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

特許第5328055号
(P5328055)

(45)発行日　平成25年10月30日(2013.10.30)　(24)登録日　平成25年8月2日(2013.8.2)

(51)Int.Cl.　　F I
G06F 9/52 (2006.01)　　G06F 9/46 472Z
G06F 12/00 (2006.01)　　G06F 12/00 518A
G06F 12/00 535Z

請求項の数 9　(全 12 頁)

(21)出願番号　特願2010-514980 (P2010-514980)
(86)(22)出願日　平成20年6月18日 (2008.6.18)
(65)公表番号　特表2010-532530 (P2010-532530A)
(43)公表日　平成22年10月7日 (2010.10.7)
(86)国際出願番号　PCT/US2008/067343
(87)国際公開番号　WO2009/006023
(87)国際公開日　平成21年1月8日 (2009.1.8)
審査請求日　平成23年6月9日 (2011.6.9)
(31)優先権主張番号　11/824,379
(32)優先日　平成19年6月29日 (2007.6.29)
(33)優先権主張国　米国 (US)

(73)特許権者　500046438
マイクロソフト　コーポレーション
アメリカ合衆国　ワシントン州　９８０５２－６３９９　レッドモンド　ワン　マイクロソフト　ウェイ
(74)代理人　100140109
弁理士　小野　新次郎
(74)代理人　100075270
弁理士　小林　泰
(74)代理人　100101373
弁理士　竹内　茂雄
(74)代理人　100118902
弁理士　山本　修
(74)代理人　100153028
弁理士　上田　忠

最終頁に続く

(54)【発明の名称】メモリトランザクションのグループ化

(57)【特許請求の範囲】
【請求項１】
トランザクションの集合を、複数のトランザクショングループのうちの１つに配置するステップと、
プログラムの複数のトランザクションの並行実行トランザクションの間で競合を検出するように構成された競合管理機構を提供するステップであって、
前記トランザクションの集合は、前記複数のトランザクションからの集合であり、
前記複数のトランザクションのそれぞれは、共有メモリにアクセスするように構成されるコードとして実装され、
前記トランザクションの集合は他のデータと重複しないデータにアクセスするように構成され、
前記トランザクションの集合に存在しない前記複数のトランザクション以外のトランザクションは前記他のデータにアクセスするように構成され、
前記複数のトランザクショングループのそれぞれは、複数のロックおよびバージョン管理機構のうちの１つを使用するように構成され、前記複数のロックおよびバージョン管理機構は、前記複数のトランザクション以外のトランザクショングループによって使用される前記複数のロックおよびバージョン管理機構以外のロックおよびバージョン管理機構と互換性がないものであり、
前記複数のトランザクション以外のトランザクショングループのうちの少なくとも１つは、前記複数のロックおよびバージョン管理機構以外のロックおよびバージョン管理機

構のうちの少なくとも１つを並行的に使用するように構成された、ステップと、
をコンピューターに実行させるためのコンピューターで実行可能な命令を格納することを特徴とするコンピューターで読み取り可能な記憶媒体。
【請求項２】
いくつかのロック機構がソフトウェアにより実施され、一方他のものがハードウェアを使用して実施されることを特徴とする請求項１に記載のコンピューターで読み取り可能な記憶媒体。
【請求項３】
トランザクショングループに命名することをコンピューターに可能にさせる命名機能を提供するステップをコンピューターに実行させるためのコンピューターで実行可能な命令をさらに格納することを特徴とする請求項１に記載のコンピューターで読み取り可能な記憶媒体。
【請求項４】
前記命名機能は、デバッガーにおいて使用されることを特徴とする請求項３に記載のコンピューターで読み取り可能な記憶媒体。
【請求項５】
前記命名機能は、プロファイラーにおいて使用されることを特徴とする請求項３に記載のコンピューターで読み取り可能な記憶媒体。
【請求項６】
前記トランザクショングループの少なくともいくつかは、自動的に割り当てられることを特徴とする請求項１に記載のコンピューターで読み取り可能な記憶媒体。
【請求項７】
前記トランザクションのグループ化機能は、前記トランザクショングループがコンパイラによって自動的に識別されることを可能にするように動作することを特徴とする請求項６に記載のコンピューターで読み取り可能な記憶媒体。
【請求項８】
前記トランザクションのグループ化機能は、前記トランザクショングループをランタイム環境によって自動的に識別することをコンピューターに可能にさせることを特徴とする請求項６に記載のコンピューターで読み取り可能な記憶媒体。
【請求項９】
トランザクションの集合を、複数のトランザクショングループのうちの１つに配置するステップと、
プログラムの複数のトランザクションの並行実行トランザクションの間で競合を検出するように構成された競合管理機構を提供するステップであって、
前記トランザクションの集合は、前記複数のトランザクションからの集合であり、
前記複数のトランザクションのそれぞれは、共有メモリにアクセスするように構成されるコードとして実装され、
前記トランザクションの集合は他のデータと重複しないデータにアクセスするように構成され、
前記トランザクションの集合に存在しない前記複数のトランザクション以外のトランザクションは前記他のデータにアクセスするように構成され、
前記複数のトランザクショングループのそれぞれは、複数のロックおよびバージョン管理機構のうちの１つを使用するように構成され、前記複数のロックおよびバージョン管理機構は、前記複数のトランザクション以外のトランザクショングループによって使用される前記複数のロックおよびバージョン管理機構以外のロックおよびバージョン管理機構と互換性がないものであり、
前記複数のトランザクション以外のトランザクショングループのうちの少なくとも１つは、前記複数のロックおよびバージョン管理機構以外のロックおよびバージョン管理機構のうちの少なくとも１つを並行的に使用するように構成された、ステップと、
を備えることを特徴とする、コンピューターにより実行される異なるトランザクション

グループと共に異なったロックおよびバージョン管理機構を使用するための方法。

【発明の詳細な説明】

【技術分野】

【０００１】

コンピューターは、より多くの処理能力およびメモリにより、時間とともに着実により一層強力になっており、高度化した演算を取り扱う。このような傾向は、最近、単一のプロセッサ（処理装置）のクロック速度を次々に増加させることに集中する方向から、単一のコンピューターで使用可能なプロセッサの数を増加させる方向に変化している。ソフトウェア開発者は、新しいハードウェアが採用されたときに、彼らのソフトウェアプログラムがより早く動作することを可能にし、コンピューター処理能力における改善を巧みに利用したいと欲する。しかしながら、上記の新しいハードウェアの傾向により、開発者は、特定のソフトウェアプログラムの１つまたは複数のタスクを、当該ソフトウェアが動作するコンピューターにより多くのプロセッサが追加されたときには、同一論理演算が同時に多くのプロセッサを利用して、より良い性能を提供可能なように、「並行に（ｃｏｎｃｕｒｒｅｎｔｌｙ）」（時々「並列に（ｉｎ　ｐａｒａｌｌｅｌ）」と呼ばれる）実行されるように準備しなければならないという従来と異なるアプローチを必要とする。

【背景技術】

【０００２】

トランザクションのメモリは、原子性（ａｔｏｍｉｃｉｔｙ）および独立性（ｉｓｏｌａｔｉｏｎ）をプログラムコードの領域に提供することによって並行プログラムの開発を緩和するように設計されている。ＴＭ（トランザクションのメモリ（ｔｒａｎｓａｃｔｉｏｎａｌ　ｍｅｍｏｒｙ））とは、並行コンピューティングにおいて共有メモリへのアクセスを制御するためのデータベーストランザクションと類似の並行性制御機構である。トランザクションのメモリという文脈におけるトランザクションとは、共有メモリに対する一連の読み出しおよび書き込みを実行する一つのコード（ａ　ｐｉｅｃｅ　ｏｆ　ｃｏｄｅ）である。ＴＭは、伝統的なロック機構に代わる手段として使用される。ＴＭは、並行プログラムをより簡単に書くことを可能にする。トランザクションは、コードのシーケンスを特定し、多くの並行的作業のために現実には通常のマルチスレッド環境において実行するのであるが、それがあたかも単独で実行しているようにする。この仮想的な独立性を、オブジェクトまたはメモリ領域に関するロックを細粒状化（ｆｉｎｅ－ｇｒａｉｎｅｄ　ｌｏｃｋｉｎｇ）すること、および、当該トランザクションが他のどれかのトランザクションと競合することが判明したなら当該トランザクションの効果を元に戻す（ｒｏｌｌｂａｃｋ）ことを可能にするモードで実行することによって実現することができる。データアクセスがこれらのロックおよび元に戻す機構によって保護されるなら、我々は、当該データアクセスが「トランザクションされる（ｔｒａｎｓａｃｔｅｄ）」と言う。

【発明の概要】

【発明が解決しようとする課題】

【０００３】

いくつかのソフトウェア準拠の、およびハードウェア準拠のアプローチを含む、いろいろなロックおよびバージョン管理機構が可能である。別の機構は、それぞれ別の状況において適当するまたは選ぶに値する特徴および品質を有する。単一の処理中に異なった機構を組み合わせることは、一般的な適用性を実現するために通常は性能において妥協した汎用機構を選択することになるため、一般には可能ではない。

【課題を解決するための手段】

【０００４】

トランザクションのメモリシステムの下で動作するプログラムにおいて使用するためのトランザクションのグループ化機能を提供するための様々な技術および技法が開示される。トランザクションのグループ化機能は、関連トランザクションを含むトランザクショングループを生成することを可能にするように動作する。トランザクショングループは、プログラムの動作を拡張するために使用される。トランザクショングループは、各グループ

におけるトランザクションが重複しないデータに対して動作することが知られるように定義され、これにより、当該各グループの中での互換性のないロックおよびバージョン管理機構が可能となり、特定の各グループに対する特定の機構の細やかな調整（ｆｉｎｅ－ｔｕｎｉｎｇ）が可能となる。

【０００５】

この概要は、以下の詳細な説明でさらに記述される選択された概念を単純化した形体で紹介するために提供される。この概要は、請求項に記載された構成の主要な特徴または本質的な特徴を特定することを意図するものではなく、また請求項に記載された構成の範囲の決定に資するように使用されることを意図するものでもない。

【図面の簡単な説明】

【０００６】

【図１】一実施方法のコンピューターシステムのブロック図である。

【図２】図１のコンピューターシステム上で動作する一実施方法のトランザクションのメモリアプリケーションのブロック図である。

【図３】図１のシステムの一実施方法に対する上位の処理フロー図である。

【図４】プログラマがトランザクションをグループ化することを可能にする際に関係する段階を示す、図１のシステムの一実施方法に対する処理フロー図である。

【図５】特定のヒューリスティックスに準拠しトランザクションを自動的にグループ化する言語コンパイラを提供する際に関係する段階を示す、図１のシステムの一実施方法に対する処理フロー図である。

【図６】トランザクションを自動的にグループ化するランタイム（ｒｕｎｔｉｍｅ）環境を提供する際に関係する段階を示す、図１のシステムの一実施方法に対する処理フロー図である。

【図７】異なるトランザクショングループに対して特化された競合管理を提供する際に関係する段階を示す、図１のシステムの一実施方法に対する処理フロー図である。

【図８】異なるトランザクショングループに対して特化されたロックおよびバージョン管理機構を提供する際に関係する段階を示す、図１のシステムの一実施方法に対する処理フロー図である。

【図９】デバッグまたは他の処理を拡張するための関連トランザクションのグループ化に命名する際に関係する段階を示す、図１のシステムの一実施方法に対する処理フロー図である。

【図１０】複数のトランザクショングループのブロック図である。

【発明を実施するための形態】

【０００７】

ここでの技術および技法は、トランザクションのメモリシステムとしての一般的文脈において記述される場合があるが、この技術および技法は、また、これらに加えて他の目的にも役立つものである。一実施方法において、ＭＩＣＲＯＳＯＦＴ（登録商標）．ＮＥＴ Ｆｒａｍｅｗｏｒｋ、または開発者がソフトウェアアプリケーションを開発するためのプラットホームを提供する他の如何なる種別のプログラムまたはサービスからなどの、フレームワークプログラム中での機能として、ここに記述される技法の内の１つまたは複数を実施することが可能である。別の実施方法においては、並行環境で実行する開発アプリケーションを取り扱う他のアプリケーションに関する機能として、ここに記述される技法の内の１つまたは複数が実施される。

【０００８】

一実施方法においては、トランザクションのメモリシステムの下で動作するプログラムにおいて使用するために、トランザクションのグループ化機能を提供する。トランザクションのグループ化機能は、トランザクションをグループのかたちに配置することを可能にする。１つのトランザクションの集合が、他の何れのトランザクションによりアクセスされたデータとも明らかに重複しないデータにアクセスする（例えば、データを読み／書く）と判定することが可能である場合に、この集合を１つの「トランザクショングループ」

であるとみなすことが可能である。

【０００９】

上の定義によると、１つのグループの部分であるトランザクションは、他のグループ中の他のトランザクションによりアクセスされた読み／書きデータと重複しない読み／書きデータに対して動作することが知られる。その結果、当該各グループに対して区別できるロックおよびバージョン管理機構を実施することが可能になり、各トランザクショングループが、当該グループにおけるトランザクションがアクセスしたデータに対して特に選択された最も適切なロックおよびバージョン管理のアルゴリズムを活用することを可能にする。

【００１０】

当該グループにおけるトランザクションがアクセスした特定のデータに加えて、他の多くの要因がグループの中で使用されるロックおよびバージョン管理のアルゴリズムの特定の選択に影響を及ぼす可能性がある。例えば、トランザクションの持続時間（ｄｕｒａｔｉｏｎ）、またはトランザクション中のコードの特性（ｎａｔｕｒｅ　ｏｆ　ｔｈｅ　ｃｏｄｅ）が、そのような他の要因のうちの２つである。一実施方法においては、通常は互換性のないロックおよびバージョン管理機構を、１つの処理中に並行的に使用することが可能であり、性能を増加させることが可能になる。

【００１１】

何時トランザクションをグループ化することが可能かの判定は、複数の手段により達成することが可能である。一実施方法では図４において記述されるように、グループを画定するためにプログラマによって提供される注釈（ａｎｎｏｔａｔｉｏｎ）を活用することができる。別の実施方法では図５において記述されるように、グループおよびグループ所属（ｇｒｏｕｐ　ｍｅｍｂｅｒｓｈｉｐ）を自動的に推測するためにコンパイラヒューリスティックスを使用することができる。さらに別の実施方法では図６において記述されるように、グループおよびグループ所属を動的かつ自動的に推測するためにランタイム環境を使用することができる。グループを生成し、そしてグループ所属を割り当てる際に関係する特定の機構は多々あり、かつ種々の方法で組み合わせが可能であることを理解されたい。

【００１２】

図１に示されるように、システムの１つまたは複数の部分を実施するために使用する代表的コンピューターシステムは、コンピューティングデバイス１００などのコンピューティングデバイスを含む。その最も基本的な構成において、コンピューティングデバイス１００は、通常、少なくとも１つの処理ユニット１０２およびメモリ１０４を含む。コンピューティングデバイスの正確な構成および種別によって、メモリ１０４は、揮発性（ＲＡＭなど）、不揮発性（ＲＯＭ、フラッシュメモリなど）、またはこの２つの何らかの組み合わせであることができる。この最も基本的な構成は、図１において破線１０６によって示される。

【００１３】

さらに、デバイス１００は、また、付加的な特徴／機能性を有することができる。例えば、デバイス１００はまた、限定的ではなく、磁気あるいは光ディスクまたはテープを含む（着脱可能および／または固定）追加的記憶装置を含むことができる。そのような追加的記憶装置は、図１において着脱可能記憶装置１０８および固定記憶装置１１０によって示される。コンピューター記憶装置媒体には、コンピューターで読み取り可能な命令、データ構造、プログラムモジュール、または他のデータなどの情報の格納のための任意の方法または技術で実施される、揮発性および不揮発性の、着脱可能および固定媒体が含まれる。メモリ１０４、着脱可能記憶装置１０８、および固定記憶装置１１０は、すべてコンピューター記憶装置媒体の例である。コンピューター記憶装置媒体には、限定的ではなく、ＲＡＭ、ＲＯＭ、ＥＥＰＲＯＭ、フラッシュメモリまたは他のメモリ技術、ＣＤ－ＲＯＭ、ＤＶＤ（デジタル多用途ディスク（Ｄｉｇｉｔａｌ　Ｖｅｒｓａｔｉｌｅ　Ｄｉｓｋ））または他の光記憶装置、磁気カセット、磁気テープ、磁気ディスク記憶装置または他

の磁気記憶装置デバイス、または必要な情報を格納するために使用可能であり、かつデバイス１００がアクセス可能な、他の如何なる媒体もが含まれる。そのようなコンピューター記憶装置媒体の何れもがデバイス１００の部分であることができる。

【００１４】

コンピューティングデバイス１００は、コンピューティングデバイス１００が他のコンピューター／アプリケーション１１５と通信することを可能にする１つまたは複数の通信接続１１４を含む。デバイス１００は、また、キーボード、マウス、ペン、音声入力デバイス、接触入力デバイスなどの入力デバイス（群）１１２を有することができる。デバイス１００は、表示、スピーカー、印刷器などの出力デバイス（群）１１１をもまた、含むことができる。これらのデバイスは、当技術分野においてよく知られており、そしてここで長々と議論する必要はない。一実施方法においては、コンピューティングデバイス１００は、トランザクションのメモリアプリケーション２００を含む。トランザクションのメモリアプリケーション２００は、図２においてさらに詳細に記述されることになる。

【００１５】

図１を継続的に参照しつつ、次に図２を見ると、コンピューティングデバイス１００上で動作するトランザクションのメモリアプリケーション２００が示されている。トランザクションのメモリアプリケーション２００は、コンピューティングデバイス１００に常駐するアプリケーションプログラムの内の１つである。しかしながら、１つまたは複数のコンピューター上でおよび／または図１上に示されるものとは別の変形における、コンピューターで実行可能な命令として、代替としてまたは追加的に、トランザクションのメモリアプリケーション２００を具現化可能であることが理解されるであろう。代替としてまたは追加的に、トランザクションのメモリアプリケーション２００の１つまたは複数の部分は、システムメモリ１０４の一部であり、他のコンピューターおよび／またはアプリケーション１１５上にあり、またはコンピューターソフトウェア技術において生起するような他の変形である可能性がある。

【００１６】

トランザクションのメモリアプリケーション２００は、プログラム論理２０４を含み、これが、ここに記述される技法のいくつかまたはすべてを実行することに関与する。プログラム論理２０４は、特定のプログラムにおける関連トランザクションを一緒にグループ化することを可能にする、（以下で図３～図６に関して記述されるような）トランザクションのグループ化機能を提供するための論理２０６、（以下で図７に関して記述されるような）トランザクショングループを使用して特化された競合管理を提供するための論理２１０、（以下で図８に関して記述されるような）異なるトランザクショングループに対して異なったロックおよびバージョン管理機構を提供するための論理２１２、（以下で図９に関して記述されるような）デバッグまたは他の処理を拡張するためにトランザクショングループに命名するための論理２１４、およびトランザクションのメモリアプリケーションを動作させるための他の論理２２０を含む。

【００１７】

図１～図２を継続的に参照しつつ次に図３～図１０を見ると、トランザクションのメモリアプリケーション２００の１つまたは複数の実施方法を実施するための段階がさらに詳細に記述される。いくつかの実施方法においては、図３～図１０の処理は、コンピューティングデバイス１００の演算論理において少なくとも部分的に実施されるものである。図３は、トランザクションのメモリアプリケーション２００に対する上位処理フロー図である。処理は開始点２４０で開始し、ソフトウェア、ハードウェア、および／またはその組み合わせを使用してトランザクションのメモリシステムを提供する（段階２４２）。システムは、特定のプログラムにおける関連トランザクションを、手動により（図４において記述されるように）、および／またはプログラム的に（図５および図６において記述されるように）、一緒にグループ化することを可能にするトランザクションのグループ化機能を提供する（段階２４４）。システムは、図７～図１０において記述されるように、プログラム性能を向上させもしくはプログラム動作を拡張するために、トランザクショングル

ープを使用する（段階２４６）。処理は、終了点２４８で終了する。

【００１８】

図４は、プログラマがトランザクションをグループ化することを可能にする際に関係する段階の一実施方法を示す。処理は、開始点２７０で開始し、トランザクションのメモリシステムの下で実行する特定のプログラムのソースコードにアクセスするための入力をプログラマから受け取る（段階２７２）。プログラマは、トランザクションに、宣言を加えもしくはグループを割り当てる（段階２７４）。システムは、図７～図９においてより詳細に記述されるように、プログラム動作を拡張するために特定されたグループを使用する（段階２７６）。処理は、終了点２７８で終了する。

【００１９】

図５は、トランザクションを自動的にグループ化する言語コンパイラを提供する際に関係する段階の一実施方法を示す。処理は、開始点２９０で開始し、トランザクションのメモリシステムの下で実行するプログラムをコンパイルするための言語コンパイラを提供する（段階２９２）。特定のプログラムのコンパイル時に、何かのトランザクションを一緒にグループ化できるかを判定するための論理が使用される（段階２９４）。いくつかの非限定的な実施例として、コンパイラによってプログラミング言語の特定のセマンティックスを活用することにより、またはグループが可能であることを示すグローバル分析を通して、トランザクションを一緒にグループ化することができる。システムは、プログラムにおいて識別されたグループを生成し（段階２９６）、そして、次に、特定された当該グループを使用して、プログラム動作を拡張する（段階２９８）。処理は、終了点３００で終了する。

【００２０】

図６は、トランザクションをグループ化するランタイム環境を提供する際に関係する段階の一実施方法を示す。処理は、開始点３１０で開始し、トランザクションのメモリシステムの下でプログラムを動かすためにランタイム環境を提供する（段階３１２）。特定のプログラムを動かすとき、一緒にグループ化されるべきであるトランザクションを識別するための論理が使用される（段階３１４）。いくつかの非限定的な実施例として、構造によって重複しない読み／書きデータに対して動作せねばならないトランザクションの集合を識別することにより、ランタイム環境によってトランザクションを一緒にグループ化することができる。ランタイムは、その分析処理能力を上げるために、コンパイラによって供給されるヒントに対して動作することができる。ランタイムは、識別されたトランザクションをグループ化し（段階３１６）、そして、グループ化された当該トランザクションを使用して特定のプログラムの動作を改良する（段階３１８）。処理は、終了点３２０で終了する。

【００２１】

次に、図７～図９では、グループ化されたトランザクションを使用して、如何にプログラム動作を拡張することが可能かを記述するための実施例が用いられている。図７は、グループ化されたトランザクションを使用して特化された競合管理を提供する際に関係する段階の一実施方法を示している。処理は、開始点３４０で開始し、１つまたは複数の特化された競合管理方針をプログラマが定義することを可能にする（段階３４２）。各方針に対してシステムは、トランザクション実行スケジューリング設定、トランザクション中止処理（ａｂｏｒｔ　ｈａｎｄｌｉｎｇ）設定、および／または他の設定を特定することを可能にする（段階３４４）。次に、以前に定義された方針を、進行中の各トランザクショングループに割り当てることが可能である（段階３４６）。システムは、当該方針を使用し、グループ化されたトランザクションに対する特化された競合管理を実施する（段階３４８）。

【００２２】

競合管理は、ランタイムシステムによって使用される機構であり、複数の並行実行トランザクションの間で競合が検出されたときはいつも、適切な挙動を選択する。競合があるなら競合管理は、競合しているトランザクションのどれを保存（ｐｒｅｓｅｒｖｅ）し、

そしてどれを中止（ａｂｏｒｔ）するかを決定する。それはさらに個々のトランザクションの実行スケジュールを、それらが完了するまで動作することが可能なように、如何に再計画するかを決定する。ここに使用されるような特化された競合管理は、所与のランタイムのデフォルトのヒューリスティックスとは区別できる競合管理ヒューリスティックスを適用する能力を参照する。

【００２３】

一実施方法においては、異なるグループに異なった方針を適用することにより、プログラムの拡張された性能を実現することが可能である。例えば、当該グループのトランザクションが含む動作種別に対して最良の性能を与えることになる特化された競合管理方針の１つの種別を、特定のトランザクショングループに割り当てることが可能である。別のトランザクショングループには、他のトランザクショングループが含む動作種別に対して最良の性能を得ることになる別の特化された競合管理方針を割り当てることが可能である。処理は、終了点３５０で終了する。

【００２４】

図８は、トランザクションの異なるグループに対して異なったロックおよびバージョン管理機構を提供する際に関係する段階の一実施方法を示している。処理は、開始点３７０で開始する。システムは、各トランザクショングループに対してロックおよびバージョン管理の何れの種別を使用するかを決定するための論理を使用し（段階３７２）、次に必要に応じて各トランザクショングループで使用されるロックおよびバージョン管理を調整する（段階３７４）。処理は、終了点３７６で終了する。

【００２５】

一実施例として、１つのトランザクションのメモリのロックおよびバージョン管理機構を１つの特定のトランザクショングループと共に使用し、一方可能性として互換性のない別のトランザクションのメモリのロックおよびバージョン管理機構を別のトランザクショングループと共に使用することが可能である。

【００２６】

複数のトランザクショングループがあるとき、異なるトランザクションのメモリ機構を如何にして一緒に組み合わせることが可能であるかをさらに示すための非限定的な一実施例を見てみよう。バージョン管理のために１つのトランザクショングループがバッファ更新方式（ｂｕｆｆｅｒｅｄ　ｕｐｄａｔｅ　ｓｃｈｅｍｅ）を使用でき、一方別のグループが復元ログ付きインプレース更新方式（ｉｎ－ｐｌａｃｅ　ｕｐｄａｔｅ　ｓｃｈｅｍｅ　ｗｉｔｈ　ｕｎｄｏ　ｌｏｇｇｉｎｇ）を使用できる。トランザクションをグループ化することによって、そのような互換性のないトランザクションのメモリのロックおよびバージョン管理機構の組み合わせを使用することを可能にする方法でデータを分離することが可能であり、総合的な性能の改良を可能にすることができる。別の実施例として、いくつかのトランザクショングループにおいて、高速であるが、制限されたハードウェア準拠のトランザクションのメモリ機構を使用することが可能であり、一方、他のトランザクショングループにおいて、当該グループのトランザクションに対してハードウェア制限が許容できないとき、互換性のない、かつ低速のソフトウェアトランザクションのメモリ機構を使用することが可能である。

【００２７】

上記の仮説的な実施例では、単に２つのトランザクショングループに関してこの技法を使用して言及したが、この概念は、２つより多いグループと共に、およびトランザクションのメモリのロックおよびバージョン管理機構の様々な組み合わせと共に、ハードウェアおよびソフトウェアの両方のアプローチを含んで使用することが可能である。

【００２８】

図９は、デバッグまたは他の処理を拡張するための関連トランザクションのグループ化に命名する際に関係する段階の一実施方法を示している。処理は、開始点４５０で開始し、関連トランザクションがユーザによっておよび／またはプログラム的に一緒にグループ化されることを可能にする（段階４５２）。ユーザによっておよび／またはプログラム的

に各トランザクショングループに名前を与えることを可能にする命名機能が提供される（段階４５４）。システムは、次に、当該グループ名をデバッグ、プロファイル（ｐｒｏｆｉｌｉｎｇ）、または他の処理を拡張するために、表示しまたは使用する（段階４５６）。例えば、グループ名を、デバッガー（ｄｅｂｕｇｇｅｒ）またはプロファイラー（ｐｒｏｆｉｌｅｒ）に表示可能であり、ユーザが、特定のトランザクショングループをより容易に識別することを可能にする。処理は、終了点４５８で終了する。

【００２９】

図１０は、複数のトランザクショングループのブロック図である。ここで示されている実施例において、第１のトランザクショングループ５００が４つのトランザクションを含み、その内の１つは他方の中に入れ子にされる。第２のトランザクショングループ５０２は、ただ単一のトランザクションを含む。異なる数のトランザクショングループ、および／または各グループ中に異なる数のトランザクションを含む、他の多数のトランザクションのグループ化のシナリオもまた、可能である。

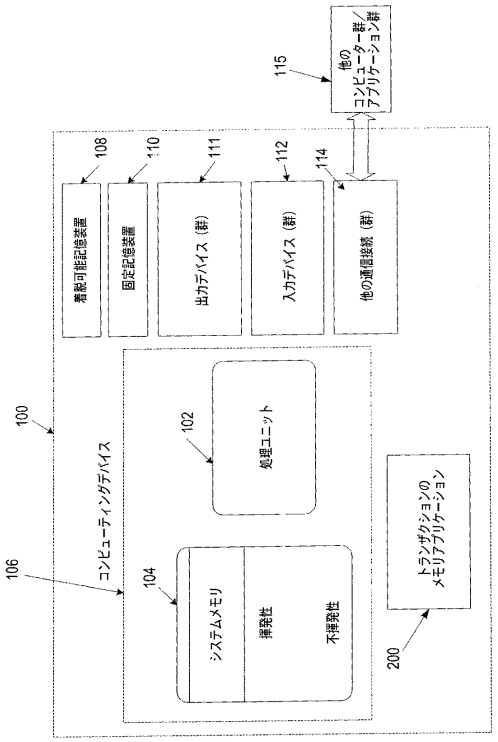
【００３０】

対象主題が、構造上の特徴および／または方法論的行為に特有な言語で記述されているが、添付された請求項において定義される対象主題が上で記述された特定の特徴または行為に必ずしも限定されるものではないことを理解されたい。上で記述された特定の特徴および行為はむしろ、当該請求項を実施する例としての形体として開示されるものである。ここに、および／または引き続く請求項により記述される実施方法の精神の中に含まれるすべての同等物、変更、および修正が保護されることが望まれる。

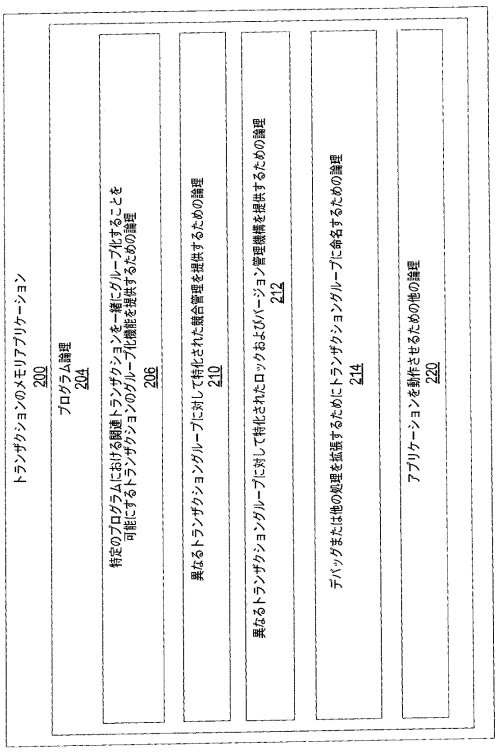
【００３１】

例えば、コンピューターソフトウェア技術分野における当業者は、１台または複数のコンピューター上で、ここにて議論された実施例が、実施例において描写されたものに対してより少ないかまたは追加的なオプションまたは特徴を含むように、異なって構成できるであろうことを認識する。

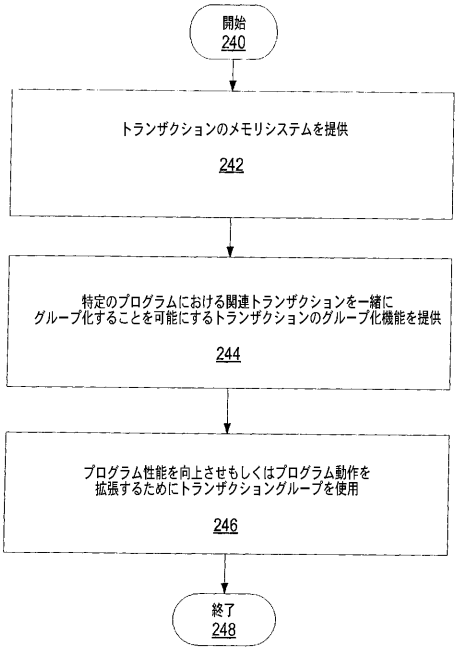
【図１】



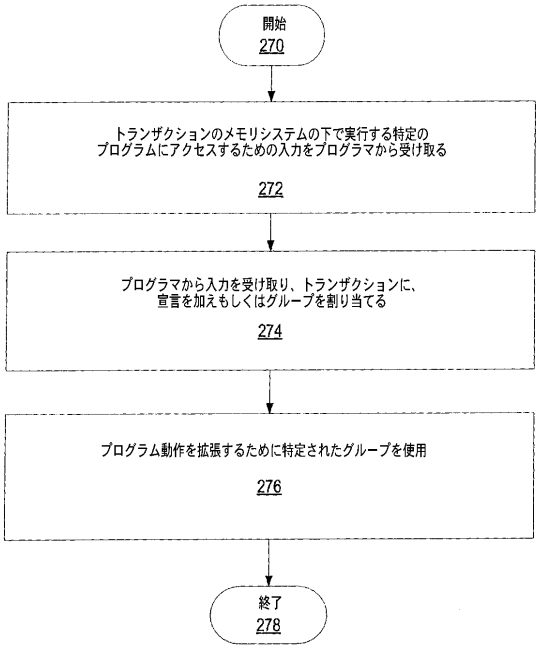
【図２】



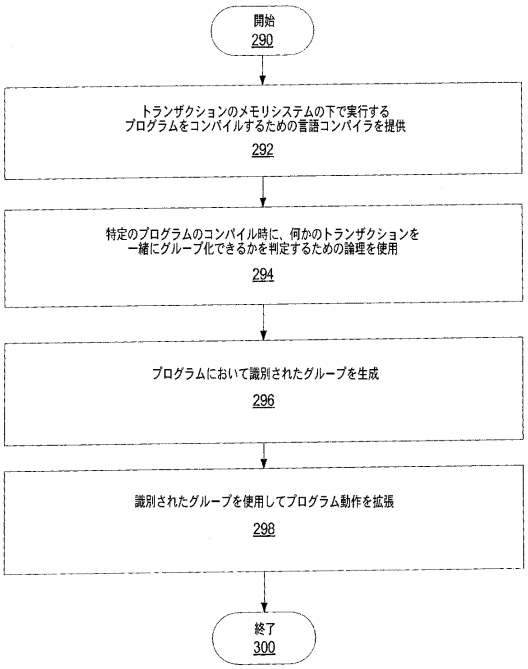
【図３】


開始
240
トランザクションのメモリシステムを提供
242
特定のプログラムにおける関連トランザクションを一緒に
グループ化することを可能にするトランザクションのグループ化機能を提供
244
プログラム性能を向上させもしくはプログラム動作を
拡張するためにトランザクショングループを使用
246
終了
248

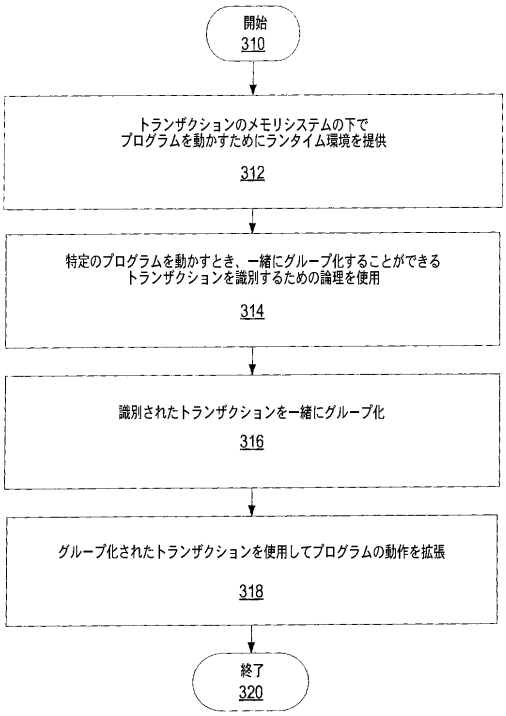
【図４】


開始
270
トランザクションのメモリシステムの下で実行する特定の
プログラムにアクセスするための入力をプログラマから受け取る
272
プログラマから入力を受け取り、トランザクションに、
宣言を加えもしくはグループを割り当てる
274
プログラム動作を拡張するために特定されたグループを使用
276
終了
278

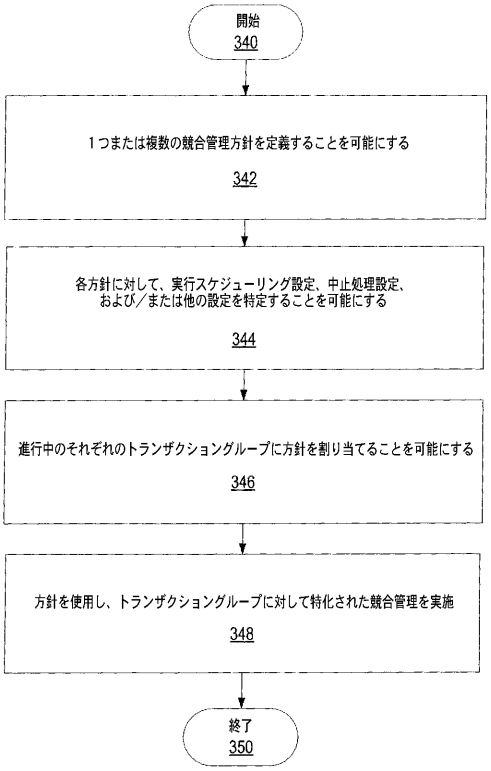
【図５】


開始
290
トランザクションのメモリシステムの下で実行する
プログラムをコンパイルするための言語コンパイラを提供
292
特定のプログラムのコンパイル時に、何かのトランザクションを
一緒にグループ化できるかを判定するための論理を使用
294
プログラムにおいて識別されたグループを生成
296
識別されたグループを使用してプログラム動作を拡張
298
終了
300

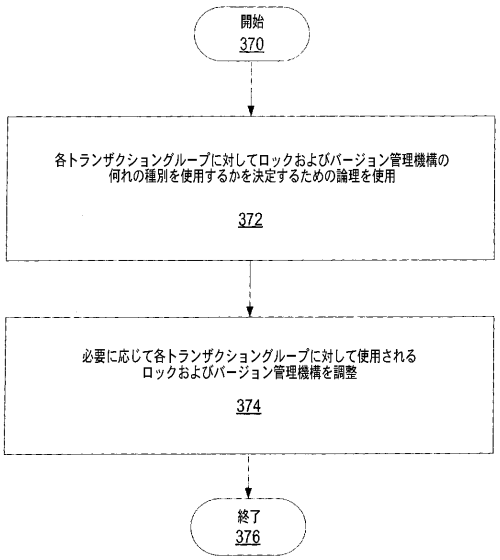
【図６】


開始
310
トランザクションのメモリシステムの下で
プログラムを動かすためにランタイム環境を提供
312
特定のプログラムを動かすとき、一緒にグループ化することができる
トランザクションを識別するための論理を使用
314
識別されたトランザクションを一緒にグループ化
316
グループ化されたトランザクションを使用してプログラムの動作を拡張
318
終了
320

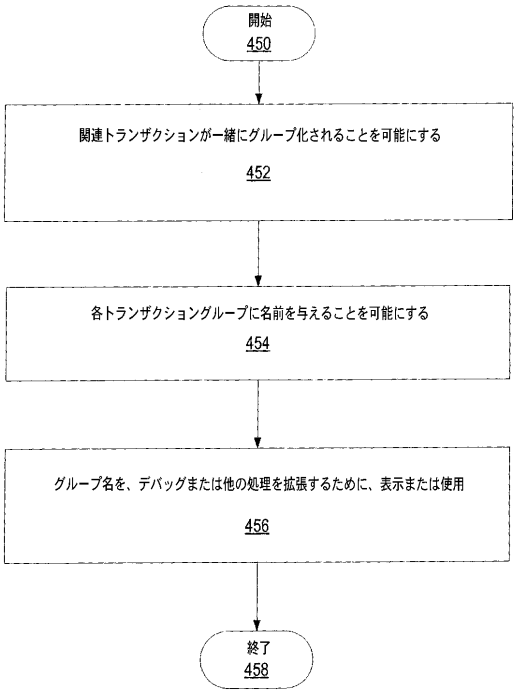
【図７】


開始
340
1つまたは複数の競合管理方針を定義することを可能にする
342
各方針に対して、実行スケジューリング設定、中止処理設定、
および／または他の設定を特定することを可能にする
344
進行中のそれぞれのトランザクショングループに方針を割り当てることを可能にする
346
方針を使用し、トランザクショングループに対して特化された競合管理を実施
348
終了
350

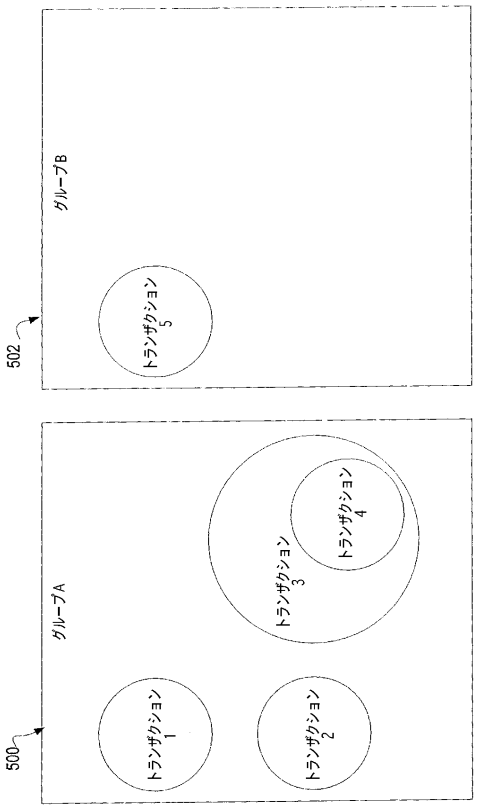
【図８】


開始
370
各トランザクショングループに対してロックおよびバージョン管理機構の
何れの種別を使用するかを決定するための論理を使用
372
必要に応じて各トランザクショングループに対して使用される
ロックおよびバージョン管理機構を調整
374
終了
376

【図９】


開始
450
関連トランザクションが一緒にグループ化されることを可能にする
452
各トランザクショングループに名前を与えることを可能にする
454
グループ名を、デバッグまたは他の処理を拡張するために、表示または使用
456
終了
458

【図１０】


グループB
トランザクション 5
502
グループA
トランザクション 3
トランザクション 4
トランザクション 1
トランザクション 2
500

フロントページの続き

(74)代理人 100120112
弁理士 中西 基晴
(74)代理人 100147991
弁理士 鳥居 健一
(74)代理人 100119781
弁理士 中村 彰吾
(74)代理人 100162846
弁理士 大牧 綾子
(74)代理人 100173565
弁理士 末松 亮太
(74)代理人 100138759
弁理士 大房 直樹
(74)代理人 100091063
弁理士 田中 英夫
(74)代理人 100077481
弁理士 谷 義一
(74)代理人 100088915
弁理士 阿部 和夫
(72)発明者 マーティン タイユファー
アメリカ合衆国 ９８０５２ ワシントン州 レッドモンド ワン マイクロソフト ウェイ マイクロソフト コーポレーション インターナショナル パテンツ内

審査官 田中 幸雄

(56)参考文献 特開平０１－１６６２３６（ＪＰ，Ａ）
特表２００３－５３０６４６（ＪＰ，Ａ）
Michel BANATRE et al., A fault tolerant tightly coupled multiprocessor architecture based on stable transactional memory, Rapports de Recherche, フランス, INRIA, ２００７年５月２４日, No. 1178, 全頁, ＵＲＬ, http://hal.inria.fr/inria-00075380/
Bratin SAHA et al., Architectual Support for Software Transactional memory, Proceedings of the 39th Annual IEEE/ACM International Symposium on Microarchitecture, 米国, IEEE, ２００６年１２月 １日, pages 185-196

(58)調査した分野(Int.Cl., ＤＢ名)
Ｇ０６Ｆ ９／５２
Ｇ０６Ｆ １２／００