

(19) 日本国特許庁(JP)

(12) 特 許 公 報(B2)

(11) 特許番号

特許第3818655号

(P3818655)

(45) 発行日 平成18年9月6日(2006.9.6)

(24) 登録日 平成18年6月23日(2006.6.23)

(51) Int.Cl.

G06F 9/50 (2006.01)

F I

G06F 9/46 465D

請求項の数 31 (全 29 頁)

(21) 出願番号	特願2003-415773 (P2003-415773)	(73) 特許権者	390009531
(22) 出願日	平成15年12月12日(2003.12.12)		インターナショナル・ビジネス・マシー ズ・コーポレーション
(65) 公開番号	特開2004-199678 (P2004-199678A)		INTERNATIONAL BUSIN ESS MASCHINES CORPO RATION
(43) 公開日	平成16年7月15日(2004.7.15)		アメリカ合衆国10504 ニューヨーク 州 アーモンク ニュー オーチャード ロード
審査請求日	平成15年12月12日(2003.12.12)		
(31) 優先権主張番号	10/321680	(74) 代理人	100086243
(32) 優先日	平成14年12月17日(2002.12.17)		弁理士 坂口 博
(33) 優先権主張国	米国 (US)	(74) 代理人	100091568
			弁理士 市位 嘉宏
		(74) 代理人	100108501
			弁理士 上野 剛史

最終頁に続く

(54) 【発明の名称】 タスク・スケジューリングの方法、システム、およびプログラム製品

(57) 【特許請求の範囲】

【請求項1】

マルチレベル・コンピュータ・システムで処理される複数の要求をスケジューリングする方法であって、前記コンピュータ・システムの各レベルが、少なくとも1つのキューを有し、各キューが、少なくとも1つの処理機能を関連付けられ、

前記方法が、

- 第1レベルの前記キューに前記要求をバッファリングするステップと、
- 前記第1レベルに隣接する他のレベルとフロー情報を交換するステップと、
- 前記交換されるフロー情報に基づいて前記要求の夫々の分類値を得るステップと、
- 前記得られた分類値の夫々に基づいて前記要求の夫々をスケジューリングするステ

ップと

を含む方法。

【請求項2】

スケジューリングする前記ステップが、前記要求をディスパッチするステップを含む、請求項1に記載の方法。

【請求項3】

スケジューリングする前記ステップが、前記要求を分類するステップを含む、請求項1に記載の方法。

【請求項4】

スケジューリングする前記ステップが、前記要求を除去するステップを含む、請求項1

10

20

に記載の方法。

【請求項 5】

前記得られた分類値が事前定義の値より大きい場合に、前記第 1 レベルで前記要求を処理するステップをさらに含む、請求項 1 に記載の方法。

【請求項 6】

前記交換するステップが、前記フロー情報に基づいて、前記他のレベルのキューで要求が残存する確率が判定されるステップを含み、

前記分類値を得る前記ステップが、

e. 前記交換されるフロー情報に基づいてユーティリティ値を計算するステップであって、前記ユーティリティ値が、前記残存する確率と、要求の優先順位とによって決定されるステップと、

10

f. 前記計算されるユーティリティ値が事前定義の値より小さい場合に訂正処置を仮定するステップと、

g. 前記仮定される訂正処置を考慮した後の前記ユーティリティ値を再計算するステップと

を含む、請求項 1 に記載の方法。

【請求項 7】

h. 前記再計算されるユーティリティ値が前記事前定義の値より大きい場合に、仮定された訂正処置を行うステップと、

i. 前記仮定された訂正処置を行った後に、前記第 1 レベルで前記要求を処理するステップと

20

をさらに含む、請求項 6 に記載の方法。

【請求項 8】

訂正処置を仮定する前記ステップが、適当なキューが要求を受け入れることができない場合に、前記適当なキューの要求処理容量の増加を仮定するステップを含む、請求項 6 に記載の方法。

【請求項 9】

訂正処置を仮定する前記ステップが、次レベルのキューが要求を受け入れることができない場合に、適当なキューからの要求ディスパッチ速度の減少を仮定するステップを含む、請求項 6 に記載の方法。

30

【請求項 10】

マルチレベル・コンピュータ・システムで処理される複数の要求をスケジューリングするシステムであって、前記コンピュータ・システムの各レベルが、少なくとも 1 つのキューを有し、各キューが、少なくとも 1 つの処理機能を関連付けられ、

前記システムが、

a. 第 1 レベルの前記キューに前記要求をバッファリングする手段と、

b. 前記第 1 レベルと前記第 1 レベルに隣接する他のレベルとの間でフロー情報を交換する手段と、

c. 前記交換されるフロー情報に基づいて前記要求の夫々の分類値を得る手段と、

d. 前記得られた分類値の夫々に基づいて前記要求の夫々をスケジューリングする手段と

40

を含むシステム。

【請求項 11】

前記スケジューリングする手段が、前記要求をディスパッチする手段をさらに含む、請求項 10 に記載のシステム。

【請求項 12】

前記スケジューリングする手段が、前記要求を分類する手段をさらに含む、請求項 10 に記載のシステム。

【請求項 13】

前記スケジューリングする手段が、前記要求を除去する手段をさらに含む、請求項 10

50

に記載のシステム。

【請求項 14】

前記得られる分類値が事前定義の値より大きい場合に、前記第 1 レベルで前記要求を処理する手段をさらに含む、請求項 10 に記載のシステム。

【請求項 15】

前記交換する手段が、前記フロー情報に基づいて、前記他のレベルのキューで要求が残存する確率が判定される手段を含み、

前記分類値を得る前記手段が、

e. 前記交換されるフロー情報に基づいてユーティリティ値を計算する手段であって、前記ユーティリティ値が、前記残存する確率と、要求の優先順位とによって決定される手段と、

10

f. 仮定された訂正処置を検討した後の前記ユーティリティ値を再計算する手段と、

g. 前記再計算されるユーティリティ値が事前定義の値より大きい場合に、前記仮定された訂正処置を行う手段と

を含む、請求項 10 に記載のシステム。

【請求項 16】

前記仮定された訂正処置を行う前記手段が、第 2 レベル・キューが余分な要求を受け入れることができない場合に、第 1 レベル・キューの要求ディスパッチ速度を減らす手段を含む、請求項 15 に記載のシステム。

【請求項 17】

20

前記仮定された訂正処置を行う前記手段が、第 2 レベル・キューが余分な要求を受け入れることができない場合に、第 2 レベル・キューの要求処理容量を増やす手段を含む、請求項 15 に記載のシステム。

【請求項 18】

マルチレベル・コンピュータ・システムで処理される複数の要求をスケジューリングするコンピュータ・プログラムであって、前記コンピュータ・システムの各レベルが、少なくとも 1 つのキューを有し、各キューが、少なくとも 1 つの処理機能を関連付けられ、コンピュータ・システムに、

a. 第 1 レベルの前記キューに前記要求をバッファリングするステップと、

b. 前記第 1 レベルに隣接する他のレベルとフロー情報を交換するステップと、

30

c. 前記交換されるフロー情報に基づいて前記要求の分類値を得るステップと、

d. 前記得られた分類値の夫々に基づいて前記要求の夫々をスケジューリングするステップと

を実行させるコンピュータ・プログラム。

【請求項 19】

コンピュータ・システムでのロード・バランシングをする装置であって、

a. 複数のキューであって、

i. 着信要求をバッファリングする第 1 レベル・キューと、

ii. 複数の後続レベル・キューであって、キューの各後続レベルが、着信要求のクラスに対応する、複数の後続レベル・キューと

40

を含む複数のキューと、

b. ユーザ定義パラメータに基づいて、前記要求を複数の後続レベル・キューに分類する手段と、

c. 前記複数のレベルのキューの間でフロー情報を交換する手段と、

d. 前記交換されるフロー情報に基づいて前記要求の夫々の分類値を得る手段と、

e. 前記得られた分類値の夫々に基づいて前記キューからの前記要求の夫々を前記キューまたはターゲット・コンポーネントの少なくとも 1 つにディスパッチする手段とを含む装置。

【請求項 20】

ディスパッチする前記手段が、要求の平行ディスパッチの手段を含む、請求項 19 に記

50

載の装置。

【請求項 2 1】

情報を交換する前記手段が、前記キューからの要求のディスパッチ速度を制御する手段を含む、請求項 1 9 に記載の装置。

【請求項 2 2】

前記要求が、Q o S / S L A および前記後続レベル・キューからの情報に基づいて前記第 1 レベル・キューで分類される、請求項 1 9 に記載の装置。

【請求項 2 3】

コンピュータ・システムでのロード・バランシングのコンピュータ・プログラムであって、コンピュータ・システムに、

- a. 複数のキューを生成するステップであって、前記複数のキューが、
 - i. 着信要求をバッファリングする第 1 レベル・キューと、
 - i i. 複数の後続レベル・キューであって、キューの各後続レベルが、着信要求のクラスに対応する、複数の後続レベル・キューと
 を含む、前記生成するステップと
- b. ユーザ定義パラメータに基づいて、前記要求を複数のクラスに分類するステップと

- c. 前記複数のレベルのキューの間でフロー情報を交換する手段と、
- d. 前記交換されるフロー情報に基づいて前記要求の夫々の分類値を得る手段と、
- e. 前記得られた分類値の夫々に基づいて前記キューからの前記要求の夫々を前記キューまたはターゲット・コンポーネントの少なくとも 1 つにディスパッチするステップと

【請求項 2 4】

コンピュータ・システムで要求をスケジューリングするシステムであって、

- a. 複数のクラスタと、
- b. 複数のロード・バランサであって、各ロード・バランサが、少なくとも 1 つのクラスタに対応し、
 - i. 要求を受け取る手段と、
 - i i. 隣接レベル・ロード・バランサからフロー情報を交換する手段と、
 - i i i. 前記交換されるフロー情報に基づいて前記要求の夫々の分類値を得る手段と

- i v. 前記得られた分類値の夫々に基づいて次レベル・クラスタに前記要求の夫々をディスパッチする手段と
- を含む、複数のロード・バランサとを含むシステム。

【請求項 2 5】

複数のロード・バランサを含むネットワーク内で複数の要求をスケジューリングする方法であって、

- a. 第 1 レベル・ロード・バランサで要求を受け取るステップと、
 - b. 前記要求に関する情報を隣接レベル・ロード・バランサと交換するステップと、
 - c. 前記交換される情報に基づいて前記要求の夫々の分類値を得るステップと、
 - d. 前記得られた分類値の夫々に基づいて前記要求の夫々をスケジューリングするステップと
- を含む方法。

【請求項 2 6】

前記交換するステップが、前記情報に基づいて、前記隣接レベルのキューで要求が残存する確率が判定されるステップを含み、

前記分類値を得る前記ステップが、

- e. 前記要求の転送の閾ユーティリティ値を判定するステップと、
- f. 前記要求の転送のユーティリティ値を得るステップであって、前記ユーティリティ

10

20

30

40

50

値が、前記残存する確率と、要求の優先順位とによって決定されるステップと、

g. 前記ユーティリティ値と前記閾ユーティリティ値とを比較するステップと、

h. 前記ユーティリティ値が前記閾ユーティリティ値より小さい場合に訂正処置を行うステップと

を含む、請求項 2 5 に記載の方法。

【請求項 2 7】

ユーティリティの閾値を判定する前記ステップが、

i. 前記ネットワークがサービスを必要とする Q o S 要件および S L A 要件を識別するステップと、

j. 前記ネットワーク内で前記要求に割り振られるリソースを判定するステップと

をさらに含む、請求項 2 6 に記載の方法。

10

【請求項 2 8】

前記ユーティリティ値を得る前記ステップが、

k. 要求ヘッダから前記要求に関係する Q o S および S L A を識別するステップと、

l. 次レベル・バランサで前記要求に対応する割り振られるリソースを判定するステップと、

m. 前記ユーティリティ値を得るために前記識別された情報および前記判定された情報を処理するステップと

を含む、請求項 2 6 に記載の方法。

20

【請求項 2 9】

訂正処置を行う前記ステップが、前記ロード・バランサでの要求ディスパッチ速度を下げるステップを含む、請求項 2 6 に記載の方法。

【請求項 3 0】

訂正処置を行う前記ステップが、余分な負荷を受け入れるためにロード・バランサ容量を増やすステップを含み、ロード・バランサ容量を増やすステップが、前記要求の前記ユーティリティ値を増やす、請求項 2 6 に記載の方法。

【請求項 3 1】

複数のロード・バランサを含むネットワーク内で複数の要求をスケジューリングするコンピュータ・プログラムであって、コンピュータ・システムに、

a. 第 1 レベル・ロード・バランサで要求を受け取るステップと、

b. 前記要求に関する情報を隣接レベル・ロード・バランサと交換するステップでと、

c. 前記交換される情報に基づいて前記要求の夫々の分類値を得るステップと、

d. 前記得られた分類値の夫々に基づいて前記要求の夫々をスケジューリングするステップと

を実行させるコンピュータ・プログラム。

30

【発明の詳細な説明】

【技術分野】

【0 0 0 1】

本発明は、コンピュータ・システムのタスク・スケジューリングの分野に関する。具体的には、本発明は、コンピュータ・システムで除去される要求の数の削減と、サービス品質 (Quality of Service、Q o S) およびサービス・レベル・アグリーメント (Service Level Agreement、S L A) の実施とタスク・スケジューリングの方法、システム、およびフレームワークに関する。

40

【背景技術】

【0 0 0 2】

コンピュータ・システムは、さまざまなタスクを実行し、さまざまな要求を処理する処理能力およびメモリを有するシステムと定義することができる。そのような要求は、ネットワークを介するデータベースへのアクセスなどの単純なデータ要求から、元の要求を処理するためにデータベース内のあるデータを必要とするアプリケーションの実行の要求などの複雑な要求までの広範囲にわたる可能性がある。

50

【0003】

要求（またはタスク）を受け取り、処理するコンピュータ・システムは、その容量に関するある制限を有する。コンピュータ・システムが同時に処理できる要求の最大の個数には限度がある。そのような制限は、コンピュータ・システムの設計および容量に起因して生じる可能性がある。したがって、システムがそれに基づいて展開されるCPU速度、メモリ容量、ネットワーク接続速度などのパラメータによって、システムの処理できる要求の数が制限される。したがって、要求の数がこの限度を超えると、完全には処理されない要求がいくつか生じる。コンピュータ・システムに達する前に除去される要求も生じる。これらの要求は、コンピュータ・システムによって全く受け取られない。また、コンピュータ・システムによって受け取られるが、部分的に実行されただけで打ち切られる要求も生じる。これは、通常は、メモリの不足またはネットワーク接続のタイムアウトに起因して発生する。

10

【0004】

そのようなシステムの1つの例を、インターネットベースのオンライン株式取引サービスとすることができる。そのようなサービスによって、ユーザの要求時に、特定の株の価格に関する情報がユーザに供給される。しかし、そのようなオンライン・インターネット・サービスのユーザの数が増加するにつれて、オンライン・インターネット・サービスのサーバによって並行に処理されるユーザ要求の数が、時々収拾不可能になる。たとえば、バスケットボールに関する特定の情報を供給するオンライン・インターネット・サービスのサーバは、全国レベルのバスケットボール・トーナメントの決勝中に、ユーザ要求で過負荷になる。実際に、要求の過負荷に起因して、部分的にしか処理されないユーザ要求が生じ、全く処理されない要求が生じる。これらの、完全には処理されないか全く処理されない要求は、除去されたと見なされる。

20

【0005】

要求自体の完全な処理にクリティカルであるトランザクションが処理されない場合に、要求は除去されたとみなされる。要求Aの部分的な処理によって生成される要求Bが、要求Aの完全な処理にクリティカルであると仮定する。しかし、要求Bが除去される（たとえば、要求Bを処理するコンポーネントに対する要求の過負荷に起因して）場合に、要求Aを完全に処理することはできない。そのような場合に、要求Aも、除去されたとみなされる。

30

【0006】

したがって、要約すると、要求が除去されたとみなされるには、（i）システムまたはシステム・コンポーネントによる要求の処理（部分的なまたは完全な）の前に、コンポーネントに対する要求過負荷に起因して、要求が、処理に関する将来の検討から除外される（すなわち、完全に除去される）か、（ii）要求の完全な処理にクリティカルなトランザクションが打ち切られるか、（iii）元の要求の処理から生成され、その完全な処理が元の要求の完全な処理にクリティカルである別の要求が除去された場合である。

【0007】

過負荷に起因する要求除去の問題に対処する方法の1つが、同一システムの複数の複製（すなわち複数のサーバ）を使用して、着信要求を処理することである。したがって、単一のサーバの代わりにサーバのクラスタを使用して、要求を処理することができる。ロード・バランサは、クラスタの前で要求を受け取り、複製の間での要求の公平な分配を保証するのに使用される。ロード・バランサは、コンポーネントの過負荷を防ぐために、クラスタのさまざまなコンポーネントの間でユーザ要求を均等に分配するハードウェア装置またはソフトウェア・プログラムである。たとえば、ロード・バランサを使用して、サーバの複数の複製（httpサーバなど）の間でユーザ要求を分配することができる。この分配は、各コンポーネントの作業負荷、要求の内容タイプ（ビデオまたはテキスト）、コンポーネントの地理的配置、および期待されるQoSおよびSLAなどの、事前定義の判断基準に基づくものとして行うことができる。さらに、後で説明するように、類似するコンポーネントの間でこれらの要求を分配する事前定義の方法や技法が存在する可能性がある。

40

50

【0008】

同一サーバの複数のインスタンスを使用すると、各サーバによって処理される要求の数が減る。これによって、サーバが過負荷になる機会が減る。しかし、いくつかのケースでの要求の数は、非常に多いので、その要件を満たすために複数のサーバを有することは非実用的である。ネットワーク伝送速度の向上およびユーザ／クライアントの数の増加によって、ネットワークを介する要求の伝送が、コンポーネントで要求を処理する速度またはロード・バランサによって要求が転送される速度あるいはその両方よりはるかに速い。このため、コンポーネント／ロード・バランサでの要求の蓄積が引き起こされ、したがって、コンポーネント／ロード・バランサが過負荷になる。これが、コンポーネント／ロード・バランサでの要求除去につながる。他のいくつかの場合に、コンポーネントが、要求を処理し始めるが、高いロード条件に起因して、処理が、事前定義の期間内に完了しない。これも、要求除去につながる可能性がある。

10

【0009】

要求除去は、コンピュータ・システムの性能に悪影響を及ぼす。第1に、リソースすなわち、要求を処理し、転送したロード・バランサのリソース、または要求の部分的処理に使用されたリソースあるいはその両方の浪費がある。これは、リソース使用率の減少につながる。第2に、処理される時に、要求が、コンポーネントのいくつか（その要求を処理したもの）の状態の変化をもたらす場合がある。要求が、中間コンポーネントで除去される場合には、そのシステムを、最も最近の一貫性のある状態にロール・バックする必要がある可能性がある。言い換えると、システム全体を、除去された要求の処理が開始される前のそのシステムの状態に戻さなければならない。各ロールバックは、要求処理でのシステムの応答時間を悪化させ、システムのスループットを低下させ、システム・リソースも浪費する。第3に、システム・リソースの浪費が、十分なシステム・リソースが使用可能でなくなることに起因して、そうでなければ処理できたはずの他のリソースの除去を引き起こす場合がある。第4に、上で述べたものと同一の理由から、そうでなければよりよいサービス品質で処理できたはずのいくつかの他の要求が、より低いサービス品質で処理される。第5に、除去される要求の部分的処理に浪費された時間が、除去されたか失敗した要求に関するユーザ応答時間に追加される。これによって、ユーザに提供されるサービス品質（QoS）が阻害される可能性があり、ユーザとネットワークの間のサービス・レベル・アグリーメント（SLA）の侵害にもつながる可能性がある。最後に、要求の処理の応答時間が長くなるにつれて、ユーザの不満が高まる。

20

30

【0010】

上で述べた要求除去の不利益に鑑みて、要求除去の数を減らすことが肝要である。要求除去を減らすためにロード・バランサを使用する複数の方法および技法がある。ロード・バランサによってユーザ要求の分配を実行する例示的な方法の1つが、ラウンド・ロビン・ドメイン・ネーム・サーバ（Round Robin Domain Name Server、RR-DNS）手法である。この方法では、要求が、回転式（すなわちラウンド・ロビン式）に1つずつ複数のサーバにルーティングされる。

【0011】

特定のユーザ要求のターゲット・コンポーネントを決定するための、クラスタ（類似するコンポーネントの）内の個々のコンポーネントの間での情報の流れという発想に基づく他のいくつかのロードバランシング解決策がある。この情報は、負荷がロード・バランサによって平衡化される、クラスタ内の各コンポーネントの負荷レベルに関連するものとすることができる。

40

【0012】

そのような解決策の1つが、米国特許第5938732号、表題"Loadbalancing and fail over of network services"に開示されている。この方法では、コンポーネントのクラスタ内の通信の維持と、それらの間の協力の調整に対処する。これは、サービスを提供するコンポーネントの1つまたは複数が、障害を介するなど、時々使用不能になる場合であっても、そのグループによって提供されるサービスが使用可能なままになるようにする

50

ために行われる。各処理要素は、グループ内の他のすべての処理要素に周期的に制御メッセージを送って、通信を維持するのを助ける。このメッセージは、送出側コンポーネントの状況ならびにグループ内の他のコンポーネントの知覚された状況に関するデータが含まれる。

【0013】

もう1つの解決策が、チョウ (Myongsu Choe) およびトロップ (CarlTroppe) 著の研究論文、"Flow Control and Dynamic Load Balancing in Time Warp"で提案された。この論文では、フロー制御および動的ロード・バランシングを一体化するアルゴリズムが開示されている。フロー制御は、クラスタ内のプロセッサによって、その間で負荷を共用または分配するのに使用される。この場合のフロー制御は、ターゲット・コンポーネントの間

10

【0014】

オスマン (Ossama Othman)、オライアン (Carlos O'Ryan)、およびシュミット (Douglas C. Schmidt) 著の研究論文 "The Design and Performance of an Adaptive CORBA Load Balancing Service"でも、解決策が提案されている。この論文は、将来の研究で、所与のクラスタ (コンポーネントの) からターゲット複製 (コンポーネント) を決定するためにさまざまなロード・バランサの間のフロー制御を使用する必要を掲げている。

【0015】

上で説明した解決策のすべてで、要求を処理するターゲット・コンポーネントを決定するために、クラスタ (類似するコンポーネントの) 内のさまざまなコンポーネントの間での情報交換が使用されている。しかし、これらのどれもが、高い要求速度で働くことを意図されたシステムでの要求除去の問題に対処していない。完了の前に最終的に除去される要求も、処理され、これによって、リソースの浪費と、処理時間および電力消費の増加が引き起こされる。これが、より低いシステム効率をもたらす。これらの解決策は、上で説明した要求除去の通常の不利益も有する。

20

【特許文献1】米国特許第5938732号、表題 "Load balancing and fail over of network services"

【非特許文献1】チョウ (Myongsu Choe) およびトロップ (CarlTroppe) 著、"Flow Control and Dynamic Load Balancing in Time Warp"

【非特許文献2】オスマン (Ossama Othman)、オライアン (Carlos O'Ryan)、およびシュミット (Douglas C. Schmidt) 著、"The Design and Performance of an Adaptive CORBA Load Balancing Service"

30

【発明の開示】

【発明が解決しようとする課題】

【0016】

したがって、上の説明に鑑みて、コンピュータ・システムでの要求除去の数を減らすことができるシステムおよび方法の必要が存在する。具体的に言うと、ロード・バランサでの要求除去の数を減らすシステムおよび方法の必要がある。ロード・バランサでのサービス品質 (QoS) およびサービス・レベル・アグリーメント (SLA) の実施を容易にするシステムおよび方法の必要も存在する。システムのスループットを高める、ロードバ

40

ランシング・フレームワークの必要も存在する。これは、システム・リソースの浪費を減らすために必要である。

【課題を解決するための手段】

【0017】

本発明の目的は、コンピュータ・システムで要求をスケジュールすることである。

【0018】

本発明の具体的な目的は、コンピュータ・システムの間コンポーネントおよびロード・バランサで除去される要求の数を減らすことである。

【0019】

本発明のもう1つの具体的な目的は、サービス品質 (QoS) およびサービス・レベル

50

・アグリーメント（S L A）を実施することである。

【0020】

本発明のもう1つの具体的な目的は、フロー制御メッセージを処理し、Q o S / S L Aを実施する、ロード・バランサ内のシステムおよび方法を提供することである。

【0021】

本発明のもう1つの具体的な目的は、ロード・バランサならびにシステムの要求スループットを高めることである。

【0022】

本発明のもう1つの目的は、システム・リソース使用率を最大にすることである。

【0023】

本発明のもう1つの目的は、要求の区別された分配を提供することである。

【0024】

本発明のもう1つの目的は、システム・ロールバックの数を最小にすることである。

【0025】

本発明のもう1つの目的は、ユーザへの応答時間を減らすことである。

【0026】

本発明の上記および他の目的は、複製の組の間で負荷を平衡化する、方法、システム、およびコンピュータ・プログラム製品によって提供される。この方法は、実施されるポリシーに基づいてコンピュータ・システムに到達する要求をスケジューリングすることを目的とする。コンピュータ・システムには、複数のレベルが含まれ、各レベルは、少なくとも1つのキューを有し、各キューは、少なくとも1つのバッファを有し、各キューは、少なくとも1つの処理コンポーネントに関連する。各キューは、それに関連する任意の個数のクラスタに対応するものとする。着信要求は、完全に処理される前に、1つまたは複数のレベルを介して処理される。フロー制御情報が、要求除去を減らすために2つのレベルの間で交換される。本発明の一実施形態では、すでに説明したロード・バランサ内の上の方法の実施形態を説明する。着信要求が、第1レベル・キューに保管される。要求は、さらに、ユーザ定義のパラメータ（Q o S / S L Aなど）に基づいて、複数のクラスに分類される。その後、要求は、事前定義の判断基準に基づいて、複数の第2レベル・キューの1つに転送される。たとえば、各第2レベル・キューを、要求のクラスに対応するものとする。キューの2つのレベルの間に、制御情報のフローが存在する。この制御情報は、第2レベル・キューの状態に基づいて要求を適当な第2レベル・キューに転送するのに使用され、これによって、ロード・バランサ内で除去される要求の数が減る。

【0027】

本発明のもう1つの実施形態では、上で述べたロード・バランサを、フレームワークでを使用して、複数のクラスタの間で負荷を動的に平衡化することができる。着信要求が、第1レベル・ロード・バランサで受け取られる。ロード・バランサは、1つまたは複数の要求を、要求のタイプまたはターゲット・コンポーネントの状態あるいはその両方に基づいて、ターゲット・コンポーネントに並行にディスパッチする。ロード・バランサは、要求除去が起ころうとしているのを感知する場合に、フロー制御メッセージを生成し、そのメッセージを前レベル・ロード・バランサに送る。フロー制御メッセージを受け取る時に、ロード・バランサは、対応する処置を講ずることを決定することができ、その結果、後のレベルの要求除去が回避される。ロード・バランサは、前レベル・ロード・バランサおよび次レベル・ロード・バランサと交換できるフロー制御メッセージも生成することができる。

【0028】

本発明の好ましい実施形態を、以下で、本発明を例示するが制限しない添付図面に関して説明する。図では、類似する符号が類似する要素を示す。

【発明を実施するための最良の形態】

【0029】

10

20

30

40

50

使用される用語法

論理インスタンス：論理インスタンスとは、物理インスタンス／コンポーネントまたは物理インスタンス／コンポーネントのある容量のいずれかである。論理インスタンスは、物理インスタンスの名前／アドレスと、物理インスタンス内で割り振られた容量との対によって表すことができる。本明細書の用語「インスタンス」および「複製」の使用のすべてが、「論理インスタンス」の同意語である。

【0030】

クラスタ：クラスタとは、サポートされる機能性、インスタンスを使用するためのプロトコルなどに関して同一である論理インスタンスの集合である。

【0031】

次レベル・ロード・ディストリビュータ：ロード・ディストリビュータ（LD）-xの次レベル・ロード・ディストリビュータ（LD）とは、LD-xに隣接し、トポロジ的にその次のLDである。たとえば、図6では、LD606およびLD608が、LD602の次レベルLDであり、LD614が、LD606およびLD608の次レベルLDである。

10

【0032】

前レベル・ロード・ディストリビュータ：LD-xの前レベル・ロード・ディストリビュータとは、LD-xに隣接し、トポロジ的にその前のLDである。たとえば、図6では、LD602が、LD606およびLD608の前レベルであり、LD606およびLD608が、LD614の前レベルLDである。

20

【0033】

フロントエンド：フロントエンドとは、エンド・ユーザから要求を受け取る、アプリケーション・インスタンスのクラスタである。

【0034】

要求分類要因（RCF）：RCFは、要求をカテゴリに分類（他の要求から区別）するのに使用される。たとえば、本発明の一実施形態で、要求に関連する顧客識別またはサービス・クラス／グレード、あるいは顧客IDと要求のサービス・クラスの組合せを、RCFとして使用して、着信要求を分類する。RCF値の計算に使用することができる他のパラメータは、要求タイプ（たとえば、カタログ表示要求、ショッピング要求など）、要求の複雑さ、要求の優先順位などとすることができる。

30

【0035】

負荷モニタ：Cの状態またはCに対する負荷あるいはその両方を監視できるような所与の時点でのシステムまたは別のコンポーネントCに関連付けられるコンポーネントM。Cの状態には、可用性、応答時間、CPU消費などが含まれる。Cに対する負荷は、CまたはCを展開する基礎になるシステムの消費される容量の尺度である。

【0036】

リソース・マネージャ：ある顧客から別の顧客へのリソースの割振り／割振り解除、または関連するシステムの容量プランニング、あるいはその両方が可能なコンポーネントR。

【0037】

本発明は、要求除去を最小にするためにフロー制御技法を使用するモデル（システムおよび方法）を提供する。要求のスケジューリングは、既存のポリシー／技法を使用して実行することができる。本明細書での説明のために、そのようなポリシーの単純で一般的な例を選ぶ、すなわち、高いユーティリティ値を有する要求を転送し、要求のユーティリティ値は、要求が完全に処理される確率と、要求の優先順位（要求クラスとも称する）とによって決定される。この例を、図3に関して後で詳細に説明する。要求除去を減らすために実施する／従うことができる多数のそのようなポリシーがありえることは、当業者に明白であろう。

40

【0038】

図1に、本発明の好ましい実施形態による、要求を処理するコンピュータ・システム100を示す。システム100は、1つまたは複数のレベルを有することができ、各レベル

50

は、その中に1つまたは複数のキューを有する。各レベルのキューは、さらに、1つまたは複数のバッファと、それに関連する1つまたは複数の処理機能を有することができる。例示のために、図1に、コンピュータ・システムの2つのレベル（すなわち、108および112）だけを示す。また、例示のために、図1に、その中に1つのキュー（すなわち102および110）だけを有する各レベルを示す。また、各キューは、1つのバッファ（すなわち、104および122）だけを有するものとして図示されている。

【0039】

広義には、要求がシステム100で処理される形は、次の通りである。キュー102に、バッファ104およびそれに関連する処理機能106が含まれる。キュー102内のバッファ104は、着信要求を受け取り、これらの着信要求を処理機能106に転送する。処理機能106は、要求ディスパッチ機能（要求のディスパッチに使用される）または要求分類機能（要求を分類するのに使用される）とすることができる。例示のために、図1に、処理機能106が要求ディスパッチ機能であるキュー102を示す。キューには、任意の数のクラスタを関連付けることができる。任意選択として、キューに、クラスタを関連付けられないことができる。そのような場合には、あるキューに別のキューを続け、最後のキューだけがクラスタを関連付けられるようにすることができる。クラスタは、同等のタスクを実行するコンポーネント・インスタンスを含む。例示のために、図1に、クラスタ108に関連するキュー102を示し、このクラスタ108は、実際には、要求実行機能を実行するコンポーネントのクラスタである。要求ディスパッチ機能によって、要求が、クラスタ108にディスパッチされ、クラスタ108によって、要求が処理され、次のレベルのキュー110に転送される。各キューに、前のレベル／次のレベルのキューからの処理制御メッセージ／フロー制御メッセージを交換する交換モジュール116と、フロー制御メッセージの指示を実施する実施モジュール118と、制御信号／フロー制御信号を作成し、送る時を感知する感知モジュール120も含まれることが好ましい。そのようなモデルの実施形態が、必ずしも3つの別個のモジュールのすべてを有するのではないことに留意されたい。たとえば、ある実施形態が、3つのすべてのモジュールの機能を実行する1つのモジュールだけを有する場合がある。しかし、3つのすべてのモジュールの機能を、どの実施形態でも実行する必要がある。これらモジュールの使用を、本明細書で後で詳細に説明する。キュー110は、バッファと、要求ディスパッチ機能も有する。要求ディスパッチ機能によって、要求が、キュー110に関連するクラスタ112にディスパッチされる。クラスタ112のコンポーネントによって、要求が適当に処理される。クラスタ112からの要求が、要求の処理要件に応じて複数のキューに転送される場合がありえることに留意されたい。この処理要件に関する情報を、要求のヘッダに保管することができ、したがって、この情報は、ターゲット・キューを決定するためにクラスタ112のコンポーネントによって処理される。

【0040】

開示されるシステムでは、異なるレベルのキューの間に情報（メッセージ）のフローがある。言い換えると、キュー102および110は、情報を交換する（114）。既に述べたように、この2つのキューの交換モジュールが、ある判断基準に基づいて要求除去を減らすためにキュー102が訂正処置を講じられるようにするために、情報114を交換する。バッファ管理ポリシーを使用して、訂正処置を完全にまたは部分的に実行することができる。バッファ管理ポリシーの簡単な例の1つを、下で例示のため説明する。

【0041】

バッファ管理ポリシーの例は、次の通りである。要求を分類（またはディスパッチ）し、F C F S（先着順サービス）に基づいて次のレベルに転送する。キューによって、フロー制御情報に基づいて、次レベルのキューで要求が残存する確率が判定される。あるレベルの各キューは、フロー制御信号を前のレベルのキューに送ることができ、制御信号を次のレベルのキューに送ることができる。この情報は、キューにある要求の残存確率に対応する負荷情報に関するものとしてすることができる。この確率および要求の優先順位を使用して、要求のユーティリティ値を判定することを図3に関して詳細に説明する。着信要求の数

10

20

30

40

50

が、キューの要求処理容量を超える場合には、キューは、より高いユーティリティ値の要求を受け取り、他の要求を除去すると決断することができる。キューは、次レベル・キューに転送できなかった要求をハッシュ・テーブルに編成することもでき、その結果、それらの要求を、次レベル・キューが要求を受け取る準備ができるとすぐに転送できるようになる。

【0042】

要求除去を減らすために、キューによって、複数の訂正処置を実行することができる。これらの訂正処置を、図2に関して詳細に説明する。これらの訂正処置は、実施モジュール118によって実行され、除去される要求の数のかなりの削減がもたらされる。また、本発明の好ましい実施形態では、これらの処置によって、QoS/SLA標準規格の厳密な実施がもたらされる。

10

【0043】

図2は、本発明の好ましい実施形態による、要求を処理するために従われるステップのシーケンスを示す流れ図である。以下の説明では、要求のユーティリティ値の例が、本発明の好ましい実施形態の動作を示すだけのために使用されることに留意されたい。以下の説明を、どのような形でも、制限的であると考えてはならない。

【0044】

ステップ202で、キュー102によって、要求を受け取り、バッファに保管する。ステップ204で、キュー102の交換モジュール116によって、キュー110の負荷およびクラスタ112の負荷に関する情報114を受け取る。ステップ206で、キュー102の実施モジュール118によって、情報114と、任意選択として要求に関する情報（要求ヘッダに保管されている可能性がある）に基づいて、クラスタ112への要求転送の「ユーティリティ値」を計算する。例示のために、ユーティリティ値が、要求の完了の確率に依存し、この確率が、後続レベルのキューおよびクラスタの負荷に基づくものと仮定する。次レベル・キューおよび対応するクラスタの負荷が高いと、要求の完了の確率が下がり、したがって、要求のユーティリティ値が下がる。こうなるのは、完了の低い確率を有する要求がキュー102によって転送される場合に、その要求によってキュー102の処理時間およびシステム・リソースが利用されるからである。この要求が、その後キュー110で除去され、これによってシステム・リソースおよび時間が浪費される可能性が、非常に高い。ユーティリティ値は、要求のクラスにも基づくものとすることができる。たとえば、QoS/SLAレベル・ゴールドの要求の転送は、QoS/SLAレベル・ブロンズの要求の転送よりも、商業的に有用である。要求のユーティリティ値を判定するために導入することができる多数の他の要因があることは、当業者に明白である。しかし、説明のために、情報114を使用して計算することができる完了の確率と、要求パケットのヘッダから得ることができる要求クラスだけを含むように要求のユーティリティ値を制限する。この例によるユーティリティ値の計算は、図3に関して詳細に例を使用して説明する。

20

30

【0045】

ステップ208で、要求の計算されたユーティリティ値を、キュー102に割り当てられた閾ユーティリティ値と比較する。閾ユーティリティ値は、計算され、各キューに割り当てられる。この閾値は、特定のキューについて固定された値である。この閾値は、次レベル・キューおよび対応するクラスタの推定負荷に関する情報を使用することと、キューによって処理される要求クラスを検討することによって計算される。この値は、システムの後続のスケール・アップまたはスケール・ダウンの度に、またはキューによって処理される要求クラスの変更について、変更されなければならない。

40

【0046】

計算されたユーティリティ値が、閾ユーティリティ値を超える場合には、ステップ210で、キュー102の処理機能106によって要求を処理する（ディスパッチするか分類する）。しかし、計算されたユーティリティ値が、閾ユーティリティ値より小さい場合には、ステップ212で、キュー102の感知モジュールによって、要求の修正されたユー

50

ティリティ値を計算する。修正されたユーティリティ値は、キューが訂正処置を実施すると仮定した後に要求のユーティリティ値を増やすために計算される。この値は、訂正処置を講じるコストに基づく。コストは、主に、訂正処置を講じるためのメモリ消費および処理能力使用量によって左右される。ユーティリティ値を、訂正処置が講じられた後の要求の完了の確率および要求のクラスに基づくものとすることもできる。

【0047】

キューによって講じられる訂正処置は、下記とすることができる。まず、キューの感知モジュールによって、制御信号を送ってその要求ディスパッチ速度を変更することができ、その結果、次レベル・キューおよび対応するクラスタの負荷が、最適レベルに達する。要求が、以前に、次レベル・キューでの過負荷に起因して低いユーティリティ値を有した場合には、制御信号によって、負荷が減らされ、したがって、要求の完了の確率が増える。これによって、要求のユーティリティ値の増加がもたらされる。この処置に関連するコストは、通常は、この処置が小さいコードの実行だけを伴うので、非常に低い。したがって、要求の修正されたユーティリティ値は、一般に要求の元のユーティリティ値より高い（完了の確率の増加に起因する）。この処置を講じることに関係するステップを、図4に関して詳細に説明する。ステップ214で、修正されたユーティリティ値を、修正された閾ユーティリティ値と比較する。スケール・アップが実行されない（すなわち、第2の処置が実行されない）場合に、修正された閾ユーティリティ値が、元の閾ユーティリティ値と同一になることに留意されたい。修正された閾ユーティリティ値が、修正された閾値より小さい場合には、ステップ216で、要求を除去する。そうでない場合には、ステップ218で、ステップ212で仮定された訂正処理を講じ、要求を、キューによってそのクラスタにディスパッチし、これによって、ステップ220で、適当な処理の後に、要求が次のレベルに転送される。

【0048】

ユーティリティ値、閾ユーティリティ値、修正されたユーティリティ値、および修正された閾ユーティリティ値の概念をより明瞭に示すため、図3の表を検討されたい。説明される形が、例示のために働くことだけを意図され、どのような形でも本発明を制限するものと考えてはならないことに留意されたい。説明を単純にするために、ユーティリティ値を計算する非常に単純な例を選ぶ、すなわち、残存確率（0から1の間）にQoSレベル（ブロンズ要求で1、シルバー要求で2、またはゴールド要求で3）をかける。したがって、0.75の残存確率を有する要求A（たとえばブロンズ要求）は、 $0.75 \times 1 = 0.75$ のユーティリティ値を有する。この要求は、キューによってディスパッチされる。というのは、ユーティリティ値が、閾ユーティリティ値（このキューについては0.5）を超えるからである。同様に、図3に示された要求Bもディスパッチされる。しかし、0.48のユーティリティ値を有する要求C（0.16の完了確率を有するゴールド要求（QoS=3））は、閾ユーティリティ値（0.5）の方が高いので、転送されない。修正された閾ユーティリティ値は、ロード・バランスのディスパッチ速度を変更する修正処置が講じられると仮定することによって計算される。この訂正処置のステップは、図4で説明する。要求C'が、修正された要求である。図3から、ディスパッチ速度を修正する訂正処置が講じられる場合に、完了確率が0.20に増える。しかし、その処置に関連するコスト要因（たとえば0.9）がある。したがって、要求C'の修正されたユーティリティ値は、 $0.2 \times 3 \times 0.9 = 0.54$ になり、これは、閾ユーティリティ値0.5より大きい。したがって、訂正処置を講じると、要求Cのユーティリティ値が増え、これによって、要求Cを除去する必要がなくなる。

【0049】

また図3を参照して、要求Dの場合（完了確率=0.4、QoSレベル=1（ブロンズ））を検討されたい。第1の処置を講じたと仮定した後であっても、修正されたユーティリティ値（0.45）は、閾ユーティリティ値を超えない。しかし、第2の処置（スケール・アップ）を講じたと仮定する。この処置に関連するコストは、第1の処置のコスト（0.9の乗算要因）より高い（0.8の乗算要因）。したがって、修正されたユーティリ

10

20

30

40

50

ティ値は、 $0.6 \times 1 \times 0.8 = 0.48$ になる。しかし、この場合に、閾ユーティリティ値も、 0.45 に減る（スケール・アップに起因して）。したがって、第2の訂正処置を講じる場合には、要求Dを渡すことができる。しかし、要求Eなど、両方の訂正処置を講じた場合であっても渡されない要求が存在する可能性がある。そのような要求は、システムによって除去される。

【0050】

図4に、キューがロード・バランサ内で実施され、ロード・バランサのディスパッチ速度を修正した後に要求が渡せるようになる、例示的な例でのステップのシーケンスを示す。やはり、要求を渡せるようにする多数の形があり、以下の説明では、その1つの可能な形を示す。この処置で使用する情報には、次レベル・ロード・バランサおよびそれに対応するクラスタの負荷に関する情報が含まれる。ステップ402で、キューが、特定の要求クラスrの要求を受け取る。ステップ404で、キューの実行可能モジュールが、要求クラスrに関する要求の現在の着信速度、要求クラスrに関する着信要求の期待される速度、および次レベル・ロード・バランサからの要求クラスrの要求の現在のディスパッチ速度に関する情報を受け取る。要求の完了の確率は、この受け取られた情報をキューの実施モジュールで処理することによって判定される。この確率は、多数の形で計算することができる。確率を計算する例示的な形の1つを、ステップ410に関して説明する。

【0051】

ステップ406で、キューが、要求クラスに関する情報を収集する。この情報は、着信要求のヘッダに格納されることが好ましい。要求のQoS／SLAレベルを、要求クラスの例の1つとすることができる。単純な実施の例を、ゴールド要求に3、シルバー要求に2、ブロンズ要求に1の値を割り当てることとすることができる。ステップ408で、キューが、ステップ404および406で収集した情報に基づいて、要求のユーティリティ値を計算する。やはり、前に述べたように、この情報をキューによって使用してユーティリティ値を得る多数の形が存在する可能性がある。前に説明したように、それを行う単純な例の1つが、完了の確率と要求クラス・レベルの乗算によるものである。したがって、次レベル・ロード・バランサの負荷が75%（完了確率=0.25）である場合に、ブロンズ要求のユーティリティ値は、0.25（ゴールド要求のユーティリティ値は0.75、シルバー要求のユーティリティ値は0.5）である。

【0052】

ブロンズ要求のユーティリティ値が、閾ユーティリティ値より小さいと仮定する。その場合には、修正されたユーティリティ値は、キューの要求ディスパッチ速度の変化を仮定することによって計算される。ステップ410で、要求ディスパッチ速度の変化を、収集した情報を使用してキューの実施モジュールによって計算する。ディスパッチ速度の計算には、どのような技法でも使用することができる。以下は、特定の要求クラスの残存確率に基づいてディスパッチ速度の変化を計算する単純な例示的な形である（この例では、要求の優先順位（要求クラス）を考慮に入れていない。要求クラスの考慮ありまたはなしでディスパッチ速度を計算する多数の方法があることは、当業者に明白である）。最後に、ステップ412で、要求ディスパッチの計算された新しい速度で要求をディスパッチする。

【0053】

$Q = \{Q_2, Q_3, Q_4, \dots, Q_k\}$ の前レベル・キューを有するキュー Q_m を検討されたい。 Q_m のバッファが、要求クラス r のクリティカル値（たとえば95%満杯）より大きい場合には、キュー Q_m は、 Q 内のキューのフロー制御メッセージを生成すると判断する。このフロー制御メッセージには、 Q_m でのクラス r の着信要求速度 R_r^m が含まれる。期待される着信速度は E_r^m である。受取り側のキュー Q_k は、このメッセージを処理し、そのディスパッチ速度を訂正すると判断する。 Q_k によって実施される処置は、ディスパッチ速度を下げることである。最終的なディスパッチ速度 d_r^k を計算するのに使用される式の例は、下記とすることができる。

D_r^k が、CLD-kからのクラス r の要求の現在のディスパッチ速度であるものとす

10

20

30

40

50

る。

着信速度への寄与の割合は、 $C = D_r^k / R_r^m$ である。

ディスパッチ速度に対する変更は、 $z_r^k = C \times (E_r^m - R_r^m)$ である。

新しいディスパッチ速度は $d_r^k = D_r^k + z_r^k$ である。

$d_r^k < D_r^k$ の場合には、 $Q - k$ は、要求クラス r のディスパッチ速度を d_r^k まで減らす。これは、ディスパッチされる要求のそれぞれの後に遅延を追加して、新しいディスパッチ速度を実施することによって達成される。

【0054】

開示されるモデル（システムおよび方法）は、ある事前定義のパラメータ（サービス品質／サービス・レベル・アグリーメント（QoS／SLA）など）に基づいて要求を分類し、処理する能力も有する。本発明を使用することによって、システムで発生する要求除去の数が大幅に減る。また、システム・リソース使用は、コンピュータ・システム内の中間コンポーネントで要求が除去される機会が減るので、はるかに効率的になる。本発明の一実施形態では、厳密なサービス品質／サービス・レベル・アグリーメント（QoS／SLA）の実施が保証され、これによって、サービス品質の改善がもたらされる。これによって、ユーザ要求に対するより高速の応答時間が保証される。

10

【0055】

これまでに説明したシステムのハードウェア実施形態は、下記の形で行うことができる。各キューを、集積回路によって実施することができる。バッファは、DRAM技術または、より高速のメモリ動作のためにキャッシュ・メモリを実施するのに使用される技術を使用して実施することができる。フロー制御メッセージを処理するサブコンポーネントに、マイクロプロセッサ、メモリ、および通信プロセッサも含めることができる。処理機能は、多少のメモリを有するマイクロプロセッサおよび通信プロセッサとして実施することもできる。バッファ、処理機能、およびコントローラの間に、通信リンクがある。各サブコンポーネントは、所望の構成および挙動のために特定の命令を介してプログラムすることができる。

20

【0056】

説明したシステムは、ソフトウェア実施形態も有することができる。これは、メモリおよび処理能力を有する計算装置で実施することができる。各コンポーネントは、一般的なプログラミング言語（C、C++、またはJava（R）など）のいずれかを使用して記述されるソフトウェア・コードの形で実施される。これらのソフトウェア・コードによって、実行時に、コンポーネントの機能性を実行するために、計算装置のメモリおよび処理能力が使用される。

30

【0057】

これまでに述べたモデルは、図5に関して説明する連繋負荷ディストリビュータ（CLD）アーキテクチャで実施することができる。このモデルは、複数のそのようなCLD（図5で説明する）を使用する連繋負荷分配フレームワークでも実施することもできる。そのようなフレームワークを、図6に関して詳細に説明する。

【0058】

実施形態1：連繋負荷ディストリビュータ（CLD）

40

これまでに説明したモデルのCLDとしての実施形態は、 n レベルのキューを有し、 $n \geq 1$ である。最初の $(n - 1)$ 個のレベルのどれにも、クラスタは関連付けられない。これらのレベルの各キューの処理機能は、要求分類機能である。 n 番目のレベルは、キューごとに1つまたは複数のクラスタを有し、これらのレベルの処理機能は、要求ディスパッチ機能である。

【0059】

これまでに説明した実施形態を、例を使用して説明する。図5に、連繋負荷ディストリビュータ（CLD）500の全体的なシステム・アーキテクチャの例を示す。CLD500には、2レベルのキューすなわち、第1レベル・キュー（グローバル要求ハンドラ（GRH）502）および第2レベル・キュー（特定要求ハンドラ（SRH）504）が含ま

50

れる。GRH502は、クラスタを関連付けられず、GRH502の処理機能は、要求分類機能である（要求クラシファイヤ（RC）とも称する）。RCは、要求分類要因（RCF）に基づいて要求を分類し、RCFは、ある事前定義のパラメータ（後で説明するように、サービス・ベースの分類または顧客ベースの分類など）に依存するものとしてすることができる。GRH502から分類された要求を受け取る複数のSRH（第2レベル・キュー、SRH504に似る）がある。これらのSRHは、要求のさまざまなクラスに対応するものとしてすることができる。図示の例では、各SRHが、要求ディスパッチ機能（要求ディスパッチャ（RD）とも称する）を有する。各SRHは、1つまたは複数のクラスタを有することができ、そのクラスタに、SRHが要求をディスパッチする。本発明のCLD実施形態を、2レベルのキューに関して説明したが、複数レベルのキューも使用できること
10
に留意されたい。たとえば、第2レベル・キューでの要求の副分類を設けることができる。3つのレベルを有するロード・バランサの例示的实施形態は、グローバル・バッファとして働く1つの第1レベル・キュー、異なるサービス・クラスの複数の第2レベル・キュー、および異なる顧客クラスの複数の第3レベル・キューを有することができる。

【0060】

CLD500は、感知モジュール、交換モジュール、および実施モジュールの機能性を実行する、要求ディスパッチ・マネージャ（RDM）506および負荷分配コントローラ（LDC）508を有する。この実施形態では、交換モジュール、感知モジュール、および実施モジュールが、キュー固有として図示されていない、すなわち、各キューが、この3つのモジュールを有しないことに留意されたい。その代わりに、この例示的CLDアー
20
キテクチャでは、これらのモジュールの機能性を実施する、1つのRDMおよび1つのLDCだけがある。RDM506は、SRH504の負荷状態のセンサとして働き、GRH502とSRH504の間のメッセージの交換を可能にする。RDM506は、SRHがほぼ満杯の状態（後で詳細に説明する）に達した時に、GRHに適当なメッセージを送る。RDM506は、各複製に関する動的負荷情報を得るために、負荷モニタ（LM）とも情報を交換する。LDC508は、他のCLDとの間で、メッセージを受け取り、生成し、送る（図6に関して説明する）。LDC508は、リソース・マネージャ（RM）とも情報を交換する。

【0061】

CLDアーキテクチャを、これから詳細に説明する。図5に関して既に述べたように、
30
CLD500は以下のコンポーネント、グローバル要求ハンドラ（GRH）502、要求ディスパッチ・マネージャ（RDM）506、特定要求ハンドラ（SRH）504、および負荷分配コントローラ（LDC）508を有する。GRH502には、バッファ、要求レシーバ、および要求分類機能が含まれる。RDM506には、LDC508およびGRH502からのメッセージのレシーバと、SRH504、LDC508、およびGRH502へのメッセージのセンダが含まれる。RDM506によって、SRH504とGRH502の間のメッセージ510の交換が可能になる。RDM506には、任意選択として、負荷モニタに関する負荷レシーバおよびメッセージ・センダも含まれる。SRH504には、要求レシーバ、第2レベル・バッファ（SLB）と称するバッファ、および要求ディスパッチ機能が含まれる。LDC508には、新しい要求クラスに基づくか、クラスタ
40
容量などの変更に基づいて、動的にCLDの構成を修正するモジュールが含まれる。LDC508には、QoS/SLA関係のインテリジェンスを実施する任意選択のモジュールも含まれる。たとえば、LDC508は、満杯になる頻度、要求除去の割合などに基づいて、QoSクラスごとのバッファ・サイズを判断することができる。

【0062】

GRH502は、第1レベル・キューであり、SRHの組は、第2レベル・キューからなる。SRHの数は、要求クラスの数によって左右される。好ましい実施形態では、その数が、要求クラスの数と等しい。GRH502は、主に、着信要求の受け取りの責任を負い、任意選択として、要求/他の定義された判断基準の正当性の認証に基づいて管理制御
50
を実行し、要求を受け入れ、各要求を分類し、要求のクラスに適当なSRHに要求を送る

。各SRH504は、主に、GRH502から要求を受け取り、バッファが満杯でない場合にその要求をバッファにバッファリングし、満杯の場合には要求を選択的に除去し（前にモデルの説明で既に説明したバッファ管理ポリシーに従って）、使用されるロード・バランシング技法に基づいて各要求をディスパッチする責任を負う。GRH502およびRDM506は、GRH502および各SRH504の間のフロー制御を実施するために、フロー制御を実行しなければならない。RDM506は、SRH504およびSRH504のそれぞれのディスパッチ速度の管理を実行する責任を負う。

【0063】

GRH502は、要求を受け取る場合に、バッファが満杯であるか否かを検査する。満杯の場合には、LDC508が、ユーティリティ値に基づいて要求を選択的に除去するか、要求を受け入れるようにGRH502に指示する。既に説明したように、実施されるバッファ管理ポリシーによって、要求がそのユーティリティ値に基づいて処理される形が左右される。要求を処理する例示的な形は、キューが、閾ユーティリティ値より高いユーティリティ値の要求を受け取り、他の要求を除去できることとすることができる。また、次レベル・キューに転送できない（ユーティリティ値が低いことに起因する）要求をハッシュテーブルに編成することができ、その結果、これらの要求を、次レベル・キューが要求を受け取る準備ができるとすぐに転送することができる。要求を受け入れた後に、GRH502は、要求をグローバル要求バッファにバッファリングする。GRH502の要求分類機能が、バッファ内の要求を分類し、そのRCFに基づいて正しいSRHにディスパッチする。アプリケーションレベルの要求を分類するために（ロード・バランサがアプリケーション・レベルのロード・バランシングを実行している場合）、GRH502の要求分類機能が、要求分類要因（RCF）を判定するために、要求の要求ヘッダを解析する。ロード・バランサが、ネットワーク・レベルのロード・バランシングを行っている場合に、GRH502の要求分類機能は、優先順位または他の情報あるいはその両方（どの種類のRCFが使用されるかに依存する）についてIPパケット・ヘッダを解析し、RCFを判定する。

【0064】

RCFを、GRH502によって計算することができる。RCFに基づいて、GRH502は、そのRCF値に関連する次レベル・バッファBの状態を検査する。SRH504のバッファが満杯の場合には、GRH502のバッファ管理ポリシーによって、要求を処理する方法が判断される。LCD508によって、SRH504に、要求のユーティリティ値に基づいて、受け取った要求の一部を除去するように指示することもできる。そうでなければ、RCFを判定した後に、GRH502の要求分類機能が、要求を適当なSRHに引き渡す。

【0065】

SRH504は、要求をキューにバッファリングする。同時に、SRHのRDが、ターゲット・クラスタ内の各複製の負荷情報を受け取る。ロード・バランシング技法が静的な技法（たとえばラウンド・ロビン・ロード・バランシング技法）である場合には、負荷情報は不要である。しかし、動的なロード・バランシング技法の場合には、要求ディスパッチ機能が、特定の要求クラス（そのSRHが属する）のクラスタの各複製の負荷情報を必要とする可能性がある。この負荷情報は、負荷モニタから受け取られる。要求ディスパッチに、2つの可能性があることに留意されたい。まず、要求ごとに、ターゲット・クラスタ内の各複製の負荷情報が入手され、対応するユーティリティ値が計算される。しかし、要求密度が高いネットワークでは、要求ディスパッチ速度が、非同期フロー制御メッセージを得る速度よりもあまりに高い。その場合には、負荷情報を、事前定義の基礎で受け取る。たとえば、この情報を、関係するクラスタ内の複製の負荷に大きい変化がある時だけ受け取ることができる。その場合の要求ディスパッチ速度は、使用可能な負荷情報に基づいて計算される。クラスタの負荷に大きい変化がある時に限って、新しい負荷情報が、入手され、ユーティリティ値の計算に使用される。説明した実施形態では、前者の形の要求のディスパッチを説明した。しかし、後者の形および多数の他の形を使用して、本発明の

10

20

30

40

50

趣旨および範囲から逸脱せずこの目的を達成できることは、当業者に明白であろう。

【0066】

好ましい実施形態では、受け取った負荷情報に基づいて、RDが、負荷が最小の複製を識別し、その複製が正の残余容量を有するかどうか、すなわち、その複製が、さらに要求を処理できるかどうかを識別する。その後、RDは、バッファから要求を検索し、その（最小の負荷の）複製にディスパッチする。RDは、ディスパッチ速度に基づいてディスパッチを継続する。

【0067】

各SRH504のRDは、使用されるロード・バランシング技法に基づいて、ターゲット複製（クラスタ）に要求をディスパッチする。要求ディスパッチ機能は、要求をターゲット複製にディスパッチする方法を知る必要がある。使用される要求ディスパッチ技法は、ネットワーク・レベルとアプリケーション・レベルのロード・バランシングで異なるので、要求ディスパッチ機能も、それぞれに別々に実施することができる。アプリケーション・ロード・バランシングの場合に、要求ディスパッチは、アプリケーションのそれぞれの使用するプロトコルに応じて、アプリケーションごとに異なる。したがって、アプリケーションのクラスについて、要求ディスパッチ機能の1つの実施形態だけがある。

【0068】

GRH内のRCおよびSRH内のRDは、CLDがアプリケーション・レベル・ロード・バランシングまたはネットワークレベル・ロード・バランシングのどちらに使用されるかに依存するので、これらを、ロード・バランサのプラグ可能コンポーネントとして実施することができる。そのようなプラグ可能コンポーネントは、スタートアップ中にロード・バランサによって使用される。このコンポーネントは、アプリケーションレベル・ロード・バランシングについてアプリケーション依存である。

【0069】

RDM506は、各SRH504の負荷の状態を追跡する。SRHが、ほぼ満杯（たとえば、75%満杯、これはユーザが定義することができる）になる場合に、RDM506は、フロー制御メッセージをGRH502に送る。ほぼ満杯状態は、必ずバッファの満杯状態以下である。GRH502がフロー制御を実施する時まで、GRH502が、特定の要求クラスについてさらに要求を送っている可能性があるので、ほぼ満杯状態は有用である。フロー制御メッセージには、とりわけ、要求クラスと、その要求クラスの着信要求速度が含まれる。GRH502は、要求の転送を中止するか、要求をディスパッチする速度を減らすか、他の手段を講じることができ、これは、モデルに関して前に説明した。SRH状態が、「ほぼ満杯」状態から「ほぼ満杯でない」状態に戻る場合に、RDM506は、状態変化に関するメッセージをGRH502に送る。GRH502は、前の状態に戻ることができる。GRH502のバッファがほぼ満杯になるか、GRH502が、あるSLBがより頻繁にほぼ満杯になるのを観察するか、GRBが満杯なのでGRH502が要求を除去しているか、着信要求速度が、SRH504にディスパッチする速度よりはるかに高いか、類似する状況が生じる場合に、GRH502は、その状態に関する制御メッセージをLDC508に送る。このメッセージには、関連するデータが含まれ、着信要求速度、要求分類速度、SRHの要求ディスパッチ速度などを含めることができる。LDCは、GRB/SLBのバッファ・サイズを増やすか、フロー制御メッセージを前のレベルのロード・バランサに送るか、その両方を行うかを判断する。

【0070】

LDC508は、要求クラスrのフロー制御メッセージを受け取る時に、rのディスパッチ速度をどれほど減らすか、rの要求のディスパッチをどれだけの時間だけ停止するか、要求クラスrのGRHおよびSRHのバッファ・サイズをどれだけ増やすか（この前の2つの判断に起因してバッファリングされる要求を処理するために）を判断する。LDC508は、rの最新の要求ディスパッチ速度を知らない場合に、その値に関する制御メッセージを介してRDMに尋ねることができる。LDC508は、値および判断を計算し、最終的な要求ディスパッチ速度、要求ディスパッチを停止する時間、またはSLBサイズ

10

20

30

40

50

の増加、あるいはこれらの組合せを実施するようにRDMに指示する。同様に、GRH502についてこれに関する指示がある場合には、LDC508は、GRH502にも指示する。RDM506は、指示に従ってSRH504のディスパッチ速度を制御する。RDM/GRHは、必要な場合に、それぞれSLB/GRBのバッファ・サイズを増やす。GRH502およびRDM506は、指示の成功裡の実施についてLDC508に肯定応答を送り、そうでない場合には、否定応答がLDC508に送られる。

【0071】

リソース・マネージャのメッセージに起因して、または他の理由に起因して（SLBまたはGRBの頻繁なオーバーフローに対抗するなど）、LDC508が、バッファのいずれかのサイズを変更する（増やす／減らす）と判断する場合がある。GRBのサイズを変更するために、LDC508は、GRH502に指示する。GRH502は、指示を実施し、成功裡の変更の際には肯定応答、そうでない場合には否定応答を送る。SLBのサイズを変更するため、またはSRHのディスパッチ速度を変更するために、LDC508は、特定の要求クラスに必要な処置に関して知らせることによって、RDM506に指示する。RDM506は、指示の処置を実施し、成功裡の変更の際には肯定応答、そうでない場合には否定応答を送る。

10

【0072】

リソース・マネージャは、新しい要求クラスrを追加する場合に、その更新に関してCLDのLDC508に通知する。更新には、可能なQoS/SLAパラメータ、または可能な最大要求速度、あるいはその両方も含まれる。LDCは、rの新しいSRHを作成し、要求クラスのリストにrを追加するようにRDMに指示する。RDMは、LDCによって指定されるSLBサイズのSRHを作成する。次に、RDMは、LDCにメッセージを送り、成功裡のSRH作成について知らせる。RDMは、SRHを作成できない場合に、LDCに否定の応答を送る。RDMからの肯定の応答を受け取る時に、LDCは、GRBサイズの変更および新しい要求クラスrの追加についてGRHに指示する。GRHは、GRBサイズを増やし、要求クラスのリストを更新することによって、LDC指示を実施する。GRHは、成功裡の完了時に肯定応答を送る。GRHは、失敗した場合に、否定応答を送る。

20

【0073】

GRH/LDC/RDMでのなんらかの種類の障害によって、ロード・バランサが効果的にロード・バランサに要求クラスrを追加できない場合には、LDCは、リソース・マネージャに否定応答を送ることができる。

30

【0074】

同様に、要求クラスrの削除について、LDCは、対応するSRHを削除するようにRDMに指示する。RDMは、そのような指示を受け取る時に、「要求rのSRHを削除せよ」という制御メッセージをGRHに送る。次にGRHは、外部からrに関する要求をそれ以上受け取るのを停止し、rのすべての要求をSRHに転送する。GRHは、肯定応答をRDMに送る。GRHからの肯定応答を受け取る時に、RDMは、rのSLBが空になる（ディスパッチされる要求がなくなる）まで待ち、その後、SRHを削除する。RDMは、その後、LDCに肯定応答を送る。LDCは、GRBのサイズを変更し、適宜レコードから要求クラスrを削除するようにGRHに指示する。GRHは、GRBのサイズを変更し、リストを更新し、その後、肯定応答をLDCに送る。GRHとRDMの両方から肯定応答を受け取る時に、LDCは、必要であれば、「要求クラスrが除去された」ことを含む肯定応答をRMに送ることができる。GRH/LDC/RDMでのなんらかの種類の障害によって、ロード・バランサが効果的に要求クラスrを削除できない場合には、LDCは、リソース・マネージャに否定応答を送ることができる。

40

【0075】

各要求クラスの要求は、他の要求クラスの要求と並行にディスパッチされる。CLDは、2つのディスパッチャが一時に同一のインスタンスに要求を送らないことを保証するために、平行ディスパッチを使用するが、各論理インスタンスは、1つのリソース・クラス

50

だけに属さなければならない。論理インスタンスが、クラスタ内の、物理インスタンス／複製または物理インスタンス／複製のある容量のいずれかであることに留意されたい。たとえば、要求クラスがゴールドおよびシルバーである場合に、論理インスタンスは、ゴールドまたはシルバーのいずれかに属すが、両方には属さない。

【0076】

代替実施形態では、このアーキテクチャに順次ディスパッチ技法も使用することができる。平行列ディスパッチの代わりに、下記の技法を使用して順次ディスパッチを実施することができる。各SRHの要求ディスパッチャを削除する。これらの要求ディスパッチャの代わりに、すべての要求クラスに関する単一要求のディスパッチャを使用する。この要求ディスパッチャは、任意のパラメータ（たとえばQoS／SLAベースの重み）に基づいてさまざまなキューの間でスケジューリングを実行し、一時に1つの要求をディスパッチする責任を負う。

10

【0077】

実施形態1で説明したCLDでは、要求の処理に二重レベル・バッファリングを使用する。2レベルのバッファの間に、情報フローが存在する。これは、要求が第2レベルで受け入れられる確率を識別するのに役立つ。その結果、低い残存確率を有する要求が、第1レベルで除去され、これによって、システム・リソース（たとえば、メモリ可用性および処理能力）の浪費が減る。

【0078】

CLDによって、要求の区別されたロード・バランシングも実行され、これによって、QoS／SLAの厳密な実施がもたらされる。

20

【0079】

各SRHは、ターゲット・コンポーネントに直接に要求をディスパッチする。したがって、要求の平行ディスパッチが行われる。要求が、ターゲット・コンポーネントに最終的にディスパッチされる前に通過しなければならないボトルネックはない。これによって、CLDのスループットが高まる。

【0080】

CLDで除去される要求の数は、システム・リソースがより効率的に使用されるので減る。したがって、第2レベルに転送される要求は、高い残存確率を有するか、転送の高いユーティリティを有するものだけである。これは、CLDアーキテクチャのレベル間のフロー制御によって達成される。

30

【0081】

実施形態1：CLDは、オブジェクト指向技術を使用して実施することができる。たとえば、C++を使用して、システムを実施することができる。各モジュール／エンティティは関連するクラス、Buffer、Queue、GRH、SRH、LDC、RDM、RDを有する。SLBおよびGRBは、Bufferのサブクラスである。オブジェクト・メッセージングが、これらのクラスのオブジェクトの間で、通信を実行するのに使用される。CLDには、1つのGRH、複数のSRH、1つのRDM、および1つのLDCのオブジェクトが含まれる。各GRHに、オブジェクトGRBが含まれる。各SRHに、SLBおよびRDのオブジェクトが含まれる。各RDMが、各SRHとの関連関係を有する。各LDCに、前レベル・ロード・バランサおよび次レベル・ロード・バランサのリストが含まれる。RDMは、ロード・バランサとの関連を有する。

40

【0082】

もう1つの実施形態では、CLDが、ICまたは論理回路あるいはその両方であり、これに、マイクロプロセッサIntel x86としてのLDC、別のマイクロプロセッサIntel x86としてのRDM、Intel x86としてのRD、GRBのDRAMメモリを有するGRHが含まれる。SLBも、DRAMメモリとして実施される。要求を受け取り、要求を送り、フロー制御メッセージを通信する入出力に標準ネットワーク・カード／インターフェースを使用することができる。ロード・バランサの間の通信のプロトコルを、TCP/IPとすることができる。

50

【0083】

実施形態2：連繫負荷バランシングのフレームワーク

図1に関して説明したモデルは、マルチレベル処理システム（フレームワーク）で実施され、クラスタの前の各ロード・バランサ（スケジューラ）を、キューとしてモデル化することができる。そのようなキューの処理機能は、ディスパッチ機能である。各キューに関連する1つまたは複数のクラスタがありえる。感知モジュール、実施モジュール、および交換モジュールの機能性を、各ロード・バランサ（CLD500）のLDC508によって実行することができる。図5に関して上で説明したCLD500は、そのようなマルチ・レベル処理システム（フレームワーク）で、図6に示されているように使用される。

【0084】

10

ここで、このフレームワークが使用される環境を短く説明する。連繫負荷バランシング・フレームワークを使用する集中システムに、システムの複製の組が含まれる。これらの複製は、着信要求について負荷が平衡化される。連繫する負荷バランシング・フレームワークを使用する分散システムに、そのコンポーネントのクラスタが含まれる。システム全体の複製は、使用されない。コンポーネントの複製のクラスタが使用される。各クラスタに、1つのコンポーネントの複製が含まれる。クラスタに関連するロード・バランサが、そのクラスタの複製の間で負荷を分配する。中間クラスタでの要求除去を減らすために、ロード・バランサの間でクラスタにまたがってフロー制御が使用される（図6）。

【0085】

連繫負荷バランシング・フレームワーク（図1で説明したモデルに基づく）の例を、図6に関してこれから説明する。本発明の趣旨および範囲から逸脱せずにCLD500を使用する連繫負荷バランシング・フレームワークの多数の変形形態がありえることに留意されたい。

20

【0086】

このフレームワークで使用されるCLDは、感知モジュール、実施モジュール、および交換モジュールの機能性を実行するために、CLD500のアーキテクチャを有する。この例のフレームワークには、それぞれクラスタ604、クラスタ610、クラスタ612、およびクラスタ616のコンポーネントに要求を分配するCLD602、CLD606、CLD608、およびCLD614（それぞれがCLD500のシステム・アーキテクチャを有する）が含まれる。情報（またはメッセージ）618が、CLD606およびCLD602の間で交換され、情報620が、CLD608とCLD602の間で交換される。同様に、情報622が、CLD614とCLD606の間で交換され、情報624が、CLD614とCLD608の間で交換される。

30

【0087】

ロード・バランシング・フレームワークへの要求は、CLD602に進み、クラスタ604のフロントエンド・コンポーネントの1つに分配される。フロントエンド・コンポーネントによる処理の後に、要求は、その要求の後続処理要件に応じて、CLD606（サーバ・アプリケーションAのクラスタ610を制御する）またはCLD608（サーバ・アプリケーションBのクラスタ612を制御する）のいずれかに進む可能性がある。この処理の後に、要求が、CLD614に送られ、CLD614によって、要求の処理を完了するために、要求が、クラスタ616内のデータベースの間で分配される。さまざまなクラスタへの要求のフローは、CLD間の情報のフローによって左右される。

40

【0088】

これから、CLD500を、上で説明した連繫負荷バランサ・フレームワークで使用方法を説明する。既に説明したように、各CLD500が、LDC508を有し、これによって、CLD500が、メッセージを生成し、フレームワーク内の他のCLDと交換できるようになる。したがって、この実施形態では、LDC508が、CLD500（キューとして働く）の感知モジュールならびに生成モジュールとして働く。LDC508は、負荷モニタ（LM）とも情報を交換する。前に説明したモジュール／コンポーネントのほかに、LDC508には、さまざまなコンポーネント、隣接するロード・バランサ、お

50

よび隣接するリソース・マネージャからのメッセージのレシーバと、そのようなコンポーネントへのメッセージのセNDER (sender) と、始動時のシステムの初期化のモジュール、クラスタ内の動的な修正に起因してCLDレコードを更新するモジュールも含まれる。LDC508は、新しい要求クラスに基づくか、クラスタの要求処理容量の変化に基づいてCLDの構成を動的に修正するモジュールも有する。手短に言えば、LDC508は、LDCシステム全体を管理し、ロード・バランサの間でクラスタにまたがってフロー制御を適用する責任を負う。

【0089】

GRH502のバッファがほぼ満杯になる（たとえば、前レベルCLDの高い要求ディスパッチ速度に起因する）場合に、GRHは、その状態に関してLDCに制御メッセージを送る。このメッセージは、関連するデータが含まれ、着信要求速度、要求分類速度、要求ディスパッチ速度などを含めることができる。LDCは、GRBのバッファ・サイズを増やす、または前レベルCLDにフロー制御メッセージを送る、あるいはその両方を行うかどうかを判断する。

10

【0090】

LDC508は、フロー制御メッセージを生成し、前レベル・ロード・ディストリビュータに送る時を判断する責任を負う。GRHがほぼ満杯である時に、LDCは、GRHにフロー制御メッセージを送ることができる。GRHは、必要な場合にGRBのバッファ・サイズを増やす。GRHは、指示を成功裡に実施した場合に肯定応答をLDCに送り、そうでない場合には、否定応答がLDCに送られる。フロー制御指示を成功裡に実施した時に、LDCは、要求クラスrの新しいディスパッチ速度をすべての次レベル・ディストリビュータに送る。そのようなCLDは、そのメッセージを受け取る時に、レコードを更新する。

20

【0091】

要求クラスrのクラスタ内のSLBの容量またはあるインスタンスの容量（要求速度に関する）の大きい増加がある場合に、LDCは、前レベル・ロード・バランサに、容量の増加、それが処理できる予測（容量増加に起因する）最大要求速度、rに関する現在のディスパッチ速度、rに関する現在の着信要求などについて通知する。LDCは、要求クラスrに関して前のロード・バランサにフロー制御メッセージを最近に送った場合に限って通知する。前レベルCLDのLDCは、そのようなメッセージを受け取るとすぐに、rに関する要求ディスパッチ速度の可能な増加の量dを計算する。その後、前レベルCLDのLDCは、GRHまたはRDMあるいはその両方に、rのディスパッチ速度を増やすように指示する。

30

【0092】

リソース・マネージャは、新しい要求クラスrを追加する場合に、CLDのLDCにその更新に関して通知する。更新には、可能なQoS/SLAパラメータまたは可能な最大要求速度あるいはその両方も含まれる。フロントエンド・クラスタへのクラスrの要求速度は、所与の点での要求クラスrのすべてのエンドユーザからの集合要求速度である。しかし、フロントエンド・クラスタのクラスrの各要求によって、次のクラスタに関する複数の要求が生成される場合がある。したがって、クラスタごとに、おそらくは異なる着信要求速度がある。LDCは、rに関する、GRBサイズに対する可能な変更およびSLBの可能なサイズに基づいて判断する。LDCは、まず、SLBへの適当な変更を行うようにRDMに指示する。RDMから肯定の応答を受け取る時に、LDCは、GRBサイズの変更および新しい要求クラスrの追加についてGRHに指示する。GRHは、GRBサイズを増やし、要求クラスのリストを更新することによって、LDC指示を実施する。GRHは、成功裡の完了の際に肯定応答を送る。GRHは、失敗する場合に、否定応答を送る。

40

【0093】

同様に、要求クラスrの削除について、LDCは、GRBサイズの可能な変更について判断する。LDCは、適当なSRHを削除するようにRDMに指示を送る。RDMから肯

50

定の応答を受け取る時に、LDCは、適当に、GRBのサイズを変更し、レコードから要求クラスrを削除するようにGRHに指示する。GRHは、GRBのサイズを変更し、リストを更新し、肯定の応答をLDCに送る。GRHおよびRDMの両方から肯定の応答を受け取る時に、LDCは、必要な場合に、「要求クラスrが削除された」ことを含む肯定応答をRMに送ることができる。GRH/LDC/RDMでのなんらかの種類の障害によって、ロード・バランサが効果的に要求クラスrを削除できない場合には、LDCは、必要な場合に、リソース・マネージャに否定応答を送ることができる。

【0094】

図7に示されているように代替実施形態では、クラスタ間ロード・バランサの間での集中フロー制御を使用することができる。この実施形態では、CLDごとにLDCを有するのではなく、集中的な形でフロー制御を実行する1つまたは複数の集中LDC702を設けることができる。この集中LDC702の重要な機能に、フロー制御が含まれる。リソース・マネージャ割振りプランの実施、RDを作成または削除するようにRDMに指示すること、GRHおよびRDMからの制御メッセージの受取、スタートアップ中のCLDの初期化を含むLDCの機能の一部を、各ロード・ディストリビュータにローカルなエンティティ（たとえば部分的なLDC）によって実行することができる。対応するロード・バランシング・フレームワークを、図7に示す。

【0095】

ロード・バランシングに関する提案されるフレームワークによって、クラスタごとの基礎でのロード・バランシングおよびクラスタ間ロード・バランサの間のフロー制御が容易になる。連繫負荷ディストリビュータ（CLD）のシステムが、クラスタ間ロード・バランサの間の分散フロー制御を実行する。連繫負荷ディストリビュータは、システムに存在する場合に、リソース・マネージャによるカスタマへのインスタンス容量の動的な割振りおよび割振り解除を考慮する。また、CLDは、システムに存在する場合に、負荷モニタによって収集される動的な負荷情報を使用して、負荷を効率的に平衡化する。CLDは、顧客ごとの現在の着信要求速度および顧客ごとの要求ディスパッチ速度、または容量割振り／割振り解除、あるいはその両方に基づいて、あるいは他のシステム状態に基づいて、前レベルCLDに制御メッセージを送るかどうかを動的に決定する。

【0096】

本発明の好ましい実施形態を図示し、説明したが、本発明が、これらの実施形態だけに制限されないことは明瞭である。請求項に記載の本発明の趣旨および範囲から逸脱しない多数の修正、変更、変形、置換、および同等物が可能であることは、当業者に明白であろう。

【0097】

まとめとして、本発明の構成に関して以下の事項を開示する。

【0098】

(1) マルチレベル・コンピュータ・システムで処理される要求のスケジューリングの方法であって、前記コンピュータ・システムの各レベルが、少なくとも1つのキューを有し、各キューが、少なくとも1つの処理機能を関連付けられ、

- 第1レベルの前記キューに前記要求をバッファリングするステップと、
- 前記第1レベルに隣接する他のレベルとフロー情報を交換するステップと、
- 前記要求の分類値を得るステップと、
- 前記得られた値に基づいて前記要求をスケジューリングするステップとを含む方法。

(2) スケジューリングする前記ステップが、前記要求をディスパッチするステップを含む、上記(1)に記載の方法。

(3) スケジューリングする前記ステップが、前記要求を分類するステップを含む、上記(1)に記載の方法。

(4) スケジューリングする前記ステップが、前記要求を除去するステップを含む、上記(1)に記載の方法。

10

20

30

40

50

(5) 前記得られた分類値が事前定義の値より大きい場合に、前記第1レベルで前記要求を処理するステップをさらに含む、上記(1)に記載の方法。

(6) 前記分類値を得る前記ステップが、

- a. 前記交換されるフロー情報に基づいてユーティリティ値を計算するステップと、
 - b. 前記計算されるユーティリティ値が事前定義の値より小さい場合に訂正処置を仮定するステップと、
 - c. 前記仮定される訂正処置を考慮した後の前記ユーティリティ値を再計算するステップと
- を含む、上記(1)に記載の方法。

(7) a. 前記再計算されるユーティリティ値が前記事前定義の値より大きい場合に、仮定された訂正処置を行うステップと、

- b. 前記仮定された訂正処置を行った後に、前記第1レベルで前記要求を処理するステップと
- をさらに含む、上記(6)に記載の方法。

(8) 訂正処置を仮定する前記ステップが、適当なキューが要求を受け入れることができない場合に、前記適当なキューの要求処理容量の増加を仮定するステップを含む、上記(6)に記載の方法。

(9) 訂正処置を仮定する前記ステップが、次レベルのキューが要求を受け入れることができない場合に、適当なキューからの要求ディスパッチ速度の減少を仮定するステップを含む、上記(6)に記載の方法。

(10) マルチレベル・コンピュータ・システムで処理される要求のスケジューリングに適するシステムであって、前記コンピュータ・システムの各レベルが、少なくとも1つのキューを有し、各キューが、少なくとも1つの処理機能を関連付けられ、

- a. 第1レベルと前記第1レベルに隣接する他のレベルとの間でフロー情報を交換する手段と、
 - b. 前記交換されるフロー情報に基づいて、前記要求の分類値を得る手段と、
 - c. 前記要求をスケジューリングする手段と
- を含むシステム。

(11) 前記スケジューリングする手段が、前記要求をディスパッチする手段をさらに含む、上記(10)に記載のシステム。

(12) 前記スケジューリングする手段が、前記要求を分類する手段をさらに含む、上記(10)に記載のシステム。

(13) 前記スケジューリングする手段が、前記要求を除去する手段をさらに含む、上記(10)に記載のシステム。

(14) 前記得られる分類値が事前定義の値より大きい場合に、前記第1レベルで前記要求を処理する手段をさらに含む、上記(10)に記載のシステム。

(15) 前記分類値を得る前記手段が、

- a. 前記交換されるフロー情報に基づいてユーティリティ値を計算する手段と、
 - b. 仮定された訂正処置を検討した後の前記ユーティリティ値を再計算する手段と、
 - c. 前記再計算されるユーティリティ値が事前定義の値より大きい場合に、前記仮定された訂正処置を行う手段と
- を含む、上記(10)に記載のシステム。

(16) 前記仮定された訂正処置を行う前記手段が、第2レベル・キューが余分な要求を受け入れることができない場合に、第1レベル・キューの要求ディスパッチ速度を減らす手段を含む、上記(15)に記載のシステム。

(17) 前記仮定された訂正処置を行う前記手段が、第2レベル・キューが余分な要求を受け入れることができない場合に、第2レベル・キューの要求処理容量を増やす手段を含む、上記(15)に記載のシステム。

(18) コンピュータ・システムで要求をスケジューリングするコンピュータ・プログラム製品であって、

10

20

30

40

50

- a. 第1レベルのキューに前記要求をバッファリングするプログラム命令手段と、
 - b. 前記第1レベルに隣接する他のレベルとフロー情報を交換するプログラム命令手段と、
 - c. 前記要求の分類値を得るプログラム命令手段と、
 - d. 前記分類値に基づいて前記要求をスケジューリングするプログラム命令手段を含むコンピュータ・プログラム製品。
- (19) コンピュータ・システムでのロード・バランシングに適する装置であって、
- a. 複数のキューであって、
 - i. 着信要求をバッファリングする第1レベル・キューと、
 - ii. 複数の後続レベル・キューであって、キューの各後続レベルが、着信要求のクラスに対応する、複数の後続レベル・キューとを含む複数のキューと、
 - b. ユーザ定義パラメータに基づいて、前記要求を複数の後続レベル・キューに分類する手段と、
 - c. 前記複数のレベルのキューの間で情報を交換する手段と、
 - d. 前記キューからの前記要求を前記キューまたはターゲット・コンポーネントの少なくとも1つにディスパッチする手段とを含む装置。
- (20) ディスパッチする前記手段が、要求の平行ディスパッチの手段を含む、上記(19)に記載の装置。 20
- (21) 情報を交換する前記手段が、前記キューからの要求のディスパッチ速度を制御する手段を含む、上記(19)に記載の装置。
- (22) 前記要求が、QoS/SLAおよび前記後続レベル・キューからの情報に基づいて前記第1レベル・キューで分類される、上記(19)に記載の装置。
- (23) コンピュータ・システムでのロード・バランシングのコンピュータ・プログラム製品であって、
- a. 複数のキューを生成するプログラム命令手段であって、前記複数のキューが、
 - i. 着信要求をバッファリングする第1レベル・キューと、
 - ii. 複数の後続レベル・キューであって、キューの各後続レベルが、着信要求のクラスに対応する、複数の後続レベル・キューとを含む、プログラム命令手段と
 - b. ユーザ定義パラメータに基づいて、前記要求を複数のクラスに分類するプログラム命令手段と、
 - c. 前記複数のレベルのキューの間で情報を交換するプログラム命令手段と、
 - d. 前記キューからの前記要求を前記キューまたはターゲット・コンポーネントの少なくとも1つにディスパッチするプログラム命令手段とを含むコンピュータ・プログラム製品。
- (24) コンピュータ・システムで要求をスケジューリングするのに適するシステムであって、
- a. 複数のクラスタと、 40
 - b. 複数のロード・バランサであって、各ロード・バランサが、少なくとも1つのクラスタに対応し、
 - i. 要求を受け取る手段と、
 - ii. 隣接レベル・ロード・バランサから情報を交換する手段と、
 - iii. 前記要求の分類値を得る手段と、
 - iv. 前記分類値に基づいて次レベル・クラスタに前記要求をディスパッチする手段とを含む、複数のロード・バランサとを含むシステム。
- (25) ネットワークで要求をスケジューリングする方法であって、前記ネットワークが、複数のロード・バランサを含み、 50

- a. 第1レベル・ロード・バランサで要求を受け取るステップと、
- b. 前記特定の要求に関する情報を隣接レベル・ロード・バランサと交換するステップと、
- c. 前記交換される情報を処理することによって前記要求の分類値を得るステップと、
- d. 前記得られる値に基づいて前記要求をスケジューリングするステップとを含む方法。

(26) 前記交換される情報を処理する前記ステップが、

- a. 前記要求の転送の閾ユーティリティ値を判定するステップと、
 - b. 前記要求の転送のユーティリティ値を得るステップと、
 - c. 前記ユーティリティ値と前記閾ユーティリティ値とを比較するステップと、
 - d. ユーティリティ値が前記閾ユーティリティ値より小さい場合に訂正処置を行うステップと
- を含む、上記(25)に記載の方法。

10

(27) ユーティリティの閾値を判定する前記ステップが、さらに

- a. 前記ネットワークがサービスを必要とするQoS要件およびSLA要件を識別するステップと、
 - b. 前記ネットワーク内で前記要求に割り振られるリソースを判定するステップと
- を含む、上記(26)に記載の方法。

(28) 前記ユーティリティ値を得る前記ステップが、

- a. 要求ヘッダから前記要求に係るQoSおよびSLAを識別するステップと、
- b. 次レベル・バランサで前記要求に対応する割り振られるリソースを判定するステップと、
- c. 前記ユーティリティ値を得るために前記識別された情報および前記判定された情報を処理するステップと

20

を含む、上記(26)に記載の方法。

(29) 訂正処置を行う前記ステップが、前記ロード・バランサでの要求ディスパッチ速度を下げるステップを含む、上記(26)に記載の方法。

(30) 訂正処置を行う前記ステップが、余分な負荷を受け入れるためにロード・バランサ容量を増やすステップを含み、ロード・バランサ容量を増やすステップが、前記要求の前記ユーティリティ値を増やす、上記(26)に記載の方法。

30

(31) ネットワーク内で要求をスケジューリングするコンピュータ・プログラム製品であって、

- a. 第1レベル・ロード・バランサで要求を受け取る第1プログラム命令手段と、
- b. 前記要求に関する情報を隣接レベル・ロード・バランサと交換する第2プログラム命令手段と、
- c. 前記交換される情報に基づいて前記要求の分類値を得る第3プログラム命令手段と、
- d. 前記分類値に基づいて前記要求をスケジューリングする第4プログラム命令手段とを含むコンピュータ・プログラム製品。

【図面の簡単な説明】

40

【0099】

【図1】本発明の好ましい実施形態による、要求を処理するシステムの例を示す図である。

【図2】本発明の好ましい実施形態による、要求を処理するために実施されるステップのシーケンスの例を示す図である。

【図3】ユーティリティ値の計算を示すのに使用される表の例を示す図である。

【図4】ロード・バランサの要求ディスパッチ速度を修正した後に要求が渡せるようになる、例示的な事例でのステップのシーケンスを示す図である。

【図5】要求除去の数を減らすための情報のフローという発想に基づく連繫負荷ディストリビュータ(CLD)の全体的なシステム・アーキテクチャを示す図である。

50

【図 6】図 5 で説明した C L D の間での情報のフローという発想に基づく連繋負荷バランシング・フレームワークを示す図である。

【図 7】ロード・バランサの間での情報のフローという発想に基づくもう 1 つの連繋負荷バランシング・フレームワークを示す図である。

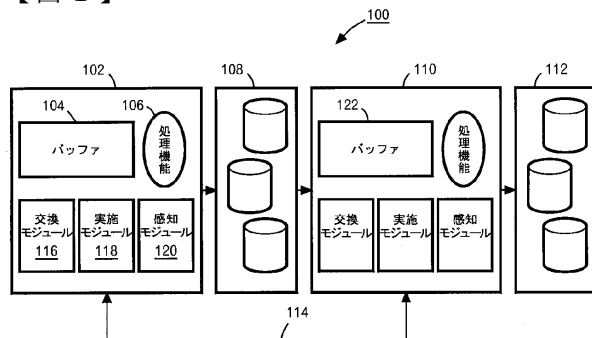
【符号の説明】

【 0 1 0 0 】

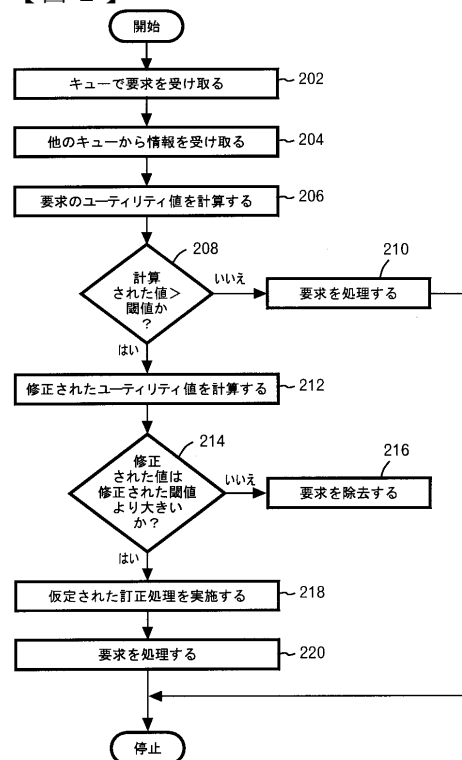
- 5 0 0 連繋負荷ディストリビュータ (C L D)
- 5 0 2 グローバル要求ハンドラ (G R H)
- 5 0 4 特定要求ハンドラ (S R H)
- 5 0 6 要求ディスパッチ・マネージャ (R D M)
- 5 0 8 負荷分配コントローラ (L D C)
- 6 0 2 C L D
- 6 0 4 クラスタ
- 6 0 6 C L D
- 6 0 8 C L D
- 6 1 0 クラスタ
- 6 1 4 C L D
- 6 1 2 クラスタ
- 6 1 6 クラスタ

10

【図 1】



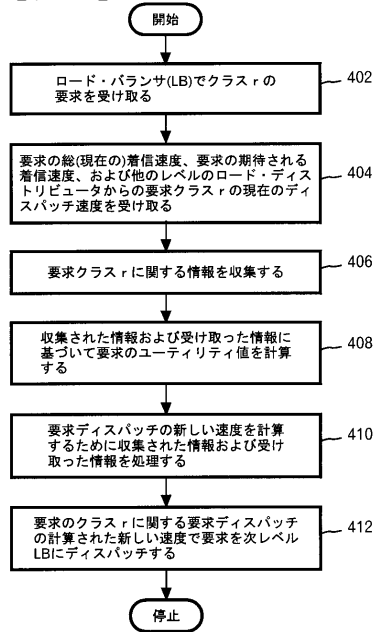
【図 2】



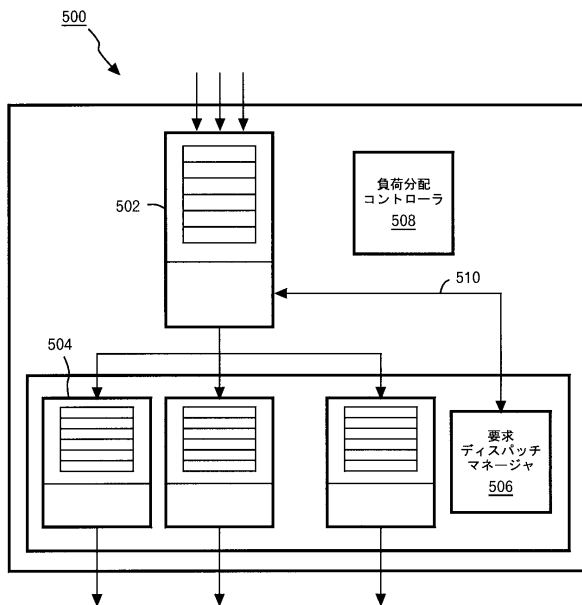
【図 3】

要求	残存確率	QoS レベル	コスト 要因	ユーティリティ値 または修正された ユーティリティ値	閾値または 修正された閾値
A	0.75	1		0.75	0.5
B	0.25	2		0.5	0.5
C	0.16	3		0.48	0.5
C'	0.2	3	0.9	0.54	0.5
D	0.4	1		0.4	0.5
D'	0.5	1	0.9	0.45	0.5
D''	0.6	1	0.8	0.48	0.45
E	0.25	1		0.25	0.5
E''	0.5	1	0.8	0.4	0.45

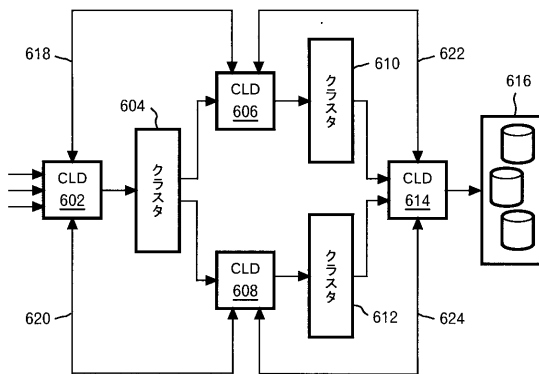
【図 4】



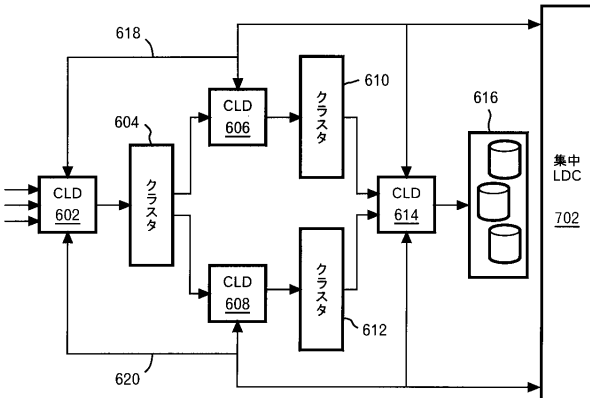
【図 5】



【図 6】



【図 7】



フロントページの続き

(72)発明者 アシシュ・クンドゥ
インド PIN-757 055 オリッサ州 マユルバンジ・ディストリクト マンガルプル局
止め ヴィア・アマルダ

審査官 鈴木 修治

(56)参考文献 特開平11-249914 (JP, A)
特表平08-511120 (JP, A)
特開2002-099435 (JP, A)

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