(19)日本国特許庁(JP)　(12)特　許　公　報(B2)　(11)特許番号
特許第6345770号
(P6345770)
(45)発行日　平成30年6月20日(2018.6.20)　(24)登録日　平成30年6月1日(2018.6.1)

(51)Int.Cl.　F I
G06F 12/00 (2006.01)　G06F 12/00 533J
G06F 12/00 535Z
G06F 12/00 517

請求項の数 10　(全 12 頁)

(21)出願番号　特願2016-516632(P2016-516632)
(86)(22)出願日　平成25年9月21日(2013.9.21)
(65)公表番号　特表2016-523400(P2016-523400A)
(43)公表日　平成28年8月8日(2016.8.8)
(86)国際出願番号　PCT/US2013/061073
(87)国際公開番号　WO2014/193458
(87)国際公開日　平成26年12月4日(2014.12.4)
審査請求日　平成28年9月21日(2016.9.21)
(31)優先権主張番号　13/904,979
(32)優先日　平成25年5月29日(2013.5.29)
(33)優先権主張国　米国(US)

(73)特許権者　314015767
マイクロソフト　テクノロジー　ライセンシング，エルエルシー
アメリカ合衆国　ワシントン州　９８０５２　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

最終頁に続く

(54)【発明の名称】同期フレームワークの拡張可能性

(57)【特許請求の範囲】
【請求項１】
ファイルの第１インスタンスを格納しているコンピューティングデバイス上で実行される同期フレームワークにアプリケーションの統合を提供する方法であって、前記コンピューティングデバイスは、ネットワークを介してクラウドサービスと通信可能であり、前記クラウドサービスは、前記ファイルの第２インスタンスを格納し、前記クラウドサービスは、同期エンジンを備え、前記方法は、
ある期間中に前記ファイルの前記第１インスタンスへの更新を許容すると共に前記第２インスタンスへの更新を許容するステップと、
前記期間中における前記ファイルの前記第１インスタンスと前記第２インスタンスへの更新に応答して、ネットワークを介して前記同期フレームワークと前記同期エンジンとの間で更新を交換することによって前記ファイルの前記第１インスタンスと前記ファイルの前記第２インスタンス間の同期を自動的に維持するステップであって、前記同期フレームワークは、前記コンピューティングデバイス上で実行される任意のアプリケーションが前記同期フレームワークと通信することを可能にするインターフェイス又はＡＰＩ（アプリケーションプログラミングインターフェイス）を備える、ステップと、
前記コンピューティングデバイス上で実行される第１アプリケーションから前記インターフェイス又はＡＰＩを介して、前記ファイルに関連する同期ロック要求を受け取るステップと、
前記同期ロック要求に応答して、前記同期フレームワークによる前記ファイルの前記第

１及び第２インスタンス間の同期を一時的に放棄することを含む同期ロックを提供するステップであって、前記同期ロック要求が行われている間に前記ファイルの前記第１及び第２インスタンスの更新が許容され続ける、ステップと、
を含む、方法。
【請求項２】
前記第１アプリケーションによって前記ファイルの前記第１及び第２インスタンスを同期させるステップを更に含む、請求項１に記載の方法。
【請求項３】
前記第１アプリケーションによって同期ロック解除を発行するステップを更に含む、請求項２に記載の方法。
【請求項４】
前記同期フレームワークによる前記ファイルの前記第１及び第２インスタンス間の同期の維持を再開することによって、前記同期ロック解除に応答するステップを更に含む、請求項３に記載の方法。
【請求項５】
前記同期の維持を再開することは、前記第１アプリケーションによって前記同期フレームワークに登録されたマージハンドラーを前記同期フレームワークによって呼び出すことを含む、請求項４に記載の方法。
【請求項６】
前記マージハンドラーが前記ファイルの前記第１インスタンスと前記第２インスタンスをマージすることができない場合に前記ファイルのバージョンを保存するステップを更に含む、請求項５に記載の方法。
【請求項７】
コンピューティングデバイスであって、
ファイルを格納し、前記ファイルのクラウドバージョンを管理するように構成されたストレージデバイスと、
プロセッサーと、
同期エンジンと、
を備え、
前記同期エンジンは、前記ファイルを前記ファイルのそれぞれのクラウドバージョンと同期させるように構成され、その結果、前記ファイルの前記それぞれのクラウドバージョンへの変化が前記ファイルに同期すると共に、前記ファイルへの変化が前記ファイルの前記それぞれのクラウドバージョンに同期し、
前記同期エンジンは、クラウド上で実行され前記ファイルの前記クラウドバージョンを管理するストレージサービスの同期エンジンとネットワークを介して更新を交換するように構成され、
前記同期エンジンは、前記ファイルのうちの第１ファイルを同期ロックする要求を前記コンピューティングデバイス上で実行されるアプリケーションから受け取るように構成され、前記第１ファイルの同期ロックが行われている間に前記第１ファイルとそのクラウドバージョンの更新が許容され続け、
前記同期エンジンは、更に、前記要求のそれぞれに対して前記第１ファイルについての自動的な同期を停止し、前記アプリケーションが前記同期ロックの対応する解除を開始すると前記ファイルについての前記自動的な同期を再開することによって応答するように構成される、
コンピューティングデバイス。
【請求項８】
前記アプリケーションは、ファイルシステムを通じて前記ファイルを変更する、請求項７に記載のコンピューティングデバイス。
【請求項９】
前記ストレージサービスは、前記コンピューティングデバイスに関連付けられたユーザ

ー認証情報と共に前記コンピューティングデバイスにリンクされる、請求項８に記載のコンピューティングデバイス。
【請求項１０】
前記ユーザー認証情報にもリンクされた別のコンピューティングデバイスが、前記第１ファイルを更新し、前記第１ファイルの前記クラウドバージョンを格納する前記ストレージサービスが、前記別のコンピューティングデバイスによってなされた前記第１ファイルの前記クラウドバージョンに対する更新を前記コンピューティングデバイスにおける前記第１ファイルへ同期させる、請求項９に記載のコンピューティングデバイス。
【発明の詳細な説明】
【背景技術】
【０００１】
[01] ネットワークやクラウドコンピューティングの分野では、クラウドに接続しているクライアントコンピューターによる使用のためのクラウドストレージを提供するサービスが、開発されてきている。クラウドストレージは多くの方法で実装されているが、最近では、クラウドストレージは、クライアントコンピューターのファイルシステム（及びオペレーティングシステム）と密接に統合されてきている。ユーザーの観点からは、クラウドと、クラウドに接続しているユーザーのクライアントコンピューティングデバイスとにおいて普遍的な、又は浮動的な存在を有するように見えるクラウドファイルを提供することが、望ましくなってきている。目的は、クラウドに格納されているファイルが、おそらくはクラウド内のソフトウェアだけでなく、クラウドに接続しているユーザーのデバイスによっても読み書きされるようにすることである。実際のところ、（ユーザーの観点から）単一のファイルエンティティが、並列で更新が行われている複数のコピー又はバージョンをクラウド上、及びクライアントコンピューター上に有するかもしれない。ファイルに対する整合性を維持するために、即ちファイルのコンテンツをクラウド及びクラウドに接続しているユーザーデバイスにわたって一致させ続けるために、ファイル同期システム、あるいは同期エンジンが利用されることができる。
【０００２】
[02] 円滑なユーザー体験とトランスペアレントな整合性を提供するために、このクラウドベースのファイルの同期は、好ましくはファイルシステム及び／又はオペレーティングシステムのレベルで実施される。これにより、同一の論理ファイルであると想定されるインスタンス間の同期を維持するために通常必要とされる準備作業が覆い隠される。このシステム管理型の同期はまた、プログラマーを彼ら独自のカスタム同期ソフトウェアをコーディングしなければならない負担から解放する。
【０００３】
[03] それにもかかわらず、同期エンジンとクライアントオペレーティングシステムの外部のアプリケーション又は他のソフトウェアが、カスタム同期を実施し、あるいはまた、ファイルのクラウド及びクライアントベースのインスタンスを管理し、調和させる必要がある時があるかもしれない。例えば、ワードプロセッサーの共同同時編集は、システム供与の同期エンジンが対処することが不可能な同期問題を伴うかもしれない。加えて、オペレーティングシステムレベル又はファイルシステムレベルの同期エンジンは、同期関連の責務を任意のアプリケーションと共有することが可能ではなかった。もしアプリケーションが、同期エンジンと競合又は干渉するかもしれない方法でクラウド共有ファイルを使用する必要があるのであれば、唯一の選択肢は、そのアプリケーションが同期エンジンからファイルに対する全面的な責務を引き受けることであろう。
【０００４】
[04] 以下に論じられるのは、数ある中でも、同期エンジンがファイルの同期を一時的に放棄し、アプリケーションによって提供される同期関連機能を利用できるようにするための手法である。
【発明の概要】
【０００５】

[05] 次に述べる概要は、以下の詳細な説明において論じられるいくつかの概念を紹介するためにのみ含まれている。この概要は包括的ではなく、末尾に提示されるクレームによって定められる請求主題の範囲を描写するようには意図されていない。

【０００６】

[06] 本明細書で説明される実施態様は、アプリケーションがシステムレベルの同期フレームワークと連携できるようにすることを含むことができる。同期フレームワークは、ユーザーデバイスとクラウドストレージサービスとの間におけるファイルのシステム同期を提供することができる。ユーザーコンピューティングデバイス上の任意のアプリケーションは、同期フレームワークと通信して、同期フレームワークによって指定されたファイルの同期を一時的に中断することが可能である。アプリケーションは、同期を中断することに関連して同期フレームワークが呼び出すことが可能な機能を、同期フレームワークに登録して、ファイルに対するシステムレベルのアクセスを任意のアプリケーションに引き続き提供し、同期を再開することが可能である。

【０００７】

[07] 付随する特徴の多くは、添付の図面と関連して考慮される下記の詳細な説明を参照して以下に説明される。

【図面の簡単な説明】

【０００８】

[08] 本説明は、添付の図面に照らして読まれる下記の詳細な説明から、より一層理解されるだろう。図面では、添付の記載における同様の部分を指定するために同様の符号が用いられる。

【０００９】

【図１】[09] 図１は、クラウドにおいて横断的に格納されているファイルを共有するユーザーのコンピューティングデバイスを示す。

【図２】[010] 図２は、ファイル同期処理が問題となり得る例示的なシナリオを示す。

【図３】[011] 図３は、ファイル操作活動を適切にコーディングされたソフトウェアと調和させるための拡張アプリケーションプログラミングインターフェイス（ＡＰＩ）を備えた同期エンジンを示す。

【図４】[012] 図４は、処理を示す。

【図５】[013] 図５は、別の処理を示す。

【図６】[014] 図６は、実施態様が具現化されることができる例示的なコンピューティングデバイスを示す。

【発明を実施するための形態】

【００１０】

[015] 以下に論じられる実施態様は、アプリケーションがファイルから同期エンジンを一時的に締め出す、即ち同期エンジンの同期監視からファイルを遮断することを可能にする拡張可能性を、同期エンジンに提供することに関する。実施態様はまた、例えば同期処理の問題を解決し、又は同期されていないファイルからデータを提供するのを手助けするために同期エンジンが呼び出すことが可能な機能を、アプリケーションが同期エンジンに登録することを可能にすることに関する。

【００１１】

[016] 以下の説明は、複数のデバイスによって共有されるクラウドベースのファイルストレージの概観から始まる。次に、同期フレームワーク、及び同期エンジンとアプリケーション間で起こり得る競合が説明され、その後に、同期エンジンが同期を実施し、また一時的に同期されていないファイルへのアクセスを提供するのを手助けすることができるアプリケーション機能の同期ロック及び同期エンジン使用を可能にする、同期エンジンに組み込まれることが可能な拡張可能性の説明が続く。

【００１２】

[017] 図１は、少なくともクラウド１０６とユーザーコンピューティングデバイス１０２にわたって格納されているファイル１０４を共有しているユーザーのコンピューティ

ングデバイス１００、１０２を示す。クラウド１０６は、図１には全てが示されていない様々なサービスを有することができる。クラウド１０６は、ユーザーアカウント又はユーザー認証情報を管理するアカウントサービスを有することができる。アカウントサービスは、各ユーザーに認証情報を提供することができる。ユーザーは、彼又は彼女の認証情報を使用して、クラウド１０６及びクラウド１０６内の様々なサービス、例えば検索エンジン、電子メールサービス、仮想マシン若しくはアプリケーションホスティングサービス、ストレージ若しくはファイルサービス１０８、又はその他にアクセスするかもしれない。そのようなクラウドサービスは、コンピューティングデバイス１００のようなユーザーのコンピューティングデバイス上のウェブブラウザー又は他のクライアントソフトウェアを用いてアクセスされるかもしれない。所与のユーザーは、同一のアカウント又は認証情報を用いて彼らの複数のコンピューティングデバイスをクラウド１０６においてリンクさせるかもしれない。図１の例では、ユーザーは、コンピューティングデバイス１００（デバイス１）とコンピューティングデバイス１０２（デバイス２）の両方をクラウドファイルサービス１０８とリンクさせ、それによりファイル１０４のシームレスな共有を可能にしている。

【００１３】

[018] いくつかのケースでは、もしユーザーの認証情報によってクラウド１０６から承認が得られれば、クライアントソフトウェアはクラウドサービスにトランスペアレントにアクセスするかもしれない。例えば、ユーザーコンピューティングデバイス１００は、オペレーティングシステム１１２によって管理されるローカルファイルシステム１１０を探索するための、ファイルエクスプローラー、システムブラウザー、又は他のユーザーインターフェイスを有するかもしれない。そのようなクライアントサイドのソフトウェアはまた、ファイルサービス１０８と接続し、あたかもローカルファイルシステム１１０の一部分であるかのように、ファイルサービス１０８上のファイルの探索を可能にするかもしれない。クライアントサイドのソフトウェアはまた、ユーザーコンピューティングデバイス１００、１０２上又は他のどこかで実行している任意のアプリケーションであることも可能である。

【００１４】

[019] いくつかのクラウドサービスは、ファイル又はファイルストアが複数のユーザーデバイス及びクラウド１０６上に同時に格納されアクセスされることを可能にする。例えば、図１を参照すると、ファイル１０４（「ファイル１」）などのファイルは、ユーザーコンピューティングデバイス１００のファイルシステム１１０にローカルに格納されることができる。ファイル１０４はまた、クラウドのファイルサービス１０８に格納されることもあるかもしれない。またファイルサービス１０８におけるファイル１０４のコピーは、ファイルサービス１０８によって、ユーザーコンピューティングデバイス１０２のようなユーザーの他のデバイス上の他のアプリケーションへ提供されることもあるかもしれない。複数のエンティティがファイル１０４にアクセスし潜在的に書き込みをするので、同期フレームワークが、ファイル１０４のインスタンス又はコピーを同期させるために利用される。

【００１５】

[020] 同期フレームワークは、ユーザーコンピューティングデバイス１００上の同期エンジン１１４と（ファイルサービス１０８の背後に存在するかもしれない）クラウド１０６上の同期エンジン１１６のような、連携する同期エンジンを含むことができる。各同期エンジンは、対応するファイル１０４のローカルコピーをファイル１０４のもう一方のコピーと同期状態に保つためのローカルステップを実施する（ファイル１０４は任意のファイルを表すことに留意されたい）。概して、このことは、同期エンジンがファイル１０４のローカルバージョンへのローカルな更新を受け取る又は検出して、それらの更新を別の同期エンジンへ伝達し、それに応じて別の同期エンジンが、ファイル１０４のコピーが同期状態となるようにファイル１０４のローカルコピーを更新することを含む。

【００１６】

[021] 一実施態様では、ファイルサービス１０８は中央ストレージ及びコーディネーターとして動作することができ、その結果ユーザーのデバイスは互いと直接的には同期しないかもしれない。例えば、ユーザーコンピューティングデバイス１０２は、同期更新情報をクラウド１０６と交換し、同様にユーザーコンピューティングデバイス１００は、更新情報をクラウド１０６と交換するかもしれない。しかしながら、コンピューティングデバイス１００とコンピューティングデバイス１０２は、ファイルサービス１０８における仲介の同期エンジン１１６を通じて間接的に互いと同期し、実際には同期エンジン１１６が、一方のユーザーデバイスの更新情報を他方へ伝達する。同期エンジンの具体的なアルゴリズムは、それがクラウド１０６にあるのか又はユーザーデバイスにあるのかによって様々であり得る。

【００１７】

[022] 上述された基本的な同期のロジックに加えて、同期エンジン１１４、１１６はまた、クラウド１０６とユーザーデバイス１００とのネットワーク接続の断続的な喪失に対処するためのロジックも有することができるであろう。例えば、ユーザーデバイス１００とクラウド１０６との間において接続が利用可能でない場合、ユーザーデバイス１００上のアプリ１　１１８は、編集又は更新をローカルファイル１０４に書き込むかもしれない。切断中の更新は、接続が再び確立されるまで両方のロケーションに蓄積されなければならない。接続が再び確立されると、同期エンジン１１４、１１６は連携して、蓄積された更新を整合させようと試みることができる（クラウド１０６にあるファイル１０４のコピーは、おそらくは依然としてクラウド１０６に接続されているユーザーデバイス１０２から、更新を受け取っていることもあるかもしれない）。ファイル１０４のインスタンスが整合できない場合、ローカルバージョン１２０及びサブバージョン（subversion）までもが作成され、保存されるかもしれない。同期フレームワークによって整合されない異なるバージョン１２０は、最終的にはユーザーの介入を必要とするかもしれない。

【００１８】

[023] 図２は、ファイル同期処理が問題となり得る例示的なシナリオを示す。この例では、（任意のアプリケーションを代表している）アプリ１　１１８は、ユーザーコンピューティングデバイス１００上のファイル１０４のローカルインスタンスへのアクセスのみならず、クラウド１０６にあるコピー又はインスタンスへのアクセスを、おそらくは、オペレーティングシステム１１２からユニフォームリソースロケーター（ＵＲＬ）又はクラウドベースファイル１０４に対する他のネットワークファイルハンドルを取得した後に、有するものと想定される。初めに同期エンジン１１４が、上で論じられたように、ファイル１０４をクラウド１０６と同期処理している。その期間中に、アプリ１　１１８が、ファイル１０４のローカルコピーとクラウドファイルサービス１０８におけるコピーの両方にアクセスし書き込むことを開始する。そのようなファイル１０４の並列使用は、最終的に競合又は干渉を引き起こし、潜在的にはファイル１０４を破損し又は同期フレームワーク若しくはアプリ１　１１８の動作を妨害することがある。

【００１９】

[024] 図３は、ファイル操作活動をユーザーデバイス１００上で走っている（アプリ１　１１８によって代表される）適切にコーディングされたソフトウェアと調和させるための拡張アプリケーションプログラミングインターフェイス（ＡＰＩ）１３０を備えた同期エンジン１１４を示す。ＡＰＩは、同期エンジン１１４がどのようにしてコール／応答１３２若しくはメッセージをアプリケーションと通信又は交換することが可能であるかの一例にすぎない。ソフトウェアのインターフェイス及び契約などの他のプログラミング手法が用いられてよい。加えて、同期エンジン１１４との通信が論じられたが、ＡＰＩ１３０は、同期フレームワーク全体のうちの一部分として現れてもよい。

【００２０】

[025] ＡＰＩ１３０は、アプリ１　１１８などの任意のアプリケーションによって呼び出し可能な様々なコール、メソッド等を含むことができる。そのような１つのコールは、同期ロックコールであってよく、同期ロックコールはまた、ファイル１０４などのファ

イルを指定し、又は対象とすることが可能である。コールが、例えばアプリ１　１１８によってファイル１０４に対して呼び出されると、同期エンジン１１４は、ファイル１０４をクラウド１０６と同期するのを停止するように進行する。このことは、未処理の更新をファイル１０４に適用するステップ、クラウド１０６における同期エンジン１１６と通信して、同期フレームワークがおそらく一時的にファイル１０４に対する同期の責務を放棄していることを示すステップなどの、様々な予備的措置を伴うことができる。同期エンジン１１６はこれを受けて、更新を適用することができ、又は、ビットをディスクに書き出すようクラウドファイルサービス１０８に命令するといったような他のタスクを実施することができる。同期エンジン１１４は、自身の状態に従って同期ロックコールに応答することができる。例えば、もしローカルファイル１０４が、おそらくはクラウド１０６とのネットワーク接続の喪失のために整合されることが不可能であるならば、同期エンジン１１４は、それに応じてアプリ１　１１８に応答することができるであろう。もしローカルファイル１０４の分岐（新たなサブバージョン）が実施されるならば、アプリ１　１１８は通知されることもできるであろう。一実施態様では、アプリ１　１１８はまず、ローカルファイル１０４及び／又はファイル１０４のクラウドベースのインスタンスを読み取り専用モードで開き、同期ロックコールが正常に返ってくると、アプリ１　１１８は次に、ファイル１０４を書き込みモードで開く。同期ロックは、少なくともローカルユーザーデバイスにおいて実行されるが、例えばファイルの名前変更及び移動動作の取り扱いを改善するために、オプションとして他のユーザーデバイスに拡張されることが可能である、ということに留意されたい。

【００２１】

[026]　一実施態様では、アプリ１　１１８は、データプロバイダー１３６をおそらく有する基礎的なファイルストリーミングサーバーとして動作するようにコーディングされることができる（マージハンドラー１３４とデータプロバイダー１３６の両方が、アプリ１　１１８によってＡＰＩ１３０を介して登録されることが可能である）。オペレーティングシステム１１２は、実際のファイルの代わりにプレースホルダーが用いられるようにするファイルプレースホルダーシステムを提供することが可能である。アプリケーションにとって、プレースホルダーファイルは、オープン可能／クローズ可能等のファイルプロパティを持った通常のファイルのように見えるが、そのファイルのデータ（コンテンツ）は、ファイルシステム１１０以外のソースに由来する。換言すれば、プレースホルダーファイルは、ファイルとして提示されるオペレーティングシステムのオブジェクトであるが、内部的には、当該ファイル内に存在するように見えるデータは、ファイルストレージ装置以外のソース、例えば、データをオペレーティングシステムへストリーミングすることが可能な機能又はクライアントを備えたアプリケーションに由来する。この例では、アプリ１　１１８のデータプロバイダー１３６が、ファイル１０４についての自身のコンテンツ（即ち、同期ロックが呼び出された後にデータプロバイダー１３６がファイル１０４に対して行ったものは何でも）をオペレーティングシステムへストリーミングし、ストリーミングされたコンテンツが、ファイル／オペレーティングシステムを介して、ファイル１０４をファイルシステムを通じて読み出す任意のものへ提供される。プレースホルダー／ストリーミング機能は同期ロックには必要とされないことに留意されたい。

【００２２】

[027]　同期ロックがアプリ１　１１８によって行われる間、同期フレームワークはユーザーデバイス又はクラウド１０６においてファイル１０４を同期させない。したがって、アプリ１　１１８は自由に自らの同期ロジックを実現することができ、あるいはまた、同期フレームワークとの干渉を懸念することなくローカルインスタンス及び／又はクラウドインスタンスの両方へ書き込むことができる。例えば、アプリ１　１１８は、おそらくは自らのマージハンドラー１３４又は整合ロジックを用いて、ファイル１０４のクラウドコピーとローカルコピーが同一であるか否かをシステムから問い合わせることができるであろう。以下で更に論じられるように、アプリ１　１１８のマージハンドラー１３４は、インターフェイス又は契約を介して同期エンジン１１４へ提示されることができ、アプリ

ケーション提供型マージを実施するために同期エンジン１１４によって呼び出されることが可能である。即ち、同期フレームワークは、外部の同期機能を用いて拡張されることが可能である。

【００２３】

[028] やがて、アプリ１　１１８は同期ロックを解除する。これは、当該アプリケーションによってその内部ロジック（例えば、アプリ１のユーザーがファイル１０４を明示的に閉じる）を通じて明示的に引き起こされることができる。あるいは、同期ロックの解除は、アプリ１　１１８が閉じられる、又はアプリ１　１１８がフォーカスから外れる、ユーザーによって切り換えられて見えなくなる（ユーザーが別のアプリケーションへ切り換える）、システムのサスペンドが生じる等のようなイベントに応答して自動的に引き起こされることが可能である。

【００２４】

[029] 同期ロックが解除されると、同期エンジン１１４はファイル１０４の同期管理を再開する。もし上述されたようにプレースホルダーモードが使用中であるなら、これは未処理のプレースホルダーを解決するためのロジックを実行するステップを伴うことができるであろう。加えて、同期エンジン１１４はファイル１０４を同期させようと試みることができ、ファイル１０４はそれに応じて、ファイル１０４の水和（hydration）を引き起こすことができるであろう。同期エンジン１１４はデータプロバイダー１３６又はマージハンドラー１３４にコンタクトすることができ、アプリ１　１１８がデータプロバイダー１３６を用いてファイルを水和させる（hydrate）ように有効化される。それに引き続いて、同期エンジン１１４は、クラウド１０６へのファイル１０４のアップロードが未処理であるかどうかを調べることができるであろう。もし未処理でないなら、同期フレームワークは、ファイル１０４に関して定常状態にある。たとえ未処理のローカル更新が何もないとしても、同期フレームワークは、もしクラウドバージョンが変化したなら、いくつかのより最近の更新をクラウドからダウンロードする理由を有することができる。そのようなケース（未処理のローカル変更が何もないが、より最近のクラウドバージョンがある）では、クラウドバージョンがローカルファイルに置き換わる（競合なし）。しかしながら、もしアップロードが未処理であるなら、クラウドバージョンがダウンロードされることができる（同期エンジンはクラウド変更を検出するためにクラウドバージョンをダウンロードする必要はなく、クラウド変更は他の利用可能な情報に反映されることが可能である）。もしクラウド変更が検出されなければ、再び同期エンジン１１４は定常同期状態にある。そうでなければ、アプリ１　１１８の登録されたマージハンドラー１３４が同期エンジン１１４によって呼び出される。もしそれが失敗したら、ファイル１０４を分岐させる（バージョンを作成する）ステップ、又はユーザーが不整合をどのように解決するか選択するのを可能にするユーザーインターフェイスを表示するステップなどの、何らかの障害対処手順が引き起こされる。

【００２５】

[030] 引き続き図４、５と図３に示された時系列とを参照すると、まず、ステップ１６０において、同期エンジン１１４は、ファイル１０４に対する同期を管理している。ファイルシステムとクラウドをそれぞれ介した任意のアプリケーションによるローカル及びクラウドインスタンスへの書き込みは、アプリケーションに対してトランスペアレントに同期される。ステップ１６２において、アプリ１　１１８などの任意のアプリケーションが、ファイル１０４に対する独自の同期管理の実行を要求することができる。ステップ１６４（第１時点１３８）において、このアプリケーションは、同期ロック要求を同期エンジン１１４へ発行し、オプションとして、（同期エンジン１１４との交換を通じて）マージハンドラー１３４及び／又はデータプロバイダー１３６を登録する。ステップ１６６において、同期エンジンは、クラウドとのファイルの同期を放棄することによって同期ロック要求を許可する。

【００２６】

[031] ステップ１６８において、アプリ１　１１８は、ファイルの制御権を有し、フ

ァイル１０４のローカル及び／又はクラウドインスタンスへ書き込みをし、おそらくはカスタムの同期のために更新をする。第２時点１４０より前に、第２アプリケーション（アプリ２）がファイルシステムを介してファイル１０４を要求するかもしれない。その場合、オペレーティングシステムが、データプロバイダー１３６からのコンテンツを備えたプレースホルダーファイルを供給する。第２時点においてステップ１７０が行われ、アプリ１　１１８はファイルを使い終わり、そしてステップ１７２において、同期アンロックコールを同期エンジン１１４へ発行する。ステップ１７４において、同期エンジン１１４は、ファイルをクラウドと同期することを再開する。これは、ファイルを同期させようと試みるステップ１７６と、そしておそらくは、ステップ１７８においてマージハンドラー１３４を呼び出すこととを含むことができるであろう。図４及び５におけるステップの特定的な順序は必要とされず、いくつかのステップが省略されてもよい。

【００２７】

[032]　要約すると、任意のアプリケーションは、例えばシステム若しくは組み込みの同期フレームワークにはあるいは利用可能ではないかもしれない同期関連サービス又は拡張を、同期フレームワークに提供することが可能である。アプリケーションがファイルを作業している間にファイル整合性が保証されることが可能であり、アプリケーションと同期フレームワークとの間における挙動の競合が回避されることが可能であり、そして同期ロック中に他のアプリケーションがプレースホルダー（同期フレームワークがファイルのコンテンツへのアクセスを仲介する）を介してトランスペアレントにファイルにアクセスすることが可能である。アプリケーションがファイルの完全な所有権を持ち、カスタムの同期ロジックを適用するのを可能にすることによって、オールオアナッシングの対処法が回避されることができる。アプリケーションは、これらのファイルを識別し取得（同期ロック）するために同期フレームワークと交渉することが可能である。契約、ＡＰＩ、アプリケーション実装可能ソフトウェアインターフェイス、又は類似のメカニズムが、組み込みの同期フレームワークに付加されることが可能である。同期フレームワークは、同期のメインエージェントとして動作することが可能であるが、それでもなお、アプリケーションがファイルを操作している際には、当該アプリケーションによってこの任務から一時的に解放されることが可能である。アプリケーションは、所与のファイルに対してスマートなマージ（smart merge）を実施することが可能であることを示すことができる。同期フレームワークが同期の競合を検出すると、同期フレームワークはその結果、競合の解決を当該アプリケーションに委任することが可能であることを知ることができる。

【００２８】

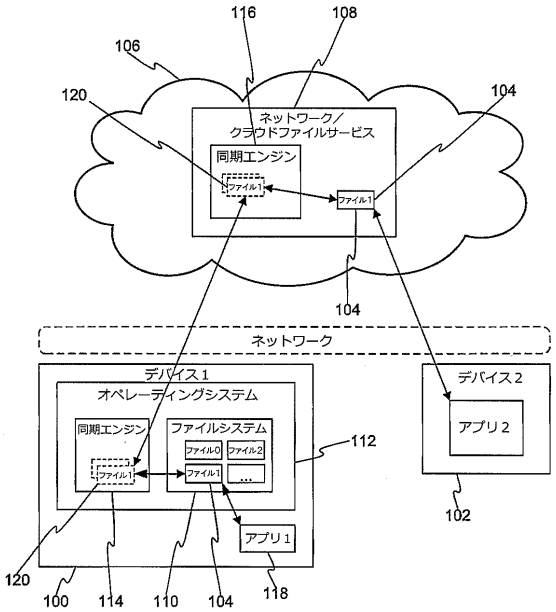
[033]　図６は、上述された実施態様が具現化されることができるコンピューティングデバイス１００の一例を示す。コンピューティングデバイス１００は、ストレージデバイス２６２及びプロセッサー２６４のみならず、１又は複数のディスプレイ２６６を有することができる。これらの要素は、コンピューティングの分野においてよく理解されている方法で連携することができる。加えて、入力デバイス２６８が、コンピューティングデバイス１００に統合され、あるいはコンピューティングデバイス１００と通信可能な状態で統合されることができる。ディスプレイ２６６は、コンピューティングデバイスによって出力された信号を表示するのに使用される様々なデバイスであってよく、例えば、固体面ディスプレイ（solid-surface display）、プロジェクター、タッチ感応表面などを含む。コンピューティングデバイス１００は、任意のフォームファクターを有することができ、又は別のデバイスに組み込まれることができる。例えば、タッチ感応式コントロールパネルが、電気製品、ロボット、及び他の機械を制御するためにしばしば用いられる。コンピューティングデバイス１００は、スマートフォン、タブレットコンピューター、ゲーミングボックス、ヘッドレスサーバーなどのような、携帯型デバイスの形であってもよい。

【００２９】

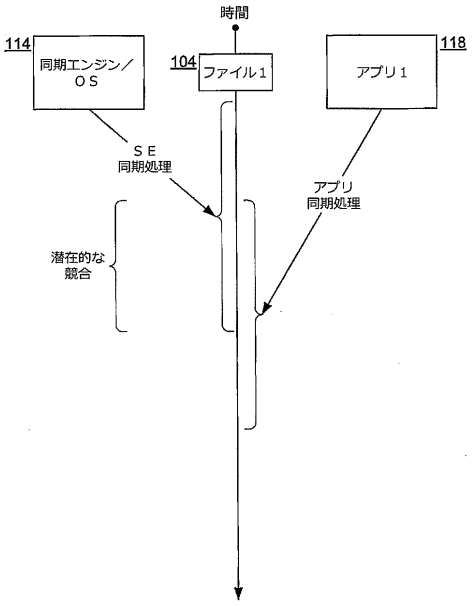
[034]　上で論じられた実施態様及び特徴は、揮発性若しくは不揮発性のコンピューター可読又はデバイス可読デバイスに格納された情報の形態で実現されることが可能である。これは、光学ストレージ（例えばコンパクトディスク読み出し専用メモリ（ＣＤ－ＲＯ

M））、磁気メディア、フラッシュ読み出し専用メモリ（ＲＯＭ）、又はデジタル情報を物理的物体に格納するための任意の他のデバイスなどのデバイスを少なくとも含むとみなされる。格納された情報は、機械実行可能命令（例えばコンパイルされた実行可能なバイナリコード）、ソースコード、バイトコード、又は、上で論じられた様々な実施態様を実施するようにコンピューティングデバイスを作動させる若しくは構成するのに用いられることが可能な任意の他の情報の形態であってよい。これもまた、少なくとも、実施態様を実現するプログラムの実行中に中央処理装置（ＣＰＵ）命令のような情報を記憶するランダムアクセスメモリ（ＲＡＭ）及び／又は仮想メモリなどの揮発性メモリと、プログラム若しくは実行可能ファイルがロードされ実行されることを可能にする情報を記憶する不揮発性媒体を含むとみなされる。本実施態様及び特徴は、ポータブルデバイス、ワークステーション、サーバー、モバイルワイヤレスデバイス等を含む任意のタイプのコンピューティングデバイス上で実施されることが可能である。

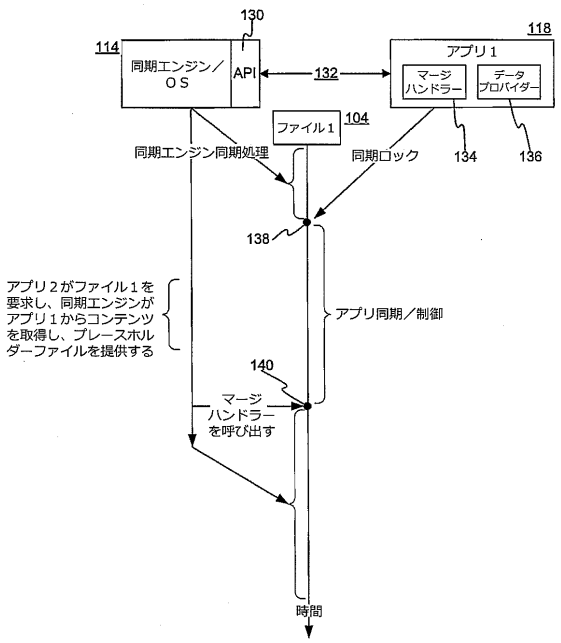
【図１】



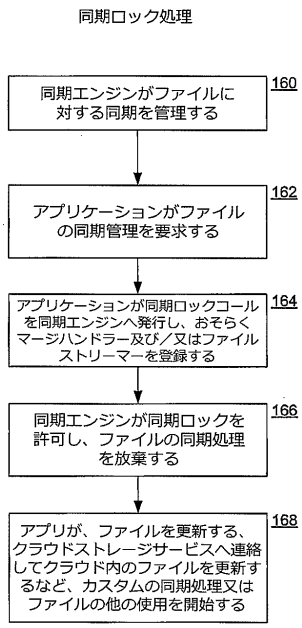
【図２】



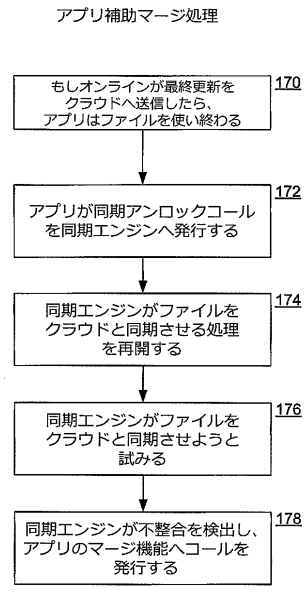
【図３】



【図４】



【図５】



【図６】

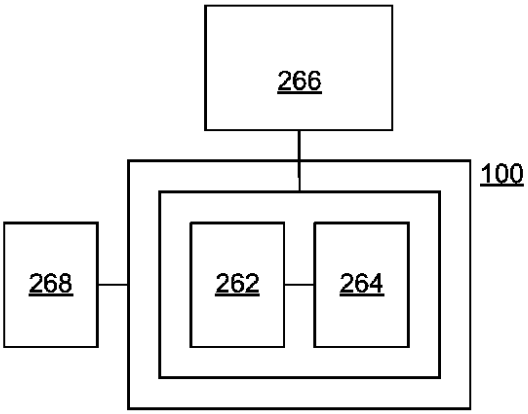


FIG. 6

フロントページの続き

(74)代理人 100138759
弁理士 大房 直樹
(72)発明者 ウォティエ, マルク
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, マイクロソフト コーポレーション, エルシーエイ－インターナショナル・パテンツ
(72)発明者 フィオルダリス, ダニエル
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, マイクロソフト コーポレーション, エルシーエイ－インターナショナル・パテンツ
(72)発明者 ボーズ, ミコ・アルナブ・エス
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, マイクロソフト コーポレーション, エルシーエイ－インターナショナル・パテンツ
(72)発明者 フーガーワーフ, スコット
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, マイクロソフト コーポレーション, エルシーエイ－インターナショナル・パテンツ
(72)発明者 シェケル, オデッド
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, マイクロソフト コーポレーション, エルシーエイ－インターナショナル・パテンツ
(72)発明者 クラーク, サイモン
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, マイクロソフト コーポレーション, エルシーエイ－インターナショナル・パテンツ
(72)発明者 グザク, クリス
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, マイクロソフト コーポレーション, エルシーエイ－インターナショナル・パテンツ
(72)発明者 バラスブラマニアン, バラジ
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, マイクロソフト コーポレーション, エルシーエイ－インターナショナル・パテンツ
(72)発明者 ノヴァック, マイケル
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, マイクロソフト コーポレーション, エルシーエイ－インターナショナル・パテンツ

審査官 甲斐 哲雄

(56)参考文献 特表２０１２－５２０５０４（ＪＰ, Ａ）
米国特許出願公開第２０１１／０１３７８７９（ＵＳ, Ａ１）

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