—ノート—

高岡短期大学紀要　第11巻　平成10年3月
Bull. Takaoka National College, Vol.11, March 1998

分散オブジェクトによるWWWとデータベースの連携

—管理者インタフェースの改善—

米川　覚

(平成9年10月31日受理)

要　旨

本稿では，CGI方式によるWWWとデータベースの連携システムにおける管理者インタフェースの問題点を検討し，その改善策として取り組んだ3層モデルへの移行および複数管理者間での情報の共有方法について述べる。インタラクティブな管理者インタフェースの提供を目的とする3層モデルでは，WWWクライアントとアプリケーションサーバがORBを介して通信し，リモートのアプリケーションサーバに実装したJavaオブジェクトによりデータベースを処理する。さらに複数の管理者間での情報共有機能を提供するために，データベースの更新状況を自動的に管理者に配布する機能を持つオブジェクトをアプリケーションサーバに配置し，プログラム自動配信システム（Castanet）との連携を図る。これらの実装により，管理者インタフェースが充実しWWWコンテンツ管理の効率化が達成された。また情報共有機能は，一般のホームページ管理や他のWWWアプリケーションの運用にも適用可能である。

キーワード

分散オブジェクト，3層モデル，ORB，IDL，プッシュ技術，Castanet

1　はじめに

現在，企業や大学においてインターネット技術を活用し，組織内部のコミュニケーションの活性化や業務の効率化，意思決定の迅速化等を目的とするイントラネット（組織内情報システム）の構築が本格化している。[1)-3)]イントラネットでは，在庫管理や商品・顧客情報の照会，ワークフローといった業務の効率化を目指したサービスを実現するため，メインフレームやデータベースサーバ，グループウェアサーバといった基幹業務を処理するサーバ群とWWWとを連携し，サーバ群に蓄積された情報をリアルタイムに更新するシステムが用いられる。こうした連携機能の実現にあたっては，CGIを中心とする汎用APIを用いた連携ツールの開発や，メーカー独自のAPI機能（NSAPI，ISAPI等）を付加したWWWサーバの導入が行われ，連携システムの構成としては，クライアントとバックエンドサーバの間に処理手続きを集約したミドルウエアをおく3層モデル[4)]が提唱されている。また最近では，CORBA（Common Object Request broker Architecture）やDCOM（Distributed Component Object Model）等の分散オブジェクト技術の進歩により，3層モデルのファ

産業情報学科

ンクション層として分散オブジェクトを実装し，シームレスな分散処理環境を構築する方式が注目されている。[5)-6)]

筆者は，これまでに，本学が所蔵する工芸品データベースの構築とWWWでの公開にあたり，一般ユーザの利便性を考慮したCGI方式によるWWWとデータベースの連携ツールを作成した。[7)]この連携ツールは，ユーザによるデータベースの検索や管理者によるデータの登録・更新の他，一般ユーザからの情報提供によりデータベースを更新する機能などを持つ。データベースと連携するWWWアプリケーションの運用において，管理者は常にデータベースの最新の状況を把握し，提供するWWWコンテンツとの整合性を維持することが求められる。また分散した複数の管理者や不特定多数による情報の更新が行われる場合には，データベースの更新状況を管理者間で共有する仕組みが必要である。既存のCGI方式による連携ツールは，管理者によるデータの追加や更新といったデータ管理の機能は充分であるが，管理者に提供されるインタフェースは，1セッション1プロセスで処理されるHTTPの特性から1画面単位での冗長なものである。また，単独の管理者による運用を前提としているため，アプリケーションの管理情報を共有する機能は持たない。

WWWアプリケーションが増大し，多様なサービスが実現されるイントラネット環境においては，ユーザへの情報提供機能の充実はもとより，サービスの運用を円滑に維持する管理者側の運用管理機能の整備も重要である。管理対象とするWWWコンテンツやデータベースの更新情報の共有は，アプリケーションの複数管理者による分散管理を可能とし，管理者の作業負担の軽減と柔軟な運用を実現するために有用な機能である。

今回，筆者は，こうした管理者間での情報の共有による管理者インタフェースの充実を目的として，既存連携ツールに情報共有機能を付加し，その有用性について検討した。設計にあたっては，既存のCGI方式では，スクリプト間の複雑な連携と管理情報を蓄積する独自のアプリケーションサーバの構築が必要となるため，開発の効率化と拡張性を考慮し，Java APIとORB（Object Request Broker）を活用した分散オブジェクトによる3層モデルとして管理用ツールを実装し，情報共有機能の実現と同時にファンクション層の改善による管理者インタフェースの操作性向上を図った。

以下，次章は既存連携ツールの概要とその管理者インタフェースの問題点，改善方法について述べた後，3章で分散オブジェクトによる3層モデルへの移行，4章でプッシュ技術による情報の共有，5章で管理用ツールの構成，6章で情報共有の仕組み，7章で管理用ツールの特徴と応用について述べる。

2 既存連携ツールの問題点と改善方法

2.1 CGI方式の特徴

WWWサーバから起動するプログラムによって外部サーバとの連携をはかるCGI方式は，HTMLでの簡易な起動設定とPerlやシェルスクリプトといった扱いやすいプログラム言語によって実現できるため，WWWとデータベースの連携機能の実装には最も広く用いられている。多様なプラットフォームとバージョンが存在するWWWクライアント（ブラウザ）に対して共通のサービスを提供できることが，その最も有用な点である。既存連携ツールでは，本学が所蔵する産業工芸資料の情報を，幅広く一般のユーザに提供することを目的としたものであるため，クライアント側はHTMLのみで実現可能なGUIのみを作成し，特定のWWWブラウザに依存しない方式を採用した。また，データベースとの接続は，一般的なソケットインタフェースを利用している（図1）。

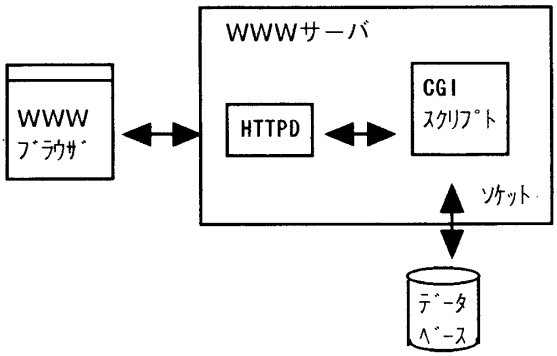

図1　CGI方式による既存連携ツールの構成

2.2 WWWコンテンツの提供方式

データベースとWWWの連携におけるWWWコンテンツの提供には，以下の2つの方式が考えられる（図2）。

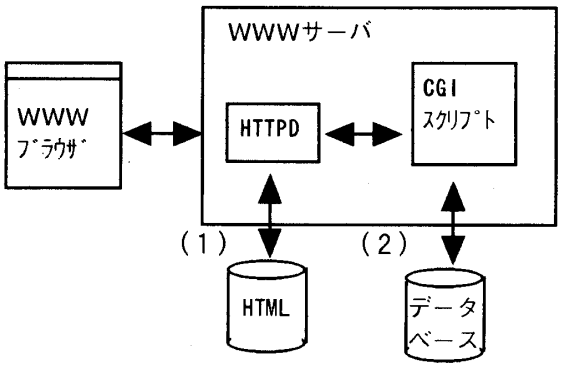

図2　WWWコンテンツの提供方式

(1) データベースの更新にともない，固定的なHTML文書を作成する。

(2) データベースの情報を仮想ドキュメントとして提供する。

データベースの情報更新間隔が長い場合（1時間，1日など）では，(1)の方式により，データベースの内容を反映したHTML文書を作成しておくことで，データベースサーバの負荷が軽減される。一方，OLTP (OnLine Transaction Processing)のようにデータベースの更新が頻繁に行われ，提供されるWWWコンテンツの寿命が短い場合には，(2)の方式が有効である。1回のトランザクションごとにHTML文書を作成することは，WWWサーバへの負荷となり処理効率が低下する。

既存の連携ツールでは，対象とした情報の更新間隔が長いため，キーワードでの情報検索で提供するコンテンツを除いては，(1)の方式を採用した。管理者にはデータベースの更新と共に，作成されるWWWコンテンツの確認を行う作業が必要とされる（図3）。

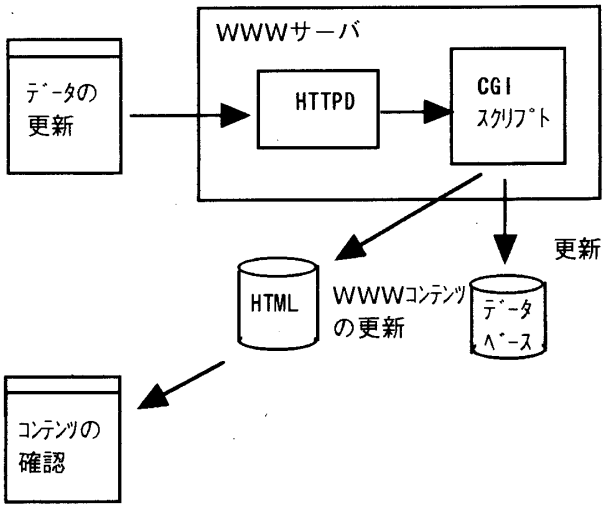

図3　データベースの更新とWWWコンテンツの確認

2.3 管理者インタフェースの課題

CGI方式によるデータベースとの連携では，セッションの確立やリソースへの負荷といった面での問題点が指摘されているが，複雑なトランザクション処理を行うシステムへの適用を除けば，ユーザに提供される情報検索機能とインタフェースは十分に実用性を備えたものである。しかし，データベースへの情報の追加や更新機能を必要とする管理者のインタフェースでは，データの更新という1つの作業の中に「データの入力」と「更新の確認」という2つの要素が含まれるため，効率的な連続作業を行える操作性が求められる。CGI方式ではページ単位でのリクエストが行われるため，更新するデータ量に比例してWWWクライアントとWWWサーバ，データベースの3者間での情報転送が頻繁となり，トラフィックの増加と冗長なページの切り替えが発生す

る。

こうしたインタフェースの問題は，HTMLのフレーム機能を用いたウインドウの分割などにより多少は改善されるが，トラフィックの増加は，WWWサーバをゲートウェイとするCGI方式特有の問題である。柔軟なインタフェースの提供とトラフィックの軽減を同時に満たすためには，WWWサーバをゲートウエイ化せず，処理手続きをおいたクライアントから直接あるいはミドルウエアを介してデータベースと接続する３層モデルでのシステム構築が必要となる。

2.4 管理情報の共有

掲示板やスケジュール管理などのWWWアプリケーションでは，ユーザの入力に伴い複数のWWWコンテンツが更新されるものがあり，既存連携ツールでは，新たな資料情報の追加により一覧情報と資料詳細情報の２つのコンテンツが更新・作成される。管理者がWWWコンテンツの更新を確認する方法の１つは，自らがユーザの立場となり該当するコンテンツにアクセスすることであるが，データベース内の１つのデータから派生するコンテンツの種類が多くなるに従い，確認作業の工程も増え，それらを取り扱うインタフェースは複雑化する。運用作業を効率化するため，データベース更新の後，作成されたWWWコンテンツをリアルタイムに管理者に提示する方式が考えられるが，これは単独の管理者による運用を前提にしたインタフェースであり，複数の管理者が共同して作業を行うような環境では，データベースの更新情報を管理者間で共有する必要があるため，分散管理の機能としては不充分である。さらに，不特定多数からの情報をもってデータベースの更新が行われるシステムでは，更新情報を管理者に自動的に通知し，その内容を簡易に確認しうる機能が必要となる。

3 分散オブジェクトによる３層モデルへの移行

2.3で述べた改善方法を具体化するため，既存のCGI方式を分散オブジェクトを利用した３層モデルに移行する。以下，移行にあたっての留意点と３層化の設計方針を示す。

3.1 ３層モデル

従来の２層クライアントサーバモデルの欠点を補い，WWWと多様なサーバ群の連携を可能とする３層モデルは，プレゼンテーション層，ファンクション層，データ層の３つにシステムの機能を分離し，ファンクション層のミドルウェアに主要な処理手続きをおく(図４)。

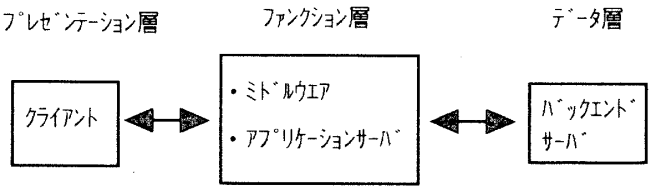

図４ ３階層モデル

プレゼンテーション層は，WWWブラウザ上にGUIを提供するユーザインタフェースを担う部分であり，データ層にはデータベースサーバやグループウェアサーバなどを配置し，ファンクション層のリクエストにより情報を提供する。こうした３階層の構造では，処理手続きがファンクション層中心に記述できるため，プレゼンテーション層の設計が単純となり，システム機能変更の影響が全体に及ぶことがなく，生産性や保守性が向上するといったメリットが得られる。

3.2 WWWサーバのファンクション層からの分離

システムの構造から見れば，既存連携ツールのCGI方式はWWWサーバとCGIモジュールがファンクション層として機能する３層

モデルとして捉えられる。クライアントではHTMLでのGUIに加え，Java Script等のスクリプト言語による処理手続きの実装も可能であるが，プレゼンテーション層の機能としては十分とはいえない。また，WWWサーバをゲートウェイ化することにより，データベースへのアクセス効率やシステムの拡張性の低下を招く。こうした観点から，管理用ツールの設計にあたってはファンクション層からWWWサーバを分離し，インタラクティブな操作性を提供するJavaアプレットによりプレゼンテーション層を設計し，WWWサーバの役割をプレゼンテーション機能を実装するモジュールの提供のみに限定する（図5）。

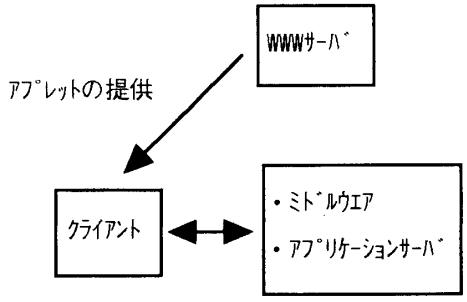

図5　WWWサーバの分離

3.3 ファンクション層の設計

ファンクション層では，プレゼンテーション層のJavaアプレットとデータ層のバックエンドサーバに対して柔軟な通信が行える機能が必要である。データベースサーバと直接に接続する2層構造化も可能であるが，アプレットのセキュリティ制限によりシステム構築の制約が大きい。また，Javaアプレットやデータベースサーバとの接続には，一般的なソケットの採用も考えられるが，独自のプロトコルの設計が必要となり，処理系も複雑化する。

3.4 Java ORB

プレゼンテーション層との親和性を保ちながらデータ層とのシームレスな通信を確保するため，ファンクション層にはORBを介してJavaアプレットと接続する分散オブジェクトをおき，将来的な相互運用性を考慮しORB間の共通通信プロトコルであるIIOP(Internet InterOrb Protocol)[8]を利用する。また，データベースサーバとの接続には，対象とするデータベースの相違を吸収するため，汎用のデータベースアクセス手法であるJDBC(Java Database Connectivity)[9]を採用する。

図6に分散オブジェクトによる3層構成を示す。

ORBとしては，現在のところアルファリリースであるが，JDK1.2に含まれる予定のJava IDL(Java Interface Definition Language)[10]を利用する。Java IDLは，OMG仕様のIDLをJava言語にマッピングする機能を持つフリーのORBであり，ORBコア（ORBのカーネル機能）を含み，IIOPプロトコルのversion1.0をサポートする。

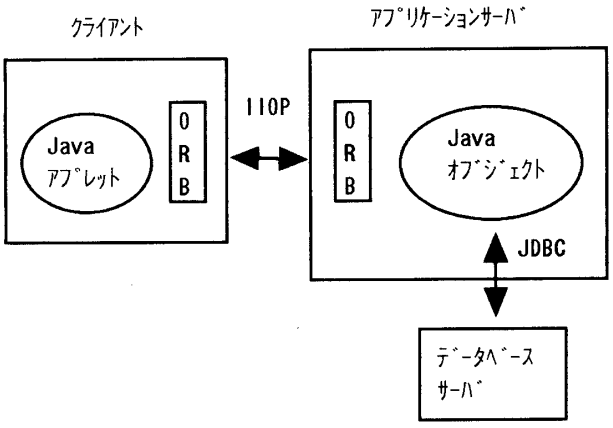

図6　分散オブジェクトによる3層構成

3.5 Java IDL

Java IDLは，CORBAに準拠したインターフェースを記述するOMGIDLをサポートしている。IDLでは，サーバオブジェクトの名前や型，インスタンス変数，メソッドなどをIDLファイルに定義し，付属のidltojavaコンパイラによってクライアント側のスタブであるプロキシとサーバ側のスタブであるスケルトンを自

動的に作成する。これらのモジュールが，クライアントおよびサーバアプリケーションとORBを結び付け，シームレスなオブジェクト呼び出しを行う。(図7)。

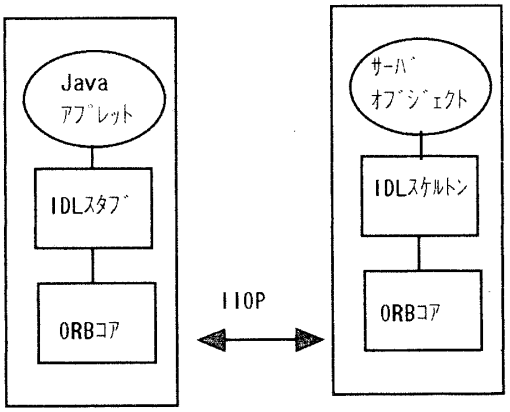

図7 ORBによる通信

4 プッシュ技術による情報の共有

2.4で述べた管理者間の情報共有機能実現にあたっては，データベースの更新という1つの処理に対して，その処理の結果が複数の管理者に自動的に伝わり，送られた情報を連携ツールのインタフェースであるWWWクライアントで扱えることが望ましい。この2つの要件を満たすため，プッシュ技術を搭載したプログラム自動配信システムとの連携を試みる。

4.1 プッシュ技術

現在，ユーザが主導的にWWWの情報にアクセスするプル (Pull) 型のWWWサービスに加え，サーバーからユーザに対して自動的に情報が配信されるプッシュ(Push)型のサービスが注目されている。PointCast[11]やNews Offline[12]などのプッシュ型ソフトウエアの利用により，最新の情報 (ニュースや政治経済，スポーツに関する情報など) がWWWブラウザやスクリーンセーバにリアルタイムに提供される。こうした情報提供の仕組みは，情報の自動配信機能を含め，特定のグループ間の情報共有にも十分に活用できる。プッシュ技術の中で最も将来性のあるシステムとして評価されているCastanet[13)-14)]と呼ばれるシステムは，HTML文書や画像ファイルに加えJavaで作成したアプリケーションの配布と更新を可能とするものであり，その応用範囲は広く，実用的な利用形態が模索されている段階でもある。今回は，3層モデルとCastanetを連携するJavaオブジェクトをファンクション層のアプリケーションサーバに実装し，情報共有の機能を持たせる。

4.2 Castanetシステムの仕組み

Castenetシステムは，コンテンツを管理し配送を行うトランスミッタ (transmitter) とコンテンツを登録するパブリッシャ(publisher)，クライアント側でコンテンツの受信を行うチューナー (tuner) から構成される (図8)。

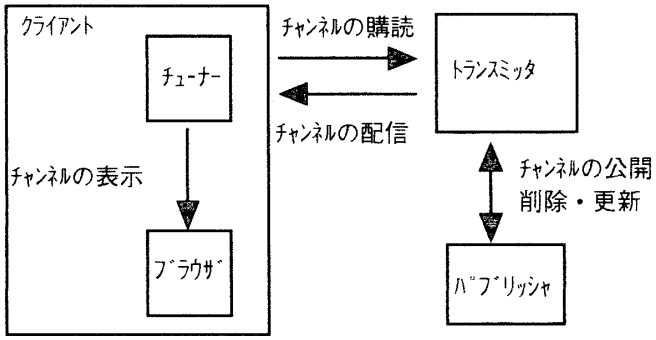

図8 Castanetシステムの構成

Castanetシステムでは，受配信するコンテンツをチャンネルと呼び，ユーザは特定のチャンネルの情報配信依頼をトランスミッタに対して登録する (購読すると呼ぶ。) ことによって，自動的な情報の受信が可能となる。

以下に各要素の主要な機能を示す。

1．トランスミッタ

チャンネルを管理するサーバであるトランスミッタは，アプリケーション配送プロトコル (ADP: Application Distribution Protocol) を用いてユーザに対してチャン

ネルを配信する。配信の際には，差分更新の機能により，既にユーザが購読しているチャネルに対しては，変更のあった部分のみを配信することで，トラフィックの軽減とユーザの利便性が確保されている。

2．パブリッシャ

チャンネルの提供者は，パブリッシャを利用してトランスミッタへのチャンネルの登録や更新，削除を行う。パブリッシャではユーザ認証が行われ，登録が許可された者のみがパブリッシャを利用できる。

3．チューナー

チューナーは，ユーザが利用するクライアントで起動され，新規チャンネルの購読や購読しているチャンネルの自動更新を行う。購読されるチャンネルは，ユーザのクライアントマシンのローカルディスクにロードされ，実行される。チャンネルがHTMLの場合では，WWWブラウザを起動し，チャンネルの内容を表示する。現在，Castanetシステムで利用可能なチャンネルは，HTML，アプレット，プレゼンテーション，アプリケーションの４つである。

5　管理用ツールの構成

5.1 管理用ツールの概要

管理用ツールは，データベースに対する情報の更新と更新情報の共有を支援する機能を持ち，管理者へのインタフェースとして，Javaアプレットによるインタラクティブな操作を提供する。データベースへの更新および情報の共有処理は，アプリケーションサーバ内のオブジェクトが担当し，データベースの更新情報をCastanetシステムを経由し，管理者に配信する。

5.2 管理用ツールの基本構成

管理用ツールの基本構成を図９に示す。ツールの主要な要素は，WWWクライアント上のJavaアプレットとアプリケーションサーバ，ネームサーバ，データベースサーバおよびCastanetシステムである。

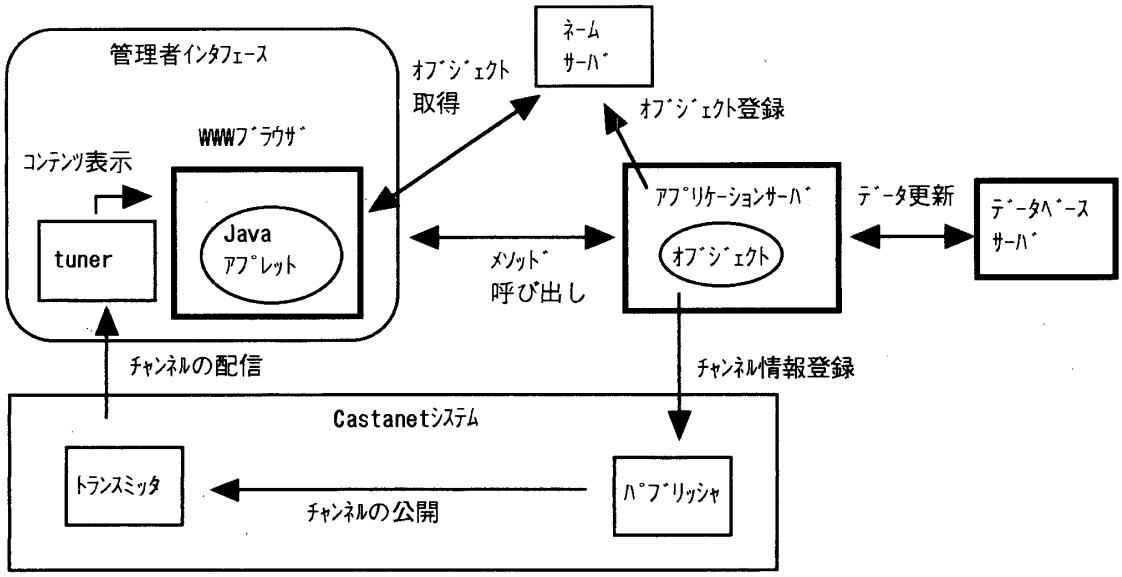

図９　管理用ツールの基本構成

5.3 Javaアプレット

管理者は，WWWブラウザをクライアントとして利用し，Javaアプレットから提供されるGUIによってアプリケーションサーバに対してリクエストを発行し，データベースの情報にアクセスする。ユーザインタフェースを担うクラスは，ボタンやリスト，テキストフィールドといったGUIを備え，WWWブラウザへの初期ロード時に，ネームサーバに対してオブジェクトリファレンス（アプリケーションサーバに存在するオブジェクトの識別子）の問い合わせを行う。ネームサーバのホスト名及びポート番号は，アプレットのパラメータとして指定する。以下にパラメータ設定例を示す。

〈PARAM name=" org.omg. CORBA. ORB Initial Port" VALUE=" 5001"〉
〈PARAM name=" org. omg. CORBA. ORB Initial Host" VALUE=" kokufu. takaoka-nc. ac. jp"〉

5.4 アプリケーションサーバ

アプリケーションサーバは，クライアントからのオペレーションによってデータベースに対してアクセスを行うオブジェクトの集合体であり，以下に示すオブジェクトがJavaのクラスとして実装されている。

1．資料オブジェクト（article）
資料情報の管理
2．作者管理オブジェクト（author）
作者情報の管理
3．管理用オブジェクト（admin）
castanetシステムとの連携
4．データベースコアオブジェクト(dbcore)
JDBCによるデータベースへの接続

データベースの更新を行うためのSQL命令は，全てオブジェクトのメソッドに隠蔽される。以下に，articleオブジェクトが持つメソッドの例を示す。

- Boolean setDetail (String no,……)
  資料情報の変更
- Boolean delArticle (String no)
  資料情報の削除
- Boolean putArticle(String no,……)
  資料情報の追加

5.5 ネームサーバ

サーバアプリケーションに実装されているオブジェクトのリファレンスを管理するサーバであり，プレゼンテーション層とファンクション層に対してネーミングサービスを提供する。アプリケーションサーバは，起動時にネームサーバに対して実装するオブジェクトを単純な名前で登録し，クライアントでは，その名前によるオブジェクト呼び出しを可能とする。

今回の実装では，5.4に示す各オブジェクトの名前（article, author, admin, dbcore）をそのままネームサーバに登録している。

5.6 データベースサーバ

データベースサーバは，フリーのリレーショナルデータベースであるPostgreSQL[15]を利用し，付属のJDBCドライバによってアプリケーションサーバと接続する。（現在使用しているPostgreSQLはUNIXプラットフォームを対象とするため，実装はUNIXマシンに限られる。）

オブジェクトが処理対象とするデータベースの詳細なスキーマについては，[7]を参照されたい。

5.7 Castanetシステム

4章で述べたように，Castanetシステムではトランスミッタにwwwコンテンツをチャンネルとして公開し，クライアント側でチューナーによってチャンネルを購読することで，自動配信が開始される。Castanetシステムには，GUIを備えたチャンネル受信用のモジュー

ルであるtunerが装備されており，これ自身が単独のアプリケーションとして動作する。今回は，管理用ツールのWWWインタフェースとtunerを併用し，更新のあったWWWコンテンツをチャンネルとして公開し，管理者間の情報共有を行う。

6 情報共有の仕組み

castanetシステムでは，特定のディレクトリをHTMLチャンネルとして公開することで，そのディレクトリ配下の全ファイルがコンテンツとして取り扱われる。(チャンネル公開時に，特定の拡張子を排除することは可能である。)一般的なHTMLチャンネルでは，管理対象とするコンテンツを含む親ディレクトリをチャンネルとして公開するが，tuner側では，ツリー構造のトップとなるindex.html等を起点として順番にコンテンツを辿る必要があり，更新されたコンテンツが特定できない。

今回は，管理者側でコンテンツを容易に把握する機能を持たせるため，提供するチャンネル専用のディレクトリを作成し，その配下のindex. htmlに対して，更新されたコンテンツへのリンクを追加すると共にコンテンツ自身を専用ディレクトリにコピーし，チューナーに提供した。

以下に，データベースの更新からtunerでの共有情報表示までの流れを示す（図10)。

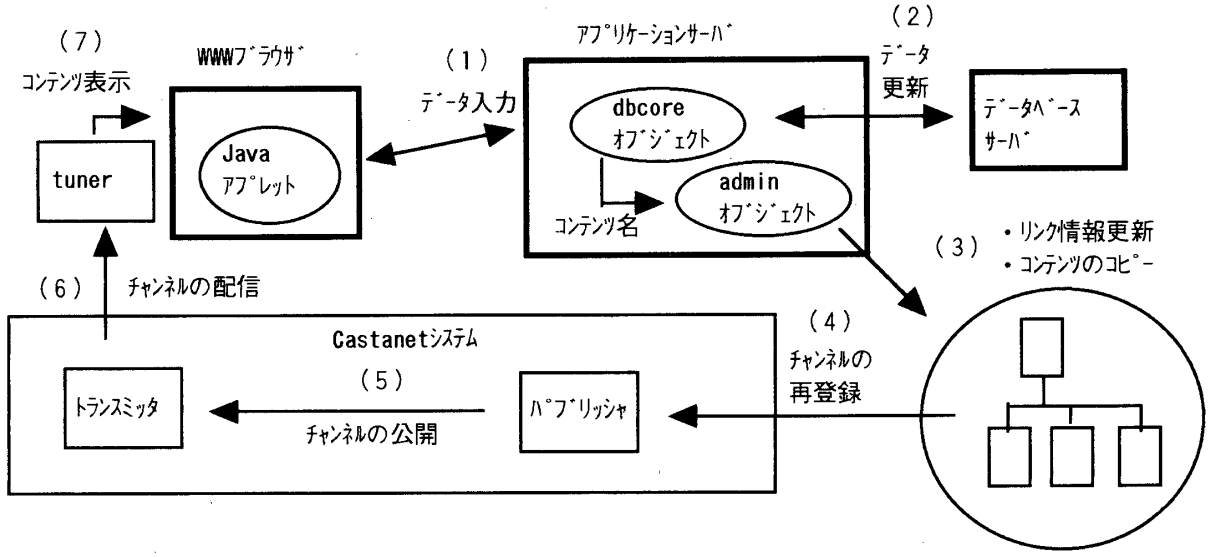

図10 情報共有の仕組み

(1) 管理用インタフェースによるデータベースの更新要求に従ってdbcoreオブジェクトがデータベースの内容を更新する。

(2) dbcoreオブジェクトがadminオブジェクトに更新されたWWWコンテンツ名を引き渡す。

(3) adminオブジェクトが，WWWコンテンツの変更分を専用ディレクトリにコピーし，index. htmlにそのリンク情報を追加する。

(4) adminオブジェクトが，パブリッシャを起動し，専用チャンネルを再登録する。

(5) パブリッシャからトランスミッタに対して，チャンネルの更新要求が出される。

(6) トランスミッタは，チャンネルを更新し，チューナーに対して変更があったチャンネルを配信する。

(7) チューナーは，更新されたチャンネルをロードし，WWWブラウザ上に表示する。

7 管理用ツールの特徴と応用

7.1 管理用ツールの特徴

(1) 管理者インタフェースの向上

管理用ツールでは,これまでのCGI方式によるインタフェースに比べ,インタラクティブ性を重視した扱いやすいGUIが提供できた。WWWブラウザでのページ単位の切り替えが発生しないというメリットに加え,処理状況を表示するステータスバーの採用により更新の確認が速やかに行え,効率的な作業環境が実現する。

図11に,管理者インタフェースを示す。

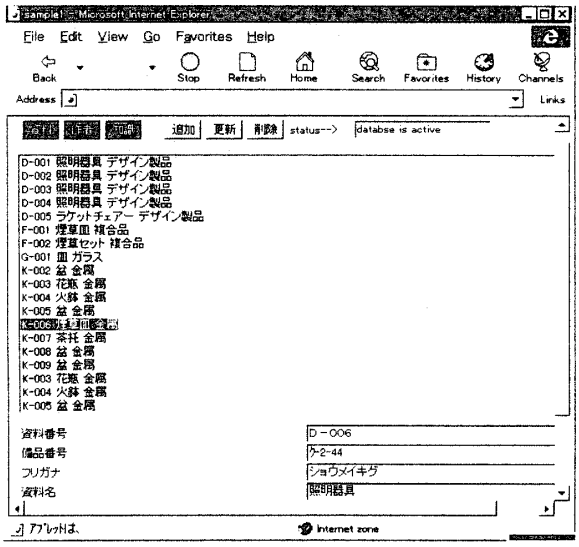

図11 管理者インタフェース

(2) 情報共有機能の提供

管理者は,専用のチャンネルを購読することにより,自動的に配信されてくるデータベースの更新結果(WWWコンテンツ)を把握することができる。複数の管理者が分散管理を行う場合には,各々が同じチャンネルを購読することで,更新情報の共有が可能である。これにより,管理者のコンテンツ管理の負担が軽減され,複数の更新結果をまとめて確認することも可能であり,柔軟な運用が行える。

図12に,tunerでの共有情報の表示例を示す。(右側がtunerインタフェースのチャンネルリスト,左側が更新されたWWWコンテンツである。)

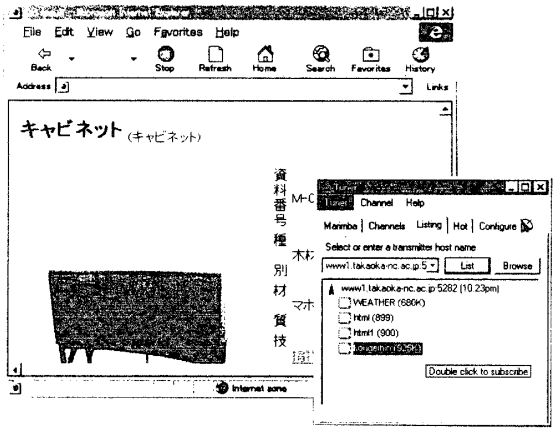

図12 共有情報の受信と表示

7.2 管理用ツールの応用

(1) ホームページ管理への応用

ホームページの運用では,その規模の大きさに応じて適切な分散管理が必要であり,各カテゴリーごとに管理者がおかれ,各管理者の権限において担当するカテゴリー配下のコンテンツの作成と更新が行われる。こうした環境において,コンテンツ作成は管理者のみに限定されず,管理者の承認を必要とするものの,そのカテゴリーでの作成許可を得た複数のユーザによって自由に行われる。その際には,作成されたコンテンツを管理者が効率良くチェックする機能が必要となるが,管理用ツールの情報共有機能は,そのまま分散したホームページ管理に適用できる。

(2) 他WWWアプリーケーションへの応用

情報共有の中心機能は1つのオブジェクトによって提供されているため,今後のWWWアプリケーションの開発にあたって分散オブジェクト方式を選択する場合には,十分

に再利用可能であり，開発の効率化が計れる。また，情報共有機能の処理ロジックは，indexファイルの更新とファイルのコピーという単純なものであるため，CGI等の他処理系で実装された既存アプリーケーションとCastanetシステムとの連携も可能であり，情報共有機能が容易に付加できる。

8 おわりに

今回，CGI方式による既存連携ツールの管理者用インタフェースを改善すべく，分散オブジェクト技術による3層モデルを構築し，管理者間におけるデータベース更新情報を共有する機能を持つオブジェクトを実装した。3層モデルへの移行により，ファンクション層の実装にあたっては，Java IDLによるスタブの自動作成機能により，オブジェクト呼び出し時にもローカルとリモートを意識する必要がなく，効率的な実装が行えたが，アプレットの開発に際しては，GUIが複雑化するほどイベント処理のロジックが多くなり，より扱いやすいGUI開発ツールの必要性を感じる。また，分散オブジェクトの開発には，JDK1.1を利用したが，以前のバージョン（JDK1.0）とはイベント処理やメソッドの取扱いに多くの違いがあり，最新のＪＤＫに対応したブラウザのみがプレゼンテーション層として機能する。筆者が確認した限りでは，現在利用できるもののうち，十分な実用に耐えたのは1つのブラウザソフトのみであり，各ブラウザに搭載されるJVM（Java Virtual Machine）のJDKへの対応が遅れているという現状がある。今回は，管理者用ツールの整備という目的に重点を絞った開発を行ったが，プレゼンテーション層に用いたJavaアプレットによるGUIは操作性にすぐれたものがあり，一般ユーザにも提供できることが望ましく，WWWブラウザの充実に期待する。

分散オブジェクト技術では，基本的な機能を実装したが，分散処理を実現する手段は今回利用したJava IDLの他，RMI（Remote Method Invocation）やHORBなど多様な選択が可能である。今後これらの導入も検討し，分散するオブジェクト間で実現するにふさわしい新たな機能を見い出したい。

最後に，Castanetシステムは，本来Javaアプリケーションを自動配信することを主な目的として開発されたシステムであるが，今回は簡易な利用方法にとどまった。専用のGUIビルダを利用すれば，より以上に洗練されたアプリケーションが構築でき，現在の管理用インタフェースを1つのアプリケーションとして集約することも可能である。この点を含め，動的にオブジェクトを生成する分散オブジェクト技術とプログラム自動配信機能の連携による充実したユーザサービスを提供するツールの開発が今後の課題である。

引用文献

1）山内他：WWWを利用した学生情報サービスについて，平成9年度情報処理教育研究集会講演論文集，pp.394-397（1997）

2）野里他：ネットワークを活用した研究所内技術情報サービス，情報処理学会第54回全国大会講演論文集，情報処理学会，4-321（1997）

3）日本インターネット協会：インターネット白書'97，インプレス，pp.84-97（1997）

4）三喜英行他：3層クライアント／サーバ・システム構築技法，P.216，ソフト・リサーチ・センター（1996）

5）㈱情報処理学会ソフトウエア工学研究会：オブジェクト指向最前線，P.191，朝倉書店（1997）

6) Timothy W.Ryan : Distributed Object Technology Concepts & Application, Prentice Hall (1997) .植野直樹, 多田征司(訳) : 分散オブジェクトテクノロジー 概念とアプリケーション, P.200, プレンティスホール出版 (1997)
7) 米川 覚 : 高岡短期大学収集産業工芸資料のデータベース化について, 平成8年度高岡短期大学開放センター事業実施報告書, pp.207-212 (1997)
8) OMG, CORBA/IIOP2.1 Specification, 1997
9) Sun Microsystems, JDK1.1.3 Documentation, 1997
10) Sun Microsystems, JavaIDL Guide, 1997
11) http://www.pointcast.com/
12) http://www.msnbc.com/
13) Laura Lemay : Official Marimba Guide to Castanet, Sams. net Publishing(1997).松田晃一, 小沼千絵, 樋江井太(訳) : Marimbaオフィシャルガイド Castanet, P.346, プレンティスホール出版 (1997)
14) http://www.marimba.com/
15) University of Calfornia at Berkeley, The POSTGRES95 User Manual, 1995
16) Randy Otte.,etal. : Understanding CORBA: The Common Object Request Broker Architecture, Prentice Hall(1996). MISCOオブジェクト指向研究会(訳) : 分散オブジェクト指向CORBA, P.255, プレンティスホール出版 (1996)

The Use of a WWW-Database Integrated System with Distributed Objects to Improve the Administrator Interface.

Satoru YONEKAWA

(Received October 31, 1997)

ABSTRACT

This paper describes how a WWW-Database Integrated System has improved the drawbacks of CGI system and network administrator interface through the creation of a three-tiered model that allows for greater ease of use, as well as more efficient exchanges of information between network administrators. In the three tiered model developed by the author, the application server communicates with the WWW client through the Java IDL-based ORB, which then makes the required changes in the database. Through Castanet, the Java-based object then automatically sends upgrades in the database to each administrator, as well as providing other information sharing capabilities. In addition, this paper describes other improvements in network management efficiency that have resulted from this system, especially in the management of the Takaoka National College home page.

KEY WORDS

Distributed Object, Three-tier model, ORB, IDL, Push technology, Castanet