Oracle® Solaris 11.4 でのシステムサービスの管理

Part No: E75097-01
2018 年 8 月

Part No: E75097-01

ドキュメントのアクセシビリティについて

オラクルのアクセシビリティについての詳細情報は、Oracle Accessibility ProgramのWeb サイト(http://www.oracle.com/pls/topic/lookup?ctx=acc&id=docacc)を参照してください。

Oracle Supportへのアクセス

サポートをご契約のお客様には、My Oracle Supportを通して電子支援サービスを提供しています。詳細情報は(http://www.oracle.com/pls/topic/lookup?ctx=acc&id=info) か、聴覚に障害のあるお客様は (http://www.oracle.com/pls/topic/lookup?ctx=acc&id=trs)を参照してください。

目次

図目次

表目次

例目次

このドキュメントの使用法

- **概要** – Oracle Solaris のサービス管理機能 (SMF) を使用する方法について説明します。SMF は、より広範な Oracle Solaris の予測的セルフヒーリング機能のコンポーネントの 1 つです。
- **対象読者** – システムサービスを管理するシステム管理者
- **前提知識** – Oracle Solaris システムの管理に関する経験

製品ドキュメントライブラリ

この製品および関連製品のドキュメントとリソースは http://www.oracle.com/pls/topic/lookup?ctx=E75431-01 で入手可能です。

フィードバック

このドキュメントに関するフィードバックを http://www.oracle.com/goto/docfeedback からお聞かせください。

♦ ♦ ♦ 第 1 章

1 サービス管理機能の概要

Oracle Solaris サービス管理機能 (SMF) フレームワークは、システムサービスおよびアプリケーションサービスを管理します。SMF は、システムの操作に不可欠な重要なシステムサービスを管理し、データベースや Web サーバーなどのアプリケーションサービスを管理します。SMF は、ハードウェアまたはソフトウェアの障害が発生した場合でも、不可欠なシステムサービスおよびアプリケーションサービスが継続して実行できるように確保して、システムの可用性を高めます。

SMF は、構成ファイルを使用した方法に置き換わるサービスの管理機能であり、アプリケーションの起動に使用する推奨のメカニズムです。SMF は、`init` スクリプト始動メカニズム、`inetd.conf` 構成、および `rc?.d` のほとんどのスクリプトに置き換わります。可能な場合には、SMF は既存の管理方法との互換性を維持します。たとえば、顧客および ISV から提供されるほとんどの `rc` スクリプトは、SMF を使用せずに作業した場合と同様に機能します。

この章では、次について説明します。

- SMF の機能
- このリリースの新機能
- SMF の概念とコンポーネント
- 構成ファイルと SMF サービス
- 一部の SMF コマンドを使用するために必要な特権を取得する方法

カスタム SMF サービスの開発については、『Oracle Solaris 12 でのシステムサービスの開発』を参照してください。

SMF の機能

SMF フレームワークは Oracle Solaris 11 システムで常にアクティブになっています。SMF には次の機能があります。

- より高速にブートします。SMF は、独立したサービスを並列で起動することによって、大規模システムのブートを高速化します。
- 失敗したサービスを再起動します。SMF サービスは、ほかのサービスとの依存関係を適切に定義しています。サービスが失敗した場合、SMF は影響を受ける依存

サービスを報告します。SMF は自動的に、依存関係の順序で、失敗したサービスの再起動を試みます。

- サービスを検査します。サービスとプロセスの関係を表示します。サービスプロパティーの値を表示します。
- サービスを管理します。サービスの有効化、無効化、および再起動を行います。アップグレードおよびリブート後もこれらの変更を維持することも、一時的な変更を指定することもできます。
- サービスを構成します。
  - サービスプロパティーの値を変更します。
  - カスタムプロパティーを追加および削除します。
- サービスの変更を監査します。SMF はサービスまたはそのプロパティーに対するすべての管理上の変更について Solaris 監査レコードを作成します。SMF は、プロパティー値またはサービス状態が管理者によって設定されたかどうかを表示できます。
- プロパティーを変更する機能や、サービスを有効化、無効化、または再起動する機能などのタスクを、安全に非ルートユーザーに委任します。
- 特定のソフトウェアイベントまたはハードウェアの障害について通知する方法を構成します。
- サービス問題をデバッグします。有効なサービスが実行していない理由、またはサービスが別のサービスの実行を妨げている理由に関する説明を簡単に表示します。
- 既存のサービスの新しいインスタンスを作成するか、既存のサービスインスタンスを変更します。
- 新しいサービスを作成します。次の機能の詳細は、『Oracle Solaris 12 でのシステムサービスの開発』を参照してください。
  - サービス作成ツールの使用。
  - SMF サービスへの `inetd.conf` 構成の変換。
  - 構成ファイルへの SMF サービスプロパティーの変換。このメカニズムは、SMF で管理されるが、従来のように構成ファイルを必要とするアプリケーションとやりとりするサービスを橋渡しします。
  - `cron` ジョブのように、継続的にではなく定期的に実行されるサービスの作成。

このリリースの新機能

今回のリリースの SMF の新機能は次のとおりです。

追加のリポジトリレイヤー

3 つの新しいリポジトリレイヤーによって、構成のよりきめ細かい指定が可能になります。詳細は、30 ページの「リポジトリレイヤー」および 104 ページの「SMF プロファイルの作成」を参照してください。

1. `enterprise-profile` レイヤー – エンタープライズのすべての Oracle Solaris システムにまたがって適用される構成。
2. `node-profile` レイヤー – 特定の Oracle Solaris インスタンスに固有の構成。

   既存の `site-profile` レイヤーは、Oracle Solaris によって提供されなかった構成 (`sysconfig` またはユーザープロファイルによって提供された構成など) に使用されてきました。Oracle Solaris 11.4 では、`site-profile` レイヤーは、同じ場所またはサイトにある多数のシステムに共通の構成のためのものです。以前 `site-profile` レイヤーに属していたほとんどの構成は現在、`node-profile` レイヤーに属しています。

   システムが最初に Oracle Solaris 11.4 に更新されると、`/etc/svc/profile/site/` ディレクトリ内のすべてのプロファイルと `/etc/svc/profile/site.xml` プロファイル (存在する場合) は `/etc/svc/profile/node/` ディレクトリに移動されます。そのあと、これらのプロファイルが記述する構成が `node-profile` レイヤーの一部になります。移動されたプロファイルには、移動されたということと、前に配置されていた場所を示す名前が付けられます。たとえば、`/etc/svc/profile/site/sc_profile.xml` プロファイルは `/etc/svc/profile/node/migrated_etc_svc_profile_site_sc_profile.xml` に移動され、`/etc/svc/profile/site.xml` は `/etc/svc/profile/node/migrated_etc_svc_profile_site.xml` に移動される可能性があります。

   IPS パッケージによって提供された `/etc/svc/profile/site/` ディレクトリ内のファイルは移動されず、その構成は `site-profile` レイヤーの一部のままです。
3. `sysconfig-profile` レイヤー – `sysconfig` コマンドを対話形式で使用して指定された構成、および AI クライアントや `zoneadm` (`install`、`attach`、または `clone`) コマンドによってインストールされたプロファイルで指定されたすべての構成。

   システムが最初に Oracle Solaris 11.4 に更新されると、`/etc/svc/profile/sysconfig/` ディレクトリ内のすべての `.xml` ファイルは `/etc/svc/profile/backup/`*timestamp*`/profiles.tar` ファイルにバックアップされます。このディレクトリ内の一部のプロファイルには、`sysconfig` または Oracle Solaris インストーラによって実行された構成や、`admin` レイヤーにあるアクティブな構成が含まれていることがあります。アクティブな構成を含むプロファイルはそのまま残されるため、構成が `admin` レイヤーから `sysconfig-profile` レイヤーに効果的に移行されます。SMF リポジトリによって参照されていない `/etc/svc/profile/sysconfig` ディレクトリ内のプロファイルはすべて、`/etc/svc/profile/backup/`*timestamp*`/profiles.tar` ファイルにバックアップされたあとに削除されます。

入れ子になったプロパティーグループ

プロパティーグループの親を、サービスやサービスインスタンスまたは別のプロパティーグループにできます。この機能を別の方法で説明すると、プロパティーグループはプロパティーや、任意の数の子プロパティーグループを持つことができます。プロパティーグループを入れ子にすると、構成データ間の関係をよりき

め細かく定義できます。入れ子になったプロパティーグループ内のプロパティーを識別するには、48 ページの「サービス構成の検査」に示すように、親プロパティーグループもすべて指定します。

プロパティーグループ名とプロパティー名のオプション

`svcprop` および `svccfg` コマンドで `-G` オプションを使用すると、プロパティーグループを指定できます。`-P` オプションを使用すると、プロパティーを指定できます。`svccfg` コマンドで `-T` オプションを使用すると、プロパティータイプを指定できます。48 ページの「サービス構成の検査」を参照してください。

プロパティーグループ名とプロパティー名内の特殊文字

プロパティーグループ名とプロパティー名には、Uniform Resource Identifier (URI) Generic Syntax RFC 3986 で定義されている任意の文字を含めることができます。『Oracle Solaris 12 でのシステムサービスの開発』 の 「Naming Property Groups and Properties」を参照してください。

ステンシル定義

アプリケーションが同じ構文を持つ複数の構成ファイルを必要としている場合は、ステンシル定義機能を使用すると、1 つのサービスを使用してすべての構成ファイルを定義できます。

目標サービス

目標サービスは、依存サービスの構成可能なセットの単一モニタリングポイントを提供します。`online` 状態に到達できないほとんどのサービスは、暗黙のうちに `offline` 状態のままになります。目標サービスは `maintenance` 状態に移行し、管理者の介入なしではその依存関係のいずれかを満足できない場合は FMA アラートを生成します。

`svcadm goals` コマンド

`svcadm goals` コマンドは、目標サービスの目標の依存関係を構成します。73 ページの「目標サービスの目標の変更」を参照してください。`svcadm` コマンドのほかのサブコマンドと同様に、`svcadm goals` コマンドでは、同期動作を要求するための `-s` オプションを指定できます。

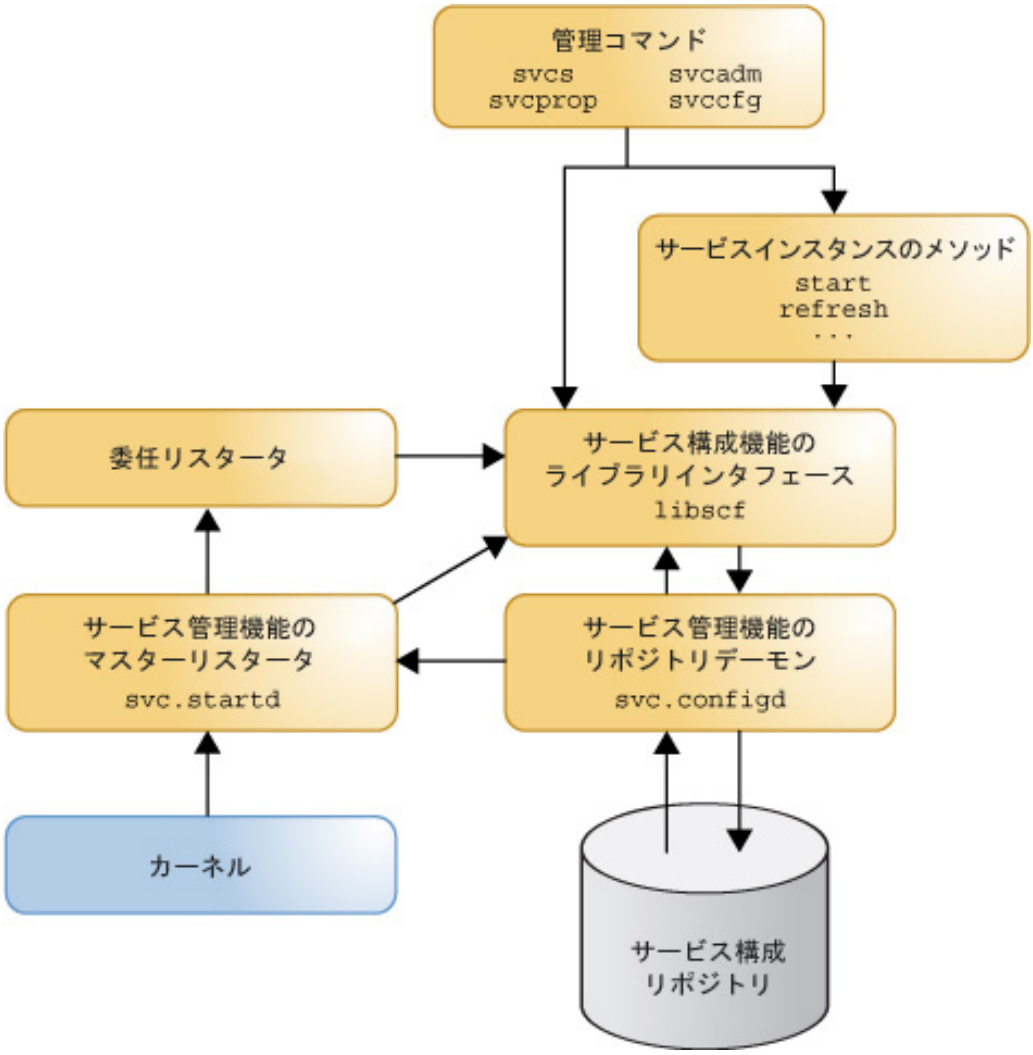
SMF の概念とコンポーネント

このセクションでは、このガイドでこれから使用する用語を定義します。

次の図に、SMF フレームワークのプライマリコンポーネントを示します。イメージをブートすると、SMF は、必要に応じてサービス構成リポジトリを更新し、リポジトリデータを読み取り、有効化されたサービスインスタンスを依存関係の正しい順序で起動します。独立したサービスは並列で起動されます。イメージを停止処理すると、依存関係の逆順序でサービスが停止処理されます。

次の図で、libscf は、リスタータがサービス構成リポジトリとのやりとりに使用するライブラリインタフェースです。サービス構成リポジトリと libscf ライブラリインタフェースとのやりとりは、svc.configd デーモンが管理します。svcs、svcprop、svcadm、および svccfg のコマンドは、管理者がサービス構成リポジトリとのやりとりに使用するインタフェースです。

図 1 サービス管理機能フレームワーク



SMF サービス

SMF サービスは、永続的に実行するアプリケーションであり、次のようなシステムエンティティーを表します。

- データベースや Web サーバーなどのアプリケーションサービス
- 基本的なシステムサービス
- デバイスのソフトウェア状態
- カーネル構成情報
- システムの即応性レベルに対応するマイルストーン

サービスインスタンスはサービスの子であり、アプリケーションとほかのサービスインスタンスに機能および依存関係をもたらします。インスタンスだけが状態を保有し、起動および停止できます。ハードウェアまたはソフトウェアの障害など何らかの理由でインスタンスが失敗した場合、SMF は自動的に障害を検出し、インスタンスと依存インスタンスを再起動します。

サービスのインスタンスでは、サービスの複数の構成を同時に実行することができます。サービスインスタンスは、共通のサービス構成を継承してカスタマイズします。たとえば、あるインスタンスはポート 80 で待機するように構成され、別のインスタンスはポート 1008 で待機するように構成された Web サーバーサービスを定義できます。ほとんどのサービスには `default` インスタンスがあります。プログラムの実行ではなく構成の格納に SMF を使用するサービスなど、一部のサービスにはインスタンスはありません。たとえば、`x11/x11-server` サービスにはインスタンスがありません。

SMF サービスは、サービスマニフェストと呼ばれるファイルに記述されています。マニフェストには、サービスインスタンス、依存関係、構成プロパティー、およびメソッドが記述されます。サービスメソッドはサービスインスタンスの起動、停止、リフレッシュを行います。メソッドは、デーモン、ほかのバイナリ実行可能ファイル、または実行可能スクリプトの場合があります。サービスプロファイルファイルでは、主にプロパティーを追加し、プロパティー値を追加およびオーバーライドすることによって、既存のサービスをカスタマイズできます。30 ページの「リポジトリレイヤー」で説明しているように、新しいプロパティーと値は、マニフェストで割り当てられた値の上にレイヤー化されます。マニフェストとプロファイルの詳細は、29 ページの「サービスバンドル」を参照してください。プロファイルは、104 ページの「SMF プロファイルの作成」で説明しているように、複数のシステムに同じカスタム構成を適用するための優れたツールでもあります。

サービス情報は、*SMF* データベースとも呼ばれるサービス構成リポジトリに格納されています。サービス構成リポジトリは、システム上の各サービスインスタンスの現在の状態と、サービスおよびサービスインスタンスごとの構成データを格納します。データは、30 ページの「リポジトリレイヤー」で説明しているように、値がどのように変更されたかに応じてレイヤーに格納されます。

SMF には、有効化、無効化、リフレッシュ、再起動など、サービスインスタンスで呼び出すことのできるアクションが用意されています。各サービスインスタンスは、リスタータがこれらの管理アクションを実行することによって管理されます。通常、リスタータは、メソッドを実行してサービスインスタンスをある状態から別の状態に移行させることによってアクションを実行します。リスタータの詳細は、26 ページの「サービスリスタータ」を参照してください。

マイルストーンサービスとは、システムの `init` 状態などのシステムの即応性レベルを表す特殊なタイプのサービスです。マイルストーンは、ほかのサービスインスタンスが起動するときに依存するサービスです。たとえば、実行レベルは、`svc:/milestone/multi-user-server` などのマイルストーンサービスによって表されます。マイルストーンはまた、`svc:/milestone/devices`、`svc:/milestone/network`、`svc:/milestone/name-services` などのサービスグループの即応性を示すために使用することもできます。`svc:/milestone/goals` サービスは、システムが目的どおりに機能するために実行する必要のあるサービスのセットを定義します。『Oracle Solaris 12 でのシステムサービスの開発』 の 第 7 章, 「Creating a Service that Notifies if Conditions are not Satisfied」を参照してください。

サービスモデル

SMF サービスは、次のいずれかのモデルになります。

一時サービス

このサービスは、何らかの作業を行なったあと、長時間実行するプロセスを開始せずに終了します。

子または待機サービス

このサービスは、その子プロセスが正常に終了したときに必ず再起動します。正常に終了した子プロセスはエラーとして扱われません。

契約またはデーモンサービス

このサービスは、長時間実行するデーモンを起動するか、サービス契約の一部としてまとめられた複数の関連するプロセスを起動します。契約サービスは、開始したプロセスと、依存サービスおよびその起動順序を管理します。ユーザーによる管理が必要なのは高度なサービスだけです。

目標サービス

このサービスは、その依存サービスのいずれかが実行されていないかどうかを管理者に通知します。『Oracle Solaris 12 でのシステムサービスの開発』 の 第 7 章, 「Creating a Service that Notifies if Conditions are not Satisfied」を参照してください。

定期的なサービスは、これらのどのモデルにも適合しない特殊なケースです。定期的なサービスまたはスケジュールされているサービスは、実行時間の短いプロセスを定期的な間隔またはスケジュールされた間隔で開始し、関連付けられた契約済みプロセスが存在しない場合は次の実行までオンラインのままになります。詳細は、『Oracle Solaris 12 でのシステムサービスの開発』 の 第 3 章, 「Creating a Service to Run Periodically」および『Oracle Solaris 12 でのシステムサービスの開発』 の 第 4 章, 「Creating a Service to Run on a Specific Schedule」を参照してください。

サービス名

各サービスおよびサービスインスタンスは、障害管理リソース識別子 (FMRI) で表されます。サービスインスタンスの完全な FMRI は次の形式になります。

svc:/*service_name*:*instance_name*

service_name は、`network/dns/client` や `application/pkg/server` などの階層的な名前です。*service_name* の最後のスラッシュ (/) の前のコンポーネントは、サービスのカテゴリです。`application`、`device`、`milestone`、`network`、`system` などのカテゴリは、サービスの目的の識別に役立ちます。

`site` カテゴリは、ユーザー自身の SMF サービスを作成するときに名前の競合を回避できるように予約されています。たとえば、`svc:/site/`*tool* という名前のサイト固有のサービスは、`svc:/`*tool* という名前の Oracle Solaris サービスと競合しません。

サービスインスタンス名は、親サービス名にコロンに続けて付加されます。たとえば、`svc:/system/identity:node` および `svc:/system/identity:domain` は `svc:/system/identity` サービスのインスタンスです。

スクリプトでは、完全なサービスインスタンス名を使用することをお勧めします。対話形式では、名前は右端部分に短縮でき、これが結果的に一意の名前になります。たとえば、`svc:/system/identity` は `identity` に短縮でき、`svc:/system/identity:domain` は `identity:domain` に短縮できます。インスタンス名は、その前にサービス名の一部とコロンを置く必要があります。

サービス状態

SMF サービスは、どの時点でも、次のいずれかの状態になっています。

- `degraded` – インスタンスは実行中であるか実行可能になっていますが、機能が制限されています。
- `disabled` – インスタンスは有効でなく、実行中でも実行可能でもありません。

- `maintenance` – インスタンスは有効になっていますが、実行できません。管理アクションがまだ完了していないためにインスタンスは `maintenance` 状態に遷移している可能性があります。それ以外の場合は、問題を解決するために管理アクションが必要です。
- `offline` – インスタンスは有効になっていますが、実行中でも実行可能でもありません。たとえば、サービスが有効になっていてもその依存関係が満たされていない場合、そのサービスは `offline` 状態に保たれます。
- `online` – インスタンスは有効になっており、実行中であるか実行可能になっています。`online` 状態は、正しく構成されたサービスインスタンスのすべての依存関係が満たされた場合に予想される動作状態です。
- `uninitialized` – この状態はすべてのサービスの初期状態です。

サービスインスタンスは、管理アクションや依存サービスの状態などの条件に応じて、状態を遷移します。たとえば、`disabled` 状態のインスタンスを有効にすると、新しく有効になったインスタンスは、最初に`offline` 状態になり、その依存関係がすべて充足されると `online` 状態に遷移します。

現在の状態に加えて、管理者は補助状態を表示できます。リスタータ (26 ページの「サービスリスタータ」を参照) は、補助状態を使用して、状態に関する情報を格納します。マスターリスタータは、補助状態を使用して、インスタンスが現在の状態に遷移した理由を格納します。たとえば、`online` 状態に遷移したあとの補助状態の値は通常 `dependencies_satisfied` です。『Oracle Solaris 12 でのシステムサービスの開発』 の 「Creating a Periodic Service」で説明されているように、定期的なリスタータは、補助状態を使用して、定期的なタスクが現在実行中であるかどうかを格納します。

これらのサービス状態と、サービスインスタンスがこれらの状態をどのように遷移するかについての詳細は、`smf`(7) のマニュアルページを参照してください。

サービス依存関係

サービスは、サービス、サービスインスタンス、またはファイルに対して依存関係を持つ場合があります。サービス依存関係はサービス間の関係を定義します。

依存関係によって、サービスが起動するタイミングや自動停止するタイミングが決まります。サービスが有効になっていてもその依存関係が充足されていない場合、そのサービスは `offline` 状態になっています。サービスが有効になっていてその依存関係が充足されている場合、そのサービスは起動します。サービスの起動が成功すると、そのサービスは `online` 状態に遷移します。

サービス依存関係は、サービスが状態を遷移すると再評価されます。サービス依存関係が充足されていても、あとから充足されなくなる場合があります。ファイル依存関係は一度だけ評価されます。

依存関係は必須の場合もオプションの場合もあります。サービス依存関係は実行している必要がある場合も、無効になっている必要がある場合もあります。依存サービスは、そのサービス依存関係のいずれかが停止またはリフレッシュしたときに再起動するかどうかを構成できます。

依存関係では次の機能が可能です。

- スケーラブルで再現可能な初期化プロセス
- 独立したサービスを並行して起動することによる、並行機能を持つシステム上でのシステムの起動の高速化
- 障害によって直接影響を受けるサービスだけを依存関係の正しい順序で再起動することによる、正確な障害隔離および障害回復

サービスリスタータ

各 SMF サービスインスタンスは、リスタータによって管理されます。リスタータは、インスタンス構成を取得し、実行環境を提供します。すべてのリスタータに共通の情報については、`smf_restarter`(7) のマニュアルページを参照してください。

マスターリスタータデーモン

`svc.startd` デーモンは、SMF のマスターリスタータデーモンであり、すべてのサービスインスタンスのデフォルトリスタータです。`svc.startd` デーモンは、すべてのサービスインスタンスおよびその依存関係の状態を管理します。インスタンスがオンライン状態に移行したときに依存関係が充足されていると、マスターリスタータは、ほかのインスタンスの起動メソッドを呼び出すか、起動メソッドを呼び出すように委任リスタータに指示します。マスターリスタータは、インスタンスの依存関係が充足されなくなると、サービスインスタンスを停止します。リスタータは、インスタンスに障害が発生すると、インスタンスの再起動を試みます。インスタンスは、そのすべての依存関係が充足されるまでオンラインにできないので、インスタンスの依存関係が、インスタンスの再起動動作の判断に役立ちます。それぞれの依存関係宣言で設定されているプロパティーによって、その依存関係が必要かどうかと、どのような場合に依存関係が再起動したときにインスタンスを再起動するかが定義されます。

その他のタスクの中では、`svc.startd` デーモンが、適切な `/etc/rc*.d` スクリプトを適切な実行レベルで起動します。これは以前に `init` で行われていた作業です。

次の例では、`svc.startd` が `network/ipmp:default` サービスインスタンスのリスタータであることを示します。ほかの出力は、この例から省略されています。

```
$ svcs -l ipmp:default
restarter   svc:/system/svc/restarter:default
```

restarter プロパティーが空または svc:/system/svc/restarter:default に設定されている場合、サービスインスタンスは svc.startd によって管理されます。svc.startd デーモンの詳細は、svc.startd(8) のマニュアルページを参照してください。

委任リスタータ

一部のサービスは、起動時に共通の動きが見られます。委任リスタータは、これらのサービスに特定の実行環境とアプリケーション固有の再起動動作を提供します。restarter プロパティーで指定された委任リスタータが利用可能になると、このリスタータがサービスインスタンスの管理を担います。

Oracle Solaris には次の委任リスタータが含まれています。

inetd
: inetd 委任リスタータは、インターネットサービスを常に実行しておくのではなく、要求に応じてサービスを起動できます。inetd リスタータは、入力および出力ファイル記述子として、ネットワーク接続から構成される環境をそのサービスインスタンスに提供します。inetd デーモンの詳細は、inetd(8) のマニュアルページを参照してください。次の例では、inetd が cups/in-lpd:default サービスインスタンスのリスタータであることを示します。ほかの出力は、この例から省略されています。

```
$ svcs -l cups/in-lpd:default
restarter   svc:/network/inetd:default
```

svc.periodicd
: 定期的なリスタータデーモン svc.periodicd は、システムの起動時に svc:/system/svc/periodic-restarter サービスの一部として自動的に呼び出され、何らかの障害が発生した場合は自動的に再起動されます。定期的なリスタータによって起動されたサービスは、永続的にオンラインのままになりますが、その起動メソッドタスクを定期的またはスケジュールされた時間にのみ実行します。定期的なサービスの起動メソッドタスクは、比較的短い期間実行されたあとで終了することになっています。定期的なリスタータの詳細は、the svc.periodicd(8) のマニュアルページを参照してください。定期的なサービスの詳細は、『Oracle Solaris 12 でのシステムサービスの開発』 の 第 3 章,「Creating a Service to Run Periodically」を参照してください。

svc.zones
: 非大域ゾーンは、svc:/system/zones:default 委任リスタータサービスによって管理されます。次のコマンドは、svc:/system/zones:default が z1 非大域ゾーンのリスタータであることを示しています。

```
$ svcs -R svc:/system/zones:default
STATE          STIME    FMRI
online         12:11:12 svc:/system/zones/zone:z1
```

ゾーンのリスタータの詳細は、svc.zones(8) のマニュアルページを参照してください。ゾーンサービスの詳細は、『Oracle Solaris ゾーンの作成と使用』を参照してください。

サービスプロパティーおよびプロパティーグループ

依存関係、メソッド、状態、アプリケーションデータなどサービスに関する情報は、一連のプロパティーとしてサービス構成リポジトリに格納されます。プロパティーは、サービスまたはサービスのインスタンスのどちらかに対して定義できます。サービスに対して設定されるプロパティーは、そのサービスのすべてのインスタンスに継承されます。インスタンスに対して設定されたプロパティーは、そのインスタンスだけで使用されます。サービスインスタンスは、継承したプロパティーの値をカスタマイズでき、親サービスに対して定義されていない追加プロパティーを定義できます。

プロパティーはプロパティーグループにまとめられます。一般的なプロパティーグループには次のものがあります。

- `general` – インスタンスが有効かどうかなどの情報を格納します。
- `restarter` – インスタンスの現在の状態など、サービスのリスタータによって保存される実行時情報を格納します。
- `start`、`refresh`、`stop` – サービスを起動、リフレッシュ、または停止するためにどのプログラムを実行するかなどの情報を格納します。
- `config` – アプリケーションデータを保持するためにサービス開発者が使用します。

プロパティーグループを、サービスやインスタンスまたは別のプロパティーグループの子にできます。別のプロパティーグループの子であるプロパティーグループは、入れ子になったプロパティーグループとも呼ばれます。入れ子になったプロパティーグループは、構成データ間の相互関係をより完全に表します。

プロパティーとプロパティーグループの詳細は、 smf(7) のマニュアルページを参照してください。

サービス構成リポジトリ

各サービスの情報は、サービス構成リポジトリに格納されており、このリポジトリは *SMF* データベースとも呼ばれます。サービス構成リポジトリには、サービス、インスタンス、プロパティーグループ、およびプロパティーとして情報が格納されます。サービス構成リポジトリには、サービス開発者によって定義された情報に加えて、システムでの各サービスインスタンスの起動時間や現在の状態などの情報が格納されます。

リポジトリには、永続的な構成情報と、サービスの SMF 実行時データが格納されます。

- 永続的な構成情報は、データのソースに応じてレイヤーに格納されます。30 ページの「リポジトリレイヤー」を参照してください。
- 実行時データ、つまり非永続的な構成情報は、リブートすると保持されず、リポジトリには、非永続データのレイヤー情報は格納されません。非永続データは通常、アクティブなプログラム状態を保持します。

リポジトリには、タイプ、値の制約、プロパティーの説明などのサービステンプレートデータも格納されます。テンプレートデータはサービスマニフェストで定義されます。テンプレートデータの詳細は、`smf_template`(7) のマニュアルページを参照してください。

サービス構成リポジトリは、SMF インタフェースを使ってのみ操作または照会できます。`svcs`、`svcprop`、`svcadm`、および `svccfg` のコマンドを使用するか、`libscf`(3LIB) のマニュアルページに一覧表示されたサービス構成機能ライブラリの関数を使用します。プロパティー値は読み書きでき、指定したレイヤーおよびスナップショットにプロパティー値を表示できます。レイヤーの詳細は、30 ページの「リポジトリレイヤー」を参照してください。スナップショットの詳細は、32 ページの「リポジトリのスナップショット」を参照してください。選択したサービスインスタンスまたは親サービスのプロパティーだけを表示することも、プロパティーの合成ビューを表示することもできます。合成ビューでは、親サービスに対して設定されたプロパティーとサービスインスタンスに対して設定されたプロパティーの両方が表示されます。表示される値はサービスインスタンスに対して設定された値です。

サービスバンドル

サービスバンドルとは、サービスまたはサービスインスタンスのサービス構成リポジトリに格納されている情報を格納した XML ファイルです。サービスバンドルで提供される情報は、サービス構成リポジトリに格納され、リポジトリからエクスポートできます。標準の場所におけるサービスバンドルは、システムブート中にリポジトリにインポートされます。

サービスバンドルには、マニフェストとプロファイルの 2 つのタイプがあります。

マニフェスト	マニフェストには、特定のサービス群またはサービスインスタンス群に関連付けられたプロパティーがすべて含まれます。
プロファイル	プロファイルは通常、マニフェストで提供される情報を追加したりオーバーライドしてカスタマイズしたサービスまたはサービスインスタンスを提供します。カスタマイズの例としては、プロパティーの追加やプロパティー値の変更があります。

マニフェストの標準の場所は `/lib/svc/manifest` です。プロファイルの標準の場所は `/etc/svc/profile` です。

システムがブートするか、マニフェストインポートサービスが再起動すると、マニフェストがインポートされ、プロファイルが新規作成または変更されていれば、これが適用されます。サービスバンドルを提供する IPS パッケージは、パッケージをインストールするときにマニフェストインポートサービスを再起動するように指定できます。

ローカルカスタマイズは、`/etc/svc/profile/site` ディレクトリ内の接尾辞が `.xml` のプロファイルファイルで提供できます。同じリポジトリレイヤー内で同じサービスまたはインスタンスに対して同じプロパティーが複数のマニフェストまたはプロファイルによって定義されている場合、SMF は使用する値を判断できません。このような競合が検出された場合、インスタンスは maintenance 状態に置かれます。レイヤーの詳細は、30 ページの「リポジトリレイヤー」を参照してください。

Oracle Solaris にサービスを提供する以外に、サービスバンドルは、さまざまなシステムに対してカスタム構成を提供することもできます。

システムプロファイル `/etc/svc/profile/generic.xml` はインストール中に適用されます。このプロファイルは変更しないでください。このシステムプロファイルに行われた変更はアップグレードで上書きされます。詳細は、`smf_bootstrap`(7) のマニュアルページを参照してください。

リポジトリレイヤー

サービス構成リポジトリには、単一のプロパティーに対して異なる値を格納できます。リポジトリは、データのソースに応じて、レイヤーにデータを格納します。ソースには、マニフェスト、プロファイル、および SMF コマンドやライブラリインタフェースを使用して実行されたカスタマイズを指定できます。さまざまなレイヤー内の値を表示して、実行中の構成で使用されている値のソース、つまり、ある値がマニフェストまたはプロファイルで割り当てられたか、あるいは管理者によって変更されかを判断できます。57 ページの「値が設定されているレイヤーの表示」を参照してください。

SMF コマンドおよびライブラリインタフェースを使用して行われた構成変更は、`admin` レイヤーにのみ表示されます。ほかのレイヤーでの構成は、標準の場所にあるプロファイルおよびマニフェストファイルで定義されます。ファイルからリポジトリにプロパティーを追加したり、ファイル内のプロパティー値を変更したりしたとき、`svccfg listprop` コマンドの `-f` または `-o file` オプションを使用することによって、その構成を提供したファイルの名前を表示できます。58 ページの「構成に関係したファイルの表示」を参照してください。

プロパティーが異なるレイヤーに異なる値を割り当てた場合、サービスインスタンスによって使用される値は、レイヤー階層内の最上位レイヤーの値になります。次

の表は、階層内のレイヤーの順序を示したものです。たとえば、あるプロパティーに node-profile レイヤー内の値が含まれている場合、その値は manifest レイヤーまたはその他のいずれかの下位レイヤー内の値をオーバーライドします。admin レイヤーにプロパティーの値があるとき、その値はほかのあらゆるレイヤーに設定されているほかのすべての値をオーバーライドします。

レイヤー	コンテンツ
admin	SMF コマンドを使用して管理者によって行われたか、SMF ライブラリインタフェースを使用してアプリケーションによって行われた変更。admin レイヤーには、マニフェストをインポートするか、標準以外の場所のプロファイルを適用することによって行われた変更も含まれます。標準以外の場所を使用する場合の注意事項については、97 ページの「マニフェストおよびプロファイルのインポートおよび適用」を参照してください。
sysconfig-profile	管理者が sysconfig コマンドを対話形式で使用して行なった変更、および Automated Installer (AI) クライアントまたはゾーンによってインストールされたプロファイルで指定された構成。対話式システム構成 (SCI) ツールを使用する場合は、sysconfig コマンドを対話形式で使用しています。SCI ツールの使用については、『Oracle Solaris 12 システムの自動インストール』 の 第 9 章, 「Unconfiguring or Reconfiguring an Oracle Solaris Instance」を参照してください。 AI クライアントプロファイル (installadm create-profile または installadm update-profile によって作成されます)、ゾーンプロファイル (zoneadm install、zoneadm attach、または zoneadm clone によって作成されます)、および sysconfig コマンドを対話形式で使用して作成されたプロファイルは、特に指定されていないかぎり /etc/svc/profile/sysconfig ディレクトリに格納されます。zoneadm コマンドの install、attach、または clone サブコマンド、あるいは sysconfig configure コマンドで -c *dir* オプションを指定し、*dir* にサブディレクトリ enterprise、site、または node が含まれている場合、これらのディレクトリ内のプロファイルからの構成は、適切な enterprise-profile、site-profile、または node-profile レイヤー内に設定されます。 システムを構成するために sysconfig コマンドが対話形式で使用されたとき、admin レイヤーでも定義されているプロパティー値が定義された場合は、sysconfig コマンドを使用して定義された新しいプロパティー値が有効になるように、admin レイヤー内のこれらのプロパティー値は削除されます。
node-profile	Oracle Solaris インスタンスに固有の構成。この構成は、/etc/svc/profile/node ディレクトリ内の SMF プロファイル、および旧バージョンの /etc/svc/profile/site.xml ファイルに基づきます。/etc/svc/profile/site.xml ファイルに新しい構成を追加しないでください。
site-profile	同じ場所またはサイトにあるシステムに共通の構成。この構成は、/etc/svc/profile/site ディレクトリ内の SMF プロファイルに基づきます。/var/svc/profile は標準の場所としては非推奨になりました。新しいプロファイルを /var/svc/profile ディレクトリに置かないでください。
enterprise-profile	すべての Oracle Solaris システムに適用される構成。この構成は、/etc/svc/profile/enterprise 内の SMF プロファイルに基づきます。
system-profile	/etc/svc/profile/generic.xml および /etc/svc/profile/platform.xml システムプロファイルのほか、/etc/svc/profile/system ディレクトリ内のすべてのプロファイルからのすべての値。
manifest	/lib/svc/manifest および /var/svc/manifest ディレクトリ内のマニフェストの値。/var/svc/manifest ディレクトリに新しいサービスマニフェストを配置しないでください。

構成の競合はどのレイヤーでも許可されていません。SMF コマンド、`sysconfig` コマンド、または SMF ライブラリインタフェースを使用して設定された構成は、以前の設定を上書きします。ある単一レイヤー内の複数のファイルで構成が競合し、その構成が上位のレイヤーでは設定されていない場合、`manifest-import` サービスログにはこの競合が記され、競合している構成を使用したサービスは開始されません。詳細は、104 ページの「競合する構成」を参照してください。

構成データのレイヤーを指定して、管理カスタマイズであるデータや、ソフトウェアで提供されたデータを表示し、したがって特定することができます。構成データの取得元のレイヤーがクライアントによって指定されない場合は、最上位のレイヤーデータが与えられます。最上位のレイヤーは上の表に示す順序で決定され、`admin` レイヤーが最上位レイヤーで、`manifest` レイヤーが優先順位がもっとも低いレイヤーです。`admin` レイヤーにプロパティーの値がある場合、これがリポジトリで提供する値になります。ローカルカスタマイズは、このようにして、システムがインストールされたときに与えられた値よりも優先されます。

リポジトリのスナップショット

リポジトリは、サービスが正しく起動するごとに、読み取り専用のスナップショットを取得します。これらのスナップショットを使用すれば、必要に応じて簡単に以前の作業状態に戻せます。次のスナップショットはどのインスタンスでも使用できます。

`initial` サービスとそのインスタンスが最初にインポートされたときの初期構成。マニフェストのインポート前にプロファイルがサービスまたはインスタンスを起動した場合、`initial` スナップショットは作成されません。

`previous` すでに提供されているサービスに対してマニフェストのインポートが実行されたときに取得された現在の構成。このサービスは、インポートされているマニフェストまたは別のマニフェストによって、すでに提供されている可能性があります。

`running` サービスインスタンスの実行中の構成。構成データを変更する場合、`svcadm refresh` コマンドまたは `svccfg refresh` コマンドを使用して、新しい値を実行中のスナップショットにプロモートします。

`start` `online` 状態への正常な遷移中に取得された構成。

リポジトリのバックアップ

SMF は、サービス構成リポジトリの次のバックアップを自動的に作成します。

- `boot` バックアップは、システムを起動するたびに、リポジトリに対する最初の変更が行われる直前に行われます。
- サービスが新しいマニフェストをインポートしたか、アップグレードスクリプトを実行した場合、`manifest_import` のバックアップは、`svc:/system/early-manifest-import:default` または `svc:/system/manifest-import:default` が完了する前に行われます。

タイプごとに 4 つのバックアップがシステムで保守され、必要に応じてもっとも古いバックアップから削除されます。

これらのいずれかのバックアップからリポジトリを復元できます。120 ページの「バックアップからリポジトリを復元する方法」を参照してください。

構成ファイルと SMF サービス

SMF は、アプリケーションの起動に使用する推奨のメカニズムです。ほとんどの場合で、SMF が構成ファイルを使用した方法に代わって、サービスの管理を行います。このセクションでは、一般的な旧バージョンの構成スクリプトおよびファイルを扱う方法について説明します。

`/etc/rc?.d` のスクリプト

`/etc/rc?.d` のディレクトリ (ここで、`?` は、実行レベルを表します) には、実行レベルの遷移時に実行するサービスを管理するための、旧バージョンの初期化および終了スクリプトが含まれます。`/etc/rc?.d` のスクリプトで以前に実装されたほとんどのサービスが SMF で管理されます。一部の `/etc/rc?.d` のスクリプトは、これらのサービスを `/etc/rc*.d` のスクリプトと予想するサードパーティー製アプリケーションを使用できるように保持されています。これらのスクリプトは、`/etc/init.d` ディレクトリのファイルにハードリンクされます。`/etc/rc?.d` のスクリプトと実行レベルに関する詳細は、`/etc/init.d/README` ファイル、`/etc/rc?.d` ディレクトリ内の `README` ファイル、および`inittab`(5)のマニュアルページを参照してください。実行制御スクリプトを変換する手順については、『Oracle Solaris 12 でのシステムサービスの開発』 の 「How to Convert a Run Control Script to an SMF Service」を参照してください。`rc?d` スクリプトを変換したあと、S*script* から s*script* にスクリプトの名前を変更して、スクリプトを実質的に削除します。

`/etc/init.d` のスクリプト

`/etc/init.d` ディレクトリには、`init` 状態を変更するための初期化および終了スクリプトが含まれます。これらのスクリプトの一部は、`/etc/rc?.d` のディレクトリ内のスクリプトにハードリンクされています。`/etc/init.d` のスクリプトの

詳細は、/etc/init.d/README とinit.d(5)のマニュアルページを参照してください。

旧バージョンの init.d 実行制御スクリプトは、svc ではなく lrc から始まる SMF FMRI で表されます。たとえば、/etc/rc2.d/S47pppd PPP 構成スクリプトは、lrc:/etc/rc2_d/S47pppd サービスで表されます。これらの lrc サービスの状態は legacy_run です。次の例に示すように、旧バージョンのサービスの名前と起動時間を一覧表示できますが、SMF を使用してこれらのサービスを管理することはできません。

```
$ svcs lrc:\*
STATE          STIME     FMRI
legacy_run      9:34:54  lrc:/etc/rc2_d/S47pppd
legacy_run      9:34:54  lrc:/etc/rc2_d/S89PRESERVE
$ svcs -l lrc:/etc/rc2_d/S47pppd
svcs: Operation not supported for legacy service 'lrc:/etc/rc2_d/S47pppd'
$ svccfg -s lrc:/etc/rc2_d/S47pppd listprop
svccfg: Operation not supported for legacy service 'lrc:/etc/rc2_d/S47pppd'
```

/etc/inittab のエントリ

init による /etc/inittab ファイル制御プロセスディスパッチのエントリ。/etc/inittab ファイルを直接編集しないでください。代わりに、SMF サービスを変更してください。ttymon に渡されるパラメータを変更する方法の例については、88 ページの「ttymon プロパティー値の変更方法」を参照してください。

/etc/inittab ファイルエントリの書式の詳細は、inittab(5) のマニュアルページを参照してください。実行レベルの詳細は、inittab(5) のマニュアルページと /etc/init.d/README を参照してください。

/etc/inetd.conf ファイル

inetd.conf ファイルを使用して以前に構成されたサービスは、現在、SMF を使用して構成されています。inetd.conf ファイル内の構成を使用できるようにするには、SMF に変換する必要があります。126 ページの「SMF サービスへの inetd サービスの変換」を参照してください。すでに SMF サービスに変換されている inetd サービスについては、98 ページの「inetd で制御されるサービスの変更」を参照してください。

/etc/nscd.conf ファイル
/etc/nsswitch.conf ファイル
/etc/resolv.conf ファイル

これらのファイルは編集しないでください。編集内容は失われます。これらのファイルは、ファイルを解析する可能性のあるアプリケーションとの下位互換性のために、SMF データから自動的に生成されます。82 ページの「プロパティー値の設定」に示すように、svccfg setprop コマンドを使用してプロパティー値を変更します。

nscd.conf ファイルの機能は、svc:/system/name-service-cache SMF サービスに置き換えられました。nscd.conf ファイルを編集する代わりに name-

service-cache のどのプロパティーを構成するかを確認するには、nscd.conf(5) のマニュアルページを参照してください。

nsswitch.conf ファイルの機能は、svc:/system/name-service/switch SMF サービスに置き換えられました。nsswitch.conf ファイルを編集する代わりに、name-service/switch のどのプロパティーを構成するかを確認するには、nsswitch.conf(5) のマニュアルページを参照してください。

resolv.conf ファイルの機能は、svc:/network/dns/client SMF サービスに置き換えられました。resolv.conf ファイルを編集する代わりに、dns/client のどのプロパティーを構成するか確認するには、resolv.conf(5) のマニュアルページを参照してください。

これらのファイルは、編集できない構成ファイルの例です。このようなファイルはその他にも存在します。101 ページの「ファイルによって構成されるサービスの変更」で説明しているように、構成を変更するときに構成ファイルを編集する方法が適切な場合も若干あります。構成ファイルを編集する前に、ファイルにおけるコメントと関連するマニュアルページをすべて読んで、ファイルの編集が、関連するサービスの構成を変更する適切な方法であることを確認してください。

サービス管理の特権

サービス状態と構成を変更するには、特権を増やす必要があります。必要な特権を得るには、次のいずれかの方法を使用します。必要な役割またはプロファイルを判断する方法や特権を割り当てる方法など、役割、プロファイル、および承認の詳細は、『Oracle Solaris 11.4 でのユーザーとプロセスのセキュリティー保護』を参照してください。

役割

roles コマンドを使用して、自分に割り当てられている役割を一覧表示します。役割の名前を指定して su コマンドを使用すると、その役割になります。この役割として、その役割に割り当てられている権利プロファイルで許可されているすべてのコマンドを実行できます。たとえば、役割にサービス構成権利プロファイルが割り当てられている場合、svccfg および svcadm コマンドを実行して、サービスプロパティーを変更してサービス状態を変更できます。

権利プロファイル

profiles コマンドを使用して、自分に割り当てられている権利プロファイルを一覧表示します。次のいずれかの方法を使用して、権利プロファイルによって実行が許可されているコマンドを実行します。

- pfbash や pfksh などのプロファイルシェルを使用する。
- 実行するコマンドの前に pfexec コマンドを使用する。一般的に、実行する特権コマンドごとに pfexec コマンドを指定する必要があります。

承認

SMF 操作に必要な承認の詳細は、`smf_security`(7)のマニュアルページを参照してください。サービス構成権利プロファイルが特定のサービスを管理するために十分でない場合は、次のプロパティーに関してそのサービスを検査してください。

- `action_authorization`、`modify_authorization`、`read_authorization`、および `value_authorization` プロパティーは、必要な承認を指定します。個々のサービスに、サービス独自の特定の承認が必要になる場合もあります。
- `method` プロパティーグループのプロパティーは、そのメソッドを実行するための要件 (ユーザーや特権セットなど) を指定できます。

sudo コマンド

サイトのセキュリティーポリシーに応じて、自分のユーザーパスワードで `sudo` コマンドを使用し、特権コマンドを実行できる場合があります。

♦ ♦ ♦ 第 2 章

2 サービスに関する情報の取得

この章では、サービスに関する次のような情報を取得する方法を示します。

- サービス状態、依存関係、およびほかのプロパティー値
- 契約サービスにより開始されたプロセス
- トラブルシューティング用のログファイルの場所
- FMA イベントおよびサービス遷移イベントの通知設定

システム上のサービスの一覧表示

`svcs` コマンドは、サービスインスタンスの状態およびステータスを一覧表示するためのプライマリコマンドです。

サービス状態の表示

これらの例で示される状態については、24 ページの「サービス状態」を参照してください。

例 1 有効なすべてのサービスの一覧表示

オプションまたは引数を使用せずに `svcs` コマンドを実行すると、このシステムで有効になっているすべてのサービスインスタンスのほか、一時的に無効になっているインスタンスが表示されます。

この一覧表示で `disabled` 状態になっているサービスインスタンスは、次回のシステムブート時に有効になります。`legacy_run` 状態のインスタンスは SMF で管理されません。これらの旧バージョンのサービスに関する詳細は、33 ページの「構成ファイルと SMF サービス」を参照してください。`maintenance`、`degraded`、または `offline` の状態のサービスがある場合は、45 ページの「サービス状態に関する詳細の取得」を参照してください。

STIME 列には、一覧表示されている状態にインスタンスが移行した時間が表示されます。インスタンスが 24 時間以前にこの状態に移行した場合は、STIME 列には日付が表示されます。

```
$ svcs
STATE          STIME    FMRI
legacy_run     Sep_09   lrc:/etc/rc2_d/S47pppd
legacy_run     Sep_09   lrc:/etc/rc2_d/S81dodatadm_udaplt
legacy_run     Sep_09   lrc:/etc/rc2_d/S89PRESERVE
disabled       Sep_09   svc:/system/vbiosd:default
online         Sep_09   svc:/system/early-manifest-import:default
online         Sep_09   svc:/system/svc/restarter:default
...
```

例 2 インストールされているすべてのサービスの一覧表示

次回のブート時に自動的には有効にならない `disabled` インスタンスを含め、このシステムにインストールされているすべてのサービスインスタンスを一覧表示するには、`svcs -a` コマンドを使用します。

```
$ svcs -a
```

一覧表示された状態から別の状態に遷移しているサービスインスタンスの場合、アスタリスク (*) がその状態に付加されます。たとえば `offline*` であれば、インスタンスがまだその起動メソッドを実行していると考えられます。

疑問符 (?) は、状態が存在しないか認識されない場合に表示されます。

例 3 サービスのすべてのインスタンスの一覧表示

サービス名を指定して `svcs` コマンドを実行すると、サービスのすべてのインスタンスが一覧表示されます。`-o` オプションの詳細は、40 ページの「選択したサービス情報の表示」を参照してください。

```
$ svcs -Ho inst identity
node
domain
```

サービスに関する詳細の表示

`svcs -l` コマンドを実行すると、指定したサービスインスタンスごとの長いリストが表示され、ここには、インスタンス状態、インスタンスのログファイルおよび構成ファイルへのパス、依存関係タイプ、依存関係再起動属性値、および依存関係状態に関する詳細が示されます。次の例では、このサービスインスタンスの必要な依存関係がすべてオンラインになっていることを示します。無効になっている 1 つの依存関係は、オプションの依存関係です。依存関係タイプと再起動属性値の詳細は、41 ページの「サービス依存関係の表示」を参照してください。`svcs -l` の出力で、24 ページの「サービス状態」で説明しているもの以外の状態が、依存関係の

状態である可能性があります。詳細は、svcs(1) のマニュアルページを参照してください。次の例では、指定したサービスインスタンスが一時的に有効になっており、オンラインであり、サービスが契約タイプのサービスであることも示しています。サービスタイプの定義については、23 ページの「サービスモデル」を参照してください。offline* のように状態値の後ろにアスタリスクが続いている場合、インスタンスは遷移中であり、次の状態を表すフィールドには none ではなく状態値が示されます。state_time は、一覧表示されている状態にインスタンスが移行した時間です。

```
$ svcs -l net-snmp
fmri         svc:/application/management/net-snmp:default
name         net-snmp SNMP daemon
enabled      true (temporary)
state        online
next_state   none
state_time   September 17, 2013 05:57:26 PM PDT
logfile      /var/svc/log/application-management-net-snmp:default.log
restarter    svc:/system/svc/restarter:default
contract_id  160
manifest     /etc/svc/profile/generic.xml
manifest     /lib/svc/manifest/application/management/net-snmp.xml
dependency   require_all/none svc:/system/filesystem/local (online)
dependency   optional_all/none svc:/milestone/name-services (online)
dependency   optional_all/none svc:/system/system-log (online)
dependency   optional_all/none svc:/network/rpc/rstat (disabled)
dependency   require_all/restart svc:/system/cryptosvc (online)
dependency   require_all/restart svc:/milestone/network (online)
dependency   require_all/refresh file://localhost/etc/net-snmp/snmp/snmpd.conf (online)
dependency   require_all/none svc:/milestone/multi-user (online)
```

例 4 契約サービスによって開始されたプロセスの表示

svcs -p コマンドを使用すると、契約サービスインスタンスによって開始されたプロセスのプロセス ID とコマンド名が表示されます。net-snmp サービスは、一連の管理情報ベース (MIB) を通じてシステムに関する情報を収集する /usr/sbin/snmpd SNMP エージェントを管理します。

```
$ svcs -p net-snmp
STATE          STIME    FMRI
online         17:57:26 svc:/application/management/net-snmp:default
               17:57:26     5022 snmpd
```

例 5 プロセス停止後に自動的に再起動する契約サービスの表示

契約サービスインスタンスは、契約が空になると自動的に再起動します。SMF は、ハードウェアまたはソフトウェアの障害イベントからの自動復旧の一環として、契約サービスインスタンスに関連したプロセスの再起動を試みます。次の例では、/usr/sbin/snmpd プロセスが強制終了されたあとに、新しいプロセス ID で自動的に再起動されています。net-snmp:default インスタンスはオンラインのままですが、開始時間が新しくなっています。

```
$ kill 5022
$ svcs -p net-snmp
STATE          STIME    FMRI
online         17:57:59 svc:/application/management/net-snmp:default
```

```
                17:57:59     5037 snmpd
```

選択したサービス情報の表示

svcs コマンドによる出力は、ほかのコマンドにパイピングしたりスクリプトで使用する場合に非常に役立ちます。svcs コマンドでは、-o オプションを使用して、必要な情報の列とその列の順序を指定できます。たとえば、サービス名とインスタンス名(別々の列)、サービスの現在の状態と次の状態、契約 ID などを出力できます。-s オプションおよび -S オプションで、1 つ以上の列について出力のソート順を指定できます。使用可能な列のリストについては、svcs(1) のマニュアルページの「COLUMNS」セクションを参照してください。複数の -s オプションが指定された場合は、相加的に動作します。

定期的なサービスでは、3 つの追加情報が表示されることがあります。定期的なサービスについては、『Oracle Solaris 12 でのシステムサービスの開発』 の 第 3 章, 「Creating a Service to Run Periodically」および svc.periodicd(8) のマニュアルページを参照してください。

LRUN
: この定期的なサービスインスタンスの起動メソッドが実行された最終時間。定期的ではないサービスインスタンスでは、この列にハイフン文字 (-) が表示されます。

NRUN
: この定期的なサービスインスタンスの起動メソッドの実行がスケジュールされている次の時間。定期的ではないサービスインスタンスでは、この列にハイフン文字 (-) が表示されます。

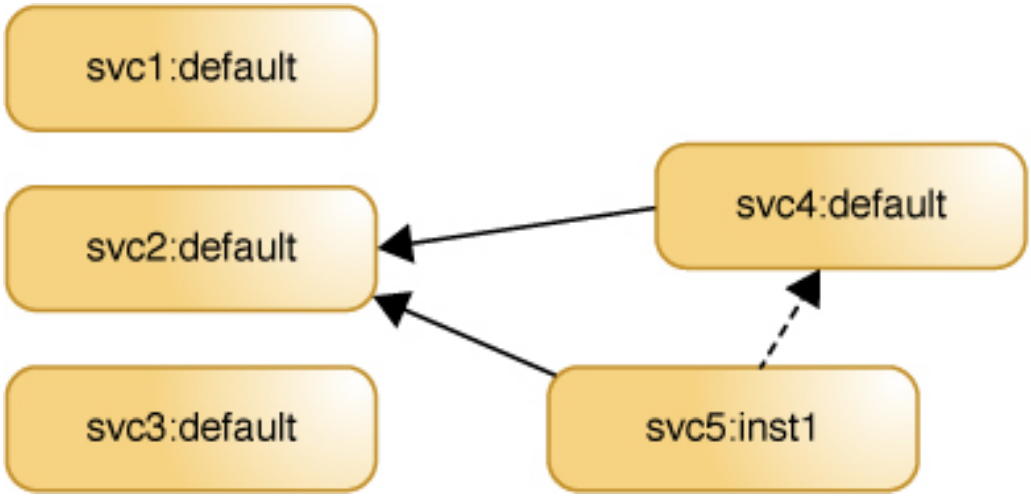
ASTATE
: サービスインスタンスの補助状態。定期的なインスタンスではないサービスインスタンスは、このプロパティーを使用して、インスタンスが最新の遷移を行なった理由を表示します。このプロパティーの値は通常、インスタンスが offline から online 状態に遷移した理由であるため、定期的なインスタンスではないインスタンスではほとんど常に、その値として dependencies_satisfied が表示されます。定期的なサービスインスタンスでは、インスタンスが起動メソッドの実行を待機しているかどうかを示す running または scheduled のいずれかがこの列に表示される可能性があります。詳細は、『Oracle Solaris 12 でのシステムサービスの開発』 の 「Specifying the periodic_method Element」を参照してください。

サービス依存関係の表示

依存関係は、サービスインスタンスの状態遷移を制御します。依存関係の高度な説明については、25 ページの「サービス依存関係」を参照してください。詳細な説明と、さまざまな種類の依存関係を指定する方法については、『Oracle Solaris 12 でのシステムサービスの開発』を参照してください。

次の図では、`svc1:default`、`svc2:default`、および `svc3:default` のサービスインスタンスは、起動するために、ほかのサービスまたはファイルなどのリソースを必要としていません。これらのインスタンスは、ほかのリソースを待機することなく、並列して起動し、それぞれの起動メソッドを実行し、オンライン状態に移行できます。`svc4:default` インスタンスは、`svc2:default` インスタンスがオンラインになるまで、その起動メソッドを実行できません。`svc5:inst1` インスタンスには、`svc2:default` リソースと `svc4:default` リソースの両方が必要です。`svc5:inst1` の `svc4:default` に対する依存関係はオプションの依存関係であり、`svc4:default` が有効かつオンライン、無効、または存在していないという状態のいずれかになっている場合に充足されます。`svc5:inst1` インスタンスは、`svc2:default` がオンラインになるまで待機する必要があります。`svc4:default` が存在し有効である場合は、`svc5:inst1` は `svc4:default` もオンラインになるまで待機する必要があります。`svc4:default` が存在し無効であったり、存在していない場合は、`svc5:inst1` は `svc4:default` を待機する必要はありません。

図 2 サービス依存関係



依存関係のグループ

各依存関係は、次のいずれかのグループに割り当てられます。グループは、そのグループ内の依存関係が充足される条件を定義します。

`require_all` 次の両方の条件を満たしている場合、この依存関係は充足します。

- このグループ内のすべてのサービス依存関係が実行している (`online` または `degraded`)。
- このグループ内のすべてのファイル依存関係が存在している。

`require_any` 次のどちらかの条件を満たしている場合、この依存関係は充足します。

- このグループ内のサービス依存関係の少なくとも 1 つが実行している (`online` または `degraded`)。
- このグループ内のファイル依存関係の少なくとも 1 つが存在している。

`optional_all` このグループ内のすべてのサービス依存関係がすべて次のどちらかの条件を満たしている場合、この依存関係は充足します。

- サービスが実行している (`online` または `degraded`)。
- サービスの実行に管理アクションが必要である。サービスは、存在していない、不完全である、`disabled` 状態である、`maintenance` 状態である、または `offline` 状態であり、起動には管理アクションが必要な依存関係を待機している。

このグループ内のファイル依存関係は存在していても、存在していなくても構いません。

サービスインスタンスが遷移中であり、起動に管理者の操作は不要である場合、この依存関係は充足されません。この場合、依存サービスは、この依存関係が起動するまで待機するか、管理アクションなしに依存関係を起動できないと判断するまで待機します。

`exclude_all` 次の両方の条件を満たしている場合、この依存関係は充足します。

- このグループ内のすべてのサービス依存関係が、`disabled` 状態であるか、`maintenance` 状態であるか、または存在していない。
- このグループ内のすべてのファイル依存関係が存在していない。

サービスが依存するインスタンスの一覧表示

svcs -d コマンドは、特定のサービスが依存するサービスインスタンスを一覧表示します。

この例では、system-repository サービスが依存するサービスインスタンスを表示します。

```
$ svcs -d system-repository
STATE          STIME    FMRI
online         Sep_09   svc:/milestone/network:default
online         Sep_09   svc:/system/filesystem/local:default
online         Sep_09   svc:/system/filesystem/autofs:default
```

svcs -l コマンドは、特定のサービスが依存するサービスも一覧表示します。-l オプションの出力では、依存関係の名前と状態に加え、依存関係のタイプ (つまりグループ) と、依存関係の restart_on プロパティーの値も表示されます。この例では、依存関係のうち 2 つが必須で、1 つがオプションです。これらのグループ内の依存関係が依存サービスに与える影響の詳細は、42 ページの「依存関係のグループ」を参照してください。依存関係の restart_on プロパティーのさまざまな値が依存サービスに与える影響の詳細は、42 ページの「依存関係のグループ」を参照してください。

```
$ svcs -l system-repository
fmri         svc:/application/pkg/system-repository:default
name         IPS System Repository
enabled      false
state        disabled
next_state   none
state_time   Mon Sep 09 18:42:28 2013
restarter    svc:/system/svc/restarter:default
manifest     /lib/svc/manifest/application/pkg/pkg-system-repository.xml
dependency   require_all/error svc:/milestone/network:default (online)
dependency   require_all/none svc:/system/filesystem/local:default (online)
dependency   optional_all/error svc:/system/filesystem/autofs:default (online)
```

svcprop コマンドを使用して、これらの依存関係を一覧表示することもできます。この形式では、依存関係のグループと restart_on 値は別々の行に表示され、依存関係の状態は表示されません。

```
$ svcprop -g dependency system-repository:default
network/entities fmri svc:/milestone/network:default
network/grouping astring require_all
network/restart_on astring error
network/type astring service
filesystem-local/entities fmri svc:/system/filesystem/local:default
filesystem-local/grouping astring require_all
filesystem-local/restart_on astring none
filesystem-local/type astring service
autofs/entities fmri svc:/system/filesystem/autofs:default
autofs/grouping astring optional_all
autofs/restart_on astring error
autofs/type astring service
```

サービスに依存するインスタンスの一覧表示

svcs -D コマンドは、特定のサービスに依存するサービスインスタンスを一覧表示します。

この例では、system-repository サービスに依存するサービスインスタンスを表示します。

```
$ svcs -D system-repository
STATE          STIME    FMRI
online         16:39:30 svc:/application/pkg/zones-proxyd:default
```

次のコマンドは、zones-proxyd が system-repository に依存していることを確認します。

```
$ svcs -do svc,desc zones-proxyd
SVC                               DESC
application/pkg/system-repository IPS System Repository
system/filesystem/minimal         minimal file system mounts
milestone/network                 Network milestone
```

次のコマンドは、zones-proxyd が system-repository にどのように依存しているかに関する詳細を表示します。この出力の最終行には、zones-proxyd サービスには system-repository サービスが実行していることが必要であることが示され、system-repository が現在実行していることが示されています。この出力には、system-repository サービスがリフレッシュされると zones-proxyd サービスが再起動されることも示されています。

```
$ svcs -l zones-proxyd
fmri         svc:/application/pkg/zones-proxyd:defaulr
name         Zones Proxy Daemon
enabled      true
state        online
next_state   none
state_time   January 6, 2014 04:39:30 PM PST
restarter    svc:/system/svc/restarter:default
manifest     /lib/svc/manifest/application/pkg/zoneproxyd.xml
dependency   require_any/none svc:/system/filesystem/minimal (online)
dependency   require_any/error svc:/milestone/network (online)
dependency   require_all/restart svc:/application/pkg/system-repository (online)
```

サービスが自動的に再起動するかどうかの表示

依存関係のいずれかが停止またはリフレッシュしたときに再起動するように、実行中のサービスを構成できます。実行中のサービス (online または degraded 状態) は、その依存関係が充足されていないと、offline 状態に遷移します。依存関係が停止またはリフレッシュしたあとにサービスを再起動すると、依存関係が再度充足され、依存サービスは実行状態に戻ることがあります。

require_all、require_any、または optional_all の依存関係が停止またはリフレッシュしたあとで、サービスが再起動するかどうかは、次の要因によって決まります。

- 依存関係が停止またはリフレッシュしたかどうか。停止した場合、依存関係が停止した理由がハードウェア障害やコアダンプなどのエラーによるものか、管理アクションなどのほかの理由によるものか。
- 依存関係の restart_on 属性の値。可能な値は、none、error、restart、および refresh です。

次の表に示すように、依存関係の restart_on 属性の値が none である場合は、依存関係が停止またはリフレッシュしても依存サービスは再起動しません。依存関係の restart_on 属性の値が refresh である場合、依存関係が停止またはリフレッシュすると常に依存サービスは再起動します。restart_on の値が error である場合、エラーのために依存関係が停止した場合にのみ依存サービスは再起動します。restart_on の値が restart である場合、依存関係がリフレッシュした場合にのみ依存サービスは再起動します。

表 1 依存関係の停止後のサービスの自動再起動

require_all、require_any、または optional_all の依存関係	依存関係 restart_on 属性の値			
停止またはリフレッシュイベント	**none**	**error**	**restart**	**refresh**
エラーによる停止	再起動なし	再起動	再起動なし	再起動
その他の停止	再起動なし	再起動なし	再起動なし	再起動
リフレッシュ	再起動なし	再起動なし	再起動	再起動

43 ページの「サービスが依存するインスタンスの一覧表示」では、system-repository サービスに、2 つの require_all 依存関係と 1 つの optional_all 依存関係があることを示しています。次のコマンドは、milestone/network サービスまたは system/filesystem/autofs サービスがエラーのために停止した場合に、system-repository サービスが再起動し、ほかの理由でこれらのサービスが停止またはリフレッシュした場合には、再起動しないことを示します。system-repository サービスは、system/filesystem/local サービスが何らかの理由でリフレッシュまたは停止した場合、再起動しません。

```
$ svccfg -s system-repository:default listprop -o propname,propval '*restart_on'
network/restart_on          astring     error
filesystem-local/restart_on astring     none
autofs/restart_on           astring     error
```

サービス状態に関する詳細の取得

引数を付けずに svcs -x コマンドを実行すると、次のサービスインスタンスに関する説明が表示されます。

- 有効だが、実行していないインスタンス。
- ほかの有効なサービスの実行を妨げているインスタンス。

すべての有効なサービスが実行している場合、`svcs -x` コマンドは出力を生成しません。

次の例では、`pkg/depot` サービスは、その起動メソッドが失敗したために `maintenance` 状態になっています。

```
$ svcs -x
svc:/application/pkg/depot:default (IPS Depot)
 State: maintenance since September 11, 2013 01:30:42 PM PDT
Reason: Start method exited with $SMF_EXIT_ERR_FATAL.
   See: http://support.oracle.com/msg/SMF-8000-KS
   See: pkg.depot-config(8)
   See: /var/svc/log/application-pkg-depot:default.log
Impact: This service is not running.
```

この出力には、My Oracle Support での予測的セルフヒーリングのナレッジ記事、マニュアルページ、およびログファイルを参照して、起動メソッドが失敗した理由を判断することが示唆されています。ログファイルを表示する別の方法については、47 ページの「サービスログファイルの表示」を参照してください。`maintenance` 状態になっているサービスを修正する方法については、113 ページの「機能低下、オフライン、または保守であるインスタンスの修復」を参照してください。

次の例では、`print/server` サービスに、実行していない依存サービスが存在します。`print/server` サービスが無効になっているので、この依存サービスを実行できません。

```
$ svcs -x
svc:/application/print/server:default (LP print server)
 State: disabled since Fri Mar 08 14:42:32 2013
Reason: Disabled by an administrator.
   See: http://sun.com/msg/SMF-8000-05
   See: lpsched(8)
Impact: 2 dependent services are not running.  (Use -v for list.)
$ svcs -xv
svc:/application/print/server:default (LP print server)
 State: disabled since Fri Mar 08 14:42:32 2013
Reason: Disabled by an administrator.
   See: http://sun.com/msg/SMF-8000-05
   See: man -M /usr/share/man -s 8 lpsched
Impact: 2 dependent services are not running:
        svc:/application/print/rfc1179:default
        svc:/application/print/ipp-listener:default
$ svcs -D print/server
STATE          STIME    FMRI
online         Mar_08   svc:/milestone/multi-user:default
offline        Mar_08   svc:/application/print/ipp-listener:default
offline        Mar_08   svc:/application/print/rfc1179:default
```

`svcs -x` コマンドに与えられた引数が、このセクションの冒頭に記した条件を満たしていない場合、出力にはインスタンスの状態に関する理由は示されませんが、詳細が得られるリソースは引き続き表示されます。

```
$ svcs -x smb
svc:/network/smb:default (SMB properties)
 State: online since Thu Sep 12 19:16:56 2013
   See: smb(5)
   See: /var/svc/log/network-smb:default.log
Impact: None.
```

サービスログファイルの表示

SMF は、サービスインスタンスごとに、重要なリスタータアクション、メソッド標準出力、および標準エラー出力に関する情報を `/var/svc/log/`*service*`:`*instance*`.log` に記録します。ログファイル名では、*service* 名におけるスラッシュの代わりにハイフンが代用されます。`-L`、`-l`、または `-x` オプションを付けて `svcs` コマンドを実行すると、指定したサービスインスタンスのログファイルの完全パス名が表示されます。`svcs -xL` コマンドには、ログファイルの最後の数行が表示され、完全なログファイルを表示するには `svcs -Lv` コマンドを使用する必要があると記されます。`svcs -Lv` コマンドは完全なファイルを表示しますが、非常に長くなる可能性があります。たとえば、エディタでログファイルを表示したり、最後の *n* 個のエントリだけを表示する場合、`svcs -L` コマンドの出力に対して操作します。

次の例では、ログファイルを使用して、`svcs` リストに示されたサービスが一時的に無効になっている理由を調べる方法を示します。

```
$ svcs
STATE          STIME    FMRI
legacy_run     Sep_09   lrc:/etc/rc2_d/S47pppd
legacy_run     Sep_09   lrc:/etc/rc2_d/S81dodatadm_udaplt
legacy_run     Sep_09   lrc:/etc/rc2_d/S89PRESERVE
disabled       Sep_09   svc:/system/vbiosd:default
online         Sep_09   svc:/system/early-manifest-import:default
online         Sep_09   svc:/system/svc/restarter:default
...
$ svcs -x vbiosd
svc:/system/vbiosd:default (BIOS call emulation)
 State: disabled since Mon Sep  9 18:42:37 2013
Reason: Temporarily disabled by service method: "vbiosd is not supported on UEFI systems."
   See: http://support.oracle.com/msg/SMF-8000-1S
   See: vbiosd(8)
   See: /var/svc/log/system-vbiosd:default.log
Impact: This service is not running.
$ svcs -xL vbiosd
svc:/system/vbiosd:default (BIOS call emulation)
 State: disabled since Mon Sep  9 18:42:37 2013
Reason: Temporarily disabled by service method: "vbiosd is not supported on UEFI systems."
   See: http://support.oracle.com/msg/SMF-8000-1S
   See: vbiosd(8)
   See: /var/svc/log/system-vbiosd:default.log
Impact: This service is not running.
   Log:
[ Sep  9 18:42:27 Enabled. ]
[ Sep  9 18:42:37 Executing start method ("/lib/svc/method/svc-vbiosd start"). ]
[ Sep  9 18:42:37 Method "start" exited with status 101. ]
[ Sep  9 18:42:37 "start" method requested temporary disable: "vbiosd is not supported on
 UEFI systems"
```

```
 ]

   Use: 'svcs -Lv svc:/system/vbiosd:default' to view the complete log.
$ svcs -L vbiosd
/var/svc/log/system-vbiosd:default.log
$ view `svcs -L vbiosd`
```

その他のログファイルで役立つ可能性のあるものとしては、マスターリスタータデーモンのログやシステムログなどがあります。ログファイル名を表示し、 `svc.startd` リスタータデーモンのログファイルを確認するには、`svcs` コマンドでサービス名 `restarter` を使用します。`syslogd` システムログデーモンのログファイル名およびログファイルを表示するには、サービス名 `system-log` を使用します。

```
$ svcs -L restarter
/var/svc/log/svc.startd.log
$ svcs -L system-log
/var/svc/log/system-system-log:default.log
/var/svc/log/system-system-log:rsyslog.log
```

システムブート時に表示するメッセージングの分量を変更する手順については、122 ページの「起動メッセージングの量の指定」を参照してください。
あるサービス状態に、あるいはその状態から遷移したとき、または FMA イベントが起きたときにユーザーに通知するようにサービスを構成する手順については、75 ページの「状態遷移および FMA イベントの通知の構成」を参照してください。

サービス構成の検査

サービス構成は、サービスおよびサービスインスタンスに対して設定されるプロパティーで表され、サービス構成リポジトリのレイヤーに格納されます。サービスに対して設定されたプロパティーは、そのサービスのすべてのインスタンスに継承されます。インスタンスに対して設定されたプロパティーは、そのインスタンスだけで使用されます。サービスインスタンスは、継承したプロパティーの値をカスタマイズでき、親サービスに対して定義されていない追加プロパティーを定義できます。

このセクションでは、プロパティー値を取得する方法と、値がサービスに対してグローバルか、インスタンスに固有か、ソフトウェアで提供されたものか、または管理カスタマイズであるかを識別する方法について説明します。

プロパティーおよびプロパティーグループの説明の表示

`svccfg describe` コマンドは、プロパティーの現在の値を含め、サービスのプロパティーグループおよびプロパティーの説明を表示します。オペランドを付けずに

describe を実行すると、選択したサービスまたはサービスインスタンスのすべてのプロパティーグループおよびプロパティーの説明が表示されます。現在の値の説明や使用可能な値のリストなどの詳細を表示するには、-v オプションを使用します。テンプレート情報を表示するには、-t オプションを使用します。

```
$ svccfg -s pkg/server describe network/restart_on
network/restart_on astring     none
    Determines whether to restart the service due to a dependency refresh, restart, or
 failure.
$ svccfg -s pkg/server describe -v network/restart_on
network/restart_on astring     none
    type: astring
    required: true
    Determines whether to restart the service due to a dependency refresh, restart, or
 failure.
    visibility: readwrite
    minimum number of values: 1
    maximum number of values: 1
  value: none
    value description: Never restart due to dependency refresh, restart, or failure.
  value constraints:
    value name: none
    value name: error
    value name: restart
    value name: refresh
```

サービスおよびインスタンスのプロパティー値の表示

このセクションの例では、サービスおよびインスタンスのプロパティーとプロパティーグループを、別々のビュー、レイヤー、およびスナップショットで表示する方法について説明します。次の定義に注意してください。レイヤーとスナップショットについては、28 ページの「サービス構成リポジトリ」を参照してください。

合成ビュー 親サービスから継承されたプロパティーとインスタンスで定義されたプロパティー。親サービスで定義されたプロパティーがインスタンスでカスタマイズされている場合は、カスタマイズされた値が表示されます。svcprop コマンドは、デフォルトでは合成ビューを表示します。

コミットされたプロパティー 現在有効なプロパティー値。svcprop コマンドは、デフォルトではコミットされたプロパティーを表示します。またコミットされていないプロパティー値を表示するには、-c オプションを指定して svcprop コマンドを使用します。

直接接続されたプロパティー コミットされていないプロパティー値。プロパティーはインスタンスまたは親サービスで設定されたが、インスタンスがリフレッシュされていません。そのインスタンスをリフレッシュすると、実行中のスナップショットが作成され、プロパティーがコミットされます。直接接続されたプロパティーを表示するには、-c オプションまたは -C オプションを指定して svcprop コマンドを使用します。

編集ビュー　　コミットされていない最新のプロパティー変更。

例 6　　現在使用中のインスタンスおよび継承したプロパティーの一覧表示

`svcprop` コマンドはデフォルトで、実行中のスナップショットでプロパティーに割り当てられている値を表示します。これが現在使用されている値です。デフォルトでは、`svcprop` コマンドは、実行中のスナップショットの合成ビューでプロパティーを表示します。つまり、インスタンス固有のプロパティーと継承されたプロパティーの両方が表示されます。継承されたプロパティーの値がインスタンスでカスタマイズされている場合、そのインスタンスで設定された値が表示されます。出力では、プロパティーごとに 1 行ずつ表示され、プロパティーグループとプロパティー名 (スラッシュで区切られます)、プロパティー値のデータタイプ、およびプロパティー値が示されます。プロパティーまたはグループ名が指定されていない場合、実行中のスナップショット内のすべてのプロパティー値が示されます。

FMRI またはパターンオペランドで完全なサービス名を指定しているが、インスタンスを指定していない場合、サービスに対してのみ設定されたプロパティーが表示されます。インスタンスに対してのみ設定されたプロパティーは表示されません。次のコマンドは、サービス依存関係、サービスのタイプ、プロファイルおよびマニフェストファイルのパスなどのプロパティーを表示します。

```
$ svcprop svc:/system/identity
```

インスタンスを指定すると、親サービスから継承され、そのインスタンスおよびプロパティーについてカスタマイズされたプロパティーの合成ビューが表示されます。次のコマンドは、親サービスおよびプロパティーからこのインスタンスに継承されたプロパティーを含め、指定されたインスタンスの実行中のスナップショット内のプロパティーをすべて一覧表示します。継承されたプロパティーの値がこのインスタンスに対してカスタマイズされている場合は、カスタマイズされた値が表示されます。この例では、追加の依存関係、このインスタンスを起動する実行可能ファイルへのパス、このインスタンスのログファイルへのパス、このインスタンスの状態に関する情報などのプロパティーを表示します。

```
$ svcprop svc:/system/identity:domain
```

例8「編集ビューでのサービスおよびインスタンス値の一覧表示」での `-c` オプションの使用も参照してください。

例 7　　現在使用されている指定のプロパティーまたはプロパティーグループの一覧表示

特定のプロパティーグループ内の特定のプロパティーまたはすべてのプロパティーを表示するには、`-p` オプションを使用します。次の例では、特定のプロパティーの値を表示します。

```
$ svcprop -p config/prop1 example
svc:/site/example:default/:properties/config/prop1 count 1
```

```
svc:/site/example:newinst/:properties/config/prop1 count 1
```

-p オプションは複数回指定できます。

```
$ svcprop -p config/prop1 -p config/prop2 example
svc:/site/example:default/:properties/config/prop1 count 1
svc:/site/example:default/:properties/config/prop2 count 2
svc:/site/example:newinst/:properties/config/prop1 count 1
svc:/site/example:newinst/:properties/config/prop2 count 2
```

次の例では、特定のプロパティーグループ内のすべてのプロパティーの値を表示します。

```
$ svcprop -p config example
svc:/site/example:default/:properties/config/prop1 count 1
svc:/site/example:default/:properties/config/prop2 count 2
svc:/site/example:newinst/:properties/config/prop3 count 3
svc:/site/example:newinst/:properties/config/prop1 count 1
svc:/site/example:newinst/:properties/config/prop2 count 2
```

次のコマンドは、config/prop1 と config/prop2 がどちらのサービスインスタンスでもなく、example 親サービスに対して定義されていることを示しています。svcprop は合成ビュー (インスタンスに対して定義されたプロパティーと継承されたプロパティー) を表示するため、前の svcprop コマンドは 3 つのプロパティーをすべて表示します。svccfg コマンドは、指定されたサービスインスタンスまたは親サービスのプロパティーのみを表示します。

```
$ svccfg -s example listprop config
config       application
config/prop1 count       1
config/prop2 count       2
$ svccfg -s example:default listprop config
$ svccfg -s example:newinst listprop config
config       application
config/prop3 count       3
```

例 8　　編集ビューでのサービスおよびインスタンス値の一覧表示

-C および -c オプションを指定すると、svcprop コマンドは、実行中のスナップショットの代わりに編集ビューを表示します。編集ビューには最新の変更が表示されます。編集ビューでの変更は、インスタンスのリフレッシュによって、実行中のスナップショットにコミットされている場合もされていない場合もあります。次のコマンドは、実行中のスナップショットと編集ビューとの違いを示します。

次のコマンドは、example:default サービスインスタンスのプロパティーの値を変更します。

```
$ svccfg -s example:default setprop config/prop1 = 11
```

インスタンスがリフレッシュされていないため、この変更は表示されません。

```
$ svcprop -p config example
svc:/site/example:default/:properties/config/prop1 count 1
svc:/site/example:default/:properties/config/prop2 count 2
svc:/site/example:newinst/:properties/config/prop3 count 3
svc:/site/example:newinst/:properties/config/prop1 count 1
```

```
svc:/site/example:newinst/:properties/config/prop2 count 2
```

`-C` オプションは、編集ビューの新しい値を表示します。

```
$ svcprop -C -p config example
svc:/site/example:default/:properties/config/prop1 count 11
svc:/site/example:newinst/:properties/config/prop3 count 3
```

`-C` オプションは、インスタンスに直接接続されたプロパティーを合成せずに表示します。インスタンスプロパティーの最新の値が表示され、継承されたプロパティーは表示されません。

`-c` オプションは、直接接続されたプロパティーの合成ビューを表示します。インスタンスプロパティーと継承されたプロパティーの両方の最新の値が表示されます。

```
$ svcprop -c -p config example
svc:/site/example:default/:properties/config/prop1 count 11
svc:/site/example/:properties/config/prop2 count 2
svc:/site/example:newinst/:properties/config/prop3 count 3
svc:/site/example/:properties/config/prop1 count 1
svc:/site/example/:properties/config/prop2 count 2
```

次のコマンドは、`example` 親サービスのプロパティーの値を変更します。

```
$ svccfg -s example setprop config/prop2 = 22
```

`-C` オプションはインスタンスに直接接続されたプロパティーを表示するため、親サービスプロパティーへのこの変更は `-C` オプションでは表示されません。

```
$ svcprop -C -p config example
svc:/site/example:default/:properties/config/prop1 count 11
svc:/site/example:newinst/:properties/config/prop3 count 3
```

`-c` オプションは合成ビューを表示するため、親サービスプロパティーへのこの変更は `-c` オプションで表示されます。

```
$ svcprop -c -p config example
svc:/site/example:default/:properties/config/prop1 count 11
svc:/site/example/:properties/config/prop2 count 22
svc:/site/example:newinst/:properties/config/prop3 count 3
svc:/site/example/:properties/config/prop1 count 1
svc:/site/example/:properties/config/prop2 count 22
```

次の例では、`example:default` インスタンスがリフレッシュされているため、そのインスタンスの新しい値が表示されます。新しい `config/prop2` 値をそのインスタンスの有効な値にするには、`example:newinst` インスタンスもリフレッシュする必要があります。

```
$ svcadm refresh example:default
$ svcprop -p config example
svc:/site/example:default/:properties/config/prop1 count 11
svc:/site/example:default/:properties/config/prop2 count 22
svc:/site/example:newinst/:properties/config/prop3 count 3
svc:/site/example:newinst/:properties/config/prop1 count 1
svc:/site/example:newinst/:properties/config/prop2 count 2
```

`svccfg` コマンドはデフォルトで、実行中のスナップショットの値ではなく、編集中のプロパティー値を表示します。59 ページの「指定のスナップショットでの

値の表示」で説明しているように selectsnap サブコマンドを使用することによって、svccfg で、実行中のスナップショットの値を強制的に表示させることができます。

svccfg コマンドは、親サービスを指定した場合は親サービスの値だけを、インスタンスを指定した場合はインスタンスの値だけを表示します。svccfg listprop コマンドの出力がない場合は、指定したプロパティーが、指定した親サービスまたはインスタンスに対して設定されていない可能性があります。プロパティーが削除された場合は、59 ページの「構成カスタマイズの表示」で説明しているように、listcust -M を使用して、マスクされた値を表示します。

次のコマンドは、プロパティーグループまたはプロパティー名が指定されていないので、指定されたサービスの編集中のプロパティー値をすべて一覧表示します。svcprop svc:/system/identity コマンドによって表示された出力に加え、この出力には、プロパティーグループの名前およびタイプとテンプレートデータが含まれます。

```
$ svccfg -s svc:/system/identity listprop
```

次のコマンドは、指定したサービスインスタンスの編集中のプロパティー値をすべて一覧表示します。このコマンドは合成ビューを表示しないので、この出力には、プロファイルファイルおよびマニフェストファイルへのパスなどは表示されません。

```
$ svccfg -s svc:/system/identity:domain listprop
```

例 9 編集ビューでの指定のプロパティーまたはプロパティーグループの一覧表示

次のコマンドは、指定したサービスインスタンスの指定したプロパティーグループでの編集中のプロパティー値をすべて一覧表示します。-o オプションを使用すると、表示する列を選択できます。有効な列名のリストについては、svccfg(8) のマニュアルページを参照してください。

```
$ svccfg -s pkg/server:s11 listprop pkg
pkg                  application
pkg/inst_root       astring      /var/share/pkg/repositories/solaris
pkg/port            count        81
$ svccfg -s pkg/server:s11 listprop -o propname,value pkg
inst_root      /var/share/pkg/repositories/solaris
port           81
```

例 10 名前に特殊文字が含まれているプロパティーの一覧表示

『Oracle Solaris 12 でのシステムサービスの開発』 の 「Naming Property Groups and Properties」には、プロパティーグループやプロパティー名で使用できる予約文字のリストが記載されています。これらの予約文字は、FMRI でエンコードされて表示されます。

```
$ svcprop -p config enchars_example:default
config/%25%20increase count 10
```

```
config/maximum%20%23 count 9
config/start%3Aend count 10
config/students%2Fteachers count 20
```

名前に予約文字が含まれているプロパティーグループまたはプロパティーを照会するには、エンコードされた名前をコピーするか、または `-G` および `-P` オプションを使用します。

```
$ svcprop -p config/%25%20increase enchars_example:default
10
$ svcprop -G config -P students/teachers enchars_example:default
20
$ svccfg -s enchars_example:default listprop -G config -P "maximum #"
config/maximum%20%23 count       9
```

入れ子になったプロパティーグループ内のプロパティーの表示

別のプロパティーグループの子であるプロパティーグループ内のプロパティーまたはプロパティー値を一覧表示するには、そのプロパティーのすべての先祖を指定します。

次の例では、`http` と `https` は `config` プロパティーグループの子プロパティーグループであり、`ssl` は `https` プロパティーグループの子プロパティーグループです。

```
$ svccfg -s npg_example listprop config
config                    application
config/port               count       80
config/http               application
config/http/port          count       80
config/https              application
config/https/port         count       443
config/https/ssl          application
config/https/ssl/certfile astring     cert.crt
config/https/ssl/keyfile  astring     key.crt
```

前の例にあるように、`svcprop` コマンドに最上位のプロパティーグループのみを指定すると、そのプロパティーグループに直接接続されたプロパティーのみが表示されます。子プロパティーグループのプロパティーは表示されません。

```
$ svcprop -p config npg_example
config/port count 80
```

例 11 子プロパティーグループ内のプロパティーの一覧表示

`svcprop` コマンドの `-p` オプションを使用して、入れ子になったプロパティーグループ内のプロパティーを表示する場合は、すべての親プロパティーグループを含める必要があります。次のコマンドは、`config/https/ssl` プロパティーグループのすべてのプロパティーを一覧表示します。

```
$ svcprop -p config/https/ssl svc:/site/npg_example
```

```
config/https/ssl/certfile astring cert.crt
config/https/ssl/keyfile astring key.crt
```

svcprop コマンドの -G オプションを使用して、入れ子になったプロパティーグループ内のプロパティーを表示する場合は、各プロパティーグループに対して個別の -G オプションを指定します。

```
$ svcprop -G config npg_example
config/port count 80
$ svcprop -G config -G https npg_example
config/https/port count 443
$ svcprop -G config -G https -G ssl npg_example
config/https/ssl/certfile astring cert.crt
config/https/ssl/keyfile astring key.crt
```

このセクションの最初に示したように、svccfg listprop コマンドは、指定されたプロパティーグループのすべての子プロパティーグループ内のプロパティーを表示します。

```
$ svccfg -s npg_example listprop config/https
config/https              application
config/https/port         count       443
config/https/ssl          application
config/https/ssl/certfile astring     cert.crt
config/https/ssl/keyfile  astring     key.crt
```

例 12 子プロパティーグループ名の代わりのワイルドカードの使用

svccfg listprop コマンドでの子プロパティーグループ名を置き換えるためにワイルドカードを使用できます。

```
$ svccfg -s npg_example listprop config/*port
config/port       count       80
config/http/port  count       80
config/https/port count       443
```

svcprop コマンドでの子プロパティーグループ名を置き換えるためにワイルドカードを使用することはできません。

```
$ svcprop -p config/*port npg_example
svcprop: Couldn't find property group `config/*port' for instance
`svc:/site/npg_example:default'.
```

例 13 子プロパティーグループ内のプロパティー値の表示

svcprop コマンドを使用する場合は、すべての親プロパティーグループ名を指定します。

```
$ svcprop -p config/https/ssl/certfile npg_example
cert.crt
```

svcprop コマンドで -G オプションを使用する場合は、-P オプションを使用してプロパティー名を指定する必要があります。

```
$ svcprop -G config -G https -G ssl -P certfile npg_example
cert.crt
```

svccfg コマンドを使用する場合は、すべての親プロパティーグループ名を指定するか、またはワイルドカードを使用できます。

```
$ svccfg -s npg_example listprop config/https/ssl/certfile
config/https/ssl/certfile astring     cert.crt
$ svccfg -s npg_example listprop config/*file
config/https/ssl/certfile astring     cert.crt
config/https/ssl/keyfile  astring     key.crt
```

プロパティーグループタイプでのプロパティーの表示

プロパティー名またはプロパティーグループ名でプロパティー値を表示するだけでなく、プロパティーグループタイプでプロパティー値を表示することもできます。

例 14 プロパティーグループとそのタイプの表示

svccfg コマンドの listpg サブコマンドは、各プロパティーグループの名前とタイプを表示します。

```
$ svccfg -s pkg/server listpg
pkg              application
pkg_bui          application
pkg_secure       application
fs               dependency
autofs           dependency
ntp              dependency
network          dependency
general          framework
manifestfiles    framework
start            method
stop             method
tm_common_name   template
$ svccfg -s pkg/server:s11 listpg
pkg        application
general    framework
restarter  framework              NONPERSISTENT
```

非永続プロパティーグループは通常、アクティブなプログラム状態を保持します。非永続プロパティーグループ内のプロパティーの値は、システムブート中にクリアされます。

プロパティーグループ名を指定すると、そのプロパティーグループのタイプだけが表示されます。

```
$ svccfg -s pkg/mirror listpg config
config  application
```

例 15 プロパティーグループタイプのプロパティーの一覧表示

-g オプションを付けて svcprop コマンドを実行すると、特定のプロパティーグループタイプのプロパティーが表示されます。

```
$ svcprop -g com.sun,fw_configuration smtp
firewall_config/apply_to astring ""
firewall_config/exceptions astring ""
firewall_config/policy astring use_global
firewall_config/value_authorization astring solaris.smf.value.firewall.config
```

複数の -g オプションが指定されると、指定されたすべてのタイプのプロパティーグループのプロパティーが表示されます。

```
$ svcprop -g application -g com.sun,fw_configuration smtp
config/db_version integer 5
config/include_info boolean false
config/local_only boolean true
config/path_to_sendmail_mc astring ""
config/value_authorization astring solaris.smf.value.sendmail
firewall_config/apply_to astring ""
firewall_config/exceptions astring ""
firewall_config/policy astring use_global
firewall_config/value_authorization astring solaris.smf.value.firewall.config
```

-p および -g の両方のオプションを使用する場合、-p オプション値でプロパティーグループの名前を指定しないでください。

```
$ svcprop -g plugin -p path auditd
audit_binfile/path astring audit_binfile.so
audit_remote/path astring audit_remote.so
audit_sstore/path astring audit_sstore.so.1
audit_syslog/path astring audit_syslog.so
```

値が設定されているレイヤーの表示

サービス構成リポジトリは、データのソースに応じて、レイヤーにプロパティーデータを格納します。svcprop と svccfg のどちらのコマンドでも、プロパティー値のソースであるレイヤーを表示できます。svcprop コマンドおよび svccfg コマンドの -l オプションには、情報を表示するレイヤーを指定する引数が必要です。引数値は、manifest、system-profile、enterprise-profile、site-profile、node-profile、sysconfig-profile、および admin です。出力には、特定のプロパティー値の設定がサービスマニフェストで行われたか、プロファイルで行われたか、管理者またはアプリケーションによって行われたかが示されます。レイヤーについては、30 ページの「リポジトリレイヤー」を参照してください。キーワード all は、すべてのレイヤーを表す別名です。指定したレイヤーが必要なプロパティー値のソースでない場合、出力は表示されません。

次のコマンドは、プロパティー値の一部がサービスマニフェストからのものであり、一部は管理者が設定したものであることを示します。プロパティーの一部は複数のレイヤーに値があります。pkg/readonly プロパティーは、サービスマニフェストに値が設定されており、管理者もその同じ値を設定します。別々のレイヤーの値は異なる可能性があります。

```
$ svcprop -l all -p pkg pkg/server:s11
pkg/port count admin 81
pkg/inst_root astring admin /var/share/pkg/repositories/solaris
pkg/address net_address manifest
```

```
pkg/cfg_file astring manifest ""
...
pkg/readonly boolean manifest true
pkg/readonly boolean admin true
...
```

`svccfg listprop` コマンドの `-l` オプションは、引数 `current` を取ることもできます。`current` を `-l` の引数として使用すると、`-l` オプションを使用しない場合に表示されるものと同じプロパティー値が表示されます。出力における唯一の違いは、レイヤーの名前も表示される点です。サービス構成リポジトリは非永続データのレイヤー情報を格納しないので、非永続データはレイヤー名を表示しません (3 番目の列には `<none>` と表示されます)。非永続プロパティーグループには通常、アクティブなプログラム状態が保持され、非永続プロパティーグループ内のプロパティーの値はシステムブート中にクリアされます。

```
$ svccfg -s pkg/server:s11 listprop -l current
pkg                              application admin
pkg/inst_root                    astring     admin     /var/share/pkg/repositories/solaris
pkg/port                         count       admin     81
general                           framework   admin
general/complete                 astring     manifest
general/enabled                  boolean     admin     true
restarter                         framework   <none>        NONPERSISTENT
restarter/logfile                astring      <none>    /var/svc/log/application-pkg-
server:default.log
restarter/contract               count        <none>    121
restarter/start_pid              count        <none>    1055
restarter/start_method_timestamp  time         <none>    1379605275.329096000
restarter/start_method_waitstatus integer      <none>    0
restarter/auxiliary_state        astring      <none>    dependencies_satisfied
restarter/next_state             astring      <none>    none
restarter/state                  astring      <none>    online
restarter/state_timestamp        time         <none>    1379605275.332259000
```

構成に関係したファイルの表示

次のコマンドは、`localtime` プロパティーがサービスマニフェストでは UTC に設定され、`/etc/svc/profile/node` ディレクトリ内のプロファイルでは US/Pacific に設定されていることを示しています。`node-profile` レイヤーで設定された値は、`manifest` レイヤーで設定された値をオーバーライドします。

```
$ svcprop -l all -p timezone/localtime system/timezone:default
timezone/localtime astring manifest UTC
timezone/localtime astring node-profile US/Pacific
$ svccfg -s system/timezone:default listprop -l all -o propname,layer,value \
> timezone/localtime
localtime          node-profile   US/Pacific
localtime          manifest       UTC
```

`svccfg listprop` コマンドの `-f` オプションまたは `-o file` オプションのいずれかを使用して、構成に関係したファイルの名前を表示します。

```
$ svccfg -s system/timezone:default listprop -l all -f timezone/localtime
localtime          /etc/svc/profile/node/migrated_etc_svc_profile_site_sc_profile.xml US/
Pacific
```

```
localtime          /lib/svc/manifest/system/timezone.xml UTC
$ svccfg -s system/timezone:default listprop -l all -o propname,value,file \
> timezone/localtime
localtime          US/Pacific /etc/svc/profile/node/
migrated_etc_svc_profile_site_sc_profile.xml
localtime          UTC /lib/svc/manifest/system/timezone.xml
```

指定のスナップショットでの値の表示

次のコマンドは、このサービスインスタンスで利用できるスナップショットを一覧表示します。`svcprop` または `svccfg` のどちらかでこれらのスナップショット名を使用すると、そのスナップショットで設定されたプロパティーの値が表示されます。スナップショットはインスタンスだけにあります。サービスにはスナップショットはありません。サービス構成リポジトリのスナップショットの詳細は、32 ページの「リポジトリのスナップショット」を参照してください。

```
$ svccfg -s pkg/server:default listsnap
initial
previous
running
start
$ svccfg -s pkg/server:s11 listsnap
previous
running
start
```

次のコマンドは、`pkg/inst_root` プロパティーの値が以前のスナップショットで異なっていたことを示します。

```
$ svcprop -s previous -p pkg/inst_root pkg/server:s11
/var/share/pkg/repositories/solaris
$ svccfg -s pkg/server:s11
svc:/application/pkg/server:s11> selectsnap previous
[previous]svc:/application/pkg/server:s11> listprop pkg/inst_root
pkg/inst_root astring     /var/share/pkg/repositories/solaris
[previous]svc:/application/pkg/server:s11> exit
```

構成カスタマイズの表示

`svccfg listcust` コマンドは、指定されたサービスの `admin` レイヤーでのカスタマイズを表示します。`enterprise-profile`、`site-profile`、`node-profile`、および `sysconfig-profile` レイヤー内のカスタマイズも表示するには、`-L` オプションを使用します。

次のコマンドは、`pkg/server:solaris` サービスの `admin` レイヤーでのすべてのカスタマイズを示しています。

```
$ svccfg -s pkg/server:solaris listcust
```

```
general                         framework   admin
general/complete                astring     admin
general/enabled                 boolean     admin               true
pkg                             application admin
pkg/inst_root                   astring     admin               /var/share/pkgrepos/
solaris
pkg/port                        count       admin               83
pkg/readonly                    boolean     admin               true
pkg/standalone                  boolean     admin               false
```

次のコマンドは、プロパティー `config/nodename` の定義は `manifest` レイヤーで指定されているが、値 `solaris` は `node-profile` レイヤーで設定されていることを示しています。

```
$ svccfg -s identity:node listprop -l all -o propname,layer,value config/nodename
nodename        node-profile   solaris
nodename        manifest
```

次のコマンドは、`identity:node` サービスの `admin` レイヤーのカスタマイズのみ示します。

```
$ svccfg -s identity:node listcust
config/loopback   astring     admin                solaris
```

次のコマンドは、`identity:node` サービスのすべてのカスタマイズを示しています。

```
$ svccfg -s identity:node listcust -L
config            application node-profile
config/loopback   astring     admin                solaris
config/nodename   astring     node-profile         solaris
general           framework   node-profile
general/enabled   boolean     node-profile         true
```

`svccfg listcust` コマンドは、すべてのマスクされたエンティティーも表示します。マスクされたエンティティーだけを表示するには、`-M` オプションを使用します。`svccfg delcust` コマンドを使用する前に、`svccfg listcust` コマンドを使用して、削除される対象を検証します。マスクされたエンティティーの詳細は、92 ページの「プロパティーグループ、プロパティー、およびプロパティー値の削除」と`smf`(7)のマニュアルページを参照してください。

イベント通知パラメータの表示

`svcs -n` コマンドは、FMA イベント通知パラメータ、システム全体の SMF 状態遷移通知パラメータ、およびサービスインスタンス状態遷移通知パラメータを表示します。これらのパラメータの詳細は、`smf`(7) のマニュアルページの「通知パラメータ」を参照してください。

```
$ svcs -n
Notification parameters for FMA Events
    Event: problem-diagnosed
        Notification Type: smtp
            Active: true
```

```
            reply-to: root@localhost
            to: root@localhost

        Notification Type: snmp
            Active: true

        Notification Type: syslog
            Active: true

    Event: problem-repaired
        Notification Type: snmp
            Active: true

    Event: problem-resolved
        Notification Type: snmp
            Active: true

System wide notification parameters:
svc:/system/svc/global:default:
    Event: to-maintenance
        Notification Type: smtp
            Active: true
            to: sysadmins@example.com

svc:/application/pkg/mirror:default:
    Event: to-maintenance
        Notification Type: smtp
            Active: true
            to: installteam@example.com
```

`problem-diagnosed`、`problem-repaired`、および `problem-resolved` の 3 つの FMA イベントが表示されます。4 番目のイベント `problem-updated` に関する通知パラメータも構成できます。

システム全体の状態遷移通知設定の場合、これらのグローバル設定を格納するサービスも一覧表示されます。このシステム全体の設定はカスタム設定です。システム全体の値、つまりグローバル値は、カスタム値が設定されていないすべてのサービスインスタンスに適用されます。

表示される最後の設定は、特定のサービスインスタンスのカスタム設定です。

指定のイベントだけに関する通知パラメータを表示するには、`svccfg listnotify` コマンドを使用します。状態遷移イベントの場合は、`-g` オプションを使用して、グローバル設定を表示します。出力には、通知パラメータ値のソースも表示されます。

```
$ svccfg listnotify problem-resolved
    Event: problem-resolved (source: svc:/system/fm/notify-params:default)
        Notification Type: snmp
            Active: true
$ svccfg listnotify -g to-maintenance
    Event: to-maintenance (source: svc:/system/svc/global:default)
        Notification Type: smtp
            Active: true
            to: sysadmins@example.com
$ svccfg -s pkg/mirror listnotify to-maintenance
    Event: to-maintenance (source: svc:/application/pkg/mirror)
        Notification Type: smtp
            Active: true
            to: installteam@example.com
```

イベント通知の構成の詳細は、75 ページの「状態遷移および FMA イベントの通知の構成」を参照してください。

♦ ♦ ♦ 第 3 章

3 サービスの管理

この章では、次について説明します。

- サービスを起動、停止および再起動する方法
- サービス構成を再読み取りする方法
- FMA イベントやサービス状態の遷移について通知するようにシステムを構成する方法

サービス状態を変更するコマンドは、`svcadm` です。`svcadm` コマンドはサービスインスタンスで動作します。インスタンスを指定せずにサービス名を入力し、そのサービスのインスタンスが 1 つしかない場合、`svcadm` はそのインスタンスで動作します。インスタンスを指定せずにサービス名を入力し、そのサービスのインスタンスが複数ある場合、または複数のインスタンスに一致するほかのパターンを指定する場合は、`svcadm` はエラーメッセージを発行します。

SMF サービスインスタンスの管理

サービスインスタンスは必ず、24 ページの「サービス状態」に記した状態のいずれかになります。このセクションでは、インスタンスを別の状態に遷移させる方法、更新済みのプロパティー値を実行中のスナップショットにコミットする方法、および通常のビューからインスタンスを削除する方法について説明します。

サービスの起動

`degraded`、`maintenance`、`offline`、`online` のいずれかの状態のサービスインスタンスはすでに有効になっており、起動する必要はありません。起動するインスタンスが `degraded`、`maintenance`、または `offline` の状態になっている場合は、113 ページの「機能低下、オフライン、または保守であるインスタンスの修復」を参照してください。起動するインスタンスが `disabled` 状態になっている場合は、次の手順に示すようにインスタンスを有効にします。インスタンスを有効にする

と、そのインスタンスのリスタータが online 状態にそのインスタンスを遷移させようと試みます。

▼ サービスインスタンスを有効にする方法

1. **インスタンス状態と依存関係を確認します。**

   インスタンスが現在無効になっていること、およびその必要な依存関係のすべてが実行していること (online または degraded 状態) を確認します。

   ```
   $ svcs -l FMRI
   ```

2. **インスタンスを有効にします。**

   サービスのリスタータは、指定のインスタンスを online 状態にしようと試みます。

   インスタンスは永続的に有効にすることも、一時的に有効にすることもできます。永続的な有効はシステムリブートのあとも持続し、これがデフォルトです。一時的な有効はリブートまでに限り継続します。

   - **永続的にインスタンスを有効にします。**

     ```
     $ svcadm enable FMRI
     ```

   - **一時的にインスタンスを有効にします。**

     一時的な有効を指定するには、-t オプションを使用します。

     ```
     $ svcadm enable -t FMRI
     ```

     現時点ではインスタンスを実行するが、次回のリブート時には実行しない場合は、インスタンスが無効になっていることを確認し、一時的にインスタンスを有効にします。インスタンスが一時的に有効になっていることを検証するには、svcs -l コマンドを使用して、enabled 行が次のようになっていることを確認します。

     ```
     enabled      true (temporary)
     ```

   - **同期的にインスタンスを有効にします。**

     -s オプションを指定すると、svcadm はインスタンスを有効にし、インスタンスが online または degraded 状態になるまで待機してから戻ります。svcadm コマンドは、インスタンスがオンライン状態に達したとき、またはインスタンスがオンライン状態に達するには管理者の操作が必要だと判断したときに戻ります。

     遷移を行うか、遷移を行えないと判断するまでの上限を秒単位で指定するには、-T オプションを -s オプションとともに使用します。

     ```
     $ svcadm enable -sT 10 FMRI
     ```

3. **インスタンスがオンラインであることを確認します。**

```
$ svcs FMRI
```

インスタンスが degraded、maintenance、または offline の状態になっている場合は、113 ページの「機能低下、オフライン、または保守であるインスタンスの修復」を参照してください。

例 16　サービスインスタンスの永続的な有効化

次のコマンドは、pkg/mirror:default サービスインスタンスが現在無効になっており、その必要な依存関係がすべてオンラインになっていることを示します。

```
$ svcs -l pkg/mirror
fmri         svc:/application/pkg/mirror:default
name         IPS Repository Mirror
enabled      false
state        disabled
next_state   none
state_time   September 17, 2013 07:16:52 AM PDT
restarter    svc:/system/svc/restarter:default
manifest     /lib/svc/manifest/application/pkg/pkg-mirror.xml
dependency   require_all/error svc:/milestone/network:default (online)
dependency   require_all/none svc:/system/filesystem/local:default (online)
dependency   optional_all/error svc:/system/filesystem/autofs:default (online)
dependency   require_all/none svc:/application/pkg/repositories-setup (online)
```

次のコマンドは、pkg/mirror:default インスタンスを有効にします。この場合、pkg/mirror:default インスタンスが正常に有効になるので、svcadm コマンドは戻ります。

```
$ svcadm enable -sT 10 pkg/mirror:default
$ svcs pkg/mirror
STATE          STIME    FMRI
online         22:03:53 svc:/application/pkg/mirror:default
```

例 17　サービスインスタンスの一時的な有効化

次のコマンドは、net-snmp:default サービスインスタンスが現在無効になっており、その必要な依存関係がすべてオンラインになっていることを示します。無効になっている 1 つの依存関係は、オプションの依存関係です。

```
$ svcs -l net-snmp
fmri         svc:/application/management/net-snmp:default
name         net-snmp SNMP daemon
enabled      false
state        disabled
next_state   none
state_time   September 17, 2013 05:56:39 PM PDT
logfile      /var/svc/log/application-management-net-snmp:default.log
restarter    svc:/system/svc/restarter:default
contract_id
manifest     /etc/svc/profile/generic.xml
manifest     /lib/svc/manifest/application/management/net-snmp.xml
dependency   require_all/none svc:/system/filesystem/local (online)
dependency   optional_all/none svc:/milestone/name-services (online)
dependency   optional_all/none svc:/system/system-log (online)
dependency   optional_all/none svc:/network/rpc/rstat (disabled)
dependency   require_all/restart svc:/system/cryptosvc (online)
```

```
dependency   require_all/restart svc:/milestone/network (online)
dependency   require_all/refresh file://localhost/etc/net-snmp/snmp/snmpd.conf (online)
dependency   require_all/none svc:/milestone/multi-user (online)
```

次の例に示すように `-t` オプションを使用してインスタンスを有効にしたあと、`svcs -p` コマンドで示されるように `snmpd` プロセスが開始されているので、インスタンスは一時的に有効になり、オンラインになり、契約 ID が与えられます。

```
$ svcadm enable -t net-snmp:default
$ svcs -l net-snmp
fmri         svc:/application/management/net-snmp:default
name         net-snmp SNMP daemon
enabled      true (temporary)
state        online
next_state   none
state_time   September 17, 2013 05:57:26 PM PDT
logfile      /var/svc/log/application-management-net-snmp:default.log
restarter    svc:/system/svc/restarter:default
contract_id  160
manifest     /etc/svc/profile/generic.xml
manifest     /lib/svc/manifest/application/management/net-snmp.xml
dependency   require_all/none svc:/system/filesystem/local (online)
dependency   optional_all/none svc:/milestone/name-services (online)
dependency   optional_all/none svc:/system/system-log (online)
dependency   optional_all/none svc:/network/rpc/rstat (disabled)
dependency   require_all/restart svc:/system/cryptosvc (online)
dependency   require_all/restart svc:/milestone/network (online)
dependency   require_all/refresh file://localhost/etc/net-snmp/snmp/snmpd.conf (online)
dependency   require_all/none svc:/milestone/multi-user (online)
$ svcs -p net-snmp
STATE          STIME    FMRI
online         17:57:26 svc:/application/management/net-snmp:default
               17:57:26     5022 snmpd
```

サービスの停止

有効になっている、または一時的に無効になっているサービスインスタンスを無効にするには、`svcadm disable` コマンドを使用します。無効になっているインスタンスは再起動できません。最初にインスタンスを有効にする必要があります。

注記 - セキュリティーのベストプラクティスでは、オンラインサービスを定期的に確認して、使用されなくなったり実行が不要になったものがないかどうか評価することが推奨されています。

▼ サービスインスタンスを無効にする方法

1. **ほかのサービスがこのインスタンスに依存しているかどうかを確認します。**

   a. **このインスタンスに依存するサービスを一覧表示します。**

      ```
      $ svcs -D FMRI
      ```

b. 依存したサービスがこのインスタンスを必要としているかどうかを確認します。

`svcs -D` コマンドの結果ごとに `svcs -l` コマンドを使用して、依存関係が必要な依存関係であるかどうかを確認します。

このインスタンスが別のサービスの必要な依存関係になっている場合は、このインスタンスを無効にしないでください。依存関係のグループと `restart_on` 値については、41 ページの「サービス依存関係の表示」を参照してください。

2. インスタンスを無効にします。

サービスのリスタータが、指定のインスタンスを `disabled` 状態にしようと試みます。停止メソッドが存在する場合、サービスのリスタータは通常、停止メソッドを実行しようとします。定期的なインスタンスによって契約されたプロセスは永続的に実行されないため、定期的なリスタータはどの停止メソッドも実行しようとしません。定期的なインスタンスは短期のプロセスを実行したあとで、次にスケジュールされた実行時間まで待機します。詳細は、『Oracle Solaris 12 でのシステムサービスの開発』の 第 3 章,「Creating a Service to Run Periodically」を参照してください。

インスタンスは永続的に無効にすることも、一時的に無効にすることもできます。永続的な無効はシステムリブートのあとも持続し、これがデフォルトです。一時的な無効はリブートまでに限り継続します。

- **永続的にインスタンスを無効にします。**

```
$ svcadm disable FMRI
```

- **一時的にインスタンスを無効にします。**

一時的な無効を指定するには、`-t` オプションを使用します。

```
$ svcadm disable -t FMRI
```

現時点ではインスタンスを無効にするが、次回のリブート時に実行する場合は、インスタンスが実行していること (`online` または `degraded` の状態) を確認し、一時的にインスタンスを無効にします。インスタンスが一時的に無効になっていることを検証するには、`svcs -l` コマンドを使用して、`enabled` 行が次のようになっているか確認します。

```
enabled      false (temporary)
```

- **同期的にインスタンスを無効にします。**

`-s` オプションを指定すると、`svcadm` はインスタンスを無効にし、インスタンスが `disabled` 状態になるまで待機してから戻ります。`svcadm` コマンドは、インスタンスが `disabled` 状態に達したとき、またはインスタンスが `disabled` 状態に達するには管理者の操作が必要だと判断したときに戻ります。

遷移を行うか、遷移を行えないと判断するまでの上限を秒単位で指定するには、`-T` オプションを `-s` オプションとともに使用します。

```
$ svcadm disable -sT 10 FMRI
```

3. **インスタンスが無効になっていることを検証します。**

```
$ svcs FMRI
```

例 18 サービスインスタンスの無効化

次のコマンドは、Generic Security Service `rpc/gss` がオンライン状態で、`rpc/gss` サービスに依存しているサービスが無効化されていることを示します。

```
$ svcs rpc/gss
STATE          STIME    FMRI
online         Oct_20   svc:/network/rpc/gss:default
$ svcs -D rpc/gss
STATE          STIME    FMRI
disabled       Oct_20   svc:/network/nfs/client:default
disabled       Oct_20   svc:/network/smb/client:default
disabled       Oct_21   svc:/network/nfs/server:default
```

次のコマンドは、依存関係がオンラインの場合であっても、`rpc/gss` サービスはオプションの依存関係であるため、これら 3 つのいずれかの依存サービスが何らかの理由でリフレッシュまたは停止された場合に `rpc/gss` サービスを開始する試行は実行されないことを示しています。

```
$ svcs -l nfs/client smb/client nfs/server | grep rpc/gss
dependency   optional_all/none svc:/network/rpc/gss (online)
dependency   optional_all/none svc:/network/rpc/gss (online)
dependency   optional_all/none svc:/network/rpc/gss (online)
```

`svcadm disable` コマンドが成功し、インスタンスは現在 `disabled` 状態になり、再起動しても失敗します。

```
$ svcadm disable rpc/gss
$ svcs rpc/gss
STATE          STIME    FMRI
disabled       12:45:55 svc:/network/rpc/gss:default
$ svcadm restart pkg/update:default
$ svcs rpc/gss
STATE          STIME    FMRI
disabled       12:45:55 svc:/network/rpc/gss:default
```

サービスの再起動

再起動操作は、現在実行している (`online` または `degraded` 状態) インスタンスだけを再起動します。たとえばインスタンスの実行中に有効にできない構成変更を行なったために、実行中のインスタンスの再起動が必要になることがあります。

サービスインスタンスを再起動しても構成はリフレッシュされません。通常、`svcadm restart` コマンドは、インスタンスの停止メソッドを実行したあとで、インスタンス

の起動メソッドを実行します。定期的なリスタータは、起動メソッドのみを実行します。`svcadm restart` コマンドは、実行中のスナップショットにプロパティーの変更をコミットせず、インスタンスのリフレッシュメソッドを実行しません。実行中のスナップショットへの構成変更のコミットの詳細は、70 ページの「サービス構成の再読み取り」を参照してください。

`manifest-import` サービスの再起動は特殊な場合です。`manifest-import` サービスを再起動すると、標準の場所にある変更されたマニフェストまたはプロファイルがインポートされ、サービス構成リポジトリに変更がコミットされ、新しい実行中のスナップショットが取得され、リフレッシュメソッドが存在する場合は変更されたインスタンスのリフレッシュメソッドが実行されます。

▼ サービスインスタンスを再起動する方法

1. **インスタンス状態を確認します。**

   インスタンスは `online` または `degraded` 状態になっている必要があります。

   ```
   $ svcs FMRI
   ```

2. **インスタンスを再起動します。**

   サービスのリスタータは、指定のインスタンスを `online` 状態にしようと試みます。ほとんどのリスタータに実装されている再起動操作は、停止操作のあとに起動操作を続けるというものです。

   - **インスタンスを再起動します。**

     ```
     $ svcadm restart FMRI
     ```

   - **同期的にインスタンスを再起動します。**

     `-s` オプションを指定すると、`svcadm` はインスタンスを再起動し、インスタンスが `online`、`degraded`、または `maintenance` の状態になるまで待機してから戻ります。`svcadm` コマンドは、インスタンスがこれらのいずれかの状態に達したとき、またはインスタンスがこれらのいずれかの状態に達するには管理者の操作が必要だと判断したときに戻ります。

     遷移を行うか、遷移を行えないと判断するまでの上限を秒単位で指定するには、`-T` オプションを `-s` オプションとともに使用します。

     ```
     $ svcadm restart -sT 10 FMRI
     ```

3. **インスタンスが起動されていることを検証します。**

   再起動が成功すると、インスタンスは `online`、`degraded`、または `maintenance` の状態になります。インスタンスが `degraded` または `maintenance` の状態になっている場合は、113 ページの「機能低下、オフライン、または保守であるインスタンスの修復」を参照してください。

```
$ svcs FMRI
```

サービス構成の再読み取り

次の方法を使用してサービス構成を変更できます。

- `svccfg setprop` コマンドを使用します。
- サービスマニフェストファイルを編集します。
- プロファイルを編集するか、サービスに関連付けられた新しいプロファイルを提供します。

サービス構成を変更するか新しい構成を提供する場合、実行中のスナップショットにはこれらの変更は即座に反映されません。構成の変更を適用するには、サービスインスタンスをリフレッシュします。この実装によって、複数のサービスに対して複数の変更を行なって、一度にすべての変更を適用できます。

`svccfg setprop` コマンドを使用して行なった変更を適用するには、`svcadm refresh` または `svccfg refresh` コマンドのどちらかを使用する必要があります。サービスインスタンスは、`refresh` メソッドが必要であり、`online` または `degraded` 状態になっている必要があります。

マニフェストファイルまたはプロファイルの変更または追加によって行われた変更を適用するには、`svcadm refresh` または `svccfg refresh` コマンドのどちらかを使用することも、`manifest-import` サービスを再起動することもできます。サービスインスタンスは `online` または `degraded` 状態になっている必要があります。`manifest-import` サービスは、マニフェストまたはプロファイルが変更されたすべての実行中のインスタンスの構成を更新します。`manifest-import` サービスは、`svccfg setprop` コマンドを使用して行われた構成変更を適用しません。`manifest-import` サービスを手動で再起動することに加えて、システムのブートおよび多くの `pkg` 操作でも `manifest-import` サービスが再起動します。

`svcadm refresh` および `svccfg refresh` コマンドはどちらも次の手順を実行します。

1. 実行中のスナップショットに編集中のプロパティーをコミットする、新しい実行中のスナップショットを作成します。
2. リフレッシュメソッドが存在し、インスタンスが `online` または `degraded` の状態になっている場合に、インスタンスのリフレッシュメソッドを実行します。リフレッシュメソッドは、変更が行われたことをアプリケーションに通知する必要があります。リフレッシュメソッドは、実行中のスナップショットからプロパティー値を再読み取りする場合があります。リフレッシュメソッドが存在しない場合でも、実行中のスナップショット内の構成は更新されます。

定期的なリスタータは、リフレッシュメソッドを実行しようとしません。定期的なインスタンスがリフレッシュされると、定期的なリスタータは、『Oracle Solaris 12 でのシステムサービスの開発』の「Storing Periodic Service Data in the Service Configuration Repository」の説明に従って、`periodic` プロパティーグループのプロパティーの値を再読み取りします。

`svcadm refresh` コマンドはサービスインスタンス上で動作します。`svccfg refresh` コマンドはサービスインスタンスまたは親サービス上で動作します。サービスが指定されている場合、`svccfg refresh` コマンドは、そのサービスのすべてのインスタンスをリフレッシュします。サービスインスタンスのスナップショットだけが取得され親サービスのスナップショットは取得されませんが、親サービスのプロパティーはサービスインスタンスに継承されます。変更された親サービスのプロパティーは、インスタンスがこれらの変更をオーバーライドしない場合、サービスインスタンスのスナップショットに表示されます。

依存関係の変更など、一部の変更は即座に有効になります。その他の変更は、68 ページの「サービスの再起動」に説明しているように、サービスが再起動するまで有効になりません。アプリケーションの実行中に行えない変更には、再起動後のリフレッシュが必要です。アプリケーションの実行中に行えない変更にはたとえば、ソケットの開閉や環境変数のリセットなどがあります。

`svcadm refresh` コマンドとともに `-s` オプションを指定すると、`svcadm` はインスタンスをリフレッシュし、インスタンスが `online`、`degraded`、または `maintenance` の状態になるまで待機してから戻ります。`svcadm` コマンドは、インスタンスがこれらのいずれかの状態に達したとき、またはインスタンスがこれらのいずれかの状態に達するには管理者の操作が必要だと判断したときに戻ります。遷移を行うか、遷移を行えないと判断するまでの上限を秒単位で指定するには、`-T` オプションを `-s` オプションとともに使用します。

サービスの削除

`svccfg delete` コマンドはシステムからサービスインスタンスを削除しません。代わりに、`svccfg delete` コマンドがインスタンスをマスクします。`svccfg delete` コマンドを実行したあとも、サービスマニフェストは `/lib/svc/manifest` に存在します。SMF では、サービス構成リポジトリとファイルシステムコンテンツとの同期は維持されます。マニフェストは引き続き標準の場所のファイルシステムに存在するので、そのサービス情報はそのままリポジトリに格納されており、通常のビューからマスクされるだけです。管理カスタマイズはすべて、マスクされたインスタンスから削除されます。マスクされたエンティティーの説明については、`smf`(7)のマニュアルページを参照してください。

サービスインスタンスをサポートするファイルは、そのサービスインスタンスがマスクされている場合でも、`pkg` コマンドを使用すると更新されます。サービスインスタ

ンスをサポートするファイルが pkg コマンドによって更新されると、サービスがまだビューからマスクされていても、SMF データストアは更新されます。サービスインスタンスがマスク解除されると、そのサービスインスタンスはすでに、pkg によって提供されたファイルから更新されており、これ以上の操作は不要です。サービスインスタンスをマスク解除するには、72 ページの「サービスインスタンスの削除を元に戻す方法」を参照してください。

▼ サービスインスタンスを削除する方法

1. **削除するインスタンスの依存関係を確認します。**

   svcs -D コマンドを使用して、このインスタンスに依存するインスタンスを表示します。このインスタンスを削除したあと、依存したインスタンスが実行できなくなる可能性があります。svcs -l コマンドを使用して、このインスタンスが、依存したインスタンスの必要な依存関係であるかどうかを確認します。

2. **インスタンスをマスクします。**

   svccfg delete コマンドを使用して、通常のビューからインスタンスをマスクします。svcs コマンドを使用して、インスタンスの状態を表示します。インスタンスが実行している場合 (online または degraded の状態)、svccfg delete -f コマンドを使用して、通常のビューからインスタンスをマスクします。

```
$ svcs -H my-svc
disabled        7:25:37 svc:/site/my-svc:default
$ svccfg delete svc:/site/my-svc:default
```

3. **インスタンスがマスクされていることを検証します。**

   svccfg listcust -M コマンドを使用して、インスタンスがマスクされていることを確認します。svcs などのコマンドでは、一致するインスタンスが見つからないというエラーメッセージが表示されます。

```
$ svccfg listcust -M
svc:/site/my-svc:default manifest MASKED
  general                       admin    MASKED
  general/complete astring      admin    MASKED
  general/enabled  boolean      admin    MASKED true
$ svcs -H my-svc
svcs: Pattern 'my-svc' doesn't match any instances
```

▼ サービスインスタンスの削除を元に戻す方法

1. **インスタンスがマスクされていることを確認します。**

   前述の手順で示したように、svccfg listcust -M コマンドを使用します。

2. **インスタンスをマスク解除します。**

```
$ svccfg -s svc:/site/my-svc:default delcust
```

```
Deleting customizations for instance: default
```

マニフェストを再インポートしてもマスクは削除されません。

3. **インスタンスがマスク解除されていることを検証します。**

   `svccfg listcust -M` コマンドを使用して、インスタンスがマスクされていないことを確認します。`svcs` コマンドでは、インスタンスの状態が表示されます。

目標サービスの目標の変更

目標サービスは、特定の機能を実現するために実行する必要のあるサービスのセットを定義します。この重要なセット内のサービスは、目標サービスの依存関係として指定されます。

目標サービスの依存関係を設定するには、次のいずれかの方法を使用します。

- `svcadm goals` コマンドを使用します。
- プロファイルファイルを変更します。

目標サービスの依存関係セットに推奨されるサービスのタイプについては、『Oracle Solaris 12 でのシステムサービスの開発』 の 第 7 章, 「Creating a Service that Notifies if Conditions are not Satisfied」を参照してください。

`svcadm goals` コマンドの使用

`svcadm goals` コマンドは、目標サービスが依存するサービスのリストを取ります。目標サービスの依存関係のセットが、`svcadm goals` コマンドのオペランドとして指定されるサービスのセットになります。依存関係を追加するには、新しい依存関係だけでなく、現在のすべての依存関係をリストする必要があります。

`svcadm goals` コマンドのオペランドとして指定されるサービスは目標サービスの管理カスタマイズ、または `admin` レイヤーの変更です。レイヤーの詳細は、30 ページの「リポジトリレイヤー」を参照してください。`svcadm goals` コマンドは、目標サービスの依存関係セットに対するすべての管理カスタマイズをリセットします。

`svcadm goals` コマンドを使用して設定された依存関係は `require_all` 依存関係グループ内にあり、`restart_on` 属性の値は `restart` です。詳細は、42 ページの「依存関係のグループ」および 44 ページの「サービスが自動的に再起動するかどうかの表示」を参照してください。目標サービスの依存関係が別のグループまたは別の `restart_on` 設定になるようにする場合は、`svccfg` コマンドを使用してその依存関

係の属性を変更します。たとえば、目標サービスの依存関係として動的に有効になるサービスを含める必要がある場合は、svccfg コマンドを使用して、その依存関係の grouping 属性の値を optional_all に変更します。

例 19 目標サービスの依存関係セットのリセット

次のコマンドは、milestone/goals 目標サービスの依存関係セットを、指定された database/oracle、database/listener、および milestone/multi-user-server サービスに置き換えます。milestone/multi-user-server サービスは、manifest レイヤーで指定されるデフォルトの目標です。svcadm goals コマンドに multi-user-server を含めない場合、multi-user-server はマスクされ、目標として使用されません。svcadm goals コマンドは、現在設定されているすべての目標を置き換えます。デフォルトでは、svcadm goals コマンドは milestone/goals サービスに対して動作します。milestone/goals サービスではない目標サービスの依存関係セットを置き換えるには、-g オプションを使用して目標サービスを指定します。

```
$ svcadm goals site/application/database/oracle:default \
> site/application/database/listener:default \
> svc:/milestone/multi-user-server:default
```

次のどちらのコマンドも、milestone/goals サービスには現在 3 つの目標の依存関係があることを示します。

```
$ svcprop -p goals-dependencies/entities milestone/goals
$ svcs -d milestone/goals
```

次のどちらのコマンドも、目標サービスの依存関係セットのリセットが管理カスタマイズであることを示します。カスタマイズの詳細は、59 ページの「構成カスタマイズの表示」を参照してください。

```
$ svccfg -s milestone/goals listcust
$ svccfg -s milestone/goals listprop -l all goals-dependencies/entities
```

例 20 目標サービスからの依存関係の削除

-c オプションは、依存関係セットの管理カスタマイズをクリアします。目標サービスの依存関係のセットが、マニフェストおよびプロファイルレイヤーで提供された依存関係のセットに戻ります。レイヤーの内容と階層については、30 ページの「リポジトリレイヤー」を参照してください。

次のコマンドは、milestone/goals サービスから、デフォルトの milestone/multi-user-server 依存関係を除くすべての依存関係を削除します。milestone/goals サービスではない目標サービスのすべての admin レイヤー依存関係をクリアするには、-g オプションを使用します。

```
$ svcadm goals -c
```

次のどちらのコマンドも、マニフェストレイヤーまたはプロファイルレイヤーにある目標の依存関係セットのみが残ることを示します。

```
$ svcprop -p goals-dependencies/entities -l all milestone/goals
$ svccfg -s milestone/goals listprop -l all goals-dependencies/entities
```

`svccfg delcust` コマンドは、サービスインスタンスのすべての `admin` レイヤーのカスタマイズを削除できます。目標サービスインスタンスの依存関係のカスタマイズのみを削除するには、`svcadm goals` コマンドの `-c` オプションを使用します。`delcust` サブコマンドの詳細は、例40「カスタマイズの削除」を参照してください。

プロファイルファイルでの目標サービスの設定

目標をファイルでバックアップされた構成 (バンドルサポート付きの構成とも呼ばれます) として設定するには、第5章「複数のシステムの構成」の説明に従って SMF プロファイルを作成します。ファイルでバックアップされた構成は、`svcadm goals -c` コマンドを使用して削除できません。

既存の目標サービスの場合、プロファイルを作成するためのもっとも簡単な方法は 106 ページの「`svccfg` を使用したプロファイルの作成方法」の説明に従って `svccfg extract` コマンドを使用することです。`goals-dependencies` セクションを変更します。

次のコマンドは、プロファイルレイヤーで作成されたカスタマイズのほか、`admin` レイヤーで作成されたすべてのカスタマイズを表示します。

```
$ svccfg -s milestone/goals listcust -L goals-dependencies
```

状態遷移および FMA イベントの通知の構成

サービスが状態を変更したり、FMA イベントが発生したときに通知するようにシステムを構成できます。Simple Mail Transfer Protocol (SMTP) または Simple Network Management Protocol (SNMP) のどちらかの通知を指定できます。

SMF は、システムの SNMP 構成を使用します。デフォルトの通知を変更する場合を除き、SMF では SNMP 構成は必要ありません。デフォルトでは、`maintenance` の遷移に関して SNMP トラップが送信されます。遷移通知に SNMP を使用する場合、ほかの状態遷移に対し追加トラップを構成できます。例については、`snmp-notify`(8) のマニュアルページを参照してください。

次の例では、SMF および FMA イベントに関する通知パラメータを設定する方法と、通知パラメータを削除する方法を示します。`maintenance`、`offline`、または `degraded` の状態でサービスの通知を受信した場合、113 ページの「機能低下、オフライン、または保守であるインスタンスの修復」の説明に従って `svcs` コマンドを使用して調査します。

例 21 サービス状態イベントに関するグローバル通知の構成

次のコマンドを実行すると、サービスが maintenance 状態に移行したときに電子メールを送信する通知が作成されます。

```
$ svccfg setnotify -g to-maintenance mailto:sysadmins@example.com
```

`-g` `-g` オプションは、カスタム値が設定されていないすべてのサービスインスタンスについて、この通知パラメータを設定します。変更したすべてのサービスインスタンスがリフレッシュされます。`-g` オプションは、サービス状態遷移に関する通知を設定する場合にのみ使用でき、FMA イベントについては使用できません。

`to-maintenance` smf(7)のマニュアルページの通知パラメータのセクションに記されているように、`to-maintenance` 引数は状態遷移イベントです。状態名だけを指定した場合、`to-`*state* および `from-`*state* の両方の遷移が含まれます。このイベントは、遷移のコンマ区切りのリストになる場合もあります。

`mailto:` `mailto` 引数は、指定されたイベントについて受信する通知を指定します。この引数は `snmp` も指定できます。`snmp` 通知値は、`snmp:active` または `snmp:inactive` のどちらかにする必要があります。`mailto` 通知値は、この例に示した形式に加え、`mailto:active` や `mailto:inactive` にすることができます。通知パラメータを設定すると、そのイベントの既存の値は上書きされます。`active` および `inactive` の設定は既存の値を上書きするのではなく、既存の通知が指定したイベントにとって有効かどうかを切り替えます。

例 22 複数の状態イベントの通知の構成

この例では、システムがシャットダウンしたとき、正常にブートしたとき、またはブートに失敗したときに常に通知します。次のコマンドは、サービスが `online` 状態に遷移したとき、`online` 状態から遷移したとき、または `maintenance` 状態に遷移したときに電子メールを送信する通知を作成します。`online` を指定することは、`to-online` と `from-online` の両方を指定することと同じです。

```
$ svccfg setnotify -g online,to-maintenance mailto:sysadmins@example.com
```

例 23 指定されたサービスインスタンスに関する通知の構成

次のコマンドを実行すると、`pkg/mirror` サービスが `maintenance` 状態に遷移したときに電子メールを送信する通知が作成されます。

```
$ svccfg -s pkg/mirror setnotify to-maintenance mailto:installteam@example.com
```

次のコマンドを実行すると、`http:apache22` サービスが `online` の状態から遷移したときに電子メールを送信する通知が作成されます。

```
$ svccfg -s http:apache22 setnotify from-online mailto:webservices@example.com
```

例 24 FMA イベントに関する通知の構成

`problem-diagnosed` 引数は FMA イベントです。この引数は、FMA イベントのコンマ区切りのリストにすることができます。`smf`(7) のマニュアルページの「通知パラメータ」の FMA イベントのリストを参照してください。

```
$ svccfg setnotify problem-diagnosed mailto:IT@example.com
```

例 25 通知設定の削除

次のコマンドを実行すると、前述の例で設定した通知設定が削除されます。

```
$ svccfg delnotify -g to-maintenance
$ svccfg -s pkg/mirror delnotify to-maintenance
$ svccfg setnotify problem-diagnosed mailto:root@localhost
```

♦ ♦ ♦ 第 4 章

4 サービスの構成

SMF は、サービス構成リポジトリに構成データを格納します。 SMF サービスを構成するとは、構成リポジトリのデータを変更してから、その変更を実行中のスナップショットにコミットすることを意味します。この章では、構成リポジトリ内のデータを変更する方法について説明します。構成リポジトリ内のデータの表示については、48 ページの「サービス構成の検査」を参照してください。実行中のスナップショットへの構成変更のコミットについては、70 ページの「サービス構成の再読み取り」を参照してください。

それぞれのサービスおよびサービスインスタンスは構成データをプロパティーに格納し、プロパティーはプロパティーグループにまとめられています。構成リポジトリ内のデータの変更には、サービスプロパティー値の変更、カスタムプロパティーグループおよびプロパティーの作成、サービスの新しいインスタンスの作成、プロファイルの適用などがあります。構成の変更には、カスタマイズの削除とリポジトリスナップショットの復元も含まれます。

この章では、次について説明します。

- プロパティー値の追加および変更
- プロパティーおよびプロパティーグループの追加および削除
- サービスインスタンスの追加
- スナップショットの復帰
- サービスマニフェストおよびプロファイルのインポートおよび適用
- `inetd` で制御されるサービスの変更
- ファイルによって構成されるサービスの変更

SMF 構成変更は、Oracle Solaris 監査フレームワークを使用してログに記録することができます。詳細は、『Managing Auditing in Oracle Solaris 11.4』 の 「Configuring the Audit Service」を参照してください。

サービス構成コマンドの使用

svccfg コマンドはサービス構成リポジトリのデータを操作します。svccfg コマンドを使用して行う変更はすべて、admin レイヤーに記録されます。レイヤーの詳細は、30 ページの「リポジトリレイヤー」を参照してください。svccfg コマンドで行われた変更は、現在のプロパティー値または編集中のプロパティー値として、サービス構成リポジトリに格納され、実行中のスナップショットに即座には表示されません。構成データを変更する場合、svcadm refresh コマンドまたは svccfg refresh コマンドを使用して、新しい値を実行中のスナップショットにコミットします。

新しく変更したデータを実行中のスナップショットと分けておくと、複数の変更を行なってからその変更をすべて一緒に実行中のスナップショットにコミットできます。複数の変更を行なっている過程で、一部のプロパティー値は互換性や整合性がなくなる場合もありますが、実行中のスナップショットは変更されません。変更が終了したら、リフレッシュを実行します。

次のどの方法でも svccfg コマンドを使用できます。

- svccfg editprop コマンドを使用して、現在選択されているエンティティーのプロパティーグループおよびプロパティーに対してエディターを呼び出す。
- コマンド行で完全な svccfg コマンドを入力し、setprop などのサブコマンドを指定する。
- svccfg または svccfg -s *FMRI* だけをコマンド行で入力し、対話式セッションを開始する。
- -f オプションを指定して、ファイルから svccfg コマンドを読み取る。

プロパティーエディターの呼び出し

次の例に示すように、svccfg コマンドを呼び出すと、選択したエンティティーのプロパティーに対してエディターが開きます。この形式の svccfg コマンドは非常に高速で、複数のプロパティー値を変更する場合に便利です。editprop サブコマンドの場合、-s オプションを付けてエンティティーを指定する必要があります。

```
$ svccfg -s pkg/server:s11 editprop
```

指定したエンティティーの各プロパティーの現在値に対する setprop コマンドのファイルは、VISUAL 環境変数で指定されたエディタで開きます。VISUAL が定義されていない場合、EDITOR で指定したエディターが開かれます。VISUAL も EDITOR も定義されていない場合は、プロパティーファイルは vi で開かれます。

ファイルの各行の先頭にはコメント文字が付けられます。svccfg で編集中の構成内のプロパティーの値を変更するには、コメント文字を削除し、値を変更して、ファイ

ルを保存します。実行中のスナップショット内のプロパティーの値を変更するには、ファイルの最終行 (refresh サブコマンドです) のコメント文字を削除します。

次のリストでは、editprop サブコマンドで作成されたファイルの部分的な例を示します。

```
##
## Change property values by removing the leading '#' from the
## appropriate lines and editing the values. svccfg subcommands
## such as delprop can also be added to the script.
##
## Property group "pkg"
## The following properties are defined in the selected instance
## (svc:/application/pkg/server:s11)

# setprop pkg/port = count: 81
# setprop pkg/inst_root = astring: /export/ipsrepos/Solaris11

## The following properties inherit from the parent service
## (svc:/application/pkg/server)

# ...

## Property group "pkg_bui"

# ...

## Property group "pkg_secure"

# ...

## Uncomment to apply these changes to this instance.
# refresh
```

ファイル状態の命令として、setprop 以外のサブコマンドを追加できます。たとえば、delprop コマンドを追加できます。framework や dependency などの一部のプロパティーグループは、デフォルトでは表示されません。すべてのプロパティーを表示するには、editprop -a を指定します。

この一時ファイル内のコメントのないコマンドは、編集セッションを保存および停止するときに実行されます。

対話式またはファイルを使用した svccfg の呼び出し

次の例に示すように svccfg コマンドを対話式に呼び出すと、複数の構成操作を実行する場合に便利です。

```
$ svccfg
svc:> select pkg/server
svc:/application/pkg/server> list
:properties
default
svc:/application/pkg/server> add s11
svc:/application/pkg/server> select s11
```

```
svc:/application/pkg/server:s11> setprop pkg/inst_root=/export/ipsrepos/Solaris11
svc:/application/pkg/server:s11> setprop pkg/port=81
svc:/application/pkg/server:s11> unselect
svc:/application/pkg/server> add oss
svc:/application/pkg/server> select oss
svc:/application/pkg/server:oss> setprop pkg/inst_root=/export/ipsrepos/SolarisStudio
svc:/application/pkg/server:oss> setprop pkg/port=82
svc:/application/pkg/server:oss> unselect
svc:/application/pkg/server> list
:properties
default
s11
oss
svc:/application/pkg/server> refresh
svc:/application/pkg/server> select pkg/mirror:default
svc:/application/pkg/mirror:default> listprop config/crontab_period
config/crontab_period astring     "30 2 25 * *"
svc:/application/pkg/mirror:default> setprop config/crontab_period="00 3 25 * *"
svc:/application/pkg/mirror:default> refresh
svc:/application/pkg/mirror:default> exit
$
```

前述の例での対話式プロンプトで示された同じコマンドをファイルで指定することもでき、次のコマンドなどのコマンドとともに実行できます。

```
$ svccfg -f cfgpkgrepos
```

プロパティー値の設定

次のコマンドはプロパティー値を設定します。

`svccfg setprop`
: プロパティーの値を変更します。

`svccfg addpropvalue`
: 複数値のプロパティーに値を追加します。

`svccfg setenv`
: サービスプロセス実行環境の環境変数の値を変更します。

必ず `svccfg refresh` コマンドまたは `svcadm refresh` コマンドを使用して、実行中のスナップショットに構成変更をコミットしてください。

例 26 単純な値の設定

`setprop` をもっとも簡単に使用するには、選択されたサービスまたはインスタンスに対して、*pg*/*name* を指定します。ここで、*pg* はプロパティーグループの名前で、*name* はプロパティーの名前であり、等号のあとに新しい値を指定します。プロパティーが

すでに存在しているか、テンプレート化されている場合、プロパティー値を指定する必要はありません。

```
$ svccfg -s pkg/server:s11 setprop pkg/port=81
```

例 27 埋め込まれた空白を含む値の設定

埋め込まれた空白を含む値を設定するには、二重引用符を使用します。シェルによっては、二重引用符の付いた文字列を一重引用符で囲む必要があります。

```
$ svccfg -s pkg/mirror setprop config/crontab_period = "00 3 25 * *"
$ svccfg -s pkg/mirror setprop config/crontab_period = '"00 3 25 * *"'
```

二重引用符またはバックスラッシュ文字を含む値を設定するには、引用符を使用します。二重引用符またはバックスラッシュ文字をエスケープするには、バックスラッシュ文字を使用します。

例 28 一連の値である値の設定

一連の値を単一の値として指定するには、括弧を使用します。シェルによっては、値のセットも一重引用符で囲む必要があります。

```
$ svccfg -s dns/client setprop config/nameserver = (10.0.0.1 192.168.0.1)
$ svccfg -s dns/client setprop config/nameserver = '(10.0.0.1 192.168.0.1)'
$ svccfg -s dns/client listprop config/nameserver
config/nameserver net_address 10.0.0.1 192.168.0.1
```

一連の値に含められる値の数を調べるには、`describe` サブコマンドを使用します。

```
$ svccfg -s dns/client describe -v config/nameserver
config/nameserver net_address 10.0.0.1 192.168.0.1
    type: net_address
    required: false
    The IP address of a DNS nameserver to be used by the resolver.
    visibility: readwrite
    minimum number of values: 1
    maximum number of values: 3
  value: 10.0.0.1
  value: 192.168.0.1
```

例 29 名前に特殊文字が含まれているプロパティーの設定

『Oracle Solaris 12 でのシステムサービスの開発』 の 「Naming Property Groups and Properties」には、プロパティーグループやプロパティー名で使用できる予約文字のリストが記載されています。これらの予約文字は、FMRI でエンコードされて表示されます。

```
$ svccfg -s enchars_example:default
svc:/site/enchars_example:default> listprop config
config                      application
config/%25%20increase       count       10
config/maximum%20%23        count       9
config/start%3Aend          count       10
```

```
config/students%2Fteachers count       20
```

これらのプロパティーを設定するには、エンコードされた名前をコピーできます。

```
svc:/site/enchars_example:default> setprop config/students%2Fteachers=21
svc:/site/enchars_example:default> listprop config/students%2Fteachers
config/students%2Fteachers count       21
```

これらのプロパティーを設定するための別の方法として、`-G` および `-P` オプションの使用があります。

```
svc:/site/enchars_example:default> setprop -G config -P students/teachers 20
svc:/site/enchars_example:default> listprop -G config -P students/teachers
config/students%2Fteachers count       20
```

スペースが含まれているプロパティーグループ名またはプロパティー名には引用符を使用します。

```
svc:/site/enchars_example:default> setprop -G config -P "% increase" 12
svc:/site/enchars_example:default> listprop -G config -P "% increase"
config/%25%20increase count       12
```

`editprop` ツールは、自動的に `-G` および `-P` オプションを使用します。

```
$ svccfg -s enchars_example:default editprop
...
$ setprop -G "config" -P "% increase" -T count 10
$ setprop -G "config" -P "maximum #" -T count 9
$ setprop -G "config" -P "start:end" -T count 10
$ setprop -G "config" -P "students/teachers" -T count 20
```

例 30 入れ子になったプロパティーグループ内のプロパティーの値の設定

別のプロパティーグループの子であるプロパティーグループ内のプロパティーの値を設定するには、そのプロパティーのすべての先祖を指定します。

次の例では、`http` と `https` は `config` プロパティーグループの子プロパティーグループであり、`ssl` は `https` プロパティーグループの子プロパティーグループです。

```
$ svccfg -s npg_example listprop config
config                    application
config/port               count       80
config/http               application
config/http/port          count       80
config/https              application
config/https/port         count       443
config/https/ssl          application
config/https/ssl/certfile astring     cert.crt
config/https/ssl/keyfile  astring     key.crt
```

入れ子になったプロパティーグループのプロパティーの別の値を指定するには、プロパティーの完全な FMRI を指定するか、または複数の `-G` オプションを使用します。次の例では、デフォルトのインスタンスの証明書と鍵ファイルの値を変更します。

```
$ svccfg -s npg_example:default
svc:/site/npg_example:default> setprop config/https/ssl/certfile=cert-1.pem
```

```
svc:/site/npg_example:default> setprop -G config -G https -G ssl -P keyfile key-1.pem
svc:/site/npg_example:default> listprop config/https/ssl
config/https/ssl          application
config/https/ssl/certfile astring     cert-1.pem
config/https/ssl/keyfile  astring     key-1.pem
```

例 31 値の追加

選択したサービスまたはサービスインスタンスの指定したプロパティーに所定の値を追加するには、addpropvalue サブコマンドを使用します。新しい値は、プロパティーのプロパティー値に関する既存のリストの末尾に追加されます。

```
$ svcprop -p keymap/layout keymap:default
US-English
$ svccfg -s keymap:default addpropvalue keymap/layout UK-English
$ svccfg -s keymap:default listprop keymap/layout
keymap/layout astring     "US-English" "UK-English"
```

前述の setprop の例では、値のセット内のすべての値を一度に指定する必要があります。1 つの値だけを指定した場合、その値が、1 つの値から成る新しいセットになります。この addpropvalue の例では、追加した値は個別の値です。これらの追加した値にアクセスするには、値に対して libscf 関数 scf_iter_property_values() を繰り返し使用する必要があります。listprop は両方の値を表示しますが、describe は最初の値だけを表示し、このプロパティーの値に許容される最大数が 1 であることを報告します。

```
$ svccfg -s keymap:default describe -v keymap/layout
keymap/layout astring     US-English
    type: astring
    required: true
    The keyboard layout
    visibility: readwrite
    minimum number of values: 1
    maximum number of values: 1
  value: US-English
```

▼ 定期的またはスケジュールされているサービスをスケジュールする方法

定期的またはスケジュールされているサービスは、一定の間隔で、またはサービスのプロパティー値を設定することによって指定した時間で、その起動メソッドを実行します。

1. **スケジュールするサービスが定期的なサービスなのかスケジュールされているサービスなのかを確認します。**

   次のコマンドは、リスタータが periodic-restarter サービスであるすべてのインスタンスを一覧表示するため、システムのすべての定期的およびスケジュールされているサービスを一覧表示します。

```
$ svcs -R svc:/system/svc/periodic-restarter:default
```

`svcs -l`、`svcprop`、または `svccfg` を使用して、`general/restarter` プロパティーの値を表示することもできます。

2. **サービスの現在のスケジュールを表示します。**
   次のコマンドは、インスタンスが定期的なサービスであるのかスケジュールされているサービスであるのかを示します。定期的なサービスには `periodic` プロパティーグループ、スケジュールされているサービスには `scheduled` プロパティーグループがあります。

   ```
   $ svcprop -p periodic -p scheduled FMRI
   ```

   FMRI が定期的なサービスである場合、`periodic/period` プロパティー、および設定されているその他の `periodic` プロパティーの値が表示されます。`scheduled` プロパティーグループに関するエラーメッセージが表示されます。
   FMRI がスケジュールされているサービスである場合、`scheduled/interval` および `scheduled/frequency` プロパティー、および設定されているその他の `scheduled` プロパティーの値が表示されます。`periodic` プロパティーグループに関するエラーメッセージが表示されます。

3. **スケジュールを変更します。**
   値のないプロパティーは、`svcprop` または `svccfg listprop` コマンドで表示されません。必要なスケジュールを実現するために設定するプロパティーを判断するには、次のリファレンスを使用します。

   - 定期的なサービスの場合は、『Oracle Solaris 12 でのシステムサービスの開発』 の「Specifying the periodic_method Element」および『Oracle Solaris 12 でのシステムサービスの開発』 の 「Scheduling Executions of a Periodic Service Start Method」を参照してください。定期的なサービスをスケジュールするには、`periodic/period` プロパティーだけが必要です。その他のプロパティーはオプションです。
   - スケジュールされているサービスの場合は、『Oracle Solaris 12 でのシステムサービスの開発』 の 「Specifying the scheduled_method Element」および『Oracle Solaris 12 でのシステムサービスの開発』 の 「Scheduling Executions of a Scheduled Service Start Method」を参照してください。スケジュールされているサービスをスケジュールするには、`scheduled/interval` および `scheduled/frequency` プロパティーが必要です。

   前に設定されていないプロパティーを設定する場合は、例32「コンプライアンス評価サービスのスケジュール」に示すように、プロパティータイプを指定する必要があります。そのプロパティー値をリセットする場合は、もう一度プロパティータイプを指定する必要はありません。また、前に設定されていないプロパティーを設定すると、次のメッセージが表示されます。

   ```
   Type required for new properties.
   ```

4. **変更されたプロパティー値を読み取ります。**

svcadm または svccfg コマンドを使用して、サービスインスタンスをリフレッシュします。svcprop コマンドを再度使用して、periodic または scheduled プロパティー値が更新されていることを確認します。

5. **サービスインスタンスがオンラインであることを確認します。**
   プロパティー値を変更しているときにインスタンスが無効になったか、またはリフレッシュを実行したときにインスタンスが保守状態になった可能性があります。インスタンスが保守状態にある場合は、47 ページの「サービスログファイルの表示」の説明に従ってログファイルを表示します。可能性のある問題は次のとおりです。

   - 競合するプロパティーが設定されています。たとえば、scheduled/day_of_month および scheduled/day プロパティー値の両方は設定できません。
   - すべての必須プロパティーが設定されていません。たとえば、day および minute だけでは設定できず、hour も設定する必要があります。

   サービスを保守状態から移行させる詳細については、114 ページの「保守状態のインスタンスの修復方法」を参照してください。

6. **スケジュールを確認します。**
   次のコマンドは、サービスが次回実行される時間を示します。

   ```
   $ svcs -o nrun FMRI
   ```

   スケジュールが表示されない場合は、インスタンスがオンラインになっていることを確認します。

例 32　コンプライアンス評価サービスのスケジュール

次のコマンドは、コンプライアンス評価サービスが現在毎週 1 回実行するようにスケジュールされていることを示しています。

```
$ svcprop -p scheduled compliance:default
scheduled/frequency integer 1
scheduled/interval astring week
```

このスケジュールでは、インスタンスは最初の週はあらゆる曜日のあらゆる時間に実行できます。その後の週では、同じ曜日のいずれかの時間に実行されます。

次のコマンドは、毎週水曜日の午前 3 時から午前 4 時までのいずれかの時間に開始メソッドが起動するようにインスタンスを設定します。

```
$ svccfg -s compliance:default
svc:/application/security/compliance:default> setprop scheduled/day=Wednesday
Type required for new properties.
svc:/application/security/compliance:default> setprop scheduled/day = astring: Wednesday
svc:/application/security/compliance:default> setprop scheduled/hour = integer: 3
svc:/application/security/compliance:default> refresh
svc:/application/security/compliance:default> exit
$ svcs compliance:default
STATE          STIME    FMRI
online         13:16:57 svc:/application/security/compliance:default
```

```
$ svcs -o nrun compliance:default
NRUN
 3:27:16
```

▼ ttymon プロパティー値の変更方法

この手順では、ttymon に渡されるパラメータを変更する方法を示します。

1. **変更するサービスを識別します。**
   ttymon(8) のマニュアルページには、変更するサービスは svc:/system/console-login であることが記載されています。ttymon(8) のマニュアルページにはまた、ttymon プロパティーグループ内のプロパティーの説明も含まれています。
   次のコマンドを実行すると、このイメージ内の console-login サービスの複数のインスタンスが表示され、default インスタンスが現在オンラインになっている唯一のインスタンスであることが示されます。

   ```
   $ svcs console-login
   STATE          STIME    FMRI
   disabled       10:49:43 svc:/system/console-login:terma
   disabled       10:49:43 svc:/system/console-login:termb
   online         10:50:54 svc:/system/console-login:default
   ```

2. **変更するプロパティーを識別します。**
   次のコマンドを実行すると、default インスタンスの ttymon プロパティーグループ内の各プロパティーの名前、データ型、値、および簡単な説明が表示されます。

   ```
   $ svccfg -s console-login:default describe ttymon
   ttymon                 application
   ttymon/device         astring     /dev/console
       The terminal device to be used for the console login prompt.
   ttymon/terminal_type astring
       Sets the initial value of the TERM environment variable
   ```

   前述の出力には、terminal_type プロパティーの値は表示されません。次のコマンドは、console-login:default インスタンスの ttymon/terminal_type プロパティーの値が、現在 null であることを確認します。

   ```
   $ svcprop -p ttymon/terminal_type console-login:default
   ""
   ```

3. **プロパティー値を変更します。**
   次のコマンドを入力して、console-login:default インスタンスの ttymon/terminal_type プロパティーの値を xterm に変更します。

   ```
   $ svccfg -s system/console-login:default setprop ttymon/terminal_type=xterm
   ```

4. **実行中のスナップショットに新しい値をコミットします。**
   次の出力では、terminal_type プロパティーの値が編集中の構成では変更されているが、実行中のスナップショットでは変更されていないことが示されています。

```
$ svccfg -s console-login:default listprop ttymon/terminal_type
ttymon/terminal_type astring     xterm
$ svcprop -p ttymon/terminal_type console-login:default
""
```

サービスインスタンスをリフレッシュしたあと、実行中のスナップショットでプロパティー値は変更されます。

```
$ svcadm refresh console-login:default
$ svcprop -p ttymon/terminal_type console-login:default
xterm
```

▼ サービスプロセス環境の環境変数を変更する方法

この手順では、サービスが開始したプロセスが実行している環境で環境変数の値を設定する方法を示します。

この手順の例では、デバッグに役立つように cron 環境変数を変更する方法を示します。

1. **サービスが実行されていることを確認します。**
   次の出力には、cron サービスがオンラインになっており、cron プロセスが実行していることが示されています。

```
$ svcs -p cron
STATE          STIME    FMRI
online         10:24:05 svc:/system/cron:default
               10:24:05     1089 cron
```

2. **環境変数を設定します。**
   setenv サブコマンドは、サービスまたはサービスインスタンスによって開始されたプロセスが実行している環境の環境変数を設定します。
   次のコマンドを使用して、設定する環境変数の現在の値を確認します。

```
$ pargs -e `pgrep -f /usr/sbin/cron`
```

   この例で設定される環境変数には、現在値が割り当てられていません。
   次のコマンドは、svc:/system/cron:default サービスインスタンスによって開始された /usr/sbin/cron プロセスの UMEM_DEBUG および LD_PRELOAD 環境変数を設定します。

```
$ svccfg -s system/cron:default setenv UMEM_DEBUG default
$ svccfg -s system/cron:default setenv LD_PRELOAD libumem.so
```

3. **サービスのリフレッシュと再起動を行います。**
   環境変数の値の変更が有効になるには、再起動とリフレッシュが必要です。

```
$ svcadm refresh system/cron:default
```

```
$ svcadm restart system/cron:default
```

4. **正しく変更されたことを確認します。**
   次の出力には、サービスが再起動し、プロセスに新しいプロセス ID が与えられ、そのプロセス環境に対して 2 つの環境変数が設定されていることが示されています。

```
$ svcs -p cron
STATE          STIME    FMRI
online          9:24:39 svc:/system/cron:default
                9:24:39     5601 cron
$ svcprop -g method -p environment system/cron:default
start/environment astring LD_PRELOAD=libumem.so UMEM_DEBUG=default
$ pargs -e `pgrep -f /usr/sbin/cron`
5601: /usr/sbin/cron
envp[0]: LOGNAME=root
envp[1]: LD_PRELOAD=libumem.so
envp[2]: PATH=/usr/sbin:/usr/bin
envp[3]: SMF_FMRI=svc:/system/cron:default
envp[4]: SMF_METHOD=start
envp[5]: SMF_RESTARTER=svc:/system/svc/restarter:default
envp[6]: SMF_ZONENAME=global
envp[7]: UMEM_DEBUG=default
```

参照 unsetenv サブコマンドは、サービスまたはサービスインスタンスによって開始されたプロセスの環境変数を設定解除します。

プロパティーグループ、プロパティー、およびプロパティー値の追加

次のコマンドはプロパティーおよびプロパティーグループを追加します。

`svccfg setprop`
`svccfg addpropvalue`
: プロパティーがまだ存在していない場合、値が設定されているプロパティーを追加します。

`svccfg addpg`
: サービスまたはサービスインスタンスに新しいプロパティーグループを追加します。

必ず `svccfg refresh` コマンドまたは `svcadm refresh` コマンドを使用して、実行中のスナップショットに構成変更をコミットしてください。

例 33 `addpg` を使用した新しいプロパティーグループの作成

`addpg` サブコマンドを使用して、選択したサービスまたはサービスインスタンスにプロパティーグループを追加します。

```
svccfg -s FMRI addpg name type [flags]
```

type　　慣例により、*type* の値は通常 `application` です。プロパティーグループタイプの詳細は、『Oracle Solaris 12 でのシステムサービスの開発』を参照してください。

flags　　*flags* の値に `P` を指定して、プロパティーグループと追加したすべてのプロパティーを非永続として格納します。`P` を指定すると、このプロパティーグループおよび含まれるプロパティーはリブート時に自動的に削除されます。値 `P` は、`SCF_PG_FLAG_NONPERSISTENT` の別名です。`scf_service_add_pg`(3SCF) のマニュアルページを参照してください。

```
$ svccfg -s svc:/site/my-svc addpg config application
$ svccfg -s my-svc listprop config
config  application
$ svccfg -s my-svc:default listprop config
$
```

この例では、管理者は、`config` プロパティーグループを親サービス `my-svc` に追加しましたが、インスタンス `my-svc:default` には追加しませんでした。`listprop` コマンドは、`config` プロパティーグループがサービスインスタンスに存在していないことを示します。

次の例では、入れ子になったプロパティーグループを追加します。入れ子になったプロパティーグループは、すべて同じタイプである必要があります。

```
$ svccfg -s my-svc addpg config/security application
$ svccfg -s my-svc listprop config
config          application
config/security application
```

次の例では、名前に予約文字が含まれているプロパティーグループを追加します。

```
$ svccfg -s my-svc addpg -G config -G security -G certs+keys application
$ svccfg -s my-svc listprop config
config                          application
config/security                 application
config/security/certs%2Bkeys application
```

例 34　　`setprop` を使用した新しいプロパティーの作成

82 ページの「プロパティー値の設定」で説明しているように `setprop` サブコマンドを使用して、プロパティー値を設定します。選択したインスタンスまたはサービスにプロパティーグループがまだ存在していない場合、テンプレート定義にタイプおよびフラグがあればプロパティーグループが作成されます。選択したインスタンスまたはサービスにプロパティーがまだ存在していない場合、プロパティー *type* を指定する必要があります。

```
$ svccfg -s my-svc:default setprop config/source = uri: http://example1.com/
```

```
$ svccfg -s my-svc:default listprop config/source
config/source uri         http://example1.com/
```

プロパティーグループと同じ名前を持つ新しいプロパティーを追加するには、`-G` および `-P` オプションを使用します。

```
$ svccfg -s my-svc setprop -G config -P security -T astring yes
$ svccfg -s my-svc listprop config/security
config/security              astring     yes
config/security              application
config/security/certs%2Bkeys application
```

次の例では、名前に予約文字が含まれているプロパティーを追加します。新しいプロパティーのタイプを指定するには、`-T` オプションを使用します。

```
$ svccfg -s my-svc setprop -G config -P "maximum %" -T integer 100
$ svccfg -s my-svc listprop -G config -P "maximum %"
config/maximum%20%25 integer     100
```

例 35 `addpropvalue` を使用した新しいプロパティーの作成

82 ページの「プロパティー値の設定」で説明しているように、`addpropvalue` サブコマンドを使用して、プロパティー値を追加します。選択したインスタンスまたはサービスにプロパティーグループがまだ存在していない場合、テンプレート定義にタイプおよびフラグがあればプロパティーグループが作成されます。選択したインスタンスまたはサービスにプロパティーがまだ存在していない場合、プロパティー *type* を指定する必要があります。

```
$ svccfg -s my-svc:default addpropvalue config/source http://example2.com/
$ svccfg -s my-svc:default addpropvalue config/target hostname: example3
$ svccfg -s my-svc:default listprop config
config                       application
config/source                uri         http://example1.com/ http://example2.com/
config/target                hostname    example3
```

プロパティーグループ、プロパティー、およびプロパティー値の削除

次のコマンドはプロパティー値、プロパティー、およびプロパティーグループを削除します。

`svccfg setprop`

プロパティーの値すべてを削除します。

`svccfg delpropvalue`

指定されたパターンに一致する指定されたプロパティーのすべての値を削除します。

`svccfg delprop`
: プロパティーを削除します。

`svccfg delpg`
: プロパティーグループを削除します。

`svccfg delcust`
: 管理カスタマイズを削除します。

必ず `svccfg refresh` コマンドまたは `svcadm refresh` コマンドを使用して、実行中のスナップショットに構成変更をコミットしてください。

管理構成の削除

`svccfg` コマンドまたは `libscf` 呼び出しを使用して行なった構成の変更は、サービス構成リポジトリの `admin` レイヤーだけを変更します。レイヤーの詳細は、30 ページの「リポジトリレイヤー」を参照してください。`admin` レイヤーだけで定義されており、ほかのレイヤーには存在しない構成を削除した場合、その構成は失われます。`-l` オプションを使用してサービス構成リポジトリのすべてのレイヤーを表示した場合でも、構成を表示するコマンドを実行しても、削除した構成は表示されません。ほかのレイヤーに存在する構成の削除については、95 ページの「非管理構成の削除」を参照してください。

例 36 プロパティーのすべての値の削除

82 ページの「プロパティー値の設定」で説明しているように `setprop` サブコマンドを使用します。プロパティーのすべての値を削除するには、タイプや値を指定しないでください。値は削除されますが、プロパティーは引き続き存在します。

```
$ svccfg -s my-svc:default setprop config/vendor =
$ svccfg -s my-svc:default listprop config/vendor
config/vendor   astring
```

例 37 プロパティーのすべての一致した値の削除

`delpropvalue` サブコマンドを使用して、所定のパターンに一致する名前付きプロパティーのすべての値を削除します。

```
$ svccfg -s my-svc:default setprop config/tool = astring: '(hammer tongs wrench)'
$ svccfg -s my-svc:default listprop config
config           application
config/customer astring     acustomer
config/vendor   astring     "vendora" "vendorb"
config/tool     astring     "hammer tongs wrench"
$ svccfg -s my-svc:default delpropvalue config/vendor '*b'
$ svccfg -s my-svc:default delpropvalue config/tool 'tong*'
$ svccfg -s my-svc:default listprop config
```

```
config           application
config/customer astring     acustomer
config/vendor   astring     vendora
config/tool     astring     "hammer tongs wrench"
$ # config/tool is a single value that is a value set
$ svccfg -s my-svc:default delpropvalue config/tool '*tong*'
$ svccfg -s my-svc:default listprop config
config           application
config/customer astring     acustomer
config/vendor   astring     vendora
config/tool     astring
```

例 38 プロパティーの削除

`delprop` サブコマンドを使用して、選択したサービスまたはサービスインスタンスの名前付きプロパティーを削除します。

```
$ svccfg -s my-svc:default delprop config/tool
$ svccfg -s my-svc:default listprop config
config           application
config/customer astring     acustomer
config/vendor   astring     vendora
```

例 39 プロパティーグループの削除

`delpg` サブコマンドと `delprop` サブコマンドはどちらもプロパティーグループを削除できます。`delpg` サブコマンドは、選択したサービスまたはサービスインスタンスの名前付きプロパティーグループを削除します。`delprop` サブコマンドは、プロパティーに名前が付いていない場合に、名前付きプロパティーグループを削除します。

```
$ svccfg -s my-svc:default delpg config
$ svccfg -s my-svc:default listprop config
$
```

例 40 カスタマイズの削除

`delcust` サブコマンドは、選択したサービスまたはサービスインスタンスの管理カスタマイズを削除します。`delcust` サブコマンドを使用する前に、同じパターンまたはオプションで `listcust` サブコマンドを使用して、削除される対象を確認してください。パターンを指定する場合、このパターンはプロパティーまたはプロパティーグループに対応している必要があります。

```
$ svccfg -s my-svc:default listcust
config                           application admin
config/customer                  astring     admin                acustomer
config/vendor                    astring     admin                "vendora" "vendorb"
config/tool                      astring     admin                "hammer tongs wrench"
$ svccfg -s my-svc:default listcust '*tool'
config/tool                      astring     admin                "hammer tongs wrench"
$ svccfg -s my-svc:default delcust '*tool'
 Deleting customizations for property: config/tool
$ svccfg -s my-svc:default listcust '*tool'
$ svccfg -s my-svc:default listcust
config                           application admin
```

```
config/customer                 astring     admin               acustomer
config/vendor                   astring     admin               "vendora" "vendorb"
```

非管理構成の削除

サービス構成リポジトリの sysconfig-profile、node-profile、site-profile、enterprise-profile、system-profile、および manifest レイヤー内に存在する構成は、サービスマニフェストおよびプロファイルファイルで定義されています。レイヤーの詳細は、30 ページの「リポジトリレイヤー」を参照してください。SMF では、サービス構成リポジトリとファイルシステムコンテンツとの同期は維持されます。標準の場所にあるマニフェストまたはプロファイルファイルに定義されたすべての構成は、管理カスタマイズ後 (削除後も含む) でもファイルシステム上にそのまま存在しており、サービス構成リポジトリに引き続き格納されています。マニフェストまたはプロファイルで定義されている構成は、バンドルをサポートしていると言われます。バンドルをサポートしている構成を削除すると、情報はファイルシステムから削除されませんが、通常のビューに表示されないようにマスクされます。マスクされたエンティティーの説明については、smf(7)のマニュアルページを参照してください。

バンドルをサポートする構成の削除は管理カスタマイズになります。この場合、delcust サブコマンドは、構成をマスク解除するのであり、何かを削除するわけではありません。マスクされた構成を表示するには、listcust -M サブコマンドを使用します。構成をマスク解除するか、構成の削除またはマスクを元に戻すには、delcust -M サブコマンドを使用します。

例 41 バンドルをサポートする構成の削除

93 ページの「管理構成の削除」では、my-svc サービスの config プロパティーグループだけが admin レイヤーに存在していました。config プロパティーグループは、どのマニフェストにもプロファイルにも存在していませんでした。これらのプロパティーは削除されると、システムから失われていました。この例は、バンドルをサポートしている構成を削除したときの別の結果を示します。

プロパティーはサービスマニフェストで定義されています。

```
$ svccfg -s pkg/server listprop -l all pkg/inst_root
pkg/inst_root astring     admin               /export/ipsrepos/Solaris11
pkg/inst_root astring     manifest            /var/pkgrepo
$ svccfg -s pkg/server delprop pkg/inst_root
```

削除後、オプションを付けずに listprop を使用してもプロパティーは表示されません。プロパティーはバンドルをサポートしているので、サービス構成リポジトリにプロパティーはまだ存在しており、listprop サブコマンドとともに -l または -M オプションを使用して表示できます。

```
$ svccfg -s pkg/server listprop pkg/inst_root
$ svccfg -s pkg/server listprop -l all pkg/inst_root
```

```
pkg/inst_root astring     admin        MASKED /export/ipsrepos/Solaris11
pkg/inst_root astring     manifest     MASKED /var/pkgrepo
$ svccfg -s pkg/server listcust -M
pkg/inst_root astring     admin        MASKED /export/ipsrepos/Solaris11
```

例 42 構成のマスク解除

プロパティーをマスク解除すると、次のように両方のカスタマイズが失われます。

- プロパティーはマスクされず非表示にもなりません。
- プロパティーからカスタマイズした値はなくなります。

```
$ svccfg -s pkg/server delcust -M
 Deleting customizations for property: pkg/inst_root
$ svccfg -s pkg/server listprop -l all pkg/inst_root
pkg/inst_root astring     manifest            /var/pkgrepo
$ svccfg -s pkg/server listprop pkg/inst_root
pkg/inst_root astring     /var/pkgrepo
```

サービスインスタンスの追加

サービスのインスタンスでは、サービスの複数の構成を同時に実行することができます。サービスインスタンスは、共通のサービス構成を継承してカスタマイズします。

add サブコマンドを使用して、選択したサービスの子として、指定された名前で新しいエントリを作成します。

```
$ svcs -Ho inst pkg/server
default
$ svccfg -s pkg/server add s11
$ svcs -Ho inst pkg/server
default
s11
```

スナップショットの復帰

次のそれぞれの操作は、新しい実行中のスナップショットを作成します。

- `svcadm restart manifest-import`
- `svcadm refresh`
- `svccfg refresh`

`revert` サブコマンドは、`-s` オプションで指定されたインスタンスの管理カスタマイズ (`admin` レイヤー) とそのサービスを、名前付きスナップショットまたは現在選択されているスナップショットに記録された値に復帰させます。`listsnap` サブコマンドを使用して、このサービスインスタンスの可能なスナップショットのリストを表示し

ます。selectsnap サブコマンドを使用して、対話式モードでスナップショットを選択します。

```
$ svcprop -p pkg/inst_root pkg/server:default
pkg/inst_root astring /export/ipsrepos/Solaris11
$ svccfg -s pkg/server:default listsnap
initial
previous
running
start
$ svcprop -s previous -p pkg/inst_root pkg/server:default
pkg/inst_root astring /var/pkgrepo
```

revert サブコマンドは、すべての管理カスタマイズを元に復帰させるので、復帰前にすべての管理カスタマイズを一覧表示して、その値を調べてください。

```
$ svcprop -s previous -l admin pkg/server:default
pkg/inst_root astring /var/pkgrepo
$ svccfg -s pkg/server:default revert previous
$ svcadm refresh pkg/server:default
$ svcprop -p pkg/inst_root pkg/server:default
pkg/inst_root astring /var/pkgrepo
```

マニフェストおよびプロファイルのインポートおよび適用

manifest-import サービスを再起動すると、マニフェストおよびプロファイルが新規または変更された場合、標準の場所のマニフェストがインポートされ、標準の場所のプロファイルが適用されます。マニフェストおよびプロファイルの標準の場所については、29 ページの「サービスバンドル」を参照してください。マニフェストをインポートするかプロファイルを適用した結果、サービスが起動または停止することになった場合、適切なメソッドが存在すればこれが実行されます。

標準の場所のファイルを svccfg import コマンドに指定すると、manifest-import サービスが再起動します。

推奨されるベストプラクティスは、svccfg import または svccfg apply コマンドを使用するのではなく、標準の場所にマニフェストおよびプロファイルファイルを配置して、manifest-import サービスを再起動するというものです。

```
$ svcadm restart manifest-import
```

manifest-import サービスを再起動すると、標準の場所にあるプロファイルおよびマニフェスト内の構成が、影響を受けるインスタンスの適切なレイヤー (sysconfig-profile、node-profile、site-profile、enterprise-profile、system-profile、または manifest) に適用され、影響を受けるインスタンスがリフレッシュおよび検証され、新しいスナップショットが作成されます。各レイヤーについては、30 ページの「リポジトリレイヤー」を参照してください。

標準以外の場所にあるプロファイルおよびマニフェストをインポートまたは適用すると、構成は影響を受けるインスタンスの admin レイヤーに適用されます。デフォルト

または初期の構成の配信場所には、標準以外の場所を使用することを強くお勧めします。多数の構成変更を行う場合、標準以外の場所からインポートまたは適用すると、多くのコマンドを発行するより簡単になりますが、manifest-import サービスの自動管理メカニズムのメリットが失われます。manifest-import サービスでは、サービス配信を管理するために、場所がわかっており、状態を予想できることが必要になります。

svccfg apply コマンドは、標準の場所にあるマニフェストおよびプロファイルについても、すべての構成を admin レイヤーに適用します。

inetd で制御されるサービスの変更

inetd によって制御されるサービスは、inetd.conf ファイルの構成から変換された SMF サービスです。inetd コマンドは、これらのサービスの機能低下版のリスタータです。

次の手順では、inetd によって制御されるサービスのプロパティー値を変更する方法を示します。

変更するサービスが inetd によって制御されていることを確認するには、次の例に示すように inetadm コマンドまたは svcs -R コマンドのいずれかを使用して、inetd で制御されるすべてのサービスを一覧表示します。次の例は、リストの一部のみを示しています。

```
$ inetadm
ENABLED   STATE          FMRI
enabled   online         svc:/application/cups/in-lpd:default
...
disabled  disabled       svc:/application/x11/xvnc-inetd:default
$ svcs -R network/inetd:default
STATE     STIME          FMRI
online    8:11:10        svc:/application/cups/in-lpd:default
...
online    8:11:11        svc:/network/rpc/smserver:default
```

-l オプションを付けて inetadm コマンドを実行すると、inetd に制御されたサービスのプロパティーがすべて一覧表示されます。次の例は、指定されたサービスが inetd に制御されたサービスではないことを示すエラーメッセージです。「No restarter property」は、マスターリスタータ svc.startd がサービスインスタンスを管理していることを意味します。

```
$ inetadm -l ssh
Error: Specified service instance "svc:/network/ssh:default" has no restarter
property.  inetd is not the delegated restarter of this instance.
```

同様に、次の例では、「Couldn't find property 'general/restarter'」というメッセージは、デフォルトのリスタータ svc.startd がサービスインスタンスを管理することを示します。

```
$ svcprop -p general/restarter ssh
svcprop: Couldn't find property 'general/restarter' for instance
'svc:/network/ssh:default'.
```

サービスが inetd によって制御されている場合、次の例に示すように、そのリスタータは inetd です。

```
$ svcprop -p general/restarter cups/in-lpd
svc:/network/inetd:default
```

svcs -l コマンドはリスタータも表示します。次の例は、出力の一部のみを示しています。

```
$ svcs -l cups/in-lpd
...
restarter    svc:/network/inetd:default
...
```

▼ inetd 制御サービスのプロパティー値を変更する方法

1. **サービスのプロパティーを一覧表示します。**

   inetadm コマンドの -l オプションを使用して、指定のサービスのプロパティーをすべて一覧表示します。プロパティーの現在の値を調べます。

   ```
   $ inetadm -l FMRI
   ```

2. **プロパティー値を変更します。**

   inetadm コマンドの -m オプションを使用して、指定のプロパティーの値を変更します。特定のサービスのプロパティーの詳細については、そのサービスのマニュアルページを参照してください。

   ```
   $ inetadm -m FMRI property-name=value
   ```

   プロパティー値を削除するには、空の値を指定します。

   ```
   $ inetadm -m svc property=""
   ```

3. **プロパティー値が変更したことを検証します。**

   プロパティーを再度一覧表示し、適切に変更されていることを確認します。

   ```
   $ inetadm -l FMRI
   ```

4. **変更が適用されていることを確認します。**

   プロパティーの変更によって予想された効果があったことを確認します。

例 43 実行が許可された同時プロセス数の制限

この例では、同時に実行することが許可されている finger プロセスの数を制限する方法を示します。

セキュリティーのベストプラクティスでは、inetd によって制御されるシステムサービスごとに、同時に実行することが許可されているプロセスの数を制限することが推奨されています。また、inetd によって制御されるサービスが必要ない場合は、66 ページの「サービスの停止」の説明に従ってそのサービスを無効にします。

同時に実行することが許可されているプロセスの数を制限するために構成するプロパティーは、max_copies プロパティーです。inetadm -p コマンドは、inetd によって管理されるすべてのサービスに共通するプロパティーおよびそのデフォルト値を一覧表示します。次の例は、出力の一部のみを示しています。

```
$ inetadm -p
NAME=VALUE
...
max_copies=-1
...
```

inetadm -l コマンドは、指定されたサービスのすべてのプロパティーを一覧表示して、そのサービスの max_copies プロパティーの現在の値を調査できるようにします。

```
$ inetadm -l finger | grep copies
default  max_copies=-1
```

プロパティーの値を変更するには、inetadm コマンドの -m オプションを使用します。

```
$ inetadm -m finger max_copies=5
```

プロパティー値が変更したことを検証します。

```
$ inetadm -l finger | grep copies
         max_copies=5
```

「default」のスコープはこのプロパティーに表示されなくなったことに注意してください。

▼ inetd によって制御されるサービスの新しいインスタンスを追加する方法

1. **サービスのマニフェストの場所を決定します。**

   ```
   $ svcs -l FMRI
   ...
   manifest     /lib/svc/manifest/path/file.xml
   ```

2. **マニフェストファイルを編集して、適切なプロパティー値の新しいインスタンスを追加します。**

   a. **既存のインスタンス要素をコピー&ペーストします。**

b. **新しいインスタンス要素に一意のインスタンス名を指定します。**

c. **必要に応じて、プロパティー値を変更します。**

3. **新しいインスタンスをプロファイルファイルに追加します。**

   `/etc/svc/profile/generic.xml` ファイルを編集して、適切なサービスの `service` 要素内で新しい `instance` 要素を追加します。

4. **マニフェストインポートサービスを再起動します。**

   ```
   $ svcadm restart manifest-import
   ```

5. **新しいインスタンスが追加されていることを確認します。**

   `svcs -a` または `inetadm` コマンドの出力で *FMRI* を検索します。

6. **新しいインスタンスのプロパティー値を確認します。**

   ```
   $ inetadm -l FMRI
   ```

ファイルによって構成されるサービスの変更

`inetd` によって管理されないいくつかの SMF サービスは、サービスプロパティーからではなくファイルから構成の一部を取得します。構成を変更するには、構成ファイルを編集し、SMF コマンドを使用してサービスを再起動します。これらの構成ファイルは、サービスが実行しているときに変更できますが、ファイルの内容が読み取られるのは、サービスの起動時に限られます。

構成ファイルを直接編集する前に、次の条件を確認してください。

- 構成ファイルに、直接編集しないように伝えるメッセージが含まれていないか確認します。
- サービスにタイプ `configfile` のプロパティーグループがないことを確認します。

  ```
  $ svcprop -g configfile network/ntp
  ```

  サービスにタイプ `configfile` のプロパティーグループがある場合、構成ファイルではなく、これらのプロパティーグルーのプロパティーを変更します。『Oracle Solaris 12 でのシステムサービスの開発』 の 第 6 章, 「Using a Stencil to Create a Configuration File」を参照してください。

たとえば、NTP クライアントをサポートする新しい NTP サーバーを追加するには、サーバーの新しいエントリを `/etc/inet/ntp.conf` ファイルに追加してから、次のコマンドに示すように NTP サービスを再起動します。

```
$ svcadm restart svc:/network/ntp:default
```

IKEv2 を有効にするには、`/etc/inet/ike/ikev2.config` ファイルを変更して IKEv2 デーモンを構成し、次のコマンドに示すように IKEv2 サービスを有効にします。`ikev2.config` ファイルを編集するには、Network IPsec Management プロファイルが割り当てられたユーザーとして`pfedit` コマンドを使用します。このようにファイルを編集すると、正しいファイル所有権が保持されます。`pfedit` の使用については、`pfedit`(8) のマニュアルページを参照してください。

```
$ svcadm enable svc:/network/ipsec/ike:ikev2
```

♦ ♦ ♦ 第 5 章

5 複数のシステムの構成

この章では、複数のシステムにわたって構成変更を管理する推奨の方法について説明します。

システムの更新で複数のシステムに整合したシステム構成を配信する場合、IPS パッケージでの SMF プロファイルの配信がもっとも正確で効率的な方法です。

この章では、次について説明します。

- 複数のシステムの構成を管理する方法
- SMF プロファイルを作成する方法
- 構成を複数のシステムに配信する方法

複数のシステムの構成の管理

次の手順は、複数のシステムの構成を管理する方法をまとめたものです。

1. 有効にするサービスとそれらのプロパティーの値を指定する SMF プロファイルを作成します。プロファイルは、既存のサービスおよびインスタンスのプロパティーを追加および設定し、新しいサービスインスタンスを指定できます。プロファイルは、マニフェストで指定できるほとんどすべてを指定できます。
2. これらのプロファイルを各システム上の `/etc/svc/profile/enterprise` または `/etc/svc/profile/site` ディレクトリにインストールするための IPS パッケージを作成します。
   - エンタープライズ内のすべてのシステムの構成を記述したプロファイルが `/etc/svc/profile/enterprise` ディレクトリに格納されます。
   - 特定の場所またはサイト内のシステムの構成を記述したプロファイルが `/etc/svc/profile/site` ディレクトリに格納されます。
   - 1 つの Oracle Solaris システムの構成を記述したプロファイルが `/etc/svc/profile/node` ディレクトリに格納されます。

   これらのプロファイルディレクトリと構成レイヤーについては、30 ページの「リポジトリレイヤー」を参照してください。

SMF プロファイルの作成

次の手順は、SMF プロファイルファイルを作成する方法をまとめたものです。

1. SMF プロファイルファイルを作成するには、次のいずれかの方法を使用します。
   - 106 ページの「svccfg を使用したプロファイルの作成方法」の説明に従って、`svccfg extract` コマンドを使用して既存のサービスからプロファイル情報を取得します。
   - 107 ページの「sysconfig を使用してプロファイルを作成する方法」の説明に従って、`sysconfig create-profile` コマンドを使用して新しいプロファイルファイルを作成します。
   - 108 ページの「svcbundle を使用したプロファイルの作成方法」の説明に従って、`bundle-type=profile` を付けて `svcbundle` コマンドを使用して新しいプロファイルファイルを作成します。
2. プロファイルファイルのプロパティー値をカスタマイズして、それぞれのカスタマイズの理由に関するコメントを含めます。競合する構成を指定しないようにします。詳細は、104 ページの「競合する構成」を参照してください。
3. プロファイルをテストするには、プロファイルファイルを適切な `/etc/svc/profile/` レイヤーサブディレクトリにコピーし、`manifest-import` サービスを再起動します。

   次のいずれかのコマンドを使用して、プロファイルファイルに記載されている構成が適用されていることを検証します。

   - `svccfg listcust -L`
   - `svccfg listprop -l all`
   - `svcprop -l all`

競合する構成

プロファイルを作成するときには、定義される構成が、同じサービスまたはサービスインスタンスの別のプロファイルで同じレイヤー内に定義された構成と競合していないことを確認してください。

競合する構成には、同じプロパティーに異なる値を定義したり、同じプロパティーまたはプロパティーグループに異なるタイプを定義したりするものがあります。

ある単一レイヤー内の複数のファイルで構成が競合し、その構成が上位のレイヤーでは設定されていない場合、`manifest-import` サービスログにはこの競合が記され、競合している構成を使用したサービスは開始されず、`maintenance` 状態に移されます。

同じプロパティーにそれぞれ異なる値を指定する 2 つのプロファイルファイルを作成した場合は、これらのプロファイルファイルの両方を同じ `/etc/svc/profile/` レイ

ヤーサブディレクトリに配置してシステムをリブートすると、次のような結果が得られます。

- 通知を構成している場合は、通知によって競合が報告されます。次の例は、サービス状態の遷移の通知です。

```
HOSTNAME: host
TIMESTAMP: June 27, 2016 at 04:15:43 PM PDT
FMRI: svc:/site/example:default
FROM-STATE: uninitialized
TO-STATE: maintenance
DESCRIPTION: The indicated service has transitioned to the maintenance state
REASON: the instance is in conflict
```

次の例は、FMA イベント通知です。「Impact」と「Recommended Action」のセクションの情報を書きとめてください。

```
SUNW-MSG-ID: SMF-8001-02, TYPE: Defect, VER: 1, SEVERITY: Major
EVENT-TIME: Mon Jun 27 16:14:07 PDT 2016
PLATFORM: Ultra 24, CSN: 0817FMB003, HOSTNAME: host
SOURCE: software-diagnosis, REV: 0.2
EVENT-ID: c46b48bb-484d-4c9f-a82b-e1349a0ddce6
DESC: The Solaris Service manager tried to import a manifest or apply a profile defining
the service, but detected one or more entities with conflicting definitions.
AUTO-RESPONSE: The service may have been placed into the maintenance state.
IMPACT: The service is not running.  It will not be started until the conflict is
 resolved
and the maintenance state is cleared.  Services with require-type dependencies on the
service will not be started.  (Use 'svcs -xv svc:/site/nm_example:default' to see a list
of services waiting for the service to start.)
REC-ACTION: Edit the problematic manifest or profile to resolve the conflict and
 reimport
or reapply it, or use svccfg to administratively override the conflicting definitions.
Then clear the maintenance state. Please refer to the associated reference document at
http://support.oracle.com/msg/SMF-8001-02 for the latest service procedures and policies
regarding this diagnosis.
```

- `svcs -x` コマンドは競合を報告します。問題がより簡単に見つかるように、競合が見つかったすべての場所が一覧表示されます。次の出力行は、読みやすさのために改行されています。

```
$  svcs -x
svc:/site/example:default (site/example)
 State: maintenance since Wed Nov 16 15:29:51 2016
Reason: Instance has conflicts.
Conflicting value: FMRI="svc:/site/example"; Name of conflicting property="config/
prop1";
```

```
from file="/etc/svc/profile/node/example_prof1.xml"; from file="/etc/svc/profile/site/
example_prof2.xml";
Conflicting value: FMRI="svc:/site/example"; Name of conflicting property="config/
prop2";
from file="/etc/svc/profile/node/example_prof1.xml";
   See: http://support.oracle.com/msg/SMF-8001-02
   See: /var/svc/log/site-example:default.log
Impact: This service is not running.
```

- `manifest-import` サービスログファイルも競合を報告します。次の出力行は、読みやすさのために改行されています。

```
$ svcs -xL manifest-import
svc:/system/manifest-import:default (service manifest import)
 State: online since Wed Nov 16 15:29:52 2016
   See: smf_bootstrap(7)
   See: /var/svc/log/system-manifest-import:default.log
Impact: None.
   Log:
svccfg: svc:/site/example: property group "config" has a conflict.
Conflicting value: FMRI="svc:/site/example"; Name of conflicting property="config/
prop1";
from file="/etc/svc/profile/node/example_prof1.xml"; from file="/etc/svc/profile/site/
example_prof2.xml";
Conflicting value: FMRI="svc:/site/example"; Name of conflicting property="config/
prop2";
from file="/etc/svc/profile/node/example_prof1.xml";
svccfg: Multiple definitions of property group reg in entity default.
[ 2016 Nov 16 15:29:52 Method "start" exited with status 0. ]
```

`svccfg listprop` コマンドの `-l`、`-f`、および `-o` オプションを使用して、競合の原因を調べることもできます。57 ページの「値が設定されているレイヤーの表示」および58 ページの「構成に関係したファイルの表示」を参照してください。

▼ svccfg を使用したプロファイルの作成方法

1. **プロファイルを作成します。**

   `svccfg extract` コマンドは、指定したサービスまたはインスタンスのサービスプロファイルを表示します。特定のレイヤーから値を抽出するには、`-l` オプションを使用します。`-l` オプション引数には 1 つのレイヤー名を指定することも、コンマ区切りのリストで複数のレイヤー名を指定することもできます。レイヤー名の包括的なリストについては、`svccfg`(8) のマニュアルページを参照してください。レイヤーの説明は、30 ページの「リポジトリレイヤー」を参照してください。`current` および `all` レイヤー名はどちらも、それぞれのプロパティーの最上位レイヤー値を選択します。

次のコマンドは、network/dns/client サービスの各プロパティーの最上位レイヤー設定を dnsclientprofile.xml ファイルに抽出します。

```
$ svccfg extract -l current network/dns/client > dnsclientprofile.xml
```

2. **必要な変更をプロファイルに行います。**
   プロファイルの名前を意味のある名前に変更します。次の例に示すように、デフォルトでは名前は extract に設定されます。

   ```
   <service_bundle type='profile' name='extract'>
   ```

   ターゲットシステムに必要な変更を行います。

3. **プロファイルを適切なディレクトリにコピーします。**
   次のコマンドは、このプロファイルが、所定のロケールまたはサイトの全システムを対象としていることを示します。

   ```
   $ cp dnsclientprofile.xml /etc/svc/profile/site/
   ```

   必要に応じて、プロファイルの所有権とアクセス権を変更します。

4. **マニフェストインポートサービスを再起動して、プロファイルをシステムに適用します。**

   ```
   $ svcadm restart manifest-import
   ```

   manifest-import サービスログファイルを調べて、プロパティー値の競合に関するメッセージやほかのエラーメッセージを確認します。

   ```
   $ svcs -Lv manifest-import
   ```

   svcs -x コマンドを使用して、保守状態のサービスがないことを確認します。

▼ sysconfig を使用してプロファイルを作成する方法

1. **プロファイルを作成します。**
   SCI ツールを実行し、システム構成プロファイルを作成します。プロファイルのデフォルトの場所は /system/volatile/profile/sc_profile.xml です。プロファイルファイル用に別のディレクトリを指定するには、-o オプションを使用します。
   次のコマンドは、network および naming_services 構成を含む siteAprofiles ディレクトリに新しいプロファイルを作成します。構成できる機能グループ (-g) の包括的なリストについては、sysconfig(8) のマニュアルページを参照してください。

   ```
   $ sysconfig create-profile -g network,naming_services -o ./siteAprofiles
   ```

   SCI ツールが開き、構成値の入力を求めるプロンプトが表示されます。

2. **必要な変更をプロファイルに行います。**

プロファイルは ./siteAprofiles/sc_profile.xml にあります。sc_profile.xml をわかりやすい一意の名前に変更できます。

```
$ cp ./siteAprofiles/sc_profile.xml ./siteAprofiles/netnamingSiteA.xml
```

ターゲットシステムに必要な変更を行います。

3. **プロファイルを適切なディレクトリにコピーします。**

   次のコマンドは、このプロファイルが、所定のロケールまたはサイトの全システムを対象としていることを示します。

   ```
   $ cp ./siteAprofiles/netnamingSiteA.xml /etc/svc/profile/site/
   ```

   必要に応じて、プロファイルの所有権とアクセス権を変更します。

4. **マニフェストインポートサービスを再起動して、プロファイルをシステムに適用します。**

   ```
   $ svcadm restart manifest-import
   ```

   manifest-import サービスログファイルを調べて、プロパティー値の競合に関するメッセージやほかのエラーメッセージを確認します。

   ```
   $ svcs -Lv manifest-import
   ```

   svcs -x コマンドを使用して、保守状態のサービスがないことを確認します。

▼ svcbundle を使用したプロファイルの作成方法

1. **プロファイルを作成します。**

   次のコマンドは example.com.xml に新しいプロファイルを作成します。

   ```
   $ svcbundle -o example.com.xml -s service-name=enterprise/example.com \
   > -s bundle-type=profile -s instance-property=pg_name:prop_name:prop_type:value \
   > -s service-property=pg_name:prop_name:prop_type:value
   ```

2. **ターゲットシステムに必要な変更を行います。**

3. **プロファイルを適切なディレクトリにコピーします。**

   次のコマンドは、このプロファイルがすべてのシステムを対象としていることを示します。

   ```
   $ cp example.com.xml /etc/svc/profile/enterprise/
   ```

4. **マニフェストインポートサービスを再起動して、プロファイルをシステムに適用します。**

   ```
   $ svcadm restart manifest-import
   ```

`manifest-import` サービスログファイルを調べて、プロパティー値の競合に関するメッセージやほかのエラーメッセージを確認します。

```
$ svcs -Lv manifest-import
```

`svcs -x` コマンドを使用して、保守状態のサービスがないことを確認します。

複数のシステムへの構成の配信

システムにプロファイルを配信する最善の方法は、組織内で構成情報を制御する方法によって異なります。情報を異なるパッケージに分割する場合、一般に、サイト別、ネットワーク別、およびシステムの機能別に行います。たとえば、DNS および NTP 構成は、DMZ 内のすべてのシステムで同一の場合がありますが、内部開発グループで使用される構成とは異なります。その同一の DMZ 環境では、Web サーバーとして機能するすべてのシステムで共通の構成を共有できますが、この構成はほかの機能を行うシステムの構成とは異なる場合があります。

小さな構成情報セットから完全な構成を構築するために、依存関係を使用できます。依存関係を使用すると、パッケージ間での情報の重複が少なくなります。

同じ構成を必要とするシステムのグループごとに少なくとも 1 つのパッケージを作成します。1 つのパッケージで複数のプロファイルを提供できます。

- 別々の SMF レイヤーの構成は、異なるディレクトリ内の別々のプロファイルファイルで提供する必要があります。
- 単一レイヤー内では、別々のプロファイルファイルで異なるサービスの構成を提供できます。
- 異なるシステムのグループには別々の構成が必要で、したがって別々のプロファイルが必要になります。

たとえば、頻繁に変更することが予想される構成を個別に提供するために、システムグループ 1 つに複数のプロファイルパッケージを作成できます。同じシステムのグループに対する複数のプロファイルパッケージは、1 つのパッケージ内で `group` 依存関係にできます。グループパッケージを使用すると、そのグループの新しいプロファイルパッケージを簡単に提供することもできます。

プロファイルを変更する必要があるときに、更新されたプロファイルでパッケージを再構築し、パッケージバージョン番号を増分します。グループパッケージを使用する場合は、グループパッケージも更新します。更新した構成は続いて、`pkg update` でインストールされます。

プロファイルパッケージを作成する方法の詳細は、『Oracle Solaris 12 でのImage Packaging System を使用したソフトウェアのパッケージ化と配布』を参照してください。次の手順は、SMF のプロファイルパッケージに固有です。

- `/etc`、`/etc/svc`、または `/etc/svc/profile` ディレクトリや、`/etc/svc/profile` ディレクトリの標準サブディレクトリはパッケージングしないでください。これらのディレクトリはシステムパッケージですでに提供されています。
- プロファイルまたはマニフェスト `file` アクションごとに `restart_fmri` アクチュエータを含めます。

インストールしたプロファイルパッケージに更新を使用できるかどうかを確認するために、定期的なサービスを使用することを検討してください。定期的なサービスの作成の詳細は、『Oracle Solaris 12 でのシステムサービスの開発』を参照してください。

個々のシステム管理者が構成を選択できるようにする場合は、`/etc/svc/profile` ディレクトリと別の場所にあるターゲットプロファイルへの調停されたリンクを提供します。詳細は、『Oracle Solaris 12 でのImage Packaging System を使用したソフトウェアのパッケージ化と配布』 の 「Delivering Multiple Implementations of an Application」を参照してください。

♦ ♦ ♦ 付 録 A

SMF ベストプラクティスおよびトラブルシューティング

この付録では、次のことに関するベストプラクティスおよびトラブルシューティングについて説明します。

- 機能低下、オフライン、または保守状態になっているサービスインスタンスの修復
- SMF リポジトリの問題の診断と修復
- SMF 起動メッセージングの量の指定
- ブートする SMF マイルストーンの指定
- システムブート問題の調査
- SMF サービスへの `inetd` サービスの変換

SMF ベストプラクティス

ほとんどのサービスは構成ポリシーを記述します。必要な構成が実装されていない場合は、サービスを変更してポリシーの説明を変更してください。サービスプロパティーの値を変更するか、別のプロパティー値で新しいサービスインスタンスを作成します。サービスインスタンスを無効にして、SMF サービスで管理する予定の構成ファイルを編集しないでください。Oracle Solaris の増加している基本機能は SMF サービスプロパティーで構成され、構成ファイルを編集して構成するわけではありません。

Oracle またはサードパーティーソフトウェアベンダーから提供されるマニフェストおよびシステムプロファイルは、変更しないでください。これらのマニフェストおよびプロファイルは、システムをアップグレードするときに置き換えられ、これらのファイルに加えた変更は失われます。代わりに、次のいずれかを行います。

- 96 ページの「サービスインスタンスの追加」の説明に従って、別のプロパティー値を持つ新しいサービスインスタンスを追加します。
- サービスをカスタマイズするためのプロファイルを作成します。`svcbundle` コマンドまたは `svccfg extract` コマンドを使用して、プロファイルファイルを作成

します。そのファイルのプロパティー値をカスタマイズして、それぞれのカスタマイズの理由に関するコメントを含めます。プロファイルファイルを適切な `/etc/svc/profile` サブディレクトリにコピーし、`manifest-import` サービスを再起動します。

複数のシステムに同じカスタム構成を適用するには、各システム上の同じ `/etc/svc/profile` サブディレクトリに同じプロファイルファイルをコピーし、各システムで `manifest-import` サービスを再起動します。各システムへのプロファイルの配信を自動化するには、プロファイルをパッケージ化します。第5章「複数のシステムの構成」を参照してください。

- `svccfg` コマンドまたは `inetadm` コマンドを使用して、プロパティーを直接操作します。`svccfg` コマンドを使用してプロパティー値を変更する場合は、必ず、113 ページの「構成変更について」の説明に従ってサービスインスタンスをリフレッシュしてください。サービス構成の変更、追加、および削除の詳細は、第4章「サービスの構成」を参照してください。すでに変更されている構成を確認するには、59 ページの「構成カスタマイズの表示」を参照してください。カスタム構成を削除するには、例40「カスタマイズの削除」および例42「構成のマスク解除」を参照してください。

カスタムプロファイルを作成する場合は、定義される構成が、同じサービスまたはサービスインスタンスの別のマニフェストまたはプロファイルの同じレイヤーで定義されている構成と競合していないことを確認してください。構成の競合はどのレイヤーでも許可されていません。ある単一レイヤー内の複数のファイルで構成が競合し、その構成が上位のレイヤーでは設定されていない場合、`manifest-import` サービスログにはこの競合が記され、競合している構成を使用したサービスは開始されません。詳細は、104 ページの「競合する構成」を参照してください。

マニフェストおよびプロファイルのファイルには、標準以外の場所を使用しないでください。マニフェストおよびプロファイルの標準の場所については、29 ページの「サービスバンドル」を参照してください。

自身で使用するサービスを作成する場合、`svc:/site/`*service_name*`:`*instance_name* のようにサービス名の先頭に `site` を使用します。

122 ページの「起動メッセージングの量の指定」で説明しているように、ロギングレベルの構成を除き、マスターリスタータサービスの構成 (`svc:/system/svc/restarter:default`) は変更しないでください。

`svccfg delcust` コマンドを使用する前に、同じオプションを付けて `svccfg listcust` コマンドを使用してください。`delcust` サブコマンドを実行すると、サービスの管理カスタマイズがすべて削除される可能性があります。`listcust` サブコマンドを使用して、`delcust` サブコマンドでどのカスタマイズが削除されるかを検証します。

スクリプトでは、サービスインスタンスの完全な FMRI (`svc:/`*service_name*`:`*instance_name*) を使用します。

サービス問題のトラブルシューティング

このセクションの内容は次のとおりです。

- 実行中のスナップショットへの構成変更のコミット
- 問題があると報告されたサービスの修正
- `degraded` または `maintenance` の状態へのイスタンスの手動による遷移
- 破壊されたサービス構成リポジトリの修正
- システムの起動時に表示または格納するメッセージングの量の構成
- 指定したマイルストーンへの遷移またはブート
- SMF を使用したブート問題の調査
- SMF サービスへの `inetd` サービスの変換

構成変更について

SMF はサービス構成リポジトリに、実行中のスナップショットのプロパティーとは別にプロパティーの変更を格納します。サービス構成を変更した場合、実行中のスナップショットにはこれらの変更は即座に反映されません。

リフレッシュ操作では、指定したサービスインスタンスの実行中のスナップショットを、編集中の構成の値で更新します。

デフォルトでは、`svcprop` コマンドを実行すると、実行中のスナップショットのプロパティーが表示され、`svccfg` コマンドを実行すると、編集中の構成のプロパティーが表示されます。プロパティー値を変更したが、構成のリフレッシュを実行していない場合、`svcprop` コマンドと `svccfg` コマンドは別々のプロパティー値を表示します。構成のリフレッシュを実行したあとでは、`svcprop` コマンドと `svccfg` コマンドは同じプロパティー値を表示します。

リブートしても、実行中のスナップショットは変更されません。`svcadm restart` コマンドを実行しても構成はリフレッシュされません。実行中のスナップショットに構成変更をコミットするには、`svcadm refresh` コマンドまたは `svccfg refresh` コマンドを使用します。

機能低下、オフライン、または保守であるインスタンスの修復

次のいずれかの記述に一致するサービスインスタンスに関する説明を表示するには、`svcs -x` コマンドを引数なしで使用します。

- サービスは有効になっているが実行していない。
- サービスは有効になっているが、通常の容量で実行されていない。
- サービスは別の有効なサービスの実行を妨げている。
- サービスは無効になっているが、`disabled` 状態への移行を完了できない。

サービスの問題への対処方法は要約すると次のようになります。

1. 問題を診断します。最初にサービスログファイルを表示します。

   ログファイルは `/var/svc/log` および `/system/volatile` にあります。サービスログファイルはタイムスタンプおよびメソッド終了の理由を表示します。

   特定のサービスのログファイルの場所は、次のコマンドで表示されます。

   ```
   $ svcs -L service-name
   ```

   次のコマンドは、特定のサービスについてのログファイルの終わりを表示します。

   ```
   $ svcs -Lx service-name
   ```
2. 問題を修正します。
   - 複数のサービス障害が確認された場合、サービスログファイルのタイムスタンプを使用して、最初に発生した障害を探すことから始めます。
   - 障害が発生したサービスの影響を受けた依存関係を表示するには、次のコマンドを使用します。

     ```
     $ svcs -l service-name
     ```

     service-name が依存するサービスを表示するには、次のコマンドを使用します。

     ```
     $ svcs -d service-name
     ```
   - 問題の修正にサービス構成の変更が含まれる場合、サービスをリフレッシュします。
3. 影響を受けたサービスを実行中の状態に移します。

▼ 保守状態のインスタンスの修復方法

保守状態にあるサービスインスタンスが有効になっているが、実行できないか、または無効になっているが、`disabled` 状態への移行を完了できません。

1. インスタンスが保守状態になっている理由を特定します。

管理アクションがまだ完了していないためにインスタンスは `maintenance` 状態に遷移している可能性があります。インスタンスが遷移している場合、その状態は、末尾にアスタリスクが付いた `maintenance*` として表示されます。

障害後に再起動するように構成されているインスタンスは、非常に頻繁に再起動したため maintenance に格納されている場合があります。この場合、一貫して発生する障害の原因を判別する必要があります。

インスタンスに競合があるかインスタンスが競合するプロパティー値を持つため、インスタンスが maintenance に格納されている場合は、104 ページの「競合する構成」を参照してください。

このインスタンスは、インスタンスが無効にされたが、stop メソッドが失敗したため disabled 状態に到達できないために maintenance 状態にある可能性があります。

次の例の「State」および「Reason」行には、その起動メソッドが失敗したために pkg/depot サービスが maintenance 状態になっていることが示されています。

```
$ svcs -x
svc:/application/pkg/depot:default (IPS Depot)
 State: maintenance since September 11, 2013 01:30:42 PM PDT
Reason: Start method exited with $SMF_EXIT_ERR_FATAL.
   See: http://support.oracle.com/msg/SMF-8000-KS
   See: pkg.depot-config(8)
   See: /var/svc/log/application-pkg-depot:default.log
Impact: This service is not running.
```

Oracle サポートサイトにログインして、対象の予測的セルフヒーリングのナレッジ記事を確認します。この場合、記事には、ログファイルを調べて起動メソッドが失敗した理由を判断するように指示されています。svcs 出力にはログファイルの名前が示されます。ログファイルを表示する方法の詳細は、47 ページの「サービスログファイルの表示」を参照してください。この例では、ログファイルに、起動メソッドの呼び出しと致命的なエラーメッセージが示されます。

```
[ Sep 11 13:30:42 Executing start method ("/lib/svc/method/svc-pkg-depot start"). ]
pkg.depot-config: Unable to get publisher information:
The path '/export/ipsrepos/Solaris11' does not contain a valid package repository.
```

2. **問題を修正します。**

   次の手順の 1 つ以上が必要になる可能性があります。

   - **サービス構成を更新します。**

     報告された問題の修正でサービス構成の変更が必要な場合は、構成が変更されたサービスに対して svccfg refresh コマンドまたは svcadm refresh コマンドを使用します。svcprop コマンドを使用してプロパティー値を確認するか、このサービス固有のほかのテストを行なって、実行中のスナップショットで構成が更新されていることを検証します。

   - **依存関係が実行していることを確認します。**

     svcs -x 出力の「Impact」行に、maintenance 状態のサービスに依存しているサービスが実行していないことが示されることがあります。svcs -l コマンドを使用して、依存サービスの現在状態を確認します。すべての必要な依存関係が実

行していることを確認します。svcs -x コマンドを使用し、すべての有効なサービスが実行していることを検証します。

- **契約プロセスが停止していることを確認します。**

  maintenance 状態のサービスが契約サービスである場合、このサービスが開始したすべてのプロセスが停止していないかどうかを判断します。契約サービスインスタンスが保守状態である場合、次の例に示すように、契約 ID は空白にする必要があり、その契約に関連付けられたすべてのプロセスを停止している必要があります。svcs -l または svcs -o ctid を使用して、保守状態のサービスインスタンスについて契約が存在していないことを確認します。svcs -p を使用して、このサービスインスタンスに関連付けられたすべてのプロセスがまだ実行しているかどうかを確認します。保守状態のサービスインスタンスについて svcs -p で示されたプロセスはすべて、強制終了する必要があります。

```
$ svcs -l system-repository
fmri         svc:/application/pkg/system-repository:default
name         IPS System Repository
enabled      true
state        maintenance
next_state   none
state_time   September 17, 2013 07:18:19 AM PDT
logfile      /var/svc/log/application-pkg-system-repository:default.log
restarter    svc:/system/svc/restarter:default
contract_id
manifest     /lib/svc/manifest/application/pkg/pkg-system-repository.xml
dependency   require_all/error svc:/milestone/network:default (online)
dependency   require_all/none svc:/system/filesystem/local:default (online)
dependency   optional_all/error svc:/system/filesystem/autofs:default (online)
```

3. **インスタンスが修復されたことをリスタータに通知します。**

   報告された問題が修正されたら、svcadm clear コマンドを使用して、インスタンスが修復されたことをそのサービスのリスタータに通知します。SMF はインスタンスを、それが構成された状態に移行しようとします。インスタンスが有効になっている場合、SMF はインスタンスをオンラインにしようとします。インスタンスが無効になっている場合、SMF はそのインスタンスを disabled 状態に移行します。

```
$ svcadm clear pkg/depot:default
```

   -s オプションを指定した場合、svcadm コマンドは、インスタンスが online 状態に達するまで、または管理者の操作なしにインスタンスが online 状態に達せないと判断するまで待機してから戻ります。遷移を行うか、遷移を行えないと判断するまでの上限を秒単位で指定するには、-T オプションを -s オプションとともに使用します。

4. **インスタンスが修復されたことを検証します。**

   svcs コマンドを使用して、保守状態だったサービスが現在オンラインになっていることを検証します。svcs -x コマンドを使用し、すべての有効なサービスが実行していることを検証します。

▼ オフライン状態のインスタンスを修復する方法

オフラインであるサービスインスタンスが有効になっていますが、実行中でも実行可能でもありません。

1. **インスタンスがオフラインになっている理由を判断します。**

   依存関係がまだ満たされていないために、インスタンスが `offline` 状態に遷移している可能性があります。インスタンスが遷移している場合、その状態は `offline*` と表示されます。

2. **問題を修正します。**

   - **サービスの依存関係を有効にします。**

     必要な依存関係が無効になっている場合は、次のコマンドを使用して有効にします。

     ```
     $ svcadm enable -r FMRI
     ```

   - **依存関係ファイルを修正します。**

     依存関係ファイルが失われていたり、読み取れない場合があります。`pkg fix` または `pkg revert` を使用して、この種の問題を修正できます。`pkg`(1) のマニュアルページを参照してください。

3. **必要に応じてインスタンスを再起動します。**

   必要な依存関係が満たされなかったためにインスタンスがオフラインだった場合、依存関係を修正または有効にすると、オフラインのインスタンスが再起動し、管理アクションを加えなくてもオンラインになることがあります。

   サービスにほかの修正を行なった場合は、インスタンスを再起動してください。

   ```
   $ svcadm restart FMRI
   ```

4. **インスタンスが修復されたことを検証します。**

   `svcs` コマンドを使用して、オフライン状態だったインスタンスが現在オンラインになっていることを検証します。`svcs -x` コマンドを使用し、すべての有効なサービスが実行していることを検証します。

▼ 機能低下状態のインスタンスを修復する方法

機能低下状態にあるサービスインスタンスが有効になっており、実行中または実行可能ですが、制限された容量で実行されています。

1. **インスタンスが機能低下状態になっている理由を判断します。**

2. **問題を修正します。**

3. **インスタンスをオンラインにするようリスタータに要求します。**

   報告された問題が修正されたら、`svcadm clear` コマンドを使用して、インスタンスを `online` 状態に戻します。`degraded` 状態のインスタンスの場合、`clear` サブコマンドは、このインスタンスのリスタータがインスタンスを `online` 状態に遷移させることを要求します。

   ```
   $ svcadm clear pkg/depot:default
   ```

4. **インスタンスが修復されたことを検証します。**

   `svcs` コマンドを使用して、機能低下状態だったインスタンスが現在オンラインになっていることを検証します。`svcs -x` コマンドを使用し、すべての有効なサービスが実行していることを検証します。

機能低下または保守状態としてのインスタンスのマーキング

`degraded` または `maintenance` のどちらかの状態としてサービスインスタンスをマークできます。たとえば、アプリケーションがループでスタックしていたり、デッドロックされている場合に、これを行えます。状態変更に関する情報は、マークされたインスタンスの依存関係に伝播されるので、ほかの関連インスタンスのデバッグに役立ちます。

即座の状態変更を要求するには、`-I` オプションを指定します。

インスタンスに `maintenance` のマークを付けるときに、`-t` オプションを指定して、一時的な状態変更を要求できます。一時的な要求はリブートまでに限り継続します。

`svcadm mark` とともに `-s` オプションを指定すると、`svcadm` はインスタンスにマークを付け、インスタンスが `degraded` または `maintenance` の状態になるまで待機してから戻ります。遷移を行うか、遷移を行えないと判断するまでの上限を秒単位で指定するには、`-T` オプションを `-s` オプションとともに使用します。

リポジトリの問題の診断と修復

システムの起動時に、リポジトリデーモン `svc.configd` は、`/etc/svc/repository.db` に格納された構成リポジトリの整合性チェックを実行します。`svc.configd` 整合性チェックに失敗すると、`svc.configd` デーモンは次のようなメッセージをコンソールに書き出します。

```
svc.configd: smf(7) database integrity check of:

    /etc/svc/repository.db

  failed.  The database might be damaged or a media error might have
  prevented it from being verified.  Additional information useful to
  your service provider is in:

    /system/volatile/db_errors

  The system will not be able to boot until you have restored a working
  database.  svc.startd(8) will provide a sulogin(8) prompt for recovery
  purposes.  The command:

    /lib/svc/bin/restore_repository

  can be run to restore a backup version of your repository. See
  http://support.oracle.com/msg/SMF-8000-MY for more information.
```

その後、`svc.configd` デーモンは終了します。この終了が `svc.startd` デーモンによって検出されると、`svc.startd` が `sulogin` を起動します。

`sulogin` プロンプトで Ctrl-D を入力して `sulogin` を入力します。`svc.startd` デーモンは `sulogin` の終了を認識して `svc.configd` デーモンを再起動し、これによってリポジトリがもう一度チェックされます。この再起動後に、問題が再現されなくなる可能性があります。

注意 - `svc.configd` デーモンを直接呼び出さないでください。`svc.startd` デーモンは `svc.configd` デーモンを起動します。

`svc.configd` が整合性チェックの失敗を再度報告し、ふたたび `sulogin` プロンプトが表示されている場合は、要求されたファイルシステムが満杯になっていないことを確認してください。`root` パスワードを使用して、リモートから、または `sulogin` プロンプトからログインします。ルートファイルシステムと `system/volatile` ファイルシステムの両方に、使用できる領域があることを確認します。これらのファイルシステムのどちらかが満杯の場合、システムをクリーンアップして再度起動します。これらのファイルシステムがどちらも満杯になっていない場合は、120 ページの「バックアップからリポジトリを復元する方法」の手順に従ってください。

サービス構成リポジトリは、次のいずれかの理由で破損されることがあります。

- ディスク障害
- ハードウェアのバグ
- ソフトウェアのバグ
- 過失によるファイルの上書き

次の手順は、破損したリポジトリをバックアップコピーしたリポジトリと交換する方法を示しています。

▼ バックアップからリポジトリを復元する方法

始める前に

注意 - 破損したリポジトリだけを復元します。不要な構成変更を削除する場合に、このリポジトリ復元手順を使用しないでください。構成の変更を元に戻すには、59 ページの「構成カスタマイズの表示」、例40「カスタマイズの削除」、および例42「構成のマスク解除」を参照してください。

1. **ログインします。**

   `root` パスワードを使用して、リモートから、または `sulogin` プロンプトからログインします。

2. **次のリポジトリ復元コマンドを実行します。**

   ```
   # /lib/svc/bin/restore_repository
   ```

   このコマンドを実行すると、破壊されていないバックアップの復元に必要な手順が示されます。SMF は、32 ページの「リポジトリのバックアップ」で説明しているように、自動的にリポジトリのバックアップを作成します。

   SMF は、永続および非永続構成データを保持します。これらの 2 つのリポジトリの詳細は、28 ページの「サービス構成リポジトリ」を参照してください。`restore_repository` コマンドは、永続リポジトリだけを復元します。`restore_repository` コマンドはシステムのリブートも行いますが、これにより非永続構成データは破棄されます。非永続データは、システムリブートのあとは不要になる実行時データです。

   `/lib/svc/bin/restore_repository` コマンドが起動すると、次のようなメッセージが表示されます。

   ```
   See http://support.oracle.com/msg/SMF-8000-MY for more information on the use of
   this script to restore backup copies of the smf(7) repository.

   If there are any problems which need human intervention, this script will
   give instructions and then exit back to your shell.
   ```

   書き込み権を付けてルート (/) ファイルシステムをマウントしたあと、またはシステムがローカルゾーンである場合は、復元するリポジトリのバックアップを選択するよう求められます。

   ```
   The following backups of /etc/svc/repository.db exists, from
   oldest to newest:

   ... list of backups ...
   ```

   バックアップには、バックアップのタイプとバックアップが作成された時間に基づいて名前が付けられています。`boot` で始まっているのは、システムのブート後、リポジトリに対して最初の変更が行われる前に作成されたバックアップです。`manifest_import` で始まっているのは、`svc:/system/manifest-import:default` のプロセス終了後に作成されたバックアップです。バックアップ時間は、*YYYYMMDD_HHMMSS* 形式で記録されます。

3. **適切な応答を入力します。**

   通常は、最新のバックアップオプションを選択します。

```
Please enter either a specific backup repository from the above list to
restore it, or one of the following choices:

        CHOICE            ACTION
        ----------------  --------------------------------------------
        boot              restore the most recent post-boot backup
        manifest_import   restore the most recent manifest_import backup
        -seed-            restore the initial starting repository  (All
                            customizations will be lost, including those
                            made by the install/upgrade process.)
        -quit-            cancel script and quit

Enter response [boot]:
```

   復元するバックアップを指定しないで Enter を押した場合は、`[]` で囲まれたデフォルトの応答が選択されます。`-quit-` を選択すると、`restore_repository` スクリプトが終了して、シェルスクリプトに戻ります。

注意 - `-seed-` を選択すると、`seed` リポジトリが復元されます。このリポジトリは、初期インストールとアップグレード時に使用する目的で作成されたものです。ほかのサービス構成変更またはバックアップサービスリポジトリが機能していない場合、回復の目的に `seed` リポジトリだけを使用します。パッケージのインストールまたは更新でもたらされた Oracle Solaris の基本機能に対する変更を含め、すべての構成変更が失われます。`seed` リポジトリを回復の目的で使用するのは、最後の手段にしてください。

   復元するバックアップを選択したあとで、そのバックアップが検証され、その整合性がチェックされます。なんらかの問題が検出されると、`restore_repository` コマンドによってエラーメッセージが出力され、別の選択を行うように促されます。有効なバックアップを選択すると、次の情報が出力され、最終確認を入力するよう促されます。

```
After confirmation, the following steps will be taken:

svc.startd(8) and svc.configd(8) will be quiesced, if running.
/etc/svc/repository.db
    -- renamed --> /etc/svc/repository.db_old_YYYYMMDD_HHMMSS
/system/volatile/db_errors
    -- copied --> /etc/svc/repository.db_old_YYYYMMDD_HHMMSS_errors
repository_to_restore
    -- copied --> /etc/svc/repository.db
and the system will be rebooted with reboot(8).

Proceed [yes/no]?
```

4. **yes と入力して障害を修復します。**

   `restore_repository` コマンドが表示されたアクションをすべて実行すると、システムがリブートします。

起動メッセージングの量の指定

デフォルトでは、システムブート中に起動した各サービスは、コンソールにメッセージを表示しません。コンソールに表示するメッセージと、`svc.startd` ログファイルに記録するだけのメッセージを変更するには、次のいずれかの方法を使用します。*logging-level* の値には、下の表に示したいずれかの値を指定できます。

- SPARC システムをブートするときに、`ok` プロンプトで `boot` コマンドに `-m` オプションを指定します。`kernel`(8) のマニュアルページの「メッセージオプション」を参照してください。

  ```
  ok boot -m logging-level
  ```

- x86 システムをブートするときに、GRUB メニューを編集して `-m` オプションを指定します。『Oracle Solaris 12 システムのブートとシャットダウン』 の 「Adding Kernel Arguments at Boot Time」および `kernel`(8) のマニュアルページの「メッセージオプション」を参照してください。
- システムをリブートする前に、`svccfg` コマンドを使用して、`options/logging` プロパティーの値を変更します。このプロパティーがこのシステムで変更されたことがない場合は終了せず、ユーザーがこれを追加する必要があります。次の例では、詳細メッセージングに変更します。この変更は、`svc.startd` デーモンを次回再起動するときに有効になります。

  ```
  $ svccfg -s system/svc/restarter:default listprop options/logging
  $ svccfg -s system/svc/restarter:default addpg options application
  $ svccfg -s system/svc/restarter:default setprop options/logging=verbose
  $ svccfg -s system/svc/restarter:default listprop options/logging
  options/logging astring     verbose
  ```

表 2 SMF 起動メッセージロギングレベル

ロギングレベルのキーワード	説明
`quiet`	管理者の操作を必要とするエラーメッセージをコンソールに表示します。また、`syslog` および `/var/svc/log/svc.startd.log` にこれらのメッセージを記録します。
`verbose`	`quiet` レベルで行われるメッセージングに加え、起動した各サービスの単一のメッセージをコンソールに表示し、管理者の操作を必要としないエラーに関する情報を `/var/svc/log/svc.startd.log` に記録します。
`debug`	`quiet` レベルで行われるメッセージングに加え、起動した各サービスの単一のメッセージをコンソールに表示し、すべての `svc.startd` デバッグメッセージを `/var/svc/log/svc.startd.log` に記録します。

ブート先の SMF マイルストーンの指定

システムをブートするときに、ブート先の SMF マイルストーンを指定できます。

デフォルトでは、`general/enabled` プロパティーの値が `true` であるすべてのサービスがシステムのブート時に起動されます。システムのブート先のマイルストーンを変更するには、次のいずれかの方法を使用します。*milestone* の値には、表3に示すように、マイルストーンサービスの FMRI またはキーワードを指定できます。

- SPARC システムをブートするときに、`ok` プロンプトで `boot` コマンドに `-m` オプションを指定します。`kernel`(8) のマニュアルページの `-m` オプションを参照してください。

  ```
  ok boot -m milestone=milestone
  ```

- x86 システムをブートするときに、GRUB メニューを編集して `-m` オプションを指定します。『Oracle Solaris 12 システムのブートとシャットダウン』 の 「Adding Kernel Arguments at Boot Time」および `kernel`(8) のマニュアルページの `-m` オプションを参照してください。
- システムのリブート前に、`-d` オプションを付けて `svcadm milestone` コマンドを使用します。`-d` オプションを付けても付けなくても、このコマンドは実行中のサービスを即座に制限して復元することに注意してください。`-d` オプションを使用した場合、このコマンドはまた、指定したマイルストーンをデフォルトのブートマイルストーンに設定します。この新しいデフォルトは、リブートのあとも持続します。

  ```
  $ svcadm milestone -d milestone
  ```

  このコマンドは、システムの現在の実行レベルを変更しません。システムの現在の実行レベルを変更するには、`init` コマンドを使用します。

  `-s` オプションを指定した場合、`svcadm` はマイルストーンを変更し、指定されたマイルストーンへの遷移が完了するまで待機してから戻ります。`svcadm` コマンドは、すべてのインスタンスが、指定したマイルストーンに達するために必要な状態に遷移したとき、または遷移を行うために管理者の操作が必要であると判断したときに戻ります。`-s` オプションとともに `-T` オプションを使用して、マイルストーンの変更操作を完了するか戻るまでの上限を秒単位で指定します。

次の表では、対応する Oracle Solaris 実行レベルを含め、SMF ブートマイルストーンについて説明します。システムの実行レベルによって、ユーザーが利用できるサービスおよびリソースが定まります。システムが一度に持つことのできる実行レベルは 1 つだけです。実行レベルの詳細については、『Oracle Solaris 12 システムのブートとシャットダウン』 の 「How Run Levels Work」、`inittab`(5)のマニュアルページ、および`/etc/init.d/README` ファイルを参照してください。SMF ブートマイルストーンの詳細は、`svcadm`(8) のマニュアルページの `milestone` サブコマンドを参照してください。

表 3 SMF ブートマイルストーンおよび対応する実行レベル

SMF マイルストーン FMRI またはキーワード	対応する実行レベル	説明
none		none キーワードは、マスターリスタータを除き、どのサービスも実行していないマイルストーンを表します。none を指定すると、`svc:/system/svc/restarter:default` を除いたすべてのサービスが一時的に無効になります。 none マイルストーンは、起動時の問題をデバッグするときに役立つことがあります。具体的な手順については、125 ページの「システムブート時のサービスの起動に関する問題を調査する方法」を参照してください。
`all`		`all` キーワードは、すべてのサービスに依存するマイルストーンを表します。`all` を指定した場合、すべてのサービスに対し一時的な enable および disable 要求は無視されます。これは、`svc.startd` で使用されるデフォルトのマイルストーンです。
`svc:/milestone/single-user`	s または S	`svc:/milestone/single-user:default` と、それが直接または間接的に依存するすべてのサービスに対する、一時的な enable および disable 要求を無視します。ほかのすべてのサービスを一時的に無効にします。
`svc:/milestone/multi-user`	2	`svc:/milestone/multi-user:default` と、それが直接または間接的に依存するすべてのサービスに対する、一時的な enable および disable 要求を無視します。ほかのすべてのサービスを一時的に無効にします。
`svc:/milestone/multi-user-server`	3	`svc:/milestone/multi-user-server:default` と、それが直接または間接的に依存するすべてのサービスに対する、一時的な enable および disable 要求を無視します。ほかのすべてのサービスを一時的に無効にします。

システムが現在ブートされているマイルストーンを判断するには、`svcs` コマンドを使用します。次の例では、システムは実行レベル 3 (`milestone/multi-user-server`) にブートされています。

```
$ svcs 'milestone/*'
STATE          STIME    FMRI
online          9:08:05 svc:/milestone/unconfig:default
online          9:08:06 svc:/milestone/config:default
online          9:08:07 svc:/milestone/devices:default
online          9:08:25 svc:/milestone/network:default
online          9:08:31 svc:/milestone/single-user:default
online          9:08:51 svc:/milestone/name-services:default
online          9:09:13 svc:/milestone/self-assembly-complete:default
online          9:09:23 svc:/milestone/multi-user:default
online          9:09:24 svc:/milestone/multi-user-server:default
```

SMF を使用したシステムブート問題の調査

このセクションでは、システムがブート中にハングアップした場合、または主要サービスがブート中に起動できなかった場合に行うアクションについて説明します。

▼ システムブート時のサービスの起動に関する問題を調査する方法

システムブートでサービスを起動するときに問題が発生した場合、ブート中にシステムがハングアップすることがあります。この手順では、ブート時に発生するサービス問題を調査する方法を示します。

1. **どのサービスも起動しないでブートします。**
   次のコマンドを実行すると、`svc.startd` デーモンはすべてのサービスを一時的に無効にし、コンソール上で `sulogin` を起動します。

   ```
   ok boot -m milestone=none
   ```

   `boot -m` コマンドとともに使用できる SMF マイルストーンのリストについては、123 ページの「ブート先の SMF マイルストーンの指定」を参照してください。

2. **システムに `root` としてログインします。**

3. **すべてのサービスを有効にします。**

   ```
   # svcadm milestone all
   ```

4. **ブートプロセスがどこでハングアップするのかを確認します。**
   ブートプロセスがハングアップしたら、動作していないサービスを確認するために、`svcs -a` を実行します。`/var/svc/log` のログファイル内でエラーメッセージの有無を確認します。

5. **問題が解決したら、すべてのサービスが起動していることを確認します。**

   a. **必要なサービスがすべてオンラインになっていることを確認します。**

      ```
      # svcs -x
      ```

   b. **`console-login` サービスの依存関係に問題がないことを確認します。**
      このコマンドを使えば、コンソール上の `login` プロセスが実行されるかどうかを確認できます。

      ```
      # svcs -l system/console-login:default
      ```

6. **通常のブートプロセスを継続します。**

▼ ブート中にローカルファイルシステムサービスが失敗した場合に、シングルユーザーログインを強制する方法

システムのブートに必要でないローカルファイルシステムは、`svc:/system/filesystem/local:default` サービスによってマウントされます。それらのすべての

ファイルシステムをマウントできない場合、filesystem/local サービスは保守状態になります。システムの起動は続行し、filesystem/local に依存していないサービスが起動します。filesystem/local サービスへの必要な依存関係があるサービスは起動しません。

次の手順では、サービスで障害が発生した場合に、システムの起動の続行を許可する代わりに、ただちに sulogin プロンプトを表示するようシステムの構成を変更する方法について説明します。

1. **system/console-login サービスを変更します。**

```
$ svccfg -s svc:/system/console-login
svc:/system/console-login> addpg site,filesystem-local dependency
svc:/system/console-login> setprop site,filesystem-local/entities = fmri: svc:/system/filesystem/
local
svc:/system/console-login> setprop site,filesystem-local/grouping = astring: require_all
svc:/system/console-login> setprop site,filesystem-local/restart_on = astring: none
svc:/system/console-login> setprop site,filesystem-local/type = astring: service
svc:/system/console-login> end
```

2. **サービスをリフレッシュします。**

```
$ svcadm refresh console-login
```

system/filesystem/local:default サービスで障害が発生した場合は、svcs -vx コマンドを使用してその障害を識別します。障害が修正されたあと、次のコマンドを使用してエラー状態をクリアし、システムブートが続行できるようにします。

```
$ svcadm clear filesystem/local
```

SMF サービスへの inetd サービスの変換

システム上の inetd.conf ファイルにはエントリを含めないでください。inetd.conf ファイルには、これがすでに直接使用されていない旧バージョンのファイルであることを示すコメントだけを含めてください。inetd.conf ファイルにエントリが含まれている場合、このセクションの説明に従って、これらの構成を SMF サービスに変換してください。inetd.conf ファイルには構成されているが、SMF サービスとしては構成されていないサービスは使用できません。inetd.conf ファイルに構成されているサービスは、inetd コマンドでは直接再起動されません。inetd コマンドはどちらかと言えば、変換したサービス用の機能低下版のリスタータです。

初期システムブート中、inetd.conf ファイル内の構成は自動的に SMF サービスに変換されます。初期システムブート後に、Image Packaging System (IPS) パッケージングで提供されていない追加ソフトウェアをインストールすることによって、inetd.conf ファイルにエントリを追加できます。IPS パッケージで提供されるソフトウェアには必要な SMF マニフェストがすべて含まれており、その SMF マニフェストはサービスインスタンスを適切なプロパティー値でインスタンス化します。

システム上の inetd.conf ファイルにエントリが含まれている場合、inetconv コマンドを使用して、これらの構成を SMF サービスに変換します。inetconv コマンドは、inetd.conf エントリを SMF サービスマニフェストファイルに変換し、サービスインスタンスをインスタンス化するためにこれらのマニフェストを SMF リポジトリにインポートします。コマンドオプションおよびこのコマンドの使用例については、inetconv(8) のマニュアルページを参照してください。

新しい SMF マニフェストの名前には、inetd.conf エントリの *service_name* が組み込まれます。inetd.conf ファイルからのエントリは、新しいサービスインスタンスのプロパティーとして保存されます。新しい SMF マニフェストはプロパティーグループとプロパティーを指定して、inetd.conf エントリに一覧表示されたアクションを定義します。inetconv コマンドの実行後、svcs および svcprop コマンドを使用して、新しいサービスインスタンスが作成され、正しいプロパティー値が設定されていることを確認します。

inetd コマンドは、SMF インターネットサービスの機能低下版のリスタータです。これらのサービスの管理に、inetd コマンドを直接使用しないでください。inetd で制御されるサービスのリストを表示するには、オプションもオペランドも付けずに inetadm コマンドを使用してください。これらの変換したサービスを構成および管理するには、inetadm、svcadm、および svccfg コマンドを使用します。

inetconv コマンドは、inetd.conf 入力ファイルを変更しません。inetconv の実行が成功したあとで、inetd.conf ファイル内のエントリを手動で削除する必要があります。

すでに SMF サービスに変換されている inetd サービスの構成の詳細は、98 ページの「inetd で制御されるサービスの変更」を参照してください。

索引

た

は

L

M

N

O

P

R

S

U

Z