

(19) 日本国特許庁 (JP)

(12) 特 許 公 報 (B2)

(11) 特許番号

特許第5938031号  
(P5938031)

(45) 発行日 平成28年6月22日 (2016. 6. 22)

(24) 登録日 平成28年5月20日 (2016. 5. 20)

(51) Int. Cl.

F I

G 0 6 F 9/44 (2006.01)

G 0 6 F 9/06 6 2 0 K

請求項の数 16 (全 20 頁)

(21) 出願番号	特願2013-504916 (P2013-504916)	(73) 特許権者	314015767
(86) (22) 出願日	平成23年3月25日 (2011. 3. 25)		マイクロソフト テクノロジー ライセン
(65) 公表番号	特表2013-524381 (P2013-524381A)		シング, エルエルシー
(43) 公表日	平成25年6月17日 (2013. 6. 17)		アメリカ合衆国 ワシントン州 9805
(86) 国際出願番号	PCT/US2011/030068		2 レッドモンド ワン マイクロソフト
(87) 国際公開番号	W02011/129989		ウェイ
(87) 国際公開日	平成23年10月20日 (2011. 10. 20)	(74) 代理人	100140109
審査請求日	平成26年3月18日 (2014. 3. 18)		弁理士 小野 新次郎
(31) 優先権主張番号	12/760, 565	(74) 代理人	100075270
(32) 優先日	平成22年4月15日 (2010. 4. 15)		弁理士 小林 泰
(33) 優先権主張国	米国 (US)	(74) 代理人	100101373
			弁理士 竹内 茂雄
		(74) 代理人	100118902
			弁理士 山本 修

最終頁に続く

(54) 【発明の名称】 プラットフォーム非依存プレゼンテーションの構成

(57) 【特許請求の範囲】

【請求項 1】

プロセッサによって実行される実行可能な命令を記憶するコンピュータ可読媒体を有するコンピュータで実施される視覚化システムであって、

ユーザエクスペリエンスに関連するコンポーネントのためのコンポーネント定義および前記コンポーネント定義に基づいて選択されたデータ定義を含むストア定義と、

ストア定義に基づいて特定のプラットフォームのために構成された視覚化ホストのユーザエクスペリエンスに対応する出力コンポーネントのインスタンスを自動的に構成する構成エンジンと、を備え、

前記コンポーネント定義は、ベースコンポーネントおよびコンテナコンポーネントの構成のための定義を含み、

前記コンテナコンポーネントはベースコンポーネントのコンテナであり、プラットフォーム非依存であり、

前記ベースコンポーネントは、前記特定のプラットフォームに向けて実装されており、

前記出力コンポーネントは前記ベースコンポーネントおよび前記コンテナコンポーネントの組み合わせを含み、グローバル変数が適用されていることを特徴とするシステム。

【請求項 2】

前記出力コンポーネントは、前記ユーザエクスペリエンスの前記出力コンポーネントにより表示されるデータのタイプであるターゲットデータタイプに基づいて構成されることを特徴とする請求項 1 に記載のシステム。

10

20

## 【請求項 3】

コンポーネントが前記ターゲットデータタイプに基づいてそれを介して検索されるコンポーネントレジストリをさらに備えることを特徴とする請求項 2 に記載のシステム。

## 【請求項 4】

前記出力コンポーネントは、関連するコンポーネント特性をデータコンテキスト要素にバインドして前記出力コンポーネントの子コンポーネントである前記ベースコンポーネントをリンク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 5】

前記構成エンジンは、出力コンポーネント間のデータ交換を可能にするグローバル変数を含むことを特徴とする請求項 1 に記載のシステム。

10

## 【請求項 6】

プロセッサおよびメモリを介して実行可能なコンピュータで実施される視覚化方法であって、

特定のプラットフォームに関する実行環境において利用されるべきコンポーネントを求める要求を受信するステップと、

前記コンポーネントに関連するコンポーネント定義を検索するステップと、

見つけられたコンポーネント定義のための 1 つまたは複数のデータ定義を選択するステップと、および

前記実行環境において前記コンポーネントを出力するために前記コンポーネント定義で前記 1 つまたは複数のデータ定義を自動的に構成するステップと、を備え、

20

前記コンポーネント定義は、ベースコンポーネントおよびコンテナコンポーネントの構成のための定義を含み、

前記コンテナコンポーネントはベースコンポーネントのコンテナであり、プラットフォーム非依存であり、

前記ベースコンポーネントは、前記特定のプラットフォームに向けて実装されており、

前記コンポーネントは前記ベースコンポーネントおよび前記コンテナコンポーネントの組み合わせを含み、特定の視覚化ホストのユーザエクスペリエンスに対応しグローバル変数が適用されていることを特徴とする方法。

## 【請求項 7】

前記コンポーネント定義が見つからないときは前記要求されたコンポーネントのデータタイプに基づいて前記コンポーネント定義を検索するステップをさらに含むことを特徴とする請求項 6 に記載の方法。

30

## 【請求項 8】

前記要求されたコンポーネントの欠如に基づいてカスタムコンポーネントを作成するステップをさらに含むことを特徴とする請求項 6 に記載の方法。

## 【請求項 9】

グローバル変数をプライベート変数でオーバーライドして前記プライベート変数を前記コンポーネントの子コンポーネントに課すステップをさらに含み、前記コンポーネントの子コンポーネントはグローバル変数の代わりにプライベート変数を使用することを特徴とする請求項 6 に記載の方法。

40

## 【請求項 10】

前記要求されたコンポーネントが見つからないときは前記コンテナコンポーネントを作成するステップと、

前記コンテナコンポーネントに前記ベースコンポーネントに関連するデータタイプの特性をロードするステップと、

前記コンポーネントとして前記コンテナコンポーネントを出力するステップとをさらに含むことを特徴とする請求項 6 に記載の方法。

## 【請求項 11】

プロセッサおよびメモリを介して実行可能なコンピュータで実施される視覚化方法であって、

50

特定のプラットフォームに関する実行環境において利用されるコンポーネントを求める要求を受信するステップと、

前記コンポーネントに関連するコンポーネント定義を検索するステップと、

コンポーネント定義が見つけられた場合に、前記コンポーネント定義のための1つまたは複数のデータ定義を選択するステップと、

前記コンポーネント定義が見つからないときは、前記要求されたコンポーネントに関連するデータタイプに基づいてカスタムコンポーネントを作成するステップと、

データコンテキスト間のデータ交換を可能にするためにグローバル変数を前記コンポーネント又はカスタムに適用するステップと、

前記実行環境において前記コンポーネントを出力するために前記コンポーネント定義で前記1つまたは複数のデータ定義を自動的に構成するステップと、を備え、

前記コンポーネント定義は、ベースコンポーネントおよびコンテナコンポーネントの構成ための定義を含み、

前記コンテナコンポーネントはベースコンポーネントのコンテナであり、プラットフォーム非依存であり、

前記ベースコンポーネントは、前記特定のプラットフォームに向けて実装されており、

前記コンポーネントは前記ベースコンポーネントおよび前記コンテナコンポーネントの組み合わせを含み、前記グローバル変数が適用されている

ことを特徴とする方法。

#### 【請求項12】

グローバル変数をプライベート変数でオーバーライドして前記プライベート変数を前記コンポーネントの子コンポーネントに課すステップをさらに含み、前記コンポーネントの子コンポーネントは前記グローバル変数の代わりに前記プライベート変数を使用することを特徴とする請求項11に記載の方法。

#### 【請求項13】

前記要求されたコンポーネントが見つからないときは前記コンテナコンポーネントを作成するステップと、

前記コンテナコンポーネントに前記ベースコンポーネントに関連するデータタイプ特性をロードするステップと、

前記コンポーネントとして前記コンテナコンポーネントを出力するステップとをさらに含むことを特徴とする請求項11に記載の方法。

#### 【請求項14】

パラメータノードを介して前記コンポーネントに渡されるべきデータを定義するステップをさらに含むことを特徴とする請求項11に記載の方法。

#### 【請求項15】

特性の子コンポーネントである前記ベースコンポーネントへのバインディングを含む親コンポーネントを構成するステップをさらに含むことを特徴とする請求項11に記載の方法。

#### 【請求項16】

前記コンポーネントはユーザエクスペリエンスに対応する出力コンポーネントにより表示されるデータのタイプであるターゲットデータタイプに基づいて構成されることを特徴とする、請求項11に記載の方法。

#### 【発明の詳細な説明】

#### 【技術分野】

#### 【0001】

プラットフォーム非依存プレゼンテーションの構成に関する。

#### 【背景技術】

#### 【0002】

ユーザエクスペリエンス(UX)の質は、UXがユーザの期待にどれほどよく合わせられるかに基づく。多くのデータタイプ、多くのデータソース、および多くのUXプラットフォーム

10

20

30

40

50

フォームを扱わなければならないので、設計者は、特定のUXプラットフォームのためのデータソースから特定のデータを消費する特定の人に関するプレゼンテーションコードを書くこと、または、いかなる1人の人のニーズをも満たさない広範囲に向けられたUXを提供することを含む魅力的でない手法から選択しなければならない。

#### 【0003】

例えば、HTML（ハイパーテキストマークアップ言語）、XAML（拡張可能アプリケーションマークアップ言語）、およびXSLT（拡張可能スタイルシート言語変換）などの既存のUX構成システムは、マークアップコードが特定のプラットフォームのために開発されるように設計されている。開発者がコードをいくつかのプラットフォーム上で使いたい場合は、プラットフォームの違いを処理するためにカスタムロジックがコードに組み込まなければならない。さらに、既存のUX構成システムは、特定のプレゼンテーションがデータインターフェース要素ごとに明示的に定義されることを必要とする。要素が表す基本的なデータ構造に基づくUX要素の動的生成を可能にする機能は、特にデータ構造が複雑で、および/または継承可能である場合は、非実在物に限定される。

#### 【0004】

これらの弱点の結果として、マスマーケット（例えば、電子メール）はサービスされてきたかもしれないが、より小さいコミュニティのユーザ（例えば、exchange管理者またはCRM（顧客関係管理）サービスオーナー）は十分にサービスされていない。

#### 【発明の概要】

#### 【0005】

下記は、本明細書に記載のいくつかの新規の実施形態の基本的な理解を提供するために、簡略化された概要を提示する。この概要は、広範囲にわたる概観ではなく、キー/クリティカル要素を識別することまたはそれらの範囲を概説することを意図するものではない。その唯一の目的は、以下で提示されるより詳細な説明の序として簡略化された形でいくつかの概念を提示することである。

#### 【0006】

開示されたアーキテクチャは、プラットフォーム非依存構成駆動プレゼンテーション構成エンジンを含む。構成エンジンは、データコントラクトに基づくマルチプラットフォームユーザエクスペリエンス（UX）の動的生成を可能にする。構成によって、ユーザは、部分、相互作用、および相互作用と部分との間の制約、ならびに相互に関する配置を選択することができる。

#### 【0007】

UXは、特定のデータクラスに向けられたコンポーネントから動的に構成される。実行時に、プラットフォーム依存コンポーネントの実装は、構成ホストの実行プラットフォームに基づいてエンジンによって自動的に選択される。

#### 【0008】

開示されたアーキテクチャは、ユーザが、多くのプラットフォーム上で機能する多くのデータソースにアクセスすることができる複数のプレゼンテーションウィジェットから構成することによってコードを書かずにUXを作成するまたはカスタマイズすることができるようにする。構成は、データクラスおよびプレゼンテーションタイプの両方に向けられ、予め定義されてもよく、または生成されてもよい。

#### 【0009】

前述のおよび関連する目的の達成のために、いくつかの例示的態様が以下の説明および添付の図面に関連して本明細書で説明される。これらの態様は、本明細書で開示される原理が実行され得る様々なやり方を示し、したがって、すべての態様およびそれらの同等物は、本特許請求の主題の範囲内にあるものとする。他の利点および新規の特徴は、図面を参照しながら考察された場合、以下の詳細な説明から明らかになるであろう。

#### 【図面の簡単な説明】

#### 【0010】

【図1】開示されたアーキテクチャによる視覚化システムを例示する図である。

10

20

30

40

50

【図 2】開示されたアーキテクチャによる代替視覚化システムを例示する図である。

【図 3】構成エンジンによって構成された例示的構成を例示する図である。

【図 4】構成システムのデータテキストおよびビジュアルベースコンポーネントを含む親コンポーネントを例示する図である。

【図 5】コンポーネント定義を例示する図である。

【図 6】コンポーネントを見つけるまたは選択するためのコンポーネントレジストリを例示する図である。

【図 7】構成エンジン内の変数の使用を表す宣言図を例示する図である。

【図 8】開示されたアーキテクチャによる視覚化方法を例示する図である。

【図 9】図 8 の方法のさらなる態様を例示する図である。

10

【図 10】代替視覚化方法を例示する図である。

【図 11】図 10 の方法のさらなる態様を例示する図である。

【図 12】構成エンジンにおいてコンポーネントを取得する方法を例示する図である。

【図 13】構成エンジンにおいてコンポーネントを取得するさらに詳細な方法を例示する図である。

【図 14】開示されたアーキテクチャによる構成を実行するコンピューティングシステムのブロック図である。

【発明を実施するための形態】

【0011】

開示されたアーキテクチャは、プレゼンテーション構成エンジンである。構成エンジンは、ユーザが様々なコンポーネントを相互に「接合する」（構成する）ことができるようにするサービスのセットとして示される汎用構成フレームワーク、およびコンポーネント（1つまたは複数）の構成（エンジンの出力構成）である。構成によって、ユーザは、部分、相互作用、および相互作用と部分との間の制約、ならびに相互に関する部分の配置を選択することができる。エンジンは、UI（ユーザインターフェース）および非UIコンポーネントの両方のためのプレゼンテーションニュートラルフレームワークである。

20

【0012】

コンポーネントは、構成エンジンのためのUI宣言の最小再利用可能構築ブロックであり、名称によって識別可能であり、任意選択でデータタイプに向けられる。コンポーネントは、ベースコンポーネント（ユニットコンポーネント）でも、またはコンテナコンポーネント（合成コンポーネント）でもよい。データコンテキストは、コンポーネントのためのターゲットデータのインスタンスである。言い換えれば、データコンテキストは、コンポーネントに関連するデータを表す名称/値のペアセットである。データコンテキストは、サポート変更通知を入力し、構成は、変更を開始する、および/または他の構成によって開始された変更をリスンすることができる。構成エンジンは、プラットフォーム非依存であるユーザエクスペリエンスとしての特定のホストのためのコンポーネントをアセンブルする。視覚化ホストは、特定のプラットフォーム（実行時）のための構成の実行環境である。

30

【0013】

構成は、データクラスおよびプレゼンテーションタイプの両方に向けられ、予め定義されていてもよく、または生成されてもよい。複数のコンポーネントが構成されてもよいが、コンポーネントチェーンは、具体的なベースコンポーネント（例えば、Text Box コントロール、データベースクエリコンポーネントなど）において終わる。

40

【0014】

構成エンジンは、データコントラクトに基づいてマルチプラットフォームUX（ユーザエクスペリエンス）の動的生成を可能にする。UXは、特定のデータクラスに向けられるコンポーネントから動的に構成される。実行時に、プラットフォーム依存コンポーネントの実装は、構成ホストの実行プラットフォームに基づいてエンジンによって自動的に選択される。

【0015】

50

次に、全体にわたって同様の参照符号は同様の要素を示すために使用される図面が参照される。以下の説明では、説明の目的で多数の特定の詳細がそれらの徹底的な理解を提供するために記述される。しかし、新規の実施形態はこれらの特定の詳細なしで実行されることが可能であることが明らかであろう。他の場合には、よく知られている構造およびデバイスがそれらの説明を容易にするためにブロック図の形で示され。この意図は、本特許請求の主題の趣旨および範囲の範囲内に入るすべての変更形態、同等形態、および代替形態をカバーすることである。

#### 【 0 0 1 6 】

図 1 は、開示されたアーキテクチャによる視覚化システム 1 0 0 を例示する。システム 1 0 0 は、ユーザエクスペリエンスに関連するコンポーネントおよびデータのためのコンポーネント定義 1 0 4 およびデータ定義 1 0 6 を含むストア定義のストア 1 0 2 を含む。コンポーネント定義 1 0 4 は、ベースコンポーネント、コンテナコンポーネント、ならびにベースコンポーネントおよびコンテナコンポーネントの構成のための定義を含んでよい。このようにして、コンポーネントの既存の構成は、動的選択および出力コンポーネント 1 1 0 への構成のために容易に利用可能である。

#### 【 0 0 1 7 】

構成エンジン 1 0 8 はストア定義に基づいて出力コンポーネント 1 1 0 のインスタンスを自動的にかつ宣言的に構成する。出力コンポーネントは、様々なホスト 1 1 2 の視覚化ホストのユーザエクスペリエンスに特有である。

#### 【 0 0 1 8 】

出力コンポーネント 1 1 0 は、ベースコンポーネント、コンテナコンポーネント、またはベースコンポーネントおよびコンテナコンポーネントの組合せを含む。出力コンポーネント 1 1 0 は、ユーザエクスペリエンスのターゲットデータタイプに基づいて構成される。システム 1 0 0 は、コンポーネントがターゲットデータタイプに基づいてそれを介して検索されるコンポーネントレジストリをさらに備えてよい。出力コンポーネントは、関連コンポーネント特性をデータコンテキスト要素にバインドして子コンポーネントをリンクする。構成エンジン 1 0 8 は、関連のないデータコンテキストにおける出力コンポーネント間のデータ交換を可能にするグローバル変数を含む。

#### 【 0 0 1 9 】

図 2 は、開示されたアーキテクチャによる代替可視化システム 2 0 0 を例示する。システム 2 0 0 は、図 1 のシステム 1 0 0 のエンティティ、ならびにデータコンテキスト 2 0 2、個人化（プライベート）オーバーライド 2 0 4、およびコンポーネントの実装 2 0 6 を含む。出力コンポーネント 1 1 0 は、既存のコンポーネント定義ではなくデータコンテキスト 2 0 2 に基づいて構成されてよい。すなわち、データに基づいて、カスタマイズされたコンポーネントが作成され、純粹にコンテキストデータ 2 0 2（ターゲット UX 内のデータのインスタンス）に基づいて出力されてよい。構成エンジン 1 0 8 は、選択されたコンポーネント定義で構成されるパーソナライズド・オーバーライド 2 0 4 を利用してグローバル変数をプライベート変数でオーバーライドする。

#### 【 0 0 2 0 】

図 3 は、構成エンジンによって構成された例示的構成 3 0 0 を例示する。ここでは、構成 3 0 0 が、ベースコンポーネント（例えば、Stack Panel ベースコンポーネント 3 0 2）およびコンテナコンポーネント 3 0 4 の点から説明される。ここでは、ベースコンポーネント 3 0 2 は、2つのテキストボックスベースコンポーネントを含み、第 1 のテキストボックスベースコンポーネントはテキスト「ABC」を示し、第 2 のテキストボックスベースコンポーネントはテキスト「DEF」を示す。ベースコンポーネント 3 0 2 はまた、ボタンベースコンポーネントを含む。

#### 【 0 0 2 1 】

ベースコンポーネントは、特定のプラットフォームに向けられた具体的な実装、構成プロセスの葉ノードであり、ビジュアルでも非ビジュアルでもよい。下記は、（コンポーネントタイプではなくコンポーネントタイプの点からの）ベースコンポーネント定義の一例

10

20

30

40

50

である。

【 0 0 2 2 】

【表 1】

```

    <ComponentType ID="EventQuery">
      <Parameters>
        <Parameter Name="Scope" Type="String" />
        <Parameter Name="Output" Type="IEnumerable" />
      </Parameters>
    </ComponentType>
    <ComponentImplementation TypeId="EventQuery">
      <SupportedPlatforms>
        <Platform>WPF</Platform>
      </SupportedPlatforms>
      <Unit>
        <MefFactory>

<ContractName>Company.EnterpriseManagement.EventQuery</ContractName>
      </MefFactory>
      <Properties>
        <Property Name="QueryVerb" Direction="In">Events</Property>
        <Property Name="Scope"
Direction="In">$Parameter/Scope$</Property>
        <Property Name="Output"
Direction="Out">$Parameter/Output$</Property>
      </Properties>
    </Unit>
  </ComponentImplementation>

```

【 0 0 2 3 】

MEF (managed extensibility framework) ファクトリは、MEF 実行時を呼び出して登録された UI アセンブリから対応するタイプを引き出す。MEF は、1 つの例示的実装方式にすぎず、他のファクトリ実装が利用されてもよいことに留意されたい。

【 0 0 2 4 】

コンテナコンポーネント（さらに合成コンポーネント）は、ベースコンポーネント（さらにユニットコンポーネント）のコンテナであり、カスタム実装を有さない、プラットフォーム非依存である。下記は、合成コンポーネント定義の一例である。

【 0 0 2 5 】

【表 2】

```

<ComponentImplementation TypeId="SampleComponent">
  <SupportedPlatforms>
    <Platform>All</Platform>
  </SupportedPlatforms>
  <Composite>
    <Variables>
      <Variable Id="abc" Type="String" />
    </Variables>
    <Component TypeId="SampleContainerComponent">
      <Parameter Id="Child1">
        <Component TypeId="SampleComponent1">
          <Parameter Id="Bla">$Variable/abc$</Parameter>
        </Component>
      </Parameter>
      <Parameter Id="Child2">
        <Component TypeId="SampleComponent2">
          <Parameter Id="Bla">$Variable/abc$</Parameter>
        </Component>
      </Parameter>
    </Component>
  </Composite>
</ComponentImplementation>

```

## 【 0 0 2 6 】

図3では、コンテナコンポーネント304が、以下のコードにおいて例示されているように、ベースコンポーネントを結合し、ここで、Property Aは「ABC」であり、Property Bは「DEF」である。

## 【 0 0 2 7 】

## 【表3】

```
<Component TypeId="StackPanel">
  <Parameter Name="Child">
    <Component TypeId="Edit">
      <Target>$Target/propertyA$</Target>
    </Component>
    <Component TypeId="Edit">
      <Target>$Target/propertyB$</Target>
    </Component>
    <Component TypeId="Button">
    </Component>
  </Parameter>
```

10

## 【 0 0 2 8 】

各コンポーネント（ベースまたはコンテナ）は、データコンテキスト（Data Context）インスタンスでバックアップされる。Data Contextは、特性変更通知およびエラーセット通知をサポートするキー（ストリング）と値（オブジェクト）のペアの集まりである。コンポーネントは、その特性をデータコンテキスト要素にバインドして子コンポーネントを相互にリンクする。

20

## 【 0 0 2 9 】

図4は、構成システムのデータコンテキスト402（DataContext要素として）およびビジュアルベースコンポーネント404（例えば、図3のStackPanelベースコンポーネント302）を含む親コンポーネント400を例示する。ここでは、データコンテキスト402は、3つの特性、すなわち、それぞれ対応する値「ABC」、「DEF」、および「XYZ」を有するProperty A、Property B、およびProperty Cを含む。ビジュアルベースコンポーネント404は、対応する双方向データバインドを介してのProperty AおよびProperty Bへのバインディングを含むが、ビューモデル408の複数のデータコンポーネント406は、Property BおよびProperty Cにバインドされる。言い換えれば、特性の子コンポーネント（テキストボックスベースコンポーネント）へのバインディングおよび特性のデータコンポーネント406へのバインディングを含む親コンポーネント400が構成される。

30

## 【 0 0 3 0 】

図5は、コンポーネント定義500を例示する。コンポーネント定義500は、2つの部分、すなわち、タイプ宣言502および実装定義504を備える。各コンポーネントは、名称、および関連コンポーネントを捜し出すために使用され得るオプションターゲットタイプ属性を有する。例示的タイプ宣言（コンポーネントに関して）は、以下の通りである。

40

## 【 0 0 3 1 】

## 【表4】

```
<ComponentType ID="SampleComponent" Target="String"
Accessibility="Internal">
```

## 【 0 0 3 2 】

コンポーネントはまた、以下に例示されているように（コンポーネントに関して）、渡されると予想されるデータ形状（例えば、ストリング）を定義するParameters

50



サブノードを含んでよい。

【 0 0 3 3 】

【表 5】

```
<ComponentType ID="AnotherComposition">
  <Parameters>
    <Parameter Name="Parameter1" Type="String" />
    <Parameter Name="SelectedText" Type="String" BindingDirection="Both"
  />
</Parameters>
```

【 0 0 3 4 】

10

図 6 は、コンポーネントを見つけるまたは選択するためのコンポーネントレジストリ 600 を例示する。レジストリ 600 は、すべての定義されたコンポーネントおよびコンポーネント構成のリストを維持する。例えば、ベースコンポーネント（例えば、図 3 の `StackPanel/Button/Edit`）は、`TypeId` および `TargetType`（ターゲットデータタイプ）に基づいて探し出される（検索される）ことが可能である。この場合、出力は、`TargetType` に対応するベースコンポーネント、または `TypeId` および `TargetType` である。

【 0 0 3 5 】

コンポーネントに関する特性を設定するために、以下のサンプルコードに示されているように、「`Parameter`」ノードが使用される。

20

【 0 0 3 6 】

【表 6】

```
<Component Id="EventView">
<Parameter Id="Scope">Microsoft.SystemCenter.SqlDB </Parameter>
```

【 0 0 3 7 】

この場合、コンポーネント「`EventView`」の特性「`Scope`」は、テキスト「`Company.SystemCenter.SqlDB`」に設定される。しかし、しばしば、パラメータは静的ではなく、他の要素にバインドされる。一般に、リファレンスは、

【 0 0 3 8 】

30

【表 7】

```
$<protocol>/<protocol-specific string>.
```

【 0 0 3 9 】

の形である。

【 0 0 4 0 】

例えば、2つのコンポーネントは、変数「`abc`」にバインドされる。

【 0 0 4 1 】

## 【表 8】

```

<ComponentImplementation TypeId="SampleComponent">
  <SupportedPlatforms>
    <Platform>All</Platform>
  </SupportedPlatforms>
  <Composite>
    <Variables>
      <Variable Id="abc" Type="String" />
    </Variables>
    <Component TypeId="SampleComponent1">
      <Parameter Id="A">$Variable/abc$</Parameter>
    </Component>
    <Component TypeId="SampleComponent2">
      <Parameter Id="A">$Variable/abc$</Parameter>
    </Component>
  </Composite>
</ComponentImplementation>

```

10

## 【 0 0 4 2 】

下記は P a r a m e t e r ノードにおけるリファレンスプロトコルの例示的リストである。

## 【 0 0 4 3 】

20

## 【表 9】

\$Parameter/<propertyName>\$	コンポーネントに渡されるパラメータ
\$Variable/<propertyName>\$	コンポーネントにおいて宣言される変数
\$Target/<propertyName>\$	コンポーネントに渡されるターゲット
インスタンスの特性	
\$Target\$	ターゲットインスタンス

## 【 0 0 4 4 】

グローバル変数は、関連のないデータコンテキストにおける構成（ベースコンポーネントおよび／またはコンテナコンポーネント）間のデータ交換を可能にするために使用され得る。最初に、グローバル変数が宣言される。

30

## 【 0 0 4 5 】

## 【表 10】

```

<GlobalVariable ID="GlobalSelectedItem" Type="String"/>

```

## 【 0 0 4 6 】

変数は、

## 【 0 0 4 7 】

## 【表 11】

```

$GlobalVariable/<variable name>$.

```

## 【 0 0 4 8 】

40

を使用して、ベースコンポーネントおよびコンテナコンポーネントにおいて参照され得る。

## 【 0 0 4 9 】

任意の所与のコンポーネント（例えば、ベース、コンテナ）は、コンポーネントの子が、以下のサンプルコードに例示されている変数のプライベートコピーを見るように、変数をプライベート実装でオーバーライドすることができる。

## 【 0 0 5 0 】

## 【表 1 2】

```
<Variables>
  <GlobalVariableOverride GlobalVariableId="GlobalSelectedItem" />
</Variables>
```

## 【0051】

図7は、構成エンジンにおける変数の使用を表す宣言図700を例示する。702において、グローバル変数「A」が宣言される。704において、パラメータ名「Blah」がコンポーネントに渡される。706において、グローバル変数のローカルコピーが作成される。708および710において、グローバル変数ではなく、変数のローカルコピーが使用される。

10

## 【0052】

本明細書には、開示されたアーキテクチャの新規の態様を実行するための例示的方法を表すフローチャートのセットが含まれている。説明を簡単にするために、例えばフローチャートまたは流れ図の形で本明細書に示されている1つまたは複数の方法は、一連の動作として示され説明されているが、これらの方法は、いくつかの動作はそれらの方法によって本明細書に示され説明されている順序とは異なる順序でおよび/または他の動作と同時に進行されてもよいので、動作の順序によって限定されないことが理解され認識されるべきである。あるいは、例えば、方法は、状態図などにおいて一連の相互関係のある状態またはイベントとして表されてもよいことを当業者は理解し認識するであろう。さらに、必ずしも方法において例示されているすべての動作が新規の実装のために必要とされるわけではない。

20

## 【0053】

図8は、開示されたアーキテクチャによる視覚化方法を例示する。800において、実行環境において利用されるべきコンポーネントを求める要求が受信される。802において、コンポーネントに関連するコンポーネント定義が検索される。804において、1つまたは複数のデータ定義が、見つけられたコンポーネント定義のために選択される。806において、1つまたは複数のデータ定義がコンポーネント定義によって自動的に構成されて環境実行時に実行環境においてコンポーネントを出力する。

## 【0054】

図9は、図8の方法のさらなる態様を例示する。900において、コンポーネント定義が見つからないときは、コンポーネント定義は要求されたコンポーネントのデータタイプに基づいて検索される。902において、カスタムコンポーネントが要求されたコンポーネントの欠如に基づいて作成される。904において、グローバル変数がコンポーネントに適用されて関連のないデータコンテキスト間のデータ交換を可能にする。906において、グローバル変数がプライベート変数でオーバーライドされてプライベート変数をコンポーネントの子コンポーネントに課す。908において、要求されたコンポーネントが見つからないときは、コンテナコンポーネントが作成される。910において、コンテナコンポーネントがベースコンポーネント関連データタイプ特性をロードされる。912において、コンテナコンポーネントがコンポーネントとして出力される。

30

## 【0055】

図10は、代替視覚化方法を例示する。1000において、コンポーネント実行環境に基づいて、コンポーネントを求める要求が受信される。1002において、コンポーネントに関連するコンポーネント定義が検索される。1004において、コンポーネント定義が見つかった場合は、コンポーネント定義のための1つまたは複数のデータ定義が選択される。1006において、コンポーネント定義が見つからないときは、要求されたコンポーネントに関連するデータタイプに基づいてカスタムコンポーネントが作成される。1008において、グローバル変数がコンポーネントまたはカスタムに適用されて関連のないデータコンテキスト間のデータ交換を可能にする。1010において、1つまたは複数のデータ定義がコンポーネント定義によって自動的に構成されて環境実行時に実行環境においてコンポーネントを出力する。

40

50

## 【 0 0 5 6 】

図 1 1 は、図 1 0 の方法のさらなる態様を例示する。1 1 0 0 において、グローバル変数がプライベート変数でオーバーライドされてコンポーネントの子コンポーネントにプライベート変数を課す。1 1 0 2 において、要求されたコンポーネントが見つからないときは、コンテナコンポーネントが作成される。1 1 0 4 において、コンテナコンポーネントがベースコンポーネント関連データタイプ特性をロードされる。1 1 0 6 において、コンテナコンポーネントがコンポーネントとして出力される。1 1 0 8 において、コンポーネントに渡されるべきデータがパラメータノードを介して定義される。1 1 1 0 において、特性の子コンポーネントへのパインディングおよび特性のデータコンポーネントへのパインディングを含む親コンポーネントが構成される。

10

## 【 0 0 5 7 】

図 1 2 は、構成エンジンにおいてコンポーネントを取得する方法を例示する。1 2 0 0 において、コンポーネントが取得される（例えば、「コンポーネントを取得する」コールを介して）。1 2 0 2 において、ターゲットデータタイプが取得される。1 2 0 4 において、ターゲットデータタイプのためのコンポーネントが存在するかどうかを判定するためにチェックが行われる。存在しない場合は、フローは 1 2 0 6 に進み、コンポーネントコンテナを作成する。コンポーネントは、1 つまたは複数の関連データ特性を有してよい。1 2 0 8 において、コンポーネントのための特性タイプが取得される。1 2 1 0 において、特性タイプのためのコール「コンポーネントを取得する」が行われる。1 2 1 2 において、コンポーネントがコンテナに付加される。フローは 1 2 0 8 に戻り、完了するまで続く。すべてのデータ特性およびタイプがコンテナに適用された後に、フローは、1 2 1 4 に進み、結果を返す。1 2 0 4 において、チェックがターゲットデータタイプのためのコンポーネントが存在すると判定した場合は、フローは 1 2 1 6 に進み、コンポーネントを選択し、次いで、1 2 1 6 において結果を返す。

20

## 【 0 0 5 8 】

図 1 3 は、構成エンジンにおいてコンポーネントを取得するさらに詳細な方法を例示する。1 3 0 0 において、ターゲットタイプおよびデータタイプなどの機能パラメータが受信される。1 3 0 2 において、ターゲットタイプおよびターゲット名のために定義されたコンポーネントに関してチェックが行われる。そうである場合は、フローは 1 3 0 4 に進み、インターフェースを検証する。あるいは、ターゲットタイプおよびターゲット名のためのコンポーネントが存在しない場合は、フローは 1 3 0 6 に進み、その結果、タイプシステムルックアップを使用して特性ごとにデータタイプのためのコンポーネントが定義される。そうである場合は、フローは 1 3 0 4 に進み、インターフェースを検証する。チェックを行ってインターフェースを検証するためにタイプシステム 1 3 0 8 へのアクセスが提供される。インターフェースが検証された後は、フローは 1 3 1 0 に進み、コンポーネントタイプをチェックする。ユニットコンポーネントであれば、フローは 1 3 1 2 に進み、正しいプラットフォームのためのユニットコンポーネントのインスタンスを作成し、すべての宣言されたパラメータをユニットコンポーネントに渡す。1 3 1 4 において、ローダが U X 構成システム（例えば、X A M L、M E F など）のためのアセンブリをロードすることができる。

30

40

## 【 0 0 5 9 】

1 3 1 0 において、コンポーネントタイプが合成コンポーネントである場合は、フローは 1 3 1 6 に進み、構成のコンポーネントに関する値を設定して（例えば、X M L で書かれた）構成において子ノードをウォークする。これは、パラメータノード（1 つまたは複数）1 3 1 8 からのパラメータ値、コンポーネントノード（1 つまたは複数）1 3 2 0 からのパラメータ、および 1 3 2 2 の「パラメータからデータを構築する」から提供される子ノードからのパラメータセットを受信することを含む。タイプ ID は、1 3 2 2 においてデータを構築するためにコンポーネントノード 1 3 2 0 から送信される。1 3 2 4 においては、コンポーネントノード 1 3 2 0 から受信された名称およびターゲット情報に基づいて、データおよび名称として、ルックアップのための名称として、ターゲットによって

50

参照されるコンポーネント。

【 0 0 6 0 】

1つまたは複数のコンポーネントが実行のプロセスおよび/またはスレッドの中に存在してよく、コンポーネントは、1つのコンピュータ上にローカライズされてもよく、および/または2つ以上のコンピュータ間に分散されてもよい。用語「例示的」は、本明細書では、実施例、例、または例示として役立つことを意味するために使用されてよい。本明細書で「例示的」として示されているいかなる態様または設計も、必ずしも他の態様または設計より好ましいまたは有利であると解釈されるわけではない。

【 0 0 6 1 】

次に、図14を参照すると、開示されたアーキテクチャによる構成を実行するコンピューティングシステム1400のブロック図が例示されている。コンピューティングシステム1400の様々な態様のための追加のコンテキストを提供するために、図14および以下の説明は、様々な態様がそこで実施され得る適切なコンピューティングシステム1400の簡単な全般的説明を提供することを意図するものである。上記の説明は1つまたは複数のコンピュータ上で実行することができるコンピュータ実行可能命令の一般的なコンテキストにおいてであるが、新規の実施形態はまた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およびソフトウェアの組合せとして実施され得ることを当業者は理解するであろう。

【 0 0 6 2 】

様々な態様を実施するためのコンピューティングシステム1400は、処理ユニット(1つまたは複数)1404、システムメモリ1406などのコンピュータ可読ストレージ、およびシステムバス1408を有するコンピュータ1402を含む。処理ユニット(1つまたは複数)1404は、シングルプロセッサ、マルチプロセッサ、シングルコアユニットおよびマルチコアユニットなど、様々な市販のプロセッサのいずれでもよい。さらに、新規の方法は、それぞれが1つまたは複数の関連デバイスに動作可能に結合され得るミニコンピュータ、メインフレームコンピュータ、ならびにパーソナルコンピュータ(例えば、デスクトップ、ラップトップなど)、ハンドヘルドコンピューティングデバイス、マイクロプロセッサベースのまたはプログラマブルなコンシューマ電子機器、および同種のものを含めて、他のコンピュータシステム構成によって実行され得ることを当業者は理解するであろう。

【 0 0 6 3 】

システムメモリ1406は、揮発性(VOL)メモリ1410(例えば、ランダムアクセスメモリ(RAM))および不揮発性メモリ(NON-VOL)1412(例えば、ROM、EPROM、EEPROMなど)などのコンピュータ可読ストレージ(物理的記憶媒体)を含んでよい。基本入出力システム(BIOS)は、不揮発性メモリ1412に記憶されることが可能であり、起動中などにコンピュータ1402の中のコンポーネント間のデータおよび信号の通信を容易にする基本ルーチンを含む。揮発性メモリ1410はまた、データをキャッシュするためのスタティックRAMなどの高速RAMを含んでよい。

【 0 0 6 4 】

システムバス1408は、処理ユニット(1つまたは複数)1404へのシステムメモリ1406を含むがそれに限定されないシステムコンポーネントのためのインターフェースを提供する。システムバス1408は、任意の様々な市販のバスアーキテクチャを使用して、(メモリコントローラを有するまたは有しない)メモリバス、および周辺バス(例えば、PCI、PCIe、AGP、LPCなど)にさらに相互接続することができるいくつかのタイプのバス構造体のいずれでもよい。

【 0 0 6 5 】

コンピュータ1402は、機械可読ストレージサブシステム(1つまたは複数)1414およびストレージサブシステム(1つまたは複数)1414をシステムバス1408および他の所望のコンピュータコンポーネントにインターフェースするためのストレージインターフェース(1つまたは複数)1416をさらに含む。ストレージサブシステム(1

10

20

30

40

50

つまたは複数) 1 4 1 4 (物理的記憶媒体) は、例えば、ハードディスクドライブ (HDD)、磁気フロッピーディスクドライブ (FDD)、および/または光ディスクストレージドライブ (例えば、CD-ROMドライブDVDドライブ) のうちの1つまたは複数を含んでよい。ストレージインターフェース (1つまたは複数) 1 4 1 6 は、例えば、EIDE、ATA、SATA、およびIEEE 1394などのインターフェース技術を含んでよい。

【0066】

1つまたは複数のプログラムおよびデータは、オペレーティングシステム 1 4 2 0、1つまたは複数のアプリケーションプログラム 1 4 2 2、他のプログラムモジュール 1 4 2 4、およびプログラムデータ 1 4 2 6を含めて、メモリサブシステム 1 4 0 6、機械可読および取外し可能メモリサブシステム 1 4 1 8 (例えば、フラッシュドライブフォームファクタ技術)、および/またはストレージサブシステム (1つまたは複数) 1 4 1 4 (例えば、光、磁気、ソリッドステート) に記憶され得る。

10

【0067】

1つまたは複数のアプリケーションプログラム 1 4 2 2、他のプログラムモジュール 1 4 2 4、およびプログラムデータ 1 4 2 6は、例えば、図1のシステム 1 0 0のエンティティおよびコンポーネント、図2のシステム 2 0 0のエンティティおよびコンポーネント、図3の構成 3 0 0、図4の親コンポーネント 4 0 0、図5のコンポーネント定義 5 0 0、図6のレジストリ 6 0 0、図7の図 7 0 0、および図8 ~ 13のフローチャートによって表された方法を含んでよい。

20

【0068】

一般に、プログラムは、特定のタスクを実行するまたは特定の抽象データタイプを実施するルーチン、方法、データ構造、他のソフトウェアコンポーネントなどを含む。オペレーティングシステム 1 4 2 0、アプリケーション 1 4 2 2、モジュール 1 4 2 4、および/またはデータ 1 4 2 6のすべてまたは一部分はまた、例えば、揮発性メモリ 1 4 1 0などのメモリにキャッシュされることも可能である。開示されたアーキテクチャは、様々な市販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 (例えば、仮想マシン) の組合せによって実施されてよいことが理解されるべきである。

【0069】

ストレージサブシステム (1つまたは複数) 1 4 1 4およびメモリサブシステム (1 4 0 6および1 4 1 8) は、データ、データ構造、コンピュータ実行可能命令などの揮発性ストレージまたは不揮発性ストレージのためのコンピュータ可読媒体として役立つ。そのような命令は、コンピュータまたは他のマシンによって実行される場合、コンピュータまたは他のマシンに方法の1つまたは複数の動作を実行させることができる。動作を実行するための命令は、命令がすべて同じ媒体上にあるかどうかに関係なく、命令が1つまたは複数のコンピュータ可読記憶媒体上にまとめて現れるように、1つの媒体上に記憶されてもよく、または複数の媒体にまたがって記憶されてもよい。

30

【0070】

コンピュータ可読媒体は、コンピュータ 1 4 0 2によってアクセスされることが可能であり、取外し可能もしくは取外し不可能な揮発性および不揮発性内部および/または外部媒体を含む任意の入手可能な媒体でよい。コンピュータ 1 4 0 2では、媒体は、任意の適切なデジタルフォーマットでのデータの記憶に対応する。開示されたアーキテクチャの新規の方法を実行するためのコンピュータ実行可能命令を記憶するためのジップドライブ、磁気テープ、フラッシュメモリカード、フラッシュドライブ、カートリッジ、および同種のものなどの他のタイプのコンピュータ可読媒体が利用され得ることが当業者には理解されるはずである。

40

【0071】

ユーザは、キーボードおよびマウスなどの外部入力デバイス 1 4 2 8を使用して、コンピュータ 1 4 0 2、プログラム、およびデータと対話することができる。他の外部ユーザ入力デバイス 1 4 2 8は、マイクロホン、IR (赤外線) リモートコントロール、ジョイ

50

スティック、ゲームパッド、カメラ認識システム、スタイラスペン、タッチスクリーン、ジェスチャシステム（例えば、目の動き、頭の動きなど）、および／または同種のものを含んでよい。ユーザは、タッチパッド、マイクロホン、キーボードなどのオンボードユーザ入力デバイス 1 4 3 0 を使用して、コンピュータ 1 4 0 2、プログラム、およびデータと対話することができ、コンピュータ 1 4 0 2 は、例えば、ポータブルコンピュータである。これらおよび他の入力デバイスは、入力／出力（I/O）デバイスインターフェース（1つまたは複数）1 4 3 2 を通ってシステムバス 1 4 0 8 を介して処理ユニット（1つまたは複数）1 4 0 4 に接続されるが、パラレルポート、IEEE 1394 シリアルポート、ゲームポート、USB ポート、IR インターフェースなどの他のインターフェースによって接続されてもよい。I/O デバイスインターフェース（1つまたは複数）1 4 3 2 はまた、サウンドカードおよび／またはオンボードオーディオ処理機能など、プリンタ、オーディオデバイス、カメラデバイスなどの出力周辺装置 1 4 3 4 の使用を容易にする。

#### 【0072】

1つまたは複数のグラフィックスインターフェース（1つまたは複数）1 4 3 6（グラフィックス処理ユニット（GPU）とも呼ばれる）は、コンピュータ 1 4 0 2 と外部ディスプレイ（1つまたは複数）1 4 3 8（例えば、LCD、プラズマ）および／またはオンボードディスプレイ 1 4 4 0（例えば、ポータブルコンピュータ用）との間にグラフィックス信号およびビデオ信号を提供する。グラフィックスインターフェース（1つまたは複数）1 4 3 6 はまた、コンピュータシステムボードの一部として製造されることも可能である

#### 【0073】

コンピュータ 1 4 0 2 は、有線／無線通信サブシステム 1 4 4 2 を介しての1つまたは複数のネットワークおよび／または他のコンピュータへの論理接続を使用して（例えば、IP ベースの）ネットワーク化された環境において動作することができる。他のコンピュータは、ワークステーション、サーバ、ルータ、パーソナルコンピュータ、マイクロプロセッサベースのエンタテインメント装置、ピアデバイス、または他の一般的なネットワークノードを含んでよく、通常、コンピュータ 1 4 0 2 に関して記述された要素の多くまたはすべてを含んでよい。論理接続は、ローカルエリアネットワーク（LAN）、広域ネットワーク（WAN）、ホットスポットなどへの有線／無線接続を含んでよい。LAN および WAN ネットワーキング環境は、オフィスおよび会社では普通であり、すべてインターネットなどのグローバル通信ネットワークに接続されることができるイントラネットなどの企業規模でのコンピュータネットワークを容易にする。

#### 【0074】

ネットワーキング環境において使用される場合、コンピュータ 1 4 0 2 は、有線／無線ネットワーク、有線／無線プリンタ、有線／無線入力デバイス 1 4 4 4 などと通信するために、有線／無線通信システム 1 4 4 2（例えば、ネットワークインターフェースアダプタ、オンボードトランシーバサブシステムなど）を介してネットワークに接続される。コンピュータ 1 4 0 2 は、ネットワークを介して通信を確立するためのモデムまたは他の手段を含んでよい。ネットワーク環境では、コンピュータ 1 4 0 2 に関するプログラムおよびデータは、分散システムに関連する場合のように、リモートメモリ／記憶デバイスに記憶されてもよい。図示されているネットワーク接続は例示であり、コンピュータ間の通信リンクを確立する他の手段が利用されてよいことが理解されるであろう。

#### 【0075】

コンピュータ 1 4 0 2 は、例えば、プリンタ、スキャナ、デスクトップコンピュータおよび／またはポータブルコンピュータ、携帯情報端末（PDA）、通信衛星、無線で検出可能なタグに関連する装置の任意の部分または位置（例えば、キオスク、ニューススタンド、レストルーム）、および電話との無線通信（例えば、IEEE 802.11 オーバジエア変調技術）において動作可能に配置された無線デバイスなど、IEEE 802.x ファミリの規格などの無線技術を使用して有線／無線デバイスまたはエンティティと通信するように動作可能である。これは、少なくともホットスポットのための Wi-Fi（ま

たはワイヤレスフィデリティ)、WiMax、およびブルートゥース(商標)無線技術を含む。したがって、通信は、従来のネットワークの場合のように、予め定義された構造または単に少なくとも2つのデバイス間のアドホック通信でよい。Wi-Fiネットワークは、安全で信頼できる高速無線接続を提供するためにIEEE802.11x(a、b、gなど)と呼ばれる無線技術を使用する。Wi-Fiネットワークは、コンピュータを相互に、インターネットに、および(IEEE802.3関係媒体および機能を使用する)有線ネットワークに接続するために使用され得る。

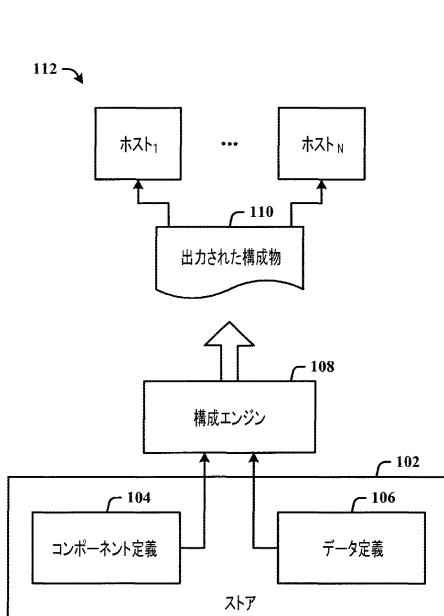
#### 【0076】

例示され説明された諸態様は、いくつかのタスクが、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デバイスによって実行される分散コンピューティング環境において実行されてよい。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、ローカルストレージおよび/またはリモートストレージおよび/またはメモリシステム内に配置されてよい。

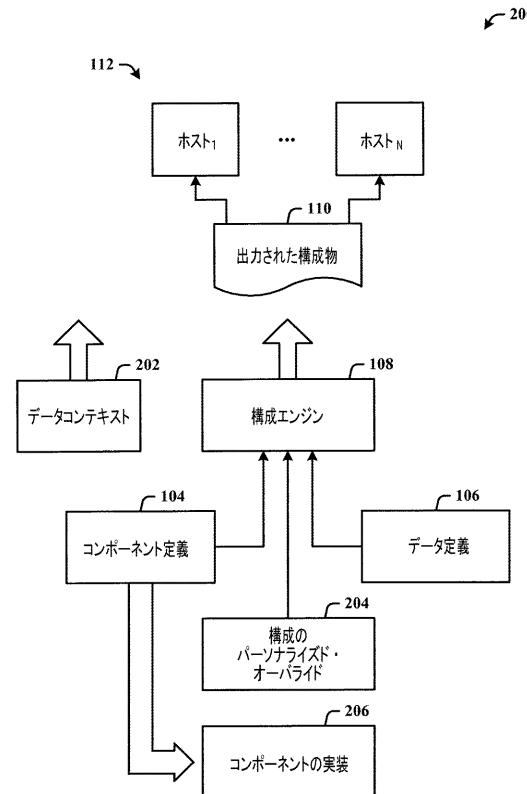
#### 【0077】

上記で説明されてきたものは、開示されたアーキテクチャの実施例を含む。もちろん、コンポーネントおよび/または技術のすべての考えられる組合せを説明することは不可能であるが、多くのさらなる組合せおよび置換が可能であることを当業者は理解することができる。したがって、新規のアーキテクチャは、添付の特許請求の範囲の趣旨および範囲内に入るすべてのそのような改変形態、変更形態、および変形形態を包含するものとする。さらに、用語「含む(includes)」が詳細な説明または請求項のいずれかで使

【図1】

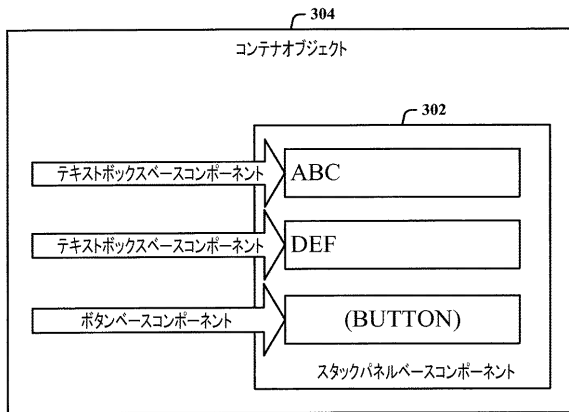


【図2】

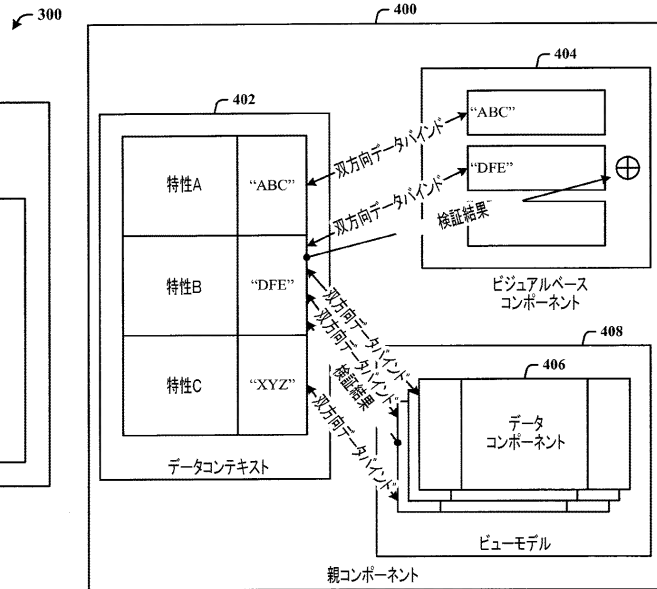




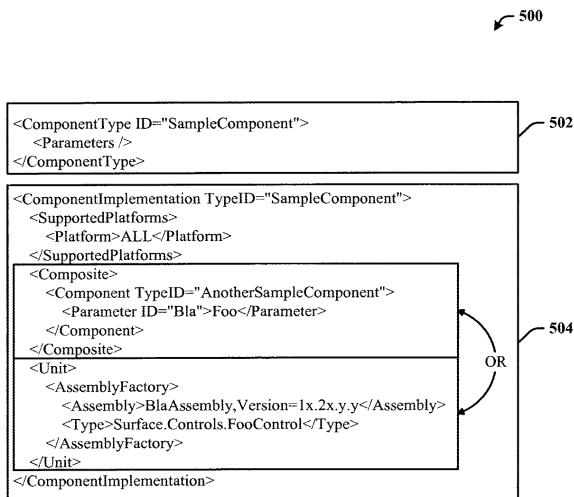
【 図 3 】



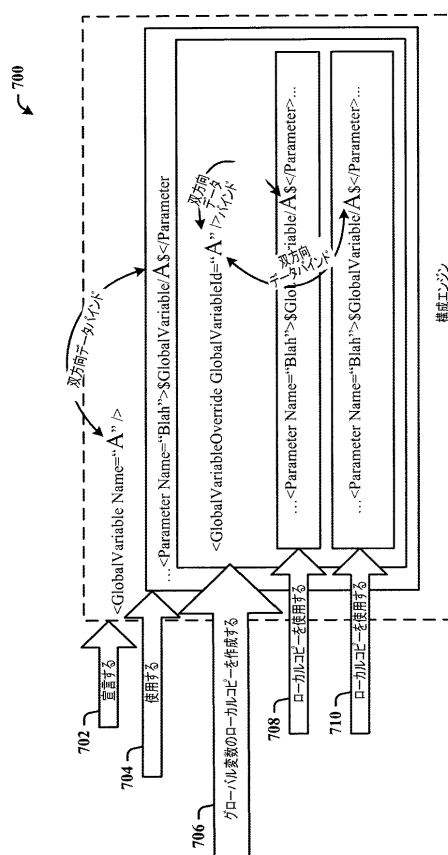
【 図 4 】



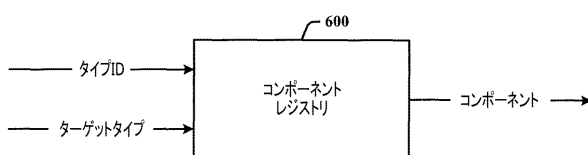
【 図 5 】



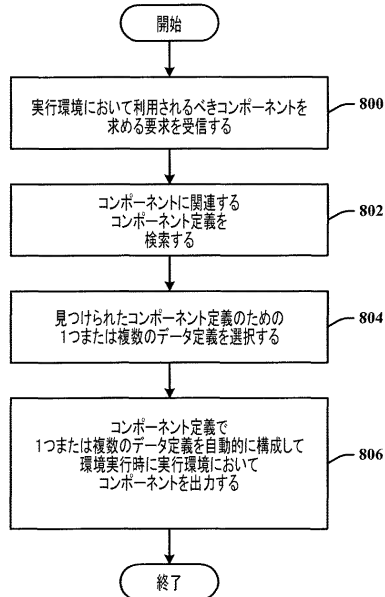
【 図 7 】



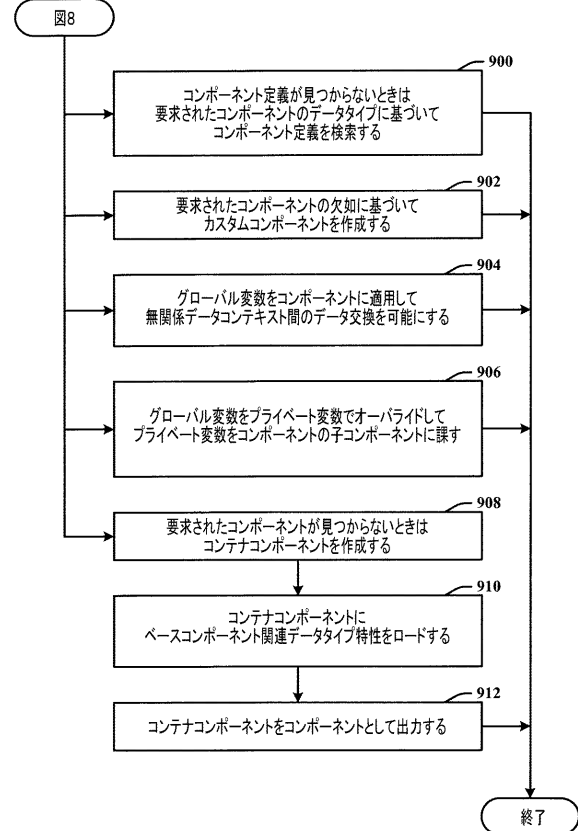
【 図 6 】



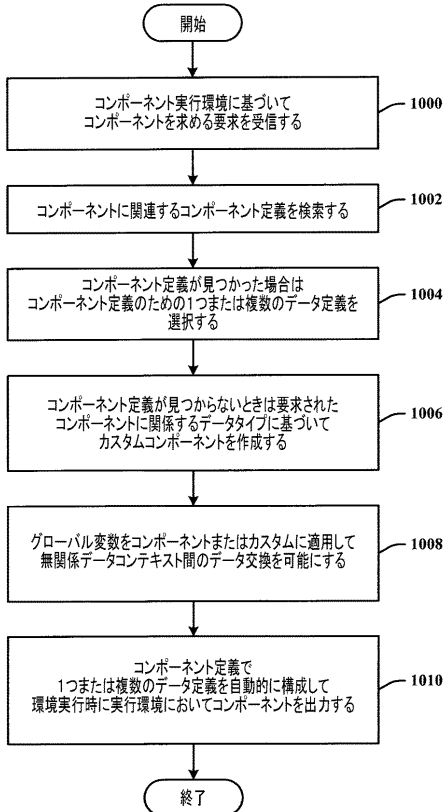
【図 8】



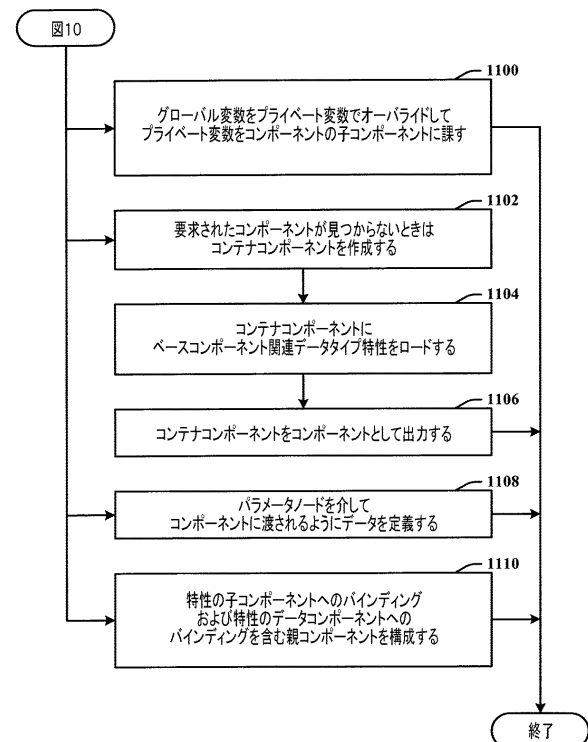
【図 9】



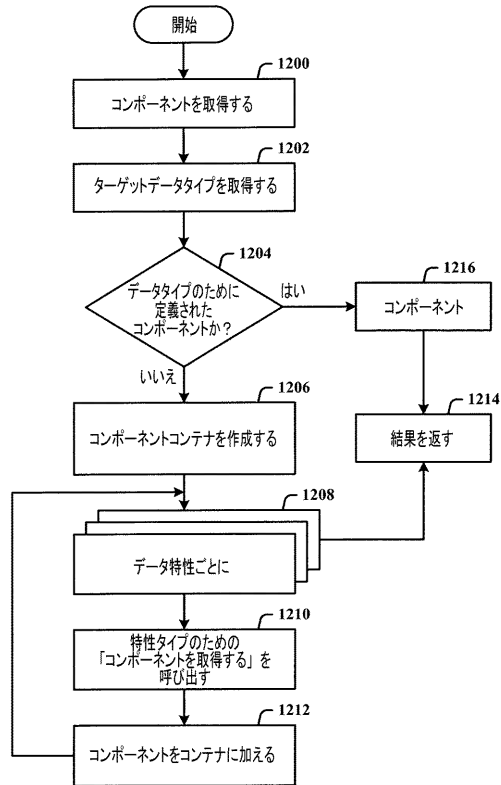
【図 10】



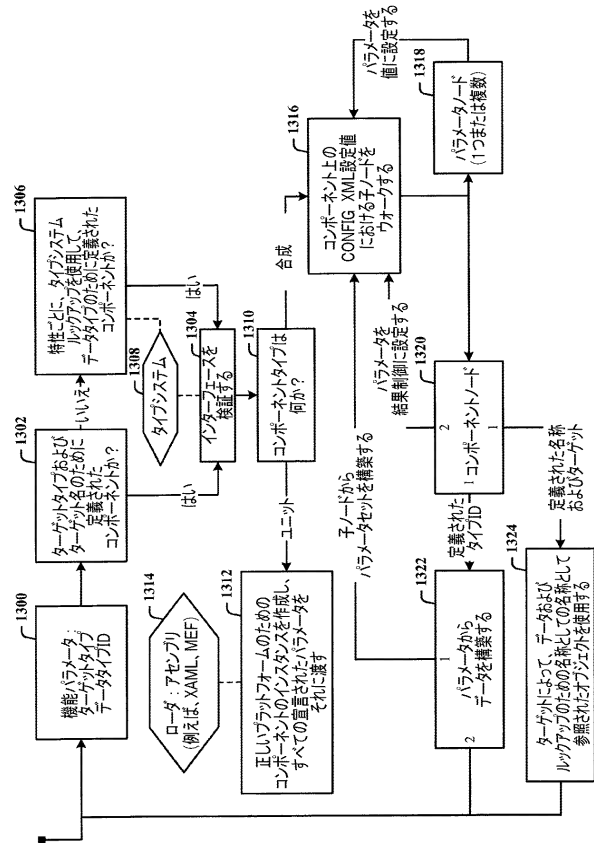
【図 11】



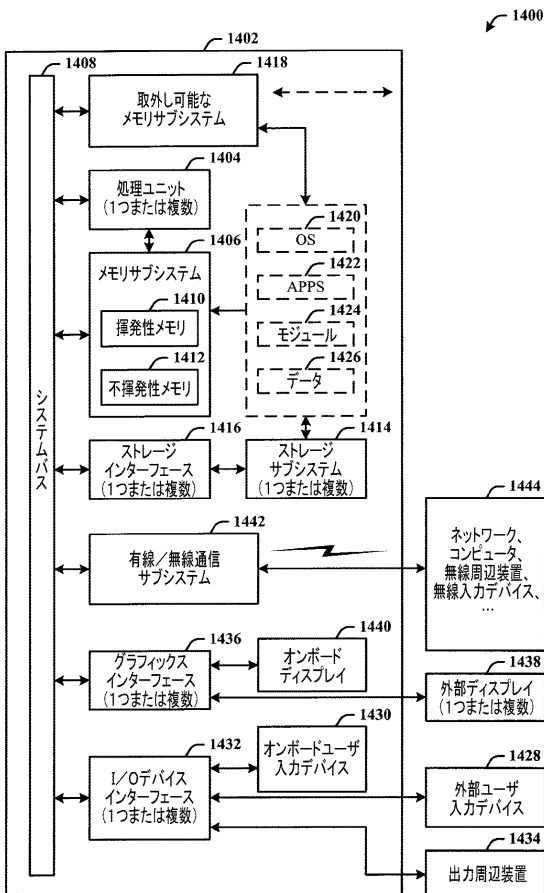
【図 12】



【図 13】



【図 14】



## フロントページの続き

- (74)代理人 100153028  
弁理士 上田 忠
- (74)代理人 100120112  
弁理士 中西 基晴
- (74)代理人 100196508  
弁理士 松尾 淳一
- (74)代理人 100147991  
弁理士 鳥居 健一
- (74)代理人 100119781  
弁理士 中村 彰吾
- (74)代理人 100162846  
弁理士 大牧 綾子
- (74)代理人 100173565  
弁理士 末松 亮太
- (74)代理人 100138759  
弁理士 大房 直樹
- (72)発明者 エフゲニー エヌ・ブイコフ  
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 エルシーエー・インターナショナル パテンツ内
- (72)発明者 フェリット フィンディク  
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 エルシーエー・インターナショナル パテンツ内
- (72)発明者 ラリアン エス・ベンソン  
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 エルシーエー・インターナショナル パテンツ内
- (72)発明者 ヴォロディミール ブイ・オトリシュコ  
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 エルシーエー・インターナショナル パテンツ内

審査官 大塚 俊範

- (56)参考文献 米国特許第06237004(US, B1)  
米国特許出願公開第2008/0163164(US, A1)  
特開2003-330715(JP, A)

- (58)調査した分野(Int.Cl., DB名)  
G06F 9/44