CA Configuration Automation®

ブループリント開発者ガイド

r12.8

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

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 エレメント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 プローブを使用して 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 エレメントの追加

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 エレメント、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 エントリの数が限られている場合は、ルート以下の該当エレメント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 エレメント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、エレメント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。

管理対象エレメントの追加

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 エレメントの追加

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。

ユーティリティ

［ユーティリティ］プライマリ フォルダには、共通の管理タスク（たとえば、コンポーネントを開始または停止させるプログラムかスクリプト、またはシステム情報の表示、メモリ統計、またはディスク ボリューム統計などのサーバまたはサービスについての追加情報を提供するプログラムかスクリプト）を実行できる実行可能ファイル、マクロ、およびスクリプトが含まれます。

ユーティリティ追加エレメント

追加対象	アクセス方法と手順
ファイル	ファイルの $(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	ファイルの $(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。

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。

ランタイム

［ランタイム］プライマリ フォルダは、コンポーネント ランタイム ファイルを定義し、それらを迅速に見つけて表示するのに役立ちます。 ランタイム ファイル（たとえばログ ファイル）は通常頻繁に変更されます。比較分析からファイル コンテンツを除外するには、［ランタイム］プライマリ フォルダでファイルを定義します。

ランタイム 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。

マニュアル

ドキュメント プライマリ フォルダは、コンポーネントの管理対象ドキュメント ファイル（たとえば readme ファイル、PDF ファイルまたは HTML バージョンの製品マニュアル）を検索する方法を定義します。 ドキュメント フォルダでは、企業のイントラネットまたはベンダー サポート サイトのようなコンポーネント情報またはドキュメントの追加ソースへの明示的な URL リンクを、ユーザが追加できます。

注: 参照 URL へのアクセスはユーザのネットワーク構成によって決定されます。

マニュアル 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。
URL	URL フォルダをクリックし、次に［URL 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 エレメント 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. ［管理］リンクをクリックし、［ブループリント］タブをクリックします。
2. ［ブループリント］リンクをクリックします。

   ［ブループリント］ページが開き、既存のすべてのブループリントがリスト表示されます。

   ［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。
3. ［コンポーネント ブループリント］ウィザードの［ブループリント］ページで、以下のフィールドを設定します。

   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。

   コンポーネント バージョン

   ソフトウェア コンポーネントのバージョン（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

この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

ブループリントを説明します。 説明はブループリントの［詳細］ページに表示されます。

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。
- IOS を使用する Cisco のルータおよびネットワーク スイッチに固有のものとなる

後で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。

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。

4. ［次へ］をクリックします。

［ディスカバリ方式］ページが開き、選択された［ファイル インジケータ］エレメントが左ペインに表示されます。

5. 以下の［検索オプション］フィールドを設定します。

検索開始場所

ディスカバリ インジケータ検索の開始点を定義します。 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

ディスカバリ インジケータ検索の階層の深さ（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

常時検出コンポーネント

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］を選択すると、CA Configuration Automation はファイル 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。 ［はい］を選択すると、検出不可能または検出不要なサービス内のコンポーネントを定義および管理できます。

デフォルト： No

6. ［ディレクトリ/ファイルの追加］をクリックします。

7. ［新規ファイルの追加］ペインで以下の［ファイル インジケータ］フィールドを設定します。

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

新規ファイルまたはディレクトリをブループリントに追加するかどうかを指定します。

(オプション)ルートからのパス

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。

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 を検索するために定義します。 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。

- 検索では表現の完全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 * および ? を ディレクトリ名で使用できます。 例： ¥*conf、¥conf*、/*bin、/bin*、/b*n、/bi?

(オプション)ルートからの階層数

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。

検索ルートからの階層の深さ（フォルダまたはディレクトリの階層数）を定義します。

8. ［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］および ¥HKEY_LOCAL_MACHINE フォルダが左ペインに表示されます。¥HKEY_LOCAL_MACHINE フォルダ用の［検索オプション］フィールドが右ペインに表示されます。

9. 以下の［検索オプション］フィールドを設定し、［保存］をクリックします。

検索開始場所

ディスカバリ インジケータ検索の開始点を定義します。 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

ディスカバリ インジケータ検索の階層の深さ（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

常時検出コンポーネント

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］を選択すると、CA Configuration Automation はファイル 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。 ［はい］を選択すると、検出不可能または検出不要なサービス内のコンポーネントを定義および管理できます。

デフォルト： No

10. ［レジストリ値/キーの追加］をクリックします。

11. ［新規レジストリの追加］ペインで以下の［レジストリ インジケータ］フィールドを設定します。

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

新規ファイルまたはディレクトリをブループリントに追加するかどうかを指定します。

ルートからのパス

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。

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

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。

検索ルートからの階層の深さ（フォルダまたはディレクトリの階層数）を指定します。

12. ［ネットワーク プローブ］リンクをクリックします。

注: 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

13. 以下の［ネットワーク プローブ］フィールドを設定します。

名前

ネットワーク プローブの名前を定義します。

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ポート番号を、カンマで区切って定義します。

正規表現一致

プローブが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。

14. ［次へ］をクリックします。

左ペインには[ディスカバリ検証ルール]フォルダが表示されます。右ペインには［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。

15. 以下のフィールドを指定します。

名前

［ディスカバリ検証ルール］の名前を定義します。

説明

ルールの目的を説明します。

ディレクティブ タイプ

サービスで管理されるエレメントから値を抽出するため、または管理対象サーバから値を取得するために使用され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 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 パラメータを定義します。

§ 変数と固定テキストを複雑な文字列に組み立て、複数のソースからデータを組み合わせます。

§ 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て、（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。

§ データベース内の行から列の値を抽出するためのクエリを実行します。

§ 管理対象サーバでの変更または値の抽出を行うためのストアド プロシージャを実行します。

§ 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ためのクエリを実行します。

ファイルの取得

特定のファイルの内容を取得し（そのファイルの場所が既知の場合）、内容をフィルタしてディレクティブ値を定義します。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。

§ 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します。

§ 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ます。必要に応じて、取得された値をフィルタしてディレクティブ値を設定します。

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ます。必要に応じて、そのリストをフィルタし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すべてリスト表示します。 必要に応じて、そのリストをフィルタし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。 必要に応じて、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。オプションのプローブをポートに送信できます。また、ディレクティブ値としてソケットから収集された応答にも送信できます。

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ます。必要に応じて、その結果をフィルタして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。コマンドからの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します（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）にアクセスします。

§ カスタム スクリプトおよびカスタム ツールを利用して、それらの出力をブループリントにインポートします。

§ ユーティリティとスクリプトを実行し、管理対象サーバを更新します。

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定変数に解析します。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。

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ときを基準にして、ディスカバリ中に検証ディレクティブが実行されるときを指定します。

後期

検証ディレクティブはコンポーネント パラメータの値を必要とします（変数置換ため）。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指定します。

- 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、CA Configuration Automation はそれを有効な値と考えます。

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*)

通常のコンポーネント変数置換を指定します。

$(#*parameter_name*)

親コンポーネント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) として。
- 値が表現に一致しない場合は値なしとして。

パーレン

かっこで囲まれた部分表現を含む正規表現を定義した場合、一致したときに返される部分表現を定義します。 正規表現にかっこで囲まれた部分表現が含まれない場合は、デフォルト値が使用されます。

デフォルト： 0

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

ディレクティブ出力に適用されるトランスレート処理を定義します。 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、多くの場合、列挙型に属する暗号の文字列または整数です。各値は設定状態に対応し、値は解釈を示します。 これらの値は、ブループリント内で表示する際に明確になるように、意味のある文字列に変換できます。

CCA データベース内の値のマッピングを定義します。このマッピングは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。 以下の例は、$CCTranslation$__IIS SERVER STATE という名前のトランスレート処理のマッピングです。

```
<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関連付けられません。名前は任意のブループリントからの参照名として使用します。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: 現在、CA Configuration Automation には、トランスレート処理テーブルを定義するためのユーザ インターフェース サポートは含まれません。 ファイルを作成した後、データ ローダ ユーティリティを使用してデータベースにそれらをロードするには、例の CA Configuration Automation データ ロード形式を使用します。

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必要な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。

別の文字列内で $(VALUE) （すべて大文字）を含む文字列は、挿入場所として機能できます。 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 になります。

修飾子

［ホストはエージェントを使用］または［Alterpoint デバイス］を指定します。

修飾子パラメータ

指定した修飾子に適用するパラメータを定義します。

16. ［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、［ファイル管理］ページが表示されます。 左ペインには $(Root) フォルダが表示され、右ペインには［ファイル管理オプション］が表示されます。 ［ファイル管理］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 エレメントを定義できます。

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 エントリの数が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 エントリのみに焦点を絞って指定します。［ファイル管理］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。

17. ［ファイル管理］ページで、以下のフィールドを設定します。

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。

すべてのファイルを取得

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ファイルをすべて取得します。

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。

18. （オプション）［ディレクトリの追加］または［ファイルの追加］（または両方）をクリックし、名前および説明を入力して、［保存］をクリックします。

    ディレクトリまたはファイルが［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。

19. （オプション）左ペインでノードを選択し、手順 18 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。

    ディレクトリまたはファイルが［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。

20. ［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 18 および 19 で作成したディレクトリおよびファイル構造が、左ペインの $(Root) フォルダの下に表示されます。

    次の手順に従ってください：

    a. $(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。

    b. （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、前の *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。

    c. （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 b を繰り返し実行します。

    d. 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。

e. 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。

選択したオプションが［選択されたカテゴリ］列に追加されます。

f. ［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。

選択したオプションが［選択されたフィルタ］列に追加されます。

g. 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。

h. ［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。

i. 以下のフィールドを指定します。

名前

ルール名を定義します。

説明

ルールの目的を説明します。

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
- § 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)
- = (等しい、大文字/小文字を区別しない)
- != (do not ignore case)
- > (次の値より大きい)
- >= (次の値以上)
- < (次の値より小さい)
- <= (次の値以下)
- 整数範囲 (両端を含める)
- 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］はルールの条件を定義します。

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

21. ［レジストリ管理］リンクをクリックします。
22. 以下の［レジストリ管理］フィールドを設定し、［保存］をクリックします。

    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。

    すべてのファイルを取得

    ディスカバリ操作が取得する要素の数を指定します。

    オン： 操作はすべての要素を取得します。

    オフ： 操作は、［最大ファイル数］フィールドで指定した数の要素まで取得します。

    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要素数を定義します。
23. （オプション）［キーの追加］または［ファイルの追加］（または両方）をクリックし、名前および説明を入力して、［保存］をクリックします。

    キーまたはファイルが［レジストリ管理］ペイン内の $(Root) ディレクトリの下に追加されます。
24. （オプション）左ペインでノードを選択し、手順 23 を繰り返してネストされた他のキーまたは値を作成します。

    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。
25. ［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。

    左ペインには $(Root) フォルダが表示され、右ペインには［優先順位］タブが表示されます。 ページに、対応するテーブルの既存のキーと値が表示されます。

    a. （オプション）ディレクトリの優先順位の設定に使用する列を選択し、［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 20 で説明されている順位付けと同様です。

    b. ［キーの追加］をクリックします。

    c. キーの名前と説明を入力します。

d. ［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。

e. ［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。［選択されフィルタ］列に追加されます。

f. キーのウェイトを選択します。また、フィルタが再帰的かどうかも選択します。

g. 以下のフィールドに入力し、［保存］をクリックします。

デフォルト

レジストリ キーのデフォルト値を指定します。

解釈方法

キーの解釈方法を指定します。

- § 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 タイプ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

値を表示するかどうかを指定します。

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。

解釈済みのインスタンス

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。

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。

a. 左ペインで、値を割り当てるキーを選択します。

b. 右ペインで、［値の追加］をクリックします。

c. ［名前］および［説明］を入力します。

d. ［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。

e. ［利用可能フィルタ］列のオプション（複数可）をダブルクリックして、［選択済みフィルタ］列に追加します。

f. ［ウェイト］選択します。また、フィルタが再帰的かどうかも選択します。

g. 以下の［値詳細］を入力し、［保存］をクリックします。

デフォルト

このレジストリ キーのデフォルト値を指定します。

解釈方法

キーが以下のいずれかとして解釈されるように指定します：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 タイプ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

値を表示するかどうかを指定します。

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 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。

解釈済みのインスタンス

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。

キーが［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。

1. ［データ管理］リンクをクリックします。

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

2. 適切なフィールドに以下の情報を入力してブループリントのデータベースを定義し、［保存］をクリックします。

   名前

   データベース名を指定します。

   説明

   データベースの目的を説明します。

   データベース

   データベース アクセスの説明を指定します。

   ユーザ

   データベースにアクセスできるユーザ名を指定します。

   パスワード

   ユーザ アカウントのパスワードを指定します。

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指定します。

ポート

データベース ホスト サーバのリスニング ポートを指定します。

DB タイプ

次のいずれかを指定します。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。

環境

コマンドが実行される前に設定される環境変数を指定します。

リモート実行可能ファイルは、CA Configuration Automation エージェントの環境内で呼び出されます。 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と値の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（たとえば、*<var_name>=<value>,<var_name2>=<value2>*）です。

エージェントはコマンドを呼び出す前にこれらの環境変数を定義します。 変数置換方法がサポートされています。

データベースが［データ管理］ペインに表示されます。

3. ［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。

4. 以下のフィールドに適切な情報を入力し、[保存]をクリックします。

名前

パラメータ名を指定します。 パラメータの代入表現の形式は $(*VariableName*) です。

VariableName は、コンポーネントで定義されているディスカバリ パラメータの名前です。 デフォルトでは、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。 パラメータ式は、文字列リテラル内に埋め込むか、または単独で使用することができます。 再帰的な複数の置換を同じ式で定義できます。たとえば、置換の値はパラメータ式の文字列の場合があり、その場合、再帰的に評価されます。

例：

ディスカバリ パラメータに以下の値があるとします。

```
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

パラメータの目的を説明します。

フォルダ

パラメータの場所を指定します。

選択されたカテゴリ

要素を整理するために以下のカテゴリを指定します。

環境管理

コンポーネントの一般使用および管理に関連する管理設定を指定します。管理設定の例は、バックアップの方法、キャッシュの削除の時期、再試行の回数などです。

設定

コンポーネント構成の設定情報を指定します。ただし、［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］は除きます。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 Web ページです。

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、またはユーザのガイドとして機能するエレメントを指定します。 たとえば、マニュアル、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

ログの場所、ログ レベル、デバッグ出力、診断変数タイプを設定できるエレメントを指定します。

ネットワーク

コンポーネントのネットワーク関連設定を表すエレメントを指定します。 たとえば、SNMP 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 LDAP 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

パフォーマンスに影響を与え、通常は構成パラメータの特定サブセットになっている設定を指定します。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

製品に関する一般的（静的）な情報を指定します。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 リソースを指定します。 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、CPU です。

注: 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

セキュリティ関連設定を表し、通常は構成パラメータの特定サブセットになっているエレメントを指定します。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

何らかの規則性によって変化するエレメントを指定します。たとえば、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

製品のバージョンまたはパッチのレベルを示すエレメントを指定します。

［利用可能なカテゴリ］列のカテゴリをダブルクリックして、［選択されたカテゴリ］列に移動します。

選択されたフィルタ

以下のエレメントが比較操作から除外されるように指定します。

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を指定します。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指定します。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

1 つのサーバに固有のエレメントを指定します。 たとえば、サーバ名や IP アドレスです。 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。

以下のエレメントが変更検出操作およびルール コンプライアンス操作から除外されるように指定します。

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のあるエレメントを指定します。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指定します。 ルール コンプライアンス操作と結果から除外するエレメントは、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

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ではないエレメントを指定します。 たとえば、ログ ファイル、プロセスの開始時刻、レジストリ イベント カウンタなどです。 時刻変化のパラメータ（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。

［利用可能なフィルタ］列のフィルタをダブルクリックして、［選択されたフィルタ］列に移動します。

ウェイト

エレメントの相対的な重要度を［低］、［中］、または［高］で指定します。 ウェイトのない（ウェイトが割り当てられていない）エレメントは、［中］とみなされます。

ディレクティブ タイプ

サービス内で管理されるエレメントから値を抽出するため、または管理対象サーバから値を取得するために使用され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 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

後続の変数置換または複雑な文字列構造に使用でき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、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をパラメータとして提供する。 ベンダー名など。

設定

解析済みの構成ファイルまたは構成実行可能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

管理対象サーバ上のデータベースに対してクエリを実行し、オプションで出力をフィルタして、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。

- データベース内の行から列の値を抽出するためのクエリを実行する。
- 管理対象サーバでの変更または値の抽出を行うためのストアド 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ためのクエリを実行する。

ファイルの取得

特定のファイルの内容を取得し（そのファイルの場所が既知の場合）、それをフィルタしてディレクティブ値を定義します。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。

- 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 ディレクティブ内で表示、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、オプションで出力をフィルタして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、その値をオプションでフィルタしてディレクティブ値を設定します。

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、そのリストをオプションでフィルタし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すべてリスト表示し、そのリストをオプションでフィルタし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。オプションで、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。オプションのプローブをポートに送信できます。また、ディレクティブ値としてソケットから収集された応答にも送信できます。

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、その結果をオプションでフィルタして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。コマンドからの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。

- 実行可能ファイルの出力からのみ利用できる構成情報にアクセスする。 一般に、オペレーティング システム構成はこのようにキャプチャされます。
- 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。
- 操作スタッフによって開発されたカスタム スクリプトおよびカスタム ツールを利用して、それらの出力をブループリントにインポートする。
- ユーティリティ プログラムおよびスクリプトを実行して、管理対象サーバを変更する。

ユーザ入力

ユーザにディレクティブ値の入力を求めます。

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指定します。ディレクティブが実行され、値を特定できない場合、未定義またはゼロ長の値の代わりに［デフォルト］の値が使用されます。

保持

サービス内でのパラメータの保持について指定します。

- ［常に保持］- パラメータが保存され、サービス ツリー ビューに表示されることを意味します。デフォルトは［常に保持］です。
- ［保持しない］ - パラメータの使用目的はディスカバリのみであり、パラメータが保存されないことを意味します。
- ［NULL ではない場合は保持］- パラメータが保存され、値がある場合にサービス ツリー ビューに表示されることを意味します。

表示/非表示

エレメントおよびエレメントの値を表示するかどうかを指定します。

- ［値を表示］は、パラメータ値が通常どおり表示されることを意味します。 デフォルトは［値を表示］です。
- ［値を非表示］は、パラメータ値が表示されず、代わりに ******* として表示されることを意味します。パラメータ値が機密情報（パスワードなど）の場合、またはパラメータ値がバイナリや長すぎる値のため表示できない場合、パラメータを非表示にします。非表示のパラメータはデータベース内で暗号化され、UI では表示されないか、または参照できません。

  注: このフィールドの内容に対するセキュリティを保証するために、後になって値を表示することにした場合は、値を入力し直す必要があります。
- ［エレメントを非表示］により、パラメータを非表示にすることができます。エレメントの値を変数置換に使用する場合でも、サービス ツリー ビューにエレメントを表示しないときは、エレメントを非表示にします。

大文字と小文字の区別

値のテキストの大文字と小文字の違いを CA Configuration Automation が無視するかどうかを指定します。 ［大文字と小文字を区別する］（デフォルト）は、CA Configuration Automation がすべての相違を識別およびレポートすることを意味します。 ［大文字と小文字を区別しない］は、テキストの大文字と小文字の違いが無視されることを意味します。

解釈方法

関連付けられたコンポーネントで使用可能な構成パラメータの文字列形式を提供します。 解釈されたパラメータ値は、CA Configuration Automation によってコンテキスト依存パーサで検査されます。コンテキスト依存パーサでは、複雑なパラメータ文字列内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

このフィールドが[はい]に設定されている場合に、CA Configuration Automation が［解釈方法］フィールドを使用して関係を決定して割り当てるように指定します。 関係が定義されていない解釈もあります。

デフォルト： ×

重要: 関係を確立するために、［解釈方法］フィールドで適切な値を定義し、［関係］フィールドを［はい］に設定してください。

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。

解釈済みのサーバ名

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。

解釈済みのインスタンス名

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。

解釈済みのアプリケーション名

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

ディレクティブ出力に適用されるトランスレート処理を指定します。 トランスレート処理が一致しない場合、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、多くの場合、列挙型に属する暗号の文字列または整数です。各値は設定状態に対応しますが、値は解釈を示しません。 これらの値は、ブループリント内で表示する際に明確になるように、意味のある文字列に変換できます。

CCA データベース内の値のマッピングを定義します。このマッピングは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。 以下の例は、$CCTranslation$__IIS SERVER STATE という名前のトランスレート処理のマッピングです。

```
<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関連付けられません。名前は任意のブループリントから参照されます。 各トランスレート処理には一意の名前を付けて、他のトランスレート処理と競合しないようにする必要があります。

注: 現在、CA Configuration Automation ユーザ インターフェースでは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（上記の例を参照）のファイルは、データ ローダ ユーティリティを使用してデータベースに作成およびロードする必要があります。

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 変換を指定します。 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 内に存在する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必要な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指定します。

別の文字列内で $(VALUE)（すべて大文字）を含む文字列は、挿入場所として機能できます。 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 になります。

修飾子

［ホストはエージェントを使用］または［Alterpoint デバイス］を指定します。

修飾子パラメータ

修飾子に適用されるパラメータを指定します。

設定が保存されます。

1. ［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。

2. ［ファイル解析］ページの以下のフィールドに入力し、［構成実行ファイル］リンク（［ファイル解析］ページの上部）をクリックします。

   名前（正規表現）

   ディスカバリ中に検索および解析する構成ファイルの名前を指定します。 正規表現パターン マッチング構文を使用して、名前の定義に正規表現（^、$ など）を使用することができます。

   表示名

   ディスカバリの結果表示される構成ファイルの名前を指定します。

   説明

   構成ファイルの目的を説明します。

   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）。

3. ［構成実行ファイル］ページで、以下のフィールドに入力し、［保存］をクリックします。

   名前

   Executables の下の Configuration フォルダに表示されるディレクティブの名前を指定します。 名前は Executables フォルダ内で一意である必要があります。

   説明

   実行可能ディレクティブの目的およびソースについて説明するオプションのテキストです。

   この説明は、ブループリント内の実行可能ディレクティブ名、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表示されます。

   ディレクティブ タイプ

   実行可能ファイルのパラメータ値を取得するために使用され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、値が実行ファイル パラメータ名の横に表示されます。

   デフォルト： 定数

   ［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。

ファイル

構成ファイルの名前を指定します。

ファイルは、configuration/files フォルダまたは configuration/executables フォルダのいずれかに存在する必要があります。 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、CA Configuration Automation は configuration フォルダ内の一致するファイルをすべて選択します。 複数のファイルが一致する場合、CA Configuration Automation は管理対象ファイル システムのルートに最も近いファイルを選択します。

変数置換は許可されています。

パス

構成ファイルのパスを指定します。 ワイルドカードまたはパターン マッチングはサポートされていません。ただし、変数置換は許可されています。 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、パスは $(Root) として定義するか、または未定義にすることができます。

パラメータ

ファイル内の構成パラメータの名前を指定します。 ワイルドカードまたはパターン マッチングはサポートされていません。ただし、変数置換は許可されています。

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］属性は未定義にできます。 その場合、［グループ パス］属性のみを指定します。

正規表現

指定した値が見つかった場合、それをフィルタする正規表現を指定します。

パーレン

正規表現が定義され、丸かっこで囲まれた部分表現が含まれている場合、一致したときに返される部分表現を指定します。

デフォルト： 0 （［パーレン］が指定されていない場合）

グループ パス

ファイル内の指定したパラメータに［グループ パス］の値を使用している階層的に構成された構成ファイルを指定します。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。

単一のファイル内で同じ名前が何回も使用されます。そのため、多くの一致する名前から 1 つの特定の値を識別するように、一意の［グループ パス］を指定する必要があります。

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>
```

［グループ パス］のエレメントは、名前のカンマ区切りリスト、または名前および値で修飾して、検索するパスの一致する子を識別することができます。

たとえば、サーバ wxp124 の buildDate の値を抽出するには、以下を指定します。

- ［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 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］の任意またはすべてのエレメントを修飾できます。また、各エレメントに対して複数の修飾子を指定できます。複数の修飾子を指定する場合、それぞれの修飾子はパスを選択できるように一致している必要があります。 変数置換は許可されています。

構造クラスに構成ファイルが割り当てられており、構造クラスでグループ ブロック修飾子を定義している場合、代替構文を使用できます。

グループ ブロックには、ブロック名に修飾子の値を丸かっこで囲んで追加できます。

たとえば、[修飾子の子]で属性名が指定されているサーバ wxp124 から buildDate の値を抽出するには、以下を指定します。

- ［パラメータ］ : buildDate
- ［グループ パス］ : configuration/server(wxp124)/setup

丸かっこ内の値は、指定されたグループ ブロックの修飾子に一致します。修飾子は、［修飾子の子］の値、構造クラスの定数の［修飾子］、またはグループの［値］のいずれかによって定義されます。

パラメータがファイル内のどこに存在するかが不明な場合は、複数のパスを指定できます。 2 つ以上の異なるパスをパイプ（|）で区切って指定します。 パスの確認は、左から右の順に、一致するまで 1 つずつ行われます。 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

指定された構成パラメータがファイル内に存在するが値が指定されていない場合、CA Configuration Automation はデフォルトでその値を空（ゼロ長）の文字列と見なします。［空の値を NULL と見なす］フィールドでは、このデフォルトの動作を上書きして、空の値を返すディレクティブと何の値も返さないディレクティブを区別できます。

- ［空の値を NULL と見なす］が［いいえ］に設定され、デフォルト値が［デフォルト］フィールドで定義されている場合、値が空のときはこの値は使用されません。
- ［空の値を NULL と見なす］が［はい］に設定され、デフォルト値が［デフォルト］フィールドで定義されている場合、値が空のときはこの値が使用されます。

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指定します。ディレクティブが実行され、値を特定できない場合、未定義またはゼロ長の値の代わりに［デフォルト］フィールドの値が使用されます。

コンポーネント ブループリント プラグインによってディレクティブの値を設定できる場合、その値をデフォルト値で置換する必要が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。 プラグインの値は、プレースホルダに設定されている場合、置換されません。

大文字と小文字の区別

値のテキストの大文字と小文字の違いを CA Configuration Automation が無視するかどうかを指定します。

- ［大文字と小文字を区別する］（デフォルト）は、CA Configuration Automation がすべての相違を識別およびレポートすることを意味します。
- ［大文字と小文字を区別しない］は、CA Configuration Automation がテキストの大文字と小文字の違いを無視することを意味します。

トランスレート

ディレクティブ出力にトランスレート処理を適用します。 トランスレート処理が一致しない場合、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、多くの場合、列挙型に属する暗号の文字列または整数です。各値は構成状態に対応しますが、値自体からは必要な解釈を明確に判断することはできません。 これらの値は、ブループリント内で表示する際に明確になるように、意味のある文字列に変換できます。

［トランスレート］フィールドで、一意のトランスレート処理名を指定します。 CA Configuration Automation は、指定された名前の前に文字列 $CCTranslation$__ を付加して、データベース内のトランスレート処理として識別可能にします。

その後、CA Configuration Automation データベースに値のマッピングを定義する必要があります。このマッピングはトランスレート処理名を参照し、*変換元の値*から*変換先の値*へのトランスレート処理をトリガします。

以下の例は、$CCTranslation$__IIS_SERVER_STATE という名前のトランスレート処理のマッピングを定義しています。

```
<row>
  <element_id>23501</element_id>
  <elem_name>$CCTranslation$__IIS_SERVER_STATE</elem_name>
  <elem_dtype>0</elem_dtype>
  <elem_value type="xml"><![CDATA[
    <IIS_SERVER_STATE>
      <Value from="2" to="Running"/>
      <Value from="4" to="Stopped"/>
      <Value from="6" to="Paused"/>
    </IIS_SERVER_STATE>
  ]]></elem_value>
  <elem_desc>Common Enumeration</elem_desc>
  <elem_state>1</elem_state>
  <created_by>1</created_by>
  <creation_time type="date">Jan 14, 2003</creation_time>
</row>
```

トランスレート処理を特定のコンポーネント ブループリントに関連付ける必要はありません。 トランスレート処理はすべてのコンポーネント ブループリントから名前で参照できるため、各トランスレート処理には一意の名前を付けて、ほかのトランスレート処理と競合しないようにする必要があります。

注: 現在、CA Configuration Automation Server ユーザ インターフェースでは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（例を参照）のファイルは、データ ローダ ユーティリティを使用してデータベースに作成およびロードする必要があります。

変換

ディレクティブ値が XML 形式で返された場合に、コンテンツをフィルタするために使用する XSL 変換を指定します。 変換された値は元の戻り値を置換し、新しい XML 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 内に存在する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

表示または以降の操作に必要な結果を生成するために、別の文字列内に挿入するディレクティブの値を指定します。

別の文字列内で $(VALUE) （すべて大文字）を含む文字列は、挿入場所として機能できます。 $(VALUE) は、フィルタされた結果を生成するためにディレクティブ値に置換されます。 たとえば、最初のディレクティブ値が 3 で、挿入文字列が 1.$(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 になります。

修飾子

次のいずれかを指定します。［ホストはエージェントを使用］または［Alterpoint デバイス］。

修飾子パラメータ

修飾子に適用されるパラメータを指定します。

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。

4. ［構成データ］リンクをクリックし、以下のフィールドに入力します。

データベース

ブループリントによって使用されるデータベースを指定します。 ドロップダウン リストには、手順 22 で作成したデータベースが表示されます。

［保存］をクリックします。データベースが左側のペインの［構成データ］ツリーに表示されます。

5. ［クエリの追加］ペインをクリックし、以下のフィールドに入力します。

名前

管理対象サーバ上のデータベースに対してクエリを実行し、オプションで出力をフィルタして、取得したデータから値を抽出するディレクティブを指定します。

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。

- データベース内の行から列の値を抽出するためのクエリを実行する。
- 管理対象サーバでの変更または値の抽出を行うためのストアド プロシージャを実行する。
- 結果セットを抽出してディレクティブのマクロ ステップに表形式で表示するためのクエリを実行する。

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

実行可能ディレクティブの目的およびソースについて説明するオプションのテキストです。

この説明は、ブループリント内の実行可能ディレクティブ名、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表示されます。

クエリ タイプ

ドロップダウン リストから、以下のいずれかのクエリ タイプを指定します。

- 選択 -- このクエリは SQL の select ステートメントに該当します。実行するとデータが返されます。
- 挿入 -- このクエリは SQL の insert ステートメントに該当します。データは返されません。
- 削除 -- このクエリは SQL の delete ステートメントに該当します。データは返されません。
- 更新 -- このクエリは SQL の update ステートメントに該当します。データは返されません。

- その他 -- このクエリは SQL の alter table ステートメントまたはその他の DDL ステートメントに該当します。データは返されません。
- ストアドプロシージャ -- このクエリ文字列はストアドプロシージャを呼び出します。データは返される場合もあれば、返されない場合もあります。

選択したクエリ タイプによって、CA Configuration Automation がクエリを実行する方法とデータが返されるかどうかが決定されます。

クエリ

有効な SQL クエリまたはストアドプロシージャ名（構文はデータベース タイプによって異なります）を指定します。 変数置換は許可されています。

クエリまたはストアドプロシージャによって選択された行がすべて返されます。 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

クエリによって取得される行の最大数を指定します。

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指定します。指定した列の値は、ディレクティブの値として使用されます。クエリによって複数の行が返される場合、指定した列の最初の行が使用されます。

列は実行可能ディレクティブに必要です。

構造クラス

パーサによってファイルから取得された名前と値のペアの特定のセットにメタデータをマップするオーバーレイとして使用する構造クラス（グループとパラメータの階層コレクション）を指定します。

ノート（編集モードのみ）

選択したエレメントに関する注意事項を定義します。 エレメントに関する注意事項の数、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、すべての注意事項の表示、新しい注意事項の追加、または既存の注意事項の表示、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、新しいページが開き、すべての注意事項のリストが表形式で表示されます。 テーブルの並べ替え順序を変更するには、テーブルのいずれかの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 2 つの追加フィールドが表示されます。

- 名前 -- 注意事項の名前を入力します。
- テキスト -- 注意事項のテキストを入力します。

既存の注意事項を選択すると、注意事項の名前が［ノート］フィールドに表示され、注意事項のテキストが名前の下に表示されます。選択した注意事項は、更新、削除、またはキャンセルできます。

［保存］をクリックします。 クエリが左側のペインの［構成データ］ツリーに表示されます。

6. ［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］タブの以下のフィールドに入力します。

名前

構造クラスの名前を指定します。 名前は［ブループリント］リストに表示されます。［表示名］を指定していない場合は、ツリービューにも表示されます。

バージョン

構造クラスのバージョンを指定します。 バージョンは［ブループリント］リストに表示されます。［表示名］を指定していない場合は、ツリー ビューにも表示されます。

表示名

［コンポーネント ブループリント］ツリー ビューに表示するクラスの名前を指定します。この名前は、［名前］および［バージョン］フィールドで指定した値と異なる場合に表示されます。 たとえば、クラスの機能をより正確に表すテキストを指定できます。

説明

（オプション）クラスの説明。 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。

修復ジョブの許可

修復ジョブで指定されたブループリントの変更可能なエレメントに対する変更を許可す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を解析するために使用するパーサを指定します。 CA Configuration Automation には、最も一般的な構成ファイル形式の解析を行うための事前定義済みのレクサおよびパーサのライブラリが含まれています。

［保存］をクリックします。 構造クラスが左側のペインの［ファイル構造クラス］ツリーに表示されます。

7. ［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、以下のフィールドに入力します

名前

グループの名前を指定します。

説明

グループの目的を説明します。

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

［低］、［中］、［高］のいずれかのウェイトを指定します。

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パラメータまたはグループの有効な値を指定します（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。 ディレクティブが実行され、値を特定できない場合、未定義またはゼロ長の値の代わりにデフォルト値が使用されます。

コンポーネント ブループリント プラグインによってディレクティブの値を設定できる場合、その値をデフォルト値で置換する必要が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。 プラグインの値は、プレースホルダに設定されている場合、置換されません。

表示/非表示

値を表示するかどうかを指定します。

- ［値を表示］は、パラメータ値が通常どおり表示されることを意味します。 デフォルトは［値を表示］です。
- ［値を非表示］は、パラメータ値が表示されず、代わりに ******* として表示されることを意味します。パラメータ値が機密情報（パスワードなど）の場合、またはパラメータ値がバイナリや長すぎる値のため表示できない場合、パラメータを非表示にします。非表示のパラメータはデータベース内で暗号化され、UI では表示されないか、または参照できません。
- ［エレメントを非表示］により、パラメータを非表示にすることができます。エレメントの値を変数置換に使用する場合でも、サービス ツリー ビューにエレメントを表示しないときは、エレメントを非表示にします。

大文字と小文字の区別

値のテキストの大文字と小文字の違いを CA Configuration Automation が無視するかどうかを指定します。 ［大文字と小文字を区別する］（デフォルト）は、CA Configuration Automation がすべての相違を識別およびレポートすることを意味します。 ［大文字と小文字を区別しない］は、テキストの大文字と小文字の違いが無視されることを意味します。

解釈方法

構成パラメータの文字列形式および関連付けられたコンポーネントでのその使用目的に関するヒントを提供します。 解釈されたパラメータ値は、CA Configuration Automation によってコンテキスト依存パーサで検査されます。コンテキスト依存パーサでは、複雑なパラメータ文字列内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

このフィールドが[はい]に設定されている場合に、CA Configuration Automation が［解釈方法］フィールドを使用して関係を決定して割り当てるように指定します。 すべての解釈が関係を定義できるとは限りません。

デフォルト： ×

重要: 関係を確立するために、［解釈方法］フィールドで適切な値を定義し、［関係］フィールドを［はい］に設定してください。

関係タイプ

パラメータの関係のタイプ（管理、ホスト、通信、使用するなど）を指定します。

解釈済みのサーバ名

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 インスタンスを指定します。

解釈済みのターゲット インスタンス

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 インスタンスを指定します。

解釈済みのアプリケーション名

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。

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指定します。

値

ファイル内のグループの一意の値を指定します。

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するパラメータを指定します。

8. ［次へ］をクリックします。

［マクロ］ページが表示されます。

9. 以下のフィールドに適切な情報を入力し、[次へ]をクリックします。

名前

マクロの名前（マクロは、それぞれが値およびステータスを返す操作のシーケンスです）を指定します。

説明

マクロの目的を説明します。

フォルダ

マクロが配置されているフォルダを指定します。

診断

マクロを診断目的で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。

デフォルト： いいえ（ターゲット システムを変更できません）。

読み取り専用マクロを実行するには、ユーザは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。

［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。このページはネスト オプションで構成されているため、あるコンポーネントを別のコンポーネント内に埋め込むことにより、サービス内のコンポーネント間の関係を強調して表示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 コンポーネントに JRE および Apache をネストすることにより表現されます。

10. 以下のフィールドに適切な情報を入力し、[完了]をクリックします。

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

この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。 このチェック ボックスがオフの場合、ブループリントは他のコンポーネントを持ちません。

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

このブループリントが他のコンポーネントにネストできるかどうかを指定します。 このチェック ボックスがオフの場合、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も、他のコンポーネントがブループリントにネストすることもありません。

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

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

この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

指定された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定義します。 リフレッシュ中に、各サーバのコンポーネントは順番にリフレッシュされます（他のサーバのコンポーネントは並列にリフレッシュされます）。

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、あるコンポーネントが別のコンポーネントからのパラメータの値を変数置換に使用する場合、この別のコンポーネントが最初にリフレッシュされることが重要です。 このような場合に［リフレッシュ順序］を指定しないと、依存するコンポーネントは、最新のリフレッシュによってまだ更新されていない古い値を取得する可能性があります。

以下に示す「初期化」から［クリーンアップ］までの 7 つのレベルを選択できます（指定されたレベル内では、特定の順序は保証されません。［考慮しない］を選択した場合、コンポーネントは［中期］の順序が設定されたように処理されます）。

- 考慮しない。プレファレンスはなく、他のコンポーネントが存在する場合は［中期］として処理されます。
- 初期化。 最初にリフレッシュされるグループ。
- 最初
- 初期
- 中期
- 後期
- 最後
- クリーンアップ。 最後にリフレッシュされるグループ。

修飾子

以下のいずれかを指定します。

- 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

修飾子に適用されるパラメータを指定します。

最近隣

コンポーネントが他の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ように指定します。

ネスト先

この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 システム関係を定義します。［ネスト先］ドロップダウン リストから以下のいずれかの値を選択します（［このブループリントはネストを使用しません］チェック ボックスをオンにした場合、この値は必要ありません）。

- 子 - ファイル システム ディレクトリ ツリー内でこのコンポーネントより上位のファイル システム ルートを持つ他のコンポーネントに、このコンポーネントがネストするように指定します。
- 子または兄弟 - ファイル システム ディレクトリ ツリー内でこの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、このコンポーネントがネストするように指定します。
- 直接の子 - ファイル システム ディレクトリ ツリー内でこの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、このコンポーネントがネストできるように指定します。
- 兄弟 - 同じファイル システム ルートを持つ他のコンポーネントに、このコンポーネントがネストするように指定します。

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指定します。［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。 このオプションは、一致する可能性のあるいくつかのネストから 1 つを選択するため、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ために使用できます。

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。 このオプションは、一致する可能性のあるいくつかのネストから 1 つを選択するため、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数にネストを制限するために使用できます。

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。

第 4 章: ブループリント エレメント 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 エレメント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

［カテゴリ］フィールドでは、コンポーネント ブループリントのエレメントを整理できます。

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 Web 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エレメント。たとえば、マニュアル、readme 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係するエレメント。

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、SNMP 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 LDAP 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、CPU 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、SSL、または HTTPS 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の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エレメントの値を、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エレメント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エレメントの値が抽出され、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エレメントには、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 エレメント（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 管理対象エレメント 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アントは、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アントを使用する方法

```
${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[s
ize]]]]}
```

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[*,NVFil
e
[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、jrmi、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>;databasename=<database>; SendStringParametersAsUnicode=false
Oracle 9、10、および 11	■ jdbc:oracle:thin:@$(host):$(port):$(database) ■ jdbc:oracle:thin:@<host>:<port>:<database> ■ jdbc:oracle:thin:@<host>:<port>/<database> ■ jdbc:oracle:thin:@((DESCRIPTION=(ADDRESS_LIST=(ADDRESS=(PROTOCOL=TCP)(HOST=<host1>)(PORT=<port1>)(ADDRESS=(PROTOCOL=TCP)(HOST=<host2>)(PORT=<port2>))(FAILOVER=ON)(LOAD_BALANCE=OFF)(CONNECT_DATA=(SERVER=DEDICATED)(SERVICE_NAME=<database>)) ■ jdbc:oracle:thin:@((DESCRIPTION=(ADDRESS_LIST=(ADDRESS=(PROTOCOL=TCP)(HOST=<host>)(PORT=<port>)))(CONNECT_DATA=(SERVICE_NAME=<database>))) ■ jdbc:oracle:thin:@(DESCRIPTION=(ADDRESS_LIST=(ADDRESS=(PROTOCOL=TCP)(HOST=<host>)(PORT=<port>)))(CONNECT_DATA=(SERVICE NAME=<database>))) ■ jdbc:bea:oracle://<host>:<port>
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	■ jdbc:mysql://<host>:<port>/<database> ■ jdbc:mysql://<host>:<port>
Postgres	jdbc:postgresql://<host>:<port>/<database>
HSQLDB	jdbc:hsqldb:hsql://<host>:<port>
ODBC	jdbc:odbc:<database>
Cloudscape	jdbc:cloudscape:<database>
Java DB （Derby）	■ jdbc:derby://<host>:<port>/<database> ■ jdbc:derby://<host>:<port>/<database>;create=true
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）。

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。
文字クラス	
[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）。
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。
標準以外の POSIX スタイル文字クラス	
[: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。
事前定義済みクラス	
.（ピリオド）	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。
境界は一致します。	
^（キャレット）	文字列の先頭のみ一致します。
$（ドル記号）	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。
最長一致閉包 （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。
non-greedy な繰り返し （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戻り、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エレメントと一致させます。 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
```

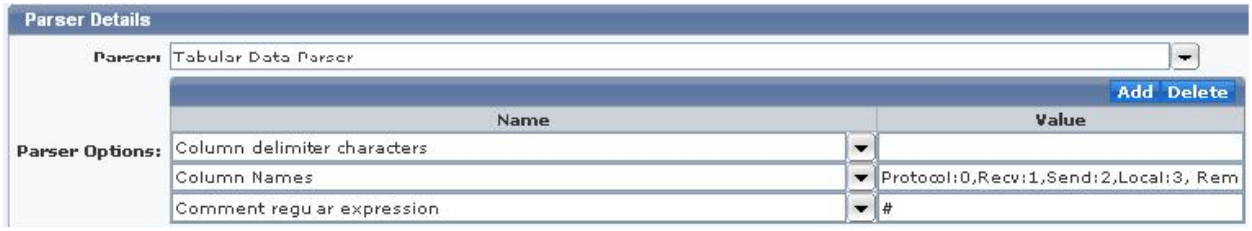
CA Configuration Automation は、構成ファイルまたは実行可能ファイル内にあるすべての形式の表形式データを解釈および解析する表形式データ パーサを提供します。また、表形式データ セットの行と列のレイアウト、列名の割り当て、ヘッダおよびコメント テキストの削除を制御するパーサ オプションを指定できます。同様にデータ階層の整理もできます。

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 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 セットの最初に無視す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,Rem
ote: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	Fred@acme.com	12Jones Cl.
Arne	3002	Arne@acme.com	33535 Eucalyptus Terrace
Mary	7331	Mary@acme.com	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。