Vol. 36 No. 8　　情　報　処　理　　Aug. 1995

解　説

日本におけるオペレーティングシステム研究の動向

# 4.1 フォールトトレラントコンピュータを支えるOS技術と最近の動向†

中　村　智　明††

## 1. はじめに

フォールトトレラントコンピュータ（FTC）は1980年代に米国で生まれ，金融，流通，宇宙，防衛などの基幹産業を支える計算機として発展してきたが，1990年代に入ると国産メーカも参入し，24時間無停止稼働が要求される分野で広く使われるようになった[1)~5)]．

FTCに搭載されるOSに要求される機能は大きく，フォールトトレランスを実現するための機能と，システムの段階的成長を支える機能に分けられる．前者には

- ハードウェア構成要素の一点障害の救済
- OSそのものの残存障害の救済
- アプリケーションプログラム障害の救済

が含まれ，後者には

- プロセッサ，メモリ，ディスク，回線，ネットワークなどハードウェア機器の増設やアップグレード
- OSを含むソフトウェアの保守やバージョンアップ

が含まれる．特にシステムの変更や拡張といった保守コストがシステムライフサイクルコストに占める割合は60%を越えると言われており[6)]，後者の特質がFTC導入の決め手になるケースもあり重要性を増している．

これらの要求機能のすべてを業務を停止させることなく実現するシステムはいまだないが，その実現に向けて研究が進められている．

本稿では，これらの要求を実現するための技術概要と課題について述べる．

## 2. フォールトトレランスの実現技術

### 2.1 ハードウェア障害の救済

冗長化ハードウェアを利用してハードウェアの一点障害を救済するためには，障害検出，障害切離し障害回復，障害機器交換，交換機器の再統合の手順を踏む必要がある．障害の回復処理以降はOS主導で行われるが，障害検出と切離しについては，ハードウェアもしくはOSで行う方式が考案されており，それぞれ一長一短がある．特にプロセッサとメモリに関して，これらの機能をハードウェア，OSいずれで実現するかは，FTCの実装において基本的な課題でありこれまでに種々の方式が考案されている[3)]．

（1） クロック同期方式（ハードウェア主体）

プロセッサボードレベルで冗長化ハードが同期動作しており，障害検出，障害切離しまでをハードウェアで行う．プロセッサ/メモリ系障害はOSには割込みで報告されるため，障害ボードで実行中のOS自身が以後の処理を行う．OSへのインパクトが最も少ないため，UNIX☆などの標準OSをベースとするシステムに向いている．

（2） キャッシュフラッシュ同期方式（OS主体）

キャッシュメモリから主メモリへのライトバックをトランザクションと見なしてOSが制御する．カーネルOS内に本トランザクションのリカバリブロック機構を実装する．プロセッサボード間の多重バックアップが可能であるという利点を持つ．

（3） 外部割込み同期方式（OS主体）

プロセッサ間の密な同期によるハードウェアの複雑さを回避するため，フリーランのプロセッサ

† Operating System Technologies for Recent Fault Tolerant Computers by Tomoaki NAKAMURA (Computer Control Systems Development Department, Omika Works, Hitachi Ltd.).

†† (株)日立製作所大みか工場計算制御システム開発部

☆UNIXは，X/Open Company Limitedがライセンスしている米国ならびに他の国における登録商標です．

がI/Oアクセスを行う場合にのみI/Oチャネルハードウェアが多重プロセッサ間の同期をとる．

（4） メッセージ同期方式（OS主体）

ソフトウェアで同期をとる方式であり，さらに，OSレベルでメッセージ転送する方式，および，ミドルソフトが，チェックポイント時点で処理結果をメッセージ転送しメモリ内容を一致化させる方式がある．リカバリブロック構造となるため，ソフトウェアのタイミングに依存する障害に効果があると言われている．

いずれの方式をとるにせよ，OSにはハードウェア機器のオンライン交換を実現するための基本技術である，ハードウェア機器の構成認識，過渡障害と固定障害の切分けと障害部位の特定，機器の接続状態制御，障害メッセージの表示，交換後のディスクやメモリの内容一致化などの機能を持つことが求められる．

### 2.2 OS障害の救済

基本的にはOSは設計上流工程での品質作り込みとテスト段階での種々のテスト技法により，その品質は高いが，それでもバグなどによる障害が皆無である保証はない．OS自身の障害に対しては，アサーションポイントへのフォワードリカバリ機構の組込み[10]，およびOSの障害は大半がタイミングに依存する障害であることを前提としたロールバックリカバリ[5]のアプローチが行われている．

### 2.3 アプリケーションプログラム障害の救済

オンライントランザクションシステムなど，データの一貫性を保持することが絶対条件であるようなシステムにおいては，アプリケーションプログラムの障害は，データベースソフトまたはトランザクションモニタより自動ロールバックされる[2]．また，リアルタイム制御分野ではOSがタスクの強制ロールバック機構を備えているものもある[3]．このようなロールバック機構は回復に時間がかかるという欠点はあるものの，タイミングに依存する障害およびデッドロックからの回復に効果がある．

## 3. システムの段階的拡張を支える機能

### 3.1 ハードウェア機器の拡張

昨今のダウンサイジング化の波およびクライアントサーバシステムの台頭は，ハードウェア機器の増設のニーズをもたらした．特に24時間無停止が要求される場合には，機器増設を無停止で行いたいという要求が生まれるのは自然である．

一方，FT機構を持ったハードウェアであれば機器の増設は，スロットが空いている限りハード的には活線で行うことができるため，大変都合がよい．しかし，これをソフトウェアの観点で見ると，自明ではない．ここでは代表的な機器についてOS実装上の課題について述べる．

（1） プロセッサ

マルチプロセッサのサポートは現在ではかなり成熟したOS技術ではあるが活線でのプロセッサの増設時には，プロセッサボードのハードレベルの自己診断の後に，プロセッサの構成を認識し，スケジューラキューへのプロセッサの追加，プロセッサ間IPC機構のオープンなどのOS管理テーブルへの組込みを行う必要がある．また，プロセススケジューリングの際にはプロセッサアフィニティ（特定のタスク/スレッドを特定のプロセッサ/プロセッサ群上で実行させるようなスケジューリング方式および属性）に留意する必要がある．

（2） 主メモリ

主メモリの場合は，通常ハードレベルで行うパリティ作り込みのためのメモリ初期化や，仮想メモリのマッピングの確立と，空きページフレーム管理テーブルへの組込みなど，OS実行中に自身の管理テーブルを更新する必要があり細心の注意を要する．さらに，増設メモリをどのような用途（プロセス用，バッファプール，共用メモリなど）に用いるかによって，通常はシステム立ち上げ時にしか行わない，これらの管理テーブルのダイナミックな再構成機能を実装する必要がある．

（3） ディスク

ディスクの構成定義（パーティション，ミラー定義など）をオンラインで行えるようにする必要がある．これらの定義が完了するまでは，アプリケーションプログラムが誤アクセスしないためのインタロックも必要になる．

これら増設にともなうソフトウェア上の再構成は，必ずしも，すべてが実現されておらずメモリの増設など，多分に研究の余地がある．なお，拡張と同様に機器の削除のニーズも希にはあるが，これは機器を使用中のアプリケーションの処理の

連続性をどう保つかという問題が新たに発生し，これも実用レベルにあるとは言えない．

### 3.2 ソフトウェアの保守・拡張

ソフトウェアが扱うデータ構造が一新するような保守/拡張は，稼働中のソフトウェアがそのデータを扱うことができないため，別システムを平列動作させて行うのが現実的である．

一般的には，ソフトウェアの保守・拡張のニーズは既存のデータベースを用いて，業務を拡大したいというものであり，業務プログラムや，そのプログラムがコールする共用ライブラリをオンライン稼働中にレビジョン/バージョンアップしたいという形態になる．

ソフトウェアの保守・拡張はメッセージ同期方式において早くから実装例がある[2),5)]．

また，クロック同期方式においても，ダイナミックリンク機構をより発展させ共用ライブラリの1個のアーカイブ内のモジュール単独入替えを実現した例や，クライアントサーバ構造に着目してイベント待ちのタイミングで旧タスクを停止させることなく新タスクにスワップさせる方式も提案されている[11),3)]．

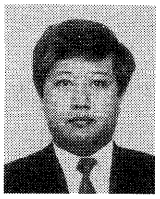
## 4. おわりに

単一計算機上でのフォールトトレラントOSの研究は，ハードウェアの一点障害の救済および機器の拡張性を軸に進められてきたが，システム的にはまだ十分ではない．この問題にシステム的にアプローチしたものとしては自律分散システムに基づく研究があり，現在も進められている[7)~9)]．

## 参考文献


1) Siewiorek, D. P.: Fault Tolerance in Commercial Computers, IEEE Computer, pp. 26-37, Vol. 23, No. 7 (1990).
2) グレイ他著　渡辺榮一編：フォールト・トレラント・システム，マグロウヒル (1986).
3) 内藤祥雄監修，渡辺榮一編：高信頼UNIXシステム，pp.14-46，日立FT—6100シリーズ，マグロウヒル (1994).
4) 金川信康，山口伸一朗，福丸広昭，宮尾健：フォールトトレラントサーバFT 6100の高信頼化技術，信学技報，FTS 93-56，pp.49-54 (1993).
5) Muramatsu, H., Date, M. et al.: Operating System SXO for Continuous Operation, IFIP92, pp. 615-621 (1992).
6) Laprie, J. C. et al.: Definition and Analysis of Hardware-and Software-Fault-Tolerant Architectures, IEEE Commputer, pp. 39-51, Vol. 23, No. 7 (1990).
7) 井原広一：フォールトトレラントコンピュータのシステム構成技術の展望，信学誌，Vol. 73, No. 11, pp. 1136 (1990).
8) Mori, K. et al.: Autonomous Decentralized Software Structure and Its Application, Proc. Fall Joint Comp. Conf., pp. 1056-1063 (Nov. 1986).
9) Kim, K. H., Yoshizawa, R., Bacellar, L. F., Masui, K. and Mori, K.: A Modular Implementation Model for DRB Computing Stations and a Supervisor Station in LAN-based Systems, Pacific Rim Intl. Symp. on Fault Tolerant Computing, pp. 56-60 (1993).
10) 内藤祥雄監修，渡辺榮一編：高信頼UNIXシステム，pp.66，Tandem Integrity Sシリーズ (1994).
11) Kobayashi, H., Nakanishi, H., Hayashi, K., Sato, H., Wataya, H., Nishiyama, S., Nakahashi, T. and Saika, T.: Online Software Expansion in Large-Scale Widely Distributed Systems, ISADS93, pp. 300-303 (1993).


(平成6年6月30日受付)



**中村　智明**（正会員）

1955年生．1977年東京大学工学部原子力工学科卒業．1979年同修士課程修了．同年(株)日立製作所に入社．以来，大みか工場計算制御システム設計部（現在同開発部）にて産業用計算機システムのオペレーティングシステム開発に従事．1985年米国スタンフォード大学コンピュータサイエンス学科修士課程修了．以後，同社リアルタイムUNIXシステム，フォールトトレラントリアルタイムUNIXシステムの開発リーダとして製品開発を行う．1995年より同社産業用マイクロカーネルの開発に従事．現在同部主任技師．リアルタイム，高信頼オペレーティングシステム，マイクロカーネルに関心を持つ．共著「高信頼UNIXシステム」（マグロウヒル，1994）．IEEE，ACM，USENIX各会員．