(19)日本国特許庁(JP)　(12)特　許　公　報(B2)　(11)特許番号
特許第4049293号
(P4049293)
(45)発行日　平成20年2月20日(2008.2.20)　(24)登録日　平成19年12月7日(2007.12.7)

(51)Int.Cl.　F I
G06F 12/00 (2006.01)　G06F 12/00　518A

請求項の数 10　(全 19 頁)

(21)出願番号	特願2000-535992(P2000-535992)
(86)(22)出願日	平成11年3月11日(1999.3.11)
(65)公表番号	特表2002-507021(P2002-507021A)
(43)公表日	平成14年3月5日(2002.3.5)
(86)国際出願番号	PCT/US1999/005382
(87)国際公開番号	WO1999/046674
(87)国際公開日	平成11年9月16日(1999.9.16)
審査請求日	平成15年2月25日(2003.2.25)
(31)優先権主張番号	09/039,281
(32)優先日	平成10年3月12日(1998.3.12)
(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

最終頁に続く

(54)【発明の名称】情報システムにおいて一貫性を維持するためのトランザクションシステム及び方法

(57)【特許請求の範囲】
【請求項１】
非同期トランザクションをサポートする情報システムにおいて一貫性を維持するための、サーバマシン上で実行されるトランザクションシステムにおいて、該情報システムは、複数のユーザ間で分配されるデータベースを含み及びクライアントマシン上に実装され、該トランザクションは、トランザクションの実行時に起動され、該トランザクションが１つ以上のサブトランザクションステップを含み、前記トランザクションシステムが下記機能を実行するためのプログラムを実行し、該機能が、
サーバマシンとデータベース又はアプリケーションとの間のトランザクションを実行するために、ユーザ定義された独立性レベルを選択できるようにするためのグラフィカルユーザインタフェースと、
各サブトランザクションステップの実行に影響を及ぼす各オブジェクトのためのオブジェクト状態情報を格納するための持続的サービスと、
前記選択された独立性レベルに従い各サブトランザクションステップが実行される際に、該サブトランザクションステップによって影響を及ぼされたオブジェクトの状態が、該オブジェクトが該トランザクションにおいて見られた最後の状態と同じであるかを検証するための状態検証サービスであって、前記オブジェクトの状態が、該オブジェクトが前記トランザクションにおいて見られた前記最後の状態と同じでない場合に、該状態検証サービスが独立性故障を生じ、そして該トランザクションについてのフォワード進行処理を中止するように動作可能であり、前記オブジェクトの状態が、該オブジェクトが前記トラン

ザクションにおいて見られた前記最後の状態と同じである場合に、前記サブトランザクションステップの実行が進行する、前記状態検証サービスと、
を含む、前記トランザクションシステム。
【請求項２】
独立性レベルの選択が、一貫サービスから連続的サービスまでの範囲にわたって変化し、一貫サービスがサブトランザクションレベルで全てのＡＣＩＤ特性に合い、連続的サービスがサブトランザクションの実行における何らかのＡＣＩＤ特性に保証しないことを提供する、請求項１記載のトランザクションシステム。
【請求項３】
一貫サービスが、各サブトランザクションが実行される期間にわたって読取りの一貫性を保証するための”独立性レベル読み取り”を含む、請求項２記載のトランザクションシステム。
【請求項４】
独立性レベルの選択が、”ストリンジェント”独立性サービス及び”独立性サービスなし”を含み、
該”ストリンジェント”独立性サービスが、サブトランザクションレベルでの全てのＡＣＩＤ特性に対するコンプライアンスを提供し、
該”独立性サービスなし”がサブトランザクションの実行において何らかのＡＣＩＤ特性の保証を提供しない、請求項１記載のトランザクションシステム。
【請求項５】
独立性レベルの選択が”最良のエフォート”独立性サービスを更に含み、該”最良のエフォート”独立性サービスでは、該”最良のエフォート”独立性サービス及び前記”ストリンジェント”独立性サービスが全てのＡＣＩＤ特性にコンプライアンスを提供するが、前記”ストリンジェント”独立性サービスは、サブトランザクションステップが実行される全時間にわたって一貫性を保証するための”独立性レベル読み取り”を含む、請求項４記載のトランザクションシステム。
【請求項６】
独立性レベルの選択が”最小のエフォート”独立性サービスを更に含み、該”最小のエフォート”独立性サービスでは、前記状態検証サービスが使用できず、オブジェクト状態チェックが、サブトランザクションステップの実行の前に実行されない、請求項５記載のトランザクションシステム。
【請求項７】
独立性レベルの選択が”エフォートなし”独立性サービスを更に含み、該”エフォートなし”独立性サービスでは、一貫性エラーがサブトランザクション処理中に戻されるまで、トランザクションの実行をもたらす”エフォートなし”独立性レベル選択を含む、請求項４記載のトランザクションシステム。
【請求項８】
サブトランザクションレコードのリンクリストを維持し、且つトランザクションが実行されたときに、トランザクションレコードを補償するためのサガサービスと、
トランザクションの実行のフォワードでの独立性故障が発生すると、補償トランザクションを実行するためのリカバリーサービスと、
を更に含む、請求項１のトランザクションシステム。
【請求項９】
非同期トランザクションをサポートする情報システムにおいて一貫性を維持するためにサーバマシン上で実行される方法であって、該情報システムは、複数のユーザ間で分配されるデータベースを含み及びクライアントマシン上に実装され、該トランザクションは、該トランザクションは、トランザクションの実行時に起動され、該トランザクションが１つ以上のサブトランザクションステップを含み、前記方法が、
サーバマシンとデータベース又はアプリケーションとの間のトランザクションを実行するために、所定のサービスレベルの選択するためのグラフィカルユーザインタフェースを

提供するステップと、
各サブトランザクションステップの実行に影響を及ぼす各オブジェクトのためのオブジェクト状態情報を格納することと、
前記選択された独立性レベルに従い各サブトランザクションステップが実行される際に、該サブトランザクションステップによって影響を及ぼされたオブジェクトの状態が、該オブジェクトが該トランザクションにおいて見られた最後の状態と同じであるかを検証し、そして前記オブジェクトの状態が、該オブジェクトが前記トランザクションにおいて見られた前記最後の状態と同じでない場合に、該トランザクションについてのフォワード進行処理を中止し、前記オブジェクトの状態が、該オブジェクトがトランザクションにおいて見られた前記最後の状態と同じである場合に、前記サブトランザクションステップの実行が進行することと、
を含む方法。
【請求項１０】
インターチェンジサーバシステムのサービスモジュールにおいて使用するための請求項１に記載のトランザクションシステムであって、該インターチェンジサーバシステムが、アプリケーションコラボレーションモジュール及びサービスモジュールを含み、前記アプリケーションコラボレーションモジュールは２つ以上のアプリケーションの間でインターオペラビリティを定義し、且つ関連アプリケーションに搬送するための一以上のコネクターにデータを送出する１つ以上のサブトランザクションステップを有するトランザクションを含む、前記トランザクションシステム。
【発明の詳細な説明】
【０００１】
【発明の属する技術分野】
本発明は、全体として、コンピューティングシステムに関し、特に情報システムの非同期トランザクションを処理する方法及び装置に関する。
【０００２】
【従来の技術】
情報システム又はデータベースシステムは、一まとまりのコンピュータ化データファイル（データベース）のためのリポジトリである。一人以上のユーザがデータベースの情報にアクセスし、その情報を検索、修正できるように、情報システムは、データベース管理システムの形でデータベース管理機能を含むことができる。データベースは、ある企業のアプリケーションシステムによって使用することができる一まとまりの持続的なデータから成る。一般的に、このような多くのユーザは、所定の時間にデータベースにアクセスする。データベース管理システムの１つの機能は、データベースのデータに示される多くの脅威に対する防御である。データベースデータを保護するためのツールは、リカバリーツール、コンカレンシツール、セキュリティツール及び完全性ツールを含むことができる。
【０００３】
リカバリー及びコンカレンシは、トランザクション処理を伴う関連した概念である。トランザクションは、作業の論理単位である。トランザクションは、データベースに対する１つ以上のアクセスを伴うことができ、また、データベースに格納される情報の１つ以上のアップデートを含むことができる。たとえ多くのデータベース動作を起動することができる場合であっても、トランザクションは原子単位であると考えられる。古典的なトランザクション処理では、トランザクションがデータベースに幾つかのアップデートを実行し、トランザクションがその正常終了に至る前に障害が発生すると、それらのアップデートは取り消されることになる。
【０００４】
トランザクション処理を容易にするために、データベース管理システムはトランザクションマネージャを含むことができる。トランザクションマネージャは、データベースに対するトランザクションを監督して、トランザクション処理をサポートするために必要な原子性を提供するためにＣＯＭＭＩＴ及びＲＯＬＬＢＡＣＫオペレーションなどのツールを

使用することができる。ＣＯＭＭＩＴオペレーションは、作業のロジカルユニット、すなわちトランザクションが完了したという”トランザクションの成功終了（成功エンド・オブ・トランザクション）”を送信する。ＲＯＬＬＢＡＣＫオペレーションは、データベースが矛盾状態にあるかもしれないことを示す”トランザクションの不成功終了（不成功エンド・オブ・トランザクション）”を送信する。リカバリーは、ロールバック（ROLLBACK）に応答してトランザクションマネージャによって開始されるそれらのオペレーションに関連する。

【０００５】

コンカレンシは、複数のユーザによって実行されるトランザクションのインタラクションに関連する。コンカレンシ管理者又は他のコンカレンシ制御メカニズムは、同時トランザクションがお互いのオペレーションを妨げないようにデータベース管理機能の一部として提供することができる。コンカレンシ制御メカニズムは、失ったアップデート、未遂の従属性及び矛盾する解析問題に関する問題を軽減することができる。コンカレンシ問題のより詳細な説明は、Addison Wesleyから出版されている「An Introduction toDatabase Systems 第5版」 (C.J. Date著、1991年)に記載されている。

【０００６】

コンカレンシ問題に対する１つの従来の解決法は、ロックの使用である。ロッキングは１つのトランザクションがデータベースの特定の部分の状態に依存することができ、データの状態を変更する他のトランザクションに関係しないという保証を提供する。

【０００７】

従来のコンカレンシ管理者は、２種類のロック、すなわち排他ロック（Ｘロック）及び共有ロック（Ｓロック））を使用する。１つのトランザクションだけが、トランザクションレコード上のＸロックを一度に保持することができる。２つ以上のトランザクションが、同じレコード上のＳロックを維持することができる。例えば、トランザクションＡがレコードＲ上の排他的（Ｘ）ロックを保持する場合、トランザクションＢから出されるＲ上のいずれのタイプのロックの要求もＢを待ち状態に入らせる。Ａのロックが解除されるまで、Ｂは待機することになる。トランザクションＡがレコードＲ上の共有（Ｓ）ロックを保持する場合、トランザクションＢから出されるＲ上のＸロックの要求はＢを待ち状態（に入らせる（そして、Ａのロックが解放されるまでＢは待機する）。しかしながら、トランザクションＢから出されるＲ上のＳロックの要求は認可される（すなわち、これ以降、ＢもＲ上のＳロックを保持することになる）。

【０００８】

レコードロックのトランザクション要求は、通常、暗黙的である。トランザクションは、レコードの検索に成功すると、自動的にそのレコード上のＳロックを取得する。トランザクションは、レコードのアップデートに成功すると、自動的にそのレコード上のＸロックを取得する。トランザクションがレコード上のＳロックをすでに保持している場合、アップデートはＳロックをＸレベルに「進める」。Ｘロックは、次の同期点(synchpoint)まで保たれる。同期点は、２つの連続するトランザクション間の境界を表す。このため、同期点は、作業のロジカルユニットの終りに一致し、従ってデータベースが一貫性の状態にある点（或いは一貫性の状態にあるはずの点）に一致する。通常、同期点を確立する唯一の操作は、ＣＯＭＭＩＴ、ＲＯＬＬＢＡＣＫ及びプログラム開始である。Ｓロックも、通常、次の同期点まで保持される。

【０００９】

トランザクションサービスの典型的役割は、トランザクションの古典的ＡＣＩＤ特性を確保にすることである。これらの特性には、原子性(Atomicity)、一貫性(Consistency)、独立性（又は隔離性）(Isolation)及び耐久性(Durability)が含まれる。

【００１０】

原子性は、トランザクションの全てのステップが実行されるか、或いはいずれのステップも実行されないトランザクションサービスの特性に関連する。一貫性は、１つの既知の状態から別の既知の状態へのデータの遷移をいう。独立性特性は、その言葉が連想させる

ように、あるトランザクションの範囲内で行われたデータ修正が他のトランザクションデータ修正から分離されていることを提示する。換言すれば、トランザクションの中間結果が、他のトランザクションに「示され」ない。耐久性特性は、トランザクションの範囲内で行われたデータ修正がディスクに保存される保証を確保する。

【００１１】

要するに、持続的なデータに対する状態変更が、他のプロセスによって行われる状態変更から分離され、「回復」できるような方法（すなわち、障害時に我々がオブジェクトを既知の状態に復元できる）で確実に行われるようにすることは、トランザクションサービスのジョブである。ロッキングはコンカレンシの保証を与えるが、長寿命又は非同期のトランザクションを処理する際に問題が起こる。上記のように、ロックは、トランザクションの完了及び次の同期点の発生まで所定の場所に残る。しかし、長寿命のトランザクションが生じると、データベースに対するロックは、効率のわるいパフォーマンスを生じさせることがある。例えば、ビジネストランザクションは、様々な供給元に送られる予定の見積額の要求を含むことがある。このトランザクションは、１つ以上の入札が返される将来の不確定な時点に完了する。明らかなように、これらのタイプの長寿命トランザクションを待つ間、データベースのリソースを拘束することは非効率的である。

【００１２】

古典的なオンライントランザクション処理（ＯＬＴＰ）システムでは、少量のデータに接触し、非常に素早く完了（例えば、古典的な航空券予約システム）する小さくて持続時間の短いトランザクションから作業負荷が構成されている。長寿命のトランザクションを示すシステムでは、ＷＡＮにわたって存続するデータに接触するトランザクション、又は信頼できるメッセージ送信プロバイダを介してデータを参照するトランザクション（すなわち、アプリケーションを実行する必要がない場合）から作業負荷が構成されることがある。。いずれにせよ、この作業負荷のセマンティックスは、古典的ＯＬＴＰ作業負荷とは劇的に異なっている。これらの作業負荷に対しては、トランザクション処理システムによって提供される従来のロッキング方式は、ひどく不適当である。

【００１３】

【課題を解決するための手段】

一般に、ある態様において本発明は、非同期トランザクションをサポートする情報システムの一貫性を維持するためにサーバマシン上で実装されるトランザクションシステムを提供する。この情報システムは、複数のユーザ間で共有され、クライアントマシン上に実装されるデータベースを含んでいる。このトランザクションシステムは、トランザクションの実行時に起動される。ここで、トランザクションは、１つ以上のサブトランザクションステップを含む。トランザクションシステムは、サーバマシンとデータベース又はアプリケーションとの間のトランザクションを実行するためのユーザ定義独立性レベル選択を受信するグラフィカルユーザインタフェースと、各サブトランザクションステップの実行において影響を受ける各オブジェクトに関するオブジェクト状態情報を記憶する持続性サービスと、サブトランザクションステップによって影響を受ける各オブジェクトの状態を検証し、そのオブジェクトの状態がそのオブジェクトがトランザクション中で最後に見られたときと同じであることを検証する状態検証サービスと、を含んでいる。

【００１４】

本発明の態様は、多数の特徴を含む。独立性レベル選択は、一貫性サービスから連続サービスまでの範囲にわたって変化することができる。ここで、一貫性サービスは、サブトランザクションレベルで全てのＡＣＩＤ特性を満たす。また、連続サービスは、サブトランザクションの実行時におけるＡＣＩＤ特性の保証を何ら提供しない。一貫性サービスは、各サブトランザクションが実行される期間にわたる読取りの一貫性を保証するために、”独立性レベル読み取り”を含んでいる。

【００１５】

独立性レベル選択は、”ストリンジェント”独立性サービス及び”独立性サービスなし”独立性サービスを含む。”ストリンジェント”独立性サービスは、サブトランザクショ

ンレベルで、全てのＡＣＩＤ特性にコンプライアンスを提供し、”独立性サービスなし”独立性サービスは、サブトランザクションの実行時におけるＡＣＩＤ特性の保証を何ら提供しない。

【００１６】

トランザクションシステムは、”最良のエフォート”独立性サービスを含む。ここで、”最良のエフォート”独立性サービス及びストリンジェント独立性サービスは、すべてのＡＣＩＤ特性にコンプライアンスを提供するが、”ストリンジェント”独立性サービスは、サブトランザクションステップが実行される期間全体にわたって一貫性を保証するために”独立性レベル読み取り”を含んでいる。トランザクションシステムは、”最小のエフォート”独立性サービスを含む。個の場合、状態検証サービスが使用禁止にされ、サブトランザクションステップの実行前にオブジェクト状態チェックは実行されない。トランザクションシステムは、”エフォートなし”独立性レベル選択を含む。この結果、一貫性エラーがサブトランザクション処理中に返されるまでトランザクションが実行されることになる。

【００１７】

トランザクションシステムは、サブトランザクションレコードのリンクリストを維持し、トランザクションが実行されるときにトランザクションレコードを補償するサガサービスと、トランザクションの実行のフォワード進行中に障害が発生したときにトランザクションの補償を実施するリカバリーサービスと、を含む。

【００１８】

別の態様では、本発明は、情報システムの一貫性を維持するためにサーバマシン上で実施される方法を提供する。ここで、この情報システムは、複数のユーザ間で共有され、クライアントマシン上で実装されるデータベースを含む。この方法は、トランザクションの実行時に呼び出される。ここで、トランザクションは、１つ以上のサブトランザクションステップを含む。この方法は、サーバマシン及びデータベースアプリケーション間のトランザクションを実行するための定義済みサービスレベル選択を行うステップと、各サブトランザクションステップの実行時に影響を受ける各オブジェクトに関するオブジェクト状態情報を記憶するステップと、サブトランザクションステップによって影響を受ける各オブジェクトの状態を検証し、そのオブジェクトの状態がそのオブジェクトがトランザクション中で最後に見られたときと同じであることを検証するステップと、を含む。その状態が最後の既知の状態と一致しない場合、トランザクションの順方向実行進行が停止される。

【００１９】

別の態様では、本発明は、独立のアプリケーションのインタラクションを指示するためのビジネス論理を含むコラボレーションであって、サーバマシン上で実行されるコラボレーションであり、複数の同様のアプリケーションと通じる複数のコネクターと、アプリケーションコラボレーションモジュール及びサービスモジュールを含むインターチェンジサーバを含む。前記アプリケーションコラボレーションモジュールは２つ以上のアプリケーションの間でインターオペラビリティを定義し、関連アプリケーションに搬送するための一以上のコネクターにデータを送出する１つ以上のサブトランザクションステップを有するトランザクションを含む。インターチェンジサーバは、アプリケーションによって管理されるデータベースで一貫性を維持するトランザクションシステムを含んでいるサービスモジュールを含む。トランザクションシステムは、トランザクション実行中に維持される独立性レベルを特徴づける定義されたユーザ独立性レベル選択を構成するグラフィカルユーザインタフェースを含む。各サブトランザクションステップの実行で影響を及ぼされる各オブジェクトに対して、オブジェクト状態情報を格納する持続的サービスを含み、オブジェクトがトランザクションで前に見られたのと同じ状態かを検証するために、サブトランザクションステップによって影響を及ぼされる各オブジェクトの状態を検証するための状態検証サービスを含む。

【００２０】

本発明の各態様は、多数の利点を含む。本発明の１つの利点は、必要とされている適切なレベルのサービスをサービスの顧客が要求できるように、柔軟なトランザクションサービスが提供されるということである。提供されるサービスの様々なレベルの各々について、このトランザクションサービスは、適切なトランザクショナルセマンティックス及びリカバリーセマンティックスを確保する。このトランザクションサービスは、構築が容易であり、適切なインタフェースを他の交換顧客に提供し、また拡張性を与える。他の利点及び特徴は、以下の説明及び特許請求の範囲から明らかになる。

【００２１】

【発明の実施の形態】

本発明の好ましい実施例は、１９９７年１月８日に本願の出願人によって出願された米国特許出願第０８／７８０５９３号に記載されたコラボレーション構成のコンテクスト中に動作する。好ましい実施例は、上記の出願で定義されるコラボレーション内のフレキシブルトランザクションシステムを提供するために用いられる。

【００２２】

一般に、コラボレーション構成は、相容しない且つ自分で直接互いにインタラクションしない独立のアプリケーションのインタラクションをサポートする。コラボレーション構成の構造の例は、図１の中で示される。コラボレーション構成は、独立アプリケーション１０２、１０４、１０６間のインタラクションのためのインターチェンジサーバ１００を提供する。インターチェンジサーバ１００は好ましく、コネクター１０８、１１０、１１２によってインターチェンジサーバ１００に接続しているアプリケーション１０２、１０４、１０６の間で動作する。そして、コネクター１０８、１１０、１１２は、インターチェンジサーバ１００内に位置するコラボレーション１１４、１１６に関連付けられる。コラボレーション１１４、１１６は、１つ以上のプロセスを表す。各プロセスは、１つ以上のアプリケーション１０２、１０４、１０６を含むトランザクションである。よって、コラボレーション１１４、１１６は、アプリケーション１０２、１０４、１０６間のインタラクション及びデータ交換のための共通ミーティングポイントとして作用する。

【００２３】

例えば、コラボレーション１１４は、アプリケーション１０２からデータを受け取り、そして異なるフォーマットでアプリケーション１０４にそのデータの一部を送ることを要求するプロセスを含んでもよい。アプリケーション１０２は、データをコネクター１０８へ送信する。コネクター１０８は、データをインターチェンジフォーマットオブジェクトに変換して、該データの利用性を示すイベントを公表する。コラボレーション１１４は、上記オブジェクトを、イベントの加入者として受け取る。コラボレーション１１４は、そのプロセスを実行し、コネクター１１０に送られるべきオブジェクトを生成する。コネクター１１０は、上記対象を適切なフォーマットに変換して、アプリケーション１０４において適切なマシン能を始める。このように、独立に相容しない２つのアプリケーション１０２、１０４からのインタラクションを必要とするプロセスが達成される。

【００２４】

本発明によるプロセス非同期トランザクションのためのより一般的な構成は、図２で示される。分散されたコンピュータシステム２００は、サーバマシン２２０とクライアントマシン２５０を含む。サーバマシン２２０とクライアントマシン２５０の各々は、セントラルプロセッシングユニット（ＣＰＵ）２０２、メモリ２０４、ディスクサブシステム２０６、ネットワークサブシステム２０８、オペレーティングシステム２１０とコミュニケーションインタフェース２１３を含んでもよい。各々は、アプリケーションプログラムの実行のために必要なサービスを提供する。メモリ２０４は、リードオンリーメモリ（ＲＯＭ）や揮発性及び不揮発性のランダムアクセスメモリ（ＲＡＭ）を含んでもよい。サーバマシン２２０内のディスクサブシステム２０６が、実行可能なプログラム（例えばビジネスアプリケーション（以下、ビジネス論理という場合がある）又はコラボレーション）の格納に用いられ、クライアントマシン２５０内のディスクサブシステム２０６が、アプリケーション２５１とデータベース２５２の格納に用いられても良い。ネットワークサブシ

ステム２０８は、他のサーバマシン２２０とクライアントマシン２５０を実行しているアプリケーションによって、コミュニケーションインタフェース２１３を介するコミュニケーションを容易にする。種々のクライアント及びサーバマシンが、ネットワーク２６によって接続される。ネットワークは、イントラネット、インタネット、ローカルエリアネットワーク又は他の類似のデバイスの形式であってもよい。

【００２５】

サーバマシン２２０は、アプリケーションＡＰＩに対して或る所定のリクエストを実行する責任の或るビジネス論理２２２を組み入れて良い。この例の目的のために、サーバマシン２２０は、クライアントマシン２５０を実行しているアプリケーション２５１と通信するために、（クライアントマシン２５０内の）ＡＰＩ２４０に、リクエストを提供する。サーバマシン２２０は、トランザクションシステム２２４と、メターデータ管理サービス２２１とユーザインタフェース２０７とを含む。トランザクションシステム２２４は、非同期或いは長寿命トランザクションのためにサポートを提供して、独立性レベル（サービスレベル）を設定する方法を含む。トランザクションシステム２２４は、クライアントが必要なサービスの適切なレベルをリクエストできるようにする。提供されるサービスの各異なるレベルで、トランザクションシステム２２４は、適切なトランザクション及びリカバリーセマンティクスを確保する。トランザクションシステム２２４は、状態検証サービス２２５、持続的サービス２２６、サガサービス２２８とリカバリーサービス２３０を含む。

【００２６】

状態検証サービス２２５は、トランザクションによって示されるオブジェクトと関連する状態情報をチェックする。特に、状態検証サービスは、トランザクションのために実行時で起動される。各サブトランザクションステップが実行される際、状態検証サービス２２５は、一定のトランザクションステップに影響される各オブジェクトの状態をチェックして、オブジェクトの現在状態が、トランザクション内に同様なオブジェクトが見られた前回と同じものであることを検証する。

【００２７】

持続的サービス２２６は、情報の持続状態を確保すること役割を果たす。トランザクションシステムの全ての他のコンポーネントは、持続的サービスの正確なオペレーションに頼る。アンドゥ操作を実行するために必要な全てのオブジェクト状態情報は、実行時で格納されなければならない。オブジェクト状態情報をロギングし、ビジネス論理２２２に従って実行されるトランザクションと関連するトランザクション情報を補償するためのリポジトリ２３７が、持続的サービス２２６に関連付けられる。

【００２８】

サガサービス２２８は、「トランザクションを開始する」、「トランザクションを行う」及び「トランザクションを中断する」のセマンティクスを制御することを含むトランザクション正当性を確保する役割を果たす。サガサービス２２８は、異常終了（トランザクション異常終了）の場合におけるトランザクションをアンドゥするための付随の補償トランザクションと共に行われる、サーバマシン２２０とクライアントマシン２５０とをそれぞれ実行するアプリケーション間のインタラクションのためのトランザクションを定義する（以下に定義される）「サガ」を形成する。

【００２９】

リカバリーサービス２３０は、異常終了後のトランザクション正当性を確保する役割を果たす。更に、リカバリーサービス２３０は、どんなタイプの異常終了が回復可能であることを定義し、回復のための方法論（すなわち、ロールバック対ロールフォーワード）を定義する役割を果たす。

【００３０】

これらのサービスの各々のオペレーションが、以下に、より詳細に論議される。

【００３１】

メタデータ管理サービス２２１は、ユーザインタフェース２０７のグラフィカルユーザ

インタフェース（ＧＵＩ）２１２上で、サービスリクエストを表示する方法を含む。サービスリクエストは、順番にリポジトリ２３７に格納されうるビジネス論理２２２を処理する際に用いられるための独立性レベル選択を定義することをユーザに促す。ビジネス論理２２２の処理及びユーザによって定義された独立性レベルの使用が、以下に、より詳細に論議される。

【００３２】

クライアントマシン２５０は、アプリケーション２５１を実行し、サーバマシン２２０への或いはサーバマシン２２０からのデータ搬送を必要とするトランザクションに応答して開始してもよい。API ２４０は、クライアントマシン２５０で実行するアプリケーション２５１に、データ検索、データインサート、データアップデート又はデータ削除のためのリクエストを伝える方法を提供する。これらの議論の目的のために、例えばデータベース２５２に位置するクライアントの中のデータレジデントは、サーバマシン２２０によって開始されたトランザクションによって操作されることになっている。API ２４０は、何らかの外部トランザクションコーディネータとのインタラクションのための何らかのインタフェースを提供しない。このように、外部の使用のために「トランザクション的」動作の何らかの概念を表さない。

【００３３】

ここで記載されているトランザクションシステムによってアドレスされる基本問題は、サーバ上でビジネス論理に代わって（トランザクションのインタフェースを示さない）一般的なＡＰＩとインタレクトすることになる。トランザクション作業は、ＡＣＩＤ特性を示すオペレーションのシーケンスとして定義される。

【００３４】

アプリケーションＡＰＩとインタラクトするトランザクションシステムは、（それがＡＰＩを通してさらされないので）排他的ロッキングを用いることができない、よって、単にＡＰＩによって提供されるマシン能性だけによって、ＡＣＩＤ特性を成し遂げなければならない。

【００３５】

本発明において、ビジネス論理２２２は、ユーザが、”ストリンジェント”、”最良のエフォート”、”最小のエフォート”、”エフォートなし”を含む４つの別々のトランザクション独立性レベルのうちの１つを指定できるようにするためのＧＵＩを示す。特定の独立性レベルの選択は、失敗したサブトランザクションステップに応答するための方法論及び状態情報の格納に対する必要な条件を定義する。「ストリンジェント」から下へ、各以降のレベルは、ユーザに、トランザクションＡＣＩＤ特性とシステム性能との保証の間で、トレードオフを提供する。しかし、より少ない制限レベルのサービス（”最小のエフォート”及び”エフォートなし”）は、ＡＣＩＤ特性に対して、少しのみの保証を提供し、或いは保証しないが、ますます少ないシステムオーバヘッドが、これらのトランザクションを実行するためにトランザクションシステムによって必要され、よって、より大きなレベルのシステムトランザクション能力を提供する。サービスの各レベルのオペレーションを理解するために、サンプルトランザクションが定義される。

【００３６】

図３ａに示されるように、ビジネス論理２２２（図２）は、１つ以上のトランザクション３００を含んでもよい。トランザクション３００は、１つ以上のサブトランザクション３０２を含んでもよい。各サブトランザクションは、サブトランザクションによって影響を受けることになっている全てのビジネスオブジェクトを識別するためのオブジェクト識別子３０４と、サブトランザクションの一部として行われる動作を記述する動詞（バーブ）３０６と、動詞と共に用いられる値（バリュー）３０８とを含む。

【００３７】

トランザクション３００は、それに関連付けられている補償サガチェイン３１０を有し、補償サガチェイン３１０は、トランザクション３００の異常終了の場合に実行されてもよい。サガチェイン３１０は、１つ以上の補償トランザクション３１２を含んでもよい。

サブトランザクションと補償トランザクションとの１対１のマッピングが存在してもよい。各補償トランザクション３１２は、補償トランザクションによって影響される全てのビジネスオブジェクトを識別するオブジェクト識別子３１４と、補償トランザクションの一部として実行される動作を記述する補償動詞（補償バーブ）３１６と、補償動詞と共に用いられる値３１８（又はビジネスオブジェクト）とを含む。

【００３８】

補償トランザクションのための内容作成は、ビジネス論理の開発者に任せられる。どんな特定のステップが補償トランザクションを形成するかは、その開発者によって定義される。トランザクションの実行のための時間（ランタイム）に、トランザクションシステム２２４（特にサガサービス２２８）は、ビジネス論理によって提供されるトランザクション（及びサブトランザクション）及び何らかの関連する補償トランザクションを定義する「サガ」を開発する。各サブトランザクションステップの影響をセマンティクス的に取り消す論理を含む１つ且つ唯一の原子性補償トランザクションスが存在するように、サガは、補正された補償トランザクションステップテップとサブトランザクションとからなる１つのセットである。トランザクションの論理的「アンドゥ」が、失敗状態を検出すると実行されるように、サガは、トランザクション及び関連する補償トランザクションステップに関連付けられた必要な情報を記録するキューを形成する。サガサービス２２８は、サガの作成及び修正のために用いられる。

【００３９】

実行（ランタイム）の時に、トランザクション３００は、データベース２５２に格納されているレコード（記録）（ビジネスオブジェクト）の検索、アップデート又は他操作を結果としてもたらす。各サブトランザクションステップで、ビジネスオブジェクトは、サブトランザクション動詞に従って操作されてもよい。サブトランザクションが完了する時点で、実行されるサブトランザクションのリスティング、並びにあらゆるオブジェクトの状態及びトランザクションサービスのためにリクエストされる独立性レベルを維持するために、持続的サービスが起動されうる。

【００４０】

基礎的なトランザクション構成が与えられる場合、独立性の各レベルのための持続的サービス、サガサービス及びリカバリーサービスを含むトランザクションのオペレーションが記載されることが可能である。

【００４１】

ストリンジェント独立性レベル

各サブトランザクションステップ（又は補償トランザクション）が実行される前に定められるストリンジェントエントリ基準によって、ストリンジェント独立性レベルが特徴づけられる。エントリ基準が失敗ならば、ステップ（又は補償）は「独立性故障（フォルト）」で失敗となる。該故障がサブトランザクションステップで発生したならば、トランザクションは中止され、リカバリーサービスが補償を開始する。該故障が補償で発生するならば、詳細なトレースイベントはロギングされてもよく、サガは中止される。好適な実施形態では、補償トランザクションを実行するときに、故障が検出されると、サガは中止され、人間の介入のための要求が生成される。

【００４２】

ストリンジェントエントリ基準は、次のように維持される：すなわち、各サブトランザクションが実行されるとき、影響を受ける全てのビジネスオブジェクトの次の状態が、トランザクションコンテキストの一部として、例えばリポジトリ２３７に絶え間なくロギングされる。新しい各サブトランザクションステップへエントリされると、テストが実行される。このステップ（オブジェクト識別子によって示される）によって影響を受ける各ビジネスオブジェクトのために、検索がトランザクションのコンテクストで行われる。より詳しくは、影響を受ける各ビジネスオブジェクトは、その特定のビジネスオブジェクトがカレントトランザクションコンテクストで作動したか（所定のトランザクションにおける以前のサブトランザクションステップによって）どうかを決定するために、チェックされ

る。動作された全てのビジネスオブジェクトのために、ビジネスオブジェクトの現在の状態は、トランザクションコンテキストにおいてオブジェクトの最後の既知の状態と照合される。適合する場合には、最後にビジネスオブジェクトがトランザクションコンテキストにおいて見出されたので、状態変更は起こらず、サブトランザクションステップの実行が進行してもよい。各サブトランザクションステップの完了で、所定のトランザクションステップによって影響を受けるビジネスオブジェクトは、トランザクションコンテキストにおいて最新バージョンでアップデートされる。適合しない場合、独立性失敗が起こり、トランザクションは中止されてもよく、そしてリカバリーが発動する。

【００４３】

図４ａは、ストリンジェント独立性サービスで構成されるトランザクションについてのフロー制御を示す。ここで、トランザクションステップが図４（ｂ）に示され、サガサービス２２８によって生成されうるサガが図４（ｃ）に示される。

【００４４】

トランザクション４００が、開始される（４５０）。トランザクション４００と関連付けられた第１のサブトランザクションフォワード実行ステップ４０２は、「新入社員」オブジェクト４０４の作成（CREATE）を提供する。サガが開始され（４５２）、第１のサブトランザクションステップが実行される（４５４）。サガ４２０が生成され、このサブトランザクションのための補償レコード４０６が、補償サブトランザクションによって影響を受ける全てのオブジェクトに関連付けられた状態情報とともに記憶される（４５６）。

【００４５】

より多くのサブトランザクションステップがトランザクションで提供されるかどうかを決定するためにチェックが実行される（４５８）。もしそうでなければ、該トランザクションはコミットされうる（４５９）。より多くのトランザクションステップが処理される場合、特定のビジネスオブジェクトがトランザクションコンテキストにおいて知られている場合に、次のサブトランザクションステップによって影響を受ける全てのビジネスオブジェクトについて決定するためにチェックが行われる（４６０）。以下で議論されているように、該チェックはＡＰＩレベルで実行される。

【００４６】

既知のように、ビジネスオブジェクトは、トランザクション４００の一部として提供されるサブトランザクションによって、既に影響を受けていることを意味する。サブトランザクション又は種々の補償トランザクションの実行の前に、ロジックはトランザクションサービスによって実行されなければならないことに注意されたい。ビジネスオブジェクトが既知でない場合（４６２）、サブトランザクションステップが実行される（４６４）。

【００４７】

ビジネスオブジェクトが既知である場合（４６２）、ビジネスオブジェクトの既存の条件はビジネスオブジェクトの最後の既知の状態と比較される（４６６）。この目的は、最後及び現在の間でトランザクションが実行される時、他のいかなるトランザクションもこのオブジェクトを修正しなかったことを確実にすることである。このチェックが成功すると、次にサブトランザクションはステップ４６４で実行される。もしそうでなければ、リカバリーサービスを起動する独立性故障が識別され（４６８）、レベルｎ－１でトランザクションを補償する実行を開始する。

【００４８】

各サブトランザクションが実行される場合、ちょうどコミットされた特定のサブトランザクションのために、サガは補償トランザクションをロギング（記録）するために修正される。サガ４２０は、レベルで配列される。各レベルは、コミットされたサブトランザクションに対応する。サブトランザクションの実行が競合するたびに、新しいエントリが加えられる。このように、サブトランザクションが失敗するとき、ロールバックは最後の成功したサブトランザクション実行ステップまで進んでもよい。第１のトランザクションステップ４０２がコミットされた後（或いは、同時並行）、次に、補償トランザクションがサガ４２０に加えられる。この例では、トランザクションにおいてコミットされる第１の

トランザクションステップを元に戻す（UNDO）ための補償トランザクションは、新しく作成（CREATE）された従業員オブジェクトの削除を含む（削除（DELETE）動詞４１０及びオブジェクト４１２によって示される）。トランザクション４００のステップがコミットされるように、サガへのエントリが実行され、サブトランザクションステップを体系的に元に戻す（UNDO）ことを許す。全てのトランザクションが完了する場合、該トランザクションがコミットされたと見なされ、そして該サガが捨てられる。

【００４９】

サブトランザクションロッキング及びオブジェクトチェック

サブトランザクション動作が完了されるまで、サブトランザクションレベルでのロッキングは、トランザクションの独立性を確実にするために要求されてもよい。しかしながら、ロッキングはトランザクションレベルで起こらない。具体的には、所定のサブトランザクションによって影響を受けるように、ビジネスオブジェクトが識別されたとき、特定のサブトランザクションステップが完了され、全てのオブジェクトデータが格納されるまで、ビジネスオブジェクトに関連付けられたレコードは所定期間ロックされてもよい。このロッキング機能は、”独立性レベル読み取り”を通じて呼び出される。”独立性レベル読み取り”、データの一時的なロッキングをもたらし、所定のサブトランザクションステップに関連付けられた何らかの変更（CHANGE）が呼び出される。該サブトランザクションが完了されるまで、ロックはレコードに置かれる。

【００５０】

状態検証サービス２２５は、サブトランザクションステップで影響を受ける各オブジェクトの状態をチェックするために呼び出されうる。状態検証サービス２２５は、サーバ２２０でこれらのチェックを実行してもよく、又はチェック機能を実行するためにＡＰＩにおける機能を利用してもよい。

【００５１】

例えば、フォワード進行の間、トランザクションサブシステムは、各サービスレベルのための上記した、動作の必要な状態情報（リポジトリを使用する）を最適に格納する。（オペレーティングシステムとともに）ネットワークサブシステムは、各サブトランザクションステップのために、ＡＰＩに動作及び状態情報を送ってもよい。リクエストがＡＰＩに到着するとき、ＡＰＩは原子性テスト及びセット動作を実行するために、動作情報とともに状態情報を使用してもよい。原子性テスト及びセット動作が成功であるならば、成功状態がサーバマシンに（及び最終的にビジネス論理）に返され、さもなければ失敗状態が返される。動作がいつでも失敗するならば、次に「実行を元に戻すこと（UNDO）」のプロセスが、リカバリーサービスを通して開始される。

【００５２】

等価決定がＡＰＩによってなされることができるまで、ＡＰＩ２４０はアプリケーション２５１において原子性テスト及びセット動作を呼び出して、データベース２５２からのレコードを読みそしてレコードをロックしうる。特に、トランザクションシステムは、所定のレコード及び変更情報（特定のサブトランザクションステップによって示される）に関連づけられる特定のビジネスオブジェクトの最後の既知状態がＡＰＩに運ばれる。最後の既知状態がトランザクションコンテキストのオブジェクトの状態と一致しているならば、レコードへのアップデートが実行される。他に、アップデートはＡＰＩによって実行されない。実行されるロッキングは、再びサブトランザクションレベルで取り扱われるだけである点に留意されたい。従って、データベース２５２でのデータレコードの長期ロッキングは、サーバマシン２２０上で実行しているトランザクションが長期間でありうることを支持するために、要求されない。

【００５３】

独立性故障が識別されたとき、サガキューに示される補償トランザクションは、一貫した状態にロールバックするために、先入れ後出し方法論により実行してもよい。ストリンジェント独立性レベルのために、ロールバック操作は実行時実行に類似した方法で実行される。すなわち、各補償トランザクションの実行の前に、所定補償トランザクションステ

ップによって影響される全てのビジネスオブジェクトの状態が検証されて、既知のトランザクションコンテキストかどうか決定される。どんなオブジェクトでも既知でその状態が最後の既知状態と一致しない場合、トランザクションコンテキストの範囲内で定められるように、独立性故障が引き起こされる。補償トランザクションの運営において起こる独立性故障は、詳細なトレースイベントのロギングで終わり、サガを中止してもよい。

【００５４】

最良のエフォート

”ストリンジェント”独立性レベル及び”最良のエフォート”（ベストエフォート）独立性の間の違いは、非常に微妙である。１つの顕著な例外で、最良のエフォートはストリンジェントと同様なアルゴリズムを使用する（すなわち、各サブトランザクションステップの前に、等価チェックが、オブジェクトの記憶された前の状態とトランザクションコンテキストにおけるオブジェクトの現在の状態との間で実行される）。等価チェックが行われるとき、”独立性レベル読み取り”を使用しているデータレコードはチェックされない。オブジェクトは、記憶された以前の状態と比較したデータベース（ＡＰＩによって）から単に読みだすだけであり、値が等しい場合、動作は進行する。チェックが失敗するならば、取消し操作はサガのために開始される。トランザクションシステムが等価チェックの間、繰り返し可能な読取りを必要としていないので、当業者は、データレコードでの整合性問題に帰着する、ウインドウが開けられていると認められる。ここでの意図は、繰り返し可能な読取りがＡＰＩで支持されていない又はＡＰＩから入手できる環境で、できるだけ同じ整合性を提供することである。

【００５５】

最小のエフォート

”最小のエフォート”独立性及び”最良のエフォート”独立性の間の違いは、トランザクションサービスが、サブトランザクションステップを実行するか、トランザクションを補償する前に、少しの等価チェックも試みないということである。ここでの意図は、交換が既知であるトランザクション混合のために、独立性パラダイムをより良く実行することである。しかし、このパラダイムを使用することは、トランザクションデザイナーが彼らのトランザクションの特定の意味を知っていることを必要とする。再び、サブトランザクションステップが失敗する場合（すなわち、否定応答がアプリケーションから受容される場合）、トランザクションサービスはサガに関連付けられたアンドゥ（UNDO）セマンティクスを呼び出してもよい。エラーの点までうまく実行される各サブトランザクションステップのために、補償トランザクションが実行されてもよい。

【００５６】

エフォートなし

”エフォートなし”独立性パラダイムは、独立性を全く提供しない。それは単に、サブトランザクションステップが失敗するとき、詳細なイベントトレースレコードがロギングされ、人間の介入（及びデータの一致）が可能となり、トランザクションは停止する。このパラダイムで、サブトランザクションステップが失敗するとき、サガは停止しロールバックを試みないことが重要である点に留意されたい。むしろ、故障事象がロギングされ、人間の介入が要請される。

【００５７】

フォワード進行及び持続的サービス

ユーザが独立性レベルを選びビジネス論理が実行を開始した後、ネットワークサブシステムがアプリケーションＡＰＩから実行され機能性が利用される前に、トランザクションシステムに関連付けられた持続的サービスが、オブジェクト状態情報を追跡する。

【００５８】

各動作がＡＰＩを使用して実行されるように、トランザクションシステムが介入して状態情報を加える。これにより、独立性（要請されるならば）は、フォワードトランザクション進行の間（同様に、バックワード進行／ロールバック、必要な場合）に維持されうる。これは、サブトランザクション又は補償トランザクションステップによって影響を受け

る各オブジェクトの状態を格納することによって達成される。

【００５９】

図５は、トランザクションコンテキスト５００を表す。トランザクションコンテキストは、トランザクションシステムによって使用される持続的構造を画成し、進行するトランザクションが元に戻す（UNDO）又は回復する（RECOVER）のに必要な状態情報及び補償情報を追跡する。好適な実施形態で、コンテクストの平行なイメージは、ランタイムで参照付けられるメモリに格納される。

【００６０】

該コンテクストは、コラボレーションネーム５０２、スレッドＩＤ５０４、独立性構成５０６、現在のサブトランザクションステップ５０８のインデックス、補償レコード５１０、サブトランザクションステップレコード５１２及びサガスケルトン５１４を含む。

【００６１】

コラボレーションネーム５０２は、特定のトランザクションコンテキストを使用するコラボレーション（トランザクション）の名前を参照する。スレッドＩＤ５０４は、それが実行しているトランザクションのインスタンスを表す。スレッドＩＤ５０４は、実行しているトランザクションのインスタンスを表す。スレッドＩＤは、トランザクション（又はコラボレーションの例）をユニークに識別し、特定のトランザクションを実行しているオペレーティングシステムのスレッドＩＤではないことに留意されたい。コラボレーションネーム及びスレッドＩＤを使用して、トランザクションサービスは、トランザクションのインスタンスのために持続的トランザクションコンテキストをさがしてもよい。独立性構成５０６は、ユーザによって要請された独立性レベルを識別する。現在のサブトランザクションステップ５０８のインデックスは、現在実行しているサブトランザクションステップを示す。この情報は、リカバリーサービスによって必要とされる。補償レコード５１０は、サブトランザクションステップの意味論的なアンドゥ（UNDO）を提供する。各補償レコードは、ビジネスオブジェクトから成る（ここで、ビジネスオブジェクトの状態が、各サブトランザクションステップ実行の間、持続的サービスによって記入されることが重要である）。ＡＰＩが関係する限り、他の動作と同様に、補償動作は単に、動詞を持ったビジネスオブジェクトである。

【００６２】

サブトランザクションステップレコード５１２は、”最良のエフォート”又は”ストリンジェント独立性レベル”を使用しているトランザクションだけのために使用される。我々が最後に出会ったように、これらのレコードはビジネスオブジェクトの状態を表す。従って、サブトランザクションステップを実行する前に、トランザクションサービスは、その状態がサブトランザクションステップレコードで記憶される状態と一致するということを知るために、オブジェクトをチェックする。

【００６３】

サガスケルトン５１４は、レコードの順序付けられたリストである。各レコードは、トランザクション内でユニークなサブトランザクションステップに対応する補償ステップを（少なくとも）表す。サガスケルトンは、何らかの影響を受けるビジネスオブジェクトでも、対応する補償動詞及びこの補償レコードと対応するサブトランザクションステップの状態インデックスのネームを含む。関連したサブトランザクションの実行の結果として展開された場合、スペースが、実行時ビジネスオブジェクト状態の記憶装置のために、トランザクションコンテキストにおいて確保されてもよい。

【００６４】

サガ

サガは、サガサービス２２８（図２）によって作成され且つ修正される。サガは、ポインタ（又はサブトランザクションインデックス）５１８、ステップ５２０によって影響を受けるだろう全てのビジネスオブジェクトの現在の状態、補償のために使用される動詞５１６、及び補償動詞で使用される値５１５（又はビジネスオブジェクト）を含む補償トランザクションステップを含む。補償トランザクションが以前に実行され該関連付けられた

サブトランザクションステップに、ポインタ５１８は「アンドゥ（UNDO）」動作を示す。オブジェクトがサガにおいて少し前のステップで修正されなかったならば、現在の状態はアプリケーションからのその状態である。

【００６５】

リカバリーサービス

トランザクションサービスのリカバリー部分が、中断されたサガに代わってトランザクションを補償することを実行するために必要である（それは順に、実行するトランザクションの代わりである）。リカバリーは、補償トランザクション（ユーザによって要請されたように）を実行するときに、正しい独立性レベルセマンティクスを確実にするように、適切なロジックを実行することに対して責任を負う。リカバリーサービスは、サガによって定義される補償トランザクションの実行を監督する。リカバリー実行は、オリジナル基本的トランザクションの実行として同じ独立性レベルで実行されるだろう。従って、ストリンジェント独立性レベルが選択されるならば、”ストリンジェント”独立性レベルロッキング又は繰り返し可能な読み取りが、ロールバック操作において使用されるだろう。代替的に、必要ならば、異なる独立性レベルが、特にリカバリーのために、ユーザによって選択されるだろう。

【００６６】

ロールバックは、サガにおいて格納された補償トランザクションの幾つか又は全ての実行を含みうる。ロールバックは、最後に（又はユーザ選択で）持続状態で止まり、そして終了しうる。代替的に、コンティンジェンシサービスが、リカバリーステップの完了の後に、フォワードトランザクションを続けることを許すために提供されうる。コンティンジェンシサービスは、独立性故障が識別された後に、トランザクションのフォワード進行を続けるために、フォワードをジャンピングすることを含む。例えば、最初の独立性故障が受け取られ、及びロールバックが始まった後、（データベースのための）最後の持続状態が達成されるまで、ロールバックは進むだろう。その時点で、コンティンジェンシサービスは、トランザクションにおける他の位置にフォワードをジャンプし、フォワード実行を再び始める。ロールバック及びコンティンジェンシオプションがユーザ定義可能であり、望ましい特定のビジネス機能を達成するために開発者によって独自にプログラムされうる。

【００６７】

図６では、本発明に従うトランザクションシステムによってトランザクションを処理するためのフロー図６００が示される。ユーザは、所望の（「要請されたレベル」）且つ最小の独立性レベルであって、実行される所定のビジネス論理機能に関連付けられた独立性レベルを定義することを指示される（６０２）。サポートの要請されたレベルが利用できる場合に決定するための所定のトランザクションによって起動される各ＡＰＩに対して、要請されたサービスレベルが検査される（６０４）。要請されたレベルが利用可能であれば、独立性レベルは、リポジトリにおいて格納される（６０６）。要請されたレベルがサポートされないならば、次にコネクターの最低の共通レベルが決定される（６０７）。最低の共通レベルが最小の要請されたレベル（６０８）よりも下の場合に故障が記録され、そしてコラボレーションが実行されることは許されない（６０９）。さもないと、最低の共通レベルが、リポジトリにおける独立性レベルとして記録される。特定のトランザクションは、実行時モードに移行され、そして該トランザクションはコラボレーションのサポートで実行されうる。

【００６８】

実行時、該コラボレーションに関連付けられたトランザクションが、実行される（６１０）。各トランザクションステップで、指定された該関連する独立性レベルを決定するためにチェックがなされ、次にサブトランザクションが独立性レベルに従って実行される（６１２）。結局、独立性故障が識別され（６１４）、トランザクションは、独立性レベルに従いロールバックし（６１６）又は中止される（６２２）。ロールバックは、独立性のレベルについての検査を含み、トランザクションの独立性レベルに従って、補償サブトラ

ンザクションを実行する。さもなくば、トランザクションはコミッティング（６２２）まで続く。

【００６９】

インプリメンテーション（実装）

発明は、ソフトウェアのハードウェア又は両方のコンビネーションで実行されるだろう。しかし、本発明は、各々、プロセッサ、データ記憶システム（揮発性及び不揮発性メモリ及び／又はストレージ要素を含む）、少なくとも１つの入力装置及び少なくとも１つの出力装置、を含むプログラム可能コンピュータ上で実行しているコンピュータプログラムで、好ましくは実行される。プログラムコードは、本明細書で説明される機能を実行するために入力データに加えられ、出力情報を発生させる。既知のやり方で、出力情報は１つ以上の出力装置に加えられる。

【００７０】

各プログラムは、コンピュータシステムに通じているためにハイレベル手続又はオブジェクト指向プログラミング言語で、好ましくは実行される。しかし、希望する場合、プログラムはアセンブリ又はマシン語で実行されることができる。いずれにせよ、言語はコンパイルされまたは翻訳された言語であるだろう。

【００７１】

そのような各コンピュータプログラムは、一般的な又は特殊目的プログラム可能なコンピュータで読み取り可能な、記憶媒体又は装置（例えばＲＯＭ、ＣＤＲＯＭ又はマグネティックディスケット）で、記憶媒体又は装置が本明細書で説明した手順を実行するために、コンピュータで読み込まれるときコンピュータを構成して及び動作するために好ましくは記録される。本発明のシステムは、コンピュータプログラムで構成されるコンピュータ読取り可能な記憶媒体として実行されると考えられ、コンピュータが、本明細書で説明した機能を実行するために、特定の及び所定の態様で動作するように記録媒体は構成されるであろう。

【００７２】

幾つかの変形と共に、本発明の好ましい実施形態を説明した。それにもかかわらず、種々の変更が精神及び発明の範囲から逸脱することなくされるだろうことが理解される。従って、発明が特定の示された実施形態によって制限されるべきでなく、添付の請求項の範囲によってのみ制限されるべきであることが、理解されるべきである。

【図面の簡単な説明】

【図１】　本発明に従うモジューラアプリケーションコラボレーションの概略ブロック図である。

【図２】　本発明に従うトランザクションシステムを含む、分散されたコンピュータシステムの概略ブロック図である。

【図３ａ】　本発明に従うトランザクションの概略ブロック図である。

【図３ｂ】　本発明に従う補償トランザクションの概略ブロック図である

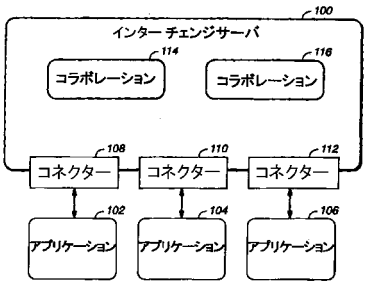
【図４ａ】　本発明に従うデータベース一貫性を提供する方法の流れ図である。

【図４ｂ】　図４ｂは、図４ａのフロー図に関連付けられたサンプルトランザクションである。図４ｃは、図４ａのフロー図に関連して生成されたサンプルサガである。

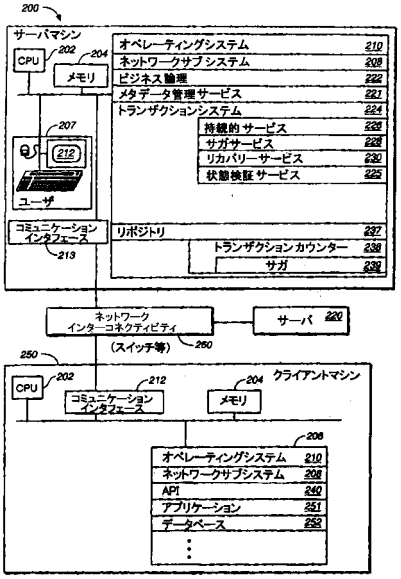
【図５】　本発明に従うトランザクションセマンティクスを格納するためのトランザクションコンテキストのための体系的なブロック図である。

【図６】　本発明に従う長寿命トランザクションのためフロー図である。

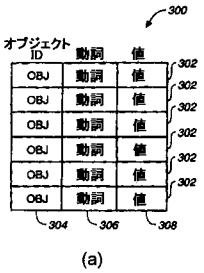
【図１】



【図２】

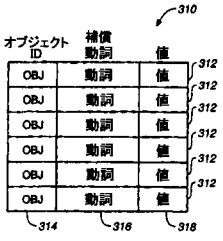


【図３ａ】



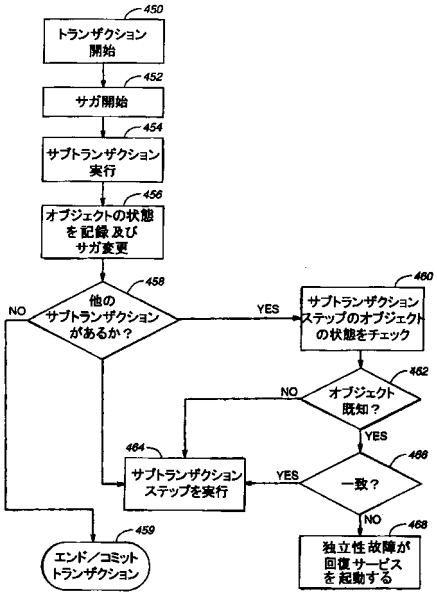
(a)

【図３ｂ】

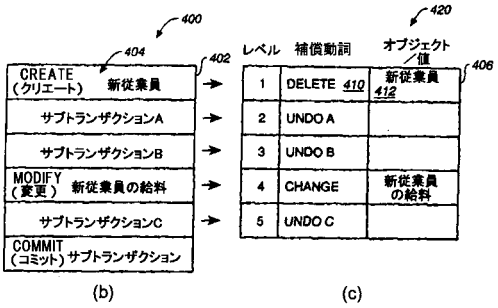


(b)

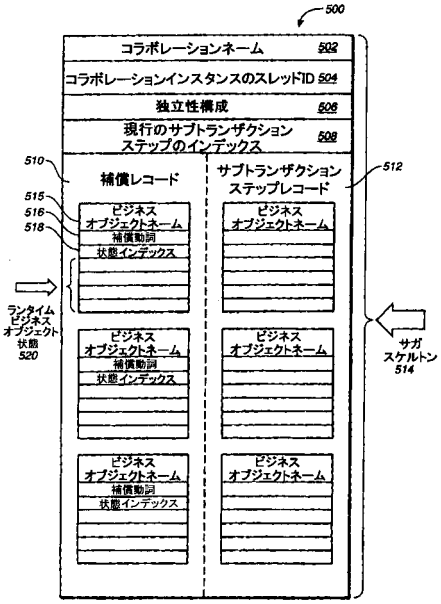
【図４ａ】



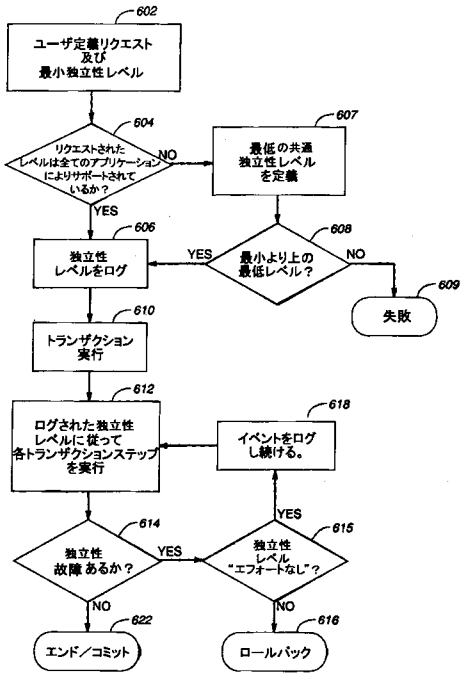
【図４ｂ】


400
404
402
420
406
CREATE (クリエート) 新従業員
サブトランザクションA
サブトランザクションB
MODIFY (変更) 新従業員の給料
サブトランザクションC
COMMIT (コミット) サブトランザクション
レベル
補償動詞
オブジェクト値
1 DELETE 410 新従業員 412
2 UNDO A
3 UNDO B
4 CHANGE 新従業員の給料
5 UNDO C
(b)
(c)

【図５】


500
コラボレーションネーム 502
コラボレーションインスタンスのスレッドID 504
独立性構成 506
現行のサブトランザクション ステップのインデックス 508
510
補償レコード
サブトランザクション ステップレコード
512
515
516
518
ビジネス オブジェクトネーム
補償動詞
状態インデックス
ランタイム ビジネス オブジェクト 状態 520
サガ スケルトン 514

【図６】


602
ユーザ定義リクエスト 及び 最小独立性レベル
604
リクエストされた レベルは全てのアプリケーション によりサポートされて いるか？
NO
YES
607
最低の共通 独立性レベル を定義
606
独立性 レベルをログ
YES
608
最小より上の 最低レベル？
NO
609
失敗
610
トランザクション 実行
612
ログされた独立性 レベルに従って 各トランザクションステップ を実行
618
イベントをログ し続ける。
YES
614
独立性 故障あるか？
YES
615
独立性 レベル “エフォートなし”？
NO
NO
622
エンド／コミット
616
ロールバック

フロントページの続き

(72)発明者 グピタ, プラシャント
アメリカ合衆国, カリフォルニア州, モンタレイ, フランクリン ストリート １０３７
(72)発明者 ルビン, デイヴィッド, エス.
アメリカ合衆国, カリフォルニア州, サン フランシスコ, サンシェッズ ストリート ９０３

審査官 高瀬 勤

(56)参考文献 特開平１０－０６９４１８（ＪＰ，Ａ）
特開平０７－２９５８７１（ＪＰ，Ａ）
特開平０８－１１５２４６（ＪＰ，Ａ）
特開平０６－２４３０７２（ＪＰ，Ａ）
特開平０９－３１９６３３（ＪＰ，Ａ）
金井，白木原，階層トランザクション機構によるＵＮＩＸ上の高信頼分散処理環境，情報処理学会研究報告，日本，社団法人情報処理学会，１９９２年 1月２４日，第92巻,第8号，(92-ARC-92-8)

(58)調査した分野(Int.Cl.，ＤＢ名)
G06F 12/00
G06F 17/30
JSTPlus(JDream2)