(19)日本国特許庁(JP)　(12)特　許　公　報(B2)　(11)特許番号
特許第3980487号
(P3980487)
(45)発行日　平成19年9月26日(2007.9.26)　(24)登録日　平成19年7月6日(2007.7.6)

(51)Int.Cl.　　F I
G06F 9/46 (2006.01)　　G06F 9/46 350
G06F 9/50 (2006.01)　　G06F 9/46 462Z

請求項の数 16　(全 26 頁)

(21)出願番号　特願2002-571989(P2002-571989)
(86)(22)出願日　平成14年2月1日(2002.2.1)
(65)公表番号　特表2004-530196(P2004-530196A)
(43)公表日　平成16年9月30日(2004.9.30)
(86)国際出願番号　PCT/GB2002/000456
(87)国際公開番号　WO2002/073397
(87)国際公開日　平成14年9月19日(2002.9.19)
審査請求日　平成15年10月3日(2003.10.3)
(31)優先権主張番号　09/801,993
(32)優先日　平成13年3月8日(2001.3.8)
(33)優先権主張国　米国(US)

(73)特許権者　390009531
インターナショナル・ビジネス・マシーンズ・コーポレーション
INTERNATIONAL BUSINESS MASCHINES CORPORATION
アメリカ合衆国10504　ニューヨーク州　アーモンク　ニュー　オーチャード　ロード
(74)代理人　100086243
弁理士　坂口　博
(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】区分処理環境におけるリソース平衡化

(57)【特許請求の範囲】

【請求項１】

第１のオペレーティング・システムを含む第１のパーティションと、第２のオペレーティング・システムを含む第２のパーティションとを有するコンピューティング・システムにおける方法であって、

ａ）前記第１のパーティションにおいて、前記第１のパーティションのネットワーク・トラフィック及び前記第１のパーティションのプロセッサの利用率を含む、前記第１のパーティションの第１スループット情報を収集するステップと、

ｂ）前記第１のパーティションにおいて、前記第１のパーティションの前記第１スループット情報に基づいて、前記第１のパーティションのプロセッサの累積利用率の変化に対する前記第１のパーティションの累積トラフィックの変化の割合である速度基準を計算するステップと、

ｃ）前記第１のパーティションから前記第２のパーティションのパーティション・マネジャに前記速度基準を伝達するステップと、

ｄ）前記パーティション・マネジャが、前記速度基準の値が、前記プロセッサの利用率が高く、かつ前記トラフィックが低いことを示す場合に、前記第１のパーティションに、少なくともプロセッサ、メモリ及びＩ／Ｏデバイスのいずれか１つを含むリソースをより多く割り振るよう指示するリソース平衡化指示を作成するステップと、

ｅ）前記リソース平衡化指示に従って、前記パーティション・マネジャにより前記第１のパーティションにリソースを割り振るステップと

を含む方法。
【請求項２】
前記パーティション・マネジャは、前記第２のパーティションで実行されるワークロード・マネジャ、およびハイパーバイザを備える、請求項１に記載の方法。
【請求項３】
パーティション間の通信は、パーティション間のメモリ共有を含む、請求項１に記載の方法。
【請求項４】
前記コンピューティング・システムは、ネットワーク・アドレス及びオフセットを物理アドレスに変換する共有Ｉ／Ｏアダプタを更に有し、前記第１のパーティションは、ソース・レジスタにより指定される物理アドレスからデスティネーション・レジスタにより指定されるの物理アドレスへデータを移動させるデバイス・ドライバを更に有し、
パーティション間の通信は、前記共有Ｉ／Ｏアダプタと前記デバイス・ドライバによるメモリからメモリへの直接のメッセージパッシングを含む、請求項１に記載の方法。
【請求項５】
前記コンピューティング・システムは、ネットワーク・アドレス及びオフセットを物理アドレスに変換し、かつ、ソース・レジスタにより指定される物理アドレスからデスティネーション・レジスタにより指定されるの物理アドレスへデータを移動させる共有Ｉ／Ｏアダプタを更に有し、
パーティション間の通信は、前記共有Ｉ／Ｏアダプタによるメモリからメモリへの直接のメッセージ・パッシングを含む、請求項１に記載の方法。
【請求項６】
前記第１のパーティションのネットワーク・トラフィックについての情報はパケット・アクティビティ・カウンタを利用して取得される、請求項１に記載の方法。
【請求項７】
前記第１のパーティションのネットワーク・トラフィックについての情報は、前記第１のパーティションに関連するネットワーク・パケットを計数することによって取得される、請求項１に記載の方法。
【請求項８】
前記第１のパーティションに関連するネットワーク・パケットとは、前記第１のパーティションにおいて受信されるパケットである、請求項７に記載の方法。
【請求項９】
前記第１のパーティションに関連するネットワーク・パケットとは、前記第１のパーティションにおいて送信されるパケットである、請求項７に記載の方法。
【請求項１０】
前記プロセッサの利用率はシステム・アクティビティ・カウンタを利用して取得される、請求項１に記載の方法。
【請求項１１】
第１のオペレーティング・システムを含む第１のパーティションと、第２のオペレーティング・システムを含む第２のパーティションとを備えるコンピューティング・システムであって、さらに、
ａ）前記第１のパーティションのネットワーク・トラフィック及び前記第１のパーティションのプロセッサの利用率を含む、前記第１のパーティションの第１スループット情報を収集する手段と、
ｂ）前記第１のパーティションの前記第１スループット情報に基づいて、前記第１のパーティションのプロセッサの累積利用率の変化に対する前記第１のパーティションの累積トラフィックの変化の割合である速度基準を計算する手段と、
ｃ）前記第１のパーティションから前記第２のパーティションのパーティション・マネジャに前記速度基準を伝達する手段と、
ｄ）前記パーティション・マネジャ中で、前記速度基準の値が、前記プロセッサの利用

率が高く、かつ前記トラフィックが低いことを示す場合に、前記第１のパーティションに、少なくともプロセッサ、メモリ及びＩ／Ｏデバイスのいずれか１つを含むリソースをより多く割り振ることを指示するリソース平衡化指示を作成する手段と、

ｅ）前記リソース平衡化指示に従って、前記パーティション・マネジャにより前記第１のパーティションにリソースを割り振る手段と

を備えるシステム。

【請求項１２】

前記パーティション・マネジャは、前記第２のパーティションで実行されるワークロード・マネジャ、およびハイパーバイザを備える、請求項１１に記載のシステム。

【請求項１３】

パーティション間の通信は、パーティション間のメモリ共有を含む、請求項１１に記載のシステム。

【請求項１４】

前記コンピューティング・システムは、ネットワーク・アドレス及びオフセットを物理アドレスに変換する共有Ｉ／Ｏアダプタを更に有し、前記第１のパーティションは、ソース・レジスタにより指定される物理アドレスからデスティネーション・レジスタにより指定されるの物理アドレスへデータを移動させるデバイス・ドライバを更に有し、

パーティション間の通信は、前記共有Ｉ／Ｏアダプタと前記デバイス・ドライバによるメモリからメモリへの直接のメッセージ・パッシングを含む、請求項１１に記載のシステム。

【請求項１５】

前記コンピューティング・システムは、ネットワーク・アドレス及びオフセットを物理アドレスに変換し、かつ、ソース・レジスタにより指定される物理アドレスからデスティネーション・レジスタにより指定されるの物理アドレスへデータを移動させる共有Ｉ／Ｏアダプタを更に有し、

パーティション間の通信は、前記共有Ｉ／Ｏアダプタによるメモリからメモリへの直接のメッセージ・パッシングを含む、請求項１１に記載のシステム。

【請求項１６】

第１のオペレーティング・システムを含む第１のパーティションと、第２のオペレーティング・システムを含む第２のパーティションとを有するコンピューティング・システム内で実行されるコンピュータ可読プログラムであって、前記プログラムは、

前記第１のパーティションを、

ａ）前記第１のパーティションのネットワーク・トラフィック及び前記第１のパーティションのプロセッサの利用率を含む、前記第１のパーティションの第１スループット情報を収集する手段と、

ｂ）前記第１のパーティションの前記第１スループット情報に基づいて、前記第１のパーティションのプロセッサの累積利用率の変化に対する前記第１のパーティションの累積トラフィックの変化の割合である速度基準を計算する手段と、

ｃ）前記第１のパーティションから前記第２のパーティションのパーティション・マネジャに前記速度基準を伝達する手段として機能させ、

前記パーティション・マネジャを、

ｄ）前記速度基準の値が、前記プロセッサの利用率が高く、かつ前記トラフィックが低いことを示す場合に、前記第１のパーティションに、少なくともプロセッサ、メモリ及びＩ／Ｏデバイスのいずれか１つを含むリソースをより多く割り振ることを指示するリソース平衡化指示を作成する手段と、

ｅ）前記リソース平衡化指示に従って、前記第１のパーティションにリソースを割り振る手段として機能させる、プログラム。

【発明の詳細な説明】

【技術分野】

【０００１】

（関連出願の相互参照）
本出願は、以下の同時係属米国特許出願に関連する。
Baskey他による米国特許出願番号０９／８０１４０７「INTER-PARTITIONMESSAGE PASSING METHOD, SYSTEM AND PROGRAM PRODUCT FOR THROUGHPUT MEASUREMENTIN A PARTITIONED PROCESSING ENVIRONMENT」（整理番号ＰＯＵ９２０００－０２００ＵＳ１）、
Baskey他による米国特許出願番号０９／８０２１８５「INTER-PARTITIONMESSAGE PASSING METHOD, SYSTEM AND PROGRAM PRODUCT FOR A SHARED I/O DRIVER」（整理番号ＰＯＵ９２０００－０２０２ＵＳ１）、および
Baskey他による米国特許出願番号０９／８０１４９２「INTER-PARTITIONMESSAGE PASSING METHOD, SYSTEM AND PROGRAM PRODUCT FOR A SECURITY SERVER IN APARTITIONED PROCESSING ENVIRONMENT」（整理番号ＰＯＵ９２００１－００１２ＵＳ１）。
【０００２】
本発明は、一般には、パーティションに区分されたデータ処理システムに関し、詳細には、システムのパーティション内で複数のオペレーティング・システム・イメージを実行することが可能なユニプロセッサ・システムおよびマルチプロセッサ・システムに関する。
【背景技術】
【０００３】
今日の中企業、大企業の大半は、かつては集中化された「温室」であったデータ・センターの到達範囲を自組織の範囲一杯まで、そして実際には範囲を越えて拡張するために各自のＩＴインフラストラクチャを発展させている。このような発展の推進力の元となるのは、部分的には、従来は本質的に異なっていた部門別の業務を相互に結びつけ、納入業者や顧客とリアルタイムで意思の疎通を図りたいという要望であり、そして、電子商取引を行う媒体としてのインターネットの急速な成長と、それに伴い、そのような接続性を得るための相互接続とビジネス間のソリューションを次第に多く利用できるようになったことによりさらに加速されている。
【０００４】
この近年の発展に伴うのは、今日の企業が、多くの異なる作動プラットフォームを動的にリンクして相互に接続されたシームレスなシステムを作り出す必要性である。諸企業には、集中化していない仕入れ業務、適用分野に応じた要件、および合併に関連する活動から生じる本質的に異なる技術プラットフォームの作成などの要因による、異種の要素を含む情報システム・インフラストラクチャを備えることを特徴とすることが多い。さらに、納入業者、提携業者、および顧客間を結ぶリアルタイムの企業外部の接続性を促進したいという要望から、異種の環境に接続性を提供しようとするさらに強い動機が生じる。
【０００５】
急速に増大する顧客の様々な要求に対して、情報技術の提供者は、企業のデータ・センターへのより高い接続性を求めるニーズに対応するデータ処理ソリューションの考案に着手している。
【０００６】
本明細書の主題に関連する背景情報には以下が含まれる。コンピュータ・システムのアクティビティ分析について記載する、Ruffin他による米国特許出願番号０９／１８３９６１「COMPUTATIONAL WORKLOAD-BASED HARDWARE SIZERMETHOD, SYSTEM AND PROGRAM PRODUCT」；論理パーティション間の共有メモリについて記載する、Temple他による米国特許出願番号０９／５８４２７６「INTER-PARTITIONSHARED MEMORY METHOD, SYSTEM AND PROGRAM PRODUCT FOR A PARTITIONED PROCESSINGENVIRONMENT」；高帯域幅の内蔵アダプタについて記載する、Baskey他による米国特許出願番号０９／２５３２４６「A METHOD OFPROVIDING DIRECT DATA PROCESSING ACCESS USING QUEUED DIRECT INPUT-OUTPUT DEVICE」；システム中の２つの異なるクライアント・サーバのパーティショニングについて記載する、Temple他による米国特許出願番号０９／５８３５０１「HeterogeneousClient Server Method, System and Program Product For A Partitioned ProcessingEnvironment」；複数

パーティションのワークロード管理について記載する、ＩＢＭ文書ＳＧ２４－５３２６－００「OS/390 WorkloadManager Implementation and Exploitation」ＩＳＢＮ：０７３８４１３０７０；および、ＥＳＡ／３９０命令セット・アーキテクチャについて記載するＩＢＭ文書ＳＡ２２－７２０１－０６「ESA/390Principles of Operation」。これらの文書は、参照により本明細書に組み込む。

【０００７】

当初、動作上の依存性を有する可能性がある各種のアプリケーションに対する処理サポートを同時に提供する統合システムを供給する必要性により、区分多重処理システムの市場が拡大した。単一の物理的なコンピューティング・システムで複数のオペレーティング・システム・イメージをサポートする能力を提供するこの区分システムは、かつてはメインフレーム・コンピュータ（ＩＢＭ　Ｓ／３９０システムなど）の分野でのみ用いられていたが、次第に広範囲の供給者から販売されるようになっている。例えば、Sun Microsystems社は最近、Ultra Enterprise 10000ハイエンド・サーバでシステム・パーティショニングの一形態を提供し始めた。これについては、１９９６年１２月１２日に出願され、１９９９年８月３日に発行され、SunMicrosystems社に譲渡された、Drogichen他による米国特許第５，９３１，９３８号「Multiprocessor Computer HavingConfigurable Hardware System Domains」に詳細に記載される。他社もこの種のシステムに対する関心を示す方向性を表明している。

【０００８】

この業界による採用は、１企業内の各種の計算ワークロードを１台（あるいは数台）の物理的なサーバ・コンピュータに統合する際に、そして動的に再構成することが可能なハードウェア環境でテスト・レベルおよび製造レベルのコードを同時に実装するために、システムの区分から得られる「systems within a system」の利点を明確に示すものである。さらに、前述の相互参照される特許出願で挙げたＩＢＭ　Ｓ／３９０コンピュータ・システムなど、特定の区分多重処理システムでは、そのシステムで行う作業に割り当てられた優先度に応じて、リソース（プロセッサ、メモリ、およびＩ／Ｏ）を論理パーティション内および論理パーティション間で動的に割り振ることができる（ＩＢＭおよびＳ／３９０は、InternationalBusiness Machines社の登録商標）。このワークロードの優先度に応じて動的にリソースを割り振る能力により、長年にわたる容量計画の問題が対処される。従来は、データ・センターの管理者が、一時的なワークロードの急増に対処するために予想される計算ワークロードに過剰な量のリソースを意図的に指定することによりこの問題に対応していた。

【特許文献１】米国特許出願番号０９／８０１４０７
【特許文献２】米国特許出願番号０９／８０２１８５
【特許文献３】米国特許出願番号０９／８０１４９２
【特許文献４】米国特許出願番号０９／１８３９６１
【特許文献５】米国特許出願番号０９／５８４２７６
【特許文献６】米国特許出願番号０９／２５３２４６
【特許文献７】米国特許出願番号０９／５８３５０１
【特許文献８】米国特許第５，９３１，９３８号
【特許文献９】米国特許第５，４４２，８０２号
【特許文献１０】米国特許出願番号０９／４１１４１７
【特許文献１１】米国特許第５，４１４，８５１号
【特許文献１２】米国特許第５，２６９，００９号
【特許文献１３】米国特許出願番号０９／６７７３３８
【非特許文献１】ＩＢＭ文書ＳＧ２４－５３２６－００「OS/390 WorkloadManager Implementation and Exploitation」ＩＳＢＮ：０７３８４１３０７０
【非特許文献２】ＩＢＭ文書ＳＡ２２－７２０１－０６「ESA/390Principles of Operation」
【非特許文献３】ＩＢＭ文書ＳＣ２８－１８５５－０６「"OS/390V2R7.0 OSA/SF User's

Guide」
【非特許文献４】Ｇ３２１－５６４０－００「S/390 clustertechnology: Parallel Sysplex」
【非特許文献５】ＩＢＭ文献ＳＣ３４－５３４９－０１「MQSeries QueueManager Clusters」
【非特許文献６】「Coupling Facility ConfigurationOptions: A Positioning Paper」(ＧＦ２２－５０４２－００,ＩＢＭ社)
【非特許文献７】IBM Redbook Document SG24-4810-01「UnderstandingRS/6000 Performance and Sizing」
【発明の開示】
【発明が解決しようとする課題】
【０００９】
こうした区分システムは、企業内の異種のシステムを取り込むようにデータ・センターを拡張することを容易にするが、現在、こうしたソリューションは、異種または同質の区分プラットフォームを相互運用性がある単一の区分システムに機能的に統合する単純な機構は提供しない。実際、これらの新しいサーバは、単一の物理的ハードウェア・プラットフォーム内でのオペレーティング・システム・イメージの統合は可能にするが、サーバの複数のパーティションに存在するオペレーティング・システム間の相互運用性に対するニーズには充分に対応していない。この相互運用性の問題はさらに、各種のパーティション内に異種のオペレーティング・システムを有する異種要素を含むシステムにおいて難しくなる。また、こうしたシステムは通例、そのような異種のプラットフォーム間でのパーティション間のリソース共有のタイプにも対応していない。パーティション間のリソース共有が実現すると、パーティション間の高帯域幅、低待ち時間の相互接続が可能になろう。これらの問題に対するシステム統合のソリューションにより、別個のパーティションで実行されるプロセス間の通信を行うためのより堅固な機能を可能にして、そのようなアプリケーションが個別のオペレーティング・システムで実行されているとき、それらのアプリケーションは実際には互いに対してローカルであるという事実を活用できることから、この相互運用性の問題に対処することが重要である。
【００１０】
上述のTemple他による「INTER-PARTITION SHARED MEMORYMETHOD, SYSTEM AND PROGRAM PRODUCT FOR A PARTITIONED PROCESSING ENVIRONMENT」という名称の米国特許出願番号０９／５８４２７６では、数個のオペレーティング・システムの「カーネル」を拡張することにより、共有ストレージの使用を容易にしてパーティションを越えたメモリ共有を実現する。「カーネル」とは、オペレーティング・システム中の中核的なシステム・サービス・コードである。ネットワーク・メッセージ・パッセージのプロトコルは、そのようにして作成したインタフェース上で実装することができるが、しばしば、１つまたは複数のオペレーティング・システムの変更に頼ることなく効率的なプロセス間通信を可能にすることが求められる。また、上述のTemple他による米国特許出願番号０９／５８４２７６、または米国特許第５，９３１，９３８号に記載されるSunMicrosystems社のUltra Enterprise 10000ハイエンド・サーバのように、メモリ領域を共有するためにパーティションの分離を制限せずに済むことも望まれる。同時に、ネットワーク速度ではなくメモリ速度によりパーティション間で情報をやり取りすることも望まれる。したがって、アドレスを共有せずにパーティション・メモリ間でメモリを移動する方法が求められる。
【００１１】
ＩＢＭ Ｓ／３９０ Ｇｂｉｔ Ｅｔｈｅｒｎｅｔ（Ｒ）（１９９５年８月１５日に発行され、ＩＢＭ社に譲渡された米国特許第５４４２８０２号「Asynchronous Coprocessor Data Mover Method and Means」）のＩ／Ｏアダプタを使用して、あるパーティションのカーネル・メモリから別のパーティションのカーネル・メモリにデータを移動することができるが、データは、第１のカーネル・メモリからアダプタのキュー・バッファに移動され、次いでアダプタの第２のキュー・バッファに転送してから第２のカーネル・メモリに

転送される。すなわち、メモリからメモリに転送する際に合計３回のデータ移動がある。どのメッセージ・パッシング通信方式でも、データ・アクセスの待ち時間が共有ストレージへの一度の格納とストレージからの取り出しの待ち時間に近づくように、データ移動操作の回数を最少に抑えることが望ましい。移動機能は、転送するデータ・ブロックごとに３回のデータ移動操作を有する。この操作のうち１つまたは２つの操作を除去する方法が望まれる。

【００１２】

同様に、ＩＢＭ　Ｓ／３９０　Ｐａｒａｌｌｅｌ　Ｓｙｓｐｌｅｘ　Ｃｏｕｐｌｉｎｇ　Ｆａｃｉｌｉｔｙマシンは、パーティション間のメッセージ・パッシングを容易にするために使用することができ、また実際に使用されている。しかし、この場合、データの転送は、第１のカーネル・メモリから結合機構へ、そして結合機構から第２のメモリへと行われる。これには、求められる１回の移動ではなく２回のデータ操作が必要となる。

【００１３】

多くのコンピュータ・システムでは、無許可あるいは不当なアクセスを通じたマシン上のデータおよびアプリケーションの不正使用を防止するために、ユーザの識別を確認することが望まれる。各種のオペレーティング・システムおよびアプリケーション・システムは、この目的のためにユーザ認証および他のセキュリティ・サービスを備える。区分システム、あるいは実際には任意のシステムのクラスタまたはネットワークに入ろうとするユーザを、エントリ時に１回のみ、あるいは重要なリソースに対する要求時や重要なシステム保守機能の実行時などの重要なチェックポイントで確認することが望ましい。この要望は「シングル・サインオン」の必要性として知られる。このために、各種パーティションのセキュリティ・サーバは相互に対話するか、または統合しなくてはならない。この例には、ＯＳ／３９０におけるウェブから受信した「デジタル証明書」を処理し、それを従来のユーザＩＤおよびパスワード検証および権利付与にマッピングするＯＳ／３９０　ＳＡＦ（ＲＡＣＦ）インタフェースの強化や、Ｋｅｒｂｅｒｏｓセキュリティ・サーバ、および台頭しつつあるディレクトリ・サービス向けのＬＤＡＰ規格がある。

【００１４】

さらに、電子商取引は競争的な性質を持つため、ユーザ認証および権利付与のパフォーマンスは、従来のシステムに比べて重要性が高い。労働者は、業務の開始時に認証されるまで待つことを予期しているかもしれないが、顧客は、認証に時間がかかり過ぎる場合には単によそに行ってしまうことが考えられる。ウェブは公衆性があるので、暗号化の使用はこの問題を難しくする。また、しばしば、他のオペレーティング・システムのためには書かれていないデバイス・ドライバがあるオペレーティング・システムに存在することがある。そのような場合には、別のパーティションからあるパーティションのデバイス・ドライバに効率的な形でインタフェースを取ることが望ましい。現在このタイプの動作に利用できるのはネットワーク接続だけである。

【００１５】

分散システムに伴う問題の１つは、他のシステムが過度に利用されている一方での、「ホワイト・スペース」、すなわちあるシステムで充分に利用されていないリソースの管理である。システム間あるいはシステム・イメージ間でワークロードを移動するＩＢＭのＬｏａｄＬｅｖｅｌｅｒやＯＳ／３９０オペレーティング・システムのワークロード・マネジャが備えるＰａｒａｌｌｅｌ　Ｓｙｓｐｌｅｘ機能などのワークロード・バランサがある。区分コンピューティング・システムでは、パーティション間でワークロードではなくリソースを移すことが可能であり、また望ましい。これは、機能の変更に伴う大量のコンテクストの切り替えとデータ移動が回避されることから望ましい。

【００１６】

外部のＳｙｓｐｌｅｘのクラスタ化した接続を使用してＵＮＩＸ（Ｒ）オペレーティング・システムのソケット間接続を実装するＩＢＭ　Ｓ／３９０の「Ｓｙｓｐｌｅｘ　Ｓｏｃｋｅｔｓ」は、従来技術の一部の一例である。この技術では、あるサービスが、利用可能なセキュリティのレベルを示し、アプリケーションが示す要求されるセキュリティ・レ

ベルに基づいて接続をセットアップする。しかし、この場合は、より高いレベルのセキュリティのために暗号化が提供され、またＳｙｓｐｌｅｘ接続自体が、本発明によって実装されるメモリ接続に比べてはるかに深い物理トランスポート層を有する。

【００１７】

同様に、ＳＳＬによる認証を提供し、証明書情報を（プロキシとして）ウェブ・アプリケーション・サーバに提供するウェブ・サーバは、本発明のメモリ共有またはメモリからメモリへの直接のメッセージが効果的に使用される別の例と考えることができる。ここでは、プロキシは、セキュリティ・サーバに渡すデータを再度暗号化する必要がなく、さらに、管理すべき深い接続インタフェースを持たない。実際、当業者には、本発明のこの実施形態では、プロキシ・サーバは、基本的に、セキュリティ・サーバと同じオペレーティング・システム下で実行されるプロキシ・サーバと実質的に同じプロセスを通じてセキュリティ・サーバと通信することが理解されよう。Baskey他による米国特許出願番号０９／４１１４１７「Methods, Systems and Computer ProgramProducts for Enhanced Security Identity Utilizing an SSL Proxy」には、プロキシ・サーバを使用してセキュアＨＴＴＰプロトコルでセキュア・ソケット・レイヤ（ＳＳＬ）を行うことについて述べられる。

【課題を解決するための手段】

【００１８】

本発明の一態様によれば、処理システムは、少なくとも第１のパーティションおよび第２のパーティションを含む。このシステムは、すべてのパーティションと通信するパーティション・リソース・マネジャを備える。第１のパーティションで実行されるプログラムは、第１のパーティションのネットワーク・トラフィック及び第１のパーティションのプロセッサの利用率を含む第１のパーティションの第１スループット情報を収集し、第１スループット情報に基づいて、第１のパーティションのプロセッサの累積利用率の変化に対する第１のパーティションの累積トラフィックの変化の割合である速度基準を計算する。パーティション・リソース・マネジャは、第１のパーティション、すなわち第１のパーティションで実行されるプログラムから、第１のパーティションの速度基準を受け取り、第１のパーティションのリソースに対してリソース平衡化の指示を決定する。ここで、リソース平衡化の指示は、速度基準の値に基づいて作成される。リソース・マネジャは、リソース平衡化の指示に従って第１のパーティションにリソースを割り振る。

【００１９】

好ましい実施形態では、パーティション・リソース・マネジャ機能は、ハイパーバイザと併せてワークロード・マネジャを使用して実装すると有利である。このシステムは、すべてのパーティションと通信するハイパーバイザを備える。第２のパーティションにはワークロード・マネジャが提供され、第１のパーティション、すなわち第１のパーティションで実行されるプログラムから速度基準についての情報を受け取り、リソース平衡化の指示を決定する。コミュニケータは、ワークロード・マネジャからハイパーバイザにリソース平衡化の指示を伝達する。第１のパーティションのカーネルは、ハイパーバイザから受け取るリソース平衡化の指示に従って第１のパーティションにリソースを割り振る。

【００２０】

共有されるメモリ・リソースは、複数のパーティションで実行される相互動作する複数プロセスのための指定メモリ・リソースに独立してマッピングすることが好ましい。このようにして、メモリ・リソースを共有する各パーティション中のプロセスによって共通の共有メモリ空間をマッピングすると、共有メモリは、パーティション内でそのプロセスに割り当てられ、通常のプロセス実行の過程中にデータの読み書きに使用できるメモリ・リソースのように見える。

【００２１】

さらなる実施形態では、プロセスは相互に依存しており、共有メモリ・リソースは、いずれかまたは両方のプロセスによる後のアクセスのために、いずれかまたは両方のプロセスから格納を行うことができる。

【００２２】
さらなる実施形態では、システムは、パーティション中の各種プロセスを共有メモリ空間に接続するプロトコルを含む。
【００２３】
別の実施形態では、区分に関係なくすべての物理メモリへの物理的アクセスを備えるＩ／Ｏアダプタにより、あるパーティションのカーネル空間から別のパーティションのカーネル空間への直接のデータ移動が可能になる。Ｉ／Ｏアダプタのすべてのメモリにアクセスする能力は、パーティション間のＩ／Ｏリソース共有を可能にする区分コンピュータ・システム中の機能から当然得られる結果である。このような共有については、１９９５年５月９日に発行された米国特許第５，４１４，８５１号「METHOD AND MEANS FOR SHARING I/O RESOURCES BY A PLURALITY OFOPERATING SYSTEMS」に記載される。ただし、アダプタは、データ・ムーバを使用することにより、あるパーティションのメモリから別のパーティションのメモリに直接データを移動する能力を有する。
【００２４】
さらなる実施形態では、カーネル・メモリ間のデータ移動のための機能は、ネットワーク通信アダプタのハードウェアおよびデバイス・ドライバ内に実装される。
【００２５】
さらなる実施形態では、メモリ間インタフェースを通じたローカルであるが異種の要素を含むセキュアな接続のためにそれぞれが最適化されたＴＣＰ／ＩＰスタックから、ネットワーク・アダプタを駆動する。
【００２６】
別の実施形態では、データ・ムーバ自体を区分処理システムの通信ファブリック中に実装し、Ｉ／Ｏアダプタによって制御して、さらに直接的なメモリからメモリへの転送を容易にする。
【００２７】
さらに別の実施形態では、オペランドとして供給されるネットワーク・アドレスおよびオフセットを物理アドレスに変換することができる、特権ＣＩＳＣ命令のマイクロコードによってデータ・ムーバを制御し、それにより、データ・ムーバは、実際のアドレスと仮想アドレスを２つのパーティションに有する物理アドレス間で、move character long命令（ＩＭＢ　Ｓ／３９０のＭＶＣＬ命令。ＩＢＭ文書ＳＡ２２－７２０１－０６　"ESA/390Principles of Operation"を参照）に相当する操作を行う。
【００２８】
さらに別の実施形態では、すべての物理メモリへの仮想メモリ・アクセスおよび実際のメモリ・アクセスを行い、オペランドとして供給されるネットワーク・アドレスおよびオフセットを物理アドレスに変換することができるハイパーバイザで実行されるルーチンによってデータ・ムーバを制御し、それによりデータ・ムーバは、実際のアドレスと仮想アドレスを２つのパーティションに有するアドレス間でmove character long命令（ＩＢＭ　Ｓ／３９０　ＭＶＣＬ）に相当する操作を行う。
【００２９】
パーティションの１つでサーバ・プロセスを実装し、他のパーティションでクライアント・プロセスを実装することにより、区分システムは、異種の要素からなる単一システムのクライアント・サーバ・ネットワークを実装することができる。既存のクライアント／サーバ・プロセスは通例ネットワーク・プロトコル接続によって相互動作するので、そうしたプロセスは、本発明のメッセージ・パッシングの実施形態で容易に実装され、インタフェースの変更に頼ることなくパフォーマンスとセキュリティの利点が得られる。ただし、共有メモリの実施形態でのクライアント／サーバ・プロセスの実装は、パフォーマンスまたは配置の速度、あるいはその両方で有利である可能性がある。
【００３０】
さらなる実施形態では、共有メモリまたはメモリ間のメッセージ・パッシングを利用するアプリケーション・サーバに対して、信頼性のある／保護されたサーバ環境が提供され

る。この環境では、現行技術のような追加的な暗号化または認証を必要とせずに、外部化を伴う認証および認証データのセキュリティの露出が回避される。

【００３１】

特定の実施形態では、ウェブ・サーバは、ＯＳ／３９０、Ｚ／ＯＳ、またはＶＭ／３９０の下で実行される「ＳＡＦ」セキュリティ・インタフェースと、メモリ・インタフェースを通じて通信するＯＳ／３９０向けのＬｉｎｕｘで実行されるＬｉｎｕｘ　Ａｐａｃｈｅである。この実施形態では、Ｌｉｎｕｘの「プラグ可能認証モジュール」を変更して、メモリ接続を通じてＳＡＦインタフェースを駆動する。

【００３２】

さらなる実施形態では、セキュリティ・クリデンシャル／コンテクストを共有メモリに格納するか、メモリ間の転送を介して複写するように、Ｐｏｌｉｃｙ　Ｄｉｒｅｃｔｏｒあるいは ＲＡＣＦのようなセキュリティ・サーバを変更する。

【００３３】

このように、本発明の好ましい実施形態は、複数の異種のオペレーティング・システム・イメージをサポートすることが可能な区分されたコンピュータ・システムを含み、それらのオペレーティング・システム・イメージは、メモリ位置を共有せずに、それらのメモリ位置間でメモリ速度により同時にメッセージをやり取りすることができる。これは、あるパーティションのあるカーネル・メモリ空間から第２のパーティションのカーネル・メモリ空間への直接のデータ移動を協働して容易にする特殊なデバイス・ドライバを備えるＩ／Ｏアダプタを使用して行う。

【００３４】

次いで添付図面を参照して単なる例として本発明の一実施形態を説明する。

【発明を実施するための最良の形態】

【００３５】

本発明の好ましい実施形態の特定の態様について述べる前に、区分処理システムの基本的な構成要素を概説すると有用であろう。これを背景知識として使用することにより、本発明の特定の有利な特徴をどのように区分システムで用いてそのパフォーマンスを向上できるかをより深く理解することができる。ＩＢＭ文書ＳＣ２８－１８５５－０６「"OS/390 V2R7.0 OSA/SF User's Guide」を参照されたい。この書籍は、ＯＳ／３９０オペレーティング・システムの一要素であるOpenSystems Adapter Support Facility (OSA/AF)の使用法について記載する。この書籍には、ＯＳＡ／ＳＦをセットアップし、ＯＳ／２インタフェースまたはＯＳＡ／ＳＦコマンドのいずれかを使用してＯＳＡをカスタマイズし、管理するための指示が提供される。Ｇ３２１－５６４０－００「S/390cluster technology: Parallel Sysplex」は、汎用目的の大規模な商用マーケットプレース向けに開発されたクラスタ化マルチプロセッサ・システムについて述べる。Ｓ／３９０　Ｐａｒａｌｌｅｌ　Ｓｙｓｐｌｅｘシステムは、高度にスケーラブルなクラスタ化コンピューティング環境で完全なデータ共有と並列処理を組み合わせるために設計されたアーキテクチャに基づく。Ｐａｒａｌｌｅｌ　Ｓｙｓｐｌｅｘシステムは、コスト、パフォーマンス範囲、および可用性の面で著しい利点をもたらす。ＩＢＭ文献ＳＣ３４－５３４９－０１「MQSeriesQueue Manager Clusters」は、ＭＱＳｅｒｉｅｓのキュー・マネジャ・クラスタについて記述し、クラスタの概念、用語、および利点を説明する。この文献は、新規のコマンドと変更されたコマンドの構文を要約し、キュー・マネジャのクラスタをセットアップし、維持するタスクの例をいくつか挙げる。ＩＢＭ出版ＳＡ２２－７２０１－０６「ESA/390Principles of Operation」は、参照のためにＥＳＡ／３９０アーキテクチャの詳細な定義を含む。この書籍は主にアセンブラ言語のプログラマが使用する参考文献として書かれたものであり、各機能に依存するアセンブラ言語プログラムを作成するのに必要とされる詳細のレベルで各機能を説明しているが、ＥＳＡ／３９０の機能面の詳細に関心を持つ者にとっては有用であろう。

【００３６】

上述の文献は、当技術分野の現行状態の事例を提供し、本発明の背景を理解する助けと

なろう。これらの参考文献は、参照により本明細書に組み込まれる。

【００３７】

図１に、区分処理システム１００を構成する基本要素を示す。システム１００は、ブロックＡおよびＢとして図示するブロックに区分することが可能な物理的なメモリ・リソースからなるメモリ・リソース・ブロック１０１と、区分されたメモリ・リソース１０１と一致するように論理的または物理的に区分することができる１つまたは複数のプロセッサからなることが可能なプロセッサ・リソース・ブロック１０２と、同様に区分することが可能な入出力（Ｉ／Ｏ）リソース・ブロック１０３とからなる。これらの区分されたリソース・ブロックは、スイッチ・マトリクスなどからなる相互接続ファブリック１０４を介して相互に接続される。相互接続ファブリック１０４は、プロセッサ１０２Ｂをメモリ１０１Ｂに接続するなど、パーティション内でリソース同士を相互接続する機能を果たすことができ、また、プロセッサ１０２Ａをメモリ１０１Ｂに接続するなどパーティション間のリソースを相互に接続する役割も果たすことができることは理解されよう。本明細書で使用する用語「ファブリック」は、システムの諸要素を相互接続するための当技術分野で知られる一般的な方法を意味する。ファブリックは、単純な二地点間バスであっても高度なルーティング機構であってもよい。この図には２つのパーティション（ＡおよびＢ）を有するシステムを示しているが、この表現は説明を簡略にするために選択したものであり、さらに、本発明は、使用可能なリソースとパーティション技術が可能にする数のパーティションを実装するように構成可能なシステムを包含することは容易に認識されよう。

【００３８】

図を考察すると、個別に見たパーティションＡおよびＢはそれぞれ、単独のデータ処理システムの構成要素、すなわちプロセッサ、メモリ、およびＩ／Ｏからなることが容易に理解されよう。この事実は、区分された処理システムに、独自の「systems within a system」の利点を与える特徴である。事実、そして本明細書で例証するように、現在利用可能なパーティション処理システムを分ける主要な差異は、システム・リソースを区分することが可能な境界と、その境界を越えてパーティション間でリソースを移動できる容易さである。

【００３９】

パーティションを分ける境界が物理的な境界である第１のケースの最適な例となるのはSunMicrosystemsのUltra Enterprise 10000システムである。Ultra Enterprise 10000システムでは、パーティションは物理的な境界に従って区画され、具体的には、ドメインまたはパーティションは、それぞれがいくつかのプロセッサ、メモリ、およびＩ／Ｏデバイスからなる１つまたは複数の物理的なシステム・ボードからなる。ドメインは、そのシステム・ボードの１つまたは複数、およびそのボードに装着されたＩ／Ｏアダプタとして定義される。ドメインは、独自開発のバスおよびスイッチ・アーキテクチャによって相互に接続される。

【００４０】

図２に、物理的に区分された処理システム２００を構成する要素の高レベル表現を示す。図２の参照から分かるように、システム２００は２つのドメインあるいはパーティションＡおよびＢを含む。パーティションＡは、２つのシステム・ボード２０２Ａ１および２０１Ａ２からなる。パーティションＡの各システム・ボードは、メモリ２０１Ａ、プロセッサ２０２Ａ、Ｉ／Ｏ　２０３Ａ、および相互接続媒体２０４Ａを含む。相互接続媒体２０４Ａは、システム・ボード２０１Ａ１上の構成要素が相互と通信することを可能にする。同様に、単一のシステム・ボードからなるパーティションＢも同様の構成処理要素、すなわちメモリ２０１Ｂ、プロセッサ２０２Ｂ、Ｉ／Ｏ２０３Ｂ、および相互接続２０４Ｂを含む。パーティションに分けられたシステム・ボードに加えて、各システム・ボードに結合され、パーティション内のシステム・ボード間の相互接続と、異なるパーティションのシステム・ボードの相互接続を可能にする相互接続ファブリック２０５が存在する。

【００４１】

次のタイプのシステム・パーティションを論理的パーティショニングと呼ぶ。このシス

テムでは、各種のパーティションへのリソースの割り当てを制約する物理的な境界がなく、システムは、その物理的な場所に関係なく任意のパーティションに割り当てることができる使用可能なリソースのプールを備えるものと考えることができる。この点が、例えば所与のシステム・ボード（システム・ボード２０１Ａ１など）上のすべてのプロセッサが必然的に同じパーティションに割り当てられる、物理的に区分されたシステムとの違いである。ＩＢＭ　ＡＳ／４００システムは、論理的に区分された専用のリソース処理システムの例である。ＡＳ／４００システムでは、ユーザは、プロセッサ、メモリおよびＩ／Ｏをそれらの物理的な場所に関係なく所与のパーティションに含めることができる。したがって、例えば、物理的に同じカード上に位置する２つのプロセッサを２つの異なるパーティションのリソースとして指定することができる。同様に、カードなど所与の物理的なパッケージ中のメモリ・リソースは、そのアドレス空間の一部を論理的に１パーティションのために確保し、残りのアドレス空間を別のパーティションのために確保することができる。

【００４２】

ＡＳ／４００システムなど論理的に区分された専用リソース・システムの特徴は、パーティションへのリソースの論理的マッピングが、手動によるシステムの再構成によってのみ変更することができる、静的に行われる割り当てである点である。図３を参照すると、プロセッサ３０２Ａ１は、物理的にはシステム中のどこに位置してもよく、論理的にはパーティションＡのために確保されたプロセッサを表す。ユーザが、プロセッサ３０２Ａ１をパーティションＢにマッピングし直したい場合は、プロセッサをオフライン状態にし、変更に対応するように手動で再度マッピングしなければならない。論理的に区分されたシステムは、例えば固定数のプロセッサをサポートするシステム・ボードなどの物理的な区分の境界による制限を受けないので、リソースの区分により高い精度を提供する。ただし、そのような論理的に区分された専用のリソース・システムの再構成を行うには、パーティションの再マッピングを行うリソースの動作を中断しなくてはならない。したがって、そのようなシステムでは、物理的に区分されたシステムに伴う制限の一部は回避されるが、なおパーティション間のリソースの静的なマッピングに伴う再構成の制限があることが分かる。

【００４３】

この結果、我々は、論理的に区分された共有リソース・システムの検討に至った。そのようなシステムの一例は、ＩＢＭ　Ｓ／３９０コンピュータ・システムである。論理的に区分された共有リソース・システムの特徴の１つは、プロセッサなど論理的に区分されたリソースを複数のパーティションで共有できる点である。この特性により、論理的に区分された専用リソース・システムの再構成の制約が効果的に克服される。

【００４４】

図４に、論理的に区分されたリソース共有システム４００の概略的な構成を示す。論理的に区分された専用のリソース・システム３００と同様に、システム４００は、システム中の物理的な場所に関係なく任意のパーティション（この例ではＡまたはＢ）に論理的に割り当てることが可能なメモリ４０１、プロセッサ４０２、およびＩ／Ｏリソース４０３を含む。ただし、システム４００から分かるように、特定のプロセッサ４０２またはＩ／Ｏ４０３の論理パーティションへの割り当ては、「ハイパーバイザ」（４０８）で実行されるスケジューラに従って仮想プロセッサ（４０６）およびＩ／Ｏドライバ（４０７）を入れ替えることにより動的に変更することができる。（ハイパーバイザは、仮想マシンのリソースをスケジュールし、割り振る管理プログラムである。）プロセッサおよびＩ／Ｏを仮想化することにより、適切な優先順位付けを行って全オペレーティング・システム・イメージを動作中および非動作時に入れ替えることが可能になり、パーティションがそれらのリソースを動的に共有できるようになる。

【００４５】

論理的に区分された共有リソース・システム４００は、プロセッサとＩ／Ｏリソースを共有する機構を提供するが、パーティション間のメッセージ・パッシングは、既存のシス

テムでは完全には対応されていなかった。これは、既存の区分システムではパーティション間の通信を可能にすることができないということではない。実際、そのような通信はここに記載する各タイプの区分システムで行われている。しかし、それらの実装で、ハイパーバイザ、共有メモリの実装、あるいはアダプタやチャネル通信デバイスの標準的なセット、あるいはパーティション間を接続するネットワークの介在なしに、カーネル・メモリからカーネル・メモリにデータを移動する手段を提供するものはない。

【００４６】

米国特許第５，９３１，９３８号に記載されるSun MicrosystemsUltra Enterprise 10000システムを代表とする、物理的に区分された多重処理システムでは、マスク・レジスタを適切に設定することにより、システム・メモリの一領域を、ハードウェア・レベルの複数のパーティションからアクセス可能にすることができる。このＳｕｎ特許は、パーティション間ネットワークのためのバッファリング機構および通信手段として使用できることを指摘する以外には、この機能をどのように活用するかを教示しない。上述のTemple他による米国特許出願番号０９／５８４２７６は、異種の要素からなる区分システムにおいて共有メモリ機構をどのように構築し、活用するかを教示する。

【００４７】

「Coupling Facility Configuration Options:A Positioning Paper」(ＧＦ２２－５０４２－００,ＩＢＭ社)に詳細に記述されるように、ＩＢＭ　Ｓ／３９０システムでは、共通にアドレス指定される物理メモリを「統合された結合機能」として使用するための同様の内部のクラスタ化機能について記載される。ここでは、共有ストレージは実際にリポジトリであるが、そこへの接続は、ＸＣＦと称されるＩ／Ｏ様のデバイス・ドライバを通じて行われる。ここでは、共有メモリは結合機能中に実装されるが、Ｓ／３９０でないオペレーティング・システムがそれを使用するには拡張を作成することが必要である。さらに、この実装では、１パーティションのカーネル・メモリから結合機能のメモリにデータを移動させ、そして第２のパーティションのカーネル・メモリにデータを移動させる。

【００４８】

カーネルとは、ハードウェア・リソースの割り振りなどの基本的機能を行うオペレーティング・システムの部分である。カーネル・メモリとは、カーネルが自身の機能を実行するために使用する、カーネルが利用できるメモリ空間である。

【００４９】

これに対し、この実施形態は、どちらかのパーティションまたはハードウェア中でオペレーティング・システムに共有ストレージの拡張を提供することなく、新規なＩ／Ｏアダプタとそのデバイス・ドライバのイネーブリング機能を使用して、あるパーティションのカーネル・メモリから別のパーティションのカーネル・メモリに１回の操作でデータを移動する手段を提供する。

【００５０】

ここで説明する実施形態の動作を理解する助けとして、オペレーティング・システム中のプロセス間通信を理解すると有用である。図５を参照すると、プロセスＡ（５０１）およびＢ（５０３）は、それぞれアドレス空間メモリＡ（５０２）およびメモリＢ（５０４）を有する。これらのアドレス空間には、カーネル（５０５）によるシステム・コールの実行により実際のメモリが割り振られている。カーネルは、独自のアドレス空間であるメモリＫ（５０６）を有する。１つの通信形態では、プロセスＡおよびＢは、バッファ５１０を作成し、そこに接続し、アクセスする適切なシステム・コールを行うことにより、メモリＫにバッファ５１０を作成することで通信する。これらの呼び出しの意味は、システムごとに異なるが効果は同じである。第２の通信形態では、メモリＳ（５０７）のセグメント５１１をメモリＡ（５０２）およびメモリＢ（５０４）のアドレス空間にマッピングする。このマッピングが完了すると、プロセスＡ（５０１）およびＢ（５０３）は、両プロセスが理解する任意のプロトコルに従って、メモリＳ（５０７）の共有セグメントを自由に使用することができる。

【００５１】

米国特許出願番号０９／５８３５０１「Heterogeneous ClientServer Method, System and Program Product For A Partitioned ProcessingEnvironment」を図６に表し、ここではプロセスＡ（６０１）およびＢ（６０３）が、異なるオペレーティング・システム・ドメイン、イメージ、あるいはパーティション（パーティション１（６１４）およびパーティション２（６１５））に存在する。この場合には、メモリＫ１（６０６）およびメモリＫ２（６０８）を各自のカーネル・メモリとして有するカーネル１（６０５）およびカーネル２（６０７）がある。メモリＳ（６０９）は、ここではパーティション１およびパーティション２の両方からアクセス可能な物理メモリの空間である。このような共有は、これらに限定しないが、ＵＥ１００００のメモリ・マッピング実装、またはＳ／３９０のハイパーバイザ実装、あるいは区分によって生じるアクセスの障壁を制限する他の任意の手段を含む任意の実装によって可能にすることができる。これに代わる例として、共有メモリを最も高い物理メモリ・アドレスにマッピングし、コンフィギュレーション・レジスタ中の先頭アドレスによって共有空間が定義される。

【００５２】

慣例として、メモリＳ（６０９）は、メモリＫ１およびメモリＫ２にマッピングされた、カーネル１およびカーネル２の拡張によって使用される共有セグメント（６１０）を有する。セグメント６１０は、メモリＫ１（６０６）およびメモリＫ２（６０８）にマッピングされて上述の第１の形態によるパーティション間通信を可能にするメモリ（６０９）のセグメントについての定義および割り振りのテーブルを保持するのに使用されるか、または上記で図５を参照して説明した第２の通信形態に従ってメモリＡ（６０２）およびメモリＢ（６０４）にマッピングされたセグメントＳ２（６１１）を定義するために使用される。本発明の一実施形態では、メモリＳは、制限されたサイズであり、実際のストレージ中で固定されている。ただし、メモリを固定する必要はなく、付随するページ管理タスクが効率的に管理されるのであれば、より大きな共有ストレージ空間を可能にすることを企図する。

【００５３】

ここで参照する本発明の第１の実施形態では、共有メモリ構成データ・セット（ＳＭＣＤＳ）（６１３）からデータを読み出し、メモリＳ（６０９）のセグメントＳ１（６１０）にテーブルを構築する共有メモリ構成プログラム（ＳＭＣＰ）（６１２）と呼ばれるスタンドアロンのユーティリティ・プログラムにより、共有ストレージの定義および割り振りのテーブルをメモリ中にセットアップする。したがって、どのカーネルがどのストレージ・セグメントを共有するかについての割り振りと定義は、このユーティリティが作成する構成によって固定され、あらかじめ決定される。各種のカーネル拡張は次いでその共有ストレージを使用して、パイプ、メッセージ・キュー、ソケットなどの、各種のイメージ間、プロセス間の通信構造を実施し、さらには、各自の規定および規則に従っていくつかのセグメントを共有メモリ・セグメントとしてユーザ・プロセスに割り振る。これらのプロセス間の通信は、ＩＰＣ　ＡＰＩ６１８および６１９を通じて可能になる。

【００５４】

共有ストレージの割り振りテーブルは、イメージ識別子、セグメント番号、ｇｉｄ、ｕｉｄ、「スティッキー・ビット」、および許可ビットからなるエントリを含む。スティッキー・ビットは、関連付けられたストアがページング可能でないことを示す。この例の実施形態では、スティッキー・ビットは確保され、１と想定する（すなわちデータはメモリ中のその場所に固定される、すなわち「固着されて」いる）。セグメントを使用する各グループ、ユーザ、およびイメージは、テーブルにエントリを有する。慣例的に、すべてのカーネルは、テーブルを読むことができるが、書き込むことはできない。初期化時に、カーネル拡張は、構成テーブルを読み取り、イメージを越えるプロセス間通信が他のプロセスから要求された際に使用する独自の割り振りテーブルを作成する。割り振られた空間の一部またはすべては、プロセス間通信を要求する他のプロセスからの要求によりカーネルが作成する「パイプ」、ファイル、およびメッセージ・キューを実装するためにカーネルが使用する。パイプとは、カーネル機能を通じて第２のプロセスに導かれる１プロセスか

らのデータである。パイプ、ファイル、およびメッセージ・キューは、Ｌｉｎｕｘ、ＯＳ／３９０　ＵＳＳ、および大半のＵＮＩＸ（Ｒ）オペレーティング・システムで使用される、標準的なＵＮＩＸ（Ｒ）オペレーティング・システムのプロセス間通信のＡＰＩおよびデータ構造である。共有空間の一部は、システムを越えた直接的なメモリ共有のために、さらなるカーネル拡張により他のプロセスのアドレス空間にマッピングすることができる。

【００５５】

共有メモリの割り振り、使用、および仮想アドレス空間へのマッピングは、各自の規定および変換プロセスに従って各カーネルによって行われるが、基本的なハードウェア・ロックおよびメモリ共有のプロトコルは、システムの残りの部分の基礎となる共通のハードウェア設計アーキテクチャによって駆動される。

【００５６】

通信を行うために、上層レベルのプロトコルは共通でなければならない。好ましい実施形態では、これは、各種のオペレーティング・システム・イメージそれぞれに、ＵＮＩＸ（Ｒ）オペレーティング・システムとともに使用するＩＰＣ（プロセス間通信）ＡＰＩを実装させることによって行われ、拡張がその要求をイメージを越えた要求であると識別する。この拡張は、パラメータか、または別個の新しい識別子／コマンド名によることができる。

【００５７】

図４および７を参照すると、本発明では、チャネルまたはネットワーク接続を通じたデータ転送と、オペレーティング・システムに対する共有メモリの拡張の使用がともに回避されることが分かる。パーティション７１４のアプリケーション・プロセス（７０１）は、カーネル１（７０５）を呼び出すソケット・インタフェース７１８にアクセスする。ソケット・インタフェースとは、ＴＣＰ／ＩＰスタックの特定のポートを、リッスン中のユーザ・プロセスに関連付ける構造である。カーネルはデバイス・ドライバ（７１６）にアクセスし、デバイス・ドライバ（７１６）は、Ｉ／Ｏアダプタ（７２０）のハードウェアを通じて、メモリ（４０１）にとってはメモリからメモリへの移動に見えるような形で、パーティション７１４および７１５のプロセッサ（４０２）またはファブリック（４０４）あるいはその両方に実装されるキャッシュ・メモリを回避して、カーネル・メモリ１（７０６）からカーネル・メモリ２（７０８）にデータを転送させる。データを移動すると、Ｉ／Ｏアダプタは次いでパーティション７１５のデバイス・ドライバ（７１７）にアクセスし、データを移動したことを通知する。デバイス・ドライバ７１７は次いで、カーネル２を待っているデータがソケット（７１９）にあることをカーネル２（７０７）に通知する。するとソケット（７１９）は、データをアプリケーション・プロセス（７０３）に提示する。このようにメモリからメモリへの直接の移動が行われ、同時に外部のインタフェースでデータを移動することが回避され、またメモリ共有のためのどちらかのオペレーティング・システムの拡張も回避される。

【００５８】

これに対し、図８に示す従来技術によるシステムでは、個別のメモリ移動操作を使用してカーネル・メモリ１（７０６）からアダプタ・メモリ・バッファ１（７２１）に移動を行う。第２のメモリ移動操作で、アダプタ・メモリ・バッファ１（７２１）からアダプタ・メモリ・バッファ２（７２２）にデータを移動する。そして第３のメモリ移動操作で、アダプタ・メモリ・バッファ２（７２２）からカーネル・メモリ２（７０８）にデータを移動する。これは、２つのカーネル・メモリ間でデータを移動するのに３回の別個のメモリ移動操作を使用することを意味し、これに対して図７の本発明では、１回のメモリ移動操作でカーネル・メモリ１（７０６）からカーネル・メモリ２（７０８）に直接データを移動する。これには、ユーザ・プロセスから見た待ち時間を低減する効果がある。

【００５９】

さらなる実施形態を図４および９によって例証する。ここでは、実際のデータ・ムーバ・ハードウェア（８２１）をファブリック（４０４）中に実施する。この実施形態の動作

は、Ｉ／Ｏアダプタ８２０内の管理状態（８２２）に従って、ファブリック（４０４）内のムーバ・ハードウェアによって実際にデータを移動する点を除いては上記の説明と同様に進行する。

【００６０】

このようなファブリックに位置するデータ・ムーバの一例が、１９９３年１２月７日発行のRobertD.Herzl他による「Processor System with Improved Memory Transfer Means」という名称の米国特許第５，２６９，００９号に記載される。この文献は参照によりその全体を本明細書に含める。参照する同特許に記載される機構を、パーティションの主要な記憶位置間のデータ転送を含むように拡張する。

【００６１】

本発明の実施形態は以下の要素を含む。ＣＰＵの設計によって定義される、基礎となる共通のデータ移動プロトコル、Ｉ／Ｏアダプタまたはファブリック・ハードウェアあるいはその両方、Ｉ／Ｏアダプタとのインタフェースを実装するデバイス・ドライバの異種のセット、好ましい実施形態ではソケット・インタフェースとして示す共通の高レベルのネットワーク・プロトコル、物理メモリ・アドレスへのネットワーク・アドレスのマッピング、およびＩ／Ｏアダプタ（８２０）が各パーティションのカーネル・メモリおよびデバイス・ドライバと通信するのに使用するＩ／Ｏ割り込みベクトルまたはポインタ。

【００６２】

データ・ムーバは、ハードウェアの状態機械として、またはマイクロコードおよびマイクロプロセッサを用いてＩ／Ｏアダプタ中に実装することができる。あるいは、Ｉ／Ｏアダプタによって制御される、マシンの通信ファブリック中のデータ・ムーバを使用するように実装してもよい。このようなデータ・ムーバの一例が、１９９３年１２月７日発行のHerzl他による米国特許第５，２６９，００９号「PROCESSOR SYSTEM WITH IMPROVED MEMORYTRANSFER MEANS」に記載される。

【００６３】

図１０を参照すると、実装に関係なくデータ・ムーバは以下の要素を有する。メモリからのデータはソース・レジスタ（９０１）に保持され、データは、データ・アライナ（９０２および９０４）を通じてデスティネーション・レジスタ（９０３）に渡され、次いでメモリに戻される。したがって、連続した動作の一部としてメモリからの取り出しそしてメモリへの格納がある。すなわち、位置合わせプロセスは、メモリ・ラインから複数のワードが取り出されると行われる。位置合わせしたデータは、メモリ・ストアが開始されるまで、デスティネーション・レジスタ（９０３）でバッファリングする。ソース・レジスタ（９０１）およびデスティネーション・レジスタ（９０３）は、移動操作中に取り出しと格納の間でどれほどの重複が許されるかに応じて、単一ラインまたは複数ライン分のメモリ・データを保持するために使用することができる。メモリのアドレス指定は、移動中に取り出しと格納が行われるアドレスを追跡するカウンタ（９０５および９０６）から行われる。制御およびバイト数の要素（９０８）は、アライナ（９０２および９０４）を通じたデータ・フローを制御し、メモリ・アドレスに合わせてソース・カウンタ（９０５）または宛先カウンタ（９０６）を選択させる。コントローラ（９０８）は、アドレス・カウンタ（９０５および９０６）の更新も制御する。

【００６４】

図１１を参照すると、データ・ムーバは、デバイス・ドライバによって実装される、特権ＣＩＳＣ命令（１０００）として実装することもできる。ＭＶＸＬは、カウント・レジスタによって指定されるバイト数を、ソース・レジスタで指定される物理アドレスから、デスティネーション・レジスタで指定される物理アドレスに移動する。この命令は特権命令である。（仮想アドレス間ではＭＶＣＬが同じ機能を行う。）ここで、デバイス・ドライバは、仮想アドレスではなく物理アドレスをレジスタにロードし、パーティションを越えたデータ移動を可能にする。このようなＣＩＳＣ命令は、Ｓ／３９０のMove Page、Move Character Longなどのパーティション間のデータ移動に適したハードウェア機能を利用するが、ネットワーク・アドレスおよびオフセットを物理メモリのアドレスにマッピング

するテーブルに従って物理的にメモリをアドレス指定する特権も有する。最後に、データ・ムーバおよびアダプタは、仮想アダプタとして機能するハイパーバイザ・コードによって実装することができる。

【００６５】

図１２に、次のステップからなる、データ・ムーバがアダプタにある場合のデータ・ムーバの動作を示す。

１１０１　ユーザがデバイス・ドライバを呼び出し、ソース・ネットワークのＩＤ、ソース・オフセット、宛先ネットワークのＩＤを提供する。

１１０２　デバイス・ドライバがアドレスをアダプタに転送する。

１１０３　アダプタがアドレスを転送する。

ＩＤから物理的な基底アドレスを検索する（テーブル・ルックアップ）

ロックおよび現在の宛先オフセットを得る

オフセットを加算する

境界を確認する

１１０４　アダプタがカウントとアドレスをレジスタにロードする

１１０５　アダプタがデータ移動を実行する

１１０６　アダプタがロックを解除する

１１０７　アダプタがユーザに「戻る」デバイス・ドライバに通知する

【００６６】

図１３に、以下のメソッドを含む、プロセッサ通信ファブリックで実装されるデータ・ムーバ・メソッドを示す。

１２０１　ユーザがデバイス・ドライバを呼び出し、ソース・ネットワークのＩＤ、ソース・オフセット、宛先ネットワークのＩＤを提供する

１２０２　デバイス・ドライバがアドレスをアダプタに送る

１２０３　アダプタがアドレスを変換する

ＩＤから物理的な基底アドレスを検索する（テーブル・ルックアップ）

ロックおよび現在の宛先オフセットを得る

オフセットを加算する

境界を確認する

アダプタが、ロックおよび物理アドレスをデバイス・ドライバに戻す

１２０４　デバイス・ドライバがデータ移動を実行する

１２０５　デバイス・ドライバがロックを解除する

１２０６　デバイス・ドライバが戻る

【００６７】

以上、区分されたコンピューティング・システム内で異種の要素を含む相互動作を実施する２つの方式を説明した。方式の１つでは、共有メモリ機能とオペレーティング・システム・カーネルの拡張を使用してパーティションを越えたプロセス間の通信プロトコルを可能にし、もう一方の方式では、共有Ｉ／Ｏアダプタの能力を使用してすべての物理メモリをアドレス指定して、メモリからメモリへのメッセージ・パッシングを１回の操作で実施する。

【００６８】

前述の構造により、単一システムのクライアント－サーバ・モデルを活用するいくつかの本発明の実装が得られる。この構造を実装する一方式は、サーバの作業キューを共有される記憶空間に入れて、各種のクライアントが要求を追加できるようにするものである。そして、クライアントがそこに入れられた情報にアクセスできるように、「リモート」クライアントのためのリターン・バッファも共有メモリ空間になければならない。あるいは、上述のメッセージ・パッシング方式を使用して、既存のネットワーク指向のクライアント／サーバを迅速かつ容易に展開することができる。これらの実装は例証として提供しており、限定的と見なすべきではない。実際、当業者は、この構造を各種の方式でさらに発展させ、単一システムの枠組みの中で様々なタイプの異種のクライアント－サーバ・シス

テムを実装できることが容易に理解される。

【００６９】

パーティション・クラスタのワークロード管理

本発明のパーティション・リソース・マネジャを用いたパーティションのワークロード管理について、ＯＳ／３９０の実装により説明する。パーティション・リソース・マネジャのタスクは、あるパーティションで実行されるワークロード・マネジャの連携からなり、ハイパーバイザと通信する別のパーティションの他のワークロード・マネジャ（あるいは特定のオペレーティング・システムに適したスループット情報ジェネレータ）からスループット情報を得、パーティション・リソースの割り振りを調整する役割を担う。当業者には、本発明は、システム・アーキテクチャの必要性に従ってリソース・マネジャ機能を実装することにより、他のシステム・アーキテクチャでも実装できることは容易に理解されよう。

【００７０】

図１４を参照すると、ＯＳ／３９０オペレーティング・システムのワークロード・マネジャ（ＷＬＭ）（１３０８）は、Ｓ／３９０のパーティション・ハイパーバイザと通信して、各パーティションに割り振られたリソースを調整する機能を持つ。これは、ＬＰＡＲクラスタリングとして知られる。ただし、ＯＳ／３９０でないパーティション（１３０１）については、ＷＬＭは、そのパーティションのオペレーティング・システムまたはアプリケーションに基づくのではなく、ハイパーバイザから供給できる利用率およびその他の情報のみに基づいて割り振りを行わなければならない。上述の待ち時間が低いパーティションを越えた通信（１３０５）を使用してパーティションからＷＬＭ（１３０８）に情報をパイピングすることは、システムを越えてリソースを割り振るジョブをより適切に行うためにＷＬＭが必要とする情報をＷＬＭ（１３０８）に与える、オーバーヘッドが非常に低い手段となる。これは、アプリケーションがワークロード管理のために装備されない場合であっても効果的でありうる。この理由は、通例、制御されるシステムは、システムで送受信されるＩＰパケットを数えるＴＣＰ／ＩＰスタック中のパケット・アクティビティ・カウンタにアクセスするコマンドであるＵＮＩＸ（Ｒ）オペレーティング・システムの「ＮＥＴＳＴＡＴ」（ＵＮＩＸ（Ｒ）オペレーティング・システムの標準的なコマンド・ライブラリの一部）を実装することができ、また、ビジー・サイクルとアイドル・サイクルを数え、利用率データ（１３０２）を生成するカーネル中のシステム・アクティビティ・カウンタにアクセスする標準的なＵＮＩＸ（Ｒ）オペレーティング・システム・コマンドであるＵＮＩＸ（Ｒ）オペレーティング・システムの「ＶＭＳＴＡＴ」（ＵＮＩＸ（Ｒ）オペレーティング・システムの標準コマンド・ライブラリの一部）を実行することができるからである。既存のＮＥＴＳＴＡＴコマンドおよびＶＭＳＴＡＴコマンドを使用することは必須ではなく、パケット数と利用率をそれらに供給する基礎的な機構を使用してリソースおよび経路長のコストを最小に抑えることが最良であることは理解されよう。このデータを組み合わせて「速度」の基準（１３０３）とし、それをワークロード・マネジャ（ＷＬＭ）のパーティション（１３０７）に送ることにより、ＷＬＭ（１３０８）は、ハイパーバイザにリソースの調整を行わせることができる。ＣＰＵの利用率が高く、パケット・トラフィックが低い場合、パーティションはより多くのリソースを必要とする。接続（１３０４および１３０６）は、相互接続（１３０５）の実施形態に応じて異なる。共有メモリの実施形態では、それらの接続はＵＮＩＸ（Ｒ）オペレーティング・システムのＰＩＰＥ、Ｍｅｓｓａｇｅ　Ｑ、ＳＨＭＥＭ、あるいはソケット構造である。データ・ムーバの実施形態では、通例ソケット接続である。

【００７１】

一実施形態では、「速度」基準は以下のように得る（IBM RedbookDocument SG24-4810-01 「Understanding RS/6000 Performance and Sizing」に記載されるＵＮＩＸ（Ｒ）オペレーティング・システム・コマンドＮＥＴＳＴＡＴおよびＶＭＳＴＡＴを参照されたい）。

（ＮＥＴＳＴＡＴ）の全パケットについての時間間隔データを使用してスループットの

概略を得る。
時間間隔についてのＣＰＵデータ（ＶＭＳＴＡＴ）を使用して、ＣＰＵの利用率の概略を得る。
ピークを１としてトラフィックを正規化してそれらの概略データをプロットし、表示する（１４０１）。
トラフィック対ＣＰＵについて累積的な相関分析を行う（１４０２）。
トラフィックの関係は、関数Ｔ（Ｃ）に沿った曲線になる。
ここで使用する例（１４０２）では、T(C) = y(x) = 0.802 +1.13xとなる。
S = dT/dC = dy/dxが速度基準であり、この例ではＳ＝１．１３となる。
Ｓが傾向線よりも小さい場合には、より多くのリソースが必要とされる。

【００７２】
図１５の例ではこれは２度発生している（１４０３および１４０４）。管理図は、諸業界で監視プロセスを作成するための標準的な方法である。１４０５ではＳを管理図として動的にプロットしている。この例でパケット・トラフィックとＣＰＵとの間に見られるような関係を与えられると、統計的な制御理論に基づいて収集データを様々な形で監視し、調整することができる。これらの方法は、通例、アクションをトリガする制御変数の閾値に依存する。あらゆるフィードバック・システムでそうであるように、制御不能状態に近いと判断された時に即座にアクションを行わせることが必要である。さもないとシステムが不安定になる可能性がある。この実施形態では、これは、内部通信が提供する低待ち時間の接続によって行われる。
【００７３】
静的な環境では、Ｓを使用して、どの利用率でより多くのリソースが必要とされるのかを明確にすることができる。これは平均にわたって有効であるが、Ｓはワークロードと時間の関数でもある。図１５を参照すると、まずそれは５０～６０％の間のどこかであることが分かり、次いでＳの谷があるとその後に少なくとも１回の時間間隔で利用率のピークがあることが分かる。したがって、ＷＬＭは利用率ではなくＳを与えられた場合により好適なジョブを行う。これは、Ｓが、よりタイムリーなリソース調整を可能にする「先行する指標」であるためである。区分されたマシンのリソースはパーティションによって共有されるので、ワークロード・マネジャはＳのデータを複数のパーティションから得なければならない。データの転送は、非常に低いオーバーヘッド、かつ高い速度で行う必要がある。この実施形態はこれら両方の条件を可能にする。図１４を参照すると、ワークロード・マネジャがないパーティション（１３０１）では、プログラム・ステップ（１３０３）でパラメータ（ここの例では「Ｓ」）を評価するために使用する利用率およびパケットのデータ（１３０２）をモニタが収集する。プログラムは次いで、低待ち時間のパーティション間通信機能（１３０５）への接続（１３０４）を使用し、パーティション間通信機能（１３０５）は、ワークロード・マネジャを備えるパーティション（１３０７）の接続（１３０６）にそのデータを渡し、その接続により「論理パーティション・クラスタ・マネジャ」（１３０８）に入力が提供される。「論理パーティション・クラスタ・マネジャ」については、２０００年１０月２日出願の米国特許出願番号０９／６７７３３８「METHOD AND APPARATUS FOR ENFORCING CAPACITY LIMITATIONS IN ALOGICALLY PARTITIONED SYSTEM」に記載される。
【００７４】
この場合、パーティション・データをワークロード・マネジャに通信する最も効率的な方法は、メモリ共有を通じて行うものであるが、ソケットの待ち時間が十分に低くデータの時間伝達が可能である場合には、内部のソケット接続も機能する。これは、ワークロードと、求められる制御の精度の両方に応じて決まる。
【００７５】
上述は、ワークロード・マネジャがリソースを割り振るための情報を供給する１方式であるが、これは決して限定的なものと解釈すべきではない。この例を選択したのは、これ

が、大量の新たなコードを用いずに、すべてではなくとも大半のオペレーティング・システムから収集することが可能な基準であるためである。クライアント・システムは、応答時間やユーザ数など、ＷＬＭサーバに渡す任意の基準の任意の手段を実装することができる。

【００７６】

非間接的Ｉ／Ｏ

時に、デバイス・ドライバは、ハードウェアによってサポートされる可能なオペレーティング・システムのうち１つのみでしか利用できない場合がある。共有メモリにデバイス・ドライバのメモリ・インタフェースを提示し、すべての付属システムがドライバ・プロトコルに準拠することにより、デバイスを複数のシステムで共有することができる。実際には、１つのパーティションが他のパーティションのＩＯＰになることができる。デバイスに過負荷を課すと、単一システムからデバイスに過負荷を課す場合と同様の否定的な結果になることを理解した上で、デバイスへのアクセスは単一システムのレベルに近いものになる。図１６を参照すると、デバイス・ドライバ（１５０１）は、共有メモリ（１５１１）を通じた、アプリケーションおよびアクセス・メソッド（１５０３）からのＩ／Ｏサービスに対する要求に応答する。

【００７７】

このメッセージ・パッシングの実施形態をいくつかのデバイスに使用することは可能であるが、ソケット、スタックおよびデータ移動の待ち時間を許容しなければならない。これは、ネイティブのデバイスとネットワークに接続されたデバイスとの中間と考えることができる。

【００７８】

デバイス・ドライバを実行するシステム・イメージに割り振られたプロセッサ・リソースを、アプリケーションを実行するシステム・イメージに割り振られたプロセッサ・リソースから分離した場合にはさらなる機能の増大が得られる。この分離を行うと、Ｉ／Ｏ割り込みの対象とならないプロセッサでは、Ｉ／Ｏ割り込みとそれに伴うコンテクストの切り替えによるキャッシュおよびプログラム・フローの中断が回避される。

【００７９】

共通のセキュリティ・サーバ

アプリケーションがウェブ対応型になり、統合されるのに従い、ユーザの確認と権利付与の確定は、従来のシステムの場合と比べて一般的な課題になる。この課題をさらに増大するのは、アプリケーションを統合するために異種のシステムをまとめる必要性である。この結果、ＬＤＡＰ、Ｋｅｒｂｅｒｏｓ、ＲＡＣＦ、およびその他のセキュリティ機能を統合された形で使用するには、通常、セキュリティ機能を行う共通のセキュリティ・サーバへのネットワーク接続が必要とされる。これは、パフォーマンスに影響を与える。また、ネットワーク盗聴者（sniffer）によるセキュリティの露出もある。共有メモリ接続またはメモリ・ムーバ接続を介して共通のセキュリティ・サーバをウェブ・サーバに接続した場合、このアクティビティは著しく速度が上がり、接続は内部化され、セキュリティが向上する。さらに、そのような環境では、一部の顧客は、特にＬＩＮＵＸの場合には、他のＵＮＩＸ（Ｒ）オペレーティング・システムに基づくパスワード保護よりも、Ｓ／３９０「ＲＡＣＦ」、あるいは他のＯＳ／３９０「ＳＡＦ」インタフェースのユーザ認証によるより高いセキュリティを選ぶ可能性がある。Ｌｉｎｕｘシステムでは、適合し、カスタマイズすることが意図される「プラグ可能認証モジュール」によってそこでユーザ認証を行うので、そのような共有サーバに対するクライアント・サイドを比較的容易に構築することができる。ここでは、共有メモリ・インタフェースまたはメモリからメモリへのデータ・ムーバ・インタフェースを介してセキュリティ・サーバにアクセスし、ウェブ・サーバはそのアクセスを求めて競争する。その結果生じる作業キューはセキュリティ・サーバによって実行し、必要に応じて共有メモリ・インタフェースを通じて応答を返す。この結果、ウェブ・アプリケーションに対する強化されたセキュリティとパフォーマンスが発揮される。図１７を参照すると、セキュリティ・サーバ（１６０１）は、共有メモリ（１６

１１）を通じたユーザ・プロセス（１６０３）からのアクセス要求に応答する。ユーザ・プロセスは、カーネル２（１６０７）のセキュリティ・クライアント・プロセス（ＬＩＮＵＸの場合にはこれがＰＡＭとなる）との標準的なプロセス間通信（ＩＰＣ）インタフェースを使用し、セキュリティ・クライアント・プロセスは、共有メモリ（１６１０）を通じてカーネル１（１６０５）のカーネル・プロセスと通信し、カーネル・プロセスは、セキュリティ・サーバ・インタフェース（ＯＳ／３９０またはＺ／ＯＳの場合はＳＡＦ）をユーザ・プロセス（１６０３）のプロキシとして駆動し、共有メモリ（１６１０）を通じてカーネル２（１６０７）のセキュリティ・クライアントに認証を返す。

【００８０】

別の実施形態では、共有メモリに入れられたデータは、単一操作のデータ・ムーバを介してカーネル・メモリ１（１６０６）からカーネル・メモリ２（１６０８）に移動し、共有メモリの開発を回避するとともに、ネットワーク接続も回避する。

【００８１】

以下で、共通のセキュリティ・サーバ（１６０１）を第１のパーティション（１６１４）で実行し、少なくとも１つのセキュリティ・クライアント（またはプロキシ）（１６０３）を少なくとも１つの第２のパーティション（１６１５）で実行する、区分された処理システムにセキュリティを提供するセキュリティ・サーバにおける通信ステップの実装の一例を挙げる。

【００８２】

ユーザが認証を要求する。セキュリティ・クライアント（１６０３）がユーザからパスワードを受け取る。セキュリティ・クライアントは、セキュリティ・サーバ（１６１０）からのアクセスが可能なメモリ位置に要求を入れ、要求を入れたことを通知する。第１のパーティション（１６１４）の「セキュリティ・デーモン」が通知を認識し、第１のパーティション（１６１４）の「プロキシ」クライアント（１６１６）を開始する。プロキシ（１６１６）クライアントは、セキュリティ・サーバ（１６０１）にとってネイティブのインタフェースを使用して、その要求をもってセキュリティ・サーバを呼び出す。セキュリティ・サーバ（１６０１）は、要求を処理し、プロキシ・クライアント（１６１６）にサーバ応答を返す。プロキシ・クライアントは、セキュリティ・サーバの応答を第２のパーティションのセキュリティ・クライアントによるアクセスが可能なメモリに入れ、要求を入れたことを通知する。この通知により、その権限付与をポイントするセキュリティ・クライアント（１６０３）が起動する。セキュリティ・クライアント（１６０３）は、ユーザに応答を戻す。一実施形態では、第２のパーティション（１６１５）のセキュリティ・クライアント（１６０３）は、共有メモリ・インタフェース（１６０９）を用いて第１のパーティション（１６１４）のセキュリティ・サーバ（１６０１）と通信し、したがってネットワーク接続のセキュリティの露出を防止し、パフォーマンスを高める。別の実施形態では、第２のパーティションのセキュリティ・クライアントは、図９に示すデータ・ムーバ（８２１）を使用した内部のメモリからメモリへの移動により、第１のパーティションのセキュリティ・サーバと通信する。図９を参照すると、この第２の実施形態は、セキュリティ・クライアントをプロセスＡ（８０３）として実装し、セキュリティ・プロキシはプロセスＢ（８０１）として実装され、したがって外部のネットワーク接続が回避され、共有メモリの実装が回避される。

【図面の簡単な説明】

【００８３】

【図１】区分データ処理システムの概要の図である。

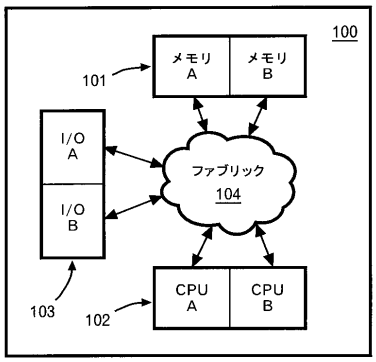
【図２】１つまたは複数のシステム・ボードからなるパーティションを有する、物理的に区分された処理システムの図である。

【図３】論理的に区分されたリソースが各自の個々のパーティションのために確保された、論理的に区分された処理システムの図である。

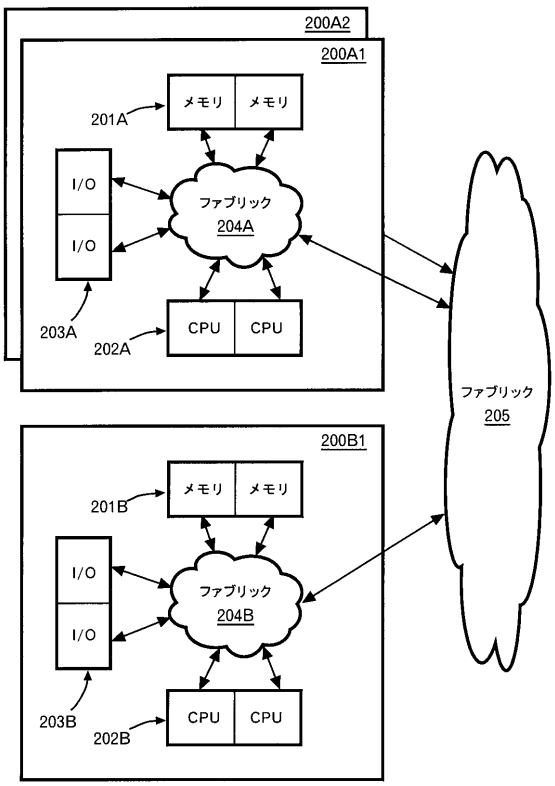
【図４】論理的に区分されたリソースをいくつかのパーティション間で動的に共有することが可能な、論理的に区分された処理システムの図である。

【図５】ＵＮＩＸ（Ｒ）オペレーティング・システムの「プロセス間通信」の構造の図である。
【図６】スタンドアロン・ユーティリティによってロードされた構成テーブルに従って実際のメモリが共有される実施形態の図である。
【図７】Ｉ／Ｏアダプタおよびそのドライバの機能を使用してパーティション間のデータ転送を容易にする実施形態の図である。
【図８】従来技術によるシステムの図である。
【図９】区分データ処理システムの通信ファブリック中に実装されたデータ・ムーバによってパーティション間の実際のデータ転送を行う実施形態の図である。
【図１０】データ・ムーバ例の構成要素の図である。
【図１１】ＩＢＭ　Ｓ／３９０移動命令の例示的フォーマットの図である。
【図１２】アダプタ・データ・ムーブを行う例示的ステップの図である。
【図１３】プロセッサ・データ・ムーブを行う例示的ステップの図である。
【図１４】ワークロード・マネジャ（ＷＬＭ）の高レベル図である。
【図１５】典型的なワークロード管理データを示す図である。
【図１６】間接的なＩ／Ｏを使用したクライアント／サーバのクラスタ化の図である。
【図１７】クライアント／サーバのサーバ・クラスタ化の図である。

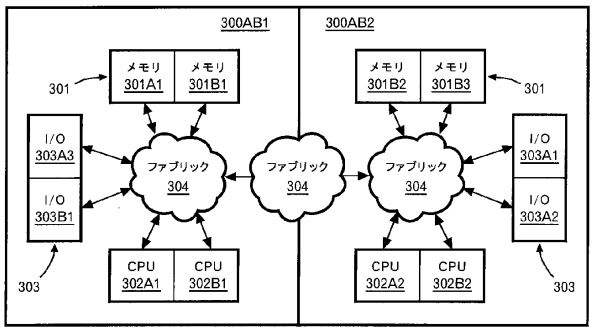
【図１】



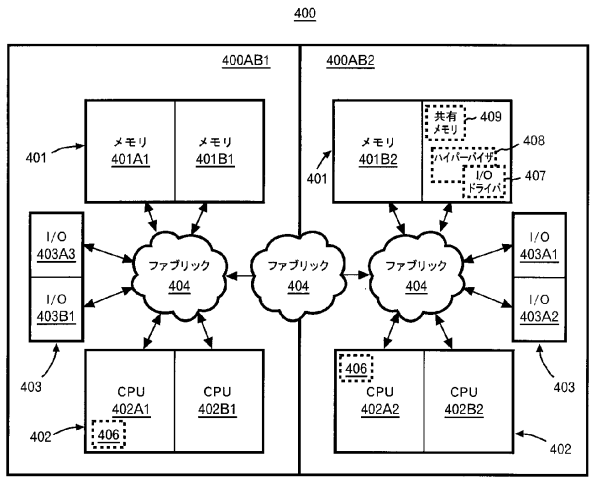
【図２】



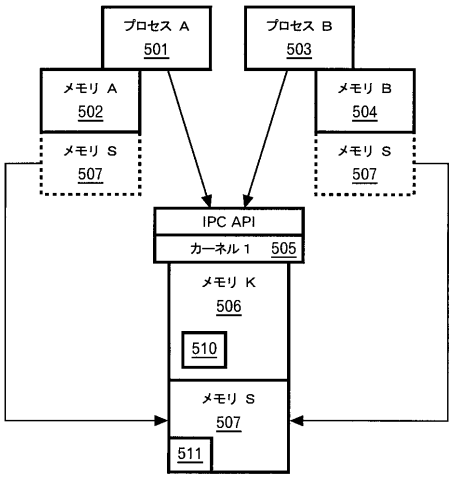
【図３】


300AB1
300AB2
メモリ 301A1
メモリ 301B1
メモリ 301B2
メモリ 301B3
301
I/O 303A3
I/O 303B1
I/O 303A1
I/O 303A2
303
ファブリック 304
CPU 302A1
CPU 302B1
CPU 302A2
CPU 302B2

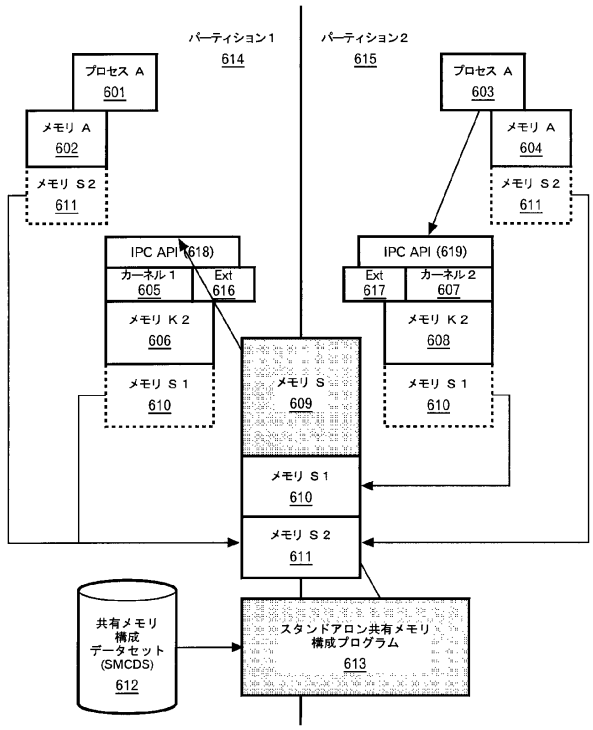
【図４】


400
400AB1
400AB2
メモリ 401A1
メモリ 401B1
メモリ 401B2
401
共有メモリ 409
ハイパーバイザ 408
I/O ドライバ 407
I/O 403A3
I/O 403B1
I/O 403A1
I/O 403A2
403
ファブリック 404
CPU 402A1
CPU 402B1
CPU 402A2
CPU 402B2
402
406

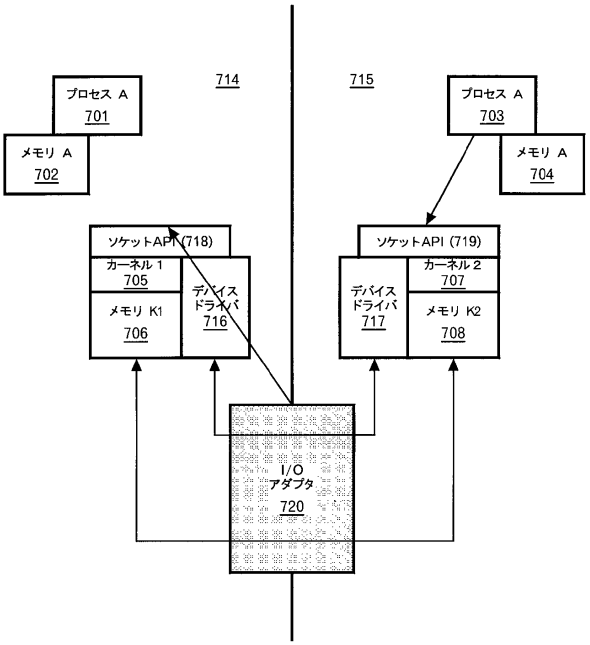
【図５】


プロセス A 501
プロセス B 503
メモリ A 502
メモリ B 504
メモリ S 507
IPC API
カーネル 1 505
メモリ K 506
510
メモリ S 507
511

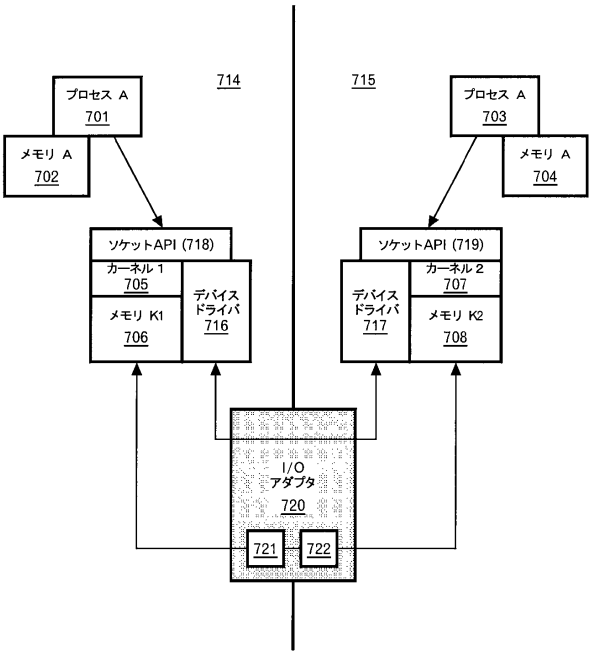
【図６】


パーティション1 614
パーティション2 615
プロセス A 601
プロセス A 603
メモリ A 602
メモリ A 604
メモリ S2 611
IPC API (618)
カーネル 1 605
Ext 616
IPC API (619)
Ext 617
カーネル 2 607
メモリ K2 606
メモリ K2 608
メモリ S1 610
メモリ S 609
メモリ S1 610
メモリ S2 611
共有メモリ構成データセット (SMCDS) 612
スタンドアロン共有メモリ構成プログラム 613

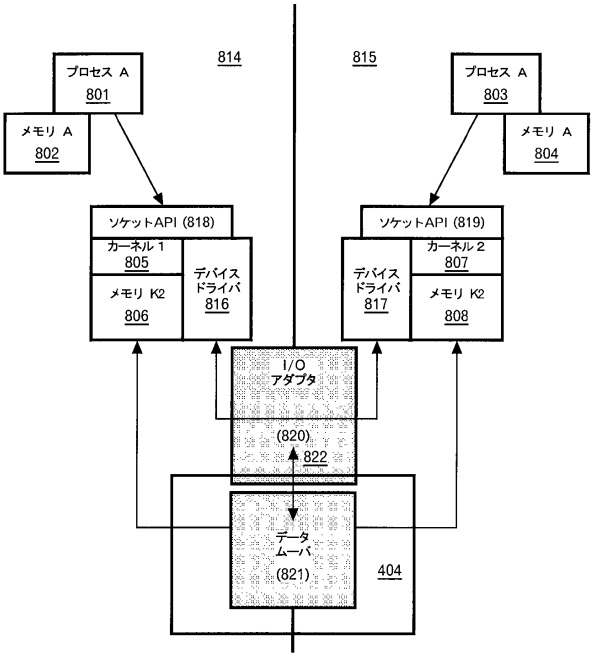
【図７】


714
715
プロセス A 701
メモリ A 702
プロセス A 703
メモリ A 704
ソケットAPI (718)
カーネル 1 705
メモリ K1 706
デバイスドライバ 716
ソケットAPI (719)
カーネル 2 707
デバイスドライバ 717
メモリ K2 708
I/O アダプタ 720

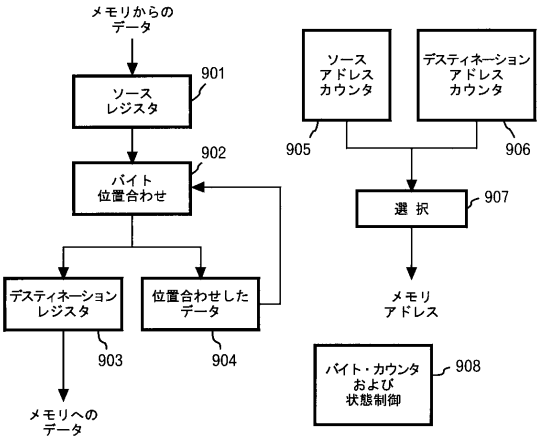
【図８】



【図９】



【図１０】



【図１１】

1000

MVXL	バイト数	デスティネーション	ソース

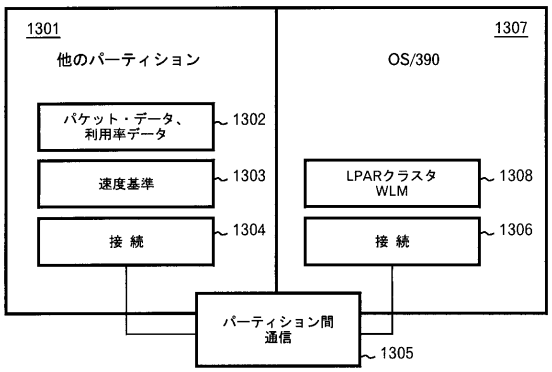
【図１２】

1101. ユーザがデバイス・ドライバを呼び出し、
－ソース・ネットワークID、
ソース・オフセット、
宛先ネットワークのIDを提供する
1102. デバイス・ドライバがアドレスをアダプタに転送する
1103. アダプタがアドレスを変換する
－IDから物理的な規定アドレスを検索する(テーブル・ルックアップ)
－ロックと現在の宛先オフセットを得る
－オフセットを加算する
－境界を確認する
1104. アダプタがレジスタにカウントとアドレスをロードする
1105. アダプタがデータ移動を実行する
1106. アダプタがロックを解除する
1107. アダプタが、ユーザに「戻る」デバイス・ドライバを通知する

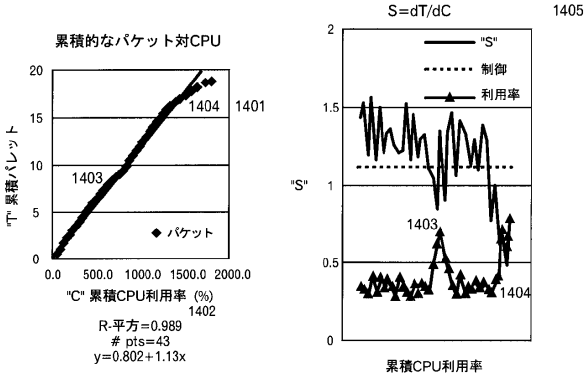
【図１３】

1201. ユーザがデバイス・ドライバを呼び出し、
－ソース・ネットワークID、
ソース・オフセット、
宛先ネットワークのIDを提供する
1202. デバイス・ドライバがアドレスをアダプタに送る
1203. アダプタが変換を行う
－IDから物理的な規定アドレスを検索する(テーブル・ルックアップ)
－ロックと現在の宛先オフセットを得る
－オフセットを加算する
－境界を確認する
－ロックと物理アドレスをデバイス・ドライバに戻す
1204. デバイス・ドライバがデータ移動を実行する
1205. デバイス・ドライバがロックを解除する
1206. デバイス・ドライバが戻る

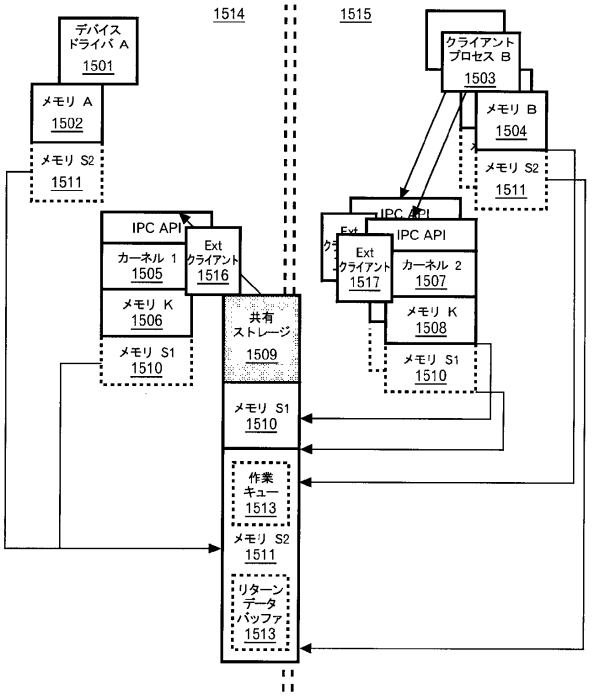
【図１４】



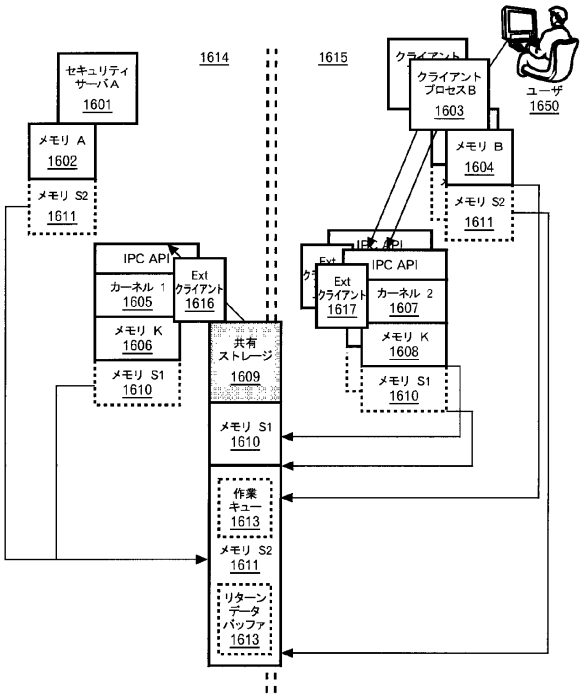
【図１５】



【図１６】



【図１７】



フロントページの続き

(72)発明者 クバラ、ジェフリー
アメリカ合衆国１２５７０ ニューヨーク州プーカグ モーガン・レーン １０
(72)発明者 ニック、ジェフリー
アメリカ合衆国１２４９３ ニューヨーク州ウエスト・パーク ルート ９ ダブリュー １９５７
(72)発明者 テンプル、ジョゼフ、サード
アメリカ合衆国１２４４３ ニューヨーク州ハーレー フック・ストリート ３１２ 私書箱５０７
(72)発明者 ヨコム、ピーター
アメリカ合衆国１２５４０ ニューヨーク州ラグランジェビル ユニット ２９ ストリンガム・ロード １２９

審査官 殿川 雅也

(56)参考文献 特開２０００－１３２５３０（ＪＰ，Ａ）
特表２００２－５３２８０６（ＪＰ，Ａ）
特開平１０－３３４０５７（ＪＰ，Ａ）
特開平１０－０２７１６７（ＪＰ，Ａ）
特開平０７－２９５８４１（ＪＰ，Ａ）
特開昭６４－００２１４５（ＪＰ，Ａ）
国際公開第００／０３６５０９（WO，Ａ１）
特許第２６３８０６５（ＪＰ，Ｂ２）
Kinshuk Govil et al., Cellular disco: resource management using virtual clusters on shared-memory multiprocessors, ACM Transactions on Computer Systems (TOCS), ２０００年８月, Volume 18, Issue 3, pp.229-262

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