(19)日本国特許庁(JP)　(12)特　許　公　報(B2)　(11)特許番号
特許第4374391号
(P4374391)
(45)発行日　平成21年12月2日(2009.12.2)　(24)登録日　平成21年9月11日(2009.9.11)

(51)Int.Cl.　F I
G06F 9/50 (2006.01)　G06F 9/46 465D

請求項の数 8 (全 11 頁)

(21)出願番号	特願2008-68757(P2008-68757)
(22)出願日	平成20年3月18日(2008.3.18)
(62)分割の表示	特願2005-214194(P2005-214194)の分割
原出願日	平成17年7月25日(2005.7.25)
(65)公開番号	特開2008-234651(P2008-234651A)
(43)公開日	平成20年10月2日(2008.10.2)
審査請求日	平成20年3月18日(2008.3.18)
(31)優先権主張番号	10/902,763
(32)優先日	平成16年7月30日(2004.7.30)
(33)優先権主張国	米国(US)

(73)特許権者　503003854
ヒューレット－パッカード　デベロップメント　カンパニー　エル．ピー．
アメリカ合衆国　テキサス州　７７０７０　ヒューストン　コンパック　センタ　ドライブ　ウエスト　１１４４５
(74)代理人　110000039
特許業務法人アイ・ピー・エス
(72)発明者　ダニエル・イー・ハーリントン
アメリカ合衆国カリフォルニア州　パロアルト　ハノーバー・ストリート　３０００　ヒューレット・パッカード・カンパニー内

最終頁に続く

(54)【発明の名称】複数のインスタンスアプリケーションに対し負荷分散装置を動作させるシステムおよび方法

(57)【特許請求の範囲】
【請求項１】
複数のインスタンスアプリケーションに対し負荷分散装置を動作させるシステムであって、
アプリケーションを実行する複数のクラスタノード（２１０）であって、前記複数のクラスタノードの少なくとも一部は、外部から単一システムであるように見えるアプリケーションクラスタを構成し、複数のアプリケーションを実行し、前記複数のアプリケーションに関連する性能データに応じて、前記複数のアプリケーション間で資源を割り当てるそれぞれの資源割付けモジュール
を有する複数のクラスタノードと、
前記複数のクラスタノード間で、前記複数のクラスタノードが実行する前記複数のアプリケーションに関するアプリケーショントランザクションを分散させる複数の負荷分散装置（２０１）と、
前記複数のクラスタノードが実行するアプリケーションを特定し、前記特定されたアプリケーションに関連するアプリケーション負荷特性を、作業負荷待ち行列を検査することによって取得し、特定されたアプリケーションに関連する性能データおよび前記アプリケーション負荷特性に基づいて、前記複数の負荷分散装置に対する重み付け係数であって、前記作業負荷待ち行列の長さで除算しプロセッサ利用率で除算したプロセッサの数に関連する重み付け係数を計算し、前記計算された重み付け係数に応じて前記複数の負荷分散装置のパラメータを動的に構成し、前記複数のクラスタノード間で、前記複数のクラスタノ

ードが実行する前記複数のアプリケーションに関するアプリケーショントランザクションを分散させる構成プロセス（２４０）と
を具備するシステム。
【請求項２】
前記構成プロセスは、前記性能データの分析中にそれぞれのクラスタノードに関連するアプリケーション負荷を分析する
請求項１に記載のシステム。
【請求項３】
前記構成プロセスは、前記クラスタシステムに対するクラスタノードの追加を自発的に検出し、前記追加に応じて前記複数の負荷分散装置のパラメータを再構成する
請求項１に記載のシステム。
【請求項４】
前記複数のクラスタノードの各々は、
性能データを生成する性能モニタプロセス（２１２）
を備える請求項１に記載のシステム。
【請求項５】
複数のクラスタノード（２１０）が、複数のアプリケーションを実行することであって、前記複数のクラスタノードの少なくとも一部は、外部から単一システムであるように見えるアプリケーションクラスタを構成し、複数のアプリケーション（ＡＰＰａ，ＡＰＰｂ）を実行することと、
前記複数のクラスタノードの資源割付けモジュールが、前記複数のアプリケーションに関連する性能データに応じて、前記複数のアプリケーション間で資源を動的に再割当てすることと、
複数の負荷分散装置（２０１）が、パラメータに従って前記複数のクラスタノード間で、前記複数のクラスタノードが実行する前記複数のアプリケーションに関するアプリケーショントランザクションを分散させることと、
構成プロセス（２４０）が、前記複数のクラスタノードが実行するアプリケーションを特定し、前記特定されたアプリケーションに関連するアプリケーション負荷特性を、作業負荷待ち行列を検査することによって取得し、特定されたアプリケーションに関連する性能データおよび前記アプリケーション負荷特性に基づいて、前記複数の負荷分散装置に対する重み付け係数であって、前記作業負荷待ち行列の長さで除算しプロセッサ利用率で除算したプロセッサの数に関連する重み付け係数を計算し、前記計算された重み付け係数に応じて、前記パラメータを動的に構成し、前記複数のクラスタノード間で、前記複数のクラスタノードが実行する前記複数のアプリケーションに関するアプリケーショントランザクションを分散させることと
を含む方法。
【請求項６】
前記動的に構成することは、
クラスタノードのアプリケーション負荷特性を分析すること
を含む
請求項５に記載の方法。
【請求項７】
前記構成プロセスが、前記クラスタシステムに対するクラスタノードの追加を検出することと、
前記構成プロセスが、前記検出することに応じて前記パラメータを動的に構成することと
をさらに含む請求項５に記載の方法。
【請求項８】
前記資源を動的に再割当てすることは、前記複数のアプリケーションに関連するサービスレベル目標に従って資源を再割当てする

請求項５に記載の方法。
【発明の詳細な説明】
【技術分野】
【０００１】
本出願は、包括的には、複数のインスタンスアプリケーションに対し負荷分散装置を動作させることに関する。
【背景技術】
【０００２】
最近のアプリケーションアーキテクチャは、アプリケーションのクラスタ化された実行を支援することが多い。
クラスタ化された実行とは、システムのセット（クラスタノード）におけるインスタンスの集まり（ほとんどの場合、同一のインスタンス）としてアプリケーションを実行することにより、作業負荷がそれらシステムにわたって分散されバランスが取られるようにすることを言う。
任意の特定のクラスタノードに障害が発生すると、作業負荷は、残りのシステムにおいて変わらずにまたは幾分か性能が低下して続行する。
【０００３】
クラスタ化された実行には複数の利点がある。
たとえば、クラスタ化された実行により、可用性がより向上する。
それは、クラスタノードの障害によって完全なアプリケーションの障害がもたらされないためである。
さらに、通常、クラスタ化された実行によってコストが下がる。
それは、拡張を、モノリシックサーバをより大型のサーバと交換するのではなく、より小さいサーバを使用して漸次的に行うことができるためである。
同じ理由で、分散アプリケーションをより高速にスケーリングすることができる。
【特許文献１】米国公報２００３００３７０９２号
【発明の開示】
【課題を解決するための手段】
【０００４】
代表的な一実施形態では、複数のインスタンスアプリケーションに対し負荷分散装置を動作させるシステムは、アプリケーションを実行する複数のクラスタノードであって、複数のクラスタノードの少なくとも一部は、複数のアプリケーションを実行し、かつ複数のアプリケーションに関連する性能データに応じて複数のアプリケーション間で資源を割り当てるそれぞれの資源割付けモジュールを有する、複数のクラスタノードと、複数のクラスタノード間でアプリケーショントランザクションを分散させる複数の負荷分散装置と、複数のアプリケーションに関連する性能データを分析し、分析に応じて複数の負荷分散装置を動的に構成する構成プロセスと、を具備する。
【０００５】
別の代表的な実施形態では、方法は、複数のクラスタノードに対して複数のアプリケーションを実行することであって、複数のクラスタノードの少なくとも一部は複数のアプリケーションを実行する、実行すること、複数のアプリケーションに関連する性能データに応じて複数のアプリケーション間で資源を動的に再割当てすること、パラメータに従って複数のクラスタノード間でアプリケーショントランザクションを分散させること、および、複数のアプリケーションに関連する性能データに応じてパラメータを動的に構成すること、を含む。
【０００６】
別の代表的な実施形態では、コンピュータ読取可能媒体は、複数のクラスタノードにおけるアプリケーションの実行に関連する性能データを検索するコードであって、複数のクラスタノードのうちの少なくとも一部が複数のアプリケーションを実行する、コードと、性能データを使用してクラスタノード重みの複数のセットを計算するコードと、クラスタ

ノード重みの複数のセットを使用して、複数のクラスタノードに対するアプリケーショントランザクションの分散を制御するように負荷分散装置を動的に構成するコードと、を備える。

【発明を実施するための最良の形態】

【０００７】

図１は、既知のクラスタシステム設計によるクラスタシステム１００を示す。

クラスタシステム１００は、各々がアプリケーション「ａ」のインスタンスを実行するクラスタノード１１０ａ～１１０ｃを備える。

クラスタノード１１０ａ～１１０ｃは、クライアント１４０に対し単一システムであるように見える「アプリケーションクラスタ」を形成する。

【０００８】

クラスタシステム１００はまた、各々がアプリケーション「ｂ」のインスタンスを実行するクラスタノード１１０ｄ～１１０ｆも備える。

ノード１１０ｄ～１１０ｆは、クライアント１４０に対し単一システムであるように見える別のアプリケーションクラスタを形成する。

【０００９】

一般に、アプリケーションに対するクラスタノードの数は、それぞれのアプリケーションの最悪の場合の要求レベルに従って選択される。

アプリケーショントランザクションは、クライアント１４０によって生成され、ネットワーク１３０を介して負荷分散装置１２０ａおよび１２０ｂにルーティングされる。

負荷分散装置１２０ａおよび１２０ｂは、通常、重み付きラウンドロビンアルゴリズムを使用して、アプリケーショントランザクションを処理されるために特定のクラスタノード１１０に向ける。

したがって、多数のクライアントがクラスタアプリケーションにアクセスしようと試みる場合、それらのクライアントに関連する処理は、複数のクラスタノードにわたって分散され、アプリケーション性能は通常、たとえ低減しても許容可能なレベルで維持される。

アプリケーションがピーク負荷中に許容可能な性能を示さない場合、クラスタシステム１００に追加のクラスタノードを付加することにより、アプリケーションの性能が向上するようにしてもよい。

そして、それぞれの負荷分散装置１２０は、既存のノード１１０の能力に対する新たなノード１１０の相対能力に従って手動で再調整される。

【００１０】

クラスタシステムは、複数の利点を提供するが、いくつかの制限がある。

資源が最悪の場合の要求レベルに従って選択される場合、クラスタノードの全体的な利用率が低くなる可能性がある。

特に、ピークアプリケーション負荷の持続時間が比較的短い場合、特定のアプリケーションに関するクラスタノードが、有意な時間にわたってアイドル状態である可能性がある。

したがって、アイドル状態のシステム資源が無駄になる。

【００１１】

代表的な一実施形態により、アイドル状態であるかまたは他の状態で十分に利用されていないシステム資源を効率的に管理することができる。

代表的な一実施形態では、複数のアプリケーションを実行するクラスタノードがあり、これらのクラスタノードは、複数のアプリケーションの性能を分析するそれぞれの性能モニタを有する。

性能モニタは、作業負荷マネージャに性能データを通信する。

作業負荷マネージャは、性能データとサービスレベル目標とに応じてアプリケーション間のプロセッサ資源または他の資源の割付けを再調整する。

したがって、任意の特定の瞬間により重く負荷がかけられるアプリケーションが、増大した負荷を満足させるために追加の処理資源または他の資源を受け取る。

【００１２】
一実施形態の構成ユーティリティは、さまざまなクラスタノードに対し定期的に問い合わせることにより、それぞれのノードによりいずれのアプリケーションが実行されているかを判断する。
また、作業負荷マネージャのさまざまなインスタンスに対し構成ユーティリティが問い合わせることにより、さまざまなアプリケーションに関連する性能情報を取得する。
それぞれのノードにおけるアプリケーションに関連する負荷特性を、作業負荷待ち行列または他の適当な待ち行列を検査することによって分析してもよい。
さらに、構成ユーティリティは、さまざまなクラスタノードに割り当てられる資源を特定してもよい。
取得された情報を使用して、構成ユーティリティは、クラスタノードにわたってアプリケーショントランザクションを分散させるための重み付け係数を計算する。
構成ユーティリティは、クラスタノードへのトランザクションの分散を制御するために、計算された重みを負荷分散装置に通信する。
【００１３】
このように、代表的な実施形態によっては、同じ数のアプリケーションを支援するために、クラスタシステムにおいて従来のクラスタアーキテクチャを使用して採用されるものより少ない数のノードを採用することができる。
複数のアプリケーションをアプリケーションノードのいくつかにおいてインスタンス化することにより、追加のノードを付加することなく複数のアプリケーションに対しより多くの資源を利用することができるようにする。
そして、資源を、負荷要求に応じて割り付けてもよい。
特に、ピーク負荷が発生すると、追加のプロセッサおよび／または他の資源を、トラフィックが重くなっているアプリケーションに割り当ててもよい。
資源が動的に割り付けられるため、負荷分散装置は適当に調整される。
したがって、資源の動的割当てにより、アプリケーション性能の向上が可能になる。
それは、アプリケーショントランザクションが、特に、それぞれのトランザクションを処理するために追加の資源を割り付けたクラスタノードに向けられるためである。
【００１４】
図２は、代表的な一実施形態によるクラスタシステム２００を示す。
クラスタシステム２００は、複数のクラスタノード（２１０ａ～２１０ｇとして示す）を含む。
クラスタリングは、一般に、モジュラーコンピューティングを容易にするハードウェアおよび／またはソフトウェア接続を必要とする。
クラスタは、通常、アプリケーションスケーラビリティおよび信頼性を向上させるように互いに協働する適当なサーバの複数のインスタンスを含む。
クラスタを構成するサーバの複数のインスタンスは、同じ物理システムまたは複数の物理システムで実行してもよい。
たとえば、各クラスタノード２１０は、それぞれのサーバを支援するために使用される単一の物理システムの仮想コンピューティング資源を表してもよい。
別法として、別個の物理プラットフォームを使用して各サーバを実行してもよい。
【００１５】
図２に示すように、アプリケーション「ａ」のインスタンスはノード２１０ａ～２１０ｆに提供され、アプリケーション「ｂ」のインスタンスは、ノード２１０ｂ～２１０ｇに提供される。
アプリケーションａおよびｂは、ネットワーク１３０を介してクライアント１４０から受け取られるアプリケーショントランザクションを処理する。
特に、クラスタエイリアシングにより、アプリケーションａのアプリケーショントランザクションが負荷分散装置２０１ａにルーティングされ、アプリケーションｂのアプリケーショントランザクションが、負荷分散装置２０１ｂにルーティングされる。

そして、負荷分散装置２０１ａおよび２０１ｂは、アプリケーションのトランザクションをノード２１０間で分散することにより、それぞれのノード２１０における負荷のバランスをとる。

代表的な一実施形態では、負荷分散装置２０１ａおよび２０１ｂは、重み付きラウンドロビン分散アルゴリズムを実施する。

負荷分散装置２０１ａおよび２０１ｂを、適当なハードウェア（たとえば、集積回路）を使用して実施してもよくかつ／または１つまたはいくつかのプロセッサで実行するソフトウェアを使用して実施してもよい。

負荷分散装置２０１ａおよび２０１ｂを、個別デバイスとして実施してもよく、あるいは別法として１つまたはいくつかのノード２１０で実施してもよい。

【００１６】

いくつかの実施形態の性能モニタ（ＰＭ）２１２は、アプリケーションの性能を検査するソフトウェアプロセスである。

たとえば、性能モニタ２１２は、プロセッサの利用率、入出力（ＩＯ）資源の利用率、或るタイプのトランザクションを処理するために費やされる時間の長さおよび／または同様のものを検査してもよい。

性能モニタ２１２は、実施形態によっては同様にソフトウェアプロセスである作業負荷マネージャ（ＷＬＭ）２１１に性能データを通信してもよい。

【００１７】

サービスレベル目標（ＳＬＯ）は、アプリケーションに関連する所望の動作目標またはルールを言う。

ＳＬＯは、所望の利用率、特定のトランザクションタイプに対する所望の処理時間の長さおよび／または同様のものを定義してもよい。

アプリケーションが１つまたはいくつかのＳＬＯを満たさない場合、作業負荷マネージャ２１１は、アプリケーション間で資源（たとえば、プロセッサ２１３、メモリ資源、ＩＯ資源、オペレーティングシステム資源および／または同様のもの）を再分散させてもよい。

たとえば、１つのアプリケーションがアイドル状態の資源を所有する場合、ＷＬＭ２１１は、それら資源を、十分に実行していないアプリケーションに対して再割当てしてもよい。

再割当てされた資源により、十分に実行していないアプリケーションがそのＳＬＯ（複数可）を満たすことができる可能性がある。

さらに、望ましい場合は、仮想パーティションおよびパーティションロードマネージャソフトウェアプロセス（図示せず）を採用してもよい。

性能モニタ、作業負荷マネージャ、仮想パーティションおよびパーティションロードマネージャに関するさらなる詳細は、参照により本明細書に援用される、２００２年７月２６に出願された、「Dynamic management of virtual partition computer workloads through service level optimization」と題する米国特許第１０／２０６，５９４号、公報番号２００３００３７０９２号により詳細に説明されている。

【００１８】

代表的な一実施形態は、資源を再割当てするために作業負荷マネージャを採用するが、任意の適当な資源割付け機構またはアルゴリズムを採用してもよい。

システム２００からの他の変形を使用してもよい。

たとえば、代表的な実施形態は、各クラスタノードに対し同一のサーバプラットフォームまたは仮想資源を採用する必要はない。

それぞれのクラスタノードにおいて異なる量のプロセッサ２１３、種々の能力を有するプロセッサ２１３および他の資源の相違が存在してもよい。

さらに、各クラスタノードにおいて作業負荷マネージャまたは他の資源割付け能力を実施する必要はない。

【００１９】

一実施形態の構成ユーティリティ２４０は、負荷分散装置２０１およびクラスタノード２１０の動作を調整するソフトウェアプロセスである。

代表的な一実施形態では、構成ユーティリティ２４０は、各クラスタノード２１０においていずれのアプリケーションが実行されるかと、いずれのノードがＷＬＭ２１１または他の動的資源割付け能力のインスタンスを所有するかと、を定期的に確定する。

さらに、構成ユーティリティ２４０は、ノード２１０からアプリケーションの実行に関連する性能データを取得する。

構成ユーティリティ２４０はまた、クラスタノード２１０の各々に対しアプリケーション負荷特性を取得してもよい。

アプリケーション負荷特性を、例として作業負荷または他の適当な待ち行列を検査することによって取得してもよい。

構成ユーティリティ２４０はまた、それぞれのクラスタノード２１０に割り当てられた資源を検査してもよい。

取得された情報に応じて、構成ユーティリティ２４０は、負荷分散装置２０１ａおよび２０１ｂに対する重みのセットを計算する。

たとえば、一実施形態では、構成ユーティリティ２４０は、各ノードに対し、関連する作業負荷待ち行列の長さで除算しプロセッサ利用率で除算したプロセッサの数に関連する重みを計算してもよい。

【００２０】

図３は、代表的な一実施形態によるクラスタシステムにおいて動作を管理するフローチャートを示す。

図３を、例として構成ユーティリティ２４０に対し適当な実行可能命令またはソフトウェアコードを使用して実施してもよい。

ブロック３０１において、各クラスタにおけるノードを特定する。

１つまたはいくつかのクラスタに対する新たなノードの追加もまた、このブロックで検出する。

これらの動作を、それぞれのクラスタを管理することに関与する管理サーバに問い合わせることによって実行してもよい。

ブロック３０２において、ＷＬＭ能力または他の適当な動的資源割付け能力を有するノードを特定する。

この特定を、適当なシステムコールまたは他のファンクションコールを使用して各クラスタノードにおいて実行するプロセスを検査することによって行ってもよい。

ブロック３０３において、特定されたノードにおいてＷＬＭから、割付け資源データ、負荷データおよび／または性能データを収集する。

かかる情報を、例としてＷＬＭ２１１に問い合わせることによって取得してもよい。

ブロック３０４において、収集されたデータを使用して、クラスタノード重みを計算する。

ブロック３０５において、負荷分散装置を再構成する。

適当な分散アルゴリズムを使用してアプリケーショントランザクションを振り向ける（direct）ために、計算されたクラスタノード重みを負荷分散装置に提供する。

さらに、新たなノードが検出される場合、適当な負荷分散装置のノードリストを更新する。

【００２１】

ソフトウェアで実施される場合、本発明の要素は、本質的に、必要なタスクを実行するコードセグメントである。

プログラムまたはコードセグメントを、コンピュータ読取可能媒体に格納してもよく、または搬送波として具現化されるコンピュータデータ信号または搬送波によって変調される信号により伝送媒体を介して送信してもよい。

「コンピュータ読取可能媒体」は、情報を格納しまたは転送することができる任意の媒体を含んでもよい。

コンピュータ読取可能媒体の例には、電子回路、半導体メモリデバイス、ＲＯＭ、フラッシュメモリ、消去可能ＲＯＭ（ＥＲＯＭ）、フロッピー（登録商標）ディスケット、コンパクトディスクＣＤ－ＲＯＭ、光ディスク、ハードディスク、光ファイバ媒体、無線周波数（ＲＦ）リンク等がある。

コードセグメントを、インターネット、イントラネット等のコンピュータネットワークを介してダウンロードしてもよい。

【００２２】

図４は、代表的な一実施形態によって適合されるコンピュータシステム４００を示す。

中央処理装置（ＣＰＵ）４０１が、システムバス４０２に結合される。

ＣＰＵ４０１は、任意の汎用ＣＰＵであってもよい。

しかしながら、本発明は、ＣＰＵ４０１が本明細書で説明するような発明的動作を支援する限りＣＰＵ４０１のアーキテクチャによって限定されない。

バス４０２は、ランダムアクセスメモリ（ＲＡＭ）４０３に結合され、ＲＡＭ４０３は、ＳＲＡＭ、ＤＲＡＭまたはＳＤＲＡＭであってもよい。

バス４０２には、ＲＯＭ４０４もまた結合され、ＲＯＭ４０４は、ＰＲＯＭ、ＥＰＲＯＭまたはＥＥＰＲＯＭであってもよい。

ＲＡＭ４０３およびＲＯＭ４０４は、当該技術分野において既知であるようにユーザおよびシステムデータおよびプログラムを保持する。

【００２３】

バス４０２はまた、入出力（Ｉ／Ｏ）コントローラカード４０５、通信アダプタカード４１１、ユーザインタフェースカード４０８およびディスプレイカード４０９に結合される。

Ｉ／Ｏカード４０５は、ハードドライブ、ＣＤドライブ、フロッピー（登録商標）ディスクドライブ、テープドライブのうちの１つまたは複数等の記憶デバイス４０６を、コンピュータシステムに接続する。

記憶デバイス４０６は、クラスタ化アーキテクチャのノードへのトランザクションのルーティングを制御するソフトウェアまたは実行可能コードを格納してもよい。

たとえば、記憶デバイス４０６は、代表的な一実施形態による構成ユーティリティ２４０を実施する実行可能コードを格納してもよい。

【００２４】

通信カード４１１は、コンピュータシステム４００をネットワーク４１２に結合するように適合される。

ネットワーク４１２は、ローカル（ＬＡＮ）、広域（ＷＡＮ）、イーサネット（登録商標）またはインターネットネットワークのうちの１つまたは複数であってもよい。

代替的な一実施形態では、構成ユーティリティ２４０を定義する実行可能命令を、通信カード４１１を通してネットワーク４１２を介して受信してもよい。

ユーザインタフェースカード４０８は、キーボード４１３およびポインティングデバイス４０７等のユーザ入力デバイスをコンピュータシステム４００に結合する。

ディスプレイカード４０９は、ＣＰＵ４０１により、表示デバイス４１０における表示を制御するために駆動される。

【００２５】

トランザクションの分散を制御することにより、いくつかの代表的な実施形態は、複数の利点を提供することができる。

たとえば、同じ数のクラスタノードを使用して、他のアプリケーションの性能を低下させることなくピーク負荷期間中により多くの資源をアプリケーションに提供することができる。

さらに、クラスタシステムにクラスタノードを追加することができ、複雑な手動による構成アクティビティを必要とすることなく、負荷分散機能が追加の資源に対して自動的に調整される。

さらに、いくつかの代表的な実施形態は、性能データ、負荷データおよび他の適当なデ

ータに応じてクラスタシステムの動作を制御するため、クラスタノードは、同一の処理能力または他の能力を所有する必要はない。

特に、アプリケーショントランザクションを、動的に変化するシステムにおいて適当なクラスタノードに自動的に振り向けることができる。

【図面の簡単な説明】

【００２６】

【図１】クラスタシステムを示す図である。

【図２】代表的な一実施形態によるクラスタシステムを示す図である。

【図３】代表的な一実施形態によるクラスタシステムにおいてアプリケーショントランザクションの分散を制御するフローチャートである。

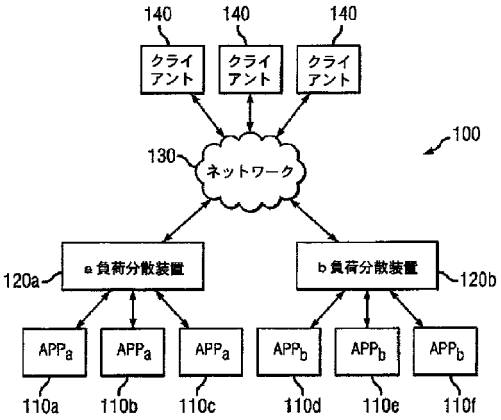
【図４】代表的な一実施形態を実施するために使用されてもよいシステムを示す図である。

【符号の説明】

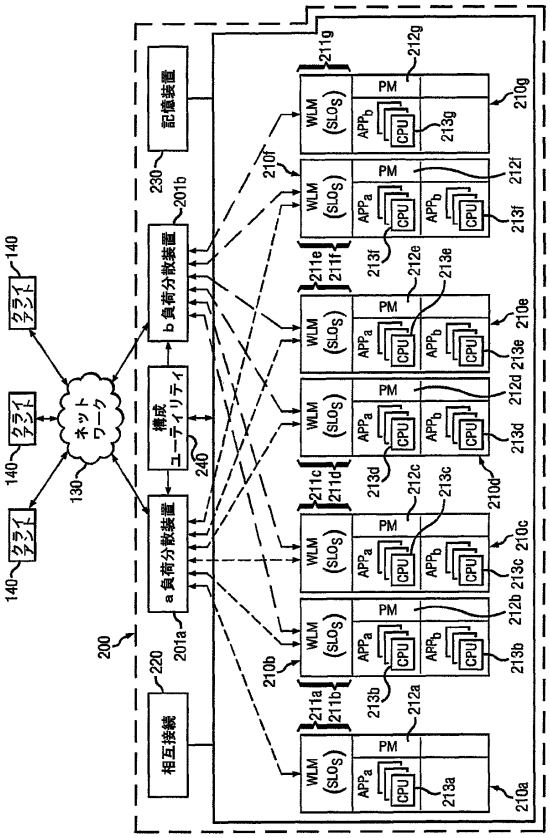
【００２７】

１２０ａ，１２０ｂ・・・負荷分散装置，
１３０・・・ネットワーク，
１４０・・・クライアント，
２０１ａ，２０１ｂ・・・負荷分散装置，
２２０・・・相互接続，
２３０・・・記憶装置，
２４０・・・構成ユーティリティ，
４０５・・・Ｉ／Ｏアダプタ，
４０８・・・ユーザインタフェースアダプタ，
４０９・・・ディスプレイアダプタ，
４１１・・・通信アダプタ，
４１２・・・ネットワーク，

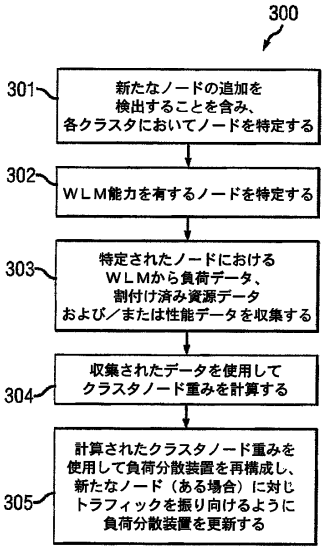
【図１】


140
140
140
クライアント
クライアント
クライアント
100
130
ネットワーク
120a
a 負荷分散装置
b 負荷分散装置
120b
APPa
APPa
APPa
APPb
APPb
APPb
110a
110b
110c
110d
110e
110f

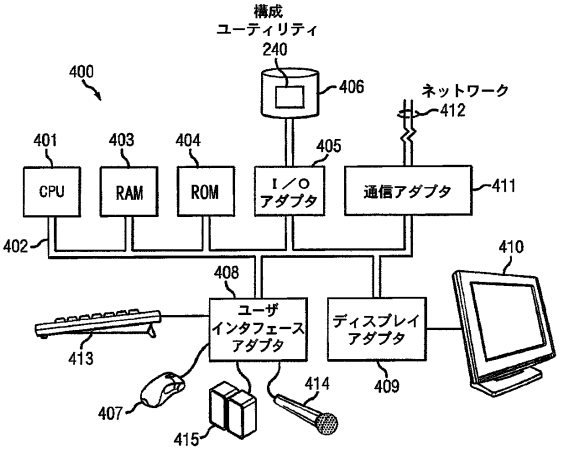
【図２】


記憶装置
230
b 負荷分散装置
201b
構成ユーティリティ
240
a 負荷分散装置
201a
相互接続
220
200
130
140
クライアント
ネットワーク
WLM
(SLOs)
APPa
APPb
CPU
PM
210a
210b
210c
210d
210e
210f
210g
211a
211b
211c
211d
211e
211f
211g
212a
212b
212c
212d
212e
212f
212g
213a
213b
213c
213d
213e
213f
213g

【図３】


300
301
新たなノードの追加を検出することを含み、各クラスタにおいてノードを特定する
302
WLM能力を有するノードを特定する
303
特定されたノードにおけるWLMから負荷データ、割付け済み資源データおよび／または性能データを収集する
304
収集されたデータを使用してクラスタノード重みを計算する
305
計算されたクラスタノード重みを使用して負荷分散装置を再構成し、新たなノード（ある場合）に対しトラフィックを振り向けるように負荷分散装置を更新する

【図４】


構成ユーティリティ
240
400
406
ネットワーク
412
401
403
404
405
CPU
RAM
ROM
I／Oアダプタ
通信アダプタ
411
402
410
408
ユーザインタフェースアダプタ
ディスプレイアダプタ
413
407
414
409
415

フロントページの続き

(72)発明者　ブライアン・バッカー
アメリカ合衆国カリフォルニア州　パロアルト　ハノーバー・ストリート　３０００　ヒューレット・パッカード・カンパニー内

審査官　大塚　良平

(56)参考文献　国際公開第０３／０６３４４７（WO，Ａ１）
特表２００５－５１６２９３（ＪＰ，Ａ）
特開平９－２８２２８７（ＪＰ，Ａ）
特開２００４－１１０７９１（ＪＰ，Ａ）
特開平１０－３３３９２５（ＪＰ，Ａ）
特開平１０－２８３２１１（ＪＰ，Ａ）
特開平１０－１９８６４２（ＪＰ，Ａ）
特開２００３－６７３５１（ＪＰ，Ａ）
佐竹伸介、稲井寛，広域分散Ｗｅｂサーバシステムにおける動的負荷分散方式，電子情報通信学会技術研究報告，日本，社団法人電子情報通信学会，２００３年１０月１７日，第103巻，第388号，pp.57－60
榊幹雄，メインフレームの性能評価手法，情報処理，日本，社団法人情報処理学会，１９９６年８月１５日，第37巻，第８号，pp.745－750

(58)調査した分野(Int.Cl.，ＤＢ名)
Ｇ０６Ｆ　　９／４６　－　　９／５４
Ｇ０６Ｆ　１５／１６　－　１５／１７７