データベース管理システム G-BASEにおけるデータ辞書の複写分散

3Q-10

平岡 昭夫，飯沢 篤志

(株) リコー ソフトウェア研究所

1．はじめに

我々は、データベース管理システム G-BASE[1] をNFS[2] によって作られる分散ファイルシステムの環境に適応させることによって、簡単な分散データベース環境を提供し、その実用性の評価を行なった。[3], [4] 結果として以下のような問題点があることがわかった。

a．データ辞書（以下、DDと呼ぶ）がネットワーク上に1つしか存在しないためDDのあるサイト（以下、サーバサイトと呼ぶ）に対してアクセスが集中してしまう。さらにサーバサイトが正常に動作していない場合、たとえデータベースの実体がローカルにあっても、そのデータにアクセスできない。

b．ロックマネージャ（以下、LMと呼ぶ）がネットワークで1つしか動作していないためにDDの場合と同様の問題が生じる。ただし、LMは原理的にはどこのホストで起動されていてもかまわないので、運用によって問題を避けることも可能。

c．OS（ここでは、UNIX）に対するファイル操作のレベルで他サイトと通信を行なうため、通信コストが非常に高い。

これらの問題点のうちaを軽減するためにDDの複写分散機能をG-BASEに追加することを検討した。2章では複写分散に関する基本的な設計方針を、3章ではG-BASEに対する実際の改良について述べる。4章でDDに対する更新処理の流れを記述し、最後にに5章で現状と今後の課題について考察する。

2．データ辞書の複写分散

DD（の全体、または一部が）が自サイトのファイルシステム上にあれば、サーバサイトに頼らず自サイトのデータベースにアクセスすることができる。サーバサイトへのアクセスの集中を軽減することにもなる。そこでG-BASEに対して、任意のサイトにDDの複写を保持する機能を持たせる。この機能を『データ辞書の複写分散機能』と呼ぶ。

今回の改良では、まったく同じDDが複数のサイトに存在するモデルを実現する。複数あるDDに親子関係はなく、すべて対等であるものとする。また、それぞれのサイトのDDに対してデータ定義情報を参照／追加／削除／更新できるようにする。

自サイトにDDが存在する場合には、従来どおり、そのDDに対してデータ定義情報の参照／追加／削除／更新を行なう。他のサイトにも同じDDが存在する場合にはシステムが自動的にDDの変更を伝えて更新の同期をとる。さらに、DDが存在しないサイトからもデータ定義情報を参照できるようにする。以下、自サイトにDDが存在するサイトをスキーマ・サイトと呼ぶ。DDが存在せず、他のサイトのDDを参照するサイトをクライアント・サイトと呼ぶ。

今回はデータベースの複写分散は許さない。データベース自体をサイト間で共有する場合には、NFS（ネットワーク・ファイルシステム）等の助けをかりなければならない。同様に、データ定義情報の変更によってデータベースの再構成が起こる場合には、変更したサイトから再構成の対象となるデータベースに関連するすべての物理ファイルに対してアクセスできなければならない。

Design of replicated data dictionary
for G-BASE database management system
Akio HIRAOKA, Atsushi IIZAWA
Software Research Center
Ricoh Co., Ltd.

DDおよび、データベースを含めたデータ実体に対する同時実行は従来どおりシステムに唯一存在するLMによって制御する。DDの更新同期にもLMの同時実行制御機能を利用する。

3．G-BASEの改良

現在のNFS対応 G-BASEに対して以下のような改良を行ない、DDの複写分散機能を実現する。

a．データ辞書の登録

G-BASEではデータ辞書も他のデータベースと全く同じ形式を持っている（アクセスの方法が他のDBとは異なる）。したがって、各サイトに存在するDDを異なる名前のDBとして登録することが出来る。このDBをオブジェクトIDからDDであることが判別できるようにする。

b．ロックメカニズムの変更

システム中に複数個あるDDの更新同期をとるため、ロックメカニズムを改良する。名前の異なる複数のDDに対するロックを1つの仮想DDに対するロックであるようにマッピングする機能をLMに持たせる。この改良よって、複数個あるDDの変更に対する同時実行制御に、DDが1個だけの時と同じメカニズムを使用することができる。

c．更新同期機能

G-BASEではDDに対する操作もデーマネージャ（DM）に対する通常のデータベース操作によって実現されている。ただし、DM内部ではDDと通常のデータベースは区別されている。たとえば、スキーマ情報を参照する場合にはキャッシュ・メカニズムを使用し、変更に対してはデータベースの自動再構成を行なう等の処理が必要となる。データベース管理者がDMを介してDDを変更する場合に、他のスキーマ・サイトにあるDDの変更も同時に行なうメカニズムを追加する。DMはスキーマ・サイト上にある常駐プロセス (DDserv) とともに2相コミット・プロトコルを実行することで更新の同期をとる。(詳細は4章で記述する)

d．DDserv

スキーマ・サイトに DDserv と呼ぶ常駐プロセスを用意する。各 DDserv は自ホストのDDをオープンしてスキーマ情報をキャッシュしている。クライアント・サイトからの要求に対して、スキーマ情報を返す。さらに、DDserv はお互いに協調して、DDの変更を他のスキーマ・サイトに伝え更新の同期制御を行なう。

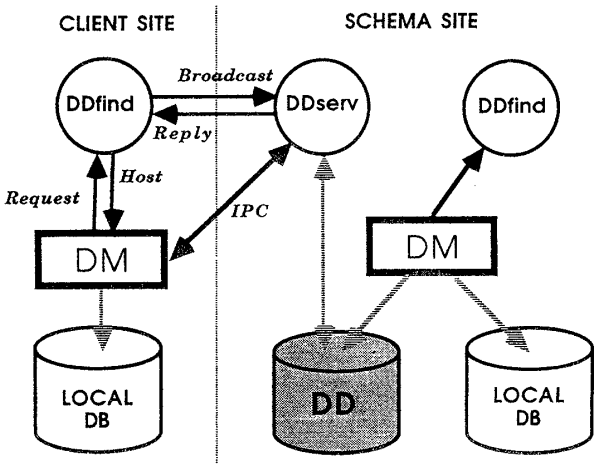

Fig.1 Reference to DD

e．DDfind

すべてのサイトにDDfindと呼ぶ常駐プロセスを用意する。DDfindは自サイトにＤＤが存在するかどうかを判断し、もしなかった場合にはサービスしてくれるスキーマ・サイトをさがし、そのサイト上のDDservプロセスとの通信路を設定する。

データ辞書参照時の処理（Fig.1 参照）

(1) DMはまずDDfindにアクセスする。スキーマ・サイトでは自サイトにＤＤが存在することを知り、自サイトのＤＤに対して従来どおりの参照を行なう。

(2) クライアント・サイトではDDfindがすべてのホスト上にあるDDservに対してリクエストをブロードキャストする。最初に応答があったDDservをサーバとしてDMに知らせる。

(3) クライアント・サイトのDMはDDfindから得られたサーバに対して、RPC (Remote Procedure Call) を実行してスキーマ情報を得る。

スキーマ・サイトでもDDservに障害があった場合にはＤＤの更新同期が保証できないので、他のサイトのＤＤを使用するようにDDfindが制御する。

f．統計情報

従来ＤＤの中に格納していた統計情報を、別のメカニズムで収集するように改良する。各サイト上にDDstatと呼ばれる常駐プロセスを用意し、DMからの統計情報をホスト単位で集めるように改良する。システム全体での統計情報は、すべてのサイト上のDDstatが保持している情報を集めることで得られる。

4．データ辞書の更新同期

ＤＤに対する更新同期のメカニズムを更新時、起動時それぞれについて記述する。

【更新時】（Fig.2 参照）

(1) DMは始めに、対象データベース（ＤＢ）に排他モードのロックをかける。（他のDMが、対象ＤＢをオープンしていないことを保証する）

(2) DMはＤＤをＳＩＸモード* でオープンする。（他のDMが、ＤＤを更新しないことを保証する）

(3) トランザクションをオープンして、関連するスキーマ情報を更新する。

(4) トランザクションをコミットする前にスキーマに対する変更を、すべてのスキーマ・サイトのDDservに、PREPAREメッセージとして送る。

(5) スキーマ・サイトのDDservは自サイトのＤＤをオープンし送られた変更を適応した後、コミットする前に、DMに対してREADYメッセージを返す。

(6) READYメッセージを受け取ったDMはトランザクションをコミットし、対象ＤＢの再構成を行なう。

(7) コミットが完了したら、READYが返ってきたDDservに対してCOMMITメッセージを出す。もし、コミットまたは再構成に失敗したら、READYを返したDDservに対してABORTメッセージを送りトランザクションをロールバックする。（自サイトのＤＤに対してもロールバックを行なう）

(8) READYが返ってこない、スキーマ・サイトについては変更の内容を自分のサイトのDDservに送って、DMは変更操作を完了する。

ＳＩＸモード：自分は参照／更新ともにできるが、他人は参照はできるが更新はできない。

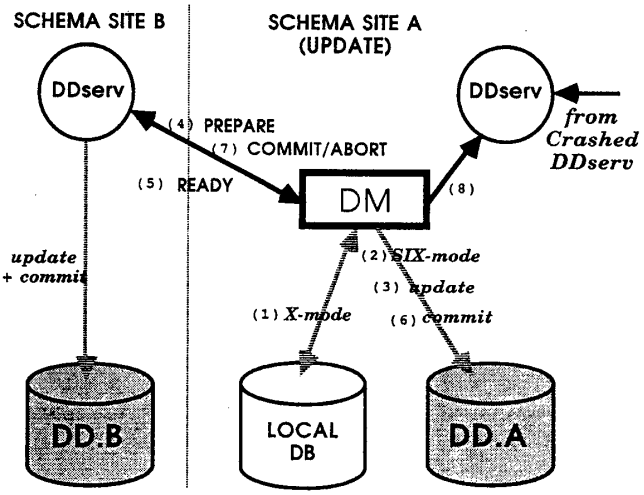

Fig.2 Update DDs

【起動時】

(1) 各サイトでマシンの起動時には、DDservとDDfindを常駐プロセスとして起動する。

(2) 起動されたスキーマ・サイトのDDservは他のスキーマ・サイトのDDservに問い合わせて、ＤＤに変更があったかどうかを知る。

(3) 変更があった場合には、DDservのログを見て自ホストのＤＤについても変更を反映する。（起動時には他のDMがＤＤにアクセス出来ないように制御しなければならない）

ＮＦＳでデータベースを共有している場合、変更対象ＤＢのあるサイトがダウンしていれば、【更新時】で述べたスキーマの変更／再構成は不可能である。したがってスキーマ・サイトが単独で立ち上がって（他のスキーマ・サイトでのＤＤの変更を知らず）古いデータ辞書に基づいて自分のサイトのデータに操作を行なったとしても問題はない。

5．現状と今後の課題

今回のＤＤの複写分散機能の実現によって、他サイトの稼働状態に左右されることなく、自ホストにあるデータベースであれば、通常のデータ定義／操作を行なうことができるようになる。また、ＤＤに対する参照によって起こるサーバサイトへの通信コストが軽減される。

現在、ＤＤの複写分散機能を持つようにDMを改良中である。DDservは基本的に、従来のDMの機能をそのまま継承したサーバ・プロセスとして実現するつもりである。

今後の課題としては、前述のＬＭの負荷分散が最大の検討項目である。さらにＤＤだけでなく一般のＤＢについても同様の複写分散を実現したいと考えている。分散化による性能評価も継続して実施していく予定である。

【参考文献】

[1]（株）リコー編：G-ＢＡＳＥシステムマニュアル

[2] R.Sandberg, D.Goldberg, S.Kleiman, D.Walsh, B.Lyon "Design and Implementation of the Sun Network Filesystem", USENIX Summer 1985, pp563-568

[3] [4] 飯沢篤志、平岡昭夫：データベース管理システム G-ＢＡＳＥのＮＦＳ対応（１）,（２）情報処理学会第３６回全国大会, Mach 1988