【公報種別】特許法第１７条の２の規定による補正の掲載
【部門区分】第６部門第３区分
【発行日】平成21年1月15日(2009.1.15)

【公表番号】特表2002-507022(P2002-507022A)
【公表日】平成14年3月5日(2002.3.5)
【出願番号】特願2000-535993(P2000-535993)
【国際特許分類】
Ｇ０６Ｆ　11/00　　(2006.01)
Ｇ０６Ｆ　 9/46　　(2006.01)
【ＦＩ】
Ｇ０６Ｆ　 9/06　　６３０Ｃ
Ｇ０６Ｆ　 9/46　　３４０Ａ

【誤訳訂正書】
【提出日】平成20年11月20日(2008.11.20)
【誤訳訂正１】
【訂正対象書類名】明細書
【訂正対象項目名】全文
【訂正方法】変更
【訂正の内容】
【書類名】　　明細書
【発明の名称】　ソフトウェア更新のための状態コピー方法
【特許請求の範囲】
　【請求項１】　更新機能を備えるタイプのソフトウェア処理デバイスであって、
　ａ）ａ１）更新プロセス中に連続して実行される第１のグループのソフトウェア・モジュールおよび関連データを蓄積する実行メモリ分割部（６）と、
　　ａ２）前記第１のグループのソフトウェア・モジュールが並行して実行される間に初期化される第２のグループのソフトウェア・モジュールおよび関連データを蓄積する待機メモリ分割部（１６）と
に分割されるメモリ手段（６，１６）を具備し、
　ｂ）前記待機メモリ分割部（１６）における新しいソフトウェアのデータ状態を、前記実行メモリ分割部（６）における古いソフトウェアのデータ状態に、その古いソフトウェアの実行の続行中に更新するよう適合される更新制御手段（２４）と、
　ｃ）前記実行メモリ分割部（６）から前記待機メモリ分割部（１６）への部分的なデータの転送のための転送手段（２６）と
を含むことを特徴とするソフトウェア処理デバイス。
　【請求項２】　前記更新制御手段（２４）が、切り替え手段（３２）およびデータ状態比較手段（２８）を更に具備し、前記データ状態比較手段（２８）によって、前記待機メモリ分割部（１６）および前記実行メモリ分割部（６）について同じデータ状態が検出されると、新しいソフトウェアの実行に切り替えることを特徴とする請求項１に記載のソフトウェア処理デバイス。
　【請求項３】　各メモリ分割部（６，１６）に少なくとも１つの引継手段（８，１８）が指定されていて、古いソフトウェアに関するデータが、その引継手段（８，１８）が切り替え後に作動されるよう部分的にのみ転送されるならば、デフォルト動作を実行することを特徴とする請求項１または２に記載のソフトウェア処理デバイス。
　【請求項４】　前記転送手段（２６）が、データを変更しないまま、または前記新しいソフトウェアのための新しい表現への変換の後に、コピーすることを特徴とする請求項１乃至３の１つに記載のソフトウェア処理デバイス。
　【請求項５】　前記転送手段（２６）が、前記実行メモリ分割部（６）における古い

ソフトウェアの実行と並行してかつそれを外乱することなく、データの変換を実行することを特徴とする請求項４に記載のソフトウェア処理デバイス。

【請求項６】　前記転送手段（２６）が専用の変換手段からなることを特徴とする請求項４または５に記載のソフトウェア処理デバイス。

【請求項７】　前記切り替え手段（３２）が、前記実行メモリ分割部（６）における変化するデータ状態の経過を追跡するために、新しいソフトウェアの実行に切り替えるまで、前記更新制御手段（２４）が、更新機能を繰り返し実行することを特徴とする請求項１乃至６の１つに記載のソフトウェア処理デバイス。

【請求項８】　前記更新制御手段（２４）が、エラー状態が切り替え前に生じるならば、前記実行メモリ分割部において古いソフトウェアの続行を命令することを特徴とする請求項１乃至７の１つに記載のソフトウェア処理デバイス。

【請求項９】　新しいソフトウェアの実行の間のエラーが切り替え後に生じるならば、古いソフトウェアを備える分割部が再度実行メモリ分割部（６）となるよう、スイッチ・バックを行うように前記切り替え手段（３２）が適合されていることを特徴とする請求項１乃至８の１つに記載のソフトウェア処理デバイス。

【請求項１０】　更新プロセス中に連続して実行される第１のグループのソフトウェア・モジュールおよび関連データを蓄積する実行メモリ分割部（６）と、前記第１のグループのソフトウェア・モジュールが並行して実行される間に初期化される第２のグループのソフトウェア・モジュールおよび関連データを蓄積する待機メモリ分割部（１６）とに分割されるメモリ手段（６，１６）とを含む、更新機能を備えるタイプのソフトウェア処理デバイスのための状態コピー方法であって、

ａ）前記第２のグループのソフトウェア・モジュールおよび関連データを初期化するために、古いソフトウェアの実行を続行しつつ、前記待機メモリ分割部（１６）における新しいソフトウェアのデータ状態を、前記実行メモリ分割部（６）における古いソフトウェアのデータ状態に前記ソフトウェア処理デバイスの更新制御手段（２４）により更新するステップと、

ｂ）前記待機メモリ分割部（１６）および前記実行メモリ分割部（６）について同じデータ状態が達成された時、前記ソフトウェア処理デバイスの切り替え手段（３２）により新しいソフトウェアの実行に切り替えるステップとを具備し、

前記更新ステップａ）が、

ｃ）前記ソフトウェア処理デバイスのソフトウェア・ローディング手段（３６）による、新しいソフトウェアの前記待機メモリ分割部（１６）へのローディングと、

ｄ）前記ソフトウェア処理デバイスの転送手段（２６）による、前記実行メモリ分割部（６）から前記待機メモリ分割部（１６）へのデータの転送と

に分かれることを特徴とする状態コピー方法。

【請求項１１】　前記実行分割部（６）から前記待機分割部（１６）へのデータの転送は、

ｅ）変更されないで転送されるデータのコピーと、

ｆ）新しいソフトウェアのための新しい表現に変換されるべきデータの変換と

に分かれることを特徴とする請求項１０に記載の状態コピー方法。

【請求項１２】　データの変換が、前記実行分割部（６）における古いソフトウェアの実行と並行してかつそれを外乱することなく行われることを特徴とする請求項１１に記載の状態コピー方法。

【請求項１３】　前記実行分割部（６）における変化するデータ状態の経過を追跡するために、新しいソフトウェアに切り替えるまで、前記更新ステップａ）が背景プロセスとして繰り返し実行されることを特徴とする請求項１０乃至１２の１つに記載の状態コピー方法。

【請求項１４】　切り替え時に転送されない古いソフトウェアに関するデータが存在するならば、このデータは、新しいソフトウェアの開始前に、必要に応じて転送されることを特徴とする請求項１０乃至１３の１つに記載の状態コピー方法。

【請求項１５】　切り替え前にエラー状態が生じるならば、更新が終結されて、前記実行分割部（６）において古いソフトウェアの実行が続行されることを特徴とする請求項１０乃至１４の１つに記載の状態コピー方法。

【請求項１６】　古いソフトウェアを備える実行分割部（６）が、新しいソフトウェアの実行の間のエラーが切り替え後に生じるならば、再度実行分割部（６）になるようにスイッチ・バック・ステップが行われることを特徴とする請求項１０乃至１５の１つに記載の状態コピー方法。

【請求項１７】　前記スイッチ・バックが、外乱が制限されまたは全く無く、必要に応じて行われるデータのコピーおよび変換を伴うデータ転送を含むことを特徴とする請求項１６に記載の状態コピー方法。

【請求項１８】　前記スイッチ・バック・ステップが、古いソフトウェアの再開の前に実行される回復ステップを含むことを特徴とする請求項１６または１７に記載の状態コピー方法。

【発明の詳細な説明】

【０００１】

（発明の技術分野）

本発明は、ソフトウェア更新の技術分野に関し、より詳しくは、請求項１，１２，１５および２９の前文による、新たに挿入される機能および／または誤りの訂正のために度重なる更新を行う、コンピュータをベースとするシステムにおいての機能変更の技術分野に関する。

【０００２】

（発明の背景）

データ処理装置およびソフトウェア技術の発展によって、インストールされたソフトウェアを更新する方法論についての需要が増大しつつある。

ＷＯ９４／０１８１９において、コンピュータ・システムの進行する動作を外乱することなく、作動するコンピュータ・システムにおけるソフトウェアの置き換えのためのシステムが記述されている。初期局面の間、そのシステムは、全てのトラフィックを古いソフトウェアのバージョンまたは変更ユニットに向ける。ローディング局面の間、そのシステムは、新しいソフトウェア・バージョンとデータ変更情報を受け取る。そのデータ変更情報は、前記変更ユニットから半永久データを受け取り、置き換えの全局面の間中、連続してそのデータおよび関連する更新を転送する。テスト局面の間、そのシステムは、まず、テスト・トラフィックを新しいバージョンを通して向け、そして続いて、そのテスト・トラフィックが成功するならば、トラフィックを抽出する。そのテスト局面が成功するならば、完成局面が、全ての新しいトラフィックを新しいバージョンへと向け、そして変更ユニットを使用していた古いトラフィックのみが変更ユニットへと向けられる。一旦全ての古いトラフィックが、完了にまで至って終結されているならば、変更ユニットはもはや用いられておらず、データ変更情報で取り除くことができる。

【０００３】

この目的を達成するためのもう一つの通常の方法論は、インストールされたソフトウェアの実行を停止し、新しいソフトウェアをローディングし、続いて新しいソフトウェアを始動するというものである。このアプローチを用いると、内部データが、古いソフトウェアと新しいソフトウェアとの間で転送されない。また、この方法では、確立された全てのサービスが失われてしまい、新しいソフトウェアのローディングおよび始動の間、サービスが完全に停止される。現在、この方法は、典型的には、例えば、ワーク・ステーションやパーソナル・コンピュータに用いられている。

【０００４】

ソフトウェア更新の問題への別のアプローチは、先にＥＰ－Ａ－０２０１２８１に記述されている。しかしながら、この解決法では、必要なデータおよびメッセージの変換が、始動の間新しくインストールされたソフトウェアそれ自体を経て行われるので、外乱のないデータ更新機能を行うことができない。

【０００５】
更には、ＵＳ－Ａ－５１５５８３７において、第１のステップで、新しいサービスのためにデータの入力を新しいソフトウェアに切り替えることが提案されている。更には、古いソフトウェアにおいて進行中のサービスが完了したとき、そのサービスからのデータの出力が、古いバージョンから新しいバージョンへと切り替えられる。しかしながら、この解決法は、古いバージョンによるソフトウェアが、新しいソフトウェア・バージョンが完全に動作可能となる前に、最初に終えられなければならないので、非常に短い期間でサービスを取り扱うソフトウェアのみしか取り扱わない。

【０００６】
したがって、全ての周知のアプローチにおいて、ソフトウェア更新が行われるならば、システムの動作にある種の外乱がある。この外乱は、何時間ものあるいは可能性として何日もの間の全てのシステムの遮断から、数秒間の、可能性として全てのシステムの機能のある限定された部分に関してのみの、短期間の崩壊という範囲に渡り得る。考えられるところでは、全く外乱が無いこともあろうが、とは言っても、通常これは、例えば、遠隔通信の交換局のような、現実に存在するシステムにおいての場合ではない。

【０００７】
（発明の概要）
前述に照らして、本発明の目的は、外乱を最小にして行われ、実質的に全く外乱が無いところまで縮小(scaleable down)できるソフトウェアの更新へのアプローチを提供することである。

【０００８】
請求項１において定義される本発明によると、この目的は、第１のグループのソフトウェア・モジュールおよび関連データを蓄積する実行メモリ分割部、および第２のグループのソフトウェア・モジュールおよび関連データを蓄積する待機メモリ分割部にサブ分割されるメモリ手段と、前記待機メモリ分割部における新しいソフトウェアの状態を前記実行メモリ分割部における古いソフトウェアの状態にその古いソフトウェアを連続して実行する間に更新するよう適合される更新制御手段と、前記実行メモリ分割部から前記待機メモリ分割部へのデータのスケーラブル(scaleable)な転送のための転送手段とを具備する、更新機能を備えるタイプのソフトウェア処理デバイスによって達成される。

【０００９】
したがって、グレードアップされるべきシステムが、２つの論理分割部へと分割される。これらの分割部は、一組のプロセッサを用いて実現されてはならない。ここで本発明によると、実行分割部と言及される１つの分割部が、通常の実行を行う古いソフトウェアを含む。更には、新しいソフトウェアが、待機分割部と言及される別の分割部に実行ソフトウェアの実行を外乱することなくローディングされる。待機分割部におけるソフトウェアは、実行分割部におけるソフトウェアと同じ状態に更新され、待機分割部の新しいソフトウェアが、外乱無く通常のプログラムの実行を引き継ぐことができる。ここで、これは実行分割部からデータをコピーすることによって行われる。古いソフトウェアと新しいソフトウェアは同一のものではないので、データは新しいソフトウェアに適合する表現に変換されなければならない。本発明によると、これは実行分割部において続行する通常のソフトウェアの実行と並行してかつこれを外乱せずに行われる。

【００１０】
また、本発明によって古いソフトウェアの全てのデータを転送することが実際的ではないならば、古いソフトウェアからデータを部分的に転送することが可能である。これによって、本システムにおけるソフトウェア更新によって引き起こされる外乱の度合いのスケールを変えることができる。

【００１１】
本発明の好ましい実施例によると、更新制御手段はさらに、切り替え手段および状態比較手段を具備していて、待機分割部および実行分割部について同じ状態が検出される限り、状態比較手段によって、新しいソフトウェアの実行へと切り替える。

【００１２】
このように、本発明によると、古いソフトウェアから新しいソフトウェアへの切り替えには、古いソフトウェアの全てのデータにおいて表現される完全な状態をコピーし、必要に応じて同時に新しいソフトウェアへと変換することが必要である。したがって、本発明によると、全く外乱無しに新しいソフトウェアの実行を続けることが可能となる。更には、切り替え時に処理されない古いソフトウェアにおいて、プログラムの間にデータが存在するならば、それはコピーされ、必要に応じて新しいソフトウェアの始動の前に変換される。

【００１３】
本発明の好ましい実施例によると、各メモリ分割部には、古いソフトウェアに関するデータが、切り替え直後に特別な引継手段が作動されるように部分的にのみ転送されるならば、少なくとも１つの引継手段が指定され、デフォルト動作を実行する。

【００１４】
ここで、その特別な引継手段は、その切り替えに続いて直ちに作動され、完全なデータの入力を必要としないデフォルト動作を行う。この場合において、古いソフトウェアからのデータが失われている程度に幾分かの外乱がある一方、本発明によると、デフォルト動作の組み込みによって適切と見なされるものに従ってそのスケールが変えられる。

【００１５】
本発明の更に別の好ましい実施例によると、更新制御手段は、エラー状態が切り替えの前に生じるならば、実行分割部において古いソフトウェアの続行を命令し、切り替え後新しいソフトウェアの実行中にエラーが生じるならば、古いソフトウェアを備える分割部が再度実行分割部になるようにスイッチ・バックを行う。

【００１６】
ここで、切り替えの前にエラー状態が生じるならば、ソフトウェアのグレードアップは終了され、実行分割部において古いソフトウェアで通常のソフトウェアの実行が続行する。これに対して、切り替え後の新しいソフトウェアの実行中のエラーの場合は、古いソフトウェアを含む分割部が再度実行分割部となるようにスイッチ・バックが行われる。ここで、そのスイッチ・バック手順は、データ・コピーを含んでおり、必要に応じて切り替え手順と同様に変換する。したがって、そのスイッチ・バック手順はまた、外乱が制限されてまたは外乱無しに行われる。代わりに、それはまた、典型的には幾分かの外乱をもたらす回復手順を走行することによって、データ・コピーおよび変換無しに行われる。

【００１７】
更には、請求項１５において定義される本発明によると、前述で概略された目的はまた、待機分割部における新しいソフトウェアの状態を、実行分割部における古いソフトウェアの状態へと、古いソフトウェアの実行を続行しつつ更新し、待機分割部および実行分割部について同じ状態が達成されている限り、新しいソフトウェアの実行へ切り替えるステップを具備する少なくとも２つの論理分割部を備えた計算システムのための状態コピー方法で達成される。

【００１８】
したがって、本発明による方法を用いると、長期間存在するサービスを取り扱う古いソフトウェアがたとえまたあるとしても、ソフトウェアの高効率かつ外乱のない更新を達成することが可能である。

【００１９】
本発明の方法の好ましい実施例によると、更新ステップは更に、実行分割部において走行している古いソフトウェアと並行に、かつその外乱を受けることなく実行される初期化サブ・ステップを具備する。

【００２０】
したがって、新しいソフトウェアの更新には、結局は初期化ルーティンが続けられる。これは、部分的にはより早く、例えば、新しいソフトウェアのローディングの直後に、行われるが、この初期化の一部は、古いソフトウェアからのデータに依存しており、したが

って前もって行われることはない。新しいソフトウェアの初期化は、実行分割部において続行されている通常のソフトウェアの実行の外乱を最小限としてこれと並列して実行される。実行分割部の状態が、連続して変更されるので、本発明による外乱の無い更新プロセスもまた、初期化と並行して連続的に行われなければならない。

【００２１】

本発明の方法の更に別の好ましい実施例によると、更新ステップは、実行分割部における変更状態の経過を追うために、新しいソフトウェアへの切り替えまで背景プロセスとして繰り返し実行される。

【００２２】

古いソフトウェアの、表現通りの完全な状態および全てのデータがコピーされ、必要に応じて新しいソフトウェアに変換されるならば、新しいソフトウェアにおいて全く外乱無しに実行を続行することが可能である。古いソフトウェアにおいてプログラム間でデータの交換があり、それが切り替え時に処理されていないならば、それらもまたコピーされかつ必要に応じて変換されなければならない。

【００２３】

本発明の方法の更に別の好ましい実施例によると、古いソフトウェアに関係するデータは、部分的にのみ転送され、データの完全な入力を必要としないデフォルト動作を行うために、切り替え直後に特別な引継ステップが実行される。この場合は、幾分かの外乱があるであろう。この外乱の度合いは、古いソフトウェアからどれだけのデータが失われているかに依存する。利点として、それは原則的に適切と見なされるものに従ってスケールを変えることができる。

【００２４】

さらに、請求項２９において定義される本発明によると、新しいソフトウェアを遠隔プロセッサの第１の／待機メモリ分割部へと更新し、新しいソフトウェアの状態を、メイン・プロセッサにおけるソフトウェアの実行を続行しつつ、メイン・プロセッサの状態とのマッチングを達成するよう更新し、また遠隔プロセッサにおけるソフトウェアの実行を、メイン・プロセッサの状態とのマッチングが達成されるや否や、新しいソフトウェアへと切り替えるステップからなり、１つのメイン・プロセッサおよび少なくとも１つの遠隔プロセッサを具備する分散型計算環境のための状態コピー方法が提供される。

【００２５】

本発明によるこの修正された方法によって、特定のソフトウェア処理デバイスに蓄積されるソフトウェア・モジュールとは別の部分に関係するソフトウェア・モジュールの更新を達成することができる。

【００２６】

それはまた、ソフトウェアのみならずハードウェアの更新をも可能とする。特に、ソフトウェアの実行を、ソフトウェア処理デバイスのハードウェアの更新の間に、別のソフトウェア処理デバイスへと切り替えることを考慮することができる。

【００２７】

更にまた、本発明による方法を用いて、まずハードウェアの部分を変更し、そして続いてソフトウェアの部分を変更することによって、ソフトウェアとハードウェアの異なるソフトウェア処理デバイスでの組み合わされた更新を考慮することもできる。ここで、全ての構成要素を同時に変更する必要はなく、その結果として分散型システムの全域的な再開の必要性は無い。

【００２８】

（好ましい実施例の説明）

図１は、本発明によるソフトウェア処理デバイスの実施例について概略図を示す。ここで、本発明によるソフトウェア処理デバイスは、２つの分割部ＡおよびＢをそれぞれ有する。分割部Ａには、第１のプロセッサ・ユニット４、第１のメモリ分割部６および第１の引継ユニット８が備えられる。その第１のメモリ分割部は、第１のデータ蓄積セクション１０および第１のソフトウェア蓄積セクション１２に分割される。

【００２９】
更には、第２のプロセッサ・ユニット１４、第２のメモリ分割部１６および第２の引継ユニット１８をそれぞれ具備するＢサイドについて同じ構造が選択される。Ａサイドについて、第２のメモリ分割部１６は、第２のデータ蓄積セクション２０および第２のソフトウェア蓄積セクション２２に分割される。

【００３０】
図１に示されるように、サイドＡからサイドＢにまたはその反対にソフトウェアの更新をコーディネイトすることに加えて、プロセッサ・ユニット４および１４の双方並びに、第１のメモリ分割部６を第２のメモリ分割部１６に結合する転送ユニット２６を制御する更新制御ユニット２４が備えられる。

【００３１】
図１に示されるとおり、第１および第２の引継ユニット８および１８は、第１および第２のメモリ分割部６および１６にそれぞれ指定され、古いソフトウェアに関するデータが部分的にのみ転送されるならば、デフォルト動作を実行する。特に、そのようなデフォルト動作は、データの完全な入力を必要としない新しいソフトウェアに関係し、また例えば、特定の値へのデータ変数の初期化からなる。

【００３２】
以上、概説された通り、これによって、転送されないデータが引継ユニット８および１８をそれぞれ経て初期化されるので、転送ユニット２６はスケーラブルなレベルでデータを転送することが可能である。また、転送ユニット２６は、データを、変更しないままでか、または更新制御ユニット２４の制御の元で新しいソフトウェアのための新しい表現への変換の後に、コピーする。ここで、データの変換は、実行分割部における古いソフトウェアのセクションと並行にかつこれを外乱することなく実行される。また、更新制御ユニット２４および転送ユニット２６は、実行ソフトウェアが、実行分割部における古いソフトウェアの更なる実行の間、既に前もって転送されたデータを書いているならば、データ転送を繰り返すよう適合される。

【００３３】
また、更新制御ユニット２６は、エラー状態が切り替えの前に発生するならば、実行分割部における古いソフトウェアの続行を命令するよう適合される。別の選択は、古いソフトウェアを備える分割部は、新しいソフトウェアの実行の間のエラーが切り替えの後発生するならば、再度、実行される分割部となるようなスイッチ・バックである。

【００３４】
図２に示されるとおり、更新制御ユニットは、２方向的に実行される更新であって、メモリ分割部６および１６のいずれかが更新の間実行分割部としての役割を果たし、かつ他方の分割部１６または６が新しいソフトウェアがローディングされる待機分割部として役割を果たすものを達成する。この更新プロセスの間、データが実行分割部から待機分割部へと転送ユニット２６を経てスケーラブルな方法で転送される。

【００３５】
スケーラビリティ(scalability)を達成するために、図１に示される更新制御ユニット２４は、図２に示されるように構成される。更新制御ユニット２４は、状態比較ユニット２８、転送制御ユニット３０、切替ユニット３２、メモリ管理ユニット３４およびソフトウェア・ローディング・ユニット３６をそれぞれ具備する。状態比較ユニット２８によって、２つのメモリ分割部６および１６において、データとソフトウェアの状態を比較することが可能となる。更には、転送制御ユニット３０が備えられ、双方のメモリ分割部６および１６の間で、スケーラビリティがあり、柔軟性のあるデータまたはソフトウェアの転送がそれぞれ達成される。切替ユニット３２は、状態比較ユニット２８が実行分割部および待機分割部について同じ状態を検出するや否や、サイドＡとサイドＢとの間で、またはその反対にソフトウェアの実行を切り替える。メモリ管理ユニット３４が備えられ、メモリ分割部６および１６のいずれかにコンパクト・メモリが割り振られまた割り振り解除され、およびまたその中に参照情報を保持する。最後に、ソフトウェア・ローディング・ユ

ニット３６は、新しいソフトウェアを各分割部６，１６のソフトウェア蓄積セクション１２，２２にローディングする役割を果たす。

【００３６】

前記で、本発明によるソフトウェア処理デバイスの基本構造が図１および図２について記述されている一方で、以下ではこれらの構成部材の機能性並びにそれらの相関関係を図３乃至図７に関して記述する。以下の記述によって、Ｂサイドについてのソフトウェアの更新が記述される一方で、これによって、Ａサイドへの反対の方向においてもまた実行して良い本発明を限定するよう解釈すべきものではない。

【００３７】

図３は、本発明による状態コピー方法の実行の基礎となる基本ステップを示す。図３に示される通り、ステップ１において、双方の分割部は、並行同期様式を実行しており、かつ例えば、同じソフトウェアを実行する。

【００３８】

更に、図３に示されるステップ２は、待機分割部における新しいソフトウェアのローディングに関係しており、一方、実行分割部における古いソフトウェアの実行が続けられる。更には、ステップ３は、実行分割部から待機分割部へのデータのコピーを行う。このステップ３に関してその下部に示されるとおり、待機分割部において、コピー・データもまた新しいソフトウェアに適合する表現へと変換される。ここで、データのコピーおよび変換は、実行分割部における古いソフトウェアの実行と並行にかつこれを外乱することなく実行される。また、本発明によると、データのコピーおよび変換は、専用のソフトウェアまたはハードウェアによって実行される。

【００３９】

図３に示される通り、ステップ４において、実行分割部において走行する古いソフトウェアに並行にかつそれを外乱することなく実行される初期化もまた行われる。ここで、初期化ステップは、ステップ２において新しいソフトウェアを待機分割部にローディングした直後かまたは、ステップ３において古いソフトウェアからコピーされるデータに依存する場合には、できるだけ早く実行される。

【００４０】

上で既に概説された通り、古いソフトウェアに関係するデータは、部分的にのみ転送され、特別な初期化ステップは、古いソフトウェアからの完全なデータの入力を必要としないデフォルト初期化動作を行うため、切り替え前にまたはその直後に実行される。

【００４１】

図３に示される通り、ステップ５で待機分割部において適切な状態が達成されるや否や、新しいソフトウェアの実行への切り替えが実行される。ここで、その切り替えは、双方のメモリ分割部において関連するソフトウェア・モジュールについて同じ状態が達成された直後に、単一ソフトウェア・モジュールについて実行されるということに注意すべきである。データの部分的な転送のみのために切り替え時に転送されない古いソフトウェアに関係するデータが存在するならば、このデータは、新しいソフトウェアの始動前に必要に応じてなおも転送される。

【００４２】

更には、図３に示されるとおり、ステップ３およびステップ４に関して、２つのメモリ分割部間のコピー・プロセスが、また待機分割部についての初期化ステップの間にもまた続行される。この理由は、更新プロセス中に連続して実行される古いソフトウェアは、既に前もって転送されたデータに書き込むからである。したがって、更新プロセスは、実行分割部の変化する状態（changing state）の経過を追跡するために、新しいソフトウェアに切り替えるまで、背景プロセスとして繰り返し実行される。この繰り返される更新プロセスは、待機分割部のための初期化ステップに並行して実行される。

【００４３】

図４は、図３で説明された更新プロセスによるフローチャートを示す。特に、ステップ１およびステップ２の後、新しいソフトウェアをローディングし、それに関係する蓄積を

初期化するために、切り替えが起こるまで背景プロセスが連続して繰り返されるということが分かる。ここで、その背景プロセスはまた、それを複数の背景プロセスに分割することによって実現されるということに注意すべきである。古いソフトウェアと新しいソフトウェアとについて同じ状態が検出されるならば、転送されるべきデータが残っているかどうか判断するための呼び掛け信号に続いて瞬間の切り替えが起こり、それによって、古いソフトウェアの実行へのループ・バックが必要である。

【００４４】

以下において、本発明による状態コピー方法についてのより特定の例が、図５および図６に関して記述される。図５は、状態図を用いてメモリ分割部の状態の表現を示し、また、図６ａ乃至図６ｅは、そのような状態図の状態コピー方法の間の修正を示す。

【００４５】

図５に示される通り、メモリ分割部における状態は、ノードおよびエッジをそれぞれ具備する状態図を用いて表される。ここで典型的には、ノードはデータの異なる状態を表し、またエッジはそのエッジに指定されるソフトウェア・モジュールの実行によって異なるデータ状態間の転送を表す。一例は、最大のノードが、入力データ処理ソフトウェア・モジュールによる更なる処理に適合するデータに転送される入力データに関係するというものである。また、２つのエッジを有していてそれらの間で走行するノードは、１方のソフトウェア・モジュールの出力データが、他方のソフトウェア・モジュールへの入力データを表し、かつその反対でもある２つのソフトウェア・モジュールの相互作用を表す。

【００４６】

図６に示される通り、この表現は、図３に示される異なるステップを表すのに良く適合される。特に、図６ａは、更新プロセスが開始する前の実行および待機分割部における同じソフトウェアの実行の同時並行同期様式を表す。そして、図６ｂに示されるとおり、ステップ２における新しいソフトウェアのローディングの間、エッジとして表される異なるソフトウェアのモジュールの相互作用が中断され、新しいソフトウェアのローディングが始まる。図６ｂに示される通り、データは、既に上で概説したように異なるカテゴリーにサブ分割される。ここで黒いノードは、古いソフトウェアから同一のものとしてコピーされるべき新しいソフトウェアにおけるデータを表す。これに対して、白で表されたノードは、古いソフトウェアのデータに全く依存しない新しいソフトウェアのデータに関係する。１つの典型は、データ構造の修正のために新たに導入されるデータである。ハッチングで表される別のカテゴリのノードは、新しいソフトウェアに適合されるような変換を必要とするデータに関する。グレーで表される更なる差異は、新しい分割部に転送されるべきデータの量を低減するための引継機構を更に用いて、データが古いソフトウェアから部分的にのみコピーされ、または変換されている。全体として、図６ｃに示されるように、最後の３つのカテゴリについてのみ、実行および待機分割部間でデータがコピーされ、かつ変換される。

【００４７】

図３に示されるステップ４の結果が、図６ｄを通して表される。新しいソフトウェアの初期化の後、異なるデータ構成要素の相互関係が再度導入される。図３および図４に関して既に概説されたように、本発明による状態コピー方法は、更新プロセスの間、古いソフトウェアによってデータが再書き込みされるならば反復される。したがって、図６ｄは、ステップ４における初期化の後もコピー／変換が続けられる切り替えの前の状態を示す。ステップ５において切り替えが行われた後、データのコピー／変換を表すこれらの円弧は、図６ｅに示される通りもはや存在していない。切り替えが行われた後、状態は、再度、前述された並行同期様式に対応する。

【００４８】

したがって、状態コピー方法において、状態は古いソフトウェアから新しいソフトウェアにコピーされ、結局、全体の状態は、新しいバージョンと古いバージョンとにおいて定義される。原則として、状態は、双方のバージョンについて完成されるので、いずれのソフトウェア・バージョンにおいても実行を続行できる。状態コピー方法について重要であ

るのは、更新機能それ自体を除いて、実行分割部および待機分割部において進行する並行のソフトウェアの実行は決して無い。

【００４９】

本発明の状態コピー方法によると、エラー状態が生じるならば、切り替え前に更新プロセスを終了することも、また、古いソフトウェアの実行を続行することも可能である。また、切り替えの後、新しいソフトウェアの実行中にエラーが生じるならば、さらにスイッチ・バックを実行して、古いソフトウェアが再度実行されるものとすることもまた可能である。このスイッチ・バックは、上で概説されたタイプのデータ・コピーおよび変換を備えたデータ転送を含む。

【００５０】

前述で、本発明の状態コピー方法が、ソフトウェア処理デバイスに関して記述されている一方で、以下では、状態コピー方法の分散型計算環境へのアプリケーションが、図７乃至図１２に関して記述される。

【００５１】

図７に示される通り、分散型計算環境は、メイン・プロセッサ３８および遠隔プロセッサ４０を具備する。典型的には、メイン・プロセッサ３８は、図１に示され、部分的にのみ図７に示される構造を有する。更には、遠隔プロセッサ４０のメモリ分割部４６に、ソフトウェアを事前ローディングするオプションを少なくとも有さなければならない遠隔プロセッサ４０が備えられる。代わりに、また遠隔プロセッサ４０は、図９に示されるような本発明のソフトウェア処理デバイスの構造を有しても良い。メイン・プロセッサ３８と遠隔プロセッサ４０は、接続線４２を通して結合される。各遠隔プロセッサには、遠隔プロセッサ４０およびメイン・プロセッサ３８との相互作用においての更新をコーディネイトする少なくとも１つの更新手段４４が備えられる。

【００５２】

ここで図７は、分散型計算環境内で、本発明の状態コピー方法を用いる最初の場合を示す。ここで、遠隔プロセッサ４０のソフトウェアのみが更新されて、新しいソフトウェアが最初に、遠隔プロセッサ４０のメモリ分割部４６に事前ローディングされるようになる。状態コピー方法を働かせるための２つの必要条件は、遠隔プロセッサ４０によって、新しいソフトウェアのローディングの間にサービスが可能となるように事前ローディングができるものとなることおよび、ローディングの後、データがメイン・プロセッサ３８から更新されることである。この場合には、ソフトウェアは、分散型計算環境の大域的な再開を行わずに、遠隔プロセッサ４０において更新される。この目的で、一旦新しいソフトウェアが遠隔プロセッサ４０にインストールされると、遠隔プロセッサ４０におけるメモリ分割部４６の状態が、メイン・プロセッサ３８においてソフトウェアの実行を続行しつつ、メイン・プロセッサ３８におけるメモリ分割部の状態に更新される。最後に、遠隔プロセッサ４０におけるソフトウェアの実行は、メイン・プロセッサ３８の状態とのマッチが達成されるや否や、新しいソフトウェアに切り替わる。

【００５３】

更に、状態コピー方法のためには、どのようなタイプのまたどれだけのソフトウェアが更新されるかによっては、遠隔プロセッサ４０に高速更新が必要である。ここで、決定的ではなくおよび／または限定された量のソフトウェアが更新されるならば、高速更新の要件は支配的ではない。このように、複数の遠隔プロセッサを更新するときでさえ、更新回数を更新プロセスのための中断時間と一貫させることが可能である。

【００５４】

図８は更に、ソフトウェアが、遠隔プロセッサ４０においてだけでなく、メイン・プロセッサ３８においてもまた更新され、かつその更新プロセスが、インタフェースの互換性(compatibility)に影響を及ぼさない場合を示す。ここで、ソフトウェアの更新は、まず上で概説されたように、遠隔プロセッサ４０においてソフトウェアを更新し、続いて上で記述された状態コピー方法を用いてメイン・プロセッサ３８においてソフトウェアを更新する２つのステップにおいて行われる。分散型計算環境において全ての遠隔プロセッサが

同時に更新されないならば、システムにおいて大域的な再開の必要はない。

【００５５】

図９は、図８において示されるのと同じ場合に関し、メイン・プロセッサ３８および遠隔プロセッサ４０におけるソフトウェアの更新の後、それらの間のインタフェースの互換性がない(incompatible)という点で異なっている。

【００５６】

この場合には、遠隔プロセッサ４０もまた、図１に関して上で概説されたのと同じ構造を有し、遠隔プロセッサ４０およびメイン・プロセッサ３８におけるソフトウェアの同時更新が、それらの間のインタフェースを修正して、メイン・プロセッサ３８および遠隔プロセッサ４０のそれぞれにおいて本発明の状態コピー方法を同時に実行することを通して達成されるようにすべきである。

【００５７】

ここで、分散型計算環境の決定的ではない部分が関係してくるならば、状態コピー方法は、変更されるべきシステムにおいての部分をブロック化し、そして同時にソフトウェアを更新し、更に最後に分散型計算環境において変更された部分を再度非ブロック化することによって用いられるべきである。データが、古いソフトウェアから新しいソフトウェアに転送されなければならないならば、コピー／変更は、新しいソフトウェアの開始および非ブロック化の前に行うべきである。これに対して、決定的な部分が、ソフトウェアの更新の間に関係してくるならば、遠隔プロセッサ４０は、更新プロセスの間、分散型計算環境の休止時間があまりに長くなることを避けるために、新しいソフトウェアを事前ローディングすべきである。

【００５８】

更なるオプションは、遠隔プロセッサ４０における新しいソフトウェアが、メイン・プロセッサ３８からのデータで更新されるものである。また、古いソフトウェアから新しいソフトウェアへのデータの転送を支持する機能を、遠隔プロセッサ４０のために導入することもできる。

【００５９】

上記において異なるシステム構成におけるソフトウェアの更新が、本発明の状態コピー方法を用いて考慮されている一方で、以下においてはハードウェアおよびソフトウェアのグレードアップされた組合せが、図１０乃至図１２に関して説明される。

【００６０】

図１０は、メイン・プロセッサ３８におけるハードウェアの更新に関する。典型的には、ハードウェア構成要素は、交換されるべきハードウェア構成要素をブロック化し、続いてそれらを置き換え、また最後にそれらを再度非ブロック化することによって交換される。

【００６１】

図１１は、ソフトウェアが、遠隔プロセッサ４０およびメイン・プロセッサ３８の双方において、インタフェースの互換性に影響を与えることなく更新される次の場合を示す。更には、図１１に示される場合においてもまた、遠隔プロセッサ４０に指定されているハードウェアが交換されるべきである。これまで遠隔プロセッサ４０に指定されるその他の構成部品は、図１０に記述されるアプローチを用いてまず交換される。そして、遠隔プロセッサ４０およびメイン・プロセッサ３８の双方におけるソフトウェアの交換は、図８に関して記述されるアプローチを用いて実現される。

【００６２】

図１２は、遠隔プロセッサ４０に割り当てられたハードウェア構成部品が、遠隔プロセッサ４０およびメイン・プロセッサ３８におけるソフトウェアの更新と同時に交換され、更新後のソフトウェアに対して非互換性(incompatibility)となる状態コピー方法のアプリケーションについて更なる場合を示す。ここで、遠隔プロセッサ４０に関するハードウェアとソフトウェアの変更が、遠隔プロセッサ４０内で、かつ新しいハードウェアおよびソフトウェア構成部品に関して非互換性をもたらすものとならないならば、遠隔プロセッ

サ４０でハードウェアがまず変更され、そして続いて、図９に関して上で概説されたようにソフトウェアの更新が実行される。

【００６３】

これに対して、遠隔プロセッサ４０におけるハードウェア構成部品の交換がまた、遠隔プロセッサにおける更新されたソフトウェアに関して非互換性をもたらすものとなるならば、状況はさらに複雑である。ここで、ハードウェアとソフトウェアの変更が、分散型計算環境における性能に関して危機的なものでないならば、図１１に関して記述されるのと同じアプローチを用いることができる。

【００６４】

しかしながら、このハードウェアの変更が危機的であるならば、それぞれのハードウェアの構成部品は、遠隔プロセッサ４０で二重に備えられ、またソフトウェアは、図７および図８に従って遠隔プロセッサ４０に事前ローディングされるか、または遠隔プロセッサ４０は２つのサイドに分割されるかすべきである。この場合について別の必要十分条件は、遠隔プロセッサ４０の処理スピードが十分速いということである。これらの条件が満たされるならば、過度のシステム休止時間を取らずに、組み合わされた更新を行うことが可能となる。

【図面の簡単な説明】

本発明の好ましい実施例は、添付された図面について記述されるが、それらの図面は以下の通りである。

【図１】

本発明によるソフトウェア処理デバイスの概略図を示す。

【図２】

図１に示される更新制御ユニットの概略図を示す。

【図３】

本発明による状態コピー方法についての図を示す。

【図４】

図３において示される状態コピー方法によるフローチャートを示す。

【図５】

ソフトウェア処理デバイスにおける１つの分割部の状態を表す状態図を示す。

【図６ａ】

図３において示されるステップ１による双方の分割部においてのソフトウェアの実行のための並行同期様式を示す。

【図６ｂ】

図３において示されるステップ２による双方の分割部においての状態を示す。

【図６ｃ】

図３において示されるステップ３による双方の分割部においての状態を示す。

【図６ｄ】

図３において示されるステップ４による双方の分割部においての状態を示す。

【図６ｅ】

図３において示されるステップ５による双方の分割部においての状態を示す。

【図７】

事前ローディング能力を有する遠隔プロセッサを備える分散型環境におけるソフトウェアの更新への本発明によるアプローチを示す。

【図８】

遠隔プロセッサを有する分散型計算環境におけるソフトウェアの更新であって、そのソフトウェアの更新後のそこへのインタフェースの互換性に影響の無いものを示す。

【図９】

遠隔プロセッサを有する分散型計算環境におけるソフトウェアの更新であって、そのソフトウェアの更新後のそこへのインタフェースの互換性に影響のあるものを示す。

【図１０】

分散型計算環境におけるメイン・プロセッサのためのハードウェアの更新への本発明によるアプローチを示す。

【図１１】

分散型計算環境の遠隔プロセッサにおけるハードウェアおよびソフトウェアの更新であって、その更新の後そこへのインタフェースの互換性に影響の無いものへの本発明のアプローチを示す。

【図１２】

分散型計算環境の遠隔プロセッサにおけるハードウェアおよびソフトウェアの更新であって、その更新の後そこへのインタフェースの互換性に影響のあるものへの本発明のアプローチを示す。

【符号の説明】

２　ソフトウェア処理デバイス
４　Ａサイド・プロセッサー・ユニット
６　Ａサイド・メモリー分割部
８　Ａサイド引継ユニット
１０　Ａサイド・データ蓄積セクションおよびＡサイド・メモリー分割部
１２　Ａサイド・ソフトウェア蓄積セクションおよびＡサイド・メモリー分割部
１４　Ｂサイド・プロセッサー・ユニット
１６　Ｂサイド・メモリー分割部
１８　Ｂサイド引継ユニット
２０　Ｂサイド・データ蓄積セクションおよびＢサイド・メモリー分割部
２２　Ｂサイド・ソフトウェア蓄積セクションおよびＢサイド・メモリー分割部
２４　更新制御ユニット
２６　転送ユニット
２８　状態比較ユニット
３０　転送制御ユニット
３２　切り替えユニット
３４　メモリー管理ユニット
３６　ソフトウェア・ローディング・ユニット
３８　メイン・プロセッサ
４０　遠隔プロセッサ
４２　接続線
４４　遠隔プロセッサにおける更新手段
４６　遠隔プロセッサのメモリー分割部