Sun Cluster の概念 (Solaris OS 版)

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

Part No: 819–0164–10
September 2004, Revision A

原典: *Sun Cluster Concepts Guide for Solaris OS*

Part No: 817-6537-10

Revision A

041118@10082

目次

はじめに

『*Sun™ Cluster* の概念 *(Solaris OS*版*)*』では、 SPARC™ 環境と x86 環境の両方における 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知識のある、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作業手順を行うには、Solaris™ オペレーティング環境に関する知識と、SunPlex 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・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*)*』
ハードウェアの設置と管理	『*Sun Cluster 3.x 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*』
コマンドと機能のリファレンス	『*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。

Sun のオンラインマニュアル

docs.sun.com では、Sun が提供しているオンラインマニュアルを参照することができます。マニュアルのタイトルや特定の主題などをキーワードとして、検索を行うこともできます。URL は、http://docs.sun.com です。

問い合わせについて

SunPlex システムのインストールまたは使用時に問題が起きた場合は、次の情報を用意したうえで、サービスプロバイダに連絡してください。

- 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 4/04)

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します
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。

第 1 章

基本知識と概要

SunPlex 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、 SunPlex 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、SunPlex の他のマニュアルと合わせて、SunPlex 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。

- SunPlex の基本知識と概要
- SunPlex の各ユーザーごとの役割と参照する情報
- SunPlex 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 SunPlex 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

SunPlex システムの基本知識

SunPlex システムは、Solaris オペレーティング環境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、SunPlex の導入目的を示します。

- 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の詳細については、『*Sun Cluster* の概要 *(Solaris OS* 版*)*』の「Sun Cluster によるアプリケーションの HA (高可用性)」を参照してください。

高可用性については、87 ページの「高可用性に関する FAQ」 を参照してください。

SunPlex システムの概要

この節では、 SunPlex 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
- アプリケーションプログラマ

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、SunPlex 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。

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

SunPlex の関連マニュアル (ハードウェア保守担当者)

次の SunPlex 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.x Hardware Administration Manual for Solaris OS*』

システム管理者

システム管理者にとって、SunPlex 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
- 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 オペレーティング環境のインストール
  - 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。

- 31 ページの「管理インタフェース」
- 32 ページの「クラスタ内の時間」
- 33 ページの「高可用性フレームワーク」
- 35 ページの「広域デバイス」
- 36 ページの「ディスクデバイスグループ」
- 40 ページの「広域名前空間」
- 41 ページの「クラスタファイルシステム」
- 44 ページの「ディスクパスの監視」
- 49 ページの「障害による影響の防止について」
- 57 ページの「データ サービス」

参照箇所 (システム管理者)

次の SunPlex 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 9/04* ご使用にあたって *(Solaris OS* 版*)*』
- 『*Sun Cluster 3.x Release Notes Supplement*』

アプリケーションプログラマ

SunPlex システムは Oracle (SPARC ベースシステム上)、NFS、DNS、 Sun™ Java System Web Server (従来の Sun Java System Web Server)、Apache Web Server (SPARC ベースシステム上)、Sun Java System Directory Server (従来の 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 ソフトウェアには、このようなアプリケーションの起動や停止、 監視を行う構成ファイルと管理メソッドが含まれています。フェイルオーバーサービスやスケーラブルサービスを新たに作成する必要がある場合には、SunPlex アプリケーションプログラミングインタフェース (API) やデータサービス実現技術 API (DSET API) を使用し、アプリケーションをクラスタ上の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や管理メソッドを作成することができます。

重要な概念 (アプリケーションプログラマ)

アプリケーションプログラマは、次の点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、汎用データサービス。プログラマは、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 (アプリケーションプログラマ)

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。

- 57 ページの「データ サービス」
- 68 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

SunPlex 関連マニュアル (アプリケーションプログラマ)

次の SunPlex のマニュアルには、アプリケーションプログラミング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*)*』

SunPlex システムの作業

すべての SunPlex システムの作業には、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

実行する作業	使用するマニュアル
クラスタハードウェアの設置	『*Sun Cluster 3.x 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.x 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*)*』
ボリュームマネージャの管理	『*Sun Cluster* のシステム管理 *(Solaris OS* 版*)*』およびボリューム管理のマニュアル
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 Data Services Developer's Guide for Solaris OS*』

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、SunPlex 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。

SunPlex 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

SunPlex システムを使用すると、25 ページの「SPARC: Sun Cluster トポロジの例」で説明しているように、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。

2 ノードクラスタ構成の例については、『*Sun Cluster* の概要 *(Solaris OS* 版*)*』の「Sun Cluster 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 オペレーティング環境と Sun Cluster ソフトウェアの両方を実行するマシンのことで、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す。

SPARC: Sun Cluster ソフトウェアを使用すると、1 つのクラスタに 2 から 8 台のノードを設定できます。サポートされるノード構成については、25 ページの「SPARC: Sun Cluster トポロジの例」を参照してください。

x86: Sun Cluster ソフトウェアを使用すると、1 つのクラスタに 2 つのノードを設定できます。サポートされるノード構成については、29 ページの「x86: Sun Cluster トポロジの例」 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。パラレルデータベース構成については、20 ページの「多重ホストデバイス」と第 3 章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、クラスタ内の他のノードとの通信を行います。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、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

クラスタメンバーとして機能するには、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 オペレーティング環境
- Sun Cluster ソフトウェア
- データサービスアプリケーション
- ボリューム管理ソフトウェア (Solaris Volume Manager™ 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 Volume Manager や VERITAS Volume Manager などのボリュームマネージャは必要ありません。
- Solaris オペレーティング環境、 Sun Cluster およびボリュームマネージャ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。

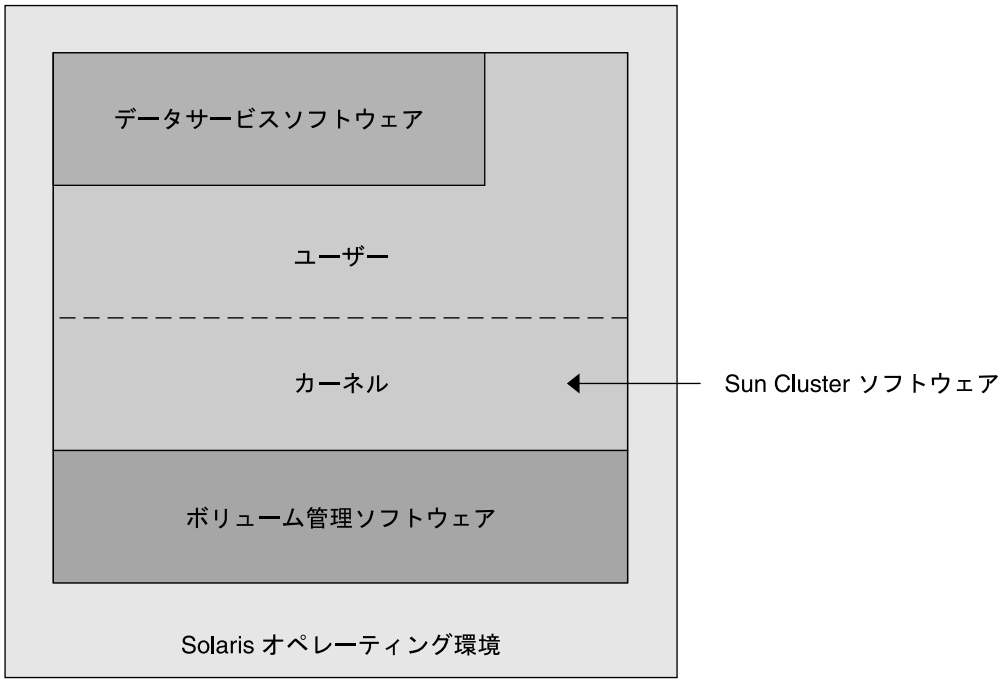


図 **2–1** Sun Cluster ソフトウェアコンポーネントの相互の関係

クラスタメンバーに関して頻繁に寄せられる質問については、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デバイス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、Sun Cluster は多重ホスト記憶装置を必要としません。

多重ホストデバイスには、次の特徴があります。

- 多重ホストディスクは、単一ノードの障害に耐えられます。
- 多重ホストディスクはアプリケーションデータだけでなく、アプリケーションバイナリと構成ファイルも保存できます。
- 多重ホストディスクはノード障害を防止します。クライアント要求があるノードを介してデータにアクセスしていて失敗した場合、これらの要求は、同じディスクへの直接接続を持つ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。
- 多重ホストディスクは、ディスクのマスターとなる主ノードを介して広域的にアクセスされるか、複数のローカルパスによって同時に直接アクセスされます。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

だけです。

ボリュームマネージャは、多重ホストデバイスのデータ冗長性に対して、ミラー化された構成または RAID-5 構成を提供します。現在、Sun Cluster がサポートするのは Solaris Volume Manager™ と VERITAS Volume Manager です。ボリュームマネージャとして SPARC ベースのクラスタ限定で使用できます。また、複数のハードウェア RAID　プラットフォームの RDAC RAID-5 ハードウェアコントローラで使用できます。

多重ホストデバイスをディスクのミラー化およびストライプ化と組み合わせると、ノードの障害および個々のディスクの障害の両方に対する防御策となります。

多重ホスト記憶装置については、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 デバイスからなる場合、この構成は多重イニシエータ SCSI と呼ばれます。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 バスの各デバイスに一意の 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。

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

- `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

このプロパティは 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。一意の `scsi-initiator-id` を各 SCSI ホストアダプタに設定するための手順は、『*Sun Cluster Hardware Collection*』の各ディスク格納装置に関する章に記載されています。

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、これらはノードの障害に対して保護されていません (可用性が低い)。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名前空間に含まれ、広域デバイスとして構成されています。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は、広域マウントポイントに置くことによって、他のノードから使用できるようになります。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で保護することはできません。

広域デバイスについては、35 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、非クラスタ化環境と同じ方法で設置および構成して使用できます。これらのデバイスは、Sun Cluster では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.x 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は、35 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。

少なくとも2 つの冗長な物理的に独立したネットワーク、またはパスを使用して、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し、単一地点による障害を回避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。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。これらは、パススルーおよび切り換え機能を実行して、3 つ以上のノードに同時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またはアダプタと接続点の間をつなぐ物理接続。

クラスタインターコネクトについては、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。

ボリュームマネージャいては、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、クラスタ上で実行され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。コンソールにアクセスするには、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)、または各ノードの `ttya` にアクセスできるその他のデバイスが必要です。

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 は、Ethernet ネットワーク上のマシンであり、Sun Enterprise E10000 サーバーをサポートするように構成されています。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

専用の UltraSPARC® ワークステーションまたは Sun Fire™ V65x サーバー (別名管理コンソール) を使用して、アクティブクラスタを管理します。通常、Cluster Control Panel (CCP)、Sun Management Center™ 用の Sun Cluster モジュール (SPARC ベースのクラスタと組み合わせた場合に限って使用可能) 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 の使用法の詳細については、『*Sun Cluster System Administration Guide*』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 プラットフォームによって構成されている場合は、管理コンソールからシステムサービスプロセッサ (SSP) にログインし、`netcon(1M)` コマンドを使用して接続を行う技術が必要となります。

通常、ノードはモニターなしで構成します。ノードのコンソールにアクセスするには、端末集配信装置を経由してそこからノードのシリアルポートに接続された管理コンソールから `telnet` セッションを行ってアクセスします(Sun Enterprise E10000 serverの場合は、システムサービスプロセッサから接続します)。詳細については、24 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。

管理コンソールについては、第 4 章を参照してください。

SPARC: Sun Cluster トポロジの例

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。

- SPARC ベースのシステムで構成される Sun Cluster は、実装するストレージ構成に関係なく、1 つのクラスタで最大 8 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 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、すべてのノードは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、すべてのノードがアプリケーションデータを保存するディスクに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する 2 ペア構成を設定できます。

次の図は、クラスタペア構成を示したものです。

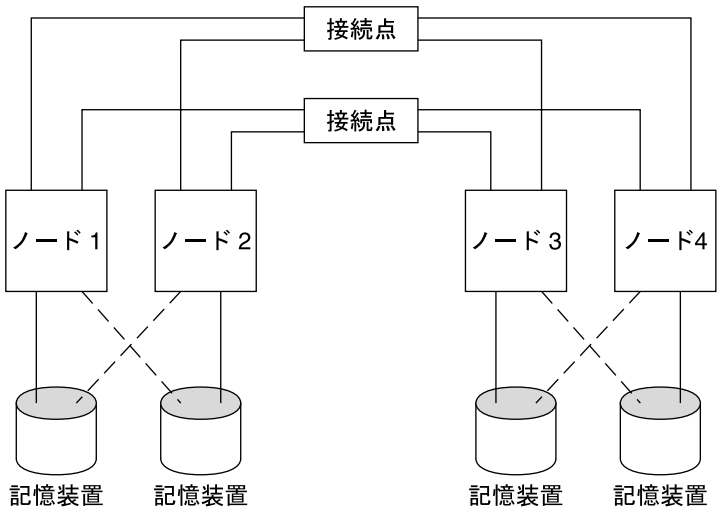


図 **2–2** SPARC: クラスタペアトポロジ

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。

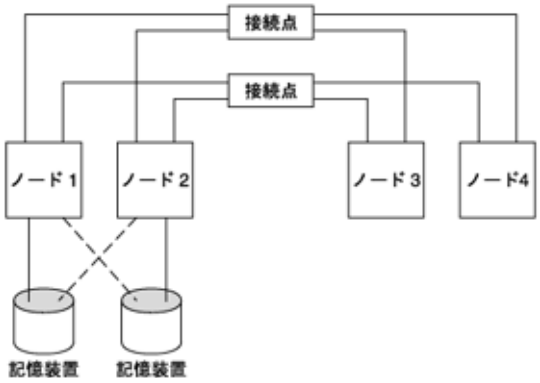


図 **2–3** SPARC: ペア +N トポロジ

SPARC: N+1 (星型) 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。

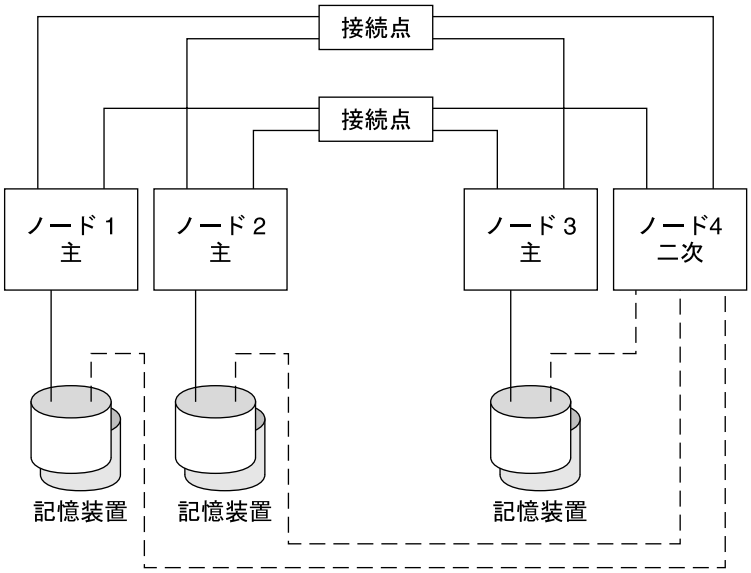


図 **2–4** SPARC: N+1 トポロジ

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は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。

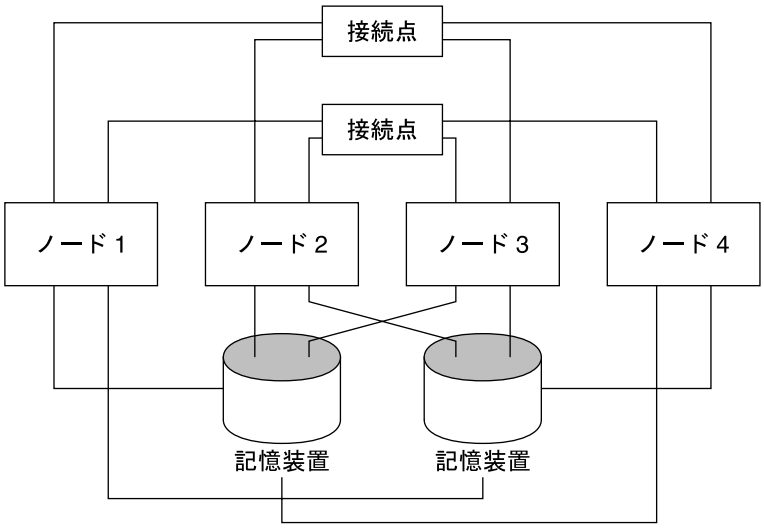


図 **2–5** SPARC: N*N トポロジ

x86: Sun Cluster トポロジの例

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、すべてのノードは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。

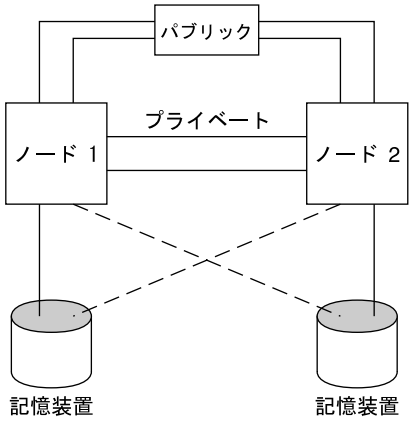


図 **2–6** x86: クラスタペアトポロジ

第 3 章

重要な概念 – 管理とアプリケーション開発

この章では、SunPlex 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、SunPlex 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

任意のユーザーインタフェースを使用して、SunPlex のインストール、構成、管理を行うことができます。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が使用できます。SunPlex システムには、Sun

Management Center の一部として実行され、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の詳細については、『*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。SunPlex 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 オペレーティング環境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、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、テンプレートファイル `ntp.cluster` を提供します (インストールされたクラスタノードの `/etc/inet/ntp.cluster` を参照)。これは、1 つのノードを優先ノードにし、すべてのクラスタノード間で対等関係を確立します。各ノードはそれぞれの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なわれます。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「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 オペレーティング環境をインストールしたときに時刻が誤って設定されていて、それを変更したい場合は、『*Sun Cluster* のシステム管理 *(Solaris OS* 版*)*』の「クラスタの管理」に手順が説明されています。

高可用性フレームワーク

SunPlex システムでは、ネットワークインタフェース、アプリケーションそのもの、ファイルシステム、および多重ホストデバイスなど、ユーザーとデータ間のパスにおけるすべてのコンポーネントの可用性が高くなっています。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。

次の表は、SunPlex 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たものです。

表 **3–1** SunPlex 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 Volume Manager と VERITAS Volume Manager、SPARC ベースのクラスタに限って使用可能)	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、ノードの障害を素早く検出して、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のを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、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。単一ノードの障害は、別ノードからのディスクに対する代替ハードウェアパスが存在するかぎり、ファイル、デバイス、およびディスクボリュームを使用する残りのノード上のプログラムに対して完全に透過的です。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 (CMM)

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、クラスタの同期化構成を実行します。これにより、クラスタリソースは、クラスタの新しいメンバーシップに基づいて再分配されます。

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。

クラスタが複数の独立したクラスタに分割されないように自身を防御する方法については、49 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メカニズム

CMM はノードで重大な問題を検出すると、クラスタフレームワークに依頼して、ノードを強制的に停止 (パニック) させ、クラスタメンバーシップからそのノードを取り除きます。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では、ノードは次の 2 つの方法で停止します。

- 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。フェイルファーストのこの機能については、49 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 1 つまたは複数のデーモン (`clexecd`、`rpc.pmfd`、`rgmd`、`rpc.ed`) が停止すると、CMM はそれを検出し、ノードを強制的に停止 (パニック) 状態にします。 クラスタデーモンが停止すると、ノードが強制的に停止させられ、次のようなメッセージがそのノードのコンソールに表示されます。

```
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、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 OpenBoot PROM `ok` プロンプトから、`eeprom` 1M で設定できます。

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しなければなりません。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

SunPlex 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、クラスタ全体で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を提供しているときに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。SunPlex 広域デバイスには、ディスク、CD-ROM、テープが含まれます。ディスクは、唯一サポートされている多重ポート広域デバイスです。つまり、CD-ROM とテープは、現在可用性の高い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、クラスタ内の各ディスク、CD-ROM、テープデバイスに一意の ID を自動的に割り当てます。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。詳細については、40 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の場合、ディスクは複数のノードがホストとな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 (DID)

Sun Cluster ソフトウェアは、デバイス ID (DID) 擬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。

デバイス ID (DID) 擬似ドライバは、クラスタの広域デバイスアクセス機能の重要な部分です。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、それぞれに対して、クラスタのすべてのノードで一貫した一意のメジャー番号およびマイナー番号を割り当てます。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 ドライバによって割り当てられた一意のデバイス ID を利用して行われます。

この方法により、ディスクを利用するすべてのアプリケーション (ボリュームマネージャまたは raw デバイスを使用するアプリケーション) が、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、ノード `1` は多重ホストディスクを `c1t2d0` と表示し、同じディスクをノード 2 は `c3t2d0` と表示する場合があります。DID ドライバは、そのノードが代わりに使用する d10 などの広域名を割り当てて、各ノードに対して多重ホストディスクとの一貫したマッピングを与えます。

デバイス ID の更新および管理は、`scdidadm(1M)` および `scgdevs(1M)` を介して行われます。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

SunPlex 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なければなりません。最初に多重ホストディスク上で、ボリュームマネージャのディスクグループ— Solaris Volume Manager ディスクセットまたは VERITAS Volume Manager ディスクグループのいずれか (SPARC ベースのクラスタでのみ使用可能)—を作成します。次に、ボリュームマネージャ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、SunPlex システムは、どのノードがどのボリュームマネージャ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 (制御可能) 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間の関係の詳細については、『*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。

ディスクデバイスグループでは、ボリュームマネージャ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、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。

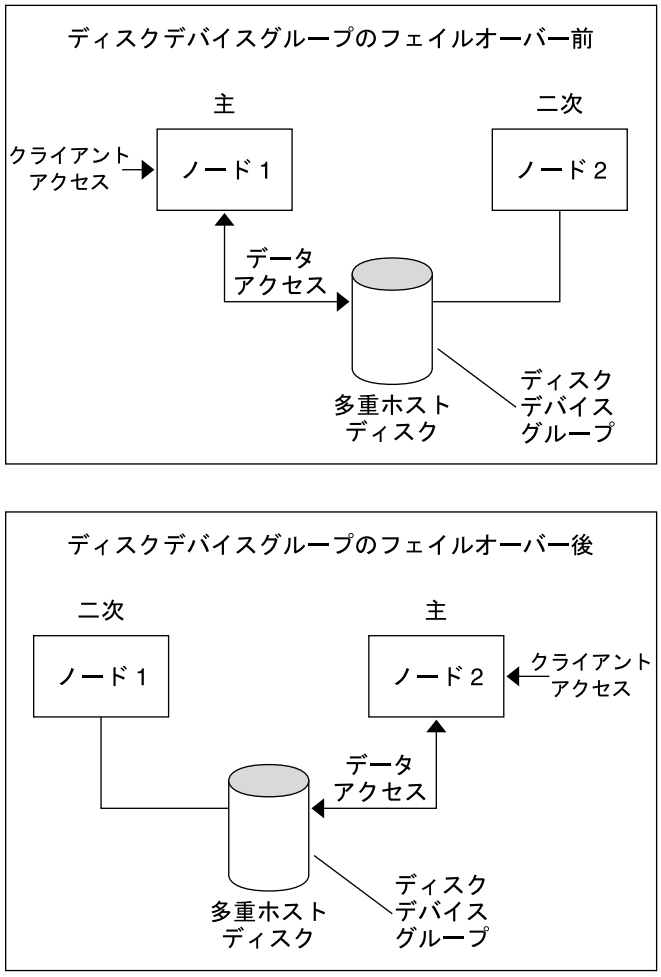


図 **3–1** ディスクデバイスグループのフェイルオーバー

多重ポートディスクデバイスグループ

ここでは、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 2 つのプロパティ`preferenced` と `numsecondaries` があります。`preferenced` プロパティを使用すると、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できます。`numsecondaries` プロパティは、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。

高可用性サービスでは、主ノードが停止し、主ノードになる資格のある二次ノードがもはや存在しないときに、停止とみなされます。サービスのフェイルオーバーが発生したときに、preferenced プロパティが true であった場合、ノードはノードリストの順番に従って、二次ノードを選択します。設定されたノードリストによって、ノードが一次制御権の獲得を試みる順序、またはスペアから二次への移行を試みる順序が決まります。デバイスサービスの設定は、scsetup(1M) ユーティリティで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) に対応する設定には、デバイスサービスの設定が適用されます。

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。このようなチェックポイントによる性能の低下やオーバーヘッドの増加を最小限に抑えるためにスペアノードのサポートが実装されています。ディスクデバイスグループには、デフォルトで 1 つの主ノードと 1 つの二次ノードがあります。使用可能な残りのプロバイダノードはスペア状態でオンラインになります。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度の高いノードが二次ノードになります。

望ましい二次ノードの数には、1 から、デバイスグループにある動作可能な非主プロバイダノードの数までの任意の整数を設定できます。

注 – Solaris ボリュームマネージャを使用する場合は、ディスクデバイスグループを作成後にのみ numsecondaries プロパティにデフォルト以外の数を設定することができます。

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の実際の数です。ただし、動作可能な非主プロバイダの数が望ましい数よりも小さい場合を除きます。構成に対してノードの追加や切り離しを行う場合には、 numsecondaries プロパティを変更してからノードリストを十分に確認する必要があります。ノードリストと望ましい二次ノード数を正しく保つことは、構成された二次ノード数と、フレームワークによって与えられる実際の数の不一致を防ぐ上で有効です。構成に対するノードの追加と削除を管理する場合、Solaris Volume Manager デバイスグループには、metaset(1M) コマンドを使用します。または、Veritas Volume Manager を使用している場合は、VxVM ディスクデバイスグループに scconf(1M) コマンドを preferenced と numsecondaries というプロパティ設定と組み合わせて使用します。ディスクデバイスグループのプロパティ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

広域名前空間は、広域デバイスを有効にする Sun Cluster ソフトウェアの機構です。広域名前空間には、ボリュームマネージャ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。

通常、Solaris Volume Manager の場合、ボリュームマネージャの名前空間は /dev/md/*diskset*/dsk (と rdsk) ディレクトリにあります。Veritas VxVM では、ボリュームマネージャの名前空間は /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、クラスタ全体にインポートされた Solaris Volume Manager 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で構成されます。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。

SunPlex システムでは、ボリュームマネージャのローカルの名前空間の各デ バイスノードは、 /global/.devices/node@*nodeID* ファイルシステム内の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。*nodeID* は、クラスタの各ノードを表す整数です。Sun Cluster ソフトウェアは、その標準的な場所にシンボリックリンクとしてボリュームマネージャデバイスも常時表示します。広域名前空間と標準ボリュームマネージャ名前空間は、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/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 Volume Manager	`/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。プロセスはファイルがシステム内のどこに置かれていても開くことができ、また、すべてのノード上のプロセスが同じパス名を使用してファイルを見つけられます。

  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。複数のクラスタノードで動作するアプリケーションは、クラスタファイルシステムのファイルに対してアドバイザリファイルロッキング機能を使用することによって、データへのアクセスを同期化することができます。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る限り、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。

広域デバイスには、`mount -g` を使用して広域に、または `mount` を使用してローカルにファイルシステムをマウントできます。

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、UFS など、実際に使用するファイルシステムしか認識できません。

クラスタファイルシステムの使用法

SunPlex 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 Volume Manager ディスクセット、VxVM 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マネージャ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。

通常のファイルシステムと同様、クラスタファイルシステムは 2 つの方法でマウントできます。

- 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、他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。詳しくは、『*Sun Cluster 3.1 9/04 Software Collection for Solaris OS (SPARC Platform Edition)* 』および『 *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) を提供します。フェイルオーバーを行うには、アフィニティ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い方については、『*Sun Cluster* データサービスの計画と管理 *(Solaris OS* 版*)*』の「HA 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しているディスクデバイスグループを同時に起動するためにも使用できます。詳細については、68 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

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は、syncdir マウントオプションを使用できます。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 を指定すると、POSIX 準拠の書き込みが保証されます。指定しないと、UFS 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 を指定しないことで生じる問題はほとんどないため、syncdir を指定しないで、パフォーマンスを向上させることを推奨します。

SPARC ベースのクラスタを使用する場合、Veritas VxFS には UFS の syncdir マウントオプションに相当する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については、88 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の監視 (Disk-Path Monitoring: 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、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 4/04 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す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。

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、クラスタ内の単一または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の詳細については、`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の詳細については、`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。次のリストに、`scdpmd` が初期起動時にどのように動作するかを説明します。

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 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 の詳細については、35 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 – DPM デーモンに関連するすべてのエラーは `pmfd(1M)` によって報告されます。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、MPxIO、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。また、`scdpm` コマンドは障害が発生した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するときにも使用できます。

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「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ありません。指定しないと、all が使用されます。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。詳細については、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。

定足数および定足数デバイス

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。

CMM の詳細については、『*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。

クラスタパーティションからは次の 2 種類の問題が生じます。

- 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。1 つのパーティションのノードが他のパーティションのノードと通信できないため、それぞれのパーティションは、それ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、最後に機能したクラスタパーティションではないノード上のクラスタを起動するときに起きる場合があります。

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。この外部投票は、定足数デバイスにより提供されます。

定足投票数について

`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

このコマンドの詳細については、`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、デバイスに伴な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、Sun Cluster ソフトウェアは定足数デバイスに投票数 *N*-1 を割り当てます。ここで *N* は、定足数デバイスに伴なう投票数となります。たとえば、投票数がゼロ以外の2 つのノードに接続された定足数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。

定足数デバイスは、クラスタのインストール中に構成するか、後で『*Sun Cluster* のシステム管理 *(Solaris OS* 版*)*』の「定足数の管理」に記載された手順に従って構成します。

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。この障害が発生すると、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ノードの一部が、個々のクラスタまたはクラスタの一部を形成しようとします。各部分、つまりパーティションは、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誤って認識します。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て、ディスクデバイスグループへのアクセスが可能になり、わずかな割り込みだけで処理が続行されます。このプロセス中、古い主ノードは、新しい主ノードが起動される前に、デバイスへのアクセスを放棄しなければなりません。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。

SunPlex システムは、SCSI ディスク予約を使用して、障害による影響の防止機能を実装します。SCSI 予約を使用すると、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て、これらのディスクへのアクセスが防止されます。

SCSI-2 ディスク予約は、ある形式の予約をサポートしています。これは、ディスクに接続されたすべてのノードへのアクセスを付与するか (予約が設定されていない場合)、または単一ノード (予約を保持するノード) へのアクセスを制限するものです。

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。この障害による影響の防止機能が実行される場合、通常、阻止されるノードは、そのコンソールに「`reservation conflict`」(予約の衝突) というメッセージを表示して停止します。

予約の衝突は、ノードがクラスタメンバーではなくなったことが検出された後で、SCSI 予約がこのノードと他のノードの間で共有されるすべてのディスクに対して設定されると発生します。阻止されるノードは阻止されていることを認識しない場合があり、共有ディスクのどれかにアクセスしようとして、予約を検出して停止します。

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 は特定のディスクドライバに対する命令です。ディスクが他のノードによって予約されているためにそのディスクにアクセスできないと、ノードは自らをパニックさせる (強制的に停止する) ことができます。

MHIOCENFAILFAST ioctl によって、ドライバはノードがディスクに対して行ったあらゆる読み書きについて、 `Reservation_Conflict` というエラーコードが戻っていないかどうかを調べます。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、予約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、予約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。フェイルファースト機構は、SCSI-2 ディスクでも SCSI-3 ディスクでも同じように機能します。

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。定足数を獲得できるパーティションのノードによって予約されている共有ディスクに、定足数をもたないノードからアクセスすると、ノードは予約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を発生します。

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 `ok` プロンプトから、`eeprom`(1M) で設定できます。または、x86 ベースのクラスタでは、BIOS のブート後に任意で実行する SCSI ユーティリティで設定できます。

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* 個の定足数デバイスが、それぞれ 1 から *N* 個のノードと N+1 ノードの 1 つに接続される N+1 構成では、1 から *N* のどのノードで障害が起きた場合でも、また、*N*/ **2** 個のノードに障害が発生した場合でも、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されている *N* ノード構成では、 *N*-**1** ノードのいずれかに障害が起きても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。

望ましくない定足数の構成例については、56 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、53 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。要件を守らないと、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しているデバイスのリストについては、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 予約に対応した多重ホスト共有ディスク
  - SCSI-2 予約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 1 つ以上の定足数デバイスを構成する必要があります。図 3–2 を参照してください。

望ましくない定足数の構成例については、56 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、53 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にクラスタを形成できません。

注 – 特定の環境によっては、ニーズに合うよう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、55 ページの「変則的な定足数の構成」に記載されるような構成では、クラスタの可用性が低下し、定足数の投票がノードの投票を上回ります。クラスタには、ノード A とノード B の間の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起きるという特性があります。

  このベストプラクティスの例外については、55 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、障害防護プロセスが高速化されます。54 ページの「2 ノード構成よりも大きい定足数」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。

望ましくない定足数の構成例については、56 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、53 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

望ましくない定足数の構成例については、56 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。

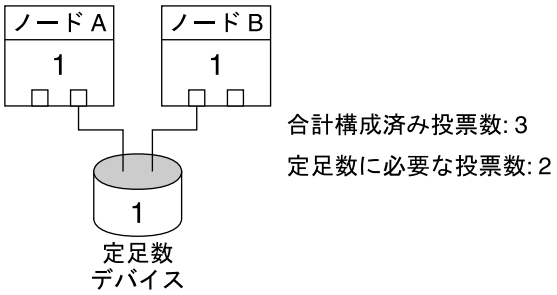
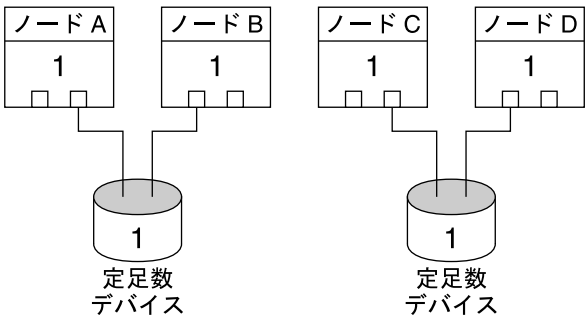


図 **3–2** 2 ノード構成

2 ノード構成よりも大きい定足数

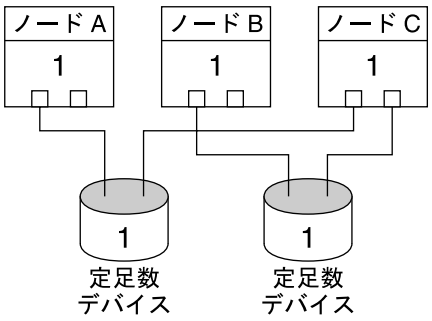
定足数デバイスを持たない 2 ノードより大きいクラスタを構成することもできます。ただし、この場合、クラスタ内の過半数のノードなしにクラスタを開始できません。



合計構成済み投票数: 6

定足数に必要な投票数: 4

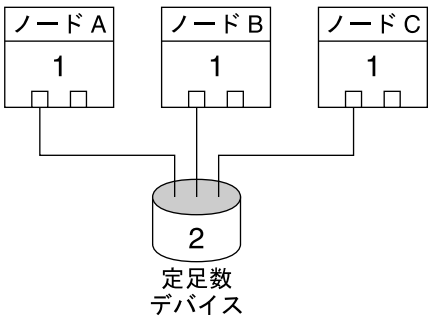
この構成は、どちらかのペアが稼働し続けるためには各ペアが稼働していなければならない。



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 は、ノード A および ノード 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) を実行していることを前提としています。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、51 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。

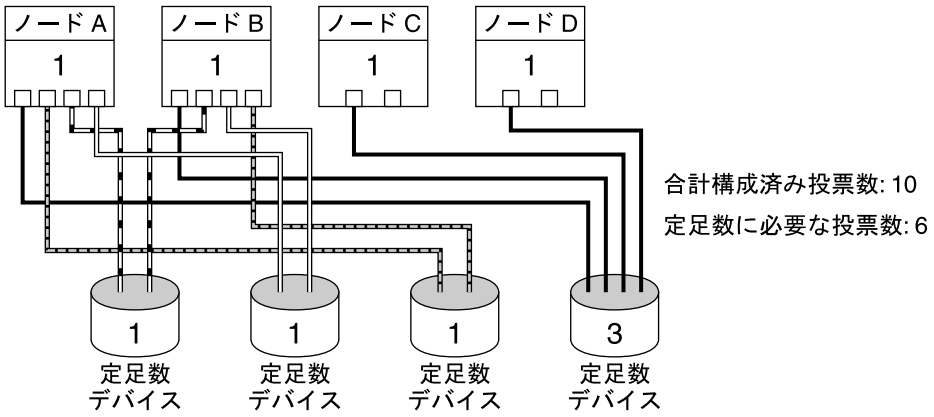
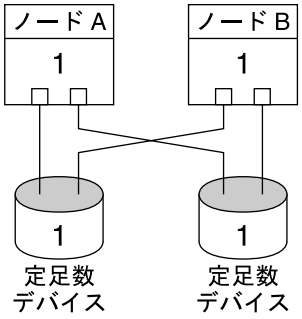


図 **3–3** 変則的な構成

望ましくない定足数の構成

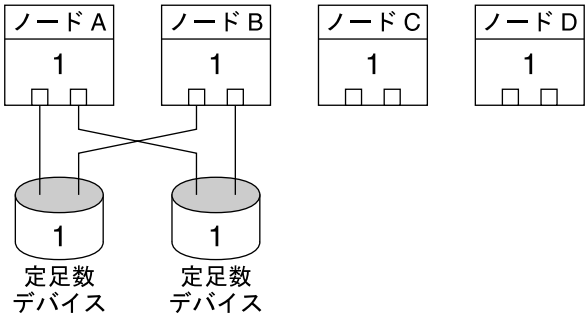
推奨される定足数の構成例については、53 ページの「推奨される定足数の構成」を参照してください。



合計構成済み投票数: 4

定足数に必要な投票数: 3

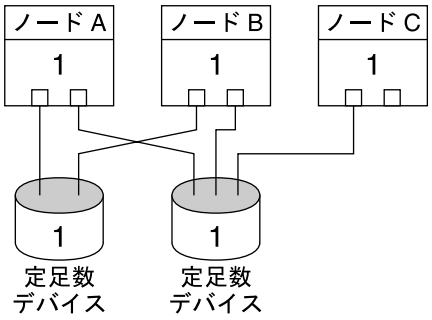
この構成は、定足数デバイス投票が、ノードの投票数より少なくなければならないというベストプラクティスに反する。



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: 5

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。

データ サービス

データサービスは、Sun Java System Web Server (従来の Sun Java System Web Server)、SPARC ベースのクラスタであれば Oracle など、単一サーバーではなくクラスタで実行するように構成された他社のアプリケーションを意味します。データサービスは、アプリケーションや、専用の Sun Cluster 構成ファイル、および、アプリケーションの以下の操作を制御する 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
- データサービスのタイプについては、『 *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。ユーザーから見れば、この 2 つの構成には何の違いもありません。しかし、クラスタ化されたアプリケーションでは、処理が速くなる可能性があるだけでなく、可用性が高まります。

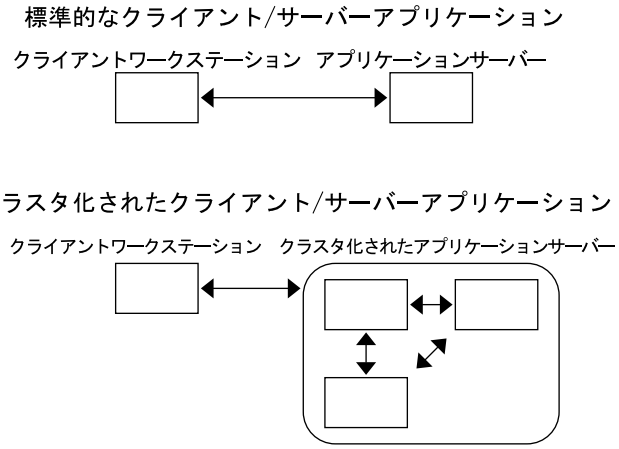


図 **3–4** 標準的なクライアントサーバー構成とクラスタ化されたクライアントサーバー構成

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設定します。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の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」か「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 (入れ替え不可) を指定する必要があります。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 (主ノード) に関連付けられています。主ノード、ネットワークリソース、アプリケーションリソースで障害が発生すると、別のクラスタノード (二次) へのフェイルオーバーが行われます。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。

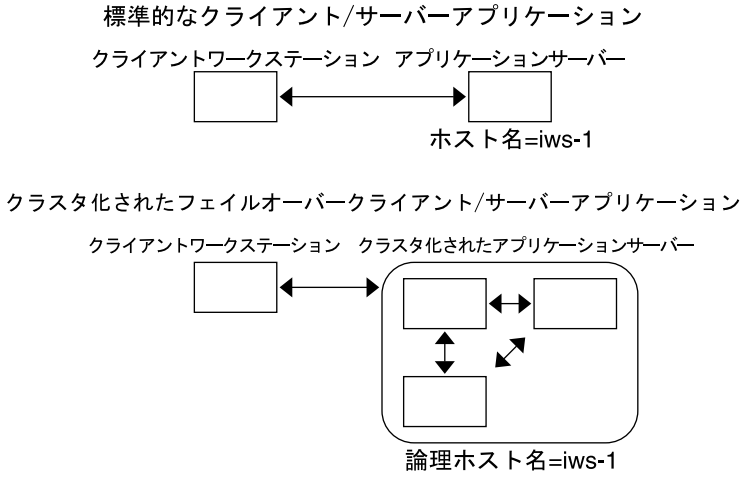


図 **3–5** 固定ホスト名と論理ホスト名

共有アドレスも最初は 1 つのノードに対応づけられます。このノード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、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。これを「広域インタフェース」といいます。

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設定される点です。この設定では、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で障害が発生した場合には、共有アドレスを、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に移すことができます (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る)。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。

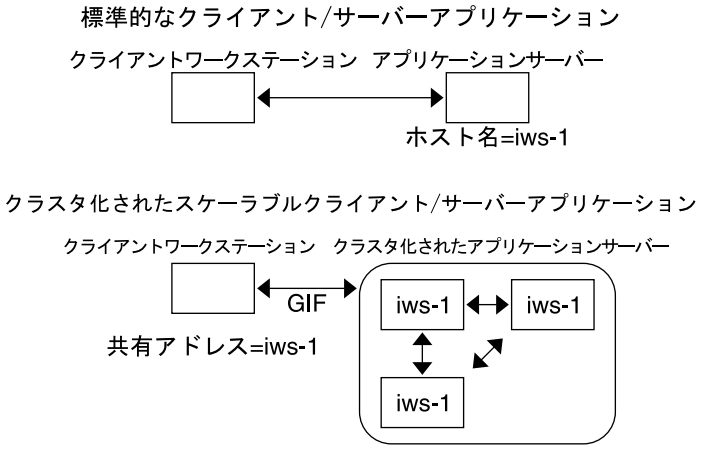


図 **3–6** 固定ホスト名と共有アドレス

データサービスメソッド

Sun Cluster ソフトウェアでは、Resource Group Manager (RGM) の制御下で動作する一連のサービス管理メソッドが提供されます。RGM は、これらのメソッドを使用し、クラスタノードで動作するアプリケーションの起動や停止、監視を行います。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他に、SunPlex システムからも API やいくつかのデータサービス開発ツールが提供されます。これらのツールを使用すれば、アプリケーションプログラマは、独自のデータサービスメソッドを開発し、他のアプリケーションを高可用性データサービスとして Sun Cluster ソフトウェアの下で実行できます。

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、データサービスの構成に従って、同じノードでそのインスタンスを再起動しようとするか、別のノードでそのインスタンスを起動 (フェイルオーバー) しようとします。

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。アプリケーションリソースが含まれる「スケーラブルリソースグループ」と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」) が含まれるフェイルオーバーリソースグループです。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) を介してクラスタに入り、負荷均衡ポリシーによって設定されたいくつかの定義済みアルゴリズムの 1 つに基づいてノードに分配されます。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。他の共有アドレスをホストしている別のノード上に、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。アプリケーションインスタンスの実行に失敗した場合、そのインスタンスは同じノード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。あるいは、残りのノードで実行し続けて、サービススループットを低下させることになります。

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 は、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ています。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 (実線)、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` に依存する (破線) ことに注意してください。

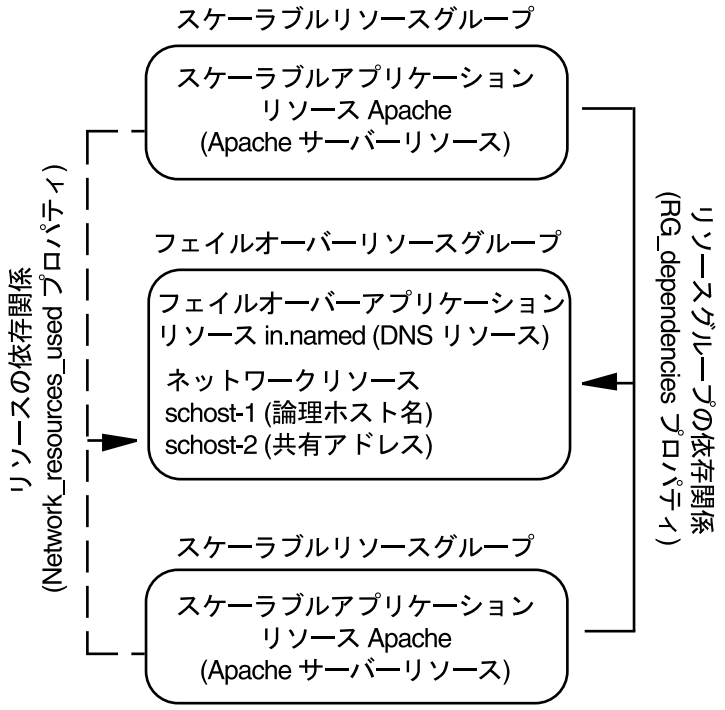


図 **3–7** SPARC: フェイルオーバーリソースグループとスケーラブルリソースグループの例

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、*pure* と *sticky* の 2 つのクラスがあります。pure サービスとは、そのいずれかのインスタンスがクライアント要求に応答できるサービスをいいます。sticky サービスとは、クライアントが同じインスタンスに要求を送るサービスをいいます。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、クライアント要求の 1/3 ずつを負担します。ウェイトは、`scrgadm(1M)` コマンドインタフェースまたは SunPlex Manager GUI を使用し、管理者が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。単一ポートを待機しているそのサーバーインスタンスという点で、そのクライアントは *sticky* であると呼ばれます。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、負荷分散ポリシーがサーバー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が、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場合、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という点で、そのクライアントは *sticky* であると呼ばれます。

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が、購入した製品の支払いをクレジットカードで行うためには、ポート 443 で 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 アドレスという点で、ポートに対して *sticky wildcard* です。

このポリシーの例としては、受動モード FTP があります。クライアントは、ポート 21 の FTP サーバーに接続して、動的ポート範囲のリスナーポートサーバーに接続するよう、そのサーバーから通知を受けます。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によってクライアントに通知した、同じノードに転送されます。

これらの各 sticky ポリシーでは、ウェイト設定した (weighted) 負荷均衡ポリシーがデフォルトで有効であるため、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にリダイレクトされます。インスタンスが実行されているノードとクライアントが関係を確立すると、そのノードがアクセス可能で、負荷分散ポリシーが変更されない限り、今後の要求はそのインスタンスに送られます。

次に、各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。アクティブノードでは、新しい接続数の合計の約 X/A がこのアクティブノードに送られると予測できます。 ただし、この場合接続数の合計が十分に大きな数であるとします。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、クライアントからの各要求が、最初の要求はノード 1、2 番目の要求はノード 2 といったように常に異なるノードに送られます。

- 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 (したがって事前には認識されない)。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。このようなリソースグループの移行が起こると、それまでの二次ノードが新しい主ノードになります。元の主ノードがオンラインに復帰したときにどのようなアクションを取るか (つまり、元の主ノードを再び主ノードに戻す (フェイルバックする) か、現在の主ノードをそのまま継続するか) は、フェイルバックの設定値で決まります。。この選択は、リソースグループのプロパティ `Failback` で設定します。

特定のインスタンスでは、リソースグループをホストする元のノードに障害が発生して再起動が繰り返される場合、フェイルバックを設定すると、リソースグループの可用性が低下することがあります。

データサービス障害モニター

SunPlex の各データサービスには、データサービスを定期的に探索してその状態を判断する障害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、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設定できます。

アプリケーションがフェイルオーバーサービスを使用できるかどうかを判断するための基準があります。個々の基準は、アプリケーションで使用できる API について説明した SunPlex のマニュアルに記載されています。

次に、それぞれのサービスがスケーラブルデータサービスの構造を利用できるかどうかを知るために役立つガイドラインをいくつか示します。スケーラブルサービスの一般的な情報については、60 ページの「スケーラブルデータサービス」を参照してください。

次のガイドラインを満たす新しいサービスは、スケーラブルサービスを利用できます。既存のサービスがこれらのガイドラインに従っていない場合は、そのサービスがガイドラインに準拠するように、一部を書き直さなければならない場合があります。

スケーラブルデータサービスには、以下の特性があります。第一に、こうしたサービスは 1 つまたは複数のサーバー「インスタンス」からなります。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。

次に、サービスが外部論理データ記憶領域を使用する場合は、この記憶領域に対する複数のサーバーインスタンスからの同時アクセスの同期をとって、更新が失われたり、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。この格納をメモリー内部の状態と区別するために「外部」と呼び、格納がそれ自体複製されている場合でも単一の実体として見えるため、「論理的」と呼んでいることに注意してください。また、この論理データ格納には、サーバーインスタンスが記憶領域を更新するたびに、その更新がすぐに他のインスタンスで見られるという特性があります。

SunPlex 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。このサービスの複数インスタンスが実行されている場合は、それぞれがこの外部ログへのアクセスを持ち、同時にこのログ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。サービスは、`fcntl(2)` および `lockf(3C)` によって通常の Solaris ファイルロックを使用して、必要な同期をとることができます。

この種のストアのもう一つの例は、可用性の高い Oracle や SPARC ベースのクラスタ用の Oracle Real Application Clusters などのバックエンドデータベースです。このようなバックエンドデータベースサーバーは、データベース照会または更新トランザクションを使用するのに内部組み込みの同期を使用するため、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がありません。

現在の実現状態ではスケーラブルサービスではないサービスの例として、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、他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。このようなケースでは、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、サービスは同

時アクセスの同期をとる必要はありません。この場合、このプライベートデータが広域にアクセス可能になる可能性があるため、このデータをクラスタファイルシステムのもとで保存しないように注意する必要があります。

データサービス API とデータサービス開発ライブラリ API

SunPlex システムには、アプリケーションの可用性を高めるための次のサービスがあります。

- SunPlex 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
- DSDL API (データサービス開発ライブラリ API)
- 汎用データサービス

『*Sun Cluster* データサービスの計画と管理 *(Solaris OS* 版*)*』では、SunPlex システムで提供されるデータサービスのインストールおよび構成方法について説明しています。『*Sun Cluster 3.1 9/04 Software Collection for Solaris OS (SPARC Platform Edition)* 』では、他のアプリケーションを実装して Sun Cluster フレームワークにおいて可用性を高める方法を説明しています。

Sun Cluster API を使用すると、アプリケーションプログラマは、障害モニターおよびデータサービスインスタンスを起動して停止する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設計できます。さらに、SunPlex システムの「汎用」データサービスを使用すれば、アプリケーションをフェイルオーバーサービスかスケーラブルサービスとして実行するための起動メソッドや停止メソッドを簡単に生成できます。

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。クラスタ化ソフトウェアは、可用性や性能を高めるために複数のインターコネクトを使用します。内部トラフィック (ファイルシステムデータ、スケーラブルサービスデータなど) に関しては、メッセージはラウンドロビン方式を使用し、利用できるすべてのインターコネクト間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、クラスタ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、ノード 1 のプライベートホスト名が `clusternode1-priv` の場合、その名前を使用してクラスタインターコネクト経由でノード 1 と通信します。この名前を使用して開いた TCP ソケットは、クラスタインターコネクトを介してルーティングされ、ネットワーク障害が発生した場合には透過的に再ルーティングされます。

複数のプライベートホスト名がインストール時に設定されているため、クラスタインターコネクトでは、その時点で選択した任意の名前を使用できます。実際の名前は、`scha_cluster_get`(3HA) に `scha_privatelink_hostname_node` 引数を指定することによって取得できます。

クラスタインターコネクトをアプリケーションレベルで使用する場合には、個々のノードペア間の通信に 1 つのインターコネクトが使用されます。ただし、可能であれば、別のノードペアには別のインターコネクトが使用されます。たとえば、3 つの SPARC ベースノードで動作するアプリケーションが、クラスタインターコネクト経由で通信しているとします。この場合、たとえば、ノード 1 とノード 2 の通信にはインタフェース `hme0` が、ノード 1 とノード 3 の通信にはインタフェース `qfe1` がそれぞれ使用されます。つまり、アプリケーションによる 2 ノード間の通信は 1 つのインターコネクトに制限されますが、内部のクラスタ通信はすべてのインターコネクト上にストライプ化されます。

インターコネクトはアプリケーションと内部のクラスタトラフィックによって共有されるため、アプリケーションから使用できる帯域幅の量は、他のクラスタトラフィックに使用される帯域幅の量に左右されます。インターコネクトに障害が発生すると、内部トラフィックは残りのインターコネクト上にラウンドロビン方式で分散されますが、障害が発生したインターコネクト上のアプリケーションは動作しているインターコネクトに切り替えられます。

クラスタインターコネクトでは、2 つのタイプのアドレスがサポートされます。さらに、プライベートホスト名に対する `gethostbyname`(3N) では、通常 2 つの IP アドレスが返されます。最初のアドレスを「論理 pairwise アドレス」と呼び、2 番目のアドレスを「論理 pernode アドレス」と呼びます。

個々のノードペアには、異なる論理 pairwise アドレスが割り当てられます。この小規模な論理ネットワークでは、接続のフェイルオーバーがサポートされます。さらに、各ノードには、固定した pernode アドレスが割り当てられます。つまり、`clusternode1-priv` の論理 pairwise アドレスはノードごとに異なりますが、`clusternode1-priv` の論理 pernode アドレスは各ノードで同じです。ただし、個々のノードが pairwise アドレスを自分で持っているわけではないため、ノード 1 で `gethostbyname(clusternode1-priv)` を実行しても、論理 pernode アドレスだけが返されます。

クラスタインターコネクト経由で接続を受け付け、セキュリティ目的で IP アドレスを検証するアプリケーションは、最初の IP アドレスだけではなく、gethostbyname から返ったすべての IP アドレスを検査する必要があります。

アプリケーション全体にわたって一貫した IP アドレスが必要な場合は、クライアント側でもサーバー側でもその 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、すべての接続にこの pernode アドレスが使用されます。

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 (従来の Sun Java System Web Server)、 Apache Web Server などのアプリケーションは、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、そのアドレスに依存します。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 はリソースタイプ SUNW.oracle-server で、Sun Cluster HA for Apache はリソースタイプ SUNW.apache です。

注 – リソースタイプ SUNW.oracle-server を使用するのは、SPARC ベースのクラスタに限られます。

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がすでに定義されています。

ネットワークリソースは、SUNW.LogicalHostname または SUNW.SharedAddress リソースタイプです。これら 2 つのリソースタイプは、Sun Cluster ソフトウェア製品によって事前に登録されています。

SUNW.HAStorage および HAStoragePlus リソースタイプは、リソースと、そのリソースが依存しているディスクデバイスグループの起動を同期させるために使用します。この同期をとることで、データサービスの起動前に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利用可能になります。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 は Sun Cluster 3.0 5/02 で追加されたリソースタイプであり、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。この機能の詳細については、43 ページの「HAStoragePlus リソースタイプ」を参照)

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。このプロセスの詳細については、『*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。

リソースグループマネージャ (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コンテナにリソースを作成して管理します。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。リソースやリソースグループは、RGM のアクションに従ってオンラインになったり、オフラインになります。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の詳細については、69 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」を参照してください。RGM 制御の下でリソース管理プロジェクトを起動する方法については、68 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。

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。リソースグループを RGM によって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ためには、リソースグループの状態が管理になっていなければなりません。

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、あるアクションが必要です。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。

INIT や FINI メソッドは、一般にスケーラブルサービスのネットワークリソースに対して使用されますが、これらのメソッドは、アプリケーションによって行われない任意の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。

  何らかの理由で、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。

  これらの状態は、スイッチオーバーまたはフェイルオーバー時の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、管理アクションによって変更することもできます。`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/オフライン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。

  あるリソースグループがオフラインであるなら、そのすべてのリソースもオフラインです。あるリソースグループがオンラインであるなら、有効にされているそのすべてのリソースもオンラインです。

  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

SunPlex 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値は構成できます。標準的なプロパティはすべてのデータサービスに共通です。拡張プロパティは各データサービスに特定のものです。標準プロパティおよび拡張プロパティのいくつかは、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を構成するために使用されます。標準プロパティ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「標準プロパティ」を参照してください。

RGM 拡張プロパティは、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

RGM を使ってデータサービスをオンラインにするときに、これを特定の Solaris プロジェクト名の下で起動することができます。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とプロジェクト ID を対応付けることによって、ユーザーは、Solaris 環境で提供される高度なコントロールを使ってクラスタ内の負荷や使用量を管理できるようになります。

注 – この構成を行うためには、Sun Cluster ソフトウェアの最新リリースと Solaris 9 が必要です。

ノードを他のアプリケーションと共有している場合には、クラスタ環境で Solaris 管理機能を使用することによって、最も重要なアプリケーションに高い優先度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、優先度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性能を高めることができます。

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、タスクや、これに類するコンポーネントを「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を「リソース」と呼んでいます。ここでは、RGM の制御下にある Sun Cluster エンティティを「リソース」と呼び、CPU 時間やプロセス、タスクなどを「サプライ」と呼びます。

以下の説明は、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を概念的に述べたものです。さらに、以下の説明では、Solaris 環境の管理機能を使用するために必要なフェイルオーバーのシナリオやヒントについて述べます。管理機能の概念や手順については、『*Solaris 9 System Administrator Collection - Japanese*』の『*Solaris* のシステム管理 *(*資源管理とネットワークサービス*)*』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すべてのリソースを有効にします。
4. リソースグループを管理可能にします。
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. ステップ 5 で作成したプロジェクトとリソースグループ名を対応付けるために標準プロパティを構成します。
7. リソースグループをオンラインにします。

標準の `Resource_project_name` または `RG_project_name` プロパティを使って Solaris プロジェクト ID とリソースまたはリソースグループを対応付ける場合には、`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「標準プロパティ」を参照してください。プロパティの説明については、`r_properties`(5) と `rg_properties`(5) 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`) に登録されていなければなりません。また、root ユーザーは指定したプロジェクトのメンバーとして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要については、『*Solaris 9 System Administrator Collection*』の『 *Solaris* のシステム管理 *(*資源管理とネットワークサービス*)* 』の「プロジェクトとタスク」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、新しいプロジェクト名を有効にするためには、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必要があります。

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。FSS をデフォルトのスケジューラとして設定するためのコマンド行インタフェースについては、`dispadmin(1M)` のマニュアルページを参照してください。さらに、`priocntl(1)`、`ps(1)`、`FSS(7)` のマニュアルページも参照してください。
- 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 での制御を使用してデータサービスを構成する場合は、スイッチオーバーやフェイルオーバーの際にリソースの制御や管理をどのように行うかを決める必要があります。まず、新しいプロジェクトを構成する前に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)` で設定された `nodelist`、`failback`、`maximum_primaries`、`desired_primaries` リソースグループプロパティを使って、使用するリソースグループのノードリスト優先度を確認します。リソースグループとディスクデバイスグループ間における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。プロパティの詳しい説明については、`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)` で構成された `preferenced` および `failback` プロパティを使って、ディスクデバイスグループのノードリスト優先度を判別します。手順については、『*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」の「ディスクデバイスのプロパティを変更する」を参照してください。ノード構成の概念やフェイルオーバーおよびスケーラブルデータサービスの動作については、17 ページの「SunPlex 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なければなりません。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 環境では、使用パラメータの柔軟な構成が可能です。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は、`rctladm(1M)` のマニュアルページを参照してください。

Sun Cluster で制御機能を使用する場合は、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。

クラスタ環境では、アプリケーションはリソースの一部として構成され、リソースはリソースグループ (RG) の一部として構成されます。障害が発生すると、対応付けられたアプリケーションと共にリソースグループが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。

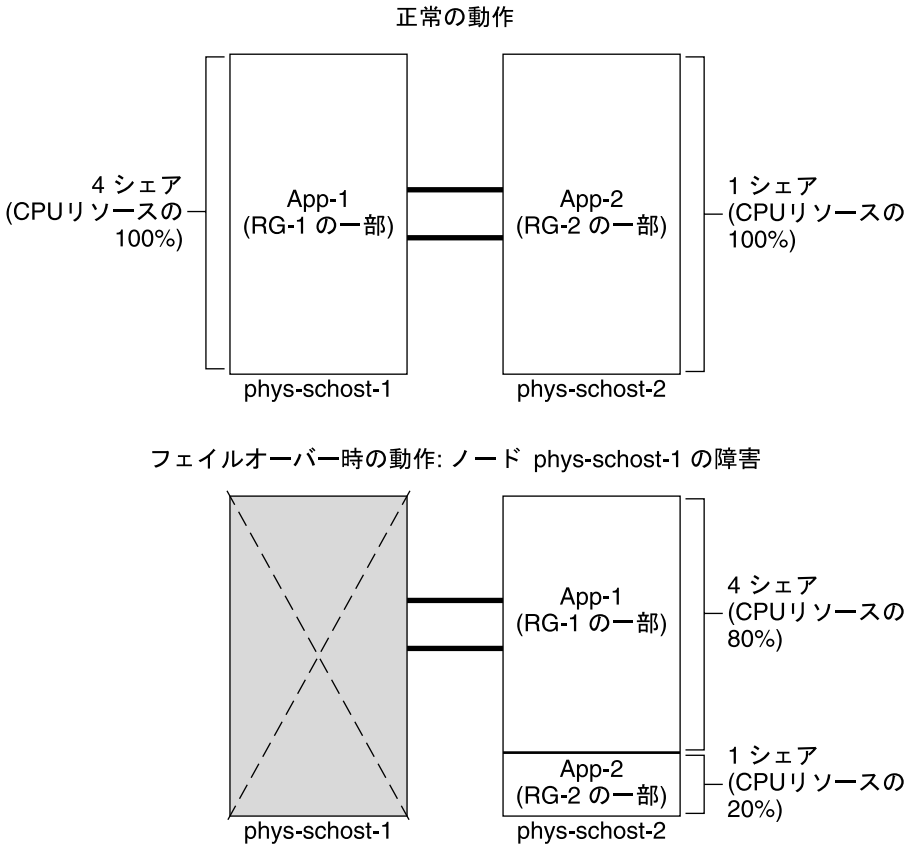
2 つのアプリケーションを供う 2 ノードクラスタ

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なければなりません。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に与えられる CPU 時間の割合は、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し、もう 1 つの物理ホスト (*phys-schost-2*) を他の 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)
```

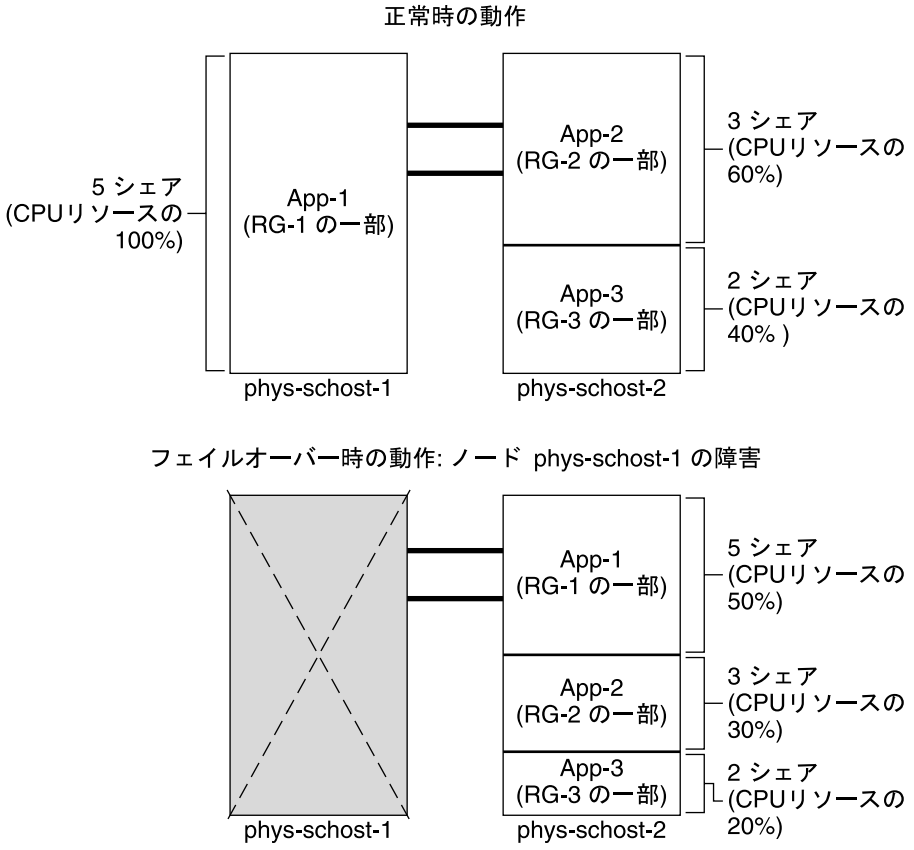
クラスタが正常に動作している間、アプリケーション 1 には、そのデフォルトマスター *phys-schost-1* で 5 シェアが割り当てられます。このノードで CPU 時間を要求するアプリケーションはこのアプリケーションだけであるため、この数は 100 パーセントの CPU 時間と同じことです。アプリケーション 2 と 3 には、それぞれのデフォル

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起り、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

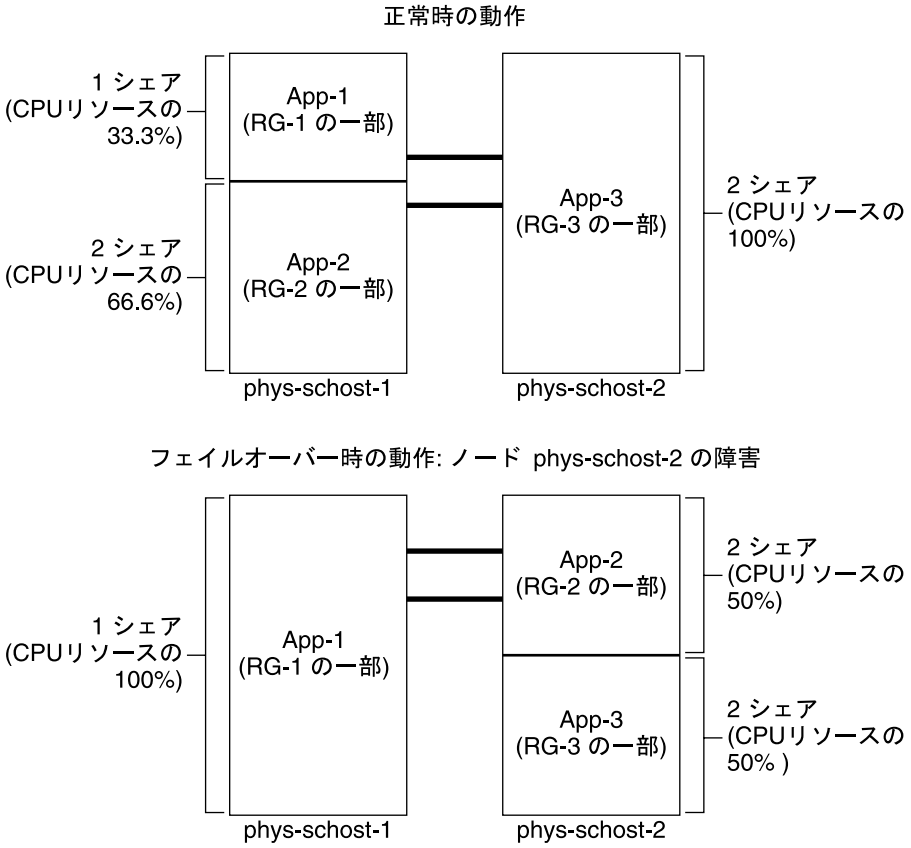
複数のリソースグループが同じデフォルトマスターに属している構成では、1 つのリソースグループ (および、それに関連付けられたアプリケーション) が 二次ノードにフェイルオーバーされたり、スイッチオーバーされることがあります。その間、クラスタのデフォルトマスターは動作を続けます。

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に与えられる 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。

パブリックネットワークアダプタを監視したり、障害の発生時に 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する基本的な機構は、Sun Cluster で動作する Solaris インターネットプロトコル (IP) ソフトウェアが提供します。各クラスタノードには独自の IP ネットワークマルチパス構成があり、これは他のクラスタノードと異なります。

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することができます。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くことができます。`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。これによって、そのノードのパブリックネットワーク接続が維持されます。デーモンは、スタンバイインタフェースが構成されていれば、この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、`in.mpathd` 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行われるため、フェイルオーバー時の一時的な短い遅れを除き、TCP など高レベルの接続への影響はありません。IP アドレスのフェイルオーバーが正常に終了すると、自動的に ARP ブロードキャストが送信されます。したがって、遠隔クライアントへの接続は維持されます。

注 – TCP の構成回復特性が原因で、正常なフェイルオーバーの後、セグメントのいくつかがフェイルオーバー中に失われて、TCP の混雑制御機構をアクティブ化するために、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。

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。論理ホスト名や共有アドレスリソースについて、『*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その障害を覆い隠すことを目的としています。この設計は、`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管理者を回復させることを目的としていません。Sun Cluster ソフトウェアは、論理アドレスや共有 IP アドレスを RGM によって管理されるリソースとみなします。管理者が IP アドレスを追加または削除する正しい方法は、`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するというものです。

IP ネットワークマルチパスが Solaris にどのように実装されているかについては、クラスタにインストールされている Solaris オペレーティング環境のマニュアルを参照してください。

オペレーティング環境のリリース	参照箇所
Solaris 8 オペレーティング環境	『*IP* ネットワークマルチパスの管理』
Solaris 9 オペレーティング環境	『*Solaris* のシステム管理 *(IP* サービス*)*』の「IP ネットワークマルチパス(トピック)」

SPARC: 動的再構成のサポート

Sun Cluster 3.1 4/04 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 4/04 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の休止操作を除く)。したがって、Sun Cluster ソフトウェアで DR 機能を使用する前に、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 Detach 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*』または『*Solaris 9 on Sun Hardware*』コレクション) はどちらも `http://docs.sun.com` で参照できます。

SPARC: 動的再構成の概要

DR 機能では、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、ボードのすべてのコンポーネントに影響を及ぼします。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。したがって、DR のボード切り離し操作はいつ行ってもかまいません。DR サブシステムが、アクティブなコンポーネントを含むボードに対する操作を拒否するからです。

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、そのボードのコンポーネントを使用するには、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“未知のエラー”を報告します。システム管理者は、上位のレベルが報告した“未知のエラー”については無視してください。

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 デバイスに関する DR クラスタリングの考慮点

Sun Cluster ソフトウェアは、CPU デバイスが存在するために DR のボード切り離し操作を拒否することはありません。

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: メモリーに関する DR クラスタリングの考慮点

DR では、メモリーを 2 種類に分けて考える必要があります。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。

オペレーティングシステムが使用するメモリーは、カーネルメモリーケージと呼ばれます。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: ディスクドライブやテープドライブに関連する DR クラスタリングの考慮点

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や二次ノードのドライブの場合だけです。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、84 ページの「SPARC: 定足数デバイスに関連する DR クラスタリングの考慮点」を参照してください。

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「Task Map: 定足数デバイスへの動的再構成」を参照してください。

SPARC: 定足数デバイスに関連する DR クラスタリングの考慮点

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 はこの操作を拒否し、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。

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「Task Map: 定足数デバイスへの動的再構成」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関連する DR クラスタリングの考慮点

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 はこの操作を拒否し、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 操作を行うためには、Sun Cluster 管理ツールを使ってアクティブなインタフェースを無効にする必要があります (下記の注意も参照)。

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。

注意 – Sun Cluster の個々のクラスタノードには、他のすべてのクラスタノードに対する有効なパスが、少なくとも 1 つは存在していなければなりません。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。

SPARC: パブリックネットワークインタフェースに関連する DR クラスタリングの考慮点

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 はこの操作を拒否し、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場合は、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他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上での DR 切り離し操作の詳細な実行方法については、『*Sun Cluster System* 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

頻繁に寄せられる質問 (FAQ)

INDEXTERM-343

この章では、SunPlex 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

- 可用性の高いシステムとは何ですか。

  SunPlex システムでは、高可用性 (HA) を、通常サーバーシステムを使用不能にするような障害が発生した場合でも、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として定義しています。

- クラスタが高可用性を提供するプロセスは何ですか。

  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。

- フェイルオーバーとスケーラブルサービスの違いは何ですか。

  高い可用性を備えたデータサービスには、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があります。

  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、別のノードに処理を引き継いで実行を続けます。

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。

ファイルシステムに関する FAQ

- **1** つまたは複数のクラスタノードを、他のクラスタノードをクライアントとして使用して、高可用性 **NFS** サーバーとして実行できますか。

  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。

- リソースグループマネージャー **(RGM)** によって制御されていないアプリケーションでクラスタファイルシステムを使用できますか。

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

- クラスタファイルシステムは例外なく、**/global** ディレクトリの下にマウントポイントが必要ですか。

  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。

- 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。

  次に示す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、ファイルが、読み取り、書き込み、ファイルロック、非同期入出力のために複数のノードからアクセスされている場合で

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。

- ファイルシステム **/global/.devices/node@<nodeID>** がクラスタノード上にあります。高可用性と広域属性を与えたいデータをこのファイルシステムに格納できますか。

  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もっていますが、広域的にアクセスされることはありません。つまり、個々のノードは、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

- 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。

  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。

- 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を使用することはできますか。

  SPARC: この構成は、ローカルディスクを管理する Solaris Volume Manager ソフトウェアと、多重ホストディスクを管理する VERITAS Volume Manager の組み合わせによってサポートされます。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は Solaris Volume Manager のみサポートされるので、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

- 利用可能な **SunPlex** データサービスは何ですか。

  サポートされているデータサービスのリストは、『*Sun Cluster 3.1 9/04* ご使用にあたって *(Solaris OS* 版*)*』の「サポートされる製品」に含まれています。

- **SunPlex** 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。

  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は、『*Sun Cluster 3.1 9/04* ご使用にあたって *(Solaris OS* 版*)* 』の「サポートされる製品」に含まれています。

- 独自のデータサービスを作成できますか。

  使用できます。詳細は、『*Sun Cluster* データサービス開発ガイド *(Solaris OS* 版*)* 』の「データサービス開発ライブラリのリファレンス」を参照してください。

- ネットワークリソースを作成する場合に、**IP** アドレスで指定するのですか。またはホスト名で指定するのですか。

  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。

- 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。

  Sun Cluster HA for NFS の場合を除き、フェイルオーバーモードリソースグループの論理ホスト名リソースを使用するときは、共有アドレスリソースと論理ホスト名リソースは同様に使用できます。共有アドレスリソースを使用すると、クラスタネットワーキングソフトウェアが論理ホスト名ではなく、共有アドレスに合わせて構成されているために、多少のオーバーヘッドが生じます。

  共有アドレスを使用する利点は、スケーラブルおよびファイルオーバーの両方のデータサービスを構成していて、クライアントが同じホスト名を使用して両方のサービスにアクセスできるようにする場合に得られます。この場合、共有アドレス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含まれますが、スケーラブルサービスリソースは個別のリソースグループに含まれ、共有アドレスを使用するように構成されます。スケーラブルおよびフェイルオーバーサービスはどちらも、共有アドレスリソースに構成された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

- **SunPlex** 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。

  現在、SunPlex 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。

- フェイルオーバーでの **MAC** アドレスの役割は何ですか。

  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。

- **SunPlex** システムは `local-mac-address?=true`の設定をサポートしますか。

  使用できます。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?` は、SPARC ベースのクラスタでは OpenBoot PROM `ok` プロンプトから `eeprom`(1M) で、または x86 ベースのクラスタでは BIOS のブート後、任意で実行する SCSI ユーティリティで設定できます。

- **IP Network Multipathing**がアダプタの切り替えを行う際に、どの程度の遅延がありますか。

  この遅延は数分に及ぶことがあります。これは、IP Network Multipathing スイッチオーバーが行われる際に余分な ARP を送信する必要があるためです。ただし、クライアントとクラスタ間のルーターが、この余分な ARP を必ず使用するとは限りません。したがって、ルータ上のこの 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、古い MAC アドレスを使用できる可能性があります。

- 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。

  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

- 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。

  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。

- ノードが起動される順序は重要ですか。

  ほとんどの場合、重要ではありません。ただし、起動順序は、amnesia (詳細は、49 ページの「障害による影響の防止について」を参照) を防止するために重要です。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。これにより、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。

- 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。

  使用できます。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。

- クラスタメンバーのバックアップの注意点は何ですか。

  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、1 つのノードをテープドライブまたはライブラリが接続された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「クラスタのバックアップと復元」を参照してください。

- 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。

  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。

クラスタ記憶装置に関する FAQ

- 多重ホスト記憶装置の可用性を高めるものは何ですか。

  多重ホスト記憶装置は、単一のディスクが失われてもミラー化 (またはハードウェアベースの RAID-5 コントローラ) のために存続できるので、可用性が高くなります。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

- **SunPlex** 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。

現在、SunPlex システムは SPARC ベースと x86 ベースの両方のクラスタで、Ethernet (100BASE-T Fast Ethernet と 1000BASE-SX Gb) クラスタインターコネクトをサポートします。SunPlex システムが SCI ネットワークインタフェースクラスタインターコネクトをサポートするのは、SPARC ベースのクラスタに限定されます。

- 「ケーブル」とトランスポート「パス」の違いは何ですか。

  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。ケーブルを検査することはできないため、そのステータスを知ることはできません。ケーブルの状態を見るには `scconf -p` 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表示するには、`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

- 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。

  クライアントシステムは、他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

アント側の構成条件の詳細については、『*Sun Cluster* データサービスの計画と管理』を参照してください。

管理コンソールに関する FAQ

- **SunPlex** システムには管理コンソールが必要ですか。

  使用できます。

- 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。

  SunPlex システムは、専用の管理コンソールを必要としません。ただし、専用のコンソールを使用すると、次の利点が得られます。

  - 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。

- 管理コンソールをクラスタの近く、たとえば同じ部屋に配置する必要がありますか。

  ハードウェアの保守担当者に確認してください。保守作業上、コンソールをクラスタの近くに配置する必要がある場合があります。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。

- 距離の条件すべてが合致するかぎり、**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。

  使用できます。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

- **SunPlex** システムは端末集配信装置を必要としますか。

  Sun Cluster 3.0 以降のすべてのソフトウェアリリースの実行には、端末集配信装置は必要はありません。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、Sun Cluster 3.0 以降の製品では端末集配信装置に依存しません。

- ほとんどの **SunPlex** サーバーは端末集配信装置を使用していますが、**Sun Enterprise E10000 server** が使用していないのはなぜですか。

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。そのコンソールポートはシリアルポートです。Sun Enterprise E10000 server サーバーには、シリアルコンソールがありません。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 E10000 server サーバーの場合は、コンソールに対して常に SSP を使用します。

- 端末集配信装置を使用した場合の利点は何ですか。

  端末集配信装置を使用すると、ノードが SPARC ベースノード上の OpenBoot PROM (OBP) にあろうと、x86 ベースノードのブートサブシステムにあろうと、ネットワーク上の任意の場所の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。

- **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。

  Sun がサポートする端末集配信装置と他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、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ぐ特殊なファームウェアがあるという点です。ブレーク、またはコンソールに対してブレークと解釈されることがある信号を送信するコンソールデバイスの場合、ノードが停止されることに注意してください。

- **Sun** がサポートする端末集配信装置を再起動しないで、そこにあるロックされたポートを開放できますか。

  使用できます。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の構成と管理の詳細は、以下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.x Hardware Administration Manual for Solaris OS*』の「Installing and Configuring the Terminal Concentrator」

- 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。

  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。

- 端末集配信装置を使用する場合に、セキュリティはどのように制御しますか。

  通常、端末集配信装置は、他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を制御できます。

- **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。
  - 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、83 ページの「SPARC: ディスクドライブやテープドライブに関連する DR クラスタリングの考慮点」 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

け

こ

さ

し

す

そ

た

ま

み

め

よ

り

ろ