ACTIAN™

Installation Toolkit Handbook

How to Integrate Zen Program Installations into Your Own

Zen v16

Activate Your Data™

目次

このドキュメントについて

このドキュメントには、Zen 製品のインストレーションを別のアプリケーション インストレーションに組み込む方法について、開発者向けの情報が記載されています。

Zen インストールのカスタマイズ

以下のトピックでは、Windows システムへの Zen インストールをカスタマイズするための概念と手順について説明します。このインストールをカスタマイズすれば、貴社のアプリケーションに Zen 製品のすべてまたは一部をバンドルすることができます。

カスタム インストールの概要

ここでは、Zen 製品のインストールのために使用されるテクノロジおよび各種設定のカスタマイズについて説明します。Windows システムの場合、Zen のインストールは Microsoft Installer（MSI）を使用します。PTKSetup.ini ファイルには、カスタム インストール用に変更できるデフォルトの設定が含まれています。

インストーラー実行可能ファイル

ほとんどのインストール シナリオでは、インストールにインストーラー実行可能ファイルを使用する必要があります。このインストーラーは InstallShield パッケージで、インストールを実行する前にさまざまなチェックを行います。Windows が 32 ビット版か 64 ビット版かの検出、適切なインストールの起動、およびシステムに適合するすべての 32 ビット /64 ビット クライアント コンポーネントを自動的に提供します。

Zen インストレーションを別のソフトウェア パッケージとバンドルする、または Zen のインストレーションのスクリプトを作成して別のソフトウェアに組み込む場合は、インストーラー実行可能ファイルを**使用しない**ようにしてください。それらのシナリオの場合、Zen をインストールする詳しい手順については、アプリケーションへの Zen の組み込みおよびサイレント インストールの使用のトピックを参照してください。

次の表は Windows オペレーティング システムでの Zen インストーラーについて説明しています。

製品	インストール パッケージ	説明
Enterprise Server	Install_Zen_EnterpriseServer.exe	• 32 ビット オペレーティング システムへ 32 ビット エンジンをインストール • 64 ビット オペレーティング システムへ 64 ビット エンジンをインストール • すべてのクライアント コンポーネントをインストール
Cloud Server	Install_Zen_CloudServer.exe	• 32 ビット オペレーティング システムへ 32 ビット エンジンをインストール • 64 ビット オペレーティング システムへ 64 ビット エンジンをインストール • すべてのクライアント コンポーネントをインストール

製品	インストール パッケージ	説明
Workgroup	Install_Zen_Workgroup_Engine.exe	• 32 ビットおよび 64 ビットの両オペレーティング システムで 32 ビット エンジンをインストール • すべてのクライアント コンポーネントをインストール
Client	Install_Zen_Client.exe	• 32 ビット オペレーティング システムで 32 ビット コンポーネントをインストール • 64 ビット システムで 32 ビットおよび 64 ビットの両コンポーネントをインストール
Client Reporting Engine	Install_Zen_Reporting_Engine.exe	• 64 ビット エンジンのみをインストール • すべてのクライアント コンポーネントをインストール

PTKSetup.ini 設定

PTKSetup.ini には、標準的なインストールに必要な設定がすべて含まれています。このファイルは、Zen 製品ごとに使用されます。インストール メディアの場合、このファイルは .msi ファイルと同じフォルダーにあります。

PTKSetup.ini は、次の 2 つのパートに分かれています。

- [PROPERTIES] セクション - 各インストール時に使用される設定が含まれています。
- [Registry Migration] セクション - PSQL v10 より前のバージョンの Zen で使用された設定が含まれています。

各設定には、標準的なインストール用のデフォルトとなる現在値があります。ファイル内のコメント行では、各設定の説明と指定可能な値が記載されています。

注意！ 組み込む製品エディションに付属している固有の PTKSetup.ini ファイルを使用する必要があります。インストーラー技術およびインストール設定はバージョンごとに異なる可能性があるため、このファイルは製品と一致していなければなりません。

[PROPERTIES] セクション

次の表では、PTKSetup.ini の［PROPERTIES］セクションにあるキーを列挙しています。各キーはカテゴリ別に分類されています。各設定の値はキーに含まれています。

カテゴリ	キー
Setup Type	PVSW_PSQL_INSTALL_TYPE
Destination Folder (カスタム セットアップのみ)	PVSW_PSQL_SKIP_INSTALLDIR
Directory Locations (カスタム セットアップのみ)	PVSW_PSQL_DATADIR PVSW_PSQL_INSTDIR32 PVSW_PSQL_INSTDIR64 PVSW_PSQL_ARCHIVE_DIR
Registration Page	PVSW_PSQL_UI_NO_REGISTER PVSW_OEM_REGISTER_HTML
License (ワークグループ エンジンまたはサーバー エンジンのみ)	PVSW_PSQL_LICENSE_KEY PVSW_PSQL_SKIP_LICENSE
Engine as Service or Application (ワークグループ エンジンまたはキャッシュ エンジンのみ)	PVSW_RUN_CCE_AS_SVC PVSW_RUN_WGE_AS_SVC PVSW_APP_WAIT_TIME
オプションの機能 - カスタム セットアップのみ	
Data Access Feature	PVSW_PSQL_INSTALL_ADONET PVSW_PSQL_INSTALL_BTRBOX PVSW_PSQL_INSTALL_DTO PVSW_PSQL_INSTALL_JCL PVSW_PSQL_INSTALL_JDBC PVSW_PSQL_INSTALL_PDAC
Java Utilities Feature	PVSW_PSQL_INSTALL_DDFB PVSW_PSQL_INSTALL_PCC PVSW_PSQL_INSTALL_DOCUMENTATION PVSW_PSQL_INSTALL_NOTIFYVIEWER PVSW_PSQL_INSTALL_JAVAUTILS

カテゴリ	キー
Other Utility Feature	PVSW_PSQL_INSTALL_CBOL PVSW_PSQL_INSTALL_COREUTILS PVSW_PSQL_INSTALL_PSA

[Registry Migration] セクション

PTKSetup.ini ファイルの［Registry Migration］セクションは、PSQL v9 以前からアップグレードする場合に使用できます。このセクションの設定によって、選択したレジストリ値を古いバージョンから新しいバージョンへ移行することができます。

各設定の詳細については、『*Advanced Operations Guide*』で説明しています。

レジストリ移行設定の書式

[Registry Migration] セクションでは、各設定について、Zen の以前のバージョンの設定が等号（=）の左側に記載され、等号の右側に Zen の新しいバージョン用の設定が記載されています。

レジストリ移行設定で使用される書式は < **古いバージョンの設定** >=< **新しいバージョンの設定** > です。

レジストリ移行設定の使用

インストール時に、v10 より前のバージョンの Zen から移行しない設定についてはコメント行にします。移行されるキーが既に存在する場合、古いレジストリ設定がそれを置き換えます。

メモ： PSQL v10 以上のバージョンをアップグレードする場合、Zen のインストールではこのセクションを無視し、すべてのレジストリ設定を自動的に移行します。

現在の設定を確認するには **bcfg** ユーティリティ使用します。このユーティリティの説明については『*Advanced Operations Guide*』を参照してください。このユーティリティを使用すれば、現在の設定を取得して確認することができるので、設定の変更が必要かどうかを検討するのに役立ちます。

Zen のインストール実行後の設定

Zen のインストール後、Distributed Tuning Interface（DTI）で使用可能な関数を使用して、お使いのデータベース エンジンの設定やチューニングを行うことができます。DTI

の関数を使用すれば、現在の設定を確認したりプログラムから設定を変更したりするなど、さまざまなタスクを実行することができます。

データベース エンジンの設定を行うための DTI 関数の使用に関する詳細については、『*Distributed Tuning Interface Guide*』の「DTI 関数グループ」セクションにある関数の一覧を参照してください。

Zen DTO オブジェクトは、DTI インターフェイスと同様、Zen 設定機能を果たすための COM インターフェイスです。

システム パフォーマンスの向上

ワークステーションを復旧するための方法として再イメージ化やその他の技術を採用し、「システムの復元」を使用しない企業の環境では、インストーラー動作用の「システムの復元」を無効にすることによって、復元ポイントを作成するための時間やディスク スペースが不要になりパフォーマンスを向上させることができます。「システムの復元」を無効にするには、Microsoft のヘルプ システムで詳しい情報を参照してください。

この設定はインストーラーで開始する復元動作のみに影響します。またこの設定はグループ ポリシーで使用することができ、各ワークステーションへの配置を支援します。

メモ：「システムの復元」は非常に重要な Windows の機能であるため、ほとんどの状況において、この機能を無効にすることはお勧めできません。「システムの復元」の無効を適用できるのは、企業のシナリオとして、この機能を使用しないとしている場合のみです。

インストール内容のカスタマイズ

Zen でインストールされるものは、以下のいずれかの方法を使用することで変更できます。

- PTKSetup.ini 設定の使用
- CAB ファイルを使用したインストール パッケージ サイズと各種機能の変更
- アプリケーションへの Zen の組み込み

PTKSetup.ini 設定の使用

PTKSetup.ini で、プロパティの行のコメントを解除し、そのプロパティに新しい値を設定すると、コマンド ラインからインストーラーを実行するだけで、それらのプロパティを使用することができます。

メモ： コマンド ラインで MSI プロパティが含まれていても、PTKSetup.ini でそのプロパティのコメントが解除され、値が設定されると、PTKSetup.ini での値が使用されます。

PTKSetup.ini で設定できるプロパティについては PTKSetup.ini 設定で説明しています。

Zen キーの指定

PSQL v11 より前のバージョンでは、Zen 認証キーは PTKSetup.ini ファイルで属性として指定できました。そのため、複数のインストールで同じキーを使用できました。PSQL v11 以降は、インストールごとに一意のキーが必要となります。現在も PTKSetup.ini にキーを指定することはできますが、そのキーはインストールを実行するマシンごとに一意でなければなりません。

キーの配布にはさまざまな手段が用いられます。たとえば、製品のパッケージ上にキーを印刷する、箱の中に同梱する、エンド ユーザーへの電子メールに記載するなどの手段があります。キーの配布にどのような方法を用いたとしても、PTKSetup.ini におけるキーの一意性を保証する 1 つの方法は、アプリケーションのインストールでエンド ユーザーにキーの入力を促すために用います。これで、アプリケーションのインストールで PTKSetup.ini のローカル コピーにそのキーを書き込むことができ、PTKSetup.ini を使用するインストール処理を続行します。PTKSetup.ini に有効なキーが指定されている場合、Zen インストール処理でそのキーが自動的に認証されます。キーが有効であると

いうことは、インストールが実行されるマシンに対してキーが一意であるということです。

インストール実行可能ファイルで別の設定ファイルの場所を使用する

インストーラー実行可能ファイルを実行するとき、コマンド ラインで、MSI のパブリック プロパティである PVSW_CFG_FILE を設定し、完全に修飾されたファイル パス名を渡すことによって、別の設定ファイルを使用することができます。このプロパティとその値は、基になる msiexec Windows プロセスに渡されます。

このプロパティを渡すことによって、その値にスペースが含まれているかどうかを判定します。スペースが含まれていない場合、コマンドは以下のようになります。

```
Install_<product>.exe /v"PVSW_CFG_FILE=c:¥temp¥ConfigFile.ini"
```

メモ： /v の後に指定されるプロパティはすべて msiexec へ渡されます。**/v** オプションの後にスペースはありません。渡されるプロパティの文字列は二重引用符で囲む必要があります。プロパティの 1 つがスペースを含むパス名である場合は、パスも二重引用符で囲む必要があります。また、次に示すように、引用符は両方ともバックスラッシュでエスケープ（¥"）する必要があります。

```
Install_<product>.exe /v"PVSW_CFG_FILE=¥"c:¥temp¥My Folder¥ConfigFile.ini¥""
```

ここで示す例を組み合わせて使用すれば、複数のプロパティ設定を渡すことができます。

```
Install_<product>.exe /v"PROPERTY1=0 PROPERTY2=¥"c:¥My Path¥File.txt¥""
```

Zen MSI をインストールするための前提条件も参照してください。

コマンド ライン プロパティを使用して Zen をマニュアルなしでインストールするには

1. MSI の前提条件を満たしていることを確認してください。Zen MSI をインストールするための前提条件を参照してください。
2. コマンド プロンプトで次のコマンドを入力します。

   ```
   Install_<product>.exe /v"/l*v ¥"%temp%¥Zen_<version>_Install.log¥"
   PVSW_PSQL_INSTALL_DOCUMENTATION=0"
   ```

3. Enter キーを押します。

   インストール プログラムはマニュアル オプションなしで対話的に実行します。

コマンド ライン プロパティを使用して、キー認証と共に Zen をインストールするには

1. MSI の前提条件を満たしていることを確認してください。Zen MSI をインストールするための前提条件を参照してください。
2. コマンド プロンプトで、インストールを実行するシステムに対する一意のキーを使用して、以下のコマンドを入力します。

```
Install_<product>.exe /v"/l*v ¥"%temp%¥Zen_<version>_Install.log¥" PVSW_PSQL_LICENSE_KEY=NG2ZE-ZKS3C-D2CFK-9IR6K-G2C3X"
```

メモ： **/v** の後に指定されるプロパティはすべて msiexec へ渡されます。**/v** オプションの後にスペースはありません。渡されるプロパティの文字列は二重引用符で囲む必要があります。プロパティの 1 つがスペースを含むパス名である場合は、パスも二重引用符で囲む必要があります。また、上記の例で示したように、引用符は両方ともバックスラッシュでエスケープ（**¥"**）する必要があります。

CAB ファイルを使用したインストール パッケージ サイズと各種機能の変更

Zen インストールにはキャビネット（CAB）ファイルが含まれています。このファイルはご自身のインストール パッケージから削除することによってインストール パッケージの全体のサイズを削減したり、不要なファイルを使用しないようにすることができます。これらのファイルを削除すると、カスタム インストール時にグラフィカル ユーザー インターフェイスで使用可能な機能のリストから該当する機能を自動的に削除します。

ここでは、以下の項目について説明します。

- CAB ファイルを使った作業と機能のアップデート
- 必要な CAB ファイル
- CAB ファイル以外で必要なファイル
- 任意の CAB ファイル
- オプション機能のキャビネット ファイルの要件
- CAB ファイル インストールの例

CAB ファイルを使った作業と機能のアップデート

CAB ファイルは、MSI と同じディレクトリにあります。先頭にアンダースコア（_）が付く CAB ファイルがそのインストール タイプに必要なものです。

CAB ファイルのコア セットは変わりません。ただし、Zen のアップデートには、製品のメジャー バージョンでリリースされているオリジナル セットから追加された CAB ファイルが含まれており、新たに追加されたコンポーネントや機能に対応します。インストール タイプに対応する最新ファイルと、必要なオプション機能を確認して入手してください。

必要な CAB ファイル

必要な CAB ファイルは、先頭にアンダースコアを付けた名前で指定されます。次の表は、各インストール パッケージ タイプに必要なキャビネット ファイルの一覧を示します。

キャビネット ファイル	Enterprise Server	Cloud Server	Client	Workgroup	Reporting
_BCW.cab			X	X	
_C32_64b.cab	X	X	X	X	
_CE32_64.cab			X		
_CmnEn64.cab	X	X			X
_DbEng32.cab	X	X		X	X
_DbEng.cab	X	X		X	X
_DRM64.cab	X	X		X	X
_PSQL.cab	X	X	X	X	X
_PSQL64.cab	X	X	X	X	X
_RpEng64.cab					X
_Srvr32.cab	X	X			
_Srvr64.cab	X	X			
_WGE.cab				X	

CAB ファイル以外で必要なファイル

必要な CAB ファイルで挙げたファイルに加え、カスタム インストールでは、インストーラー実行可能ファイルを実行するフォルダーにある Data フォルダーに以下の項目が存在していなければなりません。

- "409" フォルダーとそのフォルダー内の全ファイル
- "411" フォルダーとそのフォルダー内の全ファイル
- "Data" フォルダーとそのフォルダー内の全ファイル。ただし、次のセクションで記載されるオプション .cab ファイルは除きます。
- "ISSetupPrerequisites" フォルダーとそのフォルダー内の全ファイル
- Setup*Product*32_x86.exe
- Setup*Product*64_x64.exe

  Product 部分は EnterpriseServer、CloudServer、Client、Workgroup、または Reporting のいずれかに置き換えられますが、特殊なケースが 2 つあります。

  - Workgroup の場合、64 ビットのセットアップ ファイルは SetupWorkgroup32_x64.exe となります。
  - Client Reporting Engine の場合は、64 ビット ファイルの SetupReporting64_x64.exe のみが使用されます。

任意の CAB ファイル

次の表は、インストール パッケージ タイプ別に適用されるオプション機能の一覧を示します。オプション機能をインストールする場合は、そのオプション機能を、前のトピックで説明した Data フォルダー内に置いておく必要があります。これらの機能の詳細については、『*Getting Started with Zen*』の Zen のオプション機能を参照してください。

キャビネット ファイル	Enterprise Server	Cloud Server	Client	Workgroup [1]	Reporting
ADONET.cab	X	X	X	X	X
BtreveDos.cab	X	X	X	X	X
CSE32Cmn.cab	X	X	X	X	X

キャビネット ファイル	Enterprise Server	Cloud Server	Client	Workgroup [1]	Reporting
CSE32Eng.cab	X	X		X	
CSE64Eng.cab	X	X			X
CSECmEng.cab	X	X		X	X
DDFB.cab	X	X	X	X	X
Docs.cab	X	X	X	X	X
DTO.cab DTO64.cab	X	X	X	X	X
EclipRCP.cab	X	X	X	X	X
JCL.cab JCL64.cab	X	X	X	X	X
JDBC.cab JDBC64.cab	X	X	X	X	X
JRE.cab	X	X	X	X	X
JreUtils.cab	X	X	X	X	X
NVUtil.cab	X	X	X	X	X
PDAC.cab PDAC64.cab	X	X	X	X	X
PSA.cab	X	X	X	X	X
Utils.cab	X	X	X	X	X
Utils2.cab	X	X	X	X	X
WPMCS32.cab WPMCS64.cab	X	X			
WPMCS64R.cab					X
ZenCC.cab	X	X	X	X	X

[1] Zen Workgroup は 32 ビット アプリケーションです。しかし、32 ビットと 64 ビットのオペレーティング システム用に別個のインストール パッケージが提供されます。

オプション機能のキャビネット ファイルの要件

一部のオプション機能では、その機能を有効にするために別のコンポーネントが必要な場合があります。以下の表には、別のコンポーネントを必要とするオプション機能を一覧表示します。

	Cobol Schema Executor	DDF Builder	ZenCC[1]	System Analyzer	ユーザー向けドキュメント	Notification Viewer	その他のユーティリティ[2]
CSE32Cmn.cab	X[3]						
CSE32Eng.cab	X[4]						
CSE64Eng.cab	X[5]						
CSECmEng.cab	X[6]						
DDFB.cab		X					
Docs.cab					X		
EclipRCP.cab		X	X		X	X	X
JRE.cab		X	X		X	X	X
JreUtils.cab							X
NVUtil.cab						X	
PSA.cab				X			
Utils2.cab							X
Utils.cab				X			X
ZenCC.cab			X		X		

[1] Zen Control Center

[2] この表に記載されていないすべてのユーティリティ。手動認証ウィザード、bcfg、および bmon などの Java を必要とするほかのユーティリティでは、EclipRCP.cab、JRE.cab、および JreUtils.cab が必要です。License Administrator や Function Executor などの一般的なユーティリティ（非 Java）では Utils2.cab および Utils.cab が必要です。

[3] Client、Workgroup、Enterprise Server、および Cloud Server に必要です。

[4] Workgroup、Enterprise Server、および Cloud Server に必要です。

[5] Enterprise Server に必要です。

[6] Workgroup、Enterprise Server、および Cloud Server に必要です。

CAB ファイル インストールの例

CAB ファイルを使用したインストールのカスタマイズの例を以下に示します。この例は、ワークグループ エンジンのインストールに、唯一のオプション機能としてドキュメントを加えます。

メモ： ドキュメントには、Eclipse フレームワークと Zen Control Center をインストールする必要があります。

ワークグループ エンジンとドキュメントをパッケージするには

次の例は、Windows オペレーティング システムで実行しているシステムを対象としています。

1. ワークグループ エンジン用に必要な CAB ファイルを選択します（必要な CAB ファイルを参照）。これらのファイルは以下のとおりです。
   - _BCW.cab
   - _C32_64b.cab
   - _DbEng.cab
   - _DbEng32.cab
   - _DRM64.cab
   - _PSQL.cab
   - _PSQL64.cab
   - _WGE.cab
2. ドキュメント用のオプション機能 CAB ファイルを選択します（オプション機能のキャビネット ファイルの要件を参照）。これらのファイルは以下のとおりです。
   - Docs.cab
   - EclipRCP.cab
   - ZenCC.cab
   - JRE.cab

上記の 2 つの手順で挙げたファイルは、ワークグループ エンジンとドキュメントをインストールするために必要な CAB ファイルです。その他の CAB ファイルはインストール パッケージから削除しても構いません。CAB ファイル以外で必要なファイルで

説明しているすべての非 CAB ファイルも、インストール パッケージに入っている必要があります。

アプリケーションへの Zen の組み込み

アプリケーションに Zen を組み込むには、Windows インストーラーを使用することをお勧めします。当初、これは Microsoft Installer と呼ばれていたものです。この略語は MSI でファイル拡張子は .msi です。

ここでは、以下の項目について説明します。

- Zen MSI をインストールするための前提条件
- Zen インストーラーと関連 MSI ファイル
- Microsoft 変換ファイル（MST）を使った Zen の組み込み
- MSI オプションと PTKSetup.ini 設定の組み合わせ

Zen MSI をインストールするための前提条件

Zen MSI のインストールには以下の状況が必要です。

- Windows 8.1 または Server 2012 R2 の場合：Windows 更新プログラム 3118401 を適用します。
- ターゲット システムには Zen でサポートされる Windows インストーラーの最小バージョン Windows Installer 5.0 が含まれている必要があります。お使いのシステムに Windows インストーラーのどのバージョンがあるかを調べるには、コマンド プロンプトで「`msiexec /?`」を実行します。インストール先のシステムに MSI 5.0 以上があるかどうかが不明な場合は、**/v** オプションを用いたインストール実行可能ファイルを使用してインストールしてください。
- インストーラー実行可能ファイルを実行せず、msiexec コマンド ラインを使用して Zen をインストールするには、すべての必須コンポーネントがインストールされており、プロパティと値 `PSQL_PREREQS_INSTALLED=Y` が msiexec コマンド ラインに含まれている必要があります。
- Windows Vista 以上のエディションでは、MSI ファイルを用いた Zen インストールを実行する前に、Zen インストーラーを呼び出す処理が、システム特権を持つユーザー資格情報で実行されている必要があります。インストーラーは、システム特権を持つユーザー アカウントとして実行しなければなりません。

- PSQL v11 SP3 からのアップグレードを予定し、かつ現在のバージョンが 11.30.000 から 11.31.050 の場合には、修正プログラム PSQLv11.31.049.msp（インストール フォルダー "¥ISSetupPrerequisites{1BCDC56D-D2CC-4125-8B24-4D249AFBFFE7}¥" にあります）を適用する必要があります。そうしないと、v12 インストールで PSQL v11 SP3 が削除されるときにエラー メッセージが表示されます。このエラー メッセージは、アップグレード インストールがサイレント実行されている場合でも表示されます。あるいは、アップグレードする前に、利用可能な最新の PSQL v11 SP3 修正プログラムを適用することもできます。
- PSQL v12 Server、Vx Server、または Workgroup からのアップグレードを予定し、かつ現在のバージョンが 12.00.000 から 12.01.066 の場合には、アップグレード前に修正プログラム PSQLv12.01.068.msp（インストール フォルダー "¥ISSetupPrerequisites¥{D61586D9-7EFA-4ED8-8A47-6E569A92D4D9}" にあります）を適用する必要があります。この修正プログラムを適用しないと、アップグレードによってデータ フォルダー defaultdb が削除されます。あるいは、アップグレードする前に、利用可能な最新の PSQL v12.01 修正プログラムを適用することもできます。
- 64 ビット版の Windows にアップグレードをインストールするには、その前にまず、64 ビット Microsoft Visual C++ 2019 Runtime Libraries 以降（インストール フォルダー "¥ISSetupPrerequisites¥{2F044C80-608C-4274-AB1D-96D67E047307}" にあります）をインストールする必要があります。
- 32 ビット版と 64 ビット版両方の Windows にアップグレードをインストールするには、その前にまず、32 ビット Microsoft Visual C++ 2019 Runtime Libraries 以降（インストール フォルダー "¥ISSetupPrerequisites{7BB553E0-BAA5-4184-965C-AEEB89B82D46}" にあります）をインストールする必要があります。

コマンド プロンプトで `msiexec /?` を実行して表示される構文やオプションに加え、MSI テクノロジについては Microsoft の Web サイトに詳しい説明があります。

Zen インストーラーと関連 MSI ファイル

次の表は、Zen 製品とそれぞれの .MSI ファイルの一覧です。

製品	MSI ファイル
Enterprise Server	ActianZenv16EnterpriseServer64_x64.msi ActianZenv16EnterpriseServer32_x86.msi

製品	MSI ファイル
Cloud Server	ActianZenv16CloudServer64_x64.msi ActianZenv16CloudServer32_x86.msi
Workgroup	ActianZenv16WGE32_x64.msi ActianZenv16WGE32_x86.msi Workgroup は 32 ビット アプリケーションです。しかし、必要なクライアント コンポーネントをインストールするために、32 ビットと 64 ビットのオペレーティング システム用に別個のインストール パッケージが提供されます。
Client	ActianZenv16Client64_x64.msi ActianZenv16Client32_x86.msi
Reporting Engine	ActianZenv16Reporting64_x64.msi

Microsoft 変換ファイル（MST）を使った Zen の組み込み

Microsoft 変換ファイル（MST）では、組み込みインストールをカスタマイズする方法を提供します。.mst ファイル（変換ファイル）は、インストール時に使用する変更セットです。インストールをカスタマイズするために変換ファイルを使用すれば、柔軟な機能選択や容易な製品アップデートが可能となります。

MSI への変換ファイルを生成するには、インストール時に必要とされるアプリケーション情報を格納するインストーラー データベースを使用して作業する必要があります。変換ファイルの生成には、Windows プラットフォーム SDK の InstallShield、Microsoft Orca または MSI ユーティリティなどのツールを使用することができます。

例：ショートカットなしの組み込みインストールの作成

変換ファイルを使用する一般的なケースは、Zen インストールの組み込みです。これは［スタート］メニューまたはデスクトップ上にショートカットを作成しません。以下の手順ではこのカスタマイズの例を提供します。

1. 元の Zen .msi ファイルの複製を作成します。
2. 使用するツールで、複製した .msi を開きます。
3. Shortcut テーブルの［Action］列で、InstallExecuteSequence レコードにおける CreateShortcuts 値を変更します。

   当初

```
CreateShortcuts   <null_value>  <sequence_number>
```

修正

```
CreateShortcuts   NOSHORTCUTS   <sequence_number>
```

未定義のプロパティ名 NOSHORTCUTS によって CreateShortcuts アクションが省略されることになります。

4. ツールを使用し、元の .msi ファイルと修正した複製ファイルを比較して新たに .mst ファイルを生成します。
5. 生成した変換ファイルに MyTransform.MST などの名前を付けます。
6. その変換ファイルをインストール ファイルに加え、TRANSFORMS= オプションをコマンド ラインに追加します。この場合は、TRANSFORMS=MyTransform.mst となります。

これらの手順に基づくインストール例を次に示します。

```
msiexec.exe /i { パス }¥ActianZenv16<product-type><platform>.msi /qn REBOOT=ReallySuppress /l*v
"%temp%¥Zen_<version>_Install.log" TRANSFORMS=MyTransform.mst PSQL_PREREQS_INSTALLED=Y
```

MSI オプションと PTKSetup.ini 設定の組み合わせ

MSI オプションとパラメーター、PTKSetup.ini ファイルの設定を Zen インストールで組み合わせることができます。これを行うには、PTKSetup.ini ファイルで対象のプロパティを必ずコメント アウトし、無効にしておいてください。その後、.ini ファイルで無効にしたオプションを使用してコマンド ラインからインストールを実行します。

次の例では、前述の例で使用した方法を組み合わせています。

インストールから Zen ドキュメントを取り除くには

1. PTKSetup.ini ファイルで、以下のプロパティを探します。

```
;PVSW_PSQL_INSTALL_DOCUMENTATION=1
```

2. 次のように、セミコロンを削除し、このプロパティの値を 0 に変更します。

```
PVSW_PSQL_INSTALL_DOCUMENTATION=0
```

3. MSI の前提条件を満たしていることを確認してください。Zen MSI をインストールするための前提条件を参照してください。
4. コマンド プロンプトで次のコマンドを実行します。

```
msiexec.exe /i <ActianZenProductName>.msi /qn PSQL_PREREQS_INSTALLED=Y TRANSFORMS=1041.mst
```

インストールに Zen キーの認証を含めるには

1. PTKSetup.ini ファイルで、以下のプロパティを探します。

```
;PVSW_PSQL_LICENSE_KEY=
```

2. 次のように、セミコロンを削除し、このプロパティの値を、インストールが実行されるマシンに対する一意のキーに変更します

```
PVSW_PSQL_LICENSE_KEY=NG2ZE-ZKS3C-D2CFK-9IR6K-G2C3X
```

   Zen キーの指定も参照してください。

3. MSI の前提条件を満たしていることを確認してください。Zen MSI をインストールするための前提条件を参照してください。

4. コマンド プロンプトで次のコマンドを実行します。

```
msiexec.exe /i <ActianZenProductName>.msi /qn PSQL_PREREQS_INSTALLED=Y TRANSFORMS=1041.mst
```

MSI ファイルで別の設定ファイルの場所を使用する

MSI ファイルを使用できるのは、MSI の前提条件を満たしている場合のみです。Zen MSI をインストールするための前提条件を参照してください。

コマンド ラインで、MSI のパブリック プロパティである PVSW_CFG_FILE を設定し、完全に修飾されたファイル パス名を渡すことによって、別の設定ファイルを使用することができます。

このプロパティを渡すことによって、その値にスペースが含まれているかどうかを判定します。スペースが含まれていない場合、コマンドは以下のようになります。

```
msiexec.exe /i "{パス}¥MSIPackage.msi" PSQL_PREREQS_INSTALLED=Y PVSW_CFG_FILE=c:¥temp¥ConfigFile.ini
TRANSFORMS=1041.mst
```

プロパティの値にスペースが含まれている場合は、次の例に示すように、その値を二重引用符で囲む必要があります。

```
msiexec.exe /i "{ パス }¥MSIPackage.msi" PSQL_PREREQS_INSTALLED=Y PVSW_CFG_FILE="c:¥My Folder¥
ConfigFile.ini" TRANSFORMS=1041.mst
```

サイレント インストールの使用

サイレント インストールは、インストーラー実行中に、ユーザーによる操作を必要とすることなく、また、ユーザー インターフェイスもほとんど表示しないで実行します。これは、次の 2 つの方法で行うことができます。

- msiexec を使用し、**/quiet** オプションまたは **/q** オプションと、そのパラメーターを指定する。最も一般的なオプション パラメーターは **/qn** で、UI を表示しないとするものです。
- InstallShield ベースのインストーラー実行可能ファイルを使用し、**/s** オプションと、MSI インストール中に実行可能ファイルを実行し続けるための **/w** オプション、およびコマンド ライン オプションを msiexec に渡すための **/v** オプションを追加する。**/w** オプションを使用できるのは、Zen setup<product><bitness>.exe インストーラー実行可能ファイルのみです。Install_Zen_<product>.exe 起動アプリケーションで **/w** オプションは**使用できません**。たとえば、次の 1 番目のコマンドでは **/w** を使用できますが、2 番目のコマンドでは使用できません。

```
SetupEnterpriseServer64_x64.exe /w
Install_Zen_Client.exe
```

サイレント インストールは、ほかのインストールを自動的に起動することはできないので、製品が完全な最新版になるよう別のインストールを実行する必要がある場合もあります。サイレント インストール（または基本ユーザー インターフェイス インストール）で特に考慮すべき点を参照してください。

インストーラー実行可能ファイルを使用したサイレント インストールの例

次に、Zen Client のサイレント インストールを指定し、ユーザーの一時フォルダーにログ ファイルを作成する例を示します。

```
Install_Zen_Client.exe /s /v"/qn /l*v ¥"%temp%¥Zen_<version>_SilentInstall.log¥""
```

メモ： /v の後に指定されるプロパティはすべて msiexec へ渡されます。**/v** オプションの後にスペースはありません。渡されるプロパティの文字列は二重引用符で囲む必要があります。プロパティの 1 つがスペースを含むパス名である場合は、パスも二重引用符で囲む必要があります。また、上記の例で示したように、引用符は両方ともバックスラッシュでエスケープ（¥"）する必要があります。

次の例は、ユーザーの一時フォルダーにログ ファイルを作成し、製品キーとサービス インストール オプションを設定します。

```
Install_Zen_Workgroup_Engine.exe /s /v"/qn REBOOT=ReallySuppress PVSW_PSQL_LICENSE_KEY={key_value}
PVSW_RUN_WGE_AS_SVC=Y TRANSFORMS=MyTransform.mst /l*v ¥"%temp%¥Zen_<version>_Install.log¥""
```

メモ： ログ ファイル名とファイルの保存先を設定しない場合は、デフォルトのファイルが %temp% フォルダーに書き込まれます。このファイル名は「Zen_＜バージョン＞_＜エディション＞_＜ビット アーキテクチャ＞_Install.log」という書式で表します。たとえば、「Zen_v16_Server_x64_Install.log」となります。

メモ： **/v** の後に指定されるプロパティはすべて msiexec へ渡されます。**/v** オプションの後にスペースはありません。渡されるプロパティの文字列は二重引用符で囲む必要があります。プロパティの 1 つがスペースを含むパス名である場合は、パスも二重引用符で囲む必要があります。また、上記の例で示したように、引用符は両方ともバックスラッシュでエスケープ（¥"）する必要があります。

MSI を使用したサイレント インストールの例

ある製品の .msi ファイルを別の製品のインストールに埋め込むには、ほとんどの場合、サイレント インストールで行います。次の例は、ユーザーの一時フォルダーにログ ファイルを作成し、製品キーとサービス インストール オプションを指定します。Zen MSI をインストールするための前提条件も参照してください。

```
msiexec.exe /i {path}¥ActianZenv16WGE32_x86.msi /qn REBOOT=ReallySuppress
PVSW_PSQL_LICENSE_KEY={key_value} PSQL_PREREQS_INSTALLED=Y PVSW_RUN_WGE_AS_SVC=N
TRANSFORMS=MyTransform.mst /l*v "%temp%¥Zen_v16_Install.log"
```

この例は、Microsoft 変換ファイル（.mst）を使用してインストールをカスタマイズします。詳細については、Microsoft 変換ファイル（MST）を使った Zen の組み込み を参照してください。

メモ： ログ ファイル名とファイルの保存先を指定しない限り、ログ ファイルは作成されません。

スクリプトを使用したサイレント インストールの例

Zen のサイレント インストールのスクリプトを作成する場合、カスタム インストールの概要に挙げられているインストーラー実行可能ファイルは使用できません。インストール中、これらのファイルの実行は継続されないからです。代わりに、次の 2 つのオプションのいずれかを使用します。どちらの例も Zen Enterprise Server 用で、Microsoft PowerShell を使用しています。スクリプト記述の複雑さを処理しやすくするために、Windows バッチ スクリプトを使用することをお勧めします。Zen インストールのスクリプトを作成する場合は、セットアップ起動プログラムまたは MSI で Zen インストール

自体を起動する**前に**、Zen に必須のコンポーネント（Microsoft Visual C++ ランタイムなど）すべてが先にインストールされるように記述してください。

以下の 2 つの PowerShell スクリプト例は、コピーと貼り付けがしやすいようにそれぞれページごとに記載しています。

セットアップ起動プログラムを使って Zen Enterprise Server をサイレント インストールするための PowerShell スクリプト

```
# Sample script for silently installing Zen Enterprise Server Engine that automatically ensures that
# the required Microsoft Visual C++ 2019 runtimes are installed before invoking the Zen MSI
# installation launcher.
# 1) The PowerShell execution policy allows scripts to be executed.
# 2) The PowerShell session is running with Administrator privileges if UAC is enabled.
# 3) Zen Server Engine installation files are located in a dir named "server" along with this script.
# 4) The bundled Microsoft Visual C++ Runtime installers are invoked BEFORE Zen installation is
#    invoked.
#
$currentFolder= & {Split-Path $myInvocation.ScriptName}
$installVCRT32="$currentFolder¥Data¥ISSetupPrerequisites¥{7BB553E0-BAA5-4184-965C-AEEB89B82D46}¥
VC_redist.x86.exe"
$vc_redist32Log=join-path $env:TEMP "¥vcredist2015_x86.log"
$vc_redist32Args="/quiet","/norestart","/log $vc_redist32Log"
$is64Bit=$False
if ([System.IntPtr]::Size -eq 4) {
 $installFile="$currentFolder¥Data¥SetupEnterpriseServer32_x86.exe"
} else {
 $is64Bit=$True
 $installFile="$currentFolder¥Data¥SetupEnterpriseServer64_x64.exe"
 $installVCRT64="$currentFolder¥Data¥ISSetupPrerequisites¥{2F044C80-608C-4274-AB1D-96D67E047307}¥
VC_redist.x64.exe"
 $vc_redist64Log=join-path $env:TEMP "¥vcredist2015_x64.log"
 $vc_redist64Args="/quiet","/norestart","/log $vc_redist64Log"
}
$setupArgs='/s','/w','/v/qn'
#
# Always install 32-bit MS VCRT.
if (Test-Path "$installVCRT32") {
 Try {
  Write-Host "Starting Microsoft Visual CRT (x86) installation at"(Get-Date).ToString()"..."
  $proc=(Start-Process -FilePath $installVCRT32 -ArgumentList $vc_redist32Args -Wait -PassThru)
  $handle=$proc.Handle # cache proc.Handle http://stackoverflow.com/a/23797762/1479211
  $proc.WaitForExit();
  If ($proc.ExitCode -ne 0) {
   Write-Error "The MS install returned error code: $($proc.ExitCode)"
  } else {Write-Host "The MS install completed successfully at"(Get-Date).ToString()"."}
 }
 Catch {Write-Error "Unknown error: " + [string]$error[0]; exit -1}
}

if ($is64Bit) {
 # Install 64-bit MS VCRT
 if (Test-Path "$installVCRT64") {
  Try {
   Write-Host "Starting Microsoft Visual CRT (x64) installation at"(Get-Date).ToString()"..."
   $proc=(Start-Process -FilePath $installVCRT64 -ArgumentList $vc_redist64Args -Wait -PassThru)
   $handle=$proc.Handle # cache proc.Handle http://stackoverflow.com/a/23797762/1479211
   $proc.WaitForExit();
   If ($proc.ExitCode -ne 0) {
    Write-Error "The MS install returned error code: $($proc.ExitCode)"
```

```
   } else {
    Write-Host "The MS install completed successfully at"(Get-Date).ToString()"."
   }
  }
  Catch {Write-Error "Unknown error: " + [string]$error[0]; exit -1}
 }
} else {Write-Host "Skipping installation of Microsoft Visual C++ (x64) on 32-bit system..."}

# Zen install using setup launcher application.
If (Test-Path "$installFile") {
 Try {
  Write-Host "Starting Zen installation at"(Get-Date).ToString()"..."
  $proc=(Start-Process -FilePath $installFile -ArgumentList $setupArgs -NoNewWindow -Wait -PassThru)
  $handle=$process.Handle # cache proc.Handle http://stackoverflow.com/a/23797762/1479211
  $proc.WaitForExit();
  If ($proc.ExitCode -ne 0) {
   Write-Warning "$_ exited with status code $($proc.ExitCode)"
  } else {
   Write-Host "The setup completed successfully at"(Get-Date).ToString()"."
  }
 }
 Catch {Write-Error "Installing Zen: " + [string]$error[0]; exit -1}
} else {Write-Warning "Cannot install Zen, unable to locate '$installFile'."; exit -1}
```

Microsoft インストーラー（MSI）を使って Zen Enterprise Server をサイレント インストールするための PowerShell スクリプト

```
# Sample script for silently installing Zen Enterprise Server Engine that
# automatically ensures that the required Microsoft Visual C++ 2019 runtimes
# are installed before invoking the Zen MSI installation.
#
# Requirements:
# 1) The PowerShell execution policy allows scripts to be executed.
# 2) The PowerShell session is running with Administrator privileges
# if UAC is enabled.
# 3) The Zen Server Engine installation files are located in a folder named
# "server" in the same location as this script.
# 4) The bundled Microsoft Visual C++ Runtime installers are invoked BEFORE the Zen
# installation is invoked.
$currentFolder= & {Split-Path $myInvocation.ScriptName}
$installVCRT32="$currentFolder¥Data¥ISSetupPrerequisites¥{7BB553E0-BAA5-4184-965C-AEEB89B82D46}¥
VC_redist.x86.exe"
$vc_redist32Log=join-path $env:TEMP "¥vcredist2015_x86.log"
$vc_redist32Args="/quiet","/norestart","/log $vc_redist32Log"
$is64Bit=$False
if ([System.IntPtr]::Size -eq 4) {
 $installFile="$currentFolder¥Data¥Data¥ActianZenv16EnterpriseServer32_x86.msi"
} else {
 $is64Bit=$True
 $installFile="$currentFolder¥Data¥Data¥ActianZenv16EnterpriseServer64_x64.msi"
 $installVCRT64="$currentFolder¥Data¥ISSetupPrerequisites¥{2F044C80-608C-4274-AB1D-96D67E047307}¥
VC_redist.x64.exe"
 $vc_redist64Log=join-path $env:TEMP "¥vcredist2015_x64.log"
 $vc_redist64Args="/quiet","/norestart","/log $vc_redist64Log"
}
$msiexecArgs="/i '"$installFile'"","/qn","REBOOT=ReallySuppress","PSQL_PREREQS_INSTALLED=Y"

# Always install 32-bit MS VCRT.
if (Test-Path "$installVCRT32") {
 Try {
  Write-Host "Starting Microsoft Visual CRT (x86) installation at"(Get-Date).ToString()"..."
  $proc=(Start-Process -FilePath $installVCRT32 -ArgumentList $vc_redist32Args -Wait -PassThru)
```

```
  $handle=$proc.Handle # cache proc.Handle http://stackoverflow.com/a/23797762/1479211
  $proc.WaitForExit();
  If ($proc.ExitCode -ne 0) {
   Write-Error "The MS install returned error code: $($proc.ExitCode)"
  } else {Write-Host "The MS install completed successfully at"(Get-Date).ToString()"."}
 }
 Catch {Write-Error "Unknown error: " + [string]$error[0]; exit -1}
}

if ($is64Bit) {
 # Install 64-bit MS VCRT
 if (Test-Path "$installVCRT64") {
  Try {
   Write-Host "Starting Microsoft Visual CRT (x64) installation at"(Get-Date).ToString()"..."
   $proc=(Start-Process -FilePath $installVCRT64 -ArgumentList $vc_redist64Args -Wait -PassThru)
   $handle=$proc.Handle # cache proc.Handle http://stackoverflow.com/a/23797762/1479211
   $proc.WaitForExit();
   If ($proc.ExitCode -ne 0) {
    Write-Error "The MS install returned error code: $($proc.ExitCode)"
   } else {
    Write-Host "The MS install completed successfully at"(Get-Date).ToString()"."
   }
  }
  Catch {Write-Error "Unknown error: " + [string]$error[0]; exit -1}
 }
} else {Write-Host "Skipping installation of Microsoft Visual C++ (x64) on 32-bit system..."}

# Zen install using MSI file.
if (Test-Path "$installFile") {
 Try {
   Write-Host "Starting Zen installation at"(Get-Date).ToString()"..."
   $proc=(Start-Process -FilePath "msiexec.exe" -ArgumentList $msiexecArgs -Wait -PassThru)
   $handle=$proc.Handle # cache proc.Handle http://stackoverflow.com/a/23797762/1479211
   $proc.WaitForExit();
   If ($proc.ExitCode -ne 0) {
     Write-Error "The install returned error code: $($proc.ExitCode)"
   } else {Write-Host "The install completed successfully at"(Get-Date).ToString()"."}
 }
 Catch {Write-Error "Unknown error: " + [string]$error[0]; exit -1}
} else {Write-Warning "Cannot install Zen, unable to locate '$installFile'."; exit -1}
```

製品アップデートの処理

Zen の製品アップデートは、圧縮された自己解凍形式の .zip アーカイブです。それらのアーカイブに含まれるファイルは、既存の Zen に対するアップデートまたは修正プログラムとして使用されます。

ここでは、以下の項目について説明します。

- 製品アップデートのインストール
- 製品アップデートの削除
- サイレント インストール（または基本ユーザー インターフェイス インストール）で特に考慮すべき点

製品アップデートのインストール

製品アップデートの .exe ファイルを実行した場合、アップデート ファイルの展開（解凍）先の選択（デフォルトは %temp% フォルダー）と、アップデート実行可能ファイルを実行するかどうかの選択が求められます（デフォルトは「実行する」オプションが選択済み）。

アップデート実行可能ファイルの 1 つが実行されると、インストール済みの Zen のビット数を確認し、それに応じた適切な Microsoft 修正プログラム パッケージ（.msp）が実行されます。

- Update_Zen_Client.exe
- Update_Zen_EnterpriseServer.exe
- Update_Zen_Workgroup_Engine.exe

例：製品アップデートのサイレント インストール

```
Update_Zen_Client.exe /s /v"/qn /l*v ¥"%temp%¥Zen_Client_silent_update.log¥""
```

メモ： /v の後に指定されるプロパティはすべて msiexec へ渡されます。**/v** オプションの後にスペースはありません。渡されるプロパティの文字列は二重引用符で囲む必要があります。プロパティの 1 つがスペースを含むパス名である場合は、パスも二重引用符で囲む必要があります。また、上記の例で示したように、引用符は両方ともバックスラッシュでエスケープ（¥"）する必要があります。

製品アップデートの削除

Windows システムでは、ほとんどの場合、Zen 製品アップデートをアンインストールすれば " ロール バック " することができます。この削除操作によって、Zen バイナリを製品アップデート前のバージョンの状態に戻します。削除できない修正プログラムについては、このセクションの「メモ」を参照してください。

コマンド ラインから Zen の製品アップデートの削除を行うには、以下のものが必要です。

- インストールに使用した元のインストール パッケージ（.msi）または製品コード GUID。Zen の各インストール タイプにはそれぞれ個別の製品コード GUID があります。
- 元の Microsoft 修正プログラム（.msp）ファイル。

以下のコマンドのどちらかで製品アップデートを削除できます。

- `msiexec /package <{_MSI のパス } または製品コード GUID> /uninstall <MSP のパス > /l*v "%temp%¥uninstall_patch.log"`
- `msiexec /i <{MSI のパス } または製品コード GUID> MSIPATCHREMOVE=<MSP のパス > /l*v "%temp%¥uninstall_patch.log"`

メモ： 場合によっては、Zen の修正プログラムを削除できないこともあります。" 修正プログラム パッケージのアンインストールは、サポートされません " という警告メッセージが提示され、インストール ログ ファイルにもこの警告メッセージが記録されます。修正プログラムを削除するには、Zen をアンインストールして再インストールした後、必要な修正プログラムのみを適用します。

Zen をアンインストールまたは変更するときは管理者特権が要求されるようにするために、Windows の［プログラムと機能］に Zen アップデートは表示されません。

サイレント インストール（または基本ユーザー インターフェイス インストール）で特に考慮すべき点

これら 2 つのインストール方法では、別のインストールを自動的に起動することはできません。

- サイレント インストール

- MSI 基本ユーザー インターフェイス インストール（**/qb** や **/passive** などの MSI コマンド ライン オプションを使用して、インストール中に進行状況バーのみを表示して UI なしで実行するインストール）

どちらの場合でも、インストーラー実行可能ファイルのユーザー インターフェイスから実行されるインストールとは異なり、必要なすべての修正プログラムがインストールで適用されるようにすることはできません。サイレント インストールまたは基本ユーザー インターフェイス インストールを行う場合、製品を完全な状態にするには、それらの修正プログラムを個別に適用する必要があるかもしれません。

次の 2 つのトピックでは、個別のアップデート方法について説明しています。

- アップグレードする前の製品修正プログラム
- 現在のリリース用の製品修正プログラム

アップグレードする前の製品修正プログラム

アップグレードは、11.31 から 12.00 など、製品のバージョンを以前のリリースから最新のリリースへ移行します。最新のリリースにアップグレードする**前に**、修正プログラムを適用しておく必要がある場合があります。修正プログラムは .msp ファイルとして提供されます。最新リリースのリリース ノートを参照してください。これには必要な修正プログラムがすべて記載されています。これらの修正プログラムは、アップグレード前に必ずインストールしておいてください。

たとえば、メジャー リリースからサービス パックへのアップグレードに対しオプションの修正プログラムが使用できます。元のリリースで適用されている個別の設定を保持したい場合は、この修正プログラムをインストールします。この修正プログラムは Actian Web サイトから提供されるため、.msp ファイルをダウンロードして適用する必要があります。

任意の修正プログラムをサイレント インストールするには、以下のコマンドを使用します。

```
Product_type_platform.msp /qn /l*v "%temp%¥product_beforeupdate.log"
```

この場合、

Product は製品を表し、*type* および *platform* は製品の種類と製品を実行するプラットフォームを表します。たとえば、ActianZenv16Patch_EnterpriseServer32_x86.msp は 32 ビット プラットフォームにおける Enterprise Server を表します。

現在のリリース用の製品修正プログラム

サイレント インストール**後**、そのリリースで修正プログラムが必要となる場合があります。修正プログラムが必要な場合、その .msp ファイルはインストール メディアの ~¥Redist¥Data フォルダーにあります。

そのフォルダーにある .msp ファイルをインストールすると、現在のリリースのインストールが完全な状態になります。たとえば、以下のコマンドは 64 ビット プラットフォーム上に Enterprise Server 用の修正プログラムをサイレント インストールします。

```
インストール メディアのパス¥Redist¥Data¥ActianZenv16Patch_EnterpriseServer64_x64.msp /qn /l*v "%temp%¥Zenv16_InstalledPatch.log"
```