Microsoft

Transaction Server

トランザクショナル コンポーネント サービス

Microsoft Transaction Server
リリース 2.0 評価ガイド
Windows NT 上のアプリケーションの
開発と導入のための最高のテクノロジー

概要

Microsoft® Transaction Server バージョン 2.0(MTS)は、Microsoft Component Object Model(COM)テクノロジーを使って構築されたサーバー中心型のアプリケーションの*開発*と*導入*を単純化する、Microsoft Windows NT® オペレーティング システムの重要な新機能です。MTS は、Web ベースのインターフェイスを使用し、Microsoft Visual Basic® プログラミング システムなどの強力な開発ツールによるスケーラブルな 3 階層開発を行ったり、分散オブジェクト テクノロジーによってプロダクション クオリティのアプリケーションを構築して、基幹業務 アプリケーションと電子商取引 アプリケーションを開発するのに理想的です。

MTS 1.0 は、インフラストラクチャ コードの作成を不要にする、コンポーネント指向のランタイム環境を提供することで、サーバー中心型のアプリケーションの開発方法に革命をもたらしました。**MTS 2.0** は、Microsoft Internet Information Server 4.0(IIS)との強力な統合性、Oracle データベースへのトランザクショナル接続、Microsoft Message Queue Server 1.0(MSMQ)との統合、Microsoft SNA Server 4.0 経由でのメインフレーム分散トランザクション接続、および改善されたパフォーマンスと管理可能性をし、機能を拡張しました。

Microsoft Transaction Server の最新情報については、**http://www.microsoft.com/com** または http://microsoft.com/japan/products/ntserver を参照してください。

Microsoft Corporation • One Microsoft Way • Redmond, WA 98052-6399 • USA

1297

目次

Microsoft Transaction Server
リリース 2.0

トレンドとニーズ

*3 階層*システムの利点はきわめて単純です。つまり、アプリケーションをプレゼンテーション、アプリケーション ロジック、およびデータのセクションに明確に分割することで、スケーラビリティ、再利用可能性、セキュリティ、および管理が改善されるということです。また、3 階層アプローチは、さまざまな重要なアプリケーション開発の問題に、うまく対応することができます。インターネットおよびイントラネット アプリケーションの開発者は、ブラウザを基幹業務 アプリケーションと電子商取引 アプリケーションへのプレゼンテーション インターフェイスとして簡単に使える手段を必要としています。

Visual Basic や Microsoft Visual J++™ Web 開発システムなどの強力なプログラミング ツールを使って、2 階層構造のシステムを構築してきた開発者は、コードの作成方法を根本から変えることなく、よりスケーラブルで柔軟性の高いアプリケーションを構築したいと考えるようになりつつあります。分散コンポーネントをベースにしたアプリケーションの開発者は、セキュリティ、スケーラビリティ、または管理可能性を犠牲にすることなく、デスクトップ マシンからサーバーまで、ネットワーク上のあらゆる場所にオブジェクトを配置し、再利用できるような手段を必要としています。しかし、ほとんどあらゆる人がそう望んでいるにもかかわらず、大部分の企業は、3 階層構造のサーバー中心アーキテクチャのパワーをうまく使いこなせていません。

問題点

3 階層アーキテクチャは、2 階層アプリケーションとは異なり、アプリケーション ロジックをクライアントではなくサーバー上で実行できることから、サーバー中心型のアーキテクチャと呼ばれることがあります。実際、最大限の恩恵を受けるためには、3 階層システムはデスクトップ クライアントまたはブラウザ上でプレゼンテーション ロジックを実行し、中間層のサーバー上でアプリケーション ロジックを実行し、専用のデータベース サーバー上にデータを置かなくてはなりません。これにはいくつかの問題点があります。

- ポピュラーな開発ツールは、プレゼンテーション ロジックとアプリケーション ロジックがクライアントの中で緊密に結び付けられ、ネットワークで接続されたデータベース サーバーに直接にアクセスすることで動作する 2 階層アプリケーションを生成します。Web 環境に 2 階層アプリケーションを導入するということは、プレゼンテーションとアプリケーションのロジックをブラウザ内に置き、インターネット経由で企業データにアクセスするということを意味しています。このため、結果としてアプリケーション ロジックとデータは企業のファイヤウォールの外側に置かれることになります。
- 分散 TP モニタなど、スケーラブルな 3 階層インフラストラクチャを提供する従来のプロダクトは、きわめて複雑であり、サーバー ベースのアプリケーション ロジックの作成に Visual Basic などのツールを使うという点でも厳しい制約がありました。
- ポピュラーな CORBA 準拠のオブジェクト リクエスト ブローカは、真のコンポーネント アーキテクチャのサポートについては基本的な部分しか持っておら

ず、プロダクション クオリティのランタイム インフラストラクチャに欠け、既存のアプリケーションとデータへのアクセスにも制限があります。オブジェクトを中間層のサーバーに配置すると、再利用可能性、セキュリティ、スケーラビリティ、および接続性に問題が生じます。

さらに事態を複雑にするのが、これらの問題に対処するために、サード パーティ ベンダーから高価で複雑なアドイン ソフトウェアを購入しなければならないことが多いという事実です。これによりサード パーティへの依存性が生じ、開発者は複数のベンダーのプロダクトを統合しなければならなくなります。このため、分散アプリケーションの開発と*保守*のコストが増大します。こうした問題はすべてのソフトウェア 開発者に影響を与えますが、アドインのコストと依存性は、外部コストを最小限に抑え、プロダクトの品質を管理しなければならない独立系ソフトウェア ベンダー(ISV)に特に重大な影響を及ぼします。

ソリューションの要件

これらの問題に対処しようとするソリューションは、3 つの技術的な要件を満たさなくてはなりません。

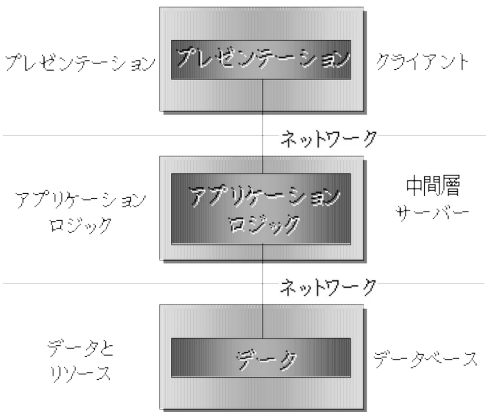
- ブラウザ上にプレゼンテーション コンポーネントを導入し、アプリケーション コンポーネントと企業データをファイヤウォールの背後に安全に配置するということが簡単に行えなくてはなりません。
- Visual Basic や Visual J++などの強力なツールを使って、3 階層アプリケーションのプレゼンテーションおよびアプリケーション ロジック コンポーネントを簡単に構築できなくてはなりません。
- セキュリティ サポート、データベース接続プーリング、アプリケーション管理ファシリティなどのプロダクション クオリティのランタイム サービスが、サーバー ベースのインフラストラクチャの一部として組み込まれている必要があります。また、コンポーネントは完全なトランザクション保護のもとで、既存のアプリケーションとデータにアクセスできなくてはなりません。

テクノロジーの要件に加えて、ソリューションはコスト効果が高く、ベンダーに対する依存性を減らし、システム統合の必要性を最小限に抑えなくてはなりません。

ソリューション: Microsoft Transaction Server

Microsoft Transaction Server(MTS)リリース 2.0 は、Microsoft の Component Object Model(COM)テクノロジーを使って構築されたサーバー中心型アプリケーションの開発と導入を単純化する、Windows NT の重要な機能です。MTS は、Web ベースのインターフェイスを使用し、Microsoft Visual Basic などの強力な開発ツールによるスケーラブルな 3 階層開発を行ったり、分散オブジェクト テクノロジーによってプロダクション クオリティのアプリケーションを構築して、基幹業務 アプリケーションと電子商取引 アプリケーションを開発するのに理想的です。MTS 2.0 は、以下の理由から、Windows NT 上でサーバー中心型のアプリケーションを構築するのに最適なテクノロジーであると言えます。

- MTS 2.0 は、Windows NT 上で COM ベースのアプリケーションを構築し、導入するための最も簡単な手段です。開発者は Visual Basic などのポピュラーな開発ツールを使うことができます。MTS のコンポーネント ベースのアーキテクチャにより、オブジェクトの利用と再利用が単純化され、管理者はグラフィカ



ルなドラッグ アンド ドロップ形式のユーザー インターフェイスを通して管理タスクを実行することができます。

- MTS 2.0 は、自動的なトランザクション サポート、単純かつ強力なロールベースのセキュリティ、ポピュラーなデータベース、メッセージ キューイング製品、およびメインフレーム ベースのアプリケーションへのアクセス、およびデータベース接続プーリングといったパフォーマンス改善機能など、包括的なコンポーネント機能を提供します。
- MTS 2.0 は他の Windows NT 機能と密に統合されています。IIS、Active Server Pages との統合によりインターネット/イントラネット アプリケーション開発が容易になり、Windows NT クラスタリング サービスとの統合によりフェイルオーバーが可能になり、MSMQ との統合により非同期での通信が行え、Windows NT セキュリティ環境との統合によりリソース管理が単純化されます。

3 階層アプリケーション

MTS 1.0 は、インフラストラクチャ コードの作成を不要にする、コンポーネント指向のランタイム環境を提供することで、サーバー中心型アプリケーションの開発方法に革命をもたらしました。MTS 2.0 は、Microsoft Internet Information Server 4.0(IIS)との強力な統合性、Oracle データベースへのトランザクショナル接続、Microsoft Message Queue Server 1.0(MSMQ)との統合、Microsoft SNA Server 4.0 経由でのメインフレーム分散トランザクション接続、および改善されたパフォーマンスと管理可能性を提供することで、この環境をさらに拡張しました。

最も重要なのは、他のベンダーの仕様、計画、および統合性の低いプロダクトとは対照的に、MTS 2.0、IIS 4.0、MSMQ 1.0、および SNA Server 4.0 が今日すでに入手可能であり、統合されているということです。

Microsoft Transaction Server - リリース 2.0 の機能一覧	
機能	説明と利点

シンプル

使いやすさ	
MTS エクスプローラの拡張	MTS エクスプローラでは、パッケージ管理のサポートが改善され、複数項目の選択といった使いやすさを改善する拡張が加えられています。また、MTS エクスプローラは、IIS アプリケーションと、SNA Server 4.0 によって生成されたコンポーネントを MTS パッケージとして管理することができます。
Microsoft Management Console サポート	MTS 2.0 では、MTS エクスプローラ管理インターフェイスを、Microsoft Management Console(MMC)へのスナップインとして実行することができます。MMC との統合により、ユーザーは一貫性のあるルック アンド フィールを持った統合管理環境を使うことができます。

自動的なクライアント インストレーション ユーティリティ	MTS 2.0 は、サーバー上で実行されている MTS ベースのコンポーネントにアクセスする必要があるクライア*包括的*ンの構成プロセスを単純化し、自動化するユーティリティを用意しています。
包括的な機能	
Oracle データベースのためのトランザクショナル ODBC ドライバ	MTS 2.0 には、Oracle バージョン 7.3 以降を実行しているデータベースのための ODBC 3.0 準拠のドライバが付属しています。ODBC 3.0 への準拠により、開発者は、(任意のプラットフォーム上の)Oracle データベースへのアクセスと、SQL Server™や Microsoft Message Queue Server などの他のリソースへのアクセスを、トランザクション作業単位の中で実行して、データの整合性を完全に保護することができます。また、ODBC 3.0 に準拠していることで、MTS は Oracle データベース セッションの接続プーリングを行って、パフォーマンスを改善することができます。
パフォーマンスの改善	MTS 2.0 は、MTS がクライアントからの要求に従って、サーバー ベースのオブジェクトのインスタンスを作成し、実行するのに要する時間を減らす、マルチスレッドのクラス ファクトリを持っています。また、MTS 2.0 はさまざまな内部操作が実行のために必要とするステップの数を減らすことによって、パフォーマンスを改善します。
Windows NT Workstation と Windows 95 のサポート	MTS 2.0 は、Windows NT Workstation および Microsoft Windows® 95 オペレーティング システム上での実行をサポートしているので、企業は MTS アプリケーションのスタンドアロン バージョンを導入することができます。

Windows NT の機能と Microsoft プロダクトとの統合	
IIS との統合	Microsoft Internet Information Server バージョン 4.0 は MTS 2.0 と統合されており、MTS を使ってトランザクション管理などのさまざまなランタイム サービスを実行します。トランザクション サポートにより、IIS Active Server Pages は、完全なデータ整合性を保ちながら、データベース、メインフレーム アプリケーション、およびメッセージ キューにアクセスすることができます。MTS との統合により、IIS はプロセスの分離によって、個々の障害の影響が Web サイトの他の部分に及ぶのを防ぎ、スレッドおよび接続のプーリングなど、改善されたランタイム サービスによってパフォーマンスの改善を図り、コンポーネントをより簡単に管理することができます。
MSMQ との統合	MTS 2.0 は Microsoft Message Queue Server(MSMQ)との統合をサポートしています。MSMQ との統合により、MTS ベースのアプリケーションは、信頼の置ける、非同期通信を行うことができます。送信や受信などの MSMQ 操作は、データの整合性を守るために、MTS トランザクションに自動的に参加します。

Microsoft Transaction Server - リリース 2.0 の機能一覧

機能	説明と利点
Windows NT の機能と Microsoft プロダクトとの統合	
SNA Server 4.0 との統合	MTS 2.0 は Microsoft SNA Server 4.0 を使用して、(たとえば CICS の下で動作している)メインフレーム アプリケーションとそのデータを、MTS が管理しているトランザクションに入れることができます。MTS ベースのアプリケーションは、1 つまたは複数のデータベースを更新し、MSMQ を使ってメッセージの送受信を行い、(VSAM などの)データを更新するメインフレーム アプリケーションを呼び出したりした上で、すべての更新を、メインフレーム上のものも含めて、全体としてコミットまたはアボートさせることができます。また、Microsoft SNA Server 4.0 は、メインフレーム アプリケーションに対する COM ベースのインターフェイスを簡単に生成できるので、MTS プログラミングが単純になります。
Windows NT クラスタリング サービスのサポート	MTS 2.0 はクラスタリングされた Windows NT Server 上での実行をサポートしています。クラスタリングにより、管理者は自動的なフェイル オーバーと、フォールト トレラントな高い可用性を持つ運用が可能なように MTS を構成することができます。

MTS の機能一覧

Microsoft Transaction Server - 機能一覧	
機能	説明と利点

簡単さ

使いやすさ	
単純化された 3 階層アプリケーション開発	MTS は、3 階層インフラストラクチャを、プレゼンテーションとアプリケーション ロジックから完全に分離します。これにより、MTS 開発者は、シングルユーザーの COM コンポーネントの集まりとしてアプリケーションを構築し、これらのコンポーネントを適切な層に導入できるようになります。中間層サーバー上で実行されるコンポーネントは、自動的にマルチユーザー操作をサポートします。
COM コンポーネント サポート	MTS は、Microsoft Visual Basic、Microsoft Visual C++®開発システム、および Microsoft Visual J++を含む、COM DLL を生成できる任意のツールをサポートします。
単純なアプリケーション プログラミング インターフェイス	MTS 開発者は、2 つの新しい API を学ぶだけで、自作のコンポーネントを MTS ランタイム環境に導入することができます。COM API と Win32® API に関する深い知識は不要です。
コンポーネント パッケージのサポート	MTS では、開発者は互いに関連するコンポーネントをパッケージにまとめることができます。パッケージは、複数のコンポーネントを 1 つのグループとして扱い、セキュリティの設定、管理、および導入を行うための手軽な手段として利用できます。
MTS エクスプローラ	MTS エクスプローラは、管理者がパッケージの作成と管理を行い、セキュリティやトランザクションの振る舞いなどのコンポーネント プロパティを構成し、実行中のサーバーの監視と管理を行うために使用する、GUI ベースのシステム管理コンソールです。MTS エクスプローラは、ドラッグ アンド ドロップや複数項目選択など、使いやすさを念頭に置いた高度な機能をサポートしています。
Microsoft Management Console サポート	MTS エクスプローラ管理インターフェイスは、Microsoft Management Console(MMC)へのスナップインとしても動作します。MMC との統合により、ユーザーは一貫性のあるルック アンド フィールを持った統合管理環境を使うことができます。
自動的なクライアント インストレーション ユーティリティ	MTS は、サーバー上で実行されている MTS ベースのコンポーネントにアクセスする必要があるクライアント マシンの構成プロセスを単純化し、自動化するユーティリティを用意しています。

包括的

包括的な機能	
自動トランザクション管理	MTS は、コンポーネントに対して、自動的なトランザクション管理サービスを提供します。開発者と管理者は、特定のコンポーネントが必要とするトランザクション保護のレベルを、単純なプロパティ ページ インターフェイスを使って指定します。実行時に、MTS は 2 フェーズ コミット プロトコルなど、すべての必要なトランザクション管理を自動的に実行します。これにより、開発が大幅に単純化され、コンポーネントの再利用が促進されます。
データベース接続のプーリング	MTS は、データベース リソースへの接続をプールから割り当てます。コンポーネントが接続を必要とするのは、そのコンポーネントがアクティブなときだけです。このため、クライアントをサポートするために必要なデータベース接続の総数を減らすことができ、データベース パフォーマンスが改善されます。

Microsoft Transaction Server - 機能一覧	
機能	説明と利点
包括的な機能(続き)	
Oracle データベースのためのトランザクショナル ODBC ドライバ	MTS には、Oracle バージョン 7.3 以降を実行しているデータベースのための ODBC 3.0 準拠のドライバがあります。ODBC 3.0 への準拠により、開発者は、(任意のプラットフォーム上の)Oracle データベースへのアクセスと、SQL Server や Microsoft Message Queue Server などの他のリソースへのアクセスを、トランザクション作業単位の中で実行して、データの整合性を完全に保護することができます。また、ODBC 3.0 に準拠していることで、MTS は Oracle データベース セッションの接続プーリングを行って、パフォーマンスを改善することができます。
複数のデータベースとリソース マネージャのサポート	MTS 上で実行されているコンポーネントは、複数のデータベースと、メッセージ キューイング システムやメインフレーム アプリケーションなどの他のリソースに、データの整合性を完全に保護しながら同時にアクセスすることができます。MTS は自動的にトランザクションの管理を行い、要求に含まれているすべてのデータベースとその他のリソースが、1 つの単位としてコミットするか、アボートするかのどちらかになるようにします。
クライアント フットプリント必用なし	MTS アプリケーションは、クライアント マシン上にライブラリやランタイム環境を必要としません。
Windows NT Workstation と Windows 95 のサポート	MTS は、Windows NT Workstation および Windows 95 オペレーティング システム上での操作をサポートしているので、企業は MTS アプリケーションのスタンドアロン バージョンを導入することができます。
プロセスの分離	MTS 管理者はパッケージを独自のプロセスの中で実行するように構成し、障害が他のパッケージに影響を及ぼさないようにすることができます。
自動的なスレッド プーリング	クライアントから要求が入ってくると、MTS はコンポーネントに対して、事前に割り当てられたプールから自動的にスレッドを割り当てます。コンポーネントが実行を終了すると、MTS はそのスレッドを解放します。このため、スレッドの作成と削除のオーバーヘッドが軽減し、パフォーマンスが改善されます。

自動的なオブジェクト インスタンス管理	MTS は要求があったときにのみ ***統合性*** ー上にコンポーネントのインスタンスを作成し、コンポーネントが実行を終了した時点でメモリ リソースを解放します。このように、ジャスト イン タイム アクティベーションと、できる限り早く行うデアクティベーションにより、サーバー マシン上での必要メモリが最小限に抑えられます。
共有プロパティ マネージャ	MTS は共有プロパティ マネージャによって、状態情報を保存したり、状態を他のコンポーネントと共有しなければならないコンポーネントのプログラミングを単純化します。

Windows NT の機能と Microsoft プロダクトとの統合	
Microsoft SQL Server 6.5 との統合	Microsoft SQL Server 6.5 は、Microsoft Distributed Transaction Coordinator(DTC)をベースにしたきわめて高速で効率的なトランザクション的統合を MTS アプリケーションに提供する OBDC 3.0 ドライバを持っています。MTS はトランザクション管理にも DTC を使用するので、管理者は同じユーティリティを使って MTS と SQL Server の間のトランザクションを管理することができ、システム管理の作業が単純化されます。

MTS の競合製品の要約

Microsoft Transaction Server - 機能一覧	
機能	説明と利点
Windows NT の機能と Microsoft プロダクトとの統合(続き)	
IIS との統合	Microsoft Internet Information Server バージョン 4.0 は MTS と統合されており、MTS をトランザクション管理などのさまざまなランタイム サービスに使用します。MTS との統合により、IIS の Active Server Pages は、データベース、メインフレーム アプリケーション、およびメッセージ キューに、データの整合性を保護しながらアクセスする MTS コンポーネントを簡単に呼び出せるようになります。MTS との統合により、IIS はプロセスの分離によって、個々の障害の影響が Web サイトの他の部分に及ぶのを防ぎ、スレッドおよび接続のプーリングなどの改善されたランタイム サービスによってパフォーマンスの改善を図り、コンポーネントをより簡単に管理することができます。
MSMQ との統合	MTS は Microsoft Message Queue Server(MSMQ)との統合を提供します。MSMQ との統合により、MTS ベースのアプリケーションは、信頼の置ける、非同期通信を行うことができます。送信や受信などの MSMQ 操作は、データの整合性を守るために、MTS トランザクションに自動的に参加します。
SNA Server 4.0 との統合	MTS は Microsoft SNA Server 4.0 を使って、(たとえば CICS や IMS 上で実行されている)メインフレーム アプリケーションとそのデータを、MTS が管理するトランザクションの中に入れることができます。MTS ベースのアプリケーションは、1 つまたは複数のデータベースを更新し、MSMQ を使ってメッセージの送受信を行い、(VSAM などの)データを更新するメインフレーム アプリケーションを呼び出したりした上で、すべての更新を、メインフレーム上のものも含めて、全体としてコミットまたはアボートさせることができます。また、Microsoft SNA Server 4.0 は、メインフレーム アプリケーションに対する COM ベースのインターフェイスを簡単に生成できるので、MTS プログラミングが単純になります。
Windows NT クラスタリング サービスのサポート	MTS はクラスタリングされた Windows NT Server 上での動作をサポートします。クラスタリングにより、管理者は自動フェイル オーバーと、フォールト トレラントな高い可用性での動作を行うように MTS パッケージを構成することができます。
Windows NT セキュリティとの統合	MTS は、Windows NT セキュリティ環境に統合された、ロールをベースにするセキュリティ メカニズムを備えています。ロールにより、開発者は、コンポーネントの最終的な導入形態を知らなくても、コンポーネントにセキュリティ保護を埋め込むことができます。管理者は、MTS にコンポーネントをインストールする時点で、ロールをユーザーにマップします。

Microsoft Transaction Server - 機能一覧			
比較	MTS	Sun の Enterprise Java Beans*	CORBA 準拠の ORB
使いやすさ			

3 階層アーキテクチャ	Yes	Yes	Yes
COM コンポーネント サポート	Yes	No	厳しい制限あり
単純なアプリケーション プログラミング インターフェイス	Yes	Yes	Yes
コンポーネント パッケージングのサポート	Yes	No	No
GUI ベースのシステム管理	Yes	不明	No
Microsoft Management Console サポート	Yes	No	No
自動的なクライアント インストレーション ユーティリティ	Yes	不明	No
包括的な機能			
自動的なトランザクション管理	Yes	No	No
データベース接続のプーリング	Yes	サード パーティによる	サード パーティによる
複数のデータベースとリソースのサポート	Yes	サード パーティによる	サード パーティによる
クライアント フットプリント必要なし	Yes	No	No
プロセスの分離	Yes	サード パーティによる	No
自動的なスレッド プーリング	Yes	サード パーティによる	No
自動的なオブジェクト インスタンス管理	Yes	不明	手動
共有データ機能	Yes	サード パーティによる	Yes
Windows NT の機能と Microsoft プロダクトとの統合			
SQL Server 6.5 との統合	Yes	No	No
Web サーバーとの統合	Yes	制限あり	制限あり
MSMQ との統合	トランザクション	非トランザクション	非トランザクション
SNA Server 4.0 との統合	トランザクション	非トランザクション	非トランザクション
Windows NT クラスタリング サービスのサポート	統合	不明	制限あり
Windows NT セキュリティとの統合	Yes	不明	制限あり

* Sun は Enterprise Java Beans に関する情報を完全には公開していないので、この評価は本書の出版時点での Microsoft の見解を表しています。

3 階層方法とライセンシングについて

入手方法

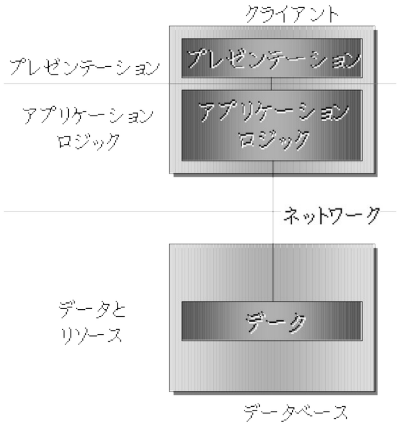
MTS 2.0 は、Microsoft Windows NT Server Enterprise Edition 4.0、Microsoft Windows NT Server Standard Edition 4.0 の機能として、これらのプロダクトに添付されています。さらに、これらのオペレーティング システム プラットフォームの既存のライセンシーは、Windows NT 4.0 Option Pack を通して MTS 2.0 を入手することができます。Option Pack は、CD 形式、または Web ベースのダウンロードで(無償)入手することができます。

ライセンシング

Windows NT Server で実行されている MTS ベースのコンポーネントを呼び出すマシンには、Microsoft Windows NT Server クライアント アクセス ライセンス(CAL)が必要です。すでにプリント サービスやファイル共有などのサービスにアクセスするために CAL を持っているマシンは、新たに CAL を購入しなくても MTS クライアントを使用することができます。Microsoft Windows NT Server ライセンシングの詳細については、http://www.microsoft.com/ntserver および http://www.microsoftcom/japan/products/ntserver を参照してください。

すべてのアプリケーションは、パーソナル コンピュータ上のスプレッドシートからメインフレーム上の支払い勘定システムにいたるまで、プレゼンテーション、アプリケーション ロジック、およびデータの 3 つのコンポーネントを持っています。プレゼンテーション コンポーネントはユーザーとの対話に焦点を当てます。アプリケーション ロジック コンポーネントは、計算を実行し、アプリケーションの流れを決定します。データ コンポーネントは、セッション、計画されたシャットダウン、およびシステムの障害の前後で永続的に存在していなくてはならない情報を管理します。すべてのアプリケーションはこれらのコンポーネントを持っていますが、アプリケーション 開発者は開発コストとスケーラビリティやコンポーネントの再利用可能性といった要素のバランスを取って、さまざまな構成オプションと導入オプションから選択を行うことができます。

2 階層アーキテクチャとその利点



2 階層アプリケーションは、プレゼンテーションとアプリケーション ロジックのコンポーネントをクライアント マシン上に置き、データにはネットワーク接続を通してリモートでアクセスします。クライアント上のアプリケーション コンポーネントは、業界標準の構造化問い合わせ言語(SQL)などの API インターフェイスを通してデータベースのデータを読み込み、アプリケーション ロジックを実行し、更新を行う必要が生じたときにはデータをデータベースに送り返します。2 階層アプリケーションは、通常は適度のユーザー数を持ち(100 人以下)、単一のデータベースとセキュアで高速なネットワーキングを持っている部門規模のアプリケーションに適しています。また、ほとんどのクライアント サーバー開発ツールは 2 階層アプリケーションを自動的に生成するので、今日、ネットワーク ベースのアプリケーションで最も広く使われている形態になっています。

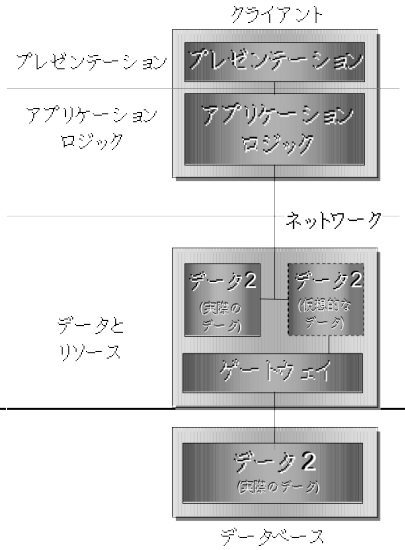
2 階層アーキテクチャの制限

2 階層アプリケーション　　多数のユーザー、マルチデータベース環境、そしてセキュアでないネットワーク環境では、2 階層アプリケーションにはさまざまな制限があります。

- データベースはすべてのアクティブな クライアントに対する接続を保持していなくてはなりません。接続はマシン リソースを消費し、クライアントの数が増えるにつれてパフォーマンスを低下させます。
- 多数のクライアントが同じデータを同時に扱おうとするときに、データベース内で競合が起こることがあります。データベースはデータの破壊を防ぐために、1 つのクライアントが特定のデータ部分(行など)へのアクセスを要求した時点でデータをロックし、他のクライアントが同じデータに同時にアクセスできないようにします。他のクライアントは、データベースがロックを解放するのを待ってから作業を続行します。
- 2 階層システムで使われているセキュリティ モデルは、信頼されている LAN 環境の外部では適切に機能しません。2 階層モデルのセキュリティは、ユーザーのデータに対するアクセスを許可するか、拒否するかという点に焦点を当てています。管理者が(正当な理由で)ユーザーに対してテーブルへのアクセスを許可すると、そのユーザーはデータに対してあらゆる操作を行えるようになります。
- 2 階層アプリケーションでは、アプリケーションは特定のデータベース システムとテーブル フォーマットに緊密に結び付けられているので、アプリケーション ロジックを幅広く再利用するのは困難です。2 階層アプリケーションにおけるロジックの再利用は、一般にアプリケーション間でコードのカット アンド ペーストを行うという意味になります。
- 2 階層システムは、一度に 1 つのデータベース システムにしかアクセスできません。他のデータベース システム、メインフレーム アプリケーション、または他のリソースへのアクセスはゲートウェイを介して行わなくてはならず、このゲートウェイがいくつもの新しい問題をもたらします(後述)。

また、2 階層アプリケーションは、あるレベルまでは優れたスケーラビリティを持っていますが、ロックの競合のせいで、それ以降は急速にスケーラビリティが低下するという点に注意する必要があります。ロックの解決はサーバーの速度には依存しないので、より強力なデータベース マシンをインストールしても、パフォーマンスが大幅に改善されるということはありません。

2.5 階層アーキテクチャとその利点



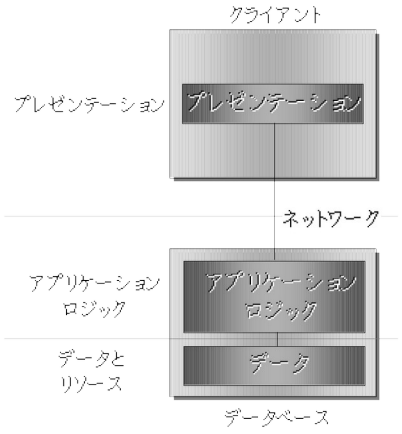
2 階層モデルの限界に直面した企業は、ほとんどの場合、データ ゲートウェイとデータベース ストアド プロシージャを使うようになります。データ ゲートウェイは、他のデータベースに存在するデータ(たとえばメインフレーム上の VSAM データ)を、クライアントが接続されているデータベースに存在しているように見せ掛けることができます。この方法は 2 階層アプリケーションがアクセスできるデータ ソースの数を増やすことができるので、データへのアクセスを拡張することで、複数のアプリケーションを統合する手軽な手段として利用できます。

ストアド プロシージャを使うと、クライアントは、クライアント上ではなく、

ストアド プロシージャを使用する 2.5 階層アプリケーション

データベース サーバー上でリモートに SQL ステートメントのグループを実行することができます。クライアント アプリケーションはストアド プロシージャを名前によって呼び出し、プロシージャにパラメータを渡したり、プロシージャから結果を取り出すことができます。ストアド プロシージャは、クライアントとの間でやり取りしなければならないデータの量を最小限に抑えることでネットワーク トラフィックを軽減し、権限を持つクライアントだけに使用を制限することでセキュリティを改善し、複数のプレゼンテーション コンポーネントが同じストアド プロシージャを呼び出せるようにすることでアプリケーションの再利用性を高めます。また、ストアド プロシージャはスケーラビリティも高めます。一般論として、ストアド プロシージャを使用するアプリケーションは、似たような構成の 2 階層アプリケーションの約 2 倍のユーザーをサポートすることができます。

データ ゲートウェイを使用する 2.5 階層アプリケーション



2 階層アーキテクチャではプレゼンテーション コンポーネントとアプリケーション コンポーネントがクライアント上でグループ化されるのに対し、ストアド プロシージャはアプリケーション ロジック コンポーネントをデータ コンポーネントとともにデータベース サーバー上でグループ化します。これは 2.5 階層アーキテクチャと呼ばれることがあります。ストアド プロシージャは 2 階層アプローチよりもスケーラビリティ、セキュリティ、および再利用可能性の点で優れていますが、真の 3 階層アーキテクチャほどのパワーはないからです。

2.5 階層アーキテクチャの制限

データ ゲートウェイとストアド プロシージャには利点がありますが、重大な制限もいくつか存在します。ストアド プロシージャに関しては、以下のような制限があります。

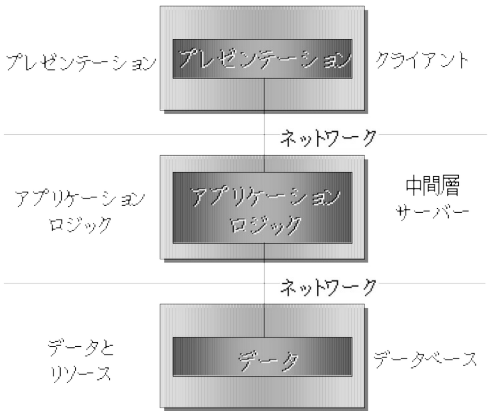
- 個々のストアド プロシージャのインプリメンテーションは、再利用可能性が高くありません。これは、複数インターフェイス サポート、外部に公開されているセキュリティおよびトランザクション プロパティといったコンポーネント機能がないためです。
- ストアド プロシージャ メカニズムはデータベース ベンダーごとに互換性がないので、アプリケーション ロジックを別のデータベース システムで再利用することはできません。
- ストアド プロシージャ アプローチでは、各クライアントについてデータベース接続が必要となり、実行時にデータベース マシンのリソースを消費するので、クライアントの数が増えるにつれてパフォーマンスが低下します。

データ ゲートウェイに関して言えば、他のアプリケーションとの統合はゲートウェイを通してデータのレベルで行わなくてはなりません。データ レベルでの統合には、以下のような問題があります。

- データ ゲートウェイ アーキテクチャが機能するためには、他のすべてのアプリケーションのオーナーが(これにはビジネス パートナーが含まれることがあ

  ります)、自分のデータを(ゲートウェイを通して)、自分の管轄外のアプリケーションに公開する意志を持っていなくてはなりません。また、自分のデータが外部のアプリケーションによって変更された後に戻されて、自分のデータベースに再挿入されることも許さなくてはなりません。
- テーブルのレイアウトや、それに関連するビジネス ルールの変更が、他のいくつものアプリケーションに影響を与える可能性があります。
- いずれかのデータ ソースがリレーショナル データベースでなかった場合(たとえばメインフレーム上の VSAM データ)、ゲートウェイはまずそのデータを SQL テーブルに変換しなければなりません。これによりパフォーマンスが大幅に低下します。

これらの制限のために、ストアド プロシージャはスケーラビリティとセキュリティを高めなければならないが、広範囲に再利用を行う必要はない単一のアプリケーションの中で使うのに適していると言うことができます。データ ゲートウェイは読み取り専用の、扱うデータ量がきわめて少ないアプリケーションに適しています。コンポーネントの再利用と、複数のデータベースやメインフレーム アプリケーションへのアクセスを必要とするアプリケーションでは、これでは明らかに不十分です。



3 階層アーキテクチャ

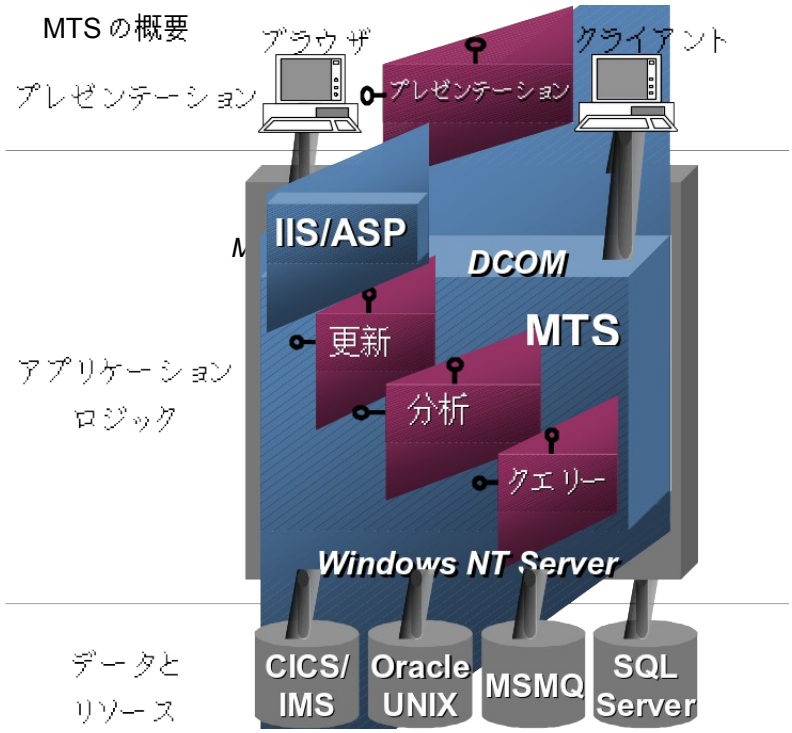
3 階層アーキテクチャでは、プレゼンテーション、アプリケーション ロジック、およびデータ コンポーネントがそれぞれ別のユニットに分離されます。プレゼンテーション コンポーネントはユーザーとの対話を管理し、中間層のコンポーネントを呼び出すことでアプリケーション サービスを要求します。アプリケーション コンポーネントはビジネス ロジックを実行し、SQL からデータベースへ、あるいは LU6.2 からメインフレーム アプリケーションへのネイティブなインターフェイスを使って、データベースやその他のリソースに対して要求を行います。クライアントは要求を完了するために、必要なだけの数のサーバー ベースのコンポーネントを呼び出すことができ、コンポーネントは他のコンポーネントを呼び出すことができるので、再利用可能性が高まります。

3 階層アーキテクチャの利点

3 階層アーキテクチャは、アプリケーション コンポーネントをプレゼンテーション インターフェイスとデータベース インプリメンテーションの両方から独立した形で中間層のサーバー上で実行することができることから、サーバー中心型のアーキテクチャと呼ばれることがあります。個々のユニットを物理的に異なるマシン上で実行しなければならないという制限はありませんが、3 階層システムの利点を最大限に引き出すためには、プレゼンテーション コンポーネントをデスクトップ クライアントまたはブラウザ上に、アプリケーション ロジックを中間層のサーバー上に、そしてデータをデータベース サーバー上に置く必要があります。アプリケーション ロジックがプレゼンテーションとデータから独立しているということに

は、いくつもの利点があります。

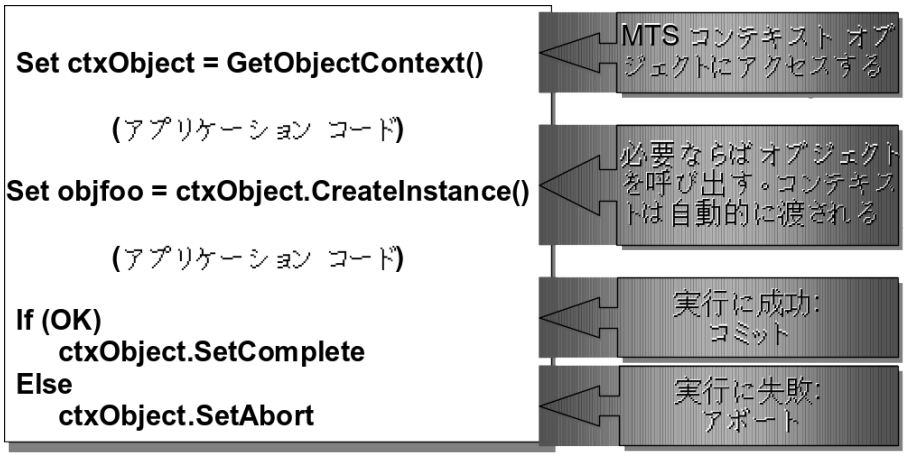
- 開発者は、制限の厳しいストアド プロシージャ言語ではなく、Visual Basic や Visual J++などの強力な開発ツールを使って、再利用可能なアプリケーション コンポーネントを開発することができます。
- 管理者はアプリケーション コンポーネントをレプリケートして、複数のマシン上で同時に実行することができます。これにより、クライアントの負荷を複数のマシンに分散させることができるので、可用性、スケーラビリティ、およびパフォーマンスが改善されます。2.5 階層アーキテクチャでは、ストアド プロシージャを 1 つのデータベースの中で実行しなければならないため、(データレプリケーションではない)アプリケーション コンポーネントのレプリケーションは不可能です。
- アプリケーション コンポーネントはデータベース接続を共有することができます。これにより、データベース サーバーがサポートしなければならないセッションの総数が減り、パフォーマンスが改善されます。2 階層および 2.5 階層のシステムでは、データベースはすべてのユーザーに対して接続を割り当てなくてはなりません。
- 中間層のアプリケーション コンポーネントのセキュリティは、共通のインフラストラクチャを使って、集中的に設定することができます。アクセスはコンポーネント単位で許可または拒否できるので、管理が単純化されます。
- メインフレーム アプリケーションやその他のデータベースなどのリソースへのアクセスは、データ ゲートウェイではなく、ネイティブなプロトコルとアプリケーション インターフェイスを通して行われます。これによりパフォーマンスが改善され、アプリケーションのオーナーが自分のデータへのアクセスを制御できるようになります。



3 階層アーキテクチャにはこのような魅力がありますが、ほとんどの企業は、2 階層および 2.5 階層のアプローチから一歩先に進むことができないでいます。これはおそらく、データベース中心のアプローチをサポートする開発ツールが充実しており、サーバー中心型の開発を支援するツールがほとんど存在していなかったためでしょう。幅広いポピュラーな開発ツールによってサポートされている Microsoft Transaction Server がリリースされたことにより、3 階層開発は、ようやく幅広い企業の手の届くものになりました。

MTS は、Microsoft の Component Object Model(COM)テクノロジーをベースにした 3 階層アプリケーションの開発と導入のための、サーバー中心型の環境を提供します。MTS アプリケーションでは、アプリケーション ロジックはサーバー上の MTS の管理下で実行され、Microsoft の Distributed COM(DCOM)テクノロジーを通して、クライアント上で実行されているプレゼンテーション中心のコンポーネントから呼び出されます。クライアントは、従来のアプリケーションと、Microsoft Internet Information Server(IIS)内で実行される Active Server Page スクリプトの組み合わせとしてインプリメントすることができます。

アプリケーション ロジック コンポーネントは、さまざまな種類のデータベース、Microsoft Message Queue Server(MSMQ)、そして Microsoft SNA Server 4.0 を通して CICS および IMS アプリケーションにアクセスすることができます。データベースとリソースへのアクセスは、ロック管理や接続プーリングといったサービスを自動的に実行する MTS リソース ディスペンサを通して行われます。また、MTS は自動トランザクションもサポートしているので、データとリソースへのアクセスはオール オア ナッシング方式で行われます。



開発者は、COM に準拠するダイナミック リンク ライブラリ(DLL)を生成できる任意のツールまたはプログラミング言語を使って、プレゼンテーション コンポーネントとアプリケーション ロジック コンポーネントを構築することができます。コンポーネントを MTS 環境の中で効率的に動作させるために、開発者はいくつかの単純な規則に従わなくてはなりません。

- コンポーネントは、シンプルな API 呼び出しを行うことで、*MTS* コンテクスト オブジェクトへの参照を作成しなければなりません。参照を作成することで、コンポーネントはトランザクションおよびセキュリティ サポートといった MTS サービスを利用できるようになります。
- コンポーネントの中にステータスを(ローカル変数またはグローバル変数の形で)保存してはなりません。コンポーネントがステータスを保存していると、コンポーネントの実行が終了しても、MTS がリソースのリサイクリングを行えないので、スケーラビリティが低下します。コンポーネントの状態はデータベースまたは MTS の共有プロパティ マネージャ(SPM)に保存し、必要に応じてそこから取り出すようにします。
- コンポーネントが実行に成功して完了したときには、MTS コンテクスト オブジェクトの *SetComplete* メソッドを呼び出さなくてはなりません。これは、トランザクションに関わったすべてのコンポーネントが実行を終了した時点で、このコンポーネントが実行したすべての作業をコミットしてもよいということを MTS に通知する役割を果たしています。また、*SetComplete* の呼び出しは、このコンポーネントが保持しているすべてのリソースをリサイクルしてもよいということを MTS に知らせます。
- コンポーネントが実行に成功しなかった場合には、MTS コンテクスト オブジェクトの *SetAbort* メソッドを呼び出さなくてはなりません。これは、カレント トランザクションをアボートし、このトランザクションに関わったコンポーネントによって加えられたすべての変更をロールバックするように MTS に通知します。また、*SetAbort* の呼び出しは、このコンポーネントが保持しているすべてのリソースをリサイクルしてもよいということを MTS に知らせます。

コンポーネントがこれらの規則に従っていれば、アプリケーションは新たな開発作業を行わなくても、スケーラビリティ、パフォーマンス、および管理の改善といっ

Web ベースのアプリケーションの問題点

た MTS の恩恵を受けることができます。コンポーネントの MTS 環境へのインストールは、コンポーネントの DLL を MTS エクスプローラウィンドウにドラッグ アンド ドロップで移動するという単純な操作で行えます。

MTS は、Microsoft Component Object Model(COM)テクノロジーをベースにした 3 階層アプリケーションの開発と導入のための強力なテクノロジーです。「3 階層コンピューティングについて」のセクションで詳しく説明したように、3 階層アーキテクチャには、スケーラビリティ、セキュリティ、およびアプリケーション ロジックの再利用可能性の改善といった幅広い利点があります。これらの利点はあらゆる業種のさまざまなアプリケーションに適用できるので、MTS の使い方を理解するためには、いくつかの一般的なシナリオを紹介し、MTS がどのような価値を付加するかを見てみるのがよいでしょう。

Web ベースのアプリケーション

Web にはさまざまな役割がありますが、そのうちの 1 つが、ユーザーに対して基幹業務 アプリケーションや電子商取引 アプリケーションを提示するための理想的な手段というものです。ブラウザは、デスクトップ フットプリントと保守コストがゼロに近く、強力でダイナミックなユーザー インターフェイスを手軽に表示することができます。また、インターネット/イントラネット ネットワーキングが世界のほぼあらゆる場所をカバーするようになった現在、ほぼすべてのデスクトップを即座に強力なプレゼンテーションおよびユーザー入力デバイスに変身させることができます。

前に述べたように、プレゼンテーションとユーザー入力は、アプリケーション全体のほんの一部に過ぎません。ユーザーの HTML ベースの入力を受け取り、適切なデータベースまたはその他のリソースに対して要求の処理を行うアプリケーションを呼び出す別の要素が必要となります。Web の場合、これは通常は Web サーバーの役割です。初期の Web サーバーは静的な HTML ページへのアクセスを提供するだけでしたが、今日の大部分の Web サーバーは、ユーザーから入力を受け取り、アプリケーションまたはスクリプトを起動し、応答を返すためのメカニズムを備えています。

Web サーバーからのアプリケーション要求の処理というプロセスは、一見するほど簡単ではありません。

- ほとんどの Web サーバーは、それ自体ではアプリケーション インフラストラクチャと呼べるものをほとんど、あるいはまったく持っていません。アプリケーションやスクリプトを呼び出すことはできますが、データベース アクセスなどの基本的なサービスを提供するのは開発者の責任となっています。
- 優れたインフラストラクチャ サービスを提供するアプリケーション開発テクノロジーのほとんどは、Web サーバーとうまく統合できません。たとえば、Web のコネクションレス型の世界においては、複数のブラウザ要求の間でユーザーに関する情報をどうやって保持するかという問題が生じます。

これらの問題に対する従来のソリューションは、Common Gateway Interface(CGI)のスクリプトやクッキーをベースにしていましたが、これらは基本的なアプリケーションには使えても、基幹業務 アプリケーションや電子商取引 アプリケーションに使うには、スケーラビリティや堅牢性が不十分でした。

MTS と Microsoft Internet Information Server 4.0(IIS)の組み合わせは、Web ベースのフロント エンドを持つ、プロダクション クオリティの 3 階層アプリケーションの構築に理想的です。IIS は、ブラウザに対してビジュアルでインタラクティブなページを提示するための最新の機能と、Active Server Pages(ASP)と呼ばれる強力なアプリケーション メカニズムを持っています。ASP は、標準的な HTML コマンドと、Visual Basic と JavaScript をベースにしたスクリプトの組み合わせを含んでいるテキスト ファイルです。

Web ページのデザイナーは、*.ASP* という拡張子を持つ URL への参照を含んでいる標準的な Web ページを作成することで ASP を使用します。たとえば、ユーザーに対して入力フォームを送付し、返信先の URL として、回答を処理する ASP を指定するということはよく行われます。ユーザーがフォームに入力を行い、回答を送信すると、IIS は URL が.ASP という拡張子を持っていることを確認し、適切な ASP スクリプトを実行します。ASP ファイルの中のスクリプトは、基本的な計算を実行し、ODBC インターフェイスを介してデータベースにアクセスし、ブラウザに返送する HTML ストリームを作成するといった処理を行うことができます。重要なのは、*ASP* が *MTS* 上で実行されているコンポーネントを呼び出すこともできるということです。

ASP が MTS コンポーネントを呼び出すとき、その要求は MTS に対して、普通のクライアントから送られてきた要求と同じ形で入ってきます。コンポーネントは以下の処理を行うことができます。

- アプリケーション ロジックが必要とする任意の計算を実行する。
- ODBC 3.0 準拠のドライバを通して、1 つまたは複数のデータベースにアクセスする(SQL Server 6.5 や Oracle 7.3 など)。
- SNA Server 4.0 を通し、標準的な COM コンポーネント インターフェイスを使ってメインフレーム アプリケーションにアクセスする。
- Microsoft Message Queue Server(MSMQ)を通してメッセージの送受信を行う。Level 8 Systems, Inc.のブリッジ プロダクトを使うと、MTS コンポーネントは MSMQ を使って、IBM の MQSeries プロダクトを使用している既存のアプリケーションにアクセスすることもできます。
- アプリケーション ロジックの他の部分を実行する他のコンポーネントを呼び出す。これによりコンポーネントの再利用が促進されます。実際、企業はビルド済みの市販のコンポーネントを組み合わせてアプリケーションを作ることで、開発に要する時間を短縮することができます。

MTS は自動トランザクションをサポートしているので、MTS コンポーネントまたは *ASP* スクリプトの中で何らかのエラー条件が発生すると、MTS はデータベース(メインフレーム上のものも含む)とメッセージ キューに加えられたすべての変更を

ロールバックします。MTS トランザクションは、プロダクション クオリティのシステムに欠かせない、重要なデータ整合性保護機能を提供します。たとえば、トランザクションによる保護がなければ、請求なしに商品を出荷したり、口座の残高が狂うといった現象が起こりかねません。さらに、MTS は多数のユーザーを処理し、一貫して優れた応答時間を実現する高度なコンポーネント ランタイム環境を提供します。

IIS Active Server Pages と MTS を組み合わせることにより、企業はインタラクティブでダイナミックな Web ベースのインターフェイスを備えた強力な基幹業務 アプリケーションや電子商取引 アプリケーションを、再利用可能なコンポーネントの組み合わせとして構築することができます。ASP だけでも、かなりの量の計算とデータベース アクセスを行うことができますが、Microsoft は ASP をプレゼンテーション管理に、MTS をアプリケーション ロジックに使用することを推奨しています。このアプローチにより、多数のユーザーをサポートし、複雑な計算を実行し、複数のデータベースや、メインフレーム アプリケーションなどの他のリソースにアクセスができる、再利用可能なコンポーネントの組み合わせとしての Web ベースのアプリケーションを構築することができます。

一般的なツールによる 3 階層アプリケーションの構築

Visual Basic や Visual J++などの開発ツールが普及した理由の 1 つが、2 階層アプリケーションが簡単に作れるということでした。データ バウンド コントロールなどの機能により、開発者は強力なユーザー インターフェイス機能を提供すると同時に、データベース アクセスの複雑な部分には触れずに済ませることができます。さらに、COM のサポートにより、モジュール的なコンポーネント ベースのアーキテクチャを使ったアプリケーションを簡単に開発することができます。

しかし、Visual Basic と Visual J++で作られた 2 階層アプリケーションは、他の 2 階層アプリケーションと同じ制限に直面することになります。たとえば、データ バウンド コントロールはデータベースに負荷をかけ、複数のユーザーが同時に使用した場合には競合の問題が発生することがあります。また、(MTS 1.0 以前には)Visual Basic(およびその他の COM 対応ツール)で作られた COM コンポーネントを、クライアント環境の外で使うのは困難でした。

MTS により、開発者は Visual Basic などのツールに関する既存のスキルを使って、高度な 3 階層アプリケーションを構築することができます。開発者は、いくつかの単純な規則に従うことで、プログラムのアプリケーション ロジック コンポーネントを、MTS の管理下にある中間層サーバー上で動作する COM 対応のダイナミック リンク ライブラリ(DLL)として生成することができます。クライアント ベースのプレゼンテーション コンポーネントがサーバー ベースのアプリケーション コンポーネントを呼び出すと、Distributed Component Object Model(DCOM)のランタイムは、自動的に要求を MTS にルーティングします。この環境で MTS を使うことには、以下のような利点があります。

- データベースや、メインフレーム アプリケーションなどの他のリソースに、完全なトランザクションによる保護の下でアクセスすることができます。

分散オブジェクトの問題点

- MTS エクスプローラのドラッグ アンド ドロップ インターフェイスと、自動化されたクライアント インストレーション ユーティリティによって、コンポーネントの導入が単純化されます。
- スレッド プーリング、データベース接続プーリング、およびジャスト イン タイムのオブジェクト インスタンス生成などのスケーラビリティ機能が自動的に使用されます。
- MTS エクスプローラによって、コンポーネント管理を簡単に行うことができます。
- IIS と Active Server Pages から同じオブジェクトを呼び出すことができます。

MTS により、企業はまったく異なるスケールで、強力な開発ツールの恩恵を受けることができます。

業務要件に適合する分散オブジェクト アプリケーション

少し前から、企業はオブジェクト指向のプログラミング ツールと開発方法論の利点に気づいていました。最初のうち、オブジェクトはほとんどの場合、ユーザー インターフェイスの開発などのクライアント中心の用途でその力を発揮していました。これは、オブジェクトをサーバー中心型の用途で利用するための、標準的で簡単な手段がなかったためです。この状況は、Object Management Group(OMG)の Common Object Request Broker Architecture(CORBA)仕様が現れたことによって変わり始めました。CORBA 仕様は、オブジェクトをネットワーク接続上で分散させる方法と、変換やセキュリティなどのさまざまな標準サービスの概要を定めています。この仕様は多数の代表的なベンダーによって支持され、ユーザーは、再利用可能なオブジェクトがネットワークの任意の場所で実行できるという期待に胸を躍らせました。

オブジェクトの作成と導入を 1 回行うだけで、ネットワーク上の任意の場所で再利用できるという発想はきわめて魅力的であり、OMG が CORBA 仕様によって重要な問題点に対する関心を呼び起こしたという事実を否定する人はいません。しかし、それと同時に、CORBA のアプローチにはいくつかの弱点があります。たとえば、CORBA はプロダクトではなく、仕様であるため、異なるベンダーが発売している CORBA 準拠のプロダクトが相互運用性を持つことは稀であり、相互運用が可能だとしても、基本的な形でしか行えません。また、データベース接続のプーリングをどのように行うかといった、オブジェクト インフラストラクチャの重要な問題について、CORBA 仕様は何も定めていません。

さらに、再利用可能性があると主張されているのにもかかわらず、CORBA オブジェクトにはトランザクションやセキュリティの振る舞いといったプロパティを外部から指定する機能がない(つまり、オブジェクトそのものか、そのインターフェイス定義ファイルの中にコーディングしなければならない)ため、実際のアプリケーションでの再利用可能性は限られています。

さらに重要なのは、オブジェクトの振る舞いとサービスのセットの仕様を定めれば、ベンダー各社が協力して、市販の ORB にプラグ アンド プレイで組み込めるパズルの断片を提供してくれるだろうと OMG が勝手に決め込んでいたことです。現実には、このようなことは起こりませんでした。仕様に定められたサービスの多く

フォーカス: 使いやすさ

が、そもそも提供されませんでした。また、プロダクトの形で存在するサービスも、一般にカスタマがかなりの労力を費やしてシステム統合を行わなくてはなりません。

MTS の利点

MTS の利点:

- 複雑なプログラミングなしで、スケーラブルかつ再利用可能なアプリケーションが実現できます。
- ポピュラーな開発ツールを使うことができます。
- 新しい API 呼び出しは 2 つ追加されるだけです。
- 構成と管理が単純化されます。

MTS により、企業は基幹業務アプリケーションで分散オブジェクトを利用することができます。開発者は、COM コンポーネントの集まりとしてアプリケーションを構築し、MTS を使ってこのコンポーネントを導入します。COM と MTS を組み合わせることで、分散オブジェクト アプリケーション開発者はさまざまな恩恵を受けることができます。

- Visual C++、Visual J++、および COM 準拠の DLL を生成できるプロダクトを含む、あらゆるツールによってサポートされている、一貫性のある特定の言語に依存しないオブジェクト モデル。COM 仕様は完全に公開されているので、任意のベンダーが COM 準拠の製品を開発することができます。
- 自動的なスレッド プーリング、データベース接続プーリング、およびオブジェクト インスタンス管理などの高度なランタイム サービスにより、優れたパフォーマンスとスケーラビリティが実現されます。
- プロパティ ページ インターフェイスを通してオブジェクトの外部から管理される、トランザクションと役割ベースのセキュリティが完全にサポートされているため、コンポーネントの再利用可能性を保ちながら、データの整合性と保護が実現されます。
- 完全なトランザクションによる保護を含む、複数のデータベース、メッセージ キューイング システム、およびメインフレームとの相互運用の統合的なサポート。
- MTS エクスプローラによる強力な GUI ベースのシステム管理。

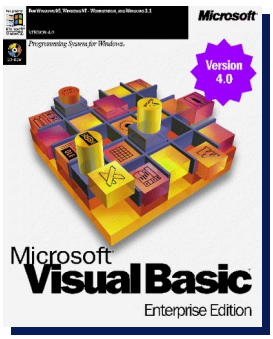
特に重要なのは、他のベンダーから提供されている各種の仕様、計画、そして統合性の劣ったプロダクトとは異なり、MTS 2.0 と、MSMQ 1.0、IIS 4.0、および*SNA Server 4.0 などの関連プロダクトが、いずれも現時点で入手可能であり、統合されているということです。

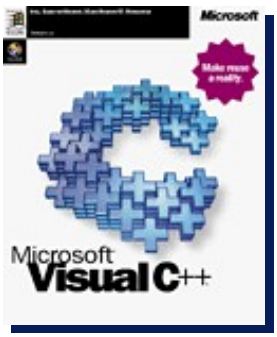
*日本においては、SNA Server 4.0 は 98 年第二四半期に提供を開始する予定です。

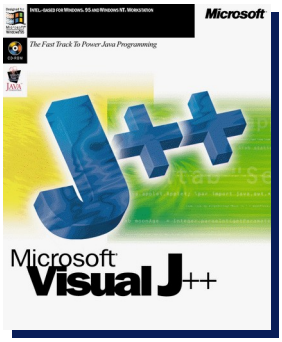
単純化された 3 階層アプリケーション開発

MTS は、3 階層インフラストラクチャを、プレゼンテーションとアプリケーション ロジックから完全に分離します。これにより、MTS 開発者は、シングルユーザーの COM コンポーネントの集まりとしてアプリケーションを構築し、これらのコンポーネントを適切な層に導入できるようになります。中間層サーバー上で動作するコンポーネントは、アプリケーションの状態をコンポーネント変数に保存しないと

いった、いくつかの単純な規則に従っている限り、マルチユーザー動作を自動的にサポートします。







COM コンポーネント サポート

MTS は、Microsoft Visual Basic、Microsoft Visual C++、および Microsoft Visual J++を含む、COM DLL を生成できる任意のツールをサポートします。これらの言語をデスクトップ ソリューションに使用している開発者と企業は、同じ言語をサーバー ソリューションにも使用できます。また、以下に示すように、自社プロダクトで COM をサポートしているツール ベンダーが他にもあります。

- PowerSoft PowerBuilder、および Optima++
- Borland Delphi
- MicroFocus COBOL
- Fujitsu COBOL

また、MTS は、現時点では、開発者が 3 階層アプリケーションを完全に Java だけで構築できる唯一のプロダクトです。これとは対照的に、他のミドルウェア アプローチでは、Java はバックエンドのオブジェクト リクエスト ブローカ(ORB)と TP モニタ アプリケーションの単純なフロントエンドとして使用されます。

シンプルなアプリケーション プログラミング インターフェイス

分散 TP モニタなど、既存の 3 階層開発環境のほとんどでは、開発者は複雑なプログラミング インターフェイスのセットを習得しなくてはなりません。このため、事前にトレーニングに対する大々的な投資を行わなければならず、開発者が生産的な仕事を始めるまでの期間も長くなります。実際、オブジェクト リクエスト ブローカと TP モニタが市場になかなか受け入れられなかった理由の 1 つが、プログラミングの複雑さだったのです。

MTS はデスクトップ 開発者にはおなじみの、既存の COM プログラミング インターフェイスを利用することで、トレーニング コストを削減し、生産性を上げるまでの期間を短縮しました。

- Visual Basic を使用するプログラマは、**Create Object** を使って MTS アプリケーションを呼び出します。
- Visual C++を使用するプログラマは、**CoCreateInstance** を使って MTS アプリケーションを呼び出します。

MTS 開発者は、2 つの新しい API を学ぶだけで、自作のコンポーネントを MTS ランタイム環境に導入することができます。COM API と Win32 API に関する深い知識は必要ありません。

コンポーネント パッケージのサポート

複数のソフトウェア コンポーネントを組み合わせて 1 つのソリューションを作るという方法では、構成と導入の際に問題が生じることがあります。これらの問題に対処するために、MTS はコンポーネント パッケージという概念を使って、開発者が複数の COM コンポーネントを 1 つの管理しやすい単位に簡単にまとめられるようにしています。パッケージの中のコンポーネントは、同じプロセス空間の中で実行され、共通のセキュリティを共有し、1 つの単位として導入することができます。また、パッケージはアプリケーション導入の柔軟性を高める役割も果たします。アプリケーション全体を 1 つのパッケージとして配布した後に、それを複数のパッケージに分割して、パフォーマンス、ロード バランシング、および障害時の分離性を最適化することができます。

MTS エクスプローラ

MTS エクスプローラは、管理者がパッケージの作成と管理を行い、セキュリティやトランザクションの振る舞いといったコンポーネント プロパティを構成し、実行中のサーバーの監視と管理を行うといった操作を簡単に行えるようにした、GUI ベースのシステム管理インターフェイスです。MTS エクスプローラには、その他にも以下のような機能があります。

- コンポーネントを MTS 環境にインストールするときのドラッグ アンド ドロップや、複数のオブジェクトを同時に変更できる複数項目の選択など、使いやすさを考慮した高度な機能があります。
- 導入時の変更を防ぐためにパッケージをロックしたり、実行ステータスを表示したり、パッケージやセキュリティロールの名前を変更するといった運用上の機能があります。
- MTS 上で実行されている IIS アプリケーションや、SNA Server 4.0 によって作成されたパッケージを管理することができます。

また、MTS エクスプローラには、MTS アプリケーションのインストールと構成を自動的に行うアプリケーションを開発するための SDK(ソフトウェア開発キット)が付属しています。

Microsoft Management Console サポート

MTS エクスプローラ管理インターフェイスは、Microsoft Management Console(MMC)へのスナップインとしても動作します。MMC との統合により、ユーザーに対して、一貫性のある外観を持つ総合的な管理環境を提供することができます。MMC インターフェイスを持つ他のソフトウェアには、Windows NT、IIS、および Microsoft BackOffice®ファミリーがあります。

自動的なクライアント インストレーション ユーティリティ

MTS アプリケーションでは、クライアント マシン上で特殊なライブラリやサービ

スを実行する必要はありませんが、DCOM に対してオブジェクトのインスタンスの場所を知らせるために、いくつかのレジストリ項目の構成を行う必要があります。MTS は、クライアントが MTS ベースのコンポーネントの呼び出しを透過的に行えるように、必要なすべての構成操作を自動的に実行するユーティリティが付属しています。

MTS の利点:

- データの整合性を自動的に保護します。
- データベースとアプリケーションのパフォーマンスを改善します。
- 一般的なデータベースをサポートします。
- クライアント管理を単純化します。
- 障害の範囲を制限します。
- コンポーネント間でのデータの共有を単純化します。

自動的なトランザクション管理

MTS は、コンポーネントに対して、自動トランザクション管理サービスを提供します。開発者と管理者は、特定のコンポーネントが必要とするトランザクション保護のレベルを、単純なプロパティ定義を使って指定します。実行時に、MTS は 2 フェーズ コミット プロトコルなど、すべての必要なトランザクション管理を自動的に実行します。これにより、開発が大幅に単純化され、コンポーネントの再利用が促進されます。MTS トランザクションの詳細については、「コンポーネントにおける MTS トランザクションの役割」のセクションを参照してください。

データベース接続プーリング

3 階層アプリケーション環境の中でのリレーショナル データベースの統合には、いくつかの問題があります。

- 個々のクライアントに対して、独立したデータベース接続を構成し、保守しなければならないので、データベース リソースを大量に消費し、スケーラビリティが制限されます。
- 実行時にデータベースへの接続を確立するためのコストが、クライアントがデータベース内で行う作業のコストを大幅に越える可能性があります。
- 複数のデータベース接続をインテリジェントに確立し、管理し、クローズするプロセスには、複雑なプログラミングが必要です。

これらの問題に対処するために、Microsoft Transaction Server は ODBC データベース接続の再利用可能なプールを自動的に作成、管理するリソース ディスペンサを用意しています。アプリケーション コンポーネントが標準的な ODBC 呼び出しを行うと、MTS は使用可能な接続を見つけ出し、それを使用して、コンポーネントが実行を終了した時点で接続をプールに戻します。その結果、スケーラビリティが向上し、コンポーネント 開発者が行うプログラミングは大幅に単純化され、マルチユーザー環境における動作が大幅に高速化されます。

Oracle データベースのためのトランザクショナル ODBC ドライバ

MTS には、Oracle バージョン 7.3 以降を実行しているデータベースのための ODBC

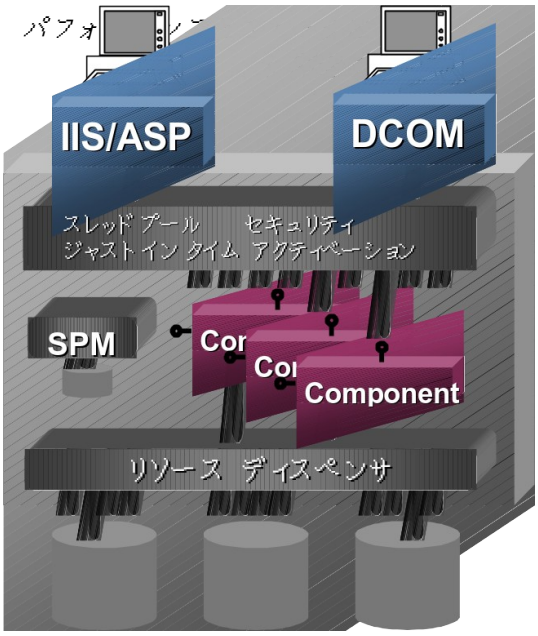
3.0準拠のドライバが付属しています。ODBC 3.0 に準拠していることにより、開発者は(任意のプラットフォーム上の)Oracle データベースと、SQL Server や Microsoft Message Queue Server などの他のリソースへのアクセスを、データの整合性を完全に保ちながら、トランザクション的な作業単位の中で行える MTS アプリケーションを構築することができます。また、ODBC 3.0 に準拠していることにより、MTS は Oracle データベース セッションの接続プーリングを行って、パフォーマンスを改善することができます。

複数のデータベースとリソース マネージャのサポート

MTS 上で実行されるコンポーネントは、複数のデータベースと、メッセージ キューイング システムやメインフレーム アプリケーションなどの他のリソースに、1 つの MTS トランザクションの中からアクセスすることができます。リソースへのアクセスは、1 つのコンポーネントの中から行うこともできますし、1 つのビジネス操作を実行する複数のコンポーネントから行うこともできます。トランザクションの中のすべての作業が完了すると、MTS は各リソースに対して 2 フェーズ コミット プロトコルを自動的に実行し、要求の中に含まれているすべてのデータベースとリソースを全体としてコミットまたはアボートします。

クライアント フットプリント不要

Microsoft Transaction Server では、クライアント プラットフォーム上に特殊なソフトウェアまたはライブラリが存在している必要はありません。ブラウザ ベースのクライアントは IIS を介して MTS コンポーネントにアクセスすることができ、Windows 95 または Windows NT 4.0 Workstation を実行しているクライアントは DCOM を通してコンポーネントを透過的に呼び出します。



Windows NT Workstation と Windows 95 のサポート

MTS は Windows NT Workstation と Windows 95 上での実行をサポートしており、企業は MTS アプリケーションのスタンド アロン バージョンを開発とテストの目的に導入することができます。同じアプリケーションを変更なしで Windows NT Server に導入することができます。

プロセスの分離

多くのランタイム環境では、1 つのスレッドまたはプロセスの中での障害がアプリケーション全体の障害を引き起こし、多数のユーザーに影響を与える可能性があります。一方、MTS 管理者は、特定のパッケージの中での障害が他のパッケージに波及しないように、パッケージを独自の Windows NT プロセスの中で実行するように構成することができます。

自動的なスレッド プーリング

開発者がアプリケーションのスケーラビリティを高めるために使える方法の 1 つが、アプリケーションの実行のために、プロセスではなく Windows NT スレッドを使うというものです。開発者がスレッドを簡単に扱えるように、MTS は自動的なスレッド プーリング メカニズムを用意しています。コンポーネントの実行要求がクライアントから MTS に送られてくると、MTS は自動的にプールから使用可能なスレッドを探し出し、そのスレッド上でコンポーネントを実行した後に、スレッドをプールに戻します。これによりプログラミングが単純化され、スレッドの作成と削除のオーバーヘッドが軽減されて、パフォーマンスが改善されます。

自動的なオブジェクト インスタンス管理

ほとんどの分散オブジェクト環境では、サーバー ベースのオブジェクトのインス

フォーカス: プラットフォームの
統合

MTS の利点:

- コンポーネントは SQL Server 6.5 に素早く効率的にアクセスすることができます。
- 強力な Web ベースのアプリケーションを簡単に開発することができます。
- コンポーネント間の非同期通信を単純化します。
- メインフレームのアプリケーションとデータへのアクセスを単純化します。
- フォールト トレラントな構成をサポートします。
- セキュリティ ポリシーの開発と実施を単純化します。

タン
1 つま
持さ
なっ
しま
で
要に
るた
デル
実行
スを
ティ

MTS
終
はコ
その
ク
には

スは、そのオブジェクトへの参照がたは複数のクライアントによって保れている間はアクティブなままにており、サーバー リソースを消費す。このため、マルチユーザー環境は、サーバー リソースが大量に必なりかねません。この問題に対処すめに、MTS は COM オブジェクト モを拡張し、コンポーネントが実際にされている間だけサーバー リソー消費するジャスト イン タイム アクベーションを取り入れました。

ベースのコンポーネントが、作業がわったことを通知してくると、MTS ンポーネントを非アクティブ化し、リソースをリサイクルします。非アティブ化されているコンポーネント最小限のサーバー リソースしか割り当てられません。コンポーネントが再び呼び出されると、MTS はコンポーネントが必要としているリソースを再獲得して、コンポーネントを再びアクティブ化します。特に重要なのは、クライアントから見て、クライアントがコンポーネントのインスタンスを作成してから、最終的に解放するまで、コンポーネントのインスタンスは 1 つしか存在していないということです。特殊なプログラミングは不要です。

共有プロパティ マネージャ

多くのアプリケーションは、コンポーネントの複数のインスタンスの間でデータを共有する機能を必要とします。マルチユーザー シナリオでは、データを安全に共有するためには、開発者は複数のコンポーネントからの同時アクセスを防ぐために、ロッキング方式をインプリメントしなければなりません。データ共有を簡単にするために、MTS は共有プロパティ マネージャと呼ばれる特殊目的のリソース ディスペンサを用意しています。共有プロパティ マネージャにより、複雑なプログラミングを行わなくても、複数のコンポーネントが同時に、安全に同じデータにアクセスすることができます。

Microsoft SQL Server 6.5 との統合

Microsoft SQL Server 6.5 は、*OLE Transactions* プロトコルと Microsoft Distributed Transaction Coordinator(DTC)をベースにしたきわめて高速で効率的なトランザクション的統合機能を MTS アプリケーションに提供する OBDC 3.0 ドライバを持っています。OLE Transactions は高速で効率的なトランザクション調整インターフェイスを提供します。MTS はトランザクション管理に DTC を使用しているので、管理者は MTS

と SQL Server の間のトランザクションを同じユーティリティを使って管理でき、システム管理が単純化されます。

IIS との統合

Microsoft Internet Information Server バージョン 4.0 は MTS と統合されており、トランザクション管理などのさまざまなランタイム サービスに MTS を使用します。MTS との統合により、IIS の Active Server Pages は、データベース、メインフレーム アプリケーション、およびメッセージ キューに、データの整合性を保護しながらアクセスする MTS コンポーネントを簡単に呼び出せるようになります。また、MTS との統合により、IIS はプロセスの分離を行って、個々の障害が Web サイトの他の部分に影響を及ぼすのを防ぐことができます。また、スレッドや接続のプーリングといった拡張ランタイム サービスにより、パフォーマンスが改善され、コンポーネント管理も簡単になります。

MSMQ との統合

MTS アプリケーションの開発者は、データベースの更新などのアクティビティを含んでいるトランザクション作業単位の中に、MSMQ 操作(送信と受信)を入れることができます。MSMQ 操作はトランザクションの中の他のリソースとともにコミットまたはアボートを行うので、データの整合性が保たれます。たとえば、アプリケーションが 1 つのトランザクションの中でデータベースを更新し、別のアプリケーションにメッセージを送信している場合、アボート条件が生じると、データベースの更新はロールバックします。また、MSMQ は送信操作をキャンセルしてロールバックを行います。MSMQ はトランザクションがコミットするまで、送信操作を完了しません。これにより、受信側がアボートされたトランザクションからのメッセージを受け取るのを防ぐことができます。

MSMQ は、複数の受信操作が 1 つのトランザクションの中で起こったときも、これと同じようなアクションを実行します。トランザクションがアボートすると、MSMQ は受信したメッセージをキューに戻すことによって、受信操作をロールバックします。このメッセージは、それ以降のトランザクションで受信できるようになります。アボート ロジックにキュー操作が含まれていないと、さまざまなタイプの問題が生じる可能性があるため、MSMQ のトランザクション機能はきわめて重要な意味を持っています。たとえば、対応するデータベース操作がロールバックしたときに、受信操作がロールバックしないと、データベース処理が起こらないため、メッセージは実質的に失われてしまいます。

SNA Server 4.0 との統合

Microsoft SNA Server 4.0 により、メインフレーム アプリケーションへの COM ベースのインターフェイスを簡単に生成することができ、MTS プログラミングが単純化されます。開発者は、SNA Server の Transaction Integrator に付属するグラフィカル ツールを使って、メインフレーム アプリケーションの中の COBOL ベースの通信エリア定義を指定します。その後、Transaction Integrator は、適切なメソッドと入出力パラメータを含んでいる COM オブジェクトを自動的に作成します。

MTS アプリケーションがコンポーネントを呼び出すと、SNA Server は適切なメイ

コンポーネントにおける
MTSトランザクションの役割

ンフレーム アプリケーション(たとえば CICS 上で動作しているもの)を自動的に呼び出
は、M
ザク

MTS の利点:

します。このアプリケーション
TS が管理しているカレント トラン
ションに参加します。MTS ベース
のアプリケーションは、1 つまたは複数のデータベースを更新し、MSMQ との間でメッセージの送受信を行い、データ(たとえば VSAM に含まれているもの)を更新するメインフレーム アプリケーションを呼び出します。これらすべての更新は、メインフレーム上のものも含めて、全体としてコミットまたはアボートします。

Windows NT クラスタリング サービスのサポート

MTS はクラスタリングされた Windows NT Server 上での動作をサポートします。クラスタリングにより、管理者は自動的なフェイル オーバーと、フォールト トレラントな高可用性の運用が可能なように MTS を構成することができます。たとえば、MTS パッケージは Windows NT クラスタ環境の中の両方のマシンにインストールすることができます。オンラインになっているマシン上で障害が発生した場合、コンポーネントに対する要求は、Microsoft Cluster Service が制御を自動的に移譲した後は、スタンバイ マシンに送られます。

Windows NT セキュリティとの統合

MTS は、コンポーネント ベースのアプリケーションのために分散セキュリティをインプリメントしています。MTS は Windows NT セキュリティを使って、MTS が Windows NT ドメイン トポロジー上にマップしたユーザーを認証します。

Transaction Server は、宣言的セキュリティとプログラム的セキュリティという、2 つの相補的なモデルを用意しています。宣言的セキュリティは自動的に実施されるもので、コンポーネントがパッケージに追加されるときに指定され、開発者はプログラミングをまったく行う必要がありません。管理者が MTS エクスプローラを使って、どのユーザーまたはユーザー グループがパッケージにアクセスできるのかを宣言します。これにより、サード パーティから購入したビルド済みのコンポーネントであっても、導入単位でのセキュリティを設定することができます。

一方、プログラム的セキュリティでは、開発者は*ロール*を定義し、使用することで、コンポーネントに直接、カスタム アクセス制御を組み込むことができます。たとえば、銀行アプリケーションのロールとして、窓口係とマネージャというものを作成しておき、コンポーネントは特定の操作を実行する前に、カレント ユーザーにマネージャの権限が与えられているかどうかをチェックするというようなことが可能です。管理者は MTS エクスプローラを使って、ユーザーにロールを関連付けることができます。特に重要なのは、コンポーネントそのものには特定のユーザーに関する知識が埋め込まれておらず、Windows NT セキュリティ環境に対して明示的に呼び出しを行う必要がないということです。これは、コンポーネントの広い意味でのセキュリティと再利用可能性を大幅に高めることになります。

トランザクションと分散コンポーネント

トランザクションなしのコンポーネント

ト

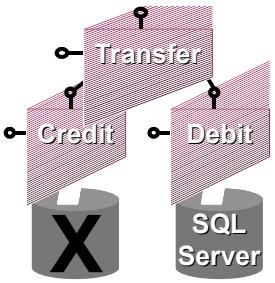
トランザクションは、コンポーネント ソフトウェアからサーバー中心型のアプリケーションを構築する上で、非常に重要なツールとなります。これまで、大部分のアプリケーションは、1 つの開発チームによってモノリシックなアプリケーションとして開発されたもので、トランザクションを適切にデザインするのが比較的簡単でした。しかし、企業がコンポーネント ベースの開発に移行し、他の企業によって開発されたビルド済みのコンポーネントを使うようになるにつれ、トランザクションの保護は保証するのがずっと難しくなっています。

- 個々のアプリケーションにトランザクショナルな整合性を埋め込みます。
- 複数の開発者が作成したコンポーネントを安全に統合することができます。
- 複数のサーバーとデータベース上で、分散アプリケーションを実現することができます。
- MTS トランザクションは開発者に対しては透過的に処理されます。

MTS トランザクションを使用するコンポーネント

単純な例

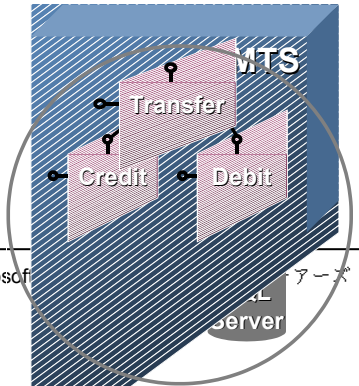
それぞれ異なる開発者が構築した、*Transfer*、*Credit*、および *Debit* の 3 つのコンポーネントから組み立てられた、単純な銀行口座振替アプリケーションを考えてみましょう。



- **Credit** は、当座預金口座データベースに一定の金額を追加します。
- **Debit** は、普通預金口座データベースから一定の残高を引きます。
- **Transfer** は、*Credit* と *Debit* を呼び出すことで、当座預金口座から普通預金口座に一定の金額を移動します。

さて、*Transfer* が *Debit* と *Credit* を呼び出すことによって作業を開始したとしましょう。*Debit* は実行に成功し、データベースから$100 を引きます。しかし、*Credit* は、データベースがコミットに成功しなかったために、実行に失敗します。*Transfer* を開発した開発者が、いずれかのコンポーネントが実行に失敗したときの動作をプログラミングしていない限り、アプリケーションは当座預金口座から$100 を引きますが、対応する普通預金口座に$100 を加えることはしません。

このシナリオでは、アプリケーションのコンポーネントが 3 つしかないので、修正するのも容易かもしれません。しかし、数百のコンポーネントが複数のサーバーを使って動作している典型的なビジネス アプリケーションでは、トランザクションを使わない限り、制御不可能になります。コンポーネント ベースの開発では、これらの問題にコスト効果の高い形で対処するためには、自動的な分散トランザクション インフラストラクチャが唯一の手段となります。



MTS におけるトランザクション

MTS は、トランザクション管理をコンポーネント 開発者に対して透過的に処理し

ます。開発者は、アプリケーション コードの中にトランザクションの開始と終了を指示するステートメントを入れる必要がありません。開発者は、コンポーネントの構築に、複数のツールや言語を使用することさえできます。Transaction Server エクスプローラを使って、コンポーネントのトランザクション処理を行うことを宣言しておけば、MTS はコンポーネントが実行を開始した時点で自動的にトランザクションを開始します。

コンポーネントがデータベース、メッセージ キュー、またはメインフレーム アプリケーションなどのリソースにアクセスすると、MTS はそのリソースを自動的にトランザクションに追加します。コンポーネントが別のコンポーネントを呼び出すと、呼び出された側のコンポーネントも自動的にトランザクションに参加します。トランザクションの中のすべてのコンポーネントが作業を完了すると、MTS は完全な 2 フェーズ コミットを実行し、作業をコミットまたはアボートします。

多くの点で、MTS はまったく新しいプロダクト カテゴリを定義したと言うことができます。MTS は、以下のものを組み合わせたというユニークな特徴を持っています。

- コンポーネント ベースの開発モデル
- トランザクションやセキュリティのサポートなどの、強力なインフラストラクチャ サービス
- メッセージ キューイング、メインフレーム アプリケーション、および Web サーバーの統合

このことを前提とした上で、MTS の競合製品に最も近いものは、CORBA 準拠のオブジェクト リクエスト ブローカ(ORB)と、Sun の Enterprise Java Beans 戦略の 2 つだと言うことができるでしょう。

CORBA 準拠のオブジェクト リクエスト ブローカ

Object Management Group のビジョンは、いまでもほとんどの企業にとって魅力を失っていませんが、CORBA アプローチにはいくつかの弱点があります。

- *CORBA* はプロダクトではありません。CORBA 仕様は、競合する分散オブジェクト テクノロジーを採用している多数の企業が参加した委員会の結果として生まれたものです。合意に達するために、OMG の委員会は、競合するアプローチを採用しているメンバーを満足させられるだけの曖昧さを持った仕様を書かなければならないことがしばしばありました。その結果、異なるベンダーの CORBA 準拠のプロダクトには、ほとんど相互運用性がありません。また、仕様は存在するが、それに準拠するプロダクトを提供しているベンダーがいないというケースもあります。
- オブジェクトのサーバー ベースのインスタンスの作成と破棄の方法、データベース接続などのリソースのプーリングの方法といった重要なランタイムの問題について、CORBA 仕様は何も定めていません。このため、CORBA 準拠のプロダクトは、一般にスケーラビリティの面で大きな問題を抱えています。
- CORBA 仕様は、オブジェクトのセキュリティとトランザクションの要件を外部から管理するといった、重要なコンポーネント指向の機能を細かくはサポートしていません。実際に運用する場合、この点はオブジェクトの再利用を大き

く妨げる要因となっています。

- CORBA準拠のオブジェクト リクエスト ブローカの大部分のベンダーは小規模であり、開発リソースが限られています。このため、これらのベンダーはコアのORB機能の開発に焦点を当てており、トランザクション管理を提供するObject Transaction Service(OTS)などの複雑なサービスはなおざりにしてきました。
- ORBでトランザクションを使用するためには、カスタマはサード パーティのOTSインプリメンテーションを購入しなければならず、このために複雑さとオーバーヘッドが増します。メッセージ キューイング機能とメインフレームベースのアプリケーションへのアクセスも、通常はサード パーティによって提供されます。ほとんどのカスタマは、最終的には、システム統合にかなりの労力を費やさなければなりません。
- ポピュラーなORBのほとんどは、システム管理機能がきわめて貧弱です。

要するに、CORBA準拠のORBは、基本的な分散オブジェクト アプリケーションを構築するための単純で強力な手段ではあるが、プロダクション レベルでの導入に必要なインフラストラクチャは備えていないと言うことができます。

Enterprise Java Beans?

最近になって、Sun MicrosystemsはCORBAのいくつもの欠点に対処することを目的とした、Enterprise Java Beansというイニシアティブを発表しました。CORBAと同様に、Enterprise Java Beansは開発者に対して、再利用可能なコンポーネントを構築し、ネットワークのどの場所でも実行することができると約束しています。しかし、CORBAと同様に、Enterprise Java Beans戦略には重大な弱点があります。

- Enterprise Java Beansはプロダクトではありません。これは、互いに競合する関心を抱いている多数の企業からのインプットの結果として生まれた仕様なのです。
- Enterprise Java Beans仕様は、データベース接続のプーリングといった実際上の問題については、直接には扱っていません。
- Enterprise Java Beans仕様は、オブジェクトのセキュリティやトランザクションの要件の外部からの管理といった、重要なコンポーネント指向の機能を細かくはサポートしていません。
- Microsoftは、企業が複数のベンダーの複数のテクノロジーを組み合わせ、システム統合を行い、トランザクションやメッセージ キューイングといった重要な機能をアプリケーションに組み込まなくてはならないだろうと考えています。

実際、SunがEnterprise Java Beansに関してより詳しい情報を公開するまで、ベンダーはこれに準拠するプロダクトを開発できないでしょうし、カスタマは自分の環境にEnterprise Java Beansが適しているかどうかを評価することもできないでしょう。

インターネット コンピューティングの成長により、サーバー上でのソリューションの導入という需要が爆発的に増大しました。インターネットは、第一段階としては、情報をオンラインで公表し、共有する簡単な手段として成長しましたが、企業

は基幹業務 アプリケーションを Web に対応させて、インターネットとイントラネット上で使用できるようにすれば、さらに大きな利益が引き出せることに気づいています。企業は、単にセールス ブローシャやプロダクト カタログを印刷する代わりに、サーバー上で課金システムとオーダー入力システムを運用し、ユーザーに広く普及しているブラウザから共有ビジネス機能にアクセスさせることができます。

企業はまた、サーバー上で動作する再利用可能なコンポーネントを使って 3 階層アプリケーションを開発する代わりに、単一用途の 2 階層および 2.5 階層アプリケーションを開発していることで、プログラミングの労力を大量に無駄にしていることにも気づきつつあります。アプリケーション ロジックをコンポーネントの形で 1 回だけ開発し、コンポーネントをサーバー上で実行することで、さまざまな種類のプレゼンテーション インターフェイスをサポートすることができます。また、コンポーネントは他のコンポーネントを簡単に呼び出すことができるので、プログラミング作業の重複も最小限に抑えられます。

これまで、このサーバー中心型のビジョンには 1 つの問題がありました。ツールとインフラストラクチャによるサポートがなかったため、サーバー ベースのコンポーネントの構築と導入は、単一用途のアプリケーションの構築よりもはるかに難しかったのです。サーバー ベースのアプリケーションでは、システムの障害やデータの損傷の影響が、1 人のユーザーだけでなくビジネス全体に影響を与え得るので、デスクトップ アプリケーションよりも高い信頼性が必要となります。このため、開発にコストがかかり、保守の困難な高度なインフラストラクチャが必須となります。さまざまな推計では、平均的なサーバー中心型の開発プロジェクトにおいて、インフラストラクチャの開発は 30%から 40%の割合を占めています。サーバー ベースのアプリケーションが価値を持つためには、開発者が特殊なスキルを持つ必要がなく、高価で複雑なインフラストラクチャ コードを構築しなくても、デスクトップ アプリケーションと同じほど簡単に導入し、保守できるようになっていなくてはなりません。

Microsoft が Microsoft Transaction Server を開発し、これを Microsoft Windows NT と Microsoft Windows 95 に添付したのは、このような目標に沿ってのことでした。MTS は、デスクトップ アプリケーションの柔軟性と低いコストを、通常はハイエンドのメインフレーム システムにのみ見られるミッション クリティカルなトランザクション処理機能と組み合わせています。また、Microsoft Transaction Server は Microsoft の Component Object Model をベースにしているので、幅広い開発者が再トレーニングを行わなくてもアクセスできます。

関連情報

Windows NT Server の最新の情報については、World Wide Web サイト(http://www.microsoft.com/com)と、Microsoft Network の Windows NT Server フォーラム(GO WORD:MSNTS)を参照してください。