

大容量ファイル制御方式†

高橋章二‡ 片岡雄二‡ 森原一郎‡

高密度記録技術の進歩に伴い大容量記憶装置 (MSS) のように磁気ディスク装置や磁気ドラム装置に比較してビットコストが安価で大容量な記憶装置が開発されている。このような特徴を有する MSS を用いたシステムについてもすでに報告されているが、これらは利用者あるいは利用者プログラムが装置種別を意識し対処するものである。筆者らは記憶装置の仮想化拡大を目的として、MSS をファイル管理プログラムで仮想化することにより MSS, DASD (ダイレクトアクセス記憶装置) を利用者プログラムから同一インタフェースで利用可能とする方式を実用化した。本論文では MSS をファイル管理プログラムで仮想化する場合生じる問題点、対処法、ならびに上記機能とマイグレーションを組み合わせた社内システムでの適用例について述べられる。

1. まえがき

情報処理技術の社会生活への浸透が深まるにつれ計算機で処理する業務範囲は拡大し、取り扱うデータ量も増大する傾向にある。このような背景から安価で大容量なファイルの提供が望まれている。

一方、高密度記録技術の進歩に伴い大容量記憶装置 (MSS) のように磁気ディスク装置 (DK) や磁気ドラム装置 (DR) 等のダイレクトアクセス記憶装置 (DASD) に比較して、ビットコストが安価で大容量な記憶装置が開発され^{1)~5)}、これを用いたシステムについても発表されている^{6)~8)}。

ところで、記憶装置の仮想化レベルについてはプログラムの階層構造より図 1 に示すように 4 段階に分類することができる。既存の MSS を用いたシステムはこの分類に従ってしてみると、レベル 1 (利用者プログラム) で仮想化したもの、言いかえれば利用者プログラムが MSS の存在を意識した対処を行うものである。

筆者らは記憶装置仮想化の拡大を図り、MSS と DASD を既存プログラムの中から同一インタフェースで使用可能とするため、図 1 に示すレベル 4 (ファイル管理レベル) で MSS を仮想化する技術を開発し、社内システムへの適用を図っている。

本論文では、DASD を主要な二次記憶媒体として利用者プログラムに対し仮想化処理を行っている既存 TSS* ファイル管理プログラムのなかで MSS の仮想

化を行う場合、DASD と MSS のハード構成、装置当りの記憶容量等の相違から生じるマウント制御、空間管理、障害処理等の問題への対処策が述べられる。また MSS と DASD の階層制御機能として開発したマイグレーション機能と組合せ検索処理を行う社内システムへの適用結果、MSS 上のデータ検索は 1~6 分で可能であることが述べられる。

2. ホストから見た MSS の構成

MSS のハードウェア構成を DIPS 4500 MSS⁹⁾ を例にとり、ホストシステム* から見た構成を図示すると図 2 のようになる。すなわち、ホストシステムから MSS を見ると仮想的なディスク装置 (VD) と仮想的な 100 MB のディスクパック (MSV) がそれぞれ独立に存在している。ディスク装置数とディスクパック数の関係は MSS のハード構成上、仮想ディスク装置数 ≪ 仮想ディスクパック数、であるためディスクパックを常時マウント状態にすることができない。このためホストシステムにおいては必要になった時点で所要のディスクパックを選択、ディスク装置にマウントし、不要になった時点でデマウントを行う。マウントが完了すれば実ディスク装置と同様の使用が可能となる。

3. MSS 導入時の問題点

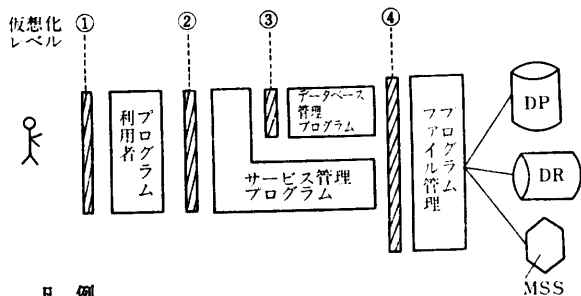
記憶媒体の仮想化とは利用者が意識する論理ファイルとオペレーティングシステムが管理する記憶媒体とを対応づけ、利用者の要求に応じたファイルアクセスを媒体の有効利用、I/O (入/出力) 処理の高速化等を維持しながら実施することと定義づけることができ

† Large Volume File Management Method by SYOJI TAKAHASHI, YUJI KATAOKA and ICHIRO MORIHARA (Data Processing Application System Sec., Yokosuka Electrical Communication Laboratory, N. T. T.).

‡ 日本電信電話公社横須賀電気通信研究所データ応用研究室

* Time-Sharing System

* MSS と接続し、MSS を制御する計算機システムをホストと呼ぶ。

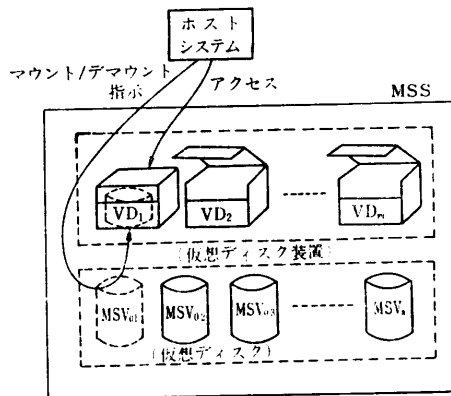


凡例

- ▨ : 記憶装置の仮想化処理
- ① : 利用者プログラムの中で記憶装置を仮想化する
- ② : サービス管理プログラムの中で記憶装置を仮想化する
- ③ : データベース管理プログラムの中で記憶装置を仮想化する
- ④ : ファイル管理プログラムの中で記憶装置を仮想化する

図 1 プログラム階層と記憶装置の仮想化

Fig. 1 Program hierarchy and virtualization of memory units.



MSV_n: ホストから仮想的に見える 100 MB DP の数
 VD_m: ホストから仮想的に見える仮想ディスク装置の数
 $n \gg m$

図 2 ホストシステムから見た MSS の構成

Fig. 2 Configuration of MSS looking from a host system.

る。

ところで既存の TSS システムではオンライン用二次記憶媒体として DASD が一般的に使用され、ファイル管理プログラムにおいて記憶媒体の仮想化がなされる。このようなシステムのファイル管理プログラムにおいて MSS を DASD と同様仮想化しようとすると、MSS と DASD のハードウェア構成、装置当りの

記憶容量等の相違から表 1 に示す問題が発生する。これら問題点を既存のファイル管理 (DIPS* TSS ファイル管理: DTF) との融合という観点で整理すると表 2 のようになる。

4. 主要方式

本章では 3 章で述べた問題点への対処方式、ならびに MSS の適用領域拡大を目的として開発したマイグレーション方式について述べる。

4.1 マウント制御

オンラインシステムにおける媒体のマウント/デマウント処理は二次記憶媒体として DASD が一般的に

表 1 MSS 導入時の問題点

Table 1 Problems of introducing an MSS.

要因	問題点
ハードウェア構成の相違	マウント制御
記憶容量の差	空間管理
	障害処理

表 2 DIPS TSS ファイル管理の特徴と MSS 導入時の問題点

Table 2 Characteristics of DIPS TSS file management program and problems for the introduction of MSS.

特徴	利用者からの見え方	ファイル管理での実現方法	DP と MSS の違いによる問題点	解決へのアプローチ
媒体不可視	ファイル作成時、どの装置/どの位置等の物理的な属性は意識せずファイルの大きさ(空間量)のみ指定すればよい。	システムで保有する全媒体の装置種別、使用量を一括管理することによりファイル空間を割り当てる最適な媒体を選択する。	(速度および媒体数の違い) (DP: 20 msec/1 レコード) (MSS: 10 sec/1 ステージング) (DP: 数十本) (MSS: 数千本) ファイルのシステム内分散配置 □ ガーベジ処理の長時間化 □ VD 不足 □ マウント処理に伴うオーバーヘッドの拡大	新しい空間管理の導入
論理的なファイル空間	ファイルの物理的な存在場所は意識せず、ファイル名+ファイル内相対位置を指定すればよい。	ファイル目録により作成済みファイルの物理的存在場所を管理	(装置制御の違い) (DP: レジデント) (MSS: ノンレジデント) □ DP ではとくにマウントを意識しない。	MSS のマウント制御
救済不可視	ファイル障害時はシステム側で自動的に復元するため利用者は救済手段/救済契機を意識する必要がない。	サービス・オフ時にシステムの全空間を MT に退避した全ダンプと、サービス中に取得したファイルの更新履歴(変分ダンプ)により媒体単位/ファイル単位に救済。	全ダンプ取得に数十時間かかり運用不可 ① グループ単位全ダンプ (全ダンプ対象媒体をグループ分割しサイクリックに回す) □ グループ分割しても 1 回に 5~6 時間かかり運用に耐えない。 ② オンライン全ダンプ □ VD 不足 ③ 変分マージロマージ中に MT 障害に遭遇したとき対象ファイル名が不明。	新しい救済方式の導入

* 電電公社が開発した大型計算機。

表 3 マウント/デマウント方式の比較
Table 3 Comparison of the methods for mount and dismount MSV.

マウント/デマウント方式	概 要	利 点	欠 点
スタティック方式	会話処理の開設時/終了時をマウント/デマウントの契機とし、利用者が契約している全 MSV をマウントする。	<ul style="list-style-type: none"> • 会話処理実行中はマウント/デマウントのためのオーバーヘッドはない。 • VD をあらかじめ確保してあるため会話処理実行中に VD 不足となることはない。 	<ul style="list-style-type: none"> • 不要な MSV もマウントするため、システム全体として VD 不足となる危険性が高い。 • 他利用者の MSV をマウントできないためファイルの共用ができない。
ノミネート方式	会話処理の開設時/終了時をマウント/デマウントの契機とし、会話処理で使用するファイル名を開設時に利用者が指定する。	<ul style="list-style-type: none"> • 不要な MSV はマウントする必要がない。 • 会話処理実行中はマウント/デマウントのためのオーバーヘッドはない。 • VD をあらかじめ確保してあるため会話処理実行中に VD 不足となることはない。 	<ul style="list-style-type: none"> • あらかじめ必要なファイルをスケジュールしなければならない (DP を媒体とした場合と異なる)。 • 会話処理実行中に急に別 MSV が必要となった場合はマウント不可。 • ファイル作成のみの会話処理は不可。
オンデマンド方式	ファイルアクセス、ファイル作成等で MSV が必要となったときをマウント/デマウントの契機とし、必要な MSV のみをマウントする。	<ul style="list-style-type: none"> • マウント/デマウントする MSV やファイル名を利用者が意識する必要がない。 • 会話処理実行中に必要となった MSV のみマウントすればよく、使用する VD は最小限である。 • ファイル管理以外のプログラムは既存のまままでよい (資産の有効利用が図れる)。 	<ul style="list-style-type: none"> • 同一 MSV を何度も必要とする場合はマウント/デマウントのためのオーバーヘッドあり。

用いられる関係上、オーバーヘッドの回避等を目的にシステムの初期設定 (IPL: Initial Program Load) 時にマウントし、システム終了時にデマウントする。一方、MSS は 2 章で述べたとおり、仮想ディスクパック数 ≧ 仮想ディスク装置数、であるため DASD と同様一律にマウント/デマウントすることができない。このためダイナミックな制御が必要となる。また、マウント処理を完了すると以降は DASD のアクセスロジックを活用することが可能となる。このような観点からファイル管理プログラムにおいて MSS を DASD と同レベルに仮想化するということは、マウント/デマウント制御をどの契機で実施するかに着目させることができる。

マウント/デマウント契機としては、(1) 処理の開始/終了に当り利用者の契約 MSV 全体をマウント/デマウントするスタティック方式、(2) 利用者が使用するファイルを指定するノミネート方式、(3) 処理実行中に必要となった時点でマウント/デマウントするオンデマンド方式がある。これら方式の特徴を表 3 に示す。このうち MSS の仮想化という観点からすると方式 (1)、(3) が適用可能であるが、(1) については表 3 の欠点のなかで述べたとおり VD 不足からジョブの実行がデッドロックになる危険が高いため、方式 (3) を選択した。またこの場合、既存の利用者プログラムにインパクトを与えず実現するためには何を契機とするかが重要な課題となる。

4.1.1 オンデマンドなマウント/デマウントの契機

オンデマンドなマウント/デマウントはジョブ実行におけるファイル資源の確保/解放契機に一致させ対応するのが効果的である。DIPS における会話処理

ではファイルの作成、プログラムの実行等論理的なひとまとまりの処理を論理的処理単位 (LU: logical unit) とし、資源の確保、解放、障害復元等は LU 単位で行っている^{10)~12)}。LU 開始時点では必要なすべてのファイルが何であるかを陽にできないためファイル資源の確保は各ファイル個々に OPEN マクロを契機としてオンデマンドに行っている。また資源の解放契機は障害対策上 LU の終了宣言 (RPU* マクロ発行) 時点としている。MSV のマウント/デマウントは上記ファイル資源の確保/解放に一致させて考えればよく OPEN/RPU 各マクロ内で対処する。

なおオンデマンドなマウント/デマウントを行うにあたっては以下の二つの問題について多重制御方式によりあわせて対処している。

(1) 1 LU 内で複数のファイルを使う場合、各ファイルに対し OPEN マクロが発行されるが、これらファイルが同一 MSV に格納されているとマウント済みの MSV に対し再度マウント処理を行うのはオーバーヘッドを増大させることになる。また何らかの原因で RPU を経由せずにジョブの終了することがある。この場合、MSV のデマウントが行われず VD を持ち切りにする問題がある。

(2) 同時に走行している複数のジョブで同一のファイルを使用するような場合、(1) で述べたマウントのオーバーヘッドとともに、先行するジョブで RPU 契機となると後続ジョブでアクセスしている MSV をデマウントしてしまう危険がある。

これらに対処するため、(1) についてはジョブ内マウント制御を、(2) についてはジョブ間マウント制御を多重制御方式として採用している。

* Restart Process Unit.

4.2 空間管理

TSS ファイル管理では不特定多数の利用者間でのファイル共有やインプリシットなワークファイルの生成/消去等を容易に行えるように、利用者は論理ファイル名ならびにその大きさのみを意識すればよいことを目的として媒体の仮想化を実施している。これは、①ファイル作成時に空間割付け可能な媒体の選択処理（アロケーション処理）と、②ファイルへのアクセス時にファイル名から物理的存在場所の検出処理（カタログ処理）により実現している。

MSS を DASD と比較すると、(i)装置当りの保有媒体数が非常に多い、(ii)ダイナミックなマウント処理が必要という違いがある。この観点で上記①の処理を見ると、とくに媒体選択時のオーバーヘッドの極小化ならびにアクセス時のマウント処理の極小化の二つを考慮したファイルのアロケーション処理が必要である。

本節では、ファイルのアロケーション処理に対して導入した専用ボリューム方式および予備ボリューム方式について述べる。

なお②は(i)、(ii)ならびにステージング処理を除き MSS は DASD と等価であることから既存の管理方法（カタログ管理）がそのまま適用可能である。

4.2.1 専用ボリューム方式

ファイルのアロケーション処理において、システム全体の媒体を対象とした場合は空間使用効率は向上するが、(1)媒体数が多くなると媒体選択処理のオーバーヘッドが大きい、(2)1利用者のファイル群はシステム全体に分散配置される（逆に見れば1媒体内に複数人のファイルが存在する）ことから、媒体障害の波及範囲が不特定の複数人におよぶという問題がある。また、利用者プログラムで複数のファイルを使用する場

合にマウント処理のオーバーヘッドが大きい、VD 不足を引き起こしやすい、という問題もある。以上より媒体数が多くなるほど、1利用者のファイル群はできるだけ小さい範囲に集中させる必要があることがわかる。

利用者と媒体の固定方法（専用ボリューム化）として、(1)契約空間量に見合う媒体本数をあらかじめ割り当てておく方法（静的割当て方法）と、(2)媒体に最初にファイルを作成した利用者に固定する方法（動的割当て方法）の2通りがある。表4に両方式の比較を示す。静的割当て方式は各利用者の空間使用率が低いとシステム全体の空きエリアが大きい、空間が固定されるため使用率が高いとマルチボリューム割当てとなる等の問題がある。動的割当て方式はファイル作成時に MSV 割当て処理の必要となる可能性が高くオーバーヘッドを招く、アロケーション方法が必然的に1媒体に集中するためファイルの拡張に弱いという問題がある。

MSS のもつ廉価かつ多媒体という特徴を効果的に生かし、TSS ではファイルの動的拡張が多い、という点に着目すると利用者への媒体固定方法は静的割当て方式がよいことがわかる。

4.2.2 予備ボリューム方式

本項では静的割当て方式におけるマルチボリューム化の解決法について述べる。

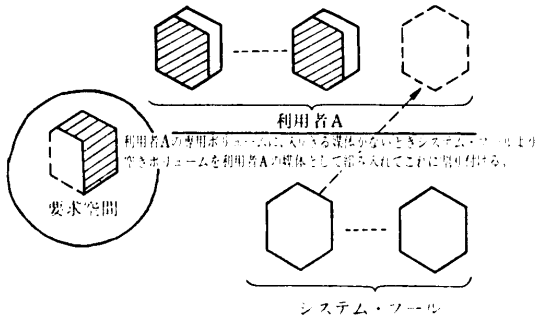
MSS 上にファイルのアロケーションするに際して

- (a) 動的拡張に対処可能なように空エリアを残す
- (b) 新規アロケーション時にマルチボリュームにしない

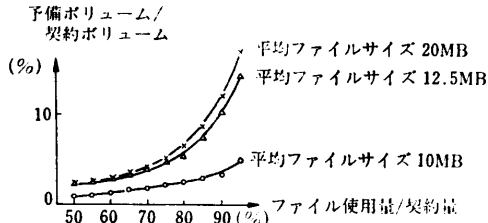
という相反する条件を満たす必要がある。大容量ファイル制御方式では、静的割当て方式をベースとし、アロケーション時にオーバーフローすると MSV を動的に

表4 専用ボリューム化の比較
Table 4 Comparison of the private volume method.

専用ボリューム方式	概要	利点	欠点	MSS 上にファイルのアロケーションするときの注意点
静的割当て方式	契約時点で契約空間量に見合う MSV をあらかじめ割り当てる。 (ファイル作成時は割り当てた各 MSV の空間使用率を平均化するようにアロケーションする。)	<ul style="list-style-type: none"> ・契約変更に伴う利用者への MSV 割当てはオンライン作業で行えるためオンライン処理への影響はない。 ・割り当てた MSV 内で平均的な空間使用が行えるためファイルの拡張に強い。 	<ul style="list-style-type: none"> ・利用者の空間使用率が低い場合、無駄なエリアが多くなる。 ・利用者の空間使用率が高いとマルチ・ボリュームにアロケーションされる。 	<ul style="list-style-type: none"> ・マルチボリュームへのアロケーションを回避する。 ・ファイルの動的拡張用エリアを残す。
動的割当て方式	契約時点では MSV の割当ては行わず、ファイルの作成時に必要量に見合う MSV を割り当てる。 ファイル作成時は確保済み MSV にアロケーションする。 (該 MSV がオーバーフローするときに新規に MSV を確保してアロケーションする。)	<ul style="list-style-type: none"> ・利用者の空間使用率に従って MSV を用意すればよく、とくに空間使用率が低い場合に MSV の有効利用が図れ、コストダウンができる。 ・マルチ・ボリューム・アロケーションが回避できる。 	<ul style="list-style-type: none"> ・ファイル作成時に MSV 割当て処理が入ることがあり性能への影響が大きい。 ・ファイルのアロケーションが1媒体に集中するためファイルの拡張に弱い。 	<ul style="list-style-type: none"> (ファイルのアクセス時、およびファイルの動的拡張時に不要なマウント動作を極小化する効果がある。)



(a) 予備ボリューム方式の概念



(b) 予備ボリュームの準備率

図3 予備ボリューム方式とファイルサイズ、空間使用量の関係
Fig. 3 Spare volume method and relation between file size and space usage.

割り当てるという予備ボリューム方式を導入することにより上記(a), (b)二つの条件を解決している。

〈予備ボリューム方式〉

- ① 利用者ごとに契約数分の媒体を割り当てる(静的割当て)。
- ② ファイルのアロケーション時、各媒体の空きエリアが少ないためにマルチボリュームとなるときは、システムプールの媒体(予備ボリューム)を利用者媒体として組み入れ、その媒体にアロケーションする(動的割付け)。

図3に契約媒体数と空間使用率による予備ボリュームの準備率を示す。たとえば、予備ボリュームとして契約数の約10%を用意することでファイル空間使用率80%まではマルチボリューム化が避けられると同時に、利用空間の20%を動的拡張用に使用できる。

4.2.3 専用ボリューム管理方法

4.2.1~2項で述べたことより、専用ボリュームの管理に必要な情報は

- (1) システムプールとしての媒体情報

- (2) 利用者に割り当てた媒体情報
- (3) 作成済みファイルと物理的所在場所を対応づけるカタログ情報

の三つである。(1)~(3)の各情報の内容、関連を図4に示す。ここで(3)のカタログ情報はDASDのカタログ情報と共用しており、専用ボリューム管理として特有なものは(1), (2)の媒体情報である。

4.3 障害対策

利用者ファイルの障害救済をどのレベルまで保障するかは、コストおよび処理速度とのトレードオフであり、利用者から要求される救済レベルはその使用目的によって異なる。一般に利用者ファイルの障害対策として、全ダンプ/ジャーナル情報を取得する方法およびファイルを二重化する方法がある(その特徴を表5に示す)。しかし、トータルの記憶容量が数十GBといっ

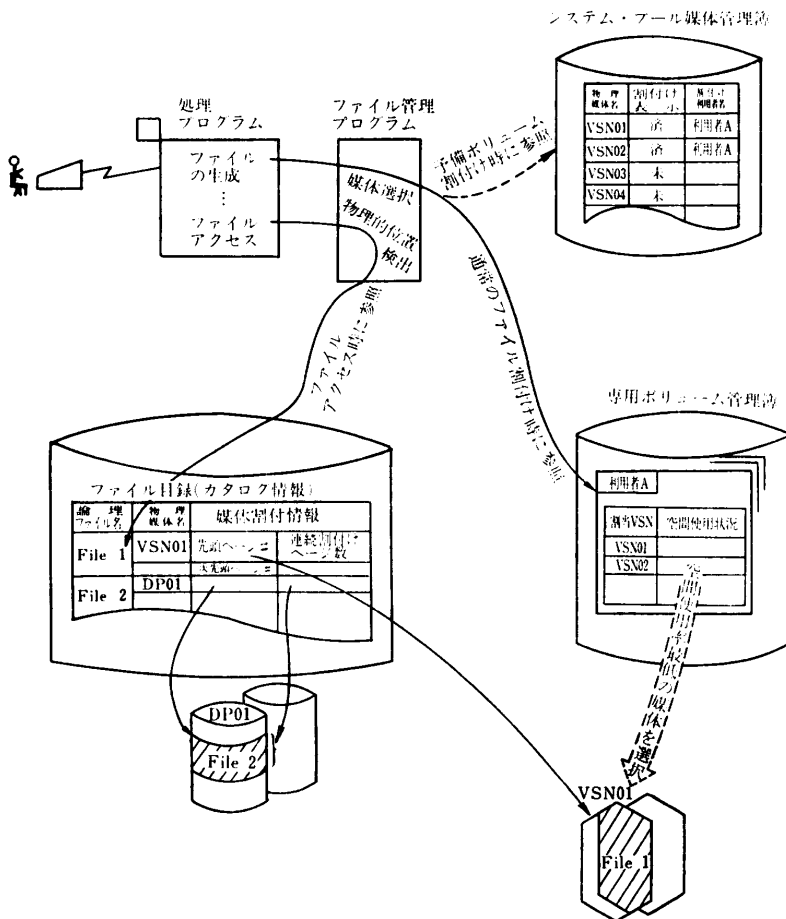


図4 DIPS TSS ファイル管理における空間管理
Fig. 4 Space management method with DIPS TSS file management program.

表 5 障害処理方式の比較
Table 5 Comparison of failure countermeasure.

方式	方式概要	利用者からの見え方	適用分野	備考
全ダンプ/ジャーナル	ファイルの全内容をあらかじめダンプした全ダンプ情報と、ファイル更新が実施されるごとに更新部分を記録するジャーナル情報をそれぞれ MT に取得し、障害時はこれらより媒体単位、ファイル単位に復元する。	救済情報（全ダンプ/ジャーナル）取得契機、復元契機を利用者は意識する必要はない。	{空間量 小} {更新量 小} {更新頻度 大} のシステムに向く	システム一憲
二重化	正面、副面を別媒体上に取り、障害時は正常面より媒体単位、ファイル単位に復元する。	全ダンプ/ジャーナル方式と同様に、利用者は救済情報取得(二重化)契機、復元契機を意識する必要はない。	{空間量 大} {更新量 中} {更新頻度 大} のシステムに向く	システム一憲/ 利用者指定 のどちらでも可。

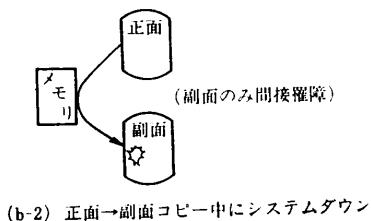
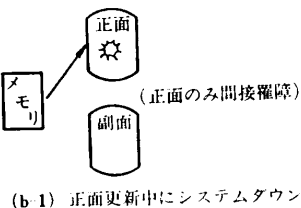
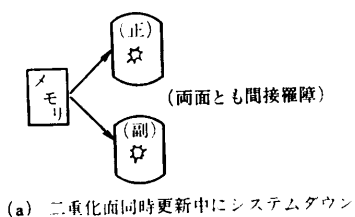


図 5 システムダウンによる間接障害
Fig. 5 Indirect trouble occurred from system down.

た MSV をファイル記憶媒体としてもつシステムでは、ダンプ情報取得に十数時間を要する全ダンプ/ジャーナル方式を採用することは運用上現実的でない。そこで、MSV そのものを救済情報格納媒体として使用する「二重化方式」を使用目的に合わせて利用者が自由に選択できる機能を提供している。

二重化方式としては、二重化面を同時更新するのではなく、WRITE 時に正面のみを更新し、RPU 時に副面へコピーすることにより更新中のシステムダウン等の間接障害にも対処可能としている点が特徴である(図 5 参照)。また、副面へは更新部分のみをコピーすることにより更新処理の高速化を図っている。

なお、二重化されたファイルの管理は図 6 に示すフ

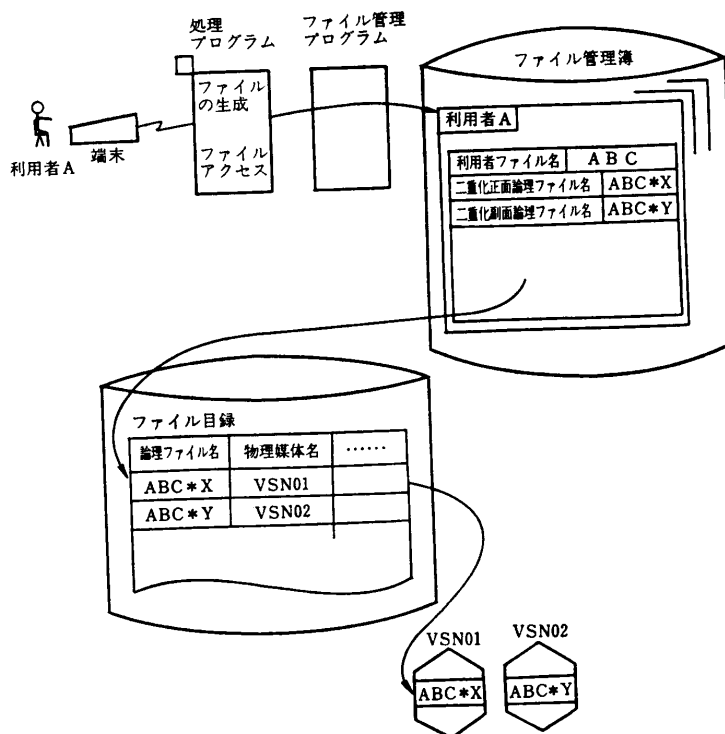


図 6 二重化ファイルの管理方法
Fig. 6 Management method of doubly file.

イル管理簿で行っており、利用者には一つのファイルと見える。これにより利用者はファイル作成時に二重化を指定するだけで、ファイルへのアクセスに際しては二重化を意識する必要がない。

4.4 マイグレーション

MSS の大容量、低価格性と DASD の高速アクセス性を組み合わせた使用を可能とするため、DP と MSV 相互間でファイル実体の移送を行うマイグレーション機能 (DP から MSV への移送は“アーカイブ”、MSV から DP への移送は“リコール”と呼ぶ) を実現している¹³⁾⁻¹⁵⁾(図 7 参照)。

ファイルを DP, MSV のどちらに格納するかは、コストと処理速度のトレードオフであり、利用者の使

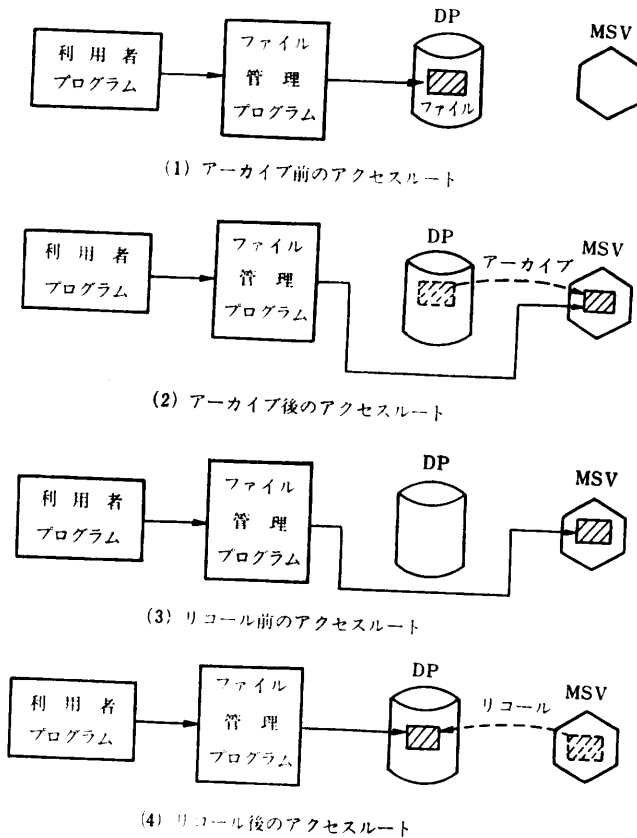


図7 マイグレーションの概念
Fig. 7 General idea of migration.

用目的によって左右されるものであるため、たんにアクセス頻度等からシステムで一意に決めることはできない。そこで利用者が使用目的に合わせて自由にアーカイブ/リコールを行うことができるようにマイグレーション用のコマンドを提供する(ただし、ファイルへのアクセスに際して、利用者はそのファイルがDP, MSV どちらに格納されているかを意識する必要はない)。さらに、会話処理におけるファイル更新は部分更新/追加が主であることに着目し、障害対策も考慮して以下のバリエーション機能を提供している。

- ① リコール後も MSV 上のファイルを消去せずにアーカイブ処理ではリコール後に更新/追加のあった部分のみアーカイブすることにより処理の高速化を図る(更新/追加がなければ DP 上のファイルを消去するだけである。図8参照)。
- ② MSV を DP 上のファイルのバックアップ用媒体として利用できるように、ダンプ相当機能としてアーカイブ処理後も DP 上にファイルを残す

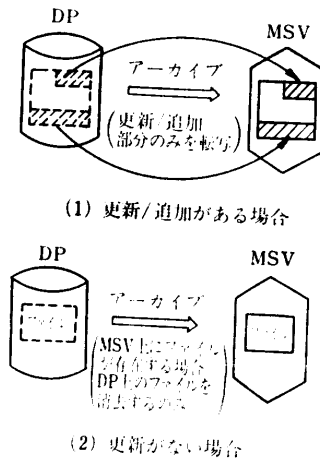


図8 アーカイブ処理の高速化
Fig. 8 High-speed archive method.

オプション機能を提供する(ファイルはリコール状態)。

なお、上記機能は大量のファイルを一齐にアーカイブする必要がある場合にも有効である。つまり、利用者はあらかじめ②で示したアーカイブ処理を順次行うことにより、リコール状態でサービスを受けることができ、必要な時点で一齐にアーカイブ(DP上にファイルを残さない)をすればたんにDP上のファイルが消去されるだけでアーカイブ処理が完了する。

5. 評価と適用例

大容量ファイル制御方式のオーバーヘッド要因、および応答特性を評価する。

5.1 オーバヘッド要因

MSS を DASD と同一レベルで使用するため必要となる制御機能は

- (1) ファイル作成/アクセス時の媒体識別処理
- (2) MSV のマウント/デマウント処理
- (3) 二重化ファイルの副面更新処理

である。このうち、(1)はファイルの空間アロケーション時およびOPEN時に必要となる機能であり、(2)はファイルのOPEN時およびLU終了時に必要となる機能であり、両機能ともファイルのI/Oに与える影響はない。また(3)はLU終了時に必要となる機能であり、データアクセス時への影響はない。したがってMSSをファイル管理レベルで仮想化する場合、性能上、既存部分に与える影響はほぼ零で可能となった。

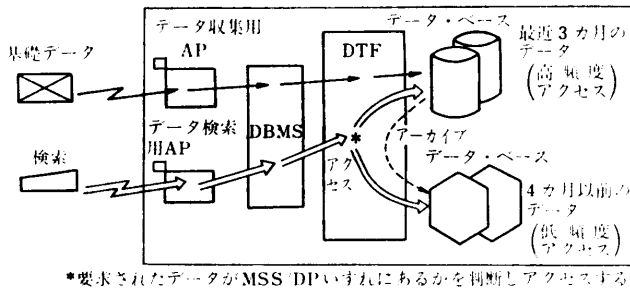


図 9 適用システムの概要
Fig. 9 Outline of application system.

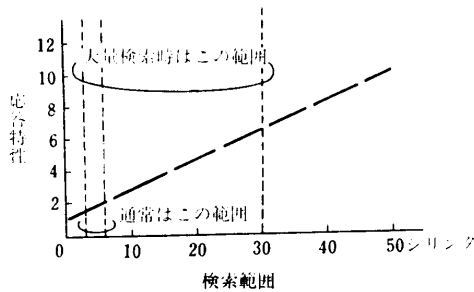


図 10 MSS 上のデータの検索応答特性
Fig. 10 Characteristic curve of retrieve data on MSS.

5.2 適用例と応答特性

本論文で述べた方式は社内データの管理、検索システムへ適用中である。適用システムの構成概要を図 9 に示す。

本システムでは月単位にデータの管理を行っており、データの収集は DASD 上で実施する。DASD 上には最近 3 カ月のデータを保管し、4 カ月を経過したデータはアーカイブ処理により DASD から MSS へ移送する。端末からの検索処理に対しデータベース管理システム (DBMS) は DTF に対しデータへのアクセス要求を行う。DTF はファイル名よりアクセスに必要なファイルの所在が DASD, MSS を識別し、MSS であればマウント処理の要否判断後データを読み出し、DBMS へ渡す。

本システムの検索特性は、1 検索当りの参照範囲が 0.2~0.5 MB 程度の小量検索と、最大 6 MB の大量検索の 2 種類がある。図 10 に両検索特性における処理時間を示す。図に示すように MSS 上のデータ検索は参照範囲により 1~6 分程度で可能である。

6. む す び

MSS の仮想化レベルをファイル管理とすることで MSS を DASD と同一のインターフェースで使用可能

とした。本論文では本方式を既存のファイル管理プログラム上で実現する時の問題点、対処策について述べた。結果を要約すると以下のとおりである。

(1) MSS の仮想化は記憶媒体である MSV とホストからのアクセス先である VD 数の差から生じる問題の対処と定義づけ、MSV の VD へのマウント処理をファイル管理のなかで動的に行うことにより対処した。この場合、マウント制御のオーバーヘッド防止のため、マウントの多重制御が有効であることを示した。

(2) MSS を既存プログラムの中でサポートするには空間管理、障害処理に関し、MSS の大容量性を考慮する必要があることを示し、前者については MSV と利用者を 1 対 1 に対応づける専用ボリューム方式、後者については二重化処理で対処した。

(3) 本論文で提示した方式を社内システムへの適用の結果、MSS 上のデータ検索処理は検索対象範囲により 1~6 分程度で可能となる見通しが得られた。

今後本論文で示した方式の評価を適用システムの中でさらに進めてゆく予定である。

謝辞 本研究の遂行にあたりご指導を賜った横須賀電気通信研究所 高村データ処理研究部長、橋本統括役、山本データ応用研究室長に感謝いたします。

参 考 文 献

- 1) Johnson, C. T.: The IBM 3850 A Mass Storage System with Disk Characteristics, *Proc. IEEE*, Vol. 63, No. 8, pp. 1166-1170 (1975).
- 2) Control Data Mass Storage Facility, CDC Japan Pub. No. J 201 (1975).
- 3) 伊藤: DIPS 用超大容量記憶装置の実用化, *通研実報*, Vol. 29, No. 2, pp. 1-10 (1980).
- 4) 浅野他: FACOM 6450 大容量記憶システム *FUJITSU*, Vol. 31, No. 1, pp. 93-110 (1980).
- 5) 大塚他: MDF システムの概要, *日電技報*, No. 136, pp. 56-59 (1980).
- 6) 野村他: MSS による大規模ソフトウェアのファイル維持管理法, 第 21 回情報処理学会全大, pp. 335-336 (1980).
- 7) 野上他: 大容量記憶システムによるデータベースの運用, 第 21 回情報処理学会全大, pp. 601-602 (1980).
- 8) 金沢他: 大容量記憶システムの利用特性と動作解析, *計算機システムの解析と制御*, 13-2 (1981).
- 9) 若林他: DIPS 用超大容量記憶制御系装置, *通研実報*, Vol. 29, No. 2, pp. 11-24 (1980).
- 10) 赤木他: DIPS タイムシェアリング用 OS にお

- けるサービス機能の制御方式, 通研実報, Vol. 28, No. 12, pp. 2675-2696 (1979).
- 11) 大沢他: 大規模タイムシェアリングシステムにおける資源制御方式, 通研実報, Vol. 28, No. 12, pp. 2657-2674 (1979).
 - 12) 村田他: DIPS 163-10 TSP の実用化, 通研実報, Vol. 30, No. 2, pp. 41-59 (1981).
 - 13) 坂他: アーカイバルファイル管理システム, 情報処理, Vol. 21, No. 4, pp. 391-395 (1980).
 - 14) IBM: OS/VS 2 MVS Hierarchical Storage Manager: General Information, IBM マニュアル, GH 35-0007-2 (1978).
 - 15) 藤井他: 階層的ファイル自動管理システムの設計, 情報処理学会論文誌, Vol. 21, No. 6, pp. 442-453 (1980).

(昭和58年2月10日受付)

(昭和58年4月19日採録)