Sun Cluster の概念 (Solaris OS 版)

Sun Microsystems, Inc.
4150 Network Circle
Santa Clara, CA 95054
U.S.A.

Part No: 819–2061–10
2005 年 8 月, Revision A

原典: Sun Cluster Concepts Guide for Solaris OS

Part No: 819-0421–10

Revision A

Adobe PostScript

050815@12762

目次

はじめに

『Sun™ Cluster の概念 (Solaris OS 版)』では、SPARC® と x86 の両方の環境における SunPlex™ システムの概念と参照情報について説明します。

注 – このマニュアルでは、「x86」という用語は、Intel 32 ビット系列のマイクロプロセッサチップ、および AMD が提供する互換マイクロプロセッサチップを意味します。

SunPlex システムでは、Sun のクラスタソリューションを構成するすべてのハードウェア/ソフトウェアコンポーネントがサポートされます。

このマニュアルは、Sun Cluster ソフトウェアについて訓練を受けた、経験豊富なシステム管理者を対象としています。販売活動のガイドとしては使用しないでください。このマニュアルを読む前に、システムの必要条件を確認し、適切な装置とソフトウェアを用意しておく必要があります。

このマニュアルで説明されている概念を理解するには、Solaris™ オペレーティングシステムに関する知識と、Sun Cluster システムと共に使用するボリューム管理ソフトウェアに関する専門知識が必要です。

注 – Sun Cluster ソフトウェアは、SPARC と x86 の 2 つのプラットフォーム上で稼動します。このマニュアル内の情報は、章、節、注、箇条書き項目、図、表、または例などで特に明記されていない限り両方に適用されます。

表記上の規則

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

表 **P–1** 表記上の規則

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

コード例は次のように表示されます。

- C シェル

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

- C シェルのスーパーユーザー

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

- Bourne シェルおよび Korn シェル

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

- Bourne シェルおよび Korn シェルのスーパーユーザー

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

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

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

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

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

関連マニュアル

関連のある Sun Cluster のトピックについては、次の表に示したマニュアルを参照してください。Sun Cluster のマニュアルはすべてhttp://docs.sun.com から利用できます。

トピック	マニュアル
概要	『Sun Cluster の概要 (Solaris OS 版)』
概念	『Sun Cluster の概念 (Solaris OS 版)』
ハードウェアの設計と管理	『Sun Cluster 3.0-3.1 Hardware Administration Manual for Solaris OS』 各ハードウェア管理ガイド
ソフトウェアのインストール	『Sun Cluster ソフトウェアのインストール (Solaris OS 版)』
データサービスのインストールと管理	『Sun Cluster データサービスの計画と管理 (Solaris OS 版)』 各データサービスガイド
データサービスの開発	『Sun Cluster データサービス開発ガイド (Solaris OS 版)』
システム管理	『Sun Cluster のシステム管理 (Solaris OS 版)』
エラーメッセージ	『Sun Cluster Error Messages Guide for Solaris OS』
コマンドと関数のリファレンス	『Sun Cluster Reference Manual for Solaris OS』

Sun Cluster のマニュアルの完全なリストについては、お使いの Sun Cluster ソフトウェアのリリースノートを http://docs.sun.com で参照してください。

マニュアル、サポート、およびトレーニング

Sun のサービス	URL	内容
マニュアル	`http://jp.sun.com/documentation/`	PDF 文書および HTML 文書をダウンロードできます。
サポートおよびトレーニング	`http://jp.sun.com/supportraining/`	技術サポート、パッチのダウンロード、および Sun のトレーニングコース情報を提供します。

問い合わせについて

Sun Cluster システムのインストールや使用に関して問題がある場合は、以下の情報をご用意の上、担当のサービスプロバイダにお問い合わせください。

- 名前と電子メールアドレス (利用している場合)
- 会社名、住所、および電話番号
- システムのモデルとシリアル番号
- オペレーティング環境のリリース番号 (例: Solaris 9)
- Sun Cluster ソフトウェアのバージョン番号 (例: 3.1 8/05)

次のコマンドを使用し、システム上の各ノードに関して、サービスプロバイダに必要な情報を収集してください。

コマンド	機能
`prtconf -v`	システムメモリーのサイズと周辺デバイス情報を表示します
`psrinfo -v`	プロセッサの情報を表示する
`showrev -p`	インストールされているパッチを報告する
SPARC:`prtdiag -v`	システム診断情報を表示する
`scinstall -pv`	Sun Cluster ソフトウェアのリリースおよびパッケージのバージョン情報を表示する

コマンド	機能
`scstat`	クラスタの状態のスナップショットを提供します
`scconf -p`	クラスタ構成情報を表示します
`scrgadm -p`	インストールされているリソースやリソースグループ、リソースタイプの情報を表示する

上記の情報にあわせて、`/var/adm/messages` ファイルの内容もご購入先にお知らせください。

製品のトレーニング

Sun Microsystems は、Sun の数多くの技術を学ぶことができるトレーニングコースを用意しています。インストラクター付きのコースから自分で学べるコースまで、さまざまな種類のコースがあります。Sun が提供するトレーニングコースや入会方法については、`http://training.sun.com/` の Sun Microsystems Training をご覧ください。

第 1 章

基本知識と概要

Sun Cluster システムはハードウェアと Sun Cluster ソフトウェアが統合されたソリューションであり、高度な可用性とスケーラビリティーを備えたサービスを提供するために使用されます。

このマニュアルでは、Sun Cluster のマニュアルの読者に必要な概念について説明します。次の読者を対象としています。

- クラスタハードウェアを設置して保守を行う担当者
- Sun Cluster ソフトウェアをインストール、構成、管理するシステム管理者
- 現在 Sun Cluster 製品に含まれていないアプリケーション用のフェイルオーバーサービスやスケーラブルサービスを開発するアプリケーション開発者

このマニュアルは、Sun Cluster の他のマニュアルと合わせて、Sun Cluster システムの全体を説明するものです。

この章では、次の内容について説明します。

- Sun Cluster システムの基本知識と概要
- 各ユーザーから見た Sun Cluster システム
- Sun Cluster で作業するにあたって理解する必要がある重要な概念
- 重要な概念に関連する手順と情報を記載した Sun Cluster のマニュアル
- クラスタに関連する作業と、これらの作業手順が記載されたマニュアル

Sun Cluster システムの紹介

Sun Cluster システムは、Solaris オペレーティングシステムをクラスタオペレーティングシステムに拡張するものです。クラスタまたは plex とは、緩やかに結合された処理ノードの集合のことで、データベース、Web サービス、ファイルサービスなどのネットワークサービスやアプリケーションを、クライアントからは 1 つのシステムに見える形で提供します。

各クラスタノードは、それ自身のプロセスを実行するスタンドアロンサーバーです。これらのプロセスは、相互にやりとりすることによって、ユーザーに提供するアプリケーション、システムリソース、データを (ネットワーククライアントにとって) 1 つのシステムのように形成します。

クラスタには、従来の単一サーバーシステムと比較した場合、いくつかの利点があります。これらの利点には、フェイルオーバーサービスとスケーラブルサービスのサポート、モジュールの成長に対応できる容量、従来のハードウェアフォルトトレラントシステムよりも低価格の製品といったものがあります。

次に、Sun Cluster システムの目的を示します。

- ソフトウェアまたはハードウェアの障害が原因のシステム停止時間を短縮、または完全になくします。
- 単一サーバーシステムを停止させるような障害が発生しても、エンドユーザーへのデータとアプリケーションの可用性を保証します。
- クラスタにノードを追加し、追加したプロセッサに応じたサービスを提供できるようにすることで、アプリケーションのスループットを向上させます。
- クラスタ全体を停止しなくても保守を実行できるようにすることで、システムの可用性を強化します。

フォルトトレラント機能と高可用性についての詳細は、『Sun Cluster の概要 (Solaris OS 版)』の「Sun Cluster によるアプリケーションの可用性の向上」を参照してください。

高可用性の FAQ については、91 ページの「高可用性に関する FAQ」を参照してください。

各ユーザーから見た Sun Cluster システム

この節では、Sun Cluster システムのユーザーを 3 種類に分け、各ユーザーに関連する概念とマニュアルについて説明します。各ユーザーは次のとおりです。

- ハードウェア保守担当者
- システム管理者
- アプリケーション開発者

ハードウェア保守担当者

ハードウェア保守担当者にとって、Sun Cluster システムは、サーバー、ネットワーク、および記憶装置を含む市販のハードウェアの集合に見えます。これらのコンポーネントは、すべてのコンポーネントにバックアップがあり、単一の障害によってシステム全体が停止しないように配線されています。

重要な概念 (ハードウェア保守担当者)

ハードウェア保守担当者は、クラスタに関する次の概念を理解する必要があります。

- クラスタハードウェアの構成と配線
- 設置と保守 (追加、取り外し、交換)
  - ネットワークインタフェースコンポーネント (アダプタ、接続点、ケーブル)
  - ディスクインタフェースカード
  - ディスクアレイ
  - ディスクドライブ
  - 管理コンソールとコンソールアクセスデバイス
- 管理コンソールとコンソールアクセスデバイスの設定

参照箇所 (ハードウェア保守担当者)

次の項には、前述の重要な概念に関連する説明が記載されています。

- 22 ページの「クラスタノード」
- 24 ページの「多重ホストデバイス」
- 26 ページの「ローカルディスク」
- 26 ページの「クラスタインターコネクト」
- 27 ページの「パブリックネットワークインタフェース」
- 28 ページの「クライアントシステム」
- 28 ページの「管理コンソール」
- 28 ページの「コンソールアクセスデバイス」
- 30 ページの「SPARC: クラスタペアトポロジ」
- 31 ページの「SPARC: N+1 (星形) トポロジ」

関連マニュアル (ハードウェア保守担当者)

次の Sun Cluster のマニュアルには、ハードウェア保守の概念に関連する手順と情報が記載されています。

『Sun Cluster 3.0-3.1 Hardware Administration Manual for Solaris OS』

システム管理者

システム管理者にとって、Sun Cluster システムは、ケーブルによって接続された、記憶装置を共有するサーバー (ノード) の集合に見えます。システム管理者は、次の作業を行うソフトウェアを扱います。

- クラスタノード間のコネクティビティーを監視するための、Solaris ソフトウェアに統合された専用のクラスタソフトウェア
- クラスタノードで実行されるユーザーアプリケーションプログラムの状態を監視するための専用のソフトウェア
- ディスクを設定して管理するためのボリュームマネージャー
- 直接ディスクに接続されていないものも含め、すべてのノードが、すべての記憶装置にアクセスできるようにするための専用のクラスタソフトウェア
- ファイルがすべてのノードに対してローカルに接続されているように表示するための専用のソフトウェア

重要な概念 (システム管理者)

システム管理者は、次の概念とプロセスについて理解する必要があります。

- ハードウェアとソフトウェアの間の対話
- クラスタをインストールして構成する方法の一般的な流れ
  - Solaris オペレーティングシステムのインストール
  - Sun Cluster ソフトウェアのインストールと構成
  - ボリュームマネージャーのインストールと構成
  - クラスタを動作可能状態にするためのアプリケーションソフトウェアのインストールと構成
  - Sun Cluster データサービスソフトウェアのインストールと構成
- クラスタハードウェアとソフトウェアのコンポーネントを追加、削除、交換、およびサービス提供するためのクラスタ管理手順
- パフォーマンスを向上させるための構成の変更方法

参照個所 (システム管理者)

次の項には、前述の重要な概念に関連する説明が記載されています。

- 36 ページの「管理インタフェース」
- 36 ページの「クラスタ内の時間」
- 37 ページの「高可用性フレームワーク」
- 40 ページの「広域デバイス」
- 41 ページの「ディスクデバイスグループ」
- 45 ページの「広域名前空間」

- 46 ページの「クラスタファイルシステム」
- 49 ページの「ディスクパスの監視」
- 54 ページの「障害による影響の防止について」
- 62 ページの「データサービス」

関連マニュアル (システム管理者)

次の Sun Cluster のマニュアルには、システム管理者の概念に関連する手順と情報が記載されています。

- 『Sun Cluster ソフトウェアのインストール (Solaris OS 版)』
- 『Sun Cluster のシステム管理 (Solaris OS 版)』
- 『Sun Cluster Error Messages Guide for Solaris OS』
- 『Sun Cluster 3.1 4/05 Release Notes for Solaris OS』
- 『Sun Cluster 3.0-3.1 Release Notes Supplement』

アプリケーション開発者

Sun Cluster システムは、Oracle、NFS、DNS、Sun™ Java System Web Server、Apache Web Server (SPARC ベースシステム上)、Sun Java System Directory Server などのアプリケーションに対応するデータサービスを提供します。データサービスを作成するには、既成のアプリケーションを Sun Cluster ソフトウェアの制御下で動作するように設定する必要があります。Sun Cluster ソフトウェアは、このようなアプリケーションの起動、停止、および監視を行う構成ファイルと管理メソッドを提供します。新しいフェイルオーバーサービスまたはスケーラブルサービスを作成する必要がある場合は、Sun Cluster Application Programming Interface (API) と Data Service Enabling Technologies API (DSET API) を使用して、そのアプリケーションがクラスタ上でデータサービスとして実行するために必要な構成ファイルと管理メソッドを開発します。

重要な概念 (アプリケーション開発者)

アプリケーション開発者は、次の概念について理解する必要があります。

- 各アプリケーションの特性。アプリケーションをフェイルオーバーまたはスケーラブルデータサービスとして実行できるかどうかを判断する必要があります。
- Sun Cluster API、DSET API、および汎用データサービス。開発者は、各自のアプリケーションをクラスタ環境に合わせて構成するプログラムまたはスクリプトを記述するために、どのツールが最も適しているかを判断する必要があります。

参照箇所 (アプリケーション開発者)

次の項には、前述の重要な概念に関連する説明が記載されています。

- 62 ページの「データサービス」
- 73 ページの「リソース、リソースグループ、リソースタイプ」

- 第 4 章

関連マニュアル (アプリケーション開発者)

次の Sun Cluster のマニュアルには、アプリケーション開発者の概念に関連する手順と情報が記載されています。

- 『Sun Cluster データサービス開発ガイド (Solaris OS 版)』
- 『Sun Cluster データサービスの計画と管理 (Solaris OS 版)』

Sun Cluster システムの作業

すべての Sun Cluster システムの作業は、いくつかの概念的な予備知識が必要です。次の表は、作業と作業手順が記載されたマニュアルを示したものです。このマニュアルの概念に関する章では、各概念がこれらの作業とどのように対応するかを説明します。

表 **1–1** Task Map: ユーザーの作業と参照するマニュアル

タスク	参照先
クラスタハードウェアの設置	『Sun Cluster 3.0-3.1 Hardware Administration Manual for Solaris OS』
クラスタへの Solaris ソフトウェアのインストール	『Sun Cluster ソフトウェアのインストール (Solaris OS 版)』
SPARC: Sun™ Management Center ソフトウェアのインストール	『Sun Cluster ソフトウェアのインストール (Solaris OS 版)』
Sun Cluster ソフトウェアのインストールと構成	『Sun Cluster ソフトウェアのインストール (Solaris OS 版)』
ボリュームマネージャーのインストールと構成	『Sun Cluster ソフトウェアのインストール (Solaris OS 版)』 各ボリュームマネージャーのマニュアル
Sun Cluster データサービスのインストールと構成	『Sun Cluster データサービスの計画と管理 (Solaris OS 版)』
クラスタハードウェアの保守	『Sun Cluster 3.0-3.1 Hardware Administration Manual for Solaris OS』
Sun Cluster ソフトウェアの管理	『Sun Cluster のシステム管理 (Solaris OS 版)』

表 1–1 Task Map: ユーザーの作業と参照するマニュアル　(続き)

タスク	参照先
ボリュームマネージャーの管理	『Sun Cluster のシステム管理 (Solaris OS 版)』 各ボリュームマネージャーのマニュアル
アプリケーションソフトウェアの管理	各アプリケーションのマニュアル
問題の識別と対処方法	『Sun Cluster Error Messages Guide for Solaris OS』
新しいデータサービスの作成	『Sun Cluster データサービス開発ガイド (Solaris OS 版)』

第 2 章

重要な概念 - ハードウェアサービスプロバイダ

この章では、Sun Cluster システム構成のハードウェアコンポーネントに関連する重要な概念について説明します。次のトピックについて述べます。

Sun Cluster システムのハードウェア/ソフトウェアコンポーネント

ここで示す情報は、主にハードウェアサービスプロバイダを対象としています。これらの概念は、サービスプロバイダが、クラスタハードウェアの設置、構成、またはサービスを提供する前に、ハードウェアコンポーネント間の関係を理解するのに役立ちます。またこれらの情報は、クラスタシステムの管理者にとっても、クラスタソフトウェアをインストール、構成、管理するための予備知識として役立ちます。

クラスタは、次のようなハードウェアコンポーネントで構成されます。

- ローカルディスク (非共有) を備えたクラスタノード
- 多重ホスト記憶装置 (ノード間で共有されるディスク)
- リムーバブルメディア (テープ、CD-ROM)

- クラスタインターコネクト
- パブリックネットワークインタフェース
- クライアントシステム
- 管理コンソール
- コンソールアクセスデバイス

Sun Cluster システムを使用すると、これらのコンポーネントを各種の構成に組み合わせることができます。これらの構成については、次の節で説明します。

- 29 ページの「SPARC: Sun Cluster トポロジ」
- 33 ページの「x86: Sun Cluster トポロジ」

2 ノードクラスタの構成例については、『Sun Cluster の概要 (Solaris OS 版)』の「Sun Cluster のハードウェア環境」を参照してください。

クラスタノード

クラスタノードとは、Solaris オペレーティングシステムと Sun Cluster ソフトウェアの両方を実行しているマシンのことです。クラスタノードは、同時に、クラスタの現在のメンバー (「クラスタメンバー」) または潜在的なメンバーのどちらかでもあります。

- SPARC: Sun Cluster ソフトウェアは、1 つのクラスタで 1 つから 16 までのノードをサポートします。サポートされるノード構成については、29 ページの「SPARC: Sun Cluster トポロジ」を参照してください。
- x86: Sun Cluster ソフトウェアは、1 つのクラスタで 2 つのノードをサポートします。サポートされるノード構成については、33 ページの「x86: Sun Cluster トポロジ」を参照してください。

一般的にクラスタノードは、1 つまたは複数の多重ホストデバイスに接続されます。多重ホストデバイスに接続されていないノードは、クラスタファイルシステムを使用して多重ホストデバイスにアクセスします。たとえば、スケーラブルサービスを 1 つ構成することで、ノードが多重ホストデバイスに直接接続されていなくてもサービスを提供することができます。

さらに、パラレルデータベース構成では、複数のノードがすべてのディスクへの同時アクセスを共有します。

- ディスクへの同時アクセスについては、24 ページの「多重ホストデバイス」を参照してください。
- パラレルデータベース構成についての詳細は、30 ページの「SPARC: クラスタペアトポロジ」と34 ページの「x86: クラスタペアトポロジ」を参照してください。

クラスタ内のノードはすべて、共通の名前 (クラスタ名) によってグループ化されます。この名前は、クラスタのアクセスと管理に使用されます。

パブリックネットワークアダプタは、ノードとパブリックネットワークを接続して、クラスタへのクライアントアクセスを可能にします。

クラスタメンバーは、1 つまたは複数の物理的に独立したネットワークを介して、クラスタ内のほかのノードと通信します。物理的に独立したネットワークの集合は、クラスタインターコネクトと呼ばれます。

クラスタ内のすべてのノードは、別のノードがいつクラスタに結合されたか、またはクラスタから切り離されたかを認識します。さらに、クラスタ内のすべてのノードは、他のクラスタノードで実行されているリソースだけでなく、ローカルに実行されているリソースも認識します。

同じクラスタ内の各ノードの処理、メモリー、および入出力機能が同等で、パフォーマンスを著しく低下させることなく処理を継続できることを確認してください。フェイルオーバーの可能性があるため、すべてのノードには、バックアップまたは二次ノードとしてすべてのノードの作業負荷を引き受けるのに十分な予備容量が必要です。

各ノードは、独自のルート (/) ファイルシステムを起動します。

クラスタハードウェアメンバー用のソフトウェアコンポーネント

ノードがクラスタメンバーとして動作するためには、ノードに次のソフトウェアがインストールされていなければなりません。

- Solaris オペレーティングシステム
- Sun Cluster ソフトウェア
- データサービスアプリケーション
- ボリューム管理 (Solaris ボリュームマネージャー™ または VERITAS Volume Manager)

  例外として、複数のディスクの冗長配列 (RAID) を使用する構成があります。この構成には、通常、Solaris ボリュームマネージャー や VERITAS Volume Manager などのボリュームマネージャーは必要ありません。
- Solaris オペレーティングシステム、Sun Cluster、およびボリュームマネージャーのインストール方法については、『Sun Cluster ソフトウェアのインストール (Solaris OS 版)』を参照してください。
- データサービスのインストールおよび構成については、『Sun Cluster データサービスの計画と管理 (Solaris OS 版)』を参照してください。
- 前述のソフトウェアコンポーネントの概念については、第 3 章を参照してください。

次の図は、Sun Cluster ソフトウェア環境を構成するソフトウェアコンポーネントとその関係の概要を示しています。

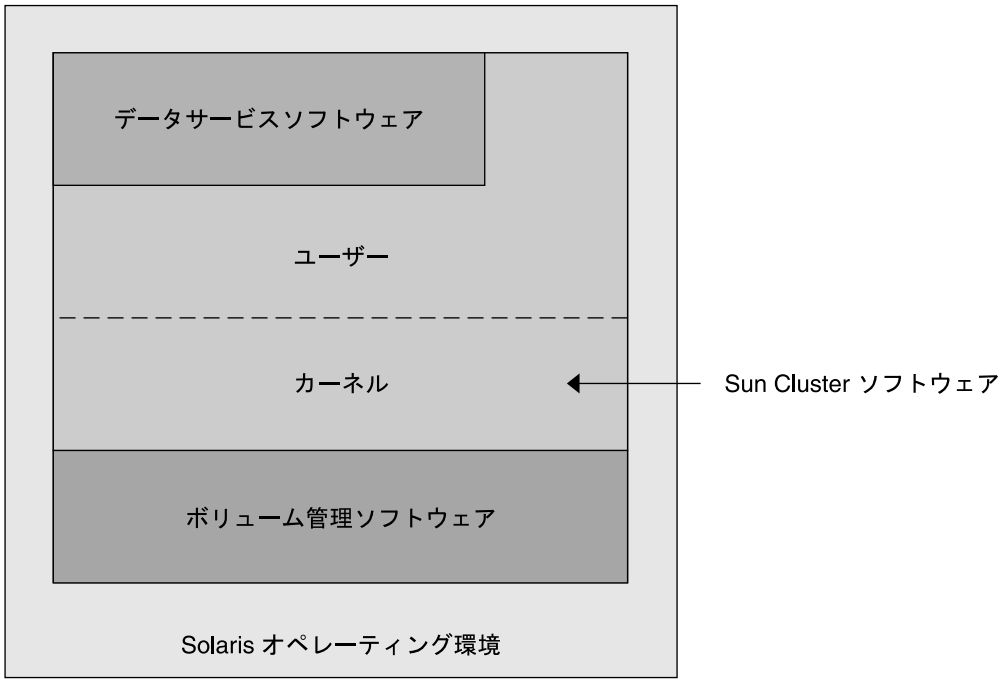


図 **2–1** Sun Cluster ソフトウェアコンポーネントとその関係の概要

クラスタメンバーの FAQ については、第 4 章を参照してください。

多重ホストデバイス

多重ホストデバイスとは、一度に複数のノードに接続できるディスクのことです。Sun Cluster 環境では、多重ホスト記憶装置によってディスクの可用性を強化できます。2 ノードクラスタでは、Sun Cluster は定足数を確立するために多重ホスト記憶装置を必要とします。3 ノードより大きなクラスタでは、定足数デバイスを必要としません。定足数についての詳細は、52 ページの「定足数と定足数デバイス」を参照してください。

多重ホストデバイスには、次の特徴があります。

- 単一ノード障害への耐性 (トレランス)。
- アプリケーションデータ、アプリケーションバイナリ、および構成ファイルを格納する機能。
- ノード障害からの保護。クライアントがあるノードを介するデータを要求して、そのノードに障害が発生した場合、これらの要求は、同じディスクに直接接続されている別のノードを使用するようにスイッチオーバーされます。
- ディスクを「マスター」する主ノードを介する広域的なアクセス、あるいは、ローカルパスを介する直接同時アクセス。現在、直接同時アクセスを使用するアプリケーションは Oracle Real Application Clusters Guard だけです。

ボリュームマネージャーは、ミラー化された構成または RAID-5 構成を提供することによって、多重ホストデバイスのデータ冗長性を実現します。現在、Sun Cluster がサポートするのは Solaris ボリュームマネージャー と VERITAS Volume Manager であり、SPARC ベースのクラスタではボリュームマネージャーとして、また、いくつかのハードウェア RAID プラットフォームでは RDAC RAID-5 ハードウェアコントローラとして使用できます。

多重ホストデバイスをミラー化したディスクやストライプ化したディスクと組み合わせると、ノードの障害や個々のディスクの障害から保護できます。

多重ホスト記憶装置の FAQ については、第 4 章を参照してください。

多重イニシエータ SCSI

この項は、多重ホストデバイスに使用されるファイバチャネル記憶装置ではなく、SCSI 記憶装置にのみ適用されます。

スタンドアロンサーバーでは、サーバーノードが、このサーバーを特定の SCSI バスに接続する SCSI ホストアダプタ回路によって、SCSI バスのアクティビティーを制御します。この SCSI ホストアダプタ回路は、*SCSI イニシエータ*と呼ばれます。この回路は、この SCSI バスに対するすべてのバスアクティビティーを開始します。Sun システムの SCSI ホストアダプタのデフォルト SCSI アドレスは 7 です。

クラスタ構成では、多重ホストデバイスを使用し、複数のサーバーノード間で記憶装置を共有します。クラスタ記憶装置が SCSI デバイスまたは Differential SCSI デバイスで構成される場合、その構成のことを「多重イニシエータ SCSI」と呼びます。この用語が示すように、複数の SCSI イニシエータが SCSI バスに存在します。

SCSI 仕様では、SCSI バス上のデバイスごとに一意の SCSI アドレスが必要 (ホストアダプタも SCSI バス上のデバイス) です。多重イニシエータ環境では、デフォルトのハードウェア構成は、すべての SCSI ホストアダプタがデフォルトの 7 になっているので、衝突が生じます。

この衝突を解決するには、各 SCSI バスで、SCSI アドレスが 7 の SCSI ホストアダプタを 1 つ残し、他のホストアダプタには、未使用の SCSI アドレスを設定します。これらの未使用の SCSI アドレスには、現在未使用のアドレスと最終的に未使用となるアドレスの両方を含めるべきです。将来未使用となるアドレスの例としては、新しいドライブを空のドライブスロットに設置することによる記憶装置の追加があります。

ほとんどの構成では、二次ホストアダプタに使用できる SCSI アドレスは 6 です。

これらのホストアダプタ用に選択された SCSI アドレスを変更するには、次のツールのいずれかを使用して、`scsi-initiator-id` プロパティーを設定します。

- `eeprom(1M)`
- SPARC ベースシステム上の OpenBoot PROM
- x86 ベースのシステムで BIOS のブート後に任意で実行する SCSI ユーティリティー

このプロパティーは 1 つのノードに対して、広域的にまたはホストアダプタごとに設定できます。SCSI ホストアダプタごとに一意の `scsi-initiator-id` を設定する手順については、『Sun Cluster 3.0-3.1 With SCSI JBOD Storage Device Manual for Solaris OS』を参照してください。

ローカルディスク

ローカルディスクとは、単一ノードにのみ接続されたディスクを表します。したがって、ローカルディスクはノードの障害から保護されません。つまり、可用性が低いということです。ただし、ローカルディスクを含むすべてのディスクは広域的な名前空間に含まれ、広域デバイスとして構成されます。したがって、ディスク自体はすべてのクラスタノードから参照できます。

ローカルディスク上のファイルシステムをほかのノードから使用できるようにするには、それらのファイルシステムを広域マウントポイントに置きます。これらの広域ファイルシステムのいずれかがマウントされているノードに障害が生じると、すべてのノードがそのファイルシステムにアクセスできなくなります。ボリュームマネージャーを使用すると、これらのディスクがミラー化されるため、これらのファイルシステムに障害が発生してもアクセス不能になることはありません。ただし、ノード障害をボリュームマネージャーで保護することはできません。

広域デバイスについての詳細は、40 ページの「広域デバイス」を参照してください。

リムーバブルメディア

クラスタでは、テープドライブや CD-ROM ドライブなどのリムーバブルメディアがサポートされています。通常、これらのデバイスは、クラスタ化していない環境と同じ方法でインストール、構成し、サービスを提供できます。これらのデバイスは、Sun Cluster で広域デバイスとして構成されるため、クラスタ内の任意のノードから各デバイスにアクセスできます。リムーバブルメディアのインストールと構成については、『Sun Cluster 3.0-3.1 Hardware Administration Manual for Solaris OS』を参照してください。

広域デバイスについての詳細は、40 ページの「広域デバイス」を参照してください。

クラスタインターコネクト

クラスタインターコネクトは、クラスタノード間のクラスタプライベート通信とデータサービス通信の転送に使用される物理的な装置構成です。インターコネクトは、クラスタプライベート通信で拡張使用されるため、パフォーマンスが制限される可能性があります。

クラスタノードだけがプライベートインターコネクトに接続できます。Sun Cluster セキュリティーモデルは、クラスタノードだけがプライベートインターコネクトに物理的にアクセスできるものと想定しています。

シングルポイント障害を回避するには、少なくとも 2 つの物理的に独立したネットワーク (つまり、パス) を使用して、すべてのノードをクラスタインターコネクトによって接続する必要があります。任意の 2 つのノード間で、複数の物理的に独立したネットワーク (2 から 6) を設定できます。

クラスタインターコネクトは、アダプタ、接続点、およびケーブルの 3 つのハードウェアコンポーネントで構成されます。次に、これらの各ハードウェアコンポーネントについて説明します。

- アダプタ – 個々のクラスタノードに存在するネットワークインタフェースカード。アダプタ名は、qfe2 のように、デバイス名に物理装置番号を加えて形成されます。物理ネットワーク接続が 1 つだけのアダプタもあれば、qfe カードをはじめ、複数の物理接続が可能なものもあります。また、ネットワークインタフェースと記憶装置インタフェースの両方を持つものもあります。

  複数のインタフェースを持つネットワークアダプタは、アダプタ全体に障害が生じると、単一地点による障害の原因となる可能性があります。可用性を最適にするには、2 つのノード間の唯一のパスが単一のネットワークアダプタに依存しないように、クラスタを設定してください。
- 接続点 – クラスタノードの外部に常駐するスイッチ。ジャンクションは、パススルーおよび切り換え機能を実行して、3 つ以上のノードに接続できるようにします。2 ノードクラスタでは、各ノードの冗長アダプタに接続された冗長物理ケーブルによって、ノードを相互に直接接続できるため、接続点は必要ありません。3 ノード以上の構成では、通常は接続点が必要です。
- ケーブル – 2 つのネットワークアダプタ間、あるいは、アダプタと接続点の間に設置する物理接続。

クラスタインターコネクトの FAQ については、第 4 章を参照してください。

パブリックネットワークインタフェース

クライアントは、パブリックネットワークインタフェースを介してクラスタに接続します。各ネットワークアダプタカードは、カードに複数のハードウェアインタフェースがあるかどうかによって、1 つまたは複数のパブリックネットワークに接続できます。複数のパブリックネットワークインタフェースカードをもつノードを設定することによって、複数のカードをアクティブにし、それぞれを相互のフェイルオーバーバックアップとすることできます。いずれかのアダプタに障害が発生すると、IP ネットワークマルチパス ソフトウェアが呼び出され、障害のあるインタフェースが同じグループの別のアダプタにフェイルオーバーされます。

パブリックネットワークインタフェースのクラスタ化に関連する特殊なハードウェアについての特記事項はありません。

パブリックネットワークの FAQ については、第 4 章を参照してください。

クライアントシステム

クライアントシステムには、パブリックネットワークによってクラスタにアクセスするワークステーションや他のサーバーが含まれます。クライアント側プログラムは、クラスタ上で動作しているサーバー側アプリケーションが提供するデータやサービスを使用します。

クライアントシステムの可用性は高くありません。クラスタ上のデータとアプリケーションは、高い可用性を備えています。

クライアントシステムの FAQ については、第 4 章を参照してください。

コンソールアクセスデバイス

すべてのクラスタノードにはコンソールアクセスが必要です。コンソールアクセスを取得するには、次のうちの 1 つのデバイスを使用します。

- クラスタハードウェアとともに購入した端末集配信装置
- Sun Enterprise E10000 サーバーのシステムサービスプロセッサ (SSP) (SPARC ベースクラスタの場合)
- Sun Fire™ サーバーのシステムコントローラ (同じく SPARC ベースクラスタの場合)
- 各ノードの `ttya` にアクセスできる別のデバイス

サポートされている唯一の端末集配信装置は、Sun から提供されています。サポートされている Sun の端末集配信装置の使用は任意です。端末集配信装置を使用すると、TCP/IP ネットワークを使用して、各ノードの `/dev/console` にアクセスできます。この結果、ネットワークの任意の場所にあるリモートワークステーションから、各ノードにコンソールレベルでアクセスできます。

システムサービスプロセッサ (SSP) は、Sun Enterprise E10000 サーバーへのコンソールアクセスを提供します。SSP とは、Sun Enterprise E10000 サーバーをサポートするように構成された Ethernet ネットワーク上のマシンのことです。SSP は、Sun Enterprise E10000 サーバーの管理コンソールです。Sun Enterprise E10000 サーバーのネットワークコンソール機能を使用すると、ネットワーク上のすべてのワークステーションからホストコンソールセッションを開くことができます。

これ以外のコンソールアクセス方式には、他の端末集配信装置、別ノードおよびダム端末からの `tip(1)` シリアルポートアクセスが含まれます。Sun™ キーボードとモニター、または他のシリアルポートデバイスも使用できます。

管理コンソール

アクティブなクラスタを管理するには、「管理コンソール」という専用の UltraSPARC® ワークステーションまたは Sun Fire V65x サーバーを使用します。通常は、Cluster Control Panel (CCP) や Sun Management Center 製品の Sun Cluster モ

ジュール (SPARC ベースクラスタのみ) などの管理ツールソフトウェアを管理コンソールにインストールして実行します。CCP で `cconsole` を使用すると、一度に複数のノードコンソールに接続できます。CCP の使用法についての詳細は、『Sun Cluster のシステム管理 (Solaris OS 版)』の第 1 章「Sun Cluster の管理の概要」を参照してください。

管理コンソールはクラスタノードではありません。管理コンソールは、パブリックネットワークを介して、または任意でネットワークベースの端末集配信装置を経由して、クラスタノードへのリモートアクセスに使用します。クラスタが Sun Enterprise E10000 プラットフォームで構成される場合、管理コンソールから SSP にログインして、`netcon(1M)` コマンドで接続する必要があります。

通常、ノードはモニターなしで構成します。そして、管理コンソールから `telnet` セッションを使用して、ノードのコンソールにアクセスします。管理コンソールは端末集配信装置に接続され、端末集配信装置から当該ノードのシリアルポートに接続されます。Sun Enterprise E1000 サーバーの場合は、システムサービスプロセッサから接続します。詳細は、28 ページの「コンソールアクセスデバイス」を参照してください。

Sun Cluster では専用の管理コンソールは必要ありませんが、専用の管理コンソールを使用すると、次のような利点があります。

- コンソールと管理ツールを同じマシンにまとめることで、クラスタ管理を一元化できます。
- ハードウェアサービスプロバイダによる問題解決が迅速に行われます。

管理コンソールの FAQ については、第 4 章を参照してください。

SPARC: Sun Cluster トポロジ

トポロジとは、Sun Cluster 環境で使用されている記憶装置プラットフォームにクラスタノードを接続するための接続スキーマをいいます。Sun Cluster ソフトウェアは、次のガイドラインに従うトポロジをサポートします。

- SPARC ベースのシステムで構成される Sun Cluster 環境は、実装する記憶装置の構成に関係なく、1 つのクラスタで最大 16 のノードをサポートします。
- 共有ストレージデバイスは、そのストレージデバイスでサポートされている数のノードに接続できます。
- 共有ストレージデバイスはクラスタのすべてのノードに接続する必要はありませんが、2 つ以上のノードに接続する必要があります。

Sun Cluster ソフトウェアでは、特定のトポロジを使用するようにクラスタを構成する必要はありません。次のトポロジには、クラスタの接続スキーマを説明するときに使用する用語を示します。これらのトポロジは典型的な接続スキーマです。

- クラスタペア
- ペア +N
- N+1 (星型)
- N*N (スケーラブル)

次の各項では、それぞれのトポロジを図で示しています。

SPARC: クラスタペアトポロジ

クラスタペアトポロジとは、単一のクラスタ管理フレームワークのもとで動作する複数のノードペアをいいます。この構成では、ペアの間でのみフェイルオーバーが発生します。ただし、すべてのノードがクラスタインターコネクトによって接続されていて、Sun Cluster ソフトウェア制御のもとで動作します。このトポロジを使用する場合、1 つのペアでパラレルデータベースアプリケーションを実行し、別のペアでフェイルオーバーまたはスケーラブルなアプリケーションを実行できます。

クラスタファイルシステムを使用すると、2 ペア構成も可能になります。アプリケーションデータが格納されているディスクにすべてのノードが直接接続されていない場合でも、複数のノードがスケーラブルサービスまたはパラレルデータベースを実行できます。

次の図は、クラスタペア構成を示したものです。

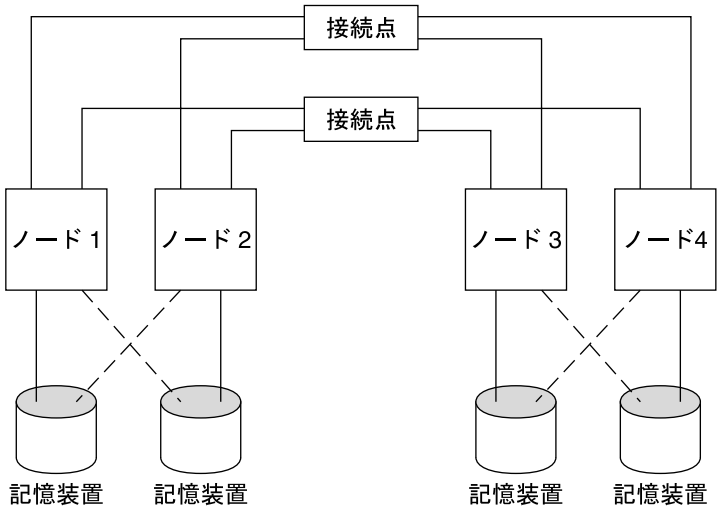


図 **2–2** SPARC: クラスタペアトポロジ

SPARC: ペア +N トポロジ

ペア +N トポロジには、共有記憶装置に直接接続されたノードのペアと、クラスタインターコネクトを使用して共有記憶装置にアクセスするノードの追加セットが含まれます。これらのノードは直接それらの共有記憶装置には接続されていません。

次の図は、4 つのノードのうち 2 つ (ノード 3 とノード 4) がクラスタインターコネクトを使用して記憶装置にアクセスする、1 つのペア +N トポロジを示したものです。この構成を拡張し、共有記憶装置には直接アクセスしない追加ノードを追加することができます。

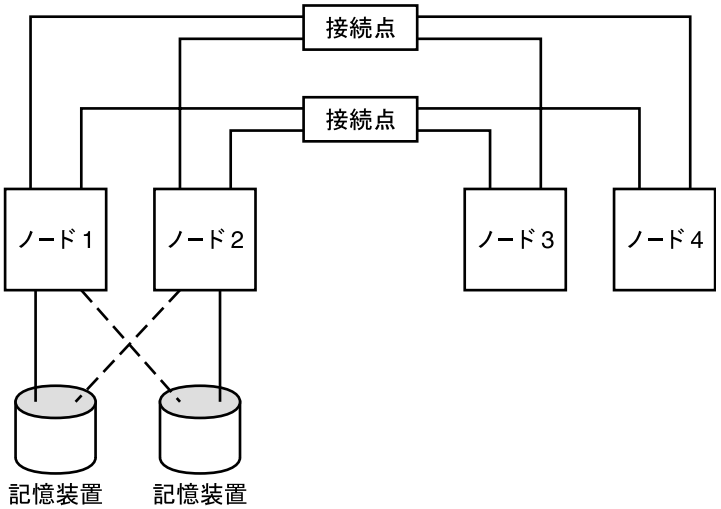


図 **2–3** ペア +N トポロジ

SPARC: N+1 (星形) トポロジ

N+1 トポロジには、いくつかの主ノードと 1 つの二次ノードが含まれます。主ノードと二次ノードを同等に構成する必要はありません。主ノードは、アプリケーションサービスをアクティブに提供します。二次ノードは、主ノードに障害が生じるのを待機する間、アイドル状態である必要はありません。

二次ノードは、この構成ですべての多重ホスト記憶装置に物理的に接続されている唯一のノードです。

主ノードで障害が発生すると、Sun Cluster はそのリソースの処理を二次ノードで続行し、リソースは自動または手動で主ノードに切り換えられるまで二次ノードで機能します。

二次ノードには、主ノードの 1 つに障害が発生した場合に負荷を処理できるだけの十分な予備の CPU 容量が常に必要です。

次の図は、N+1 構成を示したものです。

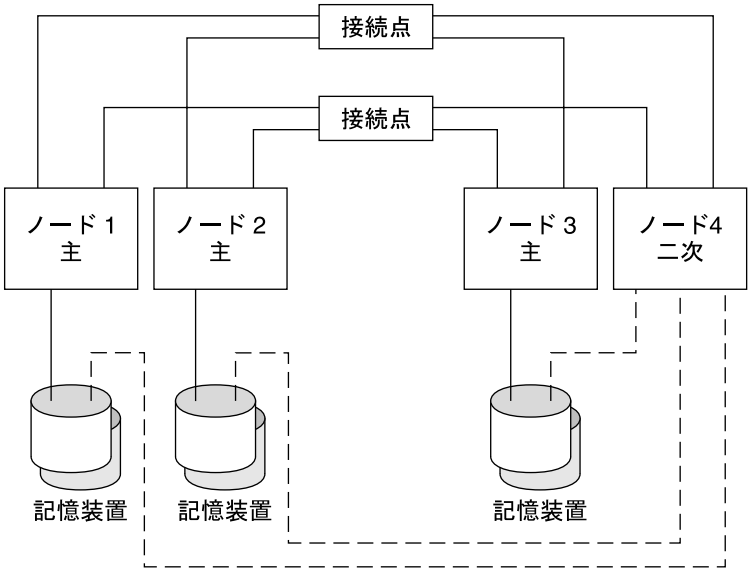


図 **2–4** SPARC: N+1 トポロジ

SPARC: N*N (スケーラブル) トポロジ

N*N トポロジを使用すると、クラスタ内のすべての共有ストレージデバイスをクラスタ内のすべてのノードに接続できます。このトポロジを使用すると、高可用性アプリケーションはサービスを低下させずに、あるノードから別のノードにフェイルオーバーできます。フェイルオーバーが発生すると、新しいノードはプライベートインターコネクトではなく、ローカルパスを使用して、ストレージデバイスにアクセスできます。

次の図に、N*N 構成を示します。

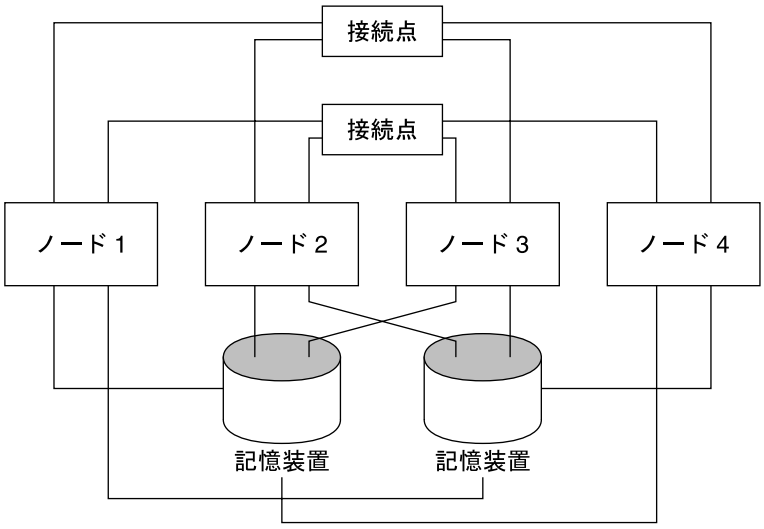


図 **2–5** SPARC: N*N トポロジ

x86: Sun Cluster トポロジ

トポロジとは、クラスタノードと、クラスタで使用される記憶装置プラットフォームを接続する接続スキーマをいいます。Sun Cluster は、次のガイドラインに従うトポロジをサポートします。

- x86 ベースのシステムで構成された Sun Cluster は、1 つのクラスタで 2 つのノードをサポートします。
- 共有記憶装置を両方のノードに接続する必要があります。

Sun Cluster では、特定のトポロジを使用するようにクラスタを構成する必要はありません。次のクラスタペアトポロジは、x86 ベースのノードからなるクラスタで可能な唯一のトポロジです。このトポロジを示すことによって、クラスタの接続スキーマを表す用語を紹介します。このトポロジは代表的な接続スキーマです。

次の項では、トポロジを図で示しています。

x86: クラスタペアトポロジ

クラスタペアトポロジとは、単一のクラスタ管理フレームワークのもとで動作する 2 つのノードをいいます。この構成では、ペアの間でのみフェイルオーバーが発生します。ただし、すべてのノードがクラスタインターコネクトによって接続されていて、Sun Cluster ソフトウェア制御のもとで動作します。このトポロジを使用する場合、ペアでパラレルデータベース、フェイルオーバー、またはスケーラブルアプリケーションを実行できます。

次の図は、クラスタペア構成を示したものです。

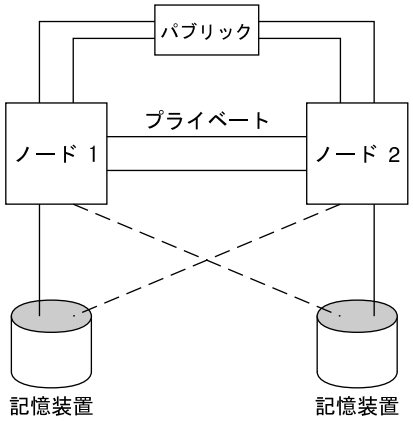


図 **2–6** x86: クラスタペアトポロジ

第 3 章

重要な概念 - システム管理者とアプリケーション開発者

この章では、Sun Cluster システムのソフトウェアコンポーネントに関連する重要な概念について説明します。次のトピックについて述べます。

この情報は、主に、Sun Cluster API および SDK を使用するシステム管理者とアプリケーション開発者を対象としています。クラスタシステムの管理者にとっては、この情報は、クラスタソフトウェアのインストール、構成、管理についての予備知識となります。アプリケーション開発者は、この情報を使用して、作業を行うクラスタ環境を理解できます。

管理インタフェース

Sun Cluster システムをインストール、構成、および管理する方法は、いくつかのユーザーインタフェースの中から選択することができます。システム管理作業は、SunPlex Manager グラフィックユーザーインタフェース (GUI) かコマンド行インタフェースから行います。コマンド行インタフェースでは、特定のインストール作業や構成作業を容易にする `scinstall` や `scsetup` などのユーティリティーが使用できます。Sun Cluster システムには、Sun Management Center の一部として実行される、特定のクラスタ作業に GUI を提供するモジュールもあります。このモジュールを使用できるのは、SPARC ベースのクラスタに限られます。管理インタフェースについての詳細は、『Sun Cluster のシステム管理 (Solaris OS 版)』の「管理ツール」を参照してください。

クラスタ内の時間

クラスタ内のすべてのノード間の時刻は同期をとる必要があります。クラスタノードの時刻と外部の時刻ソースの同期をとるかどうかは、クラスタの操作にとって重要ではありません。Sun Cluster システムは、Network Time Protocol (NTP) を使用し、ノード間のクロックの同期をとっています。

通常、システムクロックが数分の 1 秒程度変更されても問題は起こりません。しかし、システムクロックと時刻の起点の同期をとるために、`date(1)`、`rdate(1M)`、`xntpdate(1M)` を (対話形式または `cron` スクリプト内で) アクティブクラスタに対して実行すると、これよりも大幅な時刻変更を強制的に行うことが可能です。ただしこの強制的な変更を行った場合、ファイル修正時刻の表示に問題が生じたり、NTP サービスに混乱が生じる可能性があります。

Solaris オペレーティングシステムを各クラスタノードにインストールする場合は、ノードのデフォルトの時刻と日付の設定を変更できます。通常は、工場出荷時のデフォルト値を使用します。

`scinstall(1M)` を使用して Sun Cluster ソフトウェアをインストールする場合は、インストールプロセスの手順の 1 つとして、クラスタの NTP を構成します。Sun Cluster ソフトウェアは、`ntp.cluster` というテンプレートファイルを提供しています (インストールされたクラスタノードの `/etc/inet/ntp.cluster` を参照)。このテンプレートは、すべてのクラスタノード間で対等関係を確立します。1 つのノードは「優先ノード」になります。ノードはプライベートホスト名で識別され、時刻の同期化がクラスタイン ターコネクト全体で行われます。NTP 用のクラスタの構成方法については、『Sun Cluster ソフトウェアのインストール (Solaris OS 版)』の第 2 章「Sun Cluster ソフトウェアのインストールと構成」を参照してください。

また、クラスタの外部に 1 つまたは複数の NTP サーバーを設定し、`ntp.conf` ファイルを変更してその構成を反映させることもできます。

通常の操作では、クラスタの時刻を調整する必要はありません。ただし、Solaris オペレーティングシステムをインストールしたときに設定された誤った時刻を変更する場合の手順については、『Sun Cluster のシステム管理 (Solaris OS 版)』の第 7 章「クラスタの管理」を参照してください。

高可用性フレームワーク

Sun Cluster システムでは、ユーザーとデータ間の「パス」にあるすべてのコンポーネント、つまり、ネットワークインタフェース、アプリケーション自体、ファイルシステム、および多重ホストデバイスを高可用性にします。一般に、システムで単一 (ソフトウェアまたはハードウェア) の障害が発生してもあるクラスタコンポーネントが稼働し続けられる場合、そのコンポーネントは高可用性であると考えられます。

次の表に、Sun Cluster コンポーネントの障害の種類 (ハードウェアとソフトウェアの両方) と、高可用性フレームワークに組み込まれた回復の種類を示します。

表 **3–1** Sun Cluster システムの障害の検出と回復のレベル

障害が発生したクラスタリソース	ソフトウェアの回復	ハードウェアの回復
データサービス	HA API、HA フレームワーク	なし
パブリックネットワークアダプタ	IP ネットワークマルチパス	複数のパブリックネットワークアダプタカード
クラスタファイルシステム	一次複製と二次複製	多重ホストデバイス
ミラー化された多重ホストデバイス	ボリューム管理 (Solaris ボリュームマネージャー と VERITAS Volume Manager、SPARC ベースのクラスタでのみ使用可能)	ハードウェア RAID-5 (Sun StorEdge™ A3x00 など)
広域デバイス	一次複製と二次複製	デバイス、クラスタトランスポート接続点への多重パス
プライベートネットワーク	HA トランスポートソフトウェア	ハードウェアから独立した多重プライベートネットワーク
ノード	CMM、フェイルファーストドライバ	複数ノード

Sun Cluster ソフトウェアの高可用性フレームワークは、ノードの障害をすばやく検出して、クラスタ内の残りのノードにあるフレームワークリソース用に新しい同等のサーバーを作成します。どの時点でもすべてのフレームワークリソースが使用できなくなることはありません。障害が発生したノードの影響を受けないフレームワークリソースは、回復中も完全に使用できます。さらに、障害が発生したノードのフレームワークリソースは、回復されると同時に使用可能になります。回復されたフレームワークリソースは、他のすべてのフレームワークリソースが回復するまで待機する必要はありません。

最も可用性の高いフレームワークリソースは、そのリソースを使用するアプリケーション (データサービス) に対して透過的に回復されます。フレームワークリソースのアクセス方式は、ノードの障害時にも完全に維持されます。アプリケーションは単に、フレームワークリソースサーバーが別のノードに移動したことを認識できないだけです。1 つのノードで障害が発生しても、残りのノード上にあるプログラムがそのノードに接続されているファイル、デバイス、およびディスクボリュームを使用できるので、その障害は完全に透過的と言えます。別のノードからそのディスクに代替ハードウェアパスが設定されている場合に、このような透過性が実現されます。この例としては、複数ノードへのポートを持つ多重ホストデバイスの使用があります。

クラスタメンバーシップモニター

データが破壊から保護されるように保証するには、すべてのノードが、クラスタメンバーシップに対して一定の同意に達していなければなりません。必要であれば、CMM は、障害に応じてクラスタサービス (アプリケーション) のクラスタ再構成を調整します。

CMM は、クラスタのトランスポート層から、他のノードへの接続に関する情報を受け取ります。CMM は、クラスタインターコネクトを使用して、再構成中に状態情報を交換します。

CMM は、クラスタメンバーシップの変更を検出すると、それに合わせてクラスタを構成します。このような同期構成では、クラスタの新しいメンバーシップに基づいて、クラスタリソースが再配布されることがあります。

Sun Cluster ソフトウェアの以前のリリースとは異なり、CMM は完全にカーネルで実行されます。

クラスタ自身が複数の異なるクラスタに分割されないようにする方法についての詳細は、54 ページの「障害による影響の防止について」を参照してください。

フェイルファースト機構

あるノードで重大な問題を検出すると、CMM はクラスタフレームワークに依頼して、そのノードを強制的に停止 (パニック) し、クラスタメンバーシップから取り除きます。この機構を「フェイルファースト」といいます。フェイルファーストがノードを強制的に停止する方法は 2 つあります。

- クラスタから切り離されたノードが定足数を満たさずに再び新しいクラスタを起動しようとすると、ノードは共有ディスクへのアクセスを「防止」されます。この種類のフェイルファーストについての詳細は、54 ページの「障害による影響の防止について」を参照してください。
- クラスタ固有のデーモン (`clexecd`、`rpc.pmfd`、`rgmd` 、または `rpc.ed`) が 1 つまたは複数異常終了すると、CMM はその障害を検出し、そのノードはパニックします。

クラスタデーモンが異常終了すると、ノードはパニックし、そのノードのコンソールには次のようなメッセージが表示されます。

```
panic[cpu0]/thread=40e60: Failfast: Aborting because "pmfd" died 35 seconds ago.
409b8 cl_runtime:__0FZsc_syslog_msg_log_no_argsPviTCPCcTB+48 (70f900, 30, 70df54, 407acc, 0)
%l0-7: 1006c80 000000a 000000a 10093bc 406d3c80 7110340 0000000 4001 fbf0
```

パニック後、このノードは再起動して、クラスタに再び参加しようとします。あるいは、SPARC ベースのシステムで構成されているクラスタの場合、そのノードは OpenBoot™ PROM (OBP) プロンプトのままになることがあります。ノードがどちらのアクションをとるかは、`auto-boot?` パラメータの設定によって決定されます。`auto-boot?` を設定するには、OpenBoot PROM の `ok` プロンプトで `eeprom(1M)` を使用します。

クラスタ構成レポジトリ (CCR)

CCR は、更新に 2 フェーズのコミットアルゴリズムを使用します。更新はすべてのクラスタメンバーで正常に終了する必要があり、そうしないと、その更新はロールバックされます。CCR はクラスタインターコネクトを使用して、分散更新を適用します。

注意 – CCR はテキストファイルで構成されていますが、CCR ファイルを手作業で絶対に編集しないでください。各ファイルには、ノード間の一貫性を保証するための検査合計レコードが含まれています。CCR ファイルを手作業で更新すると、ノードまたはクラスタ全体の機能が停止する可能性があります。

CCR は、CMM に依存して、定足数 (quorum) が確立された場合にのみクラスタが実行されるように保証します。CCR は、クラスタ全体のデータの一貫性を確認し、必要に応じて回復を実行し、データへの更新を容易にします。

広域デバイス

Sun Cluster システムは、広域デバイスを使用して、デバイスが物理的に接続されている場所に関係なく、任意のノードからクラスタ内のすべてのデバイスに対して、クラスタ全体の可用性の高いアクセスを可能にします。通常、広域デバイスへのアクセス提供しているノードに障害が発生すると、Sun Cluster ソフトウェアはそのデバイスへの別のパスを自動的に検出して、そのパスにアクセスを切り替えます。Sun Cluster 広域デバイスには、ディスク、CD-ROM、テープが含まれます。しかし、Sun Cluster ソフトウェアがサポートする多重ポート広域デバイスはディスクだけです。つまり、CD-ROM とテープは現在、高可用性のデバイスではありません。各サーバーのローカルディスクも多重ポート化されていないため、可用性の高いデバイスではありません。

クラスタは自動的に、クラスタ内の各ディスク、CD-ROM、およびテープデバイスに一意の ID を割り当てます。この割り当てによって、クラスタ内の任意のノードから各デバイスに対して一貫したアクセスが可能になります。広域デバイス名前空間は、`/dev/global` ディレクトリにあります。詳細は、45 ページの「広域名前空間」を参照してください。

多重ポート広域デバイスは、1 つのデバイスに対して複数のパスを提供します。多重ホストディスクは複数のノードがホストするディスクデバイスグループの一部であるため、多重ホストディスクの可用性は高くなります。

デバイス ID と DID 疑似ドライバ

Sun Cluster ソフトウェアは、DID 疑似ドライバと呼ばれる構造によって広域デバイスを管理します。このドライバを使用して、多重ホストディスク、テープドライブ、CD-ROM を含め、クラスタ内のあらゆるデバイスに一意の ID を自動的に割り当てます。

DID 疑似ドライバは、クラスタの広域デバイスアクセス機能における重要な部分です。DID ドライバは、クラスタのすべてのノードを探索して、一意のディスクデ バイスのリストを作成し、クラスタのすべてのノードで一貫している一意のメジャー番号およびマイナー番号を各デバイスに割り当てます。広域デバイスへのアクセスは、ディスクを示す `c0t0d0` などの従来の Solaris デバイス ID ではなく、(DID ドライバが割り当てた) この一意のデバイス ID を利用して行われます。

この方法により、ディスクにアクセスするすべてのアプリケーション (ボリュームマネージャーまたは raw デバイスを使用するアプリケーションなど) は、一貫したパスを使用してクラスタ全体にアクセスできます。各デバイスのローカルメジャー番号およびマイナー番号はノードによって異なり、Solaris デバイス命名規則も変更する可能性があるため、この一貫性は、多重ホストディスクにとって特に重要です。たとえ

ば、Node1 は多重ホストディスクを c1t2d0 と識別し、Node2 は同じディスクをまったく異なるディスクとして、つまり、c3t2d0 と識別する場合があります。ノードはこのような名前の代わりに、DID ドライバが割り当てた広域名 (d10 など) を使用します。つまり、DID ドライバは多重ホストディスクへの一貫したマッピングを各ノードに提供します。

デバイス ID を更新および管理するには、scdidadm(1M) および scgdevs(1M) を使用します。詳しくは、以下のマニュアルページを参照してください。

- scdidadm(1M)
- scgdevs(1M)

ディスクデバイスグループ

Sun Cluster システムでは、すべての多重ホストデバイスは、Sun Cluster ソフトウェアの制御下にある必要があります。最初に、ボリュームマネージャーのディスクグループ (Solaris ボリュームマネージャー のディスクセットまたは VERITAS Volume Manager のディスクグループ。後者は SPARC ベースのクラスタでのみ使用可能) を多重ホストディスク上で作成します。次に、ボリュームマネージャーのディスクグループをディスクデバイスグループとして登録します。ディスクデバイスグループは、広域デバイスの一種です。さらに、Sun Cluster ソフトウェアは、個々のディスクデバイスやテープデバイスごとに raw ディスクデバイスグループを自動的に作成します。ただし、これらのクラスタデバイスグループは、広域デバイスとしてアクセスされるまではオフラインの状態になっています。

この登録によって、Sun Cluster システムは、どのノードがどのボリュームマネージャーディスクグループへのパスをもっているかを知ることができます。この時点でそのボリュームマネージャーデバイスグループは、クラスタ内で広域アクセスが可能になります。あるディスクデバイスグループが複数のノードから書き込み可能 (マスター) な場合は、そのディスクデバイスグループに格納されるデータは、高度な可用性を有することになります。高度な可用性を備えたディスクデバイスグループには、クラスタファイルシステムを格納できます。

注 – ディスクデバイスグループは、リソースグループとは別のものです。あるノードが 1 つのリソースグループ (データサービスプロセスのグループを表す) をマスターする一方で、別のノードが、データサービスによってアクセスされるディスクグループをマスターできます。ただし、最も良い方法は、特定のアプリケーションのデータを保存するディスクデバイスグループと、アプリケーションのリソース (アプリケーションデーモン) を同じノードに含むリソースグループを維持することです。リソースグループとディスクデバイスグループの関係についての詳細は、『Sun Cluster データサービスの計画と管理 (Solaris OS 版)』の「リソースグループとディスクデバイスグループの関係」を参照してください。

あるノードがディスクデバイスグループを使用するとき、ボリュームマネージャーのディスクグループは実際に使用するディスクに対してマルチパスサポートを提供するため、そのディスクグループは「広域」になります多重ホストディスクに物理的に接続された各クラスタノードは、ディスクデバイスグループへのパスを提供します。

ディスクデバイスグループのフェイルオーバー

ディスク格納装置は複数のノードに接続されるため、現在デバイスグループをマスターしているノードに障害が生じた場合でも、代替パスによってその格納装置にあるすべてのディスクデバイスグループにアクセスできます。デバイスグループをマスターするノードの障害は、回復と一貫性の検査を実行するために要する時間を除けば、デバイスグループへのアクセスに影響しません。この時間の間は、デバイスグループが使用可能になるまで、すべての要求は (アプリケーションには透過的に) 阻止されます。

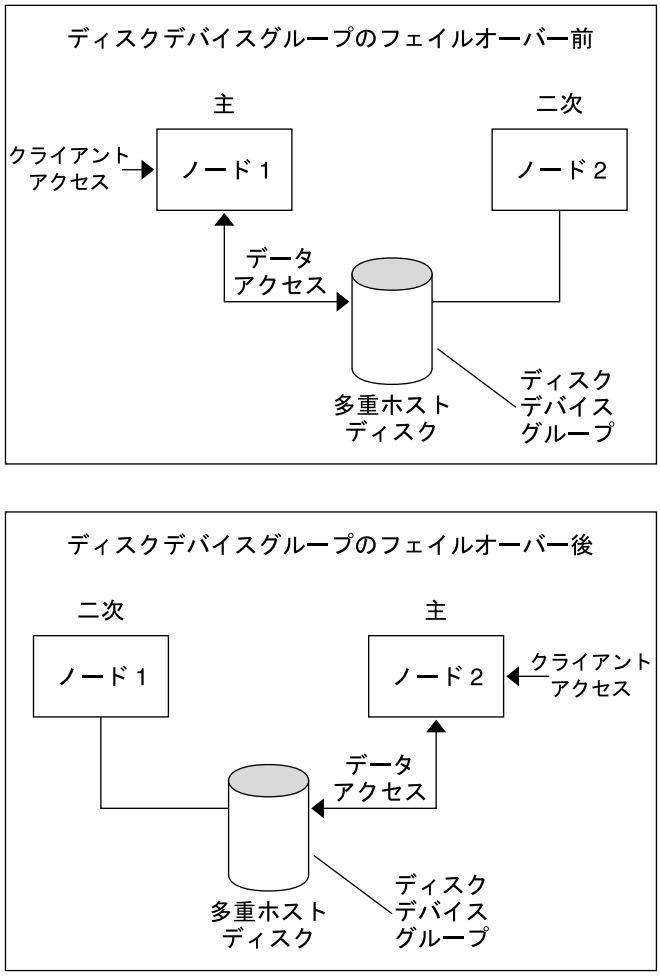


図 **3–1** フェイルオーバー前後のディスクデバイスグループ

多重ポートディスクデバイスグループ

この節では、多重ポートディスク構成において性能と可用性をバランスよく実現するディスクデバイスグループのプロパティーについて説明します。Sun Cluster ソフトウェアには、多重ポートディスク構成を設定するためのプロパティーが 2 つあります。つまり、`preferenced` と `numsecondaries` です。`preferenced` プロパティーは、フェイルオーバーの発生時に各ノードがどの順で制御を取得するかを制御します。`numsecondaries` プロパティーは、特定のデバイスグループに対する二次ノードの数を設定します。

高可用性サービスは、主ノードが停止し、かつ、主ノードになる資格のある二次ノードが存在しないときに、完全に停止したと見なされます。preferenced プロパティーが true に設定されている場合、 サービスのフェイルオーバーが発生すると、ノードリストの順序に従って、二次ノードが選択されます。設定されるノードリストは、主制御権の獲得を試みる順序、つまり、スペアノードから二次ノードに移行する順序を決定します。scsetup(1M) ユーティリティーを使用すると、デバイスサービスの設定を動的に変更できます。従属サービスプロバイダ (広域ファイルシステムなど) に関連する設定は、デバイスサービスの設定と同じになります。

主ノードは、正常な運用時に二次ノードのチェックポイントをとります。多重ポートディスク構成では、二次ノードのチェックポイントをとるたびに、クラスタの性能の低下やメモリーのオーハーヘッドの増加が発生します。スペアノードのサポートが実装されているのは、このようなチェックポイントによる性能の低下やメモリーのオーバーヘッドを最小限に抑えるためです。デフォルトでは、ディスクデバイスグループには 1 つの主ノードと 1 つの二次ノードがあります。残りのプロバイダノードはスペアノードです。フェイルオーバーが発生すると、二次ノードが主ノードになり、ノードリスト上で最も優先順位の高い (スペア) ノードが二次ノードになります。

二次ノードの望ましい数には、任意の整数 (1 から、デバイスグループ内の動作可能な主ノード以外のプロバイダノードの数まで) を設定できます。

注 – Solaris ボリュームマネージャー を使用している場合、 numsecondaries プロパティーにデフォルト以外の数字を設定するには、まず、ディスクデバイスグループを作成する必要があります。

デバイスサービスのためのデフォルトの望ましい二次ノード数は 1 です。望ましい数とは、複製フレームワークによって維持される二次プロバイダノードの実際の数です。ただし、動作可能な主ノード以外のプロバイダノードの数が望ましい数よりも小さい場合を除きます。ノードを構成に追加したり、ノードを構成から切り離す場合は、numsecondaries プロパティーを変更したあと、ノードリストを十分に確認する必要があります。ノードリストと二次ノードの望ましい数を正しく保つことによって、構成されている二次ノードの数と、フレームワークによって与えられている実際の数の不一致を防げます。

- (Solaris ボリュームマネージャー) 構成へのノードの追加または切り離しを管理するには、metaset(1M) コマンドを Solaris ボリュームマネージャー デバイスグループに使用して、さらに、preferenced プロパティーと numsecondaries プロパティーの設定も組み合わせて使用します。
- (Veritas Volume Manager) 構成へのノードの追加または切り離しを管理するには、scconf(1M) コマンドを VxVM ディスクデバイスグループに使用して、さらに、preferenced プロパティーと numsecondaries プロパティーの設定も組み合わせて使用します。
- ディスクグループのプロパティーを変更する手順については、『Sun Cluster のシステム管理 (Solaris OS 版)』の「クラスタファイルシステムの管理の概要」を参照してください。

広域名前空間

広域デバイスを有効にする Sun Cluster ソフトウェアの機構は、広域名前空間です。広域名前空間には、ボリューム管理ソフトウェアの名前空間とともに、/dev/global/ 階層が含まれます。広域名前空間は、多重ホストディスクとローカルディスクの両方 (および CD-ROM やテープなどの他のクラスタデバイスすべて) を反映して、多重ホストディスクへの複数のフェイルオーバーパスを提供します。多重ホストディスクに物理的に接続された各ノードは、クラスタ内のすべてのノードの記憶装置に対するパスを提供します。

Solaris Volume Manager の場合、ボリュームマネージャーの名前空間は、通常、/dev/md/*diskset*/dsk (と rdsk) ディレクトリにあります。Veritas VxVM の場合、 ボリュームマネージャーの名前空間は /dev/vx/dsk/*disk-group* ディレクトリと /dev/vx/rdsk/*disk-group* ディレクトリにあります。これらの名前空間は、クラスタ全体でインポートされている Solaris ボリュームマネージャー の各ディスクセットと VxVM の各ディスクグループのディレクトリから構成されます。これらの各ディレクトリには、そのディスクセットまたはディスクグループ内の各メタデバイスまたはボリュームのデバイスノードが格納されています。

Sun Cluster システムでは、ボリュームマネージャーのローカルの名前空間にある各デバイスノードは、/global/.devices/node@*nodeID* (*nodeID* はクラスタ内のノードを表す整数) というファイルシステムにあるデバイスノードへのシンボリックリンクとして表されます。Sun Cluster ソフトウェアは、その標準的な場所に引き続きシンボリックリンクとしてボリューム管理デバイスも表示します。広域名前空間と標準ボリュームマネージャー名前空間は、どちらも任意のクラスタノードから使用できます。

広域名前空間には、次の利点があります。

- 各ノードの独立性が高く、デバイス管理モデルを変更する必要がほとんどありません。
- デバイスを選択的に広域に設定できます。
- Sun の製品以外のリンクジェネレータが引き続き動作します。
- ローカルデバイス名を指定すると、その広域名を取得するために簡単なマッピングが提供されます。

ローカル名前空間と広域名前空間の例

次の表は、多重ホストディスク c0t0d0s0 でのローカル名前空間と広域名前空間のマッピングを示したものです。

表 **3–2** ローカル名前空間と広域名前空間のマッピング

コンポーネントまたはパス	ローカルノード名前空間	広域名前空間
Solaris 論理名	`/dev/dsk/c0t0d0s0`	`/global/.devices/node@`*nodeID* `/dev/dsk/c0t0d0s0`
DID 名	`/dev/did/dsk/d0s0`	`/global/.devices/node@`*nodeID* `/dev/did/dsk/d0s0`
Solaris ボリュームマネージャー	`/dev/md/`*diskset*`/dsk/d0`	`/global/.devices/node@`*nodeID* `/dev/md/`*diskset*`/dsk/ d0`
SPARC:VERITAS Volume Manager	`/dev/vx/dsk/`*disk-group*`/v0`	`/global/.devices/node@`*nodeID* `/dev/vx/dsk/`*disk-group* `/v0`

広域名前空間はインストール時に自動的に生成されて、再構成再起動のたびに更新されます。広域名前空間は、`scgdevs(1M)` コマンドを実行して生成することもできます。

クラスタファイルシステム

クラスタファイルシステムには、次の機能があります。

- ファイルのアクセス場所が透過的になります。システムのどこにあるファイルでも、プロセスから開くことができます。すべてのノードのプロセスから同じパス名を使ってファイルにアクセスできます。

  注 – クラスタファイルシステムは、ファイルを読み取る際に、ファイル上のアクセス時刻を更新しません。

- 一貫したプロトコルを使用して、ファイルが複数のノードから同時にアクセスされている場合でも、UNIX ファイルアクセス方式を維持します。
- 拡張キャッシュ機能とゼロコピーバルク入出力移動機能により、ファイルデータを効率的に移動することができます。
- クラスタファイルシステムには、`fcntl(2)` インタフェースに基づく、高度な可用性を備えたアドバイザリファイルロッキング機能があります。クラスタファイルシステム対してアドバイザリファイルロッキング機能を使えば、複数のクラスタノードで動作するアプリケーションの間で、データのアクセスを同期化できます。ファイルロックを所有するノードがクラスタから切り離されたり、ファイルロックを所有するアプリケーションが異常停止すると、それらのロックはただちに解放されます。

- 障害が発生した場合でも、データへの連続したアクセスが可能です。アプリケーションは、ディスクへのパスが有効であれば、障害による影響を受けません。この保証は、raw ディスクアクセスとすべてのファイルシステム操作で維持されます。
- クラスタファイルシステムは、基本のファイルシステムからもボリュームマネージャーからも独立しています。クラスタシステムファイルは、サポートされているディスク上のファイルシステムすべてを広域にします。

広域デバイスにファイルシステムをマウントするとき、広域にマウントする場合は `mount -g` を使用し、ローカルにマウントする場合は `mount` を使用します。

プログラムは、同じファイル名 (たとえば、`/global/foo`) によって、クラスタ内のすべてのノードからクラスタファイルシステムのファイルにアクセスできます。

クラスタファイルシステムは、すべてのクラスタメンバーにマウントされます。クラスタファイルシステムをクラスタメンバーのサブセットにマウントすることはできません。

クラスタファイルシステムは、特定のファイルシステムタイプではありません。つまり、クライアントは、実際に使用するファイルシステム (UFS など) だけを認識します。

クラスタファイルシステムの使用法

Sun Cluster システムでは、すべての多重ホストディスクがディスクデバイスグループとして構成されています。これは、Solaris ボリュームマネージャー のディスクセット、VxVM のディスクグループ、またはソフトウェアベースのボリューム管理ソフトウェアの制御下にない個々のディスクが該当します。

クラスタファイルシステムを高可用性にするには、使用するディスクストレージが複数のノードに接続されていなければなりません。したがって、ローカルファイルシステム (ノードのローカルディスクに格納されているファイルシステム) をクラスタファイルシステムにした場合は、高可用性にはなりません。

クラスタファイルシステムは、通常のファイルシステムと同様にマウントできます。

- 手作業によるマウント – `mount` コマンドと `-g` または `-o global` マウントオプションを使用し、コマンド行からクラスタファイルシステムをマウントします。次に例を示します。

```
SPARC:# mount -g /dev/global/dsk/d0s0 /global/oracle/data
```

- 自動マウント – `global` マウントオプションによって `/etc/vfstab` ファイルにエントリを作成し、起動時にクラスタファイルシステムをマウントします。さらに、すべてのノードの `/global` ディレクトリ下にマウントポイントを作成します。ディレクトリ `/global` を推奨しますが、ほかの場所 でも構いません。次に、`/etc/vfstab` ファイルの、クラスタファイルシステムを示す行の例を示します。

```
SPARC:/dev/md/oracle/dsk/d1 /dev/md/oracle/rdsk/d1 /global/oracle/data ufs 2 yes global,logging
```

注 – Sun Cluster ソフトウェアには、クラスタファイルシステムに対する特定の命名規則はありません。しかし、/global/*disk-device-group* などのように、同じディレクトリのもとにすべてのクラスタファイルシステムのマウントポイントを作成すると、管理が容易になります。詳細は、Sun Cluster 3.1 9/04 Software Collection for Solaris OS (SPARC Platform Edition) と『Sun Cluster のシステム管理 (Solaris OS 版)』を参照してください。

HAStoragePlus リソースタイプ

HAStoragePlus リソースタイプは、UFS や VxFS などの広域的ではないファイルシステム構成を高可用対応にするように設計されています。HAStoragePlus は、ローカルファイルシステムを Sun Cluster 環境に統合してそのファイルシステムを高可用対応にする場合に使用します。HAStoragePlus は、Sun Cluster でローカルファイルシステムのフェイルオーバーを行うための付加的なファイルシステム機能 (チェック、マウント、強制的なマウント解除など) も提供します。フェイルオーバーを行うには、アフィニティースイッチオーバーが有効になった広域ディスクグループ上にローカルファイルシステムが存在していなければなりません。

HAStoragePlus リソースタイプの使用法については、『Sun Cluster データサービスの計画と管理 (Solaris OS 版)』の「高可用性ローカルファイルシステムの有効化」を参照してください。

HAStoragePlus は、リソースとそのリソースが依存するディスクデバイスグループの起動を同期させるのにも使用できます。詳細は、73 ページの「リソース、リソースグループ、リソースタイプ」を参照してください。

Syncdir マウントオプション

syncdir マウントオプションは、実際に使用するファイルシステムとして UFS を使用するクラスタファイルシステムに使用できます。しかし、syncdir を指定しない方がパフォーマンスは向上します。syncdir を指定した場合、POSIX 準拠の書き込みが保証されます。syncdir を指定しない場合、NFS ファイルシステムの場合と同じ動作となります。たとえば、syncdir を指定しないと、場合によっては、ファイルを閉じるまでスペース不足条件を検出できません。syncdir (および POSIX 動作) を指定すると、スペース不足条件は書き込み動作中に検出されます。syncdir を指定しない場合に問題が生じることはほとんどありません。

SPARC ベースのクラスタを使用している場合、VxFS には、UFS の syncdir マウントオプションと同等なマウントオプションはありません。VxFS の動作は syncdir マウントオプションを指定しない場合の UFS と同じです。

広域デバイスとクラスタファイルシステムの FAQ については、92 ページの「ファイルシステムに関する FAQ」を参照してください。

ディスクパスの監視

現在のリリースの Sun Cluster ソフトウェアは、ディスクパス監視機能 (DPM) をサポートします。この節では、DPM、DPM デーモン、およびディスクパスを監視するときに使用する管理ツールについての概念的な情報を説明します。ディスクパスの状態を監視、監視解除、および表示する手順については、『Sun Cluster のシステム管理 (Solaris OS 版)』を参照してください。

注 – DPM は、Sun Cluster 3.1 10/03 ソフトウェアより前にリリースされたバージョンが動作しているノードではサポートされません。ローリングアップグレードが行われているときには DPM コマンドを使用しないでください。すべてのノードをアップグレードしたら、DPM コマンドを使用する前にこれらのノードをオンラインにする必要があります。

DPM の概要

DPM は、二次ディスクパスの可用性を監視することによって、フェイルオーバーおよびスイッチオーバーの全体的な信頼性を向上させます。リソースを切り替える前には、`scdpm` コマンドを使用して、そのリソースが使用しているディスクパスの可用性を確認します。`scdpm` コマンドのオプションを使用すると、単一ノードまたはクラスタ内のすべてのノードへのディスクパスを監視できます。コマンド行オプションについての詳細は、`scdpm(1M)` のマニュアルページを参照してください。

DPM コンポーネントは `SUNWscu` パッケージからインストールされます。`SUNWscu` パッケージは、標準の Sun Cluster インストール手順でインストールされます。インストールインタフェースについての詳細は、`scinstall(1M)` のマニュアルページを参照してください。次の表に、DPM コンポーネントのデフォルトのインストール場所を示します。

保存場所	コンポーネント
デーモン	`/usr/cluster/lib/sc/scdpmd`
コマンド行インタフェース	`/usr/cluster/bin/scdpm`
共用ライブラリ	`/user/cluster/lib/libscdpm.so`
デーモン状態ファイル (実行時に作成される)	`/var/run/cluster/scdpm.status`

マルチスレッド化された DPM デーモンは各ノード上で動作します。DPM デーモン (`scdpmd`) はノードの起動時に `rc.d` スクリプトによって起動されます。問題が発生した場合、DPM デーモンは `pmfd` によって管理され、自動的に再起動されます。以下で、最初の起動時に `scdpmd` がどのように動作するかについて説明します。

注 – 起動時、各ディスクパスの状態は UNKNOWN に初期化されます。

1. DPM デーモンは、以前の状態ファイルまたは CCR データベースから、ディスクパスとノード名の情報を収集します。CCR についての詳細は、39 ページの「クラスタ構成レポジトリ (CCR)」を参照してください。DPM デーモンの起動後、指定したファイルから監視すべきディスクのリストを読み取るように DPM デーモンに指示できます。
2. DPM デーモンは通信インタフェースを初期化して、デーモンの外部にあるコンポーネント (コマンド行インタフェースなど) からの要求に応えます。
3. DPM デーモンは `scsi_inquiry` コマンドを使用して、監視リストにある各ディスクパスに 10 分ごとに ping を送信します。各エントリはロックされるため、通信インタフェースは監視中のエントリの内容にアクセスできなくなります。
4. DPM デーモンは UNIX の `syslogd(1M)` 機構を通じて、ディスクパスの新しい状態を Sun Cluster Event Framework に通知および記録します。

注 – このデーモンに関連するすべてのエラーは、`pmfd` (1M) で報告されます。API のすべての関数は、成功時に `0` を戻し、失敗時に `-1` を戻します。

DPM デーモンは、Sun StorEdge Traffic Manager、HDLM、PowerPath などのマルチパスドライバを通じて、論理パスの可用性を監視します。このようなマルチパスドライバは物理パスの障害を DPM デーモンから隠すため、DPM デーモンはマルチパスドライバが管理する物理パスを監視できません。

ディスクパスの監視

この節では、クラスタ内のディスクパスを監視するための 2 つの方法について説明します。1 つめの方法は `scdpm` コマンドを使用する方法です。`scdpm` コマンドを使用すると、クラスタ内のディスクパスの状態を監視、監視解除、または表示できます。このコマンドはまた、障害のあるディスクのリストを表示したり、1 つのファイルからディスクパスを監視したりするのにも便利です。

2 つめの方法は、SunPlex Manager の GUI (Graphical User Interface) を使用してクラスタ内のディスクパスを監視する方法です。SunPlex Manager は、クラスタ内の監視しているディスクをトポロジビューで表示します。このトポロジビューは 10 分ごとに更新され、失敗した ping の数が表示されます。SunPlex Manager の GUI が報告する情報と `scdpm(1M)` コマンドを組み合わせて使用すると、ディスクパスを管理できます。SunPlex Manager については、『Sun Cluster のシステム管理 (Solaris OS 版)』の第 10 章「グラフィカルユーザーインタフェースによる Sun Cluster の管理」を参照してください。

scdpm コマンドによるディスクパスの監視

scdpm(1M) コマンドが提供する DPM 管理コマンドを使用すると、次の作業を行うことができます。

- 新しいディスクパスの監視
- ディスクパスの監視解除
- CCR データベースからの構成データの再読み込み
- 指定したファイルからの監視または監視解除すべきディスクの読み取り
- クラスタ内の 1 つまたはすべてのディスクパスの状態の報告
- あるノードからアクセスできるすべてのディスクパスの印刷

任意のアクティブなノードから、ディスクパス引数を付けて scdpm(1M) コマンドを発行することによって、そのクラスタ上で DPM 管理作業を実行できます。ディスクパス引数はノード名とディスク名からなります。ただし、ノード名は必須ではなく、指定しない場合は all がデフォルトで使用されます。次の表に、ディスクパスの命名規約を示します。

注 – 広域ディスクパス名はクラスタ全体で一貫性があるため、ディスクパス名には広域名を使用することを強くお勧めします。UNIX ディスクパス名には、クラスタ全体での一貫性がありません。つまり、あるディスクの UNIX ディスクパスは、クラスタノードによって異なる可能性があります。たとえば、 あるディスクパス名があるノードでは c1t0d0 、別のノードでは c2t0d0 となっている場合があります。UNIX ディスクパス名を使用する場合は、scdidadm -L コマンドを使って UNIX ディスクパス名と広域ディスクパス名を対応付けてから DPM コマンドを実行してください。詳細は、scdidadm(1M) のマニュアルページを参照してください。

表 **3–3** ディスクパス名の例

名前型	ディスクパス名の例	説明
広域ディスクパス	schost-1:/dev/did/dsk/d1	schost-1 ノード上のディスクパス d1
all:d1	クラスタのすべてのノードでのディスクパス d1	
UNIX ディスクパス	schost-1:/dev/rdsk/c0t0d0s0	schost-1 ノード上のディスクパス c0t0d0s0
schost-1:all	schost-1 ノードでのすべてのディスクパス	
すべてのディスクパス	all:all	クラスタのすべてのノードでのすべてのディスクパス

SunPlex Manager によるディスクパスの監視

SunPlex Manager を使用すると、次のような DPM の基本的な管理作業を実行できます。

- ディスクパスの監視
- ディスクパスの監視解除
- クラスタ内のすべてのディスクパスの状態の表示

SunPlex Manager を使用してディスクパスを管理する手順については、SunPlex Manager のオンラインヘルプを参照してください。

定足数と定足数デバイス

ここでは、次の内容について説明します。

注 – Sun Cluster ソフトウェアが定足数デバイスとしてサポートする特定のデバイスの一覧については、Sun のサービスプロバイダにお問い合わせください。

クラスタノードはデータとリソースを共有しており、複数のアクティブなパーティションがあるとデータが壊れる恐れがあるのでクラスタは決して複数のアクティブなパーティションに一度に分割しないでください。クラスタメンバーシップモニター (CMM) および定足数アルゴリズムにより、たとえクラスタ接続がパーティション分割されている場合でも、いつでも同じクラスタのインスタンスが 1 つだけは動作していることが保証されます。

定足数と CMM の概要については、『Sun Cluster の概要 (Solaris OS 版)』の「クラスタメンバーシップ」を参照してください。

クラスタのパーティション分割からは、次の 2 種類の問題が発生します。

- split brain
- amnesia

Split brain は、ノード間のクラスタ接続が失われ、クラスタがサブクラスタにパーティション分割されるときに起きます。あるパーティションのノードはほかのパーティションのノードと通信できないため、各パーティションは自分が唯一のパーティションであると認識します。

amnesia は、停止したクラスタが、停止時よりも古いクラスタ構成データに基づいて再起動されたときに発生します。この問題は、最後に機能していたクラスタパーティションにないノード上のクラスタを起動するときに起きる可能性があります。

Sun Cluster ソフトウェアは、split brain と amnesia を次の操作により回避します。

- 各ノードに 1 つの投票を割り当てる
- 動作中のクラスタの過半数の投票を管理する

過半数の投票数を持つパーティションは、定足数を獲得し、動作可能になります。この過半数の投票メカニズムにより、クラスタ内に 3 つ以上のノードが構成されているときに split brain と amnesia を防ぐことができます。ただし、クラスタ内に 3 つ以上のノードが構成されている場合、ノードの投票数を数えるだけでは十分ではありません。しかし、2 ノードクラスタでは過半数が 2 であるため、このような 2 ノードクラスタがパーティション分割された場合、いずれかのパーティションが定足数を獲得するために外部投票が必要です。この外部からの投票は「定足数デバイス」によって行われます。

定足数投票数について

`scstat -q` コマンドを使って、以下の情報を調べます。

- 構成済み投票数
- 現在の投票数
- 定足数に必要な投票数

このコマンドについての詳細は、`scstat(1M)` のマニュアルページを参照してください。

ノードおよび定足数デバイスの両方がクラスタへの投票に数えられ、定足数を満たすことができます。

ノードは、ノードの状態に応じて投票に数えられます。

- ノードが起動してクラスタメンバーになると、投票数は *1* となります。
- ノードがインストールされているときは、投票数は *0* となります。
- システム管理者がノードを保守状態にすると、投票数は *0* となります。

定足数デバイスは、デバイスに伴う投票数に基づいて、投票に数えられます。定足数デバイスを構成するとき、Sun Cluster ソフトウェアは定足数デバイスに *N*-1 の投票数を割り当てます (*N* は定足数デバイスに伴う投票数)。たとえば、2 つのノードに接続された、投票数がゼロ以外のクォーラムデバイスの投票数は 1 (2–1) になります。

定足数デバイスは、次の 2 つの条件のうちの *1* つを満たす場合に投票に数えられます。

- 定足数デバイスに現在接続されている 1 つ以上のノードがクラスタメンバーである。
- 定足数デバイスに現在接続されている 1 つ以上のノードが起動中で、そのノードは定足数デバイスを所有する最後のクラスタパーティションのメンバーであった。

定足数デバイスを構成するのは、クラスタをインストールするときか、このあとで『Sun Cluster のシステム管理 (Solaris OS 版)』の第 5 章「定足数の管理」で説明されている手順を使用するときです。

障害による影響の防止について

クラスタの主要な問題は、クラスタがパーティション分割される (sprit-brain と呼ばれる) 原因となる障害です。split brain が発生すると、一部のノードが通信できなくなるため、個々のノードまたは一部のノードが個々のクラスタまたは一部のクラスタを形成しようとします。各部分 (つまりパーティション) は、誤って、多重ホストデバイスに対して単独のアクセスと所有権を持つものと認識します。そのため、複数のノードがディスクに書き込もうとすると、データが破壊される可能性があります。

障害による影響の防止機能では、多重ホストデバイスへのノードのアクセスを、ディスクへのアクセスを物理的に防止することによって制限します。障害が発生するかパーティション分割され、ノードがクラスタから切り離されると、障害による影響の防止機能によって、ノードがディスクにアクセスできなくなります。現在のメンバーノードだけが、ディスクへのアクセス権を持つため、データの完全性が保たれます。

ディスクデバイスサービスは、多重ホストデバイスを使用するサービスに対して、フェイルオーバー機能を提供します。現在、ディスクデバイスグループの主ノード (所有者) として機能しているクラスタメンバーに障害が発生するか、またはこのメンバーに到達できなくなると、新しい主ノードが選択されます。この新しい主ノードによって、ディスクデバイスグループはほんのわずかの中断だけで機能し続けることができます。このプロセス中、新しい主ノードが起動される前に、古い主ノードはデバイスへのアクセスを放棄する必要があります。ただし、あるメンバーがクラスタから切り離されて到達不能になると、クラスタはそのノードに対して、主ノードであったデバイスを解放するように通知できません。したがって、存続するメンバーが、障害の発生したメンバーから広域デバイスを制御してアクセスできるようにする手段が必要です。

Sun Cluster システムは、SCSI ディスクリザベーションを使用して、二重障害の防止機能を実装します。SCSI リザベーションを使用すると、障害が発生したノードは多重ホストデバイスから阻止され、これらのディスクにアクセスできなくなります。

SCSI-2 ディスクリザベーションがサポートするリザベーションの形式には、ディスクに接続されているすべてのノードにアクセスを付与するものと (リザベーションが設定されていない場合)、単一ノード (リザベーションを保持するノード) だけにアクセスを制限するものがあります。

クラスタメンバーは、別のノードがクラスタインターコネクトを介して通信していないことを検出すると、障害による影響の防止手順を開始して、そのノードが共有ディスクへアクセスするのを防止します。この二重障害の防止機能が動作すると、アクセスを阻止されたノードはパニック状態になり、そのコンソールに「reservation conflict」メッセージが表示されます。

あるノードがすでにクラスタメンバーでないことが検出されると、そのノードとほかのノード間で共有されていたすべてのディスクに対して SCSI リザベーションが行われます。阻止されているノードは自分が阻止されていることを認識しない場合があるため、共有ディスクのどれかにアクセスしようとしてリザベーションを検出すると、そのノードはパニックします。

障害の影響を防止するフェイルファースト機構

異常のあるノードが再起動され、共有ストレージに書き込むのを防ぐクラスタフレームワークの機構をフェイルファーストといいます。

クラスタのメンバーである各ノードでは、定足数ディスクを含むアクセス可能な個々のディスクに対し ioctl (MHIOCENFAILFAST) が連続的に有効にされます。この ioctl はディスクドライバに対する命令です。あるノードがほかのノードによってリザベーションされているディスクにアクセスできない場合、この ioctl を使用すると、このノードは自らをパニックする (強制的に停止する) ことができます。

MHIOCENFAILFAST ioctl が有効になっていると、ドライバは、ノードからそのディスクに対して出されるすべての読み取りや書き込みからのエラーに、`Reservation_Conflict` エラーコードが含まれていないか検査します。ioctl はバックグラウンドでディスクに対して周期的にテスト操作を行い、`Reservation_Conflict` がないか検査します。`Reservation_Conflict` が返されると、フォアグラウンドとバックグラウンドのコントロールフローパスが両方ともパニックを発生します。

SCSI-2 ディスクの場合、リザベーションは永続的ではないため、ノードが再起動されると無効になります。Persistent Group Reservation (PGR) の SCSI-3 ディスクでは、リザベーション情報はそのディスクに格納されるため、ノードが再起動されても有効です。SCSI-2 ディスクでも SCSI-3 ディスクでも、フェイルファースト機構の機能は同じです。

定足数を獲得できるパーティションに属していないノードが、クラスタ内の他のノードとの接続を失うと、そのノードは別のノードによってクラスタから強制的に切り離されます。定足数を達成可能なパーティションに参加している別のノードが、共有ディスクにリザベーションを発行します。定足数を持たないノードが共有ディスクにアクセスしようとすると、そのノードはリザベーション衝突のエラーを受け取り、フェイルファースト機構に基づいてパニックします。

パニック後、ノードは再起動してクラスタに再度加わろうとするか、またはクラスタが SPARC ベースのシステムで構成されている場合は、OpenBoot™ PROM (OBP) プロンプトのままになります。どちらのアクションをとるかは、`auto-boot?` パラメー

タの設定に依存します。auto-boot? は、SPARC ベースのクラスタでは OpenBoot PROM の ok プロンプトから eeprom(1M) を使用することで設定できます。X86 ベースのクラスタでは、BIOS のブート後に SCSI ユーティリティーを起動することで設定できます。

定足数の構成について

次に、定足数の構成について示します。

- 定足数デバイスには、ユーザーデータを含むことができます。
- N+1 の構成 (*N* 個の定足数デバイスがそれぞれ、1 から *N* までのノードのうちの 1 つのノードと N+1 番目のノードに接続されている構成) では、1 から *N* までのどのノードで障害が発生しても、*N*/**2** 個のうちの任意のノードに障害が発生しても、そのクラスタは影響を受けません。この可用性は、定足数デバイスが正しく機能していることを前提にしています。
- *N* ノード構成 (1 つの定足数デバイスがすべてのノードに接続されている構成) では、*N*-**1** 個のうちの任意のノードに障害が発生しても、そのクラスタは影響を受けません。この可用性は、定足数デバイスが正しく機能していることを前提にしています。
- 1 つの定足数デバイスがすべてのノードに接続している *N* ノード構成では、すべてのクラスタノードが使用できる場合、定足数デバイスに障害が起きてもクラスタは影響を受けません。

回避すべき定足数の構成例については、60 ページの「望ましくない定足数の構成」を参照してください。推奨される定足数の構成例については、58 ページの「推奨される定足数の構成」を参照してください。

定足数デバイス要件の順守

以下の要件を守る必要があります。これらの要件を無視すると、クラスタの可用性が損なわれる場合があります。

- Sun Cluster ソフトウェアがご使用のデバイスを定足数デバイスとしてサポートしていることを確認します。

注 – Sun Cluster ソフトウェアが定足数デバイスとしてサポートする特定のデバイスの一覧については、Sun のサービスプロバイダにお問い合わせください。

Sun Cluster ソフトウェアは、次の 2 種類の定足数デバイスをサポートしています。

- SCSI-3 PGR リザベーションに対応した多重ホスト共有ディスク
- SCSI-2 リザベーションに対応した二重ホスト共有ディスク

- 2 ノード構成では、1 つのノードに障害が起きてももう 1 つのノードが動作を継続できるように、少なくとも 1 つの定足数デバイスを構成する必要があります。詳細は、図 3–2を参照してください。

回避すべき定足数の構成例については、60 ページの「望ましくない定足数の構成」を参照してください。推奨される定足数の構成例については、58 ページの「推奨される定足数の構成」を参照してください。

定足数デバイスのベストプラクティスの順守

以下の情報を使用して、ご使用のトポロジに最適な定足数の構成を評価してください。

- クラスタの全ノードに接続できるデバイスがありますか。
  - ある場合は、そのデバイスを 1 つの定足数デバイスとして構成してください。この構成は最適な構成なので、別の定足数デバイスを構成する必要はありません。

注意 – この要件を無視して別の定足数デバイスを追加すると、追加した定足数デバイスによってクラスタの可用性が低下します。

  - ない場合は、1 つまたは複数のデュアルポートデバイスを構成してください。
- 定足数デバイスにより提供される投票の合計数が、ノードにより提供される投票の合計数より必ず少なくなるようにします。少なくなければ、すべてのノードが機能していても、すべてのディスクを使用できない場合、そのノードはクラスタを形成できません。

注 – 特定の環境によっては、自分のニーズに合うように、全体的なクラスタの可用性を低くした方が望ましい場合があります。このような場合には、このベストプラクティスを無視できます。ただし、このベストプラクティスを守らないと、全体の可用性が低下します。たとえば、60 ページの「変則的な定足数の構成」に記載されている構成では、クラスタの可用性は低下し、定足数の投票がノードの投票を上回ります。このクラスタは、`Nodes A` と `Node B` 間にある共有ストレージへのアクセスが失われると、クラスタ全体に障害が発生するという性質を持っています。

このベストプラクティスの例外については、60 ページの「変則的な定足数の構成」を参照してください。

- 記憶装置へのアクセスを共有するノードのすべてのペア間で定足数デバイスを指定します。この定足数の構成により、障害からの影響の防止プロセスが高速化されます。詳細は、58 ページの「2 ノードより大きな構成での定足数」を参照してください。
- 通常、定足数デバイスの追加によりクラスタの投票の合計数が同じになる場合、クラスタ全体の可用性は低下します。

- ノードを追加したり、ノードに障害が発生すると、定足数デバイスの再構成は少し遅くなります。従って、必要以上の定足数デバイスを追加しないでください。

回避すべき定足数の構成例については、60 ページの「望ましくない定足数の構成」を参照してください。推奨される定足数の構成例については、58 ページの「推奨される定足数の構成」を参照してください。

推奨される定足数の構成

この節では、推奨される定足数の構成例を示します。回避すべき定足数の構成例については、60 ページの「望ましくない定足数の構成」を参照してください。

2 ノード構成の定足数

2 ノードのクラスタを形成するには、2 つの定足投票数が必要です。これらの 2 つの投票数は、2 つのクラスタノード、または 1 つのノードと 1 つの定足数デバイスのどちらかによるものです。

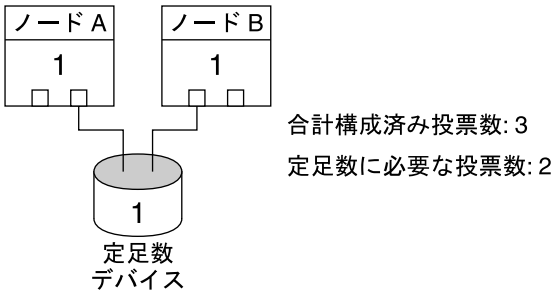
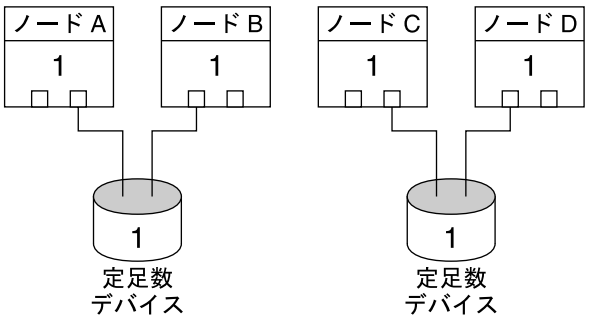


図 **3–2** 2 ノード構成

2 ノードより大きな構成での定足数

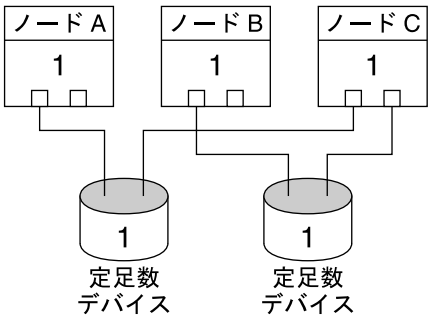
定足数デバイスを持たない、2 ノードよりも大きなクラスタも構成できます。ただし、このようなクラスタを構成した場合、そのクラスタはクラスタ内の過半数のノードなしには開始できません。



合計構成済み投票数: 6

定足数に必要な投票数: 4

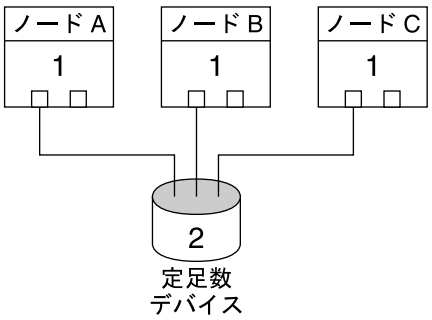
この構成は、どちらかのペアが稼働し続けるためには
各ペアが稼働していなければならない。



合計構成済み投票数: 5

定足数に必要な投票数: 3

この構成は、通常、アプリケーションがノード A とノード B で実行
されるように構成され、ノード C をホットスペアとして使用する。



合計構成済み投票数: 5

定足数に必要な投票数: 3

この構成は、任意の 1 つ以上のノードと定足数デバイスとの
組み合わせでクラスタを形成できる。

変則的な定足数の構成

図 3–3 では、Node A と Node B でミッションクリティカルなアプリケーション (Oracle データベースなど) を実行していると仮定します。ノード A とノード B を使用できず、共有データにアクセスできない場合、クラスタ全体を停止させる必要がある場合があります。停止させない場合、この構成は高可用性を提供できないため、最適な構成とはなりません。

この例外に関するベストプラクティスについては、57 ページの「定足数デバイスのベストプラクティスの順守」を参照してください。

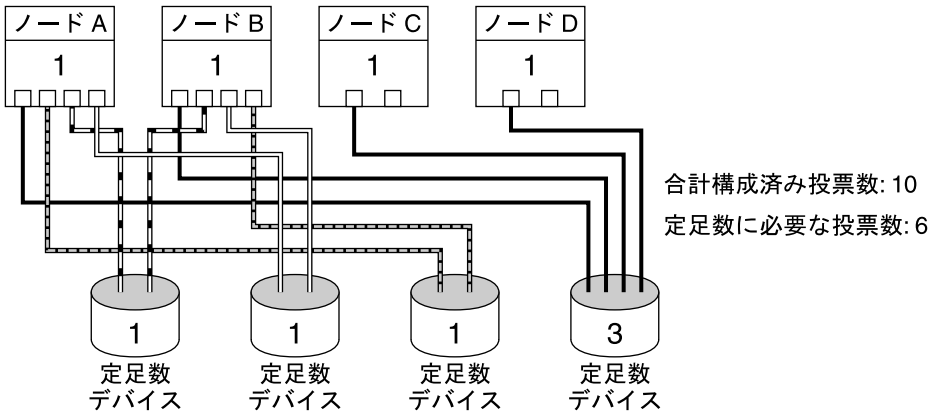
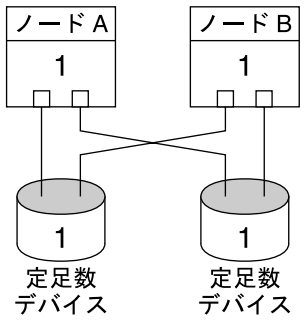


図 **3–3** 変則的な構成

望ましくない定足数の構成

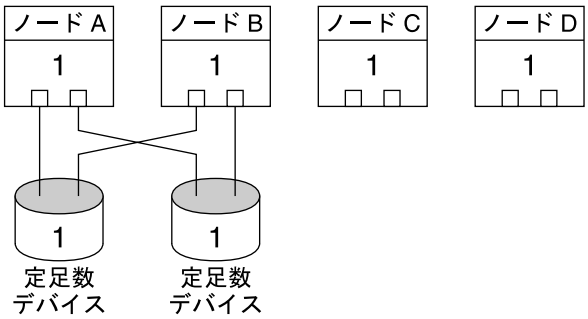
この節では、回避すべき定足数の構成例を示します。推奨される定足数の構成例については、58 ページの「推奨される定足数の構成」を参照してください。



合計構成済み投票数: 4

定足数に必要な投票数: 3

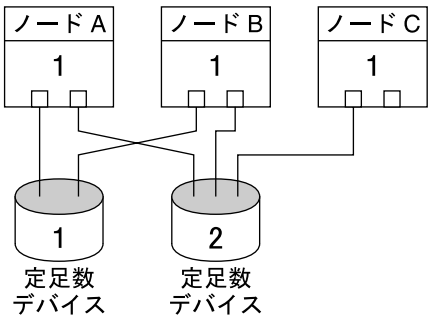
この構成は、定足数デバイス投票が、ノードの投票数より少なくなければ
ならないというベストプラクティスに反する。



合計構成済み投票数: 6

定足数に必要な投票数: 4

この構成は、定足数デバイスを追加して、合計投票数を等しくするべきではない
というベストプラクティスに反する。この構成では、可用性は向上されない。



合計構成済み投票数: 6

定足数に必要な投票数: 4

この構成は、定足数デバイス投票が、ノードの投票数より少なくなければ
ならないというベストプラクティスに反する。

データサービス

「データサービス」という用語は、Oracle や Sun Java System Web Server など、単一のサーバーではなく、クラスタで動作するように構成されたアプリケーションを意味します。データサービスは、アプリケーション、専用の Sun Cluster 構成ファイル、および、アプリケーションの次のアクションを制御する Sun Cluster 管理メソッドからなります。

- 開始
- 停止
- 監視と訂正手段の実行

データサービスタイプについては、『Sun Cluster の概要 (Solaris OS 版)』の「データサービス」を参照してください。

図 3–4に、単一のアプリケーションサーバーで動作するアプリケーション (単一サーバーモデル) と、クラスタで動作する同じアプリケーション (クラスタサーバーモデル) との比較を示します。これら 2 つの構成の唯一の違いは、クラスタ化されたアプリケーションの動作がより速くなり、その可用性もより高くなることです。

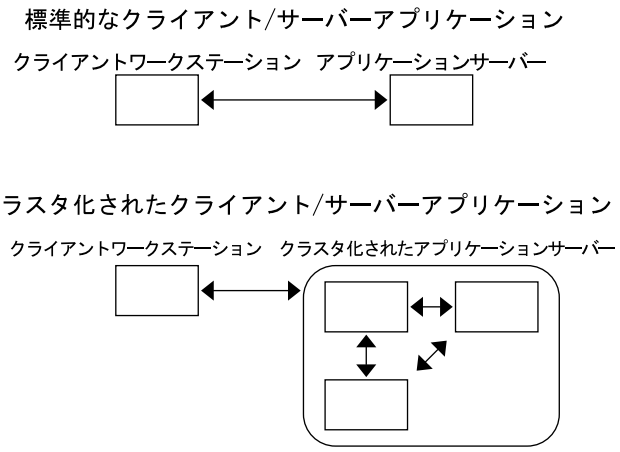


図 **3–4** 標準的なクライアントサーバー構成とクラスタ化されたクライアントサーバー構成

単一サーバーモデルでは、特定のパブリックネットワークインタフェース (ホスト名) を介してサーバーにアクセスするようにアプリケーションを構成します。ホスト名は、この物理サーバーに関係付けられています。

クラスタサーバーモデルでは、パブリックネットワークインタフェースは「論理ホスト 名」または「共有アドレス」です。「ネットワークリソース」は、論理ホスト名と共有アドレスの両方を表します。

一部のデータサービスでは、ネットワークインタフェースとして論理ホスト名か共有アドレスのいずれかを指定する必要があります。論理ホスト名と共有アドレスは相互に交換できません。しかし、別のデータサービスでは、論理ホスト名や共有アドレスをどちらでも指定することができます。どのようなタイプのインタフェースを指定する必要があるかどうかについては、各データサービスのインストールや構成の資料を参照してください。

ネットワークリソースは、特定の物理サーバーと関連付けられているわけではありません。ネットワークリソースは、ある物理サーバーから別の物理サーバーに移すことができます。

ネットワークリソースは、当初、1 つのノード (一次ノード) に関連付けられています。主ノードで障害が発生すると、ネットワークリソースとアプリケーションリソースは別のクラスタノード (二次ノード) にフェイルオーバーされます。ネットワークリソースがフェイルオーバーされても、アプリケーションリソースは、短時間の遅れの後に二次ノードで動作を続けます。

図 3–5に、単一サーバーモデルとクラスタサーバーモデルとの比較を示します。クラスタサーバーモデルのネットワークリソース (この例では論理ホスト名) は、 複数のクラスタノード間を移動できます。アプリケーションは、特定のサーバーに関連付けられたホスト名として、この論理ホスト名を使用するように設定されます。

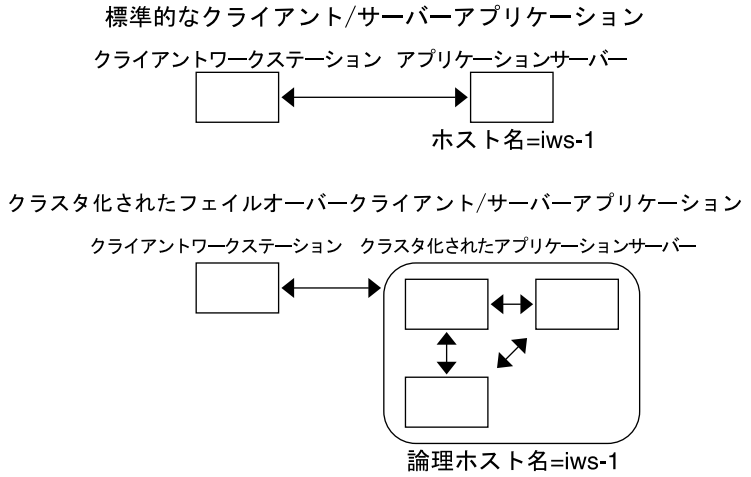


図 **3–5** 固定ホスト名と論理ホスト名

共有アドレスも最初は 1 つのノードに対応付けられます。このノードのことを「広域インタフェースノード」といいます。共有アドレスは「広域インタフェース」といい、クラスタへの単一ネットワークインタフェースとして使用されます。

論理ホスト名モデルとスケーラブルサービスモデルの違いは、スケーラブルサービスモデルでは、各ノードのループバックインタフェースにも共有アドレスがアクティブに構成される点です。この構成では、データサービスの複数のインスタンスをいくつかのノードで同時にアクティブにすることができます。「スケーラブルサービス」という用語は、クラスタノードを追加してアプリケーションの CPU パワーを強化すれば、性能が向上することを意味します。

広域インタフェースノードに障害が発生した場合、共有アドレスは同じアプリケーションのインスタンスが動作している別のノードで起動できます。これによって、このノードが新しい広域インタフェースノードになります。または、共有アドレスを、このアプリケーションを実行していない別のクラスタノードにフェイルオーバーすることができます。

図 3–6 に、単一サーバー構成とクラスタ化されたスケーラブルサービス構成との比較を示します。スケーラブルサービス構成では、共有アドレスがすべてのノードに設定されています。フェイルオーバーデータサービスに論理ホスト名が使用される場合と同じように、アプリケーションは、特定のサーバーに関連付けられたホスト名の代わりにこの共有アドレスを使用するように設定されます。

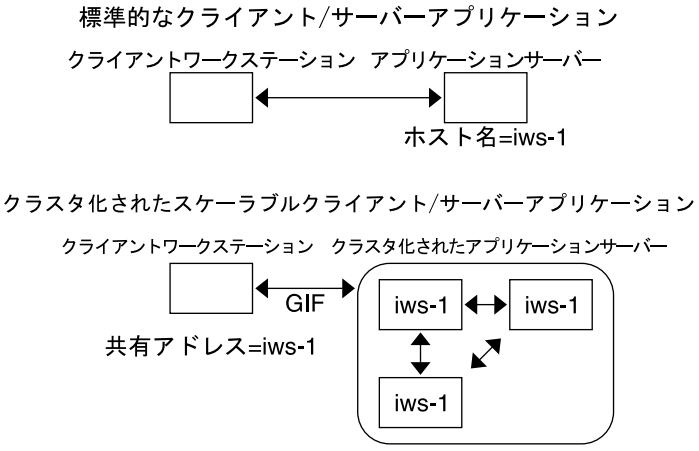


図 **3–6** 固定ホスト名と共有アドレス

データサービスメソッド

Sun Cluster ソフトウェアは、いくつかのサービス管理メソッドを提供しています。これらのメソッドは リソースグループマネージャー (RGM) の制御下で動作して、クラスタノード上のアプリケーションを起動、停止、および監視するのに使用されます。これらのメソッドとクラスタフレームワークソフトウェアおよび多重ホストデバイスにより、アプリケーションは、フェイルオーバーデータサービスやスケーラブルデータサービスとして機能します。

さらに、RGM は、アプリケーションのインスタンスやネットワークリソース (論理ホスト名と共有アドレス) といったクラスタのリソースを管理します。

Sun Cluster ソフトウェアが提供するメソッドのほかにも、Sun Cluster システムは API やいくつかのデータサービス開発ツールを提供します。これらのツールを使用すれば、アプリケーション開発者は、独自のデータサービスメソッドを開発することによって、ほかのアプリケーションを Sun Cluster ソフトウェアの下で高可用性データサービスとして実行できます。

フェイルオーバーデータサービス

データサービスが実行されているノード (主ノード) に障害が発生すると、サービスは、ユーザーによる介入なしで別の作業ノードに移行します。フェイルオーバーサービスは、アプリケーションインスタンスリソースとネットワークリソース (「論理ホスト名」) のコンテナである「フェイルオーバーリソースグループ」を使用します。論理ホスト名とは、1 つのノードに構成して、後で自動的に元のノードや別のノードに構成できる IP アドレスのことです。

フェイルオーバーデータサービスでは、アプリケーションインスタンスは単一ノードでのみ実行されます。フォルトモニターは、エラーを検出すると、そのインスタンスを同じノードで再起動しようとするか、別のノードで起動 (フェイルオーバー) しようとします。その結果は、データサービスの構成によって異なります。

スケーラブルデータサービス

スケーラブルデータサービスは、複数ノードのアクティブインスタンスに対して効果があります。スケーラブルサービスは、2 つのリソースグループを使用します。

- 「スケーラブルリソースグループ」には、アプリケーションリソースが含まれます。
- フェイルオーバーリソースグループには、スケーラブルサービスが依存するネットワークリソース (「共有アドレス」) が含まれます。

スケーラブルリソースグループは、複数のノードでオンラインにできるため、サービスの複数のインスタンスを一度に実行できます。共有アドレスのホストとなるフェイルオーバーリソースグループは、一度に 1 つのノードでしかオンラインにできません。スケーラブルサービスをホストとするすべてのノードは、サービスをホストするための同じ共有アドレスを使用します。

サービス要求は、単一ネットワークインタフェース (広域インタフェース) を通じてクラスタに入ります。これらの要求は、事前に定義されたいくつかのアルゴリズムの 1 つに基づいてノードに分配されます。これらのアルゴリズムは「負荷均衡ポリシー」によって設定されます。クラスタは、負荷均衡ポリシーを使用し、いくつかのノード間でサービス負荷均衡をとることができます。ほかの共有アドレスをホストしている異なるノード上には、複数の広域インタフェースが存在する可能性があります。

スケーラブルサービスの場合、アプリケーションインスタンスはいくつかのノードで同時に実行されます。広域インタフェースのホストとなるノードに障害が発生すると、広域インタフェースは別のノードで処理を続行します。動作していたアプリケーションインスタンスが停止した場合、そのインスタンスは同じノード上で再起動しようとします。

アプリケーションインスタンスを同じノードで再起動できず、別の未使用のノードがサービスを実行するように構成されている場合、サービスはその未使用ノードで処理を続行します。そうしないと、サービスは残りのノード上で動作し続け、サービスのスループットが低下することがあります。

注 – 各アプリケーションインスタンスの TCP 状態は、広域インタフェースノードではなく、インスタンスを持つノードで維持されます。したがって、広域インタフェースノードに障害が発生しても接続には影響しません。

図 3–7 に、フェイルオーバーリソースグループとスケーラブルリソースグループの例と、スケーラブルサービスにとってそれらの間にどのような依存関係があるのかを示します。この例は、3 つのリソースグループを示しています。フェイルオーバーリソースグループには、可用性の高い DNS のアプリケーションリソースと、可用性の高い DNS および可用性の高い Apache Web Server (SPARC ベースのクラスタに限って使用可能) の両方から使用されるネットワークリソースが含まれます。スケーラブルリソースグループには、Apache Web Server のアプリケーションインスタンスだけが含まれます。リソースグループの依存関係は、スケーラブルリソースグループとフェイルオーバーリソースグループの間に存在します (実線)。また、Apache アプリケーションリソースはすべて、共有アドレスであるネットワークリソース `schost-2` に依存します (破線)。

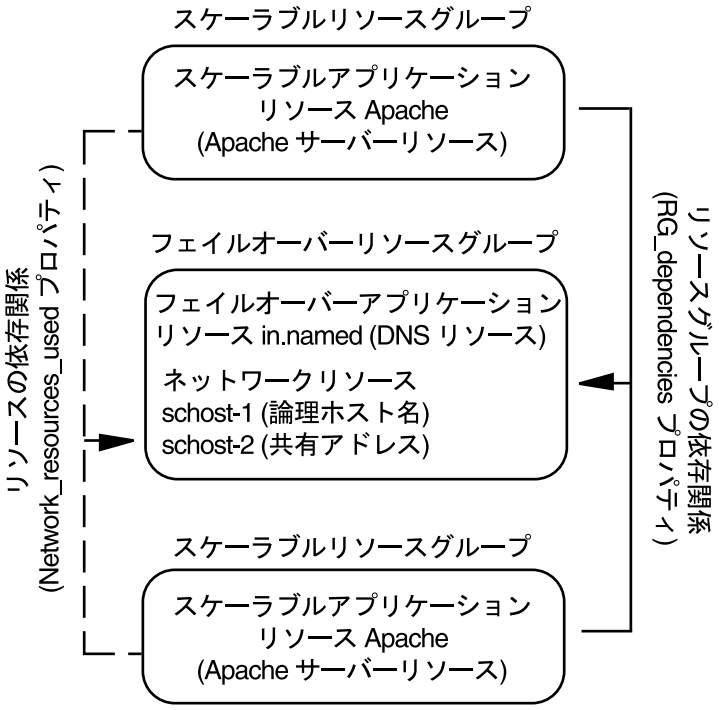


図 **3–7** SPARC: フェイルオーバーリソースグループとスケーラブルリソースグループの例

負荷均衡ポリシー

負荷均衡は、スケーラブルサービスのパフォーマンスを応答時間とスループットの両方の点で向上させます。スケーラブルデータサービスには 2 つのクラスがあります。

- pure
- sticky

pure サービスでは、任意のインスタンスがクライアント要求に応答できます。*sticky* サービスでは、クライアントは同じインスタンスに応答を送信できます。これらの要求は、別のインスタンスには変更されません。

pure サービスは、ウェイト設定した (weighted) 負荷均衡ポリシーを使用します。この負荷均衡ポリシーのもとでは、クライアント要求は、デフォルトで、クラスタ内のサーバーインスタンスに一律に分配されます。たとえば、3 ノードクラスタにおいて、各ノードのウェイトが 1 だとします。各ノードは、そのサービスに対する任意のクライアントからの要求を 1/3 ずつを負担します。このウェイトは、`scrgadm(1M)` コマンドインタフェースまたは SunPlex Manager GUI を使用すると、いつでも変更できます。

sticky サービスには「ordinary sticky」と「wildcard sticky」の 2 種類があります。sticky サービスを使用すると、内部状態メモリーを共有でき (アプリケーションセッション状態)、複数の TCP 接続でアプリケーションレベルの同時セッションが可能です。

ordinary sticky サービスを使用すると、クライアントは、複数の同時 TCP 接続で状態を共有できます。このクライアントを、単一ポートで待機するサーバーインスタンスに対して「sticky」であるといいます。クライアントは、インスタンスが起動していてアクセス可能であり、負荷均衡ポリシーがサービスのオンライン時に変更されていなければ、すべての要求が同じサーバーのインスタンスに送られることを保証されます。

たとえば、クライアント上の Web ブラウザは、3 つの異なるTCP 接続を使用して、ポート 80 にある共有 IP アドレスに接続します。そして、これらの接続はサービスでキャッシュされたセッション情報をお互いに交換します。

sticky ポリシーを一般化すると、そのポリシーは同じインスタンスの背後でセッション情報を交換する複数のスケーラブルサービスにまで及びます。これらのサービスが同じインスタンスの背後でセッション情報を交換するとき、同じノードで異なるポートと通信する複数のサーバーインスタンスに対して、そのクライアントは「sticky」であるといいます。

たとえば、電子商取引サイトの顧客はポート 80 の HTTP を使用して買い物をします。そして、購入した製品をクレジットカードで支払うときには、ポート 443 の SSL に切り替えて機密データを送ります。

Wildcard sticky サービスは、動的に割り当てられたポート番号を使用しますが、クライアント要求が同じノードに送りかえされると想定します。クライアントは、同じIPアドレスを持っているポートに対して「sticky wildcard」であるといいます。

このポリシーの例としては、受動モード FTP があります。たとえば、クライアントはポート 21 の FTP サーバーに接続します。次に、サーバーは、動的ポート範囲のリスナーポートサーバーに接続し直すようにクライアントに指示します。この IP アドレスに対する要求はすべてサーバーが制御情報を通じてクライアントに通知したのと同じノードに転送されます。

このような各 sticky ポリシーでは、ウェイト設定した負荷均衡ポリシーがデフォルトで有効です。したがって、クライアントの最初の要求は、負荷均衡によって指定されたインスタンス宛てに送られます。インスタンスが動作しているノードとのアフィニティーをクライアントが確立すると、今後の要求はそのインスタンス宛てに送られます。ただし、そのノードはアクセス可能であり、負荷均衡ポリシーが変更されていない必要があります。

次に、特定の負荷均衡ポリシーの詳細について説明します。

- weighted - 負荷は指定されたウェイト値に従って各種のノードに分配されます。負荷は指定されたウェイト値に従って各種のノードに分配されます。このポリシーは `Load_balancing_weights` プロパティーに設定された `LB_WEIGHTED` の値を使用して設定されます。ウェイトがノードについて明示的に設定されていない場合は、デフォルトで 1 が設定されます。

ウェイト設定したポリシーは、一定の割合のクライアントトラフィックを特定ノードに送るためのものです。たとえば、X=「ウエイト」、A=「すべてのアクティブノードの合計ウエイト」であるとします。アクティブノードは、新しい接続数の合計の約 X/A がこのアクティブノード宛てに送られると予測できます。ただし、接続数の合計は十分に大きな数である必要があります。このポリシーは、個々の要求には対応しません。

このポリシーは、ラウンドロビンではないことに注意してください。ラウンドロビンポリシーでは、常に、同じクライアントからの要求はそれぞれ異なるノードに送られます。たとえば、1 番目の要求はノード 1 に、2 番目の要求はノード 2 に送られます。

- sticky -このポリシーでは、ポートの集合が、アプリケーションリソースの構成時に認識されます。このポリシーは、`Load_balancing_policy` リソースプロパティーの `LB_STICKY` の値を使用して設定されます。
- sticky-wild - このポリシーは、通常の “sticky” ポリシーの上位セットです。IP アドレスによって識別されるスケーラブルサービスでは、ポートはサーバーによって割り当てられます。したがって、事前に認識できません。ポートは変更されることがあります。このポリシーは、`Load_balancing_policy` リソースプロパティーの `LB_STICKY_WILD` の値を使用して設定されます。

フェイルバック設定

リソースグループは、ノードからノードへ処理を継続します。このようなフェイルオーバーが発生すると、二次ノードが新しい主ノードになります。フェイルバック設定は、本来の主ノードがオンラインに戻ったときの動作を指定します。つまり、本来の主ノードを再び主ノードにする (フェイルバックする) か、現在の主ノードをそのままにするかです。これを指定するには、`Failback` リソースグループプロパティー設定を使用します。

リソースグループをホストしていた本来の主ノードに障害が発生し、繰り返し再起動する場合は、フェイルバックを設定することよって、リソースグループの可用性が低くなる可能性もあります。

データサービス障害モニター

Sun Cluster の各データサービスには、データサービスを定期的に検査してその状態を判断するフォルトモニターがあります。障害モニターは、アプリケーションデーモンが実行しているか、クライアントがサービスを受けているかどうかを検証します。探索によって得られた情報をもとに、デーモンの再起動やフェイルオーバーの実行などの事前に定義された処置が開始されます。

新しいデータサービスの開発

Sun が提供する構成ファイルや管理メソッドのテンプレートを使用することで、さまざまなアプリケーションをクラスタ内でフェイルオーバーサービスやスケーラブルサービスとして実行できます。フェイルオーバーサービスやスケーラブルサービスとして実行するアプリケーションが Sun から提供されていない場合は、代わりの方法があります。つまり、Sun Cluster API や DSET API を使用して、フェイルオーバーサービスやスケーラブルサービスとして動作するようにアプリケーションを構成します。しかし、必ずしもすべてのアプリケーションがスケーラブルサービスになるわけではありません。

スケーラブルサービスの特徴

アプリケーションがスケーラブルサービスになれるかどうかを判断するには、いくつかの基準があります。アプリケーションがスケーラブルサービスになれるかどうかを判断する方法については、『Sun Cluster データサービス開発ガイド (Solaris OS 版)』の「アプリケーションの適合性の分析」を参照してください。次に、これらの基準の要約を示します。

- 第 1 に、このようなサービスは 1 つまたは複数のサーバーインスタンスから構成されます。各インスタンスは、クラスタの異なるノードで実行されます。同じサービスの複数のインスタンスを、同じノードで実行することはできません。
- 第 2 に、このサービスが外部の論理データストアを提供する場合は、十分に注意する必要があります。このストアに複数のサーバーインスタンスから並行にアクセスする場合、このような並行アクセスの同期をとることによって、更新を失ったり、変更中のデータを読み取ったりすることを避ける必要があります。「外部」という用語は、このストアとメモリー内の状態を区別するために使用しています。「論理」という用語は、ストア自体複製されている場合でも、単一の実体として見えることを表します。さらに、この論理データストアには、サーバーインスタンスがデータストアを更新するたびに、その更新がすぐにほかのインスタンスで見られるという特性があります。

  Sun Cluster システムは、このような外部記憶領域をそのクラスタファイルシステムと広域 raw パーティションを介して提供します。例として、サービスが外部ログファイルに新しいデータを書き込む場合や既存のデータを修正する場合を想定してください。このサービスのインスタンスが複数実行されている場合、各インスタンスはこの外部ログへのアクセスを持ち、このログに同時にアクセスできます。各インスタンスは、このログに対するアクセスの同期をとる必要があります。そうしないと、インスタンスは相互に妨害しあうことになります。このサービスは、`fcntl(2)` と `lockf(3C)` による通常の Solaris ファイルロック機能を使用することによって、必要な同期をとることができます。

  この種類のデータストアのもう一つの例はバックエンドデータベースで、たとえば、Oracle や SPARC ベースのクラスタ用の高可用性 Oracle Real Application Clusters Guard などがあります。この種類のバックエンドデータベースサーバーには、データベース照会または更新トランザクションによる同期が組み込まれていま

す。したがって、複数のサーバーインスタンスが独自の同期を実装する必要はありません。

スケーラブルサービスではないサービスの例としては、Sun の IMAP サーバーがあります。このサービスは記憶領域を更新しますが、その記憶領域はプライベートであり、複数の IMAP インスタンスがこの記憶領域に書き込むと、更新の同期がとられないために相互に上書きし合うことになります。IMAP サーバーは、同時アクセスの同期をとるよう書き直す必要があります。

- 最後に、インスタンスは、ほかのインスタンスのデータから切り離されたプライベートデータを持つ場合があることに注意してください。このような場合、データはプライベートであり、そのインスタンスだけがデータを処理するため、サービスは並行アクセスの同期をとる必要はありません。この場合、個人的なデータをクラスタファイルシステムに保存すると、これらのデータが広域にアクセス可能になる可能性があるため、十分に注意する必要があります。

データサービス API と DSDL API

Sun Cluster システムには、アプリケーションの可用性を高めるものとして次の機能があります。

- Sun Cluster システムの一部として提供されるデータサービス
- データサービス API
- データサービス用の開発ライブラリ API
- 汎用データサービス

Sun Cluster システムが提供するデータサービスをインストールおよび構成する方法については、『Sun Cluster データサービスの計画と管理 (Solaris OS 版)』を参照してください。Sun Cluster フレームワークでアプリケーションの可用性を高める方法については、Sun Cluster 3.1 9/04 Software Collection for Solaris OS (SPARC Platform Edition)を参照してください。

アプリケーション開発者は、Sun Cluster API を使用することによって、データサービスインスタンスの起動や停止を行なう障害モニターやスクリプトを開発できます。これらのツールを使用すると、アプリケーションをフェイルオーバーまたはスケーラブルデータサービスとして実装できます。Sun Cluster システムは「汎用」のデータサービスを提供します。この汎用のデータサービスを使用すれば、簡単に、アプリケーションに必要な起動メソッドと停止メソッドを生成して、データサービスをフェイルオーバーサービスまたはスケーラブルサービスとして実装できます。

クラスタインターコネクトによるデータサービストラフィックの送受信

クラスタには、ノード間を結ぶ複数のネットワーク接続が必要です。クラスタインターコネクトは、これらの接続から構成されています。Sun Cluster ソフトウェアは、複数のインターコネクトを使用して、次の目標を達成します。

- 高可用性を保証します。
- 性能を向上させます。

内部トラフィック (ファイルシステムのデータやスケーラブルサービスのデータなど) では、メッセージが、すべての使用可能なインターコネクト上にラウンドロビン方式でストライプ化されます。クラスタインターコネクトは、ノード間の通信の可用性を高めるためにアプリケーションから使用することもできます。たとえば、分散アプリケーションでは、個々のコンポーネントが異なるノードで動作することがあり、その場合には、ノード間の通信が必要になります。パブリック伝送の代わりにクラスタインターコネクトを使用することで、個別のリンクに障害が発生しても、接続を持続することができます。

ノード間の通信にクラスタインターコネクトを使用するには、Sun Cluster のインストール時に設定したプライベートホスト名をアプリケーションで使用する必要があります。たとえば、`node 1` のプライベートホスト名が `clusternode1-priv` である場合、クラスタインターコネクトを経由して `node 1` と通信するときにこの名前を使用する必要があります。この名前を使用してオープンされた TCP ソケットは、クラスタインターコネクトを経由するように経路指定され、ネットワークに障害が発生した場合でも、この TCP ソケットは透過的に再経路指定されます。

Sun Cluster のインストール時に複数のプライベートホスト名が設定されているため、クラスタインターコネクトでは、そのときに選択した任意の名前を使用できます。実際の名前を取得するには、`scha_cluster_get(3HA)` コマンドに `scha_privatelink_hostname_node` 引数を指定して実行します。

アプリケーション通信と内部クラスタ通信は両方とも、すべてのインターコネクト上でストライプ化されます。アプリケーションは内部クラスタトラフィックとクラスタインターコネクトを共有するため、アプリケーションが使用できる帯域幅は、ほかのクラスタトラフィックに使用される帯域幅に左右されます。障害が発生すると、内部トラフィックとアプリケーショントラフィックは利用できるすべてのインターコネクト上でストライプ化されます。

各ノードには、固定した pernode アドレスが割り当てられます。この pernode アドレスは、`clprivnet` ドライブで探索されます。IP アドレスは、ノードのプライベートホスト名にマッピングされます。つまり、`clusternode1-priv` です。Sun Cluster プライベートネットワークドライバについては、`clprivnet(7)` のマニュアルページを参照してください。

アプリケーション全体で一貫した IP アドレスが必要な場合、クライアント側とサーバー側の両方でこの pernode アドレスにバインドするようにアプリケーションを設定します。これによって、すべての接続はこの pernode アドレスから始まり、そして戻されるように見えます。

リソース、リソースグループ、リソースタイプ

データサービスは、複数のリソースタイプを利用します。Sun Java System Web Server や Apache Web Server などのアプリケーションは、それらのアプリケーションが依存するネットワークアドレス (論理ホスト名と共有アドレス) を使用します。アプリケーションとネットワークリソースは RGM が管理する基本単位です。

データサービスはリソースタイプです。たとえば、Sun Cluster HA for Oracle のリソースタイプは `SUNW.oracle-server`、Sun Cluster HA for Apache のリソースタイプ `SUNW.apache` です。

リソースはリソースタイプをインスタンス化したもので、クラスタ規模で定義されます。いくつかのリソースタイプはすでに定義されています。

ネットワークリソースは、`SUNW.LogicalHostname` リソースタイプまたは `SUNW.SharedAddress` リソースタイプのどちらかです。これら 2 つのリソースタイプは、Sun Cluster ソフトウェアにより事前に登録されています。

`HAStorage` リソースタイプと `HAStoragePlus` リソースタイプは、リソースとそのリソースが依存するディスクデバイスグループの起動を同期させるのに使用します。これらのリソースタイプによって、クラスタファイルシステムのマウントポイント、広域デバイス、およびデバイスグループ名のパスがデータサービスの起動前に利用可能になることが保証されます。詳細は、『*Sun Cluster 3.1* データサービスのインストールと構成』の「リソースグループとディスクデバイスグループ間の起動の同期」を参照してください。`HAStoragePlus` リソースタイプは Sun Cluster 3.0 5/02 で追加され、ローカルファイルシステムを高可用対応にする新たな機能を備えています。この機能についての詳細は、48 ページの「`HAStoragePlus` リソースタイプ」を参照してください。

RGM が管理するリソースは、1 つのユニットとして管理できるように、「リソースグループ」と呼ばれるグループに配置されます。リソースグループ上でフェイルオーバーまたはスイッチオーバーが開始されると、リソースグループは 1 つのユニットとして移行されます。

注 – アプリケーションリソースが含まれるリソースグループをオンラインにすると、そのアプリケーションが起動します。データサービスの起動メソッドは、アプリケーションが起動され、実行されるのを待ってから、正常に終了します。アプリケーションの起動と実行のタイミングの確認は、データサービスがクライアントにサービスを提供しているかどうかをデータサービスの障害モニターが確認する方法と同じです。このプロセスについての詳細は、『Sun Cluster データサービスの計画と管理 (Solaris OS 版)』を参照してください。

リソースグループマネージャー (RGM)

RGM は、データサービス (アプリケーション) を、リソースタイプの実装によって管理されるリソースとして制御します。これらの実装は、Sun が行う場合もあれば、開発者が汎用データサービステンプレート、データサービス開発ライブラリ API (DSDL API)、またはリソース管理 API (RMAPI) を使用して作成することもあります。クラスタ管理者は、「リソースグループ」と呼ばれる入れ物 (コンテナ) の中でリソースの作成や管理を行います。RGM は、クラスタメンバーシップの変更に応じて、指定ノードのリソースグループを停止および開始します。

RGM は「リソース」と「リソースグループ」に作用します。RGM のアクションによって、リソースやリソースグループの状態はオンラインまたはオフラインに切り替えれらます。リソースとリソースグループに適用できる状態と設定値についての詳細は、74 ページの「リソースおよびリソースグループの状態と設定値」の節を参照してください。

RGM 制御下で Solaris プロジェクトを起動する方法については、76 ページの「データサービスプロジェクトの構成」を参照してください。

リソースおよびリソースグループの状態と設定値

リソースやリソースグループの値は管理者によって静的に設定されるため、これらの設定値を変更するには管理上の作業が必要です。RGM は、動的な「状態」の間でリソースグループを移動させます。これらの設定値と状態については、次のリストを参照してください。

- managed (管理) または unmanaged (非管理) - クラスタ全体に適用されるこの設定値は、リソースグループだけに使用されます。リソースグループは RGM によって管理されます。RGM でリソースグループを管理または非管理にするには、`scrgadm(1M)` コマンドを使用します。これらの設定値は、クラスタ再構成では変更されません。

  新たに作成したリソースグループの状態は非管理になっています。このグループのいずれかのリソースをアクティブにするには、リソースグループの状態が管理になっている必要があります。

  スケーラブル Web サーバーなど、ある種のデータサービスでは、ネットワークリソースの起動前や停止後に、あるアクションを行う必要があります。このアクションには、initialization (INIT) と finish (FINI) データサービスメソッドを使用し

ます。INIT メソッドが動作するためには、リソースが置かれているリソースグループが管理状態になっていなければなりません。

リソースグループを非管理から管理の状態に変更すると、そのグループに対して登録されている INIT メソッドがグループの各リソースに対して実行されます。

リソースグループを管理から非管理の状態に変更すると、登録されている FINI メソッドが呼び出され、クリーンアップが行われます。

一般的に、INIT メソッドおよび FINI メソッドは、スケーラブルサービスのネットワークリソース用です。しかし、これらのメソッドは、アプリケーションが実行しない初期設定やクリーンアップにも使用できます。

- enabled (有効) または disabled (無効) – クラスタ全体に適用されるこの設定値は、リソースだけに使用されます。リソースを有効または無効にするには、`scrgadm(1M)` コマンドを使用します。これらの設定値は、クラスタ再構成では変更されません。

  リソースの通常の設定では、リソースは有効にされ、システムでアクティブに動作しています。

  すべてのクラスタノード上でリソースを使用不能にする必要がある場合は、リソースを無効にします。無効にされたリソースは、一般的な使用には提供されません。

- online (オンライン) または offline (オフライン) – 動的に変更可能なこの状態は、リソースとリソースグループに適用されます。

  オンラインとオフラインの状態は、スイッチオーバーまたはフェイルオーバー中、クラスタ再構成手順に従ったクラスタの遷移とともに変化します。さらに、これらの状態は管理アクションでも変更できます。`scswitch(1M)` コマンドを使用すると、リソースまたはリソースグループのオンラインまたはオフラインの状態を変更できます。

  フェイルオーバーリソースまたはリソースグループを、どの時点でも 1 つのノード上でのみオンラインにすることができます。スケーラブルリソースまたはリソースグループは、いくつかのノードではオンラインにし、他のノードではオフラインにすることができます。スイッチオーバーまたはフェイルオーバー時には、含まれるリソースグループとリソースはあるノードでオフラインになり、その後、別のノードでオンラインになります。

  リソースグループがオフラインの場合、そのすべてのリソースがオフラインです。リソースグループがオンラインの場合、そのすべてのリソースがオンラインです。

  リソースグループはいくつかのリソースを持つことができますが、リソース間には相互依存関係があります。したがって、これらのリソースをオンラインまたはオフラインにするときには、特定の順序で行う必要があります。リソースをオンラインまたはオフラインにするためにメソッドが必要とする時間は、リソースによって異なります。リソースの相互依存関係と起動や停止時間の違いにより、クラスタの再構成では、同じリソースグループのリソースでもオンラインやオフラインの状態が異なることがあります。

リソースとリソースグループプロパティー

Sun Cluster データサービスのリソースやリソースグループのプロパティー値は構成できます。標準的なプロパティーはすべてのデータサービスに共通です。拡張プロパティーは各データサービスに特定のものです。標準プロパティーおよび拡張プロパティーのいくつかは、デフォルト設定によって構成されているため、これらを修正する必要はありません。それ以外のプロパティーは、リソースを作成して構成するプロセスの一部として設定する必要があります。各データサービスのマニュアルでは、設定できるリソースプロパティーの種類とその設定方法を指定しています。

標準プロパティーは、通常特定のデータサービスに依存しないリソースおよびリソースグループプロパティーを構成するために使用されます。標準プロパティーのセットについては、『Sun Cluster データサービスの計画と管理 (Solaris OS 版)』の付録 A「標準プロパティー」を参照してください。

RGM 拡張プロパティーは、アプリケーションバイナリの場所や構成ファイルなどの情報を提供するものです。拡張プロパティーは、データサービスの構成に従って 修正する必要があります。拡張プロパティーについては、データサービスの個別のガイドで説明されています。

データサービスプロジェクトの構成

データサービスは、RGM でオンラインにしたときに Solaris プロジェクト名のもとで起動するように構成できます。そのためには、データサービスを構成するときに、RGM によって管理されるリソースまたはリソースグループと Solaris プロジェクト ID を対応付ける必要があります。リソースまたはリソースグループにプロジェクト ID を対応付けることによって、Solaris オペレーティングシステムの洗練されたコントロールを使用して、クラスタ内の負荷や使用量を管理できます。

注 – この構成を実行するためには、現在のリリースの Sun Cluster ソフトウェアを Solaris 9 以降のオペレーティングシステムで実行する必要があります。

Sun Cluster 環境の Solaris 管理機能を使用すると、ほかのアプリケーションとノードを共有している場合に、最も重要なアプリケーションに高い優先順位を与えることができます。ノードを複数のアプリケーションで共有する例としては、サービスを統合した場合や、アプリケーションのフェイルオーバーが起った場合があります。ここで述べる管理機能を使用すれば、優先順位の低いアプリケーションが CPU 時間などのシステムサプライを過度に使用するのを防止し、重要なアプリケーションの可用性を向上させることができます。

注 – この機能に関連する Solaris のマニュアルでは、CPU 時間、プロセス、タスクなどのコンポーネントを「リソース」と呼んでいます。一方、Sun Cluster のマニュアルでは、RGM の制御下にあるエンティティーを「リソース」と呼んでいます。この節では、RGM の制御下にある Sun Cluster エンティティーを「リソース」と呼びます。また、CPU 時間、プロセス、およびタスクを「サプライ」と呼びます。

この節では、指定した Solaris 9 の `project(4)` でプロセスを起動するように、データサービスを構成する方法の概念について説明します。また、Solaris オペレーティングシステムの管理機能を使用するための、フェイルオーバーのシナリオやヒントについても説明します。

管理機能の概念や手順についての詳細は、『Solaris のシステム管理 (ネットワークサービス)』の第 1 章「ネットワークサービス (概要)」を参照してください。

クラスタ内で Solaris 管理機能を使用できるようにリソースやリソースグループを構成するための手順は次のようになります。

1. アプリケーションをリソースの一部として構成します。
2. リソースをリソースグループの一部として構成します。
3. リソースグループのリソースを有効にします。
4. リソースグループを管理状態にします。
5. リソースグループに対する Solaris プロジェクトを作成します。
6. 手順 5 で作成したプロジェクトにリソースグループ名を対応付けるための標準プロパティーを構成します。
7. リソースグループをオンラインにします。

標準プロパティーの `Resource_project_name` または `RG_project_name` を構成して、リソースまたはリソースグループに Solaris プロジェクト ID を対応付けるには、`scrgadm(1M)` コマンドに `-y` オプションを指定する必要があります。続いて、プロパティーの値にリソースまたはリソースグループを設定します。プロパティーの定義については、『Sun Cluster データサービスの計画と管理 (Solaris OS 版)』の付録 A「標準プロパティー」を参照してください。プロパティーの説明については、`r_properties(5)` と `rg_properties(5)` のマニュアルページを参照してください。

指定するプロジェクト名はプロジェクトデータベース (`/etc/project`) に存在するものでなければなりません。さらに、指定するプロジェクトのメンバーとして root ユーザーが設定されていなければなりません。プロジェクト名データベースの概念については、『Solaris のシステム管理 (Solaris コンテナ : 資源管理と Solaris ゾーン)』の第 2 章「プロジェクトとタスク (概要)」を参照してください。プロジェクトファイルの構文については、`project(4)` を参照してください。

RGM は、リソースまたはリソースグループをオンラインにする際に、関連するプロセスをこのプロジェクト名の下で起動します。

注 – リソースまたはリソースグループとプロジェクトを対応付けることはいつでもできます。ただし、RGM を使ってプロジェクトのリソースやリソースグループをオフラインにしてから再びオンラインに戻すまで、新しいプロジェクト名は有効になりません。

リソースやリソースグループをプロジェクト名の下で起動すれば、次の機能を構成することによってクラスタ全体のシステムサプライを管理できます。

- 拡張アカウンティング – 使用量をタスクやプロセス単位で記録できるため柔軟性が増します。拡張アカウンティングでは、使用状況の履歴を調べ、将来の作業負荷の容量要件を算定できます。
- 制御 – システムサプライの使用を制約する機構を提供します。これにより、プロセス、タスク、およびプロジェクトが特定のシステムサプライを大量に消費することを防止できます。
- フェアシェアスケジューリング (FSS) – それぞれの作業負荷に割り当てる CPU 時間を作業負荷の重要性に基づいて制御できます。作業負荷の重要性は、各作業負荷に割り当てる、CPU 時間のシェア数として表されます。詳細は、次のマニュアルページを参照してください。
  - `dispadmin(1M)`
  - `priocntl(1)`
  - `ps(1)`
  - `FSS(7)`
- プール – アプリケーションの必要性に応じて対話型アプリケーション用に仕切りを使用することができます。プールを使用すれば、サーバーを仕切り分けすることができ、同じサーバーで異なるソフトウェアアプリケーションをサポートできます。プールを使用すると、アプリケーションごとの応答が予測しやすくなります。

プロジェクト構成に応じた要件の決定

Sun Cluster 環境で Solaris が提供する制御を使用するようにデータサービスを構成するには、まず、スイッチオーバーやフェイルオーバー時にリソースをどのように制御および管理するかを決めておく必要があります。新しいプロジェクトを構成する前に、まず、クラスタ内の依存関係を明確にします。たとえば、リソースやリソースグループはディスクデバイスグループに依存しています。

次に、`scrgadm(1M)` で設定されている `nodelist`、 `failback`、`maximum_primaries`、`desired_primaries` リソースグループプロパティーを使って、使用するリソースグループのノードリストの優先順位を確認します。

- リソースグループとディスクデバイスグループ間のノードリストの依存関係の概要については、『Sun Cluster データサービスの計画と管理 (Solaris OS 版)』の「リソースグループとディスクデバイスグループの関係」を参照してください。
- プロパティーについての詳細は、`rg_properties(5)` のマニュアルページを参照してください。

`scrgadm(1M)` や `scsetup(1M)` で設定されている `preferenced` プロパティーと `failback` プロパティーを使用して、ディスクデバイスグループのノードリストの優先順位を確認します。

- `preferenced` プロパティーの概念については、43 ページの「多重ポートディスクデバイスグループ」を参照してください。
- 手順については、“How To Change Disk Device Properties” in 『Sun Cluster のシステム管理 (Solaris OS 版)』の「ディスクデバイスグループの管理」を参照してください。
- ノード構成の概念とフェイルオーバーデータサービスとスケーラブルデータサービスの動作については、21 ページの「Sun Cluster システムのハードウェア/ソフトウェアコンポーネント」を参照してください。

すべてのクラスタノードを同じように構成すると、主ノードと二次ノードに対して同じ使用限度が割り当てられます。各プロジェクトの構成パラメータは、すべてのノードの構成ファイルに定義されているすべてのアプリケーションに対して同じである必要はありません。特定のアプリケーションに対応するすべてのプロジェクトは、少なくとも、そのアプリケーションのすべての潜在的マスターにあるプロジェクトデータベースからアクセス可能である必要があります。たとえば、アプリケーション 1 は *phys-schost-1* によってマスターされているが、*phys-schost-2* や *phys-schost-3* にスイッチオーバーまたはフェイルオーバーされる可能性があるとします。アプリケーション 1 に対応付けられたプロジェクトは、これら 3 つのノード (*phys-schost-1*、*phys-schost-2*、*phys-schost-3*) 上でアクセス可能でなければなりません。

注 – プロジェクトデータベース情報は、ローカルの `/etc/project` データベースファイルに格納することも、NIS マップや LDAP ディレクトリサーバー に格納することもできます。

Solaris オペレーティングシステムでは、使用パラメータは柔軟に構成でき、Sun Cluster によって課せられる制約はほとんどありません。どのような構成を選択するかはサイトの必要性によって異なります。システムの構成を始める前に、次の各項の一般的な指針を参考にしてください。

プロセス当たりの仮想メモリー制限の設定

仮想メモリーの制限をプロセス単位で制御する場合は、`process.max-address-space` コントロールを使用します。`process.max-address-space` 値の設定方法についての詳細は、`rctladm(1M)` のマニュアルページを参照してください。

Sun Cluster で管理コントロールを使用する場合は、アプリケーションの不要なフェイルオーバーが発生したり、アプリケーションの「ピンポン」現象が発生したりするのを防止するために、メモリー制限を適切に設定する必要があります。そのためには、一般に次の点に注意する必要があります。

- メモリー制限をあまり低く設定しない。

  アプリケーションは、そのメモリーが限界に達すると、フェイルオーバーを起こすことがあります。データベースアプリケーションにとってこの指針は特に重要です。その仮想メモリーが限界を超えると予期しない結果になることがあるからです。

- 主ノードと二次ノードに同じメモリー制限を設定しない。

  同じメモリー制限を設定すると、アプリケーションのメモリーが限度に達し、アプリケーションが、同じメモリー制限をもつ二次ノードにフェイルオーバーされたときに「ピンポン」現象を引き起こすおそれがあります。そのため、二次ノードのメモリー制限には、主ノードよりもわずかに大きな値を設定します。異なるメモリー制限を設定することによって「ピンポン」現象の発生を防ぎ、管理者はその間にパラメータを適切に変更することができます。

- 負荷均衡を達成する目的でリソース管理メモリー制限を使用する。

  たとえば、メモリー制限を使用すれば、アプリケーションが誤って過度のスワップ領域を使用することを防止できます。

フェイルオーバーシナリオ

管理パラメータを適切に構成すれば、プロジェクト構成 (`/etc/project`) 内の割り当ては、通常のクラスタ操作でも、スイッチオーバーやフェイルオーバーの状況でも正常に機能します。

以下の各項ではシナリオ例を説明します。

- 最初の「2 つのアプリケーションを供う 2 ノードクラスタ」と「3 つのアプリケーションを供う 2 ノードクラスタ」の項では、すべてのノードが関係するフェイルオーバーシナリオを説明します。
- 「リソースグループだけのフェイルオーバー」の項では、アプリケーションだけのフェイルオーバー操作について説明します。

Sun Cluster 環境では、アプリケーションはリソースの一部として構成します。そして、リソースをリソースグループ (RG) の一部として構成します。障害が発生すると、リソースグループは、対応付けられたアプリケーションとともに、別のノードにフェイルオーバーされます。以下の例では、リソースは明示的に示されていません。各リソースには、1 つのアプリケーションが構成されているものとします。

注 – フェイルオーバーは、RGM に設定されているノードリスト内の優先順位に従って行われます。

以下の例は次のように構成されています。

- アプリケーション 1 (App-1) はリソースグループ RG-1 に構成されています。
- アプリケーション 2 (App-2) はリソースグループ RG-2 に構成されています。

- アプリケーション 3 (App-3) はリソースグループ RG-3 に構成されています。

フェイルオーバーが起こると、各アプリケーションに割り当てられる CPU 時間の割合が変化します。ただし、割り当てられているシェアの数はそのままです。この割合は、そのノードで動作しているアプリケーションの数と、アクティブな各アプリケーションに割り当てられているシェアの数によって異なります。

これらのシナリオでは、次のように構成が行われているものとします。

- すべてのアプリケーションが共通のプロジェクトの下に構成されています。
- 各リソースには 1 つのアプリケーションがあります。
- すべてのノードにおいて、アクティブなプロセスはこれらのアプリケーションだけです。
- プロジェクトデータベースは、クラスタの各ノードで同一に構成されています。

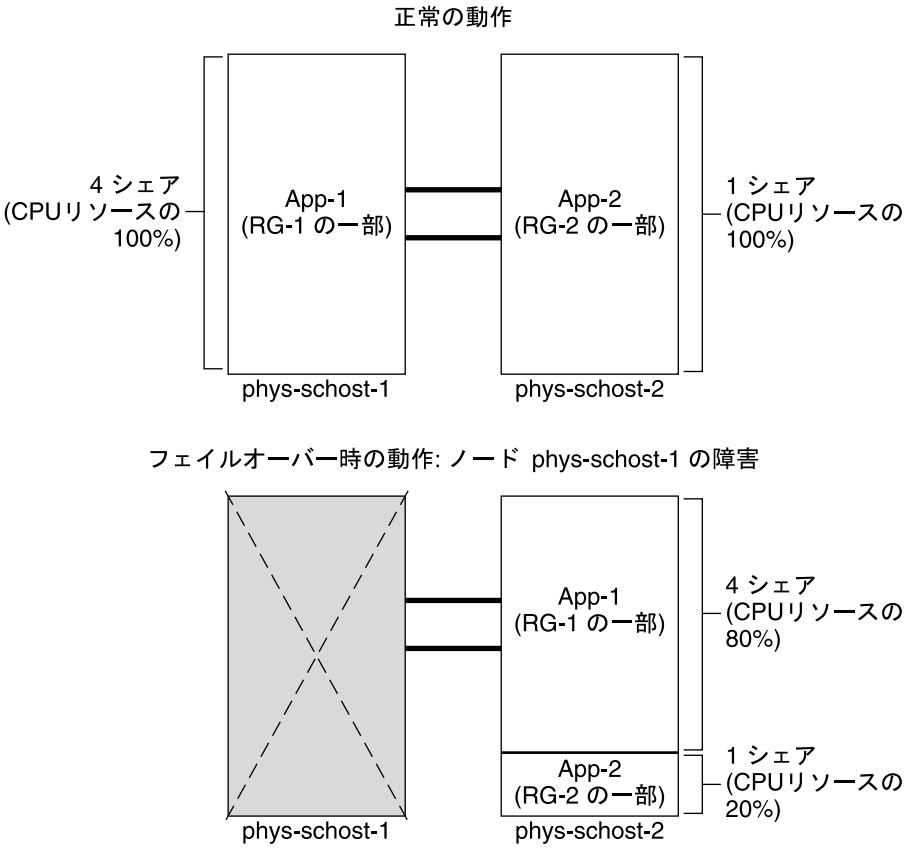
2 つのアプリケーションを供う 2 ノードクラスタ

2 ノードクラスタに 2 つのアプリケーションを構成することによって、それぞれの物理ホスト (*phys-schost-1*、 *phys-schost-2*) を 1 つのアプリケーションのデフォルトマスターにすることができます。一方の物理ホストは、他方の物理ホストの二次ノードになります。アプリケーション 1 とアプリケーション 2 に関連付けられているすべてのプロジェクトは、両ノードのプロジェクトデータベースファイルに存在している必要があります。クラスタが正常に動作している間、各アプリケーションはそれぞれのデフォルトマスターで動作し、管理機能によってすべての CPU 時間を割り当てられます。

フェイルオーバーかスイッチオーバーが起ると、これらのアプリケーションは同じノードで動作し、構成ファイルの設定に従ってシェアを割り当てられます。たとえば、`/etc/project` ファイルに次のエントリが指定されていると、アプリケーション 1 に 4 シェアが、アプリケーション 2 に 1 シェアがそれぞれ割り当てられます。

```
Prj_1:100:project for App-1:root::project.cpu-shares=(privileged,4,none)
Prj_2:101:project for App-2:root::project.cpu-shares=(privileged,1,none)
```

次の図は、この構成の正常時の動作とフェイルオーバー時の動作を表しています。割り当てられているシェアの数は変わりません。ただし、各アプリケーションが利用できる CPU 時間の割合は変わる場合があります。この割合は、CPU 時間を要求する各プロセスに割り当てられているシェア数によって異なります。



3 つのアプリケーションを供う 2 ノードクラスタ

3 つのアプリケーションが動作する 2 ノードクラスタでは、1 つの物理ホスト (*phys-schost-1*) を 1 つのアプリケーションのデフォルトマスターとして構成できます。そして、もう 1 つの物理ホスト (*phys-schost-2*) をほかの 2 つのアプリケーションのデフォルトマスターとして構成できます。各ノードには、次のサンプルプロジェクトデータベースファイルがあるものとします。フェイルオーバーやスイッチオーバーが起っても、プロジェクトデータベースファイルが変更されることはありません。

```
Prj_1:103:project for App-1:root::project.cpu-shares=(privileged,5,none)
Prj_2:104:project for App_2:root::project.cpu-shares=(privileged,3,none)
Prj_3:105:project for App_3:root::project.cpu-shares=(privileged,2,none)
```

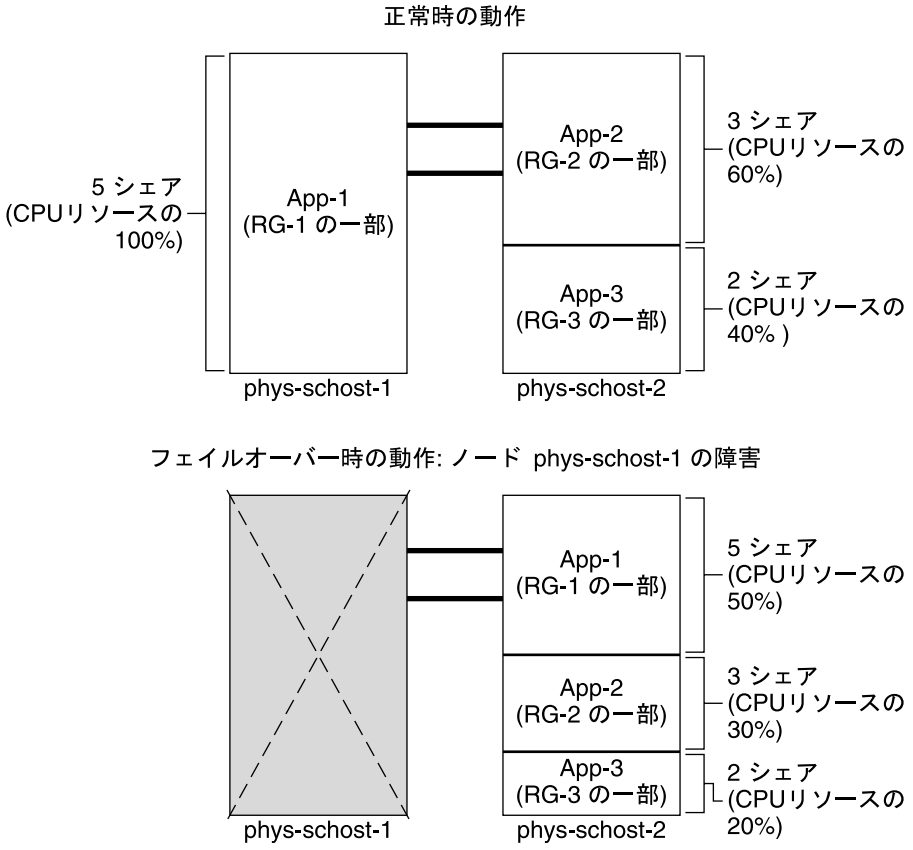
クラスタが正常に動作している間、アプリケーション 1 には、そのデフォルトマスター *phys-schost-1* で 5 シェアが割り当てられます。このノードで CPU 時間を要求するアプリケーションはこのアプリケーションだけであるため、この数は 100 パーセントの CPU 時間と同じことです。アプリケーション 2 と 3 には、それぞれのデフォル

トマスターである *phys-schost-2* で 3 シェアと 2 シェアが割り当てられます。したがって、正常な動作では、アプリケーション 2 に CPU 時間の 60 パーセントが、アプリケーション 3 に CPU 時間の 40 パーセントがそれぞれ割り当てられます。

フェイルオーバーかスイッチオーバーが発生し、アプリケーション 1 が *phys-schost-2* に切り替えられても、3 つのアプリケーションの各シェアは変わりません。ただし、割り当てられる CPU リソースの割合はプロジェクトデータベースファイルに従って変更されます。

- 5 シェアをもつアプリケーション 1 には CPU の 50 パーセントが割り当てられます。
- 3 シェアをもつアプリケーション 2 には CPU の 30 パーセントが割り当てられます。
- 2 シェアをもつアプリケーション 3 には CPU の 20 パーセントが割り当てられます。

次の図は、この構成の正常な動作とフェイルオーバー動作を示しています。



リソースグループだけのフェイルオーバー

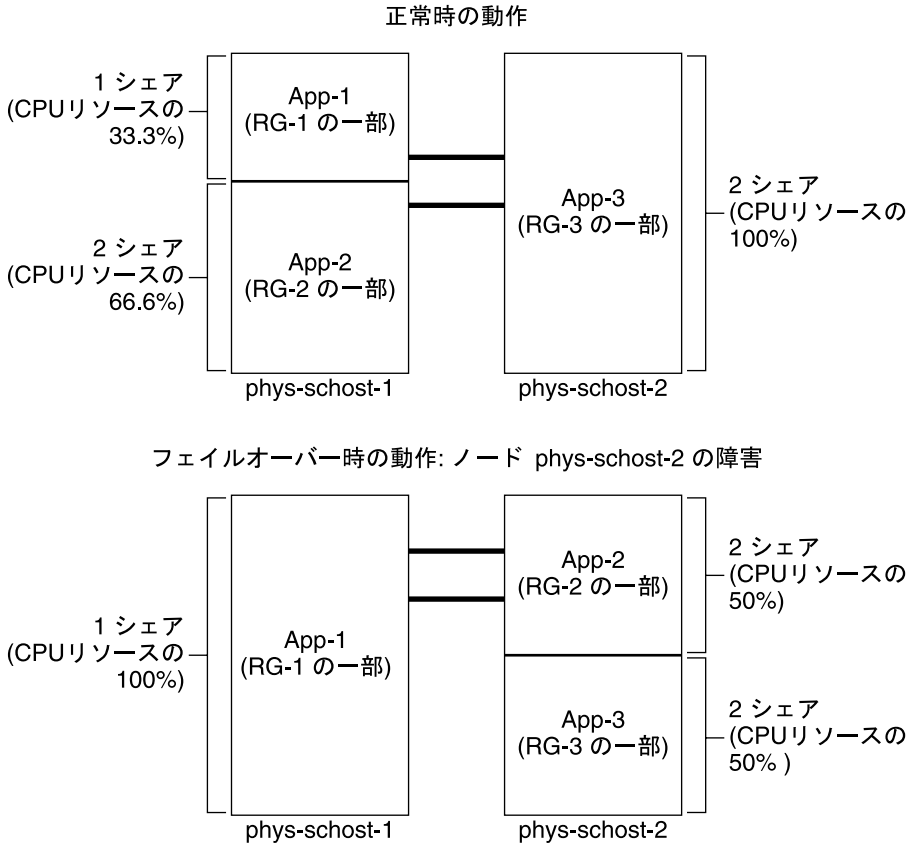
複数のリソースグループが同じデフォルトマスターに属している構成では、1 つのリソースグループ (および、それに関連付けられたアプリケーション) が 二次ノードにフェイルオーバーされたり、スイッチオーバーされることがあります。その間、クラスタのデフォルトマスターは動作を続けます。

注 – フェイルオーバーの際、フェイルオーバーされるアプリケーションには、二次ノード上の構成ファイルの指定に従ってリソースが割り当てられます。この例の場合、主ノードと二次ノードのプロジェクトデータベースファイルの構成は同じです。

次のサンプル構成ファイルでは、アプリケーション 1 に 1 シェア、アプリケーション 2 に 2 シェア、アプリケーション 3 に 2 シェアがそれぞれ割り当てられています。

```
Prj_1:106:project for App_1:root::project.cpu-shares=(privileged,1,none)
Prj_2:107:project for App_2:root::project.cpu-shares=(privileged,2,none)
Prj_3:108:project for App_3:root::project.cpu-shares=(privileged,2,none)
```

以下の図は、この構成の正常時の動作とフェイルオーバー時の動作を表しています。ここでは、アプリケーション 2 が動作する RG-2 が *phys-schost-2* にフェイルオーバーされます。割り当てられているシェアの数は変わりません。ただし、各アプリケーションが利用できる CPU 時間の割合は、CPU 時間を要求する各アプリケーションに割り当てられているシェア数によって異なります。



パブリックネットワークアダプタと IP ネットワークマルチパス

クライアントは、パブリックネットワークを介してクラスタにデータ要求を行います。各クラスタノードは、1 対のパブリックネットワークアダプタを介して少なくとも 1 つのパブリックネットワークに接続されています。

Sun Cluster で動作する Solaris インターネットプロトコル (IP) ソフトウェアは、パブリックネットワークアダプタを監視したり、障害を検出したときに IP アドレスをあるアダプタから別のアダプタにフェイルオーバーしたりする基本的な機構を提供します。各クラスタノードは独自の IP ネットワークマルチパス 構成を持っており、この構成がほかのクラスタノードの構成と異なる場合があります。

パブリックネットワークアダプタは、IP マルチパスグループ (「マルチパスグループ」) として編成されます。各マルチパスグループには、1 つまたは複数のパブリックネットワークアダプタがあります。マルチパスグループの各アダプタはアクティブにしておいてもかまいません。あるいは、スタンバイインタフェースを構成し、フェイルオーバーが起こるまでそれらを非アクティブにしておいてもかまいません。

in.mpathd マルチパスデーモンは、テスト IP アドレスを使って障害や修復を検出します。マルチパスデーモンによってアダプタの 1 つに障害が発生したことが検出されると、フェイルオーバーが行われます。すべてのネットワークアクセスは、障害のあるアダプタからマルチパスグループの別の正常なアダプタにフェイルオーバーされます。したがって、デーモンがそのノードのパブリックネットワーク接続を維持します。スタンバイインタフェースを構成していた場合、このデーモンはスタンバイインタフェースを選択します。そうでない場合、このデーモンは最も小さい IP アドレス番号を持つインタフェースを選択します。フェイルオーバーはアダプタインタフェースレベルで発生するため、これよりも高いレベルの接続 (TCP など) は影響を受けません。ただし、フェイルオーバー中には一時的にわずかな遅延が発生します。IP アドレスのフェイルオーバーが正常に終了すると、ARP ブロードキャストが送信されます。したがって、デーモンがリモートクライアントへの接続を維持します。

注 – TCP の輻輳回復特性のために、正常なフェイルオーバーのあと、TCP エンドポイントではさらに遅延が生じる可能性があります。これは、フェイルオーバー中にいくつかのセグメントが失われて、TCP の輻輳制御機構がアクティブになるためです。

マルチパスグループには、論理ホスト名と共有アドレスリソースの構築ブロックがあります。論理ホスト名と共有アドレスリソースとは別にマルチパスグループを作成して、クラスタノードのパブリックネットワーク接続を監視する必要もあります。つまり、ノード上の同じマルチパスグループは、任意の数の論理ホスト名または共有アドレスリソースをホストできます。論理ホスト名と共有アドレスリソースについての詳細は、『Sun Cluster データサービスの計画と管理 (Solaris OS 版)』を参照してください。

注 – IP ネットワークマルチパス 機構の設計は、アダプタの障害を検出してマスクすることを目的としています。この設計は、管理者が `ifconfig(1M)` を使用して論理 (または共有) IP アドレスのどれかを削除した状態から回復することを目的としているわけではありません。Sun Cluster ソフトウェアから見ると、論理アドレスや共有 IP アドレスは RGM によって管理されるリソースです。管理者が IP アドレスを追加または削除する場合、正しくは、`scrgadm(1M)` を使用してリソースを含むリソースグループを修正します。

IP ネットワークマルチパスの Solaris の実装についての詳細は、クラスタにインストールされている Solaris オペレーティングシステムのマニュアルを参照してください。

オペレーティングシステムのリリース	参照先
Solaris 8 オペレーティングシステム	『IP ネットワークマルチパスの管理』
Solaris 9 オペレーティングシステム	『IP ネットワークマルチパスの管理』の第 1 章「IP ネットワークマルチパス (概要)」
Solaris 10 オペレーティングシステム	『Solaris のシステム管理 (IP サービス)』のパート VI「IPMP」

SPARC: 動的再構成のサポート

Sun Cluster 3.1 8/05 による動的再構成 (DR: Dynamic Reconfiguration) ソフトウェア機能のサポートは段階的に開発されています。この節では、Sun Cluster 3.1 8/05 による DR 機能のサポートの概念と考慮事項について説明します。

Solaris の DR 機能の説明で述べられているすべての必要条件、手順、制限は Sun Cluster の DR サポートにも適用されます (オペレーティング環境での休止状態中を除く)。したがって、Sun Cluster ソフトウェアで DR 機能を使用する前には、必ず、Solaris の DR 機能についての説明を参照してください。特に、DR の切り離し 操作中に、ネットワークに接続されていない入出力デバイスに影響する問題について確認してください。

『Sun Enterprise 10000 Dynamic Reconfiguration User Guide』と『Sun Enterprise 10000 Dynamic Reconfiguration Reference Manual』 (Solaris 8 on Sun Hardware コレクションまたは Solaris 9 on Sun Hardware コレクション) は両方とも http://docs.sun.com からダウンロードできます。

SPARC: 動的再構成の概要

DR 機能を使用すると、システムハードウェアの切り離しなどの操作をシステムの稼動中に行うことができます。DR プロセスの目的は、システムを停止したり、クラスタの可用性を中断したりせずにシステム操作を継続できるようにすることです。

DR はボードレベルで機能します。したがって、DR 操作はボード上のすべてのコンポーネントに影響します。ボードには、CPU やメモリー、ディスクドライブやテープドライブ、ネットワーク接続の周辺機器インタフェースなど、複数のコンポーネントが取り付けられています。

アクティブなコンポーネントを含むボードを切り離すと、システムエラーになります。DR サブシステムは、ボードを切り離す前に、他のサブシステム (Sun Cluster など) に問い合わせてボード上のコンポーネントが使用されているかを判別します。ボードが使用中であることがわかると、DR のボード切り離し操作は行われません。つまり、アクティブなコンポーネントを含むボードに DR のボード切り離し操作を発行しても、DR サブシステムがその操作を拒否するため、DR のボード切り離し操作はいつ発行しても安全です。

同様に、DR のボード追加操作も常に安全です。新たに追加されたボードの CPU とメモリーは、システムによって自動的にサービス状態になります。ただし、そのボードのほかのコンポーネントを意図的に使用するには、管理者がそのクラスタを手動で構成する必要があります。

注 – DR サブシステムにはいくつかのレベルがあります。下位のレベルがエラーを報告すると、上位のレベルもエラーを報告します。ただし、下位のレベルが具体的なエラーを報告しても、上位のレベルは「`Unknown error`」を報告します。このエラーは無視してもかまいません。

次の各項では、デバイスタイプごとに DR の注意事項を説明します。

SPARC: CPU デバイスに対する DR クラスタリング

CPU デバイスが存在していても、Sun Cluster ソフトウェアは DR のボード切り離し操作を拒否しません。

DR のボード追加操作が正常に終わると、追加されたボードの CPU デバイスは自動的にシステム操作に組み込まれます。

SPARC: メモリーに対する DR クラスタリング

DR では、次の 2 種類のメモリーを考慮してください。

- カーネルメモリーケージ
- カーネル以外のメモリーケージ

これらの違いはその使用方法だけであり、実際のハードウェアは同じものです。カーネルメモリーケージとは、Solaris オペレーティングシステムが使用するメモリーのことです。Sun Cluster ソフトウェアは、カーネルメモリーケージを含むボードに対するボード切り離し操作をサポートしていないため、このような操作を拒否します。DR のボード切り離し操作がカーネルメモリーケージ以外のメモリーに関連するものである場合、Sun Cluster はこの操作を拒否しません。メモリーに関連する DR のボード追加操作が正常に終わると、追加されたボードのメモリーは自動的にシステム操作に組み込まれます。

SPARC: ディスクドライブとテープドライブに対する DR クラスタリング

Sun Cluster は、主ノードのアクティブなドライブに対する DR のボード切り離し操作を拒否します。DR のボード切り離し操作を実行できるのは、主ノードのアクティブでないドライブと、二次ノードの任意のドライブだけです。DR 操作が終了すると、クラスタのデータアクセスが前と同じように続けられます。

注 – Sun Cluster は、定足数デバイスの使用に影響を与える DR 操作を拒否します。定足数デバイスの考慮事項と、定足数デバイスに対する DR 操作の実行手順については、89 ページの「SPARC: 定足数デバイスに対する DR クラスタリング」を参照してください。

これらの操作の詳細な実行手順については、『Sun Cluster のシステム管理 (Solaris OS 版)』の「定足数デバイスへの動的再構成」を参照してください。

SPARC: 定足数デバイスに対する DR クラスタリング

DR のボード切り離し操作が、定足数デバイスとして構成されているデバイスへのインタフェースを含むボードに関連する場合、Sun Cluster ソフトウェアははこの操作を拒否します。Sun Cluster ソフトウェアはまた、この操作によって影響を受ける定足数デバイスを特定します。定足数デバイスとしてのデバイスに対して DR のボード切り離し操作を行う場合は、まずそのデバイスを無効にする必要があります。

定足数の詳細な管理手順については、『Sun Cluster のシステム管理 (Solaris OS 版)』の第 5 章「定足数の管理」を参照してください。

SPARC: クラスタインターコネクトインタフェースに対する DR クラスタリング

DR のボード切り離し操作が、アクティブなクラスタインターコネクトインタフェースを含むボードに関連する場合、Sun Cluster ソフトウェアはこの操作を拒否します。Sun Cluster ソフトウェアはまた、この操作によって影響を受けるインタフェースを特定します。DR 操作を成功させるためには、Sun Cluster 管理ツールを使用して、アクティブなインタフェースを無効にしておく必要があります。

注意 – Sun Cluster ソフトウェアでは、各クラスタノードは、ほかのすべてのクラスタノードへの有効なパスを、少なくとも1 つ、持っておく必要があります。したがって、個々のクラスタノードへの最後のパスをサポートするプライベートインターコネクトインタフェースを無効にしないでください。

これらの操作の詳細な実行方法については、『Sun Cluster のシステム管理 (Solaris OS 版)』の「クラスタインターコネクトの管理」を参照してください。

SPARC: パブリックネットワークインタフェースに対する DR クラスタリング

DR のボード切り離し操作が、アクティブなパブリックネットワークインタフェースを含むボードに関連する場合、Sun Cluster ソフトウェアはこの操作を拒否します。Sun Cluster ソフトウェアはまた、この操作によって影響を受けるインタフェースを特定します。アクティブなネットワークインタフェースが存在するボードを切り離す前に、まず、`if_mpadm(1M)` コマンドを使って、そのインタフェース上のすべてのトラフィックを、同じマルチパスグループの正常なほかのインタフェースに切り替える必要があります。

注意 – 無効にしたネットワークアダプタに対する DR 切り離し操作中に、残りのネットワークアダプタで障害が発生すると、可用性が影響を受けます。これは、DR 操作の間は、残りのネットワークアダプタのフェイルオーバー先が存在しないためです。

パブリックネットワークインタフェースに対する DR 切り離し操作の詳細な実行手順については、『Sun Cluster のシステム管理 (Solaris OS 版)』の「パブリックネットワークの管理」を参照してください。

第 4 章

よくある質問

この章では、Sun Cluster システムに関して最も頻繁に寄せられる質問に対する回答を示します。回答は、トピックごとに構成されています。

高可用性に関する FAQ

質問: 可用性の高いシステムとは何ですか。

回答: Sun Cluster システムでは、高可用性 (HA) を、クラスタがアプリケーションを実行し続けることができる能力であると定義しています。通常ならばサーバーシステムが使用できなくなるような障害が発生しても、高可用性アプリケーションは動作し続けます。

質問: クラスタが高可用性を提供するプロセスは何ですか。

回答: クラスタフレームワークは、フェイルオーバーとして知られるプロセスによって可用性の高い環境を提供します。フェイルオーバーとは、障害の発生したノードからクラスタ内の別の動作可能ノードにデータサービスリソースを移行するために、クラスタによって実行される一連のステップです。

質問: フェイルオーバーデータサービスとスケーラブルデータサービスの違いは何ですか。

回答: 高可用性データサービスには、次の 2 つの種類があります。

- フェイルオーバー
- スケーラブル

フェイルオーバーデータサービスとは、アプリケーションが一度に 1 つのクラスタ内の主ノードだけで実行されることを示します。他のノードは他のアプリケーションを実行できますが、各アプリケーションは単一のノードでのみ実行されます。主ノードに障害が発生すると、障害が発生したノードで実行されていたアプリケーションは別のノードに引き継がれて (フェイルオーバーされて)、実行を続けます。

スケーラブルサービスは、アプリケーションを複数のノードに広げて、単一の論理サービスを作成します。スケーラブルサービスは、実行されるクラスタ全体のノードとプロセッサの数を強化します。

クラスタへの物理インタフェースは、アプリケーションごとに 1 つのノードに設定されます。このノードを広域インタフェース (GIF) ノードといいます。クラスタには、複数の GIF ノードが存在することがあります。個々の GIF には、スケーラブルサービスから使用する 1 つまたは複数の論理インタフェースがあります。この論理インタフェースを「広域インタフェース」と呼びます。GIF ノードは、特定のアプリケーションに対するすべての要求を広域インタフェースを介して受け取り、それらを、そのアプリケーションサーバーが動作している複数のノードに振り分けます。GIF ノード に障害が発生すると、広域インタフェースは別のノードにフェイルオーバーされます。

アプリケーションが実行されているノードに障害が発生すると、アプリケーションは別のノードで実行を続けますが、障害が発生したノードがクラスタに戻るまで多少のパフォーマンス低下が生じます。このプロセスは、障害が発生したノードがクラスタに戻るまで続けられます。

ファイルシステムに関する FAQ

質問: 1 つまたは複数のクラスタノードを高可用性 NFS サーバーとして実行し、ほかのクラスタノードをクライアントとして実行できますか。

回答: 実行できません。ループバックマウントは行わないでください。

質問: リソースグループマネージャー の制御下にないアプリケーションにクラスタファイルシステムを使用できますか。

回答: 使用できます。ただし、RGM の制御下にないと、そのアプリケーションが実行されているノードに障害があった場合、そのアプリケーションを手動で再起動する必要があります。

質問: クラスタファイルシステムは、必ず、`/global` ディレクトリの下にマウントポイントが必要ですか。

回答: いいえ。ただし、クラスタファイルシステムを `/global` などの同一のマウントポイントのもとに置くと、これらのファイルシステムの構成と管理が簡単になります。

質問: クラスタファイルシステムを使用した場合と NFS ファイルシステムをエクスポートした場合の違いは何ですか。

回答: 次のように、いくつかの違いがあります。

1. クラスタファイルシステムは広域デバイスをサポートします。NFS は、デバイスへの遠隔アクセスをサポートしません。
2. クラスタファイルシステムには広域名前空間があります。したがって、必要なのは 1 つのマウントコマンドだけです。これに対し、NFS では、ファイルシステムを各ノードにマウントする必要があります。

3. クラスタファイルシステムは、NFS よりも多くの場合でファイルをキャッシュします。たとえば、複数のノードからファイルにアクセスしている場合 (たとえば、読み取り、書き込み、ファイルロック、非同期入出力などのために)、クラスタファイルシステムはファイルをキャッシュします。
4. クラスタファイルシステムは、リモート DMA とゼロコピー機能を提供する、将来の高速クラスタインターコネクトを利用するよう作られています。
5. クラスタファイルシステムのファイルの属性を (`chmod(1M)` などを使用して) 変更すると、変更内容はすべてのノードでただちに反映されます。エクスポートされた NFS ファイルシステムでは、この処理に時間がかかる場合があります。

質問: 私のクラスタノードには、`/global/.devices/node@`*nodeID* というファイルシステムがあります。このファイルシステムにデータを格納すると、これらのデータは高可用性および広域になりますか。

回答: 広域デバイス名前空間が格納されているこれらのファイルシステムは、一般的な使用を目的としたものではありません。これらのファイルシステムは広域的な属性を持っていますが、広域的にアクセスされることはありません。つまり、各ノードは、自身の広域デバイス名前空間にしかアクセスしません。あるノードが停止しても、他のノードがこのノードに代わってこの名前空間にアクセスすることはできません。これらのファイルシステムは、高可用性を備えてはいません。したがって、高可用性や広域属性を与えたいデータをこれらのファイルシステムに格納すべきではありません。

ボリューム管理に関する FAQ

質問: すべてのディスクデバイスをミラー化する必要がありますか。

回答: ディスクデバイスの可用性を高くするには、それをミラー化するか、RAID-5 ハードウェアを使用する必要があります。すべてのデータサービスは、可用性の高いディスクデバイスか、可用性の高いディスクデバイスにマウントされたクラスタファイルシステムのどちらかを使用する必要があります。このような構成にすることで、単一のディスク障害に耐えることができます。

質問: ローカルディスク (起動ディスク) に対してあるボリュームマネージャーを使用し、多重ホストディスクに対して別のボリュームマネージャーを使用することはできますか。

回答: SPARC: この構成をサポートするには、Solaris ボリュームマネージャー ソフトウェアでローカルディスクを管理し、VERITAS Volume Manager で多重ホストディスクを管理する必要があります。これ以外の組み合わせではサポートされません。

x86: x86 ベースのクラスタでサポートされるのは Solaris ボリュームマネージャー だけであるため、この構成はサポートされません。

データサービスに関する FAQ

質問: 利用可能な Sun Cluster データサービスは何ですか。

回答: サポートされているデータサービスのリストについては、『Sun Cluster 3.1 4/05 Release Notes for Solaris OS』の「Supported Products」を参照してください。

質問: Sun Cluster データサービスによってサポートされているアプリケーションのバージョンは何ですか。

回答: サポートされているアプリケーションのバージョンのリストについては、『Sun Cluster 3.1 4/05 Release Notes for Solaris OS』の「Supported Products」を参照してください。

質問: 独自のデータサービスを作成できますか。

回答: 作成できます。詳細は、『Sun Cluster データサービス開発ガイド (Solaris OS 版)』の第 11 章「DSDL API 関数」を参照してください。

質問: ネットワークリソースを作成する場合、IP アドレスで指定するのですか。それともホスト名で指定するのですか。

回答: ネットワークリソースを指定する場合には、IP アドレスではなく、UNIX のホスト名を使用することを推奨します。

質問: ネットワークリソースを作成する場合に、論理ホスト名 (LogicalHostname リソース) または共有アドレス (SharedAddress リソース) を使用した場合の違いは何ですか。

回答: Sun Cluster HA for NFS の場合を除き、`Failover` モードリソースグループの `LogicalHostname` リソースを使用するようにマニュアルが推奨している場合、`SharedAddress` リソースと `LogicalHostname` リソースは同様に使用できます。`SharedAddress` リソースを使用すると、余分なオーバーヘッドが発生します。クラスタネットワークソフトウェアは `SharedAddress` 用には構成されていますが、`LogicalHostname` 用には構成されていません。

`SharedAddress` リソースを使用する利点は、スケーラブルデータサービスとフェイルオーバーデータサービスを両方構成して、クライアントが同じホスト名で両方のサービスにアクセスするときに分かります。この場合、`SharedAddress` リソースは、フェイルオーバーアプリケーションリソースとともに、1 つのリソースグループに格納されます。スケーラブルサービスリソースは、異なるリソースグループに格納され、 `SharedAddress` リソースを使用するように構成されます。次に、スケーラブルサービスとフェイルオーバーサービスは両方とも、`SharedAddress` リソースに構成されている同じホスト名とアドレスのセットを使用します。

パブリックネットワークに関する FAQ

質問: Sun Cluster システムがサポートするパブリックネットワークアダプタは何ですか。

回答: 現在、Sun Cluster システムは、Ethernet (10/100BASE-T および 1000BASE-SX Gb) パブリックネットワークアダプタをサポートしています。今後新しいインタフェースがサポートされる可能性があるため、最新情報については、ご購入先に確認してください。

質問: フェイルオーバーでの MAC アドレスの役割は何ですか。

回答: フェイルオーバーが発生すると、新しいアドレス解決プロトコル (ARP) パケットが生成されて伝送されます。これらの ARP パケットには、新しい MAC アドレス (ノードの処理が続行される新しい物理アダプタのアドレス) と古い IP アドレスが含まれます。ネットワーク上の別のマシンがこれらのパケットの 1 つを受信した場合は、そのマシンは自身の ARP キャッシュから古い MAC-IP マッピングをフラッシングして、新しいマッピングを使用します。

質問: Sun Cluster システムは `local-mac-address?=true` という設定をサポートしますか。

回答: サポートします。実際、IP ネットワークマルチパスでは `local-mac-address?` を `true` に設定する必要があります。

`local-mac-address?` を設定するには、SPARC ベースのクラスタでは OpenBoot PROM の `ok` プロンプトから `eeprom(1M)` を使用します。x86 ベースのクラスタでは、BIOS のブート後に SCSI ユーティリティーを起動して設定します。

質問: IP ネットワークマルチパス がアダプタのスイッチオーバーを実行するとき、どれくらいの遅延がありますか。

回答: この遅延は数分に及ぶことがあります。これは、IP ネットワークマルチパス スイッチオーバーが実行されるときに、余分な ARP が送信されるためです。ただし、クライアントとクラスタ間のルーターは、必ずしもこの余分な ARP を使用するわけではありません。したがって、ルーター上のこの IP アドレスに対応する ARP キャッシュがタイムアウトするまでは、エントリが古い MAC アドレスを使用してしまう可能性があります。

質問: ネットワークアダプタの障害の検出にはどの程度の時間が必要ですか。

回答: デフォルトの障害検出時間は 10 秒です。アルゴリズムは障害をこの時間内に検出しようとしますが、実際の時間はネットワークの負荷によって異なります。

クラスタメンバーに関する FAQ

質問: すべてのクラスタメンバーが同じ root パスワードを持つ必要がありますか。

回答: 各クラスタメンバーに同じ root パスワードを設定する必要はありません。ただし、同じ root パスワードをすべてのノードに使用すると、クラスタの管理を簡略化できます。

質問: ノードが起動される順序は重要ですか。

回答: ほとんどの場合、重要ではありません。しかし、起動順序は amnesia を防ぐために重要です。たとえば、ノード 2 が定足数デバイスの所有者であり、ノード 1 が停止してノード 2 を停止させた場合は、ノード 2 を起動してからノード 1 を起動する必要があります。この順序によって、古いクラスタ構成情報を持つノードを誤って起動するのを防ぐことができます。amnesia についての詳細は、54 ページの「障害による影響の防止について」を参照してください。

質問: クラスタノードのローカルディスクをミラー化する必要がありますか。

回答: 必要があります。このミラー化は必要条件ではありませんが、クラスタノードのディスクをミラー化すると、ノードを停止させる非ミラー化ディスクの障害を防止できます。ただし、クラスタノードのローカルディスクをミラー化すると、システム管理の負荷が増えます。

質問: クラスタメンバーのバックアップの注意点は何ですか。

回答: クラスタには、いくつかのバックアップ方式を使用できます。1 つの方法としては、テープドライブまたはライブラリが接続された 1 つのノードをバックアップノードとして設定します。さらに、クラスタファイルシステムを使用してデータをバックアップします。このノードは共有ディスクには接続しないでください。

データのバックアップと復元方法についての詳細は、『Sun Cluster のシステム管理 (Solaris OS 版)』の第 9 章「クラスタのバックアップと復元」を参照してください。

質問: ノードが、二次ノードとして使用できる状態にあるのはいつですか。

回答: Solaris 8 と Solaris 9:

再起動後にノードがログインプロンプトを表示しているときです。

Solaris 10:

`multi-user-server` マイルストーンが動作している場合、ノードは二次ノードとして使用できる状態にあります。

```
# svcs -a | grep multi-user-server:default
```

クラスタ記憶装置に関する FAQ

質問: 多重ホスト記憶装置の可用性を高めるものは何ですか。

回答: 多重ホスト記憶装置は、ミラー化 (またはハードウェアベースの RAID-5 コントローラ) によって、単一のディスクが失われても存続できるという点で高可用性です。多重ホスト記憶装置には複数のホスト接続があるため、接続先の単一ノードが失われても耐えることができます。さらに、各ノードから、接続されている記憶装置への冗長パスは、ホストバスアダプタやケーブル、ディスクコントローラの障害に対する備えとなります。

クラスタインターコネクトに関する FAQ

質問: Sun Cluster システムがサポートするクラスタインターコネクトは何ですか。

回答: 現在のところ、Sun Cluster システムは次のクラスタインターコネクトをサポートします。

- Ethernet (100BASE-T Fast Ethernet と 1000BASE-SX Gb)。SPARC ベースのクラスタと x86 ベースのクラスタの両方。
- Infiniband。SPARC ベースのクラスタと x86 ベースのクラスタの両方。
- SCI。SPARC ベースのクラスタのみ。

質問: 「ケーブル」とトランスポート「パス」の違いは何ですか。

回答: クラスタトランスポートケーブルは、トランスポートアダプタとスイッチを使用して構成されます。ケーブルは、アダプタやスイッチをコンポーネント対コンポーネントとして結合します。クラスタトポロジマネージャーは、利用可能なケーブルを使用し、ノード間にエンドツーエンドのトランスポートパスを構築します。ただし、ケーブルとトランスポートパスが 1 対 1 で対応しているわけではありません。

ケーブルは、管理者によって静的に「有効」または「無効」にされます。ケーブルには、「状態」(有効または無効) はありますが、「ステータス」はありません。無効になっているケーブルは、構成されていないのと同じことです。無効なケーブルをトランスポートパスとして使用することはできません。ケーブルは検査できないため、その状態は不明です。ケーブルの状態を取得するには、`scconf -p` コマンドを使用します。

トランスポートパスは、クラスタトポロジマネージャーによって動的に確立されます。トランスポートパスの「ステータス」はトポロジマネージャーによって決められますが、パスは「オンライン」または「オフライン」のステータスを持つことができます。トランスポートパスのステータスを取得するには、`scstat(1M)` コマンドを使用します。

次のような 2 ノードクラスタがあるとします。これには、4 つのケーブルが使用されています。

```
node1:adapter0      to switch1, port0
node1:adapter1      to switch2, port0
node2:adapter0      to switch1, port1
node2:adapter1      to switch2, port1
```

これらの 4 つのケーブルを使用して設定できるトランスポートパスには、次の 2 つがあります。

```
node1:adapter0    to node2:adapter0
node2:adapter1    to node2:adapter1
```

クライアントシステムに関する FAQ

質問: クラスタでの使用における特殊なクライアントの要求や制約について考慮する必要がありますか。

回答: クライアントシステムは、ほかのサーバーに接続する場合と同様にクラスタに接続します。データサービスアプリケーションによっては、クライアント側ソフトウェアをインストールするか、別の構成変更を行なって、クライアントがデータサービスアプリケーションに接続できるようにしなければならないこともあります。クライアント側の構成要件についての詳細は、『Sun Cluster データサービスの計画と管理 (Solaris OS 版)』の第 1 章「Sun Cluster データサービスの計画」を参照してください。

管理コンソールに関する FAQ

質問: Sun Cluster システムには管理コンソールが必要ですか。

回答: 必要です。

質問: 管理コンソールをクラスタ専用にする必要がありますか、または別の作業に使用することができますか。

回答: Sun Cluster システムでは専用の管理コンソールは必要ありませんが、専用の管理コンソールを使用すると、次のような利点があります。

- コンソールと管理ツールを同じマシンにまとめることで、クラスタ管理を一元化できます。
- ハードウェアサービスプロバイダによる問題解決が迅速に行われます。

質問: 管理コンソールはクラスタの近く (たとえば同じ部屋) に配置する必要がありますか。

回答: ハードウェアの保守担当者に確認してください。プロバイダによっては、コンソールをクラスタの近くに置くことを要求するところもあります。コンソールを同じ部屋に配置する必要性は、技術的にはありません。

質問: 距離の条件をすべて満たしている場合、1 台の管理コンソールが複数のクラスタにサービスを提供できますか。

回答: 提供できます。複数のクラスタを1 台の管理コンソールから制御できます。また、1 台の端末集配信装置 (コンセントレータ) をクラスタ間で共有することもできます。

端末集配信装置とシステムサービスプロセッサに関する FAQ

質問: Sun Cluster システムは端末集配信装置を必要としますか。

回答: Sun Cluster 3.0 以降のソフトウェアリリースでは、端末集配信装置を実行する必要はありません。障害による影響防止に端末集配信装置を必要とした Sun Cluster 2.2 とは異なり、以降の製品では端末集配信装置に依存しません。

質問: ほとんどの Sun Cluster サーバーは端末集配信装置を使用していますが、Sun Enterprise E10000 サーバーが使用していないのはなぜですか。どうすればよいでしょうか。

回答: 端末集配信装置は、ほとんどのサーバーで効率的なシリアル - Ethernet コンバータです。端末集配信装置のコンソールポートはシリアルポートです。Sun Enterprise E1000 サーバーはシリアルポートを持っていません。システムサービスプロセッサ (SSP) は Ethernet または `jtag` ポートを介したコンソールです。Sun Enterprise E1000 サーバーの場合、コンソールには常に SSP を使用します。

質問: 端末集配信装置を使用する場合の利点は何ですか。

回答: 端末集配信装置を使用すると、コンソールレベルのアクセス権が各ノードに提供され、ネットワーク上の任意の場所にあるリモートワークステーションから各ノードにアクセスできます。このアクセス権は、そのノードが SPARC ベースのノード上にある OpenBoot PROM (OBP) である場合でも、x86 ベースのノード上にある起動サブシステムである場合でも提供されます。

質問: Sun がサポートしていない端末集配信装置を使用する場合に注意する点は何ですか。

回答: Sun がサポートする端末集配信装置とほかのコンソールデバイスの主な違いは、Sun の端末集配信装置には特殊なファームウェアがあるという点です。このファームウェアは、端末集配信装置がコンソールに対して起動時にブレークを送信するのを防

ぎます。コンソールデバイスがブレーク (あるいは、コンソールがブレークと解釈する可能性があるシグナル) を送信する可能性がある場合、そのブレークによってノードが停止されてしまうので注意してください。

質問: Sun がサポートする端末集配信装置がロックされた場合、再起動せずに、そのロックを解除できますか。

回答: 解除できます。リセットする必要があるポート番号を書きとめて、次のコマンドを入力してください。

```
telnet tc
Enter Annex port name or number: cli
annex: su -
annex# admin
admin : reset port_number
admin : quit
annex# hangup
#
```

Sun がサポートする端末集配信装置を構成および管理する方法についての詳細は、次のマニュアルを参照してください。

- 『Sun Cluster のシステム管理 (Solaris OS 版)』の「Sun Cluster の管理の概要」
- 『Sun Cluster 3.0-3.1 Hardware Administration Manual for Solaris OS』の第 2 章「Installing and Configuring the Terminal Concentrator」

質問: 端末集配信装置自体に障害が発生した場合はどのようにしたらいいですか。別の装置を用意しておく必要がありますか。

回答: ありません。端末集配信装置に障害が発生しても、クラスタの可用性はまったく失われません。ただし端末集配信装置が再び機能するまでは、ノードコンソールに接続できなくなります。

質問: 端末集配信装置を使用する場合に、セキュリティーはどのように制御しますか。

回答: 通常、端末集配信装置は、ほかのクライアントアクセスに使用されるネットワークではなく、システム管理者が使用する小規模なネットワークに接続されています。この特定のネットワークに対するアクセスを制限することでセキュリティーを制御できます。

質問: SPARC: テープドライブやディスクドライブに対して動的再構成をどのように使用するのですか。

回答: 次の手順を実行します。

- ディスクドライブやテープドライブが、アクティブなデバイスグループに属しているかどうかを確認します。ドライブがアクティブなデバイスグループに属していない場合は、そのドライブに対して DR 切り離し操作を行うことができます。
- DR 切り離し操作によってアクティブなディスクドライブやテープドライブに影響がある場合には、システムは操作を拒否し、操作によって影響を受けるドライブを特定します。そのドライブがアクティブなデバイスグループに属している場合は、89 ページの「SPARC: ディスクドライブとテープドライブに対する DR クラスタリング」に進みます。

- ドライブが主ノードのコンポーネントであるか、二次ノードのコンポーネントであるかを確認します。ドライブが二次ノードのコンポーネントである場合は、そのドライブに対して DR 切り離し操作を行うことができます。
- ドライブが主ノードのコンポーネントである場合は、主ノードと二次ノードを切り替えてから、そのデバイスに対して DR 切り離し操作を行う必要があります。

注意 – 二次ノードに対して DR 操作を行っているときに現在の主ノードに障害が発生すると、クラスタの可用性が損なわれます。これは、新しい二次ノードが提供されるまでは、主ノードのフェイルオーバー先が存在しないためです。

索引

と

な

に

ね

の