データベースシステム 116－62
（１９９８．７．１０）

スクラップ アンド ビルド型マルチメディアDBMSの構成方式

河 信司，市間健太郎，高橋賢一郎，家富誠敏，有澤 博

横浜国立大学 工学部 電子情報工学科
〒240-8501 横浜市保土ヶ谷区常盤台79-5

E-Mail: { kawa, ken, g3, eto, arisawa}@arislab.dnj.ynu.ac.jp

あらまし

様々なメディアを扱うマルチメディアデータベースでは、その分処理の内容も多岐に渡り、運用しながらのオペレータやコンピュータの追加・削除といった構造的な変更ができるような仕組みが必要になる。

本論では、このようなデータベースマネージメントシステムとして、新たなメディアやオペレータあるいはコンピュータ資源を、DBMSを止めることなく追加・削除できるスクラップアンドビルド型マルチメディアDBMSを紹介する。

A Design of Scrap-and-Build Multimedia Database Management System

Shinji KAWA, Kentaro ICHIMA, Kenichiro TAKAHASHI, Masatoshi IETOMI, Hiroshi ARISAWA

Division of Electrical and Computer Engineering
Faculty of Engineering
Yokohama National University
79-5, Tokiwadai, Hodogaya-ku, Yokohama-shi, Japan

E-mail: {kawa,ken,g3,eto,arisawa}@arislab.dnj.ynu.ac.jp

Abstract

Mulimedia database systems treat various types of media data and also involve media-dependent operations for the media. This paper presents a new DBMS architecture which enables us to include and exclude some opeators.This paper introduces Scrap and Build MMDB which can include and exclude media-dependent operators or other computer resources for new media.

1 はじめに

計算機の技術は日々発展している。計算機で扱う対象となるデータも文字列や数値から、フルテキスト、CADデータ、イメージ、音声、映像などの様々な形態をしたデータへと変化してきた。こうした様々な形態のデータを扱うためのデータベースがマルチメディアデータベースである。本論では、それぞれのデータ形態のことをメディアと呼ぶことにする。マルチメディアデータベースでは、複数のメディアに渡って統合的な操作を行なうことができる。

メディアの(特に映像や音声などの)特徴を以下に記す。(1) メディア毎に独自のオペレータが存在する。(2) 大容量で連続的なデータある。(3) メディアデータに解析を施すことによって、より高次の情報を引き出すことが出来る。以上のような特徴を持つ複数のメディアを統合して扱うためにマルチメディアデータベースには以下の項目が求められる。(1) 新たなメディア依存のオペレータを柔軟に組み込むことの出来るデータベース検索言語 (2) 大容量のデータを高速に処理するためのアーキテクチャ (3) オペレータの追加・削除を考慮したアーキテクチャ。データベースマネージメントシステムの構成要素を動的に変化できること。(4) 解析によって得られる意味情報の付加にしたがって、任意にスキーマを変化させて、それにインスタンスを追加できる仕組み。

我々の研究プロジェクトでは、上に述べた要求に答えるために(1)に対しては関数型検索言語MMQL[3]の埋め込み関数と呼ぶ枠組でサポートし、(2)に対しては並列計算機上での効果的な実装方法[4]を提案している。また、(4)に関してはスキーマ

残る(3)が問題である。通常オペレータやタイプの追加や削除を行なうことはシステム全体に対する変更を余儀なくされ、アプリケーションプログラムの再構築や、場合によってはDBMSを停止させる必要がある。このような変更が頻繁に起こり得るマルチメディアデータベースでは実用には耐えられない。本研究では、スキーマやメディアまたそれに対するオペレータの拡張を動的にDBMSを止めないまま行なうデータベースとして、スクラップアンドビルド型マルチメディアDBMSを提案する。スクラップアンドビルド型マルチメディアDBMSではネットワーク上に散らばる3次元画像処理マシンや、大量ストレージを持つマシンといった分散した計算機資源の構成ををを効率良く変化させて扱うこともでき、一つのDBMSで画像データベースから事務処理データベースまで扱うデータベースに多面性を持たせることが出来る。さらにDB運用中に画像関連の処理を強化するといったチューニングもできる。

第2章ではMMQLの実装系であるマルチメディアデータベース検索エンジンの満たすべき用件を述べる。その用件を満たすためのアーキテクチャとして、本論文では第3章でスクラップアンドビルド型マルチメディアデータベースの紹介をする。第4章ではそのシステムの実装のための一手法を説明する。

第6章ではまとめと今後の研究の方向性について述べる。

2 マルチメディアDB

2.1 マルチメディアデータ

マルチメディアDBMSでは従来のDBMSで主に扱っていた、文字や数値に加え、静止画像や動画像、さらに3次元情報や3次元+時間軸情報といった多種多様なメディアを統合的に扱うことが要求される。

これらのメディアの特徴は既に述べた通りであるが、例えば映像を例にとると、

1. 時間的に連続したイメージデータ(フレーム)の列であり、その大きさのため何か加工しようとした時の処理時間も大きなものになってしまう。

2. カメラからとった映像はそのままではデータベースの検索対象とはなりにくい。映像に何らかの解析を施して、「何が写っているのか」「何をしているのか」「映像中の2物体間の距離は」といった高次な情報を得ることが必要である。また解析した結果をデータベース中に蓄積していく時に、スキーマを変化させなければならないことも必要になる。

3. 検索結果として特定被写体の強調や、カメラからの映像とCGの合成など、映像メディア

特有の画像処理 (メディア依存処理 [3]) も多数必要になる。

2.2 マルチメディアDB

これらを扱うマルチメディアDBということを考えるとさらに、今現在あるメディア以外にもこれから増える可能性も十分にあり、それらのメディアと今まで蓄積してきたメディアとを複合的に検索することも出来る必要がある。例えば画像データベースに新しく音声メディアもデータとして入れて、「犬が二匹向かいあって吠えている映像を検索せよ」といったことがあるであろう。このように自由度の高い検索は、アプリケーションプログラムレベルではなく、DBMS レベルで多様なオペレータを追加・削除できなければ実現できない。

つまり、マルチメディアDBに必要とされる要件としては以下のものが挙げられる。

- 大きなデータを効率良く高速に扱える
- ネットワーク上に画像処理が得意なマシン、大量ストレージを持つことが出来るマシン、マルチプロセッサを用いて高速な演算ができるマシンなどが散在していた時にも、それらを効率的に利用できる
- 情報を解析することによって得られる意味情報の付加にしたがってスキーマに対しても柔軟な変更が可能
- DBMS に対して新たなメディアの追加やメディア依存処理の追加を DBMS に変更を加えることなく行なうことができる

これらの要求を満たすために、本稿では以下の章に述べるようなスクラップアンドビルド型マルチメディアDBMS を提案する。

3 スクラップアンドビルド型DBMS

本稿では各メディアやオペレータはDBMS があらかじめ用意するものではなく、データと同じように自由に追加・削除できるものと捉え、さらにDBMS を構成している計算機資源の構成もDBMS を運用しながら変化してゆくスクラップアンドビルドという考え方を提案する。

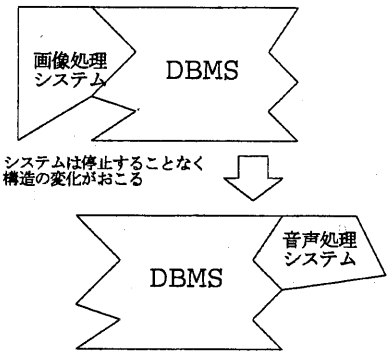

図 1: スクラップアンドビルド型マルチメディアDBMS

スクラップアンドビルド型マルチメディアDBMS では以下のことが DBMS を止めたり、既存の他モジュールの変更をすることなくできるものとする。(図 1)

- 動的にその構成モジュール (ネットワーク上に分散したものも含めて) 変更できる
- 既に実働中のモジュールに対しても、その数を増やすことによって最適な処理環境を提供できる
- スキーマ、モジュールの拡張ができる
- 事務処理から、マルチメディアの検索まで、検索言語中の任意のオペレータを追加・削除できる

スクラップアンドビルド型マルチメディアDBMS を実現することによって、多種多様なメディアを扱うマルチメディアデータベースにおいても、柔軟に扱うデータの守備範囲を広げ、さらに画像処理は画像処理を高速に行なえるマシン、出力はCG シミュレータや VR システムを装備したマシンに出力を行なうといった有用な計算機資源の利用ができる。

この時にユーザは以下のようなオペレーションを行なうことによって、スクラップアンドビルド型マルチメディアDBMS の利点を享受することができる。

1. DBMS に現在実働可能な処理の問い合わせを行うことができる。

2. ユーザが独自のメディア依存処理を行なうプログラムを書き、本システムのライブラリを使用することによって簡単に組み込むことが出来、検索文に使用することが出来る。

3. その検索内容を処理することの出来るモジュールが組み込まれていない時にはモジュールの追加を行なうことができ、何か重たい処理を行なう時にはそのモジュールの数を増やすことによって、処理の効率化を行なうことが出来る。

4. DBMS 側でも稼働率の高いモジュールを察知して、そのモジュールの数を増やすことによって処理の効率化を行なう

5. 稼働率の低いモジュールを動的に取り外すことが出来る。

一方、このような方式をとる時に問題となるのは、

- オペレータをどのようにとらえ、実装するのか
- DBMS に型とオペレーションをどのように管理させるか
- 検索言語のコンパイラに対する拡張を動捉えるか。これも、コンパイラの再コンパイルなどがない状態で行いたい。
- オペレータを適用する時の型のチェックをどうするか

ということである。本稿ではスクラップアンドビルド型マルチメディア DBMS の設計と、プロトタイプシステムの実装を通して、これらの問題に対する一つの答を示す。以下の章にその設計方針について述べる。

4 システムの設計

以下に関数型検索言語 MMQL とスクラップアンドビルド型マルチメディア DBMS のアーキテクチャを示す。

4.1 システム構成方針

オペレータの動的な追加・削除や、異なる特性を持つコンピュータを統合して作業させるために、本稿ではオペレータ毎が独立したプロセスとして起動・終了し、自立的に処理をこなすマルチエージェントシステム [7] を採用した。マルチエージェントシステムの枠組では新しい型やオペレータの追加を新しいエージェントの追加と捉え、DBMS 本体とは距離をおくことによって構造の変化に対して柔軟かつ頑強にすることができる。以下にこの設計を行なうためのデータ構造と検索処理の流れを述べる。

本システムの構成を図 2に示す。

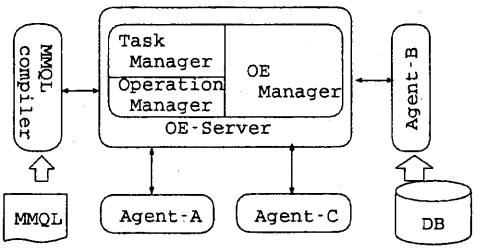

図 2: システム構成

このシステムは大きく、検索途中のデータの保持やジョブの制御をするサーバーと、実際の検索処理を行なうエージェントに分かれる。なお、ストレージはこのシステムからは切り離されている。

ここでのエージェントには、以下のような種類がある。

- DB 通信エージェント
  ストレージとの通信を行なうエージェント。検索要求に従って、ストレージから特定のデータを取り出す。
- メディア依存処理エージェント
  各メディア依存処理を行なうためのエージェント。画像合成なら画像合成のみ、四則演算なら四則演算のみを行なうエージェントが、多数存在する。各エージェントは上記の OEPL を引数及び返り値として用いる。
- 検索言語解析エージェント
  ユーザからの検索言語を解釈して、それを有

効グラフで表されるトランザクションの列としてサーバに登録する。本論文で言うトランザクションとは、DB 検索エンジン内部でのデータの変更のことである。同時実行制御のための並列性の抽出などもこのエージェントが行なう。

- 出力エージェント
  サーバ内に出来上がった複合オブジェクト (OE) を、プレーヤーに受け渡す。この時は先に示した OE の形でも良いし、1 本の映像のように加工されたものでも良い。

実際の検索処理の流れとしては以下のようになる

1. 検索言語解析エージェント (MMQL コンパイラ) が、検索言語を有効グラフで表現できるトランザクションの列に変換し (図 4)、それを、サーバのトランザクション管理部に登録する。

2. 各エージェントがサーバのトランザクション管理部に、自分が現在実行可能なトランザクションを問い合わせ、自分の出来る処理が見つかったならば OE Manager から引数となる OEPL を転送してもらい処理をおこなう。処理が終了するとその返り値となる OEPL をサーバへ転送し、同時にトランザクション管理部へ処理の終了を伝える。

3. 検索処理が終了すると、出力エージェントが検索結果を出力する。

4.2 スクラップアンドビルドの手順

このようなアーキテクチャを考えた時、全てのエージェントを常時サーバと通信させていたのでは DBMS マシンのパフォーマンスに大きな悪影響を与える。エージェントには必要の無い時には停止していて欲しい。またダイナミックリンクを使ったのでは、サーバプロセスは呼び出す相手 (エージェントの) 全ての情報を保持していなければならなくなり、システム全体に対する動的なオペレータの追加・削除が難しくなる。また呼びだし側 (サーバ) の柔軟性も低くなる。

そこで本稿ではオペレータとなるエージェントを動的に追加・削除するために、サーバ側のオペレータ管理機能を強化し、エージェントは要求された時にサーバと連絡をとり合い、またそのオペレーションの要求がなくなればサーバの負荷を軽減するためにサーバとの通信を終了するようにする。これによりスクラップアンドビルド型マルチメディア DBMS を実現する。(図 3)

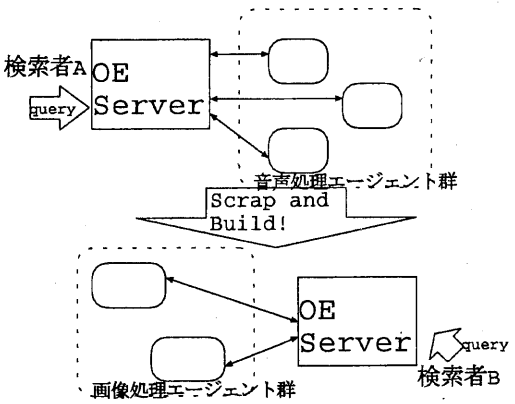

図 3: スクラップアンドビルド方式

この時、新たなエージェントの追加・削除の機構は次のようになる。

1. スクラップアンドビルドを行なう人間 (大抵は DB 設計者) が単値もしくは集合に対してのオペレータをコーディングする。

2. マルチエージェントシステム用にエージェント作成キットをつかって、DBMS との通信部分を組み込む。

3. そのエージェントを起動して検索言語を通してそのエージェントを呼び出す。

例えば、新たにメディア依存処理である FFO[3] を組み込む時には次の手順をとる。FFO は強調画像を作成するために画像合成を行なうオペレータである。

1. まず、1 フレームに対する画像合成の処理をコーディングする。

2. 次に、エージェント作成キットにそのコードと引数、返り値の型 (この場合は複数の画像

から一枚の画像を出力する) さらに、MMQL 中での埋め込み関数名 (FFO) を登録し、新たなエージェントを作成する。

3. FFO エージェントを起動することによって、サーバに FFO の存在と引数、返り値の情報が与えられ、検索言語 MMQL 中に FFO が現れると FFO エージェントが処理を始める。

以上のようにして、スクラップアンドビルドの実現を行なう。

4.3 OE サーバ

以上に述べたようなことを実装するために、サーバには少なくとも以下の機能が必要となる。

- オブジェクト式管理部
  前述の OEPL を管理している。エージェントに対して OEPL の配信・登録を行なう。OEPL の整合性管理もこの部分で行なっている。
- トランザクション管理部
  オブジェクト式に対するトランザクションの管理を行なう。具体的には MMQL コンパイラから送られてきたトランザクションツリーをもとに、各トランザクションの進行状況から現在実行可能なトランザクションをエージェントに伝える。各エージェントはここに自分の実行可能なトランザクションを問い合わせる。
- データタイプ・オペレータ管理部
  本システムではデータ型およびオペレータの種類は常に不定である。従ってサーバでは今現在どのようなエージェントが実行可能になるのかを把握しておく必要がある。具体的にはエージェントが起動する時にサーバと通信し、その時に各エージェントの ID、処理名、引数と返り値のデータ型といった情報を受け取りそれを登録内容としてサーバ内に保管する。また必要がある時はその時に停止しているエージェントにメッセージを送って、システム構成の変更を行なう。

4.4 MMQL コンパイラ

MMQL コンパイラはサーバのオペレーション管理部との通信を介して、サーバに今現在どんなエージェントが登録されているかを問い合わせる。次に MMQL 解析結果である各処理に対して、エージェントの割り振り及びスケジューリングを行なう。この時、同時に型に対するチェックも行なう。MMQL は検索文を見た時点で検索結果であるオブジェクト式がどのような形になるか判別できる言語であるため、このことと上記のオペレータ管理部からの情報により、型チェックを行なうことが出来る。型チェックがうまくいった場合はよいが、型チェックがうまく行かない時には本システムでは 2 通りの場合がある。

1. 単純に検索文が間違っていた場合
2. その型もしくはオペレータがサーバに登録されていない場合

2 番目の場合は MMQL コンパイラは Type unknown. として処理を終了する。しかし、そのエージェントを登録すれば検索は実行されることとなる。

MMQL コンパイラの役割を表した図を図 4に示す。この図において一つの処理単位 (Atomic Transaction Code) には、それぞれその処理の ID、引数、返り値が記されている。

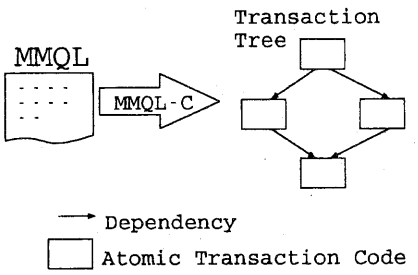

図 4: MMQL コンパイラ

4.5 メディア依存処理エージェント

今述べたように、MMQL コンパイラは検索文を 1 つのエージェントの処理できるトランザクションの列に変換する。

エージェントにはその一つ一つのトランザクション毎に対応して、四則演算エージェントや、画像の強調加工エージェントといったものが複数存在す

る。これらのエージェントは起動してまず最初にデータタイプ・オペレータ管理部に、そのエージェントに関する情報をサーバに登録する。その内容は、識別名、MMQL 中での埋め込み関数名、引数、返り値、そのエージェントを起動するためのコード、実行可能なユーザグループ、製作者、バージョンなどである。MMQL コンパイラはこの情報からどのエージェントに仕事を割り振るかを決める。

これを行なった後、エージェントは1度サーバとの通信をしなくなる。次にサーバのトランザクション管理部に検索要求が入り、その中にそのエージェントが処理可能な処理が含まれていると、サーバのオペレータ管理部から処理要求が来る。するとエージェントはトランザクション管理部と通信をしながら、オブジェクト式管理部から OEPL を受け取り処理を始める。

これにより、新たなエージェントの追加に対しても型の不整合はおきにくく、なおかつ DBMS やコンパイラを途中停止や再コンパイルの必要がなくなる。

5 プロトタイプシステムの実装

この章では検索エンジンのプロトタイプとして、実装してみたものを紹介する。

UNIX ワークステーション上に OE Server に DB 通信エージェント、四則演算エージェント、画像合成エージェント、選択演算エージェントと画像処理の例としてフレームの強調加工をする FFO エージェント、その他幾つかのエージェントを実装した。

また入力インタフェース及び出力インタフェースには Web を用いた。

現在のシステム構成図を図5に示す。

OE Server には UNIX ワークステーション、出力エージェントには3次元グラフィックシミュレータを備えたマシンを用意する。また大量ストレージを抱えるマシンをマルチメディアデータの格納場所として確保する。

今後にこの図の様に OE Server を並列計算機用に移植することによって、検索エンジンの高速化を計ってゆきたい。

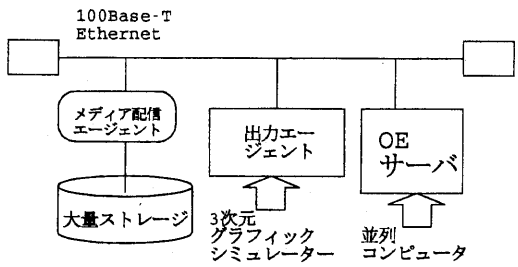

図 5: 実装したシステムの構成図

6 まとめと今後の展望

本論文では、DBMS に対して新しい型の追加や削除を行なった時に、DBMS 本体自体は再コンパイル、あるいは停止をするまでもなく柔軟に変化することの出来る DB 検索システムを設計した。

それを行なうためには、各モジュールをエージェントとして検索システム本体とは独立させ、それを動的に追加、削除できることが必要であり、検索システム本体は各モジュールの規格情報のみを管理させた。そして、各オペレータを追加する時にはしかるべきメッセージをオペレーションの管理部に伝えエージェントを起動し、そのオペレーションが不要になった時には DBMS の負化を落すためにそのエージェントを削除するという管理方法を考えた。

これにより、未知のオペレーションやデータ型が多種多様に存在し得るマルチメディアデータベースにおいても、不要な負化を DBMS にかけず、モジュールのシステム構成の変化にも頑強なシステムを実現することが出来る。

今後は並列計算機上での実装を進めることによって、本システムの有用性を示していきたいと考える。

参考文献

[1] 有澤 博: “リアルワールドデータベースとその実現技術 1-3”, bit, Vol.28, No.9-11, 1996.

[2] H.Arisawa, T.Tomii, H.Yui, H.Ishikawa: “Data Model and Architecture of Multimedia-Database for Engineering Applications”, IEICE Trans. In-

f. & Syst., Vol.E78-D No.11, pp.1362-1368, November 1995.

[3] 富井 尚志, 有澤 博: "マルチメディアデータベースにおける映像モデリングと操作言語", 電子情報通信学会論文誌, Vol.J79-D-II, No.4, pp.520-530, April 1996.

[4] 家富 誠敏, 河 信司, 市間 健太郎, 富井 尚志, 有澤 博: "関数型検索言語 MMQL における並列処理機構の設計", 電子情報通信学会第 9 回データ工学ワークショップ (DEWS'98), pp.29.1-29.4, 1998.

[5] 舛谷 和幸, 富井 尚志, 有澤 博 : "関数型データベース検索に適したデータ構造とそのオペレータの実装," 情報処理学会第 53 回全国大会, 情報処理学会, pp.57-58, 1996.9.

[6] J.Backus: "Can Programming Be Liberated from the von Neumann Style? A Functional Style and Its Algebra of Programs", CACM, Vol.21, No.8, pp.613, 1978.

[7] 沼岡 千里, 大沢 英一, 長尾 確: "マルチエージェントシステム" 共立出版, 分散強調メディアシリーズ 11,1998.