

(19)日本国特許庁(JP)

(12)特許公報(B2)

(11)特許番号
特許第7385645号
(P7385645)

(45)発行日 令和5年11月22日(2023.11.22)

(24)登録日 令和5年11月14日(2023.11.14)

(51)国際特許分類

F I

G 0 6 F 11/36 (2006.01)

G 0 6 F 11/36 1 3 6

請求項の数 21 (全46頁)

(21)出願番号	特願2021-502700(P2021-502700)	(73)特許権者	509123208
(86)(22)出願日	平成31年3月13日(2019.3.13)		アビニシオ テクノロジー エルエルシー
(65)公表番号	特表2021-520587(P2021-520587 A)		アメリカ合衆国 0 2 4 2 1 マサチュー セッツ州 レキシントン スプリング ス トリート 2 0 1
(43)公表日	令和3年8月19日(2021.8.19)	(74)代理人	100108453
(86)国際出願番号	PCT/US2019/022066		弁理士 村山 靖彦
(87)国際公開番号	WO2019/190750	(74)代理人	100110364
(87)国際公開日	令和1年10月3日(2019.10.3)		弁理士 実広 信哉
審査請求日	令和4年3月14日(2022.3.14)	(74)代理人	100133400
(31)優先権主張番号	15/942,022		弁理士 阿部 達彦
(32)優先日	平成30年3月30日(2018.3.30)	(72)発明者	ジョイス・エル・ヴィニョー
(33)優先権主張国・地域又は機関	米国(US)		アメリカ合衆国・マサチューセッツ・0 2 4 2 1・レキシントン・スプリング・ ストリート・2 0 1
		最終頁に続く	

(54)【発明の名称】 制御フローを指定する実行可能な制御フローグラフのデバッグ

(57)【特許請求の範囲】
【請求項1】

1つまたは複数のコンピュータシステムによって実装されるコンピュータ実装方法であって、

制御フローアプリケーションの機能モジュールを識別するステップであって、1つまたは複数の機能モジュール、1つまたは複数の制御アプリケーションを特定する前記識別された機能モジュール、および方法を含む前記識別された機能モジュールを識別し、前記方法が、実行されたとき、前記識別された機能モジュールによって特定される1つまたは複数の制御アプリケーションを立ち上げる、ステップと、

前記1つまたは複数の機能モジュールを表す1つまたは複数のグラフィカルプログラム要素を含む仮想プログラムに、ブレイクポイントインジケータを挿入するステップであって、前記ブレイクポイントインジケータが、前記識別された機能モジュールを表すグラフィカル要素と並列に前記ブレイクポイントインジケータを前記仮想プログラムに挿入することによって前記制御フローアプリケーションの実行が中断されることになる実行の時点を指定する、ステップと

を含み、

前記実行の時点が、前記グラフィカルプログラム要素によって表される前記識別された機能モジュールの内容が実行される状態への遷移を含む所与の遷移、前記グラフィカルプログラム要素によって表される前記識別された機能モジュールからの遷移、または前記グラフィカルプログラム要素によって表される前記識別された機能モジュールへの遷移を表

し、前記コンピュータ実装方法が、
実行環境において前記制御フローアプリケーションを実行するステップであって、
前記遷移によって指定される実行の時点において、

前記識別された機能モジュールの前記方法を実行し、

前記方法を実行することに応答して、前記識別された機能モジュールによって特定
された前記1つまたは複数の制御アプリケーションを立ち上げるステップと、
前記ブレークポイントインジケータにより指定される実行の時点において、
前記制御フローアプリケーションの実行を中断し、
前記制御フローアプリケーションの前記識別された機能モジュールが実行されている前記
実行環境の1つ又は複数の属性を決定するステップと
を含む、実行するステップ

をさらに含む、コンピュータ実装方法。

【請求項2】

前記制御フローアプリケーションが、階層レベルの第1の階層において表される前記制
御フローアプリケーション自体を用いて、および前記制御フローアプリケーションを含む
前記階層レベルの前記第1の階層と区別される前記階層レベルの第2の階層において表さ
れる各機能モジュールを用いて複数の階層レベルとして表される、請求項1に記載のコン
ピュータ実装方法。

【請求項3】

機能モジュールが、前記機能モジュールの1つまたは複数のアクションを実行する方法
を実施するように構成され、実施されるように構成された前記方法のそれぞれが、前記階
層レベルの前記第1の階層および前記階層レベルの前記第2の階層とは異なる前記階層レ
ベルの第3の階層において表される、請求項2に記載のコンピュータ実装方法。

【請求項4】

前記特定された位置が第1の特定された位置であり、前記制御フローアプリケーション
が、実施されるように構成された前記方法のうちの遷移を含み、前記方法が、

前記機能モジュールの所与の方法への遷移を表す第2の特定された位置において前記
制御フローアプリケーションの実行を、前記所与の方法の内容の実行または前記所与の方
法からの遷移の前に中断するように構成するステップと、

前記第2の特定された位置を表す実行の時点において、前記制御フローアプリケーシ
ョンの実行を停止し、かつ前記所与の方法が実行されるランタイム環境の1つまたは複数
の属性を表すデータを提供するステップと

をさらに含む、請求項3に記載のコンピュータ実装方法。

【請求項5】

前記所与の方法の属性が前記所与の方法を実行するシステムのローカル環境を含む、請
求項4に記載のコンピュータ実装方法。

【請求項6】

第1のコンピュータシステムが前記制御フローアプリケーションを実行し、第2のコン
ピュータシステムが所与の機能モジュールの生み出されたプロセスを実行し、前記第1の
コンピュータシステムが前記第2のコンピュータシステムとは異なり、前記方法が、

前記中断に応答して、前記第2のコンピュータシステムのランタイム環境の1つまたは
複数の属性の1つまたは複数の視覚的表現をレンダリングするユーザインタフェースを表
示させるステップをさらに含む、請求項1に記載のコンピュータ実装方法。

【請求項7】

データ処理システムであって、

1つまたは複数の処理デバイスと、

動作を実施するために、前記1つまたは複数の処理デバイスによって実行可能な命令を記
憶する1つまたは複数の機械可読ハードウェア記憶デバイスと
を含み、前記動作が、

制御フローアプリケーションの機能モジュールを識別することであって、1つまたは

10

20

30

40

50

複数の機能モジュール、1つまたは複数の制御アプリケーションを特定する前記識別された機能モジュール、および方法を含む前記識別された機能モジュールを識別し、前記方法が、実行されたとき、前記識別された機能モジュールによって特定される1つまたは複数の制御アプリケーションを立ち上げる、ことと、

前記1つまたは複数の機能モジュールを表す1つまたは複数のグラフィカルプログラム要素を含む仮想プログラムに、ブレークポイントインジケータを挿入することであって、前記ブレークポイントインジケータが、前記識別された機能モジュールを表すグラフィカル要素と並列に前記ブレークポイントインジケータを前記仮想プログラムに挿入することによって前記制御フローアプリケーションの実行が中断されることになる実行の時点を指定する、ことと

を含み、

前記実行の時点が、前記グラフィカルプログラム要素によって表される前記識別された機能モジュールの内容が実行される状態への遷移を含む所与の遷移、前記グラフィカルプログラム要素によって表される前記識別された機能モジュールからの遷移、または前記グラフィカルプログラム要素によって表される前記識別された機能モジュールへの遷移を表し、

前記動作が、

実行環境において前記制御フローアプリケーションを実行することであって、前記遷移によって指定される実行の時点において、

前記識別された機能モジュールの前記方法を実行し、

前記方法を実行することに応答して、前記識別された機能モジュールによって特定された前記1つまたは複数の制御アプリケーションを立ち上げることと、

前記ブレークポイントインジケータにより指定される実行の時点において、

前記制御フローアプリケーションの実行を中断し、

前記制御フローアプリケーションの前記識別された機能モジュールが実行されている実行環境の1つ又は複数の属性を決定することと

を含む、実行すること

をさらに含む、データ処理システム。

【請求項8】

前記制御フローアプリケーションが、階層レベルの第1の階層において表される前記制御フローアプリケーション自体を用いて、および前記制御フローアプリケーションを含む前記階層レベルの前記第1の階層と区別される前記階層レベルの第2の階層において表される各機能モジュールを用いて複数の階層レベルとして表される、請求項7に記載のデータ処理システム。

【請求項9】

機能モジュールが、前記機能モジュールの1つまたは複数のアクションを実行する方法を実施するように構成され、実施されるように構成された前記方法のそれぞれが、前記階層レベルの前記第1の階層および前記階層レベルの前記第2の階層とは異なる前記階層レベルの第3の階層において表される、請求項8に記載のデータ処理システム。

【請求項10】

前記特定された位置が第1の特定された位置であり、前記制御フローアプリケーションが、実施されるように構成された前記方法のうちの遷移を含み、前記方法が、

前記機能モジュールの所与の方法への遷移を表す第2の特定された位置において前記制御フローアプリケーションの実行を、前記所与の方法の内容の実行または前記所与の方法からの遷移の前に中断するように構成するステップと、

前記第2の特定された位置を表す実行の時点において、前記制御フローアプリケーションの実行を停止し、かつ前記所与の方法が実行されるランタイム環境の1つまたは複数の属性を表すデータを提供するステップと

をさらに含む、請求項9に記載のデータ処理システム。

【請求項11】

10

20

30

40

50

前記所与の方法の属性が前記所与の方法を実行するシステムのローカル環境を含む、請求項 10 に記載のデータ処理システム。

【請求項 12】

第 1 のコンピュータシステムが前記制御フローアプリケーションを実行し、第 2 のコンピュータシステムが所与の機能モジュールの生み出されたプロセスを実行し、前記第 1 のコンピュータシステムが前記第 2 のコンピュータシステムとは異なり、前記方法が、

前記中断に応答して、前記第 2 のコンピュータシステムのランタイム環境の 1 つまたは複数の属性の 1 つまたは複数の視覚的表現をレンダリングするユーザインタフェースを表示させるステップをさらに含む、請求項 7 に記載のデータ処理システム。

【請求項 13】

動作を実施するために、1 つまたは複数の処理デバイスによって実行可能な命令を記憶する 1 つまたは複数の機械可読ハードウェア記憶デバイスであって、
前記動作が、

制御フローアプリケーションの機能モジュールを識別することであって、1 つまたは複数の機能モジュール、1 つまたは複数の制御アプリケーションを特定する前記識別された機能モジュール、および方法を含む前記識別された機能モジュールを識別し、前記方法が、実行されたとき、前記識別された機能モジュールによって特定される 1 つまたは複数の制御アプリケーションを立ち上げる、ことと、

前記 1 つまたは複数の機能モジュールを表す 1 つまたは複数のグラフィカルプログラム要素を含む仮想プログラムに、ブレークポイントインジケータを挿入することであって、
前記ブレークポイントインジケータが、前記識別された機能モジュールを表すグラフィカル要素と並列に前記ブレークポイントインジケータを前記仮想プログラムに挿入することによって前記制御フローアプリケーションの実行が中断されることになる実行の時点を指定する、ことと

を含み、

前記実行の時点が、前記グラフィカルプログラム要素によって表される前記識別された機能モジュールの内容が実行される状態への遷移を含む所与の遷移、前記グラフィカルプログラム要素によって表される前記識別された機能モジュールからの遷移、または前記グラフィカルプログラム要素によって表される前記識別された機能モジュールへの遷移を表し、

前記動作が、

実行環境において前記制御フローアプリケーションを実行することであって、
前記遷移によって指定される実行の時点において、

前記識別された機能モジュールの前記方法を実行し、

前記方法を実行することに応答して、前記識別された機能モジュールによって特定された前記 1 つまたは複数の制御アプリケーションを立ち上げることと、

前記ブレークポイントインジケータにより指定される実行の時点において、

前記制御フローアプリケーションの実行を中断し、

前記制御フローアプリケーションの前記識別された機能モジュールが実行されている実行環境の 1 つ又は複数の属性を決定することと

を含む、実行すること

をさらに含む、1 つまたは複数の機械可読ハードウェア記憶デバイス。

【請求項 14】

前記制御フローアプリケーションが、階層レベルの第 1 の階層において表される前記制御フローアプリケーション自体を用いて、および前記制御フローアプリケーションを含む前記階層レベルの前記第 1 の階層と区別される前記階層レベルの第 2 の階層において表される各機能モジュールを用いて複数の階層レベルとして表される、請求項 13 に記載の 1 つまたは複数の機械可読ハードウェア記憶デバイス。

【請求項 15】

機能モジュールが、前記機能モジュールの 1 つまたは複数のアクションを実行する方法

10

20

30

40

50

を実施するように構成され、実施されるように構成された前記方法のそれぞれが、前記階層レベルの前記第 1 の階層および前記階層レベルの前記第 2 の階層とは異なる前記階層レベルの第 3 の階層において表される、請求項 1 4 に記載の 1 つまたは複数の機械可読ハードウェア記憶デバイス。

【請求項 1 6】

前記特定された位置が第 1 の特定された位置であり、前記制御フローアプリケーションが、実施されるように構成された前記方法のうちの遷移を含み、前記方法が、

前記機能モジュールの所与の方法への遷移を表す第 2 の特定された位置において前記制御フローアプリケーションの実行を、前記所与の方法の内容の実行または前記所与の方法からの遷移の前に中断するように構成するステップと、

前記第 2 の特定された位置を表す実行の時点において、前記制御フローアプリケーションの実行を停止し、かつ前記所与の方法が実行されるランタイム環境の 1 つまたは複数の属性を表すデータを提供するステップと

をさらに含む、請求項 1 5 に記載の 1 つまたは複数の機械可読ハードウェア記憶デバイス。

【請求項 1 7】

前記所与の方法の属性が前記所与の方法を実行するシステムのローカル環境を含む、請求項 1 6 に記載の 1 つまたは複数の機械可読ハードウェア記憶デバイス。

【請求項 1 8】

第 1 のコンピュータシステムが前記制御フローアプリケーションを実行し、第 2 のコンピュータシステムが所与の機能モジュールの生み出されたプロセスを実行し、前記第 1 のコンピュータシステムが前記第 2 のコンピュータシステムとは異なり、前記方法が、

前記中断に応答して、前記第 2 のコンピュータシステムのランタイム環境の 1 つまたは複数の属性の 1 つまたは複数の視覚的表現をレンダリングするユーザインタフェースを表示させるステップをさらに含む、請求項 1 7 に記載の 1 つまたは複数の機械可読ハードウェア記憶デバイス。

【請求項 1 9】

前記制御フローアプリケーションが、複数の機能モジュールの間の制御フローを指定し、前記制御フローが、前記機能モジュールの状態に基づいて、前記機能モジュールの実行の順序についての論理を含む、請求項 1 に記載のコンピュータ実装方法。

【請求項 2 0】

前記制御フローアプリケーションが、複数の機能モジュールの間の制御フローを指定し、前記制御フローが、前記機能モジュールの状態に基づいて、前記機能モジュールの実行の順序についての論理を含む、請求項 7 に記載のデータ処理システム。

【請求項 2 1】

前記制御フローアプリケーションが、複数の機能モジュールの間の制御フローを指定し、前記制御フローが、前記機能モジュールの状態に基づいて、前記機能モジュールの実行の順序についての論理を含む、請求項 1 3 に記載の 1 つまたは複数の機械可読ハードウェア記憶デバイス。

【発明の詳細な説明】

【技術分野】

【0 0 0 1】

本説明は、複数の機能モジュールの間の制御フローを指定する実行可能な制御フローグラフをデバッグするための方法、システム及びコンピュータ可読媒体に関する。

【背景技術】

【0 0 0 2】

デバッガ又はデバッグ用ツールは、他のコンピュータプログラムのテスト及びデバッグのために使用されるコンピュータプログラムである。

【発明の概要】

【課題を解決するための手段】

【0003】

一般態様1では、複数の機能モジュールの間の制御フローを指定する実行可能な制御フローグラフをデバッグするためのコンピュータ実装方法であって、制御フローが、複数の機能モジュールの間の遷移として表され、各機能モジュールが、コンピュータシステム上の1つ又は複数のアクションを実行するように構成される、コンピュータ実装方法であり、実行可能な制御フローグラフの実行が中断されることになる実行可能な制御フローグラフにおける位置を指定するステップであって、指定位置が、所定の機能モジュールへの遷移、所定の機能モジュールのコンテンツが実行される状態への遷移又は所定の機能モジュールからの遷移を表す、ステップと、実行環境において実行可能な制御フローグラフの実行を開始するステップと、指定位置を表す実行の時点において、実行可能な制御フローグラフの実行を中断するステップと、所定の機能モジュールが実行されている実行環境の1つ又は複数の属性を表すデータを提供するステップとを含む、コンピュータ実装方法が本明細書で説明される。

10

【0004】

態様1による態様2では、実行可能な制御フローグラフは、複数の階層レベルとして表され、実行可能な制御フローグラフ自体は、階層レベルの第1のレベルで表され、各機能モジュールは、実行可能な制御フローグラフを含む階層レベルの第1のレベルとは異なる階層レベルの第2のレベルで表される。

【0005】

態様1又は2による態様3では、機能モジュールは、機能モジュールの1つ又は複数のアクションを実施するメソッドを実施するように構成され、実施されるように構成されたメソッドの各々は、階層レベルの第1のレベル及び階層レベルの第2のレベルの各々とは異なる階層レベルの第3のレベルで表される。

20

【0006】

態様1～3のいずれか1つによる態様4では、指定位置は、第1の指定位置であり、実行可能な制御フローグラフは、実施されるように構成されたメソッドの間の遷移を含み、方法は、所定のメソッドのコンテンツの実行又は所定のメソッドからの遷移の前に、機能モジュールの所定のメソッドへの遷移を表す第2の指定位置で実行可能な制御フローグラフの実行を中断するように実行可能な制御フローグラフを構成するステップと、第2の指定位置を表す実行の時点において、実行可能な制御フローグラフの実行を中断するステップと、所定のメソッドが実行されるランタイム環境の1つ又は複数の属性を表すデータを提供するステップとをさらに含む。

30

【0007】

態様1～4のいずれか1つによる態様5では、所定のメソッドの属性は、所定のメソッドを実行するシステムのローカル環境を含む。

【0008】

態様1～5のいずれか1つによる態様6では、第1のコンピュータシステムは、実行可能な制御フローグラフを実行し、第2のコンピュータシステムは、所定の機能モジュールの生み出されたプロセスを実行し、第1のコンピュータシステムは、第2のコンピュータシステムとは異なり、方法は、中断に応答して、第2のコンピュータシステムのランタイム環境の1つ又は複数の属性の1つ又は複数の視覚的表現をレンダリングするユーザインタフェースを表示させるステップをさらに含む。

40

【0009】

態様1～6のいずれか1つによる態様7では、方法は、表示デバイス上にレンダリングされる際に実行可能な制御フローグラフの中断を構成するための1つ又は複数の制御を表示するグラフィカルユーザインタフェースを提供するステップをさらに含む。

【0010】

態様1～7のいずれか1つによる態様8では、位置を指定するステップは、指定位置において実行可能な制御フローグラフにブレークポイントを挿入するステップを含む。

【0011】

50

態様 1～8 のいずれか 1 つによる態様 9 では、所定の機能モジュールは、第 1 の機能モジュールであり、方法は、実行可能な制御フローグラフが中断されている間に、第 1 の機能モジュールの 1 つ又は複数の属性の 1 つ又は複数の値を修正するステップであって、それらの 1 つ又は複数の属性の 1 つ又は複数の値が、第 1 の機能モジュールから 1 つ又は複数の第 2 の機能モジュールによってアクセスされる、ステップと、実行可能な制御フローグラフの実行を再開するステップと、1 つ又は複数の属性の 1 つ又は複数の修正値での実行を 1 つ又は複数の第 2 の機能モジュールに行わせることによって、実行可能な制御フローグラフの挙動を修正するステップとをさらに含む。

【0012】

態様 1～9 のいずれか 1 つによる態様 10 では、属性はパラメータである。

10

【0013】

態様 1～10 のいずれか 1 つによる態様 11 では、所定の機能モジュールは、条件が満たされた時点で 1 つ又は複数の第 1 のアクションを実施し、条件が満たされなかった時点で 1 つ又は複数の第 2 のアクションを実施するように構成された条件付き機能モジュールであり、方法は、条件付き機能モジュールの成功又は失敗状態を変更することによって、実行可能な制御フローグラフの制御フローを修正するステップをさらに含む。

【0014】

態様 1～11 のいずれか 1 つによる態様 12 では、実行可能な制御フローグラフは、ループを経るように構成され、方法は、実行可能な制御フローグラフの実行前、実行中又は実行後に、ループの各反復又はループの特定の反復が中断されるように実行可能な制御フローグラフを構成するステップをさらに含む。

20

【0015】

態様 1～12 のいずれか 1 つによる態様 13 では、実行可能な制御フローグラフは、ループを経るように構成された下位の実行可能な制御フローグラフを含み、方法は、ループの各反復又はループの特定の反復が中断されるように下位の実行可能な制御フローグラフを構成するステップをさらに含む。

【0016】

態様 1～13 のいずれか 1 つによる態様 14 では、方法は、特定の条件が満たされた際に指定位置において実行可能な制御フローグラフの実行が中断されるように実行可能な制御フローグラフを構成するステップをさらに含む。

30

【0017】

態様 1～14 のいずれか 1 つによる態様 15 では、方法は、いずれかの機能モジュールが失敗した時点で、実行可能な制御フローグラフの実行を中断するように実行可能な制御フローグラフを構成するステップをさらに含む。

【0018】

態様 1～15 のいずれか 1 つによる態様 16 では、所定の機能モジュールは、第 2 の機能モジュールを開始するように構成された第 1 の機能モジュールであり、方法は、第 2 の機能モジュールの実行の開始時に実行を一時停止するように実行可能な制御フローグラフを構成するステップをさらに含む。

【0019】

40

態様 1～16 のいずれか 1 つによる態様 17 では、実行可能な制御フローグラフは、グラフィック層及び制御処理層を含み、グラフィック層は、グラフィカルユーザインタフェース用の視覚的要素と、視覚的要素でプログラムされたビジュアルプログラミング言語での視覚的プログラムとを含み、ビジュアルプログラミング言語は、制御フローを指定するためのものであり、制御処理層は、ビジュアルプログラミング言語によって指定された制御フローを実施するために、機能モジュールと、メソッドと、メソッドを実行する際に使用される基本的なデータ構造と、基本的なデータ構造を使用してメソッドを実行するコードとを含む。

【0020】

態様 1～17 のいずれか 1 つによる態様 18 では、方法は、グラフィカルユーザインタ

50

フェースを視覚的要素でレンダリングすることによってグラフィック層を実行するステップと、グラフィカルユーザインタフェースを通じて、グラフィカルユーザインタフェースに表示された1つ又は複数の視覚的要素のユーザ操作を表すデータを受信するステップであって、受信データが、ビジュアルプログラミング言語での視覚的プログラムを表す、ステップと、グラフィック層で作成された視覚的プログラムに従って制御処理層を実行するステップであって、視覚的プログラムによって指定された1つ又は複数の機能モジュールを実行するサブステップを含む、制御処理層を実行するステップであり、機能モジュールを実行するサブステップが、ビジュアルプログラミング言語によって指定された制御フローを実施するためにメソッドの1つ又は複数のデータ構造を使用してコードを実行することによってその機能モジュールのメソッドを実行させる、制御処理層を実行するステップとをさらに含む。

10

【0021】

態様1～18のいずれか1つによる態様19では、特定の機能モジュールは、データ処理層に含まれる制御アプリケーションを表し、方法は、特定の機能モジュールの特定のメソッドを実行するステップと、特定のメソッドを実行するステップに応答して、制御アプリケーションを立ち上げるステップとをさらに含む。

【0022】

一般態様20では、複数の機能モジュールの間の制御フローを指定する実行可能な制御フローグラフをデバッグするためのコンピュータ実装方法であって、複数の機能モジュールが、グラフィカルプログラム要素によって表され、制御フローが、グラフィカルプログラム要素間の遷移によって表され、各機能モジュールが、コンピュータシステム上の1つ又は複数のアクションを実行するように構成される、コンピュータ実装方法であり、実行可能な制御フローグラフの実行が中断されることになる実行可能な制御フローグラフにおける位置を指定するステップであって、指定位置が、複数の機能モジュールの所定の機能モジュールへの遷移、所定の機能モジュールの1つ若しくは複数のアクションのいくつかの実行を開始する状態への遷移又は所定の機能モジュールからの遷移を表す、ステップと、実行環境において実行可能な制御フローグラフの実行を開始するステップであって、実行環境が、1つ又は複数のコンピュータをホストとし、複数の機能モジュールを実行するように構成される、ステップと、指定位置を表す実行の時点において、実行可能な制御フローグラフの実行を中断するステップと、その時点で主流である実行環境の1つ又は複数の属性を表すデータを提供するステップとを含む、コンピュータ実装方法。

20

30

【0023】

一般態様21では、複数の機能モジュールの間の制御フローを指定する実行可能な制御フローグラフをデバッグするための1つ又は複数の命令を格納する1つ又は複数の機械可読ハードウェア記憶装置であって、制御フローが、複数の機能モジュールの間の遷移として表され、各機能モジュールが、コンピュータシステム上の1つ又は複数のアクションを実行するように構成され、1つ又は複数の命令が、実行可能な制御フローグラフの実行が中断されることになる実行可能な制御フローグラフにおける位置を指定するステップであって、指定位置が、所定の機能モジュールへの遷移、所定の機能モジュールのコンテンツが実行される状態への遷移又は所定の機能モジュールからの遷移を表す、ステップと、実行環境において実行可能な制御フローグラフの実行を開始するステップと、指定位置を表す実行の時点において、実行可能な制御フローグラフの実行を中断するステップと、所定の機能モジュールが実行されている実行環境の1つ又は複数の属性を表すデータを提供するステップとを含む動作を実施するために1つ又は複数の処理デバイスによって実行可能である、1つ又は複数の機械可読ハードウェア記憶装置が本明細書で説明される。

40

【0024】

態様21による態様22では、実行可能な制御フローグラフは、複数の階層レベルとして表され、実行可能な制御フローグラフ自体は、階層レベルの第1のレベルで表され、各機能モジュールは、実行可能な制御フローグラフを含む階層レベルの第1のレベルとは異なる階層レベルの第2のレベルで表される。

50

【0025】

態様21又は22による態様23では、機能モジュールは、機能モジュールの1つ又は複数のアクションを実施するメソッドを実施するように構成され、実施されるように構成されたメソッドの各々は、階層レベルの第1のレベル及び階層レベルの第2のレベルの各々とは異なる階層レベルの第3のレベルで表される。

【0026】

態様21～23のいずれか1つによる態様24では、指定位置は、第1の指定位置であり、実行可能な制御フローグラフは、実施されるように構成されたメソッドの間の遷移を含み、1つ又は複数の動作は、所定のメソッドのコンテンツの実行又は所定のメソッドからの遷移の前に、機能モジュールの所定のメソッドへの遷移を表す第2の指定位置で実行可能な制御フローグラフの実行を中断するように実行可能な制御フローグラフを構成するステップと、第2の指定位置を表す実行の時点において、実行可能な制御フローグラフの実行を中断するステップと、所定のメソッドが実行されるランタイム環境の1つ又は複数の属性を表すデータを提供するステップとをさらに含む。

10

【0027】

態様21～24のいずれか1つによる態様25では、所定のメソッドの属性は、所定のメソッドを実行するシステムのローカル環境を含む。

【0028】

態様21～25のいずれか1つによる態様26では、第1のコンピュータシステムは、実行可能な制御フローグラフを実行し、第2のコンピュータシステムは、所定の機能モジュールの生み出されたプロセスを実行し、第1のコンピュータシステムは、第2のコンピュータシステムとは異なり、1つ又は複数の動作は、中断に応答して、第2のコンピュータシステムのランタイム環境の1つ又は複数の属性の1つ又は複数の視覚的表現をレンダリングするユーザインタフェースを表示させるステップをさらに含む。

20

【0029】

態様21～26のいずれか1つによる態様27では、1つ又は複数の動作は、表示デバイス上にレンダリングされる際に実行可能な制御フローグラフの中断を構成するための1つ又は複数の制御を表示するグラフィカルユーザインタフェースを提供するステップをさらに含む。

【0030】

態様21～27のいずれか1つによる態様28では、位置を指定するステップは、指定位置において実行可能な制御フローグラフにブレークポイントを挿入するステップを含む。

30

【0031】

態様21～28のいずれか1つによる態様29では、所定の機能モジュールは、第1の機能モジュールであり、1つ又は複数の動作は、実行可能な制御フローグラフが中断されている間に、第1の機能モジュールの1つ又は複数の属性の1つ又は複数の値を修正するステップであって、それらの1つ又は複数の属性の1つ又は複数の値が、第1の機能モジュールから1つ又は複数の第2の機能モジュールによってアクセスされる、ステップと、実行可能な制御フローグラフの実行を再開するステップと、1つ又は複数の属性の1つ又は複数の修正値での実行を1つ又は複数の第2の機能モジュールに行わせることによって、実行可能な制御フローグラフの挙動を修正するステップとをさらに含む。

40

【0032】

態様21～29のいずれか1つによる態様30では、属性はパラメータである。

【0033】

態様21～30のいずれか1つによる態様31では、所定の機能モジュールは、条件が満たされた時点で1つ又は複数の第1のアクションを実施し、条件が満たされなかった時点で1つ又は複数の第2のアクションを実施するように構成された条件付き機能モジュールであり、1つ又は複数の動作は、条件付き機能モジュールの成功又は失敗状態を変更することによって、実行可能な制御フローグラフの制御フローを修正するステップをさらに含む。

50

【0034】

態様21～31のいずれか1つによる態様32では、実行可能な制御フローグラフは、ループを経るように構成され、1つ又は複数の動作は、実行可能な制御フローグラフの実行前、実行中又は実行後に、ループの各反復又はループの特定の反復が中断されるように実行可能な制御フローグラフを構成するステップをさらに含む。

【0035】

態様21～32のいずれか1つによる態様33では、実行可能な制御フローグラフは、ループを経るように構成された下位の実行可能な制御フローグラフを含み、1つ又は複数の動作は、ループの各反復又はループの特定の反復が中断されるように下位の実行可能な制御フローグラフを構成するステップをさらに含む。

10

【0036】

態様21～33のいずれか1つによる態様34では、1つ又は複数の動作は、特定の条件が満たされた際に指定位置において実行可能な制御フローグラフの実行が中断されるように実行可能な制御フローグラフを構成するステップをさらに含む。

【0037】

態様21～34のいずれか1つによる態様35では、1つ又は複数の動作は、いずれかの機能モジュールが失敗した時点で、実行可能な制御フローグラフの実行を中断するように実行可能な制御フローグラフを構成するステップをさらに含む。

【0038】

態様21～35のいずれか1つによる態様36では、所定の機能モジュールは、第2の機能モジュールを開始するように構成された第1の機能モジュールであり、1つ又は複数の動作は、第2の機能モジュールの実行の開始時に実行を一時停止するように実行可能な制御フローグラフを構成するステップをさらに含む。

20

【0039】

態様21～36のいずれか1つによる態様37では、実行可能な制御フローグラフは、グラフィック層及び制御処理層を含み、グラフィック層は、グラフィカルユーザインタフェース用の視覚的要素と、視覚的要素でプログラムされたビジュアルプログラミング言語での視覚的プログラムとを含み、ビジュアルプログラミング言語は、制御フローを指定するためのものであり、制御処理層は、ビジュアルプログラミング言語によって指定された制御フローを実施するために、機能モジュールと、メソッドと、メソッドを実行する際に使用される基本的なデータ構造と、基本的なデータ構造を使用してメソッドを実行するコードとを含む。

30

【0040】

態様21～37のいずれか1つによる態様38では、1つ又は複数の動作は、グラフィカルユーザインタフェースを視覚的要素でレンダリングすることによってグラフィック層を実行するステップと、グラフィカルユーザインタフェースを通じて、グラフィカルユーザインタフェースに表示された1つ又は複数の視覚的要素のユーザ操作を表すデータを受信するステップであって、受信データが、ビジュアルプログラミング言語での視覚的プログラムを表す、ステップと、グラフィック層で作成された視覚的プログラムに従って制御処理層を実行するステップであって、視覚的プログラムによって指定された1つ又は複数の機能モジュールを実行するサブステップを含む、制御処理層を実行するステップであり、機能モジュールを実行するサブステップが、ビジュアルプログラミング言語によって指定された制御フローを実施するためにメソッドの1つ又は複数のデータ構造を使用してコードを実行することによってその機能モジュールのメソッドを実行させる、制御処理層を実行するステップとをさらに含む。

40

【0041】

態様21～38のいずれか1つによる態様39では、特定の機能モジュールは、データ処理層に含まれる制御アプリケーションを表し、1つ又は複数の動作は、特定の機能モジュールの特定のメソッドを実行するステップと、特定のメソッドを実行するステップにตอบสนองして、制御アプリケーションを立ち上げるステップとをさらに含む。

50

【0042】

一般態様40では、複数の機能モジュールの間の制御フローを指定する実行可能な制御フローグラフをデバッグするための1つ又は複数の命令を格納する1つ又は複数の機械可読であって、複数の機能モジュールが、グラフィカルプログラム要素によって表され、制御フローが、グラフィカルプログラム要素間の遷移によって表され、各機能モジュールが、コンピュータシステム上の1つ又は複数のアクションを実行するように構成され、1つ又は複数の命令が、実行可能な制御フローグラフの実行が中断されることになる実行可能な制御フローグラフにおける位置を指定するステップであって、指定位置が、複数の機能モジュールの所定の機能モジュールへの遷移、所定の機能モジュールの1つ若しくは複数のアクションのいくつかの実行を開始する状態への遷移又は所定の機能モジュールからの遷移を表す、ステップと、実行環境において実行可能な制御フローグラフの実行を開始するステップであって、実行環境が、1つ又は複数のコンピュータをホストとし、複数の機能モジュールを実行するように構成される、ステップと、指定位置を表す実行の時点において、実行可能な制御フローグラフの実行を中断するステップと、その時点で主流である実行環境の1つ又は複数の属性を表すデータを提供するステップとを含む動作を実施するために1つ又は複数の処理デバイスによって実行可能である、1つ又は複数の機械可読。

10

【0043】

一般態様41では、複数の機能モジュールの間の制御フローを指定する実行可能な制御フローグラフをデバッグするためのデータ処理システムであって、制御フローが、複数の機能モジュールの間の遷移として表され、各機能モジュールが、コンピュータシステム上の1つ又は複数のアクションを実行するように構成される、データ処理システムであり、1つ又は複数の処理デバイスと、1つ又は複数の命令を格納する1つ又は複数の機械可読ハードウェア記憶装置とを含む、データ処理システムであり、1つ又は複数の命令が、実行可能な制御フローグラフの実行が中断されることになる実行可能な制御フローグラフにおける位置を指定するステップであって、指定位置が、所定の機能モジュールへの遷移、所定の機能モジュールのコンテンツが実行される状態への遷移又は所定の機能モジュールからの遷移を表す、ステップと、実行環境において実行可能な制御フローグラフの実行を開始するステップと、指定位置を表す実行の時点において、実行可能な制御フローグラフの実行を中断するステップと、所定の機能モジュールが実行されている実行環境の1つ又は複数の属性を表すデータを提供するステップとを含む動作を実施するために1つ又は複数の処理デバイスによって実行可能である、データ処理システムが本明細書で説明される。

20

30

【0044】

態様41による態様42では、実行可能な制御フローグラフは、複数の階層レベルとして表され、実行可能な制御フローグラフ自体は、階層レベルの第1のレベルで表され、各機能モジュールは、実行可能な制御フローグラフを含む階層レベルの第1のレベルとは異なる階層レベルの第2のレベルで表される。

【0045】

態様41又は42による態様43では、機能モジュールは、機能モジュールの1つ又は複数のアクションを実施するメソッドを実施するように構成され、実施されるように構成されたメソッドの各々は、階層レベルの第1のレベル及び階層レベルの第2のレベルの各々とは異なる階層レベルの第3のレベルで表される。

40

【0046】

態様41～43のいずれか1つによる態様44では、指定位置は、第1の指定位置であり、実行可能な制御フローグラフは、実施されるように構成されたメソッドの間の遷移を含み、1つ又は複数の動作は、所定のメソッドのコンテンツの実行又は所定のメソッドからの遷移の前に、機能モジュールの所定のメソッドへの遷移を表す第2の指定位置で実行可能な制御フローグラフの実行を中断するように実行可能な制御フローグラフを構成するステップと、第2の指定位置を表す実行の時点において、実行可能な制御フローグラフの実行を中断するステップと、所定のメソッドが実行されるランタイム環境の1つ又は複数の属性を表すデータを提供するステップとをさらに含む。

50

【0047】

態様41～44のいずれか1つによる態様45では、所定のメソッドの属性は、所定のメソッドを実行するシステムのローカル環境を含む。

【0048】

態様41～45のいずれか1つによる態様46では、第1のコンピュータシステムは、実行可能な制御フローグラフを実行し、第2のコンピュータシステムは、所定の機能モジュールの生み出されたプロセスを実行し、第1のコンピュータシステムは、第2のコンピュータシステムとは異なり、1つ又は複数の動作は、中断に応答して、第2のコンピュータシステムのランタイム環境の1つ又は複数の属性の1つ又は複数の視覚的表現をレンダリングするユーザインタフェースを表示させるステップをさらに含む。

10

【0049】

態様41～46のいずれか1つによる態様47では、1つ又は複数の動作は、表示デバイス上にレンダリングされる際に実行可能な制御フローグラフの中断を構成するための1つ又は複数の制御を表示するグラフィカルユーザインタフェースを提供するステップをさらに含む。

【0050】

態様41～47のいずれか1つによる態様48では、位置を指定するステップは、指定位置において実行可能な制御フローグラフにブレイクポイントを挿入するステップを含む。

【0051】

態様41～48のいずれか1つによる態様49では、所定の機能モジュールは、第1の機能モジュールであり、1つ又は複数の動作は、実行可能な制御フローグラフが中断されている間に、第1の機能モジュールの1つ又は複数の属性の1つ又は複数の値を修正するステップであって、それらの1つ又は複数の属性の1つ又は複数の値が、第1の機能モジュールから1つ又は複数の第2の機能モジュールによってアクセスされる、ステップと、実行可能な制御フローグラフの実行を再開するステップと、1つ又は複数の属性の1つ又は複数の修正値での実行を1つ又は複数の第2の機能モジュールに行わせることによって、実行可能な制御フローグラフの挙動を修正するステップとをさらに含む。

20

【0052】

態様41～49のいずれか1つによる態様50では、属性はパラメータである。

【0053】

態様41～50のいずれか1つによる態様51では、所定の機能モジュールは、条件が満たされた時点で1つ又は複数の第1のアクションを実施し、条件が満たされなかった時点で1つ又は複数の第2のアクションを実施するように構成された条件付き機能モジュールであり、1つ又は複数の動作は、条件付き機能モジュールの成功又は失敗状態を変更することによって、実行可能な制御フローグラフの制御フローを修正するステップをさらに含む。

30

【0054】

態様41～51のいずれか1つによる態様52では、実行可能な制御フローグラフは、ループを経るように構成され、1つ又は複数の動作は、実行可能な制御フローグラフの実行前、実行中又は実行後に、ループの各反復又はループの特定の反復が中断されるように実行可能な制御フローグラフを構成するステップをさらに含む。

40

【0055】

態様41～52のいずれか1つによる態様53では、実行可能な制御フローグラフは、ループを経るように構成された下位の実行可能な制御フローグラフを含み、1つ又は複数の動作は、ループの各反復又はループの特定の反復が中断されるように下位の実行可能な制御フローグラフを構成するステップをさらに含む。

【0056】

態様41～53のいずれか1つによる態様54では、1つ又は複数の動作は、特定の条件が満たされた際に指定位置において実行可能な制御フローグラフの実行が中断されるように実行可能な制御フローグラフを構成するステップをさらに含む。

50

【0057】

態様41～54のいずれか1つによる態様55では、1つ又は複数の動作は、いずれかの機能モジュールが失敗した時点で、実行可能な制御フローグラフの実行を中断するように実行可能な制御フローグラフを構成するステップをさらに含む。

【0058】

態様41～55のいずれか1つによる態様56では、所定の機能モジュールは、第2の機能モジュールを開始するように構成された第1の機能モジュールであり、1つ又は複数の動作は、第2の機能モジュールの実行の開始時に実行を一時停止するように実行可能な制御フローグラフを構成するステップをさらに含む。

【0059】

態様41～56のいずれか1つによる態様57では、実行可能な制御フローグラフは、グラフィック層及び制御処理層を含み、グラフィック層は、グラフィカルユーザインタフェース用の視覚的要素と、視覚的要素でプログラムされたビジュアルプログラミング言語での視覚的プログラムとを含み、ビジュアルプログラミング言語は、制御フローを指定するためのものであり、制御処理層は、ビジュアルプログラミング言語によって指定された制御フローを実施するために、機能モジュールと、メソッドと、メソッドを実行する際に使用される基本的なデータ構造と、基本的なデータ構造を使用してメソッドを実行するコードとを含む。

【0060】

態様41～57のいずれか1つによる態様58では、1つ又は複数の動作は、グラフィカルユーザインタフェースを視覚的要素でレンダリングすることによってグラフィック層を実行するステップと、グラフィカルユーザインタフェースを通じて、グラフィカルユーザインタフェースに表示された1つ又は複数の視覚的要素のユーザ操作を表すデータを受信するステップであって、受信データが、ビジュアルプログラミング言語での視覚的プログラムを表す、ステップと、グラフィック層で作成された視覚的プログラムに従って制御処理層を実行するステップであって、視覚的プログラムによって指定された1つ又は複数の機能モジュールを実行するサブステップを含む、制御処理層を実行するステップであり、機能モジュールを実行するサブステップが、ビジュアルプログラミング言語によって指定された制御フローを実施するためにメソッドの1つ又は複数のデータ構造を使用してコードを実行することによってその機能モジュールのメソッドを実行させる、制御処理層を実行するステップとをさらに含む。

【0061】

態様41～58のいずれか1つによる態様59では、特定の機能モジュールは、データ処理層に含まれる制御アプリケーションを表し、1つ又は複数の動作は、特定の機能モジュールの特定のメソッドを実行するステップと、特定のメソッドを実行するステップにตอบสนองして、制御アプリケーションを立ち上げるステップとをさらに含む。

【0062】

一般態様60では、複数の機能モジュールの間の制御フローを指定する実行可能な制御フローグラフをデバッグするためのデータ処理システムであって、複数の機能モジュールが、グラフィカルプログラム要素によって表され、制御フローが、グラフィカルプログラム要素間の遷移によって表され、各機能モジュールが、コンピュータシステム上の1つ又は複数のアクションを実行するように構成される、データ処理システムであり、1つ又は複数の処理デバイスと、1つ又は複数の命令を格納する1つ又は複数の機械可読とを含む、データ処理システムであり、1つ又は複数の命令が、実行可能な制御フローグラフの実行が中断されることになる実行可能な制御フローグラフにおける位置を指定するステップであって、指定位置が、複数の機能モジュールの所定の機能モジュールへの遷移、所定の機能モジュールの1つ若しくは複数のアクションのいくつかの実行を開始する状態への遷移又は所定の機能モジュールからの遷移を表す、ステップと、実行環境において実行可能な制御フローグラフの実行を開始するステップであって、実行環境が、1つ又は複数のコンピュータをホストとし、複数の機能モジュールを実行するように構成される、ステップ

10

20

30

40

50

と、指定位置を表す実行の時点において、実行可能な制御フローグラフの実行を中断するステップと、その時点で主流である実行環境の1つ又は複数の属性を表すデータを提供するステップとを含む動作を実施するために1つ又は複数の処理デバイスによって実行可能である、データ処理システム。

【0063】

本発明の他の特徴及び利点は、以下の説明から及び特許請求の範囲から明らかになるであろう。

【図面の簡単な説明】

【0064】

【図1A】プランの図である。

10

【図1B】プランの図である。

【図1C】プランの図である。

【図1D】プランの図である。

【図1E】プランの図である。

【図1F】プランの図である。

【図1G】プランの図である。

【図2】プランの図である。

【図3】タスク、そのメソッド、並びに、タスク及びそのメソッドに関連するブレークポイントの挿入の図である。

【図4】グラフィカル開発環境において定義されたプランの様々な側面のユーザインタフェースの例である。

20

【図5】プランデバッグの制御の例である。

【図6】プランデバッグの制御の例である。

【図7A】グラフィカル開発環境において定義されたプランの様々な側面のユーザインタフェースの例である。

【図7B】グラフィカル開発環境において定義されたプランの様々な側面のユーザインタフェースの例である。

【図8】プランの要素の属性を閲覧して編集するためのブレークポイントインスペクタの図である。

【図9A】プランの要素の属性を閲覧して編集するためのブレークポイントインスペクタの図である。

30

【図9B】プランデバッグの制御の例である。

【図9C】プランデバッグの制御の例である。

【図10】プランの実行環境の図である。

【図11】プランの図である。

【図12】タスクの図である。

【図13】グラフィカル制御フローアプリケーションの実行を中断するためのプロセスの例である。

【図14A】継承メソッドの図である。

【図14B】継承メソッドにおいてブレークポイントを設定する図である。

40

【図15】データ処理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】

【0065】

図1Aを参照すると、略図2は、プランのグラフィック部分1を示している。便宜上、限定はしないが、以下では、プラン全体（グラフィック部分1を含むもの）は、プラン1と呼ばれる。一般に、「プラン」（本明細書では、「実行可能な制御フローグラフ」又は「グラフィカル制御フローアプリケーション」とも呼ばれる）は、本明細書で使用される場合は、グラフィカルプログラム要素（各々は機能モジュールを表す）を有する視覚的プログラムを含む。次に、各機能モジュールは、制御アプリケーションを表す。

【0066】

50

プランは、機能モジュールの間の制御フローを指定する。制御フローを指定することにより、プランは、プランの様々な機能モジュールの実行の視覚的組織を提供する。一般に、プランによって指定された制御フローは、プランのグラフィカルプログラム要素間の遷移又は接続によって表される。プランは、起動又は実行することができ、実行される様々な制御アプリケーション用の機能モジュールを通じて情報をカプセル化する。

【0067】

一般に、制御アプリケーションは、プランの外部のものであり且つプランによって起動されるプログラムを含む。制御アプリケーションの例は、コンピュータプログラム、スクリプト、データフローグラフ、別のプランなどを含む。一般に、グラフィカルプログラム要素は、テキストというよりむしろ図表を用いて操作することができるビジュアルプログラミング言語（VPL）での視覚的プログラムの要素を含む。一般に、機能モジュールは、プランのデータ構造、プランのメソッド及びデータ構造でメソッドを実装するためのコードを含む。例では、機能モジュールのタイプは、実行される一連のアクション（制御アプリケーションというよりむしろ、プラン内の）を含むタスクである。便宜上、機能モジュールは、本明細書では、限定はしないが、「タスク」と呼ぶこともできる。この例では、制御フローは、プランの機能モジュールの実行の順序についての論理を表現する。一般に、機能モジュールは、条件付き論理、プログラム、ユーザスクリプト、プロセスなどを表す。機能モジュールは、データフローグラフを使用して実装するか、又は、別の実行可能な制御フローグラフ（「サブプラン」若しくは下位の実行可能な制御フローグラフと呼ばれる）を表すことができる。例えば、サブプランは、より大きなプランに埋め込まれたプランであり、組織上の便宜のためにプランにおいて使用され、様々な機能（例えば、ループ機能）を含み得る。サブプランは、多くの異なるタスク及びさらなる入れ子サブプランを表し得る。プランは、タスクをどのように接続するか並びにメソッド及びパラメータをどのように指定するかによって、タスク間の順序、関係及び通信を制御する。プランでは、タスク、起動順序、条件付き論理及びタスク間の関係は、図表を用いて表される。

【0068】

この例では、プランは、グラフィック層（以前にグラフィック部分と呼ばれたもの）及び制御処理層を含み、各層は、データ処理層（例えば、制御アプリケーション）とは異なる。グラフィック層は、グラフィカルユーザインタフェースの視覚的要素と、VPLの視覚的プログラムとを含む。制御処理層は、ビジュアルプログラミング言語によって指定された制御フローを実施するために、タスクと、メソッドと、基本的なデータ構造（メソッドを実行する際に使用される）と、基本的なデータ構造を使用してメソッドを実行するコードとを含む。

【0069】

プラン、サブプラン及びグラフィカルプログラム要素は、視覚的プログラムの一部であるグラフィカルエンティティを含み、プランのタスク及び他の機能モジュールは、グラフィカルエンティティによって表され、「実行可能な制御フローグラフ」と呼ばれる視覚的プログラムによって制御される。すなわち、実行可能な制御フローグラフは、グラフィカルアイコンの間の制御フローを指定することはないが、実際のデータ処理の実行順番を制御する。

【0070】

例では、実行可能な制御フローグラフ（又はプラン）は、機能モジュール（例えば、タスク、サブプランなど）を表すノードを有する有向グラフを含む。ノードは、機能モジュールの間の制御フローを表す有向リンク（本明細書では「遷移」とも呼ばれる）によって接続される。具体的には、有向リンクの各々は、機能モジュールの間の制御フローを指定するデータ構造によって指定される。この例では、プラン自体の実行により、プランによって指定された制御フローに従ってプランのノードによって表される基本的な制御処理層が実行される。プランは、例えば、各グラフィカルプログラム要素が異なる演算ノード上で起動されるような分散様式で、1つ又は複数の演算ノード（例えば、中央処理装置（CPU））上で起動することができ、異なる演算ノードは、ネットワーク接続によって通信

可能に相互接続される。

【0071】

実行可能な制御フローグラフの構築は、いくつかの事例では、本質的に高度に技術的なものであり得る。実行可能な制御フローグラフは、様々なサブプランやタスクの間のグラフベースの制御フローを実装することができる。実行可能な制御フローグラフは、データ記憶装置内のデータ構造によって指定され、実行可能な制御フローグラフは、複数のノードを有し、複数のノードは、データ構造によって指定され、1つ又は複数のリンクによって接続されたサブプラン及び／又はタスクを表すものであり、リンクは、データ構造によって指定され、ノード間のデータフローを表すものである。データ記憶装置に結合され、1つ又は複数のコンピュータをホストとするランタイム環境（又は実行環境）を使用することができる。ランタイム環境は、事前実行モジュールを含み、事前実行モジュールは、実行可能な制御フローグラフを指定する格納されたデータ構造を読み取り、事前実行モジュールによって実行可能な制御フローグラフに割り当てられたタスク及びサブプランの演算を実施するためのプロセスなどのコンピューティング資源を割り当てる及び構成するように構成される。ランタイム環境は、実行モジュールを含み、実行モジュールは、本明細書で説明されるメソッドのいずれか1つが実行されるように割り当てられたプロセスの実行をスケジュール及び制御するためのものである。

10

【0072】

異なるタイプのタスクが存在し、例えば、プログラムタスク（例えば、指定されたプログラムを表すタスクであり、プランを起動しているプロセスとは異なるプロセス及び／又はプランの実行可能コードの自動組込みを可能にするプロセスである）、プランタスク（例えば、子プランを表すタスクであり、プランを起動しているプロセスとは異なるプロセス（別のプランの実行のためのもの）である）、グラフタスク（例えば、データフローグラフを表すタスクであり、プランを起動しているプロセスとは異なるプロセス及び／又はプラン内のものとは異なる実行可能なデータフローグラフを起動するように構成されたプロセスでもある）並びにプランの条件付き論理を実装する条件付きタスクを含む。この例では、各タスクは、コンピュータシステム上で実行される順番付けられた一連のアクションを表す。

20

【0073】

タスクの主なアクションは、その実施メソッドと呼ばれる。例えば、実施メソッドは、タスクの主な機能を実施するために実装することができる。実施メソッドに加えて、各タスクは、任意選択により、以下でさらに詳細に説明されるように、特定の時間に又は特定の条件に応答して他のアクションを実行することができる。実施メソッドは、タスクにそのアイデンティティを与える。残りのタスクは、実施メソッドをサポートする。様々なオプションのアクションは、実施メソッドの実行、その結果についての報告及びその後処理に備えるか又はセットアップする。いくつかの例では、プランは、様々なタスク及びサブプラン並びにこれらの様々なタスク及びサブプランの間の関係を表す論理を格納する基本的なデータ構造を含む。

30

【0074】

この例では、プランの実行により、プランの様々なタスク及びメソッドに対する適切なオブジェクトをインスタンス化することによって、基本的なデータ構造の実行又は起動が行われる。例えば、システムは、プランの階層的表現を含むファイルにプランのコンテンツを格納する。階層的表現は、ノードを含み、各ノードは、プランの要素（例えば、タスク、メソッド又はサブプラン）に対応する。システムは、階層的表現のノードを走査することによってプランをインスタンス化する。各ノードに対し、システムは、そのノードによって表される要素に対するオブジェクトをインスタンス化する。例えば、ノードがタスクを表す際、システムは、そのノードに対するタスクオブジェクトをインスタンス化する。それに加えて、そのタスクのメソッドを起動することをプランが指定する際、システムは、そのメソッドに対するメソッドオブジェクトをインスタンス化する。

40

【0075】

50

図1Aの例では、プラン1は、データ記録を処理して、例えば、それらのデータ記録の処理に基づいて、どのアクション（例えば、イントラディプロセスを起動するか又は1日の終わりのポジションを計算するか）の実行が適切かを決定するためのプランである。この例では、システムは、金融サービス設定において、例えば、特定のプロセス（すなわち、イントラディプロセス）を起動するか又は1日の終わりのポジションを決定する特定の計算を実施するかを判断することによって、イントラディ取引価格を決定するか又はその日の取引を終了するための手順の一部としてプラン1を実行する。プラン1は、グラフィカルプログラム要素1a、1b、1c、1d、1e、1f、1g、1h及び遷移1i、1j、1k、1l、1m、1oを含む。グラフィカルプログラム要素1a、1b、1c、1d、1e、1f、1g、1hは、上記で紹介されるようなそれぞれのタスクを表し、遷移1i、1j、1k、1l、1m、1oは、上記で紹介されるようなプラン1によって指定された制御フローを表す。読者の便宜上、対応する図について説明する際、グラフィカルプログラム要素又は遷移を示す参照符号への言及が行われる際は常に、それぞれのタスク及び制御フローのそれぞれへの言及が直接行われる。従って、プラン1は、プログラムタスク1a及び1h、グラフタスク1b及び1d、条件付きタスク1c、プランタスク1e及び1f、並びに、サブプラン1gを含む。プログラムタスク1a及び1h、グラフタスク1b及び1d、条件付きタスク1c、並びに、プランタスク1e及び1fの各々は、本明細書では、便宜上、「タスク」と呼ぶことができる。

【0076】

この例では、タスク1bは、タスク1aが完了した後にのみ実行することができる。タスク1fは、タスク1eが完了した後にのみ実行することができる。また、プラン1は、その後、条件付きタスク1cにおいて指定される条件が評価され（「イントラディプロセスを起動すべきか？」）、「いいえ」と考えられる場合は、システムによって、サブプラン1gを反復することによって各オフィスに対して1日の終わりのポジションが計算されることも示す。「はい」の場合は、システムは、タスク1dを実行して、イントラディ起動を実施する。この例では、他のタスクが失敗した時点で、様々なタスクをトリガすることもでき、これについては、プラン1において「エラーアクション」タスク（すなわち、タスク1h）によって示されており、このタスクは、タスク2（すなわち、タスク1f）が何らかの理由で失敗した場合に起動される。

【0077】

この例では、プログラムタスク1a及び1hの各々は、外部のプログラム又は外部の若しくは埋め込みスクリプトに注目する（又は参照する）。グラフタスク1b及び1dの各々は、実行可能なデータフローグラフ（例えば、記憶装置に格納されたもの）に注目する（又は参照する）。一般に、実行可能なデータフローグラフ（また、実行可能なグラフとも呼ばれる）は、有向データフローグラフを含み、グラフの頂点は、成分（データファイル又はプロセス）を表し、グラフのリンク又は「辺」は、成分間のデータのフローを示す。そのようなグラフベースの演算を実行するためのシステムについては、参照により本明細書に組み込まれる「EXECUTING COMPUTATIONS EXPRESSED AS GRAPHS」と称する先行する米国特許第5,966,072号明細書において説明されている。

【0078】

データフローグラフの構築は、いくつかの事例では、本質的に高度に技術的なものであり得る。データフローグラフは、1つ又は複数の入力データセットから処理グラフ成分のグラフを通じて1つ又は複数の出力データセットに流れるデータにおいて実施されるグラフベースの演算を実装することができ、データフローグラフは、データ記憶装置内のデータ構造によって指定され、データフローグラフは、複数のノードを有し、複数のノードは、データ構造によって指定され、1つ又は複数のリンクによって接続されたグラフ成分を表すものであり、リンクは、データ構造によって指定され、グラフ成分間のデータフローを表すものである。データ記憶装置に結合され、1つ又は複数のコンピュータをホストとするランタイム環境（又は実行環境）を使用することができる。ランタイム環境は、事前

実行モジュールを含み、事前実行モジュールは、データフローグラフを指定する格納されたデータ構造を読み取り、事前実行モジュールによってデータフローグラフに割り当てられたグラフ成分の演算を実施するためのプロセスなどのコンピューティング資源を割り当てる及び構成するように構成される。ランタイム環境は、実行モジュールを含み、実行モジュールは、本明細書で説明されるメソッドのいずれかが1つが実行されるように割り当てられたプロセスの実行をスケジュール及び制御するためのものである。

【0079】

特定のビジネス目的を達成するために記載しているが、グラフの基本的な構造及び構築は、技術的な考慮事項に基づいて決定される。例えば、グラフ成分は、再利用性を最大化するため又は並列処理をサポートするために選択することができる。他方では、グラフをどのように及びどこで使用するかは、ビジネス上の決定であるところが多い。パラメータ化されたデータフローグラフと関連付けられたパラメータのいくつかは、ビジネスユーザがその実装の背後にある技術上の複雑性を理解する必要なく、そのユーザがデータフローグラフをカスタマイズできるようにするために使用することができる。パラメータ化されたデータフローグラフは、カスタマイズを簡素化し、再利用を容易にする。

【0080】

プランタスク1 e及び1 fの各々は、外部のファイルに位置するプランに注目する。条件付きタスク1 cは、指定された式又はプログラムに基づいて、プラン1の決定論理を実装する。サブプラン1 gは、プランの一部としてタスク（サブプラン1 gに含まれる）をまとめ、これらのタスクを反復して起動できるようにする。また、プラン1は、遷移1 iー1 oも含み、遷移1 iー1 oは、例えば、タスクの間の依存性を示し、従って、タスクの間の実行順番を制御する。すなわち、遷移1 iー1 oは、これらのタスクの間の接続を表し、従って、実行依存性を指定する。

【0081】

例えば、タスクがどの制御アプリケーションを実行するかを指定し、制御アプリケーションについての情報（例えば、場所、論証など）をカプセル化することによって、タスクは、メソッドを実行するためのフレームワークを提供する。制御アプリケーション用のプログラミングコードは、記憶装置に格納することができ、タスクによって参照することができる。これらのメソッドは、タスクのアクションを実施する。すなわち、プランは、基本的なコードソースの視覚化ではない。むしろ、プラン自体は、起動又は実行することができ、且つ、実行される様々な実行可能プログラムに対する情報又は実行される様々な実行可能プログラムを表す情報をカプセル化する視覚的プログラムである。また、プランは、タスクの間で値を渡すための属性（例えば、パラメータ）も含む。例では、パラメータの変化は、サブプラン、メソッド、及び、パラメータ値が変化したタスクと同じ（又は異なる）階層レベルの他のタスクに影響を及ぼし得る。また、プランは、タスクが起動される順番を管理する関係も指定済みである。プランは、タスクの間の接続並びにメソッド及びパラメータの仕様に従って、タスクの間の順序、関係及び通信を制御する。

【0082】

図1 Bを参照すると、略図1 pは、プラン1にブレークポイントを挿入できる様々な場所があることを示している。一般に、ブレークポイントは、実行環境においてプランの実行を一時停止又は停止するという命令（例えば、表示、指示又は仕様）を含む。実行中のプランがブレークポイントに遭遇すると、プランは、プランの内部属性（例えば、プランの動作パラメータ及び関連値、プランが使用している動的パラメータ及び関連値、そして、グラフ又はプランを立ち上げる前にブレークポイントが位置する場合は、そのグラフ又はプランに送信されることになる入力値など）の評価（例えば、自動評価又はユーザ評価）並びに／或いはプランの外部の属性（プランを実行している実行環境若しくはシステムの属性及び／又は外部の状態や資源（例えば、プランを実行している際に実行環境によって使用されるデータベースとリポジトリの状態）など）の評価を行えるように一時停止する。これらのブレークポイントは、属性のモニタリングを可能にし、それにより、実行中のプランがブレークポイントに遭遇した時点で主流である基本的なシステム（例えば、プ

10

20

30

40

50

ラン及び／又は実行環境）の内部の状態のモニタリングを可能にする。例えば、ユーザは、タスクに対して使用されるパラメータ値を閲覧することが可能になり、また、例えば、パラメータの値を変更することによって、そのタスク（及びそのタスクの下流の又はそのタスクに依存する他のタスク）の機能を変更することも可能になる。その代替として又はそれに加えて、ユーザは、プランを実行している間且つプランがブレークポイントに遭遇する前の実行環境の性能（例えば、CPU使用量、主メモリの使用量、利用可能な又は占有されたデータ記憶容量、消費された又は利用可能なネットワーク帯域幅など）を示す実行環境の動作パラメータの値などのプランの外部の側面を閲覧することが可能になる。

【0083】

この例では、ブレークポイントは、所定の機能モジュール（すなわち、タスク1b）への遷移を表す位置1qに挿入することができる。また、ブレークポイントは、タスク1bの起動中の位置1rに挿入することもできる。位置1rにおけるブレークポイントの設定は、タスク1bが、プログラムを起動する、表されているプロセス（若しくはタスク）を生み出す準備をする直前又は他の機能若しくはアクションの前に、タスク1bの実行を中断する。一般に、本明細書で説明されるシステムによるブレークポイントの設定又は挿入は、システムによるブレークポイントの記録を指す。すなわち、位置1rは、タスク1b若しくはそのアクションのいくつかの実行が開始する直前の時刻又は開始する時刻を表す。また、ブレークポイントは、タスク1bからの遷移を表す位置1sに挿入することもできる（例えば、位置1sは、タスク1bの実行が完了した後の時刻を表す）。

【0084】

図1Cを参照すると、略図3は、ブレークポイントインジケータ2aを含み、ブレークポイントインジケータ2aは、ブレークポイントがプラン1においてタスク1bに対して記録されたことを示している。ブレークポイントインジケータ2aは、1つ又は複数のブレークポイントがタスク1bにおいて記録された（すなわち、ブレークポイントはタスク1bと関連付けられる）ことを指定するが、タスク1bの開始前、起動中又は完了後のようなブレークポイントの位置は指定しない。この例では、1つ又は複数のブレークポイントが設定された位置の詳細は、「オンデマンド」で（例えば、ブレークポイントインジケータ2aを選択するか又はブレークポイントインジケータ2aにマウスオーバーすることによって）利用可能である。一般に、ブレークポイントインジケータは、1つ又は複数のブレークポイントの存在の表示又は視覚的表現である。この例では、システムがブレークポイントインジケータ2aによって表されるブレークポイントを実行すると、プラン1の実行が一時停止され、インスペクタユーザインタフェース（例えば、「ブレークポイントインスペクタ」と呼ばれる）が提示される。ブレークポイントインスペクタは、プラン実行が一時停止される際は常に提示される（例えば、以下で説明される失敗時休止（break-on-failure）モードがオンの状態で失敗が起こることによって、又は、プラン実行の過程においてユーザが設定したブレークポイントに遭遇することによって）。

【0085】

図1Dを参照すると、略図4は、タスク1bの周りの点線によって示されるように、プラン1がブレークポイントインジケータ2aによって表されるブレークポイントに遭遇したことを示している。すなわち、タスク1bは、ブレークポイントインジケータ2aによって表されるブレークポイントに達したために停止されるが、プラン1の残りは、起動を継続する。この例では、矢印1i、1j、1lは、プラン1がタスク1a、1e、1fを実行したことを表すために、極太線（例えば、図1Dに示される他の矢印の厚さと比べて）で示されている。この例では、タスク1a、1e、1fが実行され（例えば、並列で、又は、直列に順番に実行されるタスク1e及び1fと並列にタスク1aが実行される）、プラン1は、プラン実行がタスク1bで一時停止することを考慮して、できる限り先に進行する。ブレークポイントインジケータ2aによって表されるブレークポイントがプラン1に対して記録されているため、以下で図1Eを参照して説明されるように、タスク1bを実行する前にユーザがタスク1bの属性を検査するために、グラフィカルユーザインタフェース（例えば、ブレークポイントインスペクタ）が表示される。グラフィカルユーザ

10

20

30

40

50

インタフェースの表示は、プラン1がブレークポイントインジケータ2 aに遭遇すること
に
応答して、トリガするか又は着手することができる。グラフィカルユーザインタフェ
ースは、プラン1の残り（例えば、タスク1 e及び1 f）の起動を継続する間、表示す
ることができる。この例では、タスク1 cは、タスク1 bが完了及び成功するまで起動されず
、タスク1 bは、ブレークポイントインジケータ2 aによって表されるブレークポイント
において実行が停止している間は完了しない。

【0086】

図1 Eを参照すると、例えば、ブレークポイントインジケータ2 a（図1 D）によって
表されるブレークポイントの指定位置を表す実行の時点（図1 Dのプラン1の又はプラン
1を表す基本的なデータ構造の）に達した際のグラフィカルユーザインタフェース5 cが
表示されている。この例では、グラフィカルユーザインタフェース5 cは、タイトル欄の
タスク名（すなわち、「主要な日常のプロセスを起動する」）の後に実行の一時停止の理
由（すなわち、「ブレークポイント」）を表示している。また、グラフィカルユーザイン
タフェース5 cは、プラン1のタスク1 b（図1 D）に対するデータソースを指定するデ
ータソース部分4 aも含む。この例では、データソースは、例えば、「／inputsrc#dat」として解決されているため、正しく解決されていない。入力部分4 aを通じて、正しいデータソース経路を指定することができる。データソースは、本明細書では、タスクの実行の間にグラフィカルプログラム要素1 bによって表されるタスクによって使用されるパラメータの例として使用され、データソース経路は、グラフィカルユーザインタフェースの入力部分4 aを通じてユーザが修正又は補正することができるそのようなパラメータの値の例である。図1 Fを参照すると、グラフィカルユーザインタフェース5 dは、正しいデータソースが部分5 a（又は図1 Eの部分4 a）に入力された後のグラフィカルユーザインタフェース5 cのバージョンである。この例では、新しい情報（例えば、修正されたパラメータ値）が部分5 aに入力された時点で、グラフィカルユーザインタフェース5 dは、継続制御5 bを表示し、その選択により、タスク1 b（図1 D）の実行が継続され、任意選択により、グラフィカルユーザインタフェース5 c及び5 dの各々の表示が停止する。これは、プランの起動を継続しながらタスクに対する変更がどのように行われるかを示す例である。この例では、継続制御5 bが選択される。変形形態では、タスクが失敗すると、タスクによって使用される様々な値（例えば、タスクによってその実行の間に使用されるパラメータの値）及び／又は入力の修正を可能にするためのグラフィカルユーザインタフェースが表示される。修正された時点で、グラフィカルユーザインタフェースは、例えば、プランの残りの起動を依然として継続しながら、その失敗したタスクを再スタートするための制御を表示する。

【0087】

さらなる別の変形形態では、タスクが失敗した後、ユーザは、タスク1 bの開始時にブ
レークポイント（例えば、ブレークポイントインジケータ2 aによって表されるブレーク
ポイント）を設定することができる。その後、ユーザは、タスク1 bを再スタートする
ために、コマンドラインにおいてコマンド（例えば、「plan-admin restart <task>」コマンド）を実行する。次いで、上記で説明されるように、タスク1 bが開始時に一時停止すると、データソースパラメータの値を変更することができ、その結果、タスク1 bが再スタートすると、その補正值が使用される。この例では、新しい値は、プラン1のその起動に対してのみ有効である。パラメータの値を永久的に変更するには、プラン自体が編集される。

【0088】

図1 Gを参照すると、略図6は、タスク1 bの再実行（例えば、図1 Fの継続制御5 b
の選択に続いて、又は、コマンド（例えば、上記で説明されるような「plan-admin restart <task>」コマンドの実行に続いて）を示している。この例では、タスク1 bは、データソースパラメータの正しい値を使用する（ブレークポイントインジケータ2 aによって表されるブレークポイントによってタスク1 bが開始時に一時停止し、データソースパラメータの値を補正值に変更することに基づいて）。従って、プラ

ン 1 は、矢印 1 m、1 l の厚さで示されるように、タスク 1 b を通じて条件付きタスク 1 c に進行する。

【0089】

図 2 を参照すると、制御フロー図 10（プラン 10 とも呼ばれる）は、グラフィカルプログラム要素 12、14、16 と、遷移 18、20 とを含む。グラフィカルプログラム要素は、それぞれのタスク 12、14、16 を表す。遷移 18、20 は、プラン 10 によって指定された制御フローを表す。読者の便宜上、対応する図について説明する際、グラフィカルプログラム要素又は遷移を示す参照符号への言及が行われる際は常に、それぞれのタスク及び制御フローのそれぞれへの言及が直接行われる。すなわち、グラフィカルプログラム要素 12、14、16 はそれぞれ、タスク 12、14、16 と呼ばれ、遷移 18、20 はそれぞれ、接続 18、20 と呼ばれる。この例では、プラン 10 は、接続 18、20 を含み、接続 18、20 は、タスク 12 が最初に実行され、それに続いて、タスク 14 が実行され、次いで、タスク 16 が実行されることを指定する。この例では、プラン 10 の挙動は、プラン 10 の実行を停止又は一時停止させる 1 つ又は複数のブレークポイントを含むように修正することができ、それにより、プラン、タスク、サブプラン及び／又はパラメータの見直し及び／又は修正が可能になる。

【0090】

このプランデバッグ機能は、例えば、様々な階層レベルでのブレークポイントの設定を可能にすることによって、ブレークポイントの設定位置に関する柔軟性を提供する。例えば、システムは、プラン全体のレベルで、サブプランのレベルで、タスクのレベルで又はタスクのメソッドのレベルでさえ、ブレークポイントを設定することができる。それに加えて、ブレークポイントは、以下の 3 つの位置、すなわち、プラン、サブプラン、タスク又はメソッドの開始前、プラン、サブプラン、タスク又はメソッドの起動中、及び、プラン、サブプラン、タスク又はメソッドの完了後に設定することができ、その各々については、以下でさらに詳細に説明する。

【0091】

この例では、ブレークポイントは、タスク 14 の開始前の位置 22 で記録することができる。位置 22 におけるブレークポイントは、タスク 14 が開始する前に、タスク 14 の実行を停止する。プログラム、プラン及びグラフタスクの場合、この実行の停止は、資源（例えば、指定数の利用可能トークンなどの制約要素）が取得される前に起こる。また、グラフタスクの場合、この実行の停止は、実行のためにグラフがコンパイルされる前に起こる。ブレークポイントは、タスク 14 の起動中の位置 24 で記録することができる。この例では、タスク 14 は、コンピュータプログラム 28 を起動するように構成される。従って、位置 24 におけるブレークポイントは、タスク 14 がその実施メソッドを実行する前にタスク 14 の実行を中断する。位置 24 におけるブレークポイントは、プラン 10 の環境変数の修正並びにプログラム 28 において使用される入力値（例えば、入力パラメータに対する式）の閲覧及び編集を提供する。ブレークポイントは、タスク 14 が終了した後にタスク 14 の実行を停止するために、タスク 14 の完了後の位置 26 で記録することができる。

【0092】

この例では、ブレークポイントが起こり得る場所は、ブレークポイントが要求されたかどうかをチェックするのに非常に効率的な場所である。従って、ブレークポイントのチェックに起因するコンピューティング資源の消費（CPU 消費など）において測定できるほどの増加はない（又は最低限しかない）。この理由は、ブレークポイントをチェックするためにプランにおいて使用されるプログラミングコードは最低限であり、従って、演算コストが高くなることはないためである。例では、プログラムは、ブレークポイントが挿入され得る場所（例えば、プラン、サブプラン、タスク又はメソッドの開始前、起動中又は完了後を表す各場所）にプログラムが進行する度に、ブレークポイントの発生に対してハッシュ表をチェックするように構成される。例では、第 1 のプラン（「プラン A」）は、第 1 のタスク（「タスク A」）の前にブレークポイントを含む。この例では、ハッシュ表

は、タスクAの前にブレークポイントがあることを示すために、次のエントリ、すなわち、「／planA／taskA／before」を含み、そのエントリは、プランAに含まれる。従って、システムがタスクAの実行前の場所に対応するプログラムの部分を実行する際、システムは、タスクAの実行前のその場所を表すエントリが存在するかどうかを識別するために、ルックアップ（例えば、ストリングルックアップ）を実施する。この例では、そのエントリには、「／planA／taskA／before」というストリングが投入される。従って、システムは、タスクAの前にブレークポイントを実行する。別の例では、ハッシュ表は空である。この例では、システムは、ブレークポイントを指定するエントリのチェックは行わない。

【0093】

図3を参照すると、略図40は、タスクを表すグラフィカルプログラム要素42を表示している。読者の便宜上、この図について説明する際、グラフィカルプログラム要素を示す参照符号への言及が行われる際は常に、それぞれのタスクへの言及が直接行われる。従って、グラフィカルプログラム要素42は、タスク42とも呼ばれる。この例では、以前に説明されるように、ブレークポイントは、タスク42の開始前の位置44で記録することができ、別のブレークポイントは、タスク42の起動中の位置46で記録することができる。さらなる別のブレークポイントは、タスク42の完了後の位置48で記録することができる。この例では、タスク42は、スクリプトを含むか又は参照し、コンピュータシステム上で（例えば、本明細書で説明されるシステムのオペレーティングシステムにおいて）実行することができるいくつかのメソッド（以前に説明されるように、各メソッドは1つ又は複数のアクションを含む）をさらに含む。これらのメソッドは、本明細書で説明されるシステムによって提供される既存のメソッドである。この例では、タスク42は、「トリガ」メソッド49、「開始時」メソッド50、「実施」メソッド52、「成功時」メソッド54、「失敗時」メソッド56及び「ロールバック時」メソッド59を有するように構成され、各メソッドは、タスク42の1つ若しくは複数のアクションを実行するか、又は、タスク42の1つ若しくは複数の状態を表す。この例では、トリガメソッド49は、タスク42の開始時点を表すために実装することができる。トリガメソッド49は、実行を開始するための条件を含み得る。例えば、特定のファイルが存在するかどうか、フラグが真に設定されたかどうかなどを含む様々な条件が存在し得る。開始時メソッド50は、タスク42のアクションを実施するために実行をセットアップする。例えば、開始時メソッド50は、実施メソッドに向けてシステムを準備するメソッド（環境変数を所望の値に設定するか又はランタイム情報を記録するためにログファイルをセットアップするなど）として実装することができる。実施メソッド52は、例えば、タスク42によって表される実行可能プログラムを起動することによって、タスクの指定されたアクションを実施する。例えば、実施メソッド52は、タスク42の主要な機能を実施するために実装することができる。実施メソッド52がその実行の間に成功した場合は、タスク42を閉じるために成功時メソッド54が実行される。実施メソッド52がその実行の間に失敗した場合は、タスク42を閉じるために失敗時メソッド56が実行される。この例では、失敗時メソッド56の後にロールバック時メソッド59が実行される。この例では、実施メソッドの前後のメソッドは、「フレームワークメソッド」と呼ばれる。実施メソッドの前のフレームワークメソッドは、制御アプリケーションを起動するための環境（及び／又は状況）をセットアップするが、制御アプリケーションの一部ではない。また、実施メソッドの後のフレームワークメソッド（すなわち、失敗時、成功時及びロールバック時メソッド）は、プランの一部であり、制御アプリケーションの起動の結果に基づいて実行する。フレームワークメソッドは、制御アプリケーションのアクションを実施しない。これらのフレームワークメソッドは、プランの制御処理層の一部である。

【0094】

この例では、ブレークポイントは、ブレークポイント49a、49b、49cによって指定されるように、トリガメソッド49の開始前、起動中及び完了後に設定することができる。この例では、ブレークポイントは、位置50a、50b、50cによってそれぞれ

10

20

30

40

50

指定されるように、開始時メソッド 50 の開始前、起動中及び完了後に設定することができる。この例では、位置 50 a におけるブレークポイントは、開始時メソッド 50 が開始する前にタスク 42 の実行が一時停止又は中断されることを指定する。位置 50 b におけるブレークポイントは、開始時メソッド 50 の開始時にタスク 42 の実行が一時停止又は中断されることを指定する。位置 50 c におけるブレークポイントは、開始時メソッド 50 の完了時にタスク 42 の実行が一時停止又は中断されることを指定する。ブレークポイントは、位置 52 a、52 b、52 c によって指定されるように、実施メソッド 52 の開始前、起動中及び完了後に記録することができる。この例では、タスク 42 の位置 46 におけるブレークポイントの記録は、実施メソッド 52 の実行前の位置 52 a におけるブレークポイントの記録と同じである。

【0095】

それに加えて、ブレークポイントは、成功時メソッド 54 のアクションを実施する前、成功時メソッド 54 のアクションを実施している間及び成功時メソッド 54 のアクションを実施した後のそれぞれにおいて成功時メソッド 54 を中断するために、成功時メソッド 54 に対する位置 54 a、54 b、54 c で記録することができる。例では、ブレークポイントは、失敗時メソッド 56 の開始前、起動中又は完了後の（それぞれの）位置 56 a、56 b、56 c で記録することができる。例えば、ブレークポイントは、失敗時メソッド 56 を起動する前にタスク 42 の属性のセットアップを決定するために、失敗時メソッド 56 に対する位置 56 a で記録することができる。この例では、ロールバック時メソッド 59 は、例えば、これまで行われたものを失敗時点から始めて逆の実行順番でロールバックすることによって、失敗したタスクをロールバックする。例えば、本明細書で説明されるシステムは、グラフ又はプラン用の回復ファイルを格納することができる。システムは、必要に応じて、その回復ファイルを使用して（ロールバック時メソッドと共に）後処理する。例では、ブレークポイントは、ロールバック時メソッド 59 の開始前、起動中又は完了後（のそれぞれ）を表す位置 59 a、59 b、59 c で記録することができる。位置 59 a におけるブレークポイントは、ロールバック時メソッド 59 が開始する前にタスク 42 の実行が一時停止又は中断されることを指定する。位置 59 b におけるブレークポイントは、ロールバック時メソッド 59 の開始時にタスク 42 の実行が一時停止又は中断されることを指定する。位置 59 c におけるブレークポイントは、ロールバック時メソッド 59 の完了時にタスク 42 の実行が一時停止又は中断されることを指定する。変形形態では、成功時、失敗時及びロールバック時メソッドは、オプションである。従って、それらのメソッドがタスクに含まれない場合は、それらのメソッドに対してブレークポイントを設定することができなくなる。

【0096】

ブレークポイントを設定することに加えて又はブレークポイントを設定するというよりむしろ、プランは、失敗時休止モードで起動することもでき、それにより、プランが失敗に遭遇した際に、プランの実行が中断される。例えば、起動したグラフ、子プラン又はプログラムが失敗したためにグラフ、プラン又はプログラムタスクに対する実施メソッドが失敗した場合は、実施メソッドが終了した後にプランの実行は一時停止する。プランが失敗時休止モードで起動している最中に失敗に遭遇した際は、ユーザが失敗の原因及び失敗が起こった機能モジュールの属性を閲覧できるように、ブレークポイントインスペクタが表示される。

【0097】

いくつかの例では、ユーザは、失敗時休止を設定し、タスクの失敗（実施メソッドの後のブレークポイントにおけるもの）を待ち、問題を診断し、問題を解決し、次いで、プランを再起動して失敗を回避することができる。この例では、システムは、タスクを再スタートする（例えば、タスクの存続期間の任意の時点において）というコマンドを実行することができる。すなわち、その「開始前」のブレークポイントにおいてタスクが停止されたとしても依然としてコマンドをうまく発行することができる。次いで、最終的にタスクが失敗して終了すると、タスクは、以下の通り自動的に再スタートする。

【0098】

システムは、失敗時休止モードでプランを実行する（又は実施メソッドの後にブレークポイントを設定する）。ブレークポイントが設定済みの事例では、システムは、プラン実行の間、ブレークポイントとの遭遇を待つ。プランの実行が一時停止した後、ユーザ又はシステムは、失敗の理由を診断するために状態を検討する。診断に続いて、ユーザは、タスク又は実施メソッドの「開始前」のブレークポイントを設定する（又はシステムが記録する）。ユーザ又はシステムは、タスクを再スタートするというコマンドを発行し、システムは、「開始前」のブレークポイントとの遭遇を待つ（タスクが自動的に再スタートされた際）。ユーザ又はシステムは、失敗の原因を補正し（例えば、パラメータ値をオーバーライドすること、ファイルを変更することなどによって）、タスクが成功するように「開始前」のブレークポイントから継続する。

10

【0099】

特定のプラン要素（例えば、プラン、サブプラン、タスク又はメソッド）に関して実行を一時停止する特定の位置（例えば、開始前、起動中又は完了後）を指定するブレークポイントとは異なり、失敗時休止モードは、このタイプの特定のプラン要素及び位置を指定しない。むしろ、失敗時休止モードは、プランを実行する（及び失敗した時点で実行を一時停止する）ためのモードであり、特定のプラン要素を指定しない。代わりに、実行は、タスクの失敗時に自動的に一時停止する。いくつかの例では、失敗時休止モードは常に、失敗したタスクの実施メソッドの後の位置で実行を一時停止させる。

【0100】

20

図4を参照すると、グラフィカルユーザインタフェース100は、プラン（便宜上、限定はしないが、本明細書では集合的に「プラン102」と呼ばれる）のグラフィック層102を表示しており、それぞれのタスクを表すグラフィカルプログラム要素104、106、108、110を含み、また、サブプランを表すグラフィカルプログラム要素112も含む。読者の便宜上、以下の図5～7について説明する際、グラフィカルプログラム要素を示す参照符号への言及が行われる際は常に、それぞれのタスク及びサブプランへの言及が直接行われる。従って、グラフィカルプログラム要素104、106、108、110はそれぞれ、本明細書では、タスク104、106、108、110と呼ばれ、グラフィカルプログラム要素112は、本明細書では、サブプラン112と呼ばれる。

【0101】

30

この例では、タスク106は、グラフの移動及び再フォーマットタスクである（以下では、「グラフの移動及び再フォーマットタスク106」とも呼ばれる）。グラフの移動及び再フォーマットタスク106は、実行可能なデータフローグラフ（例えば、グラフの移動及び再フォーマット）を含むか又は参照する機能モジュールである。実行されるデータフローグラフは、データ処理層の一部であるのに対して、グラフの移動及び再フォーマットタスク106は、グラフィック層の一部であり、データフローグラフをどのように実行するかについての情報をカプセル化する。この例では、タスク108は、子プランであり、以下では、「簡単な依存性プラン108」とも呼ばれる。一般に、子プランは、子プランを実行するプランを起動しているプロセスとは異なる新しいプロセス（すなわち、「子プロセス」として実行されるプランである。データフローグラフ（例えば、グラフの移動及び再フォーマット）は、プランを起動しているプロセスとは異なるプロセスにおいて実行される。

40

【0102】

それに加えて、この例では、タスク104は、条件付きタスクであり、例えば、条件が満たされた時点で、1つ又は複数の第1のアクションを実施し（例えば、1つ若しくは複数のメソッドを実行することによって）、条件が満たされなかった時点で1つ又は複数の第2のアクションを実施する（例えば、1つ若しくは複数の他のメソッドを実行することによって）ように構成されたタスクを含む。例では、すべてのタスクを条件付きタスクとして構成することができる。それに加えて、条件付きタスクは、成功又は失敗以外の条件（例えば、特定の値など）を指定することができる。この例では、タスク104は、ファ

50

イルが存在するかどうかをチェックする。また、タスク104は、以下では、「ファイルが存在するかどうかをチェックするタスク104」とも呼ばれる。ファイルが存在するかどうかをチェックするタスク104は、条件付き論理の実装形態であり、特定のファイルを閉じたかどうかに応じて、その制御をグラフの移動及び再フォーマット106又は簡単な依存性プラン108に誘導する。サブプラン112は、それ自体が制御フロー図であり、プラン102を起動しているプロセスと同じプロセスにおいて実行する。

【0103】

図5を参照すると、グラフィカルユーザインタフェース120は、ブレークポイントを設定するための制御122、124、126、128、130を含む。この例では、制御122は、ユーザがブレークポイントを設定できるようにし（一般に、プランに対して）、制御124は、プランが開始する前にユーザがブレークポイントを設定できるようにする。制御126は、プランにおいてプランコンテンツが起動する前にユーザがブレークポイントを設定できるようにする。制御128は、プランを閉じる前にユーザがブレークポイントを設定できるようにする。制御130は、ユーザが様々なメソッド及び条件に対してブレークポイントを設定できるようにする。

【0104】

図6を参照すると、グラフィカルユーザインタフェース140は、タスクにおいてブレークポイントを設定するための制御を提供している。この例では、ユーザは、ファイルが存在するかどうかをチェックするタスク104（図4）を選択し、デバッグメニュー144から「ブレークポイントを設定する」オプション142を選択している。この選択に続いて、グラフィカルユーザインタフェース140は、タスク104が開始する前又はタスク104が終了した後にそれぞれブレークポイントを設定できるように、制御146、148を表示する。この例では、ユーザは、タスク104が開始する前にブレークポイントを設定するように制御146を選択している。

【0105】

図7Aを参照すると、グラフィカルユーザインタフェース160は、タスク104においてブレークポイントが記録された（ブレークポイントインジケータ164aによって示されるように）プラン102を表示している。この例では、ブレークポイントインジケータ164aは、1つ又は複数のブレークポイントがタスク104において設定されたことを指定するが、タスク104の開始前、起動中又は完了後のようなブレークポイントの位置は指定しない。それに加えて、サブプラン112においてブレークポイントが記録されており（ブレークポイントインジケータ164bによって示されるように）、この記録は、例えば、サブプラン112を選択し、次いで、「ブレークポイントを設定する」オプション142（図6）を選択することによって行われ、それにより、ユーザは、サブプラン112が開始する前にブレークポイントを設定するか又はサブプラン112が終了した後にブレークポイントを設定するかを指定することができる。この例では、ブレークポイントは、サブプラン112において、サブプラン112が開始する前に設定されている。同時に複数のブレークポイントに遭遇することができ、それにより、例えば、プランの異なる場所で同時にプランの実行を一時停止させることができる。この例では、ブレークポイントインジケータ164a、164bによって表されるブレークポイントに同時に遭遇することができる。例では、ブレークポイントが設定された時点で、デバッグメニューは、「すべてのブレークポイントを示す」という制御を提供することができ、その選択により、プランにおいて設定済みのブレークポイントのリストを有するブレークポイントエディタダイアログが表示される。このブレークポイントエディタダイアログを通じて、ユーザは、定義済みのブレークポイントのセットの追加、除去、編集及び作業を行うことができる。ブレークポイントインジケータ164a、164bによって表されるブレークポイントのうちの1つ又は複数に達した時点で、様々なブレークポイントの実行によって現在一時停止されているプラン102の場所をリストするオーバーレイ又はインタフェース（「達したブレークポイント」ペインと呼ばれる）を提示するために、グラフィカルユーザインタフェース160が更新される。

10

20

30

40

50

【0106】

図7Bを参照すると、例えば、ブレークポイントインジケータ164a、164bによって表されるブレークポイントのうちの1つ又は複数に遭遇した時点での、達したブレークポイントというユーザインタフェース170が表示されている（例えば、図7Aのグラフィカルユーザインタフェース160に近接して又は並置して）。この例では、ブレークポイントインジケータ164a、164bによって表されるブレークポイントの両方に、例えば、同時に、遭遇している。達したブレークポイントというユーザインタフェースは現在一時停止されているプランの場所をリストするという理由により、達したブレークポイントというユーザインタフェース170は、ブレークポイントインジケータ164aによって指定されるような「タスクが開始する前」のブレークポイント（部分171aのエリア172aにおいて表される）によって特定の機能モジュールにおいて（すなわち、タスク104において）プラン102が現在一時停止されていることを指定する部分171aを含む。また、達したブレークポイントというユーザインタフェース170は、ブレークポイントインジケータ164bによって指定されるような「サブプランが開始する前」のブレークポイント（部分171bのエリア172bにおいて表される）によって別の機能モジュールにおいて（すなわち、サブプラン112において）プラン102が現在一時停止されていることを指定する部分171bも含む。この例では、部分171a、171b（又は遭遇したブレークポイントのタイプを指定する部分171a、171bのそれぞれのエリア172a、172b）のうちの1つが選択された時点で、ブレークポイントインスペクタユーザインタフェースが表示され、選択部分において表される機能モジュールに対するパラメータ、環境変数などが提示される。

10

20

【0107】

図8を参照すると、ブレークポイントに達した時点でのグラフィカルユーザインタフェース180が表示されており、ブレークポイントインスペクタユーザインタフェースの例である。この例では、グラフィカルユーザインタフェース180は、ブレークポイントインジケータ164a（図7A）によって表されるブレークポイントに達し、その後に、図7Bの達したブレークポイントというユーザインタフェース170の部分171aを選択した時点で表示される。この例では、グラフィカルユーザインタフェース180は、タスク（この事例では、図7Aのタスク104である）のパラメータ及び関連パラメータ値を表示するためのパラメータタブ172を表示している。この例では、制御182、184は、様々なパラメータ値の編集を提供する。この例では、制御184の選択は、式に対する値を指定するパラメータ値の編集を提供する（すなわち、式が「0」の値を有するか又は「1」の値を有するかを指定する）。

30

【0108】

グラフィカルユーザインタフェース180は、タスク104に対する入力値を表示する入力値タブ174と、プラン102（図4及び図7A）に対する実行環境変数又はパラメータ及び関連値を表示する環境タブ176と、プランの実行の状態及び現在一時停止されている場所に対して評価される式を記載するための制御及び／又は入力ボックスを提供する評価タブ178を含む。

【0109】

タブ172、174、176、178の各々は、適切な値（例えば、変数、パラメータ、入力値、及び、環境変数又はパラメータの値）を編集するための制御を提供する。ユーザがタブ172、174、176、178のうちの1つ又は複数における値の編集を終えた時点で、ユーザは、例えば、達した可能性のあるすべてのブレークポイント場所から、プランの起動の継続を指定することができる（例えば、制御の選択を介して）。変形形態では、ユーザは、特定のブレークポイント場所を選択し、単にそのブレークポイント場所から継続することができる。

40

【0110】

図9Aを参照すると、制御184（図8）の選択後のグラフィカルユーザインタフェース190が表示されている。グラフィカルユーザインタフェース190は、パラメータ又

50

は変数の値を修正するための（例えば、式を指定するパラメータの値を修正するための）入力部分 192 を含む。この例では、入力部分 192 は、式が真（「1」の値）に評価するか又は偽（「0」の値）に評価するかを指定するパラメータ値のエントリ及び修正を可能にする。この例では、ブレイクポイントインジケータ 164 a によって表されるブレイクポイントを使用して、例えば、タスク 104 によって表されるプロセス（若しくはタスク）からタスク 106 によって表されるプロセス（若しくはタスク）への走査を強制することによって（例えば、パラメータ式が「1」の値に設定される際）又はタスク 104 からタスク 108 への走査を強制することによって（例えば、パラメータ式が「0」の値に設定される際）、制御フローを修正することができる。

【0111】

図 9 B を参照すると、例えば、図 8 及び 9 A における編集の完了後のグラフィカルユーザインタフェース 191 が表示されている。この例では、グラフィカルユーザインタフェース 191 は、継続制御 191 a を表示しており、その選択により、遭遇したブレイクポイント場所（すなわち、ブレイクポイントインジケータ 164 a、164 b によって表される）からプラン 102 の実行を継続する。

【0112】

図 9 C を参照すると、グラフィカルユーザインタフェース 192 もまた、ユーザが特定のブレイクポイント場所からプランの実行を継続するために選択することができる制御（又はユーザインタフェースの選択可能な部分）を表示している。この例では、グラフィカルユーザインタフェース 192 は、オーバーレイ 193 付きの達したブレイクポイントというユーザインタフェース 170（図 7 B で以前に示されている）を含む。達したブレイクポイントというユーザインタフェース 170 の部分 171 a のエリア 172 a の選択により、オーバーレイ 193 が提示される。オーバーレイ 193 は、制御 193 a を含み、その選択により、エリア 172 a において表されるブレイクポイント（すなわち、図 7 A のブレイクポイントインジケータ 164 a によって表されるブレイクポイント）からプラン 102 を継続するという指示がシステムに出される。

【0113】

図 10 を参照すると、実行環境 200 は、1 つ又は複数のコンピュータをホストとしている。実行環境 200 内の各要素（例えば、プラン 202、並びに、プラン 202 に含まれるサブプラン 204 及びグラフィカルプログラム要素 206（タスクを表すもの））は、パラメータが実行スコープに入るとそれらのパラメータを評価する。読者の便宜上、以下の図 10 ～ 12 について説明する際、グラフィカルプログラム要素を示す参照符号への言及が行われる際は常に、それぞれのタスクへの言及が直接行われる。従って、グラフィカルプログラム要素 206 は、以下では、タスク 206 と呼ばれる。本明細書で説明されるように、ブレイクポイントは、プランの様々なレベルでこれらのパラメータ値の閲覧及び編集を提供するために設定することができる。一般に、プラン環境は、プランの実行及び定義のためのコンピュータ化環境を含む。一般に、パラメータが実行スコープに入る順番は、以下の通りである。実行環境 200 は、実行環境パラメータ 208（例えば、恐らくはプランのサンドボックス又はプロジェクトを通じて、プランの環境において定義されるパラメータを含む）を含む。実行環境パラメータ 208 は、プラン 202 が起動を開始する前に、プラン 202 で利用可能である。プラン 202 は、プランレベルパラメータ 210（例えば、プランレベルで宣言され、プラン 202 が起動を開始する際に評価されるパラメータを含む）を含むか又はプランレベルパラメータ 210 にアクセスする。また、プラン 202 は、その実行環境（すなわち、実行環境 200）からのパラメータ（すなわち、実行環境パラメータ 208）を参照することもでき、これらの値もまた、プラン 202 が起動を開始する際に解決される。

【0114】

この例では、プラン 202 は、サブプランレベルパラメータ 212 を有するサブプラン 204 を含み、サブプランレベルパラメータ 212 は、サブプラン 204 が起動を開始する際に、システムによって解決される。この例では、サブプラン 204 は、ループシンボ

10

20

30

40

50

ル 2 1 4 によって示されるように、ループ状のサブプランである。一般に、ループ状のサブプランは、指定回数分又は条件が満たされるまでループを経るよう構成されたサブプランを含む。この例では、ループ状のサブプランは、ループを経て繰り返すことができる機能モジュールである。従って、各ループに対し、そのループの開始前、起動中又は完了後にブレークポイントを設定することができる。また、システム（又はシステム上で起動しているエンジン）は、サブプラン 2 0 4 が新しいループを開始する度に、サブプランレベルパラメータ 2 1 2 の再評価も行う。サブプラン 2 0 4 がループを経る際、そのパラメータの値は、動的なものになり、ループからループへと渡され得る。サブプラン 2 0 4 に対するブレークポイントを使用することにより、次のループでの使用のためのそれらのパラメータの値の閲覧及び修正が可能になる。しかし、サブプラン 2 0 4 が並列ループを成すように構成される場合は、パラメータ値は、各反復に対してプライベートな状態を保つ。

10

【0 1 1 5】

この例では、サブプラン 2 0 4 は、タスク 2 1 6（タスクレベル資源 2 1 8 を有する）を含み、プラン 2 0 2 は、タスク 2 0 6（タスクレベル資源 2 2 0 を有する）を含む。プランの制御のフローが特定のタスクの入力ポートに達すると、そのタスクは、実行スコープに入る。システムがタスクのメソッドを起動することができる前、システムは、そのタスクの資源を位置付けて取得する。この例では、資源へのアクセスが必要とされる隠れパラメータは別として、システムは、タスクレベルではパラメータを解決しない。従って、タスクに対して閲覧可能及び編集可能な（例えば、図 8 のタブ 1 7 2 を通じて）パラメータは、タスクに対する実施メソッドが開始する際に評価される。

20

【0 1 1 6】

この例では、タスク 2 0 6 は、トリガメソッド 2 2 2、開始時メソッド 2 2 4、実施メソッド 2 2 6 及び成功時メソッド 2 2 8 を含む。この例では、トリガメソッド 2 2 2 及び開始時メソッド 2 2 4 は、制御アプリケーションを起動するための環境（及び／又は状況）をセットアップする。実施メソッド 2 2 6 は、制御アプリケーションを起動させる。成功時メソッド 2 2 8 は、制御アプリケーションの実行の結果が制御アプリケーションの起動の成功を示すと起動する。

【0 1 1 7】

トリガメソッド 2 2 2 は、タスク 2 0 6 の開始時点を表すために実装することができる。トリガメソッド 2 2 2 は、実行を開始するための様々な条件を含み得る。これらの様々な条件は、例えば、特定のファイルが存在するかどうか、フラグが真に設定されたかどうかなどを含む。この例では、タスク 2 0 6 は、メソッドレベルパラメータ 2 3 0、2 3 2、2 3 4、2 3 6 を含む。資源が取得された後、タスクのメソッドは、実行スコープに入り始める。この例では、システムは、既定の順番でメソッドを実行する。従って、本明細書で説明されるシステムは、同じ順番でメソッドレベルパラメータを解決する。

30

【0 1 1 8】

実行スコープに入る最初のタスクのメソッドは、トリガメソッド 2 2 2 である。これが起こると、システムは、トリガメソッド 2 2 2 上で宣言されるか又はトリガメソッド 2 2 2 によって参照されるメソッドレベルパラメータ 2 3 0 を解決する。この例では、トリガメソッド 2 2 2 においてシステムによって設定されるパラメータ値は、それに続くメソッドに渡すことができる（例えば、動的パラメータとしてそれらのパラメータを宣言することによって）。トリガメソッド 2 2 2 が成功したと想定すると、実行スコープに入る次のメソッドは、開始時メソッド 2 2 4 であり、開始時メソッド 2 2 4 は、メソッドレベルパラメータ 2 3 2（例えば、トリガメソッド 2 2 2 において設定された動的パラメータ値）を含む。

40

【0 1 1 9】

この例では、実施メソッド 2 2 6 のメソッドレベルパラメータ 2 3 4 は、ユーザインタフェースにおいて（例えば、タスク特性ダイアログパラメータタブにおいて）設定される。システムは、実施メソッド 2 2 6 が実行スコープに入ると（トリガメソッド 2 2 2 及び開始時メソッド 2 2 4 のそれぞれが起動した後）、これらのパラメータを評価する。先行

50

メソッドと同様に、ブレークポイントは、パラメータ値の修正を可能にしてそれらの修正値を後続のメソッド（例えば、失敗時メソッド（図示せず）及び成功時メソッド 228）に渡すために、実施メソッド 226 の前に設定することができる。成功時メソッド 228 のメソッドレベルパラメータ 236 は、実施メソッド 226 によって設定されたパラメータ値へのアクセスを有する。この例では、タスクのメソッドが起動を終えた際、それらのローカルパラメータ値は保存されない。しかし、次のタスクは、プランパラメータ及び実行環境にエクスポートされた任意の値へのアクセスを有する。また、タスクは、動的なものであると宣言された任意のパラメータの変更値も置いていく。これらの値は、後継タスクで見ることができる。システムは、図 11 及び 12 において説明されるように、パラメータが解決されるところに対してブレークポイントを設定するように構成することができる。

10

【0120】

図 11 を参照すると、略図 300 は、プラン 308 又はサブプラン 310 に対するブレークポイントを様々な位置に設定できる（例えば、上記で説明されるように、システムによって記録できる）ことを示しており、各位置は、上記で説明されるものを含む様々なタイプのパラメータ値をモニタリング又は編集する能力を提供する。この例では、実行環境パラメータ 302 は、プラン 308 が起動する前に設定される。プラン 308 は、プランレベルパラメータ 304 を含むか又はプランレベルパラメータ 304 にアクセスする。サブプラン 310 は、サブプランレベルパラメータ 306 を含むか又はサブプランレベルパラメータ 306 にアクセスする。この例では、システムは、プラン 308 が開始する前にプラン 308 の実行を停止するために、ブレークポイントを位置 312 に設定することができる。位置 312 におけるブレークポイントの設定は、プランレベルパラメータ 304 の修正を提供する。システムは、プラン 308 がそのコンテンツを開始する直前にプラン 308 の実行を停止するために、ブレークポイントを位置 314（プラン 308 内）に設定することができる。また、システムは、プラン 308 が終了した後且つプラン 308 を閉じる前にプラン 308 の実行を停止するために、ブレークポイントを位置 316 に設定することもできる。システムは、サブプラン 310 が開始する前にサブプラン 310 の実行を停止するために、ブレークポイントを位置 318 に設定することができる。位置 318 におけるブレークポイントの設定は、サブプランレベルパラメータ 306 の修正を提供する。システムは、サブプラン 310 がそのコンテンツを開始する直前にサブプラン 310 の実行を停止するために、ブレークポイントを位置 320（サブプラン 310 内）に設定することができる。また、システム（及び／又はユーザ）は、サブプラン 310 が終了した後且つサブプラン 310 を閉じる前にサブプラン 310 の実行を停止するために、ブレークポイントを位置 322 に設定することもできる。

20

30

【0121】

図 12 を参照すると、略図 340 は、タスク 342 に対するブレークポイントを 5 つの別個の位置 344、345、346、348、350 に設定できることを示している。この例では、タスク 342 は、メソッド 349（例えば、トリガ及び開始時メソッド、各々は、オプションであり得、フレームワークメソッドである）、実施メソッド 356 並びにメソッド 352（成功時メソッド、失敗時メソッド及び／又はロールバック時メソッド、各々は、オプションであり得、フレームワークメソッドでもある）を含む。

40

【0122】

また、タスク 342 は、タスクレベル資源 358 及びメソッドレベルパラメータ 360 も含む。この例では、位置 344 におけるブレークポイントは、タスク 342 が開始する前にタスク 342 の実行を停止する。タスク 342 がプログラム、プラン又はグラフタスクである際は、このブレークポイントは、タスクレベル資源 358 が取得される前である。また、グラフタスクの場合は、このブレークポイントは、グラフがコンパイルされる前である。位置 344 におけるブレークポイントの設定は、メソッドパラメータ 360 の修正を可能にする。

【0123】

50

この例では、位置 3 4 5 は、タスク 3 4 2 の「起動中」のブレークポイント位置である（例えば、実施メソッド 5 2（図 3）の「開始前」の位置 5 2 a（図 3）と同じである）。すなわち、位置 3 4 5 におけるブレークポイントの設定は、その実施メソッドが開始する前（実施メソッドがそのスクリプト又は実行可能プログラムを起動する直前ではない）に実施メソッド（タスク 3 4 2）の実行を停止する。グラフ及びプランタスクの場合、このブレークポイント位置 3 4 5 により、ブレークポイントインスペクタが立ち上がり、ブレークポイントインスペクタを通じて、入力式（グラフ及びプランタスクによって使用又は実行される）の編集が可能になる。従って、これらの新しい式が評価され、立ち上げたグラフ又はプランに送信される。

【0 1 2 4】

また、システムは、実施メソッド 3 5 6 における位置 3 4 6 にブレークポイントを設定することもできる。すなわち、位置 3 4 6 は、実施メソッド 3 5 6 の「起動中」の位置である（例えば、図 3 の位置 5 2 b と同じ）。ブレークポイントが位置 3 4 6 に設定されると、システムは、その実施メソッドがそのスクリプト又は実行可能プログラムを起動する直前にプログラムタスクの実行を停止するか、その実施メソッドがその子プランを起動する直前にプランタスクの実行を停止するか、又は、その実施メソッドがそのグラフを起動する直前にグラフタスクの実行を停止する。プラン、グラフ及びプログラムタスクの場合、位置 3 4 6 におけるブレークポイントは、環境変数の修正（例えば、プラン、グラフ及びプログラムタスクのために生み出されるそれらのプロセスに対する環境変数の変更）を可能にする。グラフ及びプランタスクの場合、位置 3 4 6 におけるブレークポイントは、

【0 1 2 5】

この例では、実施メソッド 3 5 6 は、グラフ 3 5 4 に着手するか又はグラフ 3 5 4 を生み出すように構成される（変形形態では、実施メソッド 3 5 6 は、グラフ 3 5 4 というよりむしろ、子プランに着手するように構成することができる）。この例では、ブレークポイントは、グラフ 3 5 4 の開始時に実行を停止するために、位置 3 4 8 に設定することができ、そのグラフ 3 5 4 を新しいタブ（例えば、グラフ 3 5 4 のコンテンツを示すウィンドウ 3 7 0）で開くことができる。ウィンドウ 3 7 0 は、グラフ 3 5 4（この例ではグラフ 3 7 4）のコンテンツを表示し、実行が一時停止されていることを示す（例えば、視覚化 3 7 2 を通じて）。開いたグラフ又は子プランから、ユーザは、現在の状態を検査するか、追加のブレークポイントを設定するか、又は、グラフ若しくは子プランの実行を再開するために「継続制御」を使用することができる。このタイプのブレークポイントは、「グラフ／子プラン内の休止」又は「内部のブレークポイント」と呼ばれる。これにより、プランがグラフ又は子プランを開始し、そのグラフ又は子プランに対してデバッガが自動的に開始される（次いで、デバッガは、まさにその実行が始まる時にグラフ又は子プランを停止する）状況が作られる。例えば、この能力は、プランが起動するにつれて、実行するグラフ又は子プランが動的に決定される際（すなわち、プランの起動前の編集時にどのグラフ又は子プランを実行するか事前に決定されていない際、従って、そのプランに対してブレークポイントを明示的に決定することができない）に役立つ。この例では、内部のブレークポイントは、実際に起動されたグラフ又は子プランの実行の停止をもたらすことになる。

【0 1 2 6】

内部のブレークポイントの場合、立ち上げたプロセス用のデバッガは、自動的に開始され、G D E は、2 つの別個の個々のプロセスであるものの間の関係を本質的に表す。この理由は、1 台のマシンが親プランのプロセスと親プラン用のデバッガのプロセスとを有するためである。第 2 のマシンは、立ち上げたグラフ又は子プランのプロセスを有する。また、この第 2 のマシンは、立ち上げたグラフ又は子プラン用のデバッガのプロセスも有する。従って、あるプランのグラフは、本明細書で説明されるシステム上で起動している可能性があり、別のグラフは、別のマシン上で起動している可能性があり、ユーザは、そのマシン上のローカル環境を調査するためにグラフデバッガを使用することができる。

【0127】

別の例では、グラフ内の休止は、親タスク（例えば、タスク342）の側面を閲覧するというよりむしろ、グラフ（例えば、グラフ374）の側面及び属性を閲覧するために、切り替え（デバッグの）をGDEに行わせる。それに加えて、グラフ又は子プラン（タスク342とは別個のプロセスとして実行されているとしても）は、タスク342と同じユーザインタフェース及び同じグラフィカル開発環境（GDE）においてデバッグすることができる。この例では、システムは、タスク342が終了した後にタスク342の実行を停止するために、ブレークポイントを位置350に設定することができる。

【0128】

いくつかの例では、タスク342が停止した時点で、ユーザは、あたかも成功したかのように又はあたかも失敗したかのようにタスク342を継続すべきかどうかを指定することができる。システムは、これらのユーザ変更及び仕様を記録し、それらに基づいて行動し（例えば、ユーザ変更に従ってタスク（又はメソッド）の状態を修正することによって）、そうすることによって、次にどのタスク又はメソッドを実行するかを制御する。システムは、成功から失敗へ（その逆も同様）メソッドの状態を変更することによって、制御フローを実装する。

【0129】

それに加えて、いくつかの例では、タスクが成功したか又は失敗したか（制御が次にどこに行くかに影響を及ぼす）についての記録の修正を可能にするグラフィカルユーザインタフェースが提示される（「完了後」のブレークポイント、例えば、タスク342の位置350におけるブレークポイントにおいて）。いくつかの例では、ユーザインタフェースは、特定タイプのタスク又はメソッド（例えば、条件付きタスク）の実行後や、グラフタスク又はプランタスクの実施メソッドの後に、記録の修正を可能にする。

【0130】

それに加えて、本明細書で説明されるシステムは、特殊な位置におけるブレークポイントの設定を提供する。例えば、プラン又はサブプランは、ループ状のプラン又はループ状のサブプラン（例えば、直列ループ又は並列ループ用に構成されたもの）であり得る。一般に、ループ状のプランは、指定回数分又は条件が満たされるまでループを経るように構成されたプランを含む。ループ状のプラン又はサブプランの場合、ループの各反復において一時停止するためにブレークポイントを設定するか、又は、指定された反復においてのみ一時停止する反復特有のブレークポイント（「反復ブレークポイント」）を設定することができる。例えば、ブレークポイントを記録する際、システムは、実行の一時停止が、そのループの特定の反復に対応するプラン／サブプラン／タスク／メソッドのインスタンスに対してのみ起こることを指定することができる（例えば、それにより、プラン／サブプラン／タスク／メソッドがループ内にあり、それが理由で、実行されるその多くの別個のインスタンスが存在することを考慮して、プラン／サブプラン／タスク／メソッドのどのインスタンスにブレークポイントが適用されるか指定される）。それに加えて、システムは、条件付きブレークポイント（例えば、特定の条件が満たされた際にのみ、プラン、サブプラン又はタスクの実行を中断するように構成されたブレークポイントを含む）を設定するように構成することができる。この例では、ブレークポイントは、式を含むか又は式と関連付けられ、システムは、式が指定された値（「1」又は「0」など）を満たしている場合にのみ、ブレークポイントを実行するように構成される。システムは、指定された条件が真である場合にのみブレークポイントを停止させるために、本明細書で説明される様々なタイプのブレークポイントのいずれかに対する条件を設定するか又は指定することができる（従って、様々なタイプのブレークポイントの各々を条件付きブレークポイントにする）。例えば、反復ブレークポイントは、それに対して指定された条件を有し得る。すなわち、例えば、第2のループ実行の一部であるタスクのインスタンスにおいて設定されたブレークポイントは、条件（ループを経ることとは無関係である）が満たされた際にのみ、実行が一時停止すべきであることを指定することもできる。

【0131】

グラフィカルブランドデバッグに加えて、システムは、コマンドラインバージョンを提供する。システムは、コマンドラインデバッグを起動中のプランに接続し、図表を用いて実施することができる様々なタイプのデバッグを提供することができる。

【0132】

図13を参照すると、複数の機能モジュールの間の制御フローを指定するグラフィカル制御フローアプリケーションをデバッグする際のシステムによるプロセス400が実装されており、制御フローは、複数の機能モジュールの間の遷移として表され、各機能モジュールは、1つ又は複数のアクションを実行するように構成される。動作の際、システムは、実行可能な制御フローグラフの実行が中断されることになる実行可能な制御フローグラフにおける位置を指定する(402)。指定位置は、所定の機能モジュールへの遷移、所定の機能モジュールのコンテンツが実行される状態への遷移又は所定の機能モジュールからの遷移を表す。また、システムは、実行環境において実行可能な制御フローグラフの実行を開始する(404)。一般に、実行環境は、オペレーティングシステムによってプラン又はソフトウェアに提供される(例えば、ランタイム時に)環境である。この例では、指定位置を表す実行の時点において、システムは、実行可能な制御フローグラフの実行を中断し(406)、所定の機能モジュールが実行されている実行又はランタイム環境の1つ又は複数の属性を表すデータを提供し(408)、例えば、プランを起動している実行又はランタイム環境の内部属性(例えば、パラメータ及び関連値、プランが使用している動的パラメータ及び関連値など)、並びに、プランを起動している実行又はランタイム環境の外部属性(プランを実行しているローカル環境若しくはシステム及び外部の状態や資源(例えば、データベース、リポジトリ)など)が挙げられる。

【0133】

いくつかの例では、実行可能な制御フローグラフは、グラフィック層及び制御処理層を含み、グラフィック層は、グラフィカルユーザインタフェース用の視覚的要素と、視覚的要素でプログラムされたビジュアルプログラミング言語での視覚的プログラムとを含み、ビジュアルプログラミング言語は、制御フローを指定するためのものであり、制御処理層は、ビジュアルプログラミング言語によって指定された制御フローを実施するために、機能モジュールと、メソッドと、メソッドを実行する際に使用される基本的なデータ構造と、基本的なデータ構造を使用してメソッドを実行するコードとを含む。この例では、方法は、グラフィカルユーザインタフェースを視覚的要素でレンダリングすることによってグラフィック層を実行するステップと、グラフィカルユーザインタフェースを通じて、グラフィカルユーザインタフェースに表示された1つ又は複数の視覚的要素のユーザ操作を表すデータを受信するステップであって、受信データが、ビジュアルプログラミング言語での視覚的プログラムを表す、ステップと、グラフィック層で作成された視覚的プログラムに従って制御処理層を実行するステップであって、視覚的プログラムによって指定された1つ又は複数の機能モジュールを実行するサブステップを含む、制御処理層を実行するステップであり、機能モジュールを実行するサブステップが、ビジュアルプログラミング言語によって指定された制御フローを実施するためにメソッドの1つ又は複数のデータ構造を使用してコードを実行することによってその機能モジュールのメソッドを実行させる、制御処理層を実行するステップとも含む。この例では、特定の機能モジュールは、データ処理層に含まれる制御アプリケーションを表し、方法は、特定の機能モジュールの特定のメソッドを実行するステップと、特定のメソッドを実行するステップに応答して、制御アプリケーションを立ち上げるステップとをさらに含む。

【0134】

図13の変形形態では、グラフィカル制御フローアプリケーションを実行する前にブレイクポイントを構成することに加えて又はグラフィカル制御フローアプリケーションを実行する前にブレイクポイントを構成するというよりむしろ、ブレイクポイントの構成(例えば、設定)は、グラフィカル制御フローアプリケーションを実行している間に起こり得る。

【0135】

10

20

30

40

50

例では、本明細書で説明されるシステムは、ブレークポイントを通じてシングルステップで実行するように構成することができる。シングルステップは、ブレークポイントに到ると、そこにブレークポイントを明示的に設定する必要なく、プラン実行を中断することができる次の場所に行くようにシステムに指示する制御又は他のコマンドである。この例では、ユーザは、プランを通じてシングルステップで実行するようにシステムに指示する。この例では、システム（又はユーザ）は、シングルステップで実行する所定のレベル（例えば、タスクレベル又はメソッドレベル）を指定することになる。この例では、プランは、制御フローの複数の経路を有し得、その結果、複数の「次」の場所が生じる。従って、システムは、それらのすべての場所で実行を停止することになる。

【0136】

図14Aを参照すると、略図450は、プラン452を示しており、両方とも、部分461aにおいて定義され、部分461bにおいて実行される。この例では、プラン452は、それぞれのタスクを表すグラフィカルプログラム要素454a…454nを含む。読者の便宜上、対応する図14A、14Bについて説明する際、グラフィカルプログラム要素又は遷移を示す参照符号への言及が行われる際は常に、それぞれのタスク及び制御フローのそれぞれへの言及が直接行われる。従って、グラフィカルプログラム要素454a…454nは、以下では、タスク454a…454nと呼ばれる。

【0137】

タスク454a…454nの各々は、1つ又は複数のメソッドを含み、各メソッドは、コンピュータシステム上で実行される1つ又は複数のアクションを含む。例えば、タスク454aはメソッド456を含み、タスク454nはメソッド458を含む。メソッド456は、メソッド458と同じメソッドでも、異なるメソッドでもよい。この例では、タスク454a…454nの各々は、プラン452から継承メソッド460を継承する。一般に、継承メソッドは、プランレベルで1回定義されてプラン内のタスクに適用される（プラン内のタスクによって継承される）メソッドを含む。すなわち、継承メソッドは、プラン（又はサブプラン）レベルで定義されたアクションであり、そのレベルより低いタスクに適用される。継承メソッドの使用を通じて、システムは、プランのタスクに同じメソッドを適用することができる。継承メソッドは、タスクごとに別々にというよりむしろ、プラン全体に対して1回定義され、それにより、多くのタスクに一度に同じアクションを割り当てることが可能になる。略図450の部分461bに示されるように、システムがプラン452を実行すると、継承メソッド460は、プラン452内のすべてのタスク（すなわち、プラン452のタスク454a…454nの各々）において実行する。いくつかの例では、タスクが継承メソッドと非継承メソッドの両方を含む際、システムは、メソッドがタスクの実施メソッドの前に起こるか又は後に起こるかに応じて、異なる順番でメソッドを実行する。タスクの実施メソッドの前には、システムは、最初に継承メソッドを実行し、それに続いて、その特定のタスクに対して類似した非継承メソッドを実行する。タスクの実施メソッドの後には、システムは、最初に非継承メソッドを実行し、それに続いて、類似した継承メソッドを実行する。

【0138】

図14Bを参照すると、略図470は、プランデバッガが継承メソッド上のブレークポイントの設定を可能にすることを示している。具体的には、略図470は、プラン452を示しており、両方とも、部分471aにおいて定義され、部分471bにおいて実行される。継承メソッド460は、プランレベルで1回定義される（部分471aに示される）。実行（部分471bに示される）の間、プラン452内の各要素（すなわち、タスク454a…454nの各々）は、その特定の要素に対する継承メソッド460の特定のインスタンス（すなわち、インスタンス460a…460n）を作成することによって、そのメソッド定義を継承する。この例では、ブレークポイント（継承メソッド460のインスタンスに対するもの）は、タスク（例えば、タスク454a…454nの1つ又は複数）ごとに設定又は記録される。すなわち、ブレークポイントは、継承メソッドの特定のインスタンス上でのみ設定することができる。この例では、システムは、位置461a、4

10

20

30

40

50

6 1 b、4 6 1 c のうちの 1 つにおけるブレークポイントをそれぞれ記録することによって、タスク 4 5 4 a の継承メソッド 4 6 0 のインスタンス 4 6 0 a の開始前、起動中又は完了後にブレークポイントを記録する。この例では、システムは、位置 4 6 1 a、4 6 1 b、4 6 1 c のうちの 1 つ又は複数においてブレークポイントを記録することができる（完全性の目的のために、位置 4 6 1 a、4 6 1 b、4 6 1 c のすべてが図 1 4 B に示されているとしても）。

【0 1 3 9】

図 1 5 を参照すると、アプリケーション開発及び実行システム 5 0 0 のブロック図が示されている。本明細書で説明されるデバッグ技法は、システム 5 0 0 内で実装することができる。システム 5 0 0 は、2 つのグラフィカル開発環境、すなわち、データフローグラフィック開発に対して使用される G D E 5 1 2 と、制御フロー図開発に対して使用される G D E 5 1 4 とを含む。或いは、データフローグラフィックと制御フロー図の両方の開発に対して、1 つのグラフィカル開発環境を使用することができる。或いは、制御フロー図を開発するためにグラフィカル開発環境を使用し、データフローグラフィックに対してコマンドラインユーザインタフェースを使用することができる、その逆も同様である。

【0 1 4 0】

G D E 5 1 2 を使用することにより、データフローグラフィック 5 8 0 を含むデータ処理アプリケーション 5 1 8 が構築される。G D E 5 1 4 を使用することにより、制御フロー図 5 5 0 を含むタスク管理アプリケーション 5 1 6 が構築される。

【0 1 4 1】

また、システム 5 0 0 は、データベース 5 2 0 も含む。データベース 5 2 0 は、システム 5 0 0 の様々な種類の情報（例えば、メタデータ）の格納を提供するスケーラブルなオブジェクト指向型データベースシステムであり得る。データベース 5 2 0 は、例えば、企業メタデータデータベースであり得、グラフベースのアプリケーションの開発及び実行や、グラフベースのアプリケーションと他のシステム（例えば、オペレーティングシステム）との間のデータの交換をサポートすることができる。

【0 1 4 2】

システム 5 0 0 は、オペレーティングシステム 5 2 4 をさらに含む。オペレーティングシステム 5 2 4 は、例えば、並列動作環境であり得る。オペレーティングシステム 5 2 4 は、アプリケーション開発環境（G D E 5 1 2 及び G D E 5 1 4 など）の起動に対するサポートを提供し、開発されたアプリケーションのスケーラブルな並列及び分散実行を提供する。

【0 1 4 3】

図 1 5 では、入力データ 5 3 0 のストリームは、データがデータ処理アプリケーション 5 1 8 によって処理されている際は、データフローグラフィック 5 8 0 を通過する。データフローグラフィック 5 8 0 は、再フォーマットコンポーネント 5 8 4 である演算コンポーネントを含む。データフローグラフィック 5 8 0 では、データは、入力データセット 5 8 2 から出力データセット 5 8 6 に流れる。

【0 1 4 4】

再フォーマットした後、データは、データ処理アプリケーション 5 1 8 から流出してタスク管理アプリケーション 5 1 6 に流入し、タスク管理アプリケーション 5 1 6 を駆動するために使用され、その駆動を受けて、制御フロー図 5 5 0 が駆動される。例では、データ処理アプリケーション 5 1 8 が成功した場合は、タスクスケジューリングアプリケーション 5 1 6 は、制御フロー図 5 5 0 を立ち上げる。制御フロー図 5 5 0 は、2 つのタスク（タスク 5 5 2 及びタスク 5 5 4）を示しており、各タスクは、本明細書で説明されるように、ブレークポイント（タスクの開始前、起動中又は完了後）を有するようにプログラムすることができる。タスクは、例えば、データフローグラフィック又はスクリプト（Perl スクリプトなど）を実行することによって実施される演算であり得る。時間系列 5 5 6 は、制御フロー図 5 5 0 において指定されたタスクの起動順序を示す。この事例では、タスク 5 5 2 は、タスク 5 5 4 の前に実行される。

【0145】

図15に示されるように、制御フロー図550のタスク552は、Perlスクリプトであり、タスク554は、本明細書で説明されるシステムのオペレーティングシステム524において実行することができるいくつかのメソッドを含む。これらのメソッドは、本明細書で説明されるシステムによって提供される既存のメソッドであり得る。ユーザは、メソッドを構成する様々なパラメータ及びスクリプトに対するそれら自体のコンテンツを指定することによって、これらのメソッドの挙動をカスタマイズすることができる。ブレークポイントは、本明細書で説明されるように、これらのメソッドの各々の開始前、起動中又は完了後に挿入することができる。図15に例として示されるように、タスク554は、5つのメソッド（すなわち、トリガ、開始時、実施、失敗時及び成功時）を含む。

10

【0146】

いくつかの例では、上記の5つのメソッドは、次の通り実装することができる。

【0147】

トリガメソッドは、タスク554の開始時点を表すために実装することができる。トリガメソッドは、例えば、特定のファイルが存在するかどうか、フラグが真に設定されたかどうかなどを含む、実行を開始するための様々な条件を含み得る。

【0148】

開始時メソッドは、実施メソッドに向けてシステムを準備するメソッド（環境変数を所望の値に設定するか又はランタイム情報を記録するためにログファイルをセットアップするなど）として実装することができる。

20

【0149】

実施メソッドは、タスク554の主要な機能を実施するために実装することができる。また、タスク554は、実施メソッドの後に起こることに対処するための条件付き論理を含み得る。実施メソッドがその実行の間に成功した場合は、ゼロの戻りコードでタスク554を閉じるために成功時メソッドが実行される。実施メソッドがその実行の間に失敗した場合は、非ゼロの戻りコードでタスク554を閉じるために失敗時メソッドが実行される。任意選択により、ロールバック、エラー処理及び回復に対して追加のメソッドを追加することができる。例えば、ロールバックのメソッドは、これまで行われたものを失敗時点から始めて逆の実行順番でロールバックするために追加することができる。或いは、後処理のメソッドは、フラグ、レジスタなどをリセットすることによって、失敗した条件を後処理するために追加することができる。

30

【0150】

本明細書で説明されるように、機能モジュールは、制御アプリケーションに対する情報又は制御アプリケーションを表す情報をカプセル化し、制御アプリケーションは、実行可能なコンピュータプログラム又はコンピュータプログラムの実行可能な部分のインスタンスであり、機能モジュールは、コンピュータシステム上の1つ又は複数のアクションを実行するように構成される。制御フローは、個々の機能モジュールが実行又は評価される順番である。

【0151】

本明細書で説明される技法を使用することにより、システム及び方法は、プラン及びサブプランをデバッグする。例えば、プランデバッガは、本明細書では、これらのプランをトラブルシュートするために説明され、デバッガは、コマンドラインとグラフィカルインタフェースの両方を有する。プランデバッガは、これらのプラン内の多くのキーポイントでのプランの実行の一時停止（又は「休止」）を可能にする。このプランは、異機種環境のマシン及び技術上において分散様式で動作する。デバッガは、条件付きブレークポイント、ループの特定の反復の休止又は失敗時休止などのデバッグ及びデバッガ機能を提供する。デバッガは、プランの異なるレベル（すなわち、プラン、サブプラン、タスク又はメソッド）での休止を可能にする。休止は、本明細書で記述されている場合を除いて、上記のいずれかの実行の（a）開始前、（b）完了後、又は、（c）起動中（タスクとメソッドの事例）及び内部（グラフと子プランである制御アプリケーションの事例）に設定する

40

50

ことができる。本明細書で説明されるように、ブレークポイントを伴う設定及び起動は、プランを減速しない。それに加えて、プランは、デバッグを使用するために「デバッグモード」にする必要はない。ユーザは、プランが開始する前又はプランが起動している間にブレークポイントを設定することができる。

【0152】

本明細書で説明される技法は、多くの演算及び資源配分効率を提供する。例えば、タスクの開始前、起動中及び完了後に、また、タスクのメソッドの各々の開始前、起動中及び完了後に、ブレークポイントの設定が可能であることにより、システムは、タスク及び／又はそのメソッドをデバッグするために必要なタスクの実行回数を低減することができる。この理由は、失敗を識別するためにタスク又はプランの起動の完了を待つというよりむしろ、本明細書で説明される技法は、失敗が起こった際に失敗の識別を提供し、失敗の原因であり得る変数及び／又はパラメータ値を修正する能力を提供する。値が修正された時点で、プラン又はタスクは、以前に失敗した時点から実行を再開することができる。すなわち、「下流」のタスク及び／又はメソッド（別のタスク及び／又はメソッドに依存するタスク及び／又はメソッド）は、失敗の直後に実行され、次いで、失敗がデバッグされるにつれて繰り返し実行されるというよりむしろ、失敗が識別され、その対処が行われた時点で実行される。これにより、プランが完了するまで起動し、次いで、完了後にデバッグし、次いで、再び起動して失敗がうまく対処されたかどうかを評価することが認められている際に必要とされる処理能力と比べて、処理能力の消費が減少する（プランをデバッグするために必要なプランの実行がより少なくなるため）。また、これにより、プランをデバッグするためにプランを複数回起動する必要があるメモリ量と比べて、メモリの消費が減少する（プランは1回しか起動されず、その単一の起動の間にうまくデバッグされ得るため）。

【0153】

それに加えて、タスクのメソッドの各々の開始前、起動中及び完了後にブレークポイントの設定が可能であることによって提供される粒度レベルは、タスクのメソッドを表すデータ構造を定義する際の柔軟性の増加や、実行挙動の調査及び制御の際の柔軟性の増加を提供する。これらのデータ構造を事前に定義し、プランが実行された後にのみその値を変更する必要があるというよりむしろ、パラメータの値及び／又はこれらのデータ構造に対する値は、プランが実行されるにつれて「オンザフライ」で修正することができる。これらのオンザフライ修正は、これらの変更の影響を評価するためにプランを複数回実行する必要がないため、メモリ必要量は小さくて済む。

【0154】

上記で説明される技法は、コンピュータ上で実行するためのソフトウェアを使用して実装することができる。例えば、ソフトウェアは、1つ又は複数のプログラムされた又はプログラム可能なコンピュータシステム（分散型、クライアント／サーバ又はグリッドなどの様々なアーキテクチャのものであり得る）上で実行する1つ又は複数のコンピュータプログラムのプロシージャを形成し、コンピュータシステムの各々は、少なくとも1つのプロセッサと、少なくとも1つのデータ記憶システム（揮発性及び不揮発性のメモリ及び／又は記憶素子を含む）と、少なくとも1つの入力デバイス又はポートと、少なくとも1つの出力デバイス又はポートとを含む。ソフトウェアは、より大きなプログラムの1つ又は複数のモジュール（例えば、チャート及びフローチャートの設計及び構成に関連する他のサービスを提供するもの）を形成することができる。チャートのノード、リンク及び要素は、コンピュータ可読媒体に格納されたデータ構造又はデータリポジトリに格納されたデータモデルに適合する他の組織化データとして実装することができる。

【0155】

本明細書で説明される技法は、デジタル電子回路において、コンピュータハードウェア、ファームウェア、ソフトウェアにおいて、又は、それらの組合せにおいて実装することができる。装置は、プログラマブルプロセッサによる実行のために有形に具体化されるか又は機械可読記憶装置（例えば、非一時的な機械可読記憶装置、機械可読ハードウェア記

10

20

30

40

50

憶装置など)に格納されるコンピュータプログラム製品において実装することができ、方法アクションは、入力データに基づいて動作し且つ出力を生成することによって機能を実施するために命令のプログラムを実行するプログラマブルプロセッサによって実施することができる。本明細書で説明される実施形態並びに特許請求の範囲及び本明細書で説明される技法の他の実施形態は、データ記憶システムからデータ及び命令を受信するため及びデータ記憶システムにデータ及び命令を送信するために結合された少なくとも1つのプログラマブルプロセッサと、少なくとも1つの入力デバイスと、少なくとも1つの出力デバイスとを含むプログラマブルシステム上で実行可能な1つ又は複数のコンピュータプログラムにおいて有利に実装することができる。各コンピュータプログラムは、高レベルの手続き型若しくはオブジェクト指向型プログラミング言語で、又は、必要に応じてアセンブリ若しくは機械語で実装することができ、いかなる事例においても、言語は、コンパイラ型又はインタープリタ型言語であり得る。

【0156】

コンピュータプログラムの実行に適したプロセッサは、例として、汎用マイクロプロセッサと専用マイクロプロセッサの両方や、任意の種類のデジタルコンピュータのいずれか1つ又は複数のプロセッサを含む。一般に、プロセッサは、読み取り専用メモリ若しくはランダムアクセスメモリ又はその両方から命令及びデータを受信する。コンピュータの必須要素は、命令を実行するためのプロセッサと、命令及びデータを格納するための1つ又は複数のメモリデバイスである。また、一般に、コンピュータは、データ受信のため、データ送信のため又はその両方のために、データ格納用の1つ又は複数の大容量記憶装置(例えば、磁気ディスク、光磁気ディスク若しくは光ディスク)も含むか、或いは、データ格納用の1つ若しくは複数の大容量記憶装置に動作可能に結合される。コンピュータプログラム命令及びデータを具体化するためのコンピュータ可読媒体は、例として、半導体メモリデバイス(例えば、EPROM、EEPROM、フラッシュメモリデバイス)、磁気ディスク(例えば、内部ハードディスク又はリムーバブルディスク)、光磁気ディスク並びにCD-ROM及びDVD-ROMディスクを含む、不揮発性メモリのすべての形態を含む。プロセッサ及びメモリは、専用論理回路によって補充することも、専用論理回路に組み込むこともできる。前述のいずれも、ASIC(特定用途向け集積回路)によって補充することも、ASIC(特定用途向け集積回路)に組み込むこともできる。

【0157】

ユーザとの対話を提供するため、実施形態は、ユーザに情報を表示するための表示デバイス(例えば、LCD(液晶ディスプレイ)モニタ)や、ユーザがコンピュータに入力を提供することができる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)を有するコンピュータ上で実装することができる。他の種類のデバイスを使用してユーザとの対話を提供することもでき、例えば、ユーザに提供されるフィードバックは、いかなる形態の感覚フィードバック(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)でもよく、ユーザからの入力はいかなる形態でも(音響、音声又は触覚入力を含む)受信することができる。

【0158】

実施形態は、バックエンドコンポーネント(例えば、データサーバとして)を含むか、ミドルウェアコンポーネント(例えば、アプリケーションサーバ)を含むか、フロントエンドコンポーネント(例えば、ユーザが実施形態の実装形態と対話できるようにするためのグラフィカルユーザインタフェース若しくはウェブブラウザを有するクライアントコンピュータ)を含むか、そのようなバックエンド、ミドルウェア又はフロントエンドコンポーネントの任意の組合せを含む、コンピューティングシステムにおいて実装することができる。システムのコンポーネントは、デジタルデータ通信の任意の形態又は媒体(例えば、通信ネットワーク)によって相互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)及び広域ネットワーク(WAN)(例えば、インターネット)を含む。

【0159】

システム及び方法又はそれらの一部は、「ワールドワイドウェブ」（ウェブ又はWWW）を使用することができ、「ワールドワイドウェブ」は、ハイパーテキスト転送プロトコル（HTTP）を利用するインターネット上のサーバの集合体である。HTTPは、テキスト、グラフィックス、画像、音声、映像、ハイパーテキストマークアップ言語（HTML）などの異なるフォーマットの情報であり得る資源やプログラムへのアクセスをユーザに提供する公知のアプリケーションプロトコルである。ユーザによってリンクが指定され次第、クライアントコンピュータは、ウェブサーバへのTCP/IP要求を行い、情報（HTMLに従ってフォーマットされた別のウェブページであり得る）を受信する。また、ユーザは、画面上の命令に従ってあるデータを入力するか又は選択したアイコンをクリックすることによって、同じ又は他のサーバ上の他のページにもアクセスすることができる。また、ユーザが所定のコンポーネントに対するオプションを選択できるようにするため、ウェブページを使用する実施形態に対して、当業者に知られているいかなるタイプの選択デバイス（チェックボックス、ドロップダウンボックス及び同様のものなど）も使用できることにも留意されたい。サーバは、UNIX（登録商標）マシンを含む様々なプラットフォーム上で起動するが、Windows（登録商標）2000/2003、Windows（登録商標）NT、Sun、Linux（登録商標）、及び、Macintoshなどの他のプラットフォームを使用することもできる。コンピュータユーザは、Firefox、Netscape Navigator、Microsoft Internet Explorer又はMosaicブラウザなどの閲覧ソフトの使用を通じて、ウェブ上のサーバ又はネットワーク上で利用可能な情報を閲覧することができる。コンピューティングシステムは、クライアント及びサーバを含み得る。クライアント及びサーバは、一般に、互いにリモート設置され、典型的には、通信ネットワークを通じて対話する。クライアントとサーバとの関係は、コンピュータプログラムによって生じ、コンピュータプログラムは、それぞれのコンピュータ上で起動しており、互いにクライアント・サーバ関係を有するものである。

【0160】

他の実施形態は、説明及び請求項の範囲及び精神内である。例えば、ソフトウェアの本質により、上記で説明される機能は、ソフトウェア、ハードウェア、ファームウェア、配線又はこれらの任意の組合せを使用して実装することができる。機能を実装する特徴もまた、様々な位置に物理的に位置し得、機能の一部が異なる物理的場所を実装されるように分散されることを含む。本明細書における及び本出願全体を通じる「1つの（「a」）」という用語の使用は、制限する形では使用されず、従って、「1つの（「a」）」という用語に対して複数の意味或いは「1つ又は複数の」という意味を除外することを意図しない。それに加えて、仮特許出願に対して優先権が主張される範囲で、仮特許出願は制限するものではなく、本明細書で説明される技法をどのように実装できるかについての例を含むものであることを理解されたい。

【0161】

本発明の多くの実施形態について説明してきた。それにもかかわらず、当業者であれば、特許請求の範囲及び本明細書で説明される技法の精神及び範囲から逸脱することなく、様々な変更を行えることが理解されよう。

10

20

30

40

50

【図面】

【図 1 A】

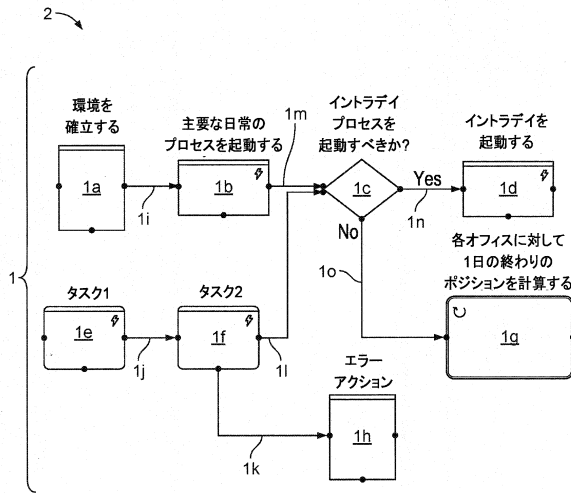


図 1A

【図 1 B】

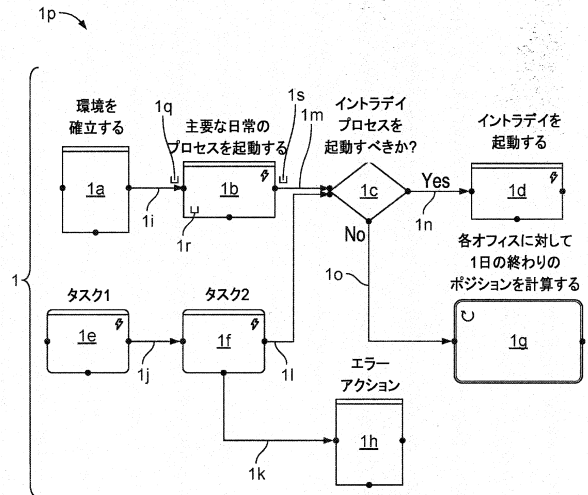


図 1B

【図 1 C】

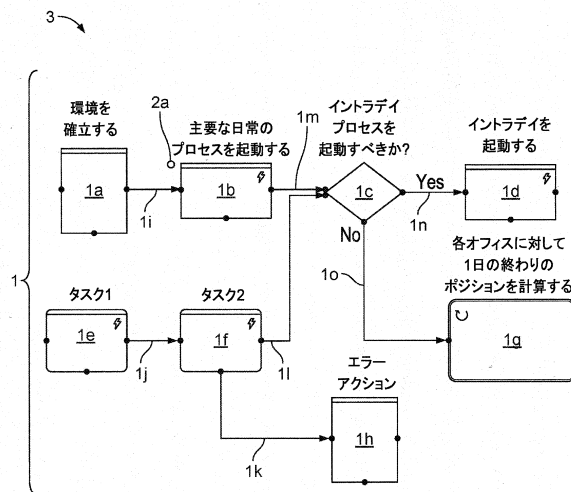


図 1C

【図 1 D】

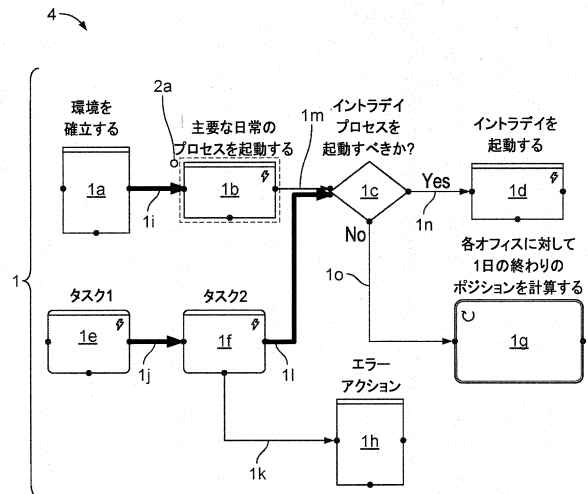


図 1D

10

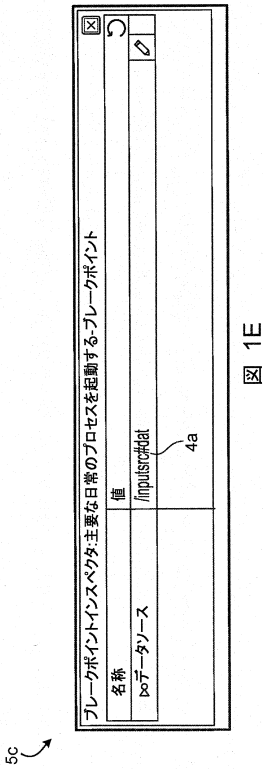
20

30

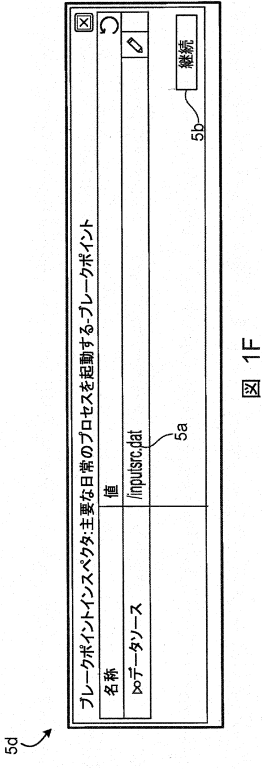
40

50

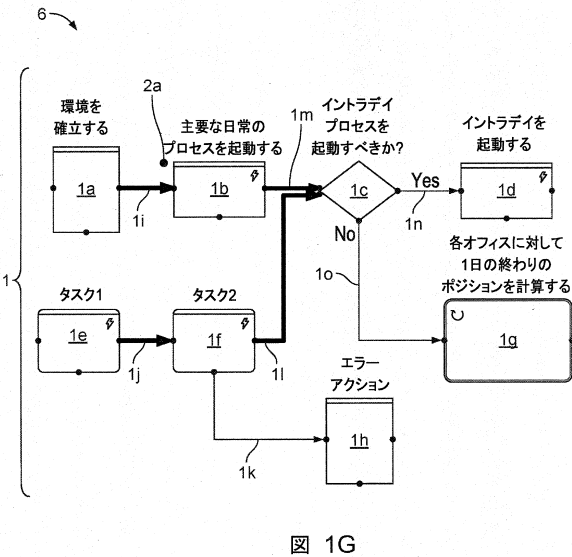
【図 1 E】



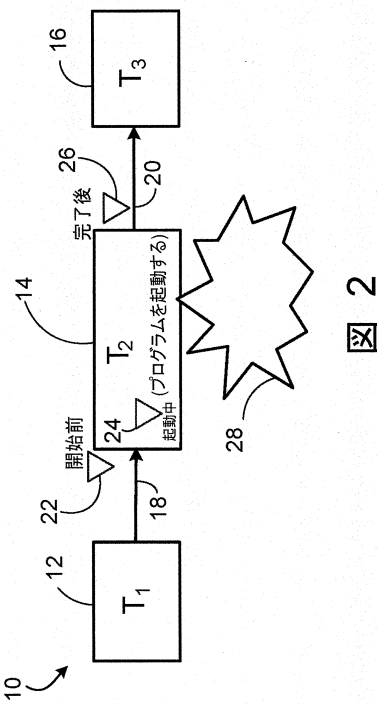
【図 1 F】



【図 1 G】



【図 2】



10

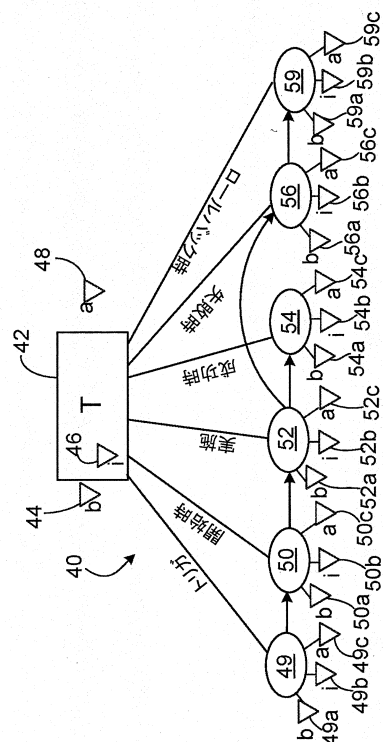
20

30

40

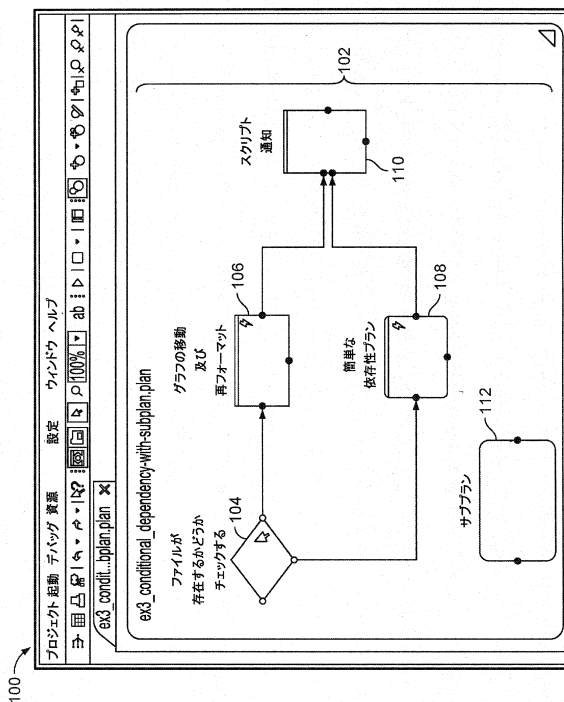
50

【図 3】



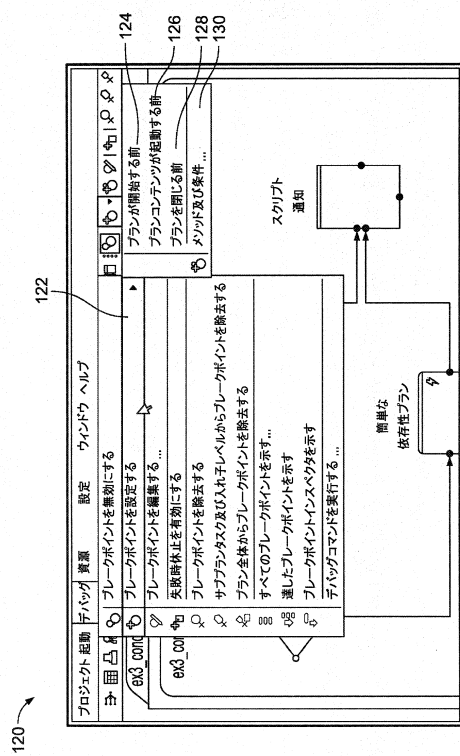
3
✕

【図 4】



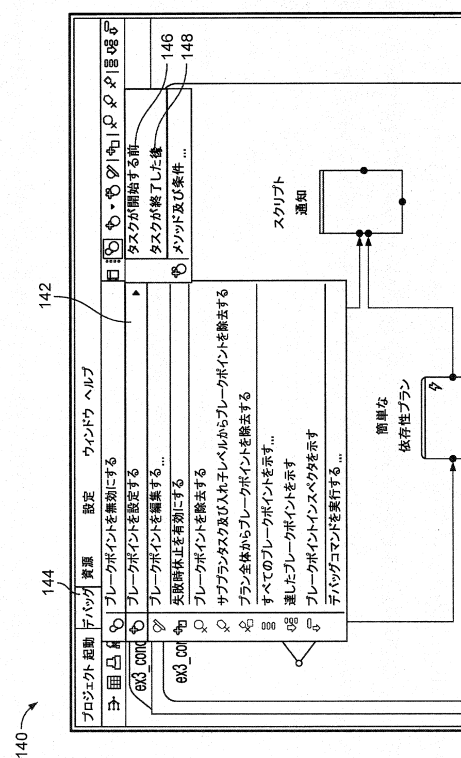
4
✕

【図 5】



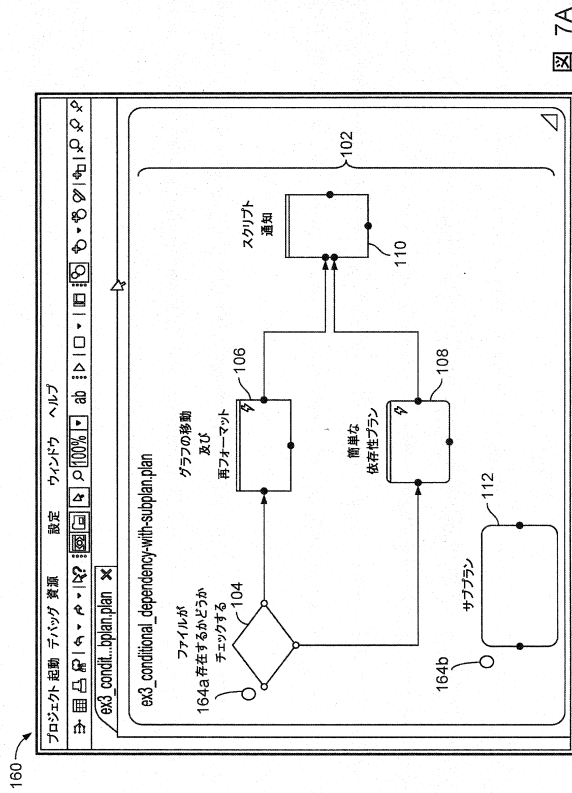
5 ☒

【図 6】



6
☒

【図 7 A】



【図 7 B】

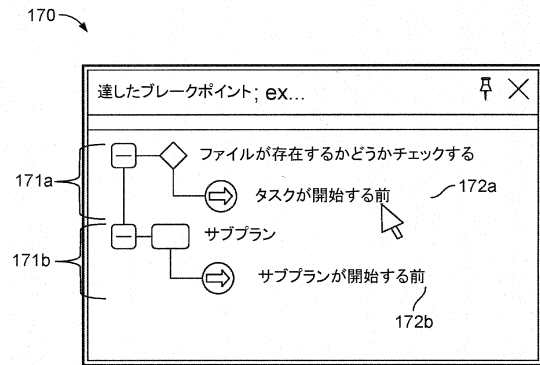


図 7B

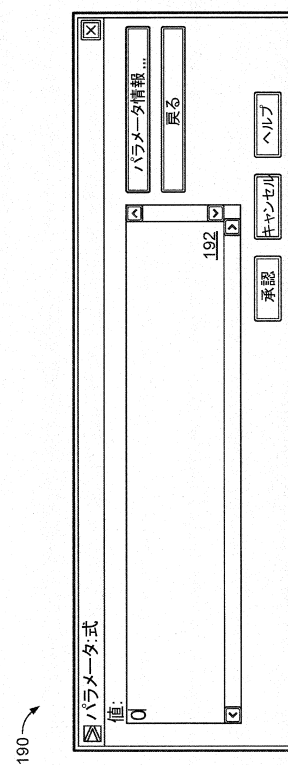
10

20

【図 8】

[illegible]

【図 9 A】



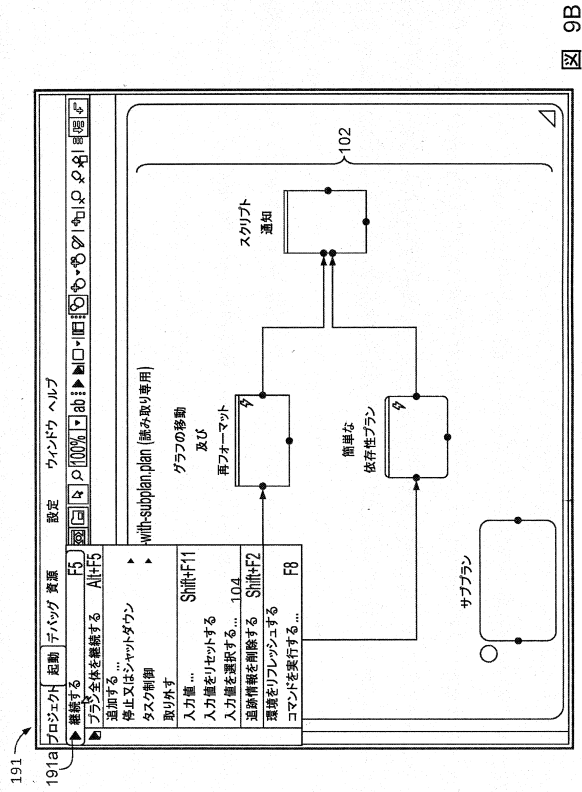
9A

30

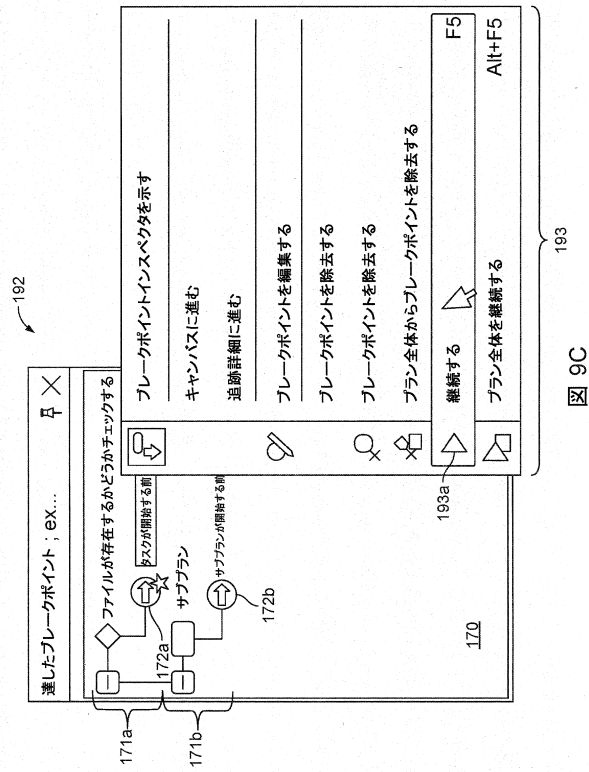
40

50

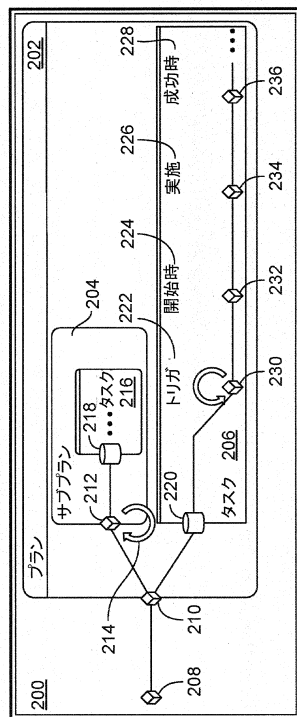
【図 9 B】



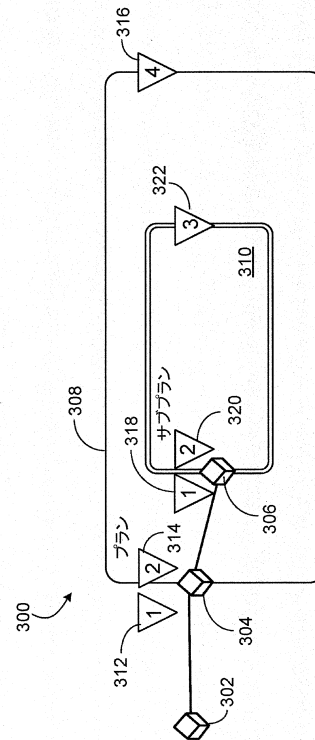
【図 9 C】



【図 10】



【図 11】



【図 1 2】

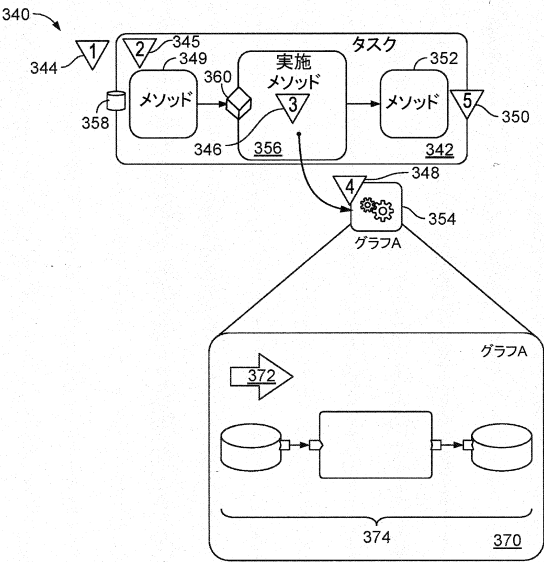


図 12

【図 1 3】

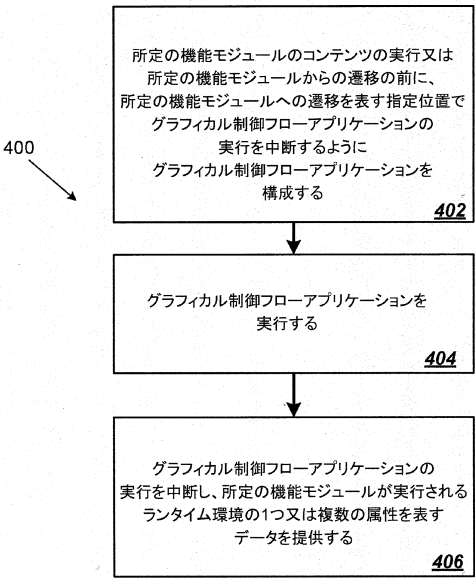


図 13

【図 1 4 A】

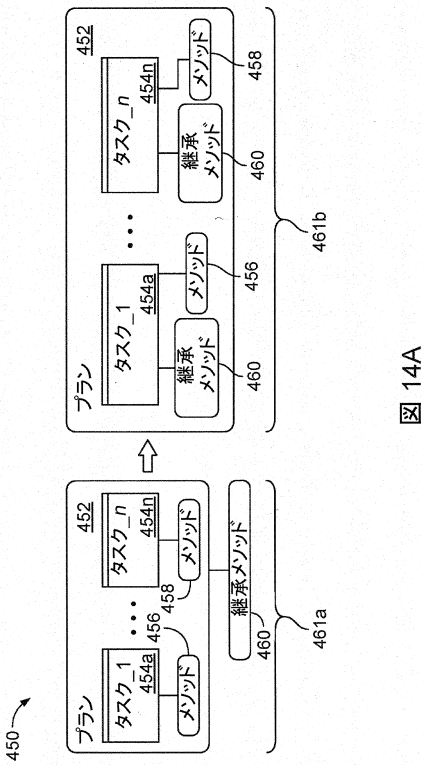


図 14A

【図 1 4 B】

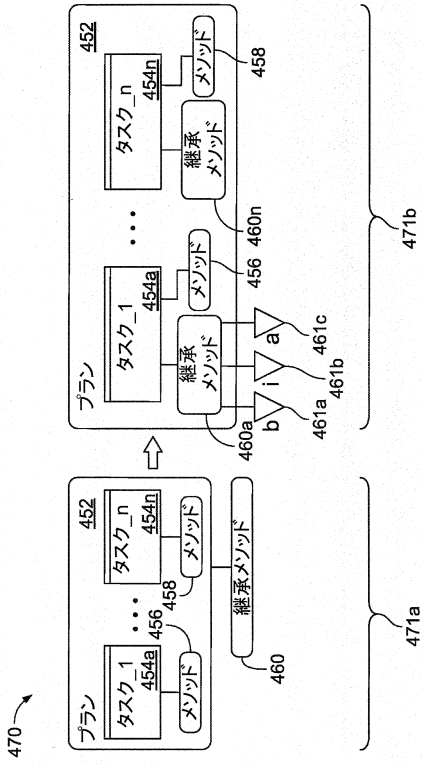


図 14B

10

20

30

40

50

【図 15】

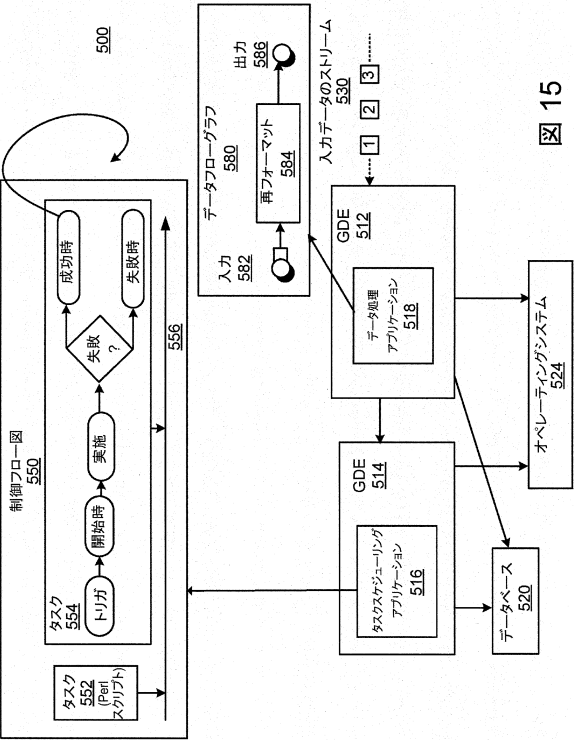


図 15

10

20

30

40

50

フロントページの続き

- (72)発明者 マーク・スタックニス
 アメリカ合衆国・マサチューセッツ・０２４２１・レキシントン・スプリング・ストリート・２０
 １
- (72)発明者 シン・リ
 アメリカ合衆国・マサチューセッツ・０２４２１・レキシントン・スプリング・ストリート・２０
 １
- 審査官 木村 雅也
- (56)参考文献 特開２００２－２８８００１（ＪＰ，Ａ）
 米国特許出願公開第２０１１／００４７４１５（ＵＳ，Ａ１）
 米国特許出願公開第２００２／０１２０９１９（ＵＳ，Ａ１）
 米国特許出願公開第２００７／０２４０１２５（ＵＳ，Ａ１）
- (58)調査した分野 (Int.Cl., D B名)
 Ｇ０６Ｆ １１／３６