(19)日本国特許庁(JP)　(12)公表特許公報(A)　(11)特許出願公表番号
特表2009-506632
(P2009-506632A)
(43)公表日　平成21年2月12日(2009.2.12)

(51)Int.Cl.			F I			テーマコード（参考）
H04L	9/32	(2006.01)	H04L	9/00	675B	5B017
G09C	1/00	(2006.01)	G09C	1/00	640D	5J104
G06F	21/24	(2006.01)	G06F	12/14	560C	
			G06F	12/14	520A	

審査請求 未請求　予備審査請求 未請求　（全 27 頁）

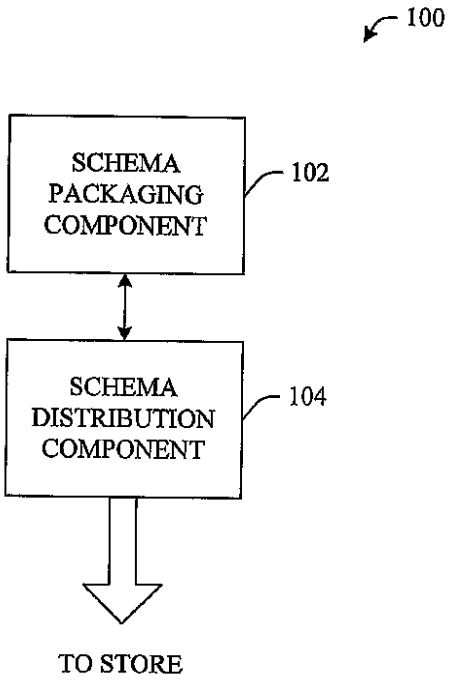
(21)出願番号　特願2008-527920 (P2008-527920)
(86)(22)出願日　平成18年7月20日 (2006.7.20)
(85)翻訳文提出日　平成20年2月25日 (2008.2.25)
(86)国際出願番号　PCT/US2006/028386
(87)国際公開番号　WO2007/024379
(87)国際公開日　平成19年3月1日 (2007.3.1)
(31)優先権主張番号　60/711,246
(32)優先日　平成17年8月25日 (2005.8.25)
(33)優先権主張国　米国 (US)
(31)優先権主張番号　11/287,076
(32)優先日　平成17年11月23日 (2005.11.23)
(33)優先権主張国　米国 (US)

(71)出願人　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　ジェイソン　ティー．ハンター
アメリカ合衆国　９８０５２　ワシントン州　レッドモンド　ワン　マイクロソフト　ウェイ　マイクロソフト　コーポレーション内

最終頁に続く

(54)【発明の名称】スキーマパッケージング、配布および可用性

(57)【要約】
スキーマパッケージング、配布および可用性を可能にするシステムおよび／または方法が開示されている。本発明の態様によれば、スキーマパッケージのセキュアな定義を容易にする署名テクノロジが採用されている。このシステムおよび／または方法によれば、スキーマパッケージをリソースとしてクライアント側アセンブリに組み込むことにより、そのアセンブリを見つけるために使用されたのと全く同じインフラストラクチャが、スキーマパッケージを見つけるときの梃子となり得ることも保証している。他の態様によれば、スキーマパッケージをサテライトアセンブル(satellite assemble)として、または別のファイルとして組み込むことを可能にしている。これらのシナリオにおいて、本発明によれば、クライアントが未インストールのスキーマに関してストアとやりとりするのを可能にするために必要なスキーマパッケージと情報の構築と配布を容易にすることができる。



【特許請求の範囲】
【請求項１】
スキーマパッケージの可用性を容易にするシステムであって、該システムは、
　前記スキーマパッケージ対してストロングネームを署名するパッケージングコンポーネントと、
　前記スキーマパッケージの所在を突き止めて、ストアに投入するために前記スキーマパッケージの可用性を容易にする配布コンポーネントと、
を備えたことを特徴とするシステム。
【請求項２】
前記パッケージングコンポーネントは、前記スキーマパッケージとクライアント側アセンブリとの間に固有の関係を作成することを特徴とする請求項１に記載のシステム。
【請求項３】
前記パッケージングコンポーネントは、署名された前記スキーマパッケージをリソースとしてクライアント側アセンブリに組み込むことを特徴とする請求項２に記載のシステム。
【請求項４】
前記パッケージングコンポーネントは、署名された前記スキーマパッケージをサテライトアセンブリとして配備することを特徴とする請求項２に記載のシステム。
【請求項５】
前記パッケージングコンポーネントは、署名された前記スキーマパッケージを別のファイルとして配置することを特徴とする請求項２に記載のシステム。
【請求項６】
前記配布コンポーネントは、
　少なくともその一部が前記クライアント側アセンブリとの固有の関係に基づいて、前記スキーマパッケージの所在を突き止めることを容易にするディスカバリコンポーネントと、
　前記スキーマパッケージへのアクセスを容易にするローディングコンポーネントと、
を含むことを特徴とする請求項２に記載のシステム。
【請求項７】
前記パッケージングコンポーネントは、共通暗号化鍵を使用して前記スキーマパッケージと前記クライアント側アセンブリに署名することを特徴とする請求項２に記載のシステム。
【請求項８】
署名された前記スキーマパッケージを個人と組織の少なくとも一方と関連付ける認証コードコンポーネントを、さらに備えたことを特徴とする請求項２に記載のシステム。
【請求項９】
署名された前記スキーマパッケージコンポーネントを圧縮して、前記スキーマパッケージの伝送オーバヘッドを軽減する圧縮コンポーネントを、さらに備えたことを特徴とする請求項２に記載のシステム。
【請求項１０】
署名された前記スキーマパッケージの暗号化を容易にする暗号化コンポーネントを、さらに備えたことを特徴とする請求項２に記載のシステム。
【請求項１１】
スキーマを配布するコンピュータ実装方法であって、該コンピュータ実装方法は、
　暗号化鍵を使用して前記スキーマパッケージに署名し、
　暗号化鍵を使用してクライアント側アセンブリに署名し、
　少なくともその一部が暗号化鍵に基づいて前記スキーマパッケージを配布する、
ことを含むことを特徴とするコンピュータ実装方法。
【請求項１２】
署名された前記スキーマパッケージをリソースとして前記クライアント側アセンブリに組み込むことを、さらに含むことを特徴とする請求項１１に記載のコンピュータ実装方法。

【請求項１３】
前記スキーマパッケージを前記クライアント側アセンブリに関係するサテライトアセンブリとして指定することを、さらに含むことを特徴とする請求項１１に記載のコンピュータ実装方法。
【請求項１４】
前記スキーマパッケージから別のファイルを生成し、その別のファイルを前記クライアント側アセンブリに関係付けることを、さらに含むことを特徴とする請求項１１に記載のコンピュータ実装方法。
【請求項１５】
配布するアクトは、さらに、
　少なくともその一部が暗号化鍵に基づいて前記スキーマパッケージを見つけ、
　アプリケーションによる使用のために前記スキーマパッケージをローディングすることを含むことを特徴とする請求項１１に記載のコンピュータ実装方法。
【請求項１６】
前記スキーマパッケージを特定の個人と組織の少なくとも一方に関連付けることを、さらに含むことを特徴とする請求項１１に記載のコンピュータ実装方法。
【請求項１７】
配布するアクトに先立って前記スキーマパッケージを圧縮することを、さらに含むことを特徴とする請求項１１に記載のコンピュータ実装方法。
【請求項１８】
配布するアクトに先立って前記スキーマパッケージを暗号化することを、さらに含むことを特徴とする請求項１１に記載のコンピュータ実装方法。
【請求項１９】
スキーマパッケージの配布を容易にするシステムであって、該システムは、
　暗号化鍵を使用して前記スキーマパッケージに署名する手段と、
　暗号化鍵を使用してクライアント側アセンブリに署名する手段と、
　前記スキーマパッケージを前記クライアント側アセンブリのリソースとして前記クライアント側アセンブリに組み込む手段と、
　アプリケーションによる使用のために前記スキーマパッケージを自動的に見つける手段と、
を備えたことを特徴とするシステム。
【請求項２０】
前記スキーマパッケージ内に組み込まれたスキーマ化タイプがインスタンス生成されると、前記スキーマパッケージに自動的にアクセスする手段を、さらに備えたことを特徴とする請求項１９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、スキーマパッケージング、配布および利用を可能にするシステムおよび／または方法に関する。
【背景技術】
【０００２】
　コンピューティングシステムにおける技術的進歩は、データの共通性を増大することによってマシン間の共有、互換および相互運用を可能にすることを目的としていた。例えば、周知であるように、ユーザはデータをある装置に置いておき、そのデータを別の装置および／またはアプリケーションと共有することを望むことがよくある。今日では、システムが特定のデータフォーマットとタイプを受け付けるように事前構成されている限り、コンピュータはアプリケーションおよびユーザ間でデータを共有することができる。しかし、これらの従来システムは拡張可能ではない。即ち、正しいスキーマが送り先装置側に用意されていないと、データは転送することができない。また、自由なデータ交換を維持す

るために、ユーザは、同じ追加または変更したタイプが送り先場所（サーバ／クライアント）にインストールされていないと、ユーザは発生元場所（例えば、クライアント／サーバ）でタイプを追加または変更することができない。即ち、送り先場所（例えば、サーバ／クライアント）は異種の装置および／またはアプリケーションから到来データを受け取るように準備されている必要がある（例えば、必要なスキーマ情報を保持している必要がある）。

【０００３】

コンピューティングシステムにおける開発は、データベーステクノロジの利点を利用するプラットフォームを採用することを目的としていた。多くの場合、これらの開発では、これらの利点はファイルシステムに組み込まれている。しかし、これらのシステムにも上述したのと同じスキーマ互換性の欠点がある。そのために、今日では、データを効果的に転送し、共有するためには、ファイルシステム内のデータは事前に定義された共通スキーマに準拠していなければならない。当然のことであるが、スキーマ(schema)とは、データのフォーム（例えば、構造）を宣言的方法で記述したものと定義することができる。

新たに出現したファイルシステムでは、オブジェクトはデータベース（例えば、ファイルシステム）に格納することが可能であり、従って、適用可能なスキーマで記述することが可能になっている。これらのファイルシステム内のデータは特定のスキーマとタイプのインスタンスであり、タイプはデータの形状（例えば、構造）を定義するスキーマに定義されている。新しいタイプをシステムに追加する必要が起ったときは、例えば、システムが取り扱うことができる新しいオブジェクトのセットまたは新しいデータのフォームを追加する必要が起ったときは、開発者はスキーマおよびそのスキーマ内にタイプを作成する必要がある。次に、プロパティがそのタイプに追加されることになる。

【発明の開示】

【発明が解決しようとする課題】

【０００４】

データをファイルシステムのストア（または他のデータベース）にセーブする必要があるが、そのデータに必要なスキーマ／タイプがまだインストールされていないときのシナリオはいくつかが存在する。これは、「ストアダウンレベル(store down-level)」問題として知られている。従来のシステムでは、システムアドミニストレータ（システム管理責任者）および／または特別な権限をもつ人だけがスキーマ情報をインストールすることが可能であった。

【課題を解決するための手段】

【０００５】

以下は、本発明のいくつかの態様の基本的理解を容易にするために、本発明の概要を簡単に説明したものである。以下の簡単な説明は、本発明の概要を網羅的に説明したものではない。また、本発明の主要／必須要素を特定することを意図したものでも、本発明の範囲を限定することを意図したものでもない。本発明の唯一の目的は、後述するより詳細な説明の序論として本発明のいくつかの概念を簡単に紹介することである。

【０００６】

ここで説明している本発明は、その一態様によれば、スキーマパケージをリソースとしてクライアント側アセンブリ(client-side assembly)に組み入れることを容易にしている。他の態様によれば、スキーマパッケージをサテライトアセンブル(satellite assemble)としてまたは別のファイルとして組み入れることを可能にしている。当然に理解されるように、スキーマ定義(schema definition)は、スキーマパッケージと呼ばれるドキュメントに収集することができる。例示のシナリオでは、システムは、未インストールのスキーマに関してクライアントがストアとやりとりするのを可能にするために必要なスキーマパッケージと情報を構築し、配布するのを容易にしている。

【０００７】

スキーマのジャストインタイム(just-in-time)インストールをサポートするために、署名されたスキーマパッケージがストアに提示されてインストールされることを可能にして

いる。スキーマパッケージにはいくつかの利点があるが、パッケージがインストールのために容易に利用可能になっていないと、これらの利点は無意味になってしまう。本発明の一態様による革新は、セキュアかつオンタイムで利用できることを保証するようにスキーマパッケージを構築し、配布するシステムおよび方法を目的としている。

【０００８】

本発明の一態様によれば、スキーマパッケージの構築期間に、ストロングネーム署名(strong name signing)オペレーションを採用することを可能にしている。別の態様では、スキーマパッケージの認証コード署名(authenticode signing)、圧縮および／または暗号化の採用を可能にしている。スキーマパッケージは、スキーマを実装したものを提供するクライアント側アセンブリ(client-side assembly)の一部として組み込むことができる。クライアント側アセンブリは周知の名前で特定することができ、スキーマがプログラミングによって使用できるすべての場所でＡＰＩが利用できるようにすることができる。即ち、クライアントアプリケーションがスキーマ化タイプ(schematized type)のインスタンスを生成してそのインスタンスがストアに残っているときは、いつでもクライアントはスキーマパッケージにアクセスすることが可能になっている。さらに別の態様では、必要とするスキーマがストアに用意されていない場合は、データの提示に先立ってスキーマパッケージをストアにインストールしておくことができる。

【０００９】

上述した目的および関連目的を達成するために、以下の説明と添付図面を参照して、本発明のいくつかの態様を例示して以下に説明する。なお、これらの態様は、本発明の原理を採用できる種々の方法のいつかを例示したものにすぎず、本発明には、かかる態様のすべておよびその同等態様が含まれるものである。本発明の他の利点と新規特徴は、図面を参照して以下に説明する本発明の詳細説明の中で明らかにする。

【発明を実施するための最良の形態】

【００１０】

以下、図面を参照して本発明について説明する。図面全体を通して類似の要素は類似の参照番号を付けて示されている。以下の説明において、本発明の完全な理解を容易にするために、説明の目的上、多数の具体的詳細が示されている。当然のことであるが、本発明はこれらの具体的詳細がなくても実施することが可能である。その他の例では、本発明の説明を容易にするために、周知の構造と装置が示されている。

【００１１】

本明細書の中で使用されている「コンポーネント(component)」および「装置(device)」という用語は、ハードウェアであるか、ハードウェアとソフトウェアの組み合わせであるか、ソフトウェアであるか、実行中のソフトウェアであるかに関係なく、コンピュータに関係するエンティティのことを指している。例えば、コンポーネントは、プロセッサ上で実行されるプロセス、プロセッサ、オブジェクト、実行可能形体のもの(executable)、実行スレッド(thread of execution)、プログラム、および／またはコンピュータにすることができるが、これらに限定されない。例示として、サーバ上で実行されるアプリケーションおよびサーバは共に、コンポーネントになることができる。１または２以上のコンポーネントは、プロセスおよび／または実行スレッド内に置いておくことができ、あるコンポーネントは１つのコンピュータ上に置いておくことも、および／または２または３以上のコンピュータ間に分散させることも可能である。

【００１２】

本明細書の中で使用されている「推論する(infer)」または「推論(inference)」という用語は、一般的に、イベントおよび／またはデータを通して収集した観察の集合からシステム、環境、および／またはユーザのステート（状態）について推理(reasoning)、つまり、ステートを推論(inferring)するプロセスのことを指している。推論は、特定のコンテキストまたはアクションを特定するために採用することができるが、例えば、ステートにわたる確率分布を生成することもできる。推論は確率的(probabilistic)にすることができる。すなわち、関心のあるステートにわたる確率分布の計算をデータおよびイベント

の考慮に基づいたものにすることができる。推論は、イベントおよび／またはデータの集合からより上位レベルのイベントを構成するために採用された手法を指すこともある。かかる推論の結果として、観察されたイベントおよび／または格納されたイベントデータの集合から新しいイベントまたはアクションが構築されるが、これらのイベントが時間的に近接した相関関係にあるかどうか、およびイベントおよびデータが１または複数のイベントおよびデータのソースから得られたものかどうかは無関係である。

【００１３】

最初に図１を参照して説明すると、図１はスキーマのパッケージング、配布および可用性を容易にするシステム１００を示す図である。一般的に、システム１００は、スキーマパッケージングコンポーネント(schema packaging component)１０２とスキーマ配布コンポーネント(schema distribution component)１０４を装備することができる。上述したように、システム１００は、システム１００のセキュリティと信頼性を維持したまま、該当スキーマの可用性を実現することができる。

【００１４】

よくあることであるが、ユーザはデータをあるコンピュータに置いておき、そのデータをアプリケーションおよびユーザの間で共有させたいことがある。しかし、事情によっては、必要なデータ（例えば、スキーマ）が用意されていないため、データの共有が行なえないことがある。従来システムがデータの共有を行なう方法には、いくつかの問題がある。上述したように、従来システムは拡張不能である。例えば、許可やアクセス上の問題に起因してタイプを追加することが困難なことがよくある。多くの場合、ファイルシステムはデータベースの利点を利用し、その利点をストレージファイルシステムに組み込むプラットフォームを採用している。そのために、ファイルシステム内のデータは、事前に定義され、事前にインストールされたスキーマに準拠することができる。すなわち、宣言的方法でデータのフォームを記述することができる。

【００１５】

新たに出現したファイルシステムでは、オブジェクトはデータベース（例えば、ファイルシステム）に格納され、特殊化されたスキーマによって記述されている。これらのファイルシステム内のデータはこれらのスキーマのタイプのインスタンスである。すなわち、タイプは、データの形状を定義しているスキーマ内に定義されている。ここで説明している例と態様はデータベーススキーマのシナリオを目的としているが、当然に理解されるように、その他のシナリオおよびここで説明している新規機能を実装したものが存在するので、適用可能なスキーマで他のドキュメントを記述することができる。これらの代替態様は、本明細書および明細書に付属する請求項の範囲に含まれるものである。

【００１６】

本発明の革新によれば、あるマシンから別のマシンへデータを無制限に転送することを可能にすると共に、データが第２マシンに到着したとき、いつでも、そのデータに関して必要なメータデータが第２マシンにすでに存在することを可能にしている。さらに、本発明の革新によれば、スキーマ情報に対してセキュリティを保持しながらスキーマ情報をコンポーネント間で調整することを可能にしている。

【００１７】

図１に戻って説明すると、本発明の態様は、スキーマのセキュアなパッケージング、配布および可用性を容易にするシステム１００（および／または方法）を目的としている。一態様では、スキーマパッケージングコンポーネント１０２によると、スキーマインストールの単位、すなわち、データの構造を記述するスキーマパッケージまたはパッケージだけ（例えば、スキーマドキュメント）を定義することができる。システム１００によって取り組むことのできる１つの問題として、重要なことは２つのスキーマが相互に独立して開発された場合、これらのスキーマが誤って衝突しないことを保証することである。即ち、重要なことは、送り先マシンから見たとき、２つのタイプのドキュメントが誤って同じドキュメントとして見えないことを保証することである。

【００１８】

前述した関連アプリケーションの中で説明されているようにスキーマのジャストインタイムインストールをサポートするために、スキーマパッケージングコンポーネント１０２によると、インストールに備えてストアに提示することができる、ユニークに署名されたスキーマパッケージを生成することができる。スキーマパッケージにはいくつかの利点があるが、パケージが容易にインストールできるようになっていないと、これらの利点は無意味になってしまう。配布コンポーネント１０４は、インストールを容易にすることができる。スキーマパッケージングコンポーネント１０２とスキーマ配布コンポーネント１０４の新規機能は、どちらも以下に詳しく説明する。

【００１９】

一般的に、本発明による革新は、その一態様によれば、データをストアに格納することを容易にするために非アドミニストレータ（非管理責任者）または無権限エージェントがスキーマをストアに投入できるようにすることを目的としている。具体的には、本発明の革新によれば、このシナリオが可能であるのは、スキーマがその所有者に対して権限があって、そのスキーマが別の所有者によって生成されたスキーマと衝突しない範囲においてである。

【００２０】

ここで説明している新規機能によれば、あるユーザがスキーマを生成したとき、別のユーザはそのスキーマを採用することができ、そのスキーマが元の所有者に対して権限があるものと判断することができる。さらに、ユーザは、そのスキーマがストアに投入された場合、既存のスキーマまたは別のユーザによって生成されたスキーマに不利な影響を与えないことを保証することができる。以上から理解されるように、もう１つの主要な新規概念は、スキーマパッケージとクライアント側アセンブリとの間の関係である。このような関係があると、スキーマパッケージをクライアント側アセンブリに対して常時利用可能にすることができる。

【００２１】

本発明の革新によれば、上述した新規の権限を有し、衝突防止機能を実現するために、いくつかのテクノロジが組み合わされている。本発明の態様によれば、スキーマパッケージがどのように構築され、配布されてその可用性が保証されるかが説明されている。以下の図面の説明から明であるように、本発明による革新の一態様の新規特徴は、暗号化して署名されたスキーマパッケージをクライアント側アセンブリにリンクできること、例えば、リソースとしてクライアント側アセンブリに組み込むことができることである。

【００２２】

次に図２を参照して説明すると、図２はシステム１００の代替アーキテクチャを示すブロック図である。新規スキーマパッケージング、配布および可用性システムの態様によれば、スキーマパッケージングコンポーネント１０２によるスキーマパッケージの構造は、ストロングネーム署名コンポーネント(strong name signing component)２０２を使用して、ストロングネーム署名オペレーションを実行することを可能にしている。ここでストロングネーム署名とは、他のすべてのスキーマからスキーマをユニークに定義し、不正変更チェックを行い、バージョン系統(version lineage)を保護する能力のことである。従って、ストロングネーム署名に関して、各々のスキーマは、スキーマ自体の定義と緊密に結合されたユニーク名をもつことができる。異なる定義および／または署名をもつスキーマが他にもあることから、スキーマがユニークでなければならないのはそのためである。

【００２３】

スキーマをその元の形体で送信するのではなく、スキーマをスキーマパッケージングコンポーネント１０２によって「パッケージ化」すると、その内容を存続したままスキーマにいくつかの望ましい特性を持たせることができる。一態様では、スキーマパッケージはスキーマ定義言語(schema definition language SDL)ドキュメント（例えば、拡張可能マークアップ言語(extensible markup language XML)にすることができる。さらに、理解すべきことは、スキーマパッケージはどのようなカスタムまたは拡張可能コードも含んでいないことである。従って、そのようなスキーマはスキーマインストールの唯一の単位

になっている。図２に図示のように、一態様では、ストロングネームには、スキーマのフレンドリネーム、バージョン、カルチャおよびスキーマに署名するために使用される暗号鍵ペアの公開鍵部分を含めることができる。

【００２４】

次に図３を参照して説明すると、図３はスキーマのパッケージング、配布および可用性を容易にする代替システム３００を示す図である。図示のように、ストロングネーム署名コンポーネント２０２のほかに、スキーマパッケージングコンポーネント１０２は、オプションとして認証コード署名コンポーネント(authenticode signing component)３０２、圧縮コンポーネント(compression component)３０４、および／または暗号化コンポーネント(encryption component)３０６をスキーマパッケージの構造の一部として採用することができる。認証コード署名コンポーネント３０２は特定のストロングネームスキーマを特定の個人または個人の集合に付けることができる。従って、このスキーマはそのストロングネームに対して権限をもつだけでなく、その物理的実際の個人または組織作成者に対しても権限を持つことができる。従って、認証コードプロバイダは、署名によるストロングネームとスキーマの作成者である認証コードエンティティとの間の対応付け(mapping)を保証することができる。

【００２５】

圧縮コンポーネント３０４はトランスポートを容易にし、効率化することができる。例えば、圧縮は、ストアとやりとりする可能性のあるクライアントと一緒に発送される、あるいはバックアップをとるために、あるいはインポート／エキスポートオペレーション、コピーオペレーションまたは同期化のためにデータがストアから取り出されるときデータと一緒に発送されるコードにスキーマパッケージを組み入れるときに利用すると、特に便利である。いずれも場合も、スキーマがデータに付けられるとき、スキーマを圧縮すると、スキーマをデータと一緒にトランスポートするときのオーバヘッドが軽減されるので特に好都合である。この圧縮は、オプションとして、圧縮コンポーネント３０４を通して実行することもできる。

【００２６】

さらに、暗号化は、オプションとして暗号化コンポーネント３０６を通して採用することができる。この暗号化は、スキーマがデータと一緒に転送されるときセキュリティを容易にすることができる。スキーマを所有権主張(proprietary)できるようにすると特に好都合であるケースがあるときに、暗号化メカニズムを採用することができる。従って、スキーマ定義は送信途中にあるときや、スキーマパッケージが検査されることがあるときに隠されて、見えないようにされる。以上の説明はスキーマパッケージを定義し、本発明の一態様によるスキーマパッケージのプロパティを明らかにしたものであり、そこでは、パッケージは、上記プロパティ（例えば、認証コード、圧縮、暗号化）の１または２以上を含むストロングネーム署名スキーマになっている。

【００２７】

スキーマパッケージが構築されたあと、本発明の新規特徴によれば、そのスキーマパッケージは、スキーマの実装を可能にするクライアント側アセンブリの一部として組み込まれる（またはクライアント側アセンブリと関係付けられる）。また、スキーマは周知の名前で特定されて、スキーマがプログラミングによって使用できるすべての場所でＡＰＩに利用できるようにされる。

【００２８】

以上から理解されるように、上述したシステム３００は、クライアントアプリケーションがスキーマ化タイプのインスタンスを生成してそのインスタンスがストアに残されているときは、いつでもクライアントがスキーマパッケージにアクセスできることを保証する。必要とするスキーマがそのときストアに利用可能になっていないときは、スキーマパッケージはスキーマ配布コンポーネント１０４を通して見つけられ、利用可能にされたあと、データをストアに提示する前に、（例えば、前述した関連アプリケーションの中で説明されているスキーマインストールコンポーネントを通して）最終的にストアにインストー

ルされることになる。

【００２９】

即ち、スキーマパッケージは、データを作成および／または操作し、ストレージとやりとりするアプリケーションであるクライアント側アセンブリ内に組み込むこと（またはそのアセンブリとリンクさせること）ができる。特定の一態様では、クライアント側アセンブリには、スキーマの定義をリソースとして組み込むことができる。これとは別に、スキーマにアセンブリとの別の固有の関係(intrinsic relationship)を持たせることによって、見つけやすくし、利用しやすくすることができる。

【００３０】

図４は、スキーマパッケージング、ディスカバリおよび可用性を容易にするさらに別の代替システム４００を示す図である。具体的には、図４は、スキーマパッケージ４０２とクライアント側アセンブリ４０４が、同じストロングネーム署名コンポーネントによって同じストロングネームで署名できることを示している。このことから理解されるように、複数タイプのデータを操作するために複数のスキーマが必要になる範囲において、これらの追加スキーマも、クライアント側アセンブリ内に組み込むこと（またはそのアセンブリと関係付けること）ができる。各々のパッケージは、周知の名前、すなわち、スキーマに署名すると生成されるストロングネームで特定することができる。アプリケーション自体が実行されるためには、コードが実行されるすべての場所にコードがなければならないので、ストアとやりとりするために必要なスキーマパッケージは、コードが利用可能であるときは、常にパッケージはリソースとしてそのコードの中にあるので、ストアに残っていなければならない。

【００３１】

インスタンスがファイルシステムのストアにセーブされるためには、その前にそのスキーマ（およびそのインスタンスのオブジェクトツリーの中のすべてのタイプのスキーマ）が存在していなければならない。上述したように、ストアにスキーマが存在しない場合には、それをインストールする必要がある。本発明のスキーマパッケージング、配布および可用性によれば、必要とするスキーマの可用性とインストールはセキュアに行なうことができる。

【００３２】

次に図５を参照して説明すると、図５は本発明の一態様に従ってスキーマにストロングネーム署名する方法を示す図である。説明を簡単にする目的上、例えば、図５にフローチャートまたはフロー図の形で示されている１または２以上の方法は一連のアクト(act)として示され、説明されているが、当然に理解されるように、あるアクトは、本発明によれば、ここに図示され、説明されているものと異なる順序でおよび／または他のアクトと同時並行的に実行されることがあるので、本発明はアクトの順序によって制限されない。例えば、この分野の当業者ならば理解されるように、方法は、例えば、ステート図における相互に関係する一連のステートまたはイベントとして表わすことも可能である。さらに、本発明による方法を実現するために図示のアクトすべてが必要になるとは限らない。

【００３３】

図５に図示のように、ストロングネームでスキームに署名するために、５０２において、スキームのハッシュ値(hash value)が計算される。一例では、セキュアハッシュアルゴリズム(Secure Hash Algorism SHAI)のハッシュを採用することができる。次に、５０４において、ハッシュ値は秘密鍵(private key)を使用して非対称暗号法(asymmetric cipher)（例えば、ＲＳＡ非対称暗号法）で署名される。５０６において、署名されたハッシュ値は対応するＲＳＡ公開鍵と一緒にスキーマに格納される。

【００３４】

次に図６を参照して説明すると、図６にはストロングネームで署名されたスキーマを検証する方法が示されている。ストロングネームで署名されたスキーマを検証するために、６０２において、格納されたハッシュ値は格納された鍵（例えば、ＲＳＡ鍵）を使用して解読される。別のハッシュ値は、６０４においてスキーマの内容から独立に生成される。

６０６において、２つのハッシュ値が一致しているかどうかの判断が行なわれる。２つのハッシュ値が一致していなければ、ストップブロックに到達する。他方、２つのハッシュ値が一致していれば、検証は６０８において正しいものとみなされる。

【００３５】

有効なスキーマパッケージは、請求項に記載のストロングネームの有効性、内容の保全性およびパブリッシャの認証性の程度を保証することができる。即ち、スキーマは元の所有者に対して権限があるものとみなすことがきる。以上から理解されるように、簡潔性（例えば、圧縮）および追加セキュリティ（例えば、暗号化および／または認証コード署名）などのプロパティを達成するために、追加アクト（例えば、図３のオプションのコンポーネントによる）をパッケージング期間に適用することができる。これらの追加態様は、当然に本明細書の開示事項および明細書に付属の請求項の範囲に含まれるものである。

【００３６】

コンパイルしてクライアント側アセンブリに組み込まれるファイルを生成するツール（図示せず）をスキーマ定義言語、この例ではＸＭＬと併用することができる。このクライアント側アセンブリは、特定のタイプに定義されたクラスとのリンクになるものである。即ち、クライアント側アセンブリは、プログラミングによってクラスへのアクセスを可能にする。

【００３７】

従って、クライアント側アセンブリとスキーマパッケージの間には非常に緊密な関係を存在させることができる。即ち、スキーマパッケージは、クライアント側アセンブリに組み込まれていなければ（例えば、リソースとして）、即時に利用可能にすることができる。クライアント側アセンブリは、データを見るためにプログラミングによるアクセスを実現したものであり、スキーマパッケージまたは定義はストアとの契約である。以上を要約すると、本発明による革新の新規特徴は、タイプを実現するクライアント側アセンブリ（例えば、コード）とストア内のタイプを記述するスキーマパッケージとの間に固有の関係(intrinsic relationship)および／またはリンケージをもたせることである。リソースはこの関係を達成する１つのメカニズムである。別の態様では、スキーマパッケージは静的変数(static variable)に格納することができ、この変数もコンパイルされてアセンブリに組み込まれているか、あるいは別のファイルに置かれていて、アセンブリと一緒に配布されるか、あるいは別の周知の場所またはキャッシュに置かれている。

【００３８】

次に図７を参照して、引き続きスキーマパッケージングコンポーネント１０２について説明するが、ファイルシステム（例えば、WinFSブランド）のスキーマの開発には、クライアント側アセンブリとスキーマパッケージの両方の開発が含まれている。互換性を達成するために、一態様では、クライアント側アセンブリ４０４とスキーマパッケージ４０２は、共に配備に先立って同じストロングネームを採用することによって署名コンポーネント２０２によって署名することができる。別の態様では、コンパイル時にも事後コンパイル時（遅延署名）にも鍵ペアを見つけて、パッケージに署名するために開発ツール（例えば、ディスカバリコンポーネント７０２とローディングコンポーネント７０４）を採用することができる。ここで理解すべきことは、遅延署名(delay signing)環境は、スキップリスト(skip list)保守とクエリのためのインフラストラクチャによってサポートできることである。

【００３９】

以下の説明では、スキーマパッケージ４０２とクライアント側アセンブリ４０４が特定の共通鍵によって署名されることを提案している。クライアント側アセンブリ４０４とスキーマパッケージ４０２との間の関係を保証するために、本発明による革新は、同じ鍵を使用した（例えば、ストロングネーム署名コンポーネント２０２による）クライアント側アセンブリの署名を要求することによって新規の関係を開示している。このように同じ鍵を使用すると、クライアント側アセンブリ４０４とスキーマパッケージ４０２の間に固有の関係が実現される。この鍵は、作成者だけが秘密鍵にアクセスできるので、作成者は、

スキーマパッケージ４０２がクライアント側アセンブリ４０４に対して権限があることを確認する秘密鍵にアクセスしている必要があることを示している。

【００４０】

上述したように、本メカニズムの１つの新規の特徴は、スキーマパッケージ４０２をリソースとしてクライアント側アセンブリ４０４に組み込むことができることである。スキーマパッケージ４０２をクライアント側アセンブリ４０４に組み込んでおくと、クライアント側アセンブリが署名されるとき、スキーマパッケージが署名のためにいつでも利用できることが保証される。別の態様では、鍵ペアは単一メカニズムを通して開発ツールに提示することができる。

【００４１】

コンパイル時署名期間には、アセンブリに署名するために使用された同じ鍵ファイルを、アセンブリのコンパイル期間にスキーマパッケージに署名するために使用できる。スキーマパッケージはアセンブリ内部のリソースとして見えるので、スキーマパッケージが最初に署名され、アセンブリに組み込まれてからアセンブリが署名される。ここで理解すべきことは、これらのオペレーションはカスタムビルドタスク(custom build task)（例えば、MSBuildブランドのタスクを使用して）にラップ(wrap)できることである。さらに、ユーザインタフェース(user interface - UI)層が用意されていると、上述したアクトのいくつかを自動化することによってユーザの体験を単純化することができる。

【００４２】

秘密鍵は非常に保護されているので、開発環境において、遅延署名(delay signing)を実行するためにスキップリスト(skip list)を使用することができる。遅延署名のとき、スキーマパッケージングコンポーネント１０２は遅延署名されたパッケージ（すなわち、正しいハッシュ値なしで構築されたパッケージ）をサポートすることができる。スキップリストが用意されていれば、スキーマパッケージスキップリストエントリを記述するために利用できる。例を挙げると、スキップリストは別の類似メカニズムのアセンブリスキップリストにすることができる。この態様によれば、スキップリストに記載されているスキーマパッケージは、いずれもインストール時には有効性検査されない。

【００４３】

この環境では、開発者が秘密鍵にアクセスできなくても、開発者は、秘密鍵にアクセスしていたものとして依然としてシステムで作業することができる。開発が完了すると、最終的署名プロセスが実行されることにより、クライアント側アセンブリとスキーマパッケージはその鍵を使用して署名することができる。

【００４４】

以上から理解されるように、スキップリストは、特定のマシン上のアセンブリ、スキーマまたはその両方を有効性検査するストロングネームチェックに成功すると働くメカニズムである。即ち、スキップリストによれば、ユーザがこのコンポーネントを見たとき、該当の秘密鍵で署名されなかったことを無視できるシナリオを可能にしている。

【００４５】

遅延署名開発環境では、アセンブリが最終的に署名されるときでも、アセンブリが署名される前にスキーマパッケージを完全に署名する必要が起ることがある。別の部門が鍵を管理し、構築済みのアセンブリに署名する完了ステップを実行するような開発環境では、スキーマパッケージとそのアセンブリが２ステップのプロセスで署名されることが頻繁に行なわれている。最初のステップでは、スキーマパッケージは署名のために署名部門に渡されている。これに応じて、署名パッケージはビルドツリーにチェックインされる。このチェックインされた署名パッケージは、スキーマ変更がチェックインされるとき、いつでも新しい署名パッケージで更新することができる。

【００４６】

チェックインされた署名スキーマパッケージは後続のビルド(build)で使用することができる。「ゴールデンビット(Golden Bits)」ビルドは、そのあと、すでに署名スキーマパッケージを収めているアセンブリ署名のために署名部門に送られる。別の態様では、こ

のプロセスは１ステップのプロセスで行なうことも可能である。例えば、署名部門に渡されたアセンブリは、遅延署名されたが、完全に署名されていないスキーマパッケージをリソースとしてすでに収めていることも可能である。この態様によれば、署名部門は、スキーマパッケージを抽出し、そのパッケージに署名し、署名されたパッケージをアセンブリに再挿入し、最終的にアセンブリに署名するツールを使用することも可能である。

【００４７】

次に、クライアント側アセンブリがどこに格納され、それがスキーマパッケージと一緒にどのようにしてロードできるかを説明するが、図７は、ディスカバリコンポーネント７０２とローディングコンポーネント７０４がスキーマ配布コンポーネント１０４と一体になっているシステム７００を示す図である。これらのコンポーネントの各々の新規機能は以下に詳しく説明する。

【００４８】

クライアントアプリケーションがスキーム化タイプのインスタンスを生成してそのインスタンスをストアに残しておくとき、クライアントは、例えば、必要なスキーマがまだストアに利用可能になっていない場合には、データを提示する前にスキーマパッケージをストアに提示しなければならないことがある。スキーマパッケージを提示するには、クライアントはそのスキーマパッケージを見つけて、ロードできなければならない。

【００４９】

ローディングコンポーネント７０４に関しては、採用できるメカニズムには、スタティックバインディング(static binding)とダイナミックバインディング(dynamic binding)の２タイプがある。しかし、スキーマパッケージは、ストアに入れて残しておくためには、その前に所在を突き止めること（例えば、見つけること）が必要である。従って、グローバルアクセスキャッシュ(global access cache GAC)やローカルストアまたはローカルファイルシステムに置かれているクライアント側アセンブリおよび/またはパッケージを見つけるためにディスカバリコンポーネント７０２を採用することができる。即ち、例示の態様では、ユーザがアプリケーションをどこに配備できるかには２つの選択が可能である。すなわち、ＧＡＣまたはローカルである。場所の判断要因としては、どれだけのユーザをアプリケーションから見えるようにするかがある。同様に、アプリケーションを特定タイプに対してどのようにコーディングするかについても、２つの選択がある。これらのシナリオの各々は以下に説明する。

【００５０】

配備のための第１の選択はＧＡＣであり、これはアセンブリがいつでもすべてのユーザに見えるようにされることを意味する。ＧＡＣまたはグローバルアセンブリキャッシュは、アプリケーション（またはその一部）がそこに配備されて、すべてのユーザがそのアプリケーション（またはその一部）にアクセスして使用できるようにするグローバルリソースである。第２の選択はローカル(Local)であり、これはアプリケーション（またはその一部）が特定の場所にセーブされ、アクセスと使用に関してシングルユーザ（または有限のユーザグループ）に限定されることを意味している。

【００５１】

コーディングに関しては、スタティックバインディングとダイナミックバインディングの２つの選択が用意されている。ユーザがタイプについて分かっていて、その分かっているタイプに対してコードを明示的に書く場合は、このことはスタティックバインディングと呼ばれる。他方、ダイナミックバインディングとは、ユーザは特定のタイプについて分かっていないが、コードが実行されるとき、特定のタイプを見つけるためにシステムにクエリできる（例えば、ディスカバリコンポーネント７０２を通して）状況のことである。タイプが見つかると、ユーザはそのタイプに対してダイナミックにプログラミングすることができる。

【００５２】

下表は、ディスカバリコンポーネント７０２とローディングコンポーネント７０４の新規機能が２つの判断のいずれかの相互セクションに適用できることを示している。

【００５３】
【表１】

	スタティックバインディング	ダイナミックバインディング
ＧＡＣ	そのアセンブリがアドミニストレータによってＧＡＣにインストールされ、コンパイル時に全てのタイプにスタティックにバインドされるアプリケーション	そのアセンブリがアドミニストレータによってＧＡＣにインストールされ、ランタイム時にダイナミックに又はプログラミングによってタイプを見つけるアプリケーション
ローカル	そのアセンブリが非アドミニストレータによってファイルシステムにインストールされ、コンパイル時に全てのタイプにスタティックにバインドされるアプリケーション	そのアセンブリが非アドミニストレータによってファイルシステムにインストールされ、ランタイム時にダイナミックに又はプログラミングによってタイプを見つけるアプリケーション

【００５４】
以上から理解されるように、共通言語ランタイム(common language runtime CLR)プラットフォーム上で開発されたコードがコンピュータにまたがって複数のアプリケーションによって共有されるときは、そのコードはＧＡＣと呼ばれるマシンワイドキャッシュ(machine-wide cache)に置かれていることがよくある。ＧＡＣに置かれているアセンブリは、異なるコードバージョンを並行に実行させることを可能にする特定のバージョン管理スキーム(versioning scheme)に準拠している必要がある。
【００５５】
この態様において、ＧＡＣにインストールされたアプリケーションに要求されることは、安全かつマシングローバルな方法でクライアント側アセンブリとスキーマパッケージの両方を配備することである。このことから理解されるように、ＣＬＲは上記要件をＧＡＣとそのインストールサービスの形で満足させるサービスを提供しなければならない。さらに理解されるように、これらのサービスは、ＧＡＣが採用されない場合には、スキーマパッケージ用に複製されている必要がある。
【００５６】
別の態様に関しては、ローカルアプリケーションはそのアセンブリを周知の場所にインストールしている。かかるアプリケーションのスキーマパッケージは、理論的には同じ場所に置いておくことが可能である。しかし、実際には、当然に理解されるように、アプリケーションは一連の異なるコンポーネントから構築することができる。各々のコンポーネント自体は、１または２以上の周知場所にインストールすることができる。ＣＬＲには複雑なバインディングルールが存在するので、これらの複数場所から特にシームレスな方法でこれらのアセンブリのディスカバリとローディングを容易にしている。このことから理解されるように、バインディングインフラストラクチャは、既存のインフラストラクチャが活用できない場合には、スキーマパッケージ用に複製されている必要がある。
【００５７】
スタティックにバインドされたアプリケーションはコンパイル時にクラスをインスタンス生成する。すなわち、これらのアプリケーションは、特定クラスの変数を宣言し、そのクラスの新しいインスタンスを作成するために新しいオペレータ（演算子）を使用する明示のコードラインを収めている。これらのクラスについては、特定タイプのバインディングはリンク時に完全修飾されたストロングネームアセンブリ(fully-qualified, strong-named assembly)に分解され、これは後にロードされる（フュージョンローディングポリシ(fusion loading policy)に従って）。スタティックアプリケーションは、特定のアセンブリのセットにバインドされる時に、つまり、リンク時に特定のスキーマパッケージの集

合にバインドされる。

【００５８】

他方、ダイナミックにバインドされたアプリケーションは、その時の条件に基づいてランタイム時にルーチンまたはオブジェクトにリンクされる。ダイナミックアプリケーションは、ユーザ発意によるディスカバリの結果としてタイプをダイナミックにロードする。例えば、Visual Studioブランドのアプリケーションは、コントロールを収めたアセンブリを、フォーム設計フェーズ期間にユーザが選択することを可能にしている。この例では、アセンブリがロードされ、そのあと、コントロールはコントロールツールバーを通してインスタンス生成することができる。

【００５９】

また、ダイナミックアプリケーションはタイプをプログラミングでロードできることがしばしばである。例を挙げると、Ｗｏｒｄブランドのアプリケーションは、登録されたアドイン(Add-In)のリストを維持することができる。そのあと、各々のアドインはプログラミングでそのリストからロードされ、そのあとインスタンス生成される。

【００６０】

オペレーション時には、ダイナミックアプリケーションは、ランタイム時にのみ特定のスキーマパッケージの集合にバインドされる。さらに、このスキーマパッケージの集合は、あるランから次のランまでおよび同じランにおいて経時的に、常に流動的にすることができる。どちらのタイプのアプリケーションの場合も、例えば、スタティックアプリケーションまたはダイナミックアプリケーションでは、あるタイプのスキーマパッケージは、クライアント側アセンブリがディスカバリ可能であるすべての事情において、ディスカバリ可能でなければならない。これは本メカニズムの新規特徴である。即ち、本発明による革新の新規特徴によれば、スキーマパッケージをクライアント側アセンブリに組み込むことによって、そのアセンブリを見つけるために使用された全く同じインフラストラクチャがスキーマパッケージを見つけるときに梃子にできることが保証されている。

【００６１】

正しくオペレーションするためには、アプリケーションは、インスタンス生成することが予想されるすべてのタイプについてクライアント側アセンブリとスキーマパッケージを配備しなければならない。インストールおよび配備インフラストラクチャは、アセンブリをＧＡＣとローカルの両方に配備することができる。当然に理解されるように、あるアプリケーションのスキーマパッケージの配備に失敗すると、クロスレベルのバージョン管理シナリオ(cross-level versioning scenario)において、すなわち、必要とするスキーマが置かれていないストアにインスタンスが持続的に置かれるようなシナリオにおいて、アプリケーションが失敗することになる。

【００６２】

クロスレベルバージョン管理シナリオは、その性質上前向きであり、複雑さと費用を伴うために、ソフトウェア出荷前に十分にテストされるのがまれのことがときどきある。ソフトウェア開発プロセスが設計時にクロスレベルバージョン管理シナリオに要求条件を課していない場合には、その要求条件はテスト時と開発時の期間に看過されると、未テストのシナリオでは現場でソフトウェアに障害が起ることがある。そのために、ソフトウェア開発プロセスが設計時に十分な制約条件を課していて、ここで用意されたインフラストラクチャを使用したこれらのシナリオでは配備が容易にされ、エラーのないようにすることが設計上の要求条件になっている。スキーマパッケージをクライアント側アセンブリに組み込むと、アセンブリの配備のための全く同じインフラストラクチャが梃子になってスキーマパッケージが配備されることが保証される。以上から理解されるように、開発者および配備ツールは、アセンブリ配備によって起る問題に精通していることがよくある。開発時間チェックを含めると、スキーマパッケージが正しくコンパイルされて、クライアント側アセンブリにリソースとして組み込まれたことを保証することができる。ランタイムチェックも同じような理由で好都合である。

【００６３】

別の態様では、上述したように、複数スキーマパッケージをクライアント側アセンブリ内に組み込むことができる。従って、リソースの名前付けに関しては、いずれかの特定スキーマに対応するスキーマパッケージは、その特定スキーマパッケージのために使用されたストロングネームに基づくリソース名との１対１の対応付けによって特定することができる。

【００６４】

以上から理解されるように、複数スキーマパッケージのシナリオでは、クライアント側アセンブリとスキーマパッケージとの間を関係付けるためには、マッチングと署名プロパティ(matching and signing properties)が満たされることになる。即ち、異なるストロングネームをもつ複数スキーマパッケージが存在する場合であっても、クライアント側アセンブリとスキーマとの間の関係を保持するために、すべてが同じ鍵で署名することが可能になっている。すなわち、アセンブリに署名するために１つの鍵だけが使用できる。

【００６５】

前述したように、オプションとして、別の態様に対して圧縮、暗号化および認証コード署名を採用することができる。スキーマファイルはサイズが大きくなる可能性があるので、ファイルサイズの管理をしやすくするために圧縮が採用されることがよくある。別の態様では、圧縮はスキーマが署名される前でも、後でも適用することができる。スキーマに署名した後で圧縮を適用することが特に好ましい理由として、少なくとも次の２つがある。第１の理由は、オリジナルのＸＭＬ内容についてハッシュが行なわれるのが通常であるからである。第２の理由は、タイミングのために署名ブロックも圧縮されるので、圧縮が良好化するのが通常であるからである。スキーマ検証期間には、スキーマパッケージは最初に圧縮復元（伸張）されてから検証される。

【００６６】

代替態様によれば、スキーマパッケージはサテライトアセンブリ(satellite assembly)にすることができる。このアプローチでは、スキーマはクライアント側アセンブリに直接に組み込まれていない。むしろ、スキーマは公然のリソース(manifest resource)として別のサテライトアセンブリになっている。署名プロセスでは、メインアセンブリとサテライトアセンブリは共に署名部門に渡すことができ、署名部門は最初にスキーマファイルに署名し、次にサテライトアセンブリ全体にハッシュするメインアセンブリに署名する。

【００６７】

別の代替態様では、スキーマパッケージは別のファイルとして維持することが可能になっている。スキーマパッケージ自体は開発者によって別のビルドステップとして作成され、インストーラによって別ファイルとして配備され、ローダ／バインダによって別々に見つけられることになる。開発プロセスは、本発明ではシームレスに行なうことができた別々のステップを開発者に実行させる必要があることが理由で、支障が生じる可能性がある。さらに、この態様の配備プロセスは、同期の乱れが生じる可能性のある余計なステップが必要になることが理由で、支障が生じる可能性がある。スキーマパッケージの正しい配備にこのようなミスマッチがあったり、無視されたりすると、多くの場合、出荷前に十分にテストされないようなシナリオにおいてはアプリケーションの一部に障害が起ることになる。ディスカバリプロセスでは、グローバル配備とダイナミックディスカバリという問題に取り掛かるためには追加のインフラストラクチャが必要になる。これらの代替態様は、当然のことながら、本明細書の開示事項および本明細書に付属の請求項の範囲内に含まれるものである。

【００６８】

上述したように、上述したシステムによれば、どのような公開／秘密暗号化鍵ペアでも採用することができる。１つの例において、公開鍵暗号化手法はスキーマパッケージに名前を付けるために（および署名するために）採用することができる。ここで公開鍵暗号化とは、公開部分と秘密部分を含む、２部分の鍵（例えば、ユニークなコード）を使用する暗号化方法と言うことができる。メッセージを暗号化するためには、送信者だけに分かっている未公開の秘密鍵が使用される。従って、メッセージを解読するには、受信側は送信

側の公開された秘密鍵を使用する。即ち、公開鍵とは、２部分の公開鍵暗号化システムの公開された部分と言うことができる。

【００６９】

鍵ペアの秘密部分は所有者にだけ分かっている。従って、スキーマ作成者は未公開の秘密鍵を使用して、スキーマパッケージを暗号化して署名することができる。この暗号化セキュリティ方法は、スキーマ情報の認証性と保全性を大幅に向上することができる。このことから当然に理解されるように、データにユニークな名前を付けおよび／または署名するどのような方法も、本発明および請求項の精神と範囲から逸脱しない限り採用することが可能である。

【実施例】

【００７０】

次に図８を参照して説明すると、図８は、スキーマパッケージング、配布および可用性の開示されたアーキテクチャを実行するために動作可能であるコンピュータを示すブロック図である。本発明の種々態様を詳しく説明するために、図８と以下の説明は、本発明の種々態様を実現することができる適当なコンピューティング環境８００の概要を要約したものである。本発明は、１または２以上のコンピュータ上で実行されるコンピュータ実行可能命令の一般的コンテキストの中で上述されているが、この分野の精通者ならば理解されるように、本発明は他のプログラムモジュールと組み合わせておよび／またはハードウェアとソフトウェアの組み合わせとして実現することも可能である。

【００７１】

一般的に、プログラムモジュールには、特定のタスクを実行する、または特定の抽象データ型を実現するルーチン、プログラム、コンポーネント、データ構造などが含まれている。さらに、この分野の精通者ならば理解されるように、本発明の方法は他のコンピュータシステム構成と共に実施することが可能であり、その中には、シングルプロセッサまたはマルチプロセッサコンピュータシステム、ミニコンピュータ、メインフレームコンピュータだけでなく、パーソナルコンピュータ、ハンドヘルドコンピューティング装置、マイクロプロセッサベースまたはプログラマブルコンシューマエレクトロニクスなどが含まれており、これらの各々は１または２以上の関連装置に動作可能に結合することが可能になっている。

【００７２】

本発明の図示態様は分散コンピューティング環境で実施されることもあり、そこでは、ある種のタスクは通信ネットワークを通してリンクされたリモート処理装置によって実行されている。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は、ローカルとリモートの両方のメモリストレージ装置に置いておくことができる。

【００７３】

コンピュータは種々のコンピュータ可読媒体を備えているのが代表的である。コンピュータ可読媒体は利用可能であれば、コンピュータによってアクセス可能である、どのような媒体にすることも可能であり、その中には、揮発性と不揮発性の両方の媒体、取り外し可能媒体と取り外し不能媒体が含まれている。例を挙げると、コンピュータ可読媒体には、コンピュータ記憶媒体と通信媒体があるが、これらに限定されない。コンピュータ記憶媒体には、コンピュータ可読命令、データ構造、プログラムモジュールまたは他のデータなどの情報を格納するための、いずれかの方法またはテクノロジで実現された揮発性と不揮発性の両方の取り外し可能媒体と取り外し不能媒体が含まれている。コンピュータ記憶媒体としては、ＲＡＭ、ＲＯＭ、ＥＥＰＲＯＭ、フラッシュメモリや他のメモリテクノロジ、ＣＤ－ＲＯＭ、デジタルビデオディスク（ＤＶＤ）や他の光ディスクストレージ、磁気カセット、磁気テープ、磁気ディスクストレージや他の磁気ストレージ装置、あるいは必要とする情報を格納するために使用でき、コンピュータによってアクセス可能である他の媒体がある。

【００７４】

通信媒体は、コンピュータ可読命令、データ構造、プログラムモジュールまたは他のデ

ータを、搬送波や他のトランスポートメカニズムなどの変調データ信号で具現化しているのが代表的であり、その中には、あらゆる情報配信媒体が含まれている。ここで「変調データ信号(modulated date signal」という用語は、その特性の１つまたは２つ以上が信号の中で情報を符号化するようにセットまたは変更された信号を意味している。例を挙げると、通信媒体には、ワイヤド（有線）ネットワークや直接ワイヤドコネクションのようなワイヤド媒体、および音響、ＲＦ、赤外線などのワイヤレス（無線）媒体や他のワイヤレス媒体が含まれているが、これらに限定されない。上記に挙げたものを任意に組み合わせたものも、コンピュータ可読媒体の範囲に含まれることは当然である。

【００７５】

再び図８を参照して説明すると、本発明の種々態様を実現するための例示環境８００にはコンピュータ８０２が含まれており、コンピュータ８０２は処理ユニット８０４、システムメモリ８０６およびシステムバス８０８を装備している。システムバス８０８は、システムメモリ８０６（これに限定されない）を含むシステムコンポーネントを処理ユニット８０４に結合している。処理ユニット８０４は、商用化されている種々のプロセッサのいずれにすることもできる。処理ユニット８０４としてデュアルマイクロプロセッサおよび他のマルチプロセッサアーキテクチャを採用することも可能である。

【００７６】

システムバス８０８はいくつかのタイプのバス構造のいずれにすることも可能であり、このバス構造は、商用化されている種々のバスアーキテクチャのいずれかを使用したメモリバス（メモリコントローラの有無に関係ない）、周辺バス、およびローカルバスにさらに相互接続することが可能になっている。システムメモリ８０６には、リードオンリメモリ８１０およびランダムアクセスメモリ８１２が含まれている。基本入出力システム(BIOS)は、ＲＯＭ、ＥＰＲＯＭ、ＥＥＰＲＯＭなどの不揮発性メモリ８１０に格納されており、ＢＩＯＳは、スタートアップ時のようにコンピュータ内の要素間の情報転送を支援する基本ルーチンから構成されている。ＲＡＭ８１２には、データをキャッシュするスタティックＲＡＭのような、高速ＲＡＭも含まれている。

【００７７】

コンピュータ８０２は、内部ハードディスクドライブ（ＨＤＤ）８１４（例えば、ＥＩＤＥ、ＳＡＴＡ）（なお、この内部ハードディスクドライブは適当なシャーシ（図示せず）内で外部使用するように構成されていることもある）、磁気フロッピディスクドライブ（ＦＤＤ）８１６（例えば、取り外し可能ディスケット８１８との間で読み書きする）および光ディスクドライブ８２０（例えば、ＣＤ－ＲＯＭディスク８２２を読み取り、あるいはＤＶＤのような他の高容量光媒体との間で読み書きする）をさらに装備している。ハードディスクドライブ８１４、磁気ディスクドライブ８１６および光ディスクドライブ８２０は、それぞれハードディスクドライブインタフェース８２４、磁気ディスクドライブインタフェース８２６および光ドライブインタフェース８２８によってシステムバス８０８に接続可能になっている。外部ドライブ装着用のインタフェース８２４には、ユニバーサルシリアルバス(USB)およびＩＥＥＥ１３９４インタフェーステクノロジのうちの少なくとも一方または両方が含まれている。その他の外部ドライブ接続テクノロジは本発明の意図する範囲に含まれるものである。

【００７８】

これらのドライブおよびそれぞれの関連コンピュータ可読媒体は、データ、データ構造、コンピュータ実行可能命令などを格納しておく不揮発性ストレージである。コンピュータ８０２の場合には、これらのドライブおよび媒体は、いずれかのデータを適当なデジタルフォーマットで格納するのに適している。上述したコンピュータ可読媒体の説明では、ＨＤＤ、取り外し可能磁気ディスケット、およびＣＤやＤＶＤのような取り外し可能光媒体が挙げられているが、この分野の精通者ならば当然に理解されるように、Ｚｉｐドライブ、磁気カセット、フラッシュメモリカード、カートリッジなどのように、コンピュータによって読み取り可能である他のタイプの媒体も、例示動作環境で使用することが可能であり、さらに、そのような媒体のいずれにも、本発明の方法を実行するためのコンピュー

タ実行可能命令を収めておくことが可能である。

【００７９】

複数のプログラムモジュールをドライブおよびＲＡＭ８１２に格納しておくことができ、その中には、オペレーティングシステム８３０、１または２以上のアプリケーションプログラム８３２、その他のプログラムモジュール８３４およびプログラムデータ８３６が含まれている。オペレーティングシステム、アプリケーション、モジュール、および／またはデータのすべてまたは一部をＲＡＭ８１２にキャッシュすることもできる。以上から理解されるように、本発明は、商用化されている種々のオペレーティングシステムまたはオペレーティングシステムの組み合わせで実現することができる。

【００８０】

ユーザは、１または２以上のワイヤド（有線）／ワイヤレス（無線）インプット装置、例えば、キーボード８３８およびマウス８４０などのポインティング装置を通してコマンドと情報をコンピュータ８０２に入力することができる。他のインプット装置（図示せず）としては、マイクロホン、ＩＲリモートコントロール、ジョイスティック、ゲームパッド、スタイラスペン、タッチスクリーンなどがある。これらのインプット装置および他のインプット装置は、多くの場合、システムバス８０８に結合されたインプット装置インタフェース８４２を通して処理ユニット８０４に接続されているが、パラレルポート、ＩＥＥＥ１３９４シリアルポート、ゲームポート、ＵＳＢポート、ＩＲインタフェースなどの、他のインタフェースによって接続することも可能である。

【００８１】

モニタ８４４または他のタイプのディスプレイ装置も、ビデオアダプタ８４６などのインタフェースを介してシステムバス８０８に接続されている。モニタ８４４のほかに、コンピュータは、スピーカ、プリンタなどの他の周辺アウトプット装置（図示せず）を備えているのが代表的である。

【００８２】

コンピュータ８０２は、リモートコンピュータ８４８のような１または２以上のリモートコンピュータとの論理的コネクションをワイヤドおよび／またはワイヤレス通信を通して使用するネットワーキング環境で動作することができる。リモートコンピュータ８４８はワークステーション、サーバコンピュータ、ルータ、パーソナルコンピュータ、ポータブルコンピュータ、マイクロプロセッサベースのエンターテイメントアプライアンス、ピア装置または他の共通ネットワークノードにすることができ、コンピュータ８０２に関連して上述した要素の多くまたはすべてを装備しているのが代表的である。なお、図には説明を簡略化するために、メモリ／ストレージ装置８５０だけが示されている。図示の論理的コネクションには、ローカルエリアネットワーク（ＬＡＮ）８５２および／または例えば、ワイドエリアネットワーク（ＷＡＮ）８５４のような大規模ネットワークが含まれている。このようなＬＡＮおよびＷＡＮネットワーキング環境はオフィスおよび企業では普通になっており、イントラネットのような社内コンピュータネットワークを容易にし、これらのすべては、例えば、インターネットのようなグローバル通信ネットワークに接続されていることがある。

【００８３】

ＬＡＮネットワーキング環境で使用されるときは、コンピュータ８０２は、ワイヤドおよび／またはワイヤレス通信ネットワークインタフェースまたはアダプタ８５６を通してローカルネットワーク８５２に接続されている。アダプタ８５６はＬＡＮ８５２とのワイヤドまたはワイヤレス通信を容易にし、ＬＡＮ８５２には、ワイヤレスアダプタ８５６と通信するためのワイヤレスアクセスポイントが置かれていることもある。

【００８４】

ＷＡＮネットワーキング環境で使用されるときは、コンピュータ８０２はモデム８５８を装備することができるが、さもなければＷＡＮ８５４上の通信サーバに接続されているか、あるいはインターネットなどを介してＷＡＮ８５４上の通信を確立するための他の手段を備えている。モデム８５８は内蔵型と外付け型があり、ワイヤドまたはワイヤライン

装置としてシリアルポートインタフェース８４２を介してシステムバス８０８に接続されている。ネットワーキング環境では、コンピュータ８０２に関連して図示したプログラムモジュールまたはその一部は、リモートメモリ／ストレージ装置８５０に格納しておくことができる。以上から理解されるように、図示のネットワークコネクションは例示であり、コンピュータ相互間に通信リンクを設定する他の手段を使用することができる。

【００８５】

コンピュータ８０２は、ワイヤレス通信で動作可能に配置された、いずれかのワイヤレス装置またはエンティティと通信するように動作可能であり、そのような装置として、例えば、プリンタ、スキャナ、デスクトップおよび／またはポータブルコンピュータ、ポータブルデータアシスタント、通信衛星、ワイヤレスに検出可能なタグと関連付けられた機器部品または場所（例えば、キオスク、ニューススタンド、レストルーム）、および電話がある。この中には、少なくともWi-FiおよびBluetooth（登録商標）ワイヤレステクノロジが含まれている。従って、通信は従来のネットワークの場合と同様に事前定義の構造にすることも、あるいは単純に少なくとも２装置間のアドホック(ad hoc)通信にすることも可能である。

【００８６】

Wi-Fi、つまり、Wireless Fidelityは、自宅のソファー、ホテルルームのベッド、または作業中の会議室からワイヤレスにインターネットに接続することを可能にしている。Wi-Fiは、セル電話で使用されて、前記装置、例えば、コンピュータが基地局の範囲内のどこにいても、室内および室外でデータを送受信することを可能にするテクノロジに類似したワイヤレステクノロジである。Wi-FiネットワークはＩＥＥＥ802.11(a,b,gなど)と呼ばれる無線テクノロジを使用して、セキュアで信頼性のある高速のワイヤレス接続性が得られるようにしている。Wi-Fiネットワークは、コンピュータを相互に接続し、インターネットに接続し、ワイヤドネットワーク（IEEE 802.3またはイーサネット(登録商標）を使用）に接続するために使用することができる。Wi-Fiネットワークは無認可の2.4および5 GHz無線バンドにおいて、例えば、11 Mbps(802.11a)または54 Mbps(802.11b)データレートで、または両バンドを含むプロダクトと共に動作し、多くのオフィスで使用されている基本10BaseTワイヤドイーサネット(登録商標）ネットワークにほぼ同等のリアルワールドのパフォーマンスがネットワークから得られるようにする。

【００８７】

次に図9を参照して説明すると、図９は本発明のスキーマパッケージング、配布および可用性システムによる例示コンピューティング環境900を示す概略ブロック図である。システム９００には、１または２以上のクライアント９０２が含まれている。クライアント９０２はハードウェアおよび／またはソフトウェア（例えば、スレッド、プロセス、コンピューティング装置）にすることができる。クライアント９０２には、例えば、本発明を採用することによってクッキー(cookie)および／または関連コンテキスト情報を収容することができる。

【００８８】

システム９００には、１または２以上のサーバ９０４も含まれている。サーバ９０４も、ハードウェアおよび／またはソフトウェア（例えば、スレッド、プロセス、コンピューティング装置）にすることができる。サーバ９０４には、例えば、本発明を採用することによって変換(transformation)を実行するスレッドを収容することができる。クライアント９０２とサーバ９０４との間の１つの可能な通信は、２または３以上のコンピュータプロセス間で送信されるのに適したデータパケットの形にすることができる。データパケットには、例えば、クッキーおよび／または関連コンテキスト情報が含まれていることがある。システム９００には、クライアント９０２とサーバ９０４との間の通信を容易にするために採用できる通信フレームワーク９０６（インターネットのようなグローバル通信ネットワーク）が含まれている。

【００８９】

通信はワイヤド（光ファイバを含む）および／またはワイヤレステクノロジを通して容

易にすることができる。クライアント９０２は、クライアント９０２に特有の情報（例えば、クッキーおよび／または関連コンテキスト情報）を格納するために採用できる１または２以上のクライアントデータストア９０８に動作可能に接続されている。同様に、サーバ９０４は、サーバ９０４に特有の情報を格納するために採用できる１または２以上のサーバデータストア９１０に動作可能に接続されている。

【００９０】

以上、本発明のいくつかの例を含めて説明してきた。当然のことながら、本発明を説明するためにコンポーネントまたは方法の想到し得る組み合わせをすべて説明することは不可能であるが、通常の知識を有するものならば理解されるように、本発明のさらに多くの組み合わせと置換が可能である。従って、本発明は、請求項に記載の本発明の精神と範囲に属するすべての変更、改良および変形を包含することを意図している。さらに、「含む(include)」という用語が詳細な説明または請求項の中で用いられている限りにおいて、この用語は、「含むまたは備えた(comprising)」という用語が請求項の中で移行語として使用されるときに解釈されるように、「comprising」と同じように包含的なものである。

【図面の簡単な説明】

【００９１】

【図１】本発明の一態様に従ってスキーマのパッケージング、配布および可用性を容易にするシステムを示すブロック図である。

【図２】本発明による革新の一態様に従ってスキーマデータに署名するストロングネーム署名コンポーネントを採用したシステムを示すブロック図である。

【図３】本発明による革新の一態様に従ってオプションとして認証コードコンポーネント、圧縮コンポーネントおよび暗号化コンポーネントを採用したシステムを示すブロック図である。

【図４】本発明による革新の一態様に従ってスキーマパッケージとクライアント側アセンブリの同じ暗号化署名を採用したシステムを示す図である。

【図５】本発明の一態様に従ってスキーマのストロングネーム署名を実行する方法の例示フローを示す図である。

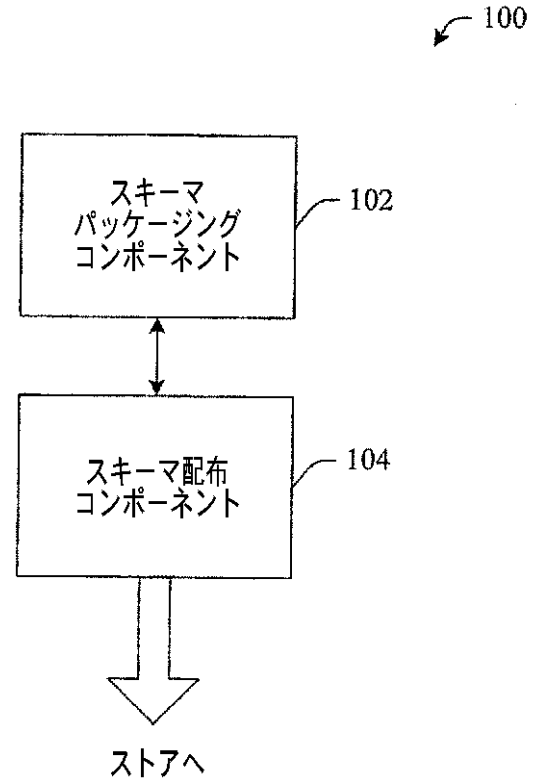
【図６】本発明の一態様に従ってストロングネーム署名されたスキーマの解読を実行する方法の例示フローを示す図である。

【図７】本発明による革新の一態様に従ってスキーマの配布を実行するためにディスカバリコンポーネントとローディングコンポーネントを採用したシステムを示すブロック図である。

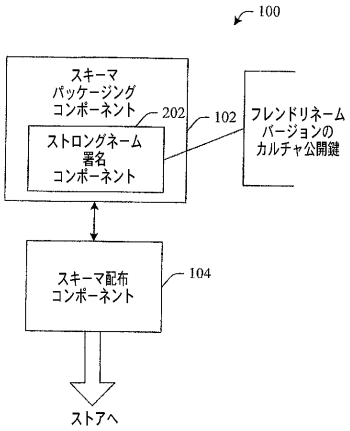
【図８】開示されているアーキテクチャを実行するように動作可能であるコンピュータを示すブロック図である。

【図９】本発明による例示コンピューティング環境を示す概略ブロック図である。

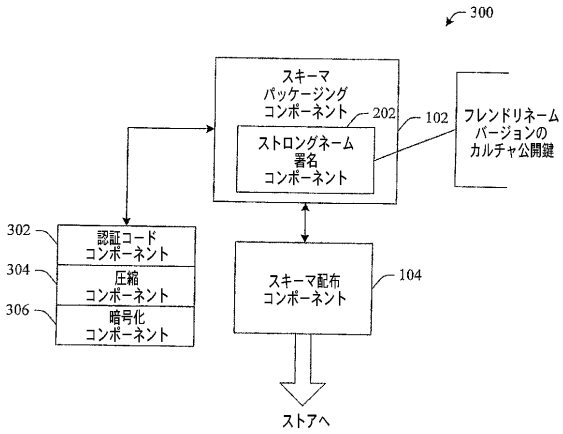
【図１】


100
スキーマ
パッケージング
コンポーネント
102
スキーマ配布
コンポーネント
104
ストアへ

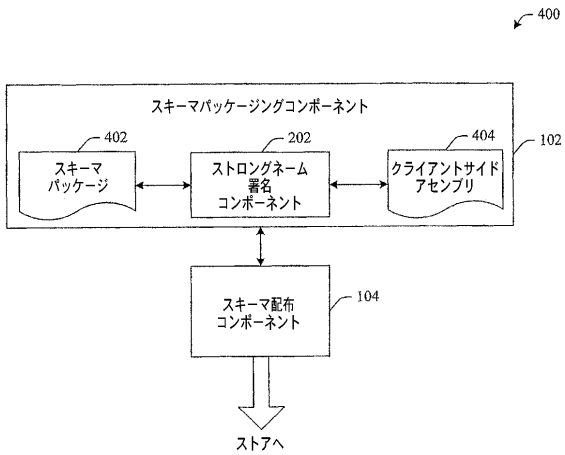
【図２】


100
スキーマ
パッケージング
コンポーネント
202
102
フレンドリネーム
バージョンの
カルチャ公開鍵
ストロングネーム
署名
コンポーネント
スキーマ配布
コンポーネント
104
ストアへ

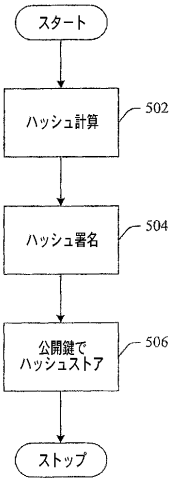
【図３】


300
スキーマ
パッケージング
コンポーネント
202
102
フレンドリネーム
バージョンの
カルチャ公開鍵
ストロングネーム
署名
コンポーネント
302
認証コード
コンポーネント
304
圧縮
コンポーネント
306
暗号化
コンポーネント
スキーマ配布
コンポーネント
104
ストアへ

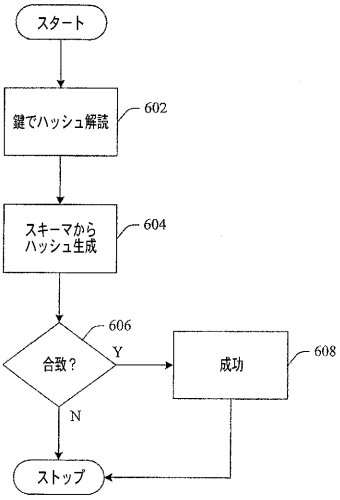
【図４】


400
スキーマパッケージングコンポーネント
402
スキーマ
パッケージ
202
ストロングネーム
署名
コンポーネント
404
クライアントサイド
アセンブリ
102
スキーマ配布
コンポーネント
104
ストアへ

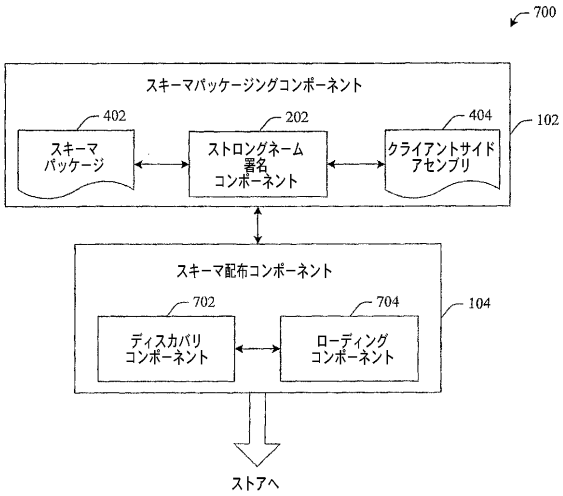
【図５】


スタート
ハッシュ計算
502
ハッシュ署名
504
公開鍵で
ハッシュストア
506
ストップ

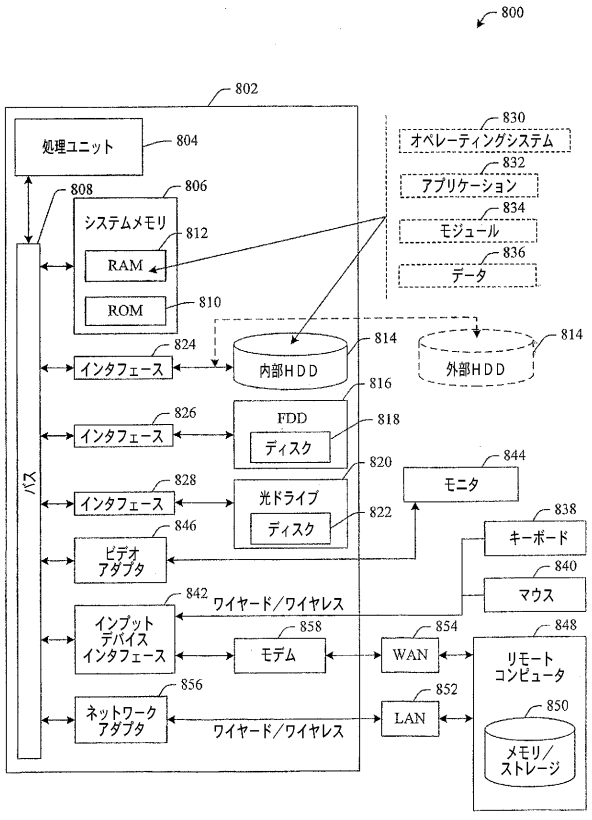
【図６】


スタート
鍵でハッシュ解読
602
スキーマから
ハッシュ生成
604
606
合致？
Y
成功
608
N
ストップ

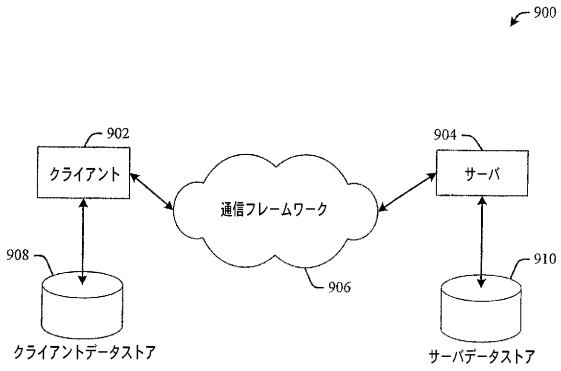
【図７】


700
スキーマパッケージングコンポーネント
402
スキーマ
パッケージ
202
ストロングネーム
署名
コンポーネント
404
クライアントサイド
アセンブリ
102
スキーマ配布コンポーネント
702
ディスカバリ
コンポーネント
704
ローディング
コンポーネント
104
ストアへ

【図８】


800
802
処理ユニット
804
システムメモリ
806
808
RAM
812
ROM
810
バス
インタフェース
824
内部HDD
814
外部HDD
814
FDD
ディスク
816
818
インタフェース
826
光ドライブ
ディスク
820
822
インタフェース
828
ビデオ
アダプタ
846
インプット
デバイス
インタフェース
842
ワイヤード／ワイヤレス
モデム
858
WAN
854
ネットワーク
アダプタ
856
ワイヤード／ワイヤレス
LAN
852
リモート
コンピュータ
848
メモリ／
ストレージ
850
オペレーティングシステム
830
アプリケーション
832
モジュール
834
データ
836
モニタ
844
キーボード
838
マウス
840

【図９】


900
902
クライアント
904
サーバ
通信フレームワーク
906
908
クライアントデータストア
910
サーバデータストア

【国際調査報告】

INTERNATIONAL SEARCH REPORT

International application No.
PCT/US06/28386

A. CLASSIFICATION OF SUBJECT MATTER
IPC: **G06F 7/00**(2006.01),**17/00**(2006.01)

USPC: 707/103R
According to International Patent Classification (IPC) or to both national classification and IPC

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)
U.S. : 707/103r

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched
Google, ACM

Electronic data base consulted during the international search (name of data base and, where practicable, search terms used)
east

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category *	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
X --- Y	Namje et al., ACM workshop on XML Security 2003, Washington D.C., USA, "Certificate Validation Service using XKMS for computation Grid", hereafter (Namje).	1-3, 5-8, 10-12, 14-20 --- 4,9, 13
Y	US 2002/0131428 A (PECUS et al.) 19 September 2002 (19.09.2002), paragraphs 0041-0046)	4, 9 and 13

☐ Further documents are listed in the continuation of Box C. ☐ See patent family annex.

* Special categories of cited documents:

"A" document defining the general state of the art which is not considered to be of particular relevance

"E" earlier application or patent published on or after the international filing date

"L" document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)

"O" document referring to an oral disclosure, use, exhibition or other means

"P" document published prior to the international filing date but later than the priority date claimed

"T" later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention

"X" document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone

"Y" document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art

"&" document member of the same patent family

Date of the actual completion of the international search	Date of mailing of the international search report
07 July 2007 (07.07.2007)	12 OCT 2007
Name and mailing address of the ISA/US Mail Stop PCT, Attn: ISA/US Commissioner for Patents P.O. Box 1450 Alexandria, Virginia 22313-1450 Facsimile No. (571) 273-3201	Authorized officer Cindy Nguyen Jacqueline A. Whitfield Special Project Asst. Telephone No. 571-272-4025

Form PCT/ISA/210 (second sheet) (April 2005)

INTERNATIONAL SEARCH REPORT

International application No.
PCT/US06/28386

Box No. II Observations where certain claims were found unsearchable (Continuation of item 2 of first sheet)

This international search report has not been established in respect of certain claims under Article 17(2)(a) for the following reasons:

1. ☒ Claims Nos.: 1-10
because they relate to subject matter not required to be searched by this Authority, namely:
Please See Continuation Sheet

2. ☐ Claims Nos.:
because they relate to parts of the international application that do not comply with the prescribed requirements to such an extent that no meaningful international search can be carried out, specifically:

3. ☐ Claims Nos.:
because they are dependent claims and are not drafted in accordance with the second and third sentences of Rule 6.4(a).

Box No. III Observations where unity of invention is lacking (Continuation of item 3 of first sheet)

This International Searching Authority found multiple inventions in this international application, as follows:

1. ☐ As all required additional search fees were timely paid by the applicant, this international search report covers all searchable claims.
2. ☐ As all searchable claims could be searched without effort justifying additional fees, this Authority did not invite payment of any additional fees.
3. ☐ As only some of the required additional search fees were timely paid by the applicant, this international search report covers only those claims for which fees were paid, specifically claims Nos.:

4. ☐ No required additional search fees were timely paid by the applicant. Consequently, this international search report is restricted to the invention first mentioned in the claims; it is covered by claims Nos.:

Remark on Protest

☐ The additional search fees were accompanied by the applicant's protest and, where applicable, the payment of a protest fee.

☐ The additional search fees were accompanied by the applicant's protest but the applicable protest fee was not paid within the time limit specified in the invitation.

☐ No protest accompanied the payment of additional search fees.

INTERNATIONAL SEARCH REPORT

International application No.
PCT/US06/28386

Box II Observations where certain claims were found unsearchable 1. because they relate to subject matter not required to be searched by this Authority, namely:

Claim 1-10 are rejected under 35 U.S.C. 101 because the claimed invention is directed to non-statutory subject matter. The claim recites a system in the preamble only, the body of the claim merely contains a packaging component and a distribution component. which can be software modules. Therefore, the claim is software per se and not statutory (computer programs claimed as computer listing per se, i.e., the descriptions of expressions of the programs, are not physical "things". They are neither computer components nor statutory processes, as they are not "acts" being performed.

フロントページの続き

(81)指定国 AP(BW,GH,GM,KE,LS,MW,MZ,NA,SD,SL,SZ,TZ,UG,ZM,ZW),EA(AM,AZ,BY,KG,KZ,MD,RU,TJ,TM),EP(AT,BE,BG,CH,CY,CZ,DE,DK,EE,ES,FI,FR,GB,GR,HU,IE,IS,IT,LT,LU,LV,MC,NL,PL,PT,RO,SE,SI,SK,TR),OA(BF,BJ,CF,CG,CI,CM,GA,GN,GQ,GW,ML,MR,NE,SN,TD,TG),AE,AG,AL,AM,AT,AU,AZ,BA,BB,BG,BR,BW,BY,BZ,CA,CH,CN,CO,CR,CU,CZ,DE,DK,DM,DZ,EC,EE,EG,ES,FI,GB,GD,GE,GH,GM,HN,HR,HU,ID,IL,IN,IS,JP,KE,KG,KM,KN,KP,KR,KZ,LA,LC,LK,LR,LS,LT,LU,LV,LY,MA,MD,MG,MK,MN,MW,MX,MZ,NA,NG,NI,NO,NZ,OM,PG,PH,PL,PT,RO,RS,RU,SC,SD,SE,SG,SK,SL,SM,SY,TJ,TM,TN,TR,TT,TZ,UA,UG,US,UZ,VC,VN,ZA,ZM,ZW

(72)発明者 ディヴィッド ジェイ．ネトルトン
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 ソン シュエ
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

Ｆターム(参考) 5B017 AA08 BA09
5J104 AA08 AA09 JA03 JA21 LA02 LA03 LA06 NA02 NA12 NA27
NA37 NA38 PA14