(19)日本国特許庁(JP)　(12)特　許　公　報(B2)　(11)特許番号

特許第6254948号
(P6254948)

(45)発行日　平成29年12月27日(2017.12.27)　(24)登録日　平成29年12月8日(2017.12.8)

(51)Int.Cl.		F I		
G06F 9/48	(2006.01)	G06F	9/46	452A
G06F 9/46	(2006.01)	G06F	9/46	350
G06F 11/20	(2006.01)	G06F	11/20	697

請求項の数 16　(全 38 頁)

(21)出願番号	特願2014-552219(P2014-552219)
(86)(22)出願日	平成25年1月7日(2013.1.7)
(65)公表番号	特表2015-503811(P2015-503811A)
(43)公表日	平成27年2月2日(2015.2.2)
(86)国際出願番号	PCT/US2013/020442
(87)国際公開番号	WO2013/106256
(87)国際公開日	平成25年7月18日(2013.7.18)
審査請求日	平成27年12月28日(2015.12.28)
(31)優先権主張番号	13/346,303
(32)優先日	平成24年1月9日(2012.1.9)
(33)優先権主張国	米国(US)

(73)特許権者　314015767
マイクロソフト　テクノロジー　ライセンシング，エルエルシー
アメリカ合衆国　ワシントン州　９８０５２　レッドモンド　ワン　マイクロソフト　ウェイ
(74)代理人　100140109
弁理士　小野　新次郎
(74)代理人　100118902
弁理士　山本　修
(74)代理人　100106208
弁理士　宮前　徹
(74)代理人　100120112
弁理士　中西　基晴

最終頁に続く

(54)【発明の名称】分散コンピューティング環境内で仮想マシンのプールにジョブを割り当て仮想マシン上でタスクを実行するための方法、プログラム、プログラムを格納した記憶媒体、及びシステム

(57)【特許請求の範囲】

【請求項１】

コンピューティング装置が、分散コンピューティング環境内で仮想マシンのプールにジョブを割り当て仮想マシン上でタスクを実行するための方法であって、

前記コンピューティング装置が、ユーザから、ジョブを表すワークアイテムを受信するステップと、

前記コンピューティング装置が、前記ワークアイテムから前記ジョブを作成するステップと、

前記コンピューティング装置が、少なくとも第１の仮想マシンを第１のプールに自動的に割り振るステップと、

前記コンピューティング装置が、前記第１のプールに前記ジョブを割り当てるステップと、

前記コンピューティング装置が、前記第１のプールに関連付けられた第１のスケジューラを用いて、ジョブキューから前記ジョブのタスクを取り出し、前記タスクを前記第１のプール内の前記第１の仮想マシンに割当てる、ステップと、

前記コンピューティング装置が、前記第１のスケジューラを用いて、前記第１のプール上で前記タスクをスケジューリングするステップと、

前記コンピューティング装置が、前記第１の仮想マシン上で前記タスクを実行するステップと、

前記コンピューティング装置が、前記第１のプールからの前記ジョブの移行を決定する

ステップと、
前記コンピューティング装置が、前記移行の決定に基づき、前記第１の仮想マシンへの前記タスクの割当てを解放するとともに、少なくとも第２の仮想マシンからなる第２のプールに関連付けられた第２のスケジューラを用いて、前記ジョブキューから前記ジョブのタスクを取り出し、前記タスクを前記第２のプール内の前記第２の仮想マシンに再割当てするステップと、
前記コンピューティング装置が、前記第２のスケジューラを用いて、前記第２のプール上で前記タスクをスケジューリングするステップと、
前記コンピューティング装置が、前記第２の仮想マシン上で前記タスクを開始するステップと
を含む方法。
【請求項２】
前記第１のプール上で前記タスクを前記スケジューリングするステップが、前記第２のプール上で前記タスクをスケジューリングするのに利用されない第１のスケジューラを少なくとも部分的に利用して行われる請求項１に記載の方法。
【請求項３】
前記移行の決定が、前記第１のプールにおける障害の検出に基づいて行われる、請求項１に記載の方法。
【請求項４】
前記タスクの前記再割当てが、前記第１のプール内の検出された障害に応答するものである請求項３に記載の方法。
【請求項５】
前記移行の決定が、前記第１のプールのロードバランシング評価に基づいて行われる請求項１に記載の方法。
【請求項６】
前記ジョブの移行が、完全なジョブの移行、又は、まだ処理されていないジョブの一部の移行を含む請求項１に記載の方法。
【請求項７】
前記再割当てするステップは、前記第１のプールに関連付けられる１つまたは複数のタスク位置サービスと通信する高レベル位置サービスにより管理される、請求項１に記載の方法。
【請求項８】
前記タスクを前記再割当てするステップが、ユーザ介入なしに前記第１のプールから前記第２のプールへの前記ジョブの移行を可能にする請求項１に記載の方法。
【請求項９】
前記第２の仮想マシンは、前記第１の仮想マシンとは異なる地理的位置にある請求項１に記載の方法。
【請求項１０】
前記第２の仮想マシンは前記第１の仮想マシンとは異なるタスクテナントにある請求項１に記載の方法。
【請求項１１】
前記第１のプールは２つ以上のタスクテナントからなる請求項１に記載の方法。
【請求項１２】
前記第２の仮想マシンは前記ワークアイテムに関連づけられた仕様に部分的に基づいて前記第２のプールに割り当てられる請求項１に記載の方法。
【請求項１３】
前記第１の仮想マシン及び前記第２の仮想マシンは異なる物理プロセッサを利用する請求項１に記載の方法。
【請求項１４】
コンピューティング装置に請求項１～１２のいずれか１項記載の方法を実行させるため

のプログラムを格納した１つまたは複数のコンピュータ記憶媒体。
【請求項１５】
コンピューティング装置に請求項１～１２のいずれか１項記載の方法を実行させるためのプログラム。
【請求項１６】
分散コンピューティング環境内で仮想マシンのプールにジョブを割り当て仮想マシン上でタスクを実行するためのシステムであって、該システムは、
前記分散コンピューティング環境でタスクアカウントに関連するユーザから、ジョブを表すワークアイテムを受信するステップと、
前記分散コンピューティング環境の第１の仮想マシンを第１のプールに自動的に割り振るステップと、
前記第１のプールに関連付けられた第１のスケジューラを用いて、ジョブキューから前記ジョブのタスクを取り出し、前記タスクを前記第１のプール内の前記第１の仮想マシンに割当てる、ステップと、
前記第１のスケジューラを用いて、前記第１のプール上で前記タスクをスケジューリングするステップと、
前記第１の仮想マシン上で前記タスクを実行するステップと、
前記第１のプールからの前記ジョブの移行を決定するステップと、
前記決定に基づき、前記第１の仮想マシンへの前記タスクの割当てを解放するとともに、少なくとも第２の仮想マシンからなる第２のプールに関連付けられた第２のスケジューラを用いて、前記ジョブキューから前記ジョブのタスクを取り出し、前記タスクを前記第２のプール内の前記第２の仮想マシンに再割当てするステップと、
前記第２のスケジューラを用いて、前記第２のプール上で前記タスクをスケジューリングするステップと、
前記第２の仮想マシン上で前記タスクを開始するステップと
を含む方法を実行する１または複数のプロセッサを備えるシステム。

【発明の詳細な説明】
【背景技術】
【０００１】
[0001]大規模な計算ジョブを実施する従来の方法ではしばしば、ユーザは、分散環境でテナントをアクティブに管理し、ジョブに関するキューを管理することが必要である。ユーザのこのアクティブな介入は、ジョブが大きなリソースクラスタに及ぶことができることを阻害すること、およびそうしたクラスタの使用を効率的な方式でスケーリングできることを阻害することがある。さらに、ジョブは従来、ジョブ、ジョブを完了するために使用されるリソース、およびリソース上のジョブのスケジューリングが密に結合される方式で作成されることがあり、障害またはロードバランシングに応答してジョブの効率的な移行が妨げられる。
【発明の概要】
【課題を解決するための手段】
【０００２】
[0002]様々な実施形態では、ジョブを実施するためにプラットフォームによって分散コンピューティング環境内のサービスとしてリソースを提供するシステムおよび方法が提示される。システムのリソース、システム上で実施中のジョブ、およびシステム上で実施中のジョブのスケジューラが、リソース間でジョブが容易に移行することを可能にする方式で分離される。リソースの第１のプールからリソースの第２のプールへのジョブの移行が、人間の介入なしにシステムによって実施されることが企図される。ジョブの移行は、種々のリソースについて種々のスケジューラを利用することができる。さらに、リソースのプールが、ジョブの移行に応答して、追加のリソースまたはより少数のリソースを自動的に割り振ることができることが企図される。

【０００３】
[0003]この「概要」は、以下の「発明を実施するための形態」でさらに説明される概念の抜粋を単純化した形で紹介するために与えられるものである。この「概要」は、特許請求される主題の主要な特徴または不可欠な特徴を特定することを意図するものではなく、孤立して、特許請求される主題の範囲を決定する際の助けとして使用されることを意図するものではない。
【０００４】
[0004]添付の図面を参照しながら、以下で本発明が詳細に説明される。
【図面の簡単な説明】
【０００５】
【図１】[0005]本発明の態様による、本発明の実施形態を実装するのに適した例示的動作環境を示す図である。
【図２】[0006]本発明の態様による、例示的ジョブを示すブロック図である。
【図３】[0007]本発明の態様による、例示的プールを示すブロック図である。
【図４】[0008]本発明の態様による、分散コンピューティング環境内のタスクを実施するのに適した例示的システムのアーキテクチャ層を示すブロック図である。
【図５】[0009]本発明の態様による、単一の分散コンピューティングアーキテクチャ内に統合することのできる、種々の位置のプロセッサの例示的構成を示すブロック図である。
【図６】[0010]本発明の態様による、タスク位置サービスに関する潜在的構成を示すブロック図である。
【図７】[0011]本発明の態様による、例示的タスク位置サービスフロントエンド（「ＴＬＳＦＥ」）を示すブロック図である。
【図８】[0012]本発明の態様による、例示的ワークアイテム／ジョブスケジューラ（「ＷＩＪ」）を示すブロック図である。
【図９】[0013]本発明の態様による、タスクテナントを示すブロック図である。
【図１０】[0014]本発明の態様による、例示的プールサーバを示すブロック図である。
【図１１】[0015]本発明の態様による、例示的分散コンピューティングシステム内の様々な構成要素間の例示的ワークフローを与える通信ダイアグラムである。
【図１２】[0016]本発明の態様による、分散コンピューティング環境内のリソースを提供する例示的方法を示すブロック図である。
【図１３】[0017]本発明の態様による、分散コンピューティング環境内のリソースを提供する追加の例示的方法を示すブロック図である。
【図１４】[0018]本発明の態様による、分散コンピューティング環境内のリソースを提供する別の方法を示すブロック図である。
【図１５】[0019]本発明の態様による、分散コンピューティング環境内の分離されたリソース、スケジューリング、およびジョブを提供する方法を示すブロック図である。
【発明を実施するための形態】
【０００６】
[0020]様々な実施形態では、ジョブを実施するためにプラットフォームによって分散コンピューティング環境内のサービスとしてリソースを提供するシステムおよび方法が提示される。システムのリソース、システム上で実施中のジョブ、およびシステム上で実施中のジョブのスケジューラが、リソース間でジョブが容易に移行することを可能にする方式で分離される。リソースの第１のプールからリソースの第２のプールへのジョブの移行が、人間の介入なしにシステムによって実施されることが企図される。ジョブの移行は、種々のリソースについて種々のスケジューラを利用することができる。さらに、リソースのプールが、ジョブの移行に応答して、追加のリソースまたはより少数のリソースを自動的に割り振ることができることが企図される。
【０００７】
[0021]第１の例示的態様は、分散コンピューティング環境内のリソースを提供するコンピュータ実装方法を含む。この方法は、タスクアカウントに関連するユーザからワークア

イテムを受信することからなる。さらに、この方法は、ワークアイテムからジョブを作成することからなる。加えて、この方法は、プロセッサおよびメモリを用いて、少なくとも第１の仮想マシンを第１のプールに自動的に割り振ることからなることが企図される。この方法は、第１のプール上にジョブを割り当て、第１のプール上のジョブに関するタスクをスケジューリングすることからなるものとさらに企図される。さらに、この方法は、少なくとも第２の仮想マシンからなる第２のプールにジョブを再割当てすることからなると企図される。加えて、この方法は、第２のプール上のジョブに関するタスクをスケジューリングすることからなる。

【０００８】

[0022]本明細書で与えられる第２の例示的態様は、プロセッサおよびメモリを有するコンピューティング装置によって実行されるとき、分散コンピューティング環境内のリソースを提供する方法を実施するコンピュータ使用可能命令を格納するコンピュータ記憶媒体を含む。この方法は、第１のスケジューラを用いて、分散コンピューティング環境内のリソースの第１のプール上のジョブをスケジューリングすることからなる。この方法はさらに、第１のプールにジョブを割り当てることからなる。加えて、この方法は、ユーザ介入なしに、ジョブが第１のプールから分散コンピューティング環境内の第２のプールに移行すべきであると決定することからなる。加えて、この方法は、第２のプール上にジョブを割り当てることからなる。この方法はさらに、プロセッサおよびメモリを使用する第２のスケジューラを用いて、第２のプール上のジョブを自動的にスケジューリングすることからなる。

【０００９】

[0023]本明細書で提示される本発明の第３の例示的態様は、分散コンピューティング環境内のリソースを提供するコンピュータ実装方法を含む。この方法は、タスク位置サービスで、分散コンピューティング環境内のタスクアカウントに関連するユーザからワークアイテムを受信することからなる。この方法はさらに、タスク位置サービスに関連するプールサーバで、分散コンピューティング環境の第１の仮想マシンのセットを第１のプールに自動的に割り振ることからなる。加えて、この方法は、第１のプール上の第１のジョブをスケジューリングすることからなり、第１のジョブは、ワークアイテムに関連する１つまたは複数のタスクからなる。さらに、この方法は、第１のプールのロードバランシング評価に基づいて、ジョブが分散コンピューティング環境内の第２のプールに移行すべきであると自動的に決定することからなる。加えて、この方法は、ワークアイテム／ジョブに関連する少なくとも１つのタスクを第２のプールに移行することからなる。この方法はまた、第２のプールの１つまたは複数のリソース上の少なくとも１つのタスクをスケジューリングすることからなる。

概説

[0024]ネットワークを介するデータ伝送の速度の向上および他のネットワーク機能の改善のために、コンピューティングリソースが大きなネットワークにわたって分散する環境で大規模コンピューティングタスクを実施することがますます可能となっている。第１の位置のユーザは、コンピューティングサービスに、ジョブまたはコンピューティングタスクをサブミットし、ユーザがそれに対する直接的な知識を持たないコンピュータのグループ上で、タスクを実施させることができる。ユーザのタスクを実施するコンピューティングリソースは、複数の物理的位置にわたって分散することがあり、複数の物理的位置は、種々の州、国、および／または大陸に及ぶことがある。１つまたは複数の位置に位置するコンピューティングリソースの第１のグループは、ユーザのコンピューティングタスクを実施するデータおよび他の情報を格納することができ、一方、同一の位置、または場合によっては１つまたは複数の位置の異なるセット内のコンピューティングリソースの第２のグループを使用して、コンピューティングタスクを実施することができる。さらに、ユーザによる知識を伴って、または伴わずに、データを格納する１つまたは複数の位置を種々の地理的領域にわたって分散させることができることが企図される。同様に、ユーザの知識を伴って、または伴わずに、コンピューティングリソースを種々の地理的位置にわたっ

て分散させることができることが企図される。

【００１０】

[0025]様々な分散コンピューティングリソースへのアクセスは、コンピューティングリソースがどこに位置するかを気にすることなく、ユーザが（例えばプロセスを実行する）ジョブタスクを実施することを可能にする。分散リソースはまた、指定の時間または所望のコスト価値でコンピューティングタスクを完了することなどのコンピューティングタスクの目標を満たすために使用されるリソース量をユーザがスケールアウト（またはスケールイン）する機会を与える。しかし、ユーザのためにこの柔軟性を提供することは、分散コンピューティングリソースのオペレータ（および／または所有者）に対していくつかの課題を課す。要求を満たすために、リソースの分散ネットワークのオペレータは、ピーク需要時のリソース要求を満たすために十分な利用可能なリソースを有することが好ましい。

例示的コンピューティング環境

[0026]図面を全般的に参照し、具体的には最初に図１を参照すると、本発明の実施形態を実装するのに適した例示的動作環境が示されており、全体的にコンピューティング装置１００として示されている。コンピューティング装置１００は、適切なコンピューティング環境の一例にすぎず、本発明の使用または機能の範囲に関して何らかの制限を示唆することを意図するものではない。コンピューティング装置１００が、図示されるモジュール／構成要素のうちのいずれか１つまたは組合せに関する何らかの依存関係または要件を有すると解釈すべきでもない。

【００１１】

[0027]コンピュータ、または携帯情報端末や他のハンドヘルド装置などの他のマシンによって実行されるプログラムモジュールなどのコンピュータ実行可能命令を含むコンピュータコードまたはマシン使用可能命令の一般的状況で実施形態を説明することができる。概して、ルーチン、プログラム、オブジェクト、モジュール、データ構造などのプログラムモジュールは、特定のタスクを実施し、または特定の抽象データタイプを実装するコードを指す。ハンドヘルド装置、コンシューマエレクトロニクス、汎用コンピュータ、専用コンピューティング装置などを含む様々なシステム構成で実施形態を実施することができる。通信ネットワークを通じてリンクされるリモート処理装置でタスクが実施される分散コンピューティング環境で実施形態を実施することもできる。

【００１２】

[0028]引き続き図１を参照すると、コンピューティング装置１００は、以下の装置を直接的または間接的に結合するバス１１０を含む：メモリ１１２、１つまたは複数のプロセッサ１１４、１つまたは複数のプレゼンテーションモジュール１１６、入出力（Ｉ／Ｏ）ポート１１８、Ｉ／Ｏモジュール１２０、および例示的電源１２２。バス１１０は、１つまたは複数のバス（アドレスバス、データバス、それらの組合せなど）とすることができるものを表す。図１の様々なブロックは、見やすいように線で示されているが、実際には、様々なモジュールを描くことはそれほど明確ではなく、比喩的には、線は、より正確にはあいまいでファジーなものとなる。例えば、ディスプレイ装置などのプレゼンテーションモジュールをＩ／Ｏモジュールとみなすことができる。さらに、プロセッサはメモリを有する。本明細書の発明者らは、そのようなことは当技術分野の性質であることを認め、図１のダイアグラムは、１つまたは複数の実施形態と共に使用することのできる例示的コンピューティング装置の例にすぎないことを繰り返し述べておく。「ワークステーション」、「サーバ」、「ラップトップ」、「ハンドヘルド装置」などのようなカテゴリ間の区別は、そのすべてが図１の範囲内で企図され、「コンピュータ」または「コンピューティング装置」に対する参照であるので、行われない。

【００１３】

[0029]コンピューティング装置１００は、典型的には、様々なコンピュータ可読媒体を含む。限定ではなく例として、コンピュータ可読媒体は、ランダムアクセスメモリ（ＲＡＭ）、読取り専用メモリ（ＲＯＭ）、電子消去可能プログラマブル読取り専用メモリ（Ｅ

ＥＰＲＯＭ）、フラッシュメモリ、または他のメモリ技術、ＣＤＲＯＭ、デジタルバーサタイルディスク（ＤＶＤ）、または他の光もしくはホログラフィック媒体、磁気カセット、磁気テープ、磁気ディスクストレージ、または他の磁気記憶装置、あるいは所望の情報を符号化するのに使用することができ、コンピューティング装置１００でアクセスすることのできる任意の他の媒体を含むことができる。

【００１４】

[0030]メモリ１１２は、揮発性および／または不揮発性メモリの形態の非一時的コンピュータ記憶媒体を含む。メモリは、取外し可能、取外し不能、またはそれらの組合せでよい。例示的ハードウェア装置は、固体メモリ、ハードドライブ、光ディスクドライブなどを含む。コンピューティング装置１００は、メモリ１１２またはＩ／Ｏモジュール１２０などの様々なエンティティからデータを読み取る１つまたは複数のプロセッサを含む。プレゼンテーションモジュール（複数可）１１６は、ユーザまたは他の装置にデータ指示を提示する。例示的プレゼンテーションモジュールは、ディスプレイ装置、スピーカ、印刷モジュール、振動モジュールなどを含む。Ｉ／Ｏポート１１８は、Ｉ／Ｏモジュール１２０を含む他の装置にコンピューティング装置１００を論理的に結合することを可能にし、他の装置の一部を内蔵することができる。例示的モジュールは、マイクロフォン、ジョイスティック、ゲームパッド、サテライトディッシュ、スキャナ、プリンタ、ワイヤレス装置などを含む。

定義

[0031]「アカウント」は、分散コンピューティング環境内のグローバル一意識別のエンティティ（global uniquely identified entity）である。一実施形態では、以下で論じるリソースおよびタスクのすべては、アカウントの範囲内にある。典型的には、ユーザは、分散コンピューティングシステムのリソースを使用する前にアカウントをまず作成する。アカウントを作成した後、ユーザは、アカウントを使用して、システムにワークアイテムをサブミットし、ワークアイテムに基づいてジョブを実施するリソースを管理することができる。

【００１５】

[0032]「ワークアイテム」は、分散コンピューティング環境内で実行されるジョブの静的表現である。ワークアイテムは、ジョブバイナリ(job binary)、処理すべきデータに対するポインタ、および任意選択でジョブを実施するタスクを立ち上げるためのコマンドラインを含む、ジョブの様々な側面を指定することができる。さらに、ワークアイテムは、反復スケジュール(reoccurrence schedule)、優先順位、および制限を指定することができる。例えば、ワークアイテムは、毎日午後５時に立ち上げるように指定することができる。

【００１６】

[0033]「ジョブ」は、ワークアイテムの実行中インスタンスである。ジョブは、分散計算を実施するために共に働くタスクの集合を含む。タスクは、分散コンピューティング環境内の１つまたは複数の仮想マシン上で実行することができる。ジョブは、以下で図２に関連してより詳細に論じられる。

【００１７】

[0034]「タスク」は、ジョブの基本実行単位である。各タスクは仮想マシン上で実行される。ユーザは、コマンドラインに対する追加の入力、および各タスクについての入力データに対するポインタを指定することができる。タスクは、タスクの実行中に、タスクを実施する仮想マシン上のその作業ディレクトリ下にファイルの階層を作成することができる。

【００１８】

[0035]「ジョブ・マネージャ・タスク」（本明細書では「ＪＭタスク」とも呼ばれる）は、ジョブ内の特別なタスクである。ジョブ・マネージャ・タスクは任意選択であり、したがって一部のジョブを、ＪＭタスクを使用せずに実施することができる。ジョブ・マネージャ・タスクは、ジョブ内のタスクのすべてについて単一の制御点を提供することがで

き、ジョブに関する「マスタ」タスクとしてマネージャタスクを使用することができる。ジョブがＪＭタスクを有する場合、システムは、ＪＭタスクをジョブ内の第１のタスクとして立ち上げる。次いで、ＪＭタスクは、ジョブにさらにタスクをサブミットすることができ、これらのタスクの進行を監視し、タスクの次のバッチをいつサブミットするかを制御することができる。しかし、ＪＭタスクがジョブに関連するときであっても、システムの外部の１つまたは複数のサービスによってタスクをジョブにサブミットすることもできることも企図される。このようにして、ＪＭタスクは、ジョブ内のタスクのスケジューリングを調整し、タスク間の依存関係を管理することができる。ジョブ・マネージャ・タスクに関するノードまたは仮想マシンに障害が発生した場合、ＪＭタスクが対応するジョブについて常に実行中となるように、別の仮想マシン上でＪＭタスクを再スタートすることができる。例示的態様では、ＪＭタスクが別の仮想マシン上で自動的に再スタートされる。加えて、ユーザは、ＪＭタスクが完了した後に、対応するジョブ内のすべてのタスクをシステムが終了することができることをシステムに指定することができる。例示的ＪＭタスク（またはジョブ・タスク・マネージャともよばれる）は、図２に関連して以下で論じられる。

ジョブ

[0036]上記で定義したように、ジョブは、ワークアイテムの実行中インスタンスである。図２に、本発明の態様による、例示的ジョブ２００のブロック図を示す。ジョブ２００は、タスク２０４などの複数のタスクからなる。上記でやはり定義したように、タスクは、タスクテナント内の仮想マシン（本明細書ではタスクマシンとも呼ばれ、以下で論じられる）によって実行されるジョブの基本実行単位である。複数のタスクに加えて、任意選択で、ジョブ２００は、ジョブ・タスク・マネージャ２０２（本明細書ではＪＭタスクとも呼ばれる）からなることがある。ジョブ・タスク・マネージャ２０２は、ジョブ２００内の他のタスク（例えば、タスク２０４）のすべてについて単一の制御点を提供することができる。

【００１９】

[0037]本明細書でより詳細に論じられるように、ワークアイテムはジョブの静的表現である。例示的実施形態では、分散コンピューティングシステムのリソースが計算リソースを開始する（例えば、ジョブをロードし、ジョブをキューに入れ、ジョブ内のタスクをインスタンス化する）と、ワークアイテムはジョブと呼ばれる。別の言い方をすれば、例示的態様では、システムがワークアイテムの処理を開始すると、ワークアイテムはジョブとなる。

仮想マシンプール

[0038]仮想マシンとは、処理能力の論理単位を指す。仮想マシンは、物理プロセッサと１対１対応を有することができ、または仮想マシンは、複数のプロセッサに対応することができ、または仮想マシンは、１つまたは複数のプロセッサ上の処理時間／サイクルの割合を表すことができる。そのことに関わらず、例示的態様では、仮想マシンを少なくとも部分的に、図１のコンピューティング装置１００によって説明できることが企図される。

【００２０】

[0039]様々な実施形態では、ワークアイテムに基づくジョブを実施することのできる仮想マシンが、使用前にワークアイテムに関するアカウントに関連付けられる。「プール」は、仮想マシンの論理グルーピングである。図３に、本発明の態様による、仮想マシン３０２などの複数の仮想マシンからなるプール３００のブロック図を示す。プールは、種々のデータセンタ、異なる地理的位置、および種々の物理的構成に及ぶ仮想マシンを割り振っていることがあることが企図される。

【００２１】

[0040]例示的態様では、ワークアイテムは常に、ワークアイテムに対応するジョブ（複数可）を実行するために少なくとも１つの関連するプールを有する。各アカウント（例えば、タスクアカウント）は、アカウントに関連するワークアイテムを実施する際に使用される、それに対する排他的アクセスをアカウントが得る１つまたは複数のプールを作成す

ることができる。ワークアイテムがユーザによってサブミットされるとき、プールを作成することができ、またはワークアイテムを既存のプールに関連付けることができる。任意選択で、アカウントに対応する単一のワークアイテムまたはワークアイテムのサブセットと共に使用するためにプールを関連付けることができる。さらに、ジョブのためにシステムによってプールを自動的に作成できることが企図される。例えば、反復ワークアイテムが毎日特定の時刻に実行され、通常は完了するのに２時間を必要とすることがある。この例では、ジョブが作成されるとき、プールを毎日自動的に作成することができ、ジョブが完了したとき、プールを削除することができる。

【００２２】

[0041]ワークアイテムがユーザによってサブミットされるとき、例示的態様では、ワークアイテムを仮想マシンの１つまたは複数のプールに関連付けることができる。さらに、ワークアイテムを単一のプールに排他的に関連付けることができることが企図される（さらに、複数のワークアイテム／ジョブを共通プールに関連付けることができることが企図される）。任意の好都合な方式で、仮想マシンをプール内に編成することができる。例えば、仮想マシンに関する基礎となるプロセッサの地理的位置の如何に関わらず、すべての仮想マシンを単一のプール内に編成することができる。別のオプションは、プールに関するすべての仮想マシンが所与の地理的位置内にあるように、地理的位置に基づいて仮想マシンを編成することである。さらに別のオプションは、他の変数（例えば、ストレージリソース、ネットワーク待ち時間、ユーザ位置／プリファレンス、およびセキュリティ要件）に対する近接などの地理的位置以外に基づいて仮想マシンを編成することである。

【００２３】

[0042]プールを形成する、別の企図されるプロセスは、システムリソースを利用してプールを自動的に生成する。自動プール作成は、ワークアイテムが作成されるとき、またはジョブ自体が作成されるときに、システムによってプールを自動的に作成することを可能にする。このプロセスは、顧客／ユーザ／クライアントからのプールの作成を抽象化する。このモード動作では、顧客／ユーザ／クライアントは、ワークアイテムまたはジョブを実行するその責任が限定される。例示的態様では、ジョブを実行するときにプールが自動的に作成され、ジョブが終了したときにプールが自動的に解体される。さらに、ワークアイテムが作成されるときにプールが自動的に作成され、ワークアイテムが削除される／終了するときにプールが自動的に解体(tear down)されることが企図される。

【００２４】

[0043]例示的態様では、仮想マシンが１つのタスクおよび／または多数のタスクを実行できることが企図される。さらに、複数のジョブが同一の仮想マシンプール上で実行できることが企図される。例示的態様では、ＶＭプールは、ユーザ介入なしに、かつジョブからの明示的な処理なしに、サイズが自動的に増大および収縮することができる。例えば、ジョブは、プールの拡大または収縮を補償する役目を果たさないことがある。同様に、ジョブが複数のプール及ぶことができることが企図される。ＶＭリソース内で独立して増大および収縮することのできる複数のプールにわたってジョブをロードバランシングすることにより、複数のプールに及ぶことを達成することができる。さらに、プールが所与の時間に０個の仮想マシンからなることがあることが企図される。このことは、ジョブが実行するタスクがないときに生じることがある。その結果、計算リソースを節約するための期間中に、プールが０個のＶＭまで収縮することがあることが企図される。

専用マシン、スタンドバイマシン、およびプリエンプティブル(Preemptible)マシン

[0044]例示的実施形態では、仮想マシンがプールに割り当てられるとき、仮想マシンは、２つのタイプのうちの（少なくとも）１つにあることができる。仮想マシンを専用仮想マシンまたはプリエンプティブル仮想マシンとしてプールに割り当てることができる。仮想マシンの専用またはプリエンプティブルとしてのステータスは、仮想マシンがプール内にある間に変化することもできる。

【００２５】

[0045]「専用」仮想マシンは、プールに割り当てられたワークアイテム／ジョブが専用

に使用するためのプールに割り当てられたマシンである。任意選択で、プールにサブミットされた任意のジョブについて全般に利用可能であるのではなく、１つまたは複数の関連するワークアイテムのために専用に使用するために専用仮想マシンを割り当てることができる。仮想マシンが専用ステータスを有する間、マシンは、プールに関連するワークアイテム／ジョブによる使用のために予約される。

【００２６】

[0046]「プリエンプティブル」仮想マシンは、アカウントの代わりにプール内のジョブを現在実施している仮想マシンであるが、仮想マシンがそのプールのために引き続き利用可能となるという保証はない。プリエンプティブル仮想マシンがプールに対して利用可能にされるとき、プリエンプティブルマシンがそのプールに追加され、次いでそのプールをワークアイテム／ジョブで使用することができる。次いで、プリエンプティブルマシンは、そのプールに関するジョブを実施するためにプロビジョニングされ、使用される。アカウントの代わりに、プールに、リソースオークションでプリエンプティブル仮想マシン上の処理時間を勝ち取らせることなどの任意の好都合な方法により、プールに対してプリエンプティブルマシンを利用可能にすることができる。

【００２７】

[0047]アカウントに対する使用のために利用可能にされる仮想マシンは典型的には、分散コンピューティング環境で別の目的を有する仮想マシンである。例えば、プリエンプティブル仮想マシンの１つのソースは、災害回復目的で分散コンピューティング環境所有者／オペレータによってプロビジョニングされる仮想マシンである。安定した動作を実現するために、分散コンピューティング環境は、予備で保持される１つまたは複数のグループ仮想マシンを含むことができる。これらの予備仮想マシンは、プロセッサ障害、ネットワーク障害、または結果として分散環境の一部がもはやジョブを実施するのに適さなくなる任意の他の種類のイベントのために失われるリソースを置き換えるために利用可能である。プールに割り当てられた１つまたは複数の専用仮想マシンがイベントのために失われたとき、予備仮想マシンを使用して、失われたマシンを置き換えることができる。これは、分散コンピューティング環境内のリソースの可用性を改善する。しかし、障害イベントはまれであることが望ましいので、災害回復マシンの確保は多くの場合、多数の仮想マシンがアイドル状態であり、使用されるのを待機していることを意味する。障害イベントを処理するために指定されるこれらの仮想マシンのＣＰＵサイクルを浪費するのではなく、これらの仮想マシンのＣＰＵサイクルをプリエンプティブルＶＭとしてプールに割り当て、ワークアイテムを実行することができる。障害が発生し、専用リソースの要件を満たすためにシステムがプリエンプティブルリソースを取り去る必要がある場合、失われたリソースまたは障害が発生したリソースを置き換えるという元の目的のためにプリエンプティブル仮想マシンを使用することができるように、そのような仮想マシン上で実行中のプリエンプティブルジョブが、実現可能な限り早く（恐らくは直ちに）停止することになる。

【００２８】

[0048]プリエンプティブルマシンの別のソースは、過剰能力仮想マシン(excess capacity virtual machine)である。典型的には、どんなネットワークのピーク負荷も平均負荷とは異なる。その結果、ピーク負荷状況に対処するのに十分なリソースを備えるコンピューティング環境はしばしば、他の時間の間、利用可能な過剰なリソースを有する。こうした過剰なリソースがリソースクッションを提供する。ユーザが追加の専用仮想マシンを求める要求を行うとき、過剰な仮想マシンを使用してユーザの要求を履行することができる。分散コンピューティング環境が専用マシンに関するピーク負荷未満の負荷を有するとき、１つまたは複数の仮想マシンが空きとなる。予備能力を提供するために指定されるこれらの仮想マシンのＣＰＵサイクルを浪費するのではなく、これらの仮想マシンのＣＰＵサイクルをプリエンプティブル的にユーザおよびプールに割り当てることができる。専用仮想マシンを求める要求の負荷が増大するとき、これらの過剰な仮想マシン上で実行中のプリエンプティブルジョブが、実現可能な限り早く（恐らくは直ちに）停止することになる。これは、必要なときに追加の専用リソースを提供するという元の目的のためにプリエン

プティブル仮想マシンを使用することを可能にする。加えて、または代わりに、専用マシンに関する負荷の一部の増大は、専用マシンを求めるスケジューリングされた要求によるものとなる。仮想マシンがスケジューリングされた時刻に専用マシンとして使用するために利用不能となろうとしている場合、プリエンプティブルジョブから専用リソースへの規則的な遷移を可能にするために、スケジューリングされた時刻の前に、仮想マシンに割り当てられたプリエンプティブルジョブを停止することができる。

【００２９】

[0049]仮想マシンのさらに別のソースは、「スタンドバイ」予約でプールまたはアカウントに関連付けられる仮想マシンである。「スタンドバイ」仮想マシン予約は、第１のアカウントまたはプールに関連付けられ、第１のアカウントまたはプールによる使用のためにプロビジョニングされる仮想マシン予約である。さらに、スタンドバイ仮想マシン予約が特定のプールにタグ付けさず、その代わりにアカウントまたはプールによってスタンドバイのために保たれる定義済みの数のＶＭをシステムが維持することが企図される。スタンドバイＶＭが必要であるとき、必要な数のＶＭを満たすようにスタンドバイＶＭの予約数を削減することができる。例示的態様では、システム内の全ＶＭ数は、専用ＶＭ＋予約されたスタンドバイＶＭ＋他の所に割り当てられない計算のために空いているＶＭ＋災害および他のプロビジョニングのために予約されているＶＭに等しくてよい。この例では、システムで使用されるプリエンプティブルＶＭの数が、予約されるスタンドバイＶＭ＋コンピューティングのために空いているＶＭの数以下であることが企図される。

【００３０】

[0050]スタンドバイマシン予約をプロビジョニングすることは、システム内のどこか（例えば、プールレベル、アカウントレベル）にＶＭ能力を確保することを含むことができる。スタンドバイ仮想マシン予約は仮想の割振りではない。そうではなく、スタンドバイ仮想マシン予約は、アイドルまたはプリエンプティブルＶＭを取り、それをそのプールまたはアカウント使用のために専用ＶＭに変更する将来の権利を予約する。カウントであるスタンドバイＶＭを２つの異なるプールに関連付けることができる。

【００３１】

[0051]スタンドバイマシンに関する１つの使用は、特定の時間フレーム中にのみ行われる高優先順位計算ジョブを有するユーザに関するものである。例えば、金融会社は、証券取引所や商品取引所などの１つまたは複数の金融市場の日々の活動の分析を実施したいことがある。この例を続けると、金融市場は、午前９：３０に開き、午後４：００に閉じるなど、定義済みのスケジュールで開き、閉じることがある。金融会社は、分析またはシミュレーションを実施する際に使用するために、金融市場が開いている時間からデータを集約することを望む。分析の目標は、市場が翌日に開く前に、従業員のための情報を提供することである。そのような分析は多数の仮想マシンを必要とする可能性があるが、仮想マシンが必要なのは、例えば午後６：００から翌朝の午前３：３０までの時間だけである。この時間中、金融会社は、仮想マシンの可用性の保証を望む。その日の残りの間は、金融会社はマシンを必要としない。金融会社のアカウントにスタンドバイ仮想マシン予約を割り当てることが、この目標を達成する。予約価格を支払うのと引き換えに、金融会社は、所望の時間中のマシンの可用性が保証される。所望の時間ウィンドウの外部では、仮想マシンを金融会社および／または他のユーザ用のプリエンプティブルマシンとして使用することができる。予約を履行するためにプリエンプティブルＶＭを取る必要があるスタンドバイ予約を実行するとき、スケジューリングされた可用性イベントの前に、プリエンプティブルジョブを規則的に停止することができる。

【００３２】

[0052]スタンドバイＶＭ予約が専用マシンに変換されるとき、これは、時間ベースの基準に基づく変換と定義される。言い換えれば、スタンドバイＶＭ予約は、所定の時間および／または日付に少なくとも部分的に基づいて専用マシンに変換される。時間ベースの基準は、活動しきい値を定義するのに使用される活動基準とは対照的である。活動しきい値は、１つまたは複数の分散リソースの使用および／または性能に基づくしきい値に対応す

る。例えば、アカウントでは、顧客は、いくつかのスタンドバイＶＭを予約するために支払いをすることができ、そのスタンドバイＶＭ予約を時間ベースの基準または動的しきい値自動スケーリング基準のどちらかのために使用することができる。さらに、予約時間または他のスケジューリング予約ポリシーの如何に関わらず、任意の時点でスタンドバイＶＭ予約を変換することができることが企図される。例えば、ユーザ（または管理者）は、予約からの１つまたは複数のスタンドバイＶＭを変換すべきであるという要求を与えることができる。

【００３３】

[0053]スタンドバイＶＭ予約に関する別の使用は、ジョブをスケールアウトするときに性能の改善を可能にすることである。例えば、小売店は、分散コンピューティングリソースを使用して、小売業者のウェブサイトを検討して注文を行うオンライントラフィックなどの、祝日前のショッピングシーズン中の追加のオンライントラフィックを処理することができる。過去の経験に基づいて、小売業者は、一定のレベルのオンライン活動を予想し、対応する数の専用仮想マシンを予約する。しかし、オンライン活動が予想よりも大きい場合、小売業者は、スタンドバイモードの追加のマシンも予約する。次いで、小売業者は、予想されるレベルの活動よりも高いことを示す１つまたは複数のしきい値をセットアップすることができる。これらのしきい値が生じたとき、小売業者の顧客が低速な応答時間を経験することなく小売業者が追加のオンライントラフィックを処理することを可能にするように、スタンドバイＶＭ予約を使用してアイドルマシンまたはプリエンプティブルマシンを変換することができる。この状況では、活動しきい値がいつ満たされるかは既知ではないことがあるので、スタンドバイＶＭ予約が予測できない時間に専用マシンに変換されることがある。活動しきい値が満たされるとき、アイドルＶＭが使用され、またはプリエンプティブルタスクが停止し、マシンが専用マシンに変換される。

分散ネットワーク環境内のコンピューティングリソースの編成の例

[0054]分散コンピューティング環境のユーザは典型的には、分散コンピューティングリソース（例えば、クラウドコンピューティングリソース）を使用してジョブを実施することを望む。ジョブは典型的には、ネットワーク（例えば、インターネット）などによって分散コンピューティング環境を介してアクセス可能な位置に格納されるデータに対するジョブを実行することを含む。オペレータが分散コンピューティング環境を提供する一方式は、環境をいくつかの層として提供することである。図４に、本発明の態様による、分散コンピューティング環境内のタスクを実施するのに適した例示的システムの層を形成するブロック図を示す。図４のシステムは、システム・エージェント・ランタイム層４０８（タスクランタイム層とも呼ばれることがある）、アプリケーション・ランタイム層４０６（サードパーティ・タスク・ランタイム層とも呼ばれることがある）、リソース管理層４０２、ならびにスケジューリングおよび実行層４０４を含む。

【００３４】

[0055]図４に示す実施形態では、システム・エージェント・ランタイム層４０８は、実行環境、および実行環境でプールおよびワークアイテム／ジョブ／タスクを実行するためのセキュリティコンテキストをセットアップする役目を果たす。システム・エージェント・ランタイム層４０８はまた、タスクを立ち上げ、タスクのステータスを監視することができる。システム・エージェント・ランタイム層４０８は、各仮想マシン上で動作するシステムエージェントの形を取ることができる。タスクランタイム層はまた、ユーザのタスク実行ファイル(task executable)にリンクすることができるランタイムライブラリも含むことができる。ランタイムライブラリをシステム・エージェント・ランタイム層４０８として有することは、システムエージェントによって実行されるタスクにより豊かな能力を潜在的に与えることができる。ランタイムライブラリの例は、タスク間の高速通信を可能にする１つまたは複数の効率的な通信ライブラリと、他の仮想マシンおよび／または他のタスクからファイルを読み取るための効率的なリモート・ファイル・アクセス・ライブラリ・サポートと、タスクが（例えば、バイナリ大型オブジェクトに）チェックポイントし、再開することを可能にするライブラリ、ロギングライブラリと、仮想マシンのプール

内の所与のタスクを実施する仮想マシンにわたって使用される分散ファイルシステムを提供するライブラリとを含む。

【００３５】

[0056]アプリケーション・ランタイム層４０６は、システム・エージェント・ランタイム層４０８の上部で追加のランタイムを構築および実行することを可能にする。アプリケーション・ランタイム層４０６は、ジョブに関するタスクの実行を調整する追加の機能を提供することができる。例は、ＶＭのプール内の所与のタスクを実施するＶＭにわたって使用される分散ファイルシステムを提供するライブラリに対するＭａｐ　Ｒｅｄｕｃｅランタイムを含むことができる。これは、ユーザがユーザのジョブまたはタスクに合わせて調整する方式で分散コンピューティング環境を編成することを可能にする。

【００３６】

[0057]リソース管理層４０２は、分散コンピューティング環境で利用可能なコンピューティングリソースを管理することを扱う。１つのオプションは、リソース管理層４０２に３つの異なるレベルでリソースを管理させることである。第１のレベルで、リソース管理層４０２は、ジョブ（すなわち、ワークアイテムの実行）に関連する仮想マシンの割振りおよび割振り解除、ならびに仮想マシンレベル４１０で示されるタスクに関連する各仮想マシン上に格納されたファイルを管理する。

【００３７】

[0058]第２のレベルで、ジョブに関連する仮想マシンを、プールレベル４１２で示されるマシンプールにグループ化することができる。プールは、１つまたは複数のジョブおよび／またはワークアイテムに関連する仮想マシンを含むことができる。実施形態に応じて、単一のプールは、データセンタ内のすべてのクラスタ、複数のデータセンタにわたる複数のクラスタなどの複数のクラスタにわたることができる。単一のプールは、数百万などの多数の仮想マシンを含むことができる。仮想マシンを数十億などの多数のプールに含めることができる。

【００３８】

[0059]リソース管理層４０２の第３のレベルで、リソース管理層は、タスクテナントのそれぞれのサイズを管理する（以下で論じる）。これは、システムが全体として、システムの現負荷、他の割振りのためのシステムへの未使用仮想マシンを適時の解放などのいくつかのメトリックに基づいて利用される計算リソースの量を動的に調節することを可能にする。このレベルは、テナントレベル４１４によって示される。

【００３９】

[0060]図４に示す実施形態では、スケジューリングおよび実行層４０４は、ユーザによって実施されているワークアイテム、ジョブ、およびタスクを管理する。スケジューリングおよび実行層４０４は、スケジューリング決定を行い、ジョブおよびタスクならびに障害時の再試行を立ち上げる役目を果たす。そのようなスケジューリングおよび実行層４０４は、ジョブおよび／またはタスクを様々なレベルで管理する構成要素を含むことができる。

【００４０】

[0061]例示的構成要素は、ワークアイテムおよびジョブ管理構成要素４１６、タスク管理およびスケジューリング構成要素４１８、ならびにジョブ・マネージャ・タスク構成要素４２０を含むことができる。ワークアイテムおよびジョブ管理構成要素４１６は、１人または複数のユーザ（またはすべてのユーザ）が作成した／システムに通信したすべてのワークアイテムを維持する。次いで、アクティブなワークアイテムのそれぞれの仕様に基づいて、システムは、タスクがそれに対してサブミットされるジョブおよびジョブキューを作成することができる。ワークアイテムは、反復スケジュールを指定することができる（例えば、毎日午後５時）。次いで、システムは、ワークアイテムに関する反復スケジュールに従ってジョブを作成することができる。ワークアイテムおよびジョブ管理構成要素４１６はまた、システム内のワークアイテムおよびジョブの終了およびクリーンアップを管理する役目も果たすことができる。

【００４１】
[0062]タスク管理およびスケジューリング構成要素４１８は、システムのユーザがシステムのジョブにタスクをサブミットする（例えば、通信する）ことを可能にする。この構成要素は、システム内のジョブ／ワークアイテムのすべてにわたってタスクをスケジューリングすると共に、これらのタスクのステータスを追跡する役目を果たす。タスク管理およびスケジューリング構成要素４１８は、タスクテナント（すなわち、タスクマシン）の１つ／一部／すべてにわたって分散するタスクスケジューラのセットを含むことができ、各スケジューラがリソース（例えば、仮想マシン）およびタスクの限られたセットのみと関連付けることを可能にする。したがって、例示的態様では、タスク管理およびスケジューリング構成要素４１８は、システムが、様々なタスクテナントにわたる仮想マシン上で実行中の数十億のアクティブなタスクをサポートすることを可能にする。
【００４２】
[0063]ジョブ・マネージャ・タスク構成要素４２０は、任意選択で各ジョブに関連付けることのできるＪＭタスクを、ジョブの初期タスクとして立ち上げることを可能にする。先に論じたように、ＪＭタスク（タスクジョブマネージャとも呼ばれる）は、特定のジョブ内のタスクの単一の制御点を提供する。これは、ＪＭタスクがジョブに追加のタスクをサブミットし、これらのタスクの進行を監視することを可能にし、それは、ＪＭタスクがジョブをいつ終了するかを制御することを可能にする。ＪＭタスクは、アプリケーション・ランタイム層４０６がそのシステムを制御および実行する援助する制御を機構の一例とすることができる。
【００４３】
[0064]上述の層を、複数の地理的位置のプロセッサを含む分散コンピューティング環境で実装することができる。図５に、本発明の態様による、単一の分散コンピューティングシステム５００内に統合することのできる、種々の位置のプロセッサの例示的構成を示すブロック図を示す。
【００４４】
[0065]図５では、１つまたは複数のタスクテナント５１４を使用して、仮想マシンのプールを管理することができる。タスクテナント５１４は、（図９に関連して以下でより詳細に論じるように）仮想マシンのセットを維持することができる。１人または複数のユーザのジョブは、タスクテナント５１４内の仮想マシン上で、仮想マシンの１つまたは複数のプールの部分として実行することができる。所与の地理的領域で１つまたは複数のタスクテナント５１４を使用することができる。タスクテナント５１４の責任は、仮想マシンのセットを維持すること、およびタスクテナント内のリソース利用に基づいてタスクテナントを動的に増大または収集することを含むことができる。これは、タスクテナント５１４が顧客需要の増大に対処するためにタスクテナント内の仮想マシン数を増加させることを可能にする。これはまた、他の顧客に関するデータセンタ処理サービス内の他のホストされるサービスに仮想マシンを割り振ることができるように、タスクテナント５１４が未使用の仮想マシンを解放することも可能にする。タスクテナント５１４の別の責任は、プール割振り／割振り解除／管理ロジックの部分を実装することとすることができる。これは、どのように仮想マシンが顧客に関するタスクに関連するプールに割り当てられるかを決定することにタスクテナント５１４が参加することを可能にする。タスクテナント５１４はまた、タスクテナント内の仮想マシン上のタスクのスケジューリングおよび実行の役目も果たすことができる。
【００４５】
[0066]図５に示す実施形態では、（図６～図８に関連して以下でより詳細に論じるように）所与の地理的領域内の、および／または様々な地理的領域にわたるタスクテナント５１４のすべてを制御する１つまたは複数のタスク位置サービス５１２（「ＴＬＳ」）が設けられる。図５では、「地理領域１」および「地理領域２」と名前が付けられた領域をサービスするタスク位置サービス５１２が示されている。タスク位置サービス５１２の責任は、その所与の領域に関するタスクアカウントの管理を含むことができる。タスク位置サ

ービス５１２はまた、ユーザが分散コンピューティング環境と対話することを可能にするアプリケーションプログラミングインターフェース（ＡＰＩ）も提供することができる。そのようなＡＰＩは、仮想マシンのプールに関連するＡＰＩ、プール管理ロジック、および所与の地理的領域内のタスクテナントにわたるプール管理ロジックの調整を処理することを含むことができる。ＡＰＩはまた、ユーザによってサブミットされたタスクを処理するＡＰＩ、ならびにユーザタスクに関連するワークアイテムまたはジョブを維持し、スケジューリングし、終了するＡＰＩも含むことができる。ＡＰＩは、地理的領域内のすべてのワークアイテム、ジョブ、タスク、およびプールに関する統計収集、アグリゲーション、およびレポーティングのためのＡＰＩをさらに含むことができる。加えて、ＡＰＩは、仮想マシンに関するスポットマーケットに基づいて短期的にユーザにとってプリエンプティブルＶＭとして利用可能な仮想マシンのオークションを可能にするＡＰＩを含むことができる。ＡＰＩはまた、使用量を計量し、課金サポート(billing support)を提供するＡＰＩも含むことができる。

【００４６】

[0067]グローバル位置サービス５０２（「ＸＬＳ」）によってタスク位置サービス５１２を互いにリンクすることができる。グローバル位置サービス５０２は、タスク位置サービス５１２と共にタスクアカウントを管理することを含む、アカウント作成およびアカウントの管理の役目を果たすことができる。例えば、グローバル位置サービスは、データセンタ災害の場合に災害回復およびワークアイテムおよびジョブの可用性の役目を果たすことができる。これは、データセンタが利用可能ではないために、異なる位置でワークアイテムまたはジョブを実行すること、ならびに顧客がそのワークアイテム、ジョブ、およびプールをあるデータセンタから別のデータセンタに移行することを可能にすることを含むことができる。典型的には、所与の時間にシステム５００内に存在するアクティブなグローバル位置サービス５０２は１つだけとなる。このアクティブなグローバル位置サービス５０２は、様々なタスク位置サービス５１２ならびにデータ記憶を管理するサービス構成要素（図示せず）と通信する。グローバル位置サービスは、ドメイン名サーバ５０４などでグローバルアカウント名前空間を維持することができる。

【００４７】

[0068]図５のシステムの動作の一例として、仮定の顧客またはユーザが、クライアントポータル５０６を使用してシステム５００にアクセスし、グローバル位置サービス５０２によって提供されるインターフェースを介してタスクアカウントを作成することができる。この例では、仮定の顧客はサリーと呼ばれる。任意選択で、タスクアカウントを作成するユーザ要求は、アカウントを作成する必要のある地理的領域を指定することができる。この例では、サリーが、地理領域１に関連するアカウントを要求し、地理領域１は地理領域２のフェイルオーバ領域を有する。それに応答して、グローバル位置サービス５０２は、要求された地理的領域（例えば、地理領域１）に対応するタスク位置サービス５１２と連絡を取り、アカウントを作成する。領域が要求されない場合、要求側ユーザまたは利用可能なリソースに関連する位置などに基づく任意の好都合な方法で選択された領域でタスクアカウントを作成することができる。タスク位置サービス５１２は、その地理的領域内のすべてのアカウントに関するすべての情報を維持する。首尾よく地理領域１に関するタスク位置サービス５１２でアカウントを作成し、潜在的には地理領域２で災害回復コピーを作成した後、グローバル位置サービス５０２は、地理領域１に関するタスク位置サービス５１２の仮想ＩＰアドレスを指すようにサリーのアカウントに関するタスク・サービス・エンドポイントを登録する。例えば、「ｓａｌｌｙ．ｔａｓｋ．ｃｏｒｅ．ｓｅｒｖｉｃｅｃｏｍｐａｎｙ．ｎｅｔ」などのホスト名を地理領域１内のタスク位置サービス５１２の仮想ＩＰアドレスにマッピングするようにドメインネームサービス（ＤＮＳ）記録を作成することができる。この例示的態様では、これは、サリーに関するタスクアカウントの作成を完了する。さらに、地理領域１での障害の場合、グローバル位置サービスは、地理領域２を指すようにＤＮＳ記録を更新できることが企図される。

【００４８】

[0069]アカウントが作成された後、顧客サリーは、「ｓａｌｌｙ．ｔａｓｋ．ｃｏｒｅ．ｓｅｒｖｉｃｅｃｏｍｐａｎｙ．ｎｅｔ」に対して、タスクアカウントにアクセスし、分散コンピューティング環境と対話するためのＡＰＩにアクセスする要求を送ることができる。例えば、サリーは、ＡＰＩにアクセスして、新しいワークアイテムまたはタスクを作成する要求を発行することができる。次いで、ＤＮＳサーバはホスト名を解決することができ、要求が正しいタスク位置サービス５１２にルーティングされる。この例では、要求が地理領域１に関するタスク位置サービス５１２にルーティングされ、地理領域１は要求を処理し、要求されたワークアイテム、ジョブ、またはタスクを作成する。

【００４９】

[0070]本発明の例示的態様では、任意の数のクライアントポータル５０６、地理領域５０８、５１０、タスク位置サービス５１２、および／またはタスクテナント５１４を任意の組合せで実装できることが企図される。例えば、例示的態様では、タスク位置サービス５１２を数十、数百、または数千のタスクテナントに関連付けることができることが企図される。

【００５０】

[0071]図６に、本発明の態様による、タスク位置サービス６０２に関する潜在的構成を示すブロック図を示す。図６に示す構成では、タスク位置サービスは１つまたは複数のアカウントサーバ６１２を含むことができる。アカウントサーバは、作成、削除、またはプロパティ更新を含む、所与の地理的領域内のアカウントに関するアカウント管理を処理する。アカウントフロントエンド６０８は、アカウントサービスに関するフロントエンドノードとして働く。図示されるように、アカウントフロントエンド６０８はアカウント仮想ＩＰアドレス６０４の背後にある。アカウントフロントエンド６０８は、アカウントを作成し、またはアカウントを削除するＡＰＩ要求などの、グローバル位置サービスから来るアカウントＡＰＩ要求を処理する。

【００５１】

[0072]図６の構成はまた、１つまたは複数のプールサーバ６１４も含むことができる。プールサーバ６１４は、所与の地理的領域内の仮想マシンのプールに関するプール管理およびプールトランザクションを処理する。プールサーバ６１４は、プール作成、削除、およびプロパティ更新を処理する。プールサーバ６１４はまた、複数のタスクテナントにわたる高レベル仮想マシン割振りアルゴリズムも管理する。仮想マシン割振りは、所与のユーザに関するストレージとの間の仮想マシンの接続性を考慮に入れることができる。プールサーバは、仮想マシンの割振りに関する他のタスクも実施することができる。

【００５２】

[0073]さらに、プールサーバ６１４がさらに自動スケーリング構成要素６１６からなることがあることが企図される。自動スケーリング構成要素６１６は、プールに関する所望の数の専用仮想マシン、スタンドバイ仮想マシン、および／またはプリエンプティブル仮想マシンを自動的に決定するユーザ提供の自動スケーリング規則（例えば、公式）を実行する役目を果たすモジュールとして形成することができる。言い換えれば、自動スケーリング構成要素は、所与のプールに関連する自動スケーリング規則に基づいて自動スケーリング決定を行う役目を果たすことができる。自動スケーリング構成要素６１６は、プールまたはプールに割り当てられたジョブに関するメトリックを受信し、それらをユーザ提供の（またはシステム提供の）規則と共に適用して、プールに対する自動スケーリングアクションを計算することができる。自動スケーリングアクションは、専用仮想マシン数を増加または減少させること、予約されるスタンドバイ仮想マシンを増加または減少させること、プールに関するプリエンプティブル仮想マシンの目標数を増加または減少させることと共に、それらのリソースに関する入札価格も更新することを含むことができる。

【００５３】

[0074]図６の構成はまた、（図８に関連して以下でより詳細に論じるように）１つまたは複数のワークアイテムまたはジョブスケジューラ（「ＷＩＪ」）６１８も含む。ＷＩＪスケジューラ６１８は、ワークアイテムおよびジョブの作成、削除、および更新を処理す

る。例示的実施形態では、ＷＩＪはまた、関連するワークアイテムで指定されるスケジュールに基づくジョブの作成、ならびにジョブへのＪＭタスクの追加（追加すべき場合）の役目も果たす。加えて、（従来のユーザによる手動作成ではなく）自動プール構成が望ましいとき、ＷＩＪスケジューラ６１８は、ワークアイテムに関連するプールの作成および削除を開始することができる。ＷＩＪスケジューラ６１８はまた、タスク位置サービス内のスケーリングのための汎用パーティショニング機構を使用することもできる。一実施形態では、各タスク位置サービス内に複数のＷＩＪスケジューラ６１８があり、ＷＩＪスケジューラのそれぞれがワークアイテムの範囲を処理する。

【００５４】

[0075]プールサーバ６１４およびＷＩＪスケジューラ６１８は、（図７に関連して以下でより詳細に論じるように）タスク位置サービスフロントエンド６１０を介してユーザから要求を受信する。タスク位置サービスフロントエンド６１０はまた、対応する構成要素を呼び出してユーザからの要求を処理する役目も果たす。図示されるように、タスク位置サービスフロントエンド６１０は、アカウント仮想ＩＰアドレス６０６の背後にある。

【００５５】

[0076]図６の構成は、タスク位置サービスマスタ６２０をさらに含む。一実施形態では、タスク位置サービスマスタ６２０は２つの主な責任を有する。第１に、タスク位置サービスマスタ６２０は、タスク位置サービス６０２内の対応するサーバに関するパーティショニングロジックを実装するマスタシステムとして働く。加えて、タスク位置サービスマスタ６２０は、タスク位置サービスの地理的領域全体に関する各スポット期間の始めにプリエンプティブル仮想マシンに関する新しい市場価格を計算し、またはスポット価格決定の役目を果たす外部サービスとの間でプリエンプティブルマシンを求める入札要求を調整する役目を果たすことができる。タスク位置サービスマスタ６２０は、プールサーバおよびタスクテナントから現入札およびリソース可用性情報を収集し、それに応じて新しい市場価格を計算し、または情報をスポットマーケットサービスに通信する。タスク位置サービスマスタ６２０はまた、地理的領域内のすべてのタスクテナントにわたるプリエンプティブル仮想マシンについてのプールサーバに対する高レベル割振りガイダンスも行う。

【００５６】

[0077]コンピューティング環境の活動および挙動を追跡するために、タスク位置サービスマスタ６２０は、１つまたは複数の統計アグリゲーションサーバ６２２と通信することができる。統計アグリゲーションサーバは、タスク、ジョブ、ワークアイテム、およびプールに関する詳細な統計を収集および集約する役目を果たす。システム内の他の構成要素は、タスクおよび仮想マシンに関する細分化された統計を放出する。統計アグリゲーションサーバは、タスクレベルまたは仮想マシンレベル統計からのこれらの細分化された統計をワークアイテム、アカウントレベル、および／またはプールレベル統計に集約する。ＡＰＩを介して使用するために統計を公開することができる。加えて、統計アグリゲーションサーバは、課金の際に使用するために各アカウントについての１時間ごとの計量記録を生成する役目を果たすことができる。

【００５７】

[0078]図７に、本発明の態様による、例示的タスク位置サービスフロントエンド（「ＴＬＳＦＥ」）７００を示すブロック図を示す。ＴＬＳＦＥ７００は、上記の例示的態様で論じた図６のタスク位置サービスフロントエンド６１０と同様であってもよい。

【００５８】

[0079]ＴＬＳＦＥ７００は、要求プロセッサ７０２、認証および許可モジュール７０４、アカウントマネージャ構成要素７０６、およびビジネスロジックモジュール７０８からなる。代替実施形態では、追加または代替のモジュールおよび／または構成要素が含まれることがあることが企図される。

【００５９】

[0080]要求プロセッサ７０２は、ＨＴＴＰ（複数可）上でエンドポイントを受信および識別し、ユーザ要求を受諾する役目を果たす構成要素である。次いで、要求プロセッサ７

０２は、各要求をキューに入れ、認証および許可モジュール７０４に転送することができる。認証および許可モジュール７０４は、ユーザ要求認証および許可の役目を果たす。例示的態様では、ＴＬＳＦＥは、共用鍵認証を使用して着信要求を認証する。他の形態の認証が企図される。さらに、認証および許可モジュール７０４は、アカウントマネージャ構成要素７０６と対話して、ユーザアカウントについての情報およびキー情報を得る。次いで、認証および許可モジュール７０４は、どこでユーザが操作の実施を要求することを許可されるかを判定することができる（例えば、要求時にアカウントを一時的に使用不能にすることができる）。

【００６０】

[0081]アカウントマネージャ構成要素７０６を使用して、記憶／アクセスアカウント情報をカプセル化することができる。アカウントマネージャ構成要素７０６は、要求時に、例示的態様では、要求が受信されたとき、アカウント情報をロードし、それは、頻繁にアクセスされるアカウントの処理を加速するための（メモリへの）ローカルな情報のキャッシングを可能にする。アカウントプロパティ（例えば、使用不能状態、認証キー）が変化したとき、アカウントテーブルを更新することができる。ＴＬＳＦＥで実行中のアカウントマネージャ構成要素７０６は、キャッシュされたコピー（それが存在する場合）の有効期限が切れたとき、その変化を認識することができ、有効期限は、構成可能な間隔であってもよい。

【００６１】

[0082]要求が認証および許可モジュール７０４を通過すると、ビジネスロジックモジュール７０８は、ユーザによって行われる要求に関するすべてのビジネスロジックを処理することができる。ビジネスロジックモジュール７０８はまた、分散コンピューティングシステム内の他のタスク構成要素と対話することもできる。加えて、ビジネスロジックモジュール７０８が、完了したジョブ・マネージャ・タスクについてＪＭタスク完了キューを監視し、次いでそれは、タスクが完了したとき、図６のＷＩＪスケジューラ６１８に通知を通信することを可能にすることが企図される。

【００６２】

[0083]図８に、本発明の態様による、例示的ワークアイテム／ジョブスケジューラ（「ＷＩＪ」）８００を示すブロック図を示す。ＷＩＪ８００は、図６の例示的ＷＩＪスケジューラ６１８であってもよい。

【００６３】

[0084]ＷＩＪ８００は、要求プロセッサ８０２、キャッシングモジュール８０４、ハートビートモジュール８０６、およびスケジューリングモジュール８０８からなる。先に論じたように、ＷＩＪ８００は、ユーザアカウントによってサブミットされたワークアイテムを受諾し、ワークアイテムスケジュール内の適切な時にジョブをスケジューリングする役割であってもよい。したがって、ＷＩＪ８００がワークアイテムに関するジョブを作成し、それぞれの生み出された新しいジョブに関するキューを作成し、次いでキューが、ジョブに関するタスクをキューに入れるのに使用されることが企図される。次いで、タスクテナントのタスクスケジューラ（図９に関連して以下で論じる）が、キューからタスクを引き抜き、それらをタスクテナントの仮想マシン上でスケジューリングすることができる。ＷＩＪがワークアイテム、および完了したワークアイテム／ジョブをマーキングすること、適切なタスクテナントと連絡を取ってジョブを開始することなどの関連するジョブの寿命管理の役目を果たすことが企図される。

【００６４】

[0085]要求プロセッサ８０２は、ＴＬＳＦＥ（図６のタスク位置サービスフロントエンド６１０など）から受信される要求に関する様々なワークアイテム／ジョブを処理する役目を果たすモジュールである。加えて、アカウントが削除されるとき、アカウントサーバ（例えば、図６のアカウントサーバ６１２）がＷＩＪ８００に要求を通信することが企図される。

【００６５】

[0086]キャッシングモジュール８０４は、ＷＩＪがサービス中のパーティション、ユーザアカウント、アクティブなワークアイテム、アクティブなジョブ、タスクテナントリストなどのメモリ内キャッシュである。このモジュールは、関連するキャッシングポリシーについてキャッシュを最新に保つ役目を果たすことができる。

【００６６】

[0087]ハートビートモジュール８０６は、タスク位置サービスマスタ（例えば、図６のＴＬＳＭ６２０）と共に協調して、ＷＩＪのヘルスおよび負荷ステータスをレポートヘルスする。加えて、ハートビートモジュール８０６が「ハートビート」協調を介してＴＬＳＭから新しいパーティション割当ても受信することが企図される。

【００６７】

[0088]スケジューリングモジュール８０８は、ワークアイテムの指定のスケジュールに従ってワークアイテムに関する新しいジョブをスケジューリングする役目を果たす。スケジューリングモジュール８０８はまた、ワークアイテムおよび／またはジョブを完了済みとマークする役目も果たす。さらに、スケジューリングモジュール８０８は、ワークアイテムまたはジョブの寿命に結びつけられたその寿命を有するプールを作成および削除する役目を果たすことができる。

【００６８】

[0089]図９に、本発明の態様による、タスクテナント９００を示すブロック図を示す。例示的態様では、タスクテナントは、図５のシステム５００でタスクテナント５１４として実装されるものとして企図される。上記のように、タスクテナントは、仮想マシンのプールを管理することを援助することができる。図９に示す実施形態では、タスクテナントは、１つまたは複数のタスクテナントフロントエンド９０４を含む。タスクテナントフロントエンド９０４は、タスク位置サービスと、タスクテナントとの間で要求を渡すことを含む、タスクテナントと、対応するタスク位置サービスとの間の通信のために内部で使用されるタスクテナント仮想ＩＰアドレスの背後にある。

【００６９】

[0090]図９に示す実施形態では、タスクテナントはまた、タスクスケジューラ９０２を含む。タスクスケジューラ９０２は、タスクテナント内のローカルタスクスケジューリング決定を行う役目を果たすことができる。タスクスケジューラ９０２は、それが制御する各仮想マシン上で何のタスクが実行すべきかを決定する。例えば、ユーザによってサブミットされるワークアイテムまたはジョブは、スケジューリングすべきタスクのリストを含むキューのセットを有することができる。タスクスケジューラ９０２は、キューのセットからタスクを取り、ジョブに関連するプール内の１つまたは複数の利用可能な仮想マシンを選択し、仮想マシン（複数可）と連絡を取ってこれらのタスクをスケジューリングする。タスクスケジューラ９０２はまた、ジョブに関連する優先順位値に基づいてスケジューリング決定も行うことができる。加えて、タスクスケジューラ９０２は、タスクテナント内部の仮想マシンを追跡する。

【００７０】

[0091]タスクスケジューラ９０２は、プールサーバと共に働き、プールに／から仮想マシンを割り振る／割振り解除する。加えて、タスクスケジューラ９０２は、すべての仮想マシンとの間でハートビートを維持し、ハートビートを介してプールメンバシップについて仮想マシンと同期し、仮想マシンの再始動／再イメージを制御する。タスクスケジューラ９０２のさらに別の機能は、タスクテナントのサイズを追跡することとすることができる。タスクテナント内の仮想マシンの現利用に基づいて、タスクスケジューラは、タスクテナントを増大または収縮することができ、その結果、タスクテナントは、タスクテナントに関連するタスクを実行するのに十分な数の仮想マシンを有する。同様に、タスクテナント内でアイドル状態にある仮想マシンが多過ぎる場合、データセンタ内の他のホストされるサービスで使用するためにマシンを解放することができる。

【００７１】

[0092]タスクスケジューラ９０２は、上記で参照した機能を達成するために以下の構成

要素およびモジュールからなる。例えば、タスクスケジューラは、要求プロセッサ９１０、ハートビートマネージャ９１２、統計アグリゲーションマネージャ９１４、スケジューリング・ループ・モジュール９１６、プール割振りマネージャ９１８、およびタスク・テナント・マネージャ９２０からなることがある。

【００７２】

[0093]例示的態様では、タスクスケジューラ９０２は、タスクテナント９００の仮想マシンを「所有し」、それらの仮想マシン上で実施されるものも「所有する」。その結果、ＶＭのプールが複数のタスクテナントにわたるＶＭからなることが企図される。ジョブがプールに割り当てられるとき、タスクスケジューラ９０２は、キューからワークアイテムを取り、タスクスケジューラ９０２が「所有する」仮想マシン（例えば、タスクテナント９００のＴＶＭ９０８）上でそれらのワークアイテムを実行することができる。プール内のＴＶＭを有する他のタスクテナントに関連する他のタスクスケジューラもキューからワークアイテムを取り、各タスクテナントの固有のタスクスケジューラのそれぞれに関連する仮想マシン上でそれらのワークアイテムを実行することができる。

【００７３】

[0094]要求プロセッサ９１０は、ＷＩＪ、プールサーバ、および／またはＴＬＳＭから受信した様々な要求を処理する役目を果たす。タスクテナントフロントエンドからタスクスケジューラに要求をルーティングすることができる。したがって、要求プロセッサ９１０は、正しいサブコンポーネントを求める要求をキューに入れ、消費することができる。

【００７４】

[0095]ハートビートマネージャ９１２は、タスクテナント内の他の仮想マシン（例えば、ＴＶＭ９０８）と共にハートビートする役目を果たす。ハートビート通信に応答して、このモジュールは、仮想マシンヘルスおよびタスクヘルスについての情報を収集する。加えて、受信したハートビートメッセージに応答して、このモジュールは、仮想マシンのうちのどれがアイドル状態であり、新しいタスクをスケジューリングするのに使用することができるかを判定することができる。加えて、ハートビートモニタは、仮想マシンについての様々な統計（例えば、ＣＰＵ、メモリ、ディスク使用）を収集することができる。次いで、これらの統計を統計アグリゲーションマネージャ９１４に渡すことができる。

【００７５】

[0096]統計アグリゲーションマネージャ９１４は、タスクテナントの様々な仮想マシンから収集される様々な統計をプールによって集約および編成する役目を果たす。次いで、自動スケーリングオペレーションで使用するために、集約された統計をプールサーバに通信することができる。

【００７６】

[0097]スケジューリング・ループ・モジュール９１６は、プール仮想マシン上のジョブタスクをスケジューリングする役目を果たす。プール割振りマネージャ９１８は、本明細書で論じるように、リソースをスケーリングすること、リソースを割り振ること、ジョブ／ワークアイテムを割り当てることなどのプールに関する操作を受け持つ。

【００７７】

[0098]タスク・テナント・マネージャ９２０は、タスクテナント自体の増大および縮小を受け持つモジュールである。このモジュールは、分散コンピューティングシステムと対話して、システム上の負荷に従ってタスクテナント仮想マシン数を拡大／縮小する。加えて、タスク・テナント・マネージャ９２０が、所与のプールに関する専用仮想マシンへの迅速な変換のために、フリー再イメージング化仮想マシンのバッファを維持する役目を果たすことができることが企図される。

【００７８】

[0099]図９はまた、タスクテナントに関連する複数の仮想マシンも示す。図９の示す実施形態では、仮想マシンのそれぞれがタスク仮想マシン９０８（ＴＶＭ）を含む。一実施形態では、タスク仮想マシン９０８は、仮想マシン上のタスクを立ち上げ、タスクに関するディレクトリ構造および許可をセットアップする役目を果たす。タスク仮想マシン９０

８はまた、同一のプール内の仮想マシン間のトラフィックのみを許可するように仮想マシン上のオペレーティングシステムファイアウォールを構成する（プールがイントラ通信を必要とする場合）。先に論じたように、タスクスケジューラ９０２は、タスク仮想マシン９０８を介して仮想マシンとの間でハートビートを維持する。これは、タスクスケジューラ９０２が、仮想マシンのヘルスを監視し、タスク仮想マシンエージェントに関するプールメンバシップ情報を同期することを可能にする。

【００７９】

[00100]例示的タスク仮想マシン９０６が、簡潔のためにＴＶＭ９０８には示されていないいくつかの構成要素からなるように示されている。しかし、任意のモジュール／構成要素を任意の仮想マシンに関連付けることができることが企図される。タスク仮想マシン９０６の構成要素／モジュールは、要求プロセッサ９２２、リソースマネージャ９２４、タスクマネージャ９２６、およびセキュリティマネージャ９２８を備える。

【００８０】

[00101]要求プロセッサ９２２は、仮想マシンがタスクスケジューラまたはタスクテナントフロントエンドから得る様々な要求を処理する役目を果たす。リソースマネージャ９２４は、ディスククォータを管理し、システムに関するディレクトリ構造を作成し、スタートアップタスクおよびジョブタスクに関するリソースをダウンロードする役目を果たす。タスクマネージャ９２６はタスク寿命時間を管理し、タスク寿命時間は、仮想マシンがタスクスケジューラから開始タスクコマンドを受信する時から始まり、システムデータ（例えば、関連ディレクトリ構造）が仮想マシンからクリーンアップされるまで有効である。セキュリティマネージャ９２８は、様々なファイアウォール規則をセットアップし、ユーザタスクを実行するための正しい特権でアカウントを作成する役目を果たす。

【００８１】

[00102]図１０に、本発明の態様による、例示的プールサーバ１０００を示すブロック図を示す。上記の例示的態様で論じた図６のプールサーバ６１４としてプールサーバ１０００を実装することができる。

【００８２】

[00103]プールサーバ１０００は以下の構成要素からなる。着信操作（例えば、プールの作成、プールの削除、およびプールの更新）を受諾する役目を果たす要求プロセッサモジュール１００２。プール内に仮想マシンを確保するためにタスクテナントにわたって仮想マシンブレークダウンを実施するタスクテナント仮想マシン割振りマネージャモジュール１００４。どのタスクテナントを選び、どれほどの仮想マシンをプールについてタスクテナント内に確保するかを選ぶのは、決定モジュールである。トランザクション・プロセッサ・モジュール１００６をプールサーバに関連付けることもできる。トランザクション・プロセッサ・モジュールは、プールトランザクションの寿命を維持するコアモジュールである。このモジュールは、トランザクションが首尾よく終了し、またはタイムアウトとなり、またはキャンセルされるまで、トランザクションに対して引き続き働くことができる。各トランザクションはテーブル内で存続し、したがって障害の場合に、様々なシステム構成要素にわたって完了することができる。例示的トランザクションは、所与のプールについてそのタスクテナントで一定の数のＶＭを割り振り、予約し、または割振り解除するための、プールサーバからタスクテナントへの要求を含むことができる。さらに、自動スケーリングモジュール１００８をプールサーバ１０００に関連付けることもできることが企図される。図６の自動スケーリング構成要素６１６に関連して先に論じたように、自動スケーリングモジュール１００８は、プールに関する所望の数の専用仮想マシン、スタンドバイ仮想マシン、および／またはプリエンプティブル仮想マシンを自動的に決定するユーザ提供の自動スケーリング規則（例えば、公式）を実行する役目を果たす。

ワークアイテム／ジョブ／タスクの管理

[00104]図１１に、本発明の態様による、例示的分散コンピューティングシステム内の様々な構成要素間の例示的ワークフロー１１００を与える通信ダイアグラムを示す。分散コンピューティングシステムは、クライアントポータル１１０２、タスク位置サービスフ

ロントエンド（ＴＬＳＦＥ）１１０４、プールサーバ１１０６、ワークアイテム／ジョブスケジューラ（ＷＩＪ）１１０８、ストレージユニット１１１２、タスクテナント１１１４からなる。構成要素のすべては先に論じられる。ＴＬＳＦＥ１１０４、プールサーバ１１０６、およびＷＩＪ１１０８は、タスク位置サービス１１１０（図６のＴＬＳ６００）に関連付けられる。

【００８３】

[00105]ワークフローは、ユーザがクライアントポータル１１０２を通じてＴＬＳＦＥ１１０４にワークアイテムをサブミットすること（１１１６）を含む。ＴＬＳＦＥへのＡＰＩ呼出しを使用して、このサブミットを行うことができる。次いでＴＬＳＦＥは、ＷＩＪ１１０８と連絡を取り、受信したワークアイテムをシステムに登録する（１１１８）。ワークアイテムの登録の間に、ワークアイテムタスクおよびオプション・ジョブ・マネージャ・タスクのために使用するプールを指定することができる。この例示的態様では、この登録の結果、ワークアイテムの持続１１２０のためにＷＩＪ１１０８からストレージ１１１２に通信することができる。さらに、ＷＩＪ１１０８が、ストレージ１１１２に格納される関連するジョブを作成する（１１２２）ことが企図される。

【００８４】

[00106]ジョブを自動プールに関連付けるべきであるので、ＷＩＪ１１０８は、ワークアイテムのタスクがそれに対して処理されるプールを作成するようにプールサーバ１１０６に命令することができる。このワークフローは、ワークアイテムが割り当てられるプールが既存ではないという仮定の下で動作している。この例では、ユーザによってサブミットされるワークアイテムは、自動プール実施形態であることを示すことができる。自動プール設定が含まれることを示すワークアイテムが受信されるとき、プールサーバ１１０６は、動的かつ自動的にプールを作成することができる（１１２４）。ＷＩＪは、プールが作成／削除されるときを制御することができ、ＷＩＪは、それに応じてプールサーバに命令することができる。次いで、プールサーバは、他のユーザがプール作成／削除要求を開始したのと同じく、ＷＩＪ要求を処理することができる。従来、プールの作成は、プールの作成を動的かつ自動的にするのを妨げるユーザ介入を必要とする。

【００８５】

[00107]ＷＩＪ１１０８によって生み出される各ジョブについて、複数のキューを形成することができる。例えば、ＷＩＪは、専用キュー、プリエンプティブルキュー、「任意のＶＭ」キュー、および／またはジョブマネージャキューを含むいくつかのジョブキューを作成することができる（１１２６）。これらの様々なキューは、ジョブに関するタスクを保持するのに使用される。

【００８６】

[00108]ワークアイテムが関連するジョブマスタタスクを有する場合、ＷＩＪは、ジョブで開始される第１のタスクとしてジョブ・マネージャ・タスクを有するように、ジョブ・マネージャ・タスクをシステムに登録することができる。加えて、ジョブマネージャキューに情報を追加することができる。次いで、システムは、システム障害の場合にジョブマネージャがジョブと共に常に実行中であることを保証することができる。それらの通信は、ＷＩＪがジョブマネージャを追加し（１１２８）、ジョブ・マネージャ・タスクを追加する（１１３０）ことによって示される。

【００８７】

[00109]ＷＩＪがジョブを生み出すと、ＡＰＩを介してジョブに関する追加のタスクをサブミットすることができる。システムは、それぞれのサブミットされたタスクについての再試行カウントを指定することをサポートする。タスクに障害が発生した場合、システムは、恐らくは種々の仮想マシン上で、再試行カウント回数までタスクを再び生み出すことができる。ジョブがＪＭタスクを有する場合、ジョブに関する追加のタスクをジョブ・マネージャ・タスク自体によってサブミットすることができる。システムは、ジョブが作成された後、システムの外部からタスクをサブミットすることを可能にすることができる。

【００８８】
[00110]ＴＬＳＦＥは、すべてのタスク追加要求を処理し、タスク追加要求の受信時に、ＴＬＳＦＥは、タスク情報をタスクテーブル内に存続させ、さらに、タスクについてのメタデータを３つのジョブキューのうちの１つの中に置く。どのジョブキューがどのタスクに依存することがあるかという選択が、実行のためにマークされる（例えば、専用のみ、プリエンプティブルのみ、またはプール内の任意のＶＭ）。
【００８９】
[00111]ジョブが生み出されると、ＷＩＪは、所与のプールに関する仮想マシンを有するすべてのタスクスケジューラと連絡を取り、ジョブ開始通信１１３２からわかるように、ジョブ開始について通知する。ユーザは、ワークアイテム作成中に、ジョブ・マネージャ・タスクが完了したときにシステムがジョブ全体を終了すべきかどうかを指定することができる。次いでシステムは、この情報を維持することができ、ＪＭタスクが完了したときが指定された場合、以下のロジックを実装することができる。ＷＩＪは、タスクテナント（複数可）１１１４がジョブキュー１１３４内のジョブを処理した後に行うことのできる、ＪＭタスク完了についての通信を待機することができる。ジョブ・マネージャ・タスクが完了したとき、メッセージがＪＭ完了キューにエンキュー(enqueue)される。例示的態様では、ＪＭキューはシステムキューである。ＴＬＳＦＥは、キューの監視を保つことができ、キュー内のメッセージを得たとき、メッセージをデキュー(dequeue)し、ジョブマネージャ完了を処理するように適切なＷＩＪに通知する。ジョブマネージャメッセージがキューから受信された後、ＷＩＪは、その永続的ストア(persistent store)で完了済みとジョブをマークし、ＪＭ完了キューからメッセージを除去することができる。
【００９０】
[00112]ジョブが関連するジョブ・マネージャ・タスクを有さない場合、またはユーザがジョブ・マネージャ・タスクの完了と共にジョブを終了することを指定しない場合、顧客による別々のジョブ終了要求を発行してジョブを完了済みとマークすることができる。ジョブが完了済みとマークされると、ジョブに関する追加のタスクを処理することはできない。さらに、ジョブを反復または１回実行とマークすることができることが企図される。反復ジョブについて、ＷＩＪは、次の反復間隔で新しいジョブを生み出す（例えば、インスタンス化する、作成する）ことができる。ワークアイテムは、「それ以後実行不可（ｄｏｎ　ｎｏｔ　ｒｕｎ　ａｆｔｅｒ）」時間を指定することができ、それは、その時の後にＷＩＪがジョブを再び生み出すことを排除する。ＷＩＪからのジョブ開始の通信の受信時に、タスクスケジューラは、タスクの存在についてジョブキュー内を調べることを開始する。キューに入れられる追加のタスクとして、例示的態様では、タスクスケジューラはタスクをデキューし、それを指定のプール仮想マシン上で実行する。
タスクスケジューラスケジューリング
[00113]タスクスケジューラ（「ＴＳ」）（図９のタスクスケジューラ９０２など）は、以下の例示的方式でタスクのスケジューリングを実施することができる。ＴＳは、メモリ内にスケジューリング情報の大部分を維持することができる。しかし、ジョブキュー（複数可）からまだピックアップされていないタスクに関する情報が、メモリ内に維持されないことがあることが企図される。
【００９１】
[00114]例示的態様では、ＴＳはプールによるスケジューリングを実施する。したがって、ＴＳは、ジョブタスクをスケジューリングすることのできる各プールについての情報を維持する。この情報を、ＷＩＪによってジョブ開始メッセージ（例えば、図１１のジョブ開始通信１１３２）を介してＴＳに渡すことができる。例示的態様では、ＴＳは、フェイルオーバシナリオの可能性を処理するためにこの情報を存続させることができる。各プールについて、複数のアクティブなジョブが存在することができる。ＴＳは、ジョブ優先順位に基づいて各プール内のスケジューリングを行う。しかし、例示的態様では、優先順位はオーバーアーチング・アカウント・レベル(overarching account level)ではなく、ジョブレベルである。

【００９２】
[00115]例示的態様では、各ＴＳは、割り当てられたジョブのリストを知っている。各スケジューリング量子(scheduling quantum)について、ＴＳは、様々なジョブキューから「Ｎ」個の項目を選び（例えば、専用タスクやプリエンプティブルタスクなどの種々のタイプのタスクをスケジューリングするための種々の優先順位を有する、ジョブ当たり複数のジョブキューが存在することがある）、各ジョブキューは優先順位キューであってもよい。この例では、「Ｎ」個の項目は、タスクをスケジューリングするためにプールで利用可能なＶＭ数に等しくてよい。言い換えれば、ＴＳがプールで利用可能な仮想マシン数に等しいいくつかの項目をキューから選ぶことが企図される。
【００９３】
[00116]タスクについての他のメタデータに加えて、各キューエントリは、メインタスクテーブルへのポインタを有することができ、それは、ＴＳがタスクテーブルからタスクについての情報を読み取ることを可能にする。タスクテーブルはタスク状態を有することができ、タスク状態は、タスクを実行する必要があるかどうかをＴＳが判定するのを助けることができる。例えば、タスクが完了した場合、タスクを再び実行する必要はない。これは、タスクが完了した後に障害があるが、ＶＭがジョブキューからエントリを除去することができなかったときに生じることがある。加えて、タスクに関するアフィニティ情報をタスクテーブルから読み取ることができることが企図される。このアフィニティは、キューから選ばれた「Ｎ」個の項目のうちのどれが特定の仮想マシンにとって最も適しているかをＴＳが判定することを可能にすることができる。
【００９４】
[00117]タスクがＶＭに割り当てられると、ＴＳは、タスクに対応するタスクテーブルに仮想マシン情報をポピュレートし、仮想マシン上でタスクをスケジュールすることができる。仮想マシンにタスクが割り当てられると、仮想マシンは、そのジョブキュー内の実行中タスクの不可視時間を延長することを引き継ぐことができ、それは、実質的にタスクに関するリースを維持する。ＶＭがタスクを受信すると、ＶＭは、タスクの状態を「実行中」（または同等のもの）に更新することができる。リソースがタスクについてダウンロードされると、ＶＭは、タスク状態を「実行中」（または同等のもの）に更新し、タスクプロセスを生み出すことができる。したがって、次いでＶＭは、ジョブキューから、タスクに関する対応するエントリを削除することができる。タスクがたまたまＪＭタスク（ジョブマスタタスク）であった場合、ＶＭは、ＪＭ完了キュー内にジョブマネージャ完了エントリを含むことができる。
タスク再試行処理およびジョブ・マネージャ・タスク再試行処理
[00118]概して、タスク再試行およびＪＭタスク再試行処理は同様に実施される。しかし、例示的態様では、プロセス内のいくつかの逸脱が存在する。例示的態様では、サブミットされる各タスクが、タスクと共に指定された再試行カウントに関連付けられる。タスクの実行に関するアプリケーション障害時に、システムは、タスクをエラーと共に完了とマークする前に、再試行カウント値に等しい回数だけタスクを再スケジューリングすることができる。システム障害（例えば、ＶＭ障害）の状況では、システムは、再試行カウントに対してカウントすることなく、障害の発生したタスクを自動的に再試行することができる。
【００９５】
[00119]仮想マシンは、生み出したタスクを監視する。タスクに障害が発生した場合、ＶＭは、ＶＭ上でタスクを再試行し、タスクテーブル内の再試行カウントも更新する。例示的態様では、ＶＭがタスクを「ｘ」回（これは構成可能である）再試行し、その数を超過した後、ＶＭは、タスクスケジューラが、キュー内でタスクを再び可視にする（例えば、タスクに関するリースを解放する）ことより、代替ＶＭでタスクを再スケジューリングすることを可能にする。次いで、タスクスケジューラは、キューから可視の項目をピックアップし、再試行カウントを更新し、それを別のＶＭに割り当てることができる。この例では、タスクは、利用可能なＶＭがある場合にピックアップされるだけでもよい。このプ

ロセスは、タスクが首尾よく完了するまで、またはタスクが指定の回数を再試行するまで続行することができる。

【００９６】

[00120]ＪＭタスクは、上記で略述したのと同様にプロセスをたどることができる。しかし、ＶＭがＪＭタスクを生み出すのに利用可能ではない場合、タスクスケジューラが非ＪＭタスクのうちの１つ（例えば、実行時に最低の優先順位）をピックアップし、ＪＭタスクが実行するためにリソースを利用可能にするようにタスクを占有できることが企図される。これは、システム障害の場合であっても、ジョブについてＪＭタスクが常に再スタートされることを保証することができる。

プール管理

[00121]各タスクアカウントについて、１つまたは複数のプールを作成する（または関連付ける）ことができる。例示的態様では、（先に論じたように）プール内に２つのタイプの仮想マシンがあり、プールに関連するスタンドバイＶＭ予約もある。ＶＭは、タスクアカウントまたはプールによる排他的利益のために現在使用されている予約済みＶＭである専用ＶＭでよい。第２のＶＭはプリエンプティブルＶＭでよい。プリエンプティブルＶＭは、スタンドバイＶＭとして予約されるＶＭ、またはシステムがプロビジョニングした任意のアイドルＶＭを含む、プリエンプティブルタスクを実行するのに利用可能であるシステム内の任意の非専用ＶＭでよい。システムが別の目的のためにそのリソースを必要とするまで、プリエンプティブルＶＭを使用することができる。

【００９７】

[00122]例示的態様では、アカウントは、プール内の専用ＶＭおよびスタンドバイ予約済みＶＭのそれぞれの数を指定することができる。スタンドバイＶＭ予約を専用ＶＭに変換することができ、逆も企図される。次いで、システムは、特定のプールのために専用およびスタンドバイリソースを保持することができる。

【００９８】

[00123]様々なプロパティをプールに関連付けることができる。それらのプロパティは、スタンドバイＶＭ予約、専用ＶＭ数、ＶＭのタイプ（例えば、小または大）、通信要件（ジョブがクロスタスク通信を必要とする）、ストレージ・アカウント・アフィニティ、プールメタデータ（例えば、ユーザによって設定されたメタデータ）、および／または開始タスク情報を含むことができる。開始タスク情報は、プールの初期セットアップ中、さらにはＶＭがリブートするときにプールＶＭのそれぞれの上で実行されるタスクである。これらのプロパティは、少なくとも部分的に、プールおよびプールのリソースを定義することができる。

【００９９】

[00124]先に論じたように、ジョブは、プール（および下に仮想マシン）を利用して、ジョブのタスクを実行する。タスクは、プール内の専用ＶＭまたはプリエンプティブルＶＭのどちらかの上で実行する。例示的実施形態では、スタンドバイＶＭ予約は、タスクを直接的に実行するのに使用されず、その代わりにスタンドバイＶＭが、専用ＶＭまたはプリエンプティブルＶＭに変換される。様々なＶＭに関する価格決定が異なることがあることが企図される。例えば、スタンドバイＶＭ予約は、リソースを予約するためのコストの点で最小であることがあるが、予約することにより、専用またはプリエンプティブルリソースに迅速に変換するのにスタンドバイＶＭ予約が利用可能である。専用ＶＭは従来の計算価格決定を有することができる。一方、スポット価格決定および他の可変価格決定構成を可能にする入札操作によってプリエンプティブルＶＭを価格決定することができる。

【０１００】

[00125]プールの作成は、以下の例示的ステップと同様のプロセスをたどることができる。しかし、代替方法が企図される。ユーザは、プール名、ＶＭサイズ、ストレージ・アカウント・アフィニティ、クロストーク通信要件、メタデータなどの様々なパラメータを指定することによってプールの作成を開始することができる。ユーザは、ＴＬＳＦＥにＡＰＩ要求を送ることができ、次いでＴＬＳＦＥは、関連する要求（または同一の要求）を

正しいプールサーバに転送することができる。プールサーバに要求を送る前に、システムはユーザを認証することができる。次いで、受信側プールサーバは、一意識別子を有するプールトランザクションを開始することができる。次いで、プールサーバは、専用ＶＭの目標数およびスタンドバイとしての予約を保つＶＭ数を満たすように、タスクテナントに関するＶＭ予約ブレークダウンを生成することができる。この予約操作は、プール予約が最終的に存在することができるように、システム障害の場合にトランザクションを存続させることを含むことができる。次いで、プールサーバは、ステータス（すなわち、ハートビート）更新を定期的に通信し、予約に関する更新を保つようにタスクスケジューラを用いてチェックすることができる。トランザクションの完了時に、プールサーバは、各タスクテナントにそのトランザクションをコミットするように通知し、次いで各タスクテナント応答の後に、保留トランザクションを除去することによってトランザクションをコミットすることができる。

【０１０１】

[00126]スタンドバイＶＭ予約を専用に変換する（または逆も同様である）ことによってプールを更新することができる。システム（またはユーザ）が所望の数の専用ＶＭおよびスタンドバイＶＭを与えることにより、これを達成することができる。次いで、プールサーバは、関連するタスクテナントと共に新しいトランザクションを開始し、新しいＶＭターゲットをそれらのタスクテナントに渡すことによってプールの更新を処理することができる。タスクテナントで、ＴＴＦＥを通じてタスクスケジューラに着信要求をルーティングすることができる。所望のターゲットが専用ＶＭの数を増加させること、および余分のスタンドバイＶＭを予約することを含む場合、タスクスケジューラは、割り振られていないＶＭを専用ＶＭに変換し、スタンドバイカウントを低減する。追加の専用が依然として望まれる場合、タスクスケジューラは、プリエンプティブルプールからＶＭを割り振る。利用可能なＶＭが不十分である場合、タスクテナントが、要求を満たすように増大することができる。言い換えれば、フリーＶＭが当初割り振られ、その後にプリエンプティブルＶＭを変換することが続くことが企図される。しかし、プリエンプティブルＶＭが当初変換され、残りのリソースの必要があればそれを満たすようにフリーＶＭが割り振られることも企図される。

Ｐｌａｔｆｏｒｍ　Ａｓ　Ａ　Ｓｅｒｖｉｃｅ（ＰＡＡＳ）を用いたジョブスケジューリング

[00127]図１２に、本発明の態様による、分散コンピューティング環境内のリソースを提供する例示的方法例示的方法１２００を示すブロック図を示す。ブロック１２０２で、ユーザからタスクアカウントを作成する要求が受信される。例えば、ユーザは、クライアントポータル（サービス管理ＡＰＩを含む）により、計算を実施するためにユーザによって使用可能なアカウントをサービスが生成することを要求することができる。システムの高レベル位置サービス（例えば、ＸＬＳ）で要求を受信することができる。図示されていないが、コンピューティングリソースの１つまたは複数の定義済みのクラスタ（例えば、地理領域）内のより低いレベルのタスク位置サービスと通信することによってアカウントが作成されることを高レベルサービスが保証することが企図される。さらに、先に論じたように、高レベル位置サービスがアカウントまたはアカウントによって使用されるリソースに関連する名前空間の登録を調整することが企図される。少なくとも部分的にドメインネームサービスを利用して、名前空間の登録を実施することができる。

【０１０２】

[00128]ブロック１２０４で、タスクアカウントに関連するワークアイテムが受信される。例えば、ユーザは、クライアントポータルによってワークアイテムをサブミットすることができる。ワークアイテムは、ユーザによってシステムのタスク位置サービスに通信されるものとして企図される。例示的態様では、ワークアイテムは、システムによって維持される登録された名前空間に少なくとも部分的に基づいてＴＬＳにルーティングされる。例えば、先に論じたように、アカウント仮想ＩＰアドレスおよび／またはタスク仮想ＩＰアドレスを利用して、様々な通信を通信することができる。

【０１０３】
[00129]他の構成要素の間で、ワークアイテムが、１つまたは複数の仮想マシンによって実行すべきバイナリ、仮想マシンによって使用することのできるコマンドラインパラメータ、および／または規則からなることがあることが企図される。規則は、例えばジョブを自動的に作成し、プールを自動的に作成し、プール上のジョブを自動的に実行し、かつ／またはプールを自動的に削除するためにシステムによって使用可能であることがある。規則によって他の機能を達成することができる。ブロック１２０６で、システムが、ワークアイテムに関連する規則を受信した。先に論じたように、ユーザによって規則を通信することができ、またはユーザ提供の情報（例えば、タスクアカウントに関連するプリファレンス）に基づいて、システムによって少なくとも部分的に生成することができる。規則は、ワークアイテム、得られるジョブ、関連するタスク、および／またはプールの特徴を定義する、より広い仕様の一部でよい。

【０１０４】
[00130]ブロック１２０８で、ジョブが、ワークアイテムに基づいて自動的に作成される。先に論じたように、システムのＷＩＪによってジョブを作成することができる。例示的態様では、ワークアイテムに関連して受信された規則／仕様に基づいてジョブが作成される。この規則／仕様と共に含まれる他の情報の間で、ユーザに対する負担なしに、プール（またはプールのセット）に関する最終的インスタンス化に関するジョブをシステムが自動的に作成することを可能にする優先順位および他の計算変更情報を含めることができることが企図される。

【０１０５】
[00131]ブロック１２１０で、ワークアイテムで自動プールが指定されるとき、ジョブを処理する際に使用するために、仮想マシンがプールに自動的に割り振られる。先に論じたように、プールに割り振られる仮想マシンは、ワークアイテムに関連する仕様および／または規則に基づくことができる。例えば、ワークアイテムに関連する情報は、所望のリソース消費、ワークアイテムを完了することが望ましい時間、タスクを計算するための地理的領域などを指定することができる。この情報を利用して、プールサーバは、０個以上の仮想マシンをプールに割り振ることができる。割振りのために使用される情報は、専用ＶＭおよびスタンドバイＶＭの数などを指定する仕様の一部でよい。

【０１０６】
[00132]ブロック１２１２で、ＷＩＪによってジョブキュー（または、先に論じたように複数のジョブキュー）を自動的に作成することができる。さらに、例示的態様では、ＪＭタスクを作成できることが企図される。加えて、ブロック１２１４で示されるように、プールにジョブを割り当て、次いでプールのＶＭ上でジョブのタスクをスケジューリングすることによってプール上のジョブとしてワークアイテムをインスタンス化できることが企図される。したがって、ワークアイテムをジョブとしてプール上でスケジューリングできることが企図される。次いで、ジョブをインスタンス化することの部分として、スケジューリングされたジョブのタスクを、実行すべきＶＭに割り当てることができる。先に論じたように、タスクテナントタのスクスケジューラが、プール内の仮想マシンのうちの１つまたは複数でキュー内の１つまたは複数のタスクをスケジューリングする役目を果たすことができることが企図される。第２のジョブがプール上でインスタンス化されることも企図される。第２のジョブは、完全に異なるワークアイテムに基づくワークアイテムまたはジョブの反復インスタンスでよい。

【０１０７】
[00133]ブロック１２１６で、ユーザの介入または要求なしに、システムがプールを自動的に削除することが企図される。ジョブまたはワークアイテムの完了に応答して、プールを削除することができる。例えば、各ジョブが完了した後にプールを削除することができる。加えて、アカウント（または別のアカウント）またはワークアイテムによる将来の使用のために、ジョブの完了後にプールを維持できることが企図される。例えば、ワークアイテムが反復のためにスケジューリングされる場合、ジョブの再生み出しを予期してプ

ールを維持し、ランニングジョブにわたってＶＭ上の状態（例えば、データファイルおよびアプリケーション）を保つことが効率的であることがある。さらに、自動スケーリング機能を使用して、プールは専用ＶＭをスタンドバイＶＭに自動的に変換し、またはプール上の保留のタスク数などのスケーリング規則に応答して、ＶＭ数を低減できることが企図される。さらに、ユーザからの要求を受信することができることが企図される。要求は、プールの削除を求める命令を含むことができる。

プールおよび階層的構造を自動的にスケーリングする

[00134]図１３に、本発明の態様による、分散コンピューティング環境内のリソースを提供する例示的方法１３００を示すブロック図を示す。具体的には、規則、要件、および現リソース負荷に基づいてリソースのプールを自動的にスケーリングすることが、分散コンピューティング環境内のリソースの効果的なプロビジョニグを可能にすることが企図される。先に論じたように、システム（例えば、分散コンピューティング環境）により、ジョブがその上でインスタンス化されるリソースのセットとしてプールを作成することができる。しかし、リソース需要の変化、スケジューリング変化、および他の変数に基づいて、サイズを増加させ、サイズを低減し、特定のＶＭタイプを増加／低減し、プールに関連する他の変数を調節することが望ましいことがある。従来、このプロセスは、手動の性質であることがあり、プール・リソース・レベルを手動で調節するために人間の介入に依拠することがある。

【０１０８】

[00135]システムのリソースを効果的に利用するために、プールなどのリソースのスケーリングに自動化を提供できることが企図される。例示的態様では、方法１３００は、ユーザの介入なしにプールの自動スケーリングを可能にする。ブロック１３０２で、システムは、プールに関連する仮想マシン数を自動的に決定した。決定は、プールサーバによって完了することができる。数の決定は、仮想マシンの総数を決定すること、または特定のタイプの仮想マシンの総数を決定することを含むことができる。例えば、決定は、プール内の専用ＶＭの総数を決定することでよい。決定は、システムの１つまたは複数の構成要素／モジュールによって維持されるテーブルまたは他のリスティングから行うことができる。例えば、プールサーバが、プールによって使用される仮想マシンを有するタスクテナントのリスティングを維持することができることが企図される。同様に、タスクテナントがプールによって使用されるＶＭのリスティングを維持することができる（例えば、タスクスケジューラがそのような情報を維持することができる）ことが企図される。したがって、システム内で既に維持されている情報の１つまたは複数のソースを参照することによって決定を行うことができる。

【０１０９】

[00136]ブロック１３０４で、自動スケーリング規則が適用され、プールに関連する仮想マシン数が調節される。例えば、合計のＶＭ数が低減または増加されることが企図される。さらに、特定のタイプのＶＭ（例えば、専用）数が増加または低減されることが企図される。先に論じたように、自動スケーリング規則は、プールのサイズ／構成を決定うる際の助けとなるように１つまたは複数のプリファレンスを含むことができる。例えば、自動スケーリング規則は、現プール統計およびスケーリングされたジョブに基づいて最適化されたプールを求めるための１つまたは複数の公式を含むことができる。自動スケーリング規則は、ジョブキュー統計（例えば、実行することを待つ保留タスク、エンキューレート、デキューレート、タスク完了レートなど）、スポット価格決定情報、利用可能なリソース、リソースの効率などのシステムに関するメトリックを考慮に入れることができる。さらに、自動スケーリング規則は、ワークアイテム、ジョブ、および／またはタスクに関する所望の完了時間も考慮に入れることができることが企図される。加えて、自動スケーリング規則が、支出することが好ましい所望の金融リソース（例えば、専用ＶＭよりも低いコストのプリエンプティブルＶＭに依拠する）を考慮に入れることができることが企図される。

【０１１０】

[00137]プールに関連する仮想マシン数を調節するために自動スケーリング規則を適用した結果として、ブロック１３０６で示されるように、システムは、プールの１つまたは複数の仮想マシンを動的に割り振ることができる。ＶＭの動的割振りは、１つまたは複数のＶＭを追加すること、１つまたは複数のＶＭを除去すること、１つまたは複数のＶＭのタイプを変更することを含むことができる。例えば、自動スケーリング規則の適用の結果、専用ＶＭである１つまたは複数の仮想マシンをプールに追加することができる。さらに、割振りは、フリーまたはプリエンプティブルＶＭを専用ＶＭに変換すること、およびスタンドバイカウントを低減することを含むことができることが企図される。他の割振りも企図される。

【０１１１】

[00138]割振りの動的性質が、ユーザ介入なしに割振りを実施するシステムに関係付けられる。例えば、自動スケーリング規則を間隔（例えば、時間間隔、プロセスカウント間隔）で適用できることが企図される。自動スケーリング規則を自動的に実行した結果として、この例示的態様では、リソースの割振りが、割振りの時間にユーザの要求なしに、または割振りを実施することを要求するユーザ入力の直接的結果として行うことができる。

【０１１２】

[00139]図１４に、本発明の態様による、分散コンピューティング環境内のリソースを提供する方法１４００を示すブロック図を示す。図１３に関連して先に論じたように、プールに適用されるときの自動スケーリング規則の利用および適用が、分散コンピューティング環境内のリソースの提供を実現することが企図される。

【０１１３】

[00140]ブロック１４０２で、ワークアイテムがユーザから受信される。ユーザは、クライアントポータルを介して来ることがあるＡＰＩを通じて、システムの高レベル位置サービスにワークアイテムを通信することができる。ワークアイテムを特定のタスクアカウントに関連付けることができ、特定のタスクアカウントは、それに関連するアカウントプリファレンスを有することができる。ブロック１４０４で、自動スケーリング規則がユーザから受信される。自動スケーリング規則は、ワークアイテムの完了のための時間に関するユーザプリファレンス、ワークアイテムの優先順位、好ましい金融リソース支出（例えば、専用ＶＭよりもプリエンプティブルＶＭを好む）を与えることができる。自動スケーリング規則はまた、プールに関する需要に応答してプールをスケーリングする助けとするためにプールに関連する１つまたは複数のメトリックを利用する１つまたは複数の公式も含むことができる。

【０１１４】

[00141]ブロック１４０６で、第１のデータセンタ内に位置する第１のＶＭが、プールに割り振られる。仮想マシンの割振りは、プール内で使用可能なリソースとしての仮想マシンの追加を含むことができる。例示的態様では、第１のＶＭが、第１のプールに自動的に割り振られる。このことは、どのＶＭ、およびどのプールかは言うまでもなく、ＶＭがプールに含まれるべきであることをユーザ介入が示すことなく行われる。例えば、過去には、ユーザは、プールに含まれるべきリソースを識別することによってプールを確立することが必要であったことがある。リソースの識別は、ＶＭ数、およびＶＭがどこから割り当てられるかを識別することを含むことができる。この例では、システム自体は、プールを作成すべきであること、およびプールが所望の性能または動作特性を満たすためにいくつかのＶＭをプールに割り振るべきであることを識別する。

【０１１５】

[00142]ブロック１４０８で、第２の仮想マシンが第１のプールに割り振られる。第２のＶＭは、第１のデータセンタとは地理的に別々である第２のデータセンタ内にある。例示的態様では、第２のＶＭは、異なるタスクテナント内ではなく、同一のデータセンタ内のＶＭリソースのセットからのものでもよい。第２のＶＭがユーザによる介入なしに第１のプールに自動的に割り振られることが企図される。この自動化（およびユーザの介入なしにシステムによって実施される他の自動タスク）が、ユーザが介入することを許される

ときに普通なら利用されないプロセスおよびステップを組み込む。例えば、第１の（または任意の）プールにＶＭを割り振るべきことに関する決定が、本発明の例示的実施形態内で企図される。

【０１１６】

[00143]第１のタスクテナントおよび第２のタスクテナントは、システムにサービスする物理的に独立したデータセンタ（または共通データセンタ内）でよい。例えば、第１のタスクテナントを第１の地理的位置（例えば、特定の住所、都市、州、領域、国、および／または大陸）内に配置できることが企図される。一例では、第１のタスクテナントが米国の第１の領域（例えば、米国西部）内に配置され、第２のタスクテナントが米国の異なる領域（例えば、米国東部）内に配置されることが企図される。この例では、第１のタスクテナントおよび第２のタスクテナントが、共通の、または異なるタスク位置サービスによってアドレス指定可能である。これは、様々なデータセンタに及ぶプールを増大させる（またはプールを低減する）複雑さへのユーザの介入なしに、プールの自動スケーリングを可能にする。例えば、物理的制限が以前には物理的データセンタのサイズを超えるプールの拡張を妨げていたことがあることが企図される。ジョブおよびプールの概念を、ローカルＶＭリソースを制御するスケジューラ（タスクテナント）から分離することは、２つ以上のデータセンタ内のリソースを含むようにプールおよびそのジョブを拡張することを、そのような拡張を可能にするようにユーザがプログラムし、あるいはシステムに介入することを必要とすることなく可能にする。したがって、システムがプールのスケーリングを自動的に扱うことに基づいて、数十億のＶＭにわたって、かつ少数のＶＭにわたってスケーリングすることのできるワークアイテムをユーザが設計できることが企図される。

【０１１７】

[00144]ブロック１４１０で、ワークアイテムが、第１の仮想マシンおよび第２の仮想マシン上でインスタンス化される。ワークアイテムのインスタンス化は、キューからタスクを取り、それをＶＭに割り当てる、タスクテナント内のスケジューラを含むことができる。ワークアイテムのインスタンス化は、仮想マシン上のワークアイテムからのタスクのスケジューリングをさらに含むことができる。インスタンス化はまた、仮想マシンがタスクの処理のためにキューからタスクを取ることも含むことができる。インスタンス化はまた、仮想マシンでのタスクの処理も含むことができる。したがって、ワークアイテムのインスタンス化は、結果としてワークアイテムの一部（例えば、タスク）がＶＭによって処理される任意の態様を含むことができる。

【０１１８】

[00145]第１のＶＭと第２のＶＭを共に有するプールに対してワークアイテムのジョブをインスタンス化することにより、この結果を実施するためにユーザがワークアイテムを変更することなく、複数のデータセンタにわたってスケーリングするプールによってワークアイテムが処理される。したがって、単一のタスクテナントによってサービスされるプール上で動作することができる同一のワークアイテムは、ユーザ介入なしに、複数の別個のタスクテナントおよびデータセンタにわたるプール上で実行することもできることが企図される。

【０１１９】

[00146]ブロック１４１２で、自動スケーリング規則がプールに適用される。例示的態様では、システムは、ユーザ介入なしに、自動スケーリング機能を自動的に開始する。自動スケーリング規則は、現在使用中のリソース、コミットされるようにスケジューリングされたリソース、およびプールに対して必要なリソース、プール、タスク、ジョブなどに割り当てられたジョブに関するキュー内の保留作業の量を考慮に入れることができる。それらを一緒にして、自動スケーリング規則は、プールがＶＭタイプを拡張、収縮、および変更することを可能にする。

【０１２０】

[00147]例えば、追加のタスクがジョブに追加される場合、またはジョブを処理するための時間が初期予測を越えている場合、自動スケーリング規則を利用して、ジョブを完了

するのにどれほどの追加のリソースが必要となるかを判定できることが企図される。同様に、自動スケーリング規則の呼出しの結果、プール内のＶＭ数が過剰である可能性があること、およびそれらのリソースの一部を変換またはドロップすることができることが企図される。例えば、自動スケーリング規則に基づいて、専用ＶＭをプリエンプティブルＶＭ予約、さらにはスタンドバイＶＭ予約に変換することができる。さらに、自動スケーリング規則による決定が適用される結果として、プールからＶＭを完全に解放することができることが企図される。

【０１２１】

[00148]ブロック１４１４で、プールに関連する１つまたは複数のＶＭの再割振りが行われる。例えば、自動スケーリング規則がプールから専用ＶＭを解放することができると判定した場合、専用ＶＭをプールから解放することができる。同様に、自動スケーリング規則が、利用可能なＶＭが所望の結果（例えば、ジョブの適時の完了）を達成するのに十分ではないと判定した場合、キュー内、またはまだキューに入れられていない未解決のタスクを完了する際に使用するために、１つまたは複数の追加のＶＭをプールに割り振ることができる。さらに、先に論じたように、例示的態様では、１つまたは複数のＶＭの再割振りが、第１のタイプから第２のタイプにＶＭを変換することを含むことができることが企図される。例示的態様では、この再割振りは、プールに関連するプールサーバによって少なくとも部分的に実施されるものとして企図される。

分離されたリソース、スケジューリング、およびジョブ

[00149]リソース、スケジューリング、およびジョブの分離は、ジョブがリソースのあるプールからリソースの別のプールに実行を続行することを可能にする。例えば、種々の計算リソースにわたって作業を移行し、作業をロードバランシングするとき、この機能を使用することができる。特定の例では、特定のデータセンタに障害（例えば、自然災害）が発生した場合、ジョブの完了のために作業を新しいデータセンタに移行することができる。さらに、リソース、スケジューリング、およびジョブの分離は、ジョブがリソースおよびスケジュールのいくつかのプールにわたって実行することを可能にし、それは、ジョブが普通なら得られない高いレベルのスケーラビリティを達成することを可能にすることができることが企図される。プールが複数のスケジューラ、タスク、テナント、および／またはデータセンタに及び、それは、プールに割り当てられたジョブもそれらのリソースにも及ぶことを可能にすることが企図される。

【０１２２】

[00150]例示的態様では、分離は、システムでこれらの別々の概念を利用することを企図する。第１の概念は、仮想マシンのプールの概念に基づく。第２の概念は、ジョブ管理、ジョブ状態、およびジョブキューの周りに形成される。第３の概念は、スケジューラ（例えば、図９のタスクスケジューラ９０２）と、これらのプールに対する割り当てられたジョブをスケジューリングする役目を果たすＶＭとを含む。これらの概念は、マルチテナンシ(multi-tenancy)分散コンピューティングシステムのロードバランシングの必要に基づくジョブ、スケジューラ、およびプールの柔軟な再割当てを可能にする。さらに、これらの概念は災害回復も可能にすることが企図される。

【０１２３】

[00151]図１５に、本発明の態様による、分散コンピューティング環境内の分離されたリソース、スケジューリング、およびジョブを提供する方法１５００を示すブロック図を示す。ブロック１５０２で、ワークアイテムがシステムで受信される。例えば、ワークアイテムをサブミットするアカウントに関連するタスク仮想ＩＰアドレスを利用して、タスク位置サービスでワークアイテムを受信することができる。ブロック１５０４で示すように、ワークアイテムからジョブを作成することができる。例示的実施形態では、ジョブ作成は、ワークアイテムをジョブとして処理すべきであるという識別である。

【０１２４】

[00152]ブロック１５０６で、仮想マシンが第１のプールに割り振られる。例えば、プールサーバがＶＭを第１のプールに割り振り、ジョブをその上でインスタンス化（処理）

できるプールを提供することができることが企図される。ブロック１５０８で、ジョブが第１のプールに割り当てられる。例えば、第１のプールに関連するプールサーバが第１のタスクテナントスケジューラによって所有されるＶＭを割り振っており、したがってジョブがプールに割り当てられると、次いで第１のスケジューラがジョブキューからタスクを取り、それらをそのスケジューラによって所有されるそのプール内のＶＭに割り当て、タスクを実行することができることが企図される。

【０１２５】

[00153]ブロック１５１０で、ジョブのタスクが、第１のプールにスケジューリングされる。先に論じたように、プールにタスクをスケジューリングすることは、第１のタスクテナントスケジューラがプールに関連する１つまたは複数のＶＭ上でジョブの１つまたは複数のタスクを処理することを含むことができる。さらに、ジョブキューからタスクをリースするプロセスはタスクをインスタンス化することの部分であることが企図される。スケジューラがＶＭを「所有」し、ＶＭ上で実行されるプロセスも「所有」することが企図される。

【０１２６】

[00154]ブロック１５１２で、ジョブを第２のプールに再割当てすべきであるという決定が行われる。第２のプールは、第１のプールとは動作可能に独立することができ、したがって、障害となる地理的に限定された災害（例えば、竜巻、ハリケーン、地震、電力グリッド障害、ネットワーク障害）は第２のプールに直接的に影響を及ぼさない。決定はまた、２つ以上のプール、タスクテナント、またはデータセンタにわたるリソース負荷の平衡を取るように動作可能な自動バランシングプロセスに応答するものでよい。例えば、共通の位置（例えば、ニューヨーク州ニューヨーク）に基づく一定のアカウントが、共通の時間（例えば、金融取引日の開始）にリソースを利用することができる。この例では、米国東部地理領域内に集中するリソースのプールに、米国西部の地理領域に位置するリソースよりも多くの負担を負わせることができる。したがって、ブロック１５１４で示されるように、待ちおよび他の要素（例えば、アフィニティ）をさらに考慮して、ロードバランシングプロセスは、ジョブの１つまたは複数の部分を第２のプールに移行することがより効率的であると決定することができる。移行は「所有権」の移転に限定されず、複数のプールへのジョブの割当てである、複数のプールにわたるロードバランシングを単に企図するにすぎない。さらに、移行の概念は、種々のタスクテナントにわたってプールを拡張することを含む。その結果、プールが複数のタスクテナントをカバーするときであっても、ジョブが単一のプールによって実施されているという印象をユーザが持つ可能性がある。

【０１２７】

[00155]ジョブ（またはジョブ内のタスク）の移行は、キュー内のタスク上のリースの解放を含むことができ、したがって、異なるプール内のリソースはそのタスクを得ることができる。代替実施形態では、ジョブの移行は、新しいタスクサービス位置のリソースに関連して再作成およびスケジューリングのために新しいタスク位置サービスにワークアイテムを再分配することを伴うことが企図される。ジョブの移行は、第１のプールから第２のプールにジョブを再割当てする形態である。ブロック１５１６で、第２のプールでのジョブの割り当てが示されている。ブロック１５１８で、ジョブが第２のプール上で実行されるようにスケジューリングされる。第１のプールのスケジューラは、第１のプールに関連するＷＩＪ、タスクテナント、およびプールサーバでよく、第２のプールのスケジューラは、第２のプールに関連する異なるＷＩＪ、タスクテナント、およびプールサーバでよい。したがって、ワークアイテム／ジョブを単一のプールまたは単一のスケジューラに結びつけないことにより、第１のプールに関連するリソースに障害が発生した場合であっても、ワークアイテムを異なるプールに移行することができる。

【０１２８】

[00156]さらに、上記で図９に関連して先に論じたように、タスクテナントのタスクスケジューラが、システムのプールおよびワークアイテム／ジョブから分離されるスケジューラであることが企図される。例えば、タスクスケジューラは、プールに割り当てられた

キューからタスクを選択し、タスクスケジューラによってスケジューリングされるタスクテナント内の関連するＶＭのうちのどれがタスクを実行するかを制御することができる。この概念は、プールが各タスクテナントに関連する各スケジューラについての複数のタスクテナントにわたるとき、タスクスケジューラのそれぞれによって所有されるＶＭによって実行するためにキューからタスクを取ることを可能にする。したがって、スケジューラ（例えば、タスクスケジューラ）から、リソース（例えば、ＶＭ）からのジョブ（例えば、ワークアイテム）を実質的に分離する方式で、プールにわたるリソース上でワークアイテムを実行することができる。

【０１２９】

[00157]次いで、システムでのロードバランシングを様々なレベルで実施することができる。例えば、タスクテナント（例えば、共通ＴＬＳによってサービスされる共通の地理領域内の複数のタスクテナント）と協調するプールサーバによってスケジューリングされるリソースの共通のグルーピング内でロードバランシングを行うことができる。共通のグルーピングは、タスクテナントのグルーピングでよい。したがって、例示的態様では、２つの異なるレベルでロードバランシングを実施できることが企図される。この第１のレベルのロードバランシングは、共通のプールサーバに関連するタスクテナントの間で行うことができる。この第１のレベルのロードバランシングでは、プールサーバは、種々のタスクテナントにわたってＶＭを割り振ることができる。第２のより高いレベルのロードバランシングは、異なるＴＬＳ（例えば、図５のＴＬＳ５１２）にわたって行うことができる。このレベルのロードバランシングでは、負荷を種々のプールサーバにわたって拡散することもできる。このタイプのロードバランシングを実装して、その結果、システムにわたって負荷を移行することができる。企図されるさらに別のレベルのロードバランシングが、システム内の複数のプールにわたってワークアイテム／ジョブを割り当てるＷＩＪによって実施される。

【０１３０】

[00158]先に論じたように、ジョブの移行が、完全なジョブの移行、まだ処理されていないジョブの一部の移行、および／または元のプールでまだ処理されていない部分の一部を維持しながら、まだ処理されていないジョブの一部の移行を含むことができることが企図される。したがって、ジョブの移行は、障害の場合と、ロードバランシング操作に応答することの両方で有用である。

【０１３１】

[00159]本明細書で与えられるように、様々なモジュール、構成要素、システム、層、およびプロセスが論じられる。第１の形態から第２の使用可能な形態に情報を変換する固有のマシンとして任意の１つを実施できることが企図される。この変換は、第１の入力を第２の使用可能な出力に変換するように機能するプロセッサおよびメモリによって実施することができる。さらに、仮想マシンが本明細書で論じられる。先に論じたように、仮想マシンが有形のプロセッサおよびメモリでタスクを用いて処理し、その結果、分散コンピューティング環境の利点のためにシステムによって使用可能である第２の形態にタスクが変換されるように機能することが企図される。

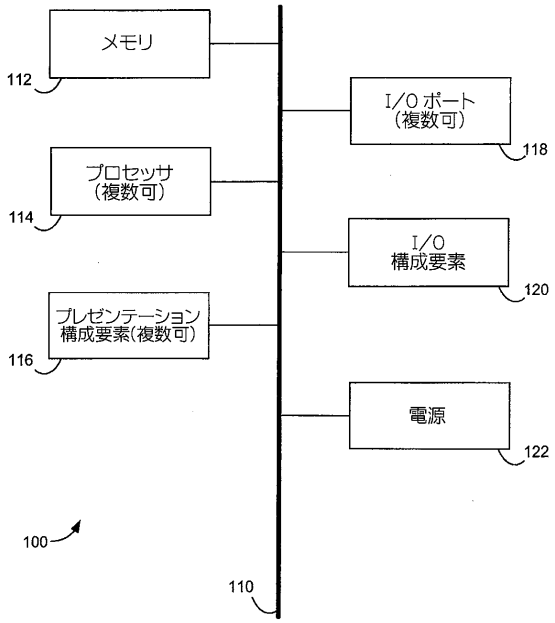
【０１３２】

[00160]上記より、本発明が、明白であり、構造に固有のである他の利点と共に上記で述べたすべての目的および目標を達成するように良く適合されたものであることがわかるであろう。

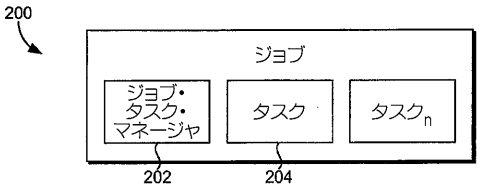
【０１３３】

[00161]いくつかの特徴および部分的組合せは有用であり、他の特徴および部分的組合せを参照することなく利用できることを理解されよう。これは、特許請求の範囲によって企図され、特許請求の範囲内にある。

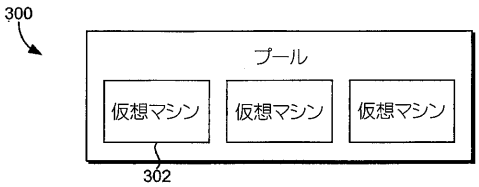
【図１】


メモリ
112
I/O ポート
（複数可）
118
プロセッサ
（複数可）
114
I/O
構成要素
120
プレゼンテーション
構成要素（複数可）
116
電源
122
100
110

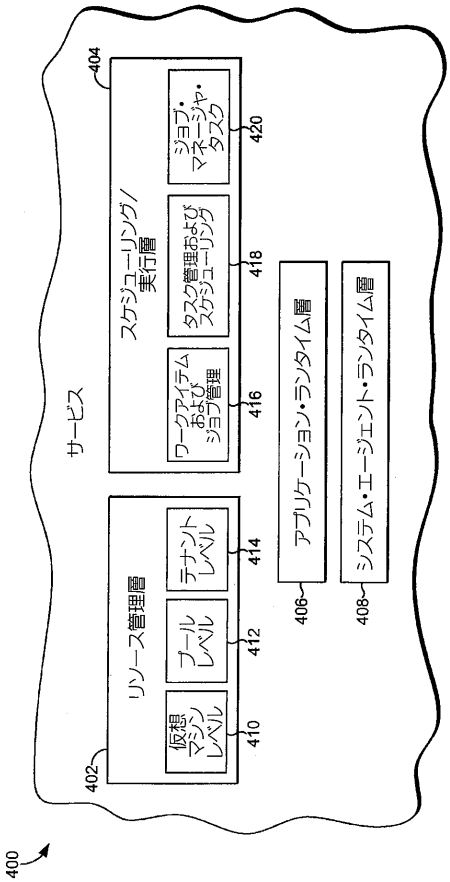
【図２】


200
ジョブ
ジョブ・
タスク・
マネージャ
タスク
タスクn
202
204

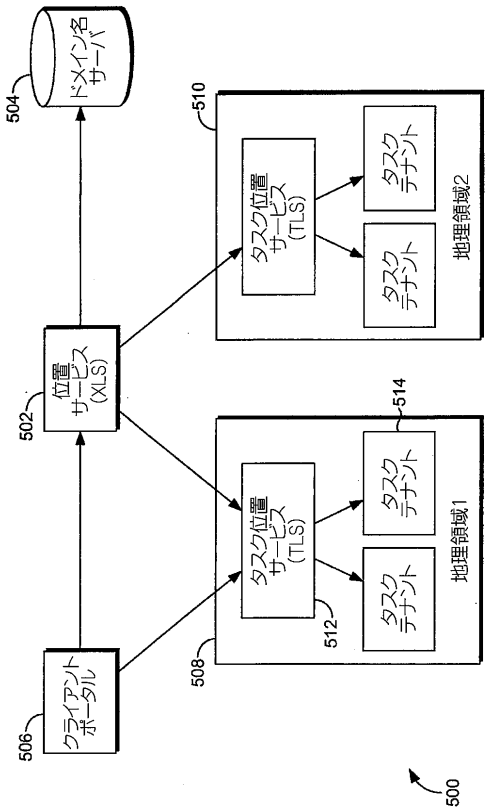
【図３】


300
プール
仮想マシン
仮想マシン
仮想マシン
302

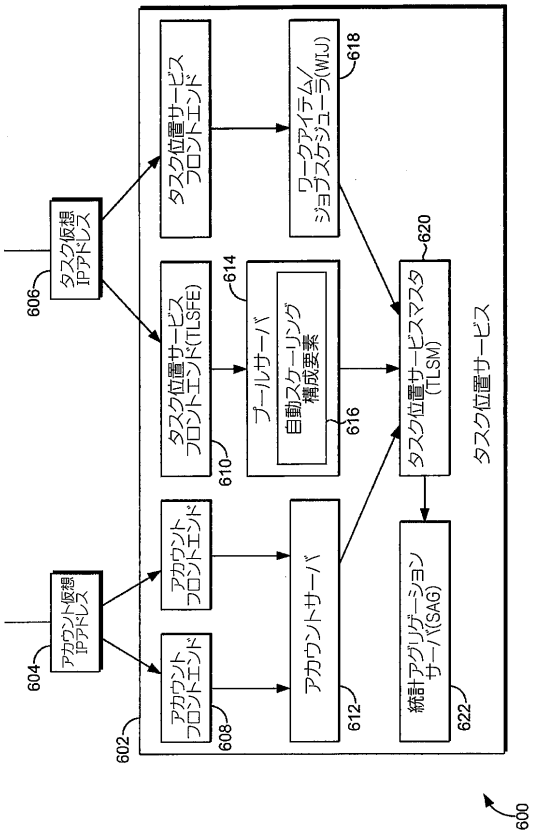
【図４】


サービス
404
スケジューリング／
実行層
ジョブ・
マネージャ・
タスク
420
タスク管理および
スケジューリング
418
ワークアイテム
および
ジョブ管理
416
リソース管理層
テナント
レベル
414
プール
レベル
412
仮想
マシン
レベル
410
アプリケーション・ランタイム層
406
システム・エージェント・ランタイム層
408
402
400

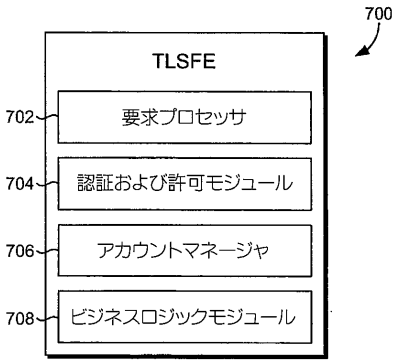
【図５】


ドメイン名
サーバ
504
510
タスク位置
サービス
(TLS)
タスク
テナント
タスク
テナント
地理領域2
位置
サービス
(XLS)
502
514
タスク位置
サービス
(TLS)
512
タスク
テナント
タスク
テナント
地理領域1
508
クライアント
ポータル
506
500

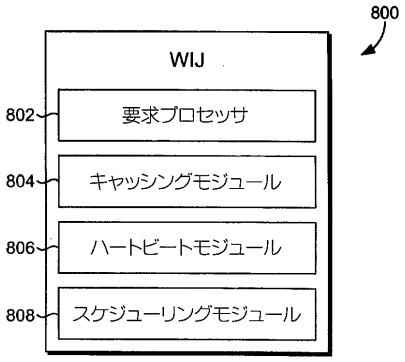
【図６】



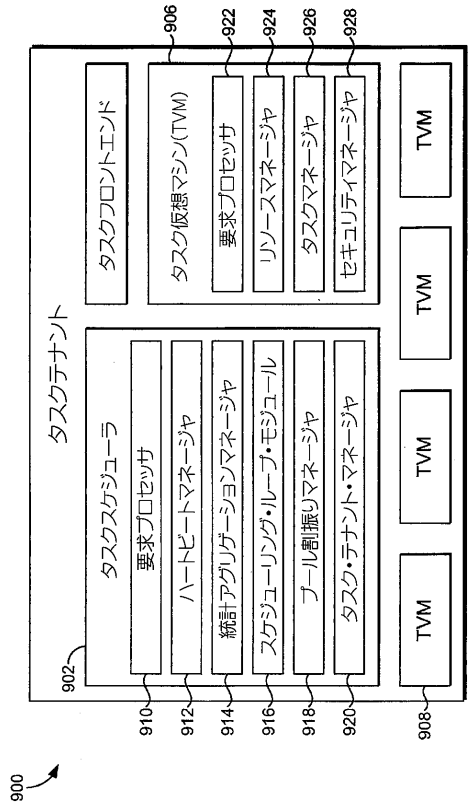
【図７】



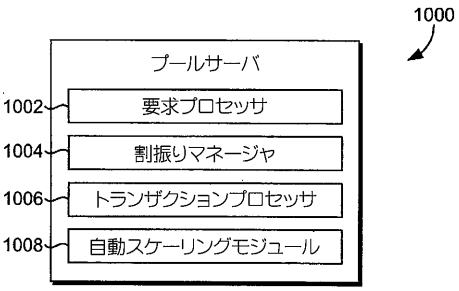
【図８】



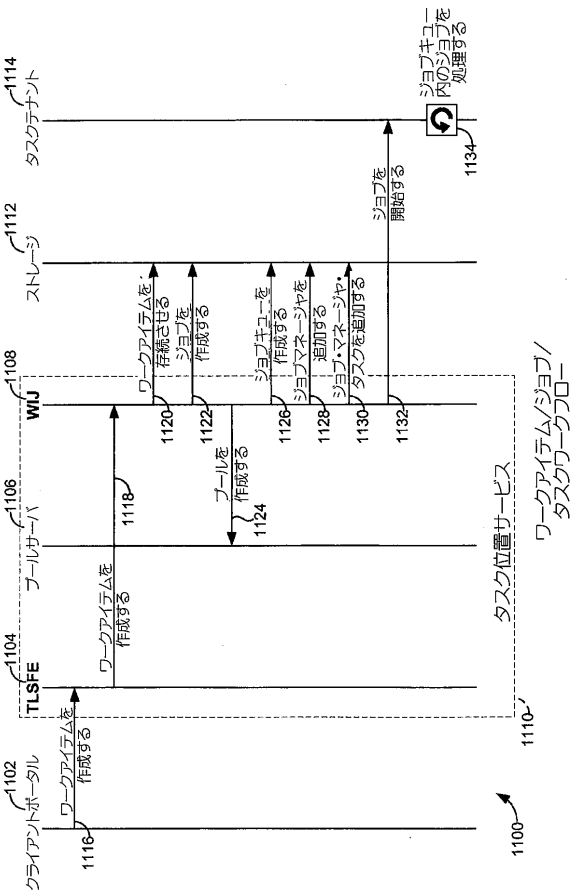
【図９】



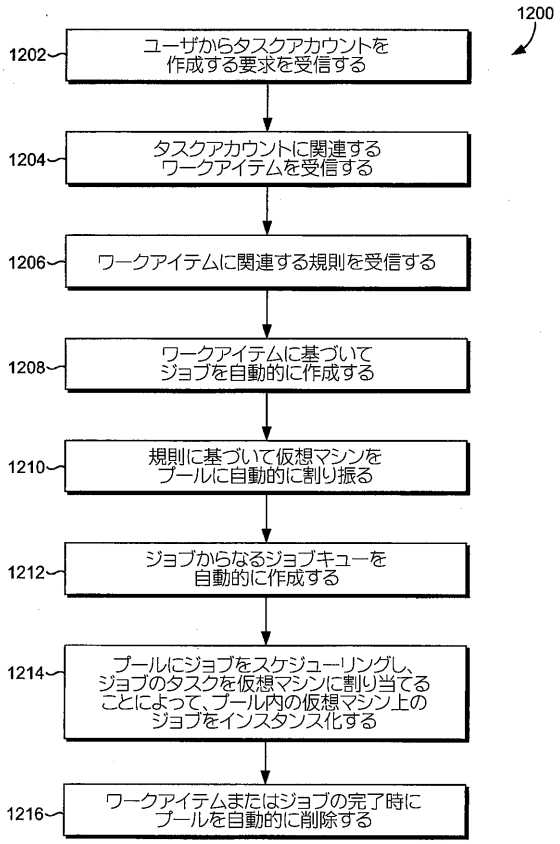
【図１０】



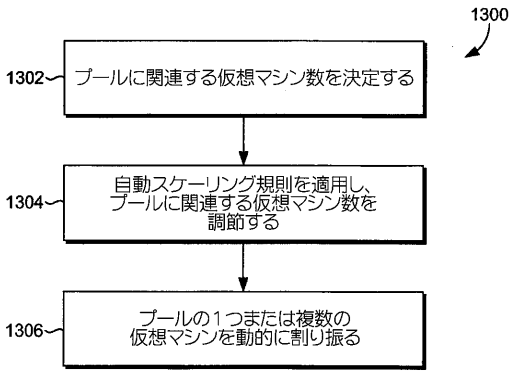
【図１１】



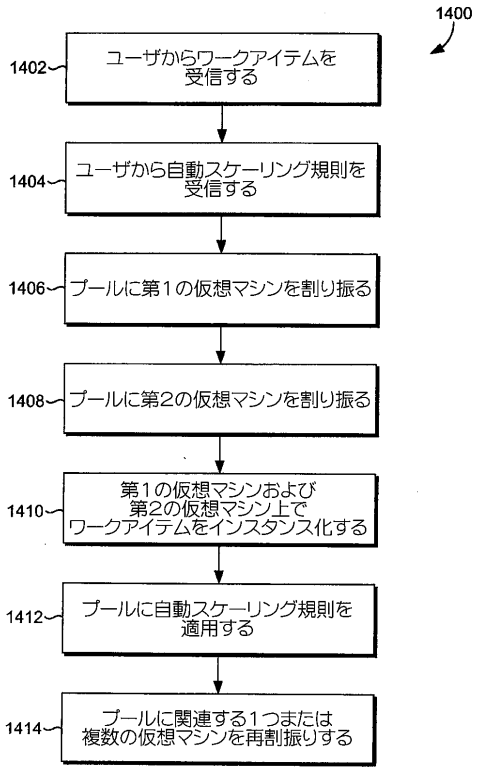
【図１２】



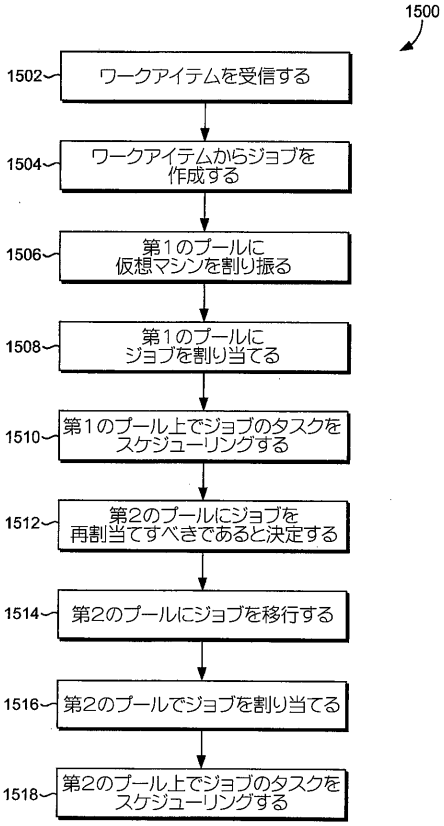
【図１３】



【図１４】



【図１５】


1500
1502
ワークアイテムを受信する
1504
ワークアイテムからジョブを作成する
1506
第1のプールに仮想マシンを割り振る
1508
第1のプールにジョブを割り当てる
1510
第1のプール上でジョブのタスクをスケジューリングする
1512
第2のプールにジョブを再割当てすべきであると決定する
1514
第2のプールにジョブを移行する
1516
第2のプールでジョブを割り当てる
1518
第2のプール上でジョブのタスクをスケジューリングする

フロントページの続き

(74)代理人 100153028
弁理士 上田 忠
(72)発明者 カルダー, ブラッドリー・ジーン
アメリカ合衆国ワシントン州９８０５２－６３９９, レッドモンド, ワン・マイクロソフト・ウェイ, マイクロソフト コーポレーション, エルシーエイ－インターナショナル・パテンツ
(72)発明者 ワーン, ジュイ
アメリカ合衆国ワシントン州９８０５２－６３９９, レッドモンド, ワン・マイクロソフト・ウェイ, マイクロソフト コーポレーション, エルシーエイ－インターナショナル・パテンツ
(72)発明者 ベデカー, ヴァマン
アメリカ合衆国ワシントン州９８０５２－６３９９, レッドモンド, ワン・マイクロソフト・ウェイ, マイクロソフト コーポレーション, エルシーエイ－インターナショナル・パテンツ
(72)発明者 サンカラン, スリラム
アメリカ合衆国ワシントン州９８０５２－６３９９, レッドモンド, ワン・マイクロソフト・ウェイ, マイクロソフト コーポレーション, エルシーエイ－インターナショナル・パテンツ
(72)発明者 マクネット, ザ・セカンド, マーヴィン
アメリカ合衆国ワシントン州９８０５２－６３９９, レッドモンド, ワン・マイクロソフト・ウェイ, マイクロソフト コーポレーション, エルシーエイ－インターナショナル・パテンツ
(72)発明者 グンダ, プラディープ・クマール
アメリカ合衆国ワシントン州９８０５２－６３９９, レッドモンド, ワン・マイクロソフト・ウェイ, マイクロソフト コーポレーション, エルシーエイ－インターナショナル・パテンツ
(72)発明者 ジャーン, ヤーン
アメリカ合衆国ワシントン州９８０５２－６３９９, レッドモンド, ワン・マイクロソフト・ウェイ, マイクロソフト コーポレーション, エルシーエイ－インターナショナル・パテンツ
(72)発明者 アントニー, シャム
アメリカ合衆国ワシントン州９８０５２－６３９９, レッドモンド, ワン・マイクロソフト・ウェイ, マイクロソフト コーポレーション, エルシーエイ－インターナショナル・パテンツ
(72)発明者 マニヴァンナン, カヴィタ
アメリカ合衆国ワシントン州９８０５２－６３９９, レッドモンド, ワン・マイクロソフト・ウェイ, マイクロソフト コーポレーション, エルシーエイ－インターナショナル・パテンツ
(72)発明者 スキョルスヴォルド, アリルド・イー
アメリカ合衆国ワシントン州９８０５２－６３９９, レッドモンド, ワン・マイクロソフト・ウェイ, マイクロソフト コーポレーション, エルシーエイ－インターナショナル・パテンツ
(72)発明者 カトリ, ヘマル
アメリカ合衆国ワシントン州９８０５２－６３９９, レッドモンド, ワン・マイクロソフト・ウェイ, マイクロソフト コーポレーション, エルシーエイ－インターナショナル・パテンツ

審査官 井上 宏一

(56)参考文献 特開２０１１－２３３０８６（ＪＰ, Ａ）
特開２０１１－１２９１１７（ＪＰ, Ａ）

(58)調査した分野(Int.Cl., ＤＢ名)
Ｇ０６Ｆ １１／２０
Ｇ０６Ｆ ９／４６ －９／５４