データベース・システム 73－4
オペレーティング・システム 44－4
（１９８９．９．８）

ＤＲＥＡＭシステムについて

疋田　定幸、川上　英、吉田　誠、坂本　明史
沖電気工業株式会社

分散データベースシステムＤＲＥＡＭシステムについての概要とＬＡＮを中心にした実現に関して述べる。
利用者の分散処理システムの構築において分散データベースシステムは価値が高い。ＤＲＥＡＭシステムはこの観点から開発された分散データベースシステムである。分散に対するいくつかの透過性を提供し、ＳＱＬで記述された検索や更新を複数サイトのデータベースに対して処理できる。また、集中型データベースシステムＲＥＡＭのアプリケーションプログラムを継承でき、各種のＰＣや他社のホストとの接続性も考慮されている。本稿ではＤＲＥＡＭシステムの開発方針、処理方式、および機能構成について述べる。

DREAM: A Distributed Database System

Sadayuki HIKITA, suguru KAWAKAMI, Makoto YOSHIDA, Akifumi SAKAMOTO

OKI Electric Industry Co.,Ltd.

16-8,Chuou 1-chome, Warabi-shi, Saitama 335,Japan

In this paper, we present the outline of distributed database system DREAM, focusing on its LAN based implementation. A distributed database system is an attractive component to organize a distributed application system. Fot this purpose, DREAM provides major transparency functions necessary for distributed processing. Any application program in DREAM can access any database without knowing its location. Because DREAM and REAM adopt standard SQL for their database language, no changes are needed for their executions in either envirionment. This paper introduces DREAM from the viewpoint of the netowrk environment. Then the architecture and implimentation of DREAM are described.

1．はじめに

情報処理システムの高度化にともないシステムの分散処理化が不可避なものとなっている．沖電気ではＤＡＡ（Distributed Application Architecuter）により利用者の分散システム構築を支援するための体系付けを行っている[沖研88a]．ＤＡＡでのキーファクタとしてネットワーク内でのデータベースへの接続性がある．すなわち利用者のアプリケーションシステムの構築において、各種の計算機の相互の接続と、それらの計算機間でのデータの分散と共有を可能とする分散システムの開発を支援することがＤＡＡの目的の一つとなっている．

分散データベースシステムは分散システム構築の中核技術となる．たとえば、データベースへの透過なアクセス、段階的なシステムの構築、トランザクションの原子性による一貫性のあるアプリケーションの実現など、分散データベースシステムの提供する機能は、利用者による分散システムの構築効率を大幅に改善することができる．

このような背景をふまえて筆者らは分散データベースシステムを開発した．本稿では、沖電気が開発した分散データベースシステムであるＤＲＥＡＭシステムを紹介する．

2．ＤＲＥＡＭシステムとしての位置づけ

ＤＲＥＡＭの全体像を図１に示し、基本的な開発方針について説明する．

（１）集中型データベースシステムREAM[沖研89b][KANK89]との関係

分散データベースシステムＤＲＥＡＭで、集中型データベースシステムREAMは、そのカーネルの位置に存在する．REAMでもＤＲＥＡＭでもアプリケーションプログラムとのインタフェースは基本標準ＳＱＬである．従ってＳＱＬで記述された集中型データベースシステムーREAM上のＳＱＬで記述されたアプリケーションプログラムはそのまま分散データベースシステムーＤＲＥＡＭへ変更せずに継承できる．ＤＲＥＡＭ内のどのサイトのアプリケーションプログラムも任意のサイトのデータベースが、その所在に関係なくアクセスできる．これにより利用者のプログラム資産の継承ができるだけでなくデータベース資産もＤＲＥＡＭのネットワークワイドな世界で継承／共有できる．

（２）ＤＲＥＡＭ本体

ＤＲＥＡＭはネットワーク上に分散したデータベースへの検索及び更新に対して分散透過性を保証する．ＤＲＥＡＭ内に存在する複数のデータベースはあたかも１つのデータベースのように問い合わせ処理、トランザクション管理が行われる．ＤＲＥＡＭ内に存在する複数のデータベースサイト間で通信を行うための分散データベースアクセスプロトコルが存在する．

（３）ＤＲＥＡＭと他社ホストとの関係

ＤＲＥＡＭは外部からは一つの集中型データベースシステムに見える．他社ホストとはクライアント／サーバの関係で接続される．プロトコルとしては今後ＩＳＯで標準化されるＲＤＡ[RDAG][RDAS]等を予定している．

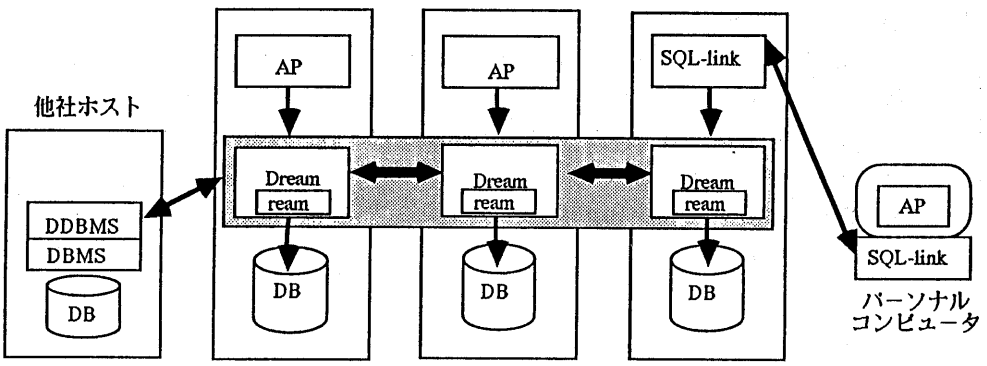

図１ DREAMシステム

（４）ＤＲＥＡＭとＰＣとのリンク

オフィスで用いられるＰＣ（パーソナルコンピュータ）をＤＲＥＡＭシステムへ簡易に接続するためにＳＱＬ－ｌｉｎｋが提供される。国際標準化されつつあるＯＳＩ／ＲＤＡは負荷が大きくＰＣへ実装するにはやや難がある。本プロトコルは自社ＰＣだけでなく他社ＰＣにも実装可能と思われる。ＰＣ内のアプリケーションプログラムはＳＱＬ－ｌｉｎｋによりＤＲＥＡＭ内のデータベースへアクセスすることができる。

３．ＤＲＥＡＭ機能概要

３．１　分散に対する透過性

（１）透過性のレベル

分散データベースシステムでは、分散に対する透過性が重要である。分散に対する透過性にはいくつかの種類がある。

格納位置に対する透過性は、利用者が格納位置を明示的に記述する必要なく、他のサイトのデータベースを集中型のシステムの場合と同様にアクセスできることをいう。移動に対する透過性は、位置に関する動的な透過性とでもいうべきもので、性能改善やデータ量の増大などの理由でデータを他サイトに移動した場合にもシステムが位置の透過性を保証するものをいう。分割に対する透過性は、一つの表を列や行の方向に分割して物理的には各断片を異なるサイトに格納することができ、さらに利用者は分割を意識せず利用できることをいう。重複に対する透過性は、論理的に一つのデータが物理的には複数のサイトに格納され、利用者が複製の存在を意識せず利用できることをいう[疋田89]。

ＤＲＥＡＭでは現在、格納位置に関する透過性と移動に関する透過性を実現している（表２）。

表２　ＤＲＥＡＭの分散透過性

分散透過性	DREAMでの実現
格納位置に対する透過性	○
移動に対する透過性	○
分割に対する透過性	×
重複に対する透過性	×

（２）表名の扱い

利用者がシステム内の各サイトに格納された表を用いるためには、利用する全ての表が一意に識別されなければならない。このためには、以下の二つの方法が考えられる。

(a)表名をシステム内で一意なものとする。
(b)表名にサイト名を附加したものを一意名とする。

(a)では表名は格納位置に対して透過なものとなる。しかしながら、ボトムアップに構築されるシステムでは、表名は各サイト独立に用いられてるので他のサイトと衝突して一意になっていない可能性がある。このため、各サイトでは表名を一意なものに変更しなければならない。これに対し(b)では、表の変更の必要はなく、新たな表の生成においても各サイトの自律性が保たれる。Dreamでは(b)によってシステム内の表を一意に識別する。例えばサイトＳの表Ｔは一意名Ｓ：Ｔによってシステム内で一意に識別できる。

この一意名はサイトの情報を含んでおり、これだけではアプリケーションプログラムに格納位置の透過性が提供できない。そこでアプリケーションプログラムは格納位置に透過な別名を使って記述し、システムはこの別名を「サイト名：表名」の一意名に変換しアクセスする。変換には分散データに関するデータディレクトリを用いる。

データディレクトリを管理する上では、サイト名を誕生サイト[WIDA82]と解釈するか、格納サイトと解釈するかの二つの方式がある。表の移動が無い場合は、両者は同一である。表の移動に対する透過性を実現する上では誕生サイト方式が更新量が少なくてすみ、別名に関するデータディレクトリは更新不要であるという利点を持つ。しかし、誕生サイトがシステムから離脱する場合には誕生サイトで発生した表の管理ができなくなる。ＤＲＥＡＭではサイトの追加・離脱が容易にできることを重視しており、格納サイト方式をとっている。表の移動に対しては、移動した表に対するディレクトリを書き換えることで移動の透過性を実現する。

３．２　問い合わせ処理

（１）実行手順

図３は利用者のアプリケーションプログラムの実行手順を示す。

ＳＱＬ文は、ＳＱＬ埋め込みホストプログラムの規則[SQL]に従い、親言語で記述されたアプリケーションプログラム中に埋め込まれる。このプログラム

はＳＱＬプリプロセッサにより対応する手続き呼び出しを含む親言語のソースプログラムに変換され、さらに親言語の処理系によって実行形式プログラムが作成される。親言語としてはＣとCobolがある。Ｃへの埋め込みに関しては現在の規格[SQL]にないので現在標準化されつつあるＳＱＬ２[SQL2]の規格に準じている。ＳＱＬプリプロセッサなどのプログラミングシステムはREAMとまったく同様のものを用いることができる。この手順はREAMおよびＤＲＥＡＭで共通である。

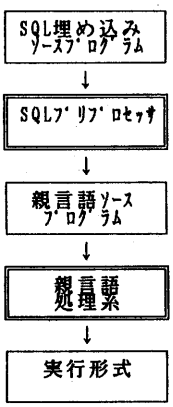

図３　アプリケーションプログラムの作成

作成された実行形式プログラム中では、問い合わせ中の表に対する格納位置は未解決で、データベースの所在に関わる最適化などの処理は行われていない。この問い合わせは実行時にＤＲＥＡＭに渡され、その時点で表の格納位置の決定とそれに伴う問い合わせの処理が行われる。

この方法では、すでにREAM用に開発されたアプリケーションプログラムは再コンパイルすることなく分散環境で実行できる。たとえばREAMを用いて作られていたシステムで１サイトにデータベースが入らなくなった場合は、サイトを一つ追加し、表の幾つかを新しく追加したサイトに移せばよく、REAMのアプリケーションプログラムは変更なく利用できる。

現在、JIS/SQLの第１水準の問い合わせを複数サイトの表について処理することができる。

（２）問い合わせの分解と最適化

アプリケーションプログラムから渡された問い合わせについて、ＤＲＥＡＭは、まず中に現れる表の格納位置の解決とスキーマ情報の取得をデータディレクトリを参照して行う。

次に表の格納位置およびスキーマの情報に基づき、複数の表を含む問い合わせＱを、各サイトに閉じた表の検索だけを行う複数の問い合わせＱ１、Ｑ２に分解し、各々のサイトへ送る（図４）。

ＤＲＥＡＭは、ネステッドループ法を基本として問い合わせを処理する。このためインナループのジョイン条をは、タプルサブスティテューション[WOY076]のためにパラメタを用いる条件に置き換える。図４では表Ａがアウタ表Ｂがインナである。

問い合わせは異なるネスティドループに分解する事が可能である。図４では表Ｂをアウタ、表Ａをインナ分解することができる。ＤＲＥＡＭは可能なループの順序について最適なものを選択する。最適化はインデックスの有無、データの自サイト／他サイト、検索条件のセレクティビティとおよそのタプル数、転送／検索データ量などにより処理のコストを評価して行う。

（３）問い合わせの実行

分解および最適化された問い合わせは、ＤＲＥＡＭではパイプライン並列[HIKH85][斉坂89][LUCA85]に処理される。サイト１で表Ａが検索され、問い合わせの出力となる属性A.aとジョイン属性A.dが転送される。サイト２では転送されたジョイン属性A.dの値を用いてＡとＢのジョインに成功する行を見つける。これによりA.aとB.bが求まり結果の行を生成する。この間サイト１では表Ａの次の値の検索をサイト２の処理と平行して行う。

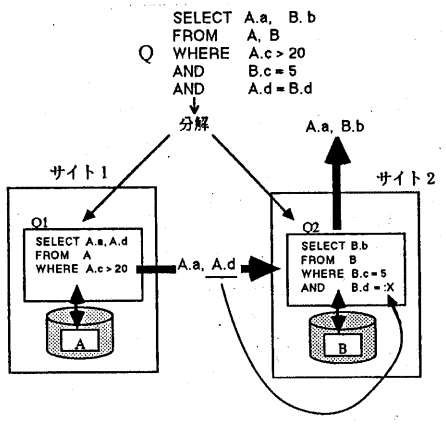

図４ 問い合わせ処理

実際には、サイト１とサイト２の転送の同期をとる必要があり、サイト１からの行の転送は、サイト２からのフェッチ要求を契機として行われる。パイプラインを実現するために、サイト２が実際に処理する行のフェッチ要求は、先行してサイト１に送られ、サイト２の処理と平行してサイト１の処理と転送が行なわれる。また、転送は行単位の通信オーバヘッドを削減するために複数行をブロック化して行う。

Dreamの問い合わせ処理では、フェッチの先行と転送のブロック化は問い合わせ処理の実行における通信オーバヘッドを大幅に削減している。パイプラインを用いた処理は、問い合わせ処理効率に大きな影響を与える。

３．３　トランザクション管理

（１）分散トランザクション

複数サイトにまたがるトランザクションの一貫性を保証するために、複数サイトの更新の同期を制御する機能が必要である。

ＤＲＥＡＭでは２相コミットメカニズムを用いてトランザクションの更新における原子性を保証する。２相コミットは障害発生時にコミット／ロールバックが決定できないサイトが発生する可能性があるという欠点がある。しかし、これを回避するために相を増やすアプローチ[SKEE81]は、正常動作する大部分のトランザクションにとって不利益となるため得策ではない。ＤＲＥＡＭでは、コミット／ロールバックを決定できないような障害を起こす可能性は低いと考え最悪の場合はＤＢＡの判断により決定するようにしている。

（２）コミットメントの最適化

トランザクションは、実際には各サイトで代理実行されるサブトランザクションの集まりによって実現される。各サブトランザクションはデータの検索、更新を行う。また、トランザクション発生サイトにはトランザクションのコミット時にこれらサブトランザクションを２相コミットで制御するためのコーディネータがある。

コーディネータは、実際に更新を行ったサブトランザクションを管理している。コミット時には、サブトランザクションの更新の状態に応じて以下のようなコミットメントプロトコルの選択を行う(表５)。

（ａ）更新サブトランザクションなしの場合

どのサブトランザクションも更新を行っていなければトランザクションはread only transactionであるので２相コミットの必要はない。この場合には全サブトランザクションをただちに終了させてよい。

（ｂ）更新サブトランザクションが１個の場合

１サイトだけが更新されている場合も他サイトとの矛盾は発生しないので２相コミットの必要はない。全サブトランザクションを終了させ、更新サブトランザクションのコミットの正否がトランザクションのコミットの正否となる。

（ｃ）自サイトと１他サイトが更新されている場合

２サイト以上が更新されているので両者に矛盾が生じないように２相コミットが必要となる。ただし、次のようにしてネットワーク上の通信回数を減らすことができる。まず、自サイトを２相コミットの第１相の終わった状態、いわゆるセキュアな状態にする。これが成功したら他サイトをコミットする。他サイトのコミットが成功したら自サイトをコミットし、失敗したら自サイトをアボートする。他サイトのコミットの正否がトランザクションのコミットの正否となる。

（ｄ）２他サイト以上が更新されている場合

この場合には通常の２相コミットプロトコルに従う。

分散システムでは多くのトランザクションが局所的なものとできる。コミットメントの最適化により、単純なトランザクションが不必要な通信コストを必要としないにすることは重要である。

表５　ＤＲＥＡＭのコミットメントプロトコル

更新サブトランザクション	コミットメントプロトコル
なし	１相
１サイト	１相
自サイト＋１他サイト	２相*
２他サイト以上	２相

*通信は１相（本文参照）

3．4　利用者の管理

分散システムでは、図６のようにあるサイト１の利用者sakamotoと別のサイトの同一名の利用者sakamotoが同一人物でない場合もある．またsakamotoとsakaのように別の名前であっても同一の人物である場合もある．

ＤＲＥＡＭでは利用者管理は完全にローカルな処理として行なわれる．すなわちあるサイトに関する利用者の登録や認証は必ずそのサイトで行なわれ他サイトの管理者や他サイトのＤＲＥＡＭによって行なわれることはない．これにより基本的に各サイトの利用者はたとえ同一名であっても異なる利用者を表わすことになる．

加えて、複数のサイトに利用者登録されている利用者が、サイトにまたがった処理においては同一利用者として扱われるための管理を行う．各サイトでは他のサイトへアクセスするとき各個人がどの利用者名によってアクセスすべきかをデータディレクトリとしてとして管理する．例えばサイト２にも利用者登録sakaをもつサイト１の利用者sakamotoがサイト２にもアクセスする場合、サイト１のデータディレクトリにサイト２では利用者sakaとしてアクセスできることを登録しておく．利用者sakamotoの利用中にサイト１からサイト２へのアクセスが発生した場合、ＤＲＥＡＭは起動のパラメタとして利用者名sakaおよびパスワードを送り、そのサイトでのアクセスの権限をsakaに設定する．

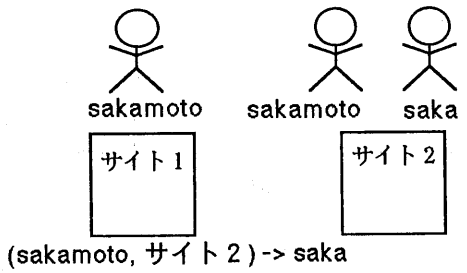

図6 利用者名の変換

4．処理構成

ＤＲＥＡＭは図７に示すように構成される．大きく分けると、ＳＱＬプロセッサ（実行時ルーチン）、クライアント、クライアントから起動される複数のサーバ、およびネットワークシステムからなる．

また図４に現れている以外に、システムの起動とシステムワイドなリカバリを行うスタートアップのプログラム、表の移動やブラウジングを行うユーティリティプログラムなどがある．

以下では、図７に従ってＤＲＥＡＭの主要な部分について構成および機能を説明する．

4．1　ＳＱＬプロセッサ

ＳＱＬプロセッサは、アプリケーションプログラムとリンクされ、ＳＱＬの機能の一部を実現する．処理の構造はREAMと同じになっており、また、同じプログラムが使える様にすることで、既存のREAMのアプリケーションプログラムを互換にしている．

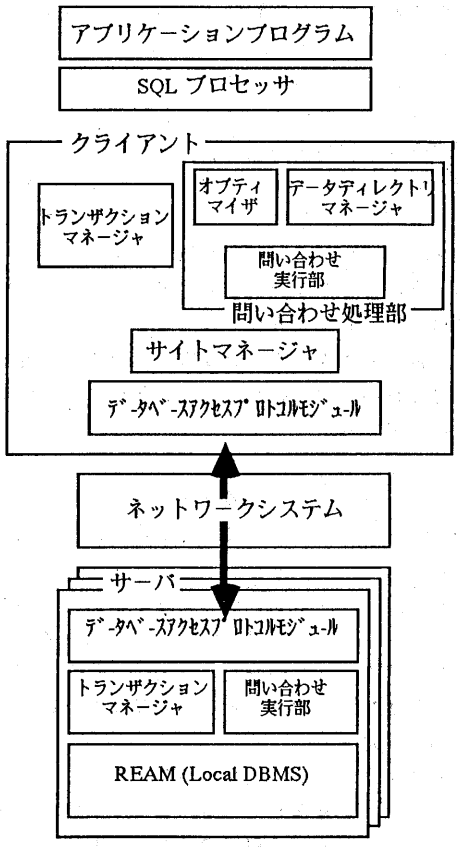

図 7 処理構成

4．2　クライアント

クライアントは、分散したデータへのアクセスに関して制御を行なう。問い合わせの分解、最適化、問い合わせの実行などを行い、コミットメント時のサブトランザクションに対するコーディネータとしても働く。クライアントは利用者のデータベースセッションに対応して各利用者毎に起動される。

クライアントおよびサーバが分散データベースアクセスプロトコルにより協調動作することで、分散データベース管理システムとしての機能が実現される。

クライアントは以下の機能モジュールから構成される。

（１）　問い合わせ処理部

オプティマイザ、データディレクトリマネージャ、問い合わせ実行部の各モジュールで処理される。

・オプティマイザ

ＳＱＬのカーソル宣言に対応して問い合わせの分解と最適化を行い、問い合わせ実行部の入力となる実行用のプランを生成する。これは、ループのネストの順番およびタプルサブスティテューションを行なう列の情報などからなる。

処理には行の長さや型など表のスキーマが必要となる。オプティマイザはデータディレクトリマネージャを呼び出すことによって必要な情報を得る。

・データディレクトリマネージャ

データディレクトリを検索して指定された表名の解釈を行い格納サイトを決定する。また、オプティマイザへ表のスキーマを提供するために、格納サイトへのスキーマのアクセスを行なう。

・問い合わせ実行部

クライアント側のオプティマイザの出力である、実行用のプランに従って各サイトのカーソルからの行のフェッチ、パラメタの転送を制御する。また、ＯＲの条件の評価もここで行う。さらに、ＳＱＬのＵＰＤＡＴＥ文など更新操作のある文を処理する場合にはトランザクションマネージャを呼び出し、更新の行われたサイトを記録しておく。

（２）トランザクションマネージャ

トランザクションの管理を行う。クライアント側のトランザクションマネージャはstart_transaction, commit_transaction, abort_transactionの利用者トランザクションを制御する制御プリミティブを提供する。また実際に更新の行われたサイトを記録するためのプリミティブも持つ。利用者からのcommit_transacionが発行されると、コミットメントプロトコルの選択を行いコミットメント制御を行う。

（３）サイトマネージャ

各マネージャは他のサイトへアクセスする必要が起きると、このマネージャを呼び出す。ここでは要求に従ってサーバの起動／停止を行う。サーバの起動時には、起動に必要な利用者名、パスワードをデータディレクトリより検索しこれを用いる。

（４）データベースアクセスプロトコルモジュール

クライアントは分散データベースアクセスプロトコルに従ってサーバに対する要求を送り、またサーバからの結果を受け取る。ここでは、このためのＰＤＵの生成と解釈を行う。

また分散データベースアクセスの効率化のためのプロトコル、例えばブロック転送された行のデブロッキング、フェッチの先行の制御を実現し、問い合わせ処理部が処理しやすいように行単位のインタフェースに変換している。

4．3　サーバ

サーバはローカルなデータベース管理システムとしてREAMを持ち、各サイトでの実際のデータベースアクセスを行う。またコミット時にはコーディネータであるクライアントに対してサブトランザクションとしての制御を行なう。サーバはクライアント毎に各サイトで起動される。

（１）データベースアクセスプロトコルモジュール

クライアントとの通信のためのＰＤＵの解釈と生成を行う。また行の転送においては行をブロッキングする処理を行う。

（２）問い合わせ処理部

サーバ側の問い合わせ処理は基本的にREAMの機能として実現されておりインタフェースの整合だけを行っている。

（３）トランザクションマネージャ

サーバ側のトランザクションマネージャはサブトランザクションに関する制御を行う。start,commit,abortのプリミティブに加えて、２相コミットに対応するsecureプリミティブを持つ。これらの機能はREAMの機能として予め実現されており、実際の処理はREAMの内部で行われる。ただし２相コミットのためのロギングはここで行う。

4．4　ネットワークシステム

ネットワークシステムはクライアント／サーバ間のプロセス間通信を実現する。クライアントとサーバはセキュリティと信頼性のためにコネクション型の通信を行う。ここでコネクション型とは通信相手に関して状態を保持することにより、再送や相手の特定が可能な通信をさす。

5．　おわりに

本稿では、ＬＡＮを中心にして構成したＤＲＥＡＭシステムについて紹介した。ＤＲＥＡＭは、OKITAC8300ミニコンピュータと10MBPSイーサネット型のLANであるOKINET2000/model20上でインプリメントされている。

今回の実現したシステムにより、筆者らが目的とした第一ステップの分散データベースシステムが実現できたものと考えられる。

今後は、実際の運用を通してシステム保守・運用上の実績を積み重ねて行きたい。今後の課題としてネットワークの広域化のためにはグローバルデッドロック検出／回避の必要性と適切な実現方式、分散データのバックアップ、障害復旧などの問題点を解決して行かなければならない。

[参考文献]

[沖研88a]林、正田、野原、三上,"OKITAC8300アーキテクチャーアプリケーションアーキテクチャ",沖電気研究開発 vol.55, No.4, pp12-14,1988

[沖研88b]正田、川上,"OKITAC8300基本ソフトウェアの開発－データベース",沖電気研究開発 vol.55, No.4, pp59-64,1988

[斉坂89]斉藤,坂本,川上,正田,"パイプライニングによるサイト間結合演算の並列処理",情報処理学会第３８回全国大会 pp.885-886

[正田89]正田、"分散型データベースシステム入門"、オーム社、１９８９

[HIKH86]Hikita,S.,Kawakami,S.,Haniuda,H.,"A Stepwize Approach to Distributed Database Systems by Database Machines",Proc. 1985 ACM SIGSMALL Symp.Small Syst.,pp.18-24,1985

[KANK89]Kawakami,S., Nakayama,T., Kashiwabara,K.,Hikita,S.,"REAM:An SQL Based and Extensible Relational Database Management System",Proc. International Symposium on Database Systems and Advanced Applications, pp.166-170,1989

[LUCA85]Lu,H., Carey,M.,"Some Experimental Results on Distributed Join Algorithms in a Local Network",Proc.VLDB85,pp.292-304,1985

[RDAG] ISO/IEC/JTC1/SC21/WG3,"General Remote Database Access Service and Protocol", ISO/JTC1/SC21 N3606, June 1989

[RDAS] ISO/IEC/JTC1/SC21/WG3,"Remote Database Access - SQL Specialization,ISO/IEC/JTC1/SC21 WG3 N844,, June 1989

[SKEE81]Skeen,D.,"Nonblocking Commit Protocols",Proc. ACM-SIGMOD81,pp133-142,1981

[SQL] 日本工業標準調査会,"データベース言語ＳＱＬ",JIS X 3005, 1987

[SQL2] ISO/IEC/JTC1/SC21/WG3,"(ISO-ANSI working draft)Database Language SQL2 and SQL3",ISO DBL FIR-3, July 1989

[WIDA82]Williams,R.,Daniels,D.,"R*:An Overview of the architecture",Improving Database Usability and Responsiveness, Academic Press, 1982

[WOYO76]Wong,E., Youssefi,K., "Decomposition - A Strategy for Query Processing",TODS vol1,No.3, 1986