(19)日本国特許庁(JP)　(12)特　許　公　報(B2)　(11)特許番号
特許第4344324号
(P4344324)
(45)発行日　平成21年10月14日(2009.10.14)　(24)登録日　平成21年7月17日(2009.7.17)

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

請求項の数 14 (全 33 頁)

(21)出願番号	特願2004-569635 (P2004-569635)
(86)(22)出願日	平成15年2月25日 (2003.2.25)
(65)公表番号	特表2006-508476 (P2006-508476A)
(43)公表日	平成18年3月9日 (2006.3.9)
(86)国際出願番号	PCT/US2003/005753
(87)国際公開番号	WO2004/084082
(87)国際公開日	平成16年9月30日 (2004.9.30)
審査請求日	平成18年2月27日 (2006.2.27)
(31)優先権主張番号	10/084,257
(32)優先日	平成14年2月25日 (2002.2.25)
(33)優先権主張国	米国(US)

(73)特許権者　504323755
シーベル・システムズ・インコーポレーテッド
アメリカ合衆国・９４４０４・カリフォルニア州・サンマテオ・ブリッジポイントパークウエイ・２２０７
(74)代理人　100064621
弁理士　山川　政樹
(74)代理人　100098394
弁理士　山川　茂樹
(72)発明者　フアン，シアオ・フレイ
アメリカ合衆国・９４４０４・カリフォルニア州・フォスター　シティ・ブライゼストリート・１１３９

最終頁に続く

(54)【発明の名称】コンピューティング・デバイスとのサーバ同期のための方法およびシステム

(57)【特許請求の範囲】
【請求項１】
ハンドヘルド・デバイスをサーバに結合するステップであって、前記サーバは第１のデータベースを有し、そして前記ハンドヘルド・デバイスは第２のデータベースとその第２のデータベースにユーザがアクセスできるようにするアプリケーションとを有する、ステップと、
アプリケーション定義を更新する必要があるか否かを判定するステップと、
前記アプリケーション定義を更新する必要がある場合、前記ハンドヘルド・デバイスが前記アプリケーション定義を更新するための情報を記憶するのに充分な利用可能なメモリを有するか否かを判定するステップと、
前記ハンドヘルド・デバイスが前記アプリケーション定義更新情報を記憶するのに充分な利用可能なメモリを有すると判定された場合、前記アプリケーション定義更新情報を前記ハンドヘルド・デバイスに前記サーバが送信し記憶するようにさせる、ステップと、
前記ハンドヘルド・デバイスが前記アプリケーション定義更新情報を記憶するのに充分な利用可能なメモリを有さないと判定された場合、充分な利用可能なメモリを有さないことを示すメッセージを前記ハンドヘルド・デバイスが表示するようにさせるステップと、
ユーザにより前記第２のデータベースになされたトランザクションを前記ハンドヘルド・デバイスが記録するようにさせるステップと、
トランザクション情報を前記サーバへ前記ハンドヘルド・デバイスが提供するようにさせるステップであって、前記トランザクション情報は前記記録されたトランザクションに

関連している、ステップと、
前記トランザクション情報に基づいて、前記第１のデータベースに対してトランザクションを前記サーバが実行するようにさせるステップと、
前記第１のデータベースから前記第２のデータベースを更新するのに使用されるデータを前記サーバが抽出するようにさせるステップと、そして
前記抽出されたデータの少なくとも一部を前記ハンドヘルド・デバイスに前記サーバが提供するようにさせるステップと
を含むコンピュータで実施される方法。
【請求項２】
ハンドヘルド・デバイスをサーバに結合する手段であって、前記サーバは第１のデータベースを有し、そして前記ハンドヘルド・デバイスは第２のデータベースとその第２のデータベースにユーザがアクセスできるようにするアプリケーションとを有する、手段と、
前記第１のデータベースから前記第２のデータベースを更新するために使用されるデータを前記サーバが抽出するようにさせる手段と、
前記抽出データの少なくとも一部分を前記ハンドヘルド・デバイスに前記サーバが提供するようにさせる手段と、
アプリケーション定義を更新する必要があるか否かを判定する手段と、
前記アプリケーション定義を更新する必要がある場合、前記ハンドヘルド・デバイスが前記アプリケーション定義を更新するための情報を記憶するのに充分な利用可能なメモリを有するか否かを判定する手段と、
前記ハンドヘルド・デバイスが前記アプリケーション定義更新情報を記憶するのに充分な利用可能なメモリを有すると判定された場合、前記アプリケーション定義更新情報を前記ハンドヘルド・デバイスに前記サーバが送信し記憶するようにさせる手段と、
前記ハンドヘルド・デバイスが前記アプリケーション定義更新情報を記憶するのに充分な利用可能なメモリを有さないと判定された場合、充分な利用可能なメモリを有さないことを示すメッセージを前記ハンドヘルド・デバイスが表示するようにさせる手段と、
ユーザにより前記第２のデータベースになされたトランザクションを前記ハンドヘルド・デバイスが記録するようにさせる手段と、
トランザクション情報を前記サーバに前記ハンドヘルド・デバイスが提供するようにさせる手段であって、前記トランザクション情報は前記記録されたトランザクションの少なくとも一部分を記述している、手段と、そして
前記トランザクション情報に記述されているように前記第１のデータベースに対してトランザクションを前記サーバが実行するようにさせる手段と
を備えるコンピュータ・システム。
【請求項３】
前記アプリケーション更新が、メタデータを含む請求項２に記載のシステム。
【請求項４】
前記ハンドヘルド・デバイスを前記サーバに結合するための前記手段が、前記サーバと前記ハンドヘルド・デバイスに接続されたコンパニオン・デバイスを含む請求項２に記載のシステム。
【請求項５】
前記サーバが前記ハンドヘルド・デバイスにアプリケーションを更新させる手段と、前記抽出されたデータを前記サーバに前記ハンドヘルド・デバイスへ提供させる手段とを含む同期エンジンをさらに含む請求項２に記載のシステム。
【請求項６】
前記同期エンジンが、前記サーバ内に存在する請求項５に記載のシステム。
【請求項７】
前記同期エンジンが、前記サーバと前記ハンドヘルド・デバイスに結合されたコンパニオン・デバイス内に存在する請求項５に記載のシステム。
【請求項８】

前記同期エンジンが、前記ハンドヘルド・デバイス内に存在する請求項５に記載のシステム。
【請求項９】
前記同期エンジンが、データを前記サーバに抽出させる手段も含む請求項５に記載のシステム。
【請求項１０】
前記同期エンジンが、トランザクションを前記サーバに実行させる手段も含む請求項５に記載のシステム。
【請求項１１】
前記トランザクション情報を前記サーバへ前記ハンドヘルド・デバイスに提供させる手段を含む同期マネージャをさらに含む請求項２に記載のシステム。
【請求項１２】
前記同期マネージャが、前記ハンドヘルド・デバイス内に存在する請求項１１に記載のシステム。
【請求項１３】
前記同期マネージャが、前記サーバと前記ハンドヘルド・デバイスに結合されたコンパニオン・デバイス内に存在する請求項１１に記載のシステム。
【請求項１４】
前記同期マネージャが、トランザクションを前記ハンドヘルド・デバイスに記録させる手段も含む請求項１１に記載のシステム。
【発明の詳細な説明】
【関連出願】
【０００１】
（関連出願の相互引用）
本開示は、２００１年９月２８日に出願した"Method and Apparatus For Detecting Insufficient Memory For Data Extraction Processes"（「データ抽出プロセスのために不足のメモリを検出するための方法および装置」）という名称の米国特許出願第０９／９６７４３９号の一部継続出願である。
【技術分野】
【０００２】
本開示は、一般に、コンピュータ・システムに関し、詳細には、それだけに限らないが、１つのコンピュータ・システムから別のコンピュータ・システムにデータを転送することに関する。
【背景技術】
【０００３】
パーム（Ｐａｌｍ）、ハンドスプリング（Ｈａｎｄｓｐｒｉｎｇ）、ヒューレットパッカード（Ｈｅｗｌｅｔｔ　Ｐａｃｋａｒｄ）、ソニー（Ｓｏｎｙ）、カシオ（Ｃａｓｉｏ）、サイオン（Ｐｓｉｏｎ）などのベンダから入手可能な携帯情報端末（ＰＤＡ）など、携帯用コンピューティング・デバイス（本明細書で、ハンドヘルド・デバイスとも呼ぶ）は、ビジネス界においてますます受け入れられている。一部のユーザは、シーベル・システムズ社（Ｓｉｅｂｅｌ　Ｓｙｓｔｅｍｓ，Ｉｎｃ．）、オラクル・コーポレーション他によって提供されるものなど、企業ビジネス・アプリケーションと対話するのに自身のハンドヘルド・デバイスを使用する必要性がある。それらの企業ビジネス・アプリケーションは、何人ものユーザが、いつでもアクセスし、かつ／または更新することができる大型データベース群を含んでいる。
【発明の開示】
【発明が解決しようとする課題】
【０００４】
ハンドヘルド・デバイスを介して企業ビジネス・アプリケーションへアクセスすることは、通常のハンドヘルド・デバイス上で利用できるコンピューティング・パワー、エネル

ギー貯蔵量、メモリの量が比較的限られていることに起因して、大きな問題に直面することがある。例えば、ユーザが、企業ビジネス・アプリケーションをサポートするのに使用されているサーバの中に存在するデータを抽出することを所望することがある。ハンドヘルド・デバイスのリソースが限られていることに鑑みて、ハンドヘルド・デバイスは、抽出されたデータを効率的に受け取るように設計され、構成されることが一般に望ましい。

【課題を解決するための手段】

【０００５】

本発明の態様によれば、システムは、サーバとハンドヘルド・デバイスを含む。ユーザは、ハンドヘルド・デバイス内に存在するアプリケーションを使用してローカル・データベースの中でトランザクションを行うことができる。本発明の一態様では、同期オペレーション中、ハンドヘルド・デバイスとサーバが結合される。次に、システムが、アプリケーションを更新するべきかどうかを判定し、更新すべき場合、サーバに更新を行わせる。一実施態様では、サーバは、メタデータを送信することによって更新を行う。また、システムは、ローカル・データベースに対してユーザが行ったトランザクションに関連する情報を、ハンドヘルド・デバイスがサーバへ提供するようにさせる。システムは、そのトランザクション情報に基づいて、サーバにメイン・データベースに対してトランザクションを実行させる。さらに、システムは、サーバに、メイン・データベースからデータを抽出させる。次に、サーバは、抽出されたデータの少なくとも一部をハンドヘルド・デバイスに提供して、ローカル・データベースを更新する。

【発明を実施するための最良の形態】

【０００６】

本発明の限定的でなく、網羅的ではない実施形態を以下の図を参照して説明する。特に明記しない限り、様々な図のすべてで、同様の符号は、同様の部分を指す。

【０００７】

ハンドヘルド・デバイスをサーバと同期させるためのシステムと方法の実施形態を以下に説明する。以下の説明では、本発明の実施形態を完全に理解させるために多数の具体的な詳細を示す。ただし、本発明は、その具体的な詳細の１つまたは複数なしに、または他の方法、コンポーネント、材料などを使用して実現することができることを当業者なら理解できるであろう。その他、周知の構造、材料、またはオペレーションは、本発明の態様を不明瞭にするのを避けるため、詳細に図示されても述べられてもいない。

【０００８】

システムの概要

図１は、本発明の一実施形態によるハンドヘルド・デバイスを介してユーザがアクセスすることができるデータベースを有するシステム１００を示している。システム１００は、メイン・データベース１２を有するメイン・コンピュータ・システム１１０とそのデータベース１１２に接続されたサーバを含む。第１の可能な実施形態では、サーバは、サーバ１１４として実装される。第２の可能な実施形態では、サーバは、同期エンジン１１８を含むサーバ１１６として実装される。第３の可能な実施形態では、メイン・コンピュータ・システム１１０が、サーバ１１４とサーバ１１６をともに含むことが可能である。サーバ１１４と１１６はともに、便宜上、図１に示している。ただし、前述した第１の実施形態や第２の実施形態は、このサーバのどちらかを省略することもできる。

【０００９】

システム１００は、メイン・データベース１１２に遠隔でアクセスするためにユーザが使用するハンドヘルド・デバイスも含む。ハンドヘルド・デバイス１２０－１、１２０－２、１２０－３、１２０－４、１２０－５を図１に示している。サーバ１１４がシステム１００内に存在する場合、ユーザは、ハンドヘルド・デバイス１２０－１を使用して、サーバ１１４に対する接続１２２を介して、メイン・データベース１１２に遠隔からアクセスすることができる。一実施形態では、接続１２２は、標準の電話モデムや、ハンドヘルド・デバイス製造業者によって定義されたシリアル・インターフェースを使用して実装される。

【００１０】
別法として、ユーザは、接続１２６を介してサーバ１１４に接続された中間コンピュータ・デバイス（本明細書で、コンパニオン・デバイスとも呼ぶ）１２４を介してメイン・データベース１１２に遠隔からアクセスすることもできる。通常、コンパニオン・デバイス１２４は、通常のハンドヘルド・デバイスと比べてより強力なコンピューティング・デバイス（すなわち、より多くのメモリ、より高性能のプロセッサ、より大きい電源などを有する）である。例えば、コンパニオン・デバイスは、デスクトップ・コンピュータまたはノートブック・コンピュータとすることが可能である。ハンドヘルド・デバイス１２０－２とコンパニオン・デバイス１２４は、接続１２７を介して情報を転送する。一実施形態では、接続１２７は、通常、ハンドヘルド・デバイスとともに提供されるシリアル・インターフェースを使用して実装される。例えば、接続１２７は、シリアル・ポートプロトコル、パラレル・ポートプロトコル、またはその他のバス・プロトコルを使用して実装することができる。一部のハンドヘルド・デバイスは、ハンドヘルド・デバイスとコンパニオン・デバイスの間の物理的な相互接続を提供するクレイドル・アセンブリを含む。
【００１１】
この実施形態では、コンパニオン・デバイス１２４は、同期エンジン１２８と同期マネージャ１３０を含む。同期エンジン１２８は、サーバ１１６の同期エンジン１１８と同様の機能を実行する。一部の実施形態では、同期マネージャ１３０と同期エンジン１１８、１２８は、コンパニオン・デバイス１２４またはサーバ１１６の１つまたは複数のプロセッサによって実行されるソフトウェアで実装される。さらに、図１には示していないが、ハンドヘルド・デバイス１２０－１のこの実施形態は、同期エンジン１２８や同期マネージャ１３０と基本的に同一の機能を果たす同期エンジンと同期マネージャを含む。
【００１２】
サーバ１１６がメイン・コンピュータ・システム１１０内に存在する場合、ユーザは、ハンドヘルド・デバイス１２０－３を使用して、サーバ１１６に対する接続１３２を介してメイン・データベース１１２にアクセスすることができる。接続１３２は、接続１２２に関して前述したように、電話モデム接続や、サーバ１１６とハンドヘルド・デバイス１２０－３の両方によってサポートされる他の任意のタイプの接続でよい。図示していないが、ハンドヘルド・デバイス１２０－３のこの実施形態は、同期マネージャ１３０と基本的に同一の機能を提供する同期マネージャを含む。前述したとおり、サーバ１１６を介してメイン・データベース１１２にアクセスするデバイスが独自の同期エンジンを持たなくてもいいように、サーバ１１６が同期エンジン１１８を備えている。
【００１３】
別法として、ユーザは、ハンドヘルド・デバイス１２０－４を使用して、コンパニオン・デバイス１３４を介してメイン・データベース１１２にアクセスすることができる。デバイス１３４は、接続１３６を介してハンドヘルド・デバイスに接続されている。接続１３６は、通常、ハンドヘルド・デバイス・ベンダによって提供されるシリアル・インターフェースである。コンパニオン・デバイス１３４は、接続１３８を介してサーバ１１６に接続されている。この実施形態では、ハンドヘルド・デバイス１２０－４は、ハンドヘルド・デバイス１２０－３と同様に同期マネージャ・コンポーネント（図示せず）を含む。コンパニオン・デバイス１３４は、可能なサーバ１１６とハンドヘルド・デバイス１２０－４の間のデータ転送を可能にするインターフェース・コンポーネント（本明細書で、プロキシとも呼ぶ）１４０を含む。プロキシでなくても良い。例えば、一実施形態では、接続１３８（すなわち、サーバーコンパニオン・デバイス間の接続）は、ＨＴＴＰ（ハイパーテキスト・トランスポート・プロトコル）接続（例えば、インターネット接続）とすることが可能であり、他方、接続１３６（すなわち、コンパニオン・デバイスーハンドヘルド・デバイス間の接続）は、独自のハンドヘルド・デバイス同期接続（例えば、シリアル・バス）とすることが可能である。このため、インターフェース・コンポーネント１４０は、実際上、サーバ１１６とハンドヘルド・デバイス１２０－４の間のプロキシの役割をする。

【００１４】
さらに、ユーザは、ハンドヘルド・デバイス１２０－５を使用して、コンパニオン・デバイス１４４を介してメイン・データベース１１２にアクセスすることができる。デバイス１４４は、接続１４６を介してハンドヘルド・デバイス１２０－５に接続されている。この実施形態では、コンパニオン・デバイス１４４は、接続１５０を介してサーバ１１６と通信する同期マネージャ１４８を含む。コンパニオン・デバイス１４４の同期マネージャ１４８により、ハンドヘルド・デバイス１２０－５は、同期マネージャ・コンポーネントを持つ必要がない。
【００１５】
実際、数名のユーザが、前述した５つのパス（すなわち、メイン・データベース１１２とハンドヘルド・デバイス１２０－１ないし１２０－５の間の通信パス）のそれぞれを介して、ハンドヘルド・デバイスを用いてメイン・データベース１１２にアクセスすることができる。このため、例えば、１つのハンドヘルド・デバイス１２０－３だけが、サーバ１１６に直接に結合されるように図１に示しているが、数名のユーザが、ハンドヘルド・デバイス１２０－３と実質的に同一の形で構成されたハンドヘルド・デバイスを使用して、メイン・データベース１１２に基本的に同一の形でアクセスすることができる。
【００１６】
さらに、前述した５つのパスの様々な組み合わせ、または並べ替えを使用して、システム１００の他の実施形態を実装することもできる。例えば、システム１００は、メイン・データベース１１２とハンドヘルド・デバイス１２０－３（サーバ１１６と直接接続１３２を介して）、およびデータベース１１２とハンドヘルド・デバイス１２０－４（サーバ１１６とコンパニオン・デバイス１３４を介して）の間のパスだけをサポートするように実装することができる。この典型的な実施形態により、同期マネージャを備えるように構成されたハンドヘルド・デバイスは、直接同期またはコンパニオン同期（プロキシ１４０を介して）を実行することができる。
【００１７】
さらに、電話モデム接続、ＨＴＴＰ接続、標準のハンドヘルド同期接続について前述したが、他の実施形態では、任意の適切な接続を使用することができる。例えば、他の実施形態は、ＨＴＴＰ以外のプロトコルを使用することができる。さらに、この接続によって使用される信号伝搬媒体は、有線（例えば、より対線、ケーブル、光ファイバなどの媒体を使用する）であっても、無線（例えば、赤外線技術、無線周波数技術、光技術などの技術を使用する）であってもよい。
【００１８】
システム１００の機能の１つは、メイン・コンピュータ・システム１１０とハンドヘルド・デバイスの間で選択された情報を同期させることである。例えば、情報は、メイン・データベース１１２や、ハンドヘルド・デバイスのローカル・データベース（図示せず）の中に格納されたデータベースのデータとすることが可能である。システム１００のオペレーション中、データベースのデータは、ユーザによって頻繁に更新される。更新されたデータベースのデータは、同期プロセスを介してユーザに配信される。さらに、同期されるべき情報には、ハンドヘルド・デバイスで実行されるアプリケーションによって使用される定義（本明細書で、メタデータとも呼ぶ）が含まれることもある。同期プロセス中に実行されるオペレーションのいくつかを図２に関連して以下に説明する。
【００１９】
図２は、本発明の一実施形態によるデータを同期させる際のハンドヘルド・デバイス２０２と他のコンピュータ・システム２０４の間におけるデータフローを示している。例えば、コンピュータ・システム２０４は、（ａ）サーバ１１４または１１６のどちらか、または（ｂ）コンパニオン・デバイス１２４、または（ｃ）サーバ１１４または１１６のどちらかと結合されたコンパニオン・デバイス１３４、または（ｄ）サーバ１１４または１１６のどちらかと結合されたコンパニオン・デバイス１４４によって実現される。図２のデータフロー・オペレーションの一部の特定の実施形態を図３～１１に関連して以下に説

明する。

【００２０】

図２を再び参照すると、ハンドヘルド・デバイス２０２が、図２に矢印２１０で示したコンピュータ・システム２０４との接続を開始する。一実施形態では、ハンドヘルド・デバイス２０２は、モデムを使用してログインすることにより（例えば、インターネット接続を介して）、コンピュータ・システム２０４とのこの接続を開始する。例えば、この接続は、標準のＨＴＴＰ接続とすることが可能である。別の実施形態では、ハンドヘルド・デバイス２０２は、ハンドヘルド・デバイス・ベンダによって提供される同期インターフェース・アプリケーションやハードウェアを使用して、コンパニオン・デバイスに対してこの接続を開始させることができる。例えば、ユーザが、ハンドヘルド・デバイス２０２をクレイドル・アクセサリに入れて、クレイドル上の同期ボタンをアクティブにしたときにこの接続が開始するようにすることが可能である。

【００２１】

この実施形態では、コンピュータ・システム２０４が、次に、図２に矢印２１２で示されているように、ハンドヘルド・デバイス２０２に初期設定データを提供する。一実施形態では、初期設定情報は、メイン・データベース１１２の最新バージョンに関連する情報を含むことが可能である。例えば、新規のテーブルが、メイン・データベース１１２に追加されていることがある。さらに、初期設定情報は、コンピュータ・システム２０４にアップロードされた最新のトランザクション（以下の説明を参照）に関連する情報を含ませることもできる。一実施形態では、接続が確立された後、この初期設定情報をハンドヘルド・デバイス２０２がプルする。別法として、コンピュータ・システム２０４が、接続が確立されたことに応答して初期設定情報をハンドヘルド・デバイス２０２にプッシュするようにしてもよい。

【００２２】

図２の矢印２１３で示すとおり、コンピュータ・システム２０４が、次に、アプリケーション定義情報をハンドヘルド・デバイス２０２に転送する。一実施形態では、このアプリケーション定義情報には、ハンドヘルド・デバイス２０２のローカル・データベースにアクセスするためにユーザが使用するアプリケーション（本明細書で、ハンドヘルド・アプリケーションとも呼ぶ）において使用される定義に関連する情報を含む。例えば、このアプリケーション定義情報には、ユーザ・インターフェースを提供する際にハンドヘルド・アプリケーションによって表示されるビューとスクリーンとを含むことが可能である。一実施形態では、ハンドヘルド・デバイス２０２は、自らのローカル・アプリケーション定義を更新する必要があるかどうかを判定し（初期設定情報を使用して）、必要がある場合、上記の情報をコンピュータ・システム２０４からプルする。他の実施形態では、コンピュータ・システム２０４は、ハンドヘルド・デバイス２０４から、ハンドヘルド・デバイス２０２のアプリケーション定義のバージョンを示す情報を受け取ることができる。次に、コンピュータ・システム２０４は、ハンドヘルド・デバイス２０２が更新されたアプリケーション定義情報を必要とするかどうかを判定し、必要とする場合、更新されたアプリケーション定義をハンドヘルド・デバイス２０２にプッシュする。

【００２３】

ハンドヘルド・デバイス２０２は、図２の矢印２１４で示すとおり、トランザクション情報をコンピュータ・システム２０４に転送する。一実施形態では、ユーザによってハンドヘルド・デバイス２０２に入力されたローカル・データベース・トランザクションのすべてが記録される。この記録されたトランザクション情報は、コンピュータ・システム２０４に転送され、次に、コンピュータ・システム２０４が、メイン・データベース１１２（図１）の中でトランザクションを実行する。一部の実施形態では、ハンドヘルド・デバイス２０２は、トランザクション情報を１つのブロックとしてコンピュータ・システム２０４に転送する。ブロック全体を受け取った後、コンピュータ・システム２０４は、ブロックが適切に受け取ったかどうかを判定する。他の実施形態では、ハンドヘルド・デバイス２０２は、いくつかの比較的小さいブロックでトランザクション情報を転送する。それ

ぞれの小さいブロックを受け取った後、コンピュータ・システム２０４は、そのブロックを適切に受け取ったかどうかを判定し、そのブロックを再送するか、または次のブロックを送るようにハンドヘルド・デバイス２０２にメッセージまたは信号を送る。このため、転送中に問題または中断が生じた場合、ハンドヘルド・デバイス２０２は、トランザクション情報のすべてを再送するのではなく、適切に受け取られなかったブロックを転送するだけでよい。

【００２４】

次に、コンピュータ・システム２０４は、図２の矢印２１６で示すとおり、エラー情報をハンドヘルド・デバイス２０２に転送する。一実施形態では、このエラー情報は、（ａ）システム１００（図１）は、ユーザが、１名または複数名の他のユーザによって変更されたデータに対してトランザクションを行うことを許さないというトランザクションに関する情報、および（ｂ）サーバによってトランザクションに対して行われた変更に関する情報を含む。次に、ユーザが、ハンドヘルド・デバイス２０２上でそれらのエラーを手動で訂正する、または処理することができる。

【００２５】

さらに、ハンドヘルド・デバイス２０２とコンピュータ・システム２０４は、図２の矢印２１８で示すとおり、フィルタ設定を更新することができる。フィルタ設定は、不要な情報または不必要な情報が、ハンドヘルド・デバイス２０２とコンピュータ・システム２０４の間で転送されず、ハンドヘルド・デバイス２０２上の限られたリソースが節約されるように設定される。この実施形態では、ユーザは、ハンドヘルド・デバイス２０２のフィルタ設定を更新した後、その設定をコンピュータ・システム２０４にアップロードすることができる。このため、コンピュータ・システム２０４は、その後、ユーザが望まない情報をダウンロードすることを回避することができる。さらに、この実施形態では、コンピュータ・システム２０４は、フィルタ設定が確実に適切であるように、ハンドヘルド・デバイス２０２から受け取ったフィルタ設定を処理する。例えば、適切に実行しようとして、ハンドヘルド・デバイス２０２で現在実行されているアプリケーションによって要求された情報を、ユーザがフィルタリングして除こうと試みている可能性がある。

【００２６】

次に、コンピュータ・システム２０４は、図２の矢印２２０で示すとおり、データベースのデータをハンドヘルド・デバイス２０２に転送する。このオペレーションを本明細書では、「データ抽出」とも呼ぶ。一実施形態では、コンピュータ・システム２０４は、ハンドヘルド・デバイス２０２に可視であり、フィルタリングされた後、データベースのデータのすべてのイメージを形成する。このイメージを本明細書では抽出とも呼ぶ。次に、コンピュータ・システム２０４は、この抽出をハンドヘルド・デバイス２０２にダウンロードする。一実施形態では、コンピュータ・システム２０４は、抽出を一連の比較的小さいブロックでダウンロードし、ハンドヘルド・デバイス２０２が、それぞれの小さいブロックの受け取りを確認する。

【００２７】

別の実施形態では、コンピュータ・システム２０４は、各ダウンロードの後、抽出を格納する。次のデータ抽出オペレーション時に、コンピュータ・システム２０４は、現在の抽出を以前の抽出と比較し、変更されているデータベースのデータだけをダウンロードする（本明細書で、デルタ抽出とも呼ぶ）。次に、以前の抽出を削除することができる。さらなる改良形態では、ある状況において、コンピュータ・システム２０４は、以前の抽出を無視し、代わりに、現在の抽出全体をダウンロードすることができる。例えば、以前の抽出以降、メイン・データベース１１２（図１）の構造が変化している場合、コンピュータ・システム２０４は、デルタ抽出を実行しようと試みるのではなく、完全抽出を実行する。次に、ハンドヘルド・デバイス２０２が、図２の矢印２２２で示すとおり、コンピュータ・システム２０４からの接続を解除する。

【００２８】

図３は、本発明の一実施形態による同期オペレーションにおいて使用される同期エンジ

ン１１６（図１）とハンドヘルド・デバイス３００のコンポーネントを示している。あらゆるハンドヘルド・デバイス１２０－３ないし１２０－５（図１）を実装するためにハンドヘルド・デバイス３００を使用することができる。

【００２９】

この実施形態では、同期エンジン１１６は、メタデータ・エクストラクタ３０１、トランザクション・プロセッサ３０３、データ・エクストラクタ３０５を含む。ハンドヘルド・デバイス３００は、ローカル・データベース３０８を含み、かつメタデータ・インポータ３１１、トランザクション・レコーダ３１３、データ・インポータ３１５を有する同期クライアント（同期クライアント）３１０を含む。この実施形態では、同期クライアント３１０とそのコンポーネントはソフトウェアで実装される。

【００３０】

同期エンジン１１６と、同期クライアント３００の前述した要素は、次のように互いに接続される。同期エンジン１１６のメタデータ・ジェネレータ／エクストラクタ３０１が、破線３１７で示すとおり、同期クライアント３１０のメタデータ・インポータ３１１に動作上、結合される。同期エンジン１１６のトランザクション・プロセッサ３０３が、破線３１９で示すとおり、同期クライアント３１０のトランザクション・レコーダ３１３に動作上、結合される。同期エンジン１１６のデータ・エクストラクタ３０５が、破線３２１で示すとおり、同期クライアント３１０のデータ・インポータ３１５に動作上、結合される。同期クライアント３１０は、線３２３で示すとおり、ローカル・データベース３０８にアクセスすることができる。以上のコンポーネントは、以下とおり動作する。

【００３１】

この実施形態では、メタデータ・ジェネレータ／エクストラクタ３０１の主要な機能の一部は、同期クライアント３１０が、更新されたメタデータを必要とするかどうかを判定し、サーバ１１６上に格納されたメタデータを抽出し、抽出されたメタデータをハンドヘルド・デバイス３００に転送することである。メタデータは、ローカル・データベース３０８にアクセスするのに使用されるハンドヘルド・アプリケーション（図示せず）のためのスクリーン、ビュー、フィールドなどの定義を含む。メタデータ・ジェネレータ／エクストラクタ３０１は、メタデータ（サーバ１１６の中に特定のフォーマットで格納された）を抽出し、同期クライアント３１０に伝送するために、そのメタデータを含むメッセージまたはデータグラムを形成する。

【００３２】

同期クライアント３１０のメタデータ・インポータ３１１は、メタデータ・ジェネレータ／エクストラクタ３０１にから送られたメタデータを処理して、ハンドヘルド・デバイス３００内部のハンドヘルド・アプリケーションを更新する。例えば、一実施形態では、メタデータ・インポータ３１１は、メタデータをダウンロードするように同期エンジン１１６に要求する前に、ハンドヘルド・デバイスが、そのメタデータを格納するのに十分なメモリを有するかどうかを判定する。ハンドヘルド・デバイス３００がメタデータを格納した後、次に、メタデータ・インポータ３１１が、メタデータの中に含まれる新しいアプリケーション定義を使用してハンドヘルド・アプリケーション（図示せず）を更新することができる。このオペレーションの一実施形態を図９に関連して以下でより詳細に説明する。

【００３３】

この実施形態では、ハンドヘルド・デバイス３００内部のトランザクション・レコーダ３１３が、ユーザによって行われたローカル・データベース３０８に対するトランザクションに関連する情報を記録する。例えば、ユーザが、ローカル・データベース３０８の中のデータを変更するたびに毎回、トランザクション・レコーダ３１３は、トランザクション識別子（トランザクションＩＤ）を割り当て、そのトランザクションＩＤと、トランザクションに関する他の関連する情報を記録する。別の実施形態では、トランザクションＩＤは、同期プロセスが実行されるときに割り当てることができる。前述した他の関連する情報は、例えば、メイン・データベース１１２（図１）上で変更されたレコードを見つけ

る、または新規のレコードを作成するためにサーバ１１６によって使用される、変更されているフィールド、以前のデータと新規のデータ、レコード識別子とレコード名、さらにレコード関係の明細を含む。次に、トランザクション・レコーダ３１３は、トランザクションを同期エンジン１１６のトランザクション・プロセッサ３０３にアップロードすることができる。

【００３４】

トランザクション・プロセッサ３０３は、ハンドヘルド・デバイス３００からトランザクションを受け取り、各トランザクションを実行してメイン・データベース１１２（図１）を更新する。トランザクションは、別のユーザからの別のトランザクションと矛盾する可能性があり、矛盾している場合、トランザクション・プロセッサ３０３は、そのトランザクションを実行することなしに、エラーを報告する。このオペレーションの一実施形態を図７に関連して以下でより詳細に説明する。別の実施形態では、トランザクション・プロセッサ３０３は、トランザクションが成功するようにトランザクションの１つまたは複数の部分を変更する。次に、トランザクション・プロセッサ３０３は、実行された変更をユーザに知らせるメッセージをハンドヘルド・デバイス３００に送ることができる。

【００３５】

この実施形態では、同期エンジン１１６のデータ・エクストラクタ３０５は、メイン・データベース１１２（図１）からハンドヘルド・デバイス３００に可視であるデータベースのデータを抽出する。可視性という用語は、データベースのリモート・アクセスという周知の意味を有する（例えば、米国特許第６２１６１３５号および第６２３３６１７号参照）。さらに、この実施形態では、データ・エクストラクタ３０５は、フィルタ設定（図２の矢印２１８に関連して前述した）に従ってデータを抽出することを避けることができる。また、データ・エクストラクタ３０５は、抽出済みのデータベースのデータをハンドヘルド・デバイス３００にダウンロードされるファイルに形成する。一実施形態では、同期エンジン１１６は、一連の小さいメッセージまたはデータグラムでファイルをハンドヘルド・デバイス３００に送ることができる。

【００３６】

ハンドヘルド・デバイス３００のデータ・インポータ３１５が、そのファイルを受け取り、ローカル・データベース３０８を更新するためにそのファイルを一時的に格納する。以下に説明するとおり、このファイルは、更新されたデータベースのデータをローカル・データベース３０８のものとは異なるフォーマットで含むことが可能である。そのような場合、別個のコンポーネントを使用してそのデータを処理して、ローカル・データベース３０８のフォーマットにすることができる。

【００３７】

ハンドヘルド・デバイス

図４は、本発明の一実施形態によるハンドヘルド・デバイス１２０－３（図１）を示している。この実施形態では、ハンドヘルド・デバイス１２０－３は、サーバ１１６と直接に同期するように構成されている。ローカル・データベース３０８に加え、ハンドヘルド・デバイス１２０－３は、同期クライアント４０１、同期ログ（同期ログ）４０３、トランザクション・データベース４０５、データ格納アプリケーション（本明細書で、データ・ストアラとも呼ぶ）４０７、データ・ストア４０９を有している。この実施形態では、同期クライアント４０１は、コンパニオン・デバイス内に存在する同期マネージャ１３０と１４８（図１）に関して説明したのと実質的に同様の諸機能を実行する。特に、同期クライアント４０１は、同期クライアント３１０（図３）のメタデータ・インポータ３１１、トランザクション・レコーダ３１３、データ・インポータ３１５の機能を実行する。

【００３８】

同期ログ４０３は、同期オペレーション（例えば、図２のオペレーションを参照）中に送られたすべてのメッセージのリストを表示するために使用され、このリストは、後に、同期中に問題が生じた場合、情報を復元するのに使用することができる。トランザクション・データベース４０５は、各トランザクションに関連する情報（例えば、図３のトラン

ザクション・レコーダ３１３によって生成された情報）を格納するのに使用される。一実施形態では、同期マネージャが、トランザクション情報をトランザクション・データベース４０５の中に格納する。別法として、ハンドヘルド・アプリケーション（図示せず）が、トランザクション情報をトランザクション・データベース４０５の中に格納する。この実施形態のデータ・ストアラ４０７は、ダウンロード済みのデータベースのデータをローカル・データベース３０８のフォーマットになるように処理するのに使用される。データ・ストア４０９は、フィルタ設定、ローカル・データベース３０８のバージョン、ハンドヘルド・アプリケーション（図示せず）のバージョン、ローカル・データベースのスキーマを定義するファイル、メタデータからの前述したアプリケーション定義を格納するのに使用される。また、データ・ストア４０９は、トランザクション・エラーメッセージを格納するのにも使用される。この実施形態では、ローカル・データベースとハンドヘルド・アプリケーションのバージョンはそれぞれ、本明細書で抽出ＩＤおよびリポジトリＩＤと呼ぶ。一部の実施形態では、データ・ストア４０９は、オペレーティング・システムのレジストリ（例えば、Ｗｉｎｄｏｗｓ（登録商標）オペレーティング・システムまたはＬｉｎｕｘオペレーティング・システムにおけるような）を含むことが可能である。

【００３９】

同期クライアント４０１が、ローカル・データベース３０８、同期ログ４０３、トランザクション・データベース４０５、データ・ストアラ４０７、データ・ストア４０９に結合されている。また、同期クライアント４０１は、接続１３２を介してサーバ１１６（図１）にも結合されている。さらに、データ・ストアラ４０７が、ローカル・データベース３０８に結合されている。ハンドヘルド・デバイス１２０－３のこの実施形態のオペレーションを図６に関連して以下で説明する。

【００４０】

図５は、本発明の一実施形態によるハンドヘルド・デバイス１２０－２（図１）とコンパニオン・デバイス１２４（図１）のコンポーネントを示している。ハンドヘルド・デバイス１２０－２とコンパニオン・デバイス１２４のこの実施形態は、ハンドヘルド・デバイス１２０－３（図４）において前述したものと同様のコンポーネントを含む。詳細には、ハンドヘルド・デバイス１２０－２は、ローカル・データベース３０８、トランザクション・データベース４０５、データ・ストアラ４０７、データ・ストア４０９を有している。コンパニオン・デバイス１２４は、同期クライアント５０１と同期ログ５０３を有し、同期クライアント５０１と同期ログ５０３はともに、ハンドヘルド・デバイス１２０－３の同期クライアント４０１（図３）および同期ログ４０３（図３）のものと同様の機能を実行する。さらに、コンパニオン・デバイス１２４は、同期エンジン５０５とコンパニオン・ローカル・データベース５０８を有している。クライアント同期エンジン５０５は、メイン・データベース１１２（図１）にアクセスする際、同期エンジン１１８（図１）のものと同様の諸機能を提供する。

【００４１】

この実施形態では、同期クライアント５０１は、ハンドヘルド・デバイス１２０－２のローカル・データベース３０８、トランザクション・データベース４０５、データ・ストアラ４０７、データ・ストア４０９に結合されている。この相互接続は、接続１２７（図１）を介して実施することができる。さらに、同期クライアント５０１は、同期ログ５０３、クライアント同期マネージャ５０５、コンパニオン・ローカル・データベース５０８に結合される。一実施形態では、コンパニオン・ローカル・データベース５０８は、ハンドヘルド・デバイス１２０－２のローカル・データベース４０５のイメージを格納することができる。そのような実施形態では、コンパニオン・デバイス１２４は、コンパニオン・ローカル・データベース５０８をメイン・データベース１１２（図１）と同期させるように構成することができる。ハンドヘルド・デバイス１２０－２のその後の同期が、ハンドヘルド・デバイス１２０－２のローカル・データベース３０８を現時点で同期済みのコンパニオン・ローカル・データベース５０８と同期させる。

【００４２】

同期クライアント・プロセス

図６は、本発明の一実施形態による同期プロセス中の同期クライアントのオペレーションを示している。この実施形態では、同期クライアントは、サーバへの直接接続を有するハンドヘルド・デバイスである。例えば、ハンドヘルド・デバイスとサーバは、ハンドヘルド・デバイス１２０－３（図４）とサーバ１１６（図１）とすることが可能である。別法として、同期クライアントは、コンパニオン・デバイスとハンドヘルド・デバイス（例えば、図５におけるようなコンパニオン・デバイス１２４とハンドヘルド・デバイス１２０－２）の組み合わせによって実装することが可能である。便宜上、同期クライアントのオペレーションをハンドヘルド・デバイス１２０－３（図４）とサーバ１１６（図１）に関連させて説明する。本開示に鑑みて、以下の説明は、必要以上に実験を行うことなしに、他のタイプの同期クライアントにも当てはまることは当業者には理解できるであろう。

【００４３】

図４、図６を参照すると、同期クライアントの一実施形態は、次のとおり動作する。ブロック６０１で、同期クライアントを、サーバ（他の実施形態ではコンパニオン・デバイス）に接続する。一実施形態では、同期クライアント４０１は、ユーザがログイン・プロセスを実行した際にサーバ１１６に接続される。一実施形態では、同期クライアント４０１は、サーバ１１６に直接にアクセスするようにモデムを動作させるインターフェースまたはドライバを実行する。他の実施形態では、この接続は、インターネット接続とすることが可能である。さらに他の実施形態では、接続は、ＲＦ技術または光技術を使用して、直接の通信リンク、Ｗｅｂベースの通信リンク、または他のタイプの通信リンクを実施する無線接続とすることが可能である。

【００４４】

ブロック６０３で、同期クライアントは初期設定データを受け取る。一実施形態では、同期クライアント４０１が、接続１３２を介してサーバ１１６から初期設定データを受け取る。図２に関連して前述したとおり、この初期設定データは、メイン・データベース１１２の最新バージョン（例えば、抽出ＩＤ）、サーバ１１６にアップロードされた最新のトランザクション、さらにハンドヘルド・アプリケーションの最新バージョン（例えば、リポジトリＩＤ）に関連する情報を含むことが可能である。この実施形態では、同期クライアント４０１は、初期設定情報をデータ・ストア４０９の中に格納する。

【００４５】

ブロック６０５で、同期クライアントはアプリケーション定義バージョンを受け取る。ブロック６０５は、ハンドヘルド・アプリケーション定義がブロック６０３で受け取った初期設定データの一部として含まれる場合、省略することができる。一実施形態では、サーバ１１６が、アプリケーション定義バージョン（更新されていない可能性がある）を同期クライアント４０１に送る。同期クライアント４０１は、新たに受け取ったアプリケーション定義バージョンをデータ・ストア４０９の中に既に格納されているアプリケーション・バージョン（すなわち、ハンドヘルド・アプリケーションの現行バージョン）と比較することができる。次に、同期クライアント４０１は、新規のアプリケーション定義バージョンをデータ・ストア４０９の中に格納する。

【００４６】

ブロック６０７で、同期クライアントはトランザクション情報をサーバまたはコンパニオン・デバイスに提供する。一実施形態では、同期クライアント４０１が、トランザクション・データベース４０５の中に格納されたトランザクション情報をサーバ１１６に送る。例えば、同期クライアント４０１は、記録済みのトランザクションのすべてに関するトランザクション情報を１つのデータ・ブロックで送ることができる。別法として、同期クライアント４０１は、トランザクション情報をいくつかのより小さいブロックで送り、次のより小さいブロックを送る前に、サーバ１１６からの受取確認を待つようにしても良い。このオペレーションの実施形態を図７、図８に関連して以下により詳細に説明する。

【００４７】

ブロック６０９で、同期クライアントはサーバからメタデータを取得する。一実施形態

では、同期クライアント４０１がサーバ１１６からメタデータを受け取る。前述したとおり、このメタデータは、ハンドヘルド・アプリケーションに関するアプリケーション定義を含む。この実施形態では、同期クライアント４０１は、ブロック６０５で受け取ったアプリケーション定義バージョンが、ローカルに格納されているアプリケーション定義バージョンと一致しない場合（例えば、この状況は、アプリケーション定義が更新されている場合に生じる可能性がある）、ブロック６０９を実行する。このオペレーションの一実施形態を図９に関連して以下でより詳細に説明する。別の実施形態では、同期クライアント４０１は、アプリケーション定義バージョンの比較を行わず、代わりに、サーバ１１６からメタデータを常に要求する。次に、サーバ１１６が、アプリケーション定義を更新する必要があるかどうかを判定する。

【００４８】

ブロック６１１で、同期クライアントは自らのローカル・データベースを更新する。一実施形態では、同期クライアント４０１が、サーバ１１６が、データ抽出オペレーションを開始して、更新されたデータベースのデータをハンドヘルド・デバイス１２０－３に提供することを要求する。この実施形態では、ユーザは、同期クライアント４０１に、サーバ１１６からデータベースのデータを取得する前に、フィルタを更新させることができる。このため、サーバ１１６は、不要なデータベースのデータをダウンロードすることを回避する。これは、電池の電力を節約するのに役立ち、同期プロセスを完了するのに必要な時間を短縮することができる。一実施形態では、サーバ１１６は、データ抽出全体（すなわち、同期クライアントに可視であり、フィルタリングした後のデータ）を単一の大きいブロックでダウンロードする。代替の実施形態では、サーバ１１６は、同期クライアント４０１による要求に応答して、データ抽出の比較的小さいブロックをダウンロードすることができる。ハンドヘルド・デバイス１２０－３が、このブロックを適切に受け取った場合、同期クライアント４０１は、次のブロックを求める要求を送ることができ、ハンドヘルド・デバイス１２０－３が、サーバ１１６から抽出全体を受け取るまで、以下同様に続ける。

【００４９】

さらなる改良形態では、同期クライアント４０１はまず、完全抽出を受け取るか、またはデルタ抽出を受け取るかを決めることができる。例えば、データベースのバージョンが変更されている場合（ブロック６０３を参照）、同期クライアント４０１は、サーバ１１６から完全抽出を要求することができる。しかし、データベースのバージョンが変更されていない場合、同期クライアント４０１は、サーバ１１６からデルタ抽出を要求することができる。例えば、デルタ抽出では、サーバ１１６は、ハンドヘルド・デバイス１２０－３への以前のダウンロードの完全抽出を現在の完全抽出と比較する。次に、サーバ１１６は、以前の抽出の中の対応するレコードとは異なる現在の完全抽出からのレコードだけをダウンロードする。

【００５０】

ブロック６１３で、同期クライアントは、サーバ（またはコンパニオン・デバイス）から切り離される。この実施形態では、ユーザからの入力に応答して、同期クライアント４０１は、ログアウト・プロセスを実行してサーバ１１６から切り離される。

【００５１】

図７は、本発明によるブロック６０７（図６）の一実施形態をより詳細に示している。同期クライアントが、図７のオペレーションを実行する。この実施形態では、同期クライアントは、サーバ１１６との直接接続を有するハンドヘルド・デバイス１２０－３内に存在する。図６の説明の場合と同様に、ブロック６０７のトランザクション処理オペレーションを、ハンドヘルド・デバイス１２０－３（図４）とサーバ１１６（図１）に関連させて説明する。ただし、本開示に鑑みて、以下の説明は、必要以上に実験を行うことなしに、他のタイプの同期クライアントにも当てはまることは当業者には理解できるであろう。図４、図７を参照すると、ブロック６０７は、この典型的な実施形態において以下のとおり実行される。

【００５２】
ブロック７０１で、同期クライアントはサーバまたは（コンパニオン・デバイス）にアップロードされた最新のトランザクションに関連する情報を受け取る。一実施形態では、同期クライアント４０１が、サーバ１１６からトランザクション識別子（トランザクションＩＤ）を取得する。このトランザクションＩＤは、サーバ１１６によって受け取られた最新のトランザクションの識別子である。一実施形態では、ハンドヘルド・デバイス１２０－３は、初期設定された際、擬似ユニークＩＤジェネレータを使用してトランザクションＩＤを生成する。別の実施形態では、ＩＤ生成は、ユニークであることが保証される。次に、同期クライアント４０１は、トランザクションに関して記録された情報をハンドヘルド・デバイス１２０－３がサーバ１１６にアップロードする（本明細書で、「トランザクションをアップロードする」という言い方もする）たびに毎回、この擬似ユニークＩＤを増分する。この実施形態では、同期クライアント４０１は、次に、受け取ったトランザクションＩＤをトランザクション・データベース４０５の中のトランザクション情報と比較して、未処理のトランザクション情報（すなわち、サーバ１１６から受け取ったトランザクションＩＤより大きいトランザクションＩＤを有するトランザクション）を送る。

【００５３】
ブロック７０３で、同期クライアントはアップロードされるべき未処理のトランザクションを圧縮する。一実施形態では、同期クライアント４０１が、任意の適切な圧縮アルゴリズムを使用してトランザクション情報を圧縮する。次に、同期クライアント４０１は、圧縮済みの情報をハンドヘルド・デバイス１２０－３とサーバ１１６の間の接続によってサポートされているフォーマット（例えば、ＨＴＴＰ接続で使用するためのテキスト）に変換することもできる。代替の実施形態では、ブロック７０３は、省略してもよい（すなわち、情報を圧縮する必要はない）。

【００５４】
ブロック７０５で、同期クライアントは、次に、未処理のトランザクションの情報をアップロードする。一実施形態では、同期クライアント４０１が、トランザクション情報をサーバ１１６に単一の大きいブロックで送る。別法として、同期クライアント４０１は、トランザクション情報を一連のより小さいブロックで送ることができる。１つのそのような実施形態では、サーバ１１６は、ハンドヘルド・デバイス１２０－３からそれぞれのより小さいブロックを適切に受け取った後、受取確認を提供する。このオペレーションの一実施形態を図８に関連して以下でより詳細に説明する。

【００５５】
ブロック７０７で、同期クライアントは、ブロック７０５中に生じたエラー（存在する場合）に関する情報を取得する。一実施形態では、サーバ１１６は、トランザクションを処理している際に生じたエラーのすべてを記録に残す。例えば、エラーは、トランザクションが、ハンドヘルド・デバイス１２０－３のユーザより前に同期を行った別のユーザによってより新しく更新されたフィールドを更新しようと試みたことである。他の例には、ハンドヘルド・アプリケーションが課していない、サーバが課している他のデータ検証規則にトランザクションが違反した場合に生じるエラーが含まれる。

【００５６】
ブロック７０９で、同期クライアントは、次に、トランザクション・エラーを処理する。一実施形態では、同期クライアント４０１が、手動でエラーを訂正するようにユーザに促す。別の実施形態では、エラーは、ユーザが見ることを選択することができる別個のエラー・スクリーン上で提供され、手動で訂正されるか、または無視される。

【００５７】
図８は、本発明の一実施形態によるブロック７０５（図７）をより詳細に示している。同期クライアントが図８のオペレーションを実行する。この実施形態では、同期クライアントは、サーバ１１６に対する直接接続を有するハンドヘルド・デバイス１２０－３内に存在する。図６の説明の場合と同様に、ブロック７０５のトランザクション処理オペレーションを、ハンドヘルド・デバイス１２０－３（図４）とサーバ１１６（図１）に関連さ

せて説明する。ただし、本開示に鑑みて、以下の説明は、必要以上に実験を行うことなしに、他のタイプの同期クライアントにも当てはまることは当業者には理解できるであろう。図４、図８を参照すると、ブロック７０５は、この典型的な実施形態において以下のとおり実行される。

【００５８】

ブロック８０１で、ブロック７０１（図７）で受け取ったトランザクションＩＤが、トランザクション・データベース４０５（図４）の中に格納されたかどうかを同期クライアント４０１が判定する。受け取ったトランザクションＩＤがトランザクション・データベース４０５の中にある場合、その受け取ったトランザクションＩＤに対応するトランザクションより前に作成されたトランザクション・データベース４０５の中のトランザクション・レコードのすべてが、サーバ１１６によって既に受け取り済みであり、したがって、もはや必要とされない。一実施形態では、同期クライアント４０１は、ブロック８０３を実行する。ブロック８０３で、同期クライアント４０１は、受け取ったトランザクションＩＤより小さい（すなわち、前に作成された）トランザクションＩＤを有するトランザクション・データベース４０５の中のトランザクション・レコードのすべてを削除する。

【００５９】

しかし、ブロック８０１で、同期クライアント４０１が、受け取ったトランザクションＩＤがトランザクション・データベース４０５の中にないと判定した場合、トランザクション・データベース４０５の中に残っているトランザクション・レコードのいずれも、サーバ１１６によって適切に受け取られていない。ブロック８０５で、同期クライアント４０１は、トランザクション・データベース４０５の中に格納されている未処理のトランザクション・レコードの数を特定する。一実施形態では、この数は、変数（本明細書で、便宜上、ＴＸＮＣＯＵＮＴと呼ぶ）の中に格納される。ブロック８０５も、ブロック８０３の完了後、実行される。

【００６０】

ブロック８０７で、同期クライアント４０１は、処理済みのトランザクション・レコードの数が、ＴＸＮＣＯＵＮＴ未満であるかどうかを判定する。つまり、ブロック８０５後、ブロック７０５の現在の実行における処理済みのトランザクション・レコードの数は、ゼロに設定される。トランザクション・データベース４０５の中の各トランザクション・レコードが処理される（例えば、サーバ１１６にアップロードされる）につれ、処理済みのトランザクション・レコードの数は、増分される。処理済みのトランザクション・レコードの数が、ＴＸＮＣＯＵＮＴの値に達した時点で、トランザクション・データベース４０５の中のトランザクション・レコードのすべてが処理済みとなる。その場合、プロセスは、ブロック８０９に進み、トランザクション・データベース４０５の中に格納されたトランザクション・レコードのすべてが削除され、ブロック７０５が終了する。

【００６１】

しかし、処理済みのトランザクション・レコードの数が、ＴＸＮＣＯＵＮＴの値未満である場合、プロセスは、ブロック８１１に進み、同期クライアント４０１が、トランザクション・データベース４０５から次のトランザクション・レコードを取得する。

【００６２】

ブロック８１３で、同期クライアント４０１は、トランザクション・レコードをサーバ１１６にアップロードされるべきトランザクション・ストリングまたはトランザクション・メッセージに追加する。一実施形態では、同期クライアント４０１は、トランザクション・レコードをサーバ１１６にアップロードされるべきトランザクション・ストリングの中にパックする。一部の他の実施形態では、トランザクション・レコードは、一連の小さいミニトランザクション・レコードとしてトランザクション・データベース４０５の中に格納することができる（特に、トランザクションが、複雑なトランザクションまたは大きいトランザクションである場合）。ブロック８１３で、同期クライアント４０１が、ミニトランザクション・レコードを既存のトランザクション・ストリングと連結する。

【００６３】

ブロック８１５で、同期クライアント４０１は、トランザクション・レコードがミニトランザクション・レコードであるかどうかを判定する。例えば、一実施形態では、トランザクションの各ミニトランザクション・レコードは、同一のトランザクションＩＤを有する。同期クライアント４０１は、現在のトランザクション・レコードのトランザクションＩＤを前のトランザクション・レコードのトランザクションＩＤと比較することにより、トランザクション・レコードがミニトランザクション・レコードであるかどうかを判定することができる。トランザクション・レコードがミニトランザクション・レコードであった場合、オペレーションの流れは、ブロック８１１にループして戻る。しかし、トランザクション・レコードがミニトランザクションではなかった場合、ブロック８１７が実行される。

【００６４】

ブロック８１７で、この実施形態では、トランザクション・ストリング（ブロック８１３からの）は、ＨＴＴＰ接続を介してサーバ１１６に送ることができるように、ＵＲＬ（ユニバーサル・リソース・ロケータ）符号化される。一実施形態では、同期クライアント４０１が、トランザクション・ストリングをＵＲＬ符号化する。

【００６５】

ブロック８１９で、ＵＲＬ符号化されたトランザクション・ストリングがアップロードされる。一実施形態では、同期クライアント４０１は、符号化済みのトランザクション・ストリングをＨＴＴＰ要求のヘッダの中に入れ、サーバ１１６に送る。トランザクション・ストリングがあまりにも大きい（例えば、２キロバイトより大きい）場合、ストリングは、複数のＨＴＴＰ要求を使用してアップロードされる。トランザクション・ストリングがアップロードされた後、プロセスは、ブロック８０７に戻る。プロセスは、トランザクション・データベース４０５の中のトランザクション・レコードのすべてが処理されるまで、繰り返される。

【００６６】

図９は、本発明の一実施形態によるメタデータを更新するためのブロック６０９（図６）を示している。同期クライアントが、図９のオペレーションを実行する。この実施形態では、同期クライアントは、サーバ１１６（図１）に対する直接接続を有するハンドヘルド・デバイス１２０－３（図４）内に存在する。図６の説明の場合と同様に、ブロック６０９のトランザクション処理オペレーションを、ハンドヘルド・デバイス１２０－３（図４）とサーバ１１６（図１）に関連させて説明する。ただし、本開示に鑑みて、以下の説明は、必要以上に実験を行うことなしに、他のタイプの同期クライアントにも当てはまることは当業者には理解できるであろう。図４と図９を参照すると、ブロック６０９は、この典型的な実施形態において以下のとおり実行される。

【００６７】

ブロック９０１で、同期クライアント４０１が、ローカルに格納されているリポジトリＩＤ（すなわち、データ・ストア４０９の中に格納された）をサーバ１１６からのリポジトリＩＤと比較する（図６のブロック６０５を参照）。ローカルに格納されているリポジトリＩＤは、ハンドヘルド・デバイス１２０－３内に存在するハンドヘルド・アプリケーションのバージョンを示すのに対し、サーバ１１６からのリポジトリＩＤは、ハンドヘルド・デバイス１２０－３が有すべきハンドヘルド・アプリケーションのバージョンを示す（例えば、ハンドヘルド・アプリケーションは、前回にハンドヘルド・デバイス１２０－３が同期されて以降に更新されている可能性がある）。

【００６８】

ブロック９０３で、同期クライアント４０１は、リポジトリＩＤが一致するかどうかを判定する。リポジトリＩＤが一致する場合、ハンドヘルド・デバイス１２０－３は、ハンドヘルド・アプリケーションの正しいバージョンを有しており、ブロック６０９が終了する。しかし、リポジトリＩＤが一致しない場合、ハンドヘルド・デバイス１２０－３は、ハンドヘルド・アプリケーションの正しいバージョンで更新されなければならない。このため、ブロック９０３でリポジトリＩＤが一致しない場合、プロセスは、ブロック９０５

に進む。

【００６９】

ブロック９０５で、同期クライアント４０１はメタデータのサイズを取得する。一実施形態では、同期クライアント４０１がサーバ１１６からメタデータのサイズを取得する。ただし、このサイズは、ダウンロードされ、データ・ストアの中に格納された際に、そのメタデータがハンドヘルド・デバイス１２０－３内部で占めるサイズを表していない。

【００７０】

ブロック９０７で、同期クライアント４０１は、ブロック９０５で受け取ったサイズにスケーリング・ファクタを適用して、実行されたメタデータの予期される最大サイズを特定する。一実施形態では、このスケーリング・ファクタは経験的に決められる。他の実施形態では、スケーリング・ファクタは、構成可能であるか、または動的に決めることができる。このスケーリング・ファクタは、実行されたメタデータの実際のサイズを、確実に予期される最大サイズ以下にする。

【００７１】

ブロック９０９で、同期クライアント４０１は、ハンドヘルド・デバイス１２０－３で利用可能なメモリが、メタデータを格納するのに十分であるかどうかどうかを判定する。一実施形態では、同期クライアント４０１は、ブロック９０７で特定された予期される最大サイズを利用可能なメモリと比較する。

【００７２】

十分な利用可能なメモリが存在する場合、同期クライアント４０１は、ブロック９１１で、サーバ１１６からメタデータを取得する。一実施形態では、同期クライアント４０１は、メタデータをサーバ１１６から１回の転送でプルする。別の実施形態では、同期クライアント４０１は、メタデータをサーバ１１６から一連のより小さい転送でプルする。

【００７３】

ブロック９０９で、十分な利用可能なメモリが存在しない場合、プロセスは、ブロック９１３に進む。ブロック９１３で、同期クライアント４０１は、エラー・ルーチンを実行してブロック６０９を優美に終了する。例えば、同期クライアント４０１は、ハンドヘルド・デバイス１２０－３が、同期プロセスを完了させるのに利用可能なメモリが不十分であるというメッセージをユーザに表示して、ユーザが、不必要なファイルを削除して、同期プロセスを再試行するように示唆する。別の実施形態では、ブロック９１３のエラー・ルーチンは、メモリ空間を開放するようにユーザに促し、ユーザがそうした後、終了するのではなく、ブロック９０９に戻る。

【００７４】

図１０は、本発明の一実施形態によるデータベースのデータを抽出するためのブロック６１１（図６）を示している。同期クライアントが図１０のオペレーションを実行する。この実施形態では、同期クライアントは、サーバ１１６（図１）に対する直接接続を有するハンドヘルド・デバイス１２０－３（図４）内に存在する。図６の説明の場合と同様に、ブロック６１１のトランザクション処理オペレーションを、ハンドヘルド・デバイス１２０－３（図４）とサーバ１１６（図１）に関連させて説明する。ただし、本開示に鑑みて、以下の説明は、必要以上に実験を行うことなしに、他のタイプの同期クライアントにも当てはまることは当業者には理解できるであろう。図４、図１０を参照すると、ブロック６１１は、この典型的な実施形態において以下のとおり実行される。

【００７５】

ブロック１００１で、同期クライアント４０１は、サーバ１１６から抽出ＩＤを取得する。一実施形態では、同期クライアント４０１は、この情報をブロック６０３（図６）で初期設定情報の一部として取得する。この抽出ＩＤは、ハンドヘルド・デバイス１２０－３が有すべきデータベースのバージョンを示す。さらに、ハンドヘルド・デバイス１２０－３は、ハンドヘルド・デバイス１２０－３の中に現在、存在するデータベースのバージョンを示す抽出ＩＤをローカルで格納する（例えば、データ・ストア４０９の中に）。これらの抽出ＩＤは、データベース・アーキテクチャが、前回にハンドヘルド・デバイス１

２０－３が同期されて以降に更新されている場合、異なることがある。例えば、メイン・データベース１１２（図１）が更新されて、フィールドまたはテーブルが追加され、かつ／または削除されていることがある。

【００７６】

ブロック１００３で、同期クライアント４０１はフィルタを処理する。一実施形態では、ユーザが、ハンドヘルド・デバイス１２０－３上のフィルタ設定を変更している可能性がある。前述したとおり、フィルタ設定は、ユーザが所望するデータベースのデータだけをダウンロードするためにサーバによって使用される。このフィルタリングの１つの利点は、フィルタリングすることにより、ハンドヘルド・デバイス１２０－３のメモリの中に入ることが可能なサイズまでダウンロードされるデータの量が縮減されることである。同期クライアント４０１は、サーバ１１６の中に格納されたフィルタ設定をダウンロードし、そのフィルタ設定を使用してローカルで更新されたフィルタ設定を更新することができる。また、同期クライアント４０１は、ユーザが変更したフィルタ設定をサーバ１１６にアップロードすることもできる。サーバは、ユーザが変更したフィルタ設定が妥当な設定であることを確実にする。妥当な新規のフィルタ設定は、データベースのデータをハンドヘルド・デバイス１２０－３にダウンロードする際にサーバによって使用される。このフィルタ設定を図１０Ａに関連して以下でより詳細に説明する。

【００７７】

ブロック１００５で、同期クライアント４０１は、ハンドヘルド・デバイス１２０－３が、サーバ１１６によってダウンロードされるデータベースのデータを格納するのに利用できる十分なメモリを有するかどうかを判定する。一実施形態では、同期マネージャ１１６が、データ抽出のサイズをサーバ１１６から取得する。前述したとおり、サーバ１１６は、ハンドヘルド・デバイス１２０－３に可視であり、ハンドヘルド・デバイスからアップロードされた妥当なフィルタ設定に従ってフィルタリング済みのデータベースのデータの抽出を作成するために、メイン・データベース１１２（図１）にアクセスする。サーバ１１６は、データ抽出のサイズ（完全抽出またはデルタ抽出）を同期クライアント４０１に提供し、次に、同期クライアント４０１が、ハンドヘルド・デバイス１２０－３内部で利用可能な十分なメモリが存在するかどうかを判定する。

【００７８】

十分なメモリが存在する場合、同期クライアント４０１は、ブロック１００７で、サーバ１１６からデータ抽出（完全またはデルタ）をプルする。同期マネージャ４１０が、データ抽出を１回だけの転送で、または一連のより小さい転送でプルすることができる。ブロック１００９で、同期クライアント４０１は、抽出をローカル・データベース３０８の中に格納する。

【００７９】

しかし、ハンドヘルド・デバイス１２０－３が、抽出を格納するのに利用できる十分なメモリを有さない場合、ブロック６１１は終了する。例えば、同期クライアント４０１は、ブロック９１３（図９）と同様のエラー・ルーチンを実行して、ブロック６１１を優美に終了する。

【００８０】

ブロック１０１１で、同期クライアント４０１は、サーバ１１６から新規の抽出ＩＤを取得する。同期クライアント４０１は、ローカルに格納されていた抽出ＩＤが、ブロック１００１でサーバ１１６から受け取った抽出ＩＤと同一である場合、ブロック１０１１を省略することができる。

【００８１】

図１０Ａは、本発明の一実施形態によるブロック１００３（図１０）で説明したフィルタを示している。この実施形態では、フィルタは、メタデータによって定義されたスクリーン（図３および図９に関連して前述した）に関連している。前述したとおり、スクリーンは、ハンドヘルド・デバイスにおいて実行されるハンドヘルド・アプリケーションによって表示されて、ユーザ・インターフェースをハンドヘルド・デバイスのローカル・デー

タベースに提供する。例えば、スクリーンは、ユーザがより容易に使用し、データを理解することができるようにするフォーマット／構成で選択されたグループのデータベースのデータを表示することができる。

【００８２】

図１０Ａに示すとおり、フィルタは、ＳＣＲＥＥＮ＿１ないしＳＣＲＥＥＮ＿Ｍとして示す複数のスクリーンを有する。ハンドヘルド・アプリケーションが実行されているとき、ユーザは、様々なスクリーンを移動させて、ハンドヘルド・デバイスに所望のデータベースのデータを表示させることができる。各スクリーンは、少なくとも１つのビジネス・オブジェクトを有する。例えば、スクリーンＳＣＲＥＥＮ＿１は、ＢＵＳＯＢＪ＿１ないしＢＵＳＯＢＪ＿Ｎとして示すいくつかのビジネス・オブジェクトを有する。ビジネス・オブジェクトは、ハンドヘルド・デバイスによって表示されるべきグループの関連するデータを定義するようにあらかじめ設定されている。スクリーンは、ブロック１００３（図１０）における処理を単純化するように１つのスクリーン当たり１つだけのビジネス・オブジェクトを有するように定義される。

【００８３】

図１１は、本発明の一実施形態によるフィルタを処理するためのブロック１００３（図１０）を示している。この実施形態では、同期クライアントが、ブロック１００３のオペレーションを実行する。この実施形態では、同期クライアントは、サーバ１１６（図１）に対する直接接続を有するハンドヘルド・デバイス１２０－３（図４）の中に存在する。図１０の説明の場合と同様に、ブロック１００３のオペレーションを、ハンドヘルド・デバイス１２０－３（図４）とサーバ１１６（図１）に関連させて説明する。ただし、本開示に鑑みて、以下の説明は、必要以上に実験を行うことなしに、他のタイプの同期クライアントにも当てはまることは当業者には理解できるであろう。図４、図１１を参照すると、ブロック１００３は、この典型的な実施形態において以下のとおり実行される。

【００８４】

ブロック１１０１で、同期クライアント４０１が、サーバ１１６（図１）からフィルタ情報をダウンロードする。このフィルタ情報は、サーバ１１６が、以前の同期オペレーションで同期クライアント４０１に対して使用したフィルタ設定を含む。さらに、このフィルタ情報は、システム管理者により、または同期クライアント４０１のベンダにより、または同期エンジン１１８（図１）により提供されたフィルタに対する更新を含むことがある。このフィルタ情報は、（ａ）スクリーン識別子、（ｂ）各スクリーンに関連するビジネス・オブジェクトの識別子、（ｃ）各ビジネス・オブジェクトの関連するフィルタなどを含む。スクリーンはそれぞれ、１つだけの関連するビジネス・オブジェクトを有するため、フィルタ情報を、スクリーンとそれに関連するフィルタだけを含むファイルとしてダウンロードすることができる。この実施形態では、ダウンロードされたファイルは、同期クライアント４０１がアクセスするために、ハンドヘルド・デバイス１２０－３（図１）の中に一時的に格納される。

【００８５】

ブロック１１０３で、同期クライアント４０１は、ローカルに格納されているフィルタ設定のファイル（本明細書で、フィルタ設定ファイルとも呼ぶ）を取得する。このフィルタ設定ファイルは、前の同期オペレーション後にユーザによって変更されたフィルタ設定を含んでもよい。次に、同期クライアント４０１は、ブロック１１０３中にそのフィルタ設定にアクセスすることができる。

【００８６】

ブロック１１０５で、同期クライアント４０１は、フィルタ設定ファイルと、ブロック１１０１でダウンロードされたフィルタ情報の両方からの情報を含むようにフィルタ情報を更新する。さらに、同期クライアント４０１は、ビジネス・オブジェクトに関するデフォルトのフィルタ設定を使用してフィルタ情報を更新することもできる。一実施形態では、同期クライアント４０１は、ユーザにフィルタ設定を選択させるコンポーネント（例えば、ダイアログ）を含む。例えば、フィルタ設定選択をユーザが行うことができるように

するメニューとともにそのコンポーネントがスクリーンを表示することができる。その後、ユーザが、所望に応じて設定を変更することができるように次のスクリーンが表示される。スクリーンのすべてが表示されるまで、これが繰り返される。フィルタ設定ファイルは、フィルタ設定選択を反映するように更新される。

【００８７】

ブロック１１０７で、同期クライアント４０１が、更新されたフィルタ情報を含むようにフィルタ設定ファイルを更新する。この更新されたフィルタ設定ファイルは、ハンドヘルド・デバイス１２０－３の中にローカルで保存される。

【００８８】

ブロック１１０９で、同期クライアント４０１が、更新されたフィルタ情報をサーバ１１６に提供する。一実施形態では、同期クライアント４０１は、更新されたフィルタ情報をＸＭＬストリングの形態でサーバ１１６にアップロードする。同期クライアント４０１がフィルタ設定情報をサーバ１１６に送る前に、そのＸＭＬストリングをＵＲＬ符号化する。他の実施形態では、フィルタ情報は、ビジネス・オブジェクトをフィルタ設定とリンクさせるため、および、次のビジネス・オブジェクトを示すために使用される、あらかじめ選択された文字とともにパックされたストリングの形態である。

【００８９】

図１２は、本発明の一実施形態によるブロック１１０３、１１０５、１１０７（図１１）をより詳細に示している。この実施形態では、同期クライアントがオペレーションを実行する。一実施形態では、同期クライアントは、サーバ１１６（図１）に対する直接接続を有するハンドヘルド・デバイス１２０－３（図４）の中に存在する。図１１の説明の場合と同様に、ブロック１１０３、１１０５、１１０７のオペレーションを、ハンドヘルド・デバイス１２０－３（図４）とサーバ１１６（図１）に関連させて説明する。ただし、本開示に鑑みて、以下の説明は、必要以上に実験を行うことなしに、他のタイプの同期クライアントにも当てはまることは当業者には理解できるであろう。図４、図１１を参照すると、ブロック１１０３、１１０５、１１０７は、この典型的な実施形態において以下のとおり実行される。

【００９０】

ブロック１２０１で、同期クライアント４０１が、ダウンロードされたフィルタ情報を解析する（図１１のブロック１１０１を参照）。一実施形態では、ダウンロードされたフィルタ情報はＸＭＬ符号化されたストリームの形態である。同期クライアント４０１は、ダウンロードされた情報から解析されたフィルタ情報を格納するためのデータ構造をセットアップする。一実施形態では、データ構造はリンクされたリストである。別の実施形態では、データ構造は、例えば、ＸＭＬ、ＳＧＭＬ、またはＨＴＭＬなどのマークアップ言語ファイルである。

【００９１】

ブロック１２０３で、同期クライアント４０１は、解析済みのダウンロードされたフィルタ情報の中で次のスクリーンを探索する。同期クライアント４０１は、スクリーンを見つけた場合、ブロック１２０５で、そのスクリーンをデータ構造の中に入れる。例えば、リンクされたリストの実施形態では、同期クライアント４０１は、見つかったスクリーンに関するリンクを作成することができる。ＸＭＬの実施形態では、同期クライアント４０１がスクリーンに関するタグまたは属性を作成する。

【００９２】

ブロック１２０７で、同期クライアント４０１は、ブロック１２０３のスクリーンに関連する次のビジネス・オブジェクトを探索する。同期クライアント４０１は、ビジネス・オブジェクトを見つけた場合、ブロック１２０９で、そのビジネス・オブジェクトをデータ構造の中に、そのビジネス・オブジェクトとスクリーンの間の関係を示す機構とともに入れる。前述したリンクされたリストの例を続けると、同期クライアント４０１は、ブロック１２０３で見つかったスクリーンからブロック１２０７で見つかったビジネス・オブジェクトへのリンクを作成することができる。ＸＭＬの実施形態では、同期クライアント

４０１は、ビジネス・オブジェクトのタグまたは属性を作成することができる。

【００９３】

ブロック１２１１で、同期クライアント４０１は、ブロック１２０７のビジネス・オブジェクトに関連する次のフィルタを探索する。同期クライアント４０１は、フィルタを見つけた場合、ブロック１２１３で、そのフィルタをデータ構造の中に入れる。前述したリンクされたリストの例を続けると、同期クライアント４０１は、ブロック１２０７で見つかったビジネス・オブジェクトからブロック１２１１で見つかったフィルタへのリンクを作成する。ＸＭＬの実施形態では、同期クライアント４０１は、フィルタに関するタグまたは属性を作成する。図１０Ａに関連して前述したとおり、ビジネス・オブジェクトは、複数のフィルタを有している。ただし、この実施形態では、一度に１つフィルタだけがアクティブとされる。

【００９４】

ブロック１２１５で、同期クライアント４０１は、ブロック１２０７で見つかったビジネス・オブジェクトに関するアクティブなフィルタが選択されているかどうかを判定する。一実施形態では、データ構造は、ブロック１２０７で見つかったビジネス・オブジェクトに対応するアクティブな・フィルタをポイントする別個の「アクティブなフィルタ」ポインタを含む。同期クライアント４０１は、このポインタが埋められている（すなわち、アドレスをポイントしている）かどうかを検査する。ポインタが空である場合、ビジネス・オブジェクトに関するアクティブなフィルタは選択されていない。

【００９５】

アクティブなフィルタが選択されている場合、オペレーションの流れは、ブロック１２１１に戻り、次のフィルタを探索する。しかし、アクティブなフィルタが選択されていない場合、同期クライアント４０１は、ブロック１２１７で、ブロック１２０７で見つかったフィルタがデフォルトのフィルタであるかどうかを判定する。デフォルトのフィルタである場合、ブロック１２１９で、同期クライアント４０１は、そのフィルタをデフォルトのフィルタとして設定する。例えば、同期クライアント４０１は、デフォルトのフィルタのアドレスを有する「アクティブなフィルタ」ポインタを読み込み、オペレーションの流れは、ブロック１２１１に戻る。ブロック１２１７で、フィルタがデフォルトのフィルタではない場合、オペレーションの流れは、ブロック１２１１に戻って、次のフィルタを探す。

【００９６】

ブロック１２０７に戻ると、同期クライアント４０１が、次のビジネス・オブジェクトを見つけられなかった場合、オペレーションの流れはブロック１２２１に進む。例えば、同期クライアント４０１は、スクリーンに関するビジネス・オブジェクトのすべてが見つかっている場合、または同期クライアント４０１が、現在のビジネス・オブジェクト（すなわち、ブロック１２０７の前回の実行で見つかった）に関するデフォルトのフィルタを探し出せない場合、次のビジネス・オブジェクトを見出させないことがある。ブロック１２２１で、同期クライアント４０１は、現在のビジネス・オブジェクトに関してアクティブなフィルタが選択済みであるかどうかを判定する。同期クライアント４０１は、アクティブなフィルタが選択済みであると判定した場合（例えば、現在のビジネス・オブジェクトに関するフィルタのすべてが見つかっており、１つがデフォルトのフィルタである場合）、オペレーションの流れは、ブロック１２０３に戻り、次のスクリーンを探す。しかし、同期クライアント４０１が、アクティブなフィルタがまったく選択されていないと判定した場合（例えば、現在のビジネス・フィルタに関してフィルタがまったく見つかっていない、または見つかったフィルタのいずれもデフォルトのフィルタではない場合）、オペレーションの流れは、ブロック１２２３に進む。

【００９７】

ブロック１２２３で、同期クライアント４０１は、「フィルタなし」フィルタではないフィルタを求めて現在のビジネス・オブジェクトのための見つかったフィルタ（群）を探索する（すなわち、ブロック１２１１～１２１９の１回または複数回の繰り返しによって

）。より詳細には、前述したとおり、ビジネス・オブジェクトは、アクティブなフィルタを有さなくてもよい。その場合、この実施形態は、「フィルタなし」フィルタを使用して、ビジネス・オブジェクトがアクティブなフィルタを有さないことを示す。ブロック１２２３で、同期クライアント４０１が、見つかったフィルタ（群）のなかで「フィルタなし」フィルタであるフィルタを見つけた場合、ブロック１２２５で、同期クライアント４０１は、そのフィルタをアクティブなフィルタとして選択する。

【００９８】

しかし、ブロック１２２３で、そのようなフィルタがまったく見つからなかった場合、オペレーションの流れはブロック１２２７に進む。ブロック１２２７の一実施形態では、同期クライアント４０１は、「フィルタなし」フィルタを求めて見つかったフィルタ（群）を探索する。同期クライアント４０１は、「フィルタなし」フィルタを見つけた場合、ブロック１２２９で、そのフィルタをアクティブなフィルタとして選択する。しかし、ブロック１２２７で、同期クライアント４０１が「フィルタなし」フィルタを見つけられなかった場合は、ブロック１２３１で示すとおり、アクティブなフィルタはまったく選択されない。ブロック１２２９、１２３１のいずれの後にも、オペレーションの流れは、ブロック１２０３に戻り、次のスクリーンを探す。

【００９９】

ブロック１２０３で、同期クライアント４０１がスクリーンを見つけられなかった（例えば、スクリーンのすべてが、ブロック１２０３の以前の繰り返しで見つかっている）場合、オペレーションの流れはブロック１２４０に進む。ブロック１２４０のこの実施形態では、同期クライアント４０１は、フィルタ設定ファイルを取得する（前述したブロック１１０３を参照）。前述したとおり、フィルタ設定ファイルは、前回の同期オペレーション後にユーザが変更したフィルタ設定も含む。この実施形態では、フィルタ設定は、各ビジネス・オブジェクトと関連するアクティブ化されたフィルタ（すなわち、ビジネス・オブジェクト／フィルタのペア）を含む。

【０１００】

ブロック１２４２で、同期クライアント４０１は、フィルタ設定ファイルからの次のビジネス・オブジェクト／フィルタのペアを解析する。同期クライアント４０１は、フィルタ設定ファイルからビジネス・オブジェクト／フィルタのペアのすべてが解析されるまで、ループしてブロック１２４２に戻る（以下に説明するとおり）。

【０１０１】

ブロック１２４４で、同期クライアント４０１は、ビジネス・オブジェクト／フィルタのペアが空である（すなわち、前回の同期オペレーションにおいてアクティブなフィルタがまったく選択されていない）かどうかを判定する。ビジネス・オブジェクト／フィルタのペアが空である場合、オペレーションの流れはブロック１２４２に戻り、次のビジネス・オブジェクト／フィルタのペアを解析して取り出す。

【０１０２】

しかし、ブロック１２４４で、同期クライアント４０１が、ビジネス・オブジェクト／フィルタのペアが空でないと判定した場合、オペレーションの流れはブロック１２４６に進む。ブロック１２４６で、同期クライアント４０１は、ビジネス・オブジェクト／フィルタのペアによって示されるビジネス・オブジェクトを求めてデータ構造を探索する（ブロック１２０５～１２１３を参照）。ブロック１２４６は、ビジネス・オブジェクトが、例えば、ハンドヘルド・アプリケーションの更新の中でもはや存在しない可能性があるため、有用である。ブロック１２４６で、同期クライアント４０１が、データ構造の中でビジネス・オブジェクトを見つけられなかった場合、オペレーションの流れはブロック１２４２に戻り、次のビジネス・オブジェクト／フィルタのペアを解析して取り出す。

【０１０３】

しかし、同期クライアント４０１が、ブロック１２４６でビジネス・オブジェクトを見つけた場合、オペレーションの流れはブロック１２４８に進む。ブロック１２４８で、同期クライアント４０１は、そのフィルタを求めて前述したデータ構造を探索する。ビジネ

ス・オブジェクトに関して前述したとおり、フィルタは、ハンドヘルド・アプリケーションの更新に起因してもはや存在しない可能性がある。ブロック１２４８で、同期クライアント４０１が、データ構造の中でフィルタを見つけられなかった場合、オペレーションの流れはブロック１２４２に戻り、次のビジネス・オブジェクト／フィルタのペアを解析して取り出す。しかし、同期クライアント４０１は、データ構造の中でフィルタを見つけた場合、その見つかったフィルタをブロック１２５０でアクティブなフィルタとして設定する。次に、オペレーションの流れはブロック１２４２に戻り、次のビジネス・オブジェクト／フィルタのペアを解析して取り出す。

【０１０４】

図１２の実施形態のオペレーションの流れは、実際、同期クライアントに関してサーバの中に格納されている最新のフィルタ設定を取り出した後、そのフィルタ設定をデフォルトのアクティブなフィルタに再設定する。次に、前回の同期オペレーションからのローカルに格納されているユーザ設定を使用して、ビジネス・オブジェクトとフィルタに関するデフォルトのアクティブなフィルタを更新する。この実施形態は、ハンドヘルド・アプリケーションが更新されたときに、確実にフィルタ設定が正しくなるのに役立つ。

【０１０５】

図１３は、本発明の一実施形態によるデータを抽出するためのブロック１００７（図１０）を示している。この実施形態では、同期クライアントが、ブロック１００７のオペレーションを実行する。図１０の説明の場合と同様に、ブロック１００７のオペレーションを、ハンドヘルド・デバイス１２０－３（図４）とサーバ１１６（図１）に関連させて説明する。ただし、本開示に鑑みて、以下の説明は、必要以上に実験を行うことなしに、他のタイプの同期クライアントにも当てはまることは当業者には理解できるであろう。図４と図１３を参照すると、ブロック１００７は、この典型的な実施形態において以下のとおり実行される。

【０１０６】

ブロック１３０１で、同期クライアント４０１が、比較のために、サーバ１１６からダウンロードされた抽出ＩＤとローカルに格納されていた抽出ＩＤを取得する。前述したとおり、抽出ＩＤにより、データベース抽出のバージョンが識別される。例えば、サーバ１１６上のデータベース構造が変更された場合、双方の抽出ＩＤは相違する。

【０１０７】

ブロック１３０３で、同期クライアント４０１は、ローカルに格納されていた抽出ＩＤとダウンロードされた抽出ＩＤが一致するかどうかを判定する。抽出ＩＤが一致する場合、オペレーションの流れはブロック１３０５に進む。ブロック１３０５で、同期クライアント４０１は、サーバ１１６からのデルタ抽出を要求する。前述したとおり、デルタ抽出は、前回の同期オペレーションのときから変更されているデータベースのデータだけを含む。例えば、一実施形態では、各同期クライアントのために、同期エンジン（例えば、図１の同期エンジン１１８または１２８）が、各同期オペレーションの後、抽出のコピーを保持する。次に、現在の同期オペレーション中、同期エンジン（例えば、同期エンジン１１８または１２８）は、メイン・データベース１１２（図１）から同期クライアントに関する完全抽出を取り出す。デルタ抽出の場合、同期クライアント４０１は、格納されている抽出をメイン・データベース１１２から取り出された完全抽出と比較し、変更されているデータベースのデータだけを同期エンジン（１１８または１２８）に伝送する。このデルタ抽出が、次に、同期クライアント４０１によって受け取られる。

【０１０８】

反対に、同期クライアント４０１が、ブロック１３０３で、双方の抽出ＩＤが一致しないと判定した場合、オペレーションの流れはブロック１３０７に進む。ブロック１３０７で、同期クライアント４０１は、サーバ１１６からの完全抽出を要求する。すると、サーバ１１６は、完全抽出を提供し、この完全抽出が、同期クライアント４０１によって受け取られる。

【０１０９】

図１４は、本発明の一実施形態による圧縮オペレーションを示している。この圧縮オペレーションは、同期クライアント（例えば、図４の同期クライアント４０１）またはサーバ（例えば、図１のサーバ１１４または１１６）によって実行することが可能である。例えば、この圧縮オペレーションは、データベースのデータ（例えば、デルタ抽出）を同期クライアント４０１に送る前に、サーバ１１６によって実行することが可能である。この圧縮アルゴリズムは、同期クライアント４０１が情報をサーバ１１６に送り返すのに使用することもできる。例えば、この圧縮オペレーションは、ブロック７０３（図７）を実施するのに使用することができる。この圧縮オペレーションにより、サーバと同期クライアントの間で情報を転送するのに必要とされる時間が短縮される。

【０１１０】

ブロック１４０１で、伝送される情報が圧縮される。一実施形態では、情報は、データベースのデータ、トランザクション・データまたはメタデータなどのバイナリ・データである。任意の適切な圧縮アルゴリズムを使用することができる。一実施形態では、バイナリ情報は、ｗｗｗ．ｉｎｆｏ－ｚｉｐ．ｏｒｇから入手可能なＺＩＰ２．３圧縮ユーティリティーを使用して圧縮される。他の実施形態では、他の圧縮アルゴリズムを使用することができる。

【０１１１】

ブロック１４０３で、圧縮済みのバイナリ・データがテキストに変換される。一実施形態では、圧縮済みのバイナリ・データは、標準のＢａｓｅ６４符号化を使用してテキストに変換される。テキストへの変換は、伝送プロセス中のデータの破壊を減少させるのに役立つ。さらなる改良形態では、テキスト・データは、ＸＭＬ（拡張マークアップ言語）またはＨＴＭＬ（ハイパーテキストマークアップ言語）などのマークアップ言語を使用するマークアップを含むことが可能である。

【０１１２】

ブロック１４０５で、ブロック１４０３で生成されたテキスト・データが、サーバと同期クライアントの間の接続用のプロトコルを使用して符号化される。一実施形態では、標準のハイパーテキスト転送プロトコル（ＨＴＴＰ）を使用して、インターネットを介する伝送のためにテキスト・データを符号化する。他の実施形態では、サーバと同期クライアントの間における接続の性質に応じて、異なるプロトコルを使用することができる。例えば、他の実施形態では、テキスト・データは、ファイル転送プロトコル（ＦＴＰ）符号化することが可能である。次に、この圧縮され、符号化された情報を宛先の受取側に送る。一部の実施形態では、ブロック１４０３は、ブロック１４０５がテキストの代わりにブロック１４０１からの圧縮済みのバイナリ・データを符号化して、省略することができる。

【０１１３】

このオペレーションのさらなる改良形態では、送られる情報は、より小さいユニットに分割することができ、このより小さいユニットが、次に、ブロック１４０１、１４０３、１４０５に従って別々に圧縮され、符号化される。このより小さいユニットが、受取側に別々に送られて、情報の次のユニットを受け取る前に、受取側が、（ａ）圧縮された情報ユニットを格納し、（ｂ）格納済みの圧縮された情報ユニットを圧縮解除し、（ｃ）圧縮解除された情報ユニットを格納し、（ｄ）圧縮された情報ユニットを削除する。この改良形態により、受取側が伝送された情報を適切に受け取り、圧縮解除するのに必要とする利用可能なメモリの量が減少する。

【０１１４】

サーバ・プロセス

図１５は、本発明の一実施形態によるサーバの同期プロセスを示している。この実施形態では、サーバは、同期クライアントに接続される。例えば、同期クライアントは、サーバに対する直接接続を有するハンドヘルド・デバイス内に存在する。一実施形態では、ハンドヘルド・デバイスとサーバは、ハンドヘルド・デバイス１２０－３（図４）とサーバ１１６（図１）とすることが可能である。別法として、同期クライアントは、コンパニオン・デバイスとハンドヘルド・デバイス（例えば、図５におけるようなコンパニオン・デ

バイス１２４やハンドヘルド・デバイス１２０－２）の組み合わせによって実装することもできる。便宜上、サーバのオペレーションを、サーバ１１６（図１）とハンドヘルド・デバイス１２０－３（図４）に関連させて説明する。本開示に鑑みて、以下の説明は、必要以上に実験を行うことなしに、他のタイプの同期クライアントにも当てはまることは当業者には理解できるであろう。

【０１１５】

図４、図１５を参照すると、サーバの一実施形態が、以下のとおり同期クライアントに連携して動作する。ブロック１５０１で、サーバが同期クライアントに接続される。同期クライアントは、何らかの実施形態におけるコンパニオン・デバイス内に存在することが可能である。この実施形態では、サーバ１１６は、ユーザがログインしたことに応答して同期クライアント４０１に接続される。一実施形態では、サーバ１１６は、モデムを動作させて同期クライアント４０１に直接にアクセスするインターフェースまたはドライバを実行する。他の実施形態では、この接続はインターネット接続である。さらに他の実施形態では、接続は、ＲＦ技術または光技術を使用して直接の通信リンク、Ｗｅｂベースの通信リンク、または他のタイプの通信リンクを実施する無線接続である。

【０１１６】

ブロック１５０３で、サーバは、初期設定データを同期クライアントにダウンロードする。一実施形態では、サーバ１１６が、接続１３２を介して初期設定データを同期クライアント４０１にダウンロードする。前述したとおり、この初期設定データは、メイン・データベース１１２の最新バージョン（例えば、抽出ＩＤ）、同期クライアント４０１によってサーバ１１６にアップロードされた最新のトランザクションの識別子、さらにハンドヘルド・アプリケーションの最新バージョン（例えば、リポジトリＩＤ）に関連する情報を含む。

【０１１７】

ブロック１５０５で、サーバは、同期クライアントからトランザクション情報を受け取る。一実施形態では、サーバ１１６が、同期クライアント４０１からトランザクション情報を受け取り、そのトランザクション情報をコンピュータ・システム（図１）のメモリ（図示せず）に格納する。例えば、サーバ１１６は、同期クライアント４０１からの記録済みのトランザクションのすべてに関するトランザクション情報を１つの比較的大きいブロックのデータで受け取る。別法として、サーバ１１６は、同期クライアント４０１からいくつかのより小さいブロックでトランザクション情報を受け取っても良い。この代替の実施形態では、サーバ１１６は、それぞれのより小さいブロックを受け取った後、次のより小さいブロックを送るように同期クライアント４０１に知らせる受取確認をその同期クライアント４０１に提供する。

【０１１８】

ブロック１５０７で、サーバは同期クライアントへメタデータを送る。一実施形態では、サーバ１１６が同期クライアント４０１にメタデータを送る。前述したとおり、このメタデータは、ハンドヘルド・アプリケーションに関するアプリケーション定義を含む。この実施形態では、サーバ１１６は、同期クライアント４０１からの要求に応答してブロック１５０７を実行する。前述したとおり、同期クライアント４０１は、サーバ１１６の中に格納された（最新の同期オペレーションから）アプリケーションが、ローカルに格納されているアプリケーション定義バージョンと一致しない場合、メタデータを要求する。この状況は、アプリケーション定義が、前回の同期オペレーション後に更新されている場合に生じることがある。別の実施形態では、同期クライアント４０１が、バージョンを比較することなしに、同期オペレーション中、メタデータを常に要求するように構成される。この代替の実施形態では、サーバ１１６は、前回の同期オペレーション以降、アプリケーション定義が更新されているかどうかを判定し、更新されている場合、メタデータを同期クライアント４０１に送る。この代替の実施形態のさらなる改良形態では、サーバ１１６は、同期クライアント４０１からの要求を待つことなしに、以上のオペレーションを自動的に実行し、アプリケーション定義が更新されている場合、メタデータを同期クライアン

ト４０１にプッシュすることができる。
【０１１９】
ブロック１５０９で、サーバは、同期クライアントのローカル・データベースを更新する。一実施形態では、サーバ１１６が、同期クライアント４０１から要求を受け取って、データ抽出オペレーションを開始する。この実施形態では、サーバ１１６は、データ抽出オペレーションを実行する前に更新されたフィルタ設定（上記の図１１、図１２の説明を参照）を受け取ることができる。このため、サーバ１１６は、データ、不要なデータベースのデータをダウンロードすることを回避し、これは、電池の電力を節約し、同期プロセスを完了させるのに必要とされる時間を短縮するのに役立つ。一実施形態では、サーバ１１６は、データ抽出全体（すなわち、同期クライアントに可視であり、フィルタリングした後のデータ）を単一の大きいブロックでダウンロードする。代替の実施形態では、サーバ１１６は、同期クライアント４０１による要求に応答して比較的小さいブロックのデータ抽出をダウンロードしてもよい。同期クライアント４０１が、この比較的小さいブロックを適切に受け取った場合、同期クライアント４０１は、次の比較的小さいブロックを求める要求を送り、同期クライアント４０１がサーバ１１６からデータ抽出全体を受け取るまで、以下同様に行われる。このデータ抽出は、完全抽出またはデルタ抽出とすることが可能である（図１６に関連して以下により詳細に説明する）。
【０１２０】
ブロック１５１１で、サーバが同期クライアントから切り離される。この実施形態では、サーバ１１６が、ユーザが開始した（または同期オペレーションの完了時に同期クライアント４０１によって自動的に実行された）ログアウト・プロセスに応答して、同期クライアント４０１から切り離される。代替の実施形態では、ブロック１５０９のオペレーションを実行した後、サーバ１１６が同期クライアント４０１から自動的に切り離される。
【０１２１】
図１６は、本発明の一実施形態によるブロック１５０９（図１５）を示している。この実施形態では、サーバがブロック１５０９のオペレーションを実行する。図１５の説明の場合と同様に、ブロック１５０９のオペレーションを、ハンドヘルド・デバイス１２０－３（図４）とサーバ１１６（図１）に関連させて説明する。ただし、本開示に鑑みて、以下の説明は、必要以上に実験を行うことなしに、他のタイプの同期クライアントにも当てはまることは当業者には理解できるであろう。図４、図１６を参照すると、ブロック１５０９は、この典型的な実施形態において以下のとおり実行される。
【０１２２】
ブロック１６０１で、サーバ１１６が、同期クライアント４０１に可視であるデータをメイン・データベース１１２から抽出する。さらに、サーバ１１６は、同期クライアント４０１によってサーバ１１６に提供されたフィルタ設定に従ってそのデータをフィルタリングする（図１１、図１２に関連して説明した）。
【０１２３】
ブロック１６０３で、サーバ１１６は、後の同期オペレーションにおいて後に使用するために、抽出済みのデータのファイルをメモリ（図示せず）に保存する。ブロック１６０５で、サーバ１１６は、この抽出を前回の同期オペレーションからの抽出されたデータを含むファイルと比較する。対応するデータベース・レコードからのデータが異なる場合、ブロック１６０７で、サーバ１１６は、より新しい抽出からのデータ（すなわち、「デルタ」）を本明細書でデルタ抽出ファイルと呼ぶ別のファイルの中に保存する。一実施形態では、サーバ１１６は、デルタ抽出を同期クライアント４０１が受け取ると、同期クライアント４０１はそのデルタ抽出から解析することができるレコード／データのペアとして各デルタを格納する。この実施形態では、予約されたデリミタが、各レコード内のフィールドを分離し、異なる予約されたデリミタがレコードを分離する。一実施形態では、エスケープ機構を使用して、これらの予約されたデリミタが、レコード・フィールド値とともに現れるようにすることができる。さらに、一部の実施形態では、デルタ抽出ファイルは、レコードが削除されているか、新規であるか、または変更されているかを示す各レコー

ド／データのペアに関するインジケータを含むこともある。

【０１２４】

さらなる改良形態では、デルタ抽出は、デルタ抽出ファイルのサイズをさらに小さくするために、レコード・レベル（いくつかのフィールドを含む可能性がある）ではなく、フィールド・レベルで実行される。

【０１２５】

次に、ブロック１６０９で、サーバ１１６は、同期クライアント４０１に完全抽出をダウンロードするか、デルタ抽出をダウンロードするかを判定する。一実施形態では、サーバ１１６は、完全抽出か、またはデルタ抽出を求める要求を同期クライアント４０１から受け取る。前述したとおり、一実施形態では、同期クライアント４０１は、サーバ１１６からダウンロードされた抽出ＩＤが同期クライアント４０１によってローカルに格納されていた抽出ＩＤに一致するかどうかに基づいてこの要求を行う。１つの典型的な実施形態では、ローカル・データベース３０８の構造が、前回の同期オペレーション以降に変更されている（例えば、メイン・データベース１１２または可視性規則の変更に起因して）場合、双方の抽出ＩＤが相違し、同期クライアントは完全抽出を要求する。ローカル・データベース３０８の構造が変更されていない場合、双方の抽出ＩＤは同一であり、同期クライアント４０１はデルタ抽出を要求する。ブロック１６０９の結果に応じて、サーバ１１６は、ブロック１６１１または１６１３で、同期クライアント４０１に、それぞれ、完全抽出を送るか、またはデルタ抽出を送る。

【０１２６】

別の実施形態では、サーバ１１６は、同期クライアントからローカルに記憶されていた抽出ＩＤを受け取り、その受け取った抽出ＩＤをサーバ１１６の中に格納されている最新の抽出ＩＤと比較する。例えば、サーバ１１６は、そのデータを同期クライアント４０１へ要求することができ、あるいは、これは、データ抽出プロセス（例えば、抽出ＩＤの代わりに、同期クライアントが、自らのローカル抽出ＩＤを提供すること以外は、図１０のブロック１１０１と同様である）の一環であってもよい。次に、サーバ１１６は、適宜、デルタ抽出をダウンロードするか、または完全抽出をダウンロードするかを判定することができる。例えば、サーバ１１６が、同期クライアント４０１の代わりに、ブロック１３０１、１３０３（図１３）を実行することもできる。

【０１２７】

同様な形で、サーバ１１６は、メタデータを同期クライアント４０１に提供するかどうかを判定する際（図６のブロック６０９を参照）、リポジトリＩＤを受信する。例えば、サーバ１１６は、そのデータを同期クライアント４０１から要求することができ、あるいは、これは、データ抽出プロセスの一環（例えば、リポジトリＩＤを取得する代わりに、同期クライアントが自らのローカルで格納されているリポジトリＩＤを提供すること以外は、図６のブロック６０５と同様である）であってもよい。次に、サーバ１１６は、メタデータをダウンロードするかどうかを判定することができる。例えば、サーバ１１６は、同期クライアント４０１の代わりにブロック９０１および９０３（図９）を実行するように構成することができる。

【０１２８】

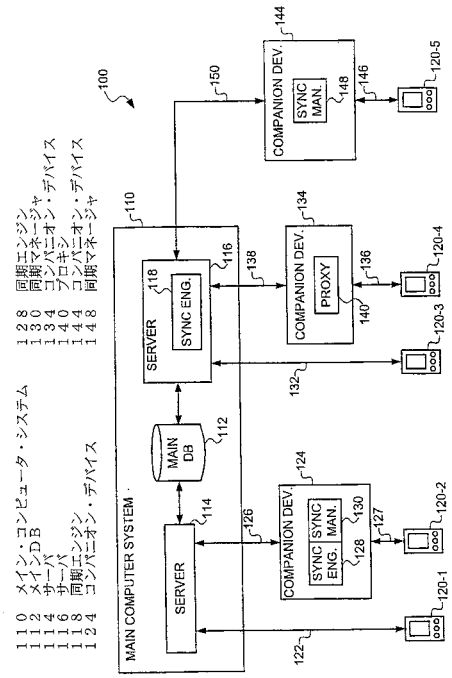
要約書で説明していることを含め、本発明の例示した実施形態の以上の説明は、本発明を網羅する、または開示した形態そのものに限定するものではない。本発明の特定の実施形態、および例を本明細書で、例示の目的で説明しているが、当業者なら理解できるように、本発明の範囲内で様々な均等な変更が可能である。

【０１２９】

以上の詳細な説明に鑑みて、それらの変更を本発明に対して行うことができる。以下の特許請求の範囲で使用する用語は、本明細書および特許請求の範囲で開示する特定の実施形態に本発明を限定するものと解釈してはならない。むしろ、本発明の範囲は、請求項解釈の既成の原則に従って解釈されるべき特許請求の範囲によって完全に確定されるべきものである。

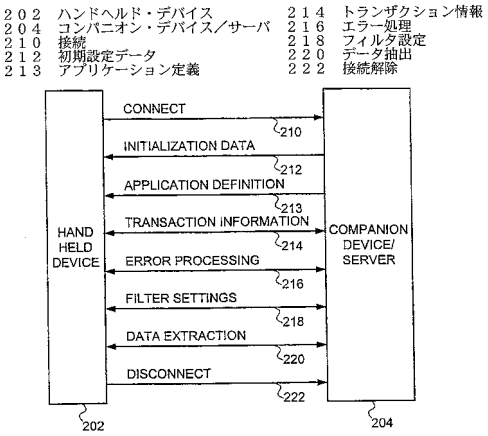
【図面の簡単な説明】
【０１３０】
【図１】本発明の一実施形態によるハンドヘルド・デバイスを介してユーザがアクセスすることができるメイン・データベースを有するシステムを示すブロック図である。
【図２】本発明の一実施形態によるハンドヘルド・デバイスと別のコンピュータ・システムの間のデータフローを示す図である。
【図３】本発明の一実施形態によるハンドヘルド・デバイスを同期する際に使用されるサーバおよびクライアントのコンポーネントを示す最上レベルのブロック図である。
【図４】本発明の一実施形態によるハンドヘルド・デバイスをサーバと直接に同期させる際に使用されるハンドヘルド・デバイスのコンポーネントを示すより詳細なブロック図である。
【図５】本発明の一実施形態によるコンパニオン・デバイスを介してハンドヘルド・デバイスをサーバと同期させる際に使用されるコンパニオン・デバイスおよびハンドヘルド・デバイスのコンポーネントを示す図である。
【図６】本発明の一実施形態によるクライアントの同期プロセスを示す流れ図である。
【図７】本発明の一実施形態によるトランザクション処理オペレーションを示す流れ図である。
【図８】本発明の一実施形態によるトランザクション送りオペレーションを示す流れ図である。
【図９】本発明の一実施形態によるメタデータ更新オペレーションを示す流れ図である。
【図１０】本発明の一実施形態によるデータ抽出オペレーションを示す流れ図である。
【図１０Ａ】本発明の一実施形態によるフィルタを示す図である。
【図１１】本発明の一実施形態によるフィルタ処理オペレーションを示す図である。
【図１２】本発明の一実施形態によるフィルタ処理オペレーションをより詳細に示す流れ図である。
【図１３】本発明の一実施形態によるデータ抽出オペレーションを示す流れ図である。
【図１４】本発明の一実施形態による圧縮オペレーションを示す流れ図である。
【図１５】本発明の一実施形態によるサーバの同期プロセスを示す流れ図である。
【図１６】本発明の一実施形態によるデータベースのデータ送りオペレーションを示す流れ図である。

【図１】



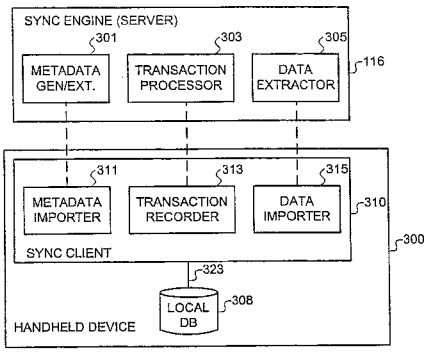
１１０　メイン・コンピュータ・システム
１１２　メインＤＢ
１１４　サーバ
１１６　同期エンジン
１１８　同期マネージャ
１２４　コンパニオン・デバイス
１２８　同期エンジン
１３０　同期マネージャ
１３４　コンパニオン・デバイス
１４０　プロキシ
１４４　コンパニオン・デバイス
１４８　同期マネージャ

【図２】



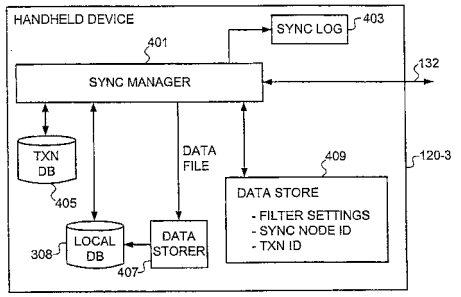
２０２　ハンドヘルド・デバイス
２０４　コンパニオン・デバイス／サーバ
２１０　接続
２１２　初期設定データ
２１３　アプリケーション定義
２１４　トランザクション情報
２１６　エラー処理
２１８　フィルタ設定
２２０　データ抽出
２２２　接続解除

【図３】



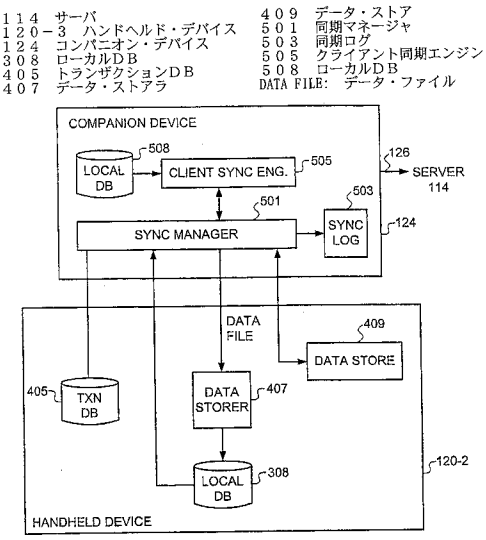
１１６　同期エンジン（サーバ）
３００　ハンドヘルド・デバイス
３０１　メタデータ・ジェネレータ／エクストラクタ
３０３　トランザクション・プロセッサ
３０５　データ・エクストラクタ
３０８　ローカルＤＢ
３１０　同期クライアント
３１１　メタデータ・インポータ
３１３　トランザクション・レコーダ
３１５　データ・インポータ

【図４】

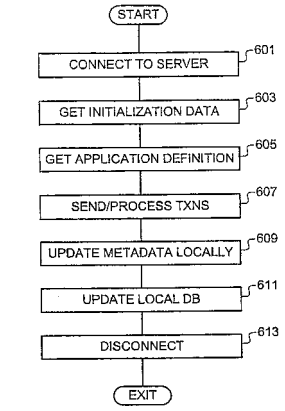


HANDOHELD DEVICE:　ハンドヘルド・デバイス
SUNC LOG:　同期ログ
SUNC MANAGER:　同期マネージャ
LOCAL DB:　ローカルＤＢ
DATA STORER:　データ・ストアラ
DATA FILE:　データ・ファイル
DATA STORE:　データ・ストア
FILTER SETTINGS:　フィルタ設定
SYNC NODE ID:　同期ノードＩＤ
TXN ID:　トランザクションＩＤ

【図５】

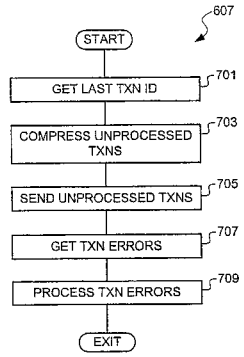


【図６】



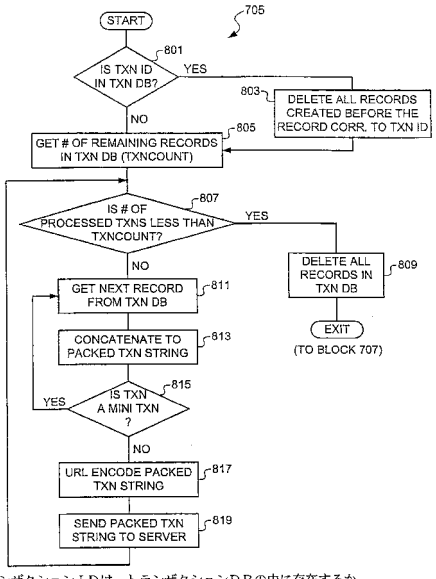
６０１　サーバに接続する
６０３　初期設定データを取得する
６０５　アプリケーション定義を取得する
６０７　トランザクションを送り／処理する
６０９　メタデータをローカルで更新する
６１１　ローカルＤＢを更新する
６１３　切り離される

【図７】



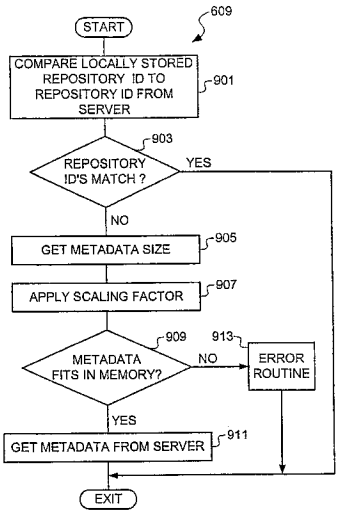
７０１　最新のトランザクションＩＤを取得する
７０３　未処理のトランザクションを圧縮する
７０５　未処理のトランザクションを送る
７０７　トランザクション・エラーを取得する
７０９　トランザクション・エラーを処理する

【図８】



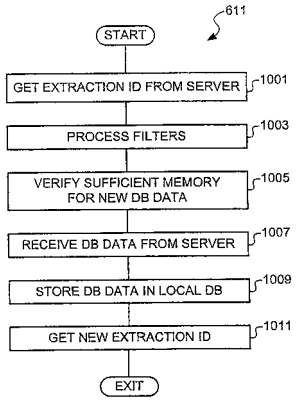
８０１　トランザクションＩＤは、トランザクションＤＢの中に存在するか
８０３　トランザクションＩＤに対応するレコードより前に作成されたすべてのレコードを削除する
８０５　トランザクションＤＢの中の残っているレコードの数（ＴＮＸＣＯＵＮＴ）を取得する
８０７　処理済みのトランザクションの数は、ＴＸＮＣＯＵＮＴ未満か
８０９　トランザクションＤＢの中のすべてのレコードを削除する
８１１　トランザクションＤＢから次のレコードを取得する
８１３　パックされたトランザクション・ストリングに連結する
８１５　トランザクションは、ミニトランザクションか
８１７　パックされたトランザクション・ストリングをＵＲＬ符号化する
８１９　パックされたトランザクション・ストリングをサーバに送る
TO BLOCK 707:　ブロック７０７へ

【図９】



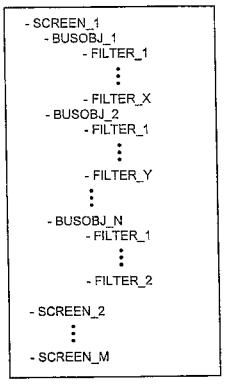
９０１　ローカルに格納されているリポジトリＩＤをサーバからのリポジトリＩＤと比較する
９０３　リポジトリＩＤは、一致するか
９０５　メタデータ・サイズを取得する
９０７　スケーリング・ファクタを適用する
９０９　メタデータは、メモリの中に入るか
９１１　サーバからメタデータを取得する
９１３　エラー・ルーチン

【図１０】

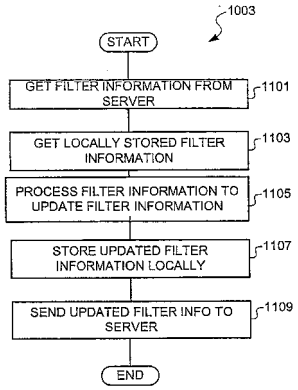


１００１　サーバから抽出ＩＤを取得する
１００３　フィルタを処理する
１００５　新規のＤＢデータ用の十分なメモリを検証する
１００７　サーバからＤＢデータを受け取る
１００９　ＤＢデータをローカルＤＢの中に格納する
１０１１　新規の抽出ＩＤを取得する

【図１０Ａ】

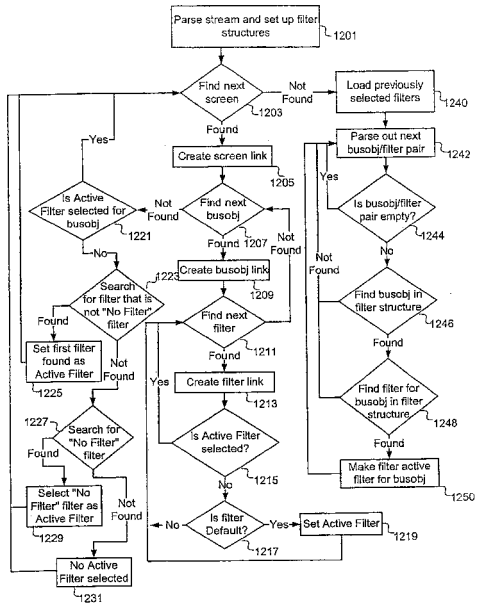


【図１１】



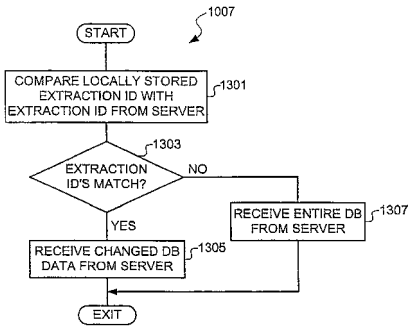
１１０１　サーバからフィルタ情報を取得する
１１０３　ローカルに格納されているフィルタ情報を取得する
１１０５　フィルタ情報を処理して、フィルタ情報を更新する
１１０７　更新されたフィルタ情報をローカルで格納する
１１０９　更新されたフィルタ情報をサーバに送る

【図１２】



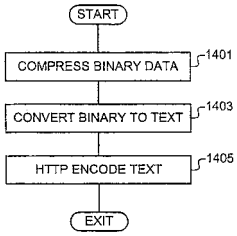
Found: 見つかった、　　Not Found: 見つからなかった
１２０１　ストリームを解析して、フィルタ構造をセットアップする
１２０３　次のスクリーンを探す
１２０５　スクリーン・リンクを作成する
１２０７　次のビジネス・オブジェクトを探し出す
１２０９　ビジネス・オブジェクト・リンクを作成する
１２１１　次のフィルタを探し出す
１２１３　フィルタ・リンクを作成する
１２１５　アクティブなフィルタは、選択されているか
１２１７　フィルタはデフォルトか
１２１９　アクティブなフィルタを設定する
１２２１　ビジネス・オブジェクトに関してアクティブなフィルタは選択されているか
１２２３　「フィルタなし」フィルタでないフィルタを探索する
１２２５　見つかった最初のフィルタをアクティブなフィルタとして設定する
１２２７　「フィルタなし」フィルタを探索する
１２２９　「フィルタなし」フィルタをアクティブなフィルタとして選択する
１２３１　アクティブなフィルタがまったく選択されていない
１２４０　以前に選択されているフィルタをロードする
１２４２　次のビジネス・オブジェクト／フィルタのペアを解析して取り出す
１２４４　ビジネス・オブジェクト／フィルタのペアは、空か
１２４６　フィルタ構造の中でビジネス・オブジェクトを探し出す
１２４８　フィルタ構造の中でビジネス・オブジェクトに関するフィルタを探し出す
１２５０　フィルタをビジネス・オブジェクトに関するアクティブなフィルタにする

【図１３】



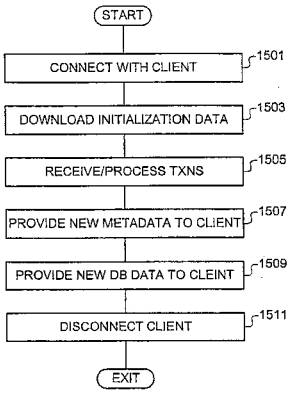
１３０１　ローカルに格納されていた抽出ＩＤをサーバからの抽出ＩＤと比較する
１３０３　抽出ＩＤは一致するか
１３０５　変更されたＤＢデータをサーバから受け取る
１３０７　ＤＢ全体をサーバから受け取る

【図１４】



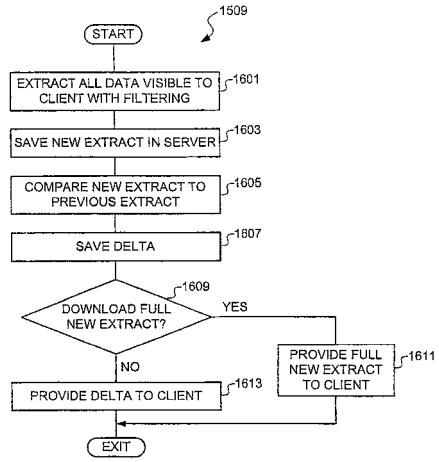
１４０１　バイナリ・データを圧縮する
１４０３　バイナリをテキストに変換する
１４０５　テキストをＨＴＴＰ符号化する

【図１５】



１５０１　クライアントに接続する
１５０３　初期設定データをダウンロードする
１５０５　トランザクションを受け取り／処理する
１５０７　新規のメタデータをクライアントに提供する
１５０９　新規のＤＢデータをクライアントに提供する
１５１１　クライアントを切り離される

【図１６】



１６０１　フィルタリングを行ってクライアントに可視であるすべてのデータを抽出する
１６０３　新規の抽出をサーバの中に保存する
１６０５　新規の抽出を以前の抽出と比較する
１６０７　デルタを保存する
１６０９　完全な新規の抽出をダウンロードするか
１６１１　完全な新規の抽出をクライアントに提供する
１６１３　デルタをクライアントに提供する

フロントページの続き

(72)発明者　ヴェジルストラップ，マグヌス
アメリカ合衆国・９４１２２・カリフォルニア州・サンフランシスコ・４１エスティ　アベニュ・１７３８
(72)発明者　ラバース，ディビッド・エル
アメリカ合衆国・９８０５９・ワシントン州・ニューキャッスル・１４２エヌデイ　ウエイ　サウスイースト・７７２０
(72)発明者　チュン，ピーユ
アメリカ合衆国・９８００６・ワシントン州・ベルビュ・サウスイースト　５８ティエイチ　ストリート・１５８０２
(72)発明者　サッサー，マーティン
アメリカ合衆国・９８４９７・ワシントン州・ユニバーシティ　プレイス・コート　ウエスト・５８ティエイチ　ストリート・７１０４
(72)発明者　ハンセン，アーロン
アメリカ合衆国・９８０２７・ワシントン州・イサクアー・サウスイースト　ニューポート　ウエイ・１８６０１・ナンバーエフ１３１
(72)発明者　スコット，ブライアン
アメリカ合衆国・９８０５８・ワシントン州・レントン・１６３アールディ　プレイス　サウスイースト・１６８１５
(72)発明者　ジョージ，ブライアン
アメリカ合衆国・９８０５８・ワシントン州・レントン・１６３アールディ　プレイス　サウスイースト・１６８１５

審査官　田川　泰宏

(56)参考文献　特表２００１－５１４７７６（ＪＰ，Ａ）
国際公開第９９／０４５４８２（WO，Ａ１）
国際公開第９９／００４６９６（WO，Ａ１）
国際公開第０２／０１２９８５（WO，Ａ１）
国際公開第００／０３３２１７（WO，Ａ１）
国際公開第９７／０３５２６５（WO，Ａ１）
国際公開第９８／０３８５８６（WO，Ａ１）
国際公開第９８／０４０８０６（WO，Ａ１）

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