

(19) 日本国特許庁(JP)

(12) 公開特許公報(A)

(11) 特許出願公開番号

特開2005-174283

(P2005-174283A)

(43) 公開日 平成17年6月30日(2005.6.30)

(51) Int. Cl.⁷

G06F 9/44

F I

G06F 9/06 620A

テーマコード (参考)

5B076

審査請求 未請求 請求項の数 42 O L 外国語出願 (全 46 頁)

(21) 出願番号 特願2004-264627 (P2004-264627)
 (22) 出願日 平成16年9月10日 (2004.9.10)
 (31) 優先権主張番号 10/693, 718
 (32) 優先日 平成15年10月24日 (2003.10.24)
 (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 9805
 2-6399 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 クリス ジェイ. ガザク
 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

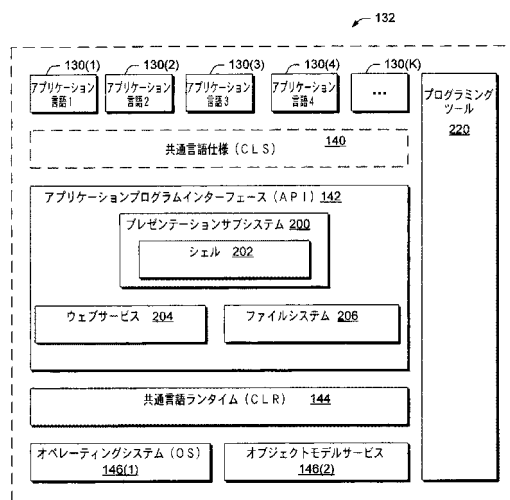
(54) 【発明の名称】 コンピュータプラットフォームのプログラミングインターフェース

(57) 【要約】 (修正有)

【課題】 .NET (商標) Framework の効率および/または性能を高める形を提供する。

【解決手段】 コンピュータプラットフォームのプログラミングインターフェースに、再利用可能なユーザインターフェースコントロールに関連するサービスの第1群と、ユーザインターフェースダイアログおよびユーザインターフェースウィザードに関連するサービスの第2群と、ユーザインターフェース機能性の拡張に関連するサービスの第3群と、ユーザインターフェースのデスクトップの機能性の拡張に関連するサービスの第4群の1つまたは複数を含める。

【選択図】 図2



【特許請求の範囲】**【請求項 1】**

1 つまたは複数のコンピュータ可読媒体で実施されるプログラミングインターフェースであって、

再利用可能なユーザインターフェースコントロールに関連するサービスの第 1 群と、
ユーザインターフェースダイアログおよびユーザインターフェースウィザードに関連するサービスの第 2 群と、

ユーザインターフェース機能性の拡張に関連するサービスの第 3 群と、

前記ユーザインターフェースのデスクトップの機能性の拡張に関連するサービスの第 4 群と

10

を備えたことを特徴とするプログラミングインターフェース。

【請求項 2】

サービスの前記第 1 群は、

ストレージユーザ経験をカプセル化する第 1 コントロールと、

アプリケーション定義の方法でアイテムを表示できるようにする第 2 コントロールと、

前記デスクトップのサイドバーにアイテムを追加できるようにする第 3 コントロールと

、

リストにアイテムを追加できるようにする第 4 コントロールと、

アイテムのプレビューイメージを表示できるようにする第 5 コントロールと

を含むことを特徴とする請求項 1 に記載のプログラミングインターフェース。

20

【請求項 3】

サービスの前記第 2 群は、

ファイルおよびフォルダを開き、および保存できるようにする第 1 ダイアログと、

光ディスクに書き込めるようにする第 1 ウィザードと、

電子メールによってイメージを送信するのを容易にする第 2 ウィザードと

を含むことを特徴とする請求項 1 に記載のプログラミングインターフェース。

【請求項 4】

サービスの前記第 3 群は、

コンテキストメニューへの追加を可能にする第 1 機能性と、

アプリケーション定義のサムネイルの識別を可能にする第 2 機能性と、

30

1 つまたは複数のファイルまたはフォルダに関する情報を表示するときに、計算を実行できるようにする第 3 機能性と

を含むことを特徴とする請求項 1 に記載のプログラミングインターフェース。

【請求項 5】

サービスの前記第 4 群は、

サイドバーを前記デスクトップ上で表示できるようにする第 1 機能性と、

アプリケーション定義の通知を前記デスクトップで表示できるようにする第 2 機能性と

を含むことを特徴とする請求項 1 に記載のプログラミングインターフェース。

【請求項 6】

ユーザインターフェースの再利用可能なコントロールを可能にする機能の第 1 セットを公開する手段と、

40

前記ユーザインターフェースの再利用可能なダイアログおよび前記ユーザインターフェースの再利用可能なウィザードを可能にする機能の第 2 セットを公開する手段と、

前記ユーザインターフェースのデスクトップの機能性の拡張を可能にする機能の第 3 セットを公開する手段と

を備えたことを特徴とするシステム。

【請求項 7】

前記ユーザインターフェースの機能性の拡張を可能にする機能の第 4 セットを公開する手段

をさらに備えたことを特徴とする請求項 6 に記載のシステム。

50

【請求項 8】

機能の前記第 4 セットを公開する前記手段は、
コンテキストメニューへの追加を可能にする 1 つまたは複数の機能を公開する手段と、
アプリケーション定義のサムネイルの識別を可能にする 1 つまたは複数の機能を公開する手段と、

1 つまたは複数のファイルまたはフォルダに関する情報を表示するときに計算を実行できるようにする 1 つまたは複数の機能を公開する手段と
を含むことを特徴とする請求項 7 に記載のシステム。

【請求項 9】

機能の前記第 1 セットを公開する前記手段は、
ストレージユーザ経験をカプセル化する 1 つまたは複数の機能を公開する手段と、
アプリケーション定義の方法でアイテムを表示できるようにする 1 つまたは複数の機能を公開する手段と、

前記デスクトップのサイドバーにアイテムを追加できるようにする 1 つまたは複数の機能を公開する手段と、

リストにアイテムを追加できるようにする 1 つまたは複数の機能を公開する手段と、
アイテムのプレビューイメージを表示できるようにする 1 つまたは複数の機能を公開する手段と

を含むことを特徴とする請求項 6 に記載のシステム。

【請求項 10】

機能の前記第 2 セットを公開する前記手段は、
ファイルおよびフォルダを開き、および保存できるようにする 1 つまたは複数の機能を公開する手段と、

光ディスクに書き込めるようにする 1 つまたは複数の機能を公開する手段と、

電子メールによって送信されるイメージのサイズを変更できるようにする 1 つまたは複数の機能を公開する手段と

を含むことを特徴とする請求項 6 に記載のシステム。

【請求項 11】

機能の前記第 3 セットを公開する前記手段は、
サイドバーを前記デスクトップに表示できるようにする 1 つまたは複数の機能を公開する手段と、

アプリケーション定義の通知を前記デスクトップに表示できるようにする 1 つまたは複数の機能を公開する手段と

を含むことを特徴とする請求項 6 に記載のシステム。

【請求項 12】

ユーザインターフェースに関する型のセットを階層名前空間に編成する方法であって、
型の前記セットから複数の群を作成するステップであって、各群が、論理的に関連する機能性を公開する少なくとも 1 つの型を含むステップと、

前記複数の群の各群に名前を割り当てるステップであって、前記複数の群の中の 1 つの群が、再利用可能なユーザインターフェースコントロールに関連する機能性を含み、前記複数の群の中のもう 1 つの群が、再利用可能なユーザインターフェースダイアログおよび再利用可能なユーザインターフェースウィザードに関連する機能性を含みステップと、

トップレベル識別子を選択し、前記トップレベル識別子を用いて各群の前記名前にプレフィックスを付け、その結果、各群の前記型が、前記型を含む前記群の名前にプレフィックスとして付けられた前記選択されたトップレベル識別子を含む階層名によって参照されるようにするステップと

を備えたことを特徴とする方法。

【請求項 13】

前記複数の群の中のもう 1 つの群は、ユーザインターフェース機能性の拡張に関連する機能性を含むことを特徴とする請求項 12 に記載の方法。

10

20

30

40

50

【請求項 14】

ユーザインターフェース機能性の拡張に関連する前記機能性は、
コンテキストメニューへの追加を可能にする第1機能性と、
サムネイルのアプリケーション識別を可能にする第2機能性と、
1つまたは複数のファイルまたはフォルダに関する情報を表示するときに計算を実行できるようにする第3機能性と
を含むことを特徴とする請求項13に記載の方法。

【請求項 15】

前記割り当てるステップは、ユーザインターフェース機能性の拡張に関連する機能性を含む前記群に `Add in` という名前を割り当て、その結果、ユーザインターフェース機能性の拡張に関連する機能性を含む前記群の前記階層名が、`System.Windows.Explorer.Add in` になるようにするステップを含むことを特徴とする請求項13に記載の方法。

【請求項 16】

前記複数の群の中のもう1つの群は、ユーザインターフェースデスクトップの機能性の拡張に関連する機能性を含むことを特徴とする請求項12に記載の方法。

【請求項 17】

ユーザインターフェースデスクトップの機能性の拡張に関連する前記機能性は、
前記デスクトップにサイドバーを表示できるようにする第1機能性と、
アプリケーション定義の通知を前記デスクトップに表示できるようにする第2機能性と
を含むことを特徴とする請求項16に記載の方法。

【請求項 18】

前記割り当てるステップは、ユーザインターフェースデスクトップの機能性の拡張に関連する機能性を含む前記群に `Desktop` という名前を割り当て、その結果、ユーザインターフェースデスクトップの機能性の拡張に関連する機能性を含む前記群の階層名が、`System.Windows.Explorer.Desktop` になるようにするステップを含むことを特徴とする請求項16に記載の方法。

【請求項 19】

前記割り当てるステップは、再利用可能なユーザインターフェースコントロールに関連する機能性を含む前記群に `Controls` という名前を割り当て、その結果、再利用可能なユーザインターフェースコントロールに関連する機能性を含む前記群の階層名が、`System.Windows.Explorer.Controls` になるようにするステップと、再利用可能なユーザインターフェースダイアログおよび再利用可能なユーザインターフェースウィザードに関連する機能性を含む前記群に `Dialogs` という名前を割り当て、その結果、再利用可能なユーザインターフェースダイアログおよび再利用可能なユーザインターフェースウィザードに関連する機能性を含む前記群の階層名が、`System.Windows.Explorer.Dialogs` になるようにするステップとを含むことを特徴とする請求項12に記載の方法。

【請求項 20】

再利用可能なユーザインターフェースコントロールに関連する前記機能性は、
ストレージユーザ経験をカプセル化する第1コントロールと、
アプリケーション定義の方法でアイテムを表示できるようにする第2コントロールと、
アイテムをリストに追加できるようにする第3コントロールと、
アイテムのプレビューイメージを表示できるようにする第4コントロールと
を含むことを特徴とする請求項12に記載の方法。

【請求項 21】

再利用可能なユーザインターフェースダイアログおよび再利用可能なユーザインターフェースウィザードに関連する前記機能性は、
ファイルおよびフォルダを開き、および保存できるようにする第1ダイアログと、
光ディスクに書き込めるようにする第1ウィザードと、

10

20

30

40

50

電子メールによってイメージを送信するのを容易にする第 2 ウィザードとを含むことを特徴とする請求項 1 2 に記載の方法。

【請求項 2 2】

ユーザインターフェースの機能性の拡張を可能にする機能を有する第 1 名前空間を作成するステップと、

前記ユーザインターフェースのデスクトップの機能性の拡張を可能にする機能を有する第 2 名前空間を作成するステップと

を備えたことを特徴とする方法。

【請求項 2 3】

再利用可能なユーザインターフェースコントロールを可能にする機能を有する第 3 名前空間を作成するステップ

をさらに備えたことを特徴とする請求項 2 2 に記載の方法。

【請求項 2 4】

前記第 3 名前空間は、

ストレージユーザ経験をカプセル化する第 1 コントロールと、

アプリケーション定義の方法でアイテムを表示できるようにする第 2 コントロールと、

アイテムを前記デスクトップのサイドバーに追加できるようにする第 3 コントロールと

、

アイテムをリストに追加できるようにする第 4 コントロールと、

アイテムのプレビューイメージを表示できるようにする第 5 コントロールと

を含むことを特徴とする請求項 2 3 に記載の方法。

【請求項 2 5】

ユーザインターフェースダイアログおよびユーザインターフェースウィザードを可能にする機能を有する第 3 名前空間を作成するステップ

をさらに備えたことを特徴とする請求項 2 2 に記載の方法。

【請求項 2 6】

前記第 3 名前空間は、

ファイルおよびフォルダを開き、および保存できるようにする第 1 ダイアログと、

光ディスクに書き込めるようにする第 1 ウィザードと、

電子メールによってイメージを送信するのを容易にする第 2 ウィザードと

を含むことを特徴とする請求項 2 5 に記載の方法。

【請求項 2 7】

前記第 1 名前空間は、

コンテキストメニューへの追加を可能にする第 1 機能性と、

アプリケーション定義のサムネイルの識別を可能にする第 2 機能性と、

1 つまたは複数のファイルまたはフォルダに関する情報を表示するときに計算を実行できるようにする第 3 機能性と

を含むことを特徴とする請求項 2 2 に記載の方法。

【請求項 2 8】

前記第 2 名前空間は、

前記デスクトップにサイドバーを表示できるようにする第 1 機能性と、

アプリケーション定義の通知を前記デスクトップに表示できるようにする第 2 機能性と

を含むことを特徴とする請求項 2 2 に記載の方法。

【請求項 2 9】

ユーザインターフェースのコントロールを使用する 1 つまたは複数の第 1 機能呼び出すステップと、

前記ユーザインターフェースの機能性を拡張する 1 つまたは複数の第 2 機能呼び出すステップと

を備えたことを特徴とする方法。

【請求項 3 0】

10

20

30

40

50

前記ユーザインターフェースのデスクトップの機能性を拡張する 1 つまたは複数の第 3 機能呼び出すステップ

をさらに備えたことを特徴とする請求項 29 に記載の方法。

【請求項 31】

前記デスクトップの機能性を拡張する前記機能は、

前記デスクトップにサイドバーを表示できるようにする 1 つまたは複数の機能と、

前記デスクトップにアプリケーション定義の通知を表示できるようにする 1 つまたは複数の機能と

を含むことを特徴とする請求項 30 に記載の方法。

【請求項 32】

前記ユーザインターフェースのダイアログおよびウィザードを使用する 1 つまたは複数の第 3 機能呼び出すステップ

をさらに備えたことを特徴とする請求項 29 に記載の方法。

【請求項 33】

前記ユーザインターフェースのダイアログおよびウィザードを使用する前記機能は、

ファイルおよびフォルダを開き、および保存できるようにするダイアログを使用する 1 つまたは複数の機能と、

光ディスクに書き込めるようにするウィザードを使用する 1 つまたは複数の機能と、

電子メールによってイメージを送信するのを容易にするもう 1 つのウィザードを使用する 1 つまたは複数の機能と

を含むことを特徴とする請求項 32 に記載の方法。

【請求項 34】

前記ユーザインターフェースのコントロールを使用する前記機能は、

ストレージユーザ経験をカプセル化する 1 つまたは複数の機能と、

アプリケーション定義の方法でアイテムを表示できるようにする 1 つまたは複数の機能と、

アイテムをリストに追加できるようにする 1 つまたは複数の機能と、

アイテムのプレビューイメージを表示できるようにする 1 つまたは複数の機能と

を含むことを特徴とする請求項 29 に記載の方法。

【請求項 35】

前記ユーザインターフェースの機能性を拡張する前記機能は、

コンテキストメニューへの追加を可能にする 1 つまたは複数の機能と、

アプリケーション定義のサムネイルの識別を可能にする 1 つまたは複数の機能と、

1 つまたは複数のファイルまたはフォルダに関する情報を表示するときに計算を実行できるようにする 1 つまたは複数の機能と

を含むことを特徴とする請求項 29 に記載の方法。

【請求項 36】

ユーザインターフェースのコントロールを使用する 1 つまたは複数の第 1 機能に対する 1 つまたは複数の呼出しを受け取るステップと、

前記ユーザインターフェースのデスクトップの機能性を拡張する 1 つまたは複数の第 2 機能に対する 1 つまたは複数の呼出しを受け取るステップと

を備えたことを特徴とする方法。

【請求項 37】

前記ユーザインターフェースの機能性を拡張する 1 つまたは複数の第 3 機能に対する 1 つまたは複数の呼出しを受け取るステップ

をさらに備えたことを特徴とする請求項 36 に記載の方法。

【請求項 38】

前記ユーザインターフェースの機能性を拡張する前記機能は、

コンテキストメニューへの追加を可能にする 1 つまたは複数の機能と、

アプリケーション定義のサムネイルの識別を可能にする 1 つまたは複数の機能と、

10

20

30

40

50

１つまたは複数のファイルまたはフォルダに関する情報を表示するときに計算を実行できるようにする１つまたは複数の機能と

を含むことを特徴とする請求項３７に記載の方法。

【請求項３９】

前記ユーザインターフェースのダイアログおよびウィザードを使用する１つまたは複数の第３機能に対する１つまたは複数の呼出しを受け取るステップ

をさらに備えたことを特徴とする請求項３６に記載の方法。

【請求項４０】

前記ユーザインターフェースのダイアログおよびウィザードを使用する前記機能は、ファイルおよびフォルダを開き、保存できるようにするダイアログを使用する１つまたは複数の機能と、

光ディスクに書き込めるようにするウィザードを使用する１つまたは複数の機能と、電子メールによってイメージを送信するのを容易にするもう１つのウィザードを使用する１つまたは複数の機能と

を含むことを特徴とする請求項３９に記載の方法。

【請求項４１】

前記ユーザインターフェースのコントロールを使用する前記機能は、ストレージユーザ経験をカプセル化する１つまたは複数の機能と、アプリケーション定義の方法でアイテムを表示できるようにする１つまたは複数の機能と、

前記デスクトップのサイドバーにアイテムを追加できるようにする１つまたは複数の機能と、

リストにアイテムを追加できるようにする１つまたは複数の機能と、

アイテムのプレビューイメージを表示できるようにする１つまたは複数の機能と

を含むことを特徴とする請求項３６に記載の方法。

【請求項４２】

前記デスクトップの機能性を拡張する前記機能は、前記デスクトップにサイドバーを表示できるようにする１つまたは複数の機能と、前記デスクトップにアプリケーション定義の通知を表示できるようにする１つまたは複数の機能と

を含むことを特徴とする請求項３６に記載の方法。

【発明の詳細な説明】

【技術分野】

【０００１】

本発明は、ソフトウェアおよびそのようなソフトウェアの開発に関する。より詳細には、アプリケーションプログラムおよびコンピュータハードウェアによる、ソフトウェアプラットフォームの使用を容易にするプログラミングインターフェースに関する。

【背景技術】

【０００２】

非常に早くから、コンピュータソフトウェアは、「オペレーティングシステム」ソフトウェアまたは「アプリケーション」ソフトウェアとして分類されるようになった。おおまかにいって、アプリケーションは、数学の方程式を解くか、またはワードプロセッシングをサポートするなど、コンピュータユーザのために特定のタスクを実行することを意図されたソフトウェアである。オペレーティングシステムは、コンピュータハードウェアを管理し、制御するソフトウェアである。オペレーティングシステムの目標は、コンピュータリソースをアプリケーションプログラムが使用可能にすると同時に、ハードウェアを実際に制御するのに必要な複雑さを隠蔽することである。

【０００３】

オペレーティングシステムは、アプリケーションプログラムインターフェースまたはＡＰＩとして周知の機能または関数を介してリソースを使用可能にする。用語ＡＰＩは、こ

10

20

30

40

50

これらの機能のうちの単一の関数を指すのにも使用される。関数は、多くの場合、アプリケーションプログラマに提供するリソースまたはサービスに関してグループ化される。アプリケーションソフトウェアは、個々のAPI関数を呼び出すことによってリソースを要求する。API関数は、オペレーティングシステムによって提供されるメッセージおよび情報をアプリケーションソフトウェアに中継する手段としても働く。

【0004】

【特許文献1】米国特許出願第09/598,105号明細書

【特許文献2】米国特許出願第09/613,289号明細書

【特許文献3】米国特許出願第10/063,296号明細書

【特許文献4】米国特許出願第10/402,179号明細書

【特許文献5】米国特許出願第10/324,746号明細書

【非特許文献1】[online]、ECMA、URL:www.ecma.ch

【発明の開示】

【発明が解決しようとする課題】

【0005】

ハードウェアの変更のほかに、オペレーティングシステムソフトウェアの進化を駆り立てるもう1つの要因が、アプリケーションソフトウェア開発を単純化し、加速する要望である。アプリケーションソフトウェア開発は、困難な作業になる場合があり、時折、数百万行のコードによって高度化されたプログラムを作成するために数年の開発者の時間を必要とする。Microsoft Windows（登録商標）オペレーティングシステム
のさまざまなバージョンなどの広く使用されているオペレーティングシステムに関して、
アプリケーションソフトウェア開発者は、そのオペレーティングシステムを利用する数千
本の異なるアプリケーションを毎年記述している。そのように多くの異なるアプリケーシ
ョン開発者をサポートするために、一貫性のある（coherent）、利便性の高いオペレー
ティングシステムベースが必要である。

【0006】

多くの場合、オペレーティングシステムをより複雑にすることによって、アプリケー
ションソフトウェアの開発を単純にすることができる。すなわち、ある機能が、複数の異
なるアプリケーションプログラムに有用であり得る場合に、その機能をオペレーティングシ
ステムに含めるために一度記述することが、数十人のソフトウェア開発者が数十本の異
なるアプリケーションに含めるために数十回記述することを要求するよりよい場合がある。
このように、オペレーティングシステムが、複数のアプリケーションによって要求される
広範囲の共通の機能性をサポートする場合、アプリケーションソフトウェア開発のコスト
および時間においてかなりの節約を達成することができる。

【0007】

オペレーティングシステムとアプリケーションソフトウェアの間のどこに線が引かれる
かに関係なく、有用なオペレーティングシステムについて、オペレーティングシステムお
よびコンピュータハードウェアとアプリケーションソフトウェアとの間のAPIは、オペ
レーティングシステム自体の効率的な内部動作と同様に重要であることは明白である。

【0008】

過去数年間に、インターネットおよびネットワーク技術全般の万国での採用によって、
コンピュータソフトウェア開発者の情景が変化した。従来、ソフトウェア開発者は、独立
型デスクトップコンピュータまたはローカルエリアネットワーク（LAN）を介して、限
られた数の他のコンピュータに接続されたLANベースコンピュータの単一サイトソフト
ウェアアプリケーションに焦点を合わせていた。ソフトウェアが、シュリンクラップパッ
ケージでマーケティングされ、販売されたので、そのようなソフトウェアアプリケーシ
ョンを、通常は、「シュリンクラップ」製品と称する。これらのアプリケーションは、明確
に定義されたAPIを使用して、コンピュータの基礎になるオペレーティングシステムに
アクセスした。

【0009】

10

20

30

40

50

インターネットが進化し、広く受け入れられたので、産業界は、ワールドワイドウェブ（または単に「ウェブ」）のさまざまなサイトでのアプリケーションのホスティングの威力を理解し始めた。ネットワーク化された世界では、あらゆる場所のクライアントが、個別の位置でホスティングされるサーバベースのアプリケーションに要求をサブミットし、数分の1秒で応答を受信することができる。しかし、これらのウェブアプリケーションは、通常は、元々は独立型コンピューティングマシンまたはローカルにネットワーク化されたコンピュータのために開発されたものと同じのオペレーティングシステムプラットフォームを使用して開発された。残念ながら、いくつかの場合に、これらのアプリケーションは、分散コンピューティング管理様式に適切に移行されていない。基礎になるプラットフォームは、単純に、無制限の数の相互接続されたコンピュータをサポートするという観念を用いて構成されていなかった。

10

【0010】

インターネットによって導入された分散コンピューティング環境へのシフトに対処するために、Microsoft Corporationは、「.NET」Framework（「ドットネット」と読む）またはMicrosoft（登録商標）.NETと称するネットワークソフトウェアプラットフォームを開発した。Microsoft（登録商標）.NETは、人、情報、システム、および装置を接続するソフトウェアである。このプラットフォームを用いると、開発者が、インターネットを介して実行されるウェブサービス（Web service）を作成できるようになる。この動的なシフトは、Microsoftの.NET（商標）FrameworkのAPI関数のセットによって達成された。

20

【0011】

.NET（商標）Frameworkの使用がますます一般的になるにつれて、このプラットフォームの効率および/または性能を高める形が特定された。本発明人は、そのような高められた効率および/または性能を可能にするプログラミングインターフェース関数の独自のセットを開発した。

【課題を解決するための手段】

【0012】

コンピュータプラットフォームのプログラミングインターフェースを本明細書で説明する。

30

【0013】

ある態様によれば、プログラミングインターフェースに、再利用可能なユーザインターフェースコントロールに関連するサービスの第1群と、ユーザインターフェースダイアログおよびユーザインターフェースウィザードに関連するサービスの第2群と、ユーザインターフェース機能性の拡張に関連するサービスの第3群と、ユーザインターフェースのデスクトップの機能性の拡張に関連するサービスの第4群との1つまたは複数を含めることができる。

【0014】

同一の符号は、複数の図面を通じて同一の特徴を指すのに使用される。

【発明を実施するための最良の形態】

40

【0015】

この開示は、開発者がウェブアプリケーションおよびウェブサービスを構築できるネットワークプラットフォームのアプリケーションプログラムインターフェース（API）などのプログラミングインターフェースを扱う。具体的に言うと、Microsoft Corporationによって作成された.NET（商標）Frameworkなどのネットワークプラットフォームを利用するオペレーティングシステムの例示的APIを説明する。.NET（商標）Frameworkは、分散コンピューティング環境で実施されるウェブサービスおよびウェブアプリケーションのソフトウェアプラットフォームである。.NET Frameworkは、特定のタスクを実行するために協力している疎結合ウェブサービスの間の通信にオープン通信標準規格（open communication standards）を

50

使用する、次世代のインターネットコンピューティングを表す。

【0016】

説明する実施形態で、ネットワークプラットフォームは、XML (extensible markup language) すなわちデータを記述するオープン標準規格を使用する。XMLは、World Wide Web Consortium (W3C) によって管理される。XMLは、ウェブページおよび企業間文書のデータ要素を定義するのに使用される。XMLでは、HTMLに類似するタグ構造が使用されるが、HTMLは、要素をどのように表示するかを定義し、XMLは、それらの要素に何が含まれるかを定義する。HTMLでは、予め定義されたタグを使用するが、XMLでは、ページの開発者がタグを定義することができる。したがって、事実上任意のデータアイテムを識別することができ、ウェブページが、データベースレコードのように機能することができる。XMLおよびシンプルオブジェクトアクセスプロトコル (SOAP) などの他のオープンプロトコルの使用を介して、ネットワークプラットフォームが、ユーザの必要に合わせて調整できる広範囲のサービスの統合を可能にする。本明細書に記載の実施形態を、XMLおよび他のオープン標準規格と共に説明するが、これは、請求される発明の動作に必要なではない。他の同等に実行可能な技術が、本明細書に記載の発明を実施するのに十分である。

10

【0017】

本明細書で使用される句、アプリケーションプログラムインターフェースまたはAPIには、メソッド呼出しまたは関数呼出しを使用する従来のインターフェースならびに、リモート呼出し (たとえばプロキシ、スタブ関係) およびSOAP/XML呼出しが含まれる。

20

【0018】

(例示的ネットワーク環境)

図1に、.NET (商標) Frameworkなどのネットワークプラットフォームを実装できるネットワーク環境100を示す。ネットワーク環境100には、ネットワーク104 (たとえばインターネット) 上でアクセスできるサービスを提供する、代表的なウェブサーバ102 (1)、...、102 (N) が含まれる。一般に符号102として参照されるウェブサービスは、リモートプロシージャコール (RPC) またはオブジェクトブローカ型技術などの、ネットワークを介してウェブサービスと相互作用する他の手段も使用することができるが、通常は、XML、SOAP、WAP (wireless application protocol)、HTTP (ハイパーテキスト転送プロトコル)、およびSMTP (シンプルメール転送プロトコル) などの産業界標準ウェブプロトコルを介してネットワーク104上でプログラマ的に相互作用する、再利用可能なプログラマブルアプリケーションコンポーネントである。ウェブサービスは、自己記述的とすることができ、しばしば、メッセージのフォーマットおよび順序に関して定義される。

30

【0019】

ウェブサービス102は、他のサービス (通信リンク106によって表されるように)、またはウェブアプリケーション110などのソフトウェアアプリケーション (通信リンク112および114によって表されるように) によって直接にアクセス可能である。各ウェブサービス102は、特定のサービスに関する要求を処理するソフトウェアを実行する1つまたは複数のサーバを含むものとして示されている。そのようなサービスは、しばしば、要求元にサービスされる情報を保管するデータベースを維持する。ウェブサービスを構成して、さまざまな異なるサービスの任意の1つを実行することができる。ウェブサービスの例に、ログイン検証、通知、データベース保管、株式市況、ディレクトリの突き止め、地図作成、音楽、電子ウォレット、カレンダー/スケジューラ、電話リスティング、ニュースおよび情報、ゲーム、発券業務などが含まれる。ウェブサービスを、互いにおよび他のアプリケーションと組み合わせて、インテリジェント対話経験を構築することができる。

40

【0020】

ネットワーク環境100には、ウェブサービス102 (通信リンク122によって表さ

50

れる)および/またはウェブアプリケーション110(通信リンク124、126、および128によって表される)を使用する代表的なクライアント装置120(1)、120(2)、120(3)、120(4)、...、120(M)も含まれる。クライアントは、クライアント120(3)と120(4)の間の例示的なXMLリンク129によって表されるように、標準プロトコルを使用して互いに通信することもできる。

【0021】

全般的に符号120として参照されるクライアント装置は、多数の異なる形で実施することができる。可能なクライアント実施形態の例に、制限なしに、ポータブルコンピュータ、ステーションリコンピュータ、タブレットPC、テレビジョン/セットトップボックス、無線通信装置、携帯情報端末、ゲーム機、プリンタ、コピー機、および他のスマート装置が含まれる。

10

【0022】

ウェブアプリケーション110は、ネットワークプラットフォームで動作するように設計されたアプリケーションであり、クライアント120からの要求を処理し、サービスするときに、ウェブサービス102を使用することができる。ウェブアプリケーション110は、プログラミングフレームワーク132上で稼動する1つまたは複数のソフトウェアアプリケーション130からなり、ソフトウェアアプリケーション130は、1つまたは複数のサーバ134または他のコンピュータシステムで実行されている。ウェブアプリケーション110の一部が、実際には1つまたは複数のクライアント120に常駐する場合があることに留意されたい。その代わりに、ウェブアプリケーション110が、実際にタスクを達成するクライアント120上の他のソフトウェアを調整することができる。

20

【0023】

プログラミングフレームワーク132は、アプリケーション開発者によって開発されるアプリケーションおよびサービスをサポートする構造である。プログラミングフレームワーク132は、複数の言語をサポートすることによって、マルチ言語開発およびシームレス統合を可能にする。プログラミングフレームワーク132は、SOAPなどのオープンプロトコルをサポートし、基礎になるオペレーティングシステムサービスおよびオブジェクトモデルサービスをカプセル化する。このフレームワークは、複数のプログラミング言語の堅牢で保護された実行環境を提供し、保護され、統合されたクラスライブラリを提供する。

30

【0024】

フレームワーク132は、アプリケーションプログラムインターフェース(API)レイヤ142、共通言語ランタイム(CLR)レイヤ144、およびオペレーティングシステム/サービスレイヤ146を含むマルチティアアーキテクチャである。この階層アーキテクチャを用いると、フレームワークの他の部分に影響せずに、さまざまなレイヤに対する更新および修正が可能になる。共通言語仕様(CLS)140を用いると、さまざまな言語の設計者が、基礎になるライブラリ機能性にアクセスできるコードを記述できるようになる。共通言語仕様140は、言語設計者とライブラリ設計者の間の、言語インターオペラビリティを促進するのに使用できるコントラクトとして機能する。CLSを厳守することによって、ある言語で記述されたライブラリは、別の言語で記述されたコードモジュールから直接にアクセスでき、ある言語で記述されたコードモジュールと別の言語で記述されたコードモジュールの間のシームレスな統合を達成することができる。CLSの1つの例示的な詳細な実施形態が、ECMA TC39/TG3の参加者によって作成されたECMA標準規格である。非特許文献1を参照されたい。

40

【0025】

APIレイヤ142は、アプリケーション130が、レイヤ146によって提供されるリソースおよびサービスにアクセスするために呼び出す関数のグループを提供する。ネットワークプラットフォームに対するAPI関数を公開することによって、アプリケーション開発者は、ネットワークリソースが実際にどのように動作するかまたは使用可能にされるかの複雑な相互作用を理解する必要なしに、ネットワークリソースおよび他のウェブサ

50

ービスを完全に利用する分散コンピューティングシステム用のウェブアプリケーションを作成することができる。さらに、ウェブアプリケーションを、任意の数のプログラミング言語で記述でき、共通言語ランタイムレイヤ 1 4 4 によってサポートされ、共通言語仕様 1 4 0 の一部として含まれる中間言語に変換することができる。この形で、A P I レイヤ 1 4 2 が、広範囲のさまざまなアプリケーションにメソッドを提供することができる。

【 0 0 2 6 】

さらに、フレームワーク 1 3 2 を、フレームワークをホスティングするサーバ 1 3 4 からリモートに実行されるリモートアプリケーションによって設定される、A P I 呼出しをサポートするように構成することができる。それぞれクライアント 1 2 0 (3) および 1 2 0 (M) に常駐する代表的なアプリケーション 1 4 8 (1) および 1 4 8 (2) は、直接的にまたは間接的に、ネットワーク 1 0 4 上で A P I レイヤ 1 4 2 への呼出しを行うことによって、A P I 関数を使用することができる。

10

【 0 0 2 7 】

このフレームワークを、クライアントで実施することもできる。クライアント 1 2 0 (3) は、フレームワーク 1 5 0 がクライアントで実施される状況を表す。このフレームワークは、サーバベースのフレームワーク 1 3 2 と同一とするか、クライアント用に修正することができる。その代わりに、クライアントが、携帯電話、携帯情報端末、ハンドヘルドコンピュータ、または他の通信 / コンピューティング装置などの制限付きまたは専用機能の装置である場合に、クライアントベースフレームワークを圧縮することができる。

【 0 0 2 8 】

20

(開発者のプログラミングフレームワーク)

図 2 に、プログラミングフレームワーク 1 3 2 を詳細に示す。共通言語仕様 (C L S) レイヤ 1 4 0 は、さまざまな言語 1 3 0 (1)、1 3 0 (2)、1 3 0 (3)、1 3 0 (4)、...、1 3 0 (K) で記述されたアプリケーションをサポートする。そのようなアプリケーション言語に、V i s u a l B a s i c、C + +、C #、C O B O L、J s c r i p t、P e r l、E i f f e l、P y t h o n などが含まれる。共通言語仕様 1 4 0 は、それに従う場合にさまざまな言語が通信できるようなる特徴のまたは特徴に関するルールのサブセットを指定する。たとえば、いくつかの言語は、共通言語ランタイム 1 4 4 によってサポートされる可能性がある所与の型 (たとえば、「i n t *」型) をサポートしない。この場合に、共通言語仕様 1 4 0 に、その型が含まれない。その一方で、すべてまたはほとんどの言語によってサポートされる型 (たとえば、「i n t []」型) は、共通言語仕様 1 4 0 に含まれ、したがって、ライブラリ開発者は、それを使用することができ、言語がそれを処理することが保証される。この通信の能力は、ある言語で記述されたコードモジュールと別の言語で記述されたコードモジュールの間のシームレスな統合をもたらす。異なる言語が、特定のタスクに特によく適するので、言語の間のシームレスな統合によって、開発者が、特定のコードモジュールに対し特定の言語を選択でき、そのコードモジュールを異なる言語で記述されたモジュールと共に使用する能力が与えられる。共通言語ランタイム 1 4 4 を用いると、クロス言語継承と共にシームレスなマルチ言語開発が可能になり、複数プログラミング言語の堅牢で保護された実行環境が提供される。共通言語仕様 1 4 0 および共通言語ランタイム 1 4 4 に関するさらなる情報については、特許文献 1、表題「Method and System for Compiling Multiple Languages」および特許文献 2、表題「Unified Data Type System and Method」を参照されたい。

30

40

【 0 0 2 9 】

フレームワーク 1 3 2 によって、オペレーティングシステム 1 4 6 (1) (たとえば、W i n d o w s (登録商標) ブランドのオペレーティングシステム) およびオブジェクトモデルサービス 1 4 6 (2) (たとえば、コンポーネントオブジェクトモデル (C O M) または分散 C O M) がカプセル化される。オペレーティングシステム 1 4 6 (1) は、ファイル管理、通知、イベント処理、ユーザインターフェース (たとえば、ウィンドウイング (windowing)、メニュー、ダイアログなど)、セキュリティ、認証、検証、プロセスおよびスレッド、メモリ管理などの従来の機能を提供する。オブジェクトモデルサービス

50

1 4 6 (2) は、さまざまなタスクを実行するために、他のオブジェクトとのインターフェースを提供する。A P I レイヤ 1 4 2 に対して行われる呼出しは、オペレーティングシステム 1 4 6 (1) および / またはオブジェクトモデルサービス 1 4 6 (2) によるローカル実行のために共通言語ランタイムレイヤ 1 4 4 に渡される。

【 0 0 3 0 】

A P I 1 4 2 は、A P I 関数を複数の名前空間にグループ化する。名前空間によって、本質的に、セットとして「型」と呼ばれる、関連する機能性の特定のセットを提供する、クラス、インターフェース、委任 (delegate)、列挙、および構造体のセットが定義される。クラスは、参照割り当てセマンティックス (semantics) を有する、管理されるヒープから割り当てられるデータを表す。委任は、オブジェクト指向関数ポインタである。列挙は、名前付き定数を表す特殊な種類の値型である。構造体は、値割り当てセマンティックスを有する、静的に割り当てられたデータを表す。インターフェースは、他の型を実施できるコントラクトを定義する。

10

【 0 0 3 1 】

名前空間を使用することによって、設計者が、型のセットを階層名前空間に編成することができる。設計者は、型のセットから複数のグループを作成することができ、各グループに、論理的に関連する機能性を公開する少なくとも 1 つの型が含まれる。例示的な実施形態では、A P I 1 4 2 が、3 つのルート名前空間を含むように編成される。図 2 には 3 つのルート名前空間だけが示されているが、追加のルート名前空間を A P I 1 4 2 に含めることもできることに留意されたい。A P I 1 4 2 に示された 3 つのルート名前空間は、プレゼンテーションサブシステムの第 1 名前空間 2 0 0 (ユーザインターフェースシェルの名前空間 2 0 2 を含む)、ウェブサービスの第 2 名前空間 2 0 4、およびファイルシステムの第 3 名前空間 2 0 6 である。各グループに名前を割り当てることができる。たとえば、プレゼンテーションサブシステム名前空間 2 0 0 に、名前「 W i n d o w s 」を割り当てることができることができ、ファイルシステム名前空間 2 0 6 の型に、名前「 S t o r a g e 」を割り当てることができる。名前付きグループを、S y s t e m 名前空間全体などのシステムレベル A P I の単一の「グローバルルート」名前空間の下で編成することができる。トップレベル識別子を選択し、プレフィックスすることによって、各グループの型を、型を含むグループの名前の前に置かれた選択されたトップレベル識別子を含む階層名によって簡単に参照することができる。たとえば、ファイルシステム名前空間 2 0 6 の型を、階層名「 S y s t e m . S t o r a g e 」を使用して参照することができる。このようにして、個々の名前空間 2 0 0、2 0 4、および 2 0 6 が、S y s t e m 名前空間からの主要な分岐になり、および個々の名前空間が「 S y s t e m . 」プレフィックスなどの指示子 (designator) を前に付けられる指定を伝える。

20

30

【 0 0 3 2 】

プレゼンテーションサブシステム名前空間 2 0 0 は、プログラミングおよびコンテンツ開発に関する。この名前空間は、アプリケーション、文書、媒体プレゼンテーション、および他のコンテンツの生成を可能にする型を供給する。たとえば、プレゼンテーションサブシステム名前空間 2 0 0 は、開発者がオペレーティングシステム 1 4 6 (1) および / またはオブジェクトモデルサービス 1 4 6 (2) からサービスを得られるようにするプログラミングモデルを提供する。

40

【 0 0 3 3 】

シェル名前空間 2 0 2 は、ユーザインターフェース機能性に関する。この名前空間は、開発者が自分のアプリケーションにユーザインターフェース機能性を埋め込めるようにし、さらに、開発者がユーザインターフェース機能性を拡張できるようにする型を供給する。

【 0 0 3 4 】

ウェブサービス名前空間 2 0 4 は、たとえば、イントラネット上の 2 つのピアの間で動作するチャットアプリケーションのように単純なアプリケーションおよび / または数百万のユーザに関するスケーラブルウェブサービスなどの複雑なアプリケーションなど、さま

50

ざまなウェブアプリケーションの作成を可能にするインフラストラクチャに関する。本明細書で説明するインフラストラクチャは、有利なことに、特定のソリューションの複雑さに適当なパーツだけを使用する必要があるという点で、非常に可変である。このインフラストラクチャは、さまざまなスケールおよび複雑さのメッセージベースアプリケーションを構築する基礎を提供する。このインフラストラクチャまたはフレームワークは、基本的メッセージング、セキュアメッセージング、信頼されるメッセージング、およびトランザクション処理されるメッセージングのAPIを提供する。下で説明する実施形態では、関連するAPIが、有用性、使い易さ、拡張性、およびバージョンナビリティ (versionability) のバランスをとるように注意深く作成された形で名前空間の階層に分解される。

【0035】

10

ファイルシステム名前空間206は、ストレージに関する。この名前空間は、情報の保管および検索を可能にする型を供給する。

【0036】

フレームワーク132のほかに、プログラミングツール220が、開発者がウェブサービスおよび/またはアプリケーションを構築するのを支援するために設けられている。プログラミングツール220の1つの例が、Microsoft社によって提供されるプログラミングツールのマルチ言語スイートであるVisual Studio (商標) である。

【0037】

(ルートAPI名前空間)

20

図3に、プレゼンテーションサブシステム名前空間200を詳細に示す。一実施形態において、名前空間が、名前のストリングがピリオドによって連結される階層命名規約に従って識別される。たとえば、プレゼンテーションサブシステム名前空間200は、ルート名「System.Windows」によって識別される。「System.Windows」名前空間内に、「System.Windows.Explorer」として識別されるもう1つの名前空間があり、さらに、「System.Windows.Explorer.Controls」と称する、コントロールに関するもう1つの名前空間が識別される。この命名規約を念頭において、以下では、プレゼンテーションサブシステム名前空間200の全般的な概要を提供するが、他の命名規約を使用して同等の効果を得ることができる。

30

【0038】

Shell名前空間202 (「System.Windows.Explorer」) に、ストレージシステム内のアイテム、FTPロケーションまたはDAV (Distributed authoring and Versioning) ロケーションのファイル、コントロールパネル内のコントロールパネルアプレット、アプリケーションなど、エンドポイントのセットをエンドユーザが調査でき、ナビゲートできるようにするユーザインターフェース機能性をサポートするクラスおよびAPIが含まれる。このナビゲーションに、1つまたは複数のフォルダを開くこと、ならびにフォルダ内のフォルダを開くことを含めることができる。Shell名前空間202を用いると、開発者が、そのような機能性を自分のアプリケーションに組み込めるようになり、さらに、開発者が、この調査およびナビゲーションの機能性を拡張できるようにする。Windows (登録商標) オペレーティングシステムの諸バージョンで、このユーザインターフェースを、通常は「エクスプローラ」と称する。

40

【0039】

ある実施形態で、Shell名前空間202のユーザインターフェース機能性を用いてユーザがそれを調査でき、ナビゲートできるようになるストレージシステムは、すべての種類の情報の編成、検索、および共有のためのアクティブストレージプラットフォームである。このプラットフォームは、リッチデータモデルを定義し、リレーショナルストレージエンジンに基づき、柔軟なプログラミングモデルをサポートし、データの監視、管理、および操作のためのデータサービスのセットを提供する。データは、ファイルベースデータまたは非ファイルベースデータとすることができ、データを、通常は「アイテム」と称

50

する。このファイルシステムでは、通常、ファイルシステムによって提供される機能性が拡張される。というのは、このファイルシステムが、個人連絡先、イベントカレンダー、電子メールメッセージなどの非ファイルデータであるアイテムも扱うからである。そのようなファイルシステムの一例が、「WinFS」ファイルシステムである。

【0040】

Shell 名前空間 202 のユーザインターフェース機能性を用いてユーザがそれを調査でき、ナビゲートできるようになるストレージシステムのユニバーサルデータストアは、そのストアに常駐するデータの編成、検索、共有、同期化、およびセキュリティをサポートするデータモデルを実施する。このデータストアのストレージ情報の基礎的な単位を、アイテムと称する。データモデルは、アイテムおよびアイテム拡張を宣言し、アイテムの間の関係を確立し、アイテムをフォルダおよびカテゴリに編成する機構を提供する。

10

【0041】

アイテムは、単純なファイルと異なって、ストレージシステムによってユーザまたはアプリケーションプログラムに公開されるすべてのオブジェクトにまたがって一般的にサポートされるプロパティの基本セットを有するオブジェクトである、保管可能な情報の単位である。アイテムは、新しいプロパティおよび関係を導入できるようにする特徴を含む、すべてのアイテムタイプにまたがって一般的にサポートされるプロパティおよび関係も有する。このプロパティおよび関係のデータを、アイテムに関連するメタデータとも称する。下で詳細に述べるように、メタデータは、アイテム装飾スキーマに従って保管することができる。このアイテム装飾スキーマによって、アイテムをユーザに提示するのに適当な形を示すことができる。

20

【0042】

アイテムは、コピー、削除、移動、開く、印刷、バックアップ、レストア、複製などの共通の動作の対象である。アイテムは、保管でき検索できる単位であり、ストレージプラットフォームによって操作される保管可能な情報のすべての形が、アイテム、アイテムのプロパティ、またはアイテムの間の関係として存在し、これらのそれぞれを、下で詳細に説明する。アイテムは、(すべてのさまざまな種類の) Contacts (連絡先)、People (人)、Services (サービス)、Locations (ロケーション)、Documents (文書) などのデータの、実世界の簡単に理解できる単位である。

30

【0043】

アイテムは、独立のオブジェクトである。したがって、アイテムを削除する場合に、アイテムのプロパティのすべても削除される。同様に、アイテムを検索するときに、受け取られるものは、アイテムと、そのアイテムのメタデータに含まれるそのプロパティのすべてである。ある実施形態では、特定のアイテムを検索するときに、プロパティのサブセットを要求できるようにすることができるが、そのような実施形態の多くに関するデフォルトは、検索されるときに、アイテムを、直接のおよび継承されたプロパティのすべてと共に提供することである。さらに、アイテムの型の既存のプロパティに新しいプロパティを追加することによって、アイテムのプロパティを拡張することもできる。これらの「拡張」は、その後、そのアイテムの真正のプロパティになり、そのアイテム型のサブタイプに、自動的に拡張プロパティを含めることができる。拡張を、ファイルに関連するメタデータとも称する場合がある。

40

【0044】

アイテムのグループを、アイテムフォルダ (item Folder) と称する特殊なアイテム (ファイルフォルダと混同してはならない) に編成することができる。しかし、ほとんどのファイルシステムと異なって、アイテムは、複数のアイテムフォルダに属することができる、アイテムが、あるアイテムフォルダでアクセスされ、改訂されるときに、この改訂されたアイテムに、別のアイテムフォルダから直接にアクセスすることができる。要するに、アイテムへのアクセスを、異なるアイテムフォルダから行うことができるが、実際にアクセスされるものは、実際にはまったく同一のアイテムである。しかし、アイテ

50

ムフォルダは、必ずしもそのメンバアイテムのすべてを所有せず、あるいは他のアイテムフォルダと共にアイテムを単純に協同所有することができ、アイテムフォルダの削除が、必ずしもアイテムの削除をもたらさない。

【0045】

アイテムのグループを、カテゴリ (Category) に編成することもできる。アイテムフォルダが、相互関係 (すなわち、共通の記述された特性) を有しないアイテムを含むことができるが、カテゴリの各アイテムは、そのカテゴリについて記述された共通の型、プロパティ、または値 (「共通性」) を有し、カテゴリへの関係およびカテゴリ内の他のアイテムとの間の関係の基礎を形成するのがこの共通性であるという点で、カテゴリは、概念的にアイテムフォルダと異なる。さらに、特定のフォルダへのアイテムのメンバシップは、そのアイテムの特定の態様に基づく必須のものではないが、ある実施形態について、あるカテゴリに断定的に関連する共通性を有するすべてのアイテムを、ハードウェア/ソフトウェアインターフェースシステムレベルで、自動的にそのカテゴリのメンバにすることができる。概念的に、カテゴリは、メンバシップが特定の照会の結果に基づく (データベースのコンテキストで、など) 仮想アイテムフォルダと考えることもでき、この照会の条件 (カテゴリの共通性によって定義される) を満たすアイテムは、したがって、カテゴリのメンバシップを備える。

10

【0046】

ファイル、フォルダ、およびディレクトリと異なって、本発明のアイテム、アイテムフォルダ、およびカテゴリは、特徴として性質において「物理的」ではない。というのは、これらが、概念的に物理コンテナの同等物を有しておらず、したがって、アイテムが、複数のそのようなロケーションに存在することができるからである。アイテムが、複数のアイテムフォルダロケーションに存在し、カテゴリに編成される能力によって、当技術分野で現在使用可能なものを超える、ハードウェア/ソフトウェアインターフェースレベルでのデータ操作およびストレージ構造能力の機能強化および質の高さがもたらされる。

20

【0047】

アイテムに、複数のアイテムの間の関係を判定できるようにする関係情報も含めることができる。関係は、バイナリ関係であり、一方のアイテムがソースとして指定され、他方のアイテムがターゲットとして指定される。ソースアイテムおよびターゲットアイテムは、関係によって関連する。

30

【0048】

ユニバーサルデータストア内のアイテムは、シェルブラウザによってユーザに提示され、このシェルブラウザは、ユーザがハードウェア/ソフトウェアインターフェースを用いて見ることができる、および対話することができるようにする (Shell 名前空間 202 のクラスおよび API を使用して) ユーザインターフェースを提供する。ユニバーサルデータストア内の各アイテムは、ユニバーサルデータスキーマに従って保管される。このスキーマには、型関連付け (type association) と称する、アイテムを記述する機構が含まれる。各型関連付けは、シェル内の基本的表現を有し、型関連付けに従ってアイテムを保管することによって、シェルは、少なくとも基本表示ビューまたはデフォルト表示ビューに従ってアイテムを表示することができる。

40

【0049】

型関連付けは、アイテムに関連するプロパティであり、データをユニバーサルデータストアに置くときに、そのデータに関連する 1 つまたは複数のプロパティを宣言して、そのアイテムの型を決定しなければならない。このプロパティは、データに関連するメタデータとして含めることができる。シェルは、デフォルト型関連付けのセットを有し、これによって、アイテムについて宣言しなければならない最も基本的な最小限のプロパティが表示される。

【0050】

プロパティ宣言を越えて、アイテムに関連するメタデータに、シェルがアイテムのプレゼンテーションをどのように装飾するかを示すデータを含めることができる。装飾は、こ

50

の場合に、アイテムをどのようにユーザに提示するかに関する「ヒント」と考えることができる。このメタデータは、アイテム装飾スキーマに従って保管することができる。アイテム装飾スキーマでは、シェルがアイテムを提示するのに使用できるアイテム装飾ビューが定義される。たとえば、アイテム装飾データによって、アイテムの最も重要な宣言されたプロパティを記述することができる。これらの「高価値」プロパティは、シェルでの提示に関して最も望ましいものとすることができる。

【0051】

アイテムを提示するために、アイテム装飾データによって、アイテムのプレゼンテーションに適切なビューフィールドのセットを示すことができる。ビューフィールドは、宣言されたプロパティの射影であり、共通のビューフィールドに、「タイトル」、「作成者」、「作成日」、または「最終編集日」を含めることができる。シェルには、標準的なビューフィールドのセットが含まれ、独立ソフトウェアベンダ（ISV）は、自分のデータのプレゼンテーションに適切なビューフィールドを定義することができる。新しいアイテム型を開発するときに、ISVは、ISVが定義するアイテムプロパティをシェルのビューフィールドにマッピングするか、ISV自身のビューフィールドを提供することができる。

10

【0052】

たとえば、特定のアイテムデータに、ソングデータを含めることができる。宣言されるプロパティのセットに、歌の題名、アーティスト、レコーディング日付、アルバム、曲の長さ、およびそのようなソングアイテムに適切な他の宣言を含めることができる。アイテム装飾データによって、シェルでアイテムを提示するときに、ビューフィールド「Title」、「Artist」、および「Album」をユーザに表示しなければならないことを指示することができる。

20

【0053】

アイテム装飾は、シェルによってサポートされる表示の任意の態様にするすることができる。ある共通の他のアイテム装飾は、たとえば、データフォーマット、デフォルトソート順序、およびデフォルトアイコンサイズである。さらに、アイテム装飾データによって、所与のアイテムを表示する際に使用されるコモンコントロールを記述することができる。たとえば、Rating フィールドで、一連の星印としてレーティングを表示するレーティングコントロールを使用することができる。アイテム装飾データによって、アイテムと共に使用するのに適するタスクまたはバンプ（verb）を記述することができる。「タスク」および「バンプ」という用語は、アイテムに関して行われるあるアクションを記述するものであり、この用語は、交換可能に使用することができる。たとえば、「編集」または「プレビュー」を、アイテムに関連する適切なタスク/バンプとすることができる。シェルをさらに構成して、アイテムに関するアクションを実行するために、ユーザ選択時にこれらのタスクをサポートするアプリケーションを起動することができる。

30

【0054】

アイテム装飾ビューは、所与のアイテムまたはアイテムの同種のセット（類似するアイテム装飾ビューを有するアイテムからなる）を完全に提示するのに十分である。異なるアイテム装飾スキーマを有するアイテムを表示するために、シェルは、シェル装飾ビューに従ってアイテムを提示するシェルビュースキーマを備える。シェルビュースキーマによって、シェルまたはISVが、異種データの所与のセットに適切なビューを宣言できるようになる。

40

【0055】

シェル装飾ビュー内での表現について選択されるアイテムに、共通の特性を含めることができる。さまざまな共通の特性を、シェル装飾ビューについて受入可能とすることができる。たとえば、シェルビュースキーマで、すべての既知のピクチャ型（たとえば、.GIF、.JPEG、.BMP、.TIFFなど）に対する共通のおよび適当な、フィールドおよびメタデータを表示するのに使用される「Picture」ビューを定義することができる。シェルビュースキーマは、所与のアイテム装飾ビューの衝突するディスプレイ

50

属性をオーバーライドし、シェルビュースキーマに従って各ピクチャアイテムを提示する。もう1つの例として、シェルは、ワードプロセッシング文書、スプレッドシート、またはデータベースなどの通常のプロダクティビティアプリケーションによって作られるアイテムの適当なカラムおよびメタデータに関して最適化された「Document」シェルビューを提供することができるが、これらのアイテムのそれぞれのアイテム装飾は互いに大きく異なる可能性がある。そのようなビューは、これらの文書のそれぞれの間で共通のプロパティを提供することによって価値を有する。後者の文書型がインストールされるときに、シェルビューは、ビューが最初に作成されたときに新しい型が考慮されていない可能性がある場合でも、一貫性のあるシェルビューに従って新しいアイテムを提示することができる。

10

【0056】

シェルビュースキーマは、多岐にわたるディスプレイ属性を提供することができ、ISVは、そのようなシェルビューを提供することを望む場合がある。ディスプレイ属性に、制限なしに、プレビューペインのサイズ、プレビューペイン内に表示されるメタデータ、使用されるカスタムコントロール、ならびに提示されるアイテムに適当なタスクおよびバースを含めることができる。

【0057】

シェルは、エクスプローラディスプレイビューに従ってアイテムを提示することができる。「エクスプローラ」を、ストレージアプリケーションと呼ぶことができ、シェルまたはISVによって提供されることができる。エクスプローラは、たとえば、ユーザが、データストア内の選択されたアイテムの表示、照会、ナビゲーション、タスクへの起動、または編成を行えるようにすることができる。「エクスプローラ」という用語は、表示されるアイテムが存在するロケーションを暗示するのではなく、「アクティビティセンタ」、「ビューワ」、および「ライブラリ」などの用語を、ストレージアプリケーションを表現するために「エクスプローラ」と交換可能に使用することができる。

20

【0058】

例のエクスプローラスキーマ階層に、アイテムビュースキーマであるボトムレイヤが含まれる。アイテムビュースキーマは、アイテムを表すのに必要な基本ディスプレイを提供し、エクスプローラビュースキーマは、そのディスプレイ要素を据え置くか、必要なときにそれに頼ることができる。

30

【0059】

エクスプローラビュースキーマには、シェルビュースキーマおよびエクスプローラ装飾も含まれる。エクスプローラ装飾は、エクスプローラ全体を装飾し、特有の色およびブランド設定 (branding) 要素などのディスプレイ要素を提供する。これらのエクスプローラ装飾は、エクスプローラが提供するさまざまなビューの間で存続する。多岐にわたるさまざまなディスプレイ属性を、エクスプローラ装飾に適用できる。たとえば、エクスプローラに関連するデータ照会またはタスク/バースを、エクスプローラを用いる表示に適用できる。表示されるタスクは、多くの場合、そのタスクを実行できるアプリケーションと結合される。

【0060】

エクスプローラビュースキーマに、任意選択として、1つのシェルビュースキーマまたは複数のシェルビュースキーマを含めることができる。シェルビュースキーマは、エクスプローラアイテムのサブセットに対するシェルビューを提供するように構成することができる。たとえば、エクスプローラを、ユーザにソングアイテムを表示するように構成することができる。第1のシェルビュースキーマを含めて、アルバムの表示を提供することができる。第2のシェルビュースキーマを含めて、ソングトラックの表示を提供することができる。この方法で、両方のタイプのアイテムは、エクスプローラ内の適当なビューを有する。上述のように、シェルビューの利用は、任意選択として共通の特性を共有することができるアイテムのセットの提示に関連する。

40

【0061】

50

エクスプローラは、シェルに含まれるシェルビューにも頼ることができる。エクスプローラ内での提示のために選択されたアイテムが、エクスプローラに含まれるシェルビューのどれによってもサポートされない場合に、シェルが、エクスプローラ内での使用のために適当なシェルビューを提供することができる。同様に、上述のように、エクスプローラは、シェルによって提供されるアイテムディスプレイビューまたはデフォルトディスプレイビューに頼ることもできる。この機能性によって、シェルによって表示できる任意のアイテムを、エクスプローラ内でも表示できることが保証される。エクスプローラは、これらのシェルが提供するディスプレイスキーマを据え置くように構成することができ、あるいは、たとえば予期せぬデータの表示を提供するために、それに頼ることができる。

【0062】

10

Shell 名前空間 202 は、アイテムのための基礎になるデータ（たとえば、ファイル内容、画像データ、連絡先情報データなど）が保管されるロケーションにかかわらず、一貫性のある方法でアイテムを操作でき、対話できるようにするユーザインターフェース機能性をサポートする。アプリケーション設計者の観点から、Shell 名前空間 202 によってサポートされる機能性は、よりよい開発体験をもたらすが、これは、少なくとも部分的に、異なるロケーションに保管される基礎になるデータにかかわらず、そのようなアイテムを一貫性のある方法で扱うことができるからである。Shell 名前空間 202 を使用するとき、そのような異なるデータストレージロケーションの潜在的な存在にかかわらず、コヒーレントな経験を設計者に提供する。

【0063】

20

Shell 名前空間 202 では、Controls 名前空間 302、Dialogs 名前空間 304、Addin 名前空間 306、および Desktop 名前空間 308 を含む追加の名前空間が定義される。Shell 名前空間内のこれらの追加名前空間 302、304、306、および 308 のそれぞれは、コントロール、ダイアログ、および / またはハンドラなど、以下で詳細に述べるさまざまな機能性を提供する 1 つまたは複数のサービスまたはコンポーネントが含まれる。

【0064】

Shell 名前空間 202 は、追加名前空間 302、304、306、および 308 のそれぞれによって使用できるオブジェクトモデルを定義する。このオブジェクトモデルには、下記のオブジェクトが含まれる。

30

- ・ファイルフォルダ、スタック、個々のファイル、個々の WinFS アイテム（たとえば、アルバム、歌、写真など）など、Shell を介してユーザに提示できるアイテムを記述するのに使用される ExplorerItem オブジェクト（Item オブジェクトとも称する）。

- ・ ExplorerItems に対する照会を記述するのに使用される Libraries オブジェクト。

- ・ユーザへの表示のためにアイテム（たとえば ExplorerItem オブジェクトによって表される）からデータを射影するのに使用される ViewFields オブジェクト（Properties オブジェクトとも称する）。ストレージシステムのプロパティと ViewFields オブジェクトの間で定義される射影は、ExplorerItems に基づく追加計算または処理を実行することができる。たとえば、記憶装置を表すフォルダが表示されており、設計者がそのフォルダにその記憶装置の空き容量を含めることを望む場合に、射影によって記憶装置の総容量とその記憶装置に保管されたファイルのサイズ（ExplorerItems に基づいて表される）に基づいて、空き容量が計算される。

40

- ・ Library オブジェクトの永続的な形である、ダイナミックリストへのリンクを記述するのに使用される StorageFavorites オブジェクト。StorageFavorites オブジェクトは、照会と、照会の結果をユーザに提示するビューの組合せである。

【0065】

50

このオブジェクトモデルは、2003年10月23日出願の米国特許出願第____号、弁理士整理番号MF C P . 1 0 9 8 3 4 (M S # 3 0 6 9 2 3 . 0 1)、表題「System and method for presenting items to a user with a contextual presentation」に詳細に記載されている。

【0066】

Controls名前空間302(「System.Windows.Explorer.Controls」)では、アプリケーションによって使用できる、再利用可能なユーザインターフェースコントロールのコレクションが定義される。複数の異なるアプリケーションが、これらのコントロールを利用することができるので、これらのコントロールは、再利用可能とみなすことができる。この再利用可能なコントロールのコレクションによって、アプリケーションの設計を単純化することができる。というのは、アプリケーション設計者が、コントロールの彼ら自身のバージョンを設計することを要求されるのではなく、これらのコントロールに頼ることができるからである。さらに、この再利用可能なコントロールのコレクションを使用することによって、複数のアプリケーションにまたがる一貫性のあるルックアンドフィールを達成することができる。

10

【0067】

Controls名前空間302によって定義されるコントロールは、ユーザインターフェース内のアイテムの操作および/または表示を含む、ユーザインターフェースのシェルの操作および/または表示を可能にする。上で述べたように、ユーザインターフェースのシェル内で表示できるこれらのアイテムは、多岐にわたる種類のデータのいずれをも表し、ファイルシステムに保管される単に伝統的な「ファイル」ではない。さらに、Controls名前空間302によって定義されるコントロールを用いると、アプリケーション設計者が、多岐にわたるロケーションからのデータを扱うときに同一のコントロールを使用できるようになり、一貫性がありコヒーレントな設計経験がもたらされる。

20

【0068】

Controls名前空間302では、下記の再利用可能なコントロールが定義される。

・ExplorerViewコントロールは、ストレージユーザ経験をカプセル化するのに使用される。ストレージユーザ経験は、関係、照会などを用いてアイテムにまたがって作業するために定義される。このストレージユーザ経験を用いると、エンドユーザが、たとえば、プロパティ値に基づく照会を発行し、共通プロパティ値を用いてWinFSアイテムをスタックし、スタックされていないアイテムからスタックへのドラッグアンドドロップを可能にする(たとえば、プロパティ述部のプロパティの値に、スタックに存在するものをセットすることによって)などが可能になる。ExplorerViewコントロールを使用して、Microsoft(登録商標)Windows(登録商標)オペレーティングシステムの以前のバージョン(たとえば、Windows(登録商標)オペレーティングシステムのWindows(登録商標)XPバージョンまたはWindows(登録商標)98バージョン)など、以前のオペレーティングシステム(レガシシステムとも称する)で見られるものと同一の外見を有するディスプレイ用のフォルダを記述することができる。

30

40

・ExplorerItemsの表示に使用されるExplorerItemView。ExplorerItemsのレイアウトおよび設計は、アプリケーション設計者が定義することができる。オペレーティングシステムは、アプリケーション設計者が定義できるレイアウトおよび設計にある制約を課すユーザ経験を定義することもでき、したがって、アプリケーション設計者によって定義されるレイアウトおよび設計が、オペレーティングシステムによって定義されるユーザ経験と一貫性を持つようになる。したがって、ExplorerItemViewコントロールを用いると、設計者が、ExplorerItemsからのデータをユーザに表示する方法(たとえば、列で、行で、または他の構成で)ならびにどのデータをユーザに表示するかを決定できるようになる。

・アイテムをサイドバーに追加できるようにするのに使用されるBasketCont

50

rol コントロール (サイドバーは、以下で詳細に説明する)。カーソル制御装置を使用してアイテムを選択し、それをサイドバーにドラッグし、そこでドロップ (たとえば、カーソル制御ボタンを離すことによって) することによって、「ドラッグアンドドロップ」でサイドバーにアイテムを追加することができる。Basket Control コントロールは、2003 年 10 月 13 日出願の米国特許出願第 _____ 号、弁理士整理番号 003797.00695 (MS# 304631.1)、表題「Extensible Creation And Editing Of Integrated Collections」に詳細に記載されている。

- ・アイテムをリストまたはグループに追加できるようにするのに使用される List Maker コントロール。カーソル制御装置を使用してアイテムを選択し、適当なリストまたはグループの上にドラッグし、そこでドロップ (たとえば、カーソル制御ボタンを離すことによって) することによって、「ドラッグアンドドロップ」の形でアイテムのリストまたはグループを作成することができる。たとえば、アイテムを歌とし、リストをプレイリストとすることができ、アイテムを画像とし、リストを写真アルバムとすることができ

10

- ・アイテムのプレビューイメージをユーザに提示できるようにするのに使用される Preview Image Control コンポーネント。アイテムが選択されるときに、プレビューイメージをユーザに提示することができ、このプレビューイメージは、通常、イメージのサムネイルより大きい、ユーザインターフェースのディスプレイ領域全体より小さい。

20

【0069】

Dialogs 名前空間 304 (「System.Windows.Explorer.Dialogs」) では、アプリケーションによって使用できるユーザインターフェースダイアログおよびウィザードのコレクションが定義される。Controls 名前空間 302 のコントロールに似て、Dialogs 名前空間 304 のダイアログおよびウィザードは、再利用することができ、その結果、複数の異なるアプリケーションが、これらのダイアログおよびウィザードを利用することができる。この再利用可能なダイアログおよびウィザードのコレクションは、複数のアプリケーションにまたがる一貫性のあるルックアンドフィールを維持するのを助ける。

【0070】

Dialogs 名前空間 304 によって定義されるダイアログおよびウィザードは、ユーザインターフェースのシェル内でアクションを実行できるようにする。上述のように、これらのシェルは、多岐にわたるさまざまな種類のデータのいずれをも表すことができるアイテムに基づき、ファイルシステムに保管される単に伝統的な「ファイル」ではない。さらに、Dialogs 名前空間 304 によって定義されるダイアログおよびウィザードを用いると、アプリケーション設計者が、さまざまなロケーションからのデータを扱うときに同一のダイアログおよびウィザードを使用できるようになり、一貫性のあるコヒーレントな設計経験がもたらされる。

30

【0071】

Dialogs 名前空間 304 では、下記のダイアログおよびウィザードが定義される。

40

- ・ファイルおよびフォルダを開き、保存できるようにするのに使用される Open / Save ダイアログ。これらのダイアログは、拡張可能でもあり、アプリケーション設計者が、ダイアログの機能性を拡張することができる。たとえば、これらのダイアログを拡張して、アプリケーション設計者が使用を望むファイルおよび / またはフォルダに関するさまざまな追加のメタデータを利用することができる。

- ・CD および / または DVD などの光ディスクに書き込めるようにするために使用される CD / DVD Burn ウィザード。ハードウェア設計者は、CD および / または DVD に書き込めるようにする彼ら自身のハードウェア固有のステップを含む彼ら自身のウィザードを提供することができる。

- ・電子メールによってイメージを送信する際にユーザを支援するのに使用される Send

50

d Photosウィザード。このウィザードを用いると、ユーザが、イメージを特定のサイズに変更できるようになる。このウィザードは、電子メール送信に適すると思われる特定のサイズにイメージのサイズを変更することができ、あるいは、その代わりに、ユーザが複数の異なるサイズから選択できるようにすることができる。

【0072】

Addin名前空間306(「System.Windows.Explorer.Addin」)では、ユーザインターフェース機能性を拡張する基本クラスおよびインターフェースのコレクションが定義される。Addin名前空間306を用いると、アプリケーション設計者が、ユーザインターフェースの機能性を拡張する管理されたコードを追加できるようになる。この拡張を用いると、たとえば、アプリケーション設計者が、適当と思われのようにユーザインターフェースの諸部分をカスタマイズできるようになる。

10

【0073】

Addin名前空間306を用いると、多岐にわたるさまざまな種類のデータのいずれをも表し、ファイルシステムに保管される単に伝統的な「ファイル」ではないアイテムを表示するのに使用されるユーザインターフェースのシェルの拡張が可能になる。さらに、Addin名前空間306を用いると、アプリケーション設計者が、コヒーレントな形でユーザインターフェースのシェルの拡張できるようになり、より管理され改善された設計経験がもたらされる。

【0074】

下記などのさまざまな機能性に関する管理されたコードが、Addin名前空間306によってサポートされる。

20

- ・コンテキストメニューハンドラ。コンテキストメニューには、たとえば表示されているアイテムを右クリックすることによって、アクセスすることができる。コンテキストメニューハンドラは、アプリケーション設計者が、コンテキストメニューに追加される彼ら自身のエントリを定義できるようにする。

- ・サムネイルエクストラクタ(extractor)。サムネイルエクストラクタを用いると、アプリケーション設計者が、彼ら自身のサムネイル(たとえば、ファイル、フォルダ、および/または表示される他のアイテムに関する)を開発でき、ユーザへの表示のためにオペレーティングシステムに対してこれらのサムネイルを識別できるようになる。

- ・WinFSからエクスプローラへの射影の合成/分解するためのハンドラ。これらのハンドラは、アプリケーション設計者が、ファイルおよび/またはフォルダに関する情報をユーザに表示するときにさまざまな計算を実行できるようにする(たとえば、記憶装置の空き容量を計算できるようにする)。

30

【0075】

Desktop名前空間308(「System.Windows.Explorer.Desktop」)は、アプリケーション設計者がユーザに表示されるデスクトップの機能性を拡張できるようにする機能性のコレクションを定義する。Desktop名前空間は、Shell名前空間202の一部とすることができ(たとえば、「System.Windows.Explorer.Desktop」)、その代わりに、プレゼンテーションサブシステム名前空間200の一部であるがShell名前空間202の一部でないものとして行うことができる(たとえば、「System.Windows.Desktop」)。Desktop名前空間には、サイドバーおよび通知にかかわる要素が含まれる。サイドバー要素および通知要素をDesktop名前空間に含めることによって、ユーザに、サイドバーおよび通知の集中化されたコントロールが与えられる。たとえば、すべてのアプリケーションからの通知に関するルールを、通知を使用するアプリケーションごとのルール定義にアクセスすることを介するユーザによるナビゲーションを必要とせずに、ユーザが1回でセットすることができる。

40

【0076】

一般に、サイドバーは、統合された対話型周辺認識ディスプレイ内のダイナミック通信アクセスおよび情報認識を提供し、この対話型周辺認識ディスプレイ内では、指定された

50

通信連絡先および情報要素が、ダイナミックに追跡または受信され、進行中の基礎でユーザに提供される。ある実施形態で、この機能が、従来のディスプレイ装置の1つの辺に沿った継続ディスプレイストリップ内の1つまたは複数の列に表示される少なくとも1つのカスタマイズ可能なダイナミックサムネイルを介して提供される。さらに、追加の実施形態で、サムネイルが、ディスプレイ全体を含むディスプレイの1つまたは複数の部分のいずれにも表示される。ディスプレイ全体がカバーされる実施形態は、たとえばハンドヘルドまたはパームトップコンピューティング装置、携帯電話、または他の限られたディスプレイ領域を有する電子装置など、比較的小さいディスプレイ領域を有する装置でサイドバーが使用される場合に特に有用である。

【0077】

10

カスタマイズ可能なダイナミックサムネイル各々は、たとえば、特定の通信連絡先（特定の個人、会社、組織、または他のエンティティなど）またはユーザが興味を持つ可能性がある情報の特定の要素を表す。そのような情報要素に、たとえば、共有ファイルまたは共有フォルダが修正されるとき、共有データベースまたは共有ワークスペースの情報が変更されるとき、電子メール状況、カレンダー、インターネットウェブページ、気象条件、面会予約、スケジュール、統計的情報、株式市況、交通情報、またはユーザが興味を持つ可能性がある他のインターネットアクセス可能情報またはネットワークアクセス可能情報が含まれる。

【0078】

20

前述のダイナミックサムネイルに、一般に、興味のある連絡先または情報を記述する「タイル」（「チケット」とも称する）と、どの通信連絡先または情報がタイルによって表されるかを表すのに特化した「ビューワ」の組合せが含まれる。各タイルおよびビューワは、たとえば、Shell名前空間202によって提供されるユーザインターフェース機能性によってアクセスできるストレージシステム内のアイテムとすることができる。1つまたは複数の「サービス」を使用して、各タイルによって記述される情報の現況および/または通信連絡先の状況と自動的に相互作用をし、これを追跡し、または受け取る。情報の現況および通信連絡先の状況が、アイテムをグラフィカルにかつ/またはテキストによって表示するために、対話型周辺認識インターフェース内に常駐する「コンテナ」内で各タイルをホスティングすることによって、動的に提供される。周辺認識インターフェースは、潜在的な注意散漫またはユーザへの妨害を減らす方法で情報および/または通信連絡先を表示する。

30

【0079】

一般に、タイルは、XMLデータファイルなどのデータ構造によって表される。各タイルに、どの情報または通信連絡先がそのタイルによって表されるかに関する命令、ならびにその情報または通信連絡先にアクセスし、かつ/またはこれと対話する複数の従来の手段のいずれかを表す特定のサービスへのポインタが含まれる。これらのサービスは、自動的にまたは手動で、事前に定義された、またはユーザ定義可能なサービスのライブラリから選択される。具体的に言うと、異なるサービスは、アクセス、受取、検索、および/または別に、従来の情報、情報のソース、もしくは通信連絡先との相互作用の機能を提供する共有コードまたは共有機能を表す。これらのサービスは、これらが単独でまたは組み合わされてのいずれかで使用され、1つまたは複数のタイルによって並列に使用することができるという意味で、共有される。その結果、ある実施形態で、複数のサービスが、従来の情報、情報のソース、または通信連絡先との複雑な相互作用を提供するために組み合わせて使用されることに留意されたい。

40

【0080】

サービスの一例は、従来のMAPIサーバに接続することによって電子メールフォルダを監視するのに必要な機能性である。サービスのもう1つの例は、電子メールメッセージを送信または受信する機能性である。関連するサービスが、たとえばインスタントメッセージングまたはピアツーピア通信方式など、複数の従来の方法を介する連絡先との通信または情報転送の機能性を提供する。サービスのもう1つの例が、テキストファイルのある

50

言語から別の言語に変換する機能性である。サービスのもう1つの例が、データベースを監視するのに必要な機能性である。サービスのもう1つの例に、ウェブサイトまたはリモートサーバからデータを受信するか検索する機能性が含まれる。明らかに、情報、情報のソース、または通信連絡先と相互作用する方法のどれをも、1つまたは複数のタイルにより使用される共有サービスとして実施することができる。

【0081】

さらに、上記のように、各タイルの命令に、どのタイプの情報または通信連絡先がタイルによって表されていても、それを表示する能力を有する複数の専用ビューワの1つへのポイントが含まれる。言い換えると、各タイルは、ユーザが特定の情報または連絡先をどのように表示することを望むかの定義と共に、ユーザが経過を追うことを望む情報または連絡先、ならびに情報または連絡先へのアクセスおよび相互作用に関する複数のサービスのいずれをも使用する能力の組合せを表す。そのようなアクセスまたは相互作用は、ローカルに、またはローカルイントラネット、エクストラネット、有線または無線ネットワーク、インターネットなどにわたって従来の通信プロトコルを介して達成することができる。

10

【0082】

ビューワは、1つまたは複数のサービスを介して検索された情報または連絡先データを有するサイズ変更可能なサムネイルまたはアイコンサイズのウィンドウとしてタイルをグラフィカルに表示する。具体的に言うと、ビューワは、テキスト情報、可聴情報、あるいは静止画像または生画像を含むグラフィカル情報、あるいはテキスト情報、可聴情報、またはグラフィカル情報の組合せを有するタイルをダイナミックに表示することができる。たとえば、あるビューワタイプは、連絡先情報を表示できるすなわち「パーソントイル」であり、別のタイプは、たとえば受信したメッセージの数または特定のソースからのメッセージの数などの特定の電子メール情報を表示することができ、もう1つのビューワは、データベースからの特定の情報のサマリをサムネイル内で提供するためにデータベースと対話するように設計される。ビューワタイプのさらなる例に、静止画像、ビデオ画像、通信状況のサマリ、データベース照会の結果などを表示できるビューワが含まれる。明らかに、任意のタイプのビューワを任意の対応するタイプの情報に関連付け、任意の可能な情報を表示できることを保証するように設計することができる。

20

【0083】

サイドバーは、特許文献3、表題「A system and process for providing dynamic communication access and information awareness」に詳細に記載されている。

30

【0084】

名前空間308のサイドバー機能性は、アプリケーション設計者が、サイドバーにタイルを追加できるようにする。

【0085】

通知（たとえば、可聴通知または視覚的通知）は、ユーザコンテキストシステムの一部であり、このユーザコンテキストシステムには、ある実施形態によれば、通知を処理する方法に関する判断のために比較される3つの要素が含まれる。第1の要素は、ユーザのコンテキスト（オペレーティングシステムおよびそのコンテキストを拡張した任意のプログラムによって提供することができる）である。第2の要素は、ユーザのルールおよびプリファレンスである。第3の要素は、通知自体（ユーザのルールと一致する可能性があるデータおよびプロパティなどの要素を含む）である。

40

【0086】

ユーザコンテキストシステムは、オペレーティングシステムおよびユーザのコンテキストを宣言する他のプログラムによって、システムがユーザのコンテキストおよびルールを仲介した後に動作する。通知は、システムへの呼出しを行う他のプログラムによって手前に上げられる。ユーザのコンテキスト、ルール、および通知の要素が、比較され、その後、通知に関して何を行うべきかに関する判定が行われる。通知に関して行うことができるものに関するさまざまなオプションの例に、拒否（たとえば、通知が人目を引くか騒ぐこ

50

とを許可されない場合に、通知は絶対にユーザによって見られない)、据置き(たとえば、ユーザのコンテキストが変化する、または配送が後に適するようになることがユーザのルールから示されるまで通知が保持される)、配送(たとえば、ユーザのコンテキストおよびルールに従って通知を配送することが許可される)、ルーティング(たとえば、通知が現在のシステムで配送を許可されるかどうかにかかわらず、通知を別のシステムに渡さなければならないことがユーザのルールによって示される)が含まれる。

【0087】

一般に、ユーザは、「求めに応じられない」とみなされる状態にある場合があり、その場合には、通知は、配送されないか、ユーザが「求めに応じられる」ようになるまで保持されるのいずれかになる。たとえば、ユーザが、フルスクリーンアプリケーションを実行している場合に、そのユーザを、求めに応じられないと見なすことができる。あるいは、ユーザは、「求めに応じることができる」が、ユーザに適するように通知を修正する必要がある状態である場合がある。たとえば、ユーザが、音楽を聴いているか会議中である場合に、そのユーザが、通知をユーザの画面に配送しなければならないが、通知が鳴らすサウンドを、音量を下げるか全く再生しないようにしなければならないことを指示することができる。

【0088】

上記のように、ユーザコンテキストは、通知をユーザの画面に表示するかどうかを部分的に決定することができる。通知を表示するときに、その通知をユーザコンテキスト内で、ある勾配 (gradient) に基づいて表示することができる。言い換えると、指定できる、描かれる通知の形態の異なるレベルの侵襲性 (invasiveness) がある。たとえば、従来の通知は、自由にクライアント領域にポップアウトし、短い間だけウィンドウを隠す。ユーザがわずかに制限的なコンテキストにいる場合、この通知は自由に表示できるが、別のウィンドウの上への描画を許可されないなど、より侵襲性でない形でのみ表示できるものとする。もう1つの例として、一実施形態で、ユーザが最大化されたアプリケーションを実行している場合に、デフォルト設定は、コンテキストがわずかに制限的であること、およびユーザが、このアプリケーションがクライアント領域全体を得ることを望むと明確に述べていることを意味するものとする。この設定では、通知はまだ描画することを許可されるが、サイドバー内に現れるだけにされることができる。言い換えると、通知描画形式におけるこのタイプの侵襲性の減少は、通知の影響を減らし、認識の負荷を全体的に減らす。

【0089】

ユーザコンテキストシステムは、特許文献4、表題「System and Method Utilizing Test Notifications」に詳細に記載されている。

【0090】

名前空間308の通知機能性は、アプリケーション設計者が、彼ら自身の通知を作成し、それらをデスクトップに表示させて、ユーザにとって重要である可能性がある何か(たとえば、電子メールメッセージが受信された、インスタントメッセージが受信されたなど)が発生したときにユーザに通知できるようにする。名前空間308の通知機能性によって、さらに、ユーザが複数のアプリケーションからの通知に関する自身のルールおよびプリファレンスを定義する中央ロケーションが提供され、ユーザが、同一のルールおよびプリファレンスを複数回(複数のアプリケーションのそれぞれに1回)セットするという煩わしさから解放される。

【0091】

さらに、Contacts名前空間で、連絡先情報に関するコントロールおよびダイアログのコレクションが定義される。この連絡先情報は、アイテムとしてストレージシステムに保管することができる。「連絡先」は、一般に、人、グループ、組織、会社、世帯、または他のタイプの識別可能なエンティティを指す。「連絡先情報」は、一般に、連絡先に対応する、連絡先の識別、接触、アクセス、対応、または通信に関係すると考えることができる情報を指す。連絡先情報は、アプリケーションによって、たとえば電子メール送

10

20

30

40

50

信、電話の発呼、ウェブサイトへのアクセス、ゲーミングセッションの開始、金融トランザクションの実行など、所望の機能を実行するのに使用される。連絡先情報の非制限的な例に、名前、別名、電話番号、電子メールアドレス、家の住所、インスタントメッセージング (IM) アドレス、およびウェブアドレスが含まれる。連絡先情報は、連絡先の状況など、他のタイプの情報を指すこともできる。たとえば、連絡先が現在オンラインであるか電話に出ていることを示す情報を、広義に連絡先情報と考えることもできる。

【0092】

Contacts 名前空間は、「System.Windows.Contacts」、「System.Windows.Controls」(図3のコントロール310)、または「System.Windows.Collaboration」など、プレゼンテーションサブシステム名前空間200の一部であるが、Shell 名前空間202の一部でないものとすることができる。その代わりに、Contacts 名前空間を、Shell 名前空間202の一部とすることができる。Contacts 名前空間は、System.Windows 名前空間の機能性を拡張するが、Shell の一部として実装する必要はない(すなわち、System.Windows.Explorer 名前空間302の一部である必要はない)。

【0093】

Contacts 名前空間を用いると、連絡先情報を、ユーザが1回だけ入力し、複数の異なるアプリケーションによってアクセスできるようになる。したがって、ユーザは、同一の連絡先情報を複数回(複数のアプリケーションのそれぞれについて1回)入力するという煩わしさから解放される。

【0094】

Contacts 名前空間のコントロールおよびダイアログには、下記が含まれる。

- ・ユーザがその保管された連絡先から連絡先を選択またはピックできるようにするために使用される Contact Picker ダイアログ。たとえば、ユーザが、電子メールメッセージが送信される連絡先またはインスタントメッセージングに関する連絡先を選択できるようにする。ユーザは、たとえば、システムストレージ内にアイテムとして保管されたユーザの連絡先のすべてのリストをブラウズでき、これらの連絡先の1つまたは複数を選択することができる。Contact Picker ダイアログは、特許文献5、表題「Contact Picker」に詳細に記載されている。

- ・自由形式の名前などの連絡先情報をユーザが入力できるようにする(たとえば名前をタイプすることによって)のに使用され、連絡先情報が入力されるときに、その情報をユーザの保管された連絡先からの1つまたは複数の連絡先に解決する Contact Textbox コントロール。解決された情報は、たとえば、ドロップダウンメニューに表示することができ、あるいは、ユーザが自由形式で名前を入力した場合にタイプイン行に自動的に挿入することができる。

- ・連絡先情報をユーザに表示するのに使用される Contact コントロール。連絡先情報の編集または自由形式入力は、このコントロールによってサポートされない。

【0095】

(名前空間メンバの例)

このセクションには、例の名前空間(たとえば、図3のプレゼンテーションサブシステム名前空間200内の名前空間)によって公開できるメンバの例を記述した複数のテーブルが含まれる。これらの公開されるメンバに、たとえば、クラス、インターフェース、列挙、および委任を含めることができる。これらの例に記載されたメンバが、例にすぎず、代替の他のメンバを名前空間によって公開できることを理解されたい。

【0096】

下記の名前空間の一部で、あるクラス、インターフェース、列挙、および委任の記述が空白のままにされていることを理解されたい。

【0097】

<<System.Windows.Explorer.Controls>>

10

20

30

40

50

以下に、System.Windows.Explorer.Controls 名前空間によって公開されるメンバの例を示す。

【0098】

< クラス >

DefaultCommandEventArgs - DefaultCommandEvent イベントに関するイベントデータを含む。

ExplorerView - フォルダの内容に関する情報をユーザが見られるようにする。

FolderSelectionChangedEventArgs - FolderSelectionChangedEvent イベントに関するイベントデータを含む。

IncludeItemEventArgs - IncludeItemEvent イベントに関するイベントデータを含む。 10

ItemVerbModifyEventArgs - ItemVerbModifyEventArgs クラスの新しいインスタンスを初期化する。

NavigationCompleteEventArgs - NavigationCompleteEvent イベントのイベントデータを含む。

NavigationFailedEventArgs - NavigationFailedEvent イベントのイベントデータを含む。

NavigationPendingEventArgs - NavigationPendingEvent イベントのイベントデータを含む。

【0099】

20

< 列挙 >

FolderViewFlags - ExplorerView のビュープロパティを示す。

FolderViewMode - ExplorerView のビューモードを示す。

【0100】

< 委任 >

DefaultCommandEventHandler - DefaultCommandEvent を処理するメソッドを表す。

FolderSelectionChangedEventHandler - FolderSelectionChangedEvent を処理するメソッドを表す。

IncludeItemEventHandler - IncludeItemEvent を処理するメソッドを表す。 30

ItemVerbModifyEventHandler - ItemVerbModifyEvent を処理するメソッドを表す。

NavigationCompleteEventHandler - NavigationCompleteEvent を処理するメソッドを表す。

NavigationFailedEventHandler - NavigationFailedEvent を処理するメソッドを表す。

NavigationPendingEventHandler - NavigationPendingEvent を処理するメソッドを表す。

【0101】

40

<< System.Windows.Explorer.Dialogs >>

以下に、System.Windows.Explorer.Dialogs 名前空間によって公開されるメンバの例を示す。

【0102】

< クラス >

ColorDialog

CommonDialog

DialogControlItemSelectedEventArgs

FileDialog - OpenFileDialog および SaveFileDialog Base の親クラスとして使用される抽象クラス。 50

FileDialogCheckBox - FileDialogに配置できるチェックボックスコントロールを表す

FileDialogComboBox

FileDialogContainerControlBase - FileDialogComboBox、FileDialogOpenDropDown、FileDialogRadioButtonGroup、およびFileDialogToolBarMenuの親クラスとして使用される抽象クラス。

FileDialogControlBase - FileDialogCheckBox、FileDialogEditBox、FileDialogPushButton、およびFileDialogContainerControlBaseの親クラスとして使用される抽象クラス。 10

FileDialogControlItem

FileDialogEditBox - FileDialogに配置できるテキストボックスコントロールを表す。

FileDialogOpenDropDown

FileDialogPushButton - FileDialogに配置できるボタンコントロールを表す。

FileDialogRadioButtonGroup

FileDialogToolBarMenu

FileOkEventArgs 20

FileType

OpenFileDialog - ユーザが1つまたは複数のファイルを開けるようにする。このクラスを継承することはできない。

SaveAsFileDialog - ユーザが、ファイルが保存されるロケーションを選択でき、ファイル名を指定できるようにする。このクラスを継承することはできない。

SaveFileDialog - ユーザが、ファイルを保存するロケーションを選択できるようにする。このクラスを継承することはできない。

SaveFileDialogBase - SaveAsFileDialogおよびSaveFileDialogの親クラスとして使用される抽象クラス。

【0103】 30

< 列挙 >

FileDialogLayout

TileAttributes

【0104】

< 委任 >

DialogControlItemSelectedEventHandler

FileOkEventHandler

【0105】

<<System.Windows.Desktop>>

以下に、System.Windows.Desktop名前空間によって公開されるメンバの例を示す。この名前空間には、サイドバーおよび通知にかかわる要素が含まれる。

【0106】

< クラス >

AnalogClockPanel

AppBar

AreaButton

BaseComTile

BaseSidebarClockSettings

BaseTile - カスタムサイドバータイル実装の親クラスとして使用される抽象クラス。 50

BasketControl	
CalendarElement	
CalendarImages	
ChildrenWontFitArgs	
ClockHacks	
ClockPanel	
DataSourceEventArgs	- データソースから渡されるイベントデータを表す。
DigitalDateTimeElement	
DragButton	
DragControlWindow	10
DraggableButton	
DragWindow	
DropArgs	
DropEventArgs	
ExtraSpaceArgs	
FillAlphaPresenter	
FillImageResourcePresenter	
FillPanel	
FillPresenter	
Flyout	20
FlyoutLinkClickEventArgs	- サイドバータイトルのフライアウト (flyout) ビューの最下部にあるオプションのリンクから発するイベントにตอบสนองして R M A A c t i o n E v e n t H a n d l e r に渡されるイベントデータを表す。
FlyoutPresenter	
FlyoutStuff	
FocusableButton	
FocusWithinWorkaroundHelper	
FolderContentsChangedEventArgs	- フォルダの内容の変化にตอบสนองして F o l d e r C o n t e n t s C h a n g e d E v e n t H a n d l e r に渡されるイベントデータを表す。
GlobalSetting	30
HackBorder	
HackImage	
ImageButton	
ImageResource	
ImageResourcePresenter	
ItemControl	
ItemToolBarControl	
ListMakerControl	
MenuStuff	40
NativeResource	
NativeResourceHelper	
NativeResources	
NativeResourceTypeConverter	
NormalButton	
Notification	- システムが1つのユーザインターフェースにおいてユーザに送るメッセージおよびそれに関連するデータを表す。
NotificationArea	
NotificationButton	- 通知に表示されるボタンを定義する。
NotificationClickedEventArgs	- 通知ウィンドウ内でのクリックイベントに関連する
	50

データを含む。

NotificationContext - アプリケーションが現在特定の状態であるか否かを宣言する。
この状態は、ユーザのコンテキストの定義の一部として使用される。

NotificationTimerPresenter

NotifyCompleteEventArgs

NotifyWindow

OptionsButton

OuterTile

OverflowableCollection

OverflowableControlCollection

10

OverflowPanel

OverflowPresenter

OverflowWrapper

PanelInfo

ProgressBar

QuickLaunchTile

RMAActionEventArgs - リッチミニマイズドアプリケーション (Rich Minimized Application、RMA) によって生成されるイベントデータを表す。RMAActionEventHandler によって消費される。

20

RMAData

SetDataArrayEventArgs - データアレイによって生成され、SetDataArrayEventHandler に送られるイベントデータを表す。

Sidebar

SidebarAlarmClock

SideBarClock

SidebarClockSettings

SlideShowTile

StackAlphaPresenter

StackImageResourcePresenter

StartButton

30

SyncHelper

SyncItemBar

SyncTile

Taskbar

TaskChevron

TaskGroup

TaskItem

TaskList

TaskOverflow

TaskOverflowableControlCollection

40

TaskPresenter

TestTile

TestTimeZone

TextElementFontInfo

Theme

ThemeHelper

ThemeHelperOld

ThemeImage

ThemeImageTypeConverter

ThemeResources

50

Tile
 TileCollection
 TileOverflow
 TileThumb
 TileThumbDotNet
 Timer
 TimerStuff
 TimeZone
 ViewStatusControl
 WebHostEventArgs 10
 WindowOrigin
 WindowOriginOld
 WindowStuff
 【 0 1 0 7 】
 < インターフェース >
 ICOMDataSourceHandler - 管理されないコードで実施されるデータソースを作成し、
 これと通信するのに使用される。
 IOverflow
 IOverflowList
 ISidebar - 個々のタイルをホスティングするサイドバーへのアクセスを提供する。サ 20
 イドバーホストは、フライアウトのオープンおよびクローズ、ショートカットメニューの
 表示、および個々のタイルに関連するイベントへの応答の責任を負う。
 ISidebarAlarm
 【 0 1 0 8 】
 < 列挙 >
 AlarmState
 BasketFlags
 DragWindowPos
 InvokeFolderActionEnum - ユーザがフォルダの I t e m C o n t r o l を呼び出すと
 きに行うことを決定するための列挙。I n v o k e F o l d e r A c t i o n および I t 30
 e m C o n t r o l I n v o k e F o l d e r A c t i o n と共に使用される。
 RMAAction - イベントアクションを定義するために A c t i o n プロパティと共に使
 用される定数。
 SelectionModeEnum
 【 0 1 0 9 】
 < 委任 >
 ChildrenWontFitHandler
 DocumentCompleteEventHandler
 DropEventHandler
 DropHandler 40
 DummyDelegateToSetupAppDomainProperly
 ExtraSpaceHandler
 FlyoutClosingEventHandler
 FlyoutLinkClickEventHandler - F l y o u t L i n k C l i c k E v e n t イベント
 を処理するメソッドを表す。
 FolderContentsChangedEventHandler
 NavigateErrorEventHandler
 NotificationClickedEventHandler - N o t i f i c a t i o n C l i c k e d E v 50
 e n t イベントを処理するメソッドを表す。
 NotifyCompleteEventHandler - N o t i f y C o m p l e t e E v e n t イベントを

処理するメソッドを表す。

RMAActionEventHandler - R M A A c t i o n E v e n t イベントを処理するメソッドを表す。

SetDataArrayEventHandler - S e t D a t a A r r a y E v e n t イベントを処理するメソッドを表す。

【 0 1 1 0 】

< < S y s t e m . W i n d o w s . C o n t r o l s > >

シェルコンポーネントによって使用されるクラスおよびインターフェースを提供する。

【 0 1 1 1 】

以下に、S y s t e m . W i n d o w s . C o n t r o l s 名前空間によって公開されるメンバの例を示す。 10

【 0 1 1 2 】

< クラス >

ContactControl

ContactPickerDialog

ContactPropertyRequest

ContactPropertyRequestCollection

ContactSelection

ContactSelectionCollection

ContactTextBox

20

ContactTextBoxSelectionChangedEventArgs

ContactTextBoxTextChangedEventArgs

ContactTextBoxTextResolvedEventArgs

IncludeContactEventArgs

IteratedEventArgs

Iterator

OKCancelApplyEventArgs

OKCancelApplyStrip

【 0 1 1 3 】

< 列挙 >

30

ContactControlPropertyPosition

ContactPickerDialogLayout

ContactPropertyCategory

ContactPropertyType

ContactType

OKCancelApplyType

【 0 1 1 4 】

< 委任 >

ContactTextBoxSelectionChangedEventHandler

ContactTextBoxTextChangedEventHandler

40

ContactTextBoxTextResolvedEventHandler

IncludeContactEventHandler

IteratedEventHandler

OKCancelApplyEventHandler

OpenedEventHandler

【 0 1 1 5 】

(例示的なコンピューティングシステムおよび環境)

図 4 に、プログラミングフレームワーク 1 3 2 を (完全にまたは部分的にのいずれかで) 実施できる、適するコンピューティング環境 4 0 0 の例を示す。コンピューティング環境 4 0 0 は、本明細書に記載のコンピュータアーキテクチャおよびネットワークアーキテ 50

クチャでを使用することができる。

【0116】

例示的なコンピューティング環境400は、コンピューティング環境の一例にすぎず、コンピュータおよびネットワークアーキテクチャの使用の範囲または機能性に関する制限を示唆することを意図されたものではない。また、コンピューティング環境400を、例示的なコンピューティング環境400に示された、構成要素のいずれかまたはその組合せに関する依存性または要件を有するものと解釈してもならない。

【0117】

フレームワーク132を、多数の他の汎用または専用コンピューティングシステム環境または構成と共に実施することができる。使用に適する可能性がある衆知のコンピューティングシステム、コンピューティング環境、および/または構成の例に、パーソナルコンピュータ、サーバコンピュータ、マルチプロセッサシステム、マイクロプロセッサベースのシステム、ネットワークPC、ミニコンピュータ、メインフレームコンピュータ、上記のシステムまたは装置のいずれかを含む分散コンピューティング環境などが含まれるが、これに制限されない。このフレームワークのコンパクト版またはサブセット版を、携帯電話、携帯情報端末、ハンドヘルドコンピュータ、または他の通信/コンピューティング装置などの限られたリソースのクライアントで実施することもできる。

10

【0118】

フレームワーク132を、1つまたは複数のコンピュータまたは他の装置によって実行される、プログラムモジュールなどのコンピュータ実行可能命令の全般的な文脈で説明することができる。一般に、プログラムモジュールには、特定のタスクを実行するか特定の抽象データ型を実装する、ルーチン、プログラム、オブジェクト、コンポーネント、データ構造などが含まれる。フレームワーク132は、通信ネットワークを介してリンクされたりリモート処理装置によってタスクが実行される分散コンピューティング環境で実践することもできる。分散コンピューティング環境では、プログラムモジュールを、メモリストレージ装置を含むローカルおよびリモートの両方のコンピュータストレージ媒体に配置することができる。

20

【0119】

コンピューティング環境400に、コンピュータ402の形の汎用コンピューティング装置が含まれる。コンピュータ402のコンポーネントには、1つまたは複数のプロセッサまたは処理装置404、システムメモリ406、および処理装置404を含むさまざまなシステムコンポーネントをシステムメモリ406に接続するシステムバス408を含めることができるが、これに制限はされない。

30

【0120】

システムバス408は、メモリバスまたはメモリコントローラ、周辺バス、AGP (Accelerated Graphics Port)、およびさまざまなバスアーキテクチャのいずれかを使用するプロセッサバスまたはローカルバスを含む、複数の可能なタイプのバス構造の1つまたは複数を表す。たとえば、そのようなアーキテクチャに、ISA (Industry Standard Architecture) バス、MCA (Micro Channel Architecture) バス、EISA (Enhanced ISA) バス、VESA (Video Electronics Standards Association) ローカルバス、およびメザニンバスとも称するPCI (Peripheral Component Interconnects) バスを含めることができる。

40

【0121】

コンピュータ402に、通常は、さまざまなコンピュータ可読媒体が含まれる。そのような媒体は、揮発性媒体および不揮発性媒体、リムーバブル媒体および固定の媒体の両方を含む、コンピュータ402によってアクセス可能である任意の使用可能な媒体とすることができる。

【0122】

50

システムメモリ 406 には、ランダムアクセスメモリ (RAM) 410 などの揮発性メモリおよび / または読取専用メモリ (ROM) 412 などの不揮発性メモリの形のコンピュータ可読媒体が含まれる。起動中などにコンピュータ 402 内の要素の間での情報の転送を助ける基本ルーチンを含む基本入出力システム (BIOS) 414 が、ROM 412 に保管される。RAM 410 には、通常は、処理装置 404 によって即座にアクセス可能および / または処理装置 404 によって現在操作されているデータおよび / またはプログラムモジュールが含まれる。

【0123】

コンピュータ 402 に、他のリムーバブル / 固定の、揮発性 / 不揮発性コンピュータストレージを含めることもできる。たとえば、図 4 に、固定の不揮発性磁気媒体 (図示せず) から読み取り、これに書き込むハードディスクドライブ 416、リムーバブル不揮発性磁気ディスク 420 (たとえば「フロッピー (登録商標) ディスク」) から読み取り、これに書き込む磁気ディスクドライブ 418、および CD-ROM、DVD-ROM、または他の光媒体などのリムーバブル不揮発性光ディスク 424 から読み取り、かつ / またはこれに書き込む光ディスクドライブ 422 が含まれる。ハードディスクドライブ 416、磁気ディスクドライブ 418、および光ディスクドライブ 422 は、それぞれ、1 つまたは複数のデータメディアインターフェース 423 によってシステムバス 408 に接続される。その代わりに、ハードディスクドライブ 416、磁気ディスクドライブ 418、および光ディスクドライブ 422 を、1 つまたは複数のインターフェース (図示せず) によってシステムバス 408 に接続することができる。

【0124】

ディスクドライブおよびそれに関連するコンピュータ可読媒体は、コンピュータ 402 の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、および他のデータの揮発性ストレージを提供する。この例では、ハードディスク 416、リムーバブル磁気ディスク 420、およびリムーバブル光ディスク 424 が示されているが、磁気カセットまたは他の磁気記憶装置、フラッシュメモリカード、CD-ROM、デジタル多用途ディスク (DVD)、または他の光ストレージ、ランダムアクセスメモリ (RAM)、読取専用メモリ (ROM)、電気的消去可能プログラマブル読取専用メモリ (EEPROM)、および類似物などのコンピュータによってアクセス可能なデータを保管できる他のタイプのコンピュータ可読媒体を使用して、例示的なコンピューティングシステムおよび環境を実施

【0125】

たとえば、オペレーティングシステム 426、1 つまたは複数のアプリケーションプログラム 428、他のプログラムモジュール 430、およびプログラムデータ 432 を含む、任意の個数のプログラムモジュールを、ハードディスク 416、磁気ディスク 420、光ディスク 424、ROM 412、および / または RAM 410 に保管することができる。オペレーティングシステム 426、1 つまたは複数のアプリケーションプログラム 428、他のプログラムモジュール 430、およびプログラムデータ 432 (またはその組合せ) に、プログラミングフレームワーク 132 の要素を含めることができる。

【0126】

ユーザは、キーボード 434 およびポインティング装置 436 (たとえば「マウス」) などの入力装置を介してコンピュータ 402 にコマンドおよび情報を入力することができる。他の入力装置 438 (具体的には図示せず) に、マイクロホン、ジョイスティック、ゲームパッド、衛星パラボラアンテナ、シリアルポート、スキャナ、および / または類似物を含めることができる。上記および他の入力装置は、システムバス 408 に接続された入出力インターフェース 440 を介して処理装置 404 に接続されるが、パラレルポート、ゲームポート、または USB (Universal Serial Bus) などの他のインターフェースおよびバス構造によって接続することができる。

【0127】

モニタ 442 または他のタイプの表示装置を、ビデオアダプタ 444 などのインターフ

10

20

30

40

50

ェースを介してシステムバス408に接続することもできる。モニタ442のほかに、他の出力周辺装置に、スピーカ（図示せず）およびプリンタ446など、入出力インターフェース440を介してコンピュータ402に接続できるコンポーネントを含めることができる。

【0128】

コンピュータ402は、リモートコンピューティング装置448などの1つまたは複数のリモートコンピュータへの論理接続を使用して、ネットワーク化された環境で動作することができる。たとえば、リモートコンピューティング装置448を、パーソナルコンピュータ、ポータブルコンピュータ、サーバ、ルータ、ネットワークコンピュータ、ピアデバイス、または他の一般的なネットワークノード、などとすることができる。リモートコンピューティング装置448は、コンピュータ402に関して本明細書で説明した要素および特徴の多くまたはすべてを含めることができるポータブルコンピュータとして図示されている。

10

【0129】

コンピュータ402とリモートコンピュータ448の間の論理接続は、ローカルエリアネットワーク（LAN）450および一般的な広域ネットワーク（WAN）452として図示されている。そのようなネットワーキング環境は、オフィス、会社規模のコンピュータネットワーク、イントラネット、およびインターネットでありふれたものである。

【0130】

LANネットワーキング環境において実施されるときに、コンピュータ402は、ネットワークインターフェースまたはネットワークアダプタ454を介してローカルネットワーク450に接続される。WANネットワーキング環境において実施されるときに、コンピュータ402に、通常は、広域ネットワーク452を介する通信を確立する、モデム456または他の手段が含まれる。モデム456は、コンピュータ402の内蔵または外付けとすることができるが、入出力インターフェース440または他の適当な機構を介してシステムバス408に接続することができる。図示のネットワーク接続が、例示的であり、コンピュータ402と448の間の通信リンクを確立する他の手段を使用できることを理解されたい。

20

【0131】

コンピューティング環境400と共に図示されたものなどのネットワーク化された環境において、コンピュータ402に関して示したプログラムモジュールまたはその一部を、リモートメモリ記憶装置に保管することができる。たとえば、リモートアプリケーションプログラム458が、リモートコンピュータ448の記憶装置に常駐している。例示のために、アプリケーションプログラムおよびオペレーティングシステムなどの他の実行可能なプログラムコンポーネントが、本明細書では別個のブロックに図示されているが、そのようなプログラムおよびコンポーネントが、さまざまなときにコンピューティング装置402の異なるストレージコンポーネントに常駐することができ、コンピュータのデータプロセッサによって実行されることを理解されたい。

30

【0132】

フレームワーク132、具体的にはAPI142またはAPI142に対して行われる呼出しの実装を、ある形のコンピュータ可読媒体を介して保管または伝送することができる。コンピュータ可読媒体は、コンピュータによってアクセスできる任意の使用可能な媒体とすることができる。制限ではなく例として、コンピュータ可読媒体に、「コンピュータストレージ媒体」および「通信媒体」を含めることができる。「コンピュータストレージ媒体」には、揮発性および不揮発性、取外し可能および取外し不能の、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のストレージに関する方法またはテクノロジーで実施される媒体が含まれる。コンピュータストレージ媒体には、RAM、ROM、EEPROM、フラッシュメモリ、または他のメモリテクノロジー、CD-ROM、デジタル多用途ディスク（DVD）、または他の光ストレージ、磁気カセット、磁気テープ、磁気ディスクストレージ、または他の磁気記憶装置、あるいは、

40

50

所望の情報を保管するのに使用でき、コンピュータによってアクセスできる他のすべての媒体が含まれるが、これに制限はされない。

【0133】

「通信媒体」によって、通常は、搬送波または他のトランスポート機構などの変調されたデータ信号内で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、または他のデータが実施される。通信媒体には、すべての情報配信媒体も含まれる。用語「変調されたデータ信号」は、信号内で情報をエンコードする形で、その特性の1つまたは複数を設定または変更された信号を意味する。制限ではなく例として、通信媒体に、有線ネットワークまたは直接配線接続などの有線媒体と、音響、RF、赤外線、および他の無線媒体などの無線媒体が含まれる。上記のいずれかの組合せも、コンピュータ可読媒体の範囲に含まれる。

10

【0134】

その代わりに、フレームワークの諸部分を、ハードウェアで、あるいはハードウェア、ソフトウェア、および/またはファームウェアの組合せで実施することができる。たとえば、1つまたは複数の特定用途向け集積回路(ASIC)またはプログラマブルロジックデバイス(PLD)を設計またはプログラムして、本フレームワークの1つまたは複数の部分を実施することができる。

【0135】

プログラミングインターフェース(または単にインターフェース)は、コードの1つまたは複数のセグメントが、コードの1つまたは複数の他のセグメントと通信するかそのセグメントによって提供される機能性にアクセスできるようにする機構、処理、プロトコルとみなすことができる。その代わりに、プログラミングインターフェースを、他のコンポーネントの1つまたは複数の機構、メソッド、関数呼出し、モジュールなどとの通信接続が可能なシステムのコンポーネントの1つまたは複数の機構、メソッド、関数呼出し、モジュール、オブジェクトなどとみなすことができる。前の文の用語「コードのセグメント」は、コードの1つまたは複数の命令または行を含むことが意図されており、これには、使用される用語、コードセグメントが別々にコンパイルされるかどうか、コードセグメントがソースコード、中間コード、またはオブジェクトコードのどれとして提供されるか、コードセグメントがランタイムシステム内またはプロセス内で使用されるかどうか、コードセグメントが同一のマシンまたは異なるマシンに配置されるか、あるいはマシンにまたがって分散されるかどうか、コードのセグメントによって表される機能性が、完全にソフトウェアで、完全にハードウェアで、あるいはハードウェアとソフトウェアの組合せのどれで実施されるかに無関係に、たとえば、コードモジュール、オブジェクト、サブルーチン、関数などが含まれる。

20

30

【0136】

概念上、プログラミングインターフェースは、一般的に、図5または図6に示されているように見ることができる。図5に、第1コードセグメントと第2コードセグメントがそれを介して通信する導管としてのインターフェースInterface 1を示す。図6に、システムの第1コードセグメントおよび第2コードセグメントが媒体Mを介して通信できるようにする、インターフェースオブジェクトI1およびI2(第1および第2コードセグメントの一部とすることも、そうしないこともできる)を含むものとしてインターフェースを示す。図6に鑑みて、インターフェースオブジェクトI1およびI2を、同一のシステムの別々のインターフェースと考えることができ、オブジェクトI1およびI2と媒体Mにインターフェースが含まれると考えることもできる。図5および7に、両方向のフローおよびフローの両側のインターフェースを示したが、ある実施形態では、一方向の情報フローだけを有する(または、下で示すように情報フローなし)ことができ、あるいは、片側だけにインターフェースオブジェクトを有することができる。制限ではなく例として、アプリケーションプログラミングまたはプログラムインターフェース(API)、エントリポイント、メソッド、関数、サブルーチン、リモートプロシージャコール、およびコンポーネントオブジェクトモデル(COM)インターフェースなどの用語が、プログ

40

50

ラミングインターフェースの定義に含まれる。

【0137】

そのようなプログラミングインターフェースの態様に、第1コードセグメントが情報（ここで、「情報」は、その最も広い意味で使用され、「情報」には、データ、コマンド、要求などが含まれる）を第2コードセグメントに送る方法と、第2コードセグメントがその情報を受け取る方法と、情報の構造、シーケンス、構文、編成、スキーマ、タイミング、および内容とを含めることができる。これに関して、基礎になるトランスポート媒体自体は、媒体が有線、無線またはその両方の組合せのどれであれ、インターフェースによって定義される形で情報がトランスポートされる限り、インターフェースの動作に重要でない可能性がある。ある状況では、情報転送が別の機構を介する（たとえば、情報が、コードセグメントの間で流れる情報と別に、バッファ、ファイルなどに置かれる）ので、従来の意味で一方向または両方向に渡されないか、あるコードセグメントが第2コードセグメントによって実行される機能性に単にアクセスするので情報が存在しない場合がある。これらの態様のいずれかまたはすべてが、所与の状況で、たとえば、コードセグメントが疎結合された構成または密結合された構成のどちらの一部であるかに応じて、重要になる可能性があり、したがって、このリストは、例示的であり、非限定的とみなされなければならない。

10

【0138】

プログラミングインターフェースのこの概念は、当業者に既知であり、本発明の前述の詳細な説明から明白である。しかし、プログラミングインターフェースを実施する他の形があり、明示的に除外されない限り、これらも、本明細書に記載の請求項に含まれることが意図されている。そのような他の形は、図5および7の極度に単純化された図よりも洗練されて見える、または複雑に見える可能性があるが、それでも、そのような他の形によって、同一の全体的な結果を達成する類似する機能が実行される。これから、プログラミングインターフェースの例示的な代替実施形態を短く説明する。

20

【0139】

A. 因数分解

あるコードセグメントから別のコードセグメントへの通信は、通信を複数の個別の通信に分解することによって、間接的に達成することができる。これを、図7および9に概略的に示す。図からわかるように、いくつかのインターフェースを、機能性の分割可能なセットとして記述することができる。したがって、数学的に24または $2 \times 2 \times 3 \times 2$ を提供するように、図5および7のインターフェースの機能性を因数分解して、同一の結果を達成することができる。したがって、図7からわかるように、インターフェースInterface 1によって提供される機能を再分割して、このインターフェースの通信を複数のインターフェース、Interface 1A、Interface 1B、Interface 1Cなどに変換し、同一の結果を達成することができる。図8からわかるように、インターフェースI1によって提供される機能性を、複数のインターフェースI1a、I1b、I1cなどに再分割し、それと同時に同一の結果を達成することができる。同様に、第1コードセグメントから情報を受け取る第2コードセグメントのインターフェースI2を、複数のインターフェースI2a、I2b、I2cなどに因数分解することができる。因数分解するときに、第1コードセグメントに含まれるインターフェースの数が、第2コードセグメントに含まれるインターフェースの数と一致する必要はない。図7および9のどちらの場合でも、インターフェースInterface 1およびI1の機能的趣旨は、それぞれ図5および7と同一のままである。インターフェースの因数分解は、関連するプロパティ、累算的プロパティ、および他の数学的プロパティに従う場合もあり、因数分解の理解が困難な場合がある。たとえば、動作の順序付けは、重要でない場合があり、その結果、あるインターフェースによって実行される機能を、別のコードまたはインターフェースによって、インターフェースに達するよりかなり前に実行することができ、あるいは、システムの別々の構成要素によって実行することができる。さらに、プログラミング技術の当業者は、同一の結果を達成する異なる関数呼出しを行うさまざまな形があること

30

40

50

を理解することができる。

【0140】

B．再定義

いくつかの場合に、初期の結果を達成しながら、プログラミングインターフェースのある態様（たとえばパラメータ）を無視し、追加し、または再定義することが可能である場合がある。これを、図9および11に示す。たとえば、図5のインターフェースInterface 1に、関数呼出しSquare(input、precision、output)すなわち、3つのパラメータinput、precision、およびoutputを含む呼出しであって、第1コードセグメントから第2コードセグメントに発行される関数呼出しが含まれると仮定する。真ん中のパラメータprecisionが、図9に示されているように、所与のシナリオで関係がない場合に、無視することができ、意味のない（この状況）パラメータと置換することができる。関係のない追加のパラメータを追加することもできる。どちらの場合でも、inputが第2コードセグメントによって二乗された後にoutputが返される限り、square（二乗）の機能性を達成することができる。precision（精度）は、コンピューティングシステムの下流または他の部分に意味のあるパラメータとすることもできるが、二乗計算の狭い目的について精度が必要でないことが理解されたならば、置換するか無視することができる。たとえば、結果に悪影響を及ぼさずに、有効な精度値を渡すのではなく、誕生日などの意味のある値を渡すことができる。同様に、図10に示されているように、インターフェースI1が、インターフェースへのパラメータを無視するか追加するために再定義されたインターフェースI1'に置換される。インターフェースI2は、同様に、不要なパラメータを無視するか、ほかで処理できるパラメータについて再定義されたインターフェースI2'として再定義することができる。ここでのポイントは、いくつかの場合に、プログラミングインターフェースに、パラメータなど、ある目的に不要な態様が含まれる場合があり、したがって、それらを無視するか再定義することができ、あるいは、他の目的のために他の形で処理できるということである。

【0141】

C．インラインコーディング

2つの別々のコードモジュールの機能性の一部またはすべてをマージし、これらの間の「インターフェース」の形を変えることが実行可能である場合もある。たとえば、図5および7の機能性を、それぞれ図11および13の機能性に変換することができる。図11では、図5の前の第1コードセグメントおよび第2コードセグメントが、その両方を含む1つのモジュールにマージされている。この場合に、コードセグメントは、まだ互いに通信することができるが、インターフェースは、単一モジュールにより適する形に適合させることができる。したがって、たとえば、前のCallステートメントおよびReturnステートメントは、もはや不要とすることができるが、インターフェースInterface 1に準ずる類似する処理または応答を、まだ有効とすることができる。同様に、図12では、図6のインターフェースI2の一部（またはすべて）を、インターフェースI1にインラインで記述して、インターフェースI1''を形成することができる。図からわかるように、インターフェースI2は、I2aおよびI2bに分割され、インターフェース部分I2aは、インターフェースI1とインラインにコーディングされて、インターフェースI1''を形成する。具体的な例として、図6のインターフェースI1が、関数呼出しsquare(input、output)を実行し、これがインターフェースI2によって受け取られ、インターフェースI2は、第2コードセグメントによるinputを用いて渡された値の処理（二乗）の後に、二乗された結果にoutputを用いて返すことを検討されたい。その場合に、第2コードセグメントによって実行される処理（inputの二乗）を、インターフェースへの呼出しなしで、第1コードセグメントによって実行することができる。

【0142】

D．分離(divorce)

あるコードセグメントから別のコードセグメントへの通信は、通信を複数の個別の通信に分解することによって、間接的に達成することができる。これを、図 13 および 15 に示す。図 13 からわかるように、1 または複数のミドルウェア（機能性および / またはインターフェース機能を元のインターフェースから分離するので、分離インターフェース）を設けて、第 1 インターフェース `Interface 1` の通信を、異なるインターフェース、この例ではインターフェース `Interface 2 A`、`Interface 2 B`、および `Interface 2 C` に従うように変換する。たとえば、これは、たとえば `Interface 1` プロトコルに従ってオペレーティングシステムと通信するように設計されたアプリケーションのインストールベースがあるが、そのオペレーティングシステムが、異なるインターフェース、この例ではインターフェース `Interface 2 A`、`Interface 2 B`、および `Interface 2 C` を使用するように変更される場合に行うことができる。ポイントは、第 2 コードセグメントによって使用された元のインターフェースが、もはや第 1 コードセグメントによって使用されるインターフェースと互換でなくなるように変更され、したがって、古いインターフェースと新しいインターフェースとに互換性を持たせるため、仲介物が使用されることである。同様に、図 14 からわかるように、インターフェース `I 1` からの通信を受け取る分離インターフェース `DI 1` を有し、インターフェース機能性を、たとえば、`DI 2` と共に働くが同一の機能的結果を提供するように再設計されているインターフェース `I 2 a` および `I 2 b` に送る、分離インターフェース `DI 2` を有する第 3 コードセグメントを導入することができる。同様に、`DI 1` および `DI 2` は、同一のまたは類似する機能的結果を提供しながら、図 6 のインターフェース `I 1` および `I 2` の機能性を変換するように一緒に働くことができる。

【0143】

E. 再記述

もう 1 つの可能な変形形態が、コードを動的に再記述して、インターフェース機能性を、同一の総合的な結果を達成する別のものに置換することである。たとえば、中間言語（たとえば、`Microsoft`（登録商標）`IL`、`Java`（登録商標）`Byte Code` など）で提示されるコードセグメントを、実行環境（`.NET framework`（登録商標）、`Java`（登録商標）ランタイム環境、または類似するランタイムタイプ環境によって提供されるものなど）内のジャストインタイム（`JIT`）コンパイラまたはインタープリタに供給するシステムがある。`JIT` コンパイラは、第 1 コードセグメントから第 2 コードセグメントへの通信を動的に変換する、すなわち、それらを、第 2 コードセグメント（元のまたは異なる第 2 コードセグメント）が必要とする可能性がある異なるインターフェースに合わせることができる。これを、図 15 および 17 に示す。図 15 からわかるように、この手法は、上で説明した分離シナリオに似ている。これは、たとえば、アプリケーションのインストールベースが、`Interface 1` プロトコルに従ってオペレーティングシステムと通信するように設計されているが、オペレーティングシステムが、異なるインターフェースを使用するように変更される場合に行うことができる。`JIT` コンパイラを使用して、通信を、オンザフライでインストールベースアプリケーションからオペレーティングシステムの新しいインターフェースに合わせることができる。図 16 からわかるように、インターフェースを動的に再記述するこの手法は、動的因数分解、または他の形のインターフェースの変更にも適用することができる。

【0144】

代替実施形態を介してインターフェースとして同一のまたは類似する結果を達成する、上で説明したシナリオを、さまざまな形で、順次および / または並列で、または他の介在コードと共に、組みわせることもできることに留意されたい。したがって、上で提示した代替実施形態は、相互に排他的ではなく、混合され、マッピングされ、かつ組み合わされて、図 5 および 7 で提示された包括的なシナリオと同一のまたは同等のシナリオを作ることができる。また、ほとんどのプログラミング構造と同様に、本明細書に記載されていないが、それでも本発明の趣旨および範囲によって表されるインターフェースの同一のまたは類似する機能性を達成する他の類似する形があることに留意されたい、すなわち、イ

ンターフェースの価値の基礎になるのは、少なくとも部分的に、インターフェースによって表される機能性と、インターフェースによって使用可能にされる有益な結果である。

【 0 1 4 5 】

(結 論)

本発明を、構造的特徴および / または方法論的動作に固有の言葉で説明したが、請求項で定義される本発明は、説明した特定の特徴または動作に必ずしも制限されないことを理解されたい。そうではなく、特定の特徴および動作は、請求される発明を実施する例示的な形として開示されたものである。

【 図面の簡単な説明 】

【 0 1 4 6 】

10

【 図 1 】クライアントが、従来のプロトコルを使用するインターネットを介してウェブサービスにアクセスする、ネットワークアーキテクチャを示す図である。

【 図 2 】アプリケーションプログラムインターフェース (A P I) を含むネットワークプラットフォームのソフトウェアアーキテクチャを示すブロック図である。

【 図 3 】 A P I によってサポートされる独自の名前空間ならびにさまざまな A P I 関数の機能クラスを示すブロック図である。

【 図 4 】ソフトウェアアーキテクチャのすべてまたは一部を実行することができる例示的コンピュータを示すブロック図である。

【 図 5 】プログラミングインターフェースの例の実施形態を示す図である。

【 図 6 】プログラミングインターフェースの例の実施形態を示す図である。

20

【 図 7 】プログラミングインターフェースの例の実施形態を示す図である。

【 図 8 】プログラミングインターフェースの例の実施形態を示す図である。

【 図 9 】プログラミングインターフェースの例の実施形態を示す図である。

【 図 1 0 】プログラミングインターフェースの例の実施形態を示す図である。

【 図 1 1 】プログラミングインターフェースの例の実施形態を示す図である。

【 図 1 2 】プログラミングインターフェースの例の実施形態を示す図である。

【 図 1 3 】プログラミングインターフェースの例の実施形態を示す図である。

【 図 1 4 】プログラミングインターフェースの例の実施形態を示す図である。

【 図 1 5 】プログラミングインターフェースの例の実施形態を示す図である。

【 図 1 6 】プログラミングインターフェースの例の実施形態を示す図である。

30

【 符号の説明 】

【 0 1 4 7 】

1 0 0 ネットワーク環境

1 0 4 ネットワーク

1 0 6 , 1 1 2 , 1 1 4 , 1 2 2 ~ 1 2 9 通信リンク

1 3 2 プログラミングフレームワーク

1 4 2 A P I レイヤ

4 0 0 コンピューティング環境

4 1 6 ハードディスクドライブ

4 1 8 磁気ディスクドライブ

40

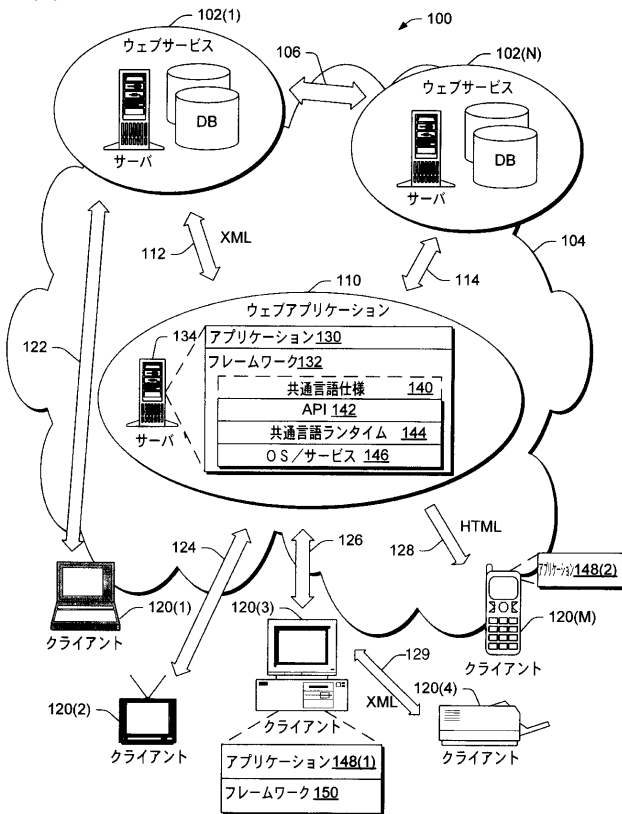
4 2 2 光ディスクドライブ

4 2 0 リムーバブル不揮発性磁気ディスク

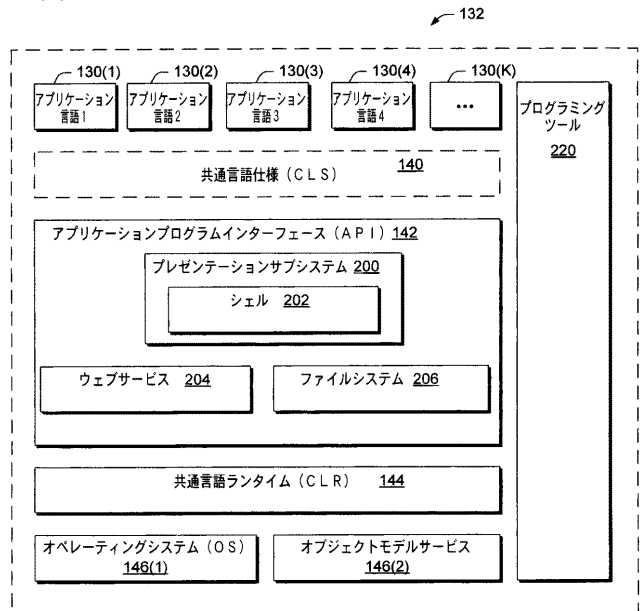
4 2 4 リムーバブル不揮発性光ディスク

4 3 8 入力装置

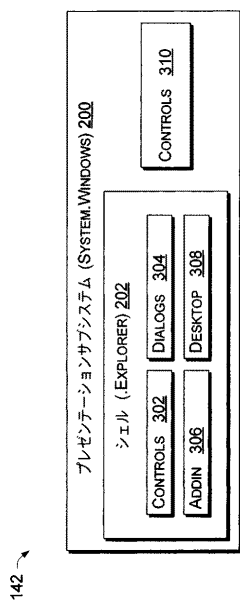
【図 1】



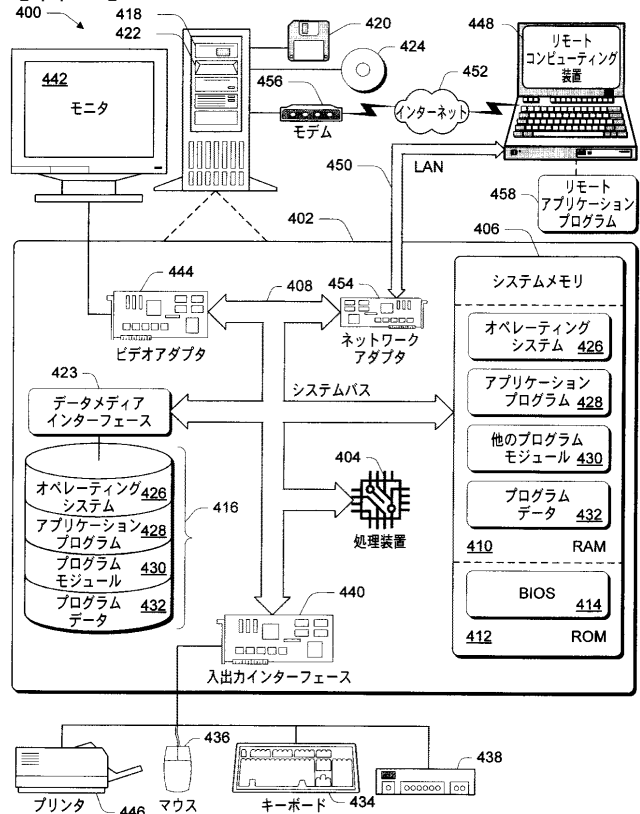
【図 2】



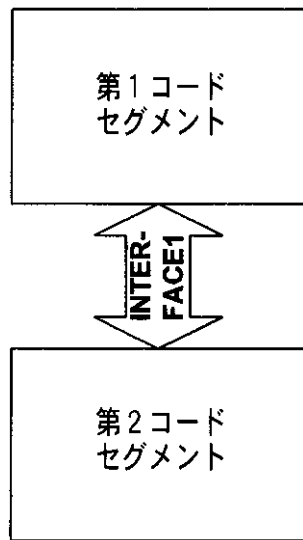
【図 3】



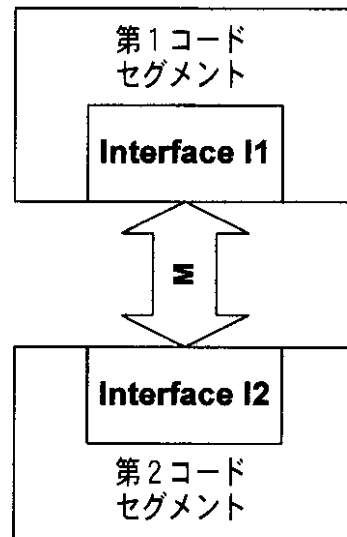
【図 4】



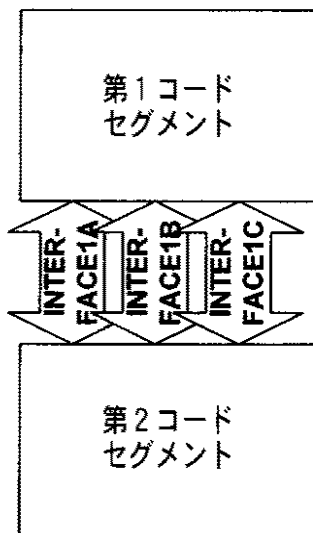
【図 5】



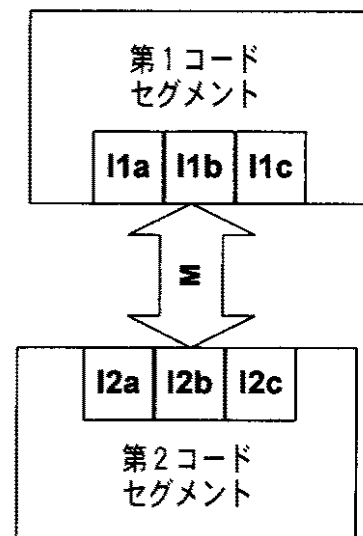
【図 6】



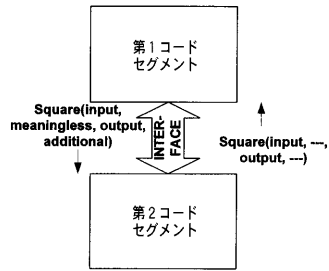
【図 7】



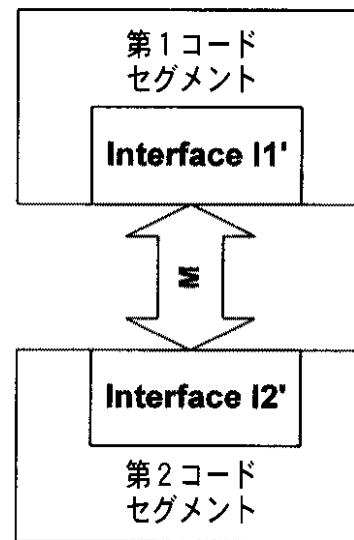
【図 8】



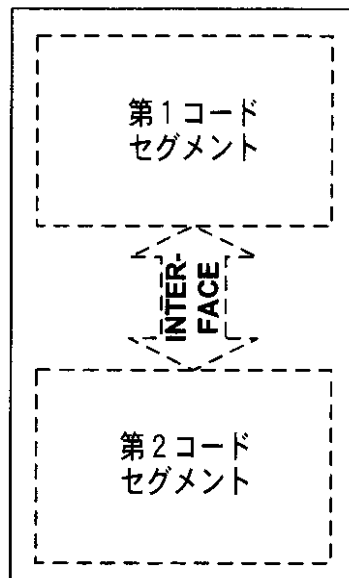
【図 9】



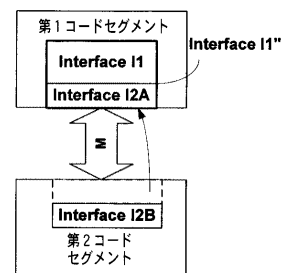
【図 10】



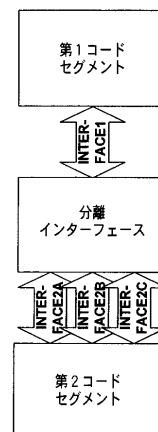
【図 11】



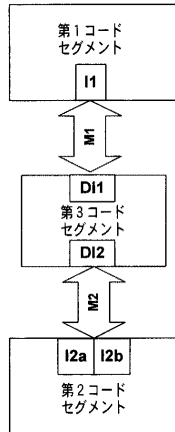
【図 12】



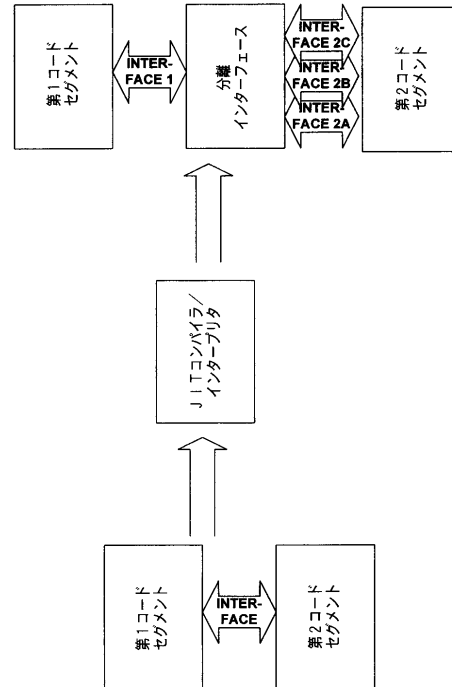
【図 13】



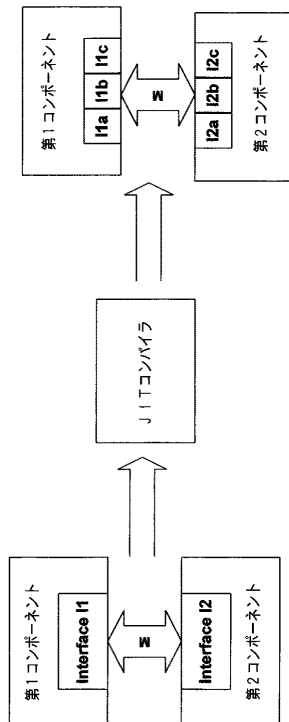
【図 14】



【図 15】



【図 16】



フロントページの続き

- (72)発明者 ケレム ビー・カラタル
アメリカ合衆国 98052 ワシントン州 レッドモンド ワン マイクロソフト ウェイ マ
イクロソフト コーポレーション内
- (72)発明者 マーク エム・ミラー
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
- (72)発明者 マイケル ジー・シェルダン
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 ティモシー ピー・マッキー
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

F ターム(参考) 5B076 AA13 DD05

【外国語明細書】

2005174283000001.pdf