(19)日本国特許庁(JP)　(12)特　許　公　報(B2)　(11)特許番号
特許第4986418号
(P4986418)
(45)発行日　平成24年7月25日(2012.7.25)　(24)登録日　平成24年5月11日(2012.5.11)

(51)Int.Cl.　F I
G06Q 10/06 (2012.01)　G06F 17/60　164

請求項の数 24　外国語出願　(全 25 頁)

(21)出願番号　特願2005-189356(P2005-189356)
(22)出願日　平成17年6月29日(2005.6.29)
(65)公開番号　特開2006-99730(P2006-99730A)
(43)公開日　平成18年4月13日(2006.4.13)
審査請求日　平成20年6月27日(2008.6.27)
(31)優先権主張番号　10/952,284
(32)優先日　平成16年9月28日(2004.9.28)
(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　100077481
弁理士　谷　義一
(74)代理人　100088915
弁理士　阿部　和夫
(72)発明者　ラジェンドラ　エイチ．ビシュヌマーティ
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　コーポレーション内

最終頁に続く

(54)【発明の名称】プロジェクトデータのキャッシュおよび同期をとる方法およびシステム

(57)【特許請求の範囲】
【請求項１】
クライアント装置のプロジェクト管理アプリケーションとプロジェクトサーバの間のプロジェクトデータの同期を管理するコンピュータ実施方法であって、
前記コンピュータ実施方法は、
前記プロジェクトサーバに格納されているプロジェクトスケジュールのコピーを前記クライアント装置にて取得するステップであって、前記プロジェクトスケジュールのコピーは、複数のプロジェクトタスクを含み、前記プロジェクトスケジュールのコピーは、前記クライアント装置のキャッシュメモリに格納され、前記キャッシュメモリは、前記プロジェクトスケジュールのコピーとは別のプロジェクトタスクのテーブルを含み、前記プロジェクトタスクのそれぞれは、前記プロジェクトタスクのバージョンを示すバージョンスタンプを含む、ステップと、
前記プロジェクトスケジュールのコピーの複数のプロジェクトタスクの少なくとも一つに対する変更を前記クライアント装置にて受信するステップであって、前記変更は、前記クライアント装置のキャッシュメモリに維持される、ステップと、
前記変更が、前記クライアント装置のキャッシュマネージャとともに格納されているプリセットの変更閾値を満たすサイズのデータ更新を含むかどうかを前記キャッシュマネージャによって判定するステップであって、前記変更閾値は、前記キャッシュメモリに格納されている前記プロジェクトスケジュールのコピーのデータ全体に関して前記キャッシュメモリに維持されているデータ更新の許容量のパーセントの閾値である、ステップと、

前記変更閾値が満たされたと判定すると、前記クライアント装置のキャッシュマネージャに、
前記キャッシュメモリに維持されている前記データ更新のそれぞれの変更のプロパティを識別させ、
前記変更のプロパティで前記プロジェクトサーバに格納されている前記プロジェクトスケジュールに対する更新を引き起こすために、前記プロジェクトスケジュールの不変部分のプロパティを送信することなく、それぞれの変更のプロパティを前記プロジェクトサーバに送信させ、
それぞれの変更に対する更新されたバージョンスタンプを前記プロジェクトサーバから受信させ、
前記キャッシュマネージャにある前記テーブルを前記バージョンスタンプで更新させる
ステップと
を備えたことを特徴とするコンピュータ実施方法。
【請求項２】
前記キャッシュマネージャに、タスク更新コンテナ、新タスクコンテナ、および削除タスクコンテナを生成させるステップをさらに備えたことを特徴とする請求項１に記載のコンピュータ実施方法。
【請求項３】
前記変更のプロパティが前記複数のプロジェクトタスクの一つに対する変更に関連することを前記キャッシュマネージャによって識別するステップと、
前記変更を前記タスク更新コンテナに格納するステップと、
前記タスク更新コンテナを前記プロジェクトサーバに送信するステップと、
前記変更についての更新されたバージョンスタンプを前記プロジェクトサーバから受信するステップと、
前記変更の関連タスクに関連して前記キャッシュマネージャのテーブルに前記更新されたバージョンスタンプを格納するステップと
さらに備えたことを特徴とする請求項２に記載のコンピュータ実施方法。
【請求項４】
前記変更のプロパティが新しいタスクに関連することを前記キャッシュマネージャによって識別するステップと、
前記変更を前記新タスクコンテナに格納するステップと、
前記新タスクコンテナを前記プロジェクトサーバに送信するステップと、
前記変更についての更新されたバージョンスタンプを前記プロジェクトサーバから受信するステップと、
前記新しいタスクのインジケーションを前記キャッシュマネージャのテーブルに生成するステップと、
前記新しいタスクについて生成されたインジケーションに関連して前記キャッシュマネージャのテーブルに前記更新されたバージョンスタンプを格納するステップと
をさらに備えたことを特徴とする請求項２に記載の方法。
【請求項５】
前記変更のプロパティが削除タスクに関連することを前記キャッシュマネージャによって識別するステップと、
前記変更を前記削除タスクコンテナに格納するステップと、
前記削除タスクコンテナを前記プロジェクトサーバに送信するステップと、
前記変更についての更新されたバージョンスタンプを前記プロジェクトサーバから受信するステップと、
前記削除タスクのインジケーションを前記キャッシュマネージャのテーブルから取り除くステップと、
タスクシャドーテーブルを前記キャッシュマネージャに生成するステップと、

前記削除タスクのインジケーションを前記キャッシュマネージャのタスクシャドーテーブルに生成するステップと、
前記削除タスクについて生成されたインジケーションに関連して前記キャッシュマネージャのタスクシャドーテーブルに前記更新されたバージョンスタンプを格納するステップと
をさらに備えたことを特徴とする請求項２に記載の方法。
【請求項６】
前記複数のプロジェクトタスクを含む前記プロジェクトスケジュールのコピーを取得するステップは、前記プロジェクトスケジュールの変更部分を前記プロジェクトサーバから取得すること、および前記キャッシュメモリに格納されている前記プロジェクトスケジュールのバージョンを更新して前記キャッシュメモリのプロジェクトスケジュールの現在バージョンを取得することを含むことを特徴とする請求項１に記載の方法。
【請求項７】
前記プロジェクトスケジュールの変更部分を前記プロジェクトサーバから取得することは、前記プロジェクトスケジュールの変更部分を二つの別個のスレッドで受信することを含み、第１のスレッドは、前記キャッシュマネージャによって最初に受信される高い優先度のスレッドであり、第２のスレッドは、前記キャッシュマネージャによって２番目に受信される低い優先度のスレッドであることを特徴とする請求項６に記載の方法。
【請求項８】
クライアント装置のプロジェクト管理アプリケーションとプロジェクトサーバの間のプロジェクトデータの同期を管理するコンピュータ実行可能命令を格納するコンピュータ読み取り可能な記憶媒体であって、
前記コンピュータ実行可能命令は、実行されると、前記クライアント装置に、
前記プロジェクトサーバに格納されているプロジェクトスケジュールのコピーを取得するステップであって、前記プロジェクトスケジュールのコピーは、複数のプロジェクトタスクを含み、前記プロジェクトスケジュールのコピーは、前記クライアント装置のメモリに格納され、前記メモリは、バージョンスタンプに関連した複数のプロジェクトタスクのインジケーションを含む、前記プロジェクトスケジュールのコピーとは別のプロジェクトタスクのテーブルを含む、ステップと、
前記プロジェクトスケジュールのコピーの複数のプロジェクトタスクの少なくとも一つのプロジェクトタスクに対する変更を受信するステップであって、前記変更は、前記プロジェクトサーバに送信されなかった他のプロジェクトタスクに対する変更を含む他のデータ更新とともに前記クライアント装置のメモリ内に格納される、ステップと、
前記変更および以前に行われた他の変更が、メモリマネージャのプリセットの変更閾値を満たすかどうかを前記メモリマネージャによって判定するステップであって、前記プリセットの変更閾値は、前記プロジェクトタスクの総数に関連して修正されたタスクの数である、ステップと、
前記変更閾値が満たされたと判定すると、前記メモリマネージャに、
前記変更のプロパティを識別させ、
前記変更のプロパティで前記プロジェクトサーバに格納されている前記プロジェクトスケジュールに対する更新を引き起こすために、前記プロジェクトスケジュールの不変部分のプロパティを送信することなく、前記変更のプロパティを前記プロジェクトサーバに送信させ、
前記変更に対する更新されたバージョンスタンプを前記プロジェクトサーバから受信させ、
前記メモリマネージャにある前記テーブルを前記バージョンスタンプで更新させる
ステップと
を含む方法を実行させることを特徴とするコンピュータ読み取り可能な記憶媒体。
【請求項９】
前記変更閾値は、前記メモリに格納されている前記プロジェクトスケジュールのコピー

のデータ全体に関して更新データの許容量のパーセントの閾値であることを特徴とする請求項８に記載のコンピュータ読み取り可能な記憶媒体。

【請求項１０】

前記メモリマネージャに、タスク更新コンテナ、新タスクコンテナ、および削除タスクコンテナを生成させるステップをさらに含むことを特徴とする請求項８に記載のコンピュータ読み取り可能な記憶媒体。

【請求項１１】

前記変更のプロパティが前記複数のプロジェクトタスクの一つに対する変更に関連することを前記メモリマネージャによって識別するステップと、

前記変更を前記タスク更新コンテナに格納するステップと、

前記タスク更新コンテナを前記プロジェクトサーバに送信するステップと、

前記変更についての更新されたバージョンスタンプを前記プロジェクトサーバから受信するステップと、

前記変更の関連タスクに関連して前記メモリマネージャのテーブルに前記更新されたバージョンスタンプを格納するステップと

さらに含むことを特徴とする請求項１０に記載のコンピュータ読み取り可能な記憶媒体。

【請求項１２】

前記変更のプロパティが新しいタスクに関連することを前記メモリマネージャによって識別するステップと、

前記変更を前記新タスクコンテナに格納するステップと、

前記新タスクコンテナを前記プロジェクトサーバに送信するステップと、

前記変更についての更新されたバージョンスタンプを前記プロジェクトサーバから受信するステップと、

前記新しいタスクのインジケーションを前記メモリマネージャのテーブルに生成するステップと、

前記新しいタスクについて生成されたインジケーションに関連して前記メモリマネージャのテーブルに前記更新されたバージョンスタンプを格納するステップと

をさらに含むことを特徴とする請求項１０に記載のコンピュータ読み取り可能な記憶媒体。

【請求項１３】

前記変更のプロパティが削除タスクに関連することを前記メモリマネージャによって識別するステップと、

前記変更を前記削除タスクコンテナに格納するステップと、

前記削除タスクコンテナを前記プロジェクトサーバに送信するステップと、

前記変更についての更新されたバージョンスタンプを前記プロジェクトサーバから受信するステップと、

前記削除タスクのインジケーションを前記メモリマネージャのテーブルから取り除くステップと、

タスクシャドーテーブルを前記メモリマネージャに生成するステップと、

前記削除タスクのインジケーションを前記メモリマネージャのタスクシャドーテーブルに生成するステップと、

前記削除タスクについて生成されたインジケーションに関連して前記メモリマネージャのタスクシャドーテーブルに前記更新されたバージョンスタンプを格納するステップと

をさらに含むことを特徴とする請求項１０に記載のコンピュータ読み取り可能な記憶媒体。

【請求項１４】

前記複数のプロジェクトタスクを含む前記プロジェクトスケジュールのコピーを取得するステップは、前記プロジェクトスケジュールの変更部分を前記プロジェクトサーバから取得すること、および前記メモリに格納されている前記プロジェクトスケジュールのバー

ジョンを更新して前記メモリのプロジェクトスケジュールの現在バージョンを取得することを含むことを特徴とする請求項８に記載のコンピュータ読み取り可能な記憶媒体。

【請求項１５】

前記プロジェクトスケジュールの変更部分を前記プロジェクトサーバから取得することは、前記プロジェクトスケジュールの変更部分を二つの別個のスレッドで受信することを含み、第１のスレッドは、前記メモリマネージャによって最初に受信される高い優先度のスレッドであり、第２のスレッドは、前記メモリマネージャによって２番目に受信される低い優先度のスレッドであることを特徴とする請求項１４に記載のコンピュータ読み取り可能な記憶媒体。

【請求項１６】

クライアント装置のプロジェクト管理アプリケーションとプロジェクトサーバの間のプロジェクトデータの同期を管理するシステムであって、

前記システムは、

プロセッサと、

コンピュータ実行可能命令を格納するメモリと

を備え、

前記コンピュータ実行可能命令は、実行されると、前記クライアント装置に、

前記プロジェクトサーバに格納されているプロジェクトスケジュールのコピーを取得するステップであって、前記プロジェクトスケジュールのコピーは、複数のプロジェクトタスクを含み、前記プロジェクトスケジュールのコピーは、前記クライアント装置のメモリに格納され、前記メモリは、バージョンスタンプに関連した複数のプロジェクトタスクのインジケーションを含む、前記プロジェクトスケジュールのコピーとは別のプロジェクトタスクのテーブルを含む、ステップと、

前記プロジェクトスケジュールのコピーの複数のプロジェクトタスクの少なくとも一つのプロジェクトタスクに対する変更を受信するステップであって、前記変更は、データ更新とともに前記クライアント装置のメモリに維持される、ステップと、

前記変更を含む前記データ更新が、前記クライアント装置のメモリマネージャのプリセットの変更閾値を満たすかどうかを前記メモリマネージャによって判定するステップと、

前記変更閾値が満たされたと判定すると、前記クライアント装置のメモリマネージャに、

前記変更のプロパティを識別させ、

前記変更のプロパティで前記プロジェクトサーバに格納されている前記プロジェクトスケジュールに対する更新を引き起こすために、前記プロジェクトスケジュールの不変部分のプロパティを送信することなく、前記変更のプロパティを前記プロジェクトサーバに送信させ、

前記変更に対する更新されたバージョンスタンプを前記プロジェクトサーバから受信させ、

前記メモリマネージャにある前記テーブルを前記バージョンスタンプで更新させる

ステップと

を含む方法を実行させることを特徴とするシステム。

【請求項１７】

前記変更閾値は、前記メモリに格納されている前記プロジェクトスケジュールのコピーのデータ全体に関して更新データの許容量のパーセントの閾値であることを特徴とする請求項１６に記載のシステム。

【請求項１８】

前記変更閾値は、前記プロジェクトスケジュールの複数のプロジェクトタスクに関連してプロジェクトタスクの変更のパーセントであることを特徴とする請求項１６に記載のシステム。

【請求項１９】

前記メモリマネージャに、タスク更新コンテナ、新タスクコンテナ、および削除タスク

コンテナを生成させるステップをさらに含むことを特徴とする請求項１６に記載のシステム。

【請求項２０】

前記変更のプロパティが前記複数のプロジェクトタスクの一つに対する変更に関連することを前記メモリマネージャによって識別するステップと、

前記変更を前記タスク更新コンテナに格納するステップと、

前記タスク更新コンテナを前記プロジェクトサーバに送信するステップと、

前記変更についての更新されたバージョンスタンプを前記プロジェクトサーバから受信するステップと、

前記変更の関連タスクに関連して前記メモリマネージャのテーブルに前記更新されたバージョンスタンプを格納するステップと

さらに含むことを特徴とする請求項１９に記載のシステム。

【請求項２１】

前記変更のプロパティが新しいタスクに関連することを前記メモリマネージャによって識別するステップと、

前記変更を前記新タスクコンテナに格納するステップと、

前記新タスクコンテナを前記プロジェクトサーバに送信するステップと、

前記変更についての更新されたバージョンスタンプを前記プロジェクトサーバから受信するステップと、

前記新しいタスクのインジケーションを前記メモリマネージャのテーブルに生成するステップと、

前記新しいタスクについて生成されたインジケーションに関連して前記メモリマネージャのテーブルに前記更新されたバージョンスタンプを格納するステップと

をさらに含むことを特徴とする請求項１９に記載のシステム。

【請求項２２】

前記変更のプロパティが削除タスクに関連することを前記メモリマネージャによって識別するステップと、

前記変更を前記削除タスクコンテナに格納するステップと、

前記削除タスクコンテナを前記プロジェクトサーバに送信するステップと、

前記変更についての更新されたバージョンスタンプを前記プロジェクトサーバから受信するステップと、

前記削除タスクのインジケーションを前記メモリマネージャのテーブルから取り除くステップと、

タスクシャドーテーブルを前記メモリマネージャに生成するステップと、

前記削除タスクのインジケーションを前記メモリマネージャのタスクシャドーテーブルに生成するステップと、

前記削除タスクについて生成されたインジケーションに関連して前記メモリマネージャのタスクシャドーテーブルに前記更新されたバージョンスタンプを格納するステップと

をさらに含むことを特徴とする請求項１９に記載のシステム。

【請求項２３】

前記複数のプロジェクトタスクを含む前記プロジェクトスケジュールのコピーを取得するステップは、前記プロジェクトスケジュールの変更部分を前記プロジェクトサーバから取得すること、および前記メモリに格納されている前記プロジェクトスケジュールのバージョンを更新して前記メモリのプロジェクトスケジュールの現在バージョンを取得することを含むことを特徴とする請求項１６に記載のシステム。

【請求項２４】

前記プロジェクトスケジュールの変更部分を前記プロジェクトサーバから取得することは、前記プロジェクトスケジュールの変更部分を二つの別個のスレッドで受信することを含み、第１のスレッドは、前記メモリマネージャによって最初に受信される高い優先度のスレッドであり、第２のスレッドは、前記メモリマネージャによって２番目に受信される

低い優先度のスレッドであることを特徴とする請求項２３に記載のシステム。

【発明の詳細な説明】

【技術分野】

【０００１】

本発明は、一般に、プロジェクト管理方法およびプロジェクト管理システムに関する。より詳細には、本発明は、プロジェクト管理アプリケーションを使用する際にプロジェクトデータを高度に自動化して（ｉｎｔｅｌｌｉｇｅｎｔｌｙ）管理するための方法およびシステムに関する。

【背景技術】

【０００２】

コンピュータ時代の到来とともに、コンピュータユーザおよびソフトウェアユーザは、書く、計算する、組織化する、プレゼンテーションを準備する、電子メールを送受信する、作曲するなどをユーザが行うのを支援するユーザフレンドリなソフトウェアアプリケーション群が登場し、そのようなアプリケーション群に慣れてきている。例えば、最新の文書処理アプリケーションにより、ユーザは、様々な役立つドキュメントを作成し、編集することができるようになった。別の例として、最新のプロジェクト管理アプリケーションにより、ユーザは、様々なプロジェクトに関連する仕事、リソース、および工数（ｌａｂｏｒ）を組織化し、管理するためのプロジェクト管理スケジュールを作成することができるようになっている。

【０００３】

手動のプロジェクト管理システムおよびコンピュータ化されたプロジェクト管理システムは、管理者および計画者が、所与のプロジェクトを完了させるために要求される仕事、リソース、およびスケジュールを組織化し、計画することができるようにする。ほとんどのプロジェクトでは、いくつかの依存関係および制約により、プロジェクト全体のタイミングおよび完了、ならびにプロジェクト全体のを構成する下位工程のタイミングおよび完了が規定される。例えば、住宅建設プロジェクトでは、化粧壁下位工程（ｓｕｂ－ｐｒｏｊｅｃｔ）は、通常、電気工事が完了するまで開始されないようにすることができる。また、いくつかの下位工程は、利用可能な労働およびリソースによって制約される可能性がある。プロジェクト管理スケジュールを作成し、自動化するため、プロジェクト管理ソフトウェアアプリケーション群が開発されている。多くのそのようなシステムでは、プロジェクト全体のを構成する仕事または下位工程は、ガントチャートなどのスケジューリングチャートにおいて設定され、それはプロジェクト全体を構成する所与のマイルストーンおよび関連する仕事に関する開始日と終了日を示し、利用されるリソース、ならびにプロジェクトを構成するマイルストーンおよび仕事に関連する制約に関する情報を提供する。

【発明の開示】

【発明が解決しようとする課題】

【０００４】

現在、一部のプロジェクト管理ソフトウェアアプリケーションは、クライアント／サーバアーキテクチャを利用してプロジェクトを管理する。それらのアプリケーションの多くでは、サーバは、プロジェクト関連データの大部分を格納するのに使用される。つまり、プロジェクトデータのほとんどは、サーバ上に保持され、複数のユーザによりアクセスが可能となっている。サーバ上のデータのサイズを非常に大きくすることが可能な場合もある。通常、ユーザがプロジェクトを保存する際には、データの全てをサーバに送信するか、またはプロジェクトを読み込む際にサーバからデータの全てを読み出される。以上の事項は、サーバに大量のデータを保存しようと、またはサーバから大量のデータを取り出そうと試みる場合、大きな問題となる可能性がある。読み込み問題および待ち時間問題は、複数のクライアントがサーバに接続し、保存動作および読み込み動作を試みる場合、悪化する可能性がある。さらに、サーバから大量のデータを読み込む／保存することを試みる場合、これらの処理は極めて煩雑となり、効率が悪化し、および費用が増加する可能性がある。パフォーマンス問題は、クライアントが、低帯域幅のＷＡＮ接続を介してサーバに

接続した場合、特に深刻となる。

【課題を解決するための手段】

【０００５】

本発明の実施態様は、プロジェクト管理アプリケーションを使用してデータを高度に自動化して（ｉｎｔｅｌｌｉｇｅｎｔｌｙ）キャッシュし、同期をとることによってプロジェクトデータを管理する方法およびシステムを提供することにより、上述の、およびその他の問題を解決する。本発明の一実施態様は、プロジェクトデータを含むプロジェクトスケジュールを生成し、プロジェクトスケジュールに対する変更を探査し（ｍｏｎｉｔｏｒ）、および探査された変更に従って所定の閾値が満たされているかどうかを判定することにより、プロジェクト管理アプリケーションのデータを管理するための方法を提供する。本発明の別の実施態様は、プロジェクトに関連するプロジェクトデータを受信すること、受信されたプロジェクトデータを格納済みのプロジェクトデータと比較することにより、データの差分を算出すること、および算出された差分に基づき、プロジェクトの中のプロジェクトデータを更新することにより、プロジェクト管理アプリケーションのプロジェクトデータを管理するための方法を提供する。本発明のさらに別の実施態様は、プロジェクトデータを含むプロジェクトを要求し、プロジェクトに関連する第１のバージョンスタンプを送信し、および第１のバージョンスタンプに関連するデータと第２のバージョンスタンプに関連するデータの比較に基づいてプロジェクトデータを受信することにより、プロジェクト管理アプリケーションのプロジェクトデータを管理するための方法を提供する。

【０００６】

本発明を特徴付ける以上、およびその他の特徴および利点は、以下の詳細な説明を読み、関連する図面を詳しく見ることで明白となろう。以上の一般的な説明と以下の詳細な説明はともに、典型的であり、説明的であるに過ぎず、請求される本発明を限定するものではないことを理解されたい。

【発明を実施するための最良の形態】

【０００７】

以上簡単に説明したとおり、本発明の実施態様は、プロジェクト管理アプリケーションを使用する際に、プロジェクトデータをキャッシュするための方法およびシステムを目的とする。この方法およびシステムは、クライアントコンピュータのローカルキャッシュを利用して、プロジェクト管理アプリケーションを使用するプロジェクトデータを格納する。ユーザが、プロジェクトサーバ／データベースからデータを保存し、または読み込むことを所望する場合、本発明は、前述した読み込み問題および待ち時間問題を回避する可能性がある。さらに、本発明は、ユーザが、プロジェクトサーバ／データベースに必ずしも接続することなしに、プロジェクトデータの作成および／または編集を行うことができるようにする。

【０００８】

以下の詳細な説明では、本明細書の一部分を成す添付の図面を参照し、図面では、例示として、特定の実施形態または実施例が示されている。本発明の趣旨または範囲を逸脱することなく、それらの実施形態を組み合わせること、他の実施形態を利用すること、および構造上の変更を行うことができる。したがって、以下の詳細な説明は、限定する意味で解釈されるべきではなく、本発明の範囲は、特許請求の範囲およびその等価な範囲によって定義される。

【０００９】

動作環境

次に、いくつかの図のすべてで同様の符号が同様の要素を表している図面を参照して、本発明の態様、および典型的な動作環境を説明する。図１、および以下の説明は、本発明を実施することができる適切なコンピューティング環境の簡単な一般的説明を提供することを目的とする。本発明は、パーソナルコンピュータ上のオペレーティングシステム上で実行されるアプリケーションプログラムと連携して実行されるプログラムモジュール群の一般的な文脈で説明するが、本発明は、その他のプログラムモジュール群と組み合わせて

実施することもできることが当業者には認識されよう。

【００１０】

一般に、プログラムモジュールには、特定の仕事を実行する、または特定の抽象データ型を実装するルーチン、プログラム、コンポーネント、データ構造、およびその他のタイプの構造が含まれる。さらに、本発明は、ハンドヘルドデバイス、マルチプロセッサシステム、マイクロプロセッサベースの家庭用電化製品またはプログラマブル家庭用電化製品、ミニコンピュータ、メインフレームコンピュータなどを含め、他のコンピュータシステム構成を使用して実施してもよいことが当業者には理解されよう。また、本発明は、通信網を介してリンクされたリモート処理装置によって仕事が実行される分散コンピューティング環境において実施することもできる。分散コンピューティング環境では、プログラムモジュール群は、ローカルメモリ記憶装置とリモートメモリ記憶装置の両方の中に配置することが可能である。

【００１１】

次に、図１を参照して、本発明の様々な実施形態を実施するためのパーソナルコンピュータ１００に関する例示的なコンピュータアーキテクチャを説明する。図１に示すコンピュータアーキテクチャは、中央処理装置１０２（「ＣＰＵ」）、ランダムアクセスメモリ１０６（「ＲＡＭ」）および読み取り専用メモリ（「ＲＯＭ」）１０８を含むシステムメモリ１０４、ならびにメモリをＣＰＵ１０２に結合するシステムバス１１０を含む、慣用のパーソナルコンピュータを示す。起動時などに、コンピュータ内部の要素間で情報を転送するのを支援する基本ルーチンを含む基本入出力システムが、ＲＯＭ１０８の中に格納される。パーソナルコンピュータ１００は、オペレーティングシステム１１４、アプリケーションプログラム１１６のようなアプリケーションプログラム群、およびデータを格納するための大容量記憶装置１１２をさらに含む。

【００１２】

大容量記憶装置１１２は、バス１１０に接続された大容量ストレージコントローラ（図示せず）を介してＣＰＵ１０２に接続される。大容量記憶装置１１２、および関連するコンピュータ読取可能な媒体により、パーソナルコンピュータ１００のための不揮発性ストレージが提供される。本発明の実施形態によれば、大容量記憶装置１１２は、ローカルキャッシュまたはクライアントキャッシュ１１３を有するハードディスクを含む。前述したとおり、コンピュータ１００は、キャッシュ１１３への保存動作および／またはキャッシュ１１３からの読み込み動作を管理するためのキャッシュマネージャアプリケーション１１５も含む。本明細書に含まれるコンピュータ読取可能な媒体の説明は、ハードディスクドライブまたはＣＤ－ＲＯＭドライブなどの大容量記憶装置について述べるが、コンピュータ読取可能な媒体は、パーソナルコンピュータ１００がアクセスすることができる任意の利用可能な媒体とすることが可能であることが、当業者には理解されるはずである。

【００１３】

例として、限定としてではなく、コンピュータ読取可能な媒体は、コンピュータ記憶媒体および通信媒体を含むことが可能である。コンピュータ記憶媒体には、コンピュータ読取可能命令、データ構造、プログラムモジュール、またはその他のデータなどの情報を格納するために、任意の方法または技術で実装された揮発性媒体、および不揮発性媒体、リムーバブルな媒体およびリムーバブルでない媒体が含まれる。コンピュータ記憶媒体には、ＲＡＭ、ＲＯＭ、ＥＰＲＯＭ、ＥＥＰＲＯＭ、フラッシュメモリまたは他のソリッドステートメモリ技術、ＣＤ－ＲＯＭ、ＤＶＤ、または他の光ストレージ、磁気カセット、磁気テープ、磁気ディスクストレージまたは他の磁気記憶装置、あるいは所望の情報を格納するのに使用することができ、コンピュータがアクセスすることができる他の任意の媒体が含まれるが、以上には限定されない。

【００１４】

本発明の様々な実施形態によれば、パーソナルコンピュータ１００は、インターネットなどの、ＴＣＰ／ＩＰ網１１８または他のネットワークを介したリモートコンピュータ群への論理接続を使用する、ネットワーク化された環境において動作することができる。パ

ーソナルコンピュータ１００は、バス１１０に接続されたネットワークインタフェースユニット１２０を介して、ＴＣＰ／ＩＰ網１１８に接続することができる。ネットワークインタフェースユニット１２０は、ローカルネットワーク、ワイドエリアネットワーク、およびその他のネットワークなどの、他のタイプのネットワーク、およびリモートコンピュータシステムに接続するのにも利用できることを理解されたい。パーソナルコンピュータ１００は、キーボードまたはマウス（図示せず）を含め、いくつかのデバイスからの入力を受け取り、処理するための入出力コントローラ１２２も含むことが可能である。同様に、入出力コントローラ１２２は、ディスプレイスクリーン、プリンタ、または他の種類の出力デバイスに出力を与えることもできる。

【００１５】

簡単に前述したとおり、ワシントン州レッドモンドのマイクロソフトコーポレーションからのＷＩＮＤＯＷＳ（登録商標）オペレーティングシステムなどの、ネットワーク化されたパーソナルコンピュータの動作を制御するのに適したオペレーティングシステム１１４を含め、いくつかのプログラムモジュール、およびデータファイルを、パーソナルコンピュータ１００の大容量記憶装置１１２およびＲＡＭ１０６の中に格納することが可能である。大容量記憶装置１１２およびＲＡＭ１０６は、１つまたは複数のアプリケーションプログラムも格納することができる。詳細には、大容量記憶装置１１２およびＲＡＭ１１６は、様々な機能をユーザに提供するためのアプリケーションプログラム１１６を格納することができる。例えば、アプリケーションプログラム１１６は、スプレッドシートアプリケーションプログラム、電子メールアプリケーションプログラム１２４、ワードプロセッシングアプリケーションプログラム１２５、データベースアプリケーションプログラムなどの、多くの種類のプログラムを含むことができる。本発明の実施形態によれば、プロジェクト管理アプリケーションプログラム１２６が、本明細書で説明するプロジェクト管理スケジュールを準備するために備えられる。本発明に従って使用するための典型的なプロジェクト管理アプリケーションは、マイクロソフトコーポレーションによって製造される、Ｐｒｏｊｅｃｔ　ＰｒｏｆｅｓｓｉｏｎａｌおよびＰｒｏｊｅｃｔ　Ｓｅｒｖｅｒである。

【００１６】

次に、図２を参照すると、分散コンピューティング環境２００の簡略ブロック図が示されている。分散コンピューティング環境２００は、親プロジェクトの所有者／管理者、および１つまたは複数の子である下位工程の所有者／管理者が、本発明の実施形態に従ってプロジェクトスケジュールを作成し、保持するために、Ｗｅｂサービスインタフェース２１５を使用してプロジェクトサーバ／データベース２１０と通信することができるようになっている。図２に示した分散コンピューティング環境２００は、本発明の実施形態のための典型的な動作環境として動作する。本明細書で説明するとおり、プロジェクト全体のスケジュールは、親プロジェクトスケジュール、および１つまたは複数の階層関係にある子である下位工程から成るようにすることが可能である。

【００１７】

プロジェクトサーバ／データベース２１０は、図１を参照して例示し、説明したコンピュータ１００に関し前述したとおり、オペレーティングシステム、プロセッサ、およびメモリ空間を含むコンポーネントを有する汎用コンピューティングシステムである。プロジェクトサーバ／データベース２１０は、全体的／親プロジェクトスケジュール、および本明細書で説明する個々の子の下位工程スケジュールの管理を構築し、表示し、可能にするための十分なコンピュータ実行可能命令を有するプロジェクト管理アプリケーション１２６をも備える。

【００１８】

プロジェクトサーバ／データベース２１０は、親プロジェクトスケジュールに関連するデータ、および個々の子である下位工程スケジュールの各々に関するデータを保持するためのバックエンドデータベースまたはリレーショナルデータベースをも含む。当業者には理解されるとおり、リレーショナルデータベースにより、本明細書で説明するプロジェク

ト全体のスケジュール、つまり親プロジェクトスケジュールを構成する異種のプロジェクトスケジュール間で格納済みのデータを交換するため、１つのプロジェクトスケジュールに関連して格納された情報が、１つまたは複数の他のプロジェクトスケジュールに関係付けることが可能となる。

【００１９】

図２に示すとおり、親／マスタコンピュータシステム２２０は、インターネットまたはイントラネット２５０などの分散コンピューティングシステムを介して、プロジェクトサーバ／データベース２１０と通信を行う。１つまたは複数の子／下位工程コンピューティングシステム２３０、２４０、．．．ｎが、インターネットまたはイントラネット２５０を介してプロジェクトサーバ／データベース２１０と通信するために示される。理解されるとおり、親プロジェクトスケジュールの所有者／管理者、および親プロジェクトスケジュールを構成する１つまたは複数の子である下位工程スケジュールの所有者／管理者によって操作されるコンピューティングシステム２２０、２３０、２４０、．．．ｎは、図１に示すコンピュータ１００に関連して前述した処理機能およびメモリ機能を含むことができる。親および子の各コンピュータは、システムの「ピア」と呼ぶことができる。

【００２０】

個々のコンピューティングシステム２２０、２３０、２４０、．．．ｎの各々は、図１を参照して前述し、プロジェクトサーバ／データベース２１０において動作する、対応するプロジェクト管理アプリケーション１２６に対するクライアント側プロジェクト管理アプリケーションの役割を持つ、プロジェクト管理アプリケーション１２６も含むことができる。本発明の実施形態によれば、プロジェクトサーバ／データベース２１０は、親プロジェクトスケジュールの所有者／管理者、および子である下位工程スケジュールの所有者／管理者の間で自動的プロジェクトスケジュール変更を送信するため、電子メールアプリケーション１２４をも含み、またはアプリケーション１２４にアクセスするように動作することが可能である。

【００２１】

親プロジェクトスケジュールの所有者／管理者は、プロジェクトサーバ／データベース２１０に対して、インターネットまたはイントラネット２５０を介して、コンピューティングシステム２２０からプロジェクトスケジュールを作成することができる。個々の子である下位工程スケジュールの作成および保持も同様に、プロジェクトサーバ／データベース２１０に対して、インターネット／イントラネット２５０を介して、コンピューティングシステム２３０、２４０、．．．ｎから、子である下位工程の所有者／管理者によって作成することが可能である。親プロジェクトスケジュールの所有者／管理者、および１つまたは複数の子である下位工程スケジュールの所有者／管理者の間における通知は、プロジェクト管理アプリケーション１２６によるコマンドが行われると、電子メールアプリケーション１２４などの通信アプリケーションによって自動的に実行することが可能である。つまり、変更、または提案される変更により、影響を受けるプロジェクトスケジュール、または関連する下位工程スケジュールの所有者／管理者への通知が要求されると、プロジェクトサーバ／データベース２１０におけるプロジェクト管理アプリケーション１２６が、電子メールアプリケーション１２４のような通信アプリケーションを呼び出して、適切な通知メッセージが、影響を受ける当事者に送信されるようにすることができる。当業者には理解されるとおり、インスタントメッセージングシステム、電話システムなどを含め、他の形態の通信も、影響を受ける当事者への通知のために利用することができる。

【００２２】

プロジェクト管理方法およびプロジェクト管理システムの分野の当業者には認識されるとおり、所与のプロジェクトは、完了されなければならないいくつかの仕事を含むことが可能であり、各仕事は、利用可能なリソース、および仕事を完了させるのに要求される期間によって制約させることができる。さらに、一部の仕事は、関連する仕事が完了するまで、開始されないようにすることができる。多くのプロジェクトの場合、所与のプロジェクトを構成する個々の仕事は、別個の関連するプロジェクトスケジュールの下で管理され

、処理される。例えば、住宅を建設するためにマスタプロジェクトスケジュールを準備することが可能である。マスタプロジェクトスケジュールは、全体的な骨組（ｇｅｎｅｒａｌ　ｆｒａｍｉｎｇ）の段階または仕事、電気の段階または仕事、および配管の段階または仕事などの、３つの段階または仕事を含むことが可能である。マスタプロジェクトの段階または仕事の各々は、所与の段階または仕事に関する自身のプロジェクトスケジュールを構築し、管理する下請け業者によって完了させることが可能である。例えば、第１の下位工程スケジュールは、全体的な骨組のために構築することが可能であり、第２の下位工程スケジュールは、電気工事のために構築することが可能であり、第３の下位工程スケジュールは、配管工事のために構築することが可能である。理解されるとおり、下位工程スケジュールの各々は、１つまたは複数の段階または仕事に細分することが可能であり、１つまたは複数の段階または仕事は、所与の下位工程内の１つまたは複数の仕事を完了させるためのさらなる下位工程に関連させることが可能である。

【００２３】

プロジェクト管理データの保存／読み込み

本発明の様々な態様によれば、プロジェクト管理データは、クライアントコンピューティングシステム１００の大容量記憶装置１１２のハードディスクキャッシュなどの、ローカルキャッシュ１１３に読み込まれ、保存することが可能である。前述した読み込み問題および待ち時間問題の多くは、プロジェクト管理データをクライアントキャッシュ１１３に高度に自動化して保存し、読み込むめば、回避することができる。本発明は、読み込み時間中、および保存時間中にプロジェクトサーバ／データベース２１０からダウンロードされ、またはプロジェクトサーバ／データベース２１０に送信されるデータのサイズを縮小するための機構を提供する。

【００２４】

本発明の一態様によれば、トリガされると、キャッシュマネージャ１１５は、プロジェクトサーバ／データベース２１０のバックエンドデータベースにデータを保存し、およびこのバックエンドデータベースからデータを読み込むことを高度に自動化して行うＷｅｂサービスインタフェース２１５と通信する。以下に説明するとおり、キャッシュマネージャ１１５は、プロジェクトサーバ／データベース２１０とも対話しながら、プロジェクトデータを格納し、操作し、および管理するようにすることが可能である。また、キャッシュマネージャ１１５は、管理者（ａｄｍｉｎｉｓｔｒａｔｏｒ）がキャッシュ１１３を操作することを可能にするインタフェースも提供する。

【００２５】

プロジェクトサーバ／データベース２１０は、プロジェクト管理者、チームメンバ、ならびにプロジェクトに積極的に関与し、またはプロジェクトによって権益が影響を受ける可能性がある様々な個人または組織の間において、オンライン共同作業を可能にする。また、プロジェクトサーバ２１０は、組織が、複数のプロジェクトにわたって標準を共有し、チェックイン機能とチェックアウト機能とによりプロジェクトを安全に保護するのを支援し、複数のプロジェクトにわたってリソースの利用可能性、およびその他のリソース情報を閲覧し、並びにプロジェクトのポートフォリオ（ｐｏｒｔｆｏｌｉｏ）を管理してポートフォリオについて報告を行うこともできるようにする。

【００２６】

親／マスタコンピュータ２２０に基づき、プロジェクト管理者は、プロジェクトサーバ／データベース２１０を使用して、多くの作業を実行することができる。すなわち、１）仕事をチームメンバに割り当て、完了した作業を常に把握しておき、チームメンバから仕事の更新を受け入れ、更新された情報をプロジェクトに組み込むことを自動的に、または手動で行うこと、２）詳細のステータスレポートを要求し、受領して、個々のステータスレポートを１つのプロジェクトステータスレポートにまとめて、他の人々に提示すること、３）ｗｈａｔ－ｉｆ分析を実行して、プロジェクトの異なるバージョンを作成した後、費用、スケジュール、およびリソース割り当てを比較すること、４）リソースの利用可能性、リソース割り当て、および費用に対するスケジュール変更の影響を評価すること、お

よび／または５）組織全体にわたるリソース利用可能性を閲覧し、チームを構成して、一般的なリソースを仕事に割り当てるようにし、技能セットに基づいてリソースを探し出し、置き換え、企業リソースの共通リストからチームを構成するといった作業である。

【００２７】

コンピュータ２３０のような子／下位コンピュータから、様々なチームメンバは、マイクロソフトコーポレーションによって製造されるＰｒｏｊｅｃｔ　Ｗｅｂ　Ａｃｃｅｓｓなどの、Ｗｅｂサービスインタフェース２１５を使用することにより、プロジェクトサーバ／データベース２１０上の情報にアクセスして、１）仕事割り当てを閲覧し、更新して、一部の割り当てについてプロジェクト管理者に応答し、完了した作業について一定の間隔で更新すること、２）新たな仕事を作成し、その仕事をプロジェクト管理者に、承認、およびプロジェクト計画への組み込みのために送信すること、３）仕事を他のチームメンバに委託すること、４）仕事の期間、および他の仕事に対する関係をグラフで示すためのガントチャート形式で仕事を閲覧すること、５）グループ化され、並べ替えられ、フィルタ処理された仕事を閲覧すること、および／または６）割り当てられた仕事だけでなく、プロジェクト全体に関する最新情報を閲覧することができる。他のプロジェクト管理者または上級管理者も、Ｗｅｂサービスインタフェース２１５を使用してプロジェクトサーバ／データベース２１０上の情報にアクセスし、プロジェクト情報、仕事情報、およびリソース情報を点検して、プロジェクトが、個々に、または組織全体にわたってどのように進んでいるかを確認することができる。

【００２８】

次に、図３を参照すると、本発明の一実施形態によるフローチャートが示されている。前述したとおり、プロジェクト管理者が、プロジェクト管理アプリケーション１２６を利用して、例えば、新たな建物建設プロジェクトなどのプロジェクトを作成し、または変更する（ブロック３０２）。プロジェクト管理アプリケーション１２６は、ユーザ入力を受け取り、これに応じてプロジェクト関連データを調整するように動作する。プロジェクト管理アプリケーション１２６が、自動保存機能によってプロジェクト関連データを自動的に保存する動作能力（ｏｐｅｒａｂｉｌｉｔｙ）を備えるか（ブロック３０４）、あるいはユーザが、指定されたファンクションキーを使用し、またはプロジェクトメニューとインタフェースをとって（すなわち、標準の「ファイル」ボタンおよび「保存」ボタンを使用して）、プロジェクト関連データを手動で保存することができる（ブロック３０６）。

【００２９】

本発明によれば、プロジェクト関連データは、非同期の形でプロジェクトサーバ／データベース２１０に保存される。つまり、プロジェクト関連データは、ユーザが保存ボタンを押した場合にではなく、所定の閾値（以下に説明する）が満たされた場合に、プロジェクトサーバ／データベース２１０に保存される。プロジェクト関連データは、１）プロジェクト、２）仕事、３）リソース、４）仕事の間の制約、５）リソースに対する仕事の割り当て、６）プロジェクトビュー（ｖｉｅｗ）およびリソースビュー、７）カレンダー労働時間および労働日、ならびに非労働時間および労働日を定義する、および／または８）ルックアップテーブル、すなわちユーザが定義できる列に関する許容できる値を指定する、ユーザが定義したテーブルを含むことができる。

【００３０】

プロジェクト管理者などのユーザが、コンピューティングシステム１００のようなクライアントコンピュータ上で初めてプロジェクトを作成すると、プロジェクト関連データが、キャッシュ１１３の中に格納される。プロジェクトが作成された後、ユーザが、プロジェクトデータを追加し、更新し、または削除するなどにより、プロジェクトを変更した場合にも、プロジェクト関連データは、キャッシュ１１３の中に格納される。しかし、本発明の実施形態によれば、特定のプロジェクトは、そのプロジェクトを変更する前に、ユーザによって「チェックアウト」されなければならない。この特定のプロジェクトは、別のユーザが変更することができるようにするためにはまず、再チェックインしなければならない。このチェックイン／チェックアウトプロセスにより、プロジェクトの無許可の変更

が防止される。

【００３１】

前述したとおり、動作環境は、複数のユーザが、インターネットを介してプロジェクトサーバ／データベース２１０を使用してプロジェクトにアクセスすることを可能にする。以下に記述するとおり、プロジェクト情報は、プロジェクトサーバ／データベース２１０に定期的に送信される。より詳細には、「新たな」情報だけが、ある閾値が満たされると、プロジェクトサーバ／データベース２１０に送信される。この「新たな」情報は、データの「差分」または「デルタ」と呼ばれる。さらに、クライアントが、プロジェクトサーバ／データベース２１０からプロジェクトにアクセスしようと試みると、「新たな」情報だけが、クライアントに転送されて、プロジェクトが更新される。読み込み／待ち時間の問題は、プロジェクトサーバ／データベース２１０からプロジェクトの一部分だけを読み込むことにより、回避することができる可能性がある。

【００３２】

本発明の一実施形態によれば、キャッシュ１１３は、クライアントコンピュータ１００によって読み込まれ、または保存されたすべてのファイルについてのメタデータ情報を保持する。メタデータ情報は、ローカルファイルパス、プロジェクトサーバ／データベース２１０との前回の同期時刻、様々なバージョンスタンプ、および／または各プロジェクトに関するその他の情報を含むが、これらには限定されない。同期時刻とは、クライアントコンピュータとプロジェクトサーバ／データベース２１０が対話して、特定のプロジェクトファイルを更新した時刻のことである。クライアントコンピュータ１００がプロジェクトサーバ／データベース２１０と通信すると毎回、バージョンスタンプが、好ましくは、クライアントキャッシュ１１３の中に格納される。

【００３３】

本発明の一実施形態によれば、バージョンスタンプは、サーバ上の現在時刻を表わすことが可能な８バイト数、または自動的に増分される８バイト数とすることができる。バージョンスタンプは、常にプロジェクトサーバ／データベース２１０によって割り当てられるのが望ましい。プロジェクトサーバ／データベース２１０は、新たなデータ、または更新されたデータをクライアントから受信すると、バージョンスタンプ（プロジェクトサーバ／データベース２１０における現在時刻、または更新されているプロジェクトに関連付けられた前のバージョンスタンプに基づく、自動的に増分される数）を生成し、そのバージョンスタンプをプロジェクトに関連付ける。プロジェクトサーバ／データベース２１０はまた、新たなバージョンスタンプをクライアントに戻す。次回にクライアントが、更新されたデータを受信することを所望する際、クライアントは、そのバージョンスタンプをプロジェクトサーバ／データベース２１０に送信し、プロジェクトサーバ／データベース２１０が、クライアントのバージョンスタンプより新しいデータを返す。クライアントは、バージョンスタンプを不透明な（ｏｐａｑｕｅ）８バイト数として扱う。バージョンスタンプのためにより多いバイト、またはより少ないバイトを使用してもよいことが理解されよう。自動、または手動の増分の保存のたびに、クライアントコンピュータは、メタデータ、およびその他の情報を使用して、特定のプロジェクトに関連する変更されたデータプロパティを常に把握しておく。キャッシュマネージャ１１５が、保存の合間に変更されたすべてのプロパティを蓄積して、それらをキャッシュファイル１１３の中に格納する。

【００３４】

本発明の一実施形態によれば、キャッシュマネージャ１１５は、変更されたプロパティを３つの別個のカテゴリに編成する。すなわち、本発明の一実施形態によれば、新たなデータ、更新されたデータ、および削除されたデータである。次に、キャッシュマネージャ１１５が、変更されたプロパティを各カテゴリに従ってキャッシュ１１３の中に格納する。

【００３５】

ブロック３０８で、ある閾値に達すると、キャッシュマネージャ１１５は、プロジェクトサーバ／データベース２１０上に格納された特定のプロジェクトファイルを更新するた

めに、プロジェクトサーバ／データベース２１０に送信されるべきデータの差分を準備する。ユーザがプロジェクトデータを編集した際、プロジェクト管理アプリケーション１２６は、追加され、変更され、または更新されたデータを常に把握しておく。本発明の一実施形態によれば、キャッシュ１１３が合計プロジェクトサイズの少なくともおよそ１０パーセントという閾値を満たすデータ更新をする場合、キャッシュマネージャ１１５は、プロジェクトサーバ／データベース２１０上に格納された特定のプロジェクトファイルを更新するため、プロジェクトサーバ／データベース２１０に送信されるべきデータの差分またはデルタを準備する。本発明の代替の実施形態によれば、プロジェクト仕事のあるパーセンテージ、例えば３０パーセントが変更された場合、キャッシュマネージャ１１５は、プロジェクトサーバ／データベース２１０上に格納された特定のプロジェクトファイルを更新するため、プロジェクトサーバ／データベース２１０に送信されるべきデータのデルタを準備する。

【００３６】

所定の閾値に達した場合、ブロック３１０で、クライアントコンピュータ１００がプロジェクトサーバ／データベース２１０に接続されているかどうかが判定される。閾値に達していない場合、ブロック３１４で、フローはブロック３０２に戻り、ユーザは、プロジェクトを変更および／または閲覧することを続けるか、または異なるプロジェクトを開くことができる。クライアントコンピュータ１００がプロジェクトサーバ／データベース２１０に接続されている場合、ブロック３１２で、データは、プロジェクトサーバ／データベース２１０に送信される。クライアントコンピュータ１００がプロジェクトサーバ／データベース２１０に接続されていない場合、ブロック３１４で、フローはブロック３０２に戻る。クライアントは、閾値を確認する前にプロジェクトサーバ／データベースに接続されることが可能であることが理解されよう。

【００３７】

プロジェクトサーバ／データベース２１０のバックエンドデータベースは、プロジェクトエンティティ、すなわちプロジェクト、リソース、仕事、カレンダなどのそれぞれに対応する、いくつかの１次テーブルを含む。テーブルの中の各行は、作成時刻に対応するバージョンスタンプ、および／または最後の変更時刻に対応するバージョンスタンプを含むことができる。データベースは、各１次テーブルに対する、対応する「シャドー」テーブルも含む。例えば、プロジェクト、仕事、リソース、カレンダなどに対応するテーブルに対するシャドーテーブルが存在する。１次テーブルからある行が削除されると、その行のキーが、削除の時刻とともに、対応するシャドーテーブルの中に記録される。「キー」は、テーブル内のレコードを一意に識別するテーブル内の列を表すようにすることができる。本発明の一実施形態によれば、グローバル一意識別子（ＧＵＩＤ）の組み合わせを、キーとして使用することが可能である。例えば、ＧＵＩＤにより一意にプロジェクトを識別することができる。プロジェクトの中の仕事も同様に、その仕事独自のＧＵＩＤ、およびその仕事が属するプロジェクトのＧＵＩＤで一意に識別することが可能であり、以下同様である。

【００３８】

本明細書で説明するとおり、クライアントコンピュータからプロジェクトサーバ／データベース２１０に送信されるデータには、新たなデータ、変更されたデータ、および削除されたデータが含まれる。サーバは、サーバにおける現在時刻に基づくか、または現在のバージョンスタンプを増分することにより、プロジェクトに対する新たなバージョンスタンプを決める。新たなデータの場合、新たな行が、対応するプロジェクトサーバ／データベース２１０テーブルの中で、作成のバージョン、および／または最後の変更のバージョンとして新たなバージョンスタンプ（すなわち、関連するバージョンスタンプ）を伴って作成される。更新されたデータの場合、対応するプロジェクトサーバ／データベース２１０テーブルの中のデータが、対応する新たなバージョンスタンプで更新される。更新された行は、データが最後に変更された際のバージョンとして新たなバージョンスタンプを含む。削除されたデータについて、対応するサーバ／データベース２１０テーブルから行が

削除される。削除された各行について、対応するシャドーテーブルに行が追加され、新たなバージョンスタンプ、および対応する１次行のキーが記入される。

【００３９】

デルタ計算が、プロジェクトサーバ／データベース２１０およびクライアントにおいてどのように行われるかの実施例として、次のシナリオを考慮されたい。この実施例に関して、「プロジェクトＩＤ」、「プロジェクト名」、および「バージョンスタンプ」という列を含む「Ｐｒｏｊｅｃｔ」というテーブルを想定されたい。「プロジェクトＩＤ」列は、「Ｐｒｏｊｅｃｔ」テーブルに対するキーである。「Ｔａｓｋ」という別のテーブルが、「プロジェクトＩＤ」、「仕事ＩＤ」、「仕事名」、「期間」、および「バージョンスタンプ」という列を含む。「プロジェクトＩＤ」および「仕事ＩＤ」という列は、Ｔａｓｋというテーブルに対するキーである。「ＴａｓｋＳｈａｄｏｗ」という別のテーブルが、削除された仕事の記録をとるのに使用される。「ＴａｓｋＳｈａｄｏｗ」テーブルは、「プロジェクトＩＤ」、「仕事ＩＤ」、および「バージョンスタンプ」という列を有する。以上のテーブルは、最初、空であるものと想定する。

【００４０】

上記の実施例で、「Ｕｓｅｒ１」というユーザは、３つの仕事、２日間の期間の仕事Ｔ１、３日間の期間の仕事Ｔ２、および５日間の期間の仕事Ｔ３を有する新たなプロジェクトＰ１を作成するものと想定する。ユーザは、マイクロソフトコーポレーションによって製造されたＰｒｏｊｅｃｔ　Ｐｒｏｆｅｓｓｉｏｎａｌなど、プロジェクト管理アプリケーション１２６を使用してプロジェクトを作成する。Ｕｓｅｒ１がプロジェクトを保存することを選択すると、キャッシュマネージャ１１５は、プロジェクトサーバ／データベース２１０に送信するのに十分なデータをマネージャ１１５が有していると判定するものと想定する。このプロジェクトは、新たなプロジェクトであるため、すべてのデータが、「新たなデータ」バケット（ｂｕｃｋｅｔ）に入れられ、プロジェクトサーバ／データベース２１０に送信される。プロジェクトサーバ／データベース２１０は、その新たなデータをプロジェクトサーバ／データベース２１０データベーステーブル群の中に挿入し、テーブル群は、プロジェクトサーバ／データベース２１０がバックエンドデータベースの更新を終了すると、以下のデータを有することとなる。すなわち、

【００４１】

【表１】

PROJECT		
プロジェクトID	プロジェクト名	バージョンスタンプ
1	P1	1

【００４２】

【表２】

TASKS				
プロジェクトID	仕事ID	仕事名	期間	バージョンスタンプ
1	1	T1	2	1
1	2	T2	3	1
1	3	T3	5	1

【００４３】

【表３】

TASKSHADOW		
プロジェクトID	仕事ID	バージョンスタンプ

【００４４】

プロジェクトサーバ／データベース２１０は、「１」という現在のバージョンスタンプ値をクライアントに戻す。キャッシュマネージャ１１５は、マネージャ１１５が、Ｕｓｅｒ１に関してマネージャ１１５のキャッシュ１１３の中にプロジェクトＰ１のバージョン「１」を有することを記録する。次いでユーザ１はプロジェクトＰ１においてチェックする。

【００４５】

次に、Ｕｓｅｒ２という別のユーザが、プロジェクトＰ１を開こうと試みたものと想定されたい。キャッシュマネージャ１１５は、マネージャ１１５が、Ｕｓｅｒ２に関して、マネージャ１１５のキャッシュ１１３の中にプロジェクトＰ１を有さないと判定するため、０というバージョンスタンプとともに、プロジェクトＰ１データを求める要求をプロジェクトサーバ／データベース２１０に送信する。この時点で、プロジェクトサーバ／データベース２１０は、０より大きいバージョンスタンプを有するすべてのレコードを上記の２つのテーブルから取り出す。すべての行は、０より大きいバージョンスタンプを有するので、すべての行が取り出される。次に、プロジェクトサーバ／データベース２１０は、プロジェクトＰ１に関する行、ならびに仕事Ｔ１、Ｔ２、およびＴ３に関する行を、「１」という現在のバージョンスタンプ値とともに、削除されたデータに関する行は全くなしに、クライアントに戻す。クライアントがデータを受信すると、キャッシュマネージャ１１５が、そのデータをキャッシュ１１３に加え、Ｕｓｅｒ２に関して、マネージャ１１５のキャッシュの中に、バージョンスタンプ１を有するプロジェクトＰ１を有することを記録する。

【００４６】

次に、Ｕｓｅｒ２が、仕事Ｔ２を削除し、仕事Ｔ３の期間を４日間に変更し、３日間の期間を有する新たな仕事Ｔ４を追加することを決めたものと想定する。プロジェクト管理アプリケーション１２６は、ユーザ活動を追跡して、ある仕事が削除され、別の仕事が変更され、新たな仕事が作成されたと判定する。Ｕｓｅｒ２が、プロジェクトを保存してチェックインすることを決めると、キャッシュマネージャ１１５は、仕事Ｔ２を削除されるデータのバケットに入れ、仕事Ｔ３を更新されるデータのバケットに入れ、仕事Ｔ４を新たなデータのバケットに入れる。次に、キャッシュマネージャ１１５は、それらのバケットをプロジェクトサーバ／データベース２１０に送信する。プロジェクトサーバ／データベース２１０は、データを受信すると、削除されたデータのバケットを適用して、Ｔａｓｋｓテーブルから仕事Ｔ２を削除し、仕事Ｔ２をＴａｓｋＳｈａｄｏｗテーブルに追加する。また、プロジェクトサーバ／データベース２１０は、仕事Ｔ２を更新し、新たな仕事Ｔ４をＴａｓｋｓテーブルの中に挿入する。以上のすべての変更は、「２」という次に高いバージョンスタンプを獲得する。したがって、変更が行われた後、テーブルは、以下のデータを有する。すなわち、

【００４７】

【表４】

PROJECT		
プロジェクトID	プロジェクト名	バージョンスタンプ
1	P1	1

【００４８】

【表５】

TASKS				
プロジェクトID	仕事ID	仕事名	期間	バージョンスタンプ
1	1	T1	2	1
1	3	T3	4	2
1	4	T4	3	2

【００４９】

【表６】

TASKSHADOW		
プロジェクトID	仕事ID	バージョンスタンプ
1	2	2

【００５０】

次に、プロジェクトサーバ／データベース２１０は、「２」という最新のバージョンスタンプ値をクライアントに戻す。キャッシュマネージャ１１５は、マネージャ１１５が、Ｕｓｅｒ２に関して、マネージャ１１５のキャッシュ１１３の中にプロジェクトＰ１のバージョン「２」を有すると記録する。

【００５１】

上記の実施例を続け、次にＵｓｅｒ１が、プロジェクトＰ１を再び開こうと試みたものと想定する。その時点で、キャッシュマネージャ１１５は、マネージャ１１５が、Ｕｓｅｒ１に関して、マネージャ１１５のキャッシュ１１３の中にプロジェクトＰ１のバージョン「１」を有すると判定し、バージョン「１」以来のプロジェクト「Ｐ１」に対する更新を求める要求をプロジェクトサーバ／データベース２１０に送信する。プロジェクトサーバ／データベース２１０は、サーバ／データベース２１０が、１つの削除された仕事のレコード（２という仕事ＩＤ値を有する仕事Ｔ２）、および「１」より大きいバージョン値を有する２つの新たな、または更新された仕事のレコード（仕事Ｔ３およびＴ４）を有すると判定する。プロジェクトサーバ／データベース２１０は、仕事Ｔ２（２という仕事ＩＤを有する）を削除されたデータのバケットに入れ、仕事Ｔ３およびＴ４を新たなデータのバケットに入れて、「２」という現在のバージョンスタンプ値とともにクライアントに送信する。キャッシュマネージャ１１５は、プロジェクトのキャッシュされたコピーから仕事Ｔ２を削除し、マネージャ１１５が、仕事Ｔ３のより古いバージョンを既に有すると判定し、そのバージョンをプロジェクトサーバ／データベース２１０からのデータで更新し、プロジェクトサーバ／データベース２１０からのデータに基づいて新たな仕事Ｔ４を作成する。キャッシュマネージャ１１５は、次に、マネージャ１１５が、Ｕｓｅｒ１に関して、マネージャ１１５のキャッシュ１１３の中にプロジェクトＰ１のバージョン「２」を有することを記録する。この時点で、Ｕｓｅｒ１に関するキャッシュ１１３は、プロジェクトＰ１のサーババージョンと完全に同期がとられる。

【００５２】

前述したとおり、クライアントコンピュータが、データをプロジェクトサーバ／データベース２１０に送信した後、プロジェクトサーバ／データベース２１０は、現在のバージョンスタンプを、プロジェクトサーバ／データベース２１０上の特定のプロジェクトの変更時刻として、クライアントコンピュータに送信する。クライアントコンピュータ１１３のキャッシュマネージャ１１５は、バージョンスタンプを、そのプロジェクトに関する最新の同期時刻としてキャッシュ１１３の中に記録する。

【００５３】

このため、キャッシュ１１３は、プロジェクトデータプロパティに対するあらゆる変更を保持するが、変更されたデータは、閾値に達し、プロジェクトサーバ／データベース２

１０に対する接続の準備が整うまで、プロジェクトサーバ／データベース２１０側で更新されない。したがって、接続の準備が整うと、特定のプロジェクトのデータが、プロジェクトサーバ／データベース２１０からクライアントコンピュータ上のプロジェクトアプリケーションに変更されたデータを転送することによって更新されるが、これを本明細書で「同期」と呼ぶ。これは、例えば、プロジェクト管理者が、自分のコンピュータ上でプロジェクトファイルを開き、プロジェクト管理者の知らないうちに、前のプロジェクト管理者がそのプロジェクトを変更している場合、特に重要である。同期ステップにより、プロジェクト管理者は、確実に最新のプロジェクトバージョンを有することとなる。

【００５４】

閾値に達することによってトリガされると、前述したとおり、クライアントコンピュータ１００からプロジェクトサーバ／データベース２１０に送信されるデータは、プロジェクトサーバ／データベース２１０の中に存在する最新コピーと、キャッシュ１１３の中に存在するコピーとの差分となる。同様に、プロジェクトサーバ／データベース２１０からコンピューティングシステム１００に読み込まれるデータも、キャッシュ１１３の中の最新コピーとプロジェクトサーバ／データベース２１０の中に存在するコピーの差分である。データの差分またはデルタだけを送信し、またはダウンロードすることにより、本発明は、コンピューティングシステム１００とプロジェクトサーバ／データベース２１０との間で処理される（ｔｒａｎｓａｃｔｅｄ）データの量を削減し、効率およびパフォーマンスを向上させる可能性がある。

【００５５】

前述したとおり、クライアントコンピュータからプロジェクトサーバ／データベース２１０にデータを保存する際、Ｗｅｂサービスインタフェース２１５が、保存されるべきデータをキュー（ｑｕｅｕｅ）に入れることにより、データベースに非同期で保存する。キャッシュマネージャ１１５は、データをプロジェクトサーバ／データベース２１０のバックエンドデータベースに保存するＷｅｂサービスインタフェース２１５と通信を行う。低い優先度のスレッドが、キューを処理し、データベースの中のデータを更新する。クライアントは、次回の保存を行う際、保存のステータスを求めるポーリングをプロジェクトサーバ／データベース２１０に対して行う。保存が何らかの理由で失敗した場合、クライアントは、すべてのデータをプロジェクトサーバ／データベース２１０に再送信して保存する。ユーザがアプリケーションを閉じているが、保存されるべきさらなるデータが存在する場合、クライアントは、保留中の保存が存在すること記録に残し（ｎｏｔｅ）、次回にユーザがオンラインになった際に、保存を再試行する。

【００５６】

ユーザが、プロジェクトサーバ／データベース２１０からプロジェクトを読み込むことを所望すると、キャッシュマネージャ１１５は、プロジェクトＩＤ、ならびにプロジェクトが前回にプロジェクトサーバ／データベース２１０と同期をとった時点を示すバージョンスタンプを伴う要求をプロジェクトサーバ／データベース２１０に送信する。クライアントコンピュータは、プロジェクトに関してデータが一度も同期がとられていない場合、「０」を送信する。読み込み動作中、データは、次の２つのカテゴリまたはビン（ｂｉｎ）に分けることが可能である。すなわち、１）プロジェクト管理アプリケーションがユーザに提供することが絶対に必要なデータ、および２）ユーザが何か別のことを行っている際に、別個のスレッド上でダウンロードすることが可能なデータである。

【００５７】

ダウンロード中、Ｗｅｂサービスインタフェース２１５は、プロジェクトデータに関してプロジェクトサーバ／データベース２１０のデータベースにクエリを行い、データのデルタをクライアントコンピュータにストリーミングする。キャッシュマネージャ１１５は、プロジェクトサーバ／データベース２１０によって受信されたデルタをキャッシュ１１３の中のプロジェクコピーに適用して、ファイル全体をダウンロードせずに、最新のデータをユーザに提供する。データを２つのカテゴリに分けることにより、ユーザは、プロジェクトサーバ／データベース２１０から大量のデータがダウンロードされるのを待たずに

、より良好な体験をする可能性がある。

【００５８】

可能な限り早くユーザに操作させるため、可能な限りダウンロードは少なくし、後に、残りのデータを背景スレッドでダウンロードすることが好ましい。エンティティ（仕事、リソースなど）の各々は、多くの属性を有する。それらの属性（プロパティ）の一部は、プロジェクトの文脈でデータの意味を理解するのにプロジェクト管理アプリケーション１２６が要する。例えば、仕事はＩＤ、開始日、期間のようなプロパティを有し、これらはプロジェクト管理アプリケーション１２６が、プロジェクト全体の文脈でその仕事の意味を理解するのに要求される可能性があるものである。仕事名、仕事の親、仕事が属するグループなどのその他のプロパティは、背景スレッドでダウンロードし、またはユーザがそのプロパティにアクセスしようと試みた際にダウンロードすることが可能である。好ましくは、プロジェクト管理アプリケーション１２６は、アプリケーション１２６が表示している行に関するコアデータをダウンロードし、その後、残りのプロパティを背景スレッドでダウンロードする。

【００５９】

要求を受信した後、プロジェクトサーバ／データベース２１０は、クライアントによって送信されたバージョンスタンプより大きいバージョンスタンプを有する行に関して、バックエンドデータベース内のすべての１次テーブルにクエリを行う。それらの行に対応するデータが、プロジェクトに関する新たなデータ、または更新されたデータとしてクライアントコンピュータに戻される。また、プロジェクトサーバ／データベース２１０は、クライアントが提供したバージョンスタンプより後に削除された行に関しても、１次テーブル群に関連するすべてのシャドーテーブルにクエリを行う。それらの行に対応するデータは、削除された行としてクライアントコンピュータに戻される。プロジェクトサーバ／データベース２１０は、最新のプロジェクトバージョンスタンプをクライアントコンピュータに戻す。キャッシュマネージャ１１５が、そのバージョンスタンプを使用して、そのプロジェクトに関する前回のデータ同期の時刻をキャッシュ１１３の中に記録する。

【００６０】

このため、伝送されたバージョンスタンプに基づき、プロジェクトサーバ／データベース２１０が、クライアントコンピュータに適用される必要があるデルタ（「新たな」データ）を計算し、データの第１のビンを送信する。データの第１のビンを受信した後、クライアントは、プロパティ全体を検査して（ｗａｌｋ　ｔｈｒｏｕｇｈ）、キャッシュ１１３の中に格納された値を更新し、プロジェクトを読み込んで、ユーザが、ファイルに対する変更を行うことを可能にする。同時に、プロジェクトサーバ／データベース２１０は、データの第２のビンを異なるスレッドに送信し、ユーザがプロジェクトに取り組むにつれ、データを更新し続ける。ユーザがダウンロードされていないデータにアクセスした場合、プロジェクト管理アプリケーション１２６は、要求されたデータをダウンロードしながら、メッセージボックスを使用してユーザに通知する。

【００６１】

次に図４を参照すると、本発明の一実施形態による接続されたプロジェクトセッションに関するフローチャートが示されている。ブロック４０２で、ユーザは、プロジェクト管理アプリケーション１２６を使用して、プロジェクトファイルを開こうと試みる。ブロック４０４で、キャッシュマネージャ１１５は、その特定のプロジェクトファイルがキャッシュ１１３の中に存在するかどうかを確認する。ファイルは、例えばユーザがそのファイルを以前に開いている場合、キャッシュ１１３の中に存在する可能性がある。プロジェクトファイルがキャッシュ１１３の中に入っていない場合、ブロック４０６で、クライアントコンピュータ１００がプロジェクトサーバ／データベース２１０に現在、接続されているかどうかが判定される。プロジェクトサーバ／データベース２１０に対する接続が存在せず、ファイルがキャッシュの中に入っていない場合、ブロック４０８で、プロジェクト管理アプリケーション１２６は、ユーザにファイルを開くことができないことを通知する。

【００６２】
プロジェクトサーバ／データベース２１０に対する接続が存在する場合、ブロック４１０で、キャッシュマネージャ１１５は、その特定のプロジェクトファイルを要求する要求を、Ｗｅｂサービスインタフェース２１５を介してプロジェクトサーバ／データベース２１０に送信する。ファイルは、プロジェクトサーバ／データベース２１０からクライアントコンピュータに送信され、クライアントコンピュータにおいて、キャッシュマネージャ１１５は、そのファイルをキャッシュ１１３の中に格納する（ブロック４１２）。ブロック４１４で、ファイルは、キャッシュ１１３から読み込まれる。ブロック４１６で、そのファイルがキャッシュ１１３の中に入っている場合、キャッシュマネージャ１１５は、そのファイルが、現在、ログされているか、またはチェックアウトされているかを判定する。ファイルがチェックアウトされている場合、ブロック４１８で、キャッシュマネージャ１１５は、Ｗｅｂサービスインタフェース２１５を介して、ユーザが依然として、ファイルを編集する権利を有するかどうかなどの、セキュリティ資格情報を調べ、管理者（ａｄｍｉｎｉｓｔｒａｔｏｒ）が、ユーザによるファイルのチェックアウトを強制的に終了させているかどうかなどの、他の任意の更新情報を要求する要求を、プロジェクトサーバ／データベース２１０に送信する。ステップ４１４で、ファイルは、キャッシュ１１３から読み込まれる。ファイルが、ブロック４１６でチェックアウトされていなかった場合、フローは、ブロック４１０～４１４に進み、前述したとおり、データの差分のキャッシュ１１３への送信、およびキャッシュ１１３からの読み込みが行われる。

【００６３】
次に図５を参照すると、本発明による、接続を解除されたプロジェクトセッションに関するフローチャートが示されている。本明細書で使用する「接続を解除された」とは、クライアントコンピュータとプロジェクトサーバ／データベース２１０との間で接続が利用可能でないことを指す。同様に、「接続された」とは、クライアントコンピュータと、プロジェクトサーバ／データベース２１０との間で接続が利用できることを指す。ブロック５０２で、ユーザは、プロジェクト管理アプリケーションを使用して、プロジェクトファイルを開こうと試みる。ブロック５０４で、キャッシュマネージャ１１５は、その特定のプロジェクトファイルがキャッシュ１１３の中に格納されているかどうかを調べる。ファイルがキャッシュ１１３の中に入っていない場合、ブロック５０６で、プロジェクト管理アプリケーションは、ユーザにそのファイルを開くことができないことを通知する。

【００６４】
そのファイルが、キャッシュ１１３の中に入っている場合、ブロック５０８で、キャッシュマネージャ１１５は、そのファイルがチェックアウトされているかどうかを判定する。ブロック５１０で、ファイルがチェックアウトされている場合、ファイルがキャッシュ１１３から読み込まれる。ファイルがチェックアウトされていない場合、ブロック５１２で、ユーザに読み取り専用モードでファイルを開くように指示が行われる。ユーザがファイルを読み取り専用モードで開くことに同意した場合、ファイルは、キャッシュから、読み取り専用モードで開かれる。ユーザがファイルを編集することを要望する場合、ファイルは開かれない。

【００６５】
前述したとおり、クライアントコンピュータは、ローカルキャッシュ１１３とインタフェースをとることができるので、ユーザは、オフライン状態である場合でもクライアント上で作業することができる。オフライン状態にある間、ユーザは、新たなプロジェクトを作成すること、または既存のプロジェクトがチェックアウトされている場合、そのプロジェクトを編集することができる。オフラインで作業している場合、キャッシュマネージャ１１５は、すべてのデータをキャッシュ１１３に格納する。次回にユーザがプロジェクトサーバ／データベース２１０に接続した際、閾値に達すると、ある妥当性検査が実行され、キャッシュ１１３の中のファイルと、プロジェクトサーバ／データベース２１０との間で同期がとられる。

【００６６】

ほとんどのユーザは、クライアントとプロジェクトサーバ／データベース２１０との間におけるセキュリティで保護された通信のためにＨＴＴＰＳを使用する。本発明の実施形態によれば、クライアントとプロジェクトサーバ／データベース２１０は、データを固有のバイナリ形式で交換する。データは、より良好な帯域幅利用のために圧縮することもできる。好ましくは、未圧縮のデータに関してチェックサムが計算され、データとともにプロジェクトサーバ／データベース２１０に送信される。クライアントとプロジェクトサーバ／データベース２１０はともに、データを伸張した後、データのチェックサムを計算し、セキュリティ検証の一環として受信されたチェックサムに照らして、計算されたチェックサムを検証する。

【００６７】

前述したとおり、Ｐｒｏｊｅｃｔ　Ｐｒｏｆｅｓｓｉｏｎａｌは、マイクロソフトコーポレーションによって製造されたプロジェクト管理アプリケーションである。Ｐｒｏｊｅｃｔ　Ｓｅｒｖｅｒは、前述したとおり、複数のユーザが新たなプロジェクトを作成し、プロジェクトサーバ／データベース２１０のようなプロジェクトサーバから進行中のプロジェクトにアクセスすることができるようにする、Ｍｉｃｒｏｓｏｆｔによって提供される別のアプリケーションである。例えば、１つの組織が、専用サーバ上にＰｒｏｊｅｃｔ　Ｓｅｒｖｅｒをインストールし、その後、数名のプロジェクト管理者が、各システムがプロジェクトサーバと通信する個々のコンピューティングシステム上にＰｒｏｊｅｃｔをインストールすることが可能である。

【００６８】

本明細書で説明するとおり、プロジェクト管理アプリケーション、および関連するリソースを使用してプロジェクトをキャッシュし、同期をとるための方法およびシステムが提供される。本発明の趣旨または範囲を逸脱することなく、本発明において様々な変更または変形を行うことができることが、当業者には明白であろう。本発明の他の実施形態は、本明細書を考慮し、本明細書で開示する本発明を実施することで、当業者には明白となろう。

【図面の簡単な説明】

【００６９】

【図１】本発明の諸実施形態に関する典型的なコンピューティングシステムを示す図である。

【図２】本発明の諸実施形態による、プロジェクトサーバ／データベースと通信している、親プロジェクトの所有者／管理者、および１つまたは複数の子の下位工程の所有者／管理者の間における相互関係を示す分散コンピューティング環境の簡略ブロック図である。

【図３】本発明の実施形態によるフローチャートである。

【図４】本発明の実施形態による別のフローチャートである。

【図５】本発明の実施形態によるさらに別のフローチャートである。

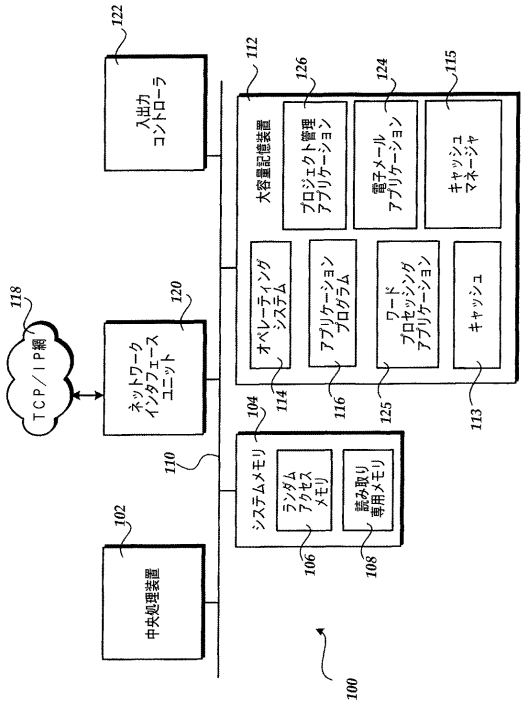
【符号の説明】

【００７０】

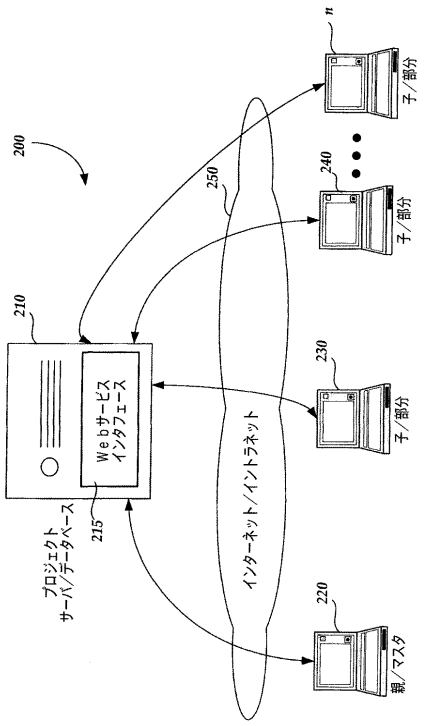
１０２　中央処理装置
１０４　システムメモリ
１０６　ランダムアクセスメモリ
１０８　読み取り専用メモリ
１１２　大容量記憶装置
１１３　キャッシュ
１１４　オペレーティングシステム
１１５　キャッシュマネージャ
１１６　アプリケーションプログラム
１１８　ＴＣＰ／ＩＰ網
１２０　ネットワークインタフェースユニット
１２２　入出力コントローラ

１２４　電子メールアプリケーション
１２５　ワードプロセッシングアプリケーション
１２６　プロジェクト管理アプリケーション

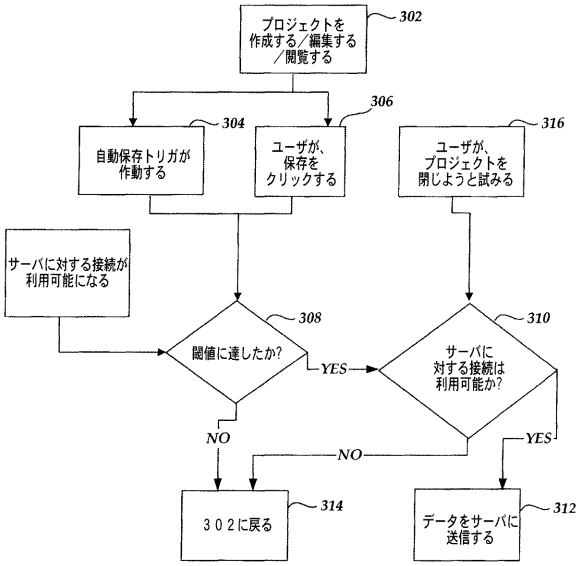
【図１】



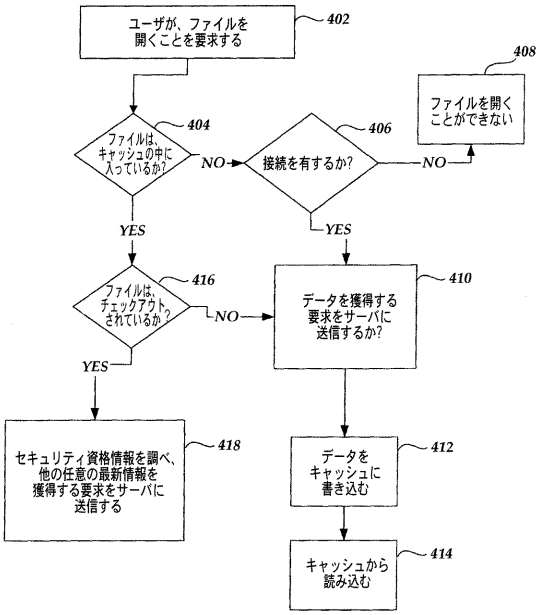
【図２】



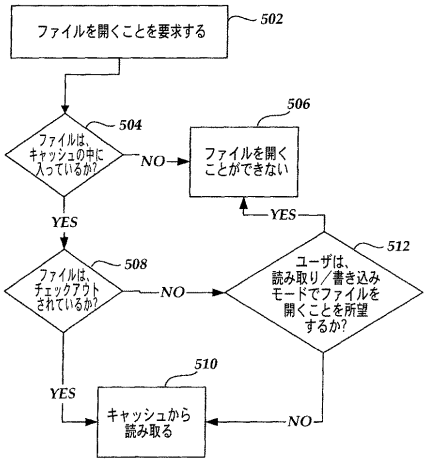
【図３】


プロジェクトを作成する／編集する／閲覧する
302
自動保存トリガが作動する
304
ユーザが、保存をクリックする
306
ユーザが、プロジェクトを閉じようと試みる
316
サーバに対する接続が利用可能になる
閾値に達したか?
308
YES
NO
サーバに対する接続は利用可能か?
310
NO
YES
３０２に戻る
314
データをサーバに送信する
312

【図４】


ユーザが、ファイルを開くことを要求する
402
ファイルは、キャッシュの中に入っているか?
404
NO
接続を有するか?
406
NO
ファイルを開くことができない
408
YES
YES
ファイルは、チェックアウトされているか?
416
NO
データを獲得する要求をサーバに送信するか?
410
YES
セキュリティ資格情報を調べ、他の任意の最新情報を獲得する要求をサーバに送信する
418
データをキャッシュに書き込む
412
キャッシュから読み込む
414

【図５】


ファイルを開くことを要求する
502
ファイルは、キャッシュの中に入っているか?
504
NO
ファイルを開くことができない
506
YES
YES
ファイルは、チェックアウトされているか?
508
NO
ユーザは、読み取り／書き込みモードでファイルを開くことを所望するか?
512
YES
キャッシュから読み取る
510
NO

フロントページの続き

(72)発明者　ラジュ　アール．アイヤル
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　コーポレーション内
(72)発明者　スディン　バト
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　コーポレーション内

審査官　岡北　有平

(56)参考文献　特開平１０－０３１６６０（ＪＰ，Ａ）
特開２００２－１７００５７（ＪＰ，Ａ）
特開２００２－１６９８８１（ＪＰ，Ａ）

(58)調査した分野(Int.Cl.，ＤＢ名)
Ｇ０６Ｑ　１０／００　－　５０／３４