(19)日本国特許庁(JP)

(12)特許公報(B2)

(11)特許番号
特許第7389246号
(P7389246)

(45)発行日　令和5年11月29日(2023.11.29)

(24)登録日　令和5年11月20日(2023.11.20)

(51)国際特許分類
G06F 8/75 (2018.01)
G06F 11/36 (2006.01)

FI
G06F 8/75
G06F 11/36 104

請求項の数 13 （全15頁）

(21)出願番号　特願2022-520829(P2022-520829)
(86)(22)出願日　令和3年8月27日(2021.8.27)
(65)公表番号　特表2023-528102(P2023-528102A)
(43)公表日　令和5年7月4日(2023.7.4)
(86)国際出願番号　PCT/CN2021/115006
(87)国際公開番号　WO2022/179070
(87)国際公開日　令和4年9月1日(2022.9.1)
審査請求日　令和4年4月4日(2022.4.4)
(31)優先権主張番号　202110219222.2
(32)優先日　令和3年2月26日(2021.2.26)
(33)優先権主張国・地域又は機関
中国(CN)

(73)特許権者　514322098
ベイジン バイドゥ ネットコム サイエンス テクノロジー カンパニー リミテッド
Ｂｅｉｊｉｎｇ Ｂａｉｄｕ Ｎｅｔｃｏｍ Ｓｃｉｅｎｃｅ Ｔｅｃｈｎｏｌｏｇｙ Ｃｏ．，Ｌｔｄ．
中華人民共和国 ペキン １０００８５，ハイディアン ディストリクト，シャンディ テンス ストリート，１０番，バイドゥ キャンパス ２階
２／Ｆ Ｂａｉｄｕ Ｃａｍｐｕｓ，Ｎｏ．１０，Ｓｈａｎｇｄｉ １０ｔｈ Ｓｔｒｅｅｔ，Ｈａｉｄｉａｎ Ｄｉｓｔｒｉｃｔ，Ｂｅｉｊｉｎｇ １０００

最終頁に続く

(54)【発明の名称】　依存グラフ生成方法、装置、機器、記憶媒体、及びプログラム製品

(57)【特許請求の範囲】

【請求項１】

関数コンポーネントで使用されるｈｏｏｋアプリケーションプログラミングインタフェースＡＰＩを取得することと、

前記関数コンポーネントで使用される前記ｈｏｏｋ　ＡＰＩのパラメータ値を取得することと、

依存グラフを生成することとを含み、

ここで、前記ｈｏｏｋ　ＡＰＩが前記依存グラフにおけるｈｏｏｋノードとし、前記パラメータ値が前記依存グラフにおける値ノードとし、値ノードと対応するｈｏｏｋノードとの間に接続関係が存在する、コンピュータによって実行される依存グラフ生成方法。

【請求項２】

前記関数コンポーネントで使用されるｈｏｏｋ　ＡＰＩがカスタムのｈｏｏｋ　ＡＰＩを含む場合、前記カスタムのｈｏｏｋ　ＡＰＩに対して再帰的分析を行い、直接的にアトミックｈｏｏｋ　ＡＰＩを取得することと、

前記再帰的分析により前記アトミックｈｏｏｋ　ＡＰＩ以外の他のｈｏｏｋ　ＡＰＩが得られた場合、前記アトミックｈｏｏｋ　ＡＰＩのパラメータ値を取得し、かつ前記再帰的分析により得られた前記アトミックｈｏｏｋ　ＡＰＩ以外の他のｈｏｏｋ　ＡＰＩのパラメータ値を取得することとをさらに含み、

ここで、前記依存グラフにおいて、

前記アトミックｈｏｏｋ　ＡＰＩに対応するｈｏｏｋノード、前記他のｈｏｏｋ　ＡＰＩに

対応するｈｏｏｋノード、前記アトミックｈｏｏｋ　ＡＰＩのパラメータ値に対応する値ノード、及び前記他のｈｏｏｋ　ＡＰＩのパラメータ値に対応する値ノードをさらに含み、かつ各値ノードと対応するｈｏｏｋノードとの間に接続関係が存在する、請求項１に記載の依存グラフ生成方法。

【請求項３】

前記依存グラフが表現式により得られた新たな識別子のターゲット値ノードを含む場合、前記依存グラフにおいて前記新たな識別子に対応する値ノードを生成し、かつ前記ターゲット値ノードと前記新たな識別子の値ノードとの接続関係を確立する、請求項１に記載の依存グラフ生成方法。

【請求項４】

前記関数コンポーネントの最上層の拡張可能なマークアップ言語ＸＭＬ表現式を取得することと、

前記依存グラフにおいて前記最上層のＸＭＬ表現式に対応するコンポーネントノードを生成することと、

前記最上層のＸＭＬ表現式には前記依存グラフにおける値ノードの識別子を引用するターゲットＸＭＬ表現式を含む場合、前記依存グラフにおいて前記ターゲットＸＭＬ表現式と対応する値ノードとの接続関係を確立することとをさらに含む、請求項１に記載の依存グラフ生成方法。

【請求項５】

前記の前記依存グラフにおいて前記最上層のＸＭＬ表現式に対応するコンポーネントノードを生成することは、

前記最上層のＸＭＬ表現式の構文木を簡略化し、少なくとも一つの所定レベルのＸＭＬ表現式を取得し、かつ前記依存グラフにおいて前記少なくとも一つの所定レベルのＸＭＬ表現式に対応するコンポーネントノードを生成することを含む、請求項４に記載の依存グラフ生成方法。

【請求項６】

関数コンポーネントで使用されるｈｏｏｋアプリケーションプログラミングインタフェースＡＰＩを取得する第一取得モジュールと、

前記関数コンポーネントで使用される前記ｈｏｏｋ　ＡＰＩのパラメータ値を取得する第二取得モジュールと、

依存グラフを生成する第一生成モジュールとを含み、

ここで、前記ｈｏｏｋ　ＡＰＩが前記依存グラフにおけるｈｏｏｋノードとし、前記パラメータ値が前記依存グラフにおける値ノードとし、値ノードと対応するｈｏｏｋノードとの間に接続関係が存在する、依存グラフ生成装置。

【請求項７】

前記関数コンポーネントで使用されるｈｏｏｋ　ＡＰＩがカスタムのｈｏｏｋ　ＡＰＩを含む場合、前記カスタムのｈｏｏｋ　ＡＰＩに対して再帰的分析を行い、直接的にアトミックｈｏｏｋ　ＡＰＩを取得する再帰的分析モジュールと、

前記再帰的分析により前記アトミックｈｏｏｋ　ＡＰＩ以外の他のｈｏｏｋ　ＡＰＩが得られた場合、前記アトミックｈｏｏｋ　ＡＰＩのパラメータ値を取得し、かつ前記再帰的分析により得られた前記アトミックｈｏｏｋ　ＡＰＩ以外の他のｈｏｏｋ　ＡＰＩのパラメータ値を取得する第三取得モジュールとをさらに含み、

ここで、前記依存グラフにおいて、

前記アトミックｈｏｏｋ　ＡＰＩに対応するｈｏｏｋノード、前記他のｈｏｏｋ　ＡＰＩに対応するｈｏｏｋノード、前記アトミックｈｏｏｋ　ＡＰＩのパラメータ値に対応する値ノード、及び前記他のｈｏｏｋ　ＡＰＩのパラメータ値に対応する値ノードをさらに含み、かつ各値ノードと対応するｈｏｏｋノードとの間に接続関係が存在する、請求項６に記載の依存グラフ生成装置。

【請求項８】

前記依存グラフが表現式により得られた新たな識別子のターゲット値ノードを含む場合、

前記依存グラフにおいて前記新たな識別子に対応する値ノードを生成し、かつ前記ターゲット値ノードと前記新たな識別子の値ノードとの接続関係を確立する第二生成モジュールをさらに含む、請求項６に記載の依存グラフ生成装置。
【請求項９】
前記関数コンポーネントの最上層の拡張可能なマークアップ言語ＸＭＬ表現式を取得する第四取得モジュールと、
前記依存グラフにおいて前記最上層のＸＭＬ表現式に対応するコンポーネントノードを生成する第三生成モジュールと、
前記最上層のＸＭＬ表現式には前記依存グラフにおける値ノードの識別子を引用するターゲットＸＭＬ表現式を含む場合、前記依存グラフにおいて前記ターゲットＸＭＬ表現式と対応する値ノードとの接続関係を確立する確立モジュールとをさらに含む、請求項６に記載の依存グラフ生成装置。
【請求項１０】
前記第三生成モジュールは、前記最上層のＸＭＬ表現式の構文木を簡略化し、少なくとも一つの所定レベルのＸＭＬ表現式を取得し、かつ前記少なくとも一つの所定レベルのＸＭＬ表現式を前記依存グラフのコンポーネントノードとする、請求項９に記載の依存グラフ生成装置。
【請求項１１】
少なくとも一つのプロセッサと、
前記少なくとも一つのプロセッサと通信接続されたメモリとを含み、
ここで、前記メモリは前記少なくとも一つのプロセッサにより実行可能な指令を記憶し、前記指令が前記少なくとも一つのプロセッサに実行されることにより、前記少なくとも一つのプロセッサが請求項１～５のいずれか一項に記載の方法を実行することができる、電子機器。
【請求項１２】
コンピュータ指令を記憶した非一時的なコンピュータ可読記憶媒体であって、
前記コンピュータ指令はコンピュータに請求項１～５のいずれか一項に記載の方法を実行させる、コンピュータ可読記憶媒体。
【請求項１３】
コンピュータプログラムであって、
前記コンピュータプログラムはプロセッサにより実行される時に請求項１～５のいずれか一項に記載の方法を実現する、コンピュータプログラム。
【発明の詳細な説明】
【技術分野】
【０００１】
本開示はコンピュータの技術分野に関し、特にコンピュータ言語技術に関する。
【背景技術】
【０００２】
フック（ｈｏｏｋ）アプリケーションプログラミングインタフェース（Ａｐｐｌｉｃａｔｉｏｎ　Ｐｒｏｇｒａｍｍｉｎｇ　Ｉｎｔｅｒｆａｃｅ、ＡＰＩ）は関数コンポーネントにおいてコンポーネント状態を組織するためのＡＰＩである。ｈｏｏｋ　ＡＰＩに対する分析は主にｈｏｏｋ　ＡＰＩのライフサイクルを分析することである。
【発明の概要】
【０００３】
本開示は、依存グラフ生成方法、装置、機器、記憶媒体、及びプログラム製品を提供する。
【０００４】
本開示の一態様によれば、依存グラフ生成方法を提供する。
依存グラフ生成方法において、
関数コンポーネントで使用されるｈｏｏｋアプリケーションプログラミングインタフェースＡＰＩを取得することと、

前記関数コンポーネントで使用される前記ｈｏｏｋ　ＡＰＩのパラメータ値を取得することと、
依存グラフを生成することとを含み、
ここで、前記ｈｏｏｋ　ＡＰＩが前記依存グラフにおけるｈｏｏｋノードとし、前記パラメータ値が前記依存グラフにおける値ノードとし、値ノードと対応するｈｏｏｋノードとの間に接続関係が存在する。
【０００５】
本開示の別の態様によれば、依存グラフ生成装置を提供する。
依存グラフ生成装置において、
関数コンポーネントで使用されるｈｏｏｋアプリケーションプログラミングインタフェースＡＰＩを取得する第一取得モジュールと、
前記関数コンポーネントで使用される前記ｈｏｏｋ　ＡＰＩのパラメータ値を取得する第二取得モジュールと、
依存グラフを生成する第一生成モジュールとを含み、
ここで、前記ｈｏｏｋ　ＡＰＩが前記依存グラフにおけるｈｏｏｋノードとし、前記パラメータ値が前記依存グラフにおける値ノードとし、値ノードと対応するｈｏｏｋノードとの間に接続関係が存在する。
【０００６】
本開示の別の態様によれば、電子機器を提供する。
電子機器において、
少なくとも一つのプロセッサと、
前記少なくとも一つのプロセッサと通信接続されたメモリとを含み、
ここで、前記メモリは前記少なくとも一つのプロセッサにより実行可能な指令を記憶し、前記指令が前記少なくとも一つのプロセッサに実行されることにより、前記少なくとも一つのプロセッサが本開示による依存グラフ生成方法を実行することができる。
【０００７】
本開示の別の態様によれば、コンピュータ可読記憶媒体を提供する。
コンピュータ指令を記憶した非一時的なコンピュータ可読記憶媒体において、
前記コンピュータ指令は前記コンピュータに本開示による依存グラフ生成方法を実行させる、コンピュータ可読記憶媒体。
【０００８】
本開示の別の態様によれば、コンピュータプログラム製品を提供する。
コンピュータプログラム製品において、
コンピュータプログラムを含み、
ここで、前記コンピュータプログラムはプロセッサにより実行される時に本開示による依存グラフ生成方法を実現する。
【０００９】
理解すべきことは、本部分に記載された内容は本開示の実施例のキー又は重要な特徴を識別することを意図するものではなく、本開示の範囲を限定するものではない。本開示の他の特徴は、以下の説明により容易に理解されるであろう。
【図面の簡単な説明】
【００１０】
図面は本解決手段をよりよく理解するために用いられ、本開示を限定するものではない。
【００１１】
【図１】図１は本開示の提供する依存グラフ生成方法のフローチャートである。
【図２】図２は本開示の提供する依存グラフの概略図である。
【図３】図３は本開示の提供する別の依存グラフの概略図である。
【図４】図４は本開示の提供する依存グラフ生成装置の構造図である。
【図５】図５は本開示の提供する別の依存グラフ生成装置の構造図である。
【図６】図６は本開示の提供する別の依存グラフ生成装置の構造図である。

【図７】図７は本開示の提供する別の依存グラフ生成装置の構造図である。
【図８】図８は本開示の実施例に係る依存グラフ生成方法を実現するための電子機器のブロック図である。
【発明を実施するための形態】
【００１２】
以下に図面を参照して本開示の例示的な実施例を説明し、ここで本開示の実施例の様々な詳細を含み理解することに役立ち、それらを例示的なものと考えるべきである。したがって、当業者であれば、ここで説明した実施例に対して様々な変更及び修正を行うことができ、本開示の範囲及び精神から逸脱することはない。同様に、明確かつ簡単に説明するために、以下の説明において公知の機能及び構造に対する説明を省略する。
【００１３】
図１に示すとおり、図１は本開示の提供する依存グラフ生成方法のフローチャートであり、図１に示すように、ステップＳ１０１～１０３を含む。
ステップＳ１０１で、関数コンポーネントで使用されるｈｏｏｋ　ＡＰＩを取得する。
【００１４】
上記関数コンポーネントは、予め作成された関数コンポーネントであってもよく、かつ一つ又は複数の関数コンポーネントで使用されるｈｏｏｋ　ＡＰＩを取得することができる。
【００１５】
また、上記ｈｏｏｋ　ＡＰＩは上記関数コンポーネントで使用される全てのｈｏｏｋ　ＡＰＩであってもよい。なお、ここで、使用が呼び出しと理解されてもよい。例えば、上記ｈｏｏｋ　ＡＰＩは、上記関数コンポーネントが呼び出した全てのｈｏｏｋ　ＡＰＩであってもよい。さらに、ｈｏｏｋ　ＡＰＩはｈｏｏｋと略称されてもよい。
【００１６】
本開示において、いくつかの実施形態において、ｈｏｏｋ　ＡＰＩは、Ｒｅａｃｔ　ｈｏｏｋ　ＡＰＩであってもよく、Ｒｅａｃｔ関数コンポーネントにおいて、コンポーネント状態を組織するために用いられる。
【００１７】
また、本開示におけるｈｏｏｋ　ＡＰＩは、保持状態、キャッシュ及び保持副作用に応じて分類することができる。例えば、保持状態でのｈｏｏｋ　ＡＰＩは、ｕｓｅＳｔａｔｅ、ｕｓｅＲｅｄｕｃｅｒ、ｕｓｅＲｅｆなどを含み、キャッシュされたｈｏｏｋ　ＡＰＩはｕｓｅＣａｌｌｂａｃｋ、ｕｓｅＭｅｍｏなどを含み、副作用を持つｈｏｏｋ　ＡＰＩはｕｓｅＥｆｆｅｃｔ、ｕｓｅＬａｙｏｕｔＥｆｆｅｃｔなどを含む。
【００１８】
一つの実施形態において、上記関数コンポーネントの抽象構文木を構築し、かつ該構文木により関数コンポーネントで使用されるｈｏｏｋ　ＡＰＩを取得することができる。これに対して本開示は限定されず、例えば、関数コンポーネントから関数コンポーネントで使用されるｈｏｏｋ　ＡＰＩを直接抽出することもできる。
【００１９】
ステップＳ１０２で、前記関数コンポーネントで使用される前記ｈｏｏｋ　ＡＰＩのパラメータ値を取得する。
【００２０】
該ステップは、ステップＳ１０２で取得された各ｈｏｏｋ　ＡＰＩのパラメータ値を取得してもよい。ここで、パラメータ値は入力パラメータ及び戻り値識別子のうちの少なくとも一つを含む。例えば、いくつかのｈｏｏｋ　ＡＰＩのパラメータ値は入力パラメータ及び戻り値識別子を含み、他のｈｏｏｋ　ＡＰＩは入力パラメータ及び戻り値識別子のうちの少なくとも一つだけを含む。例えば、あるｈｏｏｋ　ＡＰＩは入力パラメータのみがあり、戻り値識別子がなく、又は、いくつかの可能なｈｏｏｋ　ＡＰＩに戻り値識別子のみがあり、入力パラメータがない。
【００２１】
ステップＳ１０３で、依存グラフを生成する。ここで、前記ｈｏｏｋ　ＡＰＩは前記依存

グラフにおけるｈｏｏｋノードとし、前記パラメータ値は前記依存グラフにおける値ノードとし、値ノードと対応するｈｏｏｋノードとの間に接続関係が存在する。

【００２２】

該ステップは、ステップＳ１０２で取得された各ｈｏｏｋ　ＡＰＩを上記依存グラフにおけるｈｏｏｋノードとすることができる。具体的には、各ｈｏｏｋ　ＡＰＩについて、それぞれ対応するｈｏｏｋノードを生成する。ステップＳ１０２で取得された各パラメータ値を値ノードとすることができる。例えば、各入力パラメータ、及び各戻り値識別子に対応する値ノードを生成する。

【００２３】

上記値ノードと対応する前記ｈｏｏｋノードとの間に接続関係が存在することは、入力パラメータに対応する値ノードから該入力パラメータに対応するｈｏｏｋ　ＡＰＩのｈｏｏｋノードに一つの辺を接続し、ｈｏｏｋ　ＡＰＩのｈｏｏｋノードから該ｈｏｏｋ　ＡＰＩの戻り値識別子に対応する値ノードに一つの辺を接続することであってもよい。値ノードとｈｏｏｋノードとの間の接続関係により、ｈｏｏｋ　ＡＰＩと入力パラメータ及び戻り値識別子との間の依存関係を正確に表現することができる。

【００２４】

本開示において、生成された依存グラフにおけるｈｏｏｋ　ＡＰＩを依存グラフにおけるｈｏｏｋノードとし、ｈｏｏｋ　ＡＰＩのパラメータ値を前記依存グラフにおける値ノードとする。このように、該依存グラフにより上記ｈｏｏｋノードと値ノードとの間の関係を表示することができるので、開発者がｈｏｏｋ　ＡＰＩを容易に使用してコードプログラミングを行うことに役立ち、さらにコードプログラミングの指導効果を向上させ、開発してコード品質を向上させることに役立つ。

【００２５】

例えば、以下では関数コンポーネントがｕｓｅＳｔａｔｅ、ｕｓｅＣａｌｌｂａｃｋ及びｕｓｅＥｆｆｅｃｔの三つのｈｏｏｋ　ＡＰＩを含むことを例に挙げて説明する。該関数コンポーネントにおいて、ｕｓｅＳｔａｔｅの戻り値識別子はｃｏｕｎｔ及びｓｅｔＣｏｕｎｔを含み、ｕｓｅＣａｌｌｂａｃｋの入力パラメータはｃｏｕｎｔを含むことができる。ｕｓｅＣａｌｌｂａｃｋの戻り値識別子はｏｎＬｉｃｋを含み、ｕｓｅＥｆｆｅｃｔの入力パラメータはｓｅｔＣｏｕｎｔを含むことができる。

【００２６】

このように、上記関数コンポーネントについて、ステップＳ１０１で取得されたｈｏｏｋ　ＡＰＩはｕｓｅＳｔａｔｅ、ｕｓｅＣａｌｌｂａｃｋ及びｕｓｅＥｆｆｅｃｔを含み、図２に示す依存グラフを生成することができる。ここで、２０１、２０２、２０３はそれぞれｕｓｅＳｔａｔｅ、ｕｓｅＣａｌｌｂａｃｋ及びｕｓｅＥｆｆｅｃｔの三つのｈｏｏｋ　ＡＰＩを示し、２０４、２０５、２０６はそれぞれｃｏｕｎｔ、ｓｅｔＣｏｕｎｔ、ｏｎＬｉｃｋという三つのパラメータ値を示す。

【００２７】

なお、以上は簡単な関数コンポーネントのみを例に挙げて説明する。実際にはより多くのｈｏｏｋ　ＡＰＩ、及びより複雑な入力パラメータ及び戻り値識別子を含んでもよい。

【００２８】

選択的な実施形態として、

前記関数コンポーネントで使用されるｈｏｏｋ　ＡＰＩがカスタムのｈｏｏｋ　ＡＰＩを含む場合、前記カスタムのｈｏｏｋ　ＡＰＩに対して再帰的分析を行い、直接的にアトミックｈｏｏｋ　ＡＰＩを取得することと、

前記再帰的分析により前記アトミックｈｏｏｋ　ＡＰＩ以外の他のｈｏｏｋ　ＡＰＩが得られた場合、前記アトミックｈｏｏｋ　ＡＰＩのパラメータ値を取得し、かつ前記再帰的分析により得られた前記アトミックｈｏｏｋ　ＡＰＩ以外の他のｈｏｏｋ　ＡＰＩのパラメータ値を取得することとをさらに含む。

ここで、前記依存グラフにおいて、

前記アトミックｈｏｏｋ　ＡＰＩに対応するｈｏｏｋノード、前記他のｈｏｏｋ　ＡＰＩに

対応するｈｏｏｋノード、前記アトミックｈｏｏｋ　ＡＰＩのパラメータ値に対応する値ノード、及び前記他のｈｏｏｋ　ＡＰＩのパラメータ値に対応する値ノードをさらに含み、かつ各値ノードと対応するｈｏｏｋノードとの間に接続関係が存在する。

【００２９】

ここで、上記カスタムのｈｏｏｋ　ＡＰＩは、Ｒｅａｃｔが提供するアトミックｈｏｏｋに基づいて開発者自分で設定されたｈｏｏｋ　ＡＰＩである。上記他のｈｏｏｋ　ＡＰＩは、上記カスタムの他のｈｏｏｋ　ＡＰＩと上記アトミックｈｏｏｋ　ＡＰＩとの間に含まれる一つ又は複数のｈｏｏｋ　ＡＰＩを指す。

【００３０】

いくつかの実施形態において、上記関数コンポーネントは一つ又は複数のカスタムのｈｏｏｋ　ＡＰＩを含むことができる。上記カスタムのｈｏｏｋ　ＡＰＩのそれぞれに対して、以上のプロセスを実行することができる。

【００３１】

同様に、上記パラメータ値は入力パラメータ及び戻り値識別子のうちの少なくとも一つを含んでもよい。異なるｈｏｏｋ　ＡＰＩのパラメータ値は同じであってもよく異なってもよい。

【００３２】

該実施形態において、上記再帰的分析により関数コンポーネントに係るより多くのｈｏｏｋ　ＡＰＩを取得することができる。それにより、上記依存グラフはより多くのｈｏｏｋ　ＡＰＩの依存関係を表示して、依存グラフの指導効果をさらに向上させることができる。

【００３３】

なお、上記再帰的分析によりアトミックｈｏｏｋ　ＡＰＩのみを取得すれば、アトミックｈｏｏｋ　ＡＰＩのｈｏｏｋノード及び対応する値ノード、これらのｈｏｏｋノードと対応する値ノードとの接続関係を追加することができる。

【００３４】

選択的な実施形態として、

前記依存グラフが表現式により得られた新たな識別子のターゲット値ノードを含む場合、前記依存グラフにおいて前記新たな識別子に対応する値ノードを生成し、かつ前記ターゲット値ノードと前記新たな識別子の値ノードとの接続関係を確立することをさらに含む。

【００３５】

上記表現式により得られた新たな識別子のターゲット値ノードは、表現式により該値ノードに対応する識別子から得られた新たな識別子である。表現式により、ある入力パラメータの識別子又は戻り値識別子から新たな識別子を得られ、該入力パラメータ又は戻り値識別子に対応する値ノードは上記ターゲット値ノードである。例えば、ある入力パラメータの識別子又は戻り値識別子は、別の識別子の変数としてもよい。

【００３６】

該実施形態において、新たな識別子の値ノード及びその接続関係を生成することにより、上記依存グラフはより多くの値ノードの依存関係を表示することができるので、依存グラフの指導効果をさらに向上させる。

【００３７】

選択的な実施形態として、さらに、

前記関数コンポーネントの最上層の拡張可能なマークアップ言語（Ｅｘｔｅｎｓｉｂｌｅ　Ｍａｒｋｕｐ　Ｌａｎｇｕａｇｅ、ＸＭＬ）表現式を取得することと、

前記依存グラフにおいて前記最上層のＸＭＬ表現式に対応するコンポーネントノードを生成ことと、

前記最上層のＸＭＬ表現式には前記依存グラフにおける値ノードの識別子を引用するターゲットＸＭＬ表現式を含む場合、前記依存グラフにおいて前記ターゲットＸＭＬ表現式と対応する値ノードとの接続関係を確立することとを含む。

【００３８】

ここで、上記ＸＭＬ表現式はＪＳＸ表現値であってもよい。

【００３９】
前記関数コンポーネントの最上層のＸＭＬ表現式を取得することは、上記関数コンポーネントの構文木に基づいて関数コンポーネントの最上層のＸＭＬ表現式を取得することであってもよい。
【００４０】
前記依存グラフにおいて前記最上層のＸＭＬ表現式に対応するコンポーネントノードを生成することは、上記依存グラフにおいて全ての最上層のＸＭＬ表現式に係る全ての又は一部のＸＭＬ表現式に対応するコンポーネントノードを生成することであってもよい。ここで、各ＸＭＬ表現式はそれぞれ一つのコンポーネントノードとする。
【００４１】
また、上記依存グラフにおいて前記最上層のＸＭＬ表現式に対応するコンポーネントノードを生成することは、上記依存グラフの個別生成コンポーネントノードの間の関係図であってもよい。例えば、親コンポーネントノードと子コンポーネントノードとの間の関係図であり、一つのコンポーネントノードの間のフォレストプロットを構成する。該フォレストプロットを構成した後、前記ターゲットＸＭＬ表現式と対応する値ノードとの接続関係を確立する。例えば、図３のように、図３の左側はステップＳ１０３で生成された依存グラフであり、図３の右側は該実施形態の構造のフォレストプロットである。二つの図の間に接続された破線は、ターゲットＸＭＬ表現式と対応する値ノードとの接続関係を示す。
【００４２】
該実施形態において、ターゲットＸＭＬ表現式と対応する値ノードとの接続関係により、上記依存グラフはＸＭＬ表現式の値ノード間の依存関係を表示することができ、依存グラフの指導効果をさらに向上させる。
【００４３】
好ましくは、前記の前記依存グラフにおいて前記最上層のＸＭＬ表現式に対応するコンポーネントノードを生成することは、
前記最上層のＸＭＬ表現式の構文木を簡略化し、少なくとも一つの所定レベルのＸＭＬ表現式を取得し、かつ前記依存グラフにおいて前記少なくとも一つの所定レベルのＸＭＬ表現式に対応するコンポーネントノードを生成することを含む。
【００４４】
ここで、前記最上層のＸＭＬ表現式の構文木を簡略化することは、上記最上層のＸＭＬ表現式の構文木におけるＸＭＬ表現式を簡略化し、上記少なくとも一つの所定レベルのＸＭＬ表現式のみをコンポーネントノードとすることであってもよい。ここで、上記少なくとも一つの所定レベルは子孫レベルを含み、すなわち、子孫レベルのＸＭＬ表現式のみをコンポーネントノードとして残してもよい。
【００４５】
該実施形態では、最上層のＸＭＬ表現式の構文木を簡略化し、依存グラフに重要なレベルのコンポーネントノードだけを残し、依存グラフを簡略化し、依存グラフの表示効果を向上させることができる。
【００４６】
なお、本開示の上記依存グラフは、
上記依存グラフにおいて出次数が０であるｈｏｏｋノードは、必ず戻り値がないｈｏｏｋＡＰＩ又はコンポーネントノードであると特定し、
上記依存グラフにおける連通サブグラフの個数を、運転時の状態が相互に結合されたサブコンポーネントのグループ数と特定し、
上記依存グラフにおけるある経路には保持状態のｈｏｏｋノードを含まなければ、該経路はキャッシュ可能であり、
上記依存グラフにおける二つのコンポーネントノードが同一の連通サブ図にあり、かつ論理的に結合することができれば、開発者がこの二つのコンポーネントノードに対応するコンポーネントに対して一つの共通の親コンポーネントを追加し、二つのサブコンポーネント行為を一つのコンポーネントにカプセル化し、それにより他の連通サブグラフ上のコン

ポーネントと区別し、
上記依存グラフにおける二つのコンポーネントノードが同一の連通サブ図にあるが、論理的に結合することができなければ、この時に開発者が状態分割の方式により、二つのコンポーネントノードに対応する二つのサブコンポーネントをデカップリングさせ、それにより将来のコンポーネントの多重化にさらなる影響を与えず、
上記依存グラフにおけるキャッシュ可能な経路について、開発者がキャッシュ対策を使用してコンポーネントを最適化することで、より多くの性能収益を得られ、ここで、キャッシュ対策はＲｅａｃｔ．ｍｅｍｏなどに限定されず、
上記依存グラフにおける非キャッシュ可能な経路について、開発者が該パスにおける保持状態でのｈｏｏｋ　ＡＰＩがさらに状態分割された余裕があるか否かを注目し、キャッシュ可能な経路を増加させることを試み、
新たに参加した開発者にとって、上記依存グラフによりコンポーネントの状態を迅速に理解して、複雑な仕事を迅速で効果的に把握し、
異なる関数コンポーネントの依存グラフについて、データ分析の方式によりこれらのグラフにおける共通パターン及び経路を見つけ、共通パターンをコード抽象化することにより、コードの多重化性を向上させることができる、という効果を得られる。

【００４７】

図４に示すとおり、図４は本開示の提供する依存グラフ生成装置であり、図４に示すように、依存グラフ生成装置４００は、
関数コンポーネントで使用されるｈｏｏｋアプリケーションプログラミングインタフェースＡＰＩを取得する第一取得モジュール４０１と、
前記関数コンポーネントで使用される前記ｈｏｏｋ　ＡＰＩのパラメータ値を取得する第二取得モジュール４０２と、
依存グラフを生成する第一生成モジュール４０３とを含み、
ここで、前記ｈｏｏｋ　ＡＰＩが前記依存グラフにおけるｈｏｏｋノードとし、前記パラメータ値が前記依存グラフにおける値ノードとし、値ノードと対応するｈｏｏｋノードとの間に接続関係が存在する。

【００４８】

好ましくは、図５に示すように、
前記関数コンポーネントで使用されるｈｏｏｋ　ＡＰＩがカスタムのｈｏｏｋ　ＡＰＩを含む場合、前記カスタムのｈｏｏｋ　ＡＰＩに対して再帰的分析を行い、直接的にアトミックｈｏｏｋ　ＡＰＩを取得する再帰的分析モジュール４０４と、
前記再帰的分析により前記アトミックｈｏｏｋ　ＡＰＩ以外の他のｈｏｏｋ　ＡＰＩが得られた場合、前記アトミックｈｏｏｋ　ＡＰＩのパラメータ値を取得し、かつ前記再帰的分析により得られた前記アトミックｈｏｏｋ　ＡＰＩ以外の他のｈｏｏｋ　ＡＰＩのパラメータ値を取得する第三取得モジュール４０５とをさらに含み、
ここで、前記依存グラフにおいて、
前記アトミックｈｏｏｋ　ＡＰＩに対応するｈｏｏｋノード、前記他のｈｏｏｋ　ＡＰＩに対応するｈｏｏｋノード、前記アトミックｈｏｏｋ　ＡＰＩのパラメータ値に対応する値ノード、及び前記他のｈｏｏｋ　ＡＰＩのパラメータ値に対応する値ノードをさらに含み、かつ各値ノードと対応するｈｏｏｋノードとの間に接続関係が存在する。

【００４９】

好ましくは、図６に示すように、
前記依存グラフが表現式により得られた新たな識別子のターゲット値ノードを含む場合、前記依存グラフにおいて前記新たな識別子に対応する値ノードを生成し、かつ前記ターゲット値ノードと前記新たな識別子の値ノードとの接続関係を確立する第二生成モジュール４０６をさらに含む。

【００５０】

好ましくは、図７に示すように、
前記関数コンポーネントの最上層の拡張可能なマークアップ言語ＸＭＬ表現式を取得する

第四取得モジュール４０７と、
前記依存グラフにおいて前記最上層のＸＭＬ表現式に対応するコンポーネントノードを生成する第三生成モジュール４０８と、
前記最上層のＸＭＬ表現式には前記依存グラフにおける値ノードの識別子を引用するターゲットＸＭＬ表現式を含む場合、前記依存グラフにおいて前記ターゲットＸＭＬ表現式と対応する値ノードとの接続関係を確立する確立モジュール４０９とをさらに含む。
【００５１】
好ましくは、前記第三生成モジュールは、前記最上層のＸＭＬ表現式の構文木を簡略化し、少なくとも一つの所定レベルのＸＭＬ表現式を取得し、かつ前記少なくとも一つの所定レベルのＸＭＬ表現式を前記依存グラフのコンポーネントノードとする。
【００５２】
本開示の実施例において、本開示は、電子装置、可読記憶媒体及びコンピュータプログラム製品を提供する。
【００５３】
図８は、本開示の実施例を実施できる例示的な電子装置８００の概略ブロック図を示す。電子装置は、様々な形式のデジタルコンピュータ、例えば、ラップトップ型コンピュータ、デスクトップコンピュータ、ワークテーブル、パーソナルデジタルアシスタント、サーバ、ブレードサーバ、大型コンピュータ、及び他の適切なコンピュータを示すことを意図する。電子装置はさらに様々な形式の移動装置、例えば、パーソナルディジタルアシスタント、携帯電話、スマートフォン、ウェアラブル装置及び他の類似の計算装置を示すことができる。本明細書に示された部材、それらの接続及び関係、及びそれらの機能は、例示的なものに過ぎず、本明細書に記載及び／又は要求された本開示の実現を限定するものではない。
【００５４】
図８に示すように、装置８００は、リードオンリーメモリ（ＲＯＭ）８０２に記憶されたコンピュータプログラム、又は記憶ユニット８０８からランダムアクセスメモリ（ＲＡＭ）８０３にロードされたコンピュータプログラムに基づいて、様々な適切な動作及び処理を実行することができる計算ユニット８０１を含む。ＲＡＭ　８０３において、さらに装置８００の動作に必要な様々なプログラム及びデータを記憶することができる。計算ユニット８０１、ＲＯＭ　８０２、およびＲＡＭ　８０３は、バス８０４により相互に接続されている。入力／出力（Ｉ／Ｏ）インタフェース８０５もバス８０４に接続されている。
【００５５】
装置８００中の複数の部材は、Ｉ／Ｏインタフェース８０５に接続され、キーボード、マウスなどの入力ユニット８０６と、様々なタイプのディスプレイ、スピーカなどの出力ユニット８０７と、磁気ディスク、光ディスクなどの記憶ユニット８０８と、ネットワークカード、モデム、無線通信トランシーバなどの通信ユニット８０９とを含む。通信ユニット８０９は、装置８００がインターネットなどのコンピュータネットワーク及び／又は各種の通信ネットワークを介して他の装置と情報／データを変換することを可能にする。
【００５６】
計算ユニット８０１は、処理及び計算能力を有する各種の汎用及び／又は専用の処理モジュールであってよい。計算ユニット８０１の例は、中央処理装置（ＣＰＵ）、グラフィック処理装置（ＧＰＵ）、各種の専用の人工知能（ＡＩ）計算チップ、各種動作機械学習モデルアルゴリズムの計算ユニット、デジタルシグナルプロセッサ（ＤＳＰ）、任意の適切なプロセッサ、コントローラ、マイクロコントローラなどを含むが、これらに限定されない。計算ユニット８０１は、上述した各方法及び処理、例えば依存グラフ生成方法を実行する。例えば、いくつかの実施例において、依存グラフ生成方法は、コンピュータソフトウェアプログラムとして実現され、記憶ユニット８０８などの機械可読媒体に有形に含まれる。いくつかの実施例において、コンピュータプログラムの一部又は全部はＲＯＭ　８０２及び／又は通信ユニット８０９を介して装置８００にロード及び／又はインストールされてよい。コンピュータプログラムがＲＡＭ８０３にロードされ、計算ユニット８０１

によって実行される場合、上述した依存グラフ生成方法の一つ又は複数のステップを実行することができる。代替的に、他の実施例において、計算ユニット８０１は、他の任意の適切な方式で（例えば、ファームウェアにより）依存グラフ生成方法を実行するように構成されてよい。

【００５７】

本明細書で上述したシステム及び技術の様々な実施形態は、デジタル電子回路システム、集積回路システム、フィールドプログラマブルゲートアレイ（ＦＰＧＡ）、専用集積回路（ＡＳＩＣ）、専用標準製品（ＡＳＳＰ）、チップ上システムのシステム（ＳＯＣ）、負荷プログラマブルロジック装置（ＣＰＬＤ）、コンピュータハードウェア、ファームウェア、ソフトウェア、及び／又はそれらの組み合わせにおいて実現することができる。これらの様々な実施形態は、一つ又は複数のコンピュータプログラムにおける実施を含んでよく、該一つ又は複数のコンピュータプログラムは、少なくとも一つのプログラマブルプロセッサを含むプログラマブルシステムで実行及び／又は解釈され、該プログラマブルプロセッサは専用又は汎用のプログラマブルプロセッサであってよく、記憶システム、少なくとも一つの入力装置、及び少なくとも一つの出力装置からデータ及び指令を受信し、データ及び指令を該記憶システム、該少なくとも一つの入力装置、及び該少なくとも一つの出力装置に伝送することができる。

【００５８】

本開示の方法を実施するプログラムコードは、一つ又は複数のプログラミング言語の任意の組み合わせで作成することができる。これらのプログラムコードは、汎用コンピュータ、専用コンピュータ又は他のプログラマブルデータ処理装置のプロセッサ又はコントローラに提供することにより、プログラムコードがプロセッサ又はコントローラによって実行されると、フローチャート及び／又はブロック図に規定された機能／動作は実施される。プログラムコードは、機器に完全に実行され、機器に部分的に実行されてもよく、独立したソフトウェアパッケージとして機器に部分的に実行されかつ遠隔機器に部分的に実行されるか又は遠隔機器又はサーバに完全に実行されてもよい。

【００５９】

本開示のコンテキストにおいて、機械可読媒体は有形の媒体であってよく、それは指令実行システム、デバイス又は装置に使用されるか又は指令実行システム、デバイス又は装置と組み合わせて使用されるプログラムを含んでよい。機械可読媒体は、機械可読信号媒体又は機械可読記憶媒体であってよい。機械可読媒体は、電子、磁気的、光学的、電磁的、赤外線、又は半導体システム、デバイス又は装置、又は上記内容の任意の適切な組み合わせを含むが、これらに限定されない。機械可読記憶媒体のより具体的な例は、一つ又は複数のワイヤに基づく電気的接続、携帯式コンピュータディスク、ハードディスク、ランダムアクセスメモリ（ＲＡＭ）、リードオンリーメモリ（ＲＯＭ）、消去可能なプログラマブルリードオンリーメモリ（ＥＰＲＯＭ又はフラッシュメモリ）、光ファイバ、携帯式コンパクトディスクリードオンリーメモリ（ＣＤ－ＲＯＭ）、光ストレージデバイス、磁気ストレージデバイス、又は上記内容の任意の適切な組み合わせを含む。

【００６０】

ユーザとの相互作用を提供するために、コンピュータにここで説明されたシステム及び技術を実施することができ、該コンピュータは、ユーザに情報を表示する表示装置（例えば、陰極線管（ＣＲＴ）又は液晶ディスプレイ（ＬＣＤ）モニタ）、及びキーボード及びポインティングデバイス（例えば、マウス又はトラックボール）を含み、ユーザは該キーボード及び該ポインティングデバイスを介して入力をコンピュータに提供することができる。他のタイプのデバイスは、さらにユーザとの相互作用を提供し、例えば、ユーザに提供されたフィードバックはいかなる形式のセンシングフィードバック（例えば、視覚フィードバック、聴覚フィードバック、又は触覚フィードバック）であってよく、いかなる形式（音声入力、音声入力又は触覚入力を含む）でユーザからの入力を受信することができる。

【００６１】

ここで説明されたシステム及び技術は、バックグラウンド部品を含むコンピューティング

システム（例えば、データサーバ）、又はミドルウェア部品を含むコンピューティングシステム（例えば、アプリケーションサーバ）、又はフロントエンド部品を含むコンピューティングシステム（例えば、グラフィカルユーザインタフェース又はウェブブラウザを有するユーザコンピュータ、ユーザが該グラフィカルユーザインタフェース又は該ネットワークブラウザを介してここで説明されたシステム及び技術の実施形態と相互作用することができる）、又はこのようなバックグラウンド部品、ミドルウェア部品、又はフロントエンド部品の任意の組み合わせを含むコンピューティングシステムに実施することができる。任意の形式又は媒体のデジタルデータ通信（例えば、通信ネットワーク）を介してシステムの部品を相互に接続することができる。通信ネットワークの例は、ローカルエリアネットワーク（ＬＡＮ）、ワイドエリアネットワーク（ＷＡＮ）及びインターネットを含む。

【００６２】

コンピュータシステムは、クライアント及びサーバを含んでよい。クライアントとサーバとは一般的に離れており、通常は通信ネットワークを介して相互作用する。クライアントとサーバとの関係は、対応するコンピュータ上で動作し、かつクライアント－サーバの関係を有するコンピュータプログラムによって生成される。

【００６３】

以上に示された様々な形式のフローを使用し、改めてステップをソーティングし、追加するか又は削除することができることを理解されたい。例えば、本開示に記載の各ステップは、並列的に実行されてもよく、順に実行されてもよく、異なる順序で実行されてもよく、本開示の技術的解決手段の所望の結果を実現することができれば、本明細書はこれを限定しない。

【００６４】

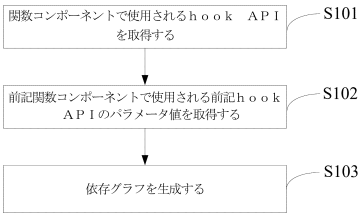
上記具体的な実施形態は、本開示の保護範囲を限定するものではない。当業者が理解できるように、設計要件及び他の要因に基づいて、様々な修正、組み合わせ、サブコンビネーション及び代替を行うことができる。本開示の精神と原則内で行われた任意の修正、均等置換及び改善などは、いずれも本開示の保護範囲内に含まれるべきである。

【００６５】

本願は２０２１年２月２６日に中国で提出された中国特許出願番号Ｎｏ．２０２１１０２１９２２２．２の優先権を主張し、その全ての内容は引用によりこれに含まれる。

【図面】

【図１】



【図２】

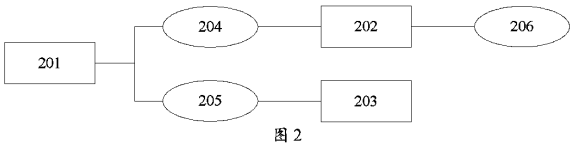
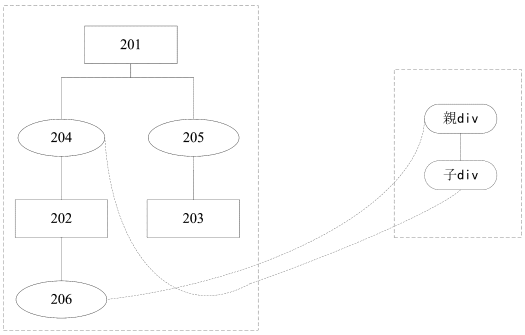
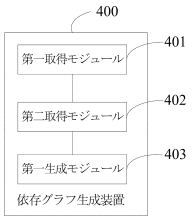


图 2

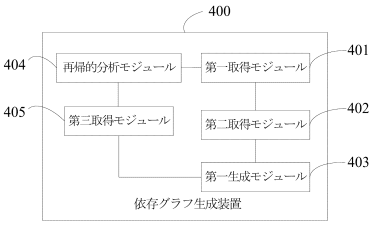
【図３】



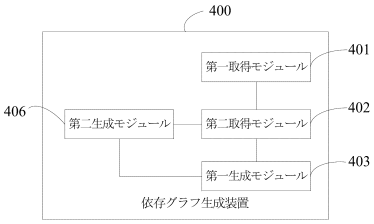
【図４】



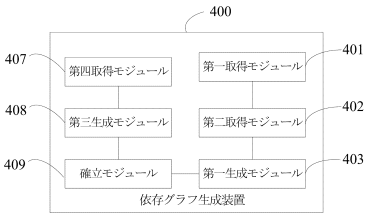
【図５】



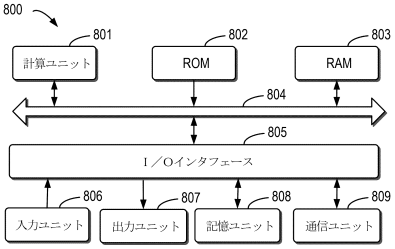
【図６】



【図７】


400
407
第四取得モジュール
第一取得モジュール
401
408
第三生成モジュール
第二取得モジュール
402
409
確立モジュール
第一生成モジュール
403
依存グラフ生成装置

【図８】


800
801
計算ユニット
802
ROM
803
RAM
804
805
I／Oインタフェース
806
入力ユニット
807
出力ユニット
808
記憶ユニット
809
通信ユニット

フロントページの続き

８５， Ｃｈｉｎａ
(74)代理人 110002468
弁理士法人後藤特許事務所
(72)発明者 魏 嘉▲ジン▼
中華人民共和国 １０００８５ 北京市海淀区上地十街１０号百度大厦２層
(72)発明者 霍 宇軒
中華人民共和国 １０００８５ 北京市海淀区上地十街１０号百度大厦２層
(72)発明者 張 聡
中華人民共和国 １０００８５ 北京市海淀区上地十街１０号百度大厦２層
(72)発明者 唐 輝
中華人民共和国 １０００８５ 北京市海淀区上地十街１０号百度大厦２層
(72)発明者 陳 澤
中華人民共和国 １０００８５ 北京市海淀区上地十街１０号百度大厦２層

審査官 多賀 実

(56)参考文献 米国特許出願公開第２０２１／００２６７５６（ＵＳ，Ａ１）
Timo McFarlane，"Managing State in React Applications with Redux"，BACHELOR ’ S THESIS [online]，フィンランド，Tampere University of Applied Sciences，2019年11月，pp.7-15，[2023.06.30 検索], インターネット:〈URL: https://www.theseus.fi/bitstream/handle/10024/265492/McFarlane_Timo.pdf〉

(58)調査した分野 (Int.Cl.，ＤＢ名)
Ｇ０６Ｆ ８／００－８／７７
Ｇ０６Ｆ ９／４４－９／４５５
Ｇ０６Ｆ １１／３６