FUJITSU

FUJITSU Software
Interstage Application Server V12.0.0

高信頼性システム運用ガイド

Windows/Solaris(64)/Linux

B1WS-1314-01Z0(00)
2017年7月

まえがき

本書の目的

本書は、Interstage Application Serverの高信頼サービスについて説明しています。
本書は、Interstage Application Serverの運用を行う方を対象に記述されています。

前提知識

本書を読む場合、以下の知識が必要です。

- インターネットに関する基本的な知識
- オブジェクト指向技術に関する基本的な知識
- 分散オブジェクト技術(CORBA)に関する基本的な知識
- リレーショナルデータベースに関する基本的な知識
- 使用するOSに関する基本的な知識

本書の構成

本書は、以下の構成になっています。

第1章 高性能・高信頼システムのための機能

高性能・高信頼のシステムのために実現している機能を説明しています。

第2章 高信頼化システムの設計

高信頼化システムの概要および高信頼化システムの特徴を説明しています。

第3章 IPCOMを利用した負荷分散

IPCOMのIIOP負荷分散を利用した負荷分散システムの運用設計について説明しています。

第4章 クラスタサービス機能

クラスタサービスの環境設定手順について説明しています。

付録A クラスタサービスの互換機能

旧バージョンでのクラスタ環境の構築手順について説明しています。

付録B ロードバランス機能を利用した場合の設計

ロードバランス機能を利用した複数サーバの運用、高信頼化システムの設計について説明しています。

輸出許可

本ドキュメントを輸出または第三者へ提供する場合は、お客様が居住する国および米国輸出管理関連法規等の規制をご確認のうえ、必要な手続きをおとりください。

登録商標について

記載されている会社名、製品名などの固有名詞は、各社の商標または登録商標です。

本製品のマニュアルに記載されている他社製品の商標表示については、「マニュアル体系と読み方」の「マニュアルの読み方」－「登録商標について」を参照してください。

著作権

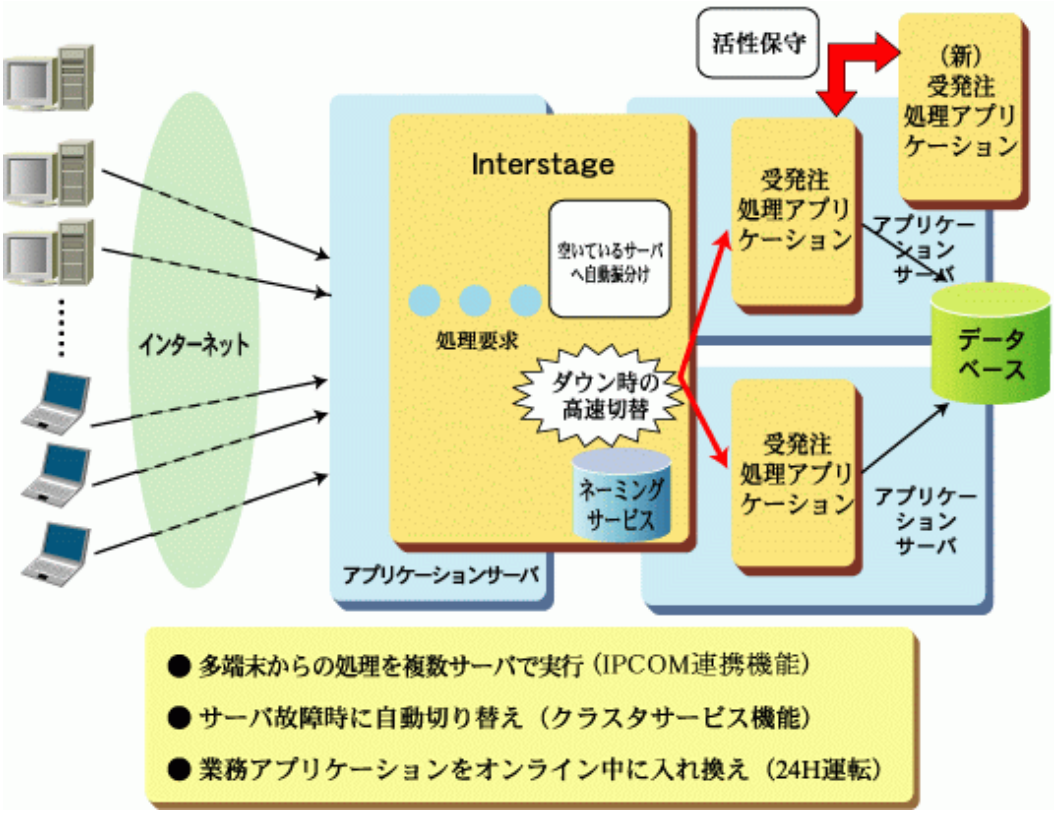
2017年7月 初版

目　次

第1章 高性能・高信頼システムのための機能

基幹サーバ構築で最も重要なことは、信頼性の確保と、スケーラビリティの確保です。
Interstageはこれらの前提機能となるノード管理、オブジェクト管理、アプリケーション管理、およびキュー管理機能は従来から提供してきましたが、さらに大規模・高信頼システムへの適用性を向上すべくInterstage Application Server Enterprise Editionで以下の機能を提供します。

- IPCOM連携機能
- クラスタサービス機能
- 動的アプリケーション入れ替え
- 動的プロセス数変更



1.1 IPCOM連携機能

IPCOMと連携することにより、サーバマシンの負荷分散を行うことができます。

注意

IPCOMでIIOPの負荷分散を行うには、IIOP負荷分散に対応した機種を選択する必要があります。IIOP負荷分散に対応しているIPCOMの機種については「IPCOMのマニュアル」を参照してください。

負荷分散対応プロトコル

Interstageの通信プロトコルには以下の2つがあります。

- HTTPプロトコル
- IIOPプロトコル

IPCOMとの連携においては、これらの2つの通信プロトコルに対応しています。
ここでは、IIOPの負荷分散について記述しています。HTTPの負荷分散については、「IPCOMのマニュアル」で説明されています。

負荷分散ポリシー

以下の負荷分散ポリシーにより負荷分散を行うことができます。

IPCOM標準の負荷分散ポリシー

- ラウンドロビン方式
- 負荷計測(CPU、メモリ、ディスクI/O)方式
- 負荷計測(コネクション数、レスポンス時間)方式
- 負荷計測(データ通信量)方式
- 静的重み付け方式

IPCOM標準の負荷分散ポリシーについては、「IPCOMのマニュアル」で説明されています。

注意

以下のタイプのIJServerワークユニットに配備されたEJBアプリケーションに対する負荷分散が可能です。

- EJBアプリケーションのみ運用

Interstage特有の負荷分散ポリシー(IPCOM連携機能) Windows32 Linux32

- 待ちメッセージ数
- 通信バッファ使用率

Interstage特有の負荷分散ポリシーは、ワークユニットとして運用するアプリケーションに対して適用することができます。以下に、適用できるアプリケーションを説明します。

	CORBAアプリケーション	トランザクションアプリケーション	EJBアプリケーション
待ちメッセージ数	○	○	×
通信バッファ使用率	×	○	×

○:適用できます　×:適用できません

待ちメッセージ数 Windows32 Linux32

ワークユニットとして運用する以下のアプリケーションに対して適用できます。待ちメッセージの滞留キュー数を監視します。

- CORBAアプリケーション

  CORBAアプリケーションのインプリメンテーションリポジトリに対する待ちメッセージ数を監視します。

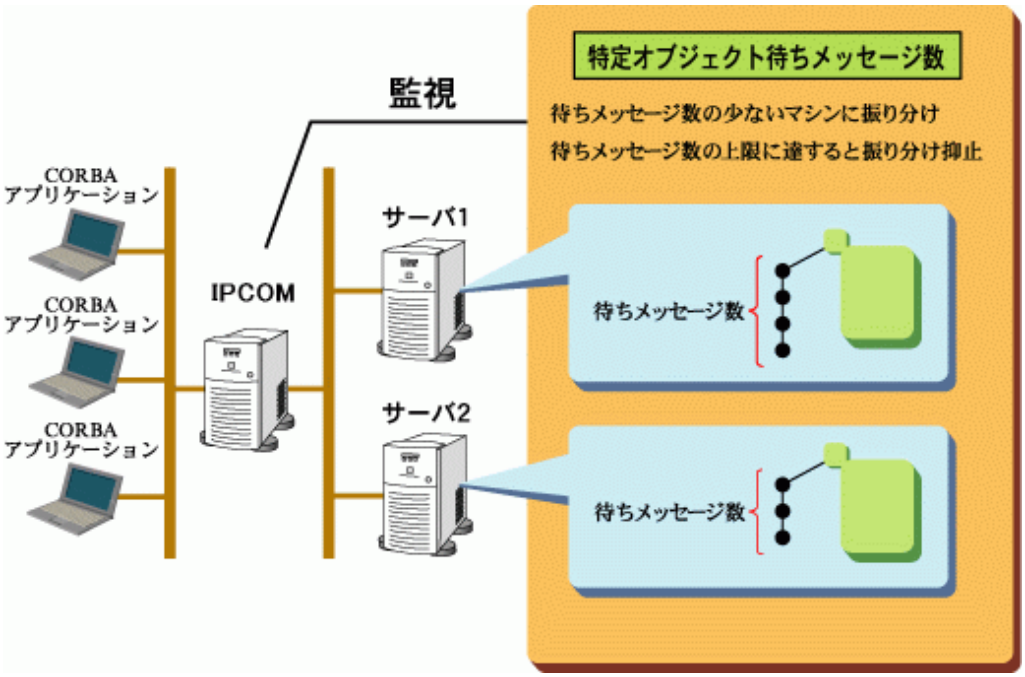
- トランザクションアプリケーション

  トランザクションアプリケーションのオブジェクトに対する待ちメッセージ数を監視します。

注意

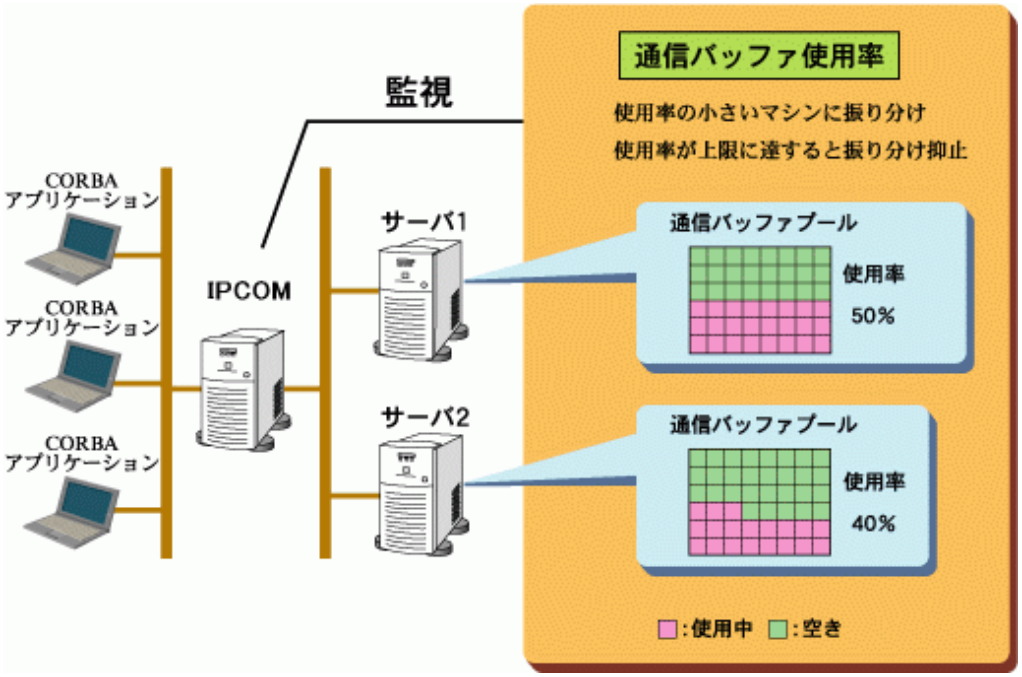
EJBアプリケーションの場合、IJServerワークユニットに配備されたEJBアプリケーションは監視できません。

特定オブジェクトでクライアントアプリケーションからの要求を受け付け、その要求をほかのオブジェクトへ分配するような形態で業務を運用する場合で、その特定のオブジェクトへの滞留キュー数で負荷状態を判断できる場合に有効です。



通信バッファ使用率 Windows32 Linux32

トランザクションアプリケーションに対する要求の通信バッファの使用率です。
本使用率により、サーバ内の全トランザクションアプリケーションに対する通信負荷状況を監視できます。クライアントアプリケーションからの要求を、複数のオブジェクトが受けつけるような業務を運用する場合に有効です。



振り分け上限に到達時の動作

負荷計測項目値が振り分け上限値に到達した場合、そのサーバへの振り分けが行われなくなります。
すべての振り分け対象サーバの負荷計測項目値が振り分け上限に達した場合、それ以降、サーバへの振り分け処理は行われなくなります。
この場合、クライアントからの要求は、通信エラー(COMM_FAILURE)で復帰します。
負荷計測項目値が復旧値を下回った時、そのサーバへの振り分けが再開されます。

ワークユニットに対する負荷分散

ワークユニットとして運用するアプリケーションに対する負荷分散では、アプリケーション単位に、以下の2種類の方式を選択できます。
ただし、IJServerワークユニットのEJBアプリケーションに対する負荷分散においては、メソッド呼出し単位の負荷分散のみ実施することができます。

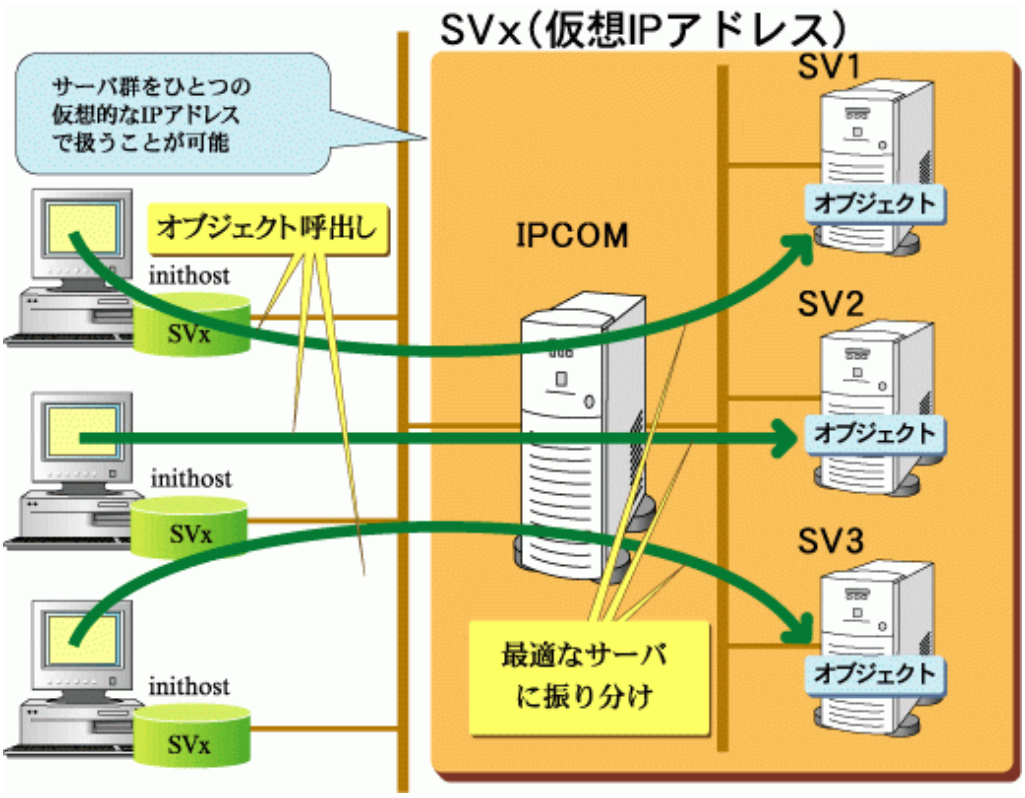
- メソッド呼び出し単位の負荷分散

  メソッド呼び出しごとに、負荷分散を行います。
  一問一答形式の通信を行う場合に使用できます。
  EJBアプリケーションの場合、IJServerワークユニットを運用するサーバである負荷分散対象サーバと、負荷分散対象サーバを管理する負荷分散サーバが必要になります(負荷分散が可能なIJServerワークユニットのタイプは、「負荷分散ポリシー」を参照してください)。
  なお、EJBアプリケーションはHomeインタフェースのメソッド(createメソッドなど)実行時に負荷分散されます。Remoteインタフェースのメソッド実行時は同一のサーバにアクセスされるため、最適なタイミングでHomeインタフェースのメソッドを実行して負荷分散してください。

- ネーミングサービスのオブジェクトリファレンス獲得時点の負荷分散

  ネーミングサービスのオブジェクトリファレンスを獲得する時点で負荷分散を行います。本方式では、ネーミングサービスのオブジェクトリファレンス獲得要求が振り分けられたサーバに対して、その後の通信が行われます。
  同一クライアントからの一連の要求を、同一サーバに振り分けたい場合に使用します。
  セション継続のような運用が可能です。

ワークユニットに対する負荷分散においては、任意の監視対象ワークユニットが停止した時に、そのサーバを負荷分散対象サーバから切り離した縮退運用が行えます。
詳細については、「第3章 IPCOMを利用した負荷分散」を参照してください。



注意

負荷分散対象となるサーバマシンにはネーミングサービスが配置されている必要があります。

1.2 クラスタサービス機能

業務運用中にハードウェア障害やソフトウェア障害など不測の事態が発生して、システムダウンなどに陥った場合に備え、クラスタサービス機能を使用することでシステムの高信頼化を実現できます。
クラスタサービス機能での引き継ぎ処理は、回線を引き継ぐこと(IPアドレス引き継ぎ)ができます。さらに、システムを構成する様々なハードウェアやソフトウェアにも対応しています。

Interstageでは、1台または複数台の運用中のサーバマシン(運用ノード)とは別に、万一の場合に備えて運用待機しているサーバ(待機ノードと呼びます)を用意します。また、運用ノードと待機ノードの共用ディスク上にデータを配置します。
ハード異常などで運用ノードがダウンした場合には、待機ノードに業務を引き継ぎます。この際、IPアドレスも引き継がれます。また、待機ノードから共用ディスクへアクセスすることで運用ノードからのデータの引継ぎが行えます。これにより、業務全体を停止させることなく運用を継続できるようにする機能をクラスタサービス機能といいます。

クラスタシステム

クラスタサービス機能は、Interstage Application Server Enterprise Editionで提供されるものであり、下記のクラスタシステム上で動作します。

Windows32 Windows64

- Microsoft Corporationのクラスタシステムとして使用されているWindows Server(R)のフェールオーバー クラスタリング(WSFC)。

Solaris64

- 富士通のクラスタシステムであるPRIMECLUSTER

Linux32 Linux64

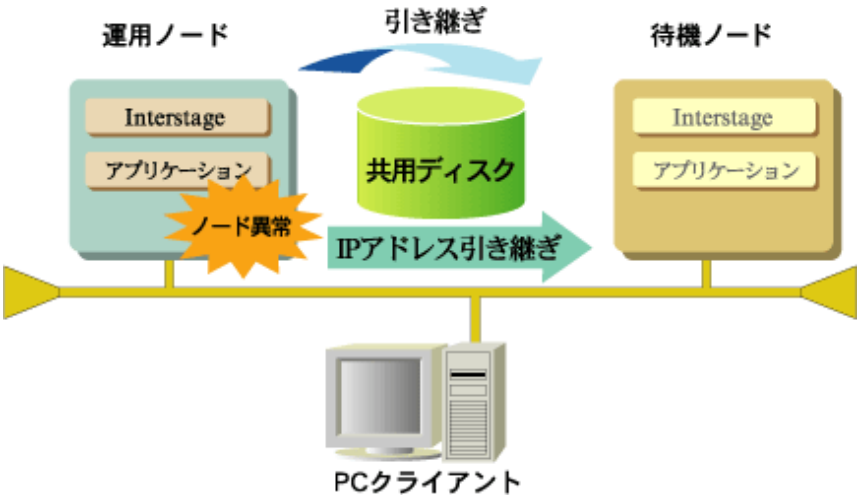
- 富士通のクラスタシステムであるPRIMECLUSTER

クラスタサービス機能を活用するためには、上記に対する知識が必要となります。詳細はクラスタシステムのマニュアルを参照してください。

運用形態

運用マシンと待機マシンについて、以下の形態を用意しています。

1:1運用待機



運用ノードが障害時に待機ノードで業務を自動的に引き継ぎます。

適用対象サーバ

クラスタサービス機能は、Interstage Application Serverのアプリケーションサーバ機能を導入しているサーバに対して使用できます。

クラスタシステム上で使用できるサービス

Interstageが提供するサービス群のうち、クラスタシステム上で使用できるサービス/使用できないサービスを以下の一覧表に示します。

サービス名	サービスの利用可否				
	Windows32	Windows64	Solaris64	Linux32	Linux64
Interstage HTTP Server	○	○	○	○	○
Interstage HTTP Server 2.2	○	○	○	○	○
Servletサービス	○	○	○	○	○
EJBサービス	○	○	○	○	○
Interstage JMS	○	○	○	○	○

CORBAサービス (ObjectDirector)	○	○	○	○	○
イベントサービス	○	○	○	○	○
コンポーネントトランザクションサービス(TransactionDirector)	○	－	－	○	－
データベース連携サービス (ObjectTransactionService)	○	○	○	○	○
MessageQueueDirector	○	○	－	○	○
Interstage シングル・サインオン	○	○	○	○	○
Interstage ディレクトリサービス	○	○	○	○	○
Java EE 6	×	×	×	×	×
Java EE 7	○	○	○	○	○

○:使用できます。
×:使用できません。
－:機能を提供していません。

注意

Solaris64

global zoneでクラスタサービス機能を使用する場合、non-global zoneにInterstage Application Serverをインストールしないでください。また、non-global zoneでクラスタサービス機能を使用する場合は、global zoneにInterstage Application Serverをインストールしないでください。

1.3 動的アプリケーション入れ替え

Interstageでは、活性変更機能により、業務システムを停止することなく、業務アプリケーションの入れ替えや設定変更を行うことができます。これにより、24時間、365日のシステム連続運転を実現できます。
業務アプリケーションに加えて、環境変数などの動作環境も運用中に変更できます。

以下の実行環境で、運用中の業務アプリケーションの入れ替えをサポートしています。

- IJServerクラスタ
- CORBAワークユニット
- IJServerワークユニット
- トランザクションアプリケーションのワークユニット

また、以下の実行環境で、運用中の動作環境の変更をサポートしています。

- IJServerクラスタ
- CORBAワークユニット
- トランザクションアプリケーションのワークユニット

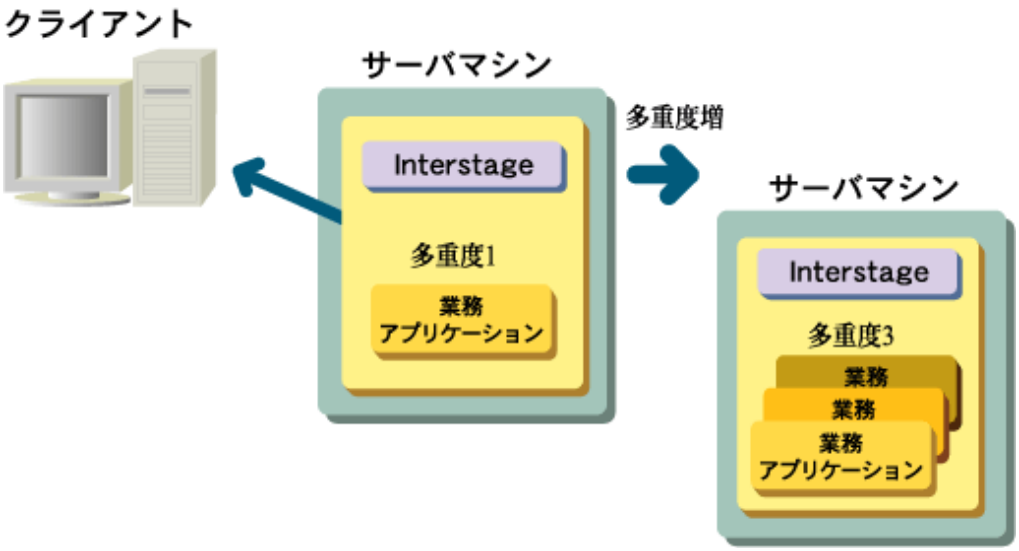
参照

- IJServerクラスタの業務アプリケーション入れ替え、設定変更については、「Java EE 7 設計・構築・運用ガイド」の「Java EEアプリケーションの運用」を参照してください。

- CORBAワークユニットとトランザクションアプリケーションのワークユニットの活性変更機能については、「OLTPサーバ運用ガイド」を参照してください。
- IJServerワークユニットの活性変更機能については、「J2EE ユーザーズガイド(旧版互換)」を参照してください。

1.4 動的プロセス数変更

業務運用では、一日の運用時間帯の中で、業務アプリケーションへの負荷がかかる時間帯とそうでない時間帯があります。通常は、業務アプリケーションへの負荷がかかり、システムへの過負担のため、一時的に業務システム利用者への応答時間が遅延するなどがみられます。
Interstageでは、上記のような状況に対し、業務アプリケーションのプロセス多重度を動的に変更可能とする機能を提供します。負荷が高くなった場合には該当の業務アプリケーションのプロセス多重度を上げ、負荷が低くなった場合には該当する業務アプリケーションのプロセス多重度を下げることで、処理能力を動的に変更して、利用者への応答時間を保証できるようになりました。



第2章 高信頼化システムの設計

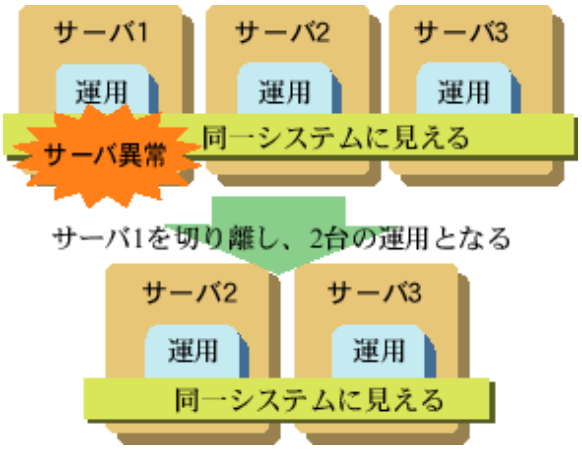
Interstageシステムを高信頼化するためには、以下の3つのパターンがあります。

- IPCOMによる縮退を利用した高信頼化システム
- クラスタサービス機能を利用した高信頼化システム
- Windows32/64 Solaris64
  両現用縮退方式を利用した高信頼化システム

2.1 IPCOMによる縮退を利用した高信頼化システム

高信頼化の方式

複数台のサーバを同一システムとして処理を並列に行い、クライアントからの処理を自動的に分散させます。サーバ異常時はサーバを切り離して、残りのサーバで運用を行います。
また、IPCOMでは、任意のワークユニットの異常終了時にも同様の運用が行えます。



高信頼化の目的

サーバ間の負荷調整を自動的に行いたい。

特徴

複数のサーバを1つのシステムとして使用できるため、業務に対する負荷を分散させることができます。そのため、大規模なシステムを構築する場合に適しています。
サーバマシン異常に対する高信頼化を構築可能です。クライアントからの負荷を自動的に分散させます。また、サーバ追加を容易に行うことができます。IPCOMに関しては、二重化構成とし、高信頼化することを推奨します。

業務復旧までの時間

DB共用を使用していない場合は、ノードダウン時も業務が停止することはありません。
DB共用の場合は、ダウンしたサーバで使用していたテーブルに関しては、復旧まで約数十秒必要です。ただし、トランザクション数によってはさらに時間がかかる場合もあります。

運用方法

IPCOMを使用した負荷分散システムを構築するためには、複数台のサーバに対してIPCOMを1台(二重化構成の場合は2台)用意します。IPCOMが監視を行い、ノードダウン時は縮退を行います。
また、IPCOMが停止した場合、システム全体に影響がありますので、IPCOMを二重化することを推奨します。IPCOMの二重化については、「IPCOMのマニュアル」を参照してください。

システム構築における選択肢

IPCOMによる縮退を利用した高信頼化システムを構築する場合、以下の選択肢があります。

DB使用形態

DBを使用しないDB未使用パターンと、ノード間でDBを共用しないDB非共用パターン、ノード間でDBを共用するDB共用パターンが選択できます。

使用DBMS製品

DB使用形態にあったDBMS製品を選択します。

以下に組み合わせを示します。

DB使用形態	DB未使用	DB非共用	DB共用
使用DBMS	ー	DBMS製品	ノード間でのDB共用に対応したDBMS製品
パターン	パターン1	パターン2	パターン3

注意

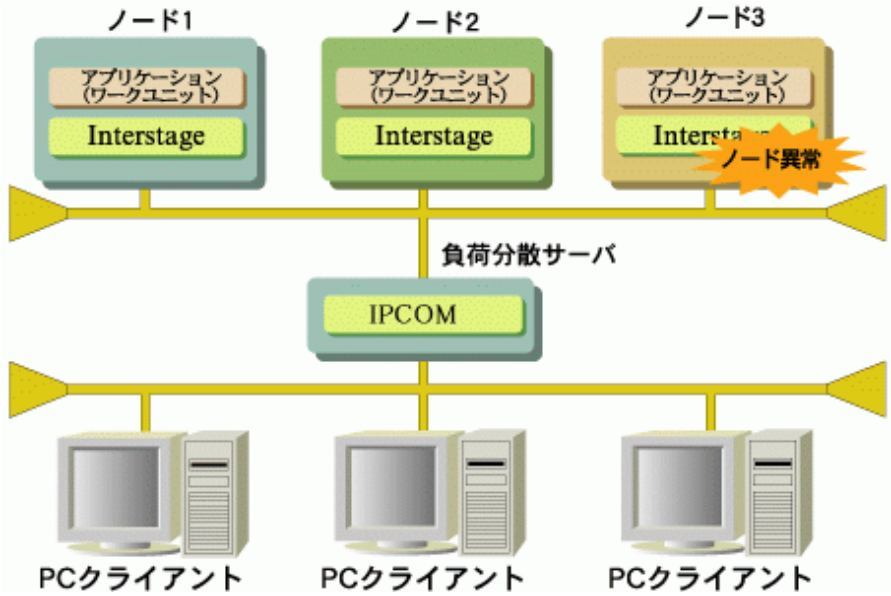
DB共用型については、クラスタシステムが必要となります。

必須製品

- Interstage Application Server Enterprise Edition
- IPCOM
  IIOP負荷分散に対応したIPCOMが必要です。IIOP負荷分散の詳細については「IPCOMのマニュアル」を参照してください。
- DBMS製品
  DB未使用時は不要です。また、DB共用の場合は、ノード間でのDB共用に対応したDBMS製品が必要です。
- 使用するクラスタシステム製品と、クラスタシステム製品が必要とする製品
  DB共用型の場合のみ、必要です。

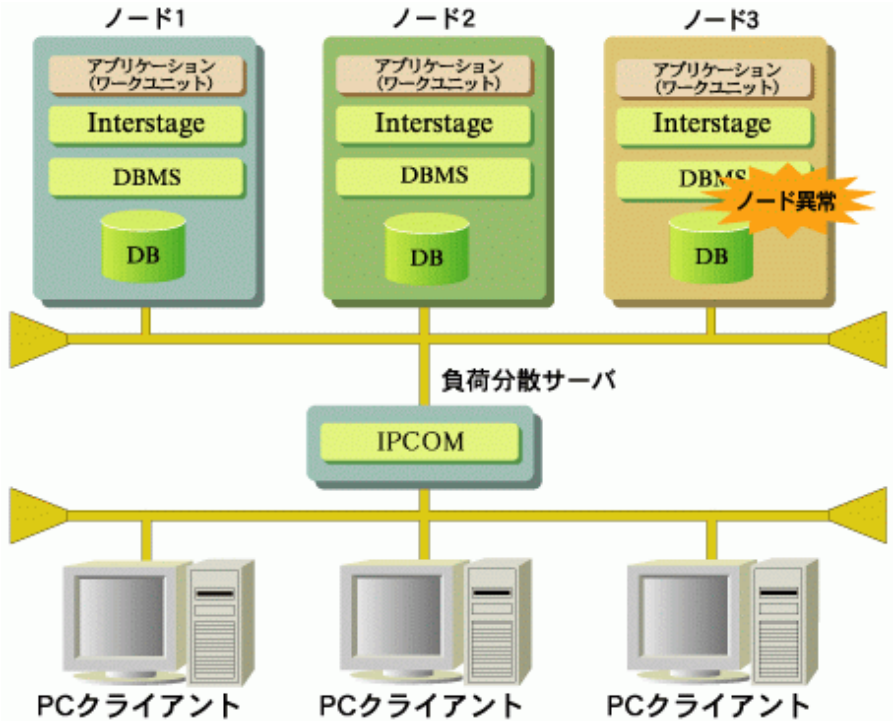
パターン1 (DB未使用)の構成図

以下に、IPCOMによる縮退で、DBMSを使用しない場合を示します。



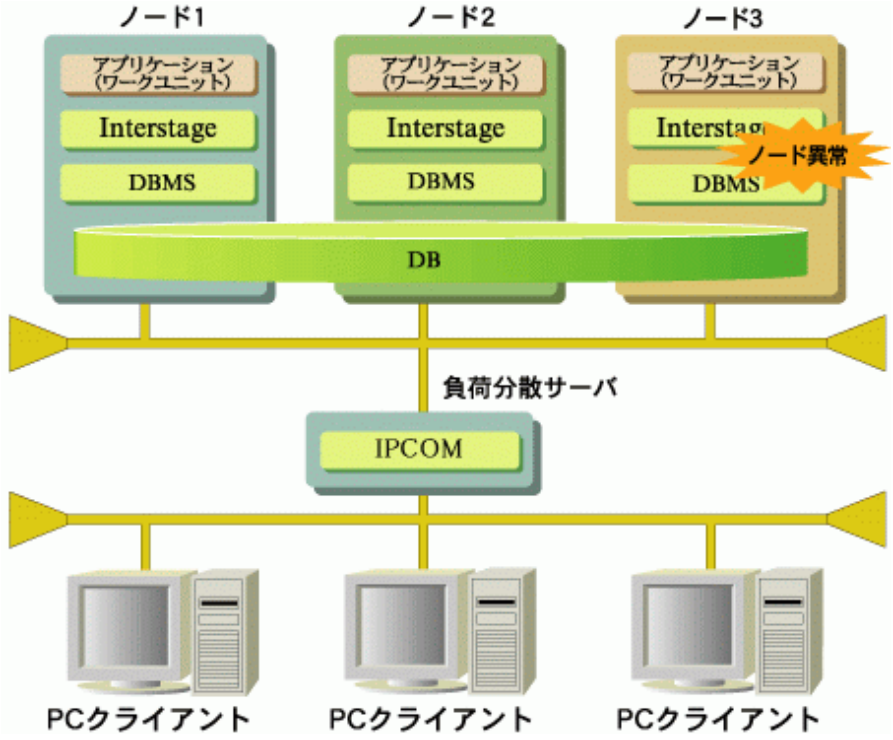
パターン2 (DB非共用)の構成図

以下に、IPCOMによる縮退で、DBMSを共用しない(ノード間で非共用)場合を示します。



パターン3 (DB共用)の構成図

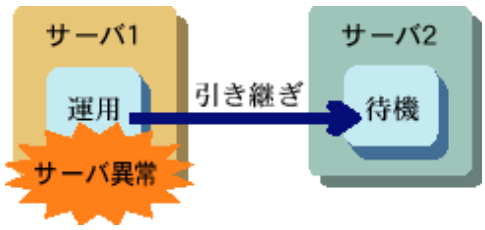
以下に、IPCOMによる縮退で、DBMSを共用する(ノード間で共用)場合を示します。



2.2 クラスタサービス機能を利用した高信頼化システム

高信頼化の方式

業務運用中のサーバに対して、同様の環境を備えたサーバを待機として準備し、業務運用中のサーバが故障時に待機していたサーバで業務を引き継ぎます。



高信頼化の目的

システム全体を堅牢化したい。

特徴

サーバマシン異常以外にシステムループやInterstageダウンなどの監視が可能で、システム全体の高信頼化を行うことができます。ただし、クラスタシステムの環境が必要になるため、通常よりも設備費が多くかかります。

業務復旧までの時間

クラスタサービス機能を利用しない場合、システムに何らかの異常(ハードウェアの故障など)が発生した場合、一般的には業務復旧まで2～3時間ほどの時間が必要ですが、クラスタサービス機能を利用した場合、およそ以下の時間で業務を復旧することができます。

Windows32/64

約2～5分

Solaris64

約10～15分

Linux32/64

約10～15分

注意

業務復旧までの時間は、クラスタサービス機能に対する設計内容に依存します。このため、業務復旧までの正確な時間を確認する場合には実機での確認を行ってください。

運用方法

クラスタサービス機能を使用する場合、クラスタシステムが必要です。クラスタシステムとは、システムを構成するノードを監視して、異常を検出すると他のノードに引き継ぐ仕組みを提供するシステムです。クラスタサービス機能では、クラスタシステムの機能を利用し、高信頼化システムを構築します。

システム構成設計

クラスタサービス機能を利用した高信頼化システムを構築する場合、以下の設計を行います。

使用クラスタシステム

クラスタサービス機能を使用する場合、利用できるクラスタシステムは以下のとおりです。

Windows32/64

WSFCだけ使用可能です。

Solaris64 Linux32/64

PRIMECLUSTERだけ使用可能です。

クラスタ形態

1:1運用待機のみ使用可能です。

使用DBMS製品

DBMS製品を使用する場合は、クラスタシステムに対応したDBMS製品を選択します。

以下に組み合わせを示します。

使用クラスタ製品	PRIMECLUSTER	WSFC
クラスタ形態	1:1運用待機	1:1運用待機
使用DBMS	クラスタシステムに対応したDBMS製品	

注意

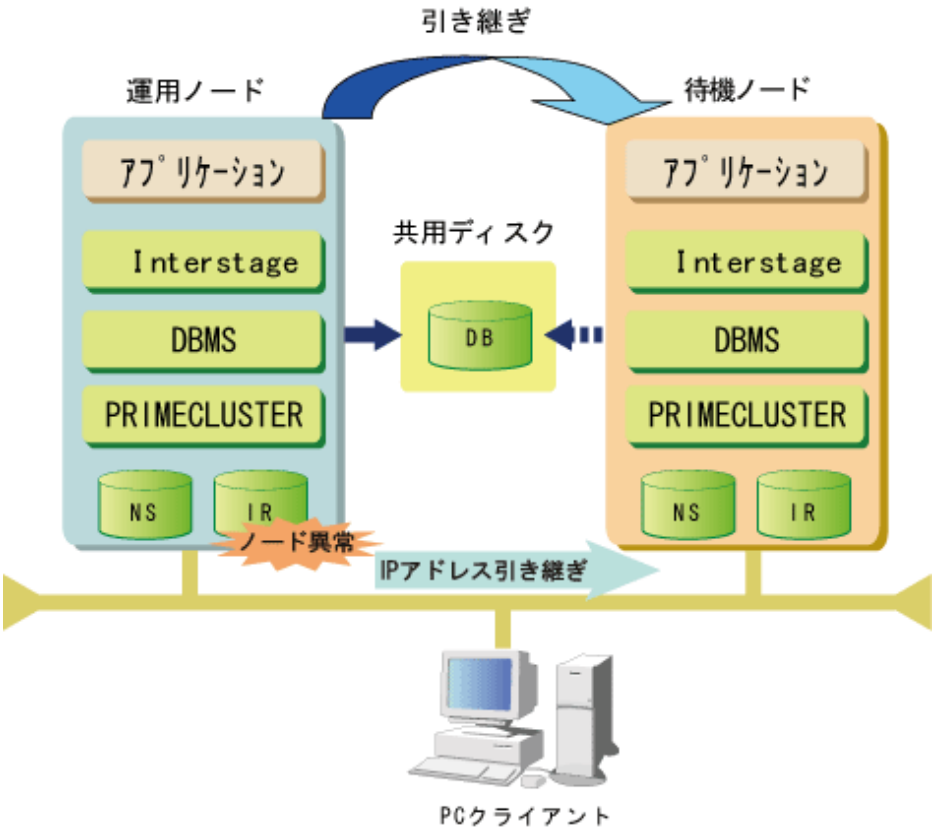
ノード名引き継ぎ(PRIMECLUSTERの機能)は使用できませんので注意してください。

必須製品

- 使用するクラスタシステム製品と、クラスタシステム製品が必要とする製品
  クラスタシステムについては、「第4章 クラスタサービス機能」を参照してください。
- クラスタシステムに対応したDBMS製品

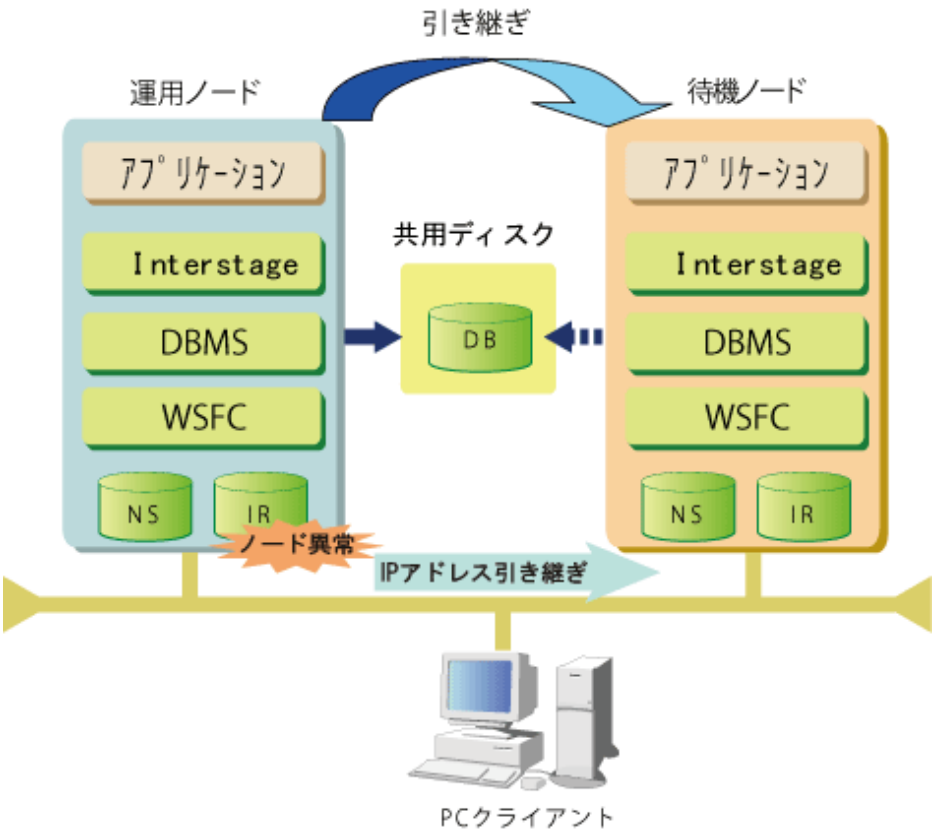
パターン1 (PRIMECLUSTER +1:1運用待機 + DBMS)の構成図

以下に、使用クラスタシステムに当社クラスタ製品であるPRIMECLUSTERの1:1 運用待機、DBMSを組み合わせた場合を示します。
データベース資源を共用ディスクに設定することにより、切り替え時にデータの引き継ぎを行います。ネーミングサービス、インタフェースリポジトリはローカルディスクに設定します。



パターン2 (WSFC + 1:1運用待機 + DBMS)の構成図

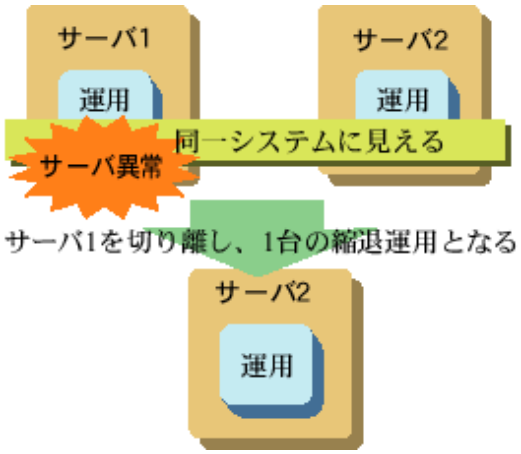
以下に、使用クラスタシステムにMicrosoft Corporationのクラスタ製品であるWSFCと、DBMSとを組み合わせた場合の、クラスタサービス機能について説明します。
データベース資源を共用ディスクに設定することにより、切り替え時にデータの引き継ぎを行います。ネーミングサービス、インタフェースリポジトリはローカルディスクに設定します。



2.3 両現用縮退方式を利用した高信頼化システム

高信頼化の方式

複数台のサーバを同一システムとして処理を並行に行い、各クライアントで事前に使用サーバの優先度を設定しておくことで静的に負荷を分散させます。サーバ異常時はサーバを切り離して、残りのサーバで運用を行います。



高信頼化の目的

簡単にシステムの信頼性を向上したい。

特徴

サーバマシン異常に対する高信頼化を構築可能。クライアントから事前に定義しておくことで負荷を分散させます。特別な設備を必要としません。
クライアントから事前に負荷を分散させるため、小/中規模のシステムに適しています。
静的に負荷を分散させるため、均一な負荷のバランスは期待できません。そのため、業務ごとに主として動作するサーバを決めておける、相互待機形態としてのシステムを構築する場合に適しています。

ポイント

Windows32 Linux32
負荷のバランスを均一にしたい場合は、ロードバランスを利用した高信頼化システムを構築してください。

業務復旧までの時間

DB共用を使用していない場合は、ノードダウン時も業務が停止することはありません。
DB共用の場合は、ダウンしたサーバで使用していたテーブルに関しては、復旧まで約数十秒必要です。ただし、トランザクション数によってはさらに時間がかかる場合もあります。

注意

業務復旧までの時間は、クラスタサービス機能に対する設計内容に依存します。このため、業務復旧までの正確な時間を確認する場合には実機での確認を行ってください。

運用方法

各ノードにネーミングサービスおよびインタフェースリポジトリを設定します。
ノードダウン時は残りの1台で運用を行います。

システム構築における選択肢

両現用縮退方式による縮退を利用した高信頼化システムを構築する場合、以下の選択肢があります。

DB使用形態

DBを使用しないDB未使用パターンと、ノード間でDBを共用しないDB非共用パターン、ノード間でDBを共用するDB共用パターンが選択できます。

使用DBMS製品

DB使用形態にあったDBMS製品を選択します。

以下に組み合わせを示します。

DB使用形態	DB未使用	DB非共用	DB共用
使用DBMS	－	DBMS製品	ノード間でのDB共用に対応したDBMS製品
パターン	パターン1	パターン2	パターン3

注意

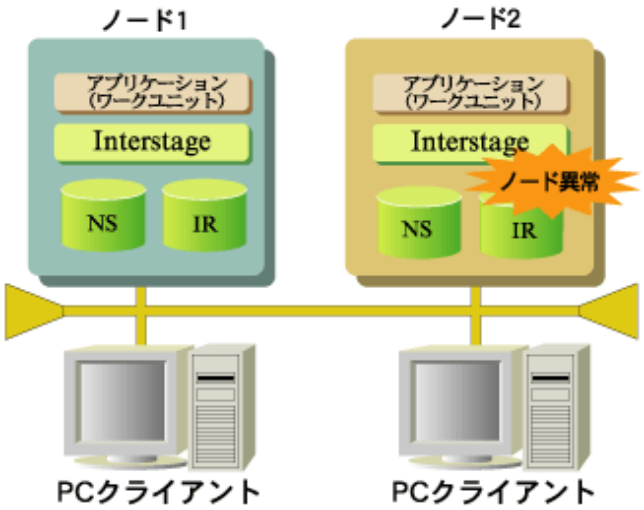
DB共用型については、クラスタシステムが必要となります。

必須製品

- Interstage Application Server Enterprise Edition
- DBMS製品
  DB未使用時は不要です。また、DB共用の場合は、ノード間でのDB共用に対応したDBMS製品が必要です。

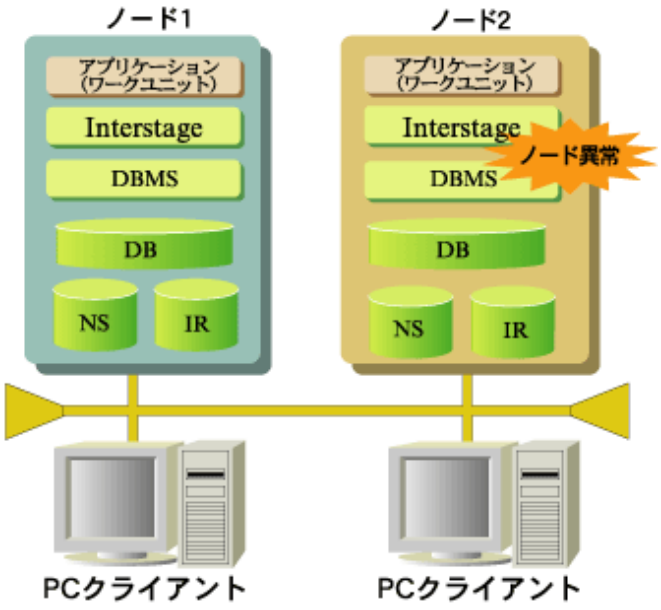
パターン1 (DB未使用)の構成図

以下に、両現用縮退方式での縮退でDBMSを使用しない場合について示します。



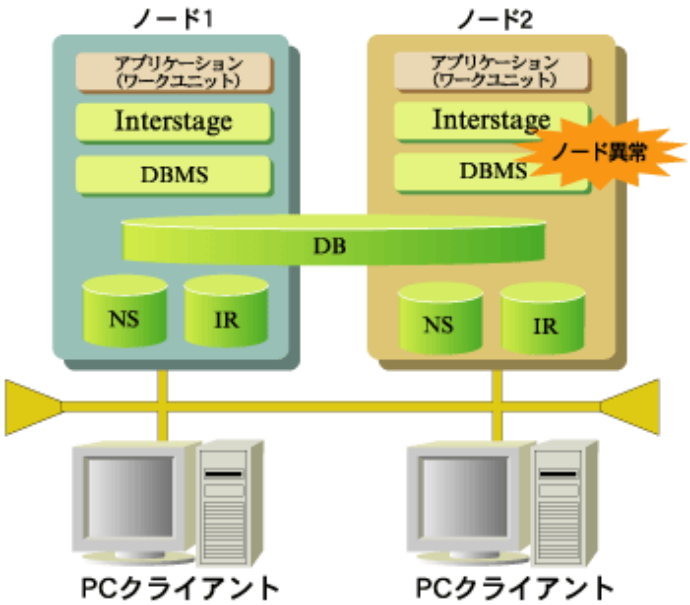
パターン2 (DB非共用)の構成図

以下に、両現用縮退方式での縮退で、DBMSを共用しない(ノード間で非共用)場合について示します。



パターン3 (DB共用)の構成図

以下に、両現用縮退方式での縮退で、DBMSを共用する(ノード間で共用)場合について示します。



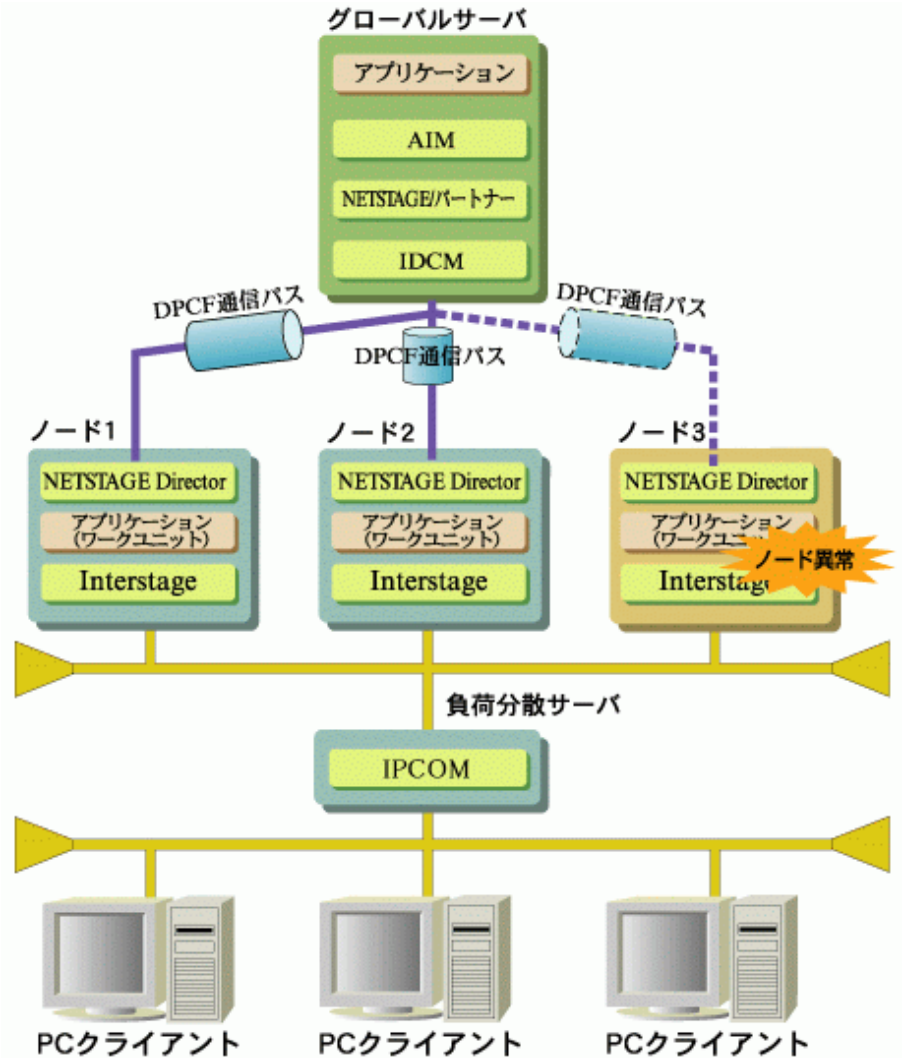
2.4 AIM連携での高信頼化システム Windows32/64 Solaris64

AIMアプリケーションと連携を行うAIM連携を行う場合、以下の機能との組み合わせによって、Interstageシステム側の高信頼化が可能です。

- IPCOMによる縮退との組み合わせ
- クラスタサービス機能との組み合わせ
- 両現用縮退方式での縮退との組み合わせ

グローバルサーバ側については、グローバルサーバの高信頼化を行ってください。
以降ではそれぞれの組み合わせについて説明します。

IPCOMによる縮退との組み合わせ

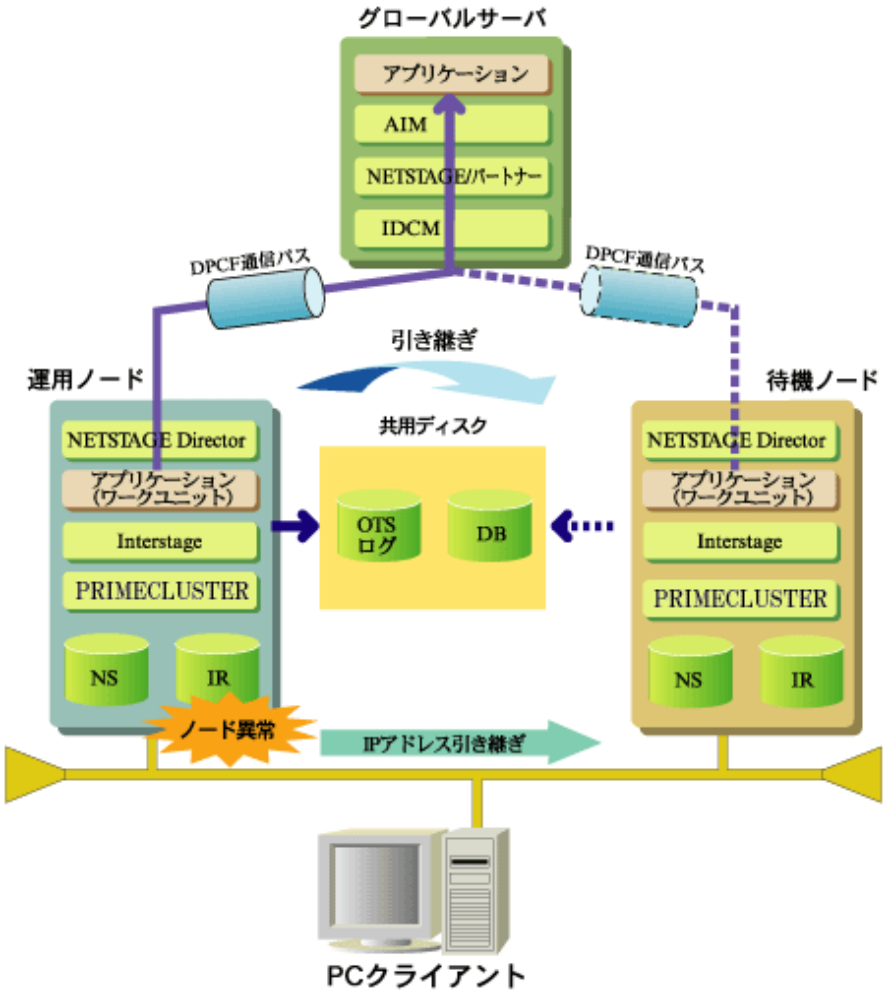


高信頼化の詳細は、「2.1 IPCOMによる縮退を利用した高信頼化システム」を参照してください。

必須製品

- Interstage Application Server Enterprise Edition
- IPCOM
  IIOP負荷分散に対応したIPCOMが必要です。IIOP負荷分散の詳細については「IPCOMのマニュアル」を参照してください。
- NETSTAGE Director

クラスタサービス機能との組み合わせ

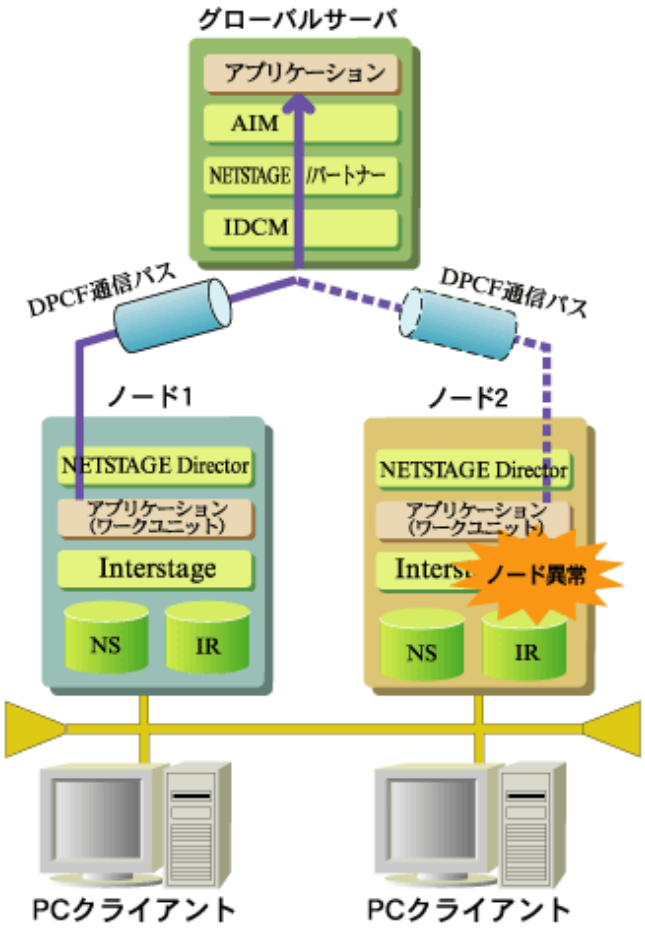


高信頼化の詳細は、「2.2 クラスタサービス機能を利用した高信頼化システム」を参照してください。

必須製品

- Interstage Application Server Enterprise Edition
- 各クラスタシステム
  クラスタシステムについては、「第4章 クラスタサービス機能」を参照してください。
- NETSTAGE Director

両現用縮退方式での縮退との組み合わせ



高信頼化の詳細は、「2.3 両現用縮退方式を利用した高信頼化システム」を参照してください。

必須製品

- Interstage Application Server Enterprise Edition
- NETSTAGE Director

2.5 推奨する高信頼化システムの形態

高信頼化システムを構築する場合、データベースの配置によりある程度形態が決定されます。
ここでは、アプリケーションサーバ形態の代表的なシステム形態と推奨高信頼化システムについてまとめています。

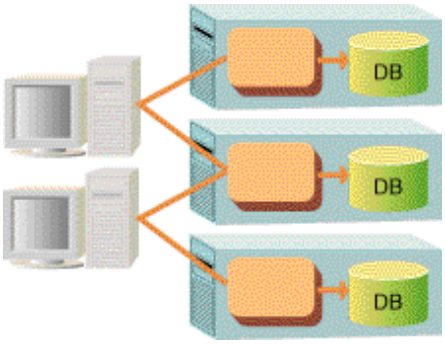
- 複数サーバを1つのシステムとして使用しデータベースをそれぞれのサーバに配置する場合
- 複数サーバを1つのシステムとして使用し1つのデータベースを複数のサーバより使用する場合
- 複数のサーバを1つのシステムとして使用しデータベースを使用しない場合
- 1台のサーバ内のデータベースに対してアクセスする場合
- 1台のサーバよりAIM連携を行う場合 Windows32/64 Solaris64
- 複数のサーバをAIMアプリケーションと連携する場合 Windows32/64 Solaris64

複数サーバを1つのシステムとして使用しデータベースをそれぞれのサーバに配置する場合

システム形態

以下を利用した高信頼化システムとなります。

- 両現用縮退方式
- IPCOMによる縮退
- Windows32 Linux32
  ロードバランスによる縮退



特徴

- クライアントの負荷を分散させることができます。データベースの参照処理に適しています。
- 両現用縮退方式の場合は、負荷をクライアント側で振り分ける必要があるため、小/中規模のシステムに適しています。
- 以下の場合は、動的に負荷を振り分けるため、大規模なシステムに適しています。
  - IPCOMによる縮退
  - Windows32 Linux32
    ロードバランスによる縮退

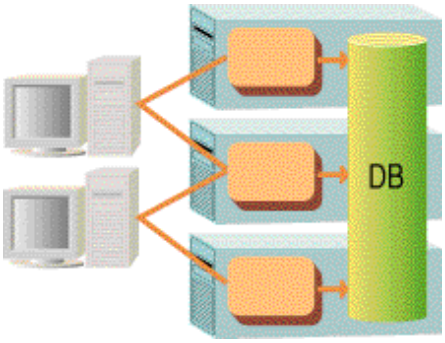
複数サーバを1つのシステムとして使用し1つのデータベースを複数のサーバより使用する場合

システム形態

以下を利用した高信頼化システムとなります。

- 両現用縮退方式
- IPCOMによる縮退

- Windows32 Linux32
  ロードバランスによる縮退



特徴

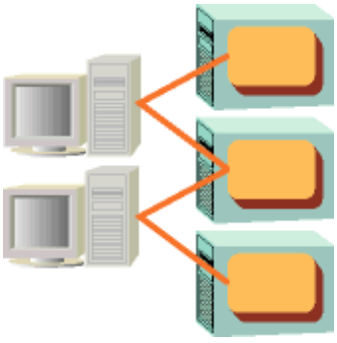
- クライアントの負荷を分散させることができます。データベースの参照/更新処理に適しています。
  ただし、サーバダウン時に使用していたテーブルはデータベースシステムが自動的にリカバリを行うまで、排他待ちになります。
- 両現用縮退方式の場合は、負荷をクライアント側で振り分ける必要があるため、小/中規模のシステムに適しています。
- 以下の場合は、動的に負荷を振り分けるため、大規模なシステムに適しています。
  - IPCOMによる縮退
  - Windows32 Linux32
    ロードバランスによる縮退

複数のサーバを1つのシステムとして使用しデータベースを使用しない場合

システム形態

以下を利用した高信頼化システムとなります。

- 両現用縮退方式
- IPCOMによる縮退
- Windows32 Linux32
  ロードバランスによる縮退



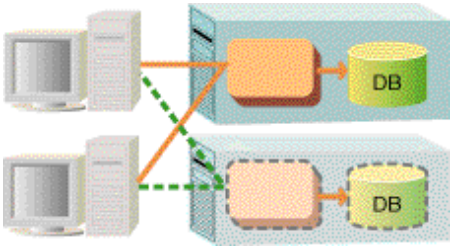
特徴

- クライアントからの負荷を分散させることができます。
  この形態はアプリケーションサーバとして使用する場合に適しています。
- 両現用縮退方式の場合は、負荷をクライアント側で振り分ける必要があるため、小/中規模のシステムに適しています。
- 以下の場合は、動的に負荷を振り分けるため、大規模なシステムに適しています。
  - IPCOMによる縮退
  - Windows32 Linux32
    ロードバランスによる縮退

1台のサーバ内のデータベースに対してアクセスする場合

システム形態

クラスタサービス機能を利用した高信頼化システムとなります。



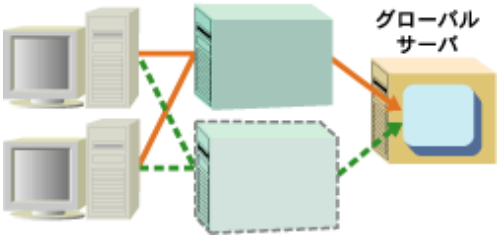
特徴

データベースの参照/更新に適しています。クラスタサービス機能を使用するため、より堅牢化したシステムを構築することができます。

1台のサーバよりAIM連携を行う場合 Windows32/64 Solaris64

システム形態

クラスタサービス機能を利用した高信頼化システムとなります。



特徴

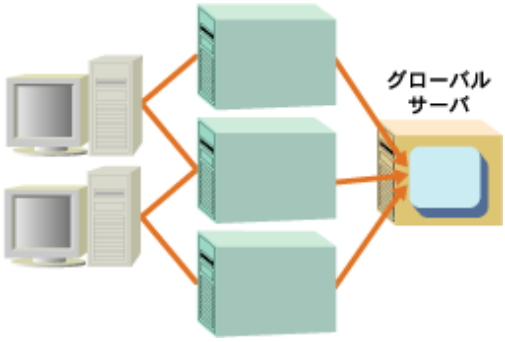
クラスタサービス機能を使用するため、より堅牢化したシステムを構築することができます。

複数のサーバをAIMアプリケーションと連携する場合 Windows32/64 Solaris64

システム形態

以下を利用した高信頼化システムとなります。

- 両現用縮退方式
- IPCOMによる縮退
- Windows32
  ロードバランスによる縮退

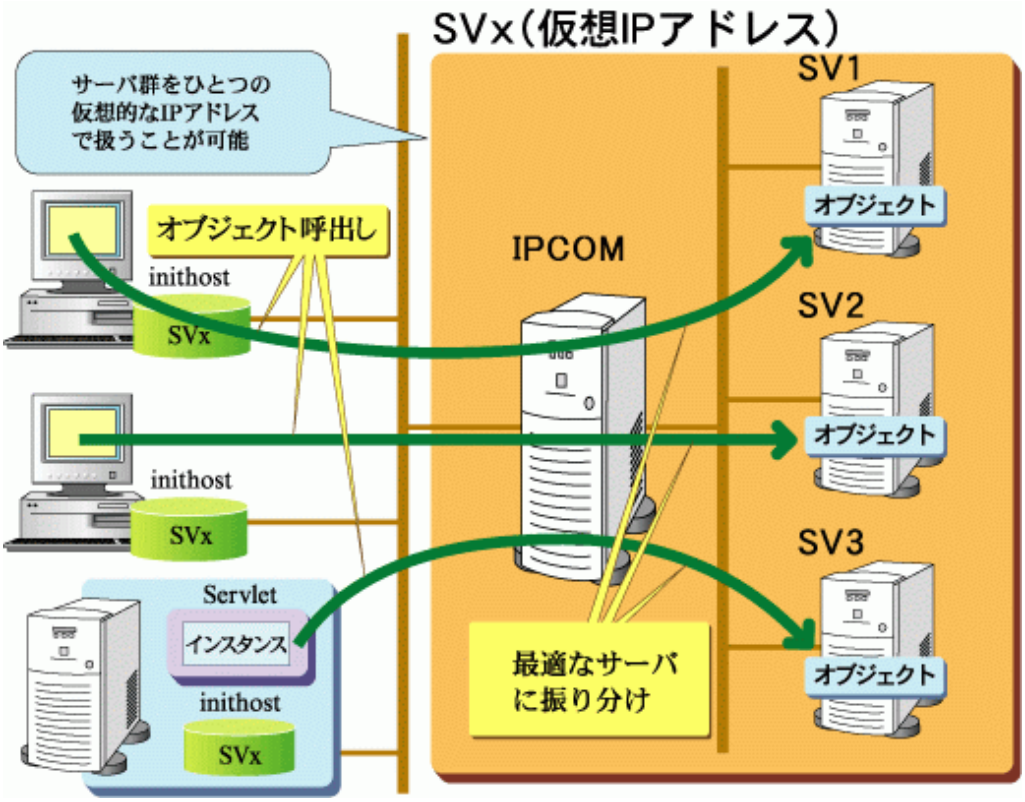


特徴

- クライアントからの負荷を分散させることができます。
  高性能なGSマシンを複数サーバより使用することができます。
- 両現用縮退方式の場合は、負荷をクライアント側で振り分ける必要があるため、小/中規模のシステムに適しています。
- 以下の場合は、動的に負荷を振り分けるため、大規模なシステムに適しています。
  - IPCOMによる縮退
  - Windows32
    ロードバランスによる縮退

第3章 IPCOMを利用した負荷分散

IPCOMと連携することにより、負荷分散システムを構築することが可能となります。
IPCOMでは、サーバ群を1つの仮想的なサーバマシンとしてクライアントに見せることが可能です。クライアントから、サーバ群に対してオブジェクト呼出しを行うことによって、IPCOMがサーバ群から最適なサーバマシンを選択し、オブジェクト呼出しを振り分けます。
Interstageの負荷分散システムを構築する場合、本方式を推奨します。



ここでは、以下の内容について説明しています。

- プログラミング設計
- 環境設定手順
- 運用手順
- サーバダウン時の動作
- ワークユニット停止時の縮退運用

IPCOMが運用中に停止した場合には、システム全体が停止します。万一の場合に備えて二重化することを推奨しています。詳細については、「IPCOMのマニュアル」を参照してください。

注意

- IPCOMを利用した負荷分散機能は、Interstage Application Server Enterprise Editionでのみ使用できます。
- 本製品では、イベントサービスを利用した負荷分散機能は提供していません。

- Java EE 7環境でIPCOMを利用した負荷分散機能を使用する場合は、「Java EE 7 設計・構築・運用ガイド」の「IPCOMを利用したIIOP通信の負荷分散」を参照してください。

3.1 運用モデル

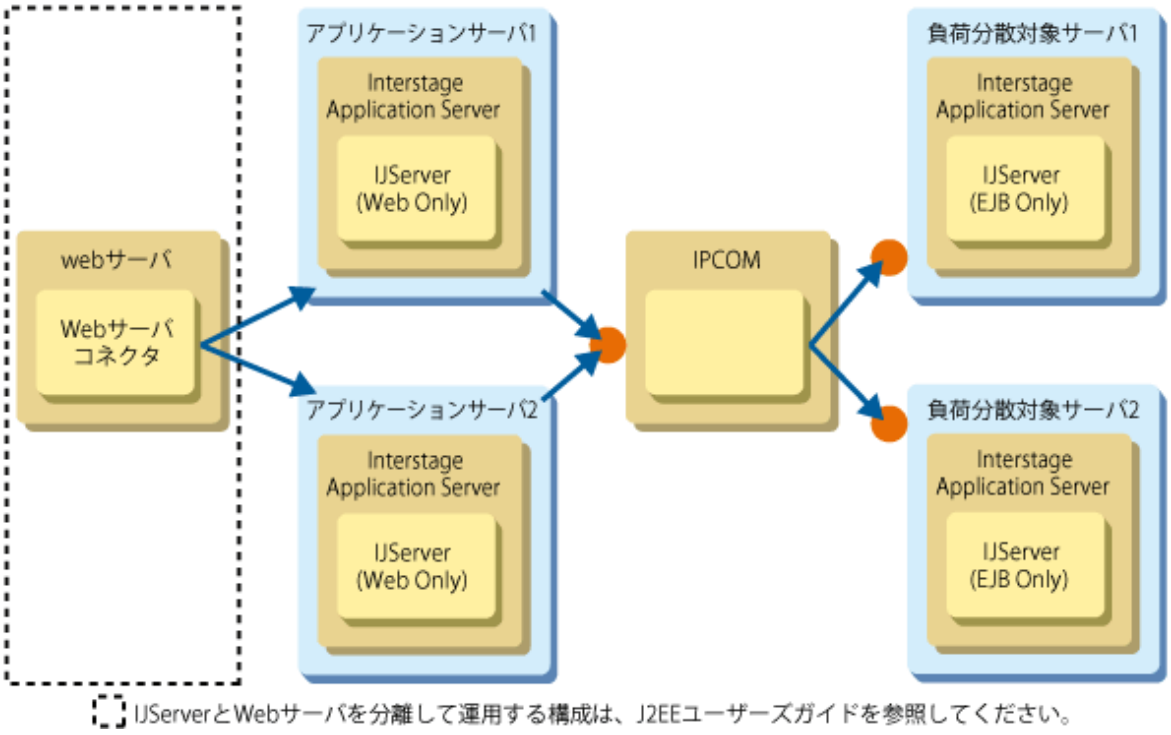
ここでは、以下について説明します。

- J2EEモデルにおける負荷分散
- CORBAワークユニットにおける負荷分散
- Windows32 Linux32
  トランザクションアプリケーションにおける負荷分散

Java EEモデルにおける負荷分散については、「Java EE 7 設計・構築・運用ガイド」の「IPCOMを利用したIIOP通信の負荷分散」を参照してください。

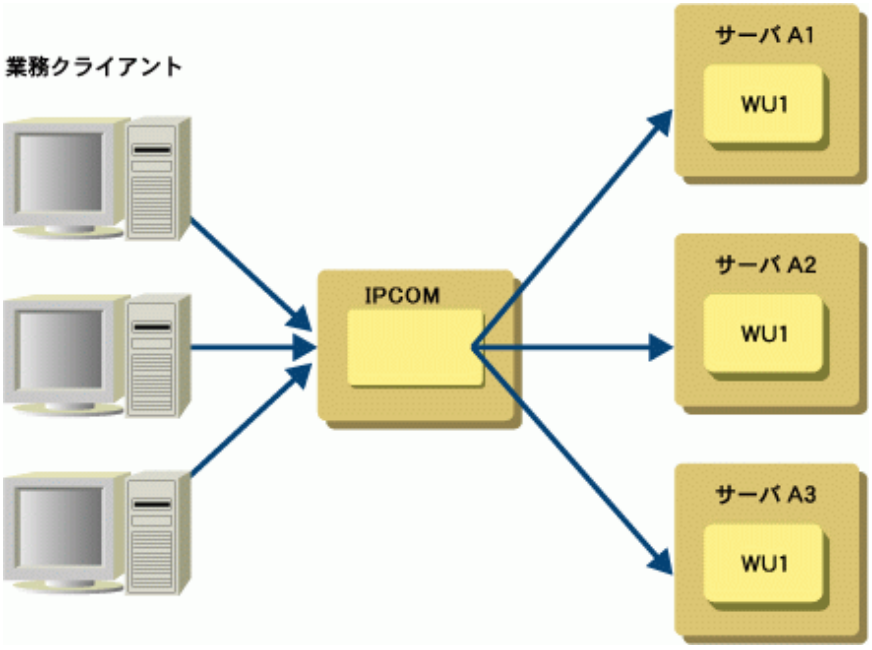
3.1.1 J2EEモデルにおける負荷分散

Interstage Application ServerはIPCOMと連携することにより、Webアプリケーション ― EJBアプリケーション間の負荷分散システムを構築することが可能となります。
Webアプリケーションのみ運用するIJServerワークユニットとEJBアプリケーションのみ運用するIJServerワークユニットを，別のサーバマシンに分離して負荷分散運用を行う場合に、分散先のサーバマシンの稼動状況を監視することで負荷分散を行います。
以下に、IJServerワークユニットに対する負荷分散環境の概要図を示します。



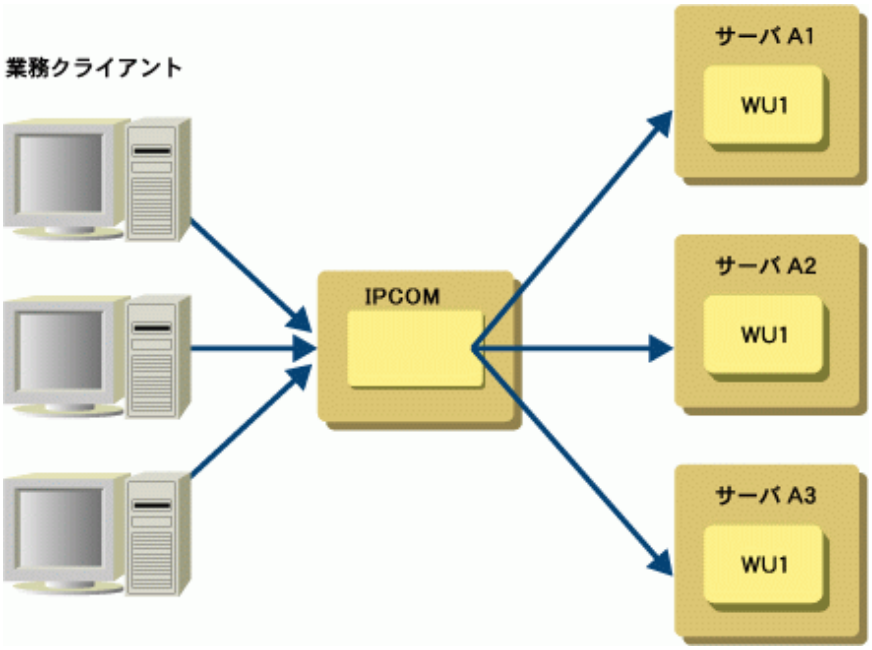
3.1.2 CORBAワークユニットにおける負荷分散

Interstage Application ServerはIPCOMと連携することにより、CORBAのクライアントーサーバアプリケーション間の負荷分散システムを構築できます。
以下にCORBAワークユニットに対する負荷分散環境の概要図を示します。



3.1.3 トランザクションアプリケーションにおける負荷分散 Windows32 Linux32

Interstage Application ServerはIPCOMと連携することにより、トランザクションアプリケーションのクライアントーサーバアプリケーション間の負荷分散システムを構築できます。
以下にトランザクションアプリケーションのワークユニットに対する負荷分散環境の概要図を示します。



3.2 環境設定方法

ここでは、以下の環境設定方法を説明します。

- J2EEモデル
- CORBAワークユニット
- Windows32 Linux32
  トランザクションアプリケーション

3.2.1 J2EEモデル

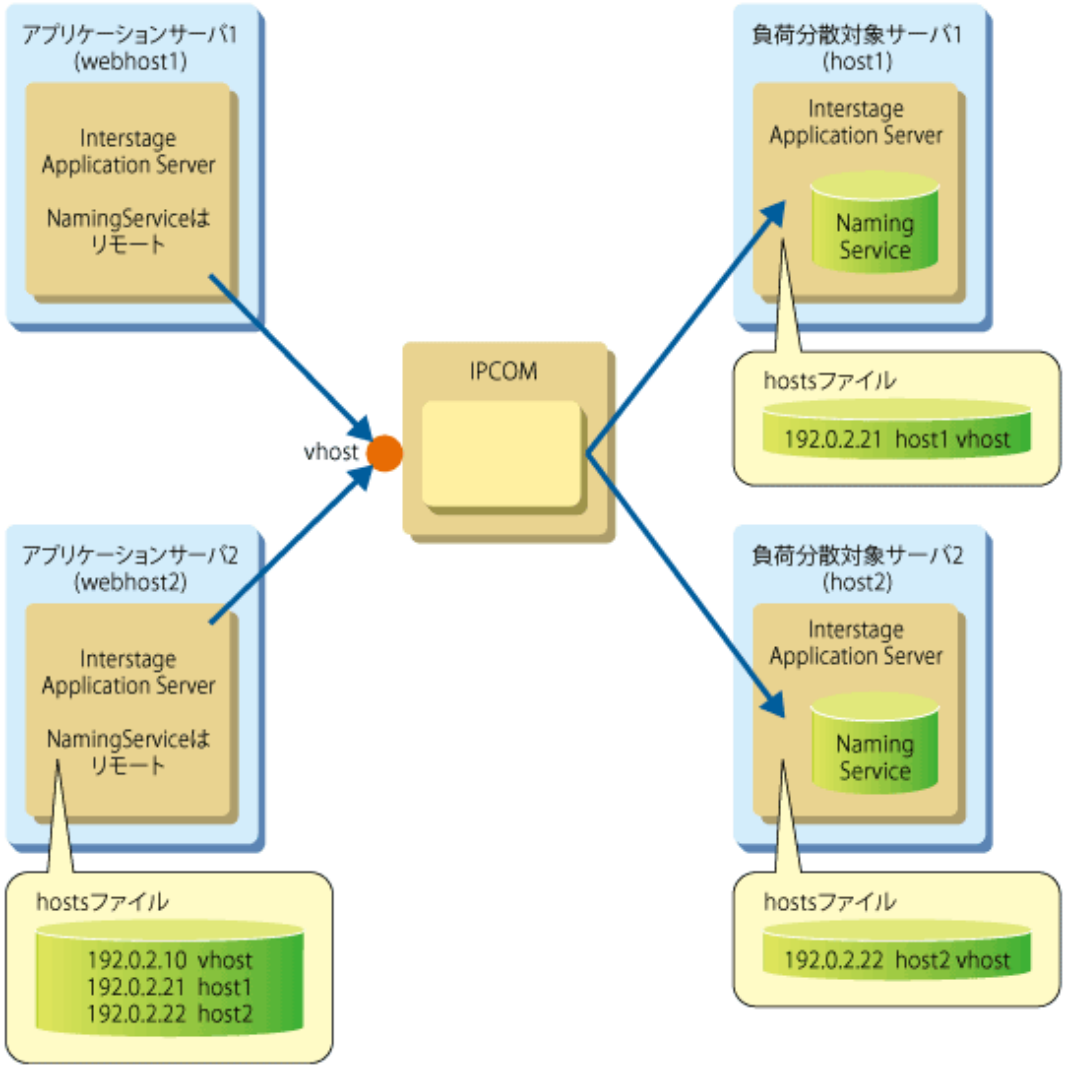
3.2.1.1 IPCOMの設定

負荷分散ポリシーの設定

「IPCOMのマニュアル」を参照してください。
なお、負荷分散ポリシーの設定は、以下のように行ってください。

- 負荷分散ポリシーのセション維持(一意性の保証)の設定は、コネクション単位の分散を行うように設定してください(コネクション単位の分散を推奨します)。
- EJBアプリケーションに対する負荷分散においては、メソッド呼び出し単位の負荷分散を行うように設定してください。

3.2.1.2 Interstageのセットアップ



Interstage Application Serverのセットアップ(アプリケーションサーバ側)

アプリケーションサーバのInterstageの環境設定は、アプリケーションサーバとなる全サーバに対して、以下の手順で実施してください。
Interstageの環境設定手順の詳細については、Interstage管理コンソールのヘルプを参照してください。

1. **hostsファイルの設定**

   以下のhostsファイルに負荷分散対象サーバのホスト名と仮想ホスト名の宣言を追加します。

   Windows32/64

   Windows(R)インストールフォルダ¥system32¥drivers¥etc¥hosts

   Solaris64 Linux32/64

   /etc/hosts

   宣言を追加した後、pingコマンドによりホスト名が解決されているかを確認してください。
   以下に、負荷分散対処となるサーバのホスト名がhost1、host2、仮想IPアドレスのホスト名がvhostの場合の記述例を示します。なお、"192.0.2.21"のIPアドレスは、host1の実IPアドレス、"192.0.2.22"のIPアドレスは、host2の実IPアドレスです。

```
192.0.2.10   vhost
192.0.2.21   host1
192.0.2.22   host2
```

2. **Interstageの停止**

   Interstage管理コンソールの[Interstage Application Server] > [システム] > [状態]から[停止] > [強制停止]からInterstageを強制停止してください。

3. **ネーミングサービスの設定**

   Interstage管理コンソールの[Interstage Application Server] > [システム] > [環境設定]タブを選択して、[ネーミングサービス詳細]で以下の設定内容を設定します。

項目		説明
ホスト定義	リモートホスト	リモートホストを選択してください。
	サーバホスト名	IPCOMのサイト負荷分散ポリシーに設定した、仮想IPアドレスに対応する仮想ホスト名を設定してください。
	ポート番号	IPCOMのIIOPサービスに設定した代表ポートの値を設定してください。有効範囲は1～65535です。 IPCOMのメソッド負荷分散設定の「する」を選択して代表ポートを省略した場合は、8002が設定されます。

ネーミングサービスの配置

アプリケーションサーバは、リモートホストのネーミングサービスを使用してください。

Interstage Application Serverのセットアップ(負荷分散対象サーバ側)

負荷分散対象サーバのInterstageの環境設定は、負荷分散対象となる全サーバに対して、以下の手順で実施してください。Interstageの環境設定手順の詳細については、「運用ガイド(基本編)」の「Interstageの環境設定」を参照してください。

1. **hostsファイルの設定**

   以下のhostsファイルに定義されている自サーバのホスト名とIPアドレスの設定文に対して、別名で、仮想IPアドレスのホスト名の宣言を追加します。

   Windows32/64

   Windows(R)インストールフォルダ¥system32¥drivers¥etc¥hosts

   Solaris64 Linux32/64

   /etc/hosts

   宣言を追加した後、pingコマンドによりホスト名が解決されているかを確認してください。
   以下に、初期化対象となるサーバのホスト名がhost1、仮想IPアドレスのホスト名がvhostの場合の記述例を示します。なお、"192.0.2.21"のIPアドレスは、host1の実IPアドレスです。

```
192.0.2.21  host1   vhost
```

2. **Interstageの環境設定**

   負荷分散対象サーバのInterstage システムの環境は、Interstageのインストール時にデフォルトの環境設定のままで運用可能です。

ネーミングサービスの配置

ネーミングサービスは、負荷分散の対象となるすべてのサーバマシンに配置してください。

3.2.2 CORBAワークユニット

3.2.2.1 IPCOMの設定

負荷分散ポリシーの設定

「IPCOMのマニュアル」を参照してください。
なお、負荷分散ポリシーの設定は、以下のように行ってください。

- 負荷分散ポリシーのセション維持(一意性の保証)の設定は、コネクション単位の分散を行うように設定してください(コネクション単位の分散を推奨します)。

Windows32 Linux32

- 負荷分散ポリシーとして待ちメッセージ数を指定する場合には、待ちメッセージの監視対象として、以下を指定してください。
  - Interstag管理コンソールを使用してCORBAワークユニットを作成した場合
    [Interstage Application Server] > [システム] > [ワークユニット] > 「ワークユニット名」 > [配備]タブの配備設定で指定した、インプリメンテーションリポジトリID
  - isaddwudefコマンドを使用してCORBAワークユニットを作成した場合
    ワークユニット定義の[Application Program]セクションの"impl ID"で指定したインプリメンテーションリポジトリID
- 「待ちメッセージ数限界値」、「待ちメッセージ数復旧値」には、適切な値を設定してください。

3.2.2.2 Interstageのセットアップ

セットアップ手順の概要

以下の手順でセットアップしてください。

1. Interstageのセットアップ
   1. hostsファイルの設定
   2. Interstageの停止
   3. Interstageシステム定義の生成
   4. Interstageシステム定義ファイルの登録
   5. Interstage動作環境定義のカストマイズ Solaris64 Linux32/64
   6. Interstageの初期化
2. オブジェクトリファレンスの登録(負荷分散方式の設定)
   以下のどちらかの方式で設定します。
   - ネーミングサービスのオブジェクトリファレンス獲得時点の負荷分散
   - メソッド呼出し単位の負荷分散
3. クライアントの環境設定

ただし、ネーミングコンテキストを使用している場合で、メソッド呼出し単位の負荷分散方式を行う場合は、以下の手順でセットアップしてください。

1. Interstageのセットアップ
   1. hostsファイルの設定
   2. Interstageの停止
   3. Interstageシステム定義の生成

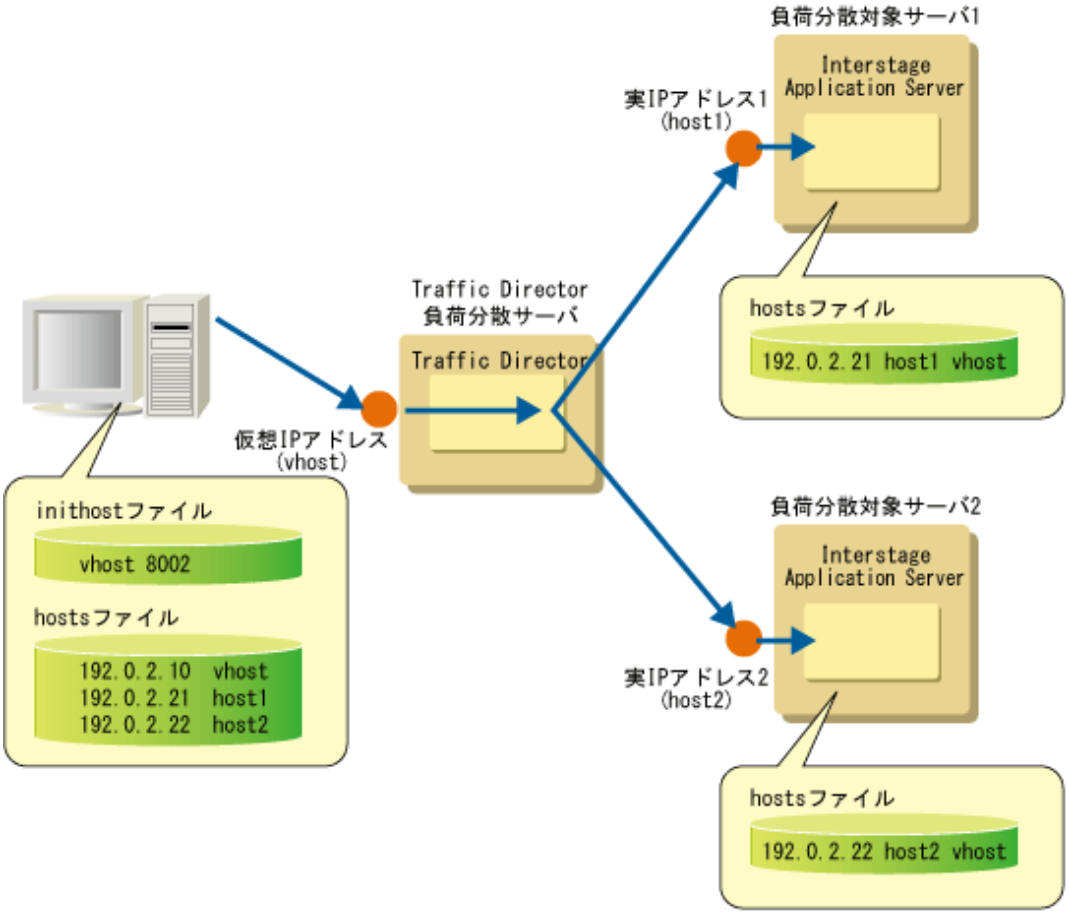
4. Interstageシステム定義ファイルの登録
5. ホスト名の設定
6. Interstageの初期化

2. オブジェクトリファレンスの登録(負荷分散方式の設定)
   メソッド呼出し単位の負荷分散方式で設定します。
3. クライアントの環境設定

セットアップ手順の詳細について、以降に示します。

Interstageのセットアップ

負荷分散対象サーバのInterstageの環境設定は、負荷分散対象となる全サーバに対して、以下の手順で実施してください。Interstageの環境設定手順の詳細については、「運用ガイド(基本編)」の「Interstageの環境設定」を参照してください。



1. **hostsファイルの設定**

   以下のhostsファイルに定義されている自サーバのホスト名とIPアドレスの設定文に対して、別名で、仮想IPアドレスのホスト名の宣言を追加します。

   Windows32/64

   Windows(R)インストールフォルダ¥system32¥drivers¥etc¥hosts

   Solaris64 Linux32/64

   /etc/hosts

宣言を追加した後、pingコマンドによりホスト名が解決されているかを確認してください。
以下に、初期化対象となるサーバのホスト名がhost1、仮想IPアドレスのホスト名がvhostの場合の記述例を示します。なお、"192.0.2.21"のIPアドレスは、host1の実IPアドレスです。

```
192.0.2.21  host1   vhost
```

2. **Interstageの停止**

isstop -fコマンドを実行し、Interstageを停止します。
以下に、isstop -fコマンドの実行例を示します。

```
> isstop -f
```

3. **Interstageシステム定義の生成**

isgendefコマンドを実行し、Interstageシステム定義を生成します。
以下に、システム規模がsmallの場合におけるisgendefコマンドの実行例を示します。

```
> isgendef small
```

4. **Interstageシステム定義ファイルの登録**

isregistdefコマンドを実行し、Interstageシステム定義ファイルを登録します。
以下に、isregistdefコマンドの実行例を示します。

```
> isregistdef
```

5. **Interstage動作環境定義のカストマイズ** Solaris64 Linux32/64

Interstage動作環境定義をカストマイズします。Interstage動作環境定義は以下に格納されています。

Solaris64

/opt/FSUNtd/etc/isreg/isinitdef.txt

Linux32/64

/opt/FJSVtd/etc/isreg/isinitdef.txt

Interstage動作環境定義に、以下の項目を設定してください。

- CORBA HOST NAME
- CORBA PORT NUMBER

以下に、各定義項目の指定例を示します。

```
CORBA HOST NAME=host1
CORBA PORT NUMBER=8003
```

注意

ネーミングコンテキストを使用しない場合に、本設定を行ってください。

6. **ホスト名の設定**

OD_set_envコマンドを実行し、オブジェクトリファレンスの生成時に埋め込むホスト名を設定します。
以下に、OD_set_envコマンドの実行例を示します。

```
> OD_set_env -n vhost
```

注意

ネーミングコンテキストを使用する場合に、本設定を行ってください。

7. **Interstageの初期化**

-fオプションを指定してisinitコマンドを実行し、Interstageを初期化します。
Windowsシステムでは、Interstageの初期化時には、ネーミングサービスを、ローカルサーバ上にセットアップしてください。
以下に、Interstageの運用形態がtype1の場合の初期化実行例を示します。

```
> isinit -f type1
```

注意

IPCOM連携機能では、負荷分散対象サーバ上のネーミングサービスに登録されたオブジェクトリファレンスの登録内容により負荷分散方式を決定します。
したがって、ネーミングサービスは、負荷分散対象となるすべてのサーバマシン上に配置します。

オブジェクトリファレンスの登録(負荷分散方式の設定)

オブジェクトリファレンスの登録方法は、負荷分散の方式により、異なります。

ネーミングサービスのオブジェクトリファレンス獲得時点の負荷分散

- Interstag管理コンソールを使用してCORBAワークユニットを作成した場合

  [Interstage Application Server] > [システム] > [ワークユニット] > 「ワークユニット名」 > [配備]タブより、詳細設定を表示し、[CORBAアプリケーション]の設定画面を表示して以下の項目を設定してください。

項目		説明
IPCOMのメソッド負荷分散設定	・ する ・ しない	「しない」を選択してください。
	仮想ホスト名	省略してください。設定した場合は無視されます。
	代表ポート	省略してください。設定した場合は無視されます。

  Interstage管理コンソールについて詳細は、Interstage管理コンソールのヘルプを参照してください。

- isaddwudefコマンドを使用してCORBAワークユニットを作成した場合

  オブジェクトリファレンスの登録は、以下の方法で登録してください。

  ― OD_or_admコマンドによる手動登録

  OD_or_admコマンドにより、各サーバマシンに配置されたネーミングサービスへサーバアプリケーションのオブジェクトリファレンスを登録する場合、オブジェクトの参照先ホスト名を指定しないでください。-hオプションを指定する場合は、オブジェクトの所在に自ホストを設定してください。

メソッド呼出し単位の負荷分散

- Interstag管理コンソールを使用してCORBAワークユニットを作成した場合

  [Interstage Application Server] > [システム] > [ワークユニット] > 「ワークユニット名」 > [配備]タブより、詳細設定を表示し、[CORBAアプリケーション]の設定画面を表示して以下の項目を設定してください。

項目		説明
IPCOMのメソッド負荷分散設定	・ する ・ しない	「する」を選択して、仮想ホスト名と代表ポートを設定してください。
	仮想ホスト名	IPCOMのサイト負荷分散ポリシーに設定した、仮想IPアドレスに対応する仮想ホスト名を設定してください。
	代表ポート	IPCOMのIIOPサービスに設定した代表ポートの値を設定してください。有効範囲は1～65535です。 IPCOMのメソッド負荷分散設定の「する」を選択して代表ポートを省略した場合は、8002が設定されます。

  Interstage管理コンソールについて詳細は、Interstage管理コンソールのヘルプを参照してください。

- isaddwudefコマンドを使用してCORBAワークユニットを作成した場合

  OD_or_admコマンドによる手動登録を行ってください。
  OD_or_admコマンドにより、ネーミングサービスへサーバアプリケーションのオブジェクトリファレンスを登録する際、-hオプションで指定するオブジェクトの所在には、仮想IPアドレスのホスト名を指定してください。

注意

ネーミングコンテキストを使用している場合は、ネーミングコンテキストを登録する必要があります。負荷分散対象サーバに複写元のサーバシステムのネーミングサービス登録情報をコピーしてください。これにより、負荷分散対象サーバのネーミングサービスに、複写元のサーバシステムと同じ構成のネーミングコンテキストを登録できます。
負荷分散対象サーバ“host2”に、複写元のサーバシステム“host1”のネーミングサービス登録情報をコピーする操作手順を以下に示します。

host1での操作手順

1. **Interstageの停止**

   isstop -fコマンドを実行し、Interstageを停止します。
   以下に、isstop -fコマンドの実行例を示します。

```
> isstop -f
```

2. **ネーミングサービス登録情報のコピー**

   ネーミングサービス登録情報が格納されているディレクトリを作業用ディレクトリにコピーします。
   以下に、カレントディレクトリ(作業用ディレクトリ)にネーミングサービス登録情報をコピーする場合の実行例を示します。

Windows32/64

```
> xcopy C:¥Interstage¥ODWIN¥etc¥CosNaming CosNaming_host1
```

   host2の作業用ディレクトリに、上記のディレクトリ“CosNaming_host1”をコピーします。

Solaris64

```
> cp -r -p /opt/FSUNod/etc/CosNaming CosNaming_host1
```

host2の作業用ディレクトリに、ftpコマンドを使用して上記のディレクトリ“CosNaming_host1”をバイナリ形式で転送します。

Linux32/64

```
> cp -r -p /opt/FJSVod/etc/CosNaming CosNaming_host1
```

host2の作業用ディレクトリに、ftpコマンドを使用して上記のディレクトリ“CosNaming_host1”をバイナリ形式で転送します。

3. **Interstageの起動**

isstartコマンドを実行し、Interstageを起動します。
以下に、isstartコマンドの実行例を示します。

```
> isstart
```

4. **ネーミングコンテキストの登録内容の確認**

odlistns -l -Rコマンドを実行し、host1のネーミングサービスに登録されているネーミングコンテキストを表示します。以下の[表示結果]とhost2のodlistns -l -Rコマンドの[表示結果]を比較して、host1とhost2のネーミングサービスにネーミングコンテキストが同じ構成で登録されているかを確認することにより、ネーミングサービス登録情報が正しくコピーされたかを判断します。
以下に、odlistns -l -Rコマンドの実行例とその表示結果を示します。

```
> odlistns -l -R
[表示結果]
Name(Type)   Object information(detail)
             Default object information(detail)
NC001(c)     IDL:CosNaming/NamingContextExt:1.0, IDL:CosNaming/NamingContext:1.0, (vhost:8002:1.0:), context.
1225331880.0
```

上記の[表示結果]から、host1のネーミングサービスには、コンテキストID“context.1225331880.0”のネーミングコンテキスト“NC001”が登録されていることが確認できます。

host2での操作手順

1. **Interstageの停止**

isstop -fコマンドを実行し、Interstageを停止します。
以下に、isstop -fコマンドの実行例を示します。

```
> isstop -f
```

2. **ネーミングサービス登録情報のバックアップ・コピー**

host2のネーミングサービス登録情報が格納されているディレクトリを退避したあと、host1のネーミングサービス登録情報をコピーします。
以下に、カレントディレクトリ(作業用ディレクトリ)にhost1のネーミングサービス登録情報が格納されている場合の実行例を示します。

Windows32/64

```
> rename C:¥Interstage¥ODWIN¥etc¥CosNaming CosNaming_backup
> xcopy CosNaming_host1 C:¥Interstage¥ODWIN¥etc¥CosNaming
```

Solaris64

```
> mv /opt/FSUNod/etc/CosNaming /opt/FSUNod/etc/CosNaming_backup
> cp -r -p CosNaming_host1 /opt/FSUNod/etc/CosNaming
```

```
> chown root /etc/opt/FSUNod/CosNaming
> chown sys /etc/opt/FSUNod/CosNaming
```

Linux32/64

```
> mv /opt/FSJSVod/etc/CosNaming /opt/FSJSVod/etc/CosNaming_backup
> cp -r -p CosNaming_host1 /opt/FSJSVod/etc/CosNaming
> chown root /etc/opt/FSJSVod/CosNaming
> chown sys /etc/opt/FSJSVod/CosNaming
```

3. **Interstageの起動**

isstartコマンドを実行し、Interstageを起動します。
以下に、isstartコマンドの実行例を示します。

```
> isstart
```

4. **ネーミングコンテキストの確認**

odlistns -l -Rコマンドを実行し、host2のネーミングサービスに登録されているネーミングコンテキストを表示します。
以下に、odlistns -l -Rコマンドの実行例とその表示結果を示します。

```
> odlistns -l -R
[表示結果]
Name(Type)  Object information(detail)
            Default object information(detail)
NC001(c)    IDL:CosNaming/NamingContextExt:1.0, IDL:CosNaming/NamingContext:1.0, (vhost:8002:1.0:), context.
1225331880.0
```

上記の[表示結果]から、host2のネーミングサービスにも、コンテキストID“context.1225331880.0”のネーミングコンテキスト“NC001”が登録されていることが確認できます。
これにより、host1とhost2のネーミングサービスにネーミングコンテキストが同じ構成で登録されていることが確認でき、ネーミングサービス登録情報が正しくコピーされていると判断できます。

クライアントの環境設定

以下のファイルに設定するネーミングサービスの参照先ホスト名に、仮想IPアドレスのホスト名を設定してください。IPアドレスは指定できませんので、注意してください。

Windows32/64

inithostファイル

Solaris64 Linux32/64

initial_hostsファイル

正しい記述例)

```
vhost          8002
```

誤った記述例)

```
192.0.2.10    8002
```

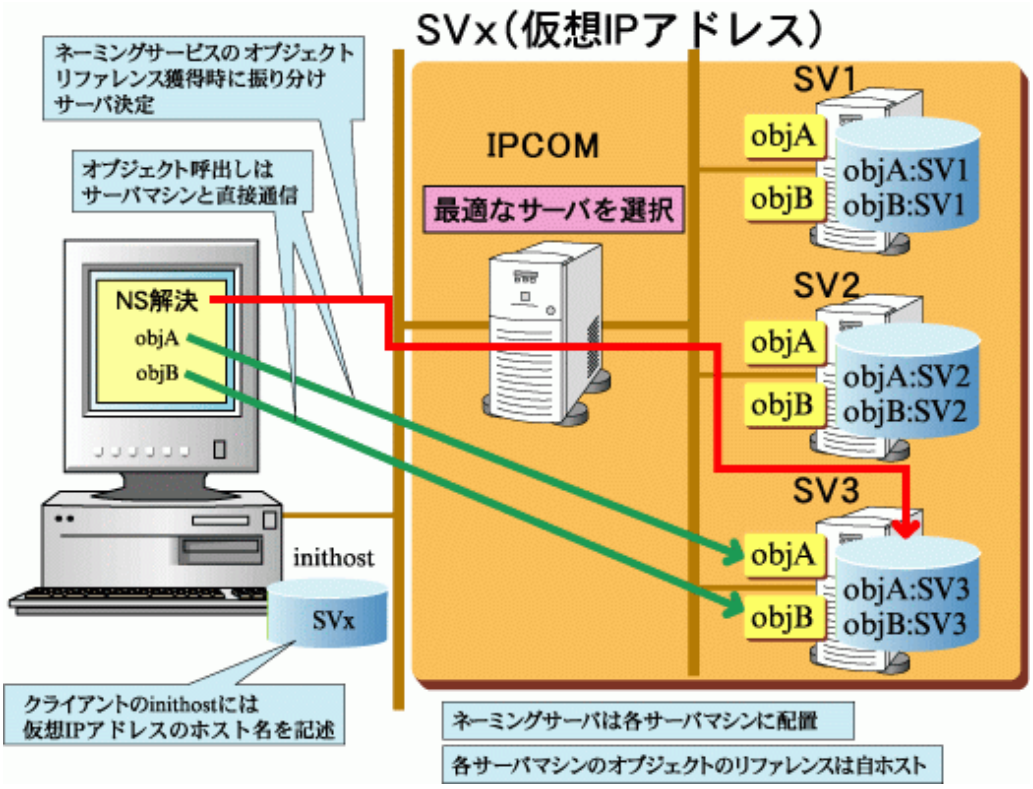
また、以下のhostsファイルに、仮想IPアドレスと仮想IPアドレスのホスト名、負荷分散対象となるすべてのサーバのホスト名とIPアドレスの対応について宣言してください。
宣言を追加した後、pingコマンドによりホスト名が解決されているかを確認してください。

Windows32/64

Windowsインストールフォルダ¥system32¥drivers¥etc¥hosts

Solaris64 Linux32/64

/etc/hosts



注意

- Interstage Application Serverを運用しているサーバ(Interstage統合コマンドで初期化した環境のみ)をクライアントとする場合、Interstage初期化時にネーミングサービスの参照先ホストとして、仮想ホスト名を指定してください。指定方法については、「運用ガイド(基本編)」の「Interstage統合コマンドによる運用操作」の「ネーミングサービス・インタフェースリポジトリの設定変更」を参照してください。
- 1台のクライアントマシンからサーバマシンへの通信に対して直接通信するパターンと負荷分散装置を経由して通信するパターンが混在する場合、送信元のアドレス変換を行っていないとネットワーク異常が発生することがあります。そのため、負荷分散装置等を経由した通信で、クライアントマシンのIPアドレスとポート番号が直接サーバマシンに通知されないように、NAT装置や負荷分散装置等のアドレス変換機能(NAPT等)を使用してください。

3.2.3 トランザクションアプリケーション Windows32 Linux32

3.2.3.1 IPCOMの設定

負荷分散ポリシーの設定

「IPCOMのマニュアル」を参照してください。
なお、負荷分散ポリシーの設定は、以下のように行ってください。

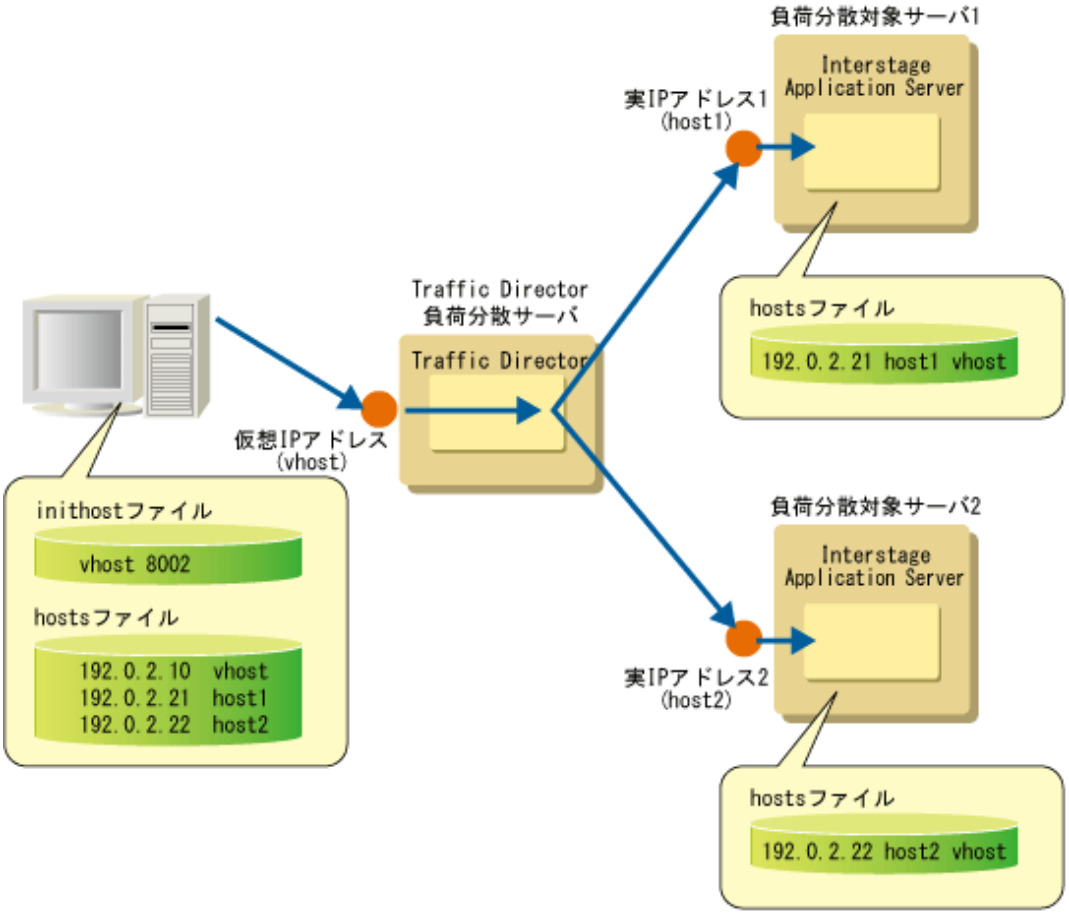
- 負荷分散ポリシーのセション維持(一意性の保証)の設定は、コネクション単位の分散を行うように設定してください(コネクション単位の分散を推奨します)。
- 負荷分散ポリシーとして待ちメッセージ数を指定する場合には、待ちメッセージの監視対象として、以下を指定してください。

  ー ワークユニット定義の[Application Program]セクションの"Destination"で指定したオブジェクト名
- 「待ちメッセージ数限界値」、「待ちメッセージ数復旧値」、「通信バッファ使用率限界値」、「通信バッファ使用率復旧値」には、適切な値を設定してください。

3.2.3.2 Interstageのセットアップ

Interstageのセットアップ

負荷分散対象サーバのInterstageの環境設定は、負荷分散対象となる全サーバに対して、以下の手順で実施してください。Interstageの環境設定手順の詳細については、「運用ガイド(基本編)」の「Interstageの環境設定」を参照してください。



1. **hostsファイルの設定**

   以下のhostsファイルに定義されている自サーバのホスト名とIPアドレスの設定文に対して、別名で、仮想IPアドレスのホスト名の宣言を追加します。

   Windows32

   Windows(R)インストールフォルダ¥system32¥drivers¥etc¥hosts

Linux32

/etc/hosts

宣言を追加した後、pingコマンドによりホスト名が解決されているかを確認してください。
以下に、初期化対象となるサーバのホスト名がhost1、仮想IPアドレスのホスト名がvhostの場合の記述例を示します。なお、"192.0.2.21"のIPアドレスは、host1の実IPアドレスです。

```
192.0.2.21   host1    vhost
```

2. **Interstageの停止**

isstop -fコマンドを実行し、Interstageを停止します。
以下に、isstop -fコマンドの実行例を示します。

```
> isstop -f
```

3. **Interstageシステム定義の生成**

isgendefコマンドを実行し、Interstageシステム定義を生成します。
以下に、システム規模がsmallの場合におけるisgendefコマンドの実行例を示します。

```
> isgendef small
```

4. **Interstageシステム定義ファイルの登録**

isregistdefコマンドを実行し、Interstageシステム定義ファイルを登録します。
以下に、isregistdefコマンドの実行例を示します。

```
> isregistdef
```

5. **Interstage動作環境定義のカストマイズ** Linux32

Interstage動作環境定義をカストマイズします。Interstage動作環境定義は以下に格納されています。

/opt/FJSVtd/etc/isreg/isinitdef.txt

Interstage動作環境定義に、以下の項目を設定してください。

- CORBA HOST NAME
- CORBA PORT NUMBER

以下に、各定義項目の指定例を示します。

```
CORBA HOST NAME=host1
CORBA PORT NUMBER=8003
```

6. **Interstageの初期化**

-fオプションを指定してisinitコマンドを実行し、Interstageを初期化します。
Windowsシステムでは、Interstageの初期化時には、ネーミングサービスを、ローカルサーバ上にセットアップしてください。
以下に、Interstageの運用形態がtype1の場合の初期化実行例を示します。

```
> isinit -f type1
```

注意

IPCOM連携機能では、負荷分散対象サーバ上のネーミングサービスに登録されたオブジェクトリファレンスの登録内容により負荷分散方式を決定します。
したがって、ネーミングサービスは、負荷分散対象となるすべてのサーバマシン上に配置します。

オブジェクトリファレンスの登録(負荷分散方式の設定)

オブジェクトリファレンスの登録方法は、負荷分散の方式により、異なります。

メソッド呼出し単位の負荷分散

OD_or_admコマンドによる手動登録を行ってください。
OD_or_admコマンドにより、ネーミングサービスへサーバアプリケーションのオブジェクトリファレンスを登録する際、-hオプションで指定するオブジェクトの所在には、仮想IPアドレスのホスト名を指定してください。

例

トランザクションアプリケーションのオブジェクトリファレンスの登録例

```
> OD_or_adm  -c  IDL:TDsample/tdtest:1.0  -a  FUJITSU-Interstage-TDLC  -h  vhost
 -p  8002  -n  TDsample::tdtest  (vhostは、仮想IPアドレスのホスト名)
```

ネーミングサービスのオブジェクトリファレンス獲得時点の負荷分散

オブジェクトリファレンスの登録は、以下のいずれかの方法で登録してください。登録方法は、ワークユニット定義の[Control Option]セクションの"Registration to Naming Service"で指定できます。

- ワークユニット起動による自動登録
- OD_or_admコマンドによる手動登録

OD_or_admコマンドにより、各サーバマシンに配置されたネーミングサービスへサーバアプリケーションのオブジェクトリファレンスを登録する場合、オブジェクトの参照先ホスト名を指定しないでください。-hオプションを指定する場合は、オブジェクトの所在に自ホストを設定してください。

クライアントの環境設定

以下のファイルに設定するネーミングサービスの参照先ホスト名に、仮想IPアドレスのホスト名を設定してください。IPアドレスは指定できませんので、注意してください。

Windows32

inithostファイル

Linux32

initial_hostsファイル

正しい記述例)

```
vhost          8002
```

誤った記述例)

```
192.0.2.10    8002
```

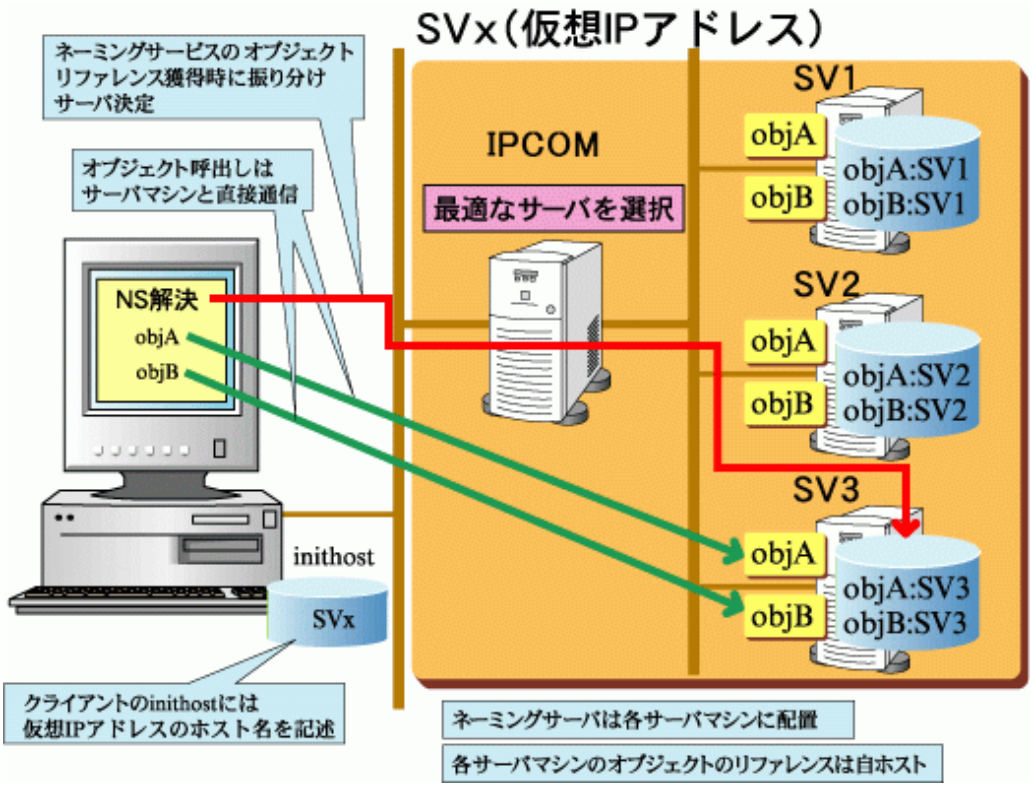
また、以下のhostsファイルに、仮想IPアドレスと仮想IPアドレスのホスト名、負荷分散対象となるすべてのサーバのホスト名とIPアドレスの対応について宣言してください。
宣言を追加した後、pingコマンドによりホスト名が解決されているかを確認してください。

Windows32

Windowsインストールフォルダ¥system32¥drivers¥etc¥hosts

Linux32

/etc/hosts



注意

- IPCOMと連携する場合の環境設定手順が、Interstage Application Server V4.1までと異なりますが、既存の手順でも運用可能です。ただし、機能範囲は、従来のとおりです。
- Interstage Application Serverを運用しているサーバ(Interstage統合コマンドで初期化した環境のみ)をクライアントとする場合、Interstage初期化時にネーミングサービスの参照先ホストとして、仮想ホスト名を指定してください。指定方法については、「運用ガイド(基本編)」の「Interstage統合コマンドによる運用操作」の「ネーミングサービス・インタフェースリポジトリの設定変更」を参照してください。

3.3 アプリケーションの設計方法

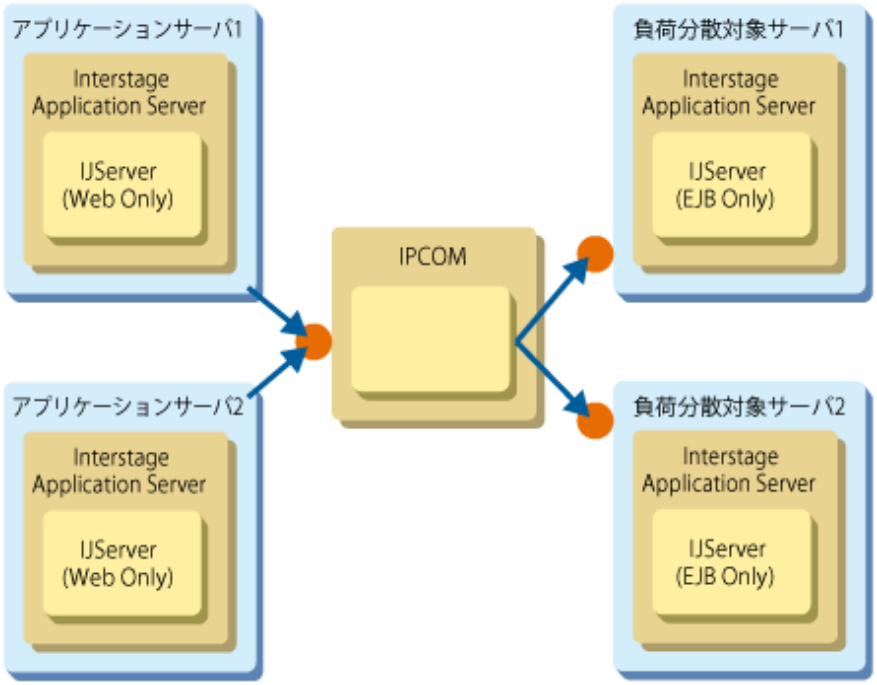
ここでは、IPCOMと連携した負荷分散システムのアプリケーションの設計方法について説明します。

3.3.1 J2EEアプリケーション

J2EEアプリケーションの配備と、プログラミング方法について説明します。

3.3.1.1 IJServerワークユニットの作成

EJBアプリケーションでIPCOMと連携する場合に必要な、環境設定について説明します。



アプリケーションサーバの環境設定

アプリケーションサーバのIJServerワークユニットのタイプは「Webアプリケーションのみを運用する」を選択して作成してください。

負荷分散対象サーバの環境設定

負荷分散対象サーバのIJServerワークユニットのタイプは「EJBアプリケーションのみを運用する」を選択し、仮想ホスト名と代表ポートを指定します。

注意

IJServerワークユニットのタイプが「EJBアプリケーションのみを運用する」の場合のみ負荷分散を使用することができます。

IJServerワークユニットを作成したときに、負荷分散ポリシーに設定した仮想IPアドレスに対応する仮想ホスト名と代表ポートを指定すると、作成したIJServerワークユニットに配備されたEJBアプリケーションの設定を自動的にIPCOMと連携可能な状態にセットアップします。また、IJServerワークユニットの環境設定から仮想ホスト名と代表ポートを指定したあとに更新すると、環境設定の変更ができます。

IJServerワークユニットの作成

Interstage管理コンソールの[ワークユニット]の新規作成画面で、EJBコンテナ設定の以下の項目を設定してください。
EJBコンテナ設定は、isj2eeadminコマンドでも設定できます。詳細は、「リファレンスマニュアル(コマンド編)」を参照してください。

項目		説明
IPCOMのメソッド負荷分散設定	する	「する」を選択して、仮想ホスト名と代表ポートを設定してください。 デフォルトは「しない」です。
	仮想ホスト名	IPCOMのサイト負荷分散ポリシーに設定した、仮想IPアドレスに対応する仮想ホスト名を設定してください。
	代表ポート	IPCOMのCORBAサービスに設定した代表ポートの値を設定してください。 有効範囲は1～65535です。 IPCOMのメソッド負荷分散設定の「する」を選択して代表ポートを省略した場合は、8002が設定されます。

Interstage管理コンソールについて詳細は、Interstage管理コンソールのヘルプを参照してください。

EJBアプリケーションの配備、IJServerワークユニットの更新が失敗したときには、Interstage管理コンソールに詳細なメッセージが出力されます。「J2EE ユーザーズガイド(旧版互換)」の「EJBサービス使用時の異常」を参照して対処してください。

注意

IJServerワークユニットのEJBアプリケーションに対する負荷分散においては、メソッド呼び出し単位の負荷分散のみ実施できます。

3.3.1.2 アプリケーションの配備

Webアプリケーション、EJBアプリケーションともに留意点はありません。

配備手順

Interstage管理コンソールの[ワークユニット] > ［ワークユニットごとの操作］ > ［IJServer］の[IJServer：配備]の画面で行います。詳細は、Interstage管理コンソールのヘルプを参照してください。

3.3.1.3 プログラミング方法

サーバ側の設計

特に留意点はありません。

クライアント側の設計

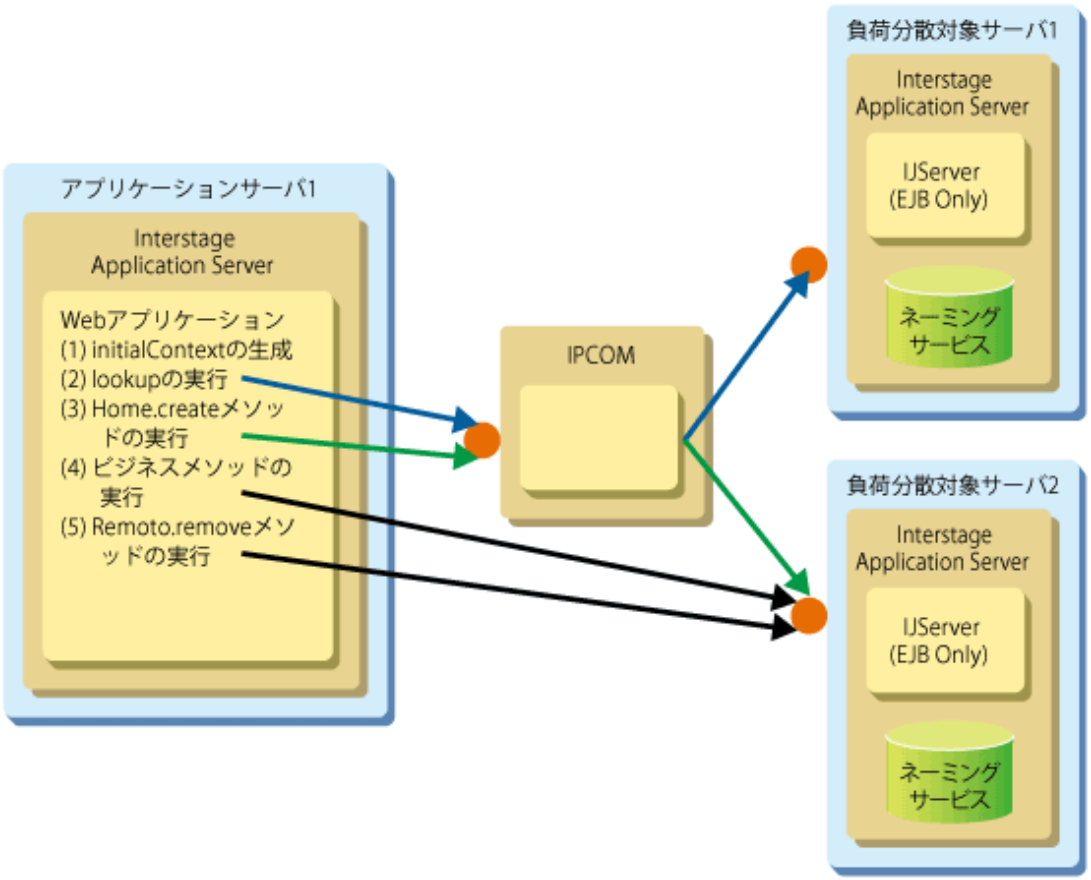
負荷分散方式ごとに説明します。

メソッド呼出し単位の負荷分散

EJBアプリケーションでメソッド呼出し単位の負荷分散を使用する場合において、負荷分散されるタイミングと、クライアント側のプログラミングの留意点について説明します。

負荷分散されるタイミング

負荷分散対象サーバは、Homeインタフェースのメソッドが実行されたタイミングで決定されます。Homeインタフェースのメソッドで EJB objectを取得し、EJB objectに対して処理を実行すると、Homeインタフェースのメソッドが割り振られたサーバで処理が実行されます。
Session Beanを呼び出す場合の処理において、createメソッドを実行後、Remoteインタフェースのremoveメソッドを実行するまで、同一のサーバに振り分けることが保証されます。createメソッドからremoveメソッド間に発行されるビジネスメソッドは常に同一のサーバに割り振られます。



プログラミングの留意点

「負荷分散されるタイミング」で説明したように、以下のようにプログラミングされたクライアントアプリケーションでは負荷分散されません。

- createメソッドを1回実行して、その後の処理は同一EJB objectに対して処理を実行する。

したがって、新規に処理を実行する場合は、createメソッドから再実行し、処理が分散されるようにプログラミングしてください。

3.3.2 CORBAアプリケーション

3.3.2.1 プログラミング方法

サーバ側の設計

特に留意点はありません。

クライアント側の設計

クライアントアプリケーションにおける留意点について説明します。

メソッド呼び出し単位の負荷分散

クライアントから呼び出すオブジェクトの振り分け先サーバは、クライアントからのオブジェクト呼出し毎に決定します。プログラミング上の留意事項はありません。

ネーミングサービスのオブジェクトリファレンス獲得時点の負荷分散

クライアントから呼び出すオブジェクトの振り分け先サーバは、ネーミングサービスのオブジェクトリファレンスの獲得時に決定します。オブジェクトのオブジェクトリファレンスは、獲得したネーミングサービスのオブジェクトリファレンスを使用して獲得するため、クライアントがネーミングサービスのオブジェクトリファレンスの獲得を再実行するまでは、クライアントからのオブジェクト呼出しは、同じサーバに振り分けられることになります。
振り分け先のサーバが停止した場合の再振り分けや、停止していたサーバが復旧した場合の切り戻しなどを行うため、クライアントアプリケーションは以下のように作成することをお勧めします。

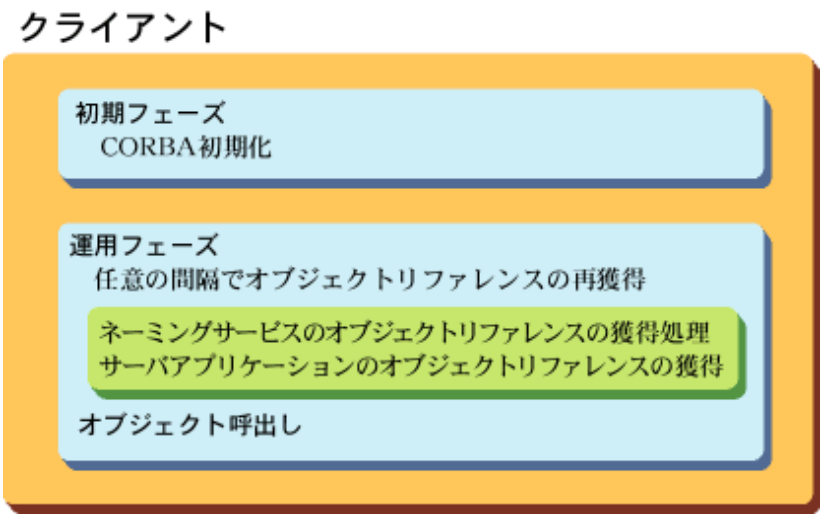
- 任意の間隔ごとにネーミングサービスのオブジェクトリファレンスの獲得処理を行う
- 定期的に獲得したネーミングサービスのオブジェクトリファレンスを使用し、オブジェクトのオブジェクトリファレンスの獲得処理を行う

例外処理

IPCOMと連携した通信においては通常発生しうる例外に加え、IPCOM特有の以下の例外が返る場合があります。

- システム例外:COMM_FAILURE
- マイナーコード:0x464a0114, 0x464a0115, 0x464a0116, 0x464a0914, 0x464a0915, 0x464a0916

上記のマイナーコードは、IPCOMにおいて異常を検出した場合に復帰します(負荷分散対象のサーバがすべて停止した場合や、IPCOMで通信時の異常を検出した場合など)。
詳細については「メッセージ集」の「CORBAサービスのマイナーコード」を参照してください。



注意

以下の機能は使用することができません。

使用できない機能	使用した場合の動作
CORBAアプリケーション情報定義ファイルにiswitch=ONを指定した運用	CORBAアプリケーション情報定義ファイルにiswitch=ONを指定した運用は動作しません。IPCOMにより振り分けられます。

3.3.3 トランザクションアプリケーション Windows32 Linux32

3.3.3.1 プログラミング方法

「CORBAアプリケーション」の「3.3.2.1 プログラミング方法」を参照してください。

注意

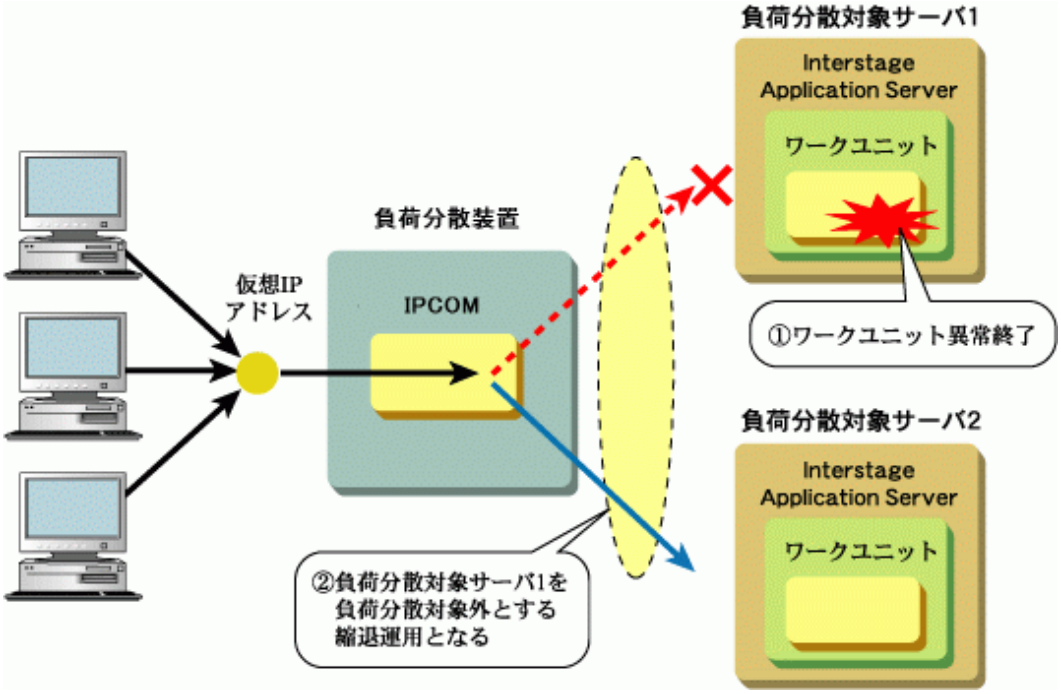
以下の機能は使用することができません。

使用できない機能	使用した場合の動作
コンポーネントトランザクションサービスのプロセスバインド機能	プロセスバインド機能は動作しません。IPCOMにより振り分けられ、クライアントアプリケーションにマイナーコード10007が復帰します。

3.4 運用方法

3.4.1 ワークユニット停止時の縮退運用

IPCOMと連携したワークユニットに対する負荷分散システムでは、任意の監視対象ワークユニットが停止した場合に、そのサーバを負荷分散対象サーバから切り離した縮退運用が行えます。負荷分散の対象となるワークユニットおよび業務上の重要なワークユニットを監視対象とします。
以下の図ではワークユニット異常終了時の縮退運用について説明します。



監視対象ワークユニットの設定方法

IPCOMによるワークユニットの稼動状態監視の設定方法について説明します。

当該ワークユニットに対する監視の実施有無を指定します。本設定方法では、複数のワークユニットを監視対象として指定できます。
Interstage管理コンソールの[ワークユニット]＞[新規作成]または[ワークユニット]＞[環境設定]タブの、IJServerワークユニットの場合は[EJBコンテナ設定]、CORBAワークユニットの場合は[ワークユニット設定]で以下の設定を行ってください。
IJServerワークユニットの場合は、isj2eeadminコマンドでも設定できます。詳細は、「リファレンスマニュアル(コマンド編)」を参照してください。

項目		説明
IPCOMによるワークユニットの稼働状態監視	・する ・しない	IPCOMによるワークユニットの稼動状態監視の実施有無を指定します。 「する」を選択した場合、ワークユニットが停止した場合に、そのサーバを負荷分散対象サーバから切り離して縮退運用を行います。 「しない」を選択した場合、ワークユニットが停止した場合でも縮退運用は行われません。そのため、IPCOMのメソッド負荷分散を使用する場合は、ワークユニットの稼動状態監視に「する」を選択することを推奨します。 なお、IPCOMのメソッド負荷分散の対象とならないワークユニットについても、稼動状態監視を行うことができます。

Windows32 Linux32
トランザクションアプリケーションの場合は、ワークユニット定義の[Traffic Director Monitor Mode：IPCOM連携時の監視有無]に「YES」を設定してください。ワークユニット定義の詳細については、「OLTPサーバ運用ガイド」を参照してください。

注意

- ワークユニットのHotDeploy処理中や起動処理中、停止処理中にも縮退運用が行われます。
- IJServerワークユニットの場合、IJServerタイプが以下の場合のみ稼動状態監視を実施することができます。
  - EJBアプリケーションのみ運用

ー WebアプリケーションとEJBアプリケーションを別JavaVMで運用

3.4.2 サーバダウン時の動作

振り分け先のサーバがダウンしている場合、ビジネスメソッドは通信エラー(COMM_FAILURE)となります。この場合、create()メソッドから再実行してください。再実行時には、運用中のサーバへ振り分けられます。サーバが異常状態から復旧した場合、クライアントアプリケーションまたは、Webアプリケーションから再振り分けを行う必要があります。そのため、create()メソッドを再実行してください。

サーバダウンの復旧時には、ダウンしたサーバを保守フェーズに切り替え、サーバマシンを運用可能な状態(Interstage、ワークユニットを起動し、運用環境が整った状態)にした後、保守フェーズを解除してください。

1. サーバダウン
2. 保守フェーズへ切り替え
3. ダウン原因を取り除く
4. Interstageの起動
5. ワークユニットの起動
6. 保守フェーズの解除

第4章 クラスタサービス機能

本章では、クラスタサービスの機能使用時の環境設定および運用方法を説明します。

参照

- Interstage HTTP Server 2.2のクラスタサービス機能使用時の環境設定については、「Interstage HTTP Server 2.2 運用ガイド」の「運用・保守」－「クラスタサービス」を参照してください。
- Java EE 7のクラスタサービス機能使用時の環境設定については、「Java EE 7 設計・構築・運用ガイド」の「高信頼性システムの運用」を参照してください。

4.1 クラスタサービスの構成

Interstage Application Serverのクラスタサービス機能を使用して運用する場合の資源構成について説明します。
クラスタシステム上でInterstageを運用する場合には、基本的に、運用ノード、待機ノードのそれぞれのサーバに、同一の資源構成でInterstage資源を配置します。ただし、クラスタシステムを構成するサーバ間で共有しなければならない資源に限って共用ディスク装置上に配置します。

Interstageのインストール資源や各種定義資源、アプリケーション資源などは、運用ノード、待機ノードに同一の資源構成で配置します。また、以下の資源に関しては共用ディスク装置上で管理します。

- トランザクションログファイル
- イベントサービスのイベントチャネルの不揮発化情報関連ファイル
- イベントサービスのユニットの不揮発化ファイル
- Interstage JMSのJMS不揮発化ファイル
- MessageQueueDirectorの制御用ファイル
- Interstage ディレクトリサービスのデータベース
- Interstage ディレクトリサービスのアクセスログ
- Servletサービスのセションリカバリ機能の永続化ファイル
- Interstage シングル・サインオンの暗号化情報(サービスID)ファイル(セションの管理を行う場合だけ)

参照

MessageQueueDirectorをクラスタサービスで使用する方法については、「MessageQueueDirector説明書」を参照してください。

4.2 サーバの環境設定

Interstage Application Serverのクラスタサービスの環境設定について説明します。
なお、クラスタサービス上で運用するアプリケーション環境の構築方法については、「4.4 アプリケーション環境作成」を参照してください。

環境設定作業の流れを以下で説明します。

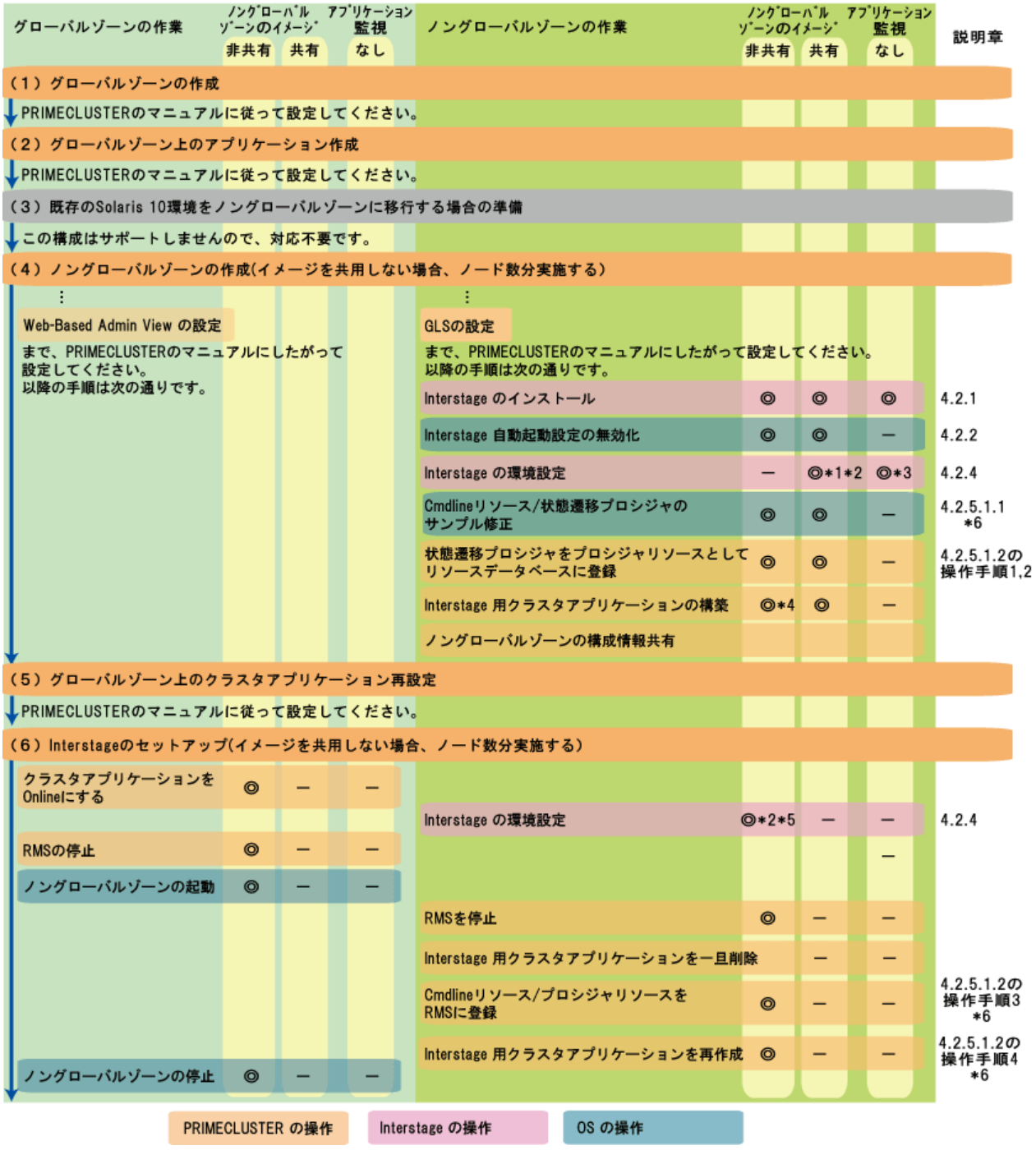
	手順	作業概要
1	Interstageのインストール	運用系ノード、待機系ノードに対して、それぞれInterstageをインストールします。
2	Interstage自動起動設定の無効化	Interstageを構成するサービスの自動起動設定を無効化します。
3	クラスタシステムの事前設定	共有ディスクや、データベース環境など、事前に準備が必要となる作業について説明します。
4	Interstageの環境設定	Interstageの初期化などの環境設定方法について説明します。
5	クラスタサービスの設定	PRIMECLUSTERやWSFCなど、クラスタ製品に対する環境設定手順を説明します。
6	クラスタシステムの動作確認	クラスタシステムを動作させ、環境が正しく構築できているかを確認します。

Solarisコンテナ環境では、「PRIMECLUSTER 導入運用手引書」に示されている構成1～7のうち、構成5で”ノングローバルゾーンイメージの配置”が”非共有”を除く構成をサポートします。

Solarisコンテナ環境の構築の流れは以下の図の通りです。PREIMECLUSTERのリソース構成に応じて、◎のついた手順を実施してください。コンテナ環境でのシングルノードクラスタ運用の場合は、Interstageについては「ノングローバルゾーンのイメージを共有」する手順で設定してください。

(6)の手順のあと、グローバルゾーンのRMSを起動すると、ノングローバルゾーンのInterstageが起動します。

なお、以下の図では、PRIMECLUSTERのマニュアルに合わせて、"global zone"、"non-global zone"をそれぞれ”グローバルゾーン”、”ノングローバルゾーン”と表記しています。また、”アプリケーションの監視”の”アプリケーション”は、ノングローバルゾーン上のInterstageを意味します。



◎: 必須、ー:設定しません、空欄:「PRIMECLUSTER 導入運用手引書」に従って設定します

*1) 「4.2.4 Interstageの環境設定」に従って設定します。クラスタアプリケーションを起動する必要ありません。

*2) マルチ言語サービスやJ2EE互換を利用する場合でも、ノングローバルゾーンの代表ホスト名([hostname]コマンドで表示される名前)に対応づけられたIPアドレスと、引継ぎIPアドレスが同じである構成の場合、「4.2.4.2 Interstage事前処理」の手順は行う必要はありません。

たとえば、

[ノングローバルゾーンのイメージを共有するケース、かつ、排他的IPゾーンの場合]

・ノングローバルゾーンのネットワークを二重化して仮想ネットワークインターフェースを作成する場合(つまり、ノングローバルゾーンにGlsリソースを作成する場合)は、Interstage事前処理が必要です。

・ノングローバルゾーンのネットワークを二重化せず、ノングローバルゾーンの代表ホスト名に対応づけられたIPアドレスを引継ぎIPアドレスとして使用する場合(つまり、ノングローバルゾーンにGlsリソースを作成しない場合)、Interstage事前処理は不要です。

[ノングローバルゾーンのイメージを共有しないケース、かつ、共有IPゾーンの場合]

・グローバルゾーンの仮想ネットワークインターフェースに対応するノングローバルゾーンのネットワークインターフェースに、ノングローバルゾーンの代表ホスト名に対応するIPアドレスを割り当てる場合、Interstage事前処理は不要です。

・グローバルゾーンの仮想ネットワークインターフェースに対応するノングローバルゾーンのネットワークインターフェースに、ノングローバルゾーンの代表ホスト名に対応するIPアドレスを割り当てない場合、Interstage事前処理は必要です。

PRIMECLUSTERのリソース構成、ネットワークの構成、およびInterstageの利用機能に応じて、Interstage事前処理を行ってください。

*3) クラスタ構成でない、通常のセットアップを行ってください。

*4) ノングローバルゾーンのイメージを共用しない場合は、Fsystemリソース(Intrerstageが共用ディスクを使用する場合)とGlsリソース(排他的IPゾーンの場合)のみを持つクラスタアプリケーションを作成します。

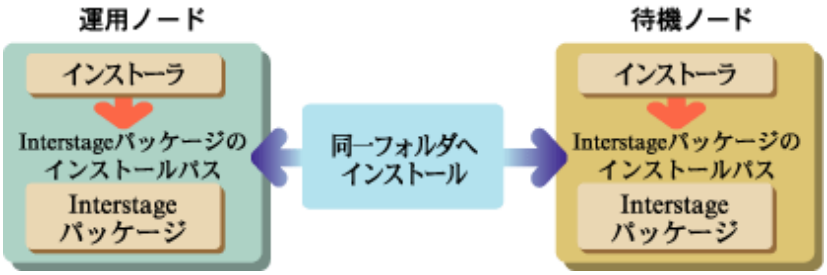
*5)「4.2.4 Interstageの環境設定」に従って設定します。グローバルゾーンのクラスタアプリケーションをOnlineにすることで、ノングローバルゾーンのクラスタアプリケーションがOnline(起動)になります。

*6) CmdlineリソースはJava EE 7をPRIMECLUSTERと連携するために使用します。Cmdlineリソースの修正、登録、およびJava EE 7用のクラスタアプリケーションの作成については、「Java EE 7 設計・構築・運用ガイド」の「高信頼システムの運用」を参照してください。

各設定手順の詳細を、以降に説明します。

4.2.1 Interstageのインストール

クラスタを構成する各ノード(運用ノード、待機ノード)のローカルディスク上に、Interstageをインストールします。
各ノードの資源構成を一致させるために、クラスタを構成する各ノード上の同一パスに製品をインストールしてください。



4.2.2 Interstage自動起動設定の無効化

Interstageのインストール後は、Interstageを構成するサービスが、マシン起動時に自動的に起動される設定となっています。
クラスタ環境では、マシン起動時にクラスタ製品がInterstageを起動しますので、Interstage独自の自動起動設定を無効化する必要があります。
以下に、無効化のための作業手順を説明します。

作業手順 Windows32/64

1. 以下のサービスの「スタートアップの種類」を確認し、「自動」が設定されている場合は「手動」に変更します。
2. 以下のサービスの状態を確認し、「開始」している場合には、サービスを停止します。

- OD_start
- ObjectTransactionService
- FJapache

- Interstage HTTP Server(Webサーバ名)
  注) Interstage HTTP ServerにおいてWebサーバを作成した場合に登録されます。
- Interstage Directory Service(リポジトリ名)
  注) リポジトリを作成した場合に登録されます。
- INTERSTAGE

注意

上記サービスでインストールされていないものについては、対処不要です。

作業手順 Solaris64 Linux32/64

Interstageパッケージインストール後には、サーバ起動時に呼び出されるスタートシェルとサーバ停止時に呼び出されるストップシェルがシンボリックリンクファイルとして格納されています。クラスタを使用する場合は、当該シェルが呼び出されないように、別ディレクトリなどに退避します。

Solaris64

以下のファイルを退避してください。

格納先ディレクトリ	退避対象ファイル(スタートシェル、ストップシェル)
/etc/rc0.d	K00stopis、K00stopod、K00FJSVirep、K01FJSVijdas、K17FJapache
/etc/rc1.d	K17FJapache
/etc/rc2.d	S98FJSVijdas、S99startis、S99startod、S99FJSVirep、K17FJapache
/etc/rc3.d	S51FJapache
/etc/rcS.d	K17FJapache

Linux32/64

- RHEL6以前

  以下のファイルを退避してください。

格納先ディレクトリ	退避対象ファイル(スタートシェル、ストップシェル)
/etc/rc0.d	K00stopis、K18FJSVirep、K14FJapache
/etc/rc1.d	K00stopis、K18FJSVirep、K14FJapache
/etc/rc2.d	S99startis、S82FJSVirep、S99startod、S86FJapache
/etc/rc3.d	S99startis、S82FJSVirep、S99startod、S86FJapache
/etc/rc4.d	S99startis、S82FJSVirep、S99startod、S86FJapache
/etc/rc5.d	S99startis、S82FJSVirep、S99startod、S86FJapache
/etc/rc6.d	K00stopis、K18FJSVirep、K14FJapache

- RHEL7以降

  以下のサービスについて自動起動/自動停止をしない設定にしてください。設定方法については「運用ガイド(基本編)」の「RHEL7でのサービス自動起動/自動停止」を参照してください。

  ー Interstage運用管理機能

  ー CORBAサービス

  ー Interstage HTTP Server

  ー Interstage ディレクトリサービス

注意

上記サービスでインストールされていないものに関しては、対処不要です。

4.2.3 クラスタシステムの事前設定

クラスタシステムのインストールやクラスタ初期化構成設定、Interstageが前提としている製品のインストールおよび環境設定を行います。ここでは、下記の設定時の留意点について説明します。これら手順の詳細は、クラスタ製品のマニュアルをご参照ください。

- 共用ディスク装置の設定
- ネットワーク(IPアドレス)の設定
- データベースの環境設定
- ノード名に関する注意事項

なお、ノード名引継ぎ(PRIMECLUSTERの機能)は使用できませんので注意してください。

注意

- 各クラスタシステムの詳細については、それぞれのマニュアルを参照してください。
- Interstageで使用する関連製品については、それぞれの製品の指示に従ってください。

共用ディスク装置の設定

以下の場合には共用ディスク装置が必要となります。

- データベース連携サービスを使用する場合
- ノーティフィケーションサービスの不揮発チャネル運用を行う場合
- Interstage ディレクトリサービスを使用する場合
- セション管理の運用を行うInterstage シングル・サインオンを使用する場合
- J2EEのServletサービスでセションリカバリ機能を使用する場合

共用ディスク(装置)は、ファイルシステムとして使用できるようにしてください。
また、共用ディスク装置については、Interstageの環境設定時に必要ですので、共用ディスクをクラスタサービス(PRIMECLUSTERではuserApplication)に登録を行い、Interstageの環境設定時に使用できるようにしてください。

ネットワーク(IPアドレス)の設定

InterstageではIPアドレスの引継ぎを前提とするため、引き継ぎネットワークの設定も事前に行ってください。
また、IPアドレスについては、Interstageの環境設定時に必要ですので、引継ぎIPアドレスをクラスタサービス(PRIMECLUSTERではuserApplication)に登録を行い、Interstageの環境設定前までに使用できるようにしてください。

データベースの環境作成

データベースを使用する場合は、データベースの環境作成も行ってください。

データベース連携サービスを利用する場合、システムログファイルを作成するための領域を確保してください。必要な領域の大きさについては、「リファレンスマニュアル(コマンド編)」のotsmklogコマンドを参照してください。

ノード名に関する注意事項 Solaris64 Linux32/64

PRIMECLUSTERではノード名引継ぎの機能をサポートしていますが、Interstageでは本機能を使用できません。
また、運用ノードと待機ノードに同一のノード名を設定しないでください。ノード名は以下の方法で確認してください。

Solaris64

/etc/nodenameの内容を確認してください。

Linux32/64 (RHEL6)

/etc/sysconfig/networkの「HOSTNAME」の設定値を確認してください。

Linux32/64 (RHEL7)

/etc/hostnameの内容を確認してください。

注意

Interstageの提供する状態遷移プロシジャはホットスタンバイ用のため、ノード名引継ぎ環境では正常に動作しません。

4.2.4 Interstageの環境設定

クラスタサービス上にInterstageの環境を構築するための手順を説明します。
本手順では、作業開始時点の運用ノードをノード1、待機ノードをノード2として説明します。また、ネットワーク(IPアドレス)や共用ディスク装置の設定は完了しているという前提で説明します。

1. クラスタサービス(PRIMECLUSTERではuserApplication)の起動
2. Interstage事前処理
3. Interstage初期化
4. 各サービスの環境設定

4.2.4.1 クラスタサービス(userApplication)の起動

Interstageのセットアップ時には、引継ぎIPアドレスの活性化が必要となります。また、共用ディスクを使用する場合は、共用ディスクも必要となります。
このため、以下の作業を行います。

1. Interstageが設定されるクラスタサービス(PRIMECLUSTERではuserApplication)を作成する。
2. 引継ぎIPアドレス、共用ディスクをクラスタサービスに登録する。
3. クラスタサービスを起動する。

注意

作業1、および2の方法や順序は、OSやクラスタシステムによって異なります。詳しくは使用しているクラスタシステムのマニュアルを参照してください。

4.2.4.2 Interstage事前処理

注意

本製品を標準インストールでインストールした場合、Interstage事前処理を行う必要はありません。
カスタムインストールで以下の機能をインストールしてクラスタシステム上で使用する場合は、Interstage事前処理を行ってください。

- マルチ言語サービス
- J2EE互換

運用ノード(ノード1)および待機ノード(ノード2)においてオブジェクトリファレンス生成時に埋め込むホスト名を、OD_set_envコマンドによって設定します。
このとき、ホスト名にはクラスタサービスで引き継がれるIPアドレスに対応するホスト名を設定する必要があります。
OD_set_envコマンドについては、「リファレンスマニュアル(コマンド編)」を参照してください。

例

```
OD_set_env -n vhost
```

上記の例では、クラスタサービスで引き継がれるIPアドレスに対応するホスト名としてvhostを使用しています。

4.2.4.3 Interstage初期化

注意

本製品を標準インストールでインストールした場合、Interstageの初期化を行う必要はありません。
カスタムインストールで以下の機能をインストールしてクラスタシステム上で使用する場合は、必要に応じて、Interstageの初期化を行ってください。

- マルチ言語サービス
- J2EE互換

1. Interstageの停止

Interstageが動作している場合は、Interstageを停止します。

例

```
isstop -f
```

2. Interstageシステム定義ファイルの生成

運用ノード(ノード1)および待機ノード(ノード2)でInterstageシステム定義ファイルを生成します。

なお、あらかじめ、クラスタ上で動作するInterstageのシステム規模を決定する必要があります。
システム規模については、「運用ガイド(基本編)」の「Interstage統合コマンドによる運用操作」－「Interstageの環境設定」－「Interstageシステム定義ファイルの生成」を参照してください。

例

```
isgendef large
```

3. Interstageシステム定義ファイルの登録

運用ノード(ノード1)および待機ノード(ノード2)において、生成したInterstageシステム定義ファイルを登録します。

Interstageシステム定義ファイルの登録については、「運用ガイド(基本編)」の「Interstage統合コマンドによる運用操作」－「Interstageの環境設定」－「Interstageシステム定義ファイルの登録」を参照してください。

例

```
isregistdef
```

4. Interstage初期化

Interstage初期化は、運用ノード(ノード1)、待機ノード(ノード2)の順に行います。

5. 運用ノード(ノード1)でのInterstageの初期化

運用ノード(ノード1)においてInterstageの初期化を行います。

Interstage初期化(isinitコマンド実行)の際、使用機能により事前にInterstage動作環境定義の設定を行う必要があります。また、構成/使用機能などを考慮して運用形態を指定します。
Interstage初期化の詳細については、「運用ガイド(基本編)」の「Interstage統合コマンドによる運用操作」－「Interstageの環境設定」－「Interstageの初期化」を参照してください。

注意

- データベース連携サービスを利用する場合は、Interstage動作環境定義ファイルの「OTS path for system log」に共用ディスクへのパスを指定してください。また、システムログファイルは共用ディスク上に作成してください。
- Solaris64 Linux32/64
  PRIMECLUSTERを使用する場合は、以下の点に注意してください。
  Interstage動作環境定義には、CORBAサービスが使用するホスト名(Corba Host Name)を指定しないでください。

例

```
isinit -f type1 ejb
```

6. 待機ノード(ノード2)でのInterstageの初期化

運用ノード(ノード1)のInterstageの初期化が完了したら、次に、クラスタの切り替えを行い、ノード2を運用ノードにした後、Interstageの初期化を行います。
Interstageの初期化時には、Interstage動作環境定義の設定や運用形態を、ノード1の初期化時と一致させます。
また、isinitコマンド実行時には「-w」を指定します。

例

```
isinit -f -w type1 ejb
```

7. 設定の確認

Interstageの初期化によって、クラスタ用のCORBAサービス環境設定が正しく行われたことを確認します。

ノード1、ノード2のそれぞれで、以下の確認を行ってください。

1. Interstageを起動する。
2. OD_or_admコマンドを実行し、ホスト/IPアドレス情報を確認します。コマンド実行結果より、「NameService」、「InterfaceRepository」等のホスト名が「4.2.4.2 Interstage事前処理」で設定したホスト名となっていれば正しく設定が行われたと判断できます。
   なお、OD_or_admコマンドの詳細については、「リファレンスマニュアル(コマンド編)」を参照してください。

例

```
> isstart
> OD_or_adm -l
ImplementationRepository      IDL:FJ/ImplementationRep:1.0      (自ホスト名:8002:1.0:)
FJ_LightInterfaceRepository   IDL:FJ/Repository:1.0             (自ホスト名:8002:1.0:)
FJ_ORB_admin                  IDL:OM_ORB/admin:1.0              (自ホスト名:8002:1.0:)
nill_oref
InterfaceRep                  IDL:CORBA/InterfaceRep:1.0        (vhost:8002:1.0:)
InterfaceRepLock              IDL:CORBA/IrOBF/backup:1.0        (vhost:8002:1.0:)
InterfaceRepository           IDL:CORBA/Repository:1.0          (vhost:8002:1.0:)
NameService                   IDL:CosNaming/NamingContextExt:1.0 (vhost:8002:1.0:)
InterfaceRep_e                IDL:CORBA/InterfaceRep:1.0        (vhost:8002:1.0:)
InterfaceRepository_e         IDL:CORBA/Repository:1.0          (vhost:8002:1.0:)
```

注) プラットフォームによって、表示される情報は異なります。

4.2.4.4 各サービスの環境設定

Interstageの使用機能によっては、機能ごとに環境設定を行う必要があります。以下の機能を使用している場合には、各々の環境設定手順を実施してください。

- Interstage HTTP Serverを使用する場合
- IJServerワークユニットを使用する場合
- Servletサービスのセションリカバリ機能を使用する場合
- ノーティフィケーションサービスの不揮発チャネル運用を行う場合
- イベントサービスおよびノーティフィケーションサービスの揮発チャネル運用を行う場合
- データベース連携サービスを使用する場合
- Interstage JMSを使用する場合
- Interstage シングル・サインオンを使用する場合
- Interstage証明書環境を使用する場合
- Interstage ディレクトリサービスを使用する場合

4.2.4.4.1 Interstage HTTP Serverを使用する場合

Interstage HTTP Serverを使用する場合に必要な設定について、以下に説明します。

環境設定方法

Interstage HTTP Serverの環境定義を運用ノード(ノード1)、待機ノード(ノード2)それぞれのローカルディスク上に設定します。このとき、それぞれ同一の環境定義を行う必要があります。
また、このとき、運用ノード(ノード1)と待機ノード(ノード2)でInterstage HTTP Serverの環境定義ファイルは、同一のディレクトリ構成とする必要があります。

各ノードで作成したInterstage HTTP Serverの環境定義ファイル(httpd.conf)のServerNameディレクティブに設定するホスト名には、DNS(Domain Name Server)に登録されている名前(共通のホスト名またはIPアドレス)を設定してください。
Interstage HTTP Serverの環境定義ファイルの設定については、「Interstage HTTP Server 運用ガイド」の「環境設定」および「ディレクティブ一覧」を参照してください。

注意事項

- クラスタシステム(PRIMECLUSTERの場合)上でInterstage HTTP Serverを運用する場合は、ismodifyserviceコマンドでInterstage運用環境にInterstage HTTP Serverのサービスの追加を設定し、Interstage統合コマンド(isstart／isstop)で起動／停止を行ってください。

```
ismodifyservice -a FJapache
```

- Interstage HTTP Serverの停止時にInterstageを停止して、クラスタシステム(PRIMECLUSTERの場合)を切り替える場合は、ismodifyserviceコマンドでInterstage稼働状態の監視モードに「mode1」を設定してください。

```
ismodifyservice -m mode1
```

- ポート番号には、システム上のアプリケーションを含むすべてのサービスにおいてそれぞれ異なるポート番号を設定する必要があります。万が一、同じポート番号を設定して運用した場合、待機ノードに切り替わる際、Webサーバの起動に失敗する可能性があります。ポート番号の設定については、「システム設計ガイド」の「ポート番号」を参照してください。

4.2.4.4.2 IJServerワークユニットを使用する場合

IJServerワークユニットを使用する場合、Interstageの初期化(isinitコマンド)での設定に加え、別途設定作業が必要になります。
手順を以下に説明します。

環境定義の設定

運用ノード(ノード1)、待機ノード(ノード2)でそれぞれIJServerワークユニットの環境定義を行います。このとき、運用ノード(ノード1)、待機ノード(ノード2)で同一の環境定義を行う必要があります。
また、各ノードで作成したIJServerワークユニットについては、Interstage起動時に自動起動しない設定としなければなりません。
自動起動しない設定ついては、Interstage管理コンソールの[ワークユニット] > [IJServerワークユニット名を選択] > [環境設定]タブ > [ワークユニット設定]の「ワークユニット自動起動」で設定します。
IJServerワークユニットの設定は、isj2eeadminコマンドでも設定できます。詳細は、「リファレンスマニュアル(コマンド編)」を参照してください。

注意事項

Interstage HTTP Serverを使用する場合、事前に「4.2.4.4.1 Interstage HTTP Serverを使用する場合」を参照してWebサーバの設定を行う必要があります。

4.2.4.4.3 Servletサービスのセションリカバリ機能を使用する場合

Session Registry Serverをクラスタスステム上で運用する場合、Interstageの初期化(isinitコマンド)での設定に加え、別途設定作業が必要になります。
手順を以下に説明します。

1. 環境定義の設定

Session Registry ServerはIJServerワークユニットとして運用します(業務用のWebアプリケーションを運用するIJServerワークユニットとは別に、Session Registry Server用のIJServerワークユニットを用意して運用します)。
Session Registry Serverを運用するIJServerワークユニットを、クラスタシステム上で運用できるように設定してください。IJServerワークユニットの設定については「4.2.4.4.2 IJServerワークユニットを使用する場合」を参照してください。

2. Session Registry Server環境定義ファイルの設定

Session Registry Serverをクラスタシステム上で運用する場合、セションの永続化を有効とすることで、待機ノードに切り替わった際にセション情報を引き継ぐことが可能となります。
セションの永続化を有効とするには、Session Registry Server環境定義ファイルに以下を定義します。

```
<param-name>session.store</param-name>
<param-value>on</param-value>
```

また、永続化ファイルの保存先は共用ディスクを指定します。
Session Registry Server環境定義ファイルに以下を定義します。

```
<param-name>serialize.file.path</param-name>
<param-value>$SWITCH</param-value>
```

$SWITCH:共用ディスク上のディレクトリ

4.2.4.4.4 ノーティフィケーションサービスの不揮発チャネル運用を行う場合

ノーティフィケーションサービスの不揮発チャネル運用を行う場合、Interstageの初期化(isinitコマンド)での設定に加え、別途設定作業が必要になります。
手順を以下に説明します。
なお、環境を再構築する場合は、「4.9.1 ノーティフィケーションサービスの不揮発チャネル運用時の注意事項」を参照し、ユニットおよびイベントチャネルを削除してから実行してください。

1.「イベントチャネルの不揮発化情報関連ファイル」を共用ディスクに作成

運用ノード(ノード1)でessetcnfコマンドにより「イベントチャネルの不揮発化情報関連ファイル」を共用ディスクに作成します。

例

```
essetcnf -f $SWITCH
```

$SWITCH:共用ディスク上のディレクトリ

注意

運用ノード(ノード1)でイベントサービスの構成情報を変更する場合は、再度essetcnfコマンドにより構成情報を設定してください。

例

```
essetcnf -s -edmax 5000
```

2. ユニット定義のファイルの準備

運用ノード(ノード1)で以下のディレクトリ配下にユニット定義を準備します。

Windows32/64

```
C:¥Interstage¥eswin¥etc¥def
```

Solaris64

```
/etc/opt/FJSVes/def
```

Linux32/64

```
/etc/opt/FJSVes/def
```

定義ファイル中のtrandir、sysdir、userdirは、共用ディスク上の不揮発用ファイルの格納ディレクトリを指定してください。また、trandirには、ローデバイスを指定しないでください。
なお、このディレクトリは、essetcnfコマンドの-fオプションで指定したディレクトリと関連はありません。

3. 不揮発用ファイルを共用ディスクに設定

運用ノード(ノード1)でesmkunitコマンドにより不揮発用のファイルを共用ディスク上に作成します。

例

```
esmkunit
```

4. Interstageの起動

運用ノード(ノード1)でisstartコマンドによりInterstageを起動します。

例

```
isstart
```

5. ユニットの起動

拡張ユニットを使用する場合は、運用ノード(ノード1)でesstartunitコマンドにより拡張ユニットを起動します。
なお、標準ユニットを使用する場合は、「4. Interstageの起動」において自動的に起動されるため、本操作を行う必要はありません。

例

```
esstartunit -unit ユニット名
```

6. イベントチャネルの作成

運用ノード(ノード1)でesmkchnlコマンドによりイベントチャネルを作成します。

例

```
esmkchnl -g グループ名 -c イベントチャネル名 -notify -persist all -unit ユニット名
```

注意

作成したイベントチャネルの動作環境を変更する場合は、ここでessetcnfchnlコマンドにより動作環境を設定してください。

7. ネーミングサービスからイベントチャネルのオブジェクトリファレンスを取得

運用ノード(ノード1)と待機ノード(ノード2)で同一のオブジェクトリファレンスを参照する必要があるため、待機ノード(ノード2)に運用ノード(ノード1)で作成したイベントチャネルのオブジェクトリファレンスを登録します。

運用ノード(ノード1)でesgetchnliorコマンドによりネーミングサービスからイベントチャネルのオブジェクトリファレンスを取得してください。指定したパス配下に、ファイル「(-gオプションに指定したグループ名).ior」が作成されます。

例

```
esgetchnlior -g グループ名 -p パス名
```

8. Interstageの停止

運用ノード(ノード1)でisstopコマンドによりInterstageを停止します。このとき、-fオプションを指定する必要があります。

例

```
isstop -f
```

9. クラスタサービスの切り替え

クラスタサービスを切り替えます。ノード2を運用ノードに、ノード1を待機ノードにします。
なお、Interstageの停止時にクラスタサービスが切り替わる設定をすでに行っている場合は、「8. Interstageの停止」においてクラスタサービスが切り替わるため、本操作を行う必要はありません。

10.「イベントチャネルの不揮発化情報関連ファイル」を共用ディスクに作成

運用ノード(ノード2)でessetcnfコマンドを、-wオプションを指定して実行し、「イベントチャネルの不揮発化情報関連ファイル」を共用ディスクに作成します。

例

```
essetcnf -f $SWITCH -w
```

$SWITCH:共用ディスク上のディレクトリ

注意

「1.「イベントチャネルの不揮発化情報関連ファイル」を共用ディスクに作成」において、ノード1でイベントサービスの構成情報を変更した場合は、運用ノード(ノード2)のイベントサービスでも再度essetcnfコマンドによりノード1と同様の構成情報を設定してください。

例

```
essetcnf -s -edmax 5000
```

11. ユニット定義ファイルの準備

運用ノード(ノード2)で以下のディレクトリ配下にユニット定義を準備します。

Windows32/64

```
C:¥Interstage¥eswin¥etc¥def
```

Solaris64

```
/etc/opt/FJSVes/def
```

Linux32/64

```
/etc/opt/FJSVes/def
```

注意

ユニット定義ファイルは、「2. ユニット定義のファイルの準備」で作成したものとすべて同じ設定を記述します。

12. 不揮発用ファイルを共用ディスクに設定

運用ノード(ノード2)でesmkunitコマンドを、-wオプションを指定して実行し、不揮発用ファイルを共用ディスクに設定します。

例

```
esmkunit -w
```

13. Interstageの起動

運用ノード(ノード2)でisstartコマンドによりInterstageを起動します。

例

```
isstart
```

14. ユニットの起動

拡張ユニットを使用する場合は、運用ノード(ノード2)でesstartunitコマンドにより拡張ユニットを起動します。
なお、標準ユニットを使用する場合は、「13. Interstageの起動」において自動的に起動されるため、本操作を行う必要はありません。

例

```
esstartunit -unit ユニット名
```

15. ネーミングサービスの登録

運用ノード(ノード2)と待機ノード(ノード1)で同一のオブジェクトリファレンスを参照する必要があるため、運用ノード(ノード2)に待機ノード(ノード1)で作成したイベントチャネルのオブジェクトリファレンスを登録します。

ノード1で取得したイベントチャネルのオブジェクトリファレンスを、essetchnliorコマンドにより運用ノード(ノード2)のネーミングサービスに登録してください。ファイル名には、「7. ネーミングサービスからイベントチャネルのオブジェクトリファレンスを取得」においてesgetchnliorコマンドで作成したファイル名を指定します。

例

```
essetchnlior -f ファイル名
```

16. イベントチャネルの作成

運用ノード(ノード2)でesmkchnlコマンドを、-wオプションを指定して実行し、イベントチャネルを作成します。

例

```
esmkchnl -g グループ名 -c イベントチャネル名 -notify -persist all -unit ユニット名 -w
```

注意

「6. イベントチャネルの作成」において、ノード1で作成したイベントチャネルの動作環境を変更した場合は、運用ノード(ノード2)で作成したイベントチャネルでもessetcnfchnlコマンドによりノード1と同様の動作環境を設定してください。

17. Interstageの停止

運用ノード(ノード2)でisstopコマンドによりInterstageを停止します。このとき、-fオプションを指定する必要があります。

例

```
isstop -f
```

4.2.4.4.5 イベントサービスおよびノーティフィケーションサービスの揮発チャネル運用を行う場合

イベントサービスおよびノーティフィケーションサービスの揮発チャネル運用を行う場合、Interstageの初期化(isinitコマンド)での設定に加え、別途設定作業が必要になります。
手順を以下に説明します。
なお、環境を再構築する場合は、「4.9.2 イベントサービスおよびノーティフィケーションサービスの揮発チャネル運用時の注意事項」を参照し、イベントチャネルを削除してから実行してください。

1. イベントサービスの構成情報の設定

運用ノード(ノード1)でイベントサービスの構成情報を変更する場合は、essetcnfコマンドにより構成情報を設定します。

 例

```
essetcnf -s -edmax 5000
```

2. Interstageの起動

運用ノード(ノード1)でisstartコマンドによりInterstageを起動します。

 例

```
isstart
```

3. イベントチャネルの作成

運用ノード(ノード1)でesmkchnlコマンドによりイベントチャネルを作成します。

 例

ノーティフィケーションサービスの場合

```
esmkchnl -g グループ名 -c イベントチャネル名 -notify
```

イベントサービスの場合

```
esmkchnl -g グループ名 -c イベントチャネル名
```

 注意

作成したイベントチャネルの動作環境を変更する場合は、ここでessetcnfchnlコマンドにより動作環境を設定してください。

4. ネーミングサービスからイベントチャネルのオブジェクトリファレンスを取得

運用ノード(ノード1)と待機ノード(ノード2)で同一のオブジェクトリファレンスを参照する必要があるため、待機ノード(ノード2)に運用ノード(ノード1)で作成したイベントチャネルのオブジェクトリファレンスを登録します。

運用ノード(ノード1)でesgetchnliorコマンドによりネーミングサービスからイベントチャネルのオブジェクトリファレンスを取得してください。指定したパス配下にファイル「(-gオプションに指定したグループ名).ior」が作成されます。

例

```
esgetchnlior -g グループ名 -p パス名
```

5. Interstageの停止

運用ノード(ノード1)でisstopコマンドによりInterstageを停止します。このとき、-fオプションを指定する必要があります。

例

```
isstop -f
```

6. クラスタサービスの切り替え

クラスタサービスを切り替えます。ノード2を運用ノードに、ノード1を待機ノードにします。
なお、Interstageの停止時にクラスタサービスが切り替わる設定をすでに行っている場合は、「5. Interstageの停止」においてクラスタサービスが切り替わるため、本操作を行う必要はありません。

7. イベントサービスの構成情報の設定

「1. イベントサービスの構成情報の設定」においてノード1でイベントサービスの構成情報を変更した場合は、運用ノード(ノード2)でもessetcnfコマンドによりノード1と同様の構成情報を設定します。

例

```
essetcnf -s -edmax 5000
```

8. Interstageの起動

運用ノード(ノード2)でisstartコマンドによりInterstageを起動します。

例

```
isstart
```

9. ネーミングサービスの登録

運用ノード(ノード2)と待機ノード(ノード1)で同一のオブジェクトリファレンスを参照する必要があるため、運用ノード(ノード2)に待機ノード(ノード1)で作成したイベントチャネルのオブジェクトリファレンスを登録します。

ノード1で取得したイベントチャネルのオブジェクトリファレンスを、essetchnliorコマンドにより運用ノード(ノード2)のネーミングサービスに登録してください。ファイル名には、「4. ネーミングサービスからイベントチャネルのオブジェクトリファレンスを取得」においてesgetchnliorコマンドで作成したファイル名を指定します。

例

```
essetchnlior -f ファイル名
```

10. イベントチャネルの作成

運用ノード(ノード2)でesmkchnlコマンドによりイベントチャネルを作成します。このとき、-wオプションを指定する必要があります。

例

ノーティフィケーションサービスの場合

```
esmkchnl -g グループ名 -c イベントチャネル名 -notify -w
```

イベントサービスの場合

```
esmkchnl -g グループ名 -c イベントチャネル名 -w
```

注意

「3. イベントチャネルの作成」において、ノード1で作成したイベントチャネルの動作環境を変更した場合は、運用ノード(ノード2)で作成したイベントチャネルでもessetcnfchnlコマンドによりノード1と同様の動作環境を設定してください。

11. Interstageの停止

運用ノード(ノード2)でisstopコマンドによりInterstageを停止します。このとき、-fオプションを指定する必要があります。

例

```
isstop -f
```

4.2.4.4.6 データベース連携サービスを使用する場合

データベース連携サービスは1:1運用待機形態のみ利用可能です。
データベース連携サービスを使用する場合、Interstageの初期化(isinitコマンドでtype2を指定)での設定に加え、別途設定作業が必要になります。

注意

Windows32/64 Linux32/64

JTSを利用したJ2EEアプリケーションを使用する場合は、isinitコマンドでtype2とejbを指定する必要があります。詳細については、「リファレンスマニュアル(コマンド編)」を参照してください。

CORBAアプリケーション/トランザクションアプリケーションによるOTS利用時の手順

1. XA連携用プログラム/リソース管理プログラム/APMの作成

XA連携用プログラムおよびリソース管理プログラムの作成を行います。

当該クラスタサービスに対する運用ノード(ノード1)、待機ノード(ノード2)で作成する必要があります。
リソース管理プログラムは、otsmkxapgmコマンドでXA連携用プログラムを作成してから、otslinkrscコマンドでリソース管理プログラムを作成します。

Windows32 Linux32

トランザクションアプリケーションを利用する場合は、tdlinkapmコマンドでAPMを作成する必要があります。

それぞれのコマンドについては、「リファレンスマニュアル(コマンド編)」を参照してください。

2. リソース管理プログラムの登録

作成したリソース管理プログラムの登録を行います。登録については、当該クラスタサービスに対する運用ノード(ノード1)と待機ノード(ノード2)で登録する必要があります。また、その場合、作成したすべてのリソース管理プログラムの登録を行ってください。待機ノード(ノード2)での登録時には、コマンドのオプションに-wを付けるようにしてください。
なお、リソース管理プログラムの登録を行う前にInterstageを起動し、すべてのリソース管理プログラムの登録完了後にInterstageを停止するようにしてください。

1. 運用ノード(ノード1)の状態確認

   ノード1が運用状態になっていなければ、クラスタの切り替えを行い、運用状態に切り替えてください。

2. Interstageの起動

   例

   ```
   isstart
   ```

3. ノード1でのリソース管理プログラムの登録

   例

   ```
   otssetrsc -a -rf リソース定義ファイル
   ```

4. ノード1でのInterstageの停止

   例

   ```
   isstop -f
   ```

5. リソース定義ファイルのコピー

   ノード1から、以下に示すディレクトリ配下のファイルを、ノード2の同一ディレクトリ配下へ、コピーします。

   Windows32/64

   C:¥Interstage¥ots¥etc¥repository配下

   Solaris64

   /opt/FSUNots/etc/repository配下

   Linux32/64

   /opt/FJSVots/etc/repository配下

   注意

   コピー時は、運用ノード(ノード1)と待機ノード(ノード2)両方のInterstageが停止していることを確認してください。

6. ノードの切り替え

   クラスタの切り替えを行い、ノード2を運用状態に切り替えてください。

7. ノード2でのInterstageの起動

例

```
isstart
```

8. ノード2でのリソース管理プログラムの登録

例

```
otssetrsc -a -w -rf リソース定義ファイル
```

注意

otssetrscコマンドには、-wオプションを指定してください。指定しない場合、すでに登録されている旨のエラーが発生します。

9. ノード2でのInterstageの停止

例

```
isstop -f
```

10. ノードの切戻し

クラスタの切戻しを行い、初期状態に戻してください。

J2EEアプリケーションによるJTS利用時の手順 Windows32/64 Linux32/64

1. JTS用リソース定義ファイルの登録

JTS用リソース定義ファイルを登録します。

1. 運用ノード(ノード1)の状態確認

ノード1が運用状態になっていなければ、クラスタの切り替えを行い、運用状態に切り替えてください。

2. Interstageの起動

例

```
isstart
```

3. ノード1でのリソース定義ファイルの登録

例

```
otssetrsc -a -rf リソース定義ファイル
```

4. ノード1でのInterstageの停止

例

```
isstop -f
```

5. リソース定義ファイルのコピー

ノード1から、以下に示すディレクトリ配下のファイルを、ノード2の同一ディレクトリ配下へ、コピーします。

Windows32/64

C:¥Interstage¥ots¥etc¥repository配下

Linux32/64

/opt/FJSVots/etc/repository配下

注意

コピー時は、運用ノード(ノード1)と待機ノード(ノード2)両方のInterstageが停止していることを確認してください。

2. JTS用リソース定義ファイルの登録CLASSPATHの設定

JTS用のリソース管理プログラムを利用する場合は、リソースと連携するために必要となるクラスライブラリへのパスを以下のファイルに設定してください。

Windows32/64

C:¥Interstage¥ots¥etc¥RMP.properties

Linux32/64

/opt/FJSVots/etc/RMP.properties

クラスライブラリの詳細な内容については、「アプリケーション作成ガイド(データベース連携サービス編)」の「リソース管理プログラムの作成から起動まで」を参照してください。

例

Windows32/64

```
RecoveryTarget=
Classpath=C:¥oracle¥jdbc¥lib¥ojdbc6.jar
```

Linux32/64

```
RecoveryTarget=
Classpath=/oracle/jdbc/lib/ojdbc6.jar
```

4.2.4.4.7 Interstage JMSを使用する場合

Interstage JMSを使用する場合、メッセージの送受信に使用する静的イベントチャネルの運用を行うために、まずノーティフィケーションサービスの環境を設定する必要があります。

ノーティフィケーションサービスの環境設定は、運用形態により設定方法が異なります。

以下の機能を使用する場合は、「4.2.4.4.4 ノーティフィケーションサービスの不揮発チャネル運用を行う場合」を参照して、ノーティフィケーションサービスの環境を設定してください。

- Durable Subscription機能
- メッセージの不揮発化機能
- ローカルトランザクション機能
- グローバルトランザクション機能

上記の機能を使用しない場合は、「4.2.4.4.5 イベントサービスおよびノーティフィケーションサービスの揮発チャネル運用を行う場合」を参照して、ノーティフィケーションサービスの環境を設定してください。

なお、ユニットの作成、静的イベントチャネルの作成で指定するオプションの詳細については、「J2EE ユーザーズガイド(旧版互換)」の「イベントチャネル運用マシンの運用前の環境設定」を参照してください。

ノーティフィケーションサービスの環境設定後に行うInterstage JMSの環境設定の手順を以下に説明します。

1. 環境の設定

運用ノード(ノード1)でInterstage JMSの環境定義を行います。詳細については、「J2EE ユーザーズガイド(旧版互換)」の「JMSアプリケーション運用マシンの運用前の環境設定」を参照してください。

2. JMS不揮発化ファイルの設定

Durable Subscription機能を使用する場合は、jmssetupclusterコマンドを使用して、JMS不揮発化ファイルを共用ディスク上に作成します。

例

```
jmssetupcluster  $SWITCH
```

$SWITCH:共用ディスク上のディレクトリ

3. ノード切り替え

クラスタサービスの切り替えを行い、ノード2を運用ノードに、ノード1を待機ノードにします。

4. 環境の設定

運用ノード(ノード2)でInterstage JMSの環境定義を行います。この時、待機ノード(ノード1)と同一の環境設定を行う必要があります。ConnectionFactory定義、Destination定義を待機ノード(ノード1)と同一のJNDI名、オプションを指定して登録してください。

5. JMS不揮発化ファイルの設定

Durable Subscription機能を使用する場合は、jmssetupclusterコマンドに-wオプションを指定して実行します。

例

```
jmssetupcluster  $SWITCH  -w
```

$SWITCH:共用ディスク上のディレクトリ

4.2.4.4.8 Interstage シングル・サインオンを使用する場合

Interstage シングル・サインオンを使用する場合は、事前に以下の作業を行ってください。

- 「4.2.4.4.1 Interstage HTTP Serverを使用する場合」を参照してWebサーバの設定を行ってください。
- リポジトリサーバでSSL通信を行う場合は、「4.2.4.4.9 Interstage証明書環境を使用する場合」を参照してInterstage証明書環境を構築してください。

また、上記作業に加えて、別途設定作業が必要になります。手順を以下に説明します。

ポイント

Interstage シングル・サインオンでは、リポジトリサーバ(リポジトリサーバが複数のマシンで構成されている場合は更新系)のみクラスタシステムに対応しています。

環境構築方法

1. 運用ノード(ノード1)での構築

1. 運用ノード(ノード1)において、Interstage管理コンソールを使用してSSOリポジトリを構築します。SSOリポジトリの構築については「4.2.4.4.10 Interstage ディレクトリサービスを使用する場合」を参照してください。
2. 運用ノード(ノード1)において、Interstage管理コンソールを使用してリポジトリサーバを構築します。リポジトリサーバの構築については「シングル・サインオン運用ガイド」の「リポジトリサーバの構築」を参照してください。
3. セション管理の運用を行うInterstage シングル・サインオンを使用する場合は、Interstage シングル・サインオン専用のパスを共用ディスクに用意します。

   例

   Windows32/64

   E:¥sso

   Solaris64 Linux32/64

   /sso

4. 共用ディスクに作成したInterstage シングル・サインオン専用のパスに、リポジトリサーバの環境定義ファイルのserviceidpathに指定されている暗号化情報(サービスID)ファイルを移動してください。

   注意

   セション管理の運用を行わない場合は、作業を行う必要はありません。

   例

   Windows32/64

   リポジトリサーバの環境定義ファイル(C:¥Interstage¥F3FMsso¥ssoatcsv¥conf¥ssoatcsv.conf)が以下のように設定されていた場合、暗号化情報(サービスID)ファイル(C:¥Interstage¥F3FMsso¥ssoatcsv¥conf¥serviceid)を、共用ディスクのInterstage シングル・サインオン専用のパスに移動します。

   ```
   serviceidpath=C:¥Interstage¥F3FMsso¥ssoatcsv¥conf¥serviceid
   ```

Solaris64 Linux32/64

リポジトリサーバの環境定義ファイル(/etc/opt/FJSVssosv/conf/ssoatcsv.conf)が以下のように設定されていた場合、暗号化情報(サービスID)ファイル(/etc/opt/FJSVssosv/conf/serviceid)を、共用ディスクのInterstage シングル・サインオン専用のパスに移動します。

```
serviceidpath=/etc/opt/FJSVssosv/conf/serviceid
```

5. リポジトリサーバの環境定義ファイル、およびセション管理を行うための定義ファイルに設定されているserviceidpathを、手順3で暗号化情報(サービスID)ファイルを移動したパスに変更してください。

注意

- セション管理の運用を行わない場合は、作業を行う必要はありません。
- serviceidpath以外は編集しないでください。serviceidpath以外を編集した場合、リポジトリサーバが正しく動作しなくなる場合があります。環境定義ファイルを編集する際には、十分注意してください。

例

Windows32/64

暗号化情報(サービスID)ファイルを共用ディスクの「E:¥sso」に移動した場合
リポジトリサーバの環境定義ファイル(C:¥Interstage¥F3FMsso¥ssoatcsv¥conf¥ssoatcsv.conf)、およびセション管理を行うための定義ファイル(C:¥Interstage¥F3FMsso¥ssoatcsv¥conf¥ssossmgr.conf)に設定されているserviceidpathを以下のように変更します。

```
serviceidpath=E:¥sso¥serviceid
```

Solaris64 Linux32/64

暗号化情報(サービスID)ファイルを共用ディスクの「/sso」に移動した場合
リポジトリサーバの環境定義ファイル(/etc/opt/FJSVssosv/conf/ssoatcsv.conf)、およびセション管理を行うための定義ファイル(/etc/opt/FJSVssosv/conf/ssossmgr.conf)に設定されているserviceidpathを以下のように変更します。

```
serviceidpath=/sso/serviceid
```

6. 構築したリポジトリサーバの資源、およびInterstage HTTP Serverの資源をバックアップします。バックアップについては「運用ガイド(基本編)」の「資源のバックアップとリストア」を参照してください。
7. 構築したSSOリポジトリの資源をバックアップします。バックアップについては「4.2.4.4.10 Interstage ディレクトリサービスを使用する場合」を参照してください。

2. 待機ノード(ノード2)での構築

1. 運用ノード(ノード1)でバックアップしたSSOリポジトリの資源を、待機ノード(ノード2)にリストアします。リストアについては「4.2.4.4.10 Interstage ディレクトリサービスを使用する場合」を参照してください。
2. 運用ノード(ノード1)でバックアップしたリポジトリサーバの資源、およびInterstage HTTP Serverの資源を、待機ノード(ノード2)にリストアします。リストアについては「運用ガイド(基本編)」の「資源のバックアップとリストア」を参照してください。

注意

- 構築後に運用ノード(ノード1)において、リポジトリサーバの環境設定を変更する場合は、待機ノード(ノード2)においても同様に変更し、リポジトリサーバの環境設定を同一にしてください。

- すでにクラスタシステムとして構成したリポジトリを「SSOリポジトリ」として使用する場合は、待機ノード(ノード2)のInterstage管理コンソールを使用して、[システム] > [セキュリティ] > [シングル・サインオン] > [認証基盤] > [リポジトリサーバ] > [環境設定]タブの[適用]ボタンをクリックしてください。
- ポート番号には、システム上のアプリケーションを含むすべてのサービスにおいてそれぞれ異なるポート番号を設定する必要があります。万が一、同じポート番号を設定して運用した場合、リポジトリサーバが正常に起動しません。ポート番号の設定については、「システム設計ガイド」の「ポート番号」を参照してください。

環境削除方法

1. 運用ノード(ノード1)において、Interstage管理コンソールを使用してリポジトリサーバを削除します。
2. クラスタサービスの切り替えを行い、待機ノード(ノード2)において、Interstage管理コンソールを使用してリポジトリサーバを削除します。
3. 共用ディスクに暗号化情報(サービスID)ファイルが存在するか確認してください。存在する場合は、暗号化情報(サービスID)ファイルを削除します。
   注) セション管理の運用を行わない場合は、作業を行う必要はありません。
4. 各ノードでSSOリポジトリを削除してください。SSOリポジトリの削除については「4.2.4.4.10 Interstage ディレクトリサービスを使用する場合」を参照してください。

注意

削除手順を誤った場合、クラスタサービスの切り替えに失敗する可能性があります。

4.2.4.4.9 Interstage証明書環境を使用する場合

Interstage証明書環境を使用するサービスにおいて、各ノードで同じサイト証明書を使用する場合には、サービスでの設定に加えて以下の手順が必要となります。

注意

認証局の運用方針によっては、異なるノードで同じサイト証明書を利用することを許可していない場合があります。また、条件付きで許可している場合もあります。そのため、想定している運用でサイト証明書が利用可能かを認証局に確認してから、以下の手順を実施してください。認証局で許可されていない場合は、ノードごとに異なるサイト証明書を利用する(ノードごとにInterstage証明書環境を構築する)か、または、想定している運用を許可している認証局からサイト証明書を入手するようにしてください。

環境設定方法

1. 運用ノードでのInterstage証明書環境の構築

運用ノードにおいて、Interstage証明書環境を構築します。Interstage証明書環境の構築方法については、「セキュリティシステム運用ガイド」の「Interstage証明書環境の構築と利用」を参照してください。

2. 待機ノードでのInterstage証明書環境の構築

待機ノードにおいては、以下の手順でInterstage証明書環境を構築します。
なお、各コマンドの詳細については、「リファレンスマニュアル(コマンド編)」を参照してください。

1. Interstage証明書環境の作成

   scsmakeenvコマンドで、Interstage証明書環境を作成します。このとき、CSRやテスト用証明書を作成しないため、-eオプションを指定してください。

   例

   Windows32/64

   ```
   scsmakeenv -e -c
   ```

   Solaris64 Linux32/64

   ```
   scsmakeenv -e -c -g iscertg
   ```

2. 認証局証明書の登録

   運用ノードに登録した認証局の証明書をすべて、scsenterコマンドで登録してください。

   例

   Windows32/64

   ```
   scsenter -n ca -f C:¥my_folder¥CA.der
   ```

   Solaris64 Linux32/64

   ```
   scsenter -n ca -f /usr/home/my_dir/CA.der
   ```

3. サイト証明書の移出(取り出し)

   運用ノードのInterstage証明書環境から、サイト証明書と秘密鍵をPKCS#12データで移出します。サイト証明書が複数ある場合は、すべてのサイト証明書を移出してください。

   例

   Windows32/64

   ```
   scsexppfx -n sitecert -f C:¥my_folder¥MyCert.p12
   ```

   Solaris64 Linux32/64

   ```
   scsexppfx -n sitecert -f /usr/home/my_dir/MyCert.p12
   ```

4. サイト証明書の移入(登録)

   3.で移出したPKCS#12データを、待機ノードのInterstage証明書環境に移入します。移出したPKCS#12データが複数ある場合は、すべてのPKCS#12データを移入してください。

   例

   Windows32/64

   ```
   scsimppfx -f C:¥my_folder¥MyCert.p12
   ```

   Solaris64 Linux32/64

   ```
   scsimppfx -f /usr/home/my_dir/MyCert.p12
   ```

5. 他サイト証明書の登録

   運用ノードに登録した他サイトの証明書をすべて、scsenterコマンドで登録してください。

   例

   Windows32/64

   ```
   scsenter -n othersite -e -f C:¥my_folder¥otherSite.der
   ```

   Solaris64 Linux32/64

   ```
   scsenter -n othersite -e -f /usr/home/my_dir/otherSite.der
   ```

6. CRLの登録

   運用ノードに登録したCRLをすべて、scsenterコマンドで登録してください。

   例

   Windows32/64

   ```
   scsenter -c -f C:¥my_folder¥crl.der
   ```

   Solaris64 Linux32/64

   ```
   scsenter -c -f /usr/home/my_dir/crl.der
   ```

注意事項

運用開始後に証明書やCRLを追加登録・削除する場合、すべてのノードで登録・削除を実施してください。

4.2.4.4.10 Interstage ディレクトリサービスを使用する場合

クラスタ環境でのリポジトリの作成手順、および操作手順を説明します。

- リポジトリの作成手順(スタンドアロン運用)
- レプリケーション導入方法
- リポジトリの更新手順
- リポジトリの削除手順
- リポジトリのバックアップ手順
- リポジトリのリストア手順
- メンテナンスの一括実行

注意

- Interstage ディレクトリサービスは、運用待機型にだけ対応しています。
- 以下に説明する手順は、各クラスタ環境で共通の操作手順です。管理者権限で実行してください。
- 以下に説明する手順は、すでにクラスタ環境が構築されていることを前提としています。
- ノード1、ノード2とも、ディレクトリ構成を完全に一致させてください。
- リポジトリの新規作成時に指定するアクセスログの格納先には、共用ディスクのパスを指定してください。

- 共用ディスクパスには、リソースに登録されているパスを指定してください。
- クラスタ環境ですでに運用されているリポジトリの環境を更新する場合は、「リポジトリの更新手順」に従ってください。
- リポジトリの新規作成時に指定するポート番号には、システム上のアプリケーションを含むすべてのサービスにおいてそれぞれ異なるポート番号を設定する必要があります。万が一、同じポート番号を設定して運用した場合、リポジトリが正常に起動しません。ポート番号の設定については、「システム設計ガイド」の「ポート番号」を参照してください。

Windows32/64

- Windows Server(R) フェールオーバー クラスタリング(WSFC)を利用したリソースは、「リポジトリの作成手順」のあとに登録してください。
- WSFCのクラスタ環境で運用されているリポジトリは停止できないため、リポジトリの更新／削除を行うことができません。
  WSFCのクラスタ環境で運用されているリポジトリの更新／削除を行う場合は、WSFCへ登録したリポジトリのリソースを削除した後、リポジトリを停止して行ってください。

参照

- irepbacksysコマンドの詳細については、「リファレンスマニュアル(コマンド編)」の「バックアップコマンド」－「irepbacksys」を参照してください。
- ireprestsysコマンドの詳細については、「リファレンスマニュアル(コマンド編)」の「バックアップコマンド」－「ireprestsys」を参照してください。
- バックアップの詳細については、「運用ガイド（基本編）」の「メンテナンス(資源のバックアップ/他サーバへの資源移行/ホスト情報の変更)」－「バックアップ手順詳細」－「Interstage ディレクトリサービス資源のバックアップ」を参照してください。

リポジトリの作成手順(スタンドアロン運用)

クラスタ環境でInterstage ディレクトリサービスのスタンドアロン運用(RDB)を行う場合は、以下の手順でリポジトリ作成します。
クラスタ環境の初期設定は、運用ノードをノード1、待機ノードをノード2とします。

1. RDBのクラスタシステムを構築します。

   #### 注意

   Symfoware Serverのクラスタシステムを構築する場合は、フェイルオーバ運用のスタンバイ機能を使用して構築してください。

   #### 参照

   ― Symfoware Serverのクラスタシステムの構築については、Symfoware Serverのマニュアル「クラスタ導入運用ガイド」の「フェイルオーバ運用のセットアップ」、および「フェイルオーバの運用」を参照してください。

   ― Oracleデータベースのクラスタシステムの構築については、Oracleデータベースのマニュアルを参照してください。

   ― RDBのクラスタシステムを構築する際の、データベースの環境構築時に指定する各種パラメタについては、「ディレクトリサービス運用ガイド」の「データベースの構築」に記載されている値を指定してください。

2. リポジトリを作成します。

   ― コマンドの場合

   irepconfigコマンドのcreateサブコマンドを使用します。

   - accesslog_dir (アクセスログの格納先)

     共用ディスクのパスを指定します。

ー Interstage管理コンソールの場合

[システム] > [サービス] > [リポジトリ] > [新規作成]タブ

- [アクセスログ定義]の[格納先]

  共用ディスクのパスを指定します。

3. 運用ノード(ノード1)で、irepbacksysコマンドを実行し、手順1.で作成したリポジトリの情報をバックアップします。

   Windows32/64
   ディレクトリにバックアップします。

   例

```
irepbacksys -d X:¥Backup¥irep¥rep001_back -R rep001 -confonly
```

   Solaris64 Linux32/64
   ファイルにバックアップします。

   例

```
irepbacksys -f /backup/irep/rep001_back -R rep001 -confonly
```

4. 手順3.で作成したバックアップファイルを、ftpなどを使用して待機ノード(ノード2)に転送します。

5. クラスタの切り替えを行い、ノード2を運用ノードにします。

6. 運用ノード(ノード2)で、irep restsysコマンドを実行し、手順1.で作成したリポジトリの情報をリストアします。

   Windows32/64
   ディレクトリからリストアします。

   例

```
irepresttsys -d X:¥Backup¥irep¥rep001_back -R rep001 -confonly
```

   Solaris64 Linux32/64
   ファイルからリストアします。

   例

```
irepresttsys -f /back/irep/rep001_back.tar.gz -R rep001 -confonly
```

7. 運用ノード(ノード2)で、リポジトリを起動します。

レプリケーション導入方法

クラスタ環境で、RDBを使用してInterstage ディレクトリサービスのレプリケーション運用を行う場合のリポジトリの作成手順は、スタンドアロン運用を行う場合の作成手順と同様です。

参照

RDBのレプリケーション機能を使用したクラスタシステムの構築手順については、各RDBのマニュアルを参照してください。

リポジトリの更新手順

クラスタ環境の初期設定を、運用ノードをノード1、待機ノードをノード2とします。

1. Windows32/64
   WSFCのクラスタ環境で運用されているリポジトリを更新する場合は、WSFCへ登録した対象リポジトリの汎用サービス・リソースをオフラインにします。
2. 運用ノード(ノード1)で、リポジトリを停止してから更新し、起動します。
3. クラスタの切り替えを行い、ノード2を運用ノードに切り替えてから、手順1と同じ操作を実行します。
4. Windows32/64
   WSFCのクラスタ環境で運用されているリポジトリを更新した場合は、手順1でオフラインにしたリソースをオンラインにします。

リポジトリの削除手順

クラスタ環境の初期設定を、運用ノードをノード1、待機ノードをノード2とします。

1. Windows32/64
   WSFCのクラスタ環境で運用されているリポジトリを削除する場合は、WSFCへ登録した対象リポジトリの汎用サービス・リソースを削除します。
2. 運用ノード(ノード1)で、リポジトリを停止してから削除します。
3. クラスタの切り替えを行い、ノード2を運用ノードに切り替えてから、手順1と同じ操作で実行します。

リポジトリのバックアップ手順

クラスタ環境で、リポジトリのバックアップを行う場合のバックアップ手順は、「運用ガイド(基本編)」の「メンテナンス(資源のバックアップ/他サーバへの資源移行/ホスト情報の変更)」－「資源のバックアップとリストア(クラスタ環境)」－「バックアップ手順(クラスタ環境の場合)」を参照してください。

リポジトリのリストア手順

クラスタ環境で、リポジトリのリストアを行う場合のリストア手順は、「運用ガイド(基本編)」の「メンテナンス(資源のバックアップ/他サーバへの資源移行/ホスト情報の変更)」－「資源のバックアップとリストア(クラスタ環境)」－「リストア手順(クラスタ環境の場合)」を参照してください。

メンテナンスの一括実行

クラスタ環境で、メンテナンスの一括実行を行う場合の詳細は、「運用ガイド(基本編)」の「メンテナンス(資源のバックアップ/他サーバへの資源移行/ホスト情報の変更)」－「メンテナンスの一括実行」－「バックアップ・リストア対象資源の定義方法」を参照してください。

4.2.5 クラスタサービスの設定

クラスタ製品に対して実施しなければならない設定手順について説明します。
Interstageで使用できるクラスタ製品ごとに設定手順を示しますので、使用するクラスタ製品に合わせて環境設定を行ってください。

- PRIMECLUSTERの場合
- WSFCの場合

4.2.5.1 PRIMECLUSTERの場合

クラスタシステムに関連する環境設定を行います。以下の作業を行います。
なお、クラスタサービスは、PRIMECLUSTERではuserApplicationになります。

- 状態遷移プロシジャの修正

  クラスタシステムの環境に応じて、Interstageの状態遷移プロシジャを修正します。

- 状態遷移プロシジャの登録

  状態遷移プロシジャをPRIMECLUSTERへ登録します。

  Solaris64

  PRIMECLUSTERのクラスタ運用設定ビューより行う必要があります。

  Linux32/64

  PRIMECLUSTERのCUIを使用して登録を行う必要があります。

4.2.5.1.1 状態遷移プロシジャの修正

状態遷移プロシジャの修正は、クラスタを構成する全てのノードで実施します。

状態遷移プロシジャのサンプル

Interstageでは状態遷移プロシジャを利用し、Interstageの起動/停止および切り替え処理を行います。Interstageでは以下のディレクトリ配下に状態遷移プロシジャのサンプルを提供します。使用者は状態遷移プロシジャを環境に合わせて修正する必要があります。

Solaris64

```
/opt/FJSVisas/etc/HA/SynfinityCLUSTER
```

Linux32/64

```
/opt/FJSVisas/etc/HA/PRIMECLUSTER
```

Interstageでは以下の状態遷移プロシジャのサンプルを提供します。
これらの状態遷移プロシジャのうち、使用している機能に該当する状態遷移プロシジャを使用します。

カテゴリ	サンプル名	制御内容
サービス操作用	IS_INTERSTAGE	Interstage(Interstage統合コマンドの起動/停止制御対象のサービス)の操作のために使用します。
	ES_INTERSTAGE	イベントサービスの操作のために使用します。
	IREP_INTERSTAGE	Interstage ディレクトリサービスの操作のために使用します。
	SSO_INTERSTAGE	Interstage シングル・サインオンの操作のために使用します。セション管理を行う場合のみ登録します。
	OTS_RMP_INTERSTAGE	データベース連携サービスリソース管理プログラムの操作のために使用します。

カテゴリ	サンプル名	制御内容
ワークユニット起動用	IJSERVER_INTERSTAGE	クラスタ切り替え時のIJServerワークユニットの起動操作のために使用します。
	ODWU_INTERSTAGE	クラスタ切り替え時のCORBAワークユニット起動操作のために使用します。
	UTYWU_INTERSTAGE	クラスタ切り替え時のユーティリティワークユニット起動操作のために使用します。
	Linux32 TDWU_INTERSTAGE	クラスタ切り替え時のトランザクションアプリケーションのワークユニット起動操作のために使用します。

サンプルの利用方法

使用者は以下の手順で状態遷移プロシジャの修正を行います。

1. 状態遷移プロシジャの複写
2. 状態遷移プロシジャの内容の修正

注意

修正を行った後、修正したプロシジャには実行権限を付与してください。

1. 状態遷移プロシジャの複写

Interstageより提供している状態遷移プロシジャのサンプルの内、使用している機能に該当する状態遷移プロシジャを使用者の任意のディレクトリ配下に以下の命名で複写してください。
なお、サンプルを複写するディレクトリはローカルディスク内に設定してください。また、プロシジャは各ノードで同一の位置に格納してください。

```
userApplication名.状態遷移プロシジャ名
```

2. 状態遷移プロシジャの内容の修正

複写した状態遷移プロシジャのうち、いくつかの状態遷移プロシジャに関しては、使用条件により修正を行う必要があります。
下表を参考に修正の実施有無を判断してください。また、修正が必要な場合には、以降の説明を参考に修正を実施してください。

カテゴリ	サンプル名	修正の必要有無	修正実施の条件
サービス操作用	IS_INTERSTAGE	○	Interstageに対する監視条件を変更する場合。または、Interstage HTTP Serverを運用するために必要なファイルディスクリプタ数が1024を超える場合。 Interstage HTTP Serverを運用するために必要なファイルディスクリプタ数の見積もりについては、「チューニングガイド」の「システムのチューニング」－「サーバ機能運用時に必要なシステム資源」－「Interstage HTTP Serverのシステム資源の設定」を参照してください。

カテゴリ	サンプル名	修正の必要有無	修正実施の条件
	ES_INTERSTAGE	○	状態遷移プロシジャからイベントチャネルを起動する場合。
	IREP_INTERSTAGE	○	Interstageディレクトリサービスを使用している場合。
	SSO_INTERSTAGE	－	修正の必要はありません。
	OTS_RMP_INTERSTAGE	○	OTSのリソース管理プログラムを使用している場合。
ワークユニット起動用	IJSERVER_INTERSTAGE	○	状態遷移プロシジャからIJServerワークユニットを起動する場合。
	ODWU_INTERSTAGE	○	状態遷移プロシジャからCORBAワークユニットを起動する場合。
	UTYWU_INTERSTAGE	○	状態遷移プロシジャからユーティリティワークユニットを起動する場合。
	Linux32 TDWU_INTERSTAGE	○	状態遷移プロシジャからトランザクションアプリケーションのワークユニットを起動する場合。

○:修正の必要あり －:修正の必要なし

複写した状態遷移プロシジャの内容の修正方法を以下に説明します。

userApplication名.IS_INTERSTAGEの修正

ネーミングサービスおよびインタフェースリポジトリが何らかの異常で停止した場合でも、待機側に切り替えたくない場合は、状態遷移プロシジャ内のIS_ISV_WATCH_MODEに「0」を設定してください。通常は「1」が設定されており、切り替えの対象となっています。切り替えたい場合は、修正の必要はありません。

例

ネーミングサービスおよびインタフェースリポジトリが異常終了してもクラスタサービスを切り替えたくない場合

```
IS_ISV_WATCH_MODE="0"
```

Interstage HTTP Serverを運用するために必要なファイルディスクリプタ数の見積もり値が1024を超える場合は、Interstageの起動処理の直前に、[ulimit -n <ファイルディスクリプタ数の見積もり値>]を実行するように修正してください。

起動処理は運用ノード用と待機ノード用の2か所あり、それぞれ、状態遷移プロシジャへのパラメタが以下のケースです。2か所とも修正してください。

起動処理	パラメタ
運用ノード	$1=START、$2=RUN、$3=AFTER、$5=STARTUP、$6=NONE|SERVICE
待機ノード	$1=START、$2=WAIT、$3=AFTER、$5=STARTUP|BUILDIN、$6=NONE|SERVICE

例

Interstage HTTP Serverを運用するために必要なファイルディスクリプタ数の見積もり値が、1050の場合

- 運用ノードの起動処理

  [修正前]

```
# start
ulimit -c unlimited
$RUN_START_COMMAND > /dev/null 2>&1
```

[修正後]

```
# start
ulimit -c unlimited
ulimit -n 1050
$RUN_START_COMMAND > /dev/null 2>&1
```

- 待機ノードの起動処理

[修正前]

```
# wait start
ulimit -c unlimited
$WAIT_STBY_COMMAND > /dev/null 2>&1
```

[修正後]

```
# wait start
ulimit -c unlimited
ulimit -n 1050
$WAIT_STBY_COMMAND > /dev/null 2>&1
```

userApplication名.ES_INTERSTAGEの修正

イベントチャネルに「イベントサービス起動時にイベントチャネルを自動起動しない」という設定を行った場合は、当該状態遷移プロシジャにより、運用ノードで起動する、および切り替え時に動作するイベントチャネルを設定する必要があります。状態遷移プロシジャ内の「ES_CHNL」に、起動するイベントチャネルグループ名を記述してください。

ただし、すべてのイベントチャネルに初期値(省略値)である「イベントサービス起動時にイベントチャネルを自動起動する」という設定を行っている場合、上記の設定を行う必要はありません。

不揮発チャネル運用を行うイベントチャネルの場合、ユニットはイベントチャネル起動時に自動起動されます。状態遷移プロシジャ内の「ES_UNIT」に、起動するユニット名を記述する必要はありません。

例

イベントチャネルグループmix1、mix2、mix3を起動したい場合

```
ES_CHNL="mix1 mix2 mix3"
```

注意

記述する項目は半角ブランクで区切ってください。

userApplication名.IREP_INTERSTAGEの修正

使用する共用ディスクパスを設定します。

例

共用ディスクパスを「/repository」として指定する場合

```
DIRECTORY=/repository
```

注意

共用ディスクパスには、リソース登録済みのパスを指定してください。

userApplication名.OTS_RMP_INTERSTAGEの修正

当該状態遷移プロシジャにより、運用ノードで起動するリソース管理プログラムおよび切り替え時に動作するリソース管理プログラムを設定します。そのため、状態遷移プロシジャ内の「DEF_NAME」にリソース管理プログラムとリソース定義名を以下のように記述してください。

例

/home/ots/resource1をresource1で、/home/ots/resource2をresource2で起動したい場合

```
DEF_NAME="/home/ots/resource1 resource1 /home/ots/resource resource2"
```

Windows32/64 Linux32/64

JTS用のリソース管理プログラムを使用する場合には、「JTS_USE」に「TRUE」を設定してください。

例

JTSをクラスタ環境で使用したい場合

```
JTS_USE="TRUE"
```

注意

記述する項目は半角ブランクで区切ってください。

userApplication名.ODWU_INTERSTAGE/userApplication名.TDWU_INTERSTAGE/userApplication名.IJSERVER_INTERSTAGE/userApplication名.UTYWU_INTERSTAGEの修正

運用ノードで起動するワークユニットおよび切り替え時に引き継ぐワークユニットを設定します。ワークユニットごとの状態遷移プロシジャは以下のとおりです。

- CORBAアプリケーションのワークユニットの場合、userApplication名.ODWU_INTERSTAGE
- トランザクションアプリケーションのワークユニットの場合、userApplication名.TDWU_INTERSTAGE Linux32
- IJServerワークユニットの場合、userApplication名.IJSERVER_INTERSTAGE
- ユーティリティワークユニットの場合、userApplication名.UTYWU_INTERSTAGE

状態遷移プロシジャ内のWU_NAMEにワークユニット名を以下のように記述してください。

例

WU1、WU2を起動および事前起動したい場合

```
WU_NAME="WU1 WU2"
```

注意

記述するワークユニット名は半角ブランクで区切ってください。

4.2.5.1.2 状態遷移プロシジャの登録

1. リソース情報の設定

状態遷移プロシジャのリソース情報を設定する場合、「プロセスの再起動回数」を「0」に、「プロセスの再起動間隔」を「0」に、「プロセスの再起動回数の初期化」を「しない」に設定してください。
設定については、claddprocrscコマンドで行います。claddprocrscコマンドについては、PRIMECLUSTERのマニュアルを参照してください。

2. 状態遷移指示タイミングの設定

Interstageから提供している状態遷移プロシジャに対し、以下に示すタイミングで呼び出されるように設定してください。
設定については、claddprocrscコマンドで行います。claddprocrscコマンドについては、PRIMECLUSTERのマニュアルを参照してください。

```
[START-RUN]
    ・AFTER
[START-WAIT]
    ・AFTER
[STOP-RUN]
    ・BEFORE
[STOP-WAIT]
    ・BEFORE
[FAIL-RUN]
    ・BEFORE
[FAIL-WAIT]
    ・BEFORE
```

3. 状態遷移プロシジャをuserApplicationに登録

状態遷移プロシジャをuserApplicationに登録します。「リソースの起動順序」については、以下の優先順位で呼び出されるように登録してください。

Solaris64

優先度の設定については、「userApplication Configuration wizard」の「Resourceの作成」より行います。「userApplication Configuration wizard」については、PRIMECLUSTERのマニュアルを参照してください。

Linux32/64

優先度の設定については、「CUI(RMS Wizard)」より行います。「CUI(RMS Wizard)」については、PRIMECLUSTERのマニュアルを参照してください。

リソースの起動優先度

- プロシジャクラスのBasicApplicationに登録するリソースの起動優先度(上から順番に呼び出されるように設定)

  userApplication.IREP_INTERSTAGE
  userApplication.IS_INTERSTAGE
  userApplication.SSO_INTERSTAGE (注1)
  userApplication.OTS_RMP_INTERSTAGE
  userApplication.ES_INTERSTAGE

- プロシジャクラスのApplicationに登録するリソースの起動優先度(任意)

  userApplication.TDWU_INTERSTAGE Linux32
  userApplication.IJSERVER_INTERSTAGE
  userApplication.UTYWU_INTERSTAGE
  userApplication.ODWU_INTERSTAGE

注意

注1) セション管理の運用を行う場合にだけ登録してください。

プロシジャリソースのSCRIPTTIMEOUT属性

SCRIPTTIMEOUT属性は、プロシジャリソースの開始、停止のタイムアウト時間(秒)です。

各状態遷移プロシジャに対応するプロシジャリソースに設定するSCRIPTTIMEOUT属性の値は、次のように見積もってください。

状態遷移プロシジャ	SCRIPTTIMEOUTの見積もり方
IS_INTERSTAGE	Interstageの起動時間、およびInterstageの停止時間(全強制停止モード)を実測し、大きい方の値を2倍にしてください 時間の測定は、運用で使用する各サービスの環境設定が完了した状態で行ってください。
Linux32 TDWU_INTERSTAGE	各TDワークユニットの起動時間の実測値の総和と、各TDワークユニットの停止時間の実測の総和の、大きい方を指定してください。(注)
ES_INTERSTAGE	イベントサービスおよびES_INTERSTAGEに記載されている各ユニット、各イベントチャネルの起動時間の実測値の総和、イベントサービスの停止時間を実測値の、大きい方を指定してください。(注)
IREP_INTERSTAGE	各リポジトリの停止時間の実測値の総和と、各リポジトリの停止時間の実測値の総和の、大きい方を2倍にしてください。
SSO_INTERSTAGE	5秒以上
OTS_RMP_INTERSTAGE	各リソース管理プログラムの起動時間の実測の総和と、各リソース管理プログラムの停止時間の実測値の総和の大きい方を2倍にしてください。 Windows32/64 Linux32/64 JTSを使用する場合は、JTS用リソース管理プログラムの起動時間、停止時間を実測して、加算した値で比較してください。
IJSERVER_INTERSTAGE	各IJServerの起動時間の実測の総和と、各IJServerの停止時間の実測の総和の、大きい方を指定してください。(注)
ODWU_INTERSTAGE	各ODワークユニットの起動時間の実測値の総和と、各ODワークユニットの停止時間の実測の総和の、大きい方を指定してください。(注)
UTYWU_INTERSTAGE	各ユーティリティワークユニットの起動時間の実測値の総和と、各ユーティリティワークユニットの停止時間の実測の総和の、大きい方を指定してください。(注)

注)何度か測定し、十分に余裕のある値を指定してください。

操作手順

各手順で、状態遷移プロシジャを実行するすべてのノードにおいて以下に示す操作を行ってください。

1. 状態遷移プロシジャの登録

   Solaris64

   clsetproc -c BasicApplication (注1)
   -m IS_INTERSTAGE (注2)
   -o /etc/opt/FJSVisas/HA/SynfinityCLUSTER/IS_INTERSTAGE (注2)

   Linux32/64

   clsetproc -c BasicApplication (注1)
   -m IS_INTERSTAGE (注2)
   -o /etc/opt/FJSVisas/HA/PRIMECLUSTER/IS_INTERSTAGE (注2)

2. 状態遷移プロシジャを使用するアプリケーションリソースの登録

   claddprocrsc -k IS_INTERSTAGE (注2)
   -m IS_INTERSTAGE (注2)
   -c BasicApplication (注1)
   -K AFTER -w -L AFTER -S BEFORE -T BEFORE -V BEFORE -W BEFORE -u 0 -t 0 -p 100

3. userApplicationに状態遷移プロシジャ用リソースのPRIMECLUSTER用リソースとして登録

   手順2.で作成したリソースをPRIMECLUSTER用のリソースとして登録します。

   Solaris64

   操作は「userApplication Configuration wizard」の「Resourceの作成」より行います。
   リソースの名前は、上記「リソースの起動優先度」の形式にしてください。
   「userApplication Configuration wizard」については、PRIMECLUSTERのマニュアルを参照してください。

   Linux32/64

   操作はCUI(RMS Wizard)より行います。
   CUI(RMS Wizard)については、PRIMECLUSTERのマニュアルを参照してください。

4. userApplicationへの状態遷移用プロシジャ用リソース登録

   手順3.で作成したすべてのリソースを「4.2.4.1 クラスタサービス(userApplication)の起動」で作成したuserApplicationに登録します。
   また「userApplicationの属性」については、「StandbyTransitions」に「ClearFaultRequest|StartUp|SwitchRequest」を設定してください。
   userApplicationへの登録方法およびuserApplicationの属性の設定方法については、PRIMECLUSTERのマニュアルを参照してください。

注意

オプションに指定するパラメタについて

注1)ODWU_INTERSTAGE/TDWU_INTERSTAGE/ IJSERVER_INTERSTAGE/ UTYWU_INTERSTAGEの場合は省略。
注2)登録するリソースの名前に合わせて変更してください。

4.2.5.2 WSFCの場合 Windows32/64

WSFCに対してInterstageのリソースを登録します。
WSFCに対してInterstageのリソースを登録する前に、ワークユニットまたはデータ連携サービスを使用する場合は、リソース管理プログラムおよびワークユニット起動用バッチファイルの作成を行う必要があります。またイベントチャネルを自動起動したい場合にも、WSFCに対してイベントチャネル起動用バッチファイルの作成を行う必要があります。

1. ワークユニット起動用バッチファイルの作成

クラスタの切り替え時にワークユニットを自動起動したい場合には、WSFCに対してワークユニット起動用バッチファイルを汎用アプリケーションとして登録します。
ワークユニット起動用バッチファイルは、以下のように記述します。

```
echo off

isstartwu   ワークユニット名１
isstartwu   ワークユニット名２

pause
```

参考

バッチファイルの最後には必ず「pause」を入れる必要があります。

2. データベース連携サービス起動用バッチファイルの作成

クラスタの切り替え時にデータベース連携サービスを自動起動したい場合には、1)で作成したワークユニット起動用バッチファイルにデータベース連携サービスに関する起動制御文を追記します。この後、データベース連携サービスおよびワークユニット起動用バッチファイルをWSFCに対して汎用アプリケーションとして登録します。
データベース連携サービスおよびワークユニット起動用バッチファイルは、以下のように記述します。

```
echo off

otsstart
otsstartrsc  -pg リソース管理プログラム１ -n リソース定義名１
otsstartrsc  -pg リソース管理プログラム２ -n リソース定義名２
isstartwu   ワークユニット名１
isstartwu   ワークユニット名２

pause
```

参考

JTS用のリソース管理プログラムを起動するには、「otsstartrsc -j」を記載してください。

3. イベントチャネル起動用バッチファイルの作成

クラスタの切り替え時にイベントチャネルを自動起動したい場合には、WSFCに対してイベントチャネル起動用バッチファイルを汎用アプリケーションとして登録します。
イベントチャネル起動用バッチファイルは、以下のように記述します。

```
echo off

esstartunit -unit ユニット名
esstartchnl -g イベントチャネルグループ名 -c イベントチャネル名

pause
```

参考

- バッチファイルの最後には、必ず「pause」を入れる必要があります。
- 揮発チャネル運用時、esstartunitコマンドの記述は必要ありません。

4. リソース登録

WSFCに対してInterstageのリソースを登録します。
クラスタアドミニストレータを使用し、下表に示すリソースの一覧を参考に、使用するサービスのリソースを登録してください。
以下は、使用するデータベースがSymfoware Serverの場合です。
なお、登録対象のリソースの「依存関係(依存先)」に、使用しないリソースが記載されている場合には、その使用しないサービスの依存先が、「依存関係(依存先)」になります。
例えば、「(10)EventServiceサービス」を使用しない場合、「(12)TransactionDirectorサービス」の依存先は「(9)InterfaceRep_Cache Serviceサービス」になります。

注意

Interstage HTTP Serverのサービスのリソースを登録する場合は、リソースの登録後、フェールオーバークラスターマネージャーで登録したリソースの[プロパティ]ー[全般]タブを開いて、「スタートアップパラメータ」を削除してください。

WSFCへのリソース登録について

	リソース資源	サービス名	リソースの種類	依存関係(依存先)	備考
(1)	Interstage用クライアントアクセスポイント	ー	クライアントアクセスポイント		
(2)	共用ディスク	ー	物理ディスク	(1)Interstage用クライアントアクセスポイント	
(3)	Symfowareサービス	Symfoware RDB (注2)	汎用サービス	(2)共用ディスク	
(4)	Interstage ディレクトリサービスのサービス	Interstage Directory Service(リポジトリ名)	汎用サービス	(2)共用ディスク	(注7)
(5)	OD_startサービス	ODloader	汎用サービス	(3)Symfowareサービス	(注1)
(6)	NamingServiceサービス	Naming	汎用サービス	(5)OD_startサービス	(注1)
(7)	InterfaceRep_Cache Serviceサービス	InterfaceRep_Cache_s InterfaceRep_e (注11)	汎用サービス	(6)NamingServiceサービス	(注1) (注10)
(8)	EventServiceサービス	esdmnmain	汎用サービス	(7)InterfaceRep_Cache Serviceサービス	(注1) (注10)

	リソース資源	サービス名	リソースの種類	依存関係(依存先)	備考
(9)	EventFactoryサービス (注5)	esfactory	汎用サービス	(8)EventServiceサービス	(注1)
(10)	TransactionDirectorサービス	TransactionDirector	汎用サービス	(8)EventServiceサービス	(注1)
(11)	Interstage APIサービス	Interstage API	汎用サービス	(10)TransactionDirectorサービス	(注1)
(12)	ObjectTransactionServiceサービス (注3)	ObjectTransactionService	汎用サービス	(11)Interstage APIサービス	(注1)
(13)	データベース連携サービスおよびワークユニット起動用バッチファイル	－	汎用アプリケーション	(12)ObjectTransactionServiceサービスまたは (11)Interstage APIサービス (注4)	(注1)
(14)	イベントチャネル起動用バッチファイル (注6)	－	汎用アプリケーション	(12)ObjectTransactionServiceサービスまたは (11)Interstage APIサービス (注4)	(注1)
(15)	Interstage HTTP Serverのサービス	FJapache Interstage HTTP Server(Webサーバ名)	汎用サービス	(1)Interstage用クライアントアクセスポイント (4)Interstage ディレクトリサービスのサービス (注8)	
(16)	Interstage シングル・サインオンのセション管理サーバ状態監視用アプリケーション (注9)	－	汎用アプリケーション	(15)Interstage HTTP Serverのサービス (注12)	

注1)

「ネットワーク名をコンピュータ名として使う」チェックボックスを選択しないでください。

注2)

Symfoware Server製品のサービス名の詳細については、Symfoware Server製品のマニュアルを確認してください。

注3)

データベース連携サービスを使用する場合のみ、本リソース資源を設定してください。

注4)

データベース連携サービスを使用する場合は、ObjectTransactionServiceサービスを依存関係に設定し、使用しない場合はInterstage APIサービスを依存関係に設定してください。

注5)

EventFactoryサービスは動的生成運用を行うときのみ登録してください。

注6)

イベントチャネルを起動する場合のみ登録してください。

注7)

リポジトリを停止したい場合は、WSFCに登録したリポジトリのリソースを削除してください。

注8)

Interstage シングル・サインオンのリポジトリサーバを使用する場合には、SSOリポジトリとして使用しているInterstage ディレクトリサービスのサービスを依存関係に設定してください。

注9)

セション管理の運用を行うInterstage シングル・サインオンのリポジトリサーバを使用する場合は、Interstage シングル・サインオンのセション管理サーバ状態監視用アプリケーション(C:¥Interstage¥F3FMsso¥ssoatcsv¥bin¥ssostatssmgr.exe)を登録してください。

注10)

「InterfaceRep_Cache Serviceサービス」が存在しない場合、依存先は「NamingServiceサービス」としてください。また、「NamingServiceサービス」が存在しない場合、依存先は「OD_startサービス」としてください。

注11)

InterfaceRep_eサービスが存在する場合には、InterfaceRep_Cache_sサービスに加え、InterfaceRep_eサービスも登録します。InterfaceRep_eサービスが存在しない場合には、InterfaceRep_eサービスを登録する必要はありません。

注12)

Interstage シングル・サインオンのリポジトリサーバを組み込んだInterstage HTTP Serverのサービスを依存関係に設定してください。

参考

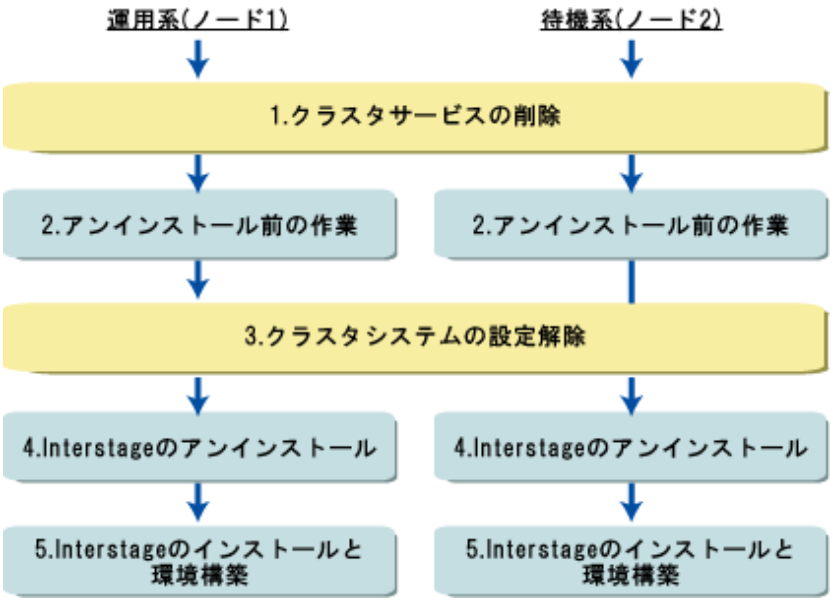
- 上記のすべてのリソースをInterstage用のグループに登録してください。
- リソースの名前には任意の文字列を指定してください。
- 実行できる所有者には、クラスタシステムを構成するノードを指定してください。
- 汎用サービス登録時は、表中のサービス名を指定してください。
- 上記のすべてのリソースについて、[再開する]の"指定期間内での再起動の試行回数"を0にして、リソースの失敗時に現在のノードで再起動を試みない設定にしてください。
- その他の設定項目については、指定の必要はありません。

4.2.6 クラスタシステムの動作確認

クラスタ環境が正しく構築できたかを確認するための動作確認を実施します。
「4.6 クラスタシステムのテスト方法について」で説明されている方法で、クラスタシステムの動作確認を実施してください。

4.3 クラスタサービス連携の解除

クラスタサービス連携を解除し、1台構成のシステム二つに分けるには、一旦、Interstageをアンインストールし、環境を再構築します。
連携解除の作業の流れは、以下の図のとおりです。1と3はクラスタシステムでの操作、2、4、および5はInterstageの操作です。



以下では、それぞれの手順について説明します。

4.3.1 クラスタサービスの削除

Interstageが設定されるクラスタサービスと、クラスタサービスに登録したリソースを削除します。手順について詳細は、ご利用のクラスタシステムのマニュアルを参照してください。

参考

クラスタシステムによって、クラスタサービスの呼び方が異なります。

Solaris64 Linux32/64 PRIMECLUSTER	Windows32/64 フェーオーバー クラスタリング機能(WSFC)
userApplication、または、クラスタアプリケーション	サービスまたはアプリケーション(注)、または、役割

注)フェールオーバークラスタ管理またはフェールオーバー クラスター マネージャーの画面における、”サービスとアプリケーション”直下に作成したものを示します。

4.3.2 アンインストール前の作業

アンインストール前に環境の削除を行います。片方のノードだけで行う作業や、両方のノードで行う作業があります。
一台構成に移行できる資源は、Interstage ディレクトリサービスのエントリデータだけです。環境を削除する前に、必要に応じてLDIF形式でエントリをバックアップしてください。

作業方法についての詳細は、“インストールガイド”-“アンインストール前の作業” Solaris64 Linux32/64、または、“アンインストール”-“アンインストール前の作業” Windows32/64を参照してください。
また、MessageQueueDirectorをご利用の場合は、合わせて、“MessageQueueDirector 説明書”を参照して、MessageQueueDirectorの環境を削除してください。

作業するにあたり、共用ディスクがアクセスできる必要がある場合は、作業するノードから共用ディスクがアクセスできる状態にし、作業が完了したらアクセスできない状態に戻してください。

参考

ノードから共用ディスクをアクセスできる状態にする方法は、クラスタシステムや共用ディスクの構築方法によって異なります。環境にあった方法で共用ディスクがアクセスできるようにしてください。
概ね以下の手順で共用ディスクがアクセスできるようになります。詳しい方法は使用しているクラスタシステムやOSのマニュアルを参照してください。

Solaris64 Linux32/64

作業するノードで、

1) 共用ディスク用のGDSボリュームを、オンライン状態にします。
2) 共用ディスクのマウントポイントとして使用していたディレクトリに、1)でオンラインにしたGDSボリュームをマウントします。

Windows32/64

フェールオーバー クラスター マネージャーなどで、
1) クラスターに記憶域として登録されている共用ディスクの所有ノードを、作業するノードに切り替えます。
2) 共用ディスクをオンラインにします。

4.3.3 クラスタシステムの設定解除

クラスタシステムの以下設定を解除します。手順について詳細は、ご利用のクラスタシステムおよびデータベースシステムのマニュアルでご確認ください。

- 共用ディスク装置の設定
- ネットワーク(引継ぎIPアドレス)の設定
- データベースの環境設定

4.3.4 Interstageのアンインストール

それぞれのノードで、Interstageをアンインストールし、アンインストール後の作業を行います。詳細は、“インストールガイド”- “アンインストール”および“アンインストール後の作業” Solaris64 Linux32/64、または、“アンインストール”- “アンインストール作業”および“アンインストール後の作業” Windows32/64 を参照してください。

4.3.5 Interstageのインストールと環境構築

それぞれのノードで、Interstageをインストールし、環境の再構築を行います。詳しくは、“インストールガイド”、“運用ガイド(基本編)”、および、ご利用の機能の運用ガイドを参照してください。また、必要に応じて、“4.3.2 アンインストール前の作業”でバックアップしたLDIFファイルを使って、リポジトリエントリを作成してください。

4.4 アプリケーション環境作成

クライアントアプリケーションおよびサーバアプリケーションの運用環境の構築方法について説明します。

4.4.1 クライアントアプリケーション環境

クライアントより運用ノードに対して接続処理を行っている時に、運用ノードシステムのダウンなどにより切り替えが発生した場合、クライアントからの接続処理(依頼処理)はエラーまたは無応答状態となります。無応答となった場合の対処として、クライアントアプリケーションのタイマ監視を行ってください。
切り替えが発生した場合、クライアントは運用ノードに対して再接続するような対処を行ってください。
また、クライアントからオブジェクトリファレンスを獲得する場合は、odsethostコマンドを使用してホスト情報を設定しておく必要があります。このとき、以下のIPアドレスを使用して設定を行ってください。

PRIMECLUSTERの場合

引継ぎIPアドレスを使用してください。

WSFCの場合

引継ぎIPアドレスを使用してください。

例

使用するIPアドレスが10.34.157.101、CORBAサービスのポート番号が8002の場合

```
odsethost -a -h 10.34.157.101 -p 8002
```

odsethostコマンドの詳細については、「リファレンスマニュアル(コマンド編)」を参照してください。

4.4.2 サーバアプリケーション環境

サーバアプリケーションの環境は、運用ノードと待機ノードと全く同じ構成で作成しなければなりません。

サーバアプリケーションプログラム

サーバアプリケーションのプログラミングについては、クラスタシステムを使用しない場合と同じです。

スケルトンファイル

スケルトンファイルについては運用ノード、待機ノードでそれぞれ同じものを出力してください。

サーバアプリケーション実行ファイル

サーバアプリケーションなどのワークユニットで使用する資源などについては、運用ノードと待機ノードで同じ構成(ファイル名、ディレクトリ構成など)で作成する必要があります。

APMの生成 Windows32

トランザクションアプリケーションを使用してグローバルトランザクション連携を行う場合は、tdlinkapmコマンドでAPMを生成し運用する必要があります。この場合、運用ノードと待機ノードそれぞれでtdlinkapmコマンドを使用し、同じ構成(ファイル名、ディレクトリ構成など)のAPM名を生成します。生成したAPMは、ワークユニット定義にAPM名を指定します。なお、事前にXA連携用プログラムを作成してからAPMを生成します。

ワークユニット定義

ワークユニット定義については、運用ノードと待機ノードで同じものを使用してください。この場合、運用ノードと待機ノードそれぞれでisaddwudefコマンドを使用し、ワークユニット定義を登録する必要があります。

注意

Windows32 Linux32

トランザクションアプリケーションのワークユニットおよびWRAPPERワークユニットは、tdadddefコマンドでワークユニット定義を登録することもできます。

ワークユニット(CORBAアプリケーション)の変更

クラスタの環境を設定後、ワークユニットを追加または削除する場合、以下の手順で変更を行ってください。

PRIMECLUSTERの場合

追加する場合

1. 運用ノード、待機ノードで追加するCORBAアプリケーションを作成します。
2. クラスタサービスを停止します。
3. 運用ノード、待機ノードで新規に追加するワークユニット定義を登録します。
4. 追加するワークユニット名を状態遷移プロシジャに設定し、再度状態遷移プロシジャをクラスタに登録します。
5. クラスタサービスを起動します。

削除する場合

1. クラスタサービスを停止します。
2. 運用ノード、待機ノードで削除するワークユニット定義を削除します。
3. 削除するワークユニット名を状態遷移プロシジャから削除し、再度状態遷移プロシジャをクラスタに登録します。
4. クラスタサービスを起動します。

WSFCの場合 Windows32/64

追加する場合

1. 運用ノード、待機ノードで追加するCORBAアプリケーションを作成します。
2. クラスタサービスを停止します。
3. 運用ノード、待機ノードで新規に追加するワークユニット定義を登録します。
4. 運用ノード、待機ノードの「ワークユニット起動用バッチファイル」に追加するワークユニットの起動処理を記載します。
5. クラスタサービスを起動します。

削除する場合

1. クラスタサービスを停止します。
2. 運用ノード、待機ノードで削除するワークユニット定義を削除します。
3. 運用ノード、待機ノードの「ワークユニット起動用バッチファイル」に削除するワークユニットの停止処理を記載します。
4. クラスタサービスを起動します。

ワークユニット(トランザクションアプリケーション)の変更 Windows32 Linux32

クラスタの環境を設定後、ワークユニットを追加または削除する場合、以下の手順で変更を行ってください。

PRIMECLUSTERの場合

追加する場合

1. 運用ノード、待機ノードでtdcコマンドを使用し、IDLコンパイルを行います。
2. 運用ノード、待機ノードで出力されたスケルトンを元にアプリケーションを作成します。
3. クラスタサービスを停止します。
4. 運用ノード、待機ノードで新規に追加するワークユニット定義を登録します。
5. 追加するワークユニット名を状態遷移プロシジャに設定し、再度状態遷移プロシジャをクラスタに登録します。
6. クラスタサービスを起動します。

削除する場合

1. 運用ノード、待機ノードでtdcコマンドを使用し、インタフェース情報を削除します。
2. クラスタサービスを停止します。
3. 運用ノード、待機ノードで削除するワークユニット定義を削除します。
4. 削除するワークユニット名を状態遷移プロシジャから削除し、再度状態遷移プロシジャをクラスタに登録します。
5. クラスタサービスを起動します。

WSFCの場合 Windows32

追加する場合

1. 運用ノード、待機ノードでtdcコマンドを使用し、IDLコンパイルを行います。
2. 運用ノード、待機ノードで出力されたスケルトンを元にアプリケーションを作成します。
3. クラスタサービスを停止します。
4. 運用ノード、待機ノードで新規に追加するワークユニット定義を登録します。
5. 運用ノード、待機ノードの「ワークユニット起動用バッチファイル」に追加するワークユニットの起動処理を記載します。
6. クラスタサービスを起動します。

削除する場合

1. 運用ノード、待機ノードでtdcコマンドを使用し、インタフェース情報を削除します。
2. クラスタサービスを停止します。
3. 運用ノード、待機ノードで削除するワークユニット定義を削除します。
4. 運用ノード、待機ノードの「ワークユニット起動用バッチファイル」に削除するワークユニットの停止処理を記載します。
5. クラスタサービスを起動します。

J2EEアプリケーション

J2EEアプリケーションの環境設定は、運用ノードおよび待機ノードそれぞれで、クラスタシステムを使用しない場合と同様な設定を行ってください。
なお、J2EEアプリケーションでJDBCリソースを使用する場合、JDBCの環境設定時に設定するJDBCデータソース等の資源は、各ノードのローカルディスク上に格納してください。

使用可能なデータベース

特に制限はありません。利用方法については各データベースのマニュアルを参照してください。

配備

J2EEアプリケーションの配備は、Interstage管理コンソール、またはijsdeploymentを使用し、運用ノードおよび待機ノードでそれぞれ行ってください。
また、Interstage管理コンソールを使用し、EJBアプリケーションの動作環境定義を行う場合は、運用ノードおよび待機ノードでそれぞれ行ってください。

1. クラスタサービスが起動していない場合は起動します。
2. 運用ノードのIJServerワークユニットをInterstage管理コンソール、またはisstopwuコマンドを使用して停止します。
3. 運用ノードにInterstage管理コンソール、またはijsdeploymentコマンドを使用して配備します。
4. クラスタサービスを待機ノードに切り替えます。
5. 待機ノード(新しい運用ノード)のIJServerワークユニットをInterstage管理コンソール、またはisstopwuコマンドを使用して停止します。
6. 待機ノード(新しい運用ノード)にInterstage管理コンソール、またはijsdeploymentコマンドを使用して配備します。
7. 待機ノード(新しい運用ノード)のIJServerワークユニットをInterstage管理コンソール、またはisstartwuコマンドを使用して起動します。
8. 必要に応じてクラスタサービスを元の運用ノードに切り替えます。クラスタサービスを元の運用ノードに切り替える場合、直前の手順は不要です。

参照

J2EEアプリケーションの環境設定等の詳細については、「J2EE ユーザーズガイド(旧版互換)」を参照してください。

4.5 クラスタサービスの運用

Interstage Application Serverをクラスタサービス上で運用する場合の運用パターンと運用手順について説明します。

4.5.1 運用パターン

クラスタサービス上での運用パターンは、クラスタ製品が提供する機能にしたがいます。
以下に、クラスタ製品ごとの運用パターンについて説明します。

PRIMECLUSTERの場合

クラスタシステムに対する操作	運用ノードの状態	待機ノードの状態
起動	停止 → 起動	停止 → 待機
切り替え	起動 → 待機	待機 → 起動
交換	起動 → 待機	待機 → 起動
組み込み	起動	停止 → 待機
停止	起動 → 停止	待機 → 停止
フェイルオーバ(異常発生時)	起動 → 異常停止	待機 → 起動
フェイルオーバ発生後の復旧	異常停止 → 待機	起動

WSFCの場合

クラスタシステムに対する操作	運用ノードの状態	待機ノードの状態
起動	停止 → 起動	停止
フェイルオーバ(操作時)	起動 → 停止	停止 → 起動
停止	起動 → 停止	停止 → 停止
フェイルオーバ(異常発生時)	起動 → 異常停止	停止 → 起動
フェイルオーバ発生後の復旧	異常停止 → 待機	起動

「切り替え」、「フェイルオーバ(操作時)」は、オペレータによる操作によりクラスタの切り替えを行うケース、「フェイルオーバ(異常発生時)」は、異常によりクラスタが自動的に切り替えられるケースになります。

注意

クラスタ環境では、切り替え/フェイルオーバ時にInterstage管理コンソールの制御の引継ぎは行われません。必ず運用ノード切り替え後に、Interstage管理コンソールの再ログオンが必要です。

4.5.2 運用方法

クラスタサービス上での運用パターンごとに、運用方法や運用上の留意点について説明します。

起動

クラスタ製品を使用してクラスタサービスの起動操作を行うと、運用ノードでInterstageが起動されます。
各ノードにおけるInterstageの動作状態は、以下のとおりとなります。

	PRIMECLUSTER	WSFC
運用ノード	起動	起動
待機ノード	待機	停止

切り替え/フェイルオーバ(操作)

運用ノードを停止し、待機ノードを新運用ノードへ切り替える操作です。
クラスタ製品を使用してクラスタサービスの切り替え操作を行うと、運用ノードで動作しているInterstageは停止もしくは待機状態となり、待機ノードでInterstageが起動します。
各ノードにおけるInterstageの動作状態は、以下のとおりとなります。

	PRIMECLUSTER	WSFC
運用ノード	待機	停止
待機ノード(新運用ノード)	起動	起動

交換

運用ノードと待機ノードを切り替える操作です。
クラスタ製品を使用して交換操作を行うと、運用ノードで動作しているInterstageは待機ノードでの動作状態に遷移し、待機ノードでInterstageが起動します。

	PRIMECLUSTER	WSFC
運用ノード(新待機ノード)	待機	－
待機ノード(新運用ノード)	起動	－

－:WSFCには該当の操作がありません。

組み込み

停止状態となっている待機ノードを起動する操作です。
クラスタ製品を使用して組み込み操作を行うと、運用ノードで動作しているInterstageの動作状態は変わらず、待機ノードのInterstageが待機状態になります。

	PRIMECLUSTER	WSFC
運用ノード	起動	－
待機ノード	待機	－

－:WSFCには該当の操作がありません。

なお、異常が発生したサーバの組み込みに関しては、「フェイルオーバ(異常発生)」もあわせて参照してください。

停止

クラスタ製品を使用してクラスタサービスの停止操作を行うと、運用ノードのInterstageが停止されます。また、待機ノードでInterstageが待機(事前起動)状態となっている場合には、停止します。
各ノードにおけるInterstageの動作状態は、以下のとおりとなります。

	PRIMECLUSTER	WSFC
運用ノード	停止	停止
待機ノード	停止	停止

フェイルオーバ(異常発生)

クラスタサービスの運用中にクラスタ監視対象のリソースで異常が発生した場合、クラスタ製品が自動的にフェイルオーバ(ノードの切り替え)を行います。
本現象発生時には、クラスタ製品により、運用ノードで起動されているInterstageが停止し、待機ノードで待機状態となっているInterstageが起動されます。
なお、異常が発生したサーバの復旧方法については、「フェイルオーバ発生後の復旧」を参照してください。

	PRIMECLUSTER	WSFC
運用ノード(新運用ノード)	停止	停止
待機ノード(新運用ノード)	起動	起動

フェイルオーバ発生後の復旧

異常の発生により、クラスタサービスのフェイルオーバが発生した場合には、異常原因を調査し、対処を実施します。
対処の完了後は、ノードを再起動してください。また、PRIMECLUSTERに関しては、クラスタの組み込み作業を行い、待機状態に遷移させます。
詳細については、「4.5.3 異常発生時の対処方法とトラブルシューティング」を参照してください。

	PRIMECLUSTER	WSFC
運用ノード(新運用ノード)	待機	停止
待機ノード(新運用ノード)	起動	起動

4.5.3 異常発生時の対処方法とトラブルシューティング

クラスタサービス上で異常(切り替え)が発生した場合の対処方法について説明します。
異常発生時には、本手順にしたがって復旧作業を実施してください。

1. トラブルシューティング
2. 切り替え後の待機ノードの組み込み

1. トラブルシューティング

クラスタ切り替えが発生した場合には、以下の手順でトラブル原因を調査してください。

1. クラスタシステムが検出した異常を確認します。Interstageの状態遷移プロシジャ、Cmdlineリソースの異常を検知してフェイルオーバが行われている場合は次へ進みます。それ以外の異常の場合は、その状態遷移プロシジャ、または、Cmdlineリソースの内容を調査してください。
2. Interstage の状態遷移プロシジャ、または、Cmdlineリソースを修正したか否かを確認します。修正している場合には、修正した部分で発生しているのか、それとも Interstageの既存制御で発生しているのかを調査します。Interstageの既存制御で発生している場合には次へ進みます。状態遷移プロシ ジャ、Cmdlineリソースの修正部分で発生している場合には、その修正箇所に対して調査を行ってください。
3. Interstageの出力メッセージを確認し、原因を調査してください。

2. 切り替え後の待機ノードの組み込み

サーバ異常による切り替え時、サーバ上のプロセスやIPC資源(Solaris、Linuxの場合)などが残っている場合があります。そのため、サーバマシンの停止/再起動により資源を回収後、待機ノードの組み込みを行ってください。

注意

サーバマシンの再起動を行わずにノードの組み込みを行った場合、Interstageの起動に失敗する場合があります。ノードの組み込み前に、必ずサーバマシンの再起動を行ってください。

4.6 クラスタシステムのテスト方法について

Interstageにおいて異常が発生した場合に、クラスタシステムの切り替えが発生することを確認するためのテスト方法を以下に示します。

Interstageのテスト方法

確認方法 Windows32/64

1. 運用ノードにおいて、Windowsのタスクマネージャを使用し、manage.exe プロセスを特定します。
2. 手順1.で特定したプロセスを、タスクマネージャから終了します。

確認方法 Solaris64 Linux32/64

1. 運用ノードにおいて"ps -ef"コマンドを実行し、manageプロセスのプロセスIDを特定します。以下の実行例のように"ps -ef"コマンドを実行し、manageプロセスのプロセスIDを特定してください。

   例

   プロセス特定方法

   ps -efコマンド実行結果よりプロセスIDを特定します。以下の場合、manageプロセスのプロセスIDは2795になります。

   Solaris64

   ```
   # ps -ef | grep /opt/FSUNtd/lib/manage
     root   2795     1  0   7月 18 ?        0:00 /opt/FSUNtd/lib/manage default
   ```

   Linux32/64

   ```
   # ps -ef | grep /opt/FJSVtd/lib/manage

   root      2795     1  0 Jul10 ?        00:00:00 /opt/FJSVtd/lib/manage default
   ```

2. 手順1.で特定したプロセスに対して"kill -9"コマンドを実行し、シグナルを送信します。

   例

   以下に、プロセスIDが2795のプロセスに対する実行例を示します。

   ```
   # kill -9 2795
   ```

   なお、kill コマンドで送るシグナルは、必ず"-9"を指定してください。

以上の手順の実施によりクラスタシステムの切り替えが発生します。
クラスタシステムとして切り替えが成功したか、また、運用ノード、待機ノードにおけるInterstageの動作状態が妥当であるかを、コマンドなどを使用し、確認してください。

4.7 Interstageの起動・停止

クラスタ環境において、isstart, isstopコマンドなどの通常のの方法で起動・停止を行うと、起動に失敗したり、ノード切り替えが発生します。
クラスタ環境でのInterstageの起動・停止は、クラスタサービスを起動・停止することで行ってください。

4.8 バックアップ・リストア

クラスタ環境でのInterstage資源のバックアップおよびリストア方法については、「運用ガイド(基本編)」の「メンテナンス(資源のバックアップ／他サーバへの資源移行／ホスト情報の変更)」－「資源のバックアップとリストア(クラスタ環境)」を参照してください。

4.9 留意事項

4.9.1 ノーティフィケーションサービスの不揮発チャネル運用時の注意事項

ノーティフィケーションサービスの不揮発チャネル運用では、「4.2.4.4.4 ノーティフィケーションサービスの不揮発チャネル運用を行う場合」における設定後に、ユニットおよびイベントチャネルを削除する場合は、以下の手順で削除する必要があります。

1. クラスタサービスの切り替え

クラスタサービスを切り替えます。ノード1を運用ノードに、ノード2を待機ノードにします。

2. 状態遷移プロシジャをクラスタサービスからリソース削除

PRIMECLUSTERの場合

クラスタサービスにリソース登録した状態遷移プロシジャを、「userApplication Configration wizard」により、有効になっているuserApplicationから削除します。
次に、クラスタを構成するアプリケーションリソースより、cldelprocrscコマンドを使用して状態遷移プロシジャ削除します。
「userApplication Configuration wizard」およびcldelprocrscコマンドについては、PRIMECLUSTERのマニュアルを参照してください。

3. Interstageの停止

運用ノード(ノード1)でisstopコマンドによりInterstageを停止します。このとき、-fオプションを指定する必要があります。

例

```
isstop -f
```

4. クラスタサービスの切り替え

クラスタサービスを切り替えます。ノード2を運用ノードに、ノード1を待機ノードに切り替えます。

5. Interstageの起動

運用ノード(ノード2)でisstartコマンドによりInterstageを起動します。

例

```
isstart
```

6. ユニットの起動

運用ノード(ノード2)でesstartunitコマンドにより拡張ユニットを起動します。

例

```
esstartunit -unit ユニット名
```

7. イベントチャネルの削除

運用ノード(ノード2)でesrmchnlコマンドにより、すべてのイベントチャネルを削除します。

例

```
esrmchnl -g グループ名
```

8. イベントチャネルのオブジェクトリファレンスの確認

運用ノード(ノード2)でodlistnsコマンドにより、ネーミングサービスに登録されているイベントチャネルのオブジェクトリファレンスを確認します。odlistnsコマンドに-lオプションを指定して実行し、「インタフェースリポジトリ名」が「IDL:CosNotifyChannelAdmin/EventChannel:1.0」または「IDL:CosEventChannelAdmin/EventChannel:1.0」である「ネーミングサービスに登録した名前」を確認します。なお、この「ネーミングサービスに登録した名前」は、「9. イベントチャネルのオブジェクトリファレンスの削除」において「グループ名::イベントチャネル名」として指定します。

例

```
odlistns -l

[表示内容]
Name(Type)  Object information(detail)
              Default object information(detail)
「ネーミングサービスに登録した名前」(登録した型) 「インタフェースリポジトリ名」,...
```

9. イベントチャネルのオブジェクトリファレンスの削除

運用ノード(ノード2)でOD_or_admコマンドにより、イベントチャネルのオブジェクトリファレンスを削除します。「グループ名::イベントチャネル名」には、「8. イベントチャネルのオブジェクトリファレンスの確認」で確認した「ネーミングサービスに登録した名前」を指定します。

例

```
OD_or_adm -d -n グループ名::イベントチャネル名
```

10. ユニットの停止(ユニットを削除する場合のみ)

運用ノード(ノード2)でesstopunitコマンドにより拡張ユニットを停止します。

例

```
esstopunit -unit ユニット名
```

11. 不揮発用ファイルを共用ディスクから削除(ユニットを削除する場合のみ)

運用ノード(ノード2)でesrmunitコマンドによりユニットを削除します。

例

```
esrmunit
```

12. Interstageの停止

運用ノード(ノード2)でisstopコマンドによりInterstageを停止します。このとき、-fオプションを指定する必要があります。

例

```
isstop -f
```

13. クラスタサービスの切り替え

クラスタサービスを切り替えます。ノード1を運用ノードに、ノード2を待機ノードに切り替えます。

14. Interstageの起動

運用ノード(ノード1)でisstartコマンドによりInterstageを起動します。

例

```
isstart
```

15. ユニットの起動

運用ノード(ノード1)でesstartunitコマンドにより拡張ユニットを起動します。

例

```
esstartunit -unit ユニット名
```

16. イベントチャネルの削除

運用ノード(ノード1)でesrmchnlコマンドにより、すべてのイベントチャネルを削除します。

例

```
esrmchnl -g グループ名
```

17. ユニットの停止(ユニットを削除する場合のみ)

運用ノード(ノード1)でesstopunitコマンドにより拡張ユニットを停止します。

例

```
esstopunit -unit ユニット名
```

18. 不揮発用ファイルを共用ディスクから削除(ユニットを削除する場合のみ)

運用ノード(ノード1)でesrmunitコマンドによりユニットを削除します。

例

```
esrmunit
```

19. Interstageの停止

運用ノード(ノード1)でisstopコマンドによりInterstageを停止します。このとき、-fオプションを指定する必要があります。

例

```
isstop -f
```

4.9.2 イベントサービスおよびノーティフィケーションサービスの揮発チャネル運用時の注意事項

イベントサービスおよびノーティフィケーションサービスの揮発チャネル運用では、「4.2.4.4.5 イベントサービスおよびノーティフィケーションサービスの揮発チャネル運用を行う場合」における設定後に、イベントチャネルを削除する場合は、以下の手順で削除する必要があります。

1. クラスタサービスの切り替え

クラスタサービスを切り替えます。ノード1を運用ノードに、ノード2を待機ノードにします。

2. 状態遷移プロシジャをクラスタサービスからリソース削除

PRIMECLUSTERの場合

クラスタサービスにリソース登録した状態遷移プロシジャを、「userApplication Configration wizard」により、有効になっているuserApplicationから削除します。

次に、クラスタを構成するアプリケーションリソースより、cldelprocrscコマンドを使用して状態遷移プロシジャ削除します。
「userApplication Configuration wizard」およびcldelprocrscコマンドについては、PRIMECLUSTERのマニュアルを参照してください。

3. Interstageの停止

運用ノード(ノード1)でisstopコマンドによりInterstageを停止します。このとき、-fオプションを指定する必要があります。

 例

```
isstop -f
```

4. クラスタサービスの切り替え

クラスタサービスを切り替えます。ノード2を運用ノードに、ノード1を待機ノードに切り替えます。

5. Interstageの起動

運用ノード(ノード2)でisstartコマンドによりInterstageを起動します。

 例

```
isstart
```

6. イベントチャネルの削除

運用ノード(ノード2)でesrmchnlコマンドにより、すべてのイベントチャネルを削除します。

 例

```
esrmchnl -g グループ名
```

7. イベントチャネルのオブジェクトリファレンスの確認

運用ノード(ノード2)でodlistnsコマンドにより、ネーミングサービスに登録されているイベントチャネルのオブジェクトリファレンスを確認します。
odlistnsコマンドに-lオプションを指定して実行し、「インタフェースリポジトリ名」が「IDL:CosNotifyChannelAdmin/EventChannel:1.0」または「IDL:CosEventChannelAdmin/EventChannel:1.0」である「ネーミングサービスに登録した名前」を確認します。なお、この「ネーミングサービスに登録した名前」は、「8. イベントチャネルのオブジェクトリファレンスの削除」において「グループ名::イベントチャネル名」として指定します。

 例

```
odlistns -l

[表示内容]
Name(Type)  Object information(detail)
                Default object information(detail)
「ネーミングサービスに登録した名前」(登録した型) 「インタフェースリポジトリ名」,...
```

8. イベントチャネルのオブジェクトリファレンスの削除

運用ノード(ノード2)でOD_or_admコマンドにより、イベントチャネルのオブジェクトリファレンスを削除します。「グループ名::イベントチャネル名」には、「7. イベントチャネルのオブジェクトリファレンスの確認」で確認した「ネーミングサービスに登録した名前」を指定します。

 例

```
OD_or_adm -d -n グループ名::イベントチャネル名
```

9. Interstageの停止

運用ノード(ノード2)でisstopコマンドによりInterstageを停止します。このとき、-fオプションを指定する必要があります。

 例

```
isstop -f
```

10. クラスタサービスの切り替え

クラスタサービスを切り替えます。ノード1を運用ノードに、ノード2を待機ノードに切り替えます。

11. Interstageの起動

運用ノード(ノード1)でisstartコマンドによりInterstageを起動します。

 例

```
isstart
```

12. イベントチャネルの削除

運用ノード(ノード1)でesrmchnlコマンドにより、すべてのイベントチャネルを削除します。

 例

```
esrmchnl -g グループ名
```

13. Interstageの停止

運用ノード(ノード1)でisstopコマンドによりInterstageを停止します。このとき、-fオプションを指定する必要があります。

 例

```
isstop -f
```

付録A　クラスタサービスの互換機能

A.1　クラスタサービス機能の互換機能

本製品では、旧バージョン製品で説明してきたクラスタサービス機能の設計方法のうち、いくつかの設計方法について推奨しておりません。
新たにシステムを構築される場合には、「2.2 クラスタサービス機能を利用した高信頼化システム」で説明されている方法で設計を行ってください。

推奨していない機能

- 既存システム(グローバルサーバ)との連携

なお、上記機能は、旧バージョン製品でクラスタサービス機能を使用しており、本製品においても同様に使用したいというようなケースに備えての互換機能としてサポートします。
旧バージョンにおいて、上記に該当する設計/運用を行っており、本製品においても同様の設計で運用する必要がある場合(製品バージョンアップにともなう環境移行など)には、従来どおり、上記機能を使用しても問題はありません。
以降、互換機能の運用方法について説明します。

A.2　既存システム(グローバルサーバ)との連携 Windows32/64 Solaris64

既存システム(グローバルサーバ)との連携は、Windows(R)、Solarisでだけ使用可能です。
既存システムとの連携を行う場合、関連製品を使用します。関連製品の環境作成については、それぞれのマニュアルを参照してください。
また、関連製品のリソースについては、Interstageと同一のサービスに設定する必要があります。起動の優先度はクラスタサービス機能を使用しない場合と同じです。

Windows32/64
IDCMは、「自動起動形態1」でインストールしてください。

A.2.1　WSFCの場合 Windows32/64

既存システムとの連携を行う場合、IDCMの起動/停止は、WSFCへIDCMのサービスをリソースとして登録することにより行います。リソースは、Interstageのリソースグループへ追加します。
以下に、設定手順を説明します。

1. Interstageのリソースグループの設定を行います。「4.2.5.2 WSFCの場合 」で説明されている手順を実施します。
2. 1.で作成したInterstageのリソースグループに対して、以下の条件にしたがってIDCMサービスを登録します。
   - 「サービス名」は「F3CTidcm」です。
   - 「リソースの種類」は「汎用サービス」です。
   - 依存先は「InterfaceRep_Cache Serviceサービス」および「ネットワーク名」です。
3. TransactionDirectorサービスの依存先として、「IDCMサービス」を指定します。

以下に、既存システムとの連携を行う場合のWSFCへのリソース登録例を示します。

WSFCへのリソース登録について

リソース資源	サービス名	リソースの種類	依存関係(依存先)
Interstage用IPアドレス	－	IPアドレス	
Interstage用ネットワーク名	－	ネットワーク名	IPアドレス
OD_startサービス	ODloader	汎用サービス	ネットワーク名
NamingServiceサービス	Naming	汎用サービス	OD_startサービス
InterfaceRep_Cache Serviceサービス	InterfaceRep_Cache_s InterfaceRep_e	汎用サービス	NamingServiceサービス
IDCMサービス	F3CTidcm	汎用サービス	InterfaceRep_Cache Serviceサービスおよびネットワーク名
TransactionDirectorサービス	TransactionDirector	汎用サービス	IDCMサービス
Windows64 Interstage APIサービス	Interstage API	汎用サービス	TransactionDirectorサービス
ワークユニット起動用バッチファイル	－	汎用アプリケーション	Windows64 Interstage APIサービス

また、DPCF通信パスの確立/解放は、以下のいずれかの方法で行ってください。

- オペレータによる操作
  [IDCM操作]ウィンドウの[確立]ボタン/[解放]ボタンを選択して行ってください。
- アプリケーションからの操作
  「IDCMソフトウェア開発キット」をインストール後、C言語で作成したアプリケーションでIDCMのAPI(idcm_psysACT/idcm_psysDCT)を使用して行ってください。
- 相手システムからの操作
  相手システムから確立/解放を行ってください。

IDCMの環境設定は、以下のいずれかの方法で行ってください。

- 運用ノード、待機ノードごとの物理IP(ホスト名)を使用する方法
  この場合、相手システムからは別ホスト(DPCF通信パスは2本)となります。また、この方法の場合、IDCMサービスのプロパティのパラメータタブで「ネットワーク名をコンピュータ名として使用する」をチェックしないでください。
- 仮想IP(ホスト名)を使用する方法
  この場合、相手システムからは同一ホスト(DPCF通信パスは1本)となります。使用できる通信プロトコルはTCP/IP通信のみです。また、この方法の場合、IDCMサービスのプロパティのパラメータタブで「ネットワーク名をコンピュータ名として使用する」をチェックしておく必要があります。

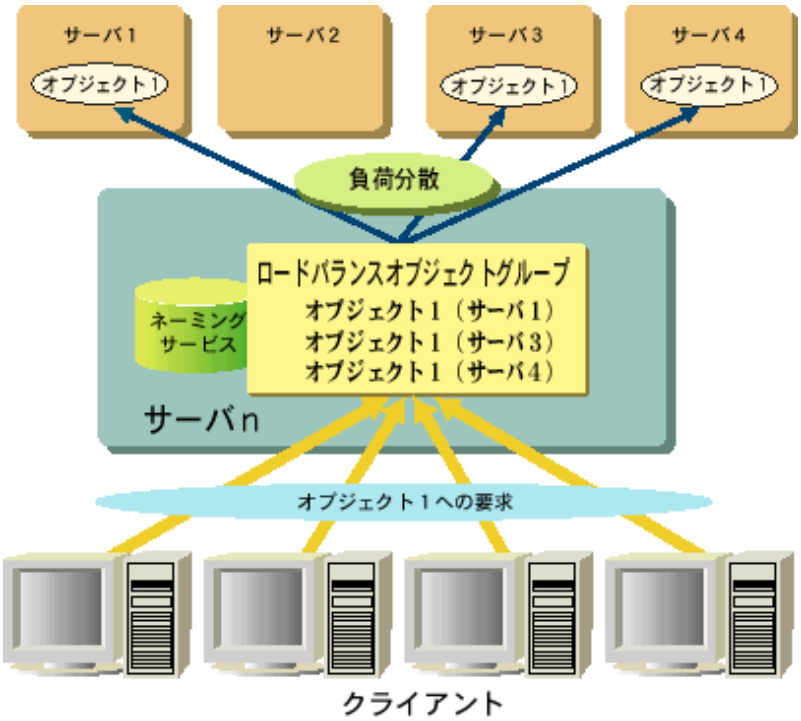
IDCMの環境設定については、「IDCMヘルプ」を参照してください。

付録B　ロードバランス機能を利用した場合の設計 Windows32 Linux32

ロードバランス機能を利用した複数サーバの運用、高信頼化システムの設計について説明します。

B.1　ロードバランス機能

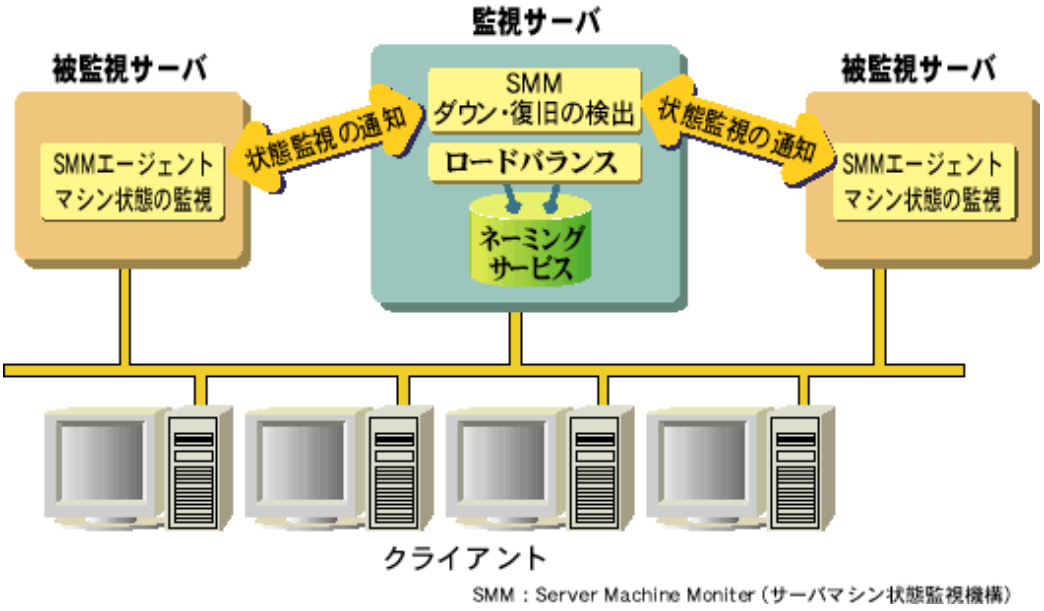
ロードバランス機能(NSLBO:Naming Service-Load Balancing Option)は、ネーミングサーバ機能が拡張され、均一な運用環境での負荷分散を実現しています。
同じサーバオブジェクト(インタフェースが同一で、同じ機能を提供するオブジェクト)を複数のサーバ上に配置します。このオブジェクトに対してクライアントから要求があった場合には、これらのオブジェクトへの負荷が均等になるように調整して、最適なサーバ上のリファレンスを返答することで複数サーバへの要求の分散が実現できます。



ロードバランスでは、同一サーバオブジェクト(オブジェクト1)を複数のサーバ上で運用し、このオブジェクトに対するクライアントからの要求に対して順番に割り当てることにより負荷分散を実現しています。なお、ロードバランスの対象となるサーバオブジェクトは、ネーミングサービスと同時にロードバランスオブジェクトグループで管理します。ロードバランスオブジェクトグループでは、ロードバランスの対象となる同一サーバオブジェクトをまとめて管理しています。

B.2　サーバマシン状態監視機構

サーバマシン状態監視機構では、ロードバランス運用時に指定したサーバマシンの状態をマシン単位で監視し、その状態をロードバランスに通知できます。
このため、監視対象のサーバがダウンしたとき、サーバマシン状態監視機構からダウンしたサーバが通知されることで、ロードバランスにより自動的に、起動中のサーバだけで運用を続けます。ダウンしたサーバが復旧すれば、また自動的に元の状態で運用を続けることができます。
サーバマシン状態監視機構で監視するサーバを監視サーバ、監視されるサーバを被監視サーバと呼びます。



サーバマシン状態監視機構では、被監視サーバ上に置かれたSMMエージェント(Server Machine Monitor Agent : SMMA)と通信してサーバマシンの状態を監視します。このSMMエージェントからの通信を、稼働通知と呼びます。
サーバマシン状態監視機構が被監視サーバのダウン、または被監視サーバの復旧を認識すると、以下の方法でロードバランスに通知できます。なお、以下の方法は、サーバマシン状態監視機構およびSMMエージェントの起動時に指定できます。

- サーバマシン状態監視機構から直接ロードバランスに通知
- サーバマシン状態監視機構からシェルプログラムを実行してロードバランスに通知
- SMMエージェントからシェルプログラムを実行してロードバランスに通知

サーバマシン状態監視機構、ロードバランス、ネーミングサービスを1つのサーバにまとめて配置・運用することで、ダウン時のリカバリ処理を容易にすることができます。
また、この監視サーバの待機サーバを用意して、マシン異常時に運用を切り替えることで、信頼性を向上させることができます。

注意

サーバマシン状態監視機構は、IPv6アドレスを指定した運用を行うことができません。IPv4アドレスを指定した運用を行ってください。

B.3 複数サーバの運用設計

Interstageを複数サーバで運用する場合に使用できるロードバランスについて説明します。

ロードバランス機能(NSLBO:Naming Service-Load Balancing Option)とは、ネーミングサービスと連携して提供される、均一な運用環境での負荷分散を実現させる機能です。
同じサーバオブジェクト(インタフェースが同一で、同じ機能を提供するオブジェクト)を複数のサーバ上に配置し、このオブジェクトに対してクライアントから要求があった場合には、これらのオブジェクトへの負荷が均等になるように調整してオブジェクトリファレンスを返します。

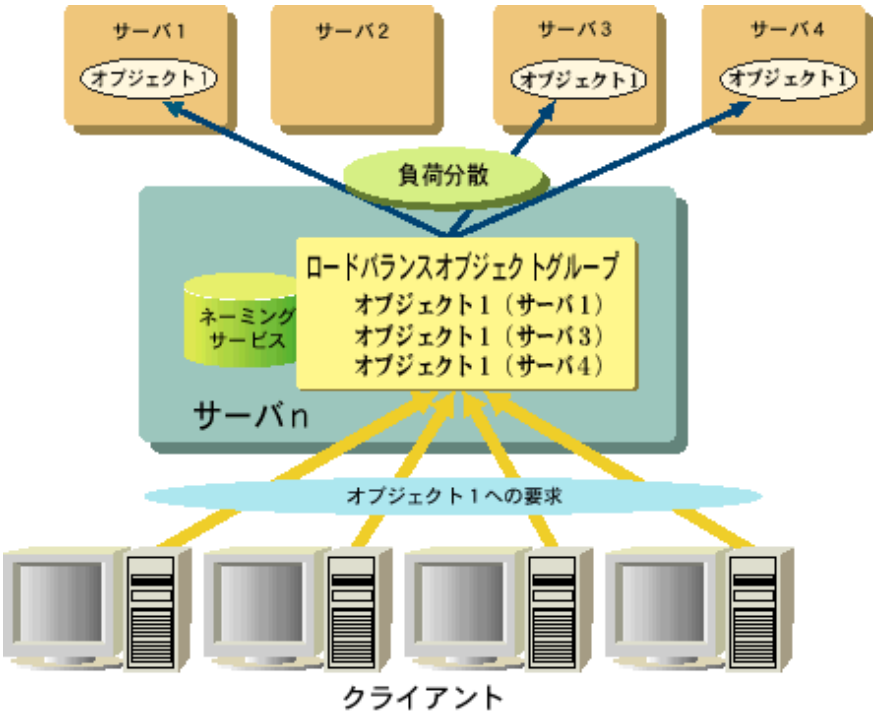
注意

ロードバランス機能は、「Interstage Application Server Enterprise Edition」でのみ使用できます。

参考

均一な運用環境とは、各クライアントが、任意のサーバオブジェクトのオブジェクトリファレンスをネーミングサービスから取得した後、そのサーバオブジェクトへの要求頻度がクライアント間である程度均等である場合をさしています。
この要求頻度が、特定のクライアントに偏っている場合には、ロードバランスでの充分な負荷分散は提供できない場合があります。

以下に、ロードバランスの仕組みについて図を示します。以下の例では、複数のサーバ上に配置したオブジェクト1をロードバランス対象オブジェクトとして運用しています。



ロードバランスでは、同一サーバオブジェクト(オブジェクト1)を複数のサーバ上で運用し、このオブジェクトに対するクライアントからの要求に対して順番に割り当てることにより負荷分散を実現しています。
なお、ロードバランスの対象となるサーバオブジェクトは、ネーミングサービスと同時にロードバランスオブジェクトグループで管理されます。ロードバランスオブジェクトグループでは、ロードバランスの対象となる同一サーバオブジェクトをまとめて管理しています。

ここでは、以下の内容について説明します。

- ロードバランスの運用手順
- サーバダウン時の縮退運転および復旧通知
- サーバマシン状態監視機構

Windows32

なお、ロードバランスが運用中に停止した場合には、サーバオブジェクトのオブジェクトリファレンスが取得できなくなるため、システム全体が停止してしまいます。ロードバランスのデータについては、信頼性をあげるため、万一の場合に備えて二重化することを推奨しています。
詳細については、「1.2 クラスタサービス機能」を参照してください。

B.3.1 ロードバランスの運用手順

ロードバランスを利用する場合の運用手順について説明します。なお、各コマンドの詳細については、「リファレンスマニュアル(コマンド編)」を参照してください。

運用手順には、ロードバランス対象のオブジェクトの登録方法によって、以下の2つがあります。

- 固定登録する場合
- サーバオブジェクトの起動時に毎回登録する場合

それぞれの運用手順について説明します。なお、ロードバランスは、ネーミングサービスを使用してオブジェクト名を管理しています。ロードバランスを利用する場合には、ネーミングサービスが起動されていることを確認してください。

注意

以降の運用手順を行う前に、サーバアドレスの設定をする必要があります。その際、以下の点に注意してください。

オブジェクトリファレンスでは、アドレス情報として以下の3つの形態を利用できます。

- ホスト名
- DNS(Domain Name Service)名
- IPアドレス(ドット形式)

ロードバランスでは、ロードバランス対象の各オブジェクトがどのサーバ上で動作しているかを、IP アドレスで管理します。このため、ロードバランスが動作するサーバで、各オブジェクトのアドレス情報をIPアドレスに変換する必要があります。したがって、アドレス情報としてホスト名または DNS 名を使用する場合は、以下の対処を行います。

ホスト名の場合

ロードバランスが動作するサーバに、サーバオブジェクトが動作するサーバと同一のホスト名およびIPアドレスの対応を定義します。

Windows32

¥%SystemRoot%¥system32¥drivers¥etc¥HOSTSファイルに、ホスト名とIPアドレスの対応を記述します。
注) %SystemRoot%は、Windows(R)システムのインストールフォルダです。

Linux32

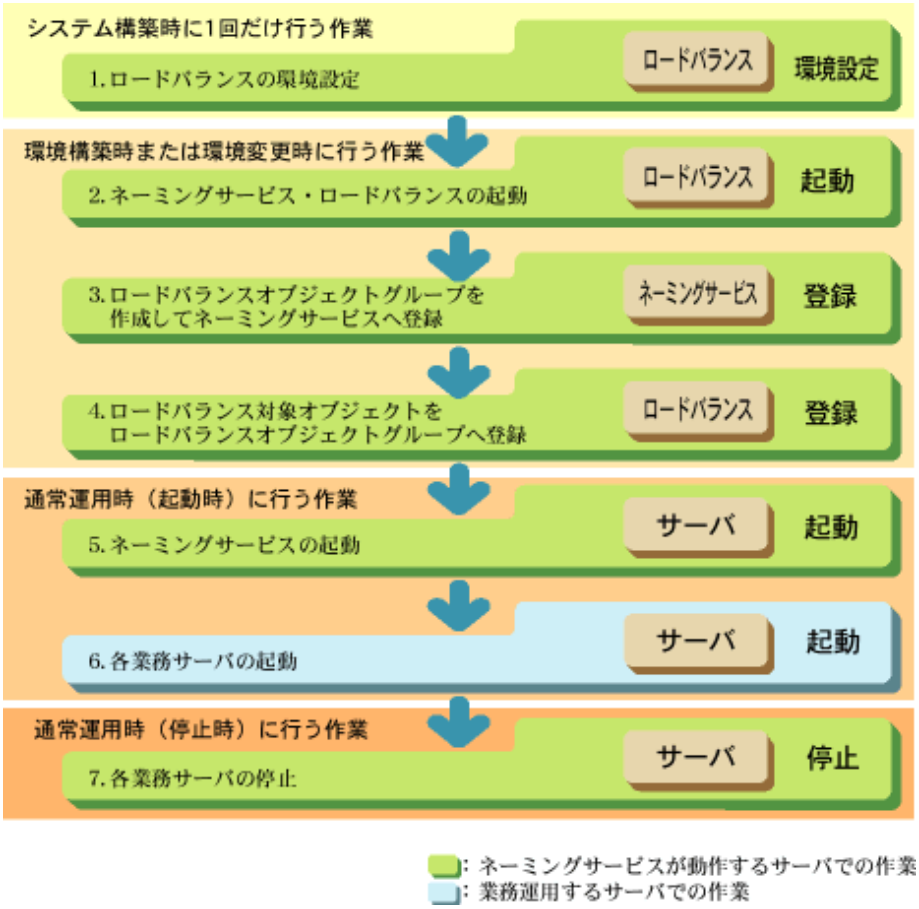
/etc/hostsファイルに、ホスト名とIPアドレスの対応を記述します。

DNSの場合

ロードバランスが動作するサーバマシンにDNSを設定します。
なお、トランザクションアプリケーションでロードバランスを使用する場合は、アドレス情報としてホスト名を使用してください。この場合のホスト名には、大文字と小文字が区別されるため、各サーバで同一の文字列を使用するように注意してください。

B.3.1.1 固定登録する場合

ロードバランス運用の対象となるオブジェクトが固定の場合には、システム環境構築時に1回だけ、ロードバランスを利用して登録を行います。
以下に、運用手順とその詳細について説明します。



1. ロードバランスの環境設定

odsetlboコマンドまたはisinitコマンドを使用して、ロードバランスの環境を設定します。

注意

odsetlboコマンドまたはisinitコマンドは、ロードバランスが動作するマシンまたはロードバランスにアクセスするマシンの全マシン上で1回行う必要があります。コンポーネントトランザクションサービスにおいてロードバランス運用を行う場合、業務運用するサーバにはネーミングサービスを配置しないでください。すなわち、業務運用するサーバ(ロードバランスにアクセスするマシン)では、isinitコマンドのオペランドとして、必ずTYPE3を指定してください。

2. ネーミングサービス・ロードバランスの起動

Windows32

Windows(R)の管理ツールの「サービス」を起動し、以下のネーミングサービスおよびロードバランスのサービスを開始します。

- ネーミングサービス:「Naming Service」サービス
- ロードバランス:「NS LoadBalancingOption」サービス

Linux32

CosNaming_sコマンドおよびodstartlboコマンドを使用して、ネーミングサービスおよびロードバランスを起動します。

3. ロードバランスオブジェクトグループを作成してネーミングサービスへ登録

OD_or_admコマンドを使用して、ロードバランスオブジェクトグループを作成します。なお、このときに-g lbオプションを指定してください。-g lbオプションを指定すると、ロードバランスオブジェクトグループの作成と同時にネーミングサービスへ登録します。
ロードバランス機能インタフェースのアプリケーションを使用して、ロードバランスオブジェクトグループの作成、およびネーミングサービスへの登録を行うことができます。ロードバランス機能インタフェースのアプリケーション作成については、「アプリケーション作成ガイド(CORBAサービス編)」の「ロードバランス機能のプログラミング」を参照してください。
なお、トランザクションアプリケーションを使用する場合は、ロードバランスオブジェクトグループの名前として、ネーミングサービスへ登録するロードバランスオブジェクトグループのモジュール名とインタフェース名を「::」(2つのコロン)で区切った名前を使用してください。

4. ロードバランス対象オブジェクトをロードバランスオブジェクトグループへ登録

odadministerlbコマンドを利用して、各サーバで動作するロードバランス対象オブジェクトを、ロードバランスオブジェクトグループへ登録します。任意のサーバから一括して登録することも、各サーバから登録することもできます。
なお、トランザクションアプリケーションでロードバランスを使用する場合、odadministerlbコマンドの-hオプションには、アドレス情報としてホスト名を指定してください。
ロードバランス機能インタフェースのアプリケーションを使用して、ロードバランス対象オブジェクトを登録することもできます。ロードバランス機能インタフェースのアプリケーション作成については、「アプリケーション作成ガイド(CORBAサービス編)」の「ロードバランス機能のプログラミング」を参照してください。

注意

1つのロードバランスオブジェクトグループには、ロードバランス対象のオブジェクトとして、必ず同一のインタフェースで同一の機能を提供するオブジェクトを登録してください。

5. ネーミングサービスの起動

Windows32

Windows(R)の管理ツールの「サービス」を起動し、以下のネーミングサービスのサービスを開始します。

ー ネーミングサービス:「Naming Service」サービス

Linux32

CosNaming_sコマンドを使用して、ネーミングサービスを起動します。

6. 各業務サーバの起動

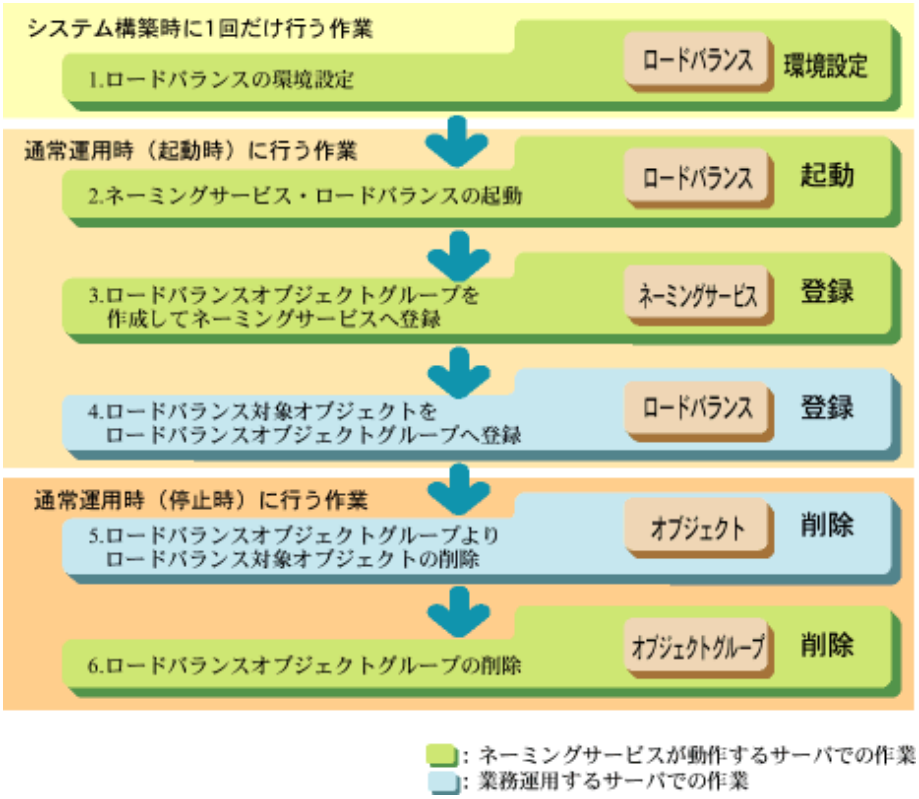
各業務サーバを起動して、起動完了したサーバの復旧をロードバランスに通知します。全サーバが起動した時点で、システム全体の起動が完了となります。
なお、トランザクションアプリケーションを使用する場合は、ワークユニット定義でネーミングサービスの手動登録を指定してください。

7. 各業務サーバの停止

サーバマシン状態監視機構などにより、ロードバランスにサーバの停止を通知することで、停止したサーバへの振り分けを停止できます。

B.3.1.2 サーバオブジェクトの起動時に毎回登録する場合

ロードバランス対象のオブジェクトを、サーバオブジェクトの起動時に毎回登録する場合の運用手順とその詳細について説明します。なお、1.～3.の作業については、固定登録の場合と同じ作業を行ってください。



4. ロードバランス対象オブジェクトをロードバランスオブジェクトグループへ登録

各業務サーバでサーバオブジェクトを起動した後、サーバオブジェクトから、ロードバランス対象オブジェクトをロードバランスオブジェクトグループへ登録します。登録作業については、固定登録の場合と同じ作業を行ってください。
なお、トランザクションアプリケーションを使用する場合は、ワークユニット定義でネーミングサービスの自動登録を指定してください。この場合、ワークユニット定義にあて先名として指定されたオブジェクト名の「/」(スラッシュ)を、「::」(2つのコロン)に置き換えたロードバランスオブジェクトグループにオブジェクトが自動的に登録されます。

5. ロードバランスオブジェクトグループよりロードバランス対象オブジェクトの削除

odadministerlbコマンドを利用して、ロードバランスオブジェクトグループからロードバランス対象オブジェクトを削除します。
ロードバランス機能インタフェースのアプリケーションを使用して、ロードバランス対象オブジェクトを削除することもできます。ロードバランス機能インタフェースのアプリケーション作成については、「アプリケーション作成ガイド(CORBAサービス編)」の「ロードバランス機能のプログラミング」を参照してください。
なお、トランザクションアプリケーションでロードバランスを使用する場合は、ワークユニットの停止により、ロードバランスオブジェクトグループからオブジェクトが自動的に削除されます。

6. ロードバランスオブジェクトグループの削除

OD_or_admコマンドを使用して、ロードバランスオブジェクトグループを削除します。
ロードバランス機能インタフェースのアプリケーションを使用して、ロードバランスオブジェクトグループを削除することもできます。ロードバランス機能インタフェースのアプリケーション作成については、「アプリケーション作成ガイド(CORBAサービス編)」の「ロードバランス機能のプログラミング」を参照してください。

B.3.2　サーバダウン時の縮退運転および復旧通知

業務サーバがダウンした場合、およびサーバが復旧した場合の対処方法について説明します。

サーバダウン時の対処方法

サーバダウンを認識した場合、まずサーバダウンの状態をロードバランスに通知する必要があります。ロードバランスには自動的に通知されないため、ダウンしたサーバのIPアドレスを特定し、odnotifydownコマンドまたはAPIを利用してダウン通知を行ってください。これにより、ロードバランスでは、ダウンしたサーバのオブジェクトリファレンスの返却を停止させ、縮退運転に切り替えます。
また、クライアント側では、オブジェクトリファレンスの取得を再度行うことにより、他サーバで動作している同一サーバオブジェクトへの接続が可能になります。

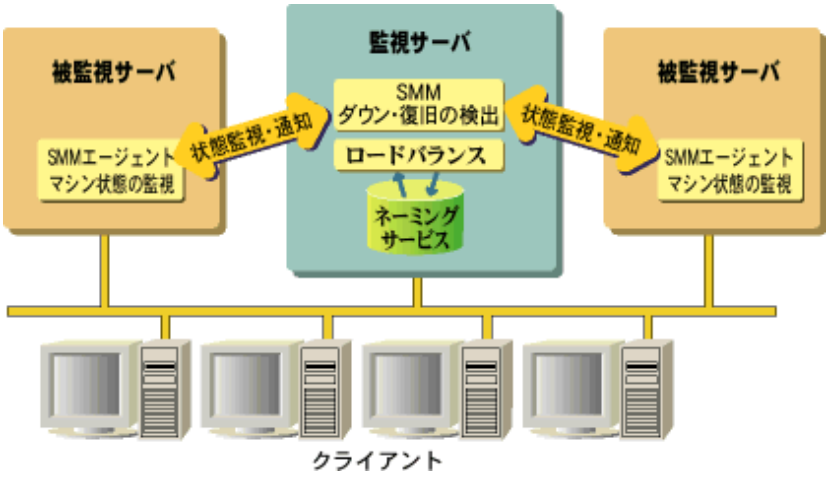
サーバ復旧時の対処方法

サーバが復旧した場合には、復旧したサーバのサーバ名を特定し、odnotifyrecoverコマンドまたはAPIを利用して、必ずロードバランスに対してサーバが復旧した旨を通知してください。復旧を通知しなかった場合、再起動後までダウンした状態が引き継がれてしまいます。サーバの復旧通知によって、復旧したサーバのオブジェクトリファレンスの返却が開始されます。
なお、サーバが復旧した旨はクライアントには通知されません。クライアントからオブジェクトリファレンスの再取得が依頼されて、はじめて、サーバの復旧はクライアント側で認識されます。

なお、サーバマシン状態監視機構を利用して、自動的に業務サーバのマシン状態を監視したり、サーバダウンや復旧をロードバランスへ通知したりできます。サーバマシン状態監視機構の利用については、「B.3.3 サーバマシン状態監視機構」を参照してください。

B.3.3 サーバマシン状態監視機構

サーバマシン状態監視機構では、ロードバランス運用時に指定したサーバマシンの状態をマシン単位で監視し、その状態をロードバランスに通知できます。サーバマシン状態監視機構で監視するサーバを監視サーバ、監視されるサーバを被監視サーバと呼びます。
以下に、サーバマシン状態監視機構の仕組みについて図を示します。



サーバマシン状態監視機構では、被監視サーバ上に置かれたSMMエージェント(Server Machine Monitor Agent : SMMA)と通信してサーバマシンの状態を監視します。このSMMエージェントからの通信を、稼働通知と呼びます。
サーバマシン状態監視機構が被監視サーバのダウン、または被監視サーバの復旧を認識すると、以下の方法でロードバランスに通知できます。なお、以下の方法は、サーバマシン状態監視機構およびSMMエージェントの起動時に指定できます。

Windows32

- サーバマシン状態監視機構から直接ロードバランスに通知
- サーバマシン状態監視機構からバッチファイルを実行してロードバランスに通知

- SMMエージェントからバッチファイルを実行してロードバランスに通知

参考

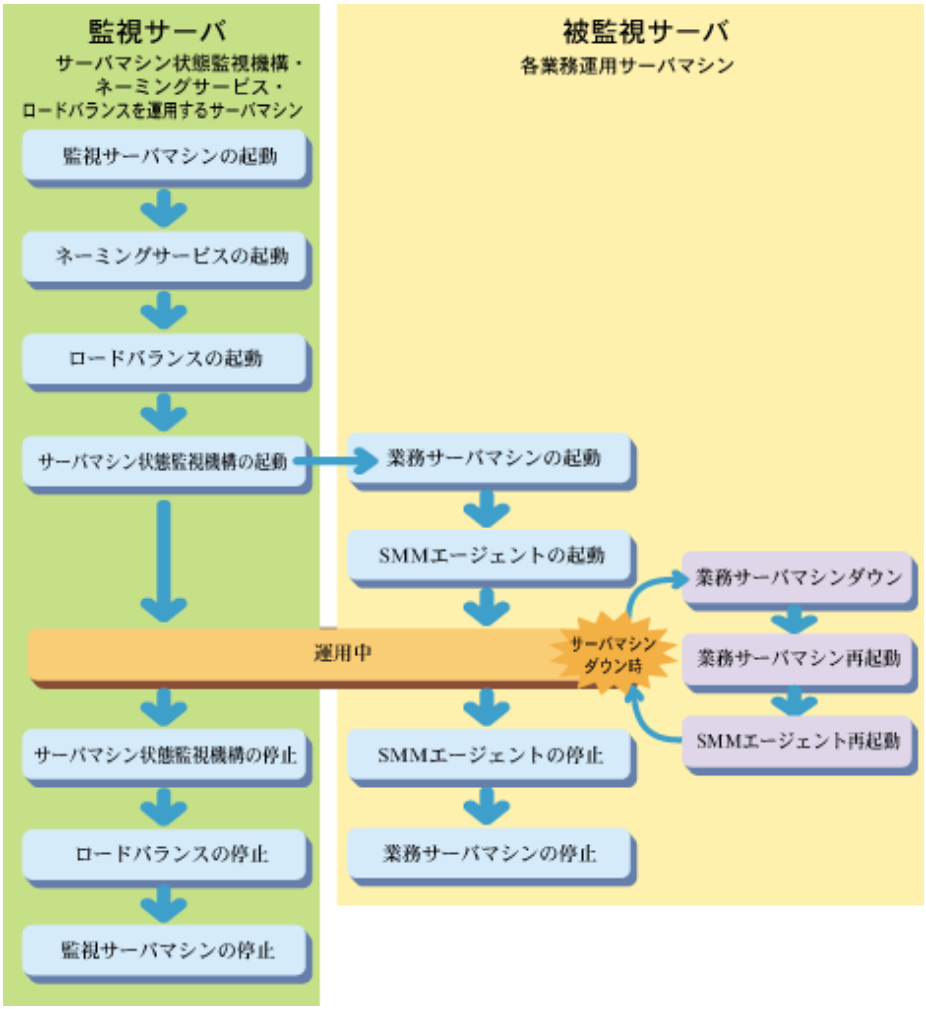
ダウン時のリカバリ処理が容易になるなど、保守の面から以下のものについては任意のサーバ上にひとつにまとめて配置し、マシン状態の監視サーバとして独立運用させることを推奨します。

- サーバマシン状態監視機構
- ロードバランス
- ネーミングサービス

Windows32

また、この監視サーバの待機サーバを用意して、マシン異常時に運用を切り替えることで、信頼性を向上させることができます。詳細については、「1.2 クラスタサービス機能」を参照してください。

以下に、監視サーバ側および被監視サーバ側の運用手順について図を示します。



サーバダウン時に実行されるダウンバッチファイル(Windows)/ダウンシェルプログラム(Solaris/Linux)、およびサーバ復旧時に実行される復旧バッチファイル(Windows)/シェルプログラム(Solaris/Linux)はユーザ側で作成します。
なお、作成するバッチファイル(Windows)/シェルプログラム(Solaris/Linux)には、以下のコマンドを記述してください。ただし、ロードバランスへの通知が不要の場合は、記述する必要はありません。

サーバダウン時

ロードバランスへダウンを通知するために、odnotifydownコマンドを記述してください。

サーバ復旧時

ロードバランスへ復旧を通知するために、odnotifyrecoverコマンドを記述してください。

注意

Windows32

「Naming Service」サービスがSMMの起動の依存関係サービスなので、SMMは必ず「Naming Service」サービスが動作しているマシンで起動してください。

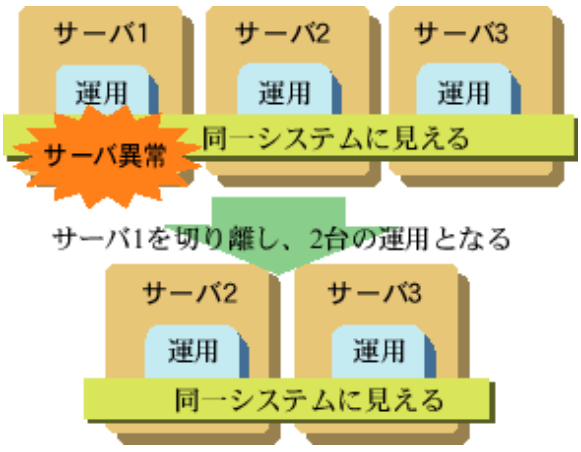
B.4 高信頼化システムの設計

B.4.1 ロードバランスによる縮退を利用した高信頼化システム

Interstageシステムを高信頼化するためのロードバランスによる縮退を利用した高信頼化システムについて説明します。

高信頼化の方式

複数台のサーバを同一システムとして処理を並列に行い、クライアントからの処理を自動的に分散させます。サーバ異常時はサーバを切り離して、残りのサーバで運用を行います。



高信頼化の目的

サーバ間の負荷調整を自動的に行いたい。

特徴

サーバマシン異常に対する高信頼化を構築可能。クライアントからの負荷を自動的に分散させます。また、サーバ追加を容易に行うことができます。ネーミングサービスを設定しているサーバに関しては、クラスタシステムを使用し、高信頼化することを推奨します。

B.4.2 ロードバランス機能での縮退

複数のサーバを同一システムとして処理を並列に行い、サーバ故障時は当該サーバを切り離して、残りのサーバで運用を行います。複数のサーバを1つのシステムとして使用できるため、業務に対する負荷を分散させることができます。そのため、大規模なシステムを構築する場合に適しています。

Interstage Application Server Enterprise Editionのロードバランスによる縮退を利用した高信頼化システムを構築する場合、以下の選択肢があります。

- DB使用形態
  DBを使用しないDB未使用パターンと、ノード間でDBを共用しないDB非共用パターン、ノード間でDBを共用するDB共用パターンが選択できます。
- 使用DBMS製品
  DB使用形態にあったDBMS製品を選択します。

以下に組み合わせを示します。

DB使用形態	DB未使用	DB非共用	DB共用
使用DBMS	―	DBMS製品	ノード間でのDB共用に対応したDBMS製品
パターン	パターン1	パターン2	パターン3

注意

DB共用型については、クラスタシステムが必要となります。

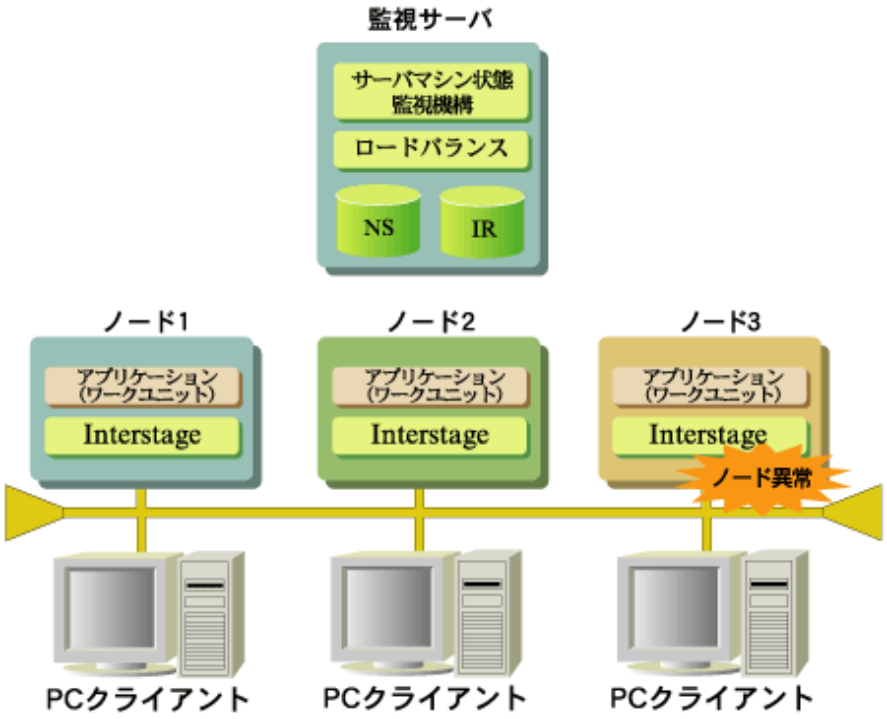
業務復旧までの時間

DB共用を使用していない場合は、ノードダウン時も業務が停止することはありません。
DB共用の場合は、ダウンしたサーバで使用していたテーブルに関しては、復旧まで約数十秒必要です。ただし、トランザクション数によってはさらに時間がかかる場合もあります。

以降では、上記のパターンについて説明します。

パターン1 (DB未使用)

以下に、ロードバランス機能を使用した縮退で、DBMSを使用しない場合について説明します。



Interstageのロードバランス機能は、複数台のサーバに対して1台の監視サーバを用意します。また、監視サーバ上にネーミングサービスおよびインタフェースリポジトリを作成します。

監視サーバでInterstageが独自に監視を行い、ノードダウン時は縮退を行います。

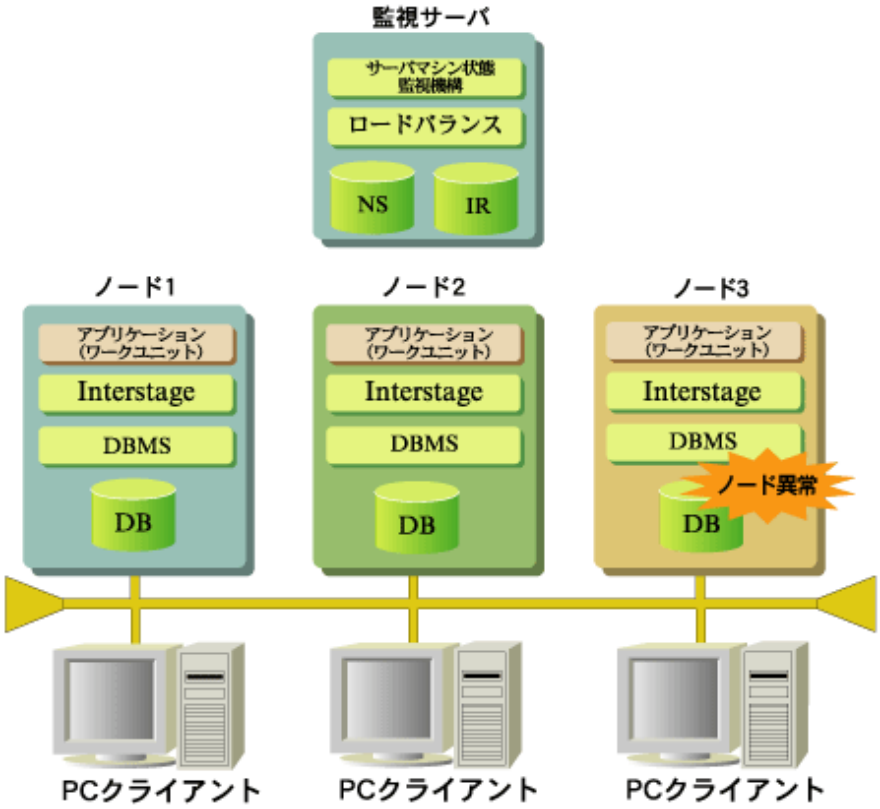
また、監視サーバの停止に備え、クラスタシステムを使用し監視サーバを二重化することによりさらに信頼性があがります。監視サーバの二重化については、「1.2 クラスタサービス機能」を参照してください。

必須製品

- Interstage Application Server Enterprise Edition

パターン2 (DB非共用)

以下に、ロードバランス機能を使用した縮退で、DBMSを共用しない(ノード間で非共用)場合について説明します。



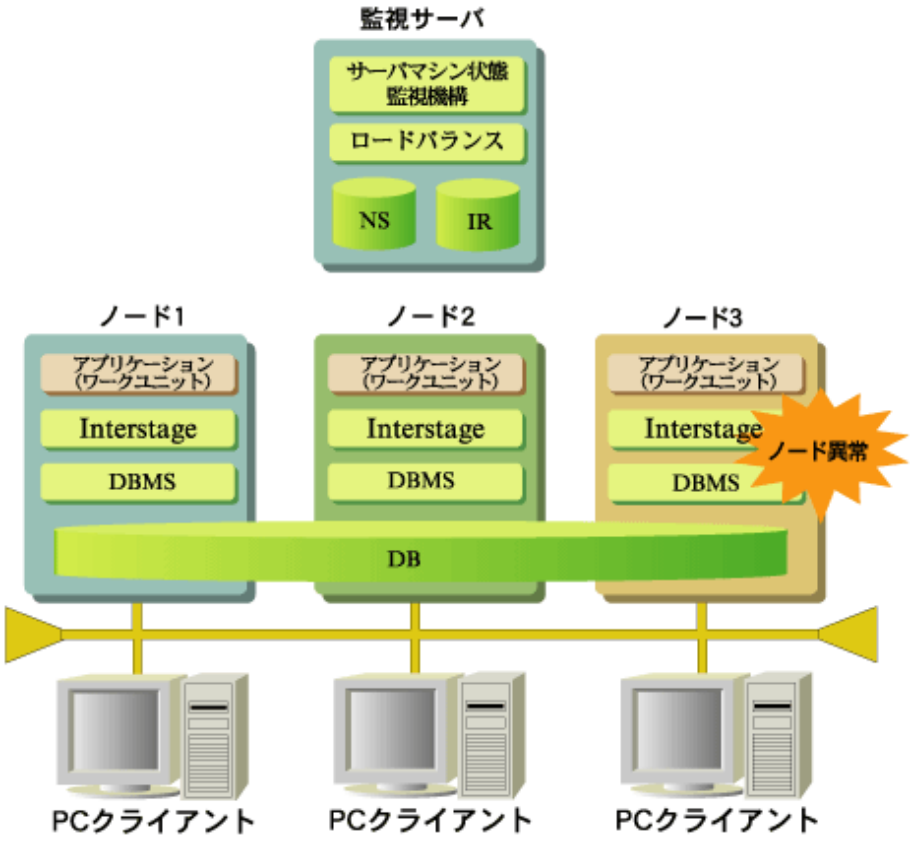
Interstageのロードバランス機能は、複数台のサーバに対して1台の監視サーバを用意します。また、監視サーバ上にネーミングサービスおよびインタフェースリポジトリを作成します。
監視サーバでInterstageが独自に監視を行い、ノードダウン時は縮退を行います。
また、監視サーバの停止に備え、クラスタシステムを使用し監視サーバを二重化することによりさらに信頼性があがります。監視サーバの二重化については、「1.2 クラスタサービス機能」を参照してください。

必須製品

- Interstage Application Server Enterprise Edition
- DBMS製品

パターン3 (DB共用)

以下に、ロードバランス機能を使用した縮退で、DBMSを共用する(ノード間で共用)場合について説明します。



Interstageのロードバランス機能は、複数台のサーバに対して1台の監視サーバを用意します。また、監視サーバ上にネーミングサービスおよびインタフェースリポジトリを作成します。
監視サーバでInterstageが独自に監視を行い、ノードダウン時は縮退を行います。

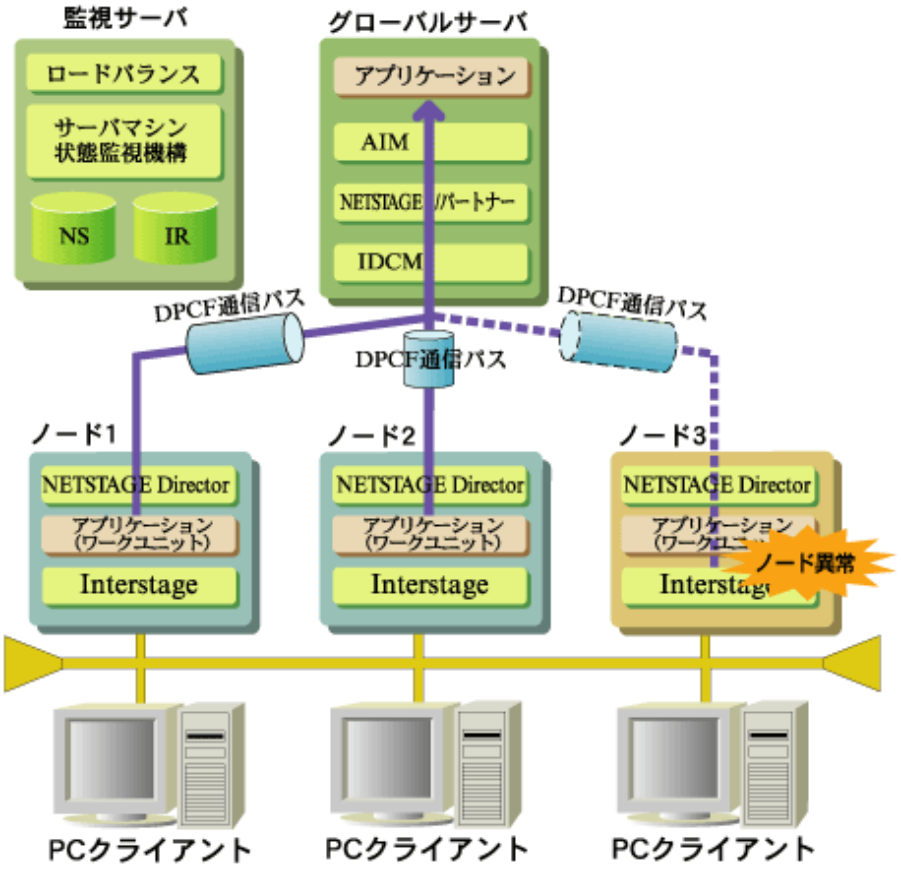
また、監視サーバの停止に備え、クラスタシステムを使用し監視サーバを二重化することによりさらに信頼性があがります。監視サーバの二重化については、「1.2 クラスタサービス機能」を参照してください。

必須製品

- Interstage Application Server Enterprise Edition
- ノード間でのDB共用に対応したDBMS製品

B.4.3 AIM連携での高信頼化システム Windows32

AIMアプリケーションと連携を行うAIM連携を行った場合の高信頼化として、「B.4.2 ロードバランス機能での縮退」で説明したロードバランスによる縮退を利用した場合について説明します。



AIM連携使用時はInterstageシステムが動作するサーバについての高信頼化を保証します。
グローバルサーバについては、グローバルサーバの高信頼化を行ってください。
Interstageシステム側の高信頼化については、「B.4.2 ロードバランス機能での縮退」を参照してください。

必須製品

- Interstage Application Server Enterprise Edition
- NETSTAGE Director

索 引