三層クライアント／サーバ・システムにおけるミドルウェアの開発
（2）データベース操作のカプセル化

4F－4

松本 諭　鈴木 勇至　成田 秀明　関根 徹
松下電器産業(株) マルチメディアシステム研究所

1. はじめに

リソースマネジャ(データベース)、サーバ、クライアント（端末）からなる三層クライアント/サーバ・システムを採用したミドルウェア[1]内の中間層である、サーバで使用するデータベース操作モジュールを開発した。

従来、異なる種類の端末からデータベース操作を行なう場合、各々のアプリケーションでデータベース操作を行なうコードを記述しなければならなかった。そのため、それらは端末依存もしくはデータベース依存の強いものとなり、新しい端末への対応や、データベース変更などの対応が困難であった。

本モジュールは上記問題点を解決するものである。本稿では本モジュールについて、機能モデル、実装方式、処理の流れを説明する。

2. 機能モデル

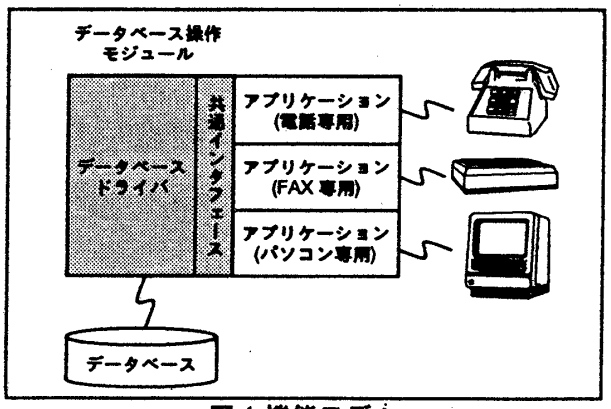

図1 機能モデル

本モジュールの機能モデルを図 1 に示す。本モジュールは、共通インタフェースとデータベースドライバに分かれている。

共通インタフェースは、様々な端末専用のアプリケーションから共通的にデータベース操作ができるよう定義されたインタフェース群である。個々のインタフェースは、システムが提供すべきサービスを検討した上で、サービス単位で定義される。インタフェース定義を端末やデータベースに依存しないものにすることで、端末依存の部分(アプリケーション)とデータベース依存の部分(本モジュールのデータベースドライバ)を分けることが可能となった。

データベースドライバは共通インタフェースの要件を満たすようデータベース操作を実際に行なう部分である。共通インタフェースのそれぞれに一意に対応したデータベース操作の集合である。

アプリケーションは共通インタフェースを通してのみデータベースドライバにアクセス可能であり、本モジュールによりデータベース操作のカプセル化(隠蔽)がなされている。また、異なる種類の端末を操作するアプリケーションが本モジュールを共通的に使用することにより、開発コストの軽減や品質の向上が期待できる。

3. 実装方式

開発したミドルウェアでは、本モジュールを適用するにあたり、共通インタフェースとデータベースドライバを図 2 に示すように、それぞれ別のプロセス上に実装している。

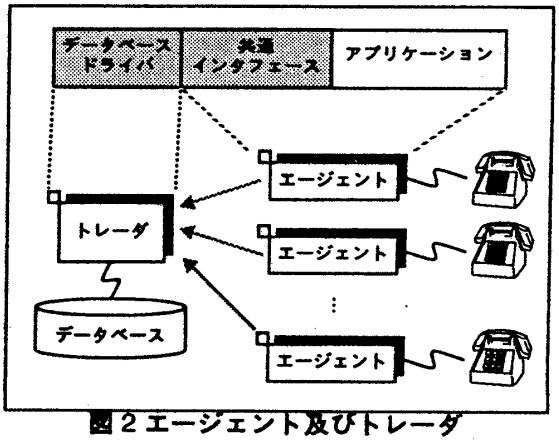

図2 エージェント及びトレーダ

ここでは、アプリケーション及び共通インタフェースが実行され端末制御を行なうプロセスを

Developing a Middleware in the Three Tier Client-Server System
(2) Capsulization of Database Manipulations
Satoru MATSUMOTO, Yuji SUZUKI, Hideaki NARITA, Toru SEKINE
Matsushita Electric Industrial Co., Ltd.

エージェント、データベースドライバが実行されデータベース制御を行なうプロセスをトレーダと呼ぶ。エージェントは起動される端末に一つずつ割り当てられるため、生成される数が多く、しかも変動する。一方トレーダはデータベース一つに対し、システムの起動時にシステム管理者によって決められた個数だけ生成される。

エージェントとトレーダ、というように端末側とデータベース操作側でプロセスを分けたのは、複数のエージェントから一つのトレーダにアクセスさせることで、計算機資源を有効に使用することができるためである。

一般に端末側では、人間がインタラクティブに端末を操作するため、データベース操作を必要とするのはごくわずかな時間であり、上記により、端末が起動している間データベースと接続したままという無駄を省いている。

4. 処理の流れ

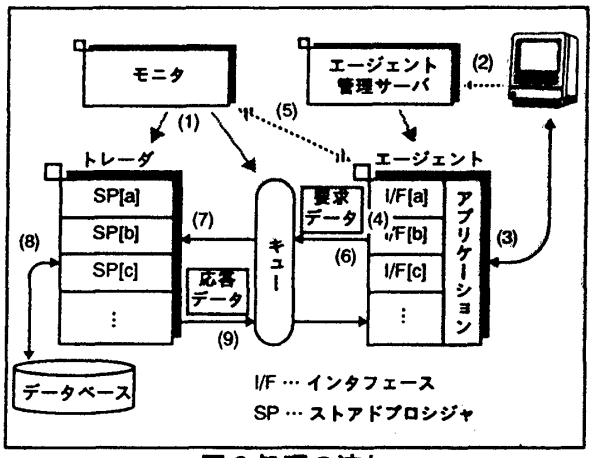

図 3 処理の流れ

次に、エージェントとトレーダを中心とした処理の流れについて、図 3 を用いて説明する。以降の番号は図中の番号に対応している。

システムの起動

(1) モニタと呼ばれるシステム全体の運用管理等を行なうプロセスにより、トレーダとキューが生成される。キューとは、エージェントとトレーダ間でメッセージ交換を行なうための手段である。

端末内の処理

(2) 端末はエージェント管理サーバにエージェント生成を依頼し、以後は当該エージェントと通信を行なう。

エージェント内の処理

(3) アプリケーションは、端末からの要求を受け取り、それに応じて共通インタフェース中から適切なものを選択し、引数を渡す。

(4) インタフェース内部では、その引数から要求データを作成する。要求データはトレーダ内部でも扱われるため、その構造はエージェント、トレーダ両者で合意のとれた構造となっている。また、インタフェースごとに異なった構造となるため、受取側であるトレーダが識別できるよう、内部に要求の種類を表すタグが付加されている。これらは後述する応答データについても同様である。

(5) モニタのネームサービス機能を用いて、要求先のトレーダの位置を得る。

(6) 要求データをエンキューする。

トレーダ内の処理

(7) 要求データをデキューする

(8) 要求データ内部のタグに対応したデータベース操作を行なう。データベース操作は今回の実装では Oracle7 のストアドプロシジャを用いて行なっており、これはエージェント内のインタフェースと一意に対応している。

(9) データベース操作の結果は応答データとして要求データが送られてきた時とは逆の道筋をたどって端末に返される。

5. まとめ

三層クライアント/ サーバ・システム内の中間層であるサーバで使用するデータベース操作モジュールを開発した。本モジュールを使用することにより、

- 端末のアプリケーションに対して、データベース操作のカプセル化 (隠蔽) が可能となる。
- 異なる端末からでも共通に使用可能なため、開発コストの軽減や品質の向上が図れる。
- トレーダというデータベース操作専門のプロセスを用意し、これを複数のエージェントで共有することにより、計算機資源の節約が図れ、また計算機の負担が軽減される。

という効果が上げられた。

[参考文献]

[1] 鈴木 他、「三層クライアント／サーバ・システムにおけるミドルウェアの開発」、情報処理学会第 51 回全国大会講演論文集