知的分散システム IDPS における データベース処理機構 IDPS-DB に関する研究

原嶋　秀次† 　大森　匡‡

†（株）東芝　ソフトウェア技術センター 〒212-8582 神奈川県川崎市幸区小向東芝町 1
‡電気通信大学大学院情報システム学研究科　〒182-8585 東京都調布市調布ヶ丘 1-5-1
E-mail: †shuji.harashima@toshiba.co.jp, ‡omori@is.uec.ac.jp

あらまし　知的分散システム IDPS は東芝が 1990 年代に開発・商用化した自律分散システムであり、主に制御用を想定した耐障害分散処理システムである。IDPS 上で動作する自律分散データベースシステムとして原嶋らは、’87-‘97 まで IDPS-DB と呼ばれるデータベース処理機構を設計・開発してきた。本稿では、IDPS-DB の設計をまとめ、その特徴的な技法として、自律分散システムに適したデータモデル、問い合わせ処理方式、トランザクション処理制御の各設計を述べる。また、実際に運用をおこなった結果から、自律分散システムにおけるデータベースシステムの要件を述べ、今日の広域データ管理技法への示唆としたい。

キーワード　分散データベース，自律分散システム，知的分散システム

Database system IDPS-DB on Intellectual Distributed Processing System IDPS

Shuji HARASHIMA† 　Tadashi OHMORI‡

† Corporate Software Engineering Center, TOSHIBA

1 Komukaitoshiba-cho, Saiwai-ku, Kawasaki-shi, Kanagawa, 212-8582 Japan
‡ Graduate School of Information Systems, The University of Electro-Communications,

1-5-5 Chofugaoka, Chofu-shi, Tokyo, 182-8585 Japan
E-mail: †shuji.harashima@toshiba.co.jp, ‡omori@is.uec.ac.jp

Abstract　Intellectual Distributed Processing System (IDPS) is the autonomous distributed system developed in TOSHIBA which is a fault-tolerant system mainly used as a control system. This paper summarizes a design of the IDPS-DB which is a data management system on the IDPS. A data model of IDPS-DB which is suitable for autonomous distributed processing systems, query processing, and transaction management on the IDPS are also described. An evaluation of those designs is shown based on the practical use. Finally, we show the necessary properties to realize the flexible and reliable database system on an autonomous distributed system.

Keyword　Distributed Database，Autonomous Distributed System，IDPS

1. はじめに

1980 年代、コンピュータシステムは LAN の普及と共に業務インフラとして急速に普及した。負荷分散や信頼性向上などの研究が各所で行われ実用化された[1]。自律分散システムはこのような背景において提案されたシステムアーキテクチャで、集中管理部が存在せず自律的な機能要素の動的な連携で、求められる機能を実現する。業務インフラとして使われるシステムの機能要件や負荷をあらかじめ明確にするのは現実的ではなく、段階的な開発に適したシステムアーキテクチャでもある[2],[3]。知的分散システム IDPS は東芝が 1990 年代に開発した自律分散システムである[4]-[9]。本稿では、IDPS 上で動作するデータベース機構として原嶋らが’97 まで開発した IDPS-DB と呼ばれる自律分散データベースシステムの設計を述べる。特に、その特徴的な技法として、自律分散処理に適したデータベース処理用のプロセス構成、問い合わせ処理方式、トランザクション処理方式を述べる。そして、実際の運用結果から、これらの設計の有効性や留意すべき点を述べて、現在の分散データ管理技法への示唆としたい。

以下、２章で IDPS の概要を述べ、３章で IDPS-DB の設計を述べる。４章では IDPS-DB の高信頼化技法を

紹介し、5 章でシステム適用を通しての評価を述べる。

2. 知的分散システム IDPS

知的分散システム IDPS は東芝において研究・開発した自律分散システムで、高い信頼性と可用性と共に柔軟性のあるシステムを構築可能なプラットフォームである。IDPS-OS は、IDPS を構成する分散オペレーティングシステムであり、オブジェクトの多重化や高信頼な同報メッセージ通信を提供する[7],[8]。

2.1 知的分散システム IDPS の基本構成

図 2.1 に知的分散システム IDPS の構成を示す。ハードウェア的には同一 LAN 下に接続された複数のコンピュータで構成される。各コンピュータには IDPS-OS が搭載され、サイト内のオブジェクト管理や後述する LAN 上の高信頼放送通信を実現する。知的分散データベースの研究開発期間で用いたハードウェアは次の通り。

コンピュータ：

東芝パソコン J-5030(CPU：80386、OS：C-DOS)

産業用コンピュータ：G200、G300(CPU：80386、OS:RTUX/386)(最終的には SUN OS 上に移植).

ネットワーク：イーサネット 10Mbps

各コンピュータ（サイト）には、自律オブジェクトを搭載する。各オブジェクトは、まわりの状況から自身のとるべき動作を判断し、処理を実行する。新たな機能が必要となった際には、新たなオブジェクトを追加することで、オブジェクト間の連携により、新規機能が実現される。

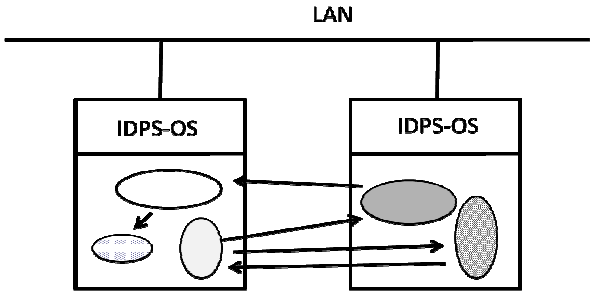


図 2.1　知的分散システム（IDPS）の構成

図 2.2 にオブジェクトの構成を示す。各オブジェクトは、すべてのオブジェクトが同様に有する共通知識と、個別の固有知識からなる。固有知識はオブジェクトに固有な機能を実現するためのデータと手続き群からなる。共通知識は排他制御、負荷分散などのシステム全体の管理に必要なデータと手続き群からなる(各オブジェクトは共有メモリを有する複数のプロセスで構成され、メソッドに対応したプロセスがメッセージに応じて処理を行う構成であった。後に一部はマルチスレッド化された。)。

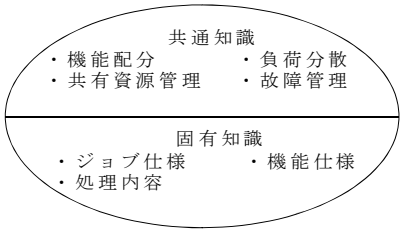


図 2.2　知的分散システムのオブジェクト

図 2.3 に IDPS の構成例と処理の特徴を示す。システムへの処理要求は、オブジェクト間の協力・協調により実行される。

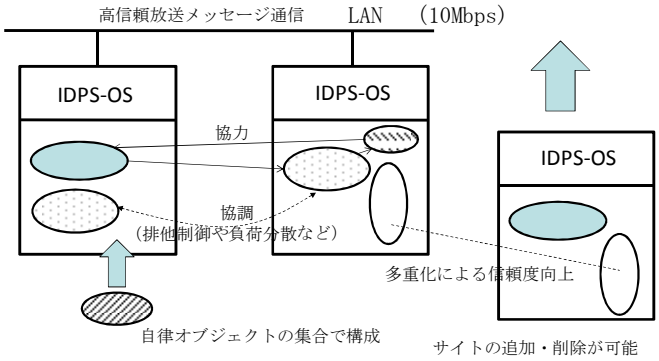


図 2.3 知的分散システム IDPS の構成と特徴

ここで、「協力」とは要求をオブジェクト群が互いに処理結果をやりとりすることにより、遂行することであり、「協調」とはオブジェクト間での処理の競合を調整し、システム資源の有効かつ正しい利用（排他制御、負荷及び機能配分）することである。これらによりオブジェクトの追加や変更、削除、サイト間移動などが局所的な操作で実行できるのみならず、サイトダウンなどのハードウェアの故障に対しても柔軟に対応でき、拡張性、適応性、信頼性に優れたシステムの構築が可能になる。

各オブジェクトの多重度を変更することで、機能単位で信頼度の設定をおこなうことができる。また、システム稼働中のサイト追加やオブジェクトのコピー作成も可能である。

2.2 IDPS における高信頼化技法

2.2.1　オブジェクトの多重化支援

IDPS では複数サイトでオブジェクトを多重化することにより、一部のサイトに障害が発生しても残りのサイトに存在するオブジェクトにより処理の継続が可能となる。

図 2.4 に多重化されたオブジェクト間でのメッセージ通信例を示す。オブジェクト A と C が 3 重化、B が 2 重化されている。A から B および、B から C へのメッセージは多重化されたレプリカから放送で送られ

る。このためいずれかのレプリカに故障が生じても、少なくとも 1 つのレプリカが動作していればシステムは止まることなく動作する。原則、通信は論理オブジェクト名（A,B,C など）指定で行なわれる。レプリカは非同期に動作し、相手オブジェクトの多重度や位置を知る必要もない。

多重化されたオブジェクトのレプリカ間の整合性は、フェイルストップ放送通信機構と、メッセージ選択機構である First-CN-Come*e* によって保証される。フェイルストップ放送通信とは、サイト間の各メッセージを放送通信によって全順序をつけてアトミックに行う機構であり、障害サイトを自動停止させる通信機構である。これを使うことで、複数サイト上で多重化されたオブジェクトの各レプリカは、(メッセージ ID によって同一と判定される)メッセージを、必ず、同じ順番で受信することが保証される。一方、First-CN-Come は、全ての正常なレプリカが、他の多重化されたオブジェクトからの送出された同じメッセージに対してコミットすることを保証する機構である。これら 2 つの機構によって、レプリカ間での整合性と正しい結果を得ることが保証される。

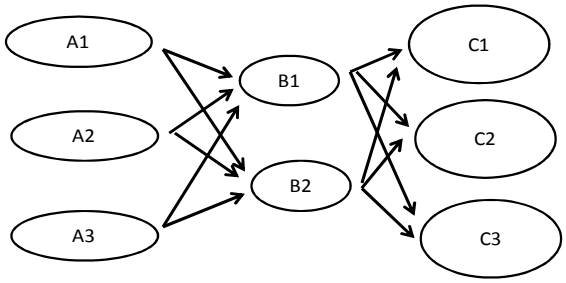


図 2.4 放送通信による多重化オブジェクトの起動

図 2.5　メッセージ ID

具体的には、First-CN-Come では、あらかじめ決められた confirmation number (CN)個の同じ内容のメッセージを受信したときそのメッセージをコミット（受信し処理）する。CN の値はオブジェクトの多重度とは独立して定義することができるので、関連するオブジェクトの多重度に関係なく定義することができる。メッセージコミットまでの時間的な遅れも従来からの多数決を用いた方法より短くすることができる。メッセージを認識するためのメッセージ ID（図 2.5）も、各レプリカ(決定的動作をすることが前提)により、非同期でそれぞれに独立して発行することが可能である。

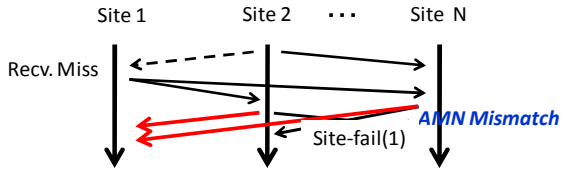
2.2.2　フェイルストップ放送通信機構について

IDPS において、オブジェクトの正しいふるまいを保証するため、LAN で接続された計算機は同じメッセージを同じ順序で受信することが必要である。IDPS-OS ではフェイルストップ放送通信プロトコルを導入している。動作は次の通りである。

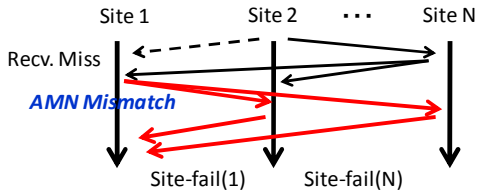
各計算機の通信ユニットは LAN 上のメッセージを常にカウントしており、Accumulated Message Number (AMN)として記憶している。あるオブジェクトがメッセージを送信する際には、そのオブジェクトが存在する計算機の通信ユニットは AMN をメッセージに付加して LAN に放送する。もし、受信したいずれかの計算機で送られたメッセージの AMN と自身の AMN が一致しなければ、どちらかの計算機が正常でないことを意味する。この場合、メッセージを送信した計算機に「site fail」メッセージを送出し、受信メッセージをキューイングする。

いずれかの通信ユニットがあらかじめ決められた一定期間内に事前に決められた数(Pre-Defined Number : PDN)の「site fail」を受信した場合は、自身が故障と判断し処理をただちに停止するか受信に失敗したメッセージの再送を要求する。

PDN の設定によってシステムの信頼性を設定することができる。図 2.6 に IDPS-OS によるメッセージ受信失敗の検出プロセスの一例を示す。(a)では、メッセージを受信ミスしたサイト（Site1）が、他のサイトへメッセージを送付することで AMN 不一致を検出するケースを示している。(b)では同じく受信ミスした Site1 が他のサイトからのメッセージで AMN 不一致を検出し、それを他のサイトに知らせることで自身の受信ミスを検出するケースを示している。



(a) 受信ミスサイトからの送信による検出



(b) 受信ミスサイトでの受診による検出

図 2.6　フェイルストップ放送通信プロトコル

このプロトコルでは、集中管理部が存在せず、同期動作も不要である。

3.知的分散データベース IDPS-DB の基本設計と提案方式

3.1 自律分散データ管理の課題

IDPS は集中管理部を持たずに信頼性、拡張の高いシステムの実現が可能であり、要求や負荷が常に変化するシステムのプラットフォームとして開発された。IDPS のデータ管理機構である IDPS-DB 設計に際しては、この特徴を継承し、かつ汎用性の高いデータ管理機能の実現を目標とし、次の設計方針とした[10]-[12],[15],[16]：

・ 自律オブジェクトがデータを管理し、IDPS 上のアプリケーションオブジェクトからの要望に応じてオブジェクト間の協力・協調でデータ操作を実現する。

・ データモデルは、開発時主流となりつつあったリレーショナルモデルとした。但し、マルチメディアデータなどの普及状況を踏まえ、新たなデータモデルの導入を可能な形とした。

・ データ（＝オブジェクト）の多重化により、信頼性・可用性を確保する。

3.2 IDPS-DB のデータモデル

IDPS-DB では、基本データモデルとしてリレーショナルデータモデルを採用した。オブジェクトの管理単位はリレーションとした。当初、データ独立性やビューの変化に対する適応性などを満足させるため、いわゆる３層スキーマにおける各層の機能を有するオブジェクトを設けた。すなわち、リレーショナルビューを提供するデータオブジェクト、内部ビューを提供するファイル処理オブジェクト、外部ビューを提供する質問処理オブジェクトの３種類のオブジェクトである。

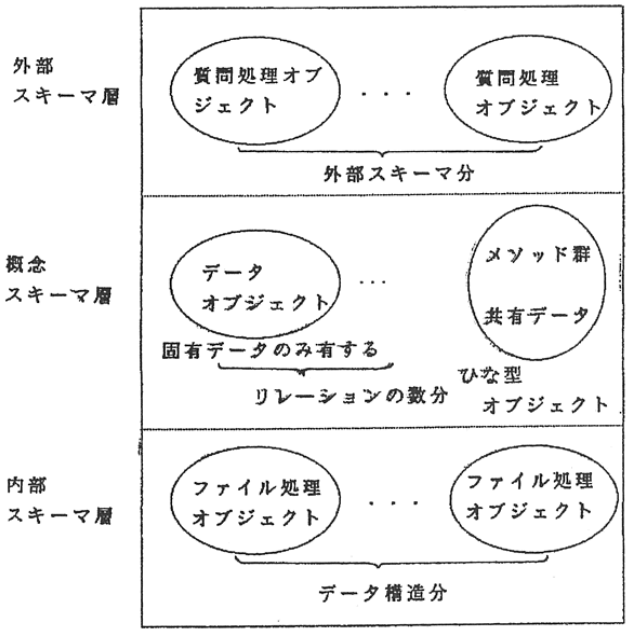


図 3.1 IDPS-DB オブジェクト構成

IDPS-DB における各オブジェクトを図 3.1 に、これらオブジェクトとその機能を実現するメソッド群の対応を表 3.1 に示す。

表 3.1 IDPS-DB における各オブジェクトのメソッド

オブジェクト	メソッド
質問処理オブジェクト	質問文処理メソッド 最適化メソッド 自動結合メソッド　など
データオブジェクト	関係操作メソッド　など
ファイル処理オブジェクト	データ検索メソッド データ整列メソッド　など

その後、効率性、管理の容易性などを考慮し、データオブジェクトにファイル処理オブジェクトの機能も統合し、さらに複数リレーションを１つのデータオブジェクトが管理する方式とした。

3.3 IDPS－DB におけるデータモデルと問い合わせ能力の設計

IDPS-DB は、リレーショナルモデル上の拡張問い合わせ機能を有する。具体的には、

－ドット表現を使った入れ子リレーショナルモデル

－不完全な SQL 記述から自動的に正確な SQL 文を作成する自動検索機能

の２つである。

前者は、リレーションの属性値としてのリレーション名の記述を許すもので、構造データの自然な表現を可能とする。後者は、質問文における条件属性（群）と、結果である目的属性（群）の間の結合パスの候補をオブジェクト間のメッセージ交換により探索する機能で、オブジェクト構成すなわちスキーマ構成が動的に変化し得る自律分散システムでは有用な機能である。詳細については**[10],[11]**を参照されたい。

3.4 システム構成の管理と関係演算の自律分散処理方式

図 3.2 に IDPS-DB のシステム構成例を示す。UNIX や RTUX 上に実現されたミドルウェアである知的分散システムをベースとし、以下に示す３種類のオブジェクトによりデータベースを構成している。

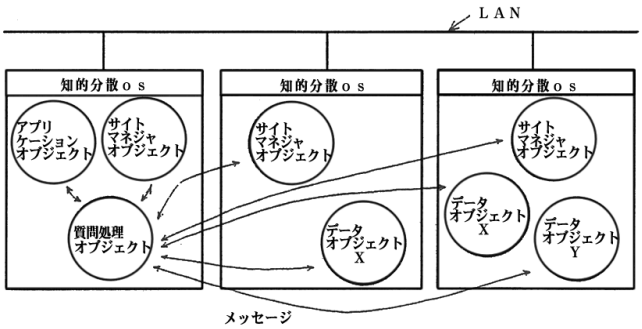


図 3.2 IDPS-DB のオブジェクト構成

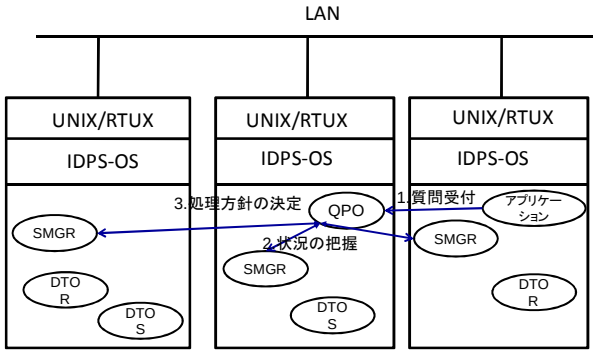
・サイトマネージャオブジェクト（SMGR）
システムを構成する各マシンに存在し、そのマシンに存在するリレーション名やオブジェクト数といったマシンに固有なデータベースのディクショナリ情報の一部を管理する。各サイトには IDPS-DB 以外の機能や役割を与えられえていることが通常であり、それらを含めたサイト情報管理機能として SMGR を設けた。

・質問処理オブジェクト（QPO）
アプリケーションからの問い合わせ処理を受け持つ。QPO は複数存在し、各 QPO は同時に複数のアプリケーションの処理を行うことはない。アプリケーションからの処理要求を受けつけるとその時点での各サイトのリレーション構成と負荷状況を同報メッセージで全サイトの SMGR に問い合わせる。その結果を受信後に最適な処理方法を決定し、関連するデータオブジェクト DTO へ処理を依頼する。結果はマージなどをおこないアプリケーションに返答する。アプリケーションは QPO とのやりとりだけを行なえば良い。

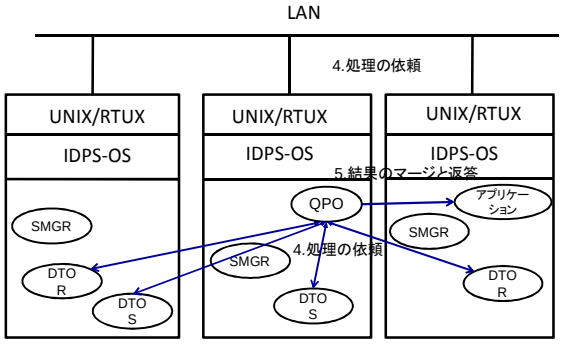
・データオブジェクト（DTO）
データベースで使用されるデータを管理するオブジェクトであり、リレーションのスキーマや実データおよびインデックス用のデータ等を管理する。信頼性や性能に対する要求に応じて複数のサイトにコピー（レプリカ）を配置することが可能である。同一オブジェクト名の DTO はレプリカとして認識される。

IDPS-DB における処理方式は、(a)質問を受け付けた際にシステム状況把握を行うステップと、(b)状況把握の結果にもとづき最適な処理方法を決定・実行するステップの２段階に分けることができる。 (a)の状況の把握では、QPO が SMGR に対する確認依頼を放送し、一定時間内に返答のあったサイトが処理対象となる（図 3.3 (a)）。(b)では QPO がコーディネータとなって(a)の結果を踏まえたレプリカの利用による高速処理をおこなう（図 3.3 (b)）。処理中のサイトダウンなどが発生した場合は、DTO からの返答に対するタイムアウトを QPO が検出し他のレプリカに振り替えることで処理を継続する。

システム状況の把握を都度行うには、処理効率が低下しないことが必要である。IDPS-DB の処理効率については次節で性能評価を、4 章で適用を通しての評価を述べる。



(a) 状況の確認



(b) 処理の実行

図 3.3 IDPS-DB の処理方式詳細

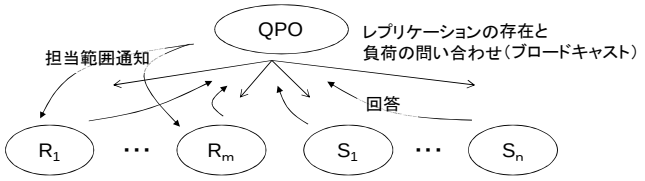
3.4 IDPS－DB における並列分散問い合わせ処理

ここでは、自律分散並列演算の例として、共通属性 x を有する２つのリレーション R と S の結合を考える。ここで、R は m 重化されており、S は n 重化されているとする。すなわち、R を管理する m 重化されたデータオブジェクトと S を管理する n 重化されたデータオブジェクトが存在する。リレーション R と S の間の並列結合は準結合(semi-join)を用いて次の様に実行する。

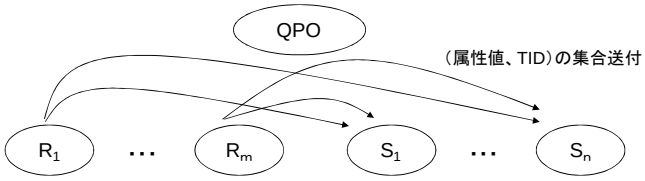
Step 1. ユーザからのリクエストにより、QPO が割り当てられる。割り当てられた QPO は、質問として結合属性 x による R と S の間の結合文を受け取る。これを受けると R と S を有するサイトへの負荷問い合わせをブロードキャストする。R と S、それぞれの多重度、結合にあたっての選択率を考慮して、R の値集合と TID の集合を S のすべてのレプリカに送るか、その逆にするかを決める。ここでは、R から S に送ると仮定

する。QPO は、R の各レプリカ Ri について、その負荷に応じて Ri が担当するべき結合属性 x の値の範囲を決めて、当該範囲の値を有するタプルの TID を S の n 個のレプリカに送付するよう指示する(R.x にインデックスがある場合である。R.x にインデックスがなければ、単純に R のタプル ID の範囲で分担する)。 指示は R の全 DTO へマルチキャストで送られる(図 3.4 (a))。同様に、S の各レプリカに、結合を実行する S のタプルの TID の担当範囲を決めて伝える。

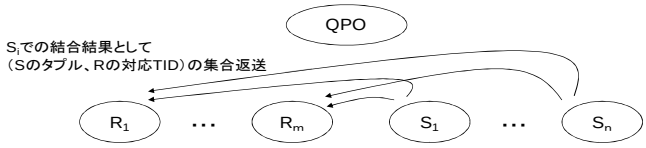
Step 2. R の各レプリカは QPO からのメッセージを受けると、属性 x が指定された範囲の値を有するタプルを読み出し、x の値と読み出したタプルの TID の対の集合を S の n 個のレプリカに送付する。すなわち、R1→S1…Sn、R2→S1…Sn、 ・・・、Rm→S1…Sn というメッセージが送付される(図 3.4 (b))。Ri からの各メッセージは放送通信であり、1 回のメッセージですべての S のレプリカに送付することができる。



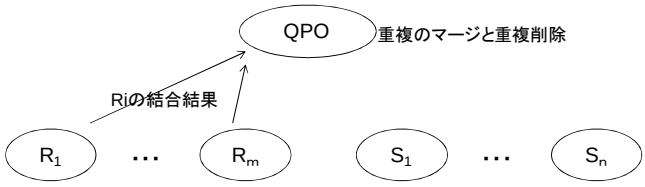
(a) レプリケーションの状況とレプリカサイトの負荷の把握と TID 範囲の割り当て



(b) R から S へ、x の値と TID の対の集合を送付



(c) 選択された S のタプルと R の TID の対の集合を R に返送



(d) R での結合実行と結果マージ

図 3.4 IDPS-DB における並列結合

Step 3. S の各レプリカは、QPO によってあらかじめ決められた TID の担当範囲内で、送られた x の値との比較をおこなう。マッチした S タプルと R の TID の対の集合を R に返答する（図 3.4 (c)）。

Step 4. R の各レプリカは結合対象の TID を受け取るので、そのデータを読み出し、送られてきた S のタプルとの結合をおこなう(図 3.4 (d))。

Step 5. リクエストを出した QPO オブジェクトは、R からの結果をすべて受け取った後に重複を削除してマージして最終結果を得る(図 3.4 (d))。

結合演算は、S の結合属性 x にインデックスがあれば index nested loop、なければ nested loop join をおこなう実装とした。

3.5 性能評価

3.4 で述べた並列結合を IDPS-DB に実装し、性能測定をおこなった結果を示す。評価は 1990 年のときのものである。各サイトはイーサネットで結合されている。各サイトはインテルの 16 ビットマイクロプロセッサー（80286）のワークステーション、ネットワークは 10Mbps のイーサネットであった。

対象リレーション R と S は、共に 1,000 タプルを有し、R は 12 バイト、S は 24 バイトのタプル長とした。結合属性 x は整数型で共に B+木型のインデックスを有する（多：多結合）。結合率は 10%としている。実験によると各サイトでの処理時間は、データの転送時間より多くかかっており、サイト数の増加が処理時間の短縮に効果があることが判明した。

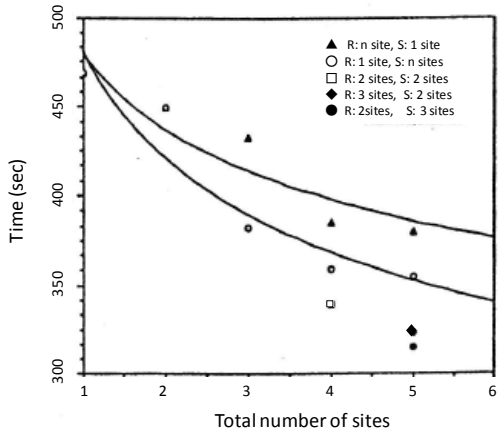


図 3.5 IDPS-DB における並列結合の性能

なお、選択演算などの 1 リレーションへの関係演算についても、当該リレーションがレプリカ m 個になって m 個の DTO に管理されていることを使って、DTO 間で演算実行する TID 範囲を分担して並列実行し、最終的に QPO で結果をマージするようになっている。

4. 高信頼化方式

IDPS-DB が想定した業務系システムへの適用では、トランザクション管理によるデータ一貫性の確保は必須である。

IDPS-DB では、２相ロック及び２相コミット規約で各トランザクションを実行する。すなわち、トランザクション要求に対応する QPO が２相ロック及び２相コミット規約のコーディネータとして動作し、DTO への必要なロックの獲得、問い合わせ・更新処理の実行指示、コミット、ロック解放を指揮する。各 DTO は、自分が管理するリレーションに相当するファイルをロック単位としている。したがって、ロック管理について集中管理部は存在しない。図 4.1 に概要を示す。

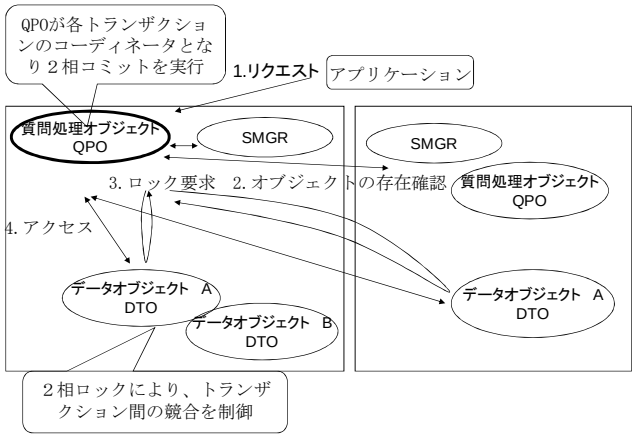


図 4.1　IDPS-DB におけるトランザクション管理

ロック失敗などの際のアボートに対応するため、ファイルシステムとしてシャドウファイルを採用し、迅速な処理を可能としている。

自律分散システムで必須となるサイト構成の運転時変化に対しても、矛盾のない対応を次のように実現している。すなわち、サイト追加の際は、処理スタート時にその都度、システム状況の確認をしているので、追加後の構成に応じた管理が自動的になされる。あるトランザクションを処理中にいずれかのサイトが削除された場合は、タイムアウトによる検出をおこない、アボートを行うことで対応している。

QPO については、IDPS 外部からの利用を考慮して、多重化しなかった。このため、トランザクション T の処理中に担当の QPO が障害を起こすと、T は中断となる。DTO はタイムアウト検査を行っており、コミット前なら矛盾の無い状態で自動アボートする。一方、コミット途中の QPO 障害では、一部の DTO がロック保持のまま判定待機になる不定状態になる可能性がある。IDPS-DB では、この場合の処理はオペレータ回復とし、シャドウファイルの機能を用いて全 DTO をアボートまたはコミットすることで整合性のある状態に復旧するよう設計した。

5. 適用による評価

IDPS-DB の大規模プラント開発時のコンカレントエンジニアリング支援システムへの適用についてその概要を述べる。大規模プラントでは、建築物や電気回路、配管などを並行して数百名規模の技術者で設計し、図面の規模は数千枚におよぶ。個々の技術者は、CAD などの設計支援システムを使って設計を行う。設計は、各種の法令や社内基準にもとづく仕様を満たしていなければならず、設計支援システムは設計結果がこれらを満たしているかどうかをチェックする機能を有している。また、各技術者による設計は他の設計と矛盾していてはならず、そのチェックも必要である。これらの要求に対応するため、設計支援システムのデータ管理システムに対しては次が求められる。

アクセス速度：設計の実施や検証はトライアル＆エラーの繰り返しであり、数百名規模の技術者がストレスなく利用できることが求められる。
拡張性：対象プラントの規模により大きく変わる設計規模への柔軟な対応が必要。
分散対応：設計作業は地理的に分散して多くの拠点で進められるため、これへの対応も必要となる。
信頼性：承認済みの図面、設計中の図面、技術的難易度の高いもの、そうでないものなどをそれぞれの重要度に応じた信頼度で管理できることが必要である。
データ構造の透過性：設計支援システムからはデータの位置や多重度を意識することなく、データにアクセスできることが求められる。

これらを解決する手段として IDPS-DB を用いたプロトタイプの構築と評価を試みた。SunOS 上のプロセスとして動く設計支援システムをそのまま利用することとし、データ管理と検索を IDPS-DB で行うシステムとして大規模プラント顧客にて運用した。IDPS-DB との I/F には RPC(Remote Procedure Call)を利用した。各設計支援システムは RPC を介して IDPS-DB の QPO を呼び出し、必要なデータの検索やロード、DB 書き換え（図面のセーブ）をおこなう。当該システムは Sun ワークステーション相当（東芝 AS シリーズ） 2 ノードから始めて適宜増設する構成であった。

１つの図面に関するデータは複数のリレーションに格納し、設計支援システムはこれらを検索(結合など)して図面を構成する。新たな設計が追加された際には、データオブジェクトの追加で対応可能である。

プロトタイプの構築により、基本動作の確認と前述の要件をほぼクリアできることを確認した。アクセス

速度については、当時の RDBMS 製品と同等以上の結果を得た。拡張性については、リレーションの追加をおこないつつ適宜サイトの追加をおこなうことで容易に対応可能であることを確認した。これにより、管理対象の図面が増えた場合に、RDBMS の停止・構成変更やアプリケーションの変更無しで対応可能である。信頼性はデータオブジェクトの多重化で、データ構造の透過性は（既に対応済みである）設計支援システムの RDBMS への対応機能により解決できた。分散対応については、データ管理システムとしては明らかに IDPS-DB は対応済みである

この事例に代表されるように、実際の運用時には、IDPS-DB には一貫性を保証したデータ管理能力と結合演算を介したコンテンツデータの再構成能力や検索能力が強く求められた。IDPS のオブジェクトや外部アプリケーションへ一貫したデータを供給することが主な役割であったからである。

質問受付時の動的システム状況の把握や関係演算の並列分散実行、トランザクション管理の自律分散化は、当初考えられた長期間運用時のシステム拡張でも有効であった。一方、IDPS-DB の実装では、オブジェクト管理やメッセージ配信の高信頼化など、IDPS-OS に依存した部分がある。特にメッセージ到着順序の同一性保証はレプリカ同時更新やコミット処理の簡素化と効率化の点で重要であった。

6. まとめ

本稿では、1990 年代に東芝で開発・商用化された知的分散システム IDPS のデータ管理機構に求められた課題と、その実現方法としての IDPS-DB について設計の概要と評価をまとめた。前章で述べたように、自律分散システムにおけるデータ管理機能方式と、適用を通した有効性の確認について述べた。

本稿で述べたレプリカを利用した並列分散結合は、現在の map/reduce アーキテクチャで通信負荷が高いときの結合方式[13],[14]に近い。また、自律分散システムが耐障害分散アプリケーションを動かすため、データベース側には一貫したデータ管理やスキーマ変化への対応能力が強く求められたことも留意される。IDPS の多重化プロセスで QPO を実現すれば耐障害処理はもっと見通しが良かったとは言える。IDPS-DB は同一 LAN 下のクラスタであり、IDPS の fail-stop 放送通信メッセージ機構のため、本質的にデータベースのネットワーク分割を想定せずに済んだ点も指摘しておきたい。

参 考 文 献

[1] A.S.タネンバウム他：“分散システム　原理とパラダイム”，ピアソンエデュケーション, 2003.10.

[2] 伊藤：“自律分散システム研究の課題と将来”、計測と制御、vol.32, No.10、pp.789-796, 1993.10.

[3] 森:“自律分散システムと制御分野での実用例”、計測と制御、vol.29, No.10, pp.923-928, 1990.10.

[4] 関、長谷川：“高信頼分散システム構築支援 OS：知的分散 OS”、情報処理学会誌、 vol.36, No.8., pp.764-768, 1995.8.

[5] 田村　他:”知的分散システムのアーキテクチャ”、電気学会誌 108-C[6]、1988.6.

[6] T.Seki et.al. : ”An Operating System for the Intellectual Distributed Processing System –An Object Oriented Approach Based on Broadcast Communication-“, Journal of Information Processing, vol.14, no.4. 1991.

[7] 関　他：“知的分散システムにおける高信頼放送通信機構”、電子情報通信学会論文誌 D-I, vol.J73-D-I, no.2, 1990.

[8] 関　他：“オブジェクト指向分散システムにおける放送待機冗長処理方式”、電気学会論文誌 D, vol.114, no.3, 1994.

[9] S.Harashima, et.al. :”A Fault-Tolerant Subway Passenger Information Control System –An Object Oriented Approach Based on Broadcast Communication-“, Proc. of 2nd ISADS, 1995.4.

[10] K.Nagase et. al: “IDPS Database System”, Proc. of the 6th International Joint Workshop on Computer Communication (JWCC-6), 1991.7.

[11] A.Howells et. al.: “ Partial Queries in a Dcentralised Distributed Relational Database.”, Proc.of Future Database, pp.97-105, World Scientific Publishers, 1992.4.

[12] S.Harashima et.al.: “Concurrency Control in IDPS Database System”, Proc. of the 6th International Joint Workshop on Computer Communication (JWCC-6), pp.167-172, 1991-7.

[13] DM. AL Hajj Hassan, M.Bamha: “Semi-join Computation on Distributed File Systems Using Map-Reduce-Merge Model”, ACM Proceedings of the 2010 ACM Symposium on Applied Computing, pp.406-413, 2010.3.

[14] S.Blanas,J.M.Patel,V.Ercegovac,J.Rao: ”A Comparison of Join Algorithms for Log Processing in MapReduce”, ACM Proceedings of the 2010 international conference on Management of data, pp.975-986, 2010.6.

[15] 原嶋：“知的分散システム IDPS におけるデータベース処理機構 IDPS-DB に関する研究”、電気通信大学大学院情報システム学研究科博士論文, 2011.9.

[16] 永瀬他：“知的分散データベースにおけるデータ管理方式”、情報処理学会データベースシステム研究会資料、pp.49-57、1993.9.