BEA WebLogic Server™

WebLogic Server
ドメイン管理

BEA WebLogic Server バージョン 7.0
改訂 : 2002 年 9 月 4 日

WebLogic Server ドメイン管理

パート番号	マニュアルの日付	ソフトウェアのバージョン
なし	2002 年 9 月 4 日	BEA WebLogic Server バージョン 7.0

目次

3. ネットワーク リソースのコンフィグレーション

4. 障害が発生したサーバの回復

5. ノード マネージャによるサーバの可用性の管理

6. WebLogic Sever ドメインのモニタ

このマニュアルの内容

このマニュアルでは、BEA WebLogic Server™ ドメインのリソースをコンフィグレーション、管理、およびモニタする方法について説明します。

このマニュアルの内容は以下のとおりです。

- 第 1 章「WebLogic Server ドメインの概要」では、WebLogic Server ドメインの概念および一般的なドメイン管理タスクについて説明します。
- 第 2 章「コンフィグレーション ウィザードを使用した新しいドメインの作成」では、コンフィグレーション ウィザードを使用して新しいドメインを作成する方法について説明します。
- 第 3 章「ネットワーク リソースのコンフィグレーション」では、ドメインで利用可能なネットワーク リソースをコンフィグレーションして、サーバおよびクラスタで使用する方法について説明します。
- 第 4 章「障害が発生したサーバの回復」では、ドメイン内のサーバ障害に対する備えと対処方法について説明します。
- 第 5 章「ノード マネージャによるサーバの可用性の管理」では、ノード マネージャをコンフィグレーションして、ドメイン内の管理対象サーバを手動または自動で起動および停止させる方法について説明します。
- 第 6 章「WebLogic Sever ドメインのモニタ」では、ドメイン内の管理対象サーバをモニタする方法について説明します。

対象読者

このマニュアルは、ドメイン内の複数のサーバまたはクラスタを管理する WebLogic Server 管理者を対象としています。

e-docs Web サイト

BEA 製品のドキュメントは、BEA の Web サイトで入手できます。BEA のホーム ページで [製品のドキュメント] をクリックします。

このマニュアルの印刷方法

Web ブラウザの [ファイル | 印刷] オプションを使用すると、Web ブラウザからこのマニュアルを一度に 1 章ずつ印刷できます。

このマニュアルの PDF 版は、WebLogic Server の Web サイトで入手できます。PDF を Adobe Acrobat Reader で開くと、マニュアルの全体 (または一部分) を書籍の形式で印刷できます。PDF を表示するには、WebLogic Server ドキュメントのホーム ページを開き、[ドキュメントのダウンロード] をクリックして、印刷するマニュアルを選択します。

Adobe Acrobat Reader は Adobe の Web サイト (http://www.adobe.co.jp) で無料で入手できます。

関連情報

BEA の Web サイトでは、WebLogic Server の全マニュアルを提供しています。WebLogic Server のシステム管理に関する詳細については、システム管理のページを参照してください。

サポート情報

BEA のドキュメントに関するユーザからのフィードバックは弊社にとって非常に重要です。質問や意見などがあれば、電子メールで docsupport-jp@beasys.com までお送りください。寄せられた意見については、ドキュメントを作成および改訂する BEA の専門の担当者が直に目を通します。

電子メールのメッセージには、ご使用のソフトウェアの名前とバージョン、およびドキュメントのタイトルと日付をお書き添えください。本バージョンの BEA WebLogic Server について不明な点がある場合、または BEA WebLogic Server のインストールおよび動作に問題がある場合は、BEA WebSupport (www.bea.com) を通じて BEA カスタマ サポートまでお問い合わせください。カスタマ サポートへの連絡方法については、製品パッケージに同梱されているカスタマ サポート カードにも記載されています。

カスタマ サポートでは以下の情報をお尋ねしますので、お問い合わせの際はあらかじめご用意ください。

- お名前、電子メール アドレス、電話番号、ファクス番号
- 会社の名前と住所
- お使いの機種とコード番号
- 製品の名前とバージョン
- 問題の状況と表示されるエラー メッセージの内容

表記規則

このマニュアルでは、全体を通して以下の表記規則が使用されています。

表記法	適用
〔Ctrl〕+ 〔Tab〕	複数のキーを同時に押すことを示す。
斜体	強調または書籍のタイトルを示す。

等幅テキスト	コード サンプル、コマンドとそのオプション、データ構造体とそのメンバー、データ型、ディレクトリ、およびファイル名とその拡張子を示す。等幅テキストはキーボードから入力するテキストも示す。
	例:
	`import java.util.Enumeration;`
	`chmod u+w *`
	`config/examples/applications`
	`.java`
	`config.xml`
	`float`

斜体の等幅テキスト	コード内の変数を示す。 例: `String CustomerName;`
すべて大文字のテキスト	デバイス名、環境変数、および論理演算子を示す。 例: LPT1 BEA_HOME OR
{ }	構文の中で複数の選択肢を示す。
[ ]	構文の中で任意指定の項目を示す。 例: `java utils.MulticastTest -n name -a address [-p portnumber] [-t timeout] [-s send]`

表記法	適用

|	構文の中で相互に排他的な選択肢を区切る。
	例：
	`java weblogic.deploy [list|deploy|undeploy|update]`
	`password {application} {source}`

...	コマンドラインで以下のいずれかを示す。 ■ 引数を複数回繰り返すことができる。 ■ 任意指定の引数が省略されている。 ■ パラメータや値などの情報を追加入力できる。
. . .	コード サンプルまたは構文で項目が省略されていることを示す。

1　WebLogic Server ドメインの概要

以下の節では、WebLogic Server ドメインとその内容について説明します。

ドメインとは

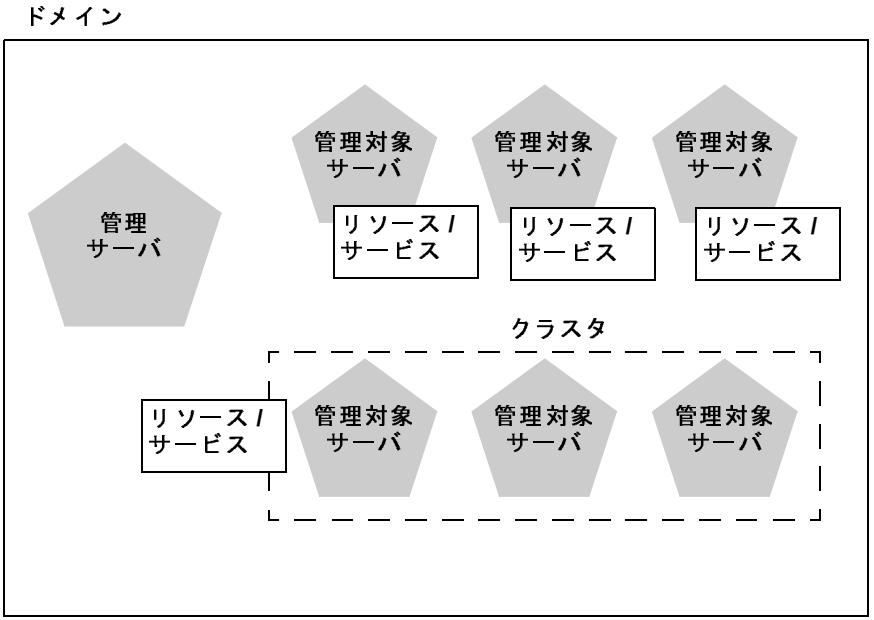
ドメインは、WebLogic Server インスタンスの基本的な管理単位です。1 つまたは複数の WebLogic Server インスタンス (および、それに関連付けられたリソース) で構成され、それらのインスタンスが単一の管理サーバで管理されます。各システム管理者の責任範囲、アプリケーションの境界、サーバの設置場所などに基づいて、複数のドメインを定義できます。また、ドメインを 1 つにして、すべての WebLogic Server 管理アクティビティを一元化することも可能です。

複数のドメインを作成した場合は、各ドメインのコンフィグレーション ファイル（`config.xml`）が別々になります。そのため、コンフィグレーション タスクまたはデプロイメント タスクを複数のドメインで同時に行うことはできません。Administration Console でコンフィグレーション タスクを行う場合、変更は現在選択されているドメインに対して適用されます。別のドメインで変更を行うには、そのドメインを選択しなくてはなりません。したがって、あるドメイン内のサーバ インスタンス、アプリケーション、リソースは、別のドメイン内のサーバ、アプリケーション、リソースから独立したものとして扱う必要があります。

ドメインの内容

ドメインには複数の WebLogic Server クラスタと、クラスタ化されていない WebLogic Server インスタンスが存在できます。厳密に言うと、1 つの WebLogic Server インスタンスだけでもドメインを構成できます。ただしその場合、その唯一のサーバ インスタンスが管理サーバになります。これは、各ドメインには必ず 1 つの管理サーバが存在する必要があるためです。ドメインの範囲と目的は多種多様ですが、この節では、ほとんどの WebLogic Server ドメインに含まれるコンポーネントについて説明します。

次の図に、管理サーバ、3 つのスタンドアロン管理対象サーバ、および 3 つの管理対象サーバを含むクラスタで構成されたプロダクション環境を示します。



管理サーバ

各 WebLogic Server ドメインには、管理サーバとして動作するサーバ インスタンスが 1 つ必要です。その管理サーバをプログラム的に、または Administration Console を介して使用することで、ドメイン内にある他のすべてのサーバ インスタンスおよびリソースをコンフィグレーションします。

管理サーバの役割

ドメイン内の管理対象サーバを起動するには、まず管理サーバを起動します。スタンドアロンの管理対象サーバ、またはクラスタ化された管理対象サーバを起動すると、その管理対象サーバは管理サーバにアクセスしてコンフィグレーション情報を取得します。このように、管理サーバはドメイン全体のコンフィグレーションの一元的な制御エンティティとして動作します。

管理サーバのサービスは、以下の方法で起動できます。

- ドメイン コンフィグレーション ウィザード — ドメイン コンフィグレーション ウィザードは、ドメインまたはクラスタを新規作成するための推奨ツールです。
- WebLogic Server Administration Console—Administration Console は、管理サーバとのグラフィカル ユーザ インタフェース (GUI) です。
- WebLogic Server アプリケーション プログラミング インタフェース（API）—WebLogic Server に付属の API を使用してコンフィグレーション属性を修正するプログラムを記述できます。
- WebLogic Server コマンドライン ユーティリティ — このユーティリティを使用すると、ドメインの管理を自動化するスクリプトを作成できます。

どの方法を使用する場合でも、ドメインの管理サーバが実行されていなければ、ドメインのコンフィグレーションを修正することはできません。

管理サーバを起動すると、その管理サーバではドメインの `config.xml` がロードされます。`config.xml` は、カレント ディレクトリで検索されます。ドメインの作成時に別のディレクトリを指定しない限り、`config.xml` は次のディレクトリに格納されます。

```
BEA_HOME/user_projects/mydomain
```

`mydomain` は、特定のドメインと同じ名前を持つそのドメイン固有のディレクトリです。

管理サーバが正常に起動すると、そのたびに、config.xml.booted という名前のバックアップ用コンフィグレーション ファイルがドメイン専用のディレクトリに作成されます。これにより、万が一サーバが終了するまでの間に config.xml ファイルが壊れるようなことがあっても、以前の状態のコンフィグレーションに戻ることができます。

管理サーバおよび WebLogic Server JMX 管理システムにおけるロールについては、『管理者ガイド』の「システム管理ツール」を参照してください。

管理サーバに障害が発生した場合

ドメインの管理サーバで障害が発生しても、ドメイン内の管理対象サーバの動作には影響しません。ドメインの管理サーバが使用できなくなっても、その管理サーバが管理対象とする (クラスタ化された、またはされていない) サーバ インスタンスが起動および実行されていれば、それらの管理対象サーバは処理を続けます。ドメインにクラスタ化されたサーバ インスタンスがある場合には、管理サーバに障害が発生しても、ドメイン コンフィグレーションでサポートされているロード バランシングおよびフェイルオーバー機能を引き続き使用できます。

ホスト マシン上のハードウェアまたはソフトウェアの障害が原因で管理サーバに障害が発生した場合、同じマシン上の他のサーバ インスタンスも同様に影響を受けることがあります。ただし、管理サーバ自体の障害がドメイン内の管理対象サーバの動作を妨げることはありません。

管理サーバを再起動する手順については、4-10 ページの「管理対象サーバの動作中における管理サーバの再起動」を参照してください。

管理対象サーバとクラスタ化された管理対象サーバ

管理サーバ以外のドメイン内のサーバ インスタンスは、管理対象サーバと呼ばれます。管理対象サーバには、アプリケーションを構成するコンポーネントと関連するリソース (JSP や EJB など) がホストされます。管理対象サーバは起動時にドメインの管理サーバに接続し、コンフィグレーションとデプロイメントの設定を取得します。

注意 : 管理サーバが使用不可能な状態のときに、管理サーバから独立してドメイン内の管理対象サーバを起動できます。詳細については、4-1 ページの「障害が発生したサーバの回復」を参照してください。

複数の管理対象サーバを WebLogic Server クラスタとしてコンフィグレーションすると、アプリケーションのスケーラビリティと可用性を向上させることができます。WebLogic Server クラスタ内では、リソースとサービスのほとんどが (単一の管理対象サーバにではなく) 各々の管理対象サーバにデプロイされ、それによってフェイルオーバーとロード バランシングが実現されます。どのコンポーネント タイプおよびサービスをクラスタ化できるのか (クラスタ内のすべてのサーバ インスタンスにデプロイできるのか) については、『WebLogic Server クラスタ ユーザーズ ガイド』の「クラスタ化可能なオブジェクトの種類」を参照してください。

クラスタ化されていない管理対象サーバを作成した後でそのサーバをクラスタに追加するには、そのサーバ インスタンスとクラスタに適切なコンフィグレーション パラメータをコンフィグレーションします。逆に、クラスタから管理対象サーバを削除するには、追加時にコンフィグレーションしたパラメータを適切に再コンフィグレーションします。管理対象サーバをクラスタ化した場合とクラスタ化しない場合の主な違いは、フェイルオーバーおよびロード バランシングをサポートするかどうかです。これらの機能は、管理対象サーバをクラスタ化した場合にのみ利用できます。

管理対象サーバをクラスタ化するかどうかは、スケーラビリティと信頼性に対する要件の度合いによって決まります。たとえば、アプリケーションが負荷の変化に影響を受けず、アプリケーション サービスで起こる可能性のある中断を許容できる場合には、クラスタ化の必要はありません。

WebLogic Server クラスタのメリットと機能の詳細については、『WebLogic Server クラスタ ユーザーズ ガイド』の「WebLogic Server クラスタ化の概要」を参照してください。単一ドメインには、クラスタとしてコンフィグレーションされていない複数の管理対象サーバを設定できるのと同様に、複数の WebLogic Server クラスタを設定することもできます。

リソースとサービス

ドメインには、管理サーバと管理対象サーバだけでなく、ドメイン内にデプロイされている管理対象サーバとその管理対象サーバにホストされるアプリケーションで必要とされる、リソースとサービスも含まれます。

ドメインレベルのリソースの例を次に示します。

- 特定の物理的なハードウェアの識別に使用するマシン定義。マシン定義は、コンピュータとそのコンピュータのホストする管理対象サーバの関連付けに使用します。この情報は、障害の発生した管理対象サーバを再起動するとき

にノード マネージャで使用されます。また、クラスタ化された管理対象サーバでは、レプリケートされたセッション データを格納する最適な場所を選択する際にこの情報を使用します。ノード マネージャの詳細については、5-1 ページの「ノード マネージャによるサーバの可用性の管理」を参照してください。

- デフォルトのポート、プロトコル、およびプロトコル設定の定義に使用できるオプションのリソースである、ネットワーク チャネル。ネットワーク チャネルは作成後、ドメイン内にある任意の数の管理対象サーバとクラスタに割り当てることができます。詳細については、3-1 ページの「ネットワーク リソースのコンフィグレーション」を参照してください。

ドメイン内の管理対象サーバは、そのドメイン独自のリソースとサービスをホストします。選択した管理対象サーバまたはクラスタに対して、リソースおよびサービスをデプロイできます。デプロイ可能なリソースの例を次に示します。

- アプリケーション コンポーネント (EJB など)
- コネクタ、起動クラス
- JDBC 接続プール
- JMS サーバ

一般的なドメインのタイプ

ドメインの基本的なタイプには次の 2 つがあります。

- **管理対象サーバのあるドメイン :** アプリケーションをホストする複数の管理対象サーバと、管理操作を実行する 1 つの管理サーバをドメインに含めることで、シンプルなプロダクション環境を構成できます。このコンフィグレーションでは、アプリケーションとリソースは個々の管理対象サーバにデプロイされ、アプリケーションにアクセスするクライアントは個々の管理対象サーバに接続します。

  アプリケーションのパフォーマンス、スループット、または可用性を高度に保つ必要のあるプロダクション環境では、複数の管理対象サーバをクラスタとしてコンフィグレーションできます。クラスタ化を行うと、複数の管理対象サーバが 1 つのユニットとして機能して、アプリケーションおよびリソースをホストできるようになります。スタンドアロンの管理対象サーバとクラ

スタ化された管理対象サーバの違いについては、1-4 ページの「管理対象サーバとクラスタ化された管理対象サーバ」を参照してください。

- **スタンドアロン サーバのドメイン：** 開発環境やテスト環境では、プロダクション ドメイン内のサーバとは独立して、単一のアプリケーションとサーバをデプロイする必要のある場合があります。そのような場合に、管理サーバとして機能し、かつ開発中のアプリケーションのホストとしても機能する単一のサーバ インスタンスで構成された、シンプルなドメインをデプロイすることが可能です。WebLogic Server と一緒にインストールできる examples ドメインは、スタンドアロン サーバ ドメインの一例です。

注意： プロダクション環境では、ドメイン内の管理対象サーバだけにアプリケーションをデプロイすることをお勧めします。管理サーバは管理タスク用に確保しておく必要があります。

ドメインの制限事項

WebLogic Server のインストール環境は単一のドメインで構成されることが多く、アプリケーションをホストするために必要なすべての管理対象サーバをそのドメインに含む形で利用されています。ドメインを複数作成する場合には、以下の制限事項に注意してください。

- 各ドメインには、管理アクティビティを実行するための固有の管理サーバが必要です。Administration Console を使用して管理タスクやモニタ タスクを実行する場合には、ドメインを切り替えることができますが、その切り替えを行うときには、別の管理サーバに接続することになります。
- クラスタ内のすべての管理対象サーバは、同じドメインに存在しなければなりません。クラスタを複数のドメインにわたって「分割」することはできません。
- コンフィグレーションしたリソースまたはサブシステムをドメイン間で共有することはできません。たとえば、あるドメインで JDBC 接続プールを作成した場合、別のドメインの管理対象サーバまたはクラスタでその接続プールを使用することはできません (代わりに、2 番目のドメインで同様の接続プールを作成する必要があります)。

ドメイン ディレクトリと config.xml

ドメインのコンフィグレーションは、そのドメインの config.xml ファイルに格納されます。config.xml には、ドメインの名前と、ドメイン内の各サーバ インスタンス、クラスタ、リソース、およびサービスのコンフィグレーション パラメータ設定が指定されます。ドメインの config.xml は、そのドメインの作成時に指定したドメイン ディレクトリに格納されます。

ドメイン ディレクトリには、ドメイン内の管理サーバと管理対象サーバの起動に使用できるデフォルトのスクリプト ファイルも格納されます。それらのスクリプト、およびサーバ インスタンスを起動する他の方法の詳細については、『管理者ガイド』の「WebLogic Server の起動と停止」を参照してください。

ドメイン ディレクトリの構造

WebLogic Server 7.0 より前のリリースでは、Weblogic Server のディレクトリ構造内にドメイン ディレクトリが作成されていました。WebLogic Server 7.0 以降では、製品のディレクトリ ツリー外部の、WebLogic Server および JDK にアクセスできるシステム上の任意の位置にドメイン ディレクトリを設定できます。

この新しいディレクトリ構造にはさらに高度な柔軟性があり、WebLogic Server の実行可能ファイルやその関連ファイルとは独立したディレクトリ構造に、アプリケーション コードを格納できます (この方法をお勧めします)。

ドメイン ディレクトリ構造には次の要素が必要です。

- ドメインと同じ名前のルート ディレクトリ (たとえば mydomain、petstore)。ルート ディレクトリには以下を格納します。
  - ドメインのコンフィグレーション ファイル (通常は config.xml)
  - サーバ インスタンスの起動および環境の確立に使用するスクリプト
- ドメインのアプリケーションを格納するサブディレクトリ (通常は applications)

注意： WebLogic Server の自動デプロイメント機能 (ドメインが開発モードで動作している場合に利用可能) を使用する場合、アプリケーションのサブディレクトリは `applications` という名前にする必要があります。自動デプロイメントの詳細については、『WebLogic Server アプリケーションの開発』の「自動デプロイメント」を参照してください。

ドメイン ディレクトリ構造には必要に応じて他のディレクトリを作成することもできます。ドメイン内のサーバ インスタンスの初回起動時には、ドメイン ディレクトリに以下のサブディレクトリが作成されます。

- `data` : セキュリティ情報を格納します。
- `logs` : ドメインレベルのログを格納します。
- *`server_name`* : ドメイン内で実行する各サーバについて、サーバレベルのログを格納します。
- `temp` : 一時ファイルを格納します。

次に例を示します。

ドメインは、コンフィグレーション ウィザードを使用して作成、コンフィグレーションできます。カスタム インストールを行うと、手順の最後に、コンフィグレーション ウィザードを実行するかどうかを尋ねるプロンプトが表示されます。コンフィグレーション ウィザードは、[スタート] メニューまたはスクリプトを使用して起動することもできます。詳細については、2-1 ページの「コンフィグレーション ウィザードを使用した新しいドメインの作成」を参照してください。

コンフィグレーション ウィザードを使用してドメインを作成すると、ドメインのコンテナとなる `user_projects` ディレクトリが BEA ホーム ディレクトリに作成されます。また、新しいドメインのルート ディレクトリ、およびドメイン作成時に選択したドメイン テンプレートに指定されているその他のディレクトリもすべて作成されます。

ドメインの管理タスク

一般的なドメイン管理タスクについては、以下の節を参照してください。

- 2-1 ページの「コンフィグレーション ウィザードを使用した新しいドメインの作成」
- 3-1 ページの「ネットワーク リソースのコンフィグレーション」
- 4-1 ページの「障害が発生したサーバの回復」
- 5-1 ページの「ノード マネージャによるサーバの可用性の管理」

2　コンフィグレーション ウィザードを使用した新しいドメインの作成

以下の節では、コンフィグレーション ウィザードを使用して WebLogic Server ドメインを作成する方法について説明します。

- 2-1 ページの「コンフィグレーション ウィザードの概要」
- 2-7 ページの「サーバ名とリスン アドレスの指定」
- 2-9 ページの「コンフィグレーション ウィザードの使用」

コンフィグレーション ウィザードの概要

コンフィグレーション ウィザードは、カスタマイズされた新しい WebLogic Server ドメインの作成を支援するアプリケーションです。スタンドアロンのアプリケーションであるため、コンフィグレーション ウィザードを実行するために WebLogic Server を起動する必要はありません。コンフィグレーション ウィザードは、WebLogic Server のインストール時に自動的にインストールされます。

WebLogic Server のカスタム インストールの最後には、必要に応じてコンフィグレーション ウィザードを実行することができます。それ以降にコンフィグレーション ウィザードを起動するには、2-10 ページの「コンフィグレーション ウィザードの起動」で説明する手順に従ってください。

コンフィグレーション ウィザードでは、個々のサーバを追加することでドメインを作成または拡張するためのダイアログが表示されます。最初にドメイン テンプレートを選択します。ドメイン テンプレートでは、新しいドメインについて、以下のようなおおよその特徴が定義されています。

- ドメインにあらかじめコンフィグレーションされた WebLogic Server サンプル アプリケーションまたは Pet Store アプリケーションが含まれるか。
- コンフィグレーションする管理対象サーバの数。

- 新しい管理対象サーバをクラスタとしてコンフィグレーションするのか。

コンフィグレーション ウィザードでは、ドメインの特性を指定できる基底のテンプレートおよびダイアログを使用することでドメインの作成プロセスが簡略化されます。

コンフィグレーション ウィザードが作成するもの

表示されたダイアログに対する応答に基づいて、コンフィグレーション ウィザードではドメインの `config.xml` ファイルが作成されます。また、ドメインのサーバ インスタンスの起動スクリプト、および新しいドメインとそのサーバを起動および使用するのに役立つ他のヘルパー ファイルとディレクトリも作成されます。

コンフィグレーション ウィザードは `config.xml` ファイルとその他の生成されたコンポーネントを、ダイアログで指定されたドメイン ディレクトリに格納します (デフォルトでは `/user_projects/mydomain`)。

次の表に、コンフィグレーション ウィザードで作成されるファイルとディレクトリを示します。

表 2-1 コンフィグレーション ウィザードでインストールされるコンポーネント

コンポーネント	機能
config.xml	config.xml ファイルは、WebLogic Server ドメインのコンフィグレーションを記述する XML ドキュメント。config.xml ファイルの内容および構造は、関連付けられた文書型定義 (DTD) である config.dtd に定義される。 config.xml は一連の XML 要素で構成される。Domain 要素はトップレベルの要素であり、Domain 内の要素はすべて Domain 要素を継承する。Domain 要素には、Server、Cluster、Application などの子要素がある。それらの子要素の中にさらに子要素がある場合もある。たとえば、Server 要素には、子要素として WebServer、SSL、および Log がある。Application 要素には EJBComponent と WebAppComponent という子要素がある。 各要素には、コンフィグレーション可能な 1 つ以上の属性がある。コンフィグレーション API には、config.dtd に定義される属性に対応する属性がある。たとえば、Server 要素には ListenPort 属性があり、同様に、weblogic.management.configuration.ServerMBean にも ListenPort 属性がある。コンフィグレーションできる属性は読み書きが可能。たとえば、ServerMBean には getListenPort メソッドと setListenPort メソッドがある。 config.xml の詳細については、『コンフィグレーション リファレンス』を参照。

コンポーネント	機能
/applications	ドメイン内のサーバのデフォルト Web アプリケーションを格納する。WebLogic Server サンプルを含むテンプレートを選択した場合、このディレクトリには個別のサンプルの JAR ファイルおよび EAR ファイルも格納される。 なお、WebLogic Server サンプル アプリケーションまたは Petstore アプリケーションをインストールするテンプレートを選択すると、examplesWebApp および petstore というアプリケーション ファイルが、WebLogic Server のインストール ディレクトリの \samples\server\stage サブディレクトリに格納される。
/logs	ドメイン内のサーバのログ ファイルを格納する。
setEnv.cmd、setEnv.sh	ドメインのサーバの環境変数を設定する。
setExamplesEnv.cmd、setExamplesEnv.sh	WebLogic Server サンプル アプリケーションを含むドメインの環境変数を設定する。
startWebLogic.cmd、startWebLogic.sh	カスタム ドメインの管理サーバを起動する。
startManagedWebLogic.cmd、startManagedWebLogic.sh	カスタム ドメインの管理対象サーバを起動する。
startExamplesServer.cmd、startExamplesServer.sh	WebLogic Server サンプル アプリケーションをホストするサーバを起動する。
startPetStore.cmd、startPetStore.sh	Pet Store サンプル アプリケーションをホストするサーバを起動する。
demokey.pem、democert.pem	ドメイン内のサーバにサンプルの SSL プロトコル サポートを提供する。
Windows の [スタート] メニュー項目	Windows 環境でドメインのサーバを起動するための [スタート] メニューがサポートされる。

コンフィグレーション ウィザードのテンプレートについて

コンフィグレーション ウィザードで利用可能なテンプレートは、実際の環境にどの WebLogic Platform コンポーネントがインストールされたのかによって異なります。この節では、WebLogic Server で利用できる典型的なテンプレートについて説明します。他の WebLogic Platform コンポーネントがインストールされている場合は、他のテンプレートもコンフィグレーション ウィザードで利用できる場合があります。他のテンプレートについては、『コンフィグレーション ウィザード テンプレート リファレンス』を参照してください。

WebLogic Server では、コンフィグレーション ウィザードのテンプレートが以下の主要なカテゴリに分けられています。

- WLS Domain - このカテゴリのテンプレートでは、1 つのサーバ インスタンスのみを含む新しいドメインが作成されます。ドメインにはアプリケーションは含まれません。
- WLS Examples - このカテゴリのテンプレートでは、1 つのサーバ インスタンスを含む新しいドメインが作成されます。サーバ インスタンスは管理サーバとして動作し、WebLogic Server のサンプル アプリケーションを実行するようにコンフィグレーションされます。このサーバは、WebLogic Server の標準インストールを選択した場合に作成される `examplesServer` と同様です。
- WLS Petstore - このカテゴリのテンプレートでは、1 つのサーバ インスタンスを含む新しいドメインが作成されます。サーバ インスタンスは管理サーバとして動作し、Pet Store サンプル アプリケーションを実行するようにコンフィグレーションされます。このサーバは、通常の WebLogic Server インストールで作成される `petstoreServer` と同様です。

BEA 製品をインストールする場合、コンフィグレーション ウィザードではその他のテンプレートも提供されます。

各テンプレート カテゴリの中で、コンフィグレーション ウィザードは、新しいドメインに含まれるサーバ インスタンスの数とタイプが違う異なるドメイン コンフィグレーションを提供します。提供されるコンフィグレーションは以下のとおりです。

- [Single Server (Standalone Server)] - このオプションを選択すると、1 つのサーバで構成される新しいドメインが作成されます。このサーバは、管理サーバとして動作すると同時に、アプリケーションをホストする機能も果たします。

  このタイプのコンフィグレーションは、プロダクション環境ではなく、開発またはテストを目的とする場合にお勧めします。1 つのサーバで構成されるドメインを作成する手順については、2-17 ページの「1 つのサーバ インスタンスで構成されるドメインの作成」を参照してください。

- [Admin Server with Managed Server(s)] - このオプションを選択すると、1 つの管理サーバと 1 つまたは複数の管理対象サーバで構成される新しいドメインが作成されます。

  このタイプのコンフィグレーションは、専用の管理サーバとアプリケーションをホストする管理対象サーバが提供されるのでプロダクション環境にお勧めします。複数のサーバで構成されるドメインを作成する手順については、2-11 ページの「管理サーバとスタンドアロンの管理対象サーバで構成されるドメインの作成」を参照してください。

- [Admin Server with Clustered Managed Server(s)] - このオプションを選択すると、1 つの管理サーバと複数のクラスタ化された管理対象サーバで構成される新しいドメインが作成されます。

  このタイプのコンフィグレーションは、高度な可用性と信頼性を要するプロダクション環境で使用することをお勧めします。専用の管理サーバと、アプリケーションのフェイルオーバおよびロード バランシングをサポートする管理対象サーバのクラスタが提供されます。複数のサーバをクラスタにまとめたドメインを作成する手順については、2-14 ページの「管理サーバとクラスタ化された管理対象サーバで構成されるドメインの作成」を参照してください。

- [Managed Server (with owning Admin Server configuration)] - このオプションを選択すると、管理サーバが動作するホストとは別の WebLogic Server ホスト マシン上にある管理対象サーバを実行するために必要な起動スクリプトとその他のファイルが作成されます。

  詳細については、2-19 ページの「リモート管理対象サーバのサポートのコンフィグレーション」を参照してください。

サーバ名とリスン アドレスの指定

コンフィグレーション ウィザードでは、作成する各サーバ インスタンスについて名前の割り当てとリスン アドレスの識別が求められます。以下の節では、サーバ名とリスン アドレスの設定に関連する重要な考慮事項を説明します。

サーバ名の考慮事項

WebLogic 環境の各サーバ インスタンスは、それが含まれているドメインまたはクラスタに関係なく、あるいはそれが管理サーバであるか管理対象サーバであるかに関係なく、その名前がユニークである必要があります。

リスン アドレスの考慮事項

次の表に、リスン アドレス値の設定に関連する重要な考慮事項を示します。

リスン アドレスが次のように設定された場合	以下のことが成立する
IP アドレスまたは DNS 名	■ サーバ インスタンスに接続するために、プロセスでは IP アドレスまたは対応する DNS 名を指定できる。 ■ `localhost` を指定するプロセスは接続に失敗する。 ■ `localhost` を使用してサーバ インスタンスに接続する既存のプロセスを更新する必要がある。 ■ リスン アドレスに IP アドレス、リスン ポートにセキュア ポートを指定する接続では、ホスト名の検証を無効にする必要がある。 **注意：** DNS 名を IP アドレスに解決するために、Weblogic Server では適切な DNS サーバにアクセスするか、IP アドレスのマッピングをローカルで取得できなければならない。したがって、リスン アドレスに DNS 名を指定する場合は、WebLogic Server のインスタンスが DNS サーバに接続できるだけの時間ポートを開いておいて、そのマッピングをキャッシュするか、ローカル ファイルで IP アドレスのマッピングを指定する必要がある。リスン アドレスに IP アドレスを指定し、クライアント リクエストで DNS 名を指定すると、WebLogic Server は DNS 名を解決しようとするが、DNS 名のマッピングにアクセスできない場合、そのリクエストは失敗する。
`localhost`	■ プロセスでは `localhost` を指定してサーバ インスタンスに接続する必要がある。 ■ サーバ インスタンスをホストするマシン上のプロセス (ローカル プロセス) のみ、サーバ インスタンスに接続できる。 ■ リモート (非ローカル) プロセスはサーバ インスタンスに接続できない。

リスン アドレスが次のように設定された場合	以下のことが成立する
未定義または空白 ("")	■ プロセスでは IP アドレス、DNS 名、または `localhost` を指定してサーバ インスタンスに接続できる。 ■ `localhost` を指定するプロセスはサーバ インスタンスをホストするマシン上にある必要がある。 ■ サーバ インスタンスが `localhost` としてアクセス可能でなければならず (管理スクリプトが `localhost` に接続するなど)、リモート プロセスからもアクセス可能でなければならない場合は、リスン アドレスを空白にする。 **注意 :** マルチホームの Windows NT マシンで動作する WebLogic サーバでは、リスン アドレスの値を未定義または空白にはしない (マルチホーム マシンは複数の IP アドレスでコンフィグレーションされる)。未定義または空白にすると、マシンの各 IP アドレスについて WebLogic Server はそのポートを確保し、そのポートでリスンする。その結果、他のサーバがマシンの同じポートを使用できなくなる。

コンフィグレーション ウィザードの使用

以下の節では、コンフィグレーション ウィザードを起動および使用する手順を説明します。

コンフィグレーション ウィザードの起動

WebLogic Server のインストール時にカスタム インストールを選択すると、コンフィグレーション ウィザードを自動的に起動するオプションが提示されます。また、以下で説明するように、GUI またはコンソール (コマンドライン) インタフェースのいずれかを使用して、WebLogic Server のインストール後にいつでもコンフィグレーション ウィザードを起動できます。

GUI モードでの起動

GUI モードでコンフィグレーション ウィザードを実行すると、コンフィグレーション ウィザード プログラムがグラフィック環境で実行されます。Windows および一部の UNIX 環境で実行できます。

Windows プラットフォーム上でコンフィグレーション ウィザードを GUI モードで起動するには、Windows の [スタート] メニューで BEA プログラム グループから [コンフィグレーション ウィザードを実行] オプションを選択します。

[スタート | プログラム | BEA WebLogic Platform 7.0 | Domain Configuration Wizard]

UNIX プラットフォーム上で (または Windows コマンド プロンプトから) コンフィグレーション ウィザードを GUI モードで起動するには、次の手順に従います。

1. WebLogic Server ソフトウェアがインストールされている Windows または UNIX システムにログインします。
2. コマンドライン シェルを開きます。
3. `WL_HOME/common/bin` ディレクトリに移動します。

   `WL_HOME` は WebLogic Server のインストール ディレクトリです。たとえば、次のように入力します。

   ```
   cd c:\bea\weblogic700\common\bin
   ```

4. `dmwiz.cmd` または `dmwiz.sh` スクリプトを起動します。

グラフィカル表示がサポートされないシステム上でコンフィグレーション ウィザードを GUI モードで起動しようとした場合、ウィザードは自動的にコンソール モードで起動します。

コンソール モードでの起動

コンソール モードでコンフィグレーション ウィザードを実行すると、コンフィグレーション ウィザード プログラムがテキストベース環境で実行されます。コンフィグレーション ウィザードをコンソール モードで起動するには、次の手順に従います。

1. 対象の Windows または UNIX システムにログインします。
2. コマンドライン シェルを開きます。
3. `WL_HOME/common/bin` ディレクトリに移動します。

   `WL_HOME` は WebLogic Server のインストール ディレクトリです。たとえば、次のように入力します。

   ```
   cd ~/bea/weblogic700/common/bin
   ```

4. `-mode=console` 引数を指定して、`dmwiz.cmd` または `dmwiz.sh` スクリプトを起動します。たとえば、UNIX の `bash` シェルで次のように入力します。

   ```
   . dmwiz.sh -mode=console
   ```

管理サーバとスタンドアロンの管理対象サーバで構成されるドメインの作成

1 つまたは複数の管理対象サーバとスタンドアロンの管理サーバを含む新しいドメインを作成するには、次の手順に従います。

1. 2-10 ページの「コンフィグレーション ウィザードの起動」の手順に従って、コンフィグレーション ウィザードを起動します。以下の手順では、コンフィグレーション ウィザードを GUI モードで実行していることを前提にしています。

   コンフィグレーション ウィザードには [ドメインのタイプと名前を選択] 画面が表示されます。

2. 以下の操作を行います。
   - **[テンプレートを選択してください] :** [WLS Domain]、[WLS Examples]、または [WLS Petstore] を選択してください。詳細については、2-5 ページ

の「コンフィグレーション ウィザードのテンプレートについて」を参照してください。

- **[名前] :** ドメイン名を英数字で入力します。コンフィグレーション ウィザードでは入力された名前を使用して、新しいドメインのドメイン サブディレクトリを作成します。

  このフィールドではスペースを使用できません。また、「Portal」はドメイン名として使用できません。

3. [次へ] ボタンをクリックして [サーバ タイプを選択します] 画面に進みます。
4. [Admin Server with Managed Server(s)] オプションを選択して、[次へ] をクリックします。コンフィグレーション ウィザードには [ドメインの場所を選択] 画面が表示されます。
5. カスタム ドメインを格納する最上位ディレクトリを入力するか、[参照] ボタンを使用してディレクトリを選択します。[次へ] をクリックして [管理サーバで管理対象サーバをコンフィグレーション] 画面に移動します。
6. ドメインに新しい管理対象サーバを追加するには、[追加] をクリックして、[サーバを追加] ダイアログ ボックスのフィールドに以下のように入力します。
   - [サーバ名] - サーバ名を英数字で入力します。このフィールドではスペースを使用できません。

     2-7 ページの「サーバ名の考慮事項」を参照。
   - [リスン アドレス] - このサーバの IP アドレスを入力します。

     2-7 ページの「リスン アドレスの考慮事項」を参照。
   - [リスン ポート] - リスン ポートの数値を入力します。範囲は 1 から 65535 までです。
7. [サーバを追加] ダイアログ ボックスで [追加] をクリックして新しい管理対象サーバを追加してから、[管理サーバで管理対象サーバをコンフィグレーション] 画面に戻ります。

   注意 : 間違えたため、追加したサーバを編集または削除する場合は、サーバ名を選択して [編集] または [削除] ボタンをクリックします。
8. 手順 6 を繰り返して管理対象サーバをさらに追加するか、[次へ] をクリックして [コンフィグレーション Admin Server] 画面に移動します。

9. [コンフィグレーション Admin Server] 画面のフィールドに以下のように入力します。
   - **[サーバ名]:** サーバ名を英数字で入力します。このフィールドではスペースを使用できません。

     2-7 ページの「サーバ名の考慮事項」を参照。
   - **[サーバ リスン アドレス]:** このサーバの IP アドレスを入力します。

     2-7 ページの「リスン アドレスの考慮事項」を参照。
   - **[サーバ リスン ポート]:** リスン ポートの数値を入力します。範囲は 1 から 65535 までです。デフォルト ポートは 7001 です。
   - **[サーバ SSL リスン ポート]:** セキュリティ コンフィグレーションの SSL リスン ポートの数値を入力します。範囲は 1 から 65535 までです。デフォルト ポートは 7002 です。
10. [次へ] をクリックして、[システム ユーザおよびパスワードを作成します] 画面に移動します。
11. コンフィグレーションした管理サーバの起動と管理サーバへの接続に必要なユーザ名とパスワードを入力します。[次へ] をクリックします。
12. Windows システムの場合、管理サーバを Windows サービスとしてインストールするかどうかを尋ねられます。サーバを Windows サービスとして起動する場合は [はい] を、コマンドラインまたは Windows の [スタート] メニューから起動する場合は [いいえ] を選択します。[次へ] をクリックして次に進みます。
13. Windows システムの場合、Windows の [スタート] メニューにドメインをインストールするかどうかを尋ねられます。サーバ起動スクリプトを [スタート] メニューにインストールする場合は [はい] を選択します。[次へ] をクリックして [コンフィグレーションの概要] 画面に移動します。
14. [コンフィグレーションの概要] 画面で、新しいドメインとサーバのコンフィグレーションを確認します。選択内容を編集する場合は、[戻る] ボタンをクリックして訂正画面に戻ります。それ以外の場合は、[作成] をクリックして、指定したサーバを含む新しいドメインを作成します。

管理サーバとクラスタ化された管理対象サーバで構成されるドメインの作成

クラスタとそのメンバーのアドレッシング情報を指定する場合のガイドラインについては、『WebLogic Server クラスタ ユーザーズ ガイド』の「名前とアドレスを識別する」を参照してください。

管理対象サーバのクラスタとスタンドアロンの管理サーバを含む新しいドメインを作成するには、次の手順に従います。

1. 2-10 ページの「コンフィグレーション ウィザードの起動」の手順に従って、コンフィグレーション ウィザードを起動します。以下の手順では、コンフィグレーション ウィザードを GUI モードで実行していることを前提にしています。

   コンフィグレーション ウィザードには [ドメインのタイプと名前を選択] 画面が表示されます。

2. 以下の操作を行います。

   - **[テンプレートを選択してください] :** [WLS Domain]、[WLS Examples]、または [WLS Petstore] を選択してください。詳細については、2-5 ページの「コンフィグレーション ウィザードのテンプレートについて」を参照してください。
   - **[名前] :** ドメイン名を英数字で入力します。コンフィグレーション ウィザードでは入力された名前を使用して、新しいドメインのドメイン サブディレクトリを作成します。

     このフィールドではスペースを使用できません。また、「Portal」はドメイン名として使用できません。

3. [次へ] ボタンをクリックして [サーバ タイプを選択します] 画面に進みます。

4. [Admin Server with Clustered Managed Server(s)] オプションを選択して、[次へ] をクリックします。コンフィグレーション ウィザードには [ドメインの場所を選択します] 画面が表示されます。

5. カスタム ドメインを格納する最上位ディレクトリを入力するか、[参照] ボタンを使用してディレクトリを選択します。[次へ] をクリックして [クラスタ化サーバをコンフィグレーションします] 画面に移動します。

6. ドメインに新しい管理対象サーバを追加するには、[追加] をクリックして、[サーバを追加] ダイアログ ボックスのフィールドに以下のように入力します。

   [サーバ名]: サーバ名を英数字で入力します。このフィールドではスペースを使用できません。

   2-7 ページの「サーバ名の考慮事項」を参照。

   - **[リスン アドレス] :** このサーバの IP アドレスを入力します。

     2-7 ページの「リスン アドレスの考慮事項」を参照。

   - **[リスン ポート] :** リスン ポートの数値を入力します。範囲は 1 から 65535 までです。

7. [サーバを追加] ダイアログ ボックスで [追加] をクリックして新しい管理対象サーバを追加してから、[管理サーバで管理対象サーバをコンフィグレーションします] 画面に戻ります。

   注意 : 間違えたため、追加したサーバを編集または削除する場合は、サーバ名を選択して [編集] または [削除] ボタンをクリックします。

8. 手順 6 を繰り返して管理対象サーバをさらに追加するか、[次へ] をクリックして [クラスタをコンフィグレーションします] 画面に移動します。

9. [クラスタをコンフィグレーションします] 画面で以下の情報を入力します。

   - **[クラスタ名] :** クラスタ名を英数字で入力します。このフィールドではスペースを使用できません。デフォルトでは mycluster です。

     クラスタの名前は、ドメイン内のコンフィグレーション コンポーネント全体でユニークでなければなりません。

   - **[クラスタ マルチキャスト アドレス] :** クラスタのマルチキャスト アドレスを入力します。マルチキャスト アドレスは、範囲が 224.0.0.0 ～ 239.255.255.255 の IP アドレスです。

   - **[クラスタ マルチキャスト ポート] :** マルチキャスト ポートの数値を入力します。値の範囲は 65535 までです。

   - **[クラスタ アドレス] :** このクラスタに接続するためにクライアントが使用するアドレスを入力します。プロダクション環境で使用する場合は、クラスタ内の管理対象サーバの個々の IP アドレスにマップされる DNS 名を入力します。テスト目的または開発目的の場合は、管理対象サーバに割り当てられる IP アドレスとポートのカンマ区切りのリストを使用します (これがデフォルトのエントリです)。

10. [次へ] をクリックして [コンフィグレーション Admin Server (with Cluster)] 画面に移動します。
11. [コンフィグレーション Admin Server (with Cluster)] 画面のフィールドに以下のように入力します。
    - **[サーバ名]:** サーバ名を英数字で入力します。このフィールドではスペースを使用できません。

      2-7 ページの「サーバ名の考慮事項」を参照。
    - **[サーバ リスン アドレス]:** このサーバの IP アドレスを入力します。

      2-7 ページの「リスン アドレスの考慮事項」を参照。
    - **[サーバ リスン ポート]:** リスン ポートの数値を入力します。範囲は 1 から 65535 までです。デフォルト ポートは 7001 です。
    - **[サーバ SSL リスン ポート]:** セキュリティ コンフィグレーションの SSL リスン ポートの数値を入力します。範囲は 1 から 65535 までです。デフォルト ポートは 7002 です。
12. [次へ] をクリックして、[システム ユーザおよびパスワードを作成します] 画面に移動します。
13. コンフィグレーションした管理サーバの起動と管理サーバへの接続に必要なユーザ名とパスワードを入力します。[次へ] をクリックします。
14. Windows システムの場合、管理サーバを Windows サービスとしてインストールするかどうかを尋ねられます。サーバを Windows サービスとして起動する場合は [はい] を、コマンドラインまたは Windows の [スタート] メニューから起動する場合は [いいえ] を選択します。[次へ] をクリックして次に進みます。
15. Windows システムの場合、Windows の [スタート] メニューにドメインをインストールするかどうかを尋ねられます。サーバ起動スクリプトを [スタート] メニューにインストールする場合は [はい] を選択します。[次へ] をクリックして [コンフィグレーションの概要] 画面に移動します。
16. [コンフィグレーションの概要] 画面で、新しいドメインとサーバのコンフィグレーションを確認します。選択内容を編集する場合は、[戻る] ボタンをクリックして訂正画面に戻ります。それ以外の場合は、[作成] をクリックして、指定したサーバを含む新しいドメインを作成します。

1 つのサーバインスタンスで構成されるドメインの作成

管理サーバとしてもアプリケーションのホスト サーバとしても動作する、単一の WebLogic Server インスタンスを含む新しいドメインを作成するには、次の手順に従います。

1. 2-10 ページの「コンフィグレーション ウィザードの起動」の手順に従って、コンフィグレーション ウィザードを起動します。以下の手順では、コンフィグレーション ウィザードを GUI モードで実行していることを前提にしています。

   コンフィグレーション ウィザードには [ドメインのタイプと名前を選択] 画面が表示されます。

2. 以下の操作を行います。

   - **[テンプレートを選択してください] :** [WLS Domain]、[WLS Examples]、または [WLS Petstore] を選択してください。詳細については、2-5 ページの「コンフィグレーション ウィザードのテンプレートについて」を参照してください。
   - **[名前] :** ドメイン名を英数字で入力します。コンフィグレーション ウィザードでは入力された名前を使用して、新しいドメインのドメイン サブディレクトリを作成します。

     このフィールドではスペースを使用できません。また、「Portal」はドメイン名として使用できません。

3. [次へ] ボタンをクリックして [サーバ タイプを選択します] 画面に進みます。

4. [Single Server (Standalone Server)] オプションを選択して、[次へ] をクリックします。コンフィグレーション ウィザードには [ドメインの場所を選択します] 画面が表示されます。

5. カスタム ドメインを格納する最上位ディレクトリを入力するか、[参照] ボタンを使用してディレクトリを選択します。[次へ] をクリックして [コンフィグレーション Single Server] 画面に移動します。

6. [コンフィグレーション Single Server] 画面のフィールドに以下のように入力します。

- **[サーバ名]:** サーバ名を英数字で入力します。このフィールドではスペースを使用できません。

  2-7 ページの「サーバ名の考慮事項」を参照。

- **[サーバ リスン アドレス]:** このサーバの IP アドレスを入力します。

  2-7 ページの「リスン アドレスの考慮事項」を参照。

- **[サーバ リスン ポート]:** リスン ポートの数値を入力します。値の範囲は 1 から 65535 までです。デフォルト ポートは 7001 です。

- **[サーバ SSL リスン ポート]:** SSL リスン ポートの数値を入力します。値の範囲は 1 から 65535 までです。デフォルト ポートは 7002 です。

7. [次へ] をクリックして、[システム ユーザおよびパスワードを作成します] 画面に移動します。
8. コンフィグレーションしたサーバの起動とサーバへの接続に必要なユーザ名とパスワードを入力します。[次へ] をクリックします。
9. Windows システムの場合、管理サーバを Windows サービスとしてインストールするかどうかを尋ねられます。サーバを Windows サービスとして起動する場合は [はい] を、コマンドラインまたは Windows の [スタート] メニューから起動する場合は [いいえ] を選択します。[次へ] をクリックして次に進みます。
10. Windows システムの場合、Windows の [スタート] メニューにドメインをインストールするかどうかを尋ねられます。サーバ起動スクリプトを [スタート] メニューにインストールする場合は [はい] を選択します。[次へ] をクリックして [コンフィグレーションの概要] 画面に移動します。
11. [コンフィグレーションの概要] 画面で、新しいドメインとサーバのコンフィグレーションを確認します。選択内容を編集する場合は、[戻る] ボタンをクリックして訂正画面に戻ります。それ以外の場合は、[作成] をクリックして、指定したスタンドアロン サーバを含む新しいドメインを作成します。

リモート管理対象サーバのサポートのコンフィグレーション

WebLogic ドメインのパフォーマンスと信頼性を向上させるために、WebLogic Server インスタンスを別々のコンピュータ (マシン) で実行できます。たとえば、管理サーバを `MachineA` というコンピュータで、`MS1` という管理対象サーバを `MachineB` で、`MS2` という管理対象サーバを `MachineC` で実行することができます。WebLogic Server インスタンスをマシンで実行するには、以下の操作が必要です。

- WebLogic Server ソフトウェアをインストールする。
- 管理サーバの場合、コンフィグレーション ウィザードを使用して `config.xml` ファイルと起動スクリプトを作成する。
- 管理対象サーバの場合、コンフィグレーション ウィザードを使用して、管理サーバの場所を指定した起動スクリプトを作成する。管理対象サーバは、起動時に自身のコンフィグレーション データを管理サーバに照会します。

  管理サーバがアクセス不能なときに管理対象サーバが起動するようコンフィグレーションする方法については、4-12 ページの「管理サーバにアクセスできない場合の管理対象サーバの起動」を参照してください。

ドメインの管理サーバが動作するホストとは別の WebLogic Server ホストで実行する管理対象サーバごとに、次の手順を実行します。

1. ドメインの `config.xml` ファイルに管理対象サーバのコンフィグレーション データが指定されていることを確認します。

   この情報はドメインの作成時 (2-11 ページの「管理サーバとスタンドアロンの管理対象サーバで構成されるドメインの作成」または 2-14 ページの「管理サーバとクラスタ化された管理対象サーバで構成されるドメインの作成」を参照) でも、ドメインの作成後 (Administration Console オンライン ヘルプの「ドメインへのサーバの追加」を参照) でも指定できます。

2. 新しい管理対象サーバを実行するマシンにログインします。

   コンフィグレーション ウィザードの実行可能ファイルを実行するには、このマシンで WebLogic Server インストール ファイルにアクセスできる必要があります。

3. 2-10 ページの「コンフィグレーション ウィザードの起動」の手順に従って、コンフィグレーション ウィザードを起動します。以下の手順では、コンフィグレーション ウィザードを GUI モードで実行していることを前提にしています。

   コンフィグレーション ウィザードには [ドメインのタイプと名前を選択] 画面が表示されます。

4. 以下の操作を行います。
   - **[テンプレートを選択してください] :** 管理対象サーバを定義したドメインを作成するために使用したテンプレートを選択します。
   - **[名前]:** 管理対象サーバを定義したドメインの名前を入力します。

     [次へ] ボタンをクリックして [サーバ タイプを選択します] 画面に移動します。

5. [Managed Server (with owning Admin Server configuration)] オプションを選択して、[次へ] をクリックします。コンフィグレーション ウィザードには [ドメインの場所を選択] 画面が表示されます。

6. 管理対象サーバの起動スクリプトとデモ セキュリティ ファイルが格納されるディレクトリの名前を入力するか、[参照] ボタンを使用してディレクトリを選択します。[次へ] をクリックして [管理サーバ接続のコンフィグレーション] 画面に移動します。

7. 管理対象サーバが管理サーバに接続するための情報を指定します。
   - **[管理サーバ リスン アドレス]:** 管理サーバをホストするコンピュータの DNS 名または IP アドレスを入力します。
   - **[管理サーバ リスン ポート]:** 管理サーバが非 SSL リクエストをリスンするようにコンフィグレーションされているポート番号を入力します。
   - **[管理サーバ SSL リスン ポート]:** 管理サーバが SSL リクエストをリスンするようにコンフィグレーションされているポート番号を入力します。
   - **[管理対象サーバ名]:** 管理対象サーバの名前 (ドメインの `config.xml` ファイルで指定されている管理対象サーバの名前と一致する必要がある) を入力します。

8. [次へ] をクリックして [スタンドアロン / 管理サーバのコンフィグレーション] ページに移動します。

9. [スタンドアロン / 管理サーバのコンフィグレーション] ページのすべての値を無視します。このページは、リモート管理対象サーバのサポートのコンフィグレーションでは使用しません。
10. [次へ] をクリックして、[システム ユーザおよびパスワードを作成します] 画面に移動します。
11. コンフィグレーションしたサーバの起動とサーバへの接続に必要なユーザ名とパスワードを入力します。[次へ] をクリックします。
12. Windows システムの場合、選択したテンプレートに応じて、新しいサーバを Windows サービスとしてインストールするかどうかを尋ねられます。サーバを Windows サービスとして起動する場合は [はい] を、コマンドラインまたは Windows の [スタート] メニューから起動する場合は [いいえ] を選択します。[次へ] をクリックして次に進みます。
13. Windows システムの場合、選択したテンプレートに応じて、ドメインを Windows の [スタート] メニューにインストールするかどうかを尋ねられます。サーバ起動スクリプトを [スタート] メニューにインストールする場合は [はい] を選択します。[次へ] をクリックして [コンフィグレーションの概要] 画面に移動します。
14. [コンフィグレーションの概要] 画面で、新しいドメインとサーバのコンフィグレーションを確認します。選択内容を編集する場合は、[戻る] ボタンをクリックして訂正画面に戻ります。それ以外の場合は、[作成] をクリックして起動スクリプトを作成します。

管理対象サーバ インスタンスを起動するには、次の手順に従います。

1. ウィザードの [**管理サーバ名または IP**] フィールドで指定 (「リモート管理対象サーバのサポートのコンフィグレーション」 の手順 7.) した WebLogic Server ホスト上のドメインの管理サーバを起動します。
2. リモート ホストにログインし、次のスクリプトを起動します。

   domain-name/startManagedWebLogic.cmd (Windows)
   domain-name/startManagedWebLogic.sh (UNIX)

   domainName は「リモート管理対象サーバのサポートのコンフィグレーション」 の手順 6. で指定したディレクトリです。

3 ネットワーク リソースのコンフィグレーション

以下の節では、ドメイン内の WebLogic Server ネットワーク リソースのコンフィグレーション方法について説明します。

ネットワーク コンフィグレーションの概要

BEA WebLogic Server 7.0 以降では、ドメインで複数のネットワーク インタフェース カード (NIC) や複数のポート番号を使用して、パフォーマンスを向上させたり、一般的なネットワークの問題を解決したりできます。それらの機能で以下のことが可能となります。

- ドメイン内で、管理トラフィックをアプリケーション トラフィックから分離する。
- 1 つの WebLogic Server インスタンスで複数の NIC を使用することにより、ネットワークのスループットを向上させる。
- 特定の WebLogic Server で使用するために、特定の NIC を指定したり、1 つの NIC 上に複数のポート番号を指定したりする。
- ドメイン内で、外部のクライアントベース トラフィックを、内部のサーバベース トラフィックから物理的に分離する。
- ドメイン内で、サーバが他のサーバに接続するために使用するネットワーク接続の優先順位を指定する。

この章では、Administration Console を使用してドメインのネットワーク設定をコンフィグレーションする方法について説明します。ネットワーク設定は、各節で説明する特定の MBean を使用してコンフィグレーションすることもできます。

WebLogic Server 7.0 の新しいネットワーク コンフィグレーション機能

WebLogic Server 7.0 より前のバージョンでは、WebLogic Server のインスタンスはただ 1 つの NIC からのみ接続を受け入れることができ、特定のサーバ インスタンスが使用する TCP ポート番号は制限されていました。次の表では、以前の TCP ポートの制限と、WebLogic Server 7.0 で利用できるネットワーク コンフィグレーション機能を比較しています。

表 3-1 ネットワーク機能の比較

バージョン 6.x の制限	バージョン 7.0 の機能
各 WebLogic Server インスタンスは 1 つの IP アドレスでリスンしていた。	WebLogic Server インスタンスは複数の IP アドレスでリスンできる。

バージョン 6.x の制限	バージョン 7.0 の機能
サーバは最大 3 つの個別のポート番号を使用できる。 ■ 非セキュア HTTP、IIOP、および T3 トラフィック用に予約された標準ポート (デフォルトは 7001) ■ HTTPS、IIOPS、および T3S トラフィック用に予約されたセキュア ポート (デフォルトは 7002) ■ 管理トラフィックを分離するために使用されるオプションのポート	複数のネットワーク チャネルを作成して割り当てることにより、サーバに複数のポート番号を割り当てられる。 詳細については、3-11 ページの「ネットワーク チャネル」を参照。
Administration Console トラフィックは、デフォルトのネットワーク コンフィグレーションで使用可能な任意のポートで発生する可能性があった。	管理トラフィック用に独立した SSL ポートをコンフィグレーションして有効にすると、ドメイン内のすべての WebLogic Server インスタンスは、すべての Administration Console ネットワーク トラフィックでその管理ポートを使用する必要がある。
使用可能なポート番号間で、異なるサービス品質レベルを混在させることはできなかった。たとえば、すべてのセキュア トラフィックは同じポート (デフォルトは 7002) に制限されていたため、あるポートで HTTPS トラフィックをサポートし、別のポートで IIOPS トラフィックをサポートすることはできなかった。	プロトコル サポートは、ネットワーク チャネルを使用して TCP ポートごとに調整できる。 詳細については、3-11 ページの「ネットワーク チャネル」を参照。
ログイン タイムアウトやバックログのコンフィグレーションなど、多くのネットワーク コンフィグレーション フィールドは、ポートではなくサーバ自体 (サーバのリスン アドレス) に適用された。それらのコンフィグレーション設定をポートごとに変えることはできなかった。	コンフィグレーションしたネットワーク チャネルのそれぞれが、TCP ポートに対して異なるプロトコル コンフィグレーションを使用できる。また、チャネルを使用するサーバでは、ネットワーク アクセス ポイントを使ってさまざまなプロトコル設定をオーバライドできる。 詳細については、3-11 ページの「ネットワーク チャネル」および 3-13 ページの「ネットワーク アクセス ポイント (NAP)」を参照。

バージョン 6.x の制限	バージョン 7.0 の機能
クラスタでは、マルチキャスト ポート番号が、各サーバのリスン ポート設定からコピーされた。クラスタのすべてのメンバーは同じマルチキャスト アドレスとポート番号を使用する必要があるため、ポート番号がコピーされる場合、クラスタ内のすべてのサーバで同じリスン ポートを使用しなければならなかった。	クラスタのマルチキャスト コンフィグレーションは、個々のサーバのネットワーク コンフィグレーションに拘束されなくなった。代わりに、クラスタ メンバーが使用するポート番号とは独立して、クラスタ マルチキャスト ポート番号をコンフィグレーションする。 クラスタ化された各サーバがマルチキャスト通信に使用する NIC を指定することもできる。

デフォルト ネットワーク コンフィグレーションについて

WebLogic Server 7.0 のネットワーク コンフィグレーション機能を利用すると、クライアントとサーバおよびサーバ間のネットワーク トラフィックで使用されるネットワーク接続の特性をこれまで以上に管理できます。実際の必要条件に基づいて、ネットワーク チャネルおよびネットワーク アクセス ポイントをコンフィグレーションしてパフォーマンスを管理することができます。

ネットワーク コンフィグレーション機能を利用するには、ドメインをコンフィグレーションする必要があります。チャネルおよび NAP をコンフィグレーションしない場合、ドメインのネットワーク コンフィグレーションは、前のバージョンの WebLogic Server でサポートされていたコンフィグレーション (3-2 ページの表 3-1「ネットワーク機能の比較」の「バージョン 6.x の制限」列で要約されている) と同じようになります。WebLogic Server 7.0 の新しいネットワーク コンフィグレーション機能を利用しないこのシンプルなコンフィグレーションは、デフォルト ネットワーク コンフィグレーションと呼びます。デフォルト ネットワーク コンフィグレーションは、サーバ インスタンスごとに、単一リスン アドレス、HTTP 通信用に 1 ポート (デフォルトは 7001)、および HTTPS 通信用に 1 ポート (デフォルトは 7002) を有効にします。リスン アドレスおよびポートの割り当ては、Administration Console の [コンフィグレーション | 一般] タブを使用してコンフィグレーションできます。割り当てた値は、前のバージョンの WebLogic Server と同じように `ServerMBean` および `SSLMBean` の属性に格納されます。

デフォルトのコンフィグレーションは、以下の場合に使用できます。

- 単純なネットワーク要件を備えたテスト環境にインストールしている場合。
- 1 つの NIC だけを使用し、デフォルト ポート番号によって、ドメイン内のネットワーク トラフィックを分割するのに十分な柔軟性が提供される場合。

デフォルト コンフィグレーションを使用すると、サードパーティの管理ツールと新しいインストールとの互換性が保証されます。ネットワーク コンフィグレーションは、この場合も `ServerMBean` および `SSLMBean` に格納されるためです。

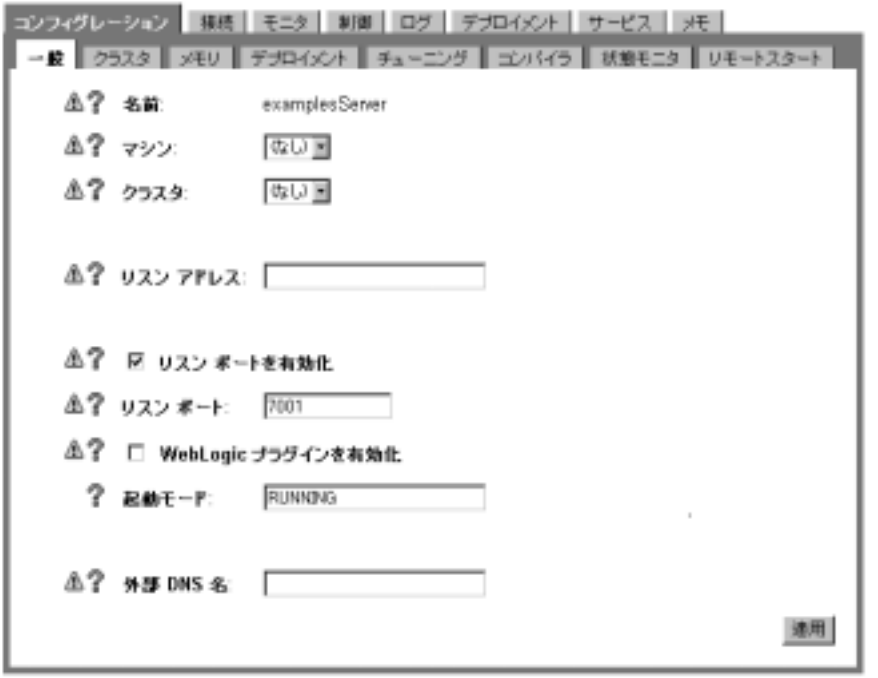
実行するコンフィグレーションに関係なく、デフォルト ネットワーク コンフィグレーションに関連付けられた設定はそのまま `ServerMBean` および `SSLMBean` に格納され、サーバ インスタンスへの接続を提供する必要がある場合に使用されます。

デフォルト ネットワーク コンフィグレーションの調整

サーバ インスタンスのデフォルトのリスン アドレス、リスン ポート、およびネットワーク設定は、そのインスタンスの `ServerMBean` で定義されます。以下の手順で、デフォルトのリスン アドレス、リスン ポート、およびリスン ポート接続のプロパティを変更できます。

1. まだ動作していない場合は、ドメインの管理サーバを起動します。
2. ドメインの Administration Console にログインします。
3. 左ペインで [サーバ] ノードを選択して、ドメインでコンフィグレーションされているサーバ インスタンスを表示します。
4. 左ペインで、コンフィグレーションするサーバ インスタンスの名前を選択します。

5. 右ペインで [コンフィグレーション | 一般] タブをクリックして、サーバ インスタンスの現在のデフォルト ネットワーク設定を表示します。



6. デフォルトのリスン アドレスとポートの値を以下のように入力します。
   - [リスン アドレス] - サーバ インスタンスが受信する接続をリスンするために使用するデフォルトの IP アドレスまたは DNS 名を入力します。

注意： サーバ インスタンスのリスン アドレスを localhost と指定すると、非ローカルのプロセスではそのサーバ インスタンスに接続できなくなります。サーバ インスタンスのホスト マシン上のプロセスのみ、そのサーバ インスタンスに接続できます。サーバ インスタンスが localhost としてアクセス可能でなければならず (管理スクリプトが localhost に接続するなど)、リモート プロセスからもアクセス可能でなければならない場合は、リ

スン アドレスを空白にします。リスン アドレスを空白にすると、サーバ インスタンスはマシンのアドレスを判別してそのアドレスでリスンします。

サーバ インスタンスがマルチホーム マシン上にある場合は、[リスン アドレス] フィールドは空白にしないで、localhost アドレスを指定します。サーバが localhost アドレスを使用する場合、[リスン ポート] と [SSL リスン ポート] は、マルチホーム マシン上で使用可能なすべての IP アドレスにバインドされます。

- [リスン ポート] - サーバが受信する接続をリスンするために使用するデフォルトの TCP ポートを入力します。

7. [適用] をクリックして変更を反映させます。

8. 右ペインで [接続 | プロトコル] タブをクリックして、サーバのデフォルトの接続プロパティを定義します。

9. このページでデフォルトの接続属性を編集し、[適用] をクリックして変更を適用します。個別の属性の詳細については、Administration Console オンライン ヘルプの「[サーバ] --> [接続] --> [プロトコル]」を参照してください。

10. 新しいデフォルト ネットワーク コンフィグレーションを使用するには、サーバを再起動します。

サーバ起動時のデフォルト コンフィグレーションの表示

3-4 ページの「デフォルト ネットワーク コンフィグレーションについて」で説明されているように、WebLogic Server 7.0 の新しいネットワーク コンフィグレーション機能をコンフィグレーションしなければ、サーバ インスタンスは前のバージョンの WebLogic Server で適用された制限に準拠するシンプルな ネットワーク コンフィグレーションを使用します。前のバージョンの WebLogic Server と同じように、この基本的なネットワーク コンフィグレーション情報は `ServerMBean` および `SSLMBean` を使用して格納されます。

WebLogic Server 7.0 の新しいネットワーク コンフィグレーション機能の 1 つは、ネットワーク チャネルをコンフィグレーションする機能です。ネットワーク チャネルを使用すると、サーバ インスタンスに複数のポート番号を割り当てることができます。ネットワーク チャネルとその機能については、3-11 ページの「ネットワーク チャネル」を参照してください。

WebLogic Server が起動すると、`ServerMBean` で定義されたリスン アドレスおよびポートの属性を使用して「デフォルト」のネットワーク チャネルが自動的に生成されます。

サーバのデフォルト ネットワーク コンフィグレーションは、そのログ ファイルで次の行の後に表示されます。

```
Network Channel:Default
```

コンフィグレーションは、ログ ファイルで次のように表示されます。

```
####<Apr 22, 2002 10:49:36 AM PDT> <Info> <RJVM> <myhostname>
<examplesServer> <main> <kernel identity> <> <000520> <Network
Configuration
Cluster Participant:false
Native Socket IO Enabled:true
Reverse DNS Allowed:false
Network Channel:Default
Listen Address:          not configured
Listen Port:             7001
SSL Listen Port:         7002
External DNS Name:       not configured
Cluster Address:         not configured
Protocol(s):             T3,T3S,HTTP,HTTPS,IIOP,IIOPS,COM
Tunneling Enabled:false
Outgoing Enabled:true
Admin Traffic Only:false
Admin Traffic OK:true
```

```
Channel Weight:          50
Accept Backlog:          50
Login Timeout:           5000 ms
Login Timeout SSL:       25000 ms
Message Timeout HTTP:    60000 ms
Message Timeout T3:      60000 ms
Message Timeout COM:     60000 ms
Message Timeout IIOP:    60000 ms
Idle Timeout IIOP:       60000 ms
Max Message Size HTTP:   10000000
Max Message Size T3:     10000000
Max Message Size COM:    10000000
Max Message Size IIOP:   10000000
>
```

ログ ファイルの後半では、サーバがバインドする実際のリスン アドレスとポートが表示されます。

```
####<Apr 22, 2002 10:58:52 AM PDT> <Notice> <WebLogicServer>
<myhost> <examplesServer> <SSLListenThread.Default> <kernel
identity> <> <000354> <Thread "SSLListenThread.Default" listening
on port 7002>
```

```
####<Apr 22, 2002 10:58:52 AM PDT> <Info> <WebLogicServer> <myhost>
<examplesServer> <ListenThread.Default> <kernel identity> <>
<000213> <Adding address: myhost/192.168.1.11 to licensed client
list>
```

ネットワーク チャネルと NAP の使用

WebLogic Server 7.0 では、ネットワーク チャネルとネットワーク アクセス ポイント (NAP) という 2 つの新しいコンフィグレーション可能リソースが導入されています。このリソースは Administration Console または WebLogic Server MBean を使用して管理できます。これらのリソースは `NetworkChannelMBean` と `NetworkAccessPointMBean` の 2 つの MBean に格納されます。

ある特定の状況では、`ServerMBean` のリスン アドレスおよびポート番号属性も使用されます。次に例を示します。

- NAP を使用してリスン アドレスを定義しない場合は、`ServerMBean` で定義されたリスン アドレスが使用される。
- ドメイン全体の管理ポートを定義するが、何らかの理由で管理対象サーバがそのコンフィグレーションされた管理ポートにバインドできない場合は、`ServerMBean` または `SSLMBean` のデフォルト リスン ポートが使用される。

以下の節では、ネットワーク チャネルおよびネットワーク アクセス ポイントで提供される機能を説明します。

ネットワーク チャネル

ネットワーク チャネルを利用すると、1 つまたは複数の WebLogic Server インスタンスで使用する追加のポート番号とプロトコル設定をコンフィグレーションできます。

これらのポート番号は、`ServerMBean` および `SSLMBean` に関連付けられているデフォルトのポート番号とは別のものです (3-4 ページの「デフォルト ネットワーク コンフィグレーションについて」を参照)。

ネットワーク チャネルでは、以下のような WebLogic Server へのネットワーク接続の基本的な属性が定義されます。

- 接続でサポートされるプロトコル (HTTP、HTTPS、T3、T3S、COM)
- セキュア通信および非セキュア通信で使用するデフォルトのリスン ポート
- ログイン タイムアウト値や最大メッセージ サイズなど、接続のデフォルト プロパティ
- 接続でトンネリングをサポートするかどうか
- 接続は、ドメイン内の他の WebLogic Server インスタンスとの通信に使用できるか、クライアントとの通信にのみ使用するか

Administration Console で個別のエンティティとしてネットワーク チャネルをコンフィグレーションし、ドメイン内のサーバに 1 つまたは複数のチャネルを割り当てます。チャネルを割り当てられたサーバ インスタンスでは、デフォルト コンフィグレーションではなく、そのチャネルと関連付けられたポート番号とプロトコル コンフィグレーションを使用します。

送信接続のコンフィグレーション

接続属性を定義するほかに、チャネルでは受信するクライアント トラフィックとドメイン内部のサーバ間トラフィックを分離することもできます。

各チャネル定義では、チャネルで送信接続をサポートするかどうかを指定します (デフォルトではサポートされます)。送信接続をサポートするチャネルとサポートしないチャネルの 2 つのチャネルをサーバ インスタンスに割り当てると、クライアント接続用とサーバ接続用のネットワーク トラフィックを別々にコンフィグレーションできます。チャネルと複数の NAP を組み合わせることにより (3-13 ページの「ネットワーク アクセス ポイント (NAP)」を参照)、異なる IP アドレスまたはポート番号上で、クライアント トラフィックとサーバ トラフィックを物理的に分離できます。

チャネルでは、発信ネットワーク トラフィックに使用される接続の優先順位を指定することもできます。サーバ インスタンスに複数の発信可能なチャネルが割り当てられている場合、重みの値によって各チャネルの優先順位を指定できます。サーバ インスタンスが送信接続を開始すると、重みの値が大きいチャネルに割り当てられている NAP が、重みの値が小さいチャネルの NAP よりも先に使用されます。この機能を利用すると、高速の NIC を使用する発信チャネルと NAP の組み合わせに最高の重みを割り当てることによって、すべてのサーバ間トラフィックで保証されたレベルのスループットを実現できます。

注意 : ネットワーク チャネルの重みは、リモート EJB 参照や JNDI を介して検索されるリソースなどのリモート参照用の内部接続にのみ適用されます。チャネルの重みは、URL を介して直接開始される接続には使用されません。

一般的な WebLogic Server のチャネル

チャネルを使用すると、ネットワーク コンフィグレーションのさまざまな目標を実現できます。ほとんどの WebLogic Server では、以下の一般的なチャネルの 1 つまたは複数が使用されます。

- デフォルト チャネル - WebLogic Server ではデフォルト チャネルを自動的に生成して、`ServerMBean` に関連付けられたリスン アドレスおよびリスン ポート設定を記述します。3-9 ページの「サーバ起動時のデフォルト コンフィグレーションの表示」で説明したように、デフォルト チャネル コンフィグレーションは起動時に参照できます。
- 管理チャネル - オプションの管理ポートを定義して、ドメイン内でアプリケーション トラフィックから管理トラフィックを分離するために使用できます。管理ポートを有効にした場合、WebLogic Server はポート設定に基づいて管理チャネルを自動的に生成します。詳細については、3-15 ページの「ドメイン全体の管理ポートのコンフィグレーション」を参照してください。

- カスタム チャネル - カスタムチャネルは、WebLogic Server が自動的に生成するチャネルではなく、ドメイン内にユーザが定義および適用するチャネルです。ドメインでのカスタム チャネルの使用例については、3-14 ページの「ネットワーク チャネルと NAP の一般的な使い方」を参照してください。

ネットワーク アクセス ポイント (NAP)

ネットワーク アクセス ポイント (NAP) とは、サーバのネットワーク チャネルで使用されるポート番号、プロトコル コンフィグレーション、および IP アドレス (省略可能) を割り当てるためにコンフィグレーションできるリソースです。NAP は常にネットワーク チャネルと一緒に使用され、サーバ インスタンスの各チャネルには NAP を 1 つだけ割り当てることができます。

NAP を使用して、特定の WebLogic Server インスタンスのネットワーク チャネル属性をオーバライドしたり、サーバのネットワーク チャネルを特定の IP アドレスまたは NIC に関連付けたりできます。

NAP はオプションです。ネットワーク チャネルを使用するようにサーバ インスタンスをコンフィグレーションし、NAP をコンフィグレーションしない場合、サーバはチャネルに関連付けられているネットワーク コンフィグレーションを使用します。チャネル自体はリスン アドレスを指定しないため、新しい接続はサーバのデフォルト ネットワーク コンフィグレーションに関連付けられているリスン アドレス (`ServerMBean` で定義されるリスン アドレス) を使用します。

サーバのネットワーク チャネルとともに NAP を指定した場合、チャネルは NAP で指定されたリスン アドレス (定義されている場合) を使用して、新しいネットワーク接続を生成します。NAP では、基底のネットワーク チャネルの特定のプロトコルおよびネットワーク コンフィグレーション設定をオーバーライドして、このサーバの接続を調整することもできます。3-34 ページの「ネットワーク チャネルと NAP の属性」では、ネットワーク チャネルと NAP の属性を示し、NAP がオーバーライドできるチャネル属性について説明しています。

警告 : マルチホーム マシン上で動作するサーバでネットワーク チャネルを使用する場合は、`ServerMBean` またはサーバに関連付けられた NAP に、有効なリスン アドレスを入力する必要があります。NAP および `ServerMBean` のリスン アドレスが空白の場合は、localhost アドレス (IP アドレス 0.0.0.0 または 127.*.*.*) を指定すると、サーバはネットワーク チャネルのリスン ポートと SSL リスン ポートを、マルチホーム マシン

で使用可能なすべての IP アドレスにバインドします。ServerMBean のリスン アドレスの設定については、3-4 ページの「デフォルト ネットワーク コンフィグレーションについて」を参照してください。

ネットワーク チャネルと NAP の一般的な使い方

ドメインで管理ポートをコンフィグレーションした場合、WebLogic Server は自動的に生成されるチャネルを使用します。詳細については、3-15 ページの「ドメイン全体の管理ポートのコンフィグレーション」を参照してください。

複数のサーバで 1 つのカスタム チャネルを使用すると、ドメインのネットワーク コンフィグレーションを簡略化できます。チャネルのコンフィグレーションを変更すると、そのチャネルを使用するすべてのサーバの接続属性が自動的に変更されます。チャネルをコンフィグレーションして適用する手順については、3-19 ページの「カスタム チャネルを使用したドメイン管理の簡略化」を参照してください。

また、1 つのサーバに対して複数のチャネルを作成して割り当てることができます。複数のチャネルを使うと、プロトコルごと、リスン ポートごと、またはその他のチャネル コンフィグレーション プロパティごとに、ネットワーク トラフィックを分けることができます。たとえば、1 つのサーバで 2 つのチャネルを使えば、セキュアなトラフィック用と非セキュアなトラフィック用に、デフォルトの接続プロパティを調整できます。複数のチャネルを使用すると、外部のクライアント トラフィックを、内部のサーバ間トラフィックから分離することもできます。複数のチャネルを使用する簡単な例については、3-27 ページの「ネットワーク トラフィックのポート番号による分割」を参照してください。チャネルを使用して送信接続をコンフィグレーションする例については、3-29 ページの「内部および外部ネットワーク トラフィックの分離」を参照してください。

関連付けられたネットワーク アクセス ポイントをコンフィグレーションすると、サーバ単位で多くのチャネル プロパティをオーバライドすることもできます。NAP の使用例については、3-21 ページの「複数の NIC を単一のサーバにコンフィグレーションするための NAP の使用」を参照してください。

ドメイン全体の管理ポートのコンフィグレーション

WebLogic Server 7.0 では、ドメイン内の管理対象サーバの管理ポートを有効にできます。管理ポートは、管理対象サーバがドメインの管理サーバと通信する目的にのみ使用するポートです。管理ポートはオプションですが、以下の 2 つの重要な機能を提供します。

- サーバをスタンバイ状態で起動できる。スタンバイ状態では、サーバの他のネットワーク接続でクライアント接続を受け入れることはできませんが、管理ポートを使用して管理対象サーバをアクティブ化または管理することはできます。スタンバイ状態の詳細については、『管理者ガイド』の「WebLogic Server の起動と停止」を参照してください。
- ドメイン内で管理トラフィックとアプリケーション トラフィックを分離できる。プロダクション環境では、2 つのトラフィックを分離すると、同じネットワーク接続上に大量のアプリケーション トラフィックがある状態で重要な管理操作 (サーバの起動と停止、サーバのコンフィグレーションの変更、およびアプリケーションのデプロイ) を行う、ということがなくなります。

WebLogic Server は、管理対象サーバの起動時に管理チャネルを生成することにより、管理ポートを実装します。

管理ポートの制限事項

管理ポートはセキュアな SSL トラフィックのみを受け付け、管理ポート経由の接続はすべて認証を必要とします。これらの機能のため、管理ポートを有効にした場合は、ドメインで以下の制限があります。

- ドメイン内の管理サーバとすべての管理対象サーバは、SSL プロトコルをサポートするようにコンフィグレーションする必要がある。SSL をサポートしない管理対象サーバは、起動時に管理サーバに接続できません。そのようなサーバをコンフィグレーションするためには、管理ポートを無効にする必要があります。
- ドメイン内のすべてのサーバ インスタンスは、管理ポートを同時に有効または無効にする必要があるため、管理ポートはドメイン レベルでコンフィグ

レーションする。個々の管理対象サーバの管理ポート番号は変更できますが、個々の管理対象サーバの管理ポートを有効または無効にすることはできません。

- 管理ポートを有効にしたら、ドメイン内の管理対象サーバを起動するには、管理サーバへの SSL 接続を確立する必要がある。管理対象サーバをコマンドラインで手動で起動する場合、またはノード マネージャを使用して起動する場合のいずれにも、この制限が適用されます。SSL 接続を確立するには、次の起動引数を使用します。

  ```
  -Dweblogic.management.server=https://host:admin_port
  ```

  注意： 以前のリリースのようにホスト名とポートの組み合わせだけを指定することはできません。HTTPS プロトコルが必要です。

  ```
  -Dweblogic.security.SSL.trustedCAKeyStore=path_to_keystore
  ```

  ```
  -Dweblogic.security.SSL.ignoreHostnameVerification=true
  ```

- 管理ポートを有効にしたら、Administration Console のトラフィックは管理ポートを介して接続しなければならない (WebLogic Server 7.0 より前のリリースでは、Administration Console トラフィックは使用可能な任意のサーバ ポートを使って接続できました)。

- ドメイン全体の管理ポートを使用するドメインの同じコンピュータ上で複数のサーバ インスタンスが動作する場合は、以下のいずれかを行う必要があります。
  - マルチホーム マシンでサーバ インスタンスをホストし、各サーバ インスタンスにユニークなリスン アドレスを割り当てる。
  - マシン上の 1 つのサーバ インスタンスを除くすべてのサーバ インスタンスでドメイン全体のポートを無効にする。ポートを無効にするには、Administration Console の [サーバ｜接続｜ SSL ポート] ページの [Advanced Attributes] パートの [Local Administration Port Override] オプションを使用する。

管理ポートのコンフィグレーションと起動

管理ポートを有効にする前に、以下のことを行う必要があります。

- Administration Console の [接続｜ SSL ポート] タブを使用して、ドメイン内の各サーバ インスタンス (管理サーバとすべての管理対象サーバ) で SSL プ

ロトコルを有効にする。『WebLogic Security の管理』の「SSL プロトコルのコンフィグレーション」の手順に従って、SSL セキュリティ コンポーネントを取得し、管理サーバで SSL を使用するようにコンフィグレーションするか、または、新しいサーバにデフォルトでインストールされるデモの証明書ファイルを使用します。

- ドメイン内の各サーバに、コンフィグレーション済みのデフォルト リスン ポートまたはデフォルト SSL リスン ポートがあることを確認する (詳細については、3-5 ページの「デフォルト ネットワーク コンフィグレーションの調整」を参照)。デフォルト ポートは、コンフィグレーションされた管理ポートにサーバがバインドできない場合に必要となります。追加のデフォルト ポートが使用できる場合、サーバは起動を続行します。その後で管理ポートを適切な値に変更できます。

上記の前提条件を満たしたら、次の手順に従って管理ポートを有効にします。

1. まだ起動していない場合は、ドメインの管理サーバを起動します。
2. Administration Console にログインします。
3. 左ペインでドメインの名前をクリックして、ドメインのコンフィグレーション プロパティを表示します。
4. 右ペインで [コンフィグレーション | 一般] タブをクリックします。
5. 右ペインの [ドメイン管理ポートを有効化 (SSL をコンフィグレーションしてください)] チェックボックスを選択します。
6. [ドメイン管理ポート] フィールドに値を入力して、すべてのサーバが使用するデフォルトの管理ポートを指定します。デフォルトでは、ドメイン全体の管理ポートは 9002 に設定されます。
7. [適用] をクリックして、ドメインに変更を適用します。
8. ドメイン内のすべてのサーバで同じ管理ポートを使用する場合は、手順 13 に進みます。
9. ドメイン内の個々のサーバが使用する管理ポートを変更するには、左ペインで [サーバ] ノードを選択して、コンフィグレーションするサーバの名前を選択します。
10. 右ペインで [接続 | SSL ポート] タブをクリックして、サーバの現在の SSL コンフィグレーションを表示します。

11. [ローカル管理ポートのオーバーライド (0: オーバーライドなし)] 属性フィールドに値を入力して、このサーバが Administration Console との通信に使用する管理ポートを指定します。値が 0 の場合、サーバは手順 6 で指定したドメイン全体の管理ポートの値を使用します。

    注意： 管理サーバの管理ポートには、よく知られた未使用のポート番号を選択してください。ドメインのすべての管理対象サーバは、ドメイン内で起動するために、このポートを指定する必要があります。

12. [適用] をクリックして変更を反映させます。

13. 新しい管理ポートを使用するには、管理サーバとすべての管理対象サーバを再起動します。

    ドメインの管理対象サーバを起動する場合、コマンドラインで (またはサーバの起動スクリプトで) 以下のオプションを指定して、管理サーバの管理ポートに接続する必要があります。

    ```
    -Dweblogic.management.server=https://host:admin_port
    -Dweblogic.security.SSL.trustedCAKeystore=path_to_keystore
    -Dweblogic.security.SSL.ignoreHostnameVerification=true
    ```

サーバ起動時の管理チャネルの表示

サーバのデフォルト ネットワーク コンフィグレーションで管理ポートを有効にした場合、サーバは ServerMBean のリスン アドレス設定と SSLMBean の SSL コンフィグレーション設定を使用して、サーバで使用するための「管理チャネル」を生成します。管理チャネルの設定は、非セキュア リスン ポート設定がないという点以外は、デフォルト チャネルの設定と似ています。以下のログファイルの抜粋では、デフォルト管理ポートの設定が示されています。

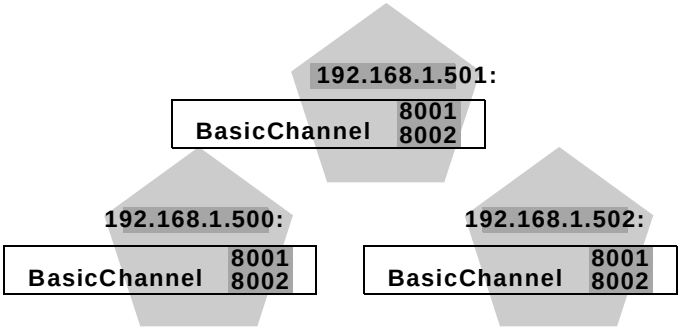
```
Network Channel:          Administrator
Listen Address:           172.17.10.55
Listen Port:none
SSL Listen Port:          9002
External DNS Name:        not configured
Cluster Address:          not configured
Protocol(s):              T3S,HTTPS
Tunneling Enabled:false
Outgoing Enabled:true
Admin Traffic Only:true
Admin Traffic OK:true
Channel Weight:           50
Accept Backlog:           50
Login Timeout:            5000 ms
```

```
Login Timeout SSL:        25000 ms
Message Timeout HTTP:     60000 ms
Message Timeout T3:       60000 ms
Max Message Size HTTP:    10000000
Max Message Size T3:      10000000
>

...

####<Apr 22, 2002 3:14:34 PM PDT> <Notice> <WebLogicServer>
<myhost> <adminserver> <SSLListenThread.Administrator> <kernel
identity> <> <000355> <Thread "SSLListenThread.Administrator"
listening on port 9002, ip address 192.168.1.11>
```

カスタム チャネルを使用したドメイン管理の簡略化

ネットワーク チャネルは、WebLogic Server 接続プロパティのテンプレートとして機能してドメインのネットワーク管理を簡略化できます。たとえば、以下の図では、1 つのカスタム チャネルを利用して、すべてのサーバのすべてのネットワーク トラフィックの接続特性を定義するドメインが示されています。

図 3-1 カスタム ネットワーク チャネルの使用



この例では、BasicChannel というネットワーク チャネルが以下のようにコンフィグレーションされています。

- ListenPort はデフォルトの 8001 に設定されている。

- SSLListenPort はデフォルトの 8002 に設定されている。
- すべてのネットワーク プロトコルはこのチャネルで有効になっている。

管理者は、ドメイン内の各サーバ インスタンスに BasicChannel を追加しています。

このサンプル ドメインでは NAP が使用されません。代わりに、各サーバのリスン アドレスは、各サーバの `ServerMBean` の ListenAddress 属性から取得されます。チャネルではセキュアおよび非セキュア トラフィック用にデフォルト ポート番号を指定しています。サーバの `ServerMBean` 属性で指定されているポート以外に、新しいポートが開かれます。

ドメインのアプリケーションのテスト段階で、管理者はこのチャネルでサポートされるすべてのプロトコルに対してデフォルト値を設定できます。ドメインがさらにベータ版のテスト段階に進んだら、管理者は BasicChannel だけを変更することにより、接続タイムアウトと最大メッセージ サイズを必要に応じて微調整できます。

カスタム ネットワーク チャネルのコンフィグレーション

ネットワーク チャネルをコンフィグレーションするには、3-34 ページの「ネットワーク チャネルと NAP の属性」を参照して、次の手順に従います。

1. コンフィグレーションするサーバが含まれるドメインの Administration Console を起動します。
2. Administration Console の左ペインで、[ネットワーク チャネル] ノードを選択します。
3. 右ペインで [新しい Network Channel のコンフィグレーション] をクリックします。
4. 新しいネットワーク チャネルの属性値を入力し、[作成] をクリックして新しいチャネルの定義を作成します。

   注意： WebLogic Server では、内部のチャネル名で `.WLDefaultChannel` および `.WLDefaultAdminChannel` を使用し、チャネル名のプレフィックスとして `.WL` を予約しています。`.WL` で始まるカスタム チャネルを作成することはできません。

5. [コンフィグレーション | チューニング] タブを選択して、新しいチャネルのバックログとタイムアウトの属性を変更します。[適用] をクリックして、このタブの変更を適用します。
6. [コンフィグレーション | プロトコル] タブを選択して、新しいチャネルのプロトコル サポートを有効化、無効化、またはコンフィグレーションします。[適用] をクリックして、このタブの変更を適用します。
7. 新しいネットワーク チャネルのプロパティをコンフィグレーションしたら、[対象 | サーバ] または [対象 | クラスタ] タブを選択して、新しいチャネルを使用するドメイン内のサーバまたはクラスタを選択します。[選択可] カラムでサーバまたはクラスタを選択し、矢印ボタンを使用して、サーバまたはクラスタを [選択済み] カラムに移動します。
8. [適用] をクリックして、選択したサーバまたはクラスタにネットワーク チャネルを割り当てます。
9. 新しいチャネル ポートの定義を使用するには、対象として割り当てたすべてのサーバを再起動する必要があります。

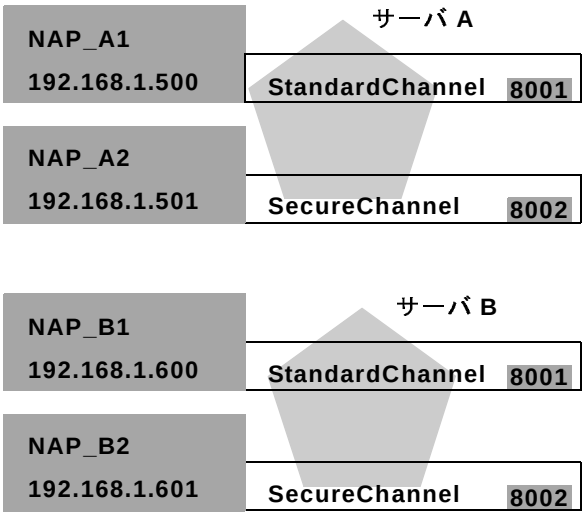
複数の NIC を単一のサーバにコンフィグレーションするための NAP の使用

デフォルトでは、新しくコンフィグレーションした WebLogic Server インスタンスは、`ServerMBean` および `SSLMBean` に関連付けられた単一の `ListenAddress`、`ListenPort`、および `SSLListenPort` 属性のみを使用します。ただし、アプリケーションのパフォーマンスがサーバよりもネットワークに拘束されるドメインでは、1 つの WebLogic Server インスタンスに複数の NIC (またはマルチホーム ハードウェアによって提供される複数の IP アドレス) をコンフィグレーションすることが望ましい場合があります。

1 つのサーバで複数の NIC またはマルチホーム ハードウェアを利用するには、複数のネットワーク チャネルおよび関連する複数の NAP を設定して、デフォルトの `ServerMBean` コンフィグレーションをオーバライドする必要があります。このようなコンフィグレーションでは、サーバで使用する個々の NIC を識別することによって、NAP が受信ネットワーク トラフィックを分割します。チャネルは各 NAP に基準となるネットワーク コンフィグレーションを提供します。

たとえば、次の図のサンプル ドメインでは、各サーバ インスタンスが 2 つの別々の NIC でリスンしています。各サーバ インスタンスでは、一方の NIC が標準のネットワーク トラフィック用に予約され、もう一方の NIC がセキュア トラフィック用に使用されています。

図 3-2　ネットワーク アクセス ポイントの使用



このコンフィグレーションを作成するために、管理者は最初に 2 つのチャネル (StandardChannel と SecureChannel) をコンフィグレーションして、両方のチャネルを両方のサーバ インスタンスに割り当てました。セキュア トラフィックと非セキュア トラフィックを分割するために、2 つのチャネルは以下のようにコンフィグレーションされました。

- StandardChannel では HTTP および T3 プロトコルが有効に、HTTPS および T3S が無効になっている。このチャネルでは、ListenPort および SSListenPort の値である 8001 および 8002 を使用しますが、SSL ポートは使用されません。
- SecureChannel では HTTPS および T3S プロトコルが有効に、HTTP および T3 が無効になっている。このチャネルでは、ListenPort および SSListenPort の値である 8001 および 8002 を使用しますが、標準の ListenPort は使用されません。

ドメイン内の NIC ごとに、4 つの NAP がコンフィグレーションされました。WebLogic Server A では、NAP_A1 がリスン アドレス 192.168.1.500 でコンフィグレーションされて、StandardChannel に割り当てられました。次に、リスン アドレス 192.168.1.501 の NAP_A2 が作成されて、SecureChannel に割り当てられました。

WebLogic Server B では、IP アドレス 192.168.1.600 と 192.168.1.601 を使用して、同じように NAP_B1 と NAP_B2 をコンフィグレーションしました。

両方のサーバで、管理者は、利用可能なチャネルに関連付けられている `ListenPort` および `SSLListenPort` のデフォルト値をそのまま使用しました。ただし、ユニークなポート番号が関連付けられた NAP で定義されている場合、ポート番号は NIC ごとに異なる可能性がありました。

ネットワーク アクセス ポイントのコンフィグレーション

3-20 ページの「カスタム ネットワーク チャネルのコンフィグレーション」の手順に従って、ネットワーク チャネルをコンフィグレーションしてそれをサーバに割り当てている場合にのみ、サーバのネットワーク アクセス ポイントをコンフィグレーションできます。

以下の手順に従って、サーバのネットワーク アクセス ポイントをコンフィグレーションします。個々の属性の詳細については、3-37 ページの「NAP の属性」を参照してください。

1. まだ動作していない場合は、ドメインの Administration Console を起動します。
2. 左ペインで、[サーバ] ノードを選択してから、コンフィグレーションするサーバ名を選択します。
3. 右ペインで、[コンフィグレーション | チューニング] タブを選択して、サーバのデフォルト ネットワーク コンフィグレーションに対する現在のバックログおよびタイムアウト設定を表示します。
4. 右ペインの下部にある [チャネルの詳細チューニングのコンフィグレーション] をクリックして、サーバに現在割り当てられているすべてのネットワーク チャネルが示されたテーブルを表示します。

5. ネットワーク アクセス ポイントを作成するネットワーク チャネルの名前をクリックします。これにより [コンフィグレーション] タブが表示されます。このタブを使用すると、選択したチャネルの属性をオーバライドできます。
6. リスン アドレスやリスン ポートなどのチャネル属性をオーバライドするには、[コンフィグレーション | 一般] タブを選択してから、[適用] をクリックします。
7. バックログおよびタイムアウト設定のチャネル属性をオーバライドするには、[コンフィグレーション | チューニング] タブを選択してから、[適用] をクリックします。
8. サポートされるネットワーク プロトコルのチャネル コンフィグレーションをオーバライドするには、[コンフィグレーション | プロトコル] タブを選択してから、[適用] をクリックします。

注意： ネットワーク アクセス ポイントでネットワーク プロトコルを有効または無効にすることはできません。ネットワーク チャネルでのみプロトコル サポートを有効または無効にできます。

クラスタにおけるネットワーク チャネルのコンフィグレーション

WebLogic Server クラスタで 1 つまたは複数のカスタム チャネルを使用するには、以下の節で説明するガイドラインに従います。

WebLogic Server クラスタでカスタム チャネルを使用しない場合は、『WebLogic Server クラスタ ユーザーズ ガイド』の「WebLogic クラスタの設定」の手順に従って、クラスタを設定します。

管理対象サーバの作成

Administration Console を使用して、クラスタに参加するすべての管理対象サーバを作成します。3-5 ページの「デフォルト ネットワーク コンフィグレーションの調整」の説明に従って、各管理対象サーバのリスン アドレスとリスン ポートをコンフィグレーションします。

クラスタ内の管理対象サーバでは、サーバのデフォルト チャネルを生成するために、デフォルトのリスン アドレスとリスン ポートが必要です。クラスタに追加するカスタム チャネルは、デフォルト チャネルとは別に使用され、そのプロパティはクラスタのすべてのメンバーに適用されます。

クラスタの作成

Administration Console オンライン ヘルプで説明されているように、Administration Console を使用して、ドメインに新しいクラスタを作成します。クラスタを作成する場合は、以下のようにします。

- 未使用のマルチキャスト アドレスを選択して、新しいクラスタの [コンフィグレーション | マルチキャスト] タブにそのアドレスを入力する。

  クラスタ内の管理対象サーバがマルチキャスト通信をどのように使用するのか、およびマルチキャスト アドレスを設定するガイドラインについては、『WebLogic Server クラスタ ユーザーズ ガイド』の「IP マルチキャストを使用した 1 対多通信」を参照してください。

- 後でクラスタにネットワーク チャネルを適用する場合は、新しいクラスタの [コンフィグレーション | 一般] タブにクラスタ アドレスを入力する必要はない。ネットワーク チャネルには、アドレスを指定するための [クラスタ アドレス] 属性があります。

- [サーバ] タブを使用すると、管理対象サーバを新しいクラスタに追加できる。プロダクション環境では、管理サーバをクラスタに参加させないでください。

ネットワーク チャネルの作成および割り当て

3-20 ページの「カスタム ネットワーク チャネルのコンフィグレーション」の手順に従って、クラスタで使用する新しいネットワーク チャネルを作成します。新しいチャネルを作成する場合は、以下のようにします。

- 新しいチャネルのリスン ポートおよび SSL リスン ポートで、クラスタ化されたサーバのデフォルト ポートと同じ値を使用していないことを確認する。ネットワーク チャネルで管理対象サーバのデフォルト ポートと同じポートが指定される場合は、カスタム ネットワーク チャネルと管理対象サーバの

デフォルト チャネルがそれぞれ同じポートにバインドしようとするので、管理対象サーバを起動できなくなります。

- 新しいチャネルの [コンフィグレーション | 一般] タブで [クラスタ アドレス] の値を入力する。クラスタ アドレスには、クラスタ内のすべての管理対象サーバを識別する、外部に公開されたアドレス (たとえば、クラスタに参加するすべての IP アドレスを解決する DNS 名) を指定してください。ネットワーク チャネルはこの値を使用して、クラスタで使用される EJB ハンドルとフェイルオーバ アドレスを生成します。

  ネットワーク チャネルでクラスタ アドレスを指定しない場合、クラスタはクラスタの定義で指定された [クラスタ アドレス] の値を使用します (使用可能な場合)。

- 新しいチャネルを作成したら、チャネルの [対象 | クラスタ] タブを使用して、3-25 ページの「クラスタの作成」で作成したクラスタを選択する。

各サーバ インスタンスのマルチキャスト アドレスの定義

オプションで、クラスタ内の各サーバがマルチキャスト トラフィックに使用する NIC を指定できます。サーバ インスタンスのマルチキャスト アドレスを指定しない場合は、3-25 ページの「クラスタの作成」でクラスタに定義したマルチキャスト アドレスが使用されます。

次の手順に従います。

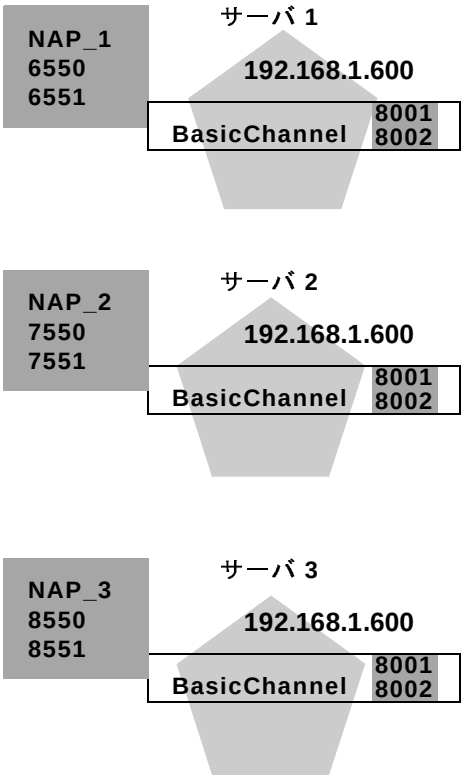
1. まだ動作していない場合は、ドメインの Administration Console を起動します。
2. 左ペインで、[サーバ] ノードを選択してから、コンフィグレーションするサーバ名を選択します。
3. 右ペインで、[コンフィグレーション | クラスタ] タブを選択します。
4. [インターフェイス アドレス] 属性フィールドに、サーバがマルチキャスト トラフィックに使用する NIC の IP アドレスを入力します。
5. [適用] をクリックして変更を反映させます。

ネットワーク トラフィックのポート番号による分割

NAP とチャネルを使用すると、ネットワーク トラフィックをポート番号ごとに分割できます。たとえば、ドメイン内の WebLogic Server がネットワークに拘束されない場合は、サーバで 1 つの NIC を使用して、複数のポート番号をコンフィグレーションすると、ドメイン内の物理的に複数のサーバに接続を振り分けることができます。NAP を使用すると、`ServerMBean` で使用可能な 3 つの標準ポート番号の他にも、ポート番号をコンフィグレーションできます。

以下のサンプル ドメインでは、管理者は複数の NAP をコンフィグレーションして、3 つの異なるサーバ インスタンスで 1 つの NIC を利用しています。

図 3-3　ネットワーク トラフィックの分割



この例では、ネットワーク接続を複数のサーバに分割するために 1 つの NIC が使用されています。BasicChannel は複数の NAP と組み合わせて使用され、ユニークなポート番号で接続を処理しています。

サンプル ドメインのすべてのサーバは、デフォルトの NIC IP アドレス 192.168.1.600 を使用しています。ただし、各サーバは NAP を使用して、同じ IP アドレスの異なるポート番号上でリスンしています。BasicChannel のコンフィグレーションではポート番号 8001 および 8002 が使用されていますが、各サーバの NAP がそれらのポート割り当てをオーバライドしています。

ロード バランサと組み合わせて、同様のネットワーク コンフィグレーションを使用し、クラスタ内の複数の WebLogic Server インスタンスに接続要求を分散することもできます。

内部および外部ネットワーク トラフィックの分離

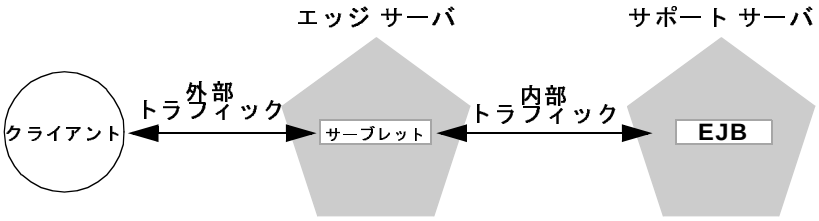
ネットワーク チャネルと NAP の特別な用法として、クライアント指向の外部ネットワーク トラフィックとサーバ指向の内部トラフィックの分離があります。特定のファイアウォール コンフィグレーションにおいて内部と外部のトラフィックを分離したり、サーバとクラスタに異なるレベルのスループットを保証したりすることができます。

エッジ サーバのコンフィグレーション

「エッジ」サーバとは、外部のクライアントが Web アプリケーションへのアクセスを開始するために使用する WebLogic Server インスタンスです。Web アプリケーションのコンポーネントはドメイン内の他の WebLogic Server に配置できますが、クライアントのアクセスはエッジ サーバに制限されます。

たとえば、ドメイン内の単一の WebLogic Server にサーブレットまたは JSP をデプロイして、そのサーバの IP アドレスを外部クライアントから使用できるようにします。サーブレットまたは JSP はドメイン内の 2 番目のサーバと対話して、EJB または他のサービスを取得できます。ただし、EJB をホストする WebLogic Server は外部クライアントから利用できません。代わりに、クライアントはエッジ サーバ経由でのみ EJB と対話します。以下の図では、単純なエッジ サーバのコンフィグレーションを示します。

図 3-4　エッジ サーバ

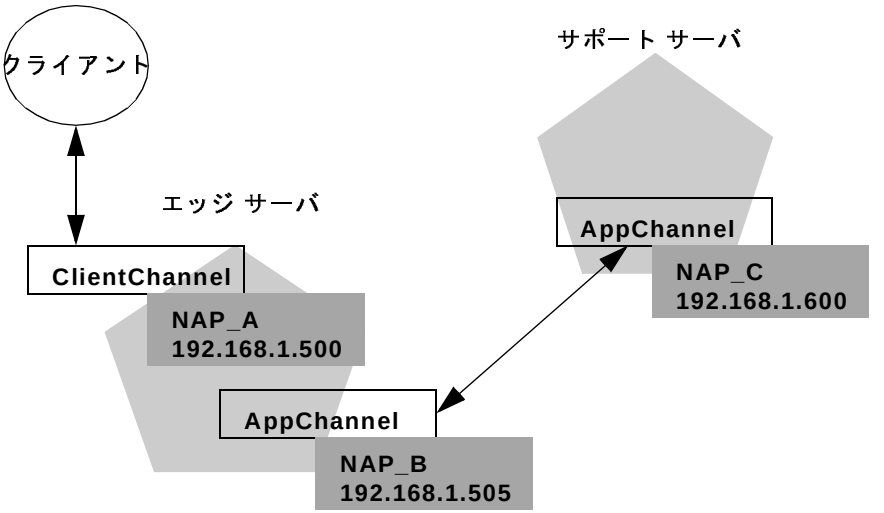


注意： WebLogic Server ドメインには、エッジ サーバをいくつでも含めることができます。エッジ サーバは、スタンドアロンの管理対象サーバでも、クラスタとしてコンフィグレーションしてもかまいません。エッジ サーバのクラスタのコンフィグレーションについては、『WebLogic Server クラスタ ユーザーズ ガイド』の「推奨多層アーキテクチャ」を参照してください。

エッジ サーバのコンフィグレーションで、ネットワーク チャネルを使用すると、外部のクライアントベースのネットワーク接続と内部のサーバベースの接続を分離できます。ネットワーク トラフィックの分離は、論理的に行う (1 つの NIC でポート番号ごとに分離する) ことも、物理的に行う (複数の NIC と IP アドレスによって分離する) こともできます。

次の図に、内部トラフィック用に 1 つの NIC を使用し、外部ネットワーク トラフィック用にもう 1 つの NIC を使用する単純なエッジ サーバのコンフィグレーションを示します。

図 3-5 単純なエッジ サーバのコンフィグレーション



この例では、2 つのサーバが同じチャネル AppChannel を使用して互いに通信します。サーバ間の相互通信を提供するために、AppChannel の `OutgoingEnabled` 属性が有効になっています。

このドメインのエッジ サーバは別のチャネル ClientChannel も使用して、Web アプリケーション クライアントからのネットワーク接続を処理しています。ClientChannel の OutgoingEnabled 属性は無効になっています。つまり、エッジ サーバでは強制的に AppChannel とそれに関連付けられた NIC を使用して、サポート サーバへの接続を開始します。

このコンフィグレーションでネットワーク トラフィックを追跡すると以下のようになります。

1. クライアントは、公開された IP アドレス 192.168.1.500 からエッジ サーバへ接続します。この IP アドレスは、エッジ サーバに関連付けられている NAP で定義されたものです。NAP は ClientChannel の接続プロパティを使用しています。
2. クライアントはエッジ サーバ上のサーブレットにアクセスします。次に、サポート サーバに存在する EJB がルックアップされます。

3. ネットワーク接続を開始して EJB にアクセスするために、エッジ サーバでは送信接続をサポートするチャネルが必要です。このコンフィグレーションでは、AppChannel のみが送信接続をサポートしているため、サポート サーバとの通信に使用されます。
4. AppChannel には、エッジ サーバに関連付けされた NAP があるため、サポート サーバとの通信には、この NAP の IP アドレス 192.168.1.505 が使用されます。これによって、クライアントが Web アプリケーションへのアクセスに使用した接続から、内部のネットワーク接続が分離されます。

WebLogic Server のドメイン レベルのネットワーク コンフィグレーションでは、ハードウェアまたはソフトウェアのファイアウォールを組み合わせて、サポート サーバへのアクセスをブロックすることができます。

送信接続の優先順位の指定

WebLogic Server に送信接続を開始できる複数のチャネルがある場合、サーバは他のサーバへの接続時に、使用するチャネルを選択する必要があります。まず、WebLogic Server は接続に必要なプロトコルに基づいてチャネルを選択します。複数のチャネルで同じプロトコルをサポートしている場合は、各チャネルに異なる重みを割り当てて、チャネルの優先順位を指定できます。

チャネルの重みは `NetworkChannelMBean` に適用される単純な数値です。送信接続の開始に使用できる同じサービス レベルのチャネルが複数ある場合にだけ、チャネルの重みが考慮されます (高いサービス レベルを持つチャネルが現在アクティブな場合は、チャネルの重みに関係なく、そのチャネルが使用されます)。送信接続用の NAP を選択する場合、重みの値が高い方のチャネルが、重みの値の低いチャネルよりも優先されます。

マルチホーム システムでは、チャネルの重みを利用すると、使用可能なネットワーク カードの能力に基づいて、同等のチャネルに優先順位を指定できます。

注意： `ServerMBean` および `SSLMBean` の値から派生するデフォルト チャネルと管理チャネルは、常に送信接続に対応すると見なされ、デフォルトの重み 50 を使用します。

チャネルの障害の処理

WebLogic Server では通常、重みの高いチャネルを重みの低いチャネルより優先して使用しますが、ネットワーク障害が発生すると、選択したチャネルが使用できなくなる可能性があります。障害の発生に対処するために、WebLogic Server では以下のようなアルゴリズムを使用して、送信用のチャネルを選択します。

1. 最初に、要求されるサービス品質を持つ、最も重みの高いチャネルを試してみます。
2. 最も重みの高いチャネルを使用して接続できない場合、要求されるサービス品質を持つ、2 番目に重みが高いチャネルで試行します。
3. 接続要求が再び失敗した場合、サーバはさらに低い重みのチャネルを使用して、すべてのチャネルが試されるまで、接続を試行し続けます。
4. 利用できるすべてのチャネルを使用しても接続できなかった場合、呼び出し側ユーザにエラー メッセージが返されます。

上記の手順では、必要なサービス品質レベルのチャネルがすべて使用された場合にだけ、ユーザは接続エラー メッセージを受け取ります。すべてのチャネルの組み合わせが使い果たされて、別のユーザが送信接続を開始しようとした場合 (または障害の後で接続が再試行される場合)、WebLogic Server では、最も重みの高いチャネルから、上記のアルゴリズムを再び開始します。

RMI のサービス品質レベルについて

RMI ルックアップの場合にだけ、WebLogic Server は送信接続のサービス レベルをアップグレードできます。たとえば、RMI ルックアップを実行するために T3 接続が必要なときに、既存のチャネルで T3S しかサポートしていない場合、ルックアップは T3S チャネルを使用して実行されます。

このアップグレードの動作は URL を使用するサーバ要求には適用されません。URL 自体にプロトコルが組み込まれているためです。たとえば、サーバでは、`https://` のみをサポートするチャネルから、`http://` で始まる URL 要求を送信することはできません。

ネットワーク チャネルと NAP の属性

以下の節では、NetworkChannelMBean および NetworkAccessPointMBean で使用可能な属性を示し、対応するチャネルの属性をオーバライドできる NAP 属性について説明します。

チャネルの属性

WebLogic Server ドメインでは、チャネルは `NetworkChannelMBean` として表現されます。この MBean は以下の表に示すコンフィグレーション属性で構成されています。Administration Console を使用して、チャネルに関連付けられた `NetworkAccessPointMBean` で補完的な値を指定すると、強調表示された属性の値はオーバライドされます。チャネル プロパティのオーバライドの詳細については、3-37 ページの「NAP の属性」を参照してください。

表 3-2 ネットワーク チャネルのコンフィグレーション属性

MBean の属性名	デフォルト値	指定できる値	説明
Name*	なし	文字列	コンソールでこのチャネルを識別するための名前。WebLogic Server では、内部のチャネル名で `.WLDefaultChannel` および `.WLDefaultAdminChannel` を使用し、チャネル名のプレフィックスとして `.WL` を予約している。`.WL` で始まるカスタム チャネルを作成することはできない。
Description	なし	文字列	このチャネルに関するテキストのメモ (省略可能)。
ChannelWeight*	50	1-100	送信接続用のチャネルを選択する場合に、このチャネルに割り当てる相対的な重み。詳細については、3-32 ページの「送信接続の優先順位の指定」を参照。

MBean の属性名	デフォルト値	指定できる値	説明
ListenPortEnabled	False	True、False	このチャネルでプレーン テキストのリスン ポートを提供するかどうかを指定する。
ListenPort*	8001	1-65534	非セキュア ネットワーク プロトコルで使用するデフォルトのリスン ポート。
SSLListenPortEnabled	False	True、False	このチャネルで SSL リスン ポートを提供するかどうかを指定する。
SSLListenPort*	8002	1-65534	セキュア ネットワーク プロトコルで使用するデフォルトのリスン ポート。
COMEnabled	False	True、False	このチャネルで COM トラフィック用のプレーン テキスト (非 SSL) ポートを提供するかどうかを指定する。
ClusterAddress	なし	文字列	クラスタで使用される EJB ハンドルとフェイルオーバ アドレスを生成するために、このチャネルが使用するアドレス。
TunnelingEnabled*	False	True、False	このネットワーク チャネルがトンネリングをサポートするかどうかを指定する。ネットワーク チャネルでトンネリングを有効にするには、以下のように属性を設定して、チャネルの HTTP および T3 プロトコルも有効にする必要がある。 ■ HTTP(S) Enabled=true ■ T3(S) Enabled=true T3 はトンネリングの有効化に必要だが、T3 プロトコルはトンネリング接続を使用するクライアントからは隠される。トンネリングの一般的な情報については、『管理者ガイド』の「HTTP トンネリングのための WebLogic Server の設定」を参照。
T3 Enabled*	False	True、False	プロトコルを有効または無効にする。
T3S Enabled*	False	True、False	プロトコルを有効または無効にする。

MBean の属性名	デフォルト値	指定できる値	説明
HTTP Enabled*	False	True、False	プロトコルを有効または無効にする。
HTTPS Enabled*	False	True、False	プロトコルを有効または無効にする。
OutgoingEnabled*	True	True、False	このチャネルが、ドメイン内の他の WebLogic Server への発信接続を開始できるかどうかを指定する。
LoginTimeoutMillis	5000	0 (無効)、1-100000	WebLogic Server が接続のタイムアウトまで待機するミリ秒数を設定する。
LoginTimeoutMillisSSL	25000	0 (無効)、1-2147483647	WebLogic Server が SSL 接続のタイムアウトまで待機するミリ秒数を設定する。
AcceptBacklog*	50	0-2147483647	バックログで使用可能な接続数を設定する。処理される接続数を増やすには、この値を大きくする。
TunnelingClientPingSecs*	45	0-2147483647	サーバがクライアントの ping を実行するまでに待機する秒数を設定する。
TunnelingClientTimeoutSecs*	40	0 (無効) 1-2147483647	サーバがタイムアウトまでに待機する秒数を設定する。
MaxT3MessageSize	10000000	4096-2000000000	T3 メッセージの最大サイズをバイト単位で指定する。
MaxHTTPMessageSize	10000000	4096-2000000000	HTTP メッセージの最大サイズをバイト単位で指定する。
MaxCOMMessageSize	10000000	4096-2000000000	COM メッセージの最大サイズをバイト単位で指定する。
CompleteT3MEssageTimeout	60	0 (無効)、1-480	T3 メッセージの受信の待機中に、システムがタイムアウトするまでの秒数。
CompleteHTTPMessageTimeout	60	0 (無効)、1-480	HTTP メッセージの受信の待機中に、システムがタイムアウトするまでの秒数。

MBean の属性名	デフォルト値	指定できる値	説明
CompleteCOMMessageTimeout	60	0 (無効)、1-480	COM メッセージの受信の待機中に、システムがタイムアウトするまでの秒数。

* 属性が動的にコンフィグレーションできないことを示します (変更を有効にするには、サーバの再起動が必要です)。

NAP の属性

WebLogic Server ドメインでは、NAP は `NetworkAccessPointMBean` として表現されます。この MBean は以下の表に示すコンフィグレーション属性で構成されています。3-34 ページの「ネットワーク チャネルと NAP の属性」で説明したように、この MBean 属性の多くは、`NetworkChannelMBean` に対応する属性があります。前述のように、チャネルに割り当てるときに、NAP はチャネルの属性値をオーバライドできます。

表 3-3 ネットワーク アクセス ポイント (NAP) のコンフィグレーション属性

MBean の属性名	デフォルト値	指定できる値	説明
ListenAddress*	Null	文字列のホスト名と IP アドレス	この NAP が受信接続をリスンするために使用するデフォルトの IP アドレスまたは DNS 名。この属性が null の場合、NAP では、関連するサーバの `ServerMBean` で指定された `ListenAddress` 属性の値を使用する (`ServerMBean` で値が指定されていない場合、WebLogic Server は localhost アドレスをリスンする)。 **注意 :** DNS 名を IP アドレスに解決するために、Weblogic Server では適切な DNS サーバにアクセスするか、IP アドレスのマッピングをローカルで取得できなければならない。したがって、リスン アドレスに DNS 名を指定する場合は、WebLogic Server のインスタンスが DNS サーバに接続できるだけの時間ポートを開いておいて、そのマッピングをキャッシュするか、ローカル ファイルで IP アドレスのマッピングを指定する必要がある。リスン アドレスに IP アドレスを指定し、クライアント リクエストで DNS 名を指定すると、WebLogic Server は DNS 名を解決しようとするが、DNS 名のマッピングにアクセスできない場合、そのリクエストは失敗する。
ListenPort*	-1	-1 1-65534	-1 の場合、NAP では、関連付けられている `NetworkChannelMBean` で定義された値を使用する。 それ以外の値を設定すると、関連付けられている `NetworkChannelMBean` で定義された値をオーバライドする。 詳細については、`ListenPort` チャネル属性の説明を参照。

MBean の属性名	デフォルト値	指定できる値	説明
SSLListenPort*	-1	-1 1-65534	-1 の場合、NAP では、関連付けられている NetworkChannelMBean で定義された値を使用する。 それ以外の値を設定すると、関連付けられている NetworkChannelMBean で定義された値をオーバライドする。 詳細については、SSLListenPort チャネル属性の説明を参照。
LoginTimeoutMillis	-1	-1 0 1-100000	-1 の場合、NAP では、NAP の NetworkChannelMBean で定義された値を使用する。 それ以外の値を設定すると、NAP の NetworkChannelMBean で定義された属性値をオーバライドする。0 を指定すると、タイムアウトは無効になる。 この属性の詳細については、3-34 ページの「ネットワーク チャネルと NAP の属性」を参照。
LoginTimeoutMillisSSL	-1	-1 0 1- 2147483647	-1 の場合、NAP では、NAP の NetworkChannelMBean で定義された値を使用する。 それ以外の値を設定すると、NAP の NetworkChannelMBean で定義された属性値をオーバライドする。0 を指定すると、タイムアウトは無効になる。 この属性の詳細については、3-34 ページの「ネットワーク チャネルと NAP の属性」を参照。

MBean の属性名	デフォルト値	指定できる値	説明
AcceptBacklog*	-1	-1 0 1-2147483647	-1 の場合、NAP では、NAP の NetworkChannelMBean で定義された値を使用する。 それ以外の値を設定すると、NAP の NetworkChannelMBean で定義された属性値をオーバライドする。0 を指定すると、バックログは無効になる。 この属性の詳細については、3-34 ページの「ネットワーク チャネルと NAP の属性」を参照。
TunnelingClient PingSecs*	-1	-1 0 1-2147483647	-1 の場合、NAP では、NAP の NetworkChannelMBean で定義された値を使用する。 それ以外の値を設定すると、NAP の NetworkChannelMBean で定義された属性値をオーバライドする。0 を指定すると、クライアントの ping は無効になる。 この属性の詳細については、3-34 ページの「ネットワーク チャネルと NAP の属性」を参照。
TunnelingClient TimeoutSecs*	-1	-1 0 1-2147483647	-1 の場合、NAP では、NAP の NetworkChannelMBean で定義された値を使用する。 それ以外の値を設定すると、NAP の NetworkChannelMBean で定義された属性値をオーバライドする。0 を指定すると、タイムアウトは無効になる。 この属性の詳細については、3-34 ページの「ネットワーク チャネルと NAP の属性」を参照。

MBean の属性名	デフォルト値	指定できる値	説明
CompleteT3 MessageTimeout	-1	-1 0 1-480	-1 の場合、NAP では、NAP の NetworkChannelMBean で定義された値を使用する。 それ以外の値を設定すると、NAP の NetworkChannelMBean で定義された属性値をオーバライドする。 この属性の詳細については、3-34 ページの「ネットワーク チャネルと NAP の属性」を参照。
CompleteHTTP MessageTimeout	-1	-1 0 1-480	-1 の場合、NAP では、NAP の NetworkChannelMBean で定義された値を使用する。 それ以外の値を設定すると、NAP の NetworkChannelMBean で定義された属性値をオーバライドする。0 を指定すると、タイムアウトは無効になる。 この属性の詳細については、3-34 ページの「ネットワーク チャネルと NAP の属性」を参照。
CompleteCOM MessageTimeout	-1	-1 0 1-480	-1 の場合、NAP では、NAP の NetworkChannelMBean で定義された値を使用する。 それ以外の値を設定すると、NAP の NetworkChannelMBean で定義された属性値をオーバライドする。0 を指定すると、タイムアウトは無効になる。 この属性の詳細については、3-34 ページの「ネットワーク チャネルと NAP の属性」を参照。

* 属性が動的にコンフィグレーションできないことを示します (変更を有効にするには、サーバの再起動が必要です)。

4　障害が発生したサーバの回復

ドメイン内の障害が発生した WebLogic Server インスタンスは、いくつかの方法を用いて回復できます。これらすべての方法では、ドメインのコンフィグレーション データとセキュリティ データをバックアップしておく必要があります。

ここでは、以下のタスクについて説明します。

- コンフィグレーション データのバックアップ
- セキュリティ データのバックアップ
- 管理対象サーバの動作中における管理サーバの再起動
- 管理サーバにアクセスできない場合の管理対象サーバの起動
- 自動状態モニタと管理対象サーバの再起動

コンフィグレーション データのバックアップ

管理サーバは、自身のコンフィグレーション ファイルを使用してドメインを管理します。このため、管理サーバが障害によって使用不能になった場合に備えて、アーカイブ コピーを保存しておくことをお勧めします。アーカイブを作成するには、定期バックアップ、フォールト トレラント ディスク、変更時の手動によるファイルのコピーなど、一般的な方法を使用できます。管理サーバに障害が発生した場合は、バックアップ ファイルを新しいマシンにコピーして、そのマシン上で管理サーバを再起動します。

デフォルトでは、管理サーバはドメインのコンフィグレーション データを *domain_name*\config.xml というファイルに格納します。ここで *domain_name* はドメインのルート ディレクトリです。Administration Console、weblogic.Admin コマンド、または JMX API のいずれかを使用してドメイン内

のサーバに対して行ったコンフィグレーションの変更の多くは、config.xml ファイルに保持されます。たとえば、WebLogic Server インスタンスをコンフィグレーションする MBean は、自身のデータを config.xml に保持します。

注意： サーバを起動するときには、異なるコンフィグレーション ファイルを使用してそのサーバをコンフィグレーションできます。詳細については、『管理者ガイド』の「weblogic.Server コマンドの使用」を参照してください。

管理サーバがその起動シーケンスを正常に完了し、要求を処理する準備が整うと、管理サーバは自身のコンフィグレーション ファイルを次のファイルに保存します。

domain_name/config.xml.booted

config.xml と config.xml.booted のバックアップ コピーを作成しておくことをお勧めします。特定のセッション中に行ったコンフィグレーションの変更を元に戻す必要がある場合、config.xml.booted に定義されているコンフィグレーションに戻すことができます。

また、1 つまたは複数の管理対象サーバをコンフィグレーションして、ドメインのコンフィグレーション データの一部をレプリケートすることもできます。この章の 4-16 ページの「ドメインのコンフィグレーション ファイルのレプリケート」では、このレプリケーションの設定について説明します。

セキュリティ データのバックアップ

管理サーバは、ドメインのセキュリティ データを管理する役割を果たします。このため、管理サーバに障害が発生した場合に備えて、そのセキュリティ データをアーカイブしておくことをお勧めします。管理サーバが使用不能になった場合、セキュリティ データを変更することはできません。

この節では、以下のタスクについて説明します。

- セキュリティ コンフィグレーション データのバックアップ
- WebLogic LDAP リポジトリのバックアップ
- SerializedSystemIni.dat およびセキュリティ証明書のバックアップ

セキュリティ コンフィグレーション データのバックアップ

使用するセキュリティ プロバイダごとに、WebLogic Security フレームワークは管理 Bean (MBean) を起動してプロバイダのコンフィグレーションを管理します。コンフィグレーションをサーバ セッション間で保持するために、ドメインは *domain_name*\userConfig\Security というディレクトリに MBean リポジトリを作成します。サーバを起動すると、サーバはこのリポジトリのデータを使用して実行時キャッシュを作成します。管理サーバは、キャッシュ内のデータに基づいて userConfig\Security ディレクトリを定期的に更新します。また、サーバを停止するときにも MBean リポジトリが更新されます。

次のいずれかの方法を使用して、セキュリティ MBean リポジトリ (バイナリ フォーマット) をバックアップできます。

- セキュリティ データに対して重要な変更を行った後に、*domain_name*\userConfig ディレクトリをバックアップする。たとえば、セキュリティ プロバイダの初期コンフィグレーションを完了したら、サーバを停止して userConfig ディレクトリをバックアップします。

  ドメインの管理サーバに障害が発生した場合、userConfig バックアップを新しい管理サーバのルート ディレクトリにコピーします。

- データを XML フォーマットでダンプして、その XML ファイルをバックアップする。管理サーバに障害が発生した場合、この XML ファイルを新しい管理サーバのコンフィグレーション ディレクトリにロードします。

  この方法ではデータを XML にダンプするので、この XML ファイルを使用すればコンフィグレーション エラーをデバッグおよび修正することもできます。

次の節では、XML ファイルを使用してセキュリティ データをバックアップおよびデバッグする方法について説明します。

- セキュリティ コンフィグレーション データの XML ファイルへのダンプ
- ダンプした XML ファイルの修正
- XML データの MBean リポジトリへのロード
- デフォルト セキュリティ データ リポジトリの作成

セキュリティ コンフィグレーション データの XML ファイルへのダンプ

WebLogicMBeanDumper ユーティリティは、*domain_name*\userConfig 内のデータを読み込んで XML ファイルを出力します。このユーティリティは MBean リポジトリ内のデータを使用するので、サーバを起動しなくても WebLogicMBeanDumper を使用できます。

すべてのセキュリティ プロバイダ MBean のデータをダンプするには、次の手順に従います。

1. 管理サーバを停止して、サーバが MBean リポジトリを定期的に更新しないようにします。
2. コマンド シェルを開きます。
3. ドメインのルート ディレクトリに移動します。
4. Java クラスパスを設定するために、次のコマンドを入力します。

   WL_HOME\server\bin\setWLSEnv.cmd (Windows の場合)
   WL_HOME/server/bin/setWLSEnv.sh (UNIX の場合)

   詳細については、『管理者ガイド』の「クラスパスの設定」を参照してください。
5. 次のコマンドを入力します。

```
java weblogic.management.commo.WebLogicMBeanDumper
    -includeDefaults -name Security:* output-file
```

コード リスト 4-1 に、セキュリティ MBean データが格納された出力ファイルの一部を示します。XML フォーマットは次のとおりです。

- 各 <SetMBean> 要素は 1 つのセキュリティ プロバイダ MBean を表します。この要素に含まれる属性は次のとおりです。
  - DisplayName は Administration Console が表示する名前。
  - ObjectName は MBean インスタンスのプログラムに基づく名前。Administration Console から作成するセキュリティ MBean の命名規約は次のとおりです。Security:Name=*securityRealmSecurityProvider*
  - TypeName は MBean タイプの名前。MBean タイプによって、MBean インスタンスの使用可能な属性とデフォルト値が指定されます。詳細につ

いては、『WebLogic Security サービスの開発』の「MBean 定義ファイル（MDF）要素の構文」を参照してください。

- `<Attributes>` サブ要素には、MBean 属性用の名前と値の組み合わせが含まれます。
- `<Defaulted AttributeNames>` サブ要素には、MBean 属性のリストが含まれます。これらの属性の値は、MBean タイプによって指定されたデフォルト値です。

  デフォルト値を確認するには、Administration Console を参照するか、または次のいずれかを行います。

  - 作成したタイプのデフォルトを確認するには、MBean タイプの作成に使用した MDF を参照するか、Administration Console の [セキュリティ | レルム | *yourRealm*] を参照します。
  - WebLogic Server がインストールしたタイプのデフォルト値を確認するには、そのタイプの WebLogic Server Javadoc を参照してください。たとえば、`DefaultAuthenticator` のデフォルト値を確認するには、`weblogic.security.providers.authentication.DefaultAuthenticatorMBean` JavaDoc を参照してください。

コード リスト 4-1 XML フォーマットのセキュリティ MBean インスタンス

```
<?xml version="1.0" encoding="UTF-8"?>

<MBeans>

  <SetMBean
      DisplayName="DefaultRoleMapper"
      ObjectName="Security:Name=myrealmDefaultRoleMapper"
      Type="weblogic.management.security.providers.authorization.DefaultRoleMap
      per"
  >

      <Attributes Realm="Security:Name=myrealm">

            <Defaulted AttributeNames="RoleDeploymentEnabled"/>

      </Attributes>

  </SetMBean>

...
```

```
</MBeans>
```

ダンプした XML ファイルの修正

ダンプした XML ファイルに対しては、次の変更を加えることができます。

- 対応する <SetMBean> 要素を削除することによって、MBean インスタンスをリポジトリから削除する。<SetMBean> タグおよび </SetMBean> タグとその間のすべてのデータを削除する必要があります。
- 上の方法で MBean をリポジトリから削除することによって、MBean をデフォルト コンフィグレーションに回復する。次に、XML ファイルをロードしてサーバを起動したら、Administration Console を使用して新しいインスタンスを作成します。
- 既存の MBean インスタンスをコピーし、DisplayName、ObjectName などの値を変更することによって、新しい MBean インスタンスを作成する。Type は既存の MBean タイプにする必要があります。このマニュアルでは、すべての ObjectName が次の既存の命名規約に従っていることを前提にしています。Security:Name=*unique-name*.
- <Attributes> 要素内の名前と値の組み合わせのひとつの値を置き換える。

次の節で説明するように修正済み XML ファイルをロードする場合、WebLogicMBeanLoader ユーティリティはこのファイル内のデータに基づいてリポジトリを再生成します。

XML データの MBean リポジトリへのロード

ドメインの MBean リポジトリ内のデータを WebLogicMBeanDumper で生成した XML ファイルに置き換えるには、次の手順に従います。

1. 管理サーバを停止して、サーバが MBean リポジトリを定期的に更新しないようにします。
2. *domain_name*/userConfig で、Security ディレクトリの名前を変更して userConfig ディレクトリ ツリーの外部に移動します。

   注意： 単にコピーを作成するのではなく、ディレクトリの**名前を変更**して移動する必要があります。これらの手順では、セキュリティ MBean リ

ポジトリ全体をダンプ済み XML ファイルのデータに置き換えるものとします。次の手順に進む前に *domain_name*/userConfig/Security ディレクトリが存在する場合、既存の MBean の変更および MBean の追加は正常に行われますが、MBean の削除は行われません。

3. Java クラスパスを設定するために、次のコマンドを入力します。

   WL_HOME\server\bin\setWLSEnv.cmd (Windows の場合)
   WL_HOME/server/bin/setWLSEnv.sh (UNIX の場合)

   詳細については、『管理者ガイド』の「クラスパスの設定」を参照してください。

4. ドメインのルート ディレクトリから、次のコマンドを入力します。

   ```
   java weblogic.management.commo.WebLogicMBeanLoader XML-file
   ```

デフォルト セキュリティ データ リポジトリの作成

ドメイン内のすべてのセキュリティ MBean を再生成して WebLogic Server によってインストールされたコンフィグレーションに戻す場合、次の手順に従います。

1. 管理サーバを停止して、サーバが MBean リポジトリを定期的に更新しないようにします。
2. *domain_name*/userConfig で、Security ディレクトリの名前を変更して userConfig ディレクトリ ツリーの外部に移動します。
3. 管理サーバを起動します。

管理サーバは、デフォルト値を持つインストールされたセキュリティ プロバイダを使用する新しいセキュリティ MBean リポジトリを生成します。このインストールされたコンフィグレーションを持つサーバにログオンするには、ドメインの作成時に指定した管理ユーザ名を入力する必要があります。詳細については、『管理者ガイド』の「初期管理ユーザ名の指定」を参照してください。

WebLogic LDAP リポジトリのバックアップ

WebLogic Server と一緒にインストールされるデフォルトの認証プロバイダ、認可プロバイダ、ロール マッピング プロバイダ、および資格マッピング プロバイダは、自身のデータを LDAP サーバに格納します。各 WebLogic Server には、組み込み LDAP サーバが存在します。管理サーバには、すべての管理対象サーバにレプリケートされるマスター LDAP サーバが存在します。これらのインストールされたプロバイダを使用するセキュリティ レルムが存在する場合、次のディレクトリ ツリーをバックアップすることをお勧めします。

*domain_name**adminServer*\ldap

ここで *domain_name* はドメインのルート ディレクトリで、*adminServer* は管理サーバが実行時データやセキュリティ データなどを格納するために生成するディレクトリです。各 WebLogic Server はこのディレクトリを生成しますが、バックアップが必要なのは管理サーバ上の LDAP データだけです。

たとえば、セキュリティ レルムは WebLogic Server と一緒にインストールされるデフォルト認証プロバイダを使用します。管理サーバの名前が `myAdminServer`、ドメインの名前が `myDomain` の場合、次のディレクトリ ツリーをバックアップします。

```
myDomain\myAdminServer\ldap
```

`ldap` ディレクトリの下の `ldapfiles` サブディレクトリには、LDAP サーバのデータ ファイルが格納されています。このディレクトリ内のデータ ファイルには、ユーザ、グループ、グループ メンバー、ポリシー、およびロール情報が格納されます。`ldap` ディレクトリの他のサブディレクトリには、LDAP サーバのメッセージ ログなどの情報およびレプリケートされる LDAP サーバに関するデータが格納されます。

`ldap` ディレクトリ ツリーのバックアップ中に自分または他人がいずれかのセキュリティ プロバイダのデータを変更した場合、`ldapfiles` サブディレクトリ内のファイルのバックアップが不整合な状態になる可能性があります。たとえば、誰かがインストールされたデフォルト認証プロバイダを使用してユーザを追加しようとする場合、その作業の開始時から終了時の間にバックアップが開始される場合もあります。

1 日に一度、サーバは書き込み処理を中断して LDAP データの専用バックアップを作成します。このバックアップは ZIP ファイルで ldap\backup ディレクトリに格納され、書き込み処理が再開されます。このバックアップは整合性が保証されますが、最新のセキュリティ データが含まれていない場合があります。

この LDAP バックアップのコンフィグレーションについては、Administration Console オンライン ヘルプの「組み込み LDAP サーバのバックアップのコンフィグレーション」を参照してください。

管理対象サーバの LDAP データをバックアップする必要はありません。マスター LDAP サーバは、自身に更新が行われると各管理対象サーバの LDAP をレプリケートします。ドメインの管理サーバが使用不能になると、WebLogic セキュリティ プロバイダはセキュリティ データを変更できません (管理対象サーバの LDAP リポジトリは複製であるため変更できません)。

SerializedSystemIni.dat およびセキュリティ証明書のバックアップ

すべてのサーバは、SerializedSystemIni.dat という名前のファイルを作成し、サーバのルート ディレクトリに格納します。このファイルには、サーバの起動時に必要な暗号化されたセキュリティ データが格納されます。このファイルはバックアップが必要です。

サーバが SSL を使用する場合、セキュリティ証明書とキーもバックアップする必要があります。これらのファイルの場所はユーザがコンフィグレーションできます。

管理対象サーバの動作中における管理サーバの再起動

管理対象サーバが動作を続けている状況で管理サーバがダウンした場合、ドメインの管理を回復するために、すでに動作している管理対象サーバを再起動する必要はありません。アクティブなドメインの管理を回復する手順は、管理サーバが起動したときと同じマシンで管理サーバを再起動できるかどうかによって異なります。

この節では、以下のタスクについて説明します。

- 同じマシンでの管理サーバの再起動
- 別のマシンでの管理サーバの再起動

同じマシンでの管理サーバの再起動

管理対象サーバが動作を続けている状況で WebLogic 管理サーバを再起動する場合、管理サーバでは動作している管理対象サーバの存在を検出できます。

注意： 起動コマンドと起動スクリプトに `-Dweblogic.management.discover=false` が含まれていないことを確認してください。これが含まれていると、管理サーバは動作している管理対象サーバを検出できません。`-Dweblogic.management.discover` の詳細については、『管理者ガイド』の「よく使用される任意指定の引数」を参照してください。

ドメインのルートディレクトリには、`running-managed-servers.xml` というファイルが含まれています。このファイルは、管理サーバが認識している管理対象サーバのリストです。管理サーバの起動時に、管理サーバはこのリストを使用して動作している管理対象サーバの存在をチェックします。

管理サーバを再起動しても、管理対象サーバは静的属性のコンフィグレーションを更新しません。静的属性とは、サーバがその起動プロセス中にのみ参照する属性です。静的なコンフィグレーション属性の変更を反映するためには、WebLogic Server を再起動する必要があります。管理対象サーバを検出した場

合、管理サーバでは管理対象サーバをモニタするか、またはサーバの動作中にコンフィグレーションできる属性 (動的属性) の値を実行時に変更することしかできません。

別のマシンでの管理サーバの再起動

マシンのクラッシュにより、同じマシンで管理サーバを再起動できない場合は、次のようにして動作している管理対象サーバの管理を回復できます。

1. 新しい管理マシンで WebLogic Server ソフトウェアをインストールします (インストールされていない場合)。
2. アプリケーション ファイルをバックアップからコピーするか、または共有ディスクを使用して、新しい管理サーバがこれらのファイルを使用できるようにします。新しいファイル システム上でのアプリケーション ファイルの相対位置は、元の管理サーバのファイル システムと同じにする必要があります。
3. コンフィグレーション データとセキュリティ データをバックアップからコピーするか、または共有ディスクを使用して、新しい管理サーバがこれらのデータを使用できるようにします。詳細については、4-1 ページの「コンフィグレーション データのバックアップ」および 4-2 ページの「セキュリティ データのバックアップ」を参照してください。
4. 新しいマシンで管理サーバを再起動します。

注意 : 起動コマンドと起動スクリプトに `-Dweblogic.management.discover=false` が含まれていないことを確認してください。これが含まれていると、管理サーバは動作している管理対象サーバを検出できません。`-Dweblogic.management.discover` の詳細については、『管理者ガイド』の「よく使用される任意指定の引数」を参照してください。

管理サーバは、起動時に管理対象サーバと通信して、管理サーバが異なる IP アドレスで動作していることを通知します。

管理サーバにアクセスできない場合の管理対象サーバの起動

通常、管理対象サーバは起動時に管理サーバとやり取りしてそのコンフィグレーション情報を取得します。起動時に管理対象サーバが指定された管理サーバに接続できない場合、コンフィグレーション ファイルおよびその他のファイルを直接読み込むことによって自身のコンフィグレーションを取得できます。

このような方法で起動する管理対象サーバは、管理対象サーバ独立モードで動作します。このモードでは、サーバはキャッシュされたアプリケーション ファイルを使用して、そのサーバに割り当てられているアプリケーションをデプロイします。管理サーバとの通信が回復するまで、管理対象サーバのコンフィグレーションを変更することはできません。

この節では、以下の項目について説明します。

- 管理対象サーバ独立モードでの起動
- セキュリティ レルムの確認
- ドメインのコンフィグレーション ファイルのレプリケート
- 管理対象サーバによる管理サーバとの通信の復元方法
- 管理対象サーバの独立の無効化

管理対象サーバ独立モードでの起動

管理対象サーバ独立モードが有効になっており (サーバのデフォルト設定)、管理対象サーバの起動時に管理サーバが使用不能の場合、管理対象サーバは自身のルート ディレクトリから次のファイルを検索します。

- コンフィグレーション ファイル (デフォルトは `config.xml`)
- `SerializedSystemIni.dat`
- `boot.properties` (ユーザ名とパスワードの暗号化バージョンが格納されているオプションのファイル)。詳細については、『管理者ガイド』の「ユーザ名とパスワードのプロンプトの回避」を参照してください。

デフォルトでは、サーバはそのルート ディレクトリがサーバ起動コマンドを発行するディレクトリだと見なします。

管理サーバと同じドメインおよび同じマシンで動作する場合、管理対象サーバはデフォルトで管理サーバとルート ディレクトリを共有します。このような管理対象サーバを起動した場合、管理対象サーバはコンフィグレーション ファイルを自動的に発見します。

管理対象サーバが管理サーバとそのルート ディレクトリを共有しない場合は、以下のいずれかが可能です。

- 管理サーバのルート ディレクトリ (またはバックアップ) から管理対象サーバのルート ディレクトリにコンフィグレーション ファイルをコピーします。管理対象サーバを起動すると、管理対象サーバはコピーしたコンフィグレーション ファイルを使用します。
- `-Dweblogic.RootDirectory=`*`path`* 起動オプションを使用して、これらのファイルが格納されているルート ディレクトリを指定します。
- 4-16 ページの「ドメインのコンフィグレーション ファイルのレプリケート」の説明に従って、コンフィグレーション データのレプリケーションを有効にします。このオプションを使用するためには、管理対象サーバ独立モードで起動される前に管理対象サーバが最低 1 回は管理サーバとの接続を確立している必要があります。

注意： サーバ用に SSL を設定した場合、各サーバには独自の証明書ファイル、キー ファイル、およびその他の SSL 関連ファイルのセットが必要になります。管理対象サーバは、SSL 関連ファイルを管理サーバから取得しません (ドメインのコンフィグレーション ファイルにサーバごとのこれらのファイルのパス名が格納されている場合でも)。管理対象サーバ独立モードで起動した場合、SSL 関連ファイルがアクセスできるマシンに格納されていれば、それらのファイルをコピーまたは移動する必要はありません。

ノード マネージャと管理対象サーバ独立モード

ノード マネージャを使用してサーバを管理対象サーバ独立モードで起動することはできません。ノード マネージャを使用する場合、管理サーバが存在する必要があります。管理サーバが使用不能の場合、ローカル ホストにログオンして

管理対象サーバを起動する必要があります。ノード マネージャの詳細については、5-1 ページの「ノード マネージャによるサーバの可用性の管理」を参照してください。

MSI モードと管理対象サーバのルート ディレクトリ

デフォルトのサーバ インスタンスは、そのインスタンスの起動元のディレクトリがそのルート ディレクトリであると判断します。サーバのルート ディレクトリの詳細については、「サーバのルート ディレクトリ」を参照してください。

コンフィグレーション データのレプリケーションを有効にした場合で (4-16 ページの「ドメインのコンフィグレーション ファイルのレプリケート」を参照)、かつ管理サーバの動作中に少なくとも 1 度管理対象サーバを起動した場合は、`msi-config.xml` と `SerializedSystemIni.dat` がすでにサーバのルート ディレクトリに存在します。`boot.properties` ファイルはレプリケートされません。まだ管理対象サーバのルート ディレクトリにない場合は、作成する必要があります。詳細については、『管理者ガイド』の「ユーザ名とパスワードのプロンプトの回避」を参照してください。

`msi-config.xml` と `SerializedSystemIni.dat` がルート ディレクトリにない場合は、以下のいずれかを行うことができます。

- 管理サーバのルート ディレクトリ (またはバックアップ) から管理対象サーバのルート ディレクトリへ、`config.xml` と `SerializedSystemIni.dat` をコピーします。次に、そのコンフィグレーション ファイルの名前を `msi-config.xml` に変更します。
- `-Dweblogic.RootDirectory=`*`path`* 起動オプションを使用して、これらのファイルが格納されているディレクトリを指定します。

MSI モードとドメイン ログ ファイル

WebLogic Server の各インスタンスは、ログ メッセージをそのローカル ログ ファイルとドメイン全体のログ ファイルに書き込みます。ドメイン ログ ファイルからは、ドメイン内のすべてのサーバのメッセージを表示できます。

通常、管理対象サーバはメッセージを管理サーバに転送し、管理サーバはそのメッセージをドメイン ログ ファイルに書き込みます。ただし、管理対象サーバ

の動作モードが MSI モードである場合は、管理対象サーバが直にドメイン ログ ファイルに書き込みます。

デフォルトでは、ローカル ログ ファイルとドメイン ログ ファイルのパス名は、管理対象サーバのルート ディレクトリを基準とした相対パスです。それらのデフォルト設定を使用し、管理対象サーバがそのルート ディレクトリに配置されていて、かつ管理サーバとルート ディレクトリを共有していない場合、MSI モードで動作する管理対象サーバはそれ専用のドメイン ログ ファイルをそのルート ディレクトリに作成します。

管理対象サーバが管理サーバとルート ディレクトリを共有しているか、ドメイン ログの絶対パス名が指定された場合、MSI モードの管理対象サーバは管理サーバが作成したドメイン ログ ファイルに書き込みを行います。

注意： 管理対象サーバには、その既存のファイルに書き込むためのパーミッションが必要です。管理サーバと管理対象サーバを異なるオペレーティング システム アカウントで実行する場合は、両方のユーザ アカウントが書き込みパーミッションを持つようにドメイン ログ ファイルのファイル パーミッションを変更する必要があります。

セキュリティ レルムの確認

コンフィグレーション ファイルに加え、サーバはセキュリティ レルムにアクセスしなければ起動プロセスを完了できません。

WebLogic Server がインストールしたセキュリティ レルムを使用する場合、管理サーバはドメインのセキュリティ データを格納するために LDAP サーバを保持します。すべての管理対象サーバは、この LDAP サーバをレプリケートします。管理サーバに障害が発生した場合、管理対象サーバ独立モードで動作する管理対象サーバはレプリケートした LDAP サーバをセキュリティ サービス用に使用できます。

サード パーティのセキュリティ プロバイダを使用する場合、管理対象サーバはセキュリティ データにアクセスできなければ起動プロセスを完了できません。

ドメインのコンフィグレーション ファイルのレプリケート

管理対象サーバ独立モードには、必須のコンフィグレーション ファイルを管理サーバのルート ディレクトリから管理対象サーバのルート ディレクトリに 5 分ごとにコピーするオプションが用意されています。バックアップ方針、およびドメインのコンフィグレーションの更新頻度によっては、このオプションは大きいファイルをネットワーク上でコピーすることで生じるパフォーマンス コストに見合わない場合があります。

この機能を利用するには、その前に必ず必要な環境を初期化する必要があります。

警告 :　別のサーバとインストレーションまたはルート ディレクトリを共有するサーバに対してはファイルのレプリケーションを有効にしないでください。両方のサーバで予測できないエラーが発生する可能性があります。

1. ドメインの管理サーバを起動します。
2. ドメインのコンフィグレーション ファイルをレプリケートするように管理対象サーバをコンフィグレーションします。

   Administration Console オンライン ヘルプの「ドメインのコンフィグレーション ファイルのレプリケート」を参照してください。
3. 少なくとも 5 分間、管理サーバを動作したままにします。

管理対象サーバが管理サーバにアクセスしてコンフィグレーション ファイルをそのルート ディレクトリにコピーした後、その管理対象サーバでは管理対象サーバ独立モードで起動するときにそのコピーを使用することができます。

このオプションでは、起動 ID ファイルはレプリケートされません。起動 ID ファイルの詳細については、『管理者ガイド』の「ユーザ名とパスワードのプロンプトの回避」を参照してください。

管理対象サーバによる管理サーバとの通信の復元方法

`-Dweblogic.management.discover=true` (デフォルト設定) の場合、管理サーバは起動時に動作している管理対象サーバの存在を検出します。起動時に、管理サーバは `running-managed-servers.xml` ファイルの永続コピーを参照し、すべての管理対象サーバにその存在を通知します。管理対象サーバが管理対象サーバ独立モードで動作している場合、管理対象サーバはこの自己管理モードを非アクティブ化し、新しいコンフィグレーション変更通知を受け取るために管理サーバに自身を登録します。

このシナリオでの管理サーバの再起動については、4-10 ページの「管理対象サーバの動作中における管理サーバの再起動」を参照してください。

管理対象サーバの独立の無効化

デフォルトでは、管理対象サーバ独立モードは有効になっています。このモードの無効化については、Administration Console オンライン ヘルプの「管理対象サーバの独立の無効化」を参照してください。

自動状態モニタと管理対象サーバの再起動

WebLogic Server 7.0 には、ドメイン内のサーバの信頼性と可用性を向上させるための自動状態モニタ機能が用意されています。各 WebLogic Server 内の選択されたサブシステムは、そのサブシステムに固有の条件に基づいて自身の状態をモニタします。たとえば、JMS サブシステムは JMS スレッド プールの状態をモニタし、コア サーバ サブシステムはデフォルトおよびユーザ定義の実行キュー統計をモニタします。個々のサブシステムは、整合性および信頼性のある状態で動作できないと判断した場合、ホスト サーバに自己の状態を「障害」として登録します。

また、各 WebLogic Server は登録されているすべてのサブシステムの状態をチェックして、サーバの全体的な有効性を調べます。1 つまたは複数の重要なサブシステムが FAILED 状態に達していることを発見した場合、サーバは自己の状態を FAILED に設定して、アプリケーションを適切にホストできないことを示します。

ノード マネージャ アプリケーションと一緒に使用すると、サーバ自動状態モニタでは障害が発生したサーバを自動的に再起動します。これにより、ドメインの全体的な信頼性が向上し、管理者の介入が不要になります。詳細については、5-1 ページの「ノード マネージャによるサーバの可用性の管理」を参照してください。

5　ノード マネージャによるサーバの可用性の管理

以下の節では、WebLogic Server ドメインでのノード マネージャのコンフィグレーションおよび使用方法について説明します。

ノード マネージャの概要

ノード マネージャは、WebLogic Server に付属のスタンドアロン Java プログラムであり、以下のことに使用できます。

- リモートの管理対象サーバの起動
- 不測の事態 (システム クラッシュ、ハードウェアの再起動、サーバの障害など) のために停止した管理対象サーバの再起動
- 管理対象サーバの状態の自動モニタおよび「障害」状態に達したサーバ インスタンスの再起動
- 停止要求に応答しなくなった管理対象サーバの停止または強制停止

環境に関する考慮事項

以下の節では、ノード マネージャを実行する環境について説明します。

ノード マネージャは管理対象サーバのホスト マシンで動作する

ノード マネージャを使用するには、管理対象サーバの各ホスト マシンで 1 つのノード マネージャ プロセスをコンフィグレーションして実行する必要があります。1 つのノード マネージャ プロセスで、1 つのマシン上にある複数の管理対象サーバを管理できます。また、1 つのノード マネージャ プロセスで複数のドメインの管理対象サーバを管理することもできます。

ノード マネージャはサービスとして実行する必要がある

ノード マネージャは、UNIX マシンではデーモンとして、Windows ベースのマシンでは Windows サービスとして動作するようにコンフィグレーションします (5-18 ページの「Windows サービスとしてのノード マネージャの起動」を参照)。このようにコンフィグレーションすることで、マシンの再起動の後にノード マネージャが確実に利用可能になります。

注意 : ノード マネージャで管理サーバを起動または強制停止することはできません。プロダクション環境の場合、管理サーバが動作しているマシンでは、管理対象サーバが動作していない限りノード マネージャを実行する必要はありません。

ノード マネージャは SSL を使用して管理サーバと通信する

ノード マネージャと管理サーバの間の通信では、セキュア ソケット レイヤ プロトコルが使用されます。セキュア ソケット レイヤ プロトコルでは、認証と暗号化を利用できます。ノード マネージャは、セキュアでない通信プロトコルと一緒に使用できません。

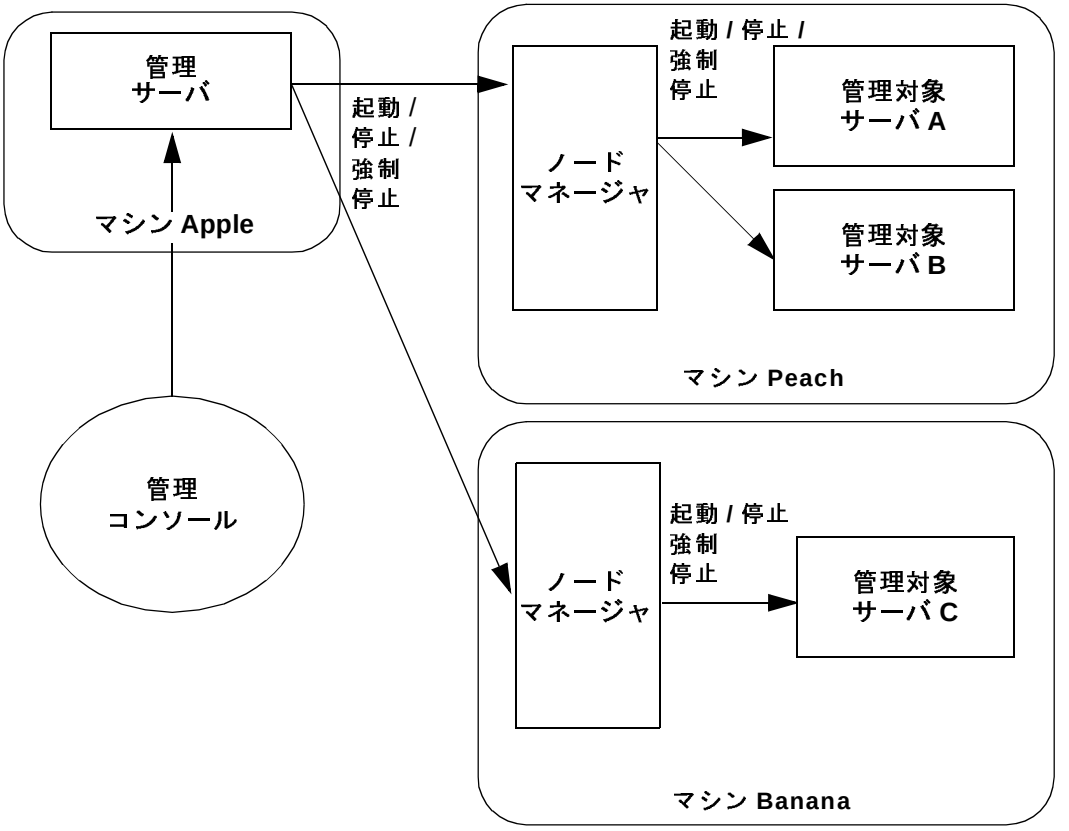
ノード マネージャのネイティブ サポート

BEA では、Windows、Solaris、および HP UX オペレーティング システム用のネイティブ ノード マネージャ ライブラリを提供しています。Solaris および HP UX 以外の UNIX オペレーティング システムの場合、ノード マネージャの起動時にコマンド ラインで `weblogic.nodemanager.nativeVersionEnabled` オプションを無効にする必要があります。詳細については、5-17 ページの「ノード マネージャの起動」を参照してください。

ノード マネージャのアーキテクチャ

次の図に、単一の WebLogic Server ドメイン用のノード マネージャ アーキテクチャを示します。

図 5-1　ノード マネージャのアーキテクチャ



ノード マネージャは、WebLogic Server ドメインの他のプロセスと共同で管理対象サーバの起動と停止を支援します。

ユーザは、Administration Console を使用して管理対象サーバを起動、中断、または停止する要求を発行できます。それらの要求は、ドメインの管理サーバによって、管理対象サーバのホスト マシン上のノード マネージャ プロセスにディスパッチされます。

また、管理サーバは管理対象サーバの状態を変更するためのプログラマティック要求を受け入れることもできます。`weblogic.Admin` コマンドライン ユーティリティなどの JMX クライアントから送信されるそれらの要求は、管理対象サーバのホスト マシン上のノード マネージャ プロセスにディスパッチされます。

ノード マネージャは、それが以前に起動した管理対象サーバの状態に関するローカルにキャッシュされた情報に基づいて、管理サーバやプログラムに基づくクライアントによって要求されたアクションが実行可能であることを確認します。受信した要求を確認した後、ノード マネージャはそれらの要求を実行のためにターゲットの管理対象サーバに転送します。

ターゲットの管理対象サーバは、要求された操作 (停止、中断、または再開) を実行します。管理対象サーバがノード マネージャからの停止要求に応答できない場合、ノード マネージャ プロセス自体が停止を実行します。

ノード マネージャは、特定のドメインに属しません。1 つのノード マネージャ プロセスだけで、アクセス可能などのドメインの管理対象サーバでも起動できます。

ノード マネージャは、ノード マネージャおよび管理サーバ プロセスとは独立した、対象マシン上の専用プロセスで管理対象サーバを起動します。ノード マネージャを使用して管理対象サーバを起動すると、通常は `STDOUT` または `STDERROR` に出力される関連付けられたステータス メッセージが Administration Console に表示され、その管理対象サーバのノード マネージャ ログ ファイルに書き込まれます (5-27 ページの「管理対象サーバのログ ファイル」を参照)。

ノード マネージャの機能

以下の節では、ノード マネージャの主要な機能を説明します。それらの機能は、5-7 ページの「ノード マネージャの機能の前提条件」で説明されている要件と制限に依存します。

ノード マネージャは管理対象サーバを起動および停止する

5-4 ページの「ノード マネージャのアーキテクチャ」で説明されているように、Administration Console のユーザまたは JMX クライアントから送られてきた要求は管理サーバによって、ターゲットの管理対象サーバのホスト マシン上にあるノード マネージャ プロセスに転送されます。ノード マネージャは、受信した各要求が実現可能かどうかを確認し、確認した要求をターゲットの管理対象サーバにディスパッチします。

注意： ノード マネージャは、5-8 ページの「ノード マネージャはサーバの起動モードを無視する」で説明されているように、必ずその最後の実行時状態で管理対象サーバを起動します。

ノード マネージャはそれが起動した管理対象サーバをモニタする

WebLogic Server 7.0 のサブシステムでは、自動状態モニタが実行されます。1 つまたは複数の重要なサブシステムで障害が発生すると、管理対象サーバはその状態を「障害」に設定します。ノード マネージャは、それが起動した各管理対象サーバの状態を定期的にチェックします。

ノード マネージャによる管理対象サーバの状態チェックの頻度を管理する手順、および管理対象サーバからの応答を待つ時間を管理する手順については、5-15 ページの「管理対象サーバのモニタ、停止、および再起動のコンフィグレーション」を参照してください。

管理対象サーバがどのようにしてそれ自体の状態をモニタするのかの詳細については、「サーバの自動状態モニタ」を参照してください。

ノード マネージャは利用できない管理対象サーバおよび障害の発生した管理対象サーバを強制停止できる

ノード マネージャでは、状態が「障害」であると報告する管理対象サーバ、および状態の問い合わせに応答しない管理対象サーバを必要に応じて停止できます。デフォルトでは、この動作は無効です。この機能を有効にする手順については、5-15 ページの「管理対象サーバのモニタ、停止、および再起動のコンフィグレーション」を参照してください。

ノード マネージャは管理対象サーバを自動的に再起動する

デフォルトでは、ノード マネージャは状態が「障害」となっている管理対象サーバを再起動しようとします。以下の方法があります。

- 自動的な再起動を無効にする。手順については、5-15 ページの「管理対象サーバのモニタ、停止、および再起動のコンフィグレーション」を参照してください。
- ノード マネージャが管理対象サーバの再起動を試みる期間および回数をコンフィグレーションする。手順については、5-15 ページの「管理対象サーバのモニタ、停止、および再起動のコンフィグレーション」を参照してください。

ノード マネージャは、5-8 ページの「ノード マネージャはサーバの起動モードを無視する」で説明されているように、必ずその最後の実行時状態で管理対象サーバを起動します。

ノード マネージャの機能の前提条件

ノード マネージャの管理対象サーバの状態をモニタする機能、および管理対象サーバを自動的に停止および再起動する機能は以下の条件に依存します。

- サーバ インスタンスは、2-7 ページの「サーバ名とリスン アドレスの指定」で説明したガイドラインに従ってコンフィグレーションしなければならない。
- ノード マネージャでは、ノード マネージャを使用して起動した管理対象サーバのみ自動的にモニタ、停止、および再起動できる。コマンドラインから直接起動された管理対象サーバではそれらのサービスを利用できません。
- ノード マネージャは、サーバ インスタンスの `AutoRestartEnabled` 属性が選択されている場合のみ管理対象サーバを自動的に再起動する。
- 管理対象サーバで障害を発生させたイベントがノード マネージャでも障害を発生させる場合は、障害の発生した管理対象サーバの自動的な再起動について以下の追加の前提条件が適用される。
  - 管理対象サーバの動作するマシン上のノード マネージャ プロセスは、オペレーティング システムによって、または手動で再起動する必要があります。

- 管理対象サーバの HostsMigratableServices 属性を「false」に設定する必要があります。HostsMigratableServices が「true」(デフォルト値) に設定されている管理対象サーバは、ノード マネージャで自動的に再起動されません。HostsMigratableServices の値は serverMBean を使用して設定します。

  管理対象サーバで障害が発生しても、ノード マネージャでは障害が発生しない場合、ノード マネージャで HostsMigratableServices の設定が考慮されることはありません。

以下のシナリオは、HostsMigratableServices の使い方を例示しています。例 1 で、HostsMigratableServices の値は管理対象サーバが自動的に再起動される要因ではありません。例 2 で、HostsMigratableServices の値は管理対象サーバが自動的に再起動される要因です。

- 例 1—(ノード マネージャによって起動された) 管理対象サーバがクラッシュするが、ノード マネージャはそのまま動作を続ける。この場合、HostsMigratableServices の値はノード マネージャが管理対象サーバを再起動するかどうかの要因ではありません。サーバ インスタンスの AutoRestartEnabled 属性が選択されている場合、ノード マネージャは管理対象サーバを再起動します。
- 例 2— 動作中のノード マネージャ プロセスとそのノード マネージャによって起動された管理対象サーバのあるマシンが再起動される。ノード マネージャはマシン上でサービスとして動作するように設定されており、マシンの再起動時に自動的に起動する。ノード マネージャは、管理対象サーバが動作していないという事実を検出する。この場合、HostsMigratableServices の値はノード マネージャが管理対象サーバを再起動するかどうかの要因となります。HostsMigratableServices の値が false で、サーバ インスタンスの AutoRestartEnabled 属性が選択されている場合、ノード マネージャは管理対象サーバを再起動します。

ノード マネージャはサーバの起動モードを無視する

自動的に、あるいはユーザ コマンドまたはスクリプトの結果としてノード マネージャが管理対象サーバを起動する場合、ノード マネージャはそのサーバにコンフィグレーションされている起動モードを使用しません。管理対象サーバの

起動モードは、-Dweblogic.management.startupMode コマンド引数を使用して指定するか、または Administration Console の [コンフィグレーション | 一般] タブで指定できます。代わりに、ノード マネージャは必ずその最後の実行時状態で管理対象サーバを起動します。

たとえば、STANDBY モードで起動するようにコンフィグレーションされている管理対象サーバが RUNNING 状態のときに障害で停止した場合、ノード マネージャは自動的に RUNNING 状態でサーバ インスタンスを再起動します (ノード マネージャが管理対象サーバの状態をモニタするようにコンフィグレーションされており、5-7 ページの「ノード マネージャの機能の前提条件」の前提条件が満たされていると想定する)。

サーバの起動モードの詳細については、『管理者ガイド』の「WebLogic Server の起動と停止」を参照してください。

ノード マネージャのコンフィグレーション

WebLogic Server ドメインでノード マネージャを使用するには、以下のタスクを行う必要があります。

- 5-10 ページの「ノード マネージャ ホスト ファイルの設定」
- 5-11 ページの「ノード マネージャに対する SSL のコンフィグレーション」
- 5-12 ページの「ノード マネージャを使用するためのマシンのコンフィグレーション」
- 5-13 ページの「管理対象サーバの起動引数のコンフィグレーション」

これらのタスクを実行した後は、Administration Console 経由でノード マネージャを使用して管理対象サーバを明示的に起動および停止できます。デフォルトでは、以下の動作がコンフィグレーションされます。

- 管理対象サーバの自動再起動が有効になる。ノード マネージャは、300 秒の間隔で最大 2 回、状態が「障害」である管理対象サーバの再起動を試みます。
- 管理対象サーバの自動停止が無効になる。ノード マネージャは、「障害」状態にあることを検知した管理対象サーバを停止しません。

- ノード マネージャが、起動した管理対象サーバをモニタする (180 秒おきに各管理対象サーバの自己申告による状態をチェックする)。ノード マネージャは、管理対象サーバからの応答を 60 秒待ちます。

これらのコンフィグレーション設定を調整するには、5-15 ページの「管理対象サーバのモニタ、停止、および再起動のコンフィグレーション」の手順に従います。

ノード マネージャ ホスト ファイルの設定

ノード マネージャは、信頼性のあるホスト上で動作する管理サーバからのみコマンドを受け入れます。ノード マネージャ プロセスの信頼性のあるホストは、ファイル内の IP アドレスまたは DNS 名で識別されます。そのファイルは、デフォルトでは `nodemanager.hosts` という名前で *`WL_HOME`*`\common\nodemanager\config` ディレクトリにインストールされます (*`WL_HOME`* は WebLogic Server の最上位のインストール ディレクトリ)。

注意 : 信頼性のあるホスト ファイルに別の名前および位置を指定するには、ノード マネージャのコマンドライン引数を使用します。詳細については、5-21 ページの「ノード マネージャのコマンドライン引数」を参照してください。

`nodemanager.hosts` には、管理サーバが実行される信頼性のあるホストごとに 1 行を格納します。デフォルトでは、`nodemanager.hosts` ファイルには、次の 2 つのエントリだけが記述されています。

```
localhost
127.0.0.1
```

信頼性のあるホストを追加または削除するには、テキスト エディタでこのファイルを編集します。

DNS 名で信頼性のあるホストを識別する場合は、ノード マネージャを起動するときに DNS の逆引き参照を有効にする必要があります。DNS の逆引き参照は、次のコマンドライン引数で有効にします。

```
-Dweblogic.nodemanager.reverseDnsEnabled=true
```

デフォルトでは、DNS の逆引き参照は無効です。

通常のプロダクション環境では、ノード マネージャは管理サーバと同じマシン上では動作しない場合があります。管理サーバのホスト マシンのみをリストするには、nodemanager.hosts を編集します。

ノード マネージャに対する SSL のコンフィグレーション

ノード マネージャは、管理サーバおよび管理対象サーバとの通信の保護に双方向 SSL を使用します。SSL のコンフィグレーションには、ノード マネージャと、ノード マネージャが通信する管理サーバおよび管理対象サーバの各々の ID と信頼を取得してから、適切な ID と信頼を使って、ノード マネージャ、管理サーバ、および任意の管理対象サーバをコンフィグレーションすることが含まれます。また、ホスト名検証の使用を考慮に入れる必要があります。

WebLogic Server 7.0 以降では、ノード マネージャは WebLogic Server 7.0 で使用される証明書フォーマットおよび公開鍵インフラストラクチャを使用します。WebLogic Server 7.0 より前では、ノード マネージャはパスワードで保護されたプライベート キーとユーザ ID 証明書の両方を含む単一の証明書ファイルを使用していました。

SSL の一般的な情報については、『WebLogic Security の管理』の「SSL の概要」を参照してください。ノード マネージャの SSL をコンフィグレーションする方法については、『WebLogic Security の管理』の「ノード マネージャに関する SSL のコンフィグレーション」を参照してください。

SSL を使用するノード マネージャの起動

ノード マネージャのプロセスを起動して SSL を使用するときには、SSL 通信に使用するキーストア、パスワード、および証明書ファイルを認識する起動引数を指定する必要があります。

- -Dweblogic.nodemanager.keyFile— 暗号化されたキー ファイルのパスを指定します。
- -Dweblogic.nodemanager.keyPassword— キー ファイルが暗号化されている場合に使用するパスワードを指定します。

- -Dweblogic.nodemanager.certificateFile— 証明書ファイルのパスを指定します。
- -Dweblogic.security.SSL.trustedCAKeyStore— プライベート キーを保持するキーストアのパスを指定します。
- -Dweblogic.nodemanager.sslHostNameVerificationEnabled— nodemanager.hosts ファイルによる管理サーバ ホスト名のチェックを有効または無効にします。このチェックは、デモ用証明書を使用するときにのみ無効にします。

たとえば、サンプル SSL 証明書およびキー ファイルの SSL コンフィグレーションを使用してノード マネージャを起動するには、次のように指定します。

```
java.exe -Xms32m -Xmx200m  -classpath %CLASSPATH% -Dbea.home=c:\bea
-Dweblogic.nodemanager.keyFile=e:\bea\user_domains\mydomain\demok
ey.pem
-Dweblogic.security.SSL.trustedCAKeyStore=e:\bea\weblogic700\serv
er\lib\cacerts
-Dweblogic.nodemanager.certificateFile=e:\bea\user_domains\mydoma
in\democert.pem
-Djava.security.policy=e:\weblogic700\server\lib\weblogic.policy
-Dweblogic.nodemanager.sslHostNameVerificationEnabled=false
weblogic.nodemanager.NodeManager
```

ノード マネージャのコマンドライン引数については、5-21 ページの「ノード マネージャのコマンドライン引数」を参照してください。

ノード マネージャを使用するためのマシンのコンフィグレーション

ノード マネージャを使用する前に、ノード マネージャ プロセスを実行するマシンごとにマシン定義を作成する必要があります。

1. 管理サーバが動作している状態で、Administration Console を起動します (まだ動作していない場合)。
2. 左ペインで [マシン] ノードを選択してマシン テーブルを表示します。
3. [新しい Machine のコンフィグレーション] (または [新しい UNIX Machine のコンフィグレーション]) を選択します。
4. 新しいマシンの名前を入力し、[作成] をクリックして新しいマシン エントリを作成します。

5. 新しいマシンの [ノード マネージャ] タブを選択します。接続属性を以下のように入力します。
   - [リスン アドレス]: ノード マネージャが要求をリスンするホスト名または IP アドレスを入力します。
   - [リスン ポート]: ノード マネージャが要求をリスンするポート番号を入力します。
6. [適用] をクリックして変更を反映させます。
7. 新しいマシンの [サーバ] タブを選択します。マシンがホストする各管理対象サーバについて、[選択可] カラムからサーバ インスタンスを選択し、矢印をクリックしてそのサーバ インスタンスを [選択済み] カラムに移動します。
8. [適用] をクリックして変更を反映させます。

管理対象サーバの起動引数のコンフィグレーション

ノード マネージャが管理対象サーバを起動するときには、Java コマンドラインまたは起動スクリプトでサーバ インスタンスを起動する場合と同じように、起動コマンド引数が使用されます。個々の引数の詳細については、『管理者ガイド』の「WebLogic Server の起動と停止」を参照してください。

管理対象サーバのオプションの起動引数を指定しない場合、ノード マネージャは独自のプロパティをデフォルトとして使用してサーバ インスタンスを起動します。これらのデフォルトでもサーバを起動できますが、カスタム起動引数を入力して各サーバが整合性および信頼性のある状態で起動するようにしてください。

ノード マネージャが管理対象サーバの起動で使用する起動引数をコンフィグレーションするには、次の手順に従います。

1. Administration Console を起動します (まだ動作していない場合)。
2. 左ペインで、[サーバ] を選択し、コンフィグレーションするサーバ インスタンスの名前を選択します。
3. 右ペインで、[コンフィグレーション | リモート スタート] を選択します。起動引数を以下のように編集します。

- [Java ホーム]— 管理対象サーバの起動時に使用する Java 実行時環境のフルパス (`c:\bea\jdk131` など) を入力します。
- [BEA Home]—BEA ホーム ディレクトリのフルパスを入力します。これは、すべての BEA 製品とライセンスがインストールされたルート ディレクトリです。
- [ルート ディレクトリ]— この管理対象サーバが存在するドメインのルート ディレクトリ (`c:\bea\Weblogic7.0\samples\server\config\examples` など) を入力します。

  ルート ディレクトリ値を入力しない場合は、デフォルトのノード マネージャ作業ディレクトリ (通常は *`WL_HOME`*`\common\nodemanager`) が使用されます。
- [クラスパス]— 管理対象サーバの起動に必要なフル クラス パスを入力します。最低でも、クラスパス オプションの以下の値を指定する必要があります。

  `WL_HOME``/server/lib/weblogic_sp.jar`

  `WL_HOME``/server/lib/weblogic.jar`

  クラスパスの要件の詳細については、『管理者ガイド』の「WebLogic Server の起動と停止」を参照してください。
- [引数]— このサーバの起動時に JVM に受け渡す追加引数を入力します。これらは、起動コマンドの `java` 部分の直後に追加される最初の引数です。たとえば、`-Xms32m` および `-Xmx200m` 引数を使用して Java ヒープ メモリを設定したり、特定のアプリケーション用のデバッグ オプションを入力できます。
- [セキュリティ ポリシー ファイル]—WebLogic セキュリティ ポリシー ファイルのフルパス (`c:\bea\weblogic700\server\lib\weblogic.policy` など) を入力します。
- [ユーザ名] (必須)—この管理対象サーバを起動する特権を持つユーザの名前 (`weblogic` など) を入力します。

注意： WebLogic Server バージョン 7.0 では、ユーザ名とパスワードを入力しなければ管理対象サーバを起動できません。管理対象サーバは、管理サーバに提供したユーザ名とパスワードを継承しません。

- [パスワード] (必須)—[変更 ...] をクリックして、サーバを起動するためのパスワードを変更します (`weblogic` など)。新しいパスワードを [新しいパスワード] および [再入力] テキスト ボックスに入力し、[適用] をクリックして変更を適用します。

4. [適用] をクリックします。

管理対象サーバのモニタ、停止、および再起動のコンフィグレーション

以下の目的に合わせてノード マネージャの機能をコンフィグレーションするには、この節の手順に従います。

- 管理対象サーバのモニタ。この機能の説明については、5-6 ページの「ノード マネージャはそれが起動した管理対象サーバをモニタする」を参照してください。
- 状態が「障害」となっている管理対象サーバの強制停止。この機能の説明については、5-6 ページの「ノード マネージャは利用できない管理対象サーバおよび障害の発生した管理対象サーバを強制停止できる」を参照してください。
- 管理対象サーバの再起動。この機能の説明については、5-7 ページの「ノード マネージャは管理対象サーバを自動的に再起動する」を参照してください。

  注意 : これらの機能は、5-7 ページの「ノード マネージャの機能の前提条件」で説明されている条件が満たされている場合に利用できます。

1. まだ行っていない場合、管理対象サーバの起動情報をコンフィグレーションします (5-13 ページの「管理対象サーバの起動引数のコンフィグレーション」を参照)。
2. 管理対象サーバが含まれるドメインの Administration Console にアクセスします。
3. 左ペインで、[サーバ] を選択し、コンフィグレーションするサーバの名前を選択します。
4. [コンフィグレーション | 状態モニタ] タブを選択して、管理対象サーバの自動再起動およびモニタ属性を表示します。

5. サーバの再起動の動作を変更するには、次の属性を編集します。
   - 右ペインの [自動再起動] チェックボックスが選択されていることを確認します。

     [自動再起動] が有効な場合、ノード マネージャは障害の後に管理対象サーバの再起動を試みます。障害の後に管理対象サーバがノード マネージャによって自動的に再起動されるのを望まない場合は、チェックボックスの選択を解除します。
   - [再起動間隔]: ノード マネージャが管理対象サーバの再起動を試みる間隔を入力します。この属性と [期間内の最大再起動回数] 属性を一緒に使用すると、サーバの再起動の試行回数を制限できます。3600 ～ 2147483647 秒の間で値を入力します (デフォルトは 300 秒)。
   - [期間内の最大再起動回数]: 上の [再起動間隔] で指定した間隔内でノード マネージャがこのサーバを再起動できる最大回数を入力します。0 ～ 2147483647 の間で値を入力します (デフォルトは 2 回)。
6. 自動的な停止と状態モニタの機能をコンフィグレーションするには、以下の属性を編集します。
   - [失敗時の自動強制停止]: このボックスをクリックすると、ノード マネージャは管理対象サーバの状態が「障害」に達したとき、またはノード マネージャが管理対象サーバの状態を問い合わせることができないときにその管理対象サーバを自動的に強制停止します。デフォルトでは、この属性は無効です。
   - [状態チェック間隔]: ノード マネージャが管理対象サーバの状態の問い合わせを行う間隔 (単位 : 秒) を指定します。1 ～ 2147483647 秒の間で値を入力します (デフォルトは 180 秒)。
   - [状態チェック タイムアウト]: ノード マネージャが状態を問い合わせたときに応答を待つ秒数を入力します。タイムアウトに達すると、ノード マネージャは管理対象サーバが「障害」状態あると想定し、プロセスを強制停止します。1 ～ 2147483647 秒の間で値を入力します (デフォルトは 60 秒)。
   - [再開始遅延]: ノード マネージャが管理対象サーバを再起動するまでの秒数を入力します。サーバ プロセスの強制停止後、そのサーバが使用していた TCP ポートを解放するために何秒か必要になる場合があります。このポートがまだアクティブな間にノード マネージャが管理対象サーバを再起動しようとした場合、その再起動は失敗します。
7. [適用] をクリックして変更を反映させます。

8. ノード マネージャを使用して管理対象サーバを起動しなかった場合は、ノード マネージャを使用してサーバを停止および再起動します。ノード マネージャで起動しなかったサーバでは、自動モニタおよび停止は実行できません。

ノード マネージャの起動

ノード マネージャ プロセスは、オペレーティング システム プロンプトで `java` コマンドを使用することによって手動で起動するか、スクリプトを用いて自動的に起動します。プロダクション環境では、起動スクリプトを作成するか (UNIX システムの場合)、ノード マネージャを Windows サービスとして設定することによって、ノード マネージャをマシンの起動時に自動的に起動する必要があります。

ノード マネージャは、どのような方法でノード マネージャ プロセスを起動するかに関係なく、同じ環境変数とコマンドライン オプションを使用します。

注意 : メモリ不足を回避するために、ノード マネージャを起動するときには常に最小ヒープ サイズを 32 MB (`-Xms32m`) に指定してください。ノード マネージャの起動スクリプトおよび `installNodeMgrSvc.cmd` スクリプトはこの Java 引数を自動的に設定します。

起動スクリプトを使用したノード マネージャの起動

初めてノード マネージャを使用するユーザのために、サンプル起動スクリプトが用意されています。これらのスクリプトは、*`WL_HOME`*`\server\bin` ディレクトリにインストールされています。標準のノード マネージャ起動スクリプトの名前は、Windows システムでは `startNodeManager.cmd`、UNIX システムでは `startNodeManager.sh` です。このスクリプトは、必要な Node Manager 環境値を設定し、ノード マネージャ プロセスを起動します。

ノード マネージャ起動スクリプトを初めて使用する場合、スクリプトはディレクトリを *WL_HOME*/common/nodemanager に変更します。ここで *WL_HOME* は WebLogic Server のインストール ディレクトリです。ノード マネージャは、このディレクトリを出力とログ ファイルを格納するための作業ディレクトリとして使用します。異なる作業ディレクトリを使用する場合は、テキスト エディタで起動スクリプトを編集し、NODEMGR_HOME 変数の値を目的のディレクトリに変更します。

また、ノード マネージャ起動引数がノード マネージャ プロセス用の適切なリスン アドレスとポート番号を設定するよう、サンプル起動スクリプトを編集する必要があります。

Windows サービスとしてのノード マネージャの起動

ディレクトリ *WL_HOME*\server\bin (*WL_HOME* は WebLogic プラットフォームの最上位ディレクトリ) には、ノード マネージャを Windows サービスとしてインストールおよびアンインストールするための 2 つのスクリプト、installNodeMgrSvc.cmd および uninstallNodeMgrSvc.cmd が存在します。

いずれかのスクリプトを使用するには、コマンド ラインでスクリプト名を呼び出します。installNodeMgrSvc.cmd は、ノード マネージャ アプリケーション用に NodeManager_localhost_5555 というデフォルト Windows サービスを作成します。前述のノード マネージャ起動スクリプトと同じように、スクリプトの実行前に installNodeMgrSvc.cmd を編集して、サービス名を変更するか、デフォルト以外の環境変数または起動引数を使用できます。

uninstallNodeMgrSvc.cmd は、デフォルトのノード マネージャ サービスをアンインストールします。installNodeMgrSvc.cmd を編集してリスン ポートまたはホスト名を変更した場合、uninstallNodeMgrSvc.cmd に対しても同じ変更を加えて適切なサービス名が削除されるようにしてください。

ノード マネージャの環境変数

ノード マネージャを起動する前に、いくつかの環境変数を設定しておく必要があります。ドメインの必須の環境変数をすべて設定する方法のひとつは、WebLogic Server に付属の環境設定スクリプトを実行することです (5-17 ページの「起動スクリプトを使用したノード マネージャの起動」および 5-18 ページの「Windows サービスとしてのノード マネージャの起動」を参照)。

コマンド ラインから直接ノード マネージャを起動する場合、次の表の情報に基づいて必須の環境変数を設定してください。

図 5-2 ノード マネージャの環境変数

環境変数	説明
JAVA_HOME	JAVA_HOME 環境変数では、必ず、ノード マネージャで使用する JDK をインストールしたルート ディレクトリを示すようにする。次に例を示す。 `set JAVA_HOME=c:\bea\jdk131` ノード マネージャには、WebLogic Server と同じ JDK のバージョンに関する必要条件がある。
WL_HOME	WL_HOME は、WebLogic Server インストール ディレクトリを指定する。次に例を示す。 `set WL_HOME=c:\bea\weblogic700`
PATH	PATH 環境変数には、WebLogic Server bin ディレクトリと Java 実行ファイルのパスを指定する必要がある。次に例を示す。 `set PATH=%WL_HOME%\server\bin;%JAVA_HOME%\bin;%PATH%`

LD_LIBRARY_PATH (UNIX のみ)	HP UX および Solaris システムの場合、LD_LIBRARY_PATH にネイティブ ノード マネージャ ライブラリのパスを指定する必要がある。次に、Solaris での例を示す。
	`LD_LIBRARY_PATH=$LD_LIBRARY_PATH:$WL_HOME/server/lib/solaris:$WL_HOME/server/lib/solaris/oci816_8`
	次に、HP UX での例を示す。
	`SHLIB_PATH=$SHLIB_PATH:$WL_HOME/server/lib/hpux11:$WL_HOME/server/lib/hpux11/oci816_8`

環境変数	説明
CLASSPATH	ノード マネージャ CLASSPATH は、ノード マネージャを起動するための java コマンド ラインのオプションか、または環境変数として設定できる。次に、環境変数としてクラスパスを設定する例を示す (Windows NT)。 `set CLASSPATH=.;%WL_HOME%\server\lib\weblogic_sp.jar;%WL_HOME%\server\lib\weblogic.jar`

ノード マネージャのコマンドライン引数

ノード マネージャを起動するための基本構文は次のとおりです。

```
java [java_property=value ...] -D[nodemanager_property=value]
-D[server_property=value] weblogic.nodemanager.NodeManager
```

このコマンドラインで、*java_property* は java 実行可能ファイルの直接引数 (-ms や -mx など) です。CLASSPATH 環境変数を設定しなかった場合、-classpath オプションを使用して必須のノード マネージャ クラスを指定します。

注意 : メモリ不足を回避するために、ノード マネージャを使用するときには常に最小ヒープ サイズを 32 MB (-Xms32m) に指定してください。WebLogic Server と一緒にインストールされるノード マネージャ起動スクリプトは、この Java 引数を自動的に設定します。

nodemanager_property はプレフィックス weblogic.property で始まり、ノード マネージャ プロセスの動作に直接影響を与えます。次の表に、すべてのノード マネージャ プロパティを示します。

表 5-1 ノード マネージャのプロパティ

ノード マネージャ プロパティ	説明	デフォルト値
weblogic.nodemanager.certificateFile	SSL 認証用の証明書ファイルのパスを指定する。	./config/democert.pem
weblogic.nodemanager.javaHome	ノード マネージャがこのマシンで管理対象サーバを起動するために使用する Java ホーム ディレクトリを指定する。	none
weblogic.nodemanager.keyFile	管理対象サーバとの SSL 通信に使用するプライベート キー ファイルのパス。	./config/demokey.pem
weblogic.nodemanager.keyPassword	キー ファイル内の暗号化されたプライベート キーにアクセスするためのパスワード。	password

ノード マネージャ プロパティ	説明	デフォルト値
weblogic.nodemanager.listenAddress (非推奨)	ノード マネージャが接続要求をリスンするアドレス。これは非推奨の引数なので、代わりに weblogic.ListenAddress を使用すること。	localhost
weblogic.nodemanager.listenPort (非推奨)	ノード マネージャが接続要求をリスンする TCP ポート番号。これは非推奨の引数なので、代わりに weblogic.ListenPort を使用すること。	5555
weblogic.ListenAddress	ノード マネージャが接続要求をリスンするアドレス。この引数は weblogic.nodemanager.listenAddress の代わりに使用する。	localhost
weblogic.ListenPort	ノード マネージャが接続要求をリスンする TCP ポート番号。この引数は weblogic.nodemanager.listenPort の代わりに使用する。	5555
weblogic.nodemanager.nativeVersionEnabled	Solaris または HP-UX 以外の UNIX システムでは、このプロパティを `false` に設定してノード マネージャを非ネイティブ モードで実行する。その結果ノード マネージャでは、管理対象サーバの起動に `StartTemplate` プロパティで指定されている起動スクリプトが使用されるようになる。	true
weblogic.nodemanager.reverseDnsEnabled	信頼性のあるホスト ファイルのエントリに (IP アドレスの代わりに) DNS 名を登録できるかどうかを指定する。	false

ノード マネージャ プロパティ	説明	デフォルト値
weblogic.nodemanager.savedLogsDirectory	ノード マネージャがログ ファイルを格納するパスを指定する。ノード マネージャは、`savedLogsDirectory` に `NodeManagerLogs` というサブディレクトリを作成する。	./NodeManagerLogs
weblogic.nodemanager.sslHostNameVerificationEnabled	ノード マネージャがホスト名検証を実行するかどうかを指定する。	false
weblogic.nodemanager.startTemplate	UNIX システムの場合、このプロパティはノード マネージャを起動するためのスクリプト ファイルのパスを指定する。 **注意：** ノードマネージャでは `NativeVersionEnabled` プロパティが `false` に設定されている場合にのみ、ここで指定されている起動スクリプトを使用して管理対象サーバを起動する。 `StartTemplate` は Unix システムでのみ使用される。 独自の起動テンプレート スクリプトを作成する場合は、ネイティブ ノードマネージャ ライブラリの例として `nodemenager.sh` を参照のこと。	./nodemanager.sh
weblogic.nodemanager.trustedHosts	ノード マネージャが使用する信頼性のあるホスト ファイルのパス。5-10 ページの「ノード マネージャ ホスト ファイルの設定」を参照。	./nodemanager.hosts

ノード マネージャ プロパティ	説明	デフォルト値
weblogic.nodemanager.weblogicHome	WebLogic Server のルート ディレクトリを指定する。これは、コンフィグレーションされたルート ディレクトリを持たないサーバ用の -Dweblogic.RootDirectory のデフォルト値として使用される。	なし

ノード マネージャは、新しい管理対象サーバ インスタンスを起動するときに *server_property* 値をデフォルト値として使用します。次の表に、すべての *server_property* 値を示します。

表 5-2 ノード マネージャで使用されるサーバ プロパティ

サーバ プロパティ	説明	デフォルト値
bea.home	このマシンの管理対象サーバが使用する BEA ホーム ディレクトリを指定する。	なし
java.security.policy	管理対象サーバが使用するセキュリティ ポリシー ファイルのパスを指定する。	なし
weblogic.security.SSL.trustedCAKeyStore	信頼性のある認証局の証明書が格納されているキーストアのパス。	java.security.keyStore

注意： Solaris または HP UX 以外の UNIX オペレーティング システム上でノード マネージャを起動する場合、`java` コマンド ラインに渡すパラメータでスペースを使用することはできません。たとえば、次のようなパラメータを使用するとします。

```
-Dweblogic.Name=big iron
```

この場合、`big iron` にスペースが含まれているので、このコマンドは無効です。

ノード マネージャを使用した管理対象サーバの起動と停止

以下の節では、それぞれの目的でのノード マネージャの使い方を説明します。

- 5-25 ページの「管理対象サーバの手動による起動と停止」
- 5-26 ページの「ドメインまたはクラスタ内のすべての管理対象サーバの起動と停止」
- 5-27 ページの「weblogic.Admin を使用したサーバの起動と停止」

管理対象サーバの手動による起動と停止

管理対象サーバのホスト マシンでノード マネージャが動作している場合には、Administration Console を使用して管理対象サーバを起動できます。

この方法で管理対象サーバを起動する場合、ノード マネージャは独立したプロセスでサーバ インスタンスを起動します。WebLogic Server の起動時に通常 STDOUT または STDERROR に出力されるメッセージは、Administration Console の右ペインに表示されます。それらのメッセージは、そのサーバのノード マネージャ ログ ファイルにも書き込まれます (5-27 ページの「管理対象サーバのログ ファイル」を参照)。

ノード マネージャは、常に管理対象サーバをその最後の実行時状態で起動し、そのサーバのコンフィグレーション済み起動モードを無視します。詳細については、5-8 ページの「ノード マネージャはサーバの起動モードを無視する」を参照してください。

ノード マネージャを使用して管理対象サーバを起動するには、次の手順に従います。

1. Administration Console を起動します (まだ動作していない場合)。
2. [サーバ] ノードを選択し、起動または停止するコンフィグレーション済みサーバの名前を選択します。
3. 右ペインで [制御 | 起動 / 停止] タブを選択します。

4. 次のいずれかのオプションを選択して、ノード マネージャを使用してサーバを起動または停止します。
   - [このサーバを起動 ...]。ノード マネージャを使用して管理対象サーバを起動します。
   - [このサーバをスタンバイ モードで起動 ...]。サーバをサスペンド状態で起動します。

   注意： [このサーバを停止 ...] または [このサーバを強制的に停止 ...] オプションを使用する場合、ノード マネージャは必要ありません。ただし、サーバが停止要求に応答できない場合は、ノード マネージャを使用してこれらの停止タスクを完了します。

   サーバの状態の詳細については、『管理者ガイド』の「WebLogic Server の起動と停止」を参照してください。

   注意： 管理対象サーバではなく管理サーバを選択した場合、[このサーバを停止 ...] オプションしか選択できません。

5. [はい] または [いいえ] をクリックして選択を確認します。

ドメインまたはクラスタ内のすべての管理対象サーバの起動と停止

ドメインまたはクラスタ内の管理対象サーバをホストする各マシンでノード マネージャが動作している場合は、Administration Console を使用してドメインまたはクラスタ内のすべての管理対象サーバを起動または停止できます。

ドメインまたはクラスタ内のすべての管理対象サーバを起動するには、次の手順に従います。

1. 左ペインでアクティブなドメインの名前を右クリックします。
2. **[このドメインを強制停止 ...]** または **[このドメインを開始 ...]** を選択します。

Administration Console からドメイン全体を起動した場合、右ペインに表示される結果はそのドメインにコンフィグレーションされた各管理対象サーバの結果への一連のリンクになります。

ドメインまたはクラスタ内のすべての管理対象サーバを停止するには、次の手順に従います。

1. 左ペインでクラスタの名前を右クリックします。
2. **[このクラスタを強制停止 ...]** または **[このクラスタを開始 ...]** を選択します。

注意： ノード マネージャを使用して管理サーバを起動または強制停止することはできません。

weblogic.Admin を使用したサーバの起動と停止

`weblogic.Admin` ユーティリティで `START`、`SHUTDOWN`、および `FORCESHUTDOWN` コマンドを使用して、管理対象サーバを起動および停止できます。管理対象サーバを `START` で起動するには、ノード マネージャがホストで動作していなければなりません。`SHUTDOWN` または `FORCESHUTDOWN` の場合、ノード マネージャは必要ありません。ただし、管理対象サーバが停止要求を正常に完了できない場合は、ノード マネージャがこの処理を実行します。

`weblogic.Admin` の使い方については、『管理者ガイド』の「WebLogic Server コマンドライン インタフェース リファレンス」を参照してください。

ノード マネージャのトラブルシューティング

以下の節では、ノード マネージャの問題を診断および解決する方法について説明します。個々の管理対象サーバの起動または停止に関する問題を解決するには、ノード マネージャのログ ファイルを利用します。ノード マネージャのコンフィグレーションおよび設定に関する問題を解決するには、5-29 ページの「一般的な問題の修正」で説明する手順に従ってください。

管理対象サーバのログ ファイル

注意： ディスク スペースを解放するために、ノード マネージャの起動および停止の試行によって蓄積されたログ ファイルを定期的に削除する必要があります。

WebLogic Server を起動するときには、起動またはエラー メッセージが STDOUT または STDERROR に出力されます。これらのメッセージを取得するには、Administration Console の左ペインで対象サーバを右クリックして、[**サーバ ログを見る**] オプションを選択します。

ノード マネージャは、これらのメッセージを NodeManagerLogs ディレクトリに保存します。デフォルトでは、NodeManagerLogs ディレクトリはノード マネージャを起動するディレクトリに作成されます。ノード マネージャによって起動される管理対象サーバごとに別々のログ ファイル サブディレクトリが作成されます。各サブディレクトリには、*domain_server* という命名規約が適用されます。これにはそれぞれドメイン名と管理対象サーバ名が指定されます。

サーバ ディレクトリに格納されるログ ファイルは次のとおりです。

*servername*_pid
: *servername* という名前の管理対象サーバのプロセス ID を保存します。管理サーバによって要求されたときにサーバ プロセスを強制停止するためにノード マネージャが使用します。

config.xml
: 管理対象サーバを起動するときに管理サーバからノード マネージャに渡される起動コンフィグレーション情報を保存します。

*servername*_output.log
: ノード マネージャが管理対象サーバを起動しようとするときに生成されるノード マネージャ起動メッセージを保存します。サーバの起動が新たに試行されると、このファイルは _PREV を付け加えることによって名前変更されます。

*servername*_error.log
: ノード マネージャが管理対象サーバを起動しようとするときに生成されるノード マネージャ エラー メッセージを保存します。サーバの起動が新たに試行されると、このファイルは _PREV を付け加えることによって名前変更されます。

また、ドメインの管理サーバも、管理対象サーバのログ ファイルのコピーを NodeManagerClientLogs というディレクトリに格納します。このディレクトリは、管理サーバのルート ディレクトリより 1 レベル上のディレクトリに作成されます (デフォルトはサーバを起動したディレクトリ)。
NodeManagerClientLogs ディレクトリには、ノード マネージャを通じて起動が試行された管理対象サーバごとにサブディレクトリが存在します。それらのサブ

ディレクトリの各ログは、サーバ プロセスの起動や強制停止といった処理の試行に対応します。ログ ファイルの名前には、アクションが試行された時刻を示すタイムスタンプが含まれます。

Administration Console では、サーバの標準出力とエラー メッセージ、および特定の管理対象サーバに対するノード マネージャのログ メッセージを参照できます。そのためには、ノード マネージャで起動したサーバを選択し、[モニタ | プロセス出力] タブを選択します。

ノード マネージャのログ ファイル

注意： ノード マネージャは起動するたびに (新しいタイムスタンプで) 新しいログ ファイルを作成するため、`NodeManagerLogs` サブディレクトリを定期的に削除して古いログ ファイルによって占有されているスペースを解放する必要があります。

また、WebLogic Server 7.0 のノード マネージャは独自のログ ファイルを生成します。このファイルには、ノード マネージャの旧バージョンでは `STDOUT` に出力されていたノード マネージャの起動およびステータス メッセージが記録されます。ノード マネージャのログ ファイルは、`NodeManagerLogs\NodeManagerInternal` サブディレクトリに置かれます。これらのログ ファイルには、`NodeManagerInternal_`*`timestamp`* という命名規約が適用されます。ここで *`timestamp`* はノード マネージャが起動した時間です。

一般的な問題の修正

次の表に、一般的なノード マネージャの問題とその解決方法を示します。

表 5-3 ノード マネージャのトラブルシューティング

症状	説明
エラー メッセージ：`Could not start server 'MyServer' via Node Manager - reason: 'Target machine configuration not found'.`	管理対象サーバをマシンに割り当てていない。5-12 ページの「ノード マネージャを使用するためのマシンのコンフィグレーション」の手順に従うこと。
エラー メッセージ：`<SecureSocketListener: Could not setup context and create a secure socket on 172.17.13.26:7001>`	ノード マネージャ プロセスが指定されたマシン上で実行されていない可能性がある。5-17 ページの「ノード マネージャの起動」を参照。
サーバの自動状態モニタ属性をコンフィグレーションしたが、ノード マネージャがサーバを自動的に再起動しない。	サーバを自動的に再起動するには、自動状態モニタ属性のほかにサーバの自動再起動属性をコンフィグレーションする必要がある。5-15 ページの「管理対象サーバのモニタ、停止、および再起動のコンフィグレーション」と 5-17 ページの「ノード マネージャの起動」を参照。 また、ノード マネージャを使用して管理対象サーバを起動する必要がある。ノード マネージャの外部で起動した管理対象サーバ (コマンド ラインで直接起動したサーバなど) を自動的に再起動することはできない。

症状	説明

管理対象サーバ上のアプリケーションが参照用に間違ったディレクトリを使用している。	WebLogic Server にデプロイされるアプリケーションでは、現在の作業ディレクトリを決めてかからないようにします。ファイルのルックアップは、通常は、`ServerMBean.getRootDirectory()` メソッドで取得したルート ディレクトリを基準に行います (デフォルトは「.」ディレクトリ)。たとえば、ファイルのルックアップを実行するには、次のようなコードを使用します。
	`String rootDir = ServerMBean.getRootDirectory(); //application root directory File f = new File(rootDir + File.separator + "foo.in");`
	次のようなシンプルなコードは使用しません。
	`File f = new File("foo.in");`
	ノード マネージャを使用して起動されるサーバにアプリケーションがデプロイされる場合は、代わりに次のメソッド呼び出しを使用します。
	`String rootDir // アプリケーションのルート ディレクトリ if ((rootDir = ServerMBean.getRootDirectory()) == null) rootDir = ServerStartMBean.getRootDirectory(); File f = new File(rootDir + File.separator + "foo.in");`
	`ServerStartMBean.getRootDirectory()` メソッドは、ノード マネージャを使用して起動するようにサーバをコンフィグレーションする際に指定した Root Directory 値を取得します。これは、Administration Console の [コンフィグレーション
		リモートスタート] ページで指定された Root Directory 属性に対応します。

ノード マネージャの内部ステート

ノード マネージャでは、サーバの再起動時に使用するために、管理対象サーバについて独自の内部的な状態が定義されています。ドメインでノード マネージャを使うときは、これらの状態を観察できます。

- FAILED_RESTARTING— 障害が発生した管理対象サーバを再起動しています。
- ACTIVATE_LATER—現在の RestartInterval において MaxRestart 回の再起動を試みており、次の RestartInterval でさらに再起動を試みます。
- FAILED_NOT_RESTARTABLE— サーバで障害が発生しましたが、サーバの AutoRestart 属性または AutoKillIfFailed 属性が False に設定されているためサーバを再起動できません。
- FAILED_NOT_RESTARTABLE— サーバで障害が発生しましたが、サーバの HostsMigratableServices 属性が False に設定されているためサーバを再起動できません。

6 WebLogic Sever ドメインのモニタ

以下の節では、WebLogic Server ドメインをモニタする方法について説明します。

モニタの概要

WebLogic Server ドメインの状態とパフォーマンスをモニタするためのツールは Administration Console です。Administration Console では、サーバ、HTTP、JTA サブシステム、JNDI、セキュリティ、CORBA 接続プール、EJB、JDBC、JMS といった WebLogic Server リソースのステータスと統計を表示できます。

モニタ情報は、Administration Console の右ペインに表示されます。ページにアクセスするには、左ペインの階層的なドメイン ツリーでコンテナまたはサブシステム、あるいはコンテナの下の特定のエンティティを選択します。

Administration Console には、モニタ情報を表示する以下の 3 種類のページがあります。

- 特定のエンティティ (JDBC 接続プールのインスタンスや特定のサーバのパフォーマンスなど) のモニタ タブ ページ。
- 特定の種類のすべてのエンティティに関するデータのテーブル (WebLogic Server テーブルなど)。

- ドメイン ログおよびローカル サーバ ログのビュー。ログ メッセージの詳細については、「ログ メッセージを使用した WebLogic Server の管理」を参照してください。

Administration Console では、ドメイン リソースについての情報が管理サーバから取得されます。管理サーバでは、Sun の Java Management Extension (JMX) 規格に基づく Management Bean (MBean) が使用されます。JMX 規格は、管理を目的としてドメイン リソースにアクセスする方法を定めています。

管理サーバには、ドメインのコンフィグレーションを管理するコンフィグレーション MBean と実行時 MBean があります。実行時 MBean では、JVM のメモリ使用率や WebLogic Server のステータスといったドメイン リソースに関する特定の時点での情報が提供されます。ドメインの特定のリソース (Web アプリケーションなど) がインスタンス化されると、その特定のリソースについての情報を収集する MBean のインスタンスが生成されます。

Administration Console で特定のリソースのモニタ ページにアクセスすると、管理サーバでは現在の属性値を取り出すための GET 処理が実行されます。

以降の節では、WebLogic Server ドメインの管理に便利なモニタ ページをいくつか選んで説明します。それらのページは、ここでは、Administration Console の機能説明を目的として取り上げています。

サーバのモニタ

サーバ テーブルおよび個別サーバのモニタ タブ ページでは、WebLogic Server をモニタできます。サーバ テーブルでは、ドメイン内のすべてのサーバのステータスが簡潔に表示されます。ログ メッセージの一部しかサーバからドメイン ログへ転送されない場合は、ローカル サーバ ログにアクセスすると、トラブルシューティングやイベントの調査に便利です。

ログ ファイルとロギング サブシステムの詳細については、「ログ メッセージを使用した WebLogic Server の管理」を参照してください。

各 WebLogic サーバのモニタ データには、そのサーバのモニタ タブからアクセスできます。ロギング タブからは、サーバのローカル ログ (サーバが稼働しているマシン上のログ) にアクセスできます。

[モニタ | 一般] タブ ページでは、現在の状態とアクティブ化時刻が表示され、アクティブ キュー テーブル、接続テーブル、およびアクティブ ソケット テーブルにアクセスできます。アクティブ実行キュー テーブルは、保留中の最も古い要求や、キューのスループットといったパフォーマンス情報を提供します。

パフォーマンス

[モニタ | パフォーマンス] タブは、JVM メモリ ヒープの使用率、要求スループット、およびキューの長さに関するリアルタイム データをグラフで示します。このタブ ページでは、メモリ ヒープでのガベージ コレクション実行を JVM に強制することもできます。

Java ヒープは、ライブ Java オブジェクトおよびデッド Java オブジェクトのリポジトリです。通常は、ガベージ コレクションを手動で実行する必要はなく、JVM で自動的に行われます。JVM でメモリが不足し始めると、すべての実行が停止され、ガベージ コレクション アルゴリズムを使用して Java アプリケーションで使用されなくなったスペースが解放されます。

その一方で、アプリケーションをデバッグする開発者には、ガベージ コレクションを手動で強制しなければならない場合もあります。手動のガベージ コレクションは、たとえば JVM メモリを急速に消費するメモリ リークをテストする場合に便利です。

セキュリティ

[モニタ | セキュリティ] タブでは、不正なログインの試行およびロックされているユーザとロックが解除されているユーザについての統計が表示されます。

JMS

[モニタ | JMS] タブでは、JMS サーバおよび接続に関する統計が表示されます。また、このページは、アクティブな JMS 接続とアクティブな JMS サーバのテーブルへのリンクも提供します。これらは、現在のセッション総数などの属性をモニタします。

JTA

[モニタ | JTA] タブでは、トランザクション総数やロールバック総数などの Java トランザクション サブシステムに関する統計が表示されます。このページは、リソースと名前によってリストされるトランザクションのテーブルと、実行中のトランザクションのテーブルへのリンクを提供します。

サーバの自動状態モニタ

WebLogic Server 7.0 には、ドメイン内のサーバの信頼性と可用性を向上させるための自動状態モニタ機能が用意されています。各 WebLogic Server 内の選択されたサブシステムは、そのサブシステムに固有の条件に基づいて自身の状態をモニタします。たとえば、JMS サブシステムは JMS スレッド プールの状態をモニタし、コア サーバ サブシステムはデフォルトおよびユーザ定義の実行キュー統計をモニタします。個々のサブシステムは、整合性および信頼性のある状態で動作できないと判断した場合、ホスト サーバに自己の状態を「障害」として登録します。

また、各 WebLogic Server は登録されているすべてのサブシステムの状態をチェックして、サーバの全体的な有効性を調べます。1 つまたは複数の重要なサブシステムが「障害」状態に達していることを発見した場合、サーバは自己の状態を「障害」に設定して、アプリケーションを適切にホストできないことを示します。

ノード マネージャ アプリケーションと一緒に使用すると、サーバ自動状態モニタでは障害が発生したサーバを自動的に再起動します。これにより、ドメインの全体的な信頼性が向上し、管理者の介入が不要になります。詳細については、5-1 ページの「ノード マネージャによるサーバの可用性の管理」を参照してください。

WebLogic Server バージョン 7.0 では、サーバとその登録済みサブシステムの状態は Administration Console では参照できません。しかし、`ServerRuntimeMBean` の `getHealthState()` メソッドを呼び出すことによって、サーバの状態をプログラム的にチェックできます。同様に、その MBean の `getHealthState()` メソッドを呼び出すことによって、登録済み WebLogic Server サブシステムの状態を取得できます。WebLogic Server バージョン 7.0 では、次の MBean がその状態をホスト サーバに自動的に登録します。

- JMSRuntimeMBean
- JMSServerRuntimeMBean
- JTARuntimeMBean
- TransactionResourceRuntimeMBean

個々の MBean の詳細については、WebLogic クラスの Javadoc を参照してください。

また、ServerMBean の属性を設定することによって、自動状態チェックの頻度とタイミングをコンフィグレーションできます。これらの属性の Administration Console での設定については、5-1 ページの「ノード マネージャによるサーバの可用性の管理」を参照してください。

JDBC 接続プール のモニタ

Java Database Connectivity (JDBC) サブシステムのリソースは、Administration Console を使用してモニタできます。JDBC 接続プールの [モニタ] タブを使用すると、そのプールのインスタンスに関する統計を示す表にアクセスできます。Administration Console の他のエンティティ テーブルと同様に、テーブルをカスタマイズして表示する属性を選択できます。

それらの属性は、クライアントのデータベース アクセスを管理するための重要な情報を提供します。

[最大待ち] フィールドは、一度に接続を待つクライアントの最大数を示します。[待ち] フィールドは、現在接続を待機中のクライアント数を示します。[最大接続数] フィールドは、一度に発生した接続の最大数を示します。[最大待ち時間 (秒)] フィールドは、クライアントがデータベース接続を待つ最長時間を示します。これらの属性から、クライアント要求への応答に関して、現在のコンフィグレーションの効果を判断できます。

[最大接続数] フィールドの値が [最大容量] フィールドの値 ([コンフィグレーション | 接続] タブで設定) に近い場合は、[最大容量] (同時接続の最大数) の値を増やすことを検討することがあります。[最大待ち] フィールドの値がクライアントがデータベース アクセスを長時間待たなければならないことを示す場合、プールのサイズを増やすことがあります。

[縮小間隔] フィールドの値は、プールが最大のサイズから縮小するまでに JDBC サブシステムが待つ時間です。サブシステムがプールを縮小するとき、データベース接続は破棄されます。データベース接続を作成すると、リソースが消費され、時間もかかります。システムでクライアント要求の発生が断続的に集中する場合、縮小間隔が短いと、データベース接続が絶えず再作成されパフォーマンスが低下することがあります。