Forms Server for Windows and UNIX

Forms アプリケーション Web 利用ガイド

リリース 6*i*

2000 年 1 月

部品番号 : J00496-01

ORACLE®

Forms Server for Windows and UNIX Forms アプリケーション Web 利用ガイド , リリース 6*i*

部品番号 : J00496-01

原本名：Forms Server Release 6*i* Deploying Forms Applications to the Web with Forms

原本部品番号：A73071-01

原本著者：Tony Wolfram, Cathy Godwin

原本協力者：Joan Carter

目次

3 インストールおよび構成時の選択肢の概説

4 クイック・スタート構成のステップ

5 Forms Server の構成

6 Web へのフォームの配置

7 アプリケーション設計に関する考慮事項

8 これまでのクライアント・サーバー・アプリケーションの資産の Web への移行

9 ネットワークに関する考慮事項

12 ロード・バランスに関する考慮事項

13 Oracle Enterprise Manager Forms のサポート

14 キャパシティ量計画の考慮事項

B Oracle JInitiator

C AppletViewer

D Oracle Installer リファレンス

E カートリッジ・サポートを使用した Forms Server の構成

F Graphics Server

第 III 部

索引

はじめに

Forms Server for Windows and UNIX Forms アプリケーション Web 利用ガイド、リリース 6*i*

対象読者

このマニュアルは、Forms アプリケーションの Web への配置に関心のあるソフトウェア開発者を対象にしています。

構成

このマニュアルには、次の章と付録が含まれています。

第 1 章　概要

アプリケーションを Web に配置する利点について説明します。

第 2 章　Forms Server の概要

Forms Server のアーキテクチャとそのコンポーネントの概要を説明することで、使用する配置ツールを紹介します。

第 3 章　インストールおよび構成時の選択肢の概説

アプリケーションを Web に配置するときのインストール・オプションと構成の選択肢を概説します。

第 4 章　クイック・スタート構成のステップ

初心者の方を対象としています。クイック・スタート構成ステップを紹介します。Forms Server の自動サーバー構成機能について説明しますので、Forms Server を自動的に構成する際の参考にしてください。

第 5 章　Forms Server の構成

アドバンスド・ユーザーを対象としています。Forms Server をサポートするためにネットワーク環境を手動で構成する際、必要となるステップについて説明します。第 4 章の自動構成ステップは実行しないことを想定します。

第 6 章　Web へのフォームの配置

実行可能ファイルの作成やアプリケーションの URL のブロードキャストなど、アプリケーションを Web に配置するために必要なステップについて説明します。これらのステップは、第 4 章または第 5 章のステップを完了した後に実行する必要があります。

第 7 章　アプリケーション設計に関する考慮事項

Web へ配置する Forms アプリケーションを設計するためのガイドラインとヒント、および機能制限が記載されています。

第 8 章　これまでのクライアント・サーバー・アプリケーションの資産の Web への移行

現在のアプリケーションを、クライアント・サーバー・ベースから Web ベースの Forms Server のインプリメンテーションに移行する際のガイドラインが記載されています。

第 9 章　ネットワークに関する考慮事項

Web アプリケーションを配置できるネットワーク・インプリメンテーションと、各タイプに Web アプリケーションを配置するときに考慮する必要のある事項について説明します。

第 10 章　セキュリティに関する考慮事項

Forms Server をネットワーク環境で設定するときに考慮する必要のある一般的なセキュリティ問題について説明します。

第 11 章　パフォーマンス・チューニングに関する考慮事項

Forms Server を使用してインターネットまたはその他のネットワーク環境にアプリケーションを配置するときの、チューニングに関する考慮事項を説明します。

第 12 章　ロード・バランスに関する考慮事項

CGI ベースのロード・バランスを使用してロード・バランスする技法について説明します。

第 13 章　Oracle Enterprise Manager Forms の サポート

Oracle Enterprise Manager（OEM）システム管理ツールについて説明します。

第 14 章	キャパシティ量計画の考慮事項 Forms Server の拡張性機能について概観します。
第 15 章	トラブルシューティング・ソリューション Forms Server のトラブルシューティング・ソリューションに関する情報が記載されています。
付録 A	Forms Server パラメータ Forms および Graphics の構成に使用するパラメータについて説明します。
付録 B	Oracle JInitiator Oracle JInitiator をユーザーの Web ブラウザのプラグインとして使用する利点を説明します。
付録 C	AppletViewer Forms Server 上で実行しているアプリケーションを参照するために、Oracle JInitiator のかわりに使用する AppletViewer について説明します。
付録 D	Oracle Installer リファレンス Forms Server の設定時に提供される各インストール・オプションについて説明します。
付録 E	カートリッジ・サポートを使用した Forms Server の構成 Oracle Application Server（OAS）とカートリッジの実装に関する情報が記載されています。
付録 F	Graphics Server Graphics Server を手動で構成するために必要なステップについて説明します。

関連マニュアル

詳細は、次のマニュアルを参照してください。

- 『Oracle Forms Developer リリースノート、リリース 6*i*』
- 『Oracle Forms Developer for Windows/NT スタート・ガイド』
- 『Oracle Reports Developer パブリッシング・レポート』
- 『Oracle Forms Developer and Oracle Reports Developer アプリケーション作成ガイド』
- 『Oracle Forms Developer Form Builder リファレンス』
- 『Oracle Forms Developer Graphics Builder リファレンス』

第Ⅰ部

1

概要

1.1 インターネットがすべてを変える

従来のような不動産、建設および流通のコストを必要としない、製品および情報の巨大な仮想商店街の発達に伴い、インターネットがビジネスおよびこれをサポートする基盤技術に革命を起こしています。

1.1.1 ビジネスにおける変革

インターネットは新しいビジネス・チャンスを生み出し、日常活動の行動様式を変えています。たとえば、ショッピング、行先案内、銀行口座および証券口座の管理、電話番号および旧友の連絡先の追跡、ニュースおよび情報の獲得などです。

企業は多くの内部処理をイントラネットに移行しています。イントラネットでは、URL を公開するだけで最新の情報を簡単に収集し、配置できます。旅行代理店および航空会社などの他企業との関連が強い企業は、時間を要する労働集約型の電話モデルから、エクストラネットを使用した低コストで効率的なセルフサービス・モデルに移行しています。企業は Web を使用して、低コストで他社および顧客と直接連絡を取り合うことにより、応答性および収益性を向上させています。

インターネットは商機を拡大し、コストを削減し、かつ有効性の高い方法でビジネス・プロセスを改善し、顧客を引きつけて離さない新しい方法を提供します。

1.1.2 基盤技術における変革

比較的少ない経費で情報およびサービスを世界中のユーザーに配信できるというビジネス上の大きな利点の他に、Web アプリケーション開発および配置には多くの利点があります。これらの利点を次に示します。

- **新規バージョンの配置は簡単、迅速かつ安価です。**Web アプリケーションを配置する場合、ユーザーにアプリケーションの URL を公開します。この配置メソッドでは、各ユーザーのデスクトップ・マシン上にアプリケーション・ソフトをインストールする必要がなくなるため、各地に分散した多くのユーザーにアプリケーションを配布するための時間、コストおよび複雑さが低減されます。

- **集中化された配布は、システム所有の総コストが低減されることを意味します。**Web への配置は、情報へのアクセス可能性を向上すると同時に、管理、メンテナンスおよびネットワークのコストを大幅に削減します。複数の場所からシステム管理サポートを提供するかわりに、システム・メンテナンスおよび管理は 1 つのセントラル・ロケーションから実行されます。Web への配置により、アプリケーションの複雑さは各ユーザーのデスクトップから消えて、集中的に配置され専門的に管理されたアプリケーション・サーバーで処理されます。これにより、少数のサーバー上のサイトを専門的に管理できるため、メンテナンス作業の単純化、高速化および標準化が行えます。したがって、コストを大幅に低減できます。
- **業界標準に準拠した開発は優れた統合性を意味します。**インターネット・アプリケーションの開発は同一の業界標準（Java、Enterprise JavaBeans、HTML、XML、CORBA、HTTP など）に準拠しています。共通言語は、新規にまたは個別に開発されたアプリケーションの容易かつ迅速な統合を意味します。
- **コンポーネントベースの開発は、生産性が向上し、メンテナンスが容易になり、再利用が可能になることを意味します。**各ユーザーの各リクエストに応じて、アプリケーションを迅速にカスタマイズします。企業の開発者は関連するコンポーネントのみを変更し、アプリケーション全体を変更する必要はありません。通常の方法で変更されたコンポーネントは他のアプリケーションで簡単に再利用できます。これらは、組織が "Web 時間 " 内でユーザーのリクエストに応えるための方法の一部です。

1.2 Oracle インターネット・プラットフォーム

標準のクライアント / サーバー・アーキテクチャには次の 2 層が含まれています。アプリケーションを管理するクライアント層、およびアプリケーションが処理するエンタープライズ・データを管理するサーバー層です。これに対して、Oracle インターネット・プラットフォームには次の 3 層が含まれています。

- データベース層 エンタープライズ・データが格納されます。
- サーバー層 アプリケーションを管理し、Web サポートを含む他の多くのサービスを提供します。
- クライアント層 アプリケーションが表示されるブラウザを管理します。

Oracle インターネット・プラットフォームでは、アプリケーションの複雑さを専門的に管理された少数のサーバーに集中させ、情報をユーザーに配信します。これは、データベース・サーバー、アプリケーション・サーバー、開発ツールおよび管理ツールで構成されます。これは、Web 開発、配置および e- ビジネス・ソリューションのメンテナンス用の統合プラットフォームです。これは、Java、Enterprise JavaBeans、CORBA、HTML および XML を含むオープン・インターネット標準インタフェースおよびプロトコルに準拠しています。

Oracle インターネット・プラットフォームは標準インターネット・ブラウザを使用して、テキスト、イメージ、Web ページ、ビデオ、サウンドおよび電子メールを含むあらゆるタイプの情報の内容を管理および処理します。

テラバイトのデータおよび数百万人のユーザーが関わる、Oracle インターネット・プラットフォーム上で作成された Web ベースのアプリケーションでは、信頼性が 24 時間、1 年中提供され、データの暗号化および整合性のための優れたセキュリティ標準が組み込まれています。

アプリケーションの配布および管理が単純化されているため、Oracle インターネット・プラットフォームは非常に低コストの配置プラットフォームです。例 :

- サーバー・スケーラビリティにより、コストを低減し管理しやすくするためにサーバーの台数を減らすことが可能です。
- サーバー側にアプリケーションおよびデータ処理を配置することで、ネットワークの使用率を適正に保ちます。
- クライアント側では、必要なものはブラウザのみです。データベースに接続されているデスクトップで必要となる追加ソフトウェアのコストは不要です。

1.3 Forms Server

Forms Server は、Oracle インターネット・プラットフォームの重要なコンポーネントです。アプリケーションサーバーは、Oracle Forms アプリケーションを多層環境に配置するために最適化されています。また、インターネットベースの Forms アプリケーションが自動的にスケーリングされ、あらゆるネットワークを介して実行されるように、アプリケーション・インフラストラクチャおよびイベント・モデルを実現します。Forms Server によって提供されるビルトイン・サービスには、トランザクション管理、レコード・キャッシュ、レコードのロック、例外処理およびロード・バランシングが含まれます。企業の開発者は、これらの共有サービスを構築するために具体的な処理を記述するコードをインプリメントする必要はありません。Forms Server は、自身のエンジンの一部としてこれらのサービスを自動的に配置します。

アプリケーションは、インターネット、イントラネットおよびエクストラネット展開のためのネットワーク・トラフィックを低減するように最適化されています。企業は Forms Server を使用する場合、システム・アーキテクチャおよびシステム要件に応じて、Java クライアントとサーバーの間のソケットベースの TCP/IP 通信でシステムを構築するか、Java クライアントとサーバーの間の HTTP 1.1（ファイアウォールをサポートしている）でシステムを構築するかを選択できます。

データベース・サーバーとの緊密な統合は、特に配列処理およびストアド・プロシージャを用いた場合、Forms Server エンジンとデータベースの間の通信がスムーズで信頼性が高いことを意味します。

Forms Server を使用する場合、他にも多くの利点があります。次にその利点の一部を紹介します。

- **拡張可能な最適化 Java クライアント。**企業の開発者は、Forms アプリケーションに JavaBeans を組み込んだり、Java クラスを再利用したりすることができます。これにより、クライアントの Java アプレットが拡張され、企業の開発者は高機能のユーザー・

インタフェースを作成できます。これらのインタフェースは Java 言語の長所を利用しており、既存の Java コンポーネントを再利用できます。

- **あらゆるネットワークで自動スケーラビリティを実現。**Forms Server では初めから、ロード・バランシング機能が提供されます。ロード・バランシングはクライアントのリクエストを、使用可能なシステム・リソース全体に効率的に配置します。このアプリケーションでは、どの Web サーバーを使用しても、HTTP リスナーと同じレベルのスケーラビリティが提供されます。これは、企業のイントラネット、エクストラネットおよびインターネットでの構築用に最適化されています。このアプリケーションは、LAN、WAN およびダイアルアップ・ネットワーク・アーキテクチャで使用できます。
- **ビルトイン最適化により高性能を実現。**Forms Server では、標準の 3 層アーキテクチャの次の 2 つの主要な制約に対応するため、多くのビルトイン最適化が行われています。ネットワーク帯域幅およびクライアントとアプリケーション・サーバー間の待ち時間。

  このアプリケーションは、先進的なアルゴリズムを使用してデータ・ストリームを高度に圧縮することにより、ネットワーク帯域幅を縮小します。

  Forms Server が待ち時間を短縮するために、次のようにイベント・バンドルを使用する方法があります。ユーザーが項目 A から項目 B にナビゲートする場合（1 つのエントリ・フィールドから別のエントリ・フィールドに移動するためにタブを選択する場合など）、事前トリガーおよび事後トリガーの範囲が起動される場合があります。これらは Forms Server で処理される必要があります。イベント・バンドルは、2 つのオブジェクト間のナビゲート中にトリガーされたすべてのイベントを " 収集し "、これらを 1 つの処理用パケットとしてサーバーに配置します。ナビゲーション時に多くのオブジェクトに問い合せる必要がある場合（離れたオブジェクトをマウスでクリックした場合など）、イベント・バンドルは問い合されたすべてのオブジェクトからすべてのイベントを収集し、これらを 1 つのネットワーク・メッセージとして Forms Server に配置します。
- **生産性が高い宣言型 Rapid Application Development（RAD）ツールとの統合。**Forms Server は、Oracle Forms アプリケーション用として開発されました。したがって、異なるベンダーのツールで作成されたアプリケーションおよびサーバーを統合する際に生じる時間を要さないため、開発の後速やかに配置できます。

1.4 このマニュアルの使用方法

アプリケーションをインターネット上に配置することを選択した場合、その実現方法に関して多くのことを決定する必要があります。このマニュアルでは、これらの決定に関する情報を提供します。また、アプリケーションを Web に配布するシステムを構成するための提案およびメソッドを提供します。

弊社は次のものを提供します。

- Forms Server アーキテクチャの概要
- アプリケーションを Web に配置する際に行うインストールおよび構成の選択の概要

- さまざまな Web への配置シナリオで Forms Server をインストールおよび構成するためのガイド
- これまでのクライアント・サーバー・アプリケーション資産の Web への移行に関する項
- 増大する作業負荷に対応するために機能し相互通信する複数のサーバーをセットアップする際に役立つ、キャパシティ計画およびロード・バランシングを行うための項
- ネットワークおよびセキュリティ考慮事項に関する項
- Web アプリケーションのアプリケーション設計上の考慮点およびパフォーマンスを最適化する際のパフォーマンス・チューニングに関する項

2

Forms Server の概要

2.1 概要

Forms Server は、インターネット・コンピューティングの利点を最大限に実現するテクノロジを提供します。この章では Forms Server アーキテクチャの概要から、特にフォームのインターネット上での配信に関する概要を提供します。

Forms Server は、新規および従来の Oracle Forms アプリケーションを、World Wide Web 上に配置できるようにする新世代の開発ツールです。アプリケーションは、社内イントラネットや社外のエクストラネットまたはインターネット上に配置できます。

Forms Server は、Oracle Forms アプリケーションを多層環境に配置するために最適化されたアプリケーションサーバーです。Oracle Developer Server では、Web の使用やアクセスが簡単なことを利用し、Web に単なる静的な情報公開メカニズムを超えた、複雑で動的なアプリケーションをサポートできる環境としての機能を与えます。

2.2 Forms Server アーキテクチャ

Forms Server では 3 層のアーキテクチャを使用して、データベース・アプリケーションを配置します。図 2-1 は、Forms Server アーキテクチャを構成する 3 層を示します。

- **クライアント層**には、アプリケーションを表示し使用する Web ブラウザが含まれます。
- **中間層**は、アプリケーション・ロジックとサーバー・ソフトウェアが格納されるアプリケーション・サーバーです。
- **データベース層**は、企業データが格納されるデータベース・サーバーです。

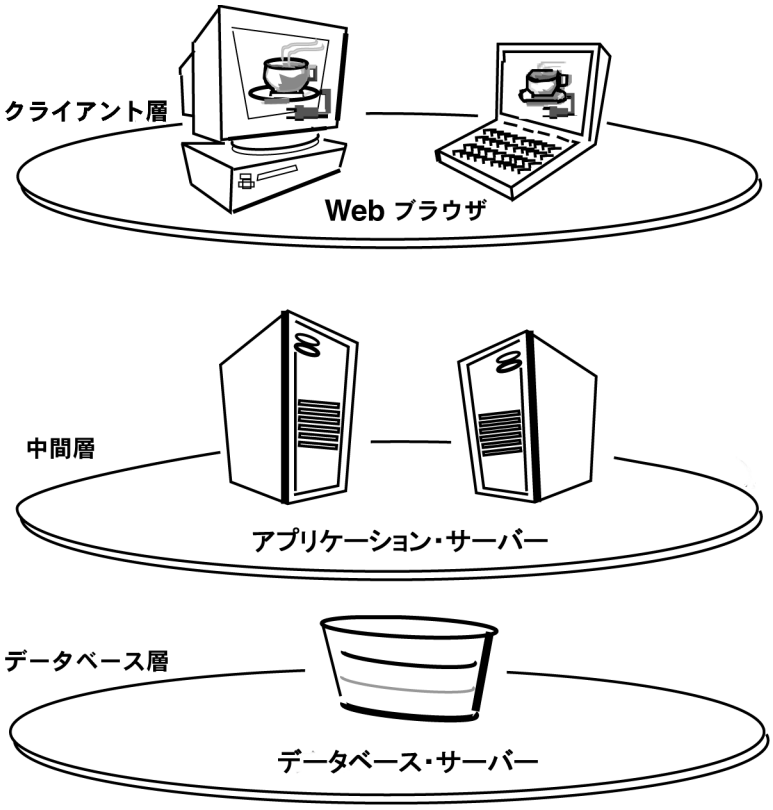

図 2-1 Forms Server アーキテクチャ

2.3 Forms Server コンポーネント

Forms Server は中間層のアプリケーション・サーバーで、複雑なトランザクション・フォーム・アプリケーションをインターネット上に配置します。開発者は Oracle Forms Developer で新規アプリケーションを構築し、Forms Server を使用してインターネット上に配置できます。また、開発者は従来のクライアント / サーバー型アプリケーションを、そのアプリケーション・コードを変更することなく 3 層のアーキテクチャに移行することもできます。

Forms Server は図 2-2 に示されるように、3 つの主要なコンポーネントで構成されます。

- **Forms アプレット** クライアントに自動的にダウンロードされ、Web ブラウザで参照します。
- **Forms Listener** 中間層にあります。
- **Form Runtime Engine** これも中間層にあります。

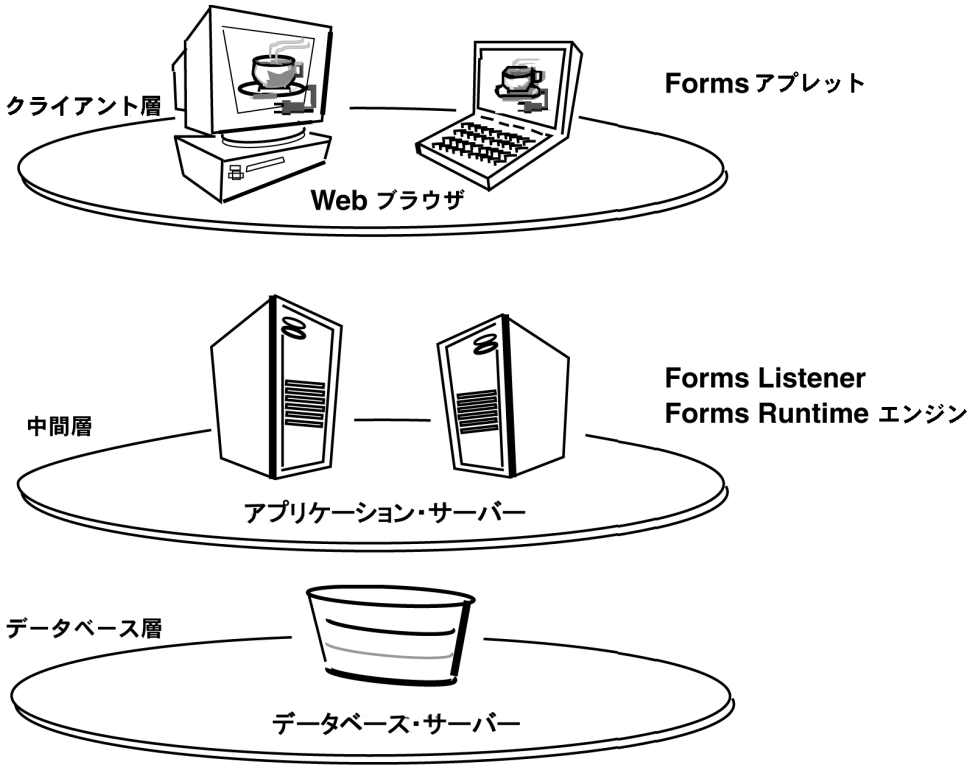

図 2-2 Web 上でフォームを実行する 3 層の構成

2.3.1 Forms アプレット

ユーザーが Web 上でフォーム・セッションを実行すると、Java ベースの小さな Forms アプレットがアプリケーション・サーバーから動的にダウンロードされて、自動的に Java クライアント・マシンにキャッシュされます。

Forms アプレットは、Forms Server Runtime エンジンにユーザー・インタフェースを提供します。拡張可能な最適化 Java アプレットとして、クライアントの Web ブラウザのフレームワーク内で操作できます。項目間の移動や、チェックボックスにチェックを付ける処理は対話的に行われ、その際に生成される情報がビジュアルに返されます。アプリケーションの表示のみを行い、特定のアプリケーション・ロジックは含まれません。

フォームのサイズや複雑度にかかわらず、どのフォームに対しても同じ Java アプレット・コードを使用できます。これは Web 上に配置するアプリケーションやフォームごとに、Java コードを記述する必要がないことを意味します。

2.3.2 Forms Listener

Forms Listener は Java クライアントと、Forms Server Runtime プロセス間におけるブローカとして機能します。Java クライアント・プロセスから接続リクエストを受け取り、Forms Server Runtime プロセスを開始します。またリスナーは Java クライアント完了後できるだけ早く接続できるように、実行するエンジンのプールを保持できます。

2.3.3 Forms Runtime エンジン

Forms Runtime エンジンはアプリケーション・ロジックと処理を管理します。Java クライアントのためにデータベース接続を保持します。クライアント / サーバー・モードで実行するのに使用されたのと同じフォーム、メニューおよびライブラリ・ファイルを使用します。これまでのクライアント・サーバー資産アプリケーションをインターネット上に配置するのに、アプリケーション・コードの変更は必要ありません。

Forms Runtime エンジンは 2 つの役割を果たします。クライアントのブラウザと通信するときは、クライアントからのリクエストを処理するサーバーとして機能します。データベース・サーバーと通信するときは、要求されたデータをデータベース・サーバーに対して問い合せるクライアントとして機能します。

2.4 Forms Server の動き

Web 上で Forms アプリケーションの実行を開始するには、Java 対応の Web ブラウザを使用して、URL にアクセスします。Forms Server に関するプロセス・フロー中に発生する一連のイベントが、図 2-3 とそれに続く記述で説明されています。

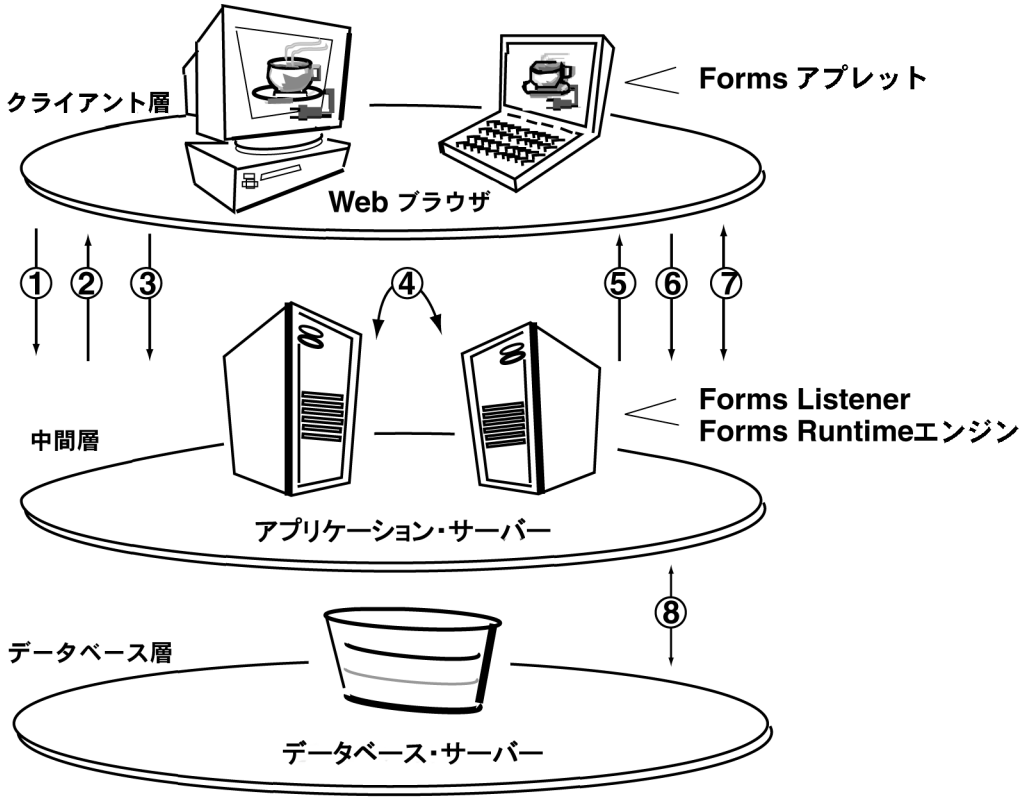

図 2-3 Forms Server プロセスのフロー

ユーザーが Web 上で Forms アプリケーションを実行するとき、次のイベントが順に起こります。

1. ユーザーが Forms アプリケーションの実行を指示する HTML ページの URL にアクセスします。
2. その HTML ページが Web ブラウザにダウンロードされます。必要であれば、クライアントは Forms アプレットを含む Java アーカイブ・ファイルもダウンロードします。Forms アプレットがインスタンス化され、HTML ページからのパラメータを使用して実行する Forms アプリケーションが決定されます。
3. Forms アプレットは Forms Listener（Forms アプレットをダウンロードしたマシンの特定のポートに存在します）にリクエストを送信します。
4. Forms Listener は Forms Runtime エンジンに連絡して、Forms Server Runtime プロセスに接続します。HTML ページに含まれている場合は、Forms コマンド・ライン・パラメータ（フォーム名、ユーザー ID およびパスワード、データベース SID、メニュー名などの）および任意のユーザー定義 Form Builder パラメータが、Forms Listener によってプロセスに渡されます。
5. リスナーは Runtime エンジンとの接続を確立して、接続情報を Forms アプレットに送信します。
6. その後 Forms アプレットは、Runtime エンジンとの直接接続を確立します。
7. この時点で Forms アプレットと Runtime エンジンは通信を直接開始し、他のユーザーからの開始リクエストを受け取れるようにリスナーを開放します。Forms アプレットはアプリケーションのユーザー・インタフェースを、ユーザーの Web ブラウザのメイン・ウィンドウに表示します。
8. Runtime エンジンで実行中のアプリケーションは、データベースと直接通信します。

3

インストールおよび構成時の選択肢の概説

3.1 概要

この章では、Forms Server のインストール時に表示されるオプションを説明します。インストールの選択項目には次のようなものがあります。

- Oracle Installer によりサーバーを自動構成するか、手動で構成するか。
- ソケット接続、HTTP 接続または SSL (secure sockets layer) の HTTP 接続にするか。
- Oracle JInitiator を使用するか、AppletViewer を使用するか。
- ロード・バランスにするか、スタンドアロン構成にするか。
- Oracle WebDB Listener を使用するか、CGI をサポートする他の Web リスナーにするか。

3.2 自動または手動構成

Forms Server のインストール時に、Oracle Installer によりサーバーを自動構成するか、手動構成するかのいずれかを選択できます。

3.2.1 サーバーの自動構成

Oracle Installer を使用すると、ユーザーは以前のリリースでは必要であった時間のかかる構成作業を行わずに、Forms Server をインストールできます。Oracle Installer によってほとんどの構成定義がインストレーションの一部として自動的に行われます。さまざまな構成定義がデフォルトとして用意されており、1 台のマシン構成でインストールすることも、ロード・バランスを使用する複数マシン構成でインストールすることもできます。

Oracle Installer を使用してサーバーを自動的に構成する場合は、第 4 章「クイック・スタート構成のステップ」を参照してください。

3.2.2 サーバーの手動構成

手動構成は次の場合に適しています。

- 既存の Web 環境があり、Forms Server がすでにインストールされ、構成済みの Web サーバーとともに使用する場合。
- サーバーの構成オプションをカスタマイズする場合。

Forms Server を手動で構成する詳細は、第 5 章「Forms Server の構成」を参照してください。

3.3 ソケット、HTTP または HTTPS

Forms Server はアプリケーションを配置するのに 3 つのモードを使用できます。

- ソケット
- HTTP
- HTTPS (HTTP with SSL)

ユーザー固有のネットワーク環境に最適な Forms Server の実装についての詳細は、9.3 項「ネットワーク環境における Forms Server の配置」を参照してください。

3.3.1 ソケット

他の多くのインターネット・ベースのテクノロジと同様に、Forms Server は当初、ソケットを使用して通信するよう設計されました。ソケットは TCP/IP に対する標準のプログラミング・インタフェースです。

ソケットがどのようなものかは、ネットワーク上で通信するプログラムのナンバリング・システムを想像してもらえば一番簡単です。一般にこれらのプログラムは共通のソケット番号を共有する、クライアント部分とサーバー部分があります。サーバーはクライアントからのリクエストを共通のソケット・ポートでリスニングします。プログラムのクライアント部分とサーバー部分間の通信は、通称*ソケット接続*で行われます。

ソケットの代表的な使用例を示します。クライアントが標準以外のポート番号（たとえば、http://www.xyz.com:9000）を持つ URL にリクエストを送信します。これはクライアントのブラウザがソケット番号 9000 へ接続しようとすることを意味します。また、これはポート 9000 上で接続をリスニングするサーバーが www.xyz.com 上で稼働されていることも意味します。

ソケット・モードでの配置は効率的で簡単に使用できます。Forms Server はネットワーク化されたホスト・マシン上で稼動し、ユーザーのマシン上で実行される Java クライアントからの接続を、特定のソケット上またはポートでリスニングします。このメソッドが機能するには、クライアントおよびサーバーのマシンがネットワーク上でお互いを識別できるか、通信できる必要があります。このモードではサーバー側でプロキシを使用できません。

注意 : サーバー側のプロキシとは、インターネットに接続またはサービスを提供するときに、サーバー・ソフトウェアを稼動させているマシンを不明または匿名にしておくメソッドです。これはクライアントには認識されずに、サーバーに対する権限のないアクセスを拒否するために使用されるセキュリティ機能です。

サーバーとクライアントが、インターネットなどの安全を保障されていないネットワークで分断されている場合は、ソケット・ベースでの配置は厳格なセキュリティを意味する可能性があります。

3.3.2 HTTP

HTTP モードでは同じくソケット接続を介して通信が確立されますが、この場合は HTTP ソケット接続になります。Forms Server はソケットによる独占接続ではなく、Java クライアントからの HTTP 接続をリスニングします。Forms Server と Java クライアント間のすべての内部メッセージが、HTTP パケット内にカプセル化されます。

HTTP ソケット接続では、サイトのクライアントとサーバー間でファイアウォールを通した安全な通信を実現します。HTTP 通信のみを許可するサイトは、構成をまったく変更しないかほとんど変更せずに既存のファイアウォールを通して Forms アプリケーションを配置できます。プロキシが使用されているという事実、はクライアントには完全にわかりません。クライアントから見ると、Forms Server に直接接続しているのと変わりありません。

ファイアウォールが存在する場合は、ソケット・モードは機能しません。ファイアウォールを通したソケット・モード接続が機能するには、Forms Server によって使用される特定のソケットまたはポートが開かれていて、ファイアウォールで使用可能になっていることが必要ですが、その場合は開かれたソケットの場所を突き止めるトラフィックにネットワークがさらされることになります。これではファイアウォールに穴が開けられ、その目的が本質的に損なわれます。

HTTP はインターネット上にアプリケーションを配置するために最もよく使用されるプロトコルです。企業はファイアウォールをロックして HTTP 通信のみを許可することで、プライベート・ネットワークのセキュリティを大幅に強化できます。ファイアウォールを提供している企業の多くは、その製品で HTTP 標準をサポートしており、多くの企業は保有するプライベート・ネットワークの中を HTTP 通信が行きかうことを好意的に認めています。

3.3.3 HTTPS

HTTPS モードでは通信は、3.3.2 項「HTTP」に説明があるように、HTTP ソケット接続を介して確立されます。ただし、HTTPS では SSL(secure sockets layer) も実装されます。

Forms Server では SSL をトランスポート・プロトコルとして使用し、機密性、整合性およびサーバー認証を提供できます。SSL は、アプリケーション・レベルの 1 つ下のレベルである、転送レベルで動作します。これは Telnet、FTP および HTTP などのアプリケーションレベルのプロトコルでメッセージが処理される前に、SSL でメッセージの暗号化と複合化ができることを意味します。

- **機密性**は、意図しない受信者によってメッセージが読まれるのを防ぐために、クライアントとサーバー間のメッセージを暗号化することで達成されます。メッセージは RC4 暗号化機能を使用して暗号化されます。

  国内ライセンスのサーバーとクライアントは、128 ビットの暗号化をサポートします。輸出ライセンスのサーバーとクライアントは、40 ビットの暗号化をサポートします。国内ライセンスの（128 ビットの暗号化）サーバーの場合、輸出（40 ビットの暗号化）ライセンスを持つクライアントはそのままではサーバーに接続できません。接続するには、サーバー側で環境変数 FORMS60_HTTPS_NEGOTIATE_DOWN を TRUE に設定する必要があります。（デフォルトの設定は FALSE です。）詳細は 5.3.1 項「環境変数のカスタマイズ」を参照してください。この環境変数を TRUE に設定すると、接続しようとするクライアントによってサポートされる最高レベルの暗号化が常にサーバーで使用されます。FALSE に設定すると、サポートする暗号化のレベルがサーバーのレベルより低いクライアントは接続できません。次の表に実現例を示します。

サーバーの暗号化レベル	クライアントの暗号化レベル	FORMS60_HTTPS_NEGOTIATE_DOWN の設定	接続
128 ビット（国内）	40 ビット（輸出） 128 ビット（国内）	TRUE	可。輸出クライアントに対しては 40 ビット、国内クライアントに対しては 128 ビットの暗号化をサポートします。
128 ビット（国内）	40 ビット（輸出）	FALSE	なし
40 ビット（輸出）	128 ビット（国内）	TRUE	可。40 ビットの暗号化をサポートします
40 ビット（輸出）	40 ビット（輸出）	TRUE	可。40 ビットの暗号化をサポートします
40 ビット（輸出）	40 ビット（輸出）	FALSE	可。40 ビットの暗号化をサポートします

- **整合性**は、メッセージが変更されるのを防ぎます。メッセージが変更されると正しく復号化できません。
- **サーバー認証**は、そのサーバーが対象サーバーに間違いないことをクライアント・マシンで検証するプロセスです。たとえば、クライアントが機密データをサーバーに送信する場合、クライアントは相手側のサーバーが安全で、送信した機密データの正しい受信者であることを検証できます。サーバー認証は、RSA コンプライアント・デジタル証明を使用して行われます。クライアントのブラウザがサーバーに接続したとき、サーバーは証明書を検証のために提示します。

HTTPS モードの使用を決定したときは、証明書リクエストの作成と管理のために、Oracle Wallet Manager のインストールが必要になります。詳細は 5.5 項「HTTPS 接続モード設定の追加ステップ」を参照してください。

3.4 Oracle JInitiator または AppletViewer

ユーザーはブラウザ (Netscape Navigator または Internet Explorer) を、Oracle JInitiator プラグインまたは AppletViewer のいずれかと組み合せて、アプリケーションを参照できます。Oracle JInitiator の使用をお薦めします。

3.4.1 Oracle JInitiator

Oracle JInitiator は Web ブラウザ内で実行され、Oracle Forms アプリケーションを Web 上で参照するのにお薦めします。ブラウザのデフォルトの JVM ではなく、クライアント上の特定の Java 仮想マシン (JVM) を使用するように指定する機能が提供されます。Oracle JInitiator はブラウザによって提供されるデフォルトの JVM を、置き換えたり変更せずに、代替の JVM をプラグイン形式で提供します。かわりに、プラグインのフォームで代替の JVM が提供されます。

Oracle JInitiator は Java ソフトウェア・プラグインのオラクル社版です。これは Netscape Navigator ではプラグインとして実行され、Internet Explorer では ActiveX コンポーネントとして実行されます。

Oracle JInitiator の構成と配置には、いくつかの指定が必要です。詳細は、付録 B「Oracle JInitiator」を参照してください。

3.4.2 AppletViewer

ユーザーは AppletViewer を使用してもアプリケーションを参照できます。AppletViewer は Java Developer Kit (JDK) コンポーネントの 1 つで、クライアント・マシンで使用して、Forms Server 上で実行されるアプリケーションを参照できます。

AppletViewer 内でアプリケーションを実行するにはいくつかの指定が必要です。アプリケーションを AppletViewer で実行する詳細は、付録 C「AppletViewer」を参照してください。

3.5 ロード・バランス

Forms Server には、1 人から数千人までのユーザーに比類のないパフォーマンスを提供するために、ハードウェアのリソースを最適化するためのロード・バランス機能が含まれています。ロード・バランスを使用すると、ハードウェアの限界に近づいたときにマシンのアップグレードや交換をしなくても、単にアプリケーションを実行するマシンを追加して負荷をいくつかのマシン間に分散することで解決できます。

ロード・バランスのインプリメントに関する特定の情報は、第 12 章「ロード・バランスに関する考慮事項」を参照してください。

3.6 Oracle WebDB Listener

Oracle Forms アプリケーションを Web 上で実行するには、Forms Server に加えて Web リスナーが必要です。次の中から選択できます。

- Oracle WebDB Listener の使用
- CGI をサポートする別の Web リスナーの使用

便宜のために、CGI (Common Gateway Interface) をサポートし、システム負荷の少ない Web リスナーである、Oracle WebDB Listener が Forms Server とともに提供されています。Oracle WebDB Listener は、Forms Server とともに提供される Oracle Installer を使用してインストールおよび構成ができます。

別の Web リスナーをすでに使用し、そのリスナーを引き続き使用する場合や Oracle Application Server を使用している場合は、WebDB Listener を使用しないように選択できます。Forms Server は、Microsoft IIS、Apache または Lotus Domino などの CGI をサポートする、任意の Web リスナーで作動します。インストレーションの完了後、Web リスナーのいくつかの仮想パスを構成して、Forms Server で使用できるようにする必要があります。作業手順の指示がインストレーション時に生成されます。

WebDB Listener のインストールに関する詳細は、付録 D「Oracle Installer リファレンス」を参照してください。

3.7 次のステップ

インストール時の選択肢を決定後、必要な Forms Server コンポーネントをインストールし構成できます。Oracle Installer を使用して Forms Server を自動的にインストールし構成する場合は、第 4 章「クイック・スタート構成のステップ」を参照してください。Forms Server を手動でインストールし構成する場合は、第 5 章「Forms Server の構成」を参照してください。

4

クイック・スタート構成のステップ

4.1 概要

この章は、Forms Server の迅速な構成とテストに役立ちます。この章では Oracle Installer で使用可能な、自動構成プロセスを主に使用するユーザーを対象にしています。

4.2 Oracle Installer について

Oracle Installer を使用すると、平易かつ実効的な方法で、Forms Server およびすべてのオプションをインストールすることができます。Oracle Installer を使用すると、ユーザーはアプリケーションをただちに実行できます。

Oracle Installer は構成ステップのほとんどを、インストレーションの一部として自動的に行います。さまざまなデフォルトのインストールが可能です。ユーザーは Oracle Installer を使用して、Forms Server を 1 台のマシンにインストールすることも、ロード・バランスを使用する複数のマシンにインストールすることもできます。実際にはいくつかの手動ステップが要求されることがありますが、その場合でもインストレーション・プロセス中に生成される "Configuration Instructions" ファイルに、行うべき操作が指示されます。

Forms Server を手動で構成する場合は、第 5 章「Forms Server の構成」を参照してください。

4.3 Oracle Installer を使用して Forms Server を構成する

この項ではできるだけ迅速に構成を完了して立上げることを目指します。この項で説明されるステップに従うと、Forms Server は最も基本的なオプションで構成されます。Forms Server と WebDB リスナーを 1 台のマシンにインストールした後で、Forms Server のインストレーションをテストする方法を説明します。

4.3.1 Oracle Installer の起動

Oracle Installer を実行する前に、Forms Server をインストールする Windows NT のワークステーションの管理者権限があることを確認してください。Oracle Installer を実行するには次のステップに従います。

1. 起動している Windows アプリケーションをすべて終了します。
2. **「スタート」→「ファイル名を指定して実行」**を選択すると「ファイル名を指定して実行」ダイアログが表示されます。
3. 「ファイル名を指定して実行」ダイアログ・ボックスに次のように入力します（D: は実際の CD-ROM ドライブの文字に置き換えます）。

   ```
   D:¥SETUP.EXE
   ```

4. **「OK」**をクリックして、Oracle Installer を起動します。

インストレーションの開始に先立ち、次の 2 つのダイアログ・ボックスが表示されます。

- 「Oracle インストール 設定」ダイアログ・ボックスでは、会社名と ORACLE_HOME ディレクトリのパスを入力します。デフォルト設定のまま使用することをお薦めします。
- 「言語」ダイアログ・ボックスでは、Forms Server を実行する言語を選択します。

4.3.2 Forms Server のインストール

WebDB Listener とすべてのデフォルト値を使用して、Forms Server を 1 台のマシンにインストールする場合は、次のステップに従ってください。

1. **Oracle Forms Server** をクリックします。

   このオプションで Forms Server をインストールします。

2. 「**標準**」をクリックします。

   このオプションでは、製品コンポーネントを自動的にインストールすることができます。

3. 「**Web 配置用 Forms Server**」をクリックします。

   このオプションでは、Oracle Forms アプリケーションを Web アプリケーションとして、1 台以上のサーバー・マシンで実行するのに必要なコンポーネントのインストールと構成を行います。今回のインストレーションを 1 台のマシン構成にするか、ロード・バランスを使用する複数マシン構成の一部にするかの選択肢が表示されます。

4. 「**単一マシン構成**」をクリックします。

   このオプションは、Forms Server を 1 台のマシンのみにインストールする場合に選択します。

5. 「**Orcle WebDB Listener の使用**」をクリックします。

   WebDB Listener のインストールと構成を行う場合は、このオプションを選択します。

6. 「**はい**」をクリックしてサービスを作成し、開始します。

   選択したコンポーネントは、使用する前に開始する必要があります。

7. 「**OK**」をクリックして、WebDB Listener のデフォルトを受け入れます。

   これにより、ホスト名および WebDB Listener ポート番号に対してデフォルト値が設定されます。

8. 「**OK**」をクリックして、Oracle Forms をインストールするディレクトリを設定します。
9. 「**OK**」をクリックして、Forms Server パラメータのデフォルト値を受け入れます。
10. 「**OK**」をクリックして、Oracle JInitiator をインストールするディレクトリを設定します。

4.3.3 Oracle Installer で行われること

Oracle Installer によってマシンの依存性が自動的に分析された後、ユーザーが選択したオプションに基づいて Forms Server が構成されます。

Oracle Installer により、インストールの終了時に構成指示ファイルが作成され、作成された設定を参照できます。インストレーション・プロセス完了後に、構成指示ファイルを印刷し、その記述に従って構成操作を完了できます。

Oracle Installer と構成指示ファイルの詳細は、付録 D「Oracle Installer リファレンス」を参照してください。

4.4 インストール後の構成のテスト

これでサーバー環境が構成されたので、インストレーションと構成をテストできます。

4.4.1 Web Form Tester へのショートカット

標準のテスト・フォームを実行するためのショートカット "Run Form on the Web" が、Forms Server プログラム・グループ内に作成されました。ショートカットのターゲットは、C:¥ORANT¥tools¥web60¥html¥runform.htm です。ショートカットをデスクトップ上に置いて今後のテストに使用できます。

4.4.2 Web Form Tester の実行

インストレーションおよび構成をテストするには、Web ブラウザからショートカットをクリックするか、「**スタート**」→「**プログラム**」→「**Oracle Forms 6i**」→「**Run Form on the Web**」をクリックします。

Web Form Tester HTML ファイル、runform.htm が Web ブラウザで表示されます。Web Form Tester に追加情報を入力せず、ただちにインストレーションをテストする場合は次のようにします。

1. 「**Run Form**」をクリックして、デフォルトのテスト・フォーム、test.fmx を別の Web ブラウザで表示します。
2. Web ブラウザでテスト・フォームが表示されたら、「**OK**」をクリックします。

注意 : デフォルトのテスト・フォームを参照するには、Jinitiator をプラグインとしてインストールする必要があります。Jinitiator がマシンにインストールされていない場合は、プラグインをダウンロードする指示に従う必要があります。インストレーション・プロセスの詳細は、付録 B「Oracle JInitiator」を参照してください。

4.4.3 他のフォームを Web Form Tester でテストする

フォーム名を「**Form**」フィールドに入力すると、他のフォームを Web Form Tester でテストできます。

Web Form Tester には次のフィールドがあります。

Runform パラメータ

- **Form:** テスト対象のフォーム名 (test.fmx)
- **Userid:** ログイン用のユーザー ID(ユーザー名 / パスワード@接続文字列)
- **Other Parameters:** 追加のフォーム・パラメータ (useSDI=yes)
- **Look and Feel:** (Oracle または Generic)
- **Color scheme:** (Teal、Red、Khaki、Blue、Olive または Purple)

Web リスナー (HTTPD) の詳細

- **Web host:** HTTP リスナーを実行するマシン (servername.domain.com)
- **Web port:** HTTP リスナー用のポート番号 (80)

4.4.4 URL のコピー

テスト・フォームが Web ブラウザに表示されたら、テスト・フォームによって生成された URL リンクを、Web サイトにリンクとして追加します。そのリンクからユーザーがフォームを実行できます。

Web ブラウザの「場所 :」フィールド (Netscape Navigator) または「アドレス」フィールド (MS Internet Explorer) から URL をコピーして Web ページに貼り付け、ユーザーがその Web ページからフォームを実行できるようにします。

4.5 次のステップ

実際にアプリケーションを配置するには、いくつかのステップを実行する必要があります。そのステップにはランタイム実行可能ファイルの作成、Web サーバーへの実行可能ファイルの配置、アプリケーションの URL のブロードキャストおよび Web ブラウザでのアプリケーションのテストが含まれます。

これらのステップは第 6 章「Web へのフォームの配置」で説明します。

5

Forms Server の構成

5.1 概要

この章では、Forms Server で CGI をサポートする環境の構成に必要なステップを説明します。

初心者レベルまたはベテランの Forms ユーザーであっても、Oracle Installer の「実行環境インストール」オプションを使用して、第 4 章「クイック・スタート構成のステップ」の説明にあるように、コンポーネントのインストールと構成を行うことをお薦めします。ロード・バランスをインプリメントする場合は、第 12 章「ロード・バランスに関する考慮事項」を参照して、ロード・バランスに関するインストレーションの選択肢を理解するようにしてください。

インストレーションの完了後、この章に戻ってインストレーションの調整や、必要な変更を行うことができます。

- Oracle Installer の「**実行環境インストール**」オプションでは Forms Server の環境変数、起動パラメータ、デフォルトのディレクトリおよび構成ファイル (formsweb.cfg、base.htm および basejini.htm) を、Oracle Installer ダイアログ・ボックスで選択した項目に従って設定します。次に選択に基づき、必要なコンポーネントのすべてをインストールします。
- Oracle Installer の「**カスタム・インストール**」オプションでも、Forms Server の環境変数、起動パラメータ、デフォルトのディレクトリおよび構成ファイル (formsweb.cfg、base.htm および basejini.htm) を設定できます。ただし、インストールするコンポーネントを選択する必要があります。

どちらのインストール方法でも、この章の情報を使用して初期の構成を変更できます。両方のインストール方法で dev6iconfig.txt ファイルが生成されます（¥<ORACLE_HOME><oracle_home>¥orainst ディレクトリ内にあります）。このテキスト ファイルにはシステムの構成を完了するのに必要なステップが含まれています。たとえば、WebDB Listener の使用を選択しない場合は、web サーバーの構成が必要です。

この章には、次の項が含まれています。

- Web サーバーの構成

- Forms Server の構成
- Oracle Installer により生成される構成ファイルの カスタマイズ
- HTTPS 接続モード設定の追加ステップ

5.2 Web サーバーの構成

Web サーバーとともに提供されたドキュメントを使用して、Forms を使用できるように Web サーバーを構成します。Apache または WebDB などの任意のリスナーを使用できます。Web サーバーを Forms Server とともに実行するよう構成するには次の作業が必要です。

- ホスト名の指定。
- ポート番号の指定。ほとんどの Web サーバーがポート 80 を使用します。
- 下記の一覧表のように、物理ディレクトリに対して仮想パス・マッピングを作成します。物理ディレクトリは Oracle Installer によって作成され、Forms によって使用されるファイルが含まれます。

仮想パス	物理ディレクトリ	説明
/forms60java/	<ORACLE_HOME>¥forms60¥java¥	Forms Java ファイル
/dev60html/	<ORACLE_HOME>¥tools¥web60¥html¥	Forms を実行する初期の HTML ファイル
/dev60cgi/	<ORACLE_HOME>¥tools¥web60¥cgi¥	CGI 実行可能ファイル
/jinitiator/	<ORACLE_HOME>¥jinit¥	JInitiator（ダウンロード用）
/dev60temp/	<ORACLE_HOME>¥tools¥web60¥temp¥	Forms テンポラリ・ファイル

5.3 Forms Server の構成

Oracle Installer によって設定された初期構成を変更するには、次の作業が必要です。

- 環境変数のカスタマイズ
- NT 上の Forms Server 起動パラメータの変更

注意： これらの変更を行うには管理者権限が必要です。また、構成の変更を有効にするには、そのほとんどの場合でサーバーの再起動が必要です。

5.3.1 環境変数のカスタマイズ

この項では環境変数のカスタマイズの方法について説明します。Oracle Installer は Forms Server 環境変数をインストレーション・プロセス中に自動的に設定します。変更が必要な場合はこれらの変数をカスタマイズできます。環境変数は次のとおりです。

環境変数	デフォルト値と説明
FORMS60_PATH	<ORACLE_HOME>¥<FORMS60> 実行する Form を検索するときに、Forms が検索するパスを指定します。パスはセミコロン (;) で区切ります。
FORMS60_OUTPUT	<ORACLE_HOME>¥tools¥web60¥temp 生成したレポート・ファイルを格納するアプリケーション・サーバー上の物理ディレクトリ。Reports を使用していない場合は、この環境変数は必要ありません。詳細は、7.5.2 項「レポートの実行」を参照してください。
FORMS60_MAPPING	/dev60temp FORMS60_OUTPUT 変数で定義された物理ディレクトリを指す仮想ディレクトリ。Reports を使用していない場合は、この環境変数は必要ありません。詳細は、7.5.2 項「レポートの実行」を参照してください。
FORMS60_MESSAGE_ENCRYPTION	TRUE 環境変数は RC4 40 ビットの暗号化機能を使用して、Forms メッセージを暗号化します。ソケットおよび HTTP 通信モードにのみ適用されます。デフォルトの設定で、通信は暗号化されます。
FORMS60_WALLET	<ORACLE_HOME>¥<FORMS60>¥wallet HTTPS 通信モードのみに使用されます。詳細は 5.5 項「HTTPS 接続モード設定の追加ステップ」を参照してください。
FORMS60_HTTPS_NEGOTIATE_DOWN	FALSE HTTPS 通信モードのみに使用されます。詳細は 5.5 項「HTTPS 接続モード設定の追加ステップ」を参照してください。

注意 : FORMS60_MAPPING 環境変数によって設定された仮想ディレクトリは、FORMS60_OUTPUT 環境変数によって設定された物理ディレクトリに対応している必要があります。

5.3.1.1 NT での環境変数のカスタマイズ

Windows NT で環境変数をカスタマイズするには、次の作業を行います。

1. レジストリ・エディタを起動します。
   a. タスクバーから「**スタート**」→「**ファイル名を指定して実行**」を選択します。

b. 「ファイル名を指定して実行」ダイアログ・ボックスに次を入力します。`regedit`

c. 「**OK**」をクリックします。

2. レジストリ・エディタで、次の手順を実行します。

   - 「HKEY_LOCAL_MACHINE」→「SOFTWARE」→「ORACLE」に移動します。

3. 環境変数の場所を指定します。たとえば、FORMS60_PATH をハイライトします。

   a. 「**編集**」→「**変更**」を選択します。レジストリ・エディタにより「文字列の編集」ダイアログ・ボックスが開かれます。

   b. 環境変数の新しい値を入力します。

   c. 「**OK**」をクリックします。

5.3.1.2 UNIX での環境変数のカスタマイズ

UNIX 上では、環境変数をコマンド・シェルで定義します。たとえば次のようにします。

```
FORMS60_PATH=/<ORACLE_HOME>/<FORMS60>
FORMS60_OUTPUT=/<ORACLE_HOME>/tools/web60/temp
FORMS60_MAPPING=/dev60temp
FORMS60_MESSAGE_ENCRYPTION=TRUE
FORMS60_WALLET=/<ORACLE_HOME>/<FORMS60>/wallet
FORMS60_HTTPS_NEGOTIATE_DOWN=FALSE
```

5.3.2 NT 上の Forms Server 起動パラメータの変更

Oracle Installer を使用して NT 上に Forms Server をインストールするとき、Forms Server は NT サービスの 1 つとして設定され、特別に指定しない限り、再起動すると自動的に開始されます。

注意： Oracle Enterprise Manager (OEM) を使用している場合は、NT サービスとして実行するコンポーネントを設定しないでください。

Oracle Installer を使用すると、インストレーション時に入力したポートやモードなどの Forms Server パラメータのすべてがレジストリに保存されるため、Forms Server は常に初期の構成で開始されます。

次のいずれかの方法で起動パラメータを変更できます。

- 既存の Forms Server サービスに関するレジストリの編集
- Forms Server サービスのアンインストールと再インストール
- Forms Server のテンポラリ・インスタンスの開始

5.3.2.1 既存の Forms Server サービスに関するレジストリの編集

すべての起動パラメータが次の Windows レジストリに保存されます。

HKEY_LOCAL_MACHINE → SYSTEM → CurrentControlSet → Services → OracleFormsServer<*ServiceName*>

Forms Server の起動パラメータを変更するには、NT サービスのコントロール パネルでサービスを停止し、レジストリのパラメータ設定を編集し、そのレジストリ設定を保存してサービスを再起動します。起動パラメータ定義は、5.3.3 項「Forms Server 起動パラメータの説明」を参照してください。

5.3.2.2 Forms Server サービスのアンインストールと再インストール

既存の Forms Server サービスを削除し、新規の起動パラメータを使用して再インストールできます。最初に NT サービスのコントロール・パネルでサービスを停止します。

コマンド・ウィンドウで、次のように入力します。

```
ifsrv60 -uninstall <FormsServerServiceNameToBeRemoved>
```

続いて次を入力します。

```
ifsrv60 -install <NewFormsServerServiceName> port=<portNum> mode=<socket/http/https>
[pool=<numOfRunforms> log=<logfilePath> exe=<RunformexeName>]
```

起動パラメータ定義は、5.3.3 項「Forms Server 起動パラメータの説明」を参照してください。

サービスが自動的に開始され、以降の起動時に対して "autostart" に設定されます。起動パラメータの設定は次の Windows レジストリの場所に保存されます。

HKEY_LOCAL_MACHINE → SYSTEM → CurrentControlSet → Services → OracleFormsServer<*ServiceName*>

5.3.2.3 Forms Server のテンポラリ・インスタンスの開始

コマンド・ラインから Forms Server のテンポラリ・インスタンスを開始できます。このインスタンスは NT サービスとしては開始されません。したがってこのインスタンス (ifsrv60) を NT サービスのコントロール・パネルからではなく、「Windows NT タスク マネージャ」ウィンドウから停止します。このインスタンスは停止され、マシンが再起動されたときに再開されません。次のように入力します。

```
ifsrv60 port=<portNum> mode=<socket/http/https> [pool=<numOfRunforms>
log=<logfilePath> exe=<RunformexeName>]
```

起動パラメータ定義は、5.3.3 項「Forms Server 起動パラメータの説明」を参照してください。

注意 : テンポラリ・インスタンスに入力されたパラメータ値は以降の開始時に保存されません。

5.3.3 Forms Server 起動パラメータの説明

Forms Server 起動時には次のパラメータが使用されます。

- Port パラメータ
- Mode パラメータ
- Pool パラメータ
- Log パラメータ

5.3.3.1 Port パラメータ

サーバー・プロセスが開始されるポートを決定します。Forms Server プロセスの開始時にポート番号を指定しないと、デフォルトのポート 9000 上でプロセスが開始されます。サーバー・プロセスを開始するポート番号は、アプリケーションの HTML ファイル、構成パラメータまたは URL に指定する serverPort 番号と一致する必要があります。

5.3.3.2 Mode パラメータ

Forms Server を、ソケット・モード（ソケットの直接接続を使用）、HTTP モード（ファイアウォールを貫通できる）または HTTPS モード（ファイアウォールを貫通でき、SSL (secure sockets layer) を追加で使用してサーバー認証およびメッセージの暗号化を行う）のいずれかで実行するかを決定します。各モードの詳細は、3.3 項「ソケット、HTTP または HTTPS」を参照してください。

5.3.3.3 Pool パラメータ

後から使用するユーザーが利用できる、アクティブなスペアの接続数を決定します。たとえば "pool" が 5 に設定された場合は、5 つのアクティブなスペア接続があります。

5.3.3.4 Log パラメータ

パス名およびログ・ファイル名が提供されると、サーバーのログ・ファイルを生成します。例、log=¥PathName¥LogFileName。

5.4 Oracle Installer により生成される構成ファイルのカスタマイズ

インストレーション時に次の構成ファイルがシステムにインストールされました。

- formsweb.cfg
- base.htm および basejini.htm

ユーザーが最初に Web 対応のアプリケーションを開始すると（アプリケーションの URL へのリンクをクリックすることで）、ベース HTML ファイルが Forms CGI によって読み込まれます。ベース HTML ファイル内の任意の変数 (%*variablename*%) は、formsweb.cfg ファイルに指定された適切なパラメータ値によって置換されるか、URL リクエストがある場合はそのクエリー・パラメータの値によって置換されます。

Oracle Installer によって作成される構成ファイルは、インストレーション・ダイアログ ボックスで選択された構成オプションを反映します。

変更が必要な場合は構成ファイルを変更できます。ファイルはインストレーション後、¥<FORMS60>¥server ディレクトリ内に置かれます。

5.4.1 formsweb.cfg

インストレーション時に設定したほとんどの構成パラメータの設定が、このファイルに含まれます。変更が必要な場合はこれらのパラメータをカスタマイズできます。

ベース HTML ファイル内の変数 (%*variablename*%) は、formsweb.cfg ファイルに指定された適切なパラメータ値によって置換されるか、URL リクエストがある場合はその問合せパラメータからの値によって置換されます。

変数 (%*variablename*%) は formsweb.cfg ファイルでも使用されます。(この場合のデリミタは常に % です)。変数は Oracle レジストリ値か環境変数 (<ORACLE_HOME> など) または特別の値、%leastloadedhost% にする必要があります。

構成の変更は formsweb.cfg ファイルに入力して、変数はベース HTML ファイル内で使用することをお薦めします。

5.4.1.1 formsweb.cfg ファイル内のパラメータ

パラメータ	必須 / 任意	パラメータ値
baseHTML	**必須**	アプレット・タグを含む HTML ファイルへの物理パス
baseHTMLJInitiator	**必須**	JInitiator タグを含む HTML ファイルへの物理パス
ie50	Internet Explorer 5.0 ブラウザを使用するユーザーがいる場合に推奨	クライアントが Internet Explorer 5.0 ブラウザを使用している場合は、JInitiator と AppletViewer のいずれかを使用できます。"JInitiator" の設定では basejini.htm ファイルと JInitiator を使用します。"Native" の設定ではブラウザ固有の JVM を使用します。
HTML delimiter	**必須**	変数名のデリミタ。デフォルトで % になります。
MetricsServerHost	任意	ロード・バランス用。第 12 章「ロード・バランスに関する考慮事項」参照。
MetricsServerPort	任意	ロード・バランス用。第 12 章「ロード・バランスに関する考慮事項」参照。
MetricsServerErrorURL	任意	ロード・バランス用。第 12 章「ロード・バランスに関する考慮事項」参照。
MetricsTimeout	任意	ロード・バランス用。第 12 章「ロード・バランスに関する考慮事項」参照。

パラメータ	必須 / 任意	パラメータ値
leastloadedhost	任意	ロード・バランス用。第 12 章「ロード・バランスに関する考慮事項」参照。 ベース HTML ファイルまたは formsweb.cfg ファイルのどちらかに指定できる変数です。ロード・バランスの設定では、負荷の最も低いマシン名を指定する必要があります。推奨するデフォルトのベース HTML ファイルを使用して、ロード・バランスを設定するときは必ず serverHost=%loastloadedhost% を formsweb.cfg ファイルに指定してください。 このプレース・ホルダは、ロード・バランスが行われる間に、負荷の最も低いシステムの名前に動的に置き換えられます。
標準のアプレットまたはオブジェクトのパラメータ **注意：**次のすべてをベース HTML ファイルに *%variablename%* として指定できます。例 : `<PARAM NAME="connectMode"  VALUE="%connectMode%">` ベース HTML ファイル内のすべての変数は、formsweb.cfg ファイルに指定された適切なパラメータ値で置き換えられます。		
codebase	**必須**	物理ディレクトリをポイントするように定義した仮想ディレクトリ <ORACLE_HOME>¥forms60¥java.
code	**必須**	コード・パラメータは削除や変更をしないでください。常に次の値にします。oracle.forms.engine.Main.
connectMode	**HTTP および HTTPS 接続では必須** ; ソケット接続では任意	Forms Server で使用する接続プロトコルのタイプをクライアントに指定します。有効な値はソケット、http および https です。デフォルトはソケットです。詳細は 3.3 項「ソケット、HTTP または HTTPS」を参照してください。
archive	任意	あらかじめロードする、カンマで区切ったアーカイブ・ファイルのリスト。絶対パスでない場合は codebase からの相対パス。
width	**必須**	Form の幅をピクセルで指定。
height	**必須**	Form の高さをピクセルで指定。
align	任意	left | center | right | top | middle | bottom
alt	任意	アプレットのかわりに表示されるテキスト（ブラウザがアプレットをサポートしない場合）
hspace	任意	水平方向の余白をピクセルで指定。
vspace	任意	垂直方向の余白をピクセルで指定。
type	**必須**	ハード コードされた値（JInitiator に対しては "application/x-jinit-applet";AppletViewer には値はリクエストされません）。

パラメータ	必須 / 任意	パラメータ値
name	任意	アプレットのインスタンス名。
title	任意	アドバイザリ・タイトル文字列。
border	任意	表示する境界線
standby	任意	ロード時に表示するテキスト。
codetype	任意	デフォルトでタイプになります。
Forms アプレットに固有のパラメータ（PARAM タグ内）		
serverHost	任意	Forms Server の ifsrv60.exe を実行するホスト (デフォルトは Web リスナー・マシン)。
serverPort	必須	Forms Server の ifsrv60.exe がリスニングするポート。ほとんどの場合、ポート番号は 9000（デフォルト）のままです。
serverArgs	**必須**	Runform 用のコマンド・ラインパラメータ下記の Runform パラメータを参照してください。 forms_param を任意の有効な Forms Runtime コマンド・ラインパラメータで置換します。user_param を任意の有効なユーザー定義パラメータで置換します。例、<param name="serverArgs" VALUE="module=order.fmx"> **注意：** Forms Runtime コマンド・ラインとユーザー定義パラメータを複数指定できます。HTML ファイルにディレクトリ・パスを含めるか、FORMS60_PATH 環境変数を定義して、.FMX ファイルの物理ディレクトリ・パスを指定する必要があります。拡張子 .FMX は任意です。
splashScreen	任意	アプレットが表示される前に表示する .GIF ファイルを指定。スプラッシュなしの場合は「NO」に設定します。デフォルトのスプラッシュを使用する場合は空白のままにします。
background	任意	背景に表示する .GIF ファイルを指定。背景なしの場合は「NO」に設定します。デフォルトの背景を使用する場合は空白のままにします。
clientDPI	任意	1 インチ当たりのドット数（DPI）を指定し、JVM によって戻される DPI 設定を上書きします。これにより、各プラットフォームのさまざまな DPI 設定を管理できます。たとえば、Win32 プラットフォームで開発されたフォームは、DPI 値の違いにより、UNIX プラットフォーム上では正しく表示されない可能性があります。clientDPI の値には、すべての正の整数を指定できます。Oracle は 50 から 200 の整数を使用することをお薦めします。<param name="clientDPI" value="200">

パラメータ	必須 / 任意	パラメータ値
separateFrame	任意	アプレットを分割フレーム内に表示するかどうかを指定。有効な値： True または False。
lookAndFeel	任意	アプリケーションのルック・アンド・フィールを指定。有効な値： Oracle または Generic（Windows 95 のルック・アンド・フィール)。
colorScheme	任意	アプリケーションの配色を指定。有効な値： Teal、Titanium、Red、Khaki、Blue、Olive または Purple。 **注意**： lookAndFeel が Generic に設定されている場合、colorScheme は無視されます。
serverApp	任意	アプリケーションのクラス名がある場合に、デフォルトを置き換えます。アプリケーション固有のフォント・マッピングの作成およびアイコン・パスの設定には、アプリケーション・クラスを使用します。
heartBeat	任意	このパラメータを使用して、クライアントが稼動中であることを示すためにサーバにパケットを送る頻度を設定します。この整数値は、分単位で定義します。デフォルトは 2 分です。
imageBase	任意	このパラメータを使用してアイコンファイルが格納される場所を指定します。次の中から選択します。 ■ **codeBase** は、アイコン検索パスが Java クラスを含むディレクトリに対応することを示します。アイコンを JAR ファイルに格納する場合にこの値を使用します（推奨)。 ■ **documentBase** は、デフォルトです。Forms Server CGI を使用した配置では、アイコンパスをカスタムアプリケーションファイル中に指定します。
registryPath	任意	このパラメータを使用して、serverApp パラメータで名前をつけたアプリケーションファイル名が格納されている仮想ディレクトリをリスト表示します。
webformsTitle	任意	このパラメータを使用して、フォームの表示ウィンドウの上端に表れるタイトルを変更します。
Runform パラメータ（serverArgs パラメータ）		
MODULE	**必須**	Form のモジュール名 (任意でパスを含みます)。
USERID	任意	scott/tiger@ORA8 などのログイン文字列
ユーザー定義パラメータ	任意	任意の名前 / 値のペア。

5.4.1.2 デフォルトの formsweb.cfg ファイル

デフォルトの formsweb.cfg ファイルには次が含まれます。

```
; Forms Web CGI Configuration File
; --------------------------------
; This file defines parameter values used by the Forms Web CGI
; ********************************
; PARAMETER VALUES USED BY DEFAULT
; ********************************
  ; SYSTEM PARAMETERS
  ; -----------------
  ; These have fixed names and give information required by the Forms
  ; Web CGI in order to function.  They cannot be specified in the URL query
  ; string.  But they can be overriden in a named configuration (see below).
baseHTML=<FORMS60>¥server¥base.htm
baseHTMLJInitiator=<FORMS60>¥server¥basejini.htm
HTMLdelimiter=%
MetricsServerPort=9020
MetricsServerErrorURL=
  ; The next parameter specifies how to execute the Forms applet under
  ; Microsoft Internet Explorer 5.0.  Put IE50=native if you want the
  ; Forms applet to run in the browser's native JVM.
IE50=JInitiator

  ; USER PARAMETERS
  ; ---------------
  ; These match variables (e.g. %form%) in the baseHTML file. Their values
  ; may be overridden by specifying them in the URL query string
  ; (e.g. "http://myhost.mydomain.com/ifcgi60.exe?form=myform&width=700")
  ; or by overriding them in a specific, named configuration (see below)
  ; 1) Runform arguments:
form=test.fmx
userid=
otherparams=

  ; 2) HTML page title, attributes for the BODY tag, and HTML to add before and
  ;    after the form:
pageTitle=Forms Server
HTMLbodyAttrs=
HTMLbeforeForm=
HTMLafterForm=

  ; 3) Values for the Forms applet parameters:
width=650
height=500
separateFrame=false
splashScreen=no
```

```
background=no
lookAndFeel=Oracle
colorScheme=teal
serverApp=default
serverPort=9000
serverHost=
connectMode=socket
archive=f60web.jar

  ; 4) Parameters for JInitiator
    ; Page displayed to Netscape users to allow them to download JInitiator.
    ; If you create your own version, set this parameter to point to it.
jinit_download_page=/jinitiator/us/jinit_download.htm
    ; Parameters related to the version of JInitiator.
    ; These are valid for Oracle JInitiator version 1.1.7.16o
    ; WARNING: You must update these if you upgrade to a later version
    ; of JInitiator (as instructed in the documentation for that version)
jinit_classid=clsid:9F77A997-F0F3-11d1-9195-00C04FC990DC
jinit_exename=jinit.exe#Version=1,1,7,16
jinit_mimetype=application/x-jinit-applet;version=1.1.7.16
    ; Values for JInitiator version 1.1.7.18o:
    ; jinit_classid=clsid:9F77A997-F0F3-11d1-9195-00C04FC990DC
    ; jinit_exename=jinit11718.exe#Version=1,1,7,18
    ; jinit_type=application/x-jinit-applet;version=1.1.7.18
; ********************************
; SPECIFIC CONFIGURATIONS
; ********************************
;  You may define your own specific, named configurations (sets of parameters)
;  by adding special sections as illustrated in the following examples.
;  Note that you need only specify the parameters you want to change.  The
;  default values (defined above) will be used for all other parameters.
;  Use of a specific configuration can be requested by including the text
;  "config=<your_config_name>" in the query string of the URL used to run
;  a form.  For example, to use the sepwin configuration, your could issue
;  a URL like "http://myhost.mydomain.com/ifcgi60.exe?config=sepwin".

; Example 1: configuration to run forms in a separate browser window with
;            "generic" look and feel (include "config=sepwin" in the URL)
[sepwin]
separateFrame=True
lookandfeel=Generic

; Example 2: configuration affecting users of MicroSoft Internet Explorer 5.0.
;            Forms applet will run under the browser's native JVM rather than
;            using Oracle JInitiator.
[ie50native]
IE50=native
```

```
; Example 3: configuration forcing use of the base.htm base HTML file in all
;            cases (means applet-style tags will always be generated and
;            JInitiator will never be used).
[applet]
baseHTMLJInitiator=

; Example 4: configuration to run the demos
;            PLEASE DO NOT REMOVE THIS EXAMPLE, !
;            It is needed to run the Forms demos (if they are installed)
[demo]
pageTitle=Forms Server Demos
width=700
height=550
form=start60
userid=%Demos_ConnectString%
archive=f60all.jar, oracle_ice-4_03_1.jar
serverApp=/forms60demo/demo
lookAndFeel=oracle
colorScheme=teal
```

5.4.2 base.htm および basejini.htm

Forms Server のインストレーションと構成時に、Oracle Installer によって 2 つのベース HTML ファイルがシステムに作成されます。**ほとんどの場合、これらのファイルの変更は必要ありません。**

ユーザーが最初に Web 対応のアプリケーションを開始すると (アプリケーションの URL へのリンクをクリックすることで)、ベース HTML ファイルが Forms CGI によって読み込まれます。ベース HTML ファイル内の任意の変数 (%*variablename*%) は、5.4.1 項「formsweb.cfg」での説明のように、formsweb.cfg ファイルに指定された適切なパラメータ値によって置き換えられるか、URL リクエストがある場合はそのクエリー・パラメータの値によって置き換えられます。次にベース HTML ファイルがユーザーの Web ブラウザにダウンロードされます。

注意 : 変更する任意のベース HTML 変数は、5.4.1 項「formsweb.cfg」の説明のように、対応する formsweb.cfg ファイルのパラメータ値を変更することで変更できます。

次のベース HTML 初期ファイルを ¥<FORMS60>¥server ディレクトリで利用できます :

- **basejini.htm**: これは、Oracle JInitiator を使用する Forms アプレットの実行に必要なタグが含まれる HTML ファイルです。オラクル社によってこの方法での動作が確認されたブラウザ (および標準の APPLET タグを使用して動作しないブラウザ) に適しています (Windows プラットフォームのみ)。例は 5.4.2.4 項「デフォルトの basejini.htm ファイル」を参照してください。JInitiator 設定の詳細は、付録 B「Oracle JInitiator」を参照してください。

- **base.htm**: これは Forms アプレットを AppletViewer またはネイティブな JVM が Forms で作動することがオラクル社によって確認済みの、任意の Web ブラウザで実行するのに必要な APPLET タグを含む、ベース HTML ファイルです。例は 5.4.2.3 項「デフォルトの base.htm ファイル」を参照してください。AppletViewer 設定の詳細は、付録 C「AppletViewer」を参照してください。

ベース HTML ファイルを新規作成する場合は次の作業を行います。

1. ¥<FORMS60>¥server ディレクトリにある、basejini.htm または base.htm 初期ファイルをコピーします。
2. ファイル名を変更します（たとえば、`order.htm`）。
3. ユーザーに表示されるテキストを追加または変更します（たとえば、<TITLE> および <BODY> タグ内のテキスト）。
4. 必要に応じてパラメータを変更します。5.4.1 項「formsweb.cfg」の説明にあるように、ベース HTML ファイルの変数を使用して、実際の値は formsweb.cfg ファイルに指定することをお薦めします。
5. 新しいベース HTML ファイルを任意のディレクトリ内に置きます。formsweb.cfg ファイル内のベース HTML パラメータ (またはベース HTMLJInitiator パラメータ) を更新して、ベース HTM ファイルの完全な物理パスが含まれるようにします。

5.4.2.1 ベース HTML ファイル内のパラメータと変数

注意： base.htm または basejini.htm ファイルで提供されるパラメータ・タグを使用しない場合は、ファイルから削除してください。

パラメータ	必須 / 任意	パラメータ値
CGI システム変数		
leastloadedhost	任意	ロード・バランス用。第 12 章「ロード・バランスに関する考慮事項」参照。 ベース HTML ファイルまたは formsweb.cfg ファイルのどちらかに指定できる変数です。ロード・バランスの設定では、負荷の最も低いマシン名を指定する必要があります。推奨するデフォルトのベース HTML ファイルを使用して、ロード・バランスを設定するときは必ず serverHost=%leastloadedhost% を formsweb.cfg ファイルに指定してください。 ロード・バランスを使用している場合は、負荷が最も低いシステムの名前によって、このプレースホルダが動的に置き換えられます。

パラメータ	必須 / 任意	パラメータ値
標準のアプレットまたはオブジェクトのパラメータ **注意**：残りのパラメータ値を変数 (%*variablename*%) としてベース HTML ファイルで指定することをお薦めします。例 : <PARAM NAME="connectMode" VALUE="%*connectMode*%"> 次に、実際のパラメータ値を 5.4.1.1 項「formsweb.cfg ファイル内のパラメータ」で定義される formsweb.cfg ファイルに指定します。すべての変数が実行時に適切なパラメータ値で置き換えられます。		

5.4.2.2 使用方法

- 変数の値はベース HTML ファイル内のどこででも使用できます。変数は特別のデリミタで囲まれた名前として指定されます。(デフォルトのデリミタは % です。) たとえば、HTML ファイルに次の行を置くことができます。

  ARCHIVE="%Archive%"

  次に値を formsweb.cfg ファイル (または URL 問合せ文字列内) の %Archive% に割り当てる必要があります。

- すべての変数は実行時に値を受け取る必要があります。変数が値を受け取らないと、ユーザーの Web ブラウザに渡す HTML ファイルを Forms Server が構築できず、エラーが発生します。
- パフォーマンスを向上するには、JAR ファイルのダウンロード用のソースとして 1 つの Web サーバーのみを使用してください。同じファイルを異なるサーバーから複数回ダウンロードすることを防げます。

5.4.2.3 デフォルトの base.htm ファイル

```
<HTML>
<!-- FILE: base.htm (Forms Server)                          -->

<!-- This is the default base HTML file for running a form on the   -->
<!-- web using APPLET-style tags to include the Forms applet.       -->
<!-- This file will be REPLACED if you reinstall "Forms Web CGI and -->
<!-- cartridge", so you are advised to make your own version if you -->
<!-- want to make any modifications.  You should then set the       -->
<!-- baseHTML parameter in the Forms web CGI configuration file     -->
<!-- (formsweb.cfg) to point to your new file instead of this one.  -->

<!-- IMPORTANT NOTE: default values for all the variables which     -->
<!-- appear below (delimited by the percent character) are defined  -->
<!-- in the formsweb.cfg file. It is preferable to make changes in  -->
<!-- that file where possible, and leave this one untouched.        -->
```

```
<HEAD><TITLE>%pageTitle%</TITLE></HEAD>

<BODY %HTMLbodyAttrs%>
%HTMLbeforeForm%

<!-- Forms applet definition (start) -->
<APPLET CODEBASE="/forms60java/"
        CODE="oracle.forms.engine.Main"
        ARCHIVE="%archive%"
        WIDTH="%Width%"
        HEIGHT="%Height%">

<PARAM NAME="serverPort" VALUE="%serverPort%">
<PARAM NAME="serverHost" VALUE="%serverHost%">
<PARAM NAME="connectMode" VALUE="%connectMode%">
<PARAM NAME="serverArgs"
       VALUE="module=%form% userid=%userid% %otherParams%">
<PARAM NAME="separateFrame" VALUE="%separateFrame%">
<PARAM NAME="splashScreen"  VALUE="%splashScreen%">
<PARAM NAME="background"  VALUE="%background%">
<PARAM NAME="lookAndFeel"  VALUE="%lookAndFeel%">
<PARAM NAME="colorScheme"  VALUE="%colorScheme%">
<PARAM NAME="serverApp" VALUE="%serverApp%">

</APPLET>
<!-- Forms applet definition (end) -->

%HTMLafterForm%

</BODY>
</HTML>
```

5.4.2.4 デフォルトの basejini.htm ファイル

```
<HTML>
<!-- FILE: basejini.htm (Forms Server)                                -->

<!-- This is the default base HTML file for running a form on the   -->
<!-- web using JInitiator-style tags to include the Forms applet.   -->
<!-- This file will be REPLACED if you reinstall "Forms Web CGI and -->
<!-- cartridge", so you are advised to make your own version if you -->
<!-- want to make any modifications.  You should then set the       -->
<!-- baseHTML parameter in the Forms web CGI configuration file     -->
<!-- (formsweb.cfg) to point to your new file instead of this one.  -->

<!-- IMPORTANT NOTE: default values for all the variables which     -->
<!-- appear below (delimited by the percent character) are defined  -->
```

```
<!-- in the formsweb.cfg file. It is preferable to make changes in  -->
<!-- that file where possible, and leave this one untouched.        -->

<HEAD><TITLE>%pageTitle%</TITLE></HEAD>

<BODY %HTMLbodyAttrs%>
%HTMLbeforeForm%

<!-- Forms applet definition (start) -->
<OBJECT classid="%jinit_classid%"
        codebase="/jinitiator/%jinit_exename%"
        WIDTH="%Width%"
        HEIGHT="%Height%"
        HSPACE="0"
        VSPACE="0">
<PARAM NAME="TYPE"       VALUE="%jinit_mimetype%">
<PARAM NAME="CODEBASE"   VALUE="/forms60java/">
<PARAM NAME="CODE"       VALUE="oracle.forms.engine.Main" >
<PARAM NAME="ARCHIVE"    VALUE="%archive%" >

<PARAM NAME="serverPort" VALUE="%serverPort%">
<PARAM NAME="serverHost" VALUE="%serverHost%">
<PARAM NAME="connectMode" VALUE="%connectMode%">
<PARAM NAME="serverArgs"
       VALUE="module=%form% userid=%userid% %otherParams%">
<PARAM NAME="separateFrame" VALUE="%separateFrame%">
<PARAM NAME="splashScreen"  VALUE="%splashScreen%">
<PARAM NAME="background"  VALUE="%background%">
<PARAM NAME="lookAndFeel"  VALUE="%lookAndFeel%">
<PARAM NAME="colorScheme"  VALUE="%colorScheme%">
<PARAM NAME="serverApp" VALUE="%serverApp%">
<COMMENT>
<EMBED SRC="" PLUGINSPAGE="%jinit_download_page%"
        TYPE="%jinit_mimetype%"
        java_codebase="/forms60java/"
        java_code="oracle.forms.engine.Main"
        java_archive="%archive%"
        WIDTH="%Width%"
        HEIGHT="%Height%"
        HSPACE="0"
        VSPACE="0"

        serverPort="%serverPort%"
        serverHost="%serverHost%"
        connectMode="%connectMode%"
        serverArgs="module=%form% userid=%userid% %otherparams%"
        separateFrame="%separateFrame%"
```

```
        splashScreen="%splashScreen%"
        background="%background%"
        lookAndFeel="%lookAndFeel%"
        colorScheme="%colorScheme%"
        serverApp="%serverApp%"
>
<NOEMBED>
</COMMENT>
</NOEMBED></EMBED>
</OBJECT>
<!-- Forms applet definition (end) -->

%HTMLafterForm%
</BODY>
</HTML>
```

5.5 HTTPS 接続モード設定の追加ステップ

HTTPS 接続モードではファイアウォールを越えるために、HTTP を使用して通信します。また、Forms Server は SSL をトランスポート・プロトコルとして使用して、機密性、整合性およびサーバー認証を提供します。この通信モードの説明は、3.3.3 項「HTTPS」を参照してください。

HTTPS 通信モードを使用するには、Forms Server を HTTPS モードで開始する前に次を行う必要があります。

- HTTPS 環境変数のカスタマイズ
- Oracle Wallet Manager を使用して Wallet を作成し、証明書を要求する

注意： HTTPS 接続モードを使用するには、Oracle Wallet Manager がインストールされている必要があります。Oracle Wallet Manager が Forms Server マシンにインストールされているかは、Oracle Installer の「カスタム・インストール」オプションを実行することでチェックできます。Oracle Wallet Manager が Installer の右側のリスト ボックスに表示されれば、製品がインストールされています。表示されない場合はこの時点でインストールしてください。

注意： サーバー認証を提供する Forms Server マシンのすべてに、Oracle Wallet Manager がインストールされている必要があります。

5.5.1 HTTPS 環境変数のカスタマイズ

Forms Server のインストール中に、HTTPS モードに関連付けられた 2 つの環境変数が設定されます。これらの環境変数がセキュリティのニーズに合うように設定されていることを確

認し、必要な場合は HTTPS モードで実行するすべての Forms Server マシン上で変更します。環境変数の変更方法は、5.3.1 項「環境変数のカスタマイズ」を参照してください。

環境変数	値
FORMS60_HTTPS_NEGOTIATE_DOWN	デフォルト値は FALSE です。 有効な値は TRUE および FALSE です。TRUE に設定されると、128 ビットの暗号化を使用するサーバーは、クライアントによってサポートされる最高レベルに合せて暗号化を調整します。FALSE に設定すると、サーバーは 128 ビットの暗号化をサポートしないクライアント接続を拒否します。詳細は 3.3.3 項「HTTPS」を参照してください。
FORMS60_WALLET	デフォルト値は ¥<ORACLE_HOME>¥<FORMS60>¥wallet です。 サーバー認証に使用される証明書を保持する "Wallet" を含むディレクトリ。

5.5.2 Oracle Wallet Manager を使用して Wallet を作成し、証明書を要求する

公開鍵暗号ではとりわけ証明書が要求されます。ユーザー証明書は*認証局* (CA) と呼ばれる第三者機関によって発行されます。証明書は安全な方法で取得され、アクセスのたびに証明書の妥当性チェックは必要ありません。

Forms Server と HTTPS モードを使用する Java クライアントの場合は、Java クライアントはユーザー証明書を使用して、サーバー証明書を検証することで、その Forms Server が妥当なサーバーであることをチェックします。Oracle Wallet Manager を使用して Wallet を作成し、ユーザー証明書を要求します。

Oracle Wallet Manager のインストール後、次の作業を行って、HTTPS 通信モードの使用時に要求されるユーザー証明書を取得する必要があります。

- Wallet の作成
- 証明書要求の作成
- ユーザー証明書のインポート
- 「自動ログイン」を「ON」に設定

次の項では、Oracle Wallet Manager における前述のステップを完了する方法の概要を提供します。詳細は、『Oracle Wallet Manager セキュリティ・ガイド』ドキュメントを参照してください。

注意： 複数の Forms Server マシンがある場合は、各マシンごとに一意の証明書を要求するか、同じ証明書をすべてのマシン上で使用できます。

- **一意の証明書を各マシンごとに使用するには**、この項内のすべてのプロシージャを、HTTPS モードで実行する各 Forms Server マシン上で実行します。

- **同じ証明書をすべてのマシンで使用するには**、この項のすべてのプロシージャを、証明書を含む Wallet を作成する Forms Server マシンのいずれかで実行します。次にその Wallet ファイル、ewallet.dev を HTTPS モードで実行する他の Forms Server マシンにコピーします。そのファイルを FORMS60_WALLET 環境変数で指定したディレクトリにコピーします。最後に、「自動ログイン」を「ON」に設定の説明にあるように、「自動ログイン」がすべてのマシンで「ON」に設定されていることを確認します。

5.5.2.1 Wallet の作成

Wallet を次のように作成します。

1. メニューバーから「**Wallet**」→「**新規作成**」をクリックします。「新規 Wallet」ダイアログ・ボックスが表示されます。
2. 「Wallet のパスワード」フィールドにパスワードを入力します。
3. パスワードの確認フィールドにパスワードを再入力します。
4. 「**OK**」をクリックして続行します。新規に空の Wallet が作成されたことを告げるメッセージが表示され、証明書要求を作成するかどうかがプロンプトで求められます。
5. 「**はい**」をクリックして、5.5.2.2 項「証明書要求の作成」を参照します。

5.5.2.2 証明書要求の作成

証明書要求を次のように作成します。

1. 次の情報を「証明書要求」ダイアログ・ボックスに入力します。
   - **共通の名前**: 証明書アイデンティティの名前を、名前が先で名字が後の書式で入力します。
   - **組織単位**: アイデンティティの組織単位での名前を入力します。たとえば、財務とします。
   - **組織**: アイデンティティの組織名を入力します。たとえば、XYZ Corp とします。
   - **市町村**: 市町村名を入力します。
   - **都道府県**: 都道府県名を入力します。
   - **国**: リストをクリックして国名の略称リストを表示します。組織が置かれている国をクリックして選択します。
   - **キー サイズ**: ドロップダウン・ボックスをクリックして、暗号 / 復号鍵のペアを作成するときの鍵のサイズを参照します。
   - **詳細**: 「詳細」をクリックして、「証明書要求詳細」ダイアログ・パネルを表示します。このフィールドを使用して、iID の識別名 (DN) を編集またはカスタマイズします。
2. 「**OK**」をクリックします。証明書要求が正常に作成されたことが「Oracle Wallet Manager」メッセージ・ボックスに表示されます。

3. メッセージ ボックスの本文から証明書要求テキストをコピーして、電子メールのメッセージに貼り付けます。要求を認証局に送信します。
4. 「**OK**」をクリックします。Oracle Wallet Manager のメイン・ウィンドウが再び表示されます。証明書のステータスが「要求済み」に変更されます。

5.5.2.3 ユーザー証明書のインポート

ユーザー証明書を CA から受け取った後、作成した Wallet にインポートする必要があります。2 つのいずれかの方法でインポートできます。

- 認証局から受信した電子メールから、ユーザー証明書を貼り付けます。
- ファイルからユーザー証明書をインポートします。

ユーザー証明書を貼り付けるには :

1. メニューバーから「**操作**」→「**ユーザー証明書のインポート**」をクリックします。「ユーザー証明書のインポート」ダイアログ・ボックスが開きます。
2. 「**証明証の貼付け**」ラジオ・ボタンをクリックして、「**OK**」をクリックします。「ユーザー証明書のインポート」ダイアログ・ボックスが開き、次のメッセージが表示されます。"base64 書式の証明書を下に貼り付けてください "。
3. 受信した電子メールの本文からユーザー証明書をコピーします。
4. 証明書をウィンドウに貼り付けて、「**OK**」をクリックします。ウィンドウの下に、ユーザー証明書が正常にインストールされましたというメッセージが表示されます。
5. 「**OK**」をクリックします。Oracle Wallet Manager のメイン・パネルが再び表示され、ユーザー証明書が User Certificates ツリーの一番下に表示されます。

ユーザー証明を含むファイルをインポートするには :

1. メニューバーから「**操作**」→「**ユーザー証明書のインポート**」をクリックします。「ユーザー証明書のインポート」ダイアログ・ボックスが開きます。
2. ユーザー証明書のあるパス名またはフォルダ名を入力します。
3. ユーザー証明書ファイルの名前（例、cert.txt）をクリックして選択します。
4. 「**OK**」をクリックします。ウィンドウの下に、ユーザー証明書が Wallet に正常にインポートされましたというメッセージが表示されます。
5. 「**OK**」をクリックして、ダイアログ・ボックスを閉じます。Oracle Wallet Manager のメイン・パネルが再び表示され、ユーザー証明書が User Certificates ツリーの一番下に表示されます。

5.5.2.4 「自動ログイン」を「ON」に設定

Oracle Wallet Manager Auto Login 機能により、Wallet のコピーが自動的に開かれます。そのため、Wallet にパスワードを提供する必要がなく、サーバー認証が行われます。

「自動ログイン」を「ON」に設定するには :

1. メニューバーから「**Wallet**」をクリックします。
2. 「**自動ログイン**」メニュー項目の隣のチェックボックスをクリックします。
3. 「ユーザー証明書が正しくインストールされました」というメッセージが、ウィンドウの下に表示されます。

注意 : 「**自動ログイン**」メニュー項目の隣のチェックボックスは、クリックのたびにオンとオフに切り替わります。チェックマークを消去するには、再度チェックボックスをクリックします。今度は autologin が使用不可になります。

注意 : 「自動ログイン」はサーバー認証を提供するすべての Forms Server マシンで、「ON」に設定される必要があります。

5.6 次のステップ

Forms Server の構成が完了したら、アプリケーションを Web に配置できます。詳細は、第 6 章「Web へのフォームの配置」を参照してください。

6

Web へのフォームの配置

6.1 概要

この章には、Oracle Forms アプリケーションを Web に配置するための情報が含まれます。Forms Server の構成後、実行可能ファイルを配置してアプリケーションの URL をブロードキャストできます。

注意： Forms Server の構成に関する情報は、第 4 章「クイック・スタート構成のステップ」または第 5 章「Forms Server の構成」を参照してください。

6.2 Forms アプリケーションの配置

Forms アプリケーションを配置するには、次のステップに従います。

- ランタイム実行可能ファイルを作成します。
- 実行可能ファイルを Web サーバー上に配置します。
- アプリケーションの URL をブロードキャストします。

6.2.1 ランタイム実行可能ファイルを作成

.FMX ランタイム実行可能ファイルは、配置する先のアプリケーション・サーバーと同じプラットフォーム上で作成する必要があります。

たとえば、アプリケーション・サーバーのオペレーティング・システムが Sun Solaris の場合は、Web に配置する .FMX ファイルを作成するのに、Solaris 版の Forms Compiler コンポーネントを使用する必要があります。

6.2.2 実行可能ファイルを Web サーバー上に配置

Forms アプリケーションは、Web サーバー上の任意のディレクトリ内に配置できます。適切なディレクトリ・パスとファイル名をベース HTML ファイルに含める必要があります。これが、ユーザーがアプリケーションを実行する際にアクセスするファイルになります。ベー

ス HTML ファイルの場所を指定する際に仮想ディレクトリ（例、/dev60html）を作成した場合は、ベース HTML ファイルをその場所に配置してください。

6.2.3 アプリケーションの URL のブロードキャスト

アプリケーションの URL のブロードキャストで必要なのは、対象ユーザーへの通知のみです。ユーザーは Java 対応の Web ブラウザを使用してその URL に接続し、該当するアプリケーションを実行できます。アプリケーション用の HTML ページを作成した場合は、ユーザーに与える URL は単にそのページを指すようにします。

たとえば、ABC 社で新しい受注追跡アプリケーションが使用可能になったことをアナウンスするには、次の URL をブロードキャストします。

http://www.abc.com:80/appshtml/order.html

ABC 社の URL は次のコンポーネントで構成されます。

- **プロトコル：** http
- **ドメイン：** www.abc.com
- **Web サーバー・リスナー・ポート：** 80（黙示的）
- **HTML ファイルの仮想ディレクトリ：** /appshtml
- **静的な HTML ファイル：** order.html

6.3 次のステップ

実行可能ファイルを Web サーバー上に配置し、アプリケーションの URL をブロードキャストした後は、Web ブラウザからのアプリケーションのテストと最適化を行いたくなるでしょう。

Forms アプリケーションを Web 上に配置するためのガイドラインとヒントは、第 7 章「アプリケーション設計に関する考慮事項」を参照してください。

Forms Server を使用してアプリケーションをインターネット上または他のネットワーク環境に配置する際に、チューニングで考慮すべき点の詳細は、第 11 章「パフォーマンス・チューニングに関する考慮事項」を参照してください。

中間層のマシン・プールを保持して、マシン間での負荷を均衡にするロード・バランスの詳細は、第 12 章「ロード・バランスに関する考慮事項」を参照してください。

7

アプリケーション設計に関する考慮事項

7.1 概要

この章では、Web で利用できる Forms アプリケーションの設計に関するガイドラインとヒントを示します。次の項が含まれています。

- 一般的なガイドライン
- Forms アプリケーション設計のためのガイドライン
- Forms Server で使用されるアイコンとイメージの配置
- グラフィックとレポートの統合
- Web 上の Forms アプリケーションの機能制限

7.2 一般的なガイドライン

Web 配置アプリケーションの設計について一般的なガイドラインを次に示します。

- Web アプリケーションのパフォーマンスに影響を与えるネットワーク要因について十分に考慮する（セキュリティ・ファイアウォールとの相互作用、多量のユーザー負荷、およびアプリケーションやデータベース・サーバーに対する頻繁なネットワーク・ラウンドトリップなど）。
- フォームおよびレポートに挿入するイメージ項目と背景イメージの数を制限する。イメージを必要とするたびに、アプリケーション・サーバーからダウンロードする必要があります。
- ネットワーク接続を見直せる箇所では、接続を最適化する。
- 問合せを、できる限り効率的に実行できるように設計し、PS/SQL プログラム単位のコンパイルもれがないようにする。

7.3 Forms アプリケーション設計のためのガイドライン

Web で利用できる Forms アプリケーションの設計に関するヒントを示します。これらは次の項で詳細に説明します。

- ユーザー独自のテンプレート HTML ファイルの作成
- HTML アプリケーション・メニューの作成
- Forms Server での Oracle Designer の使用
- ネットワーク通信量の削減
- 不要なグラフィックとイメージの削除
- 標準フォントの選択

7.3.1 ユーザー独自のテンプレート HTML ファイルの作成

（Oracle が提供するテンプレートを変更して）ユーザー独自の HTML ファイル・テンプレートを作成することを検討してください。ユーザー独自のテンプレートを作成すると、標準の Forms Client アプレット・パラメータおよびパラメータ値をテンプレートに直接に指定することができます。作成したテンプレートには、標準テキスト、ブラウザ・ウィンドウ・タイトル、またはイメージ（会社のロゴなど）を挿入でき、これらは Web で使用できるフォームを実行するときに参照できる、最初の Web ページに表示されます。標準パラメータ、値、および追加のテキストまたはイメージを追加すると、特定のアプリケーションのテンプレートをカスタマイズするために必要な作業量を減らすことができます。テキスト、イメージ、またはウィンドウ・タイトルを追加するには、テンプレート HTML ファイルに適切なタグを挿入します。

7.3.2 HTML アプリケーション・メニューの作成

Web に追加のアプリケーションを配置するときは、Web で使用できるさまざまなアプリケーションに対して 1 つにまとまったメニューを提供するために、単一の HTML ページを作成するようにしてください。このアプローチにより、利用または削除するすべてのアプリケーションの URL をブロードキャストする手間が省けます。使用できるアプリケーションの登録を変更する場合は、Web メニュー上のリンクが列挙されている箇所を変更します。このことで、ユーザーはメニュー URL に接続して、使用できるアプリケーションのリストから選択できるようになります。

7.3.3 Forms Server での Oracle Designer の使用

Forms Server は、Oracle Designer（32 ビット、リリース 1.3.2 以降）で生成したフォームをサポートします。標準の Oracle Designer のフォーム生成・テンプレート（ofg4pc1t.fmb および ofg4pc2t.fmb）を使用してフォーム定義とメニュー定義を生成する場合、Forms Server を使用して .FMX および .MMX ファイルをコンパイルし、ただちに Web 上でアプリケーションを実行できます。

7.3.4 ネットワーク通信量の削減

ユーザーが Web 上で Form Builder アプリケーションを操作する場合に発生するネットワーク・ラウンドトリップ数を減らすためには、アプリケーションにおける次の Form Builder 機能の一部またはすべてを削除する必要があります。

- **マウス・トリガー** フォームに、When-Mouse-Click、When-Mouse-DoubleClick、When-Mouse-Down、および When-Mouse-Up トリガーを含めると、スピードとパフォーマンスに影響を与えます。Forms Client は、これらのトリガーのどれかが実行されるたびに、Forms Server と通信する必要があります（ネットワーク・ラウンドトリップが必要になります）。When-Mouse-Move トリガーは、実行ごとに発生するネットワーク・ラウンドトリップ数が多いため、サポートされません。
- **タイマー** 100 分の 1 秒ごとに実行されるタイマー指定をフォームに含めると、毎分 60,000 回のネットワーク・ラウンドトリップというパフォーマンスの劣化が起こります。フォーム内のタイマー数を削減するか、タイマーが実行されるタイミング間隔を変更します。

7.3.5 不要なグラフィックとイメージの削除

アプリケーションに表示されるイメージ項目と背景イメージの数を可能な限り減らします。アプリケーション・ユーザーに対してイメージが表示されるたびに、イメージはアプリケーション・サーバーからユーザーの Web ブラウザにダウンロードする必要があります。

Web アプリケーションで会社のロゴを表示する場合には、アプリケーションの起動時にダウンロードされる HTML ファイルに会社のロゴ・イメージを含めてください。会社のロゴを背景イメージとしてアプリケーションに挿入するかわりにこれを行ってください。会社ロゴを背景イメージとして指定した場合データベースまたはファイルシステムから検索して、ユーザーのマシンにダウンロードするトラフィックが繰返し発生します。

7.3.6 標準フォントの選択

すべてのプラットフォーム間でサポートされているフォントは、あまり有りません。たとえば、Sans Serif は Microsoft Windows アプリケーションで一般的に使用されていますが、UNIX では使用できません。フォントがプラットフォームで使用できない場合、Form Builder は類似したフォントの使用を試みます。このため、Web 上で利用するフォームを設計するときには、必ず次に示すフォント・ガイドラインに従ってください。

実行時に、Forms Server はフォームのフォントを Java 等価フォントに対応づけします。次に、JAVA は配置プラットフォームに定義済みのフォントでフォントを表します。フォームのフォントを Java 等価フォントに変換するために、Java は Registry.dat と呼ばれるファイ

ル内の別名リストを使用します。次の表は、Java フォントと主要な配置プラットフォーム上の等価フォントの一覧です。

表 7-1

Java フォント	Windows フォント	X Windows フォント	Macintosh フォント
Courier	Courier New	adobe-courier	Courier
Dialog	MS Sans Serif	b&h-lucida	Geneva
DialogInput	MS Sans Serif	b&h-lucidatypewriter	Geneva
Helvetica	Arial	adobe-helvetica	Helvetica
Symbol	WingDings	itc-zapfdingbats	Symbol
TimesRoman	Times New Roman	adobe-times	Times Roman

Form Builder フォント別名表を用いてフォーム上のフォントを Java フォントに対応付けする際に対応付けできないフォントについては、Java は自動的に Java フォントを割り当てます。

7.4 Forms Server で使用されるアイコンとイメージの配置

この項では、アイコンとイメージのデフォルト・ディレクトリおよび検索パスの指定方法を説明します。

7.4.1 アイコン

Web 上に Forms アプリケーションを配置する場合、(アイコン・ボタン、メニュー、またはウィンドウに指定した）ICO 形式のアイコン・ファイルは使用しません。Web を介して接続できるファイル形式は、GIF または JPG ファイルのみです（GIF がデフォルト形式です)。

デフォルトでは、アイコンは HTML ファイルを含むディレクトリ、DocumentBase ディレクトリにあります。アイコンを別のディレクトリに格納する場合は、アプリケーション・ファイルを作成して、アイコン・ファイルを常駐させる仮想ディレクトリおよび使用するファイル形式（GIF または JPG）を指定する必要があります。このアプリケーション・ファイルは HTML ファイルで参照する必要があります。

カスタム・アプリケーション・ファイルを作成するには、次の手順に従います。

1. <ORACLE_HOME>¥forms60¥java¥oracle¥forms¥registry ディレクトリにある registry.dat テキスト・ファイルを別のディレクトリへコピーします。このディレクトリは Web サーバーの仮想ディレクトリ（/appfile など）へマップする必要があります。
2. 新規ファイルをリネームします（myapp.dat など）。
3. アイコンのディレクトリを指定する iconpath パラメータを次のとおりに変更します。

   default.icons.iconpath=/mydir or http://myhost.com/mydir
   （絶対パスの場合）

   または

   default.icons.iconpath=mydir
   （DocumentBase ディレクトリから始まる相対パスの場合）

4. iconextension パラメータを次のように変更します。

   default.icons.iconextension=gif

   または

   default.icons.iconextension=jpg

HTML ファイルでアプリケーション・ファイルを参照するには、次の手順に従います。

formsweb.cfg ファイルまたは HTML ファイルで、serverApp パラメータの値を変更し、値をアプリケーション・ファイルのディレクトリおよび名前に設定します。

```
<PARAM NAME="serverApp" VALUE="/appfile/myapp">
```
（絶対パスの場合）

または

```
<PARAM NAME="serverApp" VALUE="appfile/myapp">
```
（CodeBase ディレクトリに関連する相対パスの場合）

7.4.2 スプラッシュ画面イメージおよびバックグラウンド・イメージ

アプリケーションを Web で実行する場合、（接続中に表示される）スプラッシュ画面イメージおよびバックグラウンド・イメージ・ファイルを指定するための機能が必要です。

これらのイメージは、次に示すように HTML ファイルまたは formsweb.cfg ファイルで定義します。

```
<PARAM NAME="splashScreen" VALUE="splash.gif">

<PARAM NAME="background" VALUE="back.gif">
```

スプラッシュ画面およびバックグラウンド・イメージ・ファイルのデフォルト・ディレクトリは、ベース HTML ファイルが含まれている DocumentBase ディレクトリ内にあります。

7.4.3 アイコンおよびイメージを含むカスタム JAR ファイルの使用

（スプラッシュ画面またはバックグラウンドの）アイコンまたはイメージを使用するたびに、HTTP リクエストが Web サーバーに送信されます。クライアントとサーバー間の HTTP ラウンドトリップ数を減らすには、Java アーカイブ（JAR）ファイルにアイコンおよびイメージを格納するための機能が必要です。この方法を使用すると、JAR ファイルをダウンロードするのに、1 回の HTTP ラウンドトリップのみで済みます。

7.4.3.1 JAR ファイルの作成

SunSoft JDK には、*jar* と呼ばれる実行可能ファイルが含まれています。このユーティリティを使用すると、Java アーカイブ内にファイルを格納できます。詳細は、www.java.sun.com を参照してください。

例 :

```
jar -cvf myjar.jar Splash.gif Back.gif icon1.gif
```

このコマンドにより、myjar.jar と呼ばれる単一の JAR ファイルに 3 つのファイル（Splash.gif、Back.gif、icon1.gif）が格納されます。

7.4.3.2 JAR ファイル内でのファイルの使用

アイコンおよびイメージのデフォルトの検索パスは DocumentBase に関連します。ただし、それらのファイルを格納するために JAR ファイルを使用する場合は、検索パスは、Java アプレットを含むディレクトリ、CodeBase ディレクトリに関連する必要があります。

JAR ファイルを使用してアイコンおよびイメージを格納する場合は、検索パスが、基本の HTML ファイルで imageBase パラメータを使用している CodeBase に関連するよう指定する必要があります。

このパラメータは次の 2 つの異なる値が指定可能です。

- **DocumentBase** 検索パスは DocumentBase ディレクトリに関連します。これはデフォルトの動作です。
- **CodeBase** 検索パスは JAR ファイルを使用できるようにする CodeBase ディレクトリに関連します。

この例では、アイコンを含む JAR ファイルを使用して、検索が CodeBase の相対パスになるように指定します。パラメータ "imageBase" を設定していない場合は、検索は DocumentBase に関連し、アイコンは JAR ファイルから検索されません。

例 :

<PARAM NAME="archive" VALUE="icons.jar">

<PARAM NAME="imageBase" VALUE="CodeBase">

7.4.4 アイコンおよびイメージの検索パス

アイコンおよびイメージの検索パスは次の内容によって異なります。

- カスタム・アプリケーション・ファイルで指定した内容（アイコンの場合）
- HTML ファイルの SplashScreen パラメータおよび Background パラメータで指定した内容（イメージの場合）
- HTML ファイルの imageBase パラメータで指定した内容（アイコンとイメージの両方の場合）

Forms Server では、指定した内容に応じてアイコンを検索します。この例では、次のように仮定します。

- *host* はホスト名。
- *documentbase* は HTML ファイルを指す URL。
- *codebase* は、（HTML ファイルで指定した）開始クラス・ファイルのディレクトリを指す URL。
- *mydir* は、アイコンまたはイメージ・ディレクトリを指す URL。

7.4.4.1 DocumentBase

デフォルトの検索パスは DocumentBase ディレクトリに関連します。この場合、imageBase パラメータを指定する必要はありません。

表 7-2

	指定ディレクトリ	Forms Server で使用される検索パス
アイコン	デフォルト	http://host/documentbase
	iconpath=mydir （アプリケーション・ファイルで指定）	http://host/documentbase/mydir （相対パス）
	iconpath=/mydir （アプリケーション・ファイルで指定）	http://host/mydir （絶対パス）
イメージ	file.gif （HTML ファイルで指定）	http://host/documentbase/file.gif
	mydir/file.gif （HTML ファイルで指定）	http://host/documentbase/mydir/file.gif （相対パス）
	/mydir/file.gif （HTML ファイルで指定）	http://host/mydir/file.gif （絶対パス）

7.4.4.2 CodeBase

次に示すように、基本の HTML ファイルで imageBase=CodeBase パラメータを使用して、JAR ファイル内でのアイコンおよびイメージの検索を可能にします。

表 7-3

	指定ディレクトリ	Forms Server で使用される検索パス
アイコン	デフォルト	http://host/codebase or root of the JAR file
	iconpath=mydir （アプリケーション・ファイルで指定）	http://host/codebase/mydir or in the mydir directory in the JAR file （相対パス）
	iconpath=/mydir （アプリケーション・ファイルで指定）	http://host/mydir （絶対パス） JAR ファイルは使用されない。
イメージ	file.gif （HTML ファイルで指定）	http://host/codebase/file.gif or root of the JAR file

表 7-3

指定ディレクトリ	Forms Server で使用される検索パス
mydir/file.gif （HTML ファイルで指定）	http://host/codebase/mydir/file.gif or in the mydir directory in the JAR file （相対パス）
/mydir/file.gif （HTML ファイルで指定）	http://host/mydir/file.gif （絶対パス） JAR ファイルは使用されない。

7.5 グラフィックとレポートの統合

グラフィックまたはレポートを Web で使用できるフォームから起動するには、RUN_PRODUCT ビルトイン・サブプログラムを使用します。

7.5.1 グラフィックの実行

Web で使用できるフォームで RUN_PRODUCT をコールして、グラフィック・アプリケーションを表示する場合、特別な環境変数を設定する必要はありません。

7.5.2 レポートの実行

RUN_PRODUCT を使用して Web 上で実行中のフォームからレポートを実行するには、次に示す 3 つの環境変数を設定する必要があります。

表 7-4

環境変数	説明
FORMS60_OUTPUT	生成したレポート・ファイルを格納するアプリケーション・サーバー上の物理ディレクトリ。 例 : c:¥orant¥forms60¥my_reps¥
FORMS60_MAPPING	FORMS60_OUTPUT 変数で定義された物理ディレクトリを指す仮想ディレクトリ。 例 : /web_reps/
FORMS60_REPFORMAT	生成したレポート出力を格納する形式。 例 : PDF or HTML

Windows NT では、レジストリ内で環境変数を定義します。UNIX では、コマンド・シェル内で環境変数を定義します。環境変数設定の詳細は、付録 C「AppletViewer」を参照してください。

前述の環境変数を設定すると、Web 上で実行中のフォームが RUN_PRODUCT をコールしてレポートを起動する場合に、次の順序で自動的に発生します。

レポートの出力形式が SCREEN または PREVIEW の場合は、次のようになります。

- 結果出力は、FORMS60_OUTPUT 環境変数で指定した物理ディレクトリに（自動生成したファイル名のテンポラリ・ファイルとして）格納される。
- Web サーバーは、（FORMS60_MAPPING 環境変数で定義した仮想ディレクトリ内で）テンポラリ・ファイル名を検索する。
- Web サーバーは、FORMS60_REPFORMAT 環境変数で指定した目的の表示形式をチェックして、ユーザーのブラウザにその形式でレポートを表示する。

レポートの出力形式が FILE の場合は、次のようになります。

- レポートはユーザーのブラウザで表示されない。
- 結果を示すファイルは、FORMS60_OUTPUT 環境変数で指定された物理ディレクトリに格納される。
- レポート・ファイルのファイル名は、フォーム定義で定義した名前と同じ名前になる。

7.6 Web 上の Forms アプリケーションの機能制限

Web に配置するフォームを作成するときは、Forms の機能には、Web に配置されると異なる動作をする場合や、あるいはまったく動作しない場合があるということに注意してください。表 7-5 には、フォームの機能を一覧表示しています。機能が Web でサポートされているかどうか、またその機能に関するガイドラインやメモが記載されています。

表 7-5

機能	サポート	ガイドラインと注意事項
ActiveX、OCX、OLE、VBX	なし	ユーザーが出力を表示できないためアプリケーション・サーバー上に画面表示するサード・パーティ・コントロールはサポートされていません。
When-Mouse-Enter / Leave / Move トリガー	なし	トリガーを実行するたびに、ネットワーク・ラウンドトリップが必要になり、その結果パフォーマンスが低下する。
コンソール	あり	（ステータスとメッセージ行を含む）コンソールを表示するには、フォーム・レベル・プロパティのコンソール・ウィンドウを、コンソールを表示するウィンドウに設定する。
ファイアウォール	あり	Forms Server を HTTP または HTTPS モードで実行し、HTTP 1.1 プロトコルをサポートするファイアウォールが必要。
HOST_COMMAND、ORA_FFI、USER_EXIT	あり	これらの機能をコールすると、可視出力または GUI 要素が、クライアント / サーバー・モードのユーザーのマシン上に表示されることがよくある。Web 実装では、同じ機能をコールすると、アプリケーション・サーバー上に出力と GUI 要素を表示する（ユーザーは、それらを参照またはそれらと対話できない）。
アイコン・ボタン	あり	アイコン・イメージ・ファイルは GIF 形式にする（ICO 形式は使用できない）。
NLS、BIDI	あり	8 ビット言語のみサポートされる。

8

これまでのクライアント・サーバー・アプリケーションの資産の Web への移行

8.1 概要

現在 Forms Server のクライアント・サーバー・バージョンを使用している場合は、アプリケーションを Web 用の Forms Server へ簡単に移行できます。この章では、クライアント・サーバーと Web 実装間の違いを簡単に説明し、現在使用しているアプリケーションをクライアント・サーバー・ベースから Web ベースの Forms Server へ移行するためのガイドラインを示します。

8.2 クライアント・サーバー・ベースのアーキテクチャ

クライアント・サーバー・ベースの実装では、図 8-1 に示すように、Forms Server Runtime エンジンおよびすべてのアプリケーション・ロジックはユーザーのデスクトップ・マシンにインストールされます。いくつかのアプリケーションで指定されるデータベース・サーバー側のトリガーおよび論理以外の、すべてのユーザー・インタフェース処理およびトリガー処理は、クライアント上で行われます。

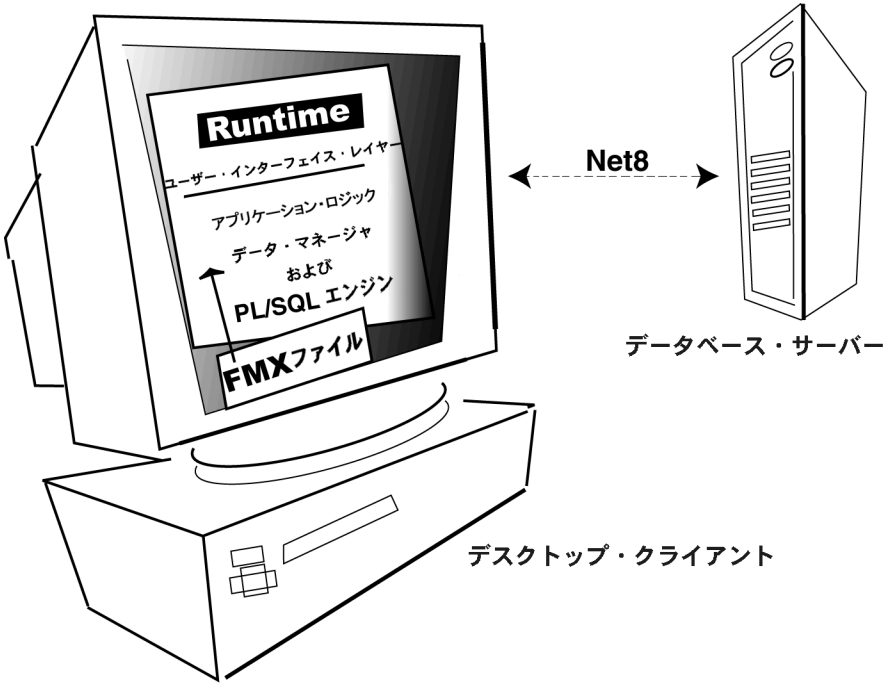

図 8-1 Forms Server のクライアント・サーバーベースのアーキテクチャ

8.3 Web ベースのアーキテクチャ

Web ベースの実装では、図 8-2 に示すように、Forms Server Runtime Engine およびすべてのアプリケーション論理は、クライアント・マシンではなくアプリケーション・サーバーにインストールされます。すべてのトリガー処理はデータベースおよびアプリケーション・サーバーで行われますが、ユーザー・インタフェース処理は、ユーザーのマシンにある Forms クライアントで行われます。

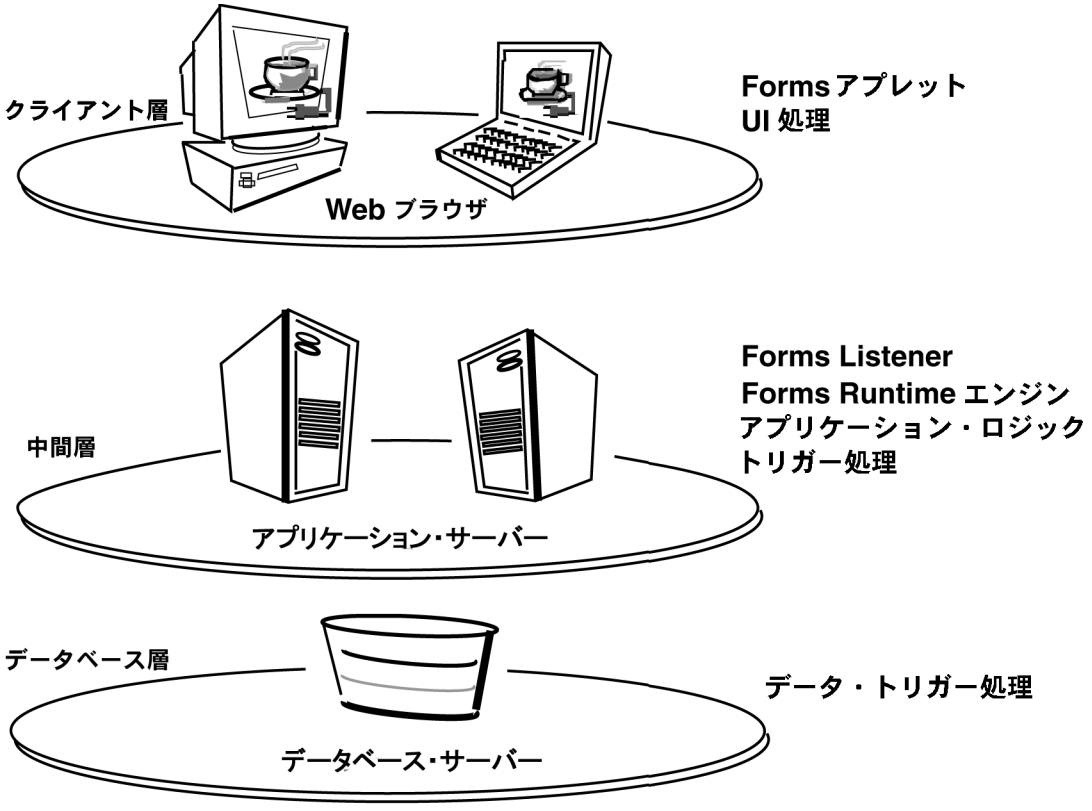

図 8-2 Forms Server の Web ベースのアーキテクチャ

8.4 移行に関するガイドライン

Forms Server の 3 層アーキテクチャ構成の他に、クライアント・サーバーから Web へ移行する場合にはいくつかの注意点があります。

たとえば、Web ベースのアプリケーションでは、次のような注意点があります。

- JPEG および GIF イメージ・タイプのみをサポートしているので、既存のイメージはこれらの形式に変換する。
- ファイル転送用に圧縮された JAR（Java アーカイブ）ファイルの使用がサポートされているので、Forms サーバーと Java クライアント間で大きいファイルの転送が必要な場合はいつでも JAR ファイルを使用する。
- ユーザー・インタフェースでは ActiveX、OCX、OLE、または VBX コントロールをサポートしない。かわりに、JavaBeans を使用して、ユーザー・インタフェースの機能を複製します。その他の Microsoft Windows ユーザー・インタフェースの依存性も JavaBeans で置換します。
- When-Mouse-Enter、When-Mouse-Leave、および When-Mouse-Move などの MouseMove トリガーはサポートしない。
- クライアントのハード・ドライブへの書込みアクセスは本来サポートしない。これは、トランスポータブルの Forms ユーザー・インタフェース用の JavaBeans を書き込むことで実行できます。
- Java フォントのみをサポートしているので、使用するフォントのタイプについてアプリケーションをチェックする。必要に応じて、Java フォントに変換します。Java は Registry.dat ファイルにあるフォント別名リストを使用します。次の表 8-1 に示すフォント別名がサポートされます。

表 8-1 Web ベースのアプリケーションでのフォントのサポート

Java フォント	Windows フォント	XWindows フォント	Macintosh フォント
Courier	Courier New	adobe-courier	Courier
Dialog	MS San Serif	b&h-lucida	Geneva
DialogInput	MS San Serif	b&h-lucidatypewriter	Geneva
Helvetica	Arial	adobe-helvetica	Helvetica
Symbol	Wingdings	itc-zapfdingbats	Symbol
Times Roman	Times New Roman	adobe-times	Times Roman

9

ネットワークに関する考慮事項

9.1 概要

Forms Server を適切に実装するためには、次の点を決定する必要があります。

- Web アプリケーションを配置するネットワークのタイプ
- ネットワークおよびセキュリティに関する問題の管理方法
- ネットワークにアクセスすることが予想されるユーザーの数とタイプ

この章では、Web アプリケーションを配置できるネットワークキング実装のタイプと、各タイプで Web アプリケーションを配置する場合に必要な事項について説明します。

9.2 ネットワーク・トポロジー

アプリケーションを配置できるさまざまなネットワークキング実装について説明する際には、多くの専門用語を使用します。通常、ネットワークは次の項目にグループ化できます。

- インターネット これは、インターネット・サービス・プロバイダ（ISP）へアクセスするすべての人にオープンなネットワークです。Internet Engineering Task Force (IETF) が起草したデータ転送標準を使用します。
- イントラネット これは、セキュリティー・ルールとネットワーク管理をコントロールする、単一の企業が「所有」しているネットワークです。
- エクストラネット これは、複数の企業によって " 所有 " されているネットワークで、各企業は独自のネットワーク・インフラストラクチャ、セキュリティ・ルール、およびユーザーを持つため、ネットワーク管理とセキュリティに関する統合的アプローチが必要になります。

インターネット、イントラネット、およびエクストラネットの主な違いは、イントラネットとエクストラネットはコントロールしている単一または複数の企業によって明確に定義され、ユーザーについて把握しているという点です。反対に、インターネットはユーザーについての明確な情報は入手できません。インターネットを介して通信するコンピュータとネッ

トワークは、接続するまで相手のことが分かりません。つまり、暗号化標準、ユーザー認証、認証など前もって調整できません。

これらの実装は次の項で詳細に説明します。

- インターネット
- イントラネット
- エクストラネット

9.2.1 インターネット

インターネットは、インターネット・サービス・プロバイダ（ISP）へアクセスするすべての人にオープンなネットワークです。ユーザーは、インターネットに接続して、世界中の他のネットワーク化されたコンピュータへアクセスします。インターネットに接続しているコンピュータが、ハードウェアやソフトウェアのセキュリティ・メソッドを使用して保護されていない場合、そのコンピュータ上のデータはインターネット上の他のどのユーザーからもアクセスされる可能性があります。

9.2.2 イントラネット

イントラネットは、セキュリティ・ルールとネットワーク管理をコントロールする、単一の企業が「所有」しているネットワークです。ネットワーク化されたコンピュータは、物理的に一つの場所（製造工場の目録管理に使用されるコンピュータなど）に置かれるか、または物理的に異なる場所（保険会社の各支店で使用されるコンピュータなど）にあります。

イントラネットは単一の企業によってコントロールされているので、ネットワークにアクセスしようとしているすべてのユーザーについて把握しており、ネットワーク構造、セキュリティ・ルール、およびソフトウェアを自由に選択することもできます。

次に示すのは、イントラネット・スタイルのネットワークの例です。

- ローカル・エリア・ネットワーク (LAN)
- 広域ネットワーク（WAN）。これは、ダイヤルアップ・アクセスにより、使用範囲をリモートの従業員まで拡張する LAN から構成されます。
- 専用通信回線を使用するインターコネクト LAN で構成される WAN
- 仮想プライベート・ネットワーク（VPN）。これは、通常は公共回線、ときにはインターネット・サービス・プロバイダ（ISP）を介して暗号化された安全な接続を作成する特別な " トンネリング " ソフトウェアを使用して、その使用範囲をリモートの従業員やネットワークに拡張する LAN または WAN で構成されます。

9.2.3 エクストラネット

エクストラネット は、複数の企業が「所有する」ネットワークで、各企業は、独自のネットワーク・インフラストラクチャ、セキュリティ・ルール、およびユーザーを持ちます。ネッ

トワーク化されたコンピュータは、通常物理的に異なる場所に置かれます。多くの場合、異なる企業がネットワーク・データ部分を互いに共有します。たとえば、旅行業界でエクストラネットを使用すると、旅行代理店は航空会社やツアー・オペレータが所有しているネットワークからのデータを利用して、航空機の予約や旅程の調整を行うことができます。

イントラネットと同様、エクストラネットではユーザーについて把握しています。ただし、エクストラネットは複数の企業によってコントロールされているので、ネットワーク管理およびセキュリティに関して統合化アプローチが必要です。旅行業界の例では、旅行代理店と航空会社は、旅行代理店が航空機予約情報にアクセスするためにネットワーキングとセキュリティの問題を調整する必要があります。

次に示すのは、エクストラネット・スタイルのネットワークの例です。

- リモート・ダイアルアップを使用してインターコネクトおよびアクセスする、複数の企業に属する LAN または WAN
- 専用回線を使用してインターコネクトおよびアクセスする、複数の企業に属する LAN または WAN
- 仮想プライベート・ネットワーク（VPN）。これは、通常は公共回線、ときには ISP を介して暗号化された安全なネットワーク接続を作成する特別な " トンネリング " ソフトウェアを使用して、使用範囲をリモート・ユーザーまで拡張する、複数の企業に属する LAN または WAN で構成されます。

エクストラネットを介して、ネットワーク化されたデータおよびアプリケーションを共有している企業は、ユーザー認証、認証およびデータ暗号化に関してセキュリティ・プロトコルに同意する必要があります。ファイアウォールやルータなどのセキュリティ・ハードウェアには、互換性が必要です。

9.3 ネットワーク環境における Forms Server の配置

Forms Server 機能および企業に最適なネットワーク設定タイプの決定方法について学ぶと、ネットワークに Forms Server をインプリメントできます。次の 5 つの項で、ネットワーキング・オプションと関連するリスクについて説明します。

- インターネットを介した配置
- ローカル・エリア・ネットワーク (Local Area Network: LAN) 上での配置
- リモート・ダイアルアップ・アクセスによるネットワークでの配置
- 公共回線でテレコムが提供する VPN を介したネットワークでの配置
- インターネットでの VPN アクセスを介したネットワークでの配置

9.3.1 インターネットを介した配置

Forms Server を使用すると、HTTP1.1 パケットに Forms メッセージをカプセル化して、Forms アプリケーションをインターネットに配置できます。HTTP はインターネット上にアプリケーションを配置するために最もよく使用されるプロトコルです。

多くの企業は、HTTP 通信のみ使用可にしてファイアウォールを " ロック・ダウン " することで、プライベート・ネットワークのセキュリティを大幅に向上させます。ファイアウォールを提供している会社の多くは、その製品で HTTP 標準をサポートしており、多くの企業は保有するプライベート・ネットワークの中を HTTP 通信が行きかうことを好意的に認めています。HTTP 通信のみ使用できるサイトでは、構成にほとんど何の変更も加えず、さらにクライアントに対して完全に透過な状態で、既存のファイアウォールを介して簡単に Forms Server を配置できます。

社内ネットワークを保護するために、厳格なセキュリティ・ルールが必要な場合でも、社内ネットワーク内のファイアウォールの後ろと非武装ゾーン（DMZ）にアプリケーション・サーバーを置くことができます。ファイアウォール内の HTTP フィルタは、VPN を使用しなくても受信トラフィックを制限するのに十分です。

また、より安全な通信を行うためには、HTTP とともに SSL (secure sockets layer) を使用できます。SSL はプライバシー、整合性、および認証を与える転送プロトコルです。SSL は、アプリケーション・レベルの 1 つ下のレベルである、転送レベルで動作します。つまり、SSL は、HTTP などのアプリケーション・レベルのプロトコルで処理される前に、メッセージを暗号化、復号化できます。

インターネット上に Forms Server を配置すると、Web 上の各ユーザーおよびエクストラネットのカスタマは、他のネットワーク配置オプションと比べて低費用でアプリケーションを使用できるようになります。企業は、スケーラブルで、安全で洗練された新規または既存の Forms アプリケーションをインターネット上で実行できます。

9.3.1.1 リスク

HTTP ソケット接続を使用して、インターネット上にアプリケーションを配置する場合、ユーザーの Forms Client PC の CPU 要件は、等価のパフォーマンスを提供するために Forms Server の以前のバージョンよりも多少高くなっています。

HTTP ラッパーで Forms データを送信するとネットワーク通信量が増える可能性があり、またより低スピードの接続で同時に実行できるセッション数に影響を与えることがあります。

9.3.1.2 その他のインターネット配置オプション

HTTP ソケット接続メソッドを使用しない場合、保護されたネットワークの外にアプリケーション・サーバーを含む DMZ を設定する他のオプションもあります。IP ルータをセットアップして、DMZ を保護するために、ポート 80（HTTP 通信）と 9000（Forms リスナーのデフォルト・ポート）の宛先となるパケット以外のすべての受信パケットをブロックできます。このアプローチを使用する場合、Forms Server Listener ポートが無防備であるというリスクがあります。複数の Forms Server Listener を使用する場合（たとえば、複数アプリケーションや複数言語をホストする場合）、リスクはさらに高くなります。

さらに、IP ルーターは、IP ルーターから DMZ 内のアプリケーション・サーバーにすべての受信トラフィックを再経路指定する、DMZ 内に常駐する複数ホームのファイアウォールによって支援する必要があります。アプリケーション・サーバーは、信頼性のある企業ネットワーク内のデータベースに接続する必要があるので、複数ホームのファイアウォールも、すべての Net8 トラフィックを信頼性のある企業ネットワーク内のデータ・サーバーに再経路指定する必要があります。

ローテーション・スケジュールは、異なる Forms Server Listener を異なる時間に使用して侵入を防ぎたい場所にセットアップできますが、これでは重大なハッカーを防げません。

内部ネットワークをハッカーの進入から守るためには、複数ホームのファイアウォールと内部ネットワーク間に追加のファイアウォールをセットアップして、IP パケットをフィルタし Net8 トラフィックのみ渡すようにすることをお薦めします。

9.3.2 ローカル・エリア・ネットワーク (Local Area Network: LAN) 上での配置

Forms アプリケーションにアクセスするすべてのユーザーが LAN 内に存在する場合、基本的な内部ネットワーク・セキュリティは十分であり、Forms Server に特別な構成をする必要はありません。

9.3.3 リモート・ダイアルアップ・アクセスによるネットワークでの配置

一部のユーザーが LAN 外または保護 WAN 外に存在し、ダイアル・インで Forms アプリケーションへアクセスする場合、リモート・アクセス・セキュリティ用に特別に設計されたサーバーが必要になります。このシナリオは、オフサイトで作業している従業員や貴社の LAN や WAN にアクセスする必要がある信頼できるカスタマには理想的です。このソリューションは、LAN にリモートでアクセスする必要があるユーザーが 1000 人より多い場合の実装には適しません。

リモート・アクセス・サーバーに登録されたユーザーが有効なユーザーです。登録されていないユーザーにはアクセス権がありません。リモート・アクセス・サービス（Remote Access Service : RAS）は、Windows NT サーバーの機能の 1 つです。Windows NT RAS サーバーは、このシナリオでリモート・アクセス・サーバーとして使用できます。

プライベート WAN は、専用線で構築されることがよくあります。侵入者は専用線の位置とデータを送信するのに使用する線の電線コードを知らなければ、侵入できません。このような条件下では、侵入されることはまず考えられません。

公共の電話回線を介してダイアルアップする場合は、機密データを送信時に暗号化することをお薦めします。Windows NT RAS サーバーには、Point-to-Point-Tunneling Protocol (PPTP) がインクルードされており、PPTP は、公共の電話回線を介して通信する機密データを暗号化する場合に使用できます。暗号化プロトコルを備えるリモート・アクセス・サーバーを使用していない場合については、以降の項を参照ください。ネットワークで Forms Server を構成するための、その他のより安全なオプションについて説明されています。

侵入者がリモート・アクセス・サーバーの電話番号にランダムにダイアルし、複数のユーザー名 / パスワードを組み合せて LAN にログインを試みるという場合、リスクはほとんど

ありません。ただし、リモート・アクセス・サーバーは、サーバーへのアクセス方法をすでに承知している悪意のある元従業員やカスタマに対しては非常に無防備です。

この問題を回避するには、次の予防策をお薦めします。

- 厳格なセキュリティ・レコードのメンテナンス。このメンテナンスにより、元従業員およびカスタマのエントリがリモート・アクセス・サーバー、自動ダイアルバック装置、およびすべての内部システムから削除されているか確認します。
- コール側 ID の検証。これは、登録されている電話番号のみがリモート・アクセス・サーバーにアクセスできるようにする方法です。
- 自動ダイアル・バック装置。この装置は以前に登録した電話番号を使用してコール側にコール・バックします。

9.3.4 公共回線でテレコムが提供する VPN を介したネットワークでの配置

前項で説明したように、従来型の WAN は通常専用線で構築されています。ただし、公共の電話回線を介してダイアルアップしている場合、ユーザー認証およびデータ送信には、より安全なメソッドを使用されることをお薦めします。

遠距離通信プロバイダから利用できる、VPN（仮想プライベート・ネットワーク）を使用するオプションがあります。遠距離通信プロバイダは、承認されたユーザーのリストを保持しており、認証済みのユーザーがダイアル・インする場合はいつでも VPN を作成します。使用しているネットワークに前項で説明したリモート・アクセス・サーバーが必要な場合は、前項のセキュリティの利点とリスクのすべてが適用されます。（このソリューションは、LAN にリモートでアクセスする必要があるユーザーが 1000 人より多い場合の実装には適しません。）

主なリスクは、サーバーへのアクセス方法をすでに知っており、VPN プロバイダの登録済みユーザー・リスト上に存在する、悪意のある元従業員またはカスタマに対して無防備であるということです。このリスクを回避するためには、リモート・アクセス・サーバーおよび VPN プロバイダの登録済みユーザー・リストの両方について、認証済みユーザーのリストを最新のものにしておくことを励行してください。

9.3.5 インターネットでの VPN アクセスを介したネットワークでの配置

ダイアルアップ・アクセスの方法としてインターネットを使用する予定がある場合は、ユーザー認証およびデータ送信について安全なメソッドを使用することをお薦めします。Forms Server HTTP ソケット構成、つまり HTTPS（改良済みプライバシ、整合性、および認証について安全なソケット層をもつ HTTP ソケット構成）を使用するオプションがあります。HTTP ソケットの詳細は、3.3 項「ソケット、HTTP または HTTPS」を参照してください。

インターネット上で VPN を使用する別のオプションもあります。このメソッドを使用すると、データは IP（インターネット・プロトコル）パケットのフォームでインターネットを介して転送されます。IP パケットは送信側および受信側の IP アドレスおよびビット（データ）のグループです。

インターネット上に VPN をセットアップすると、遠距離通信費を節約できます。リモート・ユーザーは、専用線や 800 番ではなくローカルの IPS にダイアルします。ネットワークで VPN ソフトウェアを構成およびメンテナンスする必要があり、ダイアル・インするユーザーには互換 VPN ソフトウェアが必要になります。インターネットを介して 2 つの LAN で通信している場所にエクストラネット接続をセットアップする場合は、すべての当事者が互換性のあるファイアウォールを使用する必要があります。リモートの作業者が存在する場合、リモートの作業者が使用できるモバイルのファイアウォールを提供するベンダーもありますが、これには多大な費用と管理時間が必要となります。

ほとんどの大手ファイアウォール・ベンダーは、インターネット上に VPN を実装するためのオプションを用意しています。お薦めできる VPN は、次のものを使用しています。

- 強力なユーザー認証。単純なユーザー・パスワードのメカニズムではなくリクエスト / 応答のメカニズムを含みます。
- ネットワークのより機密度の高い部分へのアクセスをコントロールする内部ファイアウォール
- 公共ネットワーク間でのデータ転送時にデータを保護するためのデータ暗号化（これは、各 IP パケットのデータを公共ネットワーク間で転送する前に暗号化し、受信先で非暗号化する「IP トンネリング」と呼ばれるものです）。

インターネット上での VPN のセットアップに関連するリスクを次に示します。

- HTTP ソケット接続を使用しない場合、ファイアウォールではデータを通過させません。ファイアウォールと Forms Server を構成すると、汎用プロキシをセットアップすることでこの問題を回避できる場合があります。
- 強力な認証とデータ暗号化に必要な追加の処理のために、ネットワーク・パフォーマンスが低下する場合があります。
- キーは適切に構成および管理してください。
- ファイアウォール構成は厳格に管理して、元従業員や元カスタマを登録解除してください。
- 潜在的なリスクとしてファイアウォール騙しがあります。（ファイアウォール騙しとは、侵入者が IP パケットに偽のアドレスを付与し、ネットワーク上の信頼性のあるノードと偽って侵入し、それらのパケットをユーザーのネットワークに送信することです。侵入者は、ネットワーク上のトラフィックを監視し、自分が付与したアドレスが受け付けれらるか否かを幾ケースもテストし当ネットワークで受け付けれられる IP アドレスを割り出すことです。）ファイアウォールにフィルタを使用すると、この妨害から守ることができます。

9.4 ネットワーク・セキュリティをメンテナンスするためのガイドライン

Forms Server を使用しているミッション・クリティカルなアプリケーションをインプリメントする場合、セキュリティは重要な問題になります。必要なネットワーク環境のタイプを判断してから、ネットワークを保護するためのセキュリティ方策を明文化します。詳細は、第 10 章「セキュリティに関する考慮事項」を参照してください。

アプリケーション・サーバーを起動して実行した後に、セキュリティを継続的にメンテナンスする必要があります。インターネットを介してアプリケーションにアクセスする場合は、サイトがハッカーによって侵入される場合があるので、メンテナンスは特に重要です。セキュリティ・ルールの強化は継続的なプロセスです。

イントラネット、エクストラネット、およびインターネットの Forms アプリケーションについて、いくつかの配置オプションを説明し、セキュリティに関連する影響を確認してきました。この結果、次のようにまとめることができます。

- ダイアルアップ WAN またはダイアルアップ VPN を使用しているイントラネットおよびエクストラネットの実装では、適切量の管理作業で合理的に安全性を確保できる。LAN の場合、内部から妨害を受けることが多いので、サーバーの保護とデータベース・ユーザーの管理を改善する必要があります。暗号化メカニズムは、認証されていないユーザーから機密データを保護するために是非使用してください。
- インターネット VPN を介してイントラネットおよびエクストラネットを実装する場合は、強力なアクセス・コントロールのみならず強力な認証と暗号化を使用してください。ほとんどの大手ファイアウォール・ベンダーは VPN オプションを備えており、認証されていないユーザーに対するアクセスのブロック、公共ネットワーク上でのデータの暗号化、およびユーザー認証を行うことができます。

インターネットでのセキュリティ方法の現実的な実装は、次の要素の組合せに基づきます。

- HTTP または HTTPS ソケット通信
- DMZ でのアプリケーション・サーバー
- 内部ネットワークを DMZ から保護するファイアウォール
- 可能な場合のデータ暗号化

10

セキュリティに関する考慮事項

10.1 概要

World Wide Web への関心が急速に高まる前から、インターネット上でユーティリティまたはプログラムを実行して、特定のリモート・コンピュータに接続し友人や仲間を見つけたり、相手側がログオンしているかどうかを確認することはよく行われていました。また、ネットワークを介してリアルタイムにその友人達と通信したり、相手側のディスク・ドライブに一時的に接続してファイルを交換することもできました。

インターネットは実際広範囲にオープンされていて、操作上、信頼レベルは高くても、セキュリティ・レベルは低いものでした。現在は、無数のユーザーが存在するため、セキュリティは重要な問題の 1 つになっています。企業は、ネットワークを保護することで、外部から独自のプライベート・ネットワークへ、コントロールされていないまたは不要なアクセスが行われないようにしています。

この章では、ネットワーク・セキュリティに関する問題を取り扱います。

10.2 共通システム・セキュリティの問題

次の項では、ネットワーク化された環境で Forms Server をセットアップする場合に考慮する必要がある共通のセキュリティの問題について説明します。

- ユーザー認証
- サーバー認証
- 認証
- 保護送信 (暗号化)
- ファイアウォール
- 仮想プライベート・ネットワーク（Virtual Private Network）(VPN)
- 非武装ゾーン (DMZ)

10.2.1 ユーザー認証

認証とは、ネットワークまたはデータベースにログインするユーザーにログイン権限を付与する検証プロセスのことです。認証の例としては、ローカル・エリア・ネットワーク（LAN）へログインする場合のユーザー名とパスワードの使用、およびインターネット上で安全な電子メールを送受信する場合のデジタル証明が含まれます。企業は、目標のセキュリティ・レベルおよび保護するネットワークやデータベースのタイプに応じて、さまざまなタイプの認証プロセスを使用できます。ただし、最終的な認証の目的は、承認されたユーザーのみがネットワークまたはデータベースおよびそのリソースにアクセスできるようにすることです。

Forms Server の場合、Web 上での Forms アプリケーションの実行は従来のクライアント / サーバー環境と似ており、アプリケーション・ユーザーは、ユーザー名 / パスワードを組み合せてユーザー自身を特定することで、データベース・ユーザーとしてログオンします。

Forms Server を使用して Forms アプリケーションをインターネット上の数百のユーザーに配置できるので、権限のないユーザーがネットワーク上で送信されるデータを（スニッファを使用して）不法に取り込んだり、認証情報を傍受したり、アプリケーションやサーバー環境へアクセスしたりする危険があるからです。このため、インターネット上にアプリケーションを配置する場合は、暗号化およびファイアウォールなどの追加のセキュリティ機能をインプリメントする必要があります。

10.2.2 サーバー認証

サーバー認証では、クライアント・マシンは、サーバーがリクエスト対象であるか検証します。たとえば、クライアントが機密データをサーバーに送信する場合、クライアントは相手側のサーバーが安全で、送信した機密データの正しい受信者であることを検証できます。

HTTPS 通信モード（SSL(secure sockets layer) をもつ HTTP を使用するモード）を使用する場合、データの送信は暗号化されサーバー認証が行われます。サーバー認証は、RSA コンプライアント・デジタル証明を使用して行われます。クライアントのブラウザがサーバーに接続する場合、サーバーはその証明を表示します。クライアントとサーバーは認証局（CA）から証明を取得します。CA は、個人または企業の識別情報を検証した後でのみ個人または企業に証明を発行する企業です。CA の 1 つに VeriSign, Inc があります。HTTPS モードを使用することにした場合、証明リクエストを作成して証明を管理するためには、Oracle Wallet Manager をインストールする必要があります。詳細は 5.5 項「HTTPS 接続モード設定の追加ステップ」を参照してください。

10.2.3 認証

認証とは、ユーザーが必要とするネットワークまたはデータベース・リソースに対するアクセス権を認証済みユーザーに付与するプロセスです。また、認証により、ユーザーは不要または使用する権限のないリソースへアクセスできません。たとえば、マネージャは従業員の給与台帳情報が記載してある表へのアクセスは認められても、在庫管理事務員はこの情報へのアクセスは認められません。ネットワークおよびデータベース・リソースでの認証を実行

する場合に使用するメソッドは、目標のセキュリティのレベル、および保護されているネットワークまたはデータベースのタイプによって異なります。

Forms Server の場合、ユーザーが認証されると、データベース・ロールがユーザーに割り当てられ、その結果データベース内のデータを参照または変更する権限が与えられます。（これは認証のフォームの 1 つです。）ユーザー ID もアプリケーション・ロールを設定する場合に使用されます。

10.2.4 保護送信 (暗号化)

情報が通信回線を介して送信されると、その通信回線が同軸ケーブル、電話回線、光ファイバー、衛星のいずれであろうとも、通信はサードパーティにより傍受可能であるというリスクがあります。データが漏洩していることを送受信者が気づかれずに、情報が傍受される可能性があります。

最もよく使用される送信保護のためのメソッドは、データを暗号化することです。暗号化を使用すると、データの送受信者は情報をエンコードおよびデコードできる「キー」を持ちます。データを送信すると、送信者側のキーは数学的アルゴリズムを使用して情報をエンコードするために使用されます。受信者側のキーは情報をデコードします。サード・パーティがデータの送信中にエンコードされたデータを傍受しても、サード・パーティがキーへのアクセス権を取得するかまたはアルゴリズムのコードを「破らない」限り、データは判読不能で使用できません。

データを暗号化するのに使用するメソッドは、目標のセキュリティ・レベルとデータを送信するネットワーク・タイプによって異なります。たとえば、対称型暗号化は、ネットワークのスピードが重要である場合に使用できます。ポピュラーな対称型暗号化システムでは、RC-4 およびデータ暗号化規格（DES）を使用します。非対称型暗号化は非常に安全ですが、ネットワーク・パフォーマンス維持に費用がかかります。ポピュラーな非対称型暗号化システムでは、Diffie-Hellman (DH) および Rivest-Shamir-Adleman (RSA) を使用します。

ネットワーク、ファイアウォール、または VPN にインクルードされている暗号化メソッドを確認する必要があります。Forms Server では、データ送信の安全性を高めるため、次の暗号化オプションを提供しています。

- HTTPS 通信モードこのモードは SSL (secure sockets layer) つきの HTTP です。SSL は、HTTP などのアプリケーション・レベルのプロトコルで処理される前に、（RC4 暗号化を使用して）メッセージを暗号化、非暗号化できます。SSL は、RSA コンプライアント・サーバー認証も提供します。HTTPS モードのセットアップ方法の詳細は、5.5 項「HTTPS 接続モード設定の追加ステップ」を参照してください。
- ORA_ENCRYPT_LOGIN この環境変数を使用して、Forms Server ログインするためのユーザー名とパスワードを暗号化します。
- DBLINK_ENCRYPT_LOGIN この環境変数を使用して、データベースへログインするためのユーザー名とパスワードを暗号化します。

- FORMS60_MESSAGE_ENCRYPTION この環境変数を使用して、RC4 40 ビットの暗号化を使用する Forms メッセージを暗号化します。ソケットおよび HTTP 通信モードにのみ適用されます。（デフォルトで、通信は暗号化されます。）
- FORMS60_HTTPS_NEGOTIATE DOWN この環境変数を使用して、128 ビットのサーバーに、より低いレベルの暗号化で構成されているクライアントの処理方法を指示します。TRUE を設定すると、サーバーはクライアントで使用可能な最高レベルの暗号化を使用できます。FALSE に設定すると、サーバーは、クライアントが 128 ビットの暗号化を使用しない限り、クライアント・リクエストを拒否します。
- DSA（デジタル署名アルゴリズム）このアルゴリズムは、Forms Server アプレットでデジタル署名を行う場合に使用されます。
- Net8 SNS/ANO この暗号化スキームは、データベースと Forms Server 間の送信を暗号化するために使用します。

10.2.5 ファイアウォール

ファイアウォールとは、通常、ネットワークで受信可能なデータのタイプをフィルタするハードウェアとソフトウェアの組合せです。たとえば、ファイアウォールは保護されたネットワークへ HTTP 通信のみ通れるよう設定できます。また、ファイアウォールは、ネットワークの IP アドレスを無名にしておくことで、外部コンピュータからアクセスできないようにします。ネットワークへのアクセスを認証および権限付与された外部のトラフィックは、ファイアウォールの IP アドレスからネットワークの IP アドレスに再送信されます。ファイアウォールは、プライベート・ネットワークの、侵入に対する防御の最初のラインです。

ネットワーク・セキュリティ・システムにファイアウォールをインクルードする場合は、標準のソケット接続ではなく、必ず HTTP ソケット接続または HTTPS ソケット接続を使用するように、Forms Server リスナーを構成します。これは、ファイアウォールが、標準の Forms メッセージングを含む、パケットまたはポート・レベルでの多くの共通サービスを使用不可にするためです。HTTP はファイアウォールを介して渡すことができるサービスです。

10.2.6 仮想プライベート・ネットワーク（Virtual Private Network）(VPN)

仮想プライベート・ネットワーク（VPN）とは、通信が完全なプライベートとみなされる場所の 2 つのネットワーク間またはネットワークとリモート・ユーザー間の認証済み接続です。ネットワークとリモート・ユーザーのコンピュータの両方にある特別な「トンネルリング」ソフトウェアにより、インターネット・サービス・プロバイダ（ISP）を介して、公共回線で保護され、暗号化された接続が作成されます。リモート・ユーザーが VPN ソフトウェアを正しく設定していなかった場合は、ネットワークに VPN を作成できません。

VPN セットアップにファイアウォールがインクルードされていることがよくあります。標準のソケット接続ではなく、必ず HTTP ソケット接続または HTTPS ソケット接続を使用するよう Forms Server リスナーを設定してください。これは、ファイアウォールが、標準の

Forms メッセージングを含む、パケットまたはポート・レベルでの多くの共通サービスを使用不可にするためです。

注意 : HTTP およびソケットの詳細は、第 3.3 章「ソケット、HTTP または HTTPS」を参照してください。

10.2.7 非武装ゾーン (DMZ)

非武装ゾーン（DMZ）とは、機密情報を含まないネットワーク内の隔離された環境のことです。たとえば、アプリケーション・サーバーを非武装ゾーン内に置き、すべてのデータベース・サーバーは保護されたネットワーク内に置くネットワークをもつことができます。そのようにすると、非武装ゾーンのセキュリティが危険な状態である場合でも、機密データは侵入者に漏洩されません。

10.3 セキュリティ改善のための簡単なステップ

次に示すステップにより、ネットワーク・セキュリティに関するリスクを減らすことができます。

- ユーザーが自分のユーザー名 / パスワードを、承認されていないユーザーに貸与しないようにする。
- 注文受付事務員、役員、製品販売員などの、さまざまなユーザーのプロファイルに合致する明確なデータベース・ロールによって、厳格な認証スキームを実行する。各ロールにより、ユーザー・プロファイルに応じてデータを変更する権限または参照する権限も制限します。
- サーバーまたはデータベースにアクセスする必要のないユーザーを削除するか、パスワード・エイジングを実行して、ユーザー・アカウントを慎重に管理する。
- 暗号化およびデジタル証明認証には、HTTPS 接続モードを使用する。
- ORA_ENCRYPT_LOGIN および DBLINK_ENCRYPT_LOGIN を使用して、送信時にユーザー名とパスワードを暗号化する。
- 侵入者に対する機密データの漏洩を避けることができる場合は常に、FORMS60_MESSAGE_ENCRYPTION および Net8 SNS/ANO などの暗号化を使用する。

次に、確実に実行するようで見落としやすいネットワーク・セキュリティの考慮事項を示します。

- 未認証のユーザーが建物に侵入したりアクセスできないようにするために、サーバー・マシンに対する物理アクセスをコントロールします。
- バックアップ・メディアの保護ストレージなど、厳格なデータ・バックアップ・システムをインプリメントします。
- テルネットや ftp などの簡単に侵入できるサービスの使用をやめるかまたは最小限に抑えます。

- すべてのセキュリティ関連オペレーティング・システム・パッチをインストールします。

11

パフォーマンス・チューニングに関する考慮事項

11.1 概要

この章では、Forms Server を使用してインターネットまたはその他のネットワーク環境上にアプリケーションを配置する場合に発生するチューニング上の考慮事項について説明します。この章では、アプリケーション・サーバー上のネットワークおよびリソースについて取り扱います。Forms Server とデータベース・サーバー間の接続のチューニングについては取り扱いません。

11.2 Forms Server のビルトイン最適化機能

Forms Server および Java クライアントには、いくつかの最適化機能が含まれており、大きく次の項目に分類できます。

- クライアント・リソース要件の最小化
- Forms Server リソース要件の最小化
- ネットワーク使用量の最小化
- ネットワークを介して送信されるパケットの効率の拡大
- クライアントでのアプリケーション画面の効率的なレンダリング

11.2.1 クライアント・リソース要件の最小化

Java クライアントは、主にアプリケーション画面のレンダリングを行います。Java クライアントには、埋込みアプリケーションのロジックはありません。Java クライアントをロードすると、複数のフォームを同時に表示できます。すべての Forms Server アプリケーションに対応する汎用 Java クライアントを使用すると、各アプリケーションにカスタマイズされた Java クライアントと比較して、クライアント上のリソースが少なくて済みます。

Java クライアントは、多くの Java クラスで構成されています。これらのクラスは、スプラッシュ画面の表示、ネットワーク通信およびルック・アンド・フィールの変更などの、機能サブコンポーネントにグループ化されます。機能サブコンポーネントを使用すると、Forms Developer および Java 仮想マシン（JVM）は、すべての機能クラスを一度にダウンロードせず、必要に応じて機能をロードできます。

11.2.2 Forms Server リソース要件の最小化

フォームの定義を FMX ファイルからロードすると、実行プロセスのプロファイルは次のものに要約できます。

- Encoded Program Units
- Boilerplate Object/Image
- Data Segments

これらの中で、Data Segments セクションのみがアプリケーションの指定したインスタンスに対して一意です。Encoded Program Units および Boilerplate Objects/Images はすべてのアプリケーション・ユーザーに対して共通です。Forms Server は共有コンポーネントを物理メモリーにマップして、同じ FMX ファイルにアクセスするすべてのプロセスでそのコンポーネントを共有します。

指定した FMX ファイルをロードする最初のユーザーは、そのフォームに必要な全メモリー量を使用します。ただし、後続のユーザーの場合は必要なメモリー量が大幅に減らされているので、ローカル・データのエクステントにのみ依存します。共有コンポーネントをマップするこのメソッドを使用すると、指定したアプリケーションに必要な、ユーザーごとの平均メモリー量を減らすことができます。

11.2.3 ネットワーク使用量の最小化

帯域幅は重要なリソースで、インターネット・コンピューティングの一般的な広がりとともに、インフラストラクチャにますます大きな負担を強いるようになっています。このため、アプリケーションはネットワークの容量を節約して使用することが重要です。

Forms Server は、メタデータ・メッセージを使用する Java クライアントと通信します。メタデータ・メッセージは、実行対象のオブジェクトとその実行方法をクライアントに通知する名前と値のペアのコレクションです。パラメータのみを Java クライアント上の汎用オブジェクトに送信することで、(同じ効果になるよう新規コードを送信した場合と比較して) 通信量を約 90% 減らすことができます。

Forms Server では、次の 3 つの方法で効果的にデータ・ストリームを圧縮します。

- 同じようなメッセージの集合（名前と値のペアのコレクション）を送信すると、2 番目以降のメッセージには、前のメッセージとの相違点のみ含まれます。この結果、ネットワーク・トラフィックを大幅に減らすことができます。このプロセスは、*message diff-ing* と呼ばれます。
- 同じ文字列がクライアント画面で繰り返されると（たとえば、同じ企業名が記載されている複数行のデータが表示される場合)、Forms Server はその文字列を一度のみ送信し、後続のメッセージではその文字列を参照します。参照によって文字列を渡すことで、帯域幅の効率は向上します。
- データ・タイプはその値に必要な最小のバイト数で送信されます。

11.2.4 ネットワークを介して送信されるパケットの効率の拡大

待ち時間は、アプリケーションの応答時間に影響を与える最も重要な要因です。待ち時間の影響をなるべく受けないようにする最もよい方法の 1 つは、Java クライアントと Forms Server 間で、対話中に送信されるネットワーク・パケットの数を最小限にすることです。

Forms Developer モデル内のトリガーを多数使用すると大きな効果がありますが、各トリガーにネットワークの往復が必要なため、待ち時間の影響が大きくなります。トリガーに関連する待ち時間を避ける方法の 1 つは、イベント・バンドルを介してトリガーをグループ化することです。たとえば、ユーザーが項目 A から項目 B にナビゲートする場合（あるエントリ・フィールドから別のフィールドへタブする場合など)、トリガー実行前後の範囲には、それぞれ Forms Server 上での処理が必要です。

イベント・バンドルは、2 つのオブジェクト間をナビゲートしている間にトリガーされたすべてのイベントを集めて、それらを単一のパケットとして Forms Server に配布して処理します。ナビゲートに多くのオブジェクトの問合せが関連している場合（離れているオブジェクト上でマウスのクリックを行った場合など)、イベント・バンドルは問い合されたすべてのオブジェクトからすべてのイベントを集めて、そのグループを単一のネットワーク・メッセージとして Forms Server に配布します。

11.2.5 クライアントでのアプリケーション画面の効率的なレンダリング

指定したフォーム内のすべてのボイラープレート・オブジェクトは仮想グラフィック・システム（VGS）ツリーの一部です。VGS は、すべての Forms Developer 製品に共通の図形サブコンポーネントです。VGS ツリー・オブジェクトは、座標、カラー、線幅およびフォントなどの属性を使用して記述します。オブジェクトの VGS ツリーを Java クライアントに送信する場合、送信する属性のみが指定したオブジェクト・タイプのデフォルトと異なる属性になります。

イメージは圧縮された JPEG イメージとして送信および格納されます。これにより、ネットワーク・オーバーヘッドとクライアントの必要なメモリー量の両方を減らすことができます。

リソースの最小化には、クライアントおよびサーバー・プロセスのメモリー・オーバーヘッドの最小化も含まれます。ネットワークを最適な状態で使用するには、帯域幅を最小に維持し、ネットワークの待ち時間の影響も含まれるため、クライアントおよび Forms Server 間の通信に使用するパケット数を最小化することが必要です。

11.3 Forms Server アプリケーションのチューニング

アプリケーションの開発者は、Forms Server のビルトイン・アーキテクチャの最適化機能により最大の利益を得ることができます。この章の後半では、多くのアプリケーションに影響を与える主要なパフォーマンスの問題および開発者がアプリケーションをチューニングしてパフォーマンスを改善し、Forms Server 機能を活用するための方法について説明します。

取り上げる内容は次のとおりです。

- データ・サーバーに関連する Form Server の位置
- アプリケーションの起動時間の最小化
- 必須ネットワーク帯域幅の削減
- パフォーマンスを改善するためのその他の方法

11.3.1 データ・サーバーに関連する Form Server の位置

Java クライアントを Forms Server へ接続する場合、イベント・バンドルなどの機能を使用してネットワーク待ち時間の影響を効率的に抑えることができます。*message diff-ing* を使用して、ネットワーク帯域幅を削減します。一方、Forms Server とデータ・サーバー間に存在するクライアント・サーバーの関係で、ネットワークの往復通信の遅延や混雑はさらに許容できません。

これらの理由から、Forms Server はデータ・サーバーと同じ高速 LAN 上に置くことが最良で、この結果 Forms Server をユーザーからさらに離れた場所に置くことがあります。これは、ユーザーの近くにサーバーを置くという標準規則に反しているように思えますが、従来のクライアント・サーバーのインプリメンテーションと比較して、ネットワークでの Forms Server の効率を改善した結果です。

最適構成では、図 11-1 に示すように、Form Server およびデータ・サーバーはデータ・センター内の同じ場所に配置されます。これはお薦めの設定方法ですが、クライアントは低帯域幅（モデム）と長い待ち時間（衛星）の接続を介してサーバーにアクセスします。

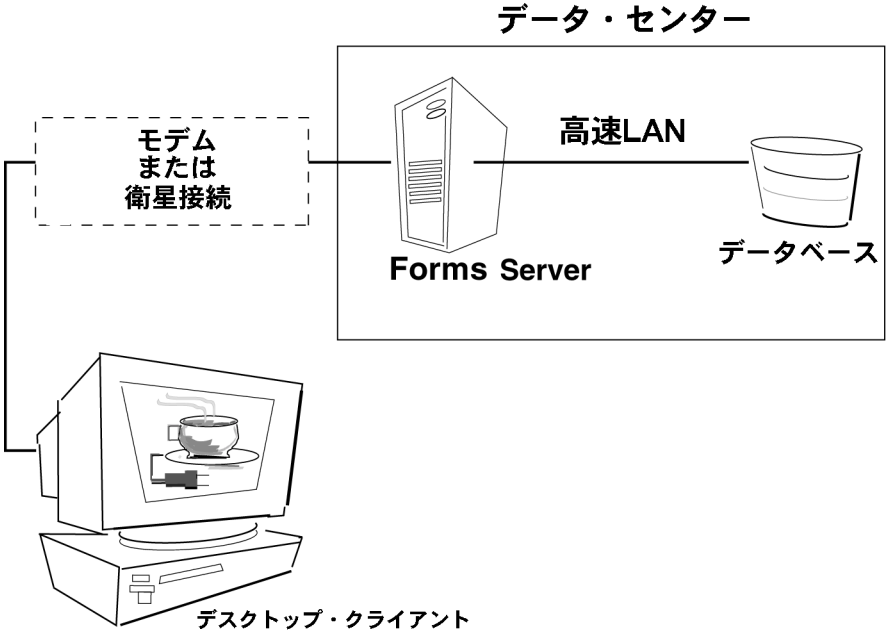

図 11-1 Forms Server およびデータ・サーバーの同じ場所への配置

11.3.2 アプリケーションの起動時間の最小化

アプリケーションをロードするためにかかる時間は、第一印象として重要であり、またどのユーザーにとっても主要な基準となります。起動時間は、オーバーヘッドとみなされます。起動時間は、今後のパフォーマンスを期待させるものでもあります。業務でクライアント・テクノロジを使用する場合、クライアント・コードのロードに必要な追加のオーバーヘッドは、ユーザーにあまりよい影響を与えません。したがって、可能な限りロード時間を最小化することが重要です。

Forms アプリケーションを要求した後に、次のステップを実行してからアプリケーションを使用します。

1. Java 仮想マシン (JVM) を起動します。
2. すべての初期 Java クライアント・クラスをロードし、クラスのセキュリティを認証します。
3. スプラッシュ画面を表示します。
4. フォームを次に示す方法で初期化します。
   a. 必要に応じて、追加の Java クラスをロードします。
   b. クラスのセキュリティを認証します。
   c. ボイラープレート・オブジェクトとイメージをレンダリングします。
   d. 初期画面ですべての要素をレンダリングします。
5. スプラッシュ画面を削除します。
6. フォームを使用する準備ができます。

アプリケーション開発者は、JVM を起動するのにかかる時間についてほとんど何も行うことができません。ただし、Java 配置モデルおよび Form Java クライアントの構造では、開発者はロードする Java クラスとその方法を決定できます。これにより、Java クラスに必要なロード時間を最小化します。

Java クライアントには、基本機能のクラス（ウィンドウを開くなど）と特定の表示オブジェクトの追加クラス（LOV 項目など）のコア集合が必要です。これらのクラスはサーバーに始めから常駐している必要がありますが、次の方法を使用すると、これらのクラスをクライアントの JVM にロードするのに必要な時間を短縮できます。

- JAR ファイルの使用
- キャッシュの使用
- 需要に応じた遅延ロード

11.3.2.1 JAR ファイルの使用

Java は Java アーカイブ（JAR）メカニズムを提供して、クライアントにネットワークを介して効率的な配布を行うために、クラスをまとめてグループ化し、圧縮（Zip 形式）すること

ができるファイルを作成します。このファイルをクライアントで使用すると、今後の使用のためにキャッシュされます。

Form Server は、次に示す構成済みの JAR ファイルを提供して、通常の配置シナリオをサポートします。

ファイル名	使用方法	説明
f60all.jar	任意	すべてのランタイム状況において Java クラス・ファイルの全体的な集合を含む。
f60common.jar	必須	アプレットごとに必要。

f60generic_laf.jar	任意	アプリケーションを Generic lookAndFeel ランタイム設定で配置する場合、または lookAndFeel 設定を指定していない場合にロードする。
		`<APPLET ...>`
		`<PARAM NAME="lookAndFeel" VALUE="Generic">`
		`...`
		`</APPLET>`

f60oracle_laf.jar	任意	アプリケーションを Oracle lookAndFeel ランタイム設定で配置する場合にのみロードする。
		`<APPLET ...>`
		`<PARAM NAME="lookAndFeel" VALUE="Oracle">`
		`...`
		`</APPLET>`

f60splash.jar	必須	アプレットごとに必要。
f60tree.jar	任意	Forms アプリケーションは、階層ツリー・コントロールを使用する場合にのみロードする。

1 つ以上の JAR ファイルをアプレットに指定するには、参照している HTML ファイルの <APPLET> タグで ARCHIVE パラメータを指定します。例 :

```
<APPLET CODEBASE="http://www.server.com/webcode/"
ARCHIVE="f60all.jar, icons.jar"
CODE="oracle.forms..">
```

11.3.2.2 キャッシュの使用

Form Server のサポート済み JVM は両方（Oracle JInitiator および Oracle JDK）とも JAR ファイルのキャッシュをサポートします。JVM がクラスを参照する場合、最初にローカルのクライアント・キャッシュをチェックして、クラスがキャッシュ済み JAR ファイル内に存在するか確認します。クラスがキャッシュ内に存在する場合、JVM はサーバーをチェックし

て、JAR ファイルの現行バージョンがあるか確認します。見つからない場合は、クラスはネットワークを介してではなくローカル・キャッシュからロードされます。

キャッシュは、効率性を最大化するために適切なサイズにします。キャッシュ・サイズが小さすぎると、有効な JAR ファイルが上書きされてしまう場合があります。その結果、アプリケーションを再度起動すると、別の JAR ファイルのダウンロードが必要になります。デフォルトのキャッシュ・サイズは 20MB です。このサイズは、アプリケーションを正常に実行した後のキャッシュ容量のサイズと比較する必要があります。

JAR ファイルはロード元のホストに関連してキャッシュされます。これには、異なるサーバーからの同一の JAR ファイルがキャッシュを埋めることができるロード・バランス・アーキテクチャという含意があります。JAR ファイルを中央に置き、ロード・バランス構成の各サーバーでそれらを参照することで、開発者は各 JAR ファイルの 1 つのコピーのみがクライアントのキャッシュでメンテナンスされていることを確認できます。この方法を使用すると、JAR ファイル内の特定のクラスに署名して、ロード元のサーバー以外のサーバーに接続を戻せるようにする必要があります。Oracle が提供する JAR ファイルでは、事前にクラスに署名してあります。

11.3.2.3 需要に応じた遅延ロード

JAR メソッドの欠点は、実行を継続する前に JAR ファイル内のすべてのクラスをロードし、JVM によって妥当性チェックする必要があるという点です。JAR ファイルの便利な点として、他の JAR ファイルを参照して、指定したアーカイブ内に格納するクラスの数を制限できるという機能があります。JVM は、アプリケーションが要求する順序で必要な JAR ファイルにナビゲートできます。

Oracle が提供する f60splash.jar ファイルには、クライアントを初期化し、初期スプラッシュ画面を表示するのに十分なロジックがあります。また、このファイルには、他の JAR ファイルに含まれるファイルの遅延参照もあるので、これらのファイルは需要に応じて続いてロードされます。需要に応じた遅延ロードを使用するには、f60splash.jar ファイルを HTML ページが参照する最初の JAR ファイルに設定する必要があります。

11.3.3 必須ネットワーク帯域幅の削減

開発者は、メッセージ間で異なる情報のみを送信する *message diff-ing* を使用して、データ・ストリーム圧縮を最大限利用できるようアプリケーションを設計できます。次のステップを実行すると、メッセージ間の相違部分を削減できます。

- **メッセージ送信順序のコントロール。**メッセージの送信順序は次の 2 つの基準で管理されます。
  - 初期画面の場合は、オブジェクト・ナビゲータの表示順
  - 実行中、プログラムの順序は項目プロパティに変更

  その結果が有用性に影響を与えない場合は、同じキャンバス上にある似たようなオブジェクトをオブジェクト・ナビゲータ内で並べて配置するようにします。たとえば、ボタンの次にボタン、テキスト項目の次にテキスト項目と配置します。(項目プロパティ「次ナビゲーション項目」を使用する場合、フォーム内の項目でもナビゲーションと同じ順序が使用されます。) オブジェクト・ナビゲータで似たような項目をまとめて順序付けすると、最初のフォームを表示するためにクライアントに送信された項目プロパティには、多くの似たような項目が連続して含まれるので、*message diff-ing* アルゴリズムが効率的に機能します。

  さらに、トリガーまたは他のロジックを使用して項目プロパティを変更する場合、別の表示タイプの項目プロパティを変更する前に似たような項目のプロパティをまとめてグループ化する必要があります。例:

```
set_item_property(text_item1_id, FONT_WEIGHT, FONT_BOLD);
set_item_property(text_item2_id, FONT_WEIGHT, FONT_BOLD);
set_item_property(text_item3_id, FONT_WEIGHT, FONT_BOLD);
set_item_property(button_item1_id, LABEL, 'Exit');
...
```

- **オブジェクト間の類似点を活用。**似たようなオブジェクトを使用すると、(ユーザーに視覚的によりアピールする上に) *message diff-ing* の効率性が向上します。次の手順で、オブジェクト間の一貫性が図られます。
  - プロパティのデフォルト値を受け入れ、オブジェクトに必要な属性のみを変更する。
  - スマート・クラスを使用して、オブジェクトのグループを記述する。
  - ルック・アンド・フィールを少数の可視属性にロックする。
- **ボイラープレート・テキストの使用を削減。**開発者である場合、可能な限り、ボイラープレート・テキストではなく PROMPT 項目プロパティを使用してください。Forms Developer 6.0 以降には、Associate Prompt 機能が含まれており、この機能を使用して、ボイラープレート・テキストを指定した項目のプロンプトとして再設計できます。

- **ボイラープレート項目 (円弧、円、多角形など) の使用を削減。**指定したフォームのすべてのボイラープレート項目をフォームの初期化時にロードします。ボイラープレート項目をロードしてクライアント上でリソースを使用するには、表示の有無にかかわらず時間がかかります。共通のボイラープレート項目（矩形と線）は最適化されます。このため、アプリケーションをこれらの基本的なボイラープレート項目に制限すると、起動時間を短縮しながらネットワーク帯域幅とクライアント・リソースを削減できます。
- **ナビゲーションを最小に維持。**Event Bundle は、2 つまたはそれ以上のオブジェクトにナビゲーショを拡張する場合でも、ナビゲーション・イベントが完了するたびに送信されます。デフォルト値が受け入れられているときに、フィールド間をナビゲートする必要がないフォームを設計します。フォームは、完了したらユーザーが素早く終了できるようにしてください。このようにすると、すべての追加のナビゲーション・イベントは 1 つの Event Bundle として実行されます。
- **初期画面を表示する時間を短縮。**Java クライアントは必須クラスをロードすると、初期画面を表示する前に、表示するすべてのオブジェクトをロードして初期化する必要があります。項目数を最小限に抑えることで、初期画面は移入され、より迅速に表示されます。初期画面を表示する時間を短縮する方法は次のとおりです。
  - オブジェクトの集合（タイトル、小さなロゴ、ユーザー名およびパスワードなど）を制限して、アプリケーションのログイン画面を提供します。
  - Form の初期画面では、要素の非表示は直ちに必要としません。次のキャンバス・プロパティを使用します。

```
RAISE ON ENTRY = YES （キャンバスのみ）
VISIBLE = NO
```

  1 つのシートのみ表示されるいくつかのシートで構成される TAB キャンバスには注意してください。タブ間で応答を切り替える場合、キャンバス上のすべてのシートに対するすべての項目がロードされます。この中には初期タブに後ろに隠れている項目も含まれます。この結果、TAB キャンバスをロードして初期化するためにかかる時間は、始めに可視できるオブジェクトのみではなく、キャンバス上のすべてのオブジェクトに関連します。

- **MENU_BUFFERING を使用不可に設定。**デフォルトでは、MENU_BUFFERING は True に設定されます。これは、変更されたメニューが完全な状態で再送信されるときに、メニューに対する変更内容が今後の " 同期化 " イベントのためにバッファされることを意味します。（ほとんどのアプリケーションは同時に複数の変更を行うことも、まったく行わないこともできます。したがって、クライアント側のメニューを更新する最も効率のよい方法は、すべてのメニューを一度に送信することです。）ただし、指定したアプリケーションはメニューに最小限の変更しか加えない場合があります。この場合、変更するたびに変更内容を送信した方が効率的です。これは、次の文を使用して実行できます。

```
Set_Application_Property (MENU_BUFFERING, 'false');
```

メニューのバッファは、LABEL、ICON、VISIBLE および CHECKED のメニュー・プロパティにのみ適用します。ENABLE/DISABLE イベントは常に送信されますが、メニュー全体を再送信する場合は不要です。

11.3.4 パフォーマンスを改善するためのその他の方法

次に示す方法を使用すると、アプリケーションを実行するのに必要なリソースをさらに削減できます。

- **MOUSE-UP、MOUSE-DOWN トリガーの使用を制限。**Java モデルでは、イベントはマウス・ボタンの動作を検出するとトリガーが発行される必要があります。イベントは、Forms Server に渡されて、このイベントが MOUSE-UP または MOUSE-DOWN のどちらであるか判別されます。指定したアプリケーションは 1 つのトリガーのみ（MOUSE-DOWN など）定義できますが、イベントを処理するためにトリガー・コードを指定していない場合でも、イベントは、関連（MOUSE-UP）イベントについてクライアントにより生成されます。マウス・イベントは非同期のため、通常のイベント・バンドルモデルの外部で処理されます。

- **タイマーを確認して JavaBeans で置換。**タイマーを起動すると、非同期イベントが生成されます。このイベントとまとめられる他のイベントはキュー内にありません。タイマーのサイズはほんの数バイトですが、毎秒実行されるタイマーは、通常の作業日には毎分 60 のネットワーク・トリップと、およそ 30,000 パケットを生成します。多くのタイマーは、時計または動画を提供するために使用されます。これらのコンポーネントを、Forms Server やネットワークの介入がなくても同じ効果をもたらす、自己完結型の JavaBeans と置換します。

- **入力項目の妥当性チェックのローカライズについて考慮。**When-Validate-Item トリガーを使用して項目に対する入力を処理することはよく行われます。トリガー自体は、Forms Server で処理されます。移動可能な Java コンポーネントを使用して、標準のクライアント項目（テキスト・ボックスなど）のデフォルト機能を置換することを考慮する必要があります。次に、日付や最大 / 最小値などの項目の妥当性チェックを項目内に含めます。この方法を使用すると、より複雑な、入力の自動フォーマットなど（(XXX) XXX-XXXX の書式をもつ電話番号など）のアプリケーション固有の妥当性チェックを行うことができます。

- **アプリケーションを、大きな 1 つのフォームではなく多数の小さいフォームに縮小。**細かく分けられたアプリケーションを指定すると、ユーザーのナビゲーションは、Form Server からロードおよび初期化されるオブジェクトを定義します。大きなフォームの場合、オブジェクトの初期化中にアプリケーションが遅延して、アプリケーションの多くが参照できなくなるという危険性があります。フォームをまとめて連鎖する場合は、ビルトインの OPEN_FORM および NEW_FORM を使用することを検討します。

  - OPEN_FORM を使用すると、コールしているフォームはクライアントとサーバーにオープンされたままの状態になるので、クライアントとサーバー両方の追加のフォームは多くのメモリーを消費します。ただし、フォームが別のユーザーによっ

て使用中である場合、サーバーのメモリー使用量は、データ・セグメントにのみ制限されます。ユーザーが初期フォームに戻ると、フォームはすでにローカル・メモリー内に常駐し追加のネットワーク・トラフィックを再表示する必要はありません。

- NEW_FORM を使用すると、コールしているフォームはクライアントとサーバー上でクローズされて、すべてのオブジェクト・プロパティが破棄されます。このため、サーバーおよびクライアント上で使用するメモリー量は少なくなります。初期フォームに戻るには、クライアントに再度ダウンロードすることが必要ですが、この場合ネットワーク・リソースが必要で、起動時間が遅延します。初期フォームを再度コールしない限り（ログイン・フォームなど）、OPEN_FORM を使用して、次のフォームをアプリケーションで表示します。

12

ロード・バランスに関する考慮事項

12.1 概要

この章では、Forms Server のロード・バランスに関する考慮事項について説明します。ロード・バランスにより、中間層マシンのプール（サーバー・ファーム）をメンテナンスし、これらのマシン間におけるサーバー・トラフィックの負荷のバランスをとります。ロード・バランスは、CGI のサポートにより任意の Web サーバーで実行できる CGI 実行可能ファイルを使用してインプリメントされます。

この章は、次のトピックに関する情報が含まれています。

- ロード・バランスに関する用語
- ロード・バランス・アクション
- Forms CGI-bin ベースのロード・バランスの設定
- Load Balancer Server トレース・ログの設定

12.2 ロード・バランスに関する用語

ロード・バランスの設定に必要な用語を示します。

- **Forms CGI**: CGI は Common Gateway Interface（共通ゲートウェイ・インタフェース）。Forms CGI は、ロード・バランスに使用するプログラムです。Forms CGI は、CGI をサポートするどの汎用リスナーでも使用できます。
- **Load Balancer クライアント**: マシンで現在実行している Forms プロセス数などの負荷情報を Load Balancer Server に送信するコンポーネント。Load Balancer Client は、Forms Server を使用して各マシン上で実行されます。
- **Load Balancer Server**: さまざまなロード・バランス・プールのすべての Forms Server を追跡するコンポーネント。指定したプールでサーバーのステータスを追跡して、その負荷を示す統計を保持します。指定したプールで要求を満たすことができる最小負荷のサーバーに、各フォームの実行リクエストを送ります。
- **プライマリ・ノード**: フォームを実行するすべての URL リクエストを扱う Web リスナー（および関連ソフトウェア）。ロード・バランスが使用中の場合、各フォームの実行リクエストは、Forms Server を実行している最小負荷マシンに経路指定されます。Load Balancer Server から最小負荷マシンの名前を取得します。
- **セカンダリ・ノード**: Forms Server、Runtime Client および Load Balancer Client を実行しているマシン。フォームの実行リクエストは、ロード・バランスが使用されているときにプライマリ・ノードからセカンダリ・ノードに送信されます。

注意： 多くの場合、プライマリ・ノードはセカンダリ・ノードとしても動作します（たとえば、Forms Server をインストールして実行している場合）。

注意： Load Balancer Server および Load Balancer クライアントは、Windows NT ではサービスとして、UNIX 上ではデーモンとしてインプリメントされます。

12.3 ロード・バランス・アクション

次に示す図 12-1 およびステップでは、CGI-bin 実行可能ファイルでロード・バランスを使用する場合に発生するイベントについて説明します。

1. Load Balancer クライアントは、負荷情報を Load Balancer Server へ定期的に送信します。この負荷情報には、各 Load Balancer クライアントで実行しているプロセスの総数が含まれます。
2. ユーザーは、Forms CGI-bin 実行可能ファイルを指す URL にアクセスします。
3. Forms CGI-bin 実行可能ファイルでは、使用できる最小負荷システムの名前について Load Balancer Server に尋ねます。
4. Forms CGI-bin 実行可能ファイルでは、Forms Server を実行しているシステムとして指定した最小負荷システムの名前で HTML ページを動的に作成し、その HTML ページをユーザーの Web ブラウザへ戻します。
5. ユーザーの Web ブラウザでは、Java アプレットを HTML ページで指定したホストからダウンロードすることを要求します。
6. Java アプレットは、特定の Form Builder アプリケーション（.FMX）を求めるリクエストを Forms Server に送信します。
7. サーバーは、Forms Server Runtime Engine と交信します。（サーバーは、アプリケーションの起動遅延を最小化するために、使用できるランタイム・エンジンのプールをメンテナンスします。）各アクティブ・ユーザーは専用のランタイム・エンジンを受信します。
8. サーバーは、ランタイム・エンジンとダイレクト・ソケット、HTTP または HTTPS との接続を確立し、ソケット、HTTP または HTTPS 情報を Java アプレットに送信します。次に Java アプレットで、ランタイム・エンジンとダイレクト・ソケット、HTTP または HTTPS との接続を確立します。Java アプレットとランタイム・エンジンは直接通信し、サーバーを解放して他のユーザーからの起動リクエストを受けられるようになります。（この時点では、アプリケーション・サーバーと Forms Server は、アプレットとランタイム・エンジン間の通信に関連していません。）Java アプレットはアプリケーションのユーザー・インタフェースをユーザーの Web ブラウザのメイン・ウィンドウに表示します。
9. ランタイム・エンジンは、データ・ソースにより、Net8 または ODBC (Open Database Connectivity) を介してデータベースと直接通信します。
10. Load Balancer クライアントは、負荷情報を Load Balancer Server へ送信し続けます。すべての新しいサービス・リクエストは、その情報に基づいて経路指定されます。

注意： Load Balancer Server が使用できない場合、ステップ 3 で、Forms CGI-bin 実行可能ファイルは最小負荷システムに関する情報を取得できません。そのかわり、Forms CGI-bin は、ユーザーのブラウザを MetricsServerErrorURL パラメータで指定した URL に再送信します。再送信はユーザーから見えないため、

ユーザーは再送信が行われていることを知る必要はありません。

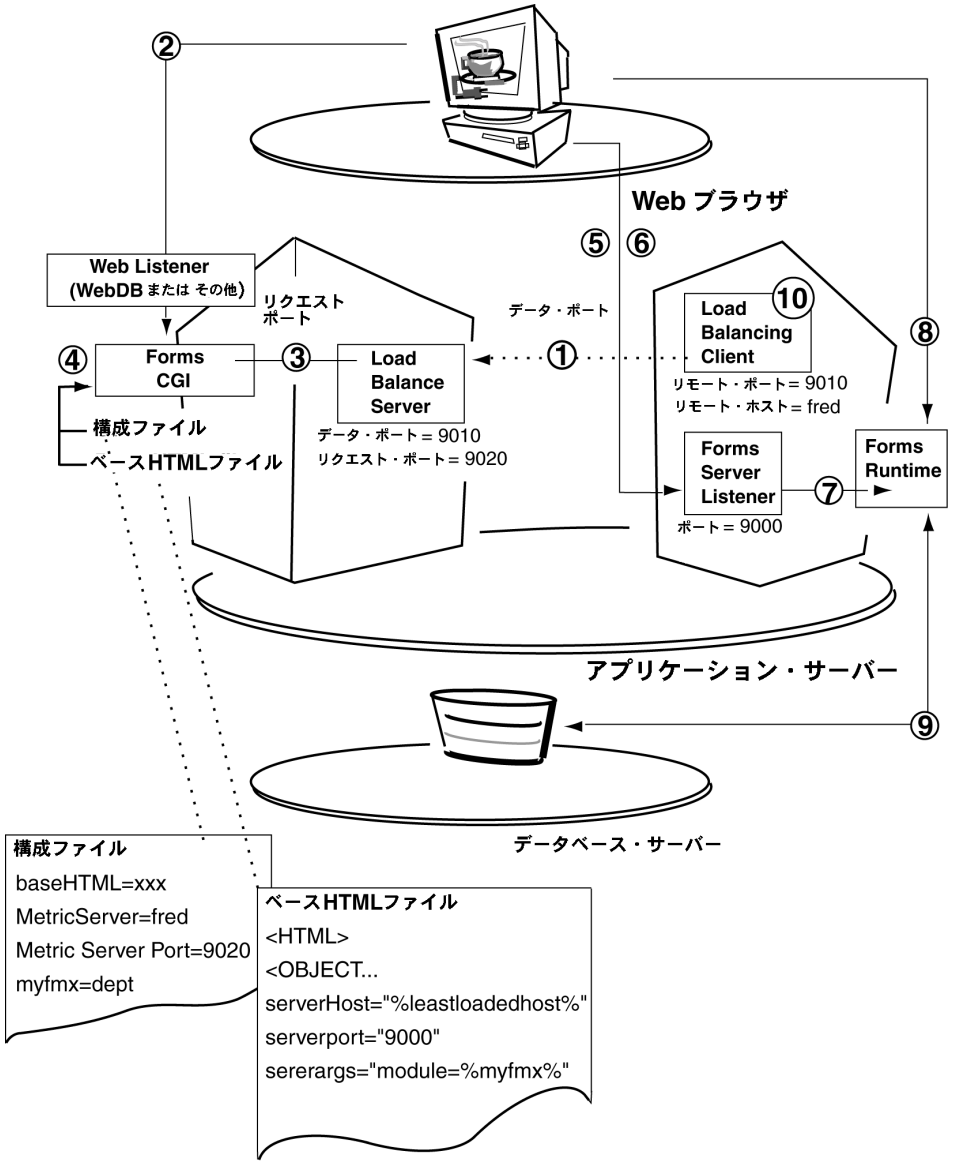

図 12-1 CGI-bin ベースのロード・バランス

12.4 Forms CGI-bin ベースのロード・バランスの設定

Forms Server リリース 6*i* が提供する CGI-bin 実行可能ファイルを使用してロード・バランスをインプリメントできます。CGI-bin ロード・バランスにより、導入済み Forms Server で任意の汎用 Web サーバーを使用できます。

次に示す項と第 5 章「Forms Server の構成」の情報を使用して、ロード・バランスを設定します。

- Oracle Installer を使用した CGI-Bin ロード・バランスの構成
- ロード・バランス構成のための Oracle Installer ダイアログ・ボックス
- Oracle Installer によって生成された構成ファイル
- Load Balancer Server と Load Balancer Client の起動

12.4.1 Oracle Installer を使用した CGI-Bin ロード・バランスの構成

Oracle Installer の「実行環境のインストール」オプションを使用して、CGI-bin ベースのロード・バランスをインストールおよび構成することをお薦めします。「実行環境のインストール」オプションは、次のことを実行します。

- ロード・バランスの設定方法についてユーザーに尋ねる。
- CGI-bin ベースのロード・バランスを使用するのに必要なコンポーネントをインストールする。
- これらのコンポーネントを自動的に構成する。

必要に応じて、Oracle Installer を使用するときはいつでも構成を更新できます。

注意 : ロード・バランスされる各マシンでロード・バランス・コンポーネントをインストールおよび構成する必要があります。

ロード・バランスを構成するには、次の手順に従います。

1. **「スタート」→「ファイル名を指定して実行」**を選択すると「ファイル名を指定して実行」ダイアログが表示されます。
2. 「ファイル名を指定して実行」ダイアログ・ボックスに次のように入力します（D: は実際の CD-ROM ドライブの文字に置き換えます）。

   ```
   D:¥SETUP.EXE
   ```

3. **「OK」**をクリックして、Oracle Installer を起動します。
4. **Oracle Forms Server** をクリックします。
5. 「**標準**」をクリックします。

6. 「**Web 配置用 Forms Server**」をクリックします。このオプションでは、Oracle Forms アプリケーションを Web アプリケーションとして実行するために必要なコンポーネントをインストールおよび構成します。
7. 「**複数マシン構成の一部**」をクリックします。（ロード・バランスは単一のマシン構成には適用されません。）
8. 最初のマシンのインストールで「**プライマリ・ノード**」をクリックします。以降のすべてのインストールについては、「**セカンダリ・ノード**」をクリックします。
   - **プライマリ・ノード :** このオプションを選択すると、マシンに Load Balancer Server がインストールされます。Load Balancer Server を使用して、プライマリ・ノードに Developer Runtime をインストールするか、Developer Runtime をインストールしなくてもプライマリ・ノードを実行できます。12.4.1.2 項「Developer Runtime のあるプライマリ・ノードのインストール」および 12.4.1.1 項「Developer Runtime のないプライマリ・ノードのインストール」を参照してください。
   - **セカンダリ・ノード :** このオプションを選択すると、Developer Runtime と共に Load Balancer Client がインストールされます。12.4.1.3 項「セカンダリ・ノードのインストール」参照。

複数のマシンの構成には次の 3 つのタイプがあります。

- Developer Runtime のあるプライマリ・ノードのインストール。これには、Developer Runtime を備えた Load Balancer Server が含まれます。（Forms リクエストも処理するようにロード・バランスをコントロールするマシンが必要な場合、このオプションを選択します。）
- Developer Runtime のないプライマリ・ノードのインストール。これには、Load Balancer Server のみ含まれます。（Forms リクエストを処理しないようにロード・バランスをコントロールするマシンが必要な場合、このオプションを選択します。）
- セカンダリ・ノードのインストール。これには、Developer Runtime のある Load Balancer Client が含まれます。（このマシンは常に Forms の要求を処理します。Forms の要求の分配はプライマリ・ノードの Load Balancer Server によって制御されます。）

12.4.1.1 Developer Runtime のないプライマリ・ノードのインストール

Oracle Installer は次のコンポーネントをインストールします。

- WebDB リスナー（要求された場合およびインストールされていない場合）
- Load Balancer Server
- JInitiator (フォームの実行に必要な場合、ユーザーがダウンロード)
- Forms Web CGI (Load Balancer Server API、Forms Applet JAR ファイルおよび Forms Java Class Support ファイルを含む)
- Oracle Net8 Client

12.4.2.1 項「Load Balancer Server パラメータ」で説明する情報を提供する必要があります。

インストールが完了したら、dev6iconfig.txt テキストに書き込まれた追加の構成ステップを、12.4.3.1 項「dev6iconfig.txt」の説明にしたがって完了します。

12.4.1.2 Developer Runtime のあるプライマリ・ノードのインストール

このオプションでは、12.4.1.1 項「Developer Runtime のないプライマリ・ノードのインストール」で説明されているすべてのものおよび Forms Server (Forms Applet Jar Files と Forms Java Class Support Files を含む) をインストールします。

12.4.2.1 項「Load Balancer Server パラメータ」および 12.4.2.3 項「Forms Server パラメータ」で説明する情報を提供する必要があります。

インストールが完了したら、dev6iconfig.txt テキストに書き込まれた追加の構成ステップを、12.4.3.1 項「dev6iconfig.txt」の説明にしたがって完了します。

12.4.1.3 セカンダリ・ノードのインストール

Oracle Installer は次のコンポーネントをインストールします。

- Forms Server (Forms アプレット JAR ファイルおよび Forms Java Class Support ファイルを含む)
- Load Balancer クライアント
- Oracle Net8 Client

12.4.2.2 項「Load Balancer クライアントパラメータ」および 12.4.2.3 項「Forms Server パラメータ」で説明する情報を提供する必要があります。

インストールが完了したら、dev6iconfig.txt テキストに書き込まれた追加の構成ステップを、12.4.3.1 項「dev6iconfig.txt」の説明にしたがって完了します。

12.4.2 ロード・バランス構成のための Oracle Installer ダイアログ・ボックス

複数マシンの構成中、Oracle Installer は次のロード・バランス構成情報についてプロンプトを表示します。

- Load Balancer Server パラメータ
- Load Balancer クライアントパラメータ
- Forms Server パラメータ

構成が完了したら、12.4.2.4 項「構成の最終チェック」の説明にしたがって、最終チェックを実行します。

12.4.2.1 Load Balancer Server パラメータ

このダイアログ・ボックスは、Forms Load Balancer Server で使用されるポート番号についてプロンプトを表示します。

- **データ・ポート**: デフォルトは 9010。Load Balancer クライアントからの負荷データをリスニングする TCP/IP ポート番号を入力します（セカンダリ・ノードで実行されます）。この値は、12.4.4 項「Load Balancer Server と Load Balancer Client の起動」で説明されているように、Load Balancer Server を手動で開始する場合に変更できるパラメータです。

注意： Load Balancer Server のデータ・ポート値は、すべての Load Balancer クライアントのデータ・ポート値に合致する必要があります。

- **リクエスト・ポート**: デフォルトは、9020。Forms Web CGI で作成された " 最小負荷ホスト " のリクエストをリスニングする TCP/IP ポート番号を入力します。この値は、MetricServerPort パラメータとして formsweb.cfg ファイルに書き込まれます。12.4.3.2 項「formsweb.cfg」および 5.4.1.1 項「formsweb.cfg ファイル内のパラメータ」を参照してください。

デフォルト値を受け入れるか、Load Balancer Server の起動パラメータ値を変更できます。この値は、ポートが別のプログラムによってすでに使用されている場合のみ変更します。

12.4.2.1.1 formsweb.cfg ファイルに対する更新： Oracle Installer で、Load Balancer Server をインストールするときは常に、インストールの終わりに formsweb.cfg ファイルの更新が行われます（これは、Forms CGI 実行可能ファイルにすべての必須情報が含まれているか確認するために行われます）。更新内容は次のとおりです。

- MetricsServerPort パラメータは、ユーザーによって入力された Load Balancer Server の " リクエスト・ポート " 起動パラメータに合致するよう設定 (MetricsServerPort=1234 など)。
- serverHost パラメータは、値 %LeastLoadedHost% に設定（serverHost=%LeastLoadedHost%）。

ドメイン名が名前の解決のためにネットワークで必須の場合、ドメイン名を serverHost パラメータに追加する必要があります。たとえば、serverHost=%LeastLoadedHost%.jp.oracle.com となります。

formsweb.cfg ファイルの詳細は、12.4.3.2 項「formsweb.cfg」を参照してください。また、5.4.1.1 項「formsweb.cfg ファイル内のパラメータ」も参照してください。

12.4.2.2 Load Balancer クライアントパラメータ

「Load Balancer Client パラメータ」ダイアログ・ボックスは、Forms Load Balancer クライアントで使用されるホスト名とポート番号を表示します。

- **データ・ホスト**: デフォルト値は、*localhost*。プライマリ・ノード（Load Balancer Server が実行されているマシン）の完全なホスト名を入力します。値は 256 文字以内で入力します。

- **データ・ポート**: デフォルトは 9010。ロード・バランス・サーバーが負荷データをリスニングする TCP/IP ポート番号を入力します。この値は、12.4.4 項「Load Balancer Server と Load Balancer Client の起動」で説明されているように、Load Balancer クライアントを手動で開始する場合に変更できる起動パラメータです。

注意 : 各 Load Balancer クライアントのデータ・ポート値は、Load Balancer Server のデータ・ポート値に合致する必要があります。

デフォルト値を受け入れるか、Load Balancer クライアントの起動パラメータ値を変更できます。このポート番号は、別のプログラムによって使用されている場合のみ変更します。

12.4.2.3 Forms Server パラメータ

このダイアログ・ボックスは、Forms Server で使用されるポートおよびプロトコルを表示します。

- **Forms Server ポート**: デフォルトは 9000。Forms Server がフォームの実行リクエストをリスニングする TCP/IP ポート番号を入力します。この値は、serverPort パラメータとして、base.htm および basejinit.htm ファイルに書き込まれます。

注意 : ロード・バランス化されるすべての Forms Server には、同じ Forms Server ポート値が必要です。

- **プロトコル**デフォルトは、ソケット。これは、Forms Runtime エンジンと Forms Java アプレット間の通信に使用するプロトコルです。この値は、ファイアウォールを介して通信を行う場合にのみ、HTTP または HTTPS に変更する必要があります。たとえば、このマシンがファイアウォールの内側にあり、Forms アプリケーションをファイアウォールの外側のユーザーが利用する必要がある場合、HTTP を選択します。HTTP を、SSL（secure sockets layer）とともに使用する場合は、HTTPS を選択します。

  この値は、12.4.4 項「Load Balancer Server と Load Balancer Client の起動」で説明されているように、Forms Server を手動で開始する場合に変更できる起動パラメータです。デフォルトのパラメータ値を受け入れるか、Forms Server の起動パラメータ値を変更できます。このポート番号は、別のプログラムによって使用されている場合のみ変更します。

注意 : インストール・プロセス中に入力したパラメータ値は、次の Windows レジストリの場所へ保存されます。

HKEY_LOCAL_MACHINE → SYSTEM → CurrentControlSet → Services → OracleFormsServer<*ServiceName*>

12.4.2.3.1 formsweb.cfg ファイルに対する更新 : Oracle Installer で、Forms Server をインストールするときは常に、インストールの終わりに formsweb.cfg ファイルの更新が行われます（これは、Forms CGI 実行可能ファイルにすべての必須情報が含まれているか確認するために行われます）。更新内容は次のとおりです。

- serverPort パラメータを、ユーザーが入力した Forms Server のポート起動パラメータと合致するよう設定。

formsweb.cfg ファイルの詳細は、12.4.3.2 項「formsweb.cfg」を参照してください。また、5.4.1.1 項「formsweb.cfg ファイル内のパラメータ」も参照してください。

12.4.2.4 構成の最終チェック

次の点を確認します。

- Load Balancer Server のデータ・ポート値が、すべての Load Balancer クライアントのデータ・ポート値と合致しているか。
- ロード・バランス化されるすべての Forms Server には、同じ Forms Server ポート値があるか。

12.4.3 Oracle Installer によって生成された構成ファイル

12.4.2 項「ロード・バランス構成のための Oracle Installer ダイアログ・ボックス」で説明されているように、Oracle Installer が構成パラメータを表示した後に、Installer は次のファイルを自動で生成します。これらのファイルは、¥<ORACLE_HOME>¥orainst and ¥<FORMS60>¥server ディレクトリに配置されます。

- dev6iconfig.txt
- formsweb.cfg
- base.HTM および basejini.HTM

これらのファイルの詳細は、次の項で説明します。

12.4.3.1 dev6iconfig.txt

このテキスト・ファイルには、システムの構成を完了するために、Oracle Installer が実行する手順およびユーザーが実行する必要がある手順の状態が記載されています。このファイルは、¥<ORACLE_HOME>¥orainst ディレクトリに置かれます。

ファイルを開いて、そのファイルに記載されている指示に従います。[*情報*] とマークされている項目は、Oracle Installer がユーザー用に構成した事項です。[アクション] とマークされている項目は、構成を完了するために必要な事項です。たとえば、WebDB リスナー以外のリスナーを使用することにした場合、[アクション] 項目を受信し、仮想パスを指定してリスナー構成を完了します。

12.4.3.2 formsweb.cfg

このファイルは、<FORMS60>¥server ディレクトリに置かれます。このファイルの例は、5.4.1 項「formsweb.cfg」を参照してください。また、5.4.1.1 項「formsweb.cfg ファイル内のパラメータ」も参照してください。このファイルには、12.4.2 項「ロード・バランス構成のための Oracle Installer ダイアログ・ボックス」で構成した多くのロード・バランスパラメータの設定が記載されています。

dev6iconfig.txt ファイルで変更するように指示されない限り、formsweb.cfg ファイルを変更する必要はありません。

12.4.3.3 base.HTM および basejini.HTM

base.htm および basejini.htm ファイルは、インストールおよび構成が完了した後、<FORMS60>¥server ディレクトリに書き込まれます。これらのファイルの例は、5.4.2 項「base.htm および basejini.htm」を参照してください。これらのファイルには、FMX ファイルを実行するために必要な情報が記載されています。

注意： Oracle Installer は、<FORMS60>¥server ディレクトリ内の同じ名前の既存ファイルを上書きします。

注意： base.htm および basejini.htm ファイルは変更しないことをお薦めします。かわりに、そのファイルをコピーして改名し、formsweb.cfg ファイル内の baseHTML パラメータの値を、作成したファイル名に変更します。

12.4.4 Load Balancer Server と Load Balancer Client の起動

Load Balancer Server および Load Balancer クライアントを手動で起動（および構成パラメータを変更）できます。

注意： 次のプロシージャには管理者権限が必要です。また、構成変更を有効にするためにはそのプロセスを停止して再起動する必要があります。

12.4.4.1 Load Balancer Server の起動

Windows NT では、Load Balancer Server は通常サービスとして実行されます。次のように、異なる起動パラメータを使って、サービスを手動で再作成および再起動できます。

```
d2ls60 -remove Oracle_Load_Balancer_60_Server

d2ls60 -install -autostart Oracle_Load_Balancer_60_Server <dataPort>   <requestPort>
[<maxNumClients>] [<traceLevel>]
```

注意： 名前 Oracle_Load_Balancer_60_Server は、インストール・プロセス中に作成されたサービスの名前ですが、任意のサービス名を使用できます。

- <dataPort> および <requestPort> は、12.4.2.1 項「Load Balancer Server パラメータ」で説明しています。
- <maxNumClients> は、デフォルトが 1000 に設定されており、負荷情報を実行して Load Balancer Server に送信している Load Balancer クライアントの最大数を指定します。
- <traceLevel> は、デフォルトが 0 に設定され、トレースを実行しないことを意味します。10 を指定すると、Load Balancer Server のトレース出力を作成できます。詳細は、12.5 項「Load Balancer Server トレース・ログの設定」を参照してください。

12.4.4.2 Load Balancer クライアントの起動

Windows NT では、Load Balancer Client は通常サービスとして実行されます。次のように、異なる起動パラメータを使って、サービスを手動で再作成および再起動できます。

```
d2lc60 -remove Oracle_Load_Balancer_60_Client

d2lc60 -install -autostart Oracle_Load_Balancer_60_Client <MetricServerhost Name>
<RemotePort> 0 [<ScaleFactor> <ProcessName>]
```

注意： 名前 Oracle_Load_Balancer_60_Client は、インストール・プロセス中に作成されたサービスの名前ですが、任意のサービス名を使用することができます。

- <MetricServerhost Name> は、12.4.2.2 項「Load Balancer クライアントパラメータ」で説明しています。
- <RemotePort> は、12.4.2.2 項「Load Balancer クライアントパラメータ」で説明しています。
- < LocalPort> は、クライアントが Load Balancer Server に情報を問い合せるために使用する Load Balancer クライアント上のポートです。これは、常に 0 に設定してください。
- <Scale factor> は、Windows NT の場合のデフォルトは 4 で、UNIX の場合は 1 に設定されます。これにより、Load Balancer クライアントの容量が異なるために起きる不均衡を軽減することができます。Forms Server のロード・バランスは各 Load Balancer クライアント上で実行されているプロセスの総数にのみ基づいているので、最小負荷システムと思われるシステムは、新規プロセスを実行するのに必ずしも最適な場所ではありません。容量が少ないシステムには、さらに大きい ScaleFactor 値を割り当てる必要があります。
- <ProcessName> は、「実行環境のインストール」オプションを使用して製品をインストールした場合は、デフォルトは "ifweb60"（Runform プロセス）に設定されます。この値を設定すると、Load Balancer Client に対して、実行可能ファイル名が指定した名前と合致する（ロード・バランスが目的の）プロセス数をカウントするように指示されます。値を指定しない場合、マシン上のすべてのプロセスがカウントされます。"ifweb60" を指定することをお薦めします。

12.5 Load Balancer Server トレース・ログの設定

この項では、Load Balancer Server トレース・メッセージの形式について説明します。

トレースを開始するには、Load Balancer Server を再起動してトレース・オプションを指定します。詳細は 12.4.4 項「Load Balancer Server と Load Balancer Client の起動」を参照してください。

12.5.1 トレース・レベル 1

トレース・レベル 1 には次のヘッダーが含まれます。

```
HOSTNAME:          neko.us.oracle.com    IP ADDRESS: 144.25.83.146
Data port number:  1234  Request port number:  1235
Maximum number of clients: 10    Trace level: 2
```

- **Hostname** および **IP Address**: D2LS サーバーのホスト名とアドレス。
- **Data port number**: D2LS サーバーが D2LC クライアント・メッセージをリスニングするポート番号。このポートは D2LC クライアント・プロセスを構成するために使用されます。
- **Request port number**: サーバーが最小負荷のホスト情報リクエストをリスニングするポート番号。
- **Maximum number of clients**: D2LC クライアントに割り当てられたスロット数。クライアントごとに 1 つのスロットが必要です。
- **Trace level**: サーバー・ログ・ファイルに印刷されたトレース情報量。

12.5.2 トレース・レベル 2

トレース・レベル 2 では、メッセージは D2LC クライアントから次の形式で提供されます。各フィールドの記述は次のとおりです。

```
D:000    144.25.83.92:1236  922541864 1     2  45 3 [cogito]
|  |          |         |        |     |     |  |  | |
^  |          |         |        |     |     |  |  | |  Packet type recv'd
   ^^^       |         |        |     |     |  |  | |  Client index
         ^^^^^^^^^^^^  |        |     |     |  |  | |  D2LC IP Address
                     ^^^^      |     |     |  |  | |  D2LC Port number
                           ^^^^^^^^^ |     |  |  | |  Time msg recv'd
                                     ^     |  |  | |  Scale factor
                                           ^  |  | |  Sequence number
                                              ^^ | |  Number of processes
                                                 ^ |  Last selected
                                                   ^  D2LC Hostname
```

トレース・ログの各フィールドの記述は次のとおりです。

- **Packet Type Received**: 受信したパケットのタイプ。次のタイプを指定できます。
  - **D**: D2LC クライアントから受信されたデータ。タイプ "D" のパケットの場合、トレース行の残りのデータはクライアントから送信された情報と一致します。
  - **S**: 最小負荷ホスト用に選択されたクライアント。タイプ "S" のパケットの場合、トレース行の残りのデータは最小負荷ホストとして選択および戻されたクライアントと一致します。
- **Client index**: D2LC クライアントの内部索引。この索引は、リスエストがクライアントから最初に受信されるときに割り当てられます。0 から始まります。
- **D2LC IP Address**: メッセージを送信するクライアントの IP アドレス。
- **D2LC Port number**: メッセージを送信するクライアントが使用する IP ポート番号。
- **Time message received**: メッセージをクライアントから受信した時刻。1970 年 1 月 1 日の 00:00:00 UTC 以降の秒単位の時刻です。
- **Scale factor**: クライアントに割り当てられたスケール・ファクタ。スケール・ファクタは、最小負荷ホストを選択する場合のプロセス数に対する乗数として使用されます。
- **Sequence number**: クライアントが D2LS サーバーにメッセージの送信を試みた回数。
- **Number of processes**: クライアントによって報告されたプロセス数。
- **Last selected**: クライアントが負荷が最も少ないホストとして最後に選択された時刻。サーバー内の内部カウンタは時間経過とともに増加します。クライアントが負荷が最も少ないホストとして選択されると、このカウンタは「**Last Selected**」フィールドに格納されます。最小の「**Last Selected**」フィールドをもつ D2LC クライアントは、最近の使用頻度が最も低いことになります。負荷が最も少ないホストをリクエストするとプロセ

ス数が最も少なくなり、最近の使用頻度が最も低いクライアントが選択されてその結合をブレークします。

- **D2LC Hostname**: D2LC クライアントのホスト名。

12.5.3 トレース・ファイルのサンプル

次に、2 つのサーバー構成のサンプルのトレース・ファイルを示します。Formsvr1 は D2L クライアントと D2L サーバーを実行します。Formsvr2 は D2L クライアントを実行します。

```
HOSTNAME:          formsvr1.us.oracle.com       IP ADDRESS:   144.25.87.101
Data port number:  1234 Request port number:  1235
Maximum number of clients: 10   Trace level: 2

D:000   144.25.87.101:1000  925260387 1     2    0 0 [formsvr1]
D:000   144.25.87.101:1000  925260387 1     3   43 0 [formsvr1]
D:001   144.25.87.102:1001  925260388 1     2    0 0 [formsvr2]
D:001   144.25.87.102:1001  925260388 1     3   43 0 [formsvr2]
S:000   144.25.87.101:1000  925260387 1     3   44 1 [formsvr1]
D:000   144.25.87.101:1000  925260387 1     4   45 1 [formsvr1]
D:001   144.25.87.102:1001  925260388 1     4   45 0 [formsvr2]
S:001   144.25.87.102:1001  925260388 1     4   46 2 [formsvr2]
D:000   144.25.87.101:1000  925260387 1     5   45 1 [formsvr1]
D:001   144.25.87.102:1001  925260388 1     5   45 2 [formsvr2]
S:000   144.25.87.101:1000  925260387 1     5   46 3 [formsvr1]
D:000   144.25.87.101:1000  925260387 1     6   47 3 [formsvr1]
D:001   144.25.87.102:1001  925260388 1     6   47 2 [formsvr2]
S:001   144.25.87.102:1001  925260388 1     6   48 4 [formsvr2]
D:000   144.25.87.101:1000  925260387 1     7   47 3 [formsvr1]
D:001   144.25.87.102:1001  925260388 1     7   47 4 [formsvr2]
```

13

Oracle Enterprise Manager Forms のサポート

13.1 概要

この章では、Forms とともに使用する Oracle Enterprise Manager（OEM）のインストールと構成の方法について説明します。OEM の特徴と機能についても説明します。OEM は、グラフィカルな Java コンソール、管理サーバー、エージェント、および Oracle 製品を管理するための統合されたシステム管理プラットフォームを提供するツールから構成されるシステム管理ツールです。

この章には、次の項が含まれています。

- OEM を使用する理由
- OEM コンポーネント
- Forms とともに使用する OEM コンポーネントのインストールと構成
- OEM コンソールからの Forms Server の管理
- OEM メニュー・オプション

詳細な OEM ドキュメントは、次のマニュアルに記載されています。

- 『Oracle Enterprise Manager 概説』
- 『Oracle Enterprise Manager 管理者ガイド』
- 『Oracle Enterprise Manager 構成ガイド』

13.2 OEM を使用する理由

OEM Forms 管理者インタフェースには、次の基本機能があります。

- **ノードおよびサービスの自動検出** Forms Listener、Forms Server、Load Balancer Server および Load Balancer Client は、管理対象のノード上で OEM の Intelligent Agent によって自動的に検出され、OEM コンソールのナビゲータ・ツリーに表示されます。
- **ノードおよびサービスの制御** 検出されたノードとサービスに対して、起動や停止などのいくつかの基本制御が提供されます。
- **ノードおよびサービスの監視** 検出された Forms Listeners、Forms Servers、Load Balancer Servers、Load Balancer Clients の、次のイベントを監視します。サービス停止、過度のメモリ使用、過度の CPU 使用。これらのイベントのいずれかが発生すると、事前にプログラムされたアクションが実行され、システム管理者に警告されるか、問題の自動修正が試行されます。

13.3 OEM コンポーネント

Forms Server を管理するためには、次の 3 つの OEM コンポーネントをインストールする必要があります。

- **OEM Management Server（OMS）**OMS は、OEM の中央リポジトリを制御し、中央リポジトリとして機能するソフトウェアです。OMS は、1 台のマシンにのみインストールします。この OMS マシンが他のマシンを管理します。
- **OEM コンソール** このソフトウェアは、OMS のユーザー・インタフェースを提供します。
- **OEM エージェント** このソフトウェアは、Forms サーバー・データを収集し、OMS に送信します。OEM エージェントは、OMS によって管理されるすべての Forms サーバー・マシンにインストールする必要があります。

13.4 Forms とともに使用する OEM コンポーネントのインストールと構成

OEM Management Server（OMS）、OEM コンソールおよび OEM エージェントソフトウェアは、Forms OEM CD-ROM に入っています。OEM エージェントソフトウェアは、Oracle Forms Developer Server 6*i* の CD-ROM にも入っています。

注意： OEM を使用する場合は、Forms Server コンポーネントを NT サービスとして設定しないでください。

13.4.1 NT での OMS のインストール

OMS を NT 上にインストールする手順は、次のとおりです。

1. Forms OEM CD-ROM 上の setup.bat ファイルをダブル・クリックします。
2. 表示されるダイアログ・ボックスで、2 番目のラジオ・ボタンを選択して OEM コンソールと OMS をインストールします。
3. ダイアログ・ボックスで、OMS をインストールする新規 ORACLE_HOME ディレクトリの名前を入力します。
4. 次のオプションを選択します。**「標準」インストール**
5. 次に、4 つの Configuration Assistant ダイアログ・ボックスの 1 つが表示されます。最初のダイアログ・ボックスでは、データベースへのアクセスに必要なユーザー名とパスワードを入力します。入力するユーザーには、表を作成できるなどのシステム権限が必要です。
6. Configuration Assistant の第 2 のダイアログ・ボックスでは、dev2000srv-pc と manager などのユーザー名とパスワードを入力することによって、リポジトリ・ユーザーを作成します。
7. Configuration Assistant の第 3 のダイアログ・ボックスでは、デフォルトを使用します。
8. Configuration Assistant の第 4 のダイアログ・ボックスでは、「**完了**」を選択してリポジトリを作成します。

13.4.2 NT 上の OEM での Forms サポートの構成

Forms と OMS をインストールした後で、次の操作を実行します。次のステップを実行している間は、OMS サービスを実行しないでください。

1. ディレクトリを $ORACLE_HOME¥sysman¥admin に変更します。
2. 次のように入力します。

   ```
   regvif  <RepositoryUser>/<RepositoryUserPassword>@<OMS Repository>
   ```

   このコマンドによって、Forms レジストリが OMS リポジトリにインストールされます。たとえば、次のように入力します。

   ```
   regvif dev2000srv-pc/manager@dev2000srv-pc
   ```

3. システム権限を持つログインを使用して、データベースに接続します。
4. "createOEMFormsUser.sql" スクリプトを実行して、OEM リポジトリ内の Forms 固有データをサポートする OEM Forms ユーザーを作成します。(このスクリプトを変更して、デフォルトの表領域や割当て制限などを追加できます。ただし、ユーザー名とパスワードをスクリプトで変更することはできません。)
5. OEM Forms ユーザーとしてデータベースに接続します。(そのユーザー名とパスワードで実行した SQL スクリプトを参照してください。)

6. "createOEMFormsTables.sql" スクリプトを実行して、OEM リポジトリ内に必要な表を作成します。
7. net80/admin の下の OMS Oracle ホームに、TNS エントリを作成します。OEM コンソールを実行しようとしているマシン上の tnsnames.ora ファイルに、OEM リポジトリのデータベース名が存在している必要があります。

13.4.3 OMS サービスの開始と OEM コンソールへの接続

OMS サービスの開始と OEM コンソールへの接続の手順は、次のとおりです。

1. NT サービス・コントロール・パネルから、OEM Management サービスを開始します。
2. Console に接続する手順は、次のとおりです。
   a. 「**スタート**」→「**プログラム**」→「**ORACLE_HOME**」→「**OEM**」→「**OEM コンソール**」を選択します。
   b. デフォルトのユーザー名 sysman とパスワード oem_temp を使用します。新しいパスワードを作成するようにプロンプトが表示されます。

13.4.4 Forms Server マシンへの OEM エージェントのインストール

OEM エージェント 8.0.6 は、Forms Developer 6*i* インストレーションの一環として、ORACLE_HOME ディレクトリにインストールされます。エージェント 8.0.6 がインストールされていることを確認するには、NT サービス・コントロール・パネルに Oracle エージェント 60 エントリがあり、ORACLE_HOME ディレクトリに agentbin ディレクトリが存在している必要があります。または、エージェント 8.1.5 を Forms OEM CD からインストールできます。

OEM エージェントをインストールする手順は、次のとおりです。

1. Forms OEM CD-ROM 上の setup.bat ファイルをダブル・クリックします。
2. 表示されるダイアログ・ボックスで、3 番目のラジオ・ボタンを選択して OEM エージェントをインストールします。
3. ダイアログ・ボックスで、エージェント・ソフトウェアをインストールする ORACLE_HOME ディレクトリの名前を入力します。
   - エージェントバージョン 8.1.5 をインストールする場合は、エージェントを OEM の ORACLE_HOME ディレクトリにインストールできます。
   - エージェントバージョン 8.0.6 をインストールする場合は、エージェントを Forms Server の ORACLE_HOME ディレクトリにインストールできます。
4. 次のオプションを選択します。
   - **NetServiceName** これは、OMS がインストールされるマシン名です。たとえば、oradevdemo2-pc などです。

- **プロトコル** TCP/IP (インターネット・プロトコル)
- **プロトコル設定**データベースが置かれている OMS マシンのホスト名。たとえば、oradevdemo2-pc.us.oracle.com などです。
- **ポート番号**データベースが置かれているマシンのポート番号。
- **サービス** Oracle 8i リリース 8.0 またはそれ以前を選択し、サービス名または SID を入力します。

5. 「**テスト**」を選択して、OMS マシン上のデータベースへの接続をチェックします。
6. 「**完了**」を選択します。
7. NT サービス・コントロール・パネルから、OEM エージェントを開始します。

13.5 OEM コンソールからの Forms Server の管理

既存の Forms Listener は、OEM から管理できません。最初に、OEM コンソールから Forms Listener を作成する必要があります。

13.5.1 ノードの検索

OEM でリモートの Forms Server マシンを管理するには、そのマシンの場所を探す必要があります。場所を探す手順は、次のとおりです。

1. OEM コンソールで、メニューから「**ノードの検出**」を選択します。
2. ノード名を入力します。たとえば、dev2000srv-pc と入力します。

13.5.2 リモート Forms Server のオペレーティング・システム（NT）での管理ユーザーの作成

OMS は、リモート・マシンのオペレーティング・システム上の管理ユーザーを通して、リモートの Forms Server マシンを管理します。この管理ユーザーには、" バッチ・ジョブとしてログオン " ユーザー権利が必要です。管理ユーザーを作成する手順は、次のとおりです。

1. NT に管理ユーザーとしてログオンします。
2. NT で、「**スタート**」→「**プログラム**」→「**管理ツール**」→「**ユーザー・マネージャ**」を選択します。
3. ユーザー名を選択します。
4. 「原則」メニューから、「**ユーザーの権利**」を選択します。
5. 「ユーザー権利の原則」ダイアログ・ボックスで、「**高度なユーザー権利の表示**」チェックボックスを選択します。

6. 「権利」ドロップダウン・リストから、「**バッチ・ジョブとしてログオン**」を選択します。
7. 「**OK**」を選択します。

13.5.3 OEM コンソールでの管理ユーザーの資格証明の入力

OEM コンソールで管理ユーザーの資格証明を入力する手順は、次のとおりです。

1. OEM コンソールを開始します。
2. 「システム」メニューから、「**作業環境**」を選択します。
3. 「**優先接続情報リスト**」タブを選択します。
4. 「サービス名」列で、管理するリモート Forms Server マシンの名前を検索します。サービス・タイプがノードの行を選択してください。
5. 13.5.2 項「リモート Forms Server のオペレーティング・システム（NT）での管理ユーザーの作成」で作成した管理ユーザーの名前とパスワードを入力します。

13.5.4 OEM コンソールからの Forms Runtime インスタンスの表示

OEM コンソールから Forms Runtime インスタンスを表示する手順は、次のとおりです。

1. OEM コンソールで、「**Developer Server**」、「**Forms_Listeners_**<*RemoteMachineName*>」を選択します。
2. マウスの右ボタンをクリックして、「**ランタイム・プロセスのリスト**」を選択します。

13.6 OEM メニュー・オプション

次のメニュー・オプションは、Forms Listener、Forms Server、Load Balancer Servere および Load Balancer Client の管理に使用できます。

13.6.1 Forms Listener グループの制御

マウスの右ボタンで表示されるメニューで使用可能なコマンドは、次のとおりです。

- **新規作成** 新規リスナー・プロセスを作成する前に、パラメータのリストを求めるプロンプトが表示されます。リスナー・プロセスが作成されると、エントリがナビゲータ・ツリーに表示され、リスナーが開始します。
- **ランタイム・プロセスのリスト** このコマンドは、このノードで実行されている Forms Runtime プロセスのリストを別のウィンドウに表示します。「ランタイム・プロセス・リスト」ウィンドウ参照。
- **リフレッシュ** このコマンドは、このノードで実行されている既存の Forms Listener を検出し、このノード上のすべての Forms Listener インスタンスの実行中 / 非実行中ステータスもリフレッシュします。

13.6.2 Forms Listener インスタンスの制御

マウスの右ボタンで表示されるメニューで使用可能なコマンドは、次のとおりです。

- **開始** リスナーが現在ダウンしている場合に、リスナーを開始します。
- **停止** リスナーがシャットダウンし、Listener インスタンスがダウンしていることが特殊なアイコンによって示されます。
- **類似作成** コピー・コマンドと非常によく似ていて、現在のリスナーと同じパラメータを使用して別のリスナーを作成します。「新規作成」コマンドと同様のダイアログによって、必要な変更を行うように求められます。
- **変更** ダイアログ・ボックスで、起動パラメータと環境変数を変更できます。
- **削除** Listener インスタンスがナビゲータ・ツリーから削除されます。Forms Listener インスタンスは、そのリスナーに関連付けられている Runtime プロセスがない場合のみ削除できます。削除されたリスナーは、ノードから自動的にシャットダウンします。
- **プロパティ** この Forms Listener インスタンスに関連付けられているパラメータ、環境変数および Runtime プロセスのリストを表示します。

13.6.3 「ランタイム・プロセス・リスト」ウィンドウ

このウィンドウは、ある特定のノード上の現在の Forms Runtime プロセスをすべて示す表タイプのリストです。各行が 1 つの Runtime プロセスを表します。次のフィールドが表示されます。

- リスナー名
- ノード
- IP アドレス
- ユーザー名
- プロセス ID
- 接続時間
- 動的ロギング・ステータス
- メモリー使用量
- CPU %

13.6.4 Forms Runtime プロセスの制御

マウスの右ボタンで表示されるメニューで使用可能なコマンドは、次のとおりです。

- **削除** 実行を停止するための削除シグナルが Runtime インスタンスに送信されます。このコマンドは主に、不正なランタイム・プロセスを停止して、それ以上損害が与えられないようにするために使用します。
- **ロギング使用可能** ランタイム・インスタンスの動的ロギングをオンにします。ログは、生成されたファイル名でテンポラリ・ファイルに書き込まれます。ファイル形式は、Forms Runtime Diagnostic（FRD）で生成される形式と同じです。
- **ロギング使用不可** ランタイム・インスタンスの動的ロギングをオフにします。
- **ログの表示** 動的ロギング・コマンドで生成されたログ・ファイルを表示します。

13.6.5 Load Balancer Server グループの制御

マウスの右ボタンで表示されるメニューで使用可能なコマンドは、次のとおりです。

- **新規作成** Load Balancer Server インスタンスが作成されます。サポートされるパラメータは次のとおりです。*<port #1> <port #2> <max. no. of client> <trace level>*。

Load Balancer Server は、Metrics Server とも呼ばれています。

13.6.6 Load Balance Server インスタンスの制御

マウスの右ボタンで表示されるメニューで使用可能なコマンドは、次のとおりです。

- **開始** Load Balancer Server 開始します。
- **停止** サーバーがシャットダウンします。
- **類似作成** コピー・コマンドと非常によく似ていて、現在と同じパラメータを使用して別の Load Balancer Server を作成します。
- **変更** 起動パラメータと環境変数を変更するためのダイアログ・ボックスが表示されます。
- **削除** Load Balancer Server がナビゲータ・ツリーから削除されます。削除されたサーバーは、ノードから自動的にシャットダウンします。
- **プロパティ** この Load Balancer Server に関する関連情報を表示する別のウィンドウが起動します。

Load Balancer Server は、Metrics Server とも呼ばれています。

13.6.7 Load Balancer Client グループの制御

コマンドは、Load Balancer Server のオブジェクト・タイプとまったく同じです。サポートされるパラメータは、次のとおりです。

<Master Server host name> <Remote port> <Local port> <Scale Factor>

Load Balancer Client は、Metrics Client とも呼ばれています。

13.6.8 Load Balancer Client インスタンスの制御

コマンドは、Load Balancer Server のオブジェクト・インスタンスとまったく同じです。

Load Balancer Client は、Metrics Client とも呼ばれています。

13.6.9 監視機能

イベントは、OEM コンソールのイベント管理画面にリストされます。これらのイベントは、OEM で登録または登録解除することによって、オンまたはオフにできます。イベントが作成されて OEM に登録されると、OEM はイベントが発生したときに、システム管理者に通知するか、回復処理を実行できます。

次のイベントを登録できます。

- **リスナー停止** このイベントは、リスナー回復処理ありまたはなしでスケジュールできます。リスナー回復処理は、このイベントが発生したときに Listener を再起動するために使用できます。Listener がダウンした場合は、イベント・ログにエントリが書き込まれます。このエントリは、OEM コンソールから表示できます。

注意： 回復処理は、回復処理を使用するイベントをスケジュールする前にスケジュールする必要があります。

- **Load Balancer Server 停止** リスナー停止と同様です。このイベントは、Load Balancer Server 回復処理ありまたはなしでスケジュールできます。
- **Load Balancer Client 停止** Load Balancer Server 停止と同様です。このイベントは、Load Balancer Client 回復処理ありまたはなしでスケジュールできます。
- **ランタイム・プロセスによる過度の CPU 使用** ランタイム・プロセスが CPU 時間を消費しすぎると、システム管理者に通知されます。このイベントは、*X* 秒ごとにチェックされます。時間間隔を設定します。アラート基準値、警告基準値および発生回数を選択できます。
- **ランタイム・プロセスによる過度の仮想メモリー使用** ランタイム・プロセスによって消費されている仮想メモリーが一定量を超えると、システム管理者に通知されます。このイベントは、過度の CPU 使用イベントと同様です。次のパラメータを設定できます。イベント間隔、警告基準値（KB 単位の仮想メモリー）、アラート基準値（KB 単位の仮想メモリー）および発生回数の各パラメータ。

14

キャパシティ量計画の考慮事項

14.1 概要

この章では、Forms Server の拡張性機能について考察します。広く使用されているハードウェア・プラットフォームとオペレーティング・システムを用いて、いくつかのベンチマーク・テストを実行することによって、サーバーの拡張性を調査しました。

次のベンチマークを測定しました。

- ユーザーあたりの RAM
- CPU あたりのユーザー数

Forms Server 6.0 では、次の結果が得られました。

Windows NT の場合 :

表 14-1 Windows NT でのベンチマーク

アプリケーションのサイズ / 複雑さ	ユーザーあたりの RAM (MB)	CPU あたりのユーザー数
標準 / 普通	2.5-6.0	100-300
小 / 単純	1.0-2.5	150-300

Sun Solaris の場合 :

表 14-2 Sun Solaris でのベンチマーク

アプリケーションのサイズ / 複雑さ	ユーザーあたりの RAM (MB)	CPU あたりのユーザー数
標準 / 普通	2.0-5.0	200-400
小 / 単純	1.0-2.0	300-500

注意 : この章に記載されている結果は、Forms Server のリリース *6i* に固有であり、以前のリリースの製品には

適用されません。このリリースは、以前のリリースと比較してパフォーマンスが改善されています。パフォーマンスの改善は、次のようないくつかのアーキテクチャおよびコードの最適化によるものです。

- Windows NT のもとでの動的リンク・ライブラリ共有の改善
- 中間層レコードのキャッシングの改善
- メッセージ・レイヤーの改善によるサーバー上の全体的な処理の削減

ベンチマーク・テストは、Oracle 社で現在でも進行中の作業です。ここに示す図は、本書の記述時点で利用可能な情報を表しています。追加の結果は、利用可能になりしだい公開されます。

14.2 拡張性とは

拡張性とは、基盤となるソフトウェアは変更しないでシステムにハードウェア・リソースを追加することで、ある単一システム上のユーザー数の増加に適応できる能力です。拡張可能なシステムは、企業のニーズの拡大に適応できます。

パフォーマンス・ニーズに合せて拡張できるハードウェアとソフトウェアを選択することは、パフォーマンス・ニーズが変わるたびに新しいソフトウェアを購入することよりもはるかに優れた方針です。

次の事項を検討してください。

- アプリケーションまたはオペレーティング・システムは、どのようにすれば追加システム・リソースを有効に利用できるか。
- *n* 人のユーザーをサポートするためには、メモリーがどの程度必要か。
- より高速なプロセッサまたは複数のプロセッサに簡単にアップグレードできるか。
- 追加のプロセッサによって処理能力はどの程度増加するか。
- パフォーマンスを向上させるために後で追加できる機能はあるか（キャッシュやドライブ・アレイ・コントローラなど）。

これらの質問に対する回答は、使用するハードウェア、オペレーティング・システムおよびアプリケーション・ソフトウェアに大きく依存します。

14.3 システム・キャパシティの評価基準

ネットワーク化されたアプリケーションの拡張性は、アプリケーション・サーバーの能力と、ユーザー負荷の増加に予想どおりに適応するためのネットワーク・トポロジに関連します。

この項で説明する各コンポーネントの役割と、それらのコンポーネントが特に Forms Server 環境で、システムの全体的な拡張性にどのように影響するかを理解しておくと役に立ちます。

この章では、最もよく使用されるサーバー・ハードウェアとオペレーティング・システムの2つの組合せを例として使用します。その組合せとは、Sun UltraSparc アーキテクチャ上で実行される Sun Solaris と、Intel アーキテクチャ上で実行される Microsoft Windows NT です。

次の領域は、Forms Server ベースシステムの評価で重要です。

- プロセッサ
- メモリー
- ネットワーク
- 共有リソース
- ユーザー負荷
- アプリケーションの複雑さ

14.3.1 プロセッサ

より高速な動作か、より効率的なな動作か。プロセッサ・テクノロジは、両方のアプローチを探索しました。通常、企業は2～3年ごとに新世代アーキテクチャ（より効率的な動作）をリリースします。これらのリリースの間に、プロセッサ速度が向上します（より高速な動作）。*クロック速度*とも呼ばれるプロセッサの速度は、通常はメガヘルツ（MHz）で表されます。プロセッサ速度は、コンピュータ・システムがどの程度高速に稼動できるかをよく示します。通常は、サーバーとして使用されるコンピュータは、複数のプロセッサを使用し、マルチプロセッサ・システムと呼ばれます。

Forms Server に関して我々が実際に興味を持つ基準値は、各プロセッサ上の同時ユーザー数です。この基準値は、プロセッサあたりのユーザー数と呼ばれることもあります。この数値は、プロセッサのタイプによって大きく異なります。この変化の例は、14-1 ページの表 14-1 と 14-1 ページの表 14-2 を参照してください。

ベンチマークで収集された経験データによると、400MHz の Intel Pentium II Xeon プロセッサと 1MB の L2 キャッシュを搭載したコンピュータは、200MHz Pentium Pro システムと比較して、約2倍のユーザー数をサポートできます。

14.3.2 メモリー

メモリーは、コンピュータ・システムがプログラムの起動と実行に使用できる RAM の容量です。コンピュータ・システムの RAM の容量は、通常はメガバイト（MB）で表されます。

プログラムの通常の実行では、プログラムは RAM にロードされ、プログラムが非アクティブになるたびに、オペレーティング・システムがプログラムをディスクにスワップします。オペレーティング・システムは、プログラムがアクティブになると、そのプログラムを RAM に戻します。

このアクティビティは、一般にスワッピングと呼ばれています。Sun Solaris や Microsoft Windows NT などのほとんどのオペレーティング・システムは、通常の操作中にスワッピングを実行します。スワッピングによって、プロセッサの需要が増加します。過度のスワッピングは、システムの処理速度をかなり低下させる傾向があります。パフォーマンスの低下を防ぐには、サーバー・ホスト・マシンに十分な RAM を搭載してください。

重要な基準値は、Forms Server を介してアプリケーションに接続し実行するすべての追加ユーザーが必要とする RAM です。この基準値は、ユーザーあたりのメモリーとも呼ばれます。通常は、パフォーマンス測定ツールは、ユーザーあたりのメモリーを正確に測定しません。この基準値を入念に調査して、メモリー要件を判断します。ユーザーあたりのメモリーの例は、14-1 ページの表 14-1 と表 14-2 を参照してください。

14.3.3 ネットワーク

Forms Server のような多層のインターネットベース・アーキテクチャでは、クライアントを Forms Server に接続する物理的なネットワーク、および Forms Server とデータベースの間の接続は、システムの全体的な拡張性の重要な要因となります。Forms Server ベース・システムのパフォーマンスを測定するときは、物理ネットワークのパフォーマンスに注意してください。

14.3.4 共有リソース

マルチユーザー、マルチプロセス環境での個々のプロセスのパフォーマンスは、メイン・メモリーで処理される個々のプロセスの能力に直接比例します。つまり、他のプロセス用の領域を空けるために、必要なページが仮想メモリーにスワップされると、パフォーマンスが悪影響を受けます。必要なページがメイン・メモリーに見つかる可能性を高める 1 つの技法は、イメージ・マップ・メモリーを使用して共有メモリー・モデルをインプリメントすることです。イメージ・マップ・メモリーは、メモリー内のファイルの内容を、プロセス間で共有される特定のアドレス空間に関連付けます。

Forms Server はイメージ・マップ・メモリーを使用します。個々の Forms プロセスは、FMX ファイル・イメージの大部分を共有するので、個々のメモリー要件が低減し、全体的な拡張性が向上します。

14.3.5 ユーザー負荷

ベンチマーク・シナリオでは、実際のアプリケーション環境を正確に作り出すために多数のクライアント・マシン（およびユーザー）を設定するのは、実際的ではありません。ベンチマークでは、負荷シミュレータを使用して、アプリケーション・サーバーでトランザクションを実行する実際のユーザーをシミュレートします。Oracle Tools の開発部門は、負荷シミュレータを開発しました。このシミュレータは、サーバーにメッセージを送信して負荷をシミュレートすることで、実世界の Forms Server ユーザーを模倣します。負荷シミュレータは、Forms Server と UI クライアントの間に位置し、これら 2 つのコンポーネント間のメッセージ・トラフィックをインターセプトする小さな Java アプリケーションです。

クライアントからのイベント・メッセージが記録されると、そのメッセージをサーバーに再生できます。これにより、実際のユーザー・セッションがシミュレートされます。（UI クライアントは、再生モードには関係しないことに注意してください。）サーバーへの再生中に、負荷シミュレータは多数のユーザー・セッションを再生できます。この方法では、負荷シミュレータは、クライアントとサーバー間のメッセージの往復時間の合計を判断することによって、ユーザーへの合計応答時間を計算できます。あるビジネス・トランザクション全体の合計応答時間を累計することによって、アプリケーション・パフォーマンスの測定可能な基準値を取得できます。

14.3.6 アプリケーションの複雑さ

値リスト（LOV）とポップアップ・ウィンドウを含む単純な単一 Form から、複数の Forms と PL/SQL ライブラリ（PLL）を同時にオープンする複雑なアプリケーションにいたるまで、さまざまな複雑さの Forms アプリケーションをテストしました。アプリケーションの複雑さは、ある 1 つのモジュールに固有の複雑さではなく、ユーザーが一度にアクセスできるモジュール数に関連付けました。

複雑さを判断するのによい方法は、Form に追加されたすべての依存性を参照することです。たとえば、フォームは、CALL_FORM または OPEN_FORM ビルトインを介して他のフォームをコールすることがあります。また、メニュー（MMX ファイル）に接続することや、PL/SQL ライブラリ（PLL ファイル）を使用して外部ビジネス・ロジックをロードすることもあります。これらすべての要因は、ユーザーあたりのメモリー使用量に寄与します。

次の表に、Oracle Forms アプリケーションの複雑さのレベルを分類します。

表 14-3 アプリケーションの複雑さの判断

アプリケーションのサイズ / 複雑さ	メモリー内の同時モジュールの合計サイズ
大 / 複雑	> 10MB
標準 / 普通	2 ～ 10MB
小 / 単純	< 2MB

複雑さの異なる 2 つのアプリケーションをテストしました。

- 最初のアプリケーションは、適切なメニューと値リストを含む単純な「顧客注文入力」画面でした。どの時点でも、アクティブなフォームは 1 つのみでした。
- 2 番目のアプリケーションは、複雑さが普通のアプリケーションでした。実際の顧客アプリケーション、ヘルプ・デスクおよび顧客サポート・システムを使用しました。このアプリケーションには、同時にオープンされる多数のモジュールがあり、個々のモジュール内には複雑なビジネス・ロジックがありました。

現実的なユーザー・コミュニティ、つまり複合的な作業負荷が存在するコミュニティを表すために、テストでは、45 分間のシナリオに、サービス・デスク要員が実行するアクティビティを模倣したいくつかのトランザクションを含めました。

インプリメントした作業のステップごとの定義。

ステップ	実行した作業
1	サービス表示アプリケーションの起動 - ログイン
2	通知画面へ [ナビゲート]
3	進捗画面を [コール] トランザクション : パラメータ化した問合せの入力
4	問題画面を [オープン] 各種タブ（PL/SQL の実行）へ [ナビゲート] 画面上のすべてのフィールドへ [ナビゲート]
5	サービス画面を [コール] トランザクション : ブラインド問合せの入力 問い合わせたすべてのフィールドへ [ナビゲート]
6	シナリオ 2 〜 5 の [繰返し]

14.4 拡張性の基準値の判断

ユーザー負荷が増えたときに感じられるパフォーマンスの低下感を測定するためには、最初に、特定のユーザーが特定のアプリケーション作業を実行するのにかかる時間を判断する必要があります。この合計応答時間基準値は、単に特定の物理トランザクションやネットワークの往復の応答時間をテストするのとは異なります。この基準値は、（平均的なユーザーが）当面のビジネス・タスクを実行するのにかかる合計時間（つまり、ビジネス・トランザクションの一部として Forms Server とデータベースの間で行われるすべての対話の合計）を参照します。

全体的なシステム・リソースに関する経験的な情報を得るために、拡張性テストでは、オペレーティング・システムに固有の監視ユーティリティ（Windows NT のパフォーマンス・モニタなど）も使用して、物理メモリーと仮想メモリーの使用量および CPU の合計使用量の値を判断します。

合計応答時間基準値を経験的な測定値とともに使用することで、ユーザー負荷が増えたときに、特定のユーザーのパフォーマンスが大幅に低下するポイントを判断できました。許容されるパフォーマンスでサポートできるユーザー数を判断したところ、個々のメモリー消費は、アプリケーションにアクセスするユーザー数で除算した合計使用可能メモリーの単純な等式になりました。

例 :

512MB の RAM のある特定のハードウェア・プラットフォームでは、60 人の同時ユーザーまではパフォーマンスが一定です。それを超えると、パフォーマンスは大幅に低下します。これにより、サポートされる最大ユーザー数は 60 人であると規定できます。

通常のオペレーティング・システム・オーバーヘッド（～ 32MB）を考慮すると、個々のメモリー使用量は、(512-32) / 60、つまりユーザーあたり 8MB になります。

14.5 サンプルのベンチマーク結果

次の項では、次のシナリオについて、テストしたシステム、テストの結果および簡単な分析を定義します。

- 低コストの Intel Pentium ベース・システム上の、標準的な複雑さのアプリケーション
- Intel Pentium II Xeon-Base システム上の、標準的な複雑さのアプリケーション
- エントリレベルの Sun UltraSparc サーバー上の、標準的な複雑さのアプリケーション
- Intel Pentium II Xeon-Base システム上の単純なアプリケーション
- エントリレベルの Sun UltraSparc サーバー上の単純なアプリケーション

14.5.1 低コストの Intel Pentium ベース・システム上の、標準的な複雑さのアプリケーション

パラメータ :

アプリケーションのサイズ / 複雑さ	CPU	RAM	オペレーティング・システム	スワップ
標準（2 ～ 10MB）	2 つの 200 MHz Pentium Pro	512MB	Windows NT 4.0 Server（SP 3）	2GB

結果 :

CPU あたりのユーザー数	ユーザーあたりのメモリー
100	2.4MB

分析 :

このシステムは、標準的な複雑さのアプリケーションの拡張性をテストするために使用した最も安価なシステムの 1 つです。システムは、約 200 ユーザーを非常に効率よく処理できました。200 ユーザーを超えると、パフォーマンスが劇的に低下しました。このシステムは、標準クラスの複雑さに分類されるアプリケーションを最大 200 ユーザーが使用する小規模部門サーバーとして、費用効果があります。

14.5.2 Intel Pentium II Xeon-Base システム上の、標準的な複雑さのアプリケーション

パラメータ :

アプリケーションのサイズ / 複雑さ	CPU	RAM	オペレーティング・システム	スワップ
標準（2 ～ 10MB）	1MB L2 キャッシュを搭載した 2 つの 400 MHz Pentium II Xeon	512MB	Windows NT 4.0 Server（SP 3）	2GB

結果 :

CPU あたりのユーザー数	ユーザーあたりのメモリー
200	1.2MB

分析 :

このシステムは、標準的な複雑さのアプリケーションの拡張性をテストするために使用した最新の Intel Pentium II Xeon-Base サーバーの 1 つです。システムは、約 400 ユーザーを非常に効率よく処理しました。400 ユーザーを超えると、パフォーマンスが劇的に低下しました。システムは、大規模な部門サーバーまたは小～標準規模ビジネス用のエントリレベル Enterprise Server として、費用効果があります。

14.5.3 エントリレベルの Sun UltraSparc サーバー上の、標準的な複雑さのアプリケーション

パラメータ :

アプリケーションのサイズ / 複雑さ	CPU	RAM	オペレーティング・システム	スワップ
標準	2 つの 248 MHz Ultra Sparc	512MB	Solaris 2.5.1	2GB

結果 :

CPU あたりのユーザー数	ユーザーあたりのメモリー
200	1.3MB

分析 :

システムは、約 375 ユーザーを非常に効率よく処理しました。375 ユーザーを超えると、パフォーマンスが劇的に低下しました。システムは、過度のページングとスワッピング・アクティビティにより速度が低下するようで、実際のボトルネックは物理メモリーであったことを示しています。このシステムは、標準的な複雑さのアプリケーションを実行する大規模部門または小～標準規模のビジネスで、費用効果があります。

14.5.4 Intel Pentium II Xeon-Base システム上の単純なアプリケーション

パラメータ :

アプリケーションのサイズ / 複雑さ	CPU	RAM	オペレーティング・システム	スワップ
小（2MB 未満）	1MB L2 キャッシュを搭載した 2 つの 400 MHz Pentium II Xeon	512MB	Windows NT Server 4.0（SP 3）	2GB

結果 :

CPU あたりのユーザー数	ユーザーあたりのメモリー
250	1MB

分析 :

Pentium II Xeon-Base のサーバーは、小規模なアプリケーションで 500 ユーザーを非常に効率よく処理しました。

14.5.5 エントリレベルの Sun UltraSparc サーバー上の単純なアプリケーション

パラメータ :

アプリケーションのサイズ / 複雑さ	CPU	RAM	オペレーティング・システム	スワップ
小（2MB 未満）	2 つの 248 MHz Ultra Sparc	512MB	Solaris 2.5.1	2GB

結果 :

CPU あたりのユーザー数	ユーザーあたりのメモリー
240	1MB

分析 :

このシステムは、エントリレベルの Sun Ultra Sparc システムです。システムは、約 480 ユーザーを非常に効率よく処理しました。480 ユーザーを超えると、パフォーマンスが劇的に低下しました。

15

トラブルシューティング・ソリューション

15.1 概要

この章には、Forms Server のトラブルシューティング・ソリューションに関する情報が次の項に記載されています。

- Forms Server のステータスのチェック
- Forms Server 開始
- Forms Server プロセスの停止
- Forms Server ログの開始
- トラブルシューティングの FAQ

15.2 Forms Server のステータスのチェック

Forms Server のステータスをチェックする手順は、次のとおりです。

Microsoft Windows NT の場合 :

1. **[Control]+[Alt]+[Delete]** を押して、「Windows NT のセキュリティ」ダイアログを表示します。
2. 「**タスク・マネージャ**」を選択します。
3. タスク・マネージャで、「**プロセス**」タブをクリックします。

サーバー・プロセスが実行中の場合、タスク・マネージャは、IFSRV60.EXE と呼ばれるプロセスと、IFWEB60.EXE と呼ばれる複数のプロセス（アクティブな接続ごとに 1 つずつ）を表示します。

UNIX の場合 :

UNIX プロンプトで `ps -ef | grep f60srvm` と入力し、**[Enter]** キーを押します。

プロセス ID のリストが画面に表示されます。Listener が実行されている場合は、リストには f60srvm というプロセスと、f60webm プロセスの複数のオカレンスが含まれます。（アクティブな接続ごとに 1 つのプロセスがあり、*pool* のデフォルト値が使用されている場合は次のユーザーに備える予備の接続が 1 つあります。*pool* が 5 に設定されている場合は、5 つの予備接続があります。）

15.3 Forms Server 開始

Forms Server を開始する手順は、次のとおりです。

Microsoft Windows NT 上のサービスとして開始する場合 :

既存の Forms Server サービスを削除し、新しい起動パラメータを再インストールできます。

1. コマンド・ウィンドウで、次のように入力します。

```
ifsrv60 -remove <FormsServerServiceNameToBeRemoved>
```

2. 次のように入力します。

```
ifsrv60 -install <NewFormsServerServiceName> port=<portNum> mode=<socket/http/https>
[pool=<numOfRunforms> log=<logfilePath> exe=<RunformexeName>]
```

3. **[Enter]** キーを押します。サーバー・プロセスが、指定されたポート番号で実行を開始します。

起動パラメータ定義は、5.3.3 項「Forms Server 起動パラメータの説明」を参照してください。

Microsoft Windows NT 上のコンソール・モードで開始する場合 :

1. タスクバーで、「**スタート**」→「**ファイル名を指定して実行**」を選択します。
2. 次のように入力します。

```
<ORACLE_HOME>¥bin¥ifsrv60 <FormsServerName> port=<portNum> mode=<socket/http/https>
[pool=<numOfRunforms> log=<logfilePath> exe=<RunformexeName>]
```

3. **[Enter]** キーを押します。サーバー・プロセスが、指定されたポート番号で実行を開始します。

起動パラメータ定義は、5.3.3 項「Forms Server 起動パラメータの説明」を参照してください。

UNIX の場合 :

1. UNIX プロンプトで、次のように入力します。

```
cd <ORACLE_HOME>/bin.
```

2. **[Enter]** キーを押します。
3. 次のように入力します。

```
f60srvm port=port_number &
```

4. **[Enter]** キーを押します。指定されたポート番号で、サーバーが（バックグラウンドで）実行を開始します。

起動パラメータ定義は、5.3.3 項「Forms Server 起動パラメータの説明」を参照してください。

15.4 Forms Server プロセスの停止

Forms Server プロセスを停止する手順は、次のとおりです。

Microsoft Windows NT 上の NT サービスとして停止する場合 :

1. コントロール・パネルで、「**サービス**」を選択します。
2. Forms Server プロセスを探し、選択します。
3. 「**停止**」をクリックします。

Microsoft Windows NT 上のコンソール・モードで停止する場合 :

1. Forms Server のステータスをチェックします。サーバーが実行中の場合、タスク・マネージャは、IFSRV60.EXE と呼ばれるプロセスを表示します。
2. IFSRV60.EXE を選択し、「**プロセスの終了**」をクリックします。

UNIX の場合 :

1. Forms Server のステータスをチェックします。プロセス ID のリストが画面に表示されます。f60srvm プロセスのプロセス ID に注意してください。
2. UNIX プロンプトで、次のように入力します。

```
kill process_ID
```

または、次のように入力します。

```
kill -g
```

3. **[Enter]** キーを押します。

15.5 Forms Server ログの開始

次のように log オプションを使用してサーバーを開始すると、Forms Server はログ・ファイルを作成します。

```
ifsrv60 -install Forms60Server log=<¥PathName¥LogFileName> port=<portNum>
mode=<socket/http/https>
```

ログには、診断情報が含まれます。

15.6 トラブルシューティングの FAQ

問題	ソリューション
Web 対応の Forms アプリケーションを、Java 対応でない Web ブラウザで実行できない。	Web ブラウザが Java 対応かどうかが不明な場合は、Web ブラウザのネットワーク作業環境をチェックします。「Java を使用可能にする」および「JavaScript を使用可能にする」チェック・ボックスがチェックされている必要があります。
Forms Server を開始しようとすると、エラー・メッセージ " ポート 9000 にバインドできません " が表示される。	別のプロセスがポートを使用している可能性があります。そのプロセスは、Forms Server の別のオカレンスである可能性があります。Forms Server がまだ実行されていないことをチェックしてください。直前に Forms Server を停止した場合は、ポート 9000 への既存の接続がリオープンされるまでに 1 ～ 2 分かかることがあります。
Forms クライアントが Web ブラウザにダウンロードされない。	Oracle Java クラス・ファイル（コードベース）をポイントする仮想ディレクトリが定義されていることをチェックします。
すべての接続データが正しいにもかかわらず、クライアントがサーバーに接続できない。	サーバーが 128 ビットの暗号化（ドメスティック・ライセンス）を使用し、クライアントが（エクスポート・ライセンスのもとで 40 ビットの暗号化を使用しているため）この暗号化をサポートできない場合は、FORMS60_HTTPS_NEGOTIATE_DOWN 環境変数をチェックします。この変数が FALSE に設定されている場合は、サーバーはクライアントの接続リクエストを拒否します。必要に応じて、Java コンソールとサーバー・ログ・ファイル（使用可能な場合）をチェックして、クライアントとサーバーが使用する暗号化のレベルを確認します。
Forms Server が、アプリケーションのベース HTML ファイルで渡したユーザー ID、パスワードおよびデータベース SID パラメータ値を無視しているように見える。	値の前にパラメータ "userid=" があることを確認します。例 : userid=scott/tiger@inventory
Forms Server が、変数の変更を認識しないように見える。	Forms Server を停止し、再起動します。

問題	ソリューション
セキュリティ・ファイアウォールを使用しているときに、プロキシ・サーバーを使用してファイアウォールの外部にアクセスすると、問題が発生する。	プロキシが手動構成に設定されていることを確認します。
HTML ページとアプレットが起動時にダウンロードされ、アプレットが実行を開始するが、他のものは実行されていないように見える。	次の項目をチェックします。 最初に、Forms クライアントが本当に実行されていることを確認します。実行されている場合は、Web ブラウザのステータス・バーに「applet oracle.forms.engine.Main を実行中」というメッセージが表示されます。 このメッセージが表示されているにもかかわらず、アプリケーションが表示されない場合は、次の項目をチェックします。 1. Forms Server と Web サーバーが、同じアプリケーション・サーバーにインストールされていることを確認します。現在の Java 制限により、これらは同じサーバーにインストールする必要があります。 2. アプリケーションのベース HTML ファイルと構成ファイルをチェックして、.FMX ファイルに対して有効なディレクトリ・パスとファイル名が指定されていることを確認します。仮想ディレクトリ・パスではなく、物理ディレクトリ・パスを使用する必要があります。 3. Java コンソールを表示するように Web ブラウザの作業環境を設定してみます。この設定によって、ランタイム Java エラー・メッセージを参照できます。
アプレットが Forms Server に接続できない。	サーバー上の "mode" 設定が、ベース HTML ファイルの "connectionType" と一致していることを確認します。
ローカル・データベースへの接続で問題が発生する。	次の原因が考えられます。 * Net8 v2 接続文字列を指定していない場合は、エラーが発生します。Forms Server のランタイム・エンジンは、LOCAL、TWO_TASK などの型の接続文字列をアクセプトしません。 * Net8 v2 接続文字列を使用しているにもかかわらず、データベースに接続できない場合は、Forms Server が実行されていることを確認します。ほとんどのインストレーションでは、サーバーはリブート後に自動的に再起動されません。 * クライアント・マシンではなくアプリケーション・サーバー上の TNSNAMES.ORA ファイルに、有効な接続文字列が必要です。アプリケーション・ロジックは、ユーザーのクライアント・マシンではなくアプリケーション・サーバー上で実行されます。
CLASSPATH 環境変数を変更した後に、予期しない動作が発生する。	アプリケーション・サーバーまたはユーザーのマシン上で CLASSPATH 環境変数の設定を変更すると、予期しない結果が生成される可能性があります。この変数を、Forms Java クラス・ファイルが存在する場所とオーバーラップするディレクトリに設定すると、ファイル名のオーバーラップの原因となることがあります。

問題	ソリューション
いくつかの使用されていないプロセスがサーバー上で実行されているように見える。	Web 対応の Form Builder アプリケーションを実行している各ユーザーについて、アプリケーション・サーバー上で Forms Server ランタイム・プロセス（Windows 上の ifweb60.exe と ifsrv60、UNIX 上の f60webm と f60srvm）が開始することを思い出してください。各ランタイム・プロセスは、ユーザーがアプリケーションを終了したときに終了する必要があります。ユーザーがアプリケーションを正しく終了しないでブラウザを終了すると、プロセスがサーバー上に残ります。アプリケーションを正しく終了するには、メニューまたは [終了 / 取消し] キー機能を使用してからブラウザを終了する必要があります。

第II部

A

Forms Server パラメータ

A.1 概要

この付録には、Forms および Graphics を Forms Server に構成するために使用するパラメータが含まれています。

A.2 Windows 95 および Windows NT のレジストリ

Windows 95 および Windows NT の場合、Oracle Installer は新規の ORACLE セクションをレジストリ内に作成します。Oracle レジストリには、Oracle ホーム・ディレクトリ名、製品の作業環境ファイルの場所、およびヘルプ・ファイルの場所などを管理する構成パラメータが含まれています。Windows の Net8 を使用する場合、構成パラメータはネットワーク通信に使用するドライバおよび Net8 が操作パラメータのために使用する値も判断します。

A.2.1 レジストリの表示および変更

レジストリ・エディタを使用して Microsoft Windows レジストリを表示したり、オプションで編集できます。このエディタは、Windows ソフトウェアがインストールされているディレクトリにあります。

エディタを起動するには、次の手順を実行します。

1. 「**スタート**」→「**ファイル名を指定して実行**」を選択します。
2. `REGEDIT` と入力します。
3. 「**OK**」をクリックします。
4. レジストリ・エディタで、HKEY_LOCAL_MACHINE ノードを拡張します。
5. SOFTWARE ノードを拡張します。
6. Oracle 構成パラメータを表示するには、［ORACLE］キーをクリックします。
7. パラメータ名をダブルクリックして「文字列の編集」ダイアログ・ボックスを表示し、パラメータ値を変更できます。

8. 「値のデータ」テキスト・ボックス内の値を変更します。
9. 「**OK**」をクリックして、新しい値を使用します。

A.3 構成パラメータ

Oracle Installer は多くのパラメータを自動的に設定します。Oracle 製品には一部のパラメータが必要です。これらのパラメータは表 A-1 に列挙されています。その他のパラメータを使用すると、製品の動作をカスタマイズできます。これらについては、A.3.2 項「カスタマイズ可能パラメータ」で説明されています。

A.3.1 必須パラメータ

この項で列挙するパラメータは、Oracle Installer によって自動的に設定されたり、削除されます。これらのパラメータは、様々な Oracle 製品を正しく機能させるために必要です。

注意 : この項で列挙されているパラメータの設定を変更しないでください。変更した場合、1 つ以上の Oracle 製品が正しく機能しなくなることがあります。

次に列挙するパラメータ内の *nn* は、製品またはコンポーネントのリリース番号を指定します。Oracle 製品の新規リリースにアップグレードする場合、この番号は変更されることがあります。

表 A-1 必須パラメータ

パラメータ	設定
BROWSER*nn*	*ORACLE_HOME*¥BROWSE*nn*
DE*nn*	*ORACLE_HOME*¥TOOLS¥COMMON*nn*
FORMS*nn*	*ORACLE_HOME*¥FORMS*nn*
GRAPHICS*nn*	*ORACLE_HOME*¥GRAPH*nn*
MM*nn*	*ORACLE_HOME*¥TOOLS¥COMMON*nn*
OCL*nn*	*ORACLE_HOME*¥GRAPH*nn*
PRO*nn*	*ORACLE_HOME*¥PRO*nn*
RDBMS*nn*	*ORACLE_HOME*¥RDBMS*nn*
RW*nn*	*ORACLE_HOME*¥REPORT*nn*
TK*nn*	*ORACLE_HOME*¥TOOLS¥COMMON*nn*
VGS*nn*	*ORACLE_HOME*¥TOOLS¥COMMON*nn*

A.3.2 カスタマイズ可能パラメータ

この項で列挙されているパラメータは、Oracle 製品の様々なアスペクトを制御します。動作をカスタマイズするために、これらのパラメータ設定を変更できます。

次の項では、各パラメータのデフォルト設定（存在する場合）を列挙します。デフォルト値に自動設定されないパラメータを説明します。パラメータ・リストには、有効値の説明および例が含まれています。

FORMS60_PATH

デフォルト : *ORACLE_HOME*¥FORMS60¥PLSQLLIB

有効値 : すべてのドライブ上のすべてのディレクトリ

例 :

FORMS60_PATH=C:¥oracle¥apps¥forms;C:¥myfiles

このパラメータは、Form Builder ランタイム・アプリケーションで使用されるファイルの検索パスを指定します。これらには、フォーム・ファイル（.fmx）、メニュー・ファイル（.mmx）、PL/SQL ライブラリ（.pll）および実行時にアプリケーションがファイルからロードしようとするその他のオブジェクトが含まれます。たとえば、イメージ・ファイル scooter.tif をインポートする場合、Form Builder はこのファイルを見つけるために、FORMS60_PATH で指定されたディレクトリを検索します。

FORMS60_PATH では複数のディレクトリを指定できます。パス・リスト内でディレクトリ名を分離するには、セミコロン（;）を使用します。

FORMS60_REPFORMAT

デフォルト : なし

有効値 : HTML、PDF

例 :

FORMS60_REPFORMAT=HTML

RUN_PRODUCT を使用してフォームからレポートを実行するためにブラウザを起動する場合、FORMS60_REPFORMAT 環境変数を設定する必要があります。このパラメータはレポート・フォーマットを指定します。

FORMS60_TIMEOUT

デフォルト : 15

有効値 : 1 – 1440（1 日）

例 :

FORMS60_TIMEOUT=1440

このパラメータは、クライアントが Forms Server と通信しない場合に Forms Server プロセスが終了するまでの所要時間（分）を指定します。

GRAPHICS60_PATH

デフォルト : なし

有効値 : すべてのドライブ上のすべてのディレクトリ

例 :

GRAPHICS60_PATH=C:¥oracle¥apps¥graphics;C:¥myfiles

このパラメータは、Graphics ランタイム・アプリケーションで使用されるファイルの検索パスを指定します。これらには、図表ファイル（.ogr）、イメージ、外部問合せ、および実行時にアプリケーションがファイルからロードしようとするその他のオブジェクトが含まれます。たとえば、イメージ・ファイル scooter.tif をインポートする場合、Graphics Builder はこのファイルを見つけるために、GRAPHICS60_PATH で指定されたディレクトリを検索します。

GRAPHICS60_PATH では複数のディレクトリを指定できます。パス内のディレクトリを分離するには円記号（¥）を使用し、完全パスを分離するにはセミコロン（;）を使用します。

NLS_LANG

デフォルト : AMERICAN_AMERICA.WE8ISO8859P1

有効値 : 現行の使用可能な値リストの詳細は『*NLS ガイド*』を参照してください。または CD の中の ¥bonus¥nls¥nlsd2r1.wri を参照してください。

例 :

NLS_LANG=AMERICAN_AMERICA.WE8ISO8859P1

このパラメータは、メッセージ・ファイルで表示される言語を設定します。NLS_LANG の構文の一例を次に示します。

NLS_LANG=<*language*>_<*territory*>.<*char_set*>

各項目の内容は次のとおりです。

- *Language* は、メッセージ、曜日および月の名前を表示するための言語およびその規則を指定します。
- *Territory* は、週および日数を計算するための地域およびその規則を指定します。
- *Char_set* は、UPPER、LOWER および INITCAP ファンクションで使用されるキャラクタ・セット、および ORDER BY 問合せで使用されるソート・タイプを指定します。この引数も、メッセージを表示するために使用されるキャラクタ・セットを制御します。

ORACLE_HOME

デフォルト : Window95 の場合は C:¥ORAWIN95、Windows NT の場合は C:¥ORANT

有効値 : すべてのドライブ上のすべてのディレクトリ

例 :

ORACLE_HOME=C:¥orawin95

このパラメータは、Windows 版 Oracle 製品がインストールされるホーム・ディレクトリを指定します。このディレクトリは、Oracle ディレクトリ階層で一番上のディレクトリです。

B

Oracle JInitiator

B.1 概要

この付録では、Oracle JInitiator をユーザーの Web ブラウザのプラグインとして使用する利点について説明します。ユーザーは Oracle JInitiator を使用すると、Netscape Navigator または Internet Explorer を使用して Forms Server アプリケーションを実行できます。ブラウザのデフォルトの JVM ではなく、クライアント上の特定の Java 仮想マシン (JVM) を使用するように指定する機能が提供されます。

Oracle JInitiator は、Netscape Navigator のプラグイン、および Internet Explorer の ActiveX コンポーネントとして実行されます。Oracle JInitiator はブラウザによって提供されるデフォルトの JVM を、置き換えたり変更せずに、代替の JVM をプラグイン形式で提供します。かわりに、プラグインのフォームで代替の JVM が提供されます。

B.1.1 Oracle JInitiator を使用する理由

Oracle JInitiator では、Web ブラウザを使用して透過的に起動できる、保証されサポート可能な Java Runtime Environment（JRE）がクライアントのデスクトップに提供されます。

Oracle JInitiator は、JavaSoft の Java Plug-in の Oracle バージョンです。JavaSoft Plug-in は、ブラウザ内から起動できる JavaSoft JRE の配布メカニズムです。また、Oracle JInitiator では、Oracle が保証した JRE の配布メカニズムが提供されます。これにより、Forms Developer アプリケーションを安定しサポートされた方法で、ブラウザ内から実行できます。

Oracle JInitiator では、Forms Developer アプリケーションを実行するために保証されたプラットフォームが提供されるのみではなく、標準 JavaSoft Java Plug-in の他に多くの追加機能も提供されます。これらの機能には、JAR ファイルのキャッシュ書込み、増分の JAR ファイルのロード、およびアプレットのキャッシュ書込みが含まれます。

B.1.2 Oracle JInitiator の利点

Oracle JInitiator には次の利点があります。

- 旧リリースのブラウザで、最新の Oracle が保証する JVM を実行できます。
- 異なるブラウザ間で JVM の一貫性が保証されます。
- 信頼性が高い実行プラットフォームです。JInitiator は、Forms Server での使用が完全にテストされ、保証されています。
- 高性能な実行環境です。JInitiator によって、アプリケーション・クラス・ファイルがキャッシュに自動的に書き込まれます。これにより、アプリケーションを高速起動できます。
- 自己インストールおよび自己メンテナンスが可能な実行環境です。JInitiator はプラグインまたは ActiveX コンポーネントのように、自身を自動的にインストールおよび更新します。ローカルにキャッシュされたアプリケーション・クラス・ファイルは、アプリケーション・サーバーから自動的に更新されます。

B.2 Oracle JInitiator の使用方法

クライアントのブラウザが Oracle JInitiator の使用が指定された HTML ファイルを最初に見つけたとき、Oracle JInitiator はアプリケーション・サーバーからクライアント・マシンに自動的にダウンロードされます。これにより、Windows 95 および Windows NT 4.0 プラットフォーム上の Netscape Navigator または Internet Explorer 内で Forms および Graphics アプリケーションを直接実行できます。

Oracle JInitiator は、ブラウザによって提供される標準プラグイン・メカニズムを使用してインストールおよび更新されます。Oracle JInitiator のインストールでは、Forms Developer アプリケーションを信頼されたアプレットとして Oracle JInitiator 環境で実行するために必要なステップが実行されます。

B.2.1 サポートされる構成

Oracle JInitiator では、次の構成がサポートされています。

	Internet Explorer 4.0	Internet Explorer 5.0	Navigator 4.0	Navigator 4.5
Windows 95	○	○	○	○
Windows NT	○	○	○	○

B.2.2 システム要件

Oracle JInitiator の最小システム要件を次に示します。

- Windows 95 または Windows NT 4.0
- Pentium 90 MHz 以上のプロセッサ
- 12MB のハード・ディスク空き領域（20MB を推奨）

- 16MB のシステム RAM（24MB を推奨）

B.2.3 Netscape Navigator での Oracle JInitiator の使用方法

Oracle JInitiator では、QuickTime ムービーまたは Shockwave アニメーション機能など他のプラグインと同様にブラウザ内で実行できるように、Netscape Navigator プラグイン・アーキテクチャが使用されています。Web アプリケーション開発者は Netscape HTML <EMBED> タグを使用して、プラグインを Web ページの一部として実行するように指定できます。これにより、ユーザーの介在を最小にして、Oracle JInitiator を Web ブラウザ内で実行できます。

Navigator が Oracle JInitiator の使用が指定された HTML ページを最初に見つけたとき、Oracle JInitiator ダウンロード・ページを指示する HTML ページに "Plug-in Not Loaded" ダイアログが表示されます。続いて、各オペレーティング・システム用の Oracle JInitiator バージョンをダウンロードし、インストールできます。

Oracle JInitiator のインストール後、Navigator をシャットダウンして、再起動します。続いて、元の HTML ページを再び表示します。Oracle JInitiator はアプレットを解放するために、<EMBED> タグ内のパラメータを実行および使用します。Navigator が Oracle JInitiator の使用が指定された Web ページを次に見つけたとき、ユーザーが介在することなく、Navigator はプラグインをローカル・ディスクから透過的にロードおよび実行します。

B.2.4 Microsoft Internet Explorer での Oracle JInitiator の使用方法

Oracle JInitiator は Microsoft Internet Explorer 拡張メカニズムを使用して、ActiveX コントロールおよび COM コンポーネントのダウンロードおよびキャッシュ書込みを行います。Web アプリケーション開発者は HTML <OBJECT> タグを使用して、ActiveX コントロールまたは COM コンポーネントを Web ページの一部として実行するように指定できます。このようなコンポーネントには Oracle JInitiator が含まれます。

Internet Explorer は Oracle JInitiator 使用を指定するように変更された HTML ファイルを最初に見つけたとき、オラクル社によって VeriSign デジタル署名が行われた ActiveX コントロールをダウンロードするかどうかをユーザーに確認します。「はい」をクリックすると、Internet Explorer は Oracle JInitiator のダウンロードを開始します。Oracle JInitiator が実行され、アプレットの内容を表現するために <OBJECT> タグ内のパラメータが使用されます。Internet Explorer が Oracle JInitiator をサポートするように変更された Web ページを次に見つけたとき、ユーザーが介在することなく、Oracle JInitiator をローカル・ディスクから透過的にロードおよび実行します。

B.2.5 Oracle JInitator プラグインの設定

Oracle JInitator プラグインを設定するには、次の手順を実行します。

- Oracle JInitiator HTML マークアップをベース HTML ファイルに追加します。
- Oracle JInitiator をサーバーにインストールします（サーバー・ベースのテスト用のみ）。

- Oracle JInitiator ダウンロード・ファイルをカスタマイズします。
- Oracle JInitiator をダウンロード可能にします。

B.2.5.1 Oracle JInitiator マークアップのベース HTML ファイルへの追加

Oracle JInitiator マークアップをベース HTML ファイルに追加するには、次の手順を実行します。

1. ベース HTML ファイルをテキスト・エディタでオープンします。
2. OBJECT および EMBED タグを追加します。

   追加マークアップの例は B.2.7 項「ベース HTML ファイルの Oracle JInitiator タグ」を参照してください。

B.2.5.2 Oracle JInitiator の Web サーバーへのインストール

Oracle JInitiator をサーバーにインストールすると、アプリケーションを顧客のために配置する前に、構成をテストおよび改善できます。これは必須のステップではないことに注意してください。これは、ローカル・システム・テストのみに役立ちます。

Oracle JInitiator を Web サーバーにインストールするには、次の手順を実行します。

1. jinit11711.EXE をダブルクリックします。
2. インストール手順に従ってください。

B.2.5.3 Oracle JInitiator ダウンロード・ファイルのカスタマイズ

Oracle JInitiator のダウンロード・ファイル（JINIT_DOWNLOAD.HTM）は、ユーザーが Oracle JInitiator ファイルをダウンロードできるテンプレート HTML ファイルです。

Oracle JInitiator ダウンロード・ファイルをカスタマイズするには、次の手順を実行します。

1. JINIT_DOWNLOAD.HTM ファイルを HTML またはテキスト・エディタでオープンします。
2. 必要に応じてテキストを変更します。
3. 変更を保存します。

B.2.5.4 Oracle JInitiator をダウンロード可能にする

Oracle JInitiator をダウンロード可能にするには、次の手順を実行します。

1. jinit11*x*.EXE を Web サーバーにコピーします。

   jinit11*x*.EXE は、ベース HTML ファイル内で指定された位置にコピーする必要があります。

2. JINIT_DOWNLOAD.HTM を Web サーバーにコピーします。

   JINIT_DOWNLOAD.HTM は、ベース HTML ファイル内で指定された位置にコピーする必要があります。

B.2.6 Oracle JInitiator プラグインの変更

Oracle JInitiator プラグインを変更するには、次の手順を実行します。

- **Oracle JInitiator のキャッシュ・サイズを変更します。**
- **Oracle JInitiator のヒープ・サイズを変更します。**
- **Oracle JInitiator 出力を表示します。**

B.2.6.1 Oracle JInitiator キャッシュ・サイズの変更

Oracle JInitiator のキャッシュ・サイズを変更するには、次の手順を実行します。

1. 「スタート」メニューで、**「スタート」→「プログラム」→「Oracle JInitiator」→「Control Panel」**を選択します。
2. 「**Basic**」タブをクリックします。
3. 「Java Run Time Parameters」フィールドで、Dcache サイズを指定します。たとえば、Dcache.size=20000000 と指定すると、キャッシュ・サイズは 20MB に設定されます。

Oracle JInitiator のデフォルト・キャッシュ・サイズは 20000000 です。これは Oracle JInitiator のインストール時に自動設定されます。

B.2.6.2 Oracle JInitiator ヒープ・サイズの変更

Oracle JInitiator のヒープ・サイズを変更するには、次の手順を実行します。

1. 「スタート」メニューで、**「スタート」→「プログラム」→「Oracle JInitiator」→「Control Panel」**を選択します。
2. 「**Basic**」タブをクリックします。
3. 「Java Run Time Parameters」フィールドで、mx サイズを指定します。たとえば、mx64m と指定すると、最大ヒープ・サイズは 64MB に設定されます。

Oracle JInitiator のデフォルト最大ヒープ・サイズは 64MB です。これは Oracle JInitiator のインストール時に自動設定されます。

B.2.6.3 Oracle JInitiator 出力の表示

Oracle JInitiator 出力を表示するには、次の手順を実行します。

1. 「スタート」メニューで、**「スタート」→「プログラム」→「Oracle JInitiator」→「Control Panel」**を選択します。
2. 「**Basic**」タブをクリックします。
3. デバッグ出力を使用可能にするために、「**Show Java Console**」チェックボックスをオンにします。

B.2.7 ベース HTML ファイルの Oracle JInitiator タグ

この例では、Microsoft Internet Explorer および Netscape Navigator 用の Oracle JInitiator マークアップを示します。これらのタグをベース HTML ファイルに追加すると、Netscape および Microsoft のブラウザ内でアプリケーションを実行できます。

```
<HTML>
<BODY>
<P>
<OBJECT classid="clsid:9F77a997-F0F3-11d1-9195-00C04FC990DC"
WIDTH=600
HEIGHT=480
codebase="http://acme.com/jinit11711.exe#Version=1,1,7,11">
<PARAM NAME="CODE" VALUE="oracle.forms.engine.Main" >
<PARAM NAME="CODEBASE" VALUE="/forms60code/" >
<PARAM NAME="ARCHIVE" VALUE="/forms60code/f60all.jar" >
<PARAM NAME="type" VALUE="application/x-jinit-applet;version=1.1.7.11">
<PARAM NAME="serverPort" VALUE="9000">
<PARAM NAME="serverArgs" VALUE="module=order.fmx">
<PARAM NAME="serverApp" VALUE="default">
<COMMENT>
<EMBED type="application/x-jinit-applet;version=1.1.7.11"
java_CODE="oracle.forms.engine.Main"
java_CODEBASE="/forms60code/"
java_ARCHIVE="/forms60code/f60all.jar"
WIDTH=600
HEIGHT=480
serverPort="9000"
serverArgs="module=order.fmx"
serverApp="default"
pluginspage="http://acme.com/jinit_download.htm">
<NOEMBED>
</COMMENT>
</NOEMBED></EMBED>
</OBJECT>
</BODY>
</HTML>
```

B.3 Oracle JInitiator FAQ

次の項では、Oracle JInitiator に関してよく聞かれる質問について詳しく説明します。

- 保証および可用性
- サポート
- インストール
- Oracle JInitiator の操作
- キャッシュ書込み

B.3.1 保証および可用性

Oracle JInitiator はいつから利用可能ですか？

カスタム Oracle Developer アプリケーションの配置のために Forms JInitiator は 1998 年 9 月から利用可能です。Oracle Applications では、1999 年 2 月から Oracle JInitiator の保証を行っています。

Oracle JInitiator はどのようにして配布されるのですか？

Developer リリース 6.0 から、Oracle JInitiator は Forms Developer の配布 CD に収録されています。Oracle JInitiator は、Oracle Web サイト、http://www.oracle.co.jp/download/index.html からダウンロードすることもできます。Oracle JInitiator の更新情報は、オラクル社カスタマ・サポートからも入手できます。

Oracle JInitiator は Windows 以外のプラットフォームで動作するようになりますか？

オラクル社では現在、Oracle JInitiator を Microsoft Windows 以外のプラットフォームに移植する予定はありません。ただし、弊社は Forms Developer アプリケーションを Microsoft Windows 以外のプラットフォームで実行するためのサポートおよび保証を行うため、多くのハードウェア・ベンダーと密接な関係を築いています。

Oracle JInitiator は、Netscape Navigator および Internet Explorer のどのバージョンで保証されていますか？

各 Oracle JInitiator リリースの最終品質保証テスト時に、Oracle JInitiator はこれらのブラウザの最新リリースに対して保証されます。オラクル社は、これらのブラウザの旧リリースについてもサポートを行う予定です。保証されるブラウザのバージョンの詳細は、Oracle JInitiator リリースの添付ドキュメントを参照してください。

JavaSoft Java Plug-in と Oracle JInitiator の違いは何ですか？

主な違いは、Oracle JInitiator には Oracle が保証する JRE が含まれているのに対し、JavaSoft Java Plug-in は JavaSoft JDK リファレンス・インプリメンテーションに同梱されることです。JavaSoft のインプリメンテーションは、Forms Developer アプリケーションで保

証されていません。Forms Developer では JRE に極度の要求が行われるので、弊社は JavaSoft の JRE を厳しい条件下で実行できるように変更しました。

オラクル社は JavaSoft に対して、拡張機能の追加を依頼していますが、JavaSoft が新規バージョンに拡張機能を組み込むことを待てません。

JavaSoft Plug-in は、ブラウザ内から起動できる JavaSoft JRE の配布メカニズムです。また、Oracle JInitiator では、Oracle が保証した JRE の配布メカニズムが提供されます。これにより、Forms Developer アプリケーションを安定しサポートされた方法で、ブラウザ内から実行できます。

オラクル社は Oracle JInitiator 製品を管理しており、製品を完全にサポートしています。オラクル社のお客さまには、オラクル社カスタマ・サポートから、アプリケーションをサポートするために必要な適切なレベルのサポートが提供されます。

Oracle JInitiator では、Forms Developer アプリケーションを実行するために保証されたプラットフォームが提供されるのみではなく、標準 JavaSoft Java Plug-in の他に多くの追加機能も提供されます。これらの機能には、JAR ファイルのキャッシュ書込み、増分の JAR ファイルのロード、およびアプレットのキャッシュ書込みが含まれます。

なぜ、Oracle は JavaSoft が提供する JRE を使用しないで、特定の JRE を保証し配布するのですか？

Forms Developer はコンピュータ化へのコストを削減する方法として、サーバー・ベースの配布に移行するお客さまに対応してきました。また、ビジネスに必要な既存アプリケーションへの投資を保護する必要性を認識しています。

お客さまに Java の一意な要求が行われる Java プラットフォームで、特に与えられたアプリケーションが大規模で複雑であっても、完全に変更のない既存のアプリケーションの実行機能を提供します。

JavaSoft Java Plug-In を使用して Forms Developer アプリケーションを実行できますか？

JavaSoft Plug-In を使用して Forms Developer アプリケーションを配置することは保証されていません。したがって、これはサポートされる配置構成ではありません。現在、Oracle JInitiator によって提供される JRE には、JavaSoft によって提供される JRE ではまだ使用できない多くの拡張機能が組み込まれています。さらに、オラクル社は、オラクル社カスタマ・サポートを通して Oracle JInitiator を完全にサポートしています。

オラクル社ではネイティブ・ブラウザ配布をサポートする予定はありますか？

ネイティブ・ブラウザ・サポートを行う際の主な問題は、Forms Server が必要とする Java の同じバージョンおよび品質レベルをサポートするために、ブラウザ・ベンダーおよびプラットフォーム提供者に依存する必要があることです。オラクル社はこの依存があるため、お客さまが必要とする期間にネイティブ・ブラウザの配布を、配布オプションとして保証できませんでした。したがって、弊社は Oracle JInitiator をインターネットによるアプリケーション配布方法として完全に保証しています。これにより、Forms Server アプリケーションを利用するプラットフォームの安定性およびサポートが保証されます。

B.3.2 サポート

誰が Oracle JInitiator のサポートを提供するのですか？

オラクル社がオラクル社カスタマ・サポートを通して Oracle JInitiator を完全にサポートします。

Oracle JInitiator は Forms Developer Server のどのバージョンをサポートしていますか？

Oracle は、クライアント上で Oracle JInitiator を実行する Forms Server リリース 1.6 以降をサポートします。

Oracle JInitiator は Oracle Application でサポートされていますか？

はい。Oracle Application グループでは、Netscape Navigator 4.06 以降および Microsoft Internet Explorer 4.0 以降で Oracle Application を実行する場合、Oracle JInitiator の使用が保証されています。

B.3.3 インストール

Forms Developer アプリケーションを Web ブラウザで実行するために、クライアント上に何をインストールする必要がありますか？

ブラウザが Oracle JInitiator を必要とする HTML ページを最初に見つけたとき、Oracle JInitiator は Netscape Navigator および Microsoft Internet Explorer によって提供される標準ブラウザ拡張メカニズムを使用して、自身をクライアント・マシンに自動的にダウンロードできます。続いて、プラグインまたは ActiveX オブジェクトの追加に必要なメソッドを使用して、Oracle JInitiator は現在使用しているブラウザにインストールされます。

クライアントにダウンロードされる Oracle JInitiator はどの程度の大きさですか？

圧縮された Oracle JInitiator 配布版は約 8MB です。クライアントに完全インストールする場合は、約 10MB になります。

ユーザーが詳細を入力する必要のない、Oracle JInitiator のサイレント・インストールは実行できますか？

Oracle JInitiator では、ユーザーが InstallShield によって提供されるインストール・プロセスをアクティブに実行する必要がないサイレント・インストール・モードがサポートされています。サイレント・インストールを実行するには、Oracle JInitiator 配布版をマシンにダウンロードする必要があります。続いて、ダウンロードした実行ファイルの実行時に、コマンド・ラインからまたは Windows の「ファイル名を指定して実行」ダイアログから "-s -sm" を指定します。

たとえば、サイレント・インストールをコマンド・ラインから実行する場合、DOS シェルをオープンし、次のように入力します。

```
C:¥TEMP> jinit1179 -s -sm
```

Windows の「ファイル名を指定して実行」ダイアログを使用してサイレント・インストールを実行するには、**「スタート」→「ファイル名を指定して実行」**をクリックしてから、表示される「ファイル名を指定して実行」ダイアログで `jinit1179 -s -sm` と入力します。

ユーザーの介在が不必要な Oracle JInitiator インストールをセントラル・サーバーから実行できますか？

ホスト・オペレーティング・システムによって提供されるこの機能を使用すると、ユーザーが介在することなく、Oracle JInitiator を各クライアント・デスクトップにインストールできます。これには、システム管理者が各クライアント・マシンにアクセスし、サイレントで GUI を使用しない Oracle JInitiator インストール・オプションを実行することも含まれます。

Oracle JInitiator で、実行中のブラウザと同じプロキシ・サーバー構成などを使用することができますか？

Oracle JInitiator の操作は、Oracle JInitiator の「コントロール・パネル」を使用して制御されます。Oracle JInitiator の「コントロール・パネル」は、Oracle JInitiator のインストール時にインストールされます。「コントロール・パネル」には、**「スタート」→「プログラム」**メニューからアクセスできます。

Oracle JInitiator の「コントロール・パネル」を使用して、Oracle JInitiator が特定のプロキシ設定または起動ブラウザによって提供されるデフォルトを使用するように構成できます。「**Proxies**」タブを選択し、適切な設定を挿入します。

どのようにして Oracle JInitiator の新規バージョンをブラウザ・クライアントでダウンロードし、インストールするのですか？

Oracle JInitiator は使用ブラウザのタイプに応じて、Netscape プラグインまたは Microsoft ActiveX オブジェクトとして機能します。ブラウザは MIME タイプを使用して、HTML ページ・リクエストと必要なプラグインまたは ActiveX オブジェクトの間でマッピングを行います。各 Oracle JInitiator インストールには、特定の MIME タイプが対応付けられています。ブラウザが不明な MIME タイプが含まれた HTML ページをロードする場合、このページに必要なプラグインまたは ActiveX オブジェクトが含まれていないことをユーザーに通知します。続いて、これを取り出すためのダイアログ・ボックスが表示されます。

アプリケーションの HTML ページで指定された MIME タイプを以降のバージョンに変更することにより、ブラウザはその MIME タイプにとって有効なプラグインまたは ActiveX オブジェクトが含まれていないことを検出します。続いて、リクエストに完全に応えるために、新規ファイルをダウンロードするようプロンプトが表示されます。

例 :

HTML ページ HR.HTML を使用すると、HR アプリケーションを実行できます。HR.HTML ページは、MIME タイプ値を使用して Oracle JInitiator バージョン 1.1.5.21.1 を使用する必要があることをブラウザに示します。

Oracle JInitiator の以降のリリースを入手しサーバーにインストールしている場合、クライアント・ブラウザでは、新規バージョン・リリース情報が含まれている HR.HTML ファイル内のバージョンに関する行を変更することにより、新規バージョンを使用できます。

Netscape の "Plug-in Not Loaded" ダイアログで「取消し」ボタンをクリックしましたが、Oracle JInitiator をインストールするためのプロンプトが表示されません。どのようにしてプラグインをインストールすればよいですか？

Netscape では、Windows レジストリを使用して、インストール済みプラグインに関する情報が格納されています。"Plug-in Not Loaded" ダイアログが表示されると同時に、Netscape はプラグインが実際にインストールされているのかどうかにかかわらず、プラグインの詳細をこのレジストリに書き込みます。特定のプラグインを使用する必要があるページが見つかった場合、Netscape はレジストリの設定に従って、プラグインがインストールされていると判断します。したがって、"Plug-in Not Loaded" ダイアログが再び表示されることはありません。これを解決するには、Netscape で「プラグイン欠落」アイコンをクリックして、プラグインをロードします。これにより、Netscape で「Plug-In を入手」ダイアログ・ボックスが表示されます。

異なる MIME タイプを含む多くの HTML ページを持っています。Oracle JInitiator の最新リリースは、これらの旧 MIME タイプで実行できますか？

Netscape ブラウザでは現在、特定プラグインの指定 MIME タイプを格納するために使用する場合、256 文字の上限があります。Microsoft Internet Explorer では拡張可能ブラウザ・オブジェクト・アーキテクチャが使用されているため、この制限はありません。Oracle JInitiator はこの制限内で、できる限り多くの旧 MIME タイプに対して逆向きのサポートを提供します。

特定リリースでサポートされている MIME タイプの詳細は、Oracle JInitiator リリースの添付ドキュメントおよびリリース・ノートを参照してください。

Oracle JInitiator のすべてのバージョンで Forms Developer アプリケーションを実行できますか？

はい。Oracle は、Oracle JInitiator のすべてのインストール・バージョンで Forms Developer アプリケーションを実行できる、汎用 MIME タイプを提供しています。この MIME タイプ・アプリケーション x-jinit-applet は、Oracle JInitiator のすべてのバージョンで認識されます。常にこの MIME タイプを使用することで、ブラウザによって Oracle JInitiator の以降のバージョンをアップグレードできます。

B.3.4 Oracle JInitiator の操作

Forms アプレット・ウィンドウを同じ起動ブラウザ・ウィンドウ内で実行できますか？

Forms Server リリース 6*i* では、同じブラウザ・ウィンドウ内および新規ウィンドウ内での Forms アプレットの実行がサポートされています。これは構成変更可能なオプションであり、ベース HTML ファイルのパラメータとして設定されます。

現行ブラウザ・ページのナビゲートを終了した場合、実行中の Forms Developer アプリケーションはどうなりますか？

Oracle JInitiator には、実行中の Java アプリケーションをキャッシュに書き込み、現行ブラウザ・セッション中必要なときに取り出すことができる追加機能が含まれています。つまり、Forms アプリケーションの実行中に異なるページにナビゲートしてから Forms アプリ

ケーション・ページに戻る場合、実行中の Forms アプリケーションは元の状態のまま表示されます。

Oracle JInitiator を使用してカスタム開発 Java アプリケーションを実行できますか？

Oracle JInitiator では、Oracle 開発チームが拡張した標準 JavaSoft JVM が使用されています。したがって、カスタム Java アプリケーションを実行できます。ただし、Oracle は現在、Forms Developer、Oracle Enterprise Manager および Oracle Discoverer など Oracle Java ベース・アプリケーションの実行時のみ、Oracle JInitiator をサポートしています。Oracle では、カスタム Java アプリケーションを実行するために Oracle JInitiator を使用することはサポートしていません。

Oracle JInitiator と JavaSoft Java Plug-in は同じマシン上で共存できますか？

はい。これらは異なる MIME タイプを使用してプラグインを起動するので、同じブラウザ・インストールで共存できます。

同じブラウザ・インスタンス内で JavaSoft Java Plug-in と同時に使用された場合、Oracle JInitiator は正しく共存し動作しますか？

いいえ。動的ロード可能なライブラリはロードされ、JVM 動的ロード可能なライブラリは指定されるので、Oracle JRE および JavaSoft JRE は同じブラウザ・インスタンス内から同時に実行できません。つまり、JavaSoft Java Plug-in を使用しているブラウザ・ユーザーが、同じブラウザ・インスタンス内で Oracle JInitiator に切り替えることはできません。Oracle JInitiator および JavaSoft の Java Plug-in を使用する異なるアプリケーションを切り替える場合、ブラウザを停止し再起動する必要があります。

JavaSoft Java Plug-in および Oracle JInitiator を使用して、異なる JRE を使用するオプションがあります。Forms Developer アプリケーションを実行するために Oracle 保証 JRE を使用するように構成されている場合、JavaSoft Java Plug-in を使用できますか？

保証およびサポートされている組合せは、Oracle JInitiator と Oracle JRE のみです。JavaSoft 標準に準拠する Oracle JRE には、JavaSoft JRE に対するバグ修正が含まれています。これにより、Forms Developer アプリケーションは正しく実行されます。オラクル社は Oracle の拡張機能が JavaSoft に伝達され、標準 JRE に適用されるように、JavaSoft と密接な関係を築いています。しかし、改良された JavaSoft JRE がリリースされるのを待つことはできません。

次の図は、Oracle JInitiator の「コントロールパネル」および Java ランタイム環境値の正しい設定を示しています。

B.3.5 キャッシュ書込み

Oracle JInitiator は、アプリケーション実行時にダウンロードされた Java クラス・ファイルをキャッシュ書込みできますか？キャッシュ書込みできる場合、Java クラス・ファイルは一回のみダウンロードされ、アプリケーションが起動されるたびにダウンロードされないのですか？

はい。Oracle JInitiator では、Java アプリケーション実行時にダウンロードする JAR ファイル用の持続キャッシュ書込みメカニズムが提供されます。JAR ファイルは、Java アプリケー

ションによって使用される一連の Java クラス・ファイルを含む標準 Java アーカイブです。必要な各クラス・ファイルを複数回ダウンロードするのではなく、すべての必要なクラス・ファイルを 1 つの JAR ファイルに入れることにより、ダウンロードは一度ですみます。

Oracle JInitiator は JAR ファイルをクライアントにキャッシュ書込みすることにより、アプリケーションで必要になるたびに JAR ファイルをダウンロードする必要性が少なくなります。JAR ファイルが最初に必要になったとき、Web サーバーからダウンロードされ、ローカル・クライアント・マシンに保存されます。JAR ファイルが次に必要になったとき、Oracle JInitiator は JAR ファイルが格納されているかどうかを確認するためにキャッシュ・ディレクトリを調べます。格納されている場合、これをローカル・ディレクトリから使用し、ファイルを Web サーバーから再ダウンロードしません。これにより、ユーザーの多くの時間および一般に使用されるアプリケーションのネットワーク・トラフィックを節約できます。たとえば、アプリケーションが 2MB の JAR ファイルを使用しており、2MB のファイルを 5 秒でダウンロードできる高速イーサネット接続を使用する場合、アプリケーション起動時に 5 秒節約できます。2MB のファイルをダウンロードするのに 10 分を要する低速ダイアルアップ・ネットワークで実行する場合、アプリケーション起動時に 10 分節約できます。

Oracle JInitiator キャッシュ書込みテクノロジはどのようにして機能するのですか？

Oracle JInitiator では、ブラウザのセッションとは無関係に JAR ファイルをキャッシュに書き込みます。Oracle JInitiator は、ダウンロードされた JAR ファイルをローカル・クライアント・マシンに格納します。したがって、次に JAR ファイルが必要なときにダウンロードする必要がありません。

JAR ファイルが要求されたとき、Oracle JInitiator は JAR ファイルが前回要求され、ダウンロードされ、格納されていないかどうかを確認するために、キャッシュ・ディレクトリをチェックします。JAR ファイルが存在しない場合、Oracle JInitiator は JAR ファイルを Web サーバーからダウンロードし、次回使用するためにキャッシュに格納します。キャッシュ・ファイルには、Oracle JInitiator が JAR ファイル、および Web サーバーによって報告される要求ファイルの最終変更日付を一意に識別するための追加情報が格納されています。

キャッシュに JAR ファイルが存在する場合、格納された JAR ファイルが現行のものであるかどうかを確認するために、Web サーバーをチェックする必要があります。Oracle JInitiator はキャッシュ書込みされた JAR ファイルに含まれる最終変更日付を調べ、サーバー上の JAR ファイルが変更されているかどうかを Web サーバーに確認します（標準 HTTP 相互作用を使用）。Web サーバーは、サーバーに格納されたファイルの最終変更日付およびタイムスタンプを使用します。続いて、ステータス・コード 200 で新しいファイルを Oracle JInitiator に提供します。または、ステータス・コード 304 を戻します。これは、キャッシュ内のファイルが現行のものであることを示します。

キャッシュ書込みされた JAR ファイルが現行のものではない場合、新しいファイルがダウンロードされ、次回使用するためにキャッシュ・ディレクトリに格納されます。ファイルが現行のものである場合、Oracle JInitiator はこのファイルをキャッシュ・ディレクトリからロードし、前回使用されたことを示すために、キャッシュ書込みされたファイルのタイムスタンプを更新します。

キャッシュ書込みされた JAR ファイルはどこに格納されるのですか？

デフォルトでは、Oracle JInitiator はダウンロードされた JAR ファイルを Oracle JInitiator インストール・ディレクトリの下の jcache サブディレクトリに格納します。

なぜ、jcache ディレクトリには、キャッシュ書込みされた JAR ファイルの見慣れない名前が含まれているのですか？

Web サーバー上の各 JAR は URL（URL = codebase + JAR filename）によって識別されるので、Oracle JInitiator のキャッシュ書込みメカニズムではこれを使用して、JAR ファイルを一意に識別します。Windows オペレーティング・システムでは、完全な URL は有効なファイル名ではないので、Oracle JInitiator は単純なハッシング・アルゴリズムを使用して、これを有効なファイル名に変更してから、格納 JAR ファイル名として使用します。JAR ファイルのリクエストが行われた場合、Oracle JInitiator は完全な URL に対してハッシング・アルゴリズムを実行し、キャッシュ内に結果として生じるファイル名が存在するかどうかをチェックします。

JAR ファイル・キャッシュ書込みはどのようにサーバー・ロード・バランシングを処理するのですか？

前述したように、キャッシュ内の JAR ファイルは、取り出された URL に基づいて識別されます。したがって、異なるサーバー上にある同じ JAR ファイルは、各サーバーからダウンロードされます。セキュリティおよびアプリケーションの整合性を保証するために、これは意識的に行われます。JAR ファイルが名前のみを使用してキャッシュ書込みされている場合、不正なアプリケーションが他のアプリケーションの JAR ファイルを置換することがあります。オリジナルのアプリケーションが実行されている場合、Java クラス・ファイルは異なります。また、JAR ファイルは一意の名前を持つことが保証されていないので、JAR ファイルが衝突することがあります。これは、2 つの異なるアプリケーションが同じ JAR ファイル名を使用しており、JAR ファイルからの異なるクラス・ファイルが必要な場合に起こります。

Forms Developer アプリケーションを実行するたびに、キャッシュ書込みされた JAR ファイルのタイムスタンプが更新されているようです。これは正常ですか？そのたびに、ファイルがダウンロードされているということですか？

いいえ。Oracle JInitiator では、構成変更可能なキャッシュ最大サイズがサポートされています。キャッシュ書込みされた JAR ファイルが使用されるたびに、Oracle JInitiator はキャッシュ書込みされたファイルが前回使用された日時を示すために、タイムスタンプを更新します。

キャッシュ・サイズが、最大キャッシュ・サイズを維持するためにファイル削除が必要になる程度まで大きくなった場合、Oracle JInitiator はキャッシュ・ファイルのタイムスタンプを使用して、最も以前に使用されたファイルを判断し、そのファイルを削除します。

キャッシュが正しく機能していること、および JAR ファイルが常にダウンロードされているわけではないことがどうしたらわかるのですか？

Oracle JInitiator が必要なファイルをダウンロードする必要がある場合、Forms Developer アプリケーションを実行するように構成された Web サーバーを使用します。最新の Web サーバーでは、ダウンロードされたファイル、ダウンロードしたユーザー、およびダウンロード日時を追跡できるログ・ファイルの使用がサポートされています。Web サーバーのログ・

ファイルでは、発生したトランザクションを記述するために標準フォーマットが使用されています。このログ・フォーマットには、リクエスト項目名およびリクエストの結果が含まれています。リクエストの結果は、一連の標準 HTTP ステータス・コードを使用して示されます。

JAR ファイルがクライアントにダウンロードされている場合、ログ・ファイルには、要求された JAR ファイル名および HTTP ステータス・コード 200 が含まれます。JAR ファイルのタイムスタンプがキャッシュ書込みされたファイルのタイムスタンプ以前のものであるため、JAR ファイルがダウンロードされていない場合、ログ・ファイルには、要求された JAR ファイル名および HTTP ステータス・コード 304 が含まれます。

次の例では、キャッシュ内の JAR ファイルが現行のものではないので、Web サーバーからダウンロードする必要がある場合、標準 NCSA ログ・フォーマットを使用してログ・ファイル内で作成されたエントリを示します。

```
ferret.us.oracle.com - - [19/Feb/1999:17:40:12 -0800] "GET /forms_
java/f60all.jar  HTTP/1.0" 200 -
```

次の例では、キャッシュ内の JAR ファイルが現行のものであるので、Web サーバーからダウンロードしない場合、標準 NCSA ログ・フォーマットを使用してログ・ファイル内で作成されたエントリを示します。

```
ferret.us.oracle.com - - [19/Feb/1999:17:42:29 -0800] "GET /forms_
java/f60all.jar  HTTP/1.0" 304 -
```

JAR ファイルのダウンロード時、jcache ディレクトリに .JCX ファイルが作成されるようです。このファイルは何ですか？

JAR ファイルのダウンロード時、JAR ファイルのテンポラリ・コピーがファイル・システムに書き込まれます。このテンポラリ・コピーは、.jcx というファイル拡張子で識別されます。ダウンロードが正常に終了した場合、.jcx ファイルは .jc ファイルになります。ダウンロードまたは接続が中断された場合、操作は完了しないで、テンポラリ・ファイルの拡張子は .jcx のままです。Oracle JInitiator は .jcx という拡張子のファイルは無効なので、ロードしません。

キャッシュ書込みが正常に機能することを確認していますが、アプリケーションの起動に時間がかかりすぎます。これはなぜですか？

Oracle JInitiator によって提供される JAR ファイル・キャッシュ書込みは、システム上の Java のスピードを上げるような技法を使用しません。実行することは、各アプリケーションの起動に必要な JAR ファイルをダウンロードする時間を節約することです。JAR ファイルの Zip 解凍、包含クラスのメモリーへのロード、および改ざんされていないことを保証するための認証操作によって、起動時間は非常に長くなります。実際に、超高速ネットワークでは、JAR ファイルのダウンロードに要する時間は、Java クラスのメモリーへのロードおよび認証に要する時間よりも短くなります。つまり、キャッシュ書込みでは、アプリケーション起動に要する時間をほとんど節約できません。低速ネットワークでは、JAR ファイルのダウンロードに要する時間は、起動時間において比例して長くなります。したがって、JAR ファイルのキャッシュ書込みはさらに重要になります。

C

AppletViewer

C.1 概要

この付録では、Oracle JInitiator のかわりに使用する AppletViewer について説明します。AppletViewer は、JDK コンポーネントであり、クライアント・マシンが Forms Server 上で実行されるアプリケーションを表示するために使用する Oracle サポート製品です。アップグレード・バージョンは、Forms Developer Web サイトからダウンロードできます。

注意 : AppletViewer は Windows 95 および Windows NT 4.0 でのみサポートされています。

C.2 AppletViewer でのアプリケーション実行

AppletViewer でアプリケーションを実行するには、次のステップを実行する必要があります。

- AppletViewer を使用してアプリケーションを実行する準備を行います。
- clientBrowser パラメータをベース HTML ファイルに追加します。
- clientBrowser パラメータを設定します。

AppletViewer でアプリケーションを実行している場合、AppletViewer は URL の表示リクエスト（たとえば、web.showDocument および RUN_PRODUCT）を無視します。この場合、この章の C.3.1 項「シグネチャを登録することによる Forms アプレットの信頼」で後述する方法で、Forms アプレットを信頼するためのプロセスを実行する必要があります。

C.2.1 AppletViewer を使用したアプリケーション実行準備

AppletViewer 内でアプリケーション実行の準備を行うには、AppletViewer をダウンロード可能にして、ユーザーに AppletViewer をクライアント・マシンにインストールする必要があることを通知します。次のことを実行します。

1. JDK_DOWNLOAD.HTM をカスタマイズします。

JDK_DOWNLOAD.HTM は、ユーザーが AppletViewer をダウンロードできるテンプレート HTML ファイルです。

2. JDK.EXE を Web サーバーにコピーします。

   JDK.EXE は、JDK_DOWNLOAD.HTM 内で指定された位置にコピーする必要があります。

3. JDK_DOWNLOAD.HTM を Web サーバーにコピーします。

   JDK_DOWNLOAD.HTM は、JDK_DOWNLOAD.HTM 内で指定された位置にコピーする必要があります。

C.2.2 clientBrowser パラメータのベース HTML ファイルへの追加

clientBrowser パラメータを使用するには、指定アプリケーションを実行するシステム・コールを発行するためのセキュリティ権限を持つ必要があります。通常、Java クラス・ファイルをロードする場合、Forms アプレットは信頼されないので、このようなシステム・コールは発行できません。ただし、Forms アプレットが信頼される場合、これらのコールを発行できます。次のいずれかを満たす場合、Forms アプレットは信頼されていると判断されます。

- C.3.1 項「シグネチャを登録することによる Forms アプレットの信頼」で記述したように、Forms アプレット・シグネチャがクライアント・マシンに「登録されている」。
- Forms Java クラス・ファイルがクライアント・システムにローカルにインストールされており、C.3.2 項「Forms Java クラス・ファイルをローカルにインストールすることによる Forms アプレットの信頼」で記述したように CLASSPATH 環境変数が設定されている。

これらの HTML ファイルの例では、マシンにシグネチャを登録することで Forms アプレットを信頼していると仮定しています。また、Forms Java クラス・ファイルをローカルにインストールすることで Forms アプレットを信頼した場合、F60ALL.JAR ファイルをダウンロードする必要はありません。したがって、ARCHIVE="/.../f60all.jar" アプレット・タグを HTML ファイルから削除します。

C.2.3 clientBrowser パラメータの設定

clientBrowser パラメータを設定するには、次のいずれかを実行します。

- clientBrowser パラメータを HTML ファイルに追加します。
- clientBrowser パラメータを HTML ファイルに追加し、各クライアントに JDK_SETUP.BAT ファイルを変更させます。

clientBrowser パラメータを HTML ファイルに追加します。

このオプションでは、ブラウザの表示パスが HTML ファイルにハードコードされているため、すべてのクライアントがブラウザ実行ファイルを同じ表示ディレクトリにインストールしていると仮定しています。例 :

```
<APPLET CODEBASE="/forms60code/"
     CODE="oracle.forms.engine.Main"
     ARCHIVE="/forms60code/f60all.jar"
     HEIGHT=480
     WIDTH=640>
     <PARAM NAME="serverArgs" VALUE="module=start.fmx userid=scott/tiger">
     <PARAM NAME="clientBrowser"
     VALUE="c:¥programfiles¥netscape¥communicator¥program¥netscape.exe">
</APPLET>
```

clientBrowser パラメータを HTML ファイルに追加し、各クライアントに JDK_SETUP.BAT ファイルを変更させます。

このオプションは、クライアントがブラウザ実行ファイルを異なる表示ディレクトリにインストールしている可能性がある場合に最適です。ただし、すべてのクライアントが同じブラウザを使用していると仮定しています。HTML ファイルの一例を次に示します。

```
<APPLET CODEBASE="/forms60code/"
     CODE="oracle.forms.engine.Main"
     ARCHIVE="/forms60code/f60all.jar"
     HEIGHT=480
     WIDTH=640>
     <PARAM NAME="serverArgs" VALUE="module=start.fmx userid=scott/tiger">
     <PARAM NAME="clientBrowser" VALUE="netscape">
</APPLET>
```

次に、JDK_SETUP.BAT の一例を示します。

```
     SET CLASSPATH=C:¥ORANT¥JDK1.1¥JDK¥LIB¥CLASSES.ZIP
     PATH C:¥PROGRAM FILES¥NETSCAPE¥COMMUNICATOR¥PROGRAM;
     C:¥ORANT¥JDK1.1¥JDK¥BIN;%PATH%
```

C.3 Forms アプレット・シグネチャの登録

クライアント・マシンはシグネチャを使用して、有効かつ信頼されたエンティティ（*署名者*）からファイルがダウンロードされていることを確認できます。これにより、クライアント・マシンは不正行為または Java アーカイブ（JAR）ファイルの誤動作から自身を保護できます。クライアントが JAR ファイルを妥当性チェックするには、クライアント・マシンにそのファイルのシグネチャが登録されていることが必要です。Javakey は、JAR ファイルのデジタル・シグネチャを生成するための Sun Microsystems のコマンド・ライン・ツールです。

Forms アプレット自体は署名済み JAR ファイルです。Forms アプレット・シグネチャを登録するには 2 つのオプションがあります。次のいずれかを選択します。

- 弊社が提供する Forms アプレット・シグネチャを使用して、シグネチャをクライアント・マシンに登録します。
- ユーザー固有のシグネチャを使用して Forms アプレットを再署名し、そのシグネチャをクライアント・マシンに登録します。このメソッドを選択する場合、JAR ファイルの作成および署名の詳細は http://java.sun.com/security/usingJavakey.html を参照してください。

C.3.1 シグネチャを登録することによる Forms アプレットの信頼

シグネチャを登録することにより、Forms アプレットを信頼するには、次の手順を実行します。

1. Forms Developer 認証をクライアント・マシン上の ¥ORACLE_HOME¥FORMS60¥JAVA にコピーします。

   この認証は Dev.x509 という名前のファイルです。このファイルは、サーバー上の ¥ORACLE_HOME¥FORMS60¥JAVA にあります。

2. DOS コマンド・プロンプトをオープンし、¥ORACLE_HOME¥FORMS60¥JAVA にナビゲートします。
3. 次のように入力します。javakey -c Developer true

   このコマンドは、認証提供者の正確な名前を使用して、クライアントの認証データベース上で AppletViewer の信頼された認証を作成します。

4. [Enter] キーを押します。
5. javakey -ic Developer Dev.x509 と入力します。

   このコマンドは、Dev.x509 認証をクライアントの JDK 認証データベースにインポートし、この認証をステップ 3 で作成された信頼された認証に関連付けます。

6. [Enter] キーを押します。

C.3.2 Forms Java クラス・ファイルをローカルにインストールすることによる Forms アプレットの信頼

Java クラス・ファイルをローカルにインストールすることにより Forms アプレットを信頼するには、次の手順を実行します。

1. ¥ORACLE_HOME¥FORMS60¥JAVA ディレクトリをクライアント・マシン上の新規ディレクトリにコピーします。

   このディレクトリは正確にコピーし、ディレクトリ構造は絶対に変更しないでください。

2. 次のようにして、ORACLE_HOME ディレクトリの JDK_SETUP.BAT を変更します。
   a. テキスト・エディタで JDK_SETUP.BAT をオープンします。
   b. 新規ディレクトリを参照するために、CLASSPATH 環境変数を変更します。
   c. 変更を JDK_SETUP.BAT に保存します。

C.4 ユーザーへの指示

アプリケーションを AppletViewer 内から実行するには、次のステップを実行します。

- AppletViewer をインストールします。
- AppletViewer を実行します。
- Web ブラウザを AppletViewer 内から起動します。

C.4.1 AppletViewer のインストール

AppletViewer をインストールするには、Oracle Installer を使用して JDK AppletViewer をインストールします。

1. 起動している Windows アプリケーションをすべて終了します。
2. タスクバーから、「**スタート**」→「*ファイル名を指定して実行*」を選択します。
3. 「ファイル名を指定して実行」ダイアログに次のように入力します（「D:」は実際の CD-ROM ドライブの文字に置き換えます）。`D:¥setup.exe`。続いて「**OK**」をクリックします。
4. 「Oracle インストール設定」ダイアログ・ボックスで、会社名および ORACLE_HOME ディレクトリのデフォルト値をチェックします。
5. **「Oracle Forms Server」**をクリックします。
6. 「**カスタム**」をクリックします。
7. 「使用可能な製品」リストから、「**JDK AppletViewer**」を選択します。
8. **「インストール」**をクリックします。

C.4.2 AppletViewer の実行

AppletViewer を実行するには、次の手順を実行します。

1. DOS コマンドで、AppletViewer 実行ファイル（appletviewer.exe）に移動します。

2. ホスト名、HTML ファイル仮想ディレクトリおよび HTML ファイルを指定して、AppletViewer 実行ファイルを実行します。

   たとえば、次のように入力します。appletviewer http://myhost.com/web_html/start.html

3. [Enter] キーを押します。

C.4.3 AppletViewer 内からの Web ブラウザ起動

Web ブラウザを AppletViewer 内から起動するには、次の手順を実行します。

1. 次の 2 つのメソッドのいずれかを使用して、Forms を信頼します。
   - Forms アプレット・シグネチャを登録する。
   - Forms Java クラス・ファイルをローカルにインストールする。
2. clientBrowser パラメータをベース HTML ファイルに追加します。

D

Oracle Installer リファレンス

D.1 概要

Oracle では、Forms Server のインストール作業が単純化されています。急いで行いたい構成の決定が自動的に行われます。

この付録では、Forms Server のインストール中に提供される多くのインストール・オプションについて説明します。また、Web への配置に必要なソフトウェアのインストールに関する推奨事項についても説明します。

この付録には、次の項目に関する情報が含まれています。

- デフォルト・インストール・オプション
- 構成指示

D.2 デフォルト・インストール・オプション

次の項では、Forms Server のインストール中に表示される Oracle Installer オプションを一覧表示し、説明します。

D.2.1 Oracle Tools のインストール・オプション

どの製品をインストールしますか？

- Oracle Forms Developer
- Oracle Forms Server
- Oracle Reports Developer
- Oracle Reports Server

推奨事項 : Oracle Forms Server を選択します。

Oracle Forms Developer を使用して、フォーム・アプリケーションを作成し、実行環境でテストを行うことが出来ます。それには、Forms Server の限定バージョンが含まれます。

Oracle Forms Server を使用すると、3 層環境でアプリケーションを使うことができます。

D.2.2 Forms Server インストール・オプション

インストールの種類を選択してください。

- 標準
- カスタム

推奨事項 : Web 配置用のコンポーネントをインストールするには、「**標準**」を選択します。

「**標準**」を選択すると、適切で実行可能な製品コンポーネントが自動的にインストールされます。製品をはじめてお使いになる場合は、これを選択してください。

「**カスタム**」は、複雑なインストール方法で、製品知識を必要とします。高度な制御が必要で、製品コンポーネントをよくご存知の場合のみ、このオプションを選択してください。

D.2.3 実行環境のインストール・オプション

実行する実行環境のインストールの種類を選択してください。

- クライアント・サーバー配置用ランタイム
- Web 配置用 Forms Server

推奨事項 :「**Web 配置用 Forms Server**」を選択します。

クライアント・サーバー配置用ランタイムは、通常クライアント・サーバーアーキテクチャで使用されるランタイムツールをインストールします。マシンは Windows クライアント・マシンとして機能します。

インストールされる製品は次のとおりです。

1) Oracle Forms Runtime

2) Oracle Graphics Runtime & Chart Wizard

3) Oracle Net8 Client

4) SQL*Plus

5) リリース・ノート

6) Information Navigator

7) Oracle Installer

「**Web 配置用 Forms Server**」は、１台以上のマシン上で、Web アプリケーションとして Oracle Developer アプリケーションを実行するために必要なコンポーネントをインストールし、構成します。今回のインストレーションを１台のマシン構成にするか、ロード・バランスを使用する複数マシン構成の一部にするかの選択肢が表示されます。

D.2.4 Forms Server のインストール

サーバーの種類を選択してください。

- 単一マシン構成
- 複数マシン構成の一部

推奨事項 :「**単一マシン構成**」を選択します。

Forms Server を正しくインストールし、構成するには、1 台のマシンのみにインストールするのか、複数のマシン上にインストールするのかを認識する必要があります。

単一マシン構成 : Forms Server を 1 台のマシンのみにインストールする場合に選択します。

複数マシン構成の一部 : スケーラビリティを向上させるために Forms Server を複数マシン構成の一部としてインストールする場合に選択します。プライマリ・ノードとセカンダリ・ノードについて追加情報が表示されます。

D.2.5 複数マシン構成

このマシンのノードの種類を選択してください。

- プライマリ・ノード
- セカンダリ・ノード

推奨事項 : 最初にマシンにインストールする場合、「**プライマリ・ノード**」を選択します。以降のすべてのインストールについては、「**セカンダリ・ノード**」をクリックします。

複数のマシンがある場合、1 台のマシンを「プライマリ・ノード」として実行します。このマシンでは Web リスナーが実行され、Forms アプリケーションを実行するためにすべての URL リクエストが受信され処理されます。各リクエストは、アプリケーションが実際に実行されるマシンに転送されます。

これは「セカンダリ・ノード」または、Forms Developer ランタイムが利用可能な場合の「プライマリ・ノード」自身で処理されます。

プライマリ・ノード : このマシンで Web リスナーを実行し、Forms Developer アプリケーションを実行するための Web URL リクエストの受信と処理を行う場合、このオプションを選択します。

セカンダリ・ノード : プライマリ・ノードの指示にしたがって、このマシンで Forms Developer アプリケーションを実行する場合、このオプションを選択します。

各オプションでインストールされるコンポーネントは次のとおりです。

プライマリ・ノード :

1) Forms Web CGI

2) Forms Java support files

3) Oracle Net8 Client

5) ダウンロード用 JInitiator

5) Load Balancer Server

6) セカンダリ・ノードのすべてのコンポーネント（オプション）

セカンダリ・ノード :

1) Forms Server および Web ランタイム

2) Load Balancer Client

3) Oracle Net8 Client

4) Oracle Installer

5) Release Notes

6) Graphics Chart Wizard

7) Graphics Runtime

D.2.6 Forms サーバー : プライマリ・ノード

Forms Developer アプリケーションをどこで実行するか選択してください。

- セカンダリ・ノードおよびこのプライマリ・ノード上
- セカンダリ・ノード上のみ

推奨事項 :「**セカンダリ・ノードおよびこのプライマリ・ノード上**」を選択します。

Forms アプリケーションをこのプライマリ・ノード上で実際に実行する必要はありません。これらを複数マシン (セカンダリ・ノード) 構成の他のマシン上でのみ実行するように選択できます。

アプリケーションを実行する場所で利用可能なオプション：

セカンダリ・ノードおよびこのプライマリ・ノード上 : このプライマリ・ノード上で Oracle Forms アプリケーションを実行する場合、このオプションを選択します。

この場合、Forms Server および Load Balancer Client がその他のプライマリ・ノードのコンポーネントとともに、このノード上にインストールされ、構成されます。

セカンダリ・ノード上のみ : このプライマリ・ノード上でアプリケーションを実行しない場合、このオプションを選択します。このマシンが頻繁に使用される Web サーバーである場合、通常このオプションを選択します。この場合、Forms Server および Load Balancer Client は、このノード上にインストールされません。

D.2.7 Web リスナー

Oracle WebDB Listener をインストールし、構成することができます。また、CGI をサポートする他の Web リスナーを構成し、使用することもできます。

- Oracle WebDB Listener の使用
- 他の Web リスナーの使用

推奨事項 : 「**Oracle WebDB Listener の使用**」を選択します。

Oracle Forms アプリケーションを Web 上で実行するには、Web リスナー（HTTP デーモン）が必要です。Oracle Forms Server では、CGI をサポートする軽量なリスナーである Oracle WebDB Listener が提供されるため便利です。また、CGI (Common Gateway Interface) がサポートする他の Web リスナーを使用することもできます。

使用するリスナーの利用可能なオプション :

Oracle WebDB Listener の使用 WebDB Listener をインストールし、構成する場合、このオプションを選択します。

他の Web リスナーの使用 すでに他の Web リスナーをインストール済みで、これを使用する場合、または Oracle Application Server を使用している場合、このオプションを選択します。インストール完了後、Oracle Forms Server で使用できるように、Web リスナーに仮想パスを構成する必要があります。インストール中に、次に実行する指示が生成されます。

D.2.8 WebDB Listener

Data Access Descriptor（DAD）構成情報を指定します。

ホスト名 : [ホスト名 . 会社名 .com]

WebDB Listener Port #: [80]

推奨事項 : デフォルトを使用します。

WebDB Listener の構成情報を入力します。

ホスト名 : Listener をインストールするマシン名を入力します。DOS コマンドプロンプトで、「hostname」と入力すると、ホスト名が分かります。完全なドメイン名を入力するようにしてください。例えば、「ホスト名 . 会社名 .com」のように入力します。

WebDB Listener Port #: WebDB Listener が、データベースと Web ブラウザの間のデータ送信に使用するポート番号を入力します。デフォルトのポート番号は 80 です。ほかのアプリケーションが 80 を使用している場合は、別のポート番号を選択してください。

D.2.9 Forms Server パラメータ

Forms Server のパラメータ値を入力してください。

Forms Server ポート : [9000]

プロトコル :

- ソケット
- HTTP

推奨事項 : デフォルトを使用します。

Forms Server のスタートアップ・パラメータ値をそのまま使用するか、変更します。

Forms Server ポート : Forms Server がフォーム実行のリクエストを受信する TCP/IP ポート番号を入力します。デフォルト値が他のプログラムによってすでに使用されている場合のみ、その値を変更する必要があります。

プロトコル : Forms Runtime エンジンと（ユーザーの Web ブラウザで実行される）Forms Java アプレット間の通信に使用されるメソッドです。通信がファイアウォールを越える必要がある場合のみ、この値を HTTP に変更してください。たとえば、このマシンがファイアウォールの内側にあり、Forms アプリケーションをファイアウォールの外側のユーザーに利用させる必要がある場合、HTTP を選択します。

D.2.10 Load Balancer Server パラメータ

Forms Load Balancer Server によって使用されるポート番号を入力してください。

データ・ポート : [9010]

リクエスト・ポート : [9020]

推奨事項 : デフォルトを使用します。

Load Balancer Server のスタートアップ・パラメータ値をそのまま使用するか、変更します。これらは TCP/IP ポート番号です。これらの番号のいずれかが、他のプログラムですでに使用されている場合のみ、変更する必要があります。

データ・ポート :（セカンダリ・ノードで実行される）Load Balancer Client のプロセスからのロード・データを受信する TCP/IP ポート番号を入力します。

リクエスト・ポート : Forms Web CGI（またはカートリッジ）によって作成される「負荷が最小のホスト」のリクエストを受信する TCP/IP ポート番号を入力します。

D.2.11 Load Balancer Client パラメータ

プライマリ・ノードの完全なホスト名、およびそのノード上で実行されている Load Balancer Server のデータ・ポート番号を入力してください。

データ・ホスト : [ ]

データ・ポート : [9010]

推奨事項 : デフォルトを使用します。

Load Balancer Client のスタートアップ・パラメータ値をそのまま使用するか、変更します。これらは Load Balancer Server への接続方法を指定します。

データ・ホスト： この複数マシン構成のプライマリ・ノード（Load Balancer Server が実行されるマシン）の完全ホスト名を入力します。

データ・ポート： 負荷データを受信する Load Balancer Server の TCP/IP ポート番号を入力します。

D.3 構成指示

この項では、特定のコンポーネントのインストール・プロセス中に生成されるメッセージについて説明します。これらのメッセージは、構成指示ファイルに書き込まれます。インストール済みの製品を使用する前に、特定の構成ステップを実行する必要があります。これらのステップの多くは自動的に実行されます。ただし、一部のステップは手動で実行する必要があります。

インストーラは詳細を記述したファイルを生成します。これは %ORACLE_HOME%¥orainst¥dev6iconfig.txt に入っています。このファイルの内容を見て、要求されている処置を行います。

このファイルには、インストール・プロセスによって自動的に実行される構成ステップ（[情報] とマークされている）、および手動で実行する必要がある処置（[アクション] とマークされている）の詳細が記述されています。

Oracle WebDB Listener メッセージ

[アクション] Oracle WebDB Listener を使用しており、仮想パスが自動的に作成されている状態でのメッセージである場合、変更が反映されるようにリスナーを再起動する必要があります（Windows の「コントロール・パネル」の「サービス」ツールから実行できます）。

仮想パス・メッセージ（webdb）

[情報] 仮想パス < 名 > は、WebDB Listener を使用して、表示パス <physical_path> にマッピングすることにより作成されます。

仮想パス・メッセージ（webdb 以外）

[アクション] Web リスナーを使用して、次の仮想パスを構成してください。表示ディレクトリ <dir> の <vpname>

サービス起動メッセージ（NT のみ）

[情報] NT サービス <servicename> が作成され、<componentname> のために起動されています。

プロセス起動メッセージ（UNIX のみ）

[アクション] Forms Developer プロセス起動スクリプト <scriptfilename> に、<componentname> を起動するためのコマンドが書き込まれています。

Reports Server TNS メッセージ

[情報] Net8 <tnsnames.ora> ファイルに、Reports Server の次の TNS 名が定義されています。<TNSname>

Reports サービス・メッセージ（NT のみ）

[アクション] Reports Multitier Server 用に、NT サービス Oracle Reports Server [<servername>] が作成されています。これは、このマシン上でレポートを実行する前に、プリンタにアクセスできる NT ユーザーが起動する必要があります。

Reports CGI メッセージ

[アクション] Oracle Reports Web CGI およびカートリッジ : 他のマシン上でレポートを実行する場合、これらのマシン上で実行される Reports Server を識別するために、Net8 "tnsnames.ora" ファイルにエントリを追加する必要があります。

プライマリ・ノード・インストール・メッセージ

[アクション] このマシンを複数マシン構成のプライマリ・ノードにすることを要求しました。Reports Server クラスタを使用して Oracle Reports レポートを異なるマシンで実行する場合、CLUSTERCONFIG パラメータを Reports Server の構成ファイルに追加する必要があります。詳細は Reports Server のドキュメントを参照してください。

テスト・フォーム・メッセージ

[情報] 標準テスト・フォームまたは選択したフォームを実行するためのインターネット・ショートカットが、Forms Developer プログラム・グループ内に自動的に作成されています。

テスト・レポート・メッセージ

[情報] 標準テスト・レポートまたは選択したレポートを実行するためのインターネット・ショートカットが、Forms Developer プログラム・グループ内に自動的に作成されています。

E

カートリッジ・サポートを使用した Forms Server の構成

E.1 概要

Forms Server をインストール済みで、Forms カートリッジとともに Oracle Application Server（OAS）を使用している場合、この付録の情報を使用して環境を構成してください。

この付録には、次の項が含まれています。

- CGI とカートリッジの実装
- OAS Web サーバーの構成
- ベース HTML ファイルの構成
- Forms Server の構成
- カートリッジベースのロード・バランスの構成

注意： E.2 項「CGI とカートリッジの実装」で記述したように、Forms CGI の実装は、カートリッジの実装よりも好まれるメソッドです。CGI を使用した Forms Server の構成の詳細は第 5 章「Forms Server の構成」を参照してください。

E.2 CGI とカートリッジの実装

Forms Developer アプリケーションは次の 2 つの技術を使用して、Web サーバーで配布できます。

- Common Gateway Interface (CGI)
- カートリッジの実装

どちらのメソッドでも、ロード・バランスは使用できます。どちらのメソッドでも、Forms アプレットを起動する HTML ページを動的に生成するために、中間層に特別なエージェントが存在します。Web サーバー（Oracle Application Server 以外）が使用できる場合、CGI の実装を選択できます。Oracle Application Server を Web サーバーとしてすでに使用してい

る場合、カートリッジの実装を選択できます。最適な実装の決定は、個々の必要性および環境に依存します。

E.2.1 Common Gateway Interface (CGI)

CGI は Web の HTTP プロトコルのコンポーネントです。これは、Forms アプレットを起動する HTML ページを動的に生成するための、プラットフォームに依存しない標準の方法です。

CGI の実装を選択することには、次のような最低 2 つの大きな利点があります。

- **オープン性。**すべての Web サーバーが CGI をサポートしています。これが最も一般的な実装です。
- **スピード。**CGI およびカートリッジの実装では、CGI の方が高速です。

E.2.2 カートリッジの実装

Oracle Application Server（OAS）を Web サーバーとして使用する場合、カートリッジを実装できます。CGI とは異なり、これは Forms アプレットを起動する HTML ページを動的に生成するための独自の（Oracle ベースの）メソッドです。

Forms アプリケーションを Web 上で起動および実行するには、URL にアクセスするため、Java が使用可能な Web ブラウザを使用します。

カートリッジの実装では、初期 HTML ファイルが Forms カートリッジによって動的に作成されるので、共通パラメータを含む汎用アプリケーション・カートリッジを作成した後、各アプリケーションでそれを再利用できます。カートリッジを他のアプリケーションで使用する場合、新規 HTML ファイルを作成するのではなく、異なるパラメータ値を指定するように、そのアプリケーションの URL を変更します。

E.3 OAS Web サーバーの構成

Oracle Application Server（OAS）を Web リスナーとして使用する場合、Forms Developer をアプリケーション・カートリッジとして配布する必要があります。ただし、Oracle Application Server（OAS）は中間層アプリケーション・サーバーにインストールする必要があります。OAS には、カートリッジのフレームワークを提供し、これらへのクライアント接続を管理するための、Web Request Broker コンポーネントが含まれています。

OAS を構成するには、次の項の作業を実行する必要があります。

- OAS Web サーバーの起動
- リスナーおよび仮想ディレクトリの作成
- Forms Web カートリッジの作成

E.3.1 OAS Web サーバーの起動

Oracle Application Server（OAS）を起動するには、次の手順を実行します。

1. （UNIX のみ）ORACLE_HOME 環境変数を設定します。たとえば、次のように入力します。`setenv %ORACLE_HOME%/private/ora_home`
2. プロンプトに対して、次のように入力します。

```
owsctl start
```

ブラウザを起動するには、次の手順を実行します。

1. プロンプトが表示されたら、ブラウザを起動します。
2. 該当する URL（http://< マシンのフルネームまたは IP アドレス >:<Web Server Admin リスナーのポート番号 >）にナビゲートすることにより、Web サーバーにアクセスします。

   たとえば、ブラウザで次の URL をポイントします。http://myserver.com:8888

E.3.2 リスナーおよび仮想ディレクトリの作成

リスナーおよび仮想ディレクトリを作成するには、次の手順を実行します。

1. Oracle Application Server（OAS）の Welcome ページ（http://myserver.com:8888 など）で、「**OAS Manager**」をクリックします。
2. OAS Manager のナビゲーション・ツリーで、Web サイト・アイコンの横の「+」アイコンをクリックします。
3. 「**HTTP リスナー**」アイコンをクリックします。
4. リスナーの追加フォームを表示するには、「**追加**」アイコンをクリックします。
5. リスナーの追加フォームで、次の手順を実行します。
   a. 「ノードの選択」フィールドで、マシン名を指定します。
   b. 「**適用**」をクリックします。Oracle Application Server（OAS）は、追加フォームを表示します。
6. 追加フォームで、次の手順を実行します。
   a. 「リスナー名」フィールドで、新規リスナー名を指定します。リスナー名は一意である必要があり、英数字で 6 文字以内です。
   b. 「ポート番号」フィールドに、Web リスナーが接続を受け付ける TCP/IP ポートを指定します。すでに他のプログラムに割り当てられているポート番号を除いた、1 ～ 65535 の番号を選択できます。ポート 1 ～ 1023 にアクセスするには、ルート権限で Web リスナー・プロセスを実行する必要があります（「ユーザー ID」および「グループ ID」のフィールドを参照してください）。

c. 「**送信**」をクリックします。

d. 確認フォームで、「**OK**」をクリックします。

7. 次の手順を実行することにより、ディレクトリ・フォームを表示します。

a. ナビゲーション・ツリーを更新するには、［Shift］キーを押し、「**再読み込み**」をクリックします。

b. ノードを拡張するために、HTTP リスナーの横の「+」アイコンをクリックします。

c. ノードを拡張するために、新規リスナーの横の「+」アイコンをクリックします。

d. ディレクトリ・フォームを表示するために、「ディレクトリ」アイコンをクリックします。

8. 仮想ディレクトリを作成するには、ディレクトリ・フォームを使用します。

仮想ディレクトリ	物理ディレクトリ	説明
/forms60java	%ORACLE_HOME%¥forms60¥java	Forms .jar ファイル
/dev60html	%ORACLE_HOME%¥tools¥web60¥html	Forms を実行する初期の HTML ファイル
/dev60cgi	%ORACLE_HOME%¥tools¥web60¥cgi	CGI 実行可能ファイル
/jinitiator	%ORACLE_HOME%¥jinit¥doc	JInitiator（ダウンロード用）
/dev60temp	%ORACLE_HOME%¥tools¥web60¥temp	Forms テンポラリ・ファイル

a. 「ファイルシステム・ディレクトリ」フィールドで、表示ディレクトリ・パスを指定します。

b. 「フラグ」フィールドで、フラグ値を指定します。

c. 「仮想ディレクトリ」フィールドで、仮想ディレクトリを指定します。

d. 「**適用**」をクリックします。

e. 確認フォームで、「**OK**」をクリックします。

E.3.3 Forms Web カートリッジの作成

Forms Web カートリッジを作成するには、次の手順を実行します。

Oracle Application Server（OAS）の Welcome ページで、次の手順を実行します。

1. 「**OAS Manager**」をクリックします。

2. OAS Manager のナビゲーション・ツリーで、Web サイト・アイコンの横の「+」アイコンをクリックします。
3. 「**アプリケーション**」アイコンをクリックします。
4. アプリケーションの追加フォームを表示するには、「**追加**」アイコンをクリックします。

アプリケーションの追加フォームで、次の手順を実行します。

1. 「アプリケーション・タイプ」フィールドで、C Web を指定します。
2. 「**適用**」をクリックします。

追加フォームで、次の手順を実行します。

1. 「アプリケーション名」フィールドで、アプリケーション名を指定します。
2. 表示名フィールドで、表示名を指定します。
3. 「アプリケーションのバージョン」フィールドで、バージョン・ナンバーを指定します。
4. 「**適用**」をクリックします。
5. 成功ダイアログで、「**このアプリケーションにカートリッジ追加**」をクリックします。

Web カートリッジ追加フォームで、次の手順を実行します。

1. 「カートリッジ名」フィールドで、カートリッジ名を入力します。Forms60Cartridge
2. 「表示名」フィールドで、カートリッジ名を入力します。Forms60Cartridge
3. 「カートリッジ共有オブジェクト」フィールドで、共有オブジェクト・ファイル（IFWEBC60.DLL）の位置を指定します。たとえば、次のように入力します。

   ```
   D:¥ORANT¥BIN¥IFWEBC60.DLL
   ```

4. 「カートリッジのエントリ・ポイント」フィールドで、次のように入力します。`form_entry`
5. 「**適用**」をクリックします。
6. ナビゲーション・ツリーを更新するには、［**Shift**］キーをクリックしてから、「**再読み込み**」をクリックします。

アプリケーション構成プロパティを設定するには、次の手順を実行します。

1. ナビゲータで該当するツリー・ノードを拡張し、アプリケーションの構成フォルダに移動します。
2. 構成フォルダ内で、「**Web パラメータ**」をクリックします。

3. リスナー・リスト・フィールドで、リスナーを指定します。
4. 「**適用**」をクリックします。

カートリッジ構成プロパティを設定するには、次の手順を実行します。

1. ナビゲータで該当するツリー・ノードを拡張し、アプリケーションのカートリッジ構成フォルダに移動します。
2. カートリッジ構成フォルダ内で、「**カートリッジ・パラメータ**」をクリックします。
3. 次のパラメータおよび値を入力します（パラメータ名では大文字と小文字が区別されます）。

パラメータ	必須 / 任意	パラメータ値
ベース HTML	必須	実行時にカートリッジによってアクセスされるベース HTML ファイルの表示ディレクトリ・パスおよびファイル名。baseHTML パラメータを定義する場合、仮想ディレクトリ位置を指定しないでください。実行時に、Forms Server は新規 HTML ファイルを動的に作成するために、アプリケーション・カートリッジ設定、URL およびベース HTML ファイルを使用します。
HTML デリミタ	必須	変数名のデリミタ。デフォルトで % になります。
codebase	必須	表示ディレクトリ %ORACLE_HOME%¥forms60¥java をポイントするように定義した仮想ディレクトリ。アーカイブ時に与えられるパスおよびコードはこの URL に関係しています。
code	必須	コード・パラメータは削除や変更をしないでください。常に次の値にします。oracle.forms.engine.Main.
connectMode	**HTTP および HTTPS 接続では必須**；ソケット接続では任意	Forms Server で使用する接続プロトコルのタイプをクライアントに指定します。有効な値はソケット、http および https です。デフォルトはソケットです。
アーカイブ	任意	あらかじめロードする、カンマで区切ったアーカイブ・ファイルのリスト。絶対パスでない場合は codebase からの相対パス。
serverApp	任意	アプリケーションのクラス名がある場合に、デフォルトを置き換えます。アプリケーション固有のフォント・マッピングの作成およびアイコン・パスの設定には、アプリケーション・クラスを使用します。

パラメータ	必須 / 任意	パラメータ値
MetricsDomainName	任意	ロード・バランス用。Load Balancer Client サーバー・マシンのドメインを指定します。これは、LEASTLOADEDHOST パラメータとともに使用するロード・バランス・パラメータです。HTML ファイルでは、metricsDomainName パラメータは LEASTLOADEDHOST パラメータの直後にある必要があります。たとえば、次のようになります。ARCHIVE="%LEASTLOADEDHOST%metricsDomainName%/forms60java"
MetricsServerHost	任意	ロード・バランス用。E.6.1 項「ロード・バランスのカートリッジの構成」参照。
MetricsServerPort	任意	ロード・バランス用。E.6.1 項「ロード・バランスのカートリッジの構成」参照。
MetricsServerErrorURL	任意	ロード・バランス用。E.6.1 項「ロード・バランスのカートリッジの構成」参照。
MetricsTimeout	任意	ロード・バランス用。E.6.1 項「ロード・バランスのカートリッジの構成」参照。

4. 「**適用**」をクリックします。
5. 成功ダイアログ・ボックスで、「**OK**」をクリックします。
6. カートリッジ構成フォルダ内で、「**チューニング**」をクリックします。
7. 「起動インスタンス」フィールドで、次のように入力します。100.
8. 「**適用**」をクリックします。
9. 成功ダイアログ・ボックスで、「**OK**」をクリックします。

E.4 ベース HTML ファイルの構成

Web で使用可能なアプリケーションを最初に起動する場合（アプリケーションの URL へのリンクをクリックすることにより）、HTML ファイルがアプリケーション・サーバーからユーザーの Web ブラウザにダウンロードされます。この初期 HTML ファイルはベース HTML ファイルと呼ばれます。このファイルには、選択されたアプリケーションを Web 上で実行するために必要なすべてのタグ、パラメータおよびパラメータ値が含まれています。

カートリッジの実装を使用する場合、ベース HTML ファイルは動的に作成されます。実行時に Forms Server は次のソースからの情報をマージして、新規 HTML ファイルを動的に作成します。

- アプリケーションのカートリッジ HTML ファイル
- アプリケーションのカートリッジ設定

- アプリケーションの URL

新規（動的に作成された）HTML ファイルは、ユーザーの Web ブラウザにダウンロードされます。

ベース HTML ファイルを作成する最も簡単な方法は、提供されたテンプレート・ファイルのいずれかを変更することです。%FORSM60%¥server ディレクトリの次のテンプレートが使用できます。

- **basejini.htm**: これは、Oracle JInitiator を使用する Forms アプレットの実行に必要なタグが含まれる HTML ファイルです。オラクル社によってこの方法での動作が確認されたブラウザ (および標準の APPLET タグを使用して動作しないブラウザ) に適しています (Windows プラットフォームのみ)。5.4.2.4 項「デフォルトの basejini.htm ファイル」参照。
- **base.htm**: これは Forms アプレットを AppletViewer またはネイティブな JVM が Forms で作動することがオラクル社によって確認済みの、任意の Web ブラウザで実行するのに必要な APPLET タグを含む、ベース HTML ファイルです。5.4.2.3 項「デフォルトの base.htm ファイル」参照。

ベース HTML ファイルを作成するには、次の手順を実行します。

1. アプリケーションのベース HTML ファイルを作成するために、base.htm または basejinit.htm テンプレート・ファイルをコピーします。このテンプレートは %FORSM60%¥server ディレクトリにあります。
2. テンプレート・ファイルを開き、`order.htm` などに改名します。

注意 : カートリッジのベース HTML ファイルを作成する場合、ベース HTML ファイル名は baseHTML カートリッジ・パラメータ内で定義したファイルの名前と一致している必要があります。

3. 必要に応じてパラメータを変更します。次の表には、パラメータが含まれています。詳細は E.4.1 項「使用方法」を参照してください。

注意 : base.htm または basejinit.htm ファイル内で提供されるパラメータ・タグを使用しない場合は、タグをファイルから削除します。

パラメータ	必須 / 任意	パラメータ値
MetricsDomainName	カートリッジ・ロード・バランスの場合のみ**必須**	ロード・バランス用。Load Balancer Client サーバー・マシンのドメインを指定します。これは、LEASTLOADEDHOST パラメータとともに使用するロード・バランス・パラメータです。HTML ファイルでは、metricsDomainName パラメータは LEASTLOADEDHOST パラメータの直後にある必要があります。たとえば、次のようになります。 ARCHIVE="%LEASTLOADEDHOST%metricsDomainName%/forms60java"

パラメータ	必須 / 任意	パラメータ値
LEASTLOADEDHOST	カートリッジ・ロード・バランスの場合のみ**必須**	ロード・バランス中、負荷が最小のシステムの名前を持つカートリッジによって、この固定名プレース・ホルダが動的に置換されます。カートリッジは実行時に Metrics Server からこの情報を取得します。
標準アプレットまたはオブジェクト・パラメータ		
codebase	必須	表示ディレクトリ %ORACLE_HOME%¥forms60¥java をポイントするように定義した仮想ディレクトリ。アーカイブ時に与えられるパスおよびコードはこの URL に関係しています。
code	必須	コード・パラメータは削除や変更をしないでください。常に次の値にします。oracle.forms.engine.Main.
connectMode	**HTTP および HTTPS 接続では必須**; ソケット接続では任意	Forms Server で使用する接続プロトコルのタイプをクライアントに指定します。有効な値はソケット、http および https です。デフォルトはソケットです。
アーカイブ	任意	あらかじめロードする、カンマで区切ったアーカイブ・ファイルのリスト。絶対パスでない場合は codebase からの相対パス。
Forms アプレット用のパラメータ（PARAM タグ内）		
serverHost	任意	Forms Server の ifsrv60.exe を実行するホスト (デフォルトは Web リスナー・マシン)。
serverPort	必須	Forms Server の ifsrv60.exe がリスニングするポート。ほとんどの場合、ポート番号は 9000（デフォルト）のままです。
serverArgs	必須	Runform 用のコマンド・ラインパラメータ下記の Runform パラメータを参照してください。 forms_param を有効な Form Builder コマンド・ライン・パラメータで置換します。user_param を任意の有効なユーザー定義パラメータで置換します。例、<param name="serverArgs" VALUE="module=order.fmx"> **注意**: 複数の Form Builder コマンド・ラインおよびユーザー定義パラメータを与えることができます。HTML ファイルにディレクトリ・パスを入れるか、FORMS60_PATH 環境変数を定義することにより、.FMX ファイルの表示ディレクトリ・パスを指定する必要があります。拡張子 .FMX は任意です。
heartBeat	任意	このパラメータを使用して、クライアントが稼動中であることを示すためにサーバにパケットを送る頻度を設定します。この整数値は、分単位で定義します。デフォルトは 2 分です。

パラメータ	必須 / 任意	パラメータ値
imageBase	任意	このパラメータを使用して、アイコンファイルが格納される場所を指定します。次の中から選択します。 ■ **codeBase** は、アイコン検索パスが Java クラスを含むディレクトリに対応することを示します。アイコンを JAR ファイルに格納する場合にこの値を使用します（推奨）。 ■ **documentBase** は、デフォルトです。Forms Server CGI を使用した配置では、アイコンパスをカスタムアプリケーションファイル中に指定します。
registryPath	任意	このパラメータを使用して、serverApp パラメータで名前をつけたアプリケーションファイル名が格納されている仮想ディレクトリをリスト表示します。
webformsTitle	任意	このパラメータを使用して、フォームの表示ウィンドウの上端に表れるタイトルを変更します。
splashScreen	任意	アプレットが表示される前に表示する .GIF ファイルを指定。スプラッシュなしの場合は「NO」に設定します。デフォルトのスプラッシュを使用する場合は空白のままにします。
background	任意	背景に表示する .GIF ファイルを指定。背景なしの場合は「NO」に設定します。デフォルトの背景を使用する場合は空白のままにします。
clientDPI	任意	1 インチ当たりのドット数（DPI）を指定し、JVM によって戻される DPI 設定を上書きします。これにより、各プラットフォームのさまざまな DPI 設定を管理できます。たとえば、Win32 プラットフォームで開発されたフォームは、DPI 値の違いにより、UNIX プラットフォーム上では正しく表示されない可能性があります。clientDPI の値には、すべての正の整数を指定できます。Oracle は 50 から 200 の整数を使用することをお薦めします。<param name="clientDPI" value="200">
separateFrame	任意	アプレットを分割フレーム内に表示するかどうかを指定。有効な値 : True または False。
lookAndFeel	任意	アプリケーションのルック・アンド・フィールを指定。有効な値 : Oracle または Generic（Windows 95 のルック・アンド・フィール）。
colorScheme	任意	アプリケーションの配色を指定。有効な値 : Teal、Titanium、Red、Khaki、Blue、Olive または Purple。 **注意** : lookAndFeel が Generic に設定されている場合、colorScheme は無視されます。
ランフォーム・パラメータ（serverArgs パラメータ）		
MODULE	必須	Form のモジュール名（任意でパスを含みます）。

パラメータ	必須 / 任意	パラメータ値
USERID	任意	scott/tiger@ORA8 などのログイン文字列
ユーザー定義パラメータ	任意	任意の名前 / 値のペア。

4. 新規ベース HTML ファイルを Web サーバー上のディレクトリに配置します。新規ファイルは、仮想ディレクトリに対応する表示ディレクトリに配置する必要があります。たとえば、c:¥orant¥webhtml に対応する /webhtml/ という仮想ディレクトリを定義した場合、新規ファイルを c:¥orant¥webhtml に配置する必要があります。

E.4.1 使用方法

- パラメータはカートリッジ・パラメータとして設定されていなかった場合のみ、必要です。たとえば、serverPort パラメータをカートリッジ内に設定する場合、このタグをベース HTML ファイル内に定義する必要はありません。
- HTML ファイル・パラメータにデリミタ付き値を指定できます。たとえば、HTML ファイルに次の行を置くことができます。

  ARCHIVE="%Archive%"

  値を %Archive% に割り当てる必要があります（アプリケーションのカートリッジ設定またはアプリケーションの URL で）。

- すべての変数パラメータは実行時に値を受け取る必要があります。パラメータが値を受け取らない場合、Forms Server は、ユーザーの Web ブラウザに戻す HTML ファイルを作成できません。これにより、エラーが発生します。
- ロード・バランスを使用し、Oracle JAR ファイル（またはカスタム JAR ファイル）が格納されている表示ディレクトリをポイントするために ARCHIVE パラメータを使用する場合、すべてのものを同じシステムからダウンロードする必要があるため、この URL は CODEBASE の URL と同じである必要があります。したがって、CODEBASE 定義で %LEASTLOADEDHOST% を使用する場合、ARCHIVE 定義でもこれを使用します。たとえば、古い HTML ファイルに ARCHIVE 用の次のエントリがある場合です。

  ARCHIVE="/jars_vdir/f60all.jar"

  新規 HTML ファイルには ARCHIVE 用の次のエントリがあります。

  ARCHIVE="%LEASTLOADEDHOST%/jars_vdir/f60all.jar"

E.5 Forms Server の構成

初期構成設定を Oracle Installer によって変更する場合、次の項の作業を実行する必要があります。

- 環境変数の構成
- NT での Forms Server 起動パラメータの変更

E.5.1 環境変数の構成

この項では環境変数のカスタマイズの方法について説明します。Oracle Installer は、インストール・プロセス中に Forms Server 環境変数を自動的に設定します。変更が必要な場合はこれらの変数をカスタマイズできます。環境変数は次のとおりです。

環境変数	デフォルト値と説明
FORMS60_PATH	¥%ORACLE_HOME%¥tools¥web60¥temp 実行する Form を検索するときに、Forms が検索するパスを指定します。
FORMS60_OUTPUT	¥%ORACLE_HOME%¥tools¥web60¥temp 生成したレポート・ファイルを格納するアプリケーション・サーバー上の物理ディレクトリ。Reports を使用していない場合は、この環境変数は必要ありません。詳細は、7.5.2 項「レポートの実行」を参照してください。
FORMS60_MAPPING	/dev60temp FORMS60_OUTPUT 変数で定義された物理ディレクトリを指す仮想ディレクトリ。Reports を使用していない場合は、この環境変数は必要ありません。詳細は、7.5.2 項「レポートの実行」を参照してください。
FORMS60_MESSAGE_ENCRYPTION	TRUE 環境変数は RC4 40 ビットの暗号化機能を使用して、Forms メッセージを暗号化します。ソケットおよび HTTP 通信モードにのみ適用されます。デフォルトの設定で、通信は暗号化されます。
FORMS60_WALLET	¥%ORACLE_HOME¥forms60¥wallet HTTPS 通信モードのみに使用されます。詳細は 5.5 項「HTTPS 接続モード設定の追加ステップ」を参照してください。
FORMS60_HTTPS_NEGOTIATE_DOWN	FALSE HTTPS 通信モードのみに使用されます。詳細は 5.5 項「HTTPS 接続モード設定の追加ステップ」を参照してください。

注意： FORMS60_MAPPING 環境変数によって設定された仮想ディレクトリは、FORMS60_OUTPUT 環境変数によって設定された物理ディレクトリに対応している必要があります。

E.5.1.1 NT での環境変数のカスタマイズ

NT で環境変数をカスタマイズするには、次の手順を実行します。

1. レジストリ・エディタを起動します。
   a. タスクバーから、「**スタート**」→「**ファイル名を指定して実行**」を選択します。
   b. 「ファイル名を指定して実行」ダイアログ・ボックスに次を入力します。`regedit`
   c. 「**OK**」をクリックします。
2. レジストリ・エディタで、HKEY_LOCAL_MACHINE → SOFTWARE → ORACLE にナビゲートします。
3. 環境変数の場所を指定します。たとえば、FORMS60_PATH をハイライトします。
   a. 「**編集**」→「**変更**」を選択します。レジストリ・エディタにより「文字列の編集」ダイアログ・ボックスが開かれます。
   b. 環境変数の新しい値を入力します。
   c. 「**OK**」をクリックします。

E.5.1.2 UNIX での環境変数のカスタマイズ

UNIX 上では、環境変数をコマンド・シェルで定義します。たとえば次のようにします。

```
FORMS60_PATH=/%ORACLE_HOME%/tools/web60/temp
FORMS60_OUTPUT=/%ORACLE_HOME%/tools/web60/temp
FORMS60_MAPPING=/dev60temp
FORMS60_MESSAGE_ENCRYPTION=TRUE
```

E.5.2 NT での Forms Server 起動パラメータの変更

Oracle Installer を使用して NT 上に Forms Server をインストールするとき、Forms Server は NT サービスの 1 つとして設定され、特別に指定しない限り、再起動すると自動的に開始されます。

注意： Oracle Enterprise Manager (OEM) を使用している場合は、NT サービスとして実行するコンポーネントを設定しないでください。

Oracle Installer を使用する場合、Forms Server が常に初期構成の状態で起動されるように、ポートおよびモードなどインストール中に入力したすべての Forms Server パラメータがレジストリに保存されます。

Oracle Installer カスタム・インストール・オプションを使用する場合、すべての Forms Server パラメータがデフォルト値に設定され、レジストリに保存されます。

いずれの場合も、次のいずれかを実行することにより、起動パラメータを変更できます。

- 既存の Forms Server サービスに関するレジストリの編集
- Forms Server サービスの削除および再インストール
- Forms Server のテンポラリ・インスタンスの開始

E.5.2.1 既存の Forms Server サービスに関するレジストリの編集

すべての起動パラメータが次の Windows レジストリに保存されます。

HKEY_LOCAL_MACHINE → SYSTEM → CONTROLSET → SERVICES → ORACLEFORMSSERVER<*SERVICENAME*>

Forms Server の起動パラメータを変更するには、NT サービスのコントロール パネルでサービスを停止し、レジストリのパラメータ設定を編集し、そのレジストリ設定を保存してサービスを再起動します。起動パラメータ定義は、E.5.3 項「Forms Server 起動パラメータの説明」を参照してください。

E.5.2.2 Forms Server サービスの削除および再インストール

既存の Forms Server サービスを削除し、新規の起動パラメータを使用して再インストールできます。

コマンド・ウィンドウで、次のように入力します。

```
ifsrv60 -remove <FormsServerServiceNameToBeRemoved>
```

続いて次を入力します。

```
ifsrv60 -install <NewFormsServerServiceName> port=<portNum> mode=<socket/http/https>
[pool=<numOfRunforms> log=<logfilePath> exe=<RunformexeName>]
```

起動パラメータ定義は、E.5.3 項「Forms Server 起動パラメータの説明」を参照してください。

サービスが自動的に開始され、以降の起動時に対して "autostart" に設定されます。起動パラメータの設定は次の Windows レジストリの場所に保存されます。

HKEY_LOCAL_MACHINE → SYSTEM → CONTROLSET → SERVICES → ORACLEFORMSSERVER<*SERVICENAME*>

E.5.2.3 Forms Server のテンポラリ・インスタンスの開始

コマンド・ラインから Forms Server のテンポラリ・インスタンスを開始できます。マシンのシャットダウン時に、このインスタンスは終了し、再起動されません。次のように入力します。

```
ifsrv60 <TemporaryFormsServerName> port=<portNum> mode=<socket/http/https>
[pool=<numOfRunforms> log=<logfilePath> exe=<RunformexeName>]
```

起動パラメータ定義は、E.5.3 項「Forms Server 起動パラメータの説明」を参照してください。

注意 : テンポラリ・インスタンスに入力されたパラメータ値は以降の開始時に保存されません。

E.5.3 Forms Server 起動パラメータの説明

Forms Server 起動時には次のパラメータが使用されます。

- Port パラメータ
- Mode パラメータ
- Pool パラメータ
- Log パラメータ

E.5.3.1 Port パラメータ

サーバー・プロセスが開始されるポートを決定します。Forms Server プロセスの開始時にポート番号を指定しないと、デフォルトのポート 9000 上でプロセスが開始されます。サーバー・プロセスを開始するポート番号は、アプリケーションの HTML ファイル、構成パラメータまたは URL に指定する serverPort 番号と一致する必要があります。

E.5.3.2 Mode パラメータ

Forms Server を、ソケット・モード（ソケットの直接接続を使用）、HTTP モード（ファイアウォールを貫通できる）または HTTPS モード（ファイアウォールを貫通でき、SSL (secure sockets layer) を追加で使用してサーバー認証およびメッセージの暗号化を行う）のいずれかで実行するかを決定します。各モードの詳細は、3.3 項「ソケット、HTTP または HTTPS」を参照してください。

E.5.3.3 Pool パラメータ

後から使用するユーザーが利用できる、アクティブなスペアの接続数を決定します。たとえば "pool" が 5 に設定された場合は、5 つのアクティブなスペア接続があります。

E.5.3.4 Log パラメータ

パス名およびログ・ファイル名が提供されると、サーバーのログ・ファイルを生成します。例、log=¥PathName¥LogFileName。

E.6 カートリッジベースのロード・バランスの構成

これを実現するには、Oracle Application Server（OAS）を使用する必要があります。ロード・バランスのカートリッジを構成するには、次のステップを実行します。詳細は次の項を参照してください。

- ロード・バランスのカートリッジの構成
- Load Balancer Server のインストール
- Load Balancer クライアントのインストール
- Load Balancer Server の起動
- Load Balancer クライアントの起動
- HTTPD/Web リスナーの各 Load Balancer クライアントシステム での起動

E.6.1 ロード・バランスのカートリッジの構成

ロード・バランスのカートリッジを構成するには、次の手順を実行します。

1. Oracle Application Server（OAS）の Welcome ページ（http://myserver.com:8888 など）で、「**OAS Manager**」をクリックします。
2. OAS Manager のナビゲーション・ツリーで、Web サイト・アイコンの横の「+」アイコンをクリックします。
3. 「アプリケーション」アイコンの横の「+」アイコンをクリックします。
4. 次の手順を実行することにより、ロード・バランスのカートリッジ構成プロパティを設定します。
   a. ナビゲータで該当するツリー・ノードを拡張し、アプリケーションのカートリッジ構成フォルダに移動します。
   b. カートリッジ構成フォルダ内で、「**カートリッジ・パラメータ**」をクリックします。
   c. 次のパラメータおよび値を入力してから、「**適用**」をクリックします。

パラメータ	パラメータ値
MetricsServerHost	Metrics Server が実行されているホスト（システム）の名前を入力します。 例 : system1.company.com
MetricsServerPort	負荷が最小のシステムのカートリッジから Metrics Server がリクエストを受信するポート番号を入力します。このパラメータ値は、Forms Server の値とは異なる必要があります。例 : 70000 **注意**：デフォルトでは、Forms Server はポート 9000 で受信します。

パラメータ	パラメータ値
MetricsServerErrorURL	Metrics Server が負荷が最小のシステムの名前に MetricsTimeout 期間内に応答しない場合、Web Client を指示する URL を入力します。URL を使用して次のことが行えます。(*) 他のホストの Forms カートリッジをポイントします。これにより、自動的にリクエストが発行されます。例 : http://altsys.company.com/web_cart?module=inv&userid=clerk@stock&deptid=90 (*)Web サーバー上の静的または動的な HTML ページをポイントします。Metrics Server がダウンし、ロード・バランスが使用できない場合、このページを使用してヘルプおよび指示をユーザーに提供できます。例 : http://altsys.company.com/apps_html/help.html
MetricsDomainName	Metrics Client サーバー・マシンのドメイン名を入力します。たとえば、次のように入力します。.us.oracle.com **注意**：各 Metrics Client は同じドメインにある必要があります。
MetricsTimeout	（任意で）Metrics Server に負荷が最小のシステム名のリクエストを行う際に Forms カートリッジが使用する、タイムアウト値（秒）を入力します。カートリッジはこの時間内に応答を受信しない場合、HTML ファイル内の %LEASTLOADEDHOST% エントリを MetricsServerErrorURL 値に置換します。MetricsTimeout パラメータは任意です。指定しない場合、デフォルト値は 30 秒です。例 : 35

注意 : Load Balancer Server のバックアップとして Load Balancer クライアントの使用を計画している場合、Load Balancer クライアントのこれらの構成ステップを実行する必要もあります。計画していない場合は、Load Balancer Server のこれらのステップのみ実行する必要があります。

E.6.2 Load Balancer Server のインストール

Windows NT の場合 :

NT サービスとして Load Balancer Server をインストールするには、プロンプトに対して、`d2ls60 -install` と入力します。

注意 : Load Balancer Server サービスを取り除くには、`d2ls60 -remove` と入力します。

UNIX の場合 :

Load Balancer Server がデーモンとして実行されます。インストールは必要ありません。UNIX では、Load Balancer Server 実行ファイル名は d2ls60 です。

注意 : Oracle Installer を使用して Load Balancer Server をインストールし、NT サービスを作成する -install を使用する場合、d2ls60 は起動パラメータを受け入れ保存します。このサービスは再度パラメータを与えなくても、手動または自動起動で起動できます。Oracle Installer はこれを使用し、正しい起動パラメータを使用してサービスを作成します。

E.6.3 Load Balancer クライアントのインストール

Windows NT の場合 :

各 Load Balancer クライアントを NT サービスとしてインストールするには、各システムで次のコマンドを実行します。`d2lc60 -install`

注意 : Load Balancer クライアントサービスを取り除くには、次のように入力します。`d2lc60 -remove`

UNIX の場合 :

Load Balancer クライアントがデーモンとして実行されます。インストールは必要ありません。UNIX では、Load Balancer クライアント実行ファイル名は d2lc60 です。

E.6.4 Load Balancer Server の起動

Windows NT の場合 :

1. **「スタート」→「設定」→「コントロール・パネル」**をクリックします。
2. 「**サービス**」をダブルクリックします。
3. 「**Oracle_Load_Balancer_60_Server**」を選択します。
4. 次の起動パラメータの値を入力します。

```
MetricsClientToServerPort# MetricsServerToCartridgePort# max_no_MetricsClients Trace
```

   - **MetricsClientToServerPort#** は、Load Balancer クライアントが接続し負荷情報を送信する Load Balancer Server のポートです。E.6.5 項「Load Balancer クライアントの起動」**で記述したように、この値は Load Balancer クライアントの MetricsClientToServerPort# に設定した値と一致している必要があります。**
   - **MetricsServerToCartridgePort#** は、カートリッジが Load Balancer Server への情報の問合せに使用する Load Balancer Server のポートです。この値は、E.6.1 項「ロード・バランスのカートリッジの構成」で記述する MetricsServerPort プロパティに対応しています。
   - **max_no_MetricsClients** は、負荷情報を実行し Load Balancer Server に送信する Load Balancer クライアントの最大数を指定するための任意のパラメータです。デフォルト値は 1000 です。
   - **trace** は、Load Balancer Server のトレース出力を作成できる任意のパラメータです。デフォルトは 0 です。詳細は 12.5 項「Load Balancer Server トレース・ログの設定」を参照してください。

5. 「**スタート**」をクリックして、サービスを起動します。

UNIX の場合 :

次のコマンドを入力します。

```
d2ls60 MetricsClientToServerPort# MetricsServerToCartridgePort# max_no_
```

```
MetricsClients trace
```

E.6.5 Load Balancer クライアントの起動

Windows NT の場合 :

1. **「スタート」→「設定」→「コントロール・パネル」**をクリックします。
2. 「**サービス**」をダブルクリックします。
3. 「**Oracle_Load_Balancer_60_Client**」を選択します。
4. 次の起動パラメータの値を入力します。

```
d2lc60 MetricsServerHostName MetricsClientToServerPort# MetricsClientLocalPort#
ScaleFactor
```

   - **MetricsServerHostName** は、Load Balancer Server が実行されるシステムの名前です。
   - **MetricsClientToServerPort#** は、Load Balancer クライアントが接続し負荷情報を送信する Load Balancer Server のポートです。E.6.4 項「Load Balancer Server の起動」**で記述したように、この値は Load Balancer Server の MetricsClientToServerPort# に設定した値と一致している必要があります**。
   - **MetricsClientLocalPort#** は、クライアントが Load Balancer Server への情報の問合せに使用する Load Balancer クライアントのポートです。
   - **ScaleFactor** は、Load Balancer クライアントの様々な能力から生じる不均衡を削減するための任意のパラメータです。Forms Server のロード・バランスは各 Load Balancer クライアント上で実行されているプロセスの総数にのみ基づいているので、最小負荷システムと思われるシステムは、新規プロセスを実行するのに必ずしも最適な場所ではありません。システムの能力が低い場合、ScaleFactor には大きい値を割り当てる必要があります。ScaleFactor のデフォルト値は、UNIX の場合は 1 で、Windows NT の場合は 4 です。

5. 「**スタート**」をクリックして、サービスを起動します。

UNIX の場合 :

UNIX で実行される各 Load Balancer クライアントを起動するには、次のコマンドを入力します。

```
d2lc60 MetricsServerHostName MetricsClientToServerPort# MetricsClientLocalPort#
ScaleFactor
```

E.6.6 HTTPD/Web リスナーの各 Load Balancer クライアントシステムでの起動

未署名のアプレット（デフォルト）を実行する場合、Oracle Application Server（OAS）の HTTPD リスナーを各 Load Balancer クライアントでインストールおよび構成する必要があります。これにより、Java クラス・ファイルをユーザーのブラウザにダウンロードできます。

HTTPD/Web リスナーを各 Load Balancer クライアントシステムで起動するには、次の手順を実行します。

1. HTTPD リスナーを各 Load Balancer クライアントシステムにインストールします。HTTPD リスナーの Load Balancer クライアントへのインストールの詳細は OAS のドキュメントを参照してください。
2. 各 HTTPD リスナーを構成します。
   a. OAS ホームページ（http://myserver.com:8888 など）で、「**Web Application Server Manager**」をクリックします。
   b. 「**Oracle Web リスナー**」をクリックします。
   c. 「Oracle Web リスナー拡張構成」ページを表示するには、「**構成**」（WEBSVR の横）をクリックします。
   d. 「**ディレクトリ**」（左フレーム内）をクリックします。
   e. 「ファイルシステム・ディレクトリ」フィールドで、`%ORACLE_HOME%¥forms60¥java¥` と入力します。
   f. 「仮想ディレクトリ」フィールドで、次のように入力します。`/forms_code/`
   g. 「**リスナーの変更**」をクリックします。

注意： /forms_code/ パラメータは、ベース HTML ファイル内またはカートリッジ内で定義された codebase パラメータに対応しています。/forms_code/ を仮想ディレクトリ名として指定しなかった場合、/forms_code/ を仮想ディレクトリ名に置換する必要があります。

3. HTTPD リスナーを起動します。
   a. 「**リスナー**」（ページの一番下）をクリックします。
   b. WEBSVR リスナーを停止するには、「**停止**」をクリックします。
   c. WEBSVR リスナーを再起動するには、「**起動**」をクリックします。

各クライアントでの HTTPD/Web リスナーの構成および起動の詳細は OAS のドキュメントを参照してください。

注意： OAS の完全バージョンを各 Load Balancer クライアントシステムにインストールする必要はありません。HTTPD リスナーのみインストールする必要があります。

F

Graphics Server

F.1 概要

この付録には、Graphics Server をサポートするための環境構成に関する、次の項が含まれています。

- Graphics Server の概要
- Graphics Server の構成
- OAS Web サーバーの起動
- Graphics の Web への配置
- Web 配置用 Graphics アプリケーションの設計

F.2 Graphics Server の概要

Graphics Server は、対話型 Graphics アプリケーションをインターネットに配置するための中間層アプリケーション・サーバーです。

図 F-1 に示す Graphics Web アーキテクチャは、次の 3 つの主要なコンポーネントで構成されます。

- Graphics Server
- Web Request Broker
- Graphics Client

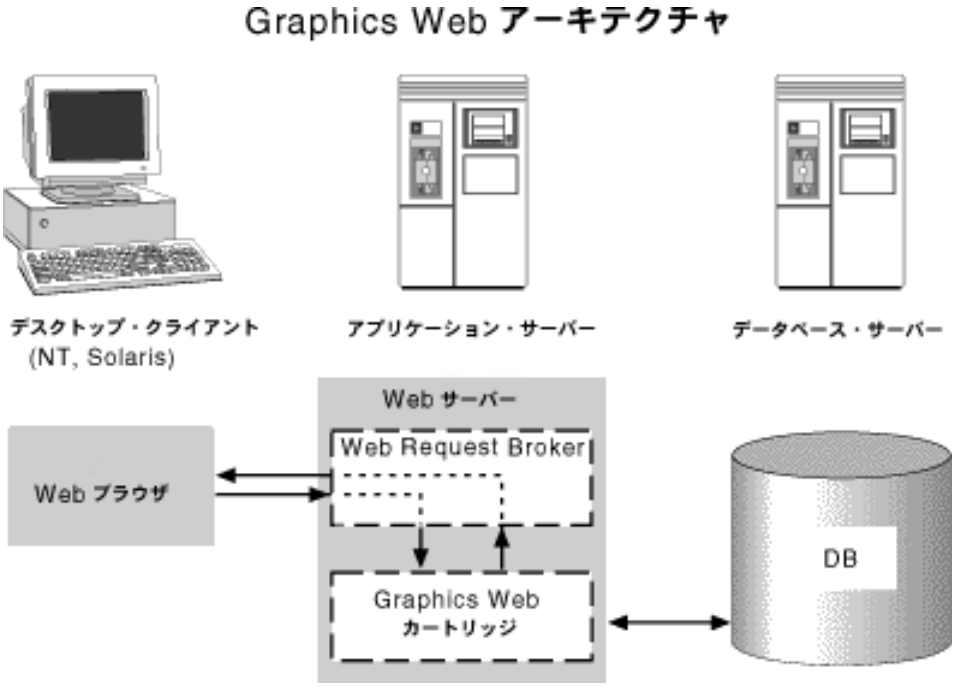

図 F-1 Graphics Server のアーキテクチャ

Graphics Client および Graphics Server は HTTP 接続を使用し、Web Request Broker を使用してリレーされた一連のリクエストおよび応答を通信します。

URL を直接入力するか、表示領域をマウスでクリックするか、HTML フォームでリクエストを投入することにより、リクエストはクライアント側で開始されます。すべてのリクエストは、次のいずれかの形式で渡されます。

- 単純な URL
- Graphics Builder Batch で使用されるものに類似した特定のキーワード引数（ユーザー定義パラメータの userids または values など）を含む、パラメータ化 URL

応答は、次のいずれかの形式で戻されます。

- Graphics Web インタフェースとして機能する HTML フォーム
- 要求された表示イメージを含む HTML フォーム

F.2.1 Graphics Server

Graphics Server は、クライアント / サーバー配置で使用される Graphics Runtime Engine のバリエーションです。Graphics Server は Web サーバー上で実行され、アプリケーション・サーバーとして機能します。

クライアントにより投入され、Web Request Broker でリレーされたリクエストまたはイベントによって起動された場合、Graphics Server はそのイベントを処理し、更新された表示イメージのリファレンスを含む HTML ページを返します。したがって、対話型グラフィカル・アプリケーションを作成するために、マウス・ボタンを押すイベントをサーバー上で処理できます。

Graphics Server は Oracle のネットワーキング・ソフトウェア Net8 を使用して、データベースと通信します。Net8 は Forms Server に組み込まれており、ネットワーク全体のデータベースとの接続および通信を実現します。

F.2.2 Web Request Broker

Web Request Broker は Oracle Application Server とともに提供され、Netscape、Microsoft Web Server などでサポートされている、独立したモジュールです。Web Request Broker では、Graphics Server などのカートリッジのフレームワークが提供されます。また、カートリッジへのクライアント接続も管理されます。

F.2.3 Graphics Client

Graphics Client では、クライアント・サーバーを実現する際に使用する Graphics Runtime インタフェースに類似したインタフェースが表示されます。Graphics Client を使用すると、これまでのクライアント・サーバー Graphics アプリケーション資産を変更することなく Web 上で実行できます。プラットフォーム上で実行されているすべての HTML Web ブラウザは、Graphics Client として機能します。

F.3 Graphics Server の構成

Graphics Server を構成するには、次の項の説明に従って、環境変数を設定しロード・バランスを構成する必要があります。

- Graphics Server 環境変数の設定
- Graphics カートリッジ用 OAS の構成

F.3.1 Graphics Server 環境変数の設定

この項では、次の OS の環境変数の設定方法について説明します。

- Microsoft Windows NT
- UNIX

F.3.1.1 Windows NT

Windows NT 上で Graphics Server の環境変数を設定するには、次の手順を実行します。

1. レジストリ・エディタを起動します。
   a. タスクバーから、「**スタート**」→「**ファイル名を指定して実行**」を選択します。
   b. 「ファイル名を指定して実行」ダイアログ・ボックスに次を入力します。`regedit`
   c. 「**OK**」をクリックします。
2. レジストリ・エディタで、HKEY_LOCAL_MACHINE → SOFTWARE → ORACLE にナビゲートします。
3. GRAPHICS60_MAPPING 環境変数を検索し、ハイライトします。
   a. 「**編集**」→「**変更**」を選択します。レジストリ・エディタにより「文字列の編集」ダイアログ・ボックスが開かれます。
   b. Web 上で実行する .OGD を含むディレクトリのフル・パスを入力します。たとえば、%ORACLE_HOME%/GRAPHICS60/WEB_OGD と入力します。
   c. 「**OK**」をクリックします。
4. OWS_IMG_DIR 環境変数を検索し、ハイライトします。
   a. /ows-img/ 仮想ディレクトリの下に web_tmp というディレクトリがあることを確認してください。このディレクトリが存在しない場合は、作成してください。
   b. 「**編集**」→「**変更**」を選択します。レジストリ・エディタにより「文字列の編集」ダイアログ・ボックスが開かれます。
   c. /ows-img/ 仮想ディレクトリの位置を入力します。Oracle Application Server (OAS) 管理ページに /ows-img/ 仮想ディレクトリを定義します。デフォルトでは、/ows-img/ は表示ディレクトリ %ORACLE_HOME%/ows/4.0/img/ にマップされます。
   d. 「**OK**」をクリックします。

F.3.1.2 UNIX

UNIX 上で Graphics Server の環境変数を設定するには、次の環境変数を定義するために、Web Server スクリプト・ディレクトリのシェル・スクリプトをセットアップします。

- Web 上で実行する .OGD を含むディレクトリのフル・パスに GRAPHICS_WEB_DIR を設定します。たとえば、%ORACLE_HOME%/GRAPHICS60/WEB_OGD と設定します。
- OWS_IMG_DIR を /ows-img/ 仮想ディレクトリの位置に設定します。Oracle Application Server (OAS) 管理ページに /ows-img/ 仮想ディレクトリを定義します。デフォルトでは、/ows-img/ は表示ディレクトリ %ORACLE_HOME%/ows/4.0/img/ にマップされます。

注意： UNIX の場合、GRAPHICS_WEB_DIR および OWS_IMG_DIR 環境変数を設定したシェルからリスナーを起動する必要があります。

注意： UNIX の場合、PRINTER および DISPLAY を有効なデバイスに設定する必要があります。

F.3.2 Graphics カートリッジ用 OAS の構成

カートリッジを作成するには、次の手順を実行します。

1. Oracle Application Server (OAS) の Welcome ページで、「**OAS Manager**」をクリックします。
2. OAS Manager のナビゲーション・ツリーで、Web サイト・アイコンの横の「+」アイコンをクリックします。
3. 「**アプリケーション**」アイコンをクリックします。
4. アプリケーションの追加フォームを表示するには、「**追加**」アイコンをクリックします。
5. アプリケーションの追加フォームで、次の手順を実行します。
   a. 「アプリケーション・タイプ」フィールドで、C Web を指定します。
   b. 「**適用**」をクリックします。
6. 追加フォームで、次の手順を実行します。
   a. 「アプリケーション名」フィールドで、アプリケーション名を指定します。
   b. 表示名フィールドで、表示名を指定します。
   c. 「アプリケーションのバージョン」フィールドで、バージョン・ナンバーを指定します。
   d. 「**適用**」をクリックします。
   e. 成功ダイアログで、「**このアプリケーションにカートリッジ追加**」をクリックします。
7. 「C Web カートリッジの追加」フォームで、次の手順を実行します。
   a. 「カートリッジ名」フィールドで、カートリッジ名を入力します。（たとえば、Graphics60Cartridge）
   b. 「表示名」フィールドで、カートリッジ名を入力します。（たとえば、Graphics60Cartridge）
   c. 「カートリッジ共有オブジェクト」フィールドで、共有オブジェクト・ファイル (GCW60.DLL) の位置を指定します。たとえば、次のように入力します。D:¥ORANT¥BIN¥GCW60.DLL

d. 「カートリッジのエントリ・ポイント」フィールドで、次のように入力します。
GCWWRBMain

e. 「クライアント・セッション」を使用可能に設定します。

f. 「最大セッション・アイドル時間」を所定のクライアント・セッションの時間切れ時間に設定します。

g. Process/Threads を Process に設定します。

h. 「**適用**」をクリックします。

注意 : 複数の Graphics Cartridge インスタンスを使用して、OAS の 1 つのインスタンスを構成できます。

8. ナビゲーション・ツリーを更新するには、[**Shift**] キーをクリックしてから、「**再読み込み**」をクリックします。

9. 次の手順を実行することにより、アプリケーション構成プロパティを設定します。

a. ナビゲータで該当するツリー・ノードを拡張し、アプリケーションの構成フォルダに移動します。

b. 構成フォルダ内で、「**Web パラメータ**」をクリックします。

c. リスナー・リスト・フィールドで、リスナーを指定します。

d. 「**適用**」をクリックします。

10. 次の手順を実行することにより、カートリッジ構成プロパティを設定します。

a. ナビゲータで該当するツリー・ノードを拡張し、アプリケーションのカートリッジ構成フォルダに移動します。

b. カートリッジ構成フォルダ内で、「**カートリッジ・パラメータ**」をクリックします。

c. 次のパラメータおよび値を入力します（パラメータ名では大文字と小文字が区別されます）。

名称	説明
GCW_IMAGES_USE_FILES	Graphics 図表を表示する HTML ページを返す際に、Graphics Server がテンポラリ GIF イメージ・ファイルをサーバー上（Web サーバー・パス内の img ディレクトリ下の web_tmp ディレクトリ下）に作成するかどうかを指定します。値が NO である場合、かわりに HTML ページが Graphics Server を参照し（2 回目）、テンポラリ・ファイルをサーバー上に作成しないで、サーバーによってクライアントに戻されたバイナリ形式のイメージ・データを取り出します。 **注意**：テンポラリ GIF ファイルを作成することにより、これらのファイルをサーバーから定期的に削除する管理作業が必要になります（これらのファイルが使用されていないことが明らかな場合）。ただし、Graphics カートリッジは古いイメージ・ファイルを自動的に削除します。テンポラリ・ファイルを使用すると、サーバー上に未使用のイメージ・ファイルが残ります。使用ストリーム・イメージがわずかに遅くなり、パフォーマンスがわずかに向上するため、これは相殺されます。ストリーム・イメージを使用すると、イメージ・リクエストが Web Request Broker を使用して Graphics Server に転送されるため（最適化された操作である、サーバーへの通常のイメージ・ファイル・リクエストの代わり）、処理は遅くなります。
GCW_LINKS_CLOSE_PREV	埋込みハイパーリンクを介して他の OGD ファイルにナビゲートする際、Graphics Server が現行の OGD ファイルを閉じるかどうかを指定します。パラメータ値が YES であり、Graphics Server 上で他の OGD をオープンするために埋込み URL が使用されている場合（たとえば、文字列 "openfile" を含む）、現行の OGD は閉じられ、新規の OGD がオープンされます。したがって、ユーザーは最初の OGD をオープンしておく必要がないため、リソースがサーバー上に保存されます。 **注意**：他のタイプの URL（たとえば、他の Web ページまたは HTML オブジェクトへの）は通常どおり機能します。これらのリンクを含む OGD は、このパラメータ値に関係なく、実行後もオープンされたままです。ブラウザの「戻る」ボタンまたは他の URL を使用して最初の OGD に戻ると、このドキュメントの最新バージョンが表示されます。以前変更した内容は表示されません。このパラメータのデフォルト値は YES です。このパラメータ値を上書きするには、埋込み URL を起動するための、OGD PL/SQL ファンクション内のバインド変数 OG_URL_CLOSE_THIS_DOC を YES または NO に設定します（つまり、OG_URL および OG_URL_TARGET を使用します）。 **注意**：OG_URL_CLOSE_THIS_DOC の値は、このパラメータ値を上書きするたびに（リンクが実行される前に）明示的に設定する必要があります。これは、特別な場合を意味するからです。
GCW_WRITE_TRACE_FILE	Graphics Server がトレース・ファイルをサーバーのファイル・システムに作成するかどうかを指定します。管理者はデバッグするためにトレース・ファイルを使用できます。このファイルの名前は GWTRACE.TXT です。トレース・ファイルは特別な状況で必要になるため、このパラメータのデフォルト値は NO です。

名称	説明
GRAPHICS_GC_TIME	この期間内にアクセスされないオープン・ファイルのメモリーを自動的にクローズおよび解放します。設定されていない場合、これは行われません。設定する場合、カートリッジの「最大セッション・アイドル時間」の値以上にする必要があります。
GCW_ORACLE_SID	Graphics カートリッジ・ユーザーが接続するデフォルト・データベース名に設定します。この設定は、GCW_REALM も設定されている場合のみ有効です。2 つの値が設定されている場合、GCW_ORACLE_SID で指定されたデータベースおよび GCW_REALM で指定された Realm に対して認証されたユーザー ID およびパスワードを指定するようプロンプトが表示されます。この認可メソッドは、OAS でサポートされている他の認証とは機能しません。
GCW_REALM	認証中にユーザーを調べるために使用される Web Realm 名に設定します。この設定は、GCW_ORACLE_SID とともに使用する必要があります。2 つの値が設定されている場合、GCW_ORACLE_SID で指定されたデータベースおよび GCW_REALM で指定された Realm に対して認証されたユーザー ID およびパスワードを指定するようプロンプトが表示されます。この認可メソッドは、OAS でサポートされている他の認証とは機能しません。

d. 「**適用**」をクリックします。

e. 成功ダイアログ・ボックスで、「**OK**」をクリックします。

f. カートリッジ構成フォルダ内で、「**チューニング**」をクリックします。

g. 「起動インスタンス」フィールドで、次のように入力します。100

h. 「**適用**」をクリックします。

i. 成功ダイアログ・ボックスで、「**OK**」をクリックします。

11. （UNIX のみ）カートリッジを実行する前に、ディレクトリ $OWS_IMG_DIR/web_tmp を作成します。次に、これにグローバル書込み許可を与えます。たとえば、プロンプトが表示されたら、次のように入力します。

```
mkdir $OWS_IMG_DIR/web_tmp
chmod 777 $OWS_IMG_DIR/web_tmp
```

F.4 OAS Web サーバーの起動

Oracle Application Server (OAS) を起動するには、次の手順を実行します。

1. （UNIX のみ）ORACLE_HOME 環境変数を設定します。たとえば、次のように入力します。`setenv %ORACLE_HOME%/private/ora_home`

2. プロンプトに対して、次のように入力します。

```
owsctl start
```

ブラウザを起動するには、次の手順を実行します。

1. プロンプトが表示されたら、ブラウザを起動します。
2. 該当する URL（http://< マシン名または IP アドレス >:<Web Server Admin リスナーのポート番号 >）にナビゲートすることにより、Web サーバーにアクセスします。

   たとえば、ブラウザで次の URL をポイントします。http://myserver.com:8888

F.5 Graphics の Web への配置

Graphics アプリケーションを配置する場合、次のことが必要です。

- ランタイム実行ファイルの作成
- Graphics ファイルの配置
- Graphics 図表へのアクセス

F.5.1 ランタイム実行ファイルの作成

.OGD ランタイム実行ファイルは、実行ファイルを配置するアプリケーション・サーバーと同じプラットフォーム上で作成する必要があります。

F.5.2 Graphics ファイルの配置

Graphics .OGD アプリケーションは、%ORACLE_HOME%¥GRAPHICS60¥WEB_OGD ディレクトリに配置する必要があります。これにより、「Graphics Web」ツールバーのプルダウン・リストを使用して、.OGD アプリケーションを使用できます。

UNIX の場合、GRAPHICS_WEB_DIR 環境変数によって設定されたディレクトリに .OGD アプリケーションを配置する必要があります。

F.5.3 Graphics 図表へのアクセス

Graphics アプリケーションにアクセスするには、Graphics Web インタフェースを使用するのが最も簡単です。また、「Graphics Web」ツールバーを使用しないで、必要なパラメータを含む URL を作成して Graphics アプリケーションにサーバーから直接アクセスすることもできます。

F.5.3.1 Graphics Server の URL

Graphics Web インタフェース・ツールバーを使用して Graphics アプリケーションにアクセスするには、Web ブラウザでツールバーを起動する URL をポイントする必要があります（http://my_server/ogweb など）。

次の手順が実行されます。

1. クライアントが URL を Web サーバーに渡します。
2. サーバーの Web Request Broker がリクエストを Graphics Server に転送します。
3. Graphics Server が、Graphics Web インタフェース・ツールバーとして機能する HTML フォームを戻します。
4. このツールバーで、データベースに接続するためのユーザー名およびパスワードを入力し、使用可能な図表リストから図表を選択してから、「Open」をクリックします。これにより、ブラウザが次のようなパラメータ化 URL フォーム内のリクエストを送ります。

   ```
   http://my_server/ogweb?openfile=my_display.ogd&userid=scott/tiger@og&my_
   parameter=my_value
   ```

5. Web Request Broker は、このリクエストを Graphics Server に転送します。
6. Graphics Server は、リクエストされたイメージを含む HTML ページを戻します。
7. たとえば図表領域でマウスをクリックしてイベントを実行するたびに、クライアントは他のリクエストを生成します。イベントは、イベントの詳細を記述するキーワードを含む自動生成パラメータ化 URL を介してリレーされます（たとえば、マウスの X および Y 位置）。

F.5.3.2 Graphics Server のパラメータ化 URL

Graphics Builder Batch をコマンド・ラインから起動するために使用される標準キーワードをよく理解している場合、「Graphics Web」ツールバーを使用しないで、Graphics アプリケーションにサーバーから直接アクセスするために必要なパラメータを含む URL を作成できます。このアプリケーションへのアクセス・メソッドを使用すると、次のことが実行できます。

- Graphics アプリケーションのカスタム HTML ページへの埋込み
- 図表に含まれる PL/SQL コードの実行
- 図表リクエスト内の追加情報の提供（ユーザー ID、ユーザー定義パラメータの値など）
- Graphics Web インタフェース・ツールバーのオン / オフ切替え

前述のいずれかを実行するには、次のようなパラメータ化 URL を送る必要があります。

```
http:/www.my_company.com/ogweb?openfile=my_display.ogd&userid=scott/tiger@og&oracle_
interpret="BEGIN; MY-PROC(my_argument); END;"
```

次の手順が実行されます。

1. クライアントが URL を Web サーバーに渡します。
2. サーバーの Web Request Broker がリクエストを Graphics Server に転送します。

3. このサーバーは追加キーワードで指定された操作を実行し、要求された図表イメージを含む HTML ページを戻します。
4. たとえば図表領域でマウスをクリックしてイベントを実行するたびに、クライアントは他のリクエストを生成します。イベントは、イベントの詳細を記述するキーワードを含む追加自動生成パラメータ化 URL を介してリレーされます（たとえば、マウスの X および Y 位置）。

Graphics Server を起動するために使用する URL 内に名前と値のペアとして表示できる、多くの Graphics キーワードがあります。たとえば、次の URL を送ることができます。

```
http://www.myserver.com/ogweb?openfile=mydisplay.ogd&userid=scott/tiger@og&showtoolbar=yes
```

データベースに接続し、図表をオープンし、Graphics Web インタフェース・ツールバーをオンにするために、このリクエストでは userid、openfile および showtoolbar キーワードが使用されます。

次の表では、有効なキーワードが定義され、各キーワードで使用可能な値の範囲が指定されます。

キーワード	使用	有効値
openfile	Graphics Builder 図表ファイル (.OGD) 名をオープンするように指定します。	有効な Graphics Builder 図表ファイルの名前です。指定しない場合、.OGD または .OGR という拡張子が付けられます。 例 : openfile=my_display.ogd
userid	必要なデータベースにログオンするための完全接続文字列を指定します。	userid/password@dbname 例 : userid=scott/tiger@og
showtoolbar	「Graphics Builder Web」ツールバーのオン / オフを切り替えます。	yes または no。 例 : showtoolbar=yes
sessionid	Graphics Server を使用して表示される Graphics Builder 図表のインスタンスの一意の識別子。	Graphics Server によって生成される英数字セッション ID 番号。 例 : sessionid=000001.091032
close	指定された図表をクローズします。 **注意**：デフォルトでは、サーバー・パラメータ GW_TIMEOUT で指定された期間が終了するとサーバーはオープン図表を自動的にクローズするので、セキュリティの理由でドキュメントを厳しく管理する必要がある場合のみ、このキーワードは必要です。	yes が close キーワードの唯一の有効値です。 例 : sessionid=000001.091032 close=yes

Graphics Server に送られるすべての URL では、スペースを + に変更し、16 進コード化された特殊文字をコード化する、標準 URL フォーマットを使用する必要があります。

URL コード化の詳細は http://www.w3.org/Addressing/Addressing.html の W3C (World Wide Web Consortium) ドキュメントの名前およびアドレス、URI、URL、URN、URC を参照してください。

F.6 Web 配置用 Graphics アプリケーションの設計

次に、Web 配置用 Graphics アプリケーションを設計する際のヒントを紹介します。

- アプリケーション内に Graphics 図表を埋め込みます。
- オブジェクトとアプリケーションのコード共有率を最大にし、ロード中のファイル・サイズを最小にするために、できる限り多くのコードをライブラリに入れます。
- アプリケーション内のマルチメディア使用を制限します。マルチメディアを使用する場合、メディア情報を含む URL をコールするボタン・トリガーを再定義します。
- 画面再表示に関するアクティビティを制限します。たとえば、ユーザー・インタフェースがベースになったタイマーの使用は避けます。タイマーによってではなく、ユーザーの介在によってイベントが開始されるように、ユーザー・インタフェースを設計します。
- ハイパーリンク・ドリルダウンを作成するには、カスタム・ハイパーリンクを利用します。
- 異なる URL （http://www.oracle.co.jp など）へのハイパーリンクに対するボタン・プロシージャを作成する場合、ogd ではパラメータ OG_URL を使用し、ボタン・プロシージャはこのパラメータを URL に設定する必要があります。URL の異なるターゲット（他のウィンドウまたはフレームなど）を指定するには、ogd ではパラメータ OG_URL_TARGET を使用し、ボタン・プロシージャはこのパラメータをこのターゲットに設定する必要があります。
- 図表内のレイヤー数を制限することにより、図表ファイル・サイズを小さくしておきます。また、オブジェクトをプログラムで作成し、データ集約的な図表のストアド・プロシージャを利用します。
- Web 用 Graphics アプリケーションの設計時に、次の未サポート機能の使用は避けます。
  - ドラッグ・アンド・ドロップ
  - When-Mouse-Up トリガー・イベント（このリリースでは、When-Mouse-Down のみサポートされています。）

第Ⅱ部

索引

数字

A

B

C

D

F

G

H

I

J

T

U

V

W

あ

い

え

か

に

ね

は

ひ

ふ

へ

ま

ゆ

よ

り

れ

ろ