではこれを使用しないため、指定不可。

表 3. v7.1.1 で削除および変更されたプロパティー (続き)

前のプロパティー (v7.0 より前)	新規プロパティー	注釈
api.remote.on=false	api.remote.on=true	RMI は IBM Security Directory Integrator サーバーでデフォルトで有効。デフォルトで有効なので、true に設定。
javax.net.ssl.trustStore= {protect}-javax.net.ssl .trustStorePassword= javax.net.ssl.trustStoreType=	javax.net.ssl.trustStore =serverapi¥testadmin.jks {protect}-javax.net.ssl .trustStorePassword=administrator javax.net.ssl.trustStoreType=jks	RMI は IBM Security Directory Integrator サーバーでデフォルトで有効。空の値をデフォルトのトラストストアで置換。
javax.net.ssl.keyStore= {protect}-javax.net.ssl .keyStorePassword= javax.net.ssl.keyStoreType=	javax.net.ssl.keyStore =serverapi¥testadmin.jks {protect}-javax.net.ssl .keyStorePassword=administrator javax.net.ssl.keyStoreType=jks	RMI は IBM Security Directory Integrator サーバーでデフォルトで有効。空の値をデフォルトの鍵ストアで置換。
com.metamerge .securityTransformation =DES/ECB/NoPadding	com.ibm.di.securityTransformation =DES/ECB/NoPadding	FIPS 140-2 認証。プロパティー名を変更。
com.ibm.di.server.keystore =myKeyStore.jks com.ibm.di.server.key.alias =myKeyAlias	api.keystore=myKeyStore.jks api.key.alias=myKeyAlias	サーバー API の鍵ストア・プロパティーの名前変更。
com.ibm.di.store.database =TDISysStore com.ibm.di.store.jdbc.driver =org.apache.derby.jdbc .EmbeddedDriver com.ibm.di.store.jdbc.urlprefix =jdbc:derby: com.ibm.di.store.jdbc.user=APP	#com.ibm.di.store.database =TDISysStore #com.ibm.di.store.jdbc.driver =org.apache.derby.jdbc .EmbeddedDriver #com.ibm.di.store.jdbc.urlprefix =jdbc:derby: #com.ibm.di.store.jdbc.user=APP	システム・ストアは現在デフォルトでネットワーク・モードで実行するため、システム・ストアの組み込みモードのプロパティーをコメント化。インストーラーはこの変更を行わない。以前に Cloudscape または Derby を組み込みモードで使用していた場合は、手動でこの変更を行う必要がある。
#com.ibm.di.store.database =jdbc:derby://localhost:1527 /TDISysStore;create=true #com.ibm.di.store.jdbc.driver =org.apache.derby.jdbc .ClientDriver #com.ibm.di.store.jdbc .urlprefix=jdbc:derby: #com.ibm.di.store.jdbc .user=APP #com.ibm.di.store.jdbc .password=APP #com.ibm.di.store.jdbc.start.mode =automatic #com.ibm.di.store.jdbc.host =localhost #com.ibm.di.store.jdbc.port=1527 #com.ibm.di.store.jdbc.sysibm=true	com.ibm.di.store.database =jdbc:derby://localhost:1527 /TDISysStore;create=true com.ibm.di.store.jdbc.driver =org.apache.derby.jdbc.ClientDriver com.ibm.di.store.jdbc.urlprefix =jdbc:derby://localhost:1527/ com.ibm.di.store.jdbc.user=APP com.ibm.di.store.jdbc.password=APP com.ibm.di.store.jdbc.start.mode =automatic com.ibm.di.store.jdbc.host =localhost com.ibm.di.store.jdbc.port=1527 com.ibm.di.store.jdbc.sysibm=true	これらは IBM Security Directory Integrator v7.1.1 の新しいデフォルトのシステム・ストアのプロパティーである。インストール済み環境をマイグレーションした場合、システム・ストアをネットワーク・モードで実行するには、global.properties ファイルにもこれらの変更を加える必要がある。 構成エディターの新しいアーキテクチャーでは、開発プロセスへの他の変更と関連して、システム・ストアを組み込みモードで実行させるのはとても難しい。したがって、ネットワーク・モードで実行することが強く推奨される。
api.config.folder=$change$/configs	api.config.folder=configs	configs フォルダーは、現在常にソリューション・ディレクトリーに対してローカルである。

表 3. v7.1.1 で削除および変更されたプロパティー (続き)

前のプロパティー (v7.0 より前)	新規プロパティー	注釈
##---------- ## System Queue settings ##---------- ## If set to "true" the System Queue is initialized on startup and can be used; ## otherwise the System Queue is not initialized and cannot be used. systemqueue.on=false ### MQe JMS driver initialization プロパティー ## Specifies the location of the MQe initialization file. ## This file is used to initialize MQe on TDI server startup. systemqueue.jmsdriver.param.mqe .file.ini=$change$/MQePWStore /pwstore_server.ini	##---------- ## System Queue settings ##---------- ## If set to "true" the System Queue is initialized on startup and can be used; ## otherwise the System Queue is not initialized and cannot be used. systemqueue.on=true ### MQe JMS driver initialization プロパティー ## Specifies the location of the MQe initialization file. ## This file is used to initialize MQe on TDI server startup. systemqueue.jmsdriver.param.mqe .file.ini=MQePWStore/pwstore_server .ini	IBM Security Directory Integrator では現在システム・キューはデフォルトで有効。IBM WebSphere MQ Everyplace 初期化ファイルも、現在はソリューション・ディレクトリーに従属するディレクトリーに置かれている。

表 4. v7.0 での新規プロパティー

プロパティー	注釈
com.ibm.di.server.fipsmode.on=false	FIPS モードを有効/無効にする新規のプロパティーが追加された。
## To enable the built-in JAAS Authentication mechanism, ## set this property to "[jaas]". api.custom.authentication ## JAAS Authentication properties ## ---------- ## java.security.auth.login.config=	サーバー API 認証プロバイダーとして JAAS のサポートを提供。JAAS 構成ファイルを指定できる空のプロパティーが提供されている。
## Encryption certificate properties com.ibm.di.server.encryption.keystore = <<value of com.ibm.di.server.keystore from 6.1.1 global.properties>> com.ibm.di.server.encryption.key.alias = <<value of com.ibm.di.server.key.alias from 6.1.1 global.properties >> ## Server API keystore passwords {protect}-api.keystore.password= << keystore password from idisrv.sth>> {protect}-api.key.password= << key password from idisrv.sth if present>>	PKI 暗号化および SSL で使用する証明書用に、別々の構成オプションを提供。
## TDI Logging com.ibm.di.logging.enabled=true	ロギングをすべて無効にするメカニズムを提供。すべてのロギングを無効にする場合は、「false」に設定する。
derby.connection.requireAuthentication=true derby.authentication.provider=BUILTIN derby.database.defaultConnectionMode=fullAccess	ネットワーク・モードのシステム・ストア (Derby) 用の追加パラメーター。
##PKCS11 options ##Set the value of following properties to use PKCS11 enabled ## devices to store TDI servers private key / certificate. com.ibm.di.pkcs11cfg=etc¥pkcs11.cfg com.ibm.di.server.pkcs11=false com.ibm.di.server.pkcs11.library= com.ibm.di.server.pkcs11.slot= {protect}-com.ibm.di.server.pkcs11.password =PASSWORD	PKCS 11 準拠の暗号デバイスでの IBM Security Directory Integrator サーバーの秘密鍵および証明書をサポート。

表 4. v7.0 での新規プロパティー (続き)

プロパティー	注釈
## Specify the unique ID for the TDI Server ## ---------- ## This property helps a client connecting to the TDI server to identify different servers ## running on the same IP and the same port in different time. (Default is blank) com.ibm.di.server.id=	リモート・サーバー・クライアントが通信先の IBM Security Directory Integrator サーバーを検出するために使用可能な、 サーバーのサーバー固有 ID を提供する必要がある。
## Timeout in minutes for loading configuration. api.config.load.timeout=2	構成の初期化およびサーバー API の初期化は、同期させる必要がある。
com.ibm.di.server.encryption.keystoretype = jks com.ibm.di.server.encryption.transformation = RSA	対称型暗号のサポート (FIPS 140-2 準拠)。
## Specifies a list of Server notification types, which will be suppressed. ## Notifications of suppressed types will not be propagated by the notifications framework. ## The notification types in the list are separated by spaces. Wildcards may be included. ## Example: ## api.notification.suppress=di.al.* di.ci.start ## The above example will suppress all AssemblyLine related notifications as well as ## notifications for starting a configuration instance. ## If the property is missing or is empty, no notifications will be suppressed. api.notification.suppress=di.server.api.authenticate di .server.api.authorize.*	IBM Security Directory Integrator 監査機能の提供。サーバー通知抑止。
api.audit.on=false	IBM Security Directory Integrator 監査機能の提供。

表 4. v7.0 での新規プロパティー (続き)

プロパティー	注釈
## This property specifies whether LDAP Group authentication is turned on. ## If it is set to 'true', the group membership of the authenticating user ## will be resolved and will be taken into account during authorization. ## If it is missing, the default value 'false' is used. api.custom.authentication.ldap.groupsupport= false ## Specifies the name of the attribute of a user in LDAP that contains a list of the groups of which the user is a member. ## It is taken into account only if 'api.custom.authentication.ldap.groupsupport' is set to true. api.custom.authentication.ldap.usermembershipattribute= ## Specifies how groups are named in the membership attribute of a user. ## For example, if the user's membership attribute contains values, ## which correspond to the 'objectSID' attributes of groups, set this property to 'objectSID'. ## If the user's membership attribute contains distinguished names of groups, then set this property to 'dn'. ## The property is required in case 'api.custom.authentication. ldap.groupsupport' is set to true. api.custom.authentication.ldap.usermembershipattributecontent= ## Specifies the name of a group's attribute in LDAP which corresponds to the way the group is named in the TDI User Registry. ## For example, if LDAP groups are addressed in the TDI registry by their common name, then set this property to 'cn'. ## If the User Registry contains the distinguished names of the groups, then set this property to 'dn'. api.custom.authentication.ldap.groupnameattribute= ## Represents the LDAP directory context, where groups will be searched. ## It is required only when LDAP group support is enabled api.custom.authentication.ldap.groupsearchbase= ## Optional property, which represents a list of space-separated attribute names. ## Specifies attributes which have non-string syntax. ## api.custom.authentication.ldap.binaryattributes=	LDAP グループをサポートするための拡張権限。

表 5. v7.1 での新規プロパティー

プロパティー	注釈
# api.remote.server.ports=8700-8900	デフォルトでコメント化。このプロパティーは RMI ポートを構成するために使用する。これは、デフォルトのポートがファイアウォールと競合する場合に役立つ。 サーバーは、他のプロパティーで定義されたポートでの listen に加え、これらのポートを着信した RMI サービス要求の listen に使用する。送信する RMI サービス要求には、ランダムなポート番号が使用される。

表 5. v7.1 での新規プロパティー (続き)

プロパティー	注釈
## The properties determine the default bind address and the remote bind address for the Server API. ## * means bind to all network interfaces. The Remote Bind Address overrides the Default one. ## Only one IP address should be set. No hostnames are accepted. ## Mind that the java.rmi.server.hostname property is set implicitly to equal the Remote Bind Address property when used. This will cause the client stubs to create sockets on the specified Remote Bind Address. # com.ibm.di.default.bind.address=* # api.remote.bind.address=*	デフォルトでコメント化。これらの 2 つのプロパティーは、リモート API が listen するネットワーク・インターフェース (ホスト名または IP アドレス) の構成に使用する。
## Touchpoint Server properties tp.server.on=false tp.server.port=1098 tp.server.config=etc/tp.xml tp.server.auth=false tp.server.auth.realm=Tivoli Directory Integrator Touchpoint Server	これらのプロパティーは、Service Control Management Protocol (SCMP) をベースにしたサービスを使用して、IBM Security Directory Integrator コネクターへの REST インターフェースを構成する。
## ## Server API client properties ## api.client.ssl.custom.properties.on=true api.client.keystore=serverapi/testadmin.jks {protect}-api.client.keystore.pass=administrator api.client.keystore.type=jks {protect}-api.client.key.pass=administrator api.client.truststore=serverapi/testadmin.jks {protect}-api.client.truststore.pass=administrator api.client.truststore.type=jks	これらのプロパティーは、サーバー API クライアントのカスタム SSL プロパティーを有効にする。api.client.ssl.custom.properties.on=true の場合は、api.client.* プロパティーがサーバー API クライアントで使用される。それ以外の場合は、デフォルトの javax.net.ssl.* プロパティーが使用される。

表 6. v7.1.1 での新規プロパティー

プロパティー	注釈
## Web container web.server.port=1098 web.server.ssl.on=false web.server.ssl.client.auth.on=false # web.server.session.timeout=300	これらのプロパティーは、REST API およびダッシュボード用の一般設定である。この設定では、HTTP アクセス・ポイントのポートおよびセキュリティー設定を IBM Security Directory Integrator REST およびダッシュボードに指定する。

表 6. v7.1.1 での新規プロパティー (続き)

プロパティー	注釈
``## Dashboard properties ## dashboard.on=true dashboard.templates.folder=dashboard/templates ## Dashboard authentication properties ## ## The values for localhost and remotehost can be: ## none: No authentication is required ## deny: All connections denied ## ldap: Authentication is done by logging into an LDAP server and optionally validating group membership ## ## dashboard.ldap.url ## Specify the LDAP host port and optionally a search base (ldap://<host>:<port>[/<search base>]) ## ## dashboard.ldap.url.group ## Specify the LDAP host port and optionally a search base (ldap://<host>:<port>[/<search base>]) ## dashboard.auth=true dashboard.auth.localhost=none dashboard.auth.remote=deny # dashboard.auth.ldap.url =ldap://localhost:389/ou=users,ou=system # dashboard.auth.ldap.url.group =ldap://localhost:389/cn=group1, ou=groups,ou=system```	これらは、ダッシュボードに固有のプロパティーである。手動で設定されるプロパティーは、「dashboard.on」(ダッシュボードの Web アプリケーションを使用可能または使用不可にする) および「dashboard.templates.folder」(テンプレート・ソリューションの場所) である。 他のすべてのプロパティーがダッシュボードで編集できる。
``## REST API ## ---------------------- api.rest.on=true api.rest.auth=false api.rest.auth.realm =Tivoli Directory Integrator REST API api.rest.jmsdriver.name =com.ibm.di.systemqueue.driver.ActiveMQ api.rest.jmsdriver.queue.sender.persistance=false api.rest.jmsdriver.queue.sender.timeToLive=60000 api.rest.jmsdriver.param.jms.broker =vm://localhost?brokerConfig =xbean:etc/activemq.xml # api.rest.jmsdriver.auth.username # api.rest.jmsdriver.auth.password```	これらのプロパティーは、IBM Security Directory Integrator REST API の有効化およびセキュリティー設定を構成する。api.rest.jmsdriver プロパティーは、非同期ログ・メッセージ内で使用する JMS キューを指定する。メッセージはダッシュボードで使用される。

表 7. v7.1.1 での削除/変更

前のプロパティー	新規プロパティー	注釈
``com.ibm.di.store.database =jdbc:derby://localhost :1527/$change$/TDISysStore; create=true```	```com.ibm.di.store.database =jdbc:derby://localhost :1527/$soldir$/TDISysStore; create=true```	IBM Security Directory Integrator バージョン 7.1.1 以降では、$soldir$ はデフォルトでは <TDI_ install_dir> で置換されない。このディレクトリーは、 JVM 内部のランタイムでユーザーの現在のソリューション・ディレクトリーによって更新される。したがって、IBM Security Directory Integrator システム・ストアはソリューション・ディレクトリーごとに固有である。
``tp.server.port=1098```	*削除*	このプロパティーは ```web.server.port=1098``` と再定義されている。

- amc.properties:

次の表に IBM Security Directory Integrator v7.1.1 で削除または変更されたプロパティーをリストします。

表 8. AMC で削除および変更されたプロパティー

前のプロパティー (v7.0 より前)	新規プロパティー	注釈
AMC.auth	*削除*	
monitor.refresh.rate	*削除*	モニター状況表示パネルのリフレッシュ頻度。頻度は分単位で指定されていた。
monitor.startup	*削除*	ユーザーがログイン時に最初に目にするパネルとしてモニター状況表示パネルを設定する。
LDAPHostName LDAPPort LDAPAdminUId LDAPAdminPwd LDAPServerType LDAPBindID LDAPBindPassword LDAPSuffix LdapUserPrefix LDAPUserSuffix LdapGroupPrefix LDAPGroupSuffix LDAPUserObjectClass LDAPGroupObjectClass LDAPGroupMember LDAPUserFilter LDAPGroupFilter LDAPsearchTimeout LDAPsslEnabled LDAPIgnoreCase	*削除*	LDAP 詳細。
com.ibm.di.amc.jdbc.start.mode	新規デフォルト値: Automatic	
com.ibm.di.amc.jdbc.host	新規デフォルト値: Localhost	
com.ibm.di.amc.jdbc.port	新規デフォルト値: 1528	
com.ibm.di.amc.jdbc.sysibm	新規デフォルト値: True	

次の表に IBM Security Directory Integrator v7.0 で追加されたプロパティーをリストします。

表 9. AMC での新規プロパティー

新規プロパティー (デフォルト)	注釈
am.logrotate (10)	AM のログ・ファイルの最大経過日数を決定するために使用。指定された値より古いログ・ファイルは削除される。最小値は 1 で 2147483647 まで増やせる。
amc.session.timeout (20)	AMC セッションの有効期限が切れて AMC から自動的にログアウトされるまでに、ユーザーが活動状態にない最大時間 (分単位) を決定するために使用。値は正整数でなければなりません。
al.workEntries.cacheSize (100)	AssemblyLine が同期モードで開始されたときに AMC で使用される。ここで指定されるキャッシュ・サイズは、作業項目のキャッシュを決定するために使用される。
amc.db.type (derby)	AMC で使用されているデータベースを指定する。
am.api.host (localhost)	Action Manager の RMI 詳細。
am.api.port (13104)	Action Manager の RMI 詳細。

表 9. AMC での新規プロパティー (続き)

新規プロパティー (デフォルト)	注釈
com.ibm.di.server.port.default (1099)	IBM Security Directory Integrator サーバーのデフォルト・ポート。 このプロパティーは 1099 と異なる値をとるよう IBM Security Directory Integrator インストーラーで変更可能。AMC が初めて始動されるとき (またはデータベースが破損した場合)、このプロパティーを読み取られ、その値が新しく作成された AMC データベースに保存される。その後は、AMC がデフォルトの IBM Security Directory Integrator サーバーのインスタンスに接続するときに使用される。

- am_config.properties:

  次の表に IBM Security Directory Integrator v7.1.1 で削除または変更されたプロパティーをリストします。

表 10. AM で削除および変更されたプロパティー

前のプロパティー (v7.0 より前)	新規プロパティー	注釈
com.ibm.di.amc.am.serverapi .fail.interval.time=120 com.ibm.di.amc.am.queryProperty .interval.time=600 com.ibm.di.amc.am.healthAL.interval .time=5	*削除*	これらのプロパティーの値は、ユーザーが「サーバーAPI 障害」トリガー、「プロパティーの場合」トリガー、およびヘルス AL の構成をそれぞれ作成する際に AMC から構成するため、これらのプロパティーはコメント化するべきである。そのため、am-config.properties ファイルに記述されているプロパティーは使用されない。
com.ibm.di.amc.am.queryAL.interval .time	*削除*	
javax.net.ssl.trustStore=$change$ /bin/amc/ActionManager/testadmin.jks javax.net.ssl.keyStore=$change$ /bin/amc/ActionManager/testadmin.jks	javax.net.ssl.trustStore=bin/amc /ActionManager/testadmin.jks javax.net.ssl.keyStore=bin/amc /ActionManager/testadmin.jks	トラストストア・ファイルは現在ソリューション・ディレクトリーに対してローカルである。

次の表に IBM Security Directory Integrator v7.1.1 で追加されたプロパティーをリストします。

表 11. AM での新規プロパティー

新規プロパティー (デフォルト)	注釈
smtp.host= smtp.port= smtp.user= {protect}-smtp.password=	SMTP サーバー詳細。IBM Security Directory Integrator 7.0 で追加。
javax.net.ssl.trustStore= *TDI_Install_dir*/serverapi/testadmin.jks {protect}-javax.net.ssl.trustStorePassword=administrator javax.net.ssl.trustStoreType=jks javax.net.ssl.keyStore=*TDI_Install_dir*/serverapi/testadmin.jks {protect}-javax.net.ssl.keyStorePassword=administrator javax.net.ssl.keyStoreType=jks	Action Manager SSL プロパティー。IBM Security Directory Integrator 7.1 で追加。

表 11. AM での新規プロパティー (続き)

新規プロパティー (デフォルト)	注釈
com.ibm.di.amc.am.encryption.keystore = *TDI_Install_dir*/testserver.jks com.ibm.di.amc.am.encryption.key.alias = server com.ibm.di.amc.am.encryption.keystoretype = jks com.ibm.di.amc.am.encryption.transformation = RSA com.ibm.di.amc.am.stash.file = *TDI_Install_dir*/idisrv.sth	Action Manager 暗号化プロパティー。IBM Security Directory Integrator 7.1 で追加。 これらのプロパティーは、サーバーにより使用される暗号化プロパティーに似ています。便宜上、stash ファイルの場所は、プロパティー com.ibm.di.amc.am.stash.file として追加されています。デフォルトでは、AM はサーバーの鍵ストアおよび AM 保護プロパティーの stash ファイルの暗号化/暗号化解除を再利用。

構成

Directory Integrator のコンポーネントおよび機能のいくつかは変更または削除されました。これらを参照する構成は、手動でマイグレーションする必要があります。以下に影響を受けるコンポーネントおよび機能をリストします。

- チェックポイント・リスタート機能

  この機能は、7.0 で削除されました。これにより、イテレーター・モードをサポートするコネクターは、単純に再接続し、正常な読み取りの回数と同じ回数だけ自動的に前方にスキップするデフォルトの機能を持つだけとなります。これは、このエントリーの数だけ前方にスキップすると、最後に中断した場所に戻ってくることを前提としています。動作が不確定、または適切でなくなる可能性があるため、ほとんどの IBM Security Directory Integrator コネクターはこれを自動的に行いません。しかし、デフォルトの動作はコネクターごとに固有です。正常な読み取りの回数と同じだけ自動的に前方にスキップする機能は、各コネクターで使用可能な新しい再接続オプションであり、「接続エラー」パネルで構成されます。IBM Security Directory Integrator の IBM Knowledge Centerの『*構成*』セクションにある**『構成エディター』->『コネクター・エディター』->『コネクター・エラー』**を参照してください。 処理されたエントリーを自動的にスキップするより多くの機能が必要な場合は、ソリューションで次のオプションの 1 つを使用する必要があります。

  – 動的変更結果セットでは、デルタをイテレーター・モードに構成します。
  – on_connection_failure フックをオーバーライドして、カスタム再接続ロジックを行います。

- 複数の JVM で使用されるシステム・ストアとしての組み込みモードの Derby/Cloudscape

  IBM Security Directory Integrator のデフォルトおよび推奨される動作は、ネットワーク・モードでの Derby の実行です。Derby を組み込みモードで使用し続ける場合は、複数の JVM が同時に同じデータベースを使用しようとすることに関する考慮事項があります。 227 ページの『Derby を使用してシステム・ストアを保持する』を参照してください。データベースのマイグレーションについては、100 ページの『Cloudscape データベースの Derby へのマイグレーション』を参照してください。

- Exchange Changelog コネクター

このコネクターは v7.0 で削除されました。現在は *TDI_install_dir*/examples/ExchangeChangelogConnector に「例」として提供されている、サポート対象外の Exchange Changelog コネクターの使用を検討してください。

- Btree コネクター

  このコネクターは v7.0 でデフォルトのインストールから削除されました。代わりに、セクション 99 ページの『BTree テーブルおよび BTree コネクターのシステム・ストアへのマイグレーション』 で説明されているように、システム・ストア・コネクターを使用します。または、(非サポートの) Btree コネクターを使用します。これは現在 *TDI_install_dir*/examples/BTreeDBConnector に 「例」として提供されています。

- IBM Domino 変更検出コネクター:

  これは 6.0 および 6.1 にのみ適用されます。

  配信モード・パラメーターは削除され、代わりに**状態キーの維持**が使用されます。このパラメーターを使用する、前の構成の動作は次のとおりです。

  – **配信モード**・パラメーターが「Assured once and only once delivery」モードに設定されている場合は、**状態キーの維持**パラメーターは 「読み取り後」に設定されます。これは同じ動作であり、同期状態は、Notes 文書が読み取られた直後に保存されます。

  – **配信モード**・パラメーターが「Normal assured delivery」モードに設定されている場合は、有効な**状態キーの維持**パラメーターのチェックが行われます。パラメーターが見つからない場合、**状態キーの維持**パラメーターは「読み取り後」に設定されます。構成内にパラメーターが検出された場合は、元の値が使用されます。

- IDS 変更ログ・コネクター

  CRAM-MD5 オプションは 7.0 では有効ではありません。他の認証メカニズムを手動で選択しなければなりません。

  IBM Security Directory Server のバージョン 6.2 では、BEREncoder および BERDecoder クラスが `com.ibm.asn1` パッケージから `com.ibm.ldap.bp.asn1` パッケージへ移動されました。IBM Security Directory Server v7.0 からは、前のクラス (`com.ibm.asn1.BEREncoder` および`com.ibm.asn1.BERDecoder`) を直接使用するカスタム・ユーザー・ソリューションは、この変更を反映させるために更新する必要があります。

- EMF XMLToSDO および EMF SDOToXML 関数コンポーネント

  これらは 7.0 では推奨されません。今後は他の機能をご検討ください。

- DSMLv2 パーサー:

  これは 6.0 にのみ適用されます。

  「dsml.request」および「dsml.response」属性は削除されました。これらの属性は、ロー要求オブジェクトおよび応答オブジェクトを ITIM DSMLv2 ライブラリーから供給するために使用されていました。これらの属性を使用している古い構成がある場合は、これらの属性がもう使用されないように古い構成を編集する必

要があります。ロー要求オブジェクトおよび応答オブジェクトを介して使用可能であったすべてのデータは、DSMLv2 パーサーで提供されるその他の属性を介しても使用できます。

- ITIM Agent コネクター

  以前のバージョンの IBM Security Directory Server で ITIM Agent コネクターを使用していた場合は、SSL 接続の構成方法を変更する必要がある場合があります。IBM Security Directory Server のITIM Agent コネクターでは、SSL 認証に JSSE (Java ベース鍵ストア/トラストストア) を使用しているため、旧 ITIM Agent コネクターの「CA Certificate File」パラメーターで証明書の名前を指定する代わりに global.properties またはsolution.properties ファイルで SSL 関連の証明書の詳細を構成する必要があります。以下にこの手順を示します。

  1. 以前「CA Certificate File」パラメーターで指定されていた ITIM Agent の証明書を、例えば *keytool* を使用して (Ikeyman も使用できます) IBM Security Directory Integrator トラストストアにインポートします。

     ```
     keytool -import -file servercertificate.der -keystore tim.jks
     ```

     この例では、トラストストアはファイル tim.jks に保管されます。

  2. global.properties または solution.properties ファイルの「server authentication」セクションでこのトラストストアを構成します。

     ```
     ## server authentication

     javax.net.ssl.trustStore=serverapi¥tim.jks
     {protect}-javax.net.ssl.trustStorePassword=administrator
     javax.net.ssl.trustStoreType=jks
     ```

  これにより、ITIM Agent コネクターは、残りの IBM Security Directory Integrator と同じ JSSE ベースのセキュアな通信アーキテクチャーを使用するようになります。

  既に global.properties または solution.properties に構成されているトラストストア・ファイルがある場合は、新しい証明書を作成する代わりに、その証明書をストアにインポートします。

- XML パーサー:

  v7.0 より前の XML パーサーは名前が変更され、単純 XML パーサーと呼ばれています。現在の XML パーサーは、より機能が充実した新しいパーサーです。特に階層オブジェクトに関する機能が増えています。IBM Security Directory Server の以前のバージョン下で作成された、XML パーサーを参照している構成ファイルは、v7.1 以降にインポートされると、単純 XML パーサーを参照します (クラス名は変更されていないため)。代わりに新しい XML パーサーを使用する場合は、AssemblyLines またはコネクター、またはその両方の構成ファイルを変更する必要があります。新しい XML パーサーの動作を前の XML パーサーと同様にするには、項目タグ・パラメーターおよび値タグ・パラメーターの両方を、単純 XML パーサーで使用される値に設定しなければなりません。

  注: 単純 XML パーサーは「xmldom」という名前のスクリプト変数をエクスポートしますが、新しい XML パーサーはエクスポートしません。新しい XML パーサーは、項目そのものに関するより深い階層を表します。「xmldom」変数に依存し、項目クラスで提供される階層構造を利用できるように再加工できないロジックは、新しい XML パーサーにマイグレーションしてはいけません。

- Castor による Java から XML および XML から Java のための関数コンポーネント:

  IBM Security Directory Server v7.1 以降では、Castor マッピング・ファイルの場所が *TDI_install_dir*/jars/functions/di_castor_mapping.xml から *TDI_install_dir*/etc/di_castor_mapping.xml に変更されました。

  したがって、**Castor マッピング・ファイル** パラメーターのデフォルト値は、現在新しい場所を反映しています。
- HTTP クライアント・コネクター:

  IBM Security Directory Server v7.1 以降では、他の HTTP 要求を受信するために TCP 接続を再利用するつもりがない場合に、HTTP クライアント・コネクターが自動的に HTTP 「接続」ヘッダーに値「close」を設定して送信するよう変更されました。この変更は、HTTP 1.1 の推奨 (http://tools.ietf.org/html/rfc2616#section-14.10) に準拠するためのものです。

  この動作は、HTTP 1.1 の仕様によると必須で、以前は AssemblyLine で自分でコード化する必要がありました。
- HTTP サーバー・コネクター:

  IBM Security Directory Server v7.1 以降では、HTTP サーバー・コネクターは、デフォルトで HTTP の持続接続を使用するよう変更されました。つまり、同じ HTTP クライアントで 1 つの TCP 接続を使用して、複数の HTTP 要求を行うことができます (http://tools.ietf.org/html/rfc2616#section-8.1)。HTTP クライアントは「接続」ヘッダーに値「Keep-Alive」を設定して送信する場合がありますが、持続接続を使用するために、この値が要求されることはありません。アイドルの TCP 接続は、アイドルになってから 20 秒後に自動的にクローズされます。

カスタマイズ・スクリプト

Directory Integrator のスクリプトをカスタマイズしていた場合 (例えば、開始スクリプト `ibmdisrv` や `ibmditk` の PATH または LD_LIBRARY_PATH 環境変数に項目を追加するなど) は、これらのカスタマイズを、新規バージョンの対応するスクリプトに適用する必要があります。

前のバージョンの IBM Security Directory Server は、これらのスクリプトの中で (MY)CLASSPATH 変数を使用していました。現行バージョンには必要なパス情報が組み込まれているため、この変数は必要なくなりました。独自のライブラリーを組み込むために、前述のスクリプトを調整済みの場合にも、CLASSPATH 変数に変更を加える必要はありません。そのライブラリーが適切な場所 (通常は `jars/` ディレクトリー) にあることを確認するだけで十分です。ライブラリーがその場所にあれば、IBM Security Directory Server によって検出されます。別の方法として、global.properties ファイル内の `com.ibm.di.loader.userjars` プロパティーを使用して、独自ライブラリーのディレクトリーがローダーのパスに組み込まれるようにすることもできます。IBM Security Directory Server では、プロパティーでいくつかのディレクトリーや jar ファイルを指定する場合があります。このとき、Java プロパティーの「path.separator」(Linux の場合は「:」、Windows の場合は「;」) で区切って指定します。jar ファイルの場合、TDILoader が再帰的にディレクトリーからクラ

スおよびリソースを含むファイルを検索します。拡張子が「.zip」または「.jar」のファイルのみを検索します。

追加または置き換えられたインストール済み環境の JAR ファイル

インストール済み環境に JAR ファイルを追加していた場合は、それらを新規バージョンにもコピーする必要があります。

現在 IBM Security Directory Server には、Java 7 準拠の JVM (J2SE バージョン 7.0.4) が必須で、これは組み込まれています。Java で独自のコードを開発して、そのコードを JVM ライブラリーにリンクして IBM Security Directory Server ソリューションと統合していた場合は、開発済みのコードを再コンパイルおよび再リンクする必要が生じる場合があります。

インストール済み環境で、オリジナルの JAR ファイルを上書きしていた場合 (例えば、必要な MQ jar を *TDI_install_dir*/jars/3rdparty/IBM に指定するなど) は、同じ変更を新規バージョンでも行う必要があります。

64 ビット Java ランタイム環境 (JRE) は現在 Windows x86-64、Linux x86-64 および Linux s390 で使用されています。32 ビット JRE と比べ、性能低下がいくつかのシナリオで認められています。性能低下に関して潜在的な問題があると思われる場合は、Windows x86-32 インストーラーを、パスワードなしのプラグイン・アクティビティーで使用することができます。

64 ビット JRE を使用する場合は、JNI に依存するいくつかのカスタム・コンポーネント (コネクター、パーサー、FC) で 64 ビット共有ライブラリーが必要であることを知っておく必要があります。

パスワード・シンクロナイザーの構成

- Windows Password Synchronizer

  IBM Security Directory Integrator の IBM Knowledge Centerの『*Password Synchronization Plug-ins*』セクションの『Windows Password Synchronizer』セクションにある『前のインストール環境からのマイグレーション』に記述されている手順に従ってください。
- その他のパスワード・シンクロナイザー

  特別なマイグレーション手順はありません。前のバージョンをアンインストールし、IBM Security Directory Integrator をインストールして必要に応じて構成してください。

重要なデータのバックアップ

以下の情報を参照して、データのバックアップについてよく理解します。

インストーラーでバックアップされるファイル

以下に、さまざまなバージョンアップでインストーラーによってバックアップされるファイルの一覧を示します。

バージョン 6.0 から 7.1 へのアップグレード:

サーバー機能がアップグレードされている場合は、リストしたファイルがバックアップされます。

TDI_install_dir¥global.properties から *TDI_install_dir*¥etc¥global.properties.v60
TDI_install_dir¥serverapi¥testadmin.jks から *TDI_install_dir*¥serverapi¥testadmin.jks.v60
TDI_install_dir¥serverapi¥testadmin.der から *TDI_install_dir*¥serverapi¥testadmin.der.v60
TDI_install_dir¥serverapi¥registry.enc から *TDI_install_dir*¥serverapi¥registry.enc.v60
TDI_install_dir¥serverapi¥registry.txt から *TDI_install_dir*¥serverapi¥registry.txt.v60
TDI_install_dir¥idisrv.sth から *TDI_install_dir*¥idisrv.sth.v60
TDI_install_dir¥testserver.jks から *TDI_install_dir*¥testserver.jks.v60
TDI_install_dir¥testserver.der から *TDI_install_dir*¥testserver.der.v60

さらに、構成ファイルおよび solution.properties がバックアップされます。

バージョン 6.1.x から 7.1 へのアップグレード:

サーバー機能がマイグレーションされている場合は、リストしたファイルがバックアップされます (新しいサフィックスは .v61 または .v611 で、前のバージョンによって異なります)。

TDI_install_dir¥etc¥global.properties から *TDI_install_dir*¥etc¥global.properties.v61x
TDI_install_dir¥serverapi¥testadmin.jks から *TDI_install_dir*¥serverapi¥testadmin.jks.v61x
TDI_install_dir¥serverapi¥testadmin.der から *TDI_install_dir*¥serverapi¥testadmin.der.v61x
TDI_install_dir¥serverapi¥registry.enc から *TDI_install_dir*¥serverapi¥registry.enc.v61x
TDI_install_dir¥serverapi¥registry.txt から *TDI_install_dir*¥serverapi¥registry.txt.v61x
TDI_install_dir¥idisrv.sth から *TDI_install_dir*¥idisrv.sth.v61x
TDI_install_dir¥testserver.jks から *TDI_install_dir*¥testserver.jks.v61x
TDI_install_dir¥testserver.der から *TDI_install_dir*¥testserver.der.v61x
TDI_install_dir¥etc¥reconnect.rules から *TDI_install_dir*¥etc¥reconnect.rules.v61x
TDI_install_dir¥etc¥derby.properties から *TDI_install_dir*¥etc¥derby.properties.v61x
TDI_install_dir¥etc¥jlog.properties から *TDI_install_dir*¥etc¥jlog.properties.v61x
TDI_install_dir¥etc¥log4j.properties から *TDI_install_dir*¥etc¥log4j.properties.v61x
TDI_install_dir¥etc¥tdisrvctl-log4j.properties から *TDI_install_dir*¥etc¥tdisrvctl-log4j.properties.v61x
TDI_install_dir¥etc¥act-jlog.properties to *TDI_install_dir*¥etc¥act-jlog.properties.v611
    (IBM Security Directory Integrator 6.1.1 only)

さらに、構成ファイルおよび solution.properties がバックアップされます。

バージョン 7.0 から 7.1 へのアップグレード:

サーバー機能がアップグレードされている場合は、リストしたファイルがバックアップされます。

TDI_install_dir¥etc¥global.properties から *TDI_install_dir*¥etc¥global.properties.v70
TDI_install_dir¥serverapi¥testadmin.jks から *TDI_install_dir*¥serverapi¥testadmin.jks.v70
TDI_install_dir¥serverapi¥testadmin.der から *TDI_install_dir*¥serverapi¥testadmin.der.v70
TDI_install_dir¥serverapi¥registry.enc から *TDI_install_dir*¥serverapi¥registry.enc.v70
TDI_install_dir¥serverapi¥registry.txt から *TDI_install_dir*¥serverapi¥registry.txt.v70
TDI_install_dir¥idisrv.sth から *TDI_install_dir*¥idisrv.sth.v70
TDI_install_dir¥testserver.jks から *TDI_install_dir*¥testserver.jks.v70
TDI_install_dir¥testserver.der から *TDI_install_dir*¥testserver.der.v70
TDI_install_dir¥etc¥reconnect.rules から *TDI_install_dir*¥etc¥reconnect.rules.v70
TDI_install_dir¥etc¥derby.properties から *TDI_install_dir*¥etc¥derby.properties.v70
TDI_install_dir¥etc¥jlog.properties から *TDI_install_dir*¥etc¥jlog.properties.v70
TDI_install_dir¥etc¥log4j.properties から *TDI_install_dir*¥etc¥log4j.properties.v70
TDI_install_dir¥etc¥tdisrvctl-log4j.properties から *TDI_install_dir*¥etc¥tdisrvctl-log4j.properties.v70
TDI_install_dir¥etc¥act-jlog.properties から *TDI_install_dir*¥etc¥act-jlog.properties.v70

さらに、構成ファイル、ワークスペースおよび solution.properties がバックアップされます。

バージョン 7.1 から 7.1.1 へのアップグレード:

サーバー機能がアップグレードされている場合は、リストしたファイルがバックアップされます。

TDI_install_dir¥etc¥global.properties から *TDI_install_dir*¥backup_tdi¥global.properties
TDI_install_dir¥serverapi¥testadmin.jks から *TDI_install_dir*¥backup_tdi¥testadmin.jks
TDI_install_dir¥serverapi¥testadmin.der から *TDI_install_dir*¥backup_tdi¥testadmin.der
TDI_install_dir¥serverapi¥registry.enc から *TDI_install_dir*¥backup_tdi¥registry.enc
TDI_install_dir¥serverapi¥registry.txt から *TDI_install_dir*¥backup_tdi¥registry.txt
TDI_install_dir¥idisrv.sth から *TDI_install_dir*¥backup_tdi¥idisrv.sth

TDI_install_dir¥testserver.jks から *TDI_install_dir*¥backup_tdi¥testserver.jks
TDI_install_dir¥testserver.der から *TDI_install_dir*¥backup_tdi¥testserver.der
TDI_install_dir¥etc¥reconnect.rules から *TDI_install_dir*¥backup_tdi¥reconnect.rules
TDI_install_dir¥etc¥derby.properties から *TDI_install_dir*¥backup_tdi¥derby.properties
TDI_install_dir¥etc¥jlog.properties から *TDI_install_dir*¥backup_tdi¥jlog.properties
TDI_install_dir¥etc¥log4j.properties から *TDI_install_dir*¥backup_tdi¥log4j.properties
TDI_install_dir¥etc¥tdisrvctl-log4j.properties から *TDI_install_dir*¥backup_tdi¥tdisrvctl-log4j.properties
TDI_install_dir¥etc¥tdimiggbl-log4j.properties から *TDI_install_dir*¥backup_tdi¥tdimiggbl-log4j.properties
TDI_install_dir¥etc¥updateinstaller-log4j.properties から *TDI_install_dir*¥backup_tdi¥updateinstaller-log4j.properties
TDI_install_dir¥etc¥it_registry.properties から *TDI_install_dir*¥backup_tdi¥it_registry.properties
TDI_install_dir¥etc¥tp.xml から *TDI_install_dir*¥backup_tdi¥tp.xml

さらに、TDI_install_dir¥configs フォルダーの構成ファイル、ワークスペースおよび solution.properties がバックアップされます。

バージョン 7.1.1 から 7.2 へのアップグレード:

サーバー機能がアップグレードされている場合は、リストしたファイルがバックアップされます。

TDI_install_dir¥etc¥reconnect.rules
TDI_install_dir¥etc¥derby.properties
TDI_install_dir¥etc¥jlog.properties
TDI_install_dir¥etc¥log4j.properties
TDI_install_dir¥etc¥tdisrvctl-log4j.properties
TDI_install_dir¥etc¥tdimiggbl-log4j.properties
TDI_install_dir¥etc¥updateinstaller-log4j.properties
TDI_install_dir¥etc¥it_registry.properties
TDI_install_dir¥etc¥tp.xml
TDI_install_dir¥etc¥activemq.xml
TDI_install_dir¥etc¥global.properties
solution.properties (デフォルトのソリューション・ディレクトリーに存在する場合)
pwsync.props (インストールされているパスワード・プラグインごと)
AMC データベース
ISC location/runtime/isc/eclipse/plugins/AMC_7.2.0.0¥amc.properties
ISC location/runtime/isc/eclipse/plugins/AMC_7.2.0.0¥conf¥amcdbhandler.properties
ISC location/runtime/isc/eclipse/plugins/AMC_7.2.0.0¥conf¥logging.properties
TDI_install_dir¥bin¥amc¥ActionManager¥am_config.properties
TDI_install_dir¥bin¥amc¥ActionManager¥am_logging.properties

バックアップ・ツール

これらのツールは、バックアップ/リストアの際にインストーラーで使用されます。

これらを手動のマイグレーションで使用することもできます。

backupamc/restoreamc
AMC 構成ファイルのバックアップ/リストアに使用します。

backupamcdb/restoreamcdb
AMC データベースのバックアップ/リストアに使用します。

backupam/restoream
Action Manager (AM) データベースのバックアップ/リストアに使用します。

詳しくは、 325 ページの『AMC および AM コマンド行ユーティリティー』を参照してください。

手動バックアップ

手動バックアップを実行する場合は、以下の手順に注意してください。

手動バックアップとは、ファイルを専用のバックアップ・フォルダーにコピーすることを意味します。反対に、リストアとは、ファイルを専用のバックアップ・フォルダーから、元の場所にコピーすることを意味します。

ファイル間の依存関係を考慮しなければならないケースがあることにご注意ください。相互に依存するファイル・グループは、1 まとめにしてバックアップする必要があります。このようなファイル・グループを以下に示します。

Derby データベース・ファイル
データベースをバックアップする場合は、データベース・ファイルを含むフォルダー全体をバックアップします。例えば、デフォルトのシステム・ストア・データベースをバックアップする場合は、*TDI_install_dir*/TDISysStore フォルダーをコピーします。デフォルトの AMC データベースをバックアップする場合は、*TDI_install_dir*/bin/amc/tdiamcdb フォルダーをコピーします。

IBM WebSphere MQ Everyplace キュー・マネージャー・ファイル
IBM WebSphere MQ Everyplace キュー・マネージャーのフォルダー全体をバックアップします。例えば、デフォルトのシステム・キューをバックアップする場合は、*TDI_install_dir*/MQePWStore フォルダーをコピーします。

CE ワークスペース・ファイル
ワークスペース・フォルダー全体をバックアップします。

OSGI osgi フォルダー全体をバックアップします。

LDAPSync
LDAPSync フォルダー全体をバックアップします。

SCIM SCIM フォルダー全体をバックアップします。

AMC 7.x 構成設定の別の AMC デプロイメントへのマイグレーション

以下の手順に従って、AMC 構成データを新しい AMC デプロイメントにマイグレーションすることができます。

注: AMC 機能は推奨されません。IBM Security Directory Integrator の将来のバージョンでは除去されます。

このセクションでは、AMC 構成データを、新規の AMC デプロイメントにマイグレーションする際に従うべき手順を説明します。これらの説明は、AMC 7.0 以降を、ISC SE から IBM Dashboard Application Services Hub に、または IBM Dashboard Application Services Hub から ISC SE にマイグレーションする場合や、別のマシン上に AMC のミラー・インスタンスを作成する場合に役立ちます。

1. すべての AMC 構成ファイルおよびデータをバックアップします。
   a. stop_tdiamc.bat(sh) スクリプトを使用して、構成データをマイグレーションしている AMC を停止させます。このスクリプトは、AMC がデプロイされているサーバーを停止します。
   b. バックアップする必要のある AMC 構成ファイルがあるディレクトリーを指定して、backupamc.bat (.sh) スクリプトを実行します。
2. すべての AMC 構成データを新規の AMC インスタンスにマイグレーションします。
   a. AMC 構成のマイグレーション先の ISC が、他のマシンに常駐している場合は、AMC バックアップ・ディレクトリーを新しいマシンにコピーする必要があります。

b. マイグレーション先の AMC が停止していることを確認します。AMC の停止については、ステップ 1a を参照してください。

c. AMC 構成情報のバックアップ先ディレクトリーを指定して、restoreamc.bat (.sh) スクリプトを実行します。このコマンドは、AMC 構成ファイルを、マイグレーション先 AMC の正しい場所に置きます。これでマイグレーション・プロセスは完了です。

注:

1. この説明は、AMC 7.0 以降を、他の ISC デプロイメントにマイグレーションするためのものです。
2. この説明は、マイグレーション元のシステムおよび、マイグレーション先のシステムの両方に、IBM Security Directory Integrator インストーラーを使用して、既に AMC がデプロイされていることを前提としています。システムは、ISC SE または IBM Dashboard Application Services Hub で AMC を実行することができます。
3. AMC 構成を、同じマシンの別の AMC デプロイメントにマイグレーションしようとする場合で、使用する IBM Security Directory Integrator デプロイメントとともに出荷されたすべての AMC コマンドが、現在の場所からデプロイメント先の AMC を使用するようにしたい場合は、AMC コマンド行ユーティリティーで構成されている ISC の場所を更新する必要があります。これは、コマンド setISCHome.bat(sh) を使用して行うことができます。このコマンドは、パラメーターとして ISC のインストール・ディレクトリーの場所を使用します。これは、IBM Dashboard Application Services Hub の IBM WebSphere Application Server のインストール場所、および ISC SE の組み込み Web プラットフォームの場所です。このコマンドは、上記の手順 1 と 2 の間で実行する必要があります。

EventHandler から対応する AssemblyLine への変換

標準的な EventHandler の特定部分をマイグレーションすることができます。以下の説明に従って、これらについて詳しく学習します。

EventHandler は IBM Security Directory Integrator バージョン 7.2 に存在しません。したがって、前のソリューションで削除された機能を置き換えるためには、EventHandler 構成を、サーバー/イテレーター構成、または変更ログ・コネクター構成にマイグレーションする必要があります。

EventHandler ごとに対応する AssemblyLine を作成する必要があります。この時、EventHandler に対応するサーバー・コネクターおよびイテレーター・コネクターを AssemblyLine の「供給」セクションに挿入する必要があります。さらに、コネクター・パラメーターを設定します。これは、EventHandler/コネクターのペアごとに固有ですが、一般にコネクター・パラメーターには対応する EventHandler パラメーター (通常は同名) と同じ値を設定する必要があります。

EventHandler に構成されるすべての処理を、AssemblyLine の「フロー」セクションに再インプリメントする必要があります。

「enabled」EventHandler パラメーター (「サービスの自動始動」と呼ばれるパラメーター)の機能は、AssemblyLine でも使用可能です。IBM Security Directory

Integrator サーバーの始動直後に AssemblyLine を開始するには、構成エディターのワークスペースでナビゲーターの**ソリューションのロギングおよび設定**セクションに移動して、使用する AssemblyLine を追加します。

一般的には、EventHandler は何らかのイベントが発生した場合に、断片的にロジックを実行します。「イベント」は、EventHandler ごとに意味が異なります。 HTTP EventHandler の場合、「イベント」とは HTTP 要求のことです。IBM Security Directory Integrator EventHandler の場合は、「イベント」とは、IBM LDAP ディレクトリーから送られる変更通知になります。

標準的な EventHandler を部分的にマイグレーションする場合の一般ガイドラインを以下に示します。これらは、7.0 より前の構成エディターにある EventHandler の UI タブのタイトルに基づいてまとめられています。

フック

EventHandler の「プロローグ」フックは、AssemblyLine の「プロローグ - 開始後」フックに対応します。このフックは着信「イベント」のたびに起動されます。

EventHandler の「エピローグ」フックは AssemblyLine の「エピローグ: クローズ後」フックに対応します。このフックは、着信「イベント」が処理されるたびに一度起動されます。

「プロローグ」EventHandler フックおよび「エピローグ」EventHandler フックの両方で、「イベント」項目は「conn」および「event」の名前でアクセス可能です。しかし、AssemblyLine フックでは、「conn」または「event」の代わりに「work」を使用するようスクリプトを変更する必要があります。

EventHandler の「シャットダウン要求」フックは、「シャットダウン要求」AssemblyLine フックに対応します。

アクション・マップ

EventHandler のアクション・マップは、「イベント」が到着したときに起こすアクションを定義します。EventHandler と置き換えようとしている AssemblyLine のロジックには、同じアクションを作成する必要があります。

例えば、アクション・マップで、イベントの属性「x」が「3£ に等しいときにあるカスタム・スクリプトを実行するということを規定した場合は、属性「x」が 3 と等しいことをチェックして、あるスクリプトを実行する「IF」コンポーネントを AssemblyLine に追加してください。

ロギング

EventHandler のカスタム・ログ Appender を構成してある場合は、その EventHandler と置き換えようとしている AssemblyLine のログ設定で、同じ Appender を構成する必要があります。

構成 これらの構成パラメーターは、EventHandler ごとに固有です。これらのマイグレーション方法についての説明は、下記のサブセクションを参照してください。サブセクション名は、対応するコネクターの名前から取っています。

TCP サーバー・コネクター

ここで説明する手順に従って、古い EventHandler の構成を新しいコネクターの構成に複製することができます。

1. 新規 AssemblyLine を作成し、TCP サーバー・コネクターをこの AssemblyLine に挿入します。
2. **tcp.port** および **debug** コネクター・パラメーターを、対応する EventHandler パラメーターの値に設定します。
3. **useSSL** および **requireClientAuth** コネクター・パラメーターを、false (構成エディターではチェックを外す) に設定します。

メールボックス・コネクター

メールボックス EventHandler を使用する構成は、以下の手順を使用してマイグレーションする必要があります。

IBM Security Directory Integrator IBM Security Directory Integrator メールボックス・コネクターは、IBM Security Directory Integrator v6.0 メールボックス・コネクターと互換性があるため、IBM Security Directory Integrator v6.0 メールボックス・コネクターを使用する既存の構成をマイグレーションする必要はありません。

1. 新規 AssemblyLine を作成し、メールボックス・コネクターをこの AssemblyLine に挿入します。
2. **mailServer** EventHandler パラメーターの内容を、同じ名前のコネクター・パラメーターにコピーします。
3. **mailProtocol** コネクター・パラメーターを、同じ名前の EventHandler パラメーターの値に設定します。
4. **mailUser** および **mailPassword** EventHandler パラメーターの内容を、同じ名前のメールボックス・コネクター・パラメーターにコピーします。
5. **mailFolder** EventHandler パラメーターの内容を、同じ名前のコネクター・パラメーターにコピーします。
6. **pollInterval** EventHandler パラメーターの内容を、同じ名前のコネクター・パラメーターにコピーします。
7. 使用可能化 EventHandler パラメーターが true の場合は、作成した AssemblyLine を、構成エディターの「構成」-> 「AutoStart」フォルダーに追加します。これで IBM Security Directory Integrator サーバーは、作成した AssemblyLine を開始時に始動します。
8. **debug** EventHandler パラメーターが true の場合は、同じ名前のコネクター・パラメーターを true に設定します。

JMX コネクター

以下の手順を使用して、IBM Security Directory Integrator v6.0 JMX EventHandler を使用する既存の構成を、JMX コネクターを使用する IBM Security Directory Integrator バージョン 7.2 構成に変換することができます。

1. 新規 AssemblyLine を作成し、JMX コネクターをこの AssemblyLine に挿入します。

2. **eventTypes** JMX EventHandler パラメーターの内容を、同じ名前の JMX コネクター・パラメーターにコピーします。
3. **モード** コネクター・パラメーターで「ローカル」を選択します。
4. **url** コネクター・パラメーターはブランクのままにしておきます。
5. **allMBeans** コネクター・パラメーターを `true` に設定します。
6. **mBeanTypes** コネクター・パラメーターはブランクのままにしておきます。

SNMP サーバー・コネクター

ここで提供する情報を使用すると、SNMP サーバー・コネクターについて詳しく知ることができます。

IBM Security Directory Integrator SNMP サーバー・コネクターは、シングル・スレッド・モードのサポートを除くすべての IBM Security Directory Integrator v6.0 SNMP EventHandler の機能を提供します。IBM Security Directory Integrator SNMP サーバー・コネクターは、マルチスレッド・モードでのみ動作します。SNMP EventHandler を使用する既存の IBM Security Directory Integrator v6.0 の構成を、SNMP サーバー・コネクターと一緒に AssemblyLine を使用する IBM Security Directory Integrator v7.0 の構成にマイグレーションする必要がある場合は、次の手順に従う必要があります。

1. 新規 AssemblyLine を作成します。
2. この AssemblyLine に SNMP サーバー・コネクターのインスタンスを挿入します。
3. **udp.port** コネクター・パラメーターを、使用する SNMP EventHandler の構成でこのパラメーターに設定されている値に設定します。
4. **snmp.community** コネクター・パラメーターを、使用する SNMP EventHandler の構成でこのパラメーターに設定されている値に設定します。
5. 使用する SNMP EventHandler が、IBM Security Directory Integrator サーバーで「Auto-started」に構成されている場合は、作成した新規 AssemblyLine を、構成エディターの「構成」-> 「AutoStart」フォルダーに追加します。

IBM Security Directory Server 変更ログ・コネクター

IBM Security Directory Server EventHandler を使用する既存の構成は、以下の情報を使用して、IBM Security Directory Integrator 変更ログ・コネクターを使用するようにマイグレーションできます。

1. 次のコネクター・パラメーターを、同じ名前の EventHandler パラメーターの値に設定します。**ldapUrl**、**ldapUsername**、**ldapPassword**、**ldapAuthenticationMethod**、**ldapUseSSL**、**ldapSearchBase**。
2. **jndiExtraProviderParams** コネクター・パラメーターは空のままにしておきます。
3. **iteratorStateKey** コネクター・パラメーターを、固有 ID に設定します。つまり、対応する状態がシステム・ストアに保存されていない ID に設定します。
4. **nsChangenumber** コネクター・パラメーターを、EventHandler が処理すると想定される次の変更番号に設定します。EventHandler が処理した最後の変更番号は、通常は外部プロパティー・ファイルに保存されています。これは、**ldapChangeNumberFileName** パラメーターで参照されます。

5. **stateKeyPersistence** コネクター・パラメーターを、「読み取り後」に設定します (EventHandler は、最後に受け取った変更番号を、変更ログ項目を読み取った後、それを処理するためにディスパッチする前に、ファイル・バックエンドに書き込みます)。
6. **mergeMode** コネクター・パラメーターを、「変更ログおよび変更データのマージ」に設定します。これで、変更ログ属性 (changenumber や targetdn など) が項目の属性として必ず表示されます。
7. **useNotifications** コネクター・パラメーターを true に設定します。
8. **batchRetrieval** コネクター・パラメーターを false に設定します。

注: EventHandler とは反対に、コネクターでは、コネクターが通知をlisten する対象として、ディレクトリー・ツリーの一部を選択することができません。コネクターは、ディレクトリー・ツリー全体の変更をサブスクライブします (コネクターには **ldapEventBase** および **ldapSearchScope** EventHandler パラメーターに相当するものがありません)。これが重大な問題である場合は、ソリューションにカスタム・フィルタリングをインプリメントして、コネクターのこの制限を克服することができます。

HTTP サーバー・コネクター

HTTP EventHandler を使用する構成は、以下の手順を使用して、HTTP サーバー・コネクターを使用するようにマイグレーションできます。

1. **tcpPort** コネクター・パラメーターを、EventHandler の **Port**パラメーターの値に設定します。
2. **backlog** コネクター・パラメーターは空のままにしておきます。
3. **contentType** コネクター・パラメーターを「text/html」に設定します。
4. **tcpDataAsProperties** コネクター・パラメーターを `true` に設定します (EventHandler は常に TCP 情報をプロパティーとして返します)。
5. **headersAsProperties** コネクター・パラメーターを EventHandler の **headersAsProperties** の値に設定します。
6. EventHandler が HTTP 基本認証 (BA) を使用する場合 (つまり、EventHandler に構成されている認証コネクターがある場合)は、**httpAuth** コネクター・パラメーターを true に設定します。
7. EventHandler が、HTTP 基本認証 (BA) を使用する場合は、**authRealm** コネクター・パラメーターを **authrealm** EventHandler パラメーターの値に設定します。**authrealm** EventHandler パラメーターが存在しない、または空の場合は、**authRealm** コネクター・パラメーターを「IBM-Directory-Integrator」に設定します。
8. **authConnector** コネクター・パラメーターを EventHandler の **AuthConnector** パラメーターの値に設定します。
9. **useSSL** コネクター・パラメーターを、EventHandler の **useSSL**パラメーターの値に設定します。
10. **needClientAuth** コネクターを false に設定します (EventHandler は SSL クライアント認証をサポートしません)。
11. **msgChunked** コネクター・パラメーターを false に設定します (EventHandler は、HTTP 応答のチャンキングをサポートしません)。

LDAP サーバー・コネクター

LDAP サーバー EventHandler を使用する構成は、以下の手順を使用して、LDAP サーバー・コネクターを使用するようにマイグレーションできます。

1. **ldapPort** コネクター・パラメーターを EventHandler の **tcp.port** パラメーターの値に設定します。
2. **backlog** コネクター・パラメーターは空のままにしておきます。
3. **ldapUseSSL** コネクター・パラメーターを EventHandler の**ldapUseSSL** パラメーターの値に設定します。
4. **charset** コネクター・パラメーターを、EventHandler の **charset**パラメーターの値に設定します。
5. **ldapBinaryAttributes** コネクター・パラメーターを EventHandler の **binary** パラメーターの値に設定します。

Sun Directory 変更検出コネクター

ここで提供する情報を参照して、Sun Directory 変更検出コネクターの詳細について理解できます。

LDAP EventHandler は、ディレクトリー・ツリーの変更に関する通知をキャッチします。EventHandler は変更ログを使用しないため、リアルタイム通知のみを受信します。Sun ディレクトリー変更検出コネクターは、基本的にはリアルタイム配信モードで実行した場合と同じ機能を提供します。ただし、少し違いがあります。

コネクターには **ldapSearchFilter** および**ldapSearchScope** EventHandler パラメーターに相当するものがありません。EventHandler と同じ機能を得るには、受信通知のセットを制限するカスタム・フィルタリングをインプリメントする必要があります。

戻されるデータのスキーマは、コネクターと EventHandler では異なります。コネクターは、コネクターが返す項目ごとにデルタ・タグ付けを適用しますが、EventHandler は、「ldap.operation」プロパティーの変更のタイプを提供します。スキーマに関する詳細は、各コンポーネントの文書を参照してください。

上記の考慮事項が解決したら、次のようにして、LDAP EventHandler を使用する既存の構成を、Sun ディレクトリー変更検出コネクターを使用するようにマイグレーションすることができます。

1. 次のコネクター・パラメーターを、同じ名前の EventHandler パラメーターの値に設定します。**ldapUrl**、**ldapUsername**、**ldapPassword**、**ldapAuthenticationMethod**、**ldapUseSSL**、**ldapSearchBase**。
2. **jndiExtraProviderParams** コネクター・パラメーターは空のままにしておきます。
3. **deliveryMode** コネクター・パラメーターを「Realtime」に設定します (EventHandler は変更ログを使用せず、リアルタイム通知のみを catch します)。
4. **mergeMode** コネクター・パラメーターを「変更データのみを戻す」に設定します (リアルタイム配信モードのコネクターでは、変更ログは使用されません)。

Active Directory 変更検出コネクター

以下の情報を参照することで、Active Directory 変更検出コネクターについて詳しく知ることができます。

AD 変更ログ EventHandler から Active Directory 変更検出コネクターへのマイグレーションは、多くの点で簡単です。EventHandler 自身が古いバージョンのこのコネクター、つまり Active Directory 変更検出コネクターを、AD から変更を取得するために取り込んでしまったためです。

EventHandler と同じように、対応するコネクターも、同期化処理中はいつでも割り込みされることがあります。その場合、コネクターはその状態をユーザー・プロパティー・ストアに保管します。EventHandler およびコネクターはどちらも、この処理では uSNChanged メカニズムに依存し、プロパティー・ストアに USN 番号を保管しています。EventHandler およびコネクターは、現在の USN 同期値を取り出すために、USN API も提供します。違いは、EventHandler `getUSNvalues` メソッドでは、次の属性付きの項目を返します:

```
START_USN
END_USN
CURRENT_USN_CREATED
CURRENT_USN_CHANGEDT
```

これに対し、コネクターは現在の同期値を *long* で返すということです。

もう 1 つの違いは、AD EventHandler は、変更通知をブロックおよび受信するために、内部的に LDAP コネクターを初期化します。この動作は ADCD コネクターで **useNotifications** パラメーターを使用可能化するとシミュレートすることもできます。

EventHandler ベースのソリューションから、コネクター・ベースのソリューションへマイグレーションするためには、次の手順を実行する必要があります。

1. 1. イテレーター・モードの Active Directory 変更検出コネクターのインスタンスを持つ新規 AssemblyLine を作成します。
2. **ldapUrl**、**ldapUsername**、**ldapPassword** および **ldapAuthenticationMethod** を、これらの接続パラメーターが EventHandler の構成で設定されている値に設定します。
3. 前の構成の値に従って、SSL 接続が使用されるかどうかを指定します。
4. **ldapSearchBase** EH パラメーターの内容を、コネクター構成の同じパラメーターにコピーします。
5. **persistentParameterName** EH パラメーターの内容を、persistentStateKey コネクター・パラメーターにコピーします。
6. パラメーター **useNotifications** を true に設定します。
7. **startAt** パラメーターを EH での値に従って設定します。
8. 他のコネクター・パラメーターはそのままにしておきます。
9. 新規 AL から呼び出される EventHandler のアクション・マップ・セクションのすべてのロジックを転送します。

DSMLv2 SOAP サーバー・コネクター

以下の情報を使用することで、DSMLv2 SOAP サーバー・コネクターについて詳しく知ることができます。

DSMLv2 EventHandler から DSMLv2 SOAP サーバー・コネクターへのマイグレーションには、DSMLv2 SOAP サーバー・コネクターのソリューションに統合されるように、以前に EventHandler で使用されている AssemblyLines に手を加える必要があります。これは、コア・アーキテクチャーが変更されたためです。現行では、単一の AssemblyLine がすべての操作を処理します。そのため、異なるタイプの DSMLv2 操作の処理を担っている古い AssemblyLines ロジックはすべて、DSMLv2 SOAP サーバー・コネクターを含む新しい AssemblyLine に取り込むか、AssemblyLine コネクターを使用して、これらの古いロジックを呼び出す必要があります。この目的のために、ブランチ・コンポーネントを使用して、(`dsml.operation` 属性で使用可能な) 特定の DSMLv2 操作のロジックを分離することができます。

DSMLv2 EventHandler を使用する構成を、DSMLv2 SOAP サーバー・コネクターを使用する類似の構成にマイグレーションするには、次の手順に従います。

1. サーバー・モードの DSMLv2 SOAP サーバー・コネクターのインスタンスを持つ新規 AssemblyLine を作成します。
2. EH **port** パラメーターの内容を**dsmlPort** コネクター・パラメーターにコピーします。
3. **authRealm**、**useSSL**、**binaryAttributes** および **msgChunked** を、これらの接続パラメーターが EventHandler の構成で設定されている値に設定します。
4. EH 構成でパラメーターとしてリストされている DSMLv2 操作ごとにブランチ・コンポーネントを作成し、対応する古い AssemblyLine にインプリメントされているロジックをブランチで利用します。これには、適切な AL コンポーネントをブランチに転送する方法と、AssemblyLine コネクターを使用して、古い AL そのものを呼び出す方法があります。どちらの場合も、ネーミング・コンテキストはもう必要ありません。
5. twoEH **WaySSL** パラメーターの内容を、**needClientAuth** コネクター・パラメーターにコピーします。
6. EH 属性 **headerAsProperties** をコネクターに渡すことはできません。これは、コネクターで内部的に初期化される HTTP パーサー が、常にこの値を「false」に設定するよう構成されているためです。そのため、ソリューションがプロパティーとしてのヘッダーにアクセスする場合は、この目的で属性を使用するよう修正する必要があります (`getProperty()` の代わりに `getAttribute()` を使用します)。
7. 準拠のため、**soapbinding** コネクター属性は、「false」に設定する必要があります。これは、EH で内部的に使用される DSMLv2 パーサーがこれを利用しないためです。
8. DSMLv2 EventHandler の構成で **authConnector** が指定されている場合は、コネクターの HTTP 基本認証 (BA) を有効にし、適切なロジックを「接続受け入れ後」フックにインプリメントしなければなりません (例えば、オーセンティケーター・コネクターを初期化し、その `lookup()` メソッドを、属性「username」お

よび「password」付きの項目を検索基準として使用して呼び出します。EventHandler と同様に、項目が返された場合は、認証は正常であると見なされます)。

9. コネクターで内部的に使用される DSMLv2 パーサーの **indentoutput** パラメーターは、EH で使用されるものと異なり、設定することができません。

BTree テーブルおよび BTree コネクターのシステム・ストアへのマイグレーション

ここでは、BTree テーブルおよび BTree コネクターのシステム・ストアへのマイグレーションについて学習できます。

BTree コネクターは推奨されていないため、現行ではサポートされない例としてのみ提供されています。そのため、デルタ情報の維持方法を以前の Btree オブジェクトからシステム・ストアのデルタ・テーブルに変更することもできます。これを行うのに最適な状況は、デルタ情報が空であり (例えば、新しくベースラインを確立する場合)、Btree オブジェクトからシステム・ストア・デルタ・テーブルに切り替える場合です。Btree オブジェクトのファイル名を保持するために使用するパラメーターは、ここではデータベースの表名を示しているため、この値の編集が必要な場合があります。

IBM Security Directory Integrator 項目を保管するために BTree コネクターの代わりにシステム・ストア・コネクターを使用するようソリューションを変更するのは簡単です。これは、どちらのコネクターも、鍵属性名および選択モード属性を指定するときに、同じロジックに従っているためです。唯一の違いは、 BTree データベースが内在するのに対し、システム・ストア・コネクターでは、データベース (例えば、組み込み Derby データベースなど) を事前定義し、保管先のテーブルを指定する必要があることです。

システム・ストア・コネクターを使用して、他の Java オブジェクトを保管することと、それらを BTree で保管することでは大きく異なります。システム・ストア・コネクターを使用する場合の方が、より多くの変換が必要になります。次のソリューションでは、内在する BTree データベースに Java オブジェクトを保管しています。このソリューションは、バックエンド・データベースに直接アクセスを行わないため、これをシステム・ストア・コネクターにこのまま適用することはできません。

```
scripts var bt = system.getConnector("btreedb");
bt.initialze (null); var db = bt.getDatabase();
db.insert ("my key", new java.lang.String("my value"));
var value = db.search ("my key"); value = value + " - modified";
db.replace ("my key", value);
```

この代わりに、コネクター API からの標準的なメソッド (`put()`、`find()` および `modify()`) を使用することができますが、 オブジェクトは最初に項目オブジェクトにラップする必要があります。項目オブジェクトは後でシステム・ストアに保管することができます。

Cloudscape データベースの Derby へのマイグレーション

以下の説明に従って、Cloudscape データベースを Derby にマイグレーションすることができます。

IBM Security Directory Integrator バージョン 7.2 は Apache Derby バージョン 10.8 をバンドルされたデータベースとして使用します。これはシステム・ストアでデフォルトで使用されます。既存の Cloudscape または Derby データベース (以前のバージョンの IBM Security Directory Integrator を使用して作成された) を、IBM Security Directory Integrator バージョン 7.2 を使用できるようにマイグレーションする必要があります。IBM Security Directory Integrator バージョン 7.2 に付属しているApache Derby バージョン 10.8 ドライバーは、以前のバージョンの Cloudscape との通信には使用できません。

詳細、および Cloudscape/Apache Derby バージョン 10.8 とその以前のバージョンとの違いについては、次の Web ページを参照してください。http://publibfp.boulder.ibm.com/epubs/html/c1894710.html。

即時に影響のある特筆すべき違いを次に示します。

- `long varbinary` データ・タイプは現在サポートされていません。その代わりに、BLOB データ・タイプが提供されました(Derby は DB2® との互換性があります)。そのため、long varbinary データ・タイプを使用していたすべての SQL ステートメントは、 BLOB を使用するように変更する必要があります。
- JDBC Java パッケージ名は、以前のリリースの com.ibm.db2j.* から Derby バージョン 10 の org.apache.derby.* に変更されました。
- Derby バージョン 10 (組み込みモード・アクセス/ネットワーク・モード・アクセス) の JDBC URL は Cloudscape 5.1 とは異なります。そのため、`global.properties` / `solution.properties` に示される JDBC プロパティーも現行バージョンの IBM Security Directory Integrator 用に変更されました。

表 12. JDBC URL の違い

接続タイプ	**Cloudscape バージョン 5.1**	**Derby バージョン 10**
組み込みの Derby / Cloudscape	jdbc:db2j:	jdbc:derby:
DB2 JDBC ユニバーサル・データベース・ドライバー (ネットワーク・モード)	jdbc:db2j:net	jdbc:derby:net (使用を推奨しない)
DerbyClient ドライバー	-	jdbc:derby (推奨)

Derby チームによって、Cloudscape 5.1 データベースを新しい Derby バージョン 10 データベースにマイグレーションするマイグレーション・ユーティリティーが提供されました。これを使用すると、すべてのテーブルとその対応するデータを、新しく生成された Derby バージョン 10 データベースにマイグレーションできます。このユーティリティーは、varbinary データ・タイプのすべてのテーブルを、BLOB データ・タイプに変更するので、マイグレーション・プロセスがとても楽になります。

このユーティリティーは、IBM Security Directory Integrator にバンドルされており、migrateCS.bat(sh) というマイグレーション・ツールを起動するラッパー・スクリプトと一緒に、*TDI_install_dir*`/tools/CSMigration` フォルダーにあります。IBM Security Directory Integrator バージョン 6.0 を使用して作成された Cloudscape

5.1 システム・ストア・データベースを Derby バージョン 10 にマイグレーションするには、次のようにマイグレーション・スクリプトを起動する必要があります。

```
migrateCS [Path_of_CloudscapeV51_Database] [Path_of_new_DerbyV10_Database]
```

注: このマイグレーション・ユーティリティーを使用できるのは、Cloudscape 5.1 から Derby バージョン 10 へのマイグレーションの場合に限られます。したがって、IBM Security Directory Integrator バージョン 6.0 からバージョン 6.1.1 以降にシステム・ストア・データベースをマイグレーションする場合には、*TDI_install_dir*/tools/CSMigration/migratCS.bat(sh) ファイルを使用できます。ただし、IBM Security Directory Integrator バージョン 6.1.1 からそれより後のバージョンにシステム・ストア・データベースをマイグレーションする場合は、バージョン 6.1.1 のインストール・ディレクトリーにある古い TDISysStore を、新規バージョンの新規インストール済み環境に単純にコピーする必要があります。

新しい Derby データベースの場所については、よく考える必要があるかもしれません。 IBM Security Directory Integrator v6.0 および v6.1.x では、システム・ストア・データベースは、多くの場合 IBM Security Directory Integrator のインストール・ディレクトリーに配置されていましたが、これはいろいろな理由で適切な場所ではありません。IBM Security Directory Integrator バージョン 7.2 では、ソリューション・ディレクトリーとインストール・ディレクトリーは別にして使用することを強くお勧めします。

データのマイグレーションに加えて、新しい JDBC URL パラメーターを取り込むために、(マイグレーション・ツールを使用して、または手動で) global.properties ファイル/ solution.properties ファイルも変更する必要があります。

マイグレーション・ツールを使用したグローバル・プロパティー・ファイルおよびソリューション・プロパティー・ファイルのマイグレーション

TDI_install_dir/bin ディレクトリーにある tdimiggbl ツールを使用して、IBM Security Directory Integrator 6.x 以降の global.properties ファイルをバージョン 7.2 にマイグレーションします。

ファイル名は、Windows の場合は tdimiggbl.bat で、UNIX/LINUX の場合は tdimiggbl.sh です。tdimggbl.bat(sh) のロギングを制御するには、tdimiggbl-4log4j.properties ファイルを使用します。

コマンドが次の場合の使用法。

```
tdimiggbl -f propfile [-b backfile] [-n newfile] [-v] [-?]
ここで、
-f propfile - マイグレーションするファイルの名前
-b backfile - 指定された名前でオリジナル・ファイルをバックアップします。
-n newfile - マイグレーションされたファイルに付ける名前
-s  dir          - ソリューション・ディレクトリーが配置されている作業ディレクトリー
-v               - 冗長モードを有効にします。
-?               - 使用法ステートメントを出力します。
```

IBM Security Directory Integrator のインストール時に、インストーラーは既存の global.properties ファイルをバックアップしてからこのコマンドを呼び出して global.properties をマイグレーションします。

マイグレーション・ツールは、global.properties ファイル (または必要に応じて solution.properties ファイル) を IBM Security Directory Integrator の最新バージョンまでマイグレーションしようとします。ツール (tdimiggbl) は、global.properties ファイルがどのリリースまで遡るかについては考慮せず、IBM Security Directory Integrator version 6.0 以降の global.properties ファイルを処理できます。 また、ツールは、マイグレーション・ツールによるマイグレーションとして適切でないマイグレーション手順が具体的に宣言されない限り、すべてのマイグレーションの変更を適応しようとします。このような場合は、マイグレーション手順を手動で実行してください。

マイグレーション・ツールのアクティビティーは、いくつかのステージに分解されます。ツールのアクティビティーを順番に示します。

1. Derby (Cloudscape) データベースをマイグレーションする必要があるかどうかをチェックします (IBM Security Directory Integrator 6.0 マイグレーション)。
2. すべてのマイグレーション・アクションを次の順番で実行します。
   a. 削除アクション。
   b. 追加アクション。
   c. Derby (Cloudscape) マイグレーション・ファイルの変更 (IBM Security Directory Integrator 6.0 マイグレーションのみ対象、必要な場合のみ)。
   d. マイグレーション変更アクション。
3. Derby (Cloudscape) マイグレーション・ツール migrateCS を呼び出して、データベースを Derby の現行バージョンまでマイグレーションします (IBM Security Directory Integrator 6.0 マイグレーションのみ)。

   各アクション・セット (マイグレーション変更アクションなど) で、マイグレーション・ツールは、最も古いリリースから開始して、最新リリースまで、マイグレーション・アクションを実行しようとします。 IBM Security Directory Integrator 6.0 からのマイグレーションの場合、呼び出し元は別々に Derby (Cloudscape) マイグレーション・ツールを呼び出して、データベースを Derby の現行バージョンまでマイグレーションする必要があります。tdimiggbl ツールは、プロパティー・ファイル自体への必要な Derby (Cloudscape) の変更のみを行います。
4. エラー・メッセージをログに記録するために、log4j ロギング API を使用します。

   起動スクリプト (bat または sh) ファイルで、log4j 構成ファイルを指定します。コマンドでは、tdimiggbl-log4j.properties と呼ばれるファイルを使用して、log4j ロギングをセットアップします。IBM Security Directory Integrator のインストール・ディレクトリーが指定されていない場合、コマンドはディレクトリーをソリューション・ディレクトリーに変更し、ソリューション・ディレクトリーの tdimiggbl-log4j.properties ファイルを使用します。

マイグレーション・ツールを使用したパスワード・プラグイン・プロパティー・ファイルのマイグレーション

IBM Security Directory Integrator には、インストール済みのパスワード・プラグインごとにある pwsync.props ファイルをアップグレードするためのマイグレーション・ユーティリティーが含まれています。このユーティリティーは、migpwsync と呼び、ネイティブ・プラグインおよび JavaProxy の両方で読み取られる pwsync.props ファイルをマイグレーションするために提供されています。

migpwsync ユーティリティーは、*TDI_install_dir*/pwd_plugins/bin ディレクトリーに含まれます。

ユーティリティーには次のようなオプションがあります。

- **-?** - このオプションを使用すると、ユーティリティーはヘルプ情報をプリントし終了します。
- **-v** - このオプションを使用すると、ユーティリティーはより詳細な情報を標準出力にプリントします。
- **-f** - これは、pwsync.props ファイルの場所を指定するために使用する必須指定のオプションです。
- **-b** - これは、バックアップとして使用するファイルの場所を指定するオプションです。これは任意指定のフィールドで、指定されない場合は、-f オプションの値に 「.backup」を付加して使用されます。
- **-n** - このオプションは、マイグレーションされた情報を書き込むファイルの場所を指定します。これは任意指定のフィールドで、指定されない場合は、-f の値がマイグレーションされた構成の出力場所として使用されます。

例

- PAM プラグインの構成ファイルのマイグレーション

```
# TDI_install_dir/pwd_plugins/bin/migpwsync.sh
     -f TDI_install_dir/pwd_plugins/pam/pwsync.props
```

- Windows プラグインの構成ファイルのマイグレーション :

```
> TDI_install_dir¥pwd_plugins¥bin¥migpwsync.bat
     - f TDI_install_dir¥pwd_plugins¥windows¥pwsync.props
```

注:

1. インストーラーは、インストール時にインストーラーが *TDI_install_dir*/pwd_plugins ディレクトリーにセットアップしたすべての pwsync.props ファイルを更新します。pwsync.props ファイルのいずれかを移動した場合は、上記のようなコマンドを使用して、移動したファイルを手動でマイグレーションする必要があります。
2. migpwsync ユーティリティーは、現行ディレクトリーをプラグインのホーム・ディレクトリー (*TDI_install_dir*/pwd_plugins) に変更します。指定されたファイル・パスが絶対パスでない場合は、そのディレクトリーへの相対パスと見なされます。
3. 古い pwsync.props ファイルをマイグレーションした後に、以下の ActiveMQ 関連プロパティーを追加します (ActiveMQ をデフォルトの JMS パスワード・ストアとして構成する場合):

- jmsDriverClass=com.ibm.di.plugin.pwstore.jms.driver.ActiveMQ
- jms.broker=<JMS Server address>。例えば、jms.broker=tcp://<activeMQhost>:61616 または jms.broker=ssl://<activeMQhost>:61617

4. ActiveMQ をデフォルトの JMS パスワード・ストアとして構成するには、`pwsync.props` ファイル内の `jms.clientId` プロパティーを設定します。

第 6 章 セキュリティー

以下の詳細説明を参照することで、セキュリティー機能を動作させ、これらの機能を使用して問題に対処することできます。

セキュリティー機能は、IBM Security Directory Integrator (IBM Security Directory Integrator) 全体に組み込まれています。IBM Security Directory Integrator からリモート・システムへのアクセスを保護する機能、リモート・システムから IBM Security Directory Integrator へのアクセスを保護する機能、リモート・システムへのユーザー信任状などのデータを保護する機構を提供する機能があります。

このセクションで説明する機能の多くは、IBM Security Directory Integrator をセキュアな環境においてスタンドアロン・モードで実行している場合は不要です。ただし、これらの機能は、他のシステムから、リモート Web Admin Console (AMC) 管理ツールまたは IBM Security Directory Integrator リモート・サーバー API などを介して IBM Security Directory Integrator と通信する必要がある場合に便利です。さらに、複数のユーザーが IBM Security Directory Integrator サーバーにアクセスする場合は、IBM Security Directory Integrator が実行する統合ルールの保全性を維持すると同時に、機密データへのアクセスを保護することも必要なことがあります。

このセクションでは以下の機能について説明します。

1. 『鍵、証明書、および鍵ストアの管理』
2. 112 ページの『Secure Sockets Layer (SSL) サポート』
3. 121 ページの『リモート・サーバー API』
4. 149 ページの『IBM Security Directory Integrator サーバー・インスタンス・セキュリティー』
5. 160 ページの『各種設定構成ファイルの機能』
6. 168 ページの『Web 管理コンソールのセキュリティー』
7. 164 ページの『セキュリティーを扱う構成ファイルおよびプロパティーの要約』
8. 168 ページの『セキュリティーのその他の側面』

このセクションでは、個々の IBM Security Directory Integrator コンポーネントのすべてのセキュリティー機能について説明するわけではありません。一般的ないくつかのエレメントについては 168 ページの『セキュリティーのその他の側面』で説明していますが、個々の IBM Security Directory Integrator コンポーネント内のセキュリティー構成の個々のエレメントについては、IBM Security Directory Integrator の IBM Knowledge Centerの『*Reference*』セクションを参照してください。

鍵、証明書、および鍵ストアの管理

以下の情報を参照することで、さまざまなタイプの鍵、鍵ストアでのそれらのリスト、および鍵の作成について学習できます。

バックグラウンド

SSL で使用する鍵、暗号化、およびセキュリティーの概念について詳しくは、ここで提供されるリンクを参照してください。

本製品内での暗号鍵の主要な用途は、SSL (112 ページの『Secure Sockets Layer (SSL) サポート』を参照) と暗号化 (149 ページの『IBM Security Directory Integrator サーバー・インスタンス・セキュリティー』を参照) です。

セキュリティーの概念および IBM JVM で使用される方法については、http://www.ibm.com/developerworks/java/jdk/security/ を参照してください。

公開鍵/秘密鍵と証明書

以下の情報を参照することで、公開鍵/秘密鍵とこれらの単独および相互の動作方法について詳しく知ることができます。

SSL と、RSA (サーバーのデフォルトの暗号化アルゴリズム) のような非対称暗号化アルゴリズムでは、公開鍵/秘密鍵が使用されます。公開鍵と秘密鍵は、1 対 1 に対応しています。対応している公開鍵と秘密鍵は、「鍵ペア」と呼ばれます。

通常、鍵ストア内には、公開鍵が X.509 証明書でラップされて格納されています。大部分の鍵ストア操作には、公開鍵だけでなく、公開鍵証明書全体の操作が含まれます。

更に、多くの場合、鍵ストア内では、秘密鍵が、対応する公開鍵の証明書に付属しています。

共通鍵

以下の知識を得ることで、共通鍵が使用されるすべてのアルゴリズムと鍵ストアについて理解できます。

共通鍵は、DES、AES、および RC4 のような対称型暗号化アルゴリズムで使用されます。JKS や PKCS#12 など、一部の鍵ストア形式は共通鍵をサポートしていないことに注意してください。

共通鍵は SSL 用には使用できません (SSL プロトコルでは、実際には共通鍵がその場で生成されますが、通常ユーザーはそれを制御できないからです)。

鍵ストア

以下の情報を参照して、鍵ストア、それらのファイル形式、原型およびさまざまな鍵ストア間の比較について詳しく学習します。

鍵ストアは、名前が意味するように、鍵のストレージです。これはファイルであったり、ハードウェア・デバイスであったりします。Java プログラムで使用される最も有名な鍵ストア・ファイル形式は、JKS、JCEKS、および PKCS#12 です。比較については、次の表を参照してください。

表 13. 鍵ストアのファイル形式

鍵ストアのファイル形式	原型	公開鍵/秘密鍵と証明書を格納	共通鍵を格納
JKS	プロプラエタリー	はい	いいえ
JCEKS	プロプラエタリー	はい	はい

表 13. 鍵ストアのファイル形式 (続き)

鍵ストアのファイル形式	原型	公開鍵/秘密鍵と証明書を格納	共通鍵を格納
PKCS#12	標準	はい	いいえ

上記の鍵ストア形式の中で唯一共通鍵を格納できるのは、JCEKS であることに注意してください。また、一般的に JCEKS は、JKS よりもはるかに強力な保護機能を備えています。JKS、JCEKS、および PKCS#12 鍵ストアは、パスワードで保護されます。更に、鍵ストア内の個々の秘密鍵と共通鍵は、個々のパスワードで保護されます。公開鍵の証明書はパスワードを持ちません。秘密にしておく理由がないからです。

SSL 用の鍵

SSL を操作するには公開鍵/秘密鍵のセットが必要です。以下の情報を参照して、この鍵セットを設定する手順を理解します。

このタスクについて

IBM JVM で SSL を使用する方法についての詳細は、http://www.ibm.com/developerworks/java/jdk/security/60/secguides/jsse2Docs/JSSE2RefGuide.html を参照してください。

SSL を使用するには、公開鍵/秘密鍵のセットを用意する必要があります。SSL では共通鍵は使用できません。

SSL 接続には 2 つのサイド (SSL サーバー・サイドと SSL クライアント・サイド) があります。各サイドには 2 つの鍵ストア (SSL 鍵ストアと SSL トラストストア) があります。「鍵ストア」という単語は、鍵の貯蔵庫と SSL 鍵ストアの両方を意味するために使用されることに注意してください。したがって、SSL 鍵ストアと SSL トラストストアは両方とも鍵ストアです。実際、SSL 鍵ストアと SSL トラストストアは単に論理的な役割のことであり、両方で同じ物理的な鍵ストア・ファイルを使用することは何ら問題はありません。SSL 鍵ストアは、この SSL サイドの認証性を SSL 接続の他方のサイドに証明するために使用される秘密鍵を含んでいます。SSL トラストストアは信頼されたパーティーの公開鍵証明書を含んでいます。

手順

1. SSL サーバー用の鍵を設定するには、秘密鍵とそれに対応する自己署名した公開鍵証明書を生成し、それを自らの SSL 鍵ストアに格納します (『公開鍵/秘密鍵ペアと自己署名証明書の生成』を参照してください)。この手順が必要になるのは、SSL 接続の自分のサイドでピアに対して認証性を証明する必要がある場合のみです。すなわち、自分が SSL サーバーの場合、または SSL クライアントであって、クライアント認証が必要な場合です。
2. [オプション] 認証局から証明書を取得して、それを自己署名証明書と交換することができます(『公開鍵証明書を鍵ストアにインポートする』を参照)。
3. [オプション] 秘密鍵の公開鍵証明書をエクスポートして、対話の相手となる SSL パーティーにそれを配布します (『鍵ストアから公開鍵証明書をエクスポートする』を参照)。認証局の証明書を使用する場合は、相手方に対して認証局の証明書を渡すだけで十分です。

4. 信頼できるパーティーの証明書を自分の SSL トラストストアにインポートします (『公開鍵証明書を鍵ストアにインポートする』を参照)。この手順は SSL クライアントでは必須です。SSL サーバーの場合は、クライアント認証が必要な場合にのみ必要です。

タスクの結果

注: デフォルトのプロパティーを使用して SSL (javax.net.ssl.*) を構成する場合、SSL 鍵ストアに秘密鍵は 1 つしか入れてはいけません。複数入れた場合に、いずれの鍵を使用するか指定する方法がないからです。

暗号化用の鍵

以下の情報を参照することで、暗号化、暗号化用の鍵および対応するアルゴリズムについて学習できます。

暗号化には 2 つの選択肢があります。

- 公開鍵/秘密鍵ペアを使用する
- 共通鍵を使用する

公開鍵による暗号化で最も一般的なアルゴリズムは RSA です。DiffieHellman (鍵交換) や DSA (デジタル署名) などその他の一般的な公開鍵アルゴリズムは、暗号化には使用できないことに注意してください。

一般に、共通鍵を使用する暗号化は、公開鍵を使用する暗号化に比べて、はるかに高速ではるかにセキュアです。しかし、Directory Integrator Server でのデフォルトは、以前のバージョンとの互換性を保つため、RSA を使用する公開鍵暗号化になっています。

ツール

鍵および証明書の操作には keytool と Ikeyman ユーティリティーを使用できます。これらのツールの詳細については、以下の情報を参照してください。

IBM JVM には、鍵と証明書を処理するために 2 つのユーティリティー (`keytool` および `Ikeyman`) が用意されています。`keytool` は、Java コミュニティーでよく使用される人気のあるコマンド行ユーティリティーです。`Ikeyman` は IBM が開発した GUI ツールであり、「keytool」の機能の多くを備えています。ツールは両方とも、*`TDI_install_dir`*`/jvm/jre/bin` フォルダーに格納されています。これらのツールについての詳細は、IBM JVM のマニュアルを参照してください (http://www.ibm.com/developerworks/java/jdk/security/)。

製品に付属しているデフォルトの鍵ストアは、次のとおりです。

表 14. IBM Security Directory Integrator 鍵ストア

鍵ストアの場所	鍵ストアのパスワード	信頼できる公開鍵	秘密鍵
TDI_install_dir/testserver.jks	server	admin	server
TDI_install_dir/serverapi/testadmin.jks	administrator	server	admin

鍵ストアの内容をリストする

`keytool` の `list` コマンドを使用して、鍵ストアの内容をリストすることができます。

例えば、次のコマンドを使用すると、鍵ストア・ファイル mystore.jck 内の鍵に関する情報 (別名とタイプ) がリストされます。鍵ストアの形式は JCEKS で、そのパスワードは「mystorepass」です。

```
keytool -list -storetype jceks -keystore mystore.jck -storepass mystorepass
```

鍵の作成

以下の情報を参照することで、鍵の作成、鍵ストアでの鍵の管理、鍵の使用について学習できます。

公開鍵/秘密鍵ペアおよび自己署名証明書を生成する

例えば、次の keytool コマンドを使用すると、別名「myserverkey」と X.509 自己署名公開鍵証明書を持つ RSA 公開鍵/秘密鍵ペアが生成されます。

```
keytool -genkeypair –alias myserverkey -dname cn=myserver.mydomain.com
-validity 365 -keyalg RSA -keysize 1024
 -keypass mykeypass -storetype jceks -keystore mystore.jck -storepass
mystorepass
```

証明書の所有者の識別名は「cn=myserver.mydomain.com」ですが、これは SSL の自己署名証明書を使用するサーバーの DNS 名と同じである必要があります (公開鍵暗号化の場合、証明書の内容は重要な意味を持ちません)。証明書は 365 日間有効です。生成される RSA 鍵のサイズは、1024 バイトです。秘密鍵のパスワードは「mykeypass」です。鍵ペアは、JCEKS 形式で鍵ストア・ファイル mystore.jck に保管されます (ファイルは、存在しなかった場合は、作成されます)。鍵ストアのパスワードは「mystorepass」です。

鍵ストア mystore.jck は、「myserver.mydomain.com」ホスト上で動作するサーバー・プログラムの SSL 鍵ストアとして使用できます。鍵ストアには秘密鍵の公開鍵証明書も含まれているため、「myserver.mydomain.com」上のサーバーに接続しているクライアントの SSL トラストストアとして使用できます (ただし、クライアントに秘密鍵を渡すことは完全に不要であり、一般的にはセキュリティー上好ましくありません)。

認証局から認証を取得する

通常、CA 署名証明書を獲得し、使用するプロセスは以下のとおりです。

最初に、鍵ペアと自己署名証明書を生成します (『公開鍵/秘密鍵ペアおよび自己署名証明書を生成する』セクションを参照)。その後、認証局から公開鍵の証明書が要求されます。認証局が署名された証明書を返送してきたら、証明書を適切なトラストストアにインポートして、自己署名証明書と交換します。

例えば、次のように keytool を使用すると、鍵ストア mystore.jck の「myserverkey」鍵に対する証明書署名要求を生成できます。

```
keytool -certreq -file myreq.csr -alias myserverkey -keypass mykeypass
-storetype jceks
 -keystore mystore.jck -storepass mystorepass
```

このコマンドを使用すると、別名「myserverkey」を持つ公開鍵に対する証明書署名要求が myRequest.csr ファイルに作成されます。これで、作成された証明書署名要求を認証局に送信できます。新しい証明書が到着すると、『公開鍵証明書を鍵ストアにインポートする』セクションの説明に従って、

それを鍵ストアにインポートできます。以下の keytool コマンドを使用すると、別名「myseckey」を持つ 256 ビットの AES 鍵が生成されます。

```
keytool -genseckey -keyalg AES -alias myseckey -keysize 256 -keypass mykeypass
-storetype jceks
 -keystore mystore.jck -storepass mystorepass
```

新しい鍵は、パスワード「mystorepass」とともに、JCEKS 鍵ストア・ファイル mystore.jck に格納されます。共通鍵を保護するパスワードは「mykeypass」です。

1 つの鍵ストアから別の鍵ストアに鍵をコピーする

例えば、『公開鍵/秘密鍵ペアおよび自己署名証明書を生成する』で作成した鍵ペアは、次の keytool コマンドを使用してコピーすることができます。

```
keytool -importkeystore  -srckeystore mystore.jck -destkeystore myotherstore.jks
-srcstoretype jceks
 -deststoretype jks -srcstorepass mystorepass -deststorepass myotherstorepass
-srcalias myserverkey
 -destalias myotherserverkey -srckeypass mykeypass -destkeypass myotherkeypass
```

コピーは別名「myotherserverkey」で JKS 鍵ストア・ファイル myotherstore.jks に格納されます (ファイルは、存在しない場合、作成されます)。

鍵ストアの形式を別の形式に変換する

例えば、『公開鍵/秘密鍵ペアおよび自己署名証明書を生成する』セクションで作成した JCEKS 鍵ストアは、以下の keytool コマンドを使用すると、JKS 鍵ストア myotherstore.jks に変換することができます。

```
keytool -importkeystore  -srckeystore mystore.jck -destkeystore
myotherstore.jks -srcstoretype jcek
 -deststoretype jks -srcstorepass mystorepass -deststorepass
myotherstorepass
```

このコマンドは最後に、ソースの鍵ストア内の個々の秘密鍵または共通鍵のパスワードを入力するよう指示します。JKS 鍵ストアおよび PKCS#12 鍵ストアは、共通鍵を保持できないことに注意してください。したがって、共通鍵を含む鍵ストアは、JKS または PKCS#12 のいずれにも変換を試みないでください。

鍵ストアから公開鍵証明書をエクスポートする

次のコマンドを使用すると、『公開鍵/秘密鍵ペアおよび自己署名証明書を生成する』で作成した公開鍵証明書を、バイナリー・ファイル myserverkey.der にエクスポートすることができます。

```
keytool -exportcert -alias myserverkey -file myserverkey.der
-storetype JCEKS -keystore mystore.jck
 -storepass mystorepass
```

この結果生成される .der ファイルには、DER エンコード方式の X.509 証明書が含まれます。これはバイナリー・ファイルです。同じバイナリー・データをテキスト形式 (DER エンコード方式の X.509 証明書の base-64 エンコード形式) で取得するには、keytool の「-rfc」オプションを使用します。

```
keytool -exportcert -alias myserverkey -file myserverkey.arm
-storetype JCEKS -keystore mystore.jck
 -storepass mystorepass -rfc
```

公開鍵証明書を鍵ストアにインポートする

新しい信頼できる証明書を鍵ストアにインポートするには、次のようなコマンドを使用します。

```
keytool -importcert -alias myserverkey -file myserverkey.der
-storetype JCEKS -keystore mystore.jck
 -storepass mystorepass
```

keytool は、インポートを試みている証明書の署名者の検証を試みます。これは、インポート対象の証明書から、特定の信頼できる証明書までの証明書チェーンを構成することを意味します。チェーンを確立できない場合、keytool は証明書をインポートする必要が本当にあるのか尋ねてきます。

証明書署名要求に対する認証局からの応答である証明書をインポートする (これは、鍵ストアにその証明書に対する秘密鍵を既に保管していることを意味します) には、次のようなコマンドを使用します。

```
keytool -importcert -alias myserverkey –keypass mykeypass -file
myserverkey.der -storetype JCEKS -keystore mystore.jck
 -storepass mystorepass
```

既存の秘密鍵用の証明書をインポートする場合、秘密鍵のパスワードを入力する必要があることに注意してください。keytool は、信頼できる証明書に至る証明書チェーンを構成することで、証明書の署名者の検証を試みます。チェーンを確立できない場合、インポートは失敗です。証明書の認証性を検証することを要求されることはありません。証明書署名要求への応答のインポートを成功させるには、証明書を発行した認証局を信頼する必要があります。認証局が著名な認証局 (VeriSign や Thawte など) の 1 つであれば、JVM のデフォルトのトラストストア (java.home/lib/security/cacerts) 内の証明書は、keytool の「-trustcacerts」オプションを使用することによって、信頼することができます。

```
keytool -importcert -alias myserverkey –keypass mykeypass -file
myserverkey.der -storetype JCEKS -keystore mystore.jck
 -storepass mystorepass –trustcacerts
```

keytool を使用して証明書の有効期限を延長する

期限切れ (または期限切れ目前) で別名が「myserverkey」の自己署名証明書を含んでいる、mystore.jck と呼ばれる JCEKS 鍵ストアがあるとします。鍵ストア内には、関連付けられた秘密鍵があります。鍵ストアのパスワードは「mystorepass」、秘密鍵のパスワードは「mykeypass」であるとします。このとき、この証明書の有効期限を更に 365 日間延長するには、keytool を使用して、次のようなコマンドを実行します。

```
keytool -selfcert -v -alias myserverkey –keypass mykeypass -validity 365
–storetype jceks -keystore mystore.jck
 -storepass mystorepass
```

この操作によって、新しい自己署名証明書が生成されます。これは元の証明書と同じ DN、SIGALG、KEYS を持ちますが、SERIAL NUMBER、および VALIDITY の期間は新しくなっています。

注: 生成された新しい証明書は、自動的に元の証明書に置き換わります。

したがって、後で参照などの理由で元の証明書が必要になる場合は、上記の説明に基づいて証明書の期限を延長する前に、元の鍵ストアのコピーを保管しておく必要があります。
上記の手順が有効なのは、自己署名証明書のみであることに注意してくださ

い。この手順では、実際に公開鍵の新しい自己署名証明書が生成されます。そのため、生成された証明書をエクスポートし、通信先となる SSL パーティーのトラストストアを更新する必要があります。

PFX/PKCS#12 ファイルに保管された鍵の操作

Java に関する限り、PKCS#12 も JCEKS や JKS と同様に、単に一種の鍵ストアに過ぎません。PKCS#12 鍵ストアを操作するには、keytool の「-storetype」オプションに「pkcs12」を設定するだけです。例えば、以下のコマンドを使用すると、パスワード「mystorepass」を持つ、mystore.p12 PKCS#12 ファイルの内容をリストすることができます。

```
keytool –list –storetype pkcs12 –keystore mystore.p12 –storepass mystorepass
```

鍵ストア・ファイルの作成

鍵ストア・ファイルは、使用する前に作成する必要はありません。keytool は、存在していない鍵ストア・ファイルに書き込む必要が生じたとき、新しい鍵ストア・ファイルを自動的に作成します。例えば、まだ存在していない鍵ストアに新しい鍵を生成する場合や証明書をインポートする場合、keytool は、最初に鍵ストア・ファイルを作成します。

keytool を FIPS モードで実行する

keytool を FIPS 準拠のモードで実行する場合は、次のように、各コマンドで「-providerClass」オプションを使用します。

```
keytool –list –storetype JCEKS –keystore mystore.jck –storepass mystorepass
 –providerClass com.ibm.crypto.fips.provider.IBMJCEFIPS
```

Secure Sockets Layer (SSL) サポート

ネットワーク・トラフィックの暗号化および認証は、SSL セキュリティー機能を通して行うことができます。また以下の情報を参照することで、サポートされているコネクターのリストを表示し、構成について理解することができます。

SSL は、IBM Security Directory Integrator の多くのセキュリティー機能にとって重要な基盤です。IBM Security Directory Integrator の機能を完全に活用するには、SSL について実用的な知識が必要になります。

適切に構成された IBM Security Directory Integrator サーバーで SSL をサポートするコネクターを以下に示します。

- コネクター
  - AD 変更検出コネクター
  - Axis Easy Web Service サーバー・コネクター
  - Axis2 Web サービス・サーバー・コネクター
  - IBM Domino 変更検出コネクター
  - IBM Domino ユーザー・コネクター
  - DSML v2 SOAP サーバー・コネクター
  - FTP クライアント・コネクター
  - HTTP サーバー・コネクター
  - IDS 変更ログ・コネクター
  - IBM MQ Series コネクター

- JMS コネクター
- JMS パスワード・ストア・コネクター
- JNDI コネクター
- LDAP コネクター
- LDAP グループ・コネクター
- LDAP サーバー・コネクター
- Lotus® Notes コネクター
- メールボックス・コネクター
- Sun Directory 変更検出コネクター
- LDAP コネクター
- TADDM 変更検出コネクター
- TADDM コネクター
- TCP コネクター
- TCP サーバー・コネクター
- TPAE IF 変更検出コネクター
- Web サービス受信側サーバー・コネクター
- zOS LDAP 変更ログ・コネクター

SSL は、リモート通信を行う 2 局間のネットワーク・トラフィックの暗号化および認証を提供します。IBM Security Directory Integrator の実動デプロイメントのほとんどで SSL が使用されます。このため、SSL サポートは、IBM Security Directory Integrator の主要なセキュリティー機能の 1 つになります。開発の観点からの Java プログラムにおける SSL の使用に関する情報や、 SSL に関する詳細については、http://download.oracle.com/javase/6/docs/technotes/guides/security/jsse/JSSERefGuide.html を参照してください。

IBM Security Directory Integrator は、クライアントとして、サーバーとして、および同時に両方として使用できます。クライアントとして使用する場合の SSL に対する IBM Security Directory Integrator の構成と、サーバーとして使用する場合の IBM Security Directory Integrator の構成は異なります。このため、このセクションは、『IBM Security Directory Integrator コンポーネントのサーバー SSL 構成』および 115 ページの『IBM Security Directory Integrator コンポーネントのクライアント SSL の構成』の 2 つのサブセクションに分割されています。

IBM Security Directory Integrator コンポーネントのサーバー SSL 構成

IBM Security Directory Integrator をサーバーとして使用する場合に SSL サポートを有効にするには、鍵ストアを定義する必要があります。ここで説明する手順に従えば、このタスクを実行できます。

このタスクについて

IBM Security Directory Server コンポーネントがサーバー (例えば、サーバー・モード・コネクター) として使用される場合、 SSL に関しては IBM Security Directory Integrator によって使用される鍵ストアを定義することが必要になります。鍵ストア

およびトラストストアの詳細については、http://download.oracle.com/javase/6/docs/technotes/guides/security/jsse/JSSERefGuide.htmlにある資料を参照してください。IBM Security Directory Integrator をサーバーとして使用する場合に SSL サポートを有効にするには、以下の手順を実行する必要があります。

注: IBM Security Directory Integrator サーバーでは、RMI はデフォルトで使用可能になっています。サーバー認証のプロパティーには、デフォルトの鍵ストア・プロパティーの値が含まれます。

1. Java (jks) 鍵ストア・ファイルがまだ IBM Security Directory Integrator にない場合は、*keytool* を使用して鍵ストア・ファイルを作成する (このツールは、プラットフォームに応じて *TDI_install_dir*/jvm/jre/bin または *TDI_install_dir*/jvm/bin にあります)。IBM Security Directory Integrator で使用する個人鍵がない場合は、認証局から入手するか、または自己署名鍵を作成する。
2. IBM Security Directory Integrator の証明書が自己署名証明書の場合は、この証明書をエクスポートする。
3. IBM Security Directory Integrator の証明書が自己署名証明書の場合は、keytool を使用して、エクスポートした IBM Security Directory Integrator 証明書をルート権限証明書としてクライアントの鍵ストア・ファイルにインポートする。
4. *TDI_install_dir*/etc/global.properties ファイルを編集し、鍵ストア・ファイルの場所、鍵ストア・ファイルのパスワード、鍵ストア・ファイルのタイプを指定する。
```
## client authentication
javax.net.ssl.keyStore=serverapi¥testadmin.jks {protect}-
javax.net.ssl.keyStorePassword=administrator
javax.net.ssl.keyStoreType=jks
```
5. クライアントの SSL を有効にする (Web ブラウザーでの https の使用など)。
6. IBM Security Directory Integrator を再始動します。

注:

1. IBM Security Directory Integrator サーバーは、鍵ストア/トラストストアを管理しません。鍵ストアのサポートに関して、 IBM Security Directory Integrator サーバーによって IBM Security Directory Integrator コンポーネントに提供されているものは、標準 Java 鍵ストア/トラストストアのプロパティーを指定できる global.properties または solution.properties ファイルのみです。
2. IBM Security Directory Integrator コンポーネントは、global.properties または solution.properties 内のデフォルト構成の鍵ストア/トラストストアを使用するか、デフォルトと異なる鍵ストア/トラストストアを使用できるように独自の SSL ソケットの処理をインプリメントする (例えば、カスタム SSLServerSocket Java クラスをインプリメントする) ことを選択できます。
3. IBM Security Directory Integrator がクライアントおよびサーバーの両方の証明書を使用する必要がある場合は、global.properties または solution.properties で構成されるデフォルトの証明書のみが使用され、これは同じ証明書である必要があります。代替手段として、 SSLSocket または SSLServerSocket Java クラスのカスタム・インプリメンテーションを記述し、デフォルトとは異なる証明書が使用されるようにします。

4. IBM Security Directory Integrator Web サービス・コンポーネントの証明書の特性については、セクション 169 ページの『IBM Security Directory Integrator Web サービス・スイートの証明書』を参照してください。

IBM Security Directory Integrator コンポーネントのクライアント SSL の構成

IBM Security Directory Integrator をクライアントとして使用する場合に SSL サポートを有効にするには、トラストストアを定義する必要があります。ここで説明する手順に従えば、このタスクを実行できます。

このタスクについて

IBM Security Directory Integrator コンポーネントがクライアント (例えば、LDAP コネクター) として使用される場合、 SSL に関しては、IBM Security Directory Integrator によって使用されるトラストストアを定義することが必要になります。鍵ストアおよびトラストストアに関する詳細については、http://download.oracle.com/javase/6/docs/technotes/guides/security/jsse/JSSERefGuide.htmlにある資料を参照してください。

クライアントとして IBM Security Directory Integrator を使用する場合に SSL サポートを有効にする手順を以下に示します。

1. サーバー (IBM Security Directory Integrator など) を構成して SSL を有効にする。
2. サーバーの証明書が自己署名証明書の場合は、この証明書をエクスポートする。
3. Java (jks) 鍵ストア・ファイルがまだない場合は、*keytool* を使用して IBM Security Directory Integrator の鍵ストア・ファイルを作成する (このツールは、プラットフォームに応じて *root_directory*/jvm/jre/bin または *root_directory*/jvm/bin にあります)。
4. サーバーの証明書が自己署名証明書の場合は、keytool を使用して、サーバーの証明書をルート権限証明書として IBM Security Directory Integrator 鍵ストア・ファイルにインポートする。
5. `root_directory/etc/global.properties` ファイルを編集し、鍵ストア・ファイルの場所、鍵ストア・ファイルのパスワード、鍵ストア・ファイルのタイプを指定します。

   注: 以下の 4 行 (# で始まるコメント) は、IBM Security Directory Integrator サーバーに対するクライアント認証とサーバー認証では不要です。IBM Security Directory Integrator に属するストアは、デフォルトで使用するように設定されています。これは、リモート・メソッド呼び出し (RMI) をデフォルトで使用可能にした措置の一環です。

   ```
   # Keystore file information for the server TDI authentication.
   # It is used to provide the public key of the TDI to the SSL enabled client.
   # javax.net.ssl.keyStore=D:¥test¥clientStore.jks
   # javax.net.ssl.keyStorePassword=secret
   # javax.net.ssl.keyStoreType=jks
   ```
6. コネクターで SSL を使用可能にします。
7. IBM Security Directory Integrator を再始動します。

注: IBM Security Directory Integrator トラストストアおよび鍵ストアは、 IBM Domino 変更検出コネクターの SSL 構成にはまったく関係しません。詳しくは、169 ページの『IBM Domino SSL の特性』 を参照してください。

SSL クライアント認証

IBM Security Directory Integrator コンポーネントをクライアントおよびサーバーとして使用する計画である場合は、SSL で鍵ストアとトラストストアの両方を使用する必要があります。詳細については、ここで提供する情報を参照してください。

IBM Security Directory Integrator コンポーネントがクライアントとして使用され、その通信相手のサーバーが SSL クライアント認証を必要とする場合、トラストストアのほかに、IBM Security Directory Integrator には鍵ストアも必要になります。この場合、鍵ストアは、IBM Security Directory Integrator がサーバーとして使用される場合と同様に定義できます。セクション 113 ページの『IBM Security Directory Integrator コンポーネントのサーバー SSL 構成』を参照してください。

注: SSL クライアント認証をサポートするクライアント IBM Security Directory Integrator コンポーネントには、IBM Security Directory Integrator サーバー・コンポーネントとは異なり、通常「SSL クライアント認証」チェック・ボックスは不要です。そのようなクライアント IBM Security Directory Integrator コンポーネントには必ず、サーバーに対してその ID を証明するために、生成され、`global.properties` または `solution.properties` で構成された鍵ストアが必要です。サーバーが SSL クライアント証明書を必要とする場合は、クライアント SSL ライブラリーによって、`global.properties` または `solution.properties` で構成された鍵ストアからクライアントの証明書が自動的に送信されます。

IBM Security Directory Integrator と Microsoft Active Directory SSL の構成

以下の手順に従って、IBM Security Directory Integrator と Microsoft Active Directory に対して SSL を構成することができます。

このタスクについて

1. Windows Server に証明書サービスをインストールし、Active Directory ドメインにエンタープライズ証明機関をインストールします。詳しくは、http://windowsitpro.com/article/articleid/14923/how-do-i-install-an-enterprise-certificate-authority.html を参照してください。必ず **エンタープライズ証明機関**をインストールしてください。
2. Certificate Server サービスを開始します。これにより、Internet Information Service (IIS) 内に仮想ディレクトリーが作成され、証明書を配布できるようになります。
3. ドメイン・コントローラーに対して認証局 (CA) から SSL 証明書を取得するよう指示する Domain Controller セキュリティー (グループ) ポリシーを作成します。
   a. 「**Active Directory ユーザーとコンピュータ**」管理ツールを開きます。
   b. ドメインの下で、「**ドメイン・コントローラー**」を右クリックします。
   c. 「**プロパティー**」を選択します。

d. 「**グループ・ポリシー**」タブを選択し、「**デフォルトのドメイン・コントローラー・ ポリシー**」を編集のためクリックします。

e. 「**コンピューターの構成 -> Windows の設定 -> セキュリティーの設定 -> 公開鍵のポリシー**」の順に進みます。

f. 「**自動証明書要求の設定**」を右クリックします。

g. 「**新規作成**」をクリックします。

h. 「**自動証明書要求**」を選択します。

i. ウィザードを実行します。「**ドメイン コントローラの証明書テンプレート (Certificate Template for a Domain Controller)**」を選択します。

j. CA として**エンタープライズ証明機関**を選択します。このとき、サード・パーティーの CA も選択できます。

k. ウィザードを完了します。

注: すべてのドメイン・コントローラーがこの CA の証明書を自動的に要求し、ポート 636 で SSL を使用して LDAP をサポートします。

4. IBM Security Directory Integrator がインストールされているコンピューターに認証局証明書を取り込みます。

   注: 証明書サーバーをインストールする前に、IIS をインストールする必要があります。

   a. IBM Security Directory Integrator がインストールされているコンピューターで Web ブラウザーを開きます。

   b. http://*server_name*/certsrv/ (*server_name* は Windows 2000 Server の名前) に移動します。ログインするよう求められます。

   c. 「**CA 証明書または証明書失効リストの取得**」タスクを選択します。

   d. 「**次へ**」をクリックします。

      次に表示されるページでは、CA 証明書が自動的に強調表示されます。

   e. 「**CA 証明書のダウンロード**」をクリックします。

      新規ダウンロード・ウィンドウが開きます。

   f. ファイルをハード・ディスクに保管します。

5. keytool を使用して証明書ストアを作成します。keytool.exe を使用して証明書ストアを作成し、この証明書ストアに CA 証明書をインポートします。

   注: keytool.exe は、ご使用のプラットフォームに応じて、*root_directory*/jvm/jre/bin または *root_directory*/jvm/bin のいずれかにあります。
   次のコマンドを使用します。

   ```
   jvm¥jre¥bin¥keytool -import -file
    certnew.cer -keystore keystore_name.jks
    -storepass password-alias keyalias_name
   ```

   例えば、以下の値を想定します。

   ```
   Keystorename = idi.jks
   Password = secret
   Keyalias name = AD_CA
   ```

   コマンドは次のスクリプトに似ています。

```
C:¥Program Files¥IBM¥TDI¥V7.2¥jvm¥jre¥bin¥keytool -import
-file certnew.cer -keystore idi.jks  -storepass secret -alias AD_CA
```

鍵ストアの内容を確認するため、以下のスクリプトを入力します。

```
C:¥Program Files¥IBM¥TDI¥V7.2¥jvm¥jre¥bin¥keytool
 -list -keystore idi.jks -storepass secret
```

この結果、以下の行が出力されます。

```
Keystore type: jks
Keystore provider: SUN

Your keystore contains 1 entry:

ad_ca, Mon Nov 04 22:11:46 MST 2002, trustedCertEntry,
Certificate fingerprint (MD5): A0:2D:0E:4A:68:34:7F:A0:21:36:78:65:A7:1B:25:55
```

6. 前のステップで作成した鍵ストアを使用するように IBM Security Directory Integrator を構成します。root_directory/global.properties ファイルを編集し、鍵ストア・ファイルの場所、鍵ストア・ファイルのパスワード、鍵ストア・ファイルのタイプを指定します。現行リリースでは、jks-type のみがサポートされています。

```
#server authentication
#example
javax.net.ssl.trustStore=c:¥test¥idi.jks
javax.net.ssl.trustStorePassword=secret
javax.net.ssl.trustStoreType=jks
#client authentication
#example
javax.net.ssl.keyStore=c:¥test¥idi.jks
javax.net.ssl.keyStorePassword=secret
javax.net.ssl.keyStoreType=jks
```

7. LDAP コネクターの SSL を使用可能にします。
   a. LDAP コネクターの構成ウィンドウを表示します。
   b. 「**LDAP URL**」をポート 636 に変更します。
   c. 「**SSL の使用**」にチェック・マークを付けます。
8. IBM Security Directory Integrator を再始動します。

注: IBM Security Directory Integrator Windows サービス・ラッパーにより、IBM Security Directory Integrator を複数サービス・インスタンスとして開始できます。

SSL および PKCS#11 サポートを使用可能にするためのプロパティーのサマリー

以下のリンクおよび情報を参照すると、SSL プロパティーの構成の詳細について理解できます。

サーバー認証、クライアント認証、および PKCS#11 サポートのために SSL プロパティーを構成することができます。 Public Key Cryptography Standards (PKCS) についての詳細は、206 ページの『ハードウェア・デバイスの暗号鍵を使用する』を参照してください。

表 15. SSL サーバー認証

プロパティー	デフォルト値	説明
javax.net.ssl.trustStore	serverapi¥testadmin.jks	トラストストア・ファイルの場所
{protect}-javax.net.ssl.trustStorePassword	管理者(デフォルトで暗号化)	トラストストアのパスワード。

表 15. SSL サーバー認証 (続き)

プロパティー	デフォルト値	説明
javax.net.ssl.trustStoreType	jks	トラストストアのタイプ。

表 16. SSL クライアント認証

プロパティー	デフォルト値	説明
javax.net.ssl.keyStore	serverapi¥testadmin.jks	鍵ストア・ファイルの場所。
{protect}- javax.net.ssl.keyStorePassword	管理者(デフォルトで暗号化)	鍵ストアのパスワード。
javax.net.ssl.keyStoreType	jks	鍵ストアのタイプ。

表 17. PKCS#11 サポート

プロパティー	デフォルト値	説明
com.ibm.di.pkcs11cfg	etc¥pkcs11.cfg	IBM PKCS11 インプリメンテーション・プロバイダーの初期化に必要な構成ファイルのパスを指定するには、これを使用します。IBM Security Directory Integrator 7.0 で追加されました。
com.ibm.di.server.pkcs11	false	SSL で PKCS11 準拠の暗号デバイスを使用します。 IBM Security Directory Integrator 7.0 で追加されました。
com.ibm.di.server.pkcs11.library	なし	PKCS11 クライアント・ライブラリーへのパスを指定します。 IBM Security Directory Integrator 7.0 で追加されました。
com.ibm.di.server.pkcs11.slot	なし	デバイスのスロット番号を指定します。
{protect}- com.ibm.di.server.pkcs11.pass	なし	このパスワードを使用して、PKCS11 準拠の暗号デバイスにアクセスします。デフォルトでは暗号化されています。IBM Security Directory Integrator 7.0 で追加されました。

SSL の例

IBM Security Directory Integrator をサーバーとして使用した場合およびクライアントとして使用した場合の SSL に関する構成方法を説明するために、以下の 2 つの例を示します。1 つ目の例では LDAP サーバー・コネクターをデプロイし、2 つ目では LDAP コネクターをデプロイします。

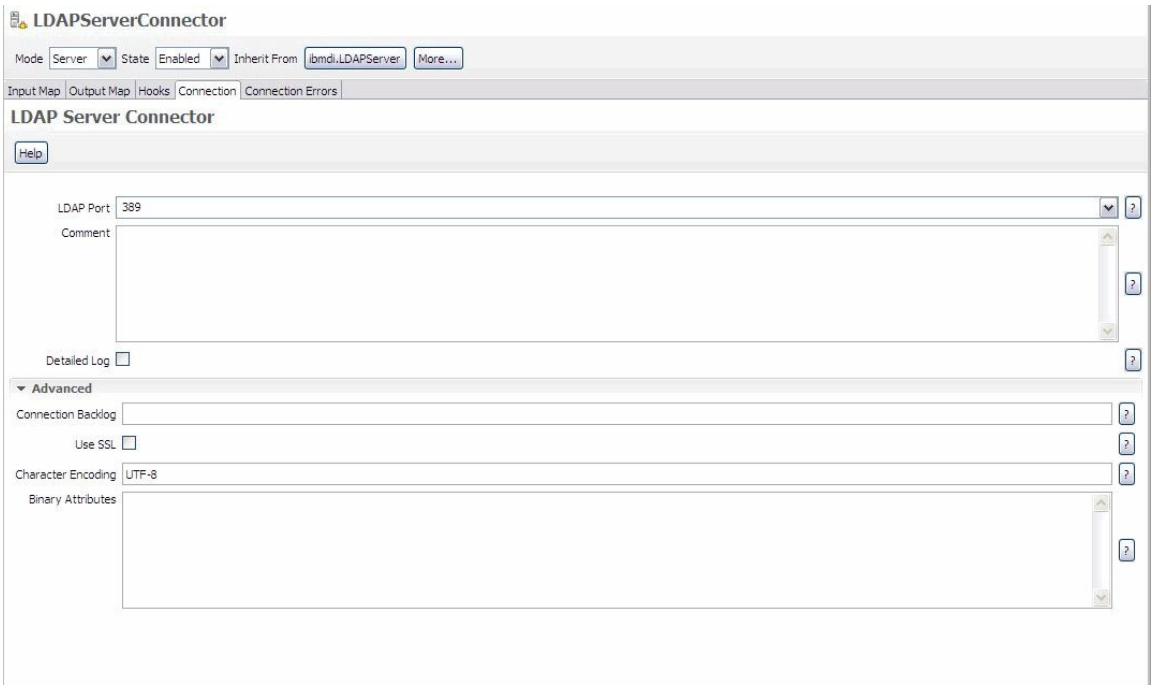
サーバーとしての IBM Security Directory Integrator コンポーネント

ここで示す手順に従って、IBM Security Directory Integrator コンポーネントを SSL 用にサーバーとして構成します。

このタスクについて

この例では、LDAP サーバー・コネクターを使用します。LDAP サーバー・コネクターは、 LDAP 要求を listen します。LDAP 要求を受信すると、コネクターは要求を構文解析し、ホスティングしている AssemblyLine に要求データを提供します。その後、 AssemblyLine は要求を処理し、LDAP サーバー・コネクターに対して応答のデータを提供します。これにより LDAP サーバー・コネクターは LDAP 応答を作成し、これを LDAP クライアントに返送することができます。以下のガイドラインでは、LDAP サーバー・コネクターが使用されている場合に IBM Security Directory Integrator を SSL 用に構成する方法について順を追って説明します。

1. 認証局 (CA) に要求するか、または『公開鍵/秘密鍵ペアおよび自己署名証明書を生成する』での説明に従って、自己署名証明書を作成することによって、サーバーの鍵ストアを取得します。
2. 113 ページの『IBM Security Directory Integrator コンポーネントのサーバー SSL 構成』セクションで説明されているように、`global.properties` または `solution.properties` で鍵ストアの詳細を設定します。
3. コネクター GUI の構成ウィンドウで、「**SSL の使用**」を選択します。パラメーターを表示するために、「詳細設定」セクションの展開が必要な場合もあります。



クライアントとしての IBM Security Directory Integrator コンポーネント

ここで示す手順に従って、IBM Security Directory Integrator コンポーネントを SSL 用にクライアントとして構成します。

このタスクについて

この例では、LDAP コネクターを使用します。LDAP コネクターは、LDAP サーバーに接続し、 LDAP 要求を送信します。サーバーが LDAP 応答を戻した後、LDAP コネクターはその応答を AssemblyLine に提供し、そこでさらに処理が行われます。以下のガイドラインでは、LDAP コネクターが使用されている場合に IBM Security Directory Integrator を SSL 用に構成する方法について順を追って説明します。

1. クライアントのトラストストアを生成します。
2. LDAP サーバーの証明書をクライアントのトラストストアにインポートします。
3. 115 ページの『IBM Security Directory Integrator コンポーネントのクライアント SSL の構成』セクションで説明されているように、`global.properties` または `solution.properties` でトラストストアの詳細を設定します。

以下のコマンド行により、既存の証明書が鍵ストアにインポートされます (鍵ストアが存在しない場合は作成されます)。

```
keytool -import -trustcacerts -file myLDAPServerCert.cer
     -keystore myClientTruststore.jks -storepass myclientTruststorePassword
     -alias myTrustedLDAPServerAlias
```

このコマンド行により、 `myLDAPServerCert.cer` 証明書が別名 `myTrustedLDAPServerAlias` で `myClientTruststore.jks` 鍵ストアにインポートされます。鍵ストアにアクセスするためのパスワードは `myclientTruststorePassword` です。

リモート・サーバー API

IBM Security Directory Integrator では、クライアント・アプリケーションをサーバーに接続させることができます。クライアント・タスクはサーバー・タスクをリモートで呼び出すことができます。詳しくは、以下のリンクと情報を参照してください。

このセクションでは、IBM Security Directory Integrator サーバーのインスタンスの保護については説明していません (これについては 149 ページの『IBM Security Directory Integrator サーバー・インスタンス・セキュリティー』で説明しています)。その代わりに、このセクションではクライアント・アプリケーションがサーバーに接続する方法について説明しています。

IBM Security Directory Integrator は、リモート API (「サーバー API」とも呼びます) の概念をサポートしています。RMI というアクセス・レイヤーにより、クライアント・タスクがリモート IBM Security Directory Integrator サーバー上のタスクを呼び出すことができます。

注: 例えばローカル・コンピューターでサーバー・インスタンスを開始し、リモート API を使用してループバック・アドレス 127.0.01 でこのサーバーにアクセスする場合などは、リモート・サーバーが、クライアント・アプリケーションと同じコンピューターで実行されている可能性があります。リモート・サーバーをローカルで実行する場合でも、以降で説明する概念はすべて適用されます。
サーバー API 呼び出しでは以下の操作を実行できます。

- サーバーの情報を取得する。

- サーバー上にインストールされているコンポーネントの情報を取得する。
- サーバーによってロードされる構成の読み取りと書き込みを行う。
- 新しい構成をサーバーにロードする。
- AssemblyLine の開始、照会、停止を行う。
- AssemblyLine を介したサイクルを実行する。

注: 動作中の各 IBM Security Directory Integrator サーバーに対するリモート・サーバー・アクセスのニーズが増えてきたため、ローカル・アクセスからリモート・アクセスにデフォルトが変更されました。v7.1.1 では、リモート・サーバー API がデフォルトで使用可能になっています。v7.1.1 より前では、デフォルトでサーバー API が使用可能なのは、ローカル・アクセスに限られていました。ローカル・アクセスとは、同じ Java ランタイム環境 (JRE) からのアクセスを意味します。セキュリティーを保証するため、リモート・アクセスには SSL クライアント認証が必要になります。クライアント認証を使用する SSL アクセスには、IBM Security Directory Integrator でデプロイされたサンプルの鍵ストアとトラストストアが用意されています。

サーバー API については、IBM Security Directory Integrator Java API 文書 (*TDI_install_dir*/docs/api。CE で「**ヘルプ**」-> 「**ようこそ**」->「**JavaDocs**」と選択すると、この文書を表示するブラウザーを起動できます) を参照してください。このコンテキストで該当するパッケージは `com.ibm.di.api` です。 IBM Security Directory Integrator の IBM Knowledge Centerの『リファレンス』セクションにある『サーバー API の使用』というセクションも参照してください。

構成エディターはリモート API を使用して、ソリューションのテスト・ランに使用するサーバーと対話します。この IBM Security Directory Integrator サーバーは、同じマシンで動作している場合、「ローカル開発サーバー」と呼ばれます。デプロイメント・プラットフォームが構成エディターをサポートしない場合のセットアップでは、デプロイメント・サーバー上で開発サーバーを動作させて、構成エディターは Windows のようなサポートされているプラットフォームで実行します (この実行方法を、「リモート構成エディター」と呼びます)。この設計では、リモートとローカルの両方の構成ファイルに対して、統一的なインターフェースが用意されています。リモート・デプロイメント・サーバーと対話する構成エディターの特定の機能については、163 ページの『リモート構成エディターの使用』を参照してください。

サーバー API は、一連のサーバー・プロパティーを使用して構成します (123 ページの『サーバー API の構成』を参照)。これらのプロパティーは、IBM Security Directory Integrator サーバーの `global.properties` 構成ファイルで指定します。一部のプロパティーは、別の構成ファイルや鍵ストア・ファイルを参照します。

サーバー API は、セキュリティーに関する多くの機能を提供します (これらの機能は、 IBM Security Directory Integrator ソリューション・ベースのクライアントと他のクライアント・アプリケーションの両方において処理する必要があります)。サーバー API のアクセス・セキュリティーには、以下の 3 つの側面があります。

1. 127 ページの『サーバー API の SSL リモート・アクセス』 (リモート IBM Security Directory Integrator サーバーへのトランスポート・チャネルを保護する)

2. 128 ページの『サーバー API 認証』 (IBM Security Directory Integrator サーバーに対するクライアント認証を処理する)
3. 140 ページの『サーバー API の許可』 (IBM Security Directory Integrator サーバーに対するクライアント許可 (すなわち、認証された後クライアントが何を実行できるか) を処理する)

サーバー API の構成

サーバー API を構成するのに役立つプロパティーのリストを参照できます。

関連するプロパティーは以下のとおりです。

プロパティー	デフォルト値	説明
api.on	**true**	**true** に設定した場合、始動時にサーバー API が初期化されて使用可能になります。それ以外の場合、サーバー API は初期化されず、使用できません。名前が「api.」で始まる他のすべてのプロパティーは、`api.on` が **true** に設定されている場合にのみ有効です。
api.audit.on	**false**	**true** を設定すると、監査機能がオンになります。**true** が設定されていると、監査通知が抑止されていたとしても、各監査ポイントに監査項目が作成されます。
api.user.registry	`serverapi/registry.txt`	構成されている場合は、サーバー・ユーザー・レジストリー・ファイル名を指しています。
api.user.registry.encryption.on	**false**	**true** が設定されていると、サーバー API はサーバー・ユーザー・レジストリー・ファイルを始動時に復号します。
api.remote.on	**true**	**true** が設定されていると、サーバー API のリモート RMI パートが初期化され、使用できる状態になります。それ以外の場合は、サーバー API のリモート RMI パートは初期化されず、使用することはできません。
api.remote.ssl.on	**true**	**true** を設定すると、サーバー API とその JMX レイヤーの RMI 接続では、クライアント認証とサーバー認証を使用する SSL が使用されます。SSL 接続では、サーバー API は、サーバー証明書と秘密鍵 (`api.keystore` および `api.key.alias` プロパティーで指定された証明書と秘密鍵) を使用します。RMI クライアントはその証明書を信頼する必要があります。**false** に設定した場合、クライアント接続に SSL は使用されず、認証および許可は実行されません。ローカル・ホストおよび `api.remote.nonssl.hosts` プロパティーにリストされたホストからの接続は受け入れられます。`api.remote.nonssl.hosts` が空の場合、ローカル・ホストからの接続のみが受け入れられます。
api.remote.ssl.client.auth.on	**true**	**true** を設定すると、リモート・サーバー API の SSL クライアント認証がオンに切り替わります。
api.remote.naming.port	1099	指定すると、RMI レジストリーが要求を listen するポートになります。
api.remote.server.ports	8700-8900	さまざまな RMI サービスで使用されるポートの範囲です。範囲にはハイフンを使用するコンマ区切りのリストで入力します (例えば、「1, 3-5, 10」)。ポート番号は、> 0 および <=65536 である必要があります。 サーバーは他のプロパティーで定義されるポートでの listen に加えて、これらのポートを使用して着信 RMI サービス要求を listen します。送信する RMI サービス要求には、ランダムなポート番号が使用される。
com.ibm.di.default.bind.address	*	IBM Security Directory Integrator サーバー全体 (コンポーネントおよびサーバー API) のデフォルトのバインド・アドレス。 * は、すべての使用可能なネットワーク・インターフェースへバインドすることを意味します。欠落している値または無効な値でもすべてのインターフェースにバインドされます。値として指定できるのは、1 つの IP アドレスのみです。このプロパティーは、サーバー・ソケットを作成するサーバー・コネクターすべてに影響します。

プロパティー	デフォルト値	説明
api.remote.bind.address	*	RMI サーバー API のバインド・アドレスです。com.ibm.di.default.bind.address プロパティーをオーバーライドします。* は、すべての使用可能なネットワーク・インターフェースに接続することを意味します。欠落している値または空の値は、com.ibm.di.default.bind.address へフォールバックすることを意味します。値として指定できるのは、1 つの IP アドレスのみです。
api.truststore	`testserver.jks`	指定すると、サーバー API のすべてのリモート・ユーザーの公開証明書を含む鍵ストア・ファイルになります。
{protect}-api.truststore.pass	server (デフォルトで暗号化されています)	指定すると、`api.remote.server.truststore` プロパティーで名前が付けられる鍵ストア・ファイルのパスワードになります。
api.remote.nonssl.hosts		指定すると、非 SSL 接続を受け付ける IP アドレスのリストとなります (ホスト名は指定できません)。IP アドレス間の区切り文字として、スペース、コンマ、またはセミコロンを使用します。このプロパティーは、 `api.remote.ssl.on` が **false** に設定されている場合にのみ有効です。
api.jmx.on	**false**	**true** を設定した場合、始動時にサーバー API の JMX レイヤーが初期化されて使用可能になります。それ以外の場合、サーバー API の JMX レイヤーは初期化されず、使用できません。
api.jmx.remote.on	**false**	**true** を設定した場合、リモート JMX インターフェース (JSR160 で定義) が初期化されて使用可能になります。それ以外の場合、リモート JMX インターフェースは初期化されず、使用できません。
api.config.folder	`configs`	*`TDI_root`*`/configs` を設定すると、このフォルダーに配置された構成ファイルのみが、サーバー API を使用した編集の対象になります。
api.config.lock.timeout	0	**0** を設定すると、タイムアウトになりません。
api.custom.method.invoke.on	**false**	`Session.invokeCustom()` メソッドを使用する機能のオン/オフを切り替えます (デフォルトは false、すなわちオフです)。このプロパティーの値に **true** を設定すると、ユーザーはこれらのメソッドを使用できます。それ以外の場合は、例外がスローされます。
api.custom.method.invoke .allowed.classes		指定すると、サーバー API メソッドからカスタム・メソッド起動用 (Session.invokeCustom(...)) に直接起動できるクラスのリストが提供されます。. このプロパティーは、`api.custom.method.invoke.on` に **true** が設定されている場合にのみ有効です。このリスト内のクラスは、スペース、コンマ、またはセミコロンで区切る必要があります。
api.custom.authentication.	[ldap]	指定した場合は、カスタム認証コードを含む JavaScript テキスト・ファイルを指します。組み込み LDAP/JAAS 認証メカニズムを使用可能にするには、このプロパティーに [ldap]/[jaas] を設定します。
com.ibm.di.server.id		指定した場合、サーバー ID が含まれます。同じ IP とポートで動作しているサーバー・グループ内の各サーバーに対して固有な値を割り当てます。
api.config.load.timeout	2	指定した場合、serverapi 構成ロードのタイムアウト値 (分単位) が含まれます。IBM Security Directory Integrator v7.0 で追加されました。
api.notification.suppress	di.server.api.authenticate di.server.api.authorize.*	指定した場合、通知を抑止させるサーバー通知タイプのリストが提供されます。抑止タイプの通知は、通知フレームワークによって伝搬されません。リスト内の通知タイプはスペースで区切られます。ワイルドカードを使用できます。 例: `api.notification.suppress=di.al.*` `di.ci.start` 上記の例では、すべての AssemblyLine 関連の通知と、構成インスタンス開始の通知が抑止されます。プロパティーが欠落しているか、空の場合、抑止される通知はありません。IBM Security Directory Integrator v7.0 で追加されました。
api.client.ssl.custom.properties.on	**true**	サーバー API クライアントのカスタム SSL プロパティーを使用可能にします。true である場合、サーバー API クライアントでは api.client.* プロパティーが使用されます。それ以外の場合は、デフォルトの javax.net.ssl.* プロパティーが使用されます。IBM Security Directory Integrator v7.1 で追加されました。

プロパティー	デフォルト値	説明
api.client.keystore	serverapi/testadmin.jks	サーバー API クライアントの鍵ストア。
api.client.keystore.pass	administrator	「api.client.keystore」プロパティーで指定されている鍵ストアのパスワード。
api.client.keystore.type	jks	「api.client.keystore」プロパティーで指定されている鍵ストア・ファイルのタイプ。オプションのプロパティー。指定しなかった場合は、JVM 用のデフォルトの鍵ストア形式が使用されます。
api.client.key.pass	administrator	秘密鍵のパスワード。この鍵は「api.client.keystore」プロパティーで指定されている鍵ストアに保管されています。
api.client.truststore	serverapi/testadmin.jks	サーバー API クライアントのトラストストア。
api.client.truststore.pass	administrator	「api.client.truststore」プロパティーで指定されているトラストストアのパスワード。
api.client.truststore.type	jks	api.client.truststore プロパティーで指定されている鍵ストア・ファイルのタイプ (オプションのプロパティー)。指定しなかった場合は、JVM 用のデフォルトの鍵ストア形式が使用されます。

注: サーバー API がその構成に使用する Java システム・プロパティーは、クライアントが Java プログラムであるか、あるいは IBM Security Directory Integrator サーバーの異なるインスタンスであるかにかかわらず同一です。ただし、これらの Java システム・プロパティーの設定方法は異なる場合があることに注意する必要があります。 IBM Security Directory Integrator では、これらのプロパティーは通常 `global.properties` または `solution.properties` ファイルを編集することによって設定しますが、Java プログラムでは、コマンド行で -D Java コマンド行スイッチを使用するか、Java プログラム内で Java コードを使用することによって (`java.lang.System.setProperty(key,value)` 標準 Java メソッドを使用する) 指定できます。

仮想プライベート・ネットワークでのリモート・サーバー API アクセス

以下の情報を参照することで、クライアント VPN からのリモート・サーバー API へのアクセスについて学習できます。

仮想私設網 (VPN) 上のクライアントからリモート・サーバー API にアクセスする場合、VPN はサーバー API クライアント・コンピューターに IP アドレスを割り当てます。この VPN が割り当てる IP アドレスは、RMI Java システム・プロパティーで指定する必要があります。サーバー API クライアントがリモート構成エディターの場合、このプロパティーは、`global.properties` または `solution.properties` 内で、プロパティー・ファイルに以下の行を追加することによって設定できます。

```
java.rmi.server.hostname=<IP_address>
```

ここで、*IP_address* は VPN が割り当てる IP アドレスです。

サーバー API クライアントがカスタム Java プログラムの場合、このプロパティーは、以下のようにして Java コード内から設定できます。

```
java.lang.System.setProperty("java.rmi.server.hostname", "IP_Address");
```

ここで、*IP_address* は VPN が割り当てる IP アドレスです。

RMI Java システム・プロパティーは、サーバー API に関連する RMI コードより前に設定する必要があることに注意してください。

サーバー API のアクセス・オプション

以下の情報を参照することで、サーバー API へのさまざまなアクセス方法を学習できます。

サーバー API は、以下に示すようにさまざまな方法で使用できます。

- ネットワーク接続を介してリモート構成エディターからサーバー API にアクセスする
- リモート IBM Security Directory Integrator サーバーで実行中の IBM Security Directory Integrator コンポーネントからサーバー API にアクセスする (リモート・サーバー API アクセス)。このようなコンポーネントの例を以下に示します。
  - システム・キュー・コネクター
  - サーバー通知コネクター

  など。
- IBM Security Directory Integrator サーバーの同一 Java 仮想マシンからサーバー API へアクセスする (ローカル・サーバー API アクセス)。この場合、前述のオプション以外にも、フック内の JavaScript またはスクリプト・コンポーネントからサーバー API にアクセスできます。
- 非 IBM Security Directory Integrator Java アプリケーションからサーバー API にアクセスする。これが機能するには、以下の条件が必要です。
  - クライアント・サイドに Java 7.0.4 以上が必要です。
  - リモート・サイドの CLASSPATH に以下の jar ファイルが組み込まれている必要があります。
    - jars/common/diserverapi.jar
    - jars/common/diserverapirmi.jar
    - jars/3rdparty/others/log4j-1.2.16.jar
    - jars/common/miconfig.jar
    - jars/common/miserver.jar
    - jars/common/mmconfig.jar
    - jars/common/tdiresource.jar
    - jars/3rdparty/IBM/icu4j-50_1_1.jar
    - jars/3rdparty/IBM/jlog.jar

    これらの jar ファイルは IBM Security Directory Integrator インストール済み環境からコピーできます。
  - サーバー API を使用してインプリメントされるソリューションで IBM Security Directory Integrator 以外のカスタム・オブジェクトが使用される場合 (項目の属性値が接続を介して転送される場合など) は、クライアント・サイドでも対応する Java クラスが使用可能でなければなりません。これらのクラスはシリアライズ可能であり、クライアント JVM の CLASSPATH に組み込まれている必要があります。

サーバー API の SSL リモート・アクセス

サーバー API のリモート・インターフェースでは SSL を使用できます。リモート・アクセスの詳細については、以下の情報を参照してください。

サーバー API は、ローカルおよびリモートの 2 つのインターフェースのセットを提供しています。 SSL を使用できるのは、リモート・インターフェースのみです。ローカル・インターフェースは、アクセスが Java 仮想マシンの境界内に限られるため SSL を使用しません。 IBM Security Directory Integrator は、サーバー API のアクセス・シナリオにおいて、サーバーとして、クライアントとして、さらにクライアントおよびサーバーの両方として機能できます。サーバー API とともに SSL を使用する場合は、鍵ストアおよびトラストストアを構成する必要があります。これらを構成するための 2 つのオプションがあります。どちらのオプションを使用するかは、Java システム・プロパティー `api.client.ssl.custom.properties.on` が存在するかどうか、およびその値によって決まります。

サーバー API 固有の SSL プロパティーの使用

リストしたプロパティー・セットを使用して SSL を構成します。

Java システム・プロパティー api.client.ssl.custom.properties.on が *true* に設定されている場合は、以下の IBM Security Directory Integrator サーバー API 固有 Java システム・プロパティーを介して SSL を構成します。

- **api.client.keystore** – クライアント証明書が含まれている鍵ストア・ファイルを指定します。
- **api.client.keystore.pass** – api.client.keystore に指定する鍵ストア・ファイルのパスワードを指定します。
- **api.client.keystore.type** – api.client.keystore オプション・プロパティーで指定されている鍵ストア・ファイルのタイプを指定します。指定しないと、JVM のデフォルトの鍵ストア形式が使用されます。
- **api.client.key.pass** – api.client.keystore に含まれている鍵ストア・ファイルに保管されている秘密鍵のパスワードを指定します。このプロパティーがない場合は、**api.client.keystore.pass** で指定されているパスワードが使用されます。
- **api.client.truststore** – IBM Security Directory Integrator サーバーの公開証明書を含む鍵ストア・ファイルを指定します。
- **api.client.truststore.pass** – api.client.truststore で指定された鍵ストア・ファイルのパスワードを指定します。
- **api.client.truststore.type** – api.client.truststore オプション・プロパティーで指定されている鍵ストア・ファイルのタイプを指定します。指定しないと、JVM のデフォルトの鍵ストア形式が使用されます。

クライアント・アプリケーションが標準の Java SSL プロパティーを使用している場合は、サーバー API 固有の SSL プロパティーを使用します。標準の Java SSL プロパティーは、同じアプリケーションで使用する別の SSL チャネルを構成するために使用するプロパティーです。

これらのプロパティーを、コマンド行で JVM 引数として指定できます。例:

```
java MyTDIServerAPIClientApp
 -Dapi.client.ssl.custom.properties.on=true
 -Dapi.client.truststore=C:¥TDI¥serverapi¥testadmin.jks
 -Dapi.client.truststore.pass=administrator
 -Dapi.client.keystore=C:¥TDI¥serverapi¥testadmin.jks
 -Dapi.client.keystore.pass=administrator
```

この例では、IBM Security Directory Integrator に同梱の「testadmin.jks」鍵ストア・ファイルを参照しています。このファイルには、クライアントの秘密鍵および IBM Security Directory Integrator サーバーの公開鍵の両方が含まれるため、これを鍵ストアおよびトラストストアの両方として使用することに注意してください。

クライアントが IBM Security Directory Integrator サーバーの場合は、これらのプロパティーは `global.properties` または `solution.properties` で指定できます。

標準 SSL Java システム・プロパティーの使用

リストした JVM コマンドを使用して SSL を構成します。

Java システム・プロパティー `api.client.ssl.custom.properties.on` が指定されていない場合、またはこれが「false」に設定されている場合は、SSL チャネルの構成に標準の JSSE システム・プロパティーが使用されます。標準 JSSE 手順に従って、クライアント・アプリケーションにより使用される鍵ストアとトラストストアを構成します。

これらのプロパティーを、コマンド行で JVM 引数として指定できます。例:

```
java MyTDIServerAPIClientApp
 -Djavax.net.ssl.keyStore=C:¥TDI¥serverapi¥testadmin.jks
 -Djavax.net.ssl.keyStorePassword=administrator
 -Djavax.net.ssl.trustStore=C:¥TDI¥serverapi¥testadmin.jks
 -Djavax.net.ssl.trustStorePassword=administrator
```

また、クライアントが IBM Security Directory Integrator サーバーの場合は、これらのプロパティーを `global.properties` または `solution.properties` で指定できます。

サーバー API 認証

サーバー API セッションの設定を試行する場合は、サーバー API 認証を使用できます。

サーバー API の認証は、通常、サーバー API セッションを確立するリモート・サーバー API クライアントに関連して言及されます。サーバー API はいくつかの異なる種類のクライアント認証を提供するため、このシナリオによりサーバー API の認証ロジックの本質が示されます。ただし、さまざまな認証メカニズムの説明に入る前に、ローカル・クライアントがローカル・サーバー API セッションを確立するシナリオについて説明します。

ローカル・クライアント・セッション

クライアントをローカルで処理する場合は、以下の指示に従ってください。オプションについては、以下の情報を参照してください。

ローカル・クライアント・セッションは、IBM Security Directory Integrator サーバーと同じ Java 仮想マシンで実行されているクライアントによって確立されるセッションです。このようなセッションの例としては、フック内の JavaScript コードまた

はスクリプト・コンポーネントや、同じ IBM Security Directory Integrator サーバーで実行されている AssemblyLine の一部として実行されるコネクターおよび関数コンポーネントなどから確立されたローカル・サーバー API にアクセスするためのローカル・セッションが挙げられます。ローカル・クライアントがローカル・サーバー API セッションを確立する場合、クライアントには以下の 2 つのオプションがあります。

- ユーザー名およびパスワードのペアを提供しない - この場合、ローカル・サーバー API セッションが確立され、クライアントは「管理 (admin)」の役割を持つものとして許可されます。サーバー API の役割についての詳細は、140 ページの『サーバー API の許可』を参照してください。
- ユーザー名およびパスワードのペアを提供する – この場合、140 ページの『サーバー API の許可』セクションで説明されているサーバー API 許可ロジックに従って、ユーザー名およびパスワードのペアで指定される「ユーザー名」が許可された後でのみ、サーバー API セッションが確立されます。このオプションは通常、デモやプロトタイピングを行う場合など、認証のために特定のユーザー ID が必要な場合に使用されます。

リモート・クライアント・セッション

クライアントをリモートで処理する場合は、以下の指示に従ってください。認証タイプの詳細については、以下の情報を参照してください。

リモート・クライアント・セッションは、IBM Security Directory Integrator サーバーと同じ Java 仮想マシンで実行されないクライアントによって確立されるセッションです。このようなセッションの例としては、構成エディターから確立されるリモート・サーバー API へのアクセスのセッション、または IBM Security Directory Integrator サーバーへの接続を求める Java アプリケーションへのアクセスのセッションなどがあります。このようなアクセスでの IBM Security Directory Integrator サーバーへの認証方式を次に説明します。

JAAS 認証

JAAS 認証を使用して、ユーザーに対するアクセス制御を確認することができます。プロパティー構成の詳細については、以下の情報を参照してください。

Java Authentication and Authorization Service (JAAS) は、IBM Security Directory Integrator サーバー API の認証モジュールとしてサポートされています。JAAS は、ユーザーに対して認証およびアクセス制御を実施するサービスを使用可能にする API のセットです。JAAS 認証は、IBM Security Directory Integrator サーバー API を利用することによって容易となります。JAAS 認証モジュールを使用するために、CLI や AMC などの IBM Security Directory Integrator サーバー API クライアントで変更を行う必要はありません。

JAAS 認証を使用するには、`global.properties` または `solution.properties` に適切なプロパティーを設定し、JAAS Logon をインストールする必要があります。

SSL ベースの認証

SSL ベースの認証を介してクライアントの資格情報の 2 段階検証を使用できます。

これは、IBM Security Directory Integrator 6.0 で使用可能な唯一の認証メカニズムです。SSL ベース認証は、クライアントの信任状の 2 段階検証に基づいています。

1. 最初に、IBM Security Directory Integrator サーバーは、クライアントの SSL 証明書が IBM Security Directory Integrator サーバーのトラストストアに含まれているかどうかをチェックすることにより、(その SSL 証明書によって表される) クライアントが IBM Security Directory Integrator サーバーにアクセスする権限を保持しているかどうかを検証します (すなわち、IBM Security Directory Integrator サーバーがこのクライアントを信頼するかどうかをチェックします)。クライアントの証明書がサーバーのトラストストアに含まれているかどうかのチェックは、SSL のハンドシェーク・シーケンスの一部です。

   重要: ファイル `testserver.jks` 内のサーバー証明書の例に対応するクライアント証明書の例が、ファイル `serverapi/testadmin.jks` に提供されています。証明書のパスワードは「administrator」です。すべてのデフォルトのセキュリティー・パラメーターと同様、これらの例に依存して独自のクライアント/サーバー証明書を生成するのではなく、プロパティー・ファイルで指定してください。169 ページの『IBM Security Directory Integrator Web サービス・スイートの証明書』を参照してください。

   トラストストアは、*api.truststore* プロパティーで指定したファイル内に保持されます。
2. トラストストアのチェックが成功すると、サーバーは、クライアント SSL 証明書の識別名 (DN)が 143 ページの『サーバー API ユーザー・レジストリー』 内のユーザー ID と一致することを検証します。クライアント証明書の DN がサーバー API のユーザー・レジストリー・ファイル内のいずれのユーザー ID とも一致しない場合は、クライアントからの接続要求は否認されます。この 2 番目のステップは、許可シーケンスの一部とみなすこともできます。

SSL ベース認証メカニズムは、IBM Security Directory Integrator ではオフにできます。IBM Security Directory Integrator サーバー構成ファイル `global.properties` または `solution.properties` には、追加プロパティー **api.remote.ssl.client.auth.on** があります。このプロパティーが「true」に設定された場合、IBM Security Directory Integrator サーバーは、SSL ハンドシェーク内でのクライアント認証を必要とします (SSL ベース認証のための IBM Security Directory Integrator 6.0 のメカニズム)。IBM Security Directory Integrator サーバー API の SSL クライアント認証は、ユーザー名およびパスワードのペアが提供されているかどうかに依存しません。これは、ユーザー名およびパスワードのペアが提供されていない場合は、SSL ベース認証に IBM Security Directory Integrator 6.0 のメカニズムが使用されることを意味します。また、ユーザー名およびパスワードのペアが提供されている場合は、クライアントはやはり認証のためにその SSL 証明書を送信する必要がありますが、認証 (および後のステップの許可) のためのユーザー ID には提供されたユーザー名が採用されます。

api.remote.ssl.client.auth.on が「false」に設定されていると、SSL ベース認証は使用できません。このプロパティーが指定されていない場合は、「false」の値であるとみなされます。

「ユーザー名/パスワード」ベースの認証

認証フックを使用して、「ユーザー名/パスワード」ベースの認証を実行できます。

このメカニズムでは、クライアントが、その IBM Security Directory Integrator サーバーに対するサーバー API 接続をオープンする際に、ユーザー名およびパスワードを提供する必要があります。この認証メソッドを構成するために、認証フックを使用します。

認証フック

このフックにより、ユーザー名およびパスワード・ベースの認証を実行するカスタム JavaScript コードのプロビジョンが可能になります。このフックにより、バンドラー/デプロイヤーによるカスタマイズされた JavaScript コードへの書き込みが可能になり、このコードは、ユーザー名およびパスワードのペアを受け取ると、認証が成功するかそうでないかを判断します。

このカスタム JavaScript 認証を許可するプロパティー **api.custom.authentication** は、IBM Security Directory Integrator サーバー構成ファイル `global.properties` または `solution.properties` に指定されます。api.custom.authentication プロパティーは、カスタム認証コードを含むディスク上の JavaScript テキスト・ファイルを指します。このプロパティーが指定されていない場合は、IBM Security Directory Integrator 6.0 の SSL ベースの認証メカニズムが使用されます。api.custom.authentication プロパティーが指定されている場合は、ユーザー名およびパスワード・ベースの各認証要求に対して、指定されたファイルに含まれる JavaScript コードが実行されます。

認証スクリプトは、事前定義されたスクリプト・オブジェクト userdata にアクセスできます。このオブジェクトは、以下の 2 つのパブリック・メンバーを提供します。

- **userdata.username** - 認証を要求するユーザーの名前が含まれます。
- **userdata.password** - ユーザーが提供するパスワードが含まれる

スクリプトは、必要なチェックおよび認証アクションをすべて自由に実行できます。スクリプトは、**ret** オブジェクトを介して、認証が成功したかどうかを戻します。

- 認証が成功したことを指定するには、**ret.auth = true** を設定します。
- 認証が失敗したことを指定するには、**ret.auth = false** を設定します。この場合、認証スクリプトにより、**ret.errordescr** 属性 (例えば、*ret.errordescr = "Invalid user name"*) および **ret.errorcode** 属性 (例えば、*ret.errorcode = 1*) を介して、認証が失敗した理由についての追加情報を提供できます。

説明およびエラー・コードのフィールドは、認証が失敗した場合にサーバー API によってスローされる AuthenticationException によって提供されます。

認証スクリプトは、メイン・スクリプト・オブジェクトにアクセスできます。これは、IBM Security Directory Integrator サーバーのログ・ファイルにカスタム・メッセージを記録するために使用できます (例えば、main.logmsg("Authentication failed for user : " + userdata.username))。

認証フックの例

サンプルの認証フック JavaScript ファイル (*TDI_install_dir*`/examples` 内) は、認証フックの JavaScript がどのようなものであるかを説明するために使用できます。

このサンプル JavaScript はまた実際の IBM Security Directory Integrator 認証フックの基礎としても使用できます。このサンプル JavaScript は、クライアント要求を認証するために、認証フックがどのようにして LDAP サーバー (IBM Security Directory Integrator、Active Directory など) を使用できるかを示します。

JavaScript ファイルは、「ldap_auth.js」という名前で、IBM Security Directory Integrator サーバー・フォルダー examples/auth_ldap にインストールされます。このサンプル LDAP 認証メカニズムをデプロイするには、ファイルを IBM Security Directory Integrator ソリューション・フォルダーにコピーし、global.properties または solution.properties で api.custom.authentication=ldap_auth.js を指定します。「ldap_auth.js」内の JavaScript コードは、指定されたユーザー名およびパスワードで LDAP サーバーにバインドしようとします。バインド操作が成功すると、スクリプトにより、認証が成功したことが示されます。それ以外の場合、認証は拒否されます。LDAP サーバーへの接続に関する サーバー URL などの詳細については、「ldap_auth.js」スクリプトで指定します。これは、スクリプトを使用する前にユーザーがこのファイルを編集し、適切な接続パラメーターを設定する必要があることを意味します。以下に、サンプルの「ldap_auth.js」スクリプトを示します。

```
env = new Packages.java.util.Hashtable();
env.put("java.naming.factory.initial", "com.sun.jndi.ldap.LdapCtxFactory");
env.put("java.naming.provider.url", "ldap://192.168.113.54:389");
env.put("java.naming.security.principal", userdata.username);
env.put("java.naming.security.credentials", userdata.password);
env.put(Packages.javax.naming.Context.SECURITY_AUTHENTICATION, "simple");

main.logmsg("Authentication request for user: " +  userdata.username);

try
{
 mCtx = new Packages.javax.naming.directory.InitialDirContext(env);
 ret.auth = true;
}
catch(e)
{
 ret.auth = false;
 ret.errordescr = e.toString();
// ret.errorcode = "49";
}
```

LDAP 認証サポート

IBM Security Directory Integrator サーバー API は LDAP 認証をサポートします。これにより、既にユーザーID およびパスワードが保持されている既存の LDAP インフラストラクチャーを活用できます。

LDAP 認証の構成:

リストされているプロパティーを操作して、LDAP 認証を構成することができます。

LDAP 認証を使用するには、global.properties または solution.properties 内で適切なプロパティーを構成する必要があります。これらのプロパティーとその説明のリストを以下に示します。

api.custom.authentication

これは、ユーザー名およびパスワードの認証に使用するものと同じプロパティーです。ユーザー名/パスワードの認証に関する詳細については、『ユーザー名/パスワード・ベースの認証』セクションを参照してください。このプロパティーは、カスタム認証コードを含むディスク上の JavaScript テキス

ト・ファイルを指します。ユーザーはこのプロパティーを指定できない場合があります。その場合、IBM Security Directory Integrator 6.0 の SSL ベースの認証メカニズムのみを使用することができます。IBM Security Directory Integrator バージョン 7.2 のユーザー名およびパスワードの認証は機能しません。このプロパティーを「**[ldap]**」に設定し (api.custom.authentication=[ldap] のようにします)、IBM Security Directory Integrator バージョン 7.2 組み込み LDAP 認証メカニズムを使用可能にします。「*api.custom.authentication.ldap.*」で始まるすべてのプロパティーは、*api.custom.authentication* が *[ldap]* に設定されている場合にのみ使用可能です。

api.custom.authentication.ldap.critical
このパラメーターは、開始時に LDAP 認証モジュールを初期化できない場合のサーバー API の振る舞いを指定します。このパラメーターが「true」に設定されていると、サーバー API の初期化は失敗し、サーバー API は開始されません。

このパラメーターが指定されていないか、「false」に設定されていると、サーバー API は LDAP 認証の初期化エラーをログに記録しますが、サーバー API は開始されます。LDAP 認証モジュールの初期化は、サーバー API が認証要求を受信するたびに、LDAP 認証モジュールが初期化されるまで試行されます。

api.custom.authentication.ldap.hostname
LDAP サーバーのホスト名。LDAP カスタム認証を使用する場合、これは必須のプロパティーです。

api.custom.authentication.ldap.port
LDAP サーバーのポート番号。例えば、非 SSL の場合は 389、SSL の場合は 636 です。LDAP カスタム認証を使用する場合、これは必須のプロパティーです。

api.custom.authentication.ldap.ssl
LDAP サーバーと通信するために SSL を使用するかどうかを指定します。「true」を設定すると、SSL が使用され、そうでない場合、SSL は使用されません。

api.custom.authentication.ldap.searchbase
ユーザー検索を実行する LDAP ディレクトリーのロケーションを指定します。このプロパティーが指定されていない場合、ユーザー検索は実行されません。

api.custom.authentication.ldap.admindn
ユーザー検索に使用する LDAP サーバー管理者の識別名を指定します。このプロパティーが指定されていない場合、ユーザー検索に匿名バインドが使用されます。

api.custom.authentication.ldap.adminpassword
LDAP サーバー管理者の識別名のパスワード。

api.custom.authentication.ldap.userattribute
検索に使用するユーザー ID 属性を指定します。このプロパティーが指定されていない場合、ユーザー検索は実行されません。このプロパティーの設定例: api.custom.authentication.ldap.userattribute=cn。

必須プロパティーが指定されていない場合、初期化時に例外がスローされます。

api.custom.authentication.ldap.searchbase または **api.custom.authentication.ldap.userattribute** のいずれかの値が指定されていない場合、実際のユーザー認証時に、検索コンテキストが初期化されず、検索が実行されません。(検索が実行されないというのは、LDAP サーバーへのバインドが、認証のために提供されたユーザー名およびパスワードを使用して直接試行されることを意味します。)

api.custom.authentication.ldap.admindn が指定されている場合、「単純」認証を使用して検索コンテキストが作成されます。検索コンテキストの初期化時にエラーが発生した場合、LDAP 認証モジュールの初期化は失敗し、例外がスローされます。

api.custom.authentication.ldap.admindn が指定されていない場合、JNDI「匿名」バインドを使用して、JNDI 検索コンテキストが作成されます。

注: api.custom.authentication.ldap.admindn を使用して検索コンテキストを初期化できない場合、直接の認証は失敗します。匿名バインドは試行されません。

LDAP 認証のロジック:

リストされているパスを使用して、LDAP 認証用の資格情報を認証します。

ユーザーの認証が試行されるたびに、認証を受けるユーザーのユーザー名およびパスワードが LDAP 認証モジュールに渡されます。以下の認証パスが考えられます。

- **api.custom.authentication.ldap.searchbase** プロパティーおよび **api.custom.authentication.ldap.userattribute** プロパティーの両方が指定されている場合:
  - 認証のために提供されるユーザー名が **api.custom.authentication.ldap.searchbase** プロパティーの値で終了している場合、完全識別名が提供されているとみなされ、ユーザー検索は実行されません。認証のために提供されたユーザー名およびパスワードを使用して、LDAP サーバーへのバインドが直接試行されます。バインドが成功した場合認証は成功したとみなされ、そうでない場合認証は失敗したとみなされます。
  - ユーザー名が **api.custom.authentication.ldap.searchbase** プロパティーの値で終了していない場合、初期化時に作成された検索コンテキストに対して、サブツリー検索範囲を使用した検索が実行されます。使用される検索照会は「(*<LDAPUserIDAttribute>=<username>*)」で、*LDAPUserIDAttribute* は **api.custom.authentication.ldap.userattribute** プロパティーの値、*username* は認証のために提供されるユーザー名です。1 つの検索結果のみが戻される場合、戻されたエントリーの識別名および認証のために提供されたパスワードを使用して、LDAP サーバーへのバインドが実行されます。LDAP サーバーへのバインドが成功した場合にのみ、認証が成功します。他のすべての場合において、認証は失敗したとみなされます。複数の検索結果が戻される場合、認証は失敗します。
- **api.custom.authentication.ldap.searchbase** プロパティーまたは **api.custom.authentication.ldap.userattribute** プロパティーの少なくとも一方が指定されていない場合。

この場合、検索は実行されず、認証のために提供されたユーザー名およびパスワードを使用して、LDAP サーバーへのバインドが直接試行されます。バインドが成功した場合認証は成功したとみなされ、そうでない場合認証は失敗したとみなされます。

LDAP グループのサポート:

ユーザー・レジストリーには、ユーザーに対するものと同じ方法でグループに対して権限を設定することができます。ユーザーとグループは、リストされているプロパティーを使用して区別できます。

管理を容易にするため、IBM Security Directory Integrator では権限を、ユーザーに対して構成する方法と同じ方法でグループに対しても構成できるようになっています。権限はユーザーに対して使用する構文と全く同じ構文でユーザー・レジストリーに設定できます。実際、ユーザー・レジストリーは、セキュリティー・エンティティーがグループかユーザーかを認識しません。ユーザーとグループの違いが現われるのは、認証プロセスが行われるときです。

LDAP ディレクトリーにはグループ・メンバーシップが構成され、IBM Security Directory Integrator はそれに対してユーザーを認証します。特定のユーザーが特定の LDAP グループのメンバーだった場合、ユーザーが認証されると、そのグループに対するすべての権限がユーザーに自動的に継承されます。グループのサポートはデフォルトでは使用不可になっているので、使用可能にする必要があります。

LDAP グループのサポートに関係するシステム・プロパティーは次のとおりです。

api.custom.authentication.ldap.groupsupport
これは、オプションのプロパティー (ブール値のフラグ) です。このプロパティーが存在しない場合は、デフォルト値「false」が使用されます。ユーザーを認証するとき、グループ・メンバーシップも解決するかどうかを指定します。グループ・メンバーシップを解決する場合は、認証時に有効となります。

api.custom.authentication.ldap.usermembershipattribute
このプロパティーが必要になるのは、**api.custom.authentication.ldap.groupsupport** が true に設定されている場合のみです。 LDAP のユーザーがメンバーになっているグループのリストを含む LDAP ユーザー属性の名前を指定します。

api.custom.authentication.ldap.usermembershipattributecontent
このプロパティーが必要になるのは、**api.custom.authentication.ldap.groupsupport** が true に設定されている場合のみです。ユーザーのメンバーシップ属性内のグループ名の付与方法を指定します。例えば、ユーザーのメンバーシップ属性にグループの「objectSID」属性に対応する値がある場合、このプロパティーには「objectSID」を設定します。ユーザーのメンバーシップ属性にグループの識別名 (distinguished name) が含まれる場合、このプロパティーには「dn」を設定します。

api.custom.authentication.ldap.groupnameattribute
このプロパティーが必要になるのは、**api.custom.authentication.ldap.groupsupport** が true に設定されている場合のみです。 IBM Security Directory Integrator ユーザー・レジストリー内で

グループ名が付与される方法に対応する LDAP グループ属性の名前を指定します。例えば、LDAP グループが IBM Security Directory Integrator レジストリー内では共通名 (common name) でアドレス指定されているとき、このプロパティーには「cn」を設定します。ユーザー・レジストリーにグループの識別名が含まれる場合は、このプロパティーには「dn」を設定します。

api.custom.authentication.ldap.groupsearchbase
このプロパティーが必要になるのは、**api.custom.authentication.ldap.groupsupport** が true に設定されている場合のみです。グループが検索される、LDAP ディレクトリーのコンテキストを示します。

api.custom.authentication.ldap.binaryattributes
これはオプションのプロパティーです。 - スペースで区切られた属性名のリストを示します。非ストリング構文の属性を指定します。

Active Directory の例

この例は、**Active Directory** サーバーで動作するためのグループ・サポートの構成方法を示します。

```
api.custom.authentication.ldap.groupsupport=true
api.custom.authentication.ldap.usermembershipattribute=tokenGroups
api.custom.authentication.ldap.usermembershipattributecontent=objectSID
api.custom.authentication.ldap.groupnameattribute=sAMAccountName
api.custom.authentication.ldap.groupsearchbase=DC=mytestadserver,DC=com
api.custom.authentication.ldap.binaryattributes=objectSID tokenGroups
```

「tokenGroups」属性は、Active Directory のすべてのユーザーに存在する、計算された属性です。

これは、ユーザーがメンバーになっている、すべてのセキュリティー・グループのセキュリティー識別子 (SID) の集合を含んでいます。

この集合にはセキュリティー・グループしか含まれません (電子メールで使用される配布グループは含まれません)。ネストされたグループやプライマリー・グループを含む、すべてのセキュリティー・グループが含まれます。

セキュリティー識別子はバイナリー属性です。したがって、**api.custom.authentication.ldap.binaryattributes** プロパティーに設定する必要があります。

上記の例では、グループは IBM Security Directory Integrator ユーザー・レジストリー内の LDAP 属性「sAMAccountName」に従って命名されています。

IBM Security Directory Server の例

この例は、IBM Security Directory Server で動作するためのグループ・サポートの構成方法を示します。

```
api.custom.authentication.ldap.groupsupport=true
api.custom.authentication.ldap.usermembershipattribute=ibm-allGroups
api.custom.authentication.ldap.usermembershipattributecontent=dn
api.custom.authentication.ldap.groupnameattribute=dn
api.custom.authentication.ldap.groupsearchbase=ou=mytestou,c=mytestcountry
```

特定のユーザー項目に対して、「ibm-allGroups」操作属性は、そのユーザーがメンバーシップを持つ、静的グループ、動的グループ、およびネストされたグループをすべて列挙しています。

注:

1. IBM Security Directory Integrator は、LDAP ユーザー項目を直接検査することによってグループ・メンバーシップを判別します (すべてのグループを走査することによってメンバーシップを間接的に判別するのとは対照的です)。このアプローチを正しく機能させるには、ユーザー項目にユーザーがメンバーになっているグループを列挙した属性が必要です。グループ・サポートが機能するのは、各ユーザー項目でこのようなメンバーシップ属性をサポートする LDAP サーバーに対してのみです。
2. ユーザーのグループ・メンバーシップを変更しても、既存のサーバー API セッションには影響ありません。ただし、変更後に確立されるセッションには反映されます。
3. 現在グループ・サポートが提供されているのは、LDAP 認証のみです。 JAAS 認証や、カスタム JavaScript を使用する認証に、グループ・サポートは提供されません。
4. サーバー API で SSL クライアント認証が使用可能になっている場合、ユーザー名を指定しなかったクライアントは、SSL クライアント証明書の所有者に基づいて、認証され、許可されます。グループ・サポートを使用する LDAP 認証も (SSL クライアント認証とともに) 使用可能になっている場合、グループ・メンバーシップは、SSL クライアント証明書の所有者について解決されます。

ホスト・ベース認証

ホスト・ベース認証を使用するには、ホスト・ベース・プロパティーを構成します。詳しくは、以下の情報を参照してください。

ホスト・ベース認証は、`global.properties` または `solution.properties` ファイルで *api.remote.ssl.on=false* を指定することにより、SSL がオフにされている場合に使用されます。ホスト・ベース認証は、*api.remote.nonssl.hosts* プロパティーを使用して構成します。このプロパティーは、リモート・サーバー API クライアントがユーザー名およびパスワードを指定することなくサーバー API を使用できるようホスト IP アドレスのリストを指定します。

このホストのリストの構文は以下のとおりです。IP アドレスのリスト (ホスト名は使用できません) および IP アドレス間の区切り文字としてスペース、コンマ、またはセミコロンを使用します。このプロパティーの値の例:

```
api.remote.nonssl.hosts=192.168.111.222, 192.168.112.158
```

ホスト・ベース認証を使用するクライアントが正常に認証された場合、クライアントには管理 (admin) 許可権限が付与されます。このため、このリストへの IP アドレスの追加には十分な注意が必要です。セキュリティーの観点から、実稼働環境におけるホスト・ベース認証の使用は推奨されません。ホスト・ベース認証は通常、ソリューションの開発中またはデモの実行時に使用されます。

サーバー API 認証オプションの要約

サーバー API 認証オプションがいくつかあります。以下にそれらの要約リストを示します。

以下の認証オプションが使用可能です。

SSL ベース認証 (IBM Security Directory Integrator 6.0 で使用可能なメカニズム)
api.remote.ssl.client.auth.on=true の場合にのみ機能します (*api.on=true*、*api.remote.on=true*、*api.remote.ssl.on=true* も必要です)。ユーザーは、サーバー API のユーザー・レジストリー内の SSL 証明書のユーザー ID に割り当てられた権限に応じて許可されます。

注: SSL が使用され、リモート・クライアント・アプリケーションがサーバー API リスナー・オブジェクトを使用する場合、クライアント・アプリケーションは、IBM Security Directory Integrator サーバーによって信頼される独自の証明書を保持している必要があります (これは、SSL クライアント認証のセットアップに類似しています)。IBM Security Directory Integrator サーバーによって信頼されたクライアント証明書がない場合、リスナー・オブジェクトは機能せず、リモート・クライアント・アプリケーションは IBM Security Directory Integrator サーバーからの通知を受信できません。

「ユーザー名/パスワード」ベースの認証
api.custom.authentication が JavaScript 認証ファイルに設定されている場合にのみ機能します。この認証方式は、SSL が使用されているかどうか、および SSL クライアント認証が使用されているかどうかに関係なく機能します。ユーザーは、143 ページの『サーバー API ユーザー・レジストリー』内の *username* ユーザーに割り当てられた権限に応じて許可されます。

LDAP 認証
この認証については、132 ページの『LDAP 認証サポート』で説明しています。この認証は、 `global.properties` または `solution.properties` の複数の *api.custom.authentication* 設定に基づいて決まります。

ホスト・ベース認証
api.remote.ssl.on=false の場合にのみ機能します。*api.remote.nonssl.hosts* プロパティーによって指定されたすべてのホストから提供されるユーザー名およびパスワードを使用しないでサーバー API セッションのオープンが認証され、管理 (admin) 権限が付与されます。*api.remote.nonssl.hosts* プロパティーは、`global.properties` または `solution.properties` ファイルで指定できます。

サーバー API の JMX レイヤー

サーバー API の JMX レイヤーでは、ユーザー名およびパスワード認証がサポートされません。以下にリストされている認証手順を使用して、JMX レイヤーを認証してください。

サーバー API のリモート JMX レイヤーでは、ユーザー名およびパスワード・ベースの認証がサポートされません。*api.custom.authentication* プロパティーが無視されます。これらのプロパティーの値、およびサーバー API についてカスタム認証が使用可能かどうかに関係なく、リモート JMX レイヤーでは以下の認証が実行されます。

- SSL および SSL クライアント認証がオンになっている場合、リモート JMX レイヤーでは SSL ベース認証が実行されます (IBM Security Directory Integrator 6.0 の場合と同様)。
- SSL がオンに、SSL クライアント認証がオフになっている場合、リモート JMX レイヤーは機能しません。
- SSL がオフになっている場合、リモート JMX クライアントのホストが *api.remote.nonssl*.hosts property で指定されている場合、すなわちホスト・ベース認証が想定されている場合にのみリモート JMX クライアントは正常に認証されます。この場合、クライアントには管理 (admin) 権限が付与されます。

最終的な結果として、サーバー API の JMX レイヤーではユーザー名およびパスワード認証はサポートされません。

サーバー API 認証のセットアップの例

サーバー API 認証例のリストを参照できます。これは、サーバーを構成する場合に役立ちます。

認証の構成例:

1. 非 SSL 構成およびカスタム認証:

   ```
   api.remote.ssl.on=false
   api.remote.nonssl.hosts=192.168.113.51, 192.168.113.52
   api.custom.authentication=ldap_auth.js
   ```

   SSL は使用されません。

   - ユーザー名およびパスワードが提供されない認証要求は、ローカル・ホストあるいは 192.168.113.51 または 192.168.113.52 から呼び出された場合にのみ成功します。
   - ユーザー名およびパスワードが提供される認証要求は、ldap_auth.js によりユーザー名およびパスワード・パラメーターを使用して指定されたユーザーが認証される場合にのみ成功します。
   - リモート JMX クライアントは、要求がローカル・ホストあるいは 192.168.113.51 または 192.168.113.52 から送信された場合にのみ認証されます。
2. SSL (クライアント認証なし) およびカスタム認証:

   ```
   api.remote.ssl.on=true
   api.remote.ssl.client.auth.on=false
   api.custom.authentication=ldap_auth.js
   ```

   リモート・サーバー API の通信に SSL が使用されます。

   - SSL クライアント認証またはホスト・ベース認証のいずれもオンになっていないため、ユーザー名およびパスワードが提供されない認証要求は失敗します。
   - ユーザー名およびパスワードが提供される認証要求は、ldap_auth.js によりユーザー名およびパスワード・パラメーターを使用して指定されたユーザーが認証される場合にのみ成功します。
   - この場合、api.remote.ssl.on が true に設定されているため、api.remote.nonssl.hosts パラメーターの値にかかわらず、ホスト・ベース認証は使用できません。

- リモート JMX レイヤーにはアクセスできません。これは、SSL がオンになっていても、 SSL クライアント認証が使用されないためです。

3. クライアント認証ありの SSL およびカスタム認証:

```
api.remote.ssl.on=true
api.remote.ssl.client.auth.on=true
api.custom.authentication=ldap_auth.js
```

リモート・サーバー API の通信に SSL が使用され、サーバーには SSL クライアント認証が必要です。

- ユーザー名およびパスワードが提供されない認証要求は、サーバーのトラストストアにクライアントの SSL 証明書がある場合 (またはトラストストア内の証明書を使用して検証可能な場合) に成功します。
- ユーザー名およびパスワードが提供される認証要求は、SSL クライアント認証が成功する場合 (サーバーのトラストストアにクライアントの SSL 証明書がある場合) および ldap_auth.js スクリプトによって、ユーザー名およびパスワード・パラメーターを使用して指定されたユーザーが正常に認証される場合にのみ成功します。この場合、許可は、SSL クライアント証明書のユーザー ID が使用されるのではなく、提供されるユーザー名およびパスワードのユーザー名パラメーターに基づいて実行されます。
- この場合、*api.remote.ssl.on* が true に設定されているため、*api.remote.nonssl.hosts* パラメーターの値にかかわらず、ホスト・ベース認証は使用できません。
- サーバーのトラストストアにクライアントの SSL 証明書がある場合 (またはトラストストア内の証明書を使用して検証可能な場合) に、リモート JMX クライアントが認証されます。

4. クライアント認証ありの SSL & カスタム認証なし:

```
api.remote.ssl.on=true
api.remote.ssl.client.auth.on=true
api.custom.authentication=
```

(代替手段として、「*api.custom.authentication*」プロパティーをまったく指定しないことも可能です)

リモート・サーバー API の通信に SSL が使用され、サーバーには SSL クライアント認証が必要です。

- ユーザー名およびパスワードが提供されない認証要求は、サーバーのトラストストアにクライアントの SSL 証明書がある場合 (またはトラストストア内の証明書を使用して検証可能な場合) に成功します。
- カスタム認証が構成されていないため、ユーザー名およびパスワードが提供される認証要求は成功しません。
- この場合、*api.remote.ssl.on* が true に設定されているため、*api.remote.nonssl.hosts* パラメーターの値にかかわらず、ホスト・ベース認証は使用できません。
- サーバーのトラストストアにクライアントの SSL 証明書がある場合にのみ、リモート JMX クライアントが正常に認証されます。

サーバー API の許可

以下の指示に従って、ユーザーに権限を割り当てることができます。

クライアントのサーバー API セッション要求が認証された後は、これが許可される必要があります。

リモート API のユーザーには、さまざまな役割を割り当てることができます。役割は、そのユーザーが実行できるサーバー API 呼び出しのリストを定義し、それらの呼び出しを実行できるコンテキストも定義します。ユーザーがサーバー API のメソッドを実行できるのは、そのユーザーが実行しようとしているコンテキストでそのメソッドの実行を許可する役割が少なくとも 1 つ、そのユーザーに割り当てられている場合です。例えば、役割によって、特定の AssemblyLine を特定の構成からのみ実行できるユーザー権限を許可することかできます。これらのユーザー権限を保持するファイルの作成方法については、143 ページの『サーバー API ユーザー・レジストリー』を参照してください。

許可はユーザー ID に基づいて行われます。使用される認証メカニズムに応じて、ユーザー ID は異なる方法で検索されます。

- SSL ベース認証 - ユーザー ID は、クライアントの SSL 証明書の識別名 (DN) です。
- ユーザー名およびパスワード・ベース認証 - ユーザー ID は、ユーザー名およびパスワードのペアで提供されるユーザー名です。
- ホスト・ベース認証 - この認証メカニズムの使用時にはクライアントからユーザー ID は検索できません。この場合、クライアント・セッションには*管理 (admin)* の役割が許可されます。

許可の役割

いくつかの役割を使用してユーザーを許可することができます。サーバー API のセキュリティー・モデルに適用できる役割のリストを示します。

リモート API のユーザーには、役割が割り当てられます。役割は、そのユーザーが実行できるサーバー API 呼び出しのリストを定義し、それらの呼び出しを実行できるコンテキストも定義します。例えば、役割によって、特定の AssemblyLine を特定の構成からのみ呼び出すことができるユーザー権限を許可することができます。

1 ユーザーに複数の役割を割り当てることができます (同じ役割を別々のパラメーターで 2 回以上割り当てることも可能)。ユーザーがサーバー API のメソッドを呼び出せるのは、そのユーザーが実行しようとしているコンテキストでそのメソッドの実行を許可する役割が少なくとも 1 つ、そのユーザーに割り当てられている場合です。

否認は意味を持ちません。アクションを明示的に禁止することはできません。 サーバー API のセキュリティー・モデルには、以下の役割が適用されます。

read 役割: **read [list_of(configuration)]**	*read* 役割は、サーバーの構成からデータを読み取ることをユーザーに許可します。 構成のリストを指定しない場合、またはリストが空である場合、そのユーザーはどの構成を読み取ることも許可されません。 構成のリストとして特殊値 * (アスタリスク) を指定できます。これは、サーバーに現在ロードされているすべての構成を (サーバー API 呼び出しを使用して) そのユーザーが読み取ることを許可することを指定します。 構成のリストが NULL や空ではなく、* も指定されていない場合、そのユーザーは指定されている構成のみ読み取ることが許可されます。 *read* 役割では、プロセス (AssemblyLine) を開始する権限や、サーバーやその構成に対する変更を適用する権限は許可されません。 以下に例を示します。 `[ROLE]:read` `  [CONFIG]:*`
execute 役割: **execute [list_of(configuration [list_of(AssemblyLines)])]**	*execute* 役割は、AssemblyLine を実行するユーザー権限を付与します。 構成のリストを指定しない場合、またはリストが空である場合、そのユーザーはどの構成からの AssemblyLine の実行も許可されません。 構成のリストとして特殊値 * (アスタリスク) を指定できます。これは、すべての構成からすべての AssemblyLine をそのユーザーが実行することを許可することを指定します。 構成のリストが存在しており、* も指定されていない場合、そのユーザーはリストに指定されている構成からのみプロセスを開始することが許可されます。リストに指定されている構成ごとに、以下のように判断されます。 • AssemblyLine のリストが指定されていない場合、そのユーザーにはこの構成から AssemblyLine を実行することは許可されません。 • AssemblyLine のリストに対して特殊値 * (アスタリスク) を指定すると、そのユーザーはこの構成からすべての AssemblyLine を実行することが許可されます。 • AssemblyLine のリストが存在しており、* も指定されていない場合、そのユーザーはリストに指定されている AssemblyLine のみを実行することが許可されます。 例: `[ROLE]:execute` `  [CONFIG]:C:/TDI/rs.xml` `  [AL]:*` `  [CONFIG]:C:/TDI/prototype.xml` `  [AL]:TestAssemblyLine`

admin 役割: **admin**	*admin* 役割は、可能な場合にすべてのサーバー API 呼び出しを実行することをユーザーに許可します。 *admin* 役割を持つユーザーには、構成の読み取りと変更、新しい構成のロード、AssemblyLine の実行、サーバー・パラメーターの読み取りと変更を行うことが許可されます。 例: `[ROLE]:admin` **注:** リモート構成エディターを使用するには admin の役割が必要です。 163 ページの『リモート構成エディターの使用』も参照してください。

[CONFIG] タグ内に指定する値は、構成ファイル名またはソリューション名 (構成ファイルにソリューション名が指定されている場合) のいずれかです。

サーバー API ユーザー・レジストリー

ユーザー・レジストリー・ファイルではサーバー証明書が暗号化されます。その構造の詳細については、以下の情報を参照してください。

ユーザー・レジストリーは、`global.properties` または `solution.properties` ファイルの *api.user.registry* プロパティー で指定するテキスト・ファイルで、 API を使用するすべてのユーザーとその役割についての情報を維持するためのものです。このファイルは、サーバーの証明書 (*api.keystore* プロパティーで指定した鍵ストアから *api.key.alias* プロパティーによって指定する) によって暗号化されます。採用される暗号化アルゴリズムは、非対称 RSA 暗号化または復号です。そのため、 169 ページの『IBM Security Directory Integrator Web サービス・スイートの証明書』には、この目的で使用できる IBM Security Directory Integrator で提供される 156 ページの『IBM Security Directory Integrator 暗号化ユーティリティー』 のデフォルトのアルゴリズムである RSA アルゴリズムを指定する必要があります。始動時に、サーバー API エンジンがこのファイルを復号して読み取り、メモリー構造に格納します。

注:

1. ユーザー・レジストリー・ファイル全体が、RSA アルゴリズムおよびサーバー公開鍵を使用して、そのまま単純にブロック単位で暗号化されます。デジタル署名または何らかのハッシュは使用されません。
2. ユーザー・レジストリーに対する許可はオプションではありません。現時点では、IBM Security Directory Integrator サーバーにはプラグイン許可メカニズムの概念はありません。

ID レジストリー・テキスト・ファイルの内容は、次のような構造になっています。

```
[USER]
[ID]:<user_identifier>
[ROLE]:<role_identifier>
    [CONFIG]:<config_identifier>
         [AL]:<assembly_line_name>
         [AL]:<assembly_line_name>
          ...
    [CONFIG]:<config_id>
         ...
```

```
[ROLE]:<role_identifier>
        ...
[ROLE]:<role_identifier>
        ...
[ENDUSER]

[USER]
[ID]:<user_identifier>
[ROLE]:<role_identifier>
...
[ENDUSER]
...
```

各タグは 1 行ずつ、単独で配置する必要があります。タブおよびスペースは問題になりません。空の行は任意に挿入できます。ID レジストリー・ファイル内で使用されるタグとその引数は、以下のとおりです。

タグ	引数
[USER]	このタグに引数はありません。このタグは、以降のタグの左大括弧としての役割を果たします。**[USER]** タグと **[ENDUSER]** タグをペアとしてそれぞれ 1 行に記述して、レジストリー・ファイル内で 1 ユーザー分の定義を囲みます。このタイプのペアは複数、指定できます。1 つのペアが、サーバー API の 1 ユーザーを指定します。
[ID]:<*user_identifier*>	このタグは、**[USER]** タグに続く最初のタグとして指定します。引数 <*user_identifier*> は、サーバー API ユーザーの固有 ID です。この ID 値は、トラストストア・ファイルから取った 118 です。このタグと引数は 1 行に記述します。**[USER]** と **[ENDUSER]** の 1 ペアの間に指定できる **[ID]:** タグは 1 つのみです。
[ROLE]:<*role_identifier*>	このタグにはユーザーの役割を指定します。指定可能な役割は、**read**、**execute**、または **admin** です。**[ROLE]:** タグとその引数、およびこれに続く内容 (別の **[ROLE]:** タグまたは **[ENDUSER]** タグのうち先に登場するものの前まで)では、このユーザー役割の詳細を指定します。このタグとその引数は 1 行に記述します。**[USER]** と **[ENDUSER]** の 1 ペアの間に複数の **[ROLE]:** タグを記述すると、そのユーザーに対して複数の役割を指定できます。
[CONFIG]:<*config_id*>	このタグには、IBM Security Directory Integrator 構成の ID を、その構成の絶対ファイル・パスで指定します。相対ファイル・パスは認識されません。このタグは **[ROLE]:** タグに従属します。これにより、このタグで指定する構成は、**[ROLE]:** タグで指定した役割に対するものとなります。また、このタグとその引数は 1 行に記述します。複数の **[CONFIG]:** タグを記述することもでき、それらの指定すべては上位の **[ROLE]:** タグに所属します。 **[ROLE]:** タグに **[CONFIG]:** タグを 1 つも関連付けない場合は、対応する役割定義に対する構成のリストが空であるという意味になります。
[AL]:<*assembly_line_name*>	このタグには、AssemblyLine 名を指定します。このタグは、**[CONFIG]:** タグに従属します。このタグとその引数は 1 行に記述します。複数の **[AL]:** タグを記述することもでき、それらの指定すべては上位の **[CONFIG]:** タグに所属します。 **[CONFIG]:** タグに **[AL]:** タグを 1 つも関連付けない場合は、対応する構成 ID に対する AssemblyLine のリストが空であるという意味になります。

次のテキストは ID レジストリー・ファイルの例です。

```
USER]
[ID]:CN=Stan, OU=TDI, O=IBM, C=US
[ROLE]:admin
[ENDUSER]
```

```
[USER]
[ID]:CN=John, OU=TDI, O=IBM, C=US
[ROLE]:read
    [CONFIG]:*
[ROLE]:execute
    [CONFIG]:C:/TDI/rs.xml
        [AL]:*
    [CONFIG]:C:/TDI/prototype.xml
        [AL]:TestAssemblyLine
[ENDUSER]

[USER]
[ID]:CN=Peter, OU=TDI, O=IBM, C=US
[ROLE]:execute
    [CONFIG]:C:/TDI/rs.xml
        [AL]:*
[ENDUSER]
```

この一連の ID レジストリー項目は、以下の制約を表わしています。

- このレジストリー・ファイルでは、ユーザー Stan がアドミニストレーターであることが指定され、各サーバー API 操作をすべて実行することが許可されています。
- John は、サーバーにロードされているすべての構成の読み取りを許可されていますが、プロセスの実行については 2 つの構成からの実行のみが許可されています。
  - `"rs.xml"` からは、John はすべての AssemblyLines を実行することが許可されています。
  - `"prototype.xml"` からは、John は「TestAssemblyLine」という名前の AssemblyLine を実行することのみが許可されています。
- Peter は、`"rs.xml"` 構成からすべての AssemblyLine を実行することのみが許可されています。

注: トラストストア・ファイルからユーザー ID を取得するには、**keytool** ユーティリティーや **Ikeyman** ユーティリティーを利用できます。次のコマンド行を使用すると、トラストストア・ファイルからすべてのユーザーが印刷されます。

```
keytool -v -list -keystore <trust_store_file> -storepass <trust_store_pass>
```

ここで、<trust_store_file> はすべてのトラステッド・ユーザーの証明書を含む鍵ストア・ファイルで、<trust_store_pass> はこの鍵ストア・ファイルのパスワードです。このコマンド行を使用すると、各ユーザー証明書について、例えば次のようなテキストが印刷されます。

```
Owner: CN=Stan, OU=TDI, O=IBM, C=US
Issuer: CN=Stan, OU=TDI, O=IBM, C=US
Serial number: 408f6a34
Valid from: 4/28/04 11:24 AM until: 7/27/04 11:24 AM
Certificate fingerprints:
         MD5:  F6:EF:81:8B:4C:0F:10:E4:A0:16:99:AB:42:29:70:8B
         SHA1: FE:37:62:8B:42:2F:54:F8:F6:F3:FC:A1:DD:7D:2A:51:9A:85:09:02
```

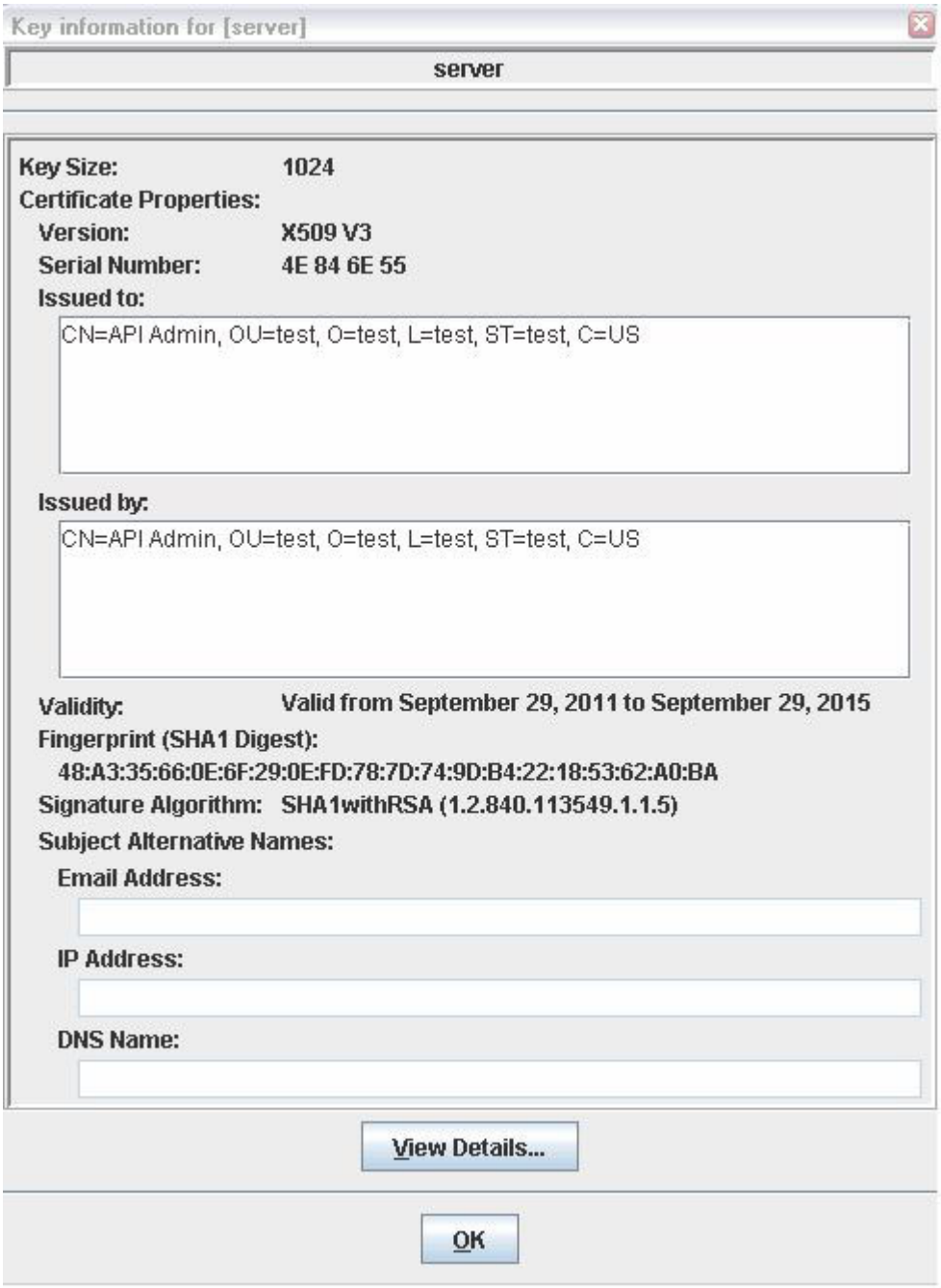
ID レジストリーの [ID]: タグの値には、**Owner** フィールドの値を空白やコンマもすべて含めてそのまま指定する**必要があります**。この例の場合であれば、ID タグの行は次のように記述します。

```
[ID]:CN=Stan, OU=TDI, O=IBM, C=US
```

トラストストア・ファイルからユーザー ID を取得する代替手段として、以下のようにして Ikeyman を使用します。

1. Ikeyman を始動します (または、ツールバーから **Key Manager** を選択します)。

2. 「**鍵データベース・ファイル (Key Database File)**」メニューから「**オープン...**」をクリックします。
3. 「**オープン**」フィールドで、適切な値を設定し、「**OK**」をクリックします。
4. 「**パスワード**」フィールドで、トラストストア・ファイルのパスワードを入力します。
5. 目的の証明書をクリックします。
6. 「**表示/編集... (View/Edit...)**」ボタンをクリックします。これにより、対象の DN に関する情報 (ユーザーID) を含むウィンドウが開きます。



サーバー監査機能

IBM Security Directory Integrator イベントを監査することができます。通知はイベントごとに作成されます。以下の情報を参照することで、監査機能について詳しく知ることができます。

IBM Security Directory Integrator 監査コンポーネントを使用すると、IBM Security Directory Integrator サーバーでサーバー API の認証や許可などのイベントを監査できるようになります。

認証や許可 (auth*) のイベントが発生すると、通知が生成されます。監査データは項目にパッケージ化され、通知のユーザー・データとして提供されます。「監査サービス」は、IBM Security Directory Integrator サーバーによって自動的にロードされる、独立した監査構成で構成されます。監査構成には、自動開始される Audit AssemblyLine が含まれます。監査 AL は、適切なフィルターを使用して、通知コネクター上で繰り返されます。IBM Security Directory Integrator ユーザーは、独自のコード内から監査イベントを作成する場合に、「ユーザー定義の通知」を生成することもできます。

IBM Security Directory Integrator 監査は 2 つの主要な部分を含んでいます。

- 必要な監査情報を生成する方法
- 既存の監査データを処理する「監査サービス」

必要な監査情報の生成は、サーバー API 内の各監査ポイントで IBM Security Directory Integrator 項目を作成し、これらの項目を通知内にラップしてブロードキャストすることによって行われます。この目的のため、サーバー API には新しいクラス (com.ibm.di.api.APIAuditor) が用意されました。これは、項目を生成し、その項目を UserData として通知に添付し、それを関心のあるリスナーすべてに送信します。

「監査サービス」は監査通知の主たる利用者です。監査サービスは、通知コネクター上で繰り返される複数の AL から成る構成です。さまざまなフィルターの使用が、各種の通知タイプに登録できます。

監査の適用範囲

監査対象と見なすことができるのは、リストされている基準を満たすイベントのみです。

IBM Security Directory Integrator 監査機能は、一般的に、ユーザーが実行することが対象であり、サーバー・イベントは対象ではありません。ユーザーがタスク (AL を停止) の実行を許可されるということは、タスク (AL が終了) が実際に実行されるということとは異なります。許可されたということは、許可イベントであり、AssemblyLine の停止のような正当なアクションを実行することは、サーバー・イベントです。ユーザーが AL に対して停止を指示し、AL が終了すると、許可イベントがサーバー・イベントとペアになって生成されます。また別の時点で、AL が自動的に完了したような場合は、サーバー・イベントが単独で発生します。直接的なユーザーとの相互作用を含むイベントのみ、監査の対象です。これによって、デフォルトの監査ポイントがサーバーAPI 内の認証および許可イベントに制限されます。サーバー API から公開されるほとんどすべてのメソッドは、独自の許可コードで自らを保護しています。監査コンポーネントはすべての許可イベントの通知を送

信することを試みるのではなく、許可が保護されているサーバー API メソッドの合理性のあるサブセットのみを選出します。選出基準は、以下の条件を備えたイベントをすべて監査することです。

- ログまたはトゥームストーンの削除
- 構成、AL、サーバーのような IBM Security Directory Integrator エンティティーの始動または停止
- 構成インスタンスの構成の交換: 構成インスタンスの構成またはチェックイン構成の交換
- ユーザーに対して重要な IBM Security Directory Integrator データの変更を許可: 外部プロパティーの設定、システム・キューへのメッセージの通知、IBM Security Directory Integrator JVM 内でのカスタム Java コードの呼び出し

通知の抑止

以下の情報を参照することで、通知の抑止について、抑止されたイベント・タイプを生成するコマンド、および抑止されたイベント・タイプを生成する方法について詳細を理解できます。

IBM Security Directory Integrator サーバー API では、特定の通知タイプを抑止して、パフォーマンスを向上させることができます。通知フレームワークは、抑止されたイベントを伝搬しません。*suppressed* タイプのイベントのブロードキャストを試みても、サーバー API はエラーは発行しません。ただし、抑止されたイベントは、登録された通知リスナーには一切到着しません。抑止されたイベント・タイプのリストは、次の名前を持つシステム・プロパティーで構成されます。

```
api.notification.suppress
```

デフォルトでは、すべての認証および許可イベントが抑止されます。

```
api.notification.suppress=di.server.api.authenticate di.server.api.authorize*
```

リスト内のイベント・タイプはスペースで区切られます。複数のイベント・タイプに一致するワイルドカードは許可されます。イベント・タイプ・プロパティーが欠落しているか、空の場合、抑止されるイベントはありません。次の指定を使用すると、すべてのカスタム通知を抑止できます。

```
api.notification.suppress=user
```

注: 抑止は、IBM Security Directory Integrator サーバー全体に影響するので、すべての通知タイプを抑止してしまう可能性があります。AssemblyLine の開始や、サーバーのシャットダウンなど、組み込み通知であったとしても、抑止される可能性があります。抑止機能を不適切に使用すると、トゥームストーン・マネージャーやサーバー通知コネクターなど、通知を listen しているコンポーネントの動作に干渉する可能性があります。

通知の送信

すべての登録済みリスナーに通知を配信することができます。以下に通知配信パラメーターのリストを示します。

通知の送信には com.ibm.di.api.APIEngine 内のメソッドが使用されます。

```
public static void sendNotification
     (String type, String id, Object data, String configInstanceId)
```

このメソッドは DIEvent を作成します。この方法により、特定のタイプの通知の受信を登録しているすべてのリスナーに対して通知が配信されます。通知配信パラメーターには次のものが含まれます。

表 18. 通知配信パラメーター

パラメーター名	定義
type	通知イベント・タイプ。
id	通知イベント ID。
data	追加情報を持つ Java オブジェクト形式の通知イベント UserData オブジェクト。
configInstanceId	通知がバインドされる通知 ConfigInstance ID。

タイプ・パラメーターが NULL の場合、com.ibm.di.api.APIEngine メソッドは DIException をスローします。次のいずれかの場合、メソッドへの呼び出しが起動されます。

- IBM Security Directory Integrator サーバー JVM からのローカル呼び出し。このタイプのアクセスは、AssemblyLine フックでのスクリプト記述を含みます。また、Java でインプリメントされ、IBM Security Directory Integrator サーバーにデプロイされた新規のコネクターからの API も使用します。
- リモート・メソッド呼び出し (RMI) を使用した、ローカル・コンピューターまたはリモート・ネットワーク・コンピューター上の他の JVM からのリモート呼び出し。このタイプのアクセスでは、以下の方法によるソリューションが使用されます。
  - IBM Security Directory Integrator へのリモート接続
  - IBM Security Directory Integrator 内のプロセスの管理
  - IBM Security Directory Integrator の上位層にビジネス・ロジックを構築
  - IBM Security Directory Integrator 専用のアプリケーション
  - 目標達成のため IBM Security Directory Integrator を使用するアプリケーション

IBM Security Directory Integrator サーバー・インスタンス・セキュリティー

暗号化アルゴリズムと各種構成ファイルを使用して、サーバー・インスタンスをセットアップできます。詳しくは、以下の情報を参照してください。

このセクションでは、IBM Security Directory Integrator サーバーへの (IBM Security Directory Integrator ベースまたはその他の) クライアント・アクセスの詳細については説明しません (121 ページの『リモート・サーバー API』で説明します)。代わりに、使用される暗号化アルゴリズムや、サーバー・インスタンスのセットアップに必要な各種構成ファイルを中心に説明します。

IBM Security Directory Integrator サーバーは、秘密鍵および関連付けられた証明書/公開鍵の両方が含まれる鍵ストアを必要とします。この鍵ストアは、構成ファイル、プロパティー・ファイル内のプロパティー、サーバー・ユーザー・レジストリー・ファイル、およびその他のオブジェクトを PKI 暗号化するときや、SSL 通信を行うときに使用します。

システム・プロパティー *api.keystore* および *api.key.alias* により、鍵ストアおよび鍵ストア内のサーバーの証明書/鍵の鍵別名を指定します。鍵ストアのパスワード、および鍵自体のパスワード (鍵ストアのパスワードと異なる場合) は、サーバーの stash ファイル内で指定します。鍵ストアへのアクセスは、鍵ストアの作成時に鍵ストアの作成者によって定義されるパスワードによって保護され、これは、現行パスワードの提供時にのみ変更可能です。また、鍵ストア内の各秘密鍵は、それ自体のパスワードによって保護できます。サーバーの stash ファイルについての詳細は、『Stash ファイル』を参照してください。

RSA アルゴリズムは、ファイルとプロパティー値の暗号化で使用されます。これは、ブロック・サイズが RSA 鍵のモジュラス・コンポーネントで決定されるブロック暗号として使用されます。暗号化は ECB (電子コードブック) モードで行われます。PKCS#1 埋め込みは、各ブロックごとに個別に適用されます。ファイルの暗号化で使用される RSA 鍵ペアと同じ RSA 鍵ペアは、サーバーとの SSL 通信にも使用されることに注意してください。IBM Security Directory Integrator は、IBMJCE セキュリティー・プロバイダーの RSA インプリメンテーションを使用します。そのプロバイダーでサポートされるすべての鍵サイズは、IBM Security Directory Integrator でもサポートされます。IBM Security Directory Integrator v7.0 から、暗号化では共通鍵暗号も採用されるようになりました。RSA は以前のバージョンとの互換性からデフォルトとして使用されているわけですが、共通鍵暗号は公開鍵暗号に比べると、はるかに高速で、はるかにセキュアです。

DES および AES のアルゴリズムは、パスワードで保護された構成ファイルの暗号化に使用されます。暗号鍵 (DES または AES) は、パスワードの UTF-8 バイナリー表現から導出されます。導出された暗号鍵は、DES の場合は 64 ビット、AES の場合は 128 ビットです。ECB モードは、埋め込みなしで使用されます。

DES/AES 鍵は、パスワードで保護された構成ファイルが使用されたとき、パスワードから導出されます。上記以外の場合に、IBM Security Directory Integrator が鍵を生成するケースはありません。既存の鍵は、外部の鍵ストアから読み込まれます。鍵の設定と鍵ストア・アクセスは、IBMJCE および IBMJSSE2 セキュリティー・プロバイダーを介して行われます。これらのプロバイダーでサポートされるすべての鍵サイズおよびアルゴリズムは、IBM Security Directory Integrator でもサポートされます。

Stash ファイル

Stash ファイルには鍵ストアのパスワードと鍵自体のパスワードが保存されます。以下の情報を参照して、stash ファイルの操作方法を理解します。

stash ファイルには、サーバーの鍵ストアのパスワード値が、固定鍵による AES128 で暗号化されて含まれます。サーバーの stash ファイル名は idisrv.sth (名前は構成できません) で、このファイルはソリューション・フォルダーからサーバーによってロードされます。stash ファイルを作成するためのコマンド行ユーティリティー *createstash.bat* または *createstash.sh* を、IBM Security Directory Integrator の bin フォルダーから利用できます。

```
createstash <keyStorePassword> [<keyPassword>] [<securityProviderClass>]]
```

ここで、*keyStorePassword* は *api.keystore* システム・プロパティーによって指定された鍵ストア・ファイルのパスワード、*<keyPassword>* は *api.key.alias* システム・プロパティーによって指定されたサーバーの秘密鍵のパスワードです。

keyPassword は、<securityProviderClass> パラメーターを指定しない場合のオプションのパラメーターです。<keyPassword> パラメーターを指定しない場合は、サーバーの秘密鍵パスワードは鍵ストアのパスワードと同じであると仮定されます。<securityProviderClass> パラメーターを指定してユーティリティーを使用するには、上記の 2 つのパラメーター (keyStorePassword および keyPassword) を両方とも指定する必要があります。セキュリティー・プロバイダーを指定すると、暗号化ではこのプロバイダーが使用されます。

このユーティリティーにより、「idisrv.sth」という名前の stash ファイルが、指定されたパスワード (1 つ以上) で現行ディレクトリーに作成されます。

重要: IBM Security Directory Integrator には、サンプル stash ファイルが組み込まれています。このファイルのパスワードは「server」です。セキュリティーを強化するため、前述のユーティリティーを使用して各自の stash ファイルを生成することを強くお勧めします。また、実際に stash ファイルを必要とする IBM Security Directory Integrator サーバー以外からはアクセスできないようにしておく必要があります。

サーバーのセキュリティー・モード

IBM Security Directory Integrator サーバーは、**標準**および**セキュア**という 2 つのモードで実行できます。詳しくは、以下の情報を参照してください。

.

標準モード

標準モードで実行されているサーバーは、PKI 暗号化の呼び出しを要求する特定のサーバー API 呼び出しの場合を除き、ディスクに保管される構成を PKI で暗号化しません。標準モードのサーバーは、暗号化された構成と暗号化されていない構成の両方を読み取ることができます。

セキュア・モード

セキュア・モードで実行されているサーバーは、ディスクに保管される構成をすべて PKI 暗号化を使用して暗号化します。サーバーは、セキュア・モードの場合、暗号化された構成のみ読み込みおよびロードできます。システム・プロパティー *com.ibm.di.server.securemode* を「true」に設定すると、サーバーはセキュア・モードで実行されます。(IBM Security Directory Integrator サーバーを使用するためのシステム・プロパティーは、これを `global.properties` または `solution.properties` ファイルに追加して設定するか、または IBM Security Directory Integrator サーバーの始動時に Java コマンド行で直接指定することができます。*-Dcom.ibm.di.server.securemode=true*)

サーバーの始動時に Java コマンド行でコマンド行オプション -e を指定すると、*com.ibm.di.server.securemode* システム・プロパティーの値がどのように設定されていても、サーバーはセキュア・モードで実行されます。

注: IBM Security Directory Integrator 6.0 より前のパスワードに基づく構成ファイルの暗号化も、以前のバージョンとの互換性のためにサポートされています。パスワードに基づく暗号化は、構成の作成時にユーザーがパスワードを指定した場合に使用されます。IBM Security Directory Integrator 6.0 より前のパスワードに基づく構成の暗号化を、PKI 暗号化と組み合わせて使用することはできません。サーバーがセキュア・モードで実行中にパスワードを指定すると、エラー・メッセージが表示されます。

暗号化 IBM Security Directory Integrator 構成ファイルの処理

構成ファイルで暗号変換を実行することができます。以下の情報を参照することで、暗号化ファイルの使用法と注意点について詳しく知ることができます。

データの機密性を保つため、IBM Security Directory Integrator では、構成ファイル、プロパティー・ファイル内のプロパティー値、サーバーのユーザー・レジストリー・ファイル、および JavaScript ファイルを暗号化できます。

IBM Security Directory Integrator 暗号化では、鍵または鍵ペアを使用する暗号変換も行われます。鍵/鍵ペアは、鍵ストア・ファイル内でホストされる必要があります。

暗号変換は、公開鍵暗号化か共通鍵暗号化のいずれかです。IBM Security Directory Integrator ではデフォルトでは公開鍵暗号化が使用されます。 (共通鍵暗号化は、IBM Security Directory Integrator 7.0 で導入されました。それより前には、公開鍵暗号化しかサポートされていませんでした。)

参照:

公開鍵暗号化では、公開鍵と秘密鍵で構成される鍵ペアを使用します。公開鍵は暗号化で使用され、秘密鍵は復号で使用されます。現在公開鍵暗号化でサポートされているのは RSA 暗号のみです。公開鍵/秘密鍵ペアは、標準の JRE ユーティリティー `keytool` および `Ikeyman` を使用して、生成と管理を行います。関連付けられた公開鍵/秘密鍵を持つ証明書の管理について詳細は、105 ページの『鍵、証明書、および鍵ストアの管理』を参照してください。

次のシステム・プロパティーを使用すると、IBM Security Directory Integrator データ暗号化を構成できます (これらは `global.properties` または `solution.properties` に設定できます)。

- *com.ibm.di.server.encryption.keystore* : 暗号化用の鍵/鍵ペアを含む鍵ストア・ファイル
- *com.ibm.di.server.encryption.keystoretype* : 鍵ストア・ファイルのタイプ
- *com.ibm.di.server.encryption.key.alias* : 鍵ストア内の鍵/鍵ペアの別名
- *com.ibm.di.server.encryption.transformation* : 暗号変換の名前; 以下の注釈を参照

鍵ストアのパスワード、および鍵自体のパスワード (鍵ストアのパスワードと異なる場合) は、サーバーの 150 ページの『Stash ファイル』内で指定します。(鍵ストアへのアクセスは、鍵ストアの作成時に鍵ストアの作成者によって定義されるパスワードによって保護され、これは、現行パスワードの提供時にのみ変更可能です。また、鍵ストア内の各秘密鍵は、それ自体のパスワードによって保護できます。)

変換の名前は、RSA または特定の共通鍵変換 (例えば、AES/CBC/PKCS5Padding) のいずれかです。変換名についての詳細は、http://www.ibm.com/developerworks/java/jdk/security/60/secguides/JceDocs/api_users_guide.html#trans を参照してください。IBM Security Directory Integrator で使用される Java Security についての概要は、http://www-128.ibm.com/developerworks/java/jdk/security/60/secguides/jsse2Docs/JSSE2RefGuide.html を参照してください。

注:

1. 「com.ibm.di.server.encryption.*」プロパティーは、構成の暗号化に影響するだけでなく、プロパティー・ファイル、JavaScript ファイル、およびサーバー API ユーザー・レジストリーの暗号化にも影響します。
2. 暗号化鍵または暗号変換 (あるいはその両方) を変更すると、サーバーは以前暗号化したファイルを復号できなくなります。この問題の回避策としては、古いファイルを古い鍵 (そのために古い鍵は保管しておく必要があります) を使用して復号し、新しい鍵で再び暗号化することです。ファイルの暗号化と復号は、cryptoutils ツールを使用して行うことができます。
3. 標準の RSA アルゴリズムには、操作できるデータの長さに制限があります。IBM Security Directory Integrator はカスタム・スキームを使用して、入力データを十分に小さな同じサイズのブロックに分割し、各ブロックを独立に暗号化します。
4. RSA で暗号化されるデータは、異なる暗号化を実行するたびに異なる暗号テキストになります。この効果は、RSA で使用される PKCS#1 埋め込みスキームの特徴です。
5. 共通鍵 (対称) 暗号は、ブロック暗号またはストリーム暗号のいずれかを指定できます。ストリーム暗号は、メッセージのビットを一度に 1 つ暗号化します。ブロック暗号は複数のビットを取り込んで、それを単一ユニット (ブロック) に暗号化します。ブロック暗号 (例えば、AES) はフィードバック・モード (プレーン・テキスト内のパターンが暗号テキストで保持されないようにするため) と埋め込みスキーム (暗号のブロック・サイズの倍数ではない長さのデータの暗号化を可能にするため) を使用します。ストリーム暗号 (例えば、RC4) は、フィードバック・モードや埋め込みスキームは使用しません。
6. 変換にブロック暗号が含まれる場合は、何らかの埋め込みスキーム (例えば、「PKCS5Padding」) を使用する必要があります。そうしないと、サーバーが暗号のブロック・サイズの倍数ではない長さのデータを暗号化できません。(ストリーム暗号では埋め込みを使用しないため、このような制限はありません。)
7. 鍵/鍵ペアのアルゴリズムは、指定した変換のアルゴリズムと一致している必要があります。例えば、変換が RSA ならば、RSA 鍵ペアを用意する必要があります。変換が DES/ECB/PKCS5Padding ならば、DES 鍵を用意する必要があります。keytool ユーティリティーを使用すると新規に共通鍵を作成できます。105 ページの『鍵、証明書、および鍵ストアの管理』を参照してください。
8. JKS 鍵ストアは共通鍵をサポートしません。したがって、共通鍵暗号化を使用する場合は、JCEKS のような他の鍵ストア・タイプを使用する必要があります。
9. ブロック暗号を初期化ベクトル (IV) が必要なフィードバック・モードで使用すると、暗号化されたデータには初期化ベクトルをプレーン・テキストにした接頭部が付加されます。IV を秘密にしておく必要はありませんが、予測不能にし

ておく必要はあります。これが暗号化されるデータの各断片ごとに、ランダムな IV が生成される理由です。ランダムなデータの生成は、時としてリソースを消費します。したがって、パフォーマンスが問題になる場合は、非 IV フィードバック・モード (ECB) を使用することを考えてください。

10. 暗号化のサポート対象となる共通鍵変換は、Java セキュリティー・プロバイダーの機能に依存します。デフォルトでは、IBM Security Directory Integrator は IBMJCE プロバイダーを使用します。サポートされるブロック暗号は、DES、AES、DESede (Triple DES)、Blowfish、および RC2 です。これらの暗号は、ECB、 CBC、 CFB、 OFB、 PCBC のいずれのフィードバック・モードでも使用できます。唯一使用可能な埋め込みスキームは「PKCS5Padding」です。MARS ブロック暗号は、埋め込みをサポートしないため、暗号化では使用できません (http://www-128.ibm.com/developerworks/java/jdk/security/50/secguides/JceDocs/api/com/ibm/crypto/provider/Mars.html)。サポートされるストリーム暗号は RC4 と ARCFOUR です (名前は異なりますが、基本的には同じ暗号です)。SEAL ストリーム暗号には大きな鍵 (160 ビット) が必要なため、使用する場合は、IBM Security Directory Integrator JRE に制限のない IBM SDK ポリシー (http://www.ibm.com/developerworks/java/jdk/security/60/#sdkpol) を構成する必要があります。

PKI 暗号化および SSL のための証明書の分離

暗号化された IBM Security Directory Integrator 構成ファイルを最初から作成する

cryptoutils コマンド行ツールを使用して、暗号化された IBM Security Directory Integrator 構成ファイルを最初から作成することができます。

ここでは暗号化された IBM Security Directory Integrator 構成ファイルを最初から作成する方法について説明します。

cryptoutils コマンド行ツールの使用

1. 構成エディターを使用して、通常の暗号化されていない IBM Security Directory Integrator 構成ファイルを作成します。
2. 156 ページの『IBM Security Directory Integrator 暗号化ユーティリティー』セクションで説明されているように、`cryptoutils` コマンド行ツールを使用してこの構成ファイルを暗号化します。
3. この暗号化された構成ファイルを実行するには、『サーバーのセキュリティー・モード』セクションで説明されているように、IBM Security Directory Integrator サーバーをセキュア・モードで開始する必要があります。
4. この暗号化された構成ファイルを編集するには、『暗号化 IBM Security Directory Integrator 構成ファイルの編集』セクションで説明されている 2 つの方法のうちのいずれかを使用できます。

暗号化 IBM Security Directory Integrator 構成ファイルの編集

ここで説明する手順を使用して、暗号化ファイルを編集することができます。

まず、156 ページの『IBM Security Directory Integrator 暗号化ユーティリティー』セクションで説明されているように、`cryptoutils` コマンド行ツールを使用して、

暗号化された構成ファイルを復号します。次に、構成エディターを使用して復号された構成ファイルを編集し、最後に cryptoutils ツールを使用して変更済みの構成ファイルを再度暗号化します。

global.properties または solution.properties の標準 暗号化

以下の手順に従って、global.properties ファイルまたは solution.properties ファイルを暗号化します。

global.properties ファイルおよび solution.properties ファイルには多くのプロパティーが保管されており、その一部はパスワードなどの機密データを表す可能性があります。この機密データを保護するために、IBM Security Directory Integrator ではこのデータの暗号化が可能です。

名前の接頭部に *{protect}-* を付けたプロパティーはすべて、サーバーの公開鍵を使用してサーバーによって PKI 暗号化されます。サーバーの鍵は、*api.keystore* プロパティーで指定した鍵ストアから *com.ibm.di.server.encryption.key.alias* プロパティーによって指定します。例えば、プロパティー *com.ibm.di.server.encryption.keystore* を暗号化する場合は、global.properties または solution.properties ファイルに次の行を追加します。

```
{protect}-com.ibm.di.any.property=some_value
```

次回にサーバーを実行すると、このプロパティーの暗号化が必要であることをサーバーが検出し、即時にファイルを上書きして、プレーン・テキストの値「some_value」を暗号化した形式で書き込みます。

注: 特定のオペレーティング・システム (Linux/UNIX システムでそのように構成されている場合) では、global.properties ファイルに書き込むためにアクセスできないことがあります。この場合、ファイルが書き換えられていないこと/暗号化されていないことを示す警告メッセージがサーバーによって出力されます。

global.properties または solution.properties 内のプロパティーの保護は、構成エディターの「**サーバー・ストアのブラウズ**」オプションからアクセス可能な「グローバル・プロパティー」および「ソリューション・プロパティー」プロパティー・ストアからも可能です。

外部プロパティー・ファイルのプロパティーの暗号化

外部プロパティー・ファイルのプロパティーを暗号化するには、サーバーの鍵ストアの指定された証明書を使用します。

外部プロパティー・ファイルに保管されるプロパティーは、global.properties または solution.properties 内のプロパティーとまったく同じ方法で暗号化によって保護できます。

サーバーのデフォルトの証明書を使用する代わりに、サーバーの鍵ストア内の証明書を明示的に指定して、外部プロパティー・ファイルのプロパティーを暗号化することもできます。

これらのファイルに保管されるプロパティーの暗号化に関する詳細については、155 ページの『global.properties または solution.properties の標準 暗号化』セクションを参照してください。外部プロパティー・ファイル内のプロパティーの構文は次のとおりです。

```
[{protect}-]keyword <colon | equals> [{encr}][{java}]value
```

- オプションの接頭部 *{protect}-* は、値が暗号化されているか、または暗号化する必要があるかのいずれかを示します。値が文字シーケンス {encr} で始まる場合は、その値が既に暗号化されていることを意味します。
- オプションの値の接頭部 *{java}* は、その値が直列化された Java オブジェクトであることを示します。値は b64 エンコードする必要があります。例:

```
{protect}-api.truststore.pass
     ={encr}J8AKimpEutu3BblOVg55F/5d5vO2kXWcNUWnCq3vINUc6K0719z9dEk3H43Ot2iTT1dZTI6FSSVin9KsCy
     BLmgv+n84w7HelKl3ro2dFmZbTYKMXuxGoqN9nL2VOvZoptNqzoWvs6IN/p3VkIIBtlao/9mEPEKuIwRnKtkQ89Bg=
```

IBM Security Directory Integrator 暗号化ユーティリティー

cryptoutils ユーティリティーを使用して、ファイルを編集することができます。その際には、以下にリストされているパラメーターに注意してください。

TDI_install_dir/serverapi ディレクトリーには、cryptoutils ユーティリティーが格納されていますが、これを使用すると、ファイル (例えば、ID レジストリー・ファイル) を復号化および再暗号化して、ファイルを手動で編集できるようになります。

このツールでは、以下のコマンド行パラメーターを使用できます。

input {必須} 暗号化または復号の対象となるファイルを指定します。

output {必須} 暗号化または復号が完了した後、結果のデータで作成される新規ファイルを指定します。このファイルは、存在した場合は上書きされます。

mode {必須} ツールの実行時のモードを指定します。以下のモードのいずれかです。

- *encrypt*: ユーザー・レジストリーを暗号化
- *decrypt*: ユーザー・レジストリーを復号
- *encrypt_config*: IBM Security Directory Integrator 暗号化ユーティリティーの構成ファイルまたは JavaScript ファイルを暗号化
- *decrypt_config*: IBM Security Directory Integrator 暗号化ユーティリティーの構成ファイルまたは JavaScript ファイルを復号
- *encrypt_props*: IBM Security Directory Integrator 暗号化ユーティリティーのプロパティー・ファイル内にある保護されたすべてのプロパティーの値の暗号化
- *decrypt_props*: IBM Security Directory Integrator 暗号化ユーティリティーのプロパティー・ファイル内にある保護されたすべてのプロパティーの値の復号

注: ユーザー・レジストリー・ファイルは、構成ファイルおよび JavaScript ファイルとは異なった方法で暗号化されます。

keystore
{必須} 暗号化/復号のための鍵を含む鍵ストア・ファイルを指定します。

storepass

{必須} 鍵ストア・ファイルのパスワードを指定します。

alias {必須} 鍵ストア内の暗号化/復号のための鍵の別名を指定します。

keypass

{オプション} 暗号化/復号のための鍵のパスワードを指定します。デフォルトでは、鍵ストア・パスワードが鍵へのアクセスのために使用されます。

transformation

{オプション} 暗号化/復号で使用される暗号変換の名前を指定します。RSA または任意の共通鍵変換 (例えば、AES/CBC/PKCS5Padding) です。デフォルトは RSA です。

storetype

{オプション} 鍵ストア・ファイルのタイプ (例えば、JKS) を指定します。このパラメーターはケース・インセンシティブ (JCEKS と jceks は同じ) です。このパラメーターを指定しない場合は、デフォルトの JRE の鍵ストア・タイプ (JRE の java.security ファイル内の「keystore.type」セキュリティー・プロパティーで構成) が使用されます。

cryptoproviderclass

{オプション} 暗号化/復号で使用される (鍵ストア・アクセスでは使用されない) Java セキュリティー・プロバイダーを指定します。デフォルトでは、JRE のセキュリティー・プロバイダー・リスト (java.security JRE ファイル内に構成されています) 内のプロバイダーが使用されます。

例:

ユーザー・レジストリーを暗号化する

セキュア・モードで稼働する IBM Security Directory Integrator サーバーでは、ユーザー・レジストリーはサーバー鍵で暗号化されていることが必要です。

プレーン・テキストのユーザー・レジストリー・ファイルは次のコマンドを使用すれば暗号化することができます。

```
cryptoutils -input registry.txt -output registry.enc -mode encrypt
     -keystore ../testserver.jks -storepass server -alias server
```

IBM Security Directory Integrator 構成を復号する

```
cryptoutils -input myconfig.enc.xml -output myconfig.xml -mode decrypt_config -keystore ../testserver.jks
 -storepass server -alias server
```

このコマンドは、「myconfig.enc.xml」構成ファイル (通常は、セキュア・モードで稼働する IBM Security Directory Integrator サーバーで作成されたもの) を復号します。復号した構成ファイル「myconfig.xml」は、構成エディターで簡単に変更できます。構成を変更後、再び暗号化して、セキュア・モードの IBM Security Directory Integrator サーバーが読み取ったり、使用することができるようにします。

(デフォルトの「RSA」ではなく) 対称型暗号を使用して IBM Security Directory Integrator 構成を暗号化する

```
cryptoutils -input myconfig.xml -output myconfig.enc.xml -mode encrypt_config -keystore ../server.jck
 -storepass server -alias server -transformation AES/CBC/PKCS5Padding -storetype jceks
```

上記のコマンドは、鍵ストア「server.jck」が存在することを前提にしています。また、その鍵ストアには、「server」という別名を持つ AES 共通鍵が含まれていると仮定しています。

global.properties ファイルを復号する

IBM Security Directory Integrator サーバーは、global.properties または solution.properties ファイルを読み取ったとき、保護されたプロパティーの値は自動的に暗号化します。

global.properties ファイル内の暗号化されたすべての値は、次のようなコマンドで復号できます。

```
cryptoutils -input ../etc/global.properties -output ../etc/global.properties -mode decrypt_props
 -keystore ../testserver.jks -storepass server -alias server
```

注: cryptoutils ツールを使用して 143 ページの『サーバー API ユーザー・レジストリー』、構成ファイル (サーバーが暗号化された構成を処理する方法は、151 ページの『サーバーのセキュリティー・モード』を参照) または 162 ページの『IBM Security Directory Integrator サーバー・フックの暗号化』 の暗号化や復号を行うと、暗号化や復号の対象はファイル全体になります。

一方、プロパティー・ファイルの暗号化/復号モードでは、暗号化/復号されるのは保護されたプロパティーの値のみであり、ファイル全体ではありません。したがって、.properties ファイルを *encrypt_props* モードを使用して暗号化した後でも、ファイル内のプロパティー鍵やコメントは相変わらず誰でも読むことができます。保護されたプロパティーについての詳細は、155 ページの『global.properties または solution.properties の標準 暗号化』と 155 ページの『外部プロパティー・ファイルのプロパティーの暗号化』を参照してください。

IBM Security Directory Integrator システム・ストアのセキュリティー

ユーザーおよびパスワードのリポジトリーを定義する場合は、Derby を使用します。以下のリストを参照して、プロパティーを適切な値に設定してください。また、以下の手順に従って、Derby で提供されるユーザー承認メカニズムを使用することもできます。

IBM Security Directory Integrator システム・ストアは、IBM Security Directory Integrator サーバーで必要となるすべての情報が保持されているデータベースまたはパーシスタント・レイヤーです。従来、このレイヤーにはセキュリティーが配慮されていませんでした。どのユーザーでもシステム・ストアにアクセスできました。しかし IBM Security Directory Integrator 7.0 からシステム・ストアにセキュリティーを構成することができるようになりました。

IBM Security Directory Integrator 7.0 では、デフォルトでは、システム・ストアはネットワーク・モードで使用されます。そのため、複数の IBM Security Directory Integrator インスタンスやその他のアプリケーションが、システム・ストアに並行してアクセスできます。 システム・ストアをネットワーク経由でどこからでもアクセスできるようにしたため、今度は IBM Security Directory Integrator サーバーが維持管理しているデータを保護するために、セキュリティーに配慮することが必要になりました。

Derby (以前は、Cloudscape として知られていました) には、ユーザーとパスワードのリポジトリーを定義する方法が複数用意されています。 Derby システムでこれらのサービスのうちどれを使用するか指定するには、以下のセクションの説明に従って、プロパティー derby.authentication.provider に適切な値を指定します。

外部ディレクトリー・サービス

ディレクトリー・サービスは、名前と名前の属性を保管します。 Derby は Java naming and directory interface (JNDI) を使用して、ユーザー名とパスワードの認証を行う外部ディレクトリー・サービスと対話します。

Derby を使用すると、企業内の既存の LDAP ディレクトリー・サービスのユーザーを認証できます。 LDAP (lightweight directory access protocol) には、TCP/IP 上で動作するオープン・ディレクトリー・アクセス・プロトコルが備わっています。LDAP ディレクトリー・サービスは、ユーザー名とパスワードを短時間で認証します。

Derby で定義された一連のプロパティーを構成することで、ユーザー名とパスワードのリポジトリーとして外部ディレクトリー・サービスを利用することを開始できます。

ユーザー定義クラス

ユーザー定義クラスのアプローチを使用すると、Derby を LDAP 以外の他の外部認証サービスにフックすることができます。

derby.authentication.provider に、パブリック・インターフェース org.apache.derby.authentication.UserAuthenticator をインプリメントするクラスの絶対パス名を設定します。特定の最低限の要件を満たす独自のクラスを作成すれば、Derby を外部認証サービスにフックすることができます。

外部認証サービスを提供するクラスは、パブリック・インターフェース org.apache.derby.authentication.UserAuthenticator をインプリメントし、必要に応じて、タイプ java.sql.SQLException の例外をスローする必要があります。

組み込み Derby ユーザー

Derby には、ユーザー名とパスワードを保管するための単純なリポジトリーが用意されています。この組み込みリポジトリーを使用するには、プロパティー derby.authentication.provider=BUILTIN を設定する必要があります。

IBM Security Directory Integrator システム・ストアは、ユーザー名とパスワードの保管に組み込みリポジトリーを使用しています。IBM Security Directory Integrator ではシステム・ストアにアクセスするユーザーは 1 人しかいないので、使用可能なプロバイダーの中で最も適しているプロバイダーです。

ユーザー認証

このセクションでは、ユーザーの認証について説明します。ユーザー認証メカニズムには、上述のリポジトリー (上述のリポジトリーのどれでも構いません) にユーザー名が存在し、指定されたユーザーのパスワードが正しかった場合の認証を行う機能しかありません。ただし、Derby に用意されているユーザー承認メカニズムでは、必要に応じて、アクセス権についてより細かい制御を行うことができます。

用意されたパラメーターに対してユーザーの認証を要求するためのマスター・スイッチは、プロパティー derby.connection.requireAuthentication です。そのデフォルトは *TRUE* です。

アクセス・モードは、プロパティー derby.database.defaultConnectionMode=fullaccess を使用して設定できます。このプロパティーは Derby リポジトリー内のすべてのユーザーのデフォルトのアクセス・モードを設定します。このプロパティーはシステム・ストア・ユーザーのアクセス・レベルの定義も行います。Derby でサポートされるアクセス・レベルは、*fullAccess*、*readOnly*、および *noAccess* です。ただし、特定のユーザーに対して特定のアクセス・レベルを付与することが必要な場合は、以下にリストしたプロパティーを使用してアクセス権を割り当てることができます。

- derby.database.fullAccessUsers=<usernames>。ユーザーに全アクセス権限を与えます。
- derby.database.readOnlyAccessUsers=<usernames>。ユーザーに読み取り専用アクセス権を与えます。
- derby.database.noAccessUsers=<usernames>。ユーザーにデータベースへのアクセスを許可しません。

usernames は、次のように、ユーザーのコンマ区切りのリストです。

```
derby.database.fullAccessUsers=sa, mary
```

現在のバージョンの IBM Security Directory Integrator では、システム・ストアにアクセスできるユーザーは 1 人のみです。このユーザーは、システム・ストアに対するすべての操作を実行する必要があります。このため、アクセス・モードは *fullAccess* に設定されています。

各種設定構成ファイルの機能

以下の情報を参照して、各種設定構成ファイルの機能を詳細に理解することができます。

「パスワード」構成パラメーター・タイプ

構成における IBM Security Directory Integrator コンポーネントの構成パラメーターには、「ストリング」、「数値」、「ブール値」などがあります。使用可能なタイプの 1 つに「パスワード」があります。構成パラメーターのタイプがパスワードの場合、新規パスワードの入力時、および編集/実行のための既存の構成のオープン時の両方において、構成エディターではその値がコンポーネントの構成ウィンドウに文字「*」のシーケンスとして表示されます。

コンポーネントのパスワード保護

ここに示されている手順を使用して、デフォルト・プロパティー・ストアにコンポーネントのパスワードを定義することができます。

IBM Security Directory Integrator では、すべての構成値の平文を含む XML ファイルに構成情報が保存されます。この情報には、パスワードなどの機密情報も含まれ

ます。IBM Security Directory Integrator では、構成ファイル全体の暗号化がサポートされますが、構成ファイルが平文として保存された場合、機密情報は暗号化または保護されません。

IBM Security Directory Integrator には各種コンポーネントで必要になるパスワードを保護する優れたメカニズムが用意されています。このメカニズムは、平文の構成のパスワードを隠蔽し、保管されているパスワードにデフォルトのセキュリティーを提供します。これを実現するために、コンポーネント・パスワードは、構成ファイルではなく、デフォルトのプロパティー・ストアで定義 (保管、および検索) します。IBM Security Directory Integrator では、コネクターがある任意のシステムをプロパティー・ストアとしてユーザーが定義することができ、デフォルトのプロパティー・ストアは外部のプロパティー・ファイルである可能性が最も高くなっています。コンポーネントのすべてのパスワードは、デフォルトでは、以前のバージョンの製品のように構成ファイル内ではなく、このデフォルトのプロパティー・ストアで保管されます。このように、ユーザーが明示的に指定変更 (開発の初期には適切である場合があります) しなければ、パスワードを構成ファイルと分離することができます。

構成済みプロパティーへのパスワードの保管

パスワード・ストアを使用してパスワードを保存するには、ここで説明する手順に従います。

このタスクについて

パスワード保護メカニズムは、ユーザーに提示される構成ウィンドウと直接関連しています。構成ウィンドウまたはフォームには、各パラメーターおよびその構文の説明が含まれます。 構文の 1 つのタイプにパスワード があり、これにより構成エディターで編集するためのパスワード・テキスト・フィールドが使用されます。パスワード構文コンポーネント・パラメーターの値が変更されるたびに、パスワードの値がパスワード・ストア と呼ばれる外部リポジトリーに保管されます。このパスワードの外部リポジトリーは、構成エディターの「*プロパティー*」ページ (パスワード・ストア) で構成し、現行 IBM Security Directory Integrator ソリューションの構成ファイルで指定します。このようなプロパティー・ストアが構成されていない場合、パスワードは構成ファイルの平文内に保存されます。

デフォルトのパスワード・ストアが構成されている場合、protected/password パラメーターの初回保存時に固有のプロパティー名が生成されます。この鍵は、パスワード・ストアにおける鍵として使用されます。構成ファイルには、同じプロパティー名が標準プロパティー参照として書き込まれます。後で値を検索する際に、標準プロパティー解決が行われ、パスワード・ストアから実際の値が検索されます。

パスワード・ストアが指定されている場合、パスワードの固有キーが生成され、パスワードは、パスワード・ストアにそのキーの下に暗号化されて保存されます。構成ファイルでは、パスワードはそのキーによってのみ参照されます。

パスワード・ストアが指定されていない場合、パスワードは構成ファイルのプレーン・テキストに表示されます。

例を示します。

1. 構成エディターで新規のプロジェクトを作成します。

2. ナビゲーション・ビューの「プロパティー」フォルダーを右クリックして、「新規プロパティー・ストア」を選択し、「MyProps」という名前を付けます。
3. 新たに作成されたプロパティー・ストアの「コネクター」タブから、「コレクション・パス/URL」フィールドに「MyProps.properties」と入力します。
4. 新規のプロパティー・ストアをパスワード・プロパティー・ストアとして使用することを指示します (ナビゲーション・ビューの「新規プロパティー・ストア」を右クリックして、「**パスワード・プロパティー・ストア**」を選択します)。
5. FTP クライアント・コネクターで新規 AssemblyLine を追加します。
6. FTP クライアント・コネクターの「ログイン・パスワード」フィールドにパスワードを入力します。
7. ソリューションを保管し、構成エディターをクローズします。

上記の手順の後、作成されたソリューションの構成ファイルには次のようなテキストを持つ行が格納されます。

```
<parameter name="ftpPass">@SUBSTITUTE{property.MyProps:ftpPass-38ae53e8779cfd65}</parameter>
 .........
<PasswordStore>MyProps</PasswordStore>
```

... そして、「MyProperties.properties」ファイルには、次のようなテキストが格納されます。

```
{protect}-ftpPass-38ae53e8779cfd65={encr}GVJC0lA7VUiW=
```

これは、ソリューション・ファイル内の FTP パスワード構成は、現行のパスワード・ストア「MyProps」から暗号化されたプロパティーを参照することを意味します。使用されるプロパティー鍵は「ftpPass-38ae53e8779cfd65」です。

トレース中に平文での属性のプリントを保護

ここに示されている方法を使用して、トレース中に機密データを保護します。

IBM Security Directory Integrator Solution Builder には、パスワードなどの機密データを、ソリューションのトレースが必要な場合に平文に印刷されることから保護する方法が必要です。このため、 IBM Security Directory Integrator では、属性が保護されているかどうかを確認できるように、属性クラスを処理する一部のメソッドが機能拡張されました。属性が保護されているものとしてマークされ、トレースがオンになっている場合、一定の個数の「*」が実際の値の代わりに出力されます。

TaskCallBlock (TCB) 内に接続パラメーターがある場合、IBM Security Directory Integrator によって値が直接ログに記録されることはありません。パラメーターが提供されたという事実は記録されますが、値自体は記録されません。ソリューションをデバッグする必要がある場合、これらの値は、例えばスクリプトを使用して手動でダンプすることができます。

IBM Security Directory Integrator サーバー・フックの暗号化

機密データは、サーバー・フック・ディレクトリーに追加する前に暗号化する必要があります。詳しくは、以下の情報を参照してください。

サーバー・フック・スクリプトは、ソリューション・ディレクトリーの「serverhooks」サブディレクトリーにファイルを作成することにより定義し、使用可能になります。機密情報を含むスクリプトは、ディレクトリーに追加する前にサ

ーバー API で暗号化する必要があります。スクリプトは cryptoutils を使用して暗号化できます (156 ページの『IBM Security Directory Integrator 暗号化ユーティリティー』を参照)。 IBM Security Directory Integrator サーバーが復号するのは、「.jse」ファイル名拡張子が付いているスクリプト・ファイルのみであることに注意してください。「.jse」拡張子は、IBM Security Directory Integrator サーバーに対してそのスクリプト・ファイルが暗号化されていることを示します。そのため、サーバー・フック・スクリプト・ファイルは暗号化した後、ファイル名拡張子を「.jse」に変更する必要があります。

リモート CE と SSL

リモート構成エディターと SSL を使用して作業を行う場合は、以下の点に注意してください。

リモート構成ファイル (リモート・システムの構成ファイル) を編集するための構成エディターは、リモート構成エディター (リモート CE) と呼ばれます。IBM Security Directory Integrator リモート CE は、編集のためにオープンされた構成内で AssemblyLine を開始できます。リモート構成エディターは、リモート IBM Security Directory Integrator サーバーのサーバー API のクライアントです。したがって、リモート構成エディターは、サーバー API のクライアントとして認証および許可されます。 SSL の使用時にこれが機能するようにするには、以下のようにします。

1. リモート構成エディターが接続するサーバーを、SSL クライアント認証が必要となるように構成する必要があります。これはサーバー API の構成です。詳細については、 129 ページの『SSL ベースの認証』を参照してください。
2. リモート構成エディター IBM Security Directory Integrator インスタンスを、SSL クライアント認証を提供するよう構成する必要があります。これは 116 ページの『SSL クライアント認証』で構成されます。

リモート構成エディターは、AssemblyLine が終了したときに通知を受けることができるようにリスナー・オブジェクトを使用し、このことが SSL の使用時に機能するようにクライアントがサーバーの ID を信頼し、サーバーがクライアントの ID を信頼する必要があるため、この SSL クライアント認証が必要になります。

リモート構成エディターの使用

リモート構成エディターの使用に関する制限に対処できます。

リモート構成エディターの使用法は、ローカルの構成エディターと多少異なります。リモート構成エディターでリモート・システムの構成を管理する場合は、リモート・モードの構成エディターに適用される制約事項に注意してください。主な制約事項を以下に示します。

- 構成ファイルをローカルで編集する場合は、構成ファイルへのファイル・システム・アクセス権限 (読み取りまたは書き込み) で十分です。ただし、リモート構成を編集する場合は、リモート構成インスタンスに対する管理権限が必要です。
- データ・ソースに接続する (マッピング・ウィンドウ内の「**接続**」ボタンを使用する) とき、これらの接続はローカルに評価されます。

例えば、ldap://localhost:389 を実行すると、構成エディター (CE) は、リモート・コンピューター上の LDAP サーバーではなく、ローカルの LDAP サーバーへの接続を試みます。

- WebServices 関連のコネクターの生成で、WSDL ファイル、.jar ファイル (Complex Type Generator を使用) などを生成する関数コンポーネントを生成した場合、それはローカルの生成です。これらのコンポーネントは、CE の接続先のリモート・システム上に生成されるわけではありません。したがって、リモート・システムには手動でアップロードして、デプロイする必要があります。
- リモート構成エディターでは、`api.config.folder` プロパティーで指定されているフォルダー内の構成の編集と表示のみが可能です。
- CE で使用可能な**システム・ストア**の操作 (イテレーター状態キーの削除など) を行う場合は、リモートの IBM Security Directory Integrator コンピューターのシステム・ストアではなく、ローカルのシステム・ストアを操作することになります。AssemblyLine (AL) が実行される場合にのみ、AL がリモート・システム・ストアに接続されます。これは、AL の実行時には、AL がリモート JVM 内部で実行されるためです。
- **パラメーター置換**エディターを使用する (Ctrl-E で使用可能) と、エディターはローカル・プロパティーのみを表示し、リモート・システムに設定されているプロパティーは表示しません。同様に、新規プロパティー・ストア (ファイル・タイプ) を作成して保管する場合、作成されたプロパティー・ストア (ファイル) はローカルに保管されます。
- 構成エディターを使用してリモート構成ファイルを編集しようとすると、CE はクライアントとして動作しているため、サーバー API の認証と許可が必要になります。したがって、CE をこのような方法で使用するには、リモート・サーバーの *admin* アクセス権を持っている必要があります。
- リモート・サーバーを使用する場合は、構成ファイルが格納されているローカル・ファイル・システムに対してリモート・サーバー自体が十分なアクセス権を持っている必要があります。ConfigfFiles が、読み取り専用ファイル・システムに格納されている場合、または、リモート・サーバーを実行しているユーザーID が書き込みアクセス権を持たないファイル・ストレージ・ロケーションに格納されている場合は、リモート構成ファイルは編集できません。

セキュリティーを扱う構成ファイルおよびプロパティーの要約

セキュリティーを扱う構成ファイルおよびプロパティーの要約リストを参照できます。

表 19. 上記で説明した構成ファイルと各ファイルの内容を以下の表に示します。

構成ファイル	ロケーション	説明
global.properties	*TDI_home*/etc	このファイルは、サーバーの 1 次構成ファイルです。
solution.properties	ソリューション・フォルダー	このファイル (`solution.properties`) は最初は現在のソリューションで使用されている `global.properties` のコピーです。変更すると、このファイルの値によって `global.properties` 内の対応する値がオーバーライドされます。

表 19. 上記で説明した構成ファイルと各ファイルの内容を以下の表に示します。 (続き)

構成ファイル	ロケーション	説明
registry.txt	*TDI_home*/serverapi	このファイルは、global.properties の「api.user.registry」プロパティーで定義されているサーバー API のユーザー・レジストリーです。
build.properties	*TDI_home*/etc	このファイルには、IBM Security Directory Integrator のビルド情報、ビルドした日付、バージョンなどが格納されます。テキスト・ファイルであり、デフォルトでは、プラットフォームでネイティブのエンコード方式が使用されます。
tdisrvctl-log-4j.properties	*TDI_home*/etc	このファイルは、tdisrvctl コマンド行ユーティリティーのロギング戦略を制御します。
Log4J.properties	*TDI_home*/etc	このファイルは、コマンド行から始動した場合、サーバー (ibmdisrv) のロギング戦略を制御します。
jlog.properties	*TDI_home*/etc	このファイルは、トレースおよび First Failure Data Capture (FFDC) 戦略を制御します。
ibmdi.ico	*TDI_home*/etc	このファイルには、IBM Security Directory Integrator のアイコンがリストされます。
idisrv.sth	*TDI_home*	このファイルには、IBM Security Directory Integrator サーバー stash が含まれます。サンプルのサーバー鍵ストア・ファイル (`testserver.jks`) の暗号化されたパスワードを含むバイナリー・ファイルです。
derby.properties	*TDI_home*/etc	このファイルには、IBM Security Directory Integrator に付属する Derby システム・ストアのデフォルトの構成が含まれます。
reconnect.rules	*TDI_home*/etc	このファイルには、IBM Security Directory Integrator による再接続例外の処理について再接続規則を定義しているテキストが含まれます。
global.properties.v611	*TDI_home*/etc	このファイルは、サンプルのプレースホルダーの役割を果たします。マイグレーション時に役に立ちます。
TDI0701.SYS2	*TDI_home*/etc	これは、ITLM エージェントが IBM Security Directory Integrator を認識するために使用する製品シグニチャー (ライセンス) ファイルです。
pkcs11.cfg	*TDI_home*/etc	このファイルは、IBM PKCS11 インプリメンテーション・プロバイダーの初期化で使用されます。詳細については、『PKCS11 構成ファイル』のセクションを参照してください。
testadmin.der	*TDI_home*/serverapi	このファイルは `testadmin.jks` からエクスポートされた証明書です。
testadmin.jks	*TDI_home*/serverapi	このファイルには、サーバー API リモート・クライアントのサンプルの鍵ストアとトラストストアが含まれます。
cryptoutils.bat(sh)	*TDI_home*/serverapi	このファイルは、IBM Security Directory Integrator の構成ファイルおよびユーザー・レジストリー・ファイルの暗号化と復号で使用されるコマンド行ユーティリティー (シェル・スクリプト) です。

表 19. 上記で説明した構成ファイルと各ファイルの内容を以下の表に示します。 (続き)

構成ファイル	ロケーション	説明
testserver.jks	*TDI_home*	このファイルは、例として参照しているサンプルのサーバーの鍵ストアとトラストストアです。
testserver.der	*TDI_home*	このファイルはエクスポートされたサンプルのサーバー証明書です。そのままトラストストアにインポートすることができます。
am_config.properties	*TDI_home*/ActionManager	このファイルは、Action Manager の構成で使用します。
am_logging.properties	*TDI_home*/ActionManager	このファイルは、Action Manager のロギングの構成で使用します。
ibmdiservice.props	*TDI_home*/win32_service	このファイルは、Windows サービスの構成で使用します。
mqeconfig.props	*TDI_home*/jars/plugins/	このファイルを使用すると IBM WebSphere MQ サービスを構成できます。IBM Security Directory Integrator では、IBM WebSphere MQ Everyplace の認証を使用すれば、IBM WebSphere MQ Everyplace にアクセスして証明書を発行できます。この証明書はその後の認証で使用されます。認証時には、IBM Security Directory Integrator から、`mqeconfig.props` プロパティー・ファイルに追加する必要がある追加プロパティーが提供されます。

注: `registry.txt` ファイルは、 156 ページの『IBM Security Directory Integrator 暗号化ユーティリティー』 を使用して、暗号化および復号を行うことができます。cryptoutil ツールは `global.properties` または `solution.properties` に適用できません。個々のプロパティー値は暗号化できますが、プロパティー・ファイル全体の暗号化はできないからです。

表 20. 上記で参照されるプロパティー、その特性、機能、考えられる値、用途を示す表。

名前	考えられる値	説明
com.ibm.di.server.securemode	true/false	セキュア・モードのオンまたはオフを切り替えるスイッチ。
api.keystore	ファイル名	SSL 証明書で使用されるサーバー鍵ストアです。従来は `com.ibm.di.server.keystore` でした。
api.key.alias	鍵の別名	SSL 証明書の鍵ストアの鍵の別名です。従来は `com.ibm.di.server.key.alias` でした。
{protect}-api.keystore.password	SSL 鍵ストアのパスワード	SSL 用の鍵ストアのパスワード。IBM Security Directory Integrator 7.0 で追加されました。
{protect}-api.key.password	SSL 鍵のパスワード	SSL 用の鍵のパスワード。IBM Security Directory Integrator 7.0 で追加されました。

表 20. 上記で参照されるプロパティー、その特性、機能、考えられる値、用途を示す表。 (続き)

名前	考えられる値	説明
com.ibm.di.server.encryption.keystore	ファイル名	サーバーで使用される鍵をホストする鍵ストアのデータ暗号化。IBM Security Directory Integrator 7.0 で追加されました。
com.ibm.di.server.encryption.key.alias	鍵の別名	暗号化鍵ストアの鍵の別名。IBM Security Directory Integrator 7.0 で追加されました。
com.ibm.di.server.encryption.keystoretype	鍵ストア・タイプ、すなわち、「JKS」、「JCEKS」など。	サーバーの暗号化鍵をホストする鍵ストア・タイプ。IBM Security Directory Integrator 7.0 で追加されました。
com.ibm.di.server.encryption.transformation	「RSA」またはその他の共通鍵変換	暗号化で使用されるサーバー変換。「RSA」(公開鍵暗号化) または特定の共通鍵変換のいずれかを設定できます。IBM Security Directory Integrator 7.0 で追加されました。
api.on	true/false	サーバー API のオンまたはオフを切り替えるスイッチ。
api.user.registry	ファイル名	サーバー API のユーザー・レジストリー・ファイル
api.user.registry.encryption.on	true/false	ユーザー・レジストリーを暗号化するかしないかのスイッチ。
api.remote.on	true/false	リモート・サーバー API のオンまたはオフを切り替えるスイッチ。デフォルトの設定値は true です。
api.remote.ssl.on	true/false	リモート・サーバー API で SSL が必要か必要ではないかを切り替えるスイッチ。
api.remote.ssl.client.auth.on	true/false	リモート・サーバー API で SSL クライアント認証が必要か必要ではないかを切り替えるスイッチ。
api.truststore	ファイル名	サーバーのトラストストア。
api.truststore.pass	*	トラストストアのパスワード。
api.remote.nonssl.hosts		非 SSL IP 接続を受け付けるための非 SSL アドレス。
api.custom.method.invoke.on	true/false	カスタム・メソッド起動用のサーバー API メソッド。 =true は使用を許可、=false は不許可。
api.custom.method.invoke.allowed.classes		`custommethod` 起動のため、サーバー API メソッドから直接起動できるサーバー API クラス。

表 20. 上記で参照されるプロパティー、その特性、機能、考えられる値、用途を示す表。 (続き)

名前	考えられる値	説明
api.custom.authentication	組み込み LDAP または JAAS 認証のためのスクリプト・ファイル名または「[ldap]/[jaas]」	カスタム認証メソッド。
api.custom.authentication.ldap.*		プロパティーの LDAP 認証構成セット。
javax.net.ssl.*		鍵ストア、トラストストア、およびそれらのパスワードのプロパティーの標準 JSSE のセット
com.ibm.di.server.pkcs11	false	pkcs11 準拠の SSL 用暗号デバイスが必要か必要ではないか。IBM Security Directory Integrator 7.0 で追加されました。
{protect}-com.ibm.di.server.pkcs11.pass	administrator	pkcs11 準拠の暗号デバイスのアクセス・パスワード。IBM Security Directory Integrator 7.0 で追加されました。
com.ibm.di.server.pkcs11.accl	false	このプロパティーに true を設定すると、暗号化にハードウェア暗号デバイスが使用されます。

注: 上記の表にリストしたプロパティーはすべて構成ファイル `global.properties` に設定できます。また接頭部 {protect}- を使用して暗号化で保護できます (詳細については、 155 ページの『global.properties または solution.properties の標準 暗号化』を参照)。

Web 管理コンソールのセキュリティー

以下のリンクを参照して、Web 管理コンソールのセキュリティーの詳細について理解します。

287 ページの『AMC および Action Manager のセキュリティー』を参照してください。

セキュリティーのその他の側面

ここにリストされているさまざまなセキュリティーの側面を参照できます。

HTTP 基本認証 (BA)

HTTP 基本認証 (BA) を使用して、コンポーネントを認証できます。リストされているパラメーターを必ず確認してください。

一部の IBM Security Directory Integrator コンポーネントでは、認証メカニズムとして HTTP 基本認証 (BA) を使用できます。この名前が示すように、これは基本の (単純な) 認証です。HTTP 基本認証は、証明書が他者が容易にデコードできる Base64 エンコードであるため、「セキュアであること」の特に厳密な定義に照らしてセキュアであると認識するべきではありません。データを保護するにはより複雑

な方式を使用する必要があります (例えば、SSL を有効にして HTTP 基本認証と組み合わせる)。コンポーネントがHTTP 基本認証をサポートする場合は、以下のパラメーターが表示されます。

authenticationMethod

HTTP 認証のタイプを指定します。HTTP 認証のタイプが「無名」に設定されている場合は、認証は実行されません。「HTTP 基本認証 (BA)」が指定されている場合は、「ユーザー名」と「パスワード」パラメーターに指定されているユーザー名とパスワードを使用して HTTP 基本認証が実行されます。

IBM Domino SSL の特性

ここに示されている IBM Domino SSL の仕様を参照できます。

SSL 用の IBM Domino API は JSSE を使用せず、IBM Domino 固有です。これは、IBM Security Directory Integrator トラストストアおよび鍵ストア (セクション 115 ページの『IBM Security Directory Integrator コンポーネントのクライアント SSL の構成』を参照) が、IBM Domino 変更検出コネクターの SSL 構成にまったく関係しないことを意味します。IBM Domino 変更検出コネクターの SSL 構成には、TrustedCerts.class ファイルを使用します。このファイルは、(IBM Domino Server で) DIIOP プロセスが開始されるたびに生成され、IBM Security Directory Integrator のクラスパスに置く必要があります (つまり、IBM Security Directory Integrator サーバーおよび IBM Security Directory Integrator 構成エディターをそれぞれ開始する *ibmdisrv* または *ibmditk* シェル・スクリプト)。 CLASSPATH に含まれるローカル・パスに TrustedCerts.class をコピーするか、クラスパス内に IBM Domino インストール済み環境の Lotus¥Domino¥Data¥Domino¥Java を含める必要があります。 `global.properties` (または `solution.properties`) で IBM Security Directory Integrator トラストストアまたは鍵ストアが設定されているかいないかは、このコネクターに関係ありません。

注: 上記の内容は、IBM Notes コネクターおよび IBM Domino 変更検出コネクターの SSL の構成に関連しています (これらは SSL over IIOP を使用するため)。

IBM Security Directory Integrator Web サービス・スイートの証明書

ここに示されている指示と例を使用して、IBM Security Directory Integrator Web サービス・スイートの証明書の名前を指定することができます。

IBM Security Directory Integrator Web サービス・サーバー・コネクターで使用する証明書の識別名 (dn) の cn= 部分は、IBM Security Directory Integrator が実行されているホスト・コンピューターの DNS 名または IP アドレスと一致する必要があります。そうでない場合、クライアントが IBM Security Directory Integrator Web サービス・サーバー・コネクターに対する SSL 接続を確立できないため、例外がスローされます。証明書の識別名の cn= の部分の例を以下に示します。`cn=www.myserver.com`。(サーバーの証明書におけるこの識別名に関する制約は、HTTPS プロトコルが原因です。rfc2818「HTTP over TLS」を参照してください)。

注: IBM Security Directory Integrator がクライアントおよびサーバーの両方の証明書を使用する必要がある場合は、`global.properties` または `solution.properties`

で構成されるデフォルトの証明書のみが使用され、これは同じ証明書である必要があります。代替手段として、 SSLSocket または SSLServerSocket Java クラスのカスタム・インプリメンテーションを記述し、デフォルトとは異なる証明書が使用されるようにします。

サーバー証明書の作成例

以下のコマンド行では、「MyServerKeyStore.jks」という名前の鍵ストアに自己署名サーバー証明書が作成されます。

```
keytool -alias MyServerCertAlias -keyalg RSA -genkey -dname cn=<server_ip_address>
 -validity 365 -keystore MyServerKeyStore.jks -storepass mystorepass -keypass mykeypass
```

作成される証明書の別名は、「MyServerCertAlias」です。鍵ペアを作成するために RSA アルゴリズムに使用されます。証明書の識別名はサーバーの IP です。証明書は 365 日 (1 年) 間有効です。鍵ストアのパスワードは「mystorepass」です。作成される秘密鍵のパスワードは「mykeypass」です。作成される証明書は、`global.properties` または `solution.properties` ファイルで以下のプロパティーを設定することにより、構成して使用することができます。

```
api.key.alias=MyServerCertAlias
api.keystore=MyServerKeyStore.jks
```

Mini-Certificate を使用した IBM WebSphere MQ Everyplace 認証

以下の手順を参照して、Mini-Certificate を使用した IBM WebSphere® MQ Everyplace® 認証について詳しく知ることができます。

IBM Security Directory Integrator の IBM WebSphere MQ Everyplace コンポーネントをデプロイして、IBM WebSphere MQ Everyplace Mini-Certificate 認証済みアクセスを利用できます。これらの IBM WebSphere MQ Everyplace 機能を利用するには、IBM WebSphere MQ Everyplace 2.0.1.7 および IBM WebSphere MQ Everyplace Server Support ES06 をダウンロードおよびインストールする必要があります。証明書の認証済みアクセスを使用することにより、匿名の IBM WebSphere MQ Everyplace クライアント・キュー・マネージャーまたはアプリケーションによって IBM WebSphere MQ Everyplace パスワード・ストア・コネクターに対してパスワード変更要求が実行依頼されることを防止できます。

Mini-Certificates を使用した IBM WebSphere MQ Everyplace 認証の構成について詳しくは、「*Password Synchronization Plug-ins*」の『Authenticated IBM WebSphere MQ Everyplace Access』を参照してください。

第 7 章 再接続ルール・エンジン

以下の情報を参照することで、再接続ルール・エンジンについて理解を深めることができます。

概要

IBM Security Directory Integrator サーバーは、コネクターのライフサイクルの間に発生する特定のエラー状況に適用される再接続ルールをサポートしています。サーバーでは、ターゲット・システムと通信中に発生する条件に応じて、ルールで規定されている対策を実施します。

AssemblyLine は、コネクターが例外を発生させるたびに再接続ルール・エンジンをポーリングし、エンジンから、現在の状況に対して推奨される対策アクションを受け取ります。AssemblyLine のコードは、提案されたアクションを実施します。試みることができるアクションには次のものがあります。

- 再接続
- 例外を未処置のまま放置し、エラー・フックのような別のエラー・メカニズムに処置をゆだねる

*再接続*アクションによって再接続が試みられるのは、CE のコネクターの「接続エラー」タブに用意されているオプションを使用して再接続を有効にしている場合のみです。この構成で再接続が有効になっていないと、エラーが起きた場合、再接続ルール・エンジンがどのように判断したとしても再接続は試行されません。

再接続には、基本的に、コネクターの自動リスタートと、それを以前のポジションまで持ってくる (そのように構成されている場合) ことが含まれます。そのためには、コネクターを一度終了させてから、コネクターの初期化を行う必要があります。また、イテレーター・コネクターの場合はオプションで再接続前のポジションに到達するまで項目をスキップする必要があります。再接続を試みるたびに対応する再接続フックが起動されます。フック内のスクリプトは、後続の再接続を成功させるために、最終的に構成を変更することがあります。ユーザーがフェイルオーバーを指定していた場合、再接続の試行が失敗すると自動フェイルオーバーまたは自動フェイルバックが試行されます。

エラー・アクションは、自動再接続が試みられないため、対応するエラー・フックが起動されることを意味します。フックは最終的に、カスタム・リカバリーまたはエラー・レポート作成を行います。

再接続ルール

以下の情報を参照することで、ルールのタイプ、ルールの要素、エラー状況の要素およびネストした例外について理解することができます。

再接続ルール・エンジンは、構成されたルールに基づいて判断を行います。各ルールには、特定のエラー状況が発生したときに実施すべきアクションが記述されています。エンジンは、次の 2 種類のルールを使用します。

- **組み込みルール**: これは、各コネクター・ファイルの tdi.xml ファイルに保管されており、コネクターの jar ファイルにパッケージ化されています。その結果、これらのルールは、必ず、特定のコネクター・クラスに固有であり、すべてのコネクター名に一致します。このルール・リストが、再接続ルール・エンジンが特定のコネクターのエラー状況に対処するときのデフォルトのリストになります。Java で独自のコネクターをプログラミングした場合は、独自の組み込みルールを作成する方法について、IBM Security Directory Integrator の IBM Knowledge Centerの『リファレンス』セクションにある付録『Java での独自コンポーネントのインプリメント』の『コネクターの再接続ルールの定義』セクションを参照してください。
- IBM Security Directory Integrator の旧リリースとの互換性のため、再接続ルール・エンジンをセットアップするときに、エンジンは暗黙のうちに、すべての IOException およびすべての CommunicationException (java.io.IOException および javax.naming.CommunicationException) 時に再接続を試みることを規定している一連のルールを組み込みルールに追加します。
- **ユーザー定義ルール**: etc/reconnect.rules という名前の外部テキスト・ファイルから読み込まれます。このルール・リストは、組み込みルールをオーバーライドします。 173 ページの『ユーザー定義ルールの構成』を参照してください。

各ルールは、特定のコネクターと特定のエラー状況に適用されます。

ルールは以下の要素で構成されます。

- **Connector Class**: ルールが適用されるコネクターの Java クラス
- **Connector Name**: 現在実行されているソリューションの構成ファイルに指定されているコネクター・コンポーネントの名前
- **Exception Class**: ルールが適用される例外の基本クラス
- **Regular Expression**: ルールが適用される例外のメッセージと照合する正規表現
- **Action**: ルールで規定されるアクション。 *error* または *reconnect*。

エラー状況は以下の要素で記述されます。

- **Connector Class**: 例外を発生させたコネクターのクラス
- **Connector Name**: 例外を発生させたコネクターの名前
- **Exception**: コネクターが発生させた例外 - java.lang.Throwable のサブクラス

以下のすべての条件が同時に満たされたとき、ルールがエラー状況に適用されます。

- ルールがエラー状況のコネクターに適用される (ルールに記述されているコネクター・クラスのサブクラスにも一致)
- ルールがエラー状況の原因となったコネクターの名前に適用される
- 例外が、ルールが適用される例外クラスのインスタンス
- ルールに例外メッセージに一致する正規表現が存在しない、または正規表現が例外メッセージに一致する

特定のエラー状況が発生したとき、再接続ルール・エンジンはエラー状況に最も的確なルールを探します。最初、エンジンはユーザー定義ルールを検索し、一致するルールがなかった場合は、組み込みルールを検索します。それでも一致するルール

が見つからなかった場合、エンジンはデフォルトのアクション「error」に従います。ユーザー定義ルール内に一致したルールが見つかった場合、組み込みルールは検索されません。組み込みルール内に更に的確に一致するルールがあるとしてもです。

注: 複数のルールがエラー状況に一致した場合、最も的確なルールが選択されます。的確なルールが複数あり、そのどれもが残りのルールよりも更に的確ではない場合は、リスト内の最初のルールが選択されます。これがルール・リスト内でのルールの順序が重要な理由です。例えば、以下のルールが存在するとします (わかりやすくするため、疑似構文を使用しています)。

```
...exceptionClass = "java.io.IOException", exceptionMessageRegExp = ".*", action = "error"...
...exceptionClass = "java.io.IOException", exceptionMessageRegExp = "¥w*", action = "reconnect"...
```

「problem」というメッセージを伴う java.io.Exception タイプの例外が発生した場合は、両方のルールが一致しても、一方のルールが他方のルールよりも更に的確ではないため、最初のルールが選択されます (正規表現が一致したという実績は、重み付けの目的には反映されません)。

ネストした例外

例外によっては、他の例外の内部にネストしていることがあります。再接続ルール・エンジンがルール・リスト (例えば、組み込みルール) を検索していくとき、エンジンは最初に最上位の例外に一致するルールを検索します。一致するルールが見つからない場合、エンジンは再び同じルール・リストを検索しますが、今回は、ネストしている例外への一致を検索します (最上位の例外がネストしているルールを持たない場合、この検索はスキップします)。再接続ルール・エンジンが照合を試みるのは、第 1 レベルのネストしている例外のみであることに注意してください。それ以上のレベルのネストしている例外があったとしても、それらは無視されます。

注: サーバー・モードのコネクターでは、自動フェイルオーバーは使用不可能です。

ユーザー定義ルールの構成

ルールの定義時にはルールのフォーマットに注意する必要があります。また、ここにリストする重要な情報に注意を払ってください。いくつかの例を参照することもできます。

ユーザー定義ルールのリストは、IBM Security Directory Integrator ソリューション・フォルダー (ソリューション・フォルダーが定義されていない場合は、IBM Security Directory Integrator インストール・フォルダー) の「etc」サブフォルダー内の `reconnect.rules` という名前のテキスト・ファイルに構成されます。各ルールは、1 行に記述されます。ルールの形式は、次のとおりです。

```
<connector_class>:<connector_name>:<exception_class>:<action>:<regular_expression>
```

ここで、

- <connector_class> は、コネクターの Java クラス完全修飾名
- <connector_name> は、AssemblyLine に挿入されたコネクターの名前
- <exception_class> は、例外の Java クラス完全修飾名
- <action> は、「エラー」か「再接続」のいずれかになります。

- <regular_expression> は、http://java.sun.com/j2se/1.5.0/docs/api/java/util/regex/Pattern.html にある `java.util.regex.Pattern` クラスの JavaDoc で説明されている、Java 正規表現です。

注:

1. action 以外の各要素は、空でも構いません。要素が空の場合は「match-all」を意味します。
2. 各要素は必須です。空であっても、囲むコロンは存在する必要があります。(そのため、各行に、少なくとも 4 つのコロンが必要になります。各コロンはルール内で隣接する 2 つの要素を区切ります。「少なくとも 4 つ」と表現したのは、正規表現もコロンを含む可能性があるためです。正規表現のコロンは、ルールの解析に干渉しません。正規表現はルールの最後に評価されるからです。)
3. 冗長なホワイト・スペースは許されません。
4. 正規表現は 4 番目のコロンの直後から始まり、行の最後までを占めます。
5. ユーザー定義ルール・ファイルは、Java プロパティー・ファイルではありません。主な理由は、再接続アクションを除くルールのキーが固有となるためにすべてのルール要素を含む必要があるからです。したがって、Java プロパティー・メカニズムを使用することの唯一の価値は、アクションと他のルール要素を分離することにあります。ただし、これには、ホワイト・スペース、コロン、および等号 (有効なプロパティー・キーにとっての要件) をエスケープする代償が伴います。Java プロパティー・フレームワークが使用されていたとしても、プロパティー・キーからルール要素を抽出するためにキーのカスタム解析は必要です。
6. 各行の最後は、再接続アクションではなく、正規表現です。正規表現内で (ルール要素の区切り文字である) コロンをエスケープしなくても済むように、このパターンは採用されています。
7. 正規表現は、メッセージ・テキスト全体に照合する必要があります。照合するメッセージ・テキストのどこかに「Some Error」という言葉が含まれるとします。適切な正規表現は次のようになります。

   ```
   .*Some Error.*
   ```

   文字「.」は、改行を除くすべての文字に一致します。* 修飾子は、0 文字以上の文字を指定します。ここで、メッセージが改行で終わるとします。この場合、上記の正規表現は一致しません。代わりに、次のような正規表現を試みることができます。

   ```
   .*Some Error.*¥r?¥n?
   ```

   「¥r」と「¥n」はそれぞれリターン文字と改行文字です。? 修飾子は、0 回または 1 回の出現を指定します。
8. それでも、コネクターの構成には reconnect を構成する必要があります。 176 ページの『一般的な再接続の構成』を参照してください。

例

2 つのルールで構成される例

```
com.ibm.di.connector.ReconnectTestConnector:myconnname:java.io.IOException:error:.*¥Wfatal¥W.*
::java.io.IOException:reconnect:
```

JDBC コネクターを使用した再接続

IBM Security Directory Integrator の JDBC コネクターは、DB2 テーブルで反復処理を行うためイテレーター・モードで構成され、再接続機能が使用可能に設定されています。しかし、ソリューションの実行時に、DB2 インスタンスはまだ始動されていません。再接続を機能させるには、`reconnect.rules` ファイルに、次のような例外時のアクションを記述する必要があります。

```
com.ibm.di.connector.JDBCConnector::com.ibm.db2.jdbc.DB2Exception:reconnect:
```

RAC コネクターを使用した再接続

このコネクターは推奨されません。IBM Security Directory Integrator の将来のバージョンでは削除されます。

IBM Security Directory Integrator の RAC コネクターはイテレーター・モードで構成され、再接続機能が使用可能に設定されています。エージェント・コントローラー・サーバーがダウンしている場合、RAC コネクターが再試行 (再接続) を試みるには、`reconnect.rules` ファイルに、次のような例外時のアクションが記述されていることが必要です。

```
com.ibm.di.connector.RACConnector::org.eclipse.tptp.platform.execution.exceptions
     .AgentControllerUnavailableException:reconnect:
```

例外についての考慮事項

例外を処理する場合は、ここで説明する考慮事項を参照できます。

IBM Security Directory Integrator を使用して特定の環境向けに作成された環境およびソリューションは、通常、固有です。ユーザー定義ルールはカスタムビルトであり、ソリューションが環境またはソリューションに固有の例外に基づいて再接続を自動的に試みられるようにこの機能が利用可能になっています。IBM Security Directory Integrator API から各コネクターに返される特定の例外についての詳細は、IBM Security Directory Integrator Java API のマニュアルを参照してください。

また、特定の IBM Security Directory Integrator コンポーネントは基本ライブラリーに依存し、これらのライブラリーの API は特定の状況で例外をスローします。以下に、いくつかの IBM Security Directory Integrator コア・コンポーネントをリストします。そこでは、例外に関する追加情報と例外の原因について説明します。この情報は、遭遇する可能性のある特定の例外についてカスタム再接続ルールの作成を試みるかどうかを判断する際に役立ちます。

- LDAP コネクター - LDAP コネクターは、JRE に付属している JNDI ライブラリーに依存します。JNDI インターフェース、その API、およびそれがスローする可能性のある例外についての詳細は、http://java.sun.com/j2se/1.5.0/docs/api/javax/naming/package-summary.html を参照してください。
- JDBC コネクター - JDBC コネクターは構成されている JDBC ドライバーに依存します。構成されている JDBC ドライバーがスローする可能性のある例外についての詳細は、Java API のマニュアルまたは参考文献を参照してください。「リファレンス」の JDBC コネクターのセクションにある『JDBC ドライバーについて』サブセクションには、一般的に使用される一連の JDBC ドライバーに関する JDBC ドライバーの資料へのリンクが含まれています。

一般的な再接続の構成

再接続ルールについては、ここにリストされている構成オプションを参照してください。

再接続が試みられるためには、再接続ルールを指定することが必要です。しかしそれだけでは不十分です。一般的な再接続構成では、もう 1 つの要件によって、再接続が可能になります。そのためには、構成エディター内の「**接続エラー**」タブを使用します。この構成で再接続が有効になっていない場合は、エラーが発生して再接続ルール・エンジンがどのように判断しようとも、再接続は試みられません。以下に、構成オプションのリストを示します。

再試行回数
問題が発生したときに、再接続を試行する回数です。後でまた新たな問題が発生した場合、また同じ回数の再接続を試行します。

再試行間隔
再接続が試行されてから次にまた試行されるまでの待機秒数、および最初に再接続が試行されるまでの秒数です。

初期接続失敗時に接続を再試行する
このフラグが設定されているときに、コネクターの初期化時に接続を確立できなかった場合は、再接続が試みられます。実際には最初の接続が確立しなかったので「再」接続という表現は適切ではありませんが、メカニズムはほぼ同じです。

接続損失時に自動再接続する
このフラグが設定されているときに、コネクターの初期化が完了してから接続が失われた場合は、再接続が試みられます。

自動前方スキップ
再接続の後に、正常な読み取りの回数と同じ数だけ自動的に前方にスキップします。

自動フェイルオーバー
このフラグを設定すると、自動再接続が失敗した後に自動フェイルオーバーが試みられます。

フェイルオーバー・コネクター
自動フェイルオーバーに使用されるリソース・ライブラリー内にあるコネクターの名前。

フェイルバックまでの時間
このフィールドを正の値に設定すると、指定した秒数の経過後に自動フェイルバックが試みられます。このフェイルバックが失敗した場合、指定した秒数が再び経過するまで自動フェイルバックは再度試みられません。

注: 「**初期接続失敗時に接続を再試行する**」フラグと「**接続損失時に自動再接続する**」フラグの両方について、再接続エンジンは、例外によって再接続を試みるか、または例外がより一般的なエラーなのかを判断します。

第 8 章 システム・キュー (System Queue)

IBM Security Directory Integrator サーバーと AssemblyLines との間でシステム・キューを使用して、メッセージをストア・アンド・フォワードすることができます。

システム・キューは、IBM Security Directory Integrator システム・ストアに類似した IBM Security Directory Integrator JMS メッセージング・サブシステムです。システム・キューにより、複数の AssemblyLine 間で作業を共有するために非同期通信が必要である IBM Security Directory Integrator ソリューションの開発が単純化されます。システム・キューは、その基礎となる JMS メッセージング・システムとして、IBM WebSphere MQ または IBM WebSphere MQ Everyplace のいずれかを使用できます。また、JMS Script Driver を備えたその他の JMS システムは、この JMS システムを適切にアドレス指定できます。

注: システム・キュー・コネクター (リファレンス を参照) はシステム・キューと直接対話せずに、サーバー API を中継として使用します。

IBM Security Directory Integrator の場合、システム・キューはインストール・プロセスによって、デフォルトで使用可能になっています。

システム・キューの構成

以下の情報を使用することで、システム・キューの構成と必要なプロパティーについて詳しく知ることができます。

システム・キューの構成には、IBM Security Directory Integrator `global.properties` または `solution.properties` ファイルで指定されている以下のドライバー固有 Java プロパティーが使用されます。

systemqueue.on

このパラメーターでは、IBM Security Directory Integrator サーバーの始動時にシステム・キューを開始および初期化するかどうかを指定します。有効な値は `true` および `false` です。デフォルト値は `true` です。

systemqueue.jmsdriver.name

このパラメーターでは、システム・キューの JMS ドライバーとして使用する Java クラスの完全修飾名を指定します。この値は、ユーザー提供のクラスの名前、または以下の 5 つの標準 IBM Security Directory Integrator JMS ドライバーのインプリメンテーションのいずれかにできます。

- com.ibm.di.systemqueue.driver.ActiveMQ (178 ページの『Apache ActiveMQ のパラメーター』、default JMS Provider)
- com.ibm.di.systemqueue.driver.IBMMQe (180 ページの『IBM WebSphere MQ Everyplace パラメーター』)
- com.ibm.di.systemqueue.driver.IBMMQ (181 ページの『IBM WebSphere MQ パラメーター』)
- com.ibm.di.systemqueue.driver.IBMMB (181 ページの『Microbroker のパラメーター』)

- com.ibm.di.systemqueue.driver.JMSScriptDriver (182 ページの『JMSScript ドライバーのパラメーター』を使用するその他の JMS システム)

デフォルト値は com.ibm.di.systemqueue.driver.ActiveMQ です。

systemqueue.jmsdriver.name パラメーターに基づいて、以下のいずれかのセクションの内容が適用されます。

Apache ActiveMQ のパラメーター

以下の情報を使用することで、Apache ActiveMQ パラメーターについて詳しく知ることができます。

ActiveMQ をシステム・キューの JMS プロバイダーとして使用するには、`global.properties/solution.properties` の `systemqueue.jmsdriver.name` プロパティーを `com.ibm.di.systemqueue.driver.ActiveMQ` に設定します。 ActiveMQ ドライバーには、次のパラメーターがあります。

- **jms.broker** - ActiveMQ サーバーのアドレス (プロトコル、IP アドレス、および TCP ポート番号)。例えば、

  `tcp://localhost:6161` または `ssl://localhost:616171` (SSL 接続を使用する場合)。

デフォルト値は以下のとおりです。

```
vm://localhost?brokerConfig=xbean:etc/activemq.xml
```

この値は、組み込みモードで ActiveMQ を実行します。 `etc/activemq.xml` ファイルには、デフォルトの ActiveMQ 構成が保持されます。

注:

1. ActiveMQ 構成 XML ファイルへのパス (xbean: の後) には、スペースを含めることはできません。詳しくは、https://issues.apache.org/activemq/browse/AMQ-1385 を参照してください。

   このパスにスペースが含まれる場合、各スペース文字を 3 回 URL エンコードする (したがって %2520 に変換する) 必要があります。
2. **systemqueue.on=true** パラメーターが `solution.properties` ファイル内で true に設定されている場合、システム・キューでは開始時に ActiveMQ を初期化します。

構成

ここで提供する情報を参照して、ActiveMQ 構成および必須パラメーターの詳細について理解できます。

ActiveMQ 構成は、`TDI_install_folder/etc` にある `activemq.xml` ファイルに依存します。 ActiveMQ の構成パラメーターは、以下のとおりです。

broker この ActiveMQ のメッセージ・ブローカーは、トランスポート・コネクター、ネットワーク・コネクター、およびブローカーの構成に使用されるプロパティーから構成されます。属性は以下のとおりです。

- brokerName="localhost" - ブローカーの名前。

- dataDirectory="./ActivemqDataStore" - ActiveMQ のデータの保管に使用されるディレクトリー。
- useShutdownHook="true" - JVM が終了された場合に、ブローカーを閉じるためにシャットダウン・ハンドラーを使用するかどうかを設定します。
- useJmx="true" - ブローカーのサービスを JMX 内で公開するかどうかを設定します。

managementContext

このパラメーターでは、ActiveMQ を JMX 内で公開する方法を構成します。属性は以下のとおりです。

- createConnector="true" - ActiveMQ でそれ独自の JMX コネクターを作成するかどうかを設定します。
- o connectorPort="1099" - コネクターのポート。この値はデフォルトでは 1099 です。

persistenceAdapter/kahaDB

このパラメーターでは、ブローカーのメッセージ・パーシスタンスを構成します。属性は以下のとおりです。

- journalMaxFileLength="32mb" - メッセージ・データ・ログの最大サイズを設定します。
- checksumJournalFiles="true" - 壊れたジャーナルの有無の検査を可能にするために、ジャーナル・ファイルのチェックサムを作成します。
- checkForCorruptJournalFiles="true" - 有効にした場合、始動時に壊れたジャーナル・ファイルがないかを検査し、壊れたジャーナル・ファイルの復旧を試みます。

transportConnectors

このパラメーターは、ActiveMQ で listen するトランスポート・コネクターから構成されます。属性は以下のとおりです。

- name="openwire" - トランスポート・コネクターの名前。
- uri="tcp://localhost:61616" - トランスポート・コネクターのアドレス。

注: XML 構成ファイル内で使用される XML オブジェクトについて詳しくは、 http://activemq.apache.org/xbean-xml-reference-50.html にある ActiveMQ の『XBean XML Reference 5.0』を参照してください。

ロギング

ActiveMQ では、log4j を使用してブローカー・クライアントおよびブローカー内の情報をログに記録します。 log4j.properties 内のリストされている以下の行では、ActiveMQ 項目のデフォルトのロギング・カテゴリーを設定することによって、ActiveMQ のロギングを構成します。

- log4j.logger.org.apache.activemq=INFO
- log4j.logger.org.apache.activemq.spring=WARN
- log4j.logger.org.apache.activemq.web.handler=WARN
- log4j.logger.org.springframework=WARN
- log4j.logger.org.apache.xbean=WARN
- log4j.logger.org.apache.camel=ERROR

ActiveMQ で SSL を使用する

SSL 接続を使用するように ActiveMQ を構成することができます。これは、ActiveMQ の XML 構成ファイル内に <sslContext> エレメントおよび正しい transportConnector の URI を指定することによって行います。

ActiveMQ では、SSL 接続を使用するために証明書に依存します。ActiveMQ は、デフォルトでは TDI_install_folder/serverapi フォルダーにある IBM Security Directory Integrator サーバーの API 証明書を keyStore および trustStore として再利用するように構成されます。再利用を行うためには、ActiveMQ の構成ファイルの <sslContext> エレメント内にクライアントおよびサーバーの鍵ストア・ファイルの名前を指定する必要があります。例:

```
<sslContext>
 <sslContext
  keyStore="file:./serverapi/testadmin.jks" keyStorePassword="administrator"
  trustStore="file:./serverapi/testadmin.jks" trustStorePassword="administrator"/>
</sslContext>
```

ここで、testadmin.jks は IBM Security Directory Integrator 証明書の名前で、password は IBM Security Directory Integrator 証明書のパスワードです。

注: javax.net.ssl プロパティーが IBM Security Directory Integrator solution.properties ファイル内で設定されていない場合に限り、<sslContext> エレメントおよびすべてのパラメーターが考慮されます。 ActiveMQ は、デフォルトでは <sslContext> タグ内に設定されているプロパティーではなく、 IBM Security Directory Integrator API の javax プロパティーを再利用します。

IBM WebSphere MQ Everyplace パラメーター

以下の情報を参照して、IBM WebSphere MQ Everyplace および必須パラメーターについて詳しく知ることができます。

システム・キューの JMS プロバイダーとして IBM WebSphere MQ Everyplace を使用できるようにするには、IBM WebSphere MQ Everyplace キュー マネージャーを作成する必要があります。この操作を行うには、IBM Security Directory Integrator にバンドルされている 185 ページの『IBM WebSphere MQ Everyplace 構成ユーティリティー』 を使用します。

systemqueue.jmsdriver.param.mqe.file.ini

これは、IBM WebSphere MQ Everyplace 初期設定ファイルの相対ファイル・システム・ファイル名を示す IBM WebSphere MQ Everyplace 固有のパラメーターです。このプロパティーは、systemqueue.jmsdriver.name プロパティーで IBM WebSphere MQ Everyplace JMS ドライバーが指定されている場合にのみ必要であり、有効です。デフォルト値は MQePWStore/pwstore_server.ini です。これは、185 ページの『IBM WebSphere MQ Everyplace 構成ユーティリティー』によって作成される IBM WebSphere MQ Everyplace 初期設定ファイルのデフォルト・ロケーションです。

システム・キューは、デフォルトでオンに設定されています。IBM WebSphere MQ Everyplace をシステム・キューとして使用する場合、使用可能にするための手順を要約すると次のようになります。

1. global.properties ファイルまたは solution.properties ファイル内の systemqueue.on プロパティーに *true* を設定します。

2. 次のコマンドを起動し、IBM WebSphere MQ Everyplace を構成します。

   cd *solution_dir* (インストール・ディレクトリーを使用している場合は、
   cd *TDI_install_dir* を使用します)
   TDI_install_dir/jars/plugins/mqeconfig.sh
    TDI_install_dir/jars/plugins/mqeconfig.props create server (1 行に収めます)

IBM WebSphere MQ パラメーター

ここで提供する情報を参照して、IBM WebSphere MQ パラメーターの詳細について理解できます。

これらは IBM WebSphere MQ 固有のパラメーターです。これらのパラメーターの詳細については、184 ページの『システム・キューの構成例』セクションの MQ JMS ドライバー初期化プロパティーを参照してください。

systemqueue.jmsdriver.param.jms.broker
(IP アドレスと TCP ポート番号)

systemqueue.jmsdriver.param.jms.serverChannel
(MQ サーバー・インスタンスに対して定義されているサーバー・チャネル)

systemqueue.jmsdriver.param.jms.qManager
(MQ サーバー・インスタンスに対して定義されているキュー・マネージャーの名前)

systemqueue.jmsdriver.param.jms.sslCipher
(MQ サーバー・チャネルの構成時に選択された暗号に対応する暗号スイート名 (SSL_RSA_WITH_RC4128_MD5 など))

systemqueue.jmsdriver.param.jms.sslUseFlag
(SSL 接続が要求される場合は true、要求されない場合は false)

Microbroker のパラメーター

ここで提供する情報を参照して、Microbroker のパラメーターの詳細について理解できます。

Microbroker (MB) をシステム・キューの JMS プロバイダーとして使用するには、global.properties または solution.properties の systemqueue.jmsdriver.name プロパティーに com.ibm.di.systemqueue.driver.IBMMB を設定する必要があります。

Microbroker ドライバーには、次のパラメーターがあります (ここでは、接頭部「systemqueue.jmsdriver.param.」を外してリストしています)。

jms.broker
MB のサーバー・アドレス (IP アドレスおよび TCP ポート番号): 例えば、「9.126.6.120:1883」

jms.clientID
クライアント ID: 必須

注: システム・キューの JMS プロバイダーとして Microbroker を使用するには、いくつかの Microbroker jar が必要になります。必要な jar のサンプル・リストについては、「リファレンス」の『外部システム構成 - JMS 接続の Microbroker』を参照してください。

JMSScript ドライバーのパラメーター

ここで提供する情報を参照して、JMSScript ドライバーのパラメーターの詳細について理解できます。

JMS ドライバーにより、Java コードを記述してビルドすることなく、JavaScript のスクリプトを使用して JMS プロバイダーへ接続できます。JMS ドライバーは、システム・キューと、ローカル・ファイル・システムにあり、`javax.jms.QueueConnectionFactory` オブジェクトまたは `javax.jms.TopicConnectionFactory` オブジェクトを作成する JavaScript のユーザー指定部分とを結びつける役割を果たします。これらのオブジェクトを取得する方法は、プロパティーによって異なります。

systemqueue.jmsdriver.param.js.jsfile
これは、ユーザーが選択した JMS システムを扱うユーザー提供の JavaScript コードが記述されているファイルの名前を指定する JMS Script Driver 固有のパラメーターです (**systemqueue.jmsdriver.name** が `com.ibm.di.systemqueue.driver.JMSScriptDriver` に設定されている場合に考慮されます)。このパラメーターの詳細については、184 ページの『システム・キューの構成例』セクションの JMS ドライバーの設定を参照してください。 Java プロパティーの名前には、接頭部 `systemqueue.jmsdriver.param` がないことに注意してください。

systemqueue.jmsdriver.param.js.jsscript
対応する JMS プロバイダーとのインターフェースをとる JavaScript コードが含まれているスクリプト本体。このパラメーターを指定しないと、実行する JavaScript のロードには**systemqueue.jmsdriver.param.js.jsfile** パラメーターが使用されます。

systemqueue.jmsdriver.param.user.xxxxx
これらは、システム・キューによって構成済み JMS ドライバーのインプリメンテーションに渡されるユーザー定義プロパティーです。例えば、以下のプロパティーが設定された場合:

```
systemqueue.jmsdriver.param.user.my.prop1=myvalue1
```

構成済みの JMS ドライバーは `user.my.prop1` という名前のプロパティーおよび値 `myvalue1` を取得します。

systemqueue.auth.username
これは、システム・キューが構成済み JMS システムに対する認証に使用するユーザー名です。このパラメーターが設定されていない場合、システム・キューは構成済み JMS システムに対して認証を使用しません。

systemqueue.auth.password
これは、システム・キューが構成済み JMS システムに対する認証に使用するパスワードです。このパラメーターは、`systemqueue.auth.username` パラメーターが指定されている場合にのみ使用されます。

env JavaScript オブジェクト

以下の情報を使用することで、env JavaScript オブジェクトについて詳しく知ることができます。

JMS ドライバーが実行する JavaScript は、JavaScript オブジェクト *env* にアクセスする必要があります。これはタイプ java.util.Hashtable のオブジェクトで、JMS プロバイダーへの接続に関するプロバイダー固有のパラメーターが含まれています。これらのパラメーターは、JavaScript コードが特定の JMS システム・サーバー・インスタンスにアクセスする目的で使用することが意図されています。

これらのパラメーターは、`systemqueue.jmsdriver.param` 接頭部を使用して `global.properties` または `solution.properties` に指定できます。例えば一部の JMS システムで URL パラメーターが必要な場合は、`global.properties` または `solution.properties` に以下のプロパティーを設定します。

```
systemqueue.jmsdriver.param.myjmssystem.url=myjmsserver.mydomain.com:12345
```

この定義により、システム・キューはこれを、鍵が「myjmssystem.url」、値が「myjmsserver.mydomain.com:12345」の env Hashtable のエントリーとして JavaScript コードに渡します。

ret JavaScript オブジェクト

以下の情報を使用することで、ret JavaScript オブジェクトと必須パラメーターについて詳しく知ることができます。

JMS ドライバーが実行する JavaScript は、JavaScript オブジェクト *ret* にアクセスします。これは、タイプ com.ibm.di.systemqueue.driver.JMSScriptDriver.Ret のオブジェクトです。これは、JMS Script Driver クラスの *Ret* 内部クラスのインスタンスです。この *ret* オブジェクトは、JMS Script Driver と、最終的にシステム・キューに対して、JavaScript コードにより JMS システムから取得されたプロバイダー固有のオブジェクトを戻すために使用されます。JMS ドライバーとシステム・キューにエラー情報を返す場合にも、*ret* オブジェクトを使用できます。

ret オブジェクトには、JavaScript から設定できる以下のメンバーがあります。

- `queueConnectionFactory` - タイプ javax.jms.QueueConnectionFactory のオブジェクト。特定の JMS システムから取得された javax.jms.QueueConnectionFactory オブジェクトが保管されます。
- `topicConnectionFactory` - タイプ javax.jms.TopicConnectionFactory のオブジェクト。特定の JMS システムから取得された javax.jms.TopicConnectionFactory オブジェクトが保管されます。
- `errorcode` - タイプ java.lang.Object のオブジェクト。エラー情報オブジェクトを保管する必要がある場所です。このオブジェクトの例として、java.lang.Exception オブジェクトがあります。
- `errordescr` - タイプ java.lang.String のオブジェクト。テキストのエラー記述が保管されます。

Fiorano MQ 用の JavaScript 例

ここで提供される例を使用することで、 Fiorano MQ について詳しく知ることができます。

サード・パーティーの Fiorano MQ システムを使用するための構成と JavaScript コードとのサンプルが *TDI_install_dir*/examples フォルダーに含まれています。以下にこの構成とコードを示します。

```
var ctx = new Packages.java.util.Hashtable();
ctx.put("jms.username",   "anonymous");
ctx.put("jms.password", "anonymous");
ctx.put("jms.broker", "http://192.168.113.220:1856");
ctx.put("jms.qManager",      "fiorano.jms.runtime.naming.FioranoInitialContextFactory");

var ic = new javax.naming.InitialContext(ctx);

var queueFactory = ic.lookup("primaryQCF");
var topicFactory = ic.lookup("primaryTCF");

ret.queueConnectionFactory = queueFactory;
main.logmsg("driverFiorano.js : QueueConnectionFactory : " +  queueFactory);

ret.topicConnectionFactory = topicFactory;
main.logmsg("driverFiorano.js : TopicConnectionFactory : " +  topicFactory);
```

注: この JavaScript は、JavaScript コードにパラメーターをハードコーディングする方法を示しています。もう 1 つの方法として、*env* JavaScript オブジェクトを使用してユーザー指定パラメーターを global.properties または solution.properties から取得する方法があります。*env* オブジェクトを使用してパラメーターを取得する方法では、構成を容易に変更できます。これは、global.properties または solution.properties のプロパティーのみを変更する必要があり、JavaScript コード編集は不要であるためです。つまり、JavaScript のスキルがないユーザーでも、構成を変更できます。

システム・キューの構成例

ここに示されているシステム・キューの構成例を参照できます。

```
##----------------------
## System Queue settings
##----------------------
## If set to "true" the System Queue is initialized on startup and can be used;
## otherwise the System Queue is not initialized and cannot be used.
systemqueue.on=true

## Specifies the fully qualified name of the class that will be used as a JMS Driver.
# systemqueue.jmsdriver.name=com.ibm.di.systemqueue.driver.JMSScriptDriver
# systemqueue.jmsdriver.name=com.ibm.di.systemqueue.driver.IBMMQ
systemqueue.jmsdriver.name=com.ibm.di.systemqueue.driver.ActiveMQ

### MQe JMS driver initialization properties
## Specifies the location of the MQe initialization file.
## This file is used to initialize MQe on TDI server startup.
# systemqueue.jmsdriver.param.mqe.file.ini=MQePWStore/pwstore_server.ini

### MQ JMS driver initialization properties
systemqueue.jmsdriver.param.jms.broker=192.168.113.54:1414
systemqueue.jmsdriver.param.jms.serverChannel=S_s04win
systemqueue.jmsdriver.param.jms.qManager=QM_s04win
systemqueue.jmsdriver.param.jms.sslCipher=SSL_RSA_WITH_RC4128_MD5
systemqueue.jmsdriver.param.jms.sslUseFlag=true

### JMS Javascript driver initialization properties
## Specifies the location of the script file
# systemqueue.jmsdriver.param.js.jsfile=driver.js

### ActiveMQ driver initialization properties
## Specifies the location of the ActiveMQ initialization file.
## This file is used to initialize ActiveMQ on TDI server startup.
systemqueue.jmsdriver.param.jms.broker=vm://localhost?brokerConfig=xbean:etc/activemq.xml

## This is the place to put any JMS provider specific properties needed by a JMS Driver,
## which connects to a 3rd party JMS system.
## All JMS Driver properties should begin with the 'systemqueue.jmsdriver.param.' prefix.
## All properties having this prefix are passes to the JMS Driver on initialization after
## removing the 'systemqueue.jmsdriver.param.' prefix from the property name.
```

```
# systemqueue.jmsdriver.param.user.param1=value1
# systemqueue.jmsdriver.param.user.param2=value2
# ...

## Credentials used for authenticating to the target JMS system
# {protect}-systemqueue.auth.username=<username>
# {protect}-systemqueue.auth.password=<password>
```

セキュリティーと認証

以下の情報を参照することで、セキュリティーと認証およびそれぞれの方式について詳しく知ることができます。

暗号化

標準 JMS ドライバーのうち、SSL がサポートされるのは MQ のドライバーのみです。IBM WebSphere MQ Everyplace JMS ドライバーはローカル・キュー・マネージャーでのみ機能します。これは IBM WebSphere MQ Everyplace アーキテクチャーによって強制されます。JMS Script Driver は、対応するユーザー提供の JavaScript によってサポートされるものをすべてサポートする汎用ドライバーです。

認証

IBM WebSphere MQ などの一部の JMS システムでは、ユーザー名およびパスワードの認証を使用または使用を要求できます。システム・キューでは、`global.properties` または `solution.properties` に、ユーザー名およびパスワードを構成して、システム・キューに提供するために使用できる 2 つの標準プロパティーが提供されています。これらのプロパティーは、 `systemqueue.auth.username` および `systemqueue.auth.password` です。これらの 2 つのプロパティーは、標準 IBM Security Directory Integrator サーバーにより `{protect}-` とマークされたプロパティーが暗号化されることで保護されます。このようにして、これらのプロパティーが設定され、IBM Security Directory Integrator サーバーが開始された後、プロパティーの値が暗号化されます。これらの 2 つのプロパティーに関する詳細については、177 ページの『システム・キューの構成』セクションを参照してください。

IBM WebSphere MQ Everyplace 構成ユーティリティー

ここで提供する情報を参照すると、IBM WebSphere MQ Everyplace 構成ユーティリティーの詳細について理解できます。

デフォルトのシステム・キューとして IBM WebSphere MQ Everyplace を構成して使用するには、IBM WebSphere MQ Everyplace 構成ユーティリティーを使用して新規のソリューション・ディレクトリーに MQ キュー・マネージャーをセットアップします。セットアップされる IBM WebSphere MQ Everyplace キュー・マネージャーには 2 つの定義済みキューがあります。

- **_default** – 汎用キューとして機能します。
- **passwords** – 汎用キュー・パスワードとして機能します。このキューは、パスワード変更の保管用に JMS パスワード・ストア・コンポーネントによって使用されます。これによって、システム・キューがより使用できるようになります。

IBM Security Directory Integrator IBM WebSphere MQ Everyplace 構成ユーティリティー (コマンド行ユーティリティー) は、 IBM WebSphere MQ Everyplace キュー・マネージャーの初回セットアップ時にデフォルトの IBM WebSphere MQ

Everyplace キューを作成します。このデフォルト IBM WebSphere MQ Everyplace キューの名前は「**_default**」です。このデフォルト・キューは、単に便宜上の理由で作成されます。すなわち、ユーザーが、IBM IBM WebSphere MQ Everyplace 構成ユーティリティーを使用して IBM WebSphere MQ Everyplace を (適切な IBM WebSphere MQ Everyplace 構成ユーティリティー・コマンドを使用して) セットアップし、システム・キューおよびシステム・キュー・コネクターの使用をすぐに開始できるようにするためです。

さらに、IBM Security Directory Integrator IBM WebSphere MQ Everyplace 構成ユーティリティーを使用して、システム・キューおよびシステム・キュー・コネクターが使用するユーザー IBM WebSphere MQ Everyplace キューを作成および削除できます。

IBM WebSphere MQ Everyplace 構成ユーティリティーを使用した IBM WebSphere MQ Everyplace キューの作成

以下のコマンド行を入力すると、mqeconfig.props 構成ファイルを使用して、「queue_name」という名前の IBM WebSphere MQ Everyplace キューが作成されます。

```
mqeconfig mqeconfig.props create queue queue_name
```

IBM WebSphere MQ Everyplace 構成ユーティリティーを使用した IBM WebSphere MQ Everyplace キューの削除

以下のコマンド行を入力すると、mqeconfig.props 構成ファイルを使用して、「queue_name」という名前の IBM WebSphere MQ Everyplace キューが削除されます。

```
mqeconfig mqeconfig.props delete queue queue_name
```

ソリューションで特殊な構成が必要な場合、IBM WebSphere MQ Everyplace Explorer を使用して IBM WebSphere MQ Everyplace 構成を微調整することができます。 IBM WebSphere MQ Everyplace Explorer は IBM Security Directory Integrator にバンドルされていませんが、http://www-1.ibm.com/support/docview.wss?rs=0&dc=D400&q1=MQe&q2=MQ+Everyplace&uid=swg24007943&loc=en_US&cs=utf-8&cc=us&lang=en から IBM WebSphere MQ Everyplace Server Support ES06 pack の一部としてダウンロードできます。

キュー・セキュリティーを提供する IBM WebSphere MQ Everyplace メッセージの認証

キュー・セキュリティーを提供する IBM WebSphere MQ Everyplace メッセージを認証することができます。

IBM Security Directory Integrator では、IBM WebSphere MQ Everyplace へのアクセスを、認証に使用される証明書を発行するために IBM WebSphere MQ Everyplace Mini-Certificate サーバーを使用する認証によって保護できます。この目的のために、IBM Security Directory Integrator で使用可能ないくつかの追加プロパティーを、IBM WebSphere MQ Everyplace 構成ユーティリティーの構成プロパティーが含まれる mqeconfig.props プロパティー・ファイルに追加する必要があります。

IBM WebSphere MQ Everyplace Mini-Certificate サーバーによって発行される証明書には、構成可能な有効期間があります。デフォルトの有効期間は 12 カ月です。IBM WebSphere MQ Everyplace の資料では、発行済み証明書は、その有効期間が満

了する前に更新する必要があることが記述されています。これを使用可能にするために、IBM WebSphere MQ Everyplace 構成ユーティリティーには証明書を更新するためのオプションが含まれています。以下のコマンドを入力すると証明書が更新されます。

```
mqeconfig mqeconfig.props renewcert {client | server}
```

1. 最後のコマンド・オプションが「client」の場合、mqeconfig.props ファイルで以下の値を設定する必要があります。
   - **clientRootFolder** - IBM WebSphere MQ Everyplace の構成インスタンスがあるディレクトリー。
   - **certServerReqPin** - この値は、IBM WebSphere MQ Everyplace Mini-Certificate サーバーから証明書の更新を要求した場合の、指定された認証可能なエンティティーについての 1 回限りの認証 PIN として使用されます。
   - **certServerIPAndPort** - この値は、IBM WebSphere MQ Everyplace Mini-Certificate サーバー要求の宛先アドレスとして使用されます。値の形式は「FastNetwork:<host>:<port>」です。ここで、host は、IBM WebSphere MQ Everyplace Mini-Certificate サーバーが実行されているマシン名、TCP IP アドレス、またはホスト名にする必要があります。
   - **certRenewalEntityName** - 証明書の更新を要求する IBM WebSphere MQ Everyplace 認証可能エンティティー。標準的なエンティティー名には以下が含まれますが、IBM WebSphere MQ Everyplace Mini-Certificate で構成されるエンティティー名は、そのエンティティーが「clientRootFolder」の値によって参照されるキュー・マネージャー・レジストリーに実際に存在することを前提として使用できます。
     - PWStoreClient - クライアント・サイド IBM WebSphere MQ Everyplace キュー・マネージャー。
     - PWStoreServer+passwords – クライアント・サイド上のリモート・キュー・プロキシー。
2. 最後のコマンド・オプションが「server」の場合、mqeconfig.props ファイルで以下の値を設定する必要があります。
   - **serverRootFolder** - IBM WebSphere MQ Everyplace の構成インスタンスがあるディレクトリー。
   - **certServerReqPin** - この値は、IBM WebSphere MQ Everyplace Mini-Certificate サーバーから証明書の更新を要求した場合の、指定された認証可能なエンティティーについての 1 回限りの認証 PIN として使用されます。
   - **certServerIPAndPort** - この値は、IBM WebSphere MQ Everyplace Mini-Certificate サーバー要求の宛先アドレスとして使用されます。値の形式は「FastNetwork:<host>:<port>」です。ここで、host は、IBM WebSphere MQ Everyplace Mini-Certificate サーバーが実行されているマシン名、TCP IP アドレス、またはホスト名にする必要があります。
   - **certRenewalEntityName** - 証明書の更新を要求する IBM WebSphere MQ Everyplace 認証可能エンティティー。標準的なエンティティー名には以下が含まれますが、IBM WebSphere MQ Everyplace Mini-Certificate で構成されるエンティティー名は、そのエンティティーが「serverRootFolder」の値によって参照されるキュー・マネージャー・レジストリーに実際に存在することを前提として使用できます。

– PWStoreServer - サーバー・サイド IBM WebSphere MQ Everyplace キュー・マネージャー。
– PWStoreServer+passwords – サーバー・サイド上の実際のキュー。

IBM WebSphere MQ Everyplace キューの構成における DNS 名のサポート

この機能をサポートするために追加のコーディングは不要です。

IBM Security Directory Integrator コンポーネントのインプリメンテーションは、構成プロパティーを単に `mqeconfig.props` から IBM WebSphere MQ Everyplace API に渡すのみであるため、DNS サポートは実際は IBM WebSphere MQ Everyplace の機能であることに注意する必要があります。DNS 名または IP アドレス値を受け取ることができる `mqeconfig.props` プロパティーは以下のとおりです。

- serverIP
- certServerIPAndPort

パスワード変更の IBM WebSphere MQ Everyplace トランスポートに対する高可用性の構成

以下の情報を使用することで、パスワード変更の IBM WebSphere MQ Everyplace トランスポートに対する高可用性の構成について学習できます。

高可用性デプロイメントをサポートするために、IBM Security Directory Integrator の IBM WebSphere MQ Everyplace コンポーネントの複数のインスタンスをデプロイおよび構成することができます。一部のデプロイメントでは、IBM Security Directory Integrator の複数の IBM WebSphere MQ Everyplace パスワード・ストア・コンポーネントを構成する必要がある場合があります。例えば、複数の Windows ドメイン・コントローラーについてパスワード変更プラグインが構成されている場合、「PWStoreClient」という名前の IBM WebSphere MQ Everyplace クライアント・サイド・キュー・マネージャーの別々のインスタンスがある可能性があります。さらに、各クライアント・キュー・マネージャーについて、IBM Security Directory Integrator の IBM WebSphere MQ Everyplace パスワード・コネクターが使用する IBM WebSphere MQ Everyplace サーバー・サイド・キュー・マネージャー・キューに対するリモート・キュー・プロキシー接続があります。リモート・キュー・プロキシーの名前は「PWStoreServer+passwords」です。このタイプのデプロイメントのシナリオを利用する場合、これらの 2 つの IBM WebSphere MQ Everyplace エンティティー (すなわち、「PWStoreClient」、「PWStoreServer+passwords」) に関連付けられた認証証明書が複数回要求され、発行されます。このことは、mqeconfig ユーティリティーが実行されるたびに発生します。2 番目以降の mqeconfig ユーティリティーのインスタンスを実行する前に、上述の各 IBM WebSphere MQ Everyplace エンティティーについて証明書の発行を再度使用可能にする必要があります。

特定のデプロイメントについて、特定の高可用性要件をサポートする、IBM Security Directory Integrator の IBM WebSphere MQ Everyplace パスワード・コネクターを構成する場合があります。ユーザーが、このタイプの要件をサポートするインプリメンテーションにおいて、それぞれが関連付けられた独自の IBM WebSphere MQ Everyplace キュー・マネージャー構成を持つ、IBM Security Directory Integrator の

IBM WebSphere MQ Everyplace パスワード・コネクターの複数のインスタンスが使用されるようにしたい場合があります。この場合、ユーザーは、複数の同じ IBM WebSphere MQ Everyplace サーバー・サイド構成をデプロイし、これによってネットワーク・ロード・バランサーは、IBM Security Directory Integrator の IBM WebSphere MQ Everyplace パスワード・ストア・クライアントから使用可能なサーバー・インスタンスに要求を送信できるようになります。サーバー・サイドの各 IBM WebSphere MQ Everyplace キュー・マネージャーは、mqeconfig ユーティリティーを使用して構成します。このユーティリティーは、実行すると、「PWStoreServer」および「PWStoreServer+passwords」という名前のエンティティーについて、IBM WebSphere MQ Everyplace Mini-Certificate サーバーから認証証明書を自動的に要求します。これらはそれぞれキュー・マネージャーおよびキュー名を表します。2 番目以降の mqeconfig ユーティリティーのインスタンスを実行する前に、上述の 2 つの IBM WebSphere MQ Everyplace エンティティーについて証明書の発行を再度使用可能にする必要があります。

IBM WebSphere MQ Everyplace 構成ユーティリティーにおけるリモート構成機能の提供

以下の説明に従って、IBM WebSphere MQ Everyplace 構成ユーティリティーでリモート構成機能を提供できます。

構成ユーティリティーを使用したリモート IBM WebSphere MQ Everyplace キューの作成

以下のコマンド行を入力すると、mqeconfig.props 構成ファイルを使用して、「queue_name」という名前のリモート IBM WebSphere MQ Everyplace キューが作成されます。

```
mqeconfig mqeconfig.props create remotequeue queue_name targetQMname [QM_ip_or hostname comm_port]
```

上記のコマンド行の QM_ip_or_hostname パラメーターおよび comm_port パラメーターはオプションです。これらを指定しない場合、リモート・キュー定義のみが作成されます。これらの 2 つのパラメーターを指定すると、リモート・キュー定義の作成前に、接続定義も作成されます。

注: リモート・キューは、接続定義がないと使用できません。また、単一の接続を共用するために、複数のリモート・キューを定義できます。targetQMname パラメーターでは、リモート IBM WebSphere MQ Everyplace キュー・マネージャーの名前を指定します。

IBM WebSphere MQ Everyplace 構成ユーティリティーを使用したリモート IBM WebSphere MQ Everyplace キューの削除

以下のコマンド行を入力すると、mqeconfig.props 構成ファイルを使用して、「queue_name」という名前のリモート IBM WebSphere MQ Everyplace キューが削除されます。

```
mqeconfig mqeconfig.props delete remotequeue queue_name targetQMname
```

上記のコマンド行の targetQMname パラメーターでは、リモート IBM WebSphere MQ Everyplace キュー・マネージャーの名前を指定します。

第 9 章 暗号化と FIPS モード

ここで提供される情報とリンクを使用することで、暗号化について学習できます。

データの機密性を確保するため、IBM Security Directory Integrator では以下を暗号化できます。

- 構成ファイル
- プロパティー・ファイル内のプロパティー値
- サーバー・ユーザー・レジストリー・ファイル
- JavaScript ファイル

*暗号化*とは、*プレーン・テキスト*と呼ばれる人間が読むことができるテキスト部分を選択し、その内容と意味を隠すことによって、プレーン・テキスト形式内のデータのセキュリティーを強化するプロセスです。プレーン・テキストは小文字で記述されます。暗号化されたテキストは*暗号文*と呼ばれます。暗号文は大文字で記述されます。

注: 構成ファイル内でプロパティー名の前に接頭部 {protect}- が付いている場合、プロパティー値は暗号化されています (あるいは、暗号化されます)。接頭部 *{protect}-* はオプションです。既に暗号値となっている値は *{encr}* で始まります。152 ページの『暗号化 IBM Security Directory Integrator 構成ファイルの処理』および 155 ページの『外部プロパティー・ファイルのプロパティーの暗号化』を参照してください。

例:

```
[{protect}-]keyword <colon | equals> [{encr}][{java}]value
```

{java} 値は b64 エンコードする必要があります。例を示します。

```
{protect}-api.truststore.pass={encr}J8AKimpEutu3BblOVg55F/5d5vO2kXWcNUWnCq3vINUc6K0719z9dEk3H43Ot2iTT1dZTI6FSSV
in9KsCyBLmgv+n84w7HelKl3ro2dFmZbTYKMXuxGoqN9nL2VOvZoptNqzoWvs6IN/p3VkIIBtlao/9mEPEKuIwRnKtkQ89Bg=
```

FIPS モードを実行するように IBM Security Directory Integrator を構成する

FIPS モードを実行するように IBM Security Directory Integrator を構成することができます。

「Federal Information Processing Standard (FIPS) Publication 140-2, FIPS PUB 140-2」は、暗号モジュールの認定のために使用される米国政府のコンピューター・セキュリティー規格です。

IBM Security Directory Integrator サーバーを FIPS モードで動作するように構成すると、サーバーでは FIPS 140-2 認定暗号モジュールが使用されるようになります。IBM Security Directory Integrator は暗号鍵は生成しません。鍵は *keytool* や *Ikeyman* のような外部ユーティリティーを使用して作成します。IBM Security Directory Integrator での暗号化の使用法については、105 ページの『第 6 章 セキュ

リティー』を参照してください。 鍵ストアおよびトラストストアを作成、編集、エクスポート、および全体的に管理するために、Ikeyman GUI ユーティリティーまたは keytool コマンド行ユーティリティーを使用できます。実行可能ファイル keytool.exe は、プラットフォームに応じて、*root_directory*/jvm/jre/bin、または *root_directory*/jvm/bin にあります。

対称型暗号サポート

FIPS 140-2 コンプライアンスを達成するためには対称型暗号サポートを使用できます。

メッセージを暗号化する理由は、メッセージを暗号文と呼ばれる無意味な形式のテキストと入れ替えて、メッセージを傍受する第三者には意味不明にすることです。暗号と呼ばれる暗号化アルゴリズムには、各種のアルゴリズムが多数あります。最も広く知られている暗号の 1 つに、*対称型*暗号があります。対称型暗号には、送信者と受信者の双方が保管する鍵が 1 つあります。送信者はその鍵を使用してメッセージを暗号化します。受信者は同じ鍵を使用してメッセージを復号します。

対称型暗号 (具体的には、Advanced Encryption Standard (AES)) を使用するためのオプションの構成が用意されています。AES を使用してエンコードされた対称型暗号は、FIPS 準拠のソリューションを必要とするユーザーに対して、サポート対象の暗号の使用を可能にします。

暗号は以下のプロパティーによって定義されます。

```
com.ibm.di.securityTransformation=DES/ECB/NoPadding
```

このプロパティーは、IBM Security Directory Integrator 構成のパスワード・ベースの暗号化または復号のための暗号を定義します。

FIPS 暗号化

FIPS を使用すると、IBM Security Directory Integrator および IBM Security Directory Integrator サーバーをセキュアに動作させることができます。 IBM Security Directory Integrator を特定モード (例えば、FIPS モード) で動作させるために、追加プロパティーを設定することもできます。

コネクター、関数コンポーネント、パーサー:

以下の情報を使用して、コネクター、関数コンポーネント、パーサーについて詳しく知ることができます。

FIPS 140-2 は、SSL、デジタル署名、暗号化、暗号学的ハッシュ関数、乱数生成などの暗号機能のみを扱っています。

SSL

FIPS 140-2 には、SSL 通信用のプロトコルとして TLS が必要です。 SSLv3 およびそれ以前のバージョンは使用できません。FIPS モードをオンにすると、SSL を使用している IBM Security Directory Integrator コンポーネントは TLS をサポートしていない外部システムとの通信に失敗します。

JDBC とシステム・ストア

IBM Security Directory Integrator に付属している DB2 Type 4 JDBC ドライバー (com.ibm.db2.jcc.DB2Driver) は、FIPS に準拠した方法で SSL をサポートします。

Apache Derby ドライバー (ネットワーク型と組み込み型) は、バージョン 10.5.3 (IBM Security Directory Integrator 7.2 より前のバージョンにバンドルされているバージョン) で SSL をサポートしていません。

ただし、Apache Derby バージョン 10.8 データベース・エンジンは、データベースの暗号化を行うことができます。IBM Security Directory Integrator は、デフォルトでは、システム・ストアとして Derby を使用します。FIPS モードのとき Apache Derby バージョン 10.8 のデータベース暗号化機能を使用する場合は、暗号化プロバイダーとして IBM 認定の暗号プロバイダー IBMJCEFIPS と、FIPS 承認の暗号を使用するようにしてください。以下に、FIPS 準拠のデータベース暗号化機能を持つ Derby を使用するようにシステム・ストアを構成する方法を示します。

```
com.ibm.di.store.database=jdbc:derby://localhost:1527/C:¥TDI¥TDISysStoreEnc;create=true;
  dataEncryption=true;encryptionKey=c566bab9ee8b62a5ddb4d9229224c678;encryptionAlgorithm=AES/CBC/NoPadding;
  encryptionProvider=com.ibm.crypto.fips.provider.IBMJCEFIPS
com.ibm.di.store.jdbc.driver=org.apache.derby.jdbc.ClientDriver
com.ibm.di.store.jdbc.urlprefix=jdbc:derby:
com.ibm.di.store.jdbc.user=APP
```

JMS とシステム・キュー

IBM WebSphere MQ Everyplace Mini-Certificates には、FIPS に準拠しない暗号が含まれているため、FIPS モードでは、この IBM WebSphere MQ Everyplace セキュリティー機能は使用できません。

IBM WebSphere MQ 5.3 JMS プロバイダーは、FIPS 準拠モードでは SSL を実行できません。FIPS モードでは、このプロバイダーで SSL を使用することはできません。

IBM Security Directory Integrator と IBM WebSphere MQ の間で FIPS 準拠の SSL 通信を使用するには、以下の操作を行います。

1. インストールされている IBM WebSphere MQ が バージョン 7.1 以降であることを確認します。
2. MQ 側の対応するキュー・マネージャーが FIPS 準拠の SSL 通信を必要としていることを確認します。
3. キュー・マネージャーの対応する SSL チャネルが FIPS 準拠の SSL 暗号化仕様を使用していることを確認します。
4. IBM Security Directory Integrator に対して FIPS モードをオンにします。IBM Security Directory Integrator で FIPS モードが有効になると、すべての JMS の IBM WebSphere MQ との SSL 接続で、自動的に FIPS モードが有効になります。
5. JMS クライアント jar を、WebSphere MQ インストール済み環境から IBM Security Directory Integrator にコピーします。MQ 7.1 に必要なクライアント・ライブラリーのリスト、およびそれらを IBM Security Directory Integrator にデプロイする方法については、「*リファレンス*」の JMS コネクターの資料を参照してください。

6. IBM Security Directory Integrator 側で、MQ キュー・マネージャーの SSL チャネルに構成した SSL 暗号化仕様と互換性のある FIPS 準拠の SSL 暗号スイートを構成します。それには、JMS コネクターの **jms.sslCipher** パラメーターとシステム・キュー用の MQ ドライバーの `systemqueue.jmsdriver.param.jms.sslCipher` システム・プロパティーを使用します。詳しくは、 WebSphere MQ の資料で、*SSL CipherSpecs と CipherSuites* マッピングおよびその FIPS コンプライアンスについてのセクションを参照してください。

IBM Security Directory Integrator サーバーおよび FIPS:

IBM Security Directory Integrator サーバーおよび FIPS を使用して作業する場合は、以下にリストされている意味に注意してください。

IBM Security Directory Integrator サーバーは、このモードで動作するときは、必ず、FIPS 140-2 暗号モジュールを使用します。

注: サーバーで FIPS と SSL を使用可能にしている場合、クライアントではセキュア・ソケット通信用に SSL は使用しないでください。その場合、サーバーでは TLS が使用されているので接続が成功しません。セキュア・ソケット通信用には SSL ではなく、サーバーと同様に TLS を使用するようにしてください。

IBM Security Directory Integrator サーバーを FIPS モードで動作させることには次の意味があります。

- 構成やプロパティーなどの暗号化と復号には、FIPS 準拠の暗号アルゴリズムのみが許可されます。
- 暗号化/復号を使用している補助ツール (Ikeyman、createstash、cryptoutils、keytool など) は、FIPS に準拠した方法で使用する必要があります。
- コンポーネントはソケット通信に TLS を使用していない外部システムとは通信できません。
- サーバーが FIPS モードのとき、一部のコンポーネントは使用してはいけません。FIPS コンプライアンスを阻害するからです。コンポーネントのコンプライアンスについてのリストは、196 ページの表 21を参照してください。

FIPS モードを使用可能にする:

FIPS を使用する場合は、多くの IBM Security Directory Integrator 構成オプションを変更します。したがって、FIPS コンプライアンスを維持するためにはいくつかの規則を守る必要があります。

そのうちいくつかの規則については本書で説明しますが、残りの規則については、https://w3.webahead.ibm.com/w3ki/download/attachments/370821/FIPS+140+Guidelines.pdf?version=1 と http://www.ibm.com/developerworks/java/jdk/security/60/FIPShowto.html を参照してください。

IBM Security Directory Integrator での FIPS モードの有効化

1. `global.properties` または `solution.properties` の `com.ibm.di.server.fipsmode.on` プロパティーに **true** を設定します。

2. com.ibm.di.securityTransformation プロパティーの値が、FIPS 準拠のアルゴリズム (例えば、AES/ECB/NoPadding) になっているか確認します。このアルゴリズムは暗号化された構成を開くことを試みたときに使用されます。
3. FIPS モードの SSL では、ハードウェア暗号デバイスは使用できません。基礎となる SSL モジュール IBMJSSE2 が FIPS モードではハードウェア暗号デバイスをサポートしていないからです。詳細については、http://www-128.ibm.com/developerworks/java/jdk/security/60/secguides/jsse2Docs/JSSE2RefGuide.html#runfips を参照してください。 FIPS モードのとき、サーバー API ではハードウェア・ベースの SSL 鍵は使用できません。com.ibm.di.server.pkcs11 プロパティーを指定しないか、または global.properties と solution.properties に false を設定する必要があります。
4. サーバー暗号化では FIPS 140-2 準拠の変換を必ず使用してください。

デフォルトでは、サーバーは RSA アルゴリズムを使用する公開鍵暗号化を使用します。しかし、RSA 暗号化オプションは FIPS 140-2 に準拠していません。そのため、手動で別の FIPS 準拠の暗号変換を構成する必要があります。以下に、 IBM Security Directory Integrator が暗号化に AES 暗号を使用するようにセットアップするためのサンプルの手順を示します。

- AES 共通鍵を生成し、それを鍵ストアに格納します。そのためには、次のように、IBM Security Directory Integrator インストール済み環境の bin フォルダー内にある keytool ユーティリティーを使用します。

  ```
  keytool –genseckey –alias server –keyalg AES –keysize 128 –keystore server.jck –storepass mypass –storetype jceks
   –keypass mykeypass –providerClass com.ibm.crypto.fips.provider.IBMJCEFIPS
  ```

  このコマンドで、サイズが 128 で別名 *server* の AES 鍵を持つタイプが JCEKS (JKS 鍵ストアは共通鍵をホストできません) の新しい鍵ストア・ファイル server.jck が作成されます。作成された鍵ストアのパスワードは *mypass* です。*keygenproviderclass* パラメーターには特に注意してください。FIPS 140-2 コンプライアンスを達成する場合は、FIPS 認定プロバイダーを指定するために必須のパラメーターだからです。これは一例に過ぎません。ファイル名、パスワード、別名は自由に変えてください。
- 新規に生成した鍵で共通鍵暗号化を使用するため、IBM Security Directory Integrator の設定を変更します。例えば、global.properties ファイルまたは solution.properties ファイルに、以下のプロパティーを設定します。

  ```
  com.ibm.di.server.encryption.keystore=server.jck
  com.ibm.di.server.encryption.keystoretype=jceks
  com.ibm.di.server.encryption.key.alias=server
  com.ibm.di.server.encryption.transformation=AES/CBC/PKCS5Padding
  ```
- 古い鍵で暗号化されていた既存のファイルすべてを移行します。

  新しい鍵を導入する前から存在したすべての暗号化ファイルは移行する必要があります。マイグレーションは、古い鍵を使用して復号する手順と、新しい鍵を使用して再暗号化する (オプション) 手順で構成されます (208 ページの『暗号化の成果物 (鍵、証明書、鍵ストア、暗号化ファイル) の管理』を参照)。例えば、global.properties を移行するには、以下のコマンドを実行します。

  ```
  cryptoutils -input ../etc/global.properties -output ../etc/global.properties
    -mode decrypt_props -keystore ../testserver.jks -storepass server -alias server
    -transformation RSA -storetype jks -keypass server

  cryptoutils -input ../etc/global.properties -output ../etc/global.properties
    -mode encrypt_props -keystore ../server.jck -storepass mypass -alias server
    -transformation AES/CBC/PKCS5Padding -storetype jceks -keypass mykeypass
  ```

- IBM Security Directory Integrator サーバーの 150 ページの『Stash ファイル』 を再生成して、暗号化鍵ストアおよび暗号化鍵の新しいパスワードを反映します。そのためには、IBM Security Directory Integrator インストール済み環境の `bin` フォルダー内にある `createstash` ユーティリティーを使用します。例:

  ```
  createstash mypass mykeypass com.ibm.crypto.fips.provider.IBMJCEFIPS
  ```

- ソリューションでは、以下の表にリストした FIPS 準拠の IBM Security Directory Integrator コンポーネントのみを使用してください。

表 21. FIPS 互換のコンポーネント

Directory Integrator のコンポーネント	**FIPS モードでの使用可否**	**注釈**
コネクター		
Active Directory 変更検出コネクター	はい	SSL 用にデフォルトの JSSE ファクトリーを使用
AssemblyLine コネクター	はい	サーバー API クライアントとして作動
Axis Easy Web Service サーバー・コネクター	はい	SSL 用にデフォルトの JSSE ファクトリーを使用
コマンド行コネクター	はい	暗号化機能なし
Domino/Lotus Notes コネクター	いいえ	Lotus Domino/Lotus Notes 7 暗号化機能は FIPS に準拠しない (一部の FIPS イネーブルメントは Notes 8.0.1 に含まれています。)
ITIM DSMLv2 コネクター	はい	SSL 用にデフォルトの JSSE ファクトリーを使用
DSMLv2 SOAP コネクター	はい	SSL 用にデフォルトの JSSE ファクトリーを使用
DSMLv2 SOAP サーバー・コネクター	はい	SSL 用にデフォルトの JSSE ファクトリーを使用
Exchange 変更ログ・コネクター	はい	SSL 用にデフォルトの JSSE ファクトリーを使用
ファイル・コネクター	はい	暗号化機能なし
FTP クライアント・コネクター	はい	暗号化機能なし
GLA コネクター	はい	暗号化機能なし。このコネクターは推奨されません。IBM Security Directory Integrator の将来のバージョンでは削除されます。
HTTP クライアント・コネクター	はい	SSL 用にデフォルトの JSSE ファクトリーを使用
旧 HTTP クライアント・コネクター	はい	SSL 用にデフォルトの JSSE ファクトリーを使用
HTTP サーバー・コネクター	はい	SSL 用にデフォルトの JSSE ファクトリーを使用
旧 HTTP サーバー・コネクター	はい	暗号化機能なし

表 21. FIPS 互換のコンポーネント (続き)

Directory Integrator のコンポーネント	**FIPS モードでの使用可否**	**注釈**
IBM Security Directory Integrator 変更ログ・コネクター	はい	SSL 用にデフォルトの JSSE ファクトリーを使用
ITIM Agent コネクター	はい	暗号化機能なし
JDBC コネクター	依存関係	暗号化機能 (SSL、暗号化) を使用しなければコネクターは FIPS に準拠します。 それ以外の場合、FIPS への準拠は、使用する JDBC ドライバーの暗号化機能が FIPS へ準拠しているかどうかに依存します。 JDBC ドライバーが FIPS へ準拠しているかどうかについては、 192 ページの『コネクター、関数コンポーネント、パーサー』を参照してください。
JMS コネクター	依存関係	暗号化機能 (SSL、暗号化) を使用しなければコネクターは FIPS に準拠します。 それ以外の場合、FIPS への準拠は、使用する JDBC ドライバーの暗号化機能が FIPS へ準拠しているかどうかに依存します。 JMS プロバイダーの FIPS への準拠に関して、詳しくは 192 ページの『コネクター、関数コンポーネント、パーサー』を参照してください。
JMX コネクター	はい	暗号化機能なし
JNDI コネクター	はい	SSL 用にデフォルトの JSSE ファクトリーを使用
LDAP コネクター	はい	SSL 用にデフォルトの JSSE ファクトリーを使用
LDAP サーバー・コネクター	はい	SSL 用にデフォルトの JSSE ファクトリーを使用
メールボックス・コネクター	はい	SSL 用にデフォルトの JSSE ファクトリーを使用
メモリー・キュー・コネクター	依存関係	システム・ストアで使用する JDBC ドライバーの FIPS コンプライアンスに依存します。 (メモリー・キューはパーシスタンスのためにシステム・ストアを使用します。) JDBC ドライバーが FIPS へ準拠しているかどうかについては、 192 ページの『コネクター、関数コンポーネント、パーサー』を参照してください。
メモリー・ストリーム・コネクター	はい	暗号化機能なし

表 21. FIPS 互換のコンポーネント (続き)

Directory Integrator のコンポーネント	FIPS モードでの使用可否	注釈
IBM WebSphere MQ Everyplace パスワード・ストア・コネクター	依存関係	メッセージ保護のために、FIPS モードで許可されているのは、PKCS#7 のみです。 RSA 暗号化オプションを使用しないでください。IBM WebSphere MQ Everyplace Mini Certificates は FIPS に準拠していないので、FIPS モードで使用しないでください。
Sun Directory 変更検出コネクター	はい	SSL 用にデフォルトの JSSE ファクトリーを使用
プロパティー・コネクター	依存関係	暗号化がオフの場合、コネクターは FIPS に準拠しています。 それ以外の場合の FIPS コンプライアンスは、暗号化で使用される暗号に依存します。 FIPS 140-2 で認定された暗号の例としては、AES があります。その他の認定された暗号について詳しくは、http://csrc.nist.gov/publications/fips/fips140-2/fips1402annexa.pdf を参照してください。 サーバーの暗号化オプションは、IBM Security Directory Integrator を正しく FIPS モードに構成する限り、常に FIPS に準拠します。 (194 ページの『FIPS モードを使用可能にする』を参照。)
サーバー通知コネクター	はい	サーバー API クライアントとして作動
システム・キュー・コネクター	依存関係	システム・キューが暗号化機能 (SSL、暗号化) を使用していない場合は、コネクターは FIPS に準拠します。 それ以外の場合、FIPS への準拠は、システム・キューで使用される JMS プロバイダーが FIPS へ準拠しているかどうかに依存します。 JMS プロバイダーの FIPS への準拠に関して、詳しくは 192 ページの『コネクター、関数コンポーネント、パーサー』を参照してください。
Windows ユーザー/グループ・コネクター	はい	暗号化機能なし
システム・ストア・コネクター	依存関係	システム・ストアで使用する JDBC ドライバーの FIPS コンプライアンスに依存します。
RAC コネクター	はい	暗号化機能なし。このコネクターは推奨されません。IBM Security Directory Integrator の将来のバージョンでは削除されます。
RDBMS 変更ログ・コネクター	依存関係	JDBC コネクターと同じ

表 21. FIPS 互換のコンポーネント (続き)

Directory Integrator のコンポーネント	**FIPS モードでの使用可否**	**注釈**
SNMP コネクター	はい	暗号化機能なし
SNMP サーバー・コネクター	はい	暗号化機能なし
TAM コネクター	はい	IBM Security Directory Integrator Runtime for Java は FIPS に準拠しています
TCP コネクター	はい	SSL 用にデフォルトの JSSE ファクトリーを使用
TCP サーバー・コネクター	はい	SSL 用にデフォルトの JSSE ファクトリーを使用
タイマー・コネクター	はい	暗号化機能なし
URL コネクター	はい	暗号化機能なし
Web サービス受信側サーバー・コネクター	はい	SSL 用にデフォルトの JSSE ファクトリーを使用
z/OS® 変更ログ・コネクター	はい	SSL 用にデフォルトの JSSE ファクトリーを使用
関数コンポーネント		
Castor Java から XML FC	はい	暗号化機能なし
Castor XML から Java FC	はい	暗号化機能なし
EMF XMLToSDO	はい	暗号化機能なし
EMF SDOToXML	はい	暗号化機能なし
AssemblyLine 関数コンポーネント	はい	サーバー API クライアントとして作動
Java クラス関数コンポーネント	依存関係	Java クラスの FIPS コンプライアンスに依存します。このメソッドは関数コンポーネントにより呼び出されます。 Java クラスで暗号化 (SSL、暗号化、署名、暗号学的ハッシュ関数など) を使用していない場合、FIPS モードで安全に使用できます。
パーサー関数コンポーネント	依存関係	関数コンポーネントに対して構成されているパーサーの FIPS コンプライアンスに依存します。
CBE ジェネレーター関数コンポーネント	はい	暗号化機能なし
SendEMail 関数コンポーネント	はい	SSL 用にデフォルトの JSSE ファクトリーを使用

表 21. FIPS 互換のコンポーネント (続き)

Directory Integrator のコンポーネント	**FIPS モードでの使用可否**	**注釈**
メモリー・キュー関数コンポーネント	依存関係	システム・ストアで使用する JDBC ドライバーの FIPS コンプライアンスに依存します。 (メモリー・キューはパーシスタンスのためにシステム・ストアを使用します。) JDBC ドライバーが FIPS へ準拠しているかどうかについては、192 ページの『コネクター、関数コンポーネント、パーサー』を参照してください。
Axis Java から SOAP への変換関数コンポーネント	はい	暗号化機能なし
WrapSoap 関数コンポーネント	はい	暗号化機能なし
Soap WS FC を起動	はい	SSL 用にデフォルトの JSSE ファクトリーを使用
Axis Soap から Java への変換関数コンポーネント	はい	暗号化機能なし
Axis EasyInvoke Soap WS 関数コンポーネント	はい	SSL 用にデフォルトの JSSE ファクトリーを使用
複素数タイプ・ジェネレーター関数コンポーネント	はい	暗号化機能なし
リモート・コマンド行関数コンポーネント	依存関係	RXA ツールキットの暗号化機能は FIPS に準拠していません。 暗号化機能が使用されていない場合、コンポーネントは FIPS モードで使用できます。
z/OS TSO/E コマンド行関数コンポーネント	依存関係	関数コンポーネントで呼び出される TSO コマンドに含まれている暗号化機能の FIPS コンプライアンスに依存します。
SAP ABAP Application Server コンポーネント・スイート	いいえ	SAP 暗号モジュールは FIPS 140-2 の認定を受けていません。 暗号化機能を使用していなければ、コンポーネントは FIPS モードで使用できます。
パーサー	はい	IBM Security Directory Integrator パーサーのコンポーネントはどれも暗号化機能を使用していないため、すべてのコンポーネントが FIPS モードで使用できます。

com.ibm.di.server.fipsmode.on の設定

IBM Security Directory Integrator で FIPS を有効にするには、`global.properties` または `solution.properties` のプロパティーで FIPS を指定する必要があります。

このプロパティーの名前は com.ibm.di.server.fipsmode.on で、**true** か **false** のいずれかを設定できます。このプロパティーを **true** に設定すると、IBM Security Directory Integrator サーバーは FIPS モードで動作します。このモードでは、IBM FIPS セキュリティー・プロバイダーは、プロバイダー・リストの IBM JCE セキュリティー・プロバイダーの前に IBM Security Directory Integrator JVM で設定されます。IBM Security Directory Integrator FIPS 使用可能化プロパティーが true に設定されている場合、IBM JSSE2 プロバイダーの FIPS モードも使用可能に設定され、デフォルトの JSSE SSL ソケット・ファクトリーを IBM JSSE2 プロバイダーに属するものとして設定します。IBM Security Directory Integrator ではデフォルトで FIPS モードは使用不可です。すなわち、com.ibm.di.server.fipsmode.on プロパティーは **false** に設定されています。

FIPS モードで暗号アルゴリズムを使用する

FIPS 準拠の暗号アルゴリズムのみを使用できます。すなわち、FIPS 準拠モードを維持するには、FIPS 準拠のアルゴリズムのみを使用する必要があります。他のアルゴリズムを使用すると FIPS コンプライアンスに違反します。

com.ibm.di.securityTransformation の設定

暗号化構成を開くときに、IBM Security Directory Integrator は com.ibm.di.securityTransformation プロパティーを使用して構成を復号するアルゴリズムを取得します。このプロパティーが FIPS 準拠ではないアルゴリズムに設定され、IBM Security Directory Integrator サーバーの FIPS モードがオンになっている場合、例外がスローされます。表示される例外メッセージは次のようなものになります。

```
CTGDIC012E Could not load file<FILE_PATH>. No such algorithm: <ALGORITHM_NAME>.
```

この例外を回避するには、FIPS モードで実行中に常に、このプロパティーに FIPS 準拠アルゴリズムを設定します。デフォルトでは、com.ibm.di.securityTransformation プロパティーには、FIPS 準拠のアルゴリズム **ではない** DES/ECB/Nopadding が設定されます。このプロパティーでは、IBM Security Directory Integrator 構成のパスワード・ベースの暗号化と復号のための暗号も定義されます。

FIPS モードで実行中にプロパティーを自動的に設定する

- IBM Security Directory Integrator は、デフォルトでは global.properties ファイルに存在しない関連システム・プロパティーを設定します。このプロパティーは com.ibm.di.cryptoProvider と呼ばれ、FIPS モードで実行中に IBMJCEFIPS セキュリティー・プロバイダーに設定されます。このプロパティーが global.properties に設定されている場合は、その特定の値が使用されることに注意してください。このプロパティーに非 FIPS 準拠のプロバイダーが設定されている場合、IBM Security Directory Integrator が FIPS モードで実行されても、IBM Security Directory Integrator は FIPS 準拠には **なりません**。
- FIPS モードの場合、特定の JSSE ソケット・ファクトリーが使用されます。この場合は、IBMJSSE2 ソケット・ファクトリーです。これは、ssl.SocketFactory.provider および ssl.ServerSocketFactory.provider プロパ

ティーを IBMJSSE2 プロバイダーの JSSE インプリメンテーション・クラスに設定する IBM Security Directory Integrator サーバーによって、自動的に行われます。

FIPS モードで stash ファイル作成コマンド行ツールを使用する

FIPS 140-2 規格に準拠する stash ファイルを作成するには、createstash ファイル・ツールを使用するときに、3 番目のパラメーターとして IBMJCEFIPS プロバイダー・クラスを指定する必要があります。 例を示します。

```
TDI_install_dir¥bin¥createstash Password Password com.ibm.crypto.fips.provider.IBMJCEFIPS
```

FIPS モードで RSA 暗号化の代替アルゴリズムを使用する

FIPS モードで RSA 暗号アルゴリズムの代わりに Advanced Encryption Standard (AES) を使用するように IBM Security Directory Integrator を構成します。FIPS 140-2 に準拠する共通鍵暗号が必要になります。RSA はこのアルゴリズムの発明者である Rivest、Shamir、および Adelman の頭文字を採って名付けられました。RSA アルゴリズムはインターネット経由のデータ送信で使用される強力な暗号化アルゴリズムです。RSA 暗号では、転送 (SSL、TLS) 時に FIPS 140-2 Level 1 の認定モードの境界内に留まるために鍵の暗号化と復号を行うことのみが許可されます。詳しくは、http://www.ibm.com/developerworks/java/jdk/security/60/FIPShowto.htmlを参照してください。

FIPS モードで補助ツールを実行する:

適切な暗号プロバイダーと、秘密鍵を生成するタイミングを指定するためのコマンド行構文を使用できます。

createstash

コマンド行で明示的なプロバイダー・パラメーターとして、FIPS 140-2 認定暗号プロバイダー IBMJCEFIPS を渡します。

```
createstash mypass mykeypass com.ibm.crypto.fips.provider.IBMJCEFIPS
```

cryptoutils

コマンド行で FIPS 140-2 認定暗号プロバイダー IBMJCEFIPS を、次のように *cryptoproviderclass* オプションを使用して、明示的なプロバイダーとして渡します。

```
cryptoutils –input registry.txt –output registry.enc –mode encrypt –keystore ../testserver.jks –storepass server
 –alias server –cryptoproviderclass com.ibm.crypto.fips.provider.IBMJCEFIPS
```

IBM Security Directory Integrator の FIPS プロパティーの構成:

ここに示されている手順を使用して、IBM Security Directory Integrator の FIPS プロパティーを構成できます。

FIPS モードで keytool/Ikeyman を実行する

FIPS モードで keytool および Ikeyman ユーティリティーを使用するために、*TDI_install_dir*/jvm/jre/lib/security 内の java.security ファイルを編集します。java.security ファイルの先頭の 2 行で、1 行目に IBMJCEFIPS プロバイダーを、2 行目に IBMJCE セキュリティー・プロバイダーをそれぞれ指定します。例を示します。

```
security.provider.1=com.ibm.crypto.fips.provider.IBMJCEFIPS
security.provider.2=com.ibm.crypto.provider.IBMJCE
```

ただし、Solaris と HP-UX の場合、セキュリティー・プロバイダー・リスト内では、SUN プロバイダーを必ず最初のプロバイダーにする必要があります。

SSL および PKI 証明書を構成する

以下の情報を参照することで、SSL および PKI 証明書について詳しく知ることができます。

IBM Security Directory Integrator は、Secure Socket Layer (SSL) 暗号化方式と Public Key Infrastructure (PKI) 暗号化方式の両方を使用します。SSL および PKI は、IBM Security Directory Integrator および IBM Security Directory Integrator サーバーの多くの機能のための重要な基盤を提供します。 SSL は、リモート通信を行う 2 局間のネットワーク・トラフィックの暗号化および認証を提供します。同様に、PKI (public key infrastructure) を使用すると、セキュアでないネットワークのユーザーが、信頼できる機関から取得して共有する公開鍵と秘密鍵の暗号鍵ペアを使用して、セキュアかつプライベートにデータを交換できます。 『SSL および PKI 証明書を構成する』を参照してください。

SSL 証明書

SSL 証明書はセキュア・サーバーに保管され、サーバーを識別するデータを暗号化するために使用されます。SSL 証明書を使用すると、サイトがその要求を行ったエンティティーに属することが証明されます。証明書には、証明書の所有者、証明書の発行先のドメイン、証明書を発行した認証局の名前、それが発行されたルートと国などの情報が含まれます。

PKI 証明書

PKI 証明書を使用すると、セキュアでないネットワークのユーザーがデータ交換の際に、セキュリティーとプライバシーを強化できます。PKI では、認証局 (CA) と呼ばれる信頼できる機関から取得して共有する暗号鍵ペアを使用します。PKI を使用すると、個人または組織を識別できる証明書と、証明書を保管できるディレクトリー・サービスを取得することができます。CA は、必要に応じて、証明書を取り消すこともできます。デジタル証明書の最も一般的な使用法は、メッセージを送信したユーザーが送信者であると主張している本人であることを検証し、受信者に暗号化した応答を返すことです。

以下の手順に従って、PKI 暗号化と SSL で使用される証明書のためのそれぞれが独立した構成オプションを用意します。

1. 以下のプロパティーを追加します。

```
com.ibm.di.server.encryption.keystore
com.ibm.di.server.encryption.key.alias
api.keystore.password
api.key.password
```

2. 以下のプロパティーを次のように名前を変更します。

```
com.ibm.di.server.keystore ----- > api.keystore
com.ibm.di.server.key.alias ------>api.key.alias
```

注: これで idisrv.sth ファイルには、暗号化ファイルのみのパスワードが格納されました。

CryptoUtils を使用した暗号化と復号

ここで説明する方法を使用することで、CryptoUtils を使用した暗号化と復号について学習できます。

IBM Security Directory Integrator を使用すると、global.properties ファイルまたは solution.properties ファイル内の機密情報を含むプロパティーを PKI で暗号化することができます。PKI で暗号化したプロパティーを復号する方法の 1 つは、構成エディター (CE) のプロパティー・エディターを使用することです。PKI で暗号化したプロパティー・ファイルを復号するためのもう 1 つの方法は、CryptoUtils コマンド行ユーティリティーを使用することです。復号では無許可のユーザーが、CryptoUtils を使用して PKI で暗号化されたプロパティーを含むプロパティー・ファイル内の重要な情報にはアクセスできないように、自分の PKI 資格情報を入力する必要があります。 156 ページの『IBM Security Directory Integrator 暗号化ユーティリティー』を参照してください。

証明書を操作する

ここで提供する情報を参照して、証明書を操作する方法を理解できます。

暗号化したメッセージを送信する場合は、CA にデジタル証明書を申請する必要があります。CA は申請者の公開鍵やその他の識別データを含む暗号化されたデジタル証明書を発行します。CA は固有の公開鍵を印刷媒体あるいはインターネットを通じて公開します。暗号化されたメッセージの受信者は、CA の公開鍵を使用してメッセージに添付されているデジタル証明書をデコードして、証明書がその CA から発行されたものであることを確認してから、送信者の公開鍵と識別データを証明書から取り出します。この情報を使用すれば、受信者は暗号化された応答を返すことができます。

デジタル証明書には 2 つのタイプがあります。

- CA が署名した証明書
- 自己署名証明書

CA が署名した証明書とは、VeriSign や thawte のような認証局によって署名された証明書のことです。自己署名証明書とは、作成者自身が署名した識別用の証明書のことです。

IBM Security Directory Integrator には、public key infrastructure (PKI) 暗号化で使用する証明書の構成オプションと Secure Socket Layer (SSL) 接続で使用する証明書の構成オプションがそれぞれ独立に用意されています。PKI 証明書と SSL 証明書は独立して構成できるので、暗号化されたプロパティーのマイグレーションは、SSL 証明書をアップグレードするプロセスとは切り離して行うことができます。

PKI の場合、認証局 (CA) は公開鍵をユーザー識別情報にバインドします。ユーザー識別情報は各 CA で一意である必要があります。公開鍵証明書には、各ユーザー、ユーザー識別情報、公開鍵、それらのバインディング、妥当性条件、および CA が発行した公開鍵証明書へ記録が必要なその他の属性を収集します。

SSL で使用される証明書には有効期限があります。またセキュリティー上の理由から、頻繁に更新する必要があります。PKI 暗号化で使用される証明書には、SSL 証明書が存続する期間よりも長い存続期間を与えることができます。PKI 証明書は、公開鍵証明書を使用して暗号化されたデータを含む場合、保管しておく必要があります。そのため、IBM Security Directory Integrator では、PKI 証明書と SSL 証明書は独立して構成できるようになっています。SSL 接続の各サーバーおよび PKI 認証を実行している各クライアントは、ローカル CA に証明書の発行を要求し、取得した証明書を鍵ストアに追加する必要があります。

これらのプロパティーは以下のとおり、global.properties ファイルに追加されます。

```
com.ibm.di.server.encryption.keystore
com.ibm.di.server.encryption.key.alias
```

これらのプロパティー変数には、以下のとおり、既に global.properties に設定されている値と同じ値が設定されます。

```
api.keystore=truststore
api.key.alias=server
```

CA が署名した証明書と自己署名証明書の比較

以下の情報を参照して、CA が署名した証明書と自己署名証明書を比較することができます。

認証局が署名した証明書

自己署名証明書

VeriSign のような認証局では、申請者が、自らの識別情報を提出し、証明書申請者の識別情報と、証明書の署名者としての認証局の識別情報の両方を認証する証明書を取得するための手順が必要です。

通常、ローカルの認証局 (CA) があります。つまり、証明書は VeriSign のような著名な認証局からは送られてきません。ローカル CA 自体は既知の CA から発行されたルート証明書を持つべきですが、常にそうとは限りません。ローカル CA のルート証明書が自己署名の場合、SSL を使用している各サーバーまたはクライアントのトラストストアにこれをインポートする必要があります。

この場合、SSL 接続の各サーバーおよび PKI 認証を行う各クライアントは、独自の自己署名証明書を生成します。その後、その証明書をファイルにエクスポートし、それをさまざまなトラストストアにインポートする必要があります。クライアント **C** がサーバー **S** に接続する場合、C はそのトラストストアに S の自己署名証明書を保持している必要があります。クライアント C がサーバー S に対して PKI 認証 (シンメトリックな SSL) を行う場合、S はそのトラストストアに C の自己署名証明書を保持している必要があります。自己署名証明書は、クライアントまたはサーバーのいずれの証明書にも使用できます。この方法の詳細については、105 ページの『鍵、証明書、および鍵ストアの管理』を参照してください。SSL 接続の各サーバーおよび PKI 認証を行う各クライアントは、ローカル CA に対して証明書の要求を発行し、その結果取得した証明書をその鍵ストアに追加する必要があります。

PKI および SSL を使用して証明書を構成する

以下の情報を使用して、PKI および SSL を使用する証明書の構成について学習できます。

IBM Security Directory Integrator には、public key infrastructure (PKI) 暗号化で使用する証明書の構成オプションと Secure Socket Layer (SSL) 接続で使用する証明書の構成オプションがそれぞれ独立に用意されています。PKI 証明書と SSL 証明書は独立して構成できるので、暗号化されたプロパティーのマイグレーションは、SSL 証明書をアップグレードするプロセスとは切り離して行うことができます。

PKI の場合、認証局 (CA) は公開鍵をユーザー識別情報にバインドします。ユーザー識別情報は各 CA で一意である必要があります。公開鍵証明書には、各ユーザー、ユーザー識別情報、公開鍵、それらのバインディング、妥当性条件、および CA が発行した公開鍵証明書へ記録が必要なその他の属性を収集します。

SSL で使用される証明書には有効期限があります。またセキュリティー上の理由から、頻繁に更新する必要があります。PKI 暗号化で使用される証明書には、SSL 証明書が存続する期間よりも長い存続期間を与えることができます。PKI 証明書は、公開鍵証明書を使用して暗号化されたデータを含む場合、保管しておく必要があります。そのため、IBM Security Directory Integrator では、PKI 証明書と SSL 証明書は独立して構成できるようになっています。SSL 接続の各サーバーおよび PKI 認証を実行している各クライアントは、ローカル CA に証明書の発行を要求し、取得した証明書を鍵ストアに追加する必要があります。

これらのプロパティー―は以下のとおり、global.properties ファイルに追加されます。

```
com.ibm.di.server.encryption.keystore
com.ibm.di.server.encryption.key.alias
```

これらのプロパティー変数には、以下のとおり、既に global.properties に設定されている値と同じ値が設定されます。

```
api.keystore=truststore
api.key.alias=server
```

ハードウェア・デバイスの暗号鍵を使用する

以下に示されている手順を使用することで、ハードウェア・デバイスの暗号鍵の使用について学習できます。

RSA 署名および暗号化アルゴリズム (Ron **R**ivest、Adi **S**hamir、および Leonard **A**dleman が開発) は、有名な公開鍵暗号です。 RSA Laboratories (EMC Corp. に所属) は、PKCS#11 標準を公表しました。この標準では、ハードウェア・セキュリティー・モジュールやスマート・カードのようなハードウェア暗号トークンに対するプラットフォーム独立の API が規定されています。PKCS#11 の API では、次のような、最も一般的に使用される暗号化オブジェクト・タイプを規定しています。

- RSA 鍵
- X.509 証明書
- データ暗号化規格 (DES)DES/Triple DES 鍵

- 上記の鍵の使用、作成または生成、変更、および削除に必要なすべての関数

Public-Key Cryptography Standards (PKCS) PKCS#11 は、各種のプラットフォーム上の各種のハードウェア暗号デバイスを使用した暗号サービスに対する一般的なアプリケーション・インターフェースを規定している標準です。ハードウェア暗号鍵ストレージ・デバイスでは、鍵をハードウェア・デバイスに保管することができます。IBM Security Directory Integrator は、PKCS#11 に準拠した暗号デバイスの秘密鍵と証明書をサポートします。Java ランタイム環境 (JRE) に付属している IBM Java PKCS ライブラリーでサポートされるすべてのハードウェア・デバイスがサポートされます。PKCS 標準は、ネットワーク上の public key infrastructure (PKI) を使用したセキュアな情報交換を実現するための一般的なプロトコルのセットです。IBM Security Directory Integratorでは、Secure Socket Layer (SSL) 鍵をハードウェア・デバイスで保管できます。鍵をハードウェア・デバイスで保管する必要が生じたため、`global.properties` ファイルには以下の新規のプロパティーが追加されました。

```
##PKCS11 options
##Set the value of following properties to use PKCS11 enabled devices to store TDI servers
##private key /certificate.
com.ibm.di.pkcs11cfg=etc¥pkcs11.cfg
com.ibm.di.server.pkcs11=false
com.ibm.di.server.pkcs11.library=
com.ibm.di.server.pkcs11.slot=
{protect}-com.ibm.di.server.pkcs11.password=PASSWORD
```

プロパティー `com.ibm.di.server.pkcs11` のデフォルト値は false です。プロパティー `com.ibm.di.server.pkcs11.password` に対応する値は暗号化されます。

IBMPCKS11 の使用

IBMPCKS11 を使用してデバイスにアクセスし、SSL 鍵と証明書を保管します。

IBM Security Directory Integrator は、IBMPCKS11 を使用してハードウェア暗号デバイスにアクセスして、SSL 鍵と証明書を保管します。IBM Java PKCS ライブラリーでサポートされ、Java ランタイム環境に付属しているすべてのハードウェア・デバイスに対してサポートが提供されます。

表 22. SSL がサポートされるプロパティー

プロパティー	デフォルト値	説明
com.ibm.di.pkcs11.cfg	etc¥pkcs11.cfg	CFG ファイルを使用して、IBM PKCS11 インプリメンテーション・プロバイダーの初期化に必要な構成ファイルのパスを指示します。
com.ibm.di.server.pkcs11	false	SSL 用に PKCS#11 準拠の暗号デバイスを使用します。
com.ibm.di.server.pkcs11.library		このプロパティーを使用して PKCS11 クライアント・ライブラリーへのパスを指定します。
com.ibm.di.server.pkcs11.slot		デバイスのスロット番号を指定します。
{protect}-com.ibm.di.server.pkcs11.pass		このパスワードを使用して PKCS11 準拠の暗号デバイスにアクセスします。
com.ibm.di.server.pkcs11.accl	false	=true を使用して暗号操作用のハードウェア暗号デバイスを設定します。

埋め込みを使用可能化または使用不可にする

ここで提供される情報とリンクを使用して、埋め込みを使用可能または使用不可にすることができます。

埋め込みとは、伝送データに追加ビットを追加して、伝送データを正確に要求されたサイズにすることです。一部の暗号化および復号アルゴリズムでは、入力が正確にブロック・サイズの倍数になっていることが要求されます。暗号化するプレーン・テキストが正確な倍数になっていない場合、暗号化する前に埋め込み文字列を追加して埋め込んでおく必要があります。復号時には受信側に埋め込みの削除方法を通知します。

注: global.properties ファイルにリストしたすべてのプロパティーは、同じ名前で構成ファイルに設定できます。ただし、ご自身の solution.properties ファイルを持っている場合は、それを編集することをお勧めします。これらのプロパティーは、{protect}- 接頭部 (詳しくは、 155 ページの『global.properties または solution.properties の標準 暗号化』を参照) を使用して暗号化することで保護できます。
埋め込みのプロパティーを設定する場合、デフォルト値は DES/ECB/NoPadding です。埋め込みプロパティーは、IBM Security Directory Integrator 構成のパスワード・ベースの暗号化と復号のアルゴリズムまたは暗号を定義します。プロパティーは、com.ibm.di.securityTransformation です。

暗号化の成果物 (鍵、証明書、鍵ストア、暗号化ファイル) の管理

ここで提供される情報とリンクを使用して、暗号化の成果物 (鍵、証明書、鍵ストア、暗号化ファイル) の管理について学習できます。

注: IBM Security Directory Integrator のデフォルトの SSL 証明書は、7.1.1 で変更されています。このため、デフォルトの IBM Security Directory Integrator 証明書を使用して暗号化されている場合、 IBM Security Directory Integrator 7.1.1 では次の項目の暗号化解除に失敗します。

- global.properties または solution.properties 内の暗号化されたパスワード
- 外部プロパティー・ストア内にある保護されたプロパティー
- 前の各バージョンで暗号化された IBM Security Directory Integrator 構成 XML ファイル

したがって、IBM Security Directory Integrator の前のバージョンを使用してすべての暗号化されたパスワードを暗号化解除します。暗号化ユーティリティーについて詳しくは、 156 ページの『IBM Security Directory Integrator 暗号化ユーティリティー』を参照してください。パスワードをテキストとして取得したら、それらを最新バージョンで使用します。サーバーまたは構成エディターが開始されると、パスワードは新規のデフォルトの証明書で暗号化されます。

変更した暗号鍵

サーバーが暗号化で使用する鍵を変更した場合は、常に、既存の暗号化ファイルをマイグレーションする必要性が生じます。暗号化ファイルをマイグレーションするには、古い暗号鍵で復号してから、新しい暗号鍵で暗号化しなおす必要があります。暗号化と復号は、 156 ページの『IBM Security Directory Integrator 暗号化ユーティリティー』 ツールを使用して行います。

しばしば暗号化されるファイル、あるいは暗号化される部分を持つファイルは、次のとおりです。 152 ページの『暗号化 IBM Security Directory Integrator 構成ファイルの処理』、 143 ページの『サーバー API ユーザー・レジストリー』、および 155 ページの

155 ページの『global.properties または solution.properties の標準 暗号化』 (IBM Security Directory Integrator プロパティー・ファイルは暗号化されたプロパティーを含むことができます。ただし、通常、ファイル全体が暗号化されることはありません)。

注: デフォルトでは `global.properties` または `solution.properties` 内の重要なプロパティー (パスワードなど) はすべて暗号化されます。おおまかな目安としては、サーバー暗号鍵を変更したときは、必ず、`global.properties` ファイルと `solution.properties` ファイルをマイグレーションする必要があります。

暗号鍵または鍵ストアの変更されたパスワード

サーバーは暗号鍵を保持する鍵ストアのパスワードと暗号鍵自体のパスワードを、サーバーの 150 ページの『Stash ファイル』から読み取ります。したがって、これらのパスワードのいずれかを変更する場合は、stash ファイルを更新する必要があります。そのためには、`createstash` ツールを使用します。

期限の切れた暗号化証明書

サーバーで公開鍵暗号化を使用している場合、暗号鍵ペアに関連付けられた証明書が期限切れになることがあります。このような場合には、『keytool を使用して証明書の有効期限を延長する』で説明されている手順に従うことで、証明書を更新することができます。この手順では、基礎となる鍵は保持されるため、既存の暗号化ファイルをマイグレーションする必要はありません。

第 10 章 IBM Security Directory Integrator サーバー API の構成

ここで提供される情報とリンクを使用して、IBM Security Directory Integrator サーバー API を構成することができます。

IBM Security Directory Integrator サーバー API には、IBM Security Directory Integrator ソリューションを開発したり、サーバーとローカルまたはリモートから対話するために使用するための一連のプログラミング呼び出しが用意されています。また、Java Management Extensions (JMX) インターフェースを介してサーバー API 呼び出しを公開する管理レイヤーも組み込まれています。このセクションでは、サーバー API を構成するために使用するプロパティーについて説明します。

- サーバー API の使用法については、「*リファレンス*」の『付録 C. サーバー API』を参照してください。
- サーバー API の構成方法については、123 ページの『サーバー API の構成』を参照してください。
- IBM Security Directory Integrator サーバーのセキュリティーについて詳細は、121 ページの『リモート・サーバー API』を参照してください。

サーバー ID

以下の情報を使用することで、サーバー ID とその動作について詳しく知ることができます。

リモート・クライアントは、サーバーを一意の ID を使用して識別できれば、対話相手のサーバーを識別できます。IBM Security Directory Integrator では、管理およびモニター・コンソール (AMC) などのリモート・クライアントが、異なる時間に同じ IP とポートを使用して、異なる IBM Security Directory Integrator サーバーに接続できるように、一意のサーバー ID を指定することができます。同じ IP とポートを使用して異なる時間に接続するには、IBM Security Directory Integrator クライアント・アプリケーションは、これらのクライアント・アプリケーションを異なるデータとデータベースが関連付けられる異なる IBM Security Directory Integrator サーバーとして登録できる必要があります。

ユーザーは、一意の ID を手動で割り当てて、どのリモート・クライアント (例えば、AMC) でも、IBM Security Directory Integrator サーバーの IP アドレス、ポート、および一意の ID に基づいて、IBM Security Directory Integrator サーバーに接続できるようにする必要があります。AMC では 1 つの IBM Security Directory Integrator サーバーが、誤って、または意図的に、複数回登録されることがないように、各サーバーに一意の ID を割り当てます。ID を手動で割り当てる場合、ユーザーは異なるサーバー IBM Security Directory Integrator には区別ができる ID を割り当てる必要があります。

一意のサーバー ID プロパティーは、`com.ibm.di.server.id` を使用して構成できます。特定のサーバーに一意のサーバー ID を割り当てるには、そのサーバーの

global.properties ファイル内または solution.properties ファイル内のこのプロパティーに一意の ID 文字列を設定します。com.ibm.di.server.id のデフォルト値はブランクです。

パスワードで保護された構成の例外

サーバー API は、パスワードを指定しないでパスワードで保護された構成の使用を試みると、例外をスローします。これについては、以下の情報を参照することで詳しく知ることができます。

IBM Security Directory Integrator サーバー API は、パスワードを指定しないでパスワードで保護された構成にアクセスを試みるクライアントがいると、サーバーの問題を検出して、処理することができます。ユーザーにその問題を通知するメッセージが表示されます。パスワードが入力されなかったり、入力したパスワードが間違っていた場合は、エラー・メッセージが表示されます。 292 ページの『AMC および暗号化された構成』を参照してください。

サーバー RMI

以下の情報とリンクを使用することで、サーバー RMI について詳しく知ることができます。

各 IBM Security Directory Integrator サーバーに対するリモート・アクセスのニーズが高まってきたため、リモート・メソッド呼び出し (RMI) はデフォルトで使用可能になっています。十分なセキュリティーを保証するため、デフォルトのリモート・アクセスではクライアント認証に Secure Socket Layer (SSL) を使用する必要があります。SSL アクセスを使用する際に、IBM Security Directory Integrator でデプロイされたサンプルの鍵ストアとトラストストアを使用すると便利です。 116 ページの『SSL クライアント認証』および 138 ページの『サーバー API 認証オプションの要約』を参照してください。

構成のロードのタイムアウト・インターバル

api.config.timeout プロパティーを使用して、タイムアウト・インターバルを追加します。

サーバー API が呼び出しを行うときに、構成インスタンスが構成ファイルを完全にはロードできない場合、サーバー API は NULL オブジェクトを返します。タイムアウト・インターバルを追加するには、global.properties ファイルに api.config.timeout プロパティーを追加します。構成ファイルをロードするインターバルには、デフォルトでは 2 分間に設定されています。構成ファイルをタイム・インターバル内にロードできないときは、例外がスローされます。

第 11 章 プロパティー

プロパティーを使用すると、IBM Security Directory Integrator コンポーネントと IBM Security Directory Integrator サーバーを構成することができます。

プロパティーはパラメーターの単純な **キーワード:値** ペアであり、構成ファイル (configs) の外部で、外部プロパティー・ファイルに保管されます。これにより、パスワードなどの機密情報を構成ファイルの外部に保管できます。`global.properties` ファイルが IBM Security Directory Integrator のメインの構成ファイルです。プロパティーは、`global.properties` ファイルまたは `solution.properties` ファイルで定義されます。`solutions.property` ファイルは `global.properties` ファイルの書き込み可能なコピーであり、サーバーがソリューション・ディレクトリーから始動されたときに使用されます。インストール時にインストール・ディレクトリーとは異なるソリューション・ディレクトリーを指定した場合、`global.properties` ファイルのコピー `solution.properties` が IBM Security Directory Integrator ソリューション・ディレクトリーに作成されます。両方のファイルはテキスト・ファイルで、プラットフォーム上で動作しているオペレーティング・システムが認識できるように作成されています。

プロパティーは、*true* や *5000* のようなパラメーター情報を持つ、単一値データ・コンテナーです。スクリプトから、`getProperty()` や `setProperty()` のような項目関数を使用して、プロパティーにアクセスできます。取得および設定のためのメソッドでは、直接プロパティー値を操作します。項目オブジェクトには、プロパティーも含まれます。属性と同様、プロパティーはデータ・コンテナーです。属性はデータの内容を保管するために使用されますが、プロパティーはパラメトリック情報を保管するために使用されます。プロパティー値と属性は、任意の型の Java オブジェクトです。プロパティーは、次の場所には表示されません。

- 属性マップ選択項目
- 作業項目リスト

プロパティーの操作

以下の情報を参照することで、ソリューション・ディレクトリーのさまざまな操作方法を理解できます。

このセクションでは、プロパティーを操作するときに必要になる基本的な概念について説明します。任意の `properties` ファイルに設定されるプロパティーが、そのコンピューター上のすべてのユーザーにとっての IBM Security Directory Integrator インストール済み環境全体のベースラインを形成します。ただし、ソリューション・ディレクトリーがインストール・ディレクトリーとは別に設定されている場合は、インストール・ディレクトリーにあるのと同じテキスト・ファイルのセットをソリューション・ディレクトリーに配置できます。これら任意のファイル内にリストされるプロパティーは、`global.properties` ファイルや `solution.properties` ファイルなどのグローバル・インストール・プロパティー・ファイル内のすべての値をオーバーライドします。また、構成ファイル内で設定する Java プロパティーの優

先順位が一番上で、グローバル・プロパティー・ファイルでの設定や、ソリューション・ディレクトリー内のプロパティー・ファイルの設定より優先されます。

ソリューション・ディレクトリーを指定する方法はいくつかあります。

- 構成エディターまたはサーバーを始動する前に、環境変数 TDI_SOLDIR を設定します。
- -s パラメーターを ibmditk スクリプトに指定して構成エディターを開始するか、または ibmdisrv スクリプトに指定して IBM Security Directory Integrator サーバーを開始します。この指定は、TDI_SOLDIR の設定より優先されます。TDI_SOLDIR がインストール・ディレクトリーと同じ場合、すべてのプロパティー・ファイルはそこから読み込まれます。そして、ソリューション・ディレクトリー内のプロパティー・ファイルに関する注釈は、適用されません。

それ以外の場合は、IBM Security Directory Integrator サーバーを初めて実行すると、ソリューション・ディレクトリーにすべてのプロパティー・ファイルのコピーが作成されます (これらのファイルが既に存在する場合は上書きされません)。その後は、インストール・ディレクトリーにあるプロパティー・ファイルに影響することなく、必要に応じてそれらのファイルを調整できます。インストール・ディレクトリーに残っているファイルは、引き続き他の IBM Security Directory Integrator インスタンスのベースライン構成となります。

注: global.properties ファイルは、ソリューション・ディレクトリーには solutions.properties という名前のファイルとしてコピーされます。Log4J.properties ファイルや、amc および serverapi フォルダー内のファイルなど、その他のファイルは元の名前のままコピーされます。

インストール・ディレクトリー内の元の global.properties ファイルのバックアップは、記録のために <Solution directory>/etc フォルダーにコピーされます。しかし単に記録されるだけであり、いかなる目的でも**使用されることはありません**。

プロパティーと tdimiggbl ツールを使用したマイグレーション

tdimiggbl ツールを使用すると、 global.properties ファイルを特定のバージョンの IBM Security Directory Integrator から上位のバージョンに移行することができます。次のセクション 67 ページの『第 5 章 マイグレーション』を参照してください。

グローバル・プロパティー

グローバル・プロパティーを使用することにより、インストール・ディレクトリーの etc フォルダーにある global.properties と呼ばれるファイルに保管される IBM Security Directory Integrator サーバー設定を構成できます。

このセクションでは、global.properties ファイルに含まれるすべてのプロパティーについて、デフォルト値をリストし、説明します。プロパティー・グループの先頭で、できる限りより詳しい資料への参照を示します。構成エディター (CE) (ibmditk) と IBM Security Directory Integrator サーバー (ibmdisrv) は、始動時に global.properties ファイルを読み込みます。このファイルは、ソリューション・ディレクトリーから solution.properties と呼ばれるファイルが読み込まれる前に読み込まれて、適用されます。

表 23. いくつかの重要な IBM Security Directory Integrator グローバル・プロパティー

プロパティーの用途	プロパティー	デフォルト値	説明
カスタム .jar または .zip ファイルを追加する	com.ibm.di.loader.userjars	c:¥myjars	Java プロパティー「path.separator (Linux の場合は「:」、Windows の場合は「;」)」で区切って、ディレクトリーまたは jar ファイルを指定します。TDI ローダーは、クラスおよびリソースを含む JAR ファイルについて、ディレクトリーを再帰的に検索します。検索対象は、.zip と .jar の拡張子を持つファイルのみです。
暗号を定義する	com.ibm.di.securityTransformation	DES/ECB/NoPadding	IBM Security Directory Integrator 構成のパスワード・ベースの暗号化または復号用の暗号を定義します。IBM Security Directory Integrator 7.0 で変更されました。
構成のオートロードを有効にする	com.ibm.di.server.autoload	autoload.tdi	「ibmdisrv -d」コマンドを使用して指定されたディレクトリー内で *.xml ファイルを検索します。-d で定義されたディレクトリー内で検出された各 *.xml ファイルを実行します。

ソリューション・プロパティー

ソリューション・プロパティーは、一般的に、グローバル・プロパティーをオーバーライドし、ソリューション・ディレクトリー内の solution.properties と呼ばれるファイルで保管されます。これについては、以下の情報を参照することで詳しく知ることができます。

solution.properties ファイルはデフォルトでは global.properties ファイルのコピーです。IBM Security Directory Integrator を構成する場合は solutions.properties ファイルを編集することをお勧めします。なぜなら、すべてのプロパティー・ファイルの中でこれが最後に読み込まれるからです。必要に応じて、solution.properties ファイル内のプロパティーを削除し、global.properties のデフォルトをオーバーライドするプロパティー構成ステートメントを追加することができます。

Java プロパティー

ここで提供する情報を参照して、Java プロパティーの詳細について理解できます。

Java プロパティーは変数であり、Java 仮想マシン (JVM) の設定です。 268 ページの『便利な JLOG パラメーター』に、Java ログ (Jlog) ファイルのプロパティーを示します。

注: 構成ファイル内で設定する Java プロパティーの優先順位が一番上で、グローバル・プロパティー・ファイルでの設定や、ソリューション・ディレクトリー内のプロパティー・ファイルの設定より優先されます。

表 24. Java プロパティー

プロパティー	デフォルト値	説明
javax.net.debug	なし	JSSE プロバイダーをデバッグ・モードに設定します。
com.ibm.di.javacmd	なし	Java インタープリターをオーバーライドします。

表 24. Java プロパティー (続き)

プロパティー	デフォルト値	説明
com.ibm.di.installdir	なし	構成エディターから AssemblyLine を始動する場合は、Java 実行可能ファイルへの、このパスを使用します。
com.ibm.di.jvmdir	TDI_root/jvm	IBM Security Directory Integrator が使用する JRE がインストールされているディレクトリー・パスを定義します。
com.ibm.di.server.maxThreadsRunning	500	このスレッド数を IBM Security Directory Integrator に設定します。効果があるためには、4 以上の値が設定される必要があります。
com.ibm.di.server.securemode	false	IBM Security Directory Integrator が実行されるモードを設定します (標準またはセキュア)
com.ibm.di.server.keystore	testserver.jks	サーバー SSL 証明書の鍵ストアの名前を指定します。IBM Security Directory Integrator 7.0 で名前が変更されました。
com.ibm.di.server.key.alias	server	サーバー SSL 証明書の鍵別名の名前を指定します。IBM Security Directory Integrator 7.0 で名前が変更されました。
{protect}-api.keystore.password	server (デフォルトで暗号化されています)	サーバー API の鍵ストアのパスワードを指定します。 IBM Security Directory Integrator 7.0 で追加されました。
{protect}-api.key.password		鍵のパスワードを指定します。指定しない場合、サーバーの鍵ストアのパスワードが使用されます。IBM Security Directory Integrator 7.0 で追加されました。
com.ibm.di.server.encryption.keystore	testserver.jks	サーバー暗号鍵の鍵ストアを指定します。IBM Security Directory Integrator 7.0 で追加されました。
com.ibm.di.server.encryption.key.alias	server	サーバー暗号鍵の鍵別名を指定します。IBM Security Directory Integrator 7.0 で追加されました。
com.ibm.di.server.encryption.keystoretype	jks	サーバーで暗号化のために使用される鍵をホストする鍵ストアのタイプを指定します。IBM Security Directory Integrator 7.0 で追加されました。
com.ibm.di.server.encryption.transformation	RSA	サーバーで暗号化のために使用される暗号変換の名前を指定します。「RSA」(公開鍵暗号化) または特定の共通鍵変換のいずれかを設定できます。IBM Security Directory Integrator 7.0 で追加されました。
com.ibm.di.server.fipsmode.on	false	IBM Security Directory Integrator で FIPS 標準を有効または無効にします。このプロパティーに true を設定すると、IBM Security Directory Integrator は FIPS 準拠モードで動作します。FIPS モードについての詳細は、IBM Security Directory Integrator 7.0 の追加機能を参照してください。
com.ibm.di.default.bind.address	*	IBM Security Directory Integrator サーバー全体 (コンポーネントおよびサーバー API) のデフォルトのバインド・アドレス。

システム・プロパティー

システム・プロパティーは、solution.properties のような外部プロパティー・ファイルに保管されるのではなく、システム・ストアに保管されます。以下の情報とリンクを参照することで、この内容について詳しく知ることができます。

一部のシステム・プロパティーと Java プロパティーは読み取り専用です。これらのシステム・プロパティーはそれぞれ対応するプロパティー・ストア (例えば、システム・ストア) に表示されます。これらの読み取り専用プロパティーの変更を試みても効果はありません。 219 ページの『第 12 章 システム・ストア』も参照してください。

第 12 章 システム・ストア

以下の情報を使用することで、システム・ストアとその動作について詳しく知ることができます。

IBM Security Directory Integrator では、密結合リレーショナル・データベースであるシステム・ストアによって永続ストレージ (JVM の再始動後も存続するオブジェクトのストレージ) がサポートされています。

システム・ストアをインプリメントするためにデフォルトでデプロイされる製品は、IBM Security Directory Integrator (旧称は Cloudscape) として知られる、Java で完全にインプリメントされたリレーショナル・データベースです。

システム・ストアは、以下のオブジェクトを保管します。

- デルタ・テーブル
- ユーザー・プロパティー・ストア
- パスワード・ストア

システム・ストアのデフォルトの場所

IBM Security Directory Integrator のネットワーク・モードのシステム・ストア・データベースのデフォルトの場所は、ソリューション・ディレクトリーです。したがって、ソリューション・ディレクトリーごとにシステム・ストアを保持できます。

すべてのソリューション・ディレクトリーにわたって 1 つのシステム・ストアを共有するには、 `global.properties` ファイルおよび `solution.properties` ファイル (既に作成されている場合) の `com.ibm.di.store.database` プロパティー内で実際の *`TDI_install_dir`* で `$soldir$` 値を置き換えます。

ソリューション・ディレクトリーを作成する場合、他のソリューション・ディレクトリーの設定との衝突を避けるために、 `solution.properties` ファイル内の以下のプロパティーを固有値で更新します。

- com.ibm.di.store.port=1527
- api.remote.naming.port=1099
- web.server.port=1098
- システム・キュー・ポートまたは `<soldir>/etc/activemq.xml` 内の Active MQ ポート

注:

以下の例では、複数のソリューション・ディレクトリーにわたって `com.ibm.di.store.port` プロパティーに同じ値を指定することの影響について説明しています。

同じ com.ibm.di.store.port の値 (1527、1527) および固有の api.remote.naming.port の値 (1099、41099) を持つ 2 つのソリューション・ディレクトリー (soldir1、soldir2) があります。

soldir1 でサーバーを始動すると、soldir1 内で、サーバーはポート 1099 で始動し、システム・ストアはポート 1527 で始動します。

soldir2 でサーバーを始動すると、サーバーはポート 41099 で始動し、soldir1 内のポート 1527 で既に listen 中のシステム・ストアに接続します。

このセクションの残りの部分では、IBM Security Directory Integrator を使用する際の運用面の問題について、特にシステム・ストアを保持するために IBM Security Directory Integrator を使用する場合に関連した点を説明します。

注: サード・パーティーの RDBMS に関しては、暗号化されたパスワード値を保持するために、それらを保持するフィールドのサイズを非常に大きくする必要がある場合があります。標準的な短いパスワードの場合、最大で 178 文字を使用する場合があります。これは、サーバーの鍵、および保管しようとしている暗号化されていないデータの長さ (バイト単位) に依存します。これはブロック化されたエンコード方式であるため、さらに大きいパスワードは同じスペース、あるいは 2 倍、3 倍の容量を使用する可能性があります。また、ブロックのサイズはサーバーの鍵に依存します。必要なサイズを確認する 1 つの方法として、最初に (保護された) パスワードをファイルに保管し、そのファイルを参照して何文字が使用されているかを確認します。

IBM Security Directory Integrator は、*組み込みモード*とネットワーク・モードのいずれかで実行します。IBM Security Directory Integrator は、デフォルトでは、global.properties ファイルで指定したとおり、*networked* モードで動作します。

V7.0 より前の IBM Security Directory Integrator リリースで使用されていたシステム・ストアは、*embedded* モードの IBM Security Directory Integrator (当時は、Cloudscape と呼ばれていました) でした。IBM Security Directory Integrator を組み込みモードで実行することには、欠点があります。組み込みモードの場合、IBM Security Directory Integrator は要求されると JVM 内の独立したスレッドとして動作します。組み込みモードでは IBM Security Directory Integrator の開始およびシャットダウンは自動的に行われます。ただし、このモードで実行すると、IBM Security Directory Integrator スレッドはデータベース・ファイルへの排他的アクセスを要求します。これが問題となるのは、それぞれが独自の IBM Security Directory Integrator スレッドを持つ別々の JVM が同じシステム・ストアにアクセスしようとした場合です。

組み込みモードの場合、これらのアクションによって、新しい独立した JVM が始動されるので、複数の JVM が同時にアクティブになったときにアクセスが競合する原因になります。

- IBM Security Directory Integrator サーバーの構成ファイルを使用したコマンド行呼び出しによって、1 つ以上の AssemblyLine の実行が引き起こされます。
- 構成エディター (GUI) の開始時
- 構成エディターからの AssemblyLine の開始時

上記のアクションは、いずれもそれ自体が IBM Security Directory Integrator スレッドを開始することはありません。ただし、システム・ストア内のいずれかのオブジェクトへのアクセスが必要な場合 (例えば、デルタ・テーブルおよびユーザー・プロパティー・ストアなどのシステム・ストアによってサポートされるオブジェクト) は、IBM Security Directory Integrator スレッドが開始されます。

前述のアクセス競合を解決するには、ネットワーク・モードで IBM Security Directory Integrator を実行します。これによって、システム・ストアへの同時アクセスが可能になります。また IBM Security Directory Integrator のユーザー認証を有効にすれば、ネットワーク・モードのセキュリティー上の問題も回避できます。データベース・レベルでセキュリティーを提供するため、IBM Security Directory Integrator は IBM Security Directory Integrator に対して BUILTIN セキュリティー・プロバイダーを使用します。BUILTIN によって正当なユーザーしか IBM Security Directory Integrator データベースにアクセスできないことが保証されます。IBM Security Directory Integrator がネットワーク・モードで構成されている場合は、システム・ストアとしてブートされた IBM Security Directory Integrator データベースの複数のインスタンスを操作することができます。また、1 つの IBM Security Directory Integrator インスタンスが特定の構成ファイル・インスタンスを使用するように構成することもできます。

注: IBM Security Directory Integrator の始動方法によっては、他のすべての IBM Security Directory Integrator プロセスが終了した後でも、IBM Security Directory Integrator のインスタンスがネットワーク・モードで動作を継続することがあります。

プロパティー `derby.drda.startNetworkSe–rver` に true を設定する (この場合は `global.properties` のデフォルト) と、IBM Security Directory Integrator を始動したときにネットワーク・サーバーが自動的に始動されます (この状況では、IBM Security Directory Integrator は組み込みドライバーがロードされたとき始動されます)。必要に応じて、IBM Security Directory Integrator を手動で終了させる必要があります。

Cloudscape コマンド行ユーティリティー

IBM Security Directory Integrator データベースを使用した作業の利便性を高めるため、以下の行を使用してスクリプト (`dbserver`) を作成することを検討してください (この例は Unix/Linux の場合です)。

```
export DB_JAR_DIR=jars/3rdparty/IBM
export DB_CLASSPATH=$DB_JAR_DIR/derby.jar:$DB_JAR_DIR/derbyclient.jar:¥
$DB_JAR_DIR/derbynet.jar:$DB_JAR_DIR/derbytools.jar
java -classpath $DB_CLASSPATH org.apache.derby.drda.NetworkServerControl "$@"
```

中間の 2 行は「¥」のところで 1 行に結合する必要があるかもしれません。

これと等価な Windows 用の dbserver.bat ファイルは次のとおりです。

```
set DB_JAR_DIR=jars/3rdparty/IBM
set DB_CLASSPATH=%DB_JAR_DIR%¥derby.jar;%DB_JAR_DIR%¥derbyclient.jar;¥
%DB_JAR_DIR%¥derbynet.jar;%DB_JAR_DIR%¥derbytools.jar;
java -classpath %DB_CLASSPATH% org.apache.derby.drda.NetworkServerControl %*
```

注: スクリプトは、作業ディレクトリーとして使用する IBM Security Directory Integrator のインストール・パスから開始する必要があります。これは、これに続くクラスパスがこのディレクトリーに対する相対パスであるためです。

以下は、このユーティリティー・スクリプトの使用例です。

```
Show all available commands:  ./dbserver

Start DBServer  ./dbserver start -p 1527

Stop DBServer  ./dbserver shutdown
```

dbserver スクリプトに指定できるサブコマンドで、IBM Security Directory Integrator に送信されるものの全リストを以下に示します。

- `start [-h <host>] [-p <portnumber>]`: 指定されたポート/ホストまたは localhost でネットワーク・サーバーを始動します。ホスト/ポートが指定されず、デフォルトをオーバーライドするプロパティーも設定されなかった場合、ポートは 1527 です。デフォルトでネットワーク・サーバーが listen するのは、それが動作しているマシンからの接続のみです。すべてのインターフェースを listen する場合は、-h 0.0.0.0 を使用します。また複数の IP を持つマシンの特定のインターフェースを listen する場合は、-h <hostname> を使用します。
- `shutdown [-h <host>][-p <portnumber>]`: 指定されたホストおよびポートまたはローカル・ホストのネットワーク・サーバーをシャットダウンします。ホストまたはポートが指定されなかった場合、ポート 1527 (デフォルト) のネットワーク・サーバーをシャットダウンします。
- `ping [-h <host>] [-p <portnumber>]`: ネットワーク・サーバーが稼働中かテストします。
- `sysinfo [-h <host>] [-p <portnumber>]`: ネットワーク・サーバー、JVM、および Cloudscape サーバーのクラスパスおよびバージョン情報を印刷します。
- `runtimeinfo [-h <host] [-p <portnumber]`: 稼働中のネットワーク・サーバーについて、セッション、スレッド、準備済みステートメント、およびメモリー使用状況などの拡張デバッグ情報を印刷します。
- `logconnections {on | off} [-h <host>] [-p <portnumber>]`: 接続および切断のロギングのオンまたはオフを切り替えます。接続および切断のログは、`derby.log` に記録されます。デフォルトはオフです。
- `maxthreads <max> [-h <host>][-p <portnumber>]`: 接続で使用できるスレッドの最大数を設定します。デフォルトは 0 (無制限) です。
- `timeslice <milliseconds> [-h <host>][-p <portnumber>]`: 各セッションが待機セッションに譲り渡すまで接続スレッドを所有していられる時間を設定します。デフォルトは 0 (譲り渡さない) です。
- `trace {on | off} [-s <session id>] [-h <host>] [-p <portnumber>]`: 指定されたセッションのデータ・トレースのオンまたはオフを切り替えます。セッションが指定されなかった場合はすべてのセッションが対象になります。デフォルトはオフです。
- `tracedirectory <tracedirectory> [-h <host>] [-p <portnumber>]`: 新規のトレース・ファイルを格納する場所を変更します。トレースが既にオンになっているセッションの場合、トレース・ファイルではそのまま同じ場所が使用されます。デフォルトは、`clousdcape.system.home` です。

ネットワーク・モードで実行する場合、IBM Security Directory Integrator インスタンスだけでなく、適切なドライバーを使用するその他のアプリケーションも、ネットワークを介して IBM Security Directory Integrator データベースに到達できます。そのようなアクセスに必要な信任状は `global.properties` ファイルに定義されてい

ます。この定義はご使用の各サイトのニーズに応じて調整が必要な場合があります。ユーザー名とパスワードのパラメーターについては、データの保全性とセキュリティーを決定付けるものであるため、特に注意が必要です。

IBM Security Directory Integrator の実行を占有モードとネットワーク・モードの間で切り替えることが多い場合は、「プロトタイプ」となる 2 種類の global.properties ファイルをファイル・システムに用意しておくことを検討してください。それぞれのファイルには、2 つのモードそれぞれに合わせた正しいパラメーター・セットを設定しておきます。サーバー・インスタンスを始動する直前に、必要に応じて適切な global.properties ファイルを所定の場所にコピーします。あるいは、個別のソリューション・ディレクトリー を使用します。ソリューション・ディレクトリーには solution.properties というファイルがあります。このファイルで定義されているプロパティーの値は、global.properties で定義されているシステム全体の対応プロパティー値よりも優先されます。

プロパティー・ストア

パスワード・ストアとユーザー・プロパティー・ストアは、システム・ストアの一種です。

パスワード・ストア

パスワード・ストア は、パスワード構文コンポーネントの値を変更した結果の値を保管する外部リポジトリーです。パスワード保護メカニズムは、ユーザーに提示される構成ウィンドウと直接関連しています。構成ウィンドウまたはフォームには、各パラメーターおよびその構文の説明が含まれます。 構文の 1 つのタイプにパスワード があり、これにより構成エディターで編集するためのパスワード・テキスト・フィールドが使用されます。このパスワードの外部リポジトリーは、構成エディターの「*プロパティー*」ページ (*パスワード・ストア*) で構成し、現行 IBM Security Directory Integrator ソリューションの構成ファイルで指定します。このようなプロパティー・ストアが構成されていない場合、パスワードは構成ファイルの平文内に保存されます。

デフォルトのパスワード・ストアが構成されている場合、protected/password パラメーターの初回保存時に固有のプロパティー名が生成されます。この鍵は、パスワード・ストアにおける鍵として使用されます。構成ファイルには、同じプロパティー名が標準プロパティー参照として書き込まれます。後で値を検索する際に、標準プロパティー解決が行われ、パスワード・ストアから実際の値が検索されます。

パスワード・ストアが指定されている場合、パスワードの固有キーが生成され、パスワードは、パスワード・ストアにそのキーの下に暗号化されて保存されます。構成ファイルでは、パスワードはそのキーによってのみ参照されます。

ユーザー・プロパティー・ストア

ユーザー・プロパティー・ストアは、キー値に関連付けられたシリアライズ Java オブジェクトを管理するために使用されるシステム・ストア・テーブルです。ここで、パーシスタント・コンポーネント・パラメーターとプロパティー (**イテレーター状態ストア**など) が、保管する他のデータとともに保守されます。システム・ストアは、IBM Security Directory Integrator コンポーネントの 3 種類のパーシスタン

ト・ストアの 1 つとしてユーザー・プロパティー・ストアをインプリメントします。プロパティー・ストアからプロパティーを選択するために使用する IBM Security Directory Integrator ユーザー・インターフェースについての詳細は、303 ページの『ソリューション・ビューの追加』を参照してください。

システム・ストアとしてのサード・パーティー RDBMS の使用

システム・ストアは、バンドルされた Apache Derby データベースを使用する代わりに、他のマルチユーザー RDBMS システムを使用するように構成することができます。

このように構成するには、適切な SQL データ定義言語 (DDL) ステートメントとドライバー・パラメーターを、`global.properties` または `solution.properties` にシステム・プロパティーとして指定します。 IBM DB2、Oracle、および MS SQL*Server のサポートされる構成用の、コメント化されたサンプル・ステートメントは、*`TDI_install_dir`*`/etc` ディレクトリー内のディストリビューション・バージョンの `global.properties` にあります。

IBM Security Directory Integrator サーバーの適切な文書にアクセスして、構成エディターに組み込まれた適切なテンプレートを利用することもできます。「サーバー」ペイン内の「サーバー」を右クリックして、「**システム・ストア設定の編集**」を選択します。ウィンドウ内の「**サーバーのシステム・ストア**」の見出しは、コンテキスト・メニューです。Derby Embedded、Derby Networked、Oracle、DB2、MS SQL*Server 2005+、および IBM solidDB の選択項目があります。

注: システム・ストアは構成エディター内にプロジェクトごとに構成することもできます。プロジェクトをエクスポートすると、これらの設定は構成ファイルで保管され、サーバーに定義されているシステム・ストアよりも優先します。

JDBC Driver のパラメーターにより、データベースのパスが指定されます。その他のプロパティーにより、IBM Security Directory Integrator がシステム・ストアで実行できる必要がある特定の操作の調整済み SQL が指定されます。プロパティーあたり複数の SQL ステートメントを指定できます。個別のステートメントの終わりには必ずセミコロンを指定してください。プロパティーの例を次に示します (本書では、読みやすくする目的でステートメントが複数の行に分割されていますが、実際のプロパティー・ファイルでは、1 つのプロパティーのステートメントはすべて 1 行で入力されています)。

```
com.ibm.di.store.create.delta.systable=CREATE TABLE {0} (ID VARCHAR(VARCHAR_LENGTH) NOT NULL,
    SEQUENCEID int, VERSION int);
    ALTER TABLE {0} ADD CONSTRAINT IDI_DS_{UNIQUE} PRIMARY KEY (ID)
```

{0} => テーブル名に置き換えられます。

{UNIQUE} => 現在のシステム時刻に基づいて固有名を生成するために使用できる特殊変数です。

サポートされている各 RDBMS システムの接続パラメーターとステートメントの例を以下のセクションに示します。

Oracle

Oracle を使用するには、JDBC ドライバー・クライアント・ライブラリー ojdbc14.jar を *TDI_install_dir*/jars ディレクトリーにドロップする必要があります。

JDBC 接続パラメーター

```
com.ibm.di.store.database=jdbc:oracle:thin:@itdidev.in.ibm.com:1521:itimdb
com.ibm.di.store.jdbc.driver=oracle.jdbc.OracleDriver
com.ibm.di.store.jdbc.urlprefix=jdbc:oracle:thin:
com.ibm.di.store.jdbc.user=SYSTEM
{protect}-com.ibm.di.store.jdbc.password=password
```

ここで、*itimdb* はシステム・ストアとして使用するデータベースの SID です。

表の作成ステートメント

```
com.ibm.di.store.create.delta.systable=CREATE TABLE {0} (ID VARCHAR(VARCHAR_LENGTH) NOT NULL,
 SEQUENCEID int, VERSION int);ALTER TABLE {0} ADD CONSTRAINT IDI_CS_{UNIQUE} PRIMARY KEY (ID)
com.ibm.di.store.create.delta.store=CREATE TABLE {0} (ID VARCHAR(VARCHAR_LENGTH) NOT NULL,
 SEQUENCEID int, ENTRY BLOB );ALTER TABLE {0} ADD CONSTRAINT IDI_DS_{UNIQUE} Primary Key (ID)
com.ibm.di.store.create.property.store=CREATE TABLE {0} (ID VARCHAR(VARCHAR_LENGTH) NOT NULL,
 ENTRY BLOB );ALTER TABLE {0} ADD CONSTRAINT IDI_PS_{UNIQUE} Primary Key (ID)
com.ibm.di.store.create.sandbox.store=CREATE TABLE {0} (ID VARCHAR(VARCHAR_LENGTH) NOT NULL,
 ENTRY BLOB )
com.ibm.di.store.create.recal.conops=CREATE TABLE {0} (METHOD varchar(VARCHAR_LENGTH), RESULT BLOB,
 ERROR BLOB)
```

MS SQL Server

MS SQL Server を使用するには、さまざまな Microsoft クライアント・ライブラリーを *TDI_install_dir*/jars ディレクトリーにインストールする必要があります。

JDBC 接続パラメーター

```
com.ibm.di.store.database=jdbc:Microsoft:sqlserver://localhost:1433;
     DatabaseName=master;selectMethod=cursor;
com.ibm.di.store.jdbc.driver=com.microsoft.jdbc.sqlserver.SQLServerDriver
com.ibm.di.store.jdbc.user=sa
com.ibm.di.store.jdbc.password=passw0rd
```

上記の接続パラメーターは、以下の Microsoft JDBC jar で使用されます。

1. Msutil.jar
2. MsBase.jar
3. MSsqlserver.jar

注: Microsoft SQL Server 2008 の場合、*TDI_install_dir*/jars ディレクトリーに配置されるドライバー jar ファイルは sqljdbc.jar (1 つのファイルのみ必要) であり、これは SQL Server 2008 インストール済み環境の <Microsoft SQL Server 2005-Install-Dir>/sqljdbc_<version>/<language>/sqljdbc.jar から取得できます。JDBC 接続パラメーターは以下のように指定する必要があります。

```
com.ibm.di.store.database=jdbc:sqlserver://localhost:1433;DatabaseName=name;selectMethod=cursor;
com.ibm.di.store.jdbc.driver=com.microsoft.sqlserver.jdbc.SQLServerDriver
com.ibm.di.store.jdbc.user=sa
com.ibm.di.store.jdbc.password=passw0rd
```

selectMethod プロパティーは jdbc URL ではオプションです。このプロパティーに「cursor」を設定すると、データベース・カーソルが作成されます。これはクライアントのメモリーには収容不可能な巨大な結果セットを読み込む場合に役立ちます。

selectMethod のデフォルトの動作は「cursor」ではなく「direct」です。この場合、結果セットがクライアントのメモリーに保持されるので、はるかに高速なパフォーマンスが提供されます。したがって、メモリーに問題さえなければ、デフォルトの「direct」の動作を選択することをお勧めします。詳細は、http://msdn.microsoft.com/en-us/library/ms378988(SQL.90).aspx を参照してください。

JDBC 接続パラメーター (JSQLConnect ドライバー)

```
com.ibm.di.store.database= jdbc:JSQLConnect://itdiderver/database=reqpro
com.ibm.di.store.jdbc.driver= com.jnetdirect.jsql.JSQLDriver
com.ibm.di.store.jdbc.urlprefix= jdbc:JSQLConnect:
com.ibm.di.store.jdbc.user=administrator
{protect}-com.ibm.di.store.jdbc.password=password
```

上記の接続パラメーターは、JSQLConnect ドライバーで使用されます。JSQLConnect.jar ファイルをダウンロードし、*TDI_install_dir*/jars ディレクトリーにコピーしてください。

表の作成ステートメント

MS SQL のデータ型は IMAGE です。

```
com.ibm.di.store.create.delta.systable=CREATE TABLE {0} (ID VARCHAR(VARCHAR_LENGTH) NOT NULL,
 SEQUENCEID int, VERSION int);
 ALTER TABLE {0} ADD CONSTRAINT IDI_MYCONSTRAINT_{UNIQUE} PRIMARY KEY (ID)
com.ibm.di.store.create.delta.store=CREATE TABLE {0} (ID VARCHAR(VARCHAR_LENGTH) NOT NULL,
 SEQUENCEID int, ENTRY IMAGE );
 ALTER TABLE {0} ADD CONSTRAINT IDI_DS_{UNIQUE} Primary Key (ID)
com.ibm.di.store.create.property.store=CREATE TABLE {0} (ID VARCHAR(VARCHAR_LENGTH) NOT NULL,
 ENTRY IMAGE );
 ALTER TABLE {0} ADD CONSTRAINT IDI_PS_{UNIQUE} Primary Key (ID)
com.ibm.di.store.create.sandbox.store=CREATE TABLE {0} (ID VARCHAR(VARCHAR_LENGTH) NOT NULL,
 ENTRY IMAGE)
com.ibm.di.store.create.recal.conops=CREATE TABLE {0} (METHOD varchar(VARCHAR_LENGTH),
 RESULT IMAGE, ERROR IMAGE)
```

IBM DB2

以下にリストされているパラメーターとステートメントを参照することで、DB2 の動作について理解できます。

JDBC 接続パラメーター

```
com.ibm.di.store.database=jdbc:db2:net://localhost:50000/ididb
com.ibm.di.store.jdbc.driver=com.ibm.db2.jcc.DB2Driver
com.ibm.di.store.jdbc.urlprefix= jdbc:db2:net:
com.ibm.di.store.jdbc.user=db2admin
{protect}-com.ibm.di.store.jdbc.password=db2admin
```

ここで、データベース URL の *ididb* は DB2 インスタンスの DSN です。

表の作成ステートメント

```
com.ibm.di.store.create.delta.systable=CREATE TABLE {0} (ID VARCHAR(VARCHAR_LENGTH) NOT NULL,
 SEQUENCEID int, VERSION int);
 ALTER TABLE {0} ADD CONSTRAINT IDI_MYCONSTRAINT_{UNIQUE} PRIMARY KEY (ID)
com.ibm.di.store.create.delta.store=CREATE TABLE {0} (ID VARCHAR(VARCHAR_LENGTH) NOT NULL,
 SEQUENCEID int, ENTRY BLOB );
 ALTER TABLE {0} ADD CONSTRAINT IDI_DS_{UNIQUE} Primary Key (ID)
com.ibm.di.store.create.property.store=CREATE TABLE {0} (ID VARCHAR(VARCHAR_LENGTH) NOT NULL,
 ENTRY BLOB ) ;ALTER TABLE {0} ADD CONSTRAINT IDI_PS_{UNIQUE} Primary Key (ID)
com.ibm.di.store.create.sandbox.store=CREATE TABLE {0} (ID VARCHAR(VARCHAR_LENGTH) NOT NULL,
 ENTRY BLOB )
```

IBM solidDB

IBMsolidDB® の場合、SolidDriver2.0.jar ファイルを *TDI_install_dir*/jars ディレクトリーに配置する必要があります。

この JAR は IBMsolidDB インストール済み環境の *SolidDB_install_dir*/jdbc/SolidDriver2.0.jar から取得できます。

JDBC 接続パラメーター

```
com.ibm.di.store.database=jdbc:solid://localhost:1315
com.ibm.di.store.jdbc.driver=solid.jdbc.SolidDriver
com.ibm.di.store.jdbc.urlprefix=jdbc:solid:
com.ibm.di.store.jdbc.user=dba
{protect}-com.ibm.di.store.jdbc.password=dba
```

表の作成ステートメント

```
com.ibm.di.store.create.delta.systable=CREATE TABLE {0} (ID VARCHAR(VARCHAR_LENGTH)
PRIMARY KEY NOT NULL, SEQUENCEID int, VERSION int)
com.ibm.di.store.create.delta.store=CREATE TABLE {0} (ID VARCHAR(VARCHAR_LENGTH)
PRIMARY KEY NOT NULL, SEQUENCEID int, ENTRY BLOB)
com.ibm.di.store.create.property.store=CREATE TABLE {0} (ID VARCHAR(VARCHAR_LENGTH)
PRIMARY KEY NOT NULL, ENTRY BLOB)
com.ibm.di.store.create.sandbox.store=CREATE TABLE {0} (ID VARCHAR(VARCHAR_LENGTH)
NOT NULL, ENTRY BLOB)
com.ibm.di.store.create.recal.conops=CREATE TABLE {0} (METHOD VARCHAR(VARCHAR_LENGTH),
 RESULT BLOB, ERROR BLOB)
```

Derby を使用してシステム・ストアを保持する

Derby を使用して、システム・ストアを保持することができます。

このセクションの残りの部分では、Derby を使用する際の運用面の問題について、特にシステム・ストアを保持するために Derby を使用する場合に関連した点を説明します。

注: サード・パーティーの RDBMS に関しては、暗号化されたパスワード値を保持するために、それらを保持するフィールドのサイズを非常に大きくする必要がある場合があります。標準的な短いパスワードの場合、最大で 178 文字を使用する場合があります。これは、サーバーの鍵、および保管しようとしている暗号化されていないデータの長さ (バイト単位) に依存します。これはブロック化されたエンコード方式であるため、さらに大きいパスワードは同じスペース、あるいは 2 倍、3 倍の容量を使用する可能性があります。また、ブロックのサイズはサーバーの鍵に依存します。必要なサイズを確認する 1 つの方法として、最初に (保護された) パスワードをファイルに保管し、そのファイルを参照して何文字が使用されているかを確認します。
Derby は、*組み込みモード*とネットワーク・モードのいずれかで実行します。Derby は、デフォルトでは、`global.properties` ファイルで指定したとおり、*networked* モードで動作します。

V7.0 より前の IBM Security Directory Integrator リリースで使用されていたシステム・ストアは、embedded モードの Derby (当時は、Cloudscape と呼ばれていました) でした。Derby を組み込みモードで実行することには、欠点があります。組み込みモードの場合、Derby は要求されると JVM 内の独立したスレッドとして動作します。組み込みモードでは Derby の開始およびシャットダウンは自動的に行われます。ただし、このモードで実行すると、Derby スレッドはデータベース・ファイルへの排他的アクセスを要求します。これが問題となるのは、それぞれが独自の Derby スレッドを持つ別々の JVM が同じシステム・ストアにアクセスしようとした場合です。

組み込みモードの場合、これらのアクションによって、新しい独立した JVM が始動されるので、複数の JVM が同時にアクティブになったときにアクセスが競合する原因になります。

- IBM Security Directory Integrator サーバーの構成ファイルを使用したコマンド行呼び出しによって、1 つ以上の AssemblyLine の実行が引き起こされます。
- 構成エディター (GUI) の開始時
- 構成エディターからの AssemblyLine の開始時

上記のアクションは、いずれもそれ自体が Derby スレッドを開始することはありません。ただし、システム・ストア内のいずれかのオブジェクトへのアクセスが必要な場合 (例えば、デルタ・テーブルおよびユーザー・プロパティー・ストアなどのシステム・ストアによってサポートされるオブジェクト) は、Derby スレッドが開始されます。

前述のアクセス競合を解決するには、ネットワーク・モードで Derby を実行します。これによって、システム・ストアへの同時アクセスが可能になります。また Derby のユーザー認証を有効にすれば、ネットワーク・モードのセキュリティー上の課題も解決できます。データベース・レベルでセキュリティーを提供するため、IBM Security Directory Integrator は Derby に対して BUILTIN セキュリティー・プロバイダーを使用します。BUILTIN によって正当なユーザーしか Derby データベースにアクセスできないことが保証されます。ネットワーク・モードで構成したときには、システム・ストアとしてブートされた Derby データベースの複数のインスタンスを操作することができます。また、1 つの Derby インスタンスが特定の構成ファイル・インスタンスを使用するように構成することもできます。

注: Derby の始動方法によっては、他のすべての IBM Security Directory Integrator プロセスが終了した後でも、Derby のインスタンスがネットワーク・モードで動作を継続することがあります。プロパティー `derby.drda.startNetworkSe-rver` に true を設定する (この場合は `global.properties` のデフォルト) と、Derby を始動したときにネットワーク・サーバーが自動的に始動されます (この状況では、Derby は組み込みドライバーがロードされたとき始動されます)。必要に応じて、Derby を手動で終了させる必要があります。

Apache Derby インスタンスの構成

以下の情報を使用して、Apache Derby インスタンスの構成について学習できます。

複数の Derby インスタンスを構成して管理したり、Derby サーバーをネットワーク・モードで始動、停止、再始動する機能を提供するため、プロジェクトの「**ソリューションのロギングおよび設定**」構成の一部として、IBM Security Directory Integrator 構成エディターには「**システム・ストア**」というメニュー・オプションが用意されました。ここで説明する構成オプションの多くは、旧バージョンの IBM Security Directory Integrator では構成の基本だった `global.properties` ファイルの値を、デフォルト値として採用しています。

「**システム・ストア**」メニュー・オプションには、バックエンド RDBMS として IBM DB2 のような他のデータベースを使用するようにシステム・ストアを構成する手段も用意されています。詳細については、「*Directory Integrator の構成*」の「**構成エディター -> ソリューションのロギングおよび設定**」の下の「システム・ストアの設定」を参照してください。

Apache Derby をネットワーク・モードで始動する

Apache Derby をネットワーク・モードで始動するには、ここに記載した手順に従ってください。

com.ibm.di.store.hostname プロパティーに localhost を設定すると、リモート接続が禁止されます。com.ibm.di.store.hostname プロパティーに、IBM Security Directory Integrator を実行しているローカル・コンピューターの IP アドレスを設定すると、リモート・クライアントは、その IP アドレスを使用してこの Derby インスタンスにアクセスできます。ローカル・コンピューターに対するネットワーク・サーバーの始動しかできません。

表 25. Apache Derby をネットワーク・モードで始動する

プロパティー	デフォルト値	説明
com.ibm.di.store.start.mode	automatic	要求されたとき、Derby サーバー・プロセスを始動するモード。automatic または manual を設定します。
com.ibm.di.store.hostname	localhost	Derby サーバーの URL。
com.ibm.di.store.port	1527	Derby サーバーに接続するポート。
com.ibm.di.store.sysibm	true	SYSIBM スキーマを使用するかしないかの状態。値は、true または false です。
com.ibm.di.store.varchar.length	512	システム・ストアおよびシステム・ストア (PES) コネクター・テーブルで使用される ID 列の varchar(length)。
com.ibm.di.store.database	jdbc:derby://localhost:1527/$soldir$/TDISysStore;create=true	システム・ストア・データベースのデフォルトの場所としてソリューション・ディレクトリーを設定します。 **注:** ソリューション・ディレクトリーの絶対パスで $soldir$ 値を置き換えないでください。パスは、JVM 内で実行時に自動的に更新されます

システム・ストアでユーザー認証を有効にする

以下のプロパティーは、global.properties ファイルのシステム・ストアのネットワーク・モード・プロパティーの後に追加することができます。

表 26. システム・ストアでユーザー認証を有効にする

プロパティー	デフォルト値	説明
derby.connection.requireAuthentication	true	システム・ストアでユーザー認証を有効にします。
derby.authentication.provider	BUILTIN	ユーザー認証プロバイダーに BUILTIN を設定します。これは Derby に付属しているプロバイダーの中で最も基本的で単純な認証プロバイダーです。
derby.database.defaultConnectionMode	fullAccess	システム・ストア・ユーザーのアクセス・レベルを定義します。Derby がサポートするアクセス・レベルは、「fullAccess」、「readOnly」、「noAccess」です。

システム・ストア・テーブル用のステートメントを作成する

以下のリスト項目用にテーブル作成 SQL ステートメントを構成することができます。

- デルタ・システム・テーブル
- デルタ・テーブル
- プロパティー・テーブル
- サンドボックス・テーブル
- レコード AssemblyLine テーブル
- トゥームストーン・マネージャー・テーブル

- ibmsnap_commitseq 列名

表 27. システム・ストア用のステートメントを作成する

プロパティー	デフォルト値	説明
com.ibm.di.store.create.delta.systable	CREATE TABLE {0} (ID VARCHAR(VARCHAR_LENGTH) NOT NULL, SEQUENCEID int, VERSION int); ALTER TABLE {0} ADD CONSTRAINT IDI_CS_{UNIQUE} PRIMARY KEY (ID)	デルタ・システム・テーブル用のテーブル作成 SQL ステートメント。
com.ibm.di.store.create.delta.store	CREATE TABLE {0} (ID VARCHAR(VARCHAR_LENGTH) NOT NULL, SEQUENCEID int, ENTRY BLOB); ALTER TABLE {0} ADD CONSTRAINT IDI_DS_{UNIQUE} Primary Key (ID)	デルタ・テーブル用のテーブル作成 SQL ステートメント。
com.ibm.di.store.create.property.store	CREATE TABLE {0} (ID VARCHAR(VARCHAR_LENGTH) NOT NULL, ENTRY BLOB); ALTER TABLE {0} ADD CONSTRAINT IDI_PS_{UNIQUE} Primary Key (ID)	プロパティー・テーブル用のテーブル作成 SQL ステートメント。
com.ibm.di.store.create.sandbox.store	CREATE TABLE {0} (ID VARCHAR(VARCHAR_LENGTH) NOT NULL, ENTRY BLOB)	サンドボックス・テーブル用のテーブル作成 SQL ステートメント。
com.ibm.di.store.create.recal.conops	CREATE TABLE {0} (METHOD varchar (VARCHAR_LENGTH), RESULT BLOB, ERROR BLOB)	レコード AL 用のテーブル作成 SQL ステートメント。
com.ibm.di.store.create.tombstones	CREATE TABLE IDI_TOMBSTONE (ID INT GENERATED ALWAYS AS IDENTITY, COMPONENT_TYPE_ID INT, EVENT_TYPE_ID INT, START_TIME TIMESTAMP, CREATED_ON TIMESTAMP, COMPONENT_NAME VARCHAR(1024), CONFIGURATION VARCHAR(1024), EXIT_CODE INT, ERROR_DESCR VARCHAR(1024), STATS LONG VARCHAR FOR BIT DATA, GUID VARCHAR(1024) NOT NULL, USER_MESSAGE VARCHAR(1024), UNIQUE (ID, GUID))	トゥームストーン・マネージャー・テーブル作成用の SQL ステートメントを指定します。テーブル名とフィールド名は変更しないでください。

表 27. システム・ストア用のステートメントを作成する (続き)

プロパティー	デフォルト値	説明
com.ibm.di.conn.rdbmschlog.cdcolname	ibmsnap_commitseq	RDBMS 変更ログ・コネクターで使用される ibmsnap_commitseq 列の名前を指定します。

Apache Derby データベースのバックアップ

ここで提供される方法を使用して、Apache Derby データベースのバックアップを取ることができます。

考慮する必要がある別の問題として、Derby データベースに含まれているデータの**バックアップ**があります。推奨される (そして最も簡単な) 方法を、以下に示します。

- Derby データベースをシャットダウンします (組み込みモードで実行されている場合は、すべての IBM Security Directory Integrator インスタンスおよびすべての構成エディター・インスタンスをシャットダウンします)。
- IBM Security Directory Integrator ホーム・ディレクトリー内の Derby ディレクトリー (または、global.properties ファイルが指している Derby ディレクトリー) 全体を別の場所にコピーし、そのデータの安全を確保します。
- Derby データベースを再始動します (ネットワーク・モードで実行している場合)。

データベースを復元するには、上記のステップにおけるコピー操作の複写元と宛先を逆にして実行します。

Apache Derby に関する問題のトラブルシューティング

このセクションは、Derby の総合的なトラブルシューティング・ガイドラインを提供することが目的ではなく、IBM Security Directory Integrator における基礎データベースとしての Derby の使用に関連して発生することのある多くの症状を示しています。

これらの属性を以下に示します。

「スキーマ 'SYSIBM' が存在しません (Schema 'SYSIBM' does not exist)」エラー

質問

ネットワーク・モードで Derby を使用しようとして、問題が発生しました。始動方法を理解し、sysinfo および testconnection を使用して照会することはできますが、 IBM Security Directory Integrator を実行し、システム・ストアをオープンしようとすると、以下のようなエラーが発生します。

```
[com.ibm.db2.jcc.a.SQLException: Schema 'SYSIBM' does not exist]
```

これはどのようにして修正できますか。

説明

このエラーが発生するのは、組み込みモードで作成されたデータベースを、 -ld フラグを使用してサーバーを始動することなく、ネットワーク・モード・サーバーでブートしようとしているためです。ネットワーク・モードの Derby サーバーで組み込みモードのデータベースをオープンするには、SYSIBM スキーマをロードする必要があることに注意してください。SYSIBM スキーマは、Derby サーバーによってロードされる特殊スキーマです。SYSIBM には、メタデータ情報を判断するための結果セットを戻す保管された準備済みステートメントが含まれます。

修正アクション

この問題を解決するには、次のように「-ld」フラグを使用して Derby ネットワーク・サーバーを始動します。

```
./dbserver start -p 1527 -ld
```

「Derby の別のインスタンスが既にブートされている可能性があります (Another Instance of Derby may already be booted)」

特に Derby を組み込みモードで使用している際に、以下のエラーを受け取る場合があります。

```
[ERROR XSDB6: Another instance of Derby may have already booted the database D:¥tdi60¥Derby.]
```

説明

Derby は、Derby の 2 つのインスタンスが同じデータベース (この場合 `D:¥tdi60¥Derby`) をブートすることを防止しようとします。これは、組み込みモードで実行中の同じ Derby データベースを更新しようとする 2 つの AssemblyLine を実行している場合に発生する可能性があります。このエラーはまた、データベースに対するクローズされていない接続がある場合にも発生する可能性があります。

修正アクション

2 つの AssemblyLine が同じ Derby データベースを更新できるようにする場合、Derby の正しいモードはネットワーク・モードです。このモードの操作には、その制限がありません。

これの回避策は、「**サーバー・ストアのブラウズ**」オプションを使用して、「**クローズ**」ボタンをクリックすることによって、データベースを閉じることです。データベースを開いていなかった場合でも、「**サーバー・ストアのブラウズ**」オプションを使用して、単純に開いて閉じるだけでこの問題を解決できます。

IBM Security Directory Integrator の将来のバージョンでは、この状況が自動的に処理され、必要に応じて Derby が停止および開始されるようにする予定です。

システム・ストアとして DB2 を使用できますか。

IBM Security Directory Integrator では、バンドルされている Derby データベース・システムの代わりに、DB2 をシステム・ストアとして使用できます。ただし、これを正しく機能させるには、システム・プロパティー・ファイルの変更が必要になります。 Derby ネッ

トワーク・モードのセクションは、次のようなセクションで置き換える必要があります (インストール済み環境に応じて適切なパラメーターを挿入してください)。

デフォルトの global.properties ファイルには、システム・ストアを使用し、セットアップするためのいくつかの CREATE_TABLE ステートメントがあります。正しい構文を使用すると、Derby 以外のデータベースをシステム・ストアとして使用できます。 以下に DB2 の構文を示します。

```
## Location of the DB2 database (networked mode)
com.ibm.di.store.database=jdbc:db2://168.199.48.4:3700/tdidb
com.ibm.di.store.jdbc.driver=com.ibm.db2.jcc.DB2Driver
com.ibm.di.store.jdbc.urlprefix=jdbc:db2:
com.ibm.di.store.jdbc.user=db2inst1
com.ibm.di.store.jdbc.password=******
com.ibm.di.store.start.mode=automatic
com.ibm.di.store.port=3700
com.ibm.di.store.sysibm=true

# the varchar(length) for the ID columns used in system store and PES Connector tables
com.ibm.di.store.varchar.length=512

# create statements for DB2 system store tables
com.ibm.di.store.create.delta.systable=CREATE TABLE {0} (ID VARCHAR(VARCHAR_LENGTH)
 NOT NULL, SEQUENCEID int, VERSION int)
com.ibm.di.store.create.delta.store=CREATE TABLE {0} (ID VARCHAR(VARCHAR_LENGTH)
 NOT NULL, SEQUENCEID int, ENTRY BLOB )
com.ibm.di.store.create.property.store=CREATE TABLE {0} (ID VARCHAR(VARCHAR_LENGTH)
  NOT NULL, ENTRY BLOB )
com.ibm.di.store.create.sandbox.store=CREATE TABLE {0} (ID VARCHAR(VARCHAR_LENGTH)
 NOT NULL, ENTRY BLOB )
```

注: 各 com.ibm.di.store.create.xxx ステートメントはそれぞれ 1 行で指定する必要があります (この例においては図示するために分割されています)。

Derby ネットワーク・サーバーに対してリモート接続できないのはなぜですか。

これは、Derby サーバーが、ホスト名として「localhost」を渡して始動されたためである可能性があります。これにより Derby へのリモート接続は許可されなくなります。Derby サーバーを停止し、ホスト名パラメーターにコンピューターの IP アドレスを指定して始動してください。そのためには、構成エディターの「**サーバーのシステム・ストア**」サーバー設定ウィンドウに移動します (「サーバー」ビューのサーバーのコンテキスト・メニューから移動できます)。

詳しくは、http://db.apache.org/derby/docs/10.5/adminguide/tadmincbdjhhfd.html を参照してください。

第 13 章 コマンド行オプション

コマンド行オプションの直後にその値を記述する必要があります。 オプションと値の間にスペースを入れないでください。

次のオプションがあります。

- 『構成エディター』
- 236 ページの『サーバー』
- 240 ページの『コマンド行インターフェース - tdisrvctl ユーティリティー』

構成エディター

以下の情報を使用することで、構成エディターおよび必要なプロパティーについて詳しく知ることができます。

CE を起動するには、ibmditk ラッパー・スクリプトを使用します。このスクリプトは、Java VM および IBM Security Directory Integrator のインストール・ロケーション・プロパティーの適切な設定 (現在の CE を起動するには両方が必須) を使用して、IBM Security Directory Integrator の Eclipse ランチャー (ce/eclipsece/miadmin) を起動します。

Eclipse ランチャー (ce/eclipsece/miadmin) は、独自のコマンド行パラメーターが使用される標準の Eclipse ランチャーです。Eclipse コマンド行オプションについての詳細は、『Eclipse のコマンド行オプション』を参照してください。

```
"%TDI_HOME_DIR%¥ce¥eclipsece¥miadmin" -vm "%TDI_JAVA_BIN_DIR%¥javaw" -vmargs
-Dcom.ibm.di.loader.IDILoader.path="%TDI_HOME_DIR%" %*
```

上記は、CE で必要となる 2 つの必須パラメーター (eclipse コマンド行パラメーター) を示す ibmditk スクリプトのフラグメントです。

使用するワークスペースの場所を指定するコマンド行オプション「-data」には注意してください。CE の複数のインスタンスの実行を計画している場合、ワークスペースは各インスタンスでロックされるため、CE の各インスタンスには異なるワークスペースを指定する必要があります。例:

```
ibmditk –data c:/instance1_workspace
```

上記のコマンドは、c:/instance1_workspace をワークスペースの場所として CE を起動します。

サーバー・シャットダウン・オプション

このコマンド行オプションは、CE のインストール・ディレクトリーと同じディレクトリーを使用している、すべての稼働中のサーバーの停止を試みます。コマンド行でこのオプションを以下のように指定すると、

```
ibmditk -tdishutdown
```

CE は始動された後、IBM Security Directory Integrator サーバー・プロジェクト内の各定義済みサーバーを検索し、CE のインストール・ディレクトリーと同じディレク

トリーを使用していないサーバーをフィルタリングし、それらの停止を試行します。完了すると、CE は Java VM をゼロの終了コードで終了させます。CE が停止を試みたサーバーが実際に停止したかどうかの保証はありません。サーバーによっては CE がこのコマンドを完了してもなお動作を継続しているものもあります。またサーバーによっては何らかの理由で停止を拒否するものもあります。

パースペクティブ・オプション

このコマンド行オプションは、CE に対して、代替パースペクティブで開始することを指示します。現在デフォルトのパースペクティブ以外の唯一のパースペクティブは Easy ETL パースペクティブであり、以下のオプションを使用して Easy ETL パースペクティブで CE を開始します。

```
ibmditk -perspective com.ibm.tdi.rcp.perspective.etl
```

サーバー

以下の情報を使用することで、サーバーでの作業について学習できます。

以下のコマンド行オプションは IBM Security Directory Integrator サーバー用です (`ibmdisrv [options]`)。

例:

```
ibmdisrv -c"C:¥demos¥rs.xml" -r"Access2LDAP" -l"c:¥metamerge¥mydemo.log"
```

注:

1. オプション文字と値の間にスペースはありません。値の中のスペースまたはコンマが原因で問題が発生しないようにするには、引用符を使用します。
2. Windows シェル・エグゼクティブを使用すると、以下のリストのうち最大 9 個の引数が使用できます。他のプラットフォームでは、制限はありません。
3. 構成ファイル名にはコンマ (,) を使用しないでください。

-s <dir>

ソリューションを配置する作業ディレクトリーを指定します。このディレクトリーは、ソリューション・ディレクトリーとして知られます。IBM Security Directory Integrator や構成ファイルなどの相対ファイル参照は、すべてこの場所からの相対位置です。最初に指定されるパラメーターでなければなりません。

指定したディレクトリーが存在しない場合は、IBM Security Directory Integrator サーバーによって作成され、ユーザーが必要に応じてカスタマイズできるプロパティー・ファイル (インストール・ディレクトリー内のプロパティー・ファイルがベース) が格納されます。詳しくは、389 ページの『付録 A. プロパティー・ファイルの例』を参照してください。

-c <file...>

構成ファイルです。このオプションを指定しないと、「自動始動」フォルダー内の項目がロードされて開始されます (**-D** の指定により抑制されていない場合)。`*.xml` のようなワイルドカードも許可されます。

注: 複数の構成ファイルを提供することが許されているのは、-d オプションも指定した場合のみです。

-n <encoding>

構成ファイルの書き込みに使用するエンコードです。これは、Java2 で有効な文字セット ID でなければなりません。すべての値リストについては、IANA Charset Registry (http://www.iana.org/assignments/character-sets) を参照してください。Java2 では、この値のサブセットのみをサポートします。

-r <al...>

開始する AssemblyLine 名のリストです。AssemblyLine **a** および **b** を開始するには、コマンド **-r a b** を使用します。他の構文もサポートされています **(-ra,b**、**-ra -rb**)。

注: インクルードとネーム・スペースを使用した場合の AssemblyLine は myNamespace:/AssemblyLines/alName のようになります (ネーム・スペースが **myNamespace**、AssemblyLine 名が **alName** であると想定します)。

-T<name>

<Tivoli_Common_Dir>/TDI/logs/ ディレクトリーの trace<name>.log ファイルに対する JLOG スタイルのトレースを使用可能にします。デフォルトは、メモリーへのトレースです (未処理の例外の場合に、JFFDC のトレースバック・ルーチンによって取得できる)。

-D 「自動始動」フォルダー内の項目、あるいはその両方の開始を使用不可にするためのフラグです。

-w -r (または -t) およびこのフラグを両方とも指定した場合、IBM Security Directory Integrator は、各 AssemblyLine の完了を待機してから次を開始します。このフラグを指定しない場合、IBM Security Directory Integrator は、-r パラメーターによって指定されたすべての AssemblyLine を並列に開始します。最後の AssemblyLine が完了すると、サーバーが停止します。

-e このオプションを指定すると、サーバーを保護モードで実行できます。サーバーの API レジストリーと同様、このサーバー固有のマスター・パスワードを使用して、すべての構成ファイルの復号と暗号化が行われます。

-v バージョン情報を表示して終了します。これは、ログ・ファイルのみに記録されます。

-P <password>

1 つまたは複数の構成ファイルが暗号化されている場合のパスワードです。

-p 始動時のダンプ Java プロパティーです。構成ファイルは Java プロパティーのダンプ前に読み取られるため、このフラグを指定した場合でも構成ファイルを提供する必要があります。

-d このシステム上で「デーモン」または*構成インスタンス*を開始します。

-d を指定して開始した場合、開始されるのは 1 つの匿名インスタンス (デーモン) のみです。その後このデーモンは、コマンド行で指定された各構成ファイルそれぞれの構成インスタンスを開始します。これにより、複数の構成インスタンスを動的に開始できます。コマンド行には、0 個以上の構成ファイルを指定できます。AssemblyLine をこのモードで実行するように指定することは無意味です。AL がどの構成ファイルに対応するのかを指定できないからです。ただし、AssemblyLine は自動始動を指定する構成インスタンスに属しているため、AssemblyLine は自動始動できます。

-d を指定しないで開始した場合、コマンド行で指定した構成ファイルをロードする構成インスタンスのみが開始されます。コマンド行で構成ファイルは 1 つしか指定できません (複数の構成ファイルを使用する必要がある場合は、パイプを使用して標準入力に指定します)。このモードでは、いくつでも実行する AssemblyLine を指定できます。これがサーバーを実行するための従来からの方法です。

-q 1 つの引数、モードを取ります。Mode=1 は、記録モードで実行、mode=2 はプレイバック・モードで実行することを意味しています。

-l <file>
ログ・ファイル (デフォルト・コンソール出力)。少数のメッセージのみがコンソールに出力されます。多くの情報をロギングするためにログ・ファイルを変更するには、`log4j.properties` を変更します。

-R `global.properties` の設定にかかわらず、リモート API を使用不可にします。

-W すべての構成ファイルを同一スレッド内で開始し、終了させずに永続的に待機させます。

-M シミュレーション・モードでの AssemblyLine を開始します。

-S このオプションは、構成エディターとサーバーとの間の通信のみに内部的に使用されます。構成ファイルの受け渡しに使用されます。このオプションを独自に使用しないでください。

-f *extProp1=file1, extProp2=file2*
ここで、*extProp* は外部プロパティー・ストアの名前です。*file* はプロパティーを読み取る対象を指定します。このオプションは、IBM Security Directory Integrator サーバーの開始時に入力できる、ユーザー定義の外部プロパティー・ストアを指定します。このオプションのコマンド行パラメーター **-f** は、「ibmdisrv」サーバー始動スクリプトで使用できます。*extProp* は外部プロパティー・ストアの名前です。*file* はプロパティーを読み取る対象を指定します。コマンド行からプロパティー・ファイルを指定するために **-f** オプションを使用する場合、サーバーは、メモリー内でのみプロパティー・ストア構成を変更します。つまり、サーバーは、ディスク上の IBM Security Directory Integrator 構成ファイルを変更して、その変更を永久に適用することはありません。この変更は IBM Security Directory Integrator サーバーの現行の実行中のみ有効になります。

プロパティー・ファイルがコマンド行で指定される場合、それらのファイルは **-c** コマンド行オプションで指定された構成インスタンス (IBM Security Directory Integrator サーバーの始動時にロードされる) に対してのみ有効です。コマンド行で指定されたプロパティー・ファイルは、**-c** コマンド行オプションを使用して明示的に指定されていない構成インスタンス (リモート・サーバー API クライアントによってロードされる構成インスタンスなど) には何ら影響しません。

-f コマンド行スイッチで指定された名前のプロパティー・ストアが構成インスタンスで検出できない場合、エラー・メッセージがサーバー・ログ (インストール・ディレクトリーに含まれる ibmdi.log) に記録されます。プロパティー・ストア名が **-f** コマンド行スイッチを使用して複数指定された場合は、次の 2 つの影響があります。(1) 警告メッセージがログに記録される。

(2) 最後に指定されたファイルが有効になる。この機能は、com.ibm.di.server.RS Java クラス (スクリプト記述時に main 変数によって参照される) にインプリメントされています。reload() メソッドが呼び出されると、MetamergeConfig オブジェクトがロードされ、コマンド行で指定された各プロパティー・ストアについて、対応する PropertyStoreConfig オブジェクトが更新されます。

注:

構成オブジェクト (AL、コネクター、FC など) のコピー・アンド・ペーストは完全にサポートされています。AL およびコンポーネントを簡単にコピーして、他の構成に貼り付けることができます。また、選択した項目の IBM Security Directory Integrator 構成の XML 定義でコピー・バッファーが満たされているため、IM チャット、E メール、テキスト・ファイルを使用してこれらを交換することもできます。これにより、データを単純かつ容易に受け渡すことができるため、サポートやオンライン支援 (ICT/NotesBuddy やフォーラムなど) において非常に役立つツールです。

注:
コピー・コマンドでは必ず、開始タグと終了タグを含む、<MetamergeConfig> ノード全体を選択してください。

-i このオプションは、IBM Security Directory Integrator サーバーが `global.properties` ファイルからのプロパティーを無視し、`solution.properties` ファイルのみ読み取ることを指定します。このオプションは `global.properties` ファイルが読み取り不能である場合に使用できます (例えば、IBM Security Directory Integrator サーバー始動時のエンコードが `global.properties` のエンコードと異なる場合)。

-? すべてのオプションを簡潔に示す*使用法* メッセージを出力します。

-j <file>
このオプションは、指定したファイルからリグレッション情報を読み取るために使用し、 AssemblyLine によって生成された詳細と比較されます。相違がある場合は、警告メッセージがログ・ファイルに書き込まれます。このオプションは、一つの AssemblyLine を実行する場合にのみ役立ちます。

-J<file>
このオプションは、指定したファイルに AssemblyLine のリグレッション情報を書き込むために使用します。

-k<file>
このオプションは、リグレッション情報を読み取る際に作業項目を無視するために使用します。

注:

1. プロパティーがコネクター、パーサー、または関数コンポーネントのパラメーターとして使用される場合に、そのプロパティーがプロパティー・ストア内に存在しないときは警告メッセージがログに記録されます。
2. 構成ファイルは、AssemblyLines を開始せずにサーバーにロードできます。使用される、存在しないプロパティーに対して、警告メッセージがログに記録されます。

IBM Security Directory Integrator は、終了時に、以下のいずれかの終了コードを戻します。

0 エラーはありません。操作は成功しました。

1

- ログ・ファイルをオープンできません (-l パラメーター)。
- 構成ファイルをオープンできません。
- AssemblyLine が失敗しました。該当するのは、サーバーが非デーモン・モードで動作しているとき、すなわち、「-d」オプションの指定がなく動作しているときのみです。例えば、「`ibmdirsv -c rs.xml -r al`」または「`ibmdisrv -c rs.xml -r al1,al2,al3`」です。

2 (廃止) 自動実行後に終了します。-w を指定して IBM Security Directory Integrator を始動すると、サーバーは -r パラメーターで指定された AssemblyLine を開始してから、終了させます。

注: 構成エディターから実行された AssemblyLine は、別の方法で開始され、状況 2 で終了することはありません。

9 (廃止) ライセンスの有効期限が切れました。またはライセンスが無効です。

注: サーバーが管理要求によってシャットダウンしたとき、カスタム終了コードが指定されていた場合は、カスタム・コードがサーバーの終了コードとして使用されます。

コマンド行インターフェース - tdisrvctl ユーティリティー

IBM Security Directory Integrator に対するコマンド行インターフェース (CLI) (*tdisrvctl* ユーティリティーと呼ばれます) は、構成、AssemblyLine 等をリモートで管理するために設計されています。

このユーティリティーは、リモート・サーバー API を使用してリモート IBM Security Directory Integrator サーバーに接続し、要求された操作を実行します。これはリモート・サーバーに対するクライアント・アプリケーション・インターフェースであるため、105 ページの『第 6 章 セキュリティー』で説明されているものと同じ接続、認証、および許可に関する問題が発生する可能性があります。

以下の機能に関するさまざまなコマンド行オプションが公開されています。

- IBM Security Directory Integrator 構成ファイルの開始、停止、または再ロードを行う。
- 特定の構成で AssemblyLines を開始または停止する。
- サーバーにロードされている構成のリストを表示する。
- サーバーをシャットダウンする。
- 構成レポートを表示する。
- TDI-p (IBM Security Directory Integrator プロパティーのフレームワーク) を介して構成プロパティーを管理する。
- カスタム通知イベントを送信する。
- 公開された AL の操作を表示する。

- 終了した構成ファイルと AssemblyLine のトゥームストーンを表示する。
- IBM Security Directory Integrator サーバーの詳細を表示する。

注:

1. コマンド行ユーティリティーは、*TDI_install_dir*/bin フォルダーに含まれます。
2. リモート・メソッド呼び出し (RMI) は、デフォルトでは global.properties で使用可能になっており (api.remote.on=true)、セキュリティーが使用可能になっています (api.remote.ssl.on=true)。api.remote.ssl.on=true の場合、鍵ストアおよびトラストストアの general_options をコマンドに含める必要があります。例:

   ```
   tdisrvctl.bat  -h mytdiserver.com -p 1099 ..¥serverapi¥testadmin.jks -P administrator -T
   ..¥testserver.jks -W server  -op srvinfo
   ```

   RMI セキュリティーが使用不可である場合 (api.remote.ssl.on=false)、tdisrvctl ユーティリティーを実行するクライアントの IP アドレスをプロパティー api.remote.nonssl.hosts に定義する必要があります。この場合、鍵ストアおよびトラストストアのパラメーターを指定せずに、指定されたクライアントから tdisrvctl コマンドを実行することができます。例を示します。

   ```
   tdisrvctl.bat  -h mytdiserver.com -p 1099   -op srvinfo
   ```

3. リモート IBM Security Directory Integrator サーバーは動作中である必要があります。

コマンド行リファレンス

以下の情報を使用することで、コマンド行リファレンスの使用法について詳しく知ることができます。

コマンドの使用法を以下に示します。

```
tdisrvctl [general_options] –op operation [operation_specific_options]
```

ここで **general_options** には以下を指定します。

-h host	リモート・サーバーの IP アドレスまたはホスト名を入力します (デフォルトは localhost)。
-K keystore	SSL 鍵データベース・ファイルの名前を入力します。
-p port	ポート番号を入力します (デフォルトは 1099)。
-P key_pwd	鍵ファイルのパスワードを入力します。
-s	ソリューション・ディレクトリーを配置する作業ディレクトリーを指定します。
-T truststore	SSL トラストストア・データベース・ファイルの名前を入力します。
-u userID	ユーザー名を入力します (カスタム認証用)。
-v	冗長モードで実行します。
-w user_pwd	ユーザー・パスワードを入力します (カスタム認証用)。
-W trust_pwd	トラスト・ファイルのパスワードを入力します。
-?	コマンドの使用法を表示します。

operation には以下を指定します。

イベント	カスタム通知イベントの送信。
prop	構成プロパティーの管理

queryop	AssemblyLine (AL) 操作の照会
reload	実行中の構成の再ロード
report	リモート・サーバーの構成レポートの生成または構成のリスト
shutdown	サーバーのシャットダウン
srvinfo	IBM Security Directory Integrator サーバー情報の表示
状況	構成ファイルまたは AL の状況の表示
start	特定の構成ファイルまたは AL の開始
stop	特定の構成ファイルまたは AL の停止
tombstone	特定の構成ファイルまたは AL のトゥームストーン項目の表示
deletetombstone	トゥームストーン項目の削除
debug	動作中の AssemblyLine のコンポーネントのデバッグ

以下のようにして、特定のオプションのヘルプを表示することができます。

```
tdisrvctl –op operation -?
```

操作

ここに示されている操作のリストを参照できます。

イベント

このオプションを使用して、カスタム通知イベントを特定のサーバーに送信します。特定のイベントに登録済みのすべてのリスナーがこの通知を受け取ります。これにより、IBM Security Directory Integrator 管理者は、計画されたカスタム・イベントに基づいてリスナー・アプリケーションをトリガーすることができます。

event 操作の使用法:

tdisrvctl [general_options] **-op event -e** event_name [**-s** source] [**-d** data]

説明:

-e event_name	送信するイベントの名前。
-s source	イベントを呼び出すソースの名前 (デフォルトは「tdisrvctl」)。
-d data	イベント・リスナーに渡されるデータ (デフォルトは NULL)。

例:

イベント「user.process.X.completed」を「admin」から送信する。

```
tdisrvctl -h itditest -op event -e "process.X.completed" -s admin -d "Admin triggered event"
```

注: -e オプションを使用して tdisrvctl から送信されるすべてのイベントに接頭部「user.」があります。

prop 「prop」オプションは、TDI-p 経由で構成のプロパティーを公開します。これにより、ユーザーは特定の構成のプロパティーを取得/設定/表示することができます。

prop 操作の使用法:

tdisrvctl [general_options] **-op prop -c** config_name
[[**-l**] |
[**-o** property_store]
[**-g** key | all] |
[**-s** key=value] [**-e**] |
[**-d** key]]

説明:

-c config_name	操作する構成ファイルの名前を指定します。
-l	構成済みのすべてのプロパティー・ストアをリストします。
-o property_store	操作するプロパティー・ストアの名前を指定します。
-g key	指定した鍵 (またはキーワード「all」を指定して、すべての鍵) の値を取得します。
-s key=value	「key」を指定されている「value」に設定します。
-e	ストアへの挿入時に値を暗号化します (-s オプションを指定している場合にのみ使用可能)。
-d key	ストアから指定されている「key」を削除します。

注:

1. 「-l」、「-g」、「-s」、「-d」オプションは相互に排他的であり、組み合わせて使用することはできません。
2. 「-e」オプションは「-s」オプションとの組み合わせでのみ使用できます。
3. **パスワード・ストア**に保管されたプロパティーの管理はサポートされません。
4. 「-c」オプションの指定時には、リモート・サーバー上の構成ファイルの完全パスを指定するか、「configs」フォルダーへの相対パスを指定します。相対パスを参照するには、次のように tdisrvctl の「report」オプションを使用します。

   ```
   tdisrvctl -op report -l
   ```

例:

構成 C1.xml のすべてのプロパティー・ストアのリストを表示するには、以下のようにします。

```
tdisrvctl -op prop -c C1.xml -l
```

構成 C1.xml のすべてのプロパティーのリストを取得するには、以下のようにします。

```
tdisrvctl -op prop -c C1.xml -g all
```

ストア MyStore から構成 C1.xml のすべてのプロパティーのリストを取得するには、以下のようにします。

```
tdisrvctl -op prop -c C1.xml -o MyStore -g all
```

ストア MyStore 内の構成 C1.xml のプロパティー MY_PROP を値 MY_VALUE に設定し、これを保護されているものとしてマークするには、以下のようにします。

```
tdisrvctl -op prop -c C1.xml -o MyStore -s MY_PROP=MY_VALUE -e
```

queryop

queryop オプションは、AssemblyLine で公開される AL の操作のリストを戻します。

このオプションはスクリプト記述環境で役立ちます。 IBM Security Directory Integrator ソリューションの開発者は、公開された操作を自動的に照会し、その結果を使用して、AssemblyLine の特定の操作 (開始操作の **-r**

-alop フラグを使用) を開始するスクリプトを開発できます。スクリプト環境では、この操作の出力は、簡単に grep で操作したり、トークン化することができます。

queryop 操作の使用法:

```
tdisrvctl [general_options] -op queryop -c <configFile> -r <ALname>
```

ここで、

configFile 構成ファイル名
ALName AssemblyLine の名前

出力:

```
ALOp:{attr_1;attr_2...attr_n;}
```

注: 「-c」オプションの指定時には、リモート・サーバー上の構成ファイルの完全パスを指定するか、「configs」フォルダーへの相対パスを指定します。相対パスを参照するには、次のように tdisrvctl の「report」オプションを使用します。

```
tdisrvctl -op report -l
```

例:

AL で公開される *操作* を照会するには、以下のようにします。

```
tdisrvctl -h itditest -T trust.kdb
  -W secret -op queryop
  -c examples/ADCustomConnector.xml
  -r ADAssemblyLine
```

出力の例:

```
$initialize: {ldapurl;loginPasswd;loginUserName}
```

reload このオプションは、特定のサーバーに実行中の構成を再ロードするために使用できます。

reload 操作の使用法:

```
tdisrvctl [general_options] -op reload -c [config_list]
```

説明:

config_list 再ロードする構成のコンマ区切りのリスト

注: 「-c」オプションの指定時には、リモート・サーバー上の構成ファイルの完全パスを指定するか、「configs」フォルダーへの相対パスを指定します。相対パスを参照するには、次のように tdisrvctl の「report」オプションを使用します。

```
tdisrvctl -op report -l
```

例:

リモート・ホスト itditest に構成 C1.xml、C2.xml、および C3.xml を再ロードするには、以下のようにします。

```
tdisrvctl -h itditest -T trust.jks -W secret -op reload -c C1.xml,C2.xml,C3.xml
```

report このオプションは、特定の構成のレポートを生成するため、またはリモート・サーバーの構成フォルダーで使用可能な構成をリストするために使用できます。

構成レポートには、特定の構成の詳細がリストされます。詳細の内容は、AssemblyLine、各 AssemblyLine 内のコネクターおよびパーサー、コネクター・ライブラリー、パーサー・ライブラリー、スクリプト・ライブラリー、関数ライブラリーなどです。このオプションにより、特定の構成の全詳細の一覧が表示されます。

構成リスト・オプションにより、リモート・サーバー上で使用可能な構成のリストおよびそれらの正確な名前を検索することができます。ただし、リモート・サーバーの「config」フォルダーにある構成しか表示できません (`global.properties` ファイルでプロパティー**api.config.folder** を参照)。このコマンドでは、システム上の「どこか」にある構成のリストは取得できません。

report 操作の使用法:

```
tdisrvctl [general_options] -op report [-c config | -l]
```

説明:

-c config　　レポートを生成する構成の名前。
-l　　リモート・サーバーの構成フォルダー内の構成。

AssemblyLine の各コネクターまたは関数コンポーネント部分について表示される詳細は、次のようになります。

```
Name          : count
Mode          : Iterator
State         : Enabled
Debug         : Disabled
Template      : system:/Connectors/ibmdi.Timer
Parser        : [parent]
Comment       : None
```

注:

1. リモート・サーバーに既にロードされている構成を指定してください。
2. 「-c」オプションまたは「-l」オプションはいずれか 1 つのみを指定できます。両方を指定することはできません。
3. 「-c」オプションの指定時には、リモート・サーバー上の構成ファイルの完全パスを指定するか、「configs」フォルダーへの相対パスを指定します。
4. -c オプションの引数では、大/小文字が区別されます。また、この引数はサーバー・インスタンスが認識している構成ファイル名と一致している必要があります。構成ファイル名を確認するには、「tdisrvctl -op status」などを実行します。

例:

リモート・サーバー上の C1.xml の詳細の完全なリストを取得するには、以下のようにします。

```
tdisrvctl -h remoteserver -op report -c C1.xml
```

リモート・サーバーの「config」フォルダー内の使用可能な構成のリストを取得するには、以下のようにします。

```
tdisrvctl -h remoteserver -op report -l
```

shutdown

このオプションは、IBM Security Directory Integrator サーバーをシャットダウンするために使用できます。

このコマンドの形式:

```
tdisrvctl [general_options] -op shutdown [-o return_code] [-f]
```

説明:

-o return_code	リモート IBM Security Directory Integrator サーバーが終了する際の戻りコード。
-f	制御下でのシャットダウンを強制し、すべての AssemblyLine を終了します。

例:

ローカル IBM Security Directory Integrator サーバーをシャットダウンするには、以下のようにします。

```
tdisrvctl -op shutdown
```

すべての AssemblyLine を制御下でシャットダウンしながら、ローカル IBM Security Directory Integrator サーバーをシャットダウンするには、次のコマンドを使います。

```
tdisrvctl -op shutdown -f
```

SSL (サーバー認証のみ) について構成されているリモート・ホスト itditest で実行中のサーバーをシャットダウンするには以下のようにします。

```
tdisrvctl -h itditest -T trust.kdb -W secret -op shutdown
```

srvinfo このオプションは、IBM Security Directory Integrator サーバーの情報を表示するために使用します。

このコマンドの使用法は、次のとおりです。

```
tdisrvctl [general_options] -op srvinfo
```

例:

localhost で動作中の IBM Security Directory Integrator サーバーのサーバー情報を表示するには、以下のようにします。

```
tdisrvctl -h localhost -op srvInfo
```

状況 このオプションを使用すると、AssemblyLine の状態を表示できます。

status 操作の使用法:

tdisrvctl [general_options] **-op status -c** [config_list | all]
-r [AL_list | all]
-listen

説明:

config_list	コンマ区切りの構成ファイルのリストまたはキーワード「all」。
AL_list	コンマ区切りの AL ファイルのリストまたはキーワード「all」。
-listen	実行中の構成ファイルまたは AssemblyLine のログの受信の開始を示します。

注:

1. 少なくとも 1 つのオプション (「-c」または「-r」) を指定する必要があります。 -
2. キーワード「all」は、すべての構成、AssemblyLine を指示します。
3. -listen オプションでは、構成または AssemblyLine を必ず 1 つ指定する必要があります。
4. 「-c」オプションの指定時には、リモート・サーバー上の構成ファイルの完全パスを指定するか、「configs」フォルダーへの相対パスを指定します。相対パスを参照するには、次のように tdisrvctl の「report」オプションを使用します。

   ```
   tdisrvctl -op report -l
   ```

例:

すべての構成ファイルと AL の状況を表示するには、次のコマンドを実行します。

```
tdisrvctl [general_options] -op status -c all -r all
```

以下のようにも記述できます。

```
tdisrvctl [general_options] -op status
```

AL1 と AL2 の状況を表示するには、次のコマンドを実行します。

```
tdisrvctl -h itditest -op status -c c1.xml -r AL1,AL2
```

出力:

```
(Component Type # Component Name # RUNNING / STOPPED # Statistics):
1 # AL1 # RUNNING # [get:571] [add:571] [del:3] [requests:2333]....
1 # AL2 # STOPPED #
```

コンポーネント・タイプは以下のとおりです。

- 構成の場合は 0
- AssemblyLine の場合は 1

統計には以下の詳細が含まれます (AssemblyLine のみ有効)。

- 属性「add」- 実行された「add」操作の総数
- 属性「mod」- 実行された「modify」操作の総数
- 属性「del」- 実行された「delete」操作の総数
- 属性「get」- 実行された「getNext」(Iterations®) の総数
- 属性「request」- AssemblyLine にサーバー・モード・コネクターがある場合に受信された要求の総数
- 属性「callReply」- 実行された「callReply」操作の回数
- 属性「err」- 検出されたエラーの総数
- 属性「skip」- 実行された「skip」操作の総数
- 属性「lookup」- 実行された「lookup」操作の総数
- 属性「ignore」- 実行された「ignore」操作の総数
- 属性「reconnect」- 実行された「reconnect」操作の総数

- 属性「exception」- 例外が発生してコンポーネントが終了した場合の例外テキスト

特定のサーバー上の構成 (実行中または停止状態) の詳細を表示するには、以下のようにします。

```
tdisrvctl -h itditest -op status -c all
```

特定のサーバー上で実行中の AssemblyLine の詳細を表示し、ログの受信を開始するには、次のコマンドを実行します。

```
tdisrvctl -h itditest -op status  -c rs.xml -r al1 -listen
```

start このオプションを使用すると、構成ファイルまたは AssemblyLine を開始できます。

start 操作の使用法:

tdisrvctl [general_options] **-op start -c** [config]
-e [password]
-r [AL_list | all] **-alop** <alop_Name> [{requiredAttr_1; requiredAttr_2; ... requiredAttr_n}] | [**-f** filename]
-s [Simulate mode]
-m [run name] **-o** [propStore1=filename1,propStore2=filename2...]
-t [temp config instance]
-listen
-sync

ここで、

-c config	開始する構成の名前。
-e password	構成ファイルが暗号化されている場合のパスワード。
-r AL_list	開始する AL のコンマ区切りのリストまたはキーワード「all」。
-o プロパティー・ファイル・リスト	プロパティー・ストア名と値のコンマ区切りのリスト
-alop operName	特定の AL 操作および指定した操作の必須属性のリスト。
-f filename	入力属性とその値が操作用に構成されているファイルの名前。
-s シミュレート・モード	指定した AssemblyLine をシミュレート・モードで実行する
-m マルチインスタンス	異なる実行名で同じ構成ファイルの複数のインスタンスを実行する
-t 一時構成インスタンス	指定した構成ファイル内の XML から一時構成インスタンスを開始する
-listen	指定した構成ファイルまたは AssemblyLine のログを受信する
-sync	AssemblyLine を同期型で実行する

注:

1. 「-c」オプションは必須です。-
2. キーワード「all」は、すべての AssemblyLine を意味します。
3. -alop オプションでは、必須属性リストが必須です。
4. -alop オプションは、-r *all* オプションと同時には使用できません。これは、特定の AL についてのみ機能します。
5. ソリューションまたは実行名を指定して一時構成を実行すると、サーバーで同じ名前を持つ別の構成ファイルが既に実行中かどうかチェックで

きなくなります。既に実行中の場合は、例外が発生します。 **status** コマンドを使用すれば、動作中の構成インスタンスをチェックできます。

6. -t オプションを指定する場合は、-c オプションで指定された構成がクライアント・マシン上に存在することが必要です。
7. -t オプションを使用したとき、-c オプションで指定した構成が相対的だった場合は、現在のフォルダー内で検索されます。
8. -listen オプションでは、構成または AssemblyLine を必ず 1 つ指定する必要があります。
9. -listen オプションは AssemblyLine を同期型で実行します。 -sync オプションを使用して結合する必要はありません。
10. -sync オプションでは、AssemblyLine を 1 つだけ指定する必要があります。

例:

1. リモート・サーバー itditest で構成 C1 の AssemblyLine AL1 および AL2 を開始するには、次のコマンドを実行します。

```
tdisrvctl -h itditest -T trust.kdb -W secret -op start -c C1.xml -r AL1,Al2
```

   -r オプションを指定するには、-c オプションも同時に指定する必要があります。これは、コマンドで指定する AssemblyLine は、-c オプション内の構成の 1 つに必ず 所属していなければならないためです。

2. AL の操作で、リモート・サーバー itditest 上の AssemblyLine AL1 を開始するには、以下のようにします。

```
tdisrvctl -h itditest -T trust.kdb -W secret -op start
  -c examples/ADCustomConnector.xml
  -r ADAssemblyLine
  -alop $initialize {ldapurl:ldap://9.182.190.149:390;loginPasswd:password;loginUsrname:cn=root}
```

3. リモート・サーバー itditest で AssemblyLine AL1 を AL 操作の更新を伴って開始するには、以下のようにします。

```
tdisrvctl -h itditest -T trust.kdb -W secret -op start
 -c examples/ADCustomConnector.xml  -r ADAssemblyLine
 -alop search {$init.ldapurl:ldap://9.182.190.149:390;$init.loginPasswd:password;
     $init.loginUsrname:cn=root;searchBase:o=ibm,c=us}
```

   注: すべての初期化属性には、接頭部 $init を付ける必要があります。

4.

```
tdisrvctl -h itditest -T trust.kdb -W secret -op start -c examples/ADCustomConnector.xml
     -r ADAssemblyLine -alop search -f inputFile
```

   入力ファイル形式:
   ==========
   Key1:value1
   Key2:value2

5. AssemblyLine AL1 をシミュレート・モードで実行するためのコマンド:

```
tdisrvctl -h itditest -T trust.kdb -W secret -op start -c examples/ADCustomConnector.xml -r AL1 -s
```

6. 複数の構成インスタンスをロードするためのコマンド:

```
tdisrvctl -op start -c C1.xml -m test -f PropertyStorename=TestProp.properties,
     PropStore2=propfile2 ... -r AL1,AL2
```

7. 一時構成インスタンスを実行するためのコマンド:

```
tdisrvctl -op  start -c C1.xml -t -r AL1
```

8. 構成を特定のサーバーで開始し、そのログを受信するためのコマンド:

```
tdisrvctl -h itditest -op start -c rs.xml -listen
```

9. AssemblyLine を特定のサーバーで開始し、そのログを受信するためのコマンド:

```
tdisrvctl -h itditest -op start -c rs.xml -r AL1 -listen
```

10. AssemblyLine を特定のサーバーで同期型で実行するためのコマンド:

```
tdisrvctl -h itditest -op start -c rs.xml -r AL1 -sync
```

stop stop 操作の使用法:

tdisrvctl [general_options] **-op stop -c** [config]
-r [AL_list | all]

説明:

-c config	構成の名前。
-r AL_list	停止する AL のコンマ区切りのリストまたはキーワード「all」
-f	制御下でのシャットダウンを強制する。

注:

1. 「-c」オプションは必須です。
2. 「-c」オプションの指定時には、リモート・サーバー上の構成ファイルの完全パスを指定するか、「configs」フォルダーへの相対パスを指定します。相対パスを参照するには、次のように tdisrvctl の「report」オプションを使用します。

```
tdisrvctl -op report -l
```

3. キーワード「all」はすべての AssemblyLine を意味します。
4. -r オプションを指定するには、-c オプションも同時に指定する必要があります。これは、コマンドで指定する AssemblyLine は、-c オプション内の構成の 1 つに必ず 所属していなければならないためです。
5. -f オプションはオプションです。
6. -c オプションの引数では、大/小文字が区別されます。また、この引数はサーバー・インスタンスが認識している構成ファイル名と一致している必要があります。構成ファイル名を確認するには、「tdisrvctl -op status」などを実行します。

例:

リモート・サーバー itditest 上の構成 C1 の AssemblyLine AL1 と AL2 を停止するには、次のコマンドを実行します。

```
tdisrvctl -h itditest -T trust.jks -W secret -op stop -c C1.xml -r AL1,Al2
```

tombstone

このオプションを使用すると、以前実行した構成ファイル、AssemblyLine、および EventHandler のトゥームストーンの詳細 (履歴) を表示できます。

トゥームストーン操作の使用法は、次のとおりです。

tdisrvctl [general_options] **-op tombstone -c** [config]
[**-r** [AL_name]]
[**-age** n]
[[attribute_list] | all]

説明:

-age n	過去「n」日間のトゥームストーン・レコード (デフォルトは 1 日)。
-c config	構成の名前。
-r AL_name	AssemblyLine の名前。
すべて	トゥームストーンの属性: すべてを表示。

attribute_list:

-ct	コンポーネント・タイプ。
-cn	コンポーネント名。
-guid	トゥームストーン項目の GUID
-et	イベント・タイプ。
-ex	終了コード。
-stime	コンポーネントの開始時刻。
-ctime	トゥームストーンの作成時刻。
-desc	エラーの説明。
-um	ユーザー・メッセージ。
-stat	統計 (AL のみ有効)。

注:

1. 「-c」オプションは必須です。
2. 「-c」オプションの指定時には、リモート・サーバー上の構成ファイルの完全パスを指定するか、「configs」フォルダーへの相対パスを指定します。相対パスを参照するには、次のように tdisrvctl の「report」オプションを使用します。

   ```
   tdisrvctl -op report -l
   ```

3. -c オプションの引数では、大/小文字が区別されます。また、この引数はサーバー・インスタンスが認識している構成ファイル名と一致している必要があります。構成ファイル名を確認するには、「tdisrvctl -op status」などを実行します。

例:

1. 構成 C1.xml の最近 2 日間のトゥームストーン項目 (すべての属性) を表示するには、以下のようにします。

   ```
   tdisrvctl [general_options] -op tombstone -c C1.xml -age 2 all
   ```

2. 過去 3 日間の C1 のトゥームストーン項目を表示するには、以下のようにします。

   ```
   tdisrvctl -h itdiserver -op tombstone -c C1 -age 3 all
   ```

3. 最近 24 時間の構成 C1 のトゥームストーン項目 (特定の属性) を表示するには、以下のようにします。

   ```
   tdisrvctl -h itdiserver -op tombstone -c C1 -ct -ctime -cn -um
   ```

4. 「rs.xml」の AL1 のトゥームストーン項目を表示するには、以下のようにします。

   ```
   tdisrvctl -h itdiserver -op tombstone -c C1 -r AL1
   ```

deletetombstone

このオプションを使用すると、以前実行した AssemblyLine のトゥームストーン項目を削除できます。

トゥームストーン削除操作の使用法は、次のとおりです。

tdisrvctl [general_options] **-op deletetombstone -guid** <GUID number>

ここで、

-guid 「GUID number」は削除するトゥームストーンの固有 ID です。トゥームストーンの GUID は、トゥームストーンの内容を表示すると取得できます。GUID を取得する方法についての詳細は、『トゥームストーンのオプション』に関する項目を参照してください。

debug このオプションを使用すると、実行中の AssemblyLine のコネクターと関数コンポーネントのデバッグ・モードの値を設定できます。コネクターのデバッグ・モードに指定したパーサーを設定すると、パーサーのデバッグ・モードも同じ値で初期化されます。

デバッグ操作の使用法は、次のとおりです。

tdisrvctl [general_options] **-op debug -c** config
-r assembly_line
[**-alc** al_component]
-on/off

説明:

-c config	構成の名前。
-r assembly_line	AssemblyLine の名前。
-alc al_component	AssemblyLine コンポーネントの名前。
-on	デバッグを有効にするフラグ。
-off	デバッグを無効にするフラグ。

注:

1. 「-c」オプションおよび「-r」オプションは必須です。これらのオプションでは、構成または AssemblyLine を必ず 1 つ指定する必要があります。
2. 「-c」オプションの指定時には、リモート・サーバー上の構成ファイルの完全パスを指定するか、「configs」フォルダーへの相対パスを指定します。相対パスを参照するには、次のように tdisrvctl の「report」オプションを使用します。

   ```
   tdisrvctl -op report -l
   ```

3. -c オプションの引数では、大/小文字が区別されます。また、この引数はサーバー・インスタンスが認識している構成ファイル名と一致している必要があります。構成ファイル名を確認するには、「tdisrvctl -op status」などを実行します。
4. **-alc** オプションを指定しない場合、指定された AssemblyLine のすべてのコンポーネントが影響を受けます。

例:

1. 指定した構成の AssemblyLine 内のコンポーネントのデバッグ・モードの値を表示するには、次のコマンドを実行します。

   ```
   tdisrvctl -op report -c C1
   ```

2. 実行中の AssemblyLine al2 内のすべてのコンポーネントのデバッグ・モードを有効にするには、次のコマンドを実行します。

```
tdisrvctl -op debug -c C1-r al2 -on C1-r al2 -on
```

3. 実行中の AssemblyLine al3 内の指定したコンポーネントのデバッグ・モードを無効にするには、次のコマンドを実行します。

```
tdisrvctl -op debug -c C1-r al3 -alc comp1,comp2 -off
```

注意すべきその他のポイント

- **-T** オプションまたは **-K** オプションを指定すると、コマンド行ユーティリティーが SSL を使用する必要があることを意味します。
- **-h** (ホスト) オプションを指定しない場合、コマンド行インターフェースは環境変数 **TDI_RSRV** を検索します。 TDI_RSRV が設定されていない場合または空の場合は、デフォルトとして「localhost」を使用します。これは、**-p** (ポート) オプションでも同様です。**-p** を指定しない場合は **TDI_RPORT** が検索され、TDI_RPORT も指定しない場合はデフォルトの「1099」が使用されます。
- tdisrvctl コマンドは、操作が成功した場合はゼロの終了コードを返し、操作が失敗した場合はゼロでない終了コードを返します。操作が失敗したとき、考えられる理由としては次のものがあります。
  - リモート・サーバーへの接続を確立できない。
  - リモート・サーバーからエラーが返された (多くは実行した操作が原因)。
  - 同期型で実行していた AssemblyLine が失敗した (「start」操作の「-sync」オプションを参照)。
- tdisrvctl コマンド行ユーティリティーは、エラー・メッセージをロギングするために Log4J ロギング API を使用します。起動スクリプト (.bat または .sh) ファイルで、Log4J 構成ファイルを指定します。このコマンドは、Log4J ロギングのセットアップのために `tdisrvctl-log4j.properties` と呼ばれるファイルを使用します。ソリューション・ディレクトリーを指定した場合、コマンドはソリューション・ディレクトリー内のログ構成ファイルを参照する環境変数を設定します。ソリューション・ディレクトリーを指定しなかった場合、コマンドはインストール・ディレクトリー内のログ構成ファイルを使用します。
- `tdisrvctl-log4j.properties` ファイルには、ログが作成される場所の完全パスが格納されます。ログ・ファイルはデフォルトでは、*`TDI_install_dir`*`/logs` ディレクトリーに作成されます。この場所は必要に応じてカスタマイズできます。
- 報告されたすべてのエラーおよび警告メッセージは、エラー・コードの接頭部付きで表示されます。このエラー・コードを使用して、「メッセージ」でエラー・メッセージおよびオペレーターの対応の説明を検索できます。

第 14 章 ロギングおよびデバッグ

以下の情報を使用することで、ロギングおよびデバッグとその動作について詳しく知ることができます。

IBM Security Directory Integrator はロギング・クラスを使用して、メッセージを各種のログ・チャネルに記録します。すべての IBM Security Directory Integrator コンポーネントはこのロギング・クラスを使用し、その結果、業界標準のロギング・ツール (Log4J) が起動されます。Log4J はさまざまな出力チャネルおよび出力フォーマットを提供していますが、IBM Security Directory Integrator ユーザーが必要とするような、重複した出力チャネルや追加の出力チャネルを持つ他のロギング・ユーティリティーもあります。これらの多くはオープン・ソース・ライブラリーであり、IBM Security Directory Integrator にバンドルされていません。これらのサード・パーティー製ロギング・ユーティリティーを組み込めるように、IBM Security Directory Integrator ロギング・コンポーネントは、IBM Security Directory Integrator と実際のロギング・インプリメンテーション (`LogInterface` インプリメンテーションと呼ばれる) 間のプロキシーとして動作するようにモデリングされています。独自の LogInterface クラスの作成、構成、およびプログラミング方法の詳細については、『リファレンス』セクションにある『追加ロガーの作成』セクションを参照してください。

注: `com.ibm.di.logging.enabled` プロパティーを構成し、IBM Security Directory Integrator のロギングを有効または無効にします。ロギングを有効にするには、`com.ibm.di.logging.enabled=true` (デフォルト) を使用します。ロギングを完全に無効にするには、`com.ibm.di.logging.enabled=false` を使用します。

ここからは、IBM Security Directory Integrator にバンドルされている `com.ibm.di.log.TDILog4J` と呼ばれるロギング・クラスの使用法を説明します。

ロギングおよびデバッグは、主にタスク・オブジェクト (現行の AssemblyLine) により実行されます。ロギングは (スクリプトで) 明示的に実行されるか、または各種コンポーネント自体により実行されます。

Log4J ロギング・エンジンは、ファイル、イベント・ログ、および Syslog にログを記録できる非常に柔軟なフレームワークです。ロギングは、ほとんどのニーズに対応するように調整できます。ソリューションのトラブルシューティングまたはデバッグを行う際に非常に役立ちます。上述したロギング・クラスを使用すると、IBM Security Directory Integrator で追加トレース機能 (265 ページの『第 15 章 トレースと FFDC』を参照) を利用できるようになりますが、多くの場合は、ここで説明するロギング機能で十分です。

IBM Security Directory Integrator コンポーネントには具体的なトラブルシューティング・ガイドラインがある場合があります。IBM Security Directory Integrator の IBM Knowledge Centerの『リファレンス』および『トラブルシューティングとサポート』セクションにある特定のコンポーネントのセクションを必ず参照して、詳細を確認してください。

サーバー (ibmdisrv) のログ・スキームは Log4J.properties ファイルおよび構成ファイルのエレメントによって記述されています。 262 ページの『Log4J のデフォルト・パラメーター』を参照してください。

注: 上述のプロパティー・ファイルはすべて、ソリューション・ディレクトリーに配置できます。その場合、ソリューション・ディレクトリーのファイルにリストされるプロパティーの方が、インストール・ディレクトリー内のファイルで設定されている値よりも優先されます。

独自に作成した Appender を Log4J ロギング・エンジンに使用させることができます。そのためには、その Appender を Log4J.properties ファイルの中で定義します。また、デフォルトのドライバーのように Log4J に組み込まれているドライバーを使用することができます。デフォルトのドライバーは、次のステートメントで定義されます。

```
Log4J.appender.Default=org.apache.Log4J.FileAppender
```

org.apache.Log4J.FileAppender という句は、この Appender が FileAppender クラスを使用することを定義しています。追加の Log4J 準拠のドライバーをインターネットから入手可能です。例えば、JMS や JDBC を使用してログを記録できるドライバーなどがあります。それらのドライバーを使用するには、IBM Security Directory Integrator のインストール先の jars ディレクトリーに、使用するドライバーをインストールする必要があります。その後 Appender がそれらの追加ドライバーを使用するように Log4J.properties ファイルの中で定義できます。詳しくは、http://jakarta.apache.org/log4j/docs を参照してください。

IBM Security Directory Integrator の組み込みロギング以外に、AssemblyLine にスクリプト・コードを追加してログを記録することもできます。詳細については、IBM Security Directory Integrator の IBM Knowledge Center の『*構成*』セクションを参照してください。ここでは、対話式デバッガーの機能についても説明されています。

スクリプト・ベースのロギング

スクリプト記述ができる場所 (フックやスクリプト・コンポーネントなど) であればどこででも、JavaScript を使用して、任意の時点で、AssemblyLine の構成ロガーに対してメッセージを発行することができます。

ユーザーが使用できる明示的な logmsg() 呼び出し (**task.logmsg()** および **main.logmsg()** には、メッセージ記録レベルとして Log4J レベルを指定するオプションのストリング・パラメーターがあります。デフォルトは INFO です。ユーザーが指定したログ・レベルが Log4J に対して無効な場合は、メッセージは DEBUG レベルでログに記録されます。レベルには、
DEBUG、INFO、WARN、ERROR、FATAL があります。

次の

```
task.logmsg()
```

を使用すると、メッセージは AssemblyLine からの他のメッセージとともに、ログに記録されます。構成エディターから AssemblyLine を実行している場合、ログは CE 出力ウィンドウに表示されます。AssemblyLine で別のロギング・メソッドも使用している場合、そのメッセージもそこに表示されます。

次の

```
main.logmsg()
```

を使用すると、メッセージは構成インスタンスからの他のメッセージとともに、ログに記録されます。これはログ・ファイルまたは構成インスタンスが作成した別のロガーに格納されますが、通常、構成エディターでは表示されません。

デフォルトの Log4J クラスを使用したロギング

以下の情報を使用することで、デフォルトの Log4J クラスを使用したロギングについて詳しく知ることができます。

Apache Log4J を使用して IBM Security Directory Integrator のデフォルトのロギングを構成する場合、グローバルに構成する (サーバー・タスクのグローバル・デフォルトを指定する `Log4J.properties` ファイルを使用する) か、または各 AssemblyLine、または構成ファイル全体について、構成エディターを使用して、個別に構成します。このようなレベルの柔軟性とカスタマイズを提供するために、Java Log4J API が使用されます。

ここでは、メッセージをログに記録する方法を指定するパラメーターについてのみ説明します。

すべてのログ構成ウィンドウは同様に機能し、各ウィンドウでは 1 つ以上のログ・スキームをセットアップできます。これらは、`Log4J.properties` ファイルにデフォルトが設定されるたびに、同時にアクティブになります。 262 ページの『Log4J のデフォルト・パラメーター』を参照してください。

多くの (すべてではありません) ロガーが、ログ・ファイルが書き込まれる際の文字セットを制御するために文字エンコード・オプションをサポートしています。数多くの文字セットがあります。非公式の概説については、http://download.oracle.com/javase/6/docs/technotes/guides/intl/encoding.doc.html を参照してください。

指定できるログ・スキームは以下のとおりです。

FileRollerAppender

ファイルはディスクを占有するため、場合によっては、ログを記録するファイルの数を一定数に制限できます。FileRollerAppender は、サーバーを実行するごとに新規ファイルを生成します。システムは、指定された数の以前のログのみを保管します。ログの名前が mylog.txt であり、2 世代を要求した場合は、3 回目以降の実行では mylog.txt (最新の実行) とファイル mylog.txt.1 および mylog.txt.2 が残ります。ここで、mylog.txt.2 が最も古いログになります。これ以降、ファイル数は増えず、同じ名前で新しいバージョンのみが生成されます。2 世代のバックアップ・ファイルを保持します。

FileRollerAppender には以下のパラメーターがあります。

ファイル・パス

記録先ファイルの名前です。パスは、IBM Security Directory Integrator のインストール先に対する相対パスです。ファイル名で使用される特殊マクロ {0} はサーバーの名前で置換されます。同様に、ファイル名で {1} を使用すると、システムによって生成された

固有 ID で置換されます。{1} マクロは、FileRollerAppender を使用する特別な場合に直接関係しませんが、固有のファイル名が必要な場合に重要です。

バックアップ・ファイル数
「ファイル・パス」が mylog.txt であり、2 つのバックアップ・ファイルを選択した場合は、3 回目を実行すると、これら 2 つはそれぞれ mylog.txt.1 および mylog.txt.2 に名前変更されます。

レイアウト
ログ・メッセージの形式を判別します。オプションは以下のとおりです。

- パターン (メッセージをログに記録する方法をカスタマイズする場合に使用します)
- 単純 (ログ・レベルおよびメッセージのみを含む形式)
- HTML (一部の (相対) 時間情報、スレッド情報、ログ・レベル、カテゴリー、およびメッセージを含む HTML ファイルを作成します)
- XML (HTML と同様ですが、XML ファイルを生成します (ネーム・スペース接頭部 Log4J を使用します))

パターン
「**レイアウト**」が「**パターン**」の場合にのみ使用します。 263 ページの『独自のログ方式の作成』を参照してください。

ログ・レベル
ログ・メッセージの重大度レベルです。オプションは以下のとおりです (情報の量が最大のものから最小のものの順)。

- DEBUG
- INFO
- WARN
- ERROR
- FATAL

文字エンコード
使用される文字エンコード: Cp1252、ISO-8859-1 など。

ログの使用
この Appender を使用可能にするには、ここをクリックします。

ConsoleAppender
コンソール (標準出力) にログを記録します。これは、サーバー (ibmdisrv) を始動したウィンドウまたは構成エディター (ibmditk) のタスク実行ウィンドウにあります。**Console** には以下のパラメーターがあります。

レイアウト
前述の **FileRollerAppender** を参照してください。

パターン
前述の **FileRollerAppender** を参照してください。

ログ・レベル
前述の **FileRollerAppender** を参照してください。

ログの使用
前述の **FileRollerAppender** を参照してください。

FileAppender
ファイルにログを記録します。**File** には以下のパラメーターがあります。

ファイル・パス
前述の **FileRollerAppender** を参照してください。

ファイルに付加
ログ情報をファイルに付加するには、ここをクリックします。オプションが有効ではない場合、ファイルは上書きされます。

レイアウト
前述の **FileRollerAppender** を参照してください。

パターン
前述の **FileRollerAppender** を参照してください。

ログ・レベル
前述の **FileRollerAppender** を参照してください。

文字エンコード
使用される文字エンコード: Cp1252、ISO-8859-1 など。

ログの使用
前述の **FileRollerAppender** を参照してください。

デフォルトではこれがアペンダーのセットアップです。 262 ページの『Log4J のデフォルト・パラメーター』を参照してください。

SyslogAppender
IBM Security Directory Integrator が UNIX の Syslog にログを記録できるようにします。**Syslog** には以下のパラメーターがあります。

ホスト名/IP アドレス
ログオンするホストです。

Syslog 機能
ドロップダウンに適切な機能が表示されます。記録先ホストでサポートされている必要があります。

機能ストリングを出力
設定すると、出力メッセージにアプリケーションの機能名が組み込まれます。

レイアウト
前述の **FileRollerAppender** を参照してください。

パターン
前述の **FileRollerAppender** を参照してください。

ログ・レベル
前述の **FileRollerAppender** を参照してください。

ログの使用
前述の **FileRollerAppender** を参照してください。

NTEventLog

アプリケーションが Windows NT イベント・ログをログに記録できるようにします (Windows プラットフォームの場合)。**NTEventLog** には以下のパラメーターがあります。

ソース NT イベント・ログに表示される「ソース」名です。通常、ロギングを実行しているアプリケーションのタイトルです。

レイアウト

前述の **FileRollerAppender** を参照してください。

パターン

前述の **FileRollerAppender** を参照してください。

ログ・レベル

前述の **FileRollerAppender** を参照してください。

ログの使用

前述の **FileRollerAppender** を参照してください。

DailyRollingFileAppender

日次ローリング・ファイル Appender は、毎日ログ・ファイルを交替します。出力ファイルを交替した場合、出力ファイルにはベース名と日付のパターン・ストリングで構成される名前 (すなわち、filename.yyyy-mm-dd) が付与されます。通常、「**ファイルに付加**」パラメーターを **true** に設定して使用します。**DailyRollingFile** には以下のパラメーターがあります。

ファイル・パス

前述の **FileRollerAppender** を参照してください。

ファイルに付加

ここをチェックしているかどうかに応じて、新規ファイルを作成するか既存ファイルに付加します。通常、DailyRollingFile を使用する場合はここをオンにします。

日付パターン

ファイルを切り替える頻度。ドロップダウンを使用して、分から月までの頻度を選択します。例えば、「ファイル・パス」を example.log に設定し、「日付パターン」を `'.'yyyy-MM-dd` に設定した場合は、 2003 年 10 月 31 日の深夜 0 時にログ・ファイル example.log が example.log.2003-10-31 にコピーされます。2003 年 11 月 1 日のロギングは、次の日にロールオーバーするまで example.log で継続されます。

レイアウト

前述の **FileRollerAppender** を参照してください。

パターン

前述の **FileRollerAppender** を参照してください。

ログ・レベル

前述の **FileRollerAppender** を参照してください。

文字エンコード

使用される文字エンコード: Cp1252、ISO-8859-1 など。

ログの使用

前述の **FileRollerAppender** を参照してください。

また 257 ページの『デフォルトの Log4J クラスを使用したロギング』の下の例も参照してください。

SystemLogAppender

この Appender はログ・ファイルを *TDI_install_dir*/system_logs の下にカタログ階層として作成します。構成ファイルごとに対応するディレクトリーが作成され、*AL_xxx* という名前 (xxx は実行される AssemblyLine の名前) のログ・ファイルが配置されます。

この Appender には以下のパラメーターがあります。

パターン

LOG4J によって定義されるログの形式を指定します。デフォルト値は以下のとおりです。

```
"%d{ISO8601} %-5p [%c] - %m%n"
```

このフィールドで使用できる追加の値は以下のとおりです。

```
"%d{HH:mm:ss} %p [%t] - %m%n"
"%p [%t] %c %d{HH:mm:ss,SSS} - %m%n"
```

ログ・レベル

前述の **FileRollerAppender** を参照してください。

文字エンコード

使用される文字エンコード: Cp1252、ISO-8859-1 など。

ログの使用

前述の **FileRollerAppender** を参照してください。

ログ・レベルとログ・レベル制御

ここにリストされているログ・レベルを参照できます。

ログ・レベルには以下のものがあります。

- ALL
- DEBUG
- INFO
- WARN
- ERROR
- FATAL
- オフ

ALL はすべてをログに記録します。DEBUG、INFO、WARN、ERROR、および FATAL の順に、メッセージ・フィルター処理のレベルが高くなります。OFF ではログが記録されません。

IBM Security Directory Integrator でスクリプトを使用できる場合は、JavaScriptの logmsg() メソッドを使用してシステム・ログまたは AssemblyLine ログにログ・メッセージを発行できます。このメソッドには 1 つまたは 2 つのパラメーターを指

定できます。logmsg() の宣言については、Java API 文書を参照してください (パッケージ **com.ibm.di.server**、クラス *AssemblyLine*、またはクラス *RS*)。

追加ログ・レベル・パラメーターが指定された logmsg() メソッド (メインおよびタスクの両方) のインターフェースは、**logmsg (String logLevel, String msg)** です。logLevel の有効な値は、「FATAL」、
「ERROR」、「WARN」、「INFO」、「DEBUG」です。これらの値はそれぞれ、log Appender の有効なログ・レベルに対応しています。認識されない値はすべて「DEBUG」として処理されます。

IBM Security Directory Integrator の `logmsg()` JavaScript 呼び出しは、デフォルトでは INFO レベルでログを記録します。つまり、ログ・レベルを WARN 以下に設定すると、すべての詳細ログ設定とともに logmsg が抑制されます。ただし、logmsg() 呼び出しのレベル・パラメーターを使用して、個々の logmsg() 呼び出しごとにログ・レベルを指定変更できます。

Log4J のデフォルト・パラメーター

内容の変更のためのデフォルト構成を参照できます。

IBM Security Directory Integrator をインストールすると、*FileAppender* がデフォルトのロガーとして使用されます。デフォルトのロガーを変更するには、*TDI_installdir*`/etc` フォルダーにある `log4j.properties` ファイルの内容を変更する必要があります。デフォルトの構成は次のようになっています。

```
# This is the default logger, you will see that it logs to ibmdi.log
log4j.appender.Default=org.apache.log4j.FileAppender
log4j.appender.Default.file=logs/ibmdi.log
log4j.appender.Default.layout=org.apache.log4j.PatternLayout
log4j.appender.Default.layout.ConversionPattern=%d{ISO8601} %-5p [%c] - %m%n
log4j.appender.Default.append=false
```

FileAppender ロガーは、IBM Security Directory Integrator サーバーが始動されるたびに、ibmdi.log ファイル (TDI_installdir/logs 内) の内容を切り捨てます。その動作を変更するには、log4j.appender.Default.append プロパティーに true を設定する必要があります。

log4j.properties ファイルには、デフォルトのロガーを RollingFileAppender または DailyRollingFileAppender に変更する別の例も格納されています。これらのロガーは、使用するロガーをアンコメントし、FileAppender ロガーをコメントにするだけで簡単に使用できます。

```
##########ROLLING FILE SIZE APPENDER
##RollingFileAppender rolls over log files when they reach a certain size specified by the
##MaxFileSize parameter

#log4j.appender.Default=org.apache.log4j.RollingFileAppender
#log4j.appender.Default.File=logs/ibmdi.log
#log4j.appender.Default.Append=true
#log4j.appender.Default.MaxFileSize=10MB
#log4j.appender.Default.MaxBackupIndex=10
#log4j.appender.Default.layout=org.apache.log4j.PatternLayout
#log4j.appender.Default.layout.ConversionPattern=%d{ISO8601} %-5p [%c] - %m%n

##########DAILY OUTPUT LOG4J SETTINGS
## With the DailyRollingFileAppender the underlying file is rolled over at a user chosen frequency.
##The rolling schedule is specified by the DatePattern option

#log4j.appender.Default=org.apache.log4j.DailyRollingFileAppender
#log4j.appender.Default.file=logs/ibmdi.log
```

```
#log4j.appender.Default.DatePattern='.'yyyy-MM-dd
#log4j.appender.Default.layout=org.apache.log4j.PatternLayout
#log4j.appender.Default.layout.ConversionPattern=%d{ISO8601} %-5p [%c] - %m%n
```

Log4J.properties ファイル (ibmdisrv および ibmditk 用) にあるパラメーターの一部を以下に示します。

詳細な資料については、http://jakarta.apache.org/log4j/docs を参照してください。

Log4J.rootCategory=DEBUG, Default
DEBUG は、指定された Appender (Default と呼ばれる Log4J 用語) のログ・レベルです。loglevel に OFF または INFO よりも上のレベルを設定すると、スクリプト logmessages (以下のログの用語を参照) からの出力がなくなります。

Log4J.appender.Default
指定された Appender Default がどのタイプの Appender であるかを定義します。以下のいずれかにすることができます。

- FileRollerAppender (サーバーを実行するごとに新規ファイルを生成します)
- ConsoleAppender (コンソールにログを記録します)
- FileAppender (ファイルにログを記録します)
- SyslogAppender (UNIX の Syslog にログを記録します)
- NTEventLog (Windows NT の EventLog にログを記録します)
- DailyRollingFileAppender (ファイル名に日付スタンプを付けて古いファイルを保管します)
- SystemLogAppender (*root_directory*/system_logs 以下のフォルダー構造にログが生成されます)

Log4J.appender.Default.file
FileAppender のデフォルト・ログ・ファイル。インストール・ディレクトリーに対して相対的です (デフォルトは ibmdi.log)。

Log4J.logger.com.ibm.di.*
IBM Security Directory Integrator の各種コンポーネントのログ・レベル。例えば、ibmditk は IBM Security Directory Integrator 構成エディター自体 (その内部で実行中のプロセスではない) のログ・レベルを示します。これは変更しないでください。

独自のログ方式の作成

このフレームワークを使用して、別々の AssemblyLine のログ方法を区別できます。

このフレームワークを使用して、別々の AssemblyLine のログ方法を区別できます。

注: ここに記載されている情報は、`global.properties` ファイルを引き続き使用してロギング出力をカスタマイズするユーザーを対象としています。構成エディター (ibmditk) を使用してロギング出力をカスタマイズできます。

以下のセクションでは、CONSOLE と呼ばれるログ・スキームを定義します。これは後で特定の AssemblyLine によって使用することができます。:

```
Log4J.appender.CONSOLE=org.apache.Log4J.ConsoleAppender
Log4J.appender.CONSOLE.layout=org.apache.Log4J.PatternLayout
Log4J.appender.CONSOLE.layout.ConversionPattern=%d [%t] %-5p - %m%n0
```

ここで、AssemblyLine myAL でこれが使用されるようにするには、以下の行が必要です。

```
Log4J.logger.AssemblyLine.myAL=INFO, CONSOLE
```

ConversionPattern パラメーターの説明については、Log4J (バージョン 1.2) の詳細資料を参照してください。一部のパラメーターを以下に示します。

%d フォーマットに依存する日時。

%p 優先順位。

%c カテゴリー。

注: 通常、これは *Type.alName.xxx* という形式です。*Type* は EventHandler または AssemblyLine、*alName* は AssemblyLine (または作成者によって指定された EventHandler) の名前、*xxx* はスレッドの固有 ID です。**%c{2}** 出力 *alName & 固有 ID*。

%m メッセージ。

%n 改行。

%t スレッド名。

第 15 章 トレースと FFDC

これは Log4J に類似したロギング・ライブラリーですが、特にトレースおよび First Failure Data Capture (FFDC) のために IBM Security Directory Integrator 内部で使用されます。

255 ページの『第 14 章 ロギングおよびデバッグ』で説明したユーザーによる構成が可能なロギング機能に加えて、IBM Security Directory Integrator のコード全体には JLOG フレームワークを使用したトレース・ステートメントが備えられています。この機能による情報がどの程度エンド・ユーザーの目に触れるかは、グローバル構成ファイル `jlog.properties` 内の一部の構成オプションや、サーバーのコマンド行オプション **-T** によって異なります。

注: 通常は、255 ページの『第 14 章 ロギングおよびデバッグ』 のセクションで説明したロギング・オプションを使用すれば、ソリューションのトラブルシューティング、デバッグ、およびサポートを十分に行えるはずです。ただし、何かの理由で IBM サポートに連絡した場合は、サポート・プロセスとして、ここで説明するトレース機能に関連した一部のパラメーターを変更するように依頼されることがあります。

トレースの機能拡張

大部分のコネクターとパーサーには、トレースの入り口と出口を示すステートメントがあります。IBM Security Directory Integrator サーバー上のいくつかのクラスに、リスト項目に追加されたトレース・ステートメントがあります。

リスト項目:

- メソッドの入り口と出口。
- サード・パーティー・ソフトウェアと対話する部分。
- スレッドの生成。

トレースの基礎知識

トレースは、JLOG の PDLogger オブジェクトを使用して、IBM Security Directory Integrator のコードで実行されます。PDLogger は「**P**roblem **D**etermination Logger (問題判別ロガー)」の略で、Logxml 形式 (Tivoli® 標準) でメッセージをログに記録します。IBM サポートではこのログを理解することができ、専用の処理ツールも持っています。

トレースされ、PDLogger API により処理される、基本レベルの情報は次のようなものです。

```
Date | Time | ClassName | methodName | MachineName | IP | {Entry/Exit/Exception} | [Parameter]
```

基本トレース情報とは、時刻、レベル (最少、中間、最大)、コード内の場所 (すなわち、メソッド名と入り口/出口) です。「|」文字は文書化の目的で追加したもので、実際のログには含まれていません。

トレースの実行に Log4J の Appender を使用しない理由は以下のとおりです。

1. トレースは常に使用可能にしておく必要がある。
2. サーバー内で複数のトレースを使用可能にすることを回避する (Appender を使用すると AL ごとにトレースが使用される)。

PDLogger は、JLOG **SnapMemory** ハンドラーおよび **JlogSnapHandler** に付加されます。

SnapMemory ハンドラーは、トレース・メッセージをメモリーに記録します。LogEvent が起動されると (例えば、`jlog.levelflt.level` フィルターにより定義された特定のログ・レベルのトレース・メッセージが発生した場合、アプリケーションがクラッシュした場合、特定の TMS XML messageID が発生した場合など)、**JlogSnapHandler** によってトレース・メモリー・バッファーがファイルに書き込まれます。

IBM Security Directory Integrator のトレースおよびログ・メッセージをすべての IBM 製品の中で固有にするため、メッセージには固有の接頭部 **CTGDI** が付加されます。

すべてのエラー・メッセージには固有の接頭部として TMSXML messageID が付加されます。このメッセージは、エラーの原因とオペレーターの対応を示します。

すべての情報メッセージにも固有の接頭部として TMSXML messageID が付加されます。このメッセージは、オペレーターの対応を示している場合と、示していない場合があります。

トレースの構成

`jlog.properties` ファイルの `jlog.logger.level` プロパティーを使用して、必要なトレース・レベルを設定することができます。

トレース・レベルには、以下の JLOG ログ・レベル (最も重大度の高いものから低いものまでの階層) のいずれかを設定できます。

- FATAL
- ERROR
- WARNING
- INFO
- DEBUG_MIN
- DEBUG_MID
- DEBUG_MAX

デフォルトのレベルは `FATAL` です。

デフォルトのトレース・レベルとトレースの出力をファイルにするか、またはメモリーにするかという指定は、デフォルトの `jlog.properties` ファイルで定義します。このファイルは、*`TDI_install_dir`*`/etc` フォルダー内にあります。ソリューション・ディレクトリーを使用している場合は、*`TDI_Solution_dir`*`/etc` フォルダー内にあります。

トレース・レベルの動的設定

IBM Security Directory Integrator には、LogCmd.bat (Windows 用) と LogCmd.sh (Unix 用) スクリプトが付属しています。それらを使用してトレース・プロパティーを動的に設定できます。

JLOG ロガーはコマンド・サーバーをデフォルトのポート (9992) で始動して、logcmd コマンド行ユーティリティーが送信するログ・コマンドを listen します。

logcmd スクリプトが機能するためには、最初にコマンド・サーバーを始動させておく必要があります。ログ・コマンド・サーバーを始動するには、`jlog.properties` ファイルに jlog.noLogCmd=false を設定する必要があります。

このサーバーの listen ポートは、`jlog.properties` ファイルの jlog.logCmdPort プロパティーに希望する値を設定すれば変更できます。これらのプロパティーについての詳細は、`jlog.properties` ファイルのコメントを参照してください。

logcmd コマンドの使用法は、次のとおりです。

```
logcmd -o port_number { [-h] | [help] |
          [list {node_name} ] |
          [config node_name] |
          [set node_name key_name=value |
          [remove node_name  {key_name} ] |
          [dump handler_name] | [save {all} ] }
```

ここで、

-o port_number
: ログ・コマンド・サーバーへの接続で使用するポート番号。指定しなかった場合は、デフォルトのポート (9992) が仮定されます。

-h | help
: コマンドの構文情報を表示します。

リスト すべての既知のロギング・オブジェクト (ノード) の名前をリストします。

list node_name
: 指定したノード名の子ノードの名前をリストします。すべてのロギング・オブジェクトが子を持つとは限りません。

config node_name
: ノードのすべての構成プロパティーをリストします。

set node_name key_name=value
: 指定したノード名のプロパティー・キーを設定します。ロギング・オブジェクト node_name が存在しなかった場合は、ロギング・オブジェクトが作成され、プロパティーが追加されます。

remove node_name
: 構成オブジェクト node_name を削除します。構成ノードを削除しても、この構成からインスタンス化されたロギング・オブジェクトは影響を受けません。

remove node_name key_name

ロギング・オブジェクト node_name から構成プロパティー key_name を削除します。オブジェクトがプロパティーの階層的継承をサポートしている場合は、後続する `logcmd config node_name` コマンドで直前に削除したキーが表示されることがあります。この場合、このキーは祖先から継承したものです。

save {all}

ロギング構成をパーシスタント・ストアに保管します。**all** を指定した場合は、構成全体が保管されます。それ以外の場合は、ファイルからロードしていた元の構成ノードのみが保管されます。

便利な JLOG パラメーター

ここでリストした JLOG パラメーターを参照できます。

プロパティー	値	説明
jlog.snapmemory.queueCapacity	デフォルトは 10000	SnapMemory ハンドラーのキューに格納できるログ・イベントの数。
jlog.snapmemory.dumpEvents	true	プロパティーに true を設定すると、ハンドラーはキューに入れられたイベントを直ちに出力リスナーに送信します。その後、このプロパティーは false にリセットできます。
jlog.snapmemory.userSnapDir	CTGDI/FFDC/user/	ユーザーが LogCmd スクリプトを使用して FFDC アクションをトリガーしたときにトレース・ダンプ・ファイルが配置されるディレクトリー。
jlog.snapmemory.isSync	デフォルトは false	プロパティーに true を設定している場合、ログ・イベントは同期してスナップショット・ファイルにダンプされます。その処理のために新しくスレッドが作成されることはなく、スナップショットが完了するまでロガーはブロックされます。
jlog.snapmemory.userSnapFile	userTrace.log	
jlog.snapmemory.triggerFilter	jlog.levelflt	JFFDC アクションを実行するために使用されるレベル・フィルター。
jlog.snapmemory.msgIds	*E	JFFDC アクションのために使用される TMSXML メッセージ・フィルター。
jlog.snapmemory.mode	PASSTHRU または BLOCK。デフォルトは PASSTHRU です。	msgIDs プロパティーに BLOCK を設定すると、リストした ID がブロックされます。PASSTHRU を設定すると、リストした ID がフィルターに送信されます。
Jlog.snapmemory.msgIDRepeatTime	10000 (単位はミリ秒)	特定の TMS メッセージ ID を持つ LogEvent を引き渡した後、同じ ID を持つ別の LogEvent を引き渡せるようになるまでの最小時間 (ミリ秒単位)。

jlog.snapmemory.triggerFilter のデフォルト値では、jlog.levelflt という名前のトリガー・フィルターがセットアップされます。この種のフィルターの属性はメッセージ重大度で、前に説明した JLOG のログ値の 1 つを取ります。デフォルトでは、次のようなエントリーが入っています。

```
jlog.levelflt.className=com.ibm.log.LevelFilter
jlog.levelflt.level=FATAL
```

これにより、重大度 FATAL のトレース・メッセージが発生した時点でメモリー・バッファーをトレース・ログにダンプするように FFDC コードがセットアップされます。jlog.levelflt.level プロパティーにはそれ以外の任意のログ・レベル値を指定で

きますが、適切な値は ERROR か FATAL のみです。それ以外の値を指定すると、FFDC ダンプの量が大幅に増え、IBM Security Directory Integrator サーバーを著しくスローダウンさせる原因になります。

第 16 章 管理およびモニター

AMC を使用して、IBM Security Directory Integrator の構成および AssemblyLine をリモートで開始、停止および管理することができます。以下の情報を参照することで、AMC について詳しく知ることができます。

IBM Security Directory Integrator 管理およびモニター・コンソール (AMC) のユーザー・インターフェースは、Integrated Solutions Console (ISC) にデプロイされます。

IBM Security Directory Integrator には、AMC とともに Action Manager も同梱されています。Action Manager は、AMC データベースと対話し、リモート・サーバー API を使用してリモート AssemblyLine を管理するスタンドアロンの Java アプリケーションです。

管理およびモニター・コンソールは、ISC SE および IBM Dashboard Application Services Hub でデプロイできる Java WAR ファイル (Web アーカイブ) および WAB ファイル (Web バンドル) で構成されています。

現在の Action Manager は、IBM Security Directory Integrator AMC とバンドルされ、IBM Security Directory Integrator バージョン 7.1.1、7.1、および 7.0 をサポートしています。IBM Security Directory Integrator のバージョン 7.0 より前のバージョンはサポートされていません。

注: IBM Security Directory Integrator およびそれとともに開発、デプロイされたソリューションは、IBM Security Directory Integrator' の Java Management Extension (JMX) インターフェースを利用して、IBM Tivoli Monitoring (ITM) Server and Portal または IBM Netcool®/OMNIBus によってもモニターできます。この方法について示されたサポート例は、付録の 399 ページの『付録 B. 外部ツールのモニタリング』にあります。

インストールおよび構成

ここに示されているリンクを使用して、管理およびモニター・コンソールをインストールすることができます。

IBM Security Directory Integrator と管理およびモニター・コンソールのインストールについては、9 ページの『IBM Security Directory Integrator のインストール』を参照してください。AMC をインストールすると、Action Manager もインストールされます。カスタムの IBM Dashboard Application Services Hub を選択して AMC をデプロイしたり、インストール中に AMC の遅延デプロイメントを実行する場合は、デプロイメントの追加要件について、52 ページの『カスタム ISC SE または IBM Dashboard Application Services Hub への AMC のデプロイ』を参照してください。

Integrated Solutions Console への AMC のデプロイ

ここで説明する手順に従って、AMC を Integrated Solutions Console にデプロイすることができます。

このタスクについて

以下の説明は、IBM Security Directory Integrator のインストール手順に習熟しているユーザーを対象としています。IBM Security Directory Integrator のインストールの詳細については、13 ページの『プラットフォーム固有の IBM Security Directory Integrator インストーラーの使用』を参照してください。AMC をインストールすると、Action Manager もインストールされます。

ISC に IBM Security Directory Integrator 管理およびモニター・コンソールを自動的にデプロイしたい場合は、以下のいずれかの 1 つのオプションを選択してください。

- ISC SE の組み込みインスタンス
- ISC の既存のインスタンス

インストール時に AMC を ISC に自動的にデプロイせずに後で AMC を手動でデプロイする場合は、「指定しない。後で AMC を手動でデプロイする。」を選択します。

IBM Security Directory Integrator のインストール中に「ISC SE の組み込みインスタンス」または「ISC の既存のインスタンス」を選択した場合、 インストーラーは自動的に ISC をインストールし、その中で AMC をデプロイします。

管理およびモニター・コンソールを ISC SE にインストールし、デプロイするためには、以下のようにします。

1. IBM Security Directory Integrator インストーラーを起動します。
2. インストール時に、「**カスタム**」インストールを選択します。(標準インストールを選択すると、AMC オプションが提供されません。)
3. インストールの「機能の選択 (Select Features)」パネルで、「**AMC: 管理およびモニター・コンソール**」および「**組み込み Web プラットフォーム**」(Integrated Solutions Console、Standard Edition (ISC SE) を含む) を選択します。
4. IBM Security Directory Integrator のインストールを完了します。

IBM Security Directory Integrator インストーラーを使用した Windows サービスまたは UNIX プロセスとしての AMC のデプロイ

IBM Security Directory Integrator インストーラーを使用して、AMC を Windows サービスまたは UNIX プロセスとしてデプロイできます。

以下の条件を満たす場合、AMC を Windows サービスまたは UNIX プロセスとして登録できます。

- IBM Security Directory Integrator をインストールするユーザーは、管理権限 (Windows の管理者グループまたは UNIX のルート) を持っている必要があります。

- AMC を「ISC Standard Edition の組み込みインスタンス」にインストールすることを選択しました。
- 「AMC をシステム・サービスとして登録」することを選択し、サービスに名前を付けました。デフォルトのサービス名は「tdiamc」です。

既存の IBM WebSphere Application Server 環境への AMC のデプロイ

既存の IBM WebSphere Application Server 環境に AMC をデプロイすることができます。

既存の IBM WebSphere Application Server 環境に AMC をデプロイするには、`tdiISCHome.bat` または `tdiISCHome.sh` のスクリプト・ファイルを以下のパラメーターを設定するように変更します。

- **TDI_ISC_RUNTIME** パラメーターを IBM WebSphere Application Server に設定します。
- **TDI_ISC_HOME** パラメーターを WAS_HOME ディレクトリーに設定します。

例:

```
set TDI_ISC_RUNTIME=WAS
set TDI_ISC_HOME=C:¥Program Files¥IBM¥WebSphere¥AppServer
```

AMC および Action Manager の開始とログイン

管理およびモニター・コンソールと Action Manager は、ここで示すスクリプトを実行することにより、開始および停止することができます。

このタスクについて

スクリプトは、*TDI_install_dir*/bin/amc フォルダーに含まれています。

- AMC を開始するには、`start_tdiamc` スクリプトを実行します。
- AM を開始するには、`startAM` スクリプトを実行します。

これらのスクリプトについて詳しくは、325 ページの『AMC および AM コマンド行ユーティリティー』を参照してください。

上記では、ポート 1528 の localhost でネットワーク・モードで実行され、かつマシンでローカルに設定された Derby データベースを使用して AMC と AM が開始されます。AMC と AM の両方の代替の設定や構成について詳しくは、274 ページの『AMC の使用可能化』および 285 ページの『Action Manager の使用可能化』を参照してください。

管理およびモニター・コンソールが開始されると、以下の URL からアクセスすることができます。http://localhost:13100/ibm/console。詳細については、293 ページの『コンソールへのログインとログアウト』を参照してください。

AMC と AM の停止は、それぞれ `stop_tdiamc` および `stopAM` スクリプトを実行することにより行うことができます。

注:

1. ユーザーおよびユーザーの役割の追加について詳しくは、セクション 277 ページの『Integrated Solution Console の AMC』を参照してください。

2. AM について詳しくは、 279 ページの『Action Manager』を参照してください。
3. AMC の個々のパネルの使用方法について詳しくは、オンライン・パネル・ヘルプまたはセクション 293 ページの『管理とモニター・コンソールのユーザー・インターフェース』を参照してください。
4. AMC に IBM Security Directory Integrator サーバーを登録して、IBM Security Directory Integrator 構成からのソリューションを管理できるようにします。AMC は始動時に各 IBM Security Directory Integrator サーバーがロードする構成を検出することのみが可能です。デフォルトで、IBM Security Directory Integrator サーバーは、リモート・サーバー API を使用可能に設定しています。*TDI_install_dir*/configs フォルダーに自分が管理およびモニターしたい構成があることを確認してください (またはサーバーがソリューション・ディレクトリーを使用している場合は、ソリューション・ディレクトリーの「config」フォルダーにそれらを入れてください)。
5. いくつかの簡単なタスクに対する AMC と Action Manager の使用のウォークスルーの例については、 332 ページの『ソリューション・ビューおよびルールの作成のウォークスルー・サンプル』を参照してください。

AMC の使用可能化

ここにリストされている手順を参照して、管理およびモニター・コンソールを構成します。

管理およびモニター・コンソールの構成ファイルは、WEB-INF ディレクトリーと同じレベルにある amc.properties ファイルです。このファイルには、 AMC のデータベース構成プロパティー、LDAP プロパティー、SSL 関連プロパティー、およびヘルプ・サーバーの詳細が含まれます。

デフォルトでは、管理およびモニター・コンソールは、データを保管するために、Derby バージョン 10 を使用します。AMC の初回開始時に、AMC は Web サーバーのディレクトリー内に tdiamcdb フォルダーを作成し、AMC が機能するために必要なテーブルを作成します。Derby データベースには、ネットワーク・モードまたは組み込みモードのいずれかでアクセスできます。デフォルトでは、AMC と同梱される Derby は、ネットワーク・モードで構成されています。amc.properties の以下のプロパティーは、ネットワーク・モードに構成された Derby に適用されます。

```
com.ibm.di.amc.jdbc.database=jdbc:derby://localhost:1528/tdiamcdb;create=true
com.ibm.di.amc.jdbc.driver=org.apache.derby.jdbc.ClientDriver
com.ibm.di.amc.jdbc.urlprefix=jdbc:derby:
com.ibm.di.amc.jdbc.user=APP
com.ibm.di.amc.jdbc.password=APP
com.ibm.di.amc.jdbc.start.mode=automatic
com.ibm.di.amc.jdbc.host=localhost
com.ibm.di.amc.jdbc.port=1528
com.ibm.di.amc.jdbc.sysibm=true
```

プロパティー com.ibm.di.amc.jdbc.database は、localhost:1528 で実行中のネットワーク・モードの Derby をポイントしています。アクセスされるデータベースの名前は tdiamcdb で、create=true は、見つからない場合は AMC がデータベースを作成することを示します。

環境が設定されたら、create=true を create=false に変更し、データベース・パスが修正されたら、AMC がデータベースを再作成せずに、「データベースが見つか

らない」という例外がスローされるようにします。また、後でデータベース・パスに関する混乱が生じることを回避するために、データベースを絶対パスで設定することを推奨します。

Derby 以外のデータベースは、適切なプロパティーを設定することにより構成できます。 300 ページの『コンソール・プロパティー』を参照してください。

AMC には、Action Manager 用に個別の開始スクリプトとシャットダウン・スクリプトがあります。AMC では Action Manager をリモートで実行することが可能であり、個別の Derby 開始またはシャットダウン・スクリプトがあります。

管理およびモニター・コンソールは、組み込みモードの Derby データベースに接続するよう構成することもできます。この場合、AMC データベースとも対話する独立したアプリケーションである Action Manager は、AMC のデータベースと接続できません。これは、組み込みモードでは、一度に 1 つの JVM のみが Derby データベースに接続できるためです。以下の例は、Derby が組み込みモード用に構成された `amc.properties` ファイルを示しています。

```
##Location of the database (embedded mode)
configured for embedded mode:
##Location of the database (embedded mode)
com.ibm.di.amc.jdbc.database=tdiamcdb
com.ibm.di.amc.jdbc.driver=org.apache.derby.jdbc.EmbeddedDriver
com.ibm.di.amc.jdbc.urlprefix=jdbc:derby:
com.ibm.di.amc.jdbc.user=APP
com.ibm.di.amc.jdbc.password=APP
```

`com.ibm.di.amc.jdbc.database` プロパティーは、AMC データベースのロケーションをポイントしています。後でデータベース・パスに関する混乱が生じることを回避するために、この値を絶対パスに設定することを推奨します。

Action Manager をリモート側で実行する

Action Manager をリモート側で実行するために以下の手順を使用できます。その場合、示されている指示に注意してください。

IBM Security Directory Integrator 7.0 から、最初に AMC を開始せずに Action Manager をリモートで実行できるようになりました。AMC のデータベースである Derby に Action Manager が接続するには、Derby をネットワーク・モードで実行する必要があります。IBM Security Directory Integrator 7.0 にはまた、Derby データ・ストアの開始スクリプトとシャットダウン・スクリプトもあり、ユーザーは AMC を開始せずに、Action Manager をリモートで開始できます。

注:

1. 初めて、Action Manager を開始する前に、少なくとも一度、AMC を実行する必要があります。AMC が AM に必要なデータベース表を作成するためです。
2. このセクションのスクリプトは、リモート・コンピューターのインストール・ディレクトリーの *TDI_install_dir*¥bin¥amc¥ActionManager¥ フォルダーにあります。
3. この後の開始およびシャットダウン・セクションでは、異なるリモート・コンピューターで実行されていて、AMC を実行していない Action Manager および Derby について説明します。
4. AMC、Action Manager または Derby が停止したことを確認するには、ログを確認してください。

AMC および Action Manager の開始

AMC を実行しながら Action Manager および Derby を実行する場合は、「start_tdiamc.bat(sh)」と入力して AMC を開始し、「startAM.bat(sh)」と入力して Action Manager を開始します。tidamc スクリプトは、startNetworkServer.bat(sh) スクリプトをコールするため、ネットワーク・モードで Derby データベースを開始します。

注: startAM.bat(sh) スクリプトには、Action Manager により要求されるすべての jar に対して定義されたクラスパスがあります。CLASSPATH と DB_CLASSPATH という名前の 2 つの変数があります。 DB_CLASSPATH には、データベースとの JDBC コネクティビティーを確立するために必要な jar ファイルのパスごとのリストがあります。AMC が Oracle、MS SQL Server または DB2 を使用するように構成されている場合は、これらのデータベースの対応する JDBC JAR ファイルを DB_CLASSPATH 変数に追加する必要があります。

AMC および Derby のシャットダウン

stop_tdiamc.bat(sh) スクリプトは、stopNetworkServer.bat(sh) スクリプトをコールします。このことにより、AMC がシャットダウンするときに、Derby ネットワーク・サーバーが必ず停止します。

注: Action Manager (AM) が実行中の場合は、まず AM をシャットダウンします。

Action Manager のリモートからの開始

このセクションでは、異なるコンピューター上で Action Manager と Derby が実行中であることを想定しています。

1. スクリプト startNetworkServer.bat(sh) を使用して Derby を開始します。
2. Action Manager をスクリプト startAM.bat(sh) を使用して開始します。

startNetworkServer スクリプトは、ネットワーク・モードで Derby データベース・サーバーを開始するために使用されます。ネットワーク・モードの Derby サーバーは、ポート 1528 で開始されます。選択されるポートは、Derby のデフォルト・ポートとは異なります。

Action Manager のシャットダウン

Action Manager は、*TDI_install_dir*/bin/amc ディレクトリーにある stopAM.bat(sh) スクリプトを使用して、Action Manager を停止します。このスクリプトは、開始されている AM の processID を使用してそれを強制終了します。processID は startAM スクリプトにより取得され、ファイルに保管されます。保管された processID は、今度は stopAM スクリプトにより読み取られます。

Derby データベースを停止するには、ネットワーク・モードで Derby データベース・サーバーを停止する stopNetworkServer を入力してください。これは、AM を停止する前ではなく、停止した後に行うようにしてください。

AMC ログ

管理およびモニター・コンソールのログは AMC が実行される環境で ISC ログに保管されます。以下の情報を参照して、ログを構成することができます。

- ISC SE の場合、ログ・ファイルは ${LWI_HOME}/logs; の下に作成されます
- IBM Dashboard Application Services Hub の場合、ログは ${WAS_HOME}/profiles/${profileName}/logs/${serverInstance}/SystemOut.log に作成されます。

AMC のログの構成は、`WEB-INF/classes/logging.properties` ファイルを変更することによって実行できます。AMC のロギングは、Java ロギングの標準 (java.util.logging) に従います。

「**AssemblyLine ログ**」ウィンドウで AssemblyLine ログを表示したり、削除できます。「**状況のモニター**」ウィンドウで **AssemblyLine ログ**」を表示する方法: (「**状況のモニター**」>「**ソリューション・ビューの詳細**」>「**ログの表示**」)。. 「**ソリューション・ビューの詳細**」ウィンドウでは、ログを表示したい AssemblyLine を選択します。「**AssemblyLine のログ**」ウィンドウで、削除するログを選択し、「**削除**」を選択します。1 つまたは複数のログを削除できます。ログを表示するには、そのハイパーリンクをクリックしてください。

「**ソリューション・ビューの詳細**」ウィンドウはまた、「**Action Manager ログ**」テーブルも含みます。「**Action Manager ログ**」からログを選択したり、削除することができます。

「**ログ管理**」ウィンドウからすべてのログを管理することができます。ログの表示基準を指定したり、すべてまたは 1 つの AssemblyLine のログを削除することができます。日付範囲を指定して 1 つまたは複数の AssemblyLine のすべてのログを削除したり、*n* 個の最新のログを削除します。n は最新のログの数を示します。

Integrated Solution Console の AMC

ここでは、AMC の統合における変更点のリストを参照することができます。また、以下の情報を参照することで、ユーザーの追加と削除、および割り当てることができるさまざまな種類の役割について学習できます。

Integrated Solutions Console (ISC) は、業界標準のテクノロジーを使用して管理コンソール機能を編成するための共通のコンソールを提供するために設計されました。IBM Security Directory Integrator 7.0 から、 Integrated Solutions Console (ISC) への AMC の統合では、以下の変更が行われました。AMC の主なナビゲーション・リンクは、以下のとおりです。

- **管理およびモニター・コンソール**
  - サーバー
  - ソリューション・ビュー
  - 状況のモニター
  - Action Manager
- **拡張**
  - ログ管理
  - コンソール・プロパティー

– 優先ソリューション・ビュー
– プロパティー・ストア

コンソールのユーザー権限

ISC **コンソールのユーザー権限**を使用して、AMC のユーザーを追加したり、削除することができます。IBM Security Directory Integrator v7.0 以降の AMC では、以下の役割が設定できます。

表 28. AMC の役割

user 役割	**説明**
administrator	この役割を割り当てられたユーザーは、他のユーザーに割り当てられた役割を構成することができます。
iscadmins	この役割を割り当てられたユーザーは、ISC コンソール自体の設定を管理する能力をもちます。
Security Directory Integrator AMC Admin	この役割は、ISC コンソールでデプロイされる IBM Security Directory Integrator AMC アプリケーションにより考慮されます。この役割を割り当てられたユーザーは、サーバー、ソリューション・ビューの役割、コンソール・プロパティーの管理を設定できます。 (これは、IBM Security Directory Integrator 7.0 以前は AMC の superadmin 役割でした。)
Security Directory Integrator AMC User	この役割は、ISC コンソールでデプロイされる IBM Security Directory Integrator AMC アプリケーションにより考慮されます。この役割を割り当てられたユーザーは、IBM Security Directory Integrator AMC Admin リソースにより提供されたものを使用することができます。

SDI AMC user 役割の中で、ユーザー特権には以下のソリューション・ビューにある役割が割り当てられます。

- 管理 (Admin)
- 構成の管理 (Config Admin)
- 実行 (Execute)
- 読み取り (Read)

この役割は、コンソールの機能へのアクセスを制御します。役割が割り当てられた機能のみを表示することができます。例えば、Security Directory Integrator AMC Admin 役割があるユーザーには、自動的に、すべてのソリューション・ビューに対して管理者権限があります。管理者は、Web 管理ツールに必要なプロパティーを構成 (左側のナビゲーション・ペインの「コンソール・プロパティー」から選択できる`amc.properties` ファイルに関連するプロパティーを修正) できます。ISC の Security Directory Integrator AMC User 役割のあるユーザーは、現在の管理権限のない AMC ユーザーと同じです。Security Directory Integrator AMC User は、IBM Security Directory Integrator サーバーおよびコンソール・プロパティーなどの管理ウィンドウにはアクセスできません。

ISC では、AMC Admin Group または iscadmins を利用できます。 iscadmins グループのユーザーは、管理者と同じ権限を持ちます。

管理者および iscadmins グループ

管理者は、アプリケーションをインストールしたユーザーとして定義されていますが、AMC ユーザーを管理することができます。管理者はローカルの OS レジストリーから AMC アプリケーションにユーザーを追加または削除することができ、役割を割り当てたり、編集することができます。IBM Security Directory Integrator v7.0 以降では、新しい **SDI AMC Admin** グループが提供されています。 superadmin 役割は、IBM Security Directory Integrator バージョン 6.1.1 以降では存在しません。

Action Manager

Action Manager は、AMC で定義されているユーザー定義ルール、トリガー条件、およびアクションを使用して複数の IBM Security Directory Integrator サーバーと AssemblyLine 実行をモニターするスタンドアロン Java アプリケーションです。ここでは、さまざまなトリガー・タイプ、アクション、およびスレッドのリストを参照できます。

管理およびモニター・コンソール (AMC) には、さまざまな Action Manager ルールを構成できる「AM の構成」パネルがあります。

ルールは、トリガー・タイプおよび関連付けられたアクションのセットの組み合わせです。ルールは、トリガー条件が検出されたときは、関連付けられたアクションのセットを実行する必要があることを指定します。AMC で使用可能なトリガー・タイプを以下で説明します。

表 29. Action Manager トリガー

トリガー・タイプ	トリガーの詳細	トリガーのユーザー入力
トリガーなし	このトリガー・タイプのルールにはトリガー条件がありません。結果として、このトリガー自体により起動されることはありません。このルールを実行できるのは、他のルールでこのルールが実行される場合のみです。	詳細は不要です。
AssemblyLine 開始時	このトリガー・タイプのルールは、Action Manager が特定の AL の AL 開始イベントを受信すると起動されます。	「AssemblyLine 名」
AssemblyLine 終了時	このトリガー・タイプのルールは、Action Manager が特定の AL の AL 終了イベントを受信すると起動されます。	「AssemblyLine 名」
構成のロード時	このトリガー・タイプのルールは、構成がロードされると起動されます。	「ロードする構成の名前」
構成のアンロード時	このトリガー・タイプのルールは、構成がアンロードされると起動されます。構成はロードされている場合にアンロードできます。	「アンロードする構成の名前」

表 29. Action Manager トリガー (続き)

トリガー・タイプ	トリガーの詳細	トリガーのユーザー入力
AssemblyLine 結果の照会時	**注:** このトリガー・タイプをもつルールは、短時間実行される AL とともに使用できません。これは、 Action Manager は「AL の開始」イベントの受信時に AssemblyLine オブジェクトのハンドルを保管するためです。後で「AssemblyLine の停止」イベントを受信すると、Action Manager はこのハンドルを使用して、最終作業項目属性を照会します。Action Manager がハンドルを保管する前に AL が終了すると、Action Manager は作業属性を照会できなくなります。通常、実行時間は 10 秒で十分です (この実行時間を実現するには、AL 終了前に system.sleep(10) を **epilog** フックなどに挿入します)。 「**AL 結果の照会時**」に実行すると、Action Manager は特定のポーリング間隔で連続して AL をポーリングします。トリガーは最初に属性値を、特定のポーリング間隔後の AL から確認します。次に、トリガーは AL 結果エントリーをチェックします。 「結果の照会時」のルールでは、指定された AL の最後の「作業」項目に含まれている指定の「属性」が、指定の「条件」および「値」に一致すると起動されます。この条件が検査されるのは、ActionManager が「AssemblyLine の停止」イベントを受信した時点でのみです。このユーザーは時間間隔を指定できます。指定された AL は、指定されたルールに応じて、定期的に実行されます。このトリガー・タイプのルールは、指定された AL の最後の「作業」項目に含まれている指定の「属性」が、指定の「条件」および「値」に一致するときに起動されます。この条件が検査されるのは、Action Manager が「AssemblyLine の停止」イベントを受信した時点のみです。 「**結果の照会時**」トリガーを構成するには、以下のフィールドに値を入力してください。 • AssemblyLine 名 • 属性 • 条件 • 値 • ポーリング間隔 • ポーリング単位	「AssemblyLine 名」、「属性」、「条件」、「値」、「ポーリング間隔」および「ポーリング単位」

表 29. Action Manager トリガー (続き)

トリガー・タイプ	トリガーの詳細	トリガーのユーザー入力
サーバーでの API 障害時	このトリガー・タイプのルールは、Action Manager がサーバー API を使用してリモート・サーバーに接続できないときに起動されます。AL の実行時間に応じて、それぞれの AssemblyLine に対して異なるポーリング時間間隔を構成できます。	「ポーリング間隔」および「ポーリング単位」。
イベント受信時	このトリガー・タイプのルールは、Action Manager が指定された条件を満たすイベントを受信すると起動されます。 **注:** いずれかの条件を無視する場合は、その条件をブランクにしてください。	「イベント・タイプ」、「イベント・ソース」、「イベント・データ」。「イベント・データ」はオプションです。「イベント・タイプ」または「イベント・ソース」のいずれか 1 つを指定する必要があります。
プロパティーの場合	このトリガー・タイプのルールは、指定されたプロパティーが指定された条件を満たすと起動されます。Action Manager はこのプロパティーを定期的に検査します。このトリガーを構成するときに、ポーリング間隔とポーリング単位を構成できます。 **注:** このルールは 1 回だけ起動されます。このプロパティーが現在指定の条件に一致していないことが Action Manager によって検出された場合にのみ、作動可能状態にリセットされます。これは、トリガー条件が発生するたびにルールが繰り返しトリガーされないようにするためです。	「ポーリング間隔」、「ポーリング単位」、「プロパティー名」、「条件」、および「値」。
ローカル変数時	このトリガー・タイプのルールは、指定された変数が指定された条件を満たすと起動されます。Action Manager はこのプロパティーを定期的に検査します。 **注:** このルールは 1 回だけ起動されます。この変数が指定の条件に一致していないことが Action Manager により検出されると、作動可能状態にリセットされます。これは、トリガー条件が発生するたびにルールが繰り返しトリガーされないようにするためです。	「ローカル変数」、「条件」、「値」。

表 29. Action Manager トリガー (続き)

トリガー・タイプ	トリガーの詳細	トリガーのユーザー入力
AssemblyLine 終了コードの検査	このトリガー・タイプのルールは、AssemblyLine がエラーで終了すると起動されます。「**AssemblyLine 終了コードの検査**」はまた、AL のすべての異常終了について、エラー・オブジェクト・ストリングも検索します。「トリガーの構成」の下で、トリガーが「**AssemblyLine 終了コードの検査**」の場合、「 **エラー・オブジェクトの検査**」を有効にすることができます。 「値」フィールドで、エラー・オブジェクトに該当するストリングを入力してください。「値」フィールドが空の場合、このルールは AL の異常終了が発生するたびに起動します。「エラー・オブジェクトの検査」が選択されていない場合、トリガーは AL の終了を待機し、終了コード中で、(ユーザーが) 「属性」フィールドに入力した値を検査します。属性名と値の両方の値を入力します。 「AssemblyLine 終了コードの検査」トリガーで、Action Manager は AL を開始しなくなるため、ポーリングは行われません。このトリガーは AL の実行後に 1 回だけ AL の結果を検査します。	「AssemblyLine 名」。「エラー・オブジェクトの検査」が有効な場合は、「値」のみを指定する必要があります。「エラー・オブジェクトの検査」が無効な場合は、「属性」、「条件」、および「値」の値を指定する必要があります。
最終実行からの時間	このトリガー・タイプのルールは、Action Manager が指定の AssemblyLine が指定の期間に実行されていなかったことを検出すると起動されます。注: このルールは 1 回のみ起動されます。その後、Action Manager はこのルールが作動可能モードにリセットされるまで「AssemblyLine の開始」イベントの受信を待機します。これは、トリガー条件が発生するたびにルールが繰り返し起動されないようにするためです。	「AssemblyLine 名」、「最終実行時」、および「単位」。
タイマー	このトリガー・タイプをもつルールは、単位数、および秒、分、時、日のいずれかの単位により定義される間隔内で、継続的に起動されます。	「間隔」および「単位」。

ルールが起動されると、Action Manager はそのルールに関連付けられているアクションを順次実行します。AMC で使用可能な各種アクションを以下に示します。

表 30. Action Manager のアクション

アクション	アクションの詳細	アクションのユーザー入力
AssemblyLine の開始	このアクションは、指定の IBM Security Directory Integrator サーバーで指定の構成ファイルの指定 AL を開始します。「構成」フィールドには、リモート・サーバーの構成の完全なパスが指定されている必要があります。「構成パスワード」フィールドはオプションですが、リモート構成がパスワードで保護されている場合は指定する必要があります。	「AssemblyLine 名」、「構成 (Of Configuration)」、「サーバー (On Server)」、および「構成パスワード」。

表 30. Action Manager のアクション (続き)

アクション	アクションの詳細	アクションのユーザー入力
AssemblyLine の停止	このアクションは、指定の IBM Security Directory Integrator サーバーで指定の構成の指定 AL を停止します。「構成」フィールドには、リモート・サーバーの構成の完全なパスが指定されている必要があります。	「AssemblyLine 名」、「構成 (Of Configuration)」、および「サーバー (On Server)」。
ルールの使用可能/不可設定	このアクションは、選択されているルールを使用可能または使用不可にします。	「ルール名」、「状態」
ルールの実行	このアクションは、指定のルールを実行します。これにより、実行されたルールに指定されているすべてのアクションが間接的に実行されます。	「ルール名」
イベントの通知	このアクションにより、Action Manager はイベントと指定された詳細を、現行ソリューション・ビューに関連付けられているサーバーに出力します。詳しくは、Session.sendCustomNotification() API を参照してください。	「イベント・タイプ」、「ソース」、および「データ」。
プロパティーの変更	このアクションにより、Action Manager は指定された操作に基づいて、選択されているプロパティーを変更します。	「プロパティー」、「操作」、および「値」。
プロパティー値のコピー	このアクションにより、Action Manager はソース・プロパティーの値を宛先プロパティーにコピーします。	「コピー元のプロパティー」、「コピー先のプロパティー」
ログへの書き込み	このアクションにより、指定された重大度/メッセージ/記述のログが Action Manager のログと AMC データベースに記録されます。ユーザーが「状況のモニター」->「ソリューション・ビューの詳細」->「AM 結果表」を選択すると、この同じログが表示されます。各ルールに常に 1 つ以上のログ・アクション (説明テキストを含む) を指定することをお勧めします。	「重大度」、「メッセージ」、「記述」
E メールの送信	このアクションにより、電子メールが指定した宛先に送信されます。ユーザーは電子メールの内容を指定します。メールの送信前に内容とともに、その他の詳細が Action Manager により示されます。内容の入力域および件名のラインでは、変数 %EVENT_DATA% の値を指定できます。%EventData% を指定すると、E メールの送信時に Eventdata 変数の実際の値が挿入されます。%Action_Error% も同じようにここで置換できます。「Action Manager ログの添付」が有効になっている場合、Action Manager のログ (am_logging.properties ファイルで指定) は電子メール添付ファイルとして送信されます。	「宛先」、「送信者」、「件名」、「Action Manager ログの添付」(選択/非選択)、「内容」。
ローカル変数の変更	このアクションにより、Action Manager は、指定された変数の値を増分、減分、または指定された値に設定します。	「変数」、「操作」、「値」。

表 30. Action Manager のアクション (続き)

アクション	アクションの詳細	アクションのユーザー入力
コマンドの実行	このアクションにより、指定されたコマンドはターゲット・コンピューターで実行されます。コマンドは、任意の汎用コマンドまたは IBM Security Directory Integrator 固有コマンドにすることができます。	「ターゲット・マシン」、「ポート」、「ユーザー名」、「パスワード」、「鍵ストア」、「鍵ストアのパスワード」、「プロトコル」、「コマンド」。

AMC でソリューション・ビューについて構成されたルールは、AMC の Derby データベースに保管されます。Action Manager は、実行されるとネットワーク・モードの AMC データベースに接続し、Action Manager の関連テーブルを読み取って、指定された各ルールについてメモリーにスレッドを作成します。これらの各スレッドは、それぞれのトリガー条件を listen/ポーリングします。スレッドは、そのトリガー条件の出現を検出すると、データベースに対してルールに関連付けられたアクションのセットを照会し、それを順次実行します。

Action Manager は、トリガー条件を listen するルール・スレッドの他に、以下のスレッドを実行します。

1. HealthAssemblyLine - ヘルス AssemblyLine スレッドは定期的に ヘルス AL を起動し、ソリューションの状況を照会し、状況を AMC データベースに記録します。ヘルス AL は、その最終作業項目の「healthAL.result」および「healthAL.status」属性に状況を保管する必要があります。
2. ServerStatusListener - ServerStatusListener スレッドは、AMC に登録されているサーバーごとに作成されます。このスレッドは、サーバーにアクセス可能であるかどうかを検査します。サーバーがアクセス不能になると、このサーバーに対して作成されたすべてのルール・スレッドが終了します (ただしトリガー・タイプが「サーバーでの API 障害時」のルールを除く)。同様に、サーバーがアクセス可能になると、このサーバーに関連するすべてのルールのルール・スレッドが作成されます。
3. ConfigLoadReloadListener - ConfigLoadReloadListener スレッドは、AMC に登録されている実行サーバーごとに作成されます。このスレッドは、構成ロード/アンロード・イベントに備えてリモート・サーバーに登録されます。ルール・スレッドは、構成イベントに基づいて適切に終了、作成、または更新されます。
4. ServerModificationListener - ServerModificationListener スレッドは、AMC に登録されている一連のサーバーに対する更新があるかどうかを検査します。変更のタイプ (追加、削除など) に基づいて、ルール・スレッドが終了、作成、または更新されます。
5. DatabaseModificationListener - ルールの追加、変更、または削除を継続的にモニターするデータベース・リスナー・スレッドです。ルールの変更が検出された場合は必ず、実行時に Action Manager スレッドが適切に追加/再作成されます。

Action Manager はまた、その実行の詳細により AMC データベースを更新します。Action Manager ルールが起動されると必ず、Action Manager は AMC データベースに項目を記録し、起動されたルール名およびトリガー時刻を登録します。また、ルールについてログ・アクションが構成されている場合は、これも AMC データベースに記録されます。これらのデータベース項目は、AMC のモニター・ウィンドウに適切な状況を表示するために使用されます。

Action Manager の使用可能化

以下の指示に従って、 Action Manager を使用可能にすることができます。

Action Manager は、*TDI_install_dir*/bin/amc/Action Manager フォルダーにインストールされます。このフォルダーには、以下のファイルが含まれます。

- am_logging.properties - このファイルは Action Manager ロギング・プロパティーを制御します。AMC と同様に、これも java.util.logging ロギング標準に準拠します。
- am_config.properties - これは Action Manager の構成ファイルです。
- testadmin.jks - Action Manager のトラストストア/鍵ストア・ファイルです。

  注: これは、サンプルのトラストストア/鍵ストア・ファイルです。セキュリティーを強化するには、トラストストア/鍵ストア・ファイルを各自で作成してください。

Action Manager は、ネットワーク・モード・ドライバーを使用して、AMC の Derby データベースに接続します。

am_config.properties 内の以下のプロパティーは、管理およびモニター・コンソールのデータベースを指す必要があります。

```
com.ibm.di.amc.am.jdbc.database=jdbc:derby://localhost:1528/C:/Program Files/IBM/AppSrv
/profiles/amcprofile/tdiamcdb;create=false
com.ibm.di.amc.am.jdbc.driver=org.apache.derby.jdbc.ClientDriver
com.ibm.di.amc.am.jdbc.urlprefix=jdbc:derby:
com.ibm.di.amc.am.jdbc.user=APP
{protect}-com.ibm.di.amc.am.jdbc.password=APP
com.ibm.di.amc.am.jdbc.start.mode=automatic
com.ibm.di.amc.am.jdbc.sysibm=true
com.ibm.di.amc.am.jdbc.networkserver.host=localhost
com.ibm.di.amc.am.jdbc.networkserver.port=1528
```

注: AMC および AM は両方とも、MS SQL、 Oracle などの代替データベースをサポートしています。AMC および AM でこれらの代替データベースの 1 つに接続する場合、amc.properties および am_config.properties の構成ステートメントの表示は非常に異なります。

Action Manager が開始されると、 AMC のデータベースへの接続が試行されます。初期セットアップ・タスクの実行に失敗した場合、例外メッセージを出して終了します。am_config.properties ファイルをチェックし、これが正しいデータベースを指していることを確認してください。データベース設定が正しいと思われる場合は、Action Manager が接続するデータベースがネットワーク・モードで稼働しており、AMC がこのデータベースに接続できることを確認してください。Derby DB をネットワーク・モードで始動するには、startNetworkServer.bat(sh) を使用します。

AM の SSL 設定および暗号化プロパティーは、以下のプロパティーのセットで構成されます。

```
# Action Manager SSL properties
javax.net.ssl.trustStore=TDI_Install_dir/serverapi/testadmin.jks
{protect}-javax.net.ssl.trustStorePassword=administrator
javax.net.ssl.trustStoreType=jks
javax.net.ssl.keyStore=TDI_Install_dir/serverapi/testadmin.jks
{protect}-javax.net.ssl.keyStorePassword=administrator
javax.net.ssl.keyStoreType=jks
# Action Manager encryption properties
com.ibm.di.amc.am.encryption.keystore = TDI_Install_dir/testserver.jks
```

```
com.ibm.di.amc.am.encryption.key.alias = server
com.ibm.di.amc.am.encryption.keystoretype = jks
com.ibm.di.amc.am.encryption.transformation = RSA
com.ibm.di.amc.am.stash.file = TDI_Install_dir/idisrv.sth
```

これらのプロパティーは、サーバーにより使用される暗号化プロパティーに似ています。便宜上、stash ファイルの場所は、プロパティー com.ibm.di.amc.am.stash.file として追加されています。デフォルトでは、AM の保護されたプロパティーの暗号化/暗号化解除のために、AM ではサーバーの鍵ストアと stash ファイルを再使用します。

ランタイム・ルール、トリガー、およびアクションの詳細な構成については、313 ページの『Action Manager』で説明します。

リアルタイムの Action Manager の状況

ここに示される指示に従って、Action Manager の状況をウィンドウに表示することができます。また、ウィンドウの内容も参照できます。

AMC にログインすると、1 行の「Action Manager 状況」が「**AMC へようこそ**」パネルに表示されます。「AMC へようこそ」パネルでは、リンクで Action Manager 状況 (例えば、「Action Manager は実行中です」または「Action Manager は実行されていません」など) が表示されます。「**Action Manager 状況**」ウィンドウを起動するには、ハイパーリンクをクリックします。「Action Manager 状況」ウィンドウでは、Action Manager 状況が、スレッド詳細およびトリガー詳細とともにリアルタイムで表示されます。このウィンドウでは、状況情報がリアルタイムで表示されます。このウィンドウには、以下の項目が表示されます。

- Action Manager 状況。例えば、ブート時間
- Action Manager のスレッド詳細
- Action Manager トリガーの詳細

AMC は Action Manager により公開される API を使用して Action Manager を直接照会します。AMC が Action Manager とのセッションを確立できない場合、AMC では Action Manager は停止しているために使用不可になっていると判断します。Action Manager 状況に加えて、AMC ではスレッド情報の詳細およびトリガーの詳細が表示されます。

Action Manager は多くのスレッドを作成します。Action Manager スレッドの中には、Database Modification Listener や ServerStatusListenerThread など、Action Manager の重要な機能をモニターするものもあります。さらに、Action Manager はこれらのスレッドから、AMC で構成される各トリガー・ルールに対してスレッドを作成します。リモート・メソッド呼び出し (RMI) レイヤーを使用すると、AMC は、さまざまなトリガー関連のスレッドの状態を照会できます。RMI ベースの照会を使用して、AMC はこれらのスレッドの状態、スレッド優先順位などを認知します。AMC はまた、一定期間に実行されたトリガーも照会できます。

2 個の新しいプロパティーが、「**Action Manager 状況のリアルタイム表示**」要件に属します。AMC が Action Manager 状況をリアルタイムで表示することを可能にするプロパティーは、am.api.host および am.api.port です。 Action Manager の状況では、Action Manager にその状況を照会するために、AMC で使用される API を公開する Action Manager 周辺の RMI レイヤーを使用しました。

指定したルールに対して、AMC でトリガーを強制する

「**トリガーの強制**」を使用して、選択したルールに対して構成されているアクションを実行できます。

AMC では、ユーザーはルールのトリガーを強制できます。トリガーを強制すると、実際にルールが起動されるときに Action Manager が何をするかをユーザーが知ることができます。「**ルールを使用不可に設定**」を選択すると、選択されたルールは使用不可になります。

Action Manager は、特定のトリガー (ルール) 内に明示的に構成されているアクションのセットを実行できます。AMC ユーザーは、構成されているアクションを実行する前に、そのルールのトリガー条件が満たされるまで待つ必要はありません。ユーザーは、実行されるべきアクションについて、それらのアクションをユーザーがテストできるように定義できます。ユーザーは、「**トリガーの強制**」を使用してサポートされたすべてのアクションを実行できます。しかし、「**復帰**」アクションは、一部のサポートされたアクション (サブセット) でのみ有効です。

- プロパティーの変更
- プロパティーのコピー
- ログへの書き込み
- ルールを使用可能/使用不可に設定する

AMC および Action Manager のセキュリティー

管理およびモニター・コンソール (AMC) は、リモート IBM Security Directory Integrator ソリューションをモニターして管理するための Web ベースのアプリケーションです。以下の情報を参照することで、その機能とさまざまなセキュリティーの組み合わせについて学習できます。

概要

AMC の以下の機能が改善されました。

- amc.properties ファイルにストアされたパスワードの暗号化または隠匿
- 鍵ストア・パスワードを保管するための stash ファイルの使用
- Derby データベースの BUILTIN 認証スキームの使用可能化

AMC は IBM Security Directory Integrator と通信するために、リモート・サーバー API を使用します。このため、 IBM Security Directory Integrator リモート・サーバー API クライアントに適用可能なセキュリティーの制限および構成の設定 (前のセクションで説明) はすべて AMC についても有効です。

Action Manager は AMC とともにインストールされます。Action Manager は、AMC ユーザーが設定した AMC データベース内のルール・セットに基づいて自らを構成し、動作します。リモート AssemblyLine をモニターし、構成済みのルールに基づいてアクションを実行するために、Action Manager は、AMC と同様に IBM Security Directory Integrator サーバーと通信するために、IBM Security Directory Integrator リモート・サーバー API を使用します。

注: RMI を使用した AMC と AM の通信はいかなる方法でも保護されていません。

AMC および SSL

ここに示されている指示に従って、管理およびモニター・コンソールを SSL モードで実行することができます。

AMC には複数の IBM Security Directory Integrator サーバーを登録することができます。各 IBM Security Directory Integrator サーバーはそれぞれ異なった構成にすることができます。1 つの IBM Security Directory Integrator サーバーは SSL をオフにし、1 つは SSL をオンにし、別の 1 つはカスタム認証および SSL をオンにするなどしてそれぞれ実行することができ、他のさまざまな組み合わせも可能です。AMC を使用して、これらのサーバーに同時に接続したり、管理したりすることができます。先述のとおり、IBM Security Directory Integrator を SSL モードで実行するために構成するには、`global.properties` (または `solution.properties`) で、**api.remote.ssl.on** プロパティーを **true** に設定する必要があります。

AMC は Web コンテナー内で実行される Web アプリケーションであるため、Web コンテナーから自動的に特定のプロパティーおよびセキュリティーの制限を継承します。例えば、Web コンテナーで SSL 鍵ストアまたは SSL トラストストアが構成されている場合、これは自動的に AMC に適用できます。ただし、AMC でこれを指定変更し、独自の鍵ストアおよびトラストストアを指定することもできます。

SSL 上で実行されている IBM Security Directory Integrator リモート・サーバー API と通信できるようにするには、AMC には IBM Security Directory Integrator リモート・サーバー API によって信頼された証明書を含む構成済みの鍵ストア (つまりこれは、IBM Security Directory Integrator のトラストストアの信頼された証明書セクションになければなりません) と、IBM Security Directory Integrator リモート・サーバー API によって送信された証明書を含む構成済みのトラストストアが必要です。言い換えると、IBM Security Directory Integrator サーバーの鍵ストアにある証明書は AMC のトラストストアになければならず、IBM Security Directory Integrator のトラストストアにある証明書は AMC の鍵ストアになければなりません。

例えば、IBM Security Directory Integrator のデフォルトのインストールには、特定のストア (.jks ファイル) が同梱されています。IBM Security Directory Integrator を SSL モードで実行する場合、AMC に接続するには、その鍵ストアとトラストストアが同じ値 *TDI_install_dir*`/serverapi/testadmin.jks` に設定され、パスワードが「administrator」である必要があります。testadmin.jks には信頼された証明書および署名者証明書の両方が含まれるため、接続が確立されます。固有の SSL 鍵ストアとトラストストアをセットアップすることをお勧めします。

AMC では、AMC に「SDI AMC Admin」(コンソール管理者) としてログインし、**「拡張」->「コンソール・プロパティー」->「SSL の設定」**ウィンドウにナビゲートすることにより、トラストストアおよび鍵ストアのパスを設定できます。トラストストアおよび鍵ストアの設定は、Web コンテナーの tdiamc フォルダー内の **amc.properties** ファイルに書き込まれます。あるいは、**amc.properties** ファイルを直接編集することも選択できます。IBM Security Directory Integrator 7.0 では、

AMC を ISC Standard Edition (SE) または ISC Advanced Edition (AE) にデプロイできます。ISC ランタイムに応じて、testadmin.jks ファイルの場所は変わります。例えば、AMC が ISC SE でデプロイされた場合、ロケーションは ISC_RUNTIME_INSTALL_DIR/runtime/isc/eclipse/plug-ins/AMC_7.0.0 になります。一方で、AMC が IBM Security Directory Integrator でデプロイされた場合、ロケーションは ISC_RUNTIME_INSTALL_DIR/systemApps/isclite.ear/tdiamc.war になります。デフォルトでは、鍵ストアとトラストストアのパスワードは「administrator」に設定されています。リモート IBM Security Directory Integrator サーバーとの SSL ベースの接続を確立するには、サーバーを「SSL 使用可能」モードで始動する必要があります。非 SSL ベースの接続の場合は、サーバーを「SSL 使用不可」モードで始動します。

重要: デフォルトの SSL 設定が提供されています。ただし、デフォルトの証明書を使用した場合、非暗号化接続よりもセキュリティーが強化されるわけではないので、インストール後にデフォルトの SSL 証明書を置換し、鍵ストアとトラストストアをそれぞれ更新して、セキュリティーを強化する必要があります。

SSL 上で実行中の各 IBM Security Directory Integrator サーバーを AMC に登録するには、必要な証明書を AMC のトラストストアに、必要な AMC の鍵証明書を IBM Security Directory Integrator のトラストストアにインポートする必要があります。これは、セキュアな両方向 SSL 接続を確立できるようにするには、AMC が IBM Security Directory Integrator を信頼し、IBM Security Directory Integrator が AMC を信頼する必要があることを意味しています。

AMC は Web コンテナーの内部で実行されるため、AMC の URL は http://hostname:port/ibm/console です。

Action Manager は、AMC で構成されたルールに基づいて、リモート IBM Security Directory Integrator サーバー上で実行中の構成および AssemblyLine をモニターします。Action Manager には、リモート IBM Security Directory Integrator サーバーに接続するために必要な鍵ストアおよびトラストストアが用意されています。SSL プロパティーは am_config.properties で定義されています。SSL についての AMC の構成方法の詳細は前のセクションを参照してください。同じことが Action Manager にも適用されます。

AMC およびリモート IBM Security Directory Integrator サーバー

AMC はリモートで複数の IBM Security Directory Integrator サーバーに接続できます。ここでは、さまざまな方法での各サーバーの構成について学習できます。

構成方法:

- 非 SSL
- SSL
- 非 SSL によるカスタム認証
- SSL によるカスタム認証

このセクションは、これらの事例のそれぞれについて、詳細に説明します。

リモート IBM Security Directory Integrator サーバーで非 SSL が構成されている (すなわち、api.remote.ssl.on=false) 場合、AMC の鍵ストアまたはトラストストアは、正しく構成されていても機能しません。これは、SSL 接続が試行されていないためです。この場合、AMC サーバーのコンピューター IP アドレスを IBM Security Directory Integrator サーバーに登録する必要があります。これは、global.properties (または solution.properties) ファイルを編集して実行します。更新するプロパティーは、**api.remote.nonssl.hosts** です。リモート IBM Security Directory Integrator サーバーの global.properties ファイルに AMC コンピューターの IP アドレスを入力すると、AMC はその特定のサーバーに接続できるようになります。これは、TDI サーバーは、その **api.remote.nonssl.hosts** プロパティーに IP アドレスが設定されているコンピューターからのリモート・サーバー接続 (AMC 接続) のみを信頼することを示します。

注: IBM Security Directory Integrator サーバーが AMC と同じコンピューター上で実行されている場合、このプロパティーの編集は不要です。
リモート IBM Security Directory Integrator サーバーで SSL が構成されている (すなわち、**api.remote.ssl.on=true**) の場合、AMC の SSL 鍵ストアおよびトラストストアを適切にセットアップする必要があります。

この詳細については、AMC および SSL に関する前のセクションを参照してください。SSL または非 SSL が構成されていることに加えて、リモート IBM Security Directory Integrator サーバーはカスタム認証も要求することがあります。カスタム認証では、リモート IBM Security Directory Integrator サーバーへの接続時にユーザー名およびパスワードを渡す必要があります。リモート IBM Security Directory Integrator サーバーは、このユーザー名およびパスワードを LDAP、ファイル、データベース、スクリプトなどのサード・パーティーのリポジトリーに対して妥当性検査し、サーバー API クライアントからの接続を許可するかどうかを判断します。このような場合、「認証モード」ウィンドウで AMC にサーバーを登録する (「**サーバーの管理**」->「**サーバーの変更**」) 際に、「**カスタムまたは LDAP 認証**」を選択し、指定されたリモート IBM Security Directory Integrator サーバーへの接続を試行するたびに AMC が渡す必要のあるユーザー名およびパスワードを設定します。

注: ユーザー名またはパスワード (カスタム認証の場合) または SSL 鍵ストアまたはトラストストア (SSL の場合) が正しくセットアップされていない場合、AMC はリモート IBM Security Directory Integrator サーバーに接続できず、そのサーバーは「停止」または「実行されていない (Not running)」状態であるとして表示されます。

AMC および役割の管理

AMC 内のすべてのユーザー (またはグループ) に、特定のソリューション・ビューについて、AMC 内で役割を割り当てることができます。以下の情報を参照することで、使用可能な役割とそれらの意味について学習できます。

この役割の割り当ては、「**ソリューション・ビュー**」ウィンドウで、特定のソリューション・ビューを選択し、「**ACL の構成**」をクリックすることによって行うことができます。「**ACL の構成**」ウィンドウが表示されます。構成するユーザーの名前を選択し、ツールバーの「**ユーザーの構成**」をクリックします。「**ユーザーの構成**」ウィンドウが表示されます。「**ユーザー ID**」を選択し、選択可能な役割の 1 つを選択します。

- 読み取り (Read)
- 実行 (Execute)
- 管理 (Admin)
- 構成の管理 (Config Admin)

注: 「自動更新」オプションを使用して作成したソリューション・ビューを再ロードする必要があります。これには「**ソリューション・ビュー**」ウィンドウにある「**ソリューション・ビューのリフレッシュ**」を使用します。自動更新としてマークされたソリューション・ビューでは、構成ファイルを再ロードし、「**ソリューション・ビューのリフレッシュ**」をクリックしてソリューション・ビューをリフレッシュする必要があります。「**単純**」オプションを使用して作成され、自動更新が選択されたソリューション・ビューをリフレッシュしないと、AMC データベースでソリューション・ビューに不整合が発生する場合があります。ソリューション・ビューを更新しなかったために不整合が発生すると、Action Manager が誤った動作をする可能性があります。

これらの役割は上記の順で権限が増加します。つまり、「構成の管理 (Config Admin)」が最高の権限を持ち、「読み取り (Read)」の権限が最低になります。ソリューション・ビューに対する「読み取り (Read)」役割を持つユーザーが使用可能な機能は、そのソリューション・ビューに対する「実行 (Execute)」権限を持つユーザーにとっても必ず使用可能です。ソリューション・ビューに対する「実行 (Execute)」権限を持つユーザーが使用可能な機能は、「管理 (Admin)」権限を持つユーザーにとっても使用可能です。

これらの役割の意味を以下に示します。

読み取り (Read)
: これは、このユーザーがこのソリューション・ビューの「詳細」を読み取ることのみ可能であることを意味します。その内容は、このビュー内の AL、このビュー内のプロパティー、これらの AL の状況、などです。このユーザーは、このソリューション・ビューの詳細を修正、開始、停止、または変更することはできません。

実行 (Execute)
: これは、本質的には読み取り (Read) ユーザーで、AssemblyLine を開始および停止することができるという 1 つの権限を余分に保持します。

管理 (Admin)
: このユーザーは、ソリューション・ビュー自体を変更することなく、ソリューション・ビューを管理できます。このユーザーは、「実行 (Execute)」権限を持つユーザーができることをすべて実行でき、さらに、このソリューション・ビューについて、プロパティーを変更、ログを削除、ルールを構成することなどが可能です。

構成の管理 (Config_Admin)
: このユーザーは、ソリューション・ビューに対して事実上すべてのことを実行できます。これには、ビュー自体の変更、このビューに対する他のユーザーの権限の変更などが含まれます。これは、特定のソリューション・ビューについてユーザーに与えることができる最高の権限です。

上記の役割は、グループにも割り当てることができます。このため、ユーザー「test」および「tdi」が「DBAdmin」グループに含まれ、「DBAdmin」グループにソリューション・ビュー「SynchDatabase」に対する「ConfigAdmin」権限が与えられている場合は、「test」および「tdi」にも自動的に「SynchDatabase」ソリューション・ビューに対する「ConfigAdmin」権限が与えられます。

注:

1. 「test」ユーザーに同じソリューション・ビューに対する「読み取り (Read)」権限が明示的に与えられている場合は、「DBAdmin」グループに属していることから得られる権限より「読み取り (Read)」権限が優先されます。これは、グループからの役割の割り当てより「固有の」役割の割り当ての方が優先されるようにするためです。これにより、特定のグループに所属することで継承されるアクセスを考慮することなく、個々のユーザーのアクセスを制限したり、高い権限を与えたりすることができます。
2. 「test」ユーザーが 2 つのグループに所属し、同じソリューション・ビューに対し、この 2 つのグループのうちの Group1 は「読み取り (Read)」アクセスを、Group2 は「管理 (Admin)」アクセスを保持している場合、test ユーザーには 2 つの権限のうち高い方の権限が与えられます。「test」に同じソリューション・ビューに対する固有の役割が割り当てられていなければ (その場合は「test」に割り当てられた固有の役割が優先されます (上記 1 を参照))、この場合においては「管理 (Admin)」権限を得ることになります。

AMC およびパスワード

以下の情報を参照することで、パスワードの保管について知ることができます。

`amc.properties` ファイルに保管される LDAP バインド・パスワード、鍵ストア・パスワードなどのパスワード・フィールドは、`amc.properties` ファイルに書き込まれる前にすべて暗号化されています。また、AMC は、パスワード・フィールドまたは保護フィールドをコンソールに表示することはありません。このようなフィールドはすべてマスクされます。

AMC および暗号化された構成

以下の情報を参照することで、パスワードで保護された構成の使用について学習できます。

AMC により、ユーザーはパスワードによって保護された構成をロードしたり、接続したりすることができます。AMC の「ロード/再ロード」ウィンドウに「パスワード」テキスト・ボックスがあります。ユーザーは、**「開始」**をクリックする前にここに開始する構成のパスワードを入力する必要があります。同様に、「AssemblyLine の開始」アクションのための「Action Manager」画面にも、パスワード・フィールドがあり、ユーザーはここに構成のパスワードを入力することができます。Action Manager は、構成の開始時にこのパスワードを渡します。

注: AMC は、開始されようとしているリモート構成がパスワードで保護された構成であることを検出できません。このため、パスワードが指定されていない場合や正しく指定されていない場合は、「構成を開始できませんでした」というエラー・メッセージが表示されます。IBM Security Directory Integrator サーバーのログで、正確なメッセージを参照できます。

管理とモニター・コンソールのユーザー・インターフェース

以下の情報を参照して、AMC ユーザー・インターフェースについて詳しく知ることができます。

コンソールへのログインとログアウト

ここに示されている指示に従って、コンソールへのログインとログアウトを行うことができます。

Web ブラウザーをオープンし、以下のアドレスを入力します。

`http://hostname:port/ibm/console`

ここで「ポート」は、Web サーバーが実行中のポートを指します。バンドルされた Web コンテナーでデプロイするときに、デフォルトでは、ポートは HTTP に対しては 13100、HTTPS 通信に対しては 13101 が設定されています。

ログイン・ページは、*TDI_install_dir*/bin/amc フォルダーの launchAMC.html ファイルを使用して起動することもできます。

「IBM Security Directory Integrator 管理およびモニター・コンソール・ログイン・ページ (Administration and Monitoring Console login page)」ウィンドウが表示されます。

コンソール管理者としてコンソールにログオン

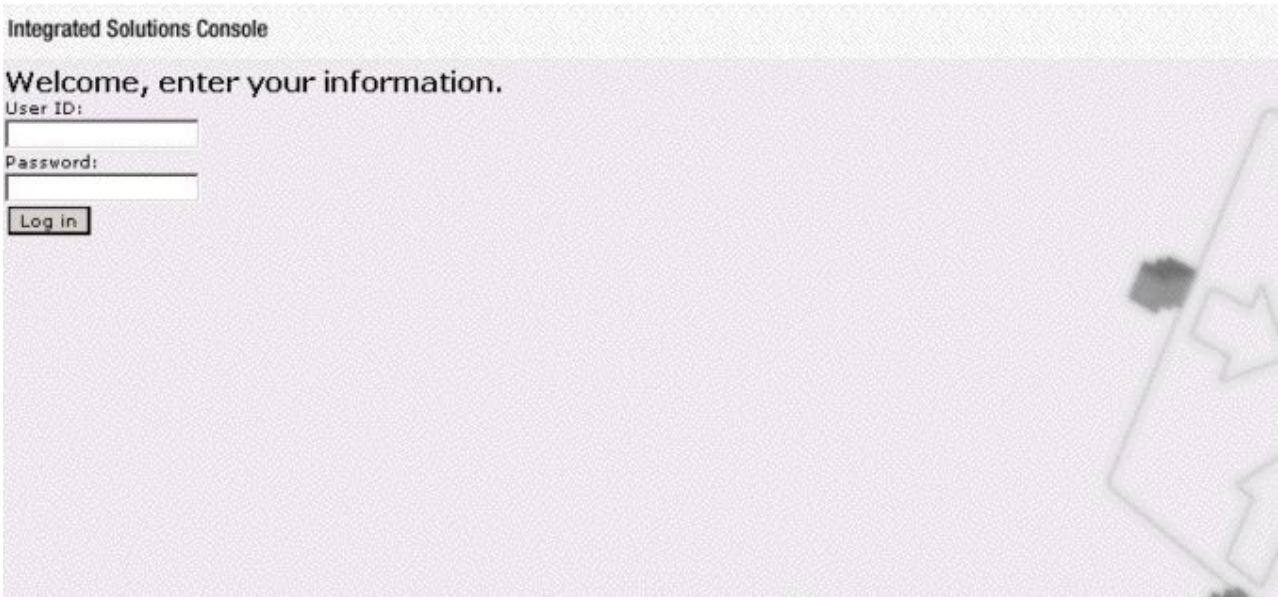
コンソール管理者は、以下を実行できるユーザーです。

- AMC に必要なプロパティーの構成
- AMC のログインに使用する認証メカニズムの設定
- 新規ユーザーの追加およびユーザーの役割の構成

初めてログインするときは、IBM Security Directory Integrator、AMC と組み込みの Web プラットフォームをインストールしたときに使用したシステムのユーザー名とパスワードを使用してください。AMC を IBM WebSphere Application Server/IBM Dashboard Application Services Hub でデプロイした場合は、*iscadmins* および *administrator* 役割が割り当てられたユーザーでログインする必要があります。

注: 組み込み Web プラットフォームは、UNIX および Linux box で PAM 認証メカニズムを使用して、ログイン時に提供されたシステムのユーザー名とパスワードを検証します。このため、AIX マシンでは、 /etc/security/login.cfg ファイルの auth_type パラメーターを PAM_AUTH に設定することが必要になります。

Integrated Solutions Console にログインするには、ログイン・ウィンドウのボックスにユーザー名とパスワードを入力し、「**ログイン**」ボタンをクリックします。



このコンソールの右上端、「**ヘルプ**」の隣には「**ログアウト**」ボタンがあります。「**ログアウト**」をクリックすると、「ログイン」ページに戻ります。

AMC コンソールのレイアウト

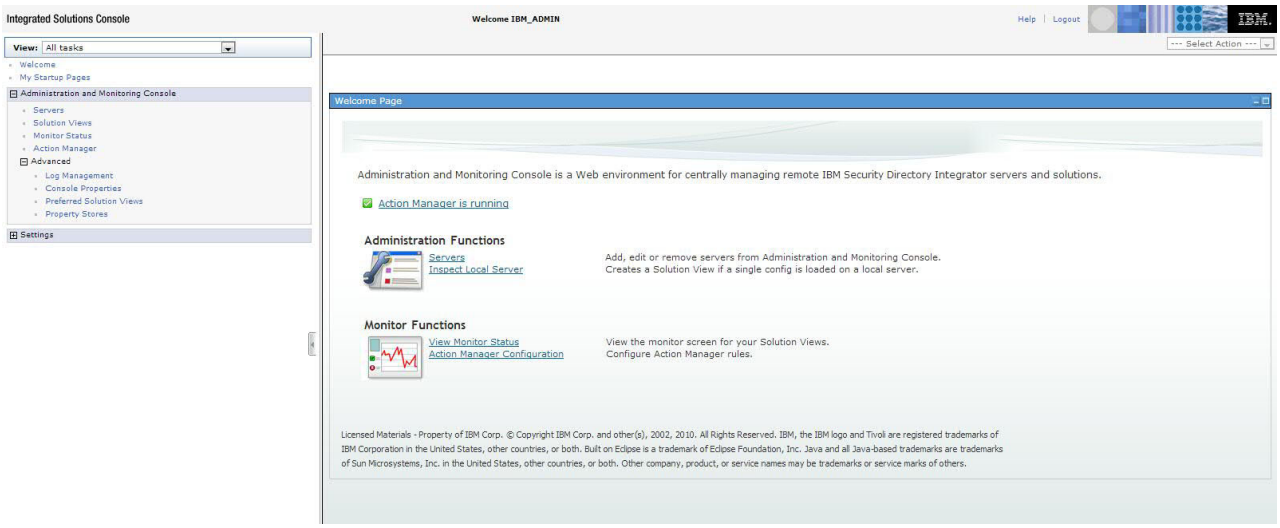
IBM Security Directory Integrator 管理およびモニター・コンソールのリストされたコンポーネントを参照することができます。

ナビゲーション・エリア

ナビゲーション・エリアでは、コンソールで使用可能なタスクにナビゲートできるツリー表示が提供されます。ナビゲーション・エリアでは、フォルダーをオープンおよびクローズしたり、タスク (非フォルダー) を選択してコンソールのフレームワークのワークエリアで起動したりすることができます。

ワークエリア

ワークエリアには、現在作業中のタスクを完了するために必要な情報および入力フィールドが表示されます。



コンソールのログオフ

コンソールからログオフするには、ナビゲーション・エリアの「**ログアウト**」をクリックします。

AMC テーブルの使用

AMC テーブル内の情報を使用し、これらのテーブル項目についてアクションを検索、編成、および実行することができます。

IBM Security Directory Integrator 管理およびモニター・コンソールでは、属性や項目のリストなどの特定の情報がテーブルで表示されます。

IBM Security Directory Integrator 管理およびモニター・コンソールのテーブルには、テーブル内の情報を編成および検索するためのアイコンが表示されます。現在のタスクに応じて、一部のアイコンが表示されるテーブルと表示されないテーブルがあります。表示される可能性のあるアイコンの全リストを以下に示します。

- 「**行フィルターの表示**」アイコンをクリックして、テーブル内の各列の行フィルターを表示します。フィルター処理の詳細については、298 ページの『フィルター処理』 を参照してください。
- 「**行フィルターの非表示**」アイコンをクリックして、テーブル内の各列の行フィルターを非表示にします。詳しくは、298 ページの『フィルター処理』を参照してください。
- 「**すべてのフィルターのクリア**」アイコンをクリックして、テーブルに設定されたすべてのフィルターをクリアします。詳しくは、298 ページの『フィルター処理』を参照してください。
- 「**ソートの編集**」アイコンをクリックして、テーブル内の情報をソートします。詳しくは、296 ページの『ソート』を参照してください。
- 「**すべてのソートのクリア**」アイコンをクリックして、テーブルに設定されたすべてのソートをクリアします。詳しくは、296 ページの『ソート』を参照してください。

- 「**テーブルの縮小表示**」アイコンをクリックして、テーブル・データを非表示にします。
- 「**テーブルの展開**」アイコンをクリックして、テーブル・データを表示します。
- 「**すべて選択**」アイコンをクリックして、テーブル内のすべての項目を選択します。
- 「**選択をすべて解除**」アイコンをクリックして、テーブル内のすべての選択済み項目を選択解除します。
- 「**エクスポート**」アイコンをクリックして、テーブル・データをエクスポートします。

「アクションの選択」ドロップダウン・メニュー

「**アクションの選択**」ドロップダウン・メニューを使用して、選択済みテーブルに対して使用可能なすべてのアクションの全リストを表示し、テーブル・コンテンツに対する操作を実行することができます。

このタスクについて

例えば、アイコンを使用してソートやフィルターを表示したり、非表示にしたりする代わりに「**アクションの選択**」ドロップダウン・メニューを使用できます。また、「**アクションの選択**」ドロップダウン・メニューを使用して、テーブル・コンテンツについて次のような操作を実行することもできます。例えば、「**属性の管理**」ウィンドウでは、「**表示**」、「**追加**」、「**編集**」、「**コピー**」、および「**削除**」などのアクションは、ツールバー上のボタンとしてのみではなく、「**アクションの選択**」ドロップダウン・メニュー内にも表示されます。テーブルでサポートされている場合は、「**アクションの選択**」ドロップダウン・メニューを使用して「**検出ツールバーを表示**」を表示したり非表示にしたりすることもできます。テーブル項目の検索に関する詳細については、『検索』を参照してください。

「アクションの選択」メニューを使用してアクションを実行するには、以下のようにします。

1. 必要な場合は、テーブルから項目を選択します。
2. 「**アクションの選択**」ドロップダウン・メニューをクリックします。
3. 実行するアクション (例えば「**サーバーのシャットダウン**」) を選択します。
4. 「**実行**」をクリックします。

ページング

さまざまなテーブル・ページを表示する場合は、テーブルの下部にあるナビゲーション・コントロールを使用できます。

ナビゲーション・フィールドに具体的なページ番号を入力して「**実行**」をクリックすると、特定のページを表示できます。また、「**次へ**」および「**前へ**」矢印を使用してページ間を移動することもできます。

ソート

テーブル内の項目のソート方法を変更できます。

このタスクについて

1. 以下のいずれかを実行します。
   - テーブルの「**ソートの編集**」アイコンをクリックします。
   - 「**アクションの選択**」ドロップダウン・メニューをクリックし、「**ソートの編集** 」を選択して、「**Go**」をクリックします。

   表内のすべての列に対して「ソート」ドロップダウン・メニューが表示されます。
2. 最初の「ソート」ドロップダウン・メニューから、ソートする列を選択します。ソートを実行する他のソート可能な列についても同様にします。
3. ドロップダウン・メニューから「**上/下 (Ascending/Descending)**」を選択し、昇順または降順のいずれでソートするかを選択します。デフォルトのソート順序は昇順です。列ヘッダーを使用してソートすることもできます。各列には、小さな矢印が表示されています。上向きの矢印は、列が昇順でソートされることを意味します。下向きの矢印は、列が降順でソートされることを意味します。ソート順を変更するには、単に列ヘッダーをクリックします。
4. ソートする準備ができたら、「**ソート**」をクリックします。

ソートをすべてクリアするには、**すべてのソートのクリア**のアイコンをクリックします。

検索

テーブル内で特定の項目 (複数可) を検索することができます。

このタスクについて

注: 現在のタスクに応じて、「検出ツールバーを表示」オプションが使用可能なテーブルと、使用できないテーブルがあります。

1. 「**アクションの選択**」ドロップダウン・メニューで「**検出ツールバーを表示**」を選択して、「**Go**」をクリックします。
2. 「**検索対象**」フィールドに検索条件を入力します。
3. 必要な場合、検索を実行する条件を「**条件**」ドロップダウン・メニューから選択します。このメニューのオプションは以下のとおりです。
   - **包含**
   - **先頭文字**
   - **終了文字**
   - **完全一致突き合わせ**
4. 「**列**」ドロップダウン・メニューから検索の基礎とする列を選択します。
5. 「**方向**」ドロップダウン・メニューから、結果を降順または昇順のいずれで表示するかを選択します。降順で結果を表示する場合は、「**下**」を選択します。結果を昇順で表示するには「**上**」を選択します。
6. 「**検索対象**」フィールド内の大文字および小文字の条件に一致する検索結果が必要な場合は、「**大/小文字の区別**」を選択します。
7. 必要な検索条件を入力し終えたら、「**検索**」をクリックして属性を検索してください。

フィルター処理

テーブル内の項目をフィルター処理することができます。

このタスクについて

1. 以下のいずれかを実行します。
   - 「**行フィルターの表示**」アイコンをクリックします。「**アクションの選択**」ドロップダウン・メニューをクリックし、「**行フィルターの表示**」を選択して、「**Go**」をクリックします。
2. 各列の上に「フィルター」ボタンが表示されます。フィルター処理を実行する列の上の「**フィルター**」をクリックします。
3. 「**条件**」ドロップダウン・メニューから以下のいずれかの条件を選択します。
   - 包含
   - 先頭文字
   - 終了文字
4. フィールドにフィルター処理を実行するテキストを入力します。例えば、「**先頭文字**」を選択した場合、「**C**」などと入力します。
5. 大/小文字 (大文字テキストまたは小文字テキスト) を突き合わせる場合は、「**大/小文字の区別**」を選択します。
6. 属性をフィルター処理する準備できたら、「**OK**をクリックします。
7. フィルター処理を実行するすべての列について、上記ステップ 2 から 6 を繰り返します。

すべてのフィルターをクリアするには、「**すべてのフィルターのクリア**」アイコンをクリックします。

行フィルターを非表示にするには、**行フィルターの表示**アイコンをもう一度クリックします。

サーバー

このウィンドウでは登録済みサーバーを表示できます。さらに、コンソール管理者は、このウィンドウから IBM Security Directory Integrator サーバーを追加、編集、削除、およびシャットダウンできるだけでなく、「構成ファイル」ウィンドウを起動することもできます。

AMC が開始されると、自動的にローカル IBM Security Directory Integrator サーバーがポート 1099 に登録されて設定されます。したがって、「**サーバー**」ウィンドウでは、LOCAL SERVER には既に、(サーバーの状況に対応して) 状態が「実行中」または「停止」の項目が 1 つあります。

構成をロードまたは再ロードするには、「**サーバー**」を選択し、「サーバー」ウィンドウのツールバー上で「**構成ファイル**」をクリックします。「**構成ファイル**」ウィンドウが表示されます。

実行する操作は、テーブルの上部にあるツールバーから選択するか、「アクションの選択」ドロップダウン・メニューを使用して選択します。例えば次のようにします。

追加　ツールバーの「追加」をクリックします。

削除　削除するサーバーの名前の横にあるラジオ・ボタンを選択して、ツールバーの「**削除**」をクリックします。

変更　情報を変更するサーバーを選択し、ツールバーの「**変更**」をクリックします。

構成ファイル
構成ファイルをリストするサーバーを選択します。「ソリューション・ビュー」ウィンドウの「**構成ファイルの表示**」リンクをクリックすると、「構成ファイル」ウィンドウが起動します。各構成ファイルは「ロード」または「未ロード」というラベルがつけられます。ツールバーはロード、アンロード、再ロードなどの多様なオプションを提供します。

サーバーのシャットダウン
シャットダウンするサーバーを選択して、ツールバーの「サーバーのシャットダウン」をクリックします。

正常なシャットダウン
実行中のサーバーを正常に終了します (AssemblyLine の停止を待機する新しいスレッドを作成)。

注: 正常なシャットダウンは、バージョン 7.1 以前の IBM Security Directory Integrator サーバーではサポートされていません。

サーバーの追加

IBM Security Directory Integrator サーバーを管理およびモニター・コンソール (AMC) に追加できます。

このタスクについて

AMC に IBM Security Directory Integrator サーバーを追加すると、他の AMC ウィンドウにある機能を使用して、IBM Security Directory Integrator サーバーにソリューション・ビューを追加したり、IBM Security Directory Integrator サーバーに関連付けられたソリューション・ビュー用のビューを作成および定義したりすることができます。

新規 IBM Security Directory Integrator サーバーを追加するには、以下のようにします。

1. IBM Security Directory Integrator サーバーの名前を「**名前**」フィールドに入力します。
2. IBM Security Directory Integrator が実行されているコンピューターのホスト名または IP アドレスを「**ホスト名**」フィールドに入力します。
3. IBM Security Directory Integrator サーバーを実行するように構成されたポートのポート番号を入力します。
4. 使用する認証モードを選択します。「カスタムまたは LDAP 認証」方式を選択した場合は、認証に使用するユーザー名とパスワードを入力します。
5. 「**OK**」をクリックします。

サーバーの変更

既存の IBM Security Directory Integrator サーバーの情報を編集できます。

このタスクについて

既存のサーバーを編集するには、以下のようにします。

1. 表示されたサーバー ID を確認します。サーバー ID を変更する場合、「**サーバー ID の変更**」をクリックします。
2. サーバーの名前を入力します。
3. IBM Security Directory Integrator サーバーが実行されているコンピューターのホスト名または IP アドレスを「**ホスト名**」フィールドに入力します。
4. IBM Security Directory Integrator サーバーを実行するように構成されたポートのポート番号を入力します。
5. 使用する認証モードを選択します。「カスタムまたは LDAP 認証」方式を選択した場合は、認証に使用するユーザー名とパスワードを入力します。
6. 変更せずにウィンドウを終了する場合は、「キャンセル」をクリックします。変更を保存する場合は、「**OK**」をクリックします。
7. 「**テスト接続**」をクリックして、現行の設定に基づいてサーバーへの接続が正常かどうかを確認します。

コンソール・プロパティー

AMC の「コンソール・プロパティー」ウィンドウを使用して、AMC のデータベース構成、 SSL の設定、Action Manager ログの循環頻度などの構成情報を管理します。

一般

AMC の「一般」ウィンドウを使用して、Action Manager のログの循環頻度を日数で設定します。

SSL

「SSL」ウィンドウを使用して、AMC 用の SSL 設定を構成します。SSL 設定は、AMC のリモート IBM Security Directory Integrator サーバーへの SSL 接続に適用されます。公開される SSL プロパティーは、AMC の鍵ストアおよびトラストストア・プロパティーのみです。リモート・サーバーで SSL がオンに設定されている場合、管理者は接続を機能させるために、必要な証明書がストア内にインポートされていることを確認する必要があります。管理者は、接続先の各リモート・サーバーの証明書を自分のストアにインポートする必要があります。

JDBC プロパティー

JDBC プロパティーは、Derby データベースへの接続設定を定義したり、管理およびモニター・コンソールと互換性のある Oracle や MS-SQL Server などの他のデータベースへの接続設定を定義するために使用します。AMC データベースは、AMC 構成情報、接続の詳細、Action Manager のルール、および結果をストアします。

IBM Security Directory Integrator AMC は、Derby に加えて代替データベースをサポートします。AMC は Derby データベースをバンドルします。AMC は Java データベース・コネクティビティー (JDBC) プロトコルを使用して、そのデータベースと通信します。 JDBC は汎用プロトコルで、他のデータベースに容易に拡張できま

す。代替データベースに対する AMC サポートにより、AMC をインストールして、既存のデータベースとの通信を可能にします。データベースは Action Manager のログ、結果などを保管します。Integrated Solution Console の「**拡張**」->「**コンソールのプロパティー**」セクションでは、Derby または別のデータベースへの「**JDBC プロパティー**」をグループ化します。Derby の場合、データベースを組み込みだけではなく、ネットワーク・モードでも実行されるように構成できます。デフォルトのデータベースは Derby で、デフォルト・モードはネットワーク・モードです。

このウィンドウでは、以下の操作を実行できます。

- 「**データベース・タイプ**」フィールドからデータベースを選択してください。Derby、MS SQL Server、Oracle、および DB2 のオプションがあります。
- JDBC URL を「**JDBC URL**」フィールドに入力します。
- 「**ユーザー名**」フィールドにデータベースのユーザー名を入力します。
- データベースのパスワードを「**パスワード**」フィールドに入力します。
- JDBC ドライバー名を「**JDBC ドライバー**」フィールドに入力します。

JDBC URL パラメーターおよび JDBC ドライバー・パラメーターについては、以下の表を参考にしてください。

表 31. ドライバーのパラメーター

データベース	JDBC URL	JDBC ドライバー	ドライバー .jar ファイル
Derby	jdbc:derby://host:port/database [;create=true | create=false]	org.apache.derby.jdbc.ClientDriver	derby.jar
MS SQL サーバー (2005)	jdbc:sqlserver://host:port; databasename=database	com.microsoft.sqlserver.jdbc.SQLServerDriver	sqljdbc.jar
Oracle	jdbc:oracle:thin:@host:port:database	oracle.jdbc.driver.OracleDriver	ojdbc14.jar
DB2	jdbc:db2://host:port/database	com.ibm.db2.jcc.DB2Driver	db2jcc.jar

注:

1. 選択されたデータベースに応じて、対応するドライバー .jar ファイルを *TDI_install_dir*/lwi /libs にコピーする必要があります。
2. 最初に使用する新規データベースを指定するためには、Action Manager の構成も必要です。同じ .jar ファイルを *TDI_install_dir*/bin/amc/ActionManager/jars に追加し、 am_config.properties ファイルの調整も行う必要があります。
3. Derby を使用しないが、代替データベースの 1 つを使用する場合、JDBC URL で指定されたデータベースは AMC を開始する前にあらかじめ存在している必要があることに注意してください (存在しない場合、AMC はデータベースを作成および取り込みできません)。Derby が使用される場合は、 JDBC URL の「create=true」オプションをサポートしているため、AMC の開始時にデータベースが自動的に作成されるため (存在しない場合)、このことは必要ありません。

ソリューション・ビュー

「ソリューション・ビュー」ウィンドウは、ソリューション・ビューを表示、追加、変更、または削除するために使用します。

- ソリューション・ビューを追加するには、ツールバーの「**追加**」ボタンをクリックします。

- 既存のソリューション・ビューを変更するには、「ソリューション・ビュー」を選択し、「**変更**」をクリックします。「**変更ウィザード**」の手順に従ってください。「**ソリューション・ビューの変更**」の下で、次の手順に進むときは「**次へ**」をクリックし、手順を完了したときは「**完了**」をクリックしてください。
- ソリューション・ビューのアクセス・コントロール・リスト (ACL) を構成するには、ACL を構成するソリューション・ビューを選択し、ツールバーの「**ACL の構成...**」を選択します。
- 既存のソリューション・ビューを削除するには、削除するソリューション・ビューを選択して、ツールバーの「**削除**」ボタンをクリックします。
- 別のパネルを起動してそのソリューション・ビューのローカル AM 変数を追加 / 編集 / 変更するためには、「**ローカル変数...**」をクリックします。

注: 「自動更新」オプションを使用して作成したソリューション・ビューを再ロードする必要があります。

ソリューション・ビューを変更すると、AMC はソリューション・ビューが「**自動更新**」を使用して作成されたかどうかを確認します。変更するために選択されたソリューション・ビューが自動更新を使用して作成された場合、次のようなメッセージが表示されます。

```
選択したソリューション・ビューは、自動更新にマークが付けられています。
ソリューション・ビューを変更するには、必ず自動更新を使用不可にしてください。
```

ソリューション・ビューはソリューション・ビュー・テーブルに表示されます。特定のソリューション・ビューが自動更新を使用して作成された場合、上向きの矢印をクリックした時に、ソリューション・ビュー名の右側に >> 簡略メニューが表示されます。「**ソリューション・ビューの更新**」または「**自動更新を使用不可に設定**」を選択できます。自動更新としてマークされたソリューション・ビューでは、構成ファイルを再ロードし、「**ソリューション・ビューのリフレッシュ**」をクリックしてソリューション・ビューをリフレッシュする必要があります。「**単純**」オプションを使用して作成され、自動更新が選択されたソリューション・ビューをリフレッシュしないと、AMC データベースでソリューション・ビューに不整合が発生する場合があります。ソリューション・ビューを更新しなかったために不整合が発生すると、Action Manager が誤った動作をする可能性があります。

ACL の構成

ユーザーのアクセス・コントロール・リスト (ACL) を設定し、そのユーザーに特定のソリューション・ビューを関連付けることができます。

このタスクについて

- ユーザー (1 人または複数) を構成するには、構成するユーザー (1 人または複数) を選択して、ツールバーの「**ユーザーの構成**」をクリックします。
  1. 「**ユーザー ID**」ドロップダウン・メニューから、役割を割り当てるユーザーを選択します。
  2. 選択したユーザーに割り当てる役割 (1 つ以上) の横にあるラジオ・ボタンを選択します。
     – 「**読み取り**」- ユーザーによるソリューション・ビューの詳細 (AL、トゥームストーン、ログ、構成のプロパティーなど) の読み取りを許可します。

– 「**実行**」- ユーザーに読み取りおよび AssemblyLine の開始/停止を許可します。
– 「**管理者**」- ユーザーにリーダー役割および実行役割を付与します。この役割のユーザーは、ログとトゥームストーンの削除操作も実行できます。
– 「**構成の管理**」- ユーザーが構成を開始および停止し、ソリューション・ビューを変更し、他のユーザーの ACL を割り当ておよび変更することを許可します。

3. 「**適用**」をクリックします。

- 既存のユーザーを除去するには、表からユーザーを選択して「**除去**」をクリックします。

変更を終えたら、「**適用**」をクリックします。

ローカル変数

以下の情報を参照することで、ローカル変数の処理について学習できます。

AMC の左側のナビゲーション・ペインからソリューション・ビューを選択します。「**ソリューション・ビュー**」ウィンドウが表示されます。ツールバーから「**ローカル変数**」を選択します。「**ローカル変数**」ウィンドウで、ソリューション・ビューのローカル変数を選択し、**追加**、**変更**、または**削除**できます。

Action Manager のトリガーとアクションは、ルールやアクションを使用して設定または増分できるローカル変数をサポートしていることが必要です。ローカル変数は、他のルールのトリガー条件として使用できます。例えば、ローカル変数は値 1 に設定することができ、イベントが発生するたびに 1 ずつ増分させることができます。またローカル変数 (この例では、数値 1 がイベントが発生するたびの増分値として設定されます) は、「AssemblyLine の停止」ルールを起動できます。変数が値 10 に達すると、新規ルールをトリガーするように構成できます。新しいルールは、異なるサーバーで新しい AssemblyLine を開始できます。これらの「ローカル」の AM 特有の変数を「ソリューション・ビュー」に設定します。すなわち、1 つのソリューション・ビューに属するルールで作成された 1 個の変数は、そのソリューション・ビューのルールでのみ使用でき、別のソリューション・ビューのルールにアクセスすることはできません。

ソリューション・ビューの追加

構成ファイルを直接編集する権限をユーザーに付与せずに、ユーザーが構成ファイル内の情報にアクセスできるようにすることが可能です。

このタスクについて

管理者はソリューション・ビューを使用して、構成ファイルを特定の情報のみにフィルター処理し、構成ファイル内の特定の情報のみが表示されるようにすることができます。ソリューション・ビューは各構成に対して複数作成でき、構成ファイル内の異なる情報を各ビューで表示することができます。

ソリューション・ビューを追加するには、「**ソリューション・ビュー**」を選択し、「**ソリューション・ビュー**」ウィンドウのツールバーの「**追加**」を選択します。

1. ビュー詳細の入力:

a. ソリューション・ビューの名前を「**ソリューション・ビュー名**」フィールドに入力します。

b. ソリューション・ビューの説明を「**説明**」フィールドに入力します。

2. ソリューション・ビューを作成するために使用する**サーバー**および**構成** (構成ファイル) の選択:

   - 「**サーバー**」から、ソリューション・ビューを作成するために使用する構成ファイルを含む IBM Security Directory Integrator サーバーを選択します。このメニューは、管理およびモニター・コンソールに IBM Security Directory Integrator サーバーが 1 つも追加されていない場合は空になります。
   - 「**構成**」リストから、ソリューション・ビューを作成するために使用する構成ファイルを選択します。メニューには、現在ロードされているすべての構成が含まれます。

     注: 「**構成ファイルの表示**」ボタンをクリックして、「構成ファイル」ウィンドウに進みます。このウィンドウで、構成ファイルのロードまたはアンロード操作を実行できます。

3. 「**ソリューション・ビュー**」ツールバーの「**追加**」をクリックします。

   a. 作成するソリューション・ビューの名前を「**ソリューション・ビュー名**」フィールドに入力します。

   b. 作成するソリューション・ビューの「**説明**」をオプションで入力します。

   c. ソリューション・ビューの作成に使用する構成ファイルおよび AssemblyLine を含む「**サーバー**」を選択します。

   d. 「**構成**」リストから、使用する構成ファイルを選択します。

   e. 「**自動更新**」を使用可能または使用不可にします。

      AssemblyLine または構成のプロパティーが変更されると、「**自動更新**」により自動的にソリューション・ビューが変更されます。

      注: 「自動更新」が選択されると、自動更新をオンにして作成したソリューション・ビューを編集したり、自動更新がオンになっている間に作成されたソリューション・ビューのルールおよびトリガーを作成することはできません。ソリューション・ビューを編集したり、ルールやトリガーを追加する場合は、「自動更新」を無効にしてください。自動更新としてマークされたソリューション・ビューのルールやトリガーを作成するためには、自動更新機能を無効にすることが必要になります。「ソリューション・ビュー」ウィンドウの「リフレッシュ」ボタンを使用することにより、ソリューション・ビューの構成の変更を確認してください。このボタンは、自動更新が「true」に設定された構成でのみ表示されます。「ソリューション・ビューの作成」ウィザードを使用して手動作成した構成は、自動更新フラグが false に設定されています。

      注: 「自動更新」オプションを使用して作成したソリューション・ビューを再ロードする必要があります。これには「**ソリューション・ビュー**」ウィンドウにある「**ソリューション・ビューのリフレッシュ**」を使用します。自動更新としてマークされたソリューション・ビューでは、構成ファイルを再ロードし、「**ソリューション・ビューのリフレッシュ**」をクリックしてソリューション・ビューをリフレッシュする必要があります。「**単純**」オプションを

使用して作成され、自動更新が選択されたソリューション・ビューをリフレッシュしないと、AMC データベースでソリューション・ビューに不整合が発生する場合があります。ソリューション・ビューを更新しなかったために不整合が発生すると、Action Manager が誤った動作をする可能性があります。

4. ソリューション・ビューの作成で、以下のオプションを使用してください。

単純
共通のデフォルト・オプションでソリューション・ビューを作成します。

自動更新
自動更新としてマークされたソリューション・ビューでは、構成ファイルを再ロードし、ソリューション・ビューをリフレッシュする必要があります。

パブリッシュ済みソリューションからソリューション・ビューを作成する。
IBM Security Directory Integrator 構成エディター (CE) で指定したとおりに、パブリッシュ済みソリューションからソリューション・ビューを作成します。このオプションでは、アクティブ構成インスタンスにパブリッシュ済みソリューションが関連付けられている必要があり、さらには IBM Security Directory Integrator 7.0 サーバーが必要です。

すべての AssemblyLine を公開してソリューション・ビューを作成する。
ソリューション・ビューを作成する際に、構成インスタンスからのすべての AssemblyLine が公開されますが、プロパティーとヘルス AL は定義されません。このオプションは、即時開始するときに使用してください (開発目的の場合に有効です)。IBM Security Directory Integrator 6.0 以降のサーバーで使用可能です。

すべての AssemblyLine とすべてのプロパティーを公開してソリューション・ビューを作成する。
ソリューション・ビューを作成する際に、構成インスタンスからのすべての AssemblyLine が公開され、すべてのプロパティーが定義されますが、ヘルス AL は定義されません。このオプションでは Java プロパティーは公開されません。IBM Security Directory Integrator 6.1 以降のサーバーで使用可能です。このオプションは、即時開始するときに使用してください (開発目的の場合に有効です)。

すべての AssemblyLine とすべてのユーザー・プロパティーを公開してソリューション・ビューを作成する。

ソリューション・ビューを作成する際に、構成インスタンスからのすべての AssemblyLine が公開され、すべてのプロパティーが定義されますが、ヘルス AL は定義されません。これは、即時開始タイプのオプションに似ています。このオプションは、IBM Security Directory Integrator 6.0 サーバーでは使用できません (IBM Security Directory Integrator プロパティーが IBM Security Directory Integrator 6.0 サーバーで使用できないため)。

注: 「プロパティー・ストア」パネルにユーザーが定義したプロパティーを表示するには、以下のいずれかのことを実行する必要があります。

- 構成ファイルを含むフォルダーに .properties ファイルを配置する

- CE でプロパティー・ストアを作成するときに、プロパティー・ファイルへの絶対パスを指定する (「**新規プロパティー・ストア**」 >「**コネクター**」タブ >「**構成**」タブ >「**コレクション・パス/URL**」パラメーター)

5. 「**OK**」をクリックして、ソリューション・ビューの作成を完了します。

構成ファイル (構成のロード/再ロードを許可)

以下の情報を参照することで、構成ファイルについて詳しく知ることができ、これらの構成ファイルの処理方法を学習できます。

「構成ファイル」ウィンドウを表示し、オプションにアクセスして、構成ファイルのロード、再ロード、アンロード、およびリフレッシュを実行するには、左ナビゲーション・エリアの「**ソリューション・ビュー**」を選択します。サーバーおよび構成ファイルを選択してから、「**構成ファイルの表示**」ボタンをクリックします。これにより、「構成ファイル」ウィンドウが起動します。このウィンドウには、ロード済みの構成およびリモート IBM Security Directory Integrator サーバーの構成フォルダー内の構成が表示されます。AMC が IBM Security Directory Integrator サーバーに接続されると、「構成ファイル」ウィンドウにはリモート構成フォルダーのすべてのファイルのリストが表示されます (ファイルが、有効な IBM Security Directory Integrator 構成ファイルであるかどうかに関係なく)。有効な IBM Security Directory Integrator 構成ファイルに対してのみロード操作を実行してください。このようにしないと、エラー・メッセージが AMC に表示されます。「ロード済み」または「アンロード」の状況が画面の「状況」カラムに緑 (ロード済み) または赤 (アンロード) のアイコンで表示されます。「構成ファイル」テーブルの「**選択**」カラムから 1 つまたは複数の構成を選択できます。構成を選択すると、テーブルの一番上のボタンを使用して、「**ロード**」、「**別名でロード...**」、「**アンロード**」、「**再ロード**」、または「**リフレッシュ**」を実行することができます。パスワードで保護された構成をロードする場合は、「構成」を選択し、「パスワード」フィールドにパスワードを入力します。

アクションが成功したか失敗したかを示すメッセージがアクション (ロード、実行名でロード、再ロード、アンロード、およびリフレッシュ) が実行された後で表示され、結果を示します。ロード、再ロード、およびアンロードの場合、選択した構成の新しい状況が「ステータス」カラムに表示されます。

注: これらの操作を実行するには、 superadmin 特権または構成管理特権が必要です。

- 構成をロードするには、ロードする構成を選択して、「**ロード**」をクリックします。
- 1 つの構成の複数のインスタンスをロードする場合は、ロードする構成を選択し、「**別名でロード...**」をクリックしてください。「**カスタム・ロード**」ウィンドウが開き、「**構成ファイル**」、「**構成ファイル実行名**」、「**構成パスワード**」、および「**プロパティー・ストア値**」を指定できます。
- 構成をアンロードするには、アンロードする構成を選択して、「**アンロード**」をクリックします。

注: サーバーをロードしても、選択した構成に関連付けられた AssemblyLine が自動的に開始されるわけではありません。「自動開始」として指定された AssemblyLine のみがロード時に開始されます。

- 構成を再ロードするには、再ロードするロード済みの構成を選択し、「**再ロード**」をクリックします。再ロードできるのは、ロード済みのステータスをもつ構成のみです。
- 構成をリフレッシュするには、「**リフレッシュ**」をクリックします。テーブルのすべての構成の情報が再表示されます。
- 変更を終えたら、「**クローズ**」をクリックします。

注: ユーザーはデータ保全性を維持する必要があります。

- 例えば、ソリューション・ビューとルールが config1.xml という名前の構成に対して作成され、実行名が ABC の場合は、例えば、config2.xml という名前の構成で、ソリューション名または実行名のいずれかが ABC という名前のような別の構成をロードしないでください。
- 特定の実行名とプロパティー・ファイルのセットを使用して作成するソリューション・ビューを再使用する場合は、同じ実行名とプロパティー・ファイルを使用してこの構成をアンロードする必要があります。

カスタム・ロード:

以下に示す手順を実行することにより、AMC を使用して複数の構成インスタンスをロードできます。

このタスクについて

IBM Security Directory Integrator サーバーは、別の実行名をもつ同じ構成の複数のインスタンスのロードをサポートします。「**別名でロード...**」を使用して構成インスタンスをロードする場合、これらの構成を使用してソリューション・ビューおよびルールを作成できます。

1. 「**ようこそ**」ページから、「**サーバー -> 構成ファイル**」を選択します。
2. 「**別名でロード...**」をクリックします。

   「**カスタム・ロード**」ウィンドウが表示されます。
   a. 複数のインスタンスを作成する「**構成ファイル**」を選択し、「**実行**」をクリックします。
   b. 「**構成ファイル実行名**」を入力します。
   c. 「**構成パスワード**」を入力します。
   d. 各プロパティー・ストア名の「**プロパティー・ストア値**」を入力します。
3. 「**OK**」をクリックし、入力した値を使用して、指定した実行名で構成のインスタンスを作成します。構成のインスタンスが作成されると、「**構成のロード/再ロード**」ウィンドウに戻ります。
4. 「**カスタム・ロード**」で指定した値を使用して構成を作成しない場合は、「**キャンセル**」をクリックします。

状況のモニターおよび Action Manager

以下の情報を参照して、状況のモニターで実行できるアクションをすべて知ることができます。

展開していない場合は、管理およびモニター・コンソールのメイン・ナビゲーション・エリアにある「**状況のモニター**」カテゴリーを展開します。

以下のいずれかを実行します。

- 各ソリューション・ビューに関する情報を確認するには、「状況のモニター」テーブルを参照します。ソリューション・ビューについて Action Manager 状況、ヘルス・チェック結果、ヘルス・チェック状況などの情報が表示されます。「**ソリューション・ビューの詳細**」、「**サーバー情報**」、および「**優先ビューの表示**」も表示できます。
- Action Manager ルールを追加、編集または削除するには、 **313 ページの『Action Manager』**をクリックします。

状況のモニター

各優先ソリューション・ビューに関する概要情報を表示することができます。

このウィンドウには、「**拡張**」 -> 「**優先ソリューション・ビュー**」からアクセスされた「優先ビュー」ウィンドウで選択したビューが表示されます。概要情報には次のようなものがあります。

Action Manager 状況
選択したソリューション・ビューの Action Manager ルールの状況が表示されます。青色の感嘆符は、最近起動された Action Manager ルールがないことを示します。中に感嘆符を含む黄色の三角形は、Action Manager ルールが最近起動されたことを示します。

ヘルス・チェック結果
ソリューション・ビューのヘルス AssemblyLine にある healthAL.result 最終作業項目属性から取得したヘルス・チェック結果が表示されます。この値はテキストとして表示されます。

ヘルス・チェック状況
ソリューション・ビューのヘルス AssemblyLine 内の healthAL.status 属性から取得されたヘルス・チェックの状況が表示されます。

さらに、管理およびモニター・コンソールの `resources/amc_images/healthAL` ディレクトリーで戻される状況値と同じ名前の .gif ファイルを指定した場合、この列には .gif イメージも表示されます。例えば、healthAL.result が「Error」として戻され、上述のディレクトリーに「Error.gif」が作成されている場合、テーブル列に Error.gif イメージが表示されます。

このウィンドウでは、以下の操作を実行できます。

- ソリューション・ビューの詳細の表示 - 特定のソリューション・ビューの詳細を表示するには、目的のソリューション・ビューを選択して、「**ソリューション・ビューの詳細**」をクリックします。

- IBM Security Directory Integrator サーバー情報の表示 - ソリューション・ビューが属するサーバーの詳細を表示するには、「 **サーバー情報**」をクリックします。
- 優先ソリューション・ビューの表示 - 「**優先ビューの表示**」をクリックして、優先ソリューション・ビューを表示します。このボタンは、優先ソリューション・ビューが定義されている場合にのみ表示されます。優先ソリューション・ビューは、「**ユーザー設定**」下の「優先ソリューション・ビュー」ウィンドウで定義できます。

ソリューション・ビューの詳細:

ソリューション・ビューの詳細を確認できます。

「ソリューション・ビューの詳細」パネルは、管理者が調べたり、アクションをとることができる「ソリューション・ビュー」に特有の詳細をさらに具体的に示します。

このウィンドウには、2 つのテーブルがあります。上部のテーブルには、選択したソリューション・ビューに関連する AssemblyLine および各ソリューション・ビューの状況が表示されます。下部のテーブルには、最近起動された Action Manager ルールに関するログ情報が表示されます。

変更を終えたら、「**クローズ**」をクリックします。

「ソリューション・ビューの詳細」テーブル:

「**ソリューション・ビューの詳細**」テーブルには、列がリストされます。

列

選択 アクションを実行する AssemblyLine の横にあるラジオ・ボタンを選択します。

AssemblyLine
AssemblyLine の名前が表示されます。

状況 AssemblyLine の状況が表示されます。例えば、「**実行中**」または「**停止**」。

開始時刻

AssemblyLine は稼働中
開始時刻は、稼働中の AL が開始した時刻です。開始時刻は、稼働中の AL に基づいています。

AssemblyLine は停止
AL の最終実行が開始した時刻。開始時刻は、AL の最新のトゥームストーン項目に基づいています。(IBM Security Directory Integrator サーバーでのみ使用可能)

最終停止時刻
AL の最終実行が終了した時刻。停止時刻は、AL の最新のトゥームストーン項目に基づいています。(IBM Security Directory Integrator サーバーでのみ使用可能)

統計 実行中の AssemblyLine について現在の統計が表示されます。

アクション

実行する操作は、テーブルの上部にあるツールバーから選択するか、「**アクションの選択**」ドロップダウン・メニューを使用して選択します。例えば次のようにします。

- トゥームストーンの表示 - 表示する AssemblyLine を選択して、ツールバーの「**トゥームストーンの表示**」ボタンをクリックします。
- ログの表示 - 表示する AssemblyLine を選択して、以下のいずれかを実行します。
  - ツールバーの「**ログの表示**」ボタンをクリックします。
  - 「**アクションの選択**」ドロップダウン・メニューから「**ログの表示**」を選択して、「**Go**」をクリックします。
- プロパティーの管理 - 管理するプロパティーのある AssemblyLine の横のラジオ・ボタンを選択して、ツールバーの「**プロパティーの管理**」ボタンをクリックします。
- AssemblyLine の開始 -
  1. 開始する AssemblyLine を選択します。
  2. 「**ポップアップの表示**」ボタンをクリックします。
  3. 「**AssemblyLine の開始**」をクリックします。
- AssemblyLine の停止 - 停止する AssemblyLine を選択して、以下のいずれかを実行します。
  1. 停止する AssemblyLine を選択します。
  2. 「**ポップアップの表示**」ボタンをクリックします。
  3. 「**AssemblyLine の停止**」をクリックします。

  注: IBM Security Directory Integrator v7.1 から、新しいオプションが使用可能です - 「AssemblyLine を正常に停止」。これが選択されると、AssemblyLine は新しいスレッドで停止されます。AssemblyLine を正常に停止は、v7.1 より前の IBM Security Directory Integrator サーバーでは使用できません。
- ソリューション・ビューの詳細 - 「**ソリューション・ビューの詳細**」ボタンをクリックします。例えば、AssemblyLine など、表示するコンポーネントを選択します。

AssemblyLine の開始

選択した AssemblyLine を実行します。

AssemblyLine の同期的な開始

AMC は AL が終了するのを待ち、実行中の AL の状況を定期的に表示します。終了後の、AssemblyLine の出力スキーマ属性を AL の同期的な実行に対して表示できます。

シミュレート・モードでの AssemblyLine の開始

AssemblyLine は、追加、更新、および削除モードのコネクターを除くすべてのコンポーネントを実行します。すなわち、コネクターの putEntry、modEntry、および deleteEntry 方式はシミュレート・モードでは起動されません。結果として、シミュレート・モードで実行される

AssemblyLine は、サード・パーティーのリポジトリーで追加、変更、または削除を全く実行しません。シミュレート・モードの詳細については、「*Directory Integrator の構成*」の対応するセクションを参照してください。

トゥームストーンの表示

リモートの IBM Security Directory Integrator サーバー上でトゥームストーンを有効にしている場合は、終了した AssemblyLine のトゥームストーン項目を管理およびモニター・コンソールで表示できます。 このウィンドウには、項目がいつトゥームストーン状態に変更されたかなど、トゥームストーン項目についての有用な情報が表示されます。

トゥームストーンの削除

「**状況のモニター**」ウィンドウで、AssemblyLine を選択します。AssemblyLine の右側の矢印を選択し、メニューから「**トゥームストーンの削除**」を選択します。これにより、「**トゥームストーンの削除**」ウィンドウが起動します。このウィンドウのコンポーネントの詳細セクションでは、作業を行っているソリューション・ビューおよび AssemblyLine が識別されます。「削除基準の選択」セクションで、削除するトゥームストーンを指定するオプションの 1 つを選択します。

- 選択した AssemblyLine のすべてのトゥームストーンを削除するには、「**すべてのトゥームストーン**」を選択します。
- 「**開始日**」と「**終了日**」を使用して、削除するトゥームストーンの日付範囲を指定します。AMC は選択した日付から現在の日付までの日数を計算します。AMC は次に、計算された日数の間に生成されたトゥームストーンを削除します。
- 「**戻す項目数**」を使用して、削除したい最新のトゥームストーン数を示す整数を指定します。「**削除**」をクリックすると、確認メッセージが表示されます。確認すると、AMC は削除コマンドを実行します。

ログの表示

指定された AssemblyLine のログは、「ログの表示」ウィンドウに表示されます。「**状況のモニター**」 -> 「**ソリューション・ビューの詳細**」 -> 「**ログの表示**」により、選択した AssemblyLine のログ・ファイルのリストを表示し、表示したいログの隣のラジオ・ボタンをクリックして、「**ログの表示**」をクリックします。

注: 管理およびモニター・コンソールで AssemblyLine ログを表示するには、 AssemblyLine が SystemLog ロガーを使用してログを記録する必要があります。

「Action Manager 結果」テーブル:

ここでは、「**Action Manager 結果**」テーブルの各列および Action Manager 結果に対する操作の実行方法について説明します。

Action Manager に設定したルールが起動された場合は、違反のソース、エラーの説明、違反が発生した時刻など、この違反に関する情報がログに記録されます。これらの詳細情報は、「**Action Manager 結果**」テーブルに表示されます。

列

「**Action Manager 結果**」テーブルには、以下の列があります。

選択 アクションを実行するメッセージの横にあるラジオ・ボタンを選択します。

ソース 起動された Action Manager ルールの名前が表示されます。

重大度 メッセージの重大度が表示されます。

メッセージ

Action Manager アクションに関連するメッセージが表示されます。

説明

メッセージに関する追加の情報が表示されます。

タイム・スタンプ

Action Manager ルールが起動され、メッセージが生成された時刻が表示されます。

アクション

削除する結果を選択して、「**削除**」をクリックします。

コンポーネントの表示:

「コンポーネントの表示」操作により、選択した AssemblyLine で構成された異なるコネクター、関数コンポーネントなどを表示できます。N

注: ブランチ・コンポーネント (IF、SWITCH、など) およびスクリプト・コンポーネントは表示されません。これは、重要な項目であるコネクター/関数コンポーネントに焦点を当てるための、意図的な設計です。

優先ソリューション・ビューの表示:

優先ソリューション・ビューは、「**状況のモニター**」ウィンドウに表示されるデフォルトのソリューション・ビューです。

AMC の「ソリューション・ビューの詳細」のリフレッシュ

以下の手順に従って、リフレッシュ間隔を変更できます。

「ソリューション・ビューの詳細」ウィンドウは、設定された時間間隔後に現在の AssemblyLine 状況を表示するためにリフレッシュされます。リフレッシュ頻度は、デフォルトでは 600 秒に設定されます。Integrated Solutions Console 管理者には、リフレッシュ間隔を変更する特権があります。

リフレッシュ間隔を変更するには、以下の手順を実行します。

1. AMC のログイン・ページを表示します。
2. ユーザー名とパスワードを入力してから「**ログイン**」をクリックします。Integrated Solutions Console のウェルカム・ページが表示されます。
3. 左側のナビゲーション・ツリーで、「**設定**」 –> 「**グローバル・リフレッシュの管理**」をクリックします。
4. 「グローバル・リフレッシュの管理」ウィンドウで、「**状況のモニター**」リンクをクリックします。
5. リフレッシュの構成設定を変更して、「**OK**」をクリックします。

Action Manager

ルール、トリガー、およびルール実行とトリガー条件の結果として実行されるアクションを追加、削除、または変更できます。

構成ルールの追加/編集:

このウィンドウでは、構成ルールを追加または編集することができます。

このウィンドウ上の設定を使用して、現在のソリューション・ビューの 279 ページの『Action Manager』を作成 (または既存のルールを変更) できます。

ルールは次の 2 つの部分から構成されます。

- ルールが呼び出される条件。これは「トリガー」と呼ばれます。

  トリガーの例としては、サーバー API の障害、AssemblyLine の障害、指定されたインターバルで実行する AssemblyLine の障害などが挙げられます。
- トリガーが検出されると実行される一連の代替アクション。

構成ルールの設定:

このウィンドウは、ルールの最初の部分 (トリガーの定義) に関するウィンドウです。名前、説明、およびトリガー・タイプを選択できます。

名前 ルールの名前を入力します。ルールを追加する場合、このフィールドは必須です。

説明 オプションのルールの説明を入力します。

トリガー・タイプ

トリガー・タイプは、ルールが呼び出される条件を定義します。 ドロップダウン・メニューから以下のトリガー・タイプを選択します。

トリガーなし

ルールがトリガー条件を持ちません。

AssemblyLine 終了時

指定された AssemblyLine の終了時にルールが起動されます。

構成のロード時

このルールは、Action Manager が、特定の構成に対して構成ロード・イベントを受信すると起動します。

構成のアンロード時

このルールは、Action Manager が特定の構成に対して構成のアンロード・イベントを受信すると起動します。

AssemblyLine 結果の照会時

指定された AssemblyLine の最後の「作業」項目に、所与の条件および値に一致する属性が含まれる場合にルールが起動されます。

サーバーでの API 障害時

Action Manager がサーバー API を使用してリモート・サーバーに接続できない場合にルールが起動されます。このルールは 1 回のみ起動されます。ルールは、サーバー API を使用してサーバーに再接続できることが検出されるとリセットされます。

イベント受信時
Action Manager が、「イベント・タイプ」、「イベント・ソース」および「イベント・データ」の各フィールドで指定された基準を満たすイベントを受信した場合に、ルールが起動されます。

プロパティー・トリガー時
指定されたプロパティーが、決められたプロパティー名、条件、および値の指定内容を満たす場合にルールが起動されます。

ローカル変数時
ルールは、指定された変数が指定された条件を満たすときに起動されます。Action Manager はこのプロパティーを定期的に検査します。

注: このルールは 1 回だけ起動されます。この変数が指定の条件に一致していないことが Action Manager により検出されると、作動可能状態にリセットされます。再検査により、トリガー条件が発生するたびにルールが繰り返し起動されないことが保証されます。

AssemblyLine 終了コードの検査
このルールは AssemblyLine が異常終了したときに起動します。Action Manager が AssemblyLine 終了コードで検索するエラー・オブジェクトを定義できます。

最終実行からの時間
指定された AssemblyLine が決められた期間実行されなかった場合にルールが起動されます。

タイマー・トリガー
ルールは指定された間隔内に継続的に起動されます。

トリガーの構成:

各トリガー・タイプでは、それぞれ異なる設定が選択されます。以下にリストされている一部のフィールドがウィンドウに表示されていない場合は、現在選択しているトリガー・タイプでそれらがサポートされていません。

ソース モニターするソースを入力します。

データ・
モニターするデータを入力します。

プロパティー名
ドロップダウン・メニューから、モニターするプロパティーの名前を選択します。

条件 プロパティーおよび値を比較するために使用する条件を選択します。使用できるオプションには以下のものがあります。

- 等しい
- 等しくない
- より大
- より小

値 モニターする値を入力します。

構成済みアクション:

このテーブルから、アクションを追加、削除、および変更できます。アクションをテーブル内で上または下に移動することもできます。

選択できる構成可能なアクション・テーブルのそれぞれのアクションに対して、特別のトリガー「**エラー発生時に実行**」を使用可能にできるカラムがあります。「**エラー発生時に実行**」はエラー条件が発生したときに選択したアクションを実行します。

- 管理するアクションを選択するためには、リストされる各アクションの前のラジオ・ボタンをオンにします。
- アクションを追加するには、「**追加**」をクリックします。
- アクションを削除するには、削除するアクションを選択して、「**削除**」をクリックします。
- アクションを変更するには、変更するアクションを選択して、「**変更**」をクリックします。
- テーブル内でアクションの位置を 1 つ上に移動するには、移動するアクションを選択して、「**上へ移動**」をクリックします。
- テーブル内でアクションの位置を 1 つ下に移動するには、移動するアクションを選択して、「**下へ移動**」をクリックします。

「**エラー発生時に実行**」は、以前のいずれかのアクションの実行中にエラーが発生した場合のみ、アクションを実行します。以前のいずれかのアクションの実行中に発生したエラーを処理するための修正手段を講ずる場合に、「エラー発生時に実行」のチェック・マークを付けたアクションを使用できます。アクション・エラー変数: AMC および Action Manager は、さまざまなアクションで、アクション・エラーを使用可能に設定できるようにします。どの時点でも、いずれかの構成済みアクションの実行中にエラーが発生した場合、ユーザーはそのエラーを、特別に予約された変数の形式で使用できるようになります。次いでそれらの予約済み変数を、構成済みの他のアクションで使用できます。以下のアクションが実行された場合、Action Manager ではストリング %Action_Error% は、以前のアクションの実行中に発生した実際のエラーにより置き換えられます。エラーが発生しない場合、変数 %Action_Error% は置き換えられず、そのままとなります。

- E メールの送信
- コマンドの実行
- イベント送信アクション
- ログへの書き込みアクション

追加/変更アクション:

ルールが起動されると、Action Manager はそのルールに関連付けられたアクションを実行します。ルールが起動されたときに Action Manager で実行されるアクションを指定または変更することができます。

ドロップダウン・メニューからアクション・タイプを選択して構成します。終了したら「**OK**」をクリックします。

AssemblyLine の開始

このアクションにより AssemblyLine が開始されます。このアクションを選択する場合は、開始する AssemblyLine の名前と、この AssemblyLine に関連付けられている構成 (場合によっては構成のパスワード) を指定する必要があります。

サーバー

構成済みサーバーのドロップダウン・リストです。LocalServer は、Action Manager が稼働しているサーバーです。

リモート構成フォルダーからの選択

このチェック・ボックスにチェック・マークを付けると、リモート・サーバーに対し、使用可能な構成ファイルの照会が実行されます。表示される構成ファイルは、`global.properties` ファイルの api.config.folder プロパティーにパスが指定されているフォルダーに含まれている構成ファイルです。

構成名 「AssemblyLine」フィールド内の AssemblyLine が属する構成を入力します。「**リモート構成フォルダーからの選択**」にチェック・マークを付けた場合は、リモート・サーバーの使用可能な構成ファイルのリストが表示されます。チェック・マークを外した場合は、ローカルで使用可能な構成ファイルの名前を入力する必要があります。

このフィールドは必須です。

構成パスワード

必要に応じて、選択した構成ファイルの構成パスワードを入力します。このフィールドは、構成がパスワードで保護さている場合にのみ適用されます。

AssemblyLine

開始する AssemblyLine の名前を入力します。

AssemblyLine 操作の構成

このハイパーリンクをクリックすると、「操作の選択」ダイアログが表示されます。 AssemblyLine が 1 つまたは複数のカスタム操作を使用して定義されている場合、このダイアログでカスタム操作を選択できます。その後、この操作に関する AssemblyLine の初期化属性と操作属性の入力を求めるプロンプトが出されます。このラベルは IBM Security Directory Integrator 6.1.X サーバーと IBM Security Directory Integrator サーバーで構成されている場合にのみ表示されます。IBM Security Directory Integrator 6.0 には適用されません。

AssemblyLine の停止

このアクションにより AssemblyLine が停止します。このアクションを選択する場合は、停止する AssemblyLine の名前と、これに関連付けられた構成を指定する必要があります。

サーバー

構成済みサーバーのドロップダウン・リストです。LocalServer は、Action Manager が稼働しているサーバーです。

リモート構成フォルダーからの選択

このチェック・ボックスにチェック・マークを付けると、リモート・サーバーに対し、使用可能な構成ファイルの照会が実行されます。

構成名 「AssemblyLine」フィールド内の AssemblyLine が属する構成を入力します。「**リモート構成フォルダーからの選択**」にチェック・マークを付けた場合は、リモート・サーバーの使用可能な構成ファイルのリストが表示されます。チェック・マークを外した場合は、ローカルで使用可能な構成ファイルの名前を入力する必要があります。

このフィールドは必須です。

AssemblyLine

停止する AssemblyLine の名前を入力します。

ルールの使用可能/不可設定

Action Manager ルールを使用可能または使用不可にするには、「ルールの使用可能/使用不可」を選択します。

ルール名

「ルールを使用可能/使用不可に設定」アクションを実行する、ルールとソリューション・ビューのペアの名前を選択します。IBM Security Directory Integrator の前のバージョンでは、ルールとソリューション・ビューのペアの代わりにルール名を選択しました。これは、現行バージョンで選択可能な機能です。このオプションは、「ルールの使用可能/不可能設定」アクションに属します。

状態 (State)

ドロップダウン・メニューから目的の状態を選択します。「**ルール名**」フィールド内のルールを使用可能に設定する場合は、「**使用可能**」を選択します。ルールを使用不可に設定する場合は、「**使用不可**」を選択します。

ルールの実行

このアクションにより、Action Manager は指定されたルールを実行します。その後 Action Manager は、指定されたルールに関連付けられたアクションを実行します。指定されたルールに関連付けられているトリガー条件を満たす必要はありません。

ルール名

「ルールの実行」アクションを実行する、ルールとソリューション・ビューのペアの名前を選択します。前のバージョンでは、ルールとソリューション・ビューのペアの代わりにルール名を選択しました。これは、現行バージョンで選択可能な機能です。このオプションは、「ルールの実行」アクションに属します。

コマンドの実行

「コマンドの実行」アクションは、「**ターゲット・コンピューターの名前**」の下で指定したターゲット・コンピューター上の「コマンド」フィールドで入力されたコマンドを実行できます。コマンドは、任意の汎用コマンドまたは IBM Security Directory Integrator 固有コマンドにすることができます。「コマンドの実行」は、ターゲット・コンピューターに固有のコマンドまた

は AMC によって公開されていない IBM Security Directory Integrator コマンドを実行するためのルールをユーザーが構成するときに使用できます。例えば、AMC には、サーバーを再始動したり構成ファイルをロードしたりできるアクションはありません。ユーザーは、「IBM Security Directory Integrator サーバー」または「構成ファイル」ウィンドウのいずれかを使用して、コマンドの再開または再ロードを実行する必要があります。コマンドの実行時にエラーが発生すると、そのエラーは %ACTION_ERROR% 変数に取り込まれ、Action Manager がさらにそれを使用できます。

ターゲット・コンピューターの名前
ターゲット・コンピューターの名前または IP アドレス。 Action Manager は、このフィールドで指定されたコンピューターと接続します。コンピューターのホスト名または IP アドレスのどちらも指定されていない場合、コマンドは Action Manager が実行しているコンピューター上で実行されます。

ポート ポートは、コマンドが実行されるターゲット・コンピューターと Action Manager が接続できるチャネルを指定します。

ユーザー名
ユーザー名は、ターゲット・コンピューターとの接続の確立時に、認証と許可のために検査されます。

パスワード
パスワードは、ターゲット・コンピューターとの接続の確立時に、認証と許可のために検査されます。

鍵ストア
鍵ストア・パスは、ターゲット・コンピューターとの接続時に証明書認証が必要な場合に入力されて使用されます。

鍵ストアのパスワード
鍵ストア・パスワードは、ターゲット・コンピューターへの接続に証明書認証が必須である場合に必要です。

プロトコル
リモート・マシンとの接続の確立に使用されるプロトコル。プロトコルは値として、WINDOWS、RSH、SSH、または REXEC (Windows、リモート・シェル、セキュア・シェル、または遠隔実行プロトコル) を取ることができます。

コマンド
実行されるコマンド。

イベントの通知
このアクションにより、Action Manager は、現在のソリューション・ビューに関連付けられた IBM Security Directory Integrator サーバーに、指定された詳細とともにイベントを送信します。このアクションをルールに追加する場合は、「**イベントの通知**」を選択します。このアクションを選択する場合は、イベント・タイプを指定する必要があります。

イベント・タイプ
イベント・タイプを入力します。このフィールドは必須です。

ソース イベント・タイプのソースを入力します。

データ イベント・タイプのデータを入力します。

プロパティーの変更

このアクションにより、Action Manager は、特定の操作および値に基づいてプロパティーを変更します。このアクションを選択する場合は、値も選択する必要があります。

プロパティー名
ドロップダウン・メニューから、変更するプロパティーを選択します。

操作 ドロップダウン・メニューから、プロパティーを変更するために使用する操作を選択します。使用できるオプションには以下のものがあります。

- 設定
- 増分
- 減分

値 目的の値を入力します。これは必須フィールドです。

プロパティー値のコピー

このアクションにより、Action Manager は、ソース・プロパティーの値を宛先プロパティーにコピーします。

コピー元のプロパティー
ドロップダウン・メニューから、コピー元のプロパティーを選択します。

コピー先のプロパティー
ドロップダウン・メニューから、コピー先のプロパティーを選択します。

ログへの書き込み

このアクションにより、指定された重大度、メッセージ、および説明に応じて、呼び出された Action Manager ルールのログが作成されます。このログは、「**状況のモニター**」下の「**AM 結果**」テーブル内の「ソリューション・ビューの詳細」ウィンドウで表示できます。すべてのルールについて、少なくとも 1 つのログ・アクションを指定することを推奨します。このアクションを選択する場合は、「**メッセージ**」フィールドにメッセージを入力する必要があります。

重大度 ドロップダウン・メニューから目的の重大度を選択します。使用できるオプションには以下のものがあります。

- SEVERE
- 警告
- 情報
- FINE

メッセージ
希望するメッセージを入力します。

説明 オプションで、説明を入力します。

E メールの送信

このアクションにより、電子メールが指定した宛先に送信されます。ユーザーは電子メールの内容を指定します。メールの送信前に内容とともに、その他の詳細が Action Manager により示されます。内容の入力域および件名のラインでは、変数 %EVENT_DATA% の値を指定できます。%EventData% を指定すると、E メールの送信時に Eventdata 変数の実際の値が挿入されます。%Action_Error% も同じようにここで置換できます。「Action Manager ログの添付」が有効になっている場合、Action Manager のログ (am_logging.properties ファイルで指定) は電子メール添付ファイルとして送信されます。内容の入力域では、変数 %EVENT_DATA% の値を指定できます。%EventData% をこの内容に指定すると、メールの送信時に Eventdata 変数の実際の値に挿入されます。 %Action_Error% も同じようにここで置換されます。「**Action Manager ログの添付**」が有効になっている場合、Action Manager のログ (am_logging.properties ファイルで指定) は電子メール添付ファイルとして送信されます。

イベント・データの変数置換:

特定の Action Manager トリガーからの出力のデータを選択し、ルールによって起動される特定のアクションでそのデータを使用することができます。

左のナビゲーション・ペインから「**Action Manager**」を選択するか、「**ようこそ**」画面から「**Action Manager**」を選択します。Action Manager の下では、ソリューション・ビューにルールを追加できます。新規ルールに名前を付け、既存のルールを編集または削除できます。他のアクションにデータを設定したり送信するときに、「イベント・データ」を使用可能にできます。

Action Manager を使用して、トリガーに対するアクションを構成するときに、イベント・データを使用可能にできます。Action Manager では、ルールを「**追加**」、「**変更**」、または「**削除**」できます。ルールを追加するときには、ルールの名前を指定し、「**トリガー・タイプ**」を選択します。AMC および Action Manager は、予約された変数の形式で起動されたアクションでデータを使用可能にします。その後、アクションでは、変数に保管されたデータが使用されます。この予約済み変数は、このトリガー用に構成された任意のアクションで使用できます。

以下のトリガー・タイプは、アクションで取り込むことができるイベント・データを作成できるトリガー・タイプです。

- AssemblyLine 開始時: イベント・データは %Event_Data% として使用可能です。
- AssemblyLine 終了時: AssemblyLine 終了時のイベント・データは、%Event_Data% として使用可能です。
- イベント受信時: 受信したイベントからのイベント・データは、%Event_Data% としてマップされます。
- ローカル変数時: ローカル変数イベントからのデータは、%Event_Data% としてマップされます。
- 構成のロード時: 構成のロード時イベントからのイベント・データは、%Event_Data% として使用可能です。
- 構成のアンロード時: このトリガーからのイベント・データは、%Event_Data% として使用可能です。

- AssemblyLine 結果の照会時: イベント・データは %attribute_name% として使用可能です。%attribute_name% 変数は、直前の作業項目の実際の属性に関する詳細で置換されます。
- AssemblyLine 終了コードの検査: イベント・データは、%attribute_name% および %Event_Data% として使用可能です。
  - 「エラー・オブジェクトの検査」を有効に設定: 「AssemblyLine 終了コードの検査」トリガーの構成中に、ユーザーが「エラー・オブジェクトの検査」を有効に設定した場合 (オプションを true に設定)、%Event_Data% 変数は実際のエラー・データで置換されます。%attribute_name% 変数は、アクションに使用できません。
  - 「エラー・オブジェクトの検査」を無効に設定: 「AssemblyLine 終了コードの検査」トリガーの構成中に、ユーザーが「エラー・オブジェクトの検査」を無効に設定した場合 (オプションを false に設定)、%attribute_name% 変数は直前の作業項目の実際の属性の詳細で置換されます。%Event_Data% 変数は、アクションに使用できません。

イベント・データを作成できるトリガー:

リストしたトリガー・タイプを使用して、アクションで取り込むことができるイベント・データを作成できます。

- AssemblyLine 開始時: イベント・データは %Event_Data% として使用可能です。
- AssemblyLine 終了時: AssemblyLine 終了時のイベント・データは、%Event_Data% として使用可能です。
- イベント受信時: 受信したイベントからのイベント・データは、%Event_Data% としてマップされます。
- ローカル変数時: ローカル変数イベントからのデータは、%Event_Data% としてマップされます。
- 構成のロード時: 構成のロード時イベントからのイベント・データは、%Event_Data% として使用可能です。
- 構成のアンロード時: このトリガーからのイベント・データは、%Event_Data% として使用可能です。
- AssemblyLine 結果の照会時: イベント・データは %attribute_name% として使用可能です。%attribute_name% 変数は、直前の作業項目の実際の属性に関する詳細で置換されます。
- AssemblyLine 終了コードの検査: イベント・データは、%attribute_name% および %Event_Data% として使用可能です。
  - 「エラー・オブジェクトの検査」を有効に設定: 「AssemblyLine 終了コードの検査」トリガーの構成中に、ユーザーが「エラー・オブジェクトの検査」を有効に設定した場合 (オプションを true に設定)、%Event_Data% 変数は実際のエラー・データで置換されます。%attribute_name% 変数は、アクションに使用できません。
  - 「エラー・オブジェクトの検査」を無効に設定: 「AssemblyLine 終了コードの検査」トリガーの構成中に、ユーザーが「エラー・オブジェクトの検査」を無効に設定した場合 (オプションを false に設定)、%attribute_name% 変数は直前の作業項目の実際の属性の詳細で置換されます。%Event_Data% 変数は、アクションに使用できません。

イベント・データにアクセスできるアクション:

以下の情報を参照することで、イベント・データにアクセスできるアクションについて詳しく知ることができます。

上記の各トリガーに対して実行されるアクションは、%Event_Data% 変数を使用してトリガーによって作成されるイベント・データにアクセスできます。%Event_Data% が発生するたびに、%Event_Data% はそのトリガーの実際のイベント・データにより置換されます。以下のアクション・タイプは、それぞれのトリガーから使用できるイベント・データを使用できます。

- イベントの通知: ユーザーは、データ・テキスト・フィールドでのみ %Event_Data% 変数を指定できます。
- ログへの書き込み: ユーザーは、データベースのログに記録されるログ・メッセージを確認できます。ログ・メッセージが、%Event_Data% 変数の置換後、500 文字を超える場合、ログ・メッセージは最初の 500 文字で切り捨てられます。データベースには、500 文字という制限があるためです。
- E メールの送信: %Event_Data% により指定されるイベント・データまたは %Action_Error% により指定されるエラー・データは、E メールの件名行で置換されます。Action Manager は、メールの送信前に実行に関してその他のデータを追加します。内容テキスト・ボックスで、変数 %EVENT_DATA% 値を指定できます。%EventData% をこの内容に指定すると、メールの送信時に Eventdata 変数の実際の値が置換されます。また、%Action_Error% も同じようにここで置換できます。「**Action Manager ログの添付**」が有効になっている場合、Action Manager のログ (am_logging.properties ファイルで指定) は電子メール添付ファイルとして送信されます。

「E メールの送信」アクション、「コマンドの実行」アクション、「ログ」アクション、「AssemblyLine の開始」アクションなど、同じトリガーへの応答として実行されるすべてのアクションの場合、%Event_Data% のストリングは、そのトリガーにより生成されたイベント・データで自動的に置換されます。

規則要約の表示:

選択した AssemblyLine の現行の Action Manager を表示できます。

「**規則要約の表示**」をクリックします。このテーブルには、ソリューション・ビューに関連する定義済みのすべてのルール、トリガー、およびアクションがリストされます。表示を終了する場合は、「**クローズ**」をクリックします。ここには、「使用可能」状態のルールのみがリストされます。

プロパティー・ストア

以下の情報を使用することで、プロパティー・ストアとその使用法について詳しく知ることができます。

展開していない場合は、管理およびモニター・コンソールのナビゲーション・エリアにある「**プロパティー・ストア**」カテゴリーを展開します。Java、ソリューション、グローバル、システム、ユーザー・プロパティー、およびパスワード・ストア・プロパティーを追加または編集するには、「**拡張**」>「**プロパティー・ストア**」をクリックします。

希望するプロパティー値を入力したら、「**OK**」をクリックして変更を保存します。

プロパティー・ストアのリスト順序は重要です。プロパティー・ストアは上から下へ順に評価されますが、特定のプロパティーの最終定義が最初に使用されます。デフォルトのシステム・セットアップでは、(ソリューション・ディレクトリーにある) ソリューション固有のプロパティー・ファイル solution.properties に定義されているプロパティーが、システム全体の global.properties ファイルの対応プロパティーよりも優先されます。

注: 一部のシステム・プロパティーと Java プロパティーは読み取り専用です。これらの読み取り専用プロパティーは、それぞれプロパティー・ストアに表示されます。これらのプロパティーを変更しようとしても効果がありません。

ソリューション・ビューの選択

このウィンドウでは、ソリューション・ビューを選択できます。メニューには、ユーザーに*読み取り*、*実行*、*構成の管理*、または*管理* のアクセス権限が付与されているソリューション・ビューのみが表示されます。作成されるソリューション・ビューに対するアクセス権限がない場合には、何も表示されません。ビューを選択したら、「**設定**」をクリックします。

ソリューション・ビューを選択すると、「**ソリューション・プロパティー**」や「**グローバル・プロパティー**」などのプロパティー・タブをクリックして、プロパティーを管理できます。

ソリューション・プロパティー

このウィンドウでは、「ソリューション・プロパティー」リストのプロパティーを追加、編集、および削除できます。

グローバル・プロパティー

このウィンドウでは、グローバル・プロパティーを追加、編集、および削除できます。

Java プロパティー

このウィンドウでは、Java プロパティーを追加、編集、および削除できます。

システム・プロパティー

このウィンドウでは、システム・プロパティーを追加、編集、および削除できます。

パスワード・ストア

このウィンドウでは、パスワード・ストアのプロパティーを追加、編集、および削除できます。

ユーザー・プロパティー・ストア

このウィンドウでは、「ユーザー・プロパティー・ストア」リストのプロパティーを追加、編集、および削除できます。

「プロパティー・ストア」ドロップダウン・メニューには、ユーザーが構成したプロパティー・ストアのリストが含まれます。グローバル、ソリューション、Java、およびパスワード・ストア・プロパティーは含まれません。表示、追加、編集、または削除するプロパティーが関連付けられているプロパティー・ストアを選択します。

ログ管理

以下の情報を使用することで、ログ管理とその使用法について詳しく知ることができます。

展開していない場合は、管理およびモニター・コンソールのナビゲーション・エリアにある「**拡張**」カテゴリーを展開します。すべての AssemblyLine または特定の AssemblyLine のログ・ファイルを削除する場合、あるいは日付を指定してログ・ファイルを削除する場合は、「**ログ管理**」をクリックします。新しいソリューション・ビューを選択すると、「**リフレッシュ**」をクリックできます。「**リフレッシュ**」をクリックすると、選択したばかりのソリューション・ビューに属するすべての AssemblyLine がリストされます。

このウィンドウでは、ソリューション・ビューの名前を選択できます。削除のためにリストされた AssemblyLine は、選択したソリューション・ビューから取り出されます。すべての AssemblyLine または特定の AssemblyLine のログ・ファイルを削除することができます。また、日付によって削除するログを指定することもできます。ログの表示と削除を管理するためには、以下の手順を実行してください。

1. 「**ソリューション・ビュー**」メニューからクリーンアップするログをもつ AssemblyLine のあるソリューション・ビューを選択します。
2. 「**コンポーネントの選択**」セクションでは、以下のいずれか 1 つを行います。
   - 「**すべての AssemblyLine**」ラジオ・ボタンを選択して、選択したソリューション・ビュー内のすべての AssemblyLine のログを削除します。
   - 「**特定の AssemblyLine**」ラジオ・ボタンを選択して、特定の AssemblyLine に関連付けられたログのみを削除します。
3. 「**特定の AssemblyLine**」を選択した場合は、削除するログを持つ AssemblyLine をメニューから選択します。
4. 「**ログ・ファイルの表示**」セクションで、以下のいずれか 1 つを実行します。
   - 選択された AssemblyLine に属するすべてのログを削除するためには、「**すべて**」を選択します。
   - 日付範囲内のログを削除するためには、「**開始日**」および「**終了日**」オプションを使用します。指定された 2 つの日付の間に作成されたログが削除されます。希望する日付を「日付」フィールドに入力します。日付の形式はロケールによって異なります。「カレンダー」ボタンも使用できます。「カレンダー」ボタンを使用すると、カレンダーから日付を選択できます。
   - 以前のバージョンの IBM Security Directory Integrator サーバーでは、特定の日付よりも古いログを削除するには、「**終了日**」オプションを選択します。指定された日付より古いすべてのログが削除されます。
   - 最新のログを保持するためには、「**Display first (最初を表示)**」ラジオ・ボタンを選択します。保管する最近のログの数を入力します。最新のログ・ファイルを維持し、他をリストするように指定するために使用します。編集ボックス

を使用して、 「*最新*」の制限を設定します。「20」を入力した場合、最新の 20 個のログ・ファイルを維持し、残りのファイルをテーブルにリストして、削除のために選択できるように AMC に指示します。「10」という数字を入力すると、最新の 10 個のログが保存されます。

5. 「**ログ・ファイル**」テーブルでは、ログ・ファイルのリストが表示されます。「選択」カラムでは、削除するログを選択し、「**削除**」をクリックします。「**アクションの選択**」メニューでは、以下のオプションのうちの任意のものを選択できます。
   - データのエクスポート
   - すべての選択項目を削除 (Change all selected)
   - テーブルの縮小表示
   - リストア

   これらのオプションのいずれか 1 つを選択し、「**実行**」をクリックします。
6. 選択した条件からの結果の表示で、削除するログを選択し、「**削除**」をクリックして、指定したログを削除します。ログの削除を終了したら、「**閉じる**」をクリックして、このウィンドウを終了します。

優先ソリューション・ビュー

「**優先ソリューション・ビュー**」パネルを使用して、デフォルトでモニター・ウィンドウにロードされるソリューション・ビューを選択できます。

「優先される」構成は、「状況のモニター」ページが開いたときにデフォルトで表示されます。定義されたビューがない場合、このパネルにはビューがないことを示すメッセージが表示されるだけです。ビューのセットが定義されると、ユーザーはデフォルト・ビューとして表示するビューを設定できます。このパネルは、superadmin により自分に割り当てられたビューのセットをもつユーザーならば、誰でも表示できます。

「**選択**」カラムでチェック・ボックスを選択して、「**優先として使用可能に設定**」をクリックすることにより、ソリューション・ビューを優先することができます。

同様に、「**選択**」カラムでチェック・ボックスを選択して、「**優先として使用不可に設定**」をクリックすることにより、ソリューション・ビューの優先状況を使用不可にすることができます。

AMC および AM コマンド行ユーティリティー

以下の情報を使用することで、AMC および Action Manager コマンド行ユーティリティーについて詳しく知ることができます。

AMC とその関連製品である Action Manager (AM) には、多くのコマンド行ユーティリティーが含まれています。これらのコマンド行ユーティリティーは、AMC war ファイルのインストール、アンインストール、再インストールの支援を行います。バックアップおよびリストアのためのスクリプトばかりではなく、マイグレーション・スクリプトもあります。マイグレーション・スクリプトは、AMC および AM の将来のバージョンへのマイグレーションのためのもので、以前のバージョンから

現行バージョンへのマイグレーションのためのものではありません。これらのすべてのスクリプトは、*TDI_install_dir*/bin/amc ディレクトリーにインストールされます。

インストール

install.bat (.sh) スクリプトは、AMC コンソール・モジュールを ISC SE または IBM Dashboard Application Services Hub でデプロイするために使用されます。このスクリプトは、必要な環境変数をセットアップするために、setupCmdLine スクリプトを使用し、また ISC ランタイムの場所と、使用されるランタイムのタイプを判別するために、つまり組み込まれた Web プラットフォームであるか IBM WebSphere Application Server であるかを判別するために、tdiISCHome スクリプトを使用します。このスクリプトは、インストーラーによりコールされます。

使用法: install

このスクリプトでは、パラメーターを使用しません。

uninstall

uninstall.bat (.sh) スクリプトは、ISC SE または IBM Dashboard Application Services Hub から AMC コンソール・モジュールをアンインストールします。このスクリプトは、必要な環境変数をセットアップするために、setupCmdLine スクリプトを使用し、また ISC ランタイムの場所と、使用されるランタイムのタイプを判別するために、つまり組み込まれた Web プラットフォームであるか IBM WebSphere Application Server であるかを判別するために、tdiISCHome スクリプトを使用します。

使用法: uninstall

このスクリプトでは、パラメーターを使用しません。

backupamc

backupamc.bat (.sh) スクリプトは、AMC のすべての構成関連情報 (構成ファイル、ログなど) をバックアップします。 backup_tdiamc フォルダーは、バックアップ・ディレクトリーの中で作成されます。

使用法: backupamc [-d *folder_to_create_backup_in*]

-d オプションが指定されない場合、ファイルは *TDI_install_dir*/bin/amc/ActionManager/backup_tdiamc ディレクトリーにコピーされます。

以下のファイルがバックアップされます。

1. amc.properties
2. logging.properties
3. amcdbschema.xml
4. amcdbhandler.properties

restoreamc

restoreamc.bat (.sh) スクリプトは、AMC の新しいデプロイに対してバックアップされたファイルをリストアします。そのためには、バックアップされたファイルをまず、backupamc スクリプトを使用して取得する必要があります。

使用法: `restoreamc`

このスクリプトでは、パラメーターを使用しません。

migrateamc

これは、1 個のバックアップ、リストア、アンインストール、およびインストール・コマンドを提供します。古い AMC データをバックアップし、古い AMC プラグイン・アーカイブをアンインストールし、新しい AMC をインストールして、古い AMC 構成データをリストアします。

このスクリプトでは、新しい AMC プラグイン・アーカイブを *TDI_install_dir*/amc ディレクトリーにコピーすることを要求します。

使用法: `migrateamc.bat [-d backup_directory]`

start_tdiamc

このスクリプトは、AMC を開始するための便利なラッパー・ユーティリティーです。このスクリプトは内部的に ISC ランタイムを開始します。ランタイムが組み込みの Web プラットフォームである場合、`lwistart` コマンドをコールします。また、ランタイムが IBM WebSphere Application Server の場合、`startServer server1` コマンドをコールします。 ISC ランタイムを開始する前に、スクリプトは `startNetworkServer` コマンドをコールします。このコマンドは保護されたネットワーク・モードで Derby データベースを開始するために使用されます。データベース・タイプが Derby 以外のものである場合、このスクリプトは ISC ランタイムを開始するだけです。

Windows プラットフォームの場合:

使用法: `start_tdiamc [Service name]`

サービス名が渡されると、lwiStart をコールする代わりに、サービスが開始されます。

UNIX プラットフォームの場合

使用法: `start_tdiamc`

stop_tdiamc

このスクリプトは、AMC を停止するための便利なラッパー・ユーティリティーです。このスクリプトは内部的に ISC ランタイムを停止します。ランタイムが組み込みの Web プラットフォームである場合、`lwistop` コマンドをコールします。また、ランタイムが IBM WebSphere Application Server の場合、`stopServer server1` コマンドをコールします。このコマンドを実行した後で、スクリプトは `stopNetworkServer` スクリプトをコールして、Derby データベースを停止します。データベース・タイプが Derby 以外のものである場合、このスクリプトは、ISC ランタイムを停止するだけです。

Windows プラットフォームの場合:

使用法: `stop_tdiamc [Service name]`

サービス名が渡されると、lwiStop をコールする代わりに、サービスが停止されます。

UNIX プラットフォームの場合

使用法: `stop_tdiamc`

startAM

TDI_install_dir/bin/amc ディレクトリーにある startAM.bat(.sh) スクリプトを使用して、Action Manager を開始します。

注: このスクリプトには、Action Manager により要求されるすべての jar に対して定義されたクラスパスがあります。CLASSPATH と DB_CLASSPATH の 2 個の変数があります。 DB_CLASSPATH には、データベースの JDBC コネクティビティーを確立するために必要な .jar ファイルのパスごとのリストがあります。AMC が Oracle、MS SQL Server または DB2 を使用するように構成されている場合は、これらのデータベースの対応する .jar ファイルを DB_CLASSPATH 変数に追加する必要があります。
Windows では、次のように、このスクリプトはすでに登録されたサービスを停止するために使用できるオプションのサービス名パラメーターを受け入れます。

```
startAM.bat [service name]
```

stopAM

Action Manager は、*TDI_install_dir*/bin/amc ディレクトリーにある stopAM.bat(.sh) スクリプトを使用して停止します。このスクリプトは、開始されている AM の processID を使用してそれを強制終了します。processID は startAM スクリプトにより取得され、ファイルに保管されます。保管された processID は、今度は stopAM スクリプトにより読み取られます。

Windows では、次のように、このスクリプトはすでに登録されたサービスを開始するために使用できるオプションのサービス名パラメーターを受け入れます。

```
stopAM.bat [service name]
```

startNetworkServer

このスクリプトは、ネットワーク・モードの Derby データベース・サーバーをポート 1528 で開始するために使用されます。選択されるポートは、Derby のデフォルト・ポートとは異なります。

使用法: `startNetworkServer`

stopNetworkServer

このスクリプトは、ネットワーク・モードの Derby データベース・サーバーを停止するために使用されます。

使用法: `stopNetworkServer`

setDBType

このスクリプトは、使用中のデータベースのタイプを設定するために使用されます。このスクリプトは、プロパティー DB_TYPE をセットします。DB_TYPE が Derby にセットされると、`startNetworkServer` スクリプトを実行するときに、 Derby データベースは `startNetworkServer` スクリプト・ファイルで指定されたホストとポートで開始されます。setDBType はまた、データベースのユーザー名とパスワードをセットします。データベースのユーザー名とパスワードは、 BUILTIN セキュリティー・メカニズムを有効にし、認定ユーザーのリストにそのユーザーを追加するために、`startNetworkServer` で要求されます。

setDBType スクリプトは、DB_TYPE および DB_USER と DB_PASSWORD プロパティーの設定のために、内部で startNetworkServer と stopNetworkServer スクリプトによりコールされます。

backupamcdb

このスクリプトは、AMC データベースのマイグレーション中に使用されます。このスクリプトは AMC データベースをバックアップし、IBM Security Directory Integrator で定義された XML フォーマットでデータをエクスポートします。このスクリプトは、マイグレーション・パスを選択すると、インストーラーによりコールされます。

Usage:

```
backupamcdb -d folder_which_contains_AMC_backup
     -p location_of_the_amc.properties_file
```

restoreamcdb

このスクリプトは、マイグレーション中に AMC データベースをリストアするために使用されます。このスクリプトは、マイグレーション・パスを選択すると、インストーラーによりコールされます。

Usage:

```
restoreamcdb -d folder_which_contains_AMC_backup
     -p location_of_the_amc.properties_file
```

backupam

このスクリプトは、Action Manager プロパティー・ファイルをバックアップするために使用されます。このスクリプトは、am_config.properties および am_logging.properties ファイルをバックアップします。

使用法: backupam [-d backup_directory]

アーカイブされた情報は、バックアップ・フォルダーに作成されます。 -d オプションが指定されない場合、ファイルは *TDI_install_dir*/bin/amc/ActionManager/backup_tdiamc ディレクトリーにコピーされます。

restoream

このスクリプトは、backupam スクリプトを使用してバックアップされた Action Manager のプロパティー・ファイルをリストアするために使用されます。リストア・スクリプトは、 am_config.properties および am_logging.properties ファイルをリストアします。

使用法: restoream [-d backup_directory]

-d オプションが指定されない場合、ファイルは *TDI_install_dir*/bin/amc/ActionManager/backup_tdiamc ディレクトリーからコピーされます。

setAMCRoles

このスクリプトは、IBM Security Directory Integrator AMC をインストールしているユーザーを、ISC admin および SDI AMC Admin 役割にマップするために使用されます。このスクリプトは IBM Security Directory Integrator 7.0 で導入されました。

これらの役割がインストール・ユーザーに付与されると、そのユーザーは新規ユーザーの追加や必要な役割の割り当てを行う権限を得ます。インストール・ユーザーは AMC コンソール・モジュールの管理者となります。

使用法: `setAMCRole` *`username`* `[OS Group]`

OS Group は、AMC を ISC SE でデプロイするときのオプションのパラメーターです。

tdimigam

このスクリプトは、`am_config.properties` ファイルのマイグレーションに使用されます。

このコマンドの使用法:

tdimigam -f propfile [-b backfile] [-n newfile] [-v] [-?]
ここで、
-f propfile - マイグレーションするファイルの名前
-b backfile - 指定された名前でオリジナル・ファイルをバックアップします。
-n newfile - マイグレーションされたファイルに付ける名前
-v - 冗長モードを有効にします。
-? - 使用法ステートメントを出力します。

このコマンドのロギングを制御するファイルは、tdimigam-Log4J.properties です。

tdimigamc

このスクリプトは、`amc.properties` ファイルのマイグレーションに使用されます。このスクリプトのオプションは、それぞれ `am_config.properties` と `global.properties` ファイルのマイグレーションに使用される tdimigam と tdimiggbl のオプションに似ています。

このコマンドの使用法:

tdimigamc -f propfile [-b backfile] [-n newfile] [-v] [-?]
ここで、
-f propfile - マイグレーションするファイルの名前
-b backfile - 指定された名前でオリジナル・ファイルをバックアップします。
-n newfile - マイグレーションされたファイルに付ける名前
-v - 冗長モードを有効にします。
-? - 使用法ステートメントを出力します。

このコマンドのロギングを制御するファイルは、tdimigamc-Log4J.properties です。

addAMCService

このスクリプトは、AMC をサービスとしてシステムに追加するために使用されます。

使用法: `addAMCService` *`Service_Name`*

Windows の場合は、このスクリプトは IBM Platform Integration Toolkit から 汎用 Windows サービス実行可能ファイル (*`TDI_install_dir`*`/bin/amc/amcwinservice.exe`) を登録します。汎用 Windows サービスでは、構成ファ

イル *TDI_install_dir*/bin/amc/amcwinservice.ini を使用します。そのファイルは、サービスの名前と開始と停止コマンドを指定します。このファイルはインストーラーまたは「addAMCService」スクリプトにより自動的に入力されます。

デフォルトでは、ファイルは次のようになります。

```
[Service]
ServiceName=$service_name$
WorkingDirectory="$install_dir$¥bin¥amc"
StartCommand=""$install_dir$¥bin¥amc¥amcservice.bat" start amc am"
StopCommand=""$install_dir$¥bin¥amc¥amcservice.bat" stop amc am"
```

これは、デフォルトでは、AMC サービスは AMC と Action Manager の両方を実行することを意味します。

「addAMCService」をコールした後で、.ini ファイルを編集して、サービスによりどのコンポーネントが実行されるかをカスタマイズします (AMC および AM の両方、AMC のみ、または AM のみ)。

例えば、AMC のみを実行するためには、以下のように開始と停止コマンドを指定します。

```
StartCommand=""$install_dir$¥bin¥amc¥amcservice.bat" start amc"
StopCommand=""$install_dir$¥bin¥amc¥amcservice.bat" stop amc"
```

サービスの開始と停止を実行するには、GUI 「サービス」ユーティリティー (「**コントロール パネル**」 ->「**管理ツール**」-> 「**サービス**」) またはサービス・コントローラー・コマンド行ツールを使用します。

```
sc start <service name>
sc stop <service name>
```

「サービス」ユーティリティーでは、登録サービスの表示名は次のようになりますので注意してください。

```
IBM Tivoli Directory Integrator Administration and Monitoring Console - myamc
```

ここで、「myamc」は「addAMCService.bat」の引数として指定するサービス名です。

UNIX では、スクリプトは/etc/inittab システム・ファイルの末尾に次のような行を追加します。

```
<service name>::once:"<install dir>/bin/amc/amcservice.sh" start amc am
```

サービスを停止するには、*TDI_install_dir*/bin/amc フォルダーから「amcservice.sh」スクリプトを使用します。

```
amcservice.sh stop amc am
```

またはサービスが AMC のみを実行する場合は、

```
amcservice.sh stop amc
```

を使用します。

deleteAMCService

このスクリプトは、サービスとしての AMC をシステムから削除するために使用されます。

使用法: deleteAMCService *Service_Name*

setDerbyProps

このスクリプトは startNetworkServer と stopNetworkServer スクリプトで使用される必要な Derby データベース・プロパティーを設定します。

使用法: setDerbyProps

amcservice

このスクリプトは、構成の管理およびモニター全体を開始/停止します。以下の構成がサポートされています。

- AMC および AM の両方
- AMC のみ
- AM のみ

内部的に、このスクリプトは「start_tdiamc」/「stop_tdiamc」および「startAM」/「stopAM」をコールします。オペレーティング・システム・サービスを登録するときに使用されることを意図しています。

使用法: amcservice [start|stop] [amc] [am]

例:

```
amcservice start amc
amcservice stop amc am
```

ソリューション・ビューおよびルールの作成のウォークスルー・サンプル

ソリューション・ビューの作成、Action Manager でのルールの構成および起動のための手順を参照できます。

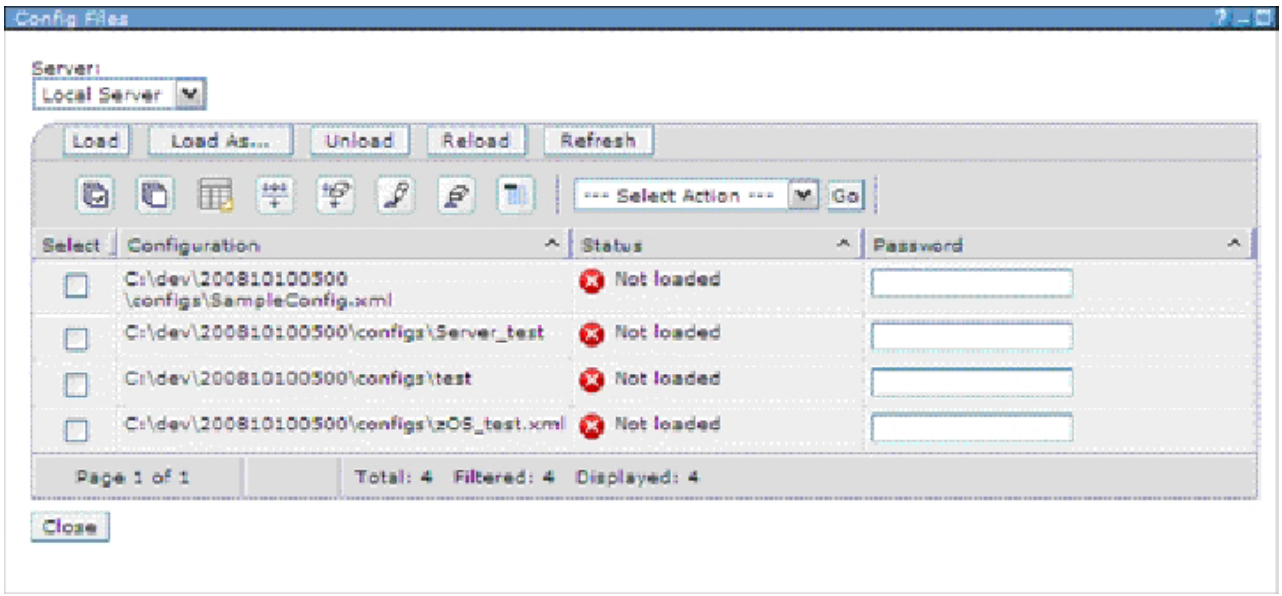
IBM Security Directory Integrator が AMC とともにインストールされていることを想定します。この例で使用されている構成 SampleConfig.xml は、「*始めに*」の『IBM Security Directory Integrator の概要』>『初めての AssemblyLine の作成』でのチュートリアルに従って作成したものです。この構成を *TDI_install_dir*/configs フォルダー にコピーする必要があります。ここで、*TDI_install_dir* は IBM Security Directory Integrator のインストール・ディレクトリーです。このソリューションは、データを「examples/Tutorial/People.csv」ファイルから読み取り、それを「examples/Tutorial/Output.xml」ファイルに書き込みます。

このセクションは、構成ビューの作成、Action Manager でのルールの構成および起動のためのすべての手順を説明します。IBM Security Directory Integrator が AMC とともにインストールされていることを想定します。サンプルの構成 (SampleConfig.xml) および関連ファイルは、ダウンロード・セクションで選択可能で、*TDI_install_dir*/configs フォルダーにコピーする必要があります。このソリューションでは、データを sample.csv ファイルから読み取り、sample.xml に書き込みます。

手順

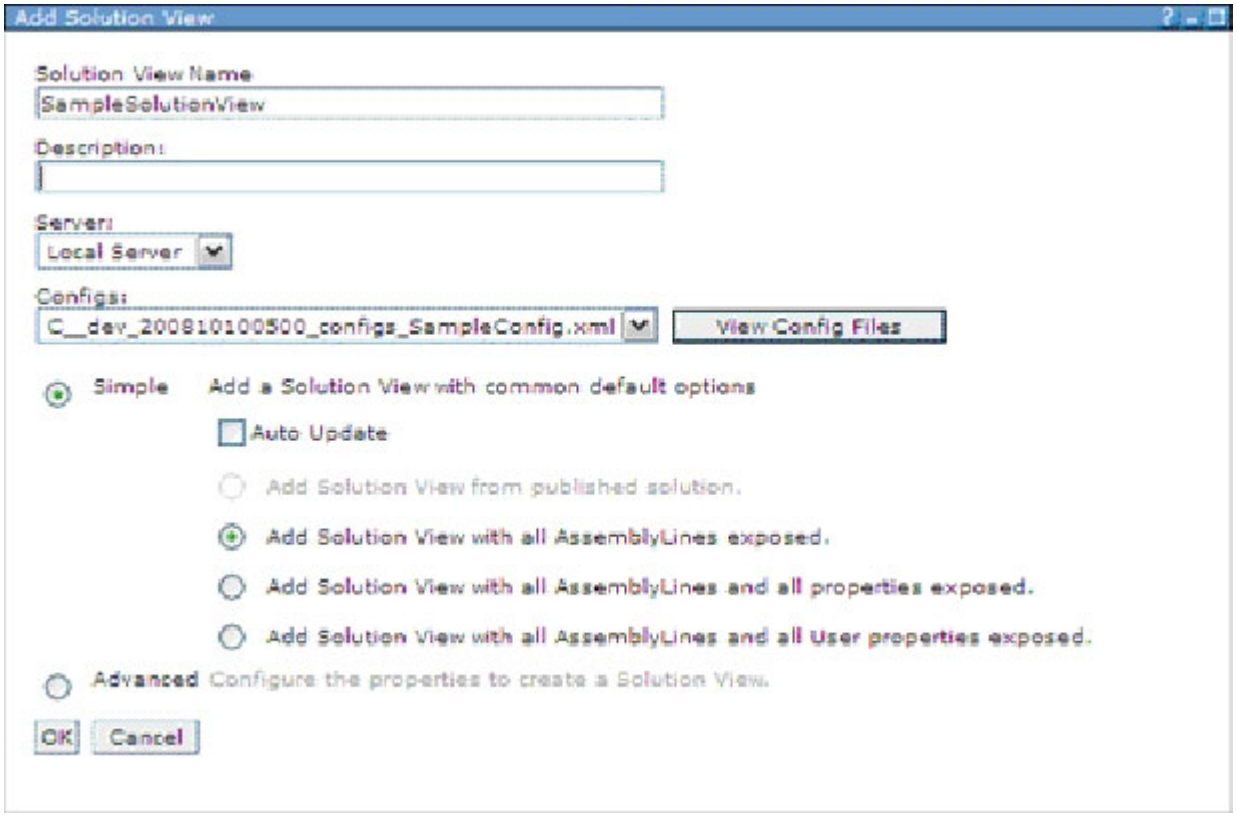
1. デーモン・モードで IBM Security Directory Integrator サーバーを開始します。
2. *TDI_install_dir*¥bin¥amc¥start_tdiamc.bat というコマンドを使用して、AMC を開始します。

3. 構文 `http://hostname:port/ibm/console` の URL とデフォルトのユーザー名とパスワードを使用して、AMC コンソールにログオンします。
4. AMC コンソールへのログオンの後に、ナビゲーション・パネルで「サーバー」を選択し、使用するサーバーを選択します。「**構成ファイル**」ボタンを押すと、以下のパネルが示されます。



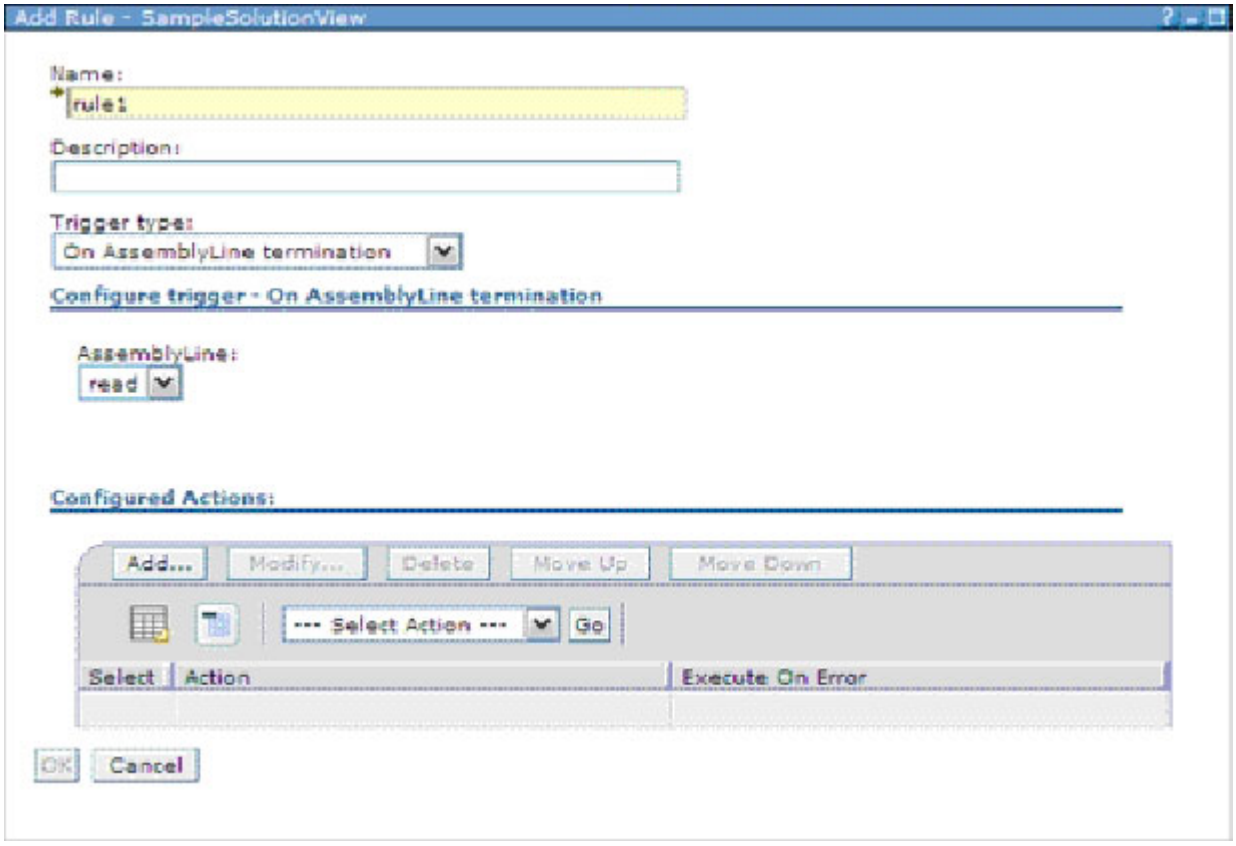
`SampleConfig.xml` ファイルを選択し、「**ロード**」ボタンをクリックします。

5. ナビゲーション・パネルの「ソリューション・ビュー」リンクを選択して、ロードされた構成のソリューション・ビューを作成します。「追加」ボタンを選択すると、次のパネルが表示されます。

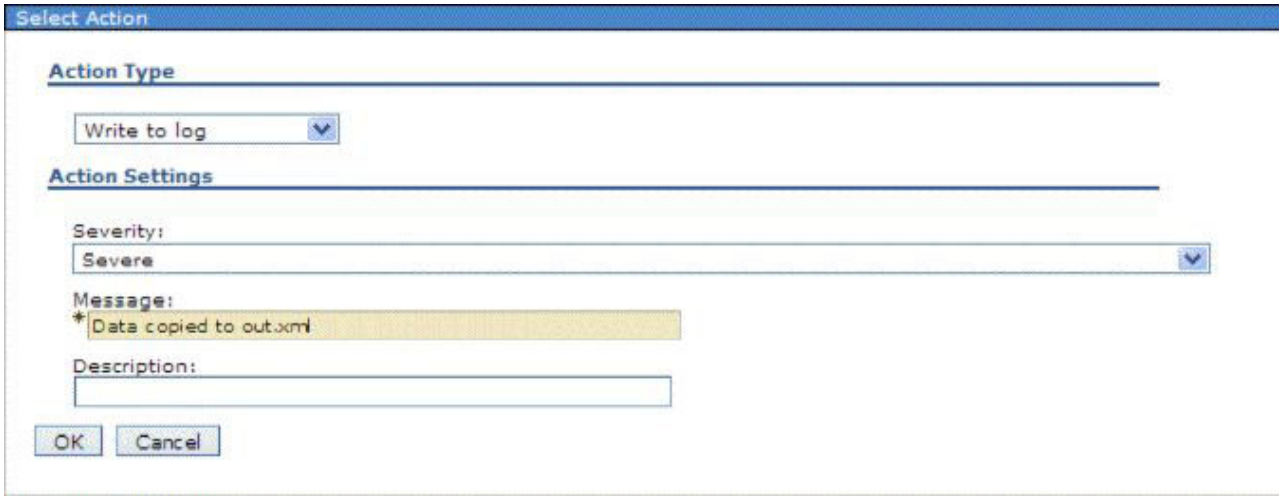


「SampleSolutionView」など、構成ファイルに適した名前を追加します。「OK」を選択すると、ソリューション・ビュー「SampleSolutionView」が正常に作成されたというメッセージが表示されます。

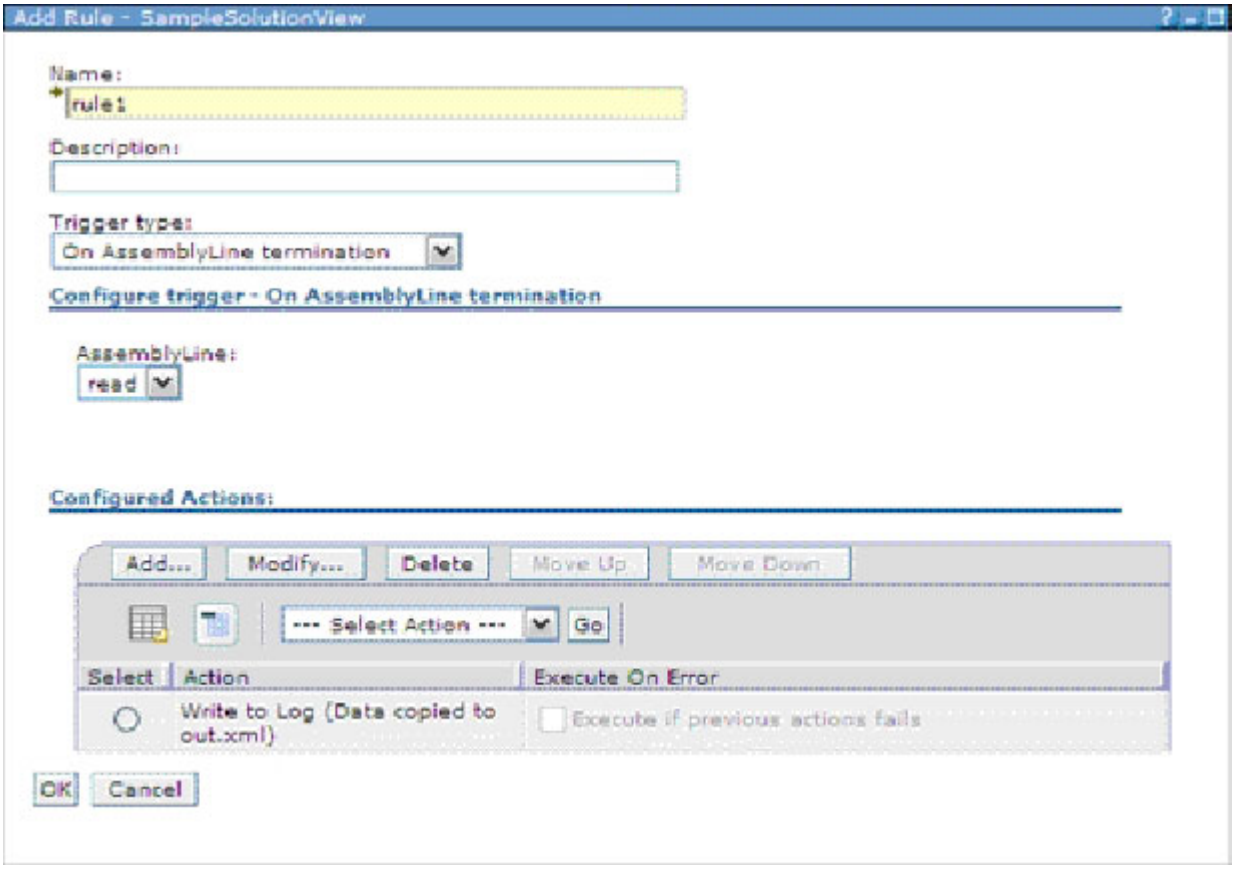
6. ナビゲーション・パネルの Action Manager リンクを選択して、Action Manager ルール構成画面を表示します。「ソリューション・ビューの選択」ドロップダウン・ボックスから、「SampleConfigView」を選択します。構成済みのルール・セクションで「**追加**」ボタンをクリックすると、以下のパネルが表示されます。



「rule 1」などのルールの名前を追加します。トリガー・タイプ「AssemblyLine 終了時」を選択し、「構成済みアクション」セクションの「**追加**」ボタンをクリックします。

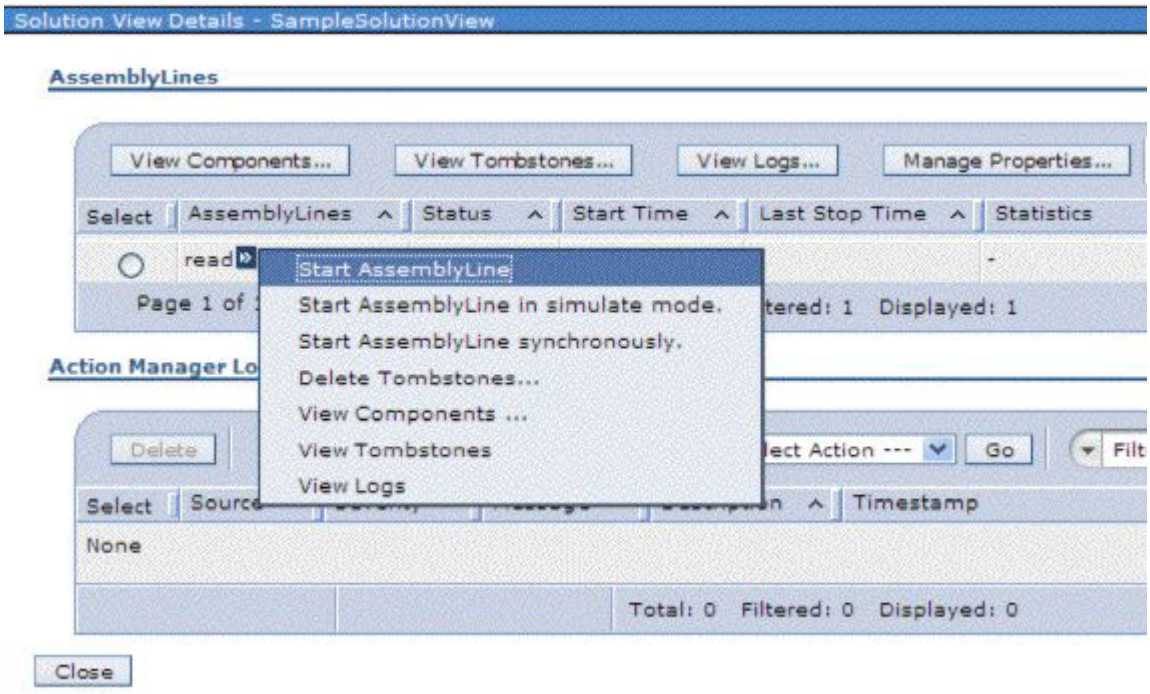


「選択アクション」パネルの「アクション・タイプ」コンボ・ボックスで、「ログ記録への書き込み」オプションを選択します。メッセージ・テキスト・ボックスにテキスト「data copied to out.xml」を追加し、「**OK**」をクリックします。以下のルール・パネルが示されます。

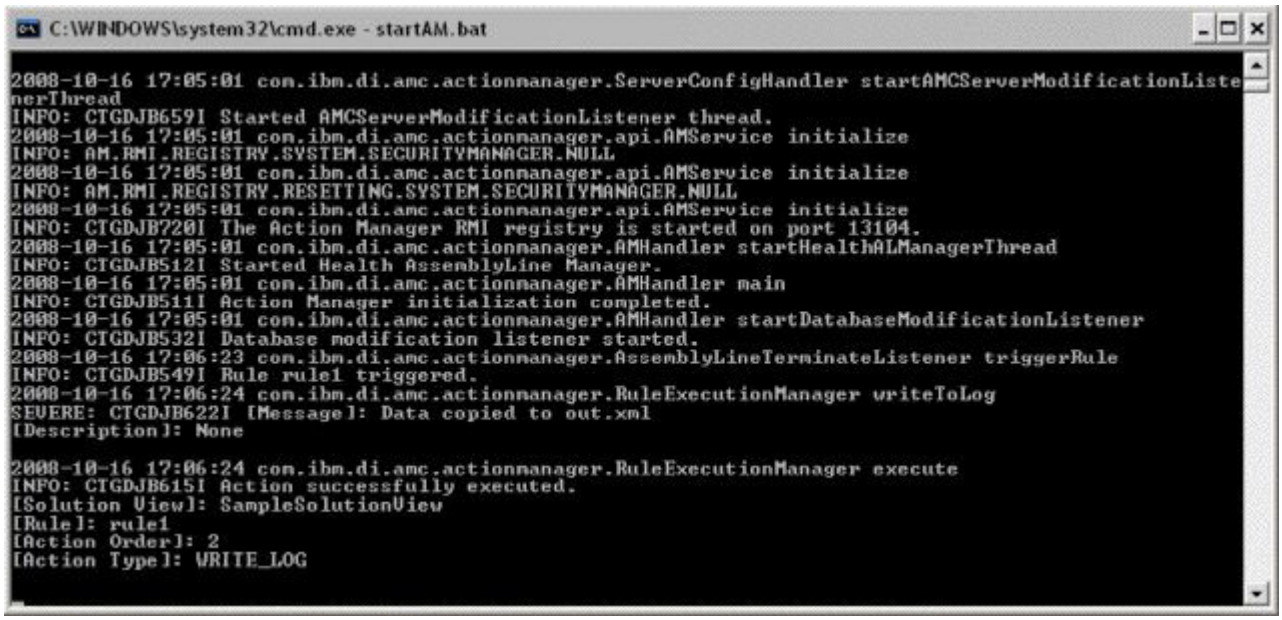


「**OK**」をクリックして、このルールとこの例で必要な AMC 構成の作成を完了します。

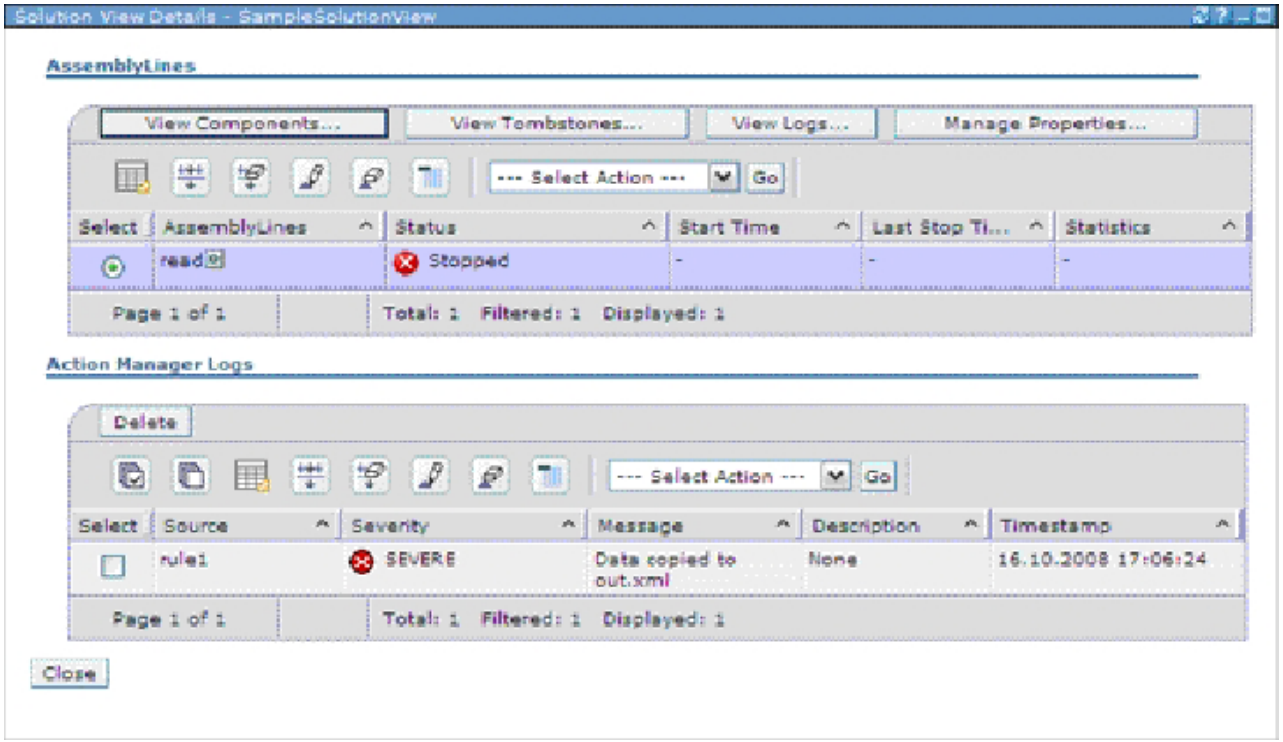
7. Action Manager を *TDI_install_dir*¥bin¥amc¥startAM.bat を使用して開始します。スレッドがルール「rule1」に作成されます。これは、指定した AL の終了を待機します。
8. このルールをトリガーするために、SampleConfig.xml の AssemblyLine の「読み取り」を実行する必要があります。ナビゲーション・パネルの「**状況のモニター**」を選択します。「状況のモニター」パネルで、「SampleSolutionView」を選択し、「**ソリューション・ビューの詳細**」ボタンをクリックします。以下のパネルが表示されます。



9. オプションを使用した AL を上記のとおり開始します。ルールが起動され、以下の状況が Action Manager のコンソールに表示されます。



AMC で、「ソリューション・ビューの詳細」パネルの Action Manager ログ・テーブルが以下のとおり表示されます。


Solution View Details - SampleSolutionView
AssemblyLines
View Components...
View Tombstones...
View Logs...
Manage Properties...
--- Select Action ---
Go
Select
AssemblyLines
Status
Start Time
Last Stop Ti...
Statistics
read
Stopped
Page 1 of 1
Total: 1 Filtered: 1 Displayed: 1
Action Manager Logs
Delete
--- Select Action ---
Go
Select
Source
Severity
Message
Description
Timestamp
rule1
SEVERE
Data copied to out.xml
None
16.10.2008 17:06:24
Page 1 of 1
Total: 1 Filtered: 1 Displayed: 1
Close

第 17 章 Touchpoint Server

タッチポイント・サーバーを使用して、IBM Security Directory Integrator コンポーネントにアクセスすることができます。その実装についての詳細は、以下の情報を参照してください。

タッチポイント・サーバーは、ReSTful 通信プロトコルを使用して IBM Security Directory Integrator コンポーネント (コネクターおよび AssemblyLine) にアクセスする機能を備えています。クライアントはタッチポイント・サーバーに HTTP 要求を送信して、IBM Security Directory Integrator サーバーからクライアントの対話相手となるタッチポイント・インスタンスを作成するように依頼します。

タッチポイント・インスタンスは標準の IBM Security Directory Integrator コンポーネントを使用してインプリメントされているため、HTTP ベースのクライアントは IBM Security Directory Integrator が通信することができるサード・パーティー・システムであれば、そのシステムにアクセスできます。この意味で、タッチポイント・インスタンスはクライアント・アプリケーションとリモート・サービス間の「プロキシー」であると考えることができます。そのため、クライアントは HTTP ベースのインターフェースを持たないさまざまなシステムに、統一化された通信プロトコルを使用してアクセスできるようになります。

タッチポイントの概念

以下の情報を参照することで、タッチポイントのさまざまな概念について詳しく知ることができます。

タッチポイント・プロトコルは、本質的には、IBM Security Directory Integrator のコネクターおよび AssemblyLine へ、HTTP を経由したアクセスを提供するためのプロビジョニング・プロトコルです。タッチポイントを作成すると、リモート・システムへの経由場所となる「プロキシー」が設定できます。一度タッチポイントを作成して、構成すれば、HTTP 要求を送信するだけで済むようになり、相手システムの仕様に配慮する必要がなくなります。

以下のセクションでは、タッチポイントに関連するさまざまな概念について詳しく説明します。

Touchpoint Server

タッチポイント・サーバーを使用して、情報の格納やインスタンスの制御など、さまざまなタスクを実行できます。タッチポイント・サーバーの使用方法については、以下の情報を参照してください。

タッチポイント・サーバーは、特定のドメイン内の定義済みのタッチポイントに関する情報を保管するアプリケーションです。タッチポイント・サーバーを使用すると、リモート・タッチポイント・インスタンスを制御することができます。タッチ

ポイント・サーバーは、その構成、開始、停止を管理します。タッチポイント・サーバーは、IBM Security Directory Integrator サーバー内で実行されるサービスとして提供されます。

タッチポイント・サーバーのクライアントは、タッチポイント・プロバイダー、タッチポイント型、およびタッチポイント・インスタンスの詳細にアクセスするため、アトム・パブリッシング・プロトコルを使用します。定義済みのスキーマについて詳細は、352 ページの『タッチポイントのスキーマ』を参照してください。

タッチポイント・プロバイダー

タッチポイント・プロバイダーを使用してインスタンスを作成できます。タッチポイント・プロバイダーの使用方法、タッチポイント・サーバーとプロバイダーとの違いについての詳細は、以下の情報を参照してください。

タッチポイント・プロバイダーは、タッチポイント・サーバーがタッチポイント・インスタンスを作成するときに使用するサーバーです。ここで説明するタッチポイント・プロバイダーは、任意のバージョン 7.1 以降の IBM Security Directory Integrator サーバーにすることができます。タッチポイント・サーバーが連携できるのは IBM Security Directory Integrator サーバーのみです。他のタッチポイント・プロバイダーはサポートされていません。

タッチポイント・サーバーは IBM Security Directory Integrator サーバーのアドオンとして出荷されます。タッチポイント・サーバーはサーバーの JVM 内で動作するため、ローカル・サーバー API を使用してサーバーと通信できます。これが IBM Security Directory Integrator サーバーがデフォルトでローカル・タッチポイント・プロバイダーとして登録されている理由です。リモート IBM Security Directory Integrator サーバーを表すタッチポイント・プロバイダーを登録するには、リモート・サーバーの RMI 設定を使用する必要があります。 ローカル・サーバーとリモート・サーバー用の登録は、いずれも標準のタッチポイント・サーバー構成ファイルを使用して行います。

注: 現在、構成エディターまたは Webadmin ツール AMC は、タッチポイント・サーバーあるいはタッチポイント・プロバイダーを設定するためのユーザー・インターフェース・パネルを備えていません。すべての構成は、XML 構成ファイルを使用して行う必要があります。

詳細については、357 ページの『タッチポイント構成』を参照してください。

タッチポイント・プロバイダーは一度登録すると、アトム・インターフェースを使用して変更することはできません。また、アトム・インターフェースは、リモート IBM Security Directory Integrator サーバーとの接続を指定しているいくつかの詳細情報を隠蔽します。これらの詳細情報は、タッチポイント・サーバーがタッチポイント・プロバイダーと通信を行うときのみ使用され、プロトコルのクライアントに表示することは意図されていません。

タッチポイント型

以下の情報を参照することで、タッチポイント型のカテゴリーについて詳しく知ることができます。

タッチポイント型は、各タッチポイント・インスタンスのメタ情報を提供し、その動作を決定する抽象表記です。各タッチポイント・インスタンスはただ 1 つのタッチポイント型しか持ちません。一方、1 つの特定のタッチポイント型を持つタッチポイント・インスタンスの数に制限はありません。

タッチポイント型には 3 つのカテゴリーがあります。

標準 このカテゴリーは、選択したタッチポイント・プロバイダーでサポートされる IBM Security Directory Integrator コネクターに対応し、接頭部 `system` で始まります。各タッチポイント・プロバイダーは、各種のコネクターを提供するため、標準タッチポイント型の独自のセットを持ちます。この種の型のいずれかを選択することによって、タッチポイント・インスタンスの構造に対してその型がベース・タッチポイント・テンプレートに依存することを指定します (テンプレートについての詳細は、346 ページの『タッチポイント・テンプレート』を参照)。更に、選択した型は、テンプレートのサービス・コネクター (サード・パーティー・システムと連携するコネクター) の継承を決定します。例えば、RDBMS からデータを読み取る必要があるときは、JDBC コネクターを使用できます。そのため、型 `system:/Connectors/ibmdi.JDBC` をもつタッチポイント・インスタンスを作成し、それを適切に構成します。これらのタッチポイント型は、プロバイダーおよびイニシエーターのタッチポイント役割のみサポートします。

custom

このカテゴリーは、ユーザーが用意するカスタム・タッチポイント・テンプレートに対応し、接頭部 `file` で識別されます。IBM Security Directory Integrator に付属しているベース・テンプレートを使用する代わりに、独自のテンプレートを作成し、自らのニーズに応じて、タッチポイントの動作をカスタマイズすることができます。ただし、その代償として、ベース・テンプレートが備えているある種の柔軟性は失われます。一度カスタム・テンプレートを作成すれば、そのコネクターの型は変更できないので、作成したタッチポイント・インスタンスを制約します。そのため、常に同じ型のリモート・システムとしか連携できなくなります。カスタム・タッチポイント型の使用例としては、RDBMS からデータを読み込んで、それに LDAP サーバーから読み込んだ情報を追加する場合のように、複数のデータ・ソースと連携する場合が挙げられます。これはベース・タッチポイント・テンプレートに依存しているタッチポイント・インスタンス 1 つでは実現できません。これを解決するため、カスタム・タッチポイント・テンプレートを作成し、それに対応したタッチポイント型を使用して (例えば、`file:/template_file_name.xml`)、タッチポイント・インスタンスを作成します。唯一の制約は、この型から作成する後続のタッチポイントもすべて RDBMS および LDAP サーバーと連携して動作するようになることです (サービス・コネクターの型が変更できないため)。これらの型は、すべてのタッチポイント役割をサポートします。

仮想 このカテゴリーは、*virtual://Intermediary* という名前の唯一のタッチポイント型で構成されます。このタッチポイント型は、実際の何らかのリソース (標準型の場合は IBM Security Directory Integrator コネクター、カスタム型の場合はテンプレート・ファイル) と接続される上記 2 つの型とは異なり、すぐに使用可能なタッチポイント中継インスタンスを作成する方法を提供するために使用されます。この目的のため、この型は IBM Security

Directory Integrator に用意されているベース・タッチポイント・テンプレートに依存します。このタッチポイント型がサポートするのは、タッチポイント中継役割のみです。

タッチポイント・インスタンスを作成する場合は、サード・パーティー・システムと通信するコネクターの構成を用意する必要があります。標準のタッチポイント型を使用する場合、これは対応する IBM Security Directory Integrator コネクターの構成を用意することを意味します。カスタム型の場合、カスタム・テンプレートのサービス・コネクターの構成を用意する必要があります。最後の仮想型の場合は構成は不要です。タッチポイント中継インスタンスは自らのタスクを遂行するに当たって HTTP コンポーネントにしか依存しないからです。

タッチポイント・インスタンスの作成に必要なパラメーターはどのように見つければいいのでしょうか。この目的のため、タッチポイント・サーバーはプロパティー・シート定義をサポートしています。これは IBM Security Directory Integrator コンポーネントのスキーマの情報が含まれている XML 文書です。プロパティー・シート定義の URL を取得するには、HTTP GET 要求を特定のタッチポイント型の URL に送信します。定義には、必要なコネクター・パラメーター、そのデフォルト値、およびタッチポイント・インスタンスの構成時に必要となるその他の役立つ情報が定義されています。

プロパティー・シート定義に含まれている情報は、全般的に IBM Security Directory Integrator の構成エディターでコネクターを構成するときに利用できる情報に似ています (パラメーターの説明は除きます)。プロパティー・シート定義とその使用法についての詳細は、362 ページの『プロパティー・シート定義』を参照してください。

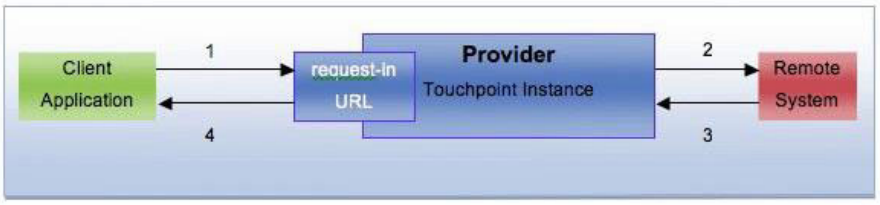
タッチポイント・インスタンス

タッチポイント・インスタンスを使用して、HTTP プロトコルでのアクセスを管理できます。この内容の詳細については、以下の情報を参照してください。

タッチポイント・インスタンスは、本質的に、一般的な HTTP プロトコルを使用してリモート・サービスへのアクセスを可能にするプロキシーに相当します。ただし、タッチポイント・インスタンスの役割に依存して、通信フローは大きく異なっています。以下にサポートされるタッチポイント役割について詳しく説明します。

Provider

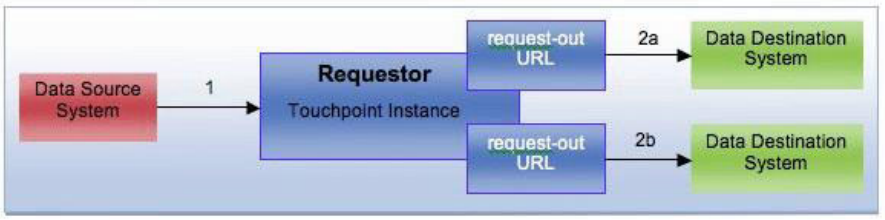
このモードはクライアントのサード・パーティーのサービスへのアクセスを提供します。クライアントは HTTP 要求をタッチポイント・インスタンスに送信します (1)。その要求はリモート・システムのネイティブ言語に変換されてから、リモート・システムに送信されます (2)。リモート・システムはタッチポイント・インスタンスに適切な結果を返送します (3)。それが HTTP 経由でクライアントに戻されます (4)。



このダイアグラムからもわかるように、プロバイダーはユーザーの要求の送信先となる URL (リクエスト・イン URL) で表現される**単一入力インターフェース**を備えています。この URL は固有のものです。すなわち、タッチポイント・インスタンスにより作成され、ユーザーに提供されます。

イニシエーター

このモードは 2 つのエンド間での情報の伝送機能を提供します。タッチポイント・インスタンスは、システム依存言語を使用して別のシステムにデータの断片を要求し (1)、このデータを HTTP 経由でいくつかのデータ宛先システム (例えば、プロバイダー・タッチポイントのようにデータを受信できる HTTP サーバー) に「押し込み」ます (2a、2b)。

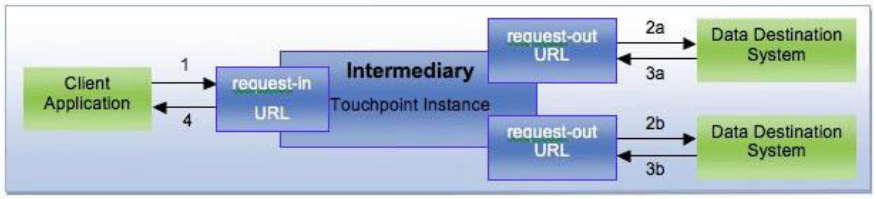


これを実現するため、イニシエーターは**複数の出力インターフェース** (利用可能なデータが送信されるリクエスト・アウト URL) を持つことができます。これらの宛先ポイント (あるいは単に宛先) は、イニシエーターに対して実行時に追加したり、削除したりできます。イニシエーターは最初のリクエスト・アウト URL が追加されると作業を開始し、すべてのリクエスト・アウト URL が削除されたとき作業を終了します。デフォルトでは、イニシエーターのタッチポイントはデータをそのすべての宛先に (応答を無視して) 送信します。この動作は、ベース・テンプレートを編集するか、またはカスタム・テンプレートを用意することによって変更 (例えば、宛先の中でデータの受信に失敗したものがあれば、イニシエーターを停止する) できます。

中継

このモードは転送サービスを提供します。タッチポイント中継インスタンスは要求を受け取り (1)、それを HTTP 経由でいくつかの宛先に送信します (2a、2b)。宛先はこのタッチポイント・インスタンスに応答し (3a、3b)、タッチポイント・インスタンスは応答をマージして結果を呼び出し元に返します (4)。

クライアント・アプリケーションにとって、中継タッチポイントはサード・パーティー・システムへのアクセス手段を提供するプロバイダーのように見えます。一方、データ宛先システムにとっては、データを送信するイニシエーターのように見えます。



中継タッチポイントは、**単一入力インターフェース** (リクエスト・イン URL) と、**複数出力インターフェース** (リクエスト・アウト URL) を持ちます。出力インターフェースはイニシエーター役割と同じように構成します。入力インターフェースはプロバイダー役割と同じように構成します (つまり、URL はタッチポイント・サーバーで設定します)。最も単純な形態では中継タッチポイントは転送データを変更しませんが、ベース・タッチポイント・テンプレートを編集するか、またはカスタム・テンプレートを用意することによって、変更するロジックを用意することもできます。

中継タッチポイントのもう 1 つの特性は、複数のプロバイダーにアクセスするためのプロキシーとして動作するときに現われます。既に説明したように、ユーザーはこの複雑なシステムを 1 つの単純なプロバイダーと見なして対話することができます。ただし、エンド・プロバイダーから返された応答をマージしなくてはならないという明示的な要件はあります。デフォルトで用意されているロジックは、以下のとおりです。

- 宛先の 1 つが正常応答を返した場合、その応答を呼び出し元に返す。
- 複数の宛先が正常応答を返した場合、応答の本文内のデータをマージして、HTTP コード 200 を付けて返す。
- すべての宛先が失敗の応答を返した場合、呼び出し元にはコード 500 のエラー応答を返す。そのとき、HTTP 本文には、各宛先について以下の情報を含める。
  - URL「宛先の URL」への要求は失敗。
  - HTTP ステータス: 返された HTTP エラー・コード
  - HTTP 本文: 返された HTTP 本文

マージ動作を変更するには、使用している 346 ページの『タッチポイント・テンプレート』を編集する必要があります。

クライアントとタッチポイント間の通信プロトコルについての詳細は、359 ページの『タッチポイント・インスタンスの通信プロトコル』を参照してください。

タッチポイント・インスタンスをセットアップする場合は、既に説明した役割に加えて、更に 2 つの構成アイテムを指定する必要があります。これらの属性を以下に示します。

- タッチポイントの**プロパティー・シート**: 各タッチポイント・インスタンスは特定のタッチポイント型に対応しています。すなわち、各タッチポイント・インスタンスは特定の構成を持ちます。この構成はタッチポイント型で定義された構成 (プロパティー・シート定義) に準拠し、このタッチポイント用にリモート・システムと連携して動作する IBM Security Directory Integrator コネクター (サービス・コネクターとして知られます) のスキーマを反映します。

注: 中継タッチポイントは動作時にサービス・コネクターを使用しないので、このような構成情報は不要です。このようなタッチポイントをセットアップする場合、指定したパラメーターはすべて無視されるので、ユーザーは空のプロパティー・シートを送信する必要があります。
プロパティー・シート・フォーマットの詳細については、 357 ページの『インスタンスの構成』を参照してください。

- タッチポイントの**管理状態**: このアイテムを使用すると、インスタンスの状態の細かな管理を行うことができます。例えば、動作中のプロバイダー・タッチポイントを一定時間無効にする状況を考えます。これは、削除する代わりに、管理状態を無効とするだけで実現できます。同様に、再び必要になったら、管理状態を有効に更新するだけで、再び動作を開始させることができます。

  別のユース・ケースは、中継役割とイニシエーター役割に対するものです。これらのタッチポイントに対して最初の宛先を追加すると、直ちに動作を開始します (そして恐らくデータを送信します)。追加の宛先は後から追加できますが、その方法ではデータを紛失する可能性があります。一部のデータが既に送信済みになってしまうからです。この問題を解決するため、無効という管理状態 (開始を阻止) を持つタッチポイントを作成し、それに必要な宛先すべてを追加します。構成が完了すれば、管理状態を有効に変更できます。すると、タッチポイントはすべての宛先にデータの送信を開始します。

タッチポイント・インスタンスを作成した後は、現在の運用状態にアクセスできます。タッチポイントには 2 種類の状態がありますが、その意味はインスタンスの役割によって異なります。

*プロバイダー・タッチポイント・インスタンス*には以下の状態があります。

- **使用不能** - 管理状態を使用して意図的にタッチポイント・インスタンスを無効にした場合です。
- **使用可能** - タッチポイントを構成し、管理状態を有効とした場合です。

プロバイダー・タッチポイントの状態にはもう 1 つの追加パラメーターがあります。運用状態の他に、タッチポイントのリクエスト・イン (プロバイダー・タッチポイントがそれらをリモート・システムに伝達できるようにユーザーが要求を送信する URL アドレス) を取得できます。

*イニシエーター・タッチポイント・インスタンス*には以下の状態があります。

- **使用不能** - タッチポイント・インスタンスがこの状態になるのは、以下の状況です。
  - 構成が完全ではない。つまり、宛先 (リクエスト・アウト URL) がない。
  - 管理状態を設定して、意図的に無効とした。
  - サービス・コネクターがデータ・ソースからの読み込みを終了した。

    この特定のケースは、イニシエーター・タッチポイント・インスタンスのサービス・コネクターの動作によるものです。このコネクターが標準のイテレーターの場合、コネクターは構成されているデータ・ソースを読み込み、終了したとき、停止します。それに応じて、タッチポイント全体が停止します。しかし、変更検出コネクターまたは JMS コネクターの場合は、コネクターは新しいデータを待ち続けるので、タッチポイント・インスタンスは永遠に動作し続

けます (使用可能の状態のままです)。このケースでは、削除するか、管理状態を無効とすることで、明示的に停止させる必要があります。

- **使用可能** - この状態のとき、タッチポイント・インスタンスは、実際に、データ・ソースからデータを読み込んでいます。

既に説明したように、*タッチポイント中継インスタンス*は、本質的に、プロバイダーとイニシエーターのタッチポイント・インスタンスを組み合わせたものです。これは状態にも反映されます。

- **使用不能** - タッチポイント・インスタンスがこの状態になるのは、以下の状況です。
  - 構成が完全ではない。つまり、宛先 (リクエスト・アウト URL) がない。
  - 管理状態を設定して、意図的に無効とした。
- **使用可能** - この状態のとき、タッチポイントの管理状態は有効であり、タッチポイントは着信した要求を宛先に転送しています。

プロバイダー・タッチポイントと同様に、中継タッチポイントもリクエスト・イン URL を持っています。これは状態項目から取得できます。

タッチポイント・テンプレート

タッチポイント・テンプレートを使用して、IBM Security Directory Integrator の構成を開始できます。

タッチポイント・インスタンスが開始されると、タッチポイント・サーバーは IBM Security Directory Integrator の構成を作成し、それを関連付けられたタッチポイント・プロバイダー (IBM Security Directory Integrator サーバー) 上で一時的な IBM Security Directory Integrator ConfigInstance として始動します。IBM Security Directory Integrator 構成は、タッチポイント・サーバーに用意されているベース・テンプレートに基づきます。このテンプレート・ファイルへのパスは、タッチポイント・サーバー 365 ページの『構成』 で指定されます (デフォルトは *TDI_install_dir*/etc/TouchpointTemplate.xml)。これはタッチポイント・サーバー側で保管され、特定の IBM Security Directory Integrator サーバーには関連付けられない一般的なテンプレートです。各タッチポイント・インスタンスの構成は、ConfigInstance として動作を開始する前に、ベース・テンプレートに埋め込まれた構成です。

デフォルトのベース・テンプレートには、以下の構造が含まれています。

AssemblyLine:

- **ProviderServer** - これは HTTP 要求を受信し、その要求に基づいて適切なハンドラー AL を始動する役割を持つ AssemblyLine です。これはタッチポイント・サーバーが管理するすべてのプロバイダーおよび中継タッチポイント・インスタンスにアクセスするための共通のエントリー・ポイントとして機能します。

  このテクニックを使用すれば、これらすべてのタッチポイントと通信するのに必要な HTTP サーバー・コネクターは 1 つのみです。つまり、開いておく必要がある TCP ポートは 1 つのみです (デフォルトでは、1097)。これにより、大量のランダムなポートを開く必要がある場合に生ずる可能性のある潜在的なファイアウォール問題は回避できます。

ProviderServer AssemblyLine は、以下の IBM Security Directory Integrator コンポーネントを使用します。

- *HttpServer* - クライアント・アプリケーションから要求を受け取るコネクターです。デフォルトではポート 1097 で listen します。これは、タッチポイント・サーバーの 365 ページの『構成』 で変更できます。
- *HandleRequest* - 要求に基づいて始動する AssemblyLine を決定する IBM Security Directory Integrator スクリプト・コンポーネントです。

- **ProviderHandler** - この AssemblyLine は、ProviderServer が、プロバイダー・タッチポイント・インスタンスへの要求を受け取ったとき、始動します。これがリモート・システムと実際に通信します。使用されるコンポーネントは、以下のとおりです。
  - *ServiceConnector* - これがリモート・システムと通信するコネクターです。受動状態が設定されます。これは AssemblyLine フローではなく、スクリプトで制御されることを意味しています。このコネクターの重要な特徴は、ベース・テンプレートのライブラリー内の GenericServiceConnector から継承しているということです (`/Connectors/GenericServiceConnector` から継承)。この親コネクターの役割は後から説明します。

    注: タッチポイント AssemblyLine のサービス・コネクターは 1 つのみにすべきです (中継タッチポイントの場合は、ゼロ個です)。複数ある場合は、すべて同じ構成になります。
  - *HandleRequest* - このスクリプトは、サーブレットに似た構造を使用して着信する要求を処理します。このスクリプトは ServiceConnector を初期化し、データ・ソースに対して受信した要求ごとに異なる操作を行います。サポートされるプロバイダー操作についての詳細は、 359 ページの『タッチポイント・インスタンスの通信プロトコル』を参照してください。
- **IntermediaryHandler** - ProviderServer AL は中継タッチポイント・インスタンスに対する要求を受信すると、その要求をこの AssemblyLine にリダイレクトします。デフォルトでは、中継タッチポイントはデータを複数の宛先に転送するだけなので、必要なコンポーネントは次の 1 つのみです。
  - *SendToDestinations* - このスクリプト・コンポーネントは、構成されているタッチポイント宛先に各要求(AL に作業項目の形式で渡されます)を次の呼び出しを使用して転送します。

    ```
    sendToDestinations(work, mergeResponsesCallback)
    ```

    このメソッドはベース・テンプレートのライブラリーにある送信側スクリプト・コンポーネントに含まれています。送信される項目の他に、各宛先から受信する応答をマージするために使用するコールバック機能が必要です。中継タッチポイントのデフォルトのマージ動作の詳細については、 342 ページの『タッチポイント・インスタンス』を参照してください。
- **Initiator** - この AssemblyLine は、イニシエーター・タッチポイント・インスタンス (ProviderServer のエントリー・ポイント経由でアクセスされ

ない唯一のタッチポイント役割) を表現するために使用されます。この AssemblyLine は、以下のコンポーネントで構成されます。

– *ServiceConnector* - これはリモート・システムから受信したデータを AL にフィードするために使用されるコネクターです。ProviderHandler AssemblyLine と同様に、このコネクターも、ベース・テンプレートのライブラリー内の GenericServiceConnector から継承しています。

注: タッチポイント AssemblyLine のサービス・コネクターは 1 つのみにすべきです (または中継タッチポイントの場合のようにゼロ個)。複数ある場合は、すべて同じ構成になります。

– *ConvertToHTTPContent* - サービス・コネクターから読み込んだデータを HTTP 項目に変換するスクリプト・コンポーネントです。

– *SendToDestinations* - このスクリプト・コンポーネントは受信した要求を、次の呼び出しを使用してタッチポイント宛先に転送します。

```
sendToDestinations(work, null)
```

中継タッチポイント・インスタンスと同じ呼び出しが使用されます。唯一の違いは、ここでは各種の宛先から受信する応答をマージするために使用するコールバック機能がないことです。イニシエーターのデフォルトのマージ動作の詳細については、342 ページの『タッチポイント・インスタンス』を参照してください。

リソース (ベース・テンプレートのライブラリー):

- **コネクター** - 各種の AssemblyLine に対して特定のタスクを実行するコネクターには、いくつかの種類があります。

  – *GenericServiceConnector* - このコネクターはすべてのサービス・コネクターの親です。タッチポイント・インスタンスの作成中、タッチポイント・サーバーはこのコネクターを構成して、特定のリモート・システムと連携して動作ができるようにします。各 AssemblyLine 内のサービス・コネクターはここから継承するので、すべて同じ構成になります。

  注:

  1. AssemblyLine 内のサービス・コネクターの名前は、GenericServiceConnector から継承した場合は、重要ではありません。
  2. 1 つの AssemblyLine にサービス・コネクターは 1 つのみ用意する必要があります。

  タッチポイント・サーバーは GenericServiceConnector をタッチポイント型に応じて異なる方法で処理します。

  - **標準**タッチポイント型を使用する場合は、サービス・コネクターの継承の設定とパラメーターの設定の両方で GenericServiceConnector が使用されます。例えば、RDBMS と連携して動作するプロバイダー・タッチポイント・インスタンスを作成するとします。このインスタンスは型 `system:/Connectors/ibmdi.JDBC` から作成し、それをプロバイダー・モードで構成してから、JDBC コネクターに必要となるパラメーターを渡します。これは、GenericServiceConnector か

らの継承が system://Connectors/ibmdi.JDBC にオーバーライドされ、用意した JDBC パラメーターがそれに設定されることを意味します。そのため、ProviderHandler とイニシエーターの AssemblyLine 内の両方の ServiceConnector は同じ構成になります (両方とも、GenericServiceConnector から継承しているからです)。これで、ProviderHandler AssemblyLine は動作を開始し、RDBMS と連携して動作するプロバイダー・タッチポイント・インスタンスが得られたことになります。

- **カスタム**・タッチポイント型を使用する場合、サービス・コネクターの型とそのパラメーター設定を計画する段階で GenericServiceConnector が使用されます。今回は GenericServiceConnector からの継承は変更されません。代わりに、それはスキーマがわかるように、サービス・コネクターの型を判別するために、タッチポイント・サーバーで使用されます。更に、このタッチポイントを構成する段階でもパラメーターには GenericServerConnector が設定されるので、それが子に伝搬します。
- **仮想**タッチポイント型を使用する場合、GenericServiceConnector は (少なくとも現在は) 使用されません。このスキームに含まれるのは、今までのところ、中継タッチポイント型のみであり、この型がサード・パーティー・システムに接続されることはないからです。

– *HTTPClientConnector* - イニシエーターおよび中継タッチポイントが宛先へのデータ送信で使用する HTTP クライアントです。これは送信側スクリプトで使用されますが、タッチポイントでは sendToDestinations() メソッドで使用します。

– *MemoryPropertiesConnector* - MemoryProperties ストアが宛先のリクエスト・アウト URL を保管するために使用するスクリプト・コネクターです。これは、動作中のタッチポイント・インスタンスと通信したり、そのタッチポイントで宛先の追加/除去を行う機能を備えています (詳細は、MemoryProperties ストアの説明を参照)。このコネクターはプロパティー・コネクターの動作を模倣していますが、大きな違いは用意されたデータをファイルに保管しないことです。

• **プロパティー:**

– *MemoryProperties* - 動作中のタッチポイント AssemblyLine と通信するために使用されるプロパティー・ストアです。渡されたデータをメモリーで保管するときに、MemoryProperties コネクターを使用します。各タッチポイントでは独立した ConfigInstance が開始されるため、それぞれが独自の MemoryProperties ストアを持っており、通信メッセージが混ぜ合わされるリスクはありません。

通信手順は、以下のとおりです。

1. タッチポイント・サーバーは、リモート・サーバー API を使用して特定のプロパティー (com.ibm.di.tp.destinations) を MemoryProperties ストアに保管します。その値は、タッチポイント用に構成された宛先パラメーター (例えば、リクエスト・アウト URL やリクエスト・エラー URL) が保管されている java.util.Maps の java.util.List です。

2. イニシエーターまたは中継タッチポイント・インスタンスで宛先の追加または削除を行うたびに、サーバーはこのプロパティーの値を更新して、現在の URL を反映するようにリストを保守します。
3. イニシエーターまたは中継タッチポイント AssemblyLine は、毎回の反復作業でこのプロパティーの現行値を取得し、そのデータを保管されている URL に送信します (そのために、`sendToDestinations()` ルーチンを使用します)。

ベース・タッチポイント・テンプレートには、正しく構成された MemoryProperties ストアが備わっています。タッチポイント・サーバーは、タッチポイントが作成されるたびに MemoryProperties が欠落していないかチェックし、欠落していた場合はそれを構成に追加します。そのため、このストアをカスタム・テンプレートで構成しなくても、デフォルトのストアが使用可能です。一方、MemoryProperties を変更して通信の動作を変更 (例えば、宛先 URL がファイルに格納されるようにプロパティー・コネクターを使用する) した場合は、タッチポイント・サーバーによって新しい値が上書きされることはありません。

- **スクリプト:**
  - *Sender* - `sendToDestinations()` メソッドを備えているスクリプトです。このルーチンは MemoryProperties ストア内の `com.ibm.di.tp.destinations` プロパティーの内容を読み取って宛先のリクエスト・アウト/リクエスト・エラー URL を取得し、HttpClientConnector を使用して用意した作業項目を送信します。

    注: リクエスト・エラー URL はデフォルトでは使用されません。TouchpointTemplate で用意されているのは、ユーザーがデフォルトのエラー・リカバリー・メカニズムを機能拡張したり、カスタム・エラー・リカバリー・メカニズムをインプリメントできるようにするためです。
  - *Utils* - このスクリプトにはタッチポイント AssemblyLine で使用される一連のユーティリティー機能が備わっています。

カスタム・テンプレートを使用すると、 AssemblyLine に複雑な動作やカスタマイズした動作を設定し、それを新規のタッチポイント型として提供することができます。デフォルトのベース・テンプレートは、IBM Security Directory Integrator 内部で何が行われるか詳しい知識がなくても、IBM Security Directory Integrator コンポーネントと中継タッチポイントを直接プロビジョニングするために使用されます。しかし、タッチポイント・インスタンスに単一コンポーネントよりも多くのロジックやコネクターを追加したくなる場合があります。これに対処するため、タッチポイント・サーバーはカスタム・テンプレートを受け付けて、新規のカスタム・タッチポイントの型や動作が利用できるようにします。

カスタム・テンプレートの構造は、ベース・テンプレートの構造と同じにする必要があります。しかし、カスタム・タッチポイント型ですべてのタッチポイント役割をサポートする必要がなければ、 AssemblyLine のサブセットのみ用意しても構いません。各役割について最小要件は以下のとおりです。

- **プロバイダー役割**: これをサポートするには、カスタム・テンプレートには *ProviderHandler* AssemblyLine と以下のライブラリー・コンポーネントが必要です: *Utils* スクリプト・コンポーネントおよび *GenericServiceConnector*。

  注: ベース・テンプレートの ProviderServer AL は、すべてのタッチポイント・インスタンスに要求を委任するために使用されます。したがって、カスタム・テンプレートに ProviderServer AL が存在していたとしても、それはタッチポイント・サーバーでは使用されません。代わりに、ベース・テンプレートの ProviderServer AL が使用されます。
- **イニシエーター役割**: この役割には、イニシエーター AssemblyLine と以下のライブラリー・コンポーネントが必要です。 *Sender* および *Utils* スクリプト・コンポーネント、*MemoryPropertiesConnector*、*GenericServiceConnector*、および *HttpClientConnector*。

  注: MemoryProperties ストアは、欠落していてもタッチポイント・サーバーで追加されるため、リストしてありません。
- **中継役割**: この役割には、*IntermediaryHandler* AssemblyLine と以下のライブラリー・コンポーネントが必要です。 *Sender* および *Utils* スクリプト・コンポーネント、*MemoryPropertiesConnector*、および *HttpClientConnector*。

要件がカスタム型のテンプレートで満たされていない役割でタッチポイント・インスタンスの作成を試みると、例外がスローされます。

ベース・タッチポイント・テンプレートを編集する場合、あるいはカスタム・テンプレートを作成する場合は、以下の重要な要件に留意しておく必要があります。

1. それらのサービス・コネクター (サード・パーティー・システムと通信するコネクター) はライブラリーにある GenericServiceConnector を継承する必要があります。
2. タッチポイント AssemblyLine のサービス・コネクターは 1 つのみにすべきです。複数あっても、同じ構成が適用されるため、無駄になります。
3. 宛先 URL との通信で使用するストアを変更する場合、MemoryProperties の名前は変更しないでください。

リソースのパーシスタンス

以下の情報を参照することで、パーシスタンスとその使用について学習できます。また、タッチポイント・インスタンスの再始動時の条件を確認することもできます。

タッチポイント・サーバーはアトム文書のパーシスタンスをサポートします。これは自らが自動的に生成する場合と、リモート・サーバーから提供される場合があります。サーバーが使用するパーシスタンスは、構成フォルダー内にツリーに似た構造で保管されます。デフォルトのパーシスタンス・ディレクトリーは、`solution_directory/tp_state` です。

パーシスタンス・リソースは、タッチポイント・サーバーの始動時に再び読み込まれます。この時点で、タッチポイント・サーバーは完全なリソース・ツリーをリストアします。タッチポイント・サーバーは、タッチポイント・インスタンスがサー

バーの再始動後でも存続するようにするため、タッチポイント・インスタンス構成を保持します。以下のすべての条件が満たされたとき、タッチポイント・インスタンスは再始動します。

- タッチポイント・インスタンスのすべての必須構成が、パーシスタンス・ストレージ内に存在する。
- リモート・タッチポイント・プロバイダーが稼働中である。
- 同じ構成を持つ他のタッチポイント・インスタンスがタッチポイント・プロバイダー上で動作していない。

リモート・タッチポイント・プロバイダーが動作中ではなかった場合、タッチポイント・プロバイダーが動作を開始した直後に、タッチポイント・インスタンスが自動的に動作を再開することはありません。タッチポイント・プロバイダーがタッチポイント・サーバーよりも前に動作していたことを確認するか、またはタッチポイント型フィード・リソースに GET 要求を送信することによって、タッチポイント・サーバーに、リモート・タッチポイント・プロバイダーとの接続を強制的に更新させる必要があります。

パーシスタンス・ディレクトリー内のファイルを編集することはお勧めできません。現在、これらのファイルの編集を行うのは、タッチポイント・サーバーのみが想定されています。

タッチポイントのスキーマ

タッチポイントのスキーマは多くのコンポーネントから構成されています。以下の情報を参照することで、スキーマについて詳しく知ることができます。さらに、リソース・ツリーおよび許可されている操作についても理解できます。

このセクションでは、タッチポイント・サーバーと、それをタッチポイント・インスタンスのプロビジョニングのために利用するクライアントとの間の通信プロトコルについて説明します。このセクションでは、クライアントとタッチポイント・インスタンス自体との間の通信プロトコルについては説明しません。

タッチポイント・サーバーは、タッチポイント・インスタンスの定義に含まれている各種のリソースにアクセスするためのメカニズムを備えています。これらのリソースはツリーに似た形式で表現されます。各リソースへのアクセスは、HTTP/HTTPS プロトコル上でアトム文書を使用して行います。

これがタッチポイント・サーバーが使用するスキーマです。

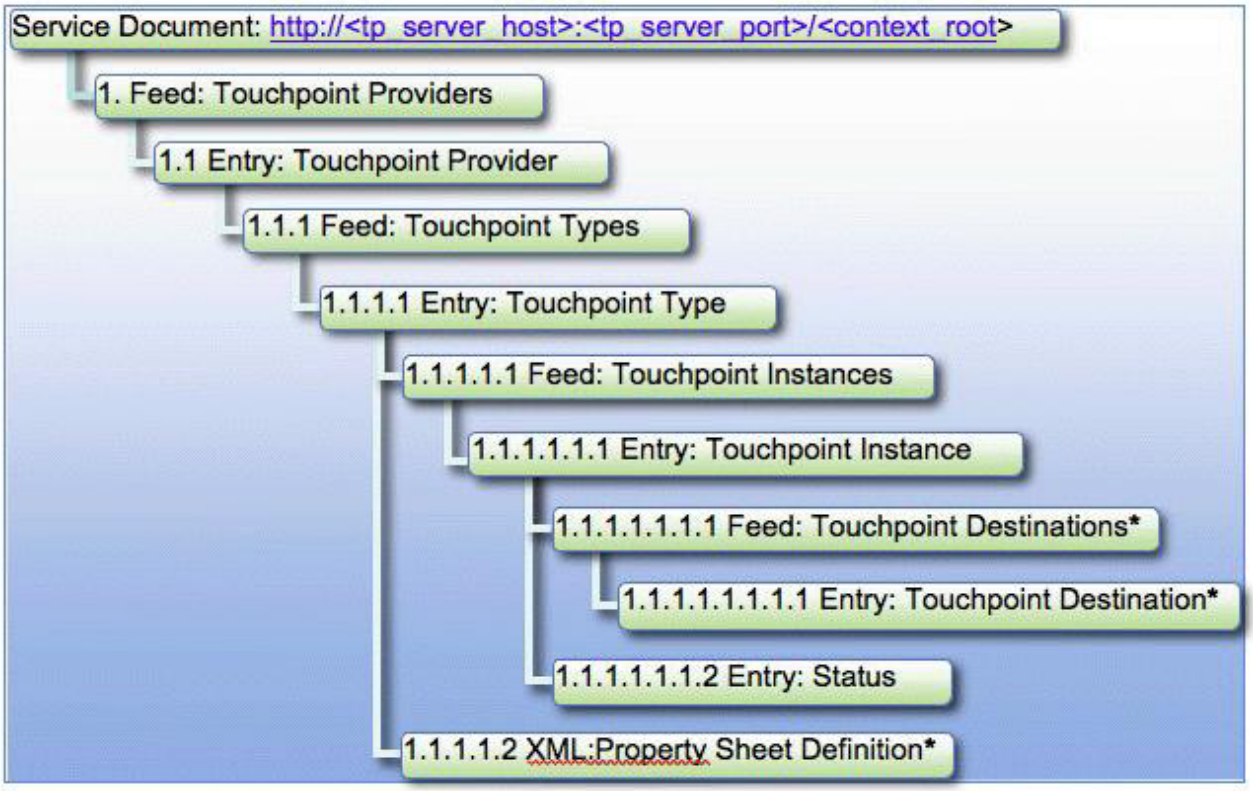


図 1. タッチポイント・サーバーのスキーマ・ツリー

* これらのツリーのノードは、特定のケースでのみ存在します。詳細については、以下の表を参照してください。

上記のスキーマでは、以下の変数が使用されています。

- **tp_server_host** - タッチポイント・サーバーが listen するホスト・アドレス
- **tp_server_port** - タッチポイント・サーバーが listen するポート (デフォルトは 1098)
- **context_root** - タッチポイント・サーバー・アプリケーションを使用可能なコンテキスト・ルート (デフォルトは「tp」)

ツリーのナビゲーションは ReSTful の流儀で行います。これは、クライアント・アプリケーションはエントリー・ポイント (すなわち、サービス文書の URL) と参照の型 (アトム・リンク) のみを認識し、タッチポイント・サーバーがリソース・ツリー内の各ノードへのアクセス方法を定義するというものです。これらの参照 (URL) はタッチポイント・サーバーが自動的に生成します。URL は一度取得されると、タッチポイント・サーバーが新バージョンにアップデートされるまで、同じ値のままです。これは、クライアント・アプリケーションは取得した URL をタッチポイント・サーバーがアップデートされるまで「覚えて」いることも可能ですが、タッチポイント・サーバーがアップデートされた場合は、特定のリソースの URL を再取得する必要があることを意味します。

以下の表に示したプロトコルを使用する場合、クライアント・アプリケーションがサービス文書から始まるタッチポイント・インスタンス・フィードをナビゲートできるようにするには、以下の手順が必要になります。

1. サービス文書 URL に対して HTTP GET 要求を送信します。すると、タッチポイント・プロバイダー・フィード URL を取得できたサービス文書が返されます。
2. タッチポイント・プロバイダー・フィード URL に対して HTTP GET 要求を送信します。すると、タッチポイント・プロバイダー項目参照 URL を取得できたフィード文書が返されます。
3. タッチポイント・プロバイダー項目参照 URL に対して HTTP GET 要求を送信します。すると、特定のタッチポイント・プロバイダーに対応する項目文書が返されます。この項目には、タッチポイント型フィードの URL が含まれています。
4. タッチポイント型フィード URL に対して HTTP GET 要求を送信します。すると、このコンテキストで有効なすべてのタッチポイント型項目の完全なコピーを含むフィード文書が返されます。クライアント・アプリケーションは、タッチポイント型項目からタッチポイント・インスタンス・フィード URL を取得できます。

以下の表で、各リソースに対して許可されている操作について説明します。また、以下の説明は、リソース・ツリー全体に当てはまります。

- 各項目には、スタンドアロン項目ドキュメントを指す、関係「自分自身」を持つリンクが含まれます。
- HTTP PUT および DELETE メソッドを受け付けるリソースは、関係「編集」のリンクを持ちます。これらの要求はすべてこのリンク (URL) に送信する必要があります。
- タッチポイント・サーバーへのすべての要求には『HTTP 仕様書』で定義されている HTTP ETag 応答ヘッダーで注釈が付けられます。ETag 値を要求ヘッダー If-Match、If-None-Match、および If-Range と組み合わせて使用すると、タッチポイント・サーバーに、要求をサービスする前のクライアント・アプリケーションの前提条件を認識させることができます。

表 32. リソースごとに許可されている操作

リソース	GET	POST	PUT	DELETE
サービス・ドキュメント	使用可能なサービスのリストを含むサービス・ドキュメントを検索します。タッチポイント・プロバイダー・フィードの URL には、「href」属性として、カテゴリー「connectivity-provider」に属するコレクションが設定されます。	該当なし	該当なし	該当なし
1. フィード: タッチポイント・プロバイダー **カテゴリー (用語: スキーム):** **connectivity-provider:** http://www.ibm.com/xmlns/prod/scmp#resource	タッチポイント・プロバイダー項目のリストを検索します。すべての項目は、使用可能なタッチポイント・プロバイダーを表わす実際の項目文書に対する参照です。	該当なし	該当なし	該当なし

表 32. リソースごとに許可されている操作 (続き)

リソース	GET	POST	PUT	DELETE
1.1 項目: タッチポイント・プロバイダー	タッチポイント・サーバー構成ファイル内で設定されているタッチポイント・プロバイダー項目を検索します。この項目には、<{http://www.ibm.com/xmlns/prod/scmp}:data/> エレメント内の追加詳細情報も含まれます。タッチポイント型フィードへのリンクは、関係「http://www.ibm.com/xmlns/prod/scmp#touchpoint」を使用して提供されます。	該当なし	該当なし	該当なし
1.1.1 フィード: タッチポイント型 **カテゴリー (用語: スキーム):** **タッチポイント:** http://www.ibm.com/xmlns/prod/scmp#resource	タッチポイント型項目のリストを取得します。これらの項目は、タッチポイント型を表わす実際の項目文書の完全なコピーです。	該当なし	該当なし	該当なし
1.1.1.1 項目: タッチポイント型 **カテゴリー (用語: スキーム):** **タッチポイント:** http://www.ibm.com/xmlns/prod/scmp#resource **resource-type:** http://www.ibm.com/xmlns/prod/scmp#aspect **タッチポイント型** http://www.ibm.com/xmlns/prod/scmp#touchpoint-type を一意に識別する用語	タッチポイント型項目を検索します。タッチポイント・インスタンス・フィードへの URL は、関係「http://www.ibm.com/xmlns/prod/scmp#instance-feed」を持つリンクとして提供されます。 プロパティー・シート定義 XML への URL は、関係「http://www.ibm.com/xmlns/prod/scmp#property-sheet-definition」を持つリンクとして提供されます。 **注:** 仮想タッチポイント型に、プロパティー・シート定義はありません。中継タッチポイントにコネクターを構成する必要がないからです。	該当なし	該当なし	該当なし
1.1.1.1.1 フィード: タッチポイント・インスタンス **カテゴリー (用語: スキーム):** **タッチポイント:** http://www.ibm.com/xmlns/prod/scmp#resource	タッチポイント・インスタンス項目のリストを検索します。すべての項目は、使用可能なタッチポイント・インスタンスを表わす実際の項目文書を参照します。	新規のタッチポイント・インスタンス項目を作成します。項目には 357 ページの『タッチポイント構成』 を含んだ 357 ページの『タッチポイント構成』 が含まれる必要があります。項目は、 "http:// www.ibm.com/ xmlns/prod/ scmp #touchpoint -role" スキームのカテゴリーを含む必要があります。	該当なし	該当なし

表 32. リソースごとに許可されている操作 (続き)

リソース	GET	POST	PUT	DELETE
1.1.1.1.1.1 項目: タッチポイント・インスタンス **カテゴリー (用語: スキーム):** **タッチポイント:** http://www.ibm.com/xmlns/prod/scmp#resource **provider-tp または initiator-tp または intermediary-tp:** http://www.ibm.com/xmlns/prod/scmp#aspect **タッチポイント型** http://www.ibm.com/xmlns/prod/scmp#touchpoint-type を一意に識別する用語	タッチポイント・インスタンス項目を検索します。この項目は、次のような、タッチポイント・インスタンスを記述しているリソースに対する 3 つのリンクを含みます。 • このタッチポイント・インスタンスの更新/削除で使用する URL は、関係「edit」を持つリンクを使用して提供されます。 • 宛先フィードの URL は、関係「http://www.ibm.com/xmlns/prod/scmp#tp-destination」を持つリンクを使用して提供されます。 **注:** プロバイダー・タッチポイントは宛先をサポートしないので、このリンクはありません。 • 状態項目の URL は、関係「http://www.ibm.com/xmlns/prod/scmp#status」を持つリンクを使用して提供されます。 • このタッチポイント・インスタンスが属するタッチポイント型項目の URL。このリンクは関係「http://www.ibm.com/xmlns/prod/scmp#resource-type」を持ちます。	該当なし	タッチポイント・インスタンス項目を更新します。用意する項目には、変更分のみではなく、項目文書の完全なコピーが必要です。この操作ではタッチポイント・インスタンスの役割は変更できません。この操作は再構成のため、動作中のタッチポイント・インスタンスを再始動します。	タッチポイント・インスタンス項目を削除します。
1.1.1.1.1.1.1 フィード: タッチポイント宛先	タッチポイント・インスタンス宛先フィードを検索します。このフィードでは複数のタッチポイント・インスタンス宛先項目を持つことができます。	新規のタッチポイント・インスタンス宛先項目を作成します。宛先へのリクエスト・アウト URL を構成するためのデータ・エレメントを含む必要があります。	該当なし	該当なし
1.1.1.1.1.1.1.1 エントリー: タッチポイント宛先 **カテゴリー (用語: スキーム):** **tp-destination:** http://www.ibm.com/xmlns/prod/scmp#resource	<{http://www.ibm.com/xmlns/prod/scmp}:data/> エレメント内のリモート HTTP サービスへのリクエスト・アウト URL を含むタッチポイント宛先項目を検索します。この項目は、このリソースに対する更新/削除を可能にする、関係「edit」を持つリンクを提供します。	該当なし	タッチポイント・インスタンス項目を更新します。用意する項目には、変更分のみではなく、項目文書の完全なコピーが必要です。この操作ではタッチポイント・インスタンスの役割は変更できません。この操作ではリクエスト・アウト URL を変更するために、動作中のタッチポイント・インスタンスを再始動することはありません。	タッチポイント宛先項目を削除します。

表 32. リソースごとに許可されている操作 (続き)

リソース	GET	POST	PUT	DELETE
1.1.1.1.1.1.2 項目: 状況 カテゴリー (用語: スキーム): **タッチポイント:** http://www.ibm.com/xmlns/prod/scmp#resource **状況:** http://www.ibm.com/xmlns/prod/scmp#aspect	特定のタッチポイント・インスタンスについて運用状態が記述されているタッチポイント・インスタンス状況項目を検索します。これは <{http://www.ibm.com/xmlns/prod/scmp}:data/> エレメントに含まれています。 **注:** プロバイダーおよび中継タッチポイントの場合、状況には、クライアントがタッチポイントの照会で使用する必要があるリクエスト・イン URL も 1 つ含まれます。	該当なし	該当なし	該当なし
1.1.1.1.2 XML: プロパティー・シート定義	選択したタッチポイント型の 362 ページの『プロパティー・シート定義』 を取得します。ここには、IBM Security Directory Integrator コネクターのスキーマが XML 文書の形式で保持されています。	該当なし	該当なし	該当なし

タッチポイント構成

タッチポイント・インスタンスを構成および始動するために構成データのスキーマを指定できます。

各構成データ・エレメントは、アトム文書内の **<data>** エレメントに含まれます。このデータ・エレメントが属する必要のある名前空間は、`http://www.ibm.com/xmlns/prod/scmp` です。

インスタンスの構成

以下の詳細を参照することで、構成データについて詳しく知ることができます。

この構成データでは以下の項目を指定します。

- タッチポイント・インスタンスが実行されるときの役割。タッチポイント・インスタンスは、テンプレートの中から使用する AL を決定するときにこのデータに依存します。後述するアトム文書内では、役割には、カテゴリー・エレメントに「{role}」トークンが付けられます。サポートされる値は、`provider-tp`、`initiator-tp`、および `intermediary-tp` です。
- 作成したタッチポイント・インスタンスの管理状態。アトム文書内では、「{admin_state}」トークンでマークされます。サポートされる値は、`enabled` および `disabled` です。
- タッチポイント・インスタンスのサービス・コネクターの構成パラメーターを含むプロパティー・シート XML。{param_name} は、標準型の場合は、選択したタッチポイント型のプロパティー・シート定義から、またカスタム型の場合は、サービス・コネクターのプロパティー・シート定義から決定されます。
  「{param_value}」は、選択した構成に応じて、ユーザーが決定します。構成パラメーターの他に、パラメーターで {param_name} に「$initMode」に等しい値を設定し、ストリング値にモード名 (例えば、`Iterator`、`AddOnly`) を設定することによって、サービス・コネクターのモードを設定することもできます。このパラメーターの詳細については、 362 ページの『プロパティー・シート定義』を参照してください。

- 特定のタッチポイント・インスタンスに対して、作成者のみに意味のある追加情報を指定できるように、2 つのパラメーター **{TouchpointID}** および **{version}** が用意されています。作成者は、これらの値がクライアント・アプリケーションのコンテキストで有効あることに責任を持ちます。これらの値は、タッチポイント・サーバーにとっては、単に存在しているだけであり、解釈を行うことはありません。

タッチポイント・インスタンス項目リソースの作成時に POST されるアトム文書は、次のとおりです。

```
<entry xmlns="http://www.w3.org/2005/Atom">
 <id>{id}</id>
 <title>Touchpoint Instance Title</title>
 <author><name>Author Name</name></author>
 <content/>
 <category term="{role}" scheme="http://www.ibm.com/xmlns/prod/scmp#touchpoint-role" />
 <scmp:data
xmlns:scmp="http://www.ibm.com/xmlns/prod/scmp"
     xsi:schemaLocation="http://www.ibm.com/xmlns/prod/scmp
     http://localhost:1098/tp/schema/touchpoint.xsd" >
  <scmp:touchpoint>
   <scmp:admin-state>{admin_state}</scmp:admin-state>
   <touchpointID>{touchpoint_id}</touchpointID>
   <version>{version}</version>
   <scmp:propertySheet>
    <scmp:property propertyName="{param_name}">
     <scmp:value>{param_value}</scmp:value>
    </scmp:property>
    ...
   </scmp:propertySheet>
  </scmp:touchpoint>
 </scmp:data>
</entry>
```

作成したタッチポイント・インスタンスの ID は「**{id}**」トークンの代わりに提供されることに注意してください。 ただし、渡された値が何であっても、タッチポイント・サーバーは自動的に生成した値で上書きします。これは、タッチポイント・インスタンス ID の一意性を保証するためです。

宛先の構成

ここでは、アトム文書の処理によるタッチポイントでの宛先の追加について学習できます。

中継タッチポイントとイニシエーター・タッチポイントでは、運用する前に、両方とも、宛先を構成する必要があります。また、両方とも複数の宛先をサポートし、実行時に追加や削除を行う機能を持っています。

構成は、作成したタッチポイント・インスタンスのタッチポイント宛先フィード URL にアトム文書を POST することによって行います。プロバイダー・タッチポイントの場合は、このような URL がないことに注意してください。以下に構成の内容を示します。

```
<entry xmlns="http://www.w3.org/2005/Atom">
 <scmp:data
xmlns:scmp="http://www.ibm.com/xmlns/prod/scmp"
     xsi:schemaLocation="http://www.ibm.com/xmlns/prod/scmp
     http://localhost:1098/tp/schema/touchpoint.xsd" >
  <scmp:destination>
   <scmp:request-out>{request-out_URL}</scmp:request-out>
   <scmp:request-error>{request-error_URL}</scmp:request-error>
  </scmp:destination>
 </scmp:data>
</entry>
```

このスニペットからもわかるように、宛先の構成に必要なのは、「**{request-out_URL}**」のみです。「**{request-error_URL}**」はオプションです。

タッチポイント・インスタンスの通信プロトコル

タッチポイント・インスタンスとの通信に使用されるプロトコルを参照できます。ほとんどの場合、タッチポイント・インスタンスはタッチポイント・テンプレートから派生します。

プロバイダー・タッチポイント

ここで説明されている HTTP メソッドを処理するプロバイダー・タッチポイントを使用できます。

表 33. プロバイダー・タッチポイントの HTTP メソッド

HTTP メソッド	URL 照会パラメーター	コネクター・モード	HTTP 要求の内容	HTTP 応答の内容	HTTP 応答コード
GET	-	イテレーター	-	すべての項目を検出しました	少なくとも 1 つの項目が検出された場合は「200 OK」 検出された項目が 1 つもなかった場合は「404 Not Found」
GET	リンク基準	ルックアップ	-	すべての項目を検出しました	少なくとも 1 つの項目が検出された場合は「200 OK」 検出された項目が 1 つもなかった場合は「404 Not Found」
POST	-	AddOnly	追加される項目	-	操作が正常に終了した場合は「201 Created」
PUT*	リンク基準	更新	更新された属性を持つ項目	-	項目が存在しなかったため新規に追加した場合は「201 Created」 1 つの項目がリンク基準に一致し、その項目の更新が正常に終了した場合は「204 No Content」
DELETE	リンク基準	削除	-	-	操作が正常に終了した場合は「204 No Content」(複数の項目がリンク基準に一致した場合は失敗)

(*) PUT メソッドの処理について、当社のインプリメンテーションと HTTP 1.1 の仕様書 (http://tools.ietf.org/html/rfc2616#section-9.6) では違いのあることにご注意ください。HTTP 仕様書によれば、PUT 要求はリソース全体を置き換えることになっています。当社のインプリメンテーションでは、項目が存在した場合、全体を置き換えるのではなく、指定された属性のみ置き換えます。

コネクター操作のリンク基準は要求された URL の照会パラメーターから派生しています。例えば、URL「http://localhost/mytp?username=jsmith」に対する GET 要求の結果、リンク基準を「username=jsmith」としたルックアップ操作が行われます。各照会パラメーターは、完全一致基準に対応します。複数の照会パラメーターから派

生した基準は、「AND」論理演算子を使用して結合されます。例えば、「?firstname=john&age=50」は ((firstname equals "john") AND (age equals "50")) に対応します。

PUT および DELETE 要求には照会パラメーターが必要です。POST 要求では照会パラメーターが含まれることは想定されていません。GET 要求は、照会パラメーターを含むと、ルックアップ・モードの操作に変換されます。それ以外の場合は、イテレーター・モードの操作に変換されます。

GET 要求の場合、「X-TDI-TP-SizeLimit」と呼ばれるオプションの HTTP ヘッダーを使用して、返す項目の数を制限することができます。ヘッダーの値は、ゼロより大きい整数である必要があります。

デフォルトのタッチポイント・テンプレートで解釈されるすべての HTTP メソッドは、安全性やべき等性などの特性に関してHTTP 仕様に準拠します。

イニシエーター・タッチポイント

イニシエーター・タッチポイントの詳細については、以下の情報を参照してください。

イニシエーター・タッチポイント・インスタンスは HTTP クライアントとして動作します。これは AssemblyLine から構成済みの宛先 URL に対して送信される項目オブジェクトを生成するイテレーター・コネクターを持っています。これは各項目に対して 1 つの POST 要求を送信します。その内容は『項目オブジェクトの HTTP コンテンツとしての表現』です。

中継タッチポイント

複数のタッチポイント・インスタンス間の仲介者として、中継タッチポイントを使用することができます。

中継タッチポイント・インスタンスはプロバイダー役割とイニシエーター役割の両方に似ています。これはプロバイダーとして特定のリクエスト・イン URL から要求を受け取り、イニシエーターとして受信したデータを複数の宛先に送信します。この転送機能のために、それは他の役割からの複数のタッチポイント・インスタンスの間の仲介者として使用することができます。

項目オブジェクトの HTTP コンテンツとしての表現

以下の例を参照して、項目オブジェクトを HTTP コンテンツとして表示することができます。

例:

```
<tp:data xmlns:tp="http://www.ibm.com/xmlns/prod/tdi/72/tp">
 <tp:entry>
  <tp:attribute name="username">
   <tp:value><![CDATA[jsmith]]</tp:value>
  </tp:attribute>

  <tp:attribute name="mail">
   <tp:value>jsmith@ibm.us.com</tp:value>
   <tp:value>john.smith@gmail.com</tp:value>
  </tp:attribute>
 </tp:entry>
</tp:data>
```

HTTP コンテンツは「UTF-8」でエンコードする必要があります。これには、ゼロ個以上の entry エレメントを含む data エレメントが 1 つ必要です。

タッチポイント・データ形式の XML スキーマ記述:

```
<?xml version="1.0" encoding="UTF-8"?>
<schema
 targetNamespace="http://www.ibm.com/xmlns/prod/tdi/72/tp"
 elementFormDefault="qualified"
 attributeFormDefault="unqualified"
 xmlns=http://www.w3.org/2001/XMLSchema xmlns:tns=http://www.ibm.com/xmlns/prod/tdi/72/tp >

 <element name="data" type="tns:TouchpointDataType" />

 <element name="entry" type="tns:EntryType" />

 <element name="attribute" type="tns:AttributeType" />

 <element name="property" type="tns:PropertyType" />

 <element name="value" type="string" />

 <complexType name="TouchpointDataType">
  <choice minOccurs="0" maxOccurs="unbounded">
   <element ref="tns:entry" />
  </choice>
 </complexType>

 <complexType name="EntryType">
  <choice minOccurs="0" maxOccurs="unbounded">
   <element ref="tns:property" />
   <element ref="tns:attribute" />
  </choice>
 </complexType>

 <complexType name="AttributeType">
  <choice minOccurs="0" maxOccurs="unbounded">
   <element ref="tns:property" />
   <element ref="tns:attribute" />
   <element ref="tns:value" />
  </choice>
  <attribute name="name" type="string" use="required" />
  <attribute name="namespaceURI" type="anyURI" />
 </complexType>

 <complexType name="PropertyType">
  <simpleContent>
   <extension base="string">
    <attribute name="name" type="string" use="required" />
    <attribute name="namespaceURI" type="anyURI" />
   </extension>
  </simpleContent>
 </complexType>

</schema>
```

タッチポイント状況項目のスキーマ

タッチポイント・インスタンスの状況は、状況項目の URL に HTTP GET 要求を送信することで取得できます。

359 ページの『タッチポイント・インスタンスの通信プロトコル』.

```
<entry xmlns="http://www.w3.org/2005/Atom">
    <id>Status ID</id>
    <link href="{touchpoint_instance_status_URL}" type="application/atom+xml;type=entry"
          rel="self"/>
    <category scheme="http://www.ibm.com/xmlns/prod/scmp#resource" term="touchpoint"/>
    <category scheme="http://www.ibm.com/xmlns/prod/scmp#aspect" term="status"/>
    <scmp:data
xmlns:scmp ="http://www.ibm.com/xmlns/prod/scmp"
xsi:schemaLocation="http://www.ibm.com/xmlns/prod/scmp
http://localhost:1098/tp/schema/touchpoint.xsd" >
        <scmp:touchpoint-status>
            <scmp:request-in>{request-in_URL}</scmp:request-in>
```

```
            <scmp:op-state>{op-state}</scmp:op-state>
        </scmp:touchpoint-status>
    </scmp:data>
</entry>
```

返されるアトム項目文書には、タッチポイント・インスタンスの状況を説明する **<data>** エレメントが含まれます。このデータ・エレメントは `http://www.ibm.com/xmlns/prod/scmp` 名前空間に属します。これは次の構文を持ちます。

- **{op-state}** - 342 ページの『タッチポイント・インスタンス』での定義に従って、タッチポイント・インスタンスの現在の状況を説明するストリング・キーワードです。
- **{request-in_URL}** - プロバイダーおよび中継タッチポイント・インスタンスにアクセスするための URL です。

プロパティー・シート定義

プロパティー・シート定義は、IBM Security Directory Integrator コネクターのスキーマを決定する XML 文書です。詳しくは、以下の情報を参照してください。

これはタッチポイント型項目内のリンクから取得できます (352 ページの『タッチポイントのスキーマ』を参照)。

プロパティー・シート定義は、タッチポイント型によって異なります。

- 標準型の場合、タッチポイント型に対応する IBM Security Directory Integrator コネクターのスキーマを含みます。
- カスタム型の場合、カスタム・タッチポイント・テンプレート内のサービス・コネクターのスキーマを含みます。
- 仮想型 (`virtual://Intermediary`) の場合、通信するサード・パーティー・システムがないため、プロパティー・シート定義はありません (この役割は HTTP のみに依存しています)。

プロパティー・シート定義には、IBM Security Directory Integrator コネクターのスキーマ・パラメーターの他に、コネクターでサポートされるモードも含まれます。モードは propertyDefinition のオプションの値として `$initMode` という名前で保管されます。それらの値はコネクターのモード名
(「Iterator」、「AddOnly」、「CallReply」など) に直接一致します。

以下にファイル・コネクター (タッチポイント型 `system:/Connectors/ibmdi.LDAP`) のプロパティー・シート定義の例を示します。

```
<?xml version="1.0" encoding="UTF-8"?>
<propertySheetDefinition xmlns="http://www.ibm.com/xmlns/prod/scmp">
    <propertyDefinition required="true" hidden="false" readonly="false"
   propertyType="string" multiple="false"
       propertyName="ldapUrl">
        <label label="LDAP URL" lang="en"/>
        <!--one label for the different languages supported by TDI -->
    </propertyDefinition>
    <!--the rest of LDAP Connector's parameters -->
    <propertyDefinition required="false" readonly="false" propertyType="string"
    multiple="false" propertyName="$initMode">
        <label label="$initMode" lang="en"/>
        <!--one label for the different languages supported by TDI -->
        <option>
            <value>AddOnly</value>
            <label label="AddOnly" lang="en"/>
            <!--one label for the different languages supported by TDI -->
        </option>
        <option>
```

```
            <value>Iterator</value>
            <label label="Iterator" lang="en"/>
            <!--one label for the different languages supported by TDI -->
        </option>
        <!--the rest of modes supported by the LDAP Connector -->
    </propertyDefinition>
</propertySheetDefinition>
```

リストを短くするため、構成パラメーターを 1 つと $initMode のプロパティー定義のみを示しています。ご覧になっておわかりのように、このコネクターには、必須で、値がストリングの **ldapUrl** と呼ばれるパラメーターがあります。また、CE で表示されている英語ラベルは **LDAP URL**です (IBM Security Directory Integrator でサポートされている残りの言語のラベルは省略)。 $initMode パラメーターもあります。そしてオプションの値からわかるように、このコネクターは、Iterator モードと AddOnly モード (それ以外は省略) の両方をサポートしています。

プロパティー・シート定義は、タッチポイント・インスタンスの作成に大きく役立ちます。前もってコネクターのスキーマを知る必要がなくなるからです。プロパティー・シート定義の情報に依存してこの作業を行います。これは IBM Security Directory Integrator の構成エディターを使用してコネクターを構成する作業 (必要なパラメーターを見ながら、期待されている値 (ストリング、数値、または一連の事前定義値) をチェックし、構成に設定する) に似ています。

XML スキーマの場所

ここで提供する情報を通して、XML スキーマの場所を定義できます。

XML スキーマ文書は、各 `scmp:data` エレメントごとに用意されます。この文書は、`scmp:data` エレメント内に現れるすべての要素を定義します。

スキーマ文書の存在する場所は、`scmp:data` エレメントで定義される xsi:schemaLocation (XML スキーマ・インスタンスの場所) 属性で指定されます。例:

```
<scmp:data
 xmlns:scmp="http://www.ibm.com/xmlns/prod/scmp"
 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 xsi:schemaLocation="http://www.ibm.com/xmlns/prod/scmp
http://localhost:1098/tp/schema/tdi-connectivity-provider.xsd">

 <scmp:connectivity-provider>
```

スキーマの場所は、Web クライアントが逆参照できる有効な URL です。タッチポイント・サーバーのクライアントは xsi:schemaLocation 属性に現れる URL を解決して、実際のスキーマ文書にアクセスすることができます。

また、スキーマに別のスキーマ文書への参照 (例えば、XML スキーマ・エレメントの「import」、 「include」、 「redefine」などを使用) が含まれる場合でも、Web クライアントが参照内の URL を解決して、参照されているスキーマを取得することができます。

エラー・フロー

エラーがスローされた場合、該当するエラー・メッセージが標準ログに発行されます。可能性のあるエラー状況と XML 文書構文について詳しくは、以下の情報を参照してください。

エラーが発生する可能性がある状況は、以下のとおりです。

- タッチポイント・サーバーの誤った構成。
- パーシスタンス・ストアとの通信で問題が発生したためスローされた例外 (ファイル・システム・エラー)。
- タッチポイント・サーバーのクライアントとの通信でエラー。
- タッチポイント・サーバーが IBM Security Directory Integrator との通信でエラー。

エラーの原因がユーザーが送信した無効な情報/要求であり、それがタッチポイント・サーバー・プロトコルに違反していた場合は、以下の構文を持つ XML 文書が生成されます。

```
<?xml version="1.0" encoding="UTF-8"?>
<ns2:error xmlns:ns2="http://www.ibm.com/xmlns/prod/scmp">
    <creation-time>2010-02-23T14:49:06.384+02:00</creation-time>
    <code>100005</code>
    <details>
        <detail>
            <name>schema</name>
            <value>http://www.ibm.com/xmlns/prod/scmp#touchpoint-role</value>
        </detail>
    </details>
    <native-msgid>ABCD1234E</native-msgid>
    <summary>Missing role category</summary>
</ns2:error>
```

ここで、

- **creation-time** エレメントは、エラーが発生した時刻を http://www.w3.org/TR/1998/NOTE-datetime-19980827 で規定された形式で示します。
- **code** エレメントは、以下のコードのいずれかです。
  - 100000 - 不明なエラーが発生 (これ以上絞り込むことができません)
  - 100001 – 必須の atom:link が欠落。details エレメントに **rel** (予期したリンクの関係名) が示されます。
  - 100002 - 必須の scmp:data エレメントが欠落。details エレメントに **qname** (欠落している qname エレメント) が示されます。
  - 100003 - POST/PUT 操作に無効な atom:entry (例えば、解析エラー)
  - 100004 - scmp:data エレメントに無効な値 - details エレメントに **qname** (無効な qname エレメント) と **value** (無効な値を指定) が示されます。
  - 100005 - 必須の atom:category が欠落。details エレメントに **scheme** (予期したスキーム名) が示されます。
  - 100006 - 無効な atom:category 値。details エレメントに **scheme** (予期したスキーム名) と **term** (無効な用語を指定) が示されます。
  - 100007 - 指定された関係には多過ぎる atom:link。details エレメントに **rel** (余分なリンクの関係名) が示されます。
  - 100008 – スキームには多過ぎる atom:category 値。details エレメントに **scheme** (値の多過ぎるスキーム名) が示されます。
  - 330000 - デフォルトの接続プロバイダー固有のエラー (これ以上絞り込めません)
- 特定のエラーの場合、**details** エレメントに詳細情報が含まれます。特定のエラー・コードを参照して、各エラーで示される詳細情報を見つけてください。
- **native-msgid** エレメントは、メッセージの短い ID を示します。
- **summary** エレメントには、人間が読めるエラー・メッセージが含まれます。

エラーの原因がプロトコル違反ではなく、異なるソースの場合は、プレーン・テキスト形式で人間が読める表現が返されます。エラーには例外スタック・トレースも含まれるので、タッチポイント・サーバーの管理者にそれを渡して解決を依頼することもできます。

構成

以下の情報を使用して、タッチポイント・サーバーを構成できます。

タッチポイント・サーバーは Web コンテナーの内部で動作します。IBM Security Directory Integrator に付属しているデフォルトの Web コンテナーは、`global.properties` または `solution.properties` 内の以下のプロパティーで構成されます。

- `tp.server.on` - バンドル Web コンテナーとタッチポイント・サーバーを始動するかどうかを指定します。デフォルト値: *false*。
- `tp.server.port` - Web コンテナーが listen するポートを指定します。デフォルト値: 1098。
- `tp.server.auth` - タッチポイント・サーバーが HTTP 基本認証を使用するかどうかを指定します。デフォルト値: *false*。
- `tp.server.auth.realm` - レルム HTTP 基本認証を指定します。デフォルト値: 「IBM Security Directory Integrator タッチポイント・サーバー」。

タッチポイント・サーバーは、最初、`api.remote.bind.address` プロパティーの値を調べます。設定されていなかった場合は、`com.ibm.di.default.bind.address` プロパティーの値を調べます。この方法で、「マルチホーム」ホストへのアクセスを効率良くフィルタリングします。

Web コンテナーは、SSL を使用してトランスポート層のセキュリティーを強化できます。リモート API の設定が再使用されています。`api.remote.ssl` プロパティーを設定することで使用可能にします。SSL クライアント認証は、`api.remote.ssl.client.auth.on` プロパティーで使用可能にします。サーバーの SSL 鍵は、リモート API のよく知られた以下のプロパティーを使用して構成します。

- api.keystore
- api.client.keystore.pass
- api.client.key.pass
- api.client.keystore.type

HTTP 基本認証 (http://tools.ietf.org/html/rfc2617) は、`tp.server.auth` および `tp.server.auth.realm` プロパティーを使用して構成できます。デフォルトでは使用不可になっています。認証について詳細は、367 ページの『認証』を参照してください。

タッチポイント・サーバーの構成は、XML ファイルを使用して指定します。このファイルのパスは、`global.properties` または `solution.properties` ファイル内で、プロパティー `tp.server.config` を使用して指定します。タッチポイント・サーバーの構成ファイルの例は、IBM Security Directory Integrator インストール済み環境の `etc` ディレクトリーに含まれています。

タッチポイント・サーバーの構成ファイルでは、以下の構文が使用されます。

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<tp:tpServerConfig xmlns:tp="http://www.ibm.com/xmlns/prod/tdi/72/tp" tp:version="1.0">

 <!-- specifies the entryption settings used when encrypting passwords -->
 <tp:encryptionConfig stash="idisrv.sth">
  <tp:keyStore>testserver.jks</tp:keyStore>
  <tp:keyStoreType>jks</tp:keyStoreType>
  <tp:keyAlias>server</tp:keyAlias>
  <tp:transformation>RSA</tp:transformation>
 </tp:encryptionConfig>

 <tp:templateConfig>
  <tp:baseTemplate>etc/TouchpointTemplate.xml</tp:baseTemplate>

  <!-- Specify the path to the directory that holds the Touchpoint templates. -->
  <!--
  <tp:customTemplatesDir>templates</tp:customTemplatesDir>
  -->
 </tp:templateConfig>

 <!-- specifies the persistence settings that configure the place to persist the state -->
 <tp:persistenceConfig>
  <tp:enabled>true</tp:enabled>
  <tp:location>tp_state</tp:location>
 </tp:persistenceConfig>

 <!-- configures the touchpuoint providers (nodes) -->
 <tp:nodeConfigs>
  <!-- Default connection to the local server -->
  <tp:tdiNodeConfig tp:local="true" tp:id="default">
   <!-- The host of the remote node which all
     Provider Touchpoint Instances will receive requests on -->
   <tp:providerHost>localhost</tp:providerHost>
   <!-- The port of the remote node which all
     Provider Touchpoint Instances will receive requests on -->
   <tp:providerPort>1097</tp:providerPort>

   <tp:title>Example Touchpoint Provider</tp:title>
   <tp:author>John Doe</tp:author>
   <tp:email>jdoe@example.org</tp:email>
   <tp:summary>Example Touchpoinet Provider Atom Entry</tp:summary>

   <tp:conatct>Local Administrator</tp:conatct>
   <tp:location>Main building, 5th fl.</tp:location>
   <tp:organization>Example Organization</tp:organization>
  </tp:tdiNodeConfig>

  <!-- Here is an example of a remote server connection -->
  <!--
        <tp:tdiNodeConfig id="remote" local="false">
   <tp:title>Example Touchpoint Provider</tp:title>
   <tp:author>John Doe</tp:author>
   <tp:email>jdoe@example.org</tp:email>
   <tp:summary>Example Touchpoinet Provider</tp:summary>

   <tp:host>localhost</tp:host>
   <tp:port>1099</tp:port>
   <tp:user>username</tp:user>
   <tp:password protect="true" encrypted="false">password</tp:password>

   <tp:contact>Jack Smith</tp:contact>
   <tp:location>5th fl.</tp:location>
   <tp:organization>Example Organization</tp:organization>

   <tp:providerHost>locahost</tp:providerHost>
   <tp:providerPort>1097</tp:providerPort>
        </tp:tdiNodeConfig>
  -->
 </tp:nodeConfigs>
</tp:tpServerConfig>
```

認証

以下の情報を参照して、タッチポイント・サーバーの認証に関する側面について詳しく知ることができます。

認証に関連して、タッチポイント・サーバーには 2 つの側面があります。

- HTTP サーバーとして
- IBM Security Directory Integrator サーバーのリモート RMI サーバー API のクライアントとして。これは接続プロバイダーとして構成されます。

タッチポイント・サーバーは、HTTP サーバーとして、HTTP クライアントの HTTP 基本認証をサポートします。独立したユーザー・レジストリーは使用しません。代わりに、タッチポイント・サーバーは認証要求を、ローカル IBM Security Directory Integrator サーバー (タッチポイント・サーバーをホストするサーバー) のサーバー API に委任します。

タッチポイント・サーバーは、リモート・サーバー API クライアントとしては、他のサーバー API クライアントと同様に、リモート IBM Security Directory Integrator サーバーで認証される必要があります。接続プロバイダーがローカル IBM Security Directory Integrator サーバーの場合は認証が不要なことに注意してください。

サーバー API 認証についての詳細は、「リファレンス」の付録『サーバー *API*』を参照してください。

例

ここで提供されるリンクを参照して、付属の例や他の例を使用し、JDBC コネクターを使用してタッチポイント・インスタンスを作成することができます。

付属の例

ここで説明する手順を使用して、タッチポイント・インスタンス作成の付属の例を参照できます。

IBM Security Directory Integrator のタッチポイント・サーバー機能の使用法を示す重要な例であるプロバイダーおよびイニシエーター・タッチポイント・インスタンスの作成手順例は、インストール済み環境に含まれています。作成手順についての詳細は、*TDI_Install_dir*/examples/TouchopintClient/Touchpoint_Example.pdf にあるドキュメントを参照してください。また、Java プログラミング言語でのコーディング例を示すために、これらの手順のサンプル・インプリメンテーションも用意されています。Java コードは次の 2 つのパッケージで構成されます。

- **com.ibm.di.tp.client.api** – このパッケージには、タッチポイント・サーバーとの通信方法を示すコードが含まれます。タッチポイント・サーバーを照会するメソッドのみ含まれています。

  注: このコードは、Apache HttpClient v3.x に依存しています。
- **com.ibm.di.tp.client.gui** – このパッケージには、タッチポイント・インスタンス作成用にタッチポイント・サーバーと対話するため、com.ibm.di.tp.client.api パッケージを使用するサンプル UI クライアントが含まれます。タッチポイント・インスタンスをプロビジョニングする場合、この UI をテスト目的で使用できます。

このユーティリティーは、用意されているスクリプト startClient.bat および startClient.sh を使用して始動できます。

JDBC コネクターを使用してタッチポイント・インスタンスを作成するための手順例

タッチポイント・インスタンスをプロビジョニングする場合は、以下の手順を参照してください。

タッチポイント・インスタンスをプロビジョニングするには、タッチポイント・サーバーに HTTP POST 要求を送信する必要があります。要求用の URL は、アプリケーションの 352 ページの『タッチポイントのスキーマ』に従って取得できます。この例では便宜上、疑似 URL: <Resource Name URL> を使用します。適切な URL は実行時に取得できます。

プロバイダー・タッチポイント・インスタンス

ここで提供される例を使用して、タッチポイント・インスタンス項目リソースを作成することができます。

```
POST <Touchpoint Instance Feed URL>

Body:<entry xmlns="http://www.w3.org/2005/Atom">
 <id>some ID</id>
 <title>Provider Touchpoint Instance</title>
 <author><name>author_name</name></author>
 <content/>
 <category term="provider-tp" scheme="http://www.ibm.com/xmlns/prod/scmp#touchpoint-role" />
 <scmp:data xmlns:scmp="http://www.ibm.com/xmlns/prod/scmp" >
  <scmp:touchpoint>
   <scmp:propertySheet>
    <scmp:property propertyName="jdbcSource">
     <scmp:value>some source value</scmp:value>
    </scmp:property>
    <scmp:property propertyName="jdbcDriver">
     <scmp:value>the driver class</scmp:value>
    </scmp:property>
    <scmp:property propertyName="jdbcTable">
     <scmp:value>the table name</scmp:value>
    </scmp:property>
    <!--The rest of the parameters required by a JDBC Connector-->
   </scmp:propertySheet>
   <scmp:admin-state>enabled</scmp:admin-state>
  </scmp:touchpoint>
 </scmp:data>
</entry>
```

タッチポイント・サーバーからは次のような応答があります。

```
201 Created
Location: <Touchpoint Instance Entry URL>
Body:
<entry xmlns="http://www.w3.org/2005/Atom" xmlns:ns2="http://a9.com/-/spec/opensearch/1.1/"
xmlns:ns3="http://www.w3.org/1999/xhtml">
    <id>generated_id</id>
    <updated>2010-02-17T18:33:55.302+02:00</updated>
    <title>Provider Touchpoint Instance</title>
    <link href="<Touchpoint Instance Entry URL>" type="application/atom+xml;type=entry" rel="self"/>
    <link href="<Touchpoint Instance Entry URL>" type="application/atom+xml;type=entry" rel="edit"/>
    <link href="<Touchpoint Type Entry URL>"
type="application/atom+xml;type=entry" rel="http://www.ibm.com/xmlns/prod/scmp#resource-type"/>
    <link href="<Touchpoint Instance Status Entry URL>" type="application/atom+xml;type=entry"
   rel="http://www.ibm.com/xmlns/prod/scmp#status"/>
    <author><name>author_name</name></author>
    <category scheme="http://www.ibm.com/xmlns/prod/scmp#touchpoint-type" term="system:/Connectors/ibmdi.JDBC"/>
    <category scheme="http://www.ibm.com/xmlns/prod/scmp#touchpoint-role" term="provider-tp"/>
    <category scheme="http://www.ibm.com/xmlns/prod/scmp#resource" term="touchpoint"/>
    <ns2:data xmlns:ns2="http://www.ibm.com/xmlns/prod/scmp">
        <ns2:touchpoint>
            <ns2:admin-state>enabled</ns2:admin-state>
            <ns2:propertySheet>
             <ns2:property propertyName="jdbcSource" xmlns="" xmlns:ns5="http://www.w3.org/2005/Atom">
                <ns2:value>some source value</ns2:value>
```

```
     </ns2:property>
       <ns2:property propertyName=" jdbcDriver " xmlns="" xmlns:ns5="http://www.w3.org/2005/Atom">
      <ns2:value>the driver class</ns2:value>
     </ns2:property>
   <!--The rest of the parameters required by a JDBC Connector-->
            </ns2:propertySheet>
        </ns2:touchpoint>
    </ns2:data>
```

一意性を保証するため、項目の ID がタッチポイント・サーバーにより変更されることに注意してください。

作成したタッチポイント・インスタンスにアクセスするために使用する URL は、状況項目 URL を使用して取得できます。そのために、URL <Touchpoint Instance Status Entry URL> に対して HTTP GET 要求を送信します。受信する応答は次のようになっています。

```
200 OK
Location: <Touchpoint Instance Status Entry URL>
Body:
<?xml version="1.0" encoding="UTF-8"?>
<entry xmlns="http://www.w3.org/2005/Atom" xmlns:ns2="http://a9.com/-/spec/opensearch/1.1/"
xmlns:ns3="http://www.w3.org/1999/xhtml">
    <id>generated_id</id>
    <link href="<Touchpoint Instance Status Entry URL>" type="application/atom+xml;type=entry" rel="self"/>
    <category scheme="http://www.ibm.com/xmlns/prod/scmp#resource" term="touchpoint"/>
    <category scheme="http://www.ibm.com/xmlns/prod/scmp#aspect" term="status"/>
    <ns2:data xmlns:ns2="http://www.ibm.com/xmlns/prod/scmp">
        <ns2:touchpoint-status>
            <ns2:request-in>Touchpoint Provider Request-in URL</ns2:request-in>
            <ns2:op-state>available</ns2:op-state>
        </ns2:touchpoint-status>
    </ns2:data>
</entry>
```

イニシエーター・タッチポイント・インスタンス

ここで提供される例を使用して、イニシエーター・タッチポイント・インスタンスを作成することができます。

```
POST <Touchpoint Instance Feed URL>

Body:
<entry xmlns="http://www.w3.org/2005/Atom">
 <id>some ID</id>
 <title>Initiator Touchpoint Instance</title>
 <author><name>author_name</name></author>
 <content/>
 <category term="initiator-tp" scheme="http://www.ibm.com/xmlns/prod/scmp#touchpoint-role" />
 <scmp:data xmlns:scmp="http://www.ibm.com/xmlns/prod/scmp" >
  <scmp:touchpoint>
   <scmp:propertySheet>
    <scmp:property propertyName="jdbcSource">
     <scmp:value>some source value</scmp:value>
    </scmp:property>
    <scmp:property propertyName="jdbcDriver">
     <scmp:value>the driver class</scmp:value>
    </scmp:property>
    <scmp:property propertyName="jdbcTable">
     <scmp:value>the table name</scmp:value>
    </scmp:property>
    <!--The rest of the parameters required by a JDBC Connector-->
   </scmp:propertySheet>
   <scmp:admin-state>enabled</scmp:admin-state>
  </scmp:touchpoint>
 </scmp:data>
</entry>
```

タッチポイント・サーバーからは次のような応答があります。

```
201 Created
Location: <Touchpoint Instance Entry URL>
Body:
<entry xmlns="http://www.w3.org/2005/Atom" xmlns:ns2="http://a9.com/-/spec/opensearch/1.1/"
xmlns:ns3="http://www.w3.org/1999/xhtml">
    <id>generated_ID</id>
    <updated>2010-02-17T18:33:55.302+02:00</updated>
    <title>Initiator Touchpoint Instance</title>
    <link href="<Touchpoint Instance Entry URL>" type="application/atom+xml;type=entry" rel="self"/>
    <link href="<Touchpoint Instance Entry URL>" type="application/atom+xml;type=entry" rel="edit"/>
    <link href="<Touchpoint Type Entry URL>"
type="application/atom+xml;type=entry" rel="http://www.ibm.com/xmlns/prod/scmp#resource-type"/>
```

```
    <link href="<Touchpoint Instance Status Entry URL>"
  type="application/atom+xml;type=entry" rel="http://www.ibm.com/xmlns/prod/scmp#status"/>
    <link href="<Touchpoint Instance Destination Feed URL>"
  type="application/atom+xml;type=feed" rel="http://www.ibm.com/xmlns/prod/scmp#tp-destination"/>
    <author><name>author_name</name></author>
    <category scheme="http://www.ibm.com/xmlns/prod/scmp#touchpoint-type" term="system:/Connectors/ibmdi.JDBC"/>
    <category scheme="http://www.ibm.com/xmlns/prod/scmp#touchpoint-role" term="initiator-tp"/>
    <category scheme="http://www.ibm.com/xmlns/prod/scmp#resource" term="touchpoint"/>
    <ns2:data xmlns:ns2="http://www.ibm.com/xmlns/prod/scmp">
        <ns2:touchpoint>
            <ns2:admin-state>enabled</ns2:admin-state>
            <ns2:propertySheet>
                 <ns2:property propertyName="jdbcSource" xmlns="" xmlns:ns5="http://www.w3.org/2005/Atom">
                    <ns2:value>some source value</ns2:value>
                 </ns2:property>
                <ns2:property propertyName="jdbcDriver" xmlns="" xmlns:ns5="http://www.w3.org/2005/Atom">
                    <ns2:value>the driver class</ns2:value>
                </ns2:property>
   <!--The rest of the parameters required by a JDBC Connector-->
             </ns2:propertySheet>
        </ns2:touchpoint>
    </ns2:data>
    <content/>
</entry>
```

一意性を保証するため、項目の ID がタッチポイント・サーバーにより変更されることに注意してください。

また、今回、タッチポイント・サーバーの応答には、タッチポイント宛先の構成に必要になるタッチポイント・インスタンス宛先フィード URL が含まれています。

次に、イニシエーター・タッチポイントに宛先を 1 つ追加します。

```
POST <Touchpoint Instance Destination Feed>

Body:
<entry xmlns="http://www.w3.org/2005/Atom">
 <scmp:data xmlns:scmp="http://www.ibm.com/xmlns/prod/scmp" >
  <scmp:destination>
   <scmp:request-out>Request-out URL</scmp:request-out>
  </scmp:destination>
 </scmp:data>
</entry>
```

タッチポイント・サーバーからは次のような応答があります。

```
201 Created
Location: <Touchpoint Instance Destination Entry URL>
Body:
<entry xmlns="http://www.w3.org/2005/Atom" xmlns:ns2="http://a9.com/-/spec/opensearch/1.1/"
xmlns:ns3="http://www.w3.org/1999/xhtml">
    <id>generated_id</id>
    <updated>2010-02-18T10:52:35.108+02:00</updated>
    <link href="<Touchpoint Instance Destination Entry URL>" type="application/atom+xml;type=entry" rel="self"/>
    <link href="<Touchpoint Instance Destination Entry URL>" rel="edit"/>
    <category scheme="http://www.ibm.com/xmlns/prod/scmp#resource" term="tp-destination"/>
    <ns2:data xmlns:ns2="http://www.ibm.com/xmlns/prod/scmp">
        <ns2:destination>
            <ns2:request-out>Request-out URL</ns2:request-out>
        </ns2:destination>
    </ns2:data>
</entry>
```

この時点で、イニシエーター・タッチポイント・インスタンスは実行を開始します。

中継タッチポイント・インスタンス

この役割のタッチポイント・インスタンスの作成に必要な手順は、プロバイダー役割とイニシエーター役割を組み合わせたものです。

初めに、タッチポイント・インスタンス項目リソースを作成します。今回のタッチポイント・インスタンス・フィード URL は、http://<tp_server_host>:<tp_server_port>/<context_root>/tp-node/default/tp-type/virtual___Intermediary/tp-inst と具体的です。

```
POST <Touchpoint Instance Feed URL>

Body:
<entry xmlns="http://www.w3.org/2005/Atom">
 <id>some ID</id>
 <title>Intermediary Touchpoint Instance</title>
 <author><name>author_name</name></author>
 <content/>
 <category term="intermediary-tp"
scheme="http://www.ibm.com/xmlns/prod/scmp#touchpoint-role" />
 <scmp:data xmlns:scmp="http://www.ibm.com/xmlns/prod/scmp" >
  <scmp:touchpoint>
   <scmp:propertySheet>
    <!--No parameters are required-->
   </scmp:propertySheet>
   <scmp:admin-state>enabled</scmp:admin-state>
  </scmp:touchpoint>
 </scmp:data>
</entry>
```

タッチポイント・サーバーからは次のような応答があります。

```
201 Created
Location: <Touchpoint Instance Entry URL>
Body:
<entry xmlns="http://www.w3.org/2005/Atom" xmlns:ns2="http://a9.com/-/spec/opensearch/1.1/"
xmlns:ns3="http://www.w3.org/1999/xhtml">
    <id>generated_ID</id>
    <updated>2010-02-18T11:20:00.546+02:00</updated>
    <title>Intermediary Touchpoint Instance</title>
    <link href="<Touchpoint Instance Entry URL>" type="application/atom+xml;type=entry" rel="self"/>
    <link href="<Touchpoint Instance Entry URL>" type="application/atom+xml;type=entry" rel="edit"/>
    <link href="<Touchpoint Type Entry URL>" type="application/atom+xml;type=entry"
  rel="http://www.ibm.com/xmlns/prod/scmp#resource-type"/>
    <category scheme="http://www.ibm.com/xmlns/prod/scmp#touchpoint-type" term="virtual://Intermediary"/>
    <link href="<Touchpoint Status Entry URL>" type="application/atom+xml;type=entry"
  rel="http://www.ibm.com/xmlns/prod/scmp#status"/>
    <link href="<Touchpoint Destinations Feed URL>" type="application/atom+xml;type=feed"
  rel="http://www.ibm.com/xmlns/prod/scmp#tp-destination"/>
    <author><name>author_name</name></author>
    <category scheme="http://www.ibm.com/xmlns/prod/scmp#touchpoint-role" term="intermediary-tp"/>
    <category scheme="http://www.ibm.com/xmlns/prod/scmp#resource" term="touchpoint"/>
    <ns2:data xmlns:ns2="http://www.ibm.com/xmlns/prod/scmp">
        <ns2:touchpoint>
            <ns2:admin-state>enabled</ns2:admin-state>
            <ns2:propertySheet/>
        </ns2:touchpoint>
    </ns2:data>
    <content/>
</entry>
```

一意性を保証するため、項目の ID がタッチポイント・サーバーにより変更されることに注意してください。

次に、中継タッチポイントに宛先を 1 つ追加します。

```
 POST <Touchpoint Instance Destinations Feed>

Body:
<entry xmlns="http://www.w3.org/2005/Atom">
 <scmp:data xmlns:scmp="http://www.ibm.com/xmlns/prod/scmp" >
  <scmp:destination>
   <scmp:request-out>Request-out URL</scmp:request-out>
  </scmp:destination>
 </scmp:data>
</entry>
```

タッチポイント・サーバーからは次のような応答があるはずです。

```
201 Created
Location: <Touchpoint Instance Destination Entry URL>
Body:
<entry xmlns="http://www.w3.org/2005/Atom" xmlns:ns2="http://a9.com/-/spec/opensearch/1.1/"
xmlns:ns3="http://www.w3.org/1999/xhtml">
    <id>generated_id</id>
    <updated>2010-02-18T10:52:35.108+02:00</updated>
    <link href="<Touchpoint Instance Destination Entry URL>" type="application/atom+xml;type=entry" rel="self"/>
    <link href="<Touchpoint Instance Destination Entry URL>" rel="edit"/>
    <category scheme="http://www.ibm.com/xmlns/prod/scmp#resource" term="tp-destination"/>
    <ns2:data xmlns:ns2="http://www.ibm.com/xmlns/prod/scmp">
        <ns2:destination>
            <ns2:request-out>Request-out URL</ns2:request-out>
        </ns2:destination>
    </ns2:data>
```

最後に、中継タッチポイント・インスタンスに要求を送信するために使用する URL を取得します。プロバイダー・タッチポイントと同様に、状況項目を使用して行います。

タッチポイント・インスタンス項目から取得できる <Touchpoint Instance Status Entry URL> に対して HTTP GET 要求を送信します。

タッチポイント・サーバーからは次のような応答があります。

```
200 OK
Location: <Touchpoint Instance Status Entry URL>
Body:
<?xml version="1.0" encoding="UTF-8"?>
<entry xmlns="http://www.w3.org/2005/Atom" xmlns:ns2="http://a9.com/-/spec/opensearch/1.1/"
xmlns:ns3="http://www.w3.org/1999/xhtml">
    <id>generated_id</id>
    <link href="<Touchpoint Instance Status Entry URL>" type="application/atom+xml;type=entry" rel="self"/>
    <category scheme="http://www.ibm.com/xmlns/prod/scmp#resource" term="touchpoint"/>
    <category scheme="http://www.ibm.com/xmlns/prod/scmp#aspect" term="status"/>
    <ns2:data xmlns:ns2="http://www.ibm.com/xmlns/prod/scmp">
        <ns2:touchpoint-status>
            <ns2:request-in>Touchpoint Intermediary Request-in URL</ns2:request-in>
            <ns2:op-state>available</ns2:op-state>
        </ns2:touchpoint-status>
    </ns2:data>
</entry>
```

第 18 章 トゥームストーン・マネージャー

トゥームストーンを介して終了時のステータスおよびその他の情報を取得できます。その機能の詳細については、以下の情報を参照してください。

IBM Security Directory Integrator では、終了した構成または AssemblyLine を追跡することができます。このため、各ログを確認しなくても、AssemblyLine が最後に実行されたのがいつであるかがわかります。

そのためには、IBM Security Directory Integrator のトゥームストーン・マネージャーを使用します。これにより、各 AssemblyLine および構成の終了時にそれらのトゥームストーンが作成され、終了時には構成も行われます。トゥームストーンには終了状況とサーバー API を通じて要求できるその他の情報が含まれます。トゥームストーン・マネージャーは同時に次の作業も行います。

- AMC 状況ウィンドウに IBM Security Directory Integrator 構成全体の状況を表示します。
- Action Manager 内での AssemblyLine の、例えば、24 時間ごとの反復実行を保証します。
- 非同期に実行された AssemblyLine に関する状況情報をサーバー API クライアントに提供します。

トゥームストーン・マネージャーの API については、Java API 文書を参照してください (クラス `com.ibm.di.api.Tombstone` で検索してください)。

トゥームストーンの構成

トゥームストーンの作成を構成するための必須指定のオプションを選択します。構成ファイルのスイッチ・リストを参照することもできます。

AssemblyLine および構成インスタンスのトゥームストーンの作成は、構成エディター (CE) のいくつかの画面のチェック・ボックスと、`global.properties` ファイルまたは `solution.properties` ファイルのいくつかのオプションを使用して構成されます。

構成が終了すると、構成ファイル内には以下のスイッチが含まれます。

構成レベル:

- 構成スイッチ: 構成インスタンス自身のトゥームストーンを作成するかしないかを指定します。
- すべての AssemblyLines スイッチ: この構成ではすべての AssemblyLine に対してトゥームストーンを作成するかしないかを指定します。

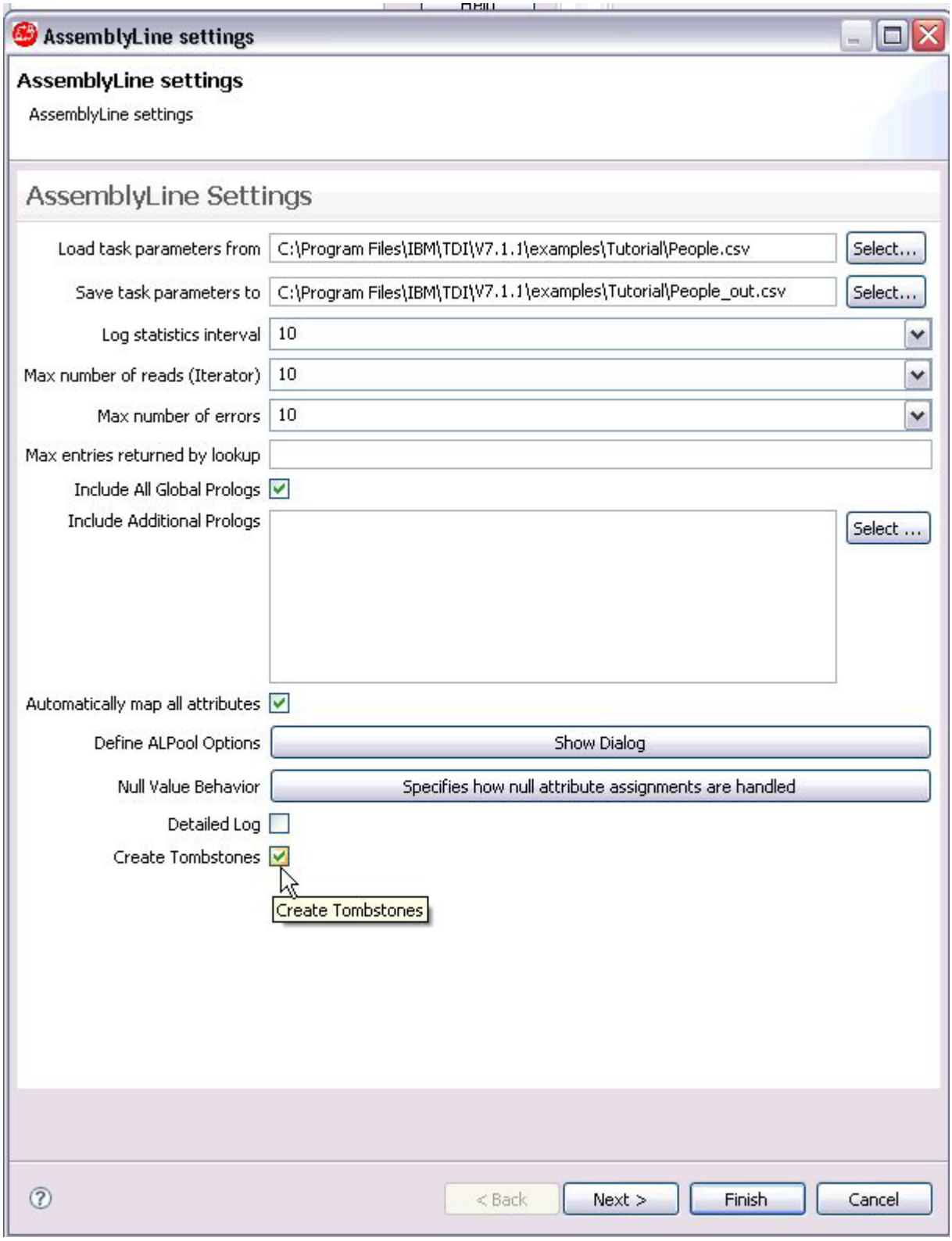
AssemblyLine レベル:

その特定の AssemblyLine についてトゥームストーンを作成するかどうかを指定するスイッチ。このスイッチは、構成レベルで「すべての AssemblyLine スイッチ」がオフになっている場合にのみ有効になります。

「構成エディター」構成画面

表示される 「AssemblyLine 構成」ウィンドウを使用して、AssemblyLine のトゥームストーンの作成を構成することができます。

「トゥームストーンの作成」オプションは、ウィンドウの下部にあります。



「**トゥームストーンの作成**」にチェックを付けると、実行時に、AssemblyLine のマスター・スイッチが使用不可になっていたとしても、この AssemblyLine のトゥームストーンが生成されます。

AssemblyLine の構成画面

トゥームストーンを使用可能にする場合は、トゥームストーン・レコードおよび属性のリストを参照してください。

上記の構成オプションで「AssemblyLine トゥームストーン」を使用不可にしている場合でも、AssemblyLine の構成画面「**トゥームストーンの作成**」の適切なオプションを使用すれば、トゥームストーンの生成は AssemblyLine ごとに独立して使用可能にすることができます。

サンプルのトゥームストーン・レコードは次のようになります。

表 34. トゥームストーン・レコード

フィールド名	**値**
コンポーネント・タイプ ID (Component Type ID)	1
イベント・タイプ ID (Event Type ID)	0
開始時刻	11.11.2005 11:11:54
トゥームストーンの作成時刻	11.11.2005 17:22:45
コンポーネント名	"ActiveDirectoryChangeLogSynchronizer"
構成	"C:¥TDI_SOL_DIR¥rs.xml"
終了コード	0
エラーの説明	""
GUID	"432640786324026346432"
統計	[get:571] [add:571] [err:0]

返される統計は、以下の属性のうち、1 つ以上の属性です。

表 35. トゥームストーン・レコードで返される統計

属性	**説明**
add	AssemblyLine が追加した項目の総数 (コネクターによって AddOnly モードで実行)
mod	AssemblyLine が変更した項目の総数 (コネクターによって更新モードで実行)
del	AssemblyLine が削除した項目の総数 (コネクターによって削除モードで実行)
get	AssemblyLine が検索した項目の総数 (コネクターによってイテレーター・モードで実行)
request	AssemblyLine にサーバー・モード・コネクターがある場合に受け入れられる要求の総数
callReply	AssemblyLine が実行した呼び出し/応答操作の総数 (コネクターによって CallReply モードで実行)
err	検出されたエラーの総数
skip	AssemblyLine がスキップした項目の総数
lookup	AssemblyLine が実行したルックアップ操作の総数 (コネクターによって更新/削除/ルックアップ・モードで実行)
ignore	AssemblyLine が無視した項目の総数 (コネクターによって更新/デルタ・モードで実行)
reconnect	AssemblyLine が別のクライアントへの再接続を試行した回数の総数
exception	AssemblyLine が例外で終了した場合の例外テキスト

表 35. トゥームストーン・レコードで返される統計 (続き)

属性	説明
getTries	AssemblyLine が項目の取得を試行した回数の総数 (コネクターによってイテレーター・モードで実行)
getclientTries	AssemblyLine が次に接続されたクライアントの取得を試行した回数の総数 (コネクターによってサーバー・モードで実行)
nochange	AssemblyLine が処理したが変更されなかった項目の総数
branchtrue	式が true に評価されたために AssemblyLine によって実行されたブランチ・コンポーネントの総数
branchfalse	式が false に評価されたために AssemblyLine によってスキップされたブランチ・コンポーネントの総数
loopstart	AssemblyLine によって実行されたループ・コンポーネントの総数
loopcycles	AssemblyLine 内に複数のサイクルを持つすべてのループ・コンポーネントについて実行されたサイクルの総数
reconnectTime	AssemblyLine が最後に再接続を試行してからの時間 (ミリ秒)

トゥームストーン・マネージャー

トゥームストーン・マネージャーを使用して複数のタスクを実行できます。これは構成プロパティーの値を参照します。詳しくは、以下の情報を参照してください。

トゥームストーン・マネージャーは、実行時にトゥームストーン・レコード数をモニターし、`com.ibm.di.tm.autodel.age`、`com.ibm.di.tm.autodel.records.trigger.on`、`com.ibm.di.tm.autodel.records.max` の構成プロパティーの値に従って古いレコードを削除します。

- トゥームストーン・マネージャーは、構成インスタンスおよび AssemblyLine の停止イベントを追跡します。
- トゥームストーン・マネージャーは、構成インスタンスおよび AssemblyLine のイベント通知の登録と停止イベントの受信でローカル・サーバー API 呼び出しを使用します。
- トゥームストーン・マネージャーは、データの維持のために IBM Security Directory Integrator システム・ストアを使用します。
- サーバー API (Java API 文書を参照) インターフェースには、トゥームストーン・マネージャーに対して、AssemblyLine や構成インスタンスのトゥームストーンなど、さまざまなデータを照会するための呼び出しが含まれます。
- トゥームストーン・マネージャーは、古いトゥームストーン・レコードを削除するためのオプションを提供します。

考えられる AssemblyLine トゥームストーンのライフ・サイクルは、以下のようなものになります。

- トゥームストーン・マネージャーは、AssemblyLine が終了したというサーバー API イベントを受信します (これは、サーバー API およびトゥームストーン・マネージャーがオンになっており、構成ファイルでこの AssemblyLine についてトゥームストーンを作成することが指定されていることが前提となります)。

- トゥームストーン・マネージャーは、サーバー API イベントから必要なデータを抽出し、システム・ストア内に対応するデータベース・トゥームストーン・レコードを作成します。
- トゥームストーン・レコードがシステム・ストア内に存在している間は、サーバー API 呼び出しを使用して照会を実行することにより、トゥームストーン・レコードに含まれるすべての情報を取得することができます。
- トゥームストーン・レコードは、これを削除する明示的なクリーンアップ・サーバー API 呼び出しが実行された場合、または古いトゥームストーン・レコードの自動削除のロジックによりこれが収集された場合に、システム・ストアから削除されます。これらのイベントはいずれも行われることが確実ではないため、理論上トゥームストーン・レコードは永久に存続します。

トゥームストーン・マネージャー

トゥームストーン・マネージャーのタスクは、構成インスタンスの `global.properties` ファイルまたは `solution.properties` ファイル内のプロパティーを使用して構成します。

注: トゥームストーン・マネージャーが機能するには、サーバー API がオンになっている必要があります。すなわち、プロパティー `api.on` を **true** に設定する必要があります。
関連するプロパティーは以下のとおりです。

com.ibm.di.tm.on
トゥームストーン・マネージャーのマスター・スイッチ。値は **on** および **off** です。off に設定すると、構成ファイルで指定されていてもトゥームストーンは生成されません。管理もされません (また、サーバー API や AMC を使用して照会もできません)。

このプロパティーのデフォルト値は **false** です。

com.ibm.di.tm.autodel.age
トゥームストーンが存続する日数。このプロパティーが存在し、0 より大きい整数値が含まれる場合、トゥームストーン・マネージャーは、指定された日数より古いすべてのトゥームストーン・レコードを自動的に削除します。

トゥームストーン・レコード削除のロジックは、IBM Security Directory Integrator サーバーの始動時、および長期間実行されている IBM Security Directory Integrator サーバーの場合は 1 日に 1 回起動されます。

このプロパティーのデフォルト値は **0** です。

com.ibm.di.tm.autodel.records.trigger.on
トゥームストーン・レコード数を特定の数にトリミングするためのロジックをトリガーするトゥームストーン・レコードの総数を指定します。

このプロパティーのデフォルト値は **10000** です。

com.ibm.di.tm.autodel.records.max
前のパラメーター `com.ibm.di.tm.autodel.records.trigger.on` によって指定されたトリガーを超過した場合に保存されるトゥームストーンの数。

このプロパティーのデフォルト値は **5000** です。

com.ibm.di.tm.create.all

このプロパティーは、構成ファイルに指定された値の指定変更スイッチとして機能します。このプロパティーを **true** に設定すると、トゥームストーン・マネージャーは、構成で指定されている値に関係なく、すべての AssemblyLine および構成インスタンスのトゥームストーンを作成します。これは、構成を変更することなく、トゥームストーン値を持たない 6.1 より前の構成についてトゥームストーンの作成をオンにする場合に役立ちます。

`com.ibm.di.tm.autodel.age` プロパティーによって決定される自動クリーンアップのロジックは、`com.ibm.di.tm.autodel.records.trigger.on` および `com.ibm.di.tm.autodel.records.max` プロパティーによって決定される自動クリーンアップのロジックとは独立しています。

トゥームストーン・マネージャーは、IBM Security Directory Integrator ロギング・フレームワークを使用し、そのメッセージを IBM Security Directory Integrator サーバーのメイン・ログに記録します。

`global.properties` ファイルまたは `solution.properties` ファイルのセクションの例を以下に示します。

```
com.ibm.di.tm.on=true
com.ibm.di.tm.autodel.age=90
com.ibm.di.tm.autodel.records.trigger.on=50000
com.ibm.di.tm.autodel.records.max=25000
com.ibm.di.tm.create.all=false
```

この構成プロパティーのセットは、トゥームストーン・マネージャーをオンにすることを指定しています。90 日より古いトゥームストーンは自動的に削除されます。また、トゥームストーン・レコードの総数が 50000 に到達すると、古いものから 25000 個のトゥームストーン・レコードが自動的に削除されます。

第 19 章 複数の IBM Security Directory Integrator サービス

IBM Security Directory Integrator サービスをさまざまなサービスとして登録することができます。詳しくは、以下の情報を参照してください。

Windows サービスとしての IBM Security Directory Integrator

Windows サービスを使用して、さまざまなタスクを実行できます。詳しくは、以下の情報を参照してください。

IBM Security Directory Integrator には、複数の IBM Security Directory Integrator サーバー・インスタンスを Windows サービスとして登録することを可能にするメカニズムがあります。各インスタンスには、個別のソリューション・ディレクトリーが必要です。ソリューション・ディレクトリーを作成したら、ユーティリティー・プログラムをその中にコピーします。プログラムの名前は、`ibmdiservice.exe` です。ユーティリティー・プログラムおよび Windows サービスの構成は、`ibmdiservice.props` という名前のプロパティー・ファイルで行います。各ソリューション・ディレクトリーに構成プロパティー・ファイルが含まれている必要があります。

各 Windows サービスにはそれぞれ異なる名前が必要です。プロパティー・ファイル内の「servicename」というプロパティーは、Windows サービス名および Windows サービス表示名の作成時に使用する名前を指定します。Windows サービス名は、「**servicename**」プロパティーの値に接頭部「ibmdisrv-」を追加することで形成されます。Windows サービス表示名は、「IBM Security Directory Integrator ()」の大括弧内に「servicename」プロパティーの値を挿入することにより形成されます。例えば、「servicename」プロパティーに「test」が設定されている場合、Windows サービス名は「ibmdisrv-test」、また Windows サービス表示名は「IBM Security Directory Integrator (test)」になります。「servicename」プロパティーがない場合または値が指定されていない場合は、デフォルト名が使用されます。Windows サービス名および Windows サービス表示名のデフォルトの名前は、「ibmdisrv」および「IBM Security Directory Integrator」です。

Windows サービスが Windows の開始時に自動的に開始されるようにするか、手動で開始する必要があるようにするかを構成できるプロパティーがあります。このプロパティーの名前は「**autostart**」で、有効な値は「true」および「false」です。

注: このプロパティーは、サービスの実行中や、インストールおよびアンインストール時に使用されます。このため、Windows サービスのインストール後にプロパティー値を変更することはできません。

IBM Security Directory Integrator Windows サービスの構成プロパティー・ファイルについて詳しくは、384 ページの『サービスの構成』セクションを参照してください。

サービスのインストールおよびアンインストール

サービスをインストールおよびアンインストールする場合は、ここにリストされている手順を実行します。

サービスのインストール

ここで示された手順に従って、サービスをインストールすることができます。

このタスクについて

IBM Security Directory Integrator サービスをインストールするには、以下の手順を実行してください。

手順

1. IBM Security Directory Integrator がインストール済みであることを確認します。IBM Security Directory Integrator のインストール・フォルダーを *root_directory* と呼びます。 Windows プラットフォーム用のインストーラーを参照してください。
2. IBM Security Directory Integrator を Windows サービスとして始動したときに使用されるソリューション・フォルダーを選択します (任意のフォルダーを選択できます)。IBM Security Directory Integrator をサービスとして一度インストールした後は、そのサービスによって使用されるソリューション・フォルダーは製品をサービスとしてアンインストールするまでは変更できません。Windows サービスに対して特定のソリューション・フォルダーを選択した場合でも、他のソリューション・フォルダーで IBM Security Directory Integrator が実行できなくなるわけではないことに注意してください。
3. ソリューション・フォルダーを選択した後は、そのフォルダーに `root_directory/win32_service` フォルダーからすべてのファイルをコピーします。コピーするファイルは、ibmdiservice.exe、ibmdiservice.props、および Log4J.properties です。
4. Windows サービスに対して選択したソリューション・フォルダーから以下のコマンドを実行します。`ibmdiservice.exe -i`

サービスのアンインストール

以下に示す手順に従って、サービスをアンインストールすることができます。

このタスクについて

注: 「ibmdiservice.exe」ユーティリティー・プログラムの IBM Security Directory Integrator バージョン 7.2 を使用するには、IBM Security Directory Integrator より前の登録済み Windows サービスをすべてアンインストールしてから、IBM Security Directory Integrator 7.2 の Windows サービスをインストールする必要があります。この作業が必要な理由は、IBM Security Directory Integrator 7.2 Windows サービスでは、これまでと異なる Windows サービス名のデフォルト名「ibmdisrv」が使用されるためです (IBM Security Directory Integrator 7.2 より前は、デフォルト名は「IBM Security Directory Integrator」でした)。

IBM Security Directory Integrator サービスをアンインストールするには、以下の手順を実行してください。

1. IBM Security Directory Integrator サービスが停止していることを確認します。

2. サービスのインストール時に選択したソリューション・フォルダーから以下のコマンドを実行します。

```
ibmdiservice.exe -u
```

注:

1. IBM Security Directory Integrator サービスをアンインストールしても、IBM Security Directory Integrator 自体はアンインストールされません。サービスのアンインストール後も IBM Security Directory Integrator を使用できますが、Windows サービスとして登録および実行されることはありません。IBM Security Directory Integrator サービスを後で再度インストールすることもできます。
2. IBM Security Directory Integrator サービスがインストールされており、IBM Security Directory Integrator を完全に (サービスだけでなく) アンインストールする場合は、以下の手順を実行します。
   a. Windows サービスをアンインストールします。
   b. IBM Security Directory Integrator をアンインストールします (Windows プラットフォーム用のアンインストーラーを参照)。

サービスの開始および停止

リストしたオプションを使用して、サービスを開始または停止します。

IBM Security Directory Integrator サービスは、システム・ブート時に自動的に IBM Security Directory Integrator を始動します。ただし、IBM Security Directory Integrator はサービスのインストール時には自動的に始動されません。サービスのインストール後に、サービスを開始するための以下の 3 つのオプションのいずれかを選択します。

- コンピューターを再始動します。
- 「Windows サービス」ウィンドウで IBM Security Directory Integrator サービスを始動します。
- コマンド行を使用します。388 ページの『コマンド行サポート』を参照してください。

手動による開始および停止

IBM Security Directory Integrator サービスは、「Windows サービス」ウィンドウから手動で開始および停止できます。

「**サービス**」ウィンドウでサービス「IBM Security Directory Integrator」を選択して、Windows のバージョンに応じて「**開始/停止**」ボタンをクリックするか、またはサービス名を右クリックしてから「**開始/停止**」を選択します。

コマンド行を使用することもできます。388 ページの『コマンド行サポート』を参照してください。

サービス開始タイプの変更

デフォルトでは、IBM Security Directory Integrator サービスはシステムのブート時に自動的に開始されるように構成されています。

サービスの開始モードは「Windows サービス」ウィンドウで「**手動**」または「**無効**」に変更できます。

ロギング

IBM Security Directory Integrator サービスはすべてのメッセージ (**エラー**、**情報**、および**デバッグ**) を Application Windows システム・ログに記録します。これらのメッセージは Windows イベント・ビューアーで表示できます。

サービスの構成

ibmdiservice.props ファイルでプロパティーを指定して、IBM Security Directory Integrator サービスを構成することができます。

IBM Security Directory Integrator サービスは、ログ・サービスのインストール時に選択したソリューション・フォルダーにある ibmdiservice.props ファイルによって構成されます。

注: サービスを実行する前に、このファイルがこのセクションの説明どおりに正しく構成されていることを確認してください。このファイルに不正な値が含まれていると、サービスが失敗する場合があります。
ibmdiservice.props ファイルでは、以下のプロパティーが指定されます。

パス IBM Security Directory Integrator のプロセスを実行するために使用される PATH 環境変数を指定します (このプロパティーは通常 ibmdisrv.bat の PATH 変数と同じですが、変更することができます)。このプロパティーはオプションです。

ibmdiroot
IBM Security Directory Integrator のルート・フォルダーを指定します (例えば、C:¥Program Files¥IBM¥TDI¥V7.2)。このプロパティーは必須です。

configfile
IBM Security Directory Integrator 構成ファイルへのファイル・パスを指定します。このプロパティーはオプションです。

assemblylines
IBM Security Directory Integrator サービスの開始時に自動的に開始される AssemblyLine をコンマで区切られた形式で指定します。このプロパティーはオプションです。

cmdoptions
サービス開始時に IBM Security Directory Integrator に直接渡されるその他のオプションを指定します (IBM Security Directory Integrator オプションの完全なリストについては、『 235 ページの『第 13 章 コマンド行オプション』』を参照してください)。

このようなオプションの 1 つが **-c** オプションです。このオプションでは、複数の構成ファイルを (コンマで区切って) 指定できます。これは **configfile** パラメーターでは許可されていません。

この構成を使用する場合の要件は以下のとおりです。

- AssemblyLine ごとに、¥¥ (Windows の場合) または / (UNIX の場合) の構文による絶対パスが必要です。

- 構成ファイルの名前を、コンマで区切って 1 組の引用符で囲む必要があります。
- **-d** オプションは必須です。
- ファイル名にスペースを含めることはできません。

例:

```
cmdoptions=-c"C:/TDI7.1-Solutions/myConfig/Config1.xml ,
     C:/TDI7.1-Solutions/AnotherConfig/TechNotes.xml" -d
```

このプロパティーはオプションです。

servicename

Windows サービス名および Windows サービス表示名の作成時に使用する名前を指定します。Windows サービス名は、**servicename** プロパティーの値に接頭部「ibmdisrv-」を追加することで設定されます。Windows サービス表示名は、「IBM Security Directory Integrator ()」の大括弧内に **servicename** プロパティーの値を挿入することにより作成されます。

例えば、プロパティー値が「test」の場合、Windows サービス名は「ibmdisrv-test」、また Windows サービス表示名は「IBM Security Directory Integrator (test)」になります。**servicename** プロパティーがない場合または値が指定されていない場合は、デフォルト名が使用されます。デフォルトの Windows サービス名は「ibmdisrv」、またデフォルトの Windows サービス表示名は、「IBM Security Directory Integrator」です。

注: このプロパティーは、サービスの実行中や、インストールおよびアンインストール時に使用されます。このため、Windows サービスのインストール後にプロパティー値を変更することはできません。

autostart

Windows サービスを、Windows の始動時に自動的に開始させるか、または手動で開始させるかを指定します。このプロパティーの有効な値は **true** および **false** です。**true** の値は Windows サービスを Windows の始動時に自動的に開始させることを指定し、**false** の値はサービスを手動で開始させる必要があることを指定します。このプロパティーがない場合または値が指定されていない場合は、デフォルト値の **true** が使用されます。

このプロパティーは、Windows サービスをインストールするときに使用されます。Windows サービスをインストールした後で変更しても、効果はありません。

controlledshutdown

Windows サービスでサーバーを安全に停止させるか、またはサーバー・プロセスを強制終了させるか指定します。このプロパティーの有効な値は「true」および「false」です。「true」の値は Windows サービスで IBM Security Directory Integrator サーバーを安全に停止させることを意味し、「false」の値はサーバー・プロセスを強制終了させることを意味します。このプロパティーがない場合または値が指定されていない場合は、デフォルト値の「false」が使用されます。

debug **true** または **false** を指定すると、それに応じてデバッグ情報がオンまたは

オフになります。デバッグ情報がオンになると、詳細トレース・メッセージが Application Windows システム・ログにダンプされます。このプロパティーはオプションです。

注: 構成ファイルでプロパティーを指定する際には、各プロパティーを単一行に指定し、以下の形式を使用してください。

```
<property_name>=<property_value>
```

等号 (=) の前後にスペースを入れないでください。

完成した ibmdiservice.props ファイルの例は以下のようになります。

```
path=C:¥Program Files¥IBM¥TDI¥V7.2¥jvm¥jre¥bin;
C:¥Program Files¥IBM¥TDI¥V7.2¥libs;
ibmdiroot=C:¥Program Files¥IBM¥TDI¥V7.2
configfile=rs.xml
assemblylines=AssemblyLine1,AssemblyLine2
cmdoptions=
debug=false
controlledshutdown=false
```

注: ibmdiservice.props でプロパティーを変更した場合は、変更を有効にするためにサービスを再始動してください。

Linux/UNIX サービスとしての IBM Security Directory Integrator

Linux/UNIX サービスを使用して、さまざまなタスクを実行できます。詳しくは、以下の情報を参照してください。

デプロイメント・メソッド

Linux および UNIX プラットフォームでは、システムの開始時および終了時にそれぞれ特定のシステム・ジョブ (または「デーモン」) を開始および停止させる方法が 2 つあります。

1. 目的のデーモンを開始および停止するロジックを含む */etc/init.d* 内のスクリプトを使用します。このスクリプトを */etc/rc3.d* 内のスクリプトに (ハード) リンクさせます。これらの名前の先頭は SXX... および KXX... になっています。 XX は、/etc/rc3.d ディレクトリー内でファイルを正しい順序で表示させるための数字です。先頭文字が S のスクリプトは、システムが始動時に実行フェーズ 3 に到達したときに呼び出され、先頭文字が K のスクリプトは、システムの終了時に呼び出されます。
2. */etc/inittab* ファイルを編集します。

ここでは、後者のプロセスについて説明します。一部の情報は、前者のデプロイメント方法を使用してスクリプトを構成する場合にも使用できます。

/etc/inittab の調整

UNIX/Linux OS の始動時に IBM Security Directory Integrator デーモン・プロセスを開始させるには、 */etc/inittab* ファイルに適切な項目を追加する必要があります。Windows における Windows サービスとしての IBM Security Directory Integrator の登録は、 UNIX/Linux においては */etc/inittab* ファイルへの 1 行のテキストの追加に形を変えます。Windows における IBM Security Directory Integrator Windows サービスのアンインストールは、 */etc/inittab* ファイルからの対応する項目の除去に形

を変えます。システムの始動時に開始する必要のある各 IBM Security Directory Integrator デーモン・プロセスについて、 */etc/inittab* ファイルに 1 行のテキストを追加する必要があります。このファイル内の項目の形式および意味について以下に説明します。*/etc/inittab* ファイル内の各項目は、以下のような形式になっています。

Identifier:RunLevel:Action:Command

これらの各フィールドについて、以下に説明します。

- **Identifier** フィールドは、オブジェクトを一意に識別するストリング (長さは最短で 1 文字) です。対応するコマンドを一意的に識別するために、このストリングが使用されます。
- **RunLevel** フィールドは、この項目を処理できる実行レベルです。実行レベルは、システム内のプロセスの構成に効果的に対応します。init コマンドで開始される各プロセスには、それが存在可能な 1 つ以上の実行レベルが割り当てられます。実行レベルは、0 から N の数字で表します。N は、UNIX/Linux オペレーティング・システムの種類に応じて異なる正整数が使用されます (例えば、一部の AIX コンピューターでは N は 9、RedHat Linux では N は 6 などとなっています)。例えば、OS が実行レベル 3 で実行されている場合は、実行レベル 3 に指定されたプロセスのみが開始されます。

  RunLevel フィールドでは、0 から N のうち任意の組み合わせで複数の実行レベルを選択することにより、1 つのプロセスに複数の実行レベルを定義できます。例えば、IBM Security Directory Integrator を実行レベル 3 および 6 で実行する必要がある場合、実行レベルを「36」として指定する必要があります。実行レベルを指定しないと、そのプロセスはすべての実行レベルで有効であると見なされます。

  特定の IBM Security Directory Integrator ソリューションについて指定する必要がある場合を除き、実行レベル番号を指定しないことを推奨します。
- **Action** フィールドは、**init** コマンドに対して、 **Command** フィールドに指定されたプロセスの処理方法を通知する事前定義アクション・セットからの値です。init コマンドが認識するアクションは数多くありますが、IBM Security Directory Integrator サーバーをデーモン・プロセスとして実行するには、**once** アクションを使用することを推奨します。**once** アクションの意味は以下のとおりです。

  init コマンドは、項目の実行レベルに一致する実行レベルに入ると、プロセスを開始し、その終了を待機しません。これが停止しても、プロセスを再開しません。システムが新しい実行レベルに移行しても、プロセスが前の実行レベルの変更からまだ実行中である場合、プログラムは再開されません。init コマンドが同じ実行レベルにいる限り `/etc/inittab` ファイルのその後のすべての読み取りにおいて、init コマンドはこの項目を無視します。
- **Command** フィールドには、実行するシェル・コマンドを指定します。

以下に、`/etc/inittab` 内の IBM Security Directory Integrator 関連項目の 3 つの例を示します。

```
tdi1::once:/opt/IBM/TDI711_1/ibmdisrv -c "/opt/IBM/TDI711_1/myconfigs/rs1.xml" -r "testAL1"
tdi2::once:/opt/IBM/TDI711_2/ibmdisrv -c "/opt/IBM/TDI711_2/myconfigs/rs2.xml" -r "testAL2"
tdi3::once:/opt/IBM/TDI711_3/ibmdisrv -c "/opt/IBM/TDI711_3/myconfigs/rs3.xml" -r "testAL3"
```

この例では、別々のフォルダーにインストールされている 3 つの IBM Security Directory Integrator サーバー・インスタンスが開始されます。

注: UNIX/Linux オペレーティング・システムが異なる場合、システムの始動に関していくつかの相違点があります。このため、ここでは、UNIX/Linux システムでの IBM Security Directory Integrator の開始に関する主要な問題についての情報のみを提供し、特定の UNIX/Linux システムには言及しません。

`/etc/inittab` ファイルの例として、AIX システムの `/etc/inittab` 構成ファイルに関する詳細な情報が、http://publib.boulder.ibm.com/infocenter/pseries/v5r3/index.jsp?topic=/com.ibm.aix.files/doc/aixfiles/inittab.htm にあります。

安全なシャットダウン

UNIX システムでは、常に、サービスは安全なシャットダウンを実行します。これは Java ランタイムに追加されたシャットダウン・フックを利用して行います。この方法を使用すれば、サーバーが SIGINT シグナルまたは SIGTERM シグナルで停止したとき、フックの実行によって、サーバーを安全に停止させることができます。

このアプローチのもう一つの利点は、すべてのプラットフォームで、コンソール・ウィンドウで CTRL+C を押してサーバーを停止させたとき、このフックを起動できることです。

注: ユーザーは、JVM シャットダウン・フックから始動させる外部プログラムを指定することもできます。この外部プログラムは、`global.properties` ファイルまたは `solution.properties` ファイル内のオプション・プロパティー `jvm.shutdown.hook` を使用して構成します。構成した場合は、サーバーを安全にシャットダウンした直後に外部プログラムが始動します。

コマンド行サポート

以下の情報を使用して、サービスを開始および停止するためのスクリプトを使用できます。

IBM Security Directory Integrator には、Windows および UNIX システムで IBM Security Directory Integrator サービスを開始および停止するためのスクリプトが用意されています。このスクリプトは *TDI_install_dir*`/bin` ディレクトリーにあり、名前は servicemgr.bat(sh) です。

このスクリプトの使用法は次のとおりです。

```
servicemgr service_name start|stop
```

説明:

service_name
: サービスの名前です。Windows システムでは、デフォルトの `ibmdisrv` の値、または `ibmdiservice.props` ファイル内の `servicename` プロパティーの値です。UNIX システムでは、`/etc/inittab` ファイルの ID フィールドの値です。

start|stop
: サービスに対して実行するコマンドです。

付録 A. プロパティー・ファイルの例

IBM SDI のインストールに関するプロパティー・ファイルをカスタマイズすることができます。使用可能なさまざまなテキスト・ファイルとソリューション・ディレクトリーの詳細については、以下の情報を参照してください。

インストールした IBM Security Directory Integrator は、1 つ以上の**プロパティー**が入ったテキスト・ファイルのセットを使用することで大幅にカスタマイズできます。プロパティーは、通常、キーワード (識別子) の後に値を続けた形式になっています。 IBM Security Directory Integrator インストール・ディレクトリーの `root/etc` レベルに以下のグローバル・プロパティー・テキスト・ファイルがあります。

- 390 ページの『Log4J.properties』
- 391 ページの『jlog.properties』
- 392 ページの『derby.properties』
- 393 ページの『global.properties』

これらのファイル内のプロパティー・セットは、 IBM Security Directory Integrator インストール済み環境全体でそのコンピューターのすべてのユーザーのベースラインを形成します。ただし、ソリューション・ディレクトリーがインストール・ディレクトリーとは別に設定されている場合は、インストール・ディレクトリーにあるのと同じテキスト・ファイルのセットをソリューション・ディレクトリーに配置できます。これらのファイルにリストされているプロパティーは、上述のグローバル・インストール・プロパティー・ファイルに設定されているプロパティーをオーバーライドします。また、構成ファイル内で設定する Java プロパティーの優先順位が一番上で、グローバル・プロパティー・ファイルでの設定や、ソリューション・ディレクトリー内のプロパティー・ファイルの設定より優先されます。

ソリューション・ディレクトリーを指定する方法はいくつかあります。

- 構成エディターまたはサーバーを始動する前に、環境変数 *TDI_SOLDIR* を設定する
- `ibmditk` スクリプトに **-s** パラメーターを指定してサーバーを始動する。この指定は、*TDI_SOLDIR* の設定より優先されます。

TDI_SOLDIR がインストール・ディレクトリーと等しい場合の動作は、IBM Security Directory Integrator の以前のバージョンと同様になります。つまり、プロパティー・ファイルはすべてインストール・ディレクトリーから読み取られ、ソリューション・ディレクトリーにあるプロパティー・ファイルについての注釈は適用されません。

それ以外の場合は、IBM Security Directory Integrator サーバーを初めて実行すると、ソリューション・ディレクトリーにすべてのプロパティー・ファイルのコピーが作成されます (これらのファイルが既に存在する場合は上書きされません)。その後は、インストール・ディレクトリーにあるプロパティー・ファイルに影響することなく、必要に応じてそれらのファイルを調整できます。インストール・ディレク

トリーに残っているファイルは、引き続き他の IBM Security Directory Integrator インスタンスのベースライン構成となります。

注: global.properties ファイルは、ソリューション・ディレクトリーには solutions.properties という名前のファイルとしてコピーされます。Log4J.properties ファイルや、amc および serverapi フォルダー内のファイルなど、その他のファイルは元の名前のままコピーされます。

また、IBM Security Directory Integrator インストーラーを使用した製品のインストール中にソリューション・ディレクトリーをセットアップしていた場合、セットアップには作業用のシステム・キューのセットアップも含まれます。ソリューション・ディレクトリーを別の手段 (手動、または -s オプションを指定してサーバーで行う) で作成する場合、solution.properties ファイル内のシステム・キューを使用不可にするか、または専用のシステム・キューをセットアップする必要があります (177 ページの『システム・キューの構成』を参照)。

Log4J.properties

このファイルはサーバー (ibmdisrv) 用の基本的なログ方式を設定します。

構成エディターの「ロギング」タブで構成したログ・オプションは構成ファイルに書き込まれ、以下の内容を補足または置換します。

```
# This file controls the logging strategy for the server (ibmdisrv) when started
# from the command line.
# Look at executetask.properties for the logging strategy of the server when started
# from the Configuration Editor (ibmditk).
# Look at ce-log4j.properties for the logging behavior of the Configuration Editor (ibmditk).
#
# You will normally configure the logging strategy of the server by adding appenders
# using the Configuration Editor (ibmditk). This file only defines the baseline
# that is independent of the configuration files you are using.
#
# See the IDI documentation for more information on the contents of this file.
#


log4j.rootCategory=INFO, Default

# This is the default logger, you will see that it logs to ibmdi.log
log4j.appender.Default=org.apache.log4j.FileAppender
log4j.appender.Default.file=logs/ibmdi.log
log4j.appender.Default.layout=org.apache.log4j.PatternLayout
log4j.appender.Default.layout.ConversionPattern=%d{ISO8601} %-5p [%c] - %m%n
log4j.appender.Default.append=false

#Example settings for changing the default logger

##########ROLLING FILE SIZE APPENDER
##RollingFileAppender rolls over log files when they reach a certain size specified by the
##MaxFileSize parameter

#log4j.appender.Default=org.apache.log4j.RollingFileAppender
#log4j.appender.Default.File=logs/ibmdi.log
#log4j.appender.Default.Append=true
#log4j.appender.Default.MaxFileSize=10MB
#log4j.appender.Default.MaxBackupIndex=10
#log4j.appender.Default.layout=org.apache.log4j.PatternLayout
#log4j.appender.Default.layout.ConversionPattern=%d{ISO8601} %-5p [%c] - %m%n

##########DAILY OUTPUT LOG4J SETTINGS
## With the DailyRollingFileAppender the underlying file is rolled over at a user chosen frequency.
##The rolling schedule is specified by the DatePattern option

#log4j.appender.Default=org.apache.log4j.DailyRollingFileAppender
#log4j.appender.Default.file=logs/ibmdi.log
#log4j.appender.Default.DatePattern='.'yyyy-MM-dd
#log4j.appender.Default.layout=org.apache.log4j.PatternLayout
#log4j.appender.Default.layout.ConversionPattern=%d{ISO8601} %-5p [%c] - %m%n

# You may change the logging category of these subsystems to DEBUG
# if you want to investigate particular problems. This may
# generate a lot of output.
# ...com.ibm.di.config describes the loading of the configuration file (.xml),
```

```
#    and how the internal configuration structure is built.
# ...com.ibm.di.loader gives information about jar files, and where classes are found.
#    It also loads idi.inf files, which provides Connectors/Parsers/EH information
#    for the Configuration Editor.
log4j.logger.com.ibm.di.config=WARN
log4j.logger.com.ibm.di.loader=WARN

# Uncomment the lines below to activate them

# Here is an example on how to make a logger that logs to the console
#log4j.appender.CONSOLE=org.apache.log4j.ConsoleAppender
#log4j.appender.CONSOLE.layout=org.apache.log4j.PatternLayout
#log4j.appender.CONSOLE.layout.ConversionPattern=%d [%t] %-5p - %m%n0

# Here is an example that logs to myFile.log
#log4j.appender.fileLOG=org.apache.log4j.FileAppender
#log4j.appender.fileLOG.file=myFILE.log
#log4j.appender.fileLOG.layout=org.apache.log4j.PatternLayout
#log4j.appender.fileLOG.layout.ConversionPattern=%d{ISO8601} %-5p [%c] - %m%n
#log4j.appender.fileLOG.append=false


# Finally, make use of the loggers defined above:
# Tell AssemblyLines myAL to log using CONSOLE logger defined above.

# log4j.logger.AssemblyLine.AssemblyLines/myAL=INFO, CONSOLE

# Or you could log to myFile.log

# log4j.logger.AssemblyLine.AssemblyLines/myAL=INFO, fileLOG
```

jlog.properties

IBM SDI サーバーの JLOG ベースの値を構成および変更することができます。

このファイルは、IBM Security Directory Integrator サーバーの JLOG ベースの 265 ページの『第 15 章 トレースと FFDC』 を構成します。サーバーの始動時にプロパティー `jlog.noLogCmd` が **false** に設定されていた場合は、LogCmd スクリプトを使用してこれらの値を動的に (サーバーの実行中に) 変更することができます。

注: ソリューションにおける実行フローをトレースするには、通常は Log4J を使用します。JLOG ベースのトレースおよび FFDC は、IBM Security Directory Integrator に問題が起きた場合に IBM サポート担当者を支援することを目的としています。

```
##############################################################################################################
# This file controls the tracing and First Failure Data Capture (FFDC) strategy for ITDI 7.2
# See the IDI documentation for more information on the contents of this file.
##############################################################################################################

#-------------------------------------------------------------------------------------------------------------
# Enable the JLOG's command server
#
#  If the jlog.noLogCmd is set to false, then the JLOG LogManager will listen on the
#  default port (9992) for JLOG log commands.
#  Setting this property to false will enable you to modify the JLOG properties dynamically using the
# logcmd scripts. The logcmd scripts are placed under ITDI_HOME directory.
#  The default value is set to true.
#-------------------------------------------------------------------------------------------------------------
jlog.noLogCmd=true

#-------------------------------------------------------------------------------------------------------------
# Set listen port for JLOG's command server
#
#       If you want LogManager to listen on different port than the default one (9992) you should
#       uncomment the property jlog.logCmdPort and set it to the desired port. If not uncommented
#       the LogManager will listen on the default port - 9992.
#-------------------------------------------------------------------------------------------------------------
#jlog.logCmdPort=9992

#-------------------------------------------------------------------------------------------------------------
# Configure Jlog FileHandler for tracing into a file.
#
#  By default the FileHandler is not attached to the Jlog Logger.
# Uncomment the properties with the prefix jlog.filehandler below to configure a FileHandler.
# After uncommenting this you need to add the filehandler to the logger's listeners names as shown
# below
# e.g: jlog.logger.listenerNames=jlog.snapmemory jlog.snaphandler jlog.filehandler
#-------------------------------------------------------------------------------------------------------------
jlog.filehandler.className=com.ibm.log.FileHandler
```

```
#jlog.filehandler.description=JLOG File Handler for Logging and Tracing
#jlog.filehandler.encoding=UTF8
#jlog.filehandler.maxFiles=10
#jlog.filehandler.maxFileSize=2048
#jlog.filehandler.appending=true
#jlog.filehandler.fileDir=logs/
#jlog.filehandler.trace.fileName=trace.log
#-----------------------------------------------------------------------------------------------------------------


#-----------------------------------------------------------------------------------------------------------------
# create a level filter.
# The level filter is used to define the level at which JFFDC action will be triggered.
# For JFFDC to be meaningful this should be set to either FATAL or ERROR (case-insensitive).
# NOTE: Setting the trigger level to other levels such as DEBUG_MIN will trigger unwanted JFFDC
#  action causing a performance drop.
#-----------------------------------------------------------------------------------------------------------------
jlog.levelflt.className=com.ibm.log.LevelFilter
jlog.levelflt.level=FATAL


#-----------------------------------------------------------------------------------------------------------------
# Configure the SnapMemoryHandler for tracing into a memory buffer.
# The SnapMemoryHandler traces into a memory buffer and dumps the contents of the memory to a file on
# trigger of a event (as defined by the level filter above) and writes the content to the specified
# file
# Properties:
# jlog.snapmemory.queueCapacity : Sets the nnumber of LogEvents that can be buffered in the memory
# jlog.snapmemory.snapFile  : name of the file to which the contents of the memory will be dumped
# jlog.snapmemory.baseDir  : The directory where the snapFile is placed.
#        daily subdirectories will be created under this base directory, as:
#         [baseDir]/[YYYY-MM-DD]/
#         Note: MS-DOS style path names need to be be escaped with backslashes
#         eg: c:¥¥CTGI¥¥FFDC
# jlog.snapmemory.userSnapFile : The name of the file to which the user initiated (from logcmd) dumps
#        will be written to.
# jlog.snapmemory.userSnapDir  : The directory where the userSnapfile is placed.
# jlog.snapmemory.msgIds  : The list of TMS IDs
# jlog.snapmemory.msgIDRepeatTime   : The minimum time, in milliseconds, after passing a log event with a
#        given TMS message id, before another log event with the same id can
#        be passed.
#-----------------------------------------------------------------------------------------------------------------
jlog.snapmemory.className=com.tivoli.log.SnapMemoryHandler
jlog.snapmemory.description=Memory handler used to trace to memory
jlog.snapmemory.queueCapacity=10000
jlog.snapmemory.dumpEvents=true
jlog.snapmemory.snapFile=trace.log
jlog.snapmemory.baseDir=CTGDI/FFDC/
jlog.snapmemory.userSnapFile=userTrace.log
jlog.snapmemory.userSnapDir=CTGDI/FFDC/user/
jlog.snapmemory.triggerFilter=jlog.levelflt
jlog.snapmemory.msgIds=*E
jlog.snapmemory.msgIDRepeatTime=10000

#-----------------------------------------------------------------------------------------------------------------
# Configure the JLogSnapHandler taking a snapshot of the SnapMemoryHanlders buffer
# The JLogSnapHanlder takes a snapshot of the associated SnapMemoryBuffer.
#-----------------------------------------------------------------------------------------------------------------
jlog.snaphandler.className=com.tivoli.log.JLogSnapHandler
jlog.snaphandler.description=snaphandler to dump the memory trace
jlog.snaphandler.baseDir=CTGDI/FFDC/
jlog.snaphandler.snapMemoryHandler=jlog.snapmemory
jlog.snaphandler.triggerFilter=jlog.levelflt

#-----------------------------------------------------------------------------------------------------------------
# Configure the PDLogger (Problem Determination) Object and attach the Listeners to it.
# jlog.logger.level can be FATAL | ERROR | WARNING | INFO | DEBUG_MIN | DEBUG_MID | DEBUG_MAX
# The heirarchy of the log levels is from the most severe (FATAL) to the least severe (DEBUG_MAX)
# The value for this property is case-insensitive
#-----------------------------------------------------------------------------------------------------------------
jlog.logger.level=FATAL
#jlog.logger.listenerNames=jlog.snapmemory jlog.snaphandler
jlog.logger.listenerNames=jlog.filehandler.trace
jlog.logger.className=com.ibm.log.PDLogger


#-----------------------------------------------------------------------------------------------------------------
# Configure the PDLogger for the Config Editor and attach the Listeners to it.
# By default, no listeners are attached
#-----------------------------------------------------------------------------------------------------------------
jlog.logger.config-editor.level=FATAL
jlog.logger.config-editor.listenerNames=
```

derby.properties

このファイルには、ネットワーク・モードの Derby の一部のデフォルト値が含まれています。

IBM Security Directory Integrator 関連の Derby パラメーターの大部分は、このファイルではなく、global.properties および solution.properties ファイルで保守されます。これらのパラメーターの詳細については、Derby の文書を参照してください。

```
# This is a sample properties file provided to show the proper format.
# We're also setting one property which make sure that
# Derby adds to the error log instead of overwriting it.
# This mode is useful for development.
derby.drda.logConnections=true
derby.drda.maxThreads=0
derby.drda.portNumber=1527
derby.drda.traceAll=true
derby.drda.timeSlice=0
derby.drda.traceDirectory=/trace
```

global.properties

このファイルは始動時に ibmditk (CE) と ibmdisrv (サーバー) によって読み取られます。

このファイルが読み取られ、適用された後に、ソリューション・ディレクトリーから solution.properties という名前のファイルが読み取られ、適用されます。

注:
行が極端に長いことが原因のここでのレンディションは、完了しない場合があります。代わりに実際の global.properties ファイルを参照してください。

```
##
## This file is read by ibmditk/ibmdisrv on startup
##
## Enter <name>=<value> to set system properties.
## Enter !include <file | url> to include other files
##

com.ibm.di.securityTransformation=DES/ECB/NoPadding

##
## Modify the line below to add your own jar/zip files.
## The property may specify several directories or jar files, separated by the Java Property "path.separator",
## which is ":" on Linux and ";" on Windows
## Directories will be searched recursively by the TDILoader for jar files containing classes and resources.
## Only files with a ".zip" or ".jar" extension are searched.
# com.ibm.di.loader.userjars=c:¥myjars

##
## Modify the line below to enable the config autoload feature.
## When this property is defined, the "ibmdisrv -d" command
## line will look for *.xml files in the directory specified by this property and start each one.
##
# com.ibm.di.server.autoload=autoload.tdi

##
## SYSTEM STORE
##

## Location of the database (embedded mode) - Cloudscape 10
#com.ibm.di.store.database=TDISysStore
#com.ibm.di.store.jdbc.driver=org.apache.derby.jdbc.EmbeddedDriver
#com.ibm.di.store.jdbc.urlprefix=jdbc:derby:
#com.ibm.di.store.jdbc.user=APP
#{protect}-com.ibm.di.store.jdbc.password=APP


## Location of the database to connect (networked mode) - Cloudscape 10 - DerbyClient driver
## The macro $soldir$ will be replaced by the value of the actual Solution Directory
com.ibm.di.store.database=jdbc:derby://localhost:1527/$soldir$/TDISysStore;create=true
com.ibm.di.store.jdbc.driver=org.apache.derby.jdbc.ClientDriver
com.ibm.di.store.jdbc.urlprefix=jdbc:derby://localhost:1527/
com.ibm.di.store.jdbc.user=APP
{protect}-com.ibm.di.store.jdbc.password={encr}n+Vum7tN0ZU0KNp7AGy7pkAZqiJMGgPnqwg
     /dBhLEL5pDBj5FY/Qp/2OmOkfWDezdSvYGUKag3UkzV+NuSSBVpJ36s3QCkFDz72VOTzJIa1REhIp/j/u9
     /3E1lZPIAHlB1gKP772OOFPJIB6mbDUUugFwIZ+FmKFH5CW6Nytp+M=

#
## Derby (Cloudscape) properties required for enabling authentication
#
derby.drda.startNetworkServer=true
derby.connection.requireAuthentication=true
derby.authentication.provider=BUILTIN
derby.database.defaultConnectionMode=fullAccess

#
## Details for starting Cloudscape in network mode.
## Note: If the com.ibm.di.store.hostname is set to localhost then remote connections will not be allowed.
## If it is set to the IP address of the local machine - then remote clients can access this Cloudscape
```

```
## instance by mentioning the IP address. The network server can only be started for the local machine.
#
#com.ibm.di.store.start.mode=automatic
com.ibm.di.store.hostname=localhost
com.ibm.di.store.port=1527
com.ibm.di.store.sysibm=true


# the varchar(length) for the ID columns used in system store and pes connector tables
com.ibm.di.store.varchar.length=512


## create statements for system store tables (CloudScape 5.1)
#com.ibm.di.store.create.delta.systable=CREATE TABLE {0}
     (ID VARCHAR(VARCHAR_LENGTH) NOT NULL, SEQUENCEID int, VERSION int)
#com.ibm.di.store.create.delta.store=CREATE TABLE {0}
     (ID VARCHAR(VARCHAR_LENGTH) NOT NULL, SEQUENCEID int, ENTRY long varbinary )
#com.ibm.di.store.create.property.store=CREATE TABLE {0}
     (ID VARCHAR(VARCHAR_LENGTH) NOT NULL, ENTRY long varbinary )
#com.ibm.di.store.create.sandbox.store=CREATE TABLE {0}
     (ID VARCHAR(VARCHAR_LENGTH) NOT NULL, ENTRY long varbinary )

## create statements for system store tables (CloudScape 10)
com.ibm.di.store.create.delta.systable=CREATE TABLE {0}
     (ID VARCHAR(VARCHAR_LENGTH) NOT NULL, SEQUENCEID int, VERSION int);
     ALTER TABLE {0} ADD CONSTRAINT IDI_CS_{UNIQUE} PRIMARY KEY (ID)
com.ibm.di.store.create.delta.store=CREATE TABLE {0}
     (ID VARCHAR(VARCHAR_LENGTH) NOT NULL, SEQUENCEID int, ENTRY BLOB );
     ALTER TABLE {0} ADD CONSTRAINT IDI_DS_{UNIQUE} Primary Key (ID)
com.ibm.di.store.create.property.store=CREATE TABLE {0}
     (ID VARCHAR(VARCHAR_LENGTH) NOT NULL, ENTRY BLOB );
     ALTER TABLE {0} ADD CONSTRAINT IDI_PS_{UNIQUE} Primary Key (ID)
com.ibm.di.store.create.sandbox.store=CREATE TABLE {0}
     (ID VARCHAR(VARCHAR_LENGTH) NOT NULL, ENTRY BLOB )
com.ibm.di.store.create.recal.conops=CREATE TABLE {0} (METHOD varchar(VARCHAR_LENGTH), RESULT BLOB, ERROR BLOB)


## create statements for system store tables DB2 on z/OS
#com.ibm.di.store.create.delta.systable=CREATE TABLESPACE TS1DSYS LOCKSIZE ROW BUFFERPOOL BP32K;CREATE TABLE {0}
     (ID VARCHAR(VARCHAR_LENGTH) NOT NULL, SEQUENCEID int, VERSION int) IN TS1DSYS;CREATE UNIQUE INDEX DSTIX1 ON {0}
     (ID ASC);ALTER TABLE {0} ADD CONSTRAINT IDI_DT_{UNIQUE} PRIMARY KEY (ID)
#com.ibm.di.store.create.delta.store=CREATE TABLESPACE TS1DST LOCKSIZE ROW BUFFERPOOL BP32K;CREATE TABLE {0}
     (ID VARCHAR(VARCHAR_LENGTH) NOT NULL, SEQUENCEID int, ENTRY BLOB) IN TS1DST; CREATE UNIQUE INDEX DSIX1 ON {0}
     (ID ASC); ALTER TABLE {0} ADD CONSTRAINT IDI_DS_{UNIQUE} Primary Key (ID);CREATE LOB TABLESPACE DSENT11 BUFFERPOOL
     BP32K LOCKSIZE LOB;CREATE AUX TABLE TBDSEN1 IN DSENT11 STORES {0} COLUMN ENTRY;CREATE INDEX IXEN1 ON TBDSEN1
#com.ibm.di.store.create.property.store=CREATE TABLESPACE PS3DST LOCKSIZE ROW BUFFERPOOL BP32K;CREATE TABLE {0}
     (ID VARCHAR(VARCHAR_LENGTH) NOT NULL, ENTRY BLOB) IN PS3DST;CREATE UNIQUE INDEX PSIX3 ON {0} (ID ASC);
     ALTER TABLE {0} ADD CONSTRAINT IDI_PS_{UNIQUE} Primary Key (ID);CREATE LOB TABLESPACE PSENT31 BUFFERPOOL BP32K
     LOCKSIZE LOB;CREATE AUX TABLE TBPSEN3 IN PSENT31 STORES {0} COLUMN ENTRY;CREATE INDEX PSIXEN3 ON TBPSEN3
#com.ibm.di.store.create.sandbox.store=CREATE TABLE {0} (ID VARCHAR(VARCHAR_LENGTH) NOT NULL, ENTRY BLOB)
#com.ibm.di.store.create.recal.conops=CREATE TABLESPACE IM{UNIQUE} LOCKSIZE ROW BUFFERPOOL BP32K;CREATE TABLE {0}
     (METHOD VARCHAR(VARCHAR_LENGTH), RESULT BLOB, ERROR BLOB) IN IM{UNIQUE};CREATE LOB TABLESPACE LB{UNIQUE}
     BUFFERPOOL BP32K LOCKSIZE LOB;CREATE AUX TABLE AT{UNIQUE} IN LB{UNIQUE} STORES {0} COLUMN RESULT;CREATE INDEX IX
     {UNIQUE} ON AT{UNIQUE};CREATE LOB TABLESPACE LS{UNIQUE} BUFFERPOOL BP32K LOCKSIZE LOB;CREATE AUX TABLE
     AE{UNIQUE} IN LS{UNIQUE} STORES {0} COLUMN ERROR;CREATE INDEX IN{UNIQUE} ON AE{UNIQUE}


# Set a customized SQL statement for creation of the Tombstone Manager table.
# Keep the same table and field names. This is the default Derby statement.
#com.ibm.di.store.create.tombstones=CREATE TABLE IDI_TOMBSTONE ( ID INT GENERATED ALWAYS AS IDENTITY,
     COMPONENT_TYPE_ID INT, EVENT_TYPE_ID INT, START_TIME TIMESTAMP, CREATED_ON TIMESTAMP, COMPONENT_NAME
     VARCHAR(1024), CONFIGURATION VARCHAR(1024), EXIT_CODE INT, ERROR_DESCR VARCHAR(1024),
     STATS LONG VARCHAR FOR BIT DATA, GUID VARCHAR(1024) NOT NULL, USER_MESSAGE VARCHAR(1024), UNIQUE (ID, GUID))

# The following two SQL statements could be used when SolidDB is used as System Store
#com.ibm.di.store.create.tombstones=CREATE TABLE IDI_TOMBSTONE (ID INT PRIMARY KEY, COMPONENT_TYPE_ID INT,
     EVENT_TYPE_ID INT, START_TIME TIMESTAMP, CREATED_ON TIMESTAMP, COMPONENT_NAME VARCHAR(1024),
     CONFIGURATION VARCHAR(1024), EXIT_CODE INT, ERROR_DESCR VARCHAR(1024), STATS LONG VARBINARY,
     GUID VARCHAR(1024) NOT NULL, USER_MESSAGE VARCHAR(1024), UNIQUE (ID, GUID));CREATE SEQUENCE IDI_TOMBSTONE_SEQ
#com.ibm.di.store.update.tombstones=INSERT INTO IDI_TOMBSTONE (ID, COMPONENT_TYPE_ID, EVENT_TYPE_ID, START_TIME,
     CREATED_ON, COMPONENT_NAME, CONFIGURATION, EXIT_CODE, ERROR_DESCR, STATS, GUID, USER_MESSAGE)
     VALUES (IDI_TOMBSTONE_SEQ.NEXT, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?)

# the ibmsnap_commitseq column name used by the RDBMS changelog connector
com.ibm.di.conn.rdbmschlog.cdcolname=ibmsnap_commitseq

## server authentication
javax.net.ssl.trustStore=serverapi/testadmin.jks
{protect}-javax.net.ssl.trustStorePassword={encr}rI2mxgg5vwrbnooxTCYUFeMSkCKHbl4cGpZuQUW20GeCAyJjqGhiSJLuQJdNQpNNqDNJi+
     isW0mJkzwl0qud8l179xlJtwqLh7zEVFgvuqEwx43ACLoSb9gkG8Je07JF0FgpO9thj6zTCCzqB4NCsmUv11agBHvGtEE8QT73Xh8=
javax.net.ssl.trustStoreType=jks

## client authentication
javax.net.ssl.keyStore=serverapi/testadmin.jks
{protect}-javax.net.ssl.keyStorePassword={encr}QnVT+6gn66nE3zUnvHQEp9TM8k52BQxJsae85mnwQkecHseSCysd0MrWltXCOMFU2RJoZL
     cqLVp3WpGp0QX+n9XQVBXGOXXmIjhwIs7iDr73DpkMloJzSruYhKPQGKOxgfC+80lIAfSoMixNgp76iIYbqiRqJHKV3sSQT9VB0mM=
javax.net.ssl.keyStoreType=jks

##PKCS11 options
##Set the value of following properties to use PKCS11 enabled devices to store SDI servers private key / certificate.
com.ibm.di.pkcs11cfg=etc/pkcs11.cfg
com.ibm.di.server.pkcs11=false
com.ibm.di.server.pkcs11.library=
com.ibm.di.server.pkcs11.slot=
{protect}-com.ibm.di.server.pkcs11.password={encr}iYbgH/Y/pw4YSUXVqpKQnWZzlZHKa52CuCRnHnnBen3/yIt0JlK+nrepEW8jN2KBDhM
     8Z+zIPs/0YlIx9y20X/nvp/QgevgPsAvzvxznD7QtWjqAySTKWjT+i1BVCbnkhJOiRXXrIQkwSxtyl/oUcdskk5+wNSYLtvuSUrD3J38=

## Turns on java debug
# javax.net.debug=true

## java interpreter override
# com.ibm.di.javacmd=
# com.ibm.di.installdir=

## Limits the number of threads IDI uses
## Must be set higher than 3 to have any effect
```

```
# com.ibm.di.server.maxThreadsRunning=500

com.ibm.di.server.securemode=false

## Following properties modified in SDI 7.1 Added property for
## keystore password and keypassword
## com.ibm.di.server.keystore
## com.ibm.di.server.key.alias

api.keystore=testserver.jks
api.keystore.type=jks
api.key.alias=server
{protect}-api.keystore.password={encr}S8BAYYIIdW1y0FHuslpXHN6F4Xc76gCUvm39ZKZHsjRMmZQmY7S1p+7gh9YHE3AIquLGgf4n0MzFybU
     S/C6xy2RIijVbq+HGz2YU+4SNnGjt5o53KAxIegLC2ml+JCUu4UY/P/ASjCXFhfL/iJsRI4hxLhFg266pl3eIJeVxflc=
{protect}-api.key.password={encr}

## Encryption properties added in SDI 7.1
com.ibm.di.server.encryption.keystore = testserver.jks
com.ibm.di.server.encryption.key.alias = server
com.ibm.di.server.encryption.keystoretype = jks
com.ibm.di.server.encryption.transformation = RSA

## Web container
web.server.port=1098
web.server.ssl.on=true
web.server.ssl.client.auth.on=false
# web.server.session.timeout=300

## Touchpoint Server properties
tp.server.on=false
tp.server.config=etc/tp.xml
tp.server.auth=false
tp.server.auth.realm=Security Directory Integrator Touchpoint Server

## Dashboard properties
##
dashboard.on=true
dashboard.templates.folder=dashboard/templates

## Dashboard authentication properties
##
## The values for localhost and remotehost can be:
## none: No authentication is required
## deny: All connections denied
## ldap: Authentication is done by logging into an LDAP server and optionally validating group membership
## properties: Authentication is done using dashboard.auth.user.[username]=[password] properties
##
## dashboard.ldap.url
##   Specify the LDAP host port and optionally a search base (ldap://<host>:<port>[/<search-base>])
##
## dashboard.ldap.url.group
##   Specify the LDAP host port and optionally a search base (ldap://<host>:<port>[/<search-base>])
##
dashboard.auth=true
dashboard.auth.localhost=properties
dashboard.auth.remote=deny
# dashboard.auth.ldap.url=ldap://localhost:389/ou=users,ou=system
# dashboard.auth.ldap.url.group=ldap://localhost:389/cn=group1,ou=groups,ou=system
#
# Default FDS username/password
{protect}-dashboard.auth.user.admin={encr}SzFT+3+aSNNWrwtySrBcCbhiVp4bB4hKqSJujGuRSSwtn69blf/UiPbRQbWmQhFidmGpxEULtS9
     S+x4nX0J7rDY2DPVmDfkOu4xqAWT8euS9NvIEp4MfB/whoipQhTWFT3PSVVt+uCc+ONhKunOQuE55IKwAQKdyHPtz+cJkeNM=


## Server API properties
## ---------------------

api.on=true
api.audit.on=false
api.user.registry=serverapi/registry.txt
api.user.registry.encryption.on=false

api.remote.on=true
api.remote.ssl.on=true
api.remote.ssl.client.auth.on=true
api.remote.naming.port=1099
# api.remote.server.ports=8700-8900
api.truststore=testserver.jks
api.truststore.type=jks
{protect}-api.truststore.pass={encr}DzTmOl+sUaose3wpkbHk9vzZ4JZxHL8aMC2ePUb4tWMuS+D7OVcLI5aS8sayg0/ktcOcH6ozy6+qxhlan
     PpYtu1Dh7mZHDsAGDL+Temard/gJUTlxuG4FkAIr5YsDxhZ3nld5fLa8h8YMTVDLd8qx6XZl6f/Ag0a0Yzn882wwFI=

## REST API
## ----------------------
api.rest.on=true
api.rest.auth=false
api.rest.auth.realm=Security Directory Integrator REST API

api.rest.jmsdriver.name=com.ibm.di.systemqueue.driver.ActiveMQ
api.rest.jmsdriver.queue.sender.persistance=false
api.rest.jmsdriver.queue.sender.timeToLive=60000
api.rest.jmsdriver.param.jms.broker=vm://localhost?brokerConfig=xbean:etc/activemq.xml
# api.rest.jmsdriver.auth.username
# api.rest.jmsdriver.auth.password

## The properties determine the default bind address and the remote bind address for the Server API.
## * means bind to all network interfaces. The Remote Bind Address overrides the Default one.
## Only one IP address should be set. No hostnames are accepted.
## Mind that the java.rmi.server.hostname property is set implicitly to equal the Remote Bind Address property when used.
##This will cause the client stubs to create sockets on the specified Remote Bind Address.
# com.ibm.di.default.bind.address=*
# api.remote.bind.address=*

## Specifies a list of IP addresses to accept non SSL connections from (host names are not accepted).
## Use space, comma or semicolon as delimiter between IP addresses. This property is only taken into account
```

```
## when api.remote.ssl.on is set to false.
## api.remote.nonssl.hosts=

api.jmx.on=false
api.jmx.remote.on=false

## The configuration files placed in this folder can be edited through the Server API.
## Configuration files placed in other folders cannot be edited through the Server API.
api.config.folder=configs

## Timeout in minutes for configuration locks. A value of 0 means no timeout.
api.config.lock.timeout=0

## Timeout in minutes for loading a configuration.
api.config.load.timeout=2

## Specifies if the Server API methods for custom method invocation (Session.invokeCustom(...)) are allowed to be used.
## When api.custom.method.invoke.on is set to false and the Server API methods for custom method invocation are used,
## then an exception will be thrown.
## Only classes listed in api.custom.method.invoke.allowed.classes are allowed to be directly invoked.
## The default value is false.
api.custom.method.invoke.on=false

## Specifies the list of classes which can be directly invoked by the Server API methods for custom
## method invocation (Session.invokeCustom(...)).
## This property is only taken into account if api.custom.method.invoke.on is set to true.
## The classes in this list must be separated by a space, a comma or a semicolon.
## Example:
## api.custom.method.invoke.allowed.classes=com.ibm.MyClass,com.ibm.MyOtherClass
## In the above example only methods from the com.ibm.MyClass and com.ibm.MyOtherClass classes are
## allowed to be directly invoked.
api.custom.method.invoke.allowed.classes=

## Specifies a list of Server notification types, which will be suppressed.
## Notifications of suppressed types will not be propagated by the notifications framework.
## The notification types in the list are separated by spaces. Wildcards may be included.
## Example:
## api.notification.suppress=di.al.* di.ci.start
## The above example will suppress all Assembly Line related notifications as well as
## notifications for starting a configuration instance.
## If the property is missing or is empty, no notifications will be suppressed.
api.notification.suppress=di.server.api.authenticate di.server.api.authorize.*

## api.custom.authentication points to a JavaScript text file that contains custom authentication code.
## For example: api.custom.authentication=ldap_auth.js.
## To enable the built-in LDAP Authentication mechanism, set this property to "[ldap]".
## To enable the built-in JAAS Authentication mechanism, set this property to "[jaas]".
## For example: api.custom.authentication=[ldap]

##api.custom.authentication=[ldap]

## LDAP Authnetication properties
## ----------------------

## If this parameter is set to "true" and the LDAP Authnetication initialization fails,
## the whole Server API will not be started.
## If this parameter is missing or is set to "false" any LDAP Authentication initialization errors will be logged
## and the Server API will be started.
api.custom.authentication.ldap.critical=false

## LDAP Server hostname.
api.custom.authentication.ldap.hostname=

## LDAP server port number. 例えば、非 SSL の場合は 389、SSL の
場合は 636 です。
api.custom.authentication.ldap.port=

## Specifies whether SSL is used to communicate with the LDAP Server.
## When set to "true" SSL will be used, otherwise SSL will not be used.
api.custom.authentication.ldap.ssl=

## Specifies the LDAP directory location where user searches will be preformed.
## When this property is not specified user searches will not be performed.
api.custom.authentication.ldap.searchbase=

## Specifies the user id attribute to be used in searches.
## When this property is not specified user searches will not be performed.
api.custom.authentication.ldap.userattribute=

## Specifies an LDAP Server administrator distinguished name that will be used for user searches.
## When this property is not specified anonymous bind will be used for user searches.
api.custom.authentication.ldap.admindn=

## Password for the LDAP Server administrator distinguished name.
{protect}-api.custom.authentication.ldap.adminpassword={encr}

## This property specifies whether LDAP Group authentication is turned on.
## If it is set to 'true', the group membership of the authenticating user will be resolved
## and will be taken into account during authorization.
## If it is missing, the default value 'false' is used.
api.custom.authentication.ldap.groupsupport=false

## Specifies the name of the attribute of a user in LDAP that contains a list of the groups
## of which the user is a member.
## It is taken into account only if 'api.custom.authentication.ldap.groupsupport' is set to true.
api.custom.authentication.ldap.usermembershipattribute=

## Specifies how groups are named in the membership attribute of a user.
## For example, if the user's membership attribute contains values, which correspond to the 'objectSID' attributes
## of groups, set this property to 'objectSID'.
## If the user's membership attribute contains distinguished names of groups, then set this property to 'dn'.
## The property is required in case 'api.custom.authentication.ldap.groupsupport' is set to true.
api.custom.authentication.ldap.usermembershipattributecontent=

## Specifies the name of a group's attribute in LDAP, which corresponds to the way the
## group is named in the SDI User Registry.
```

```
## For example, if LDAP groups are addressed in the SDI registry by their common name, then set this property to 'cn'.
## If the User Registry contains the distinguished names of the groups, then set this property to 'dn'.
api.custom.authentication.ldap.groupnameattribute=

## Represents the LDAP directory context, where groups will be searched.
## It is required only when LDAP group support is enabled
api.custom.authentication.ldap.groupsearchbase=

## Optional property, which represents a list of space-separated attribute names.
## Specifies attributes which have non-string syntax.
## api.custom.authentication.ldap.binaryattributes=

## JAAS Authnetication properties
## ----------------------
java.security.auth.login.config=

## Enabling/Disabling FIPS Mode in SDI
##-------------------------------------
## If the below property is set to true then SDI will be enforced to run in FIPS Compliant Mode.
## The default value is false, i.e. SDI will not run in FIPS Mode by default.
com.ibm.di.server.fipsmode.on=false

## Specify the unique ID for the SDI Server
## ----------------------------------------
## This property helps a client connecting to the SDI server to identify different servers
## running on the same IP and the same port in different time. (Default is DEFAULT_ID)
com.ibm.di.server.id=DEFAULT_ID

## Tombstone Manager properties
## ----------------------

com.ibm.di.tm.on=false
com.ibm.di.tm.autodel.age=0
com.ibm.di.tm.autodel.records.trigger.on=10000
com.ibm.di.tm.autodel.records.max=5000
com.ibm.di.tm.create.all=false


## ----------------------
## Help system properties
## ----------------------

## Name of help server. The Tivoli library is at the following URL:
## http://www-01.ibm.com/support/knowledgecenter/SSCQGF/welcome

## Port for help system
com.ibm.di.helpPort=80

## ------------------------------------------
## AssemblyLinePool: Connector pooling defaults
## ------------------------------------------
##
## Note! These settings are only used when an AssemblyLine uses
## an AssemblyLinePool in combination with a Server mode connector.

## The number of seconds before a pooled connector times (e.g. is closed and no longer reused)
##      Less than zero means disable connector pooling
##  Zero means never timeout
##  Greater than zero sets the number of seconds before a connector is closed
com.ibm.di.server.connectorpooltimeout=42

## Comma separated list of connector interfaces that we never pool
com.ibm.di.server.connectorpoolexclude=com.ibm.di.connector.FileConnector,com.ibm.di.connector.ScriptConnector


## Properties for Windows IPv6 communications.
## Uncomment these properties for Windows IPv6 communication only.
## These properties will not affect IPv4 communication or IPv6 communication on Unices.
#java.net.preferIPv4Stack=false
#java.net.preferIPv6Addresses=true

## ------------------------------------------
## Performance settings
## ------------------------------------------
##
## Enable/Disable performance logging
com.ibm.di.server.perfStats=false

### ----------------------------------------
### Used by Config Report
###----------------------------------------
### set this is you want to override the local language for Config Reports
# com.ibm.di.admin.configreport.translation=en

##----------------------
## System Queue settings
##----------------------
## If set to "true" the System Queue is initialized on startup and can be used;
## otherwise the System Queue is not initialized and cannot be used.
systemqueue.on=true

## Specifies the fully qualified name of the class that will be used as a JMS Driver.
# systemqueue.jmsdriver.name=com.ibm.di.systemqueue.driver.IBMMQ
# systemqueue.jmsdriver.name=com.ibm.di.systemqueue.driver.JMSScriptDriver
systemqueue.jmsdriver.name=com.ibm.di.systemqueue.driver.ActiveMQ

### MQ JMS driver initialization properties
# systemqueue.jmsdriver.param.jms.broker=<host:port>
# systemqueue.jmsdriver.param.jms.serverChannel=<channel_name>
# systemqueue.jmsdriver.param.jms.qManager=<queuemanger_name>
# systemqueue.jmsdriver.param.jms.sslCipher=<cipherSuite_name>
# systemqueue.jmsdriver.param.jms.sslUseFlag=false

### JMS Javascript driver initialization properties
## Specifies the location of the script file
# systemqueue.jmsdriver.param.js.jsfile=driver.js
```

```
### ActiveMQ driver initialization properties
## Specifies the location of the ActiveMQ initialization file.
## This file is used to initialize ActiveMQ on SDI server startup.
systemqueue.jmsdriver.param.jms.broker=vm://localhost?brokerConfig=xbean:etc/activemq.xml

## This is the place to put any JMS provider specific properties needed by a JMS Driver,
## which connects to a 3rd party JMS system.
## All JMS Driver properties should begin with the 'systemqueue.jmsdriver.param.' prefix.
## All properties having this prefix are passes to the JMS Driver on initialization after
## removing the 'systemqueue.jmsdriver.param.' prefix from the property name.
# systemqueue.jmsdriver.param.user.param1=value1
# systemqueue.jmsdriver.param.user.param2=value2
# ...

## Credentials used for authenticating to the target JMS system
# {protect}-systemqueue.auth.username=<username>
# {protect}-systemqueue.auth.password=<password>



## ------------------------------------------------
## Logging settings
## ------------------------------------------------

## When false, all log calls made through the SDI Log class will be discarded.
com.ibm.di.logging.enabled=true



## ------------------------------------------------
## IBM JavaScript Engine settings
## ------------------------------------------------

## Set the type of platform - required by the IBM JS Engine when caching is used.
com.ibm.commons.platform=com.ibm.commons.platform.GenericPlatform

##
## Set this property to a directory to enable auto dumps of assemblylines that fails
##
# com.ibm.tdi.autodump.directory=<dump-directory>



##
## Server API client properties
##
api.client.ssl.custom.properties.on=true
api.client.keystore=serverapi/testadmin.jks
{protect}-api.client.keystore.pass={encr}E/1TC2+UF4B9BACaVcdKoalYj38LFi26UncEFTsWP+qj68fuJH8EwCs392IoLXvIAKHlPaWVJo7h
     ySkeBxESorVACAF8oNbHXIBQOS2nvNPKxlkQZK32CaRlg5Wq8TNamQxHFr++UewBuDEAU1ki5z7zXidD2g8g0sN6cK1IhUo=
api.client.keystore.type=jks
{protect}-api.client.key.pass={encr}EU1+JvyysxVts7Yn236FysDdxV7IP7TmjCOy/siOmGx8H6Hcy1epHJmGyunQcqIQeN/KpL7M0a3uqzwks
     cWMURmrOWR+O8xyoN+hMpoAu1EvmXjubtd7jdBJVincesL5BYiSwcxGeTsnQJ/MN84RiCfGc1FzwYxvL53npGeyGXk=
api.client.truststore=serverapi/testadmin.jks
{protect}-api.client.truststore.pass={encr}Y8Npl9khRasEUfwYSaAM5RkJK+N0DOKezRkXgbLyZtU1V0sFiJfCkeLmw8+MndHvtVkPM/3N1n
     g/y+zv9NKFABVqBVYTJxK5RZx3YV/IgQJMpJK/YhT2hSR8w5XSM7meJOlJK3NjC+Cy9my42ioyT+svLUgpQfs74DXyL8482ww=
api.client.truststore.type=jks

## ----------------------------------
## Mail properties
## ----------------------------------

## This property needs to be set to a valid SMTP host to be able
## to send mail using the system methods.
## mail.smtp.host=

## ----------------------------------
## Enabling/Disabling NIST Mode in SDI
##----------------------------------
## If the below property is set to true then SDI will be enforced to run in NIST Compliant Mode.
## The default value is false, i.e. SDI will not run in NIST Mode by default.
com.ibm.di.server.NIST.on=false
```

付録 B. 外部ツールのモニタリング

Tivoli Monitoring および Tivoli Netcool/OMNIbus などの外部ツールを使用して、IBM SDI をモニターすることができます。その動作と対応するコンポーネントの詳細については、以下の情報を参照してください。

これは、IBM Security Directory Integrator、IBM Tivoli Monitoring、および IBM Tivoli Netcool/OMNIbus の統合の「最初のステップ」です。それは、この統合シナリオの PoC (概念検証) として開始されました。この文書に示されている、IBM Security Directory Integrator から IBM Tivoli Monitoring、および IBM Security Directory Integrator から OMNIbus の統合能力は、IBM Security Directory Integrator に同梱されている完全にサポートされたソリューションです。このソリューションは、サンプル・ディクショナリー内に同梱されていますが、完全にサポートされています。

IBM Security Directory Integrator はすぐに使用できる JMX インターフェースを提供し、IBM Security Directory Integrator 側での開発が必要ないため、JMX は、IBM Security Directory Integrator と ITM 間の通信用に選択されました。

Tivoli Netcool/OMNIbus an AssemblyLine を介した IBM Security Directory Integrator のモニタリングは、IBM Security Directory Integrator イベントの検出とそれらを OMNIbus に送信するために開発されました。このセクションの目的は、次のものを使用して IBM Security Directory Integrator をモニターする方法を示すことです。

- IBM Security Directory Integrator JMX インターフェースを使用した Tivoli Monitoring
- Tivoli Netcool/OMNIbus

どちらの統合シナリオも正式な IBM Security Directory Integrator サンプルとしてバンドルされ、*TDI_install_dir*/examples/Tivoli_Monitoring ディレクトリー内にあります。

ITM 6.2.0 および Tivoli Netcool/OMNIbus 7.2.1 は、これらの例で使用されました。

ここで説明している実験を実現するには、いくつかのソフトウェア・コンポーネントが必要でした。ここにこれらのコンポーネントのリストとそれらのインストールの実現に使用された参照文書を示します。

- ITM Agent Builder 6.2 - ITM Agent Builder 6.2 User's Guide
- ITM Tivoli Enterprise Portal - ITM Tivoli Enterprise Portal オンライン文書
- Tivoli Netcool/OMNIbus 7.2.1 - Tivoli Netcool/OMNIbus オンライン文書。

JMX は、IBM Security Directory Integrator と Tivoli Monitoring 間の通信に使用されます。 IBM Security Directory Integrator 側では、Tivoli Monitoring が接続するのは Server API の JMX 層です。

IBM Security Directory Integrator と Tivoli Netcool/OMNIbus 間の通信には 2 つのコネクターが使用されます。サーバー通知コネクターは、IBM Security Directory

Integrator Server Notification セットを受信するため、および EIF Connector がイベントを OMNIbus に送信するために使用されます。

ITM を使用した IBM Security Directory Integrator のモニター

以下の情報を参照することで、ITM のアーキテクチャー、既存の構成のインポート、エージェントの作成、およびさらに多くの側面について詳しく知ることができます。

ITM アーキテクチャーの簡易プレゼンテーション

ITM を使用して、データの収集およびシステムのモニターを行うことができます。エージェント・タイプの詳細については、以下の情報を参照してください。

中心で、ITM で提供されるブラウザーはエージェントによって収集されたデータを表します。

ITM Agent は、次の定義で特長付けられます。

「エージェント (管理対象システムと呼ばれます) は、データ収集およびモニターが必要なシステムまたはサブシステム上にインストールされます。エージェントは、ハートビート・ステータスの開始を含め、モニタリング・サーバーに対する属性のデータ収集および配布に責任があります。」(ITM 文書から抽出)

さまざまな種類のエージェントが存在する可能性があります。オペレーティング・システムまたは特別なアプリケーションをモニターするエージェント、または特別に調整されたエージェント (つまり、Universal Agent インターフェースを使用するエージェント) などです。ITM 文書からの次の図は、エージェントのアーキテクチャーおよびデプロイメント・プロセスの両方を説明しています。

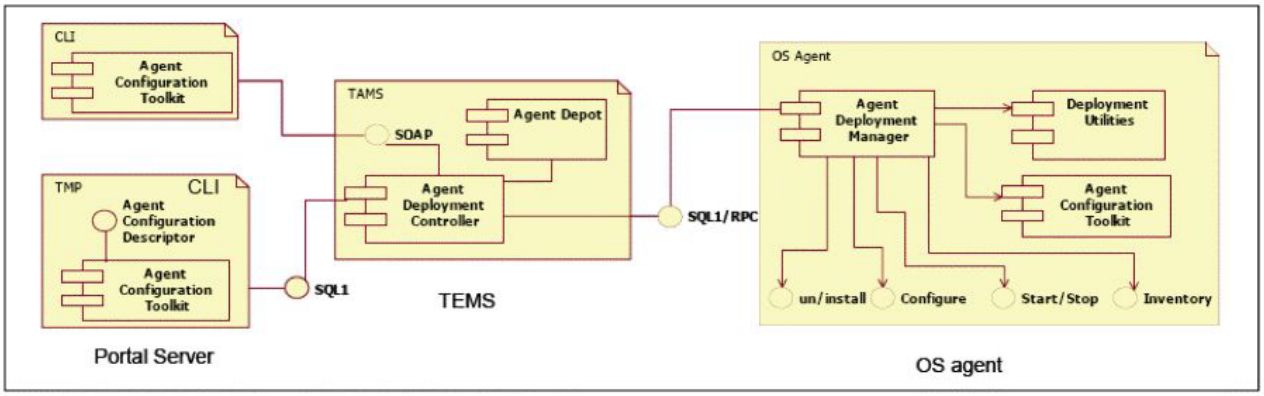


図 2. ITM Agent の図

TEMS = Tivoli Enterprise Monitoring Services
TEP = Tivoli Enterprise Portal

ITM Agent Builder 6.2 への既存のエージェント構成のインポート

ここで説明されている手順に従って、既存のエージェント構成ファイルをインポートします。

ITM Agent 構成 XML ファイルを持っている場合は、それを ITM Agent Builder 6.2 にインポートでき、ITM Agent プロジェクトが自動的に作成されます。そのようなファイルをインポートするには、Builder ワークスペース内で右クリックして「**インポート...**」を選択して「**インポート用 IBM Tivoli Monitoring Agent**」を選択します。構成 XML ファイル (デフォルト名: itm_toolkit_agent.xml) をポイントし「**完了**」をクリックします。これにより ITM Agent Builder プロジェクトが適切な名前で作成されます。

注: エージェントを自身で作成したい場合は、「『ITM Agent Builder 6.2 を使用した IBM SDI agent for ITM の作成』」セクションか「 408 ページの『ITM Agent の生成』」セクションに移動してください。

ITM Agent Builder 6.2 を使用した IBM SDI agent for ITM の作成

以下の例にリストされている手順を使用し、ITM Agent Builder 6.2 を使用して ITM 用の IBM SDI エージェントを作成することができます。

ITM Agent Builder は、Eclipse ベースの ITM Agent 作成用のプラットフォームです。この例のために作成する Agent は、JMX インターフェースを使用します。ITM Agent Builder から「**ファイル -> 新規作成 -> IBM Tivoli Monitoring Agent**」を選択します。

ITM Agent ウィザードが表示されます。最初の手順は、導入です。「**次へ**」をクリックします。2 番目の手順で、プロジェクト名を入力するように要求されます。この例では、プロジェクト名として「SDI」を使用します。「**次へ**」をクリックすると、次の手順に進みます。

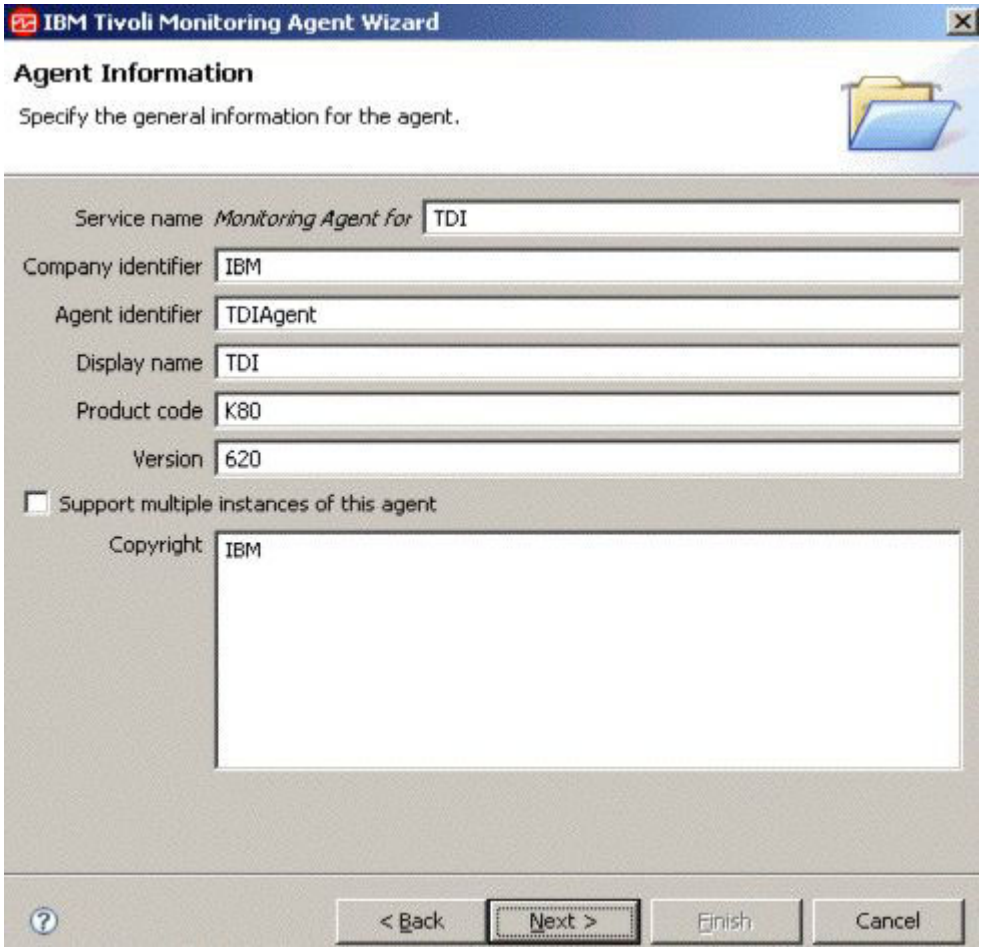


図 3. ITM Agent ウィザード Agent 情報

すべてのフィールドに適切なデータを入力してください。JMX エージェントでは、製品コードは K80 から K99 の間です。「**次へ**」をクリックします。次の手順で、「**このエージェントは、外部データ・ソースからデータを収集します。**」オプションにチェック・マークを付けて「**次へ**」をクリックします。この手順では、データ・ソース定義ウィンドウが表示されます。

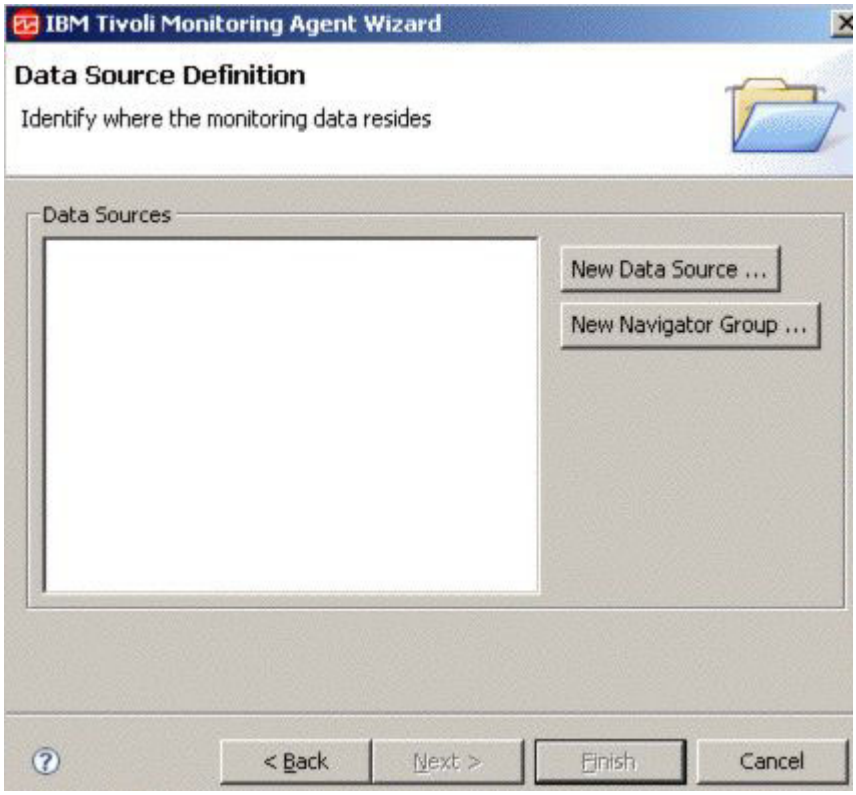


図 4. ITM Agent ウィザード、データ・ソース定義

構成のためのこの手順を簡単にするには、IBM Security Directory Integrator サーバーをデーモン・モードで開始して、終了しない AL を実行します (例えば、HTTP サーバー・コネクターが接続を listen している AL)。JMX API が IBM Security Directory Integrator 内で有効になっていること確認します (これを行う方法については後で説明があります)。

「**新規データ・ソース...**」ボタンをクリックして、「**Java Management Extensions (JMX) MBeans からデータを収集します**」オプションを選択します。「**次へ**」をクリックします。次のウィンドウで「**ブラウズ**」を選択します。JMX Browser が表示されます。

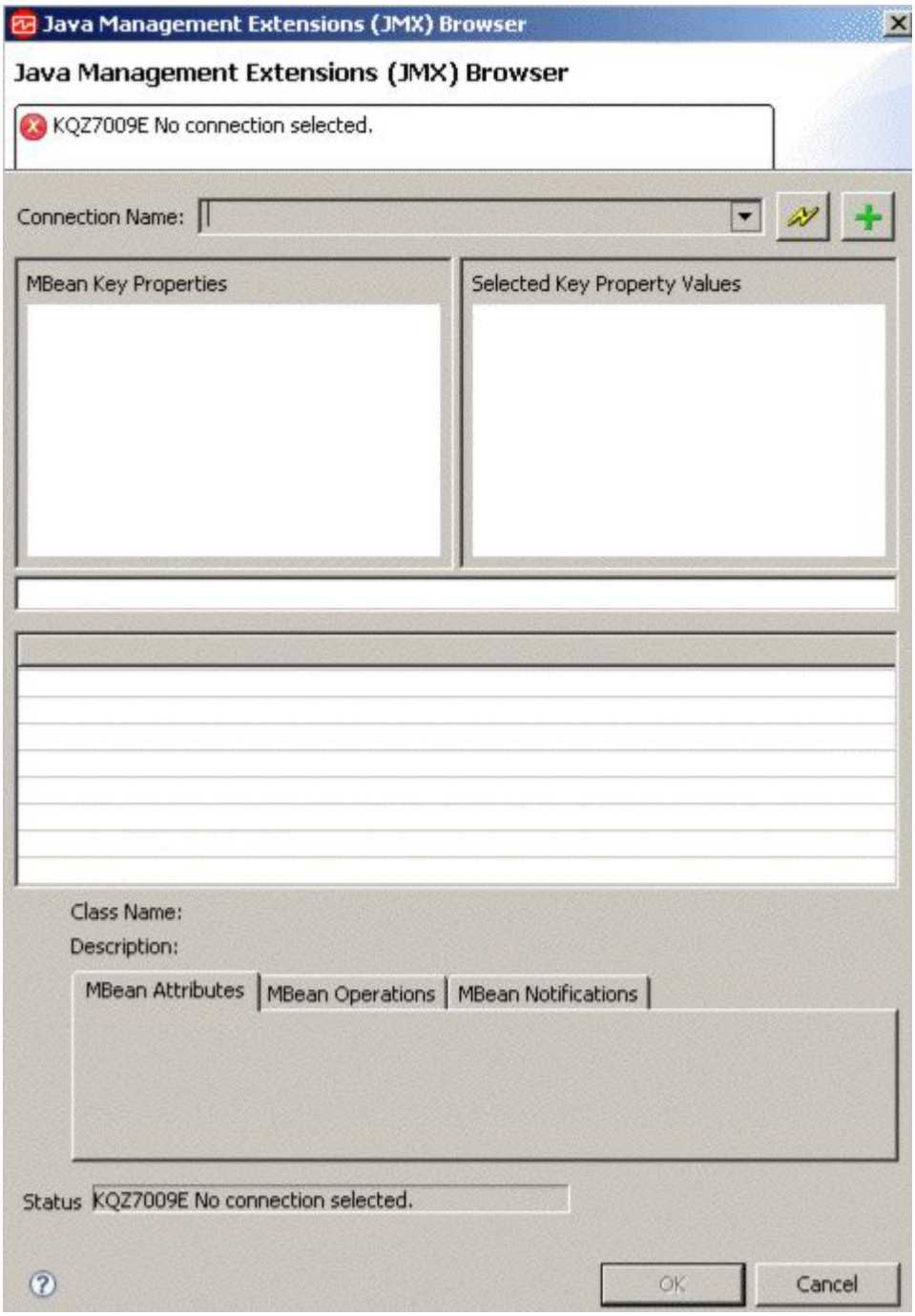


図 5. JMX Browser

「**接続定義の編集**」ボタン (緑色の押しボタン) をクリックします。次の手順で「**標準 JMX 接続 (JSR-160)**」を選択して「**次へ**」をクリックします。新しいウィザード・ウィンドウに利用可能なテンプレートが表示されます。「**JSR-160 準拠サーバー**」を選択し、もう一度「**次へ**」をクリックして JMX Server の接続プロパティーを表示します。

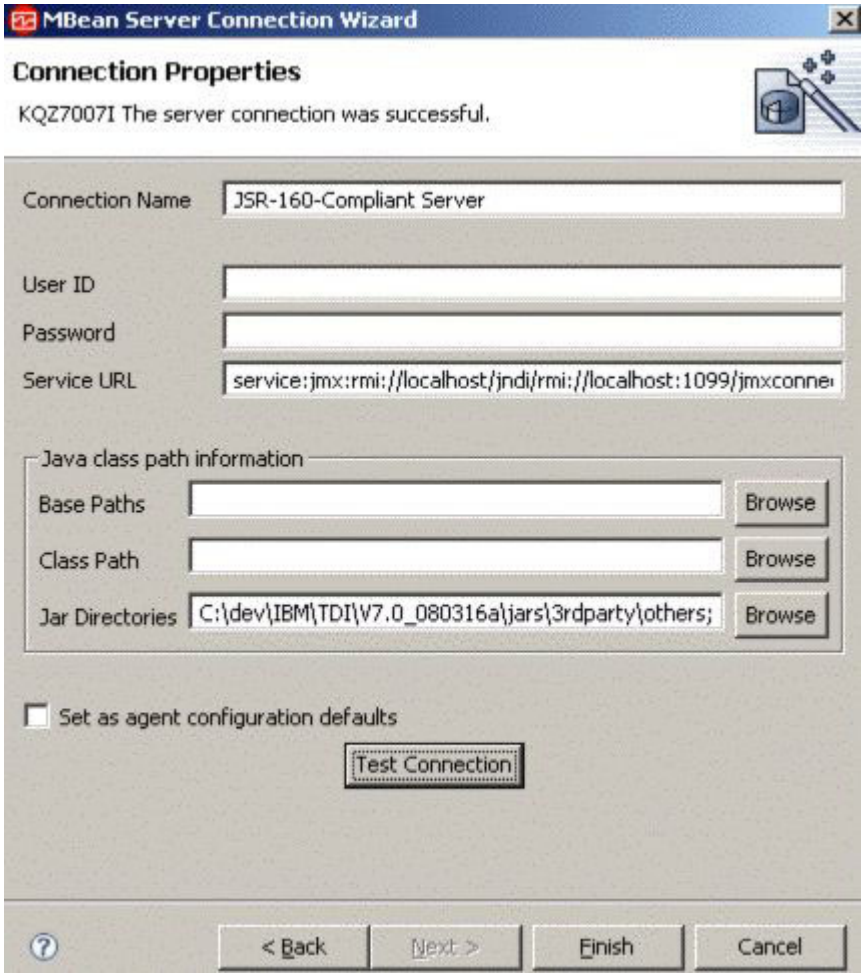


図 6. Server Connection ウィザード

IBM Security Directory Integrator JMX サービスとの正常な接続を確立するには、有効な JMX サービス URL (デフォルトの IBM Security Directory Integrator JMX サービス URL は `service:jmx:rmi://localhost/jndi/rmi://localhost:1099/jmxconnector` です) を入力し、正常な JMX MBeans の作成に必要な jar 依存関係を構成する必要があります (IBM Security Directory Integrator JMX MBeans の場合は、*TDI_install_dir*¥jars¥3rdparty¥IBM; *TDI_install_dir*¥jars¥3rdparty¥others; *TDI_install_dir*¥jars¥common ディレクトリー内に jar ファイルがなければなりません)。これらの設定は、「**テスト接続**」ボタンをクリックしてテストすることができます。構成全体が正常に設定されると、次のようなメッセージが表示されます。「サーバー接続は正常に設定されました。」

この設定後、「**完了**」をクリックします。ウィザードによって直前の構成手順に戻りますが、今度は IBM Security Directory Integrator JMX Server に接続され、追加情報が表示されます。

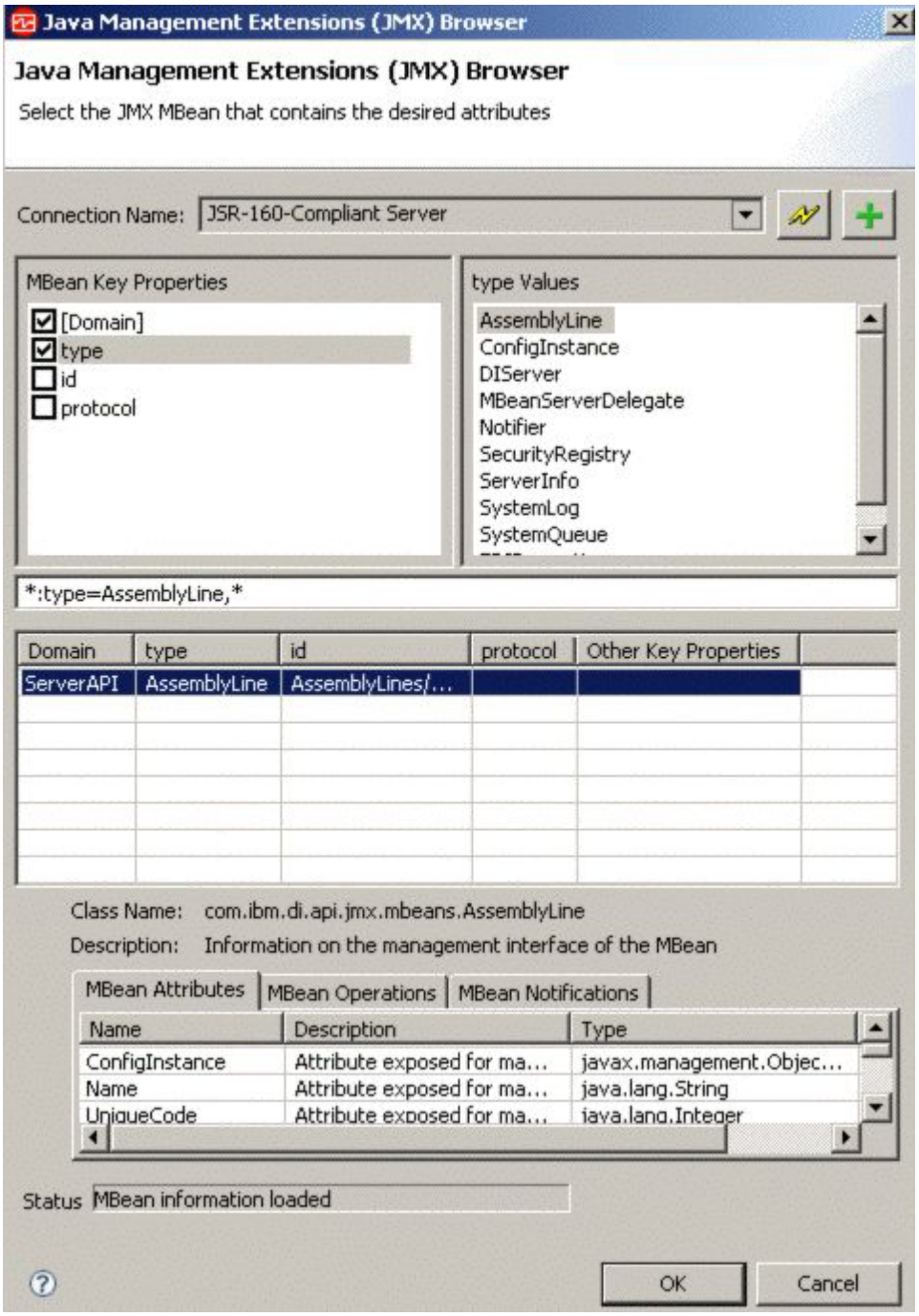


図 7. JMX Browser 内での IBM Security Directory Integrator のブラウズ

タイプ値から「**type** MBean Key Property」および「**AssemblyLine**」を選択します。MBean 属性を参照するには、それらの上にあるテーブル内の行を選択する必要があります。この例では、行は 1 行のみです。「**OK**」をクリックして、「**完了**」をクリックし、このデータ・ソースのセットアップを完了します。

もう一つ、タイプ値 **ConfigInstance** を持つデータ・ソースを AssemblyLine データ・ソースを作成したのと同じ方法で作成します。これら 2 つのデータ・ソースは、実行している AssemblyLines および開始されている構成インスタンス用の情報を JMX Server から収集します。

3 番目のデータ・ソースは、他の 2 つとは少し異なります。これは IBM Security Directory Integrator JMX Server によって送信された通知 (イベント) を listen するリスナーの 1 つです。このようなものを作成する場合、「**新規データ・ソース...**」ボタンをクリックした後に、JMX サーバーをブラウズする必要はありません。MBean パターンとして `*:type=Notifier,*` を入力し、「**完了**」をクリックするだけです。2 つのデータ・ソースが作成されます。1 つは通知部分、1 つは静的 MBean 部分用です。このデータ・ソースには静的部分は必要ないため、削除する必要があります。削除するには、右クリックして「**データ・ソースを削除**」を選択します。

これらの手順を完了した後に、3 つのデータ・ソースができているはずです。

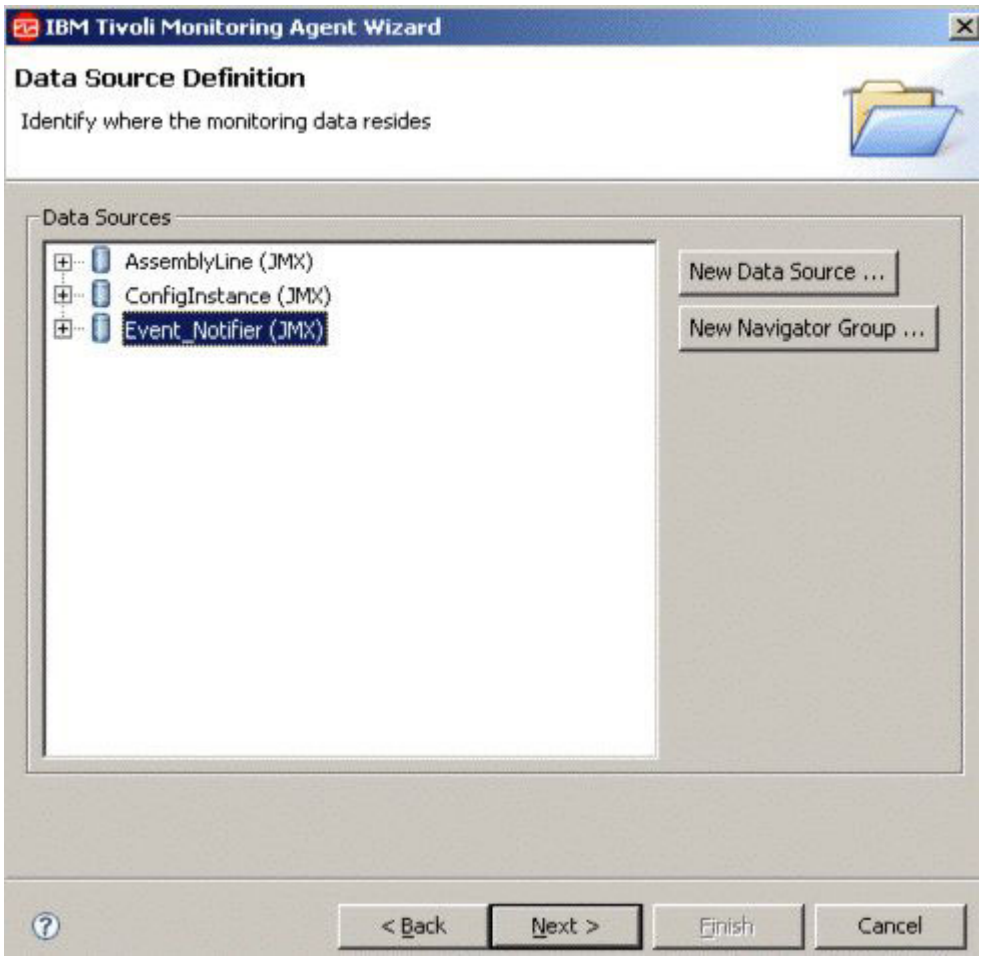


図 8. ITM ウィザード、完了したデータ・ソース定義

AssemblyLine データ・ソースを展開して、「**ConfigInstance**」属性をダブルクリックします。 ConfigInstance 属性構成で、「**キー属性**」チェック・ボックスにチェック・マークを付けます。

ConfigInstance データ・ソースを展開して「**ConfigId**」属性をダブルクリックします。ConfigId 属性構成で、「**キー属性**」チェック・ボックスにチェック・マークを付けます。

「**次へ**をクリックして、「JMX Agent - ワイド・オプション」を構成します。「JMX モニター属性グループ」チェック・ボックスのチェック・マークを外して、サーバー構成の選択肢から「**JSR-160 準拠サーバー**」を選択します。

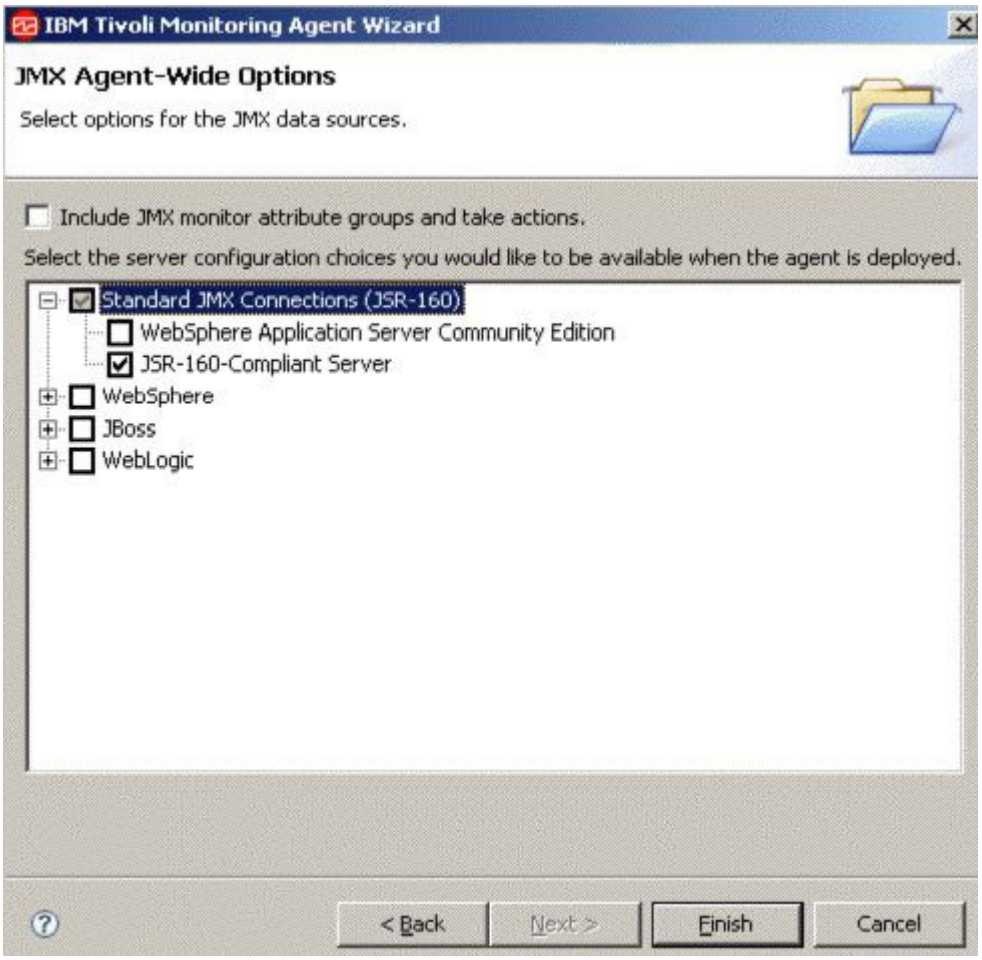


図 9. JMX Agent-wide options

「**完了**」をクリックして ITM Agent 作成手順を完了し、Agent を保存します。

ITM Agent の生成

ITM Agent が正常に作成されたら、以下の手順に従って生成します。

ITM Agent 構成が正常に作成されたら、ITM で展開するために生成する必要があります。ITM Agent Builder の IBM Tivoli Monitoring Agent Editor メニューから「**エージェントの生成**」を選択します。Generate Agent ウィザードが表示されます。このウィザードには、Agent 生成のためのいくつかのオプションがあります。ITM および ITM Agent Builder を 1 台のマシン上で使用している場合は、「**エージェント・ファイルをこのマシン上の ITM インストール内に生成します**」オプションが適しています。構成が必要な唯一のフィールドは、ITM インストール・ディレクトリーです。「**完了**」をクリックして、ITM 内に ITM Agent を生成および展開します。完了するまで数分かかる場合があります。

注: エージェントを別のマシン上に作成する場合は、別のエージェント生成オプション「**エージェントを別のシステム上にインストールできるように圧縮ファイルを作成します**」を使用できます。このオプションでは、ITM Agent インストールを含むアーカイブを生成します。そのようなアーカイブ・エージェントをインストールするには、最初に ITM がインストールされているマシンにファイルをコピーする必要があります。アーカイブからファイルを抽出してコマンド・プロンプトから `InstallIRA.bat` ファイルを ITM インストール・フォルダー内のパラメーターを使用して開始します。

例えば、ITM が `C:¥IBM¥ITM` にインストールされている場合、コマンドは次のようになります。

```
<AgentDirectory>:¥>InstallIRA.bat C:¥IBM¥ITM
```

ITM Agent の構成

ITM Agent をデプロイしたら、それを構成する必要があります。この作業はここで提供される手順および例を使用して行います。

エージェントが正常に展開されたら (アーカイブ・ファイルまたは同じ同じマシン上に展開するための ITM Agent Builder オプションのいずれかを使用して)、ITM 内で構成する必要があります。これを行うには、すべての ITM Agents を管理できる **Manage Tivoli Monitoring Enterprise Services (Tivoli Monitoring Enterprise Services の管理)** を開始します。

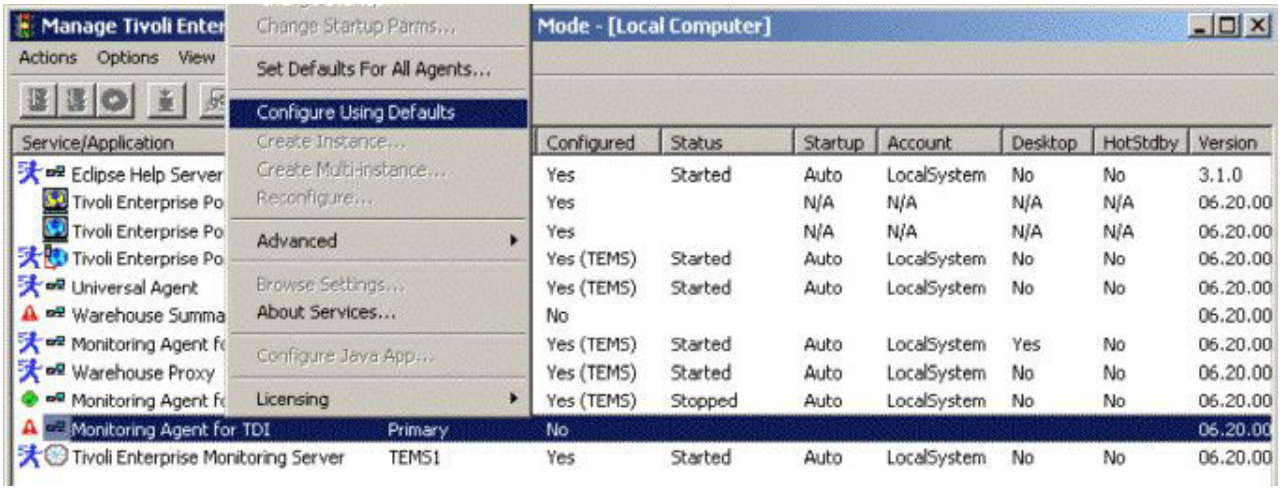


図 10. Manage Tivoli Monitoring Enterprise Services (Tivoli Monitoring Enterprise Services の管理)

IBM Security Directory Integrator Agent を右クリックして「**デフォルトを使用して構成する**」を選択します。

次の構成ウィンドウで、Agent に対して JVM プロパティーを構成する必要があります。使用する Java Home に移動します。ログ・トレース・レベルがデフォルトで「エラー」に設定されます。追加情報をログに記録するためにより高いレベルに変更することができます。Java 構成が完了したら、「**次へ**」ボタンをクリックします。

次の構成手順で、JSR-160 Compliant Server プロパティーを構成するように要求されます。つまり、ユーザー名、パスワード、Service URL および Class Path 依存関係を入力します。この例では、エージェントを作成したときに入力したように、Service URL および Jar ディレクトリーを入力する必要があります。 - Service URL `service:jmx:rmi://localhost/jndi/rmi://localhost:1099/jmxconnector` および Jar directories *TDI_install_dir*¥jars¥3rdparty¥IBM; *TDI_install_dir*¥jars¥3rdparty¥others; *TDI_install_dir*¥jars¥common。

「**OK**」をクリックして Agent 構成を完了します。

IBM Security Directory Integrator Agent が使用できる状態になります。次の手順は、Agent を開始することです。Manage TEMS ウィンドウから、IBM Security Directory Integrator Agent を右クリックして、「**開始**」を選択します。すべての手順が正常に完了していれば IBM Security Directory Integrator Agent が実行されます。

IBM Security Directory Integrator データのモニター

データをモニターするには、Tivoli Enterprise Portal (TEP) を使用します。詳しくは、リストされている手順を参照してください。

データをモニターするには、Tivoli Enterprise Portal (TEP) を開始する必要があります。これは ITM インストールから利用できます。ナビゲーター TEP ウィンドウで、実行している Agent を確認できます。IBM Security Directory Integrator Agent もそこに表示され、特定のモニタリング・データ・ソースを確認するために展開する必要があります。

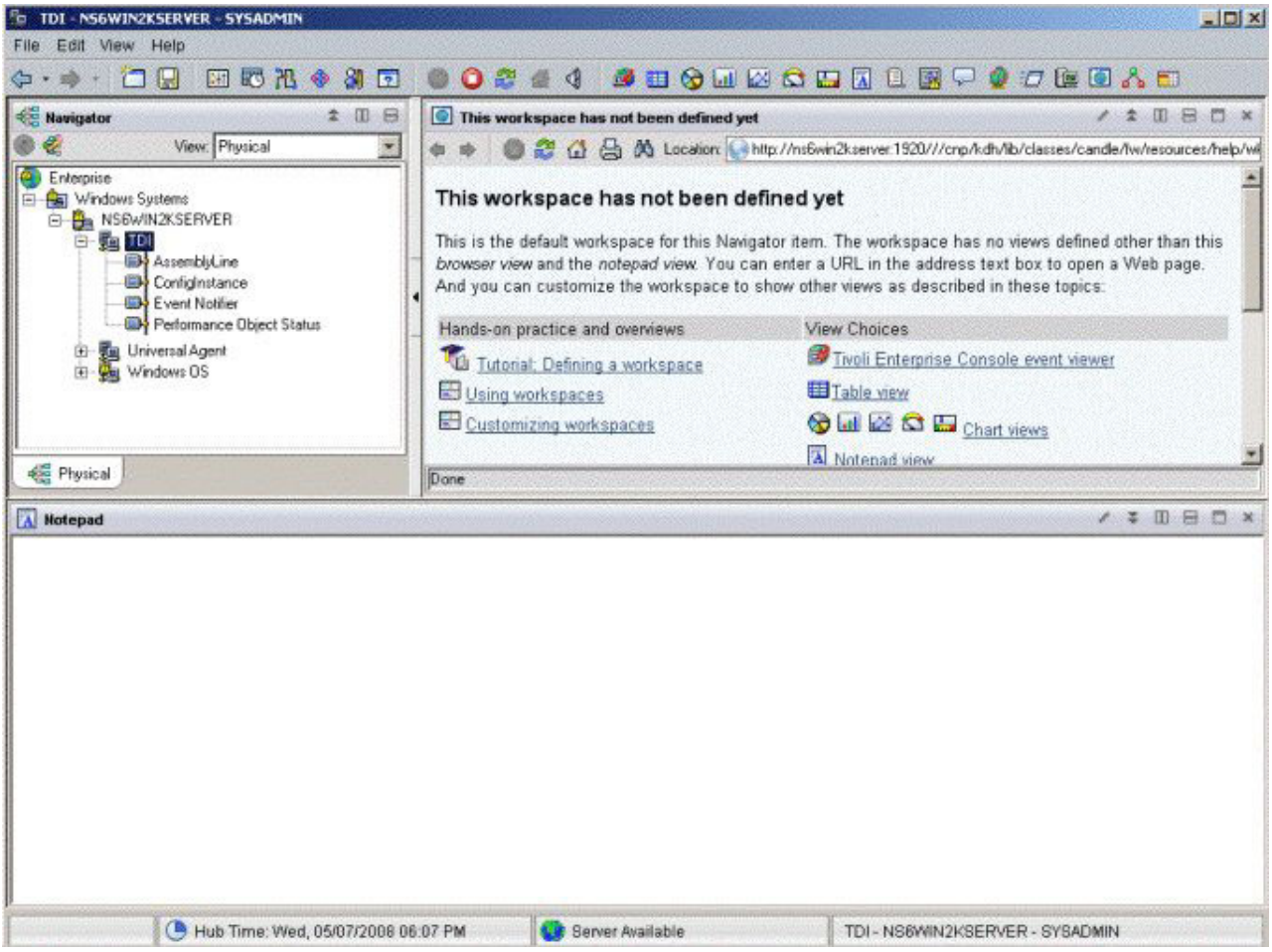


図 11. Tivoli Enterprise Portal (TEP) - ウィザード

カスタム作成したデータ・ソース、AssemblyLine、 ConfigInstance、Event Notifier は、そこに表示されます。実行中の IBM Security Directory Integrator Server および実行中の AL がそこにある場合は、AssemblyLine データ・ソースのレポート・テーブル内に表示されます。

注: データを Notifier レポート・テーブルに表示するには、IBM Security Directory Integrator 通知が起動される前に IBM Security Directory Integrator Agent を実行している必要があります。

このブラウザーは、いくつかの方法で調整できます。数値データの場合は、より読みやすいプレゼンテーション (図など) で表示することができます。テーブルのレイアウトを変更することもできます。

この機能はそれほど複雑ではなく、ITM 文書に詳細な説明が記載されています。

この文書の目的が ITM 製品全体の複雑な詳細を表すことではなく、IBM Security Directory Integrator との相関関係で実行可能な使用方法に焦点をあてることであるため、ここではもっとも慎重を要する 2 つの概念 (しきい値の定義およびテーブル間のリンク) のみを説明します。

その他の機能については、ITM 文書を参照してください。

しきい値の定義

以下の例と画面キャプチャーを使用することで、しきい値メカニズムの動作について学習できます。

しきい値メカニズムがどのように動作するかを示すために、次の簡単な例を作成します。複数の AssemblyLine が現在実行されている場合に警告を表示する。

このしきい値は AssemblyLine テーブルによって提供されるデータに依存します。最初に、エージェントのテーブルを右クリックし、コンテキスト・メニューで「**シチュエーション**」を選択することによってシチュエーションを作成する必要があります。

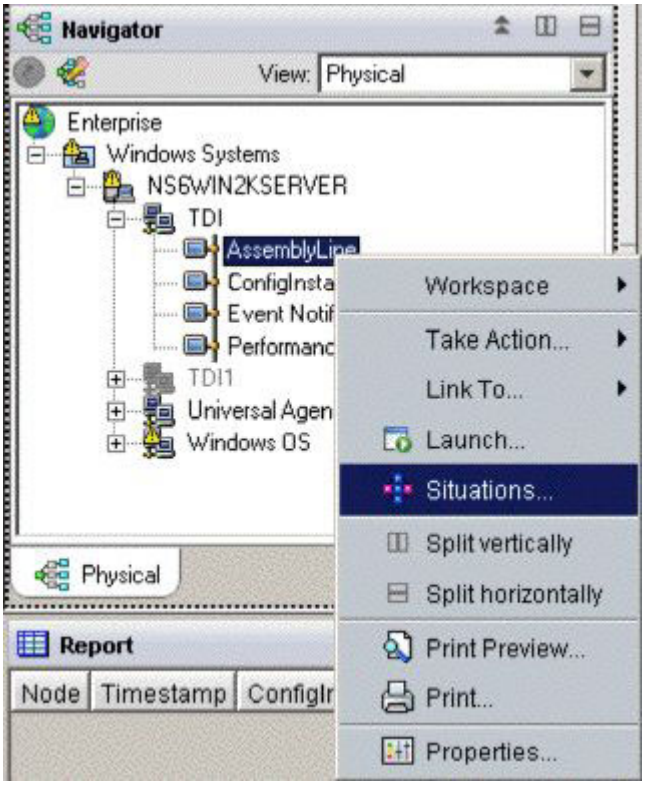


図 12. シチュエーション・コンテキスト・メニュー

左上隅にある「**シチュエーションの作成**」ボタンをクリックして表示されるフォームに入力します。

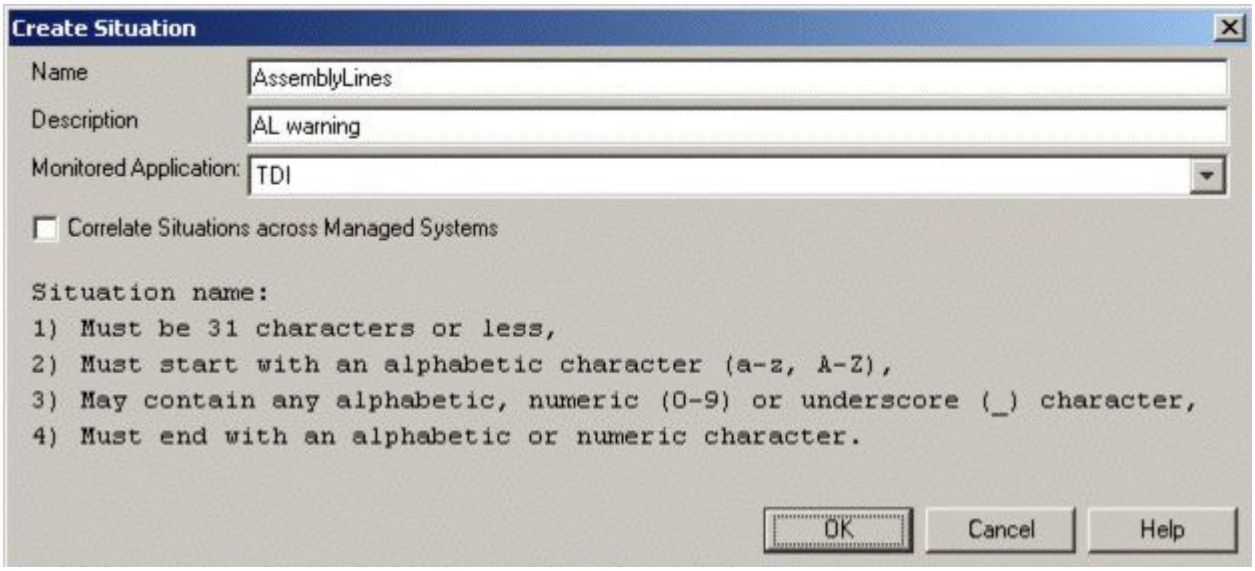


図 13. シチュエーション・フォーム

これは、警告と関連付けられている名前になります。ここでは、ケース・スタディーを説明しますが、実際のシチュエーションでは意味のある名前を指定してください。

次に、シチュエーションで扱うテーブル属性を選択します。

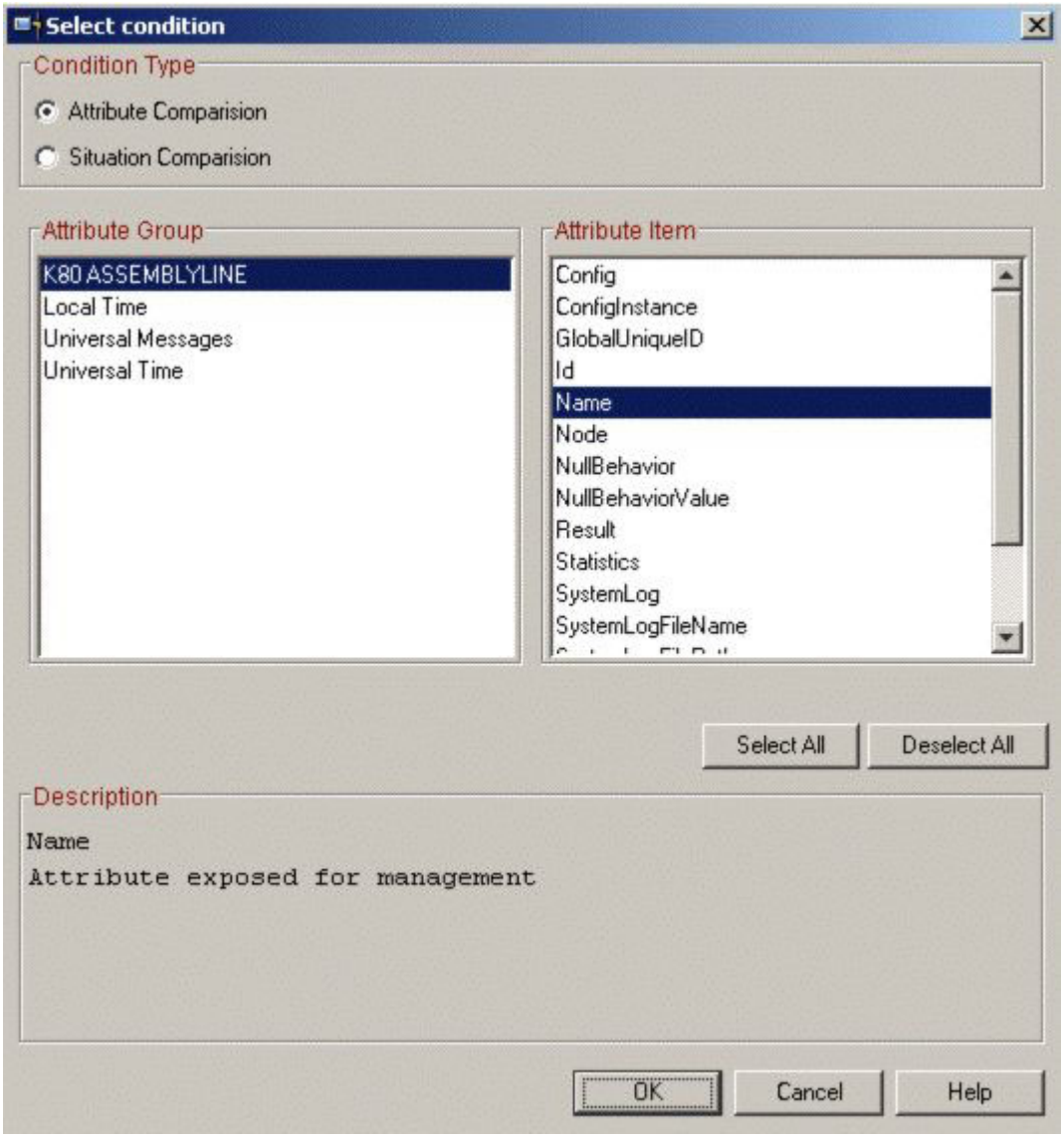
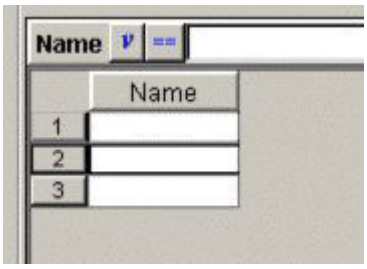


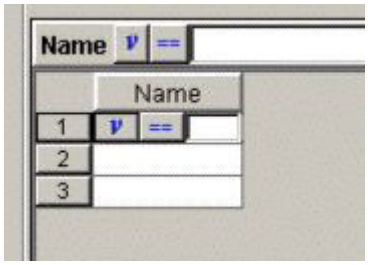
図 14. シチュエーション: 条件の選択

AL を識別するために名前を考えるだけです。

セルの 1 つ、例えば行番号 1 のセルをクリックします。



表示が変更されます。

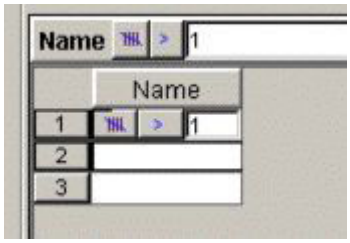


「v」をクリックして「グループ・メンバーのカウント」に変更します。

「==」をクリックして「>」に変更します。

右にある残りのセル・スペースを「1」に設定します。

条件を「**名前**」カラムに構成しました。これは複数の AssemblyLine を実行している場合に真になります。



次のデフォルト設定を変更します。

このように変更します。

シチュエーションが設定されました。このウィンドウを「**適用**」して確認してください。

IBM Security Directory Integrator サーバーを開始して、少なくとも 2 つの AssemblyLine を同時に開始します。例えば、異なるポート上で listen する 2 つの HTTP サーバー・コネクターなどです。

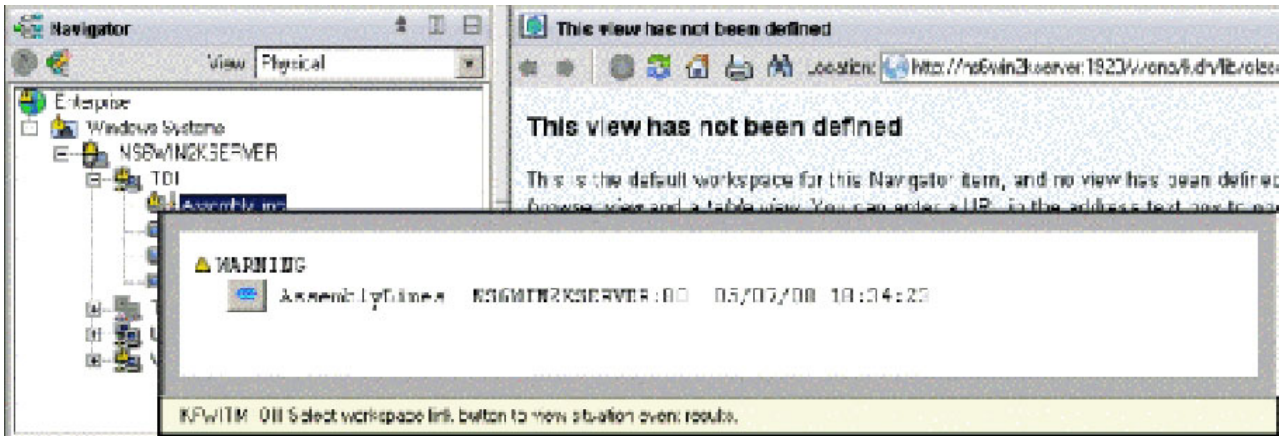


図 15. 警告を表示している ITM

「**警告**」ウィンドウは、警告アイコンを強調表示している間、開いています。

テーブル間のリンクの作成

ITM ではテーブル間のリンクを作成することができます。以下の情報を参照して、リンクの作成の目的とリンクの作成方法について詳しく学習します。

リンクの目的:

リンクをたどることで、データをフィルターし、テーブルのサブセットを直接表示することができます。

ITM 内で異なるテーブル間にリンクを指定することができます。これらのリンクは、この例で示すようにいくつかの選択基準に基づいて選択できます。リンクが作成されると、指定したリンクをたどることによって自動的にテーブルのサブセットが表示されます。この例では、AssemblyLine テーブル内で現在実行中の AssemblyLine を表示するイベント・ノーティファイヤー・テーブルからのリンクを作成します。このリンクは、「di.al.start」タイプを持つレコードに対してのみイベント・ノーティファイヤー・テーブル内で使用可能になります。このタイプは、AssemblyLine が開始されたことを示します。リンクが押されて、AssemblyLine がまだ実行中の場合は、AssemblyLine テーブルが自動的に選択され、対応する AssemblyLine のみがテーブル内に表示されます。 AssemblyLine の実行が既に完了している場合は、表示されるテーブルは空になります。

これは、AssemblyLine テーブルへのリンクが定義されたサンプルのイベント・ノーティファイヤー・テーブルです。

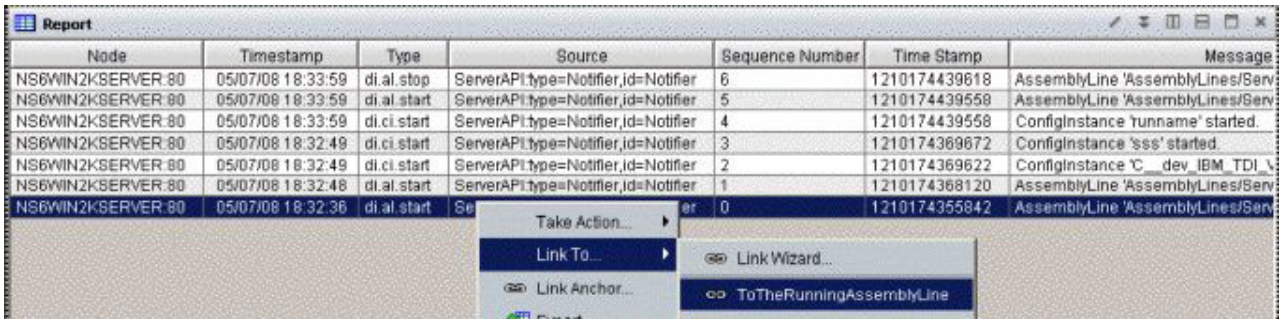


図 16. サンプルのイベント・ノーティファイヤー・テーブル

テーブルには、3 つのロード済み構成、3 つの開始された AssemblyLines (そのうち 1 つは停止されています) があります。「**ToTheRunningAssemblyLine**」リンクが選択されると、AssemblyLine テーブルに AssemblyLine が表示されます (テーブルにはその他の AssemblyLine は表示されません)。

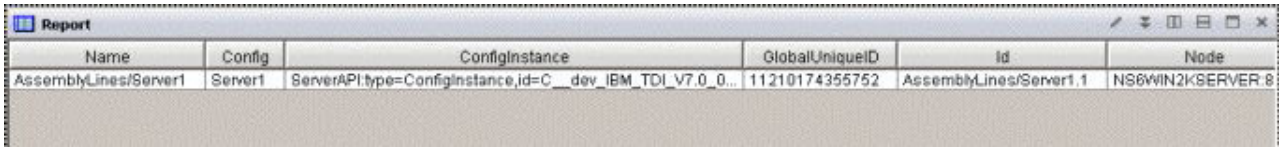


図 17. サンプル・イベント

リンクの構造:

以下にリストされている手順に従って、リンクを構成することができます。

最初に、AssemblyLine テーブル内にイベント・ノーティファイヤー・テーブルからアクセス可能で AssemblyLine ID に対応するキーを作成する必要があります。AssemblyLine テーブルを右クリックしてプロパティーを選択します。

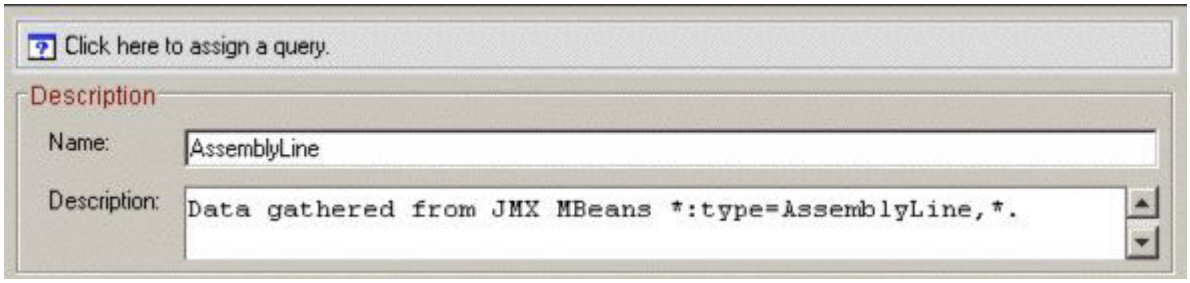


図 18. AssemblyLine プロパティー

開いたウィンドウで、「**ここをクリックして照会を割り当てる**」ボタンをクリックして新しい照会を割り当てます。

Query Editor が開きます。ここでは、既存の照会が静的であり変更できないため、別の照会を定義する必要があります。名前を入力するように求められます (例: 「AssemblyLine2」)。

図 19. 照会を作成する選択肢

テーブルの「**Id**」カラムに移動し、値として「$keyid$」を入力します。

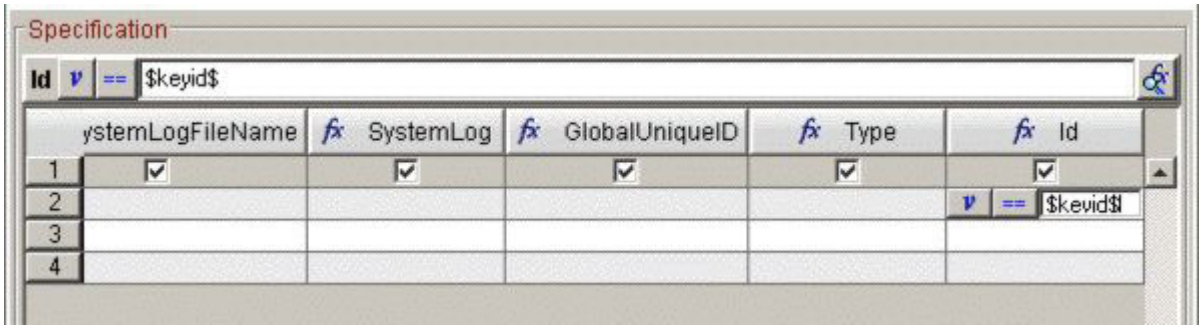


図 20. Query Editor

「**OK**」をクリックしてプロパティー・ウィンドウ内の変更を適用し、「**OK**」ボタンをクリックします。

これは、すべて AssemblyLine テーブル内で実行する必要があります。

Event Notifier テーブルに移動します (AssemblyLine テーブル内の変更内容を保存するように要求されたら「**はい**」をクリックします)。

イベント・ノーティファイヤー・テーブル内で選択した行を右クリックし、「**リンク先...**」 -> 「**リンク・ウィザード...**」を選択します (選択した行のタイプ・カラムが「di.al.start」と同じであることを確認します)。

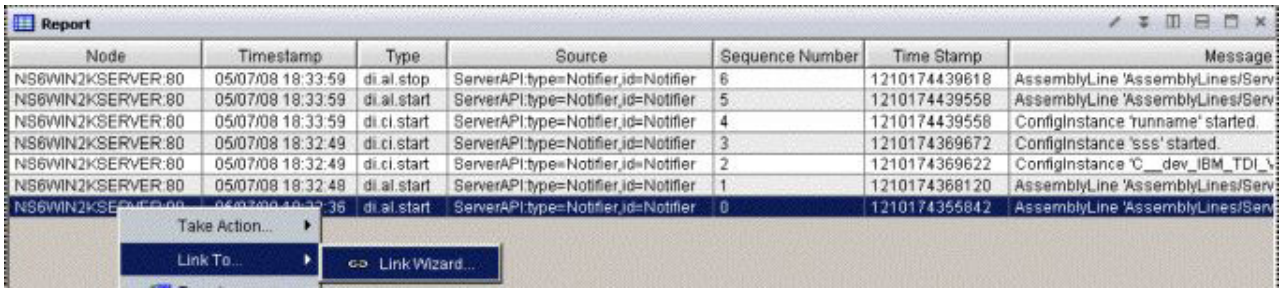


図 21. リンク・ウィザードの選択

リンク・ウィザードが表示され、新しいリンクを作成するか、既存のリンクを変更するか、または既存のリンクを削除するかが確認されます。「**新規リンクの作成**」を選択して、「**次へ**」をクリックします。次の画面で、リンクの名前と説明を入力する必要があります。この例では、名前として「**ToTheRunningAssemblyLine**」、説明として「AssemblyLine テーブルに対応する AssemblyLine を表示します。」を使

用します。次の手順は、リンク・タイプを指定するよう要求されます。この例では、リンク・タイプとして「**Absolute**」を使用します。これは、ナビゲーター・ビュー内の指定した非動的ワークスペースにリンクしているためです。次の手順に進みます。ここではリンクを誘導するワークスペースを指定する必要があります (AssemblyLine テーブル)。

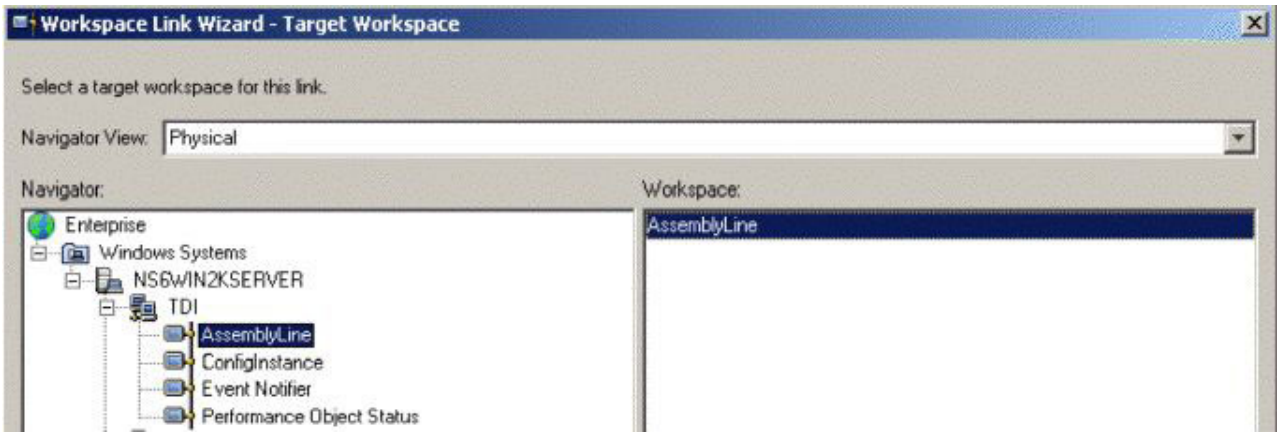


図 22. リンク・ウィザード - ターゲット・ワークスペース

AssemblyLine ワークスペースを選択し、「**次へ**」をクリックします。これはレイン句条件を作成するために必要な最後の手順です。リンクを適切に作成するために 2 つのパラメーター、*contextIsAvailable* および *keyid* を変更します。

「**contextIsAvailable**」パラメーターを選択して、「**式の変更...**」ボタンをクリックします (またはパラメーターをダブルクリックします)。「式エディター」ウィンドウが表示されます。現在の内容を削除して「**シンボル...**」ボタンをクリックします。「シンボル」から「**タイプ**」属性を選択します。

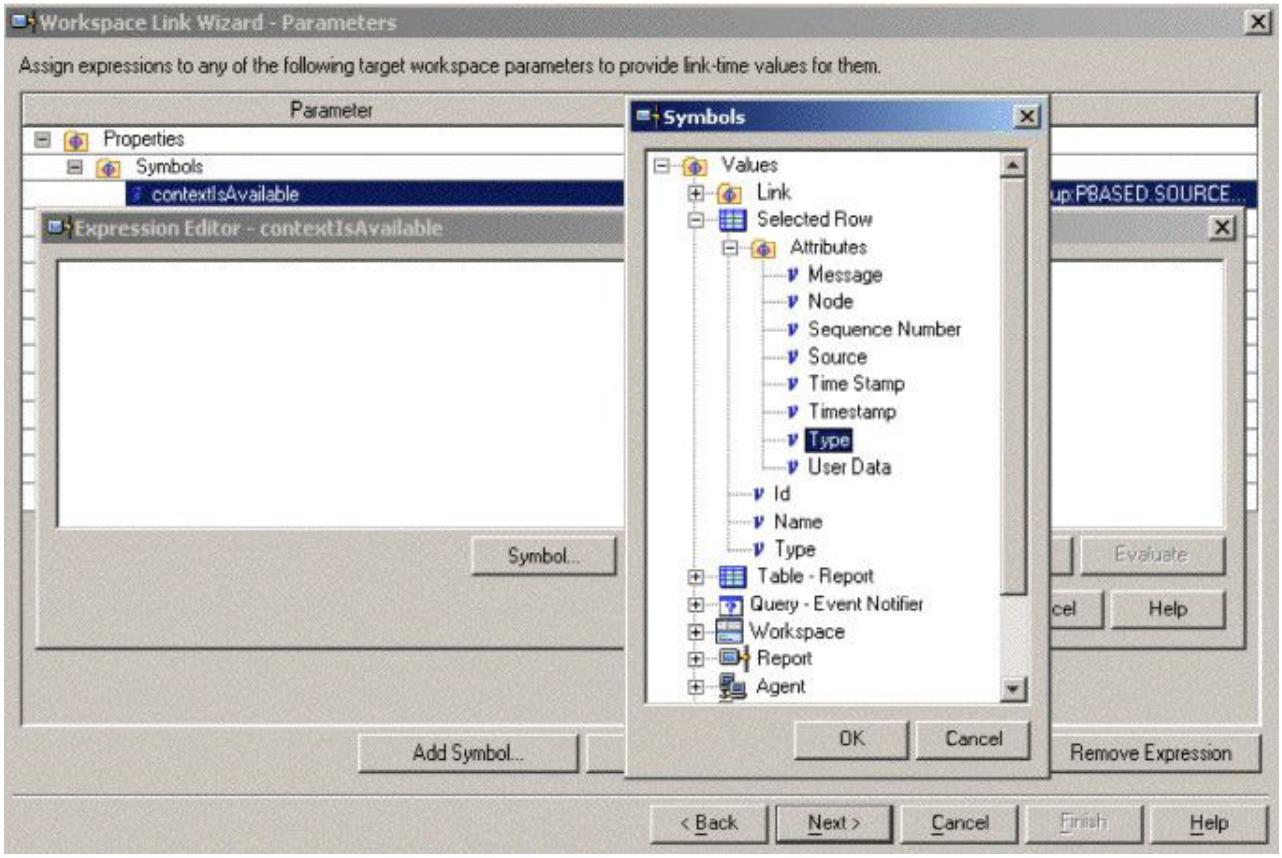


図 23. リンク・ウィザード - タイプ属性

「**OK**」をクリックして「式エディター」ウィンドウに戻り、「`== di.al.start`」を追加して条件式を作成します。

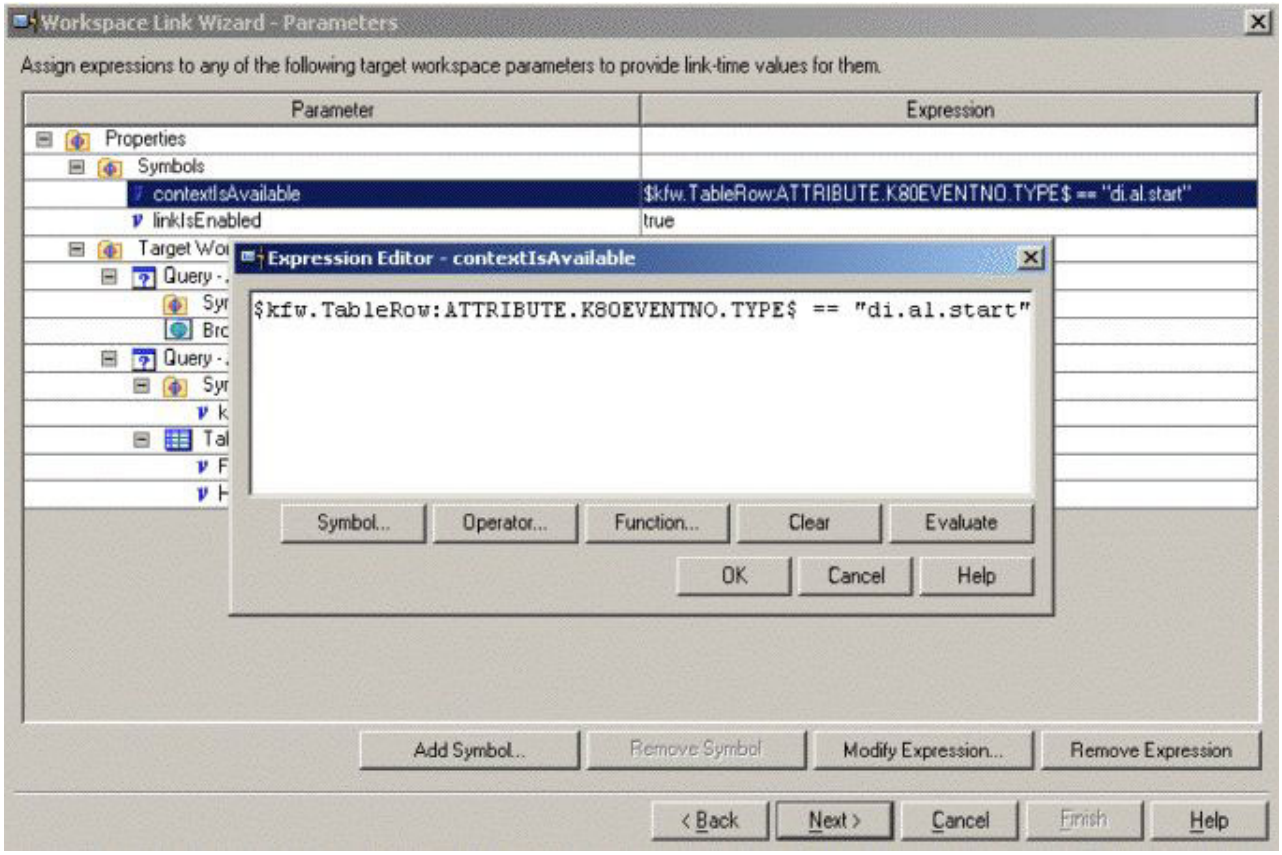


図 24. リンク・ウィザード - 式エディター

「**OK**」をクリックして式値を確認します。

「**照会 - AssemblyLine2**」の keyid シンボルの式エディターを開き、「ユーザー・データ」シンボルを追加します。

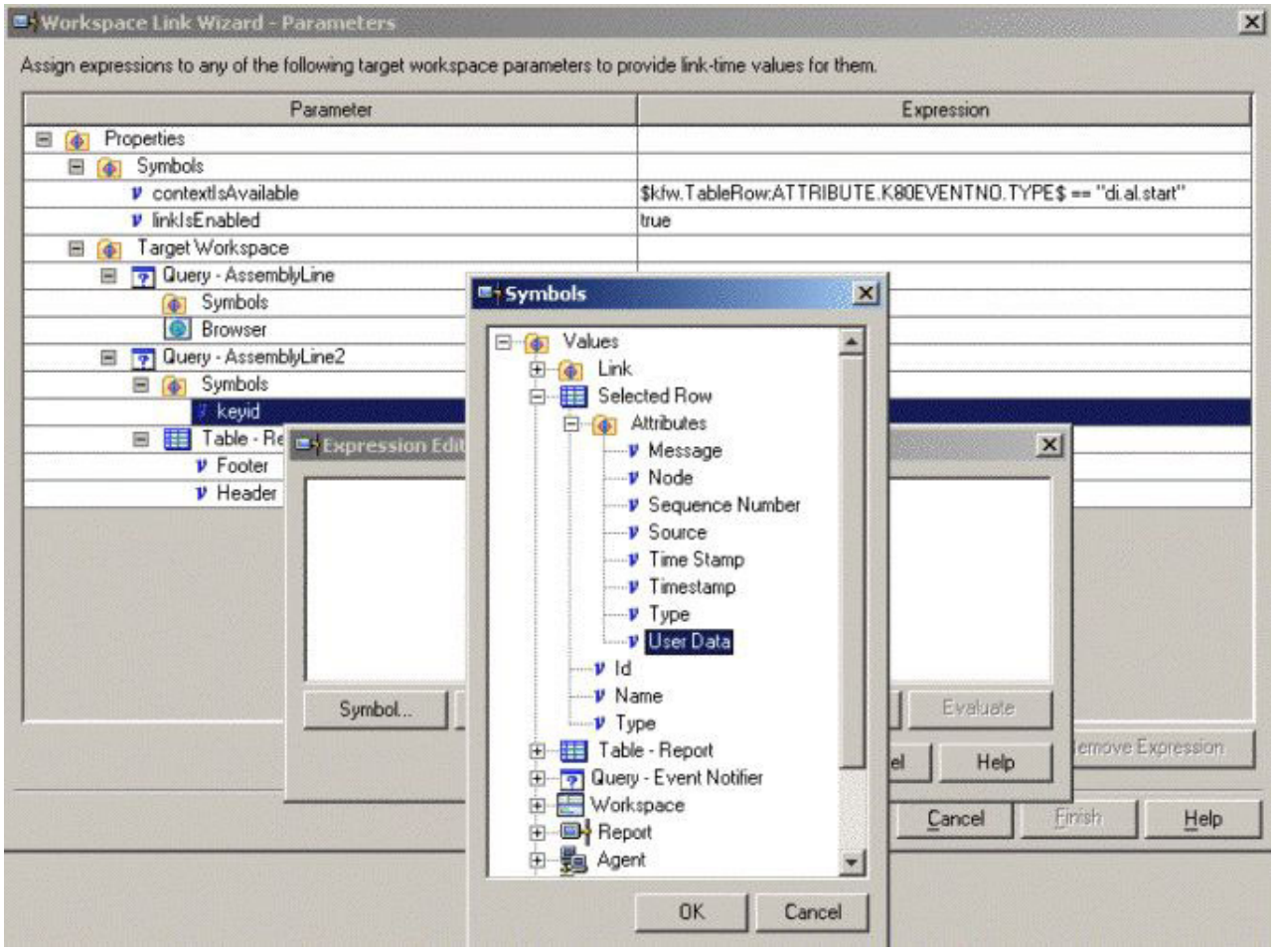


図 25. リンク・ウィザード - ユーザー属性

式エディターで「**OK**」をクリックし、リンク・ウィザード内で「**次へ**」をクリックします。作成されたリンクのサマリーが表示されます。

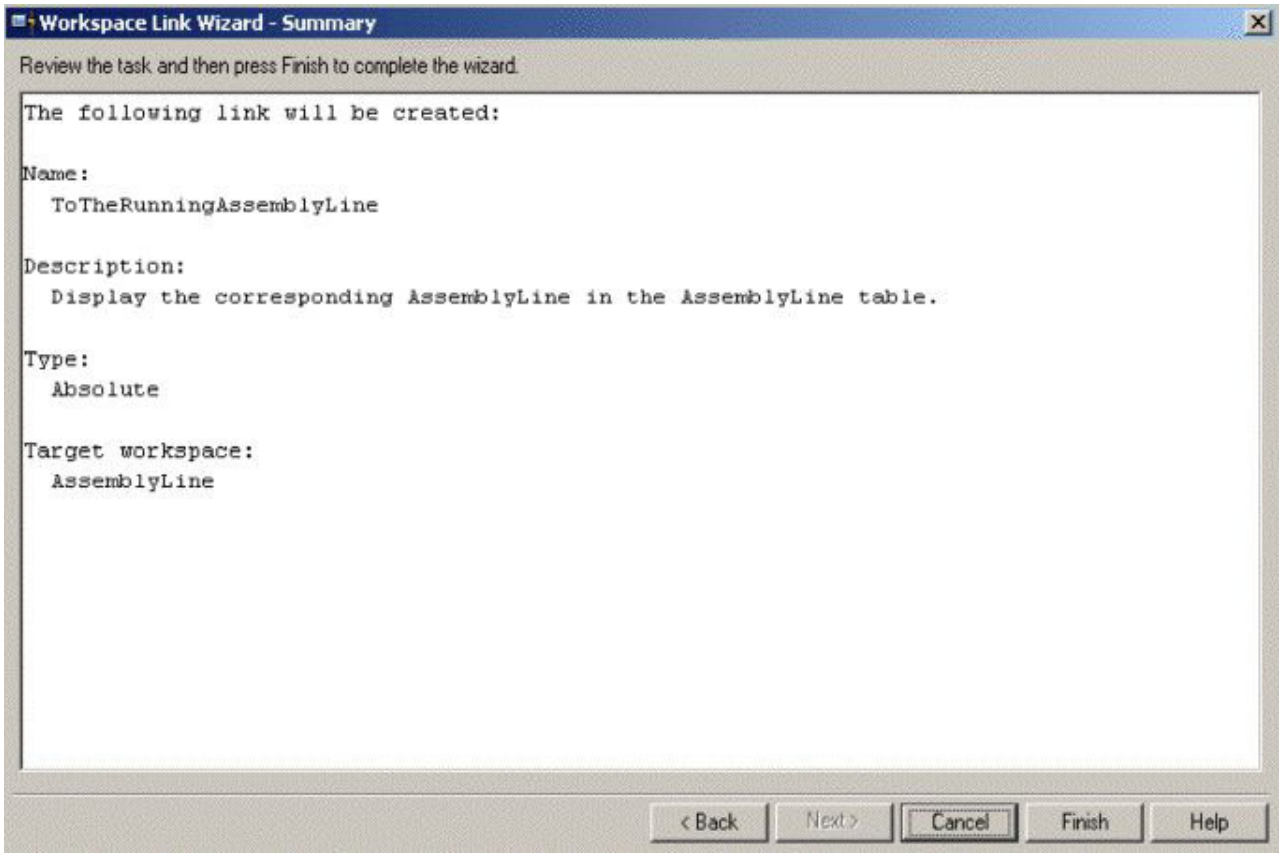


図 26. リンク・ウィザード - サマリー

「**完了**」をクリックしてリンク・ウィザードを閉じます。これで 416 ページの『リンクの目的』セクションの手順を実行する準備ができました。

カスタム通知の ITM への送信

ここで提供されるコードを使用して、カスタム通知を送信するためのスクリプトを記述します。

カスタム通知の送信をデモンストレーションする構成ファイルは、サンプルに同梱されています。このファイルは、*TDI_install_dir*/examples/Tivoli_Monitoring/TDI_Monitored_by_ITM/ custom_notifications.xml にあります。

カスタム通知を送信するには、それを行う独自のスクリプトを記述する必要があります。次のコードはそれをデモンストレーションするものです。

```
session.sendCustomNotification(aType, aId, aData);
```

このコードは、カスタム、ユーザー定義通知をすべての登録済みリスナーに送信します。**aType** パラメーターは、通知タイプです。**aId** は通知 ID です。**aData** は、カスタム・ユーザー・データです。aType は自動的に接頭部「user.」が付くことに注意してください。これは、**myType** タイプの通知を送信した場合に、**user.myType** として受信されることを意味します。

制限

作成された Agent は、IBM Security Directory Integrator サーバーの管理には使用できません。

例えば、AssemblyLines をスタート・ストップすることはできません。これは IBM Security Directory Integrator サーバー JMX レイヤーがそのようなメソッドを公開してもモニタリング目的のみで使用可能です。

OMNIbus を使用した IBM SDI のモニター

OMNIbus を使用した IBM Security Directory Integrator のモニターについては、以下のリンクを参照してください。

概要

OMNIbus の詳細は、「リファレンス」の『EIF コネクター』セクションで読むことができます。

EIF プローブ prop ファイルの構成

EIF プローブが listen するポートを設定することができます。この作業は以下の情報を使用して行います。

EIF プローブが listen するポートが予期しているポートであることを確実にするために、手動で設定することができます。

これを行うには、 *$OMNIHOME*/probes/<arch>/tivoli_eif.props を参照して、**PortNumber** プロパティーの値を EIF プローブが listen するポートの番号に設定します。

デフォルトで、EIF プローブが 600 秒よりも長く非アクティブのままの (イベントを受信しない) 場合、サービスは停止します。**Inactivity** プロパティーの値を 0 に設定することによって、タイムアウトを無限大に設定することができます。

EIF prop ファイルは、次のようになります。

```
# BufferEvents              : "YES"
# HandleMalformedAlarms  : "true"
# EIFCacheFile       : '$OMNIHOME/var/tivoli_eif.cache' (Unix)
# EIFCacheFile       : '%OMNIHOME%¥var¥tivoli_eif.cache' (Windows)
# EventCopies            : 1
  Inactivity             : 0
# MaxEventQueueSize         : 10000
# PortMapper                : "false"
# PortMapperNumber          : 100033057
  PortNumber       : 9998
# Retry                  : "false"
# StreamCapture          : "false"
# StreamCaptureFile      : '$OMNIHOME/var/tivoli_eif.stream' (Unix)
# StreamCaptureFile      : '%OMNIHOME%¥var¥tivoli_eif.stream' (Windows)
```

ただし、EIF プローブ prop ファイルを変更しないと決定した場合、EIF プローブがイベントを listen するデフォルト・ポートは 9999 であることに注意してください (OMNIbus 文書に従う)。

イベントの重大度の決定

イベントの重大度を決定するには、EIF ルール・ファイルを変更します。

イベントの重大度を決定するには、EIF ルール・ファイルにいくつかの変更を行う必要があります。重大度を管理する方法の簡単な説明を示します。この目的のために、**start** イベントを低重大度として定義し、**stop** イベントを高重大度として定義します。その後、ルール・ファイルに次のコードを入力できます。

```
if( regmatch($ClassName, "^.*¥.start$") )
{
 @Severity = 0
}
if( regmatch($ClassName, "^.*¥.stop$") )
{
 @Severity = 4
}
```

カスタム通知はデフォルト重大度 1 を持つことに注意してください。これは、次のような結果になります。

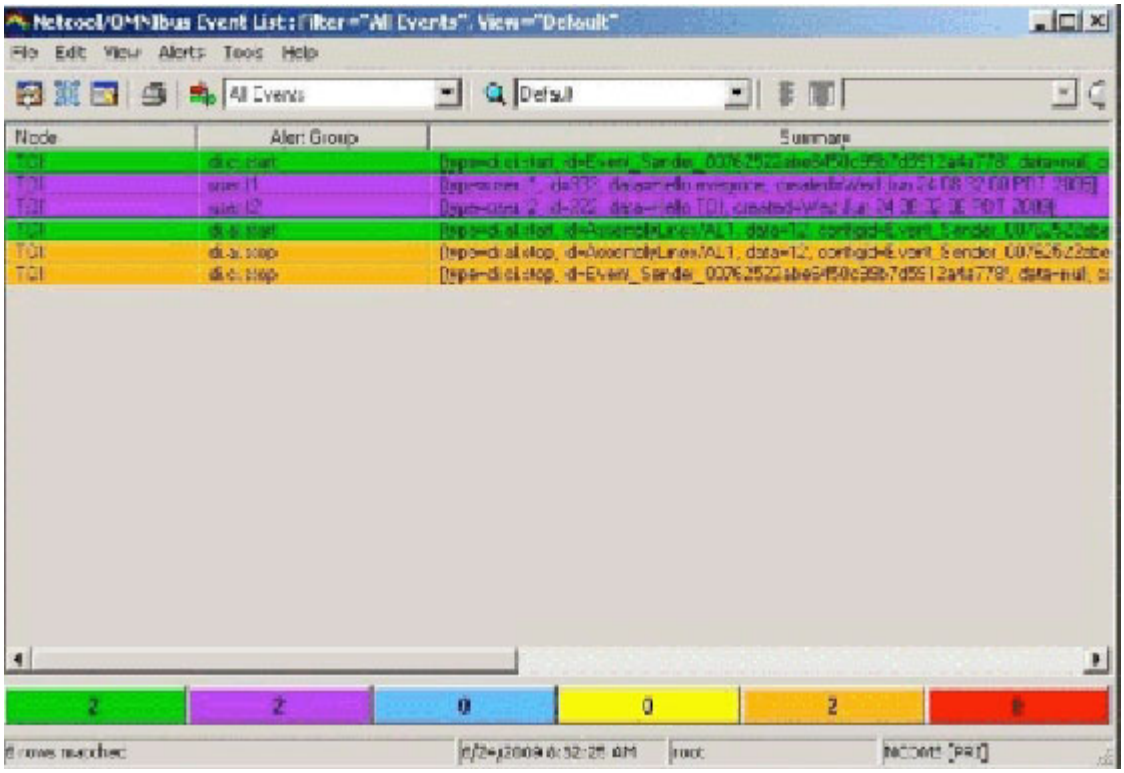


図 27. OMNIbus イベント・リスト

EventPropertyFile.properties ファイルの操作

EventPropertyFile.properties ファイルを使用して、いくつかの作業を行うことができます。詳しくは、以下の情報を参照してください。

EventPropertyFile.properties は、サーバー通知コネクターによって受信可能なイベントのデフォルト・セットを提供します。これは、ユーザーがプロパティー・ファイルのみを使用して AL を構成できるようにします。このプロパティー・ファイルは次の構造を持ちます。

```
key=value
```

key は、イベントのタイプを決定し、*value* は、このイベントが受信されるかどうかを決定します。したがって、主に *true* および *false* 値が使用されます。ただし、*event.customNotifications* キーは、ブール値を期待しません。ここには受信されるカスタム・イベントの名前を設定する必要があります。カスタム通知について詳しくは、『OMNIbus にカスタム通知を送信する』を参照してください。誤解を避けるために、その他にいくつかの考慮すべき事項があります。明確にするために、次のイベントのデフォルト・セットを示した図を参照してください。

```
->event.all
 ->event.ci.all
  ->event.ci.start
  ->event.ci.stop
 ->event.ci.fileUpdated
 ->event.al.all
  ->event.al.start
  ->event.al.stop
 ->event.server.stop
->event.hasCustomNotofications
 ->event.customNotifications
```

示されているように、いくつかのイベントはサブイベントを含みます。イベントを使用可能にすると、すべてのサブイベントが *true* または *false* に設定されているかどうかに関わらず、受信されます。これは、

```
event.ci.all=true
event.ci.stop=false
```

がある場合、event.ci.stop イベントが *false* に設定されていても受信されることを意味します。つまり、event.ci.all は、そのサブイベントに優先されます。ただし、

```
event.ci.all=false
event.ci.stop=true
event.ci.start=false
```

がある場合は、event.ci.stop イベントのみが受信されます。

デフォルトでは、プロパティー・ファイルはすべての IBM Security Directory Integrator サーバー通知を提供するように設定されます。このイベント・セットを変更する場合は、ブール値を *true* (イベントを受信する場合) および *false* (イベントを受信しない場合) に変更する必要があります。つまり、いくつかのコンポーネントの開始を通知するすべてのイベントを受信する場合は、プロパティー・ファイルは次のようになります。

図 28. OMNIbus プロパティー

カスタム通知を受信することを考慮している場合のプロパティー・ファイルの操作については、次の『OMNIbus にカスタム通知を送信する』セクションを参照してください。

OMNIbus にカスタム通知を送信する

OMNIbus に送信される通知をカスタマイズすることができます。この作業を行う場合は、以下にリストされている手順に従ってください。

カスタム通知を受信するには、`event.hasCustomNotofications` を *true* に設定する必要があります。次に、受信するイベント・セットを指定する必要があります。IBM Security Directory Integrator によって送信されるすべてのカスタム・イベントには接頭部 "user." が付くことに注意してください。. これは、*myType* タイプのカスタム・イベントを送信する場合は、次のものを設定する必要があることを意味します。

```
event.customNotifications=user.myType
```

複数のカスタム・イベントを指定するには、";" を使用して分離します。明確にするために、次のシチュエーションを想像してください。
"user.myType1"、"user.myType2"、"user.myType3" タイプのすべてのカスタム通知を受信するとします。テキスト・エディターで開いたプロパティー・ファイルは次のようになります。

```
##Determine if Server Shutdown events are received
event.server.stop=false
##Determine what Custom Notification events are received
##This property is used only if event.hasCustomNotofications is enabled
##Note that all custom notifications are prefixed with "user."
event.customNotifications=user.myType1;user.myType2;user.myType3
##Determine if Custom Notification events are received
event.hasCustomNotifications=true
```

すべてのタイプのカスタム・イベントを受信するには、event.customNotifications 値を "*" に設定する必要があります。これは、Connector が listen するカスタム・イベントのタイプを指定しないため、検出されたカスタム・イベントはすべて取り扱われます。ITM サンプルは、カスタム通知を送信可能な構成を提供します。これはカスタム通知を OMNIbus に送信するためにも使用できます。カスタム通知について詳しくは、423 ページの『カスタム通知の ITM への送信』セクションを参照してください。

付録 C. IBM Security Directory Integrator のアクセシビリティー機能

アクセシビリティー機能は、運動障害や視覚障害など、身体に障害を持つユーザーが情報技術製品を快適に使用できるようにサポートします。

アクセシビリティー機能

IBM Security Directory Integrator のアクセシビリティーの主要機能を以下に示します。

- キーボードだけを使用する操作
- スクリーン・リーダー (読み上げソフトウェア) によって通常使用されるインターフェース
- 触れることで識別できるが、触れるだけではアクティブ化されないキー
- ポートおよびコネクター用の業界標準装置
- 代替入出力装置の接続

IBM Security Directory Integrator の製品資料およびその関連資料はアクセシビリティー機能に対応しています。この製品資料のアクセシビリティー機能については、http://pic.dhe.ibm.com/infocenter/tivihelp/v2r1/topic/com.ibm.iehsc.doc/iehs34_accessibility.html を参照してください。

キーボード・ナビゲーション

この製品では、一般的な Windows の操作 (「ファイル」メニューの表示、コピー、貼り付け、削除操作へのアクセスなど) には、標準の Microsoft Windows ナビゲーション・キーを採用しています。固有の操作には、キーボード・ショートカットを採用しています。すべてのアクションに、必要に応じてキーボード・ショートカットが用意されています。

インターフェースに関する情報

ユーザー・インターフェースおよび資料のアクセシビリティー機能は、以下のとおりです。

- 構成エディターでのフォント、色、およびコントラスト設定の変更手順:
  1. Alt-W キーを押して、構成エディターの「**ウィンドウ**」メニューにアクセスします。下向き矢印キーを使用して、「**設定...**」を選択し、Enter キーを押します。
  2. 「**外観**」タブで「**色とフォント**」設定を選択して、構成エディターの任意の機能領域のフォントを変更します。
  3. 「**ビューおよびエディター・フォルダー**」で構成エディターの色を選択します。色を選択すると、コントラストも変更できます。
- IBM Security Directory Integrator に固有のキーボード・ショートカットのカスタマイズ手順:

1. Alt-W キーを押して、構成エディターの「**ウィンドウ**」メニューにアクセスします。下向き矢印キーを使用して、「**設定...**」を選択します。
2. 下向き矢印キーを使用して、「一般」カテゴリーを選択します。右矢印キーでこのカテゴリーを開き、「**キー**」項目まで下矢印キーを押します。

   「**スキーム**」セレクターの下に「フィルター・テキストの入力」と示されたフィールドがあります。 フィルター・テキスト・フィールドに「security directory integrator」と入力します。固有の IBM Security Directory Integrator ショートカットがすべて表示されます。
3. 選択した任意の IBM Security Directory Integrator コマンドにキーの組み合わせを割り当てます。
4. 「**適用**」をクリックして、変更を永続的に保存します。

構成エディターは、Eclipse ワークベンチに特化したインスタンスです。Eclipse を使用して作成されたアプリケーションのアクセシビリティー機能について詳しくは、http://help.eclipse.org/help33/topic/org.eclipse.platform.doc.user/concepts/accessibility/accessmain.htm を参照してください。

- インフォメーション・センターと関連資料には、JAWS スクリーン・リーダーと IBM ホームページ・リーダーに対応したアクセシビリティー機能が組み込まれています。これらの資料の機能はすべて、マウスの代わりにキーボードから操作できます。

ベンダー・ソフトウェア

IBM Security Directory Integrator には、IBM 使用許諾契約書の対象とならない特定のベンダーのソフトウェアが含まれている場合があります。IBM は、それらの製品のアクセシビリティー機能を保証するものではありません。ベンダー製品のアクセシビリティー情報については、それぞれのベンダーにお問い合わせください。

インストーラーでは、InstallAnywhere 2012 SP1 インストーラー・テクノロジーが使用されています。

関連アクセシビリティー情報

オンライン版の製品資料の代わりとして、ソフトコピーの Adobe Portable Document Format (PDF) 資料も用意しています。PDF 資料は、Adobe Acrobat Reader を使用して表示できます。PDF 資料を使用した場合、オプションで、フォントの拡大およびハイコントラスト表示の設定を使用でき、また、キーボードだけでナビゲートできます。ただし、この場合、スクリーン・リーダー (読み上げソフトウェア) のユーザーには、代替テキストは提供されません。

IBM Security Directory Integrator の PDF 資料は、 IBM Security Directory Integrator 資料からアクセスしてダウンロードすることができます。

IBM とアクセシビリティー

アクセシビリティーに対する IBM の取り組みについて詳しくは、『IBM Human Ability and Accessibility Center』を参照してください。

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

以下の保証は、国または地域の法律に沿わない場合は、適用されません。

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 Corporation
2Z4A/101
11400 Burnet Road
Austin, TX 78758 U.S.A.

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。

この文書に含まれるいかなるパフォーマンス・データも、管理環境下で決定されたものです。そのため、他の操作環境で得られた結果は、異なる可能性があります。一部の測定が、開発レベルのシステムで行われた可能性がありますが、その測定値が、一般に利用可能なシステムのものと同じである保証はありません。さらに、一部の測定値が、推定値である可能性があります。実際の結果は、異なる可能性があります。お客様は、お客様の特定の環境に適したデータを確かめる必要があります。

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 の将来の方向性および指針に関するすべての記述は、予告なく変更または撤回される場合があります。これらは目標および目的を提示するものにすぎません。

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。お客様は、IBM 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。

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的創作物にも、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_. All rights reserved.

この情報をソフトコピーでご覧になっている場合は、写真やカラーの図表は表示されない場合があります。

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 Corporation 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。

Adobe、PostScript は、Adobe Systems Incorporated の米国およびその他の国における登録商標または商標です。

IT Infrastructure Library は英国 Office of Government Commerce の一部である the Central Computer and Telecommunications Agency の登録商標です。

インテル、Intel、Intel ロゴ、Intel Inside、Intel Inside ロゴ、Centrino、Intel Centrino ロゴ、Celeron、Xeon、Intel SpeedStep、Itanium、Pentium は、Intel Corporation または子会社の米国およびその他の国における商標または登録商標です。

Linux は、Linus Torvalds の米国およびその他の国における登録商標です。

Microsoft、Windows、Windows NT および Windows ロゴは、Microsoft Corporation の米国およびその他の国における商標です。

ITIL は英国 The Minister for the Cabinet Office の登録商標および共同体登録商標であって、米国特許商標庁にて登録されています。

UNIX は The Open Group の米国およびその他の国における登録商標です。

Java およびすべての Java 関連の商標およびロゴは Oracle やその関連会社の米国およびその他の国における商標または登録商標です。

Cell Broadband Engine は、Sony Computer Entertainment, Inc.の米国およびその他の国における商標であり、同社の許諾を受けて使用しています。

Linear Tape-Open、LTO、LTO ロゴ、Ultrium、および Ultrium ロゴは、HP、IBM Corp. および Quantum の米国およびその他の国における商標です。

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]

[マ行]

[ヤ行]

[ラ行]

A

C

D

E

F

H

I

J

K

Printed in Japan

SC88-8415-03

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