(19)日本国特許庁(JP)　(12)特　許　公　報(B2)　(11)特許番号
特許第5649184号
(P5649184)
(45)発行日　平成27年1月7日(2015.1.7)　(24)登録日　平成26年11月21日(2014.11.21)

(51)Int.Cl.　F I
G06F 9/445 (2006.01)　G06F 9/06 610Q
G06F 9/06 610K

請求項の数 14 (全 26 頁)

(21)出願番号　特願2011-252538(P2011-252538)
(22)出願日　平成23年11月18日(2011.11.18)
(65)公開番号　特開2012-128841(P2012-128841A)
(43)公開日　平成24年7月5日(2012.7.5)
審査請求日　平成26年5月28日(2014.5.28)
(31)優先権主張番号　10194866.9
(32)優先日　平成22年12月14日(2010.12.14)
(33)優先権主張国　欧州特許庁(EP)

早期審査対象出願

(73)特許権者　390009531
インターナショナル・ビジネス・マシーンズ・コーポレーション
INTERNATIONAL BUSINESS MACHINES CORPORATION
アメリカ合衆国10504　ニューヨーク州　アーモンク　ニュー　オーチャード　ロード
(74)代理人　100108501
弁理士　上野　剛史
(74)代理人　100112690
弁理士　太佐　種一
(74)代理人　100091568
弁理士　市位　嘉宏

最終頁に続く

(54)【発明の名称】ブート・ブロックの再配置によって複数のソフトウェア・イメージを管理するための方法、コンピュータ・プログラムおよびシステム

(57)【特許請求の範囲】
【請求項１】
複数のメモリ位置を有するマス・メモリを含むデータ処理エンティティ中で複数のソフトウェア・イメージを管理するための方法であって、前記複数のメモリ位置の各々がマス・メモリ内の対応するメモリ・アドレスを有し、ソフトウェア・イメージの各々が、前記ソフトウェア・イメージ内の対応するイメージ・アドレスを有する複数のメモリ・ブロックを含み、前記方法は、
前記複数のソフトウェア・イメージの各ソフトウェア・イメージの前記メモリ・ブロックの少なくとも部分を前記マス・メモリ内の対応するイメージ部分それぞれに保存するステップであって、各メモリ・ブロックは前記対応するイメージ・アドレスに前記マス・メモリ内の前記対応するイメージ部分のオフセットを加えたものに等しい前記メモリ・アドレスを有する前記メモリ位置に保存される、前記保存するステップと、
前記データ処理エンティティに格納されている別のソフトウェア・イメージが選択されることに応じて、前記選択された別のソフトウェア・イメージ（以下、「別のソフトウェア・イメージ」という）のブート位置に保存された前記メモリ・ブロックを前記マス・メモリの再配置部分に再配置するステップであって、前記別のソフトウェア・イメージの前記ブート位置は、前記別のソフトウェア・イメージにアクセスするために適合されるアクセス機能をロードするまで前記データ処理エンティティをブートするために必要とされる前記メモリ・ブロックを含む前記別のソフトウェア・イメージのブート・ブロックの前記イメージ・アドレスに等しい前記メモリ・アドレスを有する前記メモリ位置である、前記

再配置するステップと、
前記別のソフトウェア・イメージの前記ブート・ブロックを前記対応するブート位置にコピーするステップと、
前記対応するブート位置における前記別のソフトウェア・イメージの前記ブート・ブロックから前記データ処理エンティティをブートすることによって前記アクセス機能をロードするステップと、
前記アクセス機能によって、前記別のソフトウェア・イメージの選択されたメモリ・ブロックにアクセスする各要求を提供するステップであって、前記アクセス機能は、前記対応するイメージ・アドレスに、前記別のソフトウェア・イメージを保存しているイメージ部分（以下、「別のイメージ部分」という）の前記オフセットを加えたものに等しい前記メモリ・アドレスを有する前記メモリ位置における前記選択されたメモリ・ブロックにアクセスする、前記提供するステップと
を含む、前記方法。
【請求項２】
前記別のソフトウェア・イメージの前記ブート位置に保存された前記メモリ・ブロックを前記再配置部分に再配置するステップの前に、前のソフトウェア・イメージの前記ブート位置から前記再配置部分に再配置されていた前記メモリ・ブロックを前記前のソフトウェア・イメージの前記ブート位置に復元するステップ
をさらに含む、請求項１に記載の方法。
【請求項３】
前記対応するイメージ部分はゼロのオフセットを有する最初のイメージ部分を含み、前記最初のイメージ部分にはすべての前記対応するイメージ部分の前記ブート位置が含まれ、前記別のソフトウェア・イメージの前記ブート位置に保存された前記メモリ・ブロックを再配置するステップと、前記別のソフトウェア・イメージの前記ブート・ブロックをコピーするステップとは、前記別のイメージ部分が前記最初のイメージ部分と異なるときにのみ行われる、請求項１または２に記載の方法。
【請求項４】
スナップショットとなるためのソース・イメージ部分に保存されたソース・ソフトウェア・イメージを選択するステップと、
前記ソース・ソフトウェア・イメージの各メモリ・ブロックをフリーであるイメージ部分にコピーするステップであって、前記ソース・ソフトウェア・イメージの前記メモリ・ブロックは、前記再配置部分に再配置されたときには前記再配置部分から、別様では前記ソース・イメージ部分からコピーされる、前記コピーするステップと
をさらに含む、請求項１～３のいずれか１項に記載の方法。
【請求項５】
削除されるべき古いイメージ部分に保存された古いソフトウェア・イメージを選択するステップと、
前記古いイメージ部分をフリーと設定するステップと
をさらに含む、請求項１～４のいずれか１項に記載の方法。
【請求項６】
各ソフトウェア・イメージのすべての前記メモリ・ブロックが前記対応するイメージ部分に保存される、請求項１～５のいずれか１項に記載の方法。
【請求項７】
前記別のソフトウェア・イメージの前記ブート・ブロックから前記データ処理エンティティをブートする前記ステップは、
前記別のイメージ部分のサイズに従って前記アクセス機能によって前記別のソフトウェア・イメージのサイズ変更を行うステップ
を含む、請求項１～６のいずれか１項に記載の方法。
【請求項８】
前記保存するステップが、

外部ソースから前記データ処理エンティティの新しいイメージ部分への新しいソフトウェア・イメージの配置を要求するステップと、
前記外部ソースから前記データ処理エンティティに前記新しいソフトウェア・イメージの前記ブート・ブロックをダウンロードするステップと、
前記新しいソフトウェア・イメージの前記ブート位置に保存された前記メモリ・ブロックを前記再配置部分に再配置するステップと、
前記新しいソフトウェア・イメージの前記ブート・ブロックを前記対応するブート位置にコピーするステップと、
前記対応するブート位置における前記新しいソフトウェア・イメージの前記ブート・ブロックから前記データ処理エンティティをブートすることによって前記アクセス機能をロードするステップと、
前記アクセス機能によって、前記新しいソフトウェア・イメージのさらに選択されたメモリ・ブロックにアクセスする各要求を提供するステップであって、前記アクセス機能は、
前記さらに選択されたメモリ・ブロックが前記マス・メモリにないことに応答して、前記外部ソースから前記さらに選択されたメモリ・ブロックをダウンロードして、前記さらに選択されたメモリ・ブロックを、前記対応するイメージ・アドレスに前記新しいイメージ部分の前記オフセットを加えたものに等しい前記メモリ・アドレスを有する前記メモリ位置に保存するステップ；又は
前記対応するイメージ・アドレスに前記新しいイメージ部分の前記オフセットを加えたものに等しい前記メモリ・アドレスを有する前記メモリ位置から前記さらに選択されたメモリ・ブロックを検索するステップ
を実行する、前記提供するステップと
を含む、請求項１～７のいずれか１項に記載の方法。
【請求項９】
前記保存するステップが、
前記新しいソフトウェア・イメージの前記ブート位置に保存された前記メモリ・ブロックを再配置するステップの前に、前のソフトウェア・イメージの前記ブート位置から前記再配置部分に再配置されていた前記メモリ・ブロックを前記前のソフトウェア・イメージの前記ブート位置に復元するステップ
をさらに含む、請求項８に記載の方法。
【請求項１０】
前記保存するステップが、
前記対応するイメージ・アドレスに前記新しいイメージ部分の前記オフセットを加えたものに等しい前記メモリ・アドレスを有する前記メモリ位置に、前記新しいソフトウェア・イメージの前記ブート・ブロックをコピーするステップ
をさらに含む、請求項８または９に記載の方法。
【請求項１１】
前記保存するステップが、
前記データ処理エンティティの作業負荷をモニタするステップと、
前記作業負荷が閾値よりも低くなったことに応答して、前記外部ソースから前記マス・メモリに保存されていない新しいメモリ・ブロックの組をダウンロードして、前記新しいメモリ・ブロックを、前記対応するイメージ・アドレスに前記新しいイメージ部分の前記オフセットを加えたものに等しい前記メモリ・アドレスを有する前記メモリ位置に保存するステップ
をさらに含む、請求項８～１０のいずれか１項に記載の方法。
【請求項１２】
前記アクセス機能の前記ロードまでの前記ソフトウェア・イメージからの補助データ処理エンティティのブート・シーケンスをシミュレートするステップであって、前記ソフトウェア・イメージの各メモリ・ブロックは前記ブート・シーケンスが追跡される際にア

クセスされる、前記シミュレートするステップと、
　前記追跡されたメモリ・ブロックに従って前記ソフトウェア・イメージの前記ブート・ブロックを識別するステップと
　によって各ソフトウェア・イメージを準備するステップ
　をさらに含む、請求項１～１１のいずれか１項に記載の方法。
【請求項１３】
　データ処理システムに、請求項１～１２のいずれか１項に記載の方法の各ステップを実行させる、コンピュータ・プログラム。
【請求項１４】
　コンピュータ・システムであって、請求項１～１２のいずれか１項に記載の方法の各ステップを行うための手段を備えている、前記コンピュータ・システム。

【発明の詳細な説明】
【技術分野】
【０００１】
　本発明の１つまたはそれ以上の実施形態に従う解決策は、データ処理分野に関する。より特定的には、この解決策はソフトウェア・イメージの管理に関する。
【背景技術】
【０００２】
　一般的に言って、ソフトウェア・イメージとはコンピュータ上に存在するソフトウェア・モジュール（たとえばそのオペレーティング・システム、アプリケーション・プログラム、またはデータあるいはその組合わせなど）を含む構造である。
【０００３】
　場合によっては、同じコンピュータ上で複数のオペレーティング・システムを利用可能にすることが望ましいことがある。それはたとえば異なるオペレーティング・システムを必要とするプログラムを実行するため、または新しいオペレーティング・システムもしくはその新しいバージョンをテストするためなどに有用であり得る。
【０００４】
　この目的のために、マルチブート技術を利用することが可能である。マルチブート技術は、コンピュータに複数のオペレーティング・システムをインストールして、コンピュータが始動されたときにどのオペレーティング・システムをブートするかを選択可能にすることができる。この所望の結果は、コンピュータのハードディスクを複数の区画に分割し、各区画が対応するオペレーティング・システムを保存する論理ディスクを定めることによって達成される。コンピュータは、始動させるべき実際のオペレーティング・システムの選択を可能にするブート・ローダを含む１次区画からブートする。代替的には、単に所望のオペレーティング・システムを選択するために用いられる１次ブート・ローダを有するブート区画を提供することもできる。１次ブート・ローダは、次いで選択されたオペレーティング・システムの２次ブート・ローダを始動のために呼び出す。
【０００５】
　しかし、ハードディスクの区画を予め定める必要があるため、マルチブート技術はかなり融通がきかないものである。いずれの場合にも、一旦選択されたオペレーティング・システムが始動されると、それがハードディスク全体のすべての制御を行う。したがって、選択されたオペレーティング・システムは他の区画にもアクセスすることがある（それらの区画に損傷を与える危険性を伴う）。
【０００６】
　代替的には、仮想化技術を利用して同じ結果を達成してもよい。この場合には、コンピュータにハイパーバイザがインストールされる。ハイパーバイザは仮想化層を実現し、これは複数の仮想マシンをエミュレートするものであって、各々の仮想マシンは（その仮想マシンが唯一の制御を行う）物理的コンピュータの外観を与える抽象的環境からなる。このやり方で、対応する仮想マシン上で異なるオペレーティング・システムを独立に（同時

であってもよい）実行させることが可能である。
【０００７】
しかし、仮想化技術は、仮想マシンを管理するための複雑なインフラストラクチャのインストールを必要とする。さらに、このことはコンピュータの性能効率低下を伴う（オペレーティング・システムがもはやコンピュータ上でネイティブに実行されないため）。
【０００８】
特許文献１（そのすべての開示が本明細書において引用により援用される）も、マルチブート・コンピュータを仮想マシンに変換するための方法を開示している。この目的のために、コンピュータのブート記録は、仮想マシンを管理するホスティング・オペレーティング・システムをロードするように構成される。仮想マシンは、対応するブート・イメージからコンバータによって生成され、このコンバータは同じ構成を検出および適用して、仮想マシンの同時実行によってもたらされ得るあらゆる矛盾を解決する。上記と同様、これは（ホスティング・オペレーティング・システムによって実現される）仮想化層のインストールを必要とする。さらに、オペレーティング・システムはここでも（ホスティング・オペレーティング・システムによって提供される）仮想化環境において実行され、対応する性能効率低下を伴う。
【０００９】
別の一般的な要求は、コンピュータのソフトウェア・イメージのバックアップ・コピーを作成するというものである。これはたとえば、コンピュータの誤動作が起こった場合にコンピュータの内容を復元するために有用であり得る。
【００１０】
この目的のために、ソフトウェア・イメージのスナップショット（すなわち特定の時点での整合状態におけるそのバックアップ・コピー）を取ることが可能である。スナップショットは、バックアップ・ディスクまたはバックアップ・サーバにセーブされてもよい。このやり方で、スナップショットをバックアップ・ディスクまたはバックアップ・サーバからコンピュータに再インストールすることによって、スナップショットを復元することが可能である。しかし、スナップショットを復元するプロセスは非常に遅い。加えて、バックアップ・サーバを使用する場合、これはネットワーク・リソースの高消費を伴う。さらに、バックアップ・サーバからスナップショットを復元するためにバックアップ・サーバとのネットワーク接続が必要である。代替的に、バックアップ・サーバ上のスナップショットから遠隔的にコンピュータをブートすることが可能である。しかしこの場合には、コンピュータはその動作のためにバックアップ・サーバに常時接続される必要がある。いずれの場合にも、ネットワーク上でのコンピュータの動作はその性能の効率低下を伴う。
【先行技術文献】
【特許文献】
【００１１】
【特許文献１】米国特許出願公開第２００９／０１９３２４５号
【発明の概要】
【発明が解決しようとする課題】
【００１２】
一般的な見地からみて、本発明の１つまたはそれ以上の実施形態に従う解決策は、複数のソフトウェア・イメージをそのブート・ブロックを再配置することによって管理するという考えに基づいている。
【課題を解決するための手段】
【００１３】
特に、本発明の特定の実施形態に従う解決策の１つまたはそれ以上の局面が独立請求項に示されており、同じ解決策の有利な特徴が従属請求項に示されており、その表現は本明細書において一語一句違わず引用により援用される（本発明の実施形態に従う解決策の特定の局面を参照して提供されるあらゆる有利な特徴は、そのあらゆる他の局面に準用して適用される）。

【００１４】
より特定的には、本発明の実施形態に従う解決策の１局面は、データ処理エンティティ（たとえばスタンドアロン・コンピュータなど）の中で複数のソフトウェア・イメージを管理するための方法を提供する。データ処理エンティティは、複数のメモリ位置を有するマス・メモリを含む。各メモリ位置は、マス・メモリ内に対応するメモリ・アドレスを有する。次に、各ソフトウェア・イメージは複数のメモリ・ブロックを含む。各メモリ・ブロックはソフトウェア・イメージ内に対応するイメージ・アドレスを有する。この方法は以下のステップを含む。各ソフトウェア・イメージのメモリ・ブロック（または少なくともその部分）が、マス・メモリの対応するイメージ部分に保存される。特に、各メモリ・ブロックは、対応するイメージ・アドレスにマス・メモリ内のイメージ部分のオフセットを加えたものに等しいメモリ・アドレスを有するメモリ位置に保存される。（たとえば、前にセーブされたスナップショットに切換えることなどによって）現在のイメージ部分に保存された現在のソフトウェア・イメージが選択される。現在のソフトウェア・イメージのブート位置に保存されたメモリ・ブロックが、マス・メモリの再配置部分に再配置される。現在のソフトウェア・イメージのブート位置は、現在のソフトウェア・イメージのブート・ブロックのイメージ・アドレスに等しいメモリ・アドレスを有するメモリ位置であり、このブート・ブロックは（現在のソフトウェア・イメージにアクセスするために適合される）アクセス機能をロードするまでデータ処理エンティティをブートするために必要とされる現在のソフトウェア・イメージのメモリ・ブロックを含む。現在のソフトウェア・イメージのブート・ブロックは、対応するブート位置にコピーされる。対応するブート位置にある現在のソフトウェア・イメージのブート・ブロックからデータ処理エンティティがブートされることによって、アクセス機能がロードされる。現在のソフトウェア・イメージの選択されたメモリ・ブロックにアクセスする各要求は、アクセス機能によって提供される。アクセス機能は、対応するイメージ・アドレスに現在のイメージ部分のオフセットを加えたものに等しいメモリ・アドレスを有するメモリ位置にある選択されたメモリ・ブロックにアクセスする。

【００１５】
本発明の実施形態に従う解決策のさらなる局面は、データ処理システムにおいてコンピュータ・プログラムが実行されるときにデータ処理システムに本方法のステップを行わせるためのコード手段を含むコンピュータ・プログラムを提供する。本発明の実施形態に従う解決策のさらなる局面は、コンピュータ・プログラムを包含する持続性コンピュータ読取可能媒体を含むコンピュータ・プログラム製品を提供し、このコンピュータ・プログラムはデータ処理システムのワーキング・メモリに直接ロード可能なコード手段を含むことによって、同方法を行うようにデータ処理システムを構成する。

【００１６】
本発明の実施形態に従う解決策の別の局面は、前記方法のステップを行うための手段を含むシステムを提供する。

【図面の簡単な説明】

【００１７】
【図１】本発明の実施形態に従う解決策を適用可能なデータ処理システムの概略的ブロック図である。
【図２】本発明の実施形態に従う配置プロセスを実施するために使用可能な主要ソフトウェア構成要素の役割を表す協調図である。
【図３】本発明の実施形態に従う配置プロセスを実施するために使用可能な主要ソフトウェア構成要素の役割を表す協調図である。
【図４】本発明の実施形態に従う配置プロセスを実施するために使用可能な主要ソフトウェア構成要素の役割を表す協調図である。
【図５】本発明の実施形態に従う配置プロセスを実施するために使用可能な主要ソフトウェア構成要素の役割を表す協調図である。
【図６】本発明の実施形態に従う配置プロセスを実施するために使用可能な主要ソフトウ

ェア構成要素の役割を表す協調図である。
【図７】本発明の実施形態に従う配置プロセスを実施するために使用可能な主要ソフトウェア構成要素の役割を表す協調図である。
【図８】本発明の実施形態に従う配置プロセスを実施するために使用可能な主要ソフトウェア構成要素の役割を表す協調図である。
【図９】本発明の実施形態に従う配置プロセスを実施するために使用可能な主要ソフトウェア構成要素の役割を表す協調図である。
【図１０】本発明の実施形態に従う配置プロセスを実施するために使用可能な主要ソフトウェア構成要素の役割を表す協調図である。
【図１１】本発明の実施形態に従う配置プロセスを実施するために使用可能な主要ソフトウェア構成要素の役割を表す協調図である。
【図１２】本発明の実施形態に従う配置プロセスを実施するために使用可能な主要ソフトウェア構成要素の役割を表す協調図である。
【図１３】本発明の実施形態に従うスナップショット・プロセスを実施するために使用可能な主要ソフトウェア構成要素の役割を表す協調図である。
【図１４】本発明の実施形態に従うスナップショット・プロセスを実施するために使用可能な主要ソフトウェア構成要素の役割を表す協調図である。
【図１５】本発明の実施形態に従うスナップショット・プロセスを実施するために使用可能な主要ソフトウェア構成要素の役割を表す協調図である。
【図１６】本発明の実施形態に従うスナップショット・プロセスを実施するために使用可能な主要ソフトウェア構成要素の役割を表す協調図である。
【図１７】本発明の実施形態に従うスナップショット・プロセスを実施するために使用可能な主要ソフトウェア構成要素の役割を表す協調図である。
【図１８】本発明の実施形態に従う準備プロセスを実施するために使用可能な主要ソフトウェア構成要素の役割を表す協調図である。
【発明を実施するための形態】
【００１８】
本発明の１つまたはそれ以上の実施形態に従う解決策、ならびにそのさらなる特徴および利点は、添付の図面とともに読まれるべき、純粋に非制限的表示によって与えられる以下の詳細な説明を参照することによって最もよく理解されるだろう（この説明においては簡略化のため、対応する構成要素は同一または類似の参照によって示され、その説明は繰り返されず、さらに各エンティティの名前はそのタイプおよび属性の両方、たとえばその値、内容および表現などを示すために一般的に用いられる）。
【００１９】
特に図１を参照すると、本発明の実施形態に従う解決策を適用可能なデータ処理システム（または単にシステム）１００の概略的ブロック図が示される。システム１００は分散アーキテクチャを有し、これはたとえばローカル・エリア・ネットワーク（Ｌｏｃａｌ Ａｒｅａ Ｎｅｔｗｏｒｋ：ＬＡＮ）などのネットワーク１０５に基づくものである。ネットワーク１０５を通じて複数のコンピュータが互いに接続される。特にサーバ・コンピュータ１１０は、簡略化のために図面に２つだけ示されているクライアント・コンピュータ１１５へのソフトウェア・イメージの配置を制御する。各ソフトウェア・イメージは、１つまたはそれ以上のソフトウェア・モジュール（たとえば、オペレーティング・システム、アプリケーション・プログラムまたはデータあるいはその組合わせなど）を含む構造である。
【００２０】
システム１００の一般的な（サーバまたはクライアント）コンピュータは、（システム１００中のコンピュータの実際の機能に従って好適に調整された構造を有する）システム・バス１２０に並列接続されたいくつかのユニットによって形成される。詳述すると、１つまたはそれ以上のマイクロプロセッサ（ｍｉｃｒｏｐｒｏｃｅｓｓｏｒｓ：μＰ）１２５はコンピュータの動作を制御する。ＲＡＭ１３０はマイクロプロセッサ１２５によって

ワーキング・メモリとして用いられ、ＲＯＭ１３５はコンピュータの基本コードを保存する。ローカル・バス１４０の周りに（それぞれのインタフェースによって）いくつかの周辺ユニットが集まっている。特に、マス・メモリは１つまたはそれ以上のハードディスク１４５と、光ディスク１５５（たとえばＤＶＤまたはＣＤなど）を読取るためのドライブ１５０とを含む。さらに、コンピュータは入力ユニット１６０（たとえばキーボードおよびマウスなど）と、出力ユニット１６５（たとえばモニタおよびプリンタなど）とを含む。アダプタ１７０は、コンピュータをネットワーク１０５に接続するために用いられる。ブリッジ・ユニット１７５は、システム・バス１２０をローカル・バス１４０にインタフェースする。各マイクロプロセッサ１２５およびブリッジ・ユニット１７５は、情報を送信するためにシステム・バス１２０へのアクセスを要求するマスタ・エージェントとして動作し得る。アービタ１８０は、システム・バス１２０に対する相互排除によってアクセスの認可を管理する。

【００２１】

本発明の実施形態に従う配置プロセスを実施するために使用可能な主要ソフトウェア構成要素の役割を表す協調図を図２～図１２に示す。特に、これらの図面は（全体として参照２００によって示される対応の構成要素によって）システムの静的構造と、（記号「Ａ」で始まる連続的なシーケンス番号によって示される、各々が対応の動作を表す一連の交換メッセージによって）その動的挙動とを説明するものである。

【００２２】

図２から始めると、（それぞれ参照１３０ｓおよび１４５ｓによって示されるワーキング・メモリおよびマス・メモリを有する）サーバ・コンピュータ１１０は、配置マネージャ２０５、たとえばＩＢＭ社によるＩＢＭ Ｔｉｖｏｌｉ（ＩＢＭ社の登録商標）Ｐｒｏｖｉｓｉｏｎｉｎｇ Ｍａｎａｇｅｒ ｆｏｒ ＯＳ Ｄｅｐｌｏｙｍｅｎｔ（またはＴＰＭ ｆｏｒ ＯＳＤ）のＩＢＭ Ｔｉｖｏｌｉ（ＩＢＭ社の登録商標）Ｐｒｏｖｉｓｉｏｎｉｎｇ Ｍａｎａｇｅｒ ｆｏｒ Ｉｍａｇｅｓ（またはＴＰＭｆＩ）などを実行する（ＩＢＭおよびＴｉｖｏｌｉはＩＢＭ社の商標である）。配置マネージャ２０５は、システムのクライアント・コンピュータへのソフトウェア・イメージ２１０$_i$（ｉ＝１…Ｎ）の配置を自動化するために用いられる。ソフトウェア・イメージ２１０$_i$は対応するリポジトリに保存される。各ソフトウェア・イメージ２１０$_i$は、ソフトウェア・イメージ２１０$_i$内の対応するアドレス（イメージ・アドレスと呼ばれる）を有する１組のメモリ・ブロックを含む。メモリ・ブロックはあらゆる種類の情報（たとえば、オペレーティング・システムまたはアプリケーション・プログラムのいずれかに関係する１つまたはそれ以上のセクタ、ファイル、ライブラリ、ディレクトリ、その組合わせまたは部分など）を含んでもよい。

【００２３】

特に、（それぞれ参照１３０ｃおよび１４５ｃによって示されるワーキング・メモリおよびマス・メモリを有する）特定のクライアント・コンピュータ１１５に新しいソフトウェア・イメージ２１０$_i$（たとえばソフトウェア・イメージ２１０$_1$）を配置するときにはいつも、システムのオペレータ２１２がこれらのクライアント・コンピュータ１１５および新しいソフトウェア・イメージ２１０$_1$を、配置マネージャ２０５を通じて、たとえば図面に示されない別のクライアント・コンピュータ上で実行されるブラウザに接続することなどによって選択する（動作「Ａ２０１．選択」）。それに応答して、配置マネージャ２０５はクライアント・コンピュータ１１５をオンにする。その結果、クライアント・コンピュータ１１５が何らかの機能するオペレーティング・システムを有さないと仮定すると、クライアント・コンピュータ１１５は（図面に示されない）ネットワーク上でブートする。特に、クライアント・コンピュータ１１５のファームウェアに保存されてクライアント・コンピュータ１１５をオンにすると実行されるブート・ローダ、たとえば基本入出力システム（Ｂａｓｉｃ Ｉｎｐｕｔ／Ｏｕｔｐｕｔ Ｓｙｓｔｅｍ：ＢＩＯＳ）などが、何らかのブート可能な装置を見出さずに、次いでネットワーク・ブート・ローダ、たとえばそのネットワーク・アダプタに組込まれたプリブート実行環境（Ｐｒｅｂｏｏｔ

Ｅｘｅｃｕｔｉｏｎ Ｅｎｖｉｒｏｎｍｅｎｔ：ＰＸＥ）などを開始する。ネットワーク・ブート・ローダは、たとえば動的ホスト構成プロトコル（Ｄｙｎａｍｉｃ Ｈｏｓｔ Ｃｏｎｆｉｇｕｒａｔｉｏｎ Ｐｒｏｔｏｃｏｌ：ＤＨＣＰ）などに基づく動的アドレス・サービスを利用して、（ＤＨＣＰサーバとして動作する）サーバ・コンピュータ１１０からクライアント・コンピュータ１１５に対する動的アドレスを得、サーバ・コンピュータ１１０はクライアント・コンピュータ１１５のＲＡＭディスク、すなわちマス・メモリとして扱われるそのワーキング・メモリ１３０ｃの部分にダウンロードされてから開始されるネットワーク・ブートストラップ・プログラムのアドレスも提供する。たとえばウィンドウズ・プレインストール環境（Ｗｉｎｄｏｗｓ Ｐｒｅｉｎｓｔａｌｌａｔｉｏｎ Ｅｎｖｉｒｏｎｍｅｎｔ：ＷｉｎＰＥ）（ウィンドウズはマイクロソフト（Ｍｉｃｒｏｓｏｆｔ）社の商標）などのネットワーク・ブートストラップ・プログラムは最小オペレーティング・システム２１５を提供し、これは配置マネージャ２０５と対話するための配置エージェント２２０を含む（動作「Ａ２０２．ネットワーク・ブート」）。

【００２４】

配置エージェント２２０は、マス・メモリ１４５ｃを論理的に分割して複数のイメージ部分225_j（ｊ＝０…Ｍ、たとえばＭ＝２～１０）およびサービス部分２３０にすることによって、マス・メモリ１４５ｃを初期化する。（各々がソフトウェア・イメージを保存するための）イメージ部分225_jは、すべてが同じサイズ（少なくともそこにインストールされ得る最大のソフトウェア・イメージのサイズに等しい）を有し、マス・メモリ１４５ｃの始めから連続して配置されるのに対し、（サービス情報を保存するための）サービス部分２３０はそれより小さく、マス・メモリ１４５ｃの終わりに配置される。特に、サービス部分２３０は、イメージ部分225_jの状態情報を含むイメージ・テーブル２３５を保存する。たとえば、各イメージ部分225_jに対して、イメージ・テーブル２３５はその利用可能性、すなわちフリーか使用中かを示すフラグを有する記録（すべてのフラグが最初にデアサートされることによって、すべてのイメージ部分225_jがフリーであることを示す）と、（イメージ部分225_jのサイズに従って算出された）マス・メモリ１４５ｃ内のその始めからのイメージ部分225_jの位置を示すオフセットとを含む。サービス部分２３０は、現在アクティブのイメージ部分225_jのオフセットを示すオフセット・インデックス２４０も保存する（動作「Ａ２０３．初期化」）。

【００２５】

配置エージェント２２０は、次いでサーバ・コンピュータ１１０から新しいソフトウェア・イメージ210_1のブート・ブロックの組をダウンロードする。この目的のために、配置エージェント２２０は、たとえばインターネット・スモール・コンピュータ・システム・インタフェース（Ｉｎｔｅｒｎｅｔ Ｓｍａｌｌ Ｃｏｍｐｕｔｅｒ Ｓｙｓｔｅｍ Ｉｎｔｅｒｆａｃｅ：ｉＳＣＳＩ）プロトコルなどに基づいて、サーバ・コンピュータ１１０のリモート・アクセス・サーバ２４５と対話するリモート・アクセス・イニシエータとして動作する。ブート・ブロックは、配置エージェント２２０をロードするまで新しいソフトウェア・イメージ210_1のブート・シーケンスを始動するために必要とされるメモリ・ブロックを含む。たとえば、マイクロソフト・ウィンドウズにおいてブート・ブロックは、（配置エージェント２２０に加えて）マスタ・ブート記録（Ｍａｓｔｅｒ Ｂｏｏｔ Ｒｅｃｏｒｄ：ＭＢＲ）と、ブート・セクタと、ｂｏｏｔｍｇｒ．ｅｘｅファイルと、ｂｏｏｔ＼ｂｃｄファイルと、システム・レジストリと、ｗｉｎｌｏａｄ．ｅｘｅファイルと、システム・レジストリにおいて指定されるドライバ・ファイルとを含む（動作「Ａ２０４．ダウンロード」）。

【００２６】

ここで配置エージェント２２０は、最初のイメージ部分225_1が新しいソフトウェア・イメージ210_1を受取るためにフリーであることを識別する（これはイメージ・テーブル２３５においてデアサートされた対応のフラグを有する第１のイメージ部分であるため）。その結果として、配置エージェント２２０はそれに従ってイメージ・テーブル２３５を（イメージ部分225_1のフラグをアサートしてそれが使用中であることを示すこと

によって）更新し、オフセット・インデックス２４０を（イメージ・テーブル２３５に示される新しいイメージ部分２２５$_1$のオフセット、この場合にはゼロに設定してこれが現在のものであることを示すことによって）更新する。さらに配置エージェント２２０は、ブート・ブロックのイメージ・アドレスを、イメージ・テーブル２３５中の現在のイメージ部分２２５$_1$に関連する記録にセーブする。配置エージェント２２０は、現在のイメージ部分２２５$_1$（その終わりの、新しいソフトウェア・イメージ２１０$_1$を保存するために確保された部分の後ろ）に、図面には示されない位置マップも作成する。ソフトウェア・イメージ２１０$_1$の各メモリ・ブロックに対して、位置マップはマス・メモリ１４５ｃにおけるその利用可能性を示すフラグを含む。すべてのフラグが最初にデアサートされることによって、マス・メモリ１４５ｃにおけるどのメモリ・ブロックもまだ利用可能でないことが示される（動作「Ａ２１２．構成」、以下に説明されるスキップされたシーケンス番号に対応する動作を伴う）。次いで配置エージェント２２０は、新しいソフトウェア・イメージ２１０$_1$のブート・ブロックを、現在のイメージ部分２２５$_1$に、すなわち対応のイメージ・アドレスに現在のイメージ部分２２５$_1$のオフセットを加えたものに等しいマス・メモリ１４５ｃ中のアドレス（メモリ・アドレスと呼ばれる）を有するマス・メモリ１４５ｃのメモリ位置に保存する。これらのブート・ブロックはグレーで示され、参照２５０$_1$で表示されている。同時に、位置テーブル中の対応するフラグがアサートされる（それによって、マス・メモリ１４５ｃにおいて現在そのブート・ブロックが利用可能であることが示される）。この場合には現在のイメージ部分２２５$_1$のオフセットがゼロであるため、ブート・ブロック２５０$_1$はマス・メモリ１４５ｃ中で、ブート・シーケンスの際に見出されることが期待されるまさにその場所に配置される（動作「Ａ２１３．保存」）。ここで配置エージェント２２０はクライアント・コンピュータ１１５をオフにしてからオンにする。

【００２７】

したがって、図３に示されるとおり、クライアント・コンピュータ１１５はマス・メモリ１４５ｃから正常にブートする。実際に、クライアント・コンピュータ１１５がオンになると実行されるクライアント・コンピュータ１１５のブート・ローダが今、マス・メモリ１４５ｃをブート可能な装置として識別するため、それは自身のブート・ブロック２５０$_1$から局所的にブートする。このやり方で、ブート・ブロック２５０$_1$に対応する新しいソフトウェア・イメージ２１０$_1$の実際のオペレーティング・システムの部分（参照２５５$_1$によって示される）および配置エージェント２２０がワーキング・メモリ１３０ｃにロードされる。たとえば、マイクロソフト・ウィンドウズにおいてＢＩＯＳはＭＢＲをロードし、ＭＢＲはブート・セクタをロードし、ブート・セクタはｂｏｏｔｍｇｒ．ｅｘｅファイルを見出して始動し、ｂｏｏｔｍｇｒ．ｅｘｅはｂｏｏｔ＼ｂｃｄファイルを見出して読取ることによってメモリ位置を決定し、次いでシステム・レジストリと、ｗｉｎｌｏａｄ．ｅｘｅファイルと、システム・レジストリにおいて指定されるドライバ・ファイルとをロードし、ｗｉｎｌｏａｄ．ｅｘｅは配置エージェント２２０を始動する。新しいソフトウェア・イメージ２１０$_1$のサイズが現在のイメージ部分２２５$_1$のサイズと異なる（すなわちそれより小さい）とき、配置エージェント２２０はそのサイズ変更も行って、現在のイメージ部分２２５$_1$全体が占有されるようにする。この動作は、オペレーティング・システム２５５$_1$の図面には示されないファイル－システム構造の更新しか必要としないため、非常に速い（動作「Ａ２１４．ローカル・ブート」）。ここで、クライアント・コンピュータ１１５の動作中に新しいソフトウェア・イメージ２１０$_1$の選択されたメモリ・ブロックにアクセスするすべての要求は、（オペレーティング・システム２５５$_1$の図面には示されない標準的なファイル－システム・ドライバをオーバーライドする）配置エージェント２２０のストリーミング・ドライバによって提供される。

【００２８】

特に、ファイル－システム・ドライバは、たとえば図面には示されないアプリケーション・プログラムなどから、選択されたメモリ・ブロックにアクセスする（すなわちそれを読取る）ための要求を受取る（動作「Ａ２１５．アクセス要求」）。この要求は配置エー

ジェント２２０に送られ、配置エージェント２２０は選択されたメモリ・ブロックが現在のイメージ部分２２５$_1$において利用可能かどうかを確認する（位置マップに示されるとおり）。選択されたメモリ・ブロックが現在のイメージ部分２２５$_1$において利用可能でないとき（すなわち位置マップにおけるそのフラグがデアサートされているとき）、配置エージェント２２０はその要求をオペレーティング・システム２５５$_1$の図面には示されないリモート・アクセス・ドライバ（問題となる例におけるｉＳＣＳＩイニシエータとして動作する）に送る。リモート・アクセス・ドライバは、リモート・アクセス・サーバ２４５を通じて（サーバ・コンピュータ１１０上の）ソフトウェア・イメージ２１０$_1$から選択されたメモリ・ブロックをダウンロードする。リモート・アクセス・ドライバは、次いで選択されたメモリ・ブロックを配置エージェント２２０に戻す（動作「Ａ２１６ａ．ダウンロード」）。配置エージェント２２０は（オペレーティング・システム２５５の図面には示されない物理的ディスク・ドライバを通じて）現在のイメージ部分２２５$_1$に選択されたメモリ・ブロックを保存する。特に、選択されたメモリ・ブロックは、そのイメージ・アドレスに現在のイメージ部分２２５$_1$のオフセットを加えたものに等しいメモリ・アドレスを有するメモリ位置に保存される（この場合にはゼロであるそのオフセットは、オフセット・テーブル２４０から抽出される）。さらに、配置エージェント２２０はそれに従って位置マップを更新し、すなわち対応するフラグをアサートすることによって、選択されたメモリ・ブロックが利用可能であることを示す（動作「Ａ２１７．保存」）。ここで配置エージェント２２０は選択されたメモリ・ブロックをファイル－システム・ドライバに戻し、ファイル－システム・ドライバは次にそれをアプリケーション・プログラムに戻す（動作「Ａ２１８．戻す」）。

【００２９】

逆に、図４に示されるとおり、もし選択されたメモリ・ブロックが現在のイメージ部分２２５$_1$においてすでに利用可能であれば（すなわち位置マップにおけるそのフラグがアサートされていれば）、配置エージェント２２０はその要求を物理的ディスク・ドライバに送る。物理的ディスク・ドライバは、現在のイメージ部分２２５$_1$から、すなわちそのイメージ・アドレスに現在のイメージ部分２２５$_1$のオフセット（この場合はゼロ）を加えたものに等しいメモリ・アドレスを有するメモリ位置から選択されたメモリ・ブロックを直接検索する。物理的ディスク・ドライバは、次いで選択されたメモリ・ブロックを配置エージェント２２０に戻す（動作「Ａ２１６ｂ．検索」）。この場合にも、配置エージェント２２０は選択されたメモリ・ブロックをファイル－システム・ドライバに戻し、ファイル－システム・ドライバは次にそれをアプリケーション・プログラムに戻す（同じ動作「Ａ２１８．戻す」）。

【００３０】

上述のストリーミング技術は、クライアント・コンピュータ１１５を非常に短時間のうちに使用可能にする。すなわち、たとえ配置プロセスがまだ進行中であっても、新しいソフトウェア・イメージ２１０$_1$のブート・ブロックがマス・メモリ１４５ｃに保存された直後（たとえば、１０～２００Ｍバイトの典型的なサイズのブート・ブロックに対しては１～２分後）に使用可能になる。クライアント・コンピュータ１１５の動作は完全に正常であり（通常どおりマス・メモリ１４５ｃから直接ブートする）、それはマス・メモリ１４５ｃにおけるソフトウェア・イメージ２１０$_1$の他のメモリ・ブロックが利用可能かどうかには関係せず、サーバ・コンピュータ１１０からなおもダウンロードされるべきであるメモリ・ブロックにアクセスするときにクライアント・コンピュータ１１５の性能がわずかに効率低下するだけである。さらに、クライアント・コンピュータ１１５を使用可能にするために必要な時間は、新しいソフトウェア・イメージ２１０$_1$のサイズに関係しない。ネットワークの使用量も時間とともに（たとえば対数法則によって）減少するが、これは一旦メモリ・ブロックがアクセスされると、クライアント・コンピュータ１１５においてより多くのメモリ・ブロックがすでに利用可能となるからである。このことに関して、このストリーミング技術は、ソフトウェア・イメージをオン・デマンドで提供するために当該技術分野において公知であるものとは無関係であることに注意すべきである。実際

のところ、公知のストリーミング技術において、ソフトウェア・イメージのメモリ・ブロックは即時使用のためにのみクライアント・コンピュータにダウンロードされる。しかし、これらのメモリ・ブロックはクライアント・コンピュータに永続的に保存されない（すなわちそれらは使用後、およびいずれの場合にもクライアント・コンピュータがオフにされた後に消滅する）ため、それらを次に使用するためにはそれらを再びダウンロードする必要がある。その結果として、クライアント・コンピュータをサーバ・コンピュータから切断することは決してできない。実際には、たとえメモリ・ブロックが先取りされていても、それらは次の使用（の可能性）までしかクライアント・コンピュータ上に存続しない。同様に、たとえメモリ・ブロックに対するローカル・キャッシュが実施されていても、再使用のためにローカル・キャッシュに残るのはほんのわずかなメモリ・ブロックだけである（いずれの場合にも、ローカル・キャッシュ中の最も長時間使用されていないメモリ・ブロックは、新しいメモリ・ブロックを保存するために最終的に追い出される）。

【００３１】

図５を参照して、もし選択されたメモリ・ブロックが書込のために要求されれば、配置エージェント２２０はその要求を物理的ディスク・ドライバに送る。物理的ディスク・ドライバは現在のイメージ部分２２５$_1$において、すなわちそのイメージ・アドレスに現在のイメージ部分２２５$_1$のオフセット（この場合はゼロ）を加えたものに等しいメモリ・アドレスを有するメモリ位置において、選択されたメモリ・ブロックを直接更新し、その読取の後それは常時利用可能となる（動作「Ａ２１９．更新」）。したがって、たとえ配置プロセスがまだ進行中であっても、現在のイメージ部分２２５$_1$は（あたかも新しいソフトウェア・イメージ２１０$_1$がクライアント・コンピュータ１１５にすでに完全に配置されていたかのように）正常に更新され得る。

【００３２】

まったく非同期的に、配置エージェント２２０は、サーバ・コンピュータ１１０、クライアント・コンピュータ１１５、またはそれらを接続するネットワーク、あるいはその組合わせの作業負荷を周期的に確認する（たとえば１０～１００ｍｓごと）。作業負荷が予め定められた閾値より低い（たとえば、サーバ・コンピュータ１１０またはクライアント・コンピュータ１１５あるいはその両方において何の動作も行われておらず、ネットワーク内のトラフィックが少ないために、この時点で対応するリソースの利用が少ないことを示す）とき、配置エージェント２２０は上述と同じ動作を繰り返すことによって、現在のイメージ部分２２５$_1$においてまだ利用可能でない新しいメモリ・ブロック（たとえば、位置マップにおけるそのフラグがデアサートされている最初のメモリ・ブロック）をダウンロードする（動作「Ａ２２０．ダウンロード」）。上記と同様に、配置エージェント２２０はこの新しいメモリ・ブロックを（物理的ディスク・ドライバを通じて）現在のイメージ部分２２５$_1$に、すなわちそのイメージ・アドレスに現在のイメージ部分２２５$_1$のオフセット（この場合はゼロ）を加えたものに等しいメモリ・アドレスを有するメモリ位置に保存する。さらに、配置エージェント２２０はそれに従って位置マップを更新して、新しいメモリ・ブロックが利用可能であることを示す（動作「Ａ２２１．保存」）。

【００３３】

このやり方で、図６に示されるとおり、最後に新しいソフトウェア・イメージ２１０$_1$のすべてのメモリ・ブロックが（たとえそれらがまったく使用されなくても）ダウンロードされることを確実にすることができる。

【００３４】

あらゆる時点で、オペレータ２１２は、配置マネージャ２０５を通じて同じクライアント・コンピュータ１１５に配置されるべき別の新しいソフトウェア・イメージ２１０$_i$（たとえばソフトウェア・イメージ２１０$_2$）を選択してもよい（動作「Ａ２０５．選択」）。それに応答して、上記と同様に配置エージェント２２０は、リモート・アクセス・サーバ２４５を通じてサーバ・コンピュータ１１０から新しいソフトウェア・イメージ２１０$_2$のブート・ブロックをダウンロードする（動作「Ａ２０６．ダウンロード」）。

【００３５】

図７に移って、配置エージェント２２０は、新しいソフトウェア・イメージ２１０$_2$のブート位置（すなわち新しいソフトウェア・イメージ２１０$_2$のブート・ブロックのイメージ・アドレスに等しいメモリ・アドレスを有するメモリ位置）に保存される最初のイメージ部分２２５$_1$のメモリ・ブロックを再配置する。特に、これらのメモリ・ブロックはサービス部分２３０（の専用の再配置部分）に移され、それらは再配置メモリ・ブロック２６０$_2$と呼ばれる（動作「Ａ２１０．再配置」）。次いで配置エージェント２２０は、新しいソフトウェア・イメージ２１０$_2$のブート・ブロックをそれらのブート位置に保存する。それらのブート・ブロックは濃いグレーで示され、参照２５０$_2$で表示されている。このやり方で、ここでもブート・ブロック２５０$_2$は、マス・メモリ１４５ｃ中の、ブート・シーケンスの際に見出されることが期待されるまさにその場所に配置される。しかし、このことは最初のイメージ部分２２５$_1$におけるいかなる情報の損失ももたらさない。なぜなら、オーバーライドされた対応のメモリ・ブロックはサービス部分２３０にセーブされているからである（動作「Ａ２１１．保存」）。

【００３６】

次いで、配置プロセスは上記とまったく同様に続く。特に、配置エージェント２２０は新しいソフトウェア・イメージ２１０$_2$を受取るためにフリーであるイメージ部分２２５$_i$（すなわち、問題となる例におけるイメージ部分２２５$_2$）を識別する。その結果、配置エージェント２２０はそれに従って（イメージ部分２２５$_2$のフラグをアサートして使用中であることを示すことによって）イメージ・テーブル２３５を更新し、（イメージ部分２２５$_2$のオフセットに設定してそれが現在のものであることを示すことによって）オフセット・インデックス２４０を更新する。さらに配置エージェント２２０は、ブート・ブロック２５０$_2$のイメージ・アドレスを、イメージ・テーブル２３５中の現在のイメージ部分２２５$_2$に関連する記録にセーブする。配置エージェント２２０は、現在のイメージ部分２２５$_2$に（図面には示されない）対応する位置マップも作成する（同じ動作「Ａ２１２．構成」）。次いで配置エージェント２２０は、新しいソフトウェア・イメージ２１０$_2$のブート・ブロックを、現在のイメージ部分２２５$_2$に、すなわちそれらのイメージ・アドレスに現在のイメージ部分２２５$_2$のオフセットを加えたものに等しいメモリ・アドレスを有するマス・メモリ１４５ｃのメモリ位置に保存する。それらのブート・ブロックはグレーで示される（同じ動作「Ａ２１３．保存」）。ここで配置エージェント２２０はクライアント・コンピュータ１１５をオフにしてからオンにする。

【００３７】

したがって、図８に示されるとおり、クライアント・コンピュータ１１５のブート・ローダはそのブート・ブロック２５０$_2$からブートする。このやり方で、ブート・ブロック２５０$_2$に対応する新しいソフトウェア・イメージ２１０$_2$のオペレーティング・システムの部分（参照２５５$_2$で示される）および配置エージェント２２０がワーキング・メモリ１３０ｃにロードされる（同じ動作「Ａ２１４．ローカル・ブート」）。ここでも、クライアント・コンピュータ１１５の動作中にソフトウェア・イメージ２１０$_2$の選択されたメモリ・ブロックにアクセスするすべての要求は、配置エージェント２２０のストリーミング・ドライバによって提供される。

【００３８】

特に、新しいソフトウェア・イメージ２１０$_2$の選択されたメモリ・ブロックにアクセスする（それを読取る）ための要求（同じ動作「Ａ２１５．アクセス要求」）に応答して、もし選択されたメモリ・ブロックが現在のイメージ部分２２５$_2$において利用可能でなければ、それはサーバ・コンピュータ１１０上のソフトウェア・イメージ２１０$_2$からダウンロードされ（同じ動作「Ａ２１６ａ．ダウンロード」）、現在のイメージ部分２２５$_2$に保存され（同じ動作「Ａ２１７．保存」）、次いで戻される（同じ動作「Ａ２１８．戻す」）。

【００３９】

逆に、図９に示されるとおり、もし選択されたメモリ・ブロックが現在のイメージ部分２２５$_2$においてすでに利用可能であれば、それは直接検索され（同じ動作「Ａ２１６ｂ

．検索」）、次いで戻される（同じ動作「Ａ２１８．戻す」）。
【００４０】
図１０を参照して、選択されたメモリ・ブロックが書込のために要求されたのであれば、それは現在のイメージ部分２２５$_2$において更新される（同じ動作「Ａ２１９．更新」）。作業負荷が低いとき、現在のイメージ部分２２５$_2$においてまだ利用可能でない新しいメモリ・ブロックがダウンロードされて（同じ動作「Ａ２２０．ダウンロード」）、現在のイメージ部分２２５$_2$に保存される（同じ動作「Ａ２２１．保存」）。
【００４１】
（完全にダウンロードされたソフトウェア・イメージ２１０$_2$を示す）図１１を参照して、オペレータ２１２はあらゆる時点で、配置マネージャ２０５を通じて同じクライアント・コンピュータ１１５に配置されるべき別の新しいソフトウェア・イメージ２１０$_i$（たとえばソフトウェア・イメージ２１０$_3$）を選択してもよい（動作「Ａ２０７．選択」）。それに応答して、上記と同様に配置エージェント２２０は、リモート・アクセス・サーバ２４５を通じてサーバ・コンピュータ１１０から新しいソフトウェア・イメージ２１０$_3$のブート・ブロックをダウンロードする（動作「Ａ２０８．ダウンロード」）。ここで配置エージェント２２０は、再配置されたメモリ・ブロック２６０$_2$をサービス部分２３０から最初のイメージ部分２２５$_1$に、すなわち現在のイメージ部分２２５$_2$のブート位置に（イメージ・テーブル２３５の対応の記録に示されるとおり）復元する。この動作は、オーバーライドされた現在のイメージ部分２２５$_2$のブート・ブロック２５０$_2$におけるいかなる情報の損失ももたらさない。なぜなら、（上に説明したとおりにクライアント・コンピュータ１１５の動作中に更新された可能性のある）それらの最新の値が現在のイメージ部分２２５$_2$に保存されているからである（動作「Ａ２０９．復元」）。
【００４２】
次いで、配置プロセスは上記とまったく同様に続く。特に、図１２に示されるとおり、配置エージェント２２０は新しいソフトウェア・イメージ２１０$_3$のブート位置に保存される最初のイメージ部分２２５$_1$のメモリ・ブロックをサービス部分２３０に再配置し、これらのメモリ・ブロックは参照２６０$_3$で示される（同じ動作「Ａ２１０．再配置」）。ここで配置エージェント２２０は、新しいソフトウェア・イメージ２１０$_3$のブート・ブロックをそれらのブート位置に保存する。それらのブート・ブロックは濃いグレーで示され、参照２５０$_3$で表示されている（同じ動作「Ａ２１１．保存」）。次いで配置エージェント２２０は、ソフトウェア・イメージ２１０$_2$を受取るためにフリーであるイメージ部分２２５$_i$（すなわちイメージ部分２２５$_3$）を識別する。その結果、配置エージェント２２０はそれに従って（イメージ部分２２５$_3$のフラグをアサートして使用中であることを示すことによって）イメージ・テーブル２３５を更新し、（イメージ部分２２５$_3$のオフセットに設定してそれが現在のものであることを示すことによって）オフセット・インデックス２４０を更新する。さらに配置エージェント２２０は、ブート・ブロック２５０$_3$のイメージ・アドレスを、イメージ・テーブル２３５中の現在のイメージ部分２２５$_3$に関連する記録にセーブする。配置エージェント２２０は、現在のイメージ部分２２５$_3$に（図面には示されない）対応する位置マップも作成する（同じ動作「Ａ２１２．構成」）。次いで配置エージェント２２０は、新しいソフトウェア・イメージ２１０$_3$のブート・ブロックを、現在のイメージ部分２２５$_3$に保存する。それらのブート・ブロックはグレーで示される（同じ動作「Ａ２１３．保存」）。ここで配置エージェント２２０は、クライアント・コンピュータ１１５をオフにしてからオンにすることでそれを自身のブート・ブロック２５０$_3$からブートさせ、それによってブート・ブロック２５０$_3$に対応するソフトウェア・イメージ２１０$_3$のオペレーティング・システムの部分、および新しいソフトウェア・イメージ２１０$_3$のあらゆる選択されたメモリ・ブロックにアクセスするすべての要求を提供する配置エージェントをワーキング・メモリ１３０ｃにロードする（同じ動作「Ａ２１４．ローカル・ブート」）。
【００４３】
いずれの場合にも、クライアント・コンピュータへの１つまたはそれ以上のソフトウェ

ア・イメージの配置が一旦完了すると、それはもはやサーバ・コンピュータを何ら必要とすることなく自律的に作業できる。しかし、配置プロセスの際に用いられたのと同じストリーミング技術を利用して、クライアント・コンピュータのスナップショットを作成することもできる（各スナップショットは、特定の時点におけるそのソフトウェア・イメージの整合性のあるバックアップ・コピーによって形成される）。

【００４４】

特に、図１３～図１７は、本発明の実施形態に従うスナップショット・プロセスを実施するために使用可能な主要ソフトウェア構成要素（全体として参照３００によって示される）の役割を表す協調図を示す。

【００４５】

図１３から始めて、たとえば、クライアント・コンピュータ１１５においてイメージ部分２２５$_1$および２２５$_2$に２つのソフトウェア・イメージが保存されているという状況を考えてみる。イメージ部分２２５$_2$が現在のものであり、そのブート・ブロック２５０$_2$は最初のイメージ部分２２５$_1$におけるそのブート位置にも保存される（最初のイメージ部分２２５$_1$の対応の再配置メモリ・ブロック２６０$_2$はサービス部分２３０にセーブされている）。クライアント・コンピュータ１１５のユーザは、対応するソース・イメージ部分２２５$_i$、たとえばソース・ソフトウェア・イメージ２１０$_1$を保存するソース・イメージ部分２２５$_1$などにソース・ソフトウェア・イメージの新しいスナップショットを作成するためのコマンドを提出してもよい（動作「Ａ３０１．作成」）。それに応答して、配置エージェント２２０は最初にソース・ソフトウェア・イメージ２１０$_1$を受取るためにフリーである目標イメージ部分２２５$_i$（たとえば、問題となる例におけるイメージ部分２２５$_3$）を識別する。次いで配置エージェント２２０は、ソース・ソフトウェア・イメージ２１０$_1$全体を目標イメージ部分２２５$_3$にコピーする。より具体的には、ソース・ソフトウェア・イメージ２１０$_1$の（対応する現在のイメージ・アドレスを有する）現在のメモリ・ブロックすべてに対して、もしソース・イメージ部分２２５$_1$が（この場合のように）最初のものであれば、配置エージェント２２０は現在のメモリ・ブロックがサービス部分２３０に再配置された（すなわち、イメージ・テーブル２３５におけるその記録に示されるとおり、現在のイメージ・アドレスが現在のソフトウェア・イメージ２２５$_2$のブート位置の１つのイメージ・アドレスに等しい）かどうかを確認する。もしそうであれば、対応する再配置メモリ・ブロック２６０$_2$が目標イメージ部分２２５$_3$の、現在のイメージ・アドレスに目標イメージ部分２２５$_3$のオフセットを加えたものに等しいメモリ・アドレスを有するメモリ位置にコピーされる。逆に（すなわち、現在のメモリ・ブロックが再配置されていないとき、またはソース・イメージ部分が最初のものではないときは常に）、現在のメモリ・ブロックがソース・イメージ部分２２５$_1$から目標イメージ部分２２５$_3$にコピーされる。すなわち、現在のイメージ・アドレスにソース・イメージ部分２２５$_2$のオフセットを加えたものに等しいメモリ・アドレスを有するメモリ位置から、現在のイメージ・アドレスに目標イメージ部分２２５$_3$のオフセットを加えたものに等しいメモリ・アドレスを有するメモリ位置にコピーされる（動作「Ａ３０２．コピー」）。一旦ソース・ソフトウェア・イメージ２１０$_1$全体が目標イメージ部分２２５$_3$にコピーされると、配置エージェント２２０はそれに従って、目標イメージ部分２２５$_3$のフラグをアサートして使用中であることを示し、ソース・ソフトウェア・イメージ２１０$_1$のブート・ブロックの同じイメージ・アドレスをセーブすることによって、イメージ・テーブル２３５を更新する（動作「Ａ３０３．更新」）。したがって、ソフトウェア・イメージのスナップショットを作成する動作はかなり遅い（その長さはコピーされるソース・ソフトウェア・イメージのサイズに比例する）。

【００４６】

図１４に移って、クライアント・コンピュータ１１５のユーザは、（対応する古いイメージ部分２２５$_i$中の）古いソフトウェア・イメージを削除するためのコマンドを配置エージェント２２０に提出してもよい。古いソフトウェア・イメージの削除後のクライアント・コンピュータ１１５の動作の継続性を確実にするために、古いイメージ部分２２５$_i$

は現在のイメージ部分２２５$_2$とは異なる、たとえばイメージ部分２２５$_1$などであるべきである（動作「Ａ３０３．削除」）。それに応答して、配置エージェント２２０は古いイメージ部分２２５$_1$のフラグをアサートし、それがフリーであることを示すことによってイメージ・テーブル２３５を単に更新する（動作「Ａ３０４．更新」）。したがって、ソフトウェア・イメージを削除する動作は非常に速い（そのメモリ・ブロックに対する実際の動作を何も必要としないため）。さらに、たとえ削除されるイメージ部分が最初のものであっても、この動作は何の問題も起こさない（現在のイメージ部分２２５$_2$に保存されるブート・ブロックは影響されないため）。

【００４７】

図１５を参照して、代わりにクライアント・コンピュータ１１５のユーザは、現在のソフトウェア・イメージ２２５$_2$から別（前）のソフトウェア・イメージ２２５$_i$へ、たとえばソフトウェア・イメージ２２５$_3$などへ切換えるためのコマンドを提出してもよい（動作「Ａ３０５．切換」）。それに応答して、上記と同様に配置エージェント２２０は再配置メモリ・ブロック２６０$_2$をサービス部分２３０から最初のイメージ部分２２５$_1$に、すなわち現在のイメージ部分２２５$_2$のブート位置に復元する（動作「Ａ３０６．復元」）。

【００４８】

図１６に移って、配置エージェント２２０は、前のソフトウェア・イメージ２１０$_3$のブート位置に保存される最初のイメージ部分２２５$_1$のメモリ・ブロックをサービス部分２３０に再配置し、それらのメモリ・ブロックは参照２６０$_3$で示される（動作「Ａ３０７．再配置」）。次いで配置エージェント２２０は前のソフトウェア・イメージ２１０$_3$のブート・ブロックをそれらのブート位置にコピーする。それらのブート・ブロックはグレーで示され、参照２５０$_3$で表示されている（動作「Ａ３０８．コピー」）。次いで配置エージェント２２０は、オフセット・インデックス２４０を前のイメージ部分２２５$_3$のオフセットに設定してそれが現在のものとなったことを示すことによって、オフセット・インデックス２４０を更新できる（動作「Ａ３０９．更新」）。ここで配置エージェント２２０は、クライアント・コンピュータ１１５をオフにしてからオンにすることでそれを自身のブート・ブロック２５０$_3$からブートさせ、それによってブート・ブロック２５０$_3$に対応する前のソフトウェア・イメージ２１０$_3$のオペレーティング・システムの部分、および配置エージェントをワーキング・メモリ１３０ｃにロードする（動作「Ａ３１０．リブート」）。

【００４９】

図１７を考慮すると、いずれの場合にも、クライアント・コンピュータ１１５の動作中に現在のソフトウェア・イメージ２１０$_3$の選択されたメモリ・ブロックにアクセスするすべての要求は、ここでも（オペレーティング・システム２５５$_3$の図面には示されない標準的なファイルーシステム・ドライバをオーバーライドする）配置エージェント２２０のストリーミング・ドライバによって提供される。

【００５０】

特に、ファイルーシステム・ドライバは、たとえば図面には示されないアプリケーション・プログラムなどから、選択されたメモリ・ブロックにアクセスする（それを読取るかまたは書込む）ための要求を受取る（動作「Ａ３１１．アクセス要求」）。この要求は配置エージェント２２０に送られ、配置エージェント２２０は（オペレーティング・システム２５５$_3$の図面には示されない物理的ディスク・ドライバを通じて）現在のイメージ部分２２５$_3$中の選択されたメモリ・ブロックに、すなわち選択されたメモリ・ブロックのイメージ・アドレスに現在のイメージ部分２２５$_3$のオフセットを加えたものに等しいメモリ・アドレスを有するメモリ位置に直接アクセスする（動作「Ａ３１２．アクセス」）。

【００５１】

上述の技術は、クライアント・コンピュータ上の複数のソフトウェア・イメージを非常に簡単なやり方で管理することを可能にする。さらに、異なるソフトウェア・イメージは

互いに完全に分離される。実際のところ、各ソフトウェア・イメージは対応するイメージ部分にしかアクセスできない（それによって他のイメージ部分を損なうあらゆる危険を防止する）。

【００５２】

この結果は、仮想化インフラストラクチャを何ら必要とせずに達成される。したがって、クライアント・コンピュータの性能は悪影響を受けない。実際に、この場合にはソフトウェア・イメージ（すなわちそれらが保存されるマス・メモリ）のみが仮想化される。反対に、オペレーティング・システムはクライアント・コンピュータ上でネイティブに実行され続ける。

【００５３】

特に、このことによってスナップショットを非常に速く復元できる（それはすでにクライアント・コンピュータ上で利用可能であるため）。さらに、ネットワーク接続を何ら必要とせずに所望の結果が達成され得る。

【００５４】

図１８に移ると、本発明の実施形態に従う解決策において用いられる一般的なソフトウェア・イメージの準備プロセスを実施するために使用可能な主要ソフトウェア構成要素（全体として参照４００によって示される）の役割を表す協調図が示される。特に、この準備プロセスは、（上述の配置プロセスまたはスナップショット・プロセスの際に用いられる）ソフトウェア・イメージのブート・ブロックを識別することを目的とする。

【００５５】

この目的のために、サーバ・コンピュータ１１０はマスタ・ソフトウェア・イメージ（または単にマスタ・イメージ）４０５のリポジトリを含む。各マスタ・イメージ４０５は、（たとえば、それが前にインストールされたドナー・クライアント・コンピュータのハードディスクの内容を取込むことによって作成される）対応するソフトウェア・イメージの基本バージョンを提供し、そこではドナー・クライアント・コンピュータのあらゆる構成に関する特定の内容（たとえばドライバおよびレジストリ設定など）が除去されている。サーバ・コンピュータ１１０はモデル４１０のリポジトリも含む。各モデル４１０は代わりにクライアント・コンピュータの対応する構成に対して特定的な内容を含む。

【００５６】

オペレータ２１２は、配置マネージャ２０５を通じて（対応する補助クライアント・コンピュータ１１５によって表される）クライアント・コンピュータの特定のタイプに対するソフトウェア・イメージ（選択されたマスタ・イメージ４０５および選択されたモデル４１０を含む）を選択する（動作「Ａ４０１．選択」）。それに応答して、配置マネージャ２０５は選択されたソフトウェア・イメージの識別子を送ることによって、補助クライアント・コンピュータ１１５上の配置エージェント２２０をウェイクアップする（動作「Ａ４０２．ウェイクアップ」）。その結果、配置エージェント２２０は、リモート・アクセス・サーバ２４５を通じて遠隔的にアクセスするためのリモート・ディスクとして（すなわち問題となる例におけるｉＳＣＳＩイニシエータとして動作することによって）選択されたソフトウェア・イメージをマウントする。その結果、補助クライアント・コンピュータ１１５による排他的アクセスのための一時的ソフトウェア・イメージ（または単に一時的イメージ）４１５が作成される。一時的イメージ４１５は、単に選択されたマスタ・イメージ４０５および選択されたモデル４１０のメモリ・ブロックを指すインデックス構造によって、つまりその何らかのコピーを作成することなく定められる。一時的イメージ４１５は有効にされたブロック追跡機能とともにマウントされることによって、アクセスされる一時的イメージ４１５のあらゆるメモリ・ブロックのイメージ・アドレスが追跡される（動作「Ａ４０３．マウント」）。

【００５７】

ここで配置エージェント２２０は、（配置エージェントのロードまでの）一時的イメージ４１５上の補助クライアント・コンピュータ１１５のブート・シーケンスをシミュレートする。たとえば、マイクロソフト・ウィンドウズにおいて配置エージェント２２０はＭ

ＢＲと、ブート・セクタと、ｂｏｏｔｍｇｒ．ｅｘｅファイルと、ｂｏｏｔ＼ｂｃｄファイルと、システム・レジストリと、ｗｉｎｌｏａｄ．ｅｘｅファイルと、システム・レジストリにおいて指定されるドライバ・ファイルと、配置エージェントとを読取る（動作「Ａ４０４．シミュレートされたブート」）。シミュレートされたブート・シーケンスが完了すると、配置エージェント２２０は一時的イメージ４１５をアンマウントする（動作「Ａ４０５．アンマウント」）。次いで配置エージェント２２０は一時的イメージ４１５を配置マネージャ２０５にコミットする（動作「Ａ４０６．コミット」）。それに応答して、配置マネージャ２０５は（単にそのインデックス構造によって定められる）一時的イメージ４１５から新しいソフトウェア・イメージ（または単に新しいイメージ）４２０を構築する。さらに、新しいイメージ４２０はシミュレートされたブート手順の間にアクセスされたメモリ・ブロックのリストに関連しており、このメモリ・ブロックは対応するブート・ブロックを定める（動作「Ａ４０７．構築」）。

【００５８】

当然ながら、ローカルおよび特定の要求を満たすために、当業者は上述の解決策に多くの論理的または物理的あるいはその両方の修正および変更を適用してもよい。より特定的には、この解決策はその１つまたはそれ以上の実施形態を参照してある程度詳細に説明されているが、（たとえば数値および構成に関して）その形および詳細におけるさまざまな省略、置換および変更、ならびに他の実施形態が可能であることが理解されるべきである。特に、本発明のより完全な理解を提供するために先の記載に示されている特定の詳細なしに、本発明の異なる実施形態が実施されてもよい。反対に、不必要な詳細によって説明を曖昧にしないために、周知の特徴が省略または簡略化されていることがある。さらに、開示される解決策のあらゆる実施形態と関連して記載される特定の構成要素または方法ステップあるいはその両方は、一般的な設計選択の問題としてあらゆる他の実施形態に組込まれてもよいことが明確に意図されている。

【００５９】

たとえば、（より多くのステップもしくはその部分と同じ機能を有する類似のステップを用いること、必須でないいくつかのステップを除去すること、またはさらなる任意のステップを加えることによって）もし同じ解決策が同等の方法によって実施されれば、類似の考察が適用される。さらに、これらのステップは（少なくとも部分的に）異なる順序で行われても、同時または交互に行われてもよい。

【００６０】

ソフトウェア・イメージは、あらゆるソフトウェア・プログラム（たとえば、何らかのアプリケーション・プログラムを有さないオペレーティング・システムのみ）を含んでもよい。さらに、提案される技術は物理的コンピュータにおける使用のために特定的に設計されているが、仮想マシンにおけるその適用も除外されない。加えて、アクセス機能を提供するためにあらゆる同等の構造が用いられてもよい。いずれの場合にも、あらゆるブート・シーケンスを有するあらゆるその他のオペレーティング・システムに同じ技術が適用されてもよい。たとえば、Ｌｉｎｕｘ（リーナス・トーバルズ（Ｌｉｎｕｓ　Ｔｏｒｖａｌｄｓ）の商標）において、ブート・ブロックは（配置エージェントに加えて）ＧＲＢＵブート・ローダを含むＭＢＲと、カーネルおよびｉｎｉｔｒｄファイル・システムを含む／ｂｏｏｔディレクトリとを含む。この場合、ブート・シーケンスの際にＢＩＯＳがＧＲＢＵを含むＭＢＲをロードし、ＧＲＢＵは／ｂｏｏｔディレクトリを見出してカーネルおよびｉｎｉｔｒｄファイル・システムをロードし、ＧＲＢＵはカーネルをブートし、カーネルはｉｎｉｔｒｄファイル・システムを始動し、ｉｎｉｔｒｄファイル・システムは配置エージェントを始動する。

【００６１】

クライアント・コンピュータが別のソフトウェア・イメージに切換えられたときに再配置メモリ・ブロックを復元するステップは、厳密に必要なものではない。たとえば、最初のものと異なるすべてのソフトウェア・イメージのブート位置のメモリ・ブロックを再配置部分に再配置し、次いでクライアント・コンピュータが最初のソフトウェア・イメージ

に切換えられたときにのみそれらすべてを同時に復元することも可能である（それによって切換動作が速くなるが、マス・メモリ・スペースの無駄が生じる）。

【００６２】

いずれの場合にも、ブート位置はマス・メモリ内のいずれの位置に配置されてもよい。

【００６３】

イメージ部分は同等のやり方で管理されてもよい。たとえば、（作成されるときに動的に定められる）異なるサイズを有するイメージ部分を有することが可能である。

【００６４】

さらに、イメージ部分の状態情報を管理するために同等の構造が用いられてもよい（たとえば、その各イメージ区画がその利用可能性の標識を直接含む）。

【００６５】

先行する説明では、１つまたはそれ以上のソフトウェア・イメージが完全にインストールされた（すなわちネットワーク接続が必要でない）ときにクライアント・コンピュータ上で実施されるスナップショット・プロセスを参照したが、（クライアント・コンピュータがそこに復帰したときに再始動するソフトウェア・イメージの配置を伴う）クライアント・コンピュータのマス・メモリに部分的にのみ保存されたソフトウェア・イメージにも同じ技術が適用されてもよい。

【００６６】

代替的な実施においては、配置エージェントが、対応するオペレーション・システムにそのサイズが現在のソフトウェア・イメージのサイズと等しいと信じさせることによって、現在のイメージ部分を管理する（そのいかなるサイズ変更も伴わないが、対応するマス・メモリ・スペースの無駄を伴う）。

【００６７】

ソフトウェア・イメージは、あらゆる外部ソース（たとえば取り外し可能な記憶装置など）から上述のストリーミング技術によって配置されてもよい。加えて、マス・メモリにおけるメモリ・ブロックの利用可能性を管理するためにあらゆる同等の構造が用いられてもよい（たとえば、位置マップをチャンクに分割してそれらがクライアント・コンピュータのワーキング・メモリにロードされることを可能にすることなどによる）。代替的には、ソフトウェア・イメージが完全にダウンロードされた後も常にストリーミング・プロセスをアクティブに維持することも可能である（たとえば、位置マップ中の対応のフラグのリセットに応答してメモリ・ブロックの最新バージョンをダウンロードするため）。

【００６８】

いずれの場合にも、提案される技術は、ソフトウェア・イメージがいかにしてクライアント・コンピュータに配置されたかにはまったく依存しない（たとえば何らかのサーバ・コンピュータなしに手動で配置されてもよい）。

【００６９】

対応するソフトウェア・イメージの配置の際にブート・ブロックをそれらのブート位置にのみコピーする（クライアント・コンピュータが別のソフトウェア・イメージに切換えられるときにのみブート・ブロックがそのイメージ部分にセーブされる）可能性も排除されない。

【００７０】

代替的には、（配置エージェントを何ら通すことなく）物理的ディスク・ドライバによってメモリ・ブロックの書込を直接管理することが可能である。

【００７１】

作業負荷はあらゆる他の頻度でモニタされても、特定の期間（たとえば夜間）にのみモニタされてもよい。クライアント・コンピュータ、サーバ・コンピュータ、ネットワーク、またはそのあらゆる組合わせに対してのみ作業負荷がモニタされるときにも類似の考察が適用される。さらに、作業負荷に対する閾値はあらゆるその他のやり方で（たとえば、異なる重みによってその寄与を計量することによって）定められてもよい。作業負荷が閾値よりも低くなるときに２つまたはそれ以上のメモリ・ブロックが同時にダウンロードさ

れるときにも類似の考察が適用される。いずれの場合にも、ストリーミング・プロセスが常にアクティブのままであるときには、この特徴は除外されてもよい。

【００７２】

ソフトウェア・イメージは異なるやり方で準備されてもよい（たとえば、補助クライアント・コンピュータを実際にブートして、ブート・シーケンスの間にアクセスされるメモリ・ブロックを追跡することによってそのブート・ブロックを識別することによるものなど）。

【００７３】

（本発明の各実施形態を実施するために用いられ得る）プログラムが異なるやり方で構築されても、または付加的なモジュールもしくは機能が与えられても、類似の考察が適用される。同様に、メモリ構造は他のタイプのものであってもよいし、同等のエンティティ（必ずしも物理的記憶媒体からなっていなくてもよい）によって置換されてもよい。プログラムは、あらゆるデータ処理システムによって用いられるために好適であるか、またはそれに関連するあらゆる形を取ってもよい（たとえば仮想マシン内など）。特に、プログラムは外部もしくは常駐のソフトウェア、ファームウェア、またはマイクロコード（たとえばコンパイルまたは解釈されるべきオブジェクト・コードまたはソース・コード）の形であってもよい。さらに、あらゆるコンピュータ使用可能媒体において実施される製造品としてプログラムを提供することが可能である。その媒体は、プログラムを包含、保存、通信、伝搬、または転送するために好適なあらゆる構成要素であってもよい。たとえば、その媒体は電子、磁気、光学、電磁気、赤外、または半導体のタイプのものであってもよい。こうした媒体の例は、（プログラムをプレロードできる）固定ディスク、リムーバブル・ディスク、テープ、カード、ワイヤ、ファイバ、無線接続、ネットワーク、放送波などである。いずれの場合にも、本発明の実施形態に従う解決策は、ハードウェア構造（たとえば半導体材料のチップに集積されたもの）、または別様で構成されたものに好適にプログラムされたソフトウェアおよびハードウェアの組合わせによる実施にも適する。

【００７４】

代替的には、このシステムは異なる構造を有するか、同等の構成要素を含むか、または他の動作特徴を有する。いずれの場合にも、そのあらゆる構成要素がもっと多くの構成要素に分割されてもよいし、２つまたはそれ以上の構成要素が組合わされて単一の構成要素になってもよい。さらに、対応する動作の並行した実行を支持するために、各構成要素が複製されてもよい。異なる構成要素間のあらゆる対話は（別様に指定されない限り）一般的に継続的である必要はなく、対話は直接的であっても、１つまたはそれ以上の仲介物を通じた間接的なものであってもよいことが指摘される。特に、このシステムは異なるアーキテクチャ（たとえば、広域ネットワーク、グローバル・ネットワーク、セルラー・ネットワーク、または衛星ネットワークなど）に基づいていてもよく、あらゆるタイプの（有線または無線あるいはその両方の）接続を利用してもよい。いずれの場合にも、各コンピュータは別の構造を有してもよいし、類似の構成要素（たとえばプログラムまたはその部分を一時的に保存するキャッシュ・メモリなど）を含んでもよい。さらに、コンピュータをあらゆるコード実行エンティティ（たとえばＰＤＡ、携帯電話など）、または複数のエンティティの組合わせによって置換することが可能である。

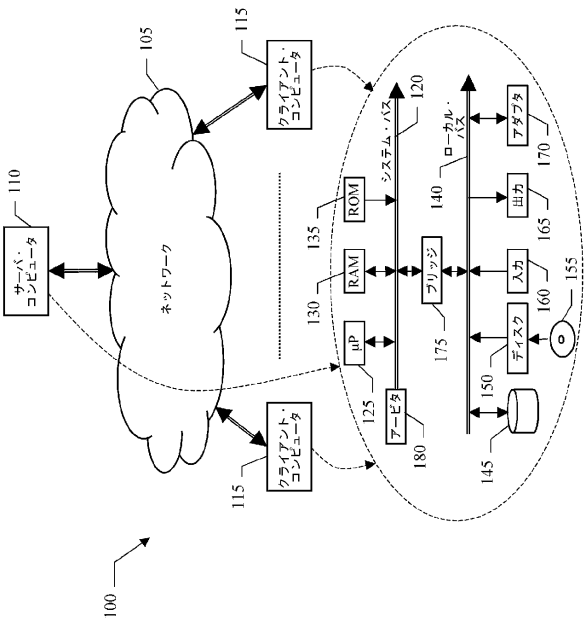
【符号の説明】

【００７５】

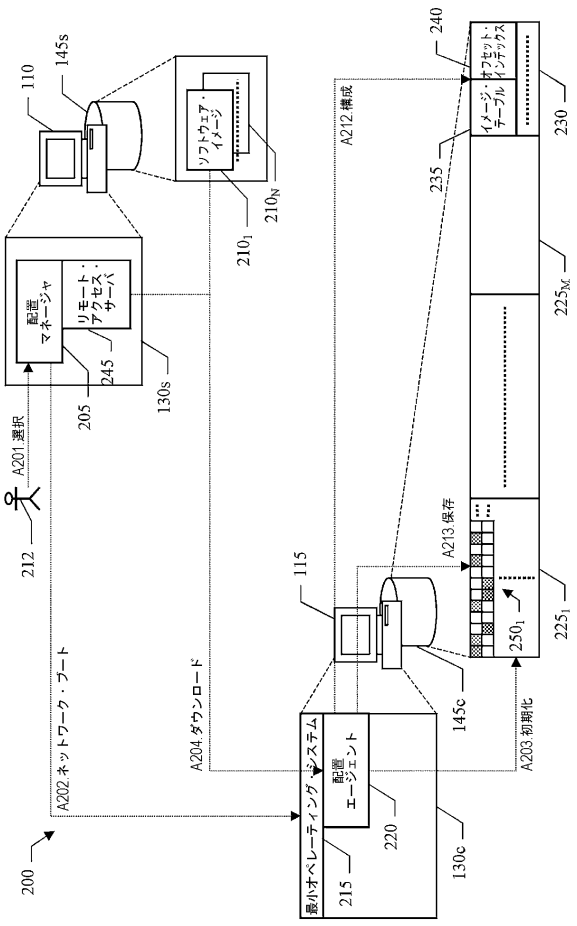
Ａ３０８　コピー
Ａ３０９　更新
Ａ３１０　リブート
１１５　クライアント・コンピュータ
１３０ｃ　ワーキング・メモリ
１４５ｃ　マス・メモリ
２２０　配置エージェント
２２５_{1-3}　イメージ部分

２３０　サービス部分
２３５　イメージ・テーブル
２４０　オフセット・インデックス
２５０$_3$　ブート・ブロック
２５５$_2$　オペレーティング・システム
２６０$_3$　再配置メモリ・ブロック

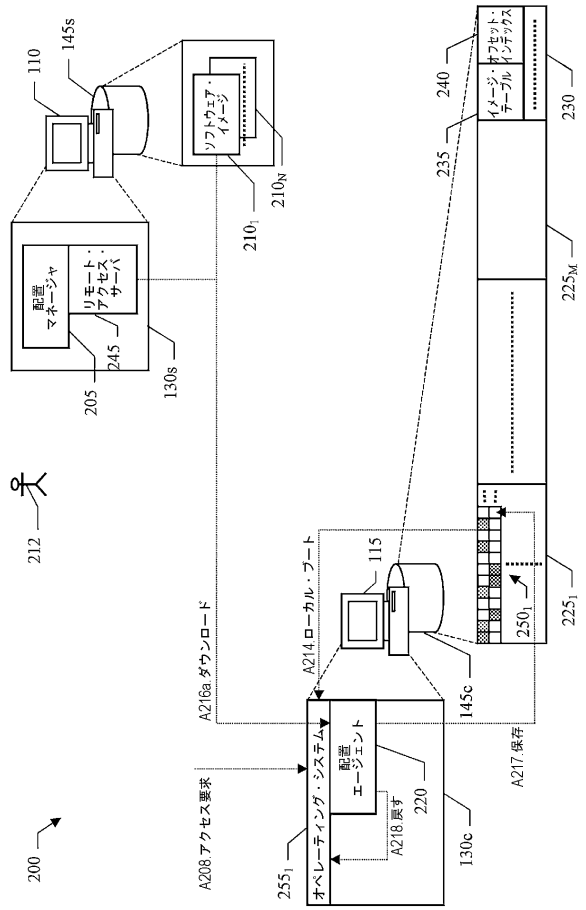
【図１】



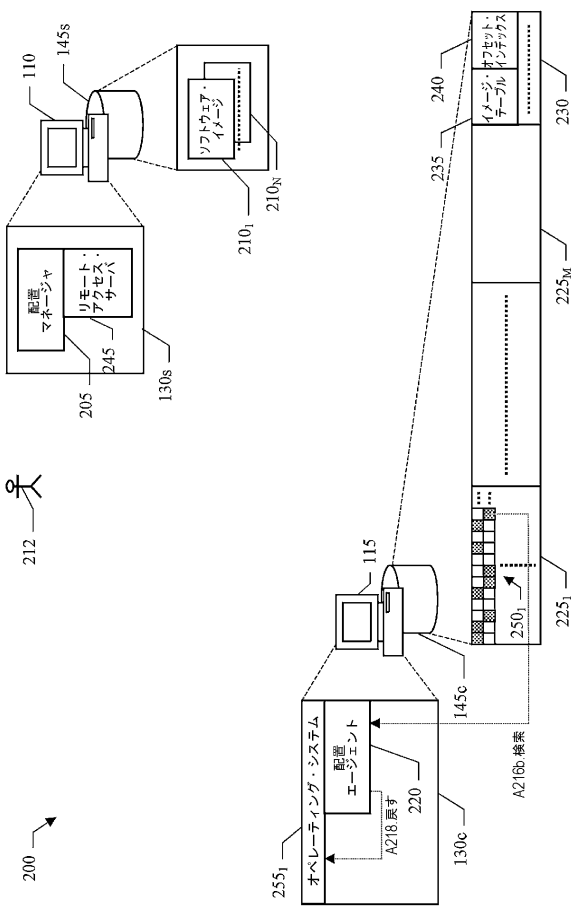
【図２】



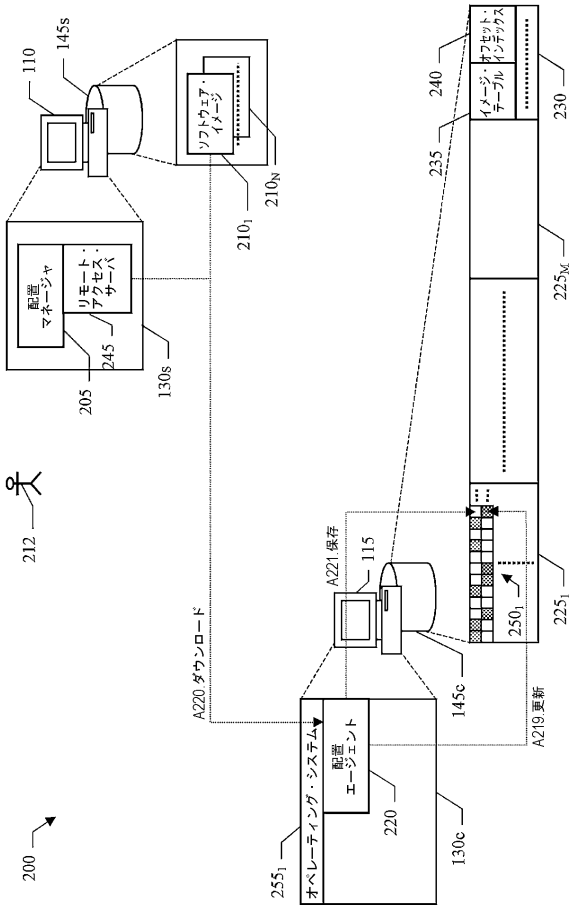
【図３】


200
212
110
145s
130s
205
245
リモート・アクセス・サーバ
配置マネージャ
ソフトウェア・イメージ
210_1
210_N
A216a ダウンロード
A214 ローカル・ブート
115
145c
250_1
225_1
225_M
235
240
イメージ・テーブル
オフセット・インデックス
230
A208 アクセス要求
255_1
オペレーティング・システム
配置エージェント
220
A218 戻す
130c
A217 保存

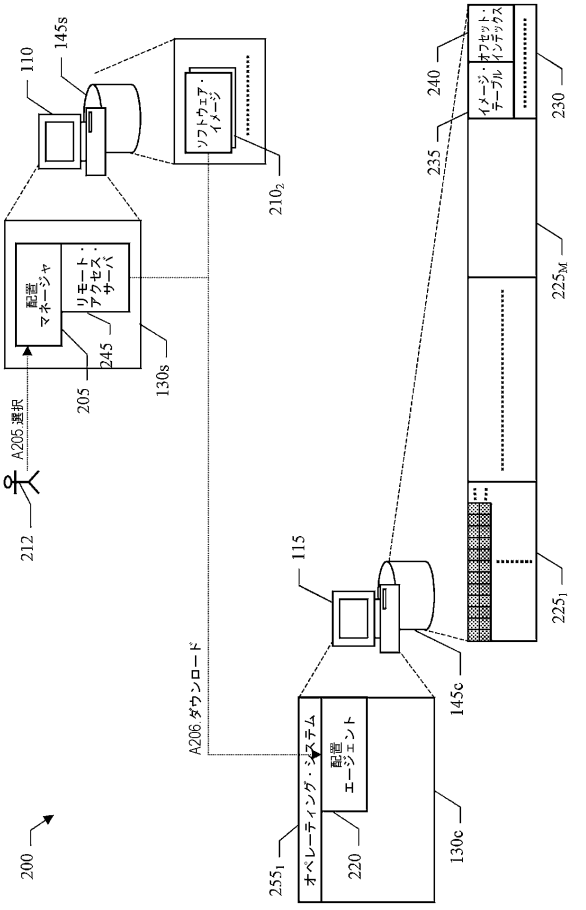
【図４】


200
212
110
145s
130s
205
245
リモート・アクセス・サーバ
配置マネージャ
ソフトウェア・イメージ
210_1
210_N
115
145c
250_1
225_1
225_M
235
240
イメージ・テーブル
オフセット・インデックス
230
255_1
オペレーティング・システム
配置エージェント
220
A218 戻す
130c
A216b 検索

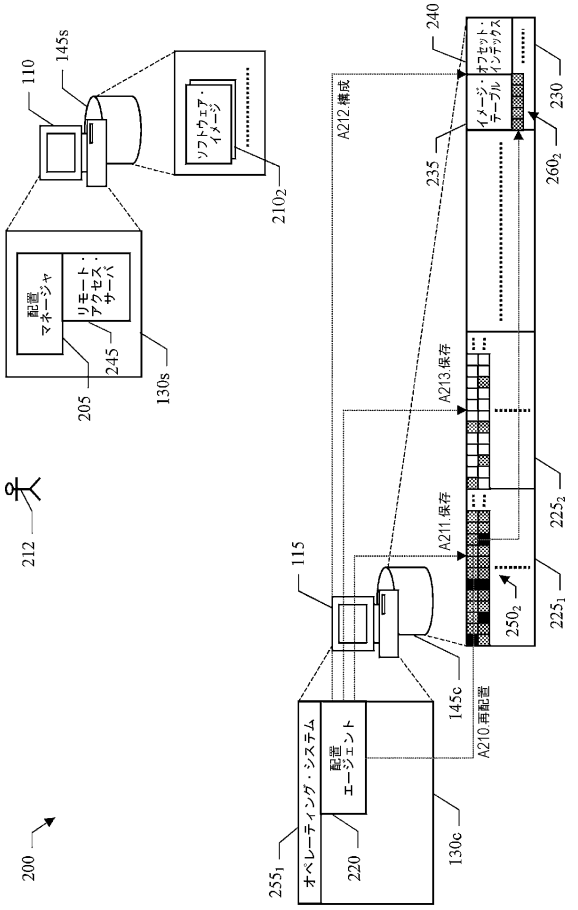
【図５】


200
212
110
145s
130s
205
245
リモート・アクセス・サーバ
配置マネージャ
ソフトウェア・イメージ
210_1
210_N
A220 ダウンロード
A221 保存
115
145c
250_1
225_1
225_M
235
240
イメージ・テーブル
オフセット・インデックス
230
255_1
オペレーティング・システム
配置エージェント
220
130c
A219 更新

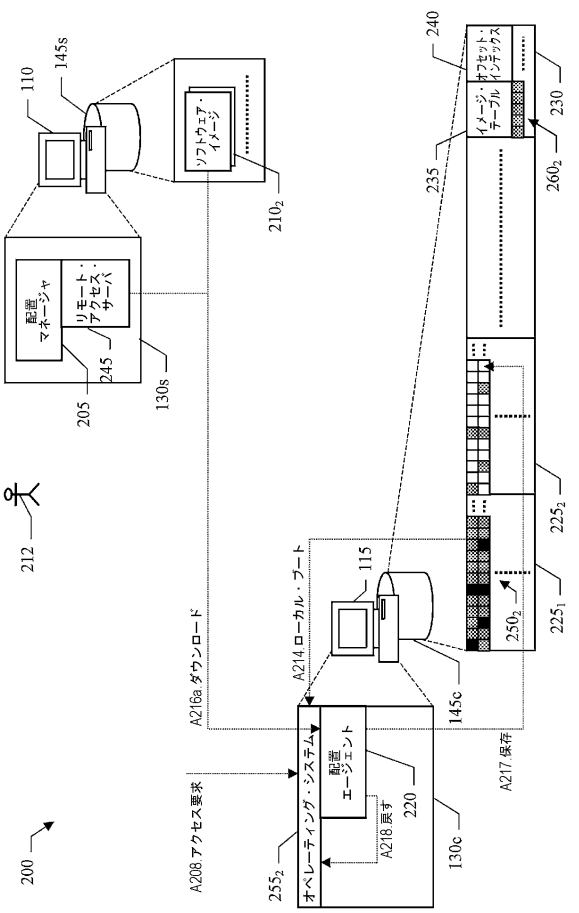
【図６】


200
212
A205 選択
110
145s
130s
205
245
リモート・アクセス・サーバ
配置マネージャ
ソフトウェア・イメージ
210_2
A206 ダウンロード
115
145c
225_1
225_M
235
240
イメージ・テーブル
オフセット・インデックス
230
255_1
オペレーティング・システム
配置エージェント
220
130c

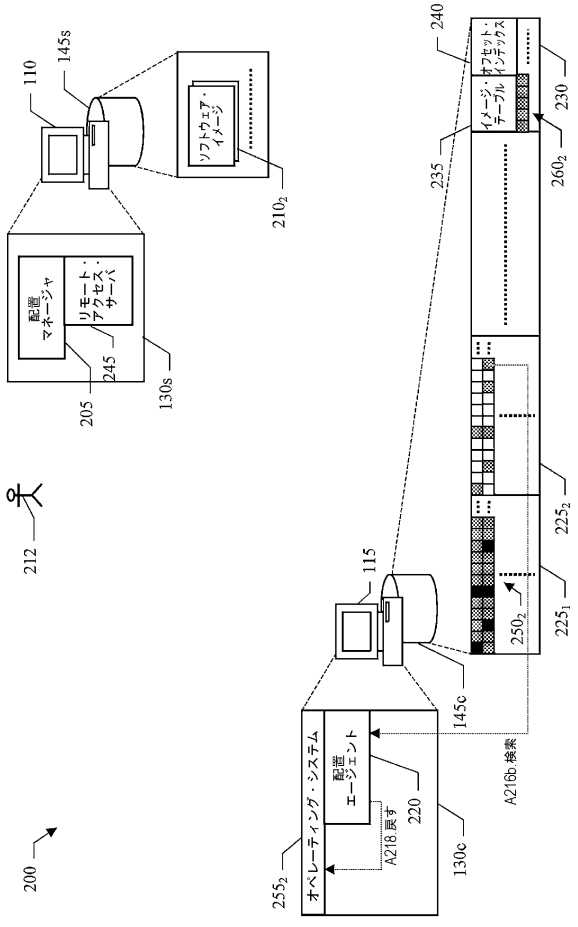
【図７】


200
212
110
145s
130s
205
245
配置マネージャ
リモート・アクセス・サーバ
ソフトウェア・イメージ
210_2
A212.構成
240
235
イメージ・テーブル
オフセット・インデックス
260_2
230
A213.保存
225_2
A211.保存
225_1
250_2
115
145c
A210.再配置
255_1
オペレーティング・システム
配置エージェント
220
130c

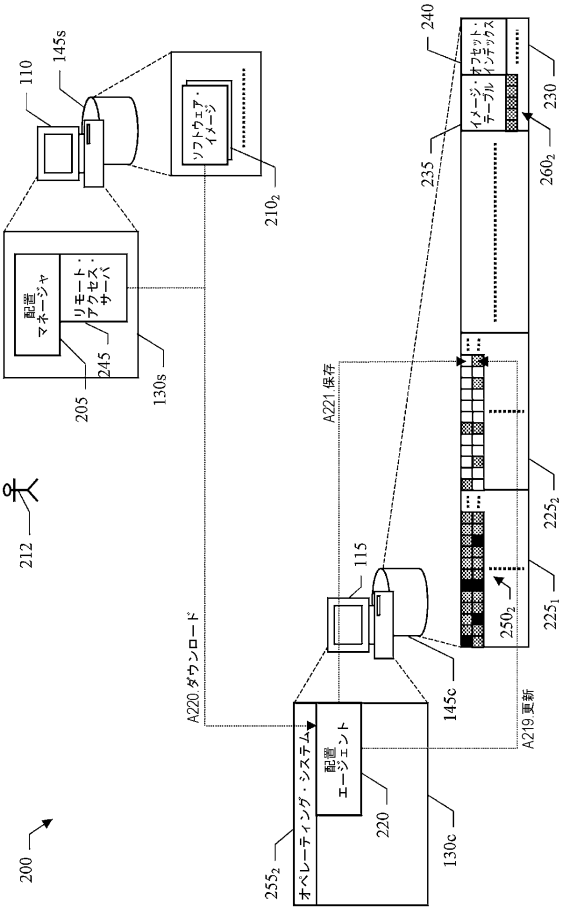
【図８】


200
212
110
145s
130s
205
245
配置マネージャ
リモート・アクセス・サーバ
ソフトウェア・イメージ
210_2
240
235
イメージ・テーブル
オフセット・インデックス
260_2
230
225_2
225_1
250_2
115
145c
A214 ローカル・ブート
A216a.ダウンロード
A208.アクセス要求
255_2
オペレーティング・システム
配置エージェント
220
A218.戻す
130c
A217.保存

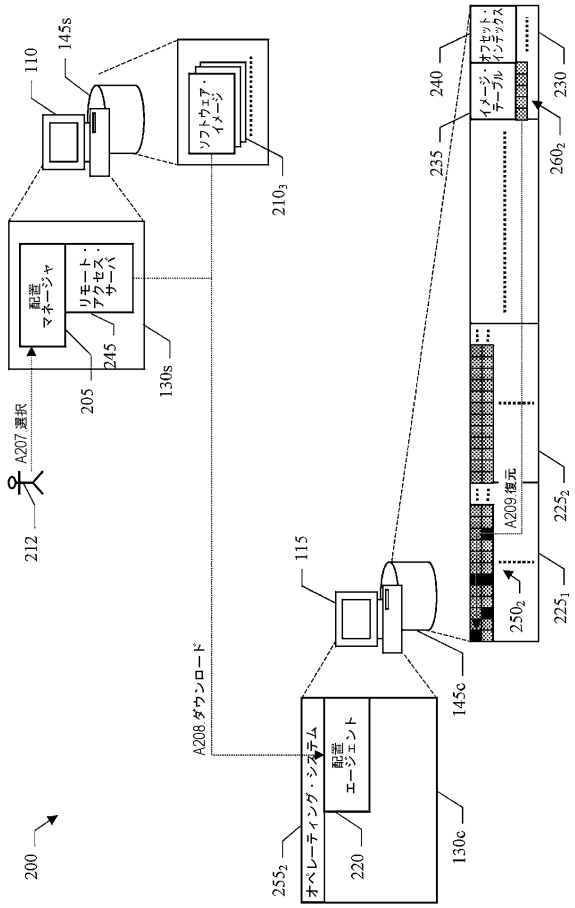
【図９】


200
212
110
145s
130s
205
245
配置マネージャ
リモート・アクセス・サーバ
ソフトウェア・イメージ
210_2
240
235
イメージ・テーブル
オフセット・インデックス
260_2
230
225_2
225_1
250_2
115
145c
A216b.検索
255_2
オペレーティング・システム
配置エージェント
A218.戻す
220
130c

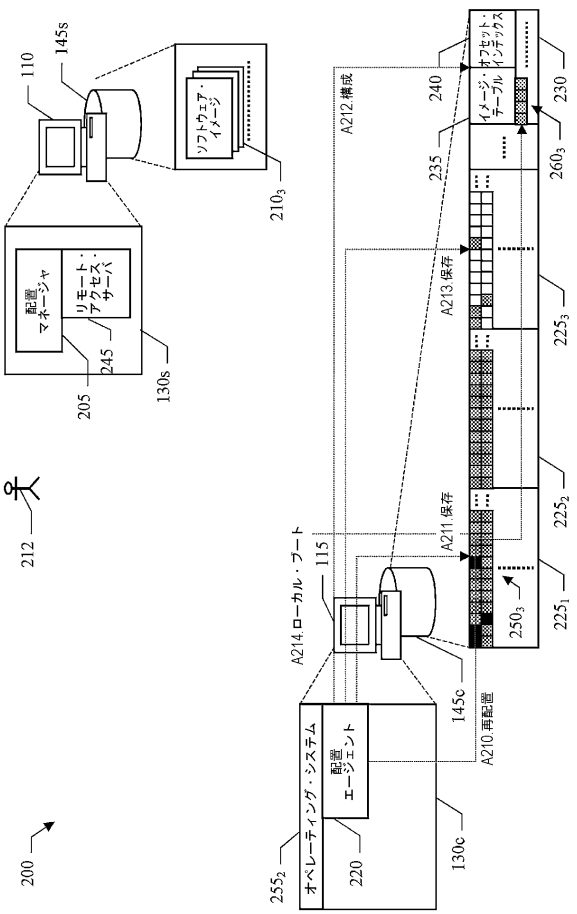
【図１０】


200
212
110
145s
130s
205
245
配置マネージャ
リモート・アクセス・サーバ
ソフトウェア・イメージ
210_2
240
235
イメージ・テーブル
オフセット・インデックス
260_2
230
A221.保存
225_2
225_1
250_2
115
145c
A220.ダウンロード
A219.更新
255_2
オペレーティング・システム
配置エージェント
220
130c

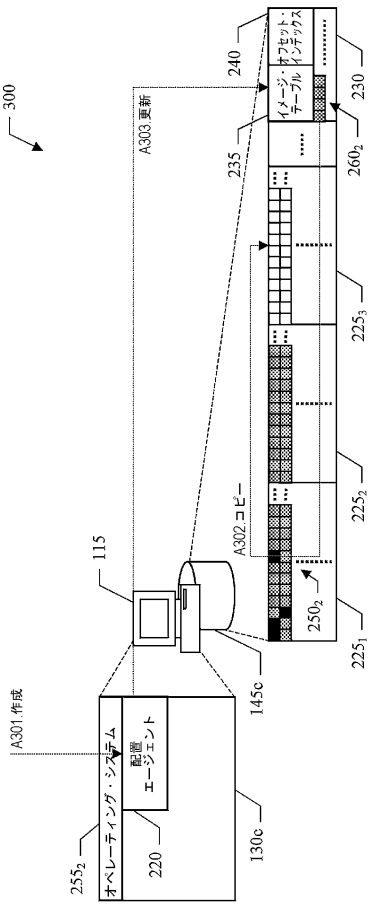
【図１１】



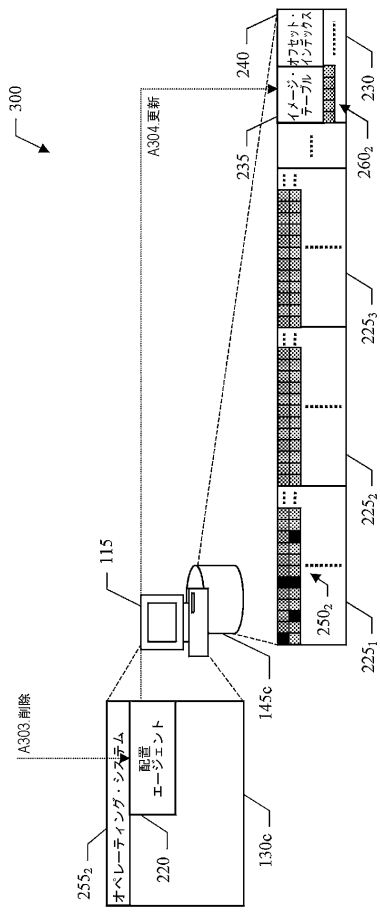
【図１２】



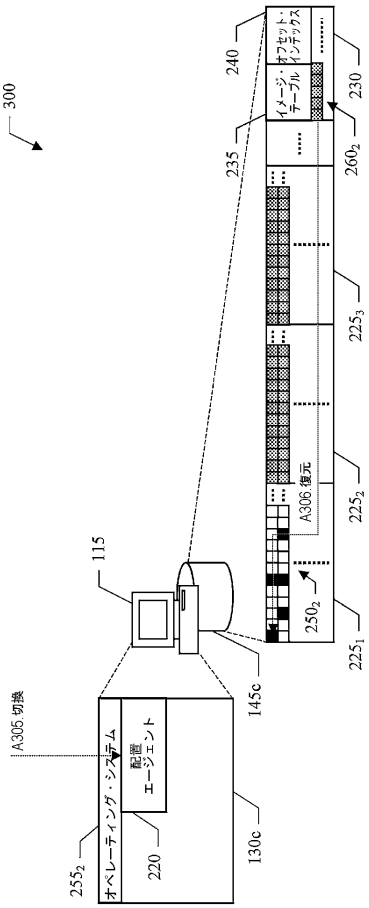
【図１３】



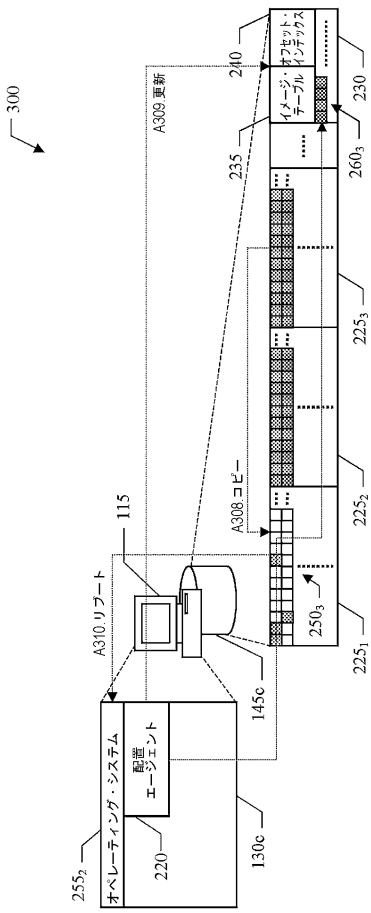
【図１４】



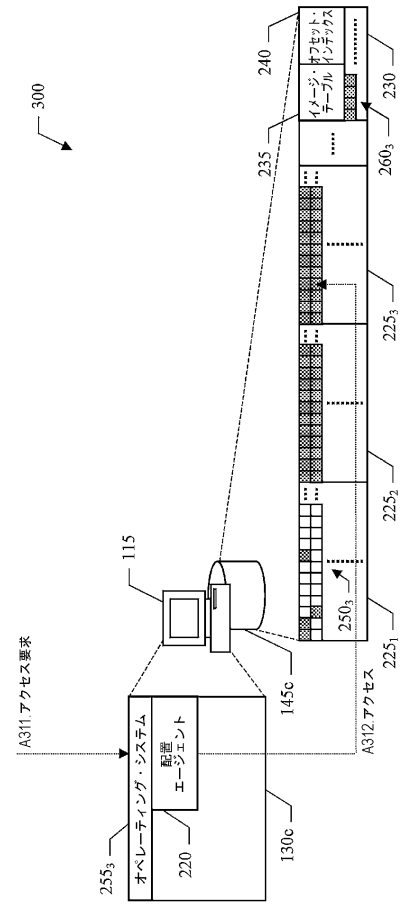
【図１５】



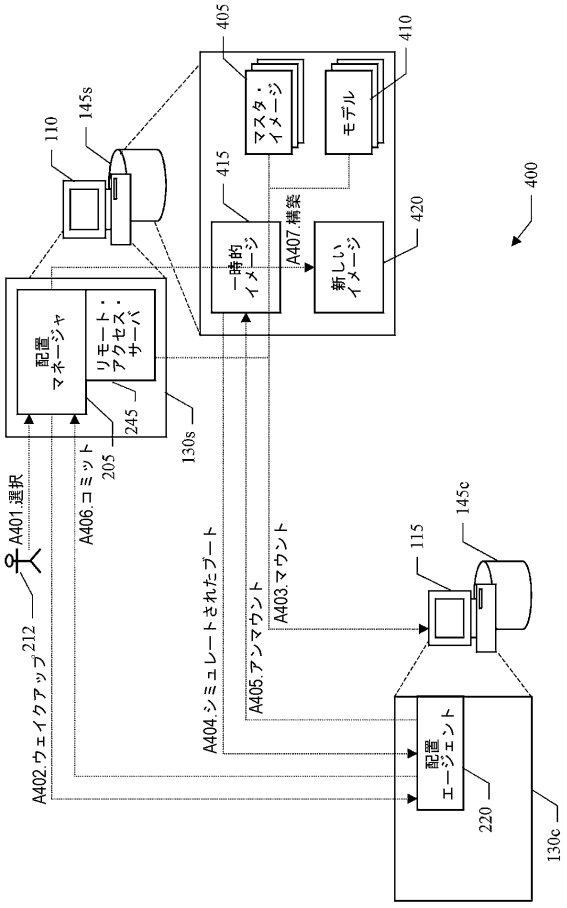
【図１６】



【図１７】



【図１８】



フロントページの続き

(72)発明者 クラウディオ・マリネッリ
イタリア共和国 ＩＴ－００１４４ ローマ市 ビア・シャンガイ ５３
(72)発明者 ジャック・フォンティニー
スイス連邦 ＣＨ－１２０５ ジュネーブ リュ・デ・バン ３５
(72)発明者 マルク・ビュイウミエ・シュテュッケルベルク
スイス連邦 １２０５ ジュネーブ州 ジュネーブ １２０５ リュ・ド・バン ３３－３５
(72)発明者 ジョン・ジー. ・ルーニー
スイス連邦 ＣＨ－８８０３ リュシュリコン ソイマーシュトラーセ ４
(72)発明者 ダヴィッド・クレール
スイス連邦 ＣＨ－１９３６ ベルビエ リュ・ド・メドラン ６６
(72)発明者 ルイス・ガルセス－エリセ
スイス連邦 ＣＨ－８８０３ リュシュリコン ソイマーシュトラーセ ４

審査官 大塚 俊範

(56)参考文献 特開２０１２－１２３７６２（ＪＰ，Ａ）
特開２００５－３１６８０９（ＪＰ，Ａ）
特開２００１－１００９８３（ＪＰ，Ａ）
特表２０１３－５４２４８６（ＪＰ，Ａ）
国際公開第２００９／１４５２７４（WO，Ａ１）

(58)調査した分野(Int.Cl.，ＤＢ名)
Ｇ０６Ｆ ９／４４５