

# CA Configuration Automation®

## ブループリント開発者ガイド

r12.8 SP02



このドキュメント（組み込みヘルプシステムおよび電子的に配布される資料を含む、以下「本ドキュメント」）は、お客様への情報提供のみを目的としたもので、日本 CA 株式会社（以下「CA」）により随時、変更または撤回されることがあります。

CA の事前の書面による承諾を受けずに本ドキュメントの全部または一部を複写、譲渡、開示、変更、複本することはできません。本ドキュメントは、CA が知的財産権を有する機密情報です。ユーザは本ドキュメントを開示したり、

(i) 本ドキュメントが関係する CA ソフトウェアの使用について CA とユーザとの間で別途締結される契約または (ii) CA とユーザとの間で別途締結される機密保持契約により許可された目的以外に、本ドキュメントを使用することはできません。

上記にかかわらず、本ドキュメントで言及されている CA ソフトウェア製品のライセンスを受けたユーザは、社内でユーザおよび従業員が使用する場合に限り、当該ソフトウェアに関連する本ドキュメントのコピーを妥当な部数だけ作成できます。ただし CA のすべての著作権表示およびその説明を当該複製に添付することを条件とします。

本ドキュメントを印刷するまたはコピーを作成する上記の権利は、当該ソフトウェアのライセンスが完全に有効となっている期間内に限定されます。いかなる理由であれ、上記のライセンスが終了した場合には、お客様は本ドキュメントの全部または一部と、それらを複製したコピーのすべてを破棄したことを、CA に文書で証明する責任を負います。

準拠法により認められる限り、CA は本ドキュメントを現状有姿のまま提供し、商品性、特定の使用目的に対する適合性、他者の権利に対して侵害のないことについて、黙示の保証も含めいかなる保証もしません。また、本ドキュメントの使用に起因して、逸失利益、投資損失、業務の中断、営業権の喪失、情報の喪失等、いかなる損害（直接損害か間接損害かを問いません）が発生しても、CA はお客様または第三者に対し責任を負いません。CA がかかる損害の発生の可能性について事前に明示に通告されていた場合も同様とします。

本ドキュメントで参照されているすべてのソフトウェア製品の使用には、該当するライセンス契約が適用され、当該ライセンス契約はこの通知の条件によっていかなる変更も行われません。

本ドキュメントの制作者は CA です。

「制限された権利」のもとでの提供: アメリカ合衆国政府が使用、複製、開示する場合は、FAR Sections 12.212、52.227-14 及び 52.227-19(c)(1)及び(2)、ならびに DFARS Section 252.227-7014(b)(3) または、これらの後継の条項に規定される該当する制限に従うものとします。

Copyright © 2015 CA. All rights reserved. 本書に記載された全ての製品名、サービス名、商号およびロゴは各社のそれぞれの商標またはサービスマークです。

## CA への連絡先

テクニカル サポートの詳細については、弊社テクニカル サポートの Web サイト (<http://www.ca.com/jp/support/>) をご覧ください。



# 目次

---

第 1 章: ドキュメントの概要	7
第 2 章: ブループリントの概要	9
コンポーネント ブループリントの構造および内容の理解	10
ネスト化	10
インジケータ	11
管理対象	13
パラメータ	14
構成	15
診断	17
第 3 章: ブループリントの作成	19
ブループリントの作成	19
ブループリント ファイル フィルタおよび属性の定義	25
ブループリント レジストリ フィルタおよび属性の定義	27
第 4 章: ブループリント エlement リファレンス	29
カテゴリーの説明	29
フィルタの説明	31
POSIX 1003.2-1992 パターン マッチング	32
POSIX パターン マッチング表現の構文	32
変数の代入	33
表現タイプ	34
解釈方法の説明	40
正規表現	48
CA Configuration Automation で提供される Java プラグイン	52
表形式データ パーサの理解と使用	54
表形式データ パーサへのアクセスと使用	55
付録 A: ブループリント ウィザード UI リファレンス	65
[ブループリント] ページ: コンポーネント ブループリント フィールド	66
[ディスカバリ方式] ページ: 検索オプション フィールド	68

---

[ディスカバリ方式] ページ: ファイルインジケータ フィールド.....	69
[ディスカバリ方式] ページ: レジストリ インジケータの検索オプション フィールド.....	70
[ディスカバリ方式] ページ: レジストリ インジケータ フィールド.....	71
[ディスカバリ方式] ページ: ネットワーク プロブ フィールド.....	71
[ディスカバリ検証ルール] ページ: ディスカバリ検証ルール フィールド.....	72
[管理] ページ: ファイル管理オプション フィールド.....	81
[管理] ページ: ディレクトリ フィールド.....	81
[管理] ページ: ディレクトリ フィールド.....	82
[ファイルフィルタおよび属性] ページの [ルール] タブのフィールド.....	82
レジストリ管理フィールド.....	85
[レジストリ フィルタおよび属性] ページのキーの追加フィールド.....	86
[レジストリ フィルタおよび属性] ページの値詳細フィールド.....	88
[データベース] ページのフィールド.....	90
[コンポーネント パラメータおよび変数] ページのフィールド.....	93
[構成] - [ファイル解析] ページ.....	106
構成実行ファイル ページ.....	106
クエリの追加ペイン.....	113
ファイル構造クラス タブ.....	115
[File Structure Class Group] フィールドと [パラメータ] フィールド.....	116
マクロ ページ.....	120
完了ページ.....	121

## 第 5 章: [コンポーネント グループ化オプション] タブでのネスト修飾子によるブループリントのカスタマイズ 127

インスタンス情報をサポートする Oracle および MSSQL のブループリント.....	127
インスタンス情報をサポートする Oracle および MSSQL のブループリントのカスタマイズ: [コンポーネント グループ化オプション] タブ.....	128
インスタンス情報をサポートする Oracle および MSSQL のブループリントのカスタマイズ: 入力修飾子としてのコンポーネント パラメータ.....	129

# 第 1 章: ドキュメントの概要

---

このドキュメントは、ブループリントの作成または変更を行う **CA Configuration Automation** ユーザを対象としています。このドキュメントでは、ブループリントのタイプと使用方法の概要について説明します。このドキュメントの大部分は、ブループリントの作成に関連する手順を説明しています。





## 第 2 章: ブループリントの概要

---

ブループリントはソフトウェア コンポーネントの概念的な定義またはメタデータです。このメタデータは、以下の処理を行うためのディレクティブおよびメカニズムを定義します。

- 指定されたコンピュータ上のソフトウェア コンポーネントを検出します
- コンポーネントのファイル システムおよびデータベース エlement をキャプチャします
- コンポーネント内およびコンポーネント間の関係および依存関係を表現および表示します
- 構成情報を検索、分析、および管理します
- 診断マクロを定義、実行、および解析します
- これらのすべてのエレメントに、推奨されるベスト プラクティス値を定義します

**CA Configuration Automation** は、構成および管理タスクを簡略化する標準化された形式でブループリントを表示します。**CA** は、共通で使用されているソフトウェア コンポーネントの、事前定義済みブループリントのライブラリを提供します。既存のブループリントを編集したり、カスタムブループリントを作成することもできます。

## コンポーネントブループリントの構造および内容の理解

コンポーネントブループリントまたはコンポーネントブループリントのエレメントを表示、追加、または変更する機能は、ユーザのユーザアカウントまたはグループに割り当てられた **CA Configuration Automation** ロールに依存します。**CA Configuration Automation** ロールおよび権限の詳細については、「**CA Configuration Automation 実装ガイド**」を参照してください。

コンポーネントブループリントはすべて、以下の構造的なフォルダエレメントで構成される、標準化されたツリービューで示されます。

- ネスト化
- インジケータ
- 管理対象
- パラメータ
- 環境設定
- 診断
- マニュアル
- ランタイム
- ユーティリティ

各エレメントについては、次のセクションで説明します。

### ネスト化

ネスト化エレメントは、コンポーネント間の関係を強調することができます。たとえば、ソフトウェアコンポーネントが使用するいくつかの下位コンポーネントに依存しており、かつそれらのコンポーネントがプライマリコンポーネントのファイルシステムルートにインストールされている場合、ネスト化はそれらの親子関係の実施に使用できます。

たとえば、**Oracle** データベースは、通常はそのインストールディレクトリ内に **Java Runtime Engine** および **Apache Web** サーバをインストールします。ユーティリティコンポーネントと **Oracle** データベースの関係は、**Oracle** コンポーネント内に **JRE** および **Apache** コンポーネントをネストすることによって表わされます。

## インジケータ

インジケータ プライマリ フォルダは、サーバ上の検索対象の場所、およびコンポーネントを識別する方法を定義します。

コンポーネントのディスカバリが **CA Configuration Automation** エージェントがインストールされていないサーバに拡張されており、**CA Configuration Automation** にこの種類のディスカバリに対して利用可能なレジストリおよびファイル システムがないため、ユーザは別のインジケータ セットを定義して、エージェントがインストールされているサーバのコンポーネントおよびエージェントがインストールされていないサーバのコンポーネントを検索する必要があります。

- **CCA** エージェントを持つサーバ上では、コンポーネント ディスカバリは、ファイル インジケータを使用してファイル システムを検索し、コンポーネントの存在を示す特定のファイルとディレクトリのセットを探し、コンポーネントの管理対象ファイルが配置された対応するルート ディレクトリを特定します。

**Windows** サーバでは、この代わりにレジストリ エントリをインジケータとして定義できます。インジケータとしてレジストリ エントリを使用する場合、パラメータのプライマリ フォルダ内のコンポーネント ルート ディレクトリを指定する必要があることに注意してください。

**注:** **Windows** プラットフォーム上ではファイル システム インジケータとレジストリ インジケータの両方を使用できますが、両方ではなく片方だけを定義することを推奨します。ただし、複数のインジケータ タイプのセットを定義して、複数のパターンマッチングにより、複数のプラットフォームにまたがって同じコンポーネントブループリントを拡張することもできます。インジケータ セットを構築するための良い方法は、お互いの既知の位置関係が [ルートからの階層数] または [ルートからのパス] に定義されている、2 つから 3 つのファイル/ディレクトリ、またはレジストリ キー/値を定義することです。定義するインジケータが多すぎると、不適切で誤った結果が生成される場合があります。

- **CCA** エージェントのないサーバでは、コンポーネント ディスカバリは、ネットワーク プロブを使用して **TCP** ポートをスキャンし、それらのポートからの応答を収集し、収集した応答と想定したパターンとの一致を比較して、そのポートでアクティブなサービスのタイプを特定することができます。

インジケータは、インストール済みコンポーネントの存在を判定するには必ずしも十分だとは限らず、また同様のインジケータを備えた2つのコンポーネント間の相違を区別できません。検証ディレクティブフォルダには、インストールが不完全なコンポーネント、間違ったバージョンのコンポーネント、または特定のサービスに関係のないコンポーネントを破棄するために使用されるディレクティブが含まれています。

**注:** 検証ディレクティブがコンポーネント ディスカバリで実行される順序により、データベースおよび構成ファイル パラメータが利用できない場合があります。

### インジケータ エLEMENTの追加

追加	アクセスと手順
ファイル検索オプション	ファイル フォルダをクリックし、次に [検索オプションの追加] ボタンをクリックします。
ディレクトリ インジケータ	ファイル、/ (ファイルルート) フォルダ、[ディレクトリの追加] ボタンの順にクリックします。
ファイル インジケータ	ファイル、/ (ファイルルート) フォルダ、[ファイルの追加] ボタンの順にクリックします。
レジストリ 検索オプション	レジストリ フォルダをクリックし、次に [検索オプションの追加] ボタンをクリックします。
レジストリ キー インジケータ	ファイル、¥ (レジストリ ルート) フォルダ、[キーの追加] ボタンの順にクリックします。
レジストリ 値インジケータ	ファイル、¥ (レジストリ ルート) フォルダ、[値の追加] ボタンの順にクリックします。
ネットワーク プローブ	サービス フォルダをクリックし、次に [ネットワーク プローブの追加] ボタンをクリックします。
検証ディレクティブ	検証ディレクティブ フォルダをクリックし、次に [ディレクティブの追加] ボタンをクリックします。

## 管理対象

〔管理対象〕プライマリ フォルダは、管理対象コンポーネントと関連付けられたデータベース エlement、重要なファイル、レジストリ エントリを定義します。

管理対象フォルダにファイルやレジストリ エントリが含まれていない場合、コンポーネントルート ディレクトリの下のすべてのファイル、およびレジストリ ルートの下のすべてのレジストリ エントリが管理されます。コンポーネントが保持するファイルまたはレジストリ エントリの数に限られている場合は、ルート以下の該当Elementをすべて管理することをお勧めします。ただし、多くのファイルを持つ複雑なコンポーネントについては、注意すべきファイルおよびレジストリ エントリのみを特定した方が効果的です。具体的なファイルおよびレジストリ エントリを特定すると、管理対象コンポーネントのビューを絞り込むことができます。

ファイル システム オーバレイおよびレジストリ オーバレイのフォルダでは、必要に応じて特定のファイルおよびレジストリ Elementをカスタマイズできます。たとえば、カテゴリ、フィルタ、またはウェイトを割り当てたり、Elementにノートおよびルールを添付したりすることができます。

データ フォルダでは、データベース接続情報が提供され、ユーザは以下を実行できます。

- コンポーネントブループリントの任意の場所を参照するデータベースの定義
- テーブル定義およびインデックスを含むデータベース メタデータの管理

**重要:** 構成、ドキュメント、またはランタイムのプライマリ フォルダ内で参照されているファイルは、管理対象ファイルとして含めてください。

## 管理対象Elementの追加

追加	アクセスと手順
ディレクトリ	ファイル、\$(Root) フォルダ、〔ディレクトリの追加〕 ボタンの順にクリックします。
ファイル	ファイル、\$(Root) フォルダ、〔ファイルの追加〕 ボタンの順にクリックします。

ファイルシステム オーバレイ ディレクトリ	ファイルシステム オーバレイ、\$(Root) フォルダ、[ディレクトリの追加] ボタンの順にクリックします。
ファイルシステム オーバレイ ファイル	ファイルシステム オーバレイ、\$(Root) フォルダ、[ファイルの追加] ボタンの順にクリックします。
レジストリ キー	レジストリ、\$(RegistryRoot) フォルダ、[キーの追加] ボタンの順にクリックします。
レジストリ 値	レジストリ、\$(RegistryRoot) フォルダ、[値の追加] ボタンの順にクリックします。
レジストリ オーバレイ キー	レジストリ オーバレイ、\$(RegistryRoot) フォルダ、[キーの追加] ボタンの順にクリックします。
レジストリ オーバレイ 値	レジストリ オーバレイ、\$(RegistryRoot) フォルダ、[値の追加] ボタンの順にクリックします。
データベース	データ フォルダをクリックし、次に[データベースの追加] ボタンをクリックします。
データベース テーブル	データ フォルダ内のデータベースをクリックし、次に[テーブルの追加] ボタンをクリックします。

## パラメータ

プライマリ フォルダにはディレクティブ サブフォルダが含まれ、コンポーネントを特定および識別する際に重要になる、ファイルシステムまたはレジストリ ルート、コンポーネントバージョン、ベンダ、およびデータベース接続情報のような詳細情報を定義および表示します。

ディスカバリは、コンポーネントブループリントのファイルシステム ルートおよびレジストリ ルートを決定し、検出されたサービスのツリービューにこれらのパラメータを表示します。コンポーネントブループリントに、特殊な **Version** および **NameQualifier** パラメータが定義されている場合は、これらが検出されたサービスのツリービュー内で名前の後に表示されます。コンポーネントブループリントのパラメータ **NameQualifier** は \$(Product Name) SP\$(Service Pack) として定義され、パラメータ **Version** は \$(RegistryRoot)¥Windows NT¥CurrentVersion¥CurrentVersion として定義されます。

パラメータ ディレクティブは、診断、ユーティリティ、およびルールとその他のパラメータ定義の変数代入に使用できます。 文字列 `$(ParameterName)` が値として入力されると、そのパラメータの実際の値が代入されます。

## パラメータ エLEMENTの追加

追加	アクセスと手順
パラメータ ディレクティブ	ディレクティブ フォルダをクリックし、次に [ディレクティブの追加] ボタンをクリックします。

## 構成

[構成] プライマリ フォルダでは、コンポーネント構成情報を検索および解釈する方法を定義します。 管理対象のファイルまたはデータベース内の構成情報を検索したり、実行可能ファイルの出力から抽出したりすることができます。

[構造クラス] フォルダでは、構成ファイル、データベース クエリ、実行可能ファイルを解釈する方法を定義します。 この情報には、構成データ セット内で有効な値の意味が含まれます。

- データ入力
- デフォルト値
- 列挙値
- 修飾子
- カテゴリ
- フィルタ
- ウェイト
- ルール

構造クラスはメタデータと見なします。

[構成ファイル] フォルダが特定するファイルは、表示および比較分析のために検索、解析、解釈されます。

データ フォルダは、構成データの抽出を実行するクエリまたはストアド プロシージャおよび比較分析結果の解釈方法を識別します。

実行可能ファイル フォルダは、サーバから構成情報を抽出するスクリプトおよびディレクティブと、比較分析の結果を解釈する方法を定義します。

### 構成エレメントの追加

追加	アクセスと手順
構造クラスのクラス	構造クラス フォルダをクリックし、次に [クラスの追加] ボタンをクリックします。
構造クラスのパラメータ	構造クラス フォルダのクラスをクリックし、次に [パラメータの追加] ボタンをクリックします。
ファイル	ファイル、\$(Root) フォルダ、 [ファイルの追加] ボタンの順にクリックします。
構造クラス グループ	構造クラス フォルダのクラスをクリックし、次に [グループの追加] ボタンをクリックします。  ユーザがグループを選択するときに表示される [グループの追加] ボタンおよび [パラメータの追加] ボタンを使用して、グループに追加グループおよびパラメータを追加することができます。  また、 [グループのコピー] ボタンを使用して、グループおよびその関連するパラメータをすべてコピーできます。
データベース	データ フォルダをクリックし、次に [データベースの追加] ボタンをクリックします。
データベース クエリ	データ フォルダのデータベースをクリックし、次に [クエリの追加] ボタンをクリックします。
実行可能ファイル ディレクティブ	実行可能ファイル フォルダをクリックし、次に [ディレクティブの追加] ボタンをクリックします。



## 診断

〔診断〕 プライマリ フォルダは、診断、トラブルシューティング、およびコンポーネントの問題の修正に使用する実行可能ファイルおよびマクロを定義します。

サーバ上で実行することができるすべての実行可能ファイル、スクリプト、またはバッチ ファイルをここで定義でき、適切なアクセス制御ロールの **CA Configuration Automation** ユーザが利用できるようにすることができます。 マクロは、コンポーネントブループリントによって管理されるデータを格納しているサーバに固有の問題の診断に役立ち、頻繁に使用されるスクリプトおよびトラブルシューティング ツールを含める方法を提供します。

### 診断エレメントの追加

追加	アクセスと手順
ファイル	ファイル、\$(Root) フォルダ、〔ファイルの追加〕 ボタンの順にクリックします。
ファイル使用状況	ファイル、\$(Root) フォルダ、〔ファイル使用状況の追加〕 ボタンの順にクリックします。
マクロ	マクロ フォルダをクリックし、次に〔マクロの追加〕 ボタンをクリックします。
マクロ ステップ	マクロ フォルダのマクロをクリックし、次に〔ステップの追加〕 ボタンをクリックします。



## 第 3 章: ブループリントの作成

---

この章では、カスタム ブループリントを作成する手順について説明します。この章に含まれているフィールドの説明は、次の章（「ブループリント エlement リファレンス」）と共に使用できます。

### ブループリントの作成

CA Configuration Automation では、カスタム ブループリントを作成および管理できます。単純なブループリントは簡単に作成できますが、詳細で複雑なブループリントは慎重な計画とテストが必要です。

次の手順に従ってください:

1. 製品のメインページの右上の [管理] をクリックし、次に、左上の [ブループリント] タブをクリックします。  
[ブループリント] ペインが開き、既存のすべてのブループリントがリスト表示されます。
2. [ブループリント] ペインで [テーブル アクション] ドロップダウンリストから [ブループリントの作成] を選択します。  
[ブループリントの作成] ウィザードが表示されます。
3. [ブループリント] ページの [コンポーネント ブループリント] ペインで、各 [フィールド](#) (P. 66) を設定して、[次へ] をクリックします。  
[ディスカバリ方式] ページが表示されます。このページには、[ファイル インジケータ] ペインおよび [新規検索オプションの追加] ペインが表示されます。
4. [新規検索オプションの追加] ペインの [検索オプション] [フィールド](#) (P. 68) を設定します。
5. [ディレクトリ/ファイルの追加] をクリックし、次に、[新規ファイルの追加] ペイン内の [ファイル インジケータ] [フィールド](#) (P. 69) を設定します。

6. [レジストリ インジケータ] リンクをクリックします。  
[レジストリ インジケータ] ペインが、[レジストリ インジケータ] が展開され、¥HKEY\_LOCAL\_MACHINE エlementが選択された状態で表示されます。 [¥HKEY\_LOCAL\_MACHINE] ペインには [検索オプション] フィールドが表示されます。
7. [¥HKEY\_LOCAL\_MACHINE] ペインの [検索オプション] [フィールド](#) (P. 70)を設定して、[保存] をクリックします。
8. [レジストリ 値/キーの追加] をクリックします。
9. [新規レジストリ 値/キーの追加] ペインの [レジストリ インジケータ] [フィールド](#) (P. 71)を設定します。
10. [ネットワーク プローブ] リンクをクリックします。

[ネットワーク プローブ] ペインが [ネットワーク プローブ] Elementが選択された状態で表示されます。 [新規ネットワーク プローブの追加] ペインに [ネットワーク プローブ] フィールドが表示されます。

**CA Configuration Automation** エージェントのないサーバでは、ディスカバリ操作はネットワーク プローブを使用して次のことが可能です。

- ポートをスキャンします
- スキャンしたポートから応答を収集します
- 応答と予期される表現を比較することにより、スキャンされたポートでアクティブなサービスのタイプを確定します

11. [ネットワーク プローブ] [フィールド](#) (P. 71)を設定して、[次へ] をクリックします。

[ディスカバリ 検証ルール] ページが、左ペインに [ディスカバリ 検証ルール] Elementが選択された状態で表示されます。 [新規ディスカバリ 検証ルールの追加] ペインに、[ディスカバリ 検証ルール] フィールドが表示されます。

ディスカバリの間に検証ルールが実行されて、検出されたコンポーネントが正しく識別されることを確認します。 場合によっては、ファイルおよびレジストリ インジケータがインストールされているコンポーネントの存在を確認できません。 同様に、ファイルおよびレジストリは、似たインジケータを持つ 2 つのコンポーネントの間で区別できないことがあります。 検証ルールが失敗した場合は、コンポーネントのディスカバリは失敗します。

12. [ディスカバリ検証ルール] [フィールド](#) (P. 72)を設定して、[次へ]をクリックします。

[ディスカバリ検証ルール] の値が保存され、[管理] ページが表示されます。[ファイル管理] ペインで、\$(Root) フォルダが選択され、[\$(Root)] ペインには、[ファイル管理オプション] フィールドが表示されます。[管理] ページは以下のページをリンクします。

- ファイルフィルタおよび属性
- レジストリ管理
- レジストリ フィルタおよび属性
- データ管理

これらのページで、重要なファイル属性、レジストリ エントリ、および管理対象のコンポーネントと関連付けられたデータベース エlementを定義できます。

ファイルおよびレジストリ エントリが定義されない場合、コンポーネント ルート ディレクトリ以下のファイルおよびレジストリ ルート以下のレジストリ エントリがすべて管理されます。コンポーネントが保持するファイルまたはレジストリ エントリの数に限られている場合は、ルート以下のファイルおよびレジストリ エントリをすべて管理することをお勧めします。これに対し、多数のファイルを保持する複雑なコンポーネントの場合は、重要なディレクトリ、ファイル、およびレジストリ エントリのみ焦点を絞って指定します。[管理] ページから特定のファイルおよびレジストリ エントリを識別すると、管理されたコンポーネント ビューを調整できます。

13. [管理] ページの [\$(Root)] ペインの [ファイル管理オプション] [フィールド](#) (P. 81)を設定します。

14. (オプション) [ディレクトリの追加] をクリックし、[新規ディレクトリの追加] ペインの [ディレクトリ] [フィールド](#) (P. 81)を設定して、[保存] をクリックします。

ディレクトリが [ファイル管理] ペイン内の \$(Root) ディレクトリの下に追加されます。

15. (オプション) [ファイルの追加] をクリックし、[新規ファイルの追加] ペインの [ファイル] [フィールド](#) (P. 82)を設定して、[保存] をクリックします。

ファイルが [ファイル管理] ペイン内の \$(Root) ディレクトリの下に追加されます。

16. (オプション) [ファイル管理] ペインでノードを選択し、必要に応じて、手順 14 および 15 を繰り返して、サブディレクトリおよびネストされた他のファイルを作成します。

新しいエレメントが [ファイル管理] ペイン内の選択したノードの下に追加されます。

17. [ファイル フィルタ および 属性] リンクをクリックします。

手順 14 ~ 16 で作成したディレクトリおよびファイル構造が、[ファイル フィルタ および 属性] ペインの \$(Root) フォルダの下に表示されます。

18. ブループリント ファイルのフィルタおよび属性を定義します。

19. [レジストリ管理] リンクをクリックし、[レジストリ管理] [フィールド](#) (P. 85) を設定して、[保存] をクリックします。

20. (オプション) [キーの追加] をクリックし、[新規キーの追加] ペインの [キー] フィールドを設定して、[保存] をクリックします。

キーファイルが [レジストリ管理] ペイン内の \$(RegistryRoot) ディレクトリの下に追加されます。

21. (オプション) [値の追加] をクリックし、[新規値の追加] ペインの [値] フィールドを設定して、[保存] をクリックします。

キーファイルが [レジストリ管理] ペイン内の \$(RegistryRoot) ディレクトリの下に追加されます。

22. (オプション) [レジストリ管理] ペインでノードを選択し、必要に応じて、手順 20 および 21 を繰り返してネストされた他のキーおよび値を作成します。

新しいキーまたは値が [レジストリ管理] ペイン内の選択したノードの下に追加されます。

23. [ブループリントのレジストリ フィルタ および 属性を定義します。](#) (P. 27)

キーファイルが [レジストリ管理] ペイン内の \$(RegistryRoot) ディレクトリの下に追加されます。

24. [データ管理] リンクをクリックします。

左ペインには [データ管理] フォルダが表示され、右ペインには [データベース] ページが表示されます。

25. [データベース] ページで各 [フィールド](#) (P. 90) を設定して、[保存] をクリックします。  
データベースが [データ管理] ペインに表示されます。
26. [次へ] をクリックします。  
[コンポーネント パラメータおよび変数] ページが表示されます。
27. [コンポーネント パラメータおよび変数] ページで各フィールドを設定して、[保存] をクリックします。
28. [次へ] をクリックします。  
[構成] - [ファイル解析] ページが表示されます。
29. [構成]-[ファイル解析] ページの各 [フィールド](#) (P. 106) を設定して、[構成実行ファイル] リンクをクリックします。
30. [構成実行ファイル] ページの各 [フィールド](#) (P. 106) を設定して、[保存] をクリックします。
31. [構成データ] リンクをクリックして、[データベース] フィールドを設定します。このフィールドは、ブループリントが使用するデータベースを定義します。ドロップダウンリストには、手順 23 で作成したデータベースが表示されます。
32. [保存] をクリックします。左ペイン内の [構成データ] ツリーに、データベースが表示されます。
33. [クエリの追加] ペインの各 [フィールド](#) (P. 113) を設定して、[保存] をクリックします。左ペイン内の [構成データ] ツリーに、クエリが表示されます。
34. [ファイル構造データ] リンクをクリックし、[ファイル構造クラス] タブの各 [フィールド](#) (P. 115) を設定して、[保存] をクリックします。左ペイン内の [ファイル構造クラス] ツリーに、構造クラスが表示されます。
35. [優先順位] タブをクリックし、[グループの追加] または [パラメータの追加] をクリックして、表示される各 [フィールド](#) (P. 116) を設定します。

36. [次へ] をクリックし、[マクロ] ページの各 [フィールド](#) (P. 120) を設定して、[次へ] をクリックします。

[コンポーネント グループ化オプション] ページが表示されます。ページに含まれるオプションを使用して、サービス内のコンポーネントをネストできます。これにより、表示でコンポーネント間の関係が強調されます。たとえば、あるコンポーネントがプライマリ コンポーネント ファイル システム ルート内の下位コンポーネントに依存する場合、ネスト化を使用して親子関係を実施できます。

たとえば、Oracle データベースは、通常はインストール ディレクトリ内に Java Runtime Engine および Apache Web サーバをインストールします。これらのユーティリティ コンポーネントと Oracle データベースの関係は、Oracle コンポーネントに JRE および Apache をネストすることにより表現されます。

37. [コンポーネント グループ化オプション] ページの各 [フィールド](#) (P. 121) を設定して、[完了] をクリックします。

ブループリントが作成され、[ブループリント] テーブルに表示されます。



## ブループリント ファイル フィルタおよび属性の定義

次の手順に従ってください:

1. \$(Root) フォルダを選択します。

[優先順位] テーブルに、\$(Root) のディレクトリがリスト表示されます。

2. (オプション) 列を選択し、[アクションの選択] ドロップダウン リストから [上へ移動] または [下へ移動] を選択して、ディレクトリの優先順位を変更します。

解析されたデータ (パラメータとグループ) およびオーバーレイの [ファイル構造クラス] (説明、ルール、フィルタ、およびカテゴリ) にメタリンクを適用する際に、順序を考慮する必要があります。変更検出操作、比較操作、およびルール コンプライアンス操作では、正しい値と照合するためにコンテンツの処理順序が重要になります。

たとえば、ブループリント ファイル構造クラス内の以下のパラメータ定義について考えます。これらは、それぞれ独自のルール、カテゴリフィルタ、およびウェイトを持っています。

ab.\*

a.\*

.\*

- a で始まるすべてのパラメータは、それらの各フィルタを取得します。
- ab で始まるすべてのパラメータは、それらの各フィルタを取得します。この場合、前にある a.\* は上書きされます。
- a または ab で始まらないすべてのパラメータは、.\* で指定された各フィルタを取得します。

したがって、より具体的なパラメータを最初に定義し、その後により一般的なパラメータを定義します。

3. (オプション) [ファイル] テーブルにリスト表示されたファイルに対し、手順 2 を繰り返し実行します。
4. カテゴリを割り当てるか、または選択したディレクトリまたはファイル用のフィルタを作成します。
5. ファイルまたはディレクトリをクリックし、[利用可能なカテゴリ] 列の 1 つ以上のオプションをダブルクリックします。

選択した項目は、[選択されたカテゴリ] 列に追加されます。

6. [利用可能なフィルタ] 列で1つ以上のオプションをダブルクリックします。

選択した項目は、[選択されたフィルタ] 列に追加されます。

7. ファイルまたはディレクトリの [ウェイト] を選択し、次に、[保存] をクリックします。

8. (オプション) [フィルタの管理] タブをクリックします。

[フィルタの管理] タブには、選択したフォルダのサブフォルダとファイルがリスト表示されます。[フィルタの管理] タブでは、特定のフォルダ、サブフォルダ、およびファイルに [変更検出を実行しない] および [時刻変化] のフィルタを適用することも、これらのフィルタを解除することもできます。

9. (オプション) リストからサブフォルダとファイルを選択し、[アクションの選択] ドロップダウンリストから以下のいずれかのアクションを選択します。

- [変更検出を実行しない] を設定

[ファイルフィルタおよび属性] ペインで定義されている選択済みのフォルダ、サブフォルダ、およびファイルに対して [変更検出を実行しない] フィルタを適用します。

- [時刻変化] を設定

[ファイルフィルタおよび属性] ペインで定義されている選択済みのフォルダ、サブフォルダ、およびファイルに対して [時刻変化] フィルタを適用します。

- [変更検出を実行しない] を解除

[ファイルフィルタおよび属性] ペインで定義されている選択済みのフォルダ、サブフォルダ、およびファイルに対して [変更検出を実行しない] フィルタを解除します。

- [時刻変化] を解除

[ファイルフィルタおよび属性] ペインで定義されている選択済みのフォルダ、サブフォルダ、およびファイルに対して [時刻変化] フィルタを解除します。

フォルダ、サブフォルダ、およびファイルに対して [変更検出を実行しない] フィルタおよび [時刻変化] フィルタが更新されているかどうかを、メッセージから確認できます。

**注:** フォルダに新しいサブフォルダまたはファイルを追加した場合、既存のフィルタは新しく追加したサブフォルダまたはファイルに適用されません。必要に応じて、新たに追加されたサブフォルダまたはファイルに新しいフィルタを適用します。

10. [ルール] タブをクリックします。

[ルール] タブでは、[管理対象] - [ファイル システム] オーバレイ内のファイルおよびディレクトリの値を制限するルールを定義します。ルールには、ユーザが作成した明示的な制約ルールと、事前定義済みの暗黙的な制約ルールの両方が含まれます。たとえば、エレメントに対して値またはデータ タイプを指定した場合、**CA Configuration Automation** は組み込みの [デフォルトの確認] ルールまたは [データ タイプの検証] ルールを自動的に作成します。

11. [ルール] タブの各 [フィールド](#) (P. 82) を設定します。

## ブループリント レジストリ フィルタおよび属性の定義

次の手順に従ってください:

1. [レジストリ フィルタおよび属性] リンクをクリックします。

[レジストリ フィルタおよび属性] ペインには `$(RegistryRoot)` フォルダが表示され、[`$(RegistryRoot)`] ペインには [優先順位] タブが表示されます。タブに、対応するテーブルの既存のキーと値が表示されます。

2. (オプション) ディレクトリの優先順位の設定に使用する行を選択し、[アクションの選択] ドロップダウン リストから [上へ移動] または [下へ移動] を選択します。

ディレクトリの順序が変更されます。順位付けのキーと値の重要度は、手順 18 で説明されている順位付けと同様です。

3. [キーの追加] をクリックして、[キー] ペインの [名前 (正規表現)] および [説明] フィールドを設定します。
4. [利用可能なカテゴリ] 列で 1 つ以上のオプションをダブルクリックします。[選択されたカテゴリ] 列に追加されます。
5. [利用可能なフィルタ] 列で 1 つ以上のオプションをダブルクリックします。[選択されたフィルタ] 列に追加されます。
6. [キー] ペイン内の残りのフィールドを設定して、[保存] をクリックします。

7. [レジストリ フィルタおよび属性] ペインで、値を割り当てるキーを選択します。
8. 右ペインで、[値の追加] をクリックして、[値] ペインの [名前 (正規表現)] および [説明] フィールドを設定します。
9. [利用可能なカテゴリ] 列のオプション (複数化) をダブルクリックして、[選択されたカテゴリ] 列に追加します。
10. [利用可能なフィルタ] 列のオプション (複数可) をダブルクリックして、[選択されたフィルタ] 列に追加します。
11. [値] ペイン内の残りのフィールドを設定して、[保存] をクリックします。

キーファイルが [レジストリ管理] ペイン内の `$(RegistryRoot)` ディレクトリの下に追加されます。

# 第 4 章: ブループリント エLEMENT リファレンス

---

このセクションでは、ソフトウェア コンポーネントの検出および管理に使用できる、さまざまなブループリント エLEMENT のリファレンスを提供します。

このセクションには、以下のトピックが含まれています。

[カテゴリの説明](#) (P. 29)

[フィルタの説明](#) (P. 31)

[POSIX 1003.2-1992 パターン マッチング](#) (P. 32)

[変数の代入](#) (P. 33)

[解釈方法の説明](#) (P. 40)

[正規表現](#) (P. 48)

[CA Configuration Automation で提供される Java プラグイン](#) (P. 52)

[表形式データ パーサの理解と使用](#) (P. 54)

## カテゴリの説明

[カテゴリ] フィールドでは、コンポーネント ブループリント のELEMENT を整理できます。

カテゴリ	説明
管理	コンポーネントの一般的な可用性および環境管理と関係する管理設定。管理設定の例は、バックアップの方法、キャッシュ フラッシュの時期、再試行の回数などです。
環境設定	環境管理、ログおよびデバッグ、またはパフォーマンスで詳細に説明されているものを除く、コンポーネントの構成の設定。構成の設定の例は、コンポーネントのエイリアス名またはデフォルト <b>Web</b> ページです。
マニュアル	コンポーネントの動作を記録する、またはユーザのガイドになるELEMENT。たとえば、マニュアル、 <b>readme</b> ファイル、FAQ、またはオンラインヘルプ ページ。
ログおよびデバッグ	ログの場所、ログ レベル、デバッグ出力、または診断の変数タイプなどの設定に関するELEMENT。

Network	コンポーネントのネットワーク関連設定を表す、または示すエレメント。たとえば、 <b>SNMP</b> のポート設定などです。[ネットワーク] と [セキュリティ] の両方に分類できるエレメント（たとえば <b>LDAP</b> 認証の有効化など）の場合、カテゴリとして [セキュリティ] を使用します。
その他	CA 内部でのみ使用されます。
パフォーマンス	パフォーマンスに重大な影響があるとわかっている設定で、通常は構成パラメータの特殊なサブセットです。パフォーマンス設定の例は、スレッドの数または同時ユーザ数になります。
製品情報	製品に関する全般（通常は静的な）情報。静的なコンポーネント情報の例は、ライセンス、インストール場所、ベンダー、またはモジュール名です。
リソース	コンポーネントのリソース。コンポーネントのリソースの例は、ストレージ、メモリおよびキャッシュの割り当てまたはサイズ、 <b>CPU</b> です。この静的なリソース カテゴリは、リアルタイム情報に近い [一時的] カテゴリとは異なることに注意してください。
セキュリティ	セキュリティ関連の設定を表すエレメントで、通常は構成パラメータの特殊なサブセットです。セキュリティ設定の例は、認証タイプ、認証の有効化、暗号化設定、ディレクトリ参照、 <b>SSL</b> 、または <b>HTTPS</b> です。
一時的	ある規則性と共に変化するエレメント。一時的なエレメントの例は、サーバ状態（たとえば、起動、ダウン、稼働中、停止）、現在接続されているクライアント数、現在のスレッド数、現在のディスク使用率などです。
バージョンおよびパッチ	製品のバージョンおよびパッチ レベルを示すエレメント。

## フィルタの説明

〔フィルタ〕フィールドでは、変更の検出およびルール コンプライアンスなど、主要な **CA Configuration Automation** の操作と結果から除外するコンポーネントブループリントエレメントをマークします。

フィルタ	説明
コンポーネント固有	単一のコンポーネント インスタンスに固有のエレメントを識別します。たとえば、インストールルート、サービス サーバ名などです。特定のコンポーネントに固有のエレメント（すでに固有と認識されているもの）を識別し、変更検出操作および結果から除外することが目的です。
サービス仕様	サービスに固有のエレメントを識別します。たとえば、サーバ名、インストールルートなどです。特定のサービスに固有のエレメント（すでに固有と認識されているもの）を識別し、変更検出操作および結果から除外することが目的です。
サーバ仕様	単一のサーバに固有のエレメントを識別します。たとえば、サーバ名、IP アドレスなどです。特定のサーバに固有のエレメント（すでに固有と認識されているもの）を識別し、変更検出操作および結果から除外することが目的です。
変更検出を実行しない	すべてのタイプの変更検出操作および結果から永続的に除外する必要があるエレメントを識別します。識別し、変更検出操作と結果から常に除外するエレメントの例は、一時ディレクトリ、管理対象フォルダ内のログ ファイル、または一時的であるとわかっているすべてのものです。
ルール コンプライアンスを実行しない	連続しない、または可変の性質であるため、すべてのタイプのルール コンプライアンス操作と結果から完全に除外するエレメントを識別します。識別し、ルール コンプライアンス操作と結果から常に除外するエレメントの例は、一時ディレクトリ、既知の古い構成ファイル、テンプレート、またはサンプル ファイルです。
時刻変化	ログ ファイル、プロセスの開始時刻、レジストリ イベント カウンタなどの、時間が経過すると変更されることがわかっているが、サーバ全体またはサービス全体に必要なではないエレメントを識別します。時刻変化のパラメータ（すでに固有と認識されているもの）を識別し、変更検出操作および結果から除外することが目的です。

## POSIX 1003.2-1992 パターン マッチング

POSIX パターン マッチング表現は、正確なファイル名またはディレクトリ名がわからない場合のファイルおよびディレクトリ検索に役立つ記述です。CA Configuration Automation は、ディスカバリ操作およびリフレッシュ操作中にファイルとディレクトリを検索するために POSIX パターン マッチング表現を使用します。

### POSIX パターン マッチング表現の構文

ファイル名のマッチング表現	説明
文字	
unicodeChar	同一の Unicode 文字に一致します。
?	任意の 1 文字に一致します。
*	NULL 文字列（長さゼロ）を含めた任意の文字列に一致します。
文字クラス	
[abc]	角かっこの中にリストされた任意の文字に一致します（単純な文字クラス）。
[a-zA-Z]	角かっこの中にリストされた任意の文字範囲に一致します（範囲のある文字クラス）。
[^abc]	角かっこの中にリストされた文字を除く任意の文字範囲に一致します（否定文字クラス）。
標準的な POSIX 文字クラス	
[:alnum:]	英数文字に一致します
[:alpha:]	英文字に一致します
[:blank:]	スペースおよびタブ文字に一致します
[:cntrl:]	制御文字に一致します
[:digit:]	数字に一致します
[:graph:]	印刷および表示ができる文字に一致します。（スペースは印刷はできますが表示ができないのに対し、「a」は両方が可能です。）



[lower:]	小文字の英文字に一致します
[print:]	印刷可能な文字（制御文字でない文字）に一致します
[punct:]	句読点（文字、数字、制御文字またはスペース文字でない文字）に一致します
[space:]	タブ、スペース、および改ページなど、スペースを作る文字に一致します
[upper:]	大文字の英文字に一致します
[xdigit:]	16進数の数字である文字に一致します

## 変数の代入

変数の代入は、CA Configuration Automation によって管理されている任意の要素の値を、コンポーネントブループリントディレクティブ内のパラメータとして使用することを許可します。CA Configuration Automation は、ブループリント内の要素を識別するために表現構文を定義します。いったん識別されると、ディレクティブの実行時に要素の値が抽出され、表現の代わりに使用されます。代入は、ディレクティブの任意の属性に適用できます。これらには値、デフォルト値、パス、ファイル名、パラメータ、環境変数、正規表現、クエリおよび列名が含まれます。

値を変数代入構文で宛先指定できる管理対象要素には、以下が含まれます。

- ディスカバリ パラメータ
- レジストリ変数
- 構成値（ファイル、データベースまたは実行可能ファイル）
- ファイルおよびディレクトリの属性
- 管理対象のデータ エlement（スキーマ メタデータ）
- サービスおよびコンポーネントの属性

### 表現タイプ

変数の代入には以下の形式があります。

#### パラメータ代入

現在のコンポーネントで定義されているパラメータの値にアクセスできるようにします。

#### オブジェクト代入

任意のサービスの任意の要素の値に対応できます。

#### グローバル変数の代入

CA Configuration Automation グローバル変数リポジトリからの値に対応できます。

以下のセクションでは、これらの代入のタイプについて説明します。

### パラメータ代入

パラメータ代入表現の形式は以下のとおりです。

`$(VariableName)`

*VariableName*

現在のコンポーネントで定義されているディスカバリ パラメータを指定します。名前は大文字と小文字が区別され、パラメータ名と完全に一致する必要があります。文字列リテラルにパラメータ表現を埋め込むか、スタンドアロンで使用できます。同じ表現で複数の代入を定義できます。また、それらを再帰的に定義することもできます。たとえば、代入値はパラメータ表現の文字列とすることができます。その場合、製品は代入値を再帰的に評価します。

**例:**

以下のディスカバリ パラメータを定義するとします。

```
User=info
Domain=ca.com
v1=$(User)
v2=$(Domain)
```

以下のパラメータ代入表現の場合

```
$(User)@$(Domain) [$(v1) at $(v2)]
```

評価後に以下の結果が返されます。

```
info@ca.com [info at ca.com]
```

## オブジェクト代入

オブジェクト代入表現では、**CA Configuration Automation** 管理対象エリメントツリーのオブジェクトへのパスが定義されます。オブジェクトが識別される場合、製品は表現の結果としてオブジェクト値を返します。または、製品は、コンポーネント スコープのオブジェクト代入表現の例のように、オブジェクトの属性を返すことができます。表現に一致するオブジェクトがない場合、製品は **NULL** 値を返します。

オブジェクト代入表現は、現在のコンポーネントのサービスのスコープで定義するか、グローバルに定義できます。

### サービス スコープのオブジェクト代入表現

サービス スコープのオブジェクト代入表現では、サービスに存在できるコンポーネントを指定する必要があります。サービス スコープのオブジェクト代入表現には以下の形式があります。

```
${Component[ComponentName,ElementType[ElementName or  
Identifier, ...]]}
```

#### 中かっこ {}

オブジェクト構文をパラメータ表現構文から区別します。

#### *ComponentName*

単一のコンポーネントの名前、またはコンポーネントのリスト ('|' 文字で区切る) のいずれかを定義します。区切られたリストのバリエーションは、サービス データベースに複数のコンポーネントが含まれる場合に、オブジェクト表現が値を解決することを許可します。

### 例:

SQL サーバまたは Oracle のいずれかのコンポーネントで、区切られたリストのバリエーションを使用する方法

```
${Component[Microsoft SQL Server|Oracle 8i  
Server,Parameter[DatabaseUser]]}
```

コンポーネントのルート パラメータにアクセスする方法

```
${Component[CCA Server,Parameter[Root]]}
```

また、コンポーネント ブループリント カテゴリによってコンポーネントを選択することもできます。

```
${ComponentCategory[Relational Databases,Parameter[DatabaseUser]]}
```

コンポーネント ブループリント ページでは、有効なカテゴリ名がリスト表示されます。

- アプリケーション プラットフォーム
- CA ソフトウェア
- クラスタ化
- コンプライアンス
- カスタム コンポーネント
- ディレクトリ サーバ
- 企業アプリケーション
- インポートされたコンポーネント
- IT 管理システム
- メッセージング システム
- ネットワーク デバイス
- オペレーティング システム
- オペレーティング システム - 制限あり
- リレーショナル データベース
- サーバ コンポーネント
- ストレージ マネージャ
- ユーティリティ

- 仮想化
- Web サーバ

### コンポーネント スコープのオブジェクト代入表現

コンポーネント スコープのオブジェクト代入表現には形式があります。

```
${ElementType[ElementName or Identifier, ...]}
```

**例:**

```
${FileSet[$(Root),Directory[admin/logs,File[filter.log,Attribute[size]]]]}
```

### グローバル スコープのオブジェクト代入表現

グローバル スコープのオブジェクト表現は、単一の CCA データベースの任意のサービスから情報にアクセスできます。 グローバル スコープのオブジェクト代入表現には形式があります。

```
${Service[ServiceBlueprintName(ServiceName),Component[ ... ]]}
```

**例:**

CA Configuration Automation 以外のサービスの構成パラメータから CA Configuration Automation メールを受け取る方法

```
${Service[CCA(MyCCA),Component[CCA Server,Configuration  
[*,Files[*,Directory[lib,File[cca.properties,FileStructure[*,NVFile  
[com.ca.mail.from]]]]]]]]}
```

### オブジェクト代入表現で利用可能な要素と属性

以下のツリーによる列挙

- 特定のエレメント タイプ
- ツリーの特定のエレメント タイプの関係
- オブジェクト値
- 利用可能な属性 (かっこ内)

この例の文字列は、オブジェクト代入表現に使用することができ、ツリー内のオブジェクトへのパスを構築します。

```
Component [name or id]
  (module_id, mod_name, mod_desc, mod_version, platform_id
  mod_instance_type, mod_instance_of, release_version, mod_state,
  created_by, creation_time, server_id, server_name, domain_name,
  ip_address, mac_address, server_state,
  cc_agent_yn, cc_agent_port,
  cc_agent_protocol, os_type, os_version, processor, platform_name)
Parameter [parameter name]
Files [$(Root)]
  Directory [directory name or path (a/b/c)]
    (name, mtime, ctime, owner, perm, bytes, depth, files,
    directories)
  Directory ...
  File
  File [file name]
    (name, mtime, size, owner, perm, prodver, filever, ctime)
  Registry [*]
    RegKey [keyname or path (a¥b¥c)]
      (name, value)
    RegKey ...
  RegValue [name]
    (name, value)
  Configuration [*]
    Files [*]
      File [name]
      FileStructure
      GroupFileBlock [name]
      GroupFileBlock [name(value)] value はグループ ブロックの
        値、名前修飾子、または名前修飾子の子の値。
      GroupFileBlock ...
      NVFileBlock [name]
      NVFileBlock [name]
        (description, view, weight, password, folder)
  Database [name]
    ResultSet [name]
```

```

        (name, type, query, queryType, description)
DataRow [name]
DataCell [name]
        (name, value)
DatabaseKey [name]
        (name, description, key, keyValues, column)
ExecutablesFileSystem [*]
File [name]
FileStructure
GroupFileBlock [name] または GroupFileBlock [name(value)]
    value はグループ ブロックの値、名前修飾子、または名前修飾子の
    子の値。
GroupFileBlock ...
NVFileBlock [name]
        (description, view, weight, password, folder)
Database [database name]
DataBaseAccessSpec
        (server, user, password, driver, databaseName,
        databaseContext, env)
Table [table name]
        (name, description, rowcount)
Column [column name](name, description, length, nullable,
default, ordinal, precision)
Index [index name]
        (name, sort, unique, description)
Column [column name]
        (name, description, length, nullable, default, ordinal,
precision)

```

## グローバル変数の代入

グローバル変数の代入表現は以下の形式です。

`$(GlobalVariableName)`

*GlobalVariableName*

**CA Configuration Automation** グローバル変数リポジトリの有効なパスを定義します。名前では大文字と小文字は区別されません。文字列にグローバル変数表現を埋め込むか、スタンドアロンで使用できます。同じ表現で複数の代入を定義できます。また、それらを再帰的に定義することもできます。たとえば、代入値がパラメータ表現での文字列である場合、アプリケーションは代入値を再帰的に評価します。

### 例:

以下の構造のグローバル変数リポジトリがあるとします。

```
グローバル変数
  サイト
    Phoenix
      Main: x4000
      Fire: x4911
    Tucson
      Main: x5000
      Fire: x5911
```

以下のグローバル変数代入表現の場合

```
$(/Site/Tucson/Main)
```

評価後に以下の結果が返されます。

```
x5000
```

## 解釈方法の説明

解釈方法は、**CA Configuration Automation** に構成パラメータの文字列形式に関するヒントおよび関連するコンポーネントによるその意図的な使用を提供します。アプリケーションでは、解釈されたパラメータ値を調べるために状況依存パーサを使用します。パーサによって複雑なパラメータ文字列から複数のサブ値が抽出されます。

たとえば、**CA Configuration Automation** が以下の値を **JDBC URL** として解釈して抽出可能な場合、データベース タイプ、サーバ、ポート、データベース名を抽出できます。

```
jdbc:oracle:thin:@dbserver:1521:MYDBNAME
```

コンテキスト依存の解析を有効にすることに加え、解釈は関係の派生も可能にします。上記の例で抽出されたサーバを使用し、現在のサーバとサーバ **dbserver** の間で関係を確立することができます。関係は、関係キーを使用して確立します。



値ごとに1つの解釈のみが可能です。解釈のない値も多くあります（そのような値は未解釈のままにします）。複数の解釈が該当する場合（たとえば、「ファイル名」と「ファイル名またはパス」）、フィールドを最も正確に説明しているものを使用します。たとえば、フィールドがファイル名として（パスなしで）定義されている場合は、「ファイル名」を選択します。フィールドにファイル名、パス、または部分的なパスが含まれる場合は、「ファイル名またはパス」を選択します。アプリケーションには選択された以下の解釈方法が含まれます。

### データベース名

値はデータベース サーバ内のデータベースの名前です。

この解釈から派生する関係はないので、[関係キー] は常に「いいえ」に設定します。

### データベース テーブル

値はデータベース内のデータベース テーブルの名前です。データベース テーブルにはスキーマ プレフィックスが含まれる場合があります。

この解釈から派生する関係はないので、[関係キー] は常に「いいえ」に設定します。

### 日付

値は任意の形式の日付です。

この解釈から派生する関係はないので、[関係キー] は常に「いいえ」に設定します。

### 日付および時刻

値は日付と時刻の組み合わせです。

この解釈から派生する関係はないので、[関係キー] は常に「いいえ」に設定します。

### 説明

アプリケーションは値を説明文として解釈します。

この解釈から派生する関係はないので、[関係キー] は常に「いいえ」に設定します。

### ディレクトリ名

値はパスのないディレクトリのみです。

この解釈から派生する関係はないので、[関係キー] は常に「いいえ」に設定します。

### ディレクトリ名またはパス

値はディレクトリ名、パス、または部分的なパスです。

この解釈から[ディレクトリ参照]関係が派生する可能性があるので、[関係キー]は「はい」に設定できます。

### 電子メール アドレス

値は電子メール メッセージの宛て先です。インタープリタは値の文字列内の1つ以上の電子メールアドレスを検索します。

### ファイル名

値はパスのないファイル名のみです。

この解釈から派生する関係はないので、[関係キー]は常に「いいえ」に設定します。

### ファイル名またはパス

値はファイル名、パス、または部分的なパスです。多くの指定された値が、これらの任意の解釈を許可します。

この解釈から[ファイル参照]関係が派生する可能性があるので、[関係キー]は「はい」に設定できます。

### サーバ名または IP アドレス

値はIPアドレスまたはサーバ名であるか、またはこれらが含まれます。この解釈は、ポート番号が値に定義されていない場合に限って使用します。値にポート番号が含まれる場合は、「サーバ名およびポート」を使用します。

アプリケーションは、より長い文字列に埋め込まれているサーバ名およびIPアドレスを認識できます。

この解釈から[サーバ参照]関係が派生する可能性があるので、[関係キー]は「はい」に設定できます。

参照されるサーバが現在のサーバに依存関係があると考えられる場合のみ、関係キーとして[サーバ参照]関係を定義します。

### サーバ名およびポート

値は、サーバ名または IP アドレスおよびポート番号であるか、その値を含みます。コロン（:）を使用してサーバとポート番号を区切る必要があります。

アプリケーションは、より長い文字列に埋め込まれているサーバ名、IP アドレスおよびポート番号を認識できます。

この解釈から [サーバ参照] 関係が派生する可能性があるので、[関係キー] は「はい」に設定できます。

参照されるサーバが現在のサーバに依存関係があると考えられる場合のみ、関係キーとして [サーバ参照] 関係を指定します。

### Java クラス名

値は Java クラス名です。クラス名、パッケージ名、またはパッケージプレフィックスを持つ完全修飾クラス名である可能性があります。

この解釈から派生する関係はないので、[関係キー] は常に「いいえ」に設定します。

### JDBC URL

値は JDBC URL を定義します。テーブルには、サポートされる形式が示されます。

この解釈から [サーバ参照] 関係が派生する可能性があるので、[関係キー] は「はい」に設定できます。

JDBC URL は、ほぼ必ず重要な関係を定義します。それらは通常関係キーとして識別される必要があります。

### LDAP のパス

値は LDAP サブツリーへのパスを定義します。

この解釈から派生する関係はないので、[関係キー] は常に「いいえ」に設定します。

### LDAP エントリ

値は、LDAP ディレクトリ エントリの名前またはエントリのフルパスです。

この解釈から派生する関係はないので、[関係キー] は常に「いいえ」に設定します。

### ネットワークドメイン

値はネットワーク ドメイン（サーバ名を含まない）です。例：

`ca.com`

この解釈から派生する関係はないので、[関係キー] は常に「いいえ」に設定します。

### ネットワークプロトコル

値は、TCP、UDP、FTP、SNMP または SMTP などの IP プロトコルを定義します。

この解釈から派生する関係はないので、[関係キー] は常に「いいえ」に設定します。

### パスワード

値はパスワードです。

この解釈から派生する関係はないので、[関係キー] は常に「いいえ」に設定します。

### レジストリ キー名

値はパスのないレジストリ キーの名前のみです。

この解釈から派生する関係はないので、[関係キー] は常に「いいえ」に設定します。

### レジストリ キー パス

値はレジストリ キーのフルパス（¥で開始）です。

この解釈から派生する関係はないので、[関係キー] は常に「いいえ」に設定します。

### レジストリ値名

値はパスのないレジストリ 値の名前のみです。

この解釈から派生する関係はないので、[関係キー] は常に「いいえ」に設定します。

### レジストリ値パス

値はレジストリ 値のフルパス（¥で開始）です。

この解釈から派生する関係はないので、[関係キー] は常に「いいえ」に設定します。

### SNMP コミュニティ文字列

値は SNMP コミュニティ文字列を指定します。例：

public

この解釈から派生する関係はないので、[関係キー] は常に「いいえ」に設定します。

### SNMP OID

値は SNMP オブジェクト ID を指定します。例：

1.3.6.1.4.1.18071.1.1.1

この解釈から派生する関係はないので、[関係キー] は常に「いいえ」に設定します。

### TCP ポート番号

値は TCP ポート番号です（UDP または未指定ではありません）。

この解釈から派生する関係はないので、[関係キー] は常に「いいえ」に設定します。

### 時間間隔

値は時間の間隔です。

この解釈から派生する関係はないので、[関係キー] は常に「いいえ」に設定します。

### 時刻

値は時刻です。

この解釈から派生する関係はないので、[関係キー] は常に「いいえ」に設定します。

### UDP ポート番号

値は UDP ポート番号です（TCP または未指定ではありません）。

この解釈から派生する関係はないので、[関係キー] は常に「いいえ」に設定します。

## URL

値は、次のプロトコルを含む URL を指定します: file、http、https、ftp、jrmis、jmx:rmi、iiop、gopher、news、telnet、mailto、jnp、t3、ldap。

インタプリタは、URL を分解し、その一部をカスタム メソッドによって利用可能にします。

この解釈から [サーバ参照] 関係が派生する可能性があるので、[関係キー] は「はい」に設定できます。

URL 関係は、以下の場合のみ関係キーとして定義します。

- URL にサーバ名が含まれる
- 名前が指定されたサーバは、常に現在のサーバと指定されたサーバの間の依存性を定義します。

## ユーザ グループ

値はユーザ グループです。

この解釈から派生する関係はないので、[関係キー] は常に「いいえ」に設定します。

## ユーザ名

値はユーザ名です。

この解釈から派生する関係はないので、[関係キー] は常に「いいえ」に設定します。

## バージョン文字列

値はバージョンとして解釈できるあらゆる文字列です。

この解釈から派生する関係はないので、[関係キー] は常に「いいえ」に設定します。

## Web サービス URL

値は、コンポーネントが使用する Web サービスを特定する URL を指定します。

この解釈から [サーバ参照] 関係が派生する可能性があるので、[関係キー] は「はい」に設定できます。

JDBC URL 関係解釈は以下の形式をサポートします。

データベース名	URL パターン
SQL Server2005 SQL Server 2008	jdbc:sqlserver://<host>:<port>;datasenname=<database>; SendStringParametersAsUnicode=false
Oracle 9、10、および 11	<ul style="list-style-type: none"> <li>■ jdbc:oracle:thin:@\$(host):\$(port):\$(database)</li> <li>■ jdbc:oracle:thin:@&lt;host&gt;:&lt;port&gt;:&lt;database&gt;</li> <li>■ jdbc:oracle:thin:@&lt;host&gt;:&lt;port&gt;/&lt;database&gt;</li> <li>■ jdbc:oracle:thin:@((DESCRIPTION=(ADDRESS_LIST=(ADDRESS=(PROTOCOL=TCP)(HOST=&lt;host1&gt;)(PORT=&lt;port1&gt;)(ADDRESS=(PROTOCOL=TCP)(HOST=&lt;host2&gt;)(PORT=&lt;port2&gt;))(FAILOVER=ON)(LOAD_BALANCE=OFF)(CONNECT_DATA=(SERVER=DEDICATED)(SERVICE_NAME=&lt;database&gt;)))</li> <li>■ jdbc:oracle:thin:@((DESCRIPTION=(ADDRESS_LIST=(ADDRESS=(PROTOCOL=TCP)(HOST=&lt;host&gt;)(PORT=&lt;port&gt;))) (CONNECT_DATA=(SERVICE_NAME=&lt;database&gt;)))</li> <li>■ jdbc:oracle:thin:@(DESCRIPTION=(ADDRESS_LIST=(ADDRESS=(PROTOCOL=TCP)(HOST=&lt;host&gt;)(PORT=&lt;port&gt;))) (CONNECT_DATA=(SERVICE_NAME=&lt;database&gt;)))</li> <li>■ jdbc:bea:oracle://&lt;host&gt;:&lt;port&gt;</li> </ul>
Informix	jdbc:informix-sqli://<host>:<port>/<database>: informixserver=<serverName
DB2	jdbc:db2://<host>:<port>/<database>
Sybase 11 および 15	jdbc:sybase:Tds:<host>:<port>/<database>
MySQL	<ul style="list-style-type: none"> <li>■ jdbc:mysql://&lt;host&gt;:&lt;port&gt;/&lt;database&gt;</li> <li>■ jdbc:mysql://&lt;host&gt;:&lt;port&gt;</li> </ul>
Postgres	jdbc:postgresql://<host>:<port>/<database>
HSQLDB	jdbc:hsqldb:hsq://<host>:<port>
ODBC	jdbc:odbc:<database>
Cloudscape	jdbc:cloudscape:<database>
Java DB (Derby)	<ul style="list-style-type: none"> <li>■ jdbc:derby://&lt;host&gt;:&lt;port&gt;/&lt;database&gt;</li> <li>■ jdbc:derby://&lt;host&gt;:&lt;port&gt;/&lt;database&gt;;create=true</li> </ul>
Ingres	jdbc:ingres://<host>:<port>/<database>

データベース名	URL パターン
Pointbase	jdbc:pointbase:server://<host>:<port>/<database>
汎用	jdbc:<xyz>:server://<host>:<port>/<database>

## 正規表現

正規表現は、綿密な文字列の一致を有効にするパターンの記述です。CA Configuration Automation は、以下に対して正規表現を使用します。

- ファイル内の一致する文字列を検索する
- 文字列が電子メールアドレスのような特定のパターンに一致することを検証する
- 大きなテキストブロックから文字列値を抽出する

正規表現の背景の概念について、さらに情報が必要な場合は **Web** に多数のソースがあります。たとえば、コンピュータ プログラミング言語に関する Google ディレクトリには正規表現についての有用なセクションがあります

([http://directory.google.com/Top/Computers/Programming/Languages/Regular\\_Expressions/FAQs, Help, and Tutorials](http://directory.google.com/Top/Computers/Programming/Languages/Regular_Expressions/FAQs,_Help,_and_Tutorials))。

### 正規表現の構文

以下の表は、CA Configuration Automation の正規表現用でサポートされている構文を示します。

正規表現	説明
文字	
unicodeChar	同一の Unicode 文字に一致します。
¥ (円記号)	メタ文字を引用するか、特殊文字を通常またはリテラルテキストとして処理するために使用します。 たとえば、¥* は、アスタリスクをワイルドカードではなく通常のテキスト文字にします。
¥¥	1 つの ¥ 文字に一致します。
¥0nnn	指定した 8 進法文字に一致します。



¥xhh	指定した 8 ビットの 16 進数の文字に一致します。
¥uhhhh	指定した 16 ビットの 16 進数の文字に一致します。
¥t	ASCII タブ文字に一致します。
¥n	ASCII 改行文字に一致します。
¥r	ASCII リターン文字に一致します。
¥f	ASCII 改ページ文字に一致します。
<b>文字クラス</b>	
[abc]	角かっこ内の文字に一致します（単純な文字クラス）。
[a-zA-Z]	角かっこ内の文字の範囲に一致します（範囲を持つ文字クラス）。
[^abc]	角かっこ内の文字を除く文字の範囲に一致します（文字クラスの否定）。
<b>標準的な POSIX 文字クラス</b>	
[:alnum:]	英数文字に一致します。
[:alpha:]	英文字に一致します。
[:blank:]	スペースおよびタブ文字に一致します。
[:cntrl:]	制御文字に一致します。
[:digit:]	数字に一致します。
[:graph:]	印刷および表示ができる文字に一致します。（スペースは印刷はできますが表示ができないのに対し、「a」は両方が可能です。）
[:lower:]	小文字の英文字に一致します。
[:print:]	印刷可能な文字（制御文字以外の文字）に一致します。
[:punct:]	句読点（文字、数字、制御文字、スペース文字でない文字）に一致します。
[:space:]	スペースを作る文字に一致します（例：タブ、スペース、改ページ文字など）。
[:upper:]	大文字の英文字に一致します。
[:xdigit:]	16 進数の数字である文字に一致します。
<b>標準以外の POSIX スタイル文字クラス</b>	
[:javastart:]	Java 識別子の先頭に一致します。

[;javapart:]	Java 識別子の一部に一致します。
<b>事前定義済みクラス</b>	
.	改行以外のすべての文字に一致します。
¥w	「単語」文字（英数字と「_」）に一致します。
¥W	非単語文字に一致します。
¥s	空白文字に一致します。
¥S	非空白文字に一致します。
¥d	10 進の数字に一致します。
¥D	10 進の数字ではない文字に一致します。
<b>境界は一致します。</b>	
^	文字列の先頭のみ一致します。
\$	文字列の最後のみ一致します。
¥b	単語境界の先頭または最後の文字に一致します。
¥B	単語境界の先頭または最後でない文字に一致します。
<b>最長一致閉包</b> (量指定子としても知られています。 詳細については、表の下の注を参照してください。)	
A*	A にゼロ回以上一致します。
A+	A に 1 回以上一致します。
A?	A にゼロ回または 1 回一致します。
A{n}	A に n 回のみ一致します。
A{n,}	A に n 回以上一致します。
A{n,m}	A に n 回以上 m 回以下一致します。
<b>non-greedy な繰り返し</b> (量指定子としても知られています。 詳細については、表の下の注を参照してください。)	
A*?	A にゼロ回以上一致します。
A+?	A に 1 回以上一致します。
A??	A にゼロ回または 1 回一致します

論理オペレータ	
AB	A の後に B が続くものと一致します。
A B	A または B のどちらかと一致します。
(A)	丸かっこは部分表現をグループ化するために使用されます。
逆参照	
(前のグループ化演算子が一致したものに帰り、何らかの一致のために再度使用します。)	
¥1	かっこで囲まれた最初の部分表現一致への逆参照。
¥2	かっこで囲まれた 2 番目の部分表現一致への逆参照。
¥3	かっこで囲まれた 3 番目の部分表現一致への逆参照。
¥4	かっこで囲まれた 4 番目の部分表現一致への逆参照。
¥5	かっこで囲まれた 5 番目の部分表現一致への逆参照。
¥6	かっこで囲まれた 6 番目の部分表現一致への逆参照。
¥7	かっこで囲まれた 7 番目の部分表現一致への逆参照。
¥8	かっこで囲まれた 8 番目の部分表現一致への逆参照。
¥9	かっこで囲まれた 9 番目の部分表現一致への逆参照。

注: すべての閉包演算子 (+、\*、?、{m,n}) は、デフォルトで「最長一致」です。つまり、全体的な一致を失敗させずに、できる限り多く文字列の要素と一致させます。non-greedy な閉包を使用するには、? (疑問符) を後ろに付けます。

## CA Configuration Automation で提供される Java プラグイン

必要に応じて、`com.ca.catalyst.object.CCICatalystPlugin` インターフェースを実装する Java プラグインは、ディレクティブ値をフィルタできます。プラグインを開発して `CLASSPATH` に追加するか、CA Configuration Automation で標準提供される以下のいずれかのプラグインを使用できます。

`com.ca.catalyst.plugin.CCParameterRuleFilter(pattern)`

**Version** パラメータをフォーマットします。このフィルタは、**Version** という名前のディレクティブを認識して変更するだけです。

たとえば、ファイルから最初に抽出された **Version** 値が **530** である場合、`CCParameterRuleFilter(##.##)` プラグインを指定すると、**Version** が **5.3.0** に変換されます。`CCParameterRuleFilter(###)` を指定した場合は、**Version** が **5.30** に変換されます。

`CCMatch(regex)`

`CCMatchAnywhere(regex)`

指定された正規表現 (**regex**) とディレクティブ値を一致させることにより、**"true"** または **"false"** 値を返します。

- 正規表現が値全体に完全に一致する場合、`CCMatch()` は **"true"** を返します。
- 正規表現が値のいずれかに一致する場合、`CCMatchAnywhere()` は **"true"** を返します。

正規表現は **DOTALL** および **MULTILINE** モードが有効な場合に解釈されます。**DOTALL** モードは、正規表現文字「**.**」が行末文字を含むすべての文字に一致することを前提としています。**MULTILINE** モードは、正規表現文字「**^**」および「**\$**」が先頭から終わりまでの値全体ではなく、行を区切ることを前提としています。

`CCReplaceAll("regex","replacement")`

`CCReplaceFirst("regex","replacement")`

正規表現に一致する値の部分を置換します。

正規表現 (**regex**) および置換 (**replacement**) の部分を引用符で囲み、カンマで区切ります。正規表現の値をキャリッジリターンで置き換えるには、置換文字に特殊文字「**¥n**」を使用します。たとえば、`CCReplaceAll(" ", "¥n")` はスペースをすべてキャリッジリターンに置き換えます。

**CCToUpper****CCToLower**

ディレクティブ値を大文字または小文字に変換します。

**CCTrim**

先頭および末尾のスペースを値から削除します。

**CCExpression**

ディレクティブの値が含まれる指定された表現を実行します。

表現は ECMA スクリプト (JavaScript) で記述されており、その言語の Version 2 で有効な任意の構文を含めることができます。表現には `$(VALUE)` として現在のパラメータの値を含めます。パラメータには値が必要で、値がないとアプリケーションはプラグインを呼び出しません。

表現では変数の代入が可能です。例：

`CCExpression($(VALUE)*50)`

値を 50 で乗算します。

`CCExpression('$(VALUE)' == 'XXX')`

true または false を返します。

`CCExpression(Math.sqrt($(VALUE)))`

値の平方根をとります。

`CCExpression(function add(a,b){return a+b;};add($(VALUE),$(Other));)`

関数を定義してコールします。

## 表形式データ パーサの理解と使用

表形式データとは、列と行でフォーマットされているすべてのテキストです。表形式データには、埋め込みコメントおよび1つ以上の見出し行も含めることができます。

表形式データの例

- netstat コマンドの出力

```
# netstat -ant

Active Internet connections (servers and established)
Proto Recv-Q Send-Q Local Address   Foreign Address State
tcp      0      0 0.0.0.0:512     0.0.0.0:*       LISTEN
tcp      0      0 0.0.0.0:32768   0.0.0.0:*       LISTEN
tcp      0      0 127.0.0.1:32769 0.0.0.0:*       LISTEN
tcp      0      0 0.0.0.0:513     0.0.0.0:*       LISTEN
tcp      0      0 0.0.0.0:2401    0.0.0.0:*       LISTEN
tcp      0      0 0.0.0.0:514     0.0.0.0:*       LISTEN
tcp      0      0 0.0.0.0:9188    0.0.0.0:*       LISTEN
tcp      0      0 127.0.0.1:8005  0.0.0.0:*       LISTEN
```

- Excel スプレッドシートからのタブ区切りの出力

Host	Address	Type	Owner
Bertha	192.168.123.12	Linux	Jerome
Factotum	192.168.123.33	Windows 2008	Bukowski
Terrapin	192.168.124.13	AIX	Hunter

CA Configuration Automation は、構成ファイルまたは実行可能ファイル内にあるすべての形式の表形式データを解釈および解析する表形式データパーサを提供します。また、表形式データセットの行と列のレイアウト、列名の割り当て、ヘッダおよびコメントテキストの削除を制御するパーサ オプションを指定できます。同様にデータ階層の整理もできます。

ユーザは、構造クラス レベルで表形式データ パーサを使用し、またパーサ オプションを指定することもできます。ただし、ファイル レベルおよび実行レベルの割り当てが、構造クラス レベルで行われた割り当てよりも優先されます。

## 表形式データ パーサへのアクセスと使用

表形式データ パーサを使用するには、以下の手順に従います。

1. クラス、ファイル、または実行可能属性シートで、[パーサ] ドロップダウンリストから [表形式データ パーサ] を選択します。 -
2. [パーサ オプション] で表レイアウトの詳細を定義します。以下の図はパーサ詳細を示します。

パーサ詳細	
パーサ: 表形式データ パーサ (日本語)	
<span>追加</span> <span>削除</span>	
名前	値
パーサ オプション: 列区切り文字	,
列名	Protocol:0,Recv:1,Send:2,Local:3,Remc
コメントの正規表現	#

[パーサ オプション] はドロップダウンでリスト表示される属性のセットです。[追加] または [削除] を使用してオプションの追加、削除を行います。オプションを編集するには [名前] および [値] のオプションをクリックします。

たとえば、図の中でリストされているパーサ オプションは以下の通りです。

- 列区切り方法
- 列名
- コメントの正規表現

3. [保存] をクリックします。  
[パーサ オプション] フィールドが更新されます。

## パーサ オプション

[パーサ オプション] フィールドの [表形式データ パーサ] に対して以下のオプションを設定できます。

**注:** オプションに指定する値はすべてかっこで囲みます。

**Column delimiter characters=**

1 つ以上の列の区切り文字を指定します。

このオプションを定義しない場合、デフォルトでタブ文字になります。

1 つ以上の列の区切り文字を指定することもできます。

たとえば、コロン、スラッシュ、およびカンマを有効な区切り文字として指定するには、以下のように指定します。

**Column delimiter characters=:、/、,**

たとえば、スペース、タブ、またはその両方を区切り文字として指定するには、以下のように指定します。

### スペースのみ

**Column delimiter characters=" "**

### タブのみ

**Column delimiter characters=" "**

### タブおよびスペース

**Column delimiter characters=" "**

**注:** 引用符 (" ") は、わかりやすく表示するために使用されているだけです。実際の操作では、スペースまたはタブ文字のみを引用符で囲まずに入力してください。



**Column delimiter method=**

連続する区切り文字を処理する方法を定義します。このオプションは "one" または "all" に設定できます。

このオプションを定義しない場合、デフォルトで "all" になります。その場合、連続する区切り文字を 1 つの区切り文字として処理します。たとえば、以下のように指定した場合

```
column delimiter characters=,  
column delimiter method=all
```

対象データ

```
ftp,tcp,udp,,,xyz
```

パーサは ftp、tcp、udp、xyz の 4 つの列を返します。

複数の連続するスペースを 1 つの区切り文字として処理する場合、または定義された値がなくてもデータ列を含める場合には値を "one" に設定します。たとえば、以下のように指定した場合

```
column delimiter characters=:  
column delimiter method=one
```

対象データ

```
root::0:XDCGBH!:
```

パーサは以下の列を返します。

```
root, "", 0, XDCGBH!, ""
```

**注:** 前の例で column delimiter method=all を指定した場合、パーサは以下の列のみを返します。

```
root, 0, XDCGBH!
```

**Header count=**

データセットの最初に見捨てる行番号を定義します。たとえば、次のように使用します。

```
Header count=2
```

これは、netstat コマンドの表形式データ結果からヘッダ情報を排除します。

**Header count=** オプションを指定した場合のみ、行全体を排除できます。このファイルの解析により、削除した行は CA Configuration Automation UI に含まれません。

### Comment regular expression=

データ セット内のコメントを識別する正規表現を定義します。たとえば、以下のように指定した場合

**Comment regular expression=#.\***

パーサは、**#** で開始されるパターン（行の末尾までのすべての文字を含む）をコメントとして解釈および無視します。例：

```
#  
# These three lines are removed  
#
```

このオプションを使用して、行の一部を解釈および無視することもできます例：

```
何らかのデータ    # this comment is also removed
```

このファイルの解析により、削除した行および行の一部は **CA Configuration Automation UI** に表示されません。

### Column Names=

データの個別の列に名前の割り当てを定義します。フィールドの形式は、名前および列のインデックス番号のカンマ区切りのリストです。列のインデックスはゼロから開始されます。例：

**"Column Names"=Protocol:0,Recv:1,Send:2,Local:3,Remote:4,State:5**

このファイルの解析により、**Column Names=** オプションで除外した列は **CA Configuration Automation UI** に表示されません。

**CA Configuration Automation** の標準内部データ形式に対して表形式データを解析することの重要な側面は、データを構造クラス グループとして組み立てることにあります。グループを使用することにより、データの各行に一意の修飾子を割り当てて、入れ子にし、ユーザ インターフェースに階層的に表示することができます。表形式データ パーサは、**Column Names=** オプションに定義された「**first name:index**」のペアを使用して、データ行が含まれるグループに名前を付けます(グループ ピボット)。

前の例で **Column Names=** オプションを指定し、**netstat** コマンド出力を入力として使用した場合、以下のデータが表示されます。



注: この例の解析済みデータを解釈するために使用される構造クラスは、**Protocol** の修飾子として **Local** を指定します。

トップレベルのグループ ピボット下のサブグループを構成するために複数の列をグループ化することもできます。たとえば、階層の 1 レベルに **Recv** と **Send** の列をネストするには、以下のようにグループ修飾子を適用します。

"Column

Names"=Protocol:0,Send/Recv(group):1-2,Local:3,Remote:4,State:5

データは以下のとおりに表示されます。



**注:** この例に示されているように、ネストされた値は、指定されたグループに属する値として表示され、名前がありません。名前がないと、親グループを正確に修飾する構造クラスを書き込むことが完全にはできない可能性があります。ネストされた列に名前を定義するには、ネストされた列のグループ修飾子の後に **name:index** ペアを指定します。  
例 :

"Column Names"=

Protocol:0,Send/Recv(group):1-2,Recv:1,Send:2,Local:3,Remote:4,State:5

データは以下のように表示されます。



name:index ペアの指定では、列の範囲、個別の列のリスト、またはその組み合わせとしてグループを定義できます。有効なグループ列の形式には以下が含まれます。

5-

列 5 およびそれ以降のすべての列

3-5|7-

列 3 ～ 5、列 7、およびそれ以降のすべての列

3-5|7|9-11

列 3 ～ 5、列 7、および列 9 ～ 11

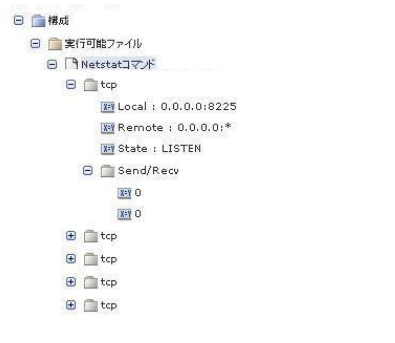
注: 階層が構築される基盤のグループ ピボットであるため、最初の name:index ペアにはグループ オプションを指定できません。

name:index ペアが指定する名前の列の値に代入するには、valueasname 修飾子を使用します。グループピボット（最初の name:index ペア）に valueasname 修飾子を指定できます。表の中の 1 つの列が通常一意のキーになるため、これは表形式のデータ表示では便利な方法です。たとえば、以下のように valueasname 修飾子を適用します。

"Column

Names"=Protocol(valueasname):0,Send/Recv(group):1-2,Local:3,Remote:4,State:5

データが表示されます：



**Line continuation regular expression=**

正規表現を定義して、行継続構文を識別します。

たとえば、行継続文字 `¥` を使用して、以下のファイル内の複数の行にわたって単一のデータの行を継続させます。

名前	電話	メール	アドレス
Fred	9000	<a href="mailto:Fred@acme.com">Fred@acme.com</a>	12 Jones Ct.
Anne	3002	<a href="mailto:Anne@acme.com">Anne@acme.com</a>	33535 Eucalyptus Terrace
Mary	7331	<a href="mailto:Mary@acme.com">Mary@acme.com</a>	31 Main Street

データを正しく解析するには、以下の **Line continuation regular expression=** オプションを使用します。

**Line continuation regular expression=** `¥¥$`

注: 最初の `¥` は、2 番目の `¥` をエスケープして有効な正規表現を生成します。





# 付録 A: ブループリント ウィザード UI リファレンス

---

このセクションには、以下のトピックが含まれています。

[\[ブループリント\] ページ: コンポーネントブループリント フィールド \(P. 66\)](#)  
[\[ディスカバリ方式\] ページ: 検索オプション フィールド \(P. 68\)](#)  
[\[ディスカバリ方式\] ページ: ファイルインジケータ フィールド \(P. 69\)](#)  
[\[ディスカバリ方式\] ページ: レジストリ インジケータの検索オプション フィールド \(P. 70\)](#)  
[\[ディスカバリ方式\] ページ: レジストリ インジケータ フィールド \(P. 71\)](#)  
[\[ディスカバリ方式\] ページ: ネットワーク プローブ フィールド \(P. 71\)](#)  
[\[ディスカバリ検証ルール\] ページ: ディスカバリ検証ルール フィールド \(P. 72\)](#)  
[\[管理\] ページ: ファイル管理オプション フィールド \(P. 81\)](#)  
[\[管理\] ページ: ディレクトリ フィールド \(P. 81\)](#)  
[\[管理\] ページ: ディレクトリ フィールド \(P. 82\)](#)  
[\[ファイルフィルタおよび属性\] ページの \[ルール\] タブの フィールド \(P. 82\)](#)  
[レジストリ管理 フィールド \(P. 85\)](#)  
[\[レジストリ フィルタおよび属性\] ページのキーの追加 フィールド \(P. 86\)](#)  
[\[レジストリ フィルタおよび属性\] ページの値詳細 フィールド \(P. 88\)](#)  
[\[データベース\] ページの フィールド \(P. 90\)](#)  
[\[コンポーネント パラメータおよび変数\] ページの フィールド \(P. 93\)](#)  
[\[構成\] - \[ファイル解析\] ページ \(P. 106\)](#)  
[構成実行ファイル ページ \(P. 106\)](#)  
[クエリの追加 ペイン \(P. 113\)](#)  
[ファイル構造クラス タブ \(P. 115\)](#)  
[\[File Structure Class Group\] フィールドと \[パラメータ\] フィールド \(P. 116\)](#)  
[マクロ ページ \(P. 120\)](#)  
[完了 ページ \(P. 121\)](#)

## [ブループリント]ページ: コンポーネント ブループリント フィールド

[ブループリント] ウィザードの [ブループリント] ページの [コンポーネント ブループリント] ペインには以下のフィールドが含まれます。

### コンポーネント ブループリント名

ブループリントの一意の名前を定義します。

**制限:** ブループリントの名前には、文字<、>、;、:、'"'\*+=、¥、/|、? を含むことはできません。

### コンポーネント バージョン

ソフトウェア コンポーネント バージョン (リリース番号) を定義します。

- 単一のブループリントで特定のコンポーネントの全バージョンをサポートするときは、このフィールドを「\*.\*」に設定します。
- それ以外の場合は、サポートされる特定のバージョンまたはバージョン シリーズを定義します (例: 1.2、2.\*)。

デフォルト: \*.\*

### ブループリント バージョン

ユーザの好みの形式で、ブループリントのバージョンを定義します。

デフォルト: 1.0.0

### ディスカバリ

ブループリントでディスカバリを有効にするか無効にするかを指定します。

デフォルト: 有効

### 説明

ブループリント項目を説明します。ブループリントの [詳細] ページには、説明が表示されます。

## オペレーティング システム

ソフトウェア コンポーネントが実行するオペレーティング システムを指定します。選択したオペレーティング システムに応じて、ブループリントは以下のようになります。

- 特定のオペレーティング システムをサポートします（たとえば、ファイル システム構造が全プラットフォームで大幅に異なる場合、または **Windows** レジストリに大きく依存している場合）。
- 複数のオペレーティング システムで動作します（ファイル システム パスおよび構成パラメータを正規化できる場合）。
- 特に、**F5 BIG-IP** ロード バランサをサポートします。
- 特に、**IOS** を使用する **Cisco** のルータおよびネットワーク スイッチをサポートします。

いつでもオペレーティング システムに固有のファイルを追加または削除できます。ただし、ターゲット オペレーティング システムを選択すると、より一般的なコンポーネント ブループリントの基本構造で開始できます。例：

- **[Windows]** を選択した場合は、ウィザードによってレジストリ関連またはレジストリ オーバーレイ関連のコンポーネントが自動的に作成されます。
- **[任意の UNIX]** または別の **UNIX** ベースのオペレーティング システムを選択した場合、ウィザードはレジストリ関連またはレジストリ オーバーレイ関連のコンポーネントを作成しません。

デフォルト：任意

## カテゴリ

[コンポーネント ブループリント] ページに新しいブループリントがリスト表示されるコンポーネントのカテゴリを指定します。注：事前定義済みブループリントからカスタム ブループリントを分離するには、**[カスタム コンポーネント]** カテゴリを使用します。カテゴリを分けると、ブループリントを容易に見つけることができ、誤って事前定義済みブループリントを編集しないようにできます。

デフォルト：カスタム コンポーネント

## [ディスカバリ方式] ページ: 検索オプション フィールド

[ブループリント] ウィザードの [ディスカバリ方式] ページの [新規検索オプションの追加] ペインには、以下の検索オプション フィールドが含まれます。

### 検索開始場所

検索の開始地点を定義します。この値を設定しないと、検索はコンポーネントのファイルシステムの最上位から開始します。

Windows の場合、ファイルシステムの最上位 (/) にはすべての物理ドライブ (C:、D: など) が含まれます。

デフォルト : / (root)

### 検索階層数

検索に含めるフォルダまたはディレクトリの階層数を定義します。

デフォルト : 10

### コンポーネント常時検出

プロセスが常にコンポーネントを検出済みと考慮するかどうかを指定します。

はい

製品はインジケータを検索しません。\$(Root) パラメータが定数として定義されている場合は、ルート以下のファイルが管理されます。製品が検出できないか、または検出する必要がないサービスコンポーネントを定義し管理する場合は、[はい] を選択します。

いいえ

製品はインジケータを検索します。

デフォルト : いいえ

## [ディスカバリ方式] ページ: ファイル インジケータ フィールド

[ブループリント] ウィザードの [ディスカバリ方式] ページの [新規ファイルの追加] ペインには、以下のフィールドが含まれます。

### 名前 (posix)

検出するディレクトリまたはファイルを定義します。POSIX パターンマッチング構文を使用し、ワイルドカード (\*) を使用して名前を定義できます (たとえば、Agent\_\*)。

### タイプ

製品がブループリントに追加するタイプ (ファイルまたはディレクトリ) を指定します。

デフォルト: ファイル

注: [ルートからのパス] フィールドまたは [ルートからの階層数] の値を指定します。

### ルートからのパス

ファイルインジケータ パスを基準にしてコンポーネント ルートを探す部分的なパスを定義します。ファイルインジケータとコンポーネント ルートの間のディレクトリまたはフォルダのみを定義します (コンポーネント パス全体ではなく)。

たとえば、ファイルインジケータ mx.ini は、設定ファイル C:¥Program Files¥mx¥setup¥conf¥mx.ini を検索します。パス C:¥Program Files¥mx をコンポーネント ルートとして定義するには、ルートからのパスを setup¥conf として定義します。

- 検索では完全表現の一致をサポートしません。中間ディレクトリの変数名がわからない場合は、パスの一部にワイルドカードを使用します。例: \*¥conf
- ワイルドカード文字として \* または ? をディレクトリ名で使用できます。例: ¥\*conf、¥conf\*、/\*bin、/bin\*、/b\*n、/bi?

### ルートからの階層数

ディスカバリに含めるフォルダまたはディレクトリの階層数を定義します。カウントは、検索ルートを起点とします。

## [ディスカバリ方式] ページ: レジストリ インジケータの検索オプション フィールド

レジストリ インジケータの [ブループリント] ウィザードの [ディスカバリ方式] ページの [¥HKEY\_LOCAL\_MACHINE] ペインには、以下の検索オプションフィールドが含まれます。

### 検索開始場所

検索の開始地点を定義します。この値を設定しないと、検索はコンポーネントのファイルシステムの最上位から開始します。

Windows の場合、ファイルシステムの最上位 (/) にはすべての物理ドライブ (C:、D: など) が含まれます。

デフォルト: ¥HKEY\_LOCAL\_MACHINE

### 検索階層数

検索に含めるフォルダまたはディレクトリの階層数を定義します。

デフォルト: 10

### コンポーネント常時検出

プロセスが常にコンポーネントを検出済みと考慮するかどうかを指定します。

はい

製品はインジケータを検索しません。\$(Root) パラメータが定数として定義されている場合は、ルート以下のファイルが管理されます。製品が検出できないか、または検出する必要がないサービスコンポーネントを定義し管理する場合は、[はい] を選択します。

いいえ

製品はインジケータを検索します。

デフォルト: いいえ

## [ディスカバリ方式] ページ: レジストリ インジケータ フィールド

レジストリ インジケータの [ブループリント] ウィザードの [ディスカバリ方式] ページの [新規レジストリ値/キーの追加] ペインには、以下のレジストリ インジケータ フィールドが含まれます。

### 名前 (posix)

検出するディレクトリまたはファイルを定義します。 **POSIX** パターンマッチング構文を使用し、ワイルドカード (\*) を使用して名前を定義できます (たとえば、**Agent\_\***)。

### タイプ

製品がブループリントに追加するタイプ (ファイルまたはディレクトリ) を指定します。

**デフォルト:** ファイル

**注:** [ルートからのパス] フィールドまたは [ルートからの階層数] の値を指定します。

### ルートからのパス

検索ルートからのパスを定義します。

### ルートからの階層数

ディスカバリに含めるフォルダまたはディレクトリの階層数を定義します。カウントは、検索ルートを起点とします。

## [ディスカバリ方式] ページ: ネットワーク プローブ フィールド

ネットワーク プローブの [ブループリント] ウィザードの [ディスカバリ方式] ページの [新規ネットワーク プローブの追加] ペインには、以下のネットワーク プローブ フィールドが含まれます。

### 名前

ネットワーク プローブの名前を定義します。

### 説明

ブループリント項目を説明します。

### プライマリ ポート

応答を収集する最初のポート番号のカンマ区切りリストを定義します。

#### 代替ポート

プライマリ ポートが失敗した場合に応答を収集する、カンマ区切りのポート番号リストを定義します。

#### プローブ

ポート通信が確立された後、CCA からリモート サービスに送信されるオプションのプローブを説明します。プローブの目的は、サービスまたはコンポーネントの存在の確認です。

#### 正規表現一致

正規表現のディレクティブ値と比較する属性を定義します。値が一致しない場合、ディレクティブは失敗します。

#### バージョン正規表現

正規表現のバージョンを定義します。

#### プロトコル

プローブが TCP (Transmission Control Protocol) または UDP (User Datagram Protocol) を使用するかどうかを指定します。

デフォルト : TCP

## [ディスカバリ検証ルール]ページ: ディスカバリ検証ルール フィールド

[ブループリント] ウィザードの [ディスカバリ検証ルール] ページの [新規ディスカバリ検証ルールの追加] ペインには、以下のディスカバリ検証ルール フィールドが含まれます。

#### 名前

ディスカバリ検証ルールの名前を定義します。

#### 説明

ブループリント項目を説明します。



### ディレクティブ タイプ

製品が次の目的で使用する、ディレクティブ タイプを定義します。

- サービスで管理されるエレメントから値を抽出する
- 管理対象サーバから値を取得する

[ディレクティブ タイプ] ドロップダウン リストには以下のオプションが含まれます。

### 定数

変数の置換または複雑な文字列の作成に使用される固定値を定義します。このディレクティブ タイプの一般的な用途は以下のとおりです。

- コンポーネントのインストール場所が明確な場合に、固定の **Root** パラメータおよび **RegistryRoot** パラメータを定義する。
- 変数と固定テキストを複雑な文字列に組み立てることにより、複数のソースからデータを組み合わせる。
- コンポーネントの定数（ベンダー名など）をパラメータとして提供する。

### データベース クエリ

管理されたサーバ上のデータベースを問い合わせ、（必要に応じて）取得されたデータから値を抽出します。このディレクティブ タイプの一般的な用途は以下のとおりです。

- データベース行から列値を抽出するクエリを実行する。
- 管理対象サーバの変更または値の抽出を行うストアードプロシージャを実行する。
- 結果セットを抽出してマクロ ステップ ディレクティブ に表形式で表示するクエリを実行する。

### ファイルの取得

特定のファイル（ファイルの場所が既知の場合）の内容を取得します。取得後、製品は、ディレクティブの値を定義するために内容をフィルタします。このディレクティブ タイプの一般的な用途は以下のとおりです。

- ファイルの内容を取得して正規表現でフィルタすることにより、非構造化ファイルからパラメータを抽出する。
- ログ ファイルを取得して、マクロ ステップ ディレクティブ で表示、格納、およびインポート/エクスポートする。

#### SNMP の取得

管理情報ベース（MIB）アドレスにある値を SNMP エージェントから取得し、その結果からディレクティブ値を（必要に応じて）設定します。

#### ファイル名一致

管理対象サーバのファイル システム内の指定された場所にあるファイルおよびディレクトリをすべてリスト表示し、そのリストから（必要に応じて）ディレクティブ値を抽出します。

#### レジストリ名一致

（Windows のみ）レジストリ ツリー内の一定の場所にあるレジストリ キーおよびレジストリ値名をリスト表示し、リストから（必要に応じて）ディレクティブ値を抽出します。

#### レジストリ名一致およびデータ取得

（Windows のみ）レジストリ ツリー内の一定の場所にあるレジストリ キーおよびレジストリ値名をすべてリスト表示します。

（オプション）選択した名前のデータ値をディレクティブ値として取得します。

#### ネットワークプローブ

リモート サービス サーバおよびポートに対して開かれている TCP ソケットのパラメータを定義します。（必要に応じて）ポートにプローブを送信し、ソケットからの応答をディレクティブ値として収集できます。

#### レジストリ

（Windows のみ）レジストリ キーまたはレジストリ値に関連付けられたデータ値を取得し、（必要に応じて）フィルタされた結果からディレクティブ値を定義します。

## リモート実行

管理対象サーバでコマンドまたはスクリプトを実行します。コマンド出力がキャプチャされ、ディレクティブ値として返されます。冗長なコマンド出力から簡潔な値を抽出するために、通常は正規表現によって値がフィルタされます。このディレクティブタイプの一般的な用途は以下のとおりです。

- 出力からのみ利用できる構成情報にアクセスする（例：オペレーティングシステムの構成）。
- 一時的なデータ（メモリ、ネットワーク、CPU の統計など）にアクセスする。
- カスタム スクリプトおよびカスタム ツールを利用して、出力をブループリントにインポートする。
- 管理対象サーバを更新するためにユーティリティおよびスクリプトを実行する。

## Web サービスコール

Web サービスとして公開されるアプリケーション構成データを取得します。ディレクティブは Web サービスのクエリを行い、返されたデータを構成変数に解析します。ディレクティブは、構成データを取得し格納できる構成実行可能ファイルをサポートします。

Web サービスが実行されている WSDL URL を提供します。Web サービス記述言語（WSDL）は、Web サービスを説明するためのモデルを提供する、XML ベースの言語です。

デフォルト：定数

## ステージ

コンポーネントパラメータディレクティブが実行されることを基準にして、ディスカバリ中に検証ディレクティブが実行されることを指定します。

## 後期

検証ディレクティブは、変数置換のためにコンポーネントパラメータの値を必要とします。コンポーネントパラメータ値とは、[コンポーネントパラメータおよび変数] タブで定義されているパラメータ名のことです。コンポーネントパラメータ値は、置換された変数が検証に必要な値を持っていることを保証します。

## 初期

検証ディレクティブはコンポーネントパラメータの値に依存しません。

デフォルト：後期

## 値

固定文字列または 1 つ以上の変数置換を含む文字列を定義します。

- ディレクティブが 0 より大きい長さの文字列に解決される場合、その文字列はディレクティブ値になります。
- 文字列の長さがゼロまたは未定義の場合、ディレクティブは値なしと見なされ、指定されたデフォルト値が代わりに使用されます。

定数値をブランク（1 つ以上のスペース）として定義できます。ブランクの値はユーザ インターフェースに値として表示されませんが、製品はそれを有効な値と considers。

例：

- 固定デフォルトでの変数置換

`$(VariableName)`

- 固定テキスト内での複数置換

`C:¥$(V)¥$(V2)¥file.xml`

以下の変数構文も使用できます。

`$(parameter_name)`

通常のコМПОНЕНТ変数置換を定義します。

`$(#parameter_name)`

親COMPONENTのパラメータを参照します。

`$(@global_variable_name)`

グローバル変数を参照します。

## 正規表現

文字列のセットを表す正規表現を定義します。定数値を定義するために変数置換を使用する場合は、正規表現を使用して値をフィルタできます。また、正規表現を条件式のように使用して値をテストし、次のいずれかの方法で返すこともできます。

- 値が表現に一致する場合は `paren(0)` を返します。
- 値と表現が一致しない場合、返される値はありません。

## パーレン

正規表現がかっこで囲まれた部分表現を含む場合、一致したときにどの部分表現が返されるのかを定義します。

デフォルト：0（[パーレン] 値を指定しない場合）

#### 正規表現一致

正規表現のディレクティブ値と比較する属性を定義します。値が一致しない場合、ディレクティブは失敗します。

#### 常に以下と等しい

ディレクティブ値と比較する属性を定義します。値が等しくない場合、ディレクティブは失敗します。

#### 大文字と小文字の区別

比較するときに大文字と小文字を区別するかどうかを指定します。

**デフォルト:** 大文字と小文字を区別する

**注:** デフォルト（[大文字と小文字を区別する]）オプションでは、`Agent_conf` と `agent_conf` は一致すると見なされません。[大文字と小文字を区別しない] オプションでは、`Agent_conf` と `agent_conf` は一致と見なされます。

## トランスレート

トランスレート処理がディレクティブの出力に適用する一意の名前を定義します。実行時に、指定された [トランスレート] 値の前に `$CCTranslation$__` を付加して、データベース内で値をトランスレート処理として識別できるようにします。トランスレート処理が一致しない場合は、元のディレクティブ結果が保持されます。

構成ファイルおよびレジストリから抽出されるディレクティブ値は、列挙型に属する暗号の文字列または整数であることがあります。各値は設定状態に対応しますが、値は状態を解釈しません。これらの値を意味のある文字列に変換して、ブループリントに表示する際にユーザにとって理解しやすいものにすることができます。

製品データベース内の値をマップして、トランスレート処理名を参照し、*変換前の値から変換後の値*へのトランスレート処理をトリガします。以下の例では、トランスレート処理 `$CCTranslation$__IIS SERVER STATE` がマップされます。

```
<BlueprintTranslation name="IIS server state"
coh_name="IIS_SERVER_STATE" coh_id="23501"
created_by="system_user">
  <BlueprintTranslationEntry translate_from="2"
    translate_to="Running" />
  <BlueprintTranslationEntry translate_from="4"
    translate_to="Stopped" />
  <BlueprintTranslationEntry translate_from="6"
    translate_to="Paused" />
</BlueprintTranslation>
```

トランスレート処理は、特定のブループリントとは関連付けられません。他のトランスレート処理との競合を回避するには、各トランスレート処理が一意の名前を持つようにします。

**注:** 製品のユーザインターフェースは、現在トランスレート処理テーブルの定義をサポートしていません。CA Configuration Automation データロード形式で (例に示すように) ファイルを作成し、データローダユーティリティを使用してそれらのファイルをデータベースにロードします。

### 変換

返された XML 形式のディレクティブ値をフィルタするために使用される XSL 変換を定義します。変換された値は元の戻り値を置換し、新しい XML 値または変換によって生成されたテキストのいずれかになります。

XSL 変換を実行するために、ディレクティブ値に適用される XSL ファイルの場所を指定します。変換の場所は、サーバから認識できる URL (file、http など)、またはサーバの CLASSPATH 内のファイル名になります。変換が取得され、ディレクティブの新しい値を生成するためにディレクティブ値に適用されます。

### 挿入

表示または後続の操作に見合った結果を生成するために、別の文字列内に挿入されるディレクティブの値を定義します。

挿入場所には \$(VALUE) (すべて大文字) を含む任意の文字列を指定できます。ディレクティブ値は \$(VALUE) を置換して、フィルタされた結果を生成します。たとえば、最初のディレクティブ値が 3 で、挿入文字列が 1.\$(VALUE).export である場合、フィルタされたディレクティブ値は 1.3.export です。

### 修飾子

検出結果を修飾します。[ホストはエージェントを使用] 修飾子または [Alterpoint デバイス] 修飾子を選択します。

### 修飾子パラメータ

製品が指定した修飾子に適用するパラメータを定義します。



## [管理] ページ: ファイル管理オプション フィールド

[ブループリント] ウィザードの [管理] ページの [\$(Root)] ペインには以下のファイル管理オプション フィールドが含まれます。

### 管理対象階層数

ディスカバリ操作によって確認される、ファイル システム ルート以下のディレクトリのレベル数を定義します。

### 設定した最大数までファイルを取得 (50,000)

ディスカバリ操作が取得するファイルの数を指定します。

オン: ディスカバリ操作は最大 50,000 ファイルを取得します。

オフ: ディスカバリ操作は、[最大ファイル数] フィールドで指定した数のファイルまで取得します。

### 最大ファイル数

ディスカバリ操作によって取得される最大ファイル数を定義します。

## [管理] ページ: ディレクトリ フィールド

[ブループリント] ウィザードの [管理] ページの [新規ディレクトリの追加] ペインには、以下のディレクトリ フィールドが含まれます。

### 名前 (posix)

追加するエレメントの名前を定義します。POSIX パターン マッチング構文を使用し、ワイルドカード (\*) を使用して名前を定義できます (たとえば、Agent\_\*)。

### ルートからのパス

エレメントを配置するパスの一部を定義します。ファイル インジケータとコンポーネント ルートの間のディレクトリまたはフォルダのみを定義します (コンポーネント パス全体ではなく)。

## [管理] ページ: ディレクトリ フィールド

[ブループリント] ウィザードの [管理] ページの [新規ファイルの追加] ペインには、以下のファイル フィールドが含まれます。

### 名前 (posix)

追加するエレメントの名前を定義します。POSIX パターンマッチング構文を使用し、ワイルドカード (\*) を使用して名前を定義できます (たとえば、Agent\_\*)。

### ルートからのパス

エレメントを配置するパスの一部を定義します。ファイルインジケータとコンポーネント ルートの間のディレクトリまたはフォルダのみを定義します (コンポーネント パス全体ではなく)。

## [ファイル フィルタおよび属性] ページの [ルール] タブのフィールド

[コンポーネント ブループリント] ウィザードの [ファイル フィルタおよび属性] ページの [ルール] タブには、以下のフィールドが含まれます。

### 名前

ルールの名前を定義します。

### 説明

ブループリント項目を説明します。

### ドキュメント URL

ルールのドキュメントが存在する URL を指定します。

### 制約タイプ

ルールが適用されるファイルまたはディレクトリの制約タイプを指定します。

#### ファイル

- ファイル変更日付
- ファイル所有者
- ファイルアクセス権
- ファイルサイズ
- ファイルバージョン
- 製品バージョン

#### ディレクトリ

- バイト数
- 階層数
- ディレクトリ変更日付
- ディレクトリが存在する必要がある
- ディレクトリ所有者
- ディレクトリ アクセス権
- ファイルが存在している必要がある
- ディレクトリ数
- ファイル数

### 操作

選択した [制約タイプ] に対してルールで使用される操作を指定します。

- =(等しい)
- != (等しくない)
- = (大文字/小文字を区別しない)
- != (大文字/小文字を区別する)
- > (より大きい)
- >= (次の値以上)
- < (より小さい)
- <= (次の値以下)
- 整数範囲 (境界値を含む)
- 常に一致する (正規表現)
- 常に一致する (正規表現、大文字/小文字を区別しない)
- 常に一致しない (正規表現)

### 値

操作に対する参照として取得される値を指定します。制約タイプ、操作、値はルール的一致条件を定義します。

### 無効化

ルールが有効 (オン) か無効 (オフ) かを指定します。

### 失敗時

このブループリントを使用したコンプライアンス操作がルールに違反した場合にルール コンプライアンス結果に書き込まれる文字列を定義します。

### 重大度

ルールの失敗を決定するエラー レベルを指定します。

- 情報
- 警告
- エラー
- クリティカル

## レジストリ管理フィールド

［ブループリント］ウィザードの［管理］ページの［\$(RegistryRoot)］ペインには以下のレジストリ管理オプションフィールドが含まれます。

### 管理対象階層数

ディスカバリ操作によって確認される、ファイルシステム ルート以下のディレクトリのレベル数を定義します。

### 設定した最大数までエレメントを取得 (50,000)

ディスカバリ操作が取得する要素の数を指定します。

**選択済み：** ディスカバリ操作は最大 50,000 エレメントを取得します。

**オフ：** ディスカバリ操作は、［最大エレメント数］フィールドで指定した数のエレメントまで取得します。

### 最大要素数

ディスカバリ操作によって取得される最大要素数を定義します。

## [レジストリ フィルタおよび属性] ページのキーの追加フィールド

[コンポーネント ブループリント] ウィザードの [レジストリ フィルタおよび属性] ページには以下のキーの追加フィールドが含まれます。

### デフォルト

デフォルト レジストリ キー値を定義します。

### 解釈方法

キーが、以下の項目のうちどのエンティティとして解釈されるのかを指定します。

- データベース名
- データベース テーブル
- ホスト名およびポート
- ホスト名または IP アドレス
- JDBC URL
- TCP ポート番号
- URL
- Web サービス URL

### 関係キー

キーが関係キーかどうかを指定します。

## 関係タイプ

以下の関係タイプのいずれかを指定します。

### 通信する

アプリケーションとデータベース サーバの関係を確立します。

### 管理する

別のサーバを管理するサーバ間の関係を確立します。たとえば、VMware ESX ホストを管理する VMware vCenter Server 間の関係を表示するには [管理] を選択します。

### ホストする

別のサーバをホストするサーバ間の関係を確立します。たとえば、VMware 仮想マシンをホストする VMware ESX ホスト間の関係を表示するには [ホスト] を選択します。

### 使用する

サーバ間の関係を確立します。

## 表示/非表示

検出されたレジストリ キーを表示するかどうかを指定します。また、ブループリント UI で定義されたデフォルト値を表示するかどうかを指定します。[表示/非表示] 値を [値を非表示] として選択すると、値のマスキ化と暗号化が行われます。

### 値を表示

デフォルト値と、検出されたレジストリ キーを表示します。

### 値を非表示

ブループリント UI のデフォルト値と、検出されたレジストリ キーを非表示にします。値の代わりに \*\*\*\*\* を表示します。以下のケースでは、値を非表示にしてください。

- 値が機密である（パスワードなど）。
- 値がバイナリである。
- 表示するには、値が長すぎる。

非表示のレジストリ キー値は、データベース内で暗号化されます。UI では表示されないか、または参照できません。

**注:** 後で値を表示することを決定した場合は、このフィールドのセキュリティを保証するために、値を再入力します。

#### 解釈済みのサーバ

他のコンポーネント パラメータで利用可能なターゲット サーバの情報を定義します。パラメータを定義するには、変数置換を使用します。

#### 解釈済みのターゲット インスタンス

他のコンポーネント パラメータで利用可能なターゲット データベース インスタンスまたはアプリケーション インスタンスを定義します。パラメータを定義するには、変数置換を使用します。

#### 解釈済みのアプリケーション

他のパラメータで利用可能なターゲット アプリケーションの情報を指定します。パラメータの定義には、変数置換を使用します。

## [レジストリ フィルタおよび属性] ページの値詳細フィールド

[コンポーネント ブループリント] ウィザードの [レジストリ フィルタおよび属性] ページには以下の値詳細フィールドが含まれます。

#### デフォルト

デフォルト レジストリ キー値を定義します。

#### 解釈方法

キーが、以下の項目のうちどのエンティティとして解釈されるのかを指定します。

- データベース名
- データベース テーブル
- ホスト名およびポート
- ホスト名または IP アドレス
- JDBC URL
- TCP ポート番号
- URL
- Web サービス URL

#### 関係キー

キーが関係キーかどうかを指定します。



### 関係タイプ

以下の関係タイプのいずれかを指定します。

#### 通信する

アプリケーションとデータベース サーバの関係を確立します。

#### 管理する

別のサーバを管理するサーバ間の関係を確立します。たとえば、VMware ESX ホストを管理する VMware vCenter Server 間の関係を表示するには [管理] を選択します。

#### ホストする

別のサーバをホストするサーバ間の関係を確立します。たとえば、VMware 仮想マシンをホストする VMware ESX ホスト間の関係を表示するには [ホスト] を選択します。

#### 使用する

サーバ間の関係を確立します。

### 表示/非表示

検出されたレジストリ値を表示するかどうかを指定します。また、ブループリント UI で定義されたデフォルト値を表示するかどうかを指定します。[表示/非表示] 値を [値を非表示] として選択すると、値のマスキ化と暗号化が行われます。

#### 値を表示

デフォルト値と、検出されたレジストリ値を表示します。

#### 値を非表示

ブループリント UI のデフォルト値と、検出されたレジストリ値を非表示にします。値の代わりに \*\*\*\*\* を表示します。以下のケースでは、値を非表示にしてください。

- 値が機密である（パスワードなど）。
- 値がバイナリである。
- 表示するには、値が長すぎる。

非表示のレジストリ キー値は、データベース内で暗号化されます。UI では表示されないか、または参照できません。

**注:** 後で値を表示することを決定した場合は、このフィールドのセキュリティを保証するために、値を再入力します。

#### 解釈済みのサーバ

他のコンポーネント パラメータで利用可能なターゲット サーバの情報を定義します。パラメータを定義するには、変数置換を使用します。

#### 解釈済みのターゲット インスタンス

他のコンポーネント パラメータで利用可能なターゲット データベース インスタンスまたはアプリケーション インスタンスを定義します。パラメータを定義するには、変数置換を使用します。

#### 解釈済みのアプリケーション

他のパラメータで利用可能なターゲット アプリケーションの情報を指定します。パラメータの定義には、変数置換を使用します。

#### 解釈されたクラスタ

他のコンポーネント パラメータで利用可能なターゲット クラスタ名を定義します。パラメータを定義するには、変数置換を使用します。

## [データベース] ページのフィールド

[コンポーネント ブループリント] ウィザードの [データベース] ページには以下のフィールドが含まれます。

#### 名前

データベースの名前を定義します。

#### 説明

ブループリント項目を説明します。

#### データベース

データベース アクセスの説明を定義します。

#### ユーザ

データベースにアクセスできるユーザのアカウント名を定義します。

#### パスワード

指定したユーザ アカウントに関連付けるパスワードを定義します。

#### サーバ

データベースがインストールされているサーバを定義します。

## ポート

データベース ホスト サーバのリスニング ポートを定義します。

### DB タイプ

以下のオプションから、使用中のデータベースのタイプを指定します。

- DEFAULT\_CONTEXT
- SQL\_SERVER\_CONTEXT
- ORACLE\_CONTEXT
- INFORMIX\_CONTEXT
- DB2\_CONTEXT
- MYSQL\_CONTEXT
- POSTGRES\_CONTEXT
- HSQldb\_CONTEXT
- ORACLE9\_CONTEXT
- ODBC
- CLOUDSCAPE
- ORACLE10\_CONTEXT
- SYBASE11\_CONTEXT
- INGRES\_CONTEXT
- SQL\_SERVER2K5\_CONTEXT
- JAVA\_DB\_CONTEXT
- SYBASE15\_CONTEXT
- ORACLE11\_CONTEXT
- SQL\_SERVER2K8\_CONTEXT

### 環境

エージェントが、コマンドを実行する前に設定する環境変数を定義します。

製品は CA Configuration Automation エージェント環境でリモート コマンドを実行します。コマンド実行に要求される環境がエージェントに設定されていない場合、必要な環境変数を名前/値ペアのカンマ区切りのリストとして定義します。形式は、名前=値です。例：

```
<var_name>=<value>,<var_name2>=<value2>
```

エージェントは変数置換方式をサポートします。

## [コンポーネント パラメータおよび変数] ページのフィールド

[コンポーネント ブループリント] ウィザードの [コンポーネント パラメータおよび変数] ページには以下のフィールドが含まれます。

### 名前

パラメータ代入表現を以下の形式で定義します。

`$(VariableName)`

*VariableName*

コンポーネントで定義されるディスカバリ パラメータの名前を定義します。デフォルトでは、[名前] 値は大文字と小文字が区別され、パラメータ名と完全に一致する必要があります。

- 文字列リテラルにパラメータ表現を埋め込むか、単独で使用できます。
- 再帰的な複数の置換を同じ式で定義できます。たとえば、代入値はパラメータ表現の文字列とすることができます。その場合、ブループリントは代入値を再帰的に評価します。

### 例:

ディスカバリ パラメータに以下の値が指定されているとします。

```
User=info
Domain=ca.com
v1=$(User)
v2=$(Domain)
```

以下のパラメータ代入表現の場合

```
$(User)@$(Domain) [$(v1) at $(v2)]
```

評価後に以下の結果が返されます。

```
info@ca.com [info at ca.com]
```

### 説明

ブループリント項目を説明します。

### フォルダ

項目の場所を定義します。

### 選択されたカテゴリ

エレメントを構成するために製品が使用するカテゴリ（以下のカテゴリのうちのいずれか）を定義します。[利用可能なカテゴリ] 列の項目をダブルクリックすると、その項目は[選択されたカテゴリ] 列に移動します。

### 環境管理

一般的なコンポーネント可用性および環境管理を基準にして、管理設定を定義します。管理設定には、以下のような例があります。

- エレメントのバックアップ方法
- キャッシュの削除時期
- 操作の再試行回数

### 設定

[環境管理]、[ログおよびデバッグ]、[ネットワーク]、または[パフォーマンス] 設定に関連していない、コンポーネント構成設定を定義します。構成の設定には、コンポーネントのエイリアスやデフォルト **Web** ページなどの例があります。

### ドキュメント

コンポーネントの動作を文書化するエレメント、またはユーザーに情報を提供するエレメントを定義します。たとえば、ガイド、readme ファイル、FAQ、オンライン ヘルプ ページなどです。

### ログおよびデバッグ

ユーザーによる、ログの場所、ログ レベル、デバッグ出力、診断変数タイプの設定を可能にするエレメントを定義します。

### ネットワーク

ネットワーク関連コンポーネント設定（たとえば **SNMP** ポート設定）を表すエレメントを定義します。[ネットワーク] と [セキュリティ] の両方に分類できるエレメント（たとえば **LDAP** 認証の有効化など）の場合、カテゴリを [セキュリティ] に設定します。

### その他

CA 内部でのみ使用されます。

### パフォーマンス

通常は構成パラメータの特定サブセットになっているパフォーマンス設定を定義します。たとえば、スレッドの数または同時ユーザ数などです。

### 製品情報

一般的（静的）な製品情報（ライセンス、インストール場所、ベンダー、モジュール名など）を定義します。

### リソース

コンポーネントのリソース（ストレージ、メモリおよびキャッシュの割り当てまたはサイズ、CPU など）を定義します。

**注:** 静的な [リソース] カテゴリは、リアルタイム情報を定義する [一時的] カテゴリとは異なることに注意してください。

### セキュリティ

通常は構成パラメータの特定サブセットになっている、セキュリティ関連設定を表すエレメントを定義します。たとえば、認証タイプ、認証の有効化、暗号化設定、ディレクトリ参照、SSL、または HTTPS です。

### 一時的

定期的に変化するエレメントを定義します。たとえば、サーバ状態（起動、ダウン、稼働中、停止）、現在接続されているクライアント数、現在のスレッド数、現在のディスク使用率などです。

### バージョンおよびパッチ

製品のバージョンまたはパッチのレベルを示すエレメントを定義します。

### 選択されたフィルタ

製品が比較操作からフィルタするエレメント（以下のエレメントのいずれか）を定義します。[利用可能なフィルタ] 列の項目をダブルクリックすると、その項目は[選択されたフィルタ] 列に移動します。

### コンポーネント固有

単一のコンポーネント インスタンスに固有のエレメント（インストール ルート、サービス サーバ名など）をフィルタします。すでに変更検出操作および結果とは異なることが認識されている、コンポーネント固有のエレメントを除外します。

### サービス固有

サービスに固有のエレメント（サーバ名、インストール ルートなど）をフィルタします。すでに変更検出操作および結果とは異なることが認識されている、サービス固有のエレメントを除外します。

### サーバ固有

単一のサーバに固有のエレメント（サーバ名、IP アドレスなど）をフィルタします。すでに変更検出操作および結果とは異なることが認識されている、サーバ固有のエレメントを除外します。

### 変更検出を実行しない

変更検出の操作および結果から永久に除外するエレメントを指定します。例として、一時ディレクトリ、管理対象フォルダ内のログ ファイル、一時的であることが認識されているエレメントなどがあります。

### ルール コンプライアンスを実行しない

連続しない、または可変であるため、ルール コンプライアンス操作と結果から完全に除外するエレメントを指定します。例として、一時ディレクトリ、既知の古い構成ファイル、テンプレート、サンプル ファイルなどがあります。

### 時刻変化

時間が経過すると変更されることがわかっているが、サーバ全体またはサービス全体に必要なではないエレメントをフィルタします。例として、ログ ファイル、プロセスの開始時刻、レジストリ イベント カウンタなどがあります。すでに変更検出操作および結果とは異なることが認識されている、時変パラメータを除外します。



### ファイル サイズ変化

[変更検出] 操作でファイル サイズに基づいてファイルをフィルタするのは、[時刻変化] フィルタが選択されていない場合のみです。

### ウェイト

エレメントの相対的な重要度を指定します。

- 低
- 中
- 高

デフォルト：中（ウェイトを割り当てていないエレメント）

### ディレクティブ タイプ

製品が次の目的で使用する、ディレクティブ タイプを定義します。

- サービスで管理されるエレメントから値を抽出する
- 管理対象サーバから値を取得する

[ディレクティブ タイプ] ドロップダウン リストには以下のオプションが含まれます。

### 定数

変数の置換または複雑な文字列の作成に使用される固定値を定義します。このディレクティブ タイプの一般的な用途は以下のとおりです。

- コンポーネントのインストール場所が明確な場合に、固定の **Root** パラメータおよび **RegistryRoot** パラメータを定義する。
- 変数と固定テキストを複雑な文字列に組み立てることにより、複数のソースからデータを組み合わせる。
- コンポーネントの定数（ベンダー名など）をパラメータとして提供する。

### 設定

解析済みの構成ファイルまたは構成プログラムファイルから構成パラメータの値を取得します。

### データベース クエリ

管理されたサーバ上のデータベースを問い合わせ、（必要に応じて）取得されたデータから値を抽出します。このディレクティブタイプの一般的な用途は以下のとおりです。

- データベース行から列値を抽出するクエリを実行する。
- 管理対象サーバの変更または値の抽出を行うストアドプロシージャを実行する。
- 結果セットを抽出してマクロ ステップ ディレクティブに表形式で表示するクエリを実行する。

### ファイルの取得

特定のファイル（ファイルの場所が既知の場合）の内容を取得します。取得後、製品は、ディレクティブの値を定義するために内容をフィルタします。このディレクティブタイプの一般的な用途は以下のとおりです。

- ファイルの内容を取得して正規表現でフィルタすることにより、非構造化ファイルからパラメータを抽出する。
- ログ ファイルを取得して、マクロ ステップ ディレクティブで表示、格納、およびインポート/エクスポートする。

### LDAP の取得

ディレクトリ サーバから名前付きデータ セットを取得し、（必要に応じて）出力からディレクティブ値を抽出します。

### SNMP の取得

管理情報ベース（MIB）アドレスにある値を SNMP エージェントから取得し、その結果からディレクティブ値を（必要に応じて）設定します。

### ファイル名一致

管理対象サーバのファイル システム内の指定された場所にあるファイルおよびディレクトリをすべてリスト表示し、そのリストから（必要に応じて）ディレクティブ値を抽出します。

### レジストリ名一致

（Windows のみ）レジストリ ツリー内の一定の場所にあるレジストリ キーおよびレジストリ値名をリスト表示し、リストから（必要に応じて）ディレクティブ値を抽出します。

### レジストリ名一致およびデータ取得

(Windows のみ) レジストリ ツリー内の一定の場所にあるレジストリ キーおよびレジストリ 値名をすべてリスト表示します。選択された名前のデータ値から、ディレクティブ値を (必要に応じて) 設定できます。

### ネットワークプローブ

リモート サービス サーバおよびポートに対して開かれている TCP ソケットのパラメータを定義します。(必要に応じて) ポートにプローブを送信し、ソケットからの応答をディレクティブ値として収集できます。

### レジストリ

(Windows のみ) レジストリ キーまたはレジストリ 値に関連付けられたデータ値を取得し、(必要に応じて) フィルタされた結果からディレクティブ値を定義します。

### リモート実行

管理対象サーバでコマンドまたはスクリプトを実行します。コマンド出力がキャプチャされ、ディレクティブ値として返されます。冗長なコマンド出力から簡潔な値を抽出するために、通常は正規表現によって値がフィルタされます。このディレクティブタイプの一般的な用途は以下のとおりです。

- 出力からのみ利用できる構成情報にアクセスする (例: オペレーティング システムの構成)。
- 一時的なデータ (メモリ、ネットワーク、CPU の統計など) にアクセスする。
- カスタム スクリプトおよびカスタム ツールを利用して、出力をブループリントにインポートする。

管理対象サーバを更新するためにユーティリティおよびスクリプトを実行する。

### デフォルト

固定文字列または 1 つ以上の変数置換を含む文字列を定義します。ディレクティブが実行されたが、値を判定できない場合、製品は未定義値または長さゼロの値の代わりにデフォルト値を使用します。

### 保持

サービスの実行中にパラメータが持続するかどうかを指定します。

#### 常に保持

パラメータは保存され、サービス ツリー ビューに表示されます。

#### 保持しない

パラメータは、ディスカバリのためにのみ使用され、保存されることはありません。

#### NULL ではない場合は保持

パラメータは保存され、(値を備えている場合は) サービス ツリー ビューに表示されます。

デフォルト: [常に保持]

## 表示/非表示

コンポーネントでコンポーネント パラメータとその値を表示するかどうかを指定します。また、ブループリント UI で定義されたデフォルト値を表示するかどうかを指定します。値を非表示にする場合、値は暗号化されます。

### 値を表示

デフォルト値と、コンポーネント パラメータおよびその値を表示します。

### 値を非表示

ブループリント UI のデフォルト値と、コンポーネント ビューア UI のコンポーネント パラメータ値を非表示にします。対応する値の代わりに **\*\*\*\*\*** を表示します。以下の場合にはコンポーネント パラメータ値とデフォルト値を非表示にします。

- 値が機密である（パスワードなど）。
- 値がバイナリである。
- 表示するには、値が長すぎる。

非表示のコンポーネント パラメータ値は、データベース内で暗号化されます。UI では表示されないか、または参照できません。

注: 後で値を表示することを決定した場合は、このフィールドのセキュリティを保証するために、値を再入力します。

### エレメントを非表示

パラメータを表示しません。値を変数置換では使用できるがサービス ツリー ビューには表示されないようにする場合は、[エレメントを非表示] を選択します。

### デフォルト: 値を表示

## 大文字と小文字の区別

比較するときに大文字と小文字を区別するかどうかを指定します。

### デフォルト: 大文字と小文字を区別する

注: デフォルト ([大文字と小文字を区別する]) オプションでは、**Agent\_conf** と **agent\_conf** は一致すると見なされません。[大文字と小文字を区別しない] オプションでは、**Agent\_conf** と **agent\_conf** は一致と見なされます。

### 解釈方法

構成パラメータの文字列形式に関するヒント、および関連するコンポーネントによるその使用方法を定義します。解釈されたパラメータ値は、コンテキスト依存パーサを使用して検査されます。製品は、パーサを使用して、複雑なパラメータ文字列から複数のサブ値を抽出できます。

**デフォルト：** 空（解釈はありません）。

### 関係キー

製品が [解釈方法] の値に従って関係を決定し割り当てるかどうかを指定します。

関係を確立するには、[解釈方法] の値を設定して、[関係キー] フィールドを [はい] に設定します。一部の解釈は関係を定義できません。

**デフォルト：** いいえ

### 関係タイプ

以下の関係タイプのいずれかを指定します。

#### 通信する

アプリケーションとデータベース サーバの関係を確立します。

#### 管理する

別のサーバを管理するサーバ間の関係を確立します。たとえば、VMware ESX ホストを管理する VMware vCenter Server 間の関係を表示するには [管理] を選択します。

#### ホストする

別のサーバをホストするサーバ間の関係を確立します。たとえば、VMware 仮想マシンをホストする VMware ESX ホスト間の関係を表示するには [ホスト] を選択します。

#### 使用する

サーバ間の関係を確立します。

### 解釈済みのサーバ

他のコンポーネント パラメータで利用可能なターゲット サーバの情報を定義します。パラメータを定義するには、変数置換を使用します。

#### 解釈済みのターゲット インスタンス

他のコンポーネント パラメータで利用可能なターゲット データベース インスタンスまたはアプリケーション インスタンスを定義します。パラメータを定義するには、変数置換を使用します。

#### 解釈済みのアプリケーション

他のパラメータで利用可能なターゲット アプリケーションの情報を指定します。パラメータの定義には、変数置換を使用します。

## トランスレート

トランスレート処理がディレクティブの出力に適用する一意の名前を定義します。実行時に、指定された [トランスレート] 値の前に `$CCTranslation$__` を付加して、データベース内で値をトランスレート処理として識別できるようにします。トランスレート処理が一致しない場合は、元のディレクティブ結果が保持されます。

構成ファイルおよびレジストリから抽出されるディレクティブ値は、列挙型に属する暗号の文字列または整数であることがあります。各値は設定状態に対応しますが、値は状態を解釈しません。これらの値を意味のある文字列に変換して、ブループリントに表示する際にユーザにとって理解しやすいものにすることができます。

製品データベース内の値をマップして、トランスレート処理名を参照し、*変換前の値から変換後の値*へのトランスレート処理をトリガします。以下の例では、トランスレート処理 `$CCTranslation$__IIS SERVER STATE` がマップされます。

```
<BlueprintTranslation name="IIS server state"
coh_name="IIS_SERVER_STATE" coh_id="23501"
created_by="system_user">
  <BlueprintTranslationEntry translate_from="2"
    translate_to="Running" />
  <BlueprintTranslationEntry translate_from="4"
    translate_to="Stopped" />
  <BlueprintTranslationEntry translate_from="6"
    translate_to="Paused" />
</BlueprintTranslation>
```

トランスレート処理は、特定のブループリントとは関連付けられません。他のトランスレート処理との競合を回避するには、各トランスレート処理が一意の名前を持つようにします。

**注:** 製品のユーザインターフェースは、現在トランスレート処理テーブルの定義をサポートしていません。CA Configuration Automation データロード形式で (例に示すように) ファイルを作成し、データローダユーティリティを使用してそれらのファイルをデータベースにロードします。



## 変換

返された XML 形式のディレクティブ値をフィルタするために使用される XSL 変換を定義します。変換された値は元の戻り値を置換し、新しい XML 値または変換によって生成されたテキストのいずれかになります。

XSL 変換を実行するために、ディレクティブ値に適用される XSL ファイルの場所を指定します。変換の場所は、サーバから認識できる URL (file、http など)、またはサーバの CLASSPATH 内のファイル名になります。変換が取得され、ディレクティブの新しい値を生成するためにディレクティブ値に適用されます。

## 挿入

表示または後続の操作に見合った結果を生成するために、別の文字列内に挿入されるディレクティブの値を定義します。

挿入場所には \$(VALUE) (すべて大文字) を含む任意の文字列を指定できます。ディレクティブ値は \$(VALUE) を置換して、フィルタされた結果を生成します。たとえば、最初のディレクティブ値が 3 で、挿入文字列が 1.\$(VALUE).export である場合、フィルタされたディレクティブ値は 1.3.export です。

## 修飾子

検出結果を修飾します。[ホストはエージェントを使用] 修飾子または [Alterpoint デバイス] 修飾子を選択します。

## 修飾子パラメータ

製品が指定した修飾子に適用するパラメータを定義します。

## 解釈されたクラスタ

他のコンポーネント パラメータで利用可能なターゲット クラスタ名を定義します。パラメータを定義するには、変数置換を使用します。

## [構成]-[ファイル解析]ページ

[構成] - [ファイル解析] ページには、以下のフィールドが含まれます。

### 名前(正規表現)

ディスカバリ中に検索および解析される構成ファイルの名前を定義します。正規表現の命名規則に従って、名前の定義にワイルドカード (^、\$ など) を使用することができます。

### 表示名

ディスカバリ後に表示される構成ファイルの名前を定義します。

### 説明

ブループリント項目を説明します。

### ファイル タイプ

構成ファイルのタイプを指定します(テキスト、バイナリ、ログなど)。

## 構成実行ファイル ページ

[構成実行ファイル] ページには、以下のフィールドが含まれます。

### 名前

構成実行ファイルの名前を定義します。この名前が **Configuration** フォルダの **Executables** の下に表示されます。名前は **Executables** フォルダ内で一意である必要があります。

### 説明

(オプション) ブループリント項目を説明します。ブループリント内のディレクティブ名およびツリー ビュー内のパラメータ名に、ツール ヒントとして説明が表示されます。

### ディレクティブ タイプ

パラメータ値を取得するために製品が使用するディレクティブのタイプを指定します。ディレクティブはすべて、値およびブールの結果を返します。検出されたパラメータがサービス ツリー ビューに表示されると、値がパラメータ名の横に表示されます。

**注:** [構成] ディレクティブ タイプは管理対象の構成ファイルから値を抽出します。

**デフォルト:** 定数

## ファイル

**Configuration - Files** フォルダまたは **Configuration - Executables** フォルダにある構成ファイルの名前を定義します。特定のファイル名または正規表現を指定できます。正規表現を指定する場合、**Configuration** フォルダ内の一致するファイルがすべて選択されます。複数のファイルが一致する場合、管理対象ファイルシステムのルートに最も近いファイルが選択されます。製品は変数置換を許可します。

## パス

構成ファイルのパスを定義します。ワイルドカードやパターンマッチングはサポートされませんが、変数置換は許可されます。パスは、絶対パス (**\$(Root)/conf** など) または管理対象ファイルシステムのルートからの相対パス (**/conf** など) で指定できます。構成ファイルが構成ファイルシステムのルートにある場合、**[パス]** を **\$(Root)** として定義するかまたは未定義のままにしておくことができます。

## パラメータ

ファイル内の構成パラメータの名前を定義します。ワイルドカードやパターンマッチングはサポートされませんが、変数置換は許可されます。値がグループ ファイル ブロックに属する場合、**[パラメータ]** 属性は未定義のままにしておくことができます。その場合、**[グループ パス]** 属性のみを指定します。

## 正規表現

正規表現を使用して名付けられた値を検出した場合に、フィルタするかどうかを指定します。

## パーレン

正規表現がカッコで囲まれた部分表現を含む場合、一致したときにどの部分表現が返されるのかを定義します。

**デフォルト : 0** ( **[パーレン]** 値を指定しない場合)

## グループ パス

階層的に構成された構成ファイルを指定します。名前パラメータにファイル内の [グループ パス] の値を使用します。これは、ファイルシステム パスの一部 (a/b/c など) のように指定します。

単一のファイルには同じ名前が何度も含まれることがよくあるので、一意のグループ パスを使用して、特定の値を一致する名前から区別します。たとえば、XML 構成ファイルは以下のような形式を持つことができます。ここでは、**server** タグと **buildDate** 属性および **type** 属性が何度も記述されています。

```
<configuration>
  <server name= "wxp123">
    <setup buildDate="10/30/2003" type="webserver"/>
  </server>
  <server name="wxp123" auxiliary="true">
    <setup buildDate="09/02/2002" type="webserver"/>
  </server>
  <server name="wxp124">
    <setup buildDate="10/29/2003" type="dataserver"/>
  </server>
</configuration>
```

検索するパスの一致する子を識別するには、[グループ パス] の名前 (または名前および値) のカンマ区切りリストを使用します。たとえば、サーバ **wxp124** の **buildDate** の値を抽出するには、以下の値を定義します。

## パラメータ

buildDate

## グループ パス

configuration/server,name=wxp124/setup

サーバ **wxp123** の補助データから **buildDate** 値を抽出するには、名前のみが必要です。これは、**server** タグと **auxiliary** 属性の組み合わせが 1 つしかないためです。

## パラメータ

buildDate

## グループ パス

configuration/server,auxiliary/setup

[グループ パス] 内のエレメントのいずれかまたはすべてを修飾できます。また、各エレメントに複数の修飾子を指定できます。ユーザが複数の修飾子を指定する場合、パスを選択するには、各修飾子が一致する必要があります。製品は変数置換を許可します。

構成ファイルの構造クラスでグループ ブロック修飾子が定義されている場合、代替構文が許可されます。グループ ブロックでは、修飾子の値をカッコで囲んで、ブロック名に付加できます。たとえば、属性名が [修飾子の子] であるサーバ **wxp124** から **buildDate** の値を抽出するには、以下の値を指定します。

#### パラメータ

**buildDate**

#### グループ パス

**configuration/server(wxp124)/setup**

丸カッコ内の値は、指定されたグループ ブロックの修飾子に一致します。構造クラス内の修飾子の子の値または定数の修飾子は、修飾子を定義します。また、修飾子がグループ 値であることもあります。

ファイル内のパラメータの位置についてよくわからない場合は、複数のパスを指定できます。指定された 2 つ以上のパスを、パイプ (|) で区切ります。製品は、一致が検出されるまで、左から右の順に、1 つずつパスを試します。例：

#### パラメータ

**buildDate**

#### グループ パス

**configuration/server(abc)/setup|configuration/server(xyz)/setup**

### 空の値を NULL と見なす

ファイル内の指定された構成パラメータに値が指定されていない場合、製品のデフォルト動作を無視するかどうかを指定します。デフォルト動作では、値は長さゼロの空の値に設定されます。[空の値を **NULL** と見なす] フィールドを指定すると、空の値を返すディレクティブと値を返さないディレクティブを区別できます。

はい

[デフォルト] フィールドの値を使用します。

いいえ

[デフォルト] フィールドの値を使用しません。

デフォルト： いいえ

### デフォルト

固定文字列または 1 つ以上の変数置換を含む文字列を定義します。ディレクティブの実行時に、値を判定できない場合、製品は、未定義値や長さゼロの値の代わりにこの値を使用します。

一部のケース（たとえば、コンポーネントブループリント プラグインがディレクティブを設定できる場合）では、デフォルト値でのディレクティブの置換が望まれないことがあります。プラグインの値をデフォルト値で置換しないようにするには、デフォルトを **Cohesion.PLACEHOLDER** に設定します。

### 大文字と小文字の区別

比較するときに大文字と小文字を区別するかどうかを指定します。

デフォルト： 大文字と小文字を区別する

注: デフォルト（[大文字と小文字を区別する]）オプションでは、**Agent\_conf** と **agent\_conf** は一致すると見なされません。[大文字と小文字を区別しない] オプションでは、**Agent\_conf** と **agent\_conf** は一致と見なされます。

## トランスレート

トランスレート処理がディレクティブの出力に適用する一意の名前を定義します。実行時に、指定された [トランスレート] 値の前に `$CCTranslation$__` を付加して、データベース内で値をトランスレート処理として識別できるようにします。トランスレート処理が一致しない場合は、元のディレクティブ結果が保持されます。

構成ファイルおよびレジストリから抽出されるディレクティブ値は、列挙型に属する暗号の文字列または整数であることがあります。各値は設定状態に対応しますが、値は状態を解釈しません。これらの値を意味のある文字列に変換して、ブループリントに表示する際にユーザにとって理解しやすいものにすることができます。

製品データベース内の値をマップして、トランスレート処理名を参照し、*変換前の値から変換後の値*へのトランスレート処理をトリガします。以下の例では、トランスレート処理 `$CCTranslation$__IIS SERVER STATE` がマップされます。

```
<BlueprintTranslation name="IIS server state"
coh_name="IIS_SERVER_STATE" coh_id="23501"
created_by="system_user">
  <BlueprintTranslationEntry translate_from="2"
    translate_to="Running" />
  <BlueprintTranslationEntry translate_from="4"
    translate_to="Stopped" />
  <BlueprintTranslationEntry translate_from="6"
    translate_to="Paused" />
</BlueprintTranslation>
```

トランスレート処理は、特定のブループリントとは関連付けられません。他のトランスレート処理との競合を回避するには、各トランスレート処理が一意の名前を持つようにします。

**注:** 製品のユーザインターフェースは、現在トランスレート処理テーブルの定義をサポートしていません。CA Configuration Automation データロード形式で (例に示すように) ファイルを作成し、データローダユーティリティを使用してそれらのファイルをデータベースにロードします。

## 変換

返された XML 形式のディレクティブ値をフィルタするために使用される XSL 変換を定義します。変換された値は元の戻り値を置換し、新しい XML 値または変換によって生成されたテキストのいずれかになります。

XSL 変換を実行するために、ディレクティブ値に適用される XSL ファイルの場所を指定します。変換の場所は、サーバから認識できる URL (file、http など)、またはサーバの CLASSPATH 内のファイル名になります。変換が取得され、ディレクティブの新しい値を生成するためにディレクティブ値に適用されます。

## 挿入

表示または後続の操作に見合った結果を生成するために、別の文字列内に挿入されるディレクティブの値を定義します。

挿入場所には \$(VALUE) (すべて大文字) を含む任意の文字列を指定できます。ディレクティブ値は \$(VALUE) を置換して、フィルタされた結果を生成します。たとえば、最初のディレクティブ値が 3 で、挿入文字列が 1.\$(VALUE).export である場合、フィルタされたディレクティブ値は 1.3.export です。

## 修飾子

検出結果を修飾します。[ホストはエージェントを使用] 修飾子または [Alterpoint デバイス] 修飾子を選択します。

## 修飾子パラメータ

製品が指定した修飾子に適用するパラメータを定義します。

## パーサ詳細

[構造クラス] または [パーサ] のいずれかを指定します。対応するドロップダウン リストからオプションを選択します。



## クエリの追加ペイン

［コンポーネント ブループリント］ ウィザードの［クエリの追加］ ペインには、以下のフィールドが含まれます。

### 名前

管理されたサーバ上のデータベースを問い合わせ、（必要に応じて）出力データから値を抽出するディレクティブを、どのディレクティブにするかを定義します。このディレクティブは、通常、以下の目的で使用されます。

- データベース行から列値を抽出するクエリを実行する。
- 管理されたサーバの変更を行うストアードプロシージャまたは値抽出のためのストアードプロシージャを実行する。
- 結果セットを抽出してディレクティブのマクロ ステップに表形式で表示するクエリを実行する。

指定される名前はブループリントの **executables** フォルダ内で一意である必要があります。

### 説明

（オプション）ブループリント項目を説明します。ブループリント内のディレクティブ名およびツリー ビュー内のパラメータ名に、ツール ヒントとして説明が表示されます。

### クエリタイプ

クエリの実行方法とクエリがデータを返すかどうかを製品に指示する以下のいずれかのクエリタイプを指定します。

#### 選択

クエリはデータを返す SQL の **Select** ステートメントです。

#### 挿入

クエリはデータを返さない SQL の **Insert** ステートメントです。

#### 削除

クエリはデータを返さない SQL の **Delete** ステートメントです。

#### 更新

クエリはデータを返さない SQL の **Update** ステートメントです。

#### その他

クエリは、データを返さない SQL の **alter table** ステートメントまたはその他の **DDL** ステートメントです。

#### ストアードプロシージャ

クエリ文字列は、データを返すことができるストアードプロシージャを呼び出します。

### クエリ

SQL クエリまたはストアードプロシージャの名前を定義します。クエリ構文はデータベースタイプによって異なる可能性があり変数置換が許可されています。製品は、クエリまたはストアードプロシージャが検出した行をすべて返します。結果セットをフィルタするには、列および正規表現の属性を使用します。

### 最大行数

クエリが返す最大行数を定義します。

デフォルト：5000

### プライマリ列

返された結果セットの列または列のエイリアスの名前を定義します。指定した列の値は、ディレクティブの値として取得されます。クエリが複数の行を返す場合、最初の行の指定列が使用されます。

## 構造クラス

パーサが取得する名前/値ペアにメタデータをマップするために使用されるグループとパラメータの階層コレクションを定義します。

### ノート(編集モードのみ)

選択したエレメントに関するノートを定義します。エレメントに関するノートの数、または「なし」が、角かっこ内に表示されます。ドロップダウンリストから、すべてのノートの表示、ノートの追加、またはノートの表示、更新、削除を行えます。

- [すべてのノートを表示] を選択すると、新しいページにすべてのノートがリスト表示されます。テーブルの並べ替え順序を変更するには、任意のテーブル見出しをクリックします。
- [新しいノート] を選択すると、以下のフィールドが表示されます。

#### 名前

ノートの名前を定義します。

#### テキスト

ノートのテキストを定義します。

既存のノートを選択すると、[ノート] フィールドにノートの名前が表示され、ノートのテキストがその下に表示されます。選択したノートは、更新、削除、またはキャンセルできます。

## ファイル構造クラス タブ

[ファイル構造クラス] タブには、以下のフィールドが含まれます。

### 名前

構造クラスの名前を定義します。[ブループリント] リストおよび([表示名] を指定しない場合) ツリー ビューに、指定された構造クラス名が表示されます。

### バージョン

構造クラスの構造クラス バージョンを定義します。[ブループリント] リストおよび([表示名] を指定しない場合) ツリー ビューに、指定されたバージョンが表示されます。

### 表示名

コンポーネント ツリー ビューに表示されるクラス名が [名前] および [バージョン] フィールドと異なる場合に、クラス名を定義します。たとえば、クラス機能をより正確に表す [表示名] の値を指定できます。

### 説明

(オプション) ブループリント項目を説明します。この説明は、[ブループリント] ツリー ビュー内のクラス名にツールヒントとして表示されます。

### 修復ジョブを許可

変更可能なブループリント エlement を、修復ジョブで指定したとおりに変更できるかどうかを指定します。

デフォルト： はい

### パーサ

定義されたタイプのファイルに対して使用するパーサを定義します。製品には、最も一般的な構成ファイル形式の製品による分析を可能にする、事前定義済みの **Lexical Analyzer** (レクサ) およびパーサのライブラリが含まれます。

## [File Structure Class Group]フィールドと[パラメータ]フィールド

[優先順位] タブの [グループの追加] および [パラメータの追加] ペインには、以下のフィールドが含まれます。

### 名前

優先順位グループの名前を定義します。

### 説明

ブループリント項目を説明します。

### 利用可能なカテゴリ

グループのカテゴリを指定します。1 つ以上のカテゴリをダブルクリックし、それらを [選択されたカテゴリ] 列へ移動します。

### 利用可能なフィルタ

グループのフィルタを指定します。1 つ以上のフィルタをダブルクリックし、それらを [選択されたフィルタ] 列へ移動します。

### ウェイト

エレメントの相対的な重要度を指定します。

- 低
- 中
- 高

デフォルト：中（ウェイトを割り当てていないエレメント）

### データタイプ

エレメントのデータ型を指定します（文字列、ブール、整数など）。

### 有効な値

データ型の範囲に対する有効なパラメータまたはグループの値を指定します（整数列挙、整数範囲、文字列列挙など）。

### デフォルト値

固定文字列または 1 つ以上の変数置換を含む文字列を定義します。実行時に、値を判定できない場合、未定義値や長さゼロの値の代わりにデフォルトの値が使用されます。

コンポーネントブループリントプラグインがディレクティブ値を設定できる場合、指定されたデフォルト値でプラグインの値が置換されないようにするために、デフォルト値を `Cohesion.PLACEHOLDER` に設定できます。

### 表示/非表示

コンポーネントで構成パラメータまたはグループ値を表示するかどうかを指定します。また、ブループリント UI で定義されたデフォルト値を表示するかどうかを指定します。値を非表示にする場合、値は暗号化されます。

#### 値を表示

デフォルト値と、構成パラメータまたはグループ値を表示します。

#### 値を非表示

ブループリント UI のデフォルト値と、コンポーネント ビューア UI の構成パラメータまたはグループ値を非表示にします。

- 値が機密である（パスワードなど）。
- 値がバイナリである。
- 表示するには、値が長すぎる。

非表示の構成パラメータまたはグループ値はデータベース内で暗号化されます。UI では表示されないか、または参照できません。

**注:** 後で値を表示することを決定した場合は、このフィールドのセキュリティを保証するために、値を再入力します。

#### デフォルト: 値を表示

### 大文字と小文字の区別

比較するときに大文字と小文字を区別するかどうかを指定します。

**デフォルト:** 大文字と小文字を区別する

**注:** デフォルト（[大文字と小文字を区別する]）オプションでは、Agent\_conf と agent\_conf は一致すると見なされません。[大文字と小文字を区別しない] オプションでは、Agent\_conf と agent\_conf は一致と見なされます。

### 解釈方法

構成パラメータの文字列形式に関するヒント、および関連するコンポーネントによるその使用方法を定義します。解釈されたパラメータ値は、コンテキスト依存パーサを使用して検査されます。製品は、パーサを使用して、複雑なパラメータ文字列から複数のサブ値を抽出できます。

**デフォルト:** 空（解釈はありません）。

### 関係キー

製品が [解釈方法] の値に従って関係を決定し割り当てるかどうかを指定します。

関係を確立するには、[解釈方法]の値を設定して、[関係キー]フィールドを [はい] に設定します。一部の解釈は関係を定義できません。

デフォルト： いいえ

### 関係タイプ

以下の関係タイプのいずれかを指定します。

#### 通信する

アプリケーションとデータベース サーバの関係を確立します。

#### 管理する

別のサーバを管理するサーバ間の関係を確立します。たとえば、VMware ESX ホストを管理する VMware vCenter Server 間の関係を表示するには [管理] を選択します。

#### ホストする

別のサーバをホストするサーバ間の関係を確立します。たとえば、VMware 仮想マシンをホストする VMware ESX ホスト間の関係を表示するには [ホスト] を選択します。

#### 使用する

サーバ間の関係を確立します。

### 解釈済みのサーバ

他のコンポーネント パラメータで利用可能なターゲット サーバの情報を定義します。パラメータを定義するには、変数置換を使用します。

### 解釈済みのソース インスタンス

ソース データベース インスタンスまたはアプリケーション インスタンスを定義します。

### 解釈済みのターゲット インスタンス

他のコンポーネント パラメータで利用可能なターゲット データベース インスタンスまたはアプリケーション インスタンスを定義します。パラメータを定義するには、変数置換を使用します。

### 解釈済みのアプリケーション

他のパラメータで利用可能なターゲットアプリケーションの情報を指定します。パラメータの定義には、変数置換を使用します。

### 解釈されたクラスタ

他のコンポーネントパラメータで利用可能なターゲットクラスタ名を定義します。パラメータを定義するには、変数置換を使用します。

### 予期される値

パラメータまたはグループの予期される値を定義します。

### 値

ファイル内のグループの一意の値を定義します。

### 修飾子の子([グループの追加]のみ)

グループ下の修飾子として使用されるパラメータを定義します。

## マクロ ページ

[マクロ] ページには、以下のフィールドが含まれます。

### 名前

マクロの名前を定義します。

### 説明

ブループリント項目を説明します。

### フォルダ

項目の場所を定義します。

### 診断

問題を診断するために、製品がマクロを使用するかどうかを指定します。

デフォルト：いいえ

### 読み取り専用

マクロがターゲットシステムを変更できるかどうかを指定します。

注: 読み取り専用マクロを実行するには、ユーザがサーバまたはサービスの表示権限を持っている必要があります。

デフォルト：いいえ(マクロはターゲットシステムを変更できません)。



## 完了ページ

このウィザード ページには、以下のフィールドが含まれます。

### このブループリントが他のコンポーネントを持つことを許可

ブループリントが他のコンポーネントを持つことができるかどうかを指定します。

### このブループリントが他のコンポーネントにネストすることを許可

他のコンポーネントがブループリントを含むことができるかどうかを指定します。

**注:** デフォルトでは、[このブループリントが他のコンポーネントを持つことを許可] および、[このブループリントが他のコンポーネントにネストすることを許可] チェックボックスはオンになっています。

### このブループリントが名前付きコンポーネントのみを持つことを許可

ブループリントが名前付きコンポーネントのみを持つことができるかどうかを指定します。

### このブループリントはネストを使用しません

ブループリントがネストを使用できるかどうかを指定します。

### リフレッシュ順序

特定のサーバのすべてのコンポーネントの中で、このコンポーネントがリフレッシュされる順位を指定します。製品は、各サーバのコンポーネントを順番にリフレッシュし、並行して他のサーバのコンポーネントをリフレッシュします。

各コンポーネントが変数置換に関して相互に依存している場合、リフレッシュ順序が重要になります。たとえば、1つのコンポーネントが変数置換の目的で別のコンポーネントのパラメータ値を使用する場合は、まず、別のコンポーネントがリフレッシュされる必要があります。このような場合に「リフレッシュ順序」が指定されていないと、依存するコンポーネントは最新のリフレッシュで更新されていない値を取得する可能性があります。

「リフレッシュ順序」は以下の値を持っています。1つのレベル内では、保証される特定の順序はありません。

- **考慮しない**：コンポーネントは、「[中期]」の順序が設定された場合と同じように、リフレッシュされます。
- **初期化**：初期化に設定されているコンポーネントが、最初にリフレッシュされます。
- **最初**
- **初期**
- **中期**
- **後期**
- **最後**
- **クリーンアップ**：クリーンアップに設定されているコンポーネントが最後にリフレッシュされます。

### 修飾子

検出された結果を修飾するために、以下のいずれかのオプションを指定します。

- デバイス権限
- IIS ネスティング
- すべて置換
- Sun アプリケーション サーバ ネスト
- TIBCO
- TIBCO ネスティング

### 修飾子パラメータ

選択した修飾子のパラメータを定義します。

### 最近隣

コンポーネントが複数のコンポーネントにネストできる場合、（階層数に関して）最も近いファイル システム ルートを持つコンポーネントが選択されるかどうかを指定します。

デフォルト： はい

### ネスト先

コンポーネントが他のコンポーネントにネストするかどうかを判別するファイルシステム関係を指定します。

注: [このブループリントはネストを使用しない] が選択されている場合、[ネスト先] は不要です。

### 子

ファイルシステム ディレクトリ ツリー内でコンポーネントより上位のファイルシステム ルートを持つコンポーネントに、コンポーネントがネストするように定義します。

### 子または兄弟

ファイルシステム ディレクトリ ツリー内でコンポーネントと同等または上位のファイルシステム ルートを持つコンポーネントに、コンポーネントがネストするように定義します。

### 直接の子

ファイルシステム ディレクトリ ツリー内でコンポーネントより 1 レベル上位のファイルシステム ルートを持つコンポーネントにのみ、コンポーネントがネストできるように定義します。

### 兄弟

同じファイルシステム ルートを持つコンポーネントに、コンポーネントがネストするように定義します。

### 以下にのみネスト

このページで定義されたネストを使用するブループリントを定義します。[利用可能なコンポーネント ブループリント] 列のブループリントをダブルクリックして、[選択されたコンポーネント ブループリント] 列に移動します。

### ルートからのパス

このコンポーネントより上位のファイルシステム ルートを持つ別のコンポーネントにのみ、それらの間の正確な相対パスにより、このコンポーネントがネストするように指定します。一致する特定のネストを選択するかまたはファイルシステム ディレクトリ ツリー内の正確な場所にネストを制限するには、このオプションを使用します。

### ルートからの階層数

このコンポーネントより上位のファイルシステム ルートを持つ別のコンポーネントにのみ、それらの間の正確な階層数により、このコンポーネントがネストするように指定します。一致する特定のネストを選択するかまたはファイルシステム ディレクトリ ツリー内の正確な階層の深さにネストを制限するには、このオプションを使用します。



## 第 5 章: [コンポーネント グループ化オプション] タブでのネスト修飾子によるブループリントのカスタマイズ

---

ごく一部のアプリケーションには、存在を判断するためのファイル インジケータまたはレジストリ インジケータがありません。ただし、これらのアプリケーションの存在は、コマンドの出力またはファイル内のエントリから判断できます。ブループリントではネスト修飾子（プラグイン プログラム）を使用して、そのようなアプリケーションを検出します。

このセクションのトピックでは、ネスト修飾子を使用するブループリントを変更することで、子コンポーネントを検出するためのガイドラインを提供します。Oracle および Microsoft SQL のブループリントを例にとります。

このセクションには、以下のトピックが含まれています。

[インスタンス情報をサポートする Oracle および MSSQL のブループリント \(P. 127\)](#)

[インスタンス情報をサポートする Oracle および MSSQL のブループリントのカスタマイズ: \[コンポーネント グループ化オプション\] タブ \(P. 128\)](#)

[インスタンス情報をサポートする Oracle および MSSQL のブループリントのカスタマイズ: 入力修飾子としてのコンポーネント パラメータ \(P. 129\)](#)

### インスタンス情報をサポートする Oracle および MSSQL のブループリント

CA Configuration Automation r12.8 SP02 を開始する場合は、以下の Oracle および MSSQL のブループリントを使用して、対応するインスタンスまたはデータベースをコンポーネントとして検出できます。

- Microsoft SQL 2008 Server インスタンス v10.\* r1.0.0 （親ブループリント）  
Microsoft SQL 2008 Server データベース v10.\* r1.0.0 （子ブループリント）

- Microsoft SQL 2012 Server インスタンス v11.\* r1.0.0 (親ブループリント)  
Microsoft SQL 2012 Server データベース v11.\* r1.0.0 (子ブループリント)
- Oracle 10g Database (UNIX) v10.\* r1.0.0 (親ブループリント)  
Oracle 10g Database インスタンス (UNIX) v10.\* r1.0.0 (子ブループリント)
- Oracle 11g Database (UNIX) v11.\* r1.0.0 (親ブループリント)  
Oracle 11g Database インスタンス (UNIX) v11.\* r1.0.0 (子ブループリント)

## インスタンス情報をサポートする Oracle および MSSQL のブループリントのカスタマイズ: [コンポーネントグループ化オプション]タブ

[コンポーネントグループ化オプション] タブで定義された修飾子では、[以下にのみネスト] フィールドにリスト表示されているブループリント参照を使用します。既存のブループリントをカスタマイズする場合は、そのブループリントへの参照を、その他のブループリント内の [以下にのみネスト] フィールドで変更してください。

例:

親ブループリント Oracle 11g Database (UNIX) v11.\* r1.0.0 は、Oracle 11g Database インスタンス (UNIX) v11.\* r1.0.0 子ブループリントで参照されます。修飾子 [Oracle インスタンス ネスティング] では、子コンポーネントを検出するために子ブループリント参照を使用します。

既存のブループリントをカスタマイズする場合は、親ブループリントと子ブループリントの両方を変更し、さらに子ブループリントから親ブループリントへの参照も変更してください。



## インスタンス情報をサポートする Oracle および MSSQL のブループリントのカスタマイズ: 入力修飾子としてのコンポーネントパラメータ

Oracle および MSSQL のブループリントは、コンポーネントパラメータを使用して、インスタンスおよびデータベースのリストを取得します。[コンポーネントグループ化オプション] タブで定義された修飾子では、子コンポーネントを検出するために、これらのインスタンスとデータベースを使用します。

たとえば、以下の親 MSSQL ブループリントには、MSSQL サーバインスタンス内のデータベースのリストを取得するために、コンポーネントパラメータとして **TheDatabasesList** が含まれています。

- Microsoft SQL 2008 Server インスタンス v10.\* r1.0.0
- Microsoft SQL 2012 Server インスタンス v11.\* r1.0.0

パラメータでは、データベースのリストを取得するために、**ADODB.Connection** オブジェクトを介して SQL サーバに対してクエリを実行するスクリプトを使用します。

同様に、以下の親 Oracle ブループリントには、Oracle データベース内の子コンポーネントを検出するために、コンポーネントパラメータとして **Instances** が含まれています。

- Oracle 10g Database (UNIX) v10.\* r1.0.0
- Oracle 11g Database (UNIX) v11.\* r1.0.0

パラメータには、**oratab** ファイルエントリを解析する以下のスクリプトが使用されます。

```
list=`cat /etc/oratab | grep "$(Root)" | cut -f 1 -d :`;for inst in $list;do echo "$inst,";done;
```

環境の要件を満たすには、修飾子で使用されているインスタンスまたはデータベースのリストが検出されるように、コンポーネントパラメータを変更します。