(19)日本国特許庁(JP)　(12)特　許　公　報(B2)　(11)特許番号
特許第5921724号
(P5921724)
(45)発行日　平成28年5月24日(2016.5.24)　(24)登録日　平成28年4月22日(2016.4.22)

(51)Int.Cl.　F I
G06F 9/50 (2006.01)　G06F 9/46 462A

請求項の数 11 (全 29 頁)

(21)出願番号　特願2014-556973(P2014-556973)
(86)(22)出願日　平成25年1月28日(2013.1.28)
(65)公表番号　特表2015-515037(P2015-515037A)
(43)公表日　平成27年5月21日(2015.5.21)
(86)国際出願番号　PCT/EP2013/051563
(87)国際公開番号　WO2013/120686
(87)国際公開日　平成25年8月22日(2013.8.22)
審査請求日　平成26年10月7日(2014.10.7)
(31)優先権主張番号　12305171.6
(32)優先日　平成24年2月15日(2012.2.15)
(33)優先権主張国　欧州特許庁(EP)

(73)特許権者　391030332
アルカテル－ルーセント
フランス国、９２１００・ブローニュ－ビヤンクール、ルート・ドゥ・ラ・レーヌ・１４８／１５２
(74)代理人　100094112
弁理士　岡部　讓
(74)代理人　100106183
弁理士　吉澤　弘司
(74)代理人　100170601
弁理士　川崎　孝
(72)発明者　サッケ，クラウス
ドイツ　７１７３２　タム，アレンシュトラーセ　８８

最終頁に続く

(54)【発明の名称】コンピューティング・デバイスおよび方法

(57)【特許請求の範囲】
【請求項１】
コンピューティング・デバイスであって、
－　対応する複数のアプリケーションの１つまたは複数のコンポーネントに関する複数のコンポーネント配置要求を受信するステップと、
－　前記複数のコンポーネント配置要求から複数の特徴ベクトルをそれぞれ決定するステップであり、それぞれの特徴ベクトルが、前記それぞれのコンポーネント配置要求のさまざまな属性を記述するベクトル次元を含む、ステップと、
－　前記複数のコンポーネント配置要求に関する複数の配置決定をそれぞれ決定するステップであり、それぞれの配置決定が、前記それぞれのアプリケーションの前記１つまたは複数のコンポーネントが配置されている１つまたは複数の実行コンピューティング・デバイスを示すことを含む、ステップと、
－　前記複数の特徴ベクトルをクラスタ化し、それによって１つまたは複数のクラスタを生み出すステップであり、それぞれのクラスタが、デフォルトのコンポーネント配置要求の前記さまざまな属性を記述するデフォルトの特徴ベクトルを含む、ステップと、
－　前記１つまたは複数のクラスタのそれぞれに関してデフォルトの配置決定を決定するステップと、
－　前記１つまたは複数のデフォルトの特徴ベクトルおよび前記それぞれの１つまたは複数のデフォルトの配置決定を前記コンピューティング・デバイスのデータベース内に格納するステップと、

－　新しいアプリケーションの１つまたは複数のコンポーネントに関する新しいコンポーネント配置要求を受信するステップと、
－　前記新しいコンポーネント配置要求から新しい特徴ベクトルを決定するステップと、
－　新しい特徴ベクトルと前記１つまたは複数のデフォルトの特徴ベクトルとの比較に基づいて前記新しいアプリケーションの前記１つまたは複数のコンポーネントをどこに配置するかを決定するステップとを行うように適合される、コンピューティング・デバイス。

【請求項２】
前記クラスタ化が、機械学習アルゴリズム、特にサポート・ベクター・マシン・アルゴリズムを使用して実行される請求項１に記載のコンピューティング・デバイス。

【請求項３】
－　前記複数の特徴ベクトルの第１のベクトル次元が、相関しきい値よりも小さい対応する前記配置決定との相関を有すると判定し、
－　前記複数の特徴ベクトルから前記第１のベクトル次元を除去するように適合される
請求項１乃至２のいずれか１項に記載のコンピューティング・デバイス。

【請求項４】
－　その他のコンピューティング・デバイスから前記複数の配置要求に関連する制御メッセージを受信し、
－　前記受信された制御メッセージに基づいて前記複数の配置決定を決定するように適合される
請求項１乃至３のいずれか１項に記載のコンピューティング・デバイス。

【請求項５】
前記ベクトル次元が、
－　アプリケーション・コンポーネントによって処理されるデータのシンクおよび／またはソースのロケーション、
－　アプリケーションによって処理される複数のシンクおよび／またはソース、
－　アプリケーション・コンポーネントによって必要とされるコンピューティング・リソースであって、プロセッサ・リソース、メモリ・リソース、帯域幅リソースのうちの１つまたは複数である、コンピューティング・リソース、
－　アプリケーション・コンポーネントによって必要とされる接続属性であって、帯域幅、待ち時間、最大ビット誤り率のうちの１つまたは複数である、接続属性、
－　アプリケーションの前記１つまたは複数のコンポーネントのグラフ構造であって、前記アプリケーションの前記１つまたは複数のコンポーネントどうしがどのように連結されているかを示す、グラフ構造のうちの１つまたは複数を示す請求項１乃至４のいずれか１項に記載のコンピューティング・デバイス。

【請求項６】
－　前記１つまたは複数のデフォルトの特徴ベクトルからの前記新しい特徴ベクトルの最短距離を決定し、
－　前記最短距離が最小しきい値を下回る場合、前記新しい特徴ベクトルから前記最短距離にある前記デフォルトの特徴ベクトルに対応する前記デフォルトの配置決定に基づいて前記新しいアプリケーションの前記１つまたは複数のコンポーネントをどこに配置するかを決定する
ように適合される請求項１乃至５のいずれか１項に記載のコンピューティング・デバイス。

【請求項７】
前記最短距離が、前記新しい特徴ベクトルおよび前記１つまたは複数のデフォルトの特徴ベクトルの前記それぞれのベクトル次元の重み付けされた差に基づいて決定される請求項６に記載のコンピューティング・デバイス。

【請求項８】

－　前記デフォルトの配置決定内に示されている実行コンピューティング・デバイスに前記コンポーネント配置要求を渡す

ように適合される請求項６乃至７のいずれか１項に記載のコンピューティング・デバイス。

【請求項９】

－　前記コンピューティング・デバイスが、第１のトポロジーのエリア内に位置付けられ、

－　前記コンピューティング・デバイスが、前記第１のトポロジーのエリア以外の複数のトポロジーのエリアにそれぞれ位置付けられる複数の基準コンピューティング・デバイスを示すトポロジーのリストを含み、

－　前記コンピューティング・デバイスが、前記コンピューティング・デバイス、および前記コンピューティング・デバイスの近隣に位置付けられた少なくとも１つの近隣のコンピューティング・デバイスの利用可能なコンピューティング・リソースを示すローカル・リソース・リストを含み、

－　前記最短距離が最小しきい値よりも大きいと判定すると、前記コンピューティング・デバイスが、

－　前記トポロジーのリストに基づいて、前記新しいアプリケーションの前記１つまたは複数のコンポーネントが、前記第１のトポロジーのエリアに置かれるべきであるか、または前記第１のトポロジーのエリア以外の前記複数のトポロジーのエリアのうちの１つに置かれるべきであるかを判定し、

－　前記新しいアプリケーションの前記１つまたは複数のコンポーネントが前記第１のトポロジーのエリア以外の前記複数のトポロジーのエリアのうちの１つに置かれるべきであると判定される場合、前記第１のトポロジーのエリア以外の前記複数のトポロジーのエリアの前記それぞれのトポロジーのエリアの前記基準コンピューティング・デバイスに前記コンポーネント配置要求を渡し、

－　前記新しいアプリケーションの前記１つまたは複数のコンポーネントが前記第１のトポロジーのエリアに配置されるべきであると判定される場合、前記ローカル・リソース・リストから、前記新しいアプリケーションの前記１つまたは複数のコンポーネントを実行するための前記コンピューティング・リソースを有する選択されたコンピューティング・デバイスを特定するように適合される

請求項６乃至８のいずれか１項に記載のコンピューティング・デバイス。

【請求項１０】

－　前記コンピューティング・デバイスが、ポイントツーマルチポイント、ポイントツーポイント、またはマルチポイントツーマルチポイントのアプリケーションのデフォルトのアプリケーション・サーバであり、

－　前記デフォルトのアプリケーション・サーバが、前記ポイントツーマルチポイント、前記ポイントツーポイント、または前記マルチポイントツーマルチポイントのアプリケーションをセットアップするための複数のコンピューティング・デバイスのクラウド内のデフォルトのアクセス・ポイントである

請求項１乃至９のいずれか１項に記載のコンピューティング・デバイス。

【請求項１１】

新しいアプリケーションの１つまたは複数のコンポーネントをメディア・クラウドのコンピューティング・デバイス上に配置するための方法であって、

－　対応する複数のアプリケーションの１つまたは複数のコンポーネントに関する複数のコンポーネント配置要求を受信するステップと、

－　前記複数のコンポーネント配置要求から複数の特徴ベクトルをそれぞれ決定するステップであり、それぞれの特徴ベクトルが、前記それぞれのコンポーネント配置要求のさまざまな属性を記述するベクトル次元を含む、ステップと、

－　前記複数のコンポーネント配置要求に関する複数の配置決定をそれぞれ決定するステップであり、それぞれの配置決定が、前記それぞれのアプリケーションの前記１つまた

は複数のコンポーネントが配置されている１つまたは複数の実行コンピューティング・デバイスを示すことを含む、ステップと、
－　前記複数の特徴ベクトルをクラスタ化し、それによって１つまたは複数のクラスタを生み出すステップであり、それぞれのクラスタが、それぞれのデフォルトのコンポーネント配置要求の前記さまざまな属性を記述するデフォルトの特徴ベクトルによって表される、ステップと、
－　前記１つまたは複数のクラスタのそれぞれに関して、デフォルトの特徴ベクトルに対応するデフォルトの配置決定を決定するステップと、
－　前記１つまたは複数のデフォルトの特徴ベクトルおよび前記それぞれの１つまたは複数のデフォルトの配置決定を前記コンピューティング・デバイスのデータベース内に格納するステップと、
－　前記新しいアプリケーションの前記１つまたは複数のコンポーネントを配置するために、前記データベース内に格納されている前記１つまたは複数のデフォルトの特徴ベクトルおよび前記それぞれの１つまたは複数のデフォルトの配置決定を使用するステップとを含み、使用するステップが、
－　新しいアプリケーションの１つまたは複数のコンポーネントに関する新しいコンポーネント配置要求を受信するステップと、
－　前記新しいコンポーネント配置要求から新しい特徴ベクトルを決定するステップと、
－　新しい特徴ベクトルと前記１つまたは複数のデフォルトの特徴ベクトルとの比較に基づいて前記新しいアプリケーションの前記１つまたは複数のコンポーネントをどこに配置するかを決定するステップとを含む、方法。
【発明の詳細な説明】
【技術分野】
【０００１】
本明細書は、クラウド・コンピューティングに関する。特に、本明細書は、機械学習を採用するメディア・コンポーネントをマップするための方法、およびクラウド内でのアプリケーション・コンポーネントの効率的で柔軟な配置を可能にするクラウド・コンピューティングのための方法およびシステムに関する。
【背景技術】
【０００２】
インターネットは、ユーザがメディアを消費する方法を変えつつある。インターネット技術によって可能にされて、発展は、ライブ３Ｄ放送、ライブ・イベントの時間をずらした視聴、またはいつでも望まれるとき、どこでも必要とされるところで、任意の好ましいデバイスでのビデオ・オン・デマンドのようなメディア・サービスをユーザが楽しむことを可能にするところに急速に向かっている。さらに、インターネットでは、ユーザは、単なる見物人ではなく没頭している参加者になる。ウェブに基づくサービスは、リアルタイムのマルチメディア・コンテンツの大規模な共有、または没入型のマルチメディア通信のような新しいパーソナライズされたメディア中心のアプリケーションの部類の全体に対する触媒である。これらのサービスは、純粋なコンテンツ・ストリームとして実現されるのではなく、適切な時間、場所、およびフォーマットで要求されたデータを提供するメディア処理機能の編成されたフローとして実現される。高精細ビデオ・フォーマットの導入により、転送されるデータ量が、データ変換プロセスにおけるコードの構築のサイズを上回る。したがって、分散型のサービス・インフラストラクチャにインテリジェントな方法でサービス・コンポーネントを配置することは、将来のインターネット・インフラストラクチャのスケーリングを向上させる方法を提供する。言い換えると、転送されるデータ量が増えるにつれて、分散型アプリケーションのＳＷコンポーネントをクラウド・ネットワーク内の適切な位置に転送し、配置することが一層効率的になる可能性がある。
【先行技術文献】
【非特許文献】

【０００３】
【非特許文献１】Ｏ．Ｎｉｅｈｏｒｓｔｅｒら、「Ａｕｔｏｎｏｍｉｃ Ｒｅｓｏｕｒｃｅ Ｍａｎａｇｅｍｅｎｔ Ｗｉｔｈ Ｓｕｐｐｏｒｔ Ｖｅｃｔｏｒ Ｍａｃｈｉｎｅｓ」、Ｇｒｉｄ Ｃｏｍｐｕｔｉｎｇ、１２ｔｈ ＩＥＥＥ／ＡＣＭ Ｉｎｔｅｒｎａｔｉｏｎａｌ Ｃｏｎｆｅｒｅｎｃｅ ｏｎ、ＩＥＥＥ、２０１１年９月２１日、１５７～１６４頁
【非特許文献２】Ｉｓｈｆａｑ Ａｈｍａｄ、「Ｍｕｌｔｉ－Ｖｉｅｗ Ｖｉｄｅｏ：Ｇｅｔ Ｒｅａｄｙ ｆｏｒ Ｎｅｘｔ－Ｇｅｎｅｒａｔｉｏｎ Ｔｅｌｅｖｉｓｉｏｎ」、ＩＥＥＥ Ｄｉｓｔｒｉｂｕｔｅｄ Ｓｙｓｔｅｍｓ Ｏｎｌｉｎｅ、第８巻、第３号、２００７年、論文番号０７０３－ｏ３００６
【非特許文献３】ＯｎＬｉｖｅ、ｈｔｔｐ：／／ｗｗｗ．ｏｎｌｉｖｅ．ｃｏｍ／
【発明の概要】
【発明が解決しようとする課題】
【０００４】
本明細書は、サービス／アプリケーション・コンポーネントの効率的で柔軟な配置を可能にするコンピューティング・デバイス（ノードとも呼ばれる）のクラウドを提供することの技術的な問題に対処する。特に、本明細書は、コンピューティング・デバイスのクラウド内にアプリケーション・コンポーネントを配置する効率的な方式を提供することの技術的な問題に対処する。
【０００５】
Ｏ．Ｎｉｅｈｏｒｓｔｅｒら、「Ａｕｔｏｎｏｍｉｃ Ｒｅｓｏｕｒｃｅ Ｍａｎａｇｅｍｅｎｔ Ｗｉｔｈ Ｓｕｐｐｏｒｔ Ｖｅｃｔｏｒ Ｍａｃｈｉｎｅｓ」、Ｇｒｉｄ Ｃｏｍｐｕｔｉｎｇ、１２ｔｈ ＩＥＥＥ／ＡＣＭ Ｉｎｔｅｒｎａｔｉｏｎａｌ Ｃｏｎｆｅｒｅｎｃｅ ｏｎ、ＩＥＥＥ、２０１１年９月２１日、１５７～１６４頁は、クラウド・サービスを管理するための人間による介入の度合いを減らすソフトウェア・ソリューションについて記載している。
【課題を解決するための手段】
【０００６】
一態様によれば、分散型クラウド・コンピューティングのために適合されたコンピューティング・デバイス（コンピューティング・ノードまたはノードとも呼ばれる）が説明される。コンピューティング・デバイスは、対応する複数のアプリケーションの１つまたは複数のコンポーネントに関する複数のコンポーネント配置要求を受信するように適合されることが可能である。さらに、コンピューティング・デバイスは、複数のコンポーネント配置要求から複数の特徴ベクトルをそれぞれ決定するように適合されることが可能である。それらの複数の特徴ベクトルのうちの１つの特徴ベクトルは、それぞれのコンポーネント配置要求のさまざまな属性（またはさまざまな側面）を記述するベクトル次元を含む。したがって、特徴ベクトルは、対応するコンポーネント配置要求の構造化されたモデルとして理解されることが可能である。
【０００７】
加えて、コンピューティング・デバイスは、複数のコンポーネント配置要求に関する複数の配置決定をそれぞれ決定するように適合されることが可能である。１つの配置決定は、それぞれのアプリケーションの１つまたは複数のコンポーネントが配置されている１つまたは複数の実行コンピューティング・デバイスの表示を含む。特に、コンピューティング・デバイスは、その他のコンピューティング・デバイスから制御メッセージを受信するように、ならびにそれらの受信された制御メッセージに基づいて複数の配置決定（および場合によっては、特徴ベクトル）を決定するように適合されることが可能である。
【０００８】
したがって、コンピューティング・デバイスは、複数の特徴ベクトルおよび対応する複数の配置決定を含むトレーニング・データを収集するように適合されることが可能である。特に、コンピューティング・デバイスは、コンポーネント配置要求内に、および配置決

定内に含まれている情報を構造化し、それによって、機械学習のために使用されることが可能であるトレーニング・データを提供し、それによって、その後のコンポーネント配置要求の加速された配置を可能にするように適合されることが可能である。

【０００９】

機械学習のコンテキストにおいては、コンピューティング・デバイスは、複数の特徴ベクトルをクラスタ化し、それによって１つまたは複数のクラスタを生み出すように適合されることが可能である。クラスタ化は、機械学習アルゴリズム、例えばサポート・ベクター・マシン・アルゴリズムを使用して実行されることが可能である。それぞれのクラスタは、典型的には、デフォルトのコンポーネント配置要求のさまざまな属性を記述するデフォルトの特徴ベクトルを含む。言い換えれば、クラスタは、デフォルトの特徴ベクトルによって表されることが可能である。さらに、コンピューティング・デバイスは、１つまたは複数のクラスタのそれぞれに関してデフォルトの配置決定を決定するように適合されることが可能である。１つまたは複数のデフォルトの特徴ベクトルおよびそれぞれの１つまたは複数のデフォルトの配置決定は、コンピューティング・デバイスのデータベース内に格納されることが可能である。特に、１つまたは複数のデフォルトの特徴ベクトルおよびそれぞれの１つまたは複数のデフォルトの配置決定は、その後の配置決定のために使用されることが可能である。

【００１０】

コンピューティング・デバイスは、機械学習のコンテキストにおいてベクトル次元の数を減らすように適合されることが可能であり、すなわち、コンピューティング・デバイスは、ベクトル空間の次元を減らし、それによって、その後のコンポーネント配置要求を処理するための全体的な複雑さを減らすように適合されることが可能である。特に、コンピューティング・デバイスは、複数の特徴ベクトルの第１のベクトル次元が、相関しきい値よりも小さい対応する複数の配置決定との相関を有すると判定するように適合されることが可能である。言い換えれば、コンピューティング・デバイスは、配置決定にほとんどからまったくまでの影響を及ぼさない第１のベクトル次元を決定することができる。そのような第１のベクトル次元が決定された場合、その第１のベクトル次元が複数の特徴ベクトルから除去されることが可能である。

【００１１】

ベクトル次元は、アプリケーション・コンポーネントによって処理されるデータのシンクおよび／もしくはソースのロケーション、アプリケーションによって処理される複数のシンクおよび／もしくはソース、アプリケーション・コンポーネントによって必要とされるコンピューティング・リソース（それらのコンピューティング・リソースは、プロセッサ・リソース、メモリ・リソース、帯域幅リソースのうちの１つもしくは複数であることが可能である）、アプリケーション・コンポーネントによって必要とされる接続属性（それらの接続属性は、帯域幅、待ち時間、最大ビット誤り率のうちの１つもしくは複数であることが可能である）、ならびに／またはアプリケーションの１つもしくは複数のコンポーネントのグラフ構造（そのグラフ構造は、アプリケーションの１つもしくは複数のコンポーネントどうしがどのように連結されているかを示すことができる）のうちの１つまたは複数を示すことができる。ベクトル次元は、それらのベクトル次元を互いに比較可能にするための適切なメトリックを有することができる。

【００１２】

コンピューティング・デバイスは、新しいコンポーネント配置要求を処理するために、格納されているデフォルトの特徴ベクトルおよび対応するデフォルトの配置決定を利用するように適合されることが可能である。特に、コンピューティング・デバイスは、新しいアプリケーションの１つまたは複数のコンポーネントの配置に関する新しいコンポーネント配置要求を受信するように適合されることが可能である。（上述の機械学習フェーズのコンテキストにおいて行われるのと同様の様式で）新しいコンポーネント配置要求から新しい特徴ベクトルが決定されることが可能である。次いで、コンピューティング・デバイスは、コンピューティング・デバイスにおいて格納されている１つまたは複数のデフォル

トの特徴ベクトルに基づいて新しいアプリケーションの１つまたは複数のコンポーネントをどこに配置するかを決定することができる。特に、コンピューティング・デバイスは、１つまたは複数のデフォルトの配置決定のうちの１つと同じ（または同様の様式）で新しい配置要求が処理されることが可能であるかどうかを判定するために、新しい特徴ベクトルを１つまたは複数のデフォルトの特徴ベクトルと比較することができる。

【００１３】

コンピューティング・デバイスは、１つまたは複数のデフォルトの特徴ベクトルからの新しい特徴ベクトルの最短距離を決定するように適合されることが可能である。新しい特徴ベクトルと、デフォルトの特徴ベクトルとの間の距離は、ベクトル次元のそれぞれのメトリックに基づいて決定されることが可能である。例として、距離は、新しい特徴ベクトルおよび１つまたは複数のデフォルトの特徴ベクトルのうちの１つのそれぞれのベクトル次元の重み付けされた差に基づいて決定されることが可能である。最短距離が最小しきい値を下回る場合、コンピューティング・デバイスは、新しい特徴ベクトルから最短距離にあるデフォルトの特徴ベクトルに対応するデフォルトの配置決定に基づいて新しいアプリケーションの１つまたは複数のコンポーネントをどこに配置するかを決定することができる。特に、コンピューティング・デバイスは、デフォルトの配置決定内に示されている実行コンピューティング・デバイスにコンポーネント配置要求を渡すことができる。

【００１４】

上で概要を説明されているように、本明細書は、複数のコンピューティング・デバイスを含むメディア・クラウドに関する。さらに本明細書は、分散型の配置方式に関し、その方式では、それぞれのコンピューティング・デバイスは、新しいアプリケーションの１つまたは複数のコンポーネントの配置に関して個々の決定を行うように構成されている。配置決定は、（機械学習を使用して）前の配置決定から導出されたデフォルトの配置決定に基づいて行われることが可能である。代替として、または追加として、コンピューティング・デバイスは、個々の配置決定を行うために、コンピューティング・デバイスにおいて利用可能なトポロジーの情報および／またはリソース情報を利用するように適合されることが可能である。

【００１５】

コンピューティング・デバイスは、第１のトポロジーのエリア（ｔｏｐｏｌｏｇｉｃａｌ　ａｒｅａ）内に位置付けられる可能性がある。通常、複数のそのようなコンピューティング・デバイスを含む分散型クラウド（本明細書においてはメディア・クラウドと呼ばれる）は、複数のトポロジーのエリアに分けられる（それらのエリアは１つまたは複数の領域にさらに細分される可能性がある）。コンピューティング・デバイスは、第１のトポロジーのエリア以外の複数のトポロジーのエリア内にそれぞれ位置付けられる複数の基準コンピューティング・デバイス（ｒｅｆｅｒｅｎｃｅ　ｃｏｍｐｕｔｉｎｇ　ｄｅｖｉｃｅ）を示すトポロジーのリスト（ｔｏｐｏｌｏｇｉｃａｌ　ｌｉｓｔ）を含む。言い換えると、コンピューティング・デバイスは、分散型クラウドのその他のエリア（または領域）のそれぞれの中に位置付けられた少なくとも１つの基準コンピューティング・デバイスの指示（例えば、ネットワーク識別子）を提供するトポロジーのリストを保有する。トポロジーのリストは、その他のエリア（または領域）毎に１つまたは２つの基準コンピューティング・デバイスを含む可能性がある。通常は、基準コンピューティング・デバイスは、各コンピューティング・デバイスが領域に対する異なるアンカー・ポイント（ａｎｃｈｏｒ　ｐｏｉｎｔ）を有することを保証し、それによって単一障害点を取り除くために、領域内のコンピューティング・デバイスの利用可能なリストからランダムに選択される。

【００１６】

コンピューティング・デバイスは、コンピューティング・デバイス、およびコンピューティング・デバイスの近隣に位置付けられた少なくとも１つの近隣のコンピューティング・デバイスの利用可能なコンピューティング・リソースを示すローカル・リソース・リストをさらに含む可能性がある。コンピューティング・デバイスの近隣は、少なくとも１つの近隣のコンピューティング・デバイスによって満たされる必要がある１つまたは複数の

近隣の条件によって定義される可能性がある。１つまたは複数の近隣の条件は、コンピューティング・デバイスと少なくとも１つの近隣のコンピューティング・デバイスとの間の最大往復遅延時間を含む可能性がある。代替的にまたは追加的に、１つまたは複数の近隣の条件は、少なくとも１つの近隣のコンピューティング・デバイスが第１のトポロジーのエリア、つまり、同じトポロジーのエリア内に位置付けられているという条件を含む可能性がある。

【００１７】

最短距離が最小しきい値よりも大きいと判定すると（すなわち、コンピューティング・デバイスが前の配置決定に依存することができないと判定すると）、コンピューティング・デバイスは、トポロジーのリスト（のみ）に基づいて、新しいアプリケーションの１つまたは複数のコンポーネントが、第１のトポロジーのエリアに置かれるべきであるか、または第１のトポロジーのエリア以外の複数のトポロジーのエリアのうちの１つに置かれるべきであるかを判定するように適合されることが可能である。概して、新しいコンポーネント配置要求は、コンポーネント／アプリケーションのシンクまたはソースの好ましい場所に関する情報を含む。例として、１つまたは複数のコンポーネントのシンクまたはソースの好ましい場所に関する情報は、１つまたは複数のコンポーネント、および１つまたは複数のコンポーネントが連係して動作しているアプリケーションのその他のコンポーネントの要件の記述から導出され得る。コンピューティング・デバイスは、好ましい場所をそのコンピューティング・デバイス自身の場所およびメディア・クラウドのトポロジーのエリアの場所と比較することができる。

【００１８】

１つまたは複数のコンポーネントが第１のトポロジーのエリア以外の複数のトポロジーのエリアのうちの１つに置かれるべきであると判定される場合、コンピューティング・デバイスは、第１のトポロジーのエリア以外の複数のトポロジーのエリアのそれぞれのトポロジーのエリアの基準コンピューティング・デバイスにコンポーネント配置要求を渡すように適合される可能性がある。言い換えると、別のトポロジーのエリアが好ましい場所により近いと判定される場合、配置要求は、別のトポロジーのエリアの基準コンピューティング・デバイスに（または基準コンピューティング・デバイスのうちの１つに）渡され、基準コンピューティング・デバイス（つまり、基準コンピューティング・デバイスへの指示）は、コンピューティング・デバイスのトポロジーのリストから得られる。したがって、コンピューティング・デバイスは、別のコンピューティング・デバイスまたは上位レベルのネットワーク管理エンティティと調整することを必要とせずに、そのコンピューティング・デバイスのトポロジーのリストに基づいてトポロジー管理タスクを実行するように適合される可能性がある。言い換えると、トポロジー管理タスクは、コンピューティング・デバイスによって自律的に実行される可能性がある。

【００１９】

１つまたは複数のコンポーネントが第１のトポロジーのエリアに配置されるべきであると判定される場合、コンピューティング・デバイスは、ローカル・リソース・リストから、新しいアプリケーションの１つまたは複数のコンポーネントを実行するためのコンピューティング・リソースを有する選択されたコンピューティング・デバイスを特定するように適合される可能性がある。言い換えると、コンピューティング・デバイスが、コンポーネントの好ましい場所が第１のトポロジーのエリア内に（またはコンピューティング・デバイスの領域内に）あることを判定する場合、コンピューティング・デバイスは、別のコンピューティング・デバイスまたは上位レベルのネットワーク管理エンティティと調整することを必要とせずに、そのコンピューティング・デバイスのローカル・リソース・リスト内の利用可能なリソース情報に基づいてリソース管理タスクを実行する可能性がある。これは、リソース管理タスクがコンピューティング・デバイスによって自律的に実行される可能性があることを意味する。

【００２０】

デフォルトの配置決定が特定された場合、およびデフォルトの配置決定内で特定された

実行コンピューティング・デバイスに新しい配置要求が渡された場合、トポロジーのリストおよび／またはローカル・リソース・リストに基づく新しい配置要求の上述の処理が使用されることも可能であるということに留意されたい。特に、実行コンピューティング・デバイスは、新しいアプリケーションの１つまたは複数のコンポーネントの処理をさらに最適化するために、自分のトポロジーのリストおよび／または自分のローカル・リソース・リストを利用することができる。

【００２１】

コンピューティング・デバイスは、ポイントツーマルチポイント、ポイントツーポイント、またはマルチポイントツーマルチポイントのアプリケーションのデフォルトのアプリケーション・サーバであることが可能である。そのようなデフォルトのアプリケーション・サーバは、ポイントツーマルチポイント、ポイントツーポイント、またはマルチポイントツーマルチポイントのアプリケーションをセットアップするための複数のコンピューティング・デバイスのクラウド内のデフォルトのアクセス・ポイントであることが可能である。そのようなデフォルトのアプリケーション・サーバに関する例は、例えば、特定のトポロジーのエリア内の中央ビデオ会議サーバ、特定のトポロジーのエリア内の中央放送サーバ、および／または特定のトポロジーのエリア内の中央コール処理サーバである。

【００２２】

別の態様によれば、デフォルトの配置決定のデータベースを決定するための方法、および／または新しいアプリケーションの１つもしくは複数のコンポーネントをコンピューティング・デバイスのクラウドのうちの１つのコンピューティング・デバイス上に配置するための方法が説明される。方法は、対応する複数のアプリケーションの１つまたは複数のコンポーネントに関する複数のコンポーネント配置要求を受信するステップを含む。方法は、複数のコンポーネント配置要求から複数の特徴ベクトルをそれぞれ決定するステップに進み、それぞれの特徴ベクトルは、それぞれのコンポーネント配置要求のさまざまな属性を記述するベクトル次元を含む。さらに方法は、複数のコンポーネント配置要求に関する複数の配置決定をそれぞれ決定するステップに進み、それぞれの配置決定は、それぞれのアプリケーションの１つまたは複数のコンポーネントが配置されている１つまたは複数の実行コンピューティング・デバイスの表示を含む。したがって方法は、機械学習を可能にするために（複数の特徴ベクトルによって表される）トレーニング配置要求および配置決定のセットを決定するステップを含む。方法は、複数の特徴ベクトルをクラスタ化し、それによって１つまたは複数のクラスタを生み出すステップを含み、それぞれのクラスタは、デフォルトのコンポーネント配置要求のさまざまな属性を記述するデフォルトの特徴ベクトルを含む。加えて、１つまたは複数のクラスタのそれぞれに関して、デフォルトの配置決定が決定される。１つまたは複数のデフォルトの特徴ベクトルおよびそれぞれの１つまたは複数のデフォルトの配置決定は、コンピューティング・デバイスのデータベース内に格納される。したがって、デフォルトの配置決定のデータベースが決定されることが可能である。さらに、データベース内に格納されている１つまたは複数のデフォルトの特徴ベクトルおよびそれぞれの１つまたは複数のデフォルトの配置決定は、新しいアプリケーションの１つまたは複数のコンポーネントを配置するために使用されることが可能である。

【００２３】

さらなる態様によれば、ソフトウェア・プログラムが説明される。ソフトウェア・プログラムは、プロセッサでの実行のために適合され、コンピューティング・デバイスで実行されるときに、本明細書において概要を説明される方法のステップを実行するように適合される可能性がある。

【００２４】

別の態様によれば、ストレージ媒体が説明される。ストレージ媒体は、プロセッサでの実行のために適合され、コンピューティング・デバイスで実行されるときに本明細書において概要を説明される方法のステップを実行するように適合されたソフトウェア・プログラムを含み得る。

【００２５】
さらなる態様によれば、コンピュータ・プログラム製品が説明される。コンピュータ・プログラムは、コンピュータで実行されるときに本明細書において概要を説明される方法のステップを実行するための実行可能命令を含み得る。
【００２６】
本明細書において概要を説明される方法およびシステムの好ましい実施形態を含む方法およびシステムは、単体で、または本明細書において開示されるその他の方法およびシステムと組み合わせて使用され得ることに留意されたい。さらに、本明細書において概要を説明される方法およびシステムのすべての態様は、任意に組み合わされ得る。特に、請求項の特徴は、互いに任意に組み合わされ得る。
【００２７】
本発明が、添付の図面を参照して以下で例示的に説明される。
【図面の簡単な説明】
【００２８】
【図１】クラウド内のコンピューティング・ノードの例示的な配置を示す図である。
【図２】複数のコンピューティング・ノードの領域のグループ分けの例示的な図である。
【図３】コンピューティング・ノードの例示的なリソースおよびトポロジー・グラフを示す図である。
【図４】アプリケーションの例示的なコンポーネントを示す図である。
【図５】アプリケーション配置状況を記述する例示的なベクトル空間を示す図である。
【図６】アプリケーション配置のための例示的な学習方法を示すフローチャートである。
【図７】機械学習を利用するアプリケーション配置のための例示的な方法を示すフローチャートである。
【発明を実施するための形態】
【００２９】
今日まで、ネットワークの増大する転送容量の需要は、主に、技術の飛躍的進歩かまたは新しいインフラストラクチャの要素の設置かのどちらかによってネットワークの設置される帯域幅を広げることによって達せられている。しかし、増大する容量の需要に影響されるネットワークのこの発展は、少なくとも無理のないコストでは継続すると期待され得ないという大きな懸念がある。将来のネットワークの改良がますます難しくなるので、増大する容量の需要を満たすための代替的な手法が必要とされている。ネットワーク容量の増大する需要に対処する十分に確立された手法は、ネットワークに「上位層の（ｈｉｇｈｅｒ　ｌａｙｅｒ）」インテリジェンスを追加することである。追加される「上位層の」インテリジェンスは、例えば、トラフィックを局所化することによって全体的なトラフィックを削減し、したがって、利用可能な転送量を増やすことを目標とする。「上位層の」インテリジェンスのこの概念の最初の成功は、コンテンツ・デリバリ・ネットワーク（ＣＤＮ）の導入であった。ＣＤＮは、基本的に、インターネットにおいて放送配信の特徴を含む（メディア）サービスの大規模な採用を可能にする。
【００３０】
しかし、メディア・ストリームがインターネット内のどこかで、すなわち、クラウドで処理を受けることを要し、それによって、例えば、ＩＰ－ＴＶをパーソナライズされた「多視点」ビデオ・サービスに発展させることを可能にする（例えば、Ｉｓｈｆａｑ　Ａｈｍａｄ、「Ｍｕｌｔｉ－Ｖｉｅｗ　Ｖｉｄｅｏ：Ｇｅｔ　Ｒｅａｄｙ　ｆｏｒ　Ｎｅｘｔ－Ｇｅｎｅｒａｔｉｏｎ　Ｔｅｌｅｖｉｓｉｏｎ」、ＩＥＥＥ　Ｄｉｓｔｒｉｂｕｔｅｄ　Ｓｙｓｔｅｍｓ　Ｏｎｌｉｎｅ、第８巻、第３号、２００７年、論文番号０７０３－ｏ３００６参照）か、または「ＯｎＬｉｖｅ」（例えば、ＯｎＬｉｖｅ、ｈｔｔｐ：／／ｗｗｗ．ｏｎｌｉｖｅ．ｃｏｍ／参照）のようなクラウドに基づくゲーム・サービスを可能にする必要があるパーソナライズされたメディア・ストリームへの潮流が起こりつつある。ＣＤＮは同じコンテンツを多数の受信者に効率的に配信するために構築されるが、ネットワーク内での処理を必要とする個別化されたコンテンツ・ストリームの新しい潮流は、

インターネット・インフラストラクチャを課題としている。

【００３１】

今日のアプリケーションおよび対応するクラウド・インフラストラクチャは、概して、データが、アプリケーションが実行される専用の場所（すなわち、データセンター）にネットワークを通じて移動させられるように設計される。将来のインターネット設計にこのクラウド・コンピューティングのパラダイムを残すことは、メディア・ストリームの処理機能が置かれている「任意の」データセンターに転送される必要がある膨大な量のトラフィックをもたらすこととなる。本明細書では、指定されたデータセンターでの集中型のアプリケーション処理のこのパラダイムを変更することが提案される。特に、アプリケーションの要件に従ってアプリケーションまたはアプリケーションの一部の移動を施行するインテリジェントなインフラストラクチャが提案される。そのような方式は、トラフィックを局所化することによってネットワークからの不必要な「長距離」トラフィックをオフロードすることができ、したがって、将来のネットワークにおける転送容量の限られた可用性の問題を克服するのに役立つ。

【００３２】

今日のクラウド・インフラストラクチャでも、コンピューティング・インフラストラクチャをインターネットにオフロードすることが、欠かせないものとなっている。Ａｍａｚｏｎ　ＥＣ２、Ｒａｃｋｓｐａｃｅ、またはＭｉｃｒｏｓｏｆｔ　Ａｚｕｒｅのようなクラウド・コンピューティング・プロバイダは、自身のインフラストラクチャまたはプラットフォームを、ＦａｃｅｂｏｏｋまたはＡｎｉｍｏｔｏのようなインターネットに基づくサービスの極めて動的なニーズをサポートする、自動化されたスケーラビリティおよび即座の展開のような特徴を提供するサービスとして提供する。

【００３３】

しかし、今日の手法は、大きなコストがかかり、トラフィックを局所的に保つのではなくより多くのトラフィックが（クラウド・コンピューティング・プロバイダによって提供される）集中型のデータセンターにルーティングされるのでコア・ネットワークに対する全体的な負荷を増やす。集中型のデータセンターは、データを処理し、そのデータを要求元に送り返す。これはこれまでの要求／応答に基づくウェブ・サービスに関しては実現できそうであるが、この集中型の手法は、パーソナライズされたＭｕｌｔｉＶｉｅｗビデオのレンダリングのような巨大なメディア中心のリアルタイム・アプリケーションのための実際のインターネット・アーキテクチャの設計をだめにする可能性がある。

【００３４】

インターネットが、開発者およびエンド・ユーザがそれらの開発者およびエンド・ユーザのパーソナライズされたアプリケーションを統合されたネットワークおよびコンピューティング・インフラストラクチャで実行することを可能にするサービス指向のコンピューティング・パラダイムを直接サポートする固有の能力を組み込むことが、提案される。

【００３５】

自律的なサービスが、そのようなアプリケーションが構築され得るコンポーネントであるべきである。自律的なサービスは、そのマシン上でのそれらのサービスの物理的なインスタンス化によってアドレス指定される特定のホスト・インフラストラクチャ・ハードウェアに拘束されるべきでなく、分散型のコンピューティング・リソース上に動的に展開され、データ・フローのソースとシンクとの間でデータ・フローに対して並べて配置（ｃｏｌｌｏｃａｔｅ）され得る移動可能なオブジェクトになるべきである。

【００３６】

自律的なサービスは、サービスの動的な組み立てと、ワークフローまたはコンテキスト（ｃｏｎｔｅｘｔ）の条件が変わる場合のサービスの潜在的な適合または再配置とを可能にするために、サービスの明確に定義された（ｗｅｌｌ－ｄｅｆｉｎｅｄ）抽象化モデルを利用する可能性がある。サービス・コンポーネントの疎結合が、要求に応じてサービスのワークフローの相互接続を可能にすべきであり、（サービスおよびそれらのサービスのインターフェースが何らかの意味的なサービスの記述を有するとすれば）サービスの組み

立てを修正することによって、ユーザに同じ関連するデータを提供するために必要とされるワークフローの適合を容易にすべきである。

【００３７】

ユーザの視点から見ると、通常、クラウドは、集中型のサーバのように振る舞う。それにもかかわらず、通常、クラウドは、一貫した方法で、空きリソースの集約されたまたは分散された集合を利用する。計算負荷およびネットワーク・リソースを監視することによって、インスタンスを動的に拡大および縮小し、必ずしもデータ・パスにＱｏＳ（サービス品質）管理メカニズムを適用することなくネットワーク負荷を管理することができる。

【００３８】

特にメディア・アプリケーションにおいて、そのようなコンポーネントは、別のデータ・ストリームを生成するためにデータを消費するデータ変換サービス、すなわち、エンティティとして実装される可能性がある。言い換えると、メディア・アプリケーションは、一連のデータ変換サービスとしてモデル化され得る。したがって、ビデオ・カメラは、ビデオ・データを生成したデータ・ソースである。ビデオ・コーデック、スケーリング・コンポーネント、またはフレーム化コンポーネントのようなビデオ処理コンポーネントは、メディア・ストリームを、例えば、モバイル端末またはＴＶディスプレイのためのふさわしいフォーマットに適合するためのデータの変換を可能にする可能性がある。画像認識は、ビデオ信号からオブジェクトを特定することができ、それらのオブジェクトは、シーンの３Ｄモデルを生成するために異なるソースからマージされる可能性がある。

【００３９】

そのようなデータ変換モデルおよびカメラの元のビデオ・ストリームを用いて、ユーザのための新しいパーソナライズされた像がレンダリングされ、ディスプレイに送信される可能性がある。そのようなサービスは、有向グラフによって表される可能性があり、展開時にインスタンス化されることになる。インスタンス化プロセス中に、必要とされるリソースが、利用可能なリソース・プールから選択される。必要とされるリソースをインスタンス化プロセス中に選択することの結果として、ネットワークに対してサービスによって課されるトラフィック全体が削減される。言い換えると、リソース選択プロセスは、ネットワークに対してサービスによって課されるトラフィック全体を削減することを目的とする可能性がある。リソース選択プロセスは、サービスの消費者のＱｏＥ（体感品質）の点を最適化することをさらに考慮する可能性がある。

【００４０】

さまざまなサービスの特徴を有するアプリケーションは、メディア・クラウド（ＭＣ）の概念から恩恵を受ける程度が異なる可能性がある。大きな恩恵は、特定の期間にわたり連続的なデータの安定したフローを必要とするアプリケーション、または処理のために大量のデータの転送を必要とするアプリケーションで実現され得る。一方、ごく限られたデータの転送しか必要としないアプリケーションに関しては、サービスの転送のオーバーヘッドおよびインスタンス化のコストが、得られる恩恵を上回る可能性がある。したがって、データに関連する「メタ情報」の取得を可能にするメカニズムを提供することが、ＭＣの概念で有益である可能性がある。データに関連するそのような「メタ情報」は、データが一定のメディア（例えば、ビデオ）ストリームであるか、またはサービスの実行前に転送される必要がある限られた量のデータ（例えば、データ・ファイル）に過ぎない場合、データがどこにあるか、サービスの実行のためにどれだけのデータが転送される必要があるかに関する情報を提供する可能性がある。

【００４１】

ネットワーク・アーキテクチャによって本質的にメディア・クラウドの筋書きをサポートするために、既存のインターネット・アーキテクチャのいくつかの基本的な原理が再考されるべきである。第１に、コンテンツ・ネットワーキングのよく知られている原理が、本明細書に記載のＭＣの手法をサポートするように拡張されるべきである。コンテンツ・ネットワークは、データの局所性を探る、つまり、ソースにおけるデータに対する要求を満たすのではなく、データのローカルのキャッシュされたコピーが配信される。コンテン

ツを直接アドレス指定し、コンテンツが生成された場所をルーティングの決定のために使用するのではなくこの情報をルーティングの目的に使用する方式が、提案され得る。

【００４２】

上述の方式の拡張は、コンテンツをアドレス指定すべきであるだけでなく、さらに、要求されたデータを提供することができるサービスをアドレス指定し、必要な変換を行うための処理パイプラインをインスタンス化すべきである。単一のドメイン内のすべてのユーザに集中型の処理を実行する代わりに、メディア・フローは、利用できる場合、ネットワーク・レイヤの固有の「マルチキャスト」能力を利用して適切な場所で組み合わされるか、または分割される可能性がある。これは、マルチキャストが、「マルチキャスト」がネットワークでサポートされるかどうかを知らないサービスの開発者によって明示的に組み込まれる必要があり、したがって、オーバーレイ・メカニズムによってのみ実現され得る既存の方式よりも有益である。

【００４３】

異なるサービス・コンポーネント間で交わされる（メディア）フローのトラフィック・パターンが正確に予測される場合、本明細書に記載のＭＣ対応ネットワークは、パケット毎にルーティングの決定を実行する代わりに、直接、そのようなフロー・パターンに対して動作することができる。したがって、ＭＣ対応ネットワークは、サービス・コンポーネント間で交わされるメディア・ストリームの利用可能なメタ情報をネットワークに提供することによって効率的なフローに基づくスイッチングを可能にすることができる。この情報は、そのようなＭＣ対応ネットワークの制御プレーンが全体的なスループットを向上させることを可能にすることができる。

【００４４】

ＭＣ方式は、フローに基づくスイッチング・パラダイムが、概して、フロー制御ハンドラにおいてより大きな動的さをサポートすることを代償として実現されることをやはり考慮し得る。そのような代償を「制限する」ために、ＭＣ対応ネットワークは、同じデータセンターで実行されるサービス・コンポーネント間のパスを共有している複数のデータ・ストリームを集約する能力を提供すべきである。データセンター間のジョイント・ストリーム（ｊｏｉｎｔ　ｓｔｒｅａｍ）の集約された粒度（ｇｒａｎｕｌａｒｉｔｙ）を導入することによって、コア・ネットワーク自体の制御の複雑性が制限され得る。

【００４５】

ＭＣを提供するときのネットワークに対するさらなる要件は、もはやマシンではなくサービス（すなわち、サービス・コンポーネント）であるメディア・フローのエンドポイントの途切れることのない再配置がサポートされるようにしてフローがネットワークで処理されるべきであることである。したがって、ＭＣ対応ネットワークに関しては、ソケット・インターフェースのようなクライアントのＡＰＩが修正される必要がある可能性がある。ＭＣ対応サービスは、概してデータの（１つまたは複数の）入力ストリームに対して動作して、このサービスの後続のコンシューマ・コンポーネント（ｃｏｎｓｕｍｅｒ　ｃｏｍｐｏｎｅｎｔ）にその後配信される自身の出力データを生成する自己充足的なコンポーネントから構築されるので、通信を目的とする専用のソケットの使用はもはや十分ではない可能性があり、新しいパラダイムが将来のインターネットに関連して考えられる必要がある可能性がある。

【００４６】

典型的には、通常のクラウド・アプリケーションの処理リソースは、サービス・ランタイムの前に割り振られなければならない。このマッピングは、どのデータセンターが特定のアプリケーションをホストすべきかを決定するという手動による管理タスクである場合が最も多い。結果として、処理されるデータの源のロケーションにかかわらず、処理されるデータは、典型的には、自分の事前に割り振られたデータセンターへ送信され、そこで処理され、自分の送り先へ送り出されなければならない。そのような静的な配置方式は、ランタイムにおいてサービス実行を動的に適合させることによってサービス・パフォーマンスおよびリソース利用を改善する、分散型のメディア・アプリケーション展開の利点を

活用するには十分でない。例えば、グローバル・ビデオ会議サービスは、そのビデオ会議サービスのユーザの使用パターンに従って拡大および縮小すること、またはより適切なロケーションへ移動することを可能にされることが可能である。リソースのこの適合は、ＭＣ内でビデオ会議サービスのアプリケーション・コンポーネントを作成することによって、取り下げることによって、または移動させることによって達成されることが可能である。本明細書は、アプリケーション・コンポーネントに関するそのような配置および再配置の決定を効率よく提供し、その一方で、サービス・スケーリングおよびコンポーネントの局所性について考慮する複雑なサービス・ロジック（ミドルウェア）を回避することを対象としている。

【００４７】

少なくともユーザ毎のコンポーネントの粒度で機能する真に分散型のサービス展開に関しては、教育されたコンポーネント配置決定は、典型的には、サービス・ランタイム中にのみ達成されることが可能である。その時点においてのみ、関連するメディア・ストリームのデータ・ソースおよびデータ・シンクがわかり、ひいては、メディア処理コンポーネントが、すぐ近くの処理リソース・ロケーションに割り振られることが可能であり、それにより、トラフィックをローカルに保持することによって、エンドツーエンドのサービス待ち時間が少なくなり、ネットワークがオフロードされる結果となる。変わりゆくサービス・プロファイルおよび変動するトラフィック・パターンは、サービス・ランタイム中にコンポーネントに関するリソース再割り振りを必要とする可能性さえある。したがって効率的なコンポーネント配置は、典型的には、アプリケーション・コンポーネントがインスタンス化されるときの単発のマッピング決定を必要とするだけでなく、典型的には、リソース割り振りの継続的な評価も必要とする。

【００４８】

本明細書に記載のメディア・クラウドは、典型的には、軽量のアプリケーション・コンポーネント（すなわち、相対的に小さなデータ・サイズを有するアプリケーション・コンポーネント）を使用し、効果的なメディア処理に適合されているフローベースのプログラミング・モデルに準拠する分散型のオーケストレーション・ミドルウェアを提供する。アプリケーション・コンポーネントの小さなサイズに起因して、コンポーネントは、実行時間中に柔軟に展開可能であり、最も低いコストで最も高いユーザ経験を提供するためにアプリケーション・コンポーネントが最も効果的に稼働するロケーションに配置されることが可能である。

【００４９】

アプリケーション・コンポーネントは、内部の機能ロジックを外部のアプリケーション・ロジックから切り離すことができる。典型的には、アプリケーション・コンポーネントは、ストリーミングされるメディアの新しいインスタンスの着信によって、例えば、ビデオ・フレームの着信によってトリガーされる。それに応じて、メッセージが、アプリケーション・コンポーネントを実行環境に接続する決定されたコネクタ（メディア・クラウドにおいては、「ポート」と呼ばれる）を介して適切なアプリケーション・コンポーネントへ配信される。新しいビデオ・フレームが受信された場合、そのビデオ・フレーム上で機能するアプリケーション・コンポーネントは、例えば、その受信されたフレームをＲＧＢ（Ｒｅｄ－Ｇｒｅｅｎ－Ｂｌｕｅ）色空間からグレースケール・イメージへ変換し、２つの入力ストリームからのフレームどうしを単一の出力イメージへとマージし、または顔検知を実行する。

【００５０】

アプリケーション・コンポーネントは、自分のオペレーションを終了して結果を生成したときには常に、処理されたデータを、アプリケーション・コンポーネントによって生成されたデータの潜在的な消費者へ転送するために実行環境を呼び出す。その実行環境は、生成された出力の消費者を特定および決定し、データ転送を処理する。実行環境によるこの処理は、アプリケーション・コンポーネントから外部バインディングを隠し、実行時間においてさえ、全体的なアプリケーションの動的なバインディングおよび再構成を可能に

する。送信者および受信者が整合した方法でメッセージを解釈していることを確実にするために、さまざまなポート・タイプが考慮されることが可能である。

【００５１】

本明細書に記載のＭＣコンテキスト内では、全体的なアプリケーション・ロジックは、アプリケーション・コンポーネントを作成または解任する、およびアプリケーション・コンポーネントどうしの間の接続を構築または解除する制御コンポーネントによって確立される。制御コンポーネントは、典型的にはメディア・パスに関与しないが、構成パラメータを提供し、制御コマンドを自分の子コンポーネントへ送信し、またはエラー・メッセージを受信するために、特別な制御ポートを通じてそれらの子コンポーネントに接続されている。

【００５２】

典型的には、新しいアプリケーション・コンポーネントは、対応する制御コンポーネントが存在する同じローカル・リソースにおいてインスタンス化される。メディア・フロー内のその他のアプリケーション・コンポーネントへの接続が確立された後に、実行環境は、上で概要を説明されているようにマッピング・プロセスを呼び出し、アプリケーション・コンポーネントおよびそのアプリケーション・コンポーネントの状態を、パフォーマンス、待ち時間、またはその他の理由から、そのアプリケーション・コンポーネントをホストするのにより適しているリソースへ転送する。

【００５３】

アプリケーション・コンポーネントに基づくアプローチの柔軟性は、さらなるアプリケーション・コンポーネントをオン・デマンドで付加することによって、例えば、任意の動く物体ではなく既知の人物の存在を特定するための顔検知機能によって、アプリケーションを拡張することを可能にする。

【００５４】

アプリケーション・コンポーネント配置プロセスのコンテキストにおいては、さまざまな技術的な問題がある。例えば、高度に分散型のリソース展開アルゴリズムの場合においてさえ十分なパフォーマンスおよびスケールを提供する効率的なリソース割り振り戦略および管理方式の実現、きめ細かいサービス展開の妥当な限度の特定（すなわち、アプリケーション・コンポーネントの最適な粒度の特定）、適切な割り振りアルゴリズムおよび戦略の設計、ならびに、完全に分散型の割り振りアプローチの実現可能性である。本明細書は、これらの技術的な問題に対処する。特に本明細書は、ＭＣ内にアプリケーション・コンポーネントを配置するための方法について記載している。

【００５５】

完全に分散型の割り振りアプローチに対する可能なソリューションは、単一の計算ステップにおけるアプリケーション・コンポーネントを「グローバルな」知識に基づいて計算リソースに割り振らずに、コンポーネントをリソースからリソースへと、リソースにおけるローカルな知識に基づいて、現時点で最も適合するリソースに到達するまで転送することであると言える。このアプローチは、ＩＰネットワークにおけるＩＰパケットのための転送アプローチと比較可能である。そのような転送アプローチにおいては、トポロジーおよび需要情報は、近隣、または選択された設備を中心とした小さな半径の円に限定されることが可能であり、その一方で需要情報は、コンポーネントを近隣の境界上のクライアントにマップすることによって、これらの近隣の外側の残っている（リモート）クライアントに関して暗黙のうちに取り込まれる。円の半径は、拡張性とパフォーマンスとの間のトレードオフを調整するために使用されることが可能である。

【００５６】

さらなるアプローチは、レプリカ選択のための分散型のマッピング・ノードの使用であると言える。顧客のレプリカに対する要求の重み付けされた分割を保持し、その一方でクライアント－レプリカ局所性を可能な最大限度まで維持するアルゴリズムが使用されることが可能である。そのようなアプローチは、クライアント・パフォーマンスおよびサーバ・ロードを併せて考慮する最適化問題を解決する。

【００５７】
上述のアプローチは、典型的には、現在の分散型のシステム状態を分散型のサービス要求とともに記憶しない。結果として、システム状態または要求レートのそれぞれの軽微な変化は、時間および処理労力を必要とする最適化問題を解決することに対する必要性に、ならびに／または低速で極小を含む場合があるマルチステップ・アプリケーション・コンポーネント再配置プロセスにつながる場合がある。さらに、上述のアプローチは、反復される同期化されたサービス攻撃の特定のための能力を提供しない。
【００５８】
図１は、コンピューティング・ノード（コンピューティング・デバイスとも呼ばれる）１０１の集合１００を示す。これらのコンピューティング・ノード１０１は、階層のないフラットな構成を形成し、つまり、集合１００のコンピューティング・ノード１０１のいずれも全体の制御または管理機能を持たない。コンピューティング・ノード１０１のそれぞれは、その他のコンピューティング・ノード１０１とは独立して働き、コンピューティング・ノード１０１で利用可能な集合１００の構造の個々の情報にのみ依存する。集合１００は、本明細書においてはメディア・クラウド（ＭＣ）１００と呼ばれる。さまざまなノード１０１は、インターネットなどの通信ネットワーク１０３を介して相互接続される。
【００５９】
（集中型の方法とは対照的に）分散型の方法でサービスまたはアプリケーションを提供するためにクラウド・コンピューティング機器１０１の分散型の構成１００を使用することが、提案される。サービスまたはアプリケーションの分散型のプロビジョニングの結果として、サービスまたはアプリケーションは、（とりわけ通信ネットワーク１０３の必要とされる送信リソースに関して）リソースがより効率的に利用される方法で提供され得ると期待される。この文脈で、クラウド・コンピューティング機器１０１に関する完全に分散型のリソース管理（ＲＭ）システムが記述され、それにより、クラウド・コンピューティング機器１０１で提供されるＲＭ機能のいずれも、利用可能なリソースおよび構成１００のトポロジーに関する完全な知識を持たない。全体として、ＭＣ１００のノード１０１のそれぞれの自律的な分散型の自立したリソース管理（ＲＭ）機能を提供することが望ましい。
【００６０】
この文脈で、「自律的な」ＲＭ機能とは、各ノード１０１が、実行されるアプリケーションまたはアプリケーションのコンポーネントをどこに持つべきかを決定するために、そのノード１０１のローカルのリソースの近隣のノードについて自律的に決定することを意味する。さらに、「自律的な」ＲＭ機能は、別のクラウドのリソースの領域の代表を自律的に決定する。言い換えると、ＭＣ１００は、複数のクラウドのエリア１０２に細分される可能性があり、第１のエリア１０２のノード１０１のそれぞれが、第２のエリア１０２全体（または第２のエリア１０２の下位領域）の代表である第２のエリアのノード１０１を自律的に選択することができる。したがって、各ノード１０１は、ノードのエリア１０２内のノード１０１の近隣で利用可能であるリソースのローカル・リソース・グラフを自律的に構築することができる。さらに、各ノード１０１は、ＭＣ１００のその他のエリア１０２の代表ノードのトポロジーのリストを構築し、それによって、各ノード１０１にＭＣ１００のすべてのエリア１０２（およびおそらくは下位領域のすべて）への入口の点（ｐｏｉｎｔ　ｏｆ　ｅｎｔｒｙ）を提供することができる。
【００６１】
各ノード１０１のＲＭ機能は、リソース管理機能があらゆるノード１０１に置かれるという点で「分散型」である。一実施形態においては、ノード１０１のいずれも、いかなる特定の特殊な役割（例えば、調整の役割）も持たない。各ノード１０１は、そのノード１０１のＲＭ機能を「自立」して実行し、つまり、ＭＣ１００内でソフトウェア・コンポーネントをどこに置くかの決定が、（上位レイヤの制御機能と調整することなく）当該ノードのＲＭ機能によって単独で実行される。「自立」して働くために、各ノード１０１は、

（例えば、ローカル・リソース・グラフによる）近くの／ローカルのリソースの個々の像と、（例えば、トポロジーのリストによる）その他のエリアおよび／または（下位）領域への個々のつながりとを保持する。

【００６２】

ＭＣ１００のノード１０１は、ＭＣ１００内のすべてのノード１０１の位置の共通の全体的なネットワーク・マップを共有しない。その代わりに、各ノード１０１が、ＭＣ１００全体の当該ノードの像を反映する個々のネットワーク・マップを含む。個々のネットワーク・マップは、ローカル・リソース・グラフ（それにより同じエリアまたは領域１０２内の近隣のノードの一部を示す）と、トポロジーのリスト（それによりＭＣ１００の各エリア１０２（または領域）の少なくとも１つの代表ノードを与える）とを含む可能性がある。

【００６３】

図２は、ノード１０１のトポロジーのクラスタ化３００を示す。上で示されたように、ＭＣ１００のトポロジーは、（例えば、１つまたは複数の領域３０２を含むエリア１０２の）階層のレベルに順序付けられる可能性がある。したがって、ノード１０１（例えば、図３のノードＢ）は、領域３０２（例えば、領域α）に属すると考えられる可能性があり、領域３０２自体は、エリア１０２（例えば、エリアａ）に属すると考えられる。

【００６４】

特定のノード１０１は、ＭＣ１００全体の限られた像しか持たない可能性がある。ＭＣ１００のこの限られた像は、「自律的な」ＲＭ機能を実行するためにノード１０１によって使用される。ＲＭ機能のこの部分は、ノード１０１によって実行されるトポロジー管理と呼ばれる可能性がある。ＭＣ１００の各エリア１０２（または領域３０２）に到達可能であるために、各ノード１０１は、別のエリア１０２（または領域３０２）の１つの（場合によってはいくつかの）代表をそのノード１０１のトポロジー・ツリーまたはトポロジー・リスト（トポロジーのリストとも呼ばれる）に追加する。ＭＣ１００のノードが１つの階層のレベル、例えば、エリア１０２内に（領域３０２へのいかなる細分化もなしに）編成される場合、各ルート・ノード１０１が、すべてのその他のエリア１０２の（任意の）代表を記憶すべきである。ノード１０１が２つの階層のレベル、例えば、領域３０２およびエリア１０２に編成される（各エリア１０２は１つまたは複数の領域３０２を保有する）場合、各ルート・ノード１０１が、すべてのその他のエリア１０２の（任意の）代表と、このエリア１０２内の領域３０２のいずれかの（任意の）代表とを記憶すべきである。

【００６５】

したがって、各ノード１０１は、エリア１０２内（または領域３０２内）のリソース管理を実行する能力をノード１０１に与えるそのノード１０１のローカル・リソース・グラフ（ＲＧ）のルートの位置にそのノード１０１自身を置く。さらに、各ノード１０１は、そのノード１０１のトポロジーグラフ（またはトポロジー・リスト）のルートの位置にそのノード１０１自身を置く。これは、ノード１０１に、ネットワークのそのノード１０１の個々の像を与える。各（ルート）ノード１０１は、そのノード１０１のトポロジー・グラフ（ＴＧ）にその他の領域３０２（および／またはエリア１０２）の１つまたは複数の代表を追加する。領域３０２内の任意のノードがこの領域３０２の代表になる可能性があり、つまり、すべてのノードは同等であり、エリア１０２または領域３０２のノードのいずれも特別なタスクを持たないことに留意されたい。（エリア１０２および領域３０２を含む）２階層のトポロジーの場合、ノード１０１のＴＧを使用してノード１０１のそれぞれから正しい領域３０２をアドレス指定するために、最大で２つのステップが必要とされる。

【００６６】

ローカルのおよび領域のトポロジーの情報は、図３に示されるテーブル６００に記憶され得る。テーブル６００は、ローカル・リソース・グラフ６０１のそれぞれのノードに関連するコスト６１１を含めて、ローカル・リソース・グラフ６０１のノードを示す。別の

ノードのコスト６１１は、別のノードに付与されるリソース値、例えば、利用可能な処理リソース、利用可能なリンクの帯域幅、利用可能なメモリ・リソース、実現可能な往復遅延時間などを含む可能性がある。さらに、テーブル６００は、その他の領域および／またはエリアの代表ノードを示すトポロジー・リスト６０２を提供する。トポロジー情報のエントリは、（領域／エリア毎に単一のエントリではなく）複数の選択肢を保有する可能性もある。したがって、メモリ・テーブル６００は、ＭＣ１００のノードの観点の表現である。通常、ローカル・リソース・リスト６０１内のノードの数は、エリア／領域内のノードの総数未満の所定の数に制限される。トポロジー・リスト６０２内のエリア／領域毎のノードの数は、エリア／領域内のノードの総数未満のいくつかのノード（例えば、１つまたは２つのノード）に制限される。これは、各ノード１０１が完全なＭＣ１００の制約された像しか持たないことを意味する。

【００６７】

ノード１０１は、テーブル６００のリソースおよびトポロジーを管理する。リソース・エントリ６１１は、近隣のノードから受信されたコストのタプルの情報を記憶する。ルート要素からの距離（ｄ）に依存して、コストのタプルの値の精度は、正確性、現実性、集約された像（ａｇｇｒｅｇａｔｅｄ　ｖｉｅｗ）などに関して変わる可能性がある。コストのタプルは、処理、メモリ、リンクの帯域幅、ＲＴＴ（往復遅延時間）などのリソース値を含み得る。コンポーネントのインスタンス化プロセス（すなわち、コンポーネントの配置プロセス）の場合、ノードは、まず、そのノード自身のリソースの状態を分析し、次いで、それをＲＧ６０１のノードと比較する。ノードは、コンポーネントをローカルでインスタンス化するのか、またはＲＧ６０１内の近隣のノードに要求を転送するのかを決定する。

【００６８】

ローカル・リソース・グラフ６０１は、ノード１０１によって実行されるリソース管理のために使用され得る。上で示されたように、各ノードは、ノード１０１で利用可能な限られた情報に基づいて、特に、ローカル・リソース・グラフ６０１に基づいて、独立したリソース管理機能を実行する。ローカル・リソース・グラフ６０１は、（概して同じ領域から取得される）ノードの部分集合に基づく。複数の領域３０２の間の境界の近くに位置付けられているノード１０１に関しては、ローカル・リソース・グラフ６０１の近隣のノードが、その他の領域のノードを含む可能性があることに留意されたい。概して、ローカル・リソース・グラフ（ＲＧ）の木の深さは、（近くのネットワークの近隣のノード、または近傍に）制限される。ノード１０１のブート・プロセスにおいて、ローカル・リソース・グラフ内の位置が、（領域の）ノードの所与の集合からネゴシエーションされる可能性がある。言い換えると、ノード１０１は、ローカル・リソース・グラフ内に置かれるべき利用可能なノードの適切な部分集合を選択することができる。継続的な（時間のかかる）最適化プロセスが、ノードを同じ領域内のその他のノードで置き換えることを可能にする。つまり、ルート・ノード１０１は、そのルート・ノード１０１のローカル・リソース・ノード内のノードがノード１０１のリソース管理機能に（あまり）寄与しないことを見つける場合、寄与しないノードを、ルート・ノード１０１の近隣の別のノードで置き換えることを決定する可能性がある。

【００６９】

上で示されたように、ローカルＲＧの各ノード１０１は、コストのスカラ／タプル６１１を与えられる。このタプル６１１は、新しいコンポーネントのインスタンス化要求がどこに行われる必要があるかを決定するのに役立つ。言い換えると、ＭＣ１００内でアプリケーションのコンポーネントの実行をどこに委ねるべきかを決定するとき、ノード１０１は、ローカルＲＧ６０１を調べ、ノードによって与えられるコスト６１１に基づいて、ローカルＲＧ６０１内に含まれるノードのうちの１つにコンポーネントを委ねる可能性がある。ローカルＲＧ６０１のノードは、それらのノードのＲＧのルート・ノード１０１に現在のリソースの状態を定期的に知らせる。言い換えると、ローカルＲＧ６０１のノードは、それらのノードのリソースに関する情報をルート・ノード１０１にプッシュし、それに

よって、ルート・ノード１０１が、裏付けのあるリソース管理の決定を行うことができることを保証する。特に、ローカルＲＧの情報（例えば、コスト６１１）は、（ルート・ノード自身を含む）ローカルＲＧ内のノードのうちの１つをコンポーネントの配置のための適切なノードとして特定するために使用される。コンポーネントの配置プロセスは、数回繰り返される可能性があることに留意されたい。一方、リソース管理機能を実行するためのいかなる中央機能または部分的な中央機能も存在せず、それによって、単一障害点のリスクをなくす。

【００７０】

したがって、各ノード１０１は、ローカル・リソース・グラフ６０１と、その他の領域の１つまたは複数の代表ノードを示すトポロジーのグラフ６０２とを含む限られたネットワークの像を有する。上で示されたように、ＭＣ１００のトポロジーは、ＶｉｖａｌｄｉおよびＭｅｒｉｄｉａｎアルゴリズムを用いて分散するようにして決定され得る。初期化段階では、各ノード１０１が、同じ領域／エリア内のノードの完全なリストにアクセスできる可能性がある。ノード１０１は、このノードのリストを使用してローカル・リソース・グラフ６０１を構築する。さらに、初期化段階では、各ノード１０１が、残りの領域／エリアから少なくとも１つのノードを選択する可能性がある。残りの領域／エリアの少なくとも１つのノードの選択は、領域のノードがその他の領域の異なる代表ノードを有することを保証し、それによって、単一障害点または欠陥を防止するために、ランダムに実行されるべきである。

【００７１】

以下で、ノード１０１によって提供されるマッピング機能４０１に関するさらなる詳細が、説明される。ノードで利用可能なトポロジーおよびリソース情報（すなわち、情報６００）に基づいて、ＭＣ１００のノードは、メディア・クラウド・システム１００のノード１０１上のソフトウェア（メディア）コンポーネントの（最適な）配置に関する決定を行うことができる。図４に示されるように、アプリケーション７００は、概して、複数のコンポーネント７０３からなる。例として、電話会議アプリケーションは、複数のオーディオ・コーデック（符号化／復号）コンポーネント（電話会議の各参加者につき１つずつ）と、（参加者の音声チャネルを接続するための）ミキサー・コンポーネントとを含む。通常、アプリケーション７００（およびコンポーネント７０３）は、ソース７０１（データの提供元）およびシンク７０２（データの提供先）を有する。上述の例においては、電話会議アプリケーションの個々の参加者が、ソースおよびシンクであると考えられ得る。コンポーネントの配置プロセスのタスクは、通信ネットワーク１０３のリソースの消費を削減するために、アプリケーション７００のコンポーネント７０３をＭＣ１００内の適切な場所に置くことである。例として、電話会議アプリケーションのオーディオ・コーデック・コンポーネントをそれぞれの参加者の近くに配置することによって、（符号化された音声トラフィックのみが通信ネットワーク１０３を通じて送信されるので）アプリケーションにより必要とされる送信帯域幅が削減され得る。さらに、電話会議アプリケーションのミキサー・コンポーネントは、電話会議の参加者の間の中心の場所に置かれるべきである。

【００７２】

図４において、異なるコンポーネント７０３の間およびソース７０１とシンク７０２との間のリンクの異なる幅は、リンクに関する異なる要件を示す（ラバー・バンド・モデル（ｒｕｂｂｅｒ　ｂａｎｄ　ｍｏｄｅｌ））。より大きなばね乗数を示すコンポーネント７０３の間のバンドは、コンポーネント７０３が互いに近くに（例えば、同じノードに）置かれるべきであることを示す。

【００７３】

配置手順は、利用可能なノードのリソースおよび利用可能なリンクのリソースを考慮に入れるべきである。さらに、（例えば、プロセッサ・リソース、メモリ・リソース、リンクの帯域幅、遅延、ジッタに関する）アプリケーション・コンポーネント７０３の要件が、考慮に入れられるべきである。

【００７４】
そのような配置の決定は、集中型の方法で実行され得る。しかし、概して、コンポーネントの配置の集中式またはメッシュ式の解決策は、大規模なシステムに見合わない。さらに、そのような集中式の解決策は、単一障害点をもたらす傾向がある。
【００７５】
本明細書において、メディア・クラウド１００のノード１０１で利用可能な限られた情報を用いる分散型の配置方式が説明される。分散型の配置方式は、個々のノード１０１によって実行される個々のマッピング機能４０１を利用する。これらのマッピング機能は、２つの下位タスク、すなわち、トポロジー管理とリソース管理とに分けられる。トポロジー管理は、ノード１０１のそれぞれで利用可能なトポロジー情報（特に、トポロジー・リスト６０２）を利用する。コンポーネント配置要求は、通常、アプリケーション（またはコンポーネント）のシンクまたはソースについての領域の情報を伴う。ノードは、このトポロジー情報を調べ、トポロジー情報がそのノード自身のトポロジー情報に合致しない場合、要求を領域（またはエリア）の代表に転送する。言い換えると、ノードは、コンポーネントの所望のシンクまたはソースの場所がノードの領域と一致するかどうかを検証する。一致しない場合、コンポーネント配置要求は、（トポロジー・リスト６０２から）ノードに知られている適切なエリアまたは領域の代表ノードに渡される。２階層のトポロジー（領域およびエリア）においては、正しい領域３０２をアドレス指定するために、最大で２つのステップが必要とされる。複数のコンポーネントに関する配置要求の場合、トポロジー・プロセスは、１回実行されれば十分である。言い換えると、（同じサービスまたはアプリケーションに属する）一連の関連するコンポーネントは、単一のステップで配置され得る。
【００７６】
リソース管理は、ＭＣ１００の異なるノードの負荷の状態に応じたローカルのリソース配置を対象とする。ノードは、コンポーネント配置要求を受信し、コンポーネントがそのノードの領域内に置かれるべきであると判定する場合、そのノードのローカル・リソース・グラフ６０１を調べて、コンポーネントを実行するための必要なリソースを有するグラフ６０１内のノードを特定する。概して、ネットワークの異なるノード１０１は、配置されるべきコンポーネントのコピーを既にキャッシュしてある。したがって、通常、特定されたノードでコンポーネントのインスタンス化を開始するだけで十分である。そうでなければ、特定されたノードは、中央コンポーネント・データベースからコンポーネントをダウンロードすることができる。
【００７７】
一例においては、図２のノード３１１（ソース）が、ノード３１２（シンク）を含むシンクを有するアプリケーションの設定を要求する。問題は、ノード３１１がシンクの近くにあるノードをどのようにして発見することができるか（またはその逆）である。上で示されたように、ＭＣ１００の各ノード１０１のマッピング機能（ＭＦ）４０１は、そのノード自身および近隣のノードのリソースの占有およびトポロジー情報を記憶し、したがって、各ノードは、自立してそのノードの配置の決定を行うことができる。第１に、利用可能なトポロジーの情報が、シンクかソースかのどちらかに近いネットワークの領域を発見するために使用される。第２に、選択された領域内の近隣のノードのローカルのリソース情報が使用され、その結果、ノードは、そのノードの近隣のどこに新しいコンポーネントを置くべきかを決定することができる。上述の配置方式を使用すると、ノードのいずれも、ＭＣ１００の完全で厳密なリソースおよびトポロジー情報を知る必要がない。それでも、実現される配置は、ほぼ完全である。本明細書において概要を説明される配置方式においては、ＭＣのノード１０１のいずれも、オンライン処理中に特別な役割を持たないことに留意されたい。結果として、任意のＭＣのノード１０１のうちの１つまたはいくつかが、システムの停止を引き起こさずに故障する可能性がある。
【００７８】
ノードのマッピング決定プロセスは、以下のステップを含み得る。第１のステップにお

いて、（コンポーネント配置要求で要求された）シンクがそのノードと同じエリア／領域内にあるかどうかが調べられる可能性がある。そうでない場合、ノードは、要求されたシンクに合致するに違いないエリア／領域内の代表ノードに関してそのノードのテーブル６０２を検索する。ノードは、代表ノードにコンポーネント配置要求を転送する。代表ノードは、シンクがその代表ノードのエリアおよび領域内にあると確認し、そうでなければ、送り先の領域からその個々の代表に要求を転送する必要がある。代表ノードは、近くにあり、コンポーネントを実行するために最良のコスト値を有する最もふさわしいＭＣのノードに関してその代表ノードのローカルＲＧ６０１を調べる。概して、アプリケーションは複数のコンポーネントからなるので、したがって、これらのコンポーネント間の相互接続はさまざまな要件を有することに留意されたい。配置の決定は、アプリケーションのグラフ情報の全体またはより大きな部分がより全体的なマッピングの決定のために提供され得る場合、改善される可能性がある。

【００７９】

上で示されたように、各ノード１０１は、ローカル・リソース・グラフ６０１を含む。コンポーネント配置要求を受信するとき、ノード１０１は、そのノード１０１のローカル・リソース・グラフ６０１内で適切なノードを探索し、このノードにコンポーネント配置要求を転送する。このノードも、ローカル・リソース・グラフ６０１を含み、コンポーネント配置要求を処理するためのそのノードのローカル・リソース・グラフ６０１内の適切なノードを探索する。この繰り返しのプロセスの収束を保証するために、コンポーネント配置要求の転送は、最低限の必要とされる改善に関する条件に従う可能性がある。特に、転送がコンポーネントの配置に関する最低限の必要とされる改善につながる場合（例えば、プロセッサの能力／帯域幅の２０％の削減など）にのみ、コンポーネント配置要求がローカル・リソース・グラフ６０１内のノードに転送され得ると規定される可能性がある。

【００８０】

したがって、ＭＣ１００の複数のノード１０１は、集中化された配置サーバの関与を伴わずに、ネットワーク内でアプリケーション・コンポーネントの分散型の配置を実行することを可能にされる。複数のノード１０１は、コンポーネント配置メッセージ（例えば、コンポーネント配置要求、および／または、１つもしくは複数の近隣のコンピューティング・デバイスにおける利用可能なコンピューティング・リソースについての情報に関するメッセージ）のやり取りに依存する。以降では、機械自己学習に基づいて、分散型の配置方式を加速するための方法が説明される。

【００８１】

上で概要を説明されているように、（分散型の配置方式の一部としての）展開アルゴリズムは、分散型のメディア／クラウド・コンピューティング・システム１００のノード１０１上での仮想マシン（ＶＭ）、アプリケーション、またはアプリケーション・コンポーネントのロケーションを計算し、それによって、サービス・ソフトウェア（ＶＭ、アプリケーション、コンポーネント）の配置は、リソース使用（ＣＰＵ、メモリ、帯域幅．．．）、およびユーザによって経験されるサービス品質に関して最適化される。例として、ビデオ会議アプリケーションが考えられる。会議の参加者のほとんどが米国に配置されている場合、ミキサー・アプリケーション・コンポーネント７０３は、好ましくは米国に配置されるべきである。会議の均等に分散された（ロケーション）参加者の別の場合においては、共通のミキサー・アプリケーション・コンポーネント７０３のロケーションは、ネットワーク・リソース割り当てが最小になるように決定されるべきである。

【００８２】

全体としては、多数のアプリケーション（および対応するアプリケーション・コンポーネント）が、ＭＣ１００内に配置される。分散型の配置プロセスの結果は、その後のアプリケーション配置要求の配置のために使用されることが可能である配置知識を抽出するために使用されることが可能である。特に、分類子が、特定のサービス（アプリケーション）がアクセスされた元であるメディア・クラウド１００の入口ポートを特定することによって、さまざまなサービス・アクセス状況（会議アプリケーション、放送アプリケーショ

ンなど）の間で区別を行うことができる。（分散型の）配置アルゴリズムが特定のサービス・アクセス状況に関して（ほとんど）最適な配置を見つけた場合、配置決定は、具体的なサービス・アクセス状況とともに記憶されることが可能である。その結果として、次のときに、格納されているアクセス状況と同様のアクセス状況を分類子が特定した場合、記憶されている配置ロケーションが使用およびセットアップされることが可能である。したがって、前の配置決定に依存することによって、その後の配置決定が加速されることが可能である。

【００８３】

典型的には、ＭＣ１００上で並行して使用される大量の多様なサービス（アプリケーション）７００がある。これらのサービス７００のそれぞれは、さまざまな入口ポートを介してサービス７００およびＭＣ１００に到達するさまざまな量のクライアントによって使用されることが可能である。典型的には、アクセス状況における軽微な変化（例えば、（ｎ－１）個の会議クライアントが米国に配置されていて、１個の会議クライアントがドイツから配置されているシナリオと比較した、ｎ個の会議クライアントが米国に配置されているシナリオ）は、ソフトウェア・コンポーネント７０３の最適な配置は変わらないという結果をもたらす。機械学習に基づく分類は、同じ配置決定を使用してどのアクセス状況が最適にサービス提供されることが可能であるかを特定するために適用されることが可能である。機械学習に基づく分類はまた、同じ配置決定を使用してサービス提供されることが可能である類似したアクセス状況どうしのいわゆる（ε）－ルームを定義する（特定のアクセス状況にまつわる）ローカル・ハイパースフィアの定義／学習を可能にする。ローカル・ハイパースフィアは、それまでにわかっている最適なコンポーネント／ソフトウェア配置を格納しているデータベース・エントリとともに割り振られることが可能である。最適なコンポーネント配置を含むデータベース・エントリを伴うハイパースフィアの割り振りは、オンライン学習手順の一部である。ＭＣリソース管理がそのデータベース・エントリに従って配置をセットアップした後に、（上で概要を説明されているように）分散型のマッピング・アルゴリズムを適用することによって、コンポーネント配置のさらなる改善が達成されることが可能である。改善された配置状況が決定されることが可能である場合、提案されたコンポーネント配置を含むデータベース・エントリが、その改善された配置で上書きされることになり、それによって、連続したオンライン学習プロセスが提供される。

【００８４】

オンライン学習プロセスは、（アクセス状況を分類するために使用される）分類子が、１つのローカル・ハイパースフィアをいくつかの異なるローカル・ハイパースフィアへと分割するという結果をもたらすことがある。これは、単一のハイパースフィアによって分類されたアクセス状況に関する一意の（前の）最良の配置ソリューションが、個々のハイパースフィアを伴う別々の配置ソリューションへと分離する場合に、生じることがある。

【００８５】

分類は、サービス・タイプ毎に別々に、またはメディア／クラウド・システム１００上で稼働しているすべてのサービスに関して単一の分類子を用いて、実行されることが可能である。サービス（またはアプリケーション）タイプは、アクセス状況のタイプ（例えば、（ビデオ会議の場合のように）多対多、（放送の場合のように）一対多、（ポイントツーポイント通信の場合のように）ポイントツーポイントに基づいて定義されることが可能である。代替として、または追加として、サービス・タイプは、領域の考慮事項（大陸、国、都市）に基づいて定義されることが可能である。代替として、または追加として、サービス・タイプは、提供される実際のサービス（例えば、会議、放送など）に基づいて定義されることが可能である。すべてのサービスに関する単一の分類子の場合、競合するサービスどうしの間の対話が考慮に入れられることも可能である。結果として、ローカル・ハイパースフィアの数が著しく増えることがある。

【００８６】

サービス・アクセス状況に加えて、（サーバおよびリンク）のリソース割り当て状況が

、分類子へのさらなる入力として使用されることが可能である。例として、機械学習（ＭＬ）分類子は、下記のことを考慮に入れることができる。
・別々のクライアントからのサービス・アクセス状況
・提供される（クラウド）サービスのタイプおよびそれらのアクセス状況
・サービスがアクセスされる入口および出口クラウド・ポート
・ノード１０１およびネットワーク・リソースの割り当て、それによって、リソース毎に別々の特性評価タプルが予想されることが可能である；
【００８７】
学習フェーズ中に、ＭＬ分類子は、等しいまたは同様の配置状況を含む複数のハイパースフィアを構築する。さらに、ＭＬ分類子は、複数のハイパースフィアに関して個々のリソース展開決定を決定する。
【００８８】
したがって、最適で効率的なアプリケーション・コンポーネント配置決定を可能にするために、機械自己学習システムが使用されることが可能である。ＭＬシステムは、２つの異なるフェーズ、すなわち、学習フェーズおよび実行フェーズにおいて機能する。ＭＬシステムは、ＭＣ１００の個々のノード１０１上で実行されることが可能である。
【００８９】
学習フェーズ中に、（例えば、特定のノード１０１上で稼働している）ＭＬシステムは、特定の配置状況を記述する特徴ベクトルを導出するために、メディア・クラウド制御トラフィック・メッセージ（例えば、配置要求、およびリソースに関するメッセージ、ならびに配置決定に関するメッセージ）を分析する。特徴ベクトルは、典型的には、アプリケーション・グラフのタイプ（アプリケーション・コンポーネントの星形グラフ、アプリケーション・コンポーネントの線形グラフ）、アプリケーション・コンポーネントの地理的分散、アプリケーション・コンポーネントどうしの間の接続の属性（例えば、帯域幅、待ち時間等）など、配置状況を記述するさまざまな次元を有する。特徴ベクトルは、特徴ベクトルのさまざまな次元によって形成される特徴空間内に組み込まれることが可能である。特定のアプリケーション配置要求の特徴ベクトルは、特定のコンポーネント・マッピングおよび対応する配置詳細とともに特徴ベクトル・データベース内に格納される。学習フェーズ中に、このシステムは、コンポーネント配置の分類にとって重要である特徴ベクトルの次元およびプロパティーと、すたれていて、したがって省略されることが可能である特徴ベクトルの次元およびプロパティーとの間で区別を行うことを学習する。特徴空間次元のこの削減は、学習プロセスの効率およびメディア・クラウド分類結果の質を高める。
【００９０】
学習フェーズの結果として、クラスタ（例えば、コンテンツ分散ネットワーク（ＣＤＮ）のような、ビデオ会議のようななど）へのメディア・クラウド・アプリケーション・アクセス状況の分類を反映するジオメトリー上の組み込みが得られる。言い換えれば、学習フェーズは、典型的なアプリケーション・アクセス状況に関する複数のクラスタ（それらの特徴ベクトルを介して特定される）を、配置決定とともに提供する。
【００９１】
実行フェーズ中には、学習フェーズ中に既知のメディア・クラウド・アプリケーションに関して得られたマッピング結果が、新しいおよび／または未知のメディア・クラウド・アプリケーション配置要求に関するアプリケーション・コンポーネント・マッピングを提案するために活用される。
【００９２】
この目的のために、新しいアプリケーション配置要求に関する特徴ベクトルが、学習フェーズに関して使用されたのと同じ方式を採用するアプリケーション配置要求から抽出される。特に、アプリケーション配置要求の特徴ベクトルを決定するために、制御メッセージが分析されることが可能である。次いで、新しいアプリケーション配置要求に関して抽出された特徴ベクトルが、メディア・クラウド特徴空間内に組み込まれる。
【００９３】

次いで、ジオメトリー上の考慮によって、特徴空間における以前に学習されたメディア・クラウド配置の最も近い表示、および対応するアプリケーション分類が得られる。結果として、この特徴ベクトルに関する対応する配置詳細を、機械学習フェーズ中にアプリケーション・マッピング情報で満たされているトレーニングされたＭＣ配置データベースから呼び出すことが可能である。

【００９４】

結果として、高速かつ効率的な様式で機械学習プロセスから前のメディア・クラウド・アプリケーション・マッピング経験に基づいてメディア・クラウド・マッピングに関する再配置提案が得られる。その結果として、アプリケーション配置要求は、高速かつ効率的な様式で適切なＭＣノード１０１に導かれることが可能である。

【００９５】

ジオメトリー上の考慮によってアプリケーション配置状況を記述する特徴ベクトルを特定することは、例えば、特徴空間における適切なメトリックを定義することによって、およびそのメトリックの考慮のもとで別々の特徴ベクトルの間の距離を計算することによって、可能である。このメトリックは、別々のベクトル次元に関して別々の重み付けを利用することができ、それによって、学習および実行フェーズ中に別々の次元を強調することができる。

【００９６】

メディア・クラウド特徴ベクトルは、アプリケーション、およびメディア・クラウド・インフラストラクチャに対するそのマッピングを特徴付ける下記の例示的な情報を含むことができる。

・アプリケーション・グラフ（例えば、星状グラフ、線形グラフ）内の関係、
・リソースへのアプリケーション・コンポーネントの割り当て、
・アプリケーション・コンポーネントのリソース消費、
・アプリケーション・コンポーネントのジオロケーション、
・クライアント（シンク７０２、ソース７０１）の入口／出口ポイント（デバイス）、および／または
・接続属性（帯域幅、待ち時間、損失率、．．．）。

【００９７】

図５は、可能なアプリケーション配置状況の例示的なベクトル空間２００を示している。特定のアプリケーション配置状況は、典型的には、特定のアプリケーション配置要求にリンクされる。例示的なベクトル空間２００は、３つのベクトル次元２０１ｑ、ｖ、ｗを含み、それぞれのベクトル次元は、可能なアプリケーション配置状況の対応する属性を表す。上で示されているように、例示的な属性（またはベクトル次元）は、アプリケーション・グラフ（すなわち、アプリケーション・コンポーネント７０３の相対的な構成、例えば星状構成で、または線形構成で）、さまざまなアプリケーション・コンポーネント７０１のリソース消費、アプリケーション７００に関するシンク７０２／ソース７０１のロケーションなどに関連することができる。したがって、ベクトル空間２００におけるポイント（または特徴ベクトル）２０３は、（特徴ベクトルの別々の次元の値を介して）特定のアプリケーション配置状況を記述する。

【００９８】

ベクトル空間２００においてアプリケーション配置状況を表すために、属性（またはベクトル次元）２０１の値は、ベクトル空間２００において表されることが可能である数値に関連付けられることが可能である。例として、アプリケーション７００に関する別々のシンク７０２／ソース７０１のジオロケーションは、２つのベクトル次元２０１（一方は緯度角度を表し、もう一方は経度角度を表す）に対応することができる。典型的なベクトル空間２００は、数百の次元２０１を有することができるということに留意されたい。

【００９９】

学習フェーズ中に、さまざまな配置要求が、分散された様式でＭＣ１００によって処理される。配置要求に関与するノード１０１は、特定の配置要求に関するノード１０１どう

しの間でやり取りされるメッセージを観察することができる。結果として、ノード１０１は、それらのメッセージから配置要求に関する情報を抽出し、それによって、特定のアプリケーション配置要求を記述する特徴ベクトル２０３への投入を行うことができる。さらに、ノード１０１は、やり取りされたメッセージから、最終的な配置詳細２０５（すなわち、ＭＣ１００内のどこにアプリケーション・コンポーネントが配置されているかという情報）を学習する。特徴ベクトル２０３および対応する配置詳細２０５は、ノード１０１のデータベース２０４内に格納されることが可能である。

【０１００】

結果として、ノード１０１は、機械学習のためのトレーニング・データとして使用されることが可能である複数の特徴ベクトル２０３および対応する配置詳細２０５を収集する。特に、特徴ベクトル２０３の複数のクラスタ２０２を決定するために分類子が使用されることが可能であり、クラスタ２０２内の特徴ベクトル２０３は、特徴空間２００において相対的に短い距離を有する。言い換えれば、クラスタ２０２の配置状況どうしは、互いに類似している。示されている例においては、分類子は、ＥＵからの参加者を伴うビデオ会議（ＶＣ）アプリケーション要求に関する別のクラスタ２０２、世界中に分散された参加者を伴うＶＣアプリケーション要求に関する別のクラスタ２０２、２人の参加者を伴うＶＣアプリケーション要求に関する別のクラスタ２０２、および放送されるＴＶストリームの視点レンダリングなどのコンテンツ分散ネットワーク（ＣＤＮ）アプリケーション要求に関する別のクラスタ２０２を決定している。分類子は、それぞれのクラスタ２０２に関して平均および／または代表の配置詳細２０５を決定するように構成されている。これらの平均および／または代表の配置詳細２０５は、新しい配置要求の配置を加速するためにノード１０１によって使用されることが可能である。特に、ノード１０１は、新しい配置要求の特徴ベクトル２０３を決定することができる。さらに、ノード１０１は、特徴ベクトル２０３がクラスタ２０２内に収まる（またはクラスタ２０２に近い）かどうかを判定し、次いで、配置要求を処理するためにクラスタ２０２の平均および／または代表の配置詳細２０５を使用することができる。

【０１０１】

図６は、例示的な学習方法４００のフローチャートを示している。ノード１０１は、複数のアプリケーション配置要求の特徴ベクトルへの投入を行うためにＭＣ制御トラフィックをモニタする（ステップ４０１）。さらに、アプリケーション配置要求に関するマッピング（すなわち配置詳細）が、制御メッセージから抽出され（ステップ４０２）、データベース内に格納される（ステップ４０４）。抽出された特徴ベクトルおよび対応するマッピングが、学習モデルのトレーニング・データとして使用される（ステップ４０３）。例示的な学習モデルは、サポート・ベクター・マシン（ＳＶＭ）方式を利用する。この学習モデルは、複数のアプリケーション配置要求をクラスタ化するための分類子を適用することができる。そのようなクラスタ化の結果として、特徴ベクトル２０３の特定の次元が配置目的のために重要である場合があり、その一方で、特徴ベクトル２０３のその他の次元は配置決定にまったくまたはほとんど影響を及ぼさない場合があるということが観察されることが可能である（ステップ４０６）。後者の次元は、特徴ベクトル２０３から除去されることが可能であり、それによって、機械学習方式のおよび配置プロセスの計算上の複雑さが低減される（ステップ４０６）。最後に、クラスタ２０２の代表の特徴ベクトル２０３が特徴空間２００内に組み込まれることが可能である（ステップ４０７）。代表の特徴ベクトル２０３は、特徴空間２００内のハイパースフィアを定義することができ、そのハイパースフィアは、代表の特徴ベクトル２０３によって記述されている配置状況に類似した配置状況を含み、それらの配置状況は、代表の特徴ベクトル２０３に関して格納されている配置詳細２０５に従って処理されることが可能である。

【０１０２】

図７は、ノード１０１のデータベース２０４内に格納されている配置情報を使用する例示的な配置方法５００のフローチャートを示している（ステップ５０１）。ノード１０１は、新しい配置要求に関連した制御メッセージを観察し、特徴ベクトル２０３への投入を

行うために、抽出された情報を使用し（ステップ５０２）、特徴ベクトル２０３は、特徴空間２００内に組み込まれる（ステップ５０３）。その後に、最も近い代表の特徴ベクトル２０３が、データベース２０４から決定され（ステップ５０４）、それによって、新しい配置要求に適している可能性がある配置詳細２０５が決定される（ステップ５０５）。新しいアプリケーション要求は、ノード１０１のデータベース２０４から得られた配置詳細２０５に従ってノード１０１によって処理されることが可能である。その後に、その配置は、本明細書に記載の分散型の配置方式を使用して改善されることが可能である（ステップ５０６）。更新された配置詳細および元の特徴ベクトル２０３は、学習方法４００におけるさらなるトレーニング・データとして使用されることが可能であるということに留意されたい。

【０１０３】

本明細書において、クラウド・コンピューティングのためのネットワークおよび対応するコンピューティング・デバイス（ノード）のアーキテクチャが説明された。説明されたアーキテクチャは、アプリケーションのコンポーネントがコンピューティングクラウド内の適切なノードに置かれることを可能にする非集中型のアプリケーション・コンポーネント配置方式の実装を可能にする。さらに、アプリケーション・コンポーネント配置方式をより効果的にするために、機械学習技術が使用される。なぜなら、同様の状況における過去の配置から得られた知識は、新しいアプリケーションのための最適なマッピングを迅速に見つけ出すために使用されることが可能であるためである。説明されたアーキテクチャおよび方法は、増大する需要に対してスケーラブルであり、単一障害点を示さない。さらに、説明されたアーキテクチャは、クラウド・コンピューティングを使用するときに通信ネットワーク内で必要とされる帯域幅のリソースの削減を可能にする。

【０１０４】

説明および図面は、提案された方法およびシステムの原理を例示するに過ぎないことに留意されたい。したがって、当業者が、本明細書において明示的に説明されていないか、または示されていないが、本発明の原理を具現化し、本発明の精神および範囲内に含まれるさまざまな構成を案出することができることは、理解されるであろう。さらに、本明細書に挙げられたすべての例は、提案された方法およびシステムの原理、ならびに当技術分野の発展のために本発明者らがもたらした概念を読者が理解するのを助ける教示のみを特に目的とするようにもっぱら意図されており、そのような特に挙げられた例および条件に限定されないと解釈されるべきである。さらに、本発明の原理、態様、および実施形態、ならびにそれらの特定の例を説明する本明細書のすべての記述は、それらの均等物を包含するように意図される。

【０１０５】

さらに、さまざまな上述の方法のステップおよび説明されたシステムのコンポーネントはプログラムされたコンピュータによって実行され得ることに留意されたい。本明細書において、一部の実施形態は、機械またはコンピュータが読むことができ、前記上述の方法のステップの一部またはすべてを実行する命令の機械が実行できるまたはコンピュータが実行できるプログラムを符号化するプログラム・ストレージ・デバイス、例えば、デジタル・データ・ストレージ媒体を包含するようにやはり意図される。プログラム・ストレージ・デバイスは、例えば、デジタル・メモリ、磁気ディスクおよび磁気テープなどの磁気ストレージ媒体、ハード・ドライブ、または光学式に読み取り可能なデジタル・データ・ストレージ媒体である可能性がある。実施形態は、上述の方法の前記ステップを実行するようにプログラミングされたコンピュータを包含するようにやはり意図される。

【０１０６】

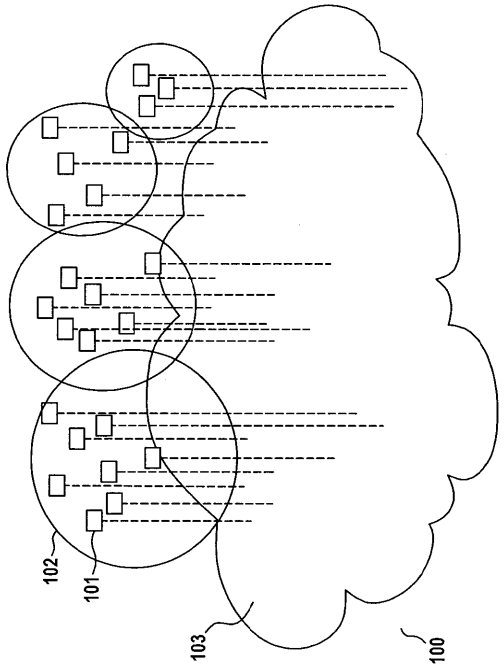
加えて、本特許明細書に記載のさまざまな要素の機能は、専用のハードウェアおよび適切なソフトウェアに関連してソフトウェアを実行することができるハードウェアを使用して提供され得ることに留意されたい。プロセッサによって提供されるとき、機能は、単一の専用のプロセッサ、単一の共有のプロセッサ、または一部が共有される可能性がある複数の個々のプロセッサによって提供される可能性がある。さらに、用語「プロセッサ」ま

たは「コントローラ」の明示的な使用は、ソフトウェアを実行することができるハードウェアを排他的に指すと考えられるべきでなく、暗黙的に、限定することなく、デジタル信号プロセッサ（ＤＳＰ）ハードウェア、ネットワーク・プロセッサ、特定用途向け集積回路（ＡＳＩＣ）、フィールド・プログラマブル・ゲートアレイ（ＦＰＧＡ）、ソフトウェアを記憶するための読み出し専用メモリ（ＲＯＭ）、ランダム・アクセス・メモリ（ＲＡＭ）、および不揮発性ストレージを含む可能性がある。通常のおよび／またはカスタムのその他のハードウェアも、含まれる可能性がある。

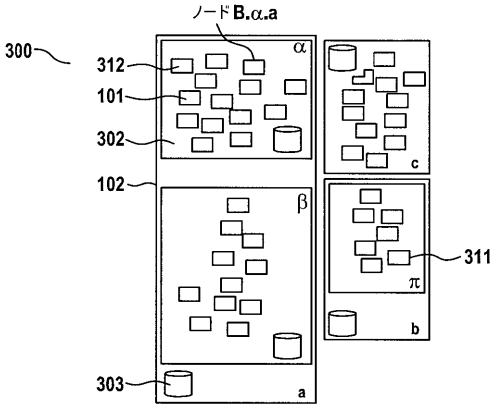
【０１０７】

最終的に、本明細書の任意のブロック図は、本発明の原理を具現化する例示的な回路の概念図を示すことに留意されたい。同様に、任意のフローチャート、流れ図、状態遷移図、擬似コードなどは、実質的にコンピュータ可読媒体で表現され、したがって、コンピュータまたはプロセッサによって、そのようなコンピュータまたはプロセッサが明示的に示されているか否かにかかわらず実行され得るさまざまなプロセスを表すことが、理解されるであろう。

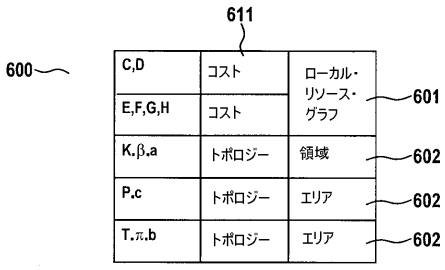
【図１】



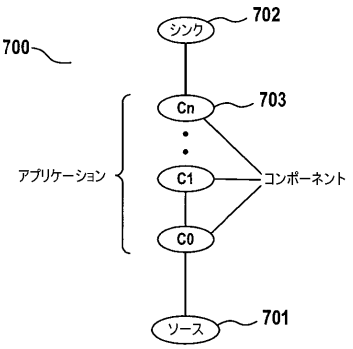
【図２】



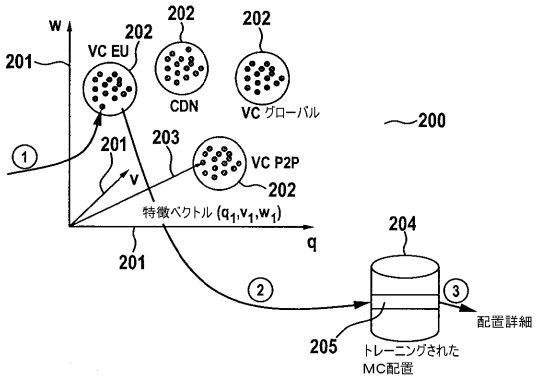
【図３】



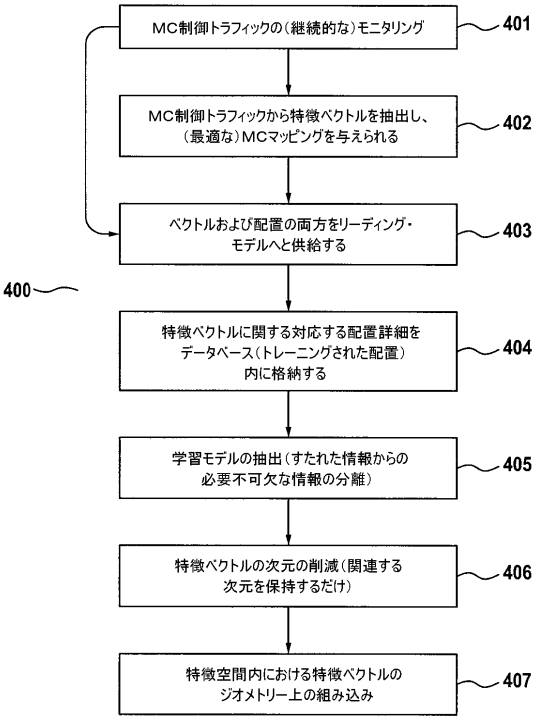
【図４】


700
シンク
702
Cn
703
C1
C0
コンポーネント
アプリケーション
ソース
701

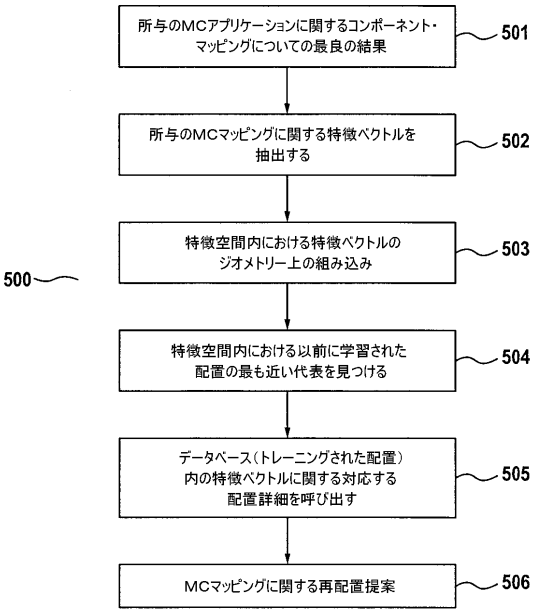
【図５】


w
201
VC EU
202
CDN
202
VC グローバル
202
200
1
201
203
VC P2P
202
v
特徴ベクトル (q_1,v_1,w_1)
q
201
204
2
3
配置詳細
205
トレーニングされた
MC配置

【図６】


MC制御トラフィックの（継続的な）モニタリング
401
MC制御トラフィックから特徴ベクトルを抽出し、
（最適な）MCマッピングを与えられる
402
ベクトルおよび配置の両方をリーディング・
モデルへと供給する
403
400
特徴ベクトルに関する対応する配置詳細を
データベース（トレーニングされた配置）
内に格納する
404
学習モデルの抽出（すたれた情報からの
必要不可欠な情報の分離）
405
特徴ベクトルの次元の削減（関連する
次元を保持するだけ）
406
特徴空間内における特徴ベクトルの
ジオメトリー上の組み込み
407

【図７】


所与のMCアプリケーションに関するコンポーネント・
マッピングについての最良の結果
501
所与のMCマッピングに関する特徴ベクトルを
抽出する
502
特徴空間内における特徴ベクトルの
ジオメトリー上の組み込み
503
500
特徴空間内における以前に学習された
配置の最も近い代表を見つける
504
データベース（トレーニングされた配置）
内の特徴ベクトルに関する対応する
配置詳細を呼び出す
505
MCマッピングに関する再配置提案
506

フロントページの続き

(72)発明者　ウォール，シュテファン
ドイツ　７１７０１　シュヴィーバーディンゲン，シュテッティナー　シュトラーセ　８

審査官　井上　宏一

(56)参考文献　特開２００６－２６８８４８（ＪＰ，Ａ）
特開２０１０－１６５２６５（ＪＰ，Ａ）
特開２００９－１３４５９０（ＪＰ，Ａ）
特開２００６－　４８２５３（ＪＰ，Ａ）
特開２００７－１９３８０６（ＪＰ，Ａ）
特開２００５－２３４９３１（ＪＰ，Ａ）
特開平　９－２５２２７５（ＪＰ，Ａ）
特開２００９－２６５８７６（ＪＰ，Ａ）
米国特許出願公開第２００２／００１９８６９（ＵＳ，Ａ１）

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