ホットスタンバイ方式による UNIX システムの高信頼化

島田一洋† 杉岡一郎††

UNIX をベースにした二重化コンピュータシステムの高信頼化技術について述べる．筆者らは，ホットスタンバイシステムを実現するための基本機能である CPU の二重化制御機能とディスクのミラー制御機能，クロスコール制御機能を標準的な UNIX を機能強化して実現した．本論文では，本システムの構成と特徴的な機能である相互監視方式の目的とその効果について述べる．更に本システムの適用事例についても紹介する．

High Reliability UNIX System Realized by Hot Standby Method

KAZUHIRO SHIMADA† and ICHIRO SUGIOKA††

This paper describes a high reliability technology of the duplicated CPU computer system realized by means of the hot-standby architecture and the functionally extended UNIX. The authors wish to report on some distinctive control functions for the CPU duplication, the mirror and cross-call disk drive control functions as extension and enhancement of some available and fundamental functions in the standard UNIX to realize the hot standby system.

1. はじめに

今日の社会は，直接あるいは間接的にコンピュータに大きく依存している．そのコンピュータ依存システムは，日増しに大規模化，広域化され続けており，システムを支えているコンピュータの信頼性を高めることがきわめて重要な問題である．例えば，金融関連システム（キャッシュカードによる現金引き落とし・照会システム，株式売買システムなど），列車や航空機の座席予約・発券システムなどが停止すると大きな社会問題になる可能性がある．

そこで，そのようなコンピュータ依存システムを構成するコンピュータの故障発生によるシステムの稼働停止を防止し，稼働性を大幅に向上させるために，コンピュータシステムの構成要素を冗長化する方式を採用した FTC (Fault Tolerant Computer) が提案され，製品化もされている．

FTC においては，従来 OLTP 専用の独自 OS が採用されていたが，最近では国際的な標準 OS として急速に普及してきている UNIX が採用され始めてきている．

† 株式会社 PFU 東京開発センター第一開発部
PFU Limited. Tokyo R&D Center, Development DIV. 1
†† 室蘭工業大学地域共同研究開発センター
Center for Cooperative Research and Development, Muroran Institute of Technology

UNIX をオンライン・トランザクション処理，産業制御処理あるいは通信制御などの分野で利用するコンピュータの OS として用いる場合には，リアルタイム処理能力の欠如，大容量ファイルにおける入出力性能の低下やファイルシステムの脆弱性などといった，高信頼化を実現する際の問題がある．

筆者らは，産業制御処理や通信制御の分野などで使用することを目的とした高信頼化システム（以下，本システムと記す）を開発した．

本システムは，UNIX をベースにした二重化コンピュータシステムであり，高信頼化のために UNIX に特徴的な機能を追加し，システムには独創的な技術を用いている．

まず，OS として UNIX を採用するにあたっては，UNIX を如何にして高信頼化するかが重要な課題となる．そこで，高信頼性システムに要求される高速応答性，データの保全性向上，障害回復処理速度の向上など，標準的な UNIX システムにはない機能の強化を行った[1]．次に機能強化を施した UNIX 上で待機冗長（デュプレックス型ホットスタンバイ）方式を実現し，システムの信頼性をより一層向上させている．

2. システムの高信頼化

2.1 高信頼システムの実現方式

高信頼システムを実現するには以下のような方式が

あり，それぞれ一長一短がある[2]．

（1） ハード FT 方式

（2） システム FT 方式

さらに，システム FT 方式には，

（a） デュプレックス型ホットスタンバイ

（b） ロードシェア型ホットスタンバイ

（c） マイクロカーネル型ホットスタンバイ

などがある．

これらの各方式を UNIX を OS に適用して実現するという観点から簡単に考察する．

（1） ハード FT 方式

ハード故障発生がハード内部で隠蔽されるため，OS 内部に FT の仕組みを埋め込む必要がなく UNIX の比較的単純な移植で実現可能であるが，それでも OS 内部に故障箇所の切り離しや活性保守のための機能を実現する必要がある．

また，ハード FT 方式は，常時複数 CPU の同期を取りながら比較検証する方式なので，ハードコスト，性能ペナルティが大きく，高速 CPU への移行には技術的困難が多い．

（2） マイクロカーネル方式

OS の基本部に FT のための方式的な手当てが必要で，UNIX など流通 OS で対応するには多くの改造量が伴う．このため，将来の UNIX のリリースアップに追随するのは容易ではない．

（3） デュプレックス型およびロードシェア型ホットスタンバイ方式

汎用モデルのコンピュータに，共用ハードウェアと相互監視機構を組み合わせることで実現できるので，UNIX など流通 OS に比較的小量の改造を加えることで実現することができる．

高信頼システムを UNIX で実現する場合の OS の変更量は，少ない順に，

（1） デュプレックス型ホットスタンバイ方式

（2） ロードシェア型ホットスタンバイ方式

（3） ハード FT 方式

（4） マイクロカーネル方式

の順になると考えられる．

2.2 システムの信頼性を阻害する要因

システムダウンの原因を要因別に分析すると，ハードウェア故障が約60%，ソフトウェア故障によるものが約30%，その他，オペレータの操作ミスなどによるものが約10%という報告がある[3]．

しかし，近年，ハードウェアの信頼性が向上している一方で，ソフトウェアの大規模化，複雑化に伴って，バグによる障害の割合がハードウェアに起因する障害よりも相対的に増加しているという事実がある[4]ので，ソフトウェア障害に対する耐故障対策が重要な課題となってきている．

そこで，筆者らは，以下の5点を重視してデュプレックス型ホットスタンバイ方式によってシステムの信頼性を向上させることにした．

（1） OS 故障，アプリケーション故障，運用ミスに対して柔軟な対応が可能

（2） UNIX の採用

（3） 将来の高速 CPU への追随性

（4） コストパフォーマンス

（5） ハードウェア故障に対する耐故障性の実現

3. 高信頼化 UNIX によるホットスタンバイ方式

本システムは，主要な構成要素を二重化し待機冗長構成とする高信頼システムである．

3.1 システム構成

本システムの構成を**図1**に示す．

（1） 処理装置系の二重化

処理装置系とは，CPU，メモリ，システムディスク等のシステム基本部のことである．

2台の処理装置系の一方を“運用系”，他方を“待機系”とする．そしてユーザの主たる業務を実行する処理装置を運用系と位置付ける．待機系は，運用系の障害に備えて待機している処理装置である．

相互の監視方式，運用系の切替え方式については後述する．

（2） ディスク・データの二重化

デュアルファイル方式によりデータの二重化を行っている．

二重化の単位としてはボリューム単位であり，ボリュームを制御するアダプタや電源を含めて二重化している．このため，片系障害時の運用継続や運用中の保守交換が可能である．

更に，二重化された処理装置系から各々別系統のディスクへのアクセスパスがある．運用系の切替え時は，自動的にアクセスパスの切替えが行われ，入出力が継続できる．

（3） 対ホスト通信回線の二重化

運用系と待機系内の通信系アダプタから出るそれぞれの通信回線が回線切替装置を経由して対ホストと接

続される．

運用系の切替え時は，自動的に回線切替えが行われ，ホストとの通信が継続される．

（4） LAN のアクセスパスの二重化

回線の切替え制御はEthernetやFDDIなどのLAN系のインタフェースにも適用される．

LAN の切替えでは，回線切替装置といった特別な装置を必要とせず，各処理装置系は，同一の LAN 伝

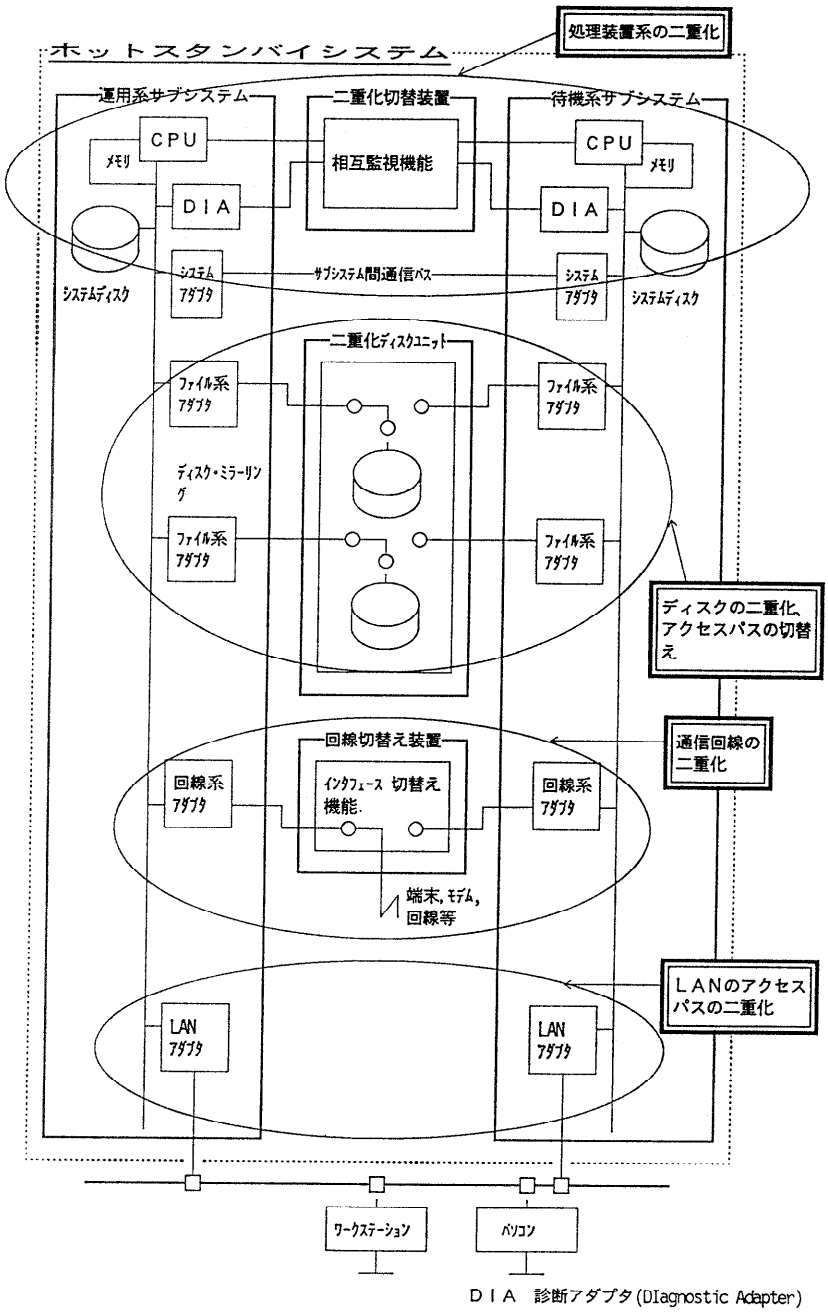


図 1 本システムの構成
Fig. 1 Example of hardware configuration.

送媒体に対して通常の接続形態と同じようにトランシーバやコンセントレータを介して接続する形態を採る.

運用系の LAN 系アダプタのみを活性状態とし，待機系の LAN 系アダプタは非活性状態で待機する.

運用系の切替え時は，旧運用系の LAN 系アダプタは非活性状態にし，新運用系の LAN 系アダプタを活性状態にする．この時，ネットワークアドレスの引継ぎを行う．LAN を介して通信する他の装置（ワークステーション，パソコン等）は，切替えによりいったん通信が途絶えても，再度同一のアドレスで運用系との通信を継続することができる.

3.2 処理装置系の切替え

運用系から待機系への切替えは，以下の手順で行われる.

（1） 運用系の異常発生が運用系と待機系間の相互の生存監視処理で検出される.

（2） 待機系は，共有資源を自系につなぎこみ，共通磁気ディスク上に記録されたチェックポイントデータを用いてファイル類を復元する.

（3） 待機系が運用系に替わって業務を再開する.

業務再開のために実行される以下のような一連の処理は，シェルスクリプトを使用し，自動的に実行される.

- 運用系が使用中であった共通磁気ディスクを引き継ぐ.
- 共通磁気ディスクに入出力中であったデータの整合性の回復処理.
- 運用系が使用中であった共有のネットワーク資源の初期化とネットワークアドレスの引き継ぎ処理.
- 運用で動作中であったアプリケーションを再起動する.

これ以降，待機系は運用系として管理される.

待機系が異常により停止した場合は，運用系にその旨が通知される．このことは運用系の主業務の継続に直接の影響はないが，これを契機に保守の依頼を行う等の必要な処理を行うシェルスクリプトを定義することができる.

3.3 高信頼化システムの課題

高信頼化システムの課題は，連続稼働性を確保し，万一の故障時でも業務の中断時間を極力短縮することである.

業務の中断時間は，ダウン検出時間，運用系の切替え時間および業務の引き継ぎ処理時間の和として表わされる.

相互監視処理はシステム本来の業務にとって不要な処理であるので，相互監視処理のオーバヘッドを削減しなければならない.

運用系の切替え時間および業務の引き継ぎ処理時間の短縮には，リカバリ処理の高速化を図らなければならない.

3.4 ダウン検出時間の短縮

相手系のダウンを検出する生存監視処理の手段としては，一般に「メッセージ交換方式」が採用される．この方式は，“I am alive” を意味するメッセージを運用系と待機系との間で定期的に相互に交換し合い，ある規定時間を越えてもメッセージが来なければ相手系に異常が発生したと判断する方法である.

メッセージ交換方式でダウン検出時間を短縮するためには，メッセージの交換周期を短くする必要がある．しかし，ダウン検出時間の短縮と相互監視処理のオーバヘッドの削減とは拮抗した関係にあり，メッセージ交換周期を短くすると監視処理のオーバヘッドが増加することになる.

また，メッセージ交換方式の欠点として，メッセージの交換周期を短縮すればする程，運用系と待機系の負荷の差異により，同期を取るのが困難になる点が上げられる．例えば，運用系で利用者業務の負荷が高くなり，メッセージ交換のための処理が動作できなくなると他方の相互監視はそれを異常と誤認する恐れがある.

本システムでは，生存監視処理としてメッセージ交換方式に比べてオーバヘッドの少なくかつ負荷の影響による誤動作の恐れがない相互監視機能を実現している.

さらに，相互監視処理においては，相互監視処理そのものの信頼性をいかに確保するかが重要である．そこで，本システムでは，相手系の異常検出精度の向上と誤認防止のために，

（1） 相互診断機能

（2） 障害原因問い合わせ機能

などを導入し，相互監視処理そのものの信頼度向上を図っている.

3.5 相互監視機能

相互監視機能は，ホットスタンバイシステムを構成する 2 つのサブシステムの正常性を相互に監視し合う機能である.

本システムの相互監視機能は，故障発生を故障が発生した側が自己申告で相手に通知する方法で実現する．この方法によれば，ダウン検出時間および監視処理に要する負荷をほとんどゼロにすることができる．

相互監視機能では，ソフトウェア故障，ハードウェア故障，電源故障などの異常発生時に，他系に対して異常通知の割り込みを発生し，自らの異常発生を自己申告通知する．**図2**に示すように，異常通知の割り込み発生を相互に監視できるように構成しておくことにより相互監視機能を実現している．

相互監視機能においては，運用系と待機系は全く同じ動作をするので，図2ではそれらを CPU #0 および CPU #1 と記している．

2つの CPU (CPU #0 と CPU #1) は，故障発生を通知する複数の割込み信号線で接続されている（以降，相互監視パスと呼ぶ）．この信号線は各々2本1組で構成され，1本は CPU #0 の故障発生を通知する割込み信号線（信号線 #0 と略す）で，もう1本は CPU #1 の故障発生を通知する割込み信号線（信号線 #1 と記す）である．

CPU #0 の側が信号線 #0 の値を ON にすることで CPU #1 側に割り込みが発生し，CPU #1 側が信号線 #1 の値を ON にすることで CPU #0 側に割り込みが発生する．

両方の CPU 上には割り込みが発生するのを待ち続けるプロセスが存在し，通常は OFF である割り込み信号線の値が ON になると，相手系に故障が発生したと理解する．

故障の通知を相手系に知らせる割込み信号には，

（a） ソフトウェア故障の通知信号
（b） ハードウェア故障の通知信号
（c） 電源故障の通知信号

がある．

ソフトウェア故障を通知する割込みには，他サブシステムで OS パニックが発生したことを通知する割り込みとウォッチドック・タイマがソフトウェアの故障を検出する割り込みの2つがある．パニックが発生通知の割り込みは OS パニックの発生とともに瞬時に発生する．

ウォッチドック・タイマとは，システムの異常を検出するために，システム内の状態を監視し，定められた時間（これを自系システム監視間隔と呼ぶ）を超えて状態の変化ない場合を異常とみなして検出するためのタイマである．

ウォッチドック・タイマを利用し，OS プログラムの異常ループ（無限ループ）やパニックに至らない OS プログラムの内部矛盾を検出する．

3.5.1 故障検出時の処理

相互監視機能で異常を検出した場合，障害原因問合せを行う．

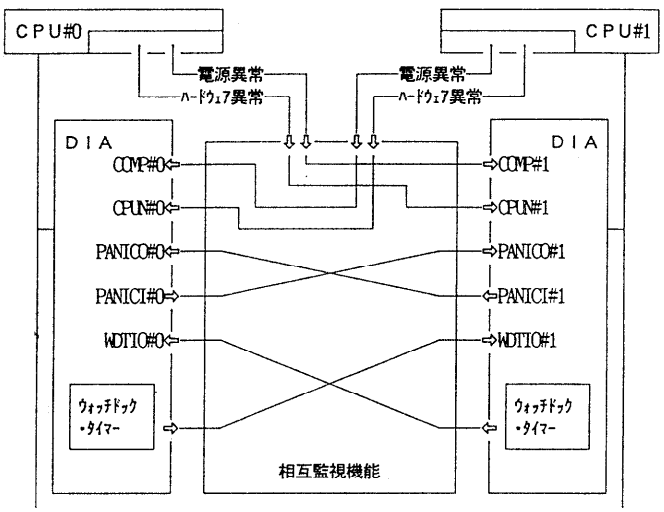
これを行う理由は，割込み発生の原因が相手系の異常によるものでなく，DIA (DIagnostic Adapter) の異常による可能性もあるためである．

信頼性は低下するが，異常発生時に運用系から待機系への切替え時間を短縮するために障害原因問合せを省略してすぐに状態遷移を開始するように設定することも可能である．

障害原因問合せ機能については後述する．

3.6 相互診断機能

相互診断機能は，通信パスを使用して運用系および待機系それぞれの処理装置間で相互診断問合せを行い，その応答により相互の処理装置の状態を確認し合う機能である．



（a） PANIC bit —OS パニックの発生を通知
（b） CPUN(CPu UNusual) bit —処理装置のハードウェア異常を通知
（c） COMP (COMPletion) bit —処理装置の電源異常を通知
（d） WDTI (Watch-Dog TImer)bit—ウォッチドック・タイマが検出する OS 異常を通知

図2 相互監視の動作概念
Fig. 2 Concept of mutual surveillance.

相互診断においては，一定の周期（相互診断周期：通常は1分）で待機系から運用系に対して相互診断問合せを行い，運用系がその問合せに対する応答を返信する．運用系では，待機系への応答を返信後，次の相互診断問合せが着信するまでの時間（これを T_m とする）を監視する．運用系では T_m が規定の時間以上であれば，待機系に異常が発生したと判断し，状態遷移処理を開始する．

待機系では，相互診断問合せを発信後，運用系からの応答を受信するまでの時間（これを T_s とする）を監視する．待機系では T_s が規定の時間以上であれば，運用系に異常が発生したと判断し，状態遷移処理を開始する．**図3**に，相互診断処理の流れを示す．

この処理は，どちらの処理装置が主体となってもよいが，一般には運用系の負荷を軽減するために，待機系主導で行う．

3.7 障害原因問合せ機能

障害原因問合せ機能は，相互監視処理の信頼性を向上させることを目的としている．

相互監視機能または相互診断機能により相手系の異常を検出した場合に，系間通信パスまたは相互監視パスを使用して障害原因を問い合わせる．

障害原因の問合せを行うのは，正常を異常と誤認しないように配慮するためである．

例えば，運用系で利用者業務の負荷が高くなり，定周期相互監視で異常と判定されたとする．しかし，実際には一時的（数秒程度）な負荷の集中に過ぎなかった場合，単純に異常と判定して直ちに切替えを行うのは適当ではない．

障害原因問合せにおいては，応答が返信されるまでの時間監視を行い，規定の時間以内に応答がない場合，相手処理装置に異常が発生したと見なし，状態遷移処理を開始する．**図4**に障害原因問合せ処理の流れを示す．

3.8 リカバリ処理

デュプレックス型ホットスタンバイ方式では，処理途中のチェックポイントデータを両系間で共通にアク

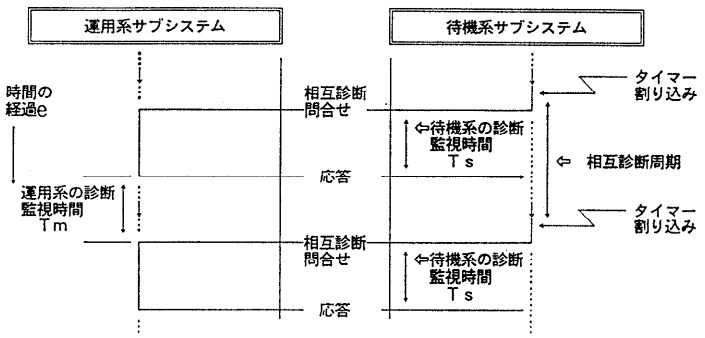


図 3 相互診断の処理の流れ
Fig. 3 Flow of diagnosing proccss.

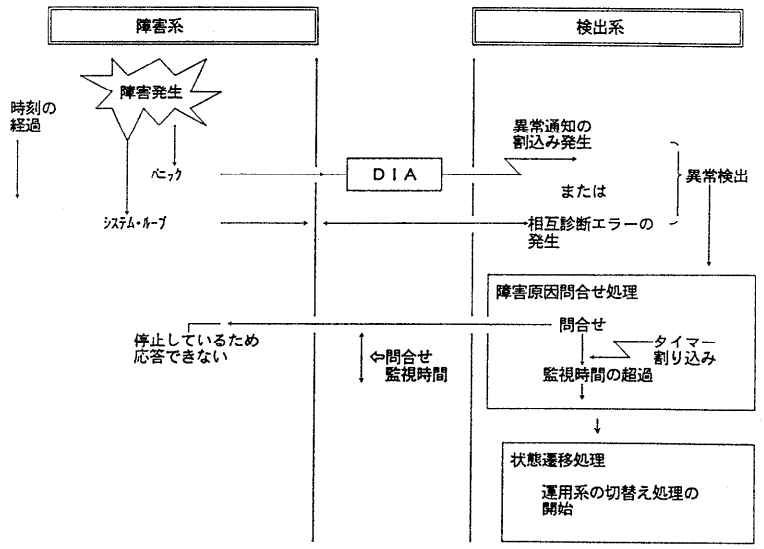


図 4 障害原因問合せの処理の流れ
Fig. 4 Operations for error inquiries.

セスできる記憶媒体に格納しておき，処理装置の切替えを契機に新運用系がリカバリ処理を行って業務を続行する．

業務中断時間を短縮するためにはリカバリ処理を高速化することが重要である．

リカバリ処理を高速化するためには，

（1） 高速なファイルシステムの引き継ぎ

（2） チェックポイントデータの高速な退避／復元

を実現しなければならない．

（1） 高速なファイルシステムの引き継ぎ

運用系に切替えが発生した場合，新運用系上のアプリケーションは，共通ディスク上のファイルシステムのリカバリ処理とファイルシステムのマウント処理が終了するのを待って処理の再開を行う必要がある．これに費やされる時間が業務中断時間の大部分である．

本システムでは，この業務の中断時間を短縮するために，

（a） データの入出力には直接転送方式を使用する

（b） ファイルは連続域ファイルとする

（c） 連続域ファイルの領域は事前に確保し，拡張は行わない

という条件の下で，ファイルシステムのリカバリ処理を不要にし，共通ディスク上のファイルシステムを両系間から同時にマウントできるようにしている．

（2） チェックポイントデータの高速な退避／復元

本システムではチェックポイントデータの格納媒体として，

（a） 共通磁気ディスク

（b） 共通半導体ディスク

（c） 共通メモリ

の3つを選択することができるようになっており，いずれの媒体も二重化されている．アプリケーションにおいては二重化されていることを全く意識する必要がない．

共通磁気ディスクおよび共通半導体ディスク内のファイルは，データが二重書きされるばかりでなく，クロスコール機能により最大4つのアクセス経路を定義してチェックポイントデータの退避／復元の確実化が図られている．

共通メモリは，主記憶と区別することなく read／write が行えるプログラム転送モードとリカバリ情報を高速に退避／復元するためのDMA転送モードの2つのアクセス手段とOSのパニック処理の延長でチェックポイントデータを自動的に退避する機能を実現している．

3.9 連続稼働性のための機能

システムを構成するハードウェア装置の中で，最も故障発生率が高いのはディスク装置である．

ディスク装置の故障に対して業務の連続稼働を保証するための技術として予備ボリューム機能を採り入れている．

3.9.1 予備ボリューム機能

ディスクの二重化によりデータのミラー化を実現しているが，2台のディスクの1台に故障が発生し，さらに残りの1台が故障するとデータは消失することになる．

このため，残りの1台が故障する前に，

- 故障ディスクの保守交換（システム一時停止）
- データのバックアップを頻繁に行う（アプリケーションの一時停止）

などの作業を行う必要がある．

しかし，システムの無停止化，業務の 24 時間連続稼働を実現するためには，このような作業を行うことはできない．

そこで，二重化されたディスクの組とは別に，ディスク故障発生時に一時的に二重化のペアとなることのできるディスク群を用意している．このディスク群を予備ボリュームと呼んでいる．

予備ボリューム機能は，ディスク装置の二重化構成の組合せを動的に変更するための機能である．

二重化ディスクの一方が故障した場合，予備ボリュームの中の1台が選ばれて二重化状態になる．

この時，予備ボリュームから取り込まれたディスクの内容は，自動コピー処理によって正常なディスクと等価な状態にされる．また，コピー中になされた書き込み要求は両方のディスクに反映される．

コピー作業中であっても，そのディスクを使っているアプリケーションを停止する必要がない．

このように，予備ボリューム機能によって，ディスク故障が発生しても，予備ボリュームの割り当て，予備ボリュームに対するコピー，故障ディスクの保守交換，新ディスクのシステムへの組み込みなど故障回復までの一連の処理は，アプリケーションに全く影響を与えずに行うことが可能である．

4. 評　価

4.1 故障シミュレーションによるダウンタイムの測定

実際のシステム運用上で想定される障害を擬似的に発生させ，異常検出から確定までに要する時間を測定した．

更に，異常確定から主業務を起動するまでの時間についても測定した．その結果を**表1**に示す．

疑似的に発生させた障害は以下の3種類である．

（1） 運用系の強制電源切断
相互監視による異常検出時間を測定する．

（2） 高レベルプロセスの無限ループ
相互診断による異常検出時間を測定する．

（3） panic コマンドによる緊急停止
運用プログラムの判断による緊急切替え要求に対する異常検出時間を測定する．

表1 故障シミュレーションによるダウンタイムの測定
Table 1 Measurement of down-time with breakdown simulation.

分類			測定値(秒)
切替え性能	異常検出から確定までの時間	相互監視による異常検出	5.8 (注1)
		相互診断による異常検出	57.0 (注1)
		panic コマンドによる緊急停止	1.2
	異常確定から利用者プロシジャの起動まで (注2)		9.0

注1：障害原因問合せの時間監視（5秒）を含む．
注2：ホスト計算機と接続するハイレベル手順の2回線を活性化する処理を含む．

4.2 応用システム事例

本システムはすでに幅広い分野で使用されている．一般社会での交通，環境の監視・制御を行う公共システム，組立・加工工場におけるファクトリオートメーションシステム，NC 工作機，ロボット，搬送機械などのサブシステムと組み合わせて工場全体を統合的に管理する CIM システム，データの集配信，交換回線制御，オンラインネットワーク等の通信制御システムなどである．

これらのシステムの共通点は，処理停止，データの損失が許されないことである．ここでは，某メガネレンズ加工製造工場への応用事例を紹介する．

本二重化システムは，メガネ小売店端末（パソコン：約4000台）から FENICS 網を経由してホスト計算機に入る特注品メガネレンズ受注に対して 24 時間以内の製品出荷を実現する CIM システムの中核をなしている．

本システムは，ホスト，本二重化システム，パソコンごとに役割分担した連携システムであることが特徴である．

ホスト　　：オーダ受注，データベース管理
本二重化システム：通信ゲートウェイ，工程管理，工作データの技術計算
パソコン　　：NC 工作機制御，製造ライン管理者用コンソール端末

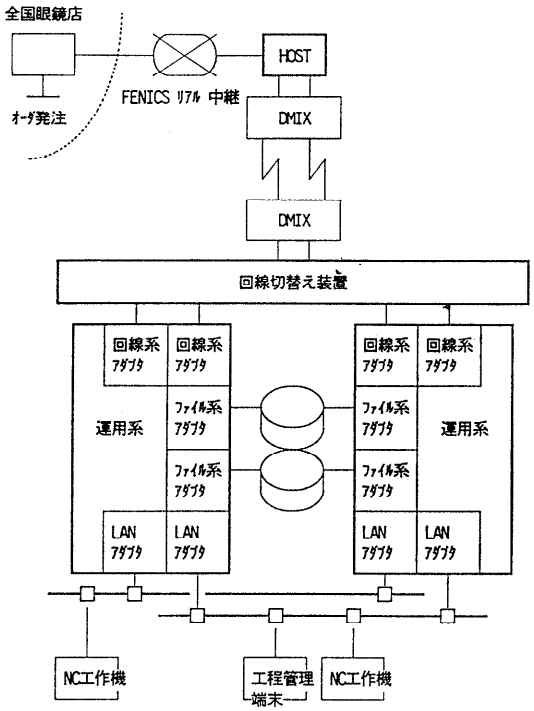


図5 適用事例
Fig. 5 Sample application of this system.

図5に適用事例の全体構成を示す．

5. おわりに

本論文では，高信頼化機能を追加した UNIX を OS とする待機冗長構成ホットスタンバイ方式による高信頼／高稼働システムについて述べた．

待機冗長方式の問題点は，相互監視処理のオーバヘッド，相互監視処理それ自体の信頼性確保および処理システム系切替え時に起こる処理の中断である．

本システムでは，相互監視処理を定周期相互監視と非同期相互監視の組合せで実現し，相互監視処理の

オーバヘッドを削減した．

また，相互監視処理に相互診断機能と障害原因問い合わせ機能の2つのメカニズムを新たに導入し，相互監視処理そのものの信頼性の向上を図った．

処理システムの切替え発生時の業務の中断については，系間のプロセス間通信機能とチェックポイントデータの高速な格納媒体サポートにより，業務中断時間の短縮化を図った．

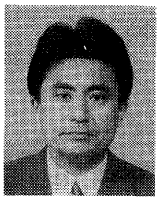
さらに本論文では，単なるハードウェアの故障だけでなく，OS 故障，アプリケーション故障に対しても有効な高信頼技術を提案した．

本二重化システムをベースにした高信頼システムは，既にトータリゼータ，新聞 CTS (Cold Type System)，金融オンラインシステムなどの分野で稼働している．また，本稼働後に本二重化システムの有効性（運用系のシステムダウンによる待機系への業務の円滑な引き継ぎ）も確認されている．

参考文献

1) 島田一洋，杉岡一郎：UNIX の高信頼化の一手法，信学論，Vol. J76-D-I, No. 1, pp. 31-35 (1993)．
2) 当麻喜弘監修，向殿正男編：コンピュータシステムの高信頼化技術入門，日本規格協会（1987. 3)．
3) 日本規格協会：システムの高信頼化技術に関する調査研究報告（1989. 3)．
4) 渡辺栄一：トランザクション処理システム，平成2年電気・情報関連学会連合大会プログラム，Vol. 5, pp. 25-28 (1990)．

（平成4年11月 4 日受付）
（平成5年 2 月12日採録）



島田 一洋（正会員）

1958 年生．1981 年室蘭工業大学工学部電子工学科卒．1981 年パナファコム株式会社（現株式会社 PFU）に入社．主に，ミニコンピュータの制御ソフトウェアの開発に従事．1993 年室蘭工業大学大学院工学研究科博士後期課程生産情報システム工学専攻修了．現在は，株式会社 PFU にて，UNIX をベースにした高信頼性／高稼働システムの研究・開発を行っている．工学博士．日本ソフトウェア科学会会員．

杉岡 一郎（正会員）

1941 年生．1965 年室蘭工業大学工学部電気工学科卒．1970 年北海道大学大学院工学研究科修士課程電気工学専攻修了．1971 年室蘭工業大学工学部電子工学科助手，1985 年同助教授．1990 年同大学地域共同研究センター助教授，大学と民間機関等との共同研究の推進およびセンター運営に従事．1993 年同大学工学部情報工学科教授．工学博士．主としてコンピュータの教育利用に関する研究に従事．電子情報通信学会，CAI 学会各会員．