

(12) 特許協力条約に基づいて公開された国際出願

(19) 世界知的所有権機関
国際事務局

(43) 国際公開日
2018年12月27日(27.12.2018)



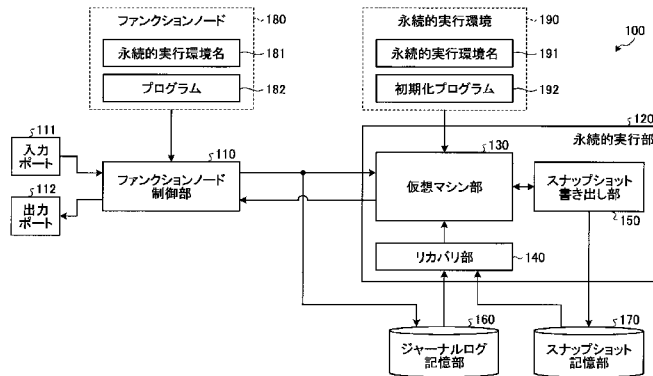
(10) 国際公開番号

WO 2018/235309 A1

- (51) 国際特許分類:
G06F 9/445 (2006.01) G06F 9/44 (2006.01)
- (21) 国際出願番号: PCT/JP2017/034647
- (22) 国際出願日: 2017年9月26日(26.09.2017)
- (25) 国際出願の言語: 日本語
- (26) 国際公開の言語: 日本語
- (30) 優先権データ:
特願 2017-121052 2017年6月21日(21.06.2017) JP
- (71) 出願人: 株式会社東芝 (KABUSHIKI KAISHA TOSHIBA) [JP/JP]; 〒1058001 東京都港区芝浦一丁目1番1号 Tokyo (JP). 東芝デジタルソリューションズ株式会社 (TOSHIBA DIGITAL SOLUTIONS CORPORATION) [JP/JP]; 〒2128585 神奈川県川崎市幸区堀川町72番地34 Kanagawa (JP).
- (72) 発明者: 遠藤 浩太郎 (ENDO, Kotaro); 〒2128585 神奈川県川崎市幸区堀川町72番地34 東芝デジタルソリューションズ株式会社内 Kanagawa (JP). 中嶋 宏 (NAKAJIMA, Hiroshi); 〒2128585 神奈川県川崎市幸区堀川町72番地34 東芝デジタルソリューションズ株式会社内 Kanagawa (JP).
- (74) 代理人: 特許業務法人酒井国際特許事務所 (SAKAI INTERNATIONAL PATENT OFFICE); 〒1000013 東京都千代田区霞が関3丁目8番1号 虎の門三井ビルディング Tokyo (JP).
- (81) 指定国(表示のない限り、全ての種類の国内保護が可能): AE, AG, AL, AM, AO, AT, AU, AZ,

(54) Title: INFORMATION PROCESSING DEVICE, INFORMATION PROCESSING SYSTEM, INFORMATION PROCESSING METHOD, AND PROGRAM

(54) 発明の名称: 情報処理装置、情報処理システム、情報処理方法およびプログラム



- 110 Function node control unit
111 Input port
112 Output port
120 Persistent execution unit
130 Virtual machine unit
140 Recovery unit
150 Snapshot export unit
160 Journal log storage unit
170 Snapshot storage unit
180 Function node
181, 191 Persistent execution environment name
182 Program
190 Persistent execution environment
192 Initialization program

(57) Abstract: A function node control unit (110) records combined sets of input messages and programs as journal logs, and sends the input messages and programs to a virtual machine unit (130). The virtual machine unit (130) executes the programs with the input messages as arguments, and sends the results of the executions to the function node control unit (110). The function node control unit (110) outputs the results of the executions as output messages. Upon an information processing device (100) being launched, a recovery unit (140) restores the virtual machine unit (130) to a state at a given previous time, sequentially extracts the journal logs recorded after the given previous time, causes the virtual machine unit (130) to sequentially carry out a re-execution of the programs using as arguments the input messages constituting the journal logs, and rolls forward the virtual machine unit (130) to the most recent state.

BA, BB, BG, BH, BN, BR, BW, BY, BZ, CA, CH, CL, CN, CO, CR, CU, CZ, DE, DJ, DK, DM, DO, DZ, EC, EE, EG, ES, FI, GB, GD, GE, GH, GM, GT, HN, HR, HU, ID, IL, IN, IR, IS, JO, KE, KG, KH, KN, KP, KR, KW, KZ, LA, LC, LK, LR, LS, LU, LY, MA, MD, ME, MG, MK, MN, MW, MX, MY, MZ, NA, NG, NI, NO, NZ, OM, PA, PE, PG, PH, PL, PT, QA, RO, RS, RU, RW, SA, SC, SD, SE, SG, SK, SL, SM, ST, SV, SY, TH, TJ, TM, TN, TR, TT, TZ, UA, UG, US, UZ, VC, VN, ZA, ZM, ZW.

- (84) 指定国(表示のない限り、全ての種類の広域保護が可能): ARIPO (BW, GH, GM, KE, LR, LS, MW, MZ, NA, RW, SD, SL, ST, SZ, TZ, UG, ZM, ZW), ユーラシア (AM, AZ, BY, KG, KZ, RU, TJ, TM), ヨーロッパ (AL, AT, BE, BG, CH, CY, CZ, DE, DK, EE, ES, FI, FR, GB, GR, HR, HU, IE, IS, IT, LT, LU, LV, MC, MK, MT, NL, NO, PL, PT, RO, RS, SE, SI, SK, SM, TR), OAPI (BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, KM, ML, MR, NE, SN, TD, TG).

添付公開書類:

- 一 国際調査報告(条約第21条(3))

(57) 要約: ファンクションノード制御部(110)は、入力メッセージとプログラムとの組をジャーナルログとして記録するとともに、入力メッセージとプログラムとを仮想マシン部(130)に送る。仮想マシン部(130)は、入力メッセージを引数としてプログラムを実行し、実行結果をファンクションノード制御部(110)に送る。ファンクションノード制御部(110)は、実行結果を出力メッセージとして出力する。リカバリ部(140)は、情報処理装置(100)の起動時に、仮想マシン部(130)を過去のある時刻の状態に復元した後、該時刻以降に記録されたジャーナルログを順に取り出して、該ジャーナルログを構成する入力メッセージを引数としたプログラムの再実行を仮想マシン部(130)において順次行わせ、仮想マシン部(130)を最新の状態にロールフォワードする。

明 細 書

発明の名称：

情報処理装置、情報処理システム、情報処理方法およびプログラム

技術分野

[0001] 本発明の実施形態は、情報処理装置、情報処理システム、情報処理方法およびプログラムに関する。

背景技術

[0002] 例えばＩｏＴ機器の動作を設計・実装するための手法として、フローベースプログラミング（Flow-Based Programming；以下、「ＦＢＰ」と呼ぶ）が広く利用されている。ＦＢＰは、メッセージドリブンのソフトウェア部品（「ノード」と呼ばれる）をつなげて処理の流れ（フロー）を作成するプログラム開発手法であり、例えばＮｏｄｅ－ＲＥＤなどのオープンソースのツールが提供されている。

[0003] 永続性が必要とされる機器のプログラムをＦＢＰにより作成する場合、ＦＢＰの実行環境自体には永続性がないため、機器の状態を外部のデータベースに保存する必要がある。また、機器の起動時（障害からのリカバリを含む）や、機器の永続的な状態を利用する場合は、外部のデータベースに保存しておいた状態を読み込んで、機器の状態を復元する必要がある。このため、例えばデータベースに接続して検索・更新等を行うデータベースノードと呼ばれる特殊なノードを追加するなど、本来のデータフローに対して永続化のための技巧的なフローを追加する必要がある、プログラム開発の負担になっている。

先行技術文献

特許文献

[0004] 特許文献１：特開２０１６－２４７９４号公報

発明の概要

発明が解決しようとする課題

[0005] 本発明が解決しようとする課題は、F B Pの実行環境において永続性を実現できる情報処理装置、情報処理システム、情報処理方法およびプログラムを提供することである。

課題を解決するための手段

[0006] 実施形態の情報処理装置は、フローベースプログラミングの実行環境となる情報処理装置であって、プログラミング環境からデプロイされたファンクションノードに対応して設けられるファンクションノード制御部と、前記ファンクションノードに関連付けられて前記プログラミング環境からデプロイされた永続的実行環境に対応して設けられ、仮想マシン部とリカバリ部とを有する永続的実行部と、を備える。前記ファンクションノード制御部は、前記ファンクションノードの入力メッセージが入力されると、該入力メッセージと前記ファンクションノードに定義されたプログラムとの組をジャーナルログとして記録するとともに、前記入力メッセージと前記プログラムとを前記仮想マシン部に送る。前記仮想マシン部は、前記入力メッセージを引数として前記プログラムを実行し、実行結果を前記ファンクションノード制御部に送る。前記ファンクションノード制御部は、前記仮想マシン部から受け取った前記実行結果を出力メッセージとして出力する。前記リカバリ部は、前記情報処理装置の起動時に、前記仮想マシン部を過去のある時刻の状態に復元した後、該時刻以降に記録された前記ジャーナルログを順に取り出して、該ジャーナルログを構成する前記入力メッセージを引数とした該ジャーナルログを構成する前記プログラムの再実行を前記仮想マシン部において順次行わせることにより、前記仮想マシン部を最新の状態にロールフォワードする。

図面の簡単な説明

[0007] [図1]図1は、フロー作成UIの一例を示す図である。

[図2]図2は、ファンクションノード設定UIの一例を示す図である。

[図3]図3は、永続的実行環境設定UIの一例を示す図である。

[図4]図4は、フロー作成UIの一例を示す図である。

[図5]図5は、第1実施形態に係る情報処理装置の機能的な構成例を示すブロック図である。

[図6]図6は、ファンクションノード制御部の処理手順の一例を示すフローチャートである。

[図7]図7は、仮想マシン部の詳細を示すブロック図である。

[図8]図8は、仮想マシン部におけるスコープルールを説明する図である。

[図9]図9は、変数名解決部の処理手順の一例を示すフローチャートである。

[図10]図10は、リカバリ部の処理手順の一例を示すフローチャートである。

[図11]図11は、ファンクションノード制御部の処理手順の一例を示すフローチャートである。

[図12]図12は、プログラムIDとコンパイルされたプログラムとの対応表の一例を示す図である。

[図13]図13は、ファンクションノード制御部の処理手順の一例を示すフローチャートである。

[図14]図14は、仮想マシン部の処理手順の一例を示すフローチャートである。

[図15]図15は、リカバリ部の処理手順の一例を示すフローチャートである。

[図16]図16は、第2実施形態に係る情報処理装置の機能的な構成例を示すブロック図である。

[図17]図17は、仮想マシン部の処理手順の一例を示すフローチャートである。

[図18]図18は、イベントリスナノード制御部の処理手順の一例を示すフローチャートである。

[図19]図19は、第3実施形態に係る情報処理装置の機能的な構成例を示すブロック図である。

[図20]図20は、第4実施形態に係る情報処理装置の機能的な構成例を示す

ブロック図である。

[図21]図 2 1 は、第 5 実施形態に係る情報処理システムの概略構成を示すブロック図である。

[図22]図 2 2 は、第 5 実施形態に係る情報処理装置の機能的な構成例を示すブロック図である。

[図23]図 2 3 は、再現装置の処理手順の一例を示すフローチャートである。

[図24]図 2 4 は、変形例に係る情報処理装置の機能的な構成例を示すブロック図である。

[図25]図 2 5 は、第 6 実施形態に係る情報処理システムの概略構成を示すブロック図である。

[図26]図 2 6 は、第 7 実施形態に係る情報処理システムの概略構成を示すブロック図である。

[図27]図 2 7 は、情報処理装置のハードウェア構成の一例を示すブロック図である。

発明を実施するための形態

[0008] 以下で示す実施形態では、本発明が適用される情報処理装置の一例として、ゲートウェイと呼ばれる I O T 機器を想定する。ゲートウェイは、周辺のセンサ等からデータを収集し、そのデータを適宜に加工した後にサーバ装置に転送したり、サーバ装置からの操作命令に基づき、周辺の I O T 機器を制御したりといった役割を持つ。

[0009] ゲートウェイは、汎用のコンピュータとしてのハードウェア資源を備え、現場の状況に合わせてソフトウェア的にカスタマイズされる。そのカスタマイズのために F B P が利用される。F B P を利用して様々な周辺機器との接続の状況をフローとして表現することにより、視覚的にわかりやすいカスタマイズが可能である。ゲートウェイ内でのデータ加工処理等の内容は、ファンクションノードと呼ばれる処理ノードを使って定義する。

[0010] 処理ノードは、F B P においてフローを構成するソフトウェア部品であり、一般に、1つの入力ポートと複数の出力ポートをもつ。出力ポートは他の

処理ノードの入力ポートにつながる。処理ノードのうち、JavaScript（登録商標）のプログラムをユーザ定義可能なソフトウェア部品がファンクションノードである。ファンクションノードを使ってゲートウェイ内でのデータ加工処理等の内容を定義することにより、ゲートウェイの動作を任意にカスタマイズすることができる。

[0011] 一般的なF B Pの実行環境は、永続性を持たない。そこで本実施形態では、永続的実行環境とよばれるJavaScriptの仮想マシンを内蔵した機能をF B Pに組み込むことにより、F B Pの実行環境において永続性を実現する。ここで仮想マシンとは、独立したJavaScript等の実行環境であり、固有のグローバルスコープを持ち、その中で実行されるプログラムの暗黙的な作用は外部に影響しないという性質を持つ。永続的実行環境は処理ノードではなく、ファンクションノードが実行される環境としてファンクションノードに関連付けることによって現れるものである。

[0012] ここで、本実施形態のF B Pのプログラミング環境におけるフロー作成の手順について説明する。本実施形態のF B Pによるフローは、例えばブラウザベースのUI上でのユーザ操作に応じて作成される。本実施形態のF B Pにおけるフロー作成UI 10の一例を図1に示す。このフロー作成UI 10は、ノード一覧領域11とフロー作成領域12とを有し、ノード一覧領域11に用意されたノードのアイコンを例えばドラッグ&ドロップ操作によりフロー作成領域12に配置し、フロー作成領域12に配置した複数のノードのアイコンをつなげることで所望のフローを作成できる構成となっている。ここでノード一覧領域11には、永続的実行環境との関連付けが可能なファンクションノードのアイコン13が用意され、このファンクションノードを用いてフローを作成できる。

[0013] フロー作成UI 10のフロー作成領域12上で、フローを構成するファンクションノードのアイコン13に対して例えばダブルクリックなどの操作を行うと、例えば図2に示すようなファンクションノード設定UI 20が表示される。このファンクションノード設定UI 20は、ファンクションノード

のノード名を書き込むテキストボックス 21 と、永続的実行環境を他のファンクションノードと共有する場合にチェックするチェックボックス 22 と、チェックボックス 22 がチェックされた場合に他のファンクションノードと共有する永続的実行環境を選択するためのプルダウンメニュー 23 と、永続的実行環境を新たに設定する場合に操作する「設定」ボタン 24 と、ファンクションノードのプログラムを記述するプログラム記述領域 25 とを有する。ユーザはこのファンクションノード設定 UI 20 上でファンクションノードのノード名を定義し、ファンクションノードと関連付ける永続的実行環境を指定し、ファンクションノードのプログラムを定義することで、上述のフローを構成するファンクションノードを設定できる。

[0014] このファンクションノード設定 UI 20 の「設定」ボタン 24 を押すと、例えば図 3 に示すような永続的実行環境設定 UI 30 が表示される。この永続的実行環境設定 UI 30 は、永続的実行環境の名前を書き込むテキストボックス 31 と、永続的実行環境の初期化プログラムを記述するプログラム記述領域 32 とを有する。ユーザはこの永続的実行環境設定 UI 30 上で永続的実行環境の名前と初期化プログラムを定義することで、ファンクションノードが使用する永続的実行環境を新たに設定できる。

[0015] 上述のフローを構成するファンクションノードの設定が終了すると、図 4 に示すようなフロー作成 UI 10 が再度表示され、フローを構成するファンクションノードのアイコン 13 にファンクションノードのノード名と、そのファンクションノードに関連付く永続的実行環境の名前が反映される。また、このフロー作成 UI 10 上で「デプロイ」ボタン 15 が表示され、この「デプロイ」ボタン 15 が押されると、上述のフローを構成するファンクションノードと、このファンクションノードに関連付く永続的実行環境が、指定されたデプロイ先にデプロイされる。

[0016] 本実施形態に係る情報処理装置は、FBP の実行環境となる情報処理装置であって、デプロイされたファンクションノードのプログラムを永続的実行環境で実行するものである。以下では、本実施形態に係る情報処理装置、情

報処理システム、情報処理方法およびプログラムの詳細を説明する。なお、以下の説明において、同様の機能を持つ構成要素については同一の符号を付して、重複した説明を適宜省略する。

[0017] <第1実施形態>

図5は、第1実施形態に係る情報処理装置100の機能的な構成例を示すブロック図である。この情報処理装置100は、1つ以上のファンクションノード制御部110と、1つ以上の永続的実行部120とを備える。永続的実行部120は、仮想マシン部130、リカバリ部140およびスナップショット書き出し部150を有する。また、情報処理装置100は、ジャーナルログ記憶部160と、スナップショット記憶部170とを備える。これらジャーナルログ記憶部160およびスナップショット記憶部170は、永続的実行部120ごとに設けられていてもよいし、複数の永続的実行部120が設けられた場合にこれらで共有される構成であってもよい。

[0018] ファンクションノード制御部110は、FBPのプログラミング環境から情報処理装置100にデPLOYされたファンクションノード180に対応して設けられ、永続的実行部120は、FBPのプログラミング環境から情報処理装置100にデPLOYされた永続的実行環境190に対応して設けられる。図5では、1つのファンクションノード180に対応する1つのファンクションノード制御部110と、1つの永続的実行環境190に対応する1つの永続的実行部120のみを図示しているが、ファンクションノード制御部110はデPLOYされたファンクションノード180ごとに設けられる。デPLOYされたファンクションノード180が複数あれば、各々のファンクションノード180に対応する複数のファンクションノード制御部110が設けられる。また、永続的実行部120は、ファンクションノード180に関連付けられてデPLOYされた永続的実行環境190の数だけ設けられる。上述したように、複数のファンクションノード180で共有される永続的実行環境190に対応する永続的実行部120は、永続的実行環境190を共有する複数のファンクションノード180に対応する複数のファンクション

ノード制御部 110 によって使用される。また、1つのファンクションノード 180 が占有する永続的実行環境 190 に対応する永続的実行部 120 は、そのファンクションノード 180 に対応するファンクションノード制御部 110 のみにより使用される。

[0019] FBP のプログラミング環境からデプロイされたファンクションノード 180 は、ユーザ定義された永続的実行環境名 181 およびプログラム 182 を含む。また、FBP のプログラミング環境からデプロイされた永続的実行環境 190 は、ユーザ定義された永続的実行環境名 191 および初期化プログラム 192 を含む。本実施形態では、ファンクションノード 180 と永続的実行環境 190 との関連付けが、永続的実行環境 190 の名前を用いて実現される。すなわち、ファンクションノード 180 とこれに関連付く永続的実行環境 190 では、双方でユーザ定義された永続的実行環境名 181, 191 が一致する。

[0020] ファンクションノード制御部 110 は、永続的実行部 120 を使用してファンクションノード 180 の機能を実現するための制御部である。ファンクションノード制御部 110 の処理手順の一例を図 6 に示す。この図 6 のフローチャートで示す一連の処理は、ファンクションノード 180 の入力メッセージが入力ポート 111 に入力されるたびに、ファンクションノード制御部 110 によって繰り返し実行される。

[0021] ファンクションノード 180 の入力メッセージが入力ポート 111 に入力されると、ファンクションノード制御部 110 は、まず、ファンクションノード 180 に定義された永続的実行環境名 181 をキーとして、ファンクションノード 180 に関連付いた永続的実行環境 190 を特定する（ステップ S101）。そして、ファンクションノード制御部 110 は、入力ポート 111 から入力された入力メッセージと、ファンクションノードに定義されたプログラム 182 との組をジャーナルログとして、ジャーナルログ記憶部 160 に記録する（ステップ S102）。さらに、ファンクションノード制御部 110 は、これら入力メッセージとプログラム 182 とを、ステップ S1

01で特定した永続的実行環境190に対応する永続的実行部120の仮想マシン部130に送る（ステップS103）。その後、仮想マシン部130からプログラム182の実行結果が送られてくると、ファンクションノード制御部110は、その実行結果を仮想マシン部130から受け取って、出力メッセージとして出力ポート112から出力する（ステップS104）。

[0022] 仮想マシン部130は、ファンクションノード制御部110から入力メッセージとプログラム182を受け取ると、入力メッセージを引数としてプログラム182を実行し、実行結果をファンクションノード制御部110に返す。図7は、仮想マシン部130の詳細を示すブロック図である。仮想マシン部130は、ファンクション準備部131と、解釈実行部137と、変数名解決部136とを備える。ファンクション準備部131は、スコープ生成部132と構文解析部133とを有する。

[0023] スコープ生成部132は、ファンクションノード制御部110から送られたプログラム182の実行に先立ち、ファンクションスコープ134を生成する。ファンクションスコープ134は、プログラム182の実行ごとに生成される変数記憶域のことであり、そのプログラム182の中からのみアクセスされる。ファンクションスコープ134は、プログラム182の実行が終了し、利用されなくなると消滅する。仮想マシン部130は、ファンクションノード制御部110から送られたプログラム182を構文解析部133で構文解析し、スコープ生成部132によって生成されたファンクションスコープ134に、プログラム182中で宣言された変数を登録する。また、仮想マシン部130は、ファンクションノード180の予約変数をファンクションスコープ134に登録する。例えば、ファンクションノード制御部110から送られた入力メッセージを保存する予約変数“msg”をファンクションスコープ134に登録し、同時にその値（つまり、送られてきた入力メッセージ）も登録する。

[0024] また、仮想マシン部130は、固有のグローバルスコープ135を持つ。グローバルスコープ135は、仮想マシン部130内に一つだけある変数記

憶域のことであり、あらゆるスコープからアクセス可能である。仮想マシン部130は、永続的実行環境190の初期化プログラム192をグローバルスコープ135で実行することにより、初期化プログラム192中で宣言された変数をグローバルスコープ135に登録する。また、仮想マシン部130は、永続的実行環境190の予約変数もグローバルスコープ135に登録する。なお、永続的実行環境190の初期化プログラム192が実行されるのは、永続的実行環境190がデプロイされた直後、ユーザが明示的に実行を指示した場合、情報処理装置100全体の初期化時に限定される。情報処理装置100の起動時には、永続的実行環境190の初期化プログラム192は実行されない。

[0025] 図8は、仮想マシン部130におけるスコープルールを説明する図である。ファンクションノード180のプログラム182を実行する際、ファンクションスコープ134に登録された変数（図中のx）については、ファンクションスコープ134でその値が参照される。一方、ファンクションノード180のプログラム182中で宣言されておらず、ファンクションスコープ134に登録されていない変数（図中のy）については、グローバルスコープ135でその値が参照される。また、ファンクションスコープに登録されていない関数（図中のg）についても、グローバルスコープ135が参照される。

[0026] 仮想マシン部130は、ファンクション準備部131によるプログラム182の実行準備が完了すると、ファンクションスコープ134を使って、解釈実行部137によりプログラム182を実行し、実行結果をファンクションノード制御部110に返す。このプログラム182を実行する際に変数へのアクセスがあった場合には、変数名解決部136が動作する。

[0027] 図9は、変数名解決部136の処理手順の一例を示すフローチャートである。変数名解決部136は、ファンクションノード180のプログラム182から変数が参照された場合、まず、その変数がプログラム182中で宣言された変数か否かを判断し（ステップS201）、プログラム182中で宣

言された変数であれば（ステップS201：Yes）、ファンクションスコープ134に登録されているその変数を参照して値を決定する（ステップS205）。一方、プログラム182中で宣言された変数でなければ（ステップS201：No）、変数名解決部136は、次に、その変数がファンクションノード180の予約変数か否かを判断し（ステップS202）、ファンクションノード180の予約変数であれば（ステップS202：Yes）、ファンクションスコープ134に登録されているその変数を参照して値を決定する（ステップS205）。

[0028] 一方、ファンクションノード180の予約変数でなければ（ステップS202：No）、変数名解決部136は、次に、その変数が永続的実行環境190の初期化プログラム192中で宣言された変数か否かを判断し（ステップS203）、初期化プログラム192中で宣言された変数であれば（ステップS203：Yes）、グローバルスコープ135に登録されているその変数を参照して値を決定する（ステップS205）。一方、初期化プログラム192中で宣言された変数でなければ（ステップS203：No）、変数名解決部136は、次に、その変数が永続的実行環境190の予約変数か否かを判断し（ステップS204）、永続的実行環境190の予約変数であれば（ステップS204：Yes）、グローバルスコープ135に登録されているその変数を参照して値を決定する（ステップS205）。なお、永続的実行環境190の予約変数でもない場合は（ステップS204：No）、参照エラーとなる（ステップS206）。

[0029] 仮想マシン部130の状態は、スナップショットとして定期的にスナップショット記憶部170に記録される。スナップショットは、ある時刻の仮想マシン部130内の全状態を表したバイナリデータであり、スナップショット書き出し部150により定期的に書き出され、スナップショット記憶部170に記録される。記録されたスナップショットは、後述のリカバリ部140が仮想マシン部130の状態を過去のある時刻の状態に復元するために利用される。

[0030] スナップショットはデータサイズが比較的大きいため、頻繁に書き出して記録することは望ましくない。そのため、スナップショット書き出し部150は、例えば、後述のロールフォワードに許容される時間が1秒程度であることを想定し、仮想マシン部130の累積の正味実行時間が1秒に達するたびに、スナップショットを書き出してスナップショット記憶部170に記録するものとする。

[0031] リカバリ部140は、情報処理装置100の電源が一旦オフした後の起動時（障害からの復旧時を含む）に、仮想マシン部130の状態を電源オフの前の状態（最新の状態）に復元する。リカバリ部140は、情報処理装置100の起動時に、スナップショット記憶部170に記録されたスナップショットを仮想マシン部130に展開することで仮想マシン部130を過去のある時刻の状態に復元した後、該時刻以降に記録されたジャーナルログをジャーナルログ記憶部160から順に取り出して、該ジャーナルログを構成する入力メッセージを引数としたプログラム182の再実行を仮想マシン部130において順次行わせることにより、仮想マシン部130を最新の状態にロールフォワードする。なお、再実行したプログラム182の実行結果は、ファンクションノード制御部110に送られることなく破棄される。

[0032] 図10は、リカバリ部140の処理手順の一例を示すフローチャートである。情報処理装置100の起動時に、リカバリ部140は、まず、最新のスナップショットをスナップショット記憶部170から読み出し、仮想マシン部130内に展開する（ステップS301）。そして、リカバリ部140は、最新のスナップショットの書き出し時刻よりも後に記録されたジャーナルログがあるかどうかを確認する（ステップS302）。そして、最新のスナップショットの書き出し時刻よりも後に記録されたジャーナルログがあれば（ステップS302：Yes）、ジャーナルログの読み出し位置を最新のスナップショット直後に設定する（ステップS303）。一方、最新のスナップショットの書き出し時刻よりも後に記録されたジャーナルログがなければ（ステップS302：No）、そのまま処理を終了する。

[0033] 次に、リカバリ部140は、ジャーナルログ記憶部160から、設定した読み出し位置のジャーナルログを読み出す（ステップS304）。そして、読み出したジャーナルログを構成する入力メッセージを引数として、その読み出したジャーナルログを構成するプログラム182を仮想マシン部130内で実行する（ステップS305）。その後、リカバリ部140は、ジャーナルログの読み出し位置を次のジャーナルログの記録時刻にシフトし（ステップS306）、読み出し位置のジャーナルログがあるか否かを確認する（ステップS307）。そして、読み出し位置のジャーナルログがあれば（ステップS307：Yes）、ステップS304に戻って以降の処理を繰り返す。一方、読み出し位置のジャーナルログがなければ（ステップS307：No）、処理を終了する。

[0034] なお、MRAM（Magnetoresistive Random Access Memory）等の不揮発メモリ技術を永続的実行部120に適用するならば、情報処理装置100が正常に電源オフした後に起動する際は、上述のリカバリ部140の処理を実施せずとも仮想マシン部130の状態を最新の状態とすることができる。しかし、不揮発メモリ技術を永続的実行部120に適用した場合であっても、障害からの復旧時には上述のリカバリ部140の処理を実施することで、アトミシティ（原子性）などの耐障害機能を実現することができる。また、上述のジャーナルログ記憶部160やスナップショット記憶部170にはストレージデバイスを利用することを想定しているが、MRAM等の不揮発メモリ技術を用いてこれらジャーナルログ記憶部160やスナップショット記憶部170を実現した場合は、処理速度の向上が見込まれる。

[0035] 以上、具体的な例を挙げながら詳細に説明したように、本実施形態によれば、FBPの実行環境となる情報処理装置100において永続性を実現することができる。したがって、これまでのように本来のデータフローに対して永続化のための技巧的なフローを追加する必要がなくなり、プログラム開発の効率向上を図ることができるとともに、プログラムの可読性を向上させることができる。また、永続的な状態がJavaScriptの変数としてアクセスでき

るため、プログラミングが容易になる。また、ファンクションノードの実行がアトミックになり、信頼性が向上する。

[0036] (変形例)

上述の第1実施形態の情報処理装置100は、様々なかたちでの最適化が可能である。つまり、構成等に若干の追加・変更を加えることにより、性能的な改善が実現される。以下では、プログラムに対応するプログラムIDの概念を導入することによる最適化の例を変形例として説明する。なお、以下で示す変形例以外にも、当業者が成し得るさまざまな既存技術による最適化が可能である。

[0037] プログラムIDは、ファンクションノード180のプログラム182に対して割り当てられる固有の識別情報であり、プログラム182を表すコードそのものよりもデータサイズが十分に小さい（例えば16バイト長）。そのため、プログラム182の代わりにプログラムIDを用いることにより、データサイズが小さくなり、容量の削減や転送時間の短縮などの効果がある。

[0038] 本実施形態では、情報処理装置100にファンクションノード180がデプロイされる際に、そのファンクションノード180にユーザ定義されたプログラム182に対応するプログラムIDが決定される。そして、ファンクションノード制御部110は、プログラムIDとプログラム182の組をジャーナルログとしてジャーナルログ記憶部160に記録するとともに、プログラムIDとプログラム182の組を永続的実行部120の仮想マシン部130に送る。仮想マシン部130に送られたプログラムIDとプログラム182の組は、仮想マシン部130内に登録される。このとき、仮想マシン部130は、プログラム182を構文解析し、内部形式に合せてコンパイルしたプログラム182をプログラムIDと対応付けて記憶しておくことが望ましい。なお、ファンクションノード180のプログラム182が変更される際は、変更されたプログラム182を含む新たなファンクションノード180がデプロイされるので、同様の処理が行われる。

[0039] また、本実施形態では、ファンクションノード180の入力メッセージが

入力されると、ファンクションノード制御部 110 が、その入力メッセージとプログラム 182 の本体との組ではなく、入力メッセージとプログラム ID との組をジャーナルログとしてジャーナルログ記憶部 160 に記録する。また、ファンクションノード制御部 110 は、入力メッセージとプログラム ID とを、永続的実行部 120 の仮想マシン部 130 に送る。仮想マシン部 130 は、ファンクションノード制御部 110 から送られた入力メッセージとプログラム ID を引数として後述する callScript を実行することにより、ファンクションノード 180 のプログラム 182 を実行し、実行結果をファンクションノード制御部 110 に返す。

[0040] また、リカバリ部 140 は、情報処理装置 100 の起動時に仮想マシン部 130 を最新の状態にロールフォワードする際に、入力メッセージとプログラム ID の組からなるジャーナルログをジャーナルログ記憶部 160 から読み出して、仮想マシン部 130 内で入力メッセージとプログラム ID を引数として callScript を実行させる。

[0041] 以下では、本変形例のファンクションノード制御部 110、仮想マシン部 130 およびリカバリ部 140 の処理手順を、フローチャートに沿って説明する。図 11 は、本変形例におけるファンクションノード制御部 110 の処理手順の一例を示すフローチャートであり、情報処理装置 100 にファンクションノード 180 がデプロイされたときに実行される処理の流れを示している。

[0042] 情報処理装置 100 にファンクションノード 180 がデプロイされると、ファンクションノード 180 に定義されたプログラム 182 に対応するプログラム ID が決定される（ステップ S401）。ファンクションノード制御部 110 は、まず、ファンクションノード 180 に定義された永続的実行環境名 181 をキーとして、ファンクションノード 180 に関連付いた永続的実行環境 190 を特定する（ステップ S402）。そして、ファンクションノード制御部 110 は、ステップ S401 で決定されたプログラム ID とプログラム 182 との組をジャーナルログとして、ジャーナルログ記憶部 16

0に記録する（ステップS403）。さらに、ファンクションノード制御部110は、これらプログラムIDとプログラム182とを、ステップS402で特定した永続的実行環境190に対応する永続的実行部120の仮想マシン部130に送る（ステップS404）。

[0043] 仮想マシン部130は、ファンクションノード制御部110からプログラムIDとファンクションノード180のプログラム182を受け取ると、上述の構文解析部133でプログラム182を構文解析し、プログラム182を仮想マシン部130の内部形式に合せて変換（コンパイル）する。コンパイルされたプログラム182は、例えば図12に示す対応表138のように、プログラムIDと対応付けられて仮想マシン部130の内部に登録される。

[0044] 図13は、本変形例におけるファンクションノード制御部110の処理手順の一例を示すフローチャートであり、ファンクションノード180の入力メッセージが入力されたときに実行される処理の流れを示している。ファンクションノード180の入力メッセージが入力ポート111に入力されると、ファンクションノード制御部110は、まず、ファンクションノード180に定義された永続的実行環境名181をキーとして、ファンクションノード180に関連付いた永続的実行環境190を特定する（ステップS501）。そして、ファンクションノード制御部110は、入力ポート111から入力された入力メッセージと、ファンクションノードに定義されたプログラム182に対応するプログラムIDとの組をジャーナルログとして、ジャーナルログ記憶部160に記録する（ステップS502）。さらに、ファンクションノード制御部110は、これら入力メッセージとプログラムIDとを、ステップS501で特定した永続的実行環境190に対応する永続的実行部120の仮想マシン部130に送る（ステップS503）。その後、仮想マシン部130からプログラム182の実行結果が送られてくると、ファンクションノード制御部110は、その実行結果を仮想マシン部130から受け取って、出力メッセージとして出力ポート112から出力する（ステップ

S 5 0 4)。

[0045] 図 1 4 は、本変形例における仮想マシン部 1 3 0 の処理手順の一例を示すフローチャートであり、ファンクションノード制御部 1 1 0 から入力メッセージとプログラム I D を受け取ったときの callScript の実行手順を示している。ファンクションノード制御部 1 1 0 から入力メッセージとプログラム I D を受け取ると、仮想マシン部 1 3 0 は、例えば図 1 2 に示した対応表 1 3 8 を参照し、ファンクションノード制御部 1 1 0 から受け取ったプログラム I D に対応するコンパイルされたプログラム 1 8 2 を取得する（ステップ S 6 0 1）。そして、仮想マシン部 1 3 0 は、ファンクションノード制御部 1 1 0 から受け取った入力メッセージを引数として、ステップ S 6 0 1 で取得したコンパイルされたプログラム 1 8 2 を実行し（ステップ S 6 0 2）、実行結果をファンクションノード制御部 1 1 0 に送る（ステップ S 6 0 3）。

[0046] 図 1 5 は、本変形例におけるリカバリ部 1 4 0 の処理手順の一例を示すフローチャートである。情報処理装置 1 0 0 の起動時に、リカバリ部 1 4 0 は、まず、最新のスナップショットをスナップショット記憶部 1 7 0 から読み出し、仮想マシン部 1 3 0 内に展開する（ステップ S 7 0 1）。そして、リカバリ部 1 4 0 は、最新のスナップショットの書き出し時刻よりも後に記録されたジャーナルログがあるかどうかを確認する（ステップ S 7 0 2）。そして、最新のスナップショットの書き出し時刻よりも後に記録されたジャーナルログがあれば（ステップ S 7 0 2 : Y e s）、ジャーナルログの読み出し位置を最新のスナップショット直後に設定する（ステップ S 7 0 3）。一方、最新のスナップショットの書き出し時刻よりも後に記録されたジャーナルログがなければ（ステップ S 7 0 2 : N o）、そのまま処理を終了する。

[0047] 次に、リカバリ部 1 4 0 は、ジャーナルログ記憶部 1 6 0 から、設定した読み出し位置のジャーナルログを読み出す（ステップ S 7 0 4）。そして、読み出したジャーナルログが入力メッセージとプログラム I D との組であることを確認する（ステップ S 7 0 5）。ここで、読み出したジャーナルログが入力メッセージとプログラム I D との組であれば（ステップ S 7 0 5 : Y e

s)、ジャーナルログを構成する入力メッセージとプログラムIDを引数として、仮想マシン部130内でcallScriptを実行させる(ステップS706)。一方、読み出したジャーナルログがプログラムIDとファンクションノード180のプログラム182との組であれば、これらプログラムIDとプログラム182との組を仮想マシン部130内に登録させる(ステップS707)。

[0048] その後、リカバリ部140は、ジャーナルログの読み出し位置を次のジャーナルログの記録時刻にシフトし(ステップS708)、読み出し位置のジャーナルログがあるか否かを確認する(ステップS709)。そして、読み出し位置のジャーナルログがあれば(ステップS709:Yes)、ステップS704に戻って以降の処理を繰り返す。一方、読み出し位置のジャーナルログがなければ(ステップS709:No)、処理を終了する。

[0049] 以上のように、本変形例では、ファンクションノード180の入力メッセージが入力されたときに、ファンクションノード180のプログラム182の本体に代えてプログラムIDを入力メッセージとともにジャーナルログとして記録したり、仮想マシン部130に送ったりしている。したがって、ジャーナルログのサイズを小さくし、ジャーナルログ記憶部160の容量削減を図ることができるとともに、仮想マシン部130に送るデータサイズを小さくし、転送時間の短縮を図ることができる。さらに、本変形例では、ファンクションノード180のプログラム182を仮想マシン部130内に登録する際に、事前にプログラム182の構文解析を行いコンパイルしておくことで、ファンクションノード180の入力メッセージが入力されたときに、プログラム182の実行時間を短縮することができる。

[0050] <第2実施形態>

本実施形態は、上述の第1実施形態に機能を加えたものである。追加の機能は、永続的実行部120を起点とするメッセージをフロー内に送り込む機能である。

[0051] 永続的実行環境190に対応して設けられる永続的実行部120は、単な

る永続的なデータストアとして機能するというだけでなく、実行環境自体が永続的であるから、その中でのプログラムの実行もまた永続的である。したがって、これを積極的に利用するならば、永続的実行部120を起点としたメッセージが外部に伝達されるような設計も可能である。しかし、他機器へのアクセス等は通常FBPの処理ノードとして表現されており、これは永続的実行環境190と関連していない既存の処理ノードである。そこで本実施形態では、イベントリスナノードと呼ぶ処理ノードを用いることにより、この間の接続を可能にする。イベントリスナノードは、永続的実行環境名とイベント名がユーザ定義される処理ノードである。

[0052] 図16は、本実施形態に係る情報処理装置100の機能的な構成例を示すブロック図である。本実施形態に係る情報処理装置100は、図16に示すように、上述の第1実施形態の構成に対して、FBPのプログラミング環境から情報処理装置100にデプロイされたイベントリスナノード200に対応して設けられるイベントリスナノード制御部210が追加された構成である。FBPのプログラミング環境からデプロイされたイベントリスナノード200は、ユーザ定義された永続的実行環境名201およびイベント名202を含む。イベントリスナノード200の永続的実行環境名201と永続的実行環境190の永続的実行環境名191とを一致させることで、これらイベントリスナノード200と永続的実行環境190とが関連付けられる。

[0053] 本実施形態における仮想マシン部130は、上述の第1実施形態と同様にファンクションノード180のプログラムを実行するが、このプログラムの実行中にemit関数を実行すると、該emit関数の引数であるイベント名とメッセージをイベントリスナノード制御部210に送る。emit関数は、永続的実行環境190の予約変数として、仮想マシン部130に固有のグローバルスコープ135に登録された予約変数“emit”に格納されるJavaScriptの関数であり、イベント名とメッセージを引数とする。

[0054] イベントリスナノード制御部210は、仮想マシン部130からイベント名とメッセージを受け取ると、受け取ったイベント名がイベントリスナノード

ド200にユーザ定義されたイベント名202と一致するか否かを確認し、一致する場合に仮想マシン部130から受け取ったメッセージを出力ポート211から出力する。

[0055] 図17は、本実施形態における仮想マシン部130の処理手順の一例を示すフローチャートであり、ファンクションノード180のプログラム182を実行する際の処理の流れを示している。仮想マシン部130は、ファンクションノード制御部110から入力メッセージとプログラム182を受け取ると、受け取った入力メッセージを引数とするプログラム182の実行を開始する（ステップS801）。そして、プログラム182の実行中、emit関数の実行を監視し（ステップS802）、emit関数を実行した場合は（ステップS802：Yes）、emit関数の引数であるイベント名とメッセージをイベントリスナノード制御部210に送る（ステップS803）。その後、プログラム182の実行が終了すると（ステップS804：Yes）、仮想マシン部130は、プログラム182の実行結果をファンクションノード制御部110に送り（ステップS805）、処理を終了する。

[0056] 図18は、イベントリスナノード制御部210の処理手順の一例を示すフローチャートである。イベントリスナノード制御部210は、仮想マシン部130からイベント名とメッセージを受け取ると、受け取ったイベント名がイベントリスナノード200に定義されたイベント名202と一致するか否かを確認する（ステップS901）。そして、仮想マシン部130から受け取ったイベント名がイベントリスナノード200に定義されたイベント名202と一致する場合は（ステップS901：Yes）、仮想マシン部130から受け取ったメッセージを出力メッセージとして、出力ポート211から出力する（ステップS902）。一方、仮想マシン部130から受け取ったイベント名がイベントリスナノード200に定義されたイベント名202と一致しない場合は（ステップS901：No）、仮想マシン部130から受け取ったメッセージを破棄して処理を終了する。

[0057] 以上のように、本実施形態では、FBPのプログラミング環境から情報処

理装置 100 にイベントリスナノード 200 をデプロイすることで情報処理装置 100 にイベントリスナノード制御部 210 が設けられ、永続的実行環境 190 の予約変数に格納した emit 関数が実行されると emit 関数の引数となるメッセージを出力する。したがって、永続的実行部 120 を起点としたメッセージを外部に伝達することができる。

[0058] <第 3 実施形態>

本実施形態は、上述の第 1 実施形態に機能を加えたものである。追加の機能は、読み込み専用（リードオンリ）での永続的実行部 120 へのアクセス機能である。

[0059] 上述の第 1 実施形態では、例えば単純にグローバルスコープ 135 の変数 x の値を参照するだけだとしても、 x の値を返すプログラム “return x ” を実行しなければならない。内部的には、このプログラムがジャーナルログとしてジャーナルログ記憶部 160 に記録され、このプログラムを実行するためのファンクションスコープ 134 が生成され、プログラムが実行され、変数名解決部 136 が変数 x を解決しその値を得るといった手順が取られる。しかし、リードオンリのアクセスは仮想マシン部 130 の状態を変更するものではないため、そもそも仮想マシン部 130 の状態を復元することに備えて上述の手順で処理を行う必要はない。そこで、本実施形態では、リードオンリで永続的実行部 120 にアクセスするための機能としてリードオンリアクセス部を設け、グローバルスコープ 135 に登録された変数の値を参照するだけであれば、このリードオンリアクセス部を通じて永続的実行部 120 にアクセスできるようにする。

[0060] また、本実施形態では、グローバルスコープ 135 に登録された変数の値を参照する処理ノードとして、テンプレート処理ノードを例示する。テンプレート処理ノードは、HTML (HyperText Markup Language) 文書等のテンプレートがユーザ定義される処理ノードであり、埋め込み箇所には変数名や変数名を使った式が指定される。また、本実施形態で使用するテンプレート処理ノードは、永続的実行環境名がユーザ定義されることで、永続的実行

環境 190 と関連付けられる。

[0061] 図 19 は、本実施形態に係る情報処理装置 100 の機能的な構成例を示すブロック図である。本実施形態に係る情報処理装置 100 は、図 19 に示すように、上述の第 1 実施形態の構成に対して、FBP のプログラミング環境から情報処理装置 100 にデプロイされたテンプレート処理ノード 220 に対応して設けられるテンプレート処理ノード制御部 230 と、リードオンリアクセス部 240 とが追加された構成である。FBP のプログラミング環境からデプロイされたテンプレート処理ノード 220 は、ユーザ定義された永続的実行環境名 221 およびテンプレート 222 を含む。テンプレート処理ノード 220 の永続的実行環境名 221 と永続的実行環境 190 の永続的実行環境名 191 とを一致させることで、これらテンプレート処理ノード 220 と永続的実行環境 190 とが関連付けられる。

[0062] リードオンリアクセス部 240 は、上述のようにリードオンリで永続的実行部 120 にアクセスするための機能であり、仮想マシン部 130 のグローバルスコープ 135 に登録された変数の値を取得する。リードオンリアクセス部 240 が値を取得する際、仮想マシン部 130 の状態を変化させない。

[0063] テンプレート処理ノード制御部 230 は、テンプレート処理ノード 220 の入力メッセージが入力ポート 231 に入力されると、テンプレート処理ノード 220 にユーザ定義されたテンプレート 222 の埋め込み箇所に指定された変数名を、仮想マシン部 130 のグローバルスコープ 135 に登録された変数の変数名として、リードオンリアクセス部 240 を通じて、仮想マシン部 130 のグローバルスコープ 135 からその値を読み出す。そして、テンプレート処理ノード制御部 230 は、テンプレート 222 の埋め込み箇所に、リードオンリアクセス部 240 を通じて仮想マシン部 130 のグローバルスコープ 135 から読み出した値を埋め込み、この値を埋め込んだテンプレート 222 を出力メッセージとして出力ポート 232 から出力する。なお、変数名を使った式の場合は、同様にリードオンリアクセス部 240 を通じて変数名を値に置き換えた後、式が計算され、その結果がテンプレート 22

2に埋め込まれる。この式の計算は永続的実行部120の仮想マシン部130の外、つまり仮想マシン部130とは別のインスタンスで実行される。

[0064] 以上のように、本実施形態では、リードオンリで永続的実行部120にアクセスするリードオンリアクセス部240を設け、テンプレート処理ノード制御部230が仮想マシン部130のグローバルスコープ135に登録された変数の値を参照する際に、リードオンリアクセス部240を通じて値を取得するようにしている。したがって、永続的実行部120にアクセスするたびにジャーナルログを記録するといった手順を踏む必要はなく、その分、実行性能が向上するとともに、情報処理装置100の起動時に仮想マシン部130の状態を復元するのに要するリカバリ時間が短縮される。

[0065] <第4実施形態>

本実施形態は、永続的実行部120にリードオンリでアクセスするためのリードオンリアクセス部240を備える点は上述の第3実施形態と共通であるが、グローバルスコープ135に登録された変数の値を参照する処理ノードが第3実施形態とは異なる。すなわち、本実施形態では、グローバルスコープ135に登録された変数の値を参照する処理ノードとして、リードオンリファンクションノードを例示する。

[0066] リードオンリファンクションノードは、プログラムをユーザ定義できる点で上述のファンクションノード180と同様であるが、プログラムの実行自体は独自のインスタンスで行い、プログラム内で未定義の変数からの読み出しに関しては永続的実行環境190を利用する処理ノードである。

[0067] 図20は、本実施形態に係る情報処理装置100の機能的な構成例を示すブロック図である。本実施形態に係る情報処理装置100は、図20に示すように、上述の第3実施形態のテンプレート処理ノード制御部230に代えて、FBPのプログラミング環境から情報処理装置100にデPLOYされたリードオンリファンクションノード250に対応して設けられるリードオンリファンクションノード制御部260を備える。FBPのプログラミング環境からデPLOYされたリードオンリファンクションノード250は、ユーザ

定義された永続的実行環境名 251 およびプログラム 252 を含む。リードオンリファンクションノード 250 の永続的実行環境名 251 と永続的実行環境 190 の永続的実行環境名 191 とを一致させることで、これらリードオンリファンクションノード 250 と永続的実行環境 190 とが関連付けられる。

[0068] リードオンリファンクションノード制御部 260 は、リードオンリファンクションノード 250 の入力メッセージが入力ポート 261 に入力されると、その入力メッセージを引数としてリードオンリファンクションノード 250 にユーザ定義されたプログラム 252 を実行し、実行結果を出力メッセージとして出力ポート 262 から出力する。リードオンリファンクションノード制御部 260 によるプログラム 252 の実行は、永続的実行部 120 の仮想マシン部 130 とは別の固有のインスタンスで行われ、仮想マシン部 130 の状態に影響を及ぼさない。

[0069] リードオンリファンクションノード制御部 260 は、プログラム 252 を実行する際、プログラム 252 内で未定義の変数からの読み出しに関しては、リードオンリアクセス部 240 を通じて、仮想マシン部 130 のグローバルスコープ 135 からその値を読み出す。なお、プログラム 252 内で未定義の変数への書き込みがあった場合には、リードオンリファンクションノード制御部 260 は、まずリードオンリアクセス部 240 を通じて仮想マシン部 130 のグローバルスコープ 135 からその値を読み出す。そして、読み出した値を書き込もうとしている値と比較して、異なっているならば参照エラーのエクセプションとする。一方、仮想マシン部 130 のグローバルスコープ 135 から読み出した値が書き込もうとしている値と一致している場合には、実際の書き込みは行わずに、書き込みが成功したかのようにふるまう。

[0070] なお、リードオンリファンクションノード 250 は、上述のファンクションノード 180 とシームレスに切り替えられるようにしておくこと、プログラミング環境での使い勝手が向上する。具体的には、プログラミング環境で表

示するUI上ではファンクションノード180とリードオンリファンクションノード250とを兼用の処理ノードとして見せ、リードオンリの属性をチェックボックス等で切り替えられるようにするといった方法が考えられる。

[0071] 以上のように、本実施形態では、リードオンリファンクションノード制御部260が仮想マシン部130のグローバルスコープ135に登録された変数の値を参照する際に、リードオンリアクセス部240を通じて値を取得するようにしている。したがって、上述の第3実施形態と同様に、永続的実行部120にアクセスするたびにジャーナルログを記録するといった手順を踏む必要はなく、その分、実行性能が向上するとともに、情報処理装置100の起動時に仮想マシン部130の状態を復元するのに要するリカバリ時間が短縮される。

[0072] <第5実施形態>

本実施形態は、上述の各実施形態の情報処理装置100の機能を利用して、過去の任意の時刻における仮想マシン部130の状態を再現できるようにしたものである。すなわち、情報処理装置100は電源オフ後の起動時に、スナップショット記憶部170に記録されたスナップショットとジャーナルログ記憶部160に記録されたジャーナルログとを利用して、永続的実行部120の仮想マシン部130を最新の状態にロールフォワードする。これと同様の手順で、スナップショットとジャーナルログとを利用することにより、過去の任意の時刻における仮想マシン部130の状態を再現することができる。

[0073] なお、仮想マシン部130の状態を再現する装置（以下、これを「再現装置」と呼ぶ）は、その仮想マシン部130を備える情報処理装置100とは別の装置である。再現装置は、例えば、ネットワークを介して情報処理装置100と接続される。あるいは、情報処理装置100に対して再現装置を直接接続する構成であってもよい。また、情報処理装置100のスナップショット記憶部170に記録されたスナップショットとジャーナルログ記憶部160に記録されたジャーナルログを記録媒体に格納し、この記録媒体を再現

装置に接続する構成であってもよい。上述の各実施形態では、情報処理装置 100 としてゲートウェイと呼ばれる IOT 機器を想定するため再現装置を情報処理装置 100 とは別の装置としているが、情報処理装置 100 自体がユーザインタフェースとなるコンソールを備える構成であれば、再現装置を情報処理装置 100 内部の機能として実現することもできる。再現装置は、情報処理装置 100 のトラブル解析などで過去の状態を詳細に解析したい場合に強力なツールとなり、実用上有用な機能である。

[0074] 本実施形態では、情報処理装置 100 と再現装置とがネットワークを介して接続された情報処理システムを例に説明する。図 21 は、本実施形態に係る情報処理システムの概略構成を示すブロック図である。情報処理システムは、図 21 に示すように、上述の実施形態として説明したいずれかの情報処理装置 100 と再現装置 300 とがネットワーク 350 を介して接続された構成である。再現装置 300 は、例えば、情報処理装置 100 と同様に汎用のコンピュータとしてのハードウェア資源と、コンピュータの周辺機器としてユーザインタフェースとなるコンソールとを備える。再現装置 300 は、コンピュータ上で実行されるプログラム（ソフトウェア）とコンソールとを用いて実現される機能的な構成要素として、図 21 に示すように、コンソール部 310 と状態再現部 320 とを備える。

[0075] コンソール部 310 は、過去の任意の時刻（情報処理装置 100 の仮想マシン部 130 の状態を再現しようとする対象となる時刻）を指定するユーザ操作を受け付ける。状態再現部 320 は、情報処理装置 100 のリカバリ部 140 と同様の手法により、ユーザ操作により指定された時刻における情報処理装置 100 の仮想マシン部 130 の状態を再現する。すなわち、状態再現部 320 は、ユーザ操作により指定された時刻よりもさらに過去のある時刻に記録されたスナップショットを情報処理装置 100 から取得して再現装置 300 の内部メモリ上に展開することで、その時刻における仮想マシン部 130 の状態を内部メモリ上に再現する。そして、状態再現部 320 は、その時刻からユーザ操作により指定された時刻までの間に記録されたジャーナ

ルログを情報処理装置１００から取得し、各ジャーナルログを構成する入力メッセージを引数としたプログラムの実行を内部メモリ上で順次行うことにより、ユーザ操作により指定された時刻における仮想マシン部１３０の状態を再現する。

[0076] どの時刻の仮想マシン部１３０の状態を再現するかは、例えば、ジャーナルログに付けられたタグにより識別することができる。標準では、ジャーナルログに対して記録日時を表すタグとしてタイムスタンプが付加されるが、記録するジャーナルログが所定の条件を満たしたときに、その条件を表す文字列などをタグとして付加したり、ジャーナルログ記憶部１６０に記録されているジャーナルログに繰り返し現れる連続パターンを認識することにより、パターンに変化があったときに自動的にタグを付加したりといったことも可能である。

[0077] コンソール部３１０は、例えば、タグが付加されたジャーナルログの一覧を表示して、その中から所望のジャーナルログを選択するユーザ操作を受け付ける構成とすることができる。この場合、一般にジャーナルログ記憶部１６０に記録されたジャーナルログは大量にあるので、時系列での範囲限定や、特定のタグの文字列の検索・絞り込み機能などを備えて、その表示を行うことが望ましい。

[0078] なお、タグは、ジャーナルログに付加する形態ではなく、通常のジャーナルログとは別のエントリとして、タグエントリをジャーナルログ記憶部１６０に記録する構成であってもよい。この場合、コンソール部３１０は、タグエントリの一覧を表示して、その中から所望のタグエントリを選択するユーザ操作を受け付ける構成とすることができる。この構成の場合、ユーザ操作により選択されたタグエントリの直前に記録されたジャーナルログを仮想マシン部１３０に反映させた状態が、再現すべき仮想マシン部１３０の状態となる。

[0079] 図２２は、本実施形態に係る情報処理装置１００の機能的な構成例を示すブロック図である。本実施形態に係る情報処理装置１００は、図２２に示す

ように、上述の第1実施形態の構成に対して、タグ付け部270が追加された構成である。なお、上述の第1実施形態では、リカバリ部140が仮想マシン部130を最新の状態に復元するために利用するのは、最新のスナップショットとそれ以降に記録されたジャーナルログであるため、最新のスナップショットが記録されるたびにそれ以前に記録されたスナップショットおよびジャーナルログを削除してもよい。これに対し本実施形態では、過去の任意の時刻における仮想マシン部130の状態を再現するため、スナップショット記憶部170およびジャーナルログ記憶部160の容量に応じて、可能な限り過去のスナップショットとジャーナルログを保存しておくことが望ましい。

[0080] タグ付け部270は、ジャーナルログの記録時に、そのジャーナルログに対して日付や任意の文字列などをタグとして付加する。タグ付け部270は、上述のように、ジャーナルログが所定の条件を満たしたときに、その条件を表す所定の文字列などをタグとして付加する構成であってもよいし、ジャーナルログ記憶部160に記録されているジャーナルログに繰り返し現れる連続パターンを認識することにより、パターンに変化があったときに、変化の内容を表す文字列などを自動的にタグとして付加する構成であってもよい。また、このような構成の場合には、タグ付け部270は上述のように、通常のジャーナルログとは別のエントリとして、タグエントリをジャーナルログ記憶部160に記録してもよい。

[0081] 情報処理装置100がこのようなタグ付け部270を備えることで、再現装置300のコンソール部310は、タグが付与されたジャーナルログの一覧あるいはタグエントリの一覧を表示して、この一覧の中から所望のジャーナルログあるいはタグエントリを選択する操作を、仮想マシン部130の状態を再現すべき時刻を指定するユーザ操作として受け付けることができる。

[0082] 図23は、再現装置300の処理手順の一例を示すフローチャートである。まず、コンソール部310が、仮想マシン部130の状態を再現すべき時刻を指定するユーザ操作を受け付ける（ステップS1001）。そして、状

態再現部 320 が、ユーザ操作により指定された時刻の直前に記録されたスナップショットを情報処理装置 100 から取得し、取得したスナップショットをメモリ上に展開する（ステップ S1002）。

[0083] 次に、状態再現部 320 は、取得したスナップショットの書き出し時刻よりも後に記録されたジャーナルログがあるかどうかを確認する（ステップ S1003）。そして、取得したスナップショットの書き出し時刻よりも後に記録されたジャーナルログがあれば（ステップ S1003：Yes）、情報処理装置 100 から取得するジャーナルログの時刻を、取得したスナップショットの直後に設定する（ステップ S1004）。一方、取得したスナップショットの書き出し時刻よりも後に記録されたジャーナルログがなければ（ステップ S1003：No）、そのまま処理を終了する。

[0084] 次に、状態再現部 320 は、設定したジャーナルログの時刻がユーザ操作により指定された時刻よりも後かどうかを確認する（ステップ S1005）。ここで、設定したジャーナルログの時刻がユーザ操作により指定された時刻以前であれば（ステップ S1005：No）、状態再現部 320 は、設定した時刻のジャーナルログを情報処理装置 100 から取得し（ステップ S1006）、取得したジャーナルログを構成する入力メッセージを引数として、その取得したジャーナルログを構成するプログラム 182 をメモリ上で実行する（ステップ S1007）。その後、状態再現部 320 は、取得するジャーナルログの時刻を次のジャーナルログの記録時刻にシフトし（ステップ S1008）、ステップ S1005 に戻って以降の処理を繰り返す。そして、情報処理装置 100 から取得するジャーナルログの時刻がユーザ操作により指定された時刻よりも後になると（ステップ S1005：Yes）、処理を終了する。

[0085] なお、以上の説明では、再現装置 300 が情報処理装置 100 からジャーナルログを時刻順に順次取得するものとして説明したが、再現装置 300 は情報処理装置 100 からスナップショットを取得する際に、その書き出し時刻からユーザ操作により指定された時刻までの間に記録されたすべてのジャ

ーナルログを一括して取得し、取得したジャーナルログを順次メモリ上に反映させるようにしてもよい。

[0086] 再現装置300の状態再現部320により再現された任意の時刻における情報処理装置100の仮想マシン部130の状態は、例えば、コンソール部310がその状態を何らかのかたちで表示することによりユーザに提示できる。この場合、ユーザはコンソール部310を通じて対話的に変数の内容を参照したり、任意のプログラムを実行したりすることができる。

[0087] また、再現装置300は、コンソール部310を利用したユーザ操作に応じて、状態再現部320により再現された任意の時刻における情報処理装置100の仮想マシン部130の状態に変更を加えることができる。さらに、その再現した時刻以降に記録されたジャーナルログを任意の数だけ情報処理装置100から取得してメモリ上で順次実行することにより、変更を加えた過去の仮想マシン部130の状態をロールフォワードすることができる。そして、例えばコンソール部310を利用したユーザ操作に応じて、ロールフォワードした仮想マシン部130の状態を再現装置300から情報処理装置100に送り、情報処理装置100の仮想マシン部130の状態を、再現装置300から送られた状態に置き換えることができる。

[0088] この機能は、情報処理装置100の仮想マシン部130の過去の状態を変更し、この変更を反映させて仮想マシン部130の最新の状態を変更することを可能にする。つまり、情報処理装置100が記録するジャーナルログは、仮想マシン部130の変更履歴ではなく、仮想マシン部130で実行されたプログラムとその引数であるため、仮想マシン部130の過去の状態を変えた後で最新の状態までロールフォワードすると、その変更を反映させた別の最新の状態が出現するということになる。これを実際の仮想マシン部130に適用することで、仮想マシン部130の最新の状態を変更することができる。例えば、過去のトラブルのあった時刻の仮想マシン部130の状態を再現装置300により再現し、問題箇所を修正した後で、ロールフォワード実行によって最新の状態まで進めると、トラブルの影響を除いた現在の状態

が生成できる。

[0089] 以上のように、本実施形態では、過去の任意の時刻における情報処理装置 100 の仮想マシン部 130 の状態を、再現装置 300 により再現することができる。そして、再現した仮想マシン部 130 の過去の状態に対して変更を加え、ロールフォワードして仮想マシン部 130 に適用することにより、仮想マシン部 130 の最新の状態を、例えばトラブルの原因となっている過去の問題箇所に対する修正を反映した状態に変更することができる。

[0090] (変形例)

上述の第5実施形態に係る情報処理装置 100 は、ジャーナルログにタグを付ける機能を、FBPのプログラミング環境で作成されるフローからは独立した機能として実現しているが、例えば、所定の処理が開始・完了するとその時刻のタグを付ける（タグエントリをジャーナルログ記憶部 160 に記録する）ためのタグ付け処理ノードを用いることで、フロー内からタグを付けることも可能である。

[0091] 図24は、本変形例に係る情報処理装置 100 の機能的な構成例を示すブロック図である。本変形例に係る情報処理装置 100 は、図24に示すように、上述の第5実施形態の構成に対して、FBPのプログラミング環境から情報処理装置 100 にデプロイされたタグ付け処理ノード 280 に対応して設けられるタグ付け処理ノード制御部 290 が追加された構成である。FBPのプログラミング環境からデプロイされたタグ付け処理ノード 280 は、ユーザ定義された永続的実行環境名 281 を含み、この永続的実行環境名 281 を永続的実行環境 190 の永続的実行環境名 191 と一致させることで、これらタグ付け処理ノード 280 と永続的実行環境 190 とが関連付けられる。

[0092] タグ付け処理ノード制御部 290 は、タグ付け処理ノード 280 の入力メッセージが入力ポート 291 に入力されると、その入力メッセージに含まれる文字列をタグ付け部 270 に送る。タグ付け部 270 は、タグ付け処理ノード制御部 290 から文字列を受け取ると、その文字列をファンクションノ

ード制御部 110 が記録するジャーナルログにタグとして付加する。あるいはタグ付け部 270 は、タグ付け処理ノード制御部 290 から受け取った文字列をタグエントリとして、ジャーナルログ記憶部 160 に記録してもよい。

[0093] <第 6 実施形態>

上述の各実施形態では、ファンクションノードとこれに関連付けられた永続的実行環境とが FBP のプログラミング環境から同一の情報処理装置 100 にデプロイされることを想定したが、ファンクションノードと永続的実行環境とを別の情報処理装置にデプロイし、一方の情報処理装置にファンクションノード制御部 110 が設けられ、他方の情報処理装置に永続的実行部 120 が設けられる構成であってもよい。すなわち、ファンクションノードの入力メッセージが入力される情報処理装置とは別の情報処理装置において、ファンクションノードのプログラムが永続的に実行される構成としてもよい。

[0094] 図 25 は、本実施形態に係る情報処理システムの概略構成を示すブロック図である。本実施形態に係る情報処理システムは、図 25 に示すように、ファンクションノード制御部 110 が設けられた情報処理装置 100A と、永続的実行部 120 が設けられた情報処理装置 100B とがネットワーク 350 を介して接続された構成である。情報処理装置 100A は、例えばゲートウェイなどの IOT 機器であり、情報処理装置 100B はリモートのサーバ装置などである。なお、ジャーナルログ記憶部 160 とスナップショット記憶部 170 は、情報処理装置 100B にある。

[0095] 本実施形態においても、ファンクションノード制御部 110 と永続的実行部 120 の動作は、基本的には上述の各実施形態と同様である。ただし、ファンクションノード制御部 110 から永続的実行部 120 へ渡される入力メッセージやプログラムは、TCP/IP や HTTP などの通信プロトコルに従って、情報処理装置 100A から情報処理装置 100B へとネットワーク 350 経由で送られる。また、プログラムの実行結果は、同じ通信プロトコ

ルに従って、情報処理装置１００Ｂから情報処理装置１００Ａへとネットワーク３５０経由で送られる。

[0096] また、本実施形態では、ファンクションノードと永続的実行環境とを関連付ける永続的実行環境名は、例えばＵＲＬ形式により情報処理装置１００Ｂを指定するなど、永続的実行環境がデプロイされた情報処理装置１００Ｂを含めた名前となる。また、ファンクションノードと永続的実行環境との関連付け以外にも、情報処理装置１００Ａと情報処理装置Ｂとの接続状態が管理され、情報処理装置１００Ａから情報処理装置Ｂへの接続が不能となった場合、ファンクションノードのプログラムの実行は、接続エラーとなって異常終了する。

[0097] また、本実施形態では、永続的実行部１２０のリカバリ部１４０が動作するのは、ファンクションノード制御部１１０が設けられた情報処理装置１００Ａ（例えばＩｏＴ機器）の起動時ではなく、永続的実行部１２０が設けられた情報処理装置１００Ｂ（例えばリモートのサーバ装置）の起動時である。

[0098] 以上のように、本実施形態では、ファンクションノード制御部１１０と永続的実行部１２０とを別々の情報処理装置に設け、ファンクションノードの入力メッセージが入力される情報処理装置１００Ａとは別の情報処理装置１００Ｂにおいて、ファンクションノードのプログラムが永続的に実行される構成である。このため、情報処理装置１００Ａの処理負荷を低減させることができる。

[0099] また、本実施形態の構成では、ファンクションノード制御部１１０が設けられた複数の情報処理装置１００Ａ（例えばＩｏＴ機器）を、永続的実行部１２０が設けられた共通の情報処理装置１００Ｂ（例えばリモートのサーバ装置）に接続し、複数の情報処理装置１００Ａに各々設けられたファンクションノード制御部１１０が共通の永続的実行部１２０を共有することができる。

[0100] <第７実施形態>

本実施形態は、上述の第6実施形態に係る情報処理システムを改良するものである。第6実施形態では、情報処理装置100Aから情報処理装置100Bへの接続が不能となった場合、ファンクションノードのプログラムの実行がエラーとなってしまいう課題がある。そこで、本実施形態では、情報処理装置100Bだけでなく情報処理装置100Aにも永続的実行部120を設け、ファンクションノードのプログラムを情報処理装置100Bの永続的実行部120だけでなく、情報処理装置100Aの永続的実行部120においても実行することで、この課題を解決する。

[0101] 図26は、本実施形態に係る情報処理システムの概略構成を示すブロック図である。本実施形態では、図26に示すように、情報処理装置100Aにファンクションノード制御部110と永続的実行部120Aとが設けられ、情報処理装置100Bに永続的実行部120Bが設けられている。永続的実行部120Aと永続的実行部120Bは、FBPのプログラミング環境から共通の永続的実行環境が情報処理装置100Aと情報処理装置100Bとの双方にデプロイされることで設けられるものである。また、本実施形態では、ジャーナルログ記憶部160とスナップショット記憶部170が、情報処理装置100Aと情報処理装置100Bの双方に設けられる。

[0102] 本実施形態では、情報処理装置100Aから情報処理装置100Bへの接続が可能な場合（通常状態）、ファンクションノードの入力メッセージやプログラムは、情報処理装置100Aと情報処理装置100Bの双方に設けられたジャーナルログ記憶部160にジャーナルログとして記録されるとともに、ファンクションノード制御部110から永続的実行部120Aと永続的実行部120Bの双方に送られる。永続的実行部120Aと永続的実行部120Bは、それぞれファンクションノード制御部110から受け取った入力メッセージを引数として、ファンクションノードのプログラムを仮想マシン部130で実行し、実行結果をファンクションノード制御部110に返す。

[0103] このとき、永続的実行部120Aと永続的実行部120Bの双方で、ファンクションノードのプログラムの実行順序は一致している必要がある。例え

ば、入力メッセージAと入力メッセージBとがほぼ同時のタイミングでファンクションノードに入力されたとき、入力メッセージAを引数とするプログラムの実行と、入力メッセージBを引数とするプログラムの実行とをどちらに先にするか（これはジャーナルログ記憶部160への記録順序と一致する）は、永続的実行部120Aと永続的実行部120Bとで一致していなければならない。

[0104] 例えば、永続的実行部120Aでは入力メッセージAを引数としてプログラムを実行した後、入力メッセージBを引数としてプログラムを実行し、一方、永続的実行部120Bでは入力メッセージBを引数とするプログラムを実行した後、入力メッセージAを引数としてプログラムを実行すると、双方の実行結果が異なる可能性が出てくるため、これは避けなければならない。プログラムの実行順序を一致させる方法としては、例えば、ファンクションノード制御部110がメッセージの送信元として順序番号を付与すればよい。あるいは、入力メッセージとプログラムの組をいったん情報処理装置100Aのジャーナルログ記憶部160に記録して順序を確定してから、情報処理装置100Bの永続的実行部120Bに送る構成としてもよい。

[0105] 本実施形態では、通常状態であればファンクションノード制御部110が永続的実行部120Aと永続的実行部120Bの双方からプログラムの実行結果を受け取るが、先に受け取った実行結果を出力メッセージとして出力し、後から受け取った実行結果は破棄すればよい。これにより、例えばプログラムの実行中に情報処理装置100Aから情報処理装置100Bへの接続が不能になったとしても、有効な結果を得ることができる。また、情報処理装置100Bがサーバ装置であれば、重い処理は情報処理装置100Bで実行した方が速いため、情報処理装置100Bに設けられた永続的実行部120Bによるプログラムの実行結果が利用され、処理の高速化が実現される。

[0106] また、本実施形態では、情報処理装置100Aから情報処理装置100Bへの接続が不能な場合、ファンクションノードの入力メッセージやプログラムは、情報処理装置100Aに設けられたジャーナルログ記憶部160にジ

ジャーナルログとして記録されるとともに、ファンクションノード制御部 110 から永続的実行部 120A に送られる。永続的実行部 120A は、それぞれファンクションノード制御部 110 から受け取った入力メッセージを引数として、ファンクションノードのプログラムを仮想マシン部 130 で実行し、実行結果をファンクションノード制御部 110 に返す。そして、ファンクションノード制御部 110 は、永続的実行部 120A から受け取ったプログラムの実行結果を出力メッセージとして出力する。

[0107] また、本実施形態では、情報処理装置 100A から情報処理装置 100B への接続が再開した場合、情報処理装置 100A に設けられたジャーナルログ記憶部 160 に記録されたジャーナルログのうち、情報処理装置 100B 側に送信されていないジャーナルログが順次取り出され、情報処理装置 100A から情報処理装置 100B に送られる。情報処理装置 100B では、情報処理装置 100A から送られるジャーナルログを順次ジャーナルログ記憶部 160 に記録するとともに、ジャーナルログを構成する入力メッセージを引数とするプログラムの実行を永続的実行部 120B の仮想マシン部 130 で順次行うことにより、結果として、永続的実行部 120B の仮想マシン部 130 の状態を、永続的実行部 120A の仮想マシン部 130 の状態に同期させる。そして、この同期が完了した後、上述の通常状態に復帰する。

[0108] なお、本実施形態では、ファンクションノードのプログラムの実行を 2 重化した例について説明したが、上述の第 6 実施形態で説明したように、複数の情報処理装置 100A を共通の情報処理装置 100B に接続し、複数の情報処理装置 100A の各々に設けられたファンクションノード制御部 110 が情報処理装置 100B に設けられた永続的実行部 120B を共有する場合には、3 重化以上の多重実行の構成とすることも可能である。その場合、ファンクションノードの入力メッセージとプログラムの送信元となるファンクションノード制御部 110 が複数あるため、その間でプログラムの実行順序を一貫性をもって決定する必要がある。これを実現するには、いわゆる分散アルゴリズムの技術を用いればよい。具体的には、例えば特許第 61003

８４号公報にて開示される技術を利用してプログラムの実行順序を決定することができる。プログラムの実行順序が決定したならば、その後は上述の２重化の場合と同様に、それぞれの永続的実行部１２０Ａ、１２０Ｂで同様の処理を行えばよい。また、接続再開時におけるジャーナルログのコピーは、複数の情報処理装置１００Ａのうち、いずれかの情報処理装置１００Ａのジャーナルログ記憶部１６０からコピーすればよい。

[0109] <補足説明>

上述した各実施形態における情報処理装置１００の機能は、例えば、一般的なコンピュータを構成するハードウェアと、コンピュータで実行されるプログラム（ソフトウェア）との協働により実現することができる。例えば、ＦＢＰの実行環境となるコンピュータにファンクションノード１８０や永続的実行環境１９０がデプロイされ、このコンピュータをＦＢＰの実行環境として動作させるためのプログラムが実行されることによって、上述したファンクションノード制御部１１０や永続的実行部１２０などの機能的な構成要素がコンピュータ上で実現される。

[0110] 図２７は、情報処理装置１００のハードウェア構成の一例を示すブロック図である。情報処理装置１００は、例えば図２７に示すように、ＣＰＵ（Central Processing Unit）やＧＰＵ（Graphics Processing Unit）などのプロセッサ１０１と、ＲＡＭ（Random Access Memory）やＲＯＭ（Read Only Memory）などの内部メモリ１０２と、ＨＤＤ（Hard Disk Drive）やＳＳＤ（Solid State Drive）などのストレージデバイス１０３と、各種周辺機器を接続するための入出力Ｉ／Ｆ１０４と、外部機器と通信を行う通信Ｉ／Ｆ１０５と、を備えた一般的なコンピュータのハードウェア構成を有する。そして、例えば、ストレージデバイス１０３などに、コンピュータをＦＢＰの実行環境として動作させるためのプログラムが格納され、ＦＢＰのプログラミング環境からファンクションノード１８０や永続的実行環境１９０がデプロイされると、これらもストレージデバイス１０３や内部メモリ１０２などに保存される。

[0111] そして、プロセッサ１０１がストレージデバイス１０３などから、コンピュータをＦＢＰの実行環境として動作させるためのプログラムを読み出して内部メモリ１０２上で実行すると、デPLOYされたファンクションノード１８０に対応するファンクションノード制御部１１０や、デPLOYされた永続的実行環境１９０に対応する永続的実行部１２０などが内部メモリ１０２上に生成される。このほか、デPLOYされた処理ノードがあれば、その処理ノードに対応する制御部が同様の手順で内部メモリ１０２上に生成される。また、上述のジャーナルログ記憶部１６０やスナップショット記憶部１７０は、ストレージデバイス１０３や内部メモリ１０２などの記憶装置を用いて実現される。

[0112] コンピュータをＦＢＰの実行環境として動作させるためのプログラムは、例えば、磁気ディスク（フレキシブルディスク、ハードディスクなど）、光ディスク（ＣＤ－ＲＯＭ、ＣＤ－Ｒ、ＣＤ－ＲＷ、ＤＶＤ－ＲＯＭ、ＤＶＤ±Ｒ、ＤＶＤ±ＲＷ、Ｂｌｕ－ｒａｙ（登録商標）Ｄｉｓｃなど）、半導体メモリ、またはこれに類する記録媒体に記録されて提供される。なお、プログラムを記録する記録媒体は、コンピュータが読み取り可能な記録媒体であれば、その記憶形式は何れの形態であってもよい。また、上記プログラムを、コンピュータに予めインストールするように構成してもよいし、ネットワークを介して配布される上記のプログラムをコンピュータに適宜インストールするように構成してもよい。

[0113] なお、上述した各実施形態における情報処理装置１００は、上述した機能的な構成要素の一部または全部を、ＡＳＩＣ（Application Specific Integrated Circuit）やＦＰＧＡ（Field-Programmable Gate Array）などの専用のハードウェアにより実現する構成であってもよい。

[0114] 以上、本発明の実施形態を説明したが、この実施形態は例として提示したものであり、発明の範囲を限定することは意図していない。この新規な実施形態は、その他の様々な形態で実施されることが可能であり、発明の要旨を逸脱しない範囲で、種々の省略、置き換え、変更を行うことができる。これ

ら実施形態やその変形は、発明の範囲や要旨に含まれるとともに、請求の範囲に記載された発明とその均等の範囲に含まれる。

符号の説明

- [0115] 1 0 0 情報処理装置
- 1 1 0 ファンクションノード制御部
- 1 2 0 永続的実行部
- 1 3 0 仮想マシン部
- 1 3 4 ファンクションスコープ
- 1 3 5 グローバルスコープ
- 1 3 6 変数名解決部
- 1 4 0 リカバリ部
- 1 5 0 スナップショット書き出し部
- 1 6 0 ジャーナルログ記憶部
- 1 7 0 スナップショット記憶部
- 1 8 2 プログラム
- 1 9 2 初期化プログラム
- 2 1 0 イベントリスナノード制御部
- 2 2 2 テンプレート
- 2 3 0 テンプレート処理ノード制御部
- 2 4 0 リードオンリアクセス部
- 2 6 0 リードオンリファンクションノード制御部
- 2 7 0 タグ付け部
- 2 9 0 タグ付け処理ノード制御部
- 3 0 0 再現装置
- 3 1 0 コンソール部
- 3 2 0 状態再現部
- 3 5 0 ネットワーク

請求の範囲

[請求項1] フローベースプログラミングの実行環境となる情報処理装置であ

って、

プログラミング環境からデプロイされたファンクションノードに対応して設けられるファンクションノード制御部と、

前記ファンクションノードに関連付けられて前記プログラミング環境からデプロイされた永続的実行環境に対応して設けられ、仮想マシン部とリカバリ部とを有する永続的実行部と、を備え、

前記ファンクションノード制御部は、前記ファンクションノードの入力メッセージが入力されると、該入力メッセージと前記ファンクションノードに定義されたプログラムとの組をジャーナルログとして記録するとともに、前記入力メッセージと前記プログラムとを前記仮想マシン部に送り、

前記仮想マシン部は、前記入力メッセージを引数として前記プログラムを実行し、実行結果を前記ファンクションノード制御部に送り、

前記ファンクションノード制御部は、前記仮想マシン部から受け取った前記実行結果を出力メッセージとして出力し、

前記リカバリ部は、前記情報処理装置の起動時に、前記仮想マシン部を過去のある時刻の状態に復元した後、該時刻以降に記録された前記ジャーナルログを順に取り出して、該ジャーナルログを構成する前記入力メッセージを引数とした該ジャーナルログを構成する前記プログラムの再実行を前記仮想マシン部において順次行わせることにより、前記仮想マシン部を最新の状態にロールフォワードする

情報処理装置。

[請求項2] 前記プログラミング環境からデプロイされた複数のファンクション

ノードに対応して複数の前記ファンクションノード制御部が設けられ、

前記永続的実行部は、複数の前記ファンクションノード制御部によ

り共有される

請求項 1 に記載の情報処理装置。

[請求項3] 前記仮想マシン部は、前記永続的実行環境の初期化プログラム中で宣言された変数および前記永続的実行環境の予約変数が登録されるグローバルスコープと、前記プログラムを実行する際に生成され、前記プログラム中で宣言された変数および前記ファンクションノードの予約変数が登録されるファンクションスコープとを有し、前記プログラムの実行時における変数の参照は、前記プログラム中で宣言された変数、前記ファンクションノードの予約変数、前記初期化プログラム中で宣言された変数、前記永続的実行環境の予約変数、の順に行う

請求項 1 または 2 に記載の情報処理装置。

[請求項4] 前記プログラミング環境からデプロイされたイベントリスナノードに対応して設けられるイベントリスナノード制御部をさらに備え、

前記仮想マシン部は、前記永続的実行環境の予約変数に格納されるemit関数を実行すると、該emit関数の引数であるイベント名とメッセージを前記イベントリスナノード制御部に送り、

前記イベントリスナノード制御部は、前記仮想マシン部から受け取ったイベント名が前記イベントリスナノードに定義されたイベント名と一致する場合に、前記仮想マシン部から受け取ったメッセージを出力する

請求項 3 に記載の情報処理装置。

[請求項5] 前記プログラミング環境からデプロイされたテンプレート処理ノードに対応して設けられるテンプレート処理ノード制御部と、

前記仮想マシン部の前記グローバルスコープに登録された変数の値を取得するリードオンリアクセス部と、をさらに備え、

前記テンプレート処理ノード制御部は、前記テンプレート処理ノードの入力メッセージが入力されると、前記テンプレート処理ノードに定義されたテンプレートの埋め込み箇所に指定された変数名を前記仮

想マシン部の前記グローバルスコープに登録された変数の変数名として、前記リードオンリアクセス部を通じて、前記仮想マシン部の前記グローバルスコープから値を読み出し、前記埋め込み箇所前記値を埋め込んだ前記テンプレートを出力メッセージとして出力する

請求項 3 に記載の情報処理装置。

[請求項6]

前記プログラミング環境からデプロイされたリードオンリファンクションノードに対応して設けられ、前記リードオンリファンクションノードの入力メッセージが入力されると、該入力メッセージを引数として前記リードオンリファンクションノードに定義されたプログラムを実行し、実行結果を出力メッセージとして出力するリードオンリファンクションノード制御部と、

前記仮想マシン部の前記グローバルスコープに登録された変数の値を取得するリードオンリアクセス部と、をさらに備え、

前記リードオンリファンクションノード制御部は、前記リードオンリファンクションノードに定義されたプログラムを実行する際、該プログラム内で未定義の変数からの読み出しに関しては、前記リードオンリアクセス部を通じて、前記仮想マシン部の前記グローバルスコープから値を読み出す

請求項 3 に記載の情報処理装置。

[請求項7]

請求項 1 乃至 6 のいずれか一項に記載の情報処理装置と、該情報処理装置とネットワークを介して接続される再現装置とを含む情報処理システムであって、

前記再現装置は、

過去の任意の時刻を指定するユーザ操作を受け付けるコンソール部と、

前記ユーザ操作により指定された時刻よりもさらに過去のある時刻における前記情報処理装置の前記仮想マシン部の状態をメモリ上に再現し、該時刻から前記ユーザ操作により指定された時刻までの間に記

録された前記ジャーナルログを前記情報処理装置から取得して、該ジャーナルログを構成する前記入力メッセージを引数とした該ジャーナルログを構成する前記プログラムの実行を前記メモリ上で順次行うことにより、前記ユーザ操作により指定された時刻における前記情報処理装置の前記仮想マシン部の状態を再現する状態再現部と、
を備える情報処理システム。

[請求項8] 前記情報処理装置は、前記ジャーナルログにタグを付加するタグ付け部をさらに備え、

前記再現装置の前記コンソール部は、前記タグを表示するとともに、表示した前記タグのいずれかを選択することで過去の任意の時刻を指定する前記ユーザ操作を受け付ける

請求項7に記載の情報処理システム。

[請求項9] 前記情報処理装置は、

前記プログラミング環境からデPLOYされたタグ付け処理ノードに対応して設けられ、前記タグ付け処理ノードの入力メッセージが入力されると、該入力メッセージに含まれる文字列を前記タグ付け部に送るタグ付け処理ノード制御部をさらに備え、

前記タグ付け部は、前記タグ付け処理ノード制御部から受け取った前記文字列を前記タグとして付加する

請求項8に記載の情報処理システム。

[請求項10] ネットワークを介して接続される第1情報処理装置と第2情報処理装置とを含み、前記第1情報処理装置および前記第2情報処理装置がそれぞれフローベースドプログラミングの実行環境となる情報処理システムであって、

前記第1情報処理装置は、

プログラミング環境から前記第1情報処理装置にデPLOYされたファンクションノードに対応して設けられるファンクションノード制御部を備え、

前記第2情報処理装置は、前記ファンクションノードに関連付けられて前記プログラミング環境から前記第2情報処理装置にデPLOYされた永続的実行環境に対応して設けられ、仮想マシン部とリカバリ部とを有する第1永続的実行部を備え、

前記ファンクションノード制御部は、前記ファンクションノードの入力メッセージが入力されると、該入力メッセージと前記ファンクションノードに定義されたプログラムとの組をジャーナルログとして記録するとともに、前記入力メッセージと前記プログラムとを前記仮想マシン部に送り、

前記仮想マシン部は、前記入力メッセージを引数として前記プログラムを実行し、実行結果を前記ファンクションノード制御部に送り、

前記ファンクションノード制御部は、前記仮想マシン部から受け取った前記実行結果を出力メッセージとして出力し、

前記リカバリ部は、前記第2情報処理装置の起動時に、前記仮想マシン部を過去のある時刻の状態に復元した後、該時刻以降に記録された前記ジャーナルログを順に取り出して、該ジャーナルログを構成する前記入力メッセージを引数とした該ジャーナルログを構成する前記プログラムの再実行を前記仮想マシン部において順次行わせることにより、前記仮想マシン部を最新の状態にロールフォワードする

情報処理システム。

[請求項11]

前記第1情報処理装置は、前記ファンクションノードに関連付けられて前記プログラミング環境から前記第1情報処理装置にデPLOYされた前記永続的実行環境に対応して設けられ、仮想マシン部とリカバリ部とを有する第2永続的実行部をさらに備え、

前記ファンクションノード制御部は、前記ファンクションノードの入力メッセージが入力されると、該入力メッセージと前記ファンクションノードに定義されたプログラムとの組をジャーナルログとして記録するとともに、前記入力メッセージと前記プログラムとを前記第1

永続的実行部の前記仮想マシン部と、前記第2永続的実行部の前記仮想マシン部との双方に送り、前記第1永続的実行部の前記仮想マシン部から受け取った前記実行結果と前記第2永続的実行部の前記仮想マシン部から受け取った前記実行結果とのいずれかを出力メッセージとして出力し、

前記第2永続的実行部の前記リカバリ部は、前記第1情報処理装置の起動時に、前記第2永続的実行部の前記仮想マシン部を過去のある時刻の状態に復元した後、該時刻以降に記録された前記ジャーナルログを順に取り出して、該ジャーナルログを構成する前記入力メッセージを引数とした該ジャーナルログを構成する前記プログラムの再実行を前記第2永続的実行部の前記仮想マシン部において順次行わせることにより、前記第2永続的実行部の前記仮想マシン部を最新の状態にロールフォワードする

請求項10に記載の情報処理システム。

[請求項12]

フローベースプログラミングの実行環境となる情報処理装置において実行される情報処理方法であって、

プログラミング環境から前記情報処理装置にデPLOYされたファンクションノードの入力メッセージが入力されると、該入力メッセージと前記ファンクションノードに定義されたプログラムとの組をジャーナルログとして記録するとともに、

前記ファンクションノードに関連付けられて前記プログラミング環境から前記情報処理装置にデPLOYされた永続的実行環境に対応して設けられた仮想マシン部において、前記入力メッセージを引数として前記プログラムを実行し、実行結果を出力メッセージとして出力し、

前記情報処理装置の起動時は、前記仮想マシン部を過去のある時刻の状態に復元した後、該時刻以降に記録された前記ジャーナルログを順に取り出して、該ジャーナルログを構成する前記入力メッセージを引数とした該ジャーナルログを構成する前記プログラムの再実行を前

記仮想マシン部において順次行わせることにより、前記仮想マシン部を最新の状態にロールフォワードする

情報処理方法。

[請求項13]

フローベースプログラミングの実行環境となるコンピュータに、プログラミング環境からデプロイされたファンクションノードに対応して設けられるファンクションノード制御部の機能と、

前記ファンクションノードに関連付けられて前記プログラミング環境からデプロイされた永続的実行環境に対応して設けられ、仮想マシン部とリカバリ部とを有する永続的実行部の機能と、を実現させ、

前記ファンクションノード制御部は、前記ファンクションノードの入力メッセージが入力されると、該入力メッセージと前記ファンクションノードに定義されたプログラムとの組をジャーナルログとして記録するとともに、前記入力メッセージと前記プログラムとを前記仮想マシン部に送り、

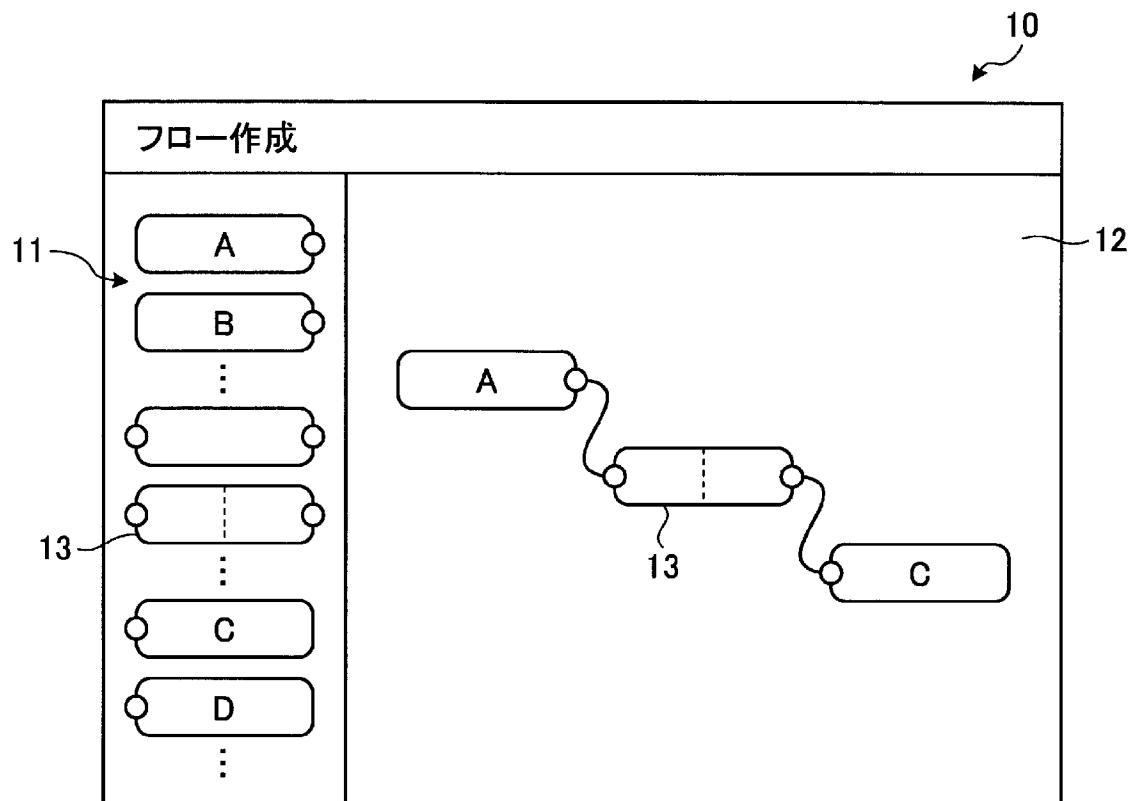
前記仮想マシン部は、前記入力メッセージを引数として前記プログラムを実行し、実行結果を前記ファンクションノード制御部に送り、

前記ファンクションノード制御部は、前記実行結果を出力メッセージとして出力し、

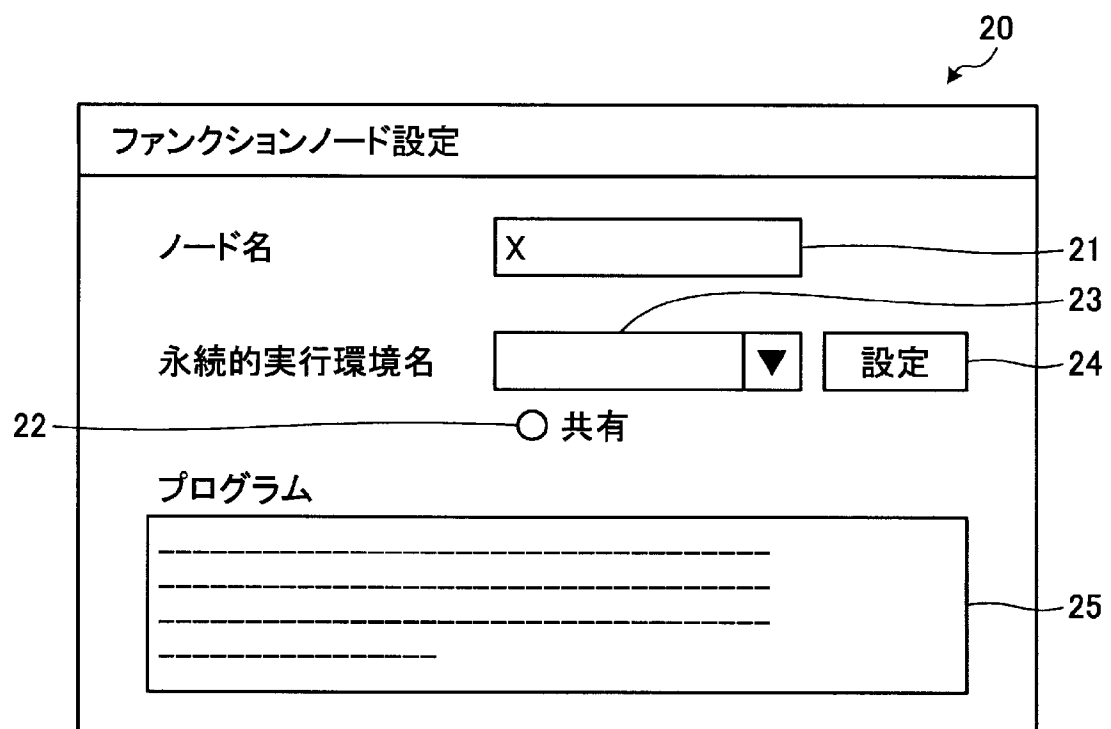
前記リカバリ部は、前記コンピュータの起動時に、前記仮想マシン部を過去のある時刻の状態に復元した後、該時刻以降に記録された前記ジャーナルログを順に取り出して、該ジャーナルログを構成する前記入力メッセージを引数とした該ジャーナルログを構成する前記プログラムの再実行を前記仮想マシン部において順次行わせることにより、前記仮想マシン部を最新の状態にロールフォワードする

プログラム。

[図1]



[図2]



[図3]

30

永続的実行環境設定

名前

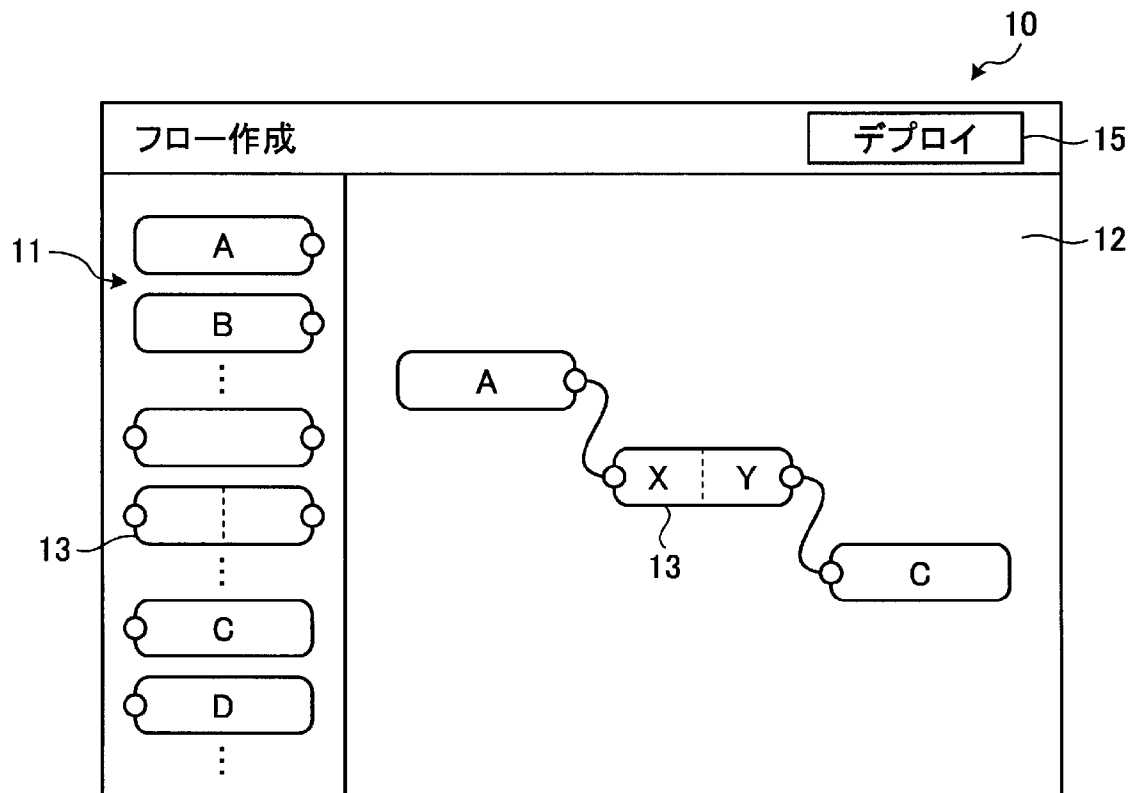
Y

31

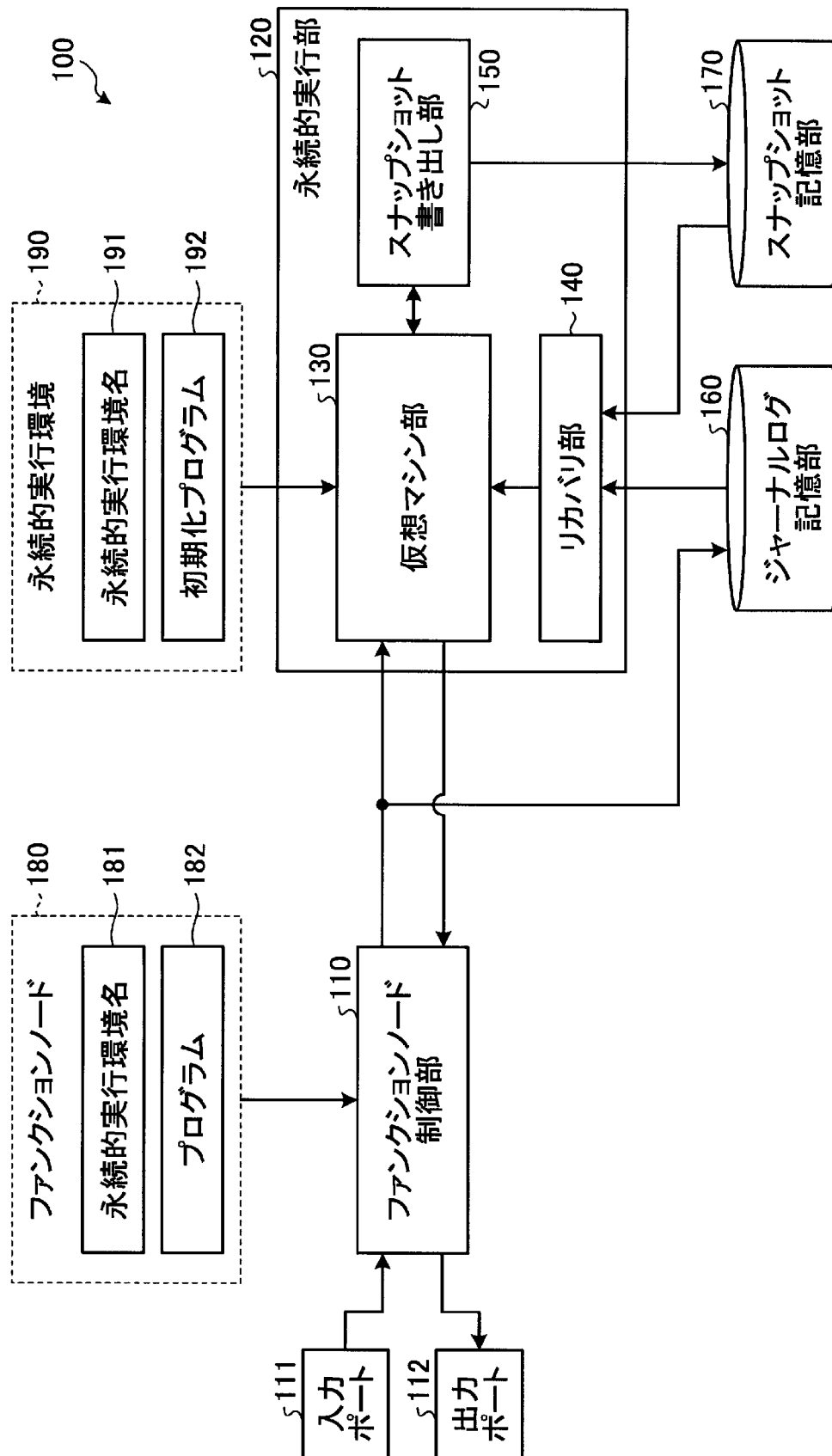
初期化プログラム

32

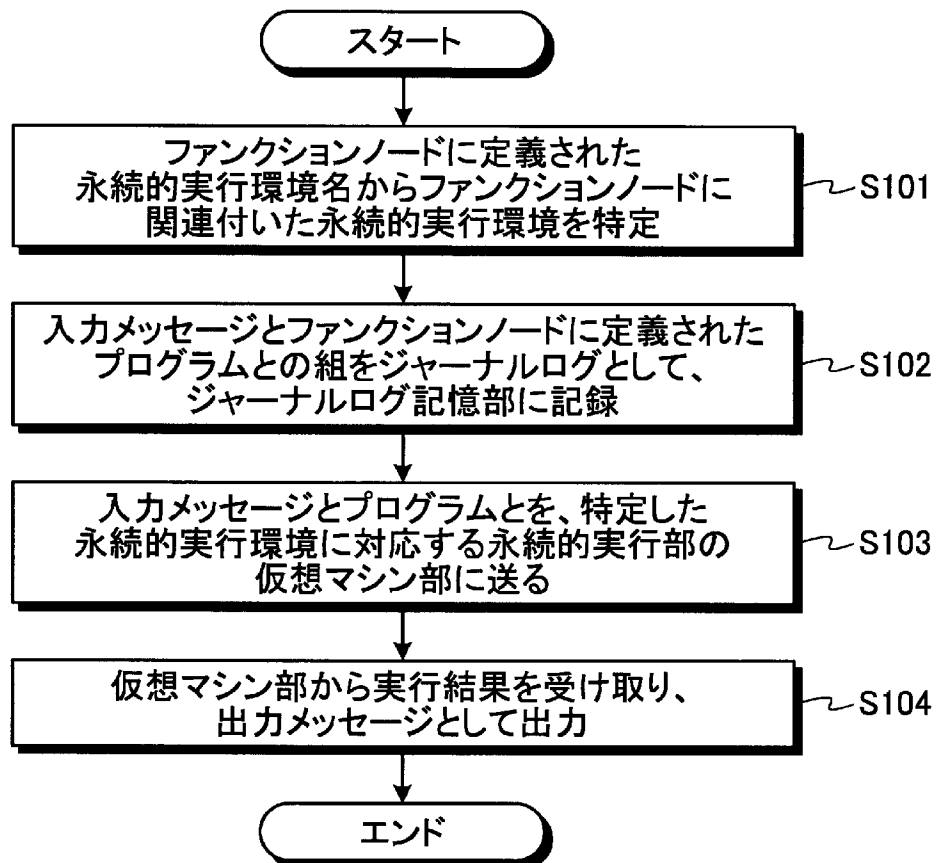
[図4]

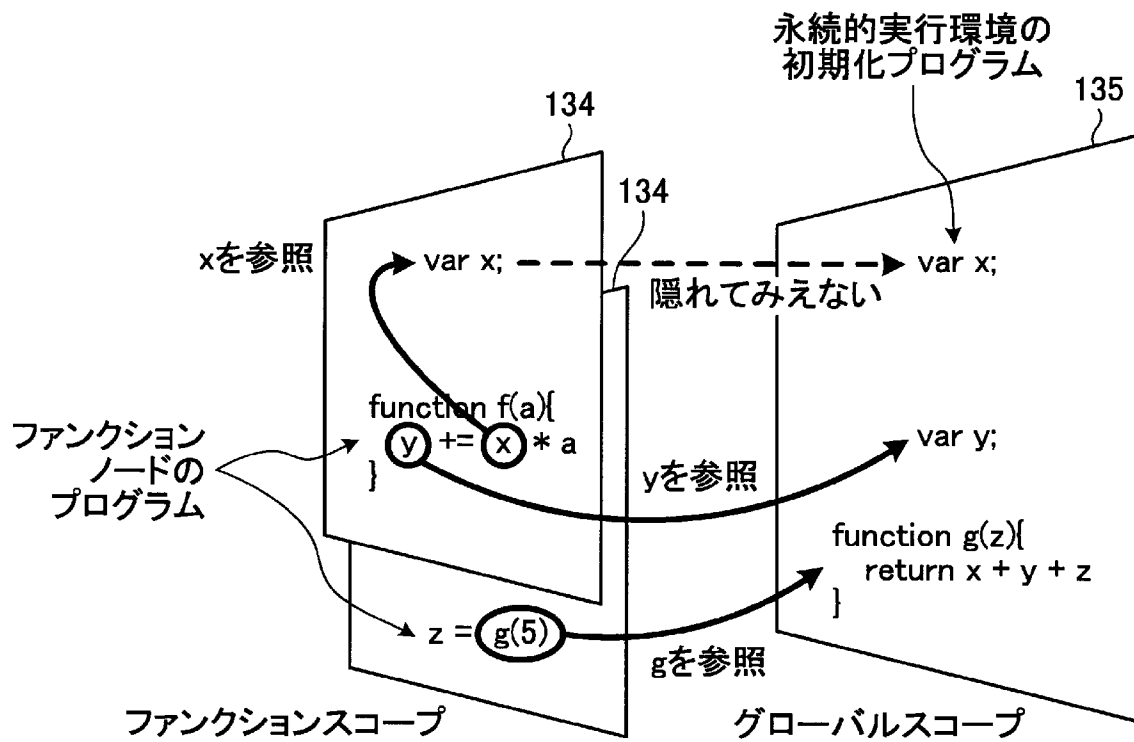
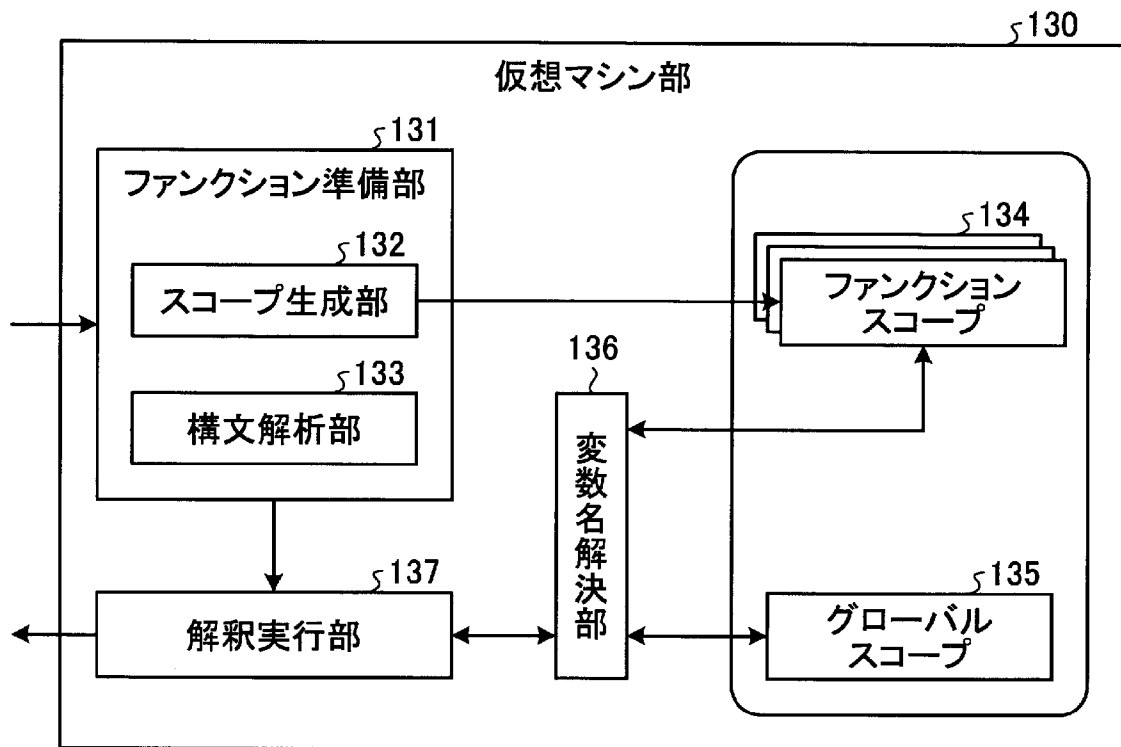


[図5]

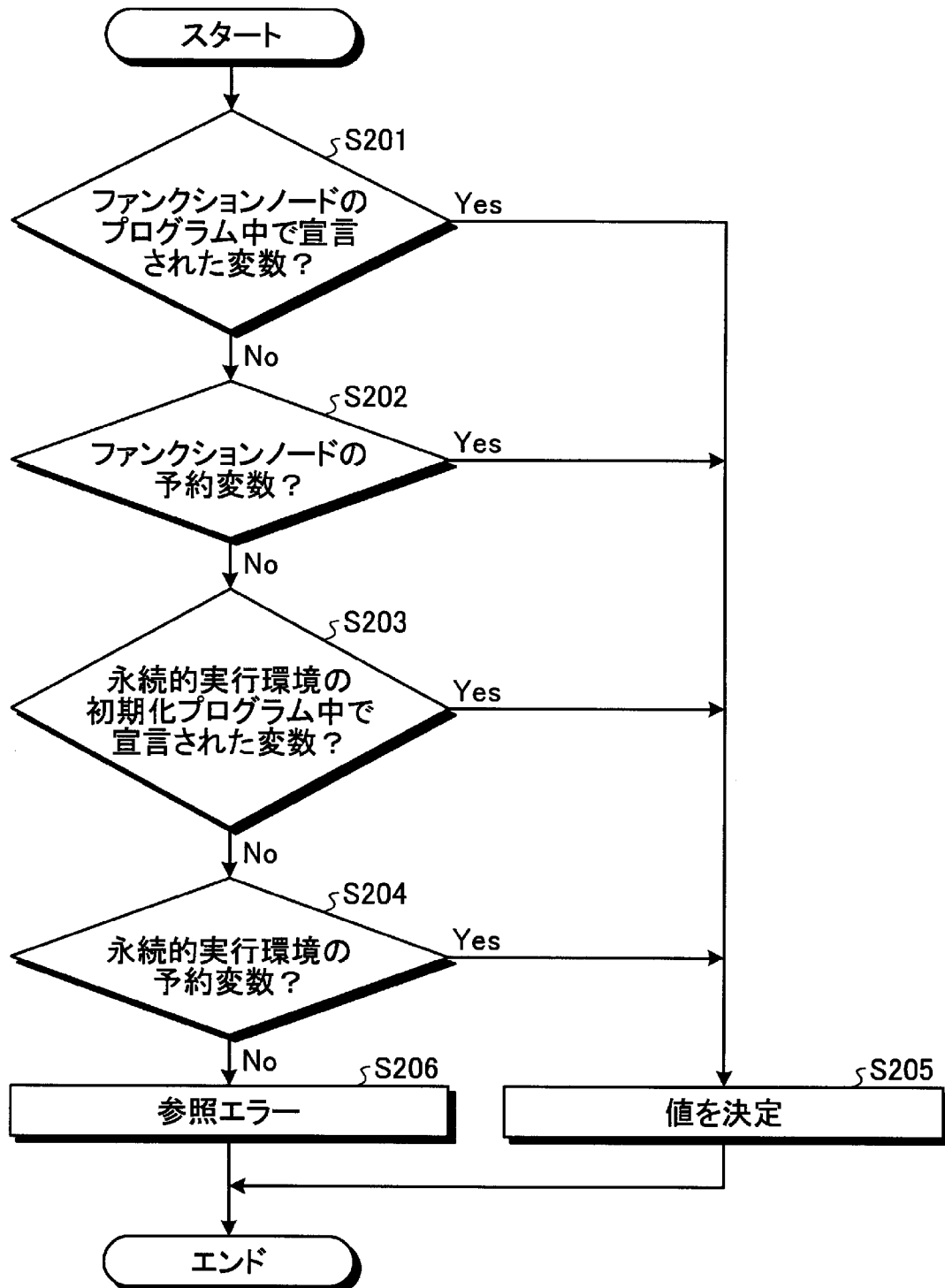


[図6]

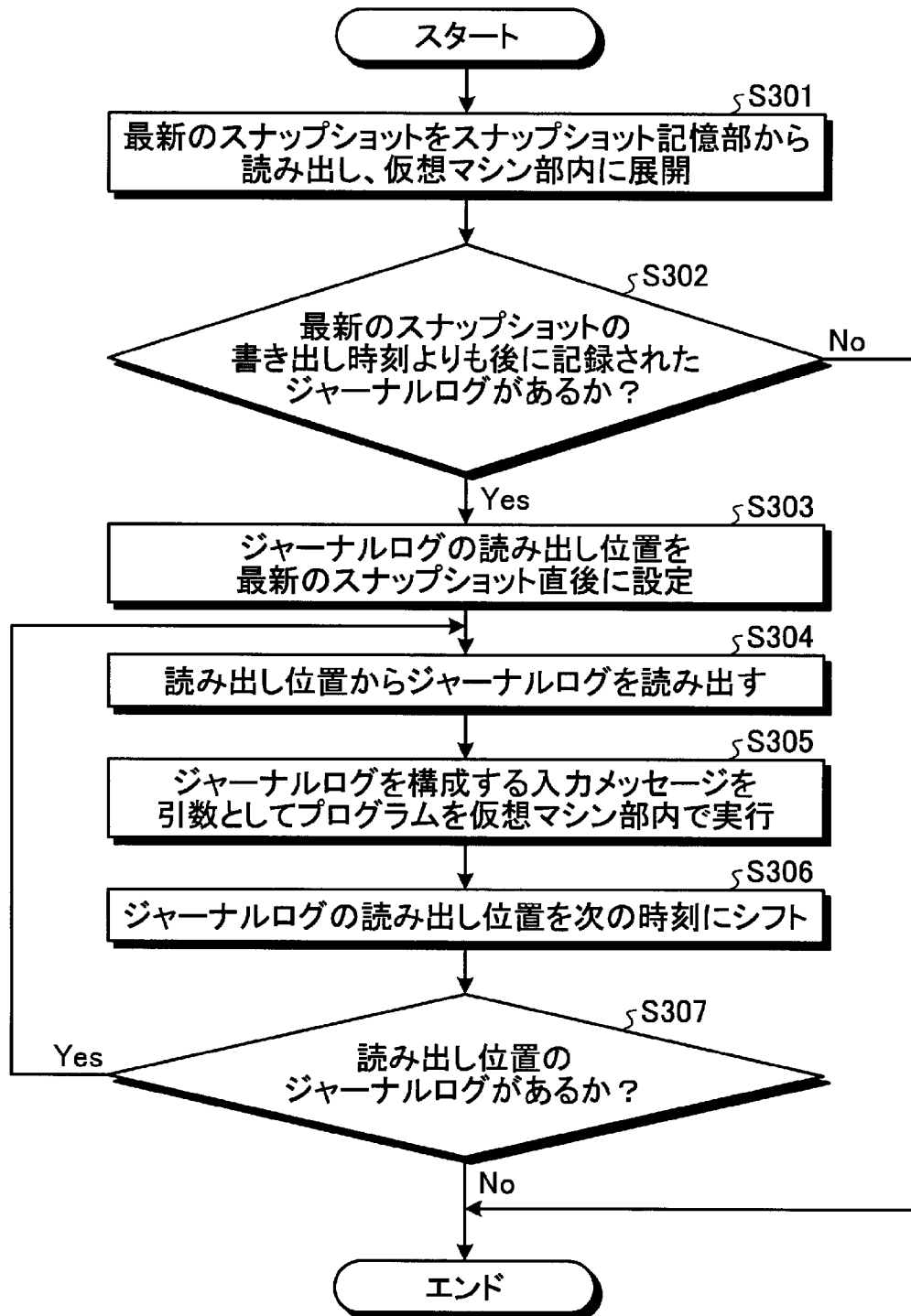




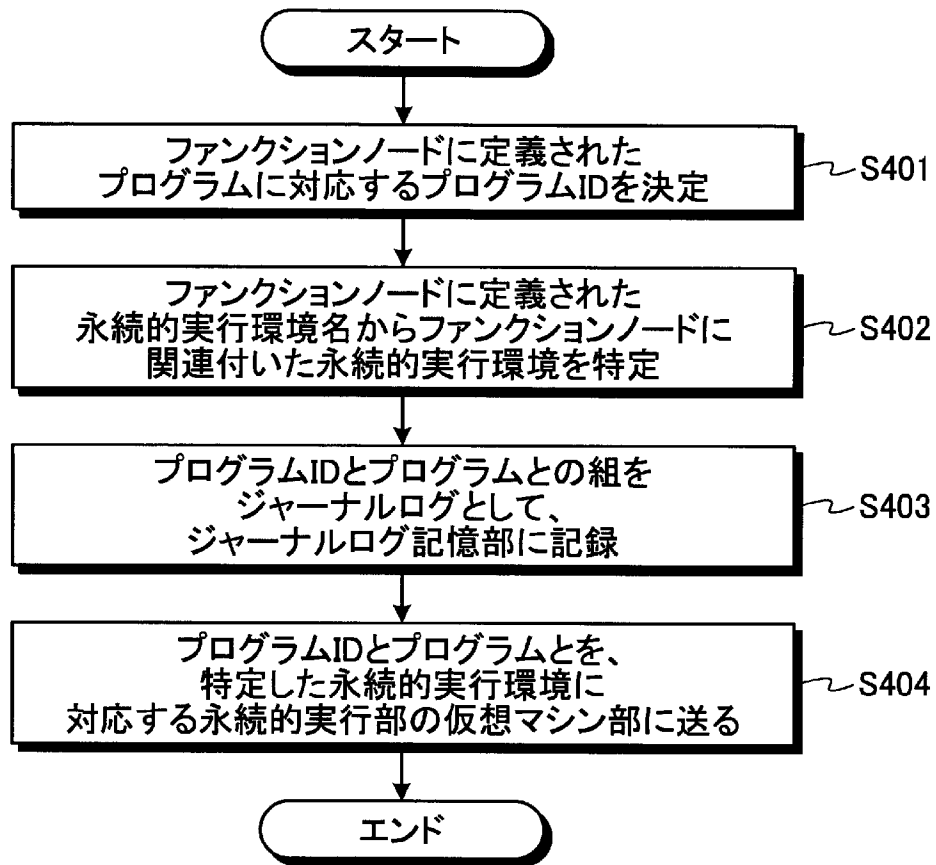
[図9]



[図10]



[図11]

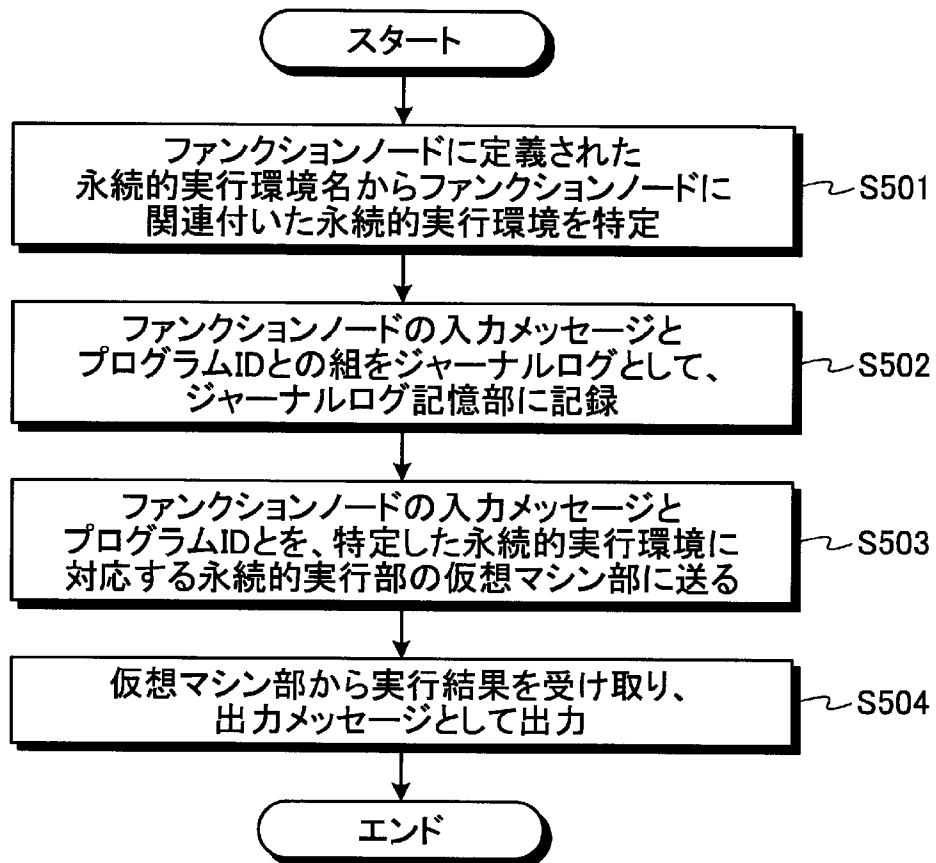


[図12]

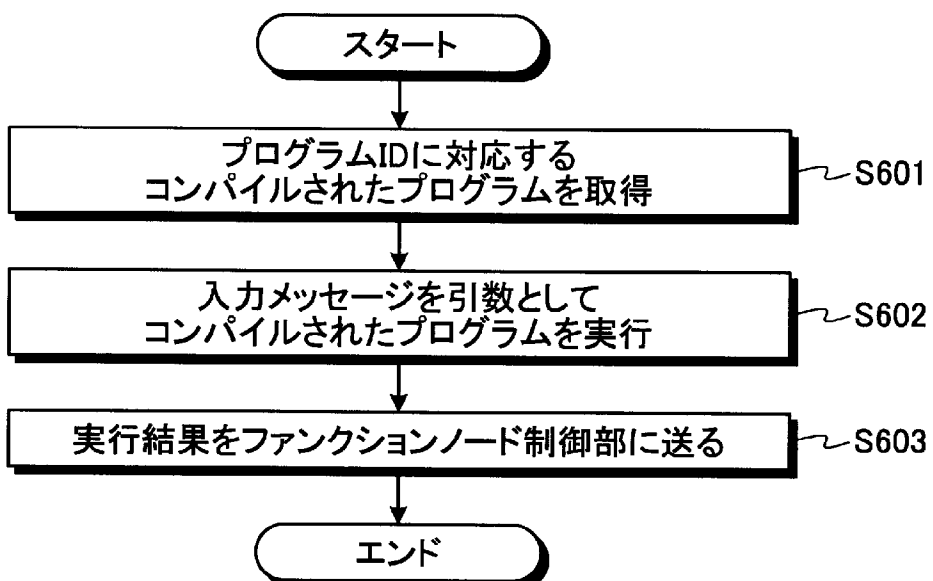
138

プログラムID	コンパイルされた プログラム
65499	function (msg, ...
74321	function (msg, ...
⋮	⋮

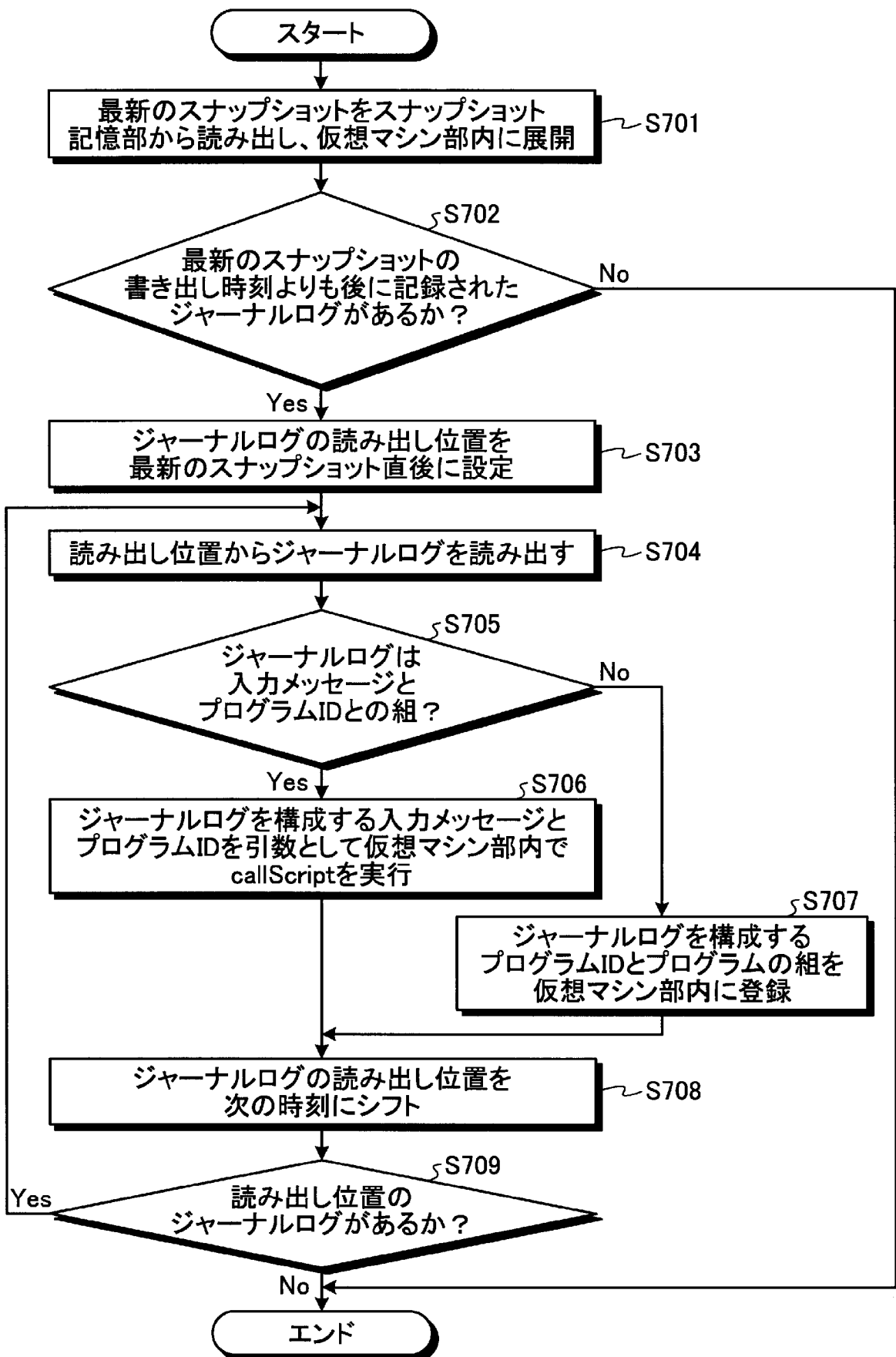
[図13]



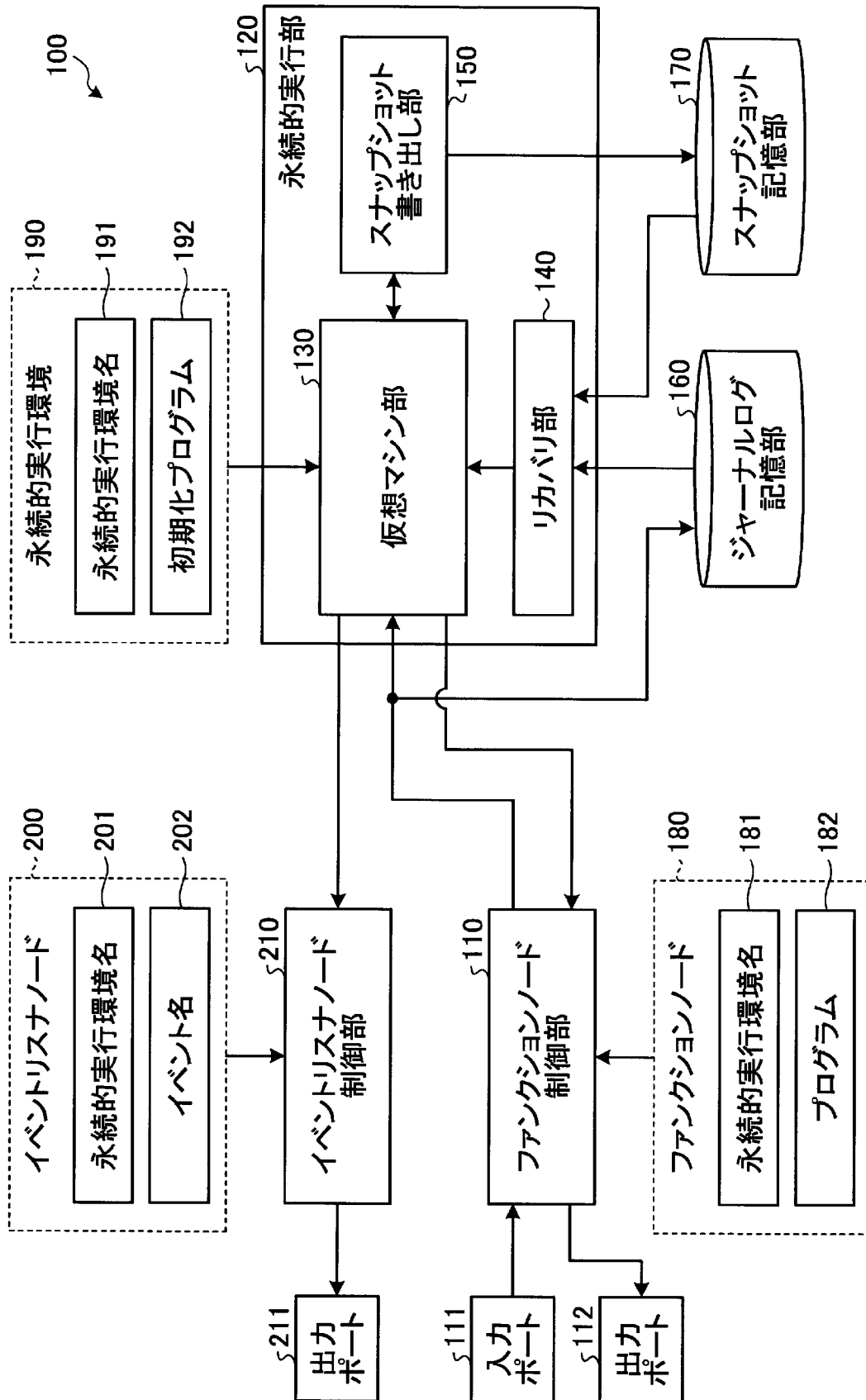
[図14]



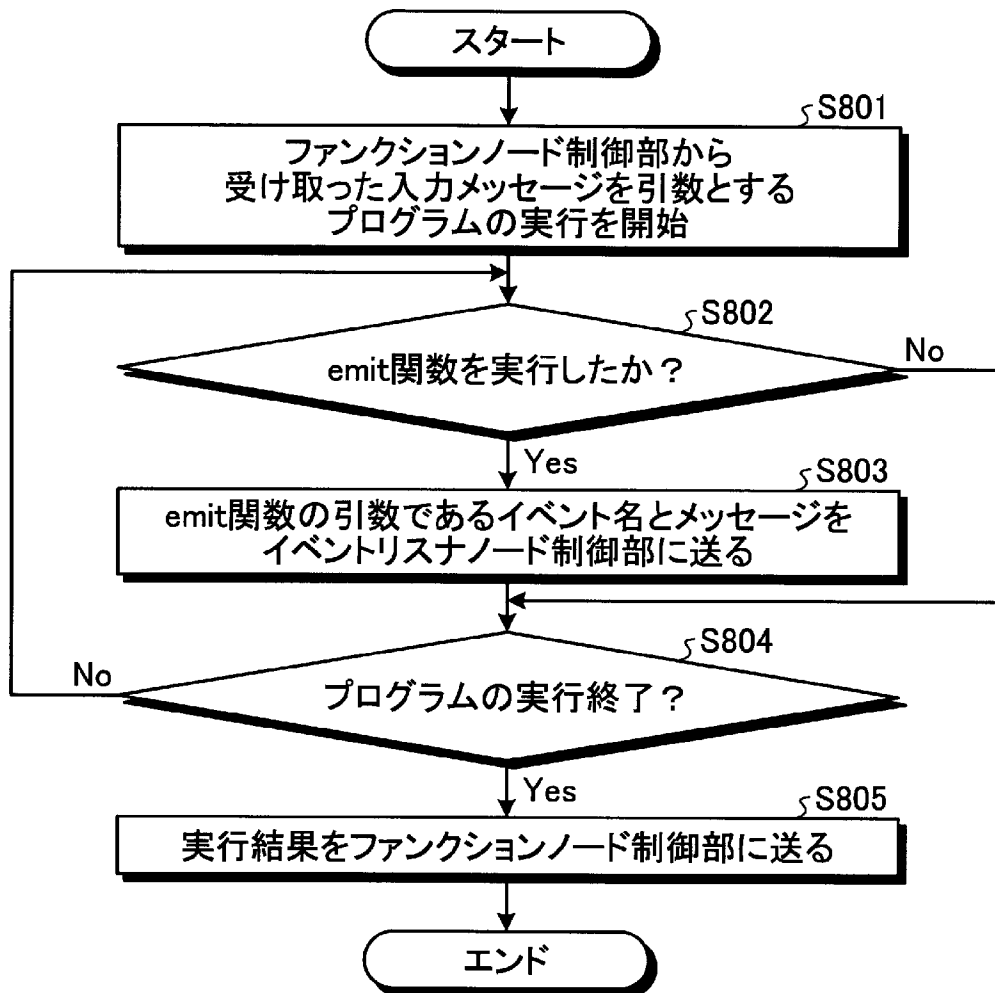
[図15]



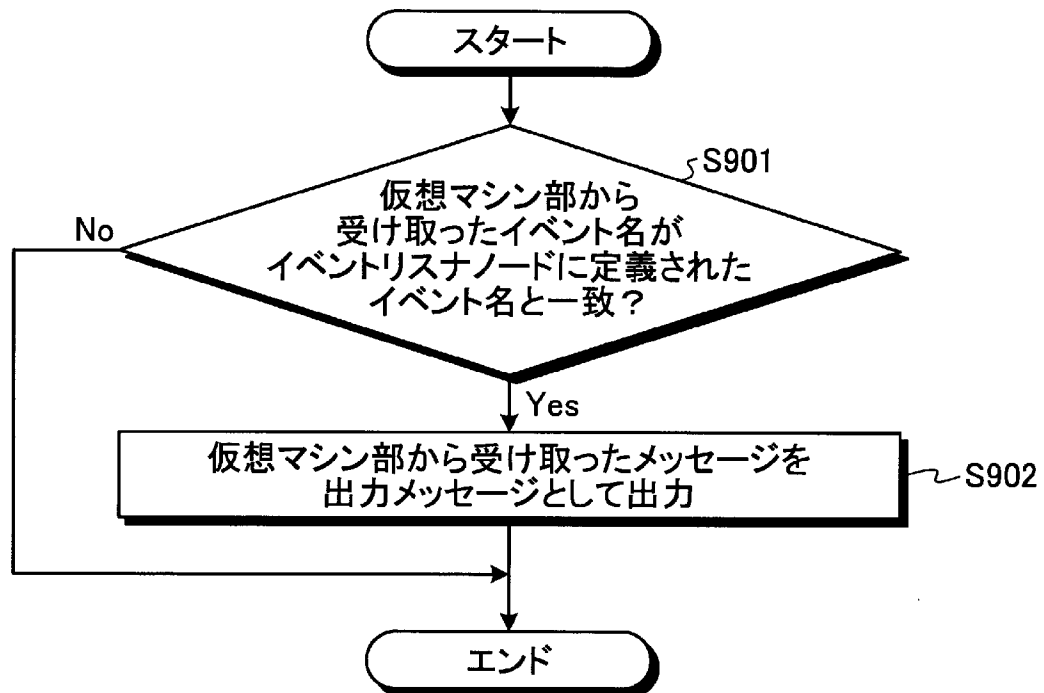
[図16]



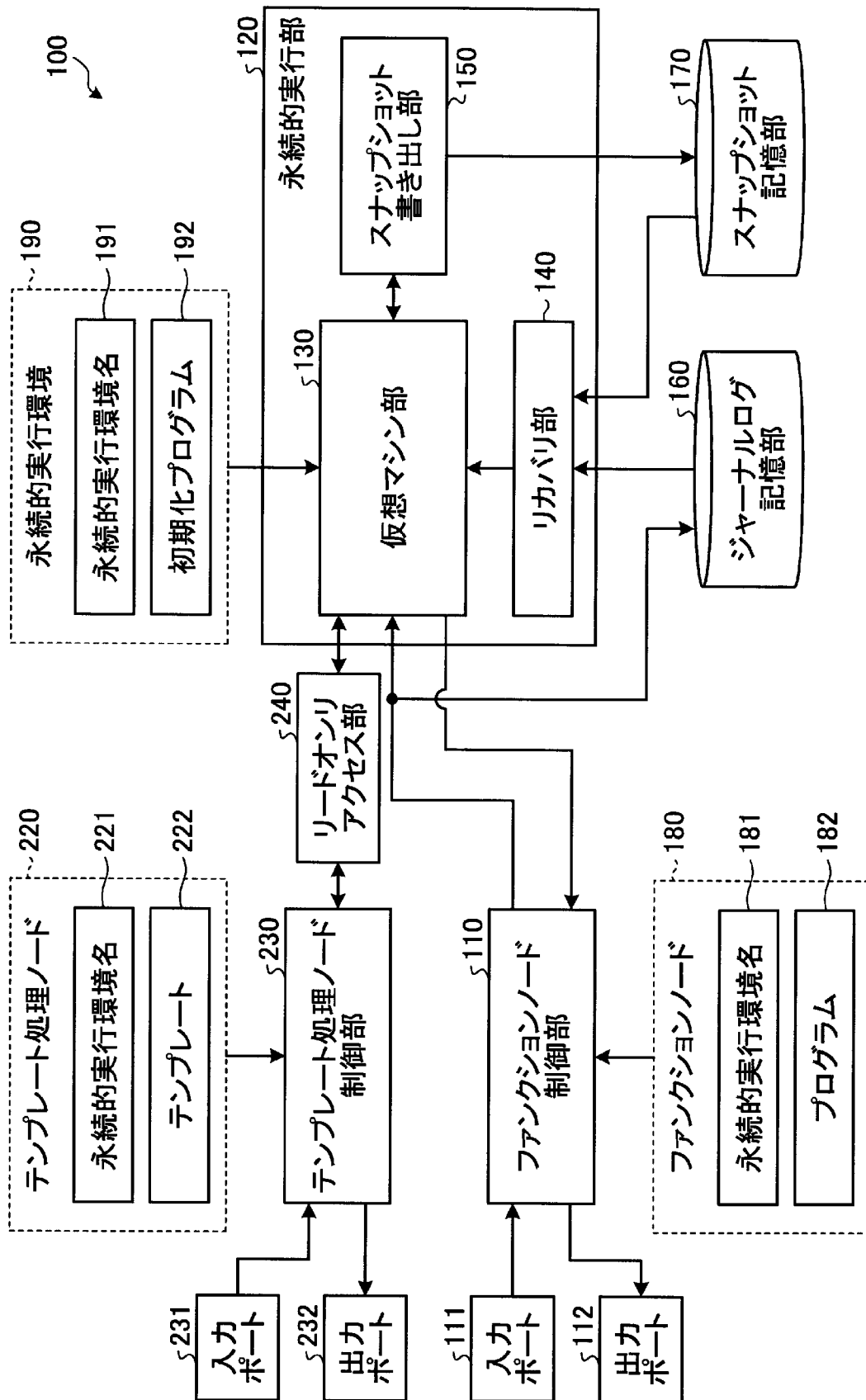
[図17]



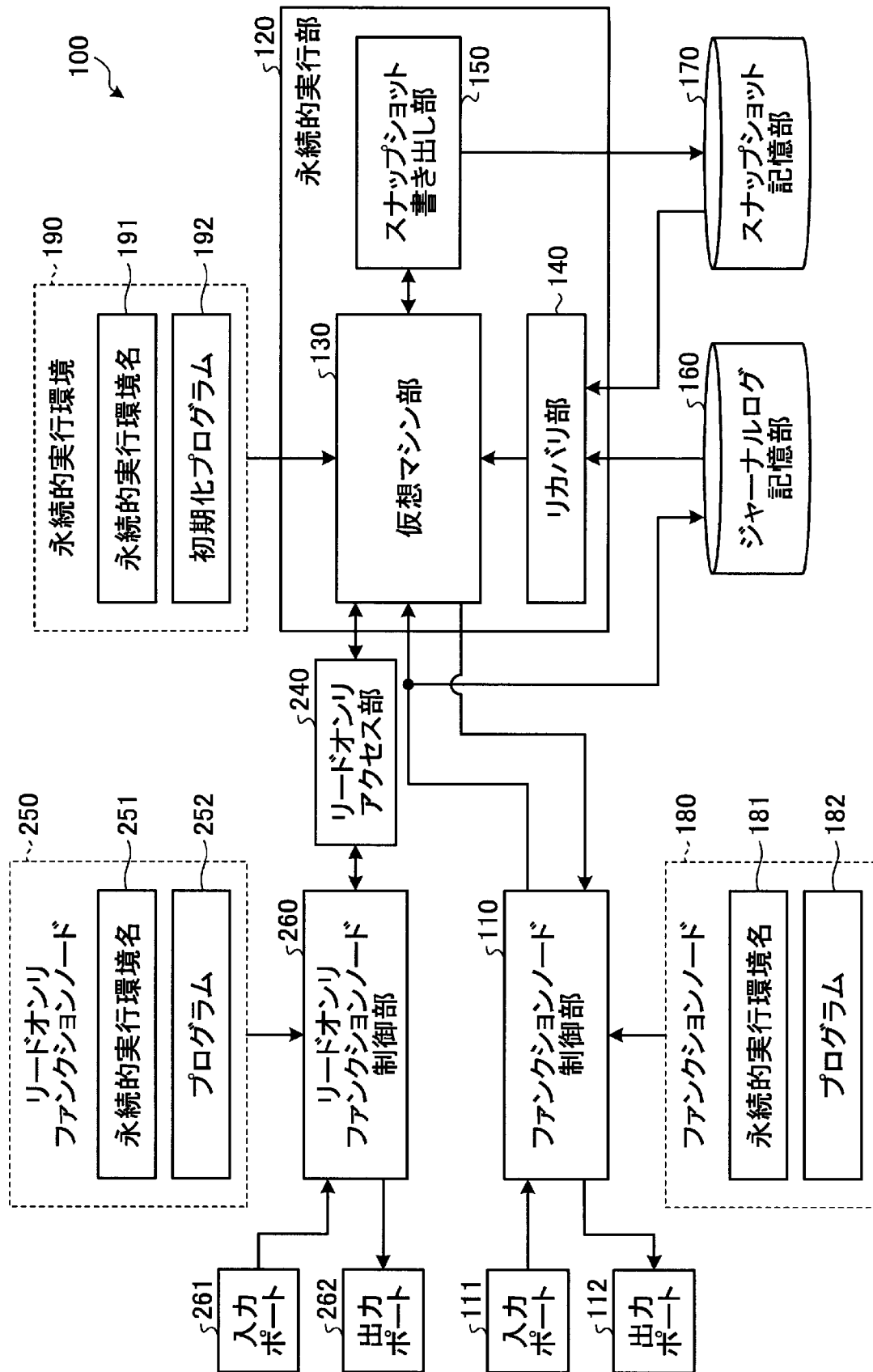
[図18]



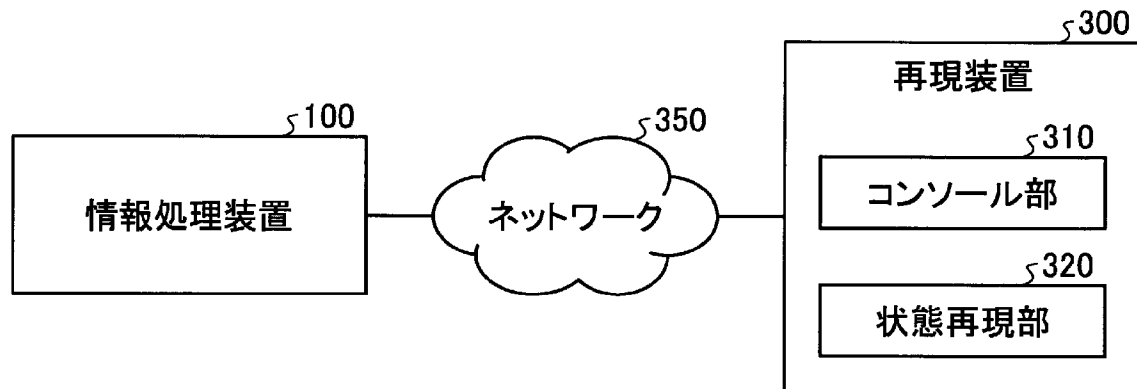
[図19]



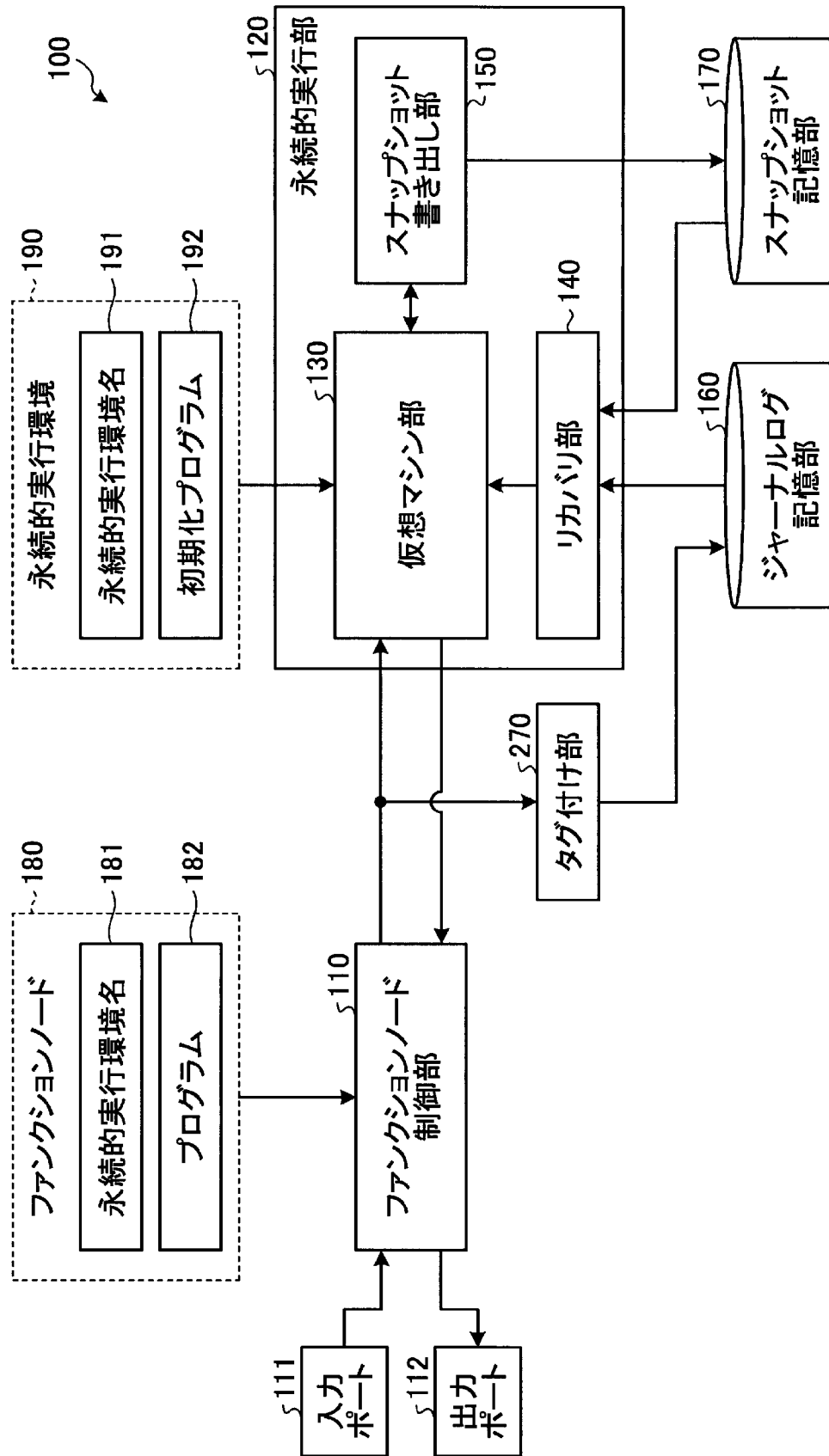
[図20]



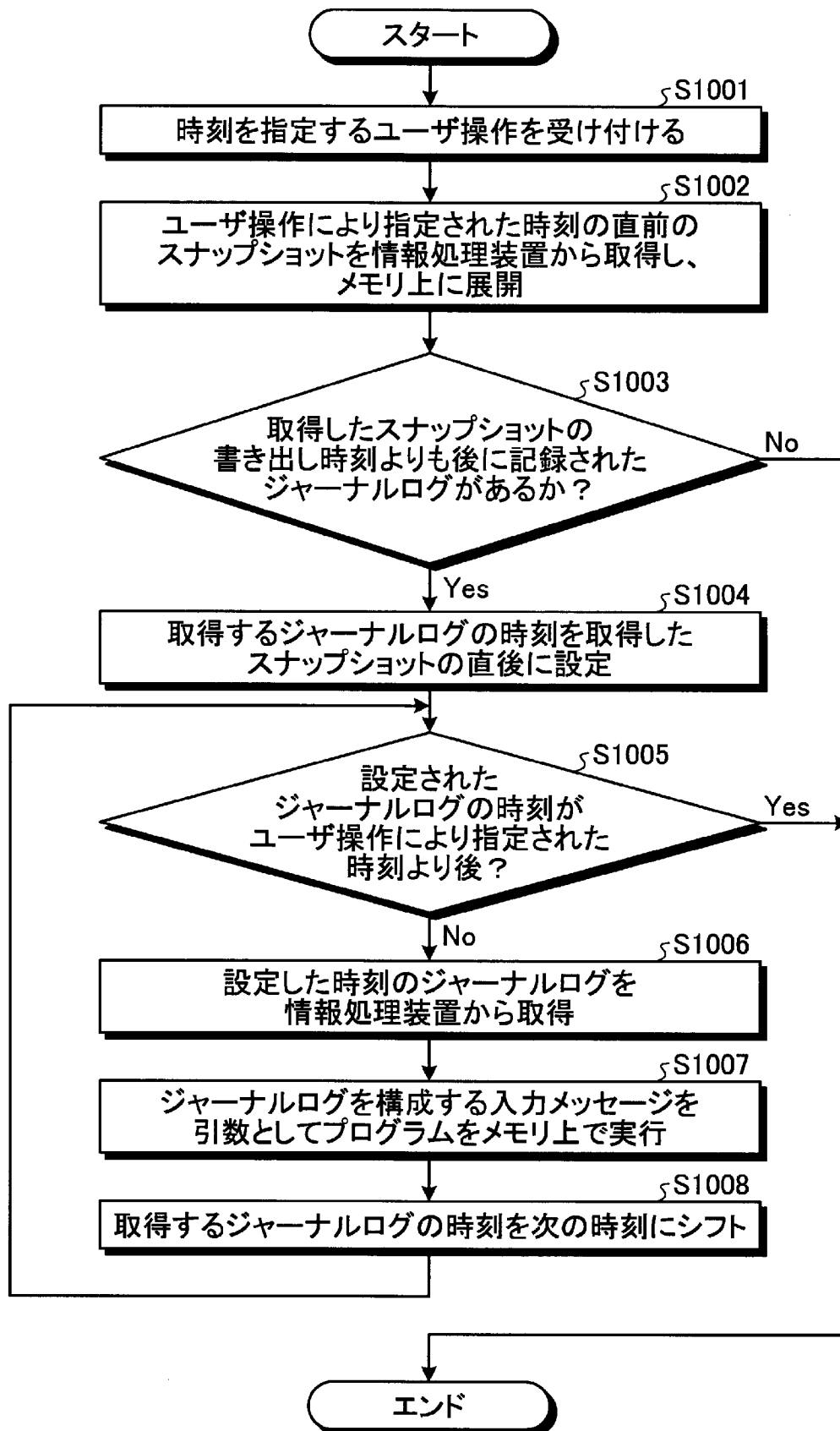
[図21]



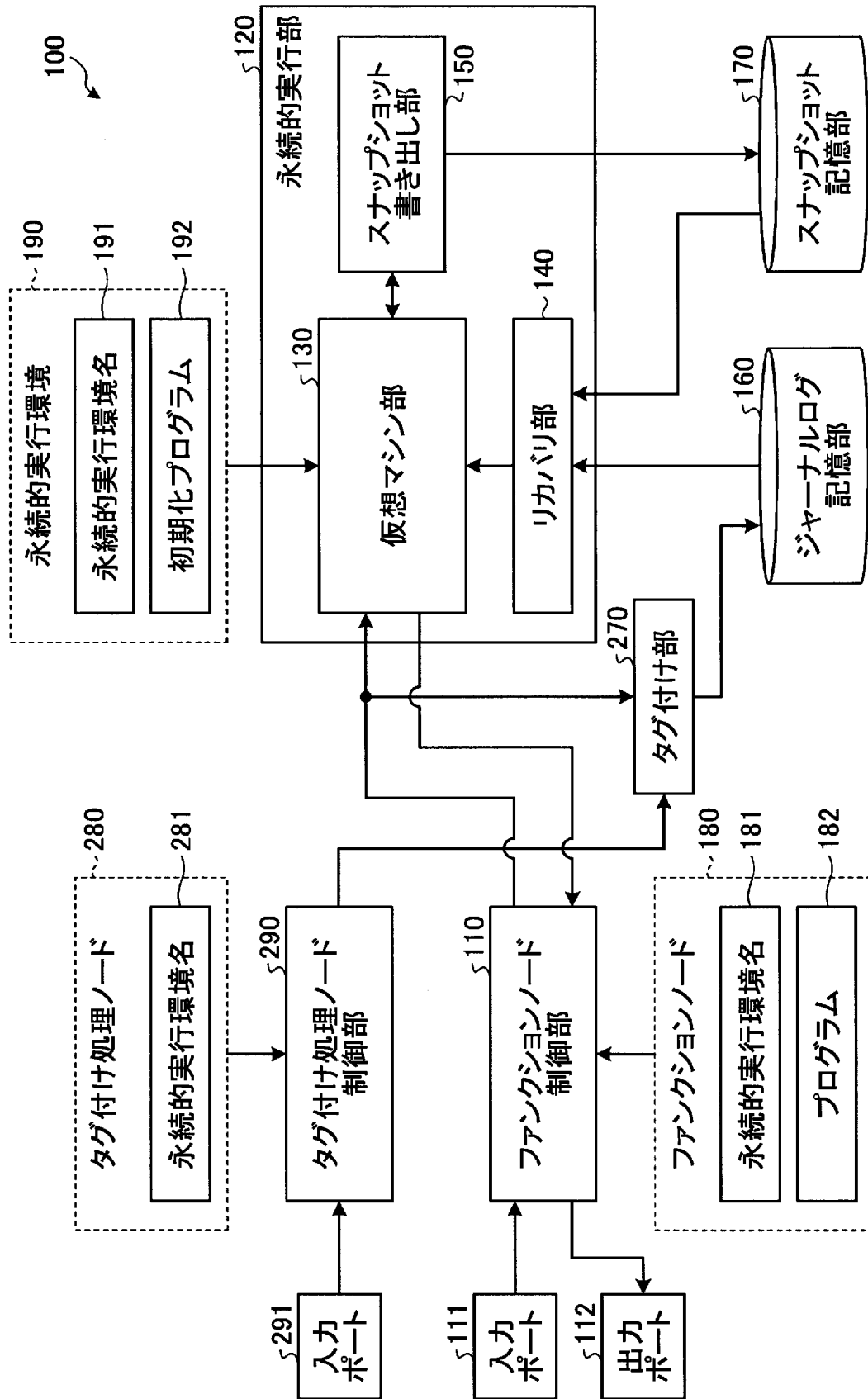
[図22]



[図23]



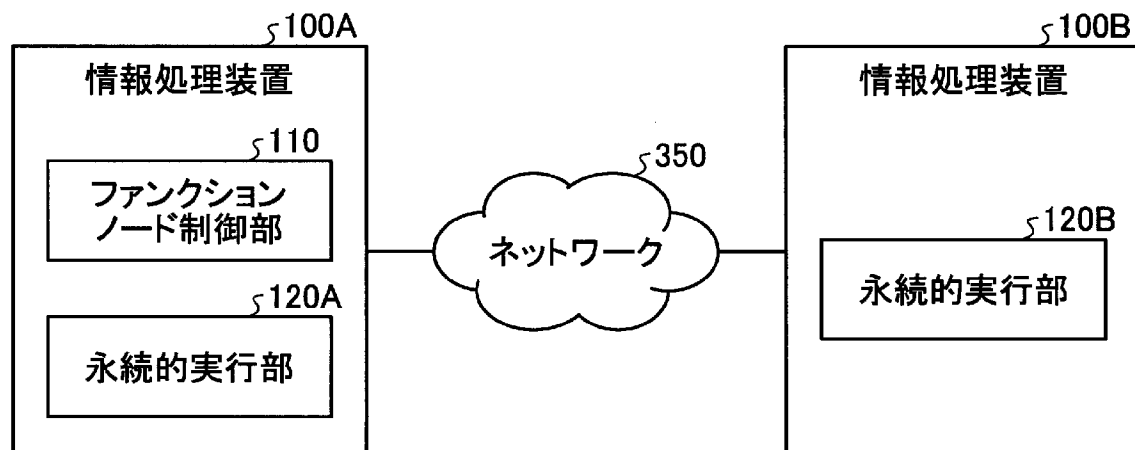
[図24]



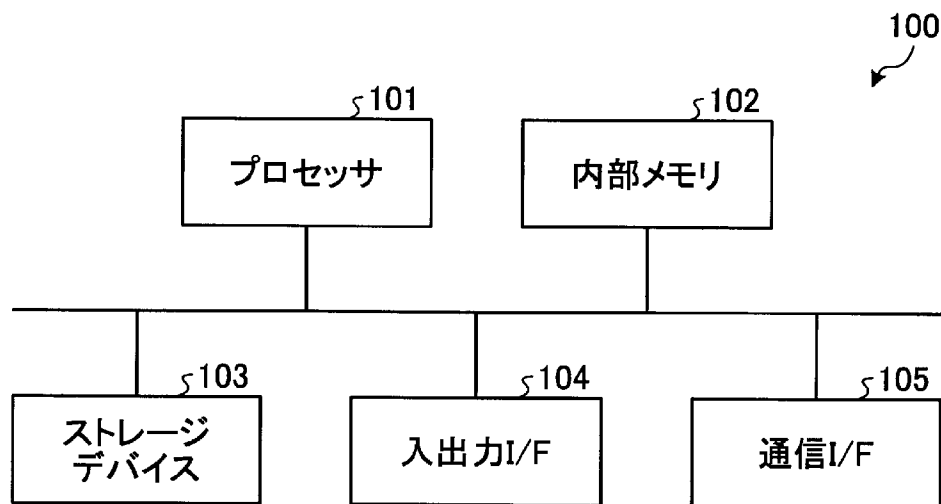
[図25]



[図26]



[図27]



INTERNATIONAL SEARCH REPORT

International application No.

PCT/JP2017/034647

A. CLASSIFICATION OF SUBJECT MATTER Int. Cl. G06F9/445 (2006.01) i, G06F9/44 (2006.01) i According to International Patent Classification (IPC) or to both national classification and IPC		
B. FIELDS SEARCHED Minimum documentation searched (classification system followed by classification symbols) Int. Cl. G06F9/445, G06F9/44, G06F9/455 Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched Published examined utility model applications of Japan 1922-1996 Published unexamined utility model applications of Japan 1971-2017 Registered utility model specifications of Japan 1996-2017 Published registered utility model applications of Japan 1994-2017 Electronic data base consulted during the international search (name of data base and, where practicable, search terms used)		
C. DOCUMENTS CONSIDERED TO BE RELEVANT		
Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
A	WO 2016/036817 A1 (AB INITIO TECHNOLOGY LLC) 10 March 2016, abstract, page 59, line 3 to page 60, line 26 & JP 2017-531851 A & CA 2959169 A1	1-13
A	JP 2009-86701 A (TOSHIBA CORP. et al.) 23 April 2009, abstract (Family: none)	1-13
☐ Further documents are listed in the continuation of Box C. ☐ See patent family annex.		
* Special categories of cited documents: "A" document defining the general state of the art which is not considered to be of particular relevance "E" earlier application or patent but published on or after the international filing date "L" document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified) "O" document referring to an oral disclosure, use, exhibition or other means "P" document published prior to the international filing date but later than the priority date claimed "T" later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention "X" document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone "Y" document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art "&" document member of the same patent family		
Date of the actual completion of the international search 04.12.2017		Date of mailing of the international search report 12.12.2017
Name and mailing address of the ISA/ Japan Patent Office 3-4-3, Kasumigaseki, Chiyoda-ku, Tokyo 100-8915, Japan		Authorized officer Telephone No.

A. 発明の属する分野の分類（国際特許分類（I P C））

Int.Cl. G06F9/445(2006.01)i, G06F9/44(2006.01)i

B. 調査を行った分野

調査を行った最小限資料（国際特許分類（I P C））

Int.Cl. G06F9/445, G06F9/44, G06F9/455

最小限資料以外の資料で調査を行った分野に含まれるもの

日本国実用新案公報	1922-1996年
日本国公開実用新案公報	1971-2017年
日本国実用新案登録公報	1996-2017年
日本国登録実用新案公報	1994-2017年

国際調査で利用した電子データベース（データベースの名称、調査に使用した用語）

C. 関連すると認められる文献

引用文献の カテゴリー*	引用文献名 及び一部の箇所が関連するときは、その関連する箇所の表示	関連する 請求項の番号
A	WO 2016/036817 A1 (AB INITIO TECHNOLOGY LLC) 2016.03.10, 要約, 第59頁第3行-第60頁第26行 & JP 2017-531851 A & CA 2959169 A1	1-13
A	JP 2009-86701 A (株式会社東芝 外1名) 2009.04.23, 要約 (ファミリーなし)	1-13

C欄の続きにも文献が列挙されている。

パテントファミリーに関する別紙を参照。

* 引用文献のカテゴリー

「A」特に関連のある文献ではなく、一般的技術水準を示すもの
「E」国際出願日前の出願または特許であるが、国際出願日後に公表されたもの
「L」優先権主張に疑義を提起する文献又は他の文献の発行日若しくは他の特別な理由を確立するために引用する文献（理由を付す）
「O」口頭による開示、使用、展示等に言及する文献
「P」国際出願日前で、かつ優先権の主張の基礎となる出願

の日の後に公表された文献

「T」国際出願日又は優先日後に公表された文献であって出願と矛盾するものではなく、発明の原理又は理論の理解のために引用するもの

「X」特に関連のある文献であって、当該文献のみで発明の新規性又は進歩性がないと考えられるもの

「Y」特に関連のある文献であって、当該文献と他の1以上の文献との、当業者にとって自明である組合せによって進歩性がないと考えられるもの

「&」同一パテントファミリー文献

国際調査を完了した日

04.12.2017

国際調査報告の発送日

12.12.2017

国際調査機関の名称及びあて先

日本国特許庁（I S A/J P）

郵便番号100-8915

東京都千代田区霞が関三丁目4番3号

特許庁審査官（権限のある職員）

坂庭 剛史

電話番号 03-3581-1101 内線 3545

5B

9288