

(19)日本国特許庁(JP)

(12)特許公報(B2)

(11)特許番号
特許第7573359号
(P7573359)

(45)発行日 令和6年10月25日(2024.10.25)

(24)登録日 令和6年10月17日(2024.10.17)

(51)国際特許分類

G 0 6 F 8/30 (2018.01)

F I G 0 6 F 8/30

請求項の数 51 (全30頁)

(21)出願番号	特願2018-563483(P2018-563483)	(73)特許権者	509123208
(86)(22)出願日	平成29年5月18日(2017.5.18)		アビニシオ テクノロジー エルエルシー
(65)公表番号	特表2019-521430(P2019-521430 A)		アメリカ合衆国 0 2 4 2 1 マサチュー
(43)公表日	令和1年7月25日(2019.7.25)		セッツ州 レキシントン スプリング ス
(86)国際出願番号	PCT/US2017/033285	(74)代理人	トリート 2 0 1
(87)国際公開番号	WO2017/209969		100079108
(87)国際公開日	平成29年12月7日(2017.12.7)		弁理士 稲葉 良幸
審査請求日	令和2年5月15日(2020.5.15)	(74)代理人	100109346
審判番号	不服2023-1955(P2023-1955/J1)		弁理士 大貫 敏史
審判請求日	令和5年2月6日(2023.2.6)	(74)代理人	100117189
(31)優先権主張番号	62/345,217		弁理士 江口 昭彦
(32)優先日	平成28年6月3日(2016.6.3)	(74)代理人	100134120
(33)優先権主張国・地域又は機関	米国(US)		弁理士 内藤 和彦
	最終頁に続く	(72)発明者	イスマン, マーシャル エー.
			アメリカ合衆国, マサチューセッツ州
			最終頁に続く

(54)【発明の名称】 フォーマット特有のデータ処理動作

(57)【特許請求の範囲】

【請求項1】

コンピュータ・プログラムの第1のバージョンをプロセッサによって解析することであって、前記コンピュータ・プログラムの前記第1のバージョンに含まれる第1のプロセスを識別することを含み、前記第1のプロセスが、第1のフォーマットのデータに第1の動作を実行するように構成されることと、

前記第1のフォーマットとは異なる第2のフォーマットのデータに第2の動作を実行するように構成された1つ又は複数の第2のプロセスを定義する指定を、プロセッサによって、前記解析に基づいて、作成することであって、前記第2の動作は、前記第1の動作に基づいており、前記指定は、前記コンピュータ・プログラムとは別である、ことと、

前記コンピュータ・プログラムの前記第1のバージョン、及び前記指定に基づいて、前記コンピュータ・プログラムの少なくとも一部分の第2のバージョンを生成するためにコンパイラ又はインタプリタを実行することであって、前記第1のプロセスを省略することを含み、前記指定により定義された、及び前記第1のフォーマットとは異なる前記第2のフォーマットのデータに前記第2の動作を実行するように構成された前記1つ又は複数の第2のプロセスを、コンピュータ・プログラムの前記少なくとも前記一部分の前記第2のバージョンに含むこととを含む、方法。

【請求項2】

前記第1のプロセスを識別することは、前記第1の動作が前記データのフォーマットに依存しているプロセスを識別することを含む、請求項1に記載の方法。

【請求項 3】

前記第 1 のプロセスを識別することが、前記第 2 のフォーマットのデータに前記第 1 の動作を実行することができないプロセスを識別することを含む、請求項 1 又は 2 に記載の方法。

【請求項 4】

前記第 1 のプロセスによって処理されるべきデータの前記第 1 のフォーマットを決定することを含む、請求項 1 ～ 3 のいずれか一項に記載の方法。

【請求項 5】

前記第 1 のプロセスを識別することが、前記コンピュータ・プログラムの第 1 のデータ処理要素を識別することを含み、前記第 1 のデータ処理要素が、前記第 1 のプロセスを実行するように構成される、請求項 1 ～ 4 のいずれか一項に記載の方法。

10

【請求項 6】

前記コンピュータ・プログラムでの前記 1 つ又は複数の第 2 のプロセスを含むことが、前記コンピュータ・プログラムの前記少なくとも前記一部分の前記第 2 のバージョンでの 1 つ又は複数の第 2 のデータ処理要素を含み、前記第 2 のデータ処理要素が、前記 1 つ又は複数の第 2 のプロセスを実行するように構成される、請求項 5 に記載の方法。

【請求項 7】

前記第 1 のフォーマットが、データ・タイプを含む、請求項 1 ～ 6 のいずれか一項に記載の方法。

【請求項 8】

前記第 1 のフォーマットが、データ要素のサイズを含む、請求項 1 ～ 7 のいずれか一項に記載の方法。

20

【請求項 9】

前記第 1 のプロセスが、第 1 のレコード・フォーマットのデータ・レコードに前記第 1 の動作を実行するように構成され、前記 1 つ又は複数の第 2 のプロセスが、第 2 のレコード・フォーマットのデータ・レコードに前記第 2 の動作を実行するように構成される、請求項 1 ～ 8 のいずれか一項に記載の方法。

【請求項 10】

前記第 1 のレコード・フォーマットが、前記第 1 のレコード・フォーマットのデータ・レコードでのフィールドの名前を含む、請求項 9 に記載の方法。

30

【請求項 11】

ユーザ・インターフェースにおいて、前記第 1 の動作の識別子を提示することを含む、請求項 1 ～ 10 のいずれか一項に記載の方法。

【請求項 12】

前記コンピュータ・プログラムの少なくとも前記一部分の前記第 2 のバージョンを生成することが、前記コンピュータ・プログラムの前記少なくとも前記一部分のコピーを生成することを含む、請求項 1 ～ 11 のいずれか一項に記載の方法。

【請求項 13】

前記コンピュータ・プログラムの前記少なくとも前記一部分の前記コピーを修正して、前記第 1 のプロセスを省略し、前記 1 つ又は複数の第 2 のプロセスを含むことを含む、請求項 12 に記載の方法。

40

【請求項 14】

前記コンピュータ・プログラムの前記第 2 のバージョンを実行することを含む、請求項 1 ～ 13 のいずれか一項に記載の方法。

【請求項 15】

前記指定がオーバーレイ指定を含む、請求項 1 ～ 14 のいずれか一項に記載の方法。

【請求項 16】

前記コンピュータ・プログラムの前記第 2 のバージョンを生成することが、前記コンピュータ・プログラムの前記第 1 のバージョン及び前記オーバーレイ指定に基づいて前記第 2 のバージョンを生成することを含む、請求項 15 に記載の方法。

50

【請求項 17】

前記オーバーレイ指定が、前記第1のプロセスの上流のプロセス及び前記第1のプロセスの下流のプロセスのうち1つ又は複数を識別する、請求項15又は16に記載の方法。

【請求項 18】

前記第1のプロセスを定義するコードの解析に基づいて、前記第1のプロセスを識別することを含む、請求項15～17のいずれか一項に記載の方法。

【請求項 19】

前記コンピュータ・プログラムがデータフローグラフを含む、請求項1～18のいずれか一項に記載の方法。

【請求項 20】

前記第1のプロセスが、前記データフローグラフの第1の構成要素によって表される実行可能なプロセスであり、前記1つ又は複数の第2のプロセスが、前記データフローグラフの1つ又は複数の第2の構成要素によって表される実行可能なプロセスである、請求項19に記載の方法。

【請求項 21】

前記1つ又は複数の第2の構成要素が、前記データフローグラフの上流構成要素からデータ・レコードを受信するように構成される、請求項20に記載の方法。

【請求項 22】

前記1つ又は複数の第2の構成要素が、前記データフローグラフの下流構成要素にデータ・レコードを提供するように構成される、請求項20又は21に記載の方法。

【請求項 23】

コンピュータ・プログラムの第1のバージョンをプロセッサによって解析するための手段であって、前記解析することが、前記コンピュータ・プログラムの前記第1のバージョンに含まれる第1のプロセスを識別することを含み、前記第1のプロセスが、第1のフォーマットを有するデータに第1の動作を実行するように構成される手段と、

前記第1のフォーマットとは異なる第2のフォーマットのデータに第2の動作を実行するように構成された1つ又は複数の第2のプロセスを定義する指定を、プロセッサによって、前記解析に基づいて、作成するための手段であって、前記第2の動作は、前記第1の動作に基づいており、前記指定は、前記コンピュータ・プログラムとは別である手段と、

前記コンピュータ・プログラムの前記第1のバージョン、及び前記指定に基づいて、前記コンピュータ・プログラムの少なくとも一部分の第2のバージョンを生成するためにコンパイラ又はインタプリタをプロセッサによって実行するための手段であって、前記第1のプロセスを省略することを含み、前記指定により定義された、及び前記第1のフォーマットとは異なる前記第2のフォーマットを有するデータに前記第2の動作を実行するように構成された前記1つ又は複数の第2のプロセスを、前記コンピュータ・プログラムの前記少なくとも前記一部分の前記第2のバージョンに含む手段とを備える、システム。

【請求項 24】

メモリに結合されたプロセッサを備え、前記プロセッサ及びメモリが、

コンピュータ・プログラムの第1のバージョンを解析するように構成され、前記解析することが、前記コンピュータ・プログラムの前記第1のバージョンに含まれる第1のプロセスを識別することを含み、前記第1のプロセスが、第1のフォーマットを有するデータに第1の動作を実行するように構成され、

前記第1のフォーマットとは異なる第2のフォーマットのデータに第2の動作を実行するように構成された1つ又は複数の第2のプロセスを定義する指定を、前記解析に基づいて、作成するように構成され、前記第2の動作は、前記第1の動作に基づいており、前記指定は、前記コンピュータ・プログラムとは別であり、

前記コンピュータ・プログラムの前記第1のバージョン、及び前記指定に基づいて、前記コンピュータ・プログラムの少なくとも一部分の第2のバージョンを生成するためにコンパイラ又はインタプリタを実行するように構成され、前記第1のプロセスを省略することを含み、前記指定により定義された、及び前記第1のフォーマットとは異なる前記第2

10

20

30

40

50

のフォーマットを有するデータに前記第2の動作を実行するように構成された前記1つ又は複数の第2のプロセスを、前記コンピュータ・プログラムの前記少なくとも前記一部分の前記第2のバージョンに含む、システム。

【請求項25】

コンピューティング・システムが、

コンピュータ・プログラムの第1のバージョンを解析するようにし、前記解析することが、前記コンピュータ・プログラムの前記第1のバージョンに含まれる第1のプロセスを識別することを含み、前記第1のプロセスが、第1のフォーマットを有するデータに第1の動作を実行するように構成され、

前記第1のフォーマットとは異なる第2のフォーマットのデータに第2の動作を実行するように構成された1つ又は複数の第2のプロセスを定義する指定を、前記解析に基づいて、作成するようにし、前記第2の動作は、前記第1の動作に基づいており、前記指定は、前記コンピュータ・プログラムとは別であり、

前記コンピュータ・プログラムの前記第1のバージョン、及び前記指定に基づいて、前記コンピュータ・プログラムの少なくとも一部分の第2のバージョンを生成するためにコンパイラ又はインタプリタを実行するようにし、前記第1のプロセスを省略することを含み、前記指定により定義された、及び前記第1のフォーマットとは異なる前記第2のフォーマットを有するデータに前記第2の動作を実行するように構成された前記1つ又は複数の第2のプロセスを、前記コンピュータ・プログラムの前記少なくとも前記一部分の前記第2のバージョンに含むようにするためのプログラム。

【請求項26】

前記第1のプロセスを識別することは、前記第1の動作が前記データのフォーマットに依存しているプロセスを識別することを含む、請求項25に記載のプログラム。

【請求項27】

前記第1のプロセスを識別することが、前記第2のフォーマットのデータに前記第1の動作を実行することができないプロセスを識別することを含む、請求項25に記載のプログラム。

【請求項28】

前記コンピューティング・システムに、前記第1のプロセスによって処理されるべきデータの前記第1のフォーマットを決定させる、請求項25に記載のプログラム。

【請求項29】

前記第1のプロセスを識別することが、前記コンピュータ・プログラムの第1のデータ処理要素を識別することを含み、前記第1のデータ処理要素が、前記第1のプロセスを実行するように構成される、請求項25に記載のプログラム。

【請求項30】

前記コンピュータ・プログラムでの前記1つ又は複数の第2のプロセスを含むことが、前記コンピュータ・プログラムの前記少なくとも前記一部分の前記第2のバージョンでの1つ又は複数の第2のデータ処理要素を含み、前記第2のデータ処理要素が、前記1つ又は複数の第2のプロセスを実行するように構成される、請求項29に記載のプログラム。

【請求項31】

前記第1のプロセスが、第1のレコード・フォーマットのデータ・レコードに前記第1の動作を実行するように構成され、前記1つ又は複数の第2のプロセスが、第2のレコード・フォーマットのデータ・レコードに前記第2の動作を実行するように構成される、請求項25に記載のプログラム。

【請求項32】

前記コンピュータ・プログラムの少なくとも一部分の前記第2のバージョンを生成することが、前記コンピュータ・プログラムの前記一部分のコピーを生成することを含む、請求項25に記載のプログラム。

【請求項33】

前記コンピューティング・システムに、前記コンピュータ・プログラムの前記一部分の前

10

20

30

40

50

記コピーを修正させ、前記第 1 のプロセスを省略させ、前記 1 つ又は複数の第 2 のプロセスを含ませる、請求項 3 2 に記載のプログラム。

【請求項 3 4】

前記コンピューティング・システムに、前記コンピュータ・プログラムの前記第 2 のバージョンを実行させる、請求項 2 5 に記載のプログラム。

【請求項 3 5】

前記指定がオーバーレイ指定を含み、前記コンピュータ・プログラムの前記第 2 のバージョンを生成することが、前記コンピュータ・プログラムの前記第 1 のバージョン及び前記オーバーレイ指定に基づいて前記第 2 のバージョンを生成することを含む、請求項 2 5 に記載のプログラム。

10

【請求項 3 6】

前記コンピューティング・システムに、前記第 1 のプロセスを定義するコードの解析に基づいて、前記第 1 のプロセスを識別させる、請求項 2 5 に記載のプログラム。

【請求項 3 7】

前記コンピュータ・プログラムがデータフローグラフを含む、請求項 2 5 に記載のプログラム。

【請求項 3 8】

前記第 1 のプロセスが、前記データフローグラフの第 1 の構成要素によって表される実行可能なプロセスであり、前記 1 つ又は複数の第 2 のプロセスが、前記データフローグラフの 1 つ又は複数の第 2 の構成要素によって表される実行可能なプロセスである、請求項 3 7 に記載のプログラム。

20

【請求項 3 9】

第 1 のプロセスを識別することは、前記第 1 の動作が前記データのフォーマットに依存しているプロセスを識別することを含む、請求項 2 4 に記載のシステム。

【請求項 4 0】

第 1 のプロセスを識別することが、前記第 2 のフォーマットのデータに前記第 1 の動作を実行することができないプロセスを識別することを含む、請求項 2 4 に記載のシステム。

【請求項 4 1】

前記プロセッサ及びメモリは、前記第 1 のプロセスによって処理されるべきデータの前記第 1 のフォーマットを決定するように構成されている、請求項 2 4 に記載のシステム。

30

【請求項 4 2】

前記第 1 のプロセスを識別することが、前記コンピュータ・プログラムの第 1 のデータ処理要素を識別することを含み、前記第 1 のデータ処理要素が、前記第 1 のプロセスを実行するように構成される、請求項 2 4 に記載のシステム。

【請求項 4 3】

前記コンピュータ・プログラムでの前記 1 つ又は複数の第 2 のプロセスを含むことが、前記コンピュータ・プログラムの前記少なくとも一部分の前記第 2 のバージョンでの 1 つ又は複数の第 2 のデータ処理要素を含み、前記第 2 のデータ処理要素が、前記 1 つ又は複数の第 2 のプロセスを実行するように構成される、請求項 4 2 に記載のシステム。

【請求項 4 4】

前記第 1 のプロセスが、第 1 のレコード・フォーマットのデータ・レコードに前記第 1 の動作を実行するように構成され、前記 1 つ又は複数の第 2 のプロセスが、第 2 のレコード・フォーマットのデータ・レコードに前記第 2 の動作を実行するように構成される、請求項 2 4 に記載のシステム。

40

【請求項 4 5】

前記コンピュータ・プログラムの少なくとも一部分の前記第 2 のバージョンを生成することが、前記コンピュータ・プログラムの前記一部分のコピーを生成することを含む、請求項 2 4 に記載のシステム。

【請求項 4 6】

前記プロセッサ及びメモリは、前記コンピュータ・プログラムの前記一部分の前記コピ

50

一を修正して、前記第1のプロセスを省略し、前記1つ又は複数の第2のプロセスを含むように構成されている、請求項45に記載のシステム。

【請求項47】

前記プロセッサ及びメモリは、前記コンピュータ・プログラムの前記第2のバージョンを実行するように構成されている、請求項24に記載のシステム。

【請求項48】

前記指定がオーバレイ指定を含み、前記コンピュータ・プログラムの前記第2のバージョンを生成することが、前記コンピュータ・プログラムの前記第1のバージョン及び前記オーバレイ指定に基づいて前記第2のバージョンを生成することを含む、請求項24に記載のシステム。

【請求項49】

前記プロセッサ及びメモリは、前記第1のプロセスを定義するコードの解析に基づいて、前記第1のプロセスを識別するように構成されている、請求項24に記載のシステム。

【請求項50】

前記コンピュータ・プログラムがデータフローグラフを含む、請求項24に記載のシステム。

【請求項51】

前記第1のプロセスが、前記データフローグラフの第1の構成要素によって表される実行可能なプロセスであり、前記1つ又は複数の第2のプロセスが、前記データフローグラフの1つ又は複数の第2の構成要素によって表される実行可能なプロセスである、請求項50に記載のシステム。

【発明の詳細な説明】

【技術分野】

【0001】

優先権の主張

本出願は、2016年6月3日出願の米国特許仮出願第62／345,217号、及び2017年2月15日出願の米国特許出願第15／433,467号の優先権を主張し、その両方の内容全体を参考としてここに組み込む。

【背景技術】

【0002】

背景

複雑な計算は、有向グラフ（「データフロー・グラフ」と呼ばれる）を用いたデータ・フローとして表すことのできる場合が多く、この計算の構成要素は、このグラフの頂点に関連付けられ、データは、グラフのリンク（弧、縁部）に対応する各構成要素間を流れる。各構成要素は、1つ又は複数の入力ポートでデータを受信し、このデータを処理し、1つ又は複数の出力ポートからデータを提供するデータ処理構成要素、及びデータ・フローの送信側又は受信側の役割を果たすデータセット構成要素を含むことができる。

【発明の概要】

【課題を解決するための手段】

【0003】

概要

一態様では、方法は、コンピュータ・プログラムの第1のバージョンをプロセッサによって解析することであって、コンピュータ・プログラムの第1のバージョンに含まれる第1のプロセスを識別することを含み、この第1のプロセスが、第1のフォーマットのデータに第1の動作を実行するように構成されることと、コンピュータ・プログラムの少なくとも一部分の第2のバージョンをプロセッサによって生成することであって、第1のプロセスを省略することを含み、第1のフォーマットとは異なる第2のフォーマットのデータに第2の動作を実行するように構成された1つ又は複数の第2のプロセスを、コンピュータ・プログラムの少なくとも一部分の第2のバージョンに含むことを含み、この第2の動作が第1の動作に基づく。

10

20

30

40

50

【0004】

各実施形態は、以下の特徴のうちの1つ又は複数を含むことができる。

【0005】

第1のプロセスを識別することは、第1の動作がデータのフォーマットに依存している第1のプロセスを識別することを含む。

【0006】

第1のプロセスを識別することは、第2のフォーマットのデータに第1の動作を実行することができない第1のプロセスを識別することを含む。

【0007】

この方法は、第1のプロセスによって処理されるデータのフォーマットを決定することを含む。第1のプロセスを識別することは、この第1のプロセスによって処理されるデータのフォーマットを有するデータに、第1の動作を実行することができない第1のプロセスを識別することを含む。

10

【0008】

第1のプロセスを識別することは、コンピュータ・プログラムの第1のデータ処理要素を識別することを含み、この第1のデータ処理要素は、第1のプロセスを実行するように構成される。コンピュータ・プログラムの少なくとも一部分の第2のバージョンでの1つ又は複数の第2のプロセスを含むことは、コンピュータ・プログラムの少なくとも一部分の第2のバージョンでの1つ又は複数の第2のデータ処理要素を含むことを含み、この第2のデータ処理要素は、1つ又は複数の第2のプロセスを実行するように構成される。

20

【0009】

第1のフォーマットは、データ・タイプを含む。

【0010】

第1のフォーマットは、データ要素のサイズを含む。

【0011】

第1のプロセスは、第1のレコード・フォーマットのデータ・レコードに第1の動作を実行するように構成され、1つ又は複数の第2のプロセスは、第2のレコード・フォーマットのデータ・レコードに第2の動作を実行するように構成される。第1のレコード・フォーマットは、各レコードでのフィールドの名前を含む。

【0012】

この方法は、ユーザ・インターフェースにおいて、1つ又は複数の動作の第1のセットの識別子を提示することを含む。

30

【0013】

コンピュータ・プログラムの少なくとも一部分の第2のバージョンを生成することは、このコンピュータ・プログラムのこの部分のコピーを生成することを含む。

【0014】

この方法は、コンピュータ・プログラムのこの部分のコピーを修正して、第1のプロセスを省き、1つ又は複数の第2のプロセスを含むことを含む。

【0015】

この方法は、コンピュータ・プログラムの第2のバージョンを実行することを含む。

40

【0016】

1つ又は複数の第2のプロセスは、オーバーレイ指定によって定義される。コンピュータ・プログラムの第2のバージョンを生成することは、コンピュータ・プログラムの第1のバージョン及びオーバーレイ指定に基づいて第2のバージョンを生成することを含む。このオーバーレイ指定は、第1のプロセスの上流のプロセス及び第1のプロセスの下流のプロセスのうちの1つ又は複数を識別する。この方法は、第1のプロセスを定義する実行可能コードの解析に基づいて、第1のプロセスを識別することを含む。

【0017】

コンピュータ・プログラムはグラフを含む。第1のプロセスは、このグラフの第1の構成要素によって表される実行可能なプロセスであり、1つ又は複数の第2のプロセスは、

50

グラフの1つ又は複数の第2の構成要素によって表される実行可能なプロセスである。1つ又は複数の第2の構成要素は、グラフの上流構成要素からデータ・レコードを受信するように構成される。1つ又は複数の第2の構成要素は、グラフの下流構成要素にデータ・レコードを提供するように構成される。

【0018】

一態様では、システムは、コンピュータ・プログラムの第1のバージョンをプロセッサによって解析するための手段であって、この解析することが、コンピュータ・プログラムの第1のバージョンに含まれる第1のプロセスを識別することを含み、この第1のプロセスが、第1のフォーマットを有するデータに第1の動作を実行するように構成される手段と、コンピュータ・プログラムの少なくとも一部分の第2のバージョンをプロセッサによって生成するための手段であって、第1のプロセスを省略することを含み、第1のフォーマットとは異なる第2のフォーマットを有するデータに第2の動作を実行するように構成された1つ又は複数の第2のプロセスを、コンピュータ・プログラムの少なくとも一部分の第2のバージョンに含む手段とを備え、この第2の動作が第1の動作に基づく。

10

【0019】

一態様では、システムは、メモリに結合されたプロセッサを備え、このプロセッサ及びメモリは、コンピュータ・プログラムの第1のバージョンを解析するように構成され、この解析することが、コンピュータ・プログラムの第1のバージョンに含まれる第1のプロセスを識別することを含み、この第1のプロセスが、第1のフォーマットを有するデータに第1の動作を実行するように構成され、コンピュータ・プログラムの少なくとも一部分の第2のバージョンを生成するように構成され、第1のプロセスを省略することを含み、第1のフォーマットとは異なる第2のフォーマットを有するデータに第2の動作を実行するように構成された1つ又は複数の第2のプロセスを、コンピュータ・プログラムの少なくとも一部分の第2のバージョンに含み、この第2の動作が第1の動作に基づく。

20

【0020】

一態様では、持続的なコンピュータ読取り可能な媒体は、コンピューティング・システムが、コンピュータ・プログラムの第1のバージョンを解析するようにし、この解析することが、コンピュータ・プログラムの第1のバージョンに含まれる第1のプロセスを識別することを含み、この第1のプロセスが、第1のフォーマットを有するデータに第1の動作を実行するように構成され、コンピュータ・プログラムの少なくとも一部分の第2のバージョンを生成するようにし、第1のプロセスを省略することを含み、第1のフォーマットとは異なる第2のフォーマットを有するデータに第2の動作を実行するように構成された1つ又は複数の第2のプロセスを、コンピュータ・プログラムの少なくとも一部分の第2のバージョンに含むようにするための命令を記憶し、この第2の動作が第1の動作に基づく。

30

【0021】

他の特徴及び利点は、以下の説明及び特許請求の範囲から明白になろう。

【図面の簡単な説明】

【0022】

図面の説明

40

【図1】グラフの例である。

【図2】グラフの例である。

【図3】グラフの例である。

【図4】オーバーレイ指定の一例である。

【図5】ブロック図である。

【図6】流れ図である。

【図7】ブロック図である。

【図8】ブロック図である。

【図9】ブロック図である。

【図10】流れ図である。

50

【発明を実施するための形態】

【0023】

説明

グラフなどの実行可能なアプリケーションは、この実行可能なアプリケーションによって処理されるデータ・レコードの1つ又は複数の特定のフォーマットに特有の、1つ又は複数のプロセスを含むことができる。このようなプロセスは、1つ又は複数の特定のフォーマットのみデータに動作を実行することができ、互いに異なるフォーマットのデータにこれらのプロセスを実行しようとする、エラー又は不正確な処理につながる場合がある。ここで、データの1つ又は複数の特定のフォーマットに特有の、実行可能なアプリケーションのプロセスを識別するための手法について述べる。実行可能なアプリケーションが、互いに異なるフォーマットのデータに動作を実行できるようにするために、これらのフォーマット特有のプロセスが省略され、1つ又は複数の他のプロセスが含まれる実行可能なアプリケーションの第2のバージョンを生成することができる。置換プロセスと呼ばれることもある、これら他のプロセスは、省略されたフォーマット特有のプロセスによって実行される動作に基づいているが、互いに異なる1つ又は複数のフォーマットのデータに特有であり、又は任意のフォーマットのデータに動作を実行できる動作を実行することができる。実行可能なアプリケーションの第2のバージョンに置換プロセスを含めることにより、実行可能なアプリケーションは、この実行可能なアプリケーションの元のバージョンが構成されたフォーマットとは異なるフォーマットのデータを処理できるようになる。

【0024】

置換プロセスは、オーバーレイ指定で定義することができ、このオーバーレイ指定は、元の実行可能なアプリケーション（第1のアプリケーションとも呼ばれることがある）とは別のファイルである。オーバーレイ指定で定義された置換プロセスは、元のアプリケーションの一部になることなく、アプリケーションの第2のバージョン（第2のアプリケーションとも呼ばれることもある）に追加することができる。たとえば、アプリケーションがコンパイルされると、コンパイラは、オーバーレイ・ファイルを考慮に入れ、1つ又は複数のプロセスが省略され、1つ又は複数の対応する置換プロセスが含まれる第2のアプリケーションを生成する。省略されると、第1のアプリケーションに含まれるプロセスが、第2のアプリケーションには含まれないことを意味する。対応する置換構成要素は、第2のアプリケーション、たとえば省略されたプロセスが第1のアプリケーションに配置されていた位置に挿入される。

【0025】

置換プロセスは、挿入の例である。挿入の他の例には、テスト・ソース及びプローブが含まれ、これらは又オーバーレイ指定によって定義することができる。テスト・ソースは、実行可能なアプリケーションによる処理向けの、テスト・データなどのデータを提供できる置換データ・ソースである。プローブは、実行可能なアプリケーションがデータを書き込む際の、代替の書き込み先である。挿入は、たとえば、グラフなどの実行可能なアプリケーションをテスト又はデバッグするのに有用になる場合がある。たとえば、試験者又は開発者は、アプリケーションに加える変更の効果を観察するために、入力データの特定のセットを使用してテストを実行したいと思うことがある。この変更の前後ともに、入力データの一貫したセットを使用してアプリケーションを実行することにより、アプリケーションが出力するデータに加えるこの変更の効果を監視することができる。例によっては、アプリケーションの機能の全てを少なくとも一度は実行することになる、テスト・データのセットなど、試験者は、アプリケーションをテストするときに使用されることになるテスト・データの特定のセットを有する場合がある。同様に、試験者は、アプリケーションが出力データを書き込む際の標準の書き込み先とは異なる特定の書き込み先に、アプリケーションが出力するこのデータを書き込みたいと考えることがある。

【0026】

例によっては、挿入は、アプリケーションの自動解析に基づいて自動的に定義することができる。たとえば、アプリケーションでのフォーマット特有のプロセスの自動識別に基

10

20

30

40

50

づいて、置換構成要素を自動的に定義することができる。テスト・ソース及びプローブは、アプリケーションのデータ・ソース及び出力データ受信側の自動識別に基づいて自動的に定義することができる。

【0027】

例によっては、実行可能なアプリケーションはグラフベースのプロセスである。グラフベースのプロセスは、ある1つの構成要素から別の構成要素へのデータの流れを示すフローによって接続された、それぞれ実行可能なプロセスを表す1つ又は複数の構成要素を含む。置換プロセスは、グラフベースのプロセスでの構成要素に関連付けられたオブジェクトである。置換プロセス（置換え構成要素とも呼ばれることがある）は、既存の構成要素によって処理されたはずのデータが置換構成要素によって代わりに処理されるように、グラフ内の既存の構成要素を置換することができる。テスト・ソース及びプローブの挿入は、グラフベースのプロセスでのフローに関連付けられたオブジェクトである。テスト・ソースは、フローを通過するデータ（たとえば、アップストリーム・データ）を新規データで置換することができ、その結果、グラフのそれぞれの実行において上流側の計算を再実行する必要はない。たとえば、テスト・ソースは、データ・ソースからではなくテスト・ソースからグラフにテスト・データが提供されるよう、このデータ・ソースを置換することができる。プローブは、グラフが実行されるとき、フローを通過するデータを監視することができ、後に検査又は再使用するために、このデータを保存することができる。たとえば、プローブは、データベースなどの出力データ受信側に他の方法で保存されていたはずのデータを受信することができる。

【0028】

オーバーレイ指定で定義される挿入は、元のアプリケーションの一部になることなく、実行中にアプリケーションに追加することができる。アプリケーションがコンパイルされると、コンパイラは、オーバーレイ・ファイルを考慮に入れ、挿入を含む実行可能なアプリケーションを生成する。アプリケーションの第1のバージョンとしての元のアプリケーション、及びアプリケーションの第2のバージョンとしての挿入を含むアプリケーションを参照する場合がある。たとえば、グラフベースのプロセスの例では、オーバーレイ指定において定義された挿入オブジェクトと結合された、グラフの第1のバージョンの構成要素を含むグラフの第2のバージョンとして、実行可能なグラフを視覚的に表すことができる。例によっては、実行可能なグラフはシェル・スクリプトであり、ファイル内には記憶されない。例によっては、実行可能なグラフ及びこのグラフは、別々のファイルに記憶される。

【0029】

グラフの第2のバージョンへの挿入の取込みは、グラフの第1のバージョンを修正しない。その代わりに、挿入定義は、別々のファイル（たとえば、別々のオーバーレイ指定）にとどまり、コード生成の開始時点において、修正済みグラフに含めるために通常のグラフ構成要素へと変換することができる。したがって、元のグラフを意図せずに破壊するリスクがない。

【0030】

図1には、グラフ100の一例が示してある。グラフ100は、フローによって接続されたデータ処理構成要素を含む、コンピュータ・プログラムの視覚表現である。2つの構成要素を接続するフローは、第1の構成要素から出力されたレコードが、第2の構成要素に渡されることを示す。フローによって第1の構成要素が第2の構成要素に接続されているとき、第1の構成要素は第2の構成要素を参照する。

【0031】

（図に示すような）データベースなどのデータ・ソース102、ファイル、キュー、実行可能文（たとえば、SQL文）、又はグラフ100の外部にある別のタイプのデータ・ソースは、グラフ100によって処理される1つ又は複数のデータ・レコードを含む。外部という言葉によって、データ・ソース102のデータが、グラフ100内に記憶されないことを意味する。データ・ソース102は、フローによってフィルタ構成要素103に接続されている。一般に、フィルタ構成要素は、所定の判定基準を満たさないレコードを

選別又は削除する。この例では、フィルタ構成要素103は、オハイオ州在住の顧客のデータ・レコードを通過させ、その他のレコードを拒否する。フィルタ構成要素103は、選別済みのデータ・レコードを郵便番号でソートする、ソート構成要素104に接続されている。ソート構成要素104は、2つの異なる方式でデータ・レコードを処理できるように、このデータ・レコードのコピーを作成する、複製構成要素106に接続されている。複製構成要素は、フォーマット変更構成要素108、及び表現式による選別の構成要素110に接続されている。たとえば、オハイオ州在住の顧客のデータ・レコードの1つの実現値は、郵便番号でソートされ、フォーマット変更構成要素108に送られ、又データ・レコードの別の実現値が、表現式による選別の構成要素110に送られる。フォーマット変更構成要素108は、データ・レコードのフォーマットをそれとは異なるデータ・フォーマットに変更し、表現式による選別の構成要素110は、データ・レコードに関連付けられた表現式に基づいて、このデータ・レコードを削除する。フォーマット変更構成要素108、及び表現式による選別の構成要素110は、受信済みのデータ・レコードを結合する集約構成要素112に接続されており、この集約構成要素は、(図に示すような)データベース、ファイル、キュー、又は下流処理構成要素など、グラフの外部に存在する出力データ受信側構成要素114に接続されている。外部という言葉によって、出力データ受信側114のデータが、グラフ100内に記憶されないことを意味する。グラフ100は、各構成要素間に多くのフローを含むが、データ・ソース102とフィルタ構成要素103との間のフロー116(ソース／フィルタ・フロー116と呼ぶことがある)、及び集約構成要素112と出力データ受信側114との間のフロー118(集約／出力フロー118と呼ぶことがある)が、この例では特に対象となる。

【0032】

グラフの各構成要素のうちの1つ又は複数は、フォーマット特有の構成要素とすることができる。フォーマット特有の構成要素は、1つ又は複数の特定のフォーマットのデータのみを処理できる構成要素である。データ・フォーマットは、個々のデータ項目の特性(たとえば、レコードのフィールド内の値の特性)、又はレコードの特性(レコード・フォーマットと呼ばれることもある)である。個々のデータ項目の特性の例は、データ項目のサイズ(たとえば、1バイトASCIIデータ項目、若しくはマルチバイト・データ項目)、データ項目のタイプ(たとえば、ストリング・タイプ、整数タイプ、ブール・タイプ、若しくは他のデータ・タイプ)、又は個々のデータ項目の別の特性など、データ項目について数多くのバイトを含む。レコード・フォーマットの例には、レコード内のフィールドの名前、レコード内のフィールドの位置、レコード内のフィールドの数、階層型レコード、フィールドの配列又は繰返しグループ、入れ子状の配列、サブレコード、又はレコードの別の特性が含まれる。

【0033】

グラフは、特定のデータ・フォーマットに特有の構成要素を含むとき、特定のフォーマットのデータのみを処理できてもよい。グラフが、互いに異なるフォーマットを有するデータを処理するのに使用される場合、エラーが発生する場合があります、又はデータが不正確に処理される場合がある。フォーマットが異なるデータをグラフが処理できるようにするには、フォーマット特有の構成要素のうちの1つ又は複数を、異なるフォーマットのデータを処理することのできる構成要素で置換することができる。置換構成要素は、様々な構成要素に特有であるフォーマット特有の構成要素とすることができ、又は任意のフォーマットのデータを処理できる構成要素(これを、フォーマットに依存しない構成要素と呼ぶことがある)とすることができる。

【0034】

たとえば、図1の例では、ソート構成要素104は、郵便番号フィールド内の値によってレコードをソートする。この例でのソート構成要素104は、整数だけを処理できるフォーマット特有の構成要素である。グラフ100のオペレータは、グラフ100を使用して、郵便番号フィールドが文字数字の列を含んでもよいデータの新規セットを処理したいと思うことがある。

【0035】

図2を参照すると、ソート構成要素104が省略され、置換ソート構成要素204が含まれる、グラフ100の第2のバージョン200が生成される。置換ソート構成要素204は、グラフの第2のバージョン200内での、ソート構成要素104と同じ位置に配置され、文字数字の列をソートすることができる。グラフの他の構成要素は変化しない。グラフの第2のバージョン200は、データの新規セットをこのように処理することができる。

【0036】

構成要素によっては、任意のフォーマットのデータを受信し、このデータに動作を実行することができるが、特定のフォーマットのデータを出力してもよい。互いに異なるフォーマットのデータが出力として望まれる場合（たとえば、特定のフォーマットを指定する別のアプリケーションへの入力として提供されるように）、グラフは、このデータを提供することができない場合がある。グラフが所望のフォーマットのデータを出力できるようにするために、フォーマット特有の構成要素のうちの1つ又は複数を、所望のフォーマットのデータを出力できる構成要素で置換することができる。

【0037】

やはり図1を参照すると、フォーマット変更構成要素108は、特定のフォーマットのデータを出力するフォーマット特有の構成要素である。たとえば、フォーマット変更構成要素108は、4つのフィールド、Name、Account__num、Balance、及びTrans__dateを有するデータ・レコードを出力してもよい。グラフ100のオペレータは、たとえば、入力データのレコード・フォーマットについて特定の要求条件を有する別のアプリケーションによって出力データを処理できるように、互いに異なるフォーマットの出力データをグラフ100が生成するようにしたいと思うことがある。この例では、出力データの所望のフォーマットは、4つのフィールド、Cust__name、Balance、Account__num、及びTrans__dateを含む。すなわち、出力データの第1のフィールドの名前を変更し、第2のフィールド及び第3のフィールドを切り替える必要がある。

【0038】

図3を参照すると、フォーマット変更構成要素108が省略され、置換フォーマット変更構成要素308が含まれる、グラフ100の第2のバージョン300が生成される。置換フォーマット変更構成要素308は、グラフの第2のバージョン300内での、フォーマット変更構成要素108と同じ位置に配置され、所望のフォーマットの出力データを生成することができる。グラフの他の構成要素は変化しない。

【0039】

例によっては、フォーマット特有の構成要素の上流又は下流の1つ又は複数の構成要素など、グラフの他の構成要素を省略することもできる。場合によっては、省略されたその他の構成要素のうちの1つ又は複数の代わりに、置換構成要素を含めることができる。

【0040】

例によっては、グラフ100の試験者は、グラフ100をデバッグして、その機能を検証したいと考えることがある。場合によっては、試験者は、データが、ある構成要素から別の構成要素に流れるとき、このデータを検証したいと思うことがある。場合によっては、試験者は、グラフ100での上流構成要素を迂回し、その代わりに、迂回された構成要素の位置にデータを挿入したいと思うことがある。場合によっては、試験者は、グラフを変更することの、グラフが出力するデータへの影響を監視するために、入力データの一貫したセットを使用してグラフ100の動作をテストしたいと思うことがある。場合によっては、試験者は、グラフの機能の全てを少なくとも1回だけは実行できるようにすることを知っている入力データのセットを使用して、グラフ100の動作をテストし、したがってグラフの完全なテストを可能にしたいと思うことがある。

【0041】

グラフ100をデバッグする際、グラフの修正をやめることが望ましい場合がある。た

10

20

30

40

50

例えば、試験者は、グラフの機能を損なうリスクを負いたくないと思うことがある。例によっては、試験者は、グラフへのアクセスを制限されているか、又はアクセスできない場合がある（たとえば、試験者には、グラフを編集するのに必要な許可がない場合がある）。グラフ100を修正することなく、このグラフ100をデバッグするために、オーバーレイを使用してグラフをデバッグすることができる。例によっては、たとえばグラフの自動解析に基づいて、このオーバーレイを自動的に指定することができる。元のグラフ100（グラフの第1のバージョンと呼ばれることがある）、及びオーバーレイ指定に基づいて、グラフ100の少なくとも一部分の第2のバージョンを生成することができる。

【0042】

グラフ100の各構成要素間のフローをデータが通過するとき、たとえば、第1の構成要素から第2の構成要素までのフローに沿って、又は出力データ受信側までのフローに沿って、プローブが、このデータを収集又は監視する。たとえば、グラフ100が実行される際にデータがフローを通過するとき、このデータを監視し、後で検査するために保存し、又は再使用するために保存することができる。オーバーレイ指定は、収集又は監視すべきデータを運ぶフローを参照するプローブを定義することができる。プローブは、データを収集又は監視すべきフローを指定する。プローブは、特定の値を報告し、又は特定の値が、所定範囲内又は所定範囲外にあるときに、そのことを報告するように構成することができる。プローブを通過するデータは、後で解析又は使用するために保存してもよく、たとえば、このデータは、単層ファイル又はリレーショナル・データベースに記憶することができる。

【0043】

例によっては、プローブは、グラフ100のある構成要素から、ファイル又はデータベースなど出力データ受信側までのフローを参照することができる。グラフ100のデバッグ中に、データ受信側までのフローに沿ってプローブを配置することによって、このプローブは、グラフ100から出力されるデータを受信する。たとえば、グラフ100がデバッグ・モードで実行されるたびに、出力データをプローブによって受信し、ファイルに書き込むことができ、その結果、様々なグラフ実行からの出力データを比較することができ、又は他の方法で評価することができる。例によっては、出力データ受信側が自動的に識別され、オーバーレイが自動的に指定されて、識別された出力データ受信側の前に挿入するためのプローブを定義する。

【0044】

例によっては、プローブは、グラフ100の上流構成要素から下流構成要素までのフローを参照することができる。グラフ100のデバッグ中に、下流構成要素までのフローに沿ってプローブを配置することによって、このプローブは、下流構成要素によって他の方法で受信されていたはずのデータを受信し、したがって下流構成要素が実行しないようにする。たとえば、試験者は、下流構成要素の前にグラフが処理する結果を監視したいと考えることがある。たとえば、この下流構成要素は、グラフの外部に影響を及ぼす機能を有してもよく、たとえば、下流構成要素は、この下流構成要素によって自身のクレジット・カードのレコードが処理されるそれぞれの人に向けて、テキスト・メッセージを送信してもよい。グラフのデバッグ中、試験者は、グラフの外部に影響を及ぼすような構成要素を使用不可にしたいと考えることがある。

【0045】

グラフ100の2つの構成要素間の特定のフローにおいて、テスト・ソースがグラフ100にデータを挿入する。オーバーレイ指定は、テスト・ソースからのデータで置換すべきデータを運ぶフローを参照する、テスト・ソースを定義することができる。例によっては、テスト・ソースは、通常、フローを通過するはずのデータを、新規のデータで置換する。状況によっては、テスト・ソースは、これまでに保存されたデータを読み取り、このデータを下流構成要素に渡すように構成することができる。例によっては、テスト・ソースは、データベース又はファイルなど、データ・ソースからのフローにおいて、データをグラフ100に挿入する。テスト・ソースは、データ・ソースによって他の方法で提供され

たはずのデータと同じフォーマットを有するデータを挿入することができる。例によっては、データ・ソースが自動的に識別され、オーバーレイが自動的に指定されて、識別されたデータ・ソースを置換するためのテスト・ソースを定義する。

【0046】

例によっては、あるポイントまで（たとえば、ある構成要素まで）グラフ100を実行した結果が、これまでに検証されていた場合がある。すなわち、上流プロセス機能が、あるポイントまで検証されていた場合がある。こうした場合、グラフ100が実行されるたびに上流構成要素が各機能を再処理することは、非効率になる場合がある。テスト・ソースは、そのあるポイントにおいて、データ（たとえば、これまでに検証されたデータ）をグラフに挿入することができる。このようにして、これまでに実行されたグラフ100のセクション全体を迂回してもよい。

10

【0047】

図4には、1つ又は複数の挿入を定義するオーバーレイ指定200の一例が示してある。挿入は、グラフ（たとえば、グラフ100）のフローに関連付けられたオブジェクトとすることができ、プローブ、テスト・ソース、又は置換構成要素の形をとることができる。図4の例では、オーバーレイ指定200は、1つのテスト・ソース定義201、及び1つのプローブ定義213を含む。オーバーレイ指定200は、グラフ100についての仕様を含むファイルとは別のファイルなど、ファイルに記憶することができる。

【0048】

オーバーレイ指定200は、挿入定義が対応できる先のグラフを指定する、3行のヘッダで開始する。ヘッダの後には、テスト・ソース定義201、プローブ定義213、及び置換構成要素定義（図示せず）が続く。

20

【0049】

テスト・ソース定義201は、名前202、上流ポート204、下流ポート206、挿入タイプ208、プロトタイプ・パス210、及びレイアウト・パラメータ212を含む。

【0050】

テスト・ソース定義201の上流ポート204は、テスト・ソースをグラフ100に挿入すべきフローのすぐ上流にある構成要素の出力ポートを参照する。フローの上流にある構成要素は、その出力ポートからフロー上にデータが出力される元の構成要素である。図4の例では、テスト・ソース定義201の上流ポート204が、データベース102の出力を指す。テスト・ソース定義201の下流ポート206は、テスト・ソースをグラフ100に挿入すべきフローのすぐ下流にある構成要素の入力ポートを参照する。フローの下流にある構成要素は、その入力ポートにおいてフローからデータが受信される構成要素である。図4の例では、テスト・ソース定義の下流ポート206が、フィルタ構成要素103の入力を指す。したがって、この例でのテスト・ソース定義201は、データベース102の出力と、フィルタ構成要素103の入力との間のフローにテスト・ソースが配置されることになり、その結果、このテスト・ソースが提供するデータが、データベース102からの入力データを置換できることを示す。

30

【0051】

挿入タイプ208は、挿入がテスト・ソースなのか、プローブなのか、それとも置換構成要素なのか定義する。図4の例では、値「0」がテスト・ソースを定義し、値「1」がプローブを定義し、値「2」が置換構成要素を定義する。他の値を使用して、挿入のタイプを定義することもできる。この挿入はテスト・ソースなので、挿入タイプ208の値は「0」である。

40

【0052】

プロトタイプ・パス210は、挿入のタイプを示す。この例では、この挿入はテスト・ソースなので、プロトタイプ・パス210は、入力ファイル構成要素を指定する。プロトタイプ・パス210は、特定のタイプの挿入を定義するコードを含むファイルを指す。レイアウト・パラメータ212は、テスト・ソースが含むことになるデータを含むソース・ファイルの位置を定義する。例によっては、この位置はファイル・パスである。ソース・

50

ファイル内のデータは、通常、上流ポート 204 及び下流ポート 206 によって定義されるフローを通過するはずのデータを置換することになる。すなわち、テスト・ソースがグラフ 100 に適用されると、フィルタ構成要素 103 が、データベース 102 からのデータを受信するのではなく、ソース・ファイル内のデータを受信する。

【0053】

このソース・ファイルは、テスト・ソースの下流の構成要素によって他の方法で受信されるはずのデータと同じフォーマットを有するデータを含む。例によっては、ソース・ファイル内のデータは、テスト・ソースの上流にあるデータ・ソース（たとえば、データベース）内のデータと同じでもよい。たとえば、データベース 102 からのデータ・レコードを、ソース・ファイルにコピーすることができる。例によっては、データ・ソースは、SQL クエリなどの実行可能文を示す。これらの例では、SQL クエリを実行することができ、このクエリ実行の結果をソース・ファイルに記憶することができる。例によっては、ソース・ファイル内のデータは、データ・ソース以外のどこかから取得することができる。たとえば、ソース・ファイル内のデータを生成して、グラフ 100 を完全にデバッグするために、あるデータ（たとえば、ある範囲の値）を確実に処理することができる。例によっては、データ・ソース内のデータが変更されても、ソース・ファイル内のデータは同じままであり、したがって、入力データの一貫したセットでデバッグを継続できるようになる。

【0054】

例によっては、ソース・ファイル内のデータは、グラフ 100 の通常の実行中にフローを通過するはずのデータと同じでもよいが、テスト・ソースを使用してこのデータを挿入することによって、上流構成要素は処理を控えることができる。たとえば、複製構成要素 106 などの上流構成要素は、データを処理するのに大量のシステム・リソースを必要とすることがあり、又は、データ・フロー・グラフ 100 内の他の構成要素と比較して、データを処理するのに要する時間が相対的に長くなることがある。したがって、既知のデータ（たとえば、通常の実行中にフローを通過するはずの同じデータ）をフローに挿入して、時間を節約し、又はシステム・リソースを保護することができる。

【0055】

プローブ定義 213 は、名前 214、上流ポート 216、下流ポート 218、挿入タイプ 220、及びプロトタイプ・パス 222 を含む。

【0056】

プローブ定義 213 の上流ポート 216 は、プローブをグラフ 100 に挿入すべきフローのすぐ上流にある構成要素の出力ポートを参照する。図 4 の例では、プローブ定義 213 の上流ポート 216 が、集約構成要素 112 の出力を指す。プローブ定義 213 の下流ポート 218 は、プローブをグラフ 100 に挿入すべきフローのすぐ下流にある構成要素の入力ポートを参照する。図 4 の例では、プローブ定義 213 の下流ポート 218 が、出力データ受信側構成要素 114 を指す。したがって、この例でのプローブ定義 213 は、集約構成要素 112 の出力と、出力データ受信側構成要素 114 との間のフローにプローブが配置されることになり、その結果、出力データ受信側構成要素に他の方法で書き込まれたはずのデータをプローブが受信することを示す。

【0057】

プローブ定義 213 の挿入タイプ 220 は、挿入がテスト・ソースなのか、プローブなのか、それとも置換構成要素なのか定義する。この挿入はプローブなので、挿入タイプ 220 の値は「1」である。

【0058】

プロトタイプ・パス 222 は、挿入のタイプを示す。この例では、この挿入はプローブなので、プロトタイプ・パス 222 は出力ファイル構成要素を指定する。プロトタイプ・パス 222 は、特定のタイプの挿入を定義するコードを含むファイルを指す。

【0059】

例によっては、プローブによって監視すべきデータは、システムによって自動的に作成

10

20

30

40

50

されるファイルに記憶される。このファイルは、システムによって決定される位置に記憶することができる。プローブは、上流ポート216及び下流ポート218によって定義されたフローを通過するデータを監視する。すなわち、プローブがグラフ100に適用されると、集約構成要素112の出力から、出力データ受信側構成要素114の入力まで通過するデータが監視され、システムによって自動的に作成されるファイルに記憶される。例によっては、このデータを、記憶する前に監視することができる。ファイルは、プローブ定義によって参照される構成要素（この例では、外部のデータ受信側構成要素114）によって受信されたはずの同じフォーマットのデータを受信することができる。

【0060】

例によっては、グラフ100の自動解析の結果として、オーバーレイ指定によって、1つ又は複数のプローブ又はテスト・ソースの挿入を定義することができる。たとえば、グラフ100の自動解析を実行して、データベース、ファイル、又は他のタイプのデータ・ソースなど、任意のデータ・ソースを識別することができる。識別されたデータ・ソースのうちの1つ又は複数を、テスト・ソースで自動的に置換することができる。置換されたデータ・ソースにより、データ・ソースのすぐ下流のフローにテスト・ソースが挿入され、その結果、データ・ソースからのデータではなく、このテスト・ソースからのデータが下流構成要素に提供されることを意味する。同様に、グラフ100の自動解析は、データベース、ファイル、又は他のタイプの出力データ受信側など、任意の出力データ受信側を識別することができる。識別された出力データ受信側のうちの1つ又は複数を、プローブで自動的に置換することができる。置換された出力データ受信側により、出力データ受信側のすぐ上流のフローにプローブが挿入され、その結果、上流構成要素からのデータが、出力データ受信側ではなくプローブによって受信されることを意味する。グラフ100の自動解析を使用して、特定のタイプの構成要素（たとえば、実行するとグラフ100の外部に影響を及ぼす特定のタイプの構成要素）など、他の構成要素を識別することもできる。

【0061】

テスト・ソース及びプローブの挿入のさらなる説明が、米国特許出願第14／715,807号に提示されており、その内容全体を参考としてここに組み込む。

【0062】

置換構成要素定義は、名前、上流ポート、下流ポート、挿入タイプ、プロトタイプ・パス、及びレイアウト・パラメータを含む。置換構成要素定義の上流ポートは、この置換構成要素がグラフ100内に挿入される位置のすぐ上流にある構成要素の出力ポートを参照する。置換構成要素定義の下流ポートは、この置換構成要素がグラフ内に挿入される位置のすぐ下流にある構成要素の入力ポートを参照する。上流ポート及び下流ポートに基づいて、置換構成要素によって置換すべきグラフ100内の既存の構成要素を識別することができる。挿入タイプは、この挿入が置換構成要素であると定義する。

【0063】

プロトタイプ・パスは、挿入のタイプを示す。この例では、この挿入が置換構成要素なので、プロトタイプ・パスは、置換構成要素を定義するコードを含むファイルを指す。置換構成要素を定義するコードは、置換すべき既存の構成要素を定義するコードに基づいているが、所望のフォーマットのデータを処理することができる。

【0064】

例によっては、グラフの自動解析の結果として、オーバーレイ指定によって、グラフ用の1つ又は複数の置換構成要素を定義することができる。たとえば、グラフ内の各構成要素の仕様を解析することができる。構成要素の仕様は、この構成要素を定義するコード、たとえば、この構成要素によって表されるデータ処理動作を定義するコードを含むか、又はそれを指す。コードを解析すると、構成要素によって表されるデータ処理動作がデータのフォーマットに依存しているかどうかを明らかにすることができる。

【0065】

置換構成要素は、識別されたフォーマット特有の構成要素のうちの1つ又は複数について定義される。例によっては、置換すべきフォーマット特有の構成要素は、ユーザ入力に

10

20

30

40

50

基づいて識別される。たとえば、ユーザは、入力データのフォーマット、フォーマット特有の構成要素のそれぞれによって表されるプロセス、又はその両方についての自らの知識を使用して、構成要素のいずれかを置換するか判定してもよい。例によっては、グラフによってこれまでに処理されたデータのフォーマットに対して入力データのフォーマットの自動解析を実行して、フォーマット特有の構成要素のいずれかを置換するかを識別することができる。

【0066】

例によっては、グラフベースではないコンピュータ・プログラムでは、コンピュータ・プログラム内の1つ又は複数のフォーマット特有のプロセスを識別し、1つ又は複数の他のプロセス、たとえば、特定のフォーマットのデータに対して作用できるプロセス、又は任意のフォーマットのデータに対して作用できるプロセスでこれを置換することができる。

【0067】

図5を参照すると、テスト・ソース、プローブ、又はその両方を挿入するため、解析エンジン300が、グラフ100を自動的に解析して、データ・ソース302及び出力データ受信側304を識別する。たとえば、解析エンジン300は、グラフ100のそれぞれのノードについて、パラメータ及び接続にアクセスすることができる（用語「ノード」及び「構成要素」は、これを区別なく使用することがある）。所与のノードが入力接続をもたない場合、解析エンジン300は、このノードをデータ・ソースと識別する。同様に、所与のノードが出力接続をもたない場合、解析エンジン300は、このノードを出力データ受信側と識別する。グラフのそれぞれのノードにアクセスし、これを解析するため、解析エンジンは、グラフの各接続（用語「接続」及び「フロー」は、これを区別なく使用することがある）の全てを「探索」して回る。例によっては、実行時（たとえば、デバッグ目的で処理が開始するとき）まで、グラフ100はインスタンス化又はパラメータ化されない。解析エンジン300は、実行時に自動解析を実行して、グラフ100内のデータ・ソース及び出力データ受信側を識別することができる。

【0068】

解析エンジン300は、データ・ソース302及び出力データ受信側304の識別子を挿入エンジン306に送信し、この挿入エンジンは、データ・ソースと出力データ受信側のいずれが、それぞれテスト・ソース及びプローブで置換されることになるのか判定する。例によっては、試験者308は、テスト・ソース及びプローブで置換すべき、データ・ソース及び出力データ受信側のリスト310を提供する。このリスト310は、ファイル、データベースとして、又は別のフォーマットで提供することができる。たとえば、試験者308は、頻繁に変更されると予想される任意のデータ・ソースを、リスト310上にもっていてもよい。このようなデータ・ソースをテスト・ソースで置換することにより、試験者308は、一貫した入力データを使用してグラフを確実にテストできるようにすることができる。

【0069】

挿入エンジン306は、識別されたそれぞれのデータ・ソース302及び出力データ受信側304と、リスト310上のデータ・ソース及び出力データ受信側とを比較する。挿入エンジンは、リスト310上に表示される任意のデータ・ソース302又は出力データ受信側304について、オーバーレイ指定312を作成する。例によっては、上流ポートや下流ポートなど、オーバーレイ指定312についてのパラメータが、解析エンジン300によって挿入エンジン306に提供される。例によっては、挿入エンジン306は、グラフ100にアクセスして、関連するパラメータを取得する。

【0070】

テスト・ソースについてオーバーレイ指定312を作成するため、挿入エンジン306は、ソース・ファイルにデータを読み込む。例によっては、挿入エンジン306は、特定のデータ・ソース302に取って代わることになるテスト・ソース用のソース・ファイルに、データ・ソース302からコピーされたデータを読み込む。例によっては、データ・ソース302は、SQL文などの実行可能な表現式を含み、挿入エンジン306は、この実

10

20

30

40

50

行可能な表現式を実行し、ソース・ファイルに実行した結果を読み込む。例によっては、挿入エンジン306は、ユーザ・インターフェース314を介して、ソース・ファイル用のデータを試験者308に要求することができる。たとえば、挿入エンジン306は、識別されたデータ・ソース302のリストを試験者308に提供することができ、その結果、試験者308は、識別されたデータ・ソース302のうちのいずれがテスト・ソースで置換されることになるのか選択することができる。試験者308は、テスト・ソース用のソース・ファイルに含むべきデータを指定することもできる。場合によっては、試験者308は、テスト・ソース用のデータを含むファイルの位置（たとえば、パス）を識別することができる。場合によっては、試験者308は、元のデータ・ソース302内のデータのコピーであるソース・ファイルを生成するよう、挿入エンジン308に指示することができる。場合によっては、試験者308は、元のデータ・ソース302に含まれるか、又はそれに関連付けられた、SQL文などの実行可能な表現式を実行するよう、挿入エンジン308に指示することができる。場合によっては、試験者308は、テスト・ソースのソース・ファイル用に、データを生成できるようにすることができる。たとえば、試験者308は、グラフ内のあらゆる機能が少なくとも1回実行されるようにする、実データ又は生成済みデータなどのデータのセットを提供してもよい。

10

【0071】

プロンプ用のオーバーレイ指定312を作成するため、挿入エンジン308は、出力データを記憶すべきファイルの位置を決定する。例によっては、この位置は、たとえばシステム技術者によって、デフォルトに設定される。例によっては、挿入エンジン306は、ユーザ・インターフェース314を介して、出力データ・ファイルについての位置を指定するよう試験者308に要求することができる。

20

【0072】

置換構成要素を挿入するため、解析エンジン300は、グラフ100を解析して、グラフ内の1つ又は複数のフォーマット特有の構成要素305を識別する。グラフの各構成要素を解析するため、解析エンジン300は、グラフの接続の全てを「探索」して回る。例によっては、解析エンジン300は、グラフの最も遠い上流構成要素から開始し、この上流構成要素からの各出力フローを「探索」して回り、したがって、最終的にはグラフの構成要素の全てを解析してもよい。逆に言えば、解析エンジン300は、グラフの最も遠い下流構成要素から開始し、この下流構成要素への各入力フローを「探索」して回り、したがって、最終的にはグラフの構成要素の全てを解析してもよい。

30

【0073】

解析エンジン300は、グラフ100内の各構成要素の仕様にアクセスすることができる。構成要素の仕様は、この構成要素を定義するコード、たとえば、この構成要素によって表されるデータ処理動作を定義するコードを含むか、又はそれを指す。コードの解析に基づいて、解析エンジン300は、データ処理動作がデータのフォーマットに依存しているかどうかを判定することができる。解析エンジン300は、フォーマット特有の構成要素の識別子を挿入エンジン306に送信し、この挿入エンジン306は、置換構成要素のためにフォーマット特有の構成要素のうちのいずれかを省略するか決定する。

【0074】

例によっては、解析エンジン300は、ユーザの要求に応じてグラフ100を解析する。たとえば、ユーザは、グラフ100を使用して、通常このグラフによって処理されるデータとは異なるフォーマットのデータを処理したいと思うことがある。ユーザは、グラフ100の解析を要求して、異なるフォーマットのデータをグラフ100が確実に処理できるようにすることができる。

40

【0075】

例によっては、たとえば、グラフが最初に定義されるとき、又はグラフが最初にインスタンス化若しくはパラメータ化されるとき（たとえば、グラフの第1の実行時に）、解析エンジン300がグラフ100を一度解析して、グラフ内のフォーマット特有の構成要素全てのリストを生成する。グラフ内のフォーマット特有の構成要素のリストは、先々参照

50

するために記憶して、たとえば、互いに異なるフォーマットのデータを処理するためにグラフ100を使用するというユーザの要求にตอบสนองして使用することができる。

【0076】

例によっては、解析エンジン300は、グラフをいつ解析するかを自動的に決定する。たとえば、グラフの仕様は、このグラフによってこれまでに処理されたデータのフォーマットの記述を含んでもよい。入力データのフォーマットが、これまでに処理されたデータのフォーマットと異なる場合、解析エンジンは、グラフを解析して、異なるフォーマットの入力データを処理するために何らかの構成要素を置換する必要があるかどうかを判定してもよい。

【0077】

挿入エンジン306は、解析エンジン300によって識別されるフォーマット特有の構成要素のいずれかを省略するか判定する。挿入エンジン306は、省略される構成要素のそれぞれについて置換構成要素を定義するオーバーレイ指定を作成する。

【0078】

例によっては、省略すべきフォーマット特有の構成要素は、ユーザによって識別される。たとえば、挿入エンジン306は、識別済みのフォーマット特有の構成要素のリストをユーザ・インターフェース314に表示することができ、ユーザは、置換すべき構成要素を選択する。ユーザは、置換すべき構成要素のそれぞれについて置換構成要素として使用される構成要素を指示することができる。たとえば、ユーザは、入力データのフォーマット、フォーマット特有の構成要素のそれぞれによって表されるプロセス、又はその両方についての自らの知識を使用して、構成要素のいずれかを省略すべきか、又どの構成要素を置換構成要素として含めるべきか判定してもよい。省略すべき構成要素、置換構成要素、又はその両方を識別するユーザ入力に基づいて、挿入エンジン306はオーバーレイ指定を作成する。

【0079】

例によっては、挿入エンジン306は、フォーマット特有の構成要素のいずれかを省略すべきか、又どの構成要素を置換構成要素として含めるべきか自動的に決定することができる。たとえば、挿入エンジンは、フォーマット特有の構成要素のそれぞれの仕様を解析して、どの構成要素が入力データを処理できるか、又は処理できないか判定することができる。挿入エンジン306は、置換構成要素、たとえば、置換すべき対応する構成要素によって表されるのと同じデータ処理動作を表すが、入力データのフォーマットのデータを処理できる置換構成要素を自動的に識別することができる。挿入エンジン306は、自動的に識別された置換構成要素についてのオーバーレイ指定を作成する。例によっては、ユーザ入力は、自動化判定に組み込まれる。たとえば、挿入エンジン306によって識別される置換構成要素を承認するよう、ユーザに求めてもよい。

【0080】

図6には、グラフ用の置換構成要素を定義するための一般的な手法が示してある。グラフによる処理のために、特定のフォーマットを有するデータのセットが受信される(400)。受信データの特定のフォーマットを処理する能力について、グラフを解析すべきかどうかに関して判定がなされる(402)。例によっては、ユーザは、たとえばユーザ・インターフェースを介して、グラフを解析すべきであると示すことができる。たとえば、ユーザは、このデータのセットが、グラフによって処理されたこれまでのデータとは異なるフォーマットを有することを知っている場合がある。例によっては、この判定を自動的に起こすことができる。たとえば、受信データのフォーマットを決定し、たとえば、グラフが構成されるデータのフォーマットを示すグラフの仕様に記憶された情報と比較することができる。グラフが構成されるデータのフォーマットと受信データのフォーマットが一致しない場合、グラフが解析される。

【0081】

グラフは、たとえばプロセッサによって解析されて、構成要素によって処理されるデータのフォーマットに依存する、グラフの1つ又は複数の構成要素を識別する(404)。

10

20

30

40

50

具体的には、グラフの構成要素のうちの1つ又は複数のそれぞれについての仕様が解析されて、グラフ内のフォーマット特有の構成要素を識別する。例によっては、グラフは、このグラフの各構成要素を介する段階的進行によって解析される。たとえば、各構成要素を解析して、この構成要素がフォーマット特有の構成要素かどうかを判定するとともに、この入力フローと出力フローを識別する。各フローは隣接する構成要素へと続き、これら構成要素のそれぞれが解析されて、この構成要素がフォーマット特有であるかどうかを判定し、入力フロー及び出力フローを識別する。このようにして、グラフの構成要素の全てを解析することができる。例によっては、この解析は、実行時に、たとえばグラフがパラメータ化された後に、自動的に実行することができる。例によっては、この解析は、自動的かつ動的に、たとえばグラフが実行されている間に実行することができる。たとえば、グラフの実行中に、あるパラメータが解決されると、動的な解析を実行することができる。例によっては、グラフは短期記憶内に受信され、そこからプロセッサによってグラフが解析されて、フォーマット特有の構成要素を識別する。

10

【0082】

フォーマット特有と識別された構成要素のうちの1つ又は複数进行评估して、構成要素を省略すべきかどうか、及び置換構成要素が含まれているかどうかを判定する(406)。たとえば、構成要素が、受信済みのデータのセットのフォーマットを有するデータを処理することができない場合、フォーマット特有の構成要素を省略してもよい。例によっては、フォーマット特有の構成要素のリストがユーザ・インターフェースに表示され、ユーザが、構成要素のいずれかを省略すべきか指定する。例によっては、フォーマット特有の構成要素のそれぞれの仕进行评估して、受信済みのデータのセットのフォーマットを有するデータを、この構成要素が処理できるかどうかを自動的に判定する。例によっては、フォーマット特有と識別された構成要素の全てが省略される。

20

【0083】

省略すべきフォーマット特有の構成要素のうちの1つ又は複数のそれぞれでの置換構成要素について、オーバーレイ指定が定義される(408)。所与の置換構成要素の仕様は、対応する省略済みのフォーマット特有の構成要素の仕様に基づいているが、受信済みのデータのセットのフォーマットを有するデータに実行できる、1つ又は複数のデータ処理動作を定義する。例によっては、置換構成要素は、受信済みのデータのセットのフォーマットに対して、フォーマット特有のものとすることができる。例によっては、置換構成要素は、一般的なもの、たとえば任意のフォーマットのデータを処理できるものとすることができる。

30

【0084】

グラフの実行に先立って、コンパイラが、グラフを実行可能なグラフにコンパイルしてもよい(410)。コンパイルの一環として、コンパイラは、置換構成要素を定義するオーバーレイ指定200を考慮に入れる。たとえば、コンパイラは、入力としてオーバーレイ指定200を受け付けてもよい。グラフの第2のバージョンが生成され、置換用に識別されたフォーマット特有の構成要素が除去され、除去済みの構成要素の代わりのオブジェクトとして、1つ又は複数の置換構成要素がグラフの第2のバージョンに挿入される。置換構成要素は、(除去済みの構成要素以外の)グラフ100の第1のバージョンに含まれるデータ処理構成要素とともに、グラフの第2のバージョンで表してもよい。オーバーレイ指定200、又はこのオーバーレイ指定を記憶するファイルは、グラフを含むファイルから分離されたままである。すなわち、置換構成要素は、グラフの第1のバージョンに含まれるデータ処理構成要素とともに、グラフの第2のバージョンに表示してもよいが、グラフの第1のバージョンを含むファイルは、置換構成要素の定義を含まない。

40

【0085】

オーバーレイ指定において定義されるテスト・ソース、プローブ、又は置換構成要素などの挿入は、少なくとも2つのモード、すなわち単一実行モード(Single-Execution Mode)及び保存状態モード(Saved-State Mode)のうちの1つを使用して実行できる。

【0086】

50

図 7 には、単一実行モードで挿入定義を実行するための、例示的なシステムが示してある。この例では、クライアント 602 が、グラフ 604 の第 1 のバージョン、及び挿入を定義するオーバレイ・ファイル 606（たとえば、オーバレイ指定）を生成又は参照する。たとえば、オーバレイ・ファイル 606 は、図 4 のオーバレイ指定 200 でもよい。次いで、グラフ 604 が、コンパイラ 608 によってコンパイルされる。コンパイラ 608 は、オーバレイ・ファイル 606 を考慮に入れ、グラフの第 2 のバージョンを作成する。グラフの第 2 のバージョンは実行可能であり、オーバレイ・ファイル 606 によって定義される挿入を含む。次いで、グラフの第 2 のバージョンは、実行することができる。例によっては、コンパイルと実行は同時に発生する。グラフの第 2 のバージョンが再び実行されることになる場合、グラフ 604 の再指定、再コンパイル、及びグラフの第 2 のバージョンの再実行を含む、このプロセスが繰り返される。実行可能なグラフの、ある実行から次の実行までの情報は保存されない。

【0087】

図 8 には、保存状態マネージャ 708 を用いて、保存状態モードで挿入定義を実行するための、例示的なシステムが示してある。この例では、クライアント 702 が、グラフ 704、及び挿入を定義するオーバレイ・ファイル 706（たとえば、オーバレイ指定）を生成又は参照する。たとえば、オーバレイ・ファイル 706 は、図 4 のオーバレイ指定 200 でもよい。保存状態リポジトリ 710 が、保存状態マネージャ 708 及びコンパイラ 712 によって管理される。保存状態マネージャ 708 は、この保存状態データが保存状態リポジトリ 710 内のどこに配置されているか識別することもできる。グラフ 704 が、コンパイラ 712 によってコンパイルされる。コンパイラ 712 は、オーバレイ・ファイル 706 を考慮に入れ、オーバレイ・ファイル 706 によって定義される挿入を含むグラフの第 2 のバージョンを作成する。次いで、グラフの第 2 のバージョンは、実行することができる。例によっては、コンパイルと実行は同時に発生する。保存状態モードによって、実行可能なグラフが、各実行間で情報を保存している間に何回も実行できるようになるという点で、保存状態モードは単一実行モードと異なる。

【0088】

保存状態マネージャ 708 は、保存状態マネージャ・ディレクトリに存在することができ、保存状態を管理する。保存状態リポジトリ 710 に保存できる情報の例には、情報のうちでもとりわけ、プローブ挿入に関連する情報、テスト・ソース挿入に関連する情報、置換構成要素挿入に関連する情報、オーバレイ・ファイル 706 に関連する情報、及びグラフ構成要素に関連付けられたパラメータ（たとえば属性）が含まれる。

【0089】

例によっては、実行可能なグラフが実行されるとき、グラフの特定の部分のみが実行される。すなわち、グラフの特定の構成要素のみが実行される。例によっては、グラフの構成要素の全てよりも少ない構成要素が実行される。実行可能なグラフは、挿入に影響を及ぼすことになる構成要素を実行するだけでよい。例によっては、グラフの第 2 のバージョンは、元のグラフ全体の第 2 のバージョンである。例によっては、グラフの第 2 のバージョンは、元のグラフ全体のほんの一部分の第 2 のバージョン、たとえば、定義された挿入に関連したグラフの部分のみの第 2 のバージョンである。たとえば、最も上流の置換構成要素の上流にある構成要素は、グラフの第 1 のバージョンによって実行してもよく、最も上流の置換構成要素で開始する構成要素は、グラフの第 2 のバージョンによって実行してもよい。

【0090】

図 9 には、ここで説明する置換構成要素技法を使用できる、例示的なデータ処理システム 800 が示してある。このシステム 800 は、記憶装置又はオンライン・データ・ストリームへの接続など、1 つ又は複数のデータのソースを含んでもよいデータ・ソース 802 を含み、そのそれぞれが、様々なフォーマット（たとえば、データベース・テーブル、スプレッドシート・ファイル、フラット・テキスト・ファイル、又はメインフレームが使用するネイティブ・フォーマット）のうちの任意のフォーマットで、データを記憶又は提

供してもよい。実行環境 804 及び開発環境 818 は、たとえば、あるバージョンの UNIX オペレーティング・システムなど、適切なオペレーティング・システムの制御下で、1 つ又は複数の汎用コンピュータ上にホスティングしてもよい。たとえば、実行環境 804 は、複数ノードの並列コンピューティング環境を含むことができ、これは、複数の中央処理装置 (CPU) 若しくはプロセッサ・コア、ローカル・システム (たとえば、対称型マルチプロセッシング (SMP) コンピュータなどのマルチプロセッサ・システム) 若しくはローカル分散システム (たとえば、クラスタ若しくは超並列処理 (MPP) として結合された複数のプロセッサ)、又は、遠隔プロセッサ若しくは遠隔分散プロセッサ (たとえば、ローカル・エリア・ネットワーク (LAN) 及び／又は広域ネットワーク (WAN) を介して結合されたマルチプロセッサ)、又はその任意の組合せを使用するコンピュータ・システムの構成を含む。

10

【0091】

実行環境 804 は、データ・ソース 802 からデータを読み取り、出力データを生成する。データ・ソース 802 を設ける記憶装置は、実行環境 804 に対してローカルでもよく、たとえばこれは、実行環境 804 をホスティングするコンピュータに接続された記憶媒体 (たとえばハード・ドライブ 808) に記憶され、又は実行環境 804 に対して遠隔でもよく、たとえばこれは、(たとえばクラウド・コンピューティング・インフラストラクチャによって提供される) 遠隔接続を介して、実行環境 804 をホスティングするコンピュータと通信する遠隔システム (たとえばメインフレーム 810) 上にホスティングされる。データ・ソース 802 は、テスト・ソース定義 (たとえば、図 4 のテスト・ソース定義 201) で定義されるデータを含んでもよい。すなわち、テスト・ソース定義 201 のレイアウト・パラメータ 212 が、データ・ソース 802 内のソース・ファイルの位置を指してもよい。

20

【0092】

出力データは、実行環境 804 からアクセス可能なデータ・ソース 802 又はデータ記憶システム 816 に記憶して戻してもよく、又は他の方法で使用してもよい。データ記憶システム 816 は、開発者 820 がグラフを、開発、デバッグ、及びテストすることのできる開発環境 818 からもアクセス可能である。実装形態によっては、開発環境 818 は、各接点間の (作業要素すなわちデータのフローを表す) 有向フローによって接続される (データ処理構成要素又はデータセットを表す) 節点を含むグラフとして、アプリケーションを開発するためのシステムである。たとえば、「グラフベースのアプリケーション用のパラメータの管理 (Managing Parameters for Graph-Based Applications)」と題する、米国特許第 2007/0011668 号に、このような環境がより詳細に記載してあり、これを参考として本明細書に援用する。「グラフとして表現された計算の実行 (EXECUTING COMPUTATIONS EXPRESSED AS GRAPHS)」と題する、米国特許第 5,966,072 号に、このようなグラフベースの計算を実行するためのシステムが記載してあり、これを参考として本明細書に援用する。このシステムに従って作成されるグラフは、グラフ構成要素によって表される個々のプロセスとの間で情報を取得し、各プロセス間で情報を転送し、各プロセスについて実行順序を定義するための方法を提供する。このシステムは、利用可能な任意の方法からプロセス間通信の方法を選択するアルゴリズムを含む (たとえば、グラフのフローによる通信経路は、TCP/IP 若しくは UNIX のドメイン・ソケットを使用することができ、又は共有メモリを使用して、各プロセス間でデータを渡すことができる)。

30

40

【0093】

開発環境 818 は、ソース・コードを記憶するためのコード・リポジトリ 822 を含む。例によっては、ソース・コード及びオーバーレイ指定 (たとえば、図 4 のオーバーレイ指定 220) は、たとえばユーザ・インターフェースを介して開発環境にアクセスする開発者 820 によって開発してもよい。例によっては、ソース・コード及びオーバーレイ指定は、たとえば、前述の解析エンジン 300 及び挿入エンジン 306 によって自動的に決定される。例によっては、グラフ及びオーバーレイ指定は、コード・リポジトリ 822 に記憶する

50

ことができる。例によっては、コード・リポジトリ 8 2 2 にグラフが記憶され、別のオーバーレイ・リポジトリ 8 2 4 にオーバーレイ指定が記憶される。

【0094】

コード・リポジトリ 8 2 2 及びオーバーレイ・リポジトリ 8 2 4 のうちの 1 つ又は両方が、コンパイラ 8 2 6 と通信してもよい。コンパイラ 8 2 6 は、グラフの第 1 のバージョン及びオーバーレイ指定（たとえば、図 4 のオーバーレイ指定 2 0 0）を、グラフの実行可能な第 2 のバージョン 8 2 8 にコンパイルすることができる。たとえば、コンパイラは、入力としてオーバーレイ指定を受け付けてもよい。1 つ又は複数の挿入が処理され、オーバーレイ指定に含まれる挿入定義にそれぞれが対応するオブジェクトの形で、グラフに挿入される。グラフの第 2 のバージョン 8 2 8 は、修正されたグラフによって視覚的に表すことができる。挿入オブジェクトは、グラフ 5 0 0 の第 2 のバージョンに示してもよい。

10

【0095】

開発環境 8 1 8 は、グラフの第 2 のバージョン 8 2 8 を実行するための実行環境 8 3 0 を含むことができる。たとえば、グラフがコンパイラ 8 2 6 によってコンパイルされると、グラフの第 2 のバージョン 8 2 8 を実行することができる。グラフの第 2 のバージョン 8 2 8 を実行することは、データ（たとえば、作業要素又はデータ・レコード）が各構成要素間を流れるときに、各構成要素、挿入（たとえば、テスト・ソース、プローブ、置換構成要素、又はこれらのうちの任意の 2 つ以上の組合せ）、及びグラフの第 2 のバージョン 8 2 8 の有向フローに関連する計算を実行することを含むことができる。例によっては、実行環境 8 3 0 は、コード・リポジトリ 8 2 2 に記憶された第 1 のバージョンのグラフのソース・コード、又はオーバーレイ・リポジトリ 8 2 4 に記憶されたソース・コードを修正することなく、グラフの第 2 のバージョン 8 2 8 を実行する。実行環境 8 3 0 は、開発環境 8 1 8 のインターフェースを介してアクセス可能でもよく、又はそれ自体のインターフェースをもっているもよい。このインターフェースは、実行に関連する情報を表示するように構成することができる。このインターフェースは、挿入に関連する情報（たとえば、プローブによって監視及び保存されているデータ、テスト・ソースによって挿入されているデータ、置換構成要素についての情報、又は他の情報）を表示するように構成することもできる。実行環境 8 3 0 により、開発者 8 2 0 は、各実行間で、グラフの第 2 のバージョン 8 2 8 を複数回実行できるようになり、グラフの第 2 のバージョン 8 2 8 の各態様を修正できるようになってもよい。

20

30

【0096】

例によっては、開発者は、グラフの挿入及びコンパイルを管理する。たとえば、開発者 8 2 0 は、コード・リポジトリ 8 2 2 から、図 1 のグラフ 1 0 0 の第 1 のバージョンを選択する。開発者 8 2 0 は又、オーバーレイ・リポジトリ 8 2 4 から、図 4 のオーバーレイ指定 2 0 0 を選択する。例によっては、オーバーレイ指定 2 0 0 を選択する代わりに、開発者 8 2 0 は、オーバーレイ・リポジトリ 8 2 4 内の様々なオーバーレイ指定から挿入定義を選択してもよい。開発者 8 2 0 は、グラフ 1 0 0 の第 1 のバージョン及びオーバーレイ指定 2 0 0 に基づいて、グラフの第 2 のバージョン 8 2 8 をコンパイルするようコンパイラ 8 2 6 に命令する。

【0097】

例によっては、挿入は自動的に挿入される。たとえば、前述の通り、グラフ 1 0 0 内の 1 つ又は複数のデータ・ソース、出力データ受信側、又はフォーマット特有の構成要素は、たとえば、入力接続又は出力接続がない構成要素を識別することによって、又はグラフ 1 0 0 内の構成要素の仕様を解析することによって自動的に識別される。識別されたデータ・ソース及び出力データ受信側は、グラフ 1 0 0 のデバッグ中に挿入によって置換されることになる、データ・ソース及び出力データ受信側のリストと自動的に比較することができる。たとえば、このリストは、開発者 8 2 0 が提供することができる。フォーマット特有の構成要素を解析して、入力されるデータのセットのフォーマットなど、特定のフォーマットのデータをこの構成要素が処理できるかどうかを判定することができる。データを処理できないフォーマット特有の構成要素のリストが生成される。例によっては、

40

50

このリストは、開発者 820 が提供することができる。このリストに従って、グラフ 100 のデータ・ソース、出力データ受信側、又はフォーマット特有の構成要素用のオーバーレイ指定が自動的に作成される。次いで、グラフの第 2 のバージョンが自動的にコンパイルされる。

【0098】

例によっては、オーバーレイ指定は、コード・リポジトリ 822 又はオーバーレイ・リポジトリ 824 内のファイルとして永続的に記憶されることはない。むしろ、通常オーバーレイ・ファイルに含まれるはずの情報（たとえば、挿入定義）は、（たとえば、ユーザ・インターフェースを介して）開発者 820 によって作成され、又は解析エンジン 300 及び挿入エンジン 306 によって自動的に判定され、メモリに一時的に記憶される。次いで、オーバーレイ情報は、コンパイラ（たとえば、図 8 の 608）又は保存状態マネージャ（たとえば、図 9 の 708）に渡される。

【0099】

図 10 を参照すると、例示的なプロセスでは、グラフの第 1 のバージョン（たとえば、図 1 のグラフ 100）が受信される（902）。たとえば、グラフの第 1 のバージョンは、プロセッサがアクセス可能な短期記憶に受信することができる。グラフ 100 の第 1 のバージョンは、構成要素及びフローを含む。この構成要素は、データ・レコードに実行される動作を表し、このフローは、各構成要素間のデータ・レコードのフローを表す。

【0100】

1 つ又は複数の挿入を定義するオーバーレイ指定が受信される（904）。例によっては、オーバーレイ指定は、開発者又は試験者から受信される。例によっては、たとえば前述の通り、オーバーレイ指定は自動的に定義される。オーバーレイ指定は、図 4 に示すオーバーレイ指定 200 でもよい。オーバーレイ指定は、1 つ又は複数の挿入定義（たとえば、1 つ若しくは複数のテスト・ソース定義、1 つ若しくは複数のプローブ定義、又は 1 つ若しくは複数の置換構成要素定義）を含むことができる。挿入定義は、名前、上流ポート、下流ポート、挿入タイプ、プロトタイプ経路、及び（テスト・ソース定義用の）レイアウト・パラメータを含むことができる。定義されたテスト・ソース及びプローブのそれぞれは、グラフ 100 のフローに関連付けることができる。定義された置換構成要素のそれぞれは、グラフ 100 の構成要素に関連付けることができる。

【0101】

定義された挿入のうちの 1 つにそれぞれが対応する、1 つ又は複数のオブジェクトが生成される（906）。オブジェクトは、テスト・ソース、プローブ、又は置換構成要素など、グラフの構成要素でもよい。

【0102】

グラフ 100 の一部分の構成要素及びフローの少なくともいくつか、並びに生成された 1 つ又は複数のオブジェクトを含む、グラフの少なくとも一部分の第 2 のバージョンが生成される（908）。例によっては、グラフの第 2 のバージョンは、グラフ 100 の一部分の構成要素及びフローの少なくともいくつか、並びに生成された 1 つ又は複数のオブジェクトを含むように修正された、元のグラフ 100 のコピーである。グラフの第 2 のバージョンは、修正されたグラフ（たとえば、図 2 のグラフ 200 の第 2 のバージョン、又は図 3 のグラフ 300 の第 3 のバージョン）によって視覚的に表すことができる。各オブジェクトは、（テスト・ソース若しくはプローブ用の）オブジェクトに対応する定義済みの挿入に関連付けられたフローにおいて、又はこのオブジェクトに対応する定義済みの置換構成要素に関連付けられた構成要素の代わりに挿入される。生成された挿入オブジェクトは、グラフ 100 のデータ処理構成要素とともに、グラフの第 2 のバージョンに表示してもよいが、グラフ 100 の第 1 のバージョン（又は、グラフ 100 の第 1 のバージョンを含むファイル）は修正されない。

【0103】

グラフ及びオーバーレイ指定をコンパイルして、オーバーレイ・ファイルによって定義された挿入を含むグラフの第 2 のバージョンを作成できるコンパイラ（たとえば、図 7 のコン

10

20

30

40

50

パイラ608、及び図8のコンパイラ712)を説明したが、実施形態によっては、グラフ及びオーバーレイ指定はコンパイルされない。たとえば、グラフ及びオーバーレイ指定は、コンパイルされることなく直接実行することができる。インタプリタは、それぞれのステートメントを、既に機械コードにコンパイルされている一連の1つ又は複数のサブルーチンに変換することによって、グラフ及びオーバーレイ指定を直接実行することができる。

【0104】

プローブ、テスト・ソース、及び置換構成要素の形で挿入を説明してきたが、実施形態によっては、挿入は他の形をとることができる。挿入を広く使用して、グラフの所与のポイントでデータを入力し、グラフの所与のポイントからデータを取り出すことができる。たとえば、グラフのフローを通過するデータの品質を監視するように、挿入を設計することができる。データ品質が閾値を下回る場合、ユーザは、自動警告を受信することができる。挿入のさらなる説明は、米国特許出願第14/715,904号に見いだすことができ、この内容全体を参考として本明細書に援用する。

【0105】

さらに、グラフとの関連で挿入を説明してきたが、実施形態によっては、挿入は他の実行可能なアプリケーションとともに使用することができる。たとえば、包括的で実行可能なアプリケーション用のデータ・ソース、出力データ受信側、又はフォーマット特有のプロセスは、このアプリケーションの自動解析を用いて識別することができる。識別されたデータ・ソース、出力データ受信側、又はフォーマット特有のプロセスのうちの1つ又は複数は、それぞれ適切なテスト・ソース、プローブ、又は置換プロセスによって置換することができる。このようにして、実行可能なアプリケーションは、テスト・ソースからのデータを処理し、データをプローブに出力することができ、又は互いに異なるフォーマットのデータを処理できるようにすることができる。この構成は、実行可能なアプリケーションをテスト又はデバッグするのに役立つことがある。

【0106】

前述の手法は、適切なソフトウェアを実行するコンピューティング・システムを使用して実施することができる。たとえば、ソフトウェアは、(分散アーキテクチャ、クライアント/サーバ、又はグリッドなど様々なアーキテクチャでよい)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ティング・システム上で実行される、1つ又は複数のコンピュータ・プログラムでの手順を含んでもよく、このコンピューティング・システムのそれぞれが、少なくとも1つのプロセッサ、(揮発性メモリ及び/又は不揮発性メモリ、及び/又は記憶素子を含む)少なくとも1つのデータ記憶装置システム、(少なくとも1つの入力装置若しくは入力ポートを使用して入力を受信し、少なくとも1つの出力装置若しくは出力ポートを使用して出力を供給するための)少なくとも1つのユーザ・インターフェースを備える。ソフトウェアは、たとえば、グラフの設計、構成、及び実行に関連するサービスを提供する、比較的大きいプログラムの1つ又は複数のモジュールを含んでもよい。プログラムの各モジュール(たとえばグラフの要素)は、データ構造として、又はデータ・リポジトリに記憶されたデータ・モデルに合致する他の編成データとして実装することができる。

【0107】

ソフトウェアは、(たとえば、汎用若しくは専用の、コンピューティング・システム若しくはコンピューティング装置によって読取り可能な)CD-ROM又は他のコンピュータ読取り可能な媒体など、有形で持続的な媒体上に実装してもよく、又はネットワークの通信媒体を介して、ソフトウェアが実行されるコンピューティング・システムの有形で持続的な媒体に送達してもよい(たとえば、伝搬信号において符号化してもよい)。処理の一部又は全ては、専用コンピュータ上で実行してもよく、又は、コプロセッサ若しくはフィールド・プログラマブル・ゲート・アレイ(FPGA)、若しくは専用の特定用途向け集積回路(ASIC)などの専用ハードウェアを使用して実行してもよい。ソフトウェアによって指定された計算の様々な部分を様々なコンピューティング要素によって実行する分散方式で、処理を実施してもよい。本明細書に記載の処理を実行するよう、記憶装置媒

体がコンピュータに読み取られるときにコンピュータを構成し動作させるために、このようなコンピュータ・プログラムはそれぞれ、汎用又は専用のプログラム可能なコンピュータでアクセス可能な記憶装置のコンピュータ読取り可能な記憶媒体（たとえば、固体記憶装置若しくは固体記憶媒体、又は磁気媒体若しくは光媒体）に、記憶又はダウンロードされることが好ましい。この発明性のあるシステムは又、コンピュータ・プログラムで構成された有形の持続的な媒体として実施されるものとみなしてもよく、ここで、そのように構成された媒体により、コンピュータは、特定の事前定義された方式で動作して、本明細書に記載の処理ステップのうちの1つ又は複数を実行できるようになる。

【0108】

いくつかの実施形態を説明してきた。それにもかかわらず、前述の説明は、本発明の範囲を例示するものであり、それを限定するものではなく、本発明の範囲は、添付の特許請求の範囲に記載の範囲によって定義されることを理解されたい。したがって、他の実施形態も、添付の特許請求の範囲に記載の範囲内にある。たとえば、本発明の範囲から逸脱することなく、様々な修正を加えてもよい。さらに、前述のステップのいくつかは、順序に依存しなくてもよく、したがって、記載された順序とは異なる順序で実行することができる。

10

20

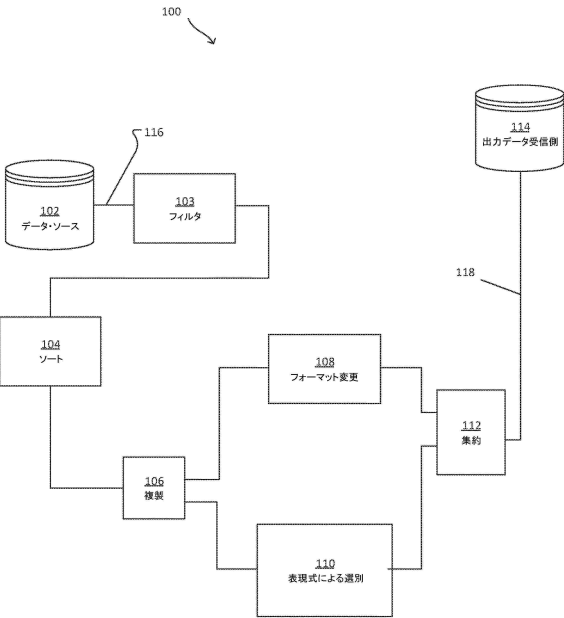
30

40

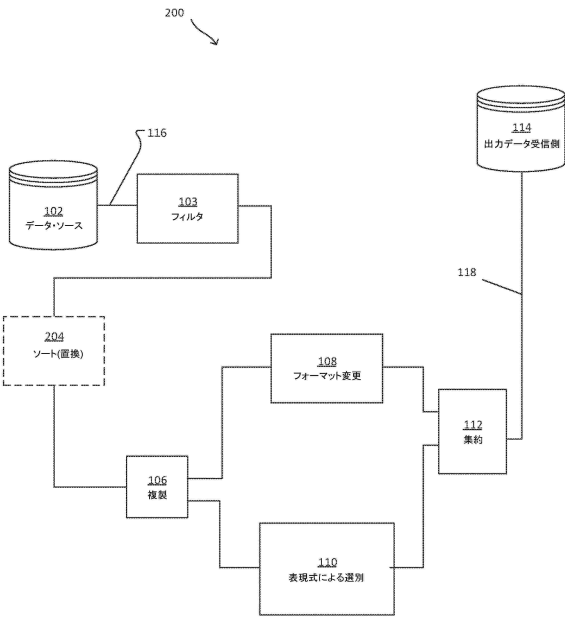
50

【図面】

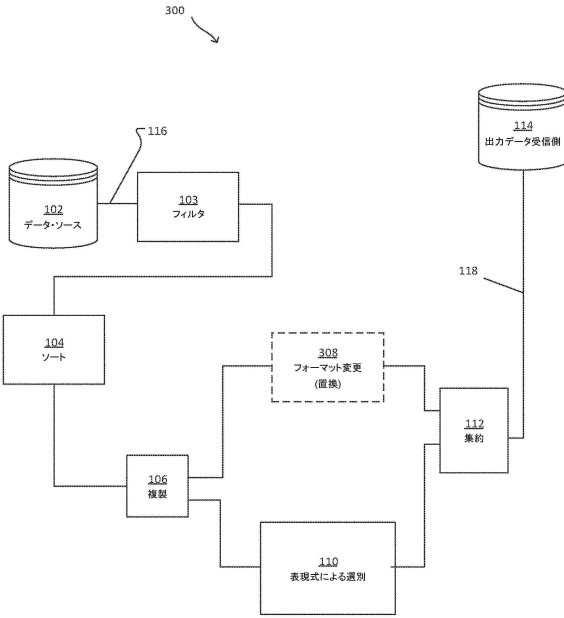
【図 1】



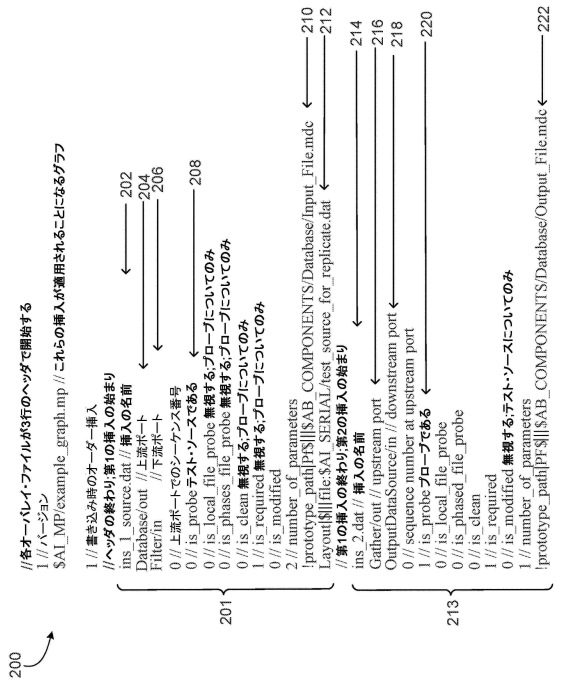
【図 2】



【図 3】



【図 4】



10

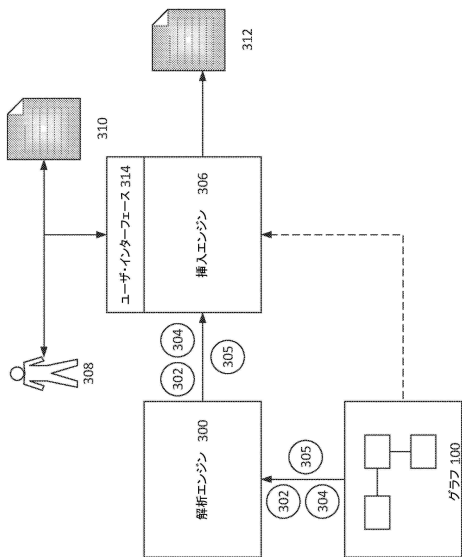
20

30

40

50

【図 5】



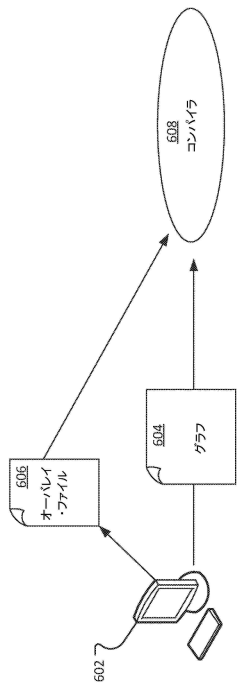
【図 6】



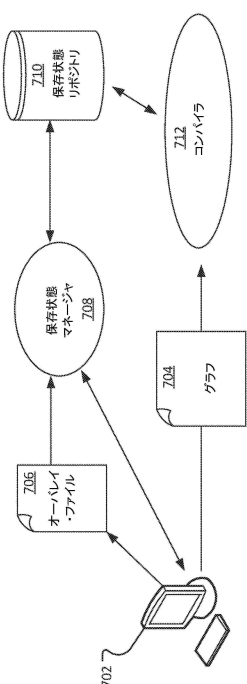
10

20

【図 7】



【図 8】

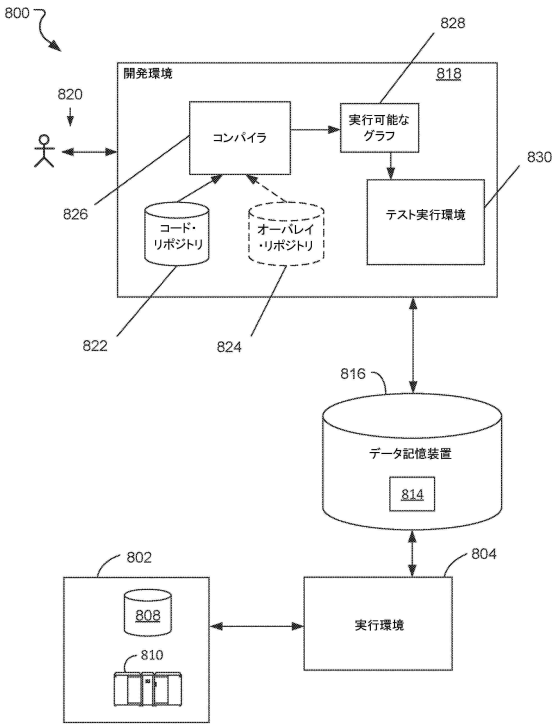


30

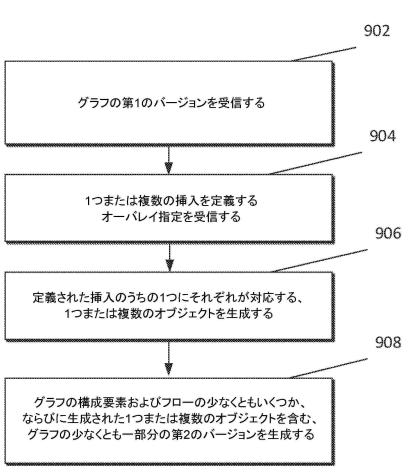
40

50

【図 9】



【図 10】



10

20

30

40

50

フロントページの続き

- (31)優先権主張番号 15/433,467
(32)優先日 平成29年2月15日(2017.2.15)
(33)優先権主張国・地域又は機関
米国(US)
02158, ニュートン, バレースプリング ロード 11
(72)発明者 ジョイス, ジョン
アメリカ合衆国, マサチューセッツ州 02460, ニュートン, ウォルナット ストリート 503
合議体
審判長 林 毅
審判官 大塚 俊範
審判官 吉田 美彦
(56)参考文献 特開2006-4099 (JP, A)
(58)調査した分野 (Int.Cl., DB名)
G06F 8/30