オペレーティング・システム 43 － 3
（１９８９．６．９）

ＯＬＴＰとフォールト・トレランス

渡辺榮一
タンデム・コンピューターズ

オンライン・トランザクション処理(OLTP)システムでは、システムの一部が故障しても、システムは機能し続けることが強く望まれる。ここでは、タンデム・コンピューターズ社(以下「タンデム」)のノンストップ・システムで実現されているフォールト・トレランスについて、ガーディアン・オペレーティング・システムに焦点を絞って述べるとともに、データベース、トランザクション管理、ネットワークなどいくつかのソフトウェアについても触れる。

OLTP and Fault Tolerance

Eiichi Watanabe
Tandem Computers Inc.

In on-line transaction processing (OLTP), it is highly desirable for the system continues to be available in the event of a component failure. In this paper, fault tolerance as implemented in the Tandem NonStop System is discussed. The discussion focuses on Guardian operating system, and touches database, transaction management, network and some other topics also.

1. まえがき

典型的なオンライン・トランザクション処理システムは、(1)人間が操作する多数の端末装置、(2)ディスク上のデータベース、および、(3)処理手続きを実行するプロセッサー(処理装置)で構成される。これらの構成要素は、動作速度の点で大きく異なる。人間が端末を操作する速度は秒、ソフトウェアがデータベースをアクセスする速度はミリ秒、そしてプロセッサーがプログラムを実行する速度はマイクロ秒のオーダーである。この動作速度の違いが、1台のコンピューターで多数のトランザクションを並行的に処理することを可能にしている。しかし、並行処理は、システムの障害対策を複雑にする。そこで、「並行性制御」(または「同時実行制御」)と「回復」(リカバリー)が、トランザクション処理に関するソフトウェア技術の中心的な課題となっている。これらのテーマは、オペレーティング・システムとデータベース管理システムという二つの分野に関係する。さらに、トランザクションが分散システムの環境で処理される場合には、システム・ネットワークおよび分散データベースの技術も関係してくる。したがって、OLTPシステムのためのフォールト・トレランスの実現には、ハードウェア、オペレーティング・システム、データベース、ネットワーク、そしてプログラミング言語にまたがる総合的な対策が要求される。

2. ガーディアン・オペレーティング・システム

ガーディアンの設計目標

ガーディアン(GUARDIAN)は、仮想記憶の管理、優先度に基づくディスパッチなど通常のオペレーティング・システムが有する機能に加えて、ハードウェアが故障した場合の再構成の機能など、フォールト・トレランスの実現を目的として設計された。タンデムのノンストップ・システムにおけるフォールト・トレランスの基本的な考え方は、次の通りである。

(1) システムは、モジュールの単一故障に耐えて作動し続けること。
(2) システムが作動中に、故障したモジュールの修理を行い、システムに再統合できること。

プロセスとメッセージ

ガーディアンの設計者は、ヨーロッパの計算機科学者たちの考え方(ダイクストラの "THE" システム[文献1]およびブリンチ・ハンセンの単一プロセッサー・システムのためのオペレーティング・システム・ニュークリアス[文献2])の影響を受けたといわれる。それは、「プロセス」と「メッセージ」という概念の採用である。その結果、ガーディアンは、「メッセージの受渡し」(message passing)に

基づくオペレーティング・システムとして実現された。このことは、以下に述べるように、フォールト・トレランスと表裏一体の関係にある。

システムは多数のプロセッサーで構成される。それと同じように、システムは多数のプロセスで構成される。プロセッサーはハードウェアであり、プロセスはソフトウェアであるという違いはあるが、概念的には同じような位置付けにある。すなわち、プロセスは、ソフトウェアのモジュール化の単位であり、フォールト・トレランスのための冗長構成の基本である。

一つのプロセッサーには複数のプロセスが存在し、プロセス毎に一つのアドレス空間が存在する。ガーディアンの基本部である「カーネル」が、この環境を制御する。一台のプロセッサーの内部においては、複数のプロセスがメモリーを共有できる。メモリーの共有は、プログラムのコード部分(「コード・セグメント」)およびデータ部分(「データ・セグメント」)のいずれにおいても可能であるが、通常は、読み出し専用のコード・セグメントに適用され、データ・セグメントには適用されない。それは、フォールトの封じ込め(fault containment)や、プロセスを他のプロセッサーに再配置したり、プロセスのコピーを増やす上で有利だからである。この考えに基づき、プロセス間の通信は、共有メモリーを介して行なうのではなく、プロセス間におけるメッセージの受渡しの形で行なわれる。

プロセス間のメッセージの伝送は、カーネルによって制御される。送信側プロセスおよび受信側プロセスは、同一のプロセッサー内にあっても、異なるプロセッサー内にあってもよい。メッセージ伝送の物理的な経路には、(1)メモリーからメモリーへのコピー、(2)ローカル・ネットワークを介しての伝送、(3)長距離の通信回線を通しての伝送がある。しかし、送信側プロセスおよび受信側プロセスは、いずれであるか関知しない。

フォールト・マスキング

カーネルはまた、ある種のハードウェアの故障を隠ぺいする。例えば、読み出し専用の仮想記憶に訂正不能の誤りが起きた場合に、カーネルは、ディスクから再び読み出すことによって、その箇所を正常な状態に戻すことができる。また、停電が起きた場合に、カーネルはプロセッサーの状態を記録し、システムを停止させる。電気の供給が回復すると、カーネルは、プロセッサーの状態を復元し、システムは、中断していた処理を続行する。

ヘルス・チェック

カーネルは、プロセッサーが正常に作動していることを示すメッセージを定期的に発信する。それは、「生存通知」("I'm alive"メッセージ)と呼ばれる。あ

るプロセッサーが停止したことは、生存通知が途絶えたことによって分かる。生きているプロセッサーは、「グループ再編成アルゴリズム」によって、どのプロセッサーが作動しているか、いないかを判断する。

プロセスの同期と通信

プロセスの同期のために、セマフォーとイベント・フラグが使われる。セマフォー操作には、基本命令として"PSEM"と"VSEM"がある。それらは、ダイクストラのP操作およびV操作に対応している。セマフォーは、同一プロセッサー内にあるプロセス相互間の通信に使われる。典型的な使用例としては、常駐メモリー・バッファー、メッセージ・コントロール・ブロック、入出力制御装置などの資源へのアクセスの制御がある。イベント・フラグは、低いレベルの動作、例えば、装置の割り込み、プロセッサーの電源投入、メッセージの完了、メッセージの到着などが起こった時にセットされる。プロセスは、そのようなイベントの発生を待つために、基本命令"WAIT"を使う。イベントの発生は、基本命令"AWAKE"によって知らされる。

プロセスがイベントの発生またはセマフォーの割当を待って自分自身を封鎖する場合に、最大の封鎖時間を設定することができる。この制限時間に達した場合には、プロセスは、誤り条件コードを伴って起動される。この機構は、障害を検出するための「ウォッチドッグ・タイマー」として使われる。

プロセス対

単一故障に耐えるという設計基準によれば、プロセスが作動できなくなった場合、例えば、ソフトウェアの(タイミングに依存した)バグやプロセッサーの故障によってプロセスに障害が起こった場合の対策が必要である。この問題に対する一つのアプローチが、プロセスを二重化する技法である。それは、「プロセス対」(process pair)と呼ばれている。プロセス対は、「主」(primary)プロセスと、他のプロセッサーにある「予備」(backup)プロセスの一組で構成される。予備プロセスは、主プロセスと同じプログラム、論理アドレス空間、セッションを有する。

主プロセスが作動している間、予備プロセスは、受け身の状態にある。主プロセスは、重要な箇所において、予備プロセスが処理を引き継ぐために必要な情報を送る。この操作は「チェックポインティング」、それによって送られる情報は「チェックポイント・メッセージ」と呼ばれる。チェックポイント・メッセージには、いくつかの考え方がある。すなわち、(1)予備プロセスが自分のアドレス空間へコピーする「新たな状態イメージ」(new state image)、(2)自分の状態を更新するための「変分」(delta)、あるいは(3)「機能」(function)である。タンデムのシステム・ソフトウェアが採用している技法は、「状態イメージ」から「変

分」または「機能」の方式へと次第に移行してきた。その理由は、伝送すべき情報量が少なくてすむこと、そして主プロセスの状態の誤りが、予備プロセスの状態へ波及する可能性が少ないことにある。

プロセス対は、単一故障に耐えるプログラム実行の仕組みを与える。単一故障には、ハードウェアのフォールトとともに、ソフトウェアの過渡的なフォールトも含まれる。稼働中のオンライン・システムのソフトウェアのフォールトの大部分は過渡的であるといわれる。

3. データベース

オンライン・トランザクション処理では、障害が起きてもデータベースの正しさを保つことが不可欠である。ここでも、単一故障に耐えるという設計基準が適用される。また、「トランザクション」の概念に基づいて、並行性制御と回復が実現されている。

ファイルの二重化

タンデムのデータ・マネジメント・システムにおいては、ファイルを複数の「パーティション」に分割し、同一システム上の異なるディスクまたは異なるシステム上にある任意のディスクに分散して配置することが可能である。そして、各パーティションは、2台のディスクに同じ内容を持つことができる。これを「ミラード・ディスク」(mirrored disc)と呼ぶ。

トランザクション管理

一つの業務処理に含まれるすべての「仕事」は、「トランザクション・モニタリング・ファシリティー」(TMF)と呼ばれるソフトウェアによって、「原子的な単位」(atomic unit)にまとめることができる。「原子」とは、それ以上細分化することのできない最小の単位を意味する。この単位業務処理を、「トランザクション」と称し、「transid」と呼ばれる識別子によって区別される。すなわち、このトランザクションに対して為されたすべての仕事は、transidをタグとして付けることによって識別可能である。資源を占有するための「ロック」も、transidによって特定のトランザクションと関連付られる。「取り消し」(undo)および「やり直し」(redo)のための「監査トレール」(audit trail、いわゆる「ログ」)にも、transidが付けられる。トランザクション処理が正常に終了した(コミットした)時には、それによって為された処理の効果は永久的となる。しかし、異常に終了した(アボートした)場合には、それによって為された処理の効果は、すべて取り消される。このことは、分散システムの環境にも当てはまる。

TMFを使うことによって、アプリケーション・プログラムは、プロセス対の構成にする必要がないので、プログラミングは容易である。ただし、プロセス対の技法は、システム・ソフトウェア(TMF自体、カーネル、ドライバーなど)で使われている。

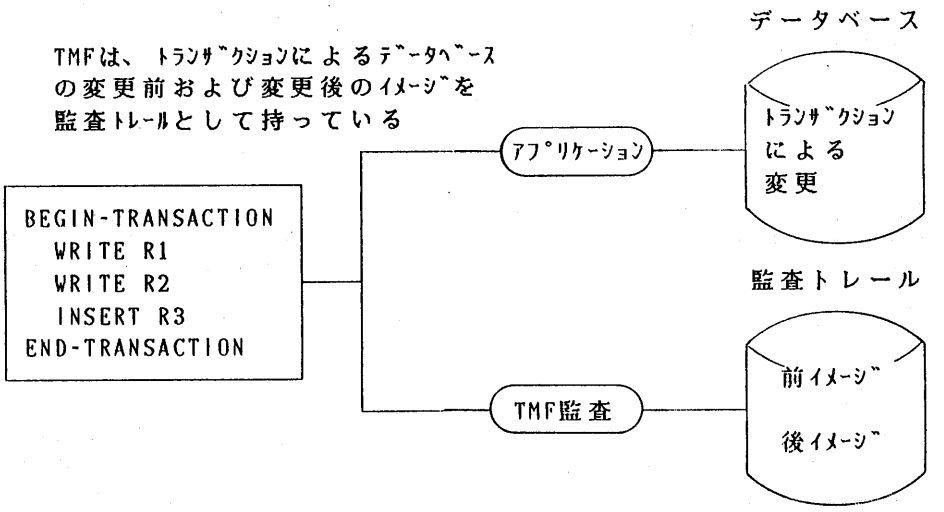

図1. トランザクション・モニタリング・ファシリティー(TMF)

遠隔地間のデータベースの二重化

1箇所のコンピューター・センターの運用が不可能な事態に対処することも、システム・レベルでのフォールト・トレランスの課題である。通常採用されている方法は、互いに離れた場所にある2箇所のコンピューター・センターの間で、オンライン・データベースを二重化することである。そうすることによって、万一、一方のセンターが機能できなくなった場合でも、トランザクションまたはオンライン・データベースを殆どまたは全く失なうことなしに、直ちにもう一方のセンターが取って替わることができる。タンデムのソフトウェア製品である「リモート・デュープリケット・データベース・ファシリティー」(RDF)は、この目的を満たすものである。

RDFは、TMFの監査トレールから必要な情報を抽出して、エクスパンド(TM)ネットワーク(次の節を参照)を通して他のノードに送る。

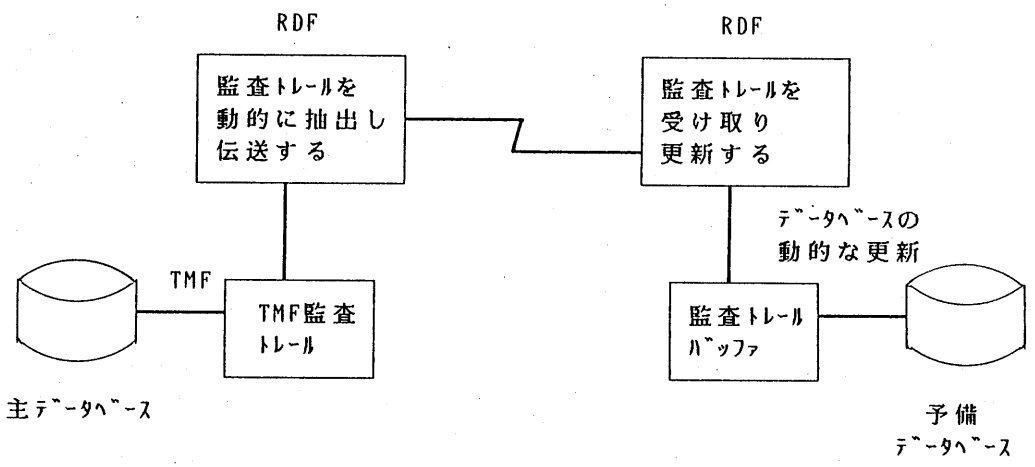

図２．　リモート・デュープリケット・データベース・ファシリティー(RDF)

４．　ネットワーク

プロセスとメッセージの概念に基づいて作られたガーディアンは、容易にネットワーク・オペレーティング・システムへと一般化することが可能である。タンデム固有のネットワーク・アーキテクチャーである「エクスパンド」(EXPAND)は、当初の設計における16プロセッサーの構成を、最大4080台のプロセッサーから成るネットワークへ拡張するものである。サポートされるネットワーク・ソフトウェアの種類には、SNA、OSI、MAP、SWIFTなどの標準的なネットワークも含まれている。

オペレーティング・システムにおけるフォールト・トレランスは、ネットワークにも拡張されている。ネットワーク・ソフトウェアは、システムの一部に障害が起きても、セッションを継続することができる。例えば、SNAをサポートするタンデムのソフトウェア製品SNAXにおいては、要所要所で重要な情報をチェックポイントとして予備プロセスに渡しておくことによって、継続的な動作を行うことが可能である。したがって、もし1台のプロセッサーまたは回線が故障しても、進行中のすべてのセッションは継続する。これと同じことが、開放型システム相互接続(OSI)のソフトウェアについても言える。

タンデムには、多数のノンストップ・システムとエクスパンド・ネットワークを管理するための、「ディストリビューテッド・システム・マネジメント」(DSM)と呼ばれるソフトウェア製品があるが、DSMにもフォールト・トレランスが備わっている。例えば、DSMの構成要素である「イベント・マネジメント・サブシステム」(EMS)は、プロセス対として設計されている。EMSにおいては、主プロセスに障害が起きた場合に、予備プロセスがイベント・ログ・ファイルの管理を引き継ぐ。

5. ハイレベルのプログラミング・ツール

プログラマーが犯すミスは、コーディングの行数に比例すると言われている。したがって、高い品質のソフトウェアを開発するためには、できるだけ少ない行数のコーディングで済ませられることが望ましい。タンデムのデータベース言語「ノッストップSQL」は、その一例である。ノンストップSQLは、高い性能を必要とするOLTPアプリケーションのために開発され、分散データベースをもサポートしている。

6. あとがき

ここでは、ソフトウェアの面から、フォールト・トレランスについて述べた。なお、[文献4]では、ハードウェアを中心としたフォールト・トレランスが紹介されている。

謝辞

本稿の内容の多くは[文献3]に基づいている。原著者に感謝する。

参考文献

(1) Dijkstra, E.W. "The Structure of the 'THE' Multiprogramming System," Comm. ACM, 11 (1968), pp 341-346.

(2) Brinch Hansen, P. "The Nucleus of a Multi-programming System," Comm. ACM, 13 (April 1970), pp 238-241.

(3) J. Bartlet, D. Garcia, J. Gray, R. Horst, D. Lenonski, D. McGuire, K. Worsencroft, "Fault Tolerance in Tandem Computer Systems," (to be published)

(4) 渡辺榮一、「タンデムノンストップシステムにおけるフォールトトレランス技法の発展」信学技法 FTS 88-37、1989年2月23日