(19)日本国特許庁(JP)　(12)特　許　公　報(B2)　(11)特許番号
特許第5028482号
(P5028482)
(45)発行日　平成24年9月19日(2012.9.19)　(24)登録日　平成24年6月29日(2012.6.29)

(51)Int.Cl.　F I
G06F 12/00 (2006.01)　G06F 12/00 531M
G06F 12/00 531J

請求項の数 14　(全 23 頁)

(21)出願番号	特願2009-513156 (P2009-513156)
(86)(22)出願日	平成19年4月26日 (2007.4.26)
(65)公表番号	特表2009-539178 (P2009-539178A)
(43)公表日	平成21年11月12日 (2009.11.12)
(86)国際出願番号	PCT/US2007/010304
(87)国際公開番号	WO2007/139647
(87)国際公開日	平成19年12月6日 (2007.12.6)
審査請求日	平成22年4月2日 (2010.4.2)
(31)優先権主張番号	1283/DEL/2006
(32)優先日	平成18年5月29日 (2006.5.29)
(33)優先権主張国	インド(IN)
(31)優先権主張番号	11/461,846
(32)優先日	平成18年8月2日 (2006.8.2)
(33)優先権主張国	米国(US)

(73)特許権者　500046438
マイクロソフト　コーポレーション
アメリカ合衆国　ワシントン州　９８０５２－６３９９　レッドモンド　ワン　マイクロソフト　ウェイ
(74)代理人　100077481
弁理士　谷　義一
(74)代理人　100088915
弁理士　阿部　和夫
(72)発明者　マノジュ　ケー．バリヤパランビル
アメリカ合衆国　９８０５２　ワシントン州　レッドモンド　ワン　マイクロソフト　ウェイ　マイクロソフト　コーポレーション　インターナショナル　パテンツ内

最終頁に続く

(54)【発明の名称】頻繁なアプリケーション整合バックアップの効率的な作成

(57)【特許請求の範囲】
【請求項１】
１つまたは複数のプロダクションサーバが１つまたは複数のボリューム上の保護すべきデータを１つまたは複数のバックアップサーバでバックアップするコンピュータ化された環境中の、プロダクションサーバにおいて、最近のデータを前記バックアップサーバから容易に復旧できるように、プロダクションサーバのデータを実質上連続的かつ整合性のある方式で複製する方法であって、
第１の時間インスタンスで、１つまたは複数のボリュームのデータのコピーであって、前記１つまたは複数のボリュームの前記データのフルベースラインに対応する、前記コピーをプロダクションサーバから作成すること、
前記１つまたは複数のボリュームの、前記第１の時間インスタンスで整合性がある、前記データの前記コピーを前記プロダクションサーバからバックアップサーバに送信すること、
前記第１の時間インスタンスの後に、前記１つまたは複数のボリューム上の前記データに対する１つまたは複数の変更であって、前記１つまたは複数の変更のうちの少なくとも１つには、前記プロダクションサーバにおけるファイルパスが前記バックアップサーバにおける前記ファイルへのパスとは異なるように、前記プロダクションサーバでの、前記１つまたは複数のデータ変更のいずれかに対応するファイルのファイルパスに対する変更が含まれる、前記１つまたは複数の変更のそれぞれについての指示を、前記プロダクションサーバ上の揮発性メモリ内に記憶される１つまたは複数のビットマップの状態で記憶する

こと、
複製サイクルのイベントを識別したとき、前記１つまたは複数のビットマップを、前記プロダクションサーバの永続記憶装置に記憶された１つまたは複数のログファイルに保存することであって、前記１つまたは複数のデータ変更は第２の時間インスタンスで整合性がある、１つまたは複数のログファイルに保存すること、
前記１つまたは複数のビットマップを前記揮発性メモリから削除すること、
前記１つまたは複数のボリュームの前記１つまたは複数のデータ変更を識別するために、前記１つまたは複数のビットマップの前記指示を使用すること、
前記プロダクションサーバと前記バックアップサーバとにおける前記ファイルへの前記パスを相関させ、それにより前記ファイルに対する新しい変更を前記ファイルへの前記パスの変更と共に前記バックアップサーバに送信できるようにすることであって、前記パスを相関させることは、
前記プロダクションサーバでＵＳＮジャーナルを少なくとも１回目に走査してファイルパスの前記変更をキャッシュすること、
前記プロダクションサーバで前記ＵＳＮジャーナルを少なくとも２回目に走査して前記最初のファイルパスを識別すること、および、
前記１回目および２回目の走査に基づいて前記バックアップサーバにおける前記ファイルへの調整済みパスを計算すること
を備える、前記バックアップサーバに送信できるようにすること、ならびに
前記１つまたは複数のボリュームの前記１つまたは複数のデータ変更のコピーを前記バックアップサーバに送信することを備え、前記バックアップサーバが、第１の時間インスタンスおよび第２の時間インスタンスで妥当である前記１つまたは複数のボリュームのデータのコピーを有することを特徴とする方法。

【請求項２】
前記ボリュームのファイルレベルのデータ変更を、変更フィルタ、変更ジャーナル、ＵＳＮジャーナルのうちの１つに保存すること、をさらに備えることを特徴とする請求項１に記載の方法。

【請求項３】
前記１つまたは複数のボリュームログファイルを、前記変更フィルタ、変更ジャーナル、ＵＳＮジャーナルのうちの１つと相関させて、各変更済みファイル中の前記１つまたは複数のデータ変更に対応する１つまたは複数の変更済みファイルを識別すること、をさらに備えることを特徴とする請求項２に記載の方法。

【請求項４】
前記１つまたは複数のボリュームログファイル中で、バイトレベルとバイトブロックレベルのいずれか一方で前記１つまたは複数のデータ変更に印を付けること、をさらに備えることを特徴とする請求項１に記載の方法。

【請求項５】
前記第２の時間インスタンスに対応する前記１つまたは複数のメモリ内ビットマップをフリーズすること、および、
第３の時間インスタンスでの、前記１つまたは複数の変更済みファイルへの新しい書込みに対応する１つまたは複数のメモリ内ビットマップの新しいセットを作成すること
をさらに備えることを特徴とする請求項１に記載の方法。

【請求項６】
ボリュームフィルタドライバが、前記１つまたは複数のデータ変更を受け取り、前記１つまたは複数のデータ変更を前記１つまたは複数のボリュームログファイルに適用することを特徴とする請求項１に記載の方法。

【請求項７】
前記第１および第２の時間インスタンスで整合性のある前記１つまたは複数のデータ変更は、アプリケーション整合性とファイルシステム整合性のうちの少なくとも一方を有することを特徴とする請求項１に記載の方法。

【請求項８】
前記プロダクションサーバにおけるボリュームデータの新しい更新を前記バックアップサーバに送信することをさらに備え、前記新しい更新は第３の時間インスタンスで整合性があり、前記第２と第３の時間インスタンス間で経過する時間は１時間未満の任意の時間に設定可能であることを特徴とする請求項１に記載の方法。
【請求項９】
１つまたは複数のファイルのコピーを求める要求を前記バックアップサーバに送信することをさらに備え、
前記バックアップサーバへの、１つまたは複数のファイルのコピーを求める前記要求は、前記１つまたは複数のファイルが前記第２と第３の時間インスタンスの一方で妥当であることを示すことを含む
ことを特徴とする請求項８に記載の方法。
【請求項１０】
前記バックアップサーバから復旧応答を受信することをさらに備え、前記復旧応答は、前記第２または第３の時間インスタンスでの、前記要求された１つまたは複数のファイルのデータのフルコピーを含むことを特徴とする請求項９に記載の方法。
【請求項１１】
前記１つまたは複数のボリュームの前記データの前記フルベースラインに対応する、データの前記コピーであって、前記第１の時間インスタンスで整合性がある、前記データの前記コピーを受信すること、
前記１つまたは複数のボリュームの前記１つまたは複数のデータ変更の前記コピーであって、前記第２の時間インスタンスで整合性がある、前記１つまたは複数のデータ変更の前記コピーを受信すること、
前記第２の時間インスタンスに従って妥当であるデータを求める復旧要求を受信すること、
１つまたは複数のバックアップサーバボリュームにおいて、前記第２の時間インスタンスでの前記要求されたデータであって、前記１つまたは複数のデータ変更の少なくとも一部を含む、前記要求されたデータを識別すること、および、
前記第２の時間インスタンスで妥当である前記要求されたデータを前記プロダクションサーバに送信すること
をさらに備えることを特徴とする請求項１に記載の方法。
【請求項１２】
前記１つまたは複数のボリュームの１つまたは複数のデータ変更の後続のコピーであって、後続の時間インスタンスで整合性がある、前記１つまたは複数のデータ変更の前記後続のコピーを受信すること、
前記後続の時間インスタンスに従って妥当であるデータを求める後続の復旧要求を受信したとき、前記フルベースラインのコピーの受信と、１つまたは複数のデータ変更の前記後続のコピーの受信と、の間に受信された、前記要求されたデータに対する変更の１つまたは複数のコピーのそれぞれを識別すること、および、
前記要求されたデータの前記フルベースラインのコピーを、前記要求されたデータに対する変更の前記識別された１つまたは複数のコピーと、結合すること
をさらに備えることを特徴とする請求項１１に記載の方法。
【請求項１３】
前記フルベースラインのコピーおよび前記１つまたは複数のデータ変更の前記コピーは、アプリケーション整合性またはファイルシステム整合性のうちの少なくとも一方が存在することを特徴とする請求項１２に記載の方法。
【請求項１４】
保護すべきデータを１つまたは複数のプロダクションサーバが１つまたは複数のバックアップサーバでバックアップするコンピュータ化された環境中の、プロダクションサーバにおける、コンピュータ実行可能命令が記憶されたコンピュータ記憶媒体であって、前記

コンピュータ実行可能命令は、実行されたとき、前記プロダクションサーバにおける１つまたは複数のプロセッサに、最近のデータを容易に前記バックアップサーバから復旧できるように、プロダクションサーバデータを実質上連続的かつアプリケーション整合性のある方式で複製する方法を実施させ、前記方法は、

第１の時間インスタンスで、１つまたは複数のボリュームのデータのコピーであって、前記１つまたは複数のボリュームの前記データのフルベースラインに対応する、前記コピーをプロダクションサーバから作成すること、

前記１つまたは複数のボリュームの、前記第１の時間インスタンスで整合性がある、前記データの前記コピーを前記プロダクションサーバからバックアップサーバに送信すること、

前記第１の時間インスタンスの後に、前記１つまたは複数のボリューム上の前記データに対する１つまたは複数の変更であって、前記１つまたは複数の変更のうちの少なくとも１つには、前記プロダクションサーバにおけるファイルパスが前記バックアップサーバにおける前記ファイルへのパスとは異なるように、前記プロダクションサーバでの、前記１つまたは複数のデータ変更のいずれかに対応するファイルのファイルパスに対する変更が含まれる、前記１つまたは複数の変更のそれぞれについての指示を、前記プロダクションサーバ上の揮発性メモリ内に記憶される１つまたは複数のビットマップの状態で記憶すること、

複製サイクルのイベントを識別したとき、前記１つまたは複数のビットマップを、前記プロダクションサーバの永続記憶装置に記憶された１つまたは複数のログファイルに保存することであって、前記１つまたは複数のデータ変更は第２の時間インスタンスで整合性がある、１つまたは複数のログファイルに保存すること、

前記１つまたは複数のビットマップを前記揮発性メモリから削除すること、

前記１つまたは複数のボリュームの前記１つまたは複数のデータ変更を識別するために、前記１つまたは複数のビットマップの前記指示を使用すること、

前記プロダクションサーバと前記バックアップサーバとにおける前記ファイルへの前記パスを相関させ、それにより前記ファイルに対する新しい変更を前記ファイルへの前記パスの変更と共に前記バックアップサーバに送信できるようにすることであって、前記パスを相関させることは、

前記プロダクションサーバでＵＳＮジャーナルを少なくとも１回目に走査してファイルパスの前記変更をキャッシュすること、

前記プロダクションサーバで前記ＵＳＮジャーナルを少なくとも２回目に走査して前記最初のファイルパスを識別すること、および、

前記１回目および２回目の走査に基づいて前記バックアップサーバにおける前記ファイルへの調整済みパスを計算すること

を備える、前記バックアップサーバに送信できるようにすること、ならびに

前記１つまたは複数のボリュームの前記１つまたは複数のデータ変更のコピーを前記バックアップサーバに送信することを備え、前記バックアップサーバが、第１の時間インスタンスおよび第２の時間インスタンスで妥当である前記１つまたは複数のボリュームのデータのコピーを有すること

を備えることを特徴とするコンピュータ記憶媒体。

【発明の詳細な説明】

【技術分野】

【０００１】

本発明は、電子ファイルのバックアップに関し、より詳細には、電子ファイルを定期的にバックアップし、それにより、元々作成されたファイルの信頼性ある復元を必要時に作成することに関する。

【背景技術】

【０００２】

コンピュータ化されたシステムの人気が高まるのに伴い、システムに関連するユーザや

アプリケーションによって作成された電子ファイルや他の通信を記憶してバックアップする必要も増大してきた。一般に、コンピュータ・システムおよび関係するデバイスは、作業環境でワード・プロセッシング文書を作成する一般的な場合、ならびに、より複雑なデータベース目的に使用されるファイルを作成する場合などの、様々な理由でファイルを作成する。加えて、これらの文書の多くは、価値のある作業生産物（work product）や保護すべき機密情報を含む可能性がある。したがって、組織が、なぜ、電子ファイルを定期的にバックアップし、それにより、元々作成されたファイルの信頼性ある復元を必要時に作成したいと望むかには、様々な理由があることを理解するであろう。

【０００３】

多くの従来バックアップ・システムによって提供される、いくつかの利便性にもかかわらず、多くの従来システムによって使用される機構はしばしば、最適ではなく効率が劣る。例えば、アプリケーション整合（application-consistent）バックアップを作成する能力は、いくつかのバックアップ・システムの重要な構成要素とすることができる。アプリケーション整合（ならびにファイル・システム整合）バックアップは基本的に、特定時点のファイル状態に整合性を有する、１組のバックアップされたデータである。例えば、バックアップ管理者が、所与のプロダクション・サーバ・ボリューム（production server volume）上にあるすべてのデータを、このデータが書込みまたは更新の途中である可能性があるにもかかわらずコピーした場合、対応するバックアップ・コピーのファイル状態は、単一時点で必ずしも整合していない可能性がある。したがって、アプリケーション整合バックアップを作成することは、一般に、ファイル状態を確認する（reconciling）追加の労力を必要とする。

【０００４】

しかし、アプリケーション整合（application-consistent）データ・バックアップの作成に関連する多くの困難があり得ることを理解することができる。例えば、バックアップを作成するための従来の機構は一般に、メールまたはデータベース・アプリケーション・プログラムのバックアップ・コンポーネントなどの、アプリケーション・プログラムが、１つまたは複数のバックアップおよび／または復元アプリケーション・プログラミング・インターフェース（ＡＰＩ）を呼び出すことを必要とする。具体的には、バックアップ・コンポーネントは、ＡＰＩに、プロダクション・サーバ上の特定の指定ディスク・データへの書込みをフリーズするよう伝え、次いで、データのバックアップ・コピー（すなわち「複製」）を作成することができる。残念ながら、一般に、プロダクション・サーバにおいてバックアップ・コンポーネントがそれらのデータをバックアップＡＰＩに対して記述するための単純な方法はない。この困難をさらに複雑にするのは、バックアップ・プロセス中に参照されることが必要となる可能性があるバックアップＡＰＩが、時として多数存在する可能性があることである。

【０００５】

加えて、あるデータを作成した特定のアプリケーションがバックアップ・サービスを要求しているということは、プロダクション・サーバ・データのすべてよりも少ないデータが、幾つかの所与の時点でバックアップされる場合があることを、多くの場合、意味する。例えば、従来のバックアップ機構はしばしば、バックアップされるデータに関してアプリケーション特有である。このようなアプリケーション特有バックアップ手法は、多くの場合に、バックアップ・プロセス中に所与のアプリケーションの複数のインスタンスを実行することを伴う。しかし、所与のアプリケーションの複数のインスタンスを実行することは、いくつかの理由で、それがコストやリソース消費の観点からであっても、非効率的である可能性があることを理解するであろう。

【０００６】

さらに、アプリケーションを使用してバックアップ・サービスを提供することすら、いくらか非効果的である可能性がある。というのは、アプリケーション固有バックアップは、一般に、多くのリソース消費を伴わずにアプリケーション・データのポイント・イン・タイム（point-in-time）コピーを提供しないからである。このことは、アプリケーショ

ンが、プロダクション・サーバとバックアップ・サーバとの両方に負荷をかけ過ぎることなく頻繁にアプリケーション整合ポイント・イン・タイム・バックアップを提供すること（したがって高い細分性の（highly granular）復旧時点（recovery points）を提供すること）ができない場合があることを意味し得る。したがって、特定のアプリケーションによって実施される従来のバックアップは、通常、ほんの数分だけ古いポイント・イン・タイム・バックアップの「ホット・スタンバイ（hot standby）」を提供することができない。

【０００７】

この、きめ細かく構成する能力が一般に欠如していることは、バックアップ・システムにおける様々な他の問題に広がる可能性がある。例えば、従来のバックアップ・システムは、ファイルのタイプ、特定のフォルダ、または特定ボリューム上でのフォルダ位置に対して構成するのが難しいことがある。したがって、実際に修正されたファイル部分のみをバックアップするのでなく、１つまたは複数のボリューム全体そのもの、あるいはファイル全体そのもの、よりもよりきめ細かく、従来のバックアップ・システムにプロダクション・サーバ・データをバックアップさせる、ことに関連する困難がある可能性がある。これらおよび他の類似の問題はしばしば、プロダクション・サーバとバックアップ・サーバとが必要以上のデータを相互間でコピーおよび転送するように構成されることを意味し、これは当然ながらシステム性能およびネットワーク帯域幅に影響する可能性がある。具体的には、プロダクション・サーバは、変更されていないファイルデータ、ならびにわずかな部分しか変更されていないファイル全体を、コピーおよび転送している場合がある。このため、プロダクション・サーバ・データをバックアップするのに、必要以上の多くの記憶容量をバックアップ・サーバが充てる必要がある場合がある。

【発明の開示】

【発明が解決しようとする課題】

【０００８】

したがって、前述の各要因（またはこれらの組合せ）が目標復旧時点（Recovery Point Objective；RPO）に悪影響を及ぼす可能性があることを理解するであろう。ＲＰＯは一般に、組織が災害（disaster）後に業務を再開するためにどのくらい時間を遡ってデータを復旧する必要があるかを指す。前述の要因はまた、目標復旧時間（Recovery Time Objective;RTO）にも悪影響を及ぼす可能性があり、ＲＴＯは一般に、災害後、業務の再開に必要なデータを復旧できるまでにどのくらいの時間が経過することになるかを指す。すなわち、従来のバックアップ・システムは一般に、特にシステム・リソースに過度の負担をかけずに、比較的速い時間量で、比較的高い復旧時点（recovery points）を提供するための備えができていない。

【０００９】

したがって、現在のバックアップ・システムは、対処される可能性があるいくつかの困難に直面している。

【課題を解決するための手段】

【００１０】

本発明の実装形態は、バックアップ・システムにおける目標復旧時点および目標復旧時間を最適化するように少なくとも部分的に構成されたシステム、方法、およびコンピュータ・プログラム製品により、当技術分野における１つまたは複数の問題を解決する。例えば、少なくとも１つの実装形態では、ボリューム・フィルタ・ドライバを使用してプロダクション・サーバ・ボリュームに対する変更を監視することによって、プロダクション・サーバにおけるリソース節約を達成することができる。加えて、主として、最後の複製サイクル以降はインクリメンタルな変更（例えば変更のバイトまたはバイト範囲）のみをバックアップ・サーバに転送することによって、ネットワーク帯域幅およびバックアップ・サーバ・リソースを効率的に使用することができる。本明細書でより完全に理解されるであろうが、このような最適化は、プロダクション・サーバ・リソースおよびバックアップ・サーバ・リソース、ならびに／あるいはネットワーク帯域幅関連の著しい浪費なしに、

実質上連続的な（またはほぼ連続的な）方式でプロダクション・サーバ・データをバックアップする能力を提供することができる。

【００１１】

例えば、プロダクション・サーバ・データを実質上連続的かつ整合性のある方式で複製する、プロダクション・サーバの視点からの方法は、プロダクション・サーバの１つまたは複数のボリュームからのボリューム・データのコピーをバックアップ・サーバに送信することを含むことができる。このような場合、ボリューム（１つまたは複数）の送信データ・コピーは一般に、第１の時間インスタンスで整合性がある（すなわちアプリケーション整合性またはファイル・システム整合性がある）ことになる。加えて、この方法は、１つまたは複数のボリューム・ログファイルを介してボリュームに対する１つまたは複数の変更を識別することを含むことができる。この方法はさらに、複製サイクル・イベントを識別したとき、１つまたは複数のデータ変更を１つまたは複数のボリューム・ログファイルに保存することを含むことができる。一般に、１つまたは複数のデータ変更もまた、第２の（すなわち後続の）時間インスタンスで整合性があることになる。さらに、この方法は、１つまたは複数の変更のコピーをバックアップ・サーバに送信することを含むことができる。したがって、バックアップ・サーバは、１つまたは複数のボリュームのデータのコピーを有することになり、このデータは、第１の時間インスタンスおよび第２の時間インスタンスで妥当（valid）である。

【００１２】

対照的に、プロダクション・サーバ・データを実質上連続的かつ整合性のある方式で複製する、バックアップ・サーバから見た方法は、プロダクション・サーバから１つまたは複数のボリューム・バックアップを受信することを含むことができる。このような場合、１つまたは複数のボリューム・バックアップは、最初の時間インスタンスで整合性がある。この方法はまた、１つまたは複数のアプリケーション整合バックアップ更新を受信することを含むことができ、これらのバックアップ更新の少なくとも１つは、後続の時間インスタンスでの、１つまたは複数のボリューム・バックアップの少なくとも１つに対する整合性のある更新である。加えて、この方法は、後続の時間インスタンスに従って妥当であるデータを求める復旧要求を受信することを含むことができる。

【００１３】

さらに、この方法はまた、１つまたは複数のバックアップ・サーバ・ボリュームにおいて、後続の時間インスタンスでの要求されたデータを識別することを含むことができる。このような場合、要求されたデータは、少なくとも１つのアプリケーション整合バックアップ更新の少なくとも一部を含む。加えて、この方法は、後続の時間インスタンスで妥当である要求されたデータをプロダクション・サーバに送信することを含むことができる。

【００１４】

この概要は、以下の詳細な記述でさらに述べる複数の概念の精選を、単純化した形で紹介するために提供する。この概要は、特許請求する主題の鍵となる特徴または本質的な特徴を識別することを意図したものではなく、特許請求する主題の範囲を決定する際の助けとして使用されることを意図してはいない。

【００１５】

本発明の例示的な実装形態の追加の特徴および利点は、以下の説明に記載され、部分的には、これらはこの説明から自明であろうし、あるいはこのような例示的な実装形態の実践によって知ることができる。このような実装形態の特徴および利点は、添付の特許請求の範囲に特に示す手段および組合せによって理解すること、および獲得することができる。これらのおよび他の特徴は、後続の記述および添付の特許請求の範囲からより完全に明らかになるであろうし、あるいは後述するこのような例示的な実装形態の実践によって知ることができる。

【００１６】

本発明の前述のおよび他の利点および特徴を得ることのできる方式について説明するために、上に簡単に説明した本発明のより具体的な説明を、添付の図面に示すその特定の実

施形態を参照して提供する。これらの図面は本発明の例示的な実施形態のみを表すものであり、したがってその範囲を限定するものと考えるべきではないことを理解した上で、添付の図面を使用して本発明をさらに具体的かつ詳細に記述および説明する。

【発明を実施するための最良の形態】

【００１７】

本発明は、バックアップ・システムにおける目標復旧時点（recovery point objectives）および目標復旧時間（recovery time objectives）を最適化するように、少なくとも部分的に構成されたシステム、方法、およびコンピュータ・プログラム製品に及ぶ。例えば、少なくとも１つの実装形態では、ボリューム・フィルタ・ドライバを使用してプロダクション・サーバ・ボリュームに対する変更を監視することによって、プロダクション・サーバにおけるリソース節約を達成することができる。加えて、主として、最後の複製サイクル以降はインクリメンタルな変更（例えば変更のバイトまたはバイト範囲）のみをバックアップ・サーバに転送することによって、ネットワーク帯域幅およびバックアップ・サーバ・リソースを効率的に使用することができる。本明細書でより完全に理解されるであろうが、このような最適化は、プロダクション・サーバ・リソースおよびバックアップ・サーバ・リソース、ならびに／あるいはネットワーク帯域幅関連の著しい浪費をせずに、実質上連続的な（またはほぼ連続的な）方式でプロダクション・サーバ・データをバックアップする能力を提供することができる。

【００１８】

後続の明細書および特許請求の範囲からより完全に理解されるであろうが、本発明の実装形態は、プロダクション・サーバ・データの「フルスナップ・ショット（full snapshot）」を使用してバックアップ・サーバをリフレッシュすることによって、幅広い「目標復旧時間」を満たすことができる。加えて、本発明の実装形態は、プロダクション・サーバで実装することのできるボリューム・フィルタ・ドライバを含む。本明細書でより完全に理解されるであろうが、ボリューム・フィルタ・ドライバは、プロダクション・サーバ・ボリューム（１つまたは複数）上のバイト（および／またはバイト・ブロック）に対する変更を監視するように構成することができる。次いでプロダクション・サーバは、これらの変更されたバイト（またはバイト・ブロック）のみをバックアップ・サーバに送信することによって、スナップ・ショット（またはバックアップ・コピー）全体を送信するように構成することができる。したがって、ボリューム・フィルタ・ドライバを使用することで、プロダクション・サーバ・データのフルスナップ・ショットをバックアップ・サーバに移動するときに通常なら消費されるかもしれないリソースに対する負担を緩和することができる。

【００１９】

さらに、またこれらおよび他の特徴の結果として、プロダクション・サーバは、互いに時間間隔の短いいくつかの「整合性のある」（すなわちアプリケーション整合性および／またはファイル・システム整合性のある）スナップ・ショットの実質上連続的なバックアップ（すなわち最初のバックアップおよび後続のシャドウ・コピー）を、バックアップ・サーバに提供することができる。加えて、バックアップの各更新はアプリケーション整合性（および／またはファイル・システム整合性）があるか、あるいは特定の時間インスタンスで妥当であるので、各更新の間の差もまたアプリケーション整合性があることになる。したがって、本発明の実装形態は、非常に高い細分性（fairly high granularity）（例えばほんの数分だけ古い）で、また通常なら必要かもしれない負担よりもずっと少ない負担で、幅広いアプリケーション整合データ（例えばファイル・レベルから、データベース・レベル、さらにはプロダクション・サーバ全体レベル）を復旧する能力をユーザに提供する。

【００２０】

一般に、本発明の実装形態による連続的かつ整合性のあるバックアップ・サービスを実装する方式には、様々なものがある。少なくとも１つの非常に基本的な意味では、整合性のあるバックアップを作成することは、１つまたは複数のボリューム（例えば１７５）の

ベースライン・コピー（例えば１４５）を作成すること、および、次いでこのベースライン・コピーに、１つまたは複数のボリュームのインクリメンタルかつ整合性のある更新（例えば１５０、１５５）を補足することを含む。例えば、図１Ａに、プロダクション・サーバ１０５が、選択されたボリューム（１つまたは複数）１７５上のデータの少なくとも１つのベースライン・バックアップ・コピー（例えば１４５）を作成するのを示す。単にベースライン・コピー１４５を作成するのに加えて、システム１００におけるバックアップ管理者は、任意の数またはタイプの機構を使用してコピー１４５を整合性のあるものにすることができる。

【００２１】

一実装形態では、バックアップ管理者は、複製プロセスをガイドするためにバックアップ・サーバ１１０および／またはプロダクション・サーバ１０５にインストールされた、複製エージェント（replica agent）（図示せず）を使用することができる。複製サイクルの間、例えば複製エージェントは、プロダクション・サーバ１０５上のいずれか１つまたは複数の適切なアプリケーション・ライタに、単一時点に対する、いずれか１つまたは複数のボリューム上（例えばデータベースがいくつかのボリュームにわたる場合があり、いくつかのアプリケーションが同じボリュームを異なるスケジュールで使用する可能性がある）への書込アクティビティを少しの間だけ保留するよう命令するように構成することができる（ファイル共有バックアップの場合は、アプリケーション・ライタがまったく関係しないことすらある）。これは、複製エージェントが、１つまたは複数のボリュームの単一のポイント・イン・タイム・バックアップ（すなわち「シャドウコピー」）を作成することを可能にする。

【００２２】

複製エージェントはまた、各アプリケーション・ライタに、それらの当該データに関するいくつかの機能を実施させて、それによりすべてのデータおよびメタ・データが複製サイクルの時点で整合性を有することを確実にする命令を、提供することもできる。アプリケーション・ライタまたは、それらに関連付けられた対応するプラグインを有しない可能性のある、より単純なアプリケーションについては、複製エージェントは、単に、複製サイクルの間にフリーズまたはシャットダウンするようこれらのアプリケーションに命令するように構成されてもよい。整合性のあるバックアップを作成するための前述のエージェント、コンポーネント、および機能は、ＭＩＣＲＯＳＯＦＴ環境における少なくとも１つの実装形態で、例えば、ボリューム・シャドウ・コピー・サービス（VSS、Volume Shadow Copy Service）を使用して、提供することができる。

【００２３】

いずれの場合でも、特定の１つのボリューム（または複数のボリューム）への、また特定の時間インスタンスでの、対象の書込みをフリーズすると、次いでプロダクション・サーバ１０５は、対象のボリューム（１つまたは複数）（あるいは別法として、選択されたフォルダ、ファイル、またはファイル・タイプのみ）のコピーを作成して送信することができる。例えば、図１Ａは、プロダクション・サーバ１０５がこの最初のベースライン・コピー１４５をバックアップ・サーバ１１０に提供することができることを示す。一般に、プロダクション・サーバ１０５は、多くの方法でベースライン・コピー１４５を提供することができる。一実装形態では、例えば、プロダクション・サーバ１０５は単にネットワーク接続を介してコピー１４５を送信する。ネットワーク帯域幅がより限られているかもしれない場合など、他の実装形態では、バックアップ管理者が、テープ（または別の中間記憶ノード（図示せず））にボリューム・コピーを転送し、後でこのテープをバックアップ・サーバ１１０に接続することができる。しかしながら、実施されると、プロダクション・サーバ１０５は、同じ複製サイクルを共有する対象のボリューム、フォルダ、ファイル、またはファイル・セットに対する、すべてのデータの、整合性ある少なくとも１つのベースライン・コピー（すなわち時間「t_0」で妥当）をバックアップ・サーバ１１０に提供する。

【００２４】

１つまたは複数のプロダクション・サーバ・ボリュームの、１つまたは複数のベースライン・コピーを提供した後、バックアップ・サーバ１１０は、ベースラインバックアップ（１つまたは複数）への更新を受信し続けることができる。例えば、バックアップ・サーバ１１０は、約５～１０分、１０～１５分、１５～３０分などにわたる様々な構成可能な複製スケジュールで、プロダクション・サーバ１０５をバックアップし続けることができる。一般に、バックアップ管理者が複製サイクルを構成するレベルは、特定の「復旧時点」にアクセスできる細分性のレベルとなる。

【００２５】

前に論じたように、普通、ポイント・インタイム・バックアップのアクセス可能性（accessibility）の細分性レベルが比較的高いと、バックアップ・システムによってはリソースに対する負荷が法外となる可能性がある。このため、「復旧にかかる時間」を不必要に損なわずに（例えば大きなオーバーヘッドを被らずに）前述の「復旧時点」の細分性レベルを生み出すために、本発明の実装形態は、いくつかの重要なコンポーネントおよび機能を提供することができる。

【００２６】

後でより完全に論じる一実装形態では、例えば、ボリューム・フィルタ・ドライバ１１５は、プロダクション・サーバ１０５における１つまたは複数のボリュームのいずれか（例えば１７５）に対する反復的な変更を、メモリ内ビットマップを使用して、あるいはディスク上のボリューム・ログファイル中で特定の変更済みバイト（またはバイト・ブロック）に印を付けることによって、監視するのに使用することができる。一般に、ボリューム・フィルタ・ドライバ（例えば１１５）は、どのようにハードウェアまたはソフトウェア・ベースの「スナップ・ショット」がプロダクション・サーバ１０５上で実装されるかということから独立していることになる。しかし、ボリューム・フィルタ・ドライバ１１５は必ずしも必要とされないこと、および、本明細書にも論じるように他のコンポーネントによって同様の機能を実施することもできることを理解するであろう。追加のまたは代替の実装形態では、プロダクション・サーバ１０５はまた、従来のシャドウ・コピー監視機構および／または更新シーケンス番号（Update Sequence Number）ジャーナル（すなわち「ＵＳＮジャーナル」１４０）などの使用を通して、ボリュームに対する変更（例えばファイル１２０、１２５、１３０などに対する変更）を監視することもできる。例えば、特にＭＩＣＲＯＳＯＦＴオペレーティング環境に関しては、このようなコンポーネントは、ボリューム・シャドウ・コピー・サービス（ＶＳＳ）とＵＳＮジャーナルとを組み合わせて使用することによって提供される。

【００２７】

一般に、ボリューム・ログファイル（例えば１３５）は、ボリュームへの書込みごとの、特定の複製サイクル中のボリュームに対するすべての変更（例えばボリュームオフセット、データ長変更）、および／または、変更のメモリ内ビットマップ（in-memory bitmap）を含むことができる。特定の複製サイクルが発生したとき、既存のボリューム・ログファイル（例えば１３５）はフリーズされ、新しいボリューム・ログファイル（図示せず）を生成して、次の複製サイクルに対する変更を収集することができる。一実装形態では、ボリューム・レベルの変更は、どんな追加の相関情報（correlating information）もなしに、直接、バックアップ・サーバ１１０に送信することができる。次いで、バックアップ・サーバ１１０に送信される対応する更新は、「バイトｎ」（または「バイト・ブロックｎがｎ＋１に変更されたもの」）としてその複製に適用することができる。追加のまたは代替の実装形態では、プロダクション・サーバ１０５におけるボリューム・データは、ＵＳＮジャーナル（または関連コンポーネント）１４０の情報と相関させることもできる。

【００２８】

具体的には、ＵＳＮジャーナル（例えば１４０）は、ファイル・システム中のファイル名データに関するこのようなアクティビティのタイム・スタンプ付き情報を含む。ＵＳＮジャーナル１４０と同様または同一のコンポーネントは、変更フィルタまたは変更ジャー

ナルなどとも呼ばれる。したがって、本明細書で具体的に「ＵＳＮジャーナル１４０」に言及するのは、主として便宜上である。いずれの場合でも、またファイル・システム・アクティビティに関して、プロダクション・サーバ１０５は、ボリューム・ログファイル（例えば１３５）データと変更ジャーナル・データ（例えばＵＳＮジャーナル１４０）とを組み合わせて、ボリューム１７５への様々な書込みの、時間、アクティビティのタイプ、およびファイル名などを相関させることができる。

【００２９】

これらの方向に沿って、ＵＳＮジャーナル１４０は、ボリューム・ログファイル１３５と共に使用して、特定のバイトまたは「バイト・ブロック」の変更（ならびに変更に対応するファイル）のアドレスなどを相関させることもできる。具体的には、ボリューム上の各ファイルは、アドレス指定可能なバイトの開集合（open set）、ならびにアドレス指定可能な固定長バイト・ブロックの開集合、と考えることができる。場合によっては、バイト・ブロック（個々のバイトではなく）を監視および転送することの方が、変更を監視および転送するための、ならびにバックアップ目的でどのくらいの空間が必要とされるかを決定するための、より効率的な方法である可能性がある。具体的には、これは一部には、バイト・ブロックが、通常ファイル全体よりもいくぶん小さいが、単一バイトよりも大きい、細分性レベルを表すからである。したがって、図１Ａに、プロダクション・サーバ１０５が、保護されるべきそのファイルに対する様々なバイトまたはバイト・ブロックの変更をログするのを示す。

【００３０】

例えば、図１Ａは、プロダクション・サーバ１０５が、最後の複製サイクル（例えば最後の５、１０、１５、２０、２５、または３０分など）以降にファイル１２０のバイト（または「バイト・ブロック」）１２１、１２２、および１２３が変化した（例えば１２０が新しいファイルである）ことをログすることを示す。同様に、最後の複製サイクル以降、ファイル１２５は、バイト（またはバイトブロック）１２７、１２８、および１２９を含むが、バイト１２８および１２９のみが変化しており、ファイル１３０は、バイト１３１、１３２、および１３３を含むが、バイト１３３のみが変化している。一般に、プロダクション・サーバ１０５は、これらの変更済みバイトを、ボリューム、フォルダ、または関連ファイルの読取専用シャドウ・コピーにログすることができる。前述のように、プロダクション・サーバ１０５はまた、ボリューム・ログファイルについてのこれらの変更済みバイトをメモリ内ビットマップとして（例えばボリューム上のデータ・ブロックにつき１ビットを使用して）記憶することもでき、これらのメモリ内ビットマップは、後で複製中に物理ディスクに渡される。どのようにログまたは監視されようと、また適切な時間（すなわち次の複製サイクル）に、次いでプロダクション・サーバ１０５は、これらのファイル変更（すなわち１２１、１２２、１２３、１２８、１２９、１３３など）のみを、バックアップ・サーバ１１０に送信されるよう準備することができる。

【００３１】

この特定の例では、変更（すなわち１２１、１２２、１２３、１２８、１２９、１３３など）はそれぞれ、一番最近の時点（すなわち「t_1」）で妥当であり、したがって整合性がある（すなわちアプリケーション整合性またはファイル・システム整合性がある）。とりわけ、これらのバイト（またはバイト・ブロック）は、２つの整合するポイント・インタイム・バックアップ間の差を表すので、バックアップ・サーバ１１０でこれらのバイト（またはバイト・ブロック）を適用してもやはり整合性があることになる。これらのデータ変更を識別した後、プロダクション・サーバ１０５は、これらの変更を、更新１５０（時間t_1での）としてバックアップ・サーバ１１０に送信することができる。同様に、プロダクション・サーバ１０５は、次の複製サイクルでの次のインクリメンタルな更新で、インクリメンタルなデータ変更（すなわち１つまたは複数ボリュームについてログされたデータ変更）の次の各セットを識別して送信することができる。例えば、図１Ａにはまた、プロダクション・サーバ１０５が更新１５５（時間t_2での）を準備してバックアップ・サーバ１１０に送信し、以下同様にすることを示す。

【００３２】
プロダクション・サーバ１０５が変更済みデータを読み取ってメッセージ１５０を作成した時点から、そのボリューム・データに追加の変更がある場合があり、この変更は、読み取られているデータを、他の場合には整合（すなわち時間「t_1」で整合性があり、後続の時間で妥当である）の状態にしない可能性があることを理解するであろう。したがって、ベースライン・コピー１４５に関して前に論じたように、ボリューム・シャドウ・コピー・サービス（または他のＶＳＳのような機構）は、少なくとも１つの実装形態で使用して、フリーズされた特定の時間インスタンスのみのデータを読み取り、その後のどんな変更も読み取らないことができる。これは、スナップ・ショット更新１５０（ならびに１５５など）が、スナップ・ショット（バックアップ更新または更新とも呼ばれる）操作が開始した指定の時間インスタンスまで整合の状態にあるようにするのを助けることができる。

【００３３】
受信時、バックアップ・サーバ１１０は、各バックアップおよび対応する更新（１つまたは複数）を特定の複製ボリュームに記憶することができる。一実装形態では、バックアップ・サーバ１１０は、そのバックアップと更新とを、同じ記憶媒体の同じボリューム・アロケーションに記憶する。他の実装形態では、バックアップ・サーバ１１０（および／あるいは追加のバックアップ・サーバまたは記憶ノード）は、バックアップと対応する更新とを、バックアップ管理者によって望まれる、別々のボリュームに、さらには別々の記憶媒体に記憶することができる。

【００３４】
場合によっては、またプロダクション・サーバ１０５の実質上連続的かつ反復的なバックアップの性質により、バックアップ管理者は、データのいくつかの状況（aspects）をプロダクション・サーバ１０５のデータと同期させる必要がある場合がある。例えば、ネットワーク停止（network outage）、およびログ・オーバーフロー（例えばＵＳＮジャーナルラップ（wrap）など）に起因する場合など、複製サイクル中のおよび／またはいくつかの複製サイクルにわたる障害の場合がある。したがって、一実装形態では、バックアップ管理者は、プロダクション・サーバ１０５の新しいベースライン・フル・スナップショット（例えば１４５と同様のもの）を作成することによって、妥当性検査または訂正を実施することができる。次いでバックアップ管理者は、プロダクション・サーバ１０５上のデータのスナップ・ショットとバックアップ・サーバ１１０上のデータとの間でチェック・サム比較（または他の妥当性検査）を実施することができる（例えばプロダクション・サーバ１０５を介して）。次いで、バックアップ・サーバ１１０上に誤ったデータ（errant data）があれば、必要ならこれを修復することができる。

【００３５】
例えば、特にＷＩＮＤＯＷＳ（登録商標）オペレーティング・コンポーネントに関しては、このチェック・サムは、少なくとも１つの実装形態では、ＷＩＮＤＯＷＳ（登録商標）　ＳＥＲＶＥＲ２００３で使用されるリモート差分圧縮（RDC、Remote Differential Compression）を用いて実施することができる。場合によっては、ワイドエリアネットワーク（ＷＡＮ）環境ではＲＤＣタイプの機構の使用が好ましいことがある。別の実装形態では、ローカルエリアネットワーク（ＬＡＮ）において好ましいが、バックアップ管理者は、スナップ・ショット中の各ファイルを「チャンク（chunks）」（例えばバイト・ブロック）からなる複数のセットに分割し、次いで各チャンク毎にチェック・サムを計算することができる。

【００３６】
いずれの場合でも、一部には、本発明の実装形態によって提供される複製の細分性のレベル（level of replication granularity）により、ほんの数分だけ古い特定バージョンの（例えばパーソナル・コンピュータの少し前に起こったクラッシュから必要とされる）ファイル（または他のデータ表現）をユーザが要求する場合、そのユーザは、ファイルのこの特定バージョンを求める要求をバックアップ・サーバ１１０に送信することができる

。例えば、ユーザは、５分前（例えば「t_0」、または更新１２１、１２２、１２３の前）の時点で妥当であったファイル１２０の特定コピーを要求することができる。同様に、管理者は、１つまたは複数のボリューム１７５の全体の再現（entire reproduction）を要求する（図示せず）こともできる。

【００３７】

要求の受信時に、また要求の性質に応じて、次いでバックアップ・サーバ１１０は、それぞれに見合った、要求されたデータを見つけることができる。例えば、基本的なファイル・システム・データに関しては、ボリューム１７５の各更新は、要求されたデータの完全なコピーを収容する可能性がある。したがって、バックアップ・サーバ１１０は、ユーザによって要求された時間を識別し、この時間に対応する更新内のデータを識別し、次いでこのデータのコピーを、元のユーザに提供する（例えば復旧メッセージ１６０）だけで済む場合がある。

【００３８】

メールまたは他のタイプのデータベース・アプリケーション・データなど、他の場合では、バックアップ・サーバ１１０によって受信されたインクリメンタルな各更新（例えば１５０、１５５）は、要求されたデータのインクリメンタルな更新のみを含む場合がある。したがって、バックアップ・サーバ１１０は、インクリメンタルな各更新を、要求された復旧時点から最後のベースライン・フルまで遡って、再生する（play back）ように構成することができる。次いでバックアップ・サーバ１１０は、要求中で指定された時間に達するまで、再生中に識別された、要求されたデータ（例えば１４５、１５０、１５５、またはt_{0-n}）を結合することができる。元のバックアップおよび対応する更新からの、関連するすべてのデータが結合されて準備されると、次いでバックアップ・サーバ１１０は復旧応答（例えば１６０）を送信することができ、この復旧応答は、要求された時間に従って妥当である。例えば、図１Ａに、バックアップ・サーバ１１０が応答１６０を送信するのを示すが、この応答１６０は、復旧されたデータが時間「t_1」で妥当であることを示す。

【００３９】

前述の実装形態では、バックアップ・サーバ１１０はしたがって、インクリメンタルな更新を再生するためにアプリケーション・サポートを必要とする場合がある。別の実装形態では、ベースライン・フル・コピー、および、ベースライン・フルと要求された時点との間の対応するインクリメンタルな更新があればそれらの更新を、単にプロダクション・サーバ１０５に逆にコピーすることができる。次いで、プロダクション・サーバ１０５における対応するアプリケーション・ライタ（例えばシャドウ・コピー・サービス・フレームワーク内のアプリケーション・ライタ）が、フル・バックアップに関するログを再生することができる。

【００４０】

一般に、特定データの要求と、対応する応答との間で経過する時間は、少なくとも２つの部分の関数とすることができる。

１．バックアップ・サーバ１１０からプロダクション・サーバ１０５にデータを転送するための時間

２．バックアップ・サーバ１１０が（例えば関連するバックアップ・エージェントを介して）復旧（recovery；リカバリ）を完了するための時間

【００４１】

当然、ターゲットからソースにデータを転送するための時間は一般に、利用可能なネットワーク帯域幅、ならびに、バックアップ・サーバ１１０およびプロダクション・サーバ１０５におけるディスク速度およびリソース使用の関数である。対照的に、特定のリカバリ（recovery）を作成するための時間は通常、プロダクション・サーバ・データのフル・コピーを所与のベースラインから復旧するのに必要とされる時間、ならびに、特定の時点を復旧するためにベースラインから累積した累積更新（例えば「t_{n-1}」）を識別して再生するのに必要とされる時間の関数である。したがって、定期的なベースライン・フル

・コピー（例えば１４５）を作成することなどによって、バックアップ・サーバ１１０（またはプロダクションサーバ１０５）がいずれか所与の復旧要求に対して再生しなければならない更新の量を制限することにより、復旧時間を大きく向上させることができることを理解するであろう。

【００４２】

前述のように、バックアップ・サーバ１１０が再生する必要がある可能性のあるインクリメンタルな更新の量を制限する方法の１つは、新しい「フル」ベースライン・スナップショットを定期的に作成することを含むことができる。新しいフルスナップ・ショットを作成してバックアップ・サーバ１１０に送信することは、場合によってはリソース・コスト（例えば必要とされるネットワーク帯域幅、プロダクション・サーバ１０５のリソース、およびバックアップ・サーバ１１０のディスク空間の量）がかかる可能性があるので、本発明の実装形態はまた、「インテリジェント・フル・スナップショット」の作成を可能にする。これらのインテリジェント・フル・スナップショットは、実質的に、所定時点のベースライン境界設定（baseline demarcation）である。例えば、２週間ごとなどの所定期間ごとに、バックアップ・サーバ１１０は、２週間分のインクリメンタルな更新（例えば１５０、１５５など）を、データの最後のベースライン・コピー（例えば１４５またはより新しいもの）と共にまとめ、それによりプロダクション・サーバ１０５のデータの本質的に新しい「t_0」コピーを作成することができる。

【００４３】

これらのインクリメンタルな更新のそれぞれを効率的にまとめるために、バックアップ・サーバ１１０は、最後のフルスナップ・ショット以降の、プロダクション・サーバ１０５のボリュームへのすべての書込みを監視するように構成することができる。少なくとも１つの実装形態では、例えば、バックアップ・サーバ１１０は、プロダクション・サーバ１０５でボリューム・フィルタ・ドライバ１１５を実装して、ボリューム（すなわち１つまたは複数のボリューム）に対する変更を監視し、各複製サイクル中にこれらの書込みをプロダクション・サーバ１０５のメモリ１７０に記憶する。例えば、図１Ａ（図１Ｂも参照されたい）は、プロダクション・サーバ１０５上でボリューム・フィルタ・ドライバ１１５がボリューム１７５とメモリ１７０との間をインターフェースするのを示す。図１Ｂでより完全に理解されるであろうが、ボリューム・データに変更が加えられる度に、ボリューム・フィルタ・ドライバ１１５は、この変更（または変更のセット）をボリューム・ログファイル１３５に記録することができる。少なくとも１つの実装形態では、これらの変更は、１つまたは複数のボリュームそれぞれにつき、メモリ内ビットマップ（例えば１１７ａ）としてシステム・メモリ１７０に記録される。プロダクション・サーバ１０５が特定の複製サイクルに向けて準備ができたとき、次いでボリューム・フィルタ・ドライバ１１５は、すべてのメモリ内ビットマップをボリューム１７５に渡すことができ、次いでプロダクション・サーバ１０５は、対応するデータをバックアップ・サーバ１１０に送信することができる。

【００４４】

例えば、図１Ｂに、メモリ１７０を使用して様々なメモリ内ビットマップに対応するスナップショット・データが収集されているのを示す。具体的には、図１Ｂには、ボリューム・フィルタ・ドライバ１１５がいくつかのファイル変更（すなわちファイル変更１２１、１２２、１２３など）を識別し、その後これらの変更を、対応するメモリ内ビットマップ１９３、１９５などとしてメモリ・アロケーション１９０ａに記憶するのを示す。この特定の例では、ボリューム・フィルタ・ドライバ１１５は、最後の複製サイクル（すなわちスナップ・ショット１８５ａ、「t_2」）以降の、対応する１つまたは複数のボリュームに対するすべての変更を記憶し、したがって、各ビットマップ１９３、１９５などは、対応するスナップ・ショット（すなわち１９０ａ、スナップ・ショット「t_3」）における一番最近の時間インスタンスで妥当である。

【００４５】

複製エージェントから受信された命令などにより複製サイクルがトリガされたとき、ボ

リューム・フィルタ・ドライバ１１５は、スナップ・ショット１９０ａのすべてのビットマップ（すなわちビットマップ１９３、１９５など）をボリューム１７５の適切なアロケーション１９０ｂに転送する。例えば、図１Ｂに、メモリ・スナップショット部分１８０ａおよび１８５ａが、これらが生成された複製サイクルがすでに経過したため空になっているのを示す。さらに、ボリューム１７５の対応するアロケーション１８０ｂ、１８５ｂなどは今や、前にメモリ部分１８０ａ、１８５ａにそれぞれ記憶されていた「すべてのビットマップ」１８３、１８７を含む。最終的には、ビットマップのセットに対応するスナップ・ショットが削除されたとき、ビットマップ（例えば１８３、１８７）がボリューム（例えば１７５）上に残るものとすることができる。

【００４６】

これらのメモリ内ビットマップ（例えば１９３、１９５）は、いくつかの異なる方法で作成し実装することができる。一実装形態では、例えば、バックアップ・サーバ１１０が、プロダクション・サーバ１０５のボリュームのシャドウ・コピー・スナップショット（例えばスナップ・ショット１５０）を取り、入出力制御（ＩＯＣＴＬ）が、シャドウ・コピー・プロバイダ（ソフトウェアまたはハードウェア）に送られるものとすることができる。このＩＯＣＴＬをボリューム・フィルタ・ドライバ１１５によってインターセプトして、アクティブなビットマップを分割することができる。これに応答して、またシャドウ・コピーの作成中に、ボリューム・フィルタ・ドライバ１１５は、フリーズされたビットマップセット（例えば１８０ａ／１８０ｂ、１８５ａ／１８５ｂ）および新しいアクティブなビットマップセット（例えば１９０ａ／１９０ｂ）を作成することにより、分割を同期させる。別の代替実装形態では、シャドウ・コピー差分領域（例えばＶＳＳ差分領域）を採取することのできるエンティティが、ボリューム・レベルの変更を認識していることができ、またＵＳＮ／ファイル・システムをも認識していることができる。したがって、このエンティティは、変更されたファイルのセットとファイル中の変更のセットとを与えることになる抽象化を提供することができる。次いで、適切な複製またはバックアップアプリケーションが、複製を達成するためにこのインフラストラクチャを使用することができる。

【００４７】

複製サイクルがトリガされたとき、ボリューム・フィルタ・ドライバ１１５は、メモリ１７０の使用を低減するために、フリーズされたビットマップをディスクに渡す。ボリューム・フィルタ・ドライバ１１５はまた、一番最近のスナップショット（例えば「t_{n-1}」）以降に生じたすべての変更について照会することのできる１つまたは複数のＩＯＣＴＬを公開することができる。一実装形態では、これらのＩＯＣＴＬに照会すると、一番最近のスナップ・ショット以降に累積したすべてのビットマップが返される。本明細書に述べるメモリ内ビットマップを使用して変更を監視することは、少なくとも一部には、メモリ内ビットマップがプロダクション・サーバ１０５のリソースに大きく影響しない傾向があるので、非常に効率的となり得ることを理解するであろう。

【００４８】

代替の一実装形態では、またこれらの監視された変更のそれぞれを識別するために、バックアップ・サーバ１１０はまた、例えばＵＳＮジャーナル１４０を使用して（または他の監視されたファイル・メタデータを使用して）、変更されたファイルのセットを識別することもできる。次いで、バックアップ・サーバ１１０（または関連するコンポーネント）は、変更された各ファイルが占めるファイル範囲を、プロダクション・サーバ１０５のファイル・システムに照会することができる。ファイル・システムからの照会されたファイル範囲と、ボリューム・フィルタ・ドライバ１１５が報告するファイル範囲との共通部分が、最後の複製サイクル以降に変化したファイルの範囲を提供することができ、したがっていくつかのファイル（例えばデータベース・ファイル）をいくつかの複製プロセスから除外することを可能にする。次いで、バックアップ・サーバ１１０（または関連するコンポーネント）は、変更された各ファイル（ＵＳＮジャーナルによって、または同様に構成されたメタ・データ文書によって報告されたもの）についてこのプロセスを繰り返すこ

とができる。

【００４９】

ＵＳＮジャーナルを使用してプロダクション・サーバ１０５における変更を監視することに特に関して、プロダクション・サーバ１０５におけるファイル参照番号（ＦＲＮ）が、バックアップ・サーバ１１０に記憶された同じファイルのＦＲＮと一致しない場合があることを理解するであろう。このため、監視された変更をプロダクション・サーバ１０５からバックアップ・サーバ１１０に送信するために、場合によっては、変更された特定ファイルへの正しい一致するパスを計算することが重要であることがある。例えば、ボリュームが以下の変更を有する場合があり、これらの変更は、プロダクション・サーバ１０５におけるファイル「ｙ．ｔｘｔ」へのパスを修正することを含む。

１）修正　Ｃ：＼ａ＼ｂ＼ｃ＼ｙ．ｔｘｔ

２）名前変更　Ｃ：＼ａ＼ｂからＣ：＼ｐ＼ｑ＼ｒへ

３）修正　Ｃ：＼ｐ＼ｑ＼ｒ＼ｂ＼ｃ＼ｙ．ｔｘｔ

４）削除　Ｃ：＼ａ

すぐ上に示した例では、プロダクション・サーバ１０５における元のパス「ａ＼ｂ」がパス「ｐ＼ｑ＼ｒ」として名前変更され、元のパスのディレクトリ「＼ａ」が削除される。これにより、プロダクション・サーバ１０５におけるファイル「ｙ．ｔｘｔ」へのパスは「Ｃ：＼ｐ＼ｑ＼ｒ＼ｂ＼ｃ＼ｙ．ｔｘｔ」になる。しかし、プロダクション・サーバ１０５における「ｙ．ｔｘｔ」に対するこれらのパス変更は、自動的にバックアップ・サーバ１１０におけるパスに対する変更をもたらさない場合がある。具体的には、バックアップ・サーバ１１０におけるｙ．ｔｘｔへのパスは一般に、「ａ＼ｂ＼ｃ＼ｙ．ｔｘｔ」のままとなる。

【００５０】

スナップ・ショット時、プロダクション・サーバ１０５はまた、例示的なレコード１～５について、以下に示す変更をＵＳＮジャーナルに記録することができる。

【００５１】

【表１】

1) {rsn = modify,	FRN = yFRN,	parentFRN = cFRN,	filename = y.txt}
2) {rsn = rename-old,	FRN = bFRN,	parentFRN = aFRN,	filename = b}
3) {rsn = rename-new,	FRN = bFRN,	parentFRN = rFRN,	filename = b}
4) {rsn = modify,	FRN = yFRN,	parentFRN = cFRN,	filename = y.txt}
5) {rsn = delete,	FRN = aFRN,	parentFRN = root,	filename = a}

【００５２】

加えて、前述の例から、複製時のプロダクション・サーバ１０５の関連するボリュームの状態は「Ｃ：＼ｐ＼ｑ＼ｒ＼ｂ＼ｃ＼ｙ．ｔｘｔ」として示される。

【００５３】

この特定の例において、レコード１の変更をバックアップ・サーバ１１０に送信するためには、プロダクション・サーバ１０５は、「ｙ．ｔｘｔ」へのファイル・パスをバックアップ・サーバ１１０から取り出す必要がある。具体的には、前述のパス変更のせいで、「ｃ－ＦＲＮ」についてのスナップ・ショット中のプロダクション・サーバ１０５における「ｙ．ｔｘｔ」へのパスはＣ：＼ｐ＼ｑ＼ｒ＼ｂ＼ｃ＼ｙ．ｔｘｔであり、これは、バックアップ・サーバ１１０におけるそのパス（すなわちＣ：＼ａ＼ｂ＼ｃ＼ｙ．ｔｘｔ）とは異なる。本発明の実装形態は、この問題を少なくとも２つの代替手法で解決することができる。一実装形態では、例えば、ＵＳＮジャーナルが単にバックアップ・サーバ１１０からパスメタデータを取り出してリレーショナル・データベースに記憶し、それによりプロダクション・サーバ１０５とバックアップ・サーバ１１０とでファイル・パスを継続

的に相関させることができる。

【００５４】

代替の一実装形態では、プロダクション・サーバ１０５がＵＳＮジャーナルを２回走査することができる。１回目の通過では、プロダクション・サーバ１０５は、この情報をＵＳＮジャーナル１４０の反復的な走査（または「通過」）を介して相関させることができる。例えば、１回の走査では、プロダクション・サーバ１０５は、各フォルダ名変更をキャッシュする。２回目の通過では、プロダクション・サーバ１０５は、キャッシュした名前変更および現在のパスに基づいて、バックアップ・サーバ１１０における対応するパスを計算することができる。例えば、１回目の通過の最後に、プロダクション・サーバ１０５は、削除および／または名前変更されたディレクトリに関する以下の情報をキャッシュすることができる。

{FRN=bFRN,　replica_parent=aFRN,　replica_name=b}

{FRN=aFRN,　replica_parent=root,　replica_name=a}

ｙ．ｔｘｔへのファイル・パスを計算するために、プロダクション・サーバ１０５はまず、レコード「１」中のｙ．ｔｘｔの親ＦＲＮ（すなわち「ファイル参照番号」）がｃＦＲＮであることを識別する。次のステップで、プロダクション・サーバ１０５は、ｃＦＲＮのファイル名、ならびに親ＦＲＮのファイル名を計算する。次いでプロダクション・サーバ１０５は、ファイル・システムに照会する前にキャッシュ内を調べる。この例では、キャッシュは親ｃＦＲＮのファイル名エントリを有さないので、プロダクション・サーバ１０５は、ファイル・システムに照会し、ファイル名がｃであり親がｂＦＲＮであることを決定する。

【００５５】

次いでプロダクション・サーバ１０５は、ｂＦＲＮのファイル名、ならびにｂＦＲＮに対応する親ファイル名を計算する。前と同様、プロダクション・サーバ１０５はまず、ファイル・システムに照会する前に、調べる。この例では、キャッシュはｂＦＲＮのエントリを有するので、プロダクション・サーバ１０５は、ファイル名がｂであり親がａＦＲＮであることを決定する。次いでプロダクション・サーバは、ａＦＲＮのファイル名、およびａＦＲＮの親ファイル名を計算する。この場合もやはり、プロダクション・サーバ１０５はまず、ファイル・システムに照会する前にキャッシュを調べ、この例ではキャッシュはａＦＲＮのエントリを有するので、プロダクション・サーバ１０５は、ａＦＲＮのファイル名が「ａ」であり親ＦＲＮが「ｒｏｏｔ」であることを決定する。

【００５６】

最後に、プロダクション・サーバ１０５は、最終的なパスを「ｃ：＼ａ＼ｂ＼ｃ＼ｙ．ｔｘｔ」として計算する。次に、レコード３（rename-new）が２回目の通過で処理されたとき、キャッシュは、新しい親ファイル名について以下のように更新される。

{FRN=bFRN,　replica_parent=rFRN,　replica_name=b}

{FRN=aFRN,　replica_parent=root,　replica_name=a}

プロダクション・サーバ１０５がレコード４を処理したとき、このレコードはレコード１と同一であるにもかかわらず、ｙ．ｔｘｔについて計算されたパスは今や「Ｃ：＼ｐ＼ｑ＼ｒ＼ｂ＼ｃ＼ｙ．ｔｘｔ」である。キャッシュ中の親「ｂＦＲＮ」が今や「ｒＦＲＮ」なので、計算されたパスは、この特定の場合では異なる。したがって、前述の本文は、プロダクション・サーバ１０５がバックアップ・サーバ１１０に転送するデータの量を最適化するのを２段階通過アルゴリズムがどのように助けることができるかを例示する。具体的には、前述の本文は、プロダクション・サーバ１０５が２回目の通過時に、作成されたファイルがあればそれらを、ならびに修正された後で削除されたファイルがあればそれらを識別し、それによりファイル・パスをバックアップ・サーバ１１０と正しく相関させることのできる、複数の方法を記述する。

【００５７】

したがって、図１Ａ～１Ｂおよび対応する本文は、プロダクション・サーバのデータを実質上連続的な方式で効率的にバックアップするためのいくつかのシステム、コンポーネ

ント、および機構を提供する。上述したことに加えて、本発明の実装形態はまた、特定の結果を達成するための一連の動作を有する方法のフローチャートで記述することもできる。具体的には、図２に、データをバックアップおよび復旧するための、本発明の実装形態によるプロダクション・サーバ１０５とバックアップ・サーバ１１０とから見たフローチャートを示す。以下では、図２に示す動作を、図１Ａ～１Ｂに示したコンポーネントおよび図に関して述べる。

【００５８】

前置きとして、本明細書では、一連の時間における「第１の」、「第２の」、あるいは「第３の」イベント（または「インスタンス」）に言及することがある。しかし、このような指定は単に、連続体上の一意のインスタンスを区別するためであり、したがって「第１の」イベントまたはインスタンスは、「第２の」または「第３の」インスタンスと異なるだけでなく、「第２の」および／または「第３の」インスタンスの前の何らかの時点で生じることを理解するであろう。例えば、データのベースライン「フル」コピー（例えば１４５）の作成および送信は、更新１５０に対する第１のインスタンスとして主に述べるが、何らかの先行イベント（図示せず）に対する第２または第３の（あるいはより後の）インスタンスとして考えることもできる。同様に、更新１５０は、更新１５５（例えば「第２の」インスタンス）に対する「第１の」時間インスタンスで生じるものとして述べることができ、以下同様であるが、本明細書では、これらの用語は主に、ベースライン・フル１４５に対して図示の例で「第２の」および「第３の」時間インスタンスとして述べる。連続的なイベントに関する相対的な意味でのこのような用語の用法は、本明細書における用語「最初の」または「後続の」の用法にも当てはまる。

【００５９】

例えば、図２は、最近のデータを容易にバックアップ・サーバから復旧できるように、実質上連続的かつアプリケーション（すなわち、またはファイル・システム）整合性のある方式でプロダクション・サーバ・データを複製するための、プロダクション・サーバ１０５から見た方法が、ボリューム・データの整合性のあるコピー（consistent copy）をバックアップ・サーバに送信する動作２００を含むのを示す。動作２００は、ボリューム・データのコピーをプロダクション・サーバからバックアップ・サーバに送信することを含み、データは第１の時間インスタンスで整合性がある。例えば、プロダクション・サーバ１０５は（例えば複製エージェントに応答して、またはバックアップ・サーバ１１０からの他のコマンドに応答して）、特定時点（例えば「t_0」）で整合性のあるボリューム・バックアップ全体１４５を作成する。一実装形態では、これは、プロダクション・サーバ１０５におけるすべてのアプリケーション・ライタに、フリーズしてバックアップの準備を開始するよう命じることを含む。次いでプロダクション・サーバ１０５は、ボリューム（あるいは特定のファイル、フォルダ、またはファイル・タイプなど）上のすべてのデータをコピーし、このデータを、バックアップ・サーバ１１０に送信（および記憶）されるよう準備する。

【００６０】

同様に、図２には、最近のデータを容易にバックアップ・サーバから復旧できるように、実質上連続的かつ整合性のある方式でプロダクション・サーバ・データを複製するための、バックアップ・サーバ１１０から見た方法が、整合性のあるボリューム・バックアップをプロダクション・サーバから受信する動作２４０を含むのを示す。動作２４０は、１つまたは複数のボリューム・バックアップをプロダクション・サーバから受信することを含み、１つまたは複数のボリューム・バックアップは、最初の時間インスタンスで整合性がある（すなわちアプリケーションまたはファイル・システム整合性がある）。例えば、図１Ａに、バックアップ・サーバ１１０がフル・バックアップ１４５を受信して記憶するのを示すが、フル・バックアップ１４５は、プロダクション・サーバ１０５からネットワークを介して受信されるか、あるいは、何らかの時点でバックアップ・サーバ１１０と接続されるテープドライブまたは他の記憶ノードから受け取られる。具体的には、バックアップ管理者は、プロダクション・サーバ１０５のハードウェア・スナップ・ショットを取

り、次いでこのスナップ・ショットをバックアップ・サーバ１１０に付与することができる。

【００６１】

加えて、図２には、プロダクション・サーバ１０５から見た方法が、ボリューム・データに対する１つまたは複数の変更を識別する動作２１０を含むのを示す。動作２１０は、１つまたは複数のボリューム・ログファイルを介して、ボリューム・データに対する１つまたは複数の変更を識別することを含む。例えば、図１Ａおよび１Ｂに示したように、ボリューム・フィルタ・ドライバ１１５が、ファイル１２０、１２５、および１３０に対する変更を追跡し、これらの変更をボリューム・ログファイル１３５に記憶することができる。一実装形態では、これらの１つまたは複数の変更は、別法として、ボリューム・ログファイル１３５中に配置する前にメモリ内ビットマップとしてメモリ１７０に記憶することもできる。

【００６２】

図２にはまた、プロダクション・サーバ１０５から見た方法が、１つまたは複数の整合性のある更新をディスクに保存する動作２２０を含むのを示す。動作２２０は、複製サイクル・イベントを識別したときに１つまたは複数のデータ変更を１つまたは複数のボリューム・ログファイルに保存することを含み、１つまたは複数のデータ変更は、第２の時間インスタンスで整合性がある。例えば、複製サイクル・トリガ（例えば複製エージェントからの）を識別すると、ボリューム・フィルタ・ドライバは、メモリ・アロケーション１９０ａ中のビットマップ１９３、１９５を、割り振られた物理ディスク空間１９０ｂ（例えばボリューム・ログファイル１３５にも対応する）に渡す。したがって、この物理ディスク・アロケーション１９０ｂは、ディスク・アロケーション１８５ｂに記憶されているスナップ・ショット１８７（すなわち「すべてのビットマップ」１８７）とは異なる１つの時点のスナップ・ショットを含む。

【００６３】

さらに、図２には、プロダクション・サーバ１０５から見た方法が、複製中に整合性のある更新のコピーをバックアップ・サーバに送信する動作２３０を含むのを示す。動作２３０は、１つまたは複数のデータ変更のコピーをバックアップ・サーバに送信することを含み、それによりバックアップ・サーバは、第１の時間インスタンスおよび第２の時間インスタンスで妥当であるデータのコピーを有する。例えば、フル更新１４５をバックアップ・サーバ１１０に渡すことに加えて、プロダクション・サーバは、整合性のあるスナップ・ショット更新１５０および１５５も送信し、スナップ・ショット更新１５０および１５５はそれぞれ、異なる時点（すなわち「t_1」、「t_2」など）で妥当である。

【００６４】

したがって、図２には、バックアップ・サーバ１１０から見た方法が、１つまたは複数の整合性のある更新（consistent updates）を受信する動作２５０を含む。動作２５０は、１つまたは複数の整合性のあるバックアップ更新を受信することを含み、これらのバックアップ更新の少なくとも１つは、後続の時間インスタンスでの、１つまたは複数のボリューム・バックアップの少なくとも１つに対する整合性のある更新である。例えば、バックアップ・サーバ１１０は、整合性のあるスナップ・ショット更新１５０、１５５のいずれかを受信し、それによりバックアップ・サーバは今や、フル・バックアップ１４５（「t_0」または他の何らかのベースライン・フル）と比較して、種々のインクリメンタルな時点（すなわち「t_1」、「t_2」など）のデータ・コピーを有する。

【００６５】

加えて、図２には、バックアップ・サーバ１１０から見た方法が、復旧要求（recovery request）を受信する動作２６０を含むのを示す。動作２６０は、後続の時間インスタンスに従って妥当であるデータを求める復旧要求を受信することを含む。例えば、バックアップ・サーバ１１０は、時間「t_1」で妥当である特定のファイルを求める要求（図示せず）を受信するが、このファイルは、ベースライン・フル・バックアップ１４５と更新１５０との両方に見られるファイルである。さらに、図２には、バックアップ・サーバ１１

０から見た方法が、後続の時間インスタンスで妥当である要求されたデータを識別する動作２７０を含むのを示す。動作２７０は、後続の時間インスタンスでの要求されたデータを１つまたは複数のバックアップ・サーバ・ボリュームにおいて識別することを含み、要求されたデータは、少なくとも１つの整合性のあるバックアップ更新の少なくとも一部を含む。

【００６６】

例えば、データベース・データに関しては、バックアップ・サーバ１１０は、要求に対して妥当である先行時点および現在時点のそれぞれにおけるファイルのコピーをまとめることができる。すなわち、バックアップ・サーバ１１０は、時間「t_0」および「t_1」からのファイルのコピーを結合する。一方、ファイル・システム・データの場合は、後続の各更新（例えば１５０、１５５）は、要求されたファイルの完全な更新済みコピーを含む場合があり、したがってバックアップ・サーバ１１０は、要求された時点に対する一番最近の更新（または他の何らかの後続の更新）中で、要求されたデータを識別するだけで済むことがある。したがって、要求されたデータのタイプに応じて、バックアップ・サーバ１１０は、要求された時点から最新のベースライン・フルまで遡ってインクリメンタルな各更新を識別することが必要な場合もあり、あるいは単に、最新のポイント・イン・タイム更新中で、要求されたデータを識別すれば済む場合もある。

【００６７】

したがって、図２にはまた、バックアップ・サーバ１１０から見た方法が、要求されたデータを返す動作２８０を含むのを示す。動作２８０は、後続の時間インスタンスで妥当である要求されたデータをプロダクション・サーバに送信することを含む。例えば、データベース・データの場合は、バックアップ・サーバ１１０は、ベースライン・フルデータ（例えば１４５）とインクリメンタルな更新データ（例えば１５０、１５５）との両方を含むリカバリ（復旧）１６０を提供し、ファイル・システム・データの場合は、バックアップ・サーバ１１０は、要求された更新時点からのファイルデータを少なくとも有するリカバリを提供する。バックアップ・サーバ１１０の応答１６０はまた、応答がどの復旧時間で妥当かを示すこともできる。図１Ａに示すように、復旧データ１６０は、データが時間「t_1」で妥当であることを示す。

【００６８】

したがって、本発明の実装形態による図面、コンポーネント、および方法は、従来のバックアップ・システムに勝るいくつかの利点を提供する。具体的には、本発明の実装形態は、データソースにとらわれない（例えばボリューム・フィルタ・ドライバ１１５を介する）、またバックアップ・サーバ上で稼動するアプリケーションのバージョンを必ずしも必要としない、また実質上連続的な複製を表す、データ・バックアップ方式を提供する。前述のように、少なくとも部分的には、これらの最適化は、ボリューム・フィルタ・ドライバ（例えば１１５）を使用して、ボリューム・レベルで、低オーバーヘッドの状態で変更を追跡または監視することを基本とし、また、例えばＵＳＮジャーナルを使用してファイル・レベルで複製サイクルを実施し、この結果、ファイル・レベルの包含／除外も、さらに適用することができる。

【００６９】

加えて、本発明の実装形態は、最適化された方式で、またネットワークを介して完全なデータを転送することを必ずしも必要とせずに、バックアップ・サーバでプロダクション・サーバ・データのフルベースラインコピーを作成することを可能にする。バックアップ・サーバに転送される後続のデータの量を最小限に抑えることにより、ネットワーク上およびプロダクション・サーバ上での潜在的なリソース消費をかなり削減することができる。したがって、本発明の実装形態はさらに、厳格な目標復旧時間を満たすためのいくつかの代替方法を提供する。本発明の実装形態はまた、低い性能オーバーヘッドでデータ変更（例えばバイトまたはバイト・ブロックのレベルの）を追跡し、ファイル・システムおよびハードウェア／ソフトウェア・スナップショットから独立した方式でデータ変更を追跡するための、いくつかの方法を提供する。さらに、本発明の実装形態はまた、関連するプ

ロダクション・サーバ上の永続的な状態を必ずしも必要とせずにパス情報を（ＵＳＮベースの複製などによって）再構築するための１つまたは複数の方法を提供する。

【００７０】

本発明の実施形態は、以下により詳細に論じる様々なコンピュータハードウェアを備えた専用または汎用コンピュータを含むことができる。本発明の範囲内の実施形態はまた、コンピュータ実行可能命令またはデータ構造を搬送するための、あるいはこれらが記憶された、コンピュータ可読媒体も含む。このようなコンピュータ可読媒体は、汎用または専用コンピュータによってアクセスできる任意の利用可能な媒体とすることができる。

【００７１】

限定ではなく例として、このようなコンピュータ可読媒体は、ＲＡＭ、ＲＯＭ、ＥＥＰＲＯＭ、ＣＤ－ＲＯＭ、または他の光学ディスク記憶装置、磁気ディスク記憶装置、または他の磁気記憶デバイスを含むことができ、あるいは、所望のプログラム・コード手段をコンピュータ実行可能命令またはデータ構造の形で搬送または記憶するのに使用でき汎用または専用コンピュータによってアクセスできる他の任意の媒体を含むことができる。情報がネットワークまたは別の通信接続（配線式、ワイヤレス、または配線式とワイヤレスの組合せ）を介してコンピュータに転送または提供されるとき、コンピュータはこの接続をコンピュータ可読媒体として適切に見なす。したがって、このような接続はどれも、コンピュータ可読媒体と適切に呼ばれる。以上の組合せも、コンピュータ可読媒体の範囲内に含めるべきである。

【００７２】

コンピュータ実行可能命令は、例えば、ある機能または機能グループを汎用コンピュータ、専用コンピュータ、または専用処理デバイスに実施させる命令およびデータを含む。構造上の特徴および／または方法上の動作に特有の言語で本主題を述べたが、添付の特許請求の範囲に定義する本主題は、前述の特定の特徴または動作に必ずしも限定されないことを理解されたい。そうではなく、前述の特定の特徴および動作は、特許請求の範囲を実施する例示的な形として開示する。

【００７３】

本発明は、その趣旨または本質的な特性から逸脱することなく他の特定の形で具体化することもできる。述べた実施形態は、あらゆる点で、例示としてのみ考えるべきであり、限定として考えるべきではない。したがって、本発明の範囲は、以上の記述によってではなく添付の特許請求の範囲によって示す。特許請求の範囲の均等物の意味および範囲の内に入るあらゆる変更は、それらの範囲に包含されるべきである。

【図面の簡単な説明】

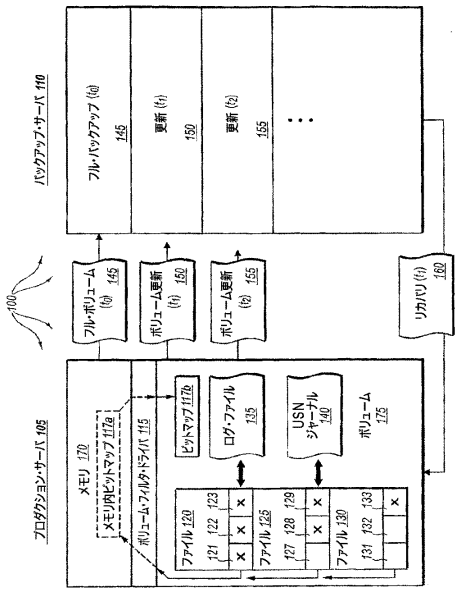
【００７４】

【図１Ａ】プロダクション・サーバがインクリメンタルなアプリケーション（またはファイル・システム）整合バックアップを作成して、これらのバックアップをバックアップ・サーバに送信する、本発明の一実装形態によるアーキテクチャ概観図である。

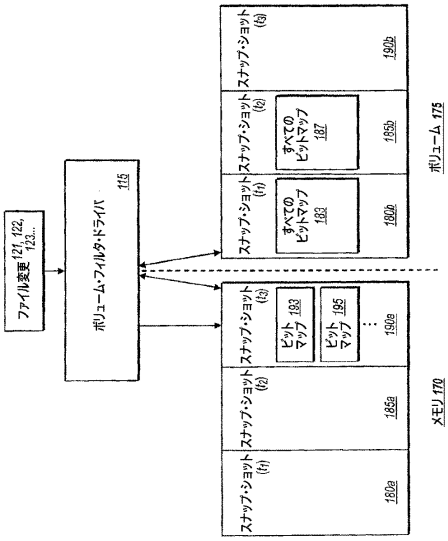
【図１Ｂ】ボリューム・フィルタ・ドライバがシステム・メモリおよび１つまたは複数の物理ディスクを使用してボリュームに対する変更を監視する、本発明の一実装形態による概観図である。

【図２】本発明の実装形態による、プロダクション・サーバおよびバックアップ・サーバの観点から実施される一連の動作を含む方法のフローチャートである。

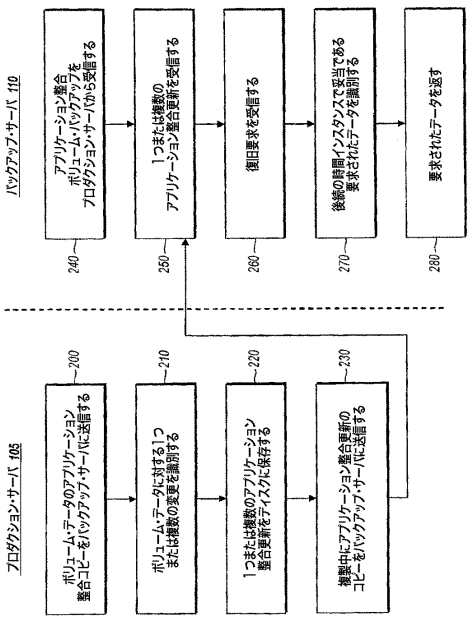
【図１Ａ】


バックアップ・サーバ 110
フル・バックアップ (t0) 145
更新 (t1) 150
更新 (t2) 155
フル・ボリューム (t0) 145
ボリューム更新 (t1) 150
ボリューム更新 (t2) 155
リカバリ (t1) 160
100
プロダクション・サーバ 105
メモリ 170
メモリ内ビットマップ 117a
ボリューム・フィルタ・ドライバ 115
ビットマップ 117b
ログ・ファイル 135
USN ジャーナル 140
ボリューム 175
ファイル 120
ファイル 125
ファイル 130

【図１Ｂ】


ファイル変更 121, 122, 123...
ボリューム・フィルタ・ドライバ 115
スナップショット (t1) 180a
スナップショット (t2) 180b
スナップショット (t3) 180c
すべてのビットマップ 183
ビットマップ 193
ビットマップ 195
メモリ 170
ボリューム 175

【図２】


バックアップ・サーバ 110
アプリケーション整合ボリューム・バックアップをプロダクション・サーバから受信する 240
1つまたは複数のアプリケーション整合更新を受信する 250
御旨要求を受信する 260
後続の御旨インスタンスで要求されたデータを識別する 270
要求されたデータを返す 280
プロダクション・サーバ 105
ボリューム・データのアプリケーション整合コピーをバックアップ・サーバに送信する 200
ボリューム・データに対する1つまたは複数の変更を識別する 210
1つまたは複数のアプリケーション整合更新をディスクに保存する 220
複製中にアプリケーション整合更新のコピーをバックアップ・サーバに送信する 230

フロントページの続き

(72)発明者 ヴィヴェック サハスラナマン
アメリカ合衆国 ９８０５２ ワシントン州 レッドモンド ワン マイクロソフト ウェイ マイクロソフト コーポレーション インターナショナル パテンツ内
(72)発明者 ビナイ エス．バダミ
アメリカ合衆国 ９８０５２ ワシントン州 レッドモンド ワン マイクロソフト ウェイ マイクロソフト コーポレーション インターナショナル パテンツ内
(72)発明者 アビッド アリ
アメリカ合衆国 ９８０５２ ワシントン州 レッドモンド ワン マイクロソフト ウェイ マイクロソフト コーポレーション インターナショナル パテンツ内
(72)発明者 アミット シングラ
アメリカ合衆国 ９８０５２ ワシントン州 レッドモンド ワン マイクロソフト ウェイ マイクロソフト コーポレーション インターナショナル パテンツ内
(72)発明者 キャランディープ シン アナンド
アメリカ合衆国 ９８０５２ ワシントン州 レッドモンド ワン マイクロソフト ウェイ マイクロソフト コーポレーション インターナショナル パテンツ内
(72)発明者 ロバート エム．フライズ
アメリカ合衆国 ９８０５２ ワシントン州 レッドモンド ワン マイクロソフト ウェイ マイクロソフト コーポレーション インターナショナル パテンツ内

審査官 田川 泰宏

(56)参考文献 特開２００５－１２２６１１（ＪＰ，Ａ）
特開２００６－０９２５５３（ＪＰ，Ａ）
特開平１０－２１４２１４（ＪＰ，Ａ）
特開２００５－２６７５６９（ＪＰ，Ａ）
特開２００６－０１２１２１（ＪＰ，Ａ）
特開２００４－２２７５７２（ＪＰ，Ａ）
国際公開第９４／２９８０７（WO，Ａ３）
特開２００４－０３８９２８（ＪＰ，Ａ）
米国特許出願公開第２００４／１３９１２８（ＵＳ，Ａ１）
米国特許出願公開第２００５／２６２０９７（ＵＳ，Ａ１）

(58)調査した分野(Int.Cl.，ＤＢ名)
G06F 12/00