カスタム Oracle® Solaris 11 インストールイメージの作成

Part No: E25771
2011 年 11 月 E25771-01

120411@25097

目次

はじめに

カスタム Oracle Solaris 11 インストールイメージの作成では、カスタムの Oracle Solaris インストールイメージを構築するための Oracle Solaris ディストリビューションコンストラクタ (DC) ツールを使用する手順について説明します。

関連情報

『Oracle Solaris 11 システムのインストール』では、次の方法のいずれかを使用して Oracle Solaris オペレーティングシステム (OS) をインストールして構成する手順について説明します。

- LiveCD イメージ
- 対話型テキストインストーラ
- Oracle Solaris 自動インストーラ (AI) 機能
- Oracle Solaris SCI ツールの対話型システム構成ツール
- `sysconfig`(1M) コマンド行のシステム構成ツール

『Oracle Solaris 11 ブート環境の作成と管理』では、Oracle Solaris システムで複数のブート環境 (非大域ゾーンを含む) を管理する方法について説明します。

『Oracle Solaris の管理: 一般的なタスク』の第 6 章「サービスの管理 (概要)」では、Oracle Solaris サービス管理機能 (SMF) について説明します。SMF プロファイルを使用して、システムを構成できます。

`pkg`(5) のマニュアルページでは、Oracle Solaris Image Packaging System (IPS) 機能について説明します。IPS を使用すると、インストールするソフトウェアパッケージを格納および取得できます。`pkg`(1) のマニュアルページでは、IPS パッケージをインストールする方法について説明します。

Oracle Solaris 11 システムを管理する方法の詳細は、Oracle Solaris 11 システム管理ドキュメントを参照してください。

『Oracle Solaris 10 JumpStart から Oracle Solaris 11 自動インストーラへの移行』では、JumpStart から AI に移行する際に役立つ情報を提供します。どちらも、Oracle Solaris の自動インストール機能です。

Oracle サポートへのアクセス

Oracle のお客様は、My Oracle Support を通じて電子的なサポートを利用することができます。詳細は、http://www.oracle.com/pls/topic/lookup?ctx=acc&id=info を参照してください。聴覚に障害をお持ちの場合は、http://www.oracle.com/pls/topic/lookup?ctx=acc&id=trs を参照してください。

表記上の規則

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

表 P–1 表記上の規則

字体または記号	意味	例
AaBbCc123	コマンド名、ファイル名、ディレクトリ名、画面上のコンピュータ出力、コード例を示します。	.login ファイルを編集します。 ls -a を使用してすべてのファイルを表示します。 system%
AaBbCc123	ユーザーが入力する文字を、画面上のコンピュータ出力と区別して示します。	system% **su** password:
AaBbCc123	変数を示します。実際に使用する特定の名前または値で置き換えます。	ファイルを削除するには、rm *filename* と入力します。
『 』	参照する書名を示します。	『コードマネージャ・ユーザーズガイド』を参照してください。
「 」	参照する章、節、ボタンやメニュー名、強調する単語を示します。	第 5 章「衝突の回避」を参照してください。 この操作ができるのは、「スーパーユーザー」だけです。
\	枠で囲まれたコード例で、テキストがページ行幅を超える場合に、継続を示します。	sun% **grep '^#define ** **XV_VERSION_STRING'**

Oracle Solaris OS に含まれるシェルで使用する、UNIX のデフォルトのシステムプロンプトとスーパーユーザープロンプトを次に示します。コマンド例に示されるデフォルトのシステムプロンプトは、Oracle Solaris のリリースによって異なります。

- C シェル

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

- C シェルのスーパーユーザー

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

- Bash シェル、Korn シェル、および Bourne シェル

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

- Bash シェル、Korn シェル、および Bourne シェルのスーパーユーザー

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

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

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

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

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

◆ ◆ ◆ 第 1 章

1 カスタムインストールイメージの作成入門

システム管理者およびアプリケーション開発者は、ディストリビューションコンストラクタツールを使用して、カスタムの Oracle Solaris インストールイメージを構築できます。

- これまでにカスタムインストールイメージを作成したことがない場合は、9 ページの「ディストリビューションコンストラクタとは」からお読みください。
- カスタムイメージをすぐに構築できる場合は、13 ページの「イメージを構築するためのシステム要件」に進んでください。

ディストリビューションコンストラクタとは

ディストリビューションコンストラクタは、事前構成済みの Oracle Solaris イメージを作成するためのコマンド行ツールです。このツールは XML マニフェストファイルを入力として受け取り、マニフェストファイルに指定されているパラメータに基づいて、イメージを構築します。

ディストリビューションコンストラクタは、ISO イメージを構築できます。ISO イメージは、International Organization for Standardization (ISO) によって定義された形式の光学式ディスクのアーカイブファイルで、ディスクイメージとも呼ばれます。また、生成された ISO イメージに基づいて、USB イメージを作成することもできます。ただし、USB イメージは ISO イメージとは異なり、x86 システムでのみ作成および使用できます。

次の事項に注意してください。

- イメージの構成に応じて、ISO イメージや USB イメージを起動可能にすることもできます。
- ISO イメージと USB イメージは、どちらもシステムにインストールしたり、ライブメディア環境で実行したりできます。
- ISO イメージは、CD または DVD に書き込むことができます。

- USB イメージは、フラッシュドライブにコピーできます。
- ISO イメージおよび USB イメージは、どちらもインターネットで配布できます。

ディストリビューションコンストラクタでは、さまざまな種類のフラッシュメモリーデバイスで動作する USB イメージを作成できます。ただし、Oracle Solaris リリースでドライバがサポートされているデバイスに限ります。USB イメージを USB フラッシュドライブにコピーするには、`usbcopy` ユーティリティーを使用する必要があります。この `usbcopy` ユーティリティーは、`distribution-constructor` パッケージに付属しています。

Oracle Solaris イメージの種類

ディストリビューションコンストラクタでは、次の種類の Oracle Solaris イメージを作成できます。

- **Oracle Solaris x86 LiveCD** – Oracle Solaris リリースとして配布される LiveCD イメージと互換性がある x86 ISO イメージを作成できます。また、パッケージを追加または削除することにより、この ISO イメージの内容を変更できます。作成されたブート環境のデフォルト設定を変更して、カスタムの ISO イメージまたは USB イメージを作成することもできます。

  注 – LiveCD イメージに含まれるパッケージのサイズに応じて、LiveCD イメージは代わりに LiveDVD になる場合があります。

  LiveCD インストールの詳細は、『Oracle Solaris 11 システムのインストール』の第 3 章「LiveCD の使用」を参照してください。
- **Oracle Solaris x86** または **SPARC** テキストインストールイメージ – Oracle Solaris オペレーティングシステムのテキストインストールを実行する際に使用できる SPARC または x86 ISO イメージを作成できます。テキストインストーラは、グラフィックカードを必要としないシステムで使用できます。

  注 – テキストインストールでは、LiveCD からインストールするときに含まれるすべてのソフトウェアパッケージがインストールされるわけではありません。たとえば、テキストインストーラではデスクトップはインストールされません。テキストインストールが完了したら、`solaris-desktop` パッケージなどの追加パッケージを追加できます。

  テキストインストールの詳細は、『Oracle Solaris 11 システムのインストール』の第 4 章「テキストインストーラの使用」を参照してください。

- 自動インストール用の **x86** または **SPARC ISO** イメージ – Oracle Solaris リリースには自動インストーラツールが含まれています。自動インストーラ (AI) は、1 つ以上の SPARC および x86 システムにネットワーク経由で Oracle Solaris OS を自動インストールするために使用します。アーキテクチャー、インストールするパッケージ、ディスク容量、およびその他のパラメータは、インストールごとに異なる場合があります。自動インストーラは、SPARC または x86 AI ISO イメージを使用して Oracle Solaris OS をクライアントシステムにインストールします。ディストリビューションコンストラクタを使用して、Oracle Solaris OS を SPARC クライアントにインストールするための SPARC AI ISO イメージ、または Oracle Solaris OS を x86 クライアントにインストールするための x86 AI ISO イメージを作成できます。

  自動インストーラの使用の詳細は、『Oracle Solaris 11 システムのインストール』のパート III「インストールサーバーを使用したインストール」を参照してください。

イメージ作成のプロセス

ディストリビューションコンストラクタは、マニフェストファイルと呼ばれる XML ファイルに指定された設定に基づいてイメージを作成します。マニフェストファイルには、ディストリビューションコンストラクタを使用して作成する ISO イメージの内容およびパラメータの仕様が記述されています。ディストリビューションコンストラクタには、カスタム LiveCD、x86 または SPARC AI ISO イメージ、x86 または SPARC テキストインストールイメージを作成する際に使用できるサンプルマニフェストが含まれます。

各マニフェストファイルのすべてのフィールドには、必要とする種類のイメージを作成するためのデフォルト値が事前設定されています。マニフェストファイル内のフィールドを編集することで、結果として生成されるイメージをさらにカスタマイズできます。たとえば、マニフェストのターゲット要素を編集すれば、イメージを構築できる構築領域に別の場所を指定できます。さらに、指定されたパブリッシャーをチェックして、イメージの構築に必要なパッケージをダウンロードすることについて、使用中のシステムがそのパブリッシャーに連絡できることを確認します。必要に応じて、ソフトウェア名要素を編集して、別のパブリッシャーおよびリポジトリの場所を指定できます。手順については、14 ページの「イメージのカスタマイズ」を参照してください。

カスタムスクリプトを作成して、インストールイメージを変更することもできます。その後、マニフェストファイルにチェックポイントを追加して、これらのカスタムスクリプトを実行できます。詳細は、23 ページの「カスタムスクリプトの作成と使用」を参照してください。

ディストリビューションコンストラクタパッケージには、マニフェストの仕様を解釈して、イメージを構築するためのコマンド行ユーティリティーである `distro_const` コマンドも含まれています。要件に合わせてマニフェストファイルで

イメージの青写真を編集する作業が完了した後に、distro_const コマンドを実行してイメージを作成します。詳細は、第 3 章「イメージの構築」を参照してください。

distro_const コマンドには、構築中のイメージのチェックおよびデバッグを行うために、イメージ生成プロセスのさまざまな段階で構築プロセスを停止および再開するためのオプションが用意されています。構築プロセス中に停止および再開するこのプロセスをチェックポイント処理と呼びます。チェックポイント処理は省略可能です。デフォルトのチェックポイントは、各マニフェストファイルに指定されます。手順については、29 ページの「段階的にイメージを構築する方法」、または distro_const(1M) のマニュアルページを参照してください。

SPARC と x86 のアーカイブの違い

x86 イメージのルートアーカイブと SPARC イメージのルートアーカイブは異なります。x86 イメージのルート全体のアーカイブである boot_archive は、UFS ファイルシステムであり、lzma を使用して圧縮されます。SPARC プラットフォームでは、ルート全体のアーカイブの圧縮はサポートされません。その代わりに、SPARC のルートアーカイブでは DCFS が使用され、各ファイルが個別に圧縮されます。個別に圧縮されたファイルは、マニフェストで特定の処理が必要になる場合があります。手順については、dc_manifest(4) のマニュアルページの <boot_archive_contents> フィールドを参照してください。

◆ ◆ ◆ 第 2 章

2 カスタムインストールイメージの設計

この章で説明するように、システム要件をレビューし、カスタムインストールイメージを設計します。

イメージを構築するためのシステム要件

ディストリビューションコンストラクタを使用するには、システムが次の要件を満たしている必要があります。

表 2–1 システム要件

要件	説明
ディスク容量	ディストリビューションコンストラクタのワークスペースの推奨最小サイズは 8G バイトです。システムにディストリビューションコンストラクタを使用するための十分なディスク容量があることを確認してください。
Oracle Solaris リリース	Oracle Solaris オペレーティングシステム (OS) がシステムにインストールされている必要があります。次の点に注意してください。 ■ インストールされている システムはネットワークにアクセスできる必要があります。ディストリビューションコンストラクタは、ネットワーク経由で利用できる Image Packaging System (IPS) リポジトリにアクセスして、ISO イメージのパッケージを取得します。マニフェストファイル内に指定されたリポジトリへのネットワークアクセスが可能である必要があります。 ■ ディストリビューションコンストラクタを使用している場合は、SPARC システムには SPARC イメージのみ、 x86 システムには x86 イメージのみを作成できます。 ■ Oracle Solaris のリリースバージョンは、ディストリビューションコンストラクタで使用する イメージのリリースバージョンと同じである必要があります。 注 – ディストリビューションコンストラクタを実行するには、root 役割になる必要があります。

表 2–1 システム要件 (続き)

要件	説明
必要なパッケージ	ディストリビューションコンストラクタアプリケーションが含まれる `distribution-constructor` パッケージ。

イメージのカスタマイズ

ディストリビューションコンストラクタは、マニフェストファイルと呼ばれる XML ファイルに指定された設定に基づいてイメージを作成します。マニフェストファイルには、ディストリビューションコンストラクタを使用して作成する ISO イメージの内容およびパラメータの仕様が記述されています。`distribution-constructor` パッケージは、カスタム LiveCD、x86 または SPARC AI ISO イメージ、x86 または SPARC テキストインストールイメージを作成する際に使用できるサンプルマニフェストを提供します。

各マニフェストファイルの要素には、必要とする種類の ISO イメージを作成するためのデフォルト値が事前設定されています。マニフェストファイル内の事前設定要素を手動で編集することで、結果として生成されるイメージをカスタマイズできます。また、カスタムスクリプトを作成すれば、さらにイメージを変更できます。次に、マニフェストファイルで新しいスクリプトを参照してください。

サンプルマニフェスト

`distribution-constructor` パッケージは、次のサンプルマニフェストファイルを提供します。

表 2–2 サンプルマニフェスト

マニフェストの種類	マニフェストの場所	説明
x86 LiveCD ISO イメージ	`/usr/share/distro_const/dc_livecd.xml`	Oracle Solaris LiveCD と互換性がある ISO イメージを作成する場合に使用されます。
x86 テキストインストールイメージ	`/usr/share/distro_const/dc_text_x86.xml`	x86 Oracle Solaris オペレーティングシステムのテキストインストールを実行する際に使用可能な ISO イメージを作成する場合に使用されます。
SPARC テキストインストールイメージ	`/usr/share/distro_const/dc_text_sparc.xml`	SPARC Oracle Solaris オペレーティングシステムのテキストインストールを実行する際に使用可能な ISO イメージを作成する場合に使用されます。

表 2–2 サンプルマニフェスト (続き)

マニフェストの種類	マニフェストの場所	説明
x86 AI ISO イメージ	/usr/share/distro_const/dc_ai_x86.xml	Oracle Solaris OS を x86 クライアントに自動インストールするための x86 AI ISO イメージを作成する場合に使用されます。
SPARC AI ISO イメージ	/usr/share/distro_const/dc_ai_sparc.xml	Oracle Solaris OS を SPARC クライアントに自動インストールするための SPARC AI ISO イメージを作成する場合に使用されます。

▼ カスタムイメージの作成方法と構築方法

1 ディストリビューションコンストラクタアプリケーションおよびサンプルマニフェストが含まれる **distribution-constructor** パッケージをダウンロードします。

パッケージマネージャーツールを使用して、必要なパッケージをインストールできます。パッケージマネージャーは、Oracle Solaris オペレーティングシステムのデスクトップのメニューバーから利用できます。メニューバーで、「システム」>「システム管理」>「パッケージマネージャー」の順に選択します。

または、次のような IPS コマンドを使用してこのパッケージをインストールしてください。

```
# pkg install distribution-constructor
```

2 サンプルマニフェストをコピーし、新しいファイル名を付けてカスタムのマニフェストファイルを作成します。

`distro_const` コマンドを使用してイメージを作成するときに、このマニフェストファイルの名前を指定します。

注 – 元のマニフェストファイルおよびデフォルトのスクリプトをコピーするときは、その前に必ずバックアップを作成してください。

---

3 必要に応じて、マニフェスト要素を編集します。

たとえば、マニフェストのターゲット要素を編集すれば、イメージを構築できる構築領域に別の場所を指定できます。さらに、パブリッシャーをチェックして、イメージの構築に必要なパッケージをダウンロードすることについて、システムがそのパブリッシャーに連絡できることを確認します。必要に応じて、ソフトウェア名要素を編集して、別のパブリッシャーおよびリポジトリの場所を指定できます。

詳細は、16 ページの「マニフェストの内容の変更」および `dc_manifest(4)` のマニュアルページを参照してください。

4 (省略可能)イメージをさらに変更するには、カスタムスクリプトを作成します。
新しいスクリプトを作成する場合は、マニフェストファイル内の実行セクションのスクリプト参照を更新します。

詳細は、23 ページの「カスタムスクリプトの作成と使用」を参照してください。

5 **distro_const** ユーティリティーを実行して、イメージを作成します。
手順については、第 3 章「イメージの構築」を参照してください。

マニフェストの内容の変更

各マニフェストファイルのすべてのフィールドには、必要とする種類の ISO イメージを作成するためのデフォルト値が事前設定されています。マニフェストファイル内の事前設定フィールドを手動で編集することで、結果として生成されるイメージをさらにカスタマイズできます。

選択したサンプルマニフェストに応じて、主要な要素は次のようになります。

表 2–3 マニフェスト要素

要素	説明
`<distro name="Oracle_Solaris_Text_X86" add_timestamp="false">`	省略可能なタイムスタンプ付きでイメージ名を指定します。
`<boot_mods>`	イメージ用の GRUB メニュー変更を指定します。
`<target>`	イメージが構築される ZFS 構築データセットを定義します。
`<software name="transfer-ips-install" type="IPS">`	インストールするソフトウェアパッケージのソースを指定します。
`<software_data action="install">`	インストールするパッケージを一覧表示します。
`<software_data action="uninstall">`	アンインストールするパッケージを一覧表示します。
`<software name="set-ips-attributes">`	インストールが完了したあとに、IPS の各種属性を設定します。
`<software name="ba-init">`	ブートアーカイブの内容を指定します。 注意 – 注意して変更してください。ブートアーカイブが正しくない場合は、インストールされたシステムのブートに失敗します。
`<execution stop_on_error="true">` `<checkpoint name="transfer-ips-install"/>`	構築チェックポイントを一覧表示します。

表 2–3 マニフェスト要素 (続き)

要素	説明
<configuration name="pre-pkg-img-mod" type="sysconf" source="/etc/svc/profile/generic_limited_net.xml">	構築中にメディアに適用される SMF サービスを指定します。 注意 – なるべく変更しないでください。

イメージタイトルの指定

次の要素を使用して、構築するイメージのカスタム名またはデフォルト名を指定します。

```
<distro name="Oracle_Solaris_Text_X86" add_timestamp="false">
```

イメージの構築作業を続けて実行して複数の増分イメージを保持する場合、タイムスタンプ変数を「true」に変更すると、タイムスタンプが各イメージの名前に自動的に追加されます。

HTTP プロキシを指定する必要がある場合、プロキシ変数を含む配布名要素のコメントを解除して、プロキシの場所を入力します。

ブートメニューの変更

このブートメニューは、イメージに適用されるブートメニュー変更を指定します。

次の例では、「boot1」というタイトルの特殊なブートメニューがイメージに適用されます。タイムアウト属性は、デフォルトのブートエントリが自動的に有効にされるまでの時間を指定します。

```
<boot_mods title="boot1" timeout="5">
```

新しいエントリごとに boot_entry 要素を追加することで、ブートメニューエントリを個別に追加できます。エントリは、各ブートエントリの「start」または「end」の insert_at 属性値に基づいた順序でブートメニューに順次追加されます。

注 – 新しいエントリは、既存の「with magnifier」エントリの前に追加します。

個々の boot_entry 要素については、次の例を参照してください。

```
<boot_entry>
   <title_suffix>with screen reader</title_suffix>
   <kernel_args>-B assistive_tech=reader</kernel_args>
</boot_entry>
```

詳細は、dc_manifest(4) のマニュアルページを参照してください。

構築領域の指定

target 要素はカスタマイズできます。この要素は、構築に使用する ZFS 構築データセットを定義します。このデータセットは、イメージが作成される場所です。有効なデータセットの場所を入力する必要があります。システムで保持する必要がある内容が構築によって破棄されないように、デフォルトの構築領域をチェックする必要があります。必要に応じて、構築領域を変更します。

注 – ファイルシステム名には、zpool という名前を含めないでください。

次の例を参照してください。

```
<target>
  <logical>
    <zpool action="use_existing" name="rpool">
      <dataset>
        <filesystem name="dc/sample-dataset-location"
        action="preserve"/>
      </dataset>
    </zpool>
  </logical>
</target>
```

パブリッシャーの指定

次の要素には、イメージ構築のためにダウンロードおよび使用するパッケージをディストリビューションコンストラクタが取得できるパブリッシャーを指定します。

```
<software name="transfer-ips-install">
```

このセクションのソース要素で、パブリッシャー名要素と起点名要素を編集し、使用するパブリッシャーとパッケージリポジトリが存在する場所を指定します。複数のパブリッシャーを一覧表示できます。ディストリビューションコンストラクタがインストールするパッケージの検出を試みると、ここに一覧表示されている順序でパブリッシャーが検索されます。

パブリッシャーのミラーを指定する必要がある場合は、ミラー名要素をコメント解除して編集します。

次の例を参照してください。

```
<source>
   <publisher name="publisher1">
      <origin name="http://example.oracle.com/primary-pub"/>
      <mirror name="mirror.example.com"/>
   </publisher>
   <publisher name="publisher2">
       <origin name="http://example2.com/dev/solaris"></origin>
```

```
   </publisher>
   <publisher name="publisher3.org">
       <origin name="http://example3.com/dev"></origin>
    </publisher>
</source>
```

パブリッシャーの使用の詳細は、『Oracle Solaris 11 ソフトウェアパッケージの追加および更新』を参照してください。

インストールするパッケージの一覧表示

install 属性を持つ software_data 要素は、使用しているマニフェストに応じて、特定の種類のイメージを構築するためにインストールされるパッケージのセットを一覧表示します。たとえば、dc_livecd.xml マニフェストは、Live CD イメージの構築に必要なパッケージを一覧表示します。各名前タグは、1 つのパッケージの名前、または多数のパッケージを含む 1 つのグループパッケージの名前を一覧表示します。

```
<software_data action="install">
   <name>pkg:/group/system/solaris-desktop</name>
   <name>pkg:/system/install/gui-install</name>
   <name>pkg:/system/install/media/internal</name>
</software_data>
```

イメージに追加するパッケージがある場合、パッケージごとに名前タグを追加することによってパッケージ名を追加します。

デフォルトでは、指定されたリポジトリで利用できる最新のパッケージバージョンがインストールされます。他のバージョンが必要な場合、次の形式を使用してパッケージ参照にバージョン番号を追加します。

```
<name>pkg:/group/system/solaris-desktop@0.5.11-0.build#</name>
```

注 – 競合するバージョンがインストールされている他のパッケージが存在する場合は、特定のバージョンが指定されたパッケージがインストールされない可能性があります。詳細は、pkg(5) のマニュアルページを参照してください。

例 2–1　パッケージおよび追加パブリッシャーの追加

この例では、mypublisher という 2 番目のパブリッシャーが指定されます。さらに、mypackage1 および mypackage2 という追加パッケージも指定されます。

構築プロセス中に、パブリッシャーは一覧表示される順序でチェックされます。1 番目のパブリッシャーでパッケージが見つからない場合は、次のパブリッシャーで指定されたパッケージが検索されます。

```
<software name="transfer-ips-install" type="IPS">
   <destination>
      <xi:include xmlns:xi="http://www.w3.org/2003/XInclude"
            href="/usr/share/distro_const/lang_facets.xml"/>
```

例 2–1 パッケージおよび追加パブリッシャーの追加 *(続き)*

```
   </destination>
   <source>
      <publisher name="solaris">
          <origin name="http://pkg.oracle.com/solaris/release"/>
      </publisher>
      <publisher name="mypublisher">
         <origin name="http://mypublisher.company.com"/>
      </publisher>
   </source>
   <software_data action="install">
        <name>pkg:/group/system/solaris-large-server</name>
        <name>pkg:/system/install/text-install</name>
        <name>pkg:/system/install/media/internal</name>
        <name>pkg:/mypackage1</name>
        <name>pkg:/mypackage2</name>
   </software_data>
</software>
```

アンインストールするパッケージの一覧表示

アンインストール属性を持つ software_data 要素は、個々のパッケージのアンインストールまたはグループパッケージ定義のアンインストールに使用できます。

次の例で、`solaris-desktop` は、多数の個別パッケージを含むグループパッケージの名前です。

```
<software_data action="uninstall">
   <name>pkg:/group/system/solaris-desktop</name>
</software_data>
```

グループパッケージはアンインストールできます。グループパッケージをアンインストールしても、実際にアンインストールされるのはグループ定義だけです。そのグループの一部として以前にインストールされた個々のパッケージはアンインストールされません。ただし、グループパッケージをアンインストールせずに、それらの個々のパッケージをアンインストールできます。グループパッケージを保持すると、参照を継続する場合に役立つ可能性があります。また、名前タグを使用して個々のパッケージをアンインストールすることもできます。アンインストールするパッケージをアンインストールセクションの終わりに追加します。

インストールされたシステムのパブリッシャーの指定

ディストリビューションコンストラクタを使用して作成されたイメージでシステムがインストールされたあと、次の要素がシステムに影響を与えます。

```
<software name="set-ips-attributes">
```

ダウンロードおよびインストールする追加パッケージにインストール済みシステムがアクセスできる場所を指定するための、パブリッシャー名とオプションのミラー名のタグを提供します。

この要素では、IPS 属性も設定できます。IPS プロパティーの詳細は、pkg(1) のマニュアルページを参照してください。

構築チェックポイントの設定

マニフェストの実行要素は、イメージ作成処理中に実行される一連のチェックポイントを一覧表示します。チェックポイントは、このセクションに一覧表示された順序で実行されます。デフォルトのインストールイメージの構築に必要なデフォルトのチェックポイントは、各マニフェストに含まれています。

各チェックポイント名タグには、チェックポイントスクリプトが存在する場所を指定する mod-path 属性が含まれています。

一部のデフォルトチェックポイントタグには、デフォルト値を持つ引数が含まれています。dc_ai_sparc.xml サンプルマニフェストの次のチェックポイント例では、イメージ構築のためのブートアーカイブを作成し、そのタスクを実行するスクリプトを指定します。チェックポイント例には、引数ごとに特定の値が指定された引数フィールドも含まれています。

```
<checkpoint name="ba-arch"
    desc="Boot Archive Archival"
    mod_path="solaris_install/distro_const/checkpoints/
    boot_archive_archive"
    checkpoint_class="BootArchiveArchive">
    <kwargs>
        <arg name="size_pad">0</arg>
        <arg name="bytes_per_inode">0</arg>
        <arglist name="uncompressed_files">
             <argitem>etc/svc/repository.db</argitem>
             <argitem>etc/name_to_major</argitem>
             <argitem>etc/minor_perm</argitem>
             <argitem>etc/driver_aliases</argitem>
             <argitem>etc/driver_classes</argitem>
             <argitem>etc/path_to_inst</argitem>
             <argitem>etc/default/init</argitem>
             <argitem>etc/nsswitch.conf</argitem>
             <argitem>etc/passwd</argitem>
             <argitem>etc/shadow</argitem>
             <argitem>etc/inet/hosts</argitem>
        </arglist>
     </kwargs>
   </checkpoint>
```

この例で示すように、<kwargs> 要素には、構築中にチェックポイントに渡す必要があるキーワード引数が含まれています。<kwargs> 要素には、チェックポイントに渡される個々のキーワードを指定する際に使用可能な <arg name> 要素が含まれています。さらに、<arglist> 要素には、チェックポイントに渡される複数の<argitem> 値の一覧も含まれています。この例には、<arglist> 要素の未圧縮ファイルの一覧が含まれています。

各 <kargs> 一覧項目は二重引用符で囲まれています。二重引用符が使用されていない場合、または文字列全体が 1 組の二重引用符で囲まれている場合は、スペースおよび改行を含む文字列全体が 1 つの引数と解釈されます。引数をコンマで区切らないでください。

イメージの構築中に使用されるカスタムスクリプトを作成する場合、スクリプトの場所を指し示すチェックポイント要素を追加する必要があります。カスタムスクリプトのチェックポイントには、カスタムスクリプトの場所を示す <args> 要素のみが必要です。詳細および例については、23 ページの「カスタムスクリプトの作成と使用」を参照してください。

特定のチェックポイントで構築処理の一時停止と再開を制御するには、distro_const コマンドオプションを使用します。29 ページの「段階的にイメージを構築する方法」を参照してください。

例 2–2　SVR4 パッケージの追加

この例では、新しいチェックポイントがマニフェストに追加されます。この新しいチェックポイントは、イメージに追加される SVR4 パッケージとその場所を一覧表示します。その後、この新しいチェックポイントは実行セクションで参照されます。

最初に、新規 software 要素を追加することによって、新しいチェックポイントが作成されます。このチェックポイントには、ソフトウェアタイプとして SVR4、パッケージを検索する場所、およびパッケージをインストールする場所を指定します。

さらに、software_data 要素には、インストールされる特定の SVR4 パッケージが一覧表示されます。

```
<software name=transfer-svr4-install type="SVR4">
   <destination>
       <dir path={PKG_IMAGE_PATH}/>
   </destination>
   <source>
    <dir path="/path/to/packages"/>
   </source>
   <software_data action="install">
      <name>SUNWpackage1</name>
      <name>SUNWpackage2</name>
   </software_data>
</software>
```

PKG_IMAGE_PATH および BOOT_ARCHIVE の値は、チェックポイントに含まれる場合、distro_const ユーティリティーによって、それぞれ <ZFS Dataset>/build_data/pkg_image および <ZFS Dataset>/build_data/boot_archive に置き換えられます。この例では、SVR4 パッケージが <ZFS Dataset>/build_data/pkg_image にインストールされます。

最後に、この新しいチェックポイントは実行セクションで参照されます。

例 2–2 SVR4 パッケージの追加 (続き)

```
<execution stop_on_error="true">
   <checkpoint name="transfer-ips-install"
       desc="Transfer pkg contents from IPS"
       mod_path="solaris_install/transfer/ips"
       checkpoint_class="TransferIPS"/>
   <checkpoint name="set-ips-attributes"
       desc="Set post-install IPS attributes"
       mod_path="solaris_install/transfer/ips"
       checkpoint_class="TransferIPS"/>
   <checkpoint name="transfer-svr4-install"
      desc="Transfer pkg contents from SVR4 packages"
      mod_path="solaris_install/transfer/svr4"
      checkpoint_class="TransferSVR4"/>
```

ソフトウェア名はチェックポイント名と一致する必要があることに注意してください。この例では、どちらも「transfer-svr4–install」です。

カスタムスクリプトの作成と使用

ディストリビューションコンストラクタでは、構築するイメージの種類に基づいてカスタマイズするために使用する追加スクリプトを指定できます。マニフェストファイルはスクリプトを示し、スクリプトは汎用イメージをメディア固有の配布用に変換します。これらのスクリプトは、マニフェストファイルの実行セクションで参照されます。カスタムスクリプトチェックポイントはいくつでも指定できます。

注 – スクリプトのサポートは、アプリケーションパッケージに付属している、変更されていないデフォルトのスクリプトに限定されます。デフォルトのスクリプトをカスタマイズする場合は、まず、元のスクリプトのバックアップを作成してください。

▼ カスタムスクリプトの作成方法と使用方法

始める前に 独自のカスタムスクリプトを作成する場合は、次の点に注意してください。

- スクリプトには、Python プログラム、シェルスクリプト、またはバイナリを使用できます。
- スクリプトは、マニフェストファイルの実行セクションに記載されている順に実行されます。
- スクリプト (シェルと Python の両方のモジュール) 内で実行されるコマンドの標準出力 (`stdout`) およびエラー出力 (`stderr`) が、構築の完了時または試行時に報告するログファイルで取得されます。

1 新しいスクリプトを作成します。

2 新しいスクリプトをホームディレクトリ、またはシステム上かネットワーク上の他の場所に追加します。

root 役割を引き受けるユーザーがこれらのスクリプトを実行できることを確認してください。

3 チェックポイントを適切なマニフェストファイルの実行セクションに追加することによって、新しいスクリプトを参照します。

スクリプトには必ずフルパスを指定してください。チェックポイントは、マニフェストの実行セクションに記載されている順に実行されます。

マニフェストファイルの完了セクションに新しいスクリプトの参照を追加するときには、そのスクリプトのタスクの実行前または実行後にイメージ構築を一時停止するのに使用するチェックポイント名を指定する必要があります。必要に応じて、チェックポイント名に関連付けたカスタムメッセージを指定することもできます。このメッセージを省略すると、スクリプトのパスがデフォルトのチェックポイントメッセージとして使用されます。構築プロセス中にチェックポイントが実行されると、チェックポイントメッセージが表示されます。

注 – チェックポイント名には、番号ではなく、意味のある名前を使用してください。新しいスクリプトが追加された場合に、新しいスクリプトの新しいチェックポイントが使われるため、番号を使用したチェックポイントの秩序は乱れます。

次のチェックポイントの例では、「my-script」という名前のカスタムスクリプトが参照されます。

```
<checkpoint name="my-script"
        desc="my new script"
        mod_path="solaris_install/distro_const/checkpoints/custom_script"
        checkpoint_class="CustomScript">
        <args>/tmp/myscript.sh</args>
    </checkpoint>
```

4 (省略可能) 次のように、構築パラメータをチェックポイントの一部として指定します。

ここで、引数セクションの構築パラメータとして {PKG_IMAGE_PATH} が指定されます。

```
<checkpoint name="my-script"
        desc="my new script"
        mod_path="solaris_install/distro_const/checkpoints/my_script"
        checkpoint_class="CustomScript">
        <args>/tmp/myscript.sh {PKG_IMAGE_PATH}</args>
    </checkpoint>
```

PKG_IMAGE_PATH および BOOT_ARCHIVE の値は、チェックポイントに含まれる場合、distro_const ユーティリティーによって、それぞれ <ZFS Dataset>/build_data/pkg_image および <ZFS Dataset>/build_data/boot_archive に置き換えられます。

5 イメージを構築します。

イメージは1ステップで構築できます。または、構築の状態をチェックするために、さまざまなチェックポイントで構築を停止および再開できます。

手順については、第3章「イメージの構築」を参照してください。

6 (省略可能) 構築が完了したら、構築プロセスで報告するログファイルを参照できます。

構築出力には、ログファイルの場所が表示されます。

◆ ◆ ◆ 第 3 章

3 イメージの構築

使用するマニフェストファイルの準備が完了し、必要に応じてファイナライザスクリプトのカスタマイズが完了したら、`distro_const` コマンドを実行してイメージを構築できます。

`distro_const` コマンドを使用して、次のどちらかの方法でイメージを構築できます。

- 1 ステップで実行する
- 構築プロセス中にイメージおよびデバッグの内容を確認するために、必要に応じて構築を一時停止および再開する

distro_const コマンド

`distro_const` コマンドの完全な構文は、次のとおりです。

`Syntax: distro_const build [-v] [-r` *checkpoint_name*`] [-p` *checkpoint_name*`] [-l]` *manifest*

次のコマンドオプションをレビューします。

表 3–1 distro_const コマンドのオプション

コマンドオプション	説明
`distro_const build` *manifest*	指定されたマニフェストファイルを使用して、1 ステップでイメージを構築します。
`distro_const build- v`	冗長モード
`distro_const build -l` *manifest*	イメージの構築を一時停止または再開できるすべての有効なチェックポイントを一覧表示します。

表 3–1 distro_const コマンドのオプション (続き)

コマンドオプション	説明
`distro_const build -p` *checkpoint_name manifest*	指定されたチェックポイントでイメージの構築を一時停止します。
`distro_const build -r` *checkpoint_name manifest*	指定されたチェックポイントからイメージの構築を再開します。
`distro_const build -h`	コマンドのヘルプを表示します。

注 – `distro_const` コマンドを使用するには、root 役割になる必要があります。

▼ 1 ステップでイメージを構築する方法

始める前に `distribution-constructor` パッケージをダウンロードし、イメージ用のマニフェストを選択します。必要に応じて、マニフェストをカスタマイズし、カスタムスクリプトを追加します。

1. **root** の役割になります。
2. 一時停止せずにイメージの完全な構築を実行するには、次のように、オプションを指定せずに **`distro_const`** コマンドを使用します。

   ```
   # distro_const build manifest
   ```

   注 – `build` サブコマンドは必須です。

   manifest は、イメージの青写真として使用するマニフェストファイルの名前に置き換えてください。

   コマンド例を示します。

   ```
   # distro_const build /usr/share/distro_const/dc_livecd.xml
   ```

3. ディストリビューションコンストラクタは、イメージに必要なパッケージを取得します。
4. ディストリビューションコンストラクタは、マニフェストファイルで設定される仕様に従ってイメージを構築します。
5. **(省略可能)** 構築が完了したら、構築プロセスで報告するログファイルを参照できます。

   構築出力には、ログファイルの場所が表示されます。

▼ 段階的にイメージを構築する方法

distro_const コマンドには、構築するイメージのファイル、パッケージ、スクリプトなどのチェックやデバッグを行うために、イメージ生成プロセスのさまざまな段階で構築プロセスを停止および再開するためのオプションが用意されています。このプロセスでは、次に説明する基本的な手順に従って、distro_const コマンドで指定できるチェックポイントオプションを使用します。

1 **root** の役割になります。

2 イメージを構築する前に、構築を一時停止または再開できる有効なチェックポイントを確認します。

```
# distro_const build -l manifest.xml
```

注 – build サブコマンドは必須です。

このコマンドを実行すると、イメージの構築を一時停止または再開できる有効なチェックポイントが表示されます。このコマンドによって表示されたチェックポイント名を、他のチェックポイント処理コマンドオプションの有効な値として使用してください。

たとえば、次のコマンドでは、dc_livecd.xml という名前のマニフェストファイルを指定して、使用可能なチェックポイントを確認します。

```
# distro_const build -l /usr/share/distro_const/dc_livecd.xml
```

このコマンドを実行すると、有効なチェックポイントが表示されます。たとえば、チェックポイントには次のものが含まれている可能性があります。

```
Checkpoint        Resumable Description
--------------    --------- -------------
transfer-ips-install X Transfer package contents from IPS
set-ips-attributes   X Set post-installation IPS attributes
pre-pkg-img-mod      X Pre-package image modification
ba-init                Boot archive initialization
ba-config              Boot archive configuration
ba-arch                Boot archive archiving
grub-setup             Set up the GRUB menu
pkg-img-mod            Package image area modifications
create-iso             ISO image creation
```

注 – このサンプルコマンド出力の「Resumable」列の「X」は、そのチェックポイントから構築を再開できることを示します。

---

3 イメージを構築し、指定されたチェックポイントで構築を一時停止します。

```
# distro_const build -p checkpoint_name manifest
```

注 – build サブコマンド、および *checkpoint_name* と *manifest* フィールドは必須です。

たとえば、次のコマンドは、イメージの構築を開始し、ba-arch でイメージ領域が変更される前に構築を一時停止します。

```
# distro_const build -p ba-arch /usr/share/distro_const/dc_livecd.xml
```

4 指定されたチェックポイントからイメージの構築を再開します。

```
# distro_const build -r checkpoint_name  manifest
```

注 – 指定するチェックポイントは、前に構築の実行を停止したチェックポイント、またはそれ以前のチェックポイントにしてください。それより後のチェックポイントは無効です。*checkpoint_name* と *manifest* フィールド、および build サブコマンドは必須です。

たとえば、次のコマンドは、 ba-arch ステージでイメージの構築を再開します。

```
# distro_const build -r ba-arch /usr/share/distro_const/dc_livecd.xml
```

注 – build コマンドでは、一時停止オプションと再開オプションを組み合わせることができます。

5 (省略可能) 構築が完了したら、構築プロセスで報告するログファイルを参照できます。

構築出力には、ログファイルの場所が表示されます。