(19)日本国特許庁(JP)　(12)特　許　公　報(B2)　(11)特許番号
特許第6639418号
(P6639418)
(45)発行日　令和2年2月5日(2020.2.5)　(24)登録日　令和2年1月7日(2020.1.7)

(51)Int.Cl.　F I
G06F 8/36 (2018.01)　G06F 8/36
G06F 16/00 (2019.01)　G06F 16/00

請求項の数 15　(全 33 頁)

(21)出願番号　特願2016-568687(P2016-568687)
(86)(22)出願日　平成27年6月1日(2015.6.1)
(65)公表番号　特表2017-522639(P2017-522639A)
(43)公表日　平成29年8月10日(2017.8.10)
(86)国際出願番号　PCT/US2015/033554
(87)国際公開番号　WO2015/187567
(87)国際公開日　平成27年12月10日(2015.12.10)
審査請求日　平成30年4月26日(2018.4.26)
(31)優先権主張番号　62/006,662
(32)優先日　平成26年6月2日(2014.6.2)
(33)優先権主張国・地域又は機関
米国(US)
(31)優先権主張番号　14/539,521
(32)優先日　平成26年11月12日(2014.11.12)
(33)優先権主張国・地域又は機関
米国(US)

(73)特許権者　314015767
マイクロソフト　テクノロジー　ライセンシング，エルエルシー
アメリカ合衆国　ワシントン州　９８０５２　レッドモンド　ワン　マイクロソフト　ウェイ
(74)代理人　100079108
弁理士　稲葉　良幸
(74)代理人　100109346
弁理士　大貫　敏史
(74)代理人　100117189
弁理士　江口　昭彦
(74)代理人　100134120
弁理士　内藤　和彦

最終頁に続く

(54)【発明の名称】開発システムにおける意味論的なコンテンツアクセス

(57)【特許請求の範囲】
【請求項１】
プロセッサと、前記プロセッサに結合されたメモリとを備えた開発システムであって、
前記メモリは、前記プロセッサにより実行されると、前記開発システムを、
コンピュータシステムにおいてモデル化された複数の異なる要素タイプのうちの要素を備えるコンピュータシステムを開発するためのユーザ開発入力の指示を受信し、各要素タイプは、その要素タイプの要素のためのランタイム挙動を定義するプロパティおよびメソッドのセットを有し、
探索ユーザ入力メカニズムを有するユーザインターフェースディスプレイの表現を生成し、
前記コンピュータシステムの前記要素を探索するための探索基準を識別するユーザ探索クエリの指示を、前記探索ユーザ入力メカニズムから受信し、
前記ユーザ探索クエリに基づくタイプベースの探索パラメータを識別し、前記コンピュータシステムにおいてモデル化された前記異なる要素タイプの特定の１つを識別し、
前記特定の要素タイプの構造をパースし、前記特定の要素タイプの前記構造を探索するように構成された探索コードを生成することによって探索構成要素を生成し、前記探索構成要素は、前記特定の要素タイプのプロパティおよびメソッドの前記セットを探索するように構成され、
前記探索構成要素をインスタンス化して、前記探索コードを実行することによって、前記探索基準および前記タイプベースの探索パラメータに基づいて要素探索を実行し、探索

結果を取得する、
ように構成する命令を記憶する、開発システム。
【請求項２】
前記要素のコードまたはメタデータのうちの少なくとも１つを定義することによって、前記ユーザ開発入力の前記指示に基づいて前記コンピュータシステムの要素を開発するように構成されたインタラクティブ開発環境（IDE）を備える、請求項１に記載の開発システム。
【請求項３】
前記要素探索は、前記特定の要素タイプを有する要素を返すフィルタ基準として前記特定の要素タイプを使用して制約される、請求項１に記載の開発システム。
【請求項４】
前記ユーザ探索クエリは、キャラクタ文字列および前記特定の要素タイプを識別する、請求項３に記載の開発システム。
【請求項５】
前記返される要素は、前記キャラクタ文字列に一致するプロパティ値を有する前記特定の要素タイプの要素を備える、請求項４に記載の開発システム。
【請求項６】
前記命令は、前記開発システムを、各探索構成要素が前記異なるタイプのうちの所与の１つに対応する複数の探索構成要素から、前記特定の要素タイプに対応する探索構成要素を選択し、前記特定の要素タイプを有する前記コンピュータシステムの要素を識別し、前記識別された要素を、前記選択された探索構成要素を使用して、前記ユーザ探索クエリに基づいて探索することによって、前記探索結果を取得するように構成する、請求項１に記載の開発システム。
【請求項７】
前記選択された探索構成要素は、前記特定の要素タイプを有する複数の識別された要素の各々についてインスタンス化され、前記インスタンス化された複数の探索構成要素からの探索結果を集約することによって探索結果のセットを取得し、
前記命令は、前記集約された探索結果を、前記ユーザインターフェースディスプレイにおいて表示するようにディスプレイデバイスに指示するように、前記開発システムを構成する、請求項６に記載の開発システム。
【請求項８】
前記命令は、前記開発システムを、並列的に、
前記集約された探索結果を前記ユーザインターフェースディスプレイにおいて表示するように前記ディスプレイデバイスに指示し、
前記要素探索からの追加の探索結果を取得する
ように構成する、請求項７に記載の開発システム。
【請求項９】
前記要素のコードおよびメタデータを備える、前記要素の各々のシリアル化された表現を記憶するモデルストアをさらに備え、
前記要素探索は、前記モデルストア内の前記シリアル化された表現へアクセスすることによって実行される、請求項１に記載の開発システム。
【請求項１０】
前記命令は、前記開発システムを、
前記モデルストアにおけるシリアル化された表現のうちの特定の１つを、前記タイプベースの探索パラメータに基づいて識別し、
前記ユーザ探索クエリに基づいて、前記特定のシリアル化された表現を探索する、
ように構成する、請求項９に記載の開発システム。
【請求項１１】
前記命令は、前記開発システムを、
前記ユーザ探索クエリに一致する所与の要素の一部を識別するために前記特定のシリア

ル化された表現を分析し、
前記所与の要素の一部をユニークに識別するパス情報を取得する、
ように構成する請求項１０に記載の開発システム。
【請求項１２】
前記パス情報は、ユニフォームリソース識別子（URI）を備え、前記命令は、前記開発システムを、
前記所与の要素の一部を表示するエディタユーザインターフェースを生成するために選択可能である、前記ＵＲＩのユーザ選択可能な表現を生成するように構成する、請求項１１に記載の開発システム。
【請求項１３】
プロセッサと前記プロセッサに結合されたメモリを備える開発システムであって、
前記メモリは、前記プロセッサにより実行されると、前記開発システムを、
異なる要素タイプのアプリケーション要素を備えるアプリケーションを開発するための開発環境で開発者入力を受信し、
各要素タイプの構造を分析し、当該要素タイプの前記構造に基づいて当該要素タイプに特有の探索コードを生成することにより、前記異なる要素タイプのための複数の探索構成要素を生成し、各探索構成要素は、前記異なる要素タイプのうちの１つに対応し、前記対応する要素タイプのアプリケーション要素を探索するように構成されており、前記対応する要素タイプの前記構造は、前記対応する要素タイプを有する要素のランタイム挙動を定義するプロパティおよびメソッドのセットによって定義され、
前記探索構成要素を記憶する、
ように構成する命令を記憶し、
前記探索構成要素は、前記対応する要素タイプの前記アプリケーション要素を探索するようにインスタンス化される、開発システム。
【請求項１４】
前記命令は、前記開発システムを、
少なくとも１つの探索用語を有する探索クエリの指示を受信し、
前記探索クエリのためのタイプベースの探索パラメータを識別し、
前記タイプベースの探索パラメータに基づいて前記探索構成要素の１つを選択する、
ように構成し、
前記選択された探索構成要素は、前記探索用語に基づいて前記アプリケーション要素を探索するようにインスタンス化される、請求項１３に記載の開発システム。
【請求項１５】
コンピュータシステムを開発する、コンピュータによって実施される方法であって、
前記コンピュータシステムの要素を開発するための開発者ユーザ入力を受信するステップであって、前記コンピュータシステムは、複数の異なる要素タイプを備え、各要素タイプは、前記要素タイプの要素のためのプロパティ構造によって定義される、ステップと、
複数の異なる要素タイプをモデル化するデータストアにアクセスするステップと、
各異なる要素タイプについて、前記要素タイプのプロパティ構造に基づいて、対応するタイプベースの探索構成要素を生成するステップと、
前記コンピュータシステムの前記要素を探索するためのユーザ探索クエリを受信する探索インターフェースディスプレイを生成するステップと、
コンピュータシステムの前記要素の前記プロパティ構造に基づく意味論的な探索制約に基づいて前記タイプベースの探索構成要素の１つを選択するステップと、
選択されたタイプベースの探索構成要素をインスタンス化して、前記ユーザ探索クエリに基づいて、前記コンピュータシステムの前記要素の探索を実行することにより、探索結果を取得するステップと、
前記探索結果を表示する結果ディスプレイを生成するステップと、を含む方法。

【発明の詳細な説明】

【背景技術】
【０００１】
[0001] コンピュータプログラムは、様々な開発ツールにおいて開発される。たとえば、多くのソフトウェア開発者は、ソフトウェアを開発するために、インタラクティブ（または統合）開発環境（IDE）を使用する。開発者は、コンピュータシステム内でタイプのモデルを開発するため、および、これらモデルをカスタマイズするために、ＩＤＥを使用する。
【０００２】
[0002] 典型的なインタラクティブ開発環境は、開発者が、開発される必要のあるコードを開発および試験できるように、および、コンピュータシステムを所望されるようにカスタマイズするために、複数の異なるツールを含む。例によって、ＩＤＥは、ソースコードエディタ、１つまたは複数のビルド自動化ツール、および、コンピュータプログラマがソフトウェアを開発することを可能にするデバッガを含み得る。いくつかのＩＤＥは、例示的に、コンパイラ、インタプリタ、またはその両方を含む。それらは、グラフィックユーザインターフェースの構築を簡素化するためのバージョン制御システムおよび様々なツールを含み得る。それらはまた、オブジェクト指向ソフトウェア開発とともに使用するためのクラスブラウザ、オブジェクトブラウザ、およびクラス階層図を含み得る。したがって、開発者は、コードおよびメタデータを生成するために、コードおよびメタデータへのカスタマイズ化とともに、ＩＤＥを使用し得る。これは、所与の組織において使用するためにシステムを開発する際に利用され得る。たとえば、開発者は、アプリケーション要素に関連するソースコードおよびメタデータファイルを用いて作業し得る。１つのアプリケーションは、メタデータと、様々な方式でメタデータを使うコードとの両方を、生成または変更することを必要とし得る。
【０００３】
[0003] ＩＤＥを使用してソフトウェアを生成またはカスタマイズする際に、アプリケーション開発者は、アプリケーション内の特定の概念（これは、タイプとして表され得る）をモデル化し、必要な場合、コードを書く。開発者がしばしばそのためにＩＤＥを使用する大規模なアプリケーションは、数千もの異なるタイプを含み得る。
【０００４】
[0004] 例によって、いくつかのコンピュータシステムは、他のシステムの中でも、エンタープライズリソースプラニング（ERP）システム、顧客関係管理（CRM）システム、取扱商品（LOB）システムなどのビジネスシステムを含む。これらのタイプのコンピュータシステムはしばしば、モデル化されカスタマイズされた何千もの異なるタイプを有する。例によって、そのようないくつかのビジネスシステムはしばしば、他の多くのタイプは言うに及ばず、数千もの異なる形式を有する。
【０００５】
[0005] ビジネスシステムは、多くのタイプを有するコンピュータシステムのタイプだけではない。たとえば、ゲームシステム、または、種々様々な他のシステムのタイプもまた、しばしば、ソフトウェアシステムでモデル化された何千もの異なるタイプを有する。
【０００６】
[0006] 上記議論は単に、一般的な背景情報のために提供されており、権利主張された主題の範囲を決定する際の助けとして使用されることは意図されていない。
【発明の概要】
【０００７】
[0007] ソフトウェア開発中、開発者は、開発処理を促進するための要素を探索する。探索アーキテクチャは、開発者が、ある基準を満足するメタデータおよびコードを探索することを可能にする。探索アーキテクチャは、開発者のクエリに適切である結果を返すために、意味論的な要素情報を導入する。
【０００８】
[0008] 開発システムは、一例では、ユーザ開発入力を感知し、ユーザ開発入力に基づい

て、コンピュータシステムの要素を変化させる、開発モジュールを備える。これら要素は、コンピュータシステムにおいてモデル化されたタイプを備える。ユーザインターフェースモジュールは、ユーザ入力メカニズムを用いてユーザインターフェースディスプレイを生成し、コンピュータシステムの要素を探索するためのユーザ探索クエリを示す、ユーザ入力メカニズムを介して受け取られたユーザ探索入力を感知する。探索エンジンは、ユーザ探索クエリのためのタイプベースの探索パラメータを識別する。探索エンジンは、タイプベースの探索パラメータに基づいて、タイプベースの探索構成要素を活性化するように制御される。タイプベースの探索構成要素は、ユーザインターフェースディスプレイにおいて、探索結果のセットを返すために、要素探索を実行する。

【０００９】

[0009] この要約は、発明を実施するための形態において以下にさらに説明される概念の選択をより簡単な形式で導くために提供される。この概要は、権利主張された主題の主要な特徴または本質的な特徴を特定することは意図されておらず、また、権利主張された主題の範囲を決定する際における助けとして使用されることも意図されていない。権利主張された主題は、背景技術において注目された何れかまたはすべての欠点を解決する実施に限定されない。

【図面の簡単な説明】

【００１０】

【図１】[0010]意味論的な探索アーキテクチャの一例のブロック図である。

【図２】[0011]意味論的な探索構成要素を生成するためのメソッド（方法）の一例を例示するフロー図である。

【図３】[0012]一例における意味論的な探索機能を例示するブロック図である。

【図４】[0013]意味論的な探索構成要素を使用して探索を実行するためのメソッド（方法）の一例を例示するフロー図である。

【図５】[0014]ユーザインターフェースディスプレイの一例を例示する図である。

【図６】[0015]ユーザインターフェースディスプレイの一例を例示する図である。

【図７】[0016]クラウドコンピューティングアーキテクチャにおいて展開された、図１に例示されるアーキテクチャの一例を図示するブロック図である。

【図８】[0017]図１に図示されるアーキテクチャとともに使用され得るモバイルデバイスの様々な例を図示する図である。

【図９】図１に図示されるアーキテクチャとともに使用され得るモバイルデバイスの様々な例を図示する図である。

【図１０】図１に図示されるアーキテクチャとともに使用され得るモバイルデバイスの様々な例を図示する図である。

【図１１】図１に図示されるアーキテクチャとともに使用され得るモバイルデバイスの様々な例を図示する図である。

【図１２】図１に図示されるアーキテクチャとともに使用され得るモバイルデバイスの様々な例を図示する図である。

【図１３】[0018]一例のコンピューティング環境のブロック図である。

【発明を実施するための形態】

【００１１】

[0019] 図１は、意味論的な探索アーキテクチャ１００の一例のブロック図である。アーキテクチャ１００は、開発機能１０４を有するインタラクティブ開発システム（たとえば、IDE）１０２を含む。図１は、開発者１０６が、コンピュータシステムにおいて実行されているアプリケーション要素１０７の開発および／またはカスタマイズを実行するために、システム１０２とインタラクトすることを図示する。たとえば、アプリケーション要素の各々は、メタデータ１０９を含み、コード１１１も同様に含み得る。例によって、開発者１０６は、メタデータ１０９およびコード１１１を生成または変更することによって、アプリケーションのための要素１０７を開発するために、機能１０４を使用する。限定によってではなく、一例では、要素１０７は、オブジェクト指向プログラミング環境にお

けるオブジェクトを備える。任意の適切なプログラミング言語が、システム１０２において利用され得る。

【００１２】

[0020] 例示された例では、モデルストア１０８は、様々な異なるアプリケーション要素のタイプ（たとえば、type）に対応するメタデータおよびコードを記憶し、たとえば、システム１０２および探索構成要素コード生成器（search component code generator）１３０によってアクセス可能である。「ｔｙｐｅ」は、システムにおいてモデル化されたコンセプトを表す抽象的概念を称する。たとえば、ビジネスシステムでは、要素タイプは、いくつか名前を挙げると、形式、エンティティ、クラス、テーブル、メニューアイテム、セキュリティ役割、および／または、許可を含み得る。一例では、テーブルオブジェクトは、データベース内にアプリケーションデータを維持するためのメタデータおよびコードを含む。別の例では、形式オブジェクトは、アプリケーションユーザが、情報を使い、アプリケーションとインタラクトするために、様々なデバイスにおいて表示されるべき情報コンテンツを説明するためのメタデータおよびコードを含む。

【００１３】

[0021] 一例では、アプリケーション要素１０７を開発するために開発機能１０４を利用している場合、開発者１０６は、アプリケーション要素１０７をコーディングするための統合された、すなわちＩＤＥビューを提示される。単純化された一例が、例示のために以下の表１に図示される。

【００１４】

【表１】

表１

```
public class Table1 extends common
{
        /// <summary>
        ///
        /// </summary>
        private void Method1()
        {
        }

        /// <summary>
        ///
        /// </summary>
        public void insert()
        {
                super();
        }

}
```

【００１５】

[0022] このように、アプリケーション要素１０７を開発するために開発者１０６によってオーサリングされるコードおよびメタデータは、たとえば、アプリケーション要素１０７をコーディングするためのユーザフレンドリなインターフェースを提供するコードエディタビューにおいて、第１のフォーマットで表される。しかしながら、開発者１０６が、第１のフォーマットで、コードおよびメタデータをビューおよびオーサリングする一方、インタラクティブ開発システム１０２は、第１のフォーマットとは異なる第２のフォーマットで、開発されたアプリケーション要素のソースコード表現を維持し、このソースコード表現において動作する。一例では、コードおよびメタデータを備えるシリアル化された表現は、各要素のためにシステム１０２によって維持される。第２のフォーマットは、マシン読取可能であり、システム１０２による実行に対して従順である。一例では、限定に

よってではなく、モデルストア１０８は、ソースコード表現をＸＭＬファイルとして記憶するファイルシステムを備える。メタデータおよびコードＸＭＬは、各々が自身のタイプを有するシリアル化された要素構造（element structure）を備える。以下の表２は、表１に図示された統合ビューに対応する例示的なＸＭＬファイルを図示する。

【００１６】

【表２－１】

表２

```
<?xml version="1.0" encoding="utf-8"?>
<AxTable xmlns:i="http://www.w3.org/2001/XMLSchema-instance">
    <Name>Table1</Name>
    <SourceCode>
        <Declaration><![CDATA[
public class Table1 extends common
{
}
]]></Declaration>
        <Methods>
            <Method>
                <Name>Method1</Name>
                <Source><![CDATA[
    /// <summary>
    ///
    /// </summary>
    private void Method1()
    {
    }

]]></Source>
            </Method>
            <Method>
                <Name>insert</Name>
                <Source><![CDATA[
    /// <summary>
    ///
    /// </summary>
    public void insert()
    {
      super();
    }
```

【００１７】

【表２－２】

```
]]></Source>
                </Method>
            </Methods>
        </SourceCode>
        <Label>@SYS1234</Label>
        <DeleteActions />
        <FieldGroups>
            <AxTableFieldGroup>
                <Name>AutoReport</Name>
                <Fields />
            </AxTableFieldGroup>
            <AxTableFieldGroup>
                <Name>AutoLookup</Name>
                <Fields />
            </AxTableFieldGroup>
            <AxTableFieldGroup>
                <Name>AutoIdentification</Name>
                <AutoPopulate>Yes</AutoPopulate>
                <Fields />
            </AxTableFieldGroup>
            <AxTableFieldGroup>
                <Name>AutoSummary</Name>
                <Fields />
            </AxTableFieldGroup>
            <AxTableFieldGroup>
                <Name>AutoBrowse</Name>
                <Fields />
            </AxTableFieldGroup>
        </FieldGroups>
        <Fields>
            <AxTableField xmlns=""
                i:type="AxTableFieldString">
                <Name>Field1</Name>
            </AxTableField>
```

【００１８】

【表２－３】

```
        </Fields>
        <FullTextIndexes />
        <Indexes />
        <Mappings />
        <Relations />
        <StateMachines />
    </AxTable>
```

【００１９】

[0023] 上記例において、メタデータおよびコードは、１つのＸＭＬファイルへシリアル化される。すなわち、コードの断片（すなわち、構造化されていない文字列）とメタデータの断片（すなわち、構造化されたプロパティおよび値のセット）が、ＸＭＬファイルに点在する。しかしながら、当業者であれば、他のフォーマットが利用され得ることを理解する。

【００２０】

[0024] 開発者１０６は、（パーソナルコンピュータ、タブレット、別のモバイルデバイス等のような）個別の開発者デバイスを介して、またはダイレクトに、インタラクティブ開発システム１０２とインタラクトし得る。開発者１０６はまた、ネットワークを介して（たとえば、遠隔的に）システム１０２とインタラクトし得る。開発者１０６は、例示のみのために、図１におけるシステム１０２とダイレクトに（たとえば、ローカルに）インタラクトしていることが図示される。

【００２１】

[0025] インタラクティブ開発システム１０２は、一例では、プロセッサ１１０およびユーザインターフェースモジュール１１２を含む。ユーザインターフェースモジュール１１２は、開発者１０６によるインタラクションのために、ユーザ入力メカニズム１１８を用いてユーザインターフェースディスプレイ１１６を生成する。開発者１０６は、インタラクティブ開発システム１０２を制御および操作するために、ユーザ入力メカニズム１１８とインタラクトする。一例では、開発者１０６は、開発機能１０４を実施するためのみならず、探索モジュール１２０およびナビゲーションモジュール１２２を使用するために、これを行い得る。システム１０２は、同様に他のアイテム１１４も含み得る。

【００２２】

[0026] 開発者１０６は、モデルストア１０８における既存のコードとメタデータを使用し得るか、または、新たなコードとメタデータ、または、既存および新たなコードとメタデータとの組合せを生成し得る。そうする際に、モデルストア１０８における既存の要素が、変更または消去され、新たな要素が追加され得る。開発を促進するために、開発者１０６は、興味のある要素を発見するために、モデルストア１０８の探索を所望し得る。たとえば、開発者１０６は、アプリケーション内でカスタマイズするための特定の要素を位置決めすることを所望し得る。

【００２３】

[0027] しかしながら、しばしば極めて大きな、コードベースのサイズに部分的に起因して、ある開発者探索基準を満足する要素を発見することは困難であり得る。１つの探索実施は、事前にインデクスをビルドすることに依存する。これに対して開発者クエリが実行される。たとえば、コンテンツをナビゲートしインデクスをビルドするクローラが存在する。これは、その後、探索するために使用される。開発プラットフォームのケースでは、いったん要素が変更または追加されると、インデクスは、期限切れになる。さらに、コードベースのサイズを考慮すると、インデクスを繰り返しリビルドすることは、多大な時間

を費やす。

【００２４】

[0028] 例示された例では、意味論的な探索アーキテクチャ１００は、探索モデル１０８が要素の名前、タイプ、および／または、プロパティを意味論的に考慮するために、探索モジュール１２０を使用することによって探索結果を取得する。この探索は、要素タイプの構造と、要素タイプおよびこれら要素タイプ内のプロパティの意味との理解を導入するという点において意味論的である。以下にさらに詳細に議論されるように、要素タイプの特定の構造は、モデルストア１０８の要素を探索することに関連し得る。限定によってではなく、例示によって、各要素タイプは、その要素タイプの要素のためのランタイム挙動を定義するプロパティ、メソッド、および／または、計算の特定の構造を有する。たとえば、テーブル要素タイプは、名前（たとえば、「customer table」）、およびカスタマのための属性を識別するプロパティ（たとえば、カスタマID、アドレス等）のセットを含み得る。また、この例では、テーブル要素タイプは、カスタマのための値を計算するためのメソッド、および／または、この値を表示するためのメソッドを含み得る。

【００２５】

[0029] アーキテクチャ１００の全体動作をより詳細に説明する前に、概要が提供されるであろう。一例では、探索モジュール１２０は、１つまたは複数のトークンの形式で探索基準を定義するユーザ探索クエリを受け取る探索エンジンを備える。トークンは、探索パラメータを定義し、文字列または用語を形成する１つまたは複数のキャラクタを含み得る。探索エンジンは、意味論的な探索パラメータまたは制約を識別するために、開発者１０６からの探索クエリをパースし、開発者１０６へ提供される探索結果のセットを取得するために、モデルストア１０８に対する探索クエリを実行する。一例では、クエリを実行することは、アプリケーション要素において、１つまたは複数のトークンを、プロパティおよび／またはメソッドに対して一致させることを含む。

【００２６】

[0030] 意味的な探索パラメータは、探索クエリ自身において明示的に提供され得るか、または、探索クエリから示唆または導出され得る。たとえば、図５に関して以下に説明される例では、開発者１０６は、

ｔｙｐｅ：ｔａｂｌｅ、ｍｅｔｈｏｄ ｎａｍｅ：ｉｎｓｅｒｔ ｐｅｒｐｅｒｔｙ：“ｓｏｕｒｃｅ＝ｃｒｏｓｓｃｏｍｐａｎｙ”

からなる探索クエリを入力する。

【００２７】

[0031] 本明細書で、クエリから識別された意味論的な探索パラメータは、タイプベースの制約を備える。すなわち、開発者１０６は、要素タイプ「ｔａｂｌｅ」からなる要素が、トークン「ｉｎｓｅｒｔ」に一致する名前と、トークン「ｃｒｏｓｓｃｏｍｐａｎｙ」に一致する値を有するソースプロパティとを有するメソッドを有することを所望する。本明細書では、実施形態がタイプベースの制約のコンテキストで議論されているが、他の意味論的な探索パラメータまたは制約が使用され得ることが注目される。

【００２８】

[0032] 例示された例では、探索を実行するために、探索モジュール１２０が、探索構成要素コード生成器１３０によって生成された複数の探索構成要素（すなわち、探索構成要素１２４および１２６）を記憶する探索構成要素ストア１２８にアクセスする。探索構成要素コード生成器１３０を使用して探索構成要素を生成する一例が、図２に関して以下にさらに詳細に議論される。手短に言えば、探索構成要素コード生成器１３０は、モデルストア１０８においてモデル化された異なる各要素タイプのための探索構成要素を生成するように構成されたプロセッサ１３１を含む。各探索構成要素は、要素タイプのうちの特定の１つのために生成される。このように、各探索構成要素は、そのために生成された特定の要素タイプの構造に特有である。一例では、探索構成要素は、開発者１０６によって使用され得るすべての可能な要素タイプのために生成され、ストア１２８に記憶される。たとえば、一例では、あらかじめ定義された要素タイプのセットが、開発者１０６に利用可

能であり、任意の新たな要素タイプが、システム１０２への更新を介して、システム１０２へ追加される。

【００２９】

[0033] 探索モジュール１２０は、モデルストア１０８内の要素から、結果のリストを返すために使用されるべき探索構成要素ストア１２８からの探索構成要素のうちの１つまたは複数を識別するために、探索クエリからのタイプベースの探索制約を使用する。探索構成要素を使用する探索モデルストア１０８の一例が、図４に関して以下にさらに詳細に議論される。手短に言えば、探索モジュール１２０は、各タイプベースの探索制約のための対応する探索構成要素を識別する。上記例において、探索モジュール１２０は、テーブル要素タイプのために生成された探索構成要素（すなわち、探索構成要素１２４または１２６）を識別する。識別された探索構成要素は、探索クエリにおけるメソッド名およびプロパティ値に一致する要素を識別するために、テーブル要素タイプを有するモデルストア１０８内の要素ごとにインスタンス化される。探索モジュール１２０は、インスタンス化された探索構成要素から取得された探索結果を集約する。ナビゲーションモジュール１２２は、探索結果のユーザナビゲーションを容易にする。

【００３０】

[0034] したがって、探索アーキテクチャ１００は、事前にインデクスをビルドまたは維持する必要なく、モデルストア１０８の探索を実行する際に、アプリケーション要素１０７に関する意味論的な情報を導入する。これは、開発システムにおける探索機能を実行する際における処理負荷および時間、ならびにメモリ要件を低減し得、ユーザのクエリに対する探索結果関連性を改善し得る。

【００３１】

[0035] 例示のために、図１の例では、異なる各タイプのアプリケーション要素のために、アーキテクチャ１００は、その要素タイプのモデルストア１０８の既存の要素を探索するように構成された特定の探索構成要素を維持する。しかしながら、これらの探索要素はまた、新たな要素のタイプ（すなわち、すべての要素タイプは、あらかじめ定義された探索構成要素を有する）または特定のプロパティに関わらず、開発者１０６によってモデルストア１０８へ追加された任意の新たな要素を探索することができる。逆に、インデクスされた探索システムのケースでは、モデルストア１０８へ新たな要素を追加することは、インデクスが、新たな要素を含めるように更新されることを要求するであろう。

【００３２】

[0036] さらなる例示のために、モデルストア１０８が、２つの異なる要素タイプ（すなわち、テーブル要素タイプ１３２および形式要素タイプ１３４）を含むと仮定されたい。第１の探索構成要素１２４は、要素タイプ１３２のために生成され、第２の探索構成要素１２６は、要素タイプ１３４のために生成される。図１の例では、コード生成器１３０は、各要素タイプのための一度だけ実行される必要がある。このように、探索構成要素１２４および１２６が生成されると、コード生成器１３０は、たとえ、モデルストア１０８内の既存のタイプ１３２および１３４の要素が修正されても、および／または、モデルストア１０８に、新たなタイプ１３２および１３４の要素が追加されても、これらを再生成または修正する必要はない。

【００３３】

[0037] 探索構成要素１２４は、探索モジュール１２０が、タイプ１３２の要素を探索する場合にインスタンス化され、探索構成要素１２６は、探索モジュール１２０が、タイプ１３４の要素を探索する場合にインスタンス化される。一例において、要素タイプ１３２および１３４の両方が探索されている場合、探索構成要素１２４および１２６は、探索時間を短縮するために並列的に動作し得る。図１には、２つの要素タイプとタイプベースの探索構成要素しか図示されていないが、任意の数の要素タイプおよび意味論的な探索構成要素が実施され得ることが注目される。

【００３４】

[0038] モデルストア１０８および探索構成要素ストア１２８は、図１では、インタラク

ティブ開発システム１０２と分離しているとして例示されているが、モデルストア１０８および／または探索構成要素ストア１２８は、インタラクティブ開発システム１０２の一部であり得ることが注目される。しかしながら、帯域幅およびレイテンシの考慮によって、いくつかの実施では、モデルストア１０８および探索構成要素ストア１２８は、同じコンピューティングシステムにおいて維持され得る。しかし、これは単なる一例である。このように、探索要求および結果は、ネットワークを介して送信され得る一方、探索アーキテクチャ１００は、モデルストア１０８の送信を必要としない。繰り返すが、これはアーキテクチャの単なる一例である。

【００３５】

[0039] また図１は、種々の異なる機能ブロックを図示する。ブロックは、より多くの機能が各ブロックによって実行されるように、統合され得るか、または、これら機能がさらに分散されるように、分割され得ることが注目されるであろう。

【００３６】

[0040] 上記議論は、モデルストア１０８および探索構成要素ストア１２８を含む多くのデータストアを図示していることが注目されるべきである。これらは２つの独立したデータストアとして図示されているが、単一のデータストア内においても形成され得る。それに加えて、これらデータストアにおけるデータは、追加の多くのデータストア内にも同様に記憶され得る。また、データストアは、それらにアクセスする環境、エージェント、モジュール、および／または、構成要素に対してローカルであり得るか、または、そこから離れており、これら環境、エージェント、モジュール、および／または、構成要素によってアクセス可能であり得る。同様に、いくつかは局所的であり得る一方、他は遠隔的である。

【００３７】

[0041] 例示された例では、プロセッサ１１０および１３１は、関連付けられたメモリおよびタイミング回路（個別に図示されていない）を備えたコンピュータプロセッサを備える。これらは、これらが属するエージェントまたは環境の機能的な部分であり、その環境またはエージェントにおける他のアイテムによって例示的に活性化され、その環境またはエージェントにおける他のアイテムの機能を容易にする。

【００３８】

[0042] 図２は、意味論的な探索構成要素を生成するためのメソッド２００の一例を例示するフロー図である。例示のために、限定によってではなく、メソッド２００は、アーキテクチャ１００がタイプベースの探索構成要素を生成するコンテキストにおいて説明されるであろう。

【００３９】

[0043] メソッド２００は、定期的に、および／または、条件またはイベントに応じて開始され得る。たとえば、メソッド２００は、システム１０２によってサポートされている要素タイプを追加または修正するシステム１０２への更新に応じて開始され得る。別の例では、メソッド２００は、（たとえば、開く、閉じる、保存する等のような制御を、ユーザインターフェース１１６上において選択することによって）開発者１０６からの入力に応じて開始され得る。

【００４０】

[0044] ステップ２０２において、探索構成要素コード生成器１３０が、モデルストア１０８にアクセスし、ステップ２０４において、タイプベースの探索構成要素を生成するための新たな任意の要素タイプが存在するか否かを判定する。一例では、探索構成要素コード生成器１３０は、モデルストア１０８内の要素のうちのいくつか（たとえば、直近の変更および追加）またはすべてを分析し、これら要素を、既存のまたは既知の要素タイプ（すなわち、タイプ１３２および１３４）と比較する。たとえば、探索構成要素コード生成器１３０は、開発者１０６によって変更または追加された要素を識別する。

【００４１】

[0045] 新たな要素タイプが識別されると、探索構成要素コード生成器１３０は、ステッ

プ２０８において、新たな要素タイプのためのタイプベースの探索構成要素を生成するために、ステップ２０６において、新たな要素タイプの構造を分析する。一例では、探索構成要素コード生成器１３０は、新たな要素タイプの構造を、任意のサブタイプへパースし、タイプおよび／またはサブタイプがどのプロパティを含んでいるのか、任意の子要素タイプ、要素タイプからどの要素タイプが由来するのか、および要素タイプのプロパティゲッタのための実施を決定する。各プロパティゲッタは、たとえば、要素タイプにおけるプロパティの場所、および／または、他のプロパティに対する関係に基づいて、要素タイプのプロパティを取得するための機能を定義する。一例では、探索構成要素コード生成器１３０は、要素タイプ構造の異なる部分を探索するために、異なるプロパティゲッタコードを生成する。たとえば、コードの１ピースは、与えられた要素の部分におけるメソッドを探索し得、コードの１ピースは、制御等を見得る。上記で議論されたカスタマテーブル要素タイプの例に関して、１つのプロパティゲッタは、「ｃｕｓｔｏｍｅｒ　ＩＤ」プロパティを返すように構成され得、別のプロパティゲッタは、「ａｄｄｒｅｓｓ」プロパティを返すように構成され得る。

【００４２】

[0046] 各探索構成要素は、探索構成要素が生成される要素タイプに基づく、定義された要素パターン（たとえば、子要素のパターン、プロパティ、メソッド等）に従うように構成される。たとえば、限定によってではなく、図１では、要素タイプ１３２および１３４は、互いに異なる子要素のパターンを有する。探索構成要素１２４は、要素タイプ１３２に関連付けられた子要素の値を検証し返すための探索メソッドを呼び出すように構成され、探索構成要素１２６は、要素タイプ１３４に関連付けられた子要素の値を検証し返すための探索メソッドを呼び出すように構成される。

【００４３】

[0047] 例によって、１つのメタデータ要素は、ツリーデータ構造を備え、名前およびメタデータ要素タイプによって定義される。メタデータ要素タイプはさらに、プロパティのセットによって定義され、各プロパティは、プロパティ値の名前およびタイプによって定義される。プロパティ値のタイプは、たとえば、限定によってではなく、（文字列（はいいいえ、日付、タグ等）に変換可能な）プリミティブであり得る。そのようなプロパティは「ｓｉｍｐｌｅ　ｐｒｏｐｅｒｔｙ」と称される。プロパティ値の別のタイプは、子メタデータ要素を含むメタデータ要素タイプである。そのようなプロパティは「ｎｏｄｅ　ｐｒｏｐｅｒｔｙ」と称され得る。ルートメタデータ要素は、メタデータストレージにダイレクトに記憶され、親を有していない要素である。子メタデータ要素は、他の要素ノードプロパティのいくつかに含まれる要素である。メタデータパスは、メタデータ要素をユニーク識別し、メタデータ要素の位置決めを容易にする文字列を備える。一例では、パスの形式は以下の通りである。

ｄｙｎａｍｉｃｓ：／／＜Ｒｏｏｔ＿ｔｙｐｅ＞／＜Ｒｏｏｔ＿ｅｌｅｍｅｎｔ＿ｎａｍｅ＞［／＜Ｓｕｂｔｙｐｅ＿１＞／＜Ｓｕｂｅｌｅｍｅｎｔ＿ｎａｍｅ＿１＞［／＜Ｓｕｂｔｙｐｅ＿２＞／＜Ｓｕｂｅｌｅｍｅｎｔ＿ｎａｍｅ＿２＞［．．．］］］

ここで、

＜Ｒｏｏｔ＿ｔｙｐｅ＞－ルートメタデータ要素のタイプ

＜Ｒｏｏｔ＿ｅｌｅｍｅｎｔ＿ｎａｍｅ＞－ルート要素の名前

＜Ｓｕｂｔｙｐｅ＿ｉ＞－ツリーにおける各子メタデータ要素のタイプ

＜Ｓｕｂｅｌｅｍｅｎｔ＿ｎａｍｅ＿ｉ＞－ツリーにおける各子メタデータ要素の名前

【００４４】

[0048] ステップ２１０では、生成された意味論的な探索構成要素が、探索構成要素ストア１２８に記憶される。任意の追加の新たな要素タイプが、ステップ２１２において識別されると、ステップ２０６、２０８、および２１０が、新たな要素タイプのために繰り返される。

【００４５】

[0049] 図３は、一例における、意味論的な探索機能を例示するブロック図である。例示のために、限定によってではなく、図３は、アーキテクチャ１００における意味論的な探索機能のコンテキストにおいて説明されるであろう。

【００４６】

[0050] ブロック２５０は、インタラクティブ開発システム１０２へインターフェースを提供する。ブロック２５０によって、探索モジュール１２０は、ブロック２５２におけるクエリパーサへ提供される探索クエリを受け取る。クエリは、フィルタを定義する１つまたは複数の探索基準を提供し、任意の適切な構文または文法を有し得る。

【００４７】

[0051] １つの比較的単純な構文の例が以下に提供される。

探索クエリは、ｓｅａｒｃｈ＿ｓｔｒｉｎｇであり、ここで、

ｓｅａｒｃｈ＿ｓｔｒｉｎｇ＝ｅｍｐｔｙ＿ｓｔｒｉｎｇ

ｓｅａｒｃｈ＿ｓｔｒｉｎｇ＝ｔｅｘｔ＿ｗｉｔｈｏｕｔ＿ｃｏｌｏｎ

ｓｅａｒｃｈ＿ｓｔｒｉｎｇ＝ｆｉｌｔｅｒ

ｓｅａｒｃｈ＿ｓｔｒｉｎｇ＝ｓｅａｒｃｈ＿ｓｔｒｉｎｇ ｆｉｌｔｅｒ

ｆｉｌｔｅｒ＝ｆｉｌｔｅｒ＿ｎａｍｅ：ｆｉｌｔｅｒ＿ｖａｌｕｅ

ｆｉｌｔｅｒ＿ｖａｌｕｅ＝ｔｅｘｔ＿ｗｉｔｈｏｕｔ＿ｃｏｍｍａ

ｆｉｌｔｅｒ＿ｖａｌｕｅ＝“ａｎｙ＿ｔｅｘｔ”

ｆｉｌｔｅｒ＿ｖａｌｕｅ＝ｆｉｌｔｅｒ＿ｖａｌｕｅ，ｆｉｌｔｅｒ＿ｖａｌｕｅ

ｆｉｌｔｅｒ＿ｎａｍｅ＝ｎａｍｅ ＯＲ ｔｙｐｅ ＯＲ ｍｏｄｅｌ ＯＲ ｐｒｏｐｅｒｔｙ

したがって、探索文字列は、一般的な形式においてフィルタのセットからなる。

＜ｆｉｌｔｅｒ＿１＞：＜ｆｉｌｔｅｒ＿１＿ｖａｌｕｅ＞［＜ｆｉｌｔｅｒ＿２＞：＜ｆｉｌｔｅｒ＿２＿ｖａｌｕｅ〉．．．［＜ｆｉｌｔｅｒ＿Ｎ＞：＜ｆｉｌｔｅｒ＿Ｎ＿ｖａｌｕｅ＞］］

ここで、＜ｆｉｌｔｅｒ＿ｉ＞は、許容可能なフィルタ名のうちの１つであり、＜ｆｉｌｔｅｒ＿ｉ＿ｖａｌｕｅ＞は、カンマ区切りされ、可能な引用符付きのフィルタリング値である。

【００４８】

[0052] 上記に例示され図３に図示されるように、ユーザ探索基準の一例は、要素名である。これは、１つの文字列または文字列のセットを指定し得る。要素の名前が、文字列のうちの少なくとも１つを含むのであれば、要素は、この基準を満足していると考慮される。カンマ区切りされた各値は、許容可能な要素名であり得る。一例では、要素名は、デフォルトフィルタである。したがって、探索クエリが、単一のトークンを含んでいるのであれば、それは要素名であると仮定される。この例では、タイプベースの制約が識別されないと、探索アーキテクチャは、利用可能なすべての要素タイプのための探索構成要素をインスタンス化し得る。

【００４９】

[0053] 別の例示的な基準は、要素タイプである。これは、１つの要素タイプまたは要素タイプのセットを指定し得る。要素は、それが、指定されたタイプのうちの１つであれば、この基準を満足していると考慮される。カンマ区切りされた各値は、要素タイプ（すなわち、テーブル、クラス、フィールド）のうちの１つの名前であり得る。探索クエリは、ルートとサブタイプの両方を、値として指定し得る。一例では、フィルタリングロジックは以下の通りであり得る。

（ｒｏｏｔｔｙｐｅ＿１ ＯＲ ｒｏｏｔｔｙｐｅ＿２ ＯＲ．．．ＯＲ ｒｏｏｔｔｙｐｅ＿Ｎ）ＡＮＤ（ｓｕｂｔｙｐｅ＿１ ＯＲ ｓｕｂｔｙｐｅ＿２ ＯＲ．．．ＯＲ ｓｕｂｔｙｐｅ＿Ｎ）

【００５０】

[0054] 別の例示的な基準は、要素プロパティである。これは、キー値ペアのセット“ｐｒｏｐｅｒｔｙ’ｓ ｎａｍｅ－ｐｒｏｐｅｒｔｙ’ｓ ｖａｌｕｅ”を指定し得る。要

素は、各ペアについて、ａ）要素が、指定された名前の「ｓｉｍｐｌｅ」プロパティを含むこと、および、ｂ）文字列へ変換されたこのプロパティの値が、指定された値を含むことが真であれば、基準を満足していると考慮される。カンマ区切りされた各値は、ｐｒｏｐｅｒｔｙ＿ｎａｍｅ＝ｐｒｏｐｅｒｔｙ＿ｖａｌｕｅの形式であり得る。

【００５１】

[0055] ブロック２５４では、１つまたは複数のタイプベースの探索構成要素（たとえば、構成要素１２４および／または１２６）が、識別された要素タイプに基づいてインスタンス化される。たとえば、これは、パーサブロック２５２からのタイプフィルタ基準に基づいて、ブロック２５６において（たとえば、タイプベースの探索構成要素ストア１２８からの）タイプ情報にアクセスすることによって実行され得る。ブロック２５６は、限定されないが、タイプがどのプロパティを含んでいるか、要素タイプの子要素のタイプ、要素タイプからどの要素タイプが由来しているか、および、要素タイプのためのプロパティゲッタの実施、を含む要素タイプに関する情報を提供する。

【００５２】

[0056] 一例では、ブロック２５４は、タイプ基準を、プロパティ基準に一致させるために、探索オプションを処理するためにブロック２５６によって提供されたタイプ情報を使用する。探索基準が、１つまたは複数のプロパティを含んでいるのであれば、タイプ情報ブロック２５４を使用することは、探索されたプロパティを含むことができないすべての要素タイプをフィルタアウトし得る。

【００５３】

[0057] 探索されるべき各要素タイプについて、対応するタイプベースの探索構成要素が、コード生成器１３０によって生成されたコードに従ってインスタンス化される。

【００５４】

[0058] ブロック２５８では、モデルストア１０８における要素に対する参照が、ブロック２５４からの探索基準と、ブロック２５６においてインスタンス化された意味論的な探索構成要素とに従って取得される。たとえば、メタデータ要素参照は、ルート要素の名前を取得すること（ストレージアクセスに関係しない迅速な動作）、および／または、要素をロードすること（ストレージアクセスに関係のある比較的長い動作）を容易にし得る。

【００５５】

[0059] ブロック２６０では、ブロック２５８で取得された要素参照は、たとえば、要素の名前または他のヒューリスティックスの指定された基準に基づいて、チャンク（chunk）へ優先付けられる。たとえば、探索された名前のうちの何れかを含む名前を有するルート要素は、この名前を含まないルート要素の前に処理されるであろう。

【００５６】

[0060] ブロック２６２は、モデルストア１０８における特定の要素が探索基準を満足するのであれば、モデルストア１０８における特定の要素を処理する。一例では、要素のタイプが、ブロック２５４において指定された、要求されたタイプ、すなわち、要素の名前が、要求された名前またはその一部のうちの１つを含む、探索基準によって指定された各ペア“ｐｒｏｐｅｒｔｙ’ｓ　ｎａｍｅ－ｐｒｏｐｅｒｔｙ’ｓ　ｖａｌｕｅ”について、要素が、そのような名前を持つプロパティを含んでいることが真である、および、このプロパティの値が、指定されたプロパティの値を含む、のうちの１つであれば、要素は、探索基準を満足していると考慮される。

【００５７】

[0061] 一例では、ブロック２６２は、ブロック２６４から、要素述語関数または他の情報を取得する。これは、要素が探索基準を満足しているか否かを判定するために使用される。ブロック２６４は、２６２によって提供された要素タイプの各々の述語関数を、ブロック２５６からの情報を使用して生成する。たとえば、ブロック２６４は、ブロック２５６へ要素タイプを提供し、各要素タイプのためのプロパティゲッタの実施に関する情報を受け取る。

【００５８】

[0062] モデルストア１０８における要素が、探索基準を満足するのであれば、その結果は、インターフェースブロック２５０を介して開発者２６０へ提供される。

【００５９】

[0063] 図４は、意味論的な探索構成要素を使用して、探索を実行するためのメソッド３００の一例を例示するフロー図である。例示のために、限定によってではなく、メソッド３００は、アーキテクチャ１００がタイプベースの探索構成要素を使用して探索を実行するコンテキストにおいて説明されるであろう。

【００６０】

[0064] ステップ３０２では、たとえばユーザインターフェースディスプレイ１１６を使用して、開発外観が表示される。ステップ３０４では、探索入力が受け取られ、ステップ３０６では、探索基準を識別するために探索入力がパースされる。探索基準の例は、限定されないが、タイプベースの制約、メソッド名、およびプロパティ値を含む。その後、探索モジュール１２０は、探索基準を満足する要素を求めて、モデルストア１０８を探索する。

【００６１】

[0065] ステップ３０８では、モデルストア１０８を探索するために、１つまたは複数のタイプベースの探索構成要素が識別されインスタンス化される。たとえば、図３に関して上記で議論されたように、タイプベースの探索制約は、探索入力において明示的に定義され得る。別の例では、タイプベースの探索制約が、探索入力において提供されたトークンから推論され得る。たとえば、探索入力において提供されたプロパティ値について、ステップ３０８は、どの要素タイプが、対応するプロパティを有しているのかを判定し得る。

【００６２】

[0066] その後、１つまたは複数のタイプベースの探索構成要素は、探索クエリに基づいてモデルストア１０８における要素を探索するために、探索モジュール１２０によってインスタンス化される。一例では、対応する要素タイプの各要素について、探索構成要素の個別の事例が生成される。

【００６３】

[0067] インスタンス化された探索構成要素は、ステップ３１０において、モデルストア１０８内の要素を探索するため、および、ステップ３１２において、ステップ３０６から識別された基準を満足する要素を識別するために使用される。上記で議論されたように、一例では、探索構成要素は、開発者１０６によって開発された要素をダイレクトに探索するのではなく、シリアル化された要素の表現（たとえば、XMLファイル）を探索し得る。

【００６４】

[0068] 例によって、限定によってではなく、モデルストア１０８におけるシリアル化された要素の表現を探索している間、探索構成要素は、シリアル化された表現における対応する要素に対する参照（たとえば、行および列番号位置）を発見することによって、探索基準を満足する要素の部分を識別する。探索構成要素は、コードをメタデータと区別し、シリアル化された表現において識別された一致について、あたかも開発者へ提示された統合されたコードビューを探索したかのように、コードにおける位置を計算する。したがって、開発者１０６の観点から、探索モジュール１２０は、シリアル化された表現ではなく、コードエディタ、および／または、メタデータエディタビュー内で探索し、結果を返す。

【００６５】

[0069] 一例では、探索構成要素は、この要素を読み取り、オブジェクト指向表現へ変換し、そのプロパティゲッタを適用して、プロパティを識別し、探索クエリからのプロパティベースの探索基準に対して一致させる。探索構成要素は、一致されたプロパティを、オブジェクト内のプロパティをユニークに識別する対応するパスへ変換する。たとえば、このパスは、ユニフォームリソース識別子（URI）を備える。これは、上記で議論されたように、メタデータパスであり得る。

【００６６】

[0070] ステップ３１４では、探索基準に一致する要素を示すリンクのセットとして結果が返される。たとえば、探索構成要素は、対応するＵＲＩを、探索モジュール１２０のアグリゲータ構成要素へ返すことによって要素一致を識別する。アグリゲータＵＲＩは、開発者１０６へ提示されるために、ユーザインターフェースモジュール１１２へ提供される。

【００６７】

[0071] ステップ３１６では、特定のＵＲＩの開発者１０６による選択が、たとえば、マウスクリックまたは他のユーザ入力のようなユーザインタラクションを介して、受け取られる。ナビゲーションモジュール１２２は、対応する要素位置を識別するために、選択されたＵＲＩを復号する。一例では、ＵＲＩは、別のプロパティへの参照（たとえば、"source=crosscompany"）を備える。ＵＲＩの選択は、ＵＲＬによって識別された場所においてメタデータエディタを開く。別の例では、ＵＲＩは、値を含むメソッドボディへの参照を含む。ここでは、ＵＲＩの選択は、コードエディタを開く。

【００６８】

[0072] 一例では、探索結果は、非同期的に取得および表示される。これは、矢印３２０によって図４において表される。すなわち、ステップ３１２において、インスタンス化された意味論的な探索構成要素が、探索基準を満足する要素を識別すると、識別された要素のＵＲＩが開発者へ表示される一方、探索はバックグランドで継続する。

【００６９】

[0073] 図５は、開発者１０６が、アーキテクチャ１００を使用して、アプリケーション要素を開発し、探索を実行する開発外観を提供するユーザインターフェースディスプレイ４００の一例を例示する。例示のために、限定によってではなく、ユーザインターフェースディスプレイ４００は、アーキテクチャ１００のコンテキストで説明されるであろう。

【００７０】

[0074] ユーザインターフェースディスプレイ４００は、アプリケーション要素１０７をオーサリングするための開発者入力を受け取るコードエディタビュー４０２と、開発者探索クエリを受け取る意味論的な探索インターフェース４０４とを含む。例によって、以下の探索クエリが、要素４０４に入力された。

ｔｙｐｅ：ｔａｂｌｅ，ｍｅｔｈｏｄ ｎａｍｅ：ｉｎｓｅｒｔ ｐｒｏｐｅｒｔｙ：“ｓｏｕｒｃｅ＝ｃｒｏｓｓｃｏｍｐａｎｙ”

【００７１】

[0075] 図３に関して説明された例示的な構文を使用して、探索クエリは、「ｔａｂｌｅ」であるタイプフィルタ、「ｉｎｓｅｒｔ」であるメソッド名フィルタ、および「ｃｒｏｓｓｃｏｍｐａｎｙ」であるプロパティ名フィルタを指定する。探索モジュール１２０は、テーブルタイプに対応するタイプベースの探索構成要素をインスタンス化する。探索クエリは、非同期的に実行され得、これは、探索結果ＵＲＩが取得されると、結果ウィンドウ４０６に、探索結果ＵＲＩを入力する。すなわち、この探索は、１つまたは複数の探索結果ＵＲＩをウィンドウ４０６内に表示させることによって始まり、その後、追加の探索結果ＵＲＩが取得されると、ウィンドウ４０６へ、追加の探索結果ＵＲＩを追加する。このように、開発者１０６は、ビュー４０２を、対応する探索結果へ向けるために、たとえば所望されるＵＲＩをクリックすることによって、ユーザインターフェースディスプレイ４００とインタラクトし続け、その間、探索は、任意の追加の結果を返すためにバックグランドで実行し続ける。例示された例では、各ＵＲＩは、ラベル情報４０８と、要素および要素の場所を識別する場所情報４１０とを含む。

【００７２】

[0076] 一例では、探索モジュール１２０の探索能力が、アプリケーションプログラミングインターフェース（API）として、探索クエリ構文とは独立したオブジェクトモデルとともに表される。ＡＰＩを使用して、探索動作は、複数の異なるデバイスのうちの何れかから（たとえば、アクセス権利およびセキュリティに従って）遠隔的に使われ得るネットワークにおけるサービスとして起動され得る。探索ＡＰＩのためのパラメータは、オブジ

ェクトモデルにおけるオブジェクトであり、構文に一致させるためのクエリ文字列ではない。したがって、探索クエリ構文は、探索部からデカップルされる。

【００７３】

[0077] 例によって、オブジェクトモデルのためのクラス図解は、複数の異なるクラスを含み得る。各クラスは、１つまたは複数の意味的な探索制約と、対応する要素を探索および検査するために呼び出されるべきメソッドとを定義する。オブジェクトモデルクラスによって定義された意味論的な探索制約の例は、限定されないが、タイプ制約、プロパティ制約、コード制約、および名前制約を含む。

【００７４】

[0078] 図６は、探索ＡＰＩを使用して探索結果をレンダリングする例示的なユーザインターフェース４５０を例示する。ユーザインターフェース４５０は、探索パラメータを定義する探索クエリを受け取るクエリ入力フィールド４５２と、探索モジュールから返された対応するクエリ結果を表示するクエリ結果フィールド４５４とを含む。例示された例では、探索パラメータは、コード制約クラスを含み、コード制約のための文字列（すなわち、「選択中」）を識別する。コード制約クラスは、文字列を一致させ、探索を優先度付ける等のためのメソッドを含む。探索モジュールは、コード制約クラスのオブジェクトをインスタンス化し、モデルストアに対する探索を実行する。

【００７５】

[0079] 一例では、探索が開始されるデバイスに依存して、異なる構文が提供され得る。たとえば、開発者は、より大きなフォーム・ファクタ・スクリーン（form factor screen）を備えた開発者デスクトップコンピュータから、フォーマルな構文でクエリ文字列を入力することを可能とされ得る。一方、フォーマルな構文における入力は、より小さなフォーム・ファクタを備えたモバイルデバイスからは、開発者にとってより困難であり得る。探索アーキテクチャは、より単純な形式でクエリ入力を容易にするように構成され得る。たとえば、モバイルデバイス等を使用する場合、開発者は、特定の探索制約（たとえば、特定のタイプベースの探索）のセットに割り当てられたボタンのようなあらかじめ定義された探索機能を有する制御とともに提示され得る。

【００７６】

[0080] 本議論は、プロセッサおよびサーバに言及する。一例では、プロセッサおよびサーバは、個別に図示されていないが、関連付けられたメモリおよびタイミング回路を備えたコンピュータプロセッサを含む。これらは、これらが属するシステムまたはデバイスの機能部分であり、これらシステムにおける他のモジュール、構成要素、および／または、アイテムの機能性によって起動され、これらシステムにおける他のモジュール、構成要素の機能、および／または、アイテムの機能性を容易にする。

【００７７】

[0081] また、多くのユーザインターフェースディスプレイが議論された。それらは、種々様々な異なる形式を採り得、そこに配置された種々様々な異なるユーザ起動可能な入力メカニズムを有し得る。たとえば、ユーザ起動可能な入力メカニズムは、テキストボックス、チェックボックス、アイコン、リンク、ドロップダウンメニュー、探索ボックス等であり得る。これらはまた、種々様々な異なる方式で起動され得る。たとえば、これらは、（トラックボールまたはマウスのような）ポイントおよびクリックデバイスを使用して起動され得る。これらは、ハードウェアボタン、スイッチ、ジョイスティックまたはキーボード、サムスイッチ、またはサムパッド等を使用して起動され得る。これらはまた、仮想的なキーボードまたは他の仮想的なアクチュエータを使用して起動され得る。それに加えて、これらが表示されるスクリーンがタッチセンサ式スクリーンである場合、これらはタッチジェスチャを使用して起動され得る。また、これらを表示するデバイスが音声認識構成要素を有する場合、これらは音声コマンドを使用して起動され得る。

【００７８】

[0082] 多くのデータストアも議論された。これらは各々の、多数のデータストアへ分割され得ることが注目されるであろう。すべてが、これらにアクセスするシステムに局所的

であり得、すべてが、遠隔的であり得、または、いくつかが、局所的であり得る一方、他が、遠隔的である。本明細書では、これらの構成のすべてが考慮される。

【００７９】

[0083] また、図面は、各ブロックへ割り当てられた機能性を備えた多くのブロックを図示する。より少ない構成要素によって機能性が実行されるように、より少ないブロックが使用され得ることが注目されるであろう。また、より多くのブロックが、より多くの構成要素にわたって分散された機能とともに使用され得る。

【００８０】

[0084] 図７は、その要素がクラウドコンピューティングアーキテクチャ５００に配置されていることを除いた、図１に図示されたアーキテクチャ１００のブロック図である。クラウドコンピューティングは、サービスを提供するシステムの物理的な場所または構成をエンドユーザが知ることを必要としない計算、ソフトウェア、データアクセス、およびストレージサービスを提供する。様々な例において、クラウドコンピューティングは、適切なプロトコルを使用して、インターネットのような広域ネットワークを介してサービスを提供する。たとえば、クラウドコンピューティングプロバイダは、広域ネットワークを介してアプリケーションを提供し、これらは、ウェブブラウザまたは他の任意のコンピューティング構成要素を介してアクセスされ得る。アーキテクチャ１００のソフトウェア、モジュール、または構成要素のみならず、対応するデータが、遠隔場所におけるサーバに記憶され得る。クラウドコンピューティング環境におけるコンピューティングリソースは、遠隔のデータセンタ場所において統合され得るか、または、分散され得る。クラウドコンピューティングインフラストラクチャは、たとえユーザにとって単一のアクセスのポイントとして見えようとも、共有データセンタを介してサービスを提供し得る。したがって、本明細書で説明されたモジュール、構成要素、および機能は、クラウドコンピューティングアーキテクチャを使用して遠隔場所におけるサービスプロバイダから提供され得る。あるいは、これらは、従来式のサーバから提供され得るか、または、クライアントデバイスにダイレクトに、または他の方式でインストールされ得る。

【００８１】

[0085] 本説明は、公衆のクラウドコンピューティングおよび個人のクラウドコンピューティングの両方を含むことが意図される。クラウドコンピューティング（公衆および個人の両方）は、実質的にシームレスなリソースのプールのみならず、根底をなすハードウェアインフラストラクチャを管理および構成するための低減された必要性を提供する。

【００８２】

[0086] 公衆のクラウドは、ベンダによって管理され、典型的には、同じインフラストラクチャを使用して多くのコンシューマをサポートする。また、公衆のクラウドは、個人のクラウドとは逆に、エンドユーザを、ハードウェアを管理することから解放し得る。個人のクラウドは、組織自身によって管理され得、インフラストラクチャは、典型的には、他の組織と共有されない。組織はまだ、インストールおよび修理等のように、ハードウェアをある程度維持する。

【００８３】

[0087] 図７に図示される例では、いくつかのアイテムは、図１に図示されるこれらに類似しており、これらは同様に付番されている。図７は特に、インタラクティブ開発システム１０２、モデルストア１０８、探索構成要素ストア１２８、および探索構成要素コード生成器１３０が、（公衆、個人、または、一部が公衆であり他が個人である組合せであり得る）クラウド５０２に配置され得る。したがって、開発者１０６は、クラウド５０２を介してこれらシステムへアクセスするためにユーザデバイス５０４を使用する。

【００８４】

[0088] 図７はまた、クラウドアーキテクチャの別の例を図示する。図７は、アーキテクチャ１００のいくつかの要素がクラウド５０２に配置され、他は配置されないことも考慮されることを図示する。例によって、モデルストア１０８は、クラウド５０２の外部に配置され得、クラウド５０２を介してアクセスされ得る。別の例では、探索構成要素ストア

１２８もまた、クラウド５０２の外部であり得る。別の例において、探索構成要素コード生成器１３０もまた、クラウド５０２の外部であり得る。これらがどこに配置されているかに関わらず、これらは、（広域ネットワークまたはローカルエリアネットワークの何れかである）ネットワークを介してデバイス５０４によってダイレクトにアクセスされ得、これらは、サービスによって遠隔サイトにおいてホストされ得るか、または、クラウドを介したサービスとして提供され得るか、または、クラウドに存在する接続サービスによってアクセスされ得る。これらアーキテクチャのすべてが本明細書で考慮される。

【００８５】

[0089] アーキテクチャ１００またはその一部は、種々様々な異なるデバイス上に配置され得ることもまた注目されるであろう。これらのデバイスのうちのいくつかは、サーバと、デスクトップコンピュータと、ラップトップコンピュータと、タブレットコンピュータと、または、パームトップコンピュータ、セル電話、スマートフォン、マルチメディアプレーヤ、携帯情報端末等のような他のモバイルデバイスとを含む。

【００８６】

[0090] 図８は、本システム（またはその一部）が展開され得る、ユーザのまたはクライアントのハンドヘルドデバイス１６として使用され得るハンドヘルドデバイスまたはモバイルコンピューティングデバイスの一例の簡略ブロック図である。図９～図１２は、ハンドヘルドデバイスまたはモバイルデバイスの例である。

【００８７】

[0091] 図８は、アーキテクチャ１００のモジュールまたは構成要素を実行し得る、または、アーキテクチャ１００とインタラクトする、またはその両方を行うクライアントデバイス１６の構成要素の一般的なブロック図を提供する。デバイス１６では、ハンドヘルドデバイスが他のコンピューティングデバイスと通信することを可能にし、いくつかの例では、たとえばスキャニングによって、情報を自動的に受信するためのチャネルを提供する、通信リンク１３が提供される。通信リンク１３の例は、赤外線ポート、シリアル／ＵＳＢポート、イーサネットポートのようなケーブルネットワークポート、および、ネットワークへセルラアクセスを提供するために使用されるワイヤレスサービスである汎用パケット無線サービス（GPRS）、ＬＴＥ、ＨＳＰＡ、ＨＳＰＡ＋、および他の３Ｇおよび４Ｇ無線プロトコル、ｌＸｒｔｔ、およびショートメッセージサービスのみならず、ネットワークへローカルワイヤレス接続を提供する８０２．１１および８０２．１１ｂ（Wi-Fi）プロトコル、およびＢｌｕｅｔｏｏｔｈプロトコルを含む１つまたは複数の通信プロトコルを介した通信を可能にするワイヤレスネットワークポートを含む。

【００８８】

[0092] 他の例では、アプリケーションまたはシステムは、セキュアデジタル（SD）カードインターフェース１５へ接続されたリムーバブルなＳＤカードにおいて受け取られる。ＳＤカードインターフェース１５および通信リンク１３は、メモリ２１および入力／出力（I/O）構成要素２３のみならず、クロック２５および位置決めシステム２７にも接続されているバス１９に沿ってプロセッサ１７（これは、図１からのプロセッサ１１０をも具体化し得る）と通信する。

【００８９】

[0093] Ｉ／Ｏ構成要素２３は、一例では、入力操作および出力操作を容易にするように提供される。デバイス１６の様々な例のためのＩ／Ｏ構成要素２３は、ボタン、タッチセンサ、マルチタッチセンサ、光またはビデオセンサ、音声センサ、タッチスクリーン、近接センサ、マイクロホン、チルトセンサ、および重力スイッチのような入力構成要素と、ディスプレイデバイス、スピーカ、およびまたはプリンタポートのような出力構成要素とを含み得る。他のＩ／Ｏ構成要素２３もまた同様に使用され得る。

【００９０】

[0094] クロック２５は、時間と日付を出力するリアルタイムクロック構成要素を備える。それはプロセッサ１７のためのタイミング機能をも提供し得る。

【００９１】

[0095] 位置決めシステム２７は、デバイス１６の現在の地理的位置を出力する構成要素を含む。これは、たとえば、全地球測位システム（GPS）受信機、ＬＯＲＡＮシステム、推測航法システム、セルラ三角測量システム、または他の測位システムを含み得る。それはたとえば、所望される地図、ナビゲーションルート、および他の地理的機能を生成するマッピングソフトウェアまたはナビゲーションソフトウェアをも含み得る。

【００９２】

[0096] メモリ２１は、オペレーティングシステム２９、ネットワーク設定３１、アプリケーション３３、アプリケーション構成設定３５、データストア３７、通信ドライバ３９、および通信構成設定４１を記憶する。それはまた、アーキテクチャ１００の一部またはすべてであり得るクライアントシステム２４を記憶し得る。メモリ２１は、すべてのタイプの有形的な揮発性および不揮発性のコンピュータ読取可能なメモリデバイスを含み得る。それはまた、（以下に説明される）コンピュータ記憶媒体をも含み得る。メモリ２１は、プロセッサ１７によって実行された場合、プロセッサに対して、コンピュータ実施されるステップまたは機能を、命令に従って実行させるコンピュータ読取可能な命令を記憶する。プロセッサ１７は、それらの機能性を容易にするために他のモジュールまたは構成要素によっても活性化され得る。

【００９３】

[0097] ネットワーク設定３１の例は、プロキシ情報、インターネット接続情報、およびマッピングのようなものを含む。アプリケーション構成設定３５は、特定の企業またはユーザのためのアプリケーションを調整する設定を含む。通信構成設定４１は、他のコンピュータと通信するためのパラメータを提供し、ＧＰＲＳパラメータ、ＳＭＳパラメータ、接続ユーザ名、およびパスワードのようなアイテムを含む。

【００９４】

[0098] アプリケーション３３は、デバイス１６に以前に記憶されていたアプリケーション、または、使用中にインストールされたアプリケーションであり得るが、これらは、オペレーティングシステム２９の一部であり得るか、またはデバイス１６に対して外部にも同様にホストされ得る。

【００９５】

[0099] 図９は、デバイス１６がタブレットコンピュータ６００である一例を図示する。図９では、コンピュータ６００は、ユーザインターフェースディスプレイスクリーン６０２を備えて図示される。スクリーン６０２は、タッチスクリーン（したがって、アプリケーションとインタラクトするために、ユーザの指からのタッチジェスチャが使用され得る）、または、ペンまたはスタイラスから入力を受け取るペン対応インターフェースであり得る。それはまた、スクリーン上の仮想的なキーボードをも使用し得る。もちろん、それはまた、たとえばワイヤレスリンクまたはＵＳＢポートのような適切なアタッチメントメカニズムを介してキーボードまたは他のユーザ入力デバイスへ取り付けられ得る。コンピュータ６００はまた、音声入力をも受け取り得る。

【００９６】

[00100] 図１０および図１１は、使用され得るデバイス１６の追加の例を提供するが、他のものもまた同様に使用され得る。図１０では、フィーチャフォン、スマートフォン、またはモバイル電話４５が、デバイス１６として提供される。電話４５は、電話番号をダイヤルするためのキーパッド４７のセットと、アプリケーション画像、アイコン、ウェブページ、写真、およびビデを含む画像を表示することが可能なディスプレイ４９と、ディスプレイ上に図示されたアイテムを選択するための制御ボタン５１とを含む。電話は、汎用パケットラジオサービス（GPRS）および１Ｘｒｔｔのようなセルラ電話信号と、ショートメッセージサービス（SMS）信号とを受信するためのアンテナ５３を含む。いくつかの例では、電話４５はまた、セキュアデジタル（SD）カード５７を受け取るＳＤカードスロット５５をも含む。

【００９７】

[00101] 図１１のモバイルデバイスは、携帯情報端末（PDA）５９またはマルチメディア

プレーヤまたはタブレットコンピューティングデバイス等（以下ＰＤＡ５９と称される）である。ＰＤＡ５９は、スタイラス６３がスクリーンの上方に位置された場合、スタイラス６３（または、ユーザの指のような他のポインタ）の位置を感知する誘導スクリーン６１を含む。これによって、ユーザは、スクリーン上のアイテムの選択、強調、および移動のみならず、描画、および書込も可能となる。ＰＤＡ５９はまた、多くのユーザ入力キーまたはボタン（たとえば、ボタン６５）を含む。これによって、ユーザは、ディスプレイ６１上に表示されたメニューオプションまたは他のディスプレイオプションをスクロールすることが可能となり、また、ユーザは、ディスプレイ６１に触れることなく、アプリケーションを変更したり、または、ユーザ入力機能を選択することが可能となる。図示されないが、ＰＤＡ５９は、他のコンピュータとのワイヤレス通信を可能にする内部アンテナおよび赤外線送信機／受信機のみならず、他のコンピューティングデバイスへのハードウェア接続を可能にする接続ポートを含み得る。そのようなハードウェア接続は典型的には、シリアルまたはＵＳＢポートを介して他のコンピュータへ接続するクレードルによってなされる。そのため、これらの接続は、非ネットワーク接続である。一例では、モバイルデバイス５９はまた、ＳＤカード６９を受け取るＳＤカードスロット６７を含む。

【００９８】

[00102] 図１２は、電話がスマートフォン７１であることを除いて図１０に類似している。スマートフォン７１は、アイコンまたはタイルまたは他のユーザ入力メカニズム７５を表示するタッチセンサ式ディスプレイ７３を有する。メカニズム７５は、アプリケーションの実行、通話、データ転送操作の実行等のためにユーザによって使用され得る。一般に、スマートフォン７１は、モバイルオペレーティングシステム上にビルドされ、フィーチャフォンよりもより進んだコンピューティング能力および接続性を提供する。

【００９９】

[00103] 他の形式のデバイス１６が可能であることに注目されたい。

【０１００】

[00104] 図１３は、アーキテクチャ１００、またはその一部が（たとえば）展開され得るコンピューティング環境の一例である。図１３を参照して示すように、いくつかの例を実施するための典型的なシステムは、コンピュータ８１０の形式である汎用コンピューティングデバイスを含む。コンピュータ８１０の構成要素は、限定されないが、（プロセッサ１１０を備え得る）処理ユニット８２０、システムメモリ８３０、および、システムメモリを含む様々なシステム構成要素を処理ユニット８２０へ結合するシステムバス８２１を含み得る。システムバス８２１は、メモリバスまたはメモリコントローラ、周辺バス、および、種々のバスアーキテクチャのうちの何れかを使用したローカルバスを含むいくつかのタイプのバス構造のうちの何れかであり得る。例によって、限定することなく、そのようなアーキテクチャは、業界標準アーキテクチャ（ISA）バス、マイクロチャネルアーキテクチャ（MCA）バス、エンハンストＩＳＡ（EISA）バス、ビデオエレクトロニクス標準化団体（VESA）ローカルバス、および、Ｍｅｚｚａｎｉｎｅバスとしても知られている周辺機器インターコネクト（PCI）バスを含む。図１に関して説明されたメモリおよびプログラムは、図１３の対応する部分において展開され得る。

【０１０１】

[00105] コンピュータ８１０は、典型的には、様々なコンピュータ読取可能な媒体を含む。コンピュータ読取可能な媒体は、コンピュータ８１０によってアクセスされ得、揮発性および不揮発性の媒体、リムーバブルおよび非リムーバブルの媒体の両方を含む利用可能な任意の媒体であり得る。例によって、限定せず、コンピュータ読取可能な媒体は、コンピュータ記憶媒体および通信媒体を備え得る。コンピュータ記憶媒体は、変調されたデータ信号または搬送波とは異なり、変調されたデータ信号または搬送波を含まない。それは、コンピュータ読取可能な命令、データ構造、プログラムモジュール、または他のデータのような情報の記憶のための任意の方法または技術において実施される揮発性および不揮発性の、リムーバブルおよび非リムーバブルの両方の媒体を含むハードウェア記憶媒体を含む。コンピュータ記憶媒体は、限定されないが、ＲＡＭ、ＲＯＭ、ＥＥＰＲＯＭ、フ

ラッシュメモリまたは他のメモリ技術、ＣＤ－ＲＯＭ、デジタル多用途ディスク（DVD）または他の光ディスクストレージ、磁気カセット、磁気テープ、磁気ディスクストレージまたは他の磁気記憶デバイス、または、所望された情報を記憶するために使用され得、コンピュータ８１０によってアクセスされ得る他の任意の媒体、を含む。通信媒体は典型的には、伝送メカニズムにおいて、コンピュータ読取可能な命令、データ構造、プログラムモジュール、または他のデータを具体化し、任意の情報伝送媒体を含む。「変調されたデータ信号」という用語は、信号における情報をエンコードするような方式で設定または変更されたその特性のうちの１つまたは複数を有する信号を意味する。例によって、限定することなく、通信媒体は、ワイヤネットワークまたはダイレクトワイヤ接続のようなワイヤ媒体、および、音響、ＲＦ、赤外線、および他のワイヤレス媒体のようなワイヤレス媒体を含む。上記のうちの任意の組合せもまた、コンピュータ読取可能な媒体の範囲内に含まれるべきである。

【０１０２】

[00106] システムメモリ８３０は、読取専用メモリ（ROM）８３１およびランダムアクセスメモリ（RAM）８３２のような揮発性および／または不揮発性メモリの形式のコンピュータ記憶媒体を含む。起動中のような、コンピュータ８１０内の要素間の情報の転送を助ける基本ルーチンを含む基本入力／出力システム８３３（BIOS）は、典型的に、ＲＯＭ８３１に記憶される。ＲＡＭ８３２は典型的には、処理ユニット８２０に直ちにアクセス可能な、および／または、処理ユニット８２０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を含む。例によって、限定ではなく、図１３は、オペレーティングシステム８３４、アプリケーションプログラム８３５、他のプログラムモジュール８３６、およびプログラムデータ８３７を例示する。

【０１０３】

[00107] コンピュータ８１０はまた、他のリムーバブル／非リムーバブルな揮発性／不揮発性のコンピュータ記憶媒体を含み得る。例のみによって、図１３は、非リムーバブルな不揮発性磁気媒体からの読取、または、この不揮発性磁気媒体への書込を行うハードディスクドライブ８４１と、ＣＤ　ＲＯＭまたは他の光学媒体のようなリムーバブルな不揮発性光ディスク８５６からの読取、または、この不揮発性光ディスク８５６への書込を行う光ディスクドライブ８５５とを例示する。典型的なオペレーティング環境において使用され得る他のリムーバブル／非リムーバブルな揮発性／不揮発性コンピュータ記憶媒体は、限定されないが、磁気テープカセット、フラッシュメモリカード、デジタル多用途ディスク、デジタルビデオテープ、ソリッドステートＲＡＭ、ソリッドステートＲＯＭ等を含む。ハードディスクドライブ８４１は典型的には、インターフェース８４０のような非リムーバブルなメモリインターフェースを介してシステムバス８２１へ接続され、光ディスクドライブ８５５は典型的には、インターフェース８５０のようなリムーバブルなメモリインターフェースによってシステムバス８２１へ接続される。

【０１０４】

[00108] あるいは、またはそれに加えて、本明細書で説明された機能性は、１つまたは複数のハードウェア論理構成要素によって少なくとも部分的に実行され得る。たとえば、限定なしで、使用され得るハードウェア論理構成要素のタイプは、フィールドプログラマブルゲートアレイ（FPGA）、特定プログラム向け集積回路（ASIC）、特定プログラム向け基準製品（ASSP）、システムオンチップシステム（SOC）、コンプレックスプログラマブル論理デバイス（CPLD）等を含む。

【０１０５】

[00109] 上記で議論され図１３に例示されたドライブおよびそれらの関連するコンピュータ記憶媒体は、コンピュータ読取可能な命令のストレージ、データ構造、プログラムモジュール、およびコンピュータ８１０のための他のデータを提供する。図１３では、たとえば、ハードディスクドライブ８４１は、オペレーティングシステム８４４、アプリケーションプログラム８４５、他のプログラムモジュール８４６、およびプログラムデータ８４７を記憶するとして例示される。これら構成要素は、オペレーティングシステム８３４

、アプリケーションプログラム８３５、他のプログラムモジュール８３６、およびプログラムデータ８３７と同じ、または、異なるかの何れかであり得る。オペレーティングシステム８４４、アプリケーションプログラム８４５、他のプログラムモジュール８４６、およびプログラムデータ８４７は、最小でも、これらが異なるコピーであることを例示するために、本明細書では異なる番号を付される。

【０１０６】

[00110] ユーザは、コマンドおよび情報を、キーボード８６２、マイクロホン８６３、および、マウスやトラックボールまたはタッチパッドのようなポインティングデバイス８６１のような入力デバイスを介してコンピュータ８１０へ入力し得る。他の入力デバイス（図示せず）は、ジョイスティック、ゲームパッド、衛星放送アンテナ、スキャナ等を含み得る。これらおよび他の入力デバイスはしばしば、システムバスへ結合されたユーザ入力インターフェース８６０を介して処理ユニット８２０へ接続されるが、他のインターフェースと、パラレルポート、ゲームポート、またはユニバーサルシリアルバス（USB）のようなバス構造とによって接続され得る。ビジュアルディスプレイ８９１または他のタイプのディスプレイデバイスも、ビデオインターフェース８９０のようなインターフェースを介してシステムバス８２１へ接続される。モニタに加えて、コンピュータはまた、スピーカ８９７およびプリンタ８９６のように、出力周辺インターフェース８９５を介して接続され得る他の周辺出力デバイスを含み得る。

【０１０７】

[00111] コンピュータ８１０は、遠隔コンピュータ８８０のような１つまたは複数の遠隔コンピュータへの論理的な接続を使用してネットワーク化された環境で操作される。遠隔コンピュータ８８０は、パーソナルコンピュータ、ハンドヘルドデバイス、サーバ、ルータ、ネットワークＰＣ、ピアデバイスまたは他の共通のネットワークノードであり得て、典型的には、コンピュータ８１０に関して上記説明された要素のうちの多くまたはすべてを含む。図１３に描写される論理的な接続は、ローカルエリアネットワーク（LAN）８７１および広域ネットワーク（WAN）７７３を含むが、他のネットワークをも含み得る。そのようなネットワーキング環境は、オフィス、企業ワイドな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において普通である。

【０１０８】

[00112] ＬＡＮネットワーク環境で使用される場合、コンピュータ８１０は、ネットワークインターフェースまたはアダプタの８７０を介してＬＡＮ８７１へ接続される。ＷＡＮネットワーク環境で使用される場合、コンピュータ８１０は典型的には、モデム８７２、または、インターネットのようなＷＡＮ８７３を介して通信を確立するための他の手段を含む。内部または外部であり得るモデム８７２は、ユーザ入力インターフェース８６０または他の適切なメカニズムを介してシステムバス８２１へ接続され得る。ネットワーク化された環境では、コンピュータ８１０またはその一部に関して描写されたプログラムモジュールは、遠隔メモリ記憶デバイスに記憶され得る。例によって、限定なく、図１３は、遠隔アプリケーションプログラム８８５を、遠隔コンピュータ８８０上に存在するものとして例示する。図示されたネットワーク接続は、例示的であり、コンピュータ間の通信リンクを確立する他の手段が使用され得ることが認識されるであろう。

【０１０９】

[00113] 本明細書で説明された異なる実施形態が、異なる方式で結合され得ることもまた注目されるべきである。すなわち、１つまたは複数の実施形態の一部が、１つまたは複数の他の実施形態の一部と結合され得る。このすべてが本明細書で考慮される。

【０１１０】

[00114] 例１は、ユーザ開発入力を感知し、ユーザ開発入力に基づいてコンピュータシステムの要素を変化させる開発モジュールを備える開発システムである。これら要素は、コンピュータシステムにおいてモデル化されたタイプを備える。ユーザインターフェースモジュールは、ユーザ入力メカニズムを用いてユーザインターフェースディスプレイを生成し、コンピュータシステムの要素を探索するためのユーザ探索クエリを示す、ユーザ入

力メカニズムを介して受け取られたユーザ探索入力を感知する。探索エンジンは、ユーザ探索クエリのためのタイプベースの探索パラメータを識別する。探索エンジンは、タイプベースの探索パラメータに基づいて、タイプベースの探索構成要素を活性化するように制御される。タイプベースの探索構成要素は、ユーザインターフェースディスプレイにおける探索結果のセットを返すために、要素探索を実行する。

【０１１１】

[00115] 例２は、任意またはすべての以前の例の開発システムであり、開発モジュールは、インタラクティブ開発環境（IDE）の一部である。

【０１１２】

[00116] 例３は、任意またはすべての以前の例の開発システムであり、ユーザは開発者であり、コンピュータシステムの要素は、開発者によってカスタマイズされたアプリケーション要素を備える。

【０１１３】

[00117] 例４は、任意またはすべての以前の例の開発システムであり、タイプベースの探索パラメータは、コンピュータシステムにおいてモデル化されたタイプから選択された特定の要素タイプを識別し、探索エンジンは、要素探索を、特定の要素タイプを有する要素へ制約するように制御される。

【０１１４】

[00118] 例５は、任意またはすべての以前の例の開発システムであり、ユーザ探索クエリは、キャラクタ文字列および特定の要素タイプを含む。

【０１１５】

[00119] 例６は、任意またはすべての以前の例の開発システムであり、探索結果のセットは、キャラクタ文字列に一致するプロパティ値を有する特定の要素タイプの要素を備える。

【０１１６】

[00120] 例７は、任意またはすべての以前の例の開発システムであり、コンピュータシステムの要素は、複数の異なるタイプを備え、各タイプは、その要素タイプの要素のためのランタイム挙動を定義するプロパティおよびメソッドのセットを有する。システムはさらに、複数の探索構成要素を記憶する探索構成要素ストアを備え、各探索要素は、異なるタイプのうちの所与の１つに対応し、所与のタイプの要素のプロパティおよびメソッドのセットを探索するように構成されている。

【０１１７】

[00121] 例８は、任意またはすべての以前の例の開発システムであり、探索エンジンは、特定の要素タイプに対応する探索構成要素ストアからタイプベースの探索構成要素を識別し、特定の要素タイプを有するコンピュータシステムの複数の要素の各々を識別し、識別された要素を、識別された探索構成要素を使用して、ユーザ探索クエリに基づいて探索する。

【０１１８】

[00122] 例９は、任意またはすべての以前の例の開発システムであり、識別された探索構成要素は、特定の要素タイプを有する複数の識別された要素の各々についてインスタンス化され、探索エンジンは、インスタンス化された複数の探索構成要素からの探索結果を集約することによって探索結果のセットを取得し、集約された探索結果を、ユーザインターフェースディスプレイにおいて表示する。

【０１１９】

[00123] 例１０は、任意またはすべての以前の例の開発システムであり、探索結果のセットが、非同期的に取得および表示される。

【０１２０】

[00124] 例１１は、任意またはすべての以前の例の開発システムであり、要素のコードおよびメタデータを備える要素のシリアル化された表現を、要素の各々について記憶するモデルストアをさらに備える。探索エンジンは、モデルストア内のシリアル化された表現

へアクセスすることによって要素探索を実行する。

【０１２１】

[00125] 例１２は、任意またはすべての以前の例の開発システムであり、探索エンジンは、要素のうちの所与の１つに対応するモデルストアにおける特定のシリアル化された表現を、タイプベースの探索パラメータに基づいて識別し、ユーザ探索クエリに基づいて、特定のシリアル化された表現を探索する。

【０１２２】

[00126] 例１３は、任意またはすべての以前の例の開発システムであり、探索エンジンは、ユーザ探索クエリに一致する所与の要素の一部を、特定のシリアル化された表現から識別し、所与の要素の一部をユニークに識別するパス情報を識別する。

【０１２３】

[00127] 例１４は、任意またはすべての以前の例の開発システムであり、パス情報は、ユニフォームリソース識別子（URI）を備え、ユーザインターフェースモジュールは、ＵＲＩのユーザ選択可能な表現を生成する。それは、エディタユーザインターフェースにおいて所与の要素の一部を示すために選択可能である。

【０１２４】

[00128] 例１５は、複数の異なる要素タイプをモデル化するデータストアと、開発者入力を感知し、開発者入力に基づいて、異なる要素タイプのアプリケーション要素を変化させる開発者モジュールと、データストアにおいてモデル化された要素タイプの各々のために異なる探索構成要素を生成する探索構成要素生成器とを備える開発システムである。開発システムはまた、複数の要素タイプのために探索構成要素生成器によって生成された探索構成要素を記憶する探索構成要素ストアと、ユーザ探索入力を感知する探索エンジンであって、要素タイプの所与の１つのアプリケーション要素を探索するために探索構成要素のうちの選択された１つを活性化するように制御される探索エンジンとを備える。

【０１２５】

[00129] 例１６は、任意またはすべての以前の例の開発システムであり、探索構成要素生成器は、要素タイプの所与の１つの構造を分析し、所与の要素タイプの構造に基づいて、対応する探索機能を生成することによって、各探索構成要素を生成するように構成される。

【０１２６】

[00130] 例１７は、任意またはすべての以前の例の開発システムであり、所与の要素タイプの構造は、所与の要素タイプを有する要素のランタイム挙動を定義するプロパティおよびメソッドのセットによって定義される。

【０１２７】

[00131] 例１８は、任意またはすべての以前の例の開発システムであり、探索エンジンは、少なくとも１つの探索用語を有する探索クエリを受け取り、探索クエリのためのタイプベースの探索パラメータを識別し、タイプベースの探索パラメータに基づいて、探索構成要素ストアから探索構成要素のうちの１つを識別する。識別された探索構成要素は、探索用語に基づいて、アプリケーション要素のうちの１つまたは複数を探索するためにインスタンス化される。

【０１２８】

[00132] 例１９は、コンピュータシステムの要素を開発し、これら要素の探索を制御するためのコンピュータによって実施される方法である。この方法は、開発者ユーザ入力を感知し、開発者ユーザ入力に基づいて、コンピュータシステムの要素を変化させるステップを含む。コンピュータシステムは、複数の異なる要素タイプを備え、各要素タイプは、要素タイプの要素のためのプロパティ構造によって定義されている。この方法は、探索インターフェースディスプレイを生成するステップと、コンピュータシステムの要素を探索するために、ユーザ探索クエリを示すユーザ入力を、探索インターフェースディスプレイを介して感知するステップとを含む。この方法は、ユーザ探索クエリと、コンピュータシステムの要素のプロパティ構造に基づく意味論的な探索制約とに基づいて、コンピュータ

システムの要素の探索を制御するステップを含む。この方法は、探索から探索結果を返すステップと、探索結果を表示する結果ディスプレイを生成するステップとを含む。

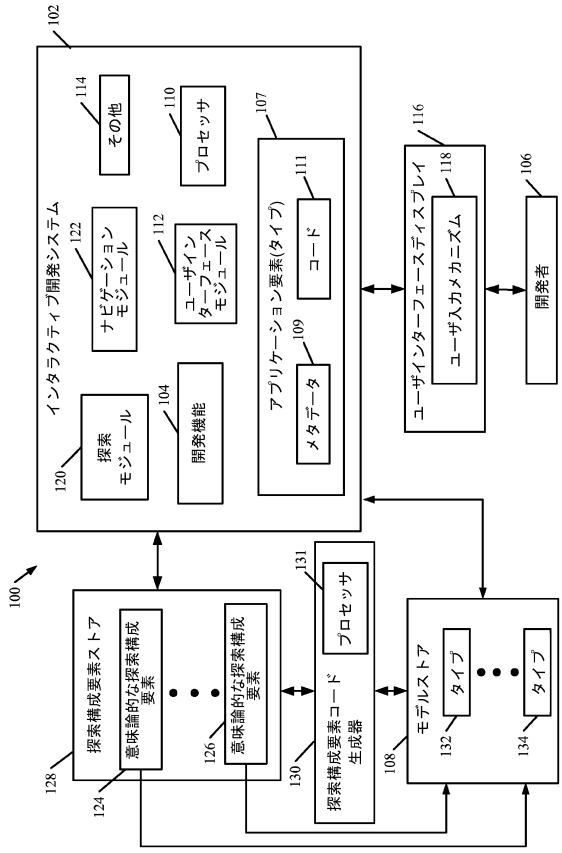
【０１２９】

[00133]　例２０は、任意またはすべての以前の例のコンピュータによって実施される方法であり、さらに、複数の異なる要素タイプをモデル化するデータストアへアクセスするステップと、異なる各要素タイプのために、要素タイプのプロパティ構造に基づいて、対応するタイプベースの探索構成要素を生成するステップとを含む。この方法は、ユーザ探索クエリに基づいてデータストアを探索するために、意味論的な探索制約に基づいて選択された、生成されたタイプベースの探索構成要素のうちの少なくとも１つを使用するステップを含む。

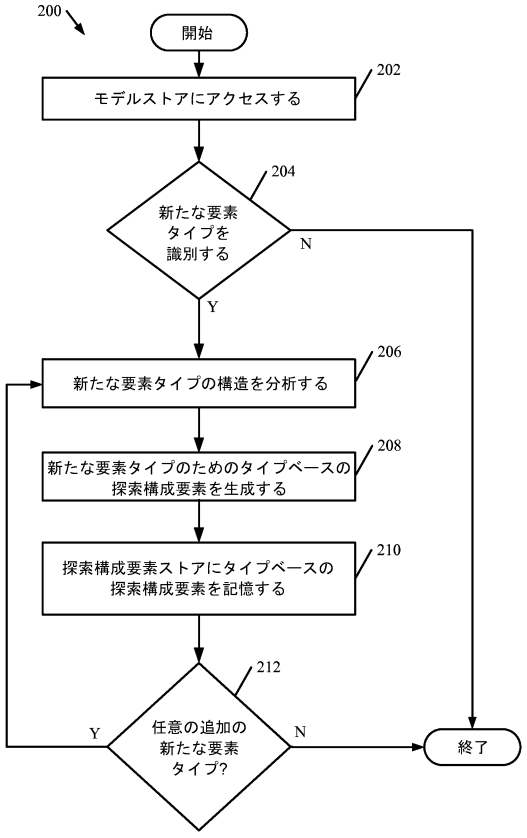
【０１３０】

[00134]　これら主題は、構造的な特徴および／または方法的な動作に特有の文言で記述されているが、添付された特許請求の範囲で定められた主題は、必ずしも上記説明された特定の特徴または動作に限定される必要はないことが理解されるべきである。むしろ、上記説明された特定の特徴および動作は、特許請求の範囲を実施する例示的な形式で開示され、他の均等な特徴および動作が、特許請求の範囲の範囲内にあるべきことが意図されている。

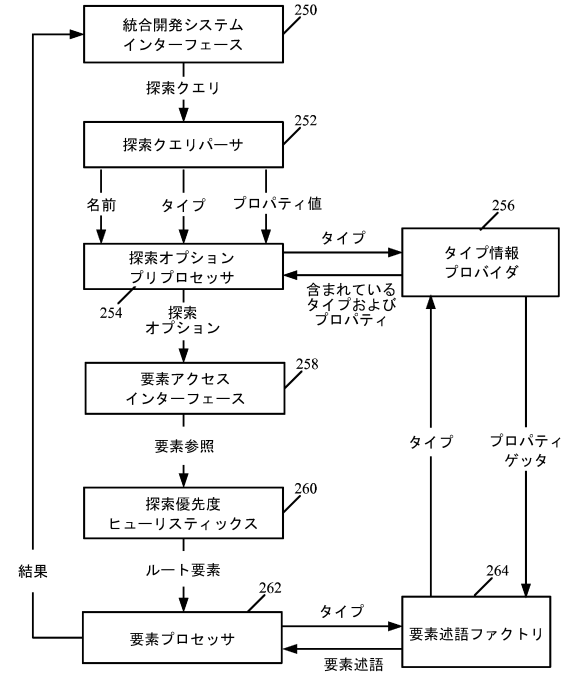
【図１】



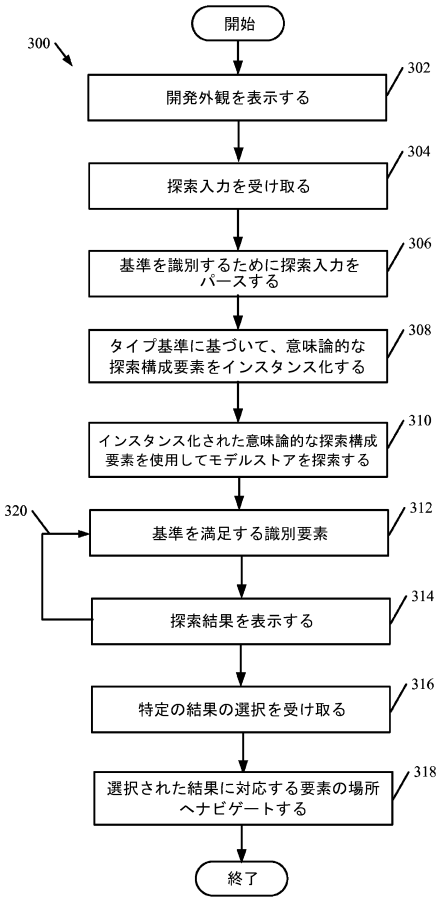
【図２】



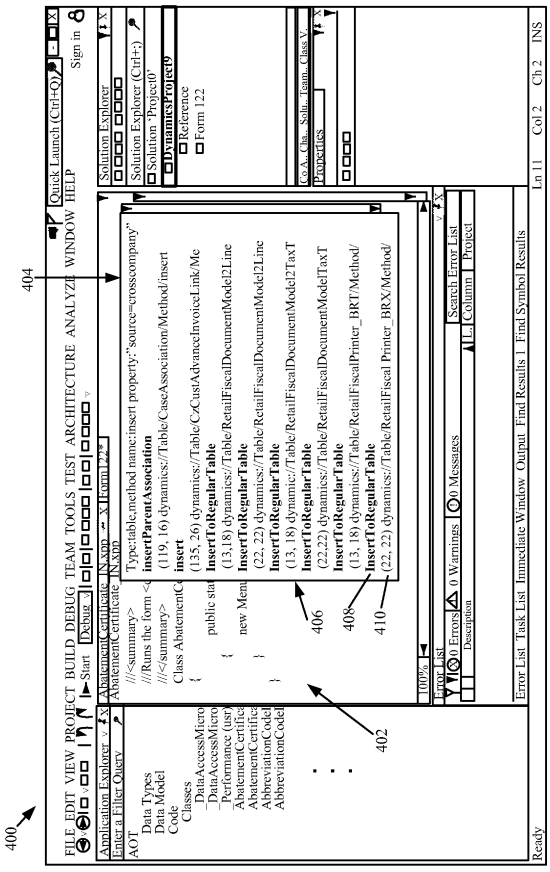
【図３】


統合開発システム
インターフェース
250
探索クエリ
探索クエリパーサ
252
名前
タイプ
プロパティ値
探索オプション
プリプロセッサ
254
探索
オプション
タイプ
含まれている
タイプおよび
プロパティ
タイプ情報
プロバイダ
256
要素アクセス
インターフェース
258
要素参照
探索優先度
ヒューリスティックス
260
ルート要素
結果
タイプ
プロパティ
ゲッタ
要素プロセッサ
262
タイプ
要素述語
要素述語ファクトリ
264

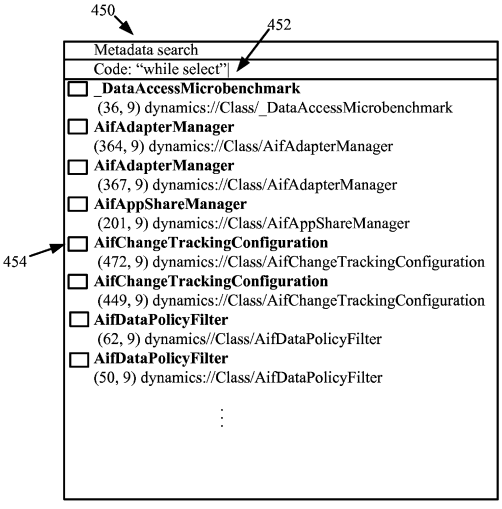
【図４】


開始
300
開発外観を表示する
302
探索入力を受け取る
304
基準を識別するために探索入力を
パースする
306
タイプ基準に基づいて、意味論的な
探索構成要素をインスタンス化する
308
インスタンス化された意味論的な探索構成
要素を使用してモデルストアを探索する
310
320
基準を満足する識別要素
312
探索結果を表示する
314
特定の結果の選択を受け取る
316
選択された結果に対応する要素の場所
へナビゲートする
318
終了

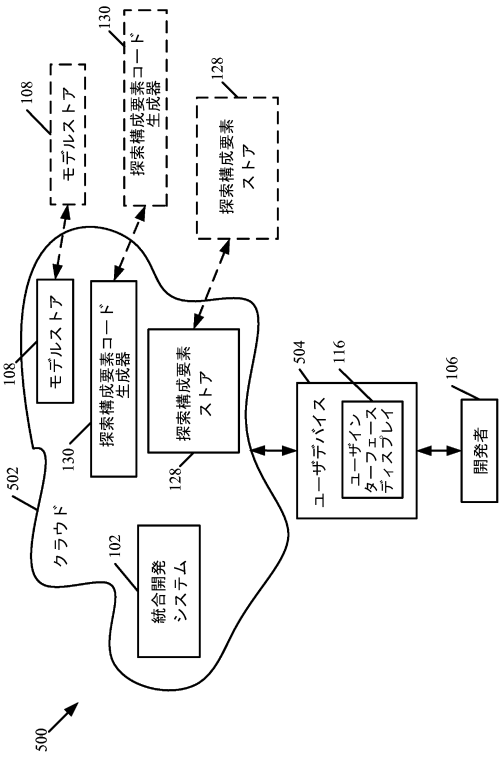
【図５】


400
404
402
406
408
410
FILE EDIT VIEW PROJECT BUILD DEBUG TEAM TOOLS TEST ARCHITECTURE ANALYZE WINDOW HELP
Quick Launch (Ctrl+Q)
Sign in
Start
Debug
Application Explorer
Enter a Filter Query
AOT
Data Types
Data Model
Code
Classes
Solution Explorer
Solution Explorer (Ctrl+;)
Solution 'Project0'
Dynamics Project0
References
Form 122
Properties
Error List
Task List
Immediate Window
Output
Find Results 1
Find Symbol Results
Search Error List
0 Errors
0 Warnings
0 Messages
Ready
Ln 11
Col 2
Ch 2
INS

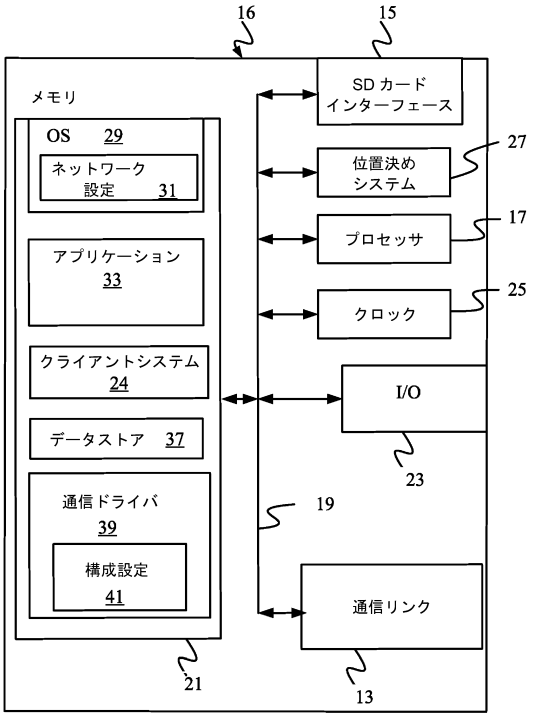
【図６】


450
452
Metadata search
Code: “while select”|
_DataAccessMicrobenchmark
(36, 9) dynamics://Class/_DataAccessMicrobenchmark
AifAdapterManager
(364, 9) dynamics://Class/AifAdapterManager
AifAdapterManager
(367, 9) dynamics://Class/AifAdapterManager
AifAppShareManager
(201, 9) dynamics://Class/AifAppShareManager
454
AifChangeTrackingConfiguration
(472, 9) dynamics://Class/AifChangeTrackingConfiguration
AifChangeTrackingConfiguration
(449, 9) dynamics://Class/AifChangeTrackingConfiguration
AifDataPolicyFilter
(62, 9) dynamics//Class/AifDataPolicyFilter
AifDataPolicyFilter
(50, 9) dynamics://Class/AifDataPolicyFilter

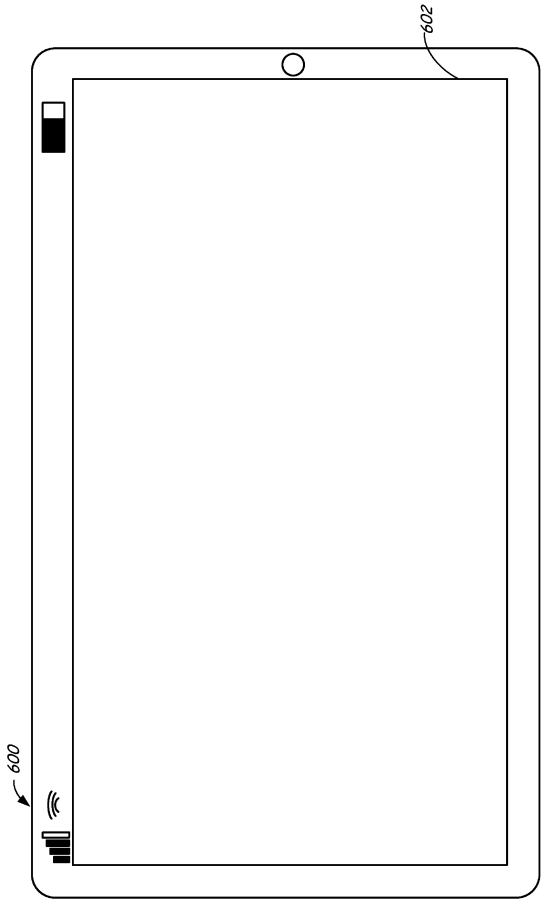
【図７】


モデルストア
探索構成要素コード生成器
探索構成要素ストア
108
130
128
モデルストア
探索構成要素コード生成器
探索構成要素ストア
504
116
106
ユーザデバイス
ユーザインターフェースディスプレイ
開発者
502
クラウド
102
統合開発システム
500

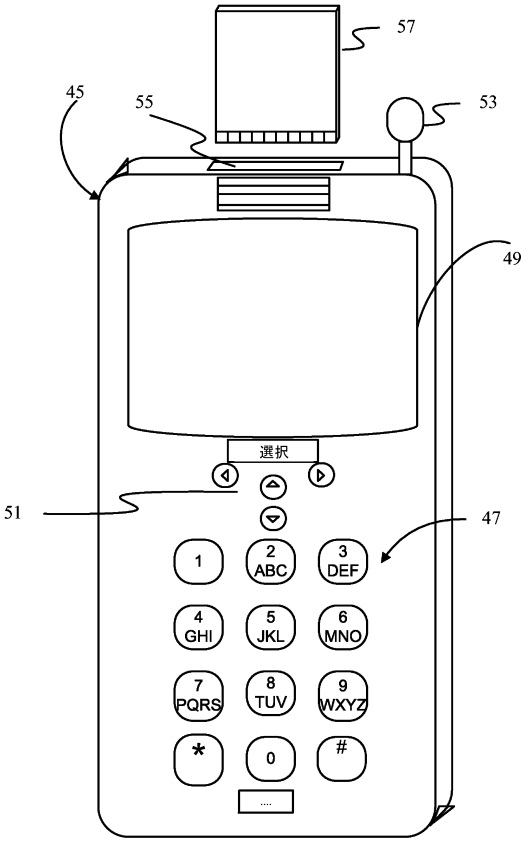
【図８】


16
15
メモリ
OS 29
ネットワーク設定 31
アプリケーション 33
クライアントシステム 24
データストア 37
通信ドライバ 39
構成設定 41
21
SD カードインターフェース
位置決めシステム
プロセッサ
クロック
I/O
通信リンク
27
17
25
23
19
13

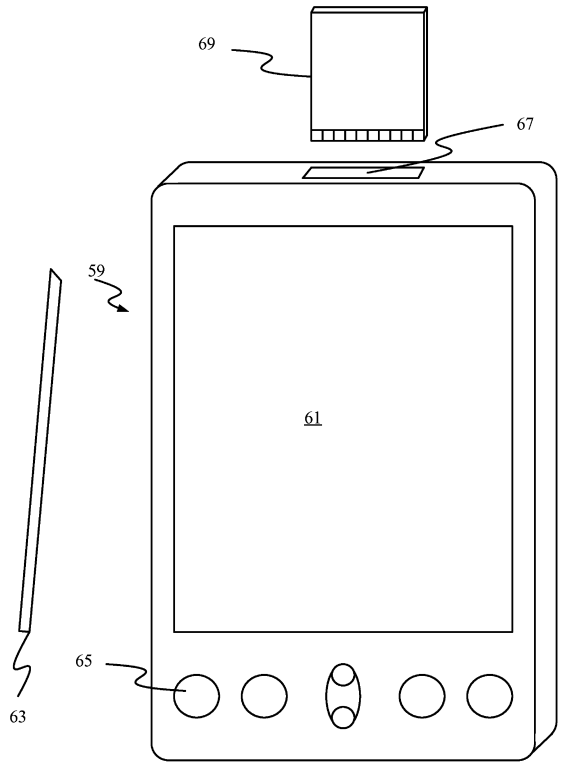
【図９】


602
600

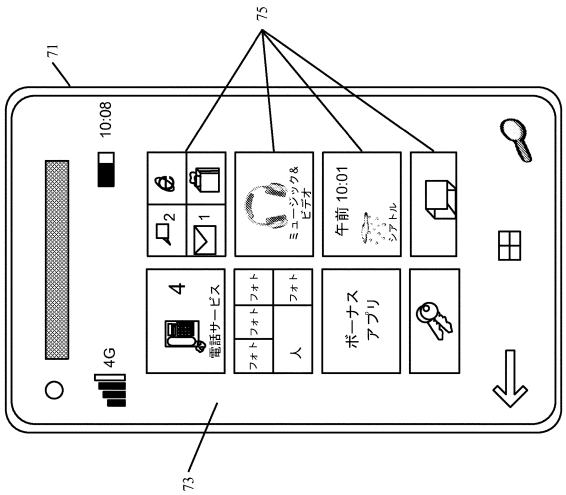
【図１０】


57
45
55
53
49
選択
51
47
1
2 ABC
3 DEF
4 GHI
5 JKL
6 MNO
7 PQRS
8 TUV
9 WXYZ
*
0
#

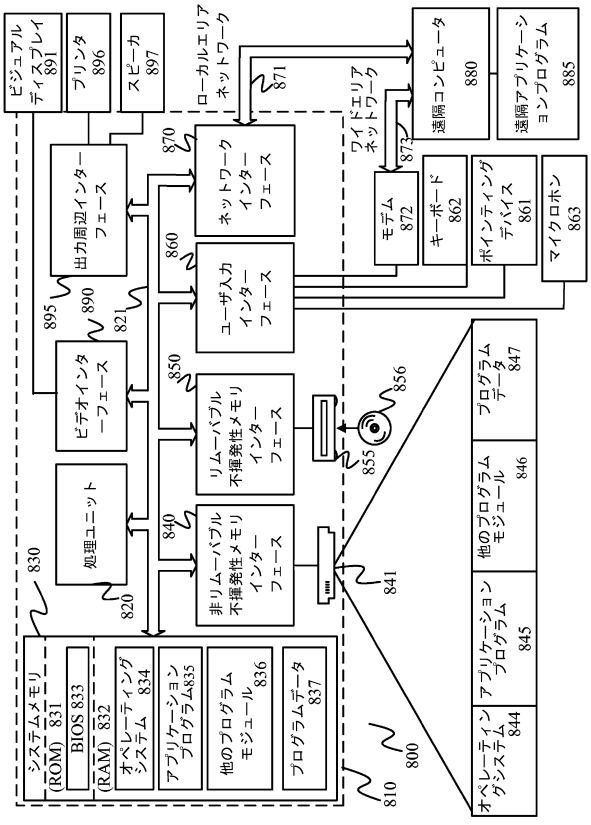
【図１１】


69
67
59
61
65
63

【図１２】


71
75
10:08
4G
電話サービス
ミュージック&ビデオ
午前 10:01
ボーナス アプリ
フォト
人
73

【図１３】


ビジュアルディスプレイ 891
プリンタ 896
スピーカ 897
ローカルエリアネットワーク 871
ワイドエリアネットワーク 872
遠隔コンピュータ 880
遠隔アプリケーションプログラム 885
出力周辺インターフェース 890
ネットワークインターフェース 870
モデム 872
キーボード 862
ポインティングデバイス 861
マイクロホン 863
ユーザ入力インターフェース 860
ビデオインターフェース 895
リムーバブル不揮発性メモリインターフェース 850
非リムーバブル不揮発性メモリインターフェース 840
処理ユニット 820
システムメモリ (ROM) 831
BIOS 833
(RAM) 832
オペレーティングシステム 834
アプリケーションプログラム 835
他のプログラムモジュール 836
プログラムデータ 837
オペレーティングシステム 844
アプリケーションプログラム 845
他のプログラムモジュール 846
プログラムデータ 847
830
821
856
855
841
800
810

フロントページの続き

(74)代理人 100108213
弁理士 阿部 豊隆
(74)代理人 100142044
弁理士 渡邉 直幸
(72)発明者 シャキルチアノフ, アントン
アメリカ合衆国, ワシントン州 ９８０５２－６３９９, レッドモンド, ワン マイクロソフト ウェイ, マイクロソフト テクノロジー ライセンシング, エルエルシー内, エルシーエー －インターナショナル パテンツ （８／１１７２）
(72)発明者 ナラヤナン, スリヤ
アメリカ合衆国, ワシントン州 ９８０５２－６３９９, レッドモンド, ワン マイクロソフト ウェイ, マイクロソフト テクノロジー ライセンシング, エルエルシー内, エルシーエー －インターナショナル パテンツ （８／１１７２）
(72)発明者 ユ, リアン
アメリカ合衆国, ワシントン州 ９８０５２－６３９９, レッドモンド, ワン マイクロソフト ウェイ, マイクロソフト テクノロジー ライセンシング, エルエルシー内, エルシーエー －インターナショナル パテンツ （８／１１７２）
(72)発明者 カミンスキ, トマシュ
アメリカ合衆国, ワシントン州 ９８０５２－６３９９, レッドモンド, ワン マイクロソフト ウェイ, マイクロソフト テクノロジー ライセンシング, エルエルシー内, エルシーエー －インターナショナル パテンツ （８／１１７２）

審査官 坂庭 剛史

(56)参考文献 特開平０６－３４８４７１（ＪＰ，Ａ）
特開２００６－１０６９４８（ＪＰ，Ａ）
特開２００８－２４２８１１（ＪＰ，Ａ）
特開２０１０－１９１８５８（ＪＰ，Ａ）
米国特許出願公開第２０１２／０２５４８３５（ＵＳ，Ａ１）
特開平０２－０６７６２９（ＪＰ，Ａ）
特開平０６－２７４３３２（ＪＰ，Ａ）
特表２００８－５３３５４４（ＪＰ，Ａ）
特開２０１２－１２３６７５（ＪＰ，Ａ）
米国特許出願公開第２０１３／０１２４５２９（ＵＳ，Ａ１）

(58)調査した分野(Int.Cl.，ＤＢ名)
Ｇ０６Ｆ ８／３６
Ｇ０６Ｆ １６／００