(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
2019年7月18日(18.07.2019)

WIPO | PCT

(10) 国際公開番号
WO 2019/138570 A1

(51) 国際特許分類:
G06F 8/00 (2018.01)

(21) 国際出願番号： PCT/JP2018/000795

(22) 国際出願日： 2018年1月15日(15.01.2018)

(25) 国際出願の言語： 日本語

(26) 国際公開の言語： 日本語

(71) 出願人: 富士通株式会社(**FUJITSU LIMITED**) [JP/JP]; 〒2118588 神奈川県川崎市中原区上小田中４丁目１番１号 Kanagawa (JP).

(72) 発明者: 仲道 耕二(**NAKAMICHI, Koji**); 〒2118588 神奈川県川崎市中原区上小田中４丁目１番１号 富士通株式会社内 Kanagawa (JP). 大木 憲二(**OKI, Kenji**); 〒2118588 神奈川県川崎市中原区上小田中４丁目１番１号 富士通株式会社内 Kanagawa (JP). 小野 純嗣(**ONO, Junji**); 〒8148588 福岡県福岡市早良区百道浜２丁目２番１号 富士通九州ネットワークテクノロジーズ株式会社内 Fukuoka (JP).

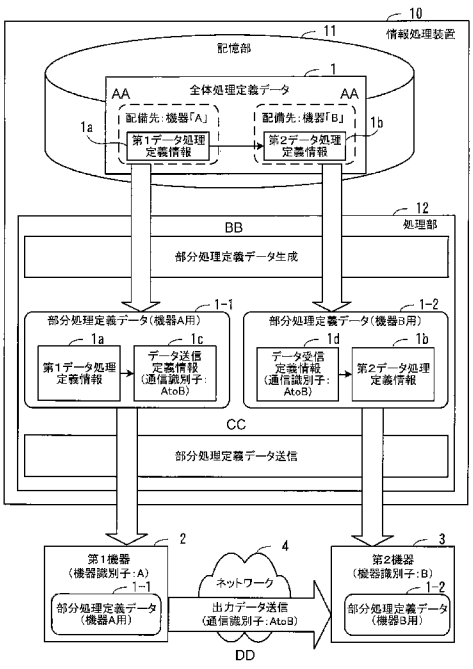
(74) 代理人: 服部 毅巖(**HATTORI, Kiyoshi**); 〒1920082 東京都八王子市東町９番８号 八王子東町センタービル 服部特許事務所 Tokyo (JP).

(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, 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, JP, KE, KG, KH,

(54) **Title:** DATA STRUCTURE OF PARTIAL PROCESS DEFINITION DATA, METHOD FOR GENERATING PARTIAL PROCESS DEFINITION DATA, AND PROGRAM FOR GENERATING PARTIAL PROCESS DEFINITION DATA

(54) 発明の名称： 部分処理定義データのデータ構造、部分処理定義データ生成方法、および部分処理定義データ生成プログラム

[図1]



1 Overall process definition data
1-1 Partial process definition data (for device A)
1-2 Partial process definition data (for device B)
1a First data process definition information
1b Second data process definition information
1c Data transmission definition information (communication identifier: AtoB)
1d Data reception definition information (communication identifier: AtoB)
2 First device (device identifier: A)
3 Second device (device identifier: B)
4 Network
10 Information processing device
11 Storage part
12 Processing part
AA Disposition: Device
BB Partial process definition data generation
CC Partial process definition data transmission
DD Output data transmission (communication identifier: AtoB)

(57) **Abstract:** An objective of the present invention is to enable automated generation of software for causing a plurality of devices to operate in coordination. From the perspective of data structure, partial process definition data (1-1) used in the execution of a first data process by a first device (2) comprises first data process definition information (1a) and

［続葉有］

WO 2019/138570 A1

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))

data transmission definition information (1c). The first data process definition information (1a) is part of overall process definition data (1) defining the execution content of the first data process and execution content of a second data process using a processing result of the first data process and defines the execution content of the first data process. The data transmission definition information (1c) defines the execution content of a process for transmitting output data, which is the result of the first data process, to a second device (3) via a network (4). The data transmission definition information (1c) comprises a communication identifier for identifying a communication between the first device (2) for executing the first data process and the second device (3) for executing the second data process.

(57) 要約:複数の機器を連係動作させるソフトウェアの自動生成を可能とする。 第１機器（２）による第１データ処理の実行に用いられる部分処理定義データ（１－１）には、データ構造上、第１データ処理定義情報（１ａ）とデータ送信定義情報（１ｃ）と、を有する。第１データ処理定義情報（１ａ）は、第１データ処理の実行内容と、第１データ処理の処理結果を用いた第２データ処理の実行内容とを定義した全体処理定義データ（１）のうちの、第１データ処理の実行内容の定義部分である。データ送信定義情報（１ｃ）は、第１データ処理の結果である出力データをネットワーク（４）経由で第２機器（３）に送信する処理の実行内容を定義している。データ送信定義情報（１ｃ）は、第１データ処理を実行させる第１機器（２）と第２データ処理を実行させる第２機器（３）との通信を識別する通信識別子を含む。

明　細　書

発明の名称 ：

部分処理定義データのデータ構造、部分処理定義データ生成方法、および部分処理定義データ生成プログラム

技術分野

[0001] 本発明は、部分処理定義データのデータ構造、部分処理定義データ生成方法、および部分処理定義データ生成プログラムに関する。

背景技術

[0002] 現在、ＩＣＴ（Information and Communication Technology）システムの大きな潮流として、ＩｏＴ（Internet of Things）が注目を浴びており、様々な分野においてＩｏＴ実現に向けた技術開発が行われ、新サービスが立ち上がってきている。ＩｏＴとは、従来はインターネットに直接接続されてこなかった機器やモノが、それ自身がインターネットに接続できるようになることである。例えば、従来から、コンピュータやモバイル端末などの人間が操作する情報通信機器はインターネットに接続されている。これらの機器に加え、ＩｏＴでは、センサや計測器機器、家電や制御機器、さらにはＲＦＩＤ（Radio Frequency IDentifier）などによってタグ付けされた物品までも、インターネットに接続される。

[0003] ＩｏＴを用いたサービスを実現するシステム（ＩｏＴシステム）は、様々な機器によって構成される分散処理システムであり、それらの機器が連携動作することでＩｏＴを用いたサービスが実現される。各機器には、その機器に実行させる処理に対応するＩｏＴアプリケーションソフトウェア（ＩｏＴアプリ）がインストールされる。ＩｏＴアプリは、ＩｏＴシステムに含まれる機器に、その機器に割り当てられたデータ処理を実行させるためのソフトウェアである。各機器が、その機器に割り当てられたデータ処理を実行することで、複数の機器による連係動作が可能となる。

[0004] このような分散処理システムで動作するＩｏＴアプリの開発は、使用する

各機器、その機器上で動作するプラットフォーム（例えばＯＳ（Operating System））、開発用のプログラミング言語など、さまざまな条件それぞれに適合する開発環境を利用して行われる。そのため様々な機器を連係動作させる場合、複数の開発環境を利用して機器ごとのＩｏＴアプリを作成することとなり、開発者の負担が大きい。

[0005] そこで、ＩｏＴの利用を促進させるためにも、ＩｏＴアプリを容易に開発できることが重要となる。単一の機器に処理を実行させるためのソフトウェアの開発ツールであれば、処理機能に対応するノードを接続したグラフから、プログラムを自動生成するものがある。このように、グラフに基づいてソフトウェアを自動生成するソフトウェア開発ツールとして、例えばＮｏｄｅ－ＲＥＤがある。

[0006] 分散処理システムでのソフトウェア開発に有用な技術としては、例えば実行モジュールを実行するノードを動的に決定することでシステム全体として効率的な処理が可能な分散システムがある。またＩｏＴに関する技術として、ネットワーク内のノードが、ネットワーク内の他のノードがスリーピ（スリープモードになり得ること）であることを決定する技術もある。

先行技術文献

特許文献

[0007] 特許文献1：国際公開第２０１６／２０７９８９号

特許文献2：国際公開第２０１５／００６６３６号

発明の概要

発明が解決しようとする課題

[0008] しかし、従来のソフトウェア自動生成可能なソフトウェア開発ツールでは、複数の機器をネットワーク経由で通信させて連係動作させる際に、各機器に実行させるソフトウェア（例えばＩｏＴアプリ）を自動生成することができない。すなわち、従来のソフトウェア自動生成可能なソフトウェア開発ツールでは、単一のデバイス上で動作するアプリを作成することが前提である

。そのため従来のソフトウェア開発ツールで生成されるアプリには、そのアプリが動作する機器から、連係動作させる他の機器を一意に識別するための情報が不足している。

[0009] このように、従来の技術では、複数の機器をネットワーク経由で通信させて連係動作させるための機器ごとのソフトウェアを自動生成することができない。その結果、アプリ開発者が、様々な機器それぞれに応じた開発ツールを用いて機器ごとのソフトウェアを作成することとなり、複数の機器のネットワークを介して連係動作させるソフトウェアの開発に手間がかかっている。

[0010] １つの側面では、本件は、複数の機器を連係動作させるソフトウェアの自動生成を可能とすることを目的とする。

課題を解決するための手段

[0011] １つの案では、以下のような部分処理定義データのデータ構造が提供される。

部分処理定義データは、データ処理定義情報とデータ送信定義情報とを有する。データ処理定義情報は、第１データ処理の実行内容と、第１データ処理の処理結果を用いた第２データ処理の実行内容とを定義した全体処理定義データのうちの、第１データ処理の実行内容の定義部分である。データ送信定義情報は、第１データ処理を実行させる第１機器と第２データ処理を実行させる第２機器との通信を識別する通信識別子を含み、第１データ処理の結果である出力データをネットワーク経由で第２機器に送信する処理の実行内容を定義したものである。そしてデータ処理定義情報は、第１機器による第１データ処理の実行に用いられる。

発明の効果

[0012] １態様によれば、複数の機器を連係動作させるソフトウェアの自動生成が可能となる。

本発明の上記および他の目的、特徴および利点は本発明の例として好ましい実施の形態を表す添付の図面と関連した以下の説明により明らかになるで

あろう。

図面の簡単な説明

[0013] [図1]第１の実施の形態の概要を示す図である。

[図2]フローベースのアプリケーション開発環境の一例を示す図である。

[図3]全体アプリフローデータに示される各ノードに対応する分配アプリの第１の配備例を示す図である。

[図4]全体アプリフローデータに示される各ノードに対応する分配アプリの第２の配備例を示す図である。

[図5]ＩｏＴシステム構成の一例を示す図である。

[図6]アプリ開発サーバのハードウェアの一構成例を示す図である。

[図7]アプリ開発サーバの機能の一例を示すブロック図である。

[図8]デバイス情報記憶部の一例を示す図である。

[図9]全体アプリフローデータ記憶部の一例を示す図である。

[図10]分配アプリの自動生成および配備処理の一例を示す図である。

[図11]通信プロトコルの決定例を示す図である。

[図12]分配アプリの生成および配備手順を示すフローチャートである。

[図13]分配アプリ生成処理の手順の一例を示すフローチャートである。

[図14]配備先機器の機能の一例を示すブロック図である。

[図15]配備先機器による処理の手順の一例を示すフローチャートである。

[図16]全体アプリフロー編集画面の一例を示す図である。

[図17]全体アプリフローデータの一例を示す図である。

[図18]配備先機器の指定例を示す図である。

[図19]「日付データ生成」機能を配備する配備先機器用の分配アプリフローデータの一例を示す図である。

[図20]「日付データ整形」機能を配備する配備先機器用の分配アプリフローデータの一例を示す図である。

[図21]「日付データ出力」機能を配備する配備先機器用の分配アプリフローデータの一例を示す図である。

[図22]表形式で表した分配アプリフローデータの一例を示す図である。

発明を実施するための形態

[0014] 以下、本実施の形態について図面を参照して説明する。なお各実施の形態は、矛盾のない範囲で複数の実施の形態を組み合わせて実施することができる。

〔第１の実施の形態〕

まず、第１の実施の形態について説明する。

[0015] 図１は、第１の実施の形態の概要を示す図である。第１の実施の形態では、ネットワーク４を介して接続されている第１機器２および第２機器３に実行させるソフトウェアを、情報処理装置１０で自動生成する。ここで、第１機器２をシステム上で識別するための機器識別子は「Ａ」、第２機器３をシステム上で識別するための機器識別子は「Ｂ」である。なお第１の実施の形態におけるソフトウェアは、単位処理の処理内容を示す定義情報を組み合わせることによって、１つの機器に実行させる処理が定義されている。

[0016] 情報処理装置１０は、記憶部１１と処理部１２とを有する。記憶部１１は、例えば情報処理装置１０が有するメモリ、またはストレージ装置である。処理部１２は、例えば情報処理装置１０が有するプロセッサ、または演算回路である。

[0017] 記憶部１１には、分散処理を行わず、１つの機器にすべての処理を実行させた場合の処理手順を定義した全体処理定義データ１を記憶する。図１の例では、全体処理定義データ１には、第１データ処理の実行内容と、第１データ処理の処理結果を用いた第２データ処理の実行内容とが定義されている。第１データ処理の実行内容は、第１データ処理定義情報１ａによって定義されている。また、第２データ処理の実行内容は、第２データ処理定義情報１ｂによって定義されている。全体処理定義データ１は、１つの機器に第１データ処理と第２データ処理とを実行させるためのソフトウェアの一例である。

[0018] 処理部１２は、全体処理定義データ１に基づいて、第１データ処理と第２

データ処理とを異なる機器で分散処理させるための、機器ごとのソフトウェアを自動生成する。

例えば処理部１２は、全体処理定義データ１から、第１データ処理の実行内容の定義部分である第１データ処理定義情報１ａを抽出する。次に処理部１２は、第１データ処理の結果である出力データをネットワーク４経由で送信する処理の実行内容を定義したデータ送信定義情報１ｃを生成する。そして処理部１２は、データ送信定義情報１ｃに、第１データ処理を実行させる第１機器２と第２データ処理を実行させる第２機器３との通信を識別する通信識別子を設定する。処理部１２は、例えば第１機器２を識別する機器識別子「Ａ」と、第２機器３を識別する機器識別子「Ｂ」とを含む情報を、通信識別子として生成し、データ送信定義情報１ｃに設定する。そして処理部１２は、第１データ処理定義情報１ａと、通信識別子が設定されたデータ送信定義情報１ｃとを含む送信側の部分処理定義データ１－１を生成する。例えば処理部１２は、部分処理定義データ１－１を含むファイルを生成し、メモリまたはストレージ装置に格納する。部分処理定義データ１－１は、第１機器２に第１データ処理を実行させるためのソフトウェアの一例である。

[0019] このようにして生成された部分処理定義データ１－１は、第１機器２による第１データ処理の実行に用いられるものであり、第１データ処理定義情報１ａと、データ送信定義情報１ｃとを有するデータ構造である。第１データ処理定義情報１ａは、第１データ処理の実行内容と、第１データ処理の処理結果を用いた第２データ処理の実行内容とを定義した全体処理定義データ１のうちの、第１データ処理の実行内容の定義部分である。またデータ送信定義情報１ｃは、第１データ処理の結果である出力データをネットワーク４経由で第２機器３に送信する処理の実行内容を定義している。データ送信定義情報１ｃは、第１データ処理を実行させる第１機器２と第２データ処理を実行させる第２機器３との通信を識別する通信識別子を含む。

[0020] また処理部１２は、全体処理定義データ１から、第２データ処理の実行内容の定義部分である第２データ処理定義情報１ｂを抽出する。次に処理部１

２は、第１データ処理の結果である出力データをネットワーク４経由で受信する処理の実行内容を定義したデータ受信定義情報１ｄに、データ送信定義情報１ｃに設定したものと同じ通信識別子を設定する。そして処理部１２は、第２データ処理定義情報１ｂと、通信識別子が設定されたデータ受信定義情報１ｄとを含む受信側の部分処理定義データ１－２を生成する。例えば処理部１２は、部分処理定義データ１－２を含むファイルを生成し、メモリまたはストレージ装置に格納する。処理定義データ１－２は、第２機器３に第２データ処理を実行させるためのソフトウェアの一例である。

[0021] このようにして生成された部分処理定義データ１－２は、第２機器３による第２データ処理の実行に用いられるものであり、第２データ処理定義情報１ｂと、データ受信定義情報１ｄとを有するデータ構造である。第２データ処理定義情報１ｂは、第１データ処理の実行内容と、第１データ処理の処理結果を用いた第２データ処理の実行内容とを定義した全体処理定義データ１のうちの、第２データ処理の実行内容の定義部分である。またデータ受信定義情報１ｄは、第１データ処理の結果である出力データをネットワーク４経由で第２機器３が受信する処理の実行内容を定義している。データ受信定義情報１ｄは、第１データ処理を実行させる第１機器２と第２データ処理を実行させる第２機器３との通信を識別する通信識別子を含む。

[0022] そして処理部１２は、送信側の部分処理定義データ１－１を第１機器２に送信し、受信側の部分処理定義データ１－２を第２機器３に送信する。例えば処理部１２は、部分処理定義データ１－１，１－２それぞれを含むファイルを、第１機器２と第２機器３とに送信する。

[0023] これにより、第１機器２は、部分処理定義データ１－１に基づいて、第１機器２に割り当てられた処理を実行することができる。また第２機器３は、部分処理定義データ１－２に基づいて、第２機器３に割り当てられた処理を実行することができる。例えば第１機器２は、第１データ処理定義情報１ａに従って第１データ処理を実行し、出力データを生成する。さらに第１機器２は、データ送信定義情報１ｃに従って、通信識別子によって通信路を一意

に特定して、ネットワーク４を介して、出力データを第２機器３に送信する。第２機器３は、データ受信定義情報１ｄに従って、通信識別子によって通信路を一意に特定して、第１機器２から出力データを受信する。そして第２機器３は、第２データ処理定義情報１ｂに従って、受信した出力データを用いた第２データ処理を実行する。

[0024] このように、第１の実施の形態では、部分処理定義データ１－１，１－２が、データ構造上、通信識別子を有する。そのため、部分処理定義データ１－１，１－２を用いて第１機器２と第２機器３が処理を実行することで、第１機器２と第２機器３とは、通信相手を一意に識別した連携処理が可能となる。しかも部分処理定義データ１－１，１－２は、全体処理定義データ１に基づいて自動生成されている。すなわち、第１の実施の形態により、第１機器２と第２機器３とを連携動作させるソフトウェアの自動生成が可能となっている。

[0025] なお処理部１２は、各データ処理を実行させる機器を比較することで、第１データ処理と第２データ処理とを特定することができる。例えば処理部１２は、まず全体処理定義データ１に定義された複数のデータ処理の中から判断対象データ処理を選択する。次に処理部１２は、複数のデータ処理のそれぞれを実行する機器を示す情報に基づいて、判断対象データ処理を実行する機器と、判断対象データ処理の処理結果を用いた他のデータ処理を実行する機器とが同じか否かを判断する。配備先の機器が異なる場合、処理部１２は、判断対象データ処理を第１データ処理と判断し、判断対象データ処理の処理結果を用いた他のデータ処理を第２データ処理と判断する。このようにして、異なる機器間の通信が発生する部分を特定することができる。その結果、データ送信定義情報１ｃやデータ受信定義情報１ｄを適切に追加することができ、信頼性の高い分散処理が可能となる。

[0026] また処理部１２は、機器間の通信に使用する通信プロトコルを、通信に使用されるネットワーク４の種別に応じて決定してもよい。例えば処理部１２は、第１データ処理を実行する第１機器２と第２データ処理を実行する第２

機器３とを接続するネットワーク４の種別に基づいて、出力データを送信する際の通信プロトコルを決定する。例えばネットワーク４が、広域ネットワークのような外部ネットワークであれば、処理部１２は、安全性の高い通信プロトコルで送信することを決定する。またネットワーク４が、例えば企業内のローカルネットワークであれば、処理部１２は、効率的な処理が可能な通信プロトコルで送信することを決定する。そして処理部１２は、決定した通信プロトコルで出力データを送信するデータ送信定義情報１ｃを含む部分処理定義データ１－１を生成する。また処理部１２は、決定した通信プロトコルで出力データを受信するデータ受信定義情報１ｄを含む部分処理定義データ１－２を生成する。このように、適切な通信プロトコルで機器間の通信を行わせることで、データの安全性を維持したまま処理の効率化を図ることができる。

[0027] 〔第２の実施の形態〕

次に第２の実施の形態について説明する。第２の実施の形態は、分散機器環境上で動作するＩｏＴアプリの生成および配備を、フローベースのアプリケーション開発環境を用いて容易に実現できるようにしたものである。なおＩｏＴアプリを配備するとは、ＩｏＴアプリを機器に送信し、ＩｏＴアプリを機器にインストールし、機器にＩｏＴアプリの実行を開始させることである。ＩｏＴアプリには、例えばアプリレイヤ上のデータ処理、およびサービス提供処理を、配備先の機器に実行させるための命令が記述されている。

[0028] 図２は、フローベースのアプリケーション開発環境の一例を示す図である。フローベースのアプリケーション開発環境を有するアプリ開発サーバは、全体アプリフロー編集画面９１を表示する。全体アプリフロー編集画面９１内には、機能ブロック表示領域９１ａとフロー表示領域９１ｂとが設けられている。

[0029] 機能ブロック表示領域９１ａには、アプリケーション開発環境内に予め用意された機能に対応するオブジェクト（機能ブロック９２ａ，９２ｂ，・・・）が表示されている。フロー表示領域９１ｂは、ＩｏＴシステムに実行さ

せる手順を示す全体アプリフロー９３を作成するための編集領域である。全体アプリフロー９３は、機能に対応するノード９３ａ，９３ｂと、ノード９３ａ，９３ｂ間を接続する線により、ＩｏＴシステムに処理を実行させるための機能と、処理の手順とが表されている。ノード９３ａ，９３ｂは、機能ブロックに対して、その機能ブロック対応する機能に実行させる処理の定義情報（パラメータなど）を付加したものである。

[0030] ユーザは、アプリ開発サーバに対して、機能ブロック表示領域９１ａから、ＩｏＴシステムに実行させる機能に対応する機能ブロックを選択し、フロー表示領域９１ｂ内に、全体アプリフロー９３のノード９３ａ，９３ｂとして配置する操作を行う。ユーザは、配置したノード９３ａ，９３ｂに、例えばパラメータを設定することで、ノード９３ａ，９３ｂに対応する機能に実行させる処理を指定する。

[0031] さらにユーザは、アプリ開発サーバに対して、先に実行させる機能を示すノードの右側と、次に実行させる機能を示すノードの左側とを線で接続する操作を行う。図２の例では、「機能α」に対応するノード９３ａの右側と「機能β」に対応するノード９３ｂの左側とが接続されている。これは、「機能α」による処理の実行結果を、「機能β」による処理の入力とすることを表している。

[0032] アプリ開発サーバは、全体アプリフロー編集画面９１上で全体アプリフロー９３が作成されると、全体アプリフロー９３を表す全体アプリフローデータ９４を作成する。全体アプリフローデータ９４は、例えばＪＳＯＮ（JavaScript（登録商標） Object Notation）によって記述される。全体アプリフローデータ９４には、ノード９３ａ，９３ｂに応じて機器に実行させる処理を定義する情報（データ処理定義情報９４ａ，９４ｂ）が記述されている。

[0033] また、ノード９３ａに関するデータ処理定義情報９４ａ内には、処理結果の転送先ノードの識別子「566fcac9.d0f784」がｗｉｒｅｓプロパティ情報として設定されている。つまり、あるノードのｗｉｒｅｓプロパティ情報をたどることでノード間の接続関係がわかる。

[0034] なお、複数のノードをグループ化してサブフローとして定義することも可能である。サブフローを複数接続することで１つの全体アプリを定義してもよい。

このように全体アプリフロー編集画面９１上で特定のデータ処理機能を実現するノード同士を繋げることで、ＩｏＴアプリとして実行させるデータ処理の流れ（フロー）を定義できる。すなわち、ノードをグラフィカルに接続するだけで、プログラミング言語によるコーディングなしにＩｏＴアプリを開発できる。またＩｏＴシステム内の各機器に、全体アプリフローデータ９４に基づく処理の実行基盤を構築しておけば、ノードの定義部分をＩｏＴアプリとして、各機器に実行させることができる。例えばフローベースのアプリケーション開発環境の１つであるＮｏｄｅ－ＲＥＤの動作環境は、さまざまなＯＳ上に構築することができる。以下、全体アプリフロー９３内の一部のノードに対応する処理を機器に実行させるＩｏＴアプリを、分配アプリと呼ぶ。

[0035] なお、全体アプリフローデータ９４のデータ構造は、分散機器を跨ることを想定したデータ構造になっていない。すなわち、全体アプリフローデータ９４には、ノード９３ａ，９３ｂ間でのデータ転送先を示す情報としてｗｉｒｅｓプロパティ情報があるが、これは単にノード間のデータ処理順序を示すものである。仮に各ノード９３ａ，９３ｂに対応する機能を複数の機器に配備し、ｗｉｒｅｓプロパティ情報を、データ転送先の機器の識別情報（通信経路情報）とみなしても、デバイス間を跨って動作させるために、通信相手を一意に識別する情報としては不足している。

[0036] 図３は、全体アプリフローデータに示される各ノードに対応する分配アプリの第１の配備例を示す図である。図３の例では、データの送信元のノード９５ａに対応する分配アプリ９７ａを機器９６ａに配備し、データの送信先のノード９５ｂに対応する分配アプリ９７ｂ，９７ｃを２台の機器９６ｂ，９６ｃに配備している。この場合、機器９６ａが分配アプリ９７ａを実行したときの実行結果の送信先として指定された識別子（通信経路情報）を有す

る分配アプリ９７ｂ，９７ｃが２つ存在する。そのため、機器９６ａは、分配アプリ９７ａの実行結果の送信先を一意に識別することができず、処理結果の送信先を決定することができない。

[0037] 図４は、全体アプリフローデータに示される各ノードに対応する分配アプリの第２の配備例を示す図である。図４に示す全体アプリフローでは、２つのノード９５ａ，９５ｂそれぞれの処理結果を、１つのノード９５ｃへ送信することが定義されている。そしてノード９５ａに対応する分配アプリ９７ａを機器９６ａに配備し、ノード９５ｂに対応する分配アプリ９７ｂを機器９６ｂに配備し、データの送信先のノード９５ｃに対応する分配アプリ９７ｃを機器９６ｃに配備している。この場合、機器９６ａ，９６ｂは、ノード９５ａ，９５ｂに対応する分配アプリ９７ａ，９７ｂが持つ転送先情報を転送先の機器９６ｃの識別子とみなして通信経路情報とすると、機器９６ａ，９６ｂからデータを送信することはできる。しかし、データを受信した機器９６ｃでは、同じ通信識別子を持つ２つの通信経路それぞれから受信したデータに関して、どちらの機器９６ａ，９６ｂから送信されたデータなのかが判別できない。

[0038] このように、全体アプリフローデータ９４のままでは、分散環境上で動作する分配アプリとして動作させることはできない。そこで、第２の実施の形態では、アプリ開発サーバにおいて、個別機器間で一意な経路を持つような通信識別子を備えた、各機器に対応した分配アプリを自動生成する。

[0039] 図５は、ＩｏＴシステム構成の一例を示す図である。ＩｏＴシステムは、様々な機器を用いて構築される。例えばＩｏＴシステムには、センサ３１，３２，・・・、およびセンサ３１，３２，・・・と接続してセンサデータの送受信を行うセンサノード４１，４２，・・・が含まれる。また、ＩｏＴシステムには、センサノードとクラウドコンピューティングシステムとの間でデータを中継するＧＷ（ゲートウェイ）ノード５１が含まれる。ＧＷノード５１は、例えば広域ネットワーク２０を介してクラウドコンピューティングシステムに接続される。

[0040] クラウドコンピューティングシステムは、複数のサーバ５２，５３，・・・を含んでいる。サーバ５２，５３，・・・には、Ｗｅｂサーバ、アプリケーションサーバ、データベースサーバなどの、各種サーバが含まれる。

[0041] さらに広域ネットワーク２０には、複数の端末装置６１，６２，・・・が接続されている。端末装置６１，６２，・・・は、例えばパーソナルコンピュータ、またはモバイル端末装置である。

[0042] 広域ネットワーク２０には、さらにアプリ開発サーバ１００が接続されている。アプリ開発サーバ１００は、ＩｏＴシステムに含まれる複数の機器それぞれに実行させる分配アプリを作成し、分配アプリを各機器に配備する。

[0043] このように、ＩｏＴシステムには、様々な種別の装置が、分配アプリの配備先として存在する。以下、分配アプリの配備先となり得る機器を、配備先機器と呼ぶ。図５の例では、センサノード４１，４２，・・・、ＧＷノード５１、サーバ５２，５３，・・・、端末装置６１，６２，・・・それぞれが、配備先機器である。なお、センサノード４１，４２，・・・、ＧＷノード５１、サーバ５２，５３，・・・、端末装置６１，６２，・・・、およびアプリ開発サーバ１００は、例えばコンピュータによって実現される。

[0044] 図６は、アプリ開発サーバのハードウェアの一構成例を示す図である。アプリ開発サーバ１００は、プロセッサ１０１によって装置全体が制御されている。プロセッサ１０１には、バス１０９を介してメモリ１０２と複数の周辺機器が接続されている。プロセッサ１０１は、マルチプロセッサであってもよい。プロセッサ１０１は、例えばＣＰＵ（Central Processing Unit）、ＭＰＵ（Micro Processing Unit）、またはＤＳＰ（Digital Signal Processor）である。プロセッサ１０１がプログラムを実行することで実現する機能の少なくとも一部を、ＡＳＩＣ（Application Specific Integrated Circuit）、ＰＬＤ（Programmable Logic Device）などの電子回路で実現してもよい。

[0045] メモリ１０２は、アプリ開発サーバ１００の主記憶装置として使用される。メモリ１０２には、プロセッサ１０１に実行させるＯＳのプログラムやア

プリケーションプログラムの少なくとも一部が一時的に格納される。また、メモリ１０２には、プロセッサ１０１による処理に利用する各種データが格納される。メモリ１０２としては、例えばＲＡＭ（Random Access Memory）などの揮発性の半導体記憶装置が使用される。

[0046] バス１０９に接続されている周辺機器としては、ストレージ装置１０３、グラフィック処理装置１０４、入力インタフェース１０５、光学ドライブ装置１０６、機器接続インタフェース１０７およびネットワークインタフェース１０８がある。

[0047] ストレージ装置１０３は、内蔵した記録媒体に対して、電気的または磁気的にデータの書き込みおよび読み出しを行う。ストレージ装置１０３は、コンピュータの補助記憶装置として使用される。ストレージ装置１０３には、ＯＳのプログラム、アプリケーションプログラム、および各種データが格納される。なお、ストレージ装置１０３としては、例えばＨＤＤ（Hard Disk Drive）やＳＳＤ（Solid State Drive）を使用することができる。

[0048] グラフィック処理装置１０４には、モニタ２１が接続されている。グラフィック処理装置１０４は、プロセッサ１０１からの命令に従って、画像をモニタ２１の画面に表示させる。モニタ２１としては、有機ＥＬ（Electro Luminescence）を用いた表示装置や液晶表示装置などがある。

[0049] 入力インタフェース１０５には、キーボード２２とマウス２３とが接続されている。入力インタフェース１０５は、キーボード２２やマウス２３から送られてくる信号をプロセッサ１０１に送信する。なお、マウス２３は、ポインティングデバイスの一例であり、他のポインティングデバイスを使用することもできる。他のポインティングデバイスとしては、タッチパネル、タブレット、タッチパッド、トラックボールなどがある。

[0050] 光学ドライブ装置１０６は、レーザ光などを利用して、光ディスク２４に記録されたデータの読み取りを行う。光ディスク２４は、光の反射によって読み取り可能なようにデータが記録された可搬型の記録媒体である。光ディスク２４には、ＤＶＤ（Digital Versatile Disc）、ＤＶＤ－ＲＡＭ、ＣＤ

－ＲＯＭ（Compact Disc Read Only Memory）、ＣＤ－Ｒ（Recordable）／ＲＷ（ReWritable）などがある。

[0051] 機器接続インタフェース１０７は、アプリ開発サーバ１００に周辺機器を接続するための通信インタフェースである。例えば機器接続インタフェース１０７には、メモリ装置２５やメモリリーダライタ２６を接続することができる。メモリ装置２５は、機器接続インタフェース１０７との通信機能を搭載した記録媒体である。メモリリーダライタ２６は、メモリカード２７へのデータの書き込み、またはメモリカード２７からのデータの読み出しを行う装置である。メモリカード２７は、カード型の記録媒体である。

[0052] ネットワークインタフェース１０８は、広域ネットワーク２０に接続されている。ネットワークインタフェース１０８は、広域ネットワーク２０を介して、他のコンピュータまたは通信機器との間でデータの送受信を行う。

[0053] 以上のようなハードウェア構成によって、第２の実施の形態におけるアプリ開発サーバ１００の処理機能を実現することができる。また、センサノード４１，４２，・・・、ＧＷノード５１、サーバ５２，５３，・・・、および端末装置６１，６２，・・・も、アプリ開発サーバ１００と同様のハードウェアにより実現することができる。さらに、第１の実施の形態に示した情報処理装置１０も、図６に示したアプリ開発サーバ１００と同様のハードウェアにより実現することができる。

[0054] アプリ開発サーバ１００は、例えばコンピュータ読み取り可能な記録媒体に記録されたプログラムを実行することにより、第２の実施の形態の処理機能を実現する。アプリ開発サーバ１００に実行させる処理内容を記述したプログラムは、様々な記録媒体に記録しておくことができる。例えば、アプリ開発サーバ１００に実行させるプログラムをストレージ装置１０３に格納しておくことができる。プロセッサ１０１は、ストレージ装置１０３内のプログラムの少なくとも一部をメモリ１０２にロードし、プログラムを実行する。またアプリ開発サーバ１００に実行させるプログラムを、光ディスク２４、メモリ装置２５、メモリカード２７などの可搬型記録媒体に記録しておく

こともできる。可搬型記録媒体に格納されたプログラムは、例えばプロセッサ１０１からの制御により、ストレージ装置１０３にインストールされた後、実行可能となる。またプロセッサ１０１が、可搬型記録媒体から直接プログラムを読み出して実行することもできる。

[0055] 次に、アプリ開発サーバ１００が有する機能について説明する。

図７は、アプリ開発サーバの機能の一例を示すブロック図である。アプリ開発サーバ１００は、デバイス情報記憶部１１０、全体アプリフローデータ記憶部１２０、配備指示部１３０、全体アプリフロー編集部１４０、分配アプリ生成部１５０、および分配アプリ配備部１６０を有する。

[0056] デバイス情報記憶部１１０は、ＩｏＴシステムに含まれる配備先機器についての情報を記憶する。例えばデバイス情報記憶部１１０として、メモリ１０２またはストレージ装置１０３の記憶領域の一部が使用される。

[0057] 図８は、デバイス情報記憶部の一例を示す図である。デバイス情報記憶部１１０には、デバイス管理テーブル１１１が格納されている。デバイス管理テーブル１１１には、例えば配備先機器の機器名に対応付けて、配備先機器のＩＰアドレス、サブネットマスク、および機器識別子が設定されている。サブネットマスクは、ＩＰアドレスのうちのネットワーク部の範囲を示す情報である。ＩＰアドレスとサブネットマスクとのビット単位の論理積により、ネットワーク部の値（ネットワークアドレス）が取得できる。ネットワークアドレスが共通の複数の配備先機器は、内部ネットワークで接続されているものと判断できる。機器識別子は、配備先機器を一意に識別する識別子である。機器識別子は、配備先機器間の通信路の識別子に用いることができる。例えば、データ送信元の配備先機器の機器識別子とデータ送信先の配備先機器の機器識別子とを組み合わせて通信識別子を作成することで、機器間の通信路を一意に識別する識別子を生成することができる。なお、ＩＰアドレスを配備先機器の機器識別子として用いてもよい。

[0058] なお、デバイス情報記憶部１１０内の情報は、例えばシステムの運用者によって予め登録されている。また、配備先機器がネットワークに接続された

際に、その配備先機器が、アプリ開発サーバ１００へ、機器名、ＩＰアドレス、サブネットマスク、機器識別子を送信してもよい。この場合、アプリ開発サーバ１００は、配備先機器から送られた情報を、デバイス情報記憶部１１０に格納する。

[0059] 以下、図７の説明に戻る。

全体アプリフローデータ記憶部１２０は、ＩｏＴシステムに実行させる処理を、単一の機器に実行させた場合の処理手順を示すデータ（全体アプリフローデータ）を記憶する。

[0060] 図９は、全体アプリフローデータ記憶部の一例を示す図である。全体アプリフローデータ記憶部１２０には、全体アプリフローデータ１２１が格納されている。全体アプリフローデータ１２１は、全体アプリフロー編集部１４０により作成され、例えば配備指示部１３０により、全体アプリフローデータ記憶部１２０に格納される。全体アプリフローデータ１２１の内容の詳細は後述する（図１７参照）。

[0061] 以下、図７の説明に戻る。

配備指示部１３０は、全体アプリフローに基づく分配アプリの生成、および分配アプリの配備を制御する。例えば配備指示部１３０は、全体アプリフロー編集部１４０から全体アプリフローデータ１２１を受け取ると、全体アプリフローデータ１２１を全体アプリフローデータ記憶部１２０に格納する。また配備指示部１３０は、アプリ開発者からの、全体アプリフローに含まれる各ノードに対応する機能の配備先である配備先機器の指定入力を受け付ける。この指定入力の受け付けは、例えば画面に表示された配備先機器のリストの中から、アプリ開発者に配備先機器を選択させることで行うことができる。この場合、配備指示部１３０は、デバイス情報記憶部１１０を参照して、デバイス管理テーブル１１１に登録されている配備先機器の名称のリストを画面に表示する。アプリ開発者は、全体アプリフローから配備先機器の指定対象とするノードを選択すると共に、そのノードに対応する機能の配備先となる配備先機器の名称を、配備先機器の名称のリストから選択する。す

ると配備指示部１３０は、選択されたノードに対応する機能の配備先が、選択された配備先機器であると認識する。

[0062] 全体アプリフロー内のすべてのノードについて配備先機器が指定されると、配備指示部１３０は、全体アプリフローデータ記憶部１２０内の全体アプリフローデータ１２１を分配アプリ生成部１５０に送信する。さらに配備指示部１３０は、分配アプリ生成部１５０に対して、全体アプリフローデータ１２１に示される各ノードに対応する機能の配備先を指定する。その後、分配アプリ生成部１５０によって、配備先機器ごとの分配アプリフローデータが生成されると、配備指示部１３０は、分配アプリ配備部１６０に対して、分配アプリの配備先を指定して、分配アプリの配備を指示する。

[0063] 全体アプリフロー編集部１４０は、アプリ開発者からの入力に従って、全体アプリフローを作成する。全体アプリフロー編集部１４０は、例えばＮｏｄｅ－ＲＥＤなどのフローベースのアプリ開発環境である。例えば全体アプリフロー編集部１４０は、Ｗｅｂブラウザ上での全体アプリフローの編集画面を表示し、アプリ開発者による入力に応じて、全体アプリフローの編集画面内で全体アプリフローを作成する。全体アプリフロー編集部１４０は、全体アプリフロー編集画面上での全体アプリフローを作成すると、全体アプリフローデータ１２１を生成し、生成した全体アプリフローデータ１２１を、例えば配備指示部１３０を介して全体アプリフローデータ記憶部１２０に格納する。

[0064] 分配アプリ生成部１５０は、配備指示部１３０からの指示に従って、全体アプリフローデータ１２１に基づいて、ノードに対応する機能の配備先ごとの分配アプリを生成する。分配アプリ生成部１５０は、生成した分配アプリを分配アプリ配備部１６０に送信する。

[0065] 分配アプリ配備部１６０は、配備指示部１３０からの指示に従って、分配アプリを、その分配アプリに対応する配備先機器に配備する。例えば分配アプリ配備部１６０は、分配アプリを示す分配アプリフローデータを配備先機器に送信し、配備先機器に対して分配アプリのインストールおよび起動を指

示する。

[0066] 次に、全体アプリフローに基づく分配アプリの自動生成と配備処理について説明する。

図１０は、分配アプリの自動生成および配備処理の一例を示す図である。全体アプリフロー編集部１４０は、アプリ開発者の入力に基づいて、全体アプリフロー７１を作成する。全体アプリフロー７１に含まれる各ノード７１ａ～７１ｄは、アプリ開発者によって、対応する機能の配備先が指定されている。例えばノード７１ａ，７１ｂに対応する機能（センサデバイス制御、センサ入力値加工）の配備先は「機器Ａ」である。ノード７１ｃに対応する機能（フィルタリング）の配備先は「機器Ｂ」である。ノード７１ｄに対応する機能（可視化）の配備先は「機器Ｃ」である。この全体アプリフロー７１は、例えばセンサから取得したデータを入力データとして、特定デバイス固有のデータ取得処理、取得したデータの加工処理、データのフィルタリング処理（例えば平均化処理など）、取得データの可視化などを順に行うことを示す。

[0067]

アプリ開発者から分配アプリの生成指示が入力されると、全体アプリフロー編集部１４０が作成した全体アプリフロー７１を定義する全体アプリフローデータ１２１が、配備指示部１３０により全体アプリフローデータ記憶部１２０から読み出される。そして配備指示部１３０から分配アプリ生成部１５０へ全体アプリフローデータ１２１が送信される。すると分配アプリ生成部１５０は、全体アプリフローデータ１２１に基づいて、分配アプリ７２～７４を生成する。

[0068] 例えば分配アプリ生成部１５０は、まず全体アプリフロー７１内のノードを、配備先機器ごとに分割する。図１０の例では、配備先が「機器Ａ」のノード７１ａ，７１ｂと、配備先が「機器Ｂ」のノード７１ｃと、配備先が「機器Ｃ」のノード７１ｄとに分割される。

[0069] さらに分配アプリ生成部１５０は、全体アプリフロー７１の分割部分のリ

ンク（全体アプリフロー７１における接続線）を通信処理ノードに置き換える。具体的には、分配アプリ生成部１５０は、データ送信元のノードの出力側（図中右側）にデータ送信処理のノードを追加し、データ送信先のノードの入力側（図中左側）にデータ受信処理のノードを追加する。図１０の例では、ノード７１ｂの出力側に、データ送信処理のノード７１ｅが追加されている。またノード７１ｃの入力側に、データ受信処理のノード７１ｆが追加され、ノード７１ｃの出力側に、データ送信処理のノード７１ｇが追加されている。さらにノード７１ｄの入力側に、データ受信処理のノード７１ｈが追加されている。追加されたデータ送信処理のノード７１ｅ，７１ｇには、例えばデータ送信先の機器との通信を一意に示す識別子が付与される。また追加されたデータ受信処理のノード７１ｆ，７１ｈには、例えばデータ送信元の機器との通信を一意に示す識別子が付与される。

[0070] なお、既存のノードの出力側にノードを追加するとは、分配アプリフローにおいて、追加するノードをアプリフローに追加すると共に、既存のノードの出力側と追加するノードの入力側とを通信のリンクを示す線で接続することである。また既存のノードの入力側にノードを追加するとは、分配アプリフローにおいて、追加するノードをアプリフローに追加すると共に、既存のノードの入力側と追加するノードの出力側とを通信のリンクを示す線で接続することである。

[0071] このようにして、配備先機器ごとの分配アプリ７２～７４が生成される。生成された分配アプリ７２～７４は、それぞれ指定された配備先機器に配備される。具体的には、「機器Ａ」用の分配アプリ７２は「機器Ａ」に配備され、「機器Ｂ」用の分配アプリ７３は「機器Ｂ」に配備され、「機器Ｃ」用の分配アプリ７４は「機器Ｃ」に配備される。

[0072] なお分配アプリ生成部１５０は、データ送信用またはデータ受信用のノードを追加する際、データ送信側とデータ受信側とで共通の通信プロトコルにより通信を行うノードを追加する。例えば分配アプリ生成部１５０は、通信に高いセキュリティが求められる場合には、通信プロトコルを、高いセキュ

リティ機能を有するプロトコルに決定する。すなわち、インターネットのような広域ネットワークを経由した通信の場合、データが第３者に漏洩する危険がある。このような場合、分配アプリ生成部１５０は、送信されるデータが第３者に漏洩しないように、ＨＴＴＰ（Hypertext Transfer Protocol）のような高いセキュリティ機能を有するプロトコルで通信するためのデータ送信およびデータ受信のノードを追加する。ＨＴＴＰは、例えばＨＴＴＰＳ（Hypertext Transfer Protocol Secure）のような認証や暗号化技術を用いた通信が可能なプロトコルである。

[0073] また分配アプリ生成部１５０は、高い処理効率が求められる場合には、通信プロトコルを、効率的な通信が可能なプロトコルに決定する。例えば、企業内のローカルネットワークのようなプライベートのネットワークを経由して、大量の機器からデータを収集する場合、個々のデータ通信のための処理負荷が高いと、収集する機器における負荷が過大となると共に、ネットワーク上の通信負荷も過大となる。このような場合、分配アプリ生成部１５０は、ＭＱＴＴ（Message Queue Telemetry Transport）のような効率的なデータ通信が可能なプロトコルで通信するためのデータ送信およびデータ受信のノードを追加する。ＭＱＴＴは、ＨＴＴＰに比べてヘッダサイズが非常に小さく、通信量や送受信する機器の負荷を低く抑えることができるプロトコルである。

[0074] 図１０の例では、「機器Ａ」と「機器Ｂ」とは、ローカルネットワーク２０ａによって接続されている。すなわち、「機器Ａ」にとって、「機器Ｂ」は、内部ネットワーク経由で接続された機器である。また「機器Ｂ」と「機器Ｃ」とは、広域ネットワーク２０によって接続されている。すなわち、「機器Ｂ」にとって、「機器Ｃ」は、外部ネットワークで接続された機器である。分配アプリ生成部１５０は、データの送信元の機器に対して、データ送信先が内部ネットワークを経由して接続されているのか、外部ネットワークを経由して接続されているのかによって、通信プロトコルを決定する。

[0075] 図１１は、通信プロトコルの決定例を示す図である。分配アプリ生成部１

５０は、全体アプリフロー７１から、データ送信元となっているノードのうち、データ送信先の配備先機器が自身の配備先機器と異なるノードを抽出する。図１１の例では、ノード７１ｂとノード７１ｃとの２つのノードが抽出される。

[0076] 次に、分配アプリ生成部１５０は、抽出したノードそれぞれについて、抽出したノードの配備先機器とデータ送信先のノードの配備先機器とが、内部ネットワーク経由で接続されているのか、外部ネットワーク経由で接続されているのかを判断する。

[0077] 例えば分配アプリ生成部１５０は、デバイス情報記憶部１１０内のデバイス管理テーブル１１１を参照し、配備先機器のＩＰアドレスとサブネットマスクとに基づいて、配備先機器のネットワークアドレスを取得する。そして分配アプリ生成部１５０は、送信元の配備先機器と送信先の配備先機器とのネットワークアドレスが共通であれば、内部ネットワーク経由の通信であると判断する。また分配アプリ生成部１５０は、送信元の配備先機器と送信先の配備先機器とのネットワークアドレスが異なっていれば、外部ネットワーク経由の通信であると判断する。図１１の例では、ノード７１ｂについては、内部ネットワーク経由の通信であると判断され、ノード７１ｃについては、外部ネットワーク経由の通信であると判断される。

[0078] 分配アプリ生成部１５０は、内部ネットワーク経由でデータを送信するノード７１ｂの出力側に、ＭＱＴＴによるデータ送信を行うデータ送信処理のノード７１ｅを追加する。また分配アプリ生成部１５０は、外部ネットワーク経由でデータを送信するノード７１ｃの出力側に、ＨＴＴＰによるデータ送信を行うデータ送信処理のノード７１ｇを追加する。

[0079] さらに分配アプリ生成部１５０は、内部ネットワーク経由でデータを送信するノード７１ｂのデータ送信先のノード７１ｃの入力側に、ＭＱＴＴによるデータ受信を行うデータ受信処理のノード７１ｆを追加する。また分配アプリ生成部１５０は、外部ネットワーク経由でデータを送信するノード７１ｃのデータ送信先のノード７１ｄの入力側に、ＨＴＴＰによるデータ受信を

行うデータ受信処理のノード７１ｈを追加する。

[0080] このようにして、配備先機器に対して、機器間の通信を適切なプロトコルを用いて行わせることができる。

次に、分配アプリの生成および配備の手順について詳細に説明する。

[0081] 図１２は、分配アプリの生成および配備手順を示すフローチャートである。以下、図１２に示す処理をステップ番号に沿って説明する。

［ステップＳ１０１］全体アプリフロー編集部１４０は、アプリ開発者からの入力に基づいて、ＩｏＴアプリとして実行するデータ処理を定義する全体アプリフローを作成する。

[0082] ［ステップＳ１０２］配備指示部１３０は、アプリ開発者からの、全体アプリフローで定義したノードあるいはサブフローに対して、実際のＩｏＴシステム上のどの機器でその処理を行うか（配備先機器）の指定を受け付ける。例えば配備指示部１３０は、デバイス管理テーブル１１１に登録されている機器名を、配備先機器の候補としてモニタ２１に表示する。アプリ開発者は、ノードまたはサブフローごとに、そのノードまたはサブフローに対応する機能を実行させる配備先機器を、候補として表示された機器名の中から選択する。配備指示部１３０は、ノードまたはサブフローごとに選択された機器名を、そのノードまたはサブフローの配備先機器の機器名としてメモリ１０２に格納する。

[0083] ［ステップＳ１０３］配備指示部１３０は、アプリ開発者からの、分配アプリの配備指示を受け付ける。配備の指示とは、全体アプリを指定した配備先機器ごとに分離し、適切な機能を追加して分配アプリを生成し、分配アプリを対応する配備先機器に転送する処理を指示するものである。配備指示部１３０は、配備指示が入力されると、全体アプリフローを表す全体アプリフローデータと、全体アプリフローデータ内の各ノードの配備先情報を分配アプリ生成部１５０に送信する。

[0084] ［ステップＳ１０４］分配アプリ生成部１５０は、全体アプリフローと配備先情報とに基づいて、分配アプリを生成する。分配アプリ生成処理の詳細

は後述する（図１３参照）。分配アプリ生成部１５０は、生成した分配アプリを示す分配アプリフローデータを、分配アプリ配備部１６０に送信する。

[0085] ［ステップＳ１０５］分配アプリ配備部１６０は、分配アプリを、その分配アプリの配備先機器に配備する。具体的には、分配アプリ配備部１６０は、配備先機器ごとの分配アプリの分配アプリフローデータを、その分配アプリに対応する配備先機器に対して送信する。分配アプリ配備部１６０から配備先機器への分配アプリフローデータの送付の際の通信プロトコルとして、ＨＴＴＰ通信、ｗｅｂｓｏｃｋｅｔ通信などの各種の通信プロトコルが利用可能である。さらに分配アプリ配備部１６０は、送信した分配アプリフローデータによる分配アプリのインストールおよび起動要求を、配備先機器に送信する。これにより、配備先機器は、分配アプリフローデータに定義された処理が実行できるように、インストール作業を行い、分配アプリを起動する。

[0086] なお配備先機器が、分配アプリフローデータをインポートするＡＰＩ（Application Programming Interface）を備えている場合がある。この場合、分配アプリ配備部１６０は、例えば配備先機器の分配アプリフロー登録用のＡＰＩに対してデータを送付する。これにより、配備先機器内の分配アプリフローデータの実行ソフトウェアに分配アプリフローデータが実行対象として設定される。このように、分配アプリフロー登録用のＡＰＩを用いて分配アプリを配備することも可能である。

[0087] なお、分配アプリ配備部１６０による分配アプリの配備は、分配アプリ生成処理での分配アプリ生成から連続して行ってもよいし、あるいは分配アプリ生成後にさらにアプリ開発者からの任意のタイミングの指示に応じて行ってもよい。

[0088] 次に、分配アプリ生成処理について詳細に説明する。

図１３は、分配アプリ生成処理の手順の一例を示すフローチャートである。以下、図１３に示す処理をステップ番号に沿って説明する。

[0089] ［ステップＳ１１１］分配アプリ生成部１５０は、全体アプリフローから

ノードを１つ選択する。

［ステップＳ１１２］分配アプリ生成部１５０は、ステップＳ１０２において受け付けた配備先機器の指定に基づいて、選択したノード（選択ノード）の出力データの転送先となるノード（転送先ノード）の配備先機器が、選択ノードの配備先機器と同じか否かを判断する。配備先機器が同じであれば、処理をステップＳ１２５に進める。配備先機器が異なれば、処理をステップＳ１１３に進める。

[0090] ［ステップＳ１１３］分配アプリ生成部１５０は、選択ノードと転送先ノードとの間で、選択ノードを含むアプリフローを分離する。例えば分配アプリ生成部１５０は、選択ノードを含むアプリフローを、選択ノードを最後尾とするアプリフローと、転送先ノードを先頭とするアプリフローとに分離する。分離により生成されたアプリフローが、分配アプリフローとなる。なお、分配アプリフローがさらに分離される場合もある。アプリフローが分離されることで、分離後の分配アプリフローを示すアプリフローデータ（分配アプリフローデータ）が生成される。

[0091] 分配アプリ生成部１５０は、以下のステップＳ１１４～ステップＳ１２４の処理により、分配アプリフローを自動編集する。分配アプリ生成部１５０の編集処理は、具体的には、メモリ１０２内で分配アプリフローデータを編集する処理である。

[0092] ［ステップＳ１１４］分配アプリ生成部１５０は、転送先ノードが、外部のネットワークを介して接続されているか否かを判断する。例えば分配アプリ生成部１５０は、選択したノードと転送先ノードとのＩＰアドレス内のネットワーク部のアドレスが異なる場合、外部のネットワークを介して接続されていると判断する。分配アプリ生成部１５０は、外部のネットワークを介して接続されている場合、処理をステップＳ１１５に進める。また分配アプリ生成部１５０は、内部のネットワークを介して接続されている場合、処理をステップＳ１２０に進める。

[0093] ［ステップＳ１１５］分配アプリ生成部１５０は、選択ノードの出力側に

ＨＴＴＰ送信処理ノードを追加する。追加されるＨＴＴＰ送信処理ノードは、例えば外部ネットワークへの接続のために設置されたＨＴＴＰプロキシサーバを通過するための処理が記述された機能ブロックに対応するノードである。さらに、分配アプリ生成部１５０は、ＨＴＴＰ送信処理ノードの出力側に、転送先ノードからの応答を処理するための応答ノードを追加する。

[0094] ［ステップＳ１１６］分配アプリ生成部１５０は、転送先ノードの入力側にＨＴＴＰ受信処理ノードを追加する。さらに、分配アプリ生成部１５０は、ＨＴＴＰ受信処理ノードの出力側に、選択ノードへのＨＴＴＰ受信応答を行うための応答ノードを追加する。

[0095] ［ステップＳ１１７］分配アプリ生成部１５０は、選択ノードの転送先を、追加したＨＴＴＰ送信処理ノードに変更する。例えば分配アプリ生成部１５０は、選択ノードの転送先を示す情報を、ＨＴＴＰ送信処理ノードのノードＩＤに変更する。

[0096] ［ステップＳ１１８］分配アプリ生成部１５０は、追加したＨＴＴＰ受信処理ノードの転送先情報に、転送先ノードのノードＩＤを付与する。

［ステップＳ１１９］分配アプリ生成部１５０は、追加したＨＴＴＰ送信処理ノードに、転送先ノードの配備先機器のＩＰアドレスを設定する。その後、分配アプリ生成部１５０は、処理をステップＳ１２５に進める。

[0097] ［ステップＳ１２０］分配アプリ生成部１５０は、選択ノードの出力側に、ＭＱＴＴ送信処理ノードを追加する。

［ステップＳ１２１］分配アプリ生成部１５０は、転送先ノードの入力側に、ＭＱＴＴ受信処理ノードを追加する。

[0098] ［ステップＳ１２２］分配アプリ生成部１５０は、選択ノードの転送先を、追加したＭＱＴＴ送信処理ノードに変更する。例えば分配アプリ生成部１５０は、選択ノードの転送先を示す情報を、ＭＱＴＴ送信処理ノードのノードＩＤに変更する。

[0099] ［ステップＳ１２３］分配アプリ生成部１５０は、追加したＭＱＴＴ受信処理ノードの転送先情報に、転送先ノードのノードＩＤを付与する。

［ステップＳ１２４］分配アプリ生成部１５０は、追加したＭＱＴＴ送信処理ノードと追加したＭＱＴＴ受信処理ノードとのそれぞれに、ＭＱＴＴ送信処理ノードとＭＱＴＴ受信処理ノードとの組を一意に示す通信識別子を設定する。なお、ＭＱＴＴにおける通信識別子はトピックと呼ばれる。

[0100] ［ステップＳ１２５］分配アプリ生成部１５０は、全体アプリフロー内のすべてのノードを選択したか否かを判断する。分配アプリ生成部１５０は、すべてのノードが選択済みであれば、分配アプリ生成処理を終了する。分配アプリ生成部１５０は、未選択のノードが存在すれば、処理をステップＳ１１１に進める。

[0101] このようにして、全体アプリを、指定された配備先機器が共通のノードごとに分割し、分散環境上での分配アプリの連携のための配備先機器間の通信処理機能を、分配アプリフロー上の適切な位置に自動的に付加することができる。しかも、配備先機器間の通信が、外部ネットワークを経由するのか否かにより、適切な通信プロトコルを適用することができる。

[0102] すなわち、外部ネットワーク経由の通信であれば、分配アプリ生成部１５０は、通信処理機能としてＨＴＴＰの通信処理を追加する。そして分配アプリ生成部１５０が、アプリフローの分離により自動的に付加したＨＴＴＰ送信処理ノードに対して、送信先となる配備先機器のＩＰアドレスを設定することで、ＨＴＴＰ通信が可能となる。なおＨＴＴＰ通信では、パケットの送信元のＩＰアドレスが含まれるため、ＨＴＴＰ受信処理ノードでは、複数の分配アプリの処理結果を受信する場合であっても、受信したパケットのＩＰアドレスによって送信元を一意に区別することができる。

[0103] また内部ネットワーク経由の通信であれば、分配アプリ生成部１５０は、通信処理機能としてＭＱＴＴの通信処理を追加する。そして分配アプリ生成部１５０が、アプリフローの分離により自動的に付加したＭＱＴＴノードに対して、適切なトピックを設定することで、ＭＱＴＴ通信が可能となる。

[0104] 次に配備先機器における処理について説明する。配備先機器は、アプリ開発サーバ１００側から転送されてきた分配アプリフローデータを受信すると

、分配アプリフローデータに基づいて分配アプリを実行する。

[0105] 図１４は、配備先機器の機能の一例を示すブロック図である。配備先機器２００は、分配アプリ受信部２１０、分配アプリフローデータ記憶部２２０、分配アプリ制御部２３０、機能実行モジュール記憶部２４０、およびアプリ実行部２５０を有する。

[0106] 分配アプリ受信部２１０は、アプリ開発サーバ１００から、分配アプリのインストールおよび起動の指示と共に、分配アプリフローデータを受信する。分配アプリ受信部２１０は、受信した分配アプリフローデータを分配アプリフローデータ記憶部２２０に格納する。

[0107] 分配アプリフローデータ記憶部２２０は、分配アプリフローデータを記憶する。例えば分配アプリフローデータ記憶部２２０は、メモリまたはストレージ装置の記憶領域の一部によって実現される。

[0108] 分配アプリ制御部２３０は、分配アプリフローデータ記憶部２２０に格納された分配アプリフローデータに基づいて、分配アプリ２６１の実行を制御する。例えば分配アプリ制御部２３０は、分配アプリフローデータ記憶部２２０から分配アプリフローデータを読み出し、読み出した分配アプリフローデータをアプリ実行部２５０に送信する。そして分配アプリ制御部２３０は、分配アプリフローデータに基づく分配アプリの起動を、アプリ実行部２５０に指示する。

[0109] 機能実行モジュール記憶部２４０は、機能ごとの実行モジュール２４１，２４２，・・・を記憶する。実行モジュール２４１，２４２，・・・は、アプリ開発サーバ１００の全体アプリフロー編集部１４０に予め用意された機能ブロックそれぞれの機能を、配備先機器２００で実現するためのプログラムモジュールである。機能実行モジュール記憶部２４０は、例えばメモリまたはストレージ装置の記憶領域の一部によって実現される。

[0110] アプリ実行部２５０は、分配アプリフローデータに基づいて分配アプリ２６１の実行を開始する。例えばアプリ実行部２５０は、分配アプリフローデータに示される機能名に対応する実行モジュールを用いて分配アプリ２６１

を実行する。その際、アプリ実行部２５０は、分配アプリフローデータに示されているパラメータなどの定義情報を、分配アプリ２６１の実行条件として設定する。

[0111] アプリ実行部２５０は、分配アプリ２６１の実行を開始すると、例えば他の配備先機器２００ａによる分配アプリの実行結果を示すデータを取得し、分配アプリ２６１に基づいて、取得したデータを処理する。またアプリ実行部２５０は、分配アプリ２６１に基づく処理の実行結果を、例えば他の配備先機器２００ｂに送信する。

[0112] 図１４に示した配備先機器２００の機能を、センサノード４１，４２，・・・、GWノード５１、サーバ５２，５３，・・・、端末装置６１，６２，・・・それぞれが有している。

[0113] 図１５は、配備先機器による処理の手順の一例を示すフローチャートである。以下、図１５に示す処理をステップ番号に沿って説明する。

［ステップＳ２０１］分配アプリ受信部２１０は、アプリ開発サーバ１００から分配アプリフローデータを受信する。

[0114] ［ステップＳ２０２］分配アプリ受信部２１０は、受信した分配アプリフローデータを、分配アプリフローデータ記憶部２２０に格納する。

［ステップＳ２０３］分配アプリ制御部２３０は、アプリ実行部２５０の実行対象として、受信した分配アプリフローデータを設定する。例えばアプリ実行部２５０にアプリフローデータを取り込む（インポート）ためのインタフェースがある場合、分配アプリ制御部２３０は、そのインタフェースを介して、受信した分配アプリフローデータをアプリ実行部２５０に取り込ませる。そして分配アプリ制御部２３０は、アプリ実行部２５０に、分配アプリフローデータの実行（分配アプリの起動）を指示する。

[0115] ［ステップＳ２０４］アプリ実行部２５０は、分配アプリフローデータに基づいて、分配アプリの実行を開始する。例えば、アプリ実行部２５０は、分配アプリフローデータに示される順番で、各ノードに対応する実行モジュールを機能実行モジュール記憶部２４０から読み出し、読み出した機能実行

モジュールを実行する。

[0116] このようにして、配備先機器に送信された分配アプリフローデータに基づいて、分配アプリの処理が実行される。

以下、図１６～図２１を参照して、分配アプリフローデータの生成例を説明する。

[0117] 図１６は、全体アプリフロー編集画面の一例を示す図である。図１６の例では、アプリ開発者は、日付データを生成した後、生成した日付データのデータ形式を日付文字形式（Fri Dec 15 2017 14:10:21 GMT+0900（東京（標準時）など）に変換し、出力するアプリを作成するものとする。

[0118] 全体アプリフロー編集画面８１の機能ブロック表示領域８１ａ内に表示されている機能ブロックには、「日付データ生成」の機能ブロック８２ａ、「日付データ整形」の機能ブロック８２ｂ、「日付データ出力」の機能ブロック８２ｃが含まれている。アプリ開発者は、全体アプリフロー編集部１４０に対して入力を行い、フロー表示領域８１ｂ内に、機能ブロック８２ａ～８２ｃそれぞれに対応するノード８３ａ～８３ｃを配置させる。次にアプリ開発者は、全体アプリフロー編集部１４０に対して入力を行い、「日付データ生成」のノード８３ａの出力側と「日付データ整形」のノード８３ｂの入力側とを、通信のリンクを示す線で接続させる。さらにアプリ開発者は、全体アプリフロー編集部１４０に対して入力を行い、「日付データ整形」のノード８３ｂの出力側と「日付データ出力」のノード８３ｃの入力側とを、通信のリンクを示す線で接続させる。これにより、全体アプリフロー８３が完成する。

[0119] 全体アプリフロー８３が完成すると、全体アプリフロー編集部１４０は、全体アプリフロー８３を示す全体アプリフローデータを生成する。全体アプリフローデータは、配備指示部１３０を介して、全体アプリフローデータ記憶部１２０に格納される。

[0120] 図１７は、全体アプリフローデータの一例を示す図である。図１７の例では、ＪＳＯＮデータ構造によって全体アプリフローデータ８４が作成されて

いる。全体アプリフローデータ８４には、全体アプリフロー８３内のノード８３ａ，８３ｂ，８３ｃそれぞれを定義する記述（データ処理定義情報８４－１～８４－３）が含まれている。各ノード８３ａ，８３ｂ，８３ｃを定義するデータ処理定義情報８４－１～８４－３には、項目名「ｉｄ」の値としてそのノードの識別子が示されている。また、項目名「ｎａｍｅ」の値として、そのノードに対応する機能ブロックの名前が示されている。そして、出力側が他のノードに接続されたノードを定義するデータ処理定義情報には、項目名「ｗｉｒｅｓ」の値として、データ送信先のノードの識別子が示されている。

[0121] 全体アプリフローデータ８４が全体アプリフローデータ記憶部１２０に格納されると、アプリ開発者は、配備指示部１３０に対して、各ノード８３ａ～８３ｃに対応する機能の配備先機器を指定する入力を行う。

[0122] 図１８は、配備先機器の指定例を示す図である。図１８の例では、「日付データ生成」のノード８３ａに対応する機能を配備する配備先機器として、配備先機器８５ａが指定されている。「日付データ整形」のノード８３ｂに対応する機能を配備する配備先機器として、配備先機器８５ｂが指定されている。「日付データ出力」のノード８３ｃに対応する機能を配備する配備先機器として、配備先機器８５ｃが指定されている。

[0123] なお、図１８の例では、配備先機器８５ａの識別子は「ｄｅｖ１」であり、配備先機器８５ａはネットワーク「Ｎ１」に属している。配備先機器８５ｂの識別子は「ｄｅｖ２」であり、配備先機器８５ｂはネットワーク「Ｎ１」に属している。配備先機器８５ｃの識別子は「ｄｅｖ３」であり、配備先機器８５ｃはネットワーク「Ｎ２」に属している。すなわち、配備先機器８５ａと配備先機器８５ｂとは、同じローカルネットワークに属するが、配備先機器８５ａ，８５ｂそれぞれと配備先機器８５ｃとは、異なるネットワークに属する。

[0124] 配備先機器の指定を受けた配備指示部１３０は、全体アプリフローデータ８４と各ノード８３ａ～８３ｃの配備先機器を示す情報とを、分配アプリ生

成部１５０に送信する。すると分配アプリ生成部１５０は、全体アプリフローデータ８４に基づいて、配備先機器８５ａ～８５ｃごとの分配アプリフローデータを生成する。

[0125] 図１９は、「日付データ生成」機能を配備する配備先機器用の分配アプリフローデータの一例を示す図である。「日付データ生成」機能の配備先である配備先機器８５ａと、「日付データ整形」機能の配備先機器８５ｂとは、同じローカルネットワークに属する。この場合、分配アプリ生成部１５０は、「日付データ生成」機能によって生成された日付データを、ＭＱＴＴで送信する分配アプリフロー８３－１を表す分配アプリフローデータ８４ａを生成する。

[0126] 分配アプリフロー８３－１は、「日付データ生成」のノード８３ａと、ＭＱＴＴでデータ送信を行う機能ブロック「データ送信＿ＭＱＴＴ」に対応するノード８３ｄとを有する。そしてノード８３ａの出力側とノード８３ｄの入力側とが、通信のリンクを示す線で接続をされている。

[0127] 分配アプリフローデータ８４ａでは、「日付データ生成」に対応するノード８３ａを定義するデータ処理定義情報８４ａ－１における項目名「ｗｉｒｅｓ」の値に示される送信先ノードの識別子が、ノード８３ｄの識別子に変更されている。また分配アプリフローデータ８４ａには、ノード８３ｄを定義するデータ処理定義情報８４ａ－２が追加されている。ノード８３ｄを定義するデータ処理定義情報８４ａ－２内には、項目名「ｔｏｐｉｃ」の値として、ＭＱＴＴトピックの値が設定されている。ＭＱＴＴトピックの値は、「日付データ生成」のノード８３ａの配備先である配備先機器８５ａの識別子「ｄｅｖ１」と、「日付データ整形」のノード８３ｂの配備先である配備先機器８５ｂの識別子「ｄｅｖ２」とを含む値である。図１９の例では、ＭＱＴＴトピックとして値「ｄｅｖ１ｔｏｄｅｖ２」が設定されている。

[0128] なお、ＭＱＴＴ通信に用いるブローカ（転送制御機能）の詳細情報は、項目名「ｂｒｏｋｅｒ」の値を、項目名「ｉｄ」の値として有する定義部分に記述されている。

「日付データ生成」機能の配備先である配備先機器８５ａは、このような分配アプリフローデータ８４ａに基づいて、日付データを生成し、生成した日付データをＭＱＴＴによって配備先機器８５ｂに送信することができる。

[0129] 図２０は、「日付データ整形」機能を配備する配備先機器用の分配アプリフローデータの一例を示す図である。「日付データ整形」機能の配備先である配備先機器８５ｂと、「日付データ出力」機能の配備先機器８５ｃとは、異なるネットワークに属する。この場合、分配アプリ生成部１５０は、「日付データ整形」機能によって整形された日付データを、ＨＴＴＰで送信する分配アプリフロー８３－２を表す分配アプリフローデータ８４ｂを生成する。

[0130] 分配アプリフロー８３－２は、「データ受信＿ＭＱＴＴ」のノード８３ｅ、「日付データ整形」のノード８３ｂ、「データ送信＿ＨＴＴＰ」のノード８３ｆ、および「ＨＴＴＰ応答」のノード８３ｇを有する。ノード８３ｅは、ＭＱＴＴでデータ受信を行う機能ブロックに対応する。ノード８３ｆは、ＨＴＴＰでデータ送信を行う機能ブロックに対応する。ノード８３ｇは、ＨＴＴＰによるＰＵＴ要求に対する応答に応じた処理を行う機能ブロックに対応する。そしてノード８３ｅの出力側とノード８３ｂの入力側とが、通信のリンクを示す線で接続されている。またノード８３ｂの出力側とノード８３ｆの入力側とが、通信のリンクを示す線で接続されている。さらにノード８３ｆの出力側とノード８３ｇの入力側とが、通信のリンクを示す線で接続されている。

[0131] 「データ受信＿ＭＱＴＴ」に対応するノード８３ｅを定義するデータ処理定義情報８４ｂ－１には、項目名「ｎａｍｅ」の値として、ノード８３ｅに対応する機能ブロックの名前が示されている。そして、項目名「ｗｉｒｅｓ」の値として、データ送信先のノード８３ｂの識別子が示されている。ＭＱＴＴ通信に用いるブローカ（転送制御機能）の詳細情報は、項目名「ｂｒｏｋｅｒ」の値を、項目名「ｉｄ」の値として有する定義部分に記述されている。

[0132] 「日付データ整形」に対応するノード８３ｂを定義するデータ処理定義情報８４ｂ－２では、項目名「ｗｉｒｅｓ」の値が、ノード８３ｆの識別子に変更されている。

「データ送信＿ＨＴＴＰ」に対応するノード８３ｆを定義するデータ処理定義情報８４ｂ－３には、項目名「ｎａｍｅ」の値に、ノード８３ｆに対応する機能ブロックの名前が示されている。また項目名「ｍｅｔｈｏｄ」の値に、ＨＴＴＰのＰＵＴ処理でデータ送信を行うことが示されている。データの送信先は、項目名「ｕｒｌ」の値として設定された、ＰＵＴ処理での送信先となる配備先機器のＵＲＬによって示されている。さらに、項目名「ｗｉｒｅｓ」の値に、ＰＵＴ処理結果（ＰＴＵ処理要求を送信したこと）を示すデータの送信先のノード８３ｇの識別子が示されている。

[0133] 「ＨＴＴＰ応答」に対応するノード８３ｇを定義するデータ処理定義情報８４ｂ－４には、項目名「ｎａｍｅ」の値として、ノード８３ｆに対応する機能ブロックの名前が示されている。

[0134] 「日付データ整形」機能の配備先である配備先機器８５ｂは、このような分配アプリフローデータ８４ｂに基づいて、日付データを受信し、受信した日付データを整形し、整形後の日付データを、ＨＴＴＰによって配備先機器８５ｃに送信することができる。

[0135] 図２１は、「日付データ出力」機能を配備する配備先機器用の分配アプリフローデータの一例を示す図である。分配アプリ生成部１５０は、整形された日付データを受信し、受信した日付データを出力する分配アプリフロー８３－３を表す分配アプリフローデータ８４ｃを生成する。

[0136] 分配アプリフロー８３－３は、「定期取得」のノード８３ｈ、「データ受信＿ＨＴＴＰ」のノード８３ｉ、「日付データ出力」のノード８３ｃ、および「ＨＴＴＰ応答」のノード８３ｊを有する。ノード８３ｈは、データの定期取得を行う機能ブロックに対応する。ノード８３ｉは、ＨＴＴＰでデータ受信を行う機能ブロックに対応する。ノード８３ｊは、ＨＴＴＰのＧＥＴ要求に対する応答に応じた処理を行う機能ブロックに対応する。そしてノード

８３ｈの出力側とノード８３ｉの入力側とが、通信のリンクを示す線で接続されている。またノード８３ｉの出力側とノード８３ｃの入力側とが、通信のリンクを示す線で接続されている。さらにノード８３ｉの出力側とノード８３ｊの入力側とが、通信のリンクを示す線で接続されている。

[0137] 「定期取得」に対応するノード８３ｈを定義するデータ処理定義情報８４ｃ－１には、項目名「ｎａｍｅ」の値として、ノード８３ｈに対応する機能ブロックの名前が示されている。また項目名「ｒｅｐｅａｔ」の値として、１０秒ごとにデータ受信を行うことが示されている。そして、項目名「ｗｉｒｅｓ」の値として、データ送信先のノード８３ｉの識別子が示されている。ノード８３ｈに対応する定期取得機能を実行することで、データ受信タイミングであることを示す情報が、１０秒間隔で出力される。

[0138] 「データ受信＿ＨＴＴＰ」に対応するノード８３ｉを定義するデータ処理定義情報８４ｃ－２には、項目名「ｎａｍｅ」の値として、ノード８３ｉに対応する機能ブロックの名前が示されている。また項目名「ｍｅｔｈｏｄ」の値として、ＨＴＴＰのＧＥＴ処理でデータを取得することが示されている。データ取得先としては、ＧＥＴ処理での取得先となる位置（配備先機器８５ｃ内の位置）が、項目名「ｕｒｌ」の値に設定されたＵＲＬによって示されている。そして、項目名「ｗｉｒｅｓ」の値として、データ送信先となる２つのノード８３ｃ，８３ｊそれぞれの識別子が示されている。

[0139] 「日付データ出力」に対応するノード８３ｃを定義するデータ処理定義情報８４ｃ－３には、項目名「ｎａｍｅ」の値として、ノード８３ｃに対応する機能ブロックの名前が示されている。

[0140] 「ＨＴＴＰ応答」に対応するノード８３ｊを定義するデータ処理定義情報８４ｃ－４には、項目名「ｎａｍｅ」の値として、ノード８３ｊに対応する機能ブロックの名前が示されている。

[0141] 「日付データ出力」機能の配備先である配備先機器８５ｃは、このような分配アプリフローデータ８４ｃに基づいて、整形された日付データを受信し、その日付データを、予め指定された出力先（所定のファイル、モニタなど

）に出力することができる。

[0142] 図１９～図２１に示したような各分配アプリフローデータ８４ａ～８４ｃは、それぞれ配備先機器８５ａ～８５ｃに配備される。その結果、配備先機器８５ａ～８５ｃの連携処理により、日付データを生成した後、生成した日付データのデータ形式を日付文字形式に変換し、出力する処理が実行される。

[0143] このように、アプリ開発サーバ１００は、全体アプリフローを、各ノードに指定された配備先機器に基づいて、複数の分配アプリそれぞれの処理手順を定義した複数の分配アプリフローに自動で分割することができる。しかもアプリ開発サーバ１００は、分配アプリフローに、ネットワークを介したデータ送受信用のノードを追加することで、分配アプリの配備先機器間のデータ通信を可能としている。この際、データ送受信用のノードには、通信相手となるノードが配備される配備先機器を一意に特定可能な識別子（トピックまたはＵＲＬ）が付与されるため、分配アプリが配備された各配備先機器は、通信相手を一意に識別することができる。その結果、例えば図３に示したように、データの送信先が複数存在する場合であっても、送信元の機器は、送信先の機器それぞれを区別して、適切なデータを送信することができる。また例えば図４に示したように、複数の機器からデータを受信する場合であっても、データを受信する機器は、データの送信元の機器を一意に識別し、送信元と受信したデータとを関連付けた適切なデータ処理を実施することができる。

[0144] しかもアプリ開発サーバ１００は、通信がローカルネットワーク内で行われる場合には、通信プロトコルとしてＭＱＴＴを用い、効率的な通信を行うことができる。それに対し、アプリ開発サーバ１００は、通信が外部ネットワークを経由する場合には、通信プロトコルとしてＨＴＴＰを用い、信頼性の高い通信を行うことができる。その結果、ＩｏＴシステムにおける分散処理による信頼性の低下を回避しながら、処理の効率化を図ることができる。

[0145] 〔その他の実施の形態〕

第２の実施の形態では、全体アプリフローデータまたは分配アプリフローデータを、ＪＳＯＮデータ構造のデータで表しているが、他のデータ構造のデータで表すこともできる。例えば、全体アプリフローデータまたは分配アプリフローデータを、表形式のデータで表すこともできる。

[0146] 図２２は、表形式で表した分配アプリフローデータの一例を示す図である。図２２には、分配アプリフロー８３－１を、複数のデータテーブル８６ａ～８６ｃを用いた表形式のデータ構造で表した例が示されている。

[0147] 例えば「日付データ生成」のノード８３ａは、１つのデータテーブル８６ａで表されている。また「データ送信＿ＭＱＴＴ」のノード８３ｄは、２つのデータテーブル８６ｂ，８６ｃで表されている。各データテーブル８６ａ～８６ｃには、データの項目名に対応付けて、その項目名に対応する値が設定されている。例えば各データテーブル８６ａ～８６ｃの先頭のレコードには、項目名「ｉｄ」に対応付けて、そのデータテーブルの識別子が設定されている。また「日付データ生成」のノード８３ａを示すデータテーブル８６ａの最後のレコードには、項目名「ｗｉｒｅｓ」に対応付けて、データ送信先のノード８３ｄに対応するデータテーブル８６ｄの識別子が設定されている。データテーブル８６ｂの項目名「ｂｒｏｋｅｒ」のレコードには、ＭＱＴＴによる通信の詳細情報を示すデータテーブル８６ｃの識別子が設定されている。

[0148] このように、分配アプリフローデータを表形式で表すこともできる。なお図２２には、分配アプリフローデータの例を示したが、全体アプリフローデータも同様に表形式で表すことができる。

[0149] また第２の実施の形態では、全体アプリフローの編集を、アプリ開発サーバ１００に接続した入出力デバイス（モニタ２１、キーボード２２、マウス２３）を介して行うことを想定しているが、他の端末装置を用いて全体アプリフローを編集することもできる。例えば全体アプリフロー編集部１４０は、ＨＴＭＬ形式の全体アプリフロー編集画面用のデータを生成する。そして全体アプリフロー編集部１４０は、全体アプリフロー編集画面用のデータを

、アプリ開発サーバ１００にネットワークを介して接続された端末装置に送信する。端末装置は、例えばＷｅｂブラウジング機能を有するパーソナルコンピュータである。アプリ開発者は、端末装置を用いて全体アプリフロー編集部１４０にアクセスし、全体アプリフローを編集することができる。

[0150] またアプリ開発者から配備指示部１３０への配備先機器の指定についても、アプリ開発サーバ１００に接続された端末装置を用いて行うこともできる。例えば配備指示部１３０は、ＨＴＭＬ形式の配備先機器指定画面用のデータを生成する。そして配備指示部１３０は、配備先機器指定画面用のデータを、アプリ開発サーバ１００にネットワークを介して、例えばＷｅｂブラウジング機能を有する端末装置に送信する。アプリ開発者は、端末装置を用いて配備指示部１３０にアクセスし、全体アプリフロー内の各ノードについて、配備先機器を指定することができる。

[0151] 上記については単に本発明の原理を示すものである。さらに、多数の変形、変更が当業者にとって可能であり、本発明は上記に示し、説明した正確な構成および応用例に限定されるものではなく、対応するすべての変形例および均等物は、添付の請求項およびその均等物による本発明の範囲とみなされる。

符号の説明

[0152] １　全体処理定義データ

１－１，１－２　部分処理定義データ

１ａ　第１データ処理定義情報

１ｂ　第２データ処理定義情報

１ｃ　データ送信定義情報

１ｄ　データ受信定義情報

２　第１機器

３　第２機器

４　ネットワーク

１０　情報処理装置

１１　記憶部

１２　処理部

請求の範囲

[請求項1] 第１データ処理の実行内容と、前記第１データ処理の処理結果を用いた第２データ処理の実行内容とを定義した全体処理定義データのうちの、前記第１データ処理の実行内容の定義部分であるデータ処理定義情報と、

前記第１データ処理を実行させる第１機器と前記第２データ処理を実行させる第２機器との通信を識別する通信識別子を含み、前記第１データ処理の結果である出力データをネットワーク経由で前記第２機器に送信する処理の実行内容を定義したデータ送信定義情報と、を有し、

前記第１機器による前記第１データ処理の実行に用いられる、

部分処理定義データのデータ構造。

[請求項2] 前記第１機器が、前記データ処理定義情報に基づいて、前記第１データ処理を実行し、前記第１データ処理の結果生成された前記出力データを、前記データ送信定義情報に基づいて、前記通信識別子によって特定される前記第２機器に対して送信する、処理の実行に用いられる、

請求項１記載の部分処理定義データのデータ構造。

[請求項3] 前記通信識別子は、前記第１機器を識別する第１機器識別子と、前記第２機器を識別する第２機器識別子とを含む、

請求項１または２記載の部分処理定義データのデータ構造。

[請求項4] 前記データ送信定義情報は、前記第１機器と前記第２機器との通信に用いるプロトコルを指定する情報を含む、

請求項１ないし３のいずれかに記載の部分処理定義データのデータ構造。

[請求項5] 前記全体処理定義データには、第３データ処理の実行内容が定義されており、前記第１データ処理は、前記第３データ処理の処理結果を用いたデータ処理であり、

さらに、前記第３データ処理を実行させる第３機器と前記第１機器との通信を識別する受信用通信識別子を含み、前記第３データ処理の結果である入力データをネットワーク経由で前記第３機器から受信する処理の実行内容を定義したデータ受信定義情報を有する、

請求項１ないし４のいずれかに記載の部分処理定義データのデータ構造。

[請求項6] コンピュータが、

第１データ処理の実行内容と、前記第１データ処理の処理結果を用いた第２データ処理の実行内容とを定義した全体処理定義データから、前記第１データ処理の実行内容の定義部分であるデータ処理定義情報を抽出し、

前記第１データ処理の結果である出力データをネットワーク経由で送信する処理の実行内容を定義したデータ送信定義情報に、前記第１データ処理を実行させる第１機器と前記第２データ処理を実行させる第２機器との通信を識別する通信識別子を設定し、

前記データ処理定義情報と、前記通信識別子が設定された前記データ送信定義情報とを含む部分処理定義データを生成する、

部分処理定義データ生成方法。

[請求項7] 前記コンピュータは、さらに、

前記複数のデータ処理のそれぞれを実行する機器を示す情報に基づいて、前記全体処理定義データに定義された複数のデータ処理の中から判断対象データ処理を選択し、前記判断対象データ処理を実行する機器と、前記判断対象データ処理の処理結果を用いた他のデータ処理を実行する機器とが異なる場合、前記判断対象データ処理を前記第１データ処理と判断し、前記他のデータ処理を前記第２データ処理と判断する、

請求項６記載の部分処理定義データ生成方法。

[請求項8] 前記コンピュータは、さらに、

前記第１データ処理を実行する前記第１機器と前記第２データ処理を実行する前記第２機器とを接続する前記ネットワークの種別に基づいて、前記出力データを送信する際の通信プロトコルを決定し、

決定した前記通信プロトコルで前記出力データを送信する前記データ送信定義情報を含む前記部分処理定義データを生成する、

請求項６または７記載の部分処理定義データ生成方法。

[請求項9] 前記コンピュータは、さらに、

生成した前記部分処理定義データを前記第１機器に送信する、

請求項６ないし８のいずれかに記載の部分処理定義データ生成方法。

[請求項10] 前記コンピュータは、さらに、

前記全体処理定義データから、前記第２データ処理の実行内容の定義部分である受信側データ処理定義情報を抽出し、

前記第１データ処理の結果である前記出力データを前記ネットワーク経由で受信する処理の実行内容を定義したデータ受信定義情報に、前記通信識別子を設定し、

前記受信側データ処理定義情報と、前記通信識別子が設定された前記データ送信定義情報とを含む受信側部分処理定義データを生成する、

請求項６ないし９のいずれかに記載の部分処理定義データ生成方法。

[請求項11] コンピュータに、

第１データ処理の実行内容と、前記第１データ処理の処理結果を用いた第２データ処理の実行内容とを定義した全体処理定義データから、前記第１データ処理の実行内容の定義部分であるデータ処理定義情報を抽出し、

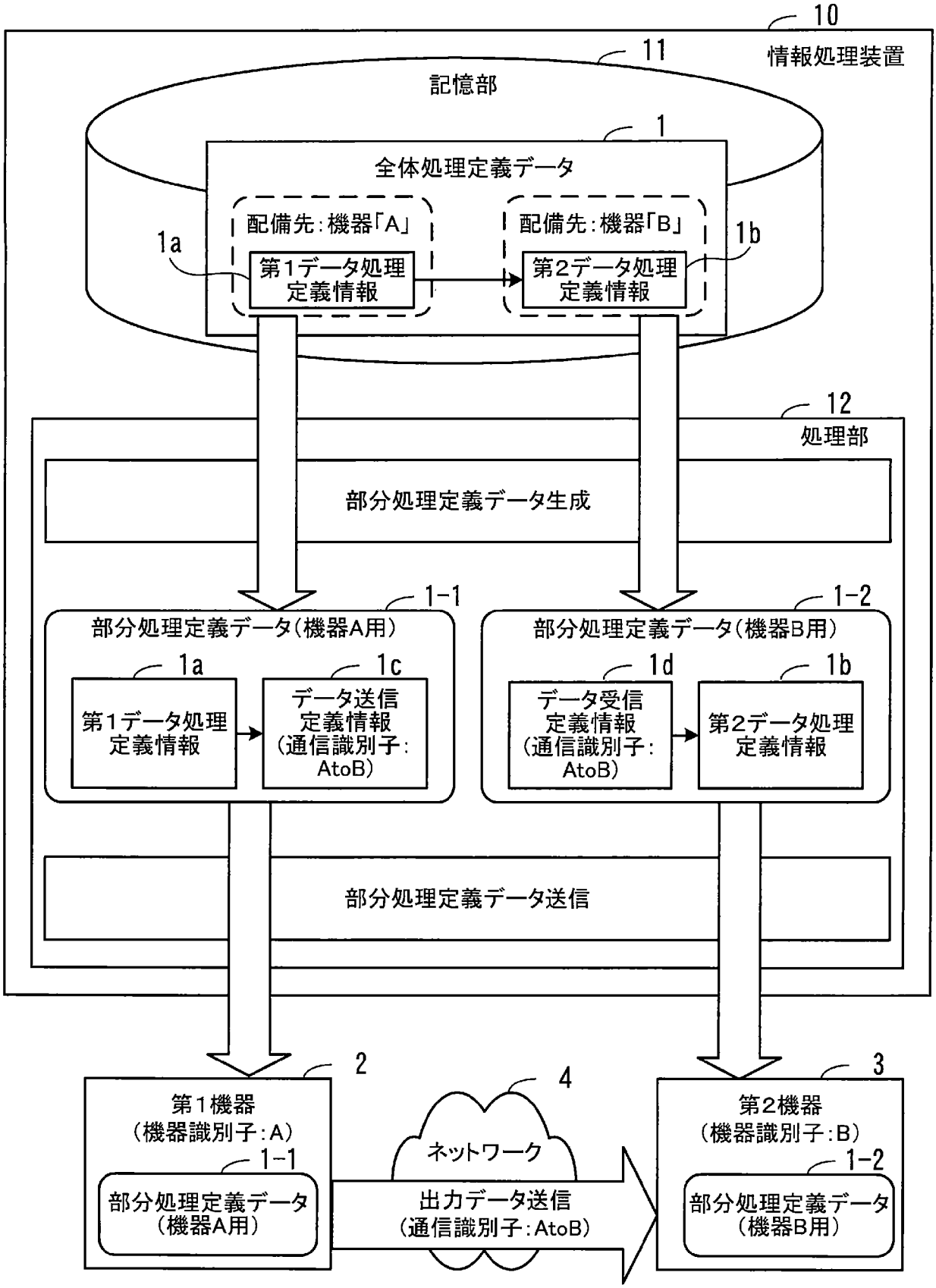
前記第１データ処理の結果である出力データをネットワーク経由で送信する処理の実行内容を定義したデータ送信定義情報に、前記第１

データ処理を実行させる第１機器と前記第２データ処理を実行させる第２機器との通信を識別する通信識別子を設定し、

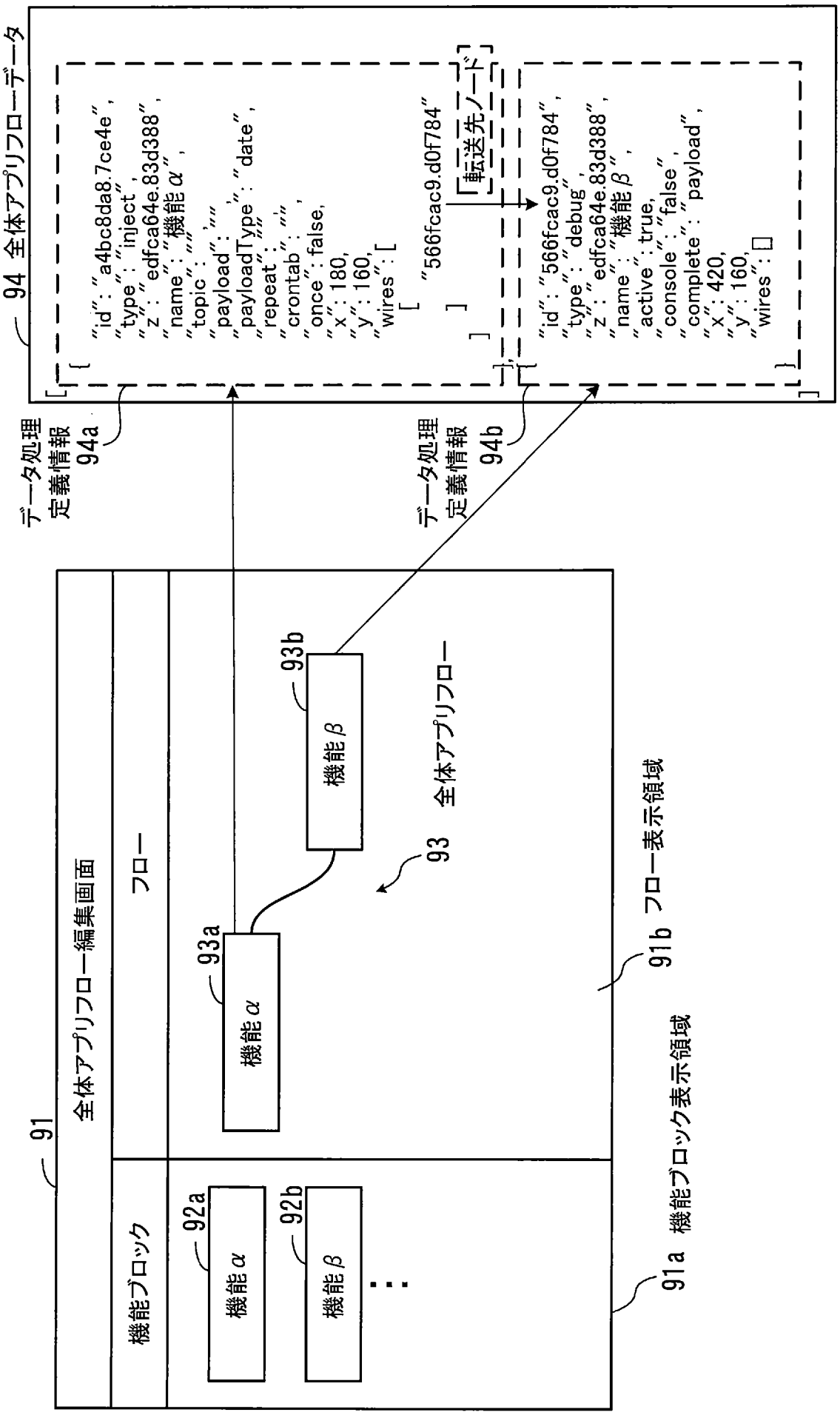
前記データ処理定義情報と、前記通信識別子が設定された前記データ送信定義情報とを含む部分処理定義データを生成する、

処理を実行させる部分手順定義データ生成プログラム。

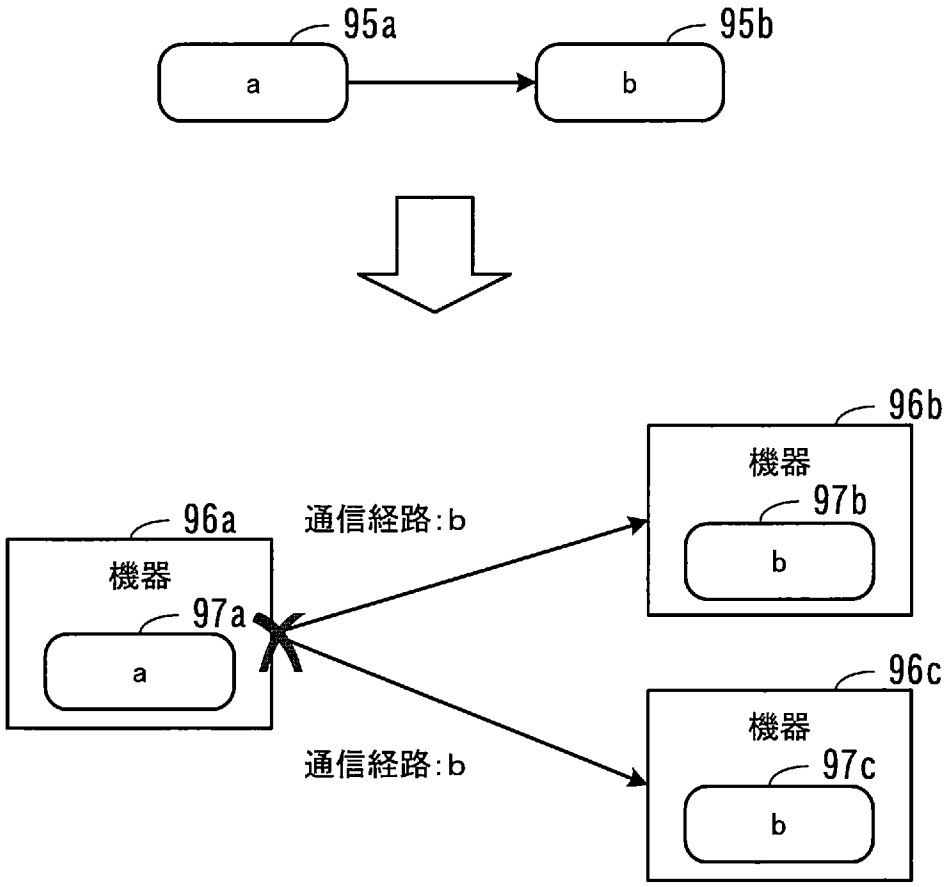
[図1]


10
情報処理装置
11
記憶部
1
全体処理定義データ
配備先:機器「A」
配備先:機器「B」
1a
第1データ処理定義情報
第2データ処理定義情報
1b
12
処理部
部分処理定義データ生成
1-1
部分処理定義データ(機器A用)
1a
第1データ処理定義情報
1c
データ送信定義情報(通信識別子:AtoB)
1-2
部分処理定義データ(機器B用)
1d
データ受信定義情報(通信識別子:AtoB)
1b
第2データ処理定義情報
部分処理定義データ送信
2
第1機器(機器識別子:A)
1-1
部分処理定義データ(機器A用)
4
ネットワーク
出力データ送信(通信識別子:AtoB)
3
第2機器(機器識別子:B)
1-2
部分処理定義データ(機器B用)

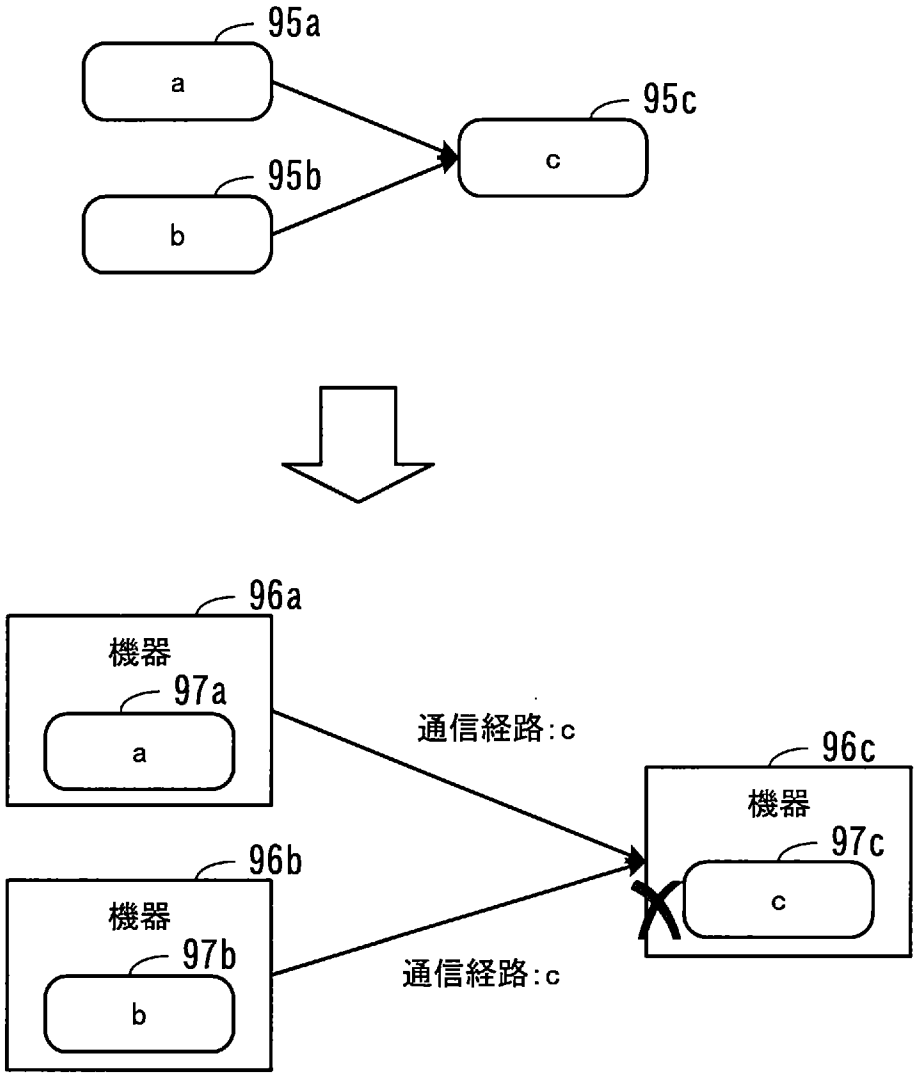
[図2]


91
全体アプリフロー編集画面
フロー
機能ブロック
92a
機能α
92b
機能β
93a
機能α
93b
機能β
93 全体アプリフロー
91a 機能ブロック表示領域
91b フロー表示領域
94 全体アプリフローデータ
データ処理定義情報 94a
データ処理定義情報 94b
"id": "a4bc8da8.7ce4e",
"type": "inject",
"z": "edfca64e.83d388",
"name": "機能α",
"topic": "",
"payload": "",
"payloadType": "date",
"repeat": "",
"crontab": "",
"once": false,
"x": 180,
"y": 160,
"wires": [
"566fcac9.d0f784"
]
"転送先ノード"
"id": "566fcac9.d0f784",
"type": "debug",
"z": "edfca64e.83d388",
"name": "機能β",
"active": true,
"console": "false",
"complete": "payload",
"x": 420,
"y": 160,
"wires": []

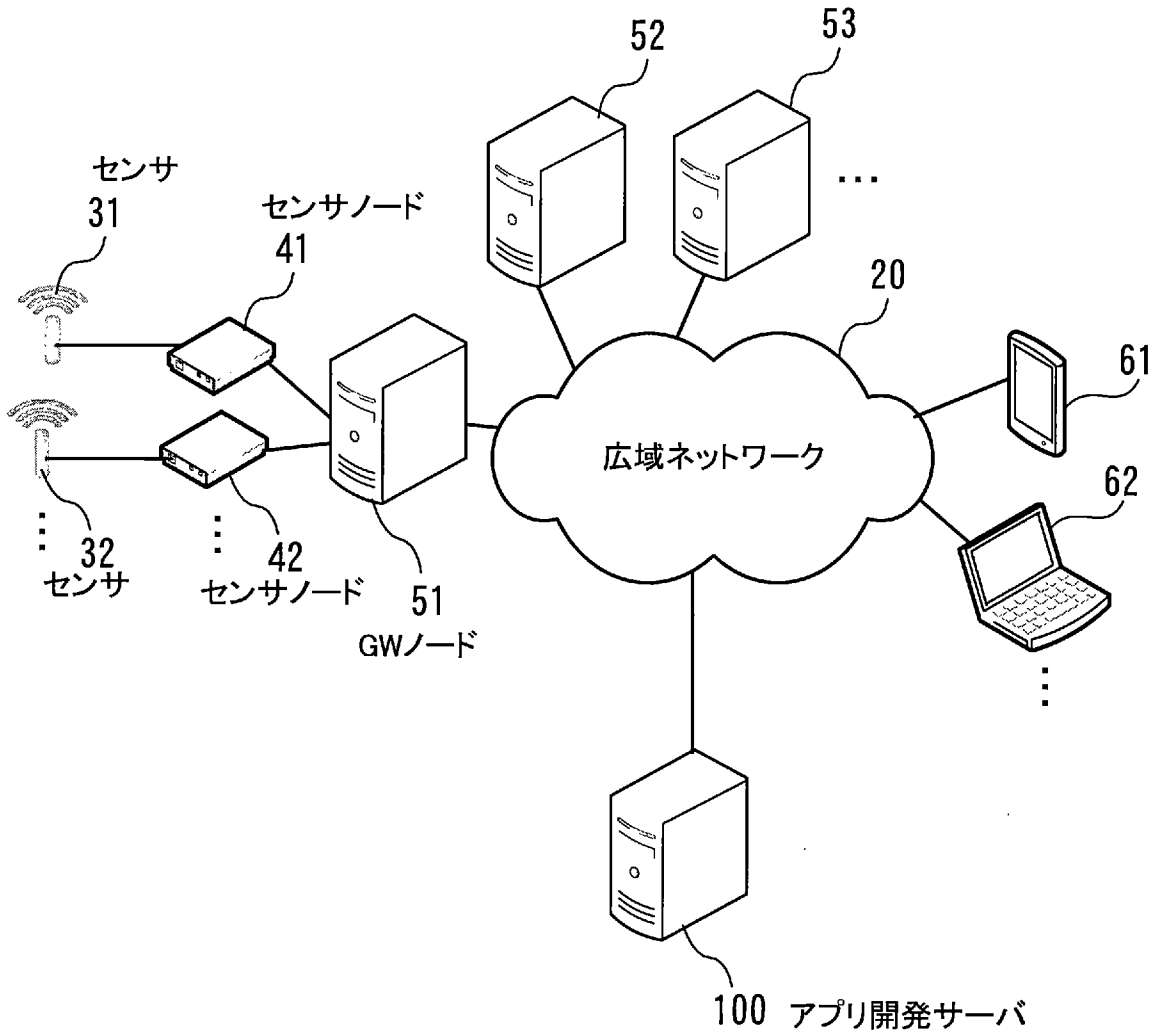
[図3]



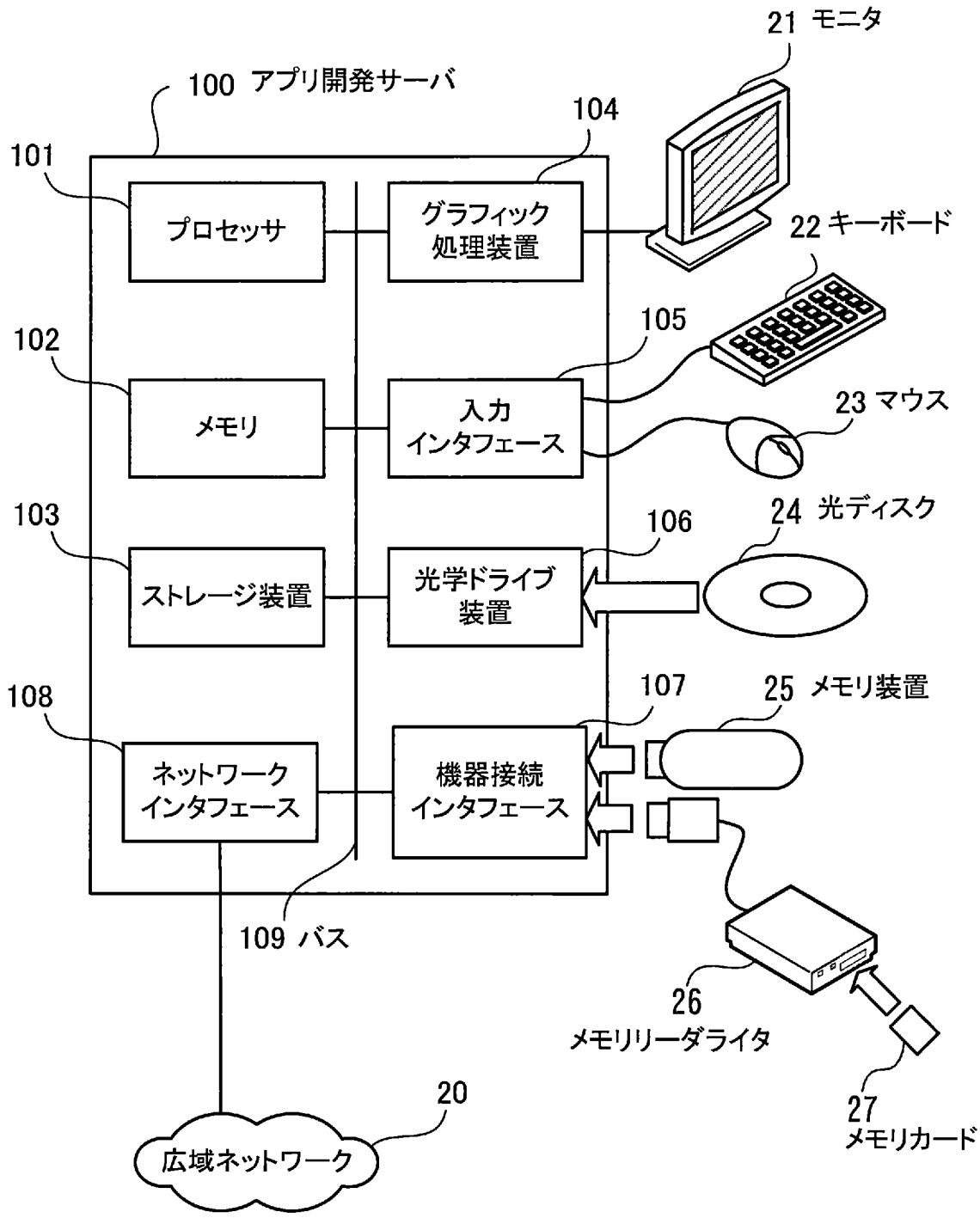
[図4]



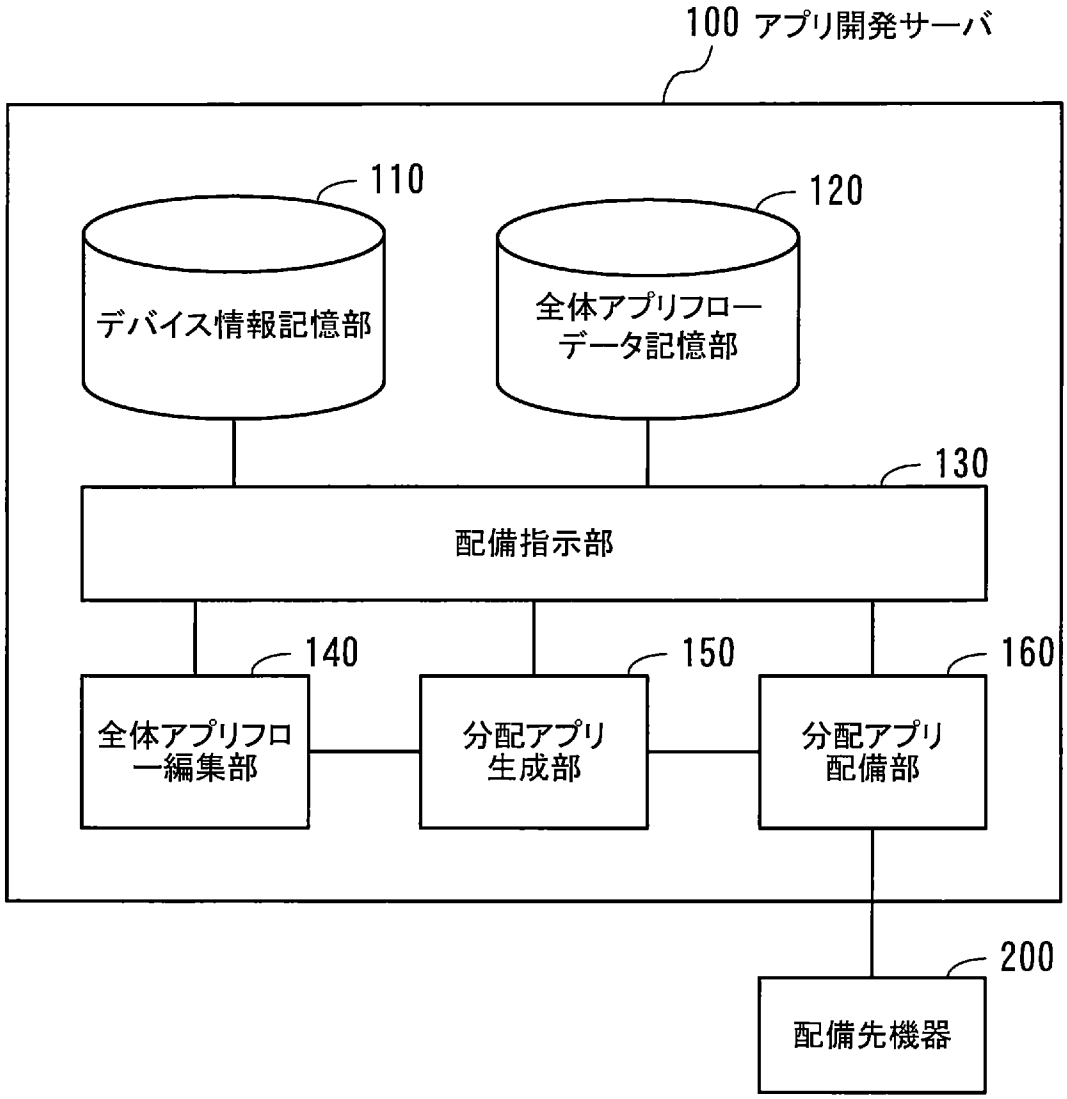
[図5]


52
53
センサ
31
センサノード
41
20
61
広域ネットワーク
62
32
センサ
42
センサノード
51
GWノード
100 アプリ開発サーバ

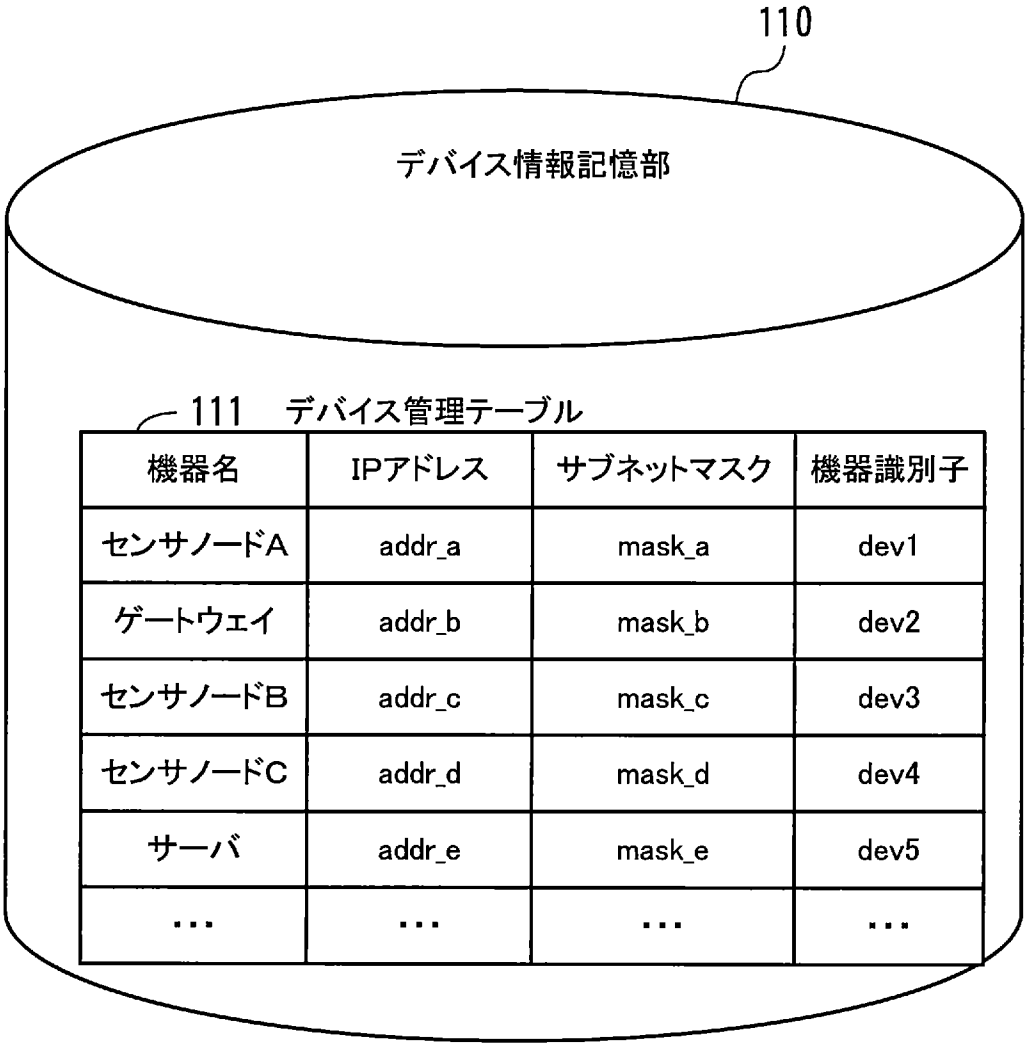
[図6]


21 モニタ
100 アプリ開発サーバ
104
101
プロセッサ
グラフィック
処理装置
22 キーボード
105
102
メモリ
入力
インタフェース
23 マウス
103
106
24 光ディスク
ストレージ装置
光学ドライブ
装置
107
25 メモリ装置
108
ネットワーク
インタフェース
機器接続
インタフェース
109 バス
26
メモリリーダライタ
27
メモリカード
20
広域ネットワーク

[図7]


100 アプリ開発サーバ
110
デバイス情報記憶部
120
全体アプリフロー
データ記憶部
130
配備指示部
140
全体アプリフロ
一編集部
150
分配アプリ
生成部
160
分配アプリ
配備部
200
配備先機器

[図8]



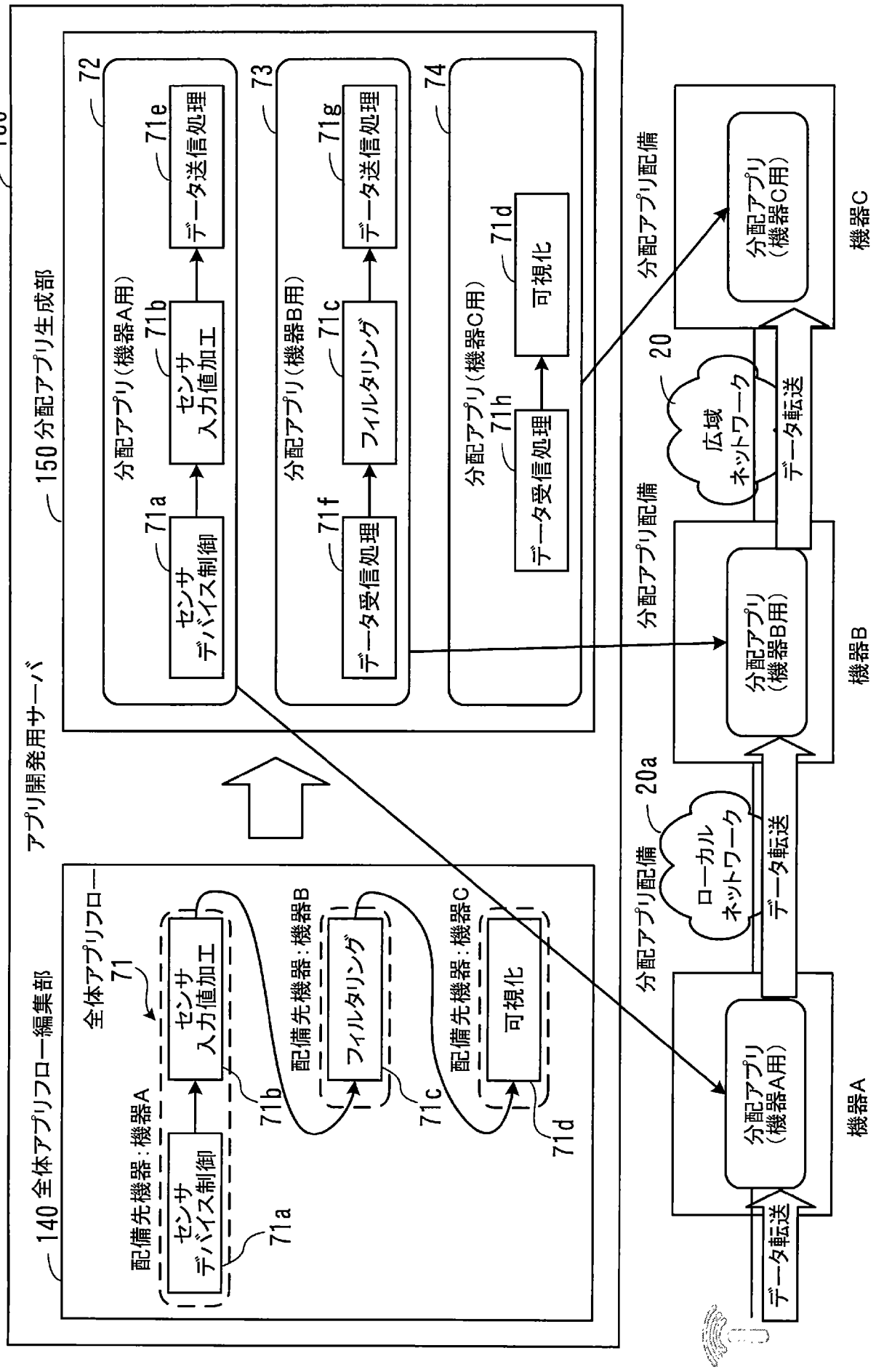
機器名	IPアドレス	サブネットマスク	機器識別子
センサノードA	addr_a	mask_a	dev1
ゲートウェイ	addr_b	mask_b	dev2
センサノードB	addr_c	mask_c	dev3
センサノードC	addr_d	mask_d	dev4
サーバ	addr_e	mask_e	dev5
・・・	・・・	・・・	・・・

[図9]

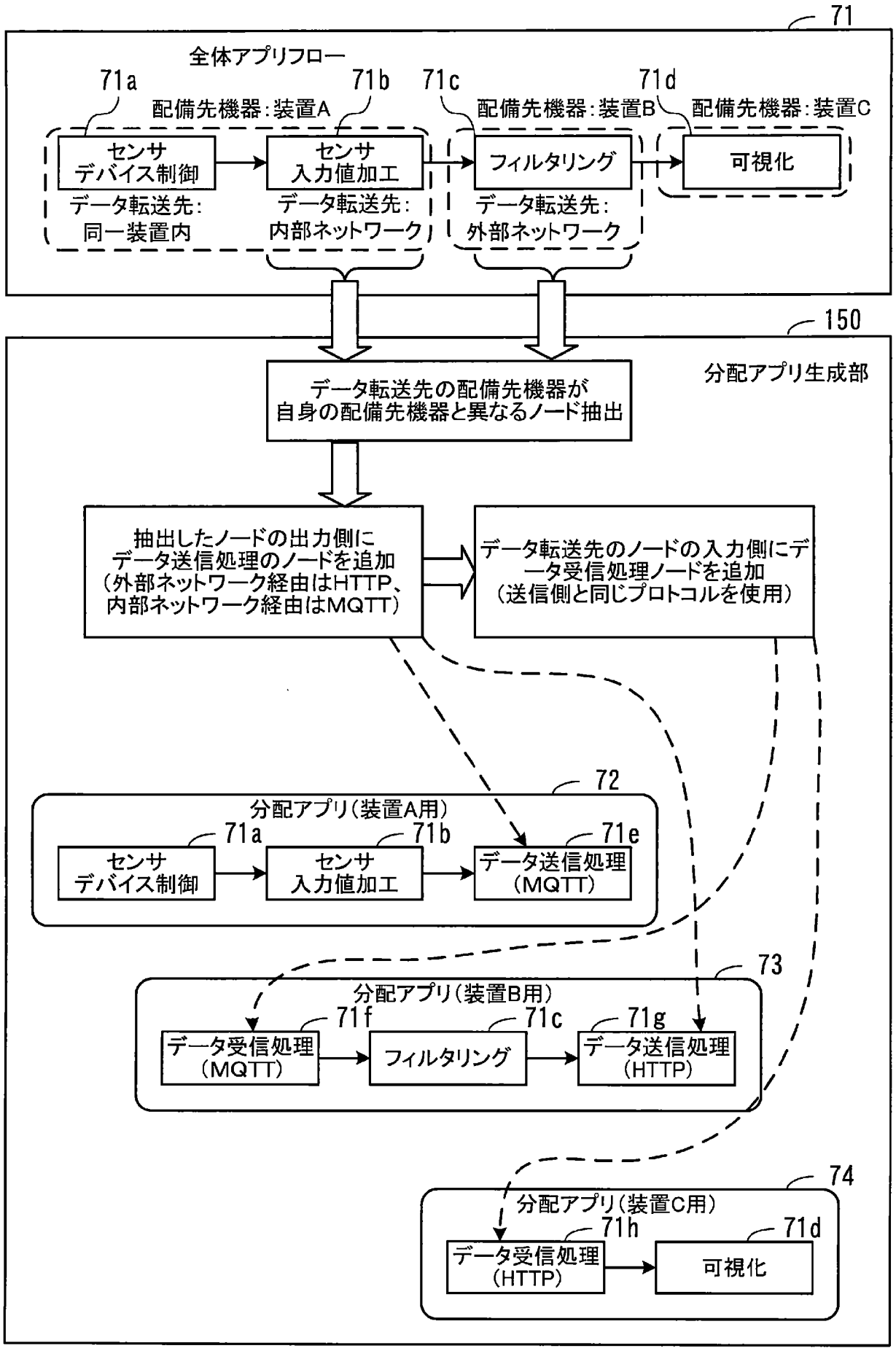



```
[
    {
        "id": "7ece43b4.7db15c",
        "type": "inject",
        "z": "893814eb.280428",
        "name": "日付データ生成",
        "topic": "",
        "payload": "",
        "payloadType": "date",
        "repeat": "",
        "crontab": "",
        "once": false,
        "x": 132,
        "y": 206,
        "wires": [
            [
                "273505d.97f64fa"
            ]
        ]
    },
    {
        "id": "273505d.97f64fa",
        "type": "function",
        "z": "893814eb.280428",
        "name": "日付データ整形",
        "func": "var date = new Date(msg.payload);
                ¥nmsg.payload = date.toString();¥nreturn msg;",
        "outputs": 1,
        "noerr": 0,
        "x": 378,
        "y": 207,
        "wires": [
            [
                "6c598d04.691054"
            ]
        ]
    },
    {
        "id": "6c598d04.691054",
        "type": "debug",
        "z": "893814eb.280428",
        "name": "日付データ出力",
        "active": true,
        "console": "false",
        "complete": "payload",
        "x": 632,
        "y": 208,
        "wires": []
    }
]
```

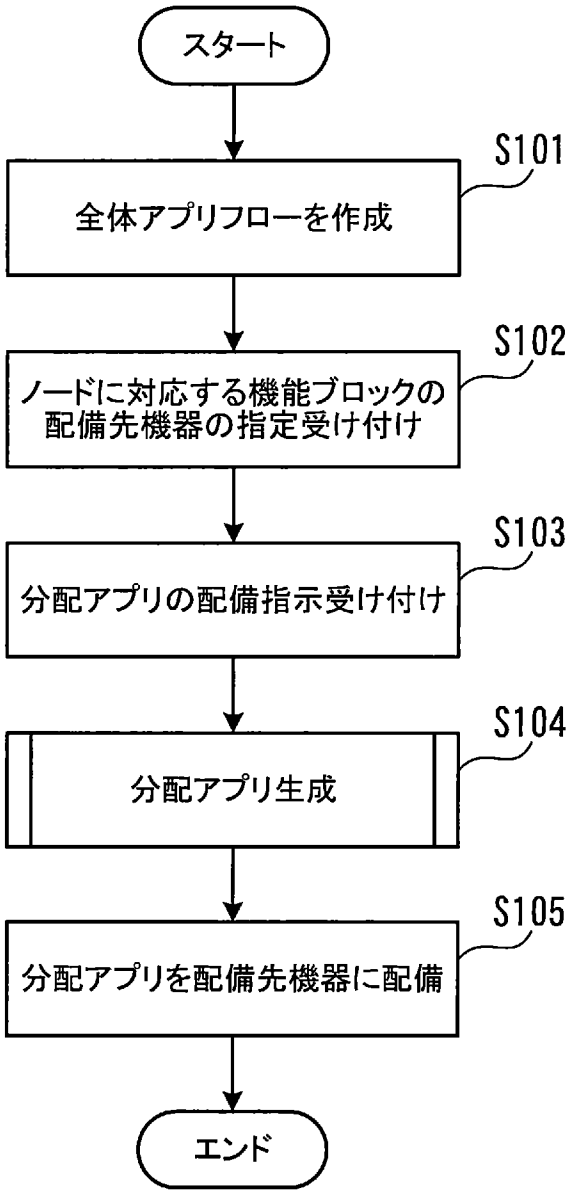
[図10]


100
アプリ開発用サーバ
150 分配アプリ生成部
72
分配アプリ（機器A用）
71e
データ送信処理
71b
センサ入力値加工
71a
センサデバイス制御
73
分配アプリ（機器B用）
71g
データ送信処理
71c
フィルタリング
71f
データ受信処理
74
分配アプリ（機器C用）
71d
可視化
71h
データ受信処理
140 全体アプリフロー編集部
全体アプリフロー
71
配備先機器：機器A
センサデバイス制御
センサ入力値加工
71a
71b
配備先機器：機器B
フィルタリング
71c
配備先機器：機器C
可視化
71d
分配アプリ配備
20a
ローカルネットワーク
データ転送
分配アプリ配備
20
広域ネットワーク
データ転送
分配アプリ配備
データ転送
分配アプリ（機器A用）
分配アプリ（機器B用）
分配アプリ（機器C用）
機器A
機器B
機器C

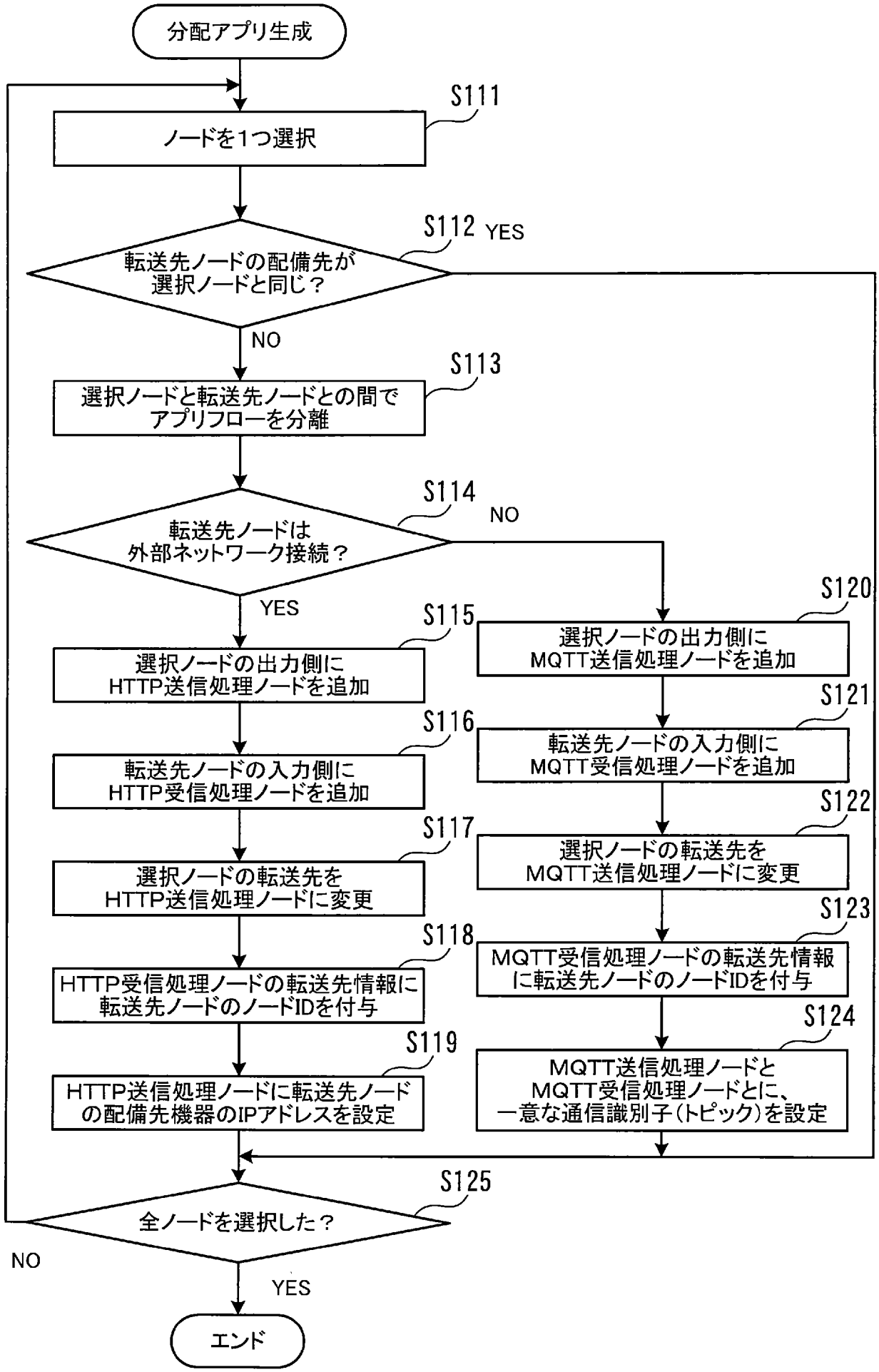
[図11]


71
全体アプリフロー
71a
71b
71c
71d
配備先機器:装置A
配備先機器:装置B
配備先機器:装置C
センサ
デバイス制御
センサ
入力値加工
フィルタリング
可視化
データ転送先:
同一装置内
データ転送先:
内部ネットワーク
データ転送先:
外部ネットワーク
150
分配アプリ生成部
データ転送先の配備先機器が
自身の配備先機器と異なるノード抽出
抽出したノードの出力側に
データ送信処理のノードを追加
(外部ネットワーク経由はHTTP、
内部ネットワーク経由はMQTT)
データ転送先のノードの入力側にデ
ータ受信処理ノードを追加
(送信側と同じプロトコルを使用)
72
分配アプリ(装置A用)
71a
71b
71e
センサ
デバイス制御
センサ
入力値加工
データ送信処理
(MQTT)
73
分配アプリ(装置B用)
71f
71c
71g
データ受信処理
(MQTT)
フィルタリング
データ送信処理
(HTTP)
74
分配アプリ(装置C用)
71h
71d
データ受信処理
(HTTP)
可視化

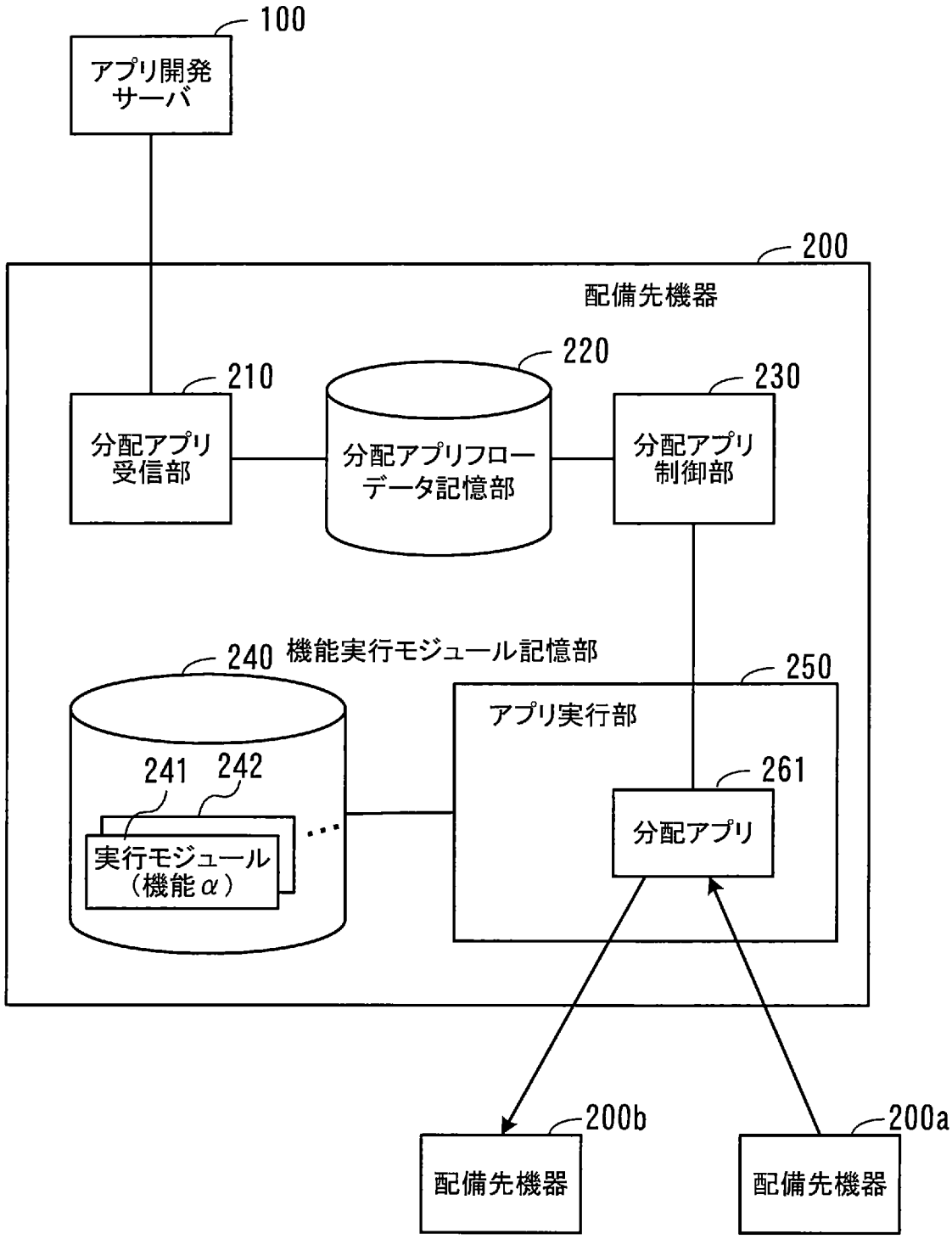
[図12]


スタート
S101
全体アプリフローを作成
S102
ノードに対応する機能ブロックの
配備先機器の指定受け付け
S103
分配アプリの配備指示受け付け
S104
分配アプリ生成
S105
分配アプリを配備先機器に配備
エンド

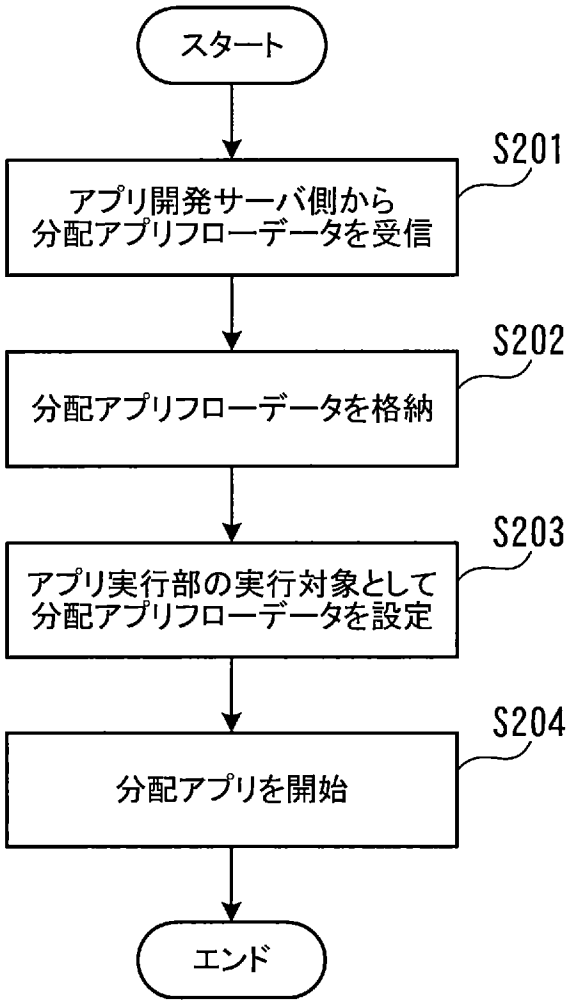
[図13]


分配アプリ生成
S111
ノードを1つ選択
S112
転送先ノードの配備先が選択ノードと同じ？
YES
NO
S113
選択ノードと転送先ノードとの間でアプリフローを分離
S114
転送先ノードは外部ネットワーク接続？
NO
YES
S115
選択ノードの出力側にHTTP送信処理ノードを追加
S116
転送先ノードの入力側にHTTP受信処理ノードを追加
S117
選択ノードの転送先をHTTP送信処理ノードに変更
S118
HTTP受信処理ノードの転送先情報に転送先ノードのノードIDを付与
S119
HTTP送信処理ノードに転送先ノードの配備先機器のIPアドレスを設定
S120
選択ノードの出力側にMQTT送信処理ノードを追加
S121
転送先ノードの入力側にMQTT受信処理ノードを追加
S122
選択ノードの転送先をMQTT送信処理ノードに変更
S123
MQTT受信処理ノードの転送先情報に転送先ノードのノードIDを付与
S124
MQTT送信処理ノードとMQTT受信処理ノードとに、一意な通信識別子（トピック）を設定
S125
全ノードを選択した？
NO
YES
エンド

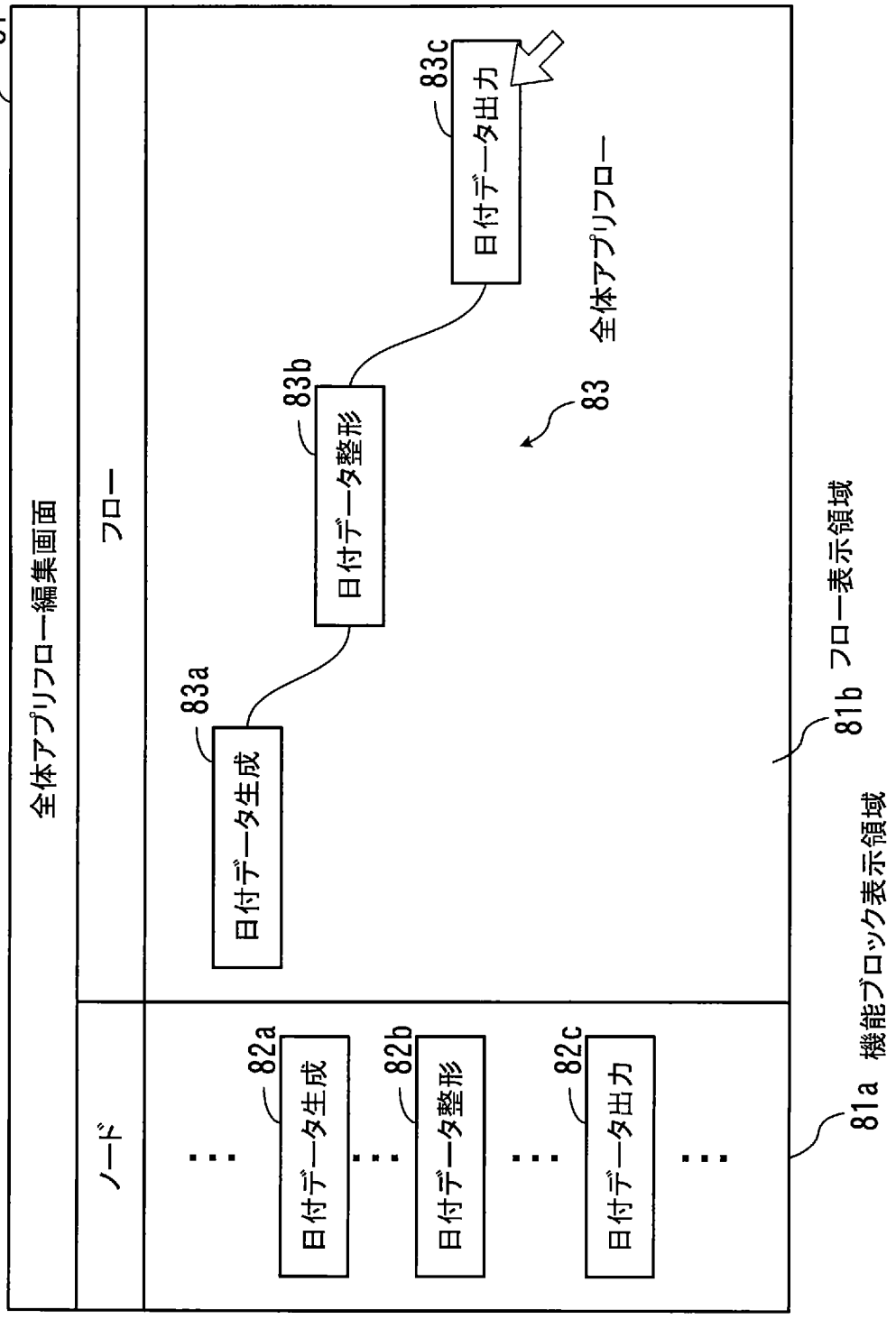
[図14]


100
アプリ開発
サーバ
200
配備先機器
210
分配アプリ
受信部
220
分配アプリフロー
データ記憶部
230
分配アプリ
制御部
240
機能実行モジュール記憶部
241
242
実行モジュール
（機能α）
250
アプリ実行部
261
分配アプリ
200b
配備先機器
200a
配備先機器

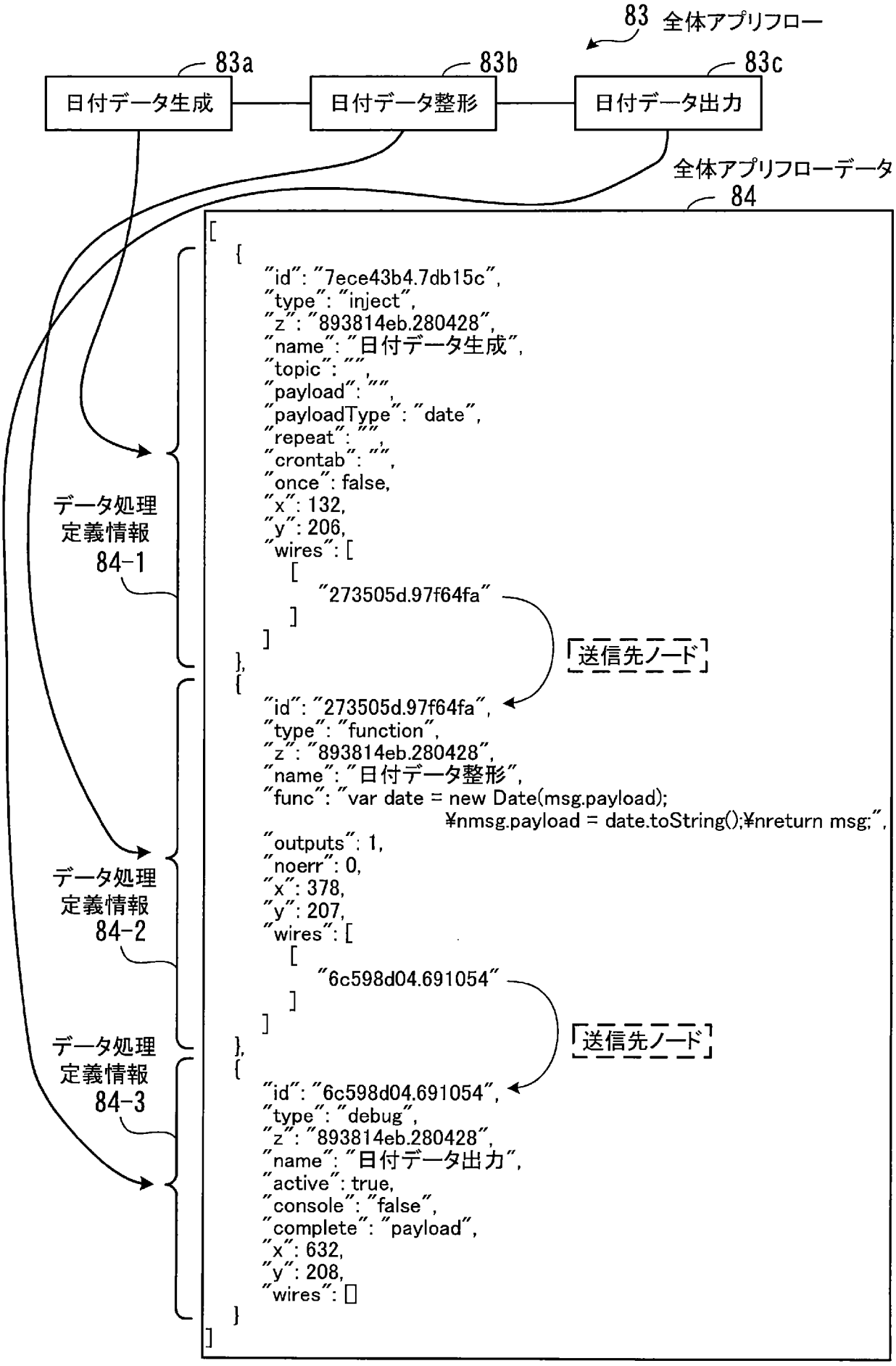
[図15]


スタート
S201
アプリ開発サーバ側から
分配アプリフローデータを受信
S202
分配アプリフローデータを格納
S203
アプリ実行部の実行対象として
分配アプリフローデータを設定
S204
分配アプリを開始
エンド

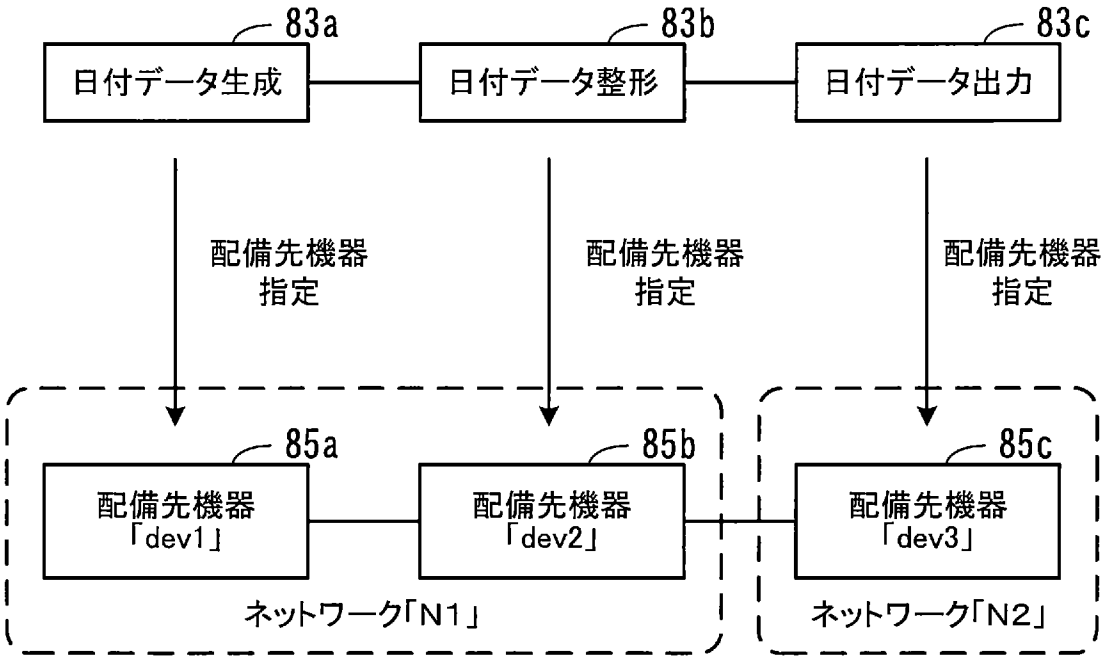
[図16]


81
全体アプリフロー編集画面
フロー
ノード
83a
日付データ生成
83b
日付データ整形
83c
日付データ出力
83 全体アプリフロー
81b フロー表示領域
82a
日付データ生成
82b
日付データ整形
82c
日付データ出力
81a 機能ブロック表示領域

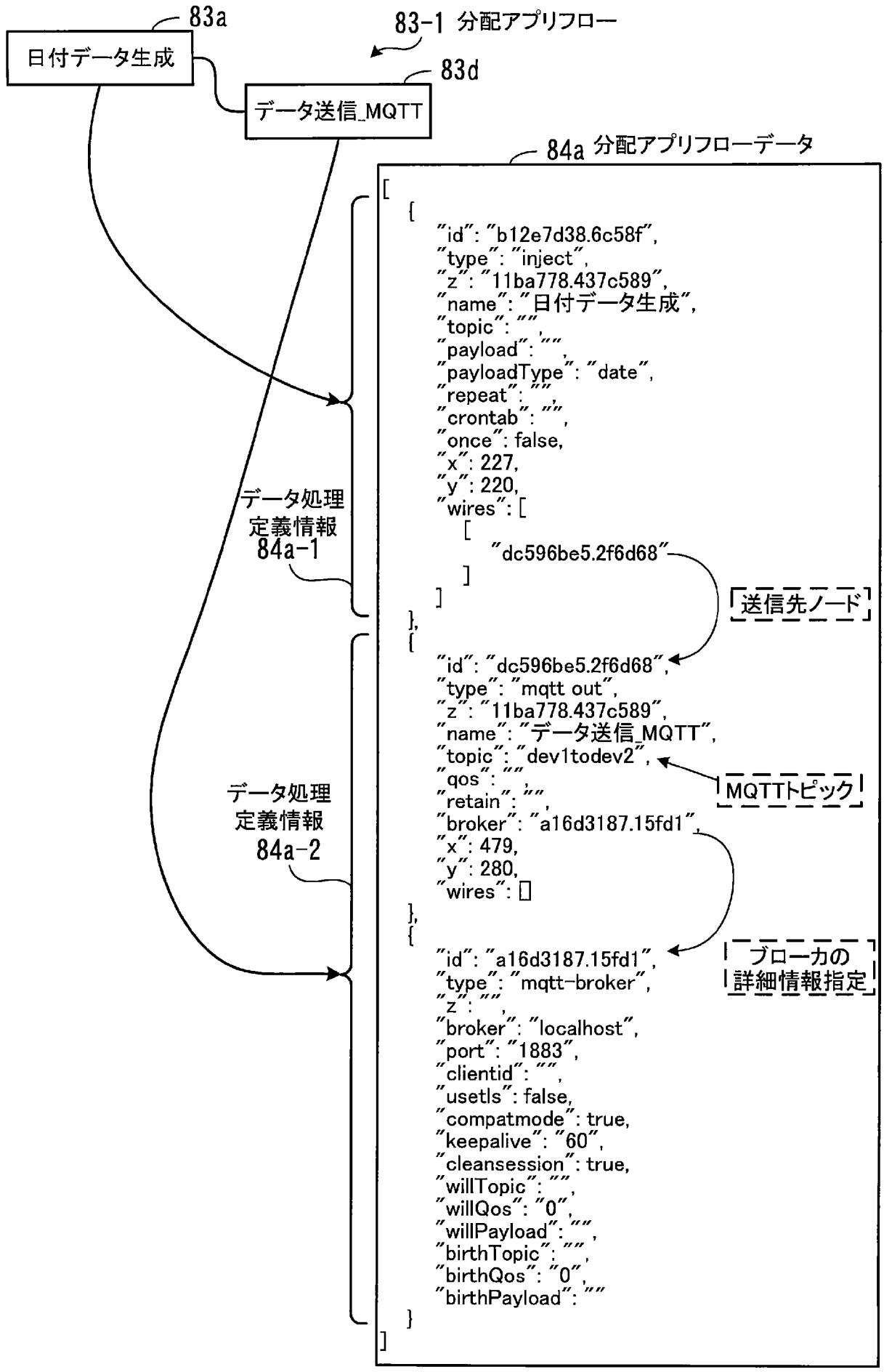
[図17]


83 全体アプリフロー
83a
83b
83c
日付データ生成
日付データ整形
日付データ出力
全体アプリフローデータ
84
データ処理
定義情報
84-1
データ処理
定義情報
84-2
データ処理
定義情報
84-3
送信先ノード
送信先ノード
[
{
"id": "7ece43b4.7db15c",
"type": "inject",
"z": "893814eb.280428",
"name": "日付データ生成",
"topic": "",
"payload": "",
"payloadType": "date",
"repeat": "",
"crontab": "",
"once": false,
"x": 132,
"y": 206,
"wires": [
[
"273505d.97f64fa"
]
]
},
{
"id": "273505d.97f64fa",
"type": "function",
"z": "893814eb.280428",
"name": "日付データ整形",
"func": "var date = new Date(msg.payload);
¥nmsg.payload = date.toString();¥nreturn msg;",
"outputs": 1,
"noerr": 0,
"x": 378,
"y": 207,
"wires": [
[
"6c598d04.691054"
]
]
},
{
"id": "6c598d04.691054",
"type": "debug",
"z": "893814eb.280428",
"name": "日付データ出力",
"active": true,
"console": "false",
"complete": "payload",
"x": 632,
"y": 208,
"wires": []
}
]

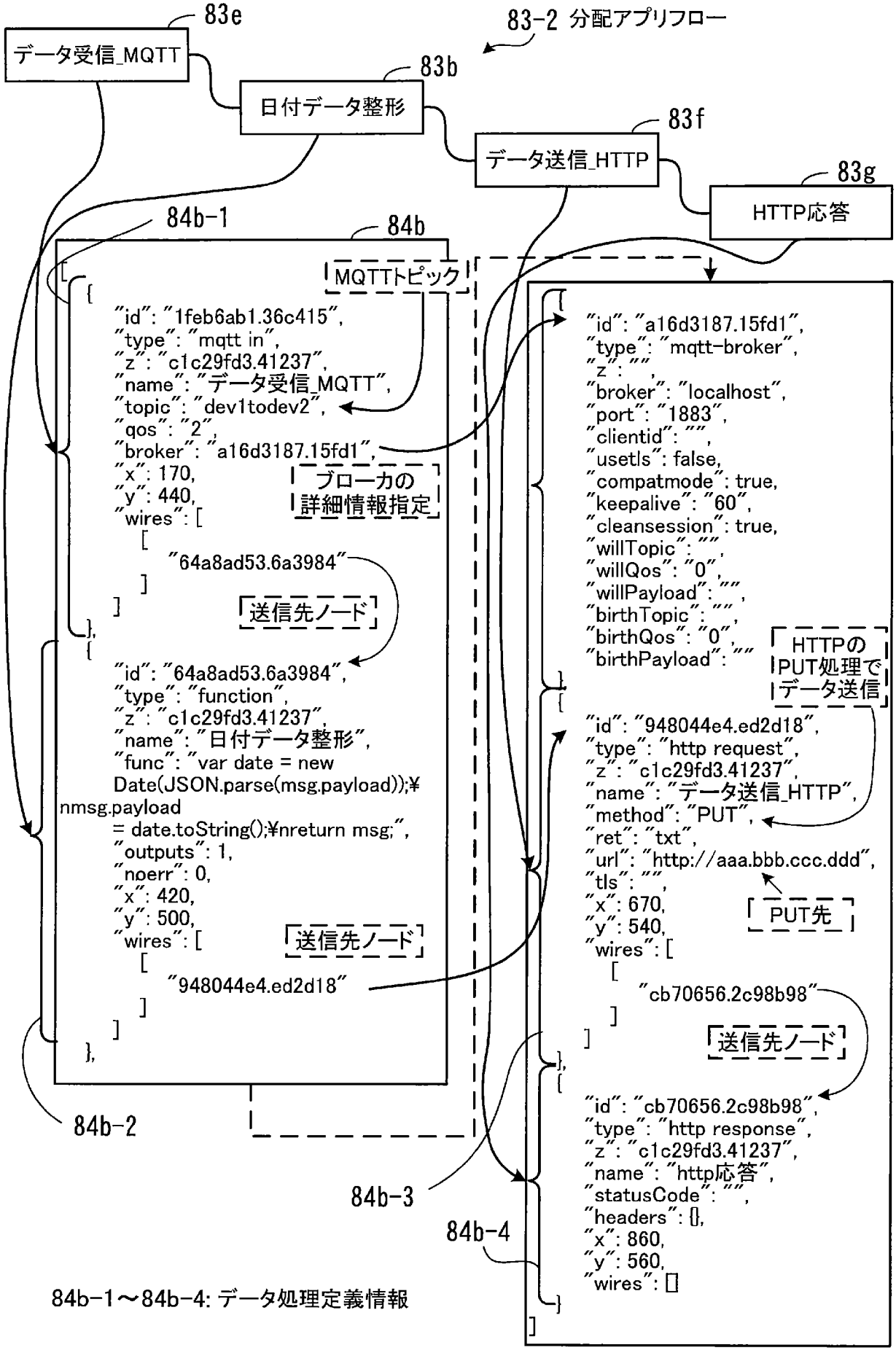
[図18]


83a
日付データ生成
83b
日付データ整形
83c
日付データ出力
配備先機器
指定
配備先機器
指定
配備先機器
指定
85a
配備先機器
「dev1」
85b
配備先機器
「dev2」
85c
配備先機器
「dev3」
ネットワーク「N1」
ネットワーク「N2」

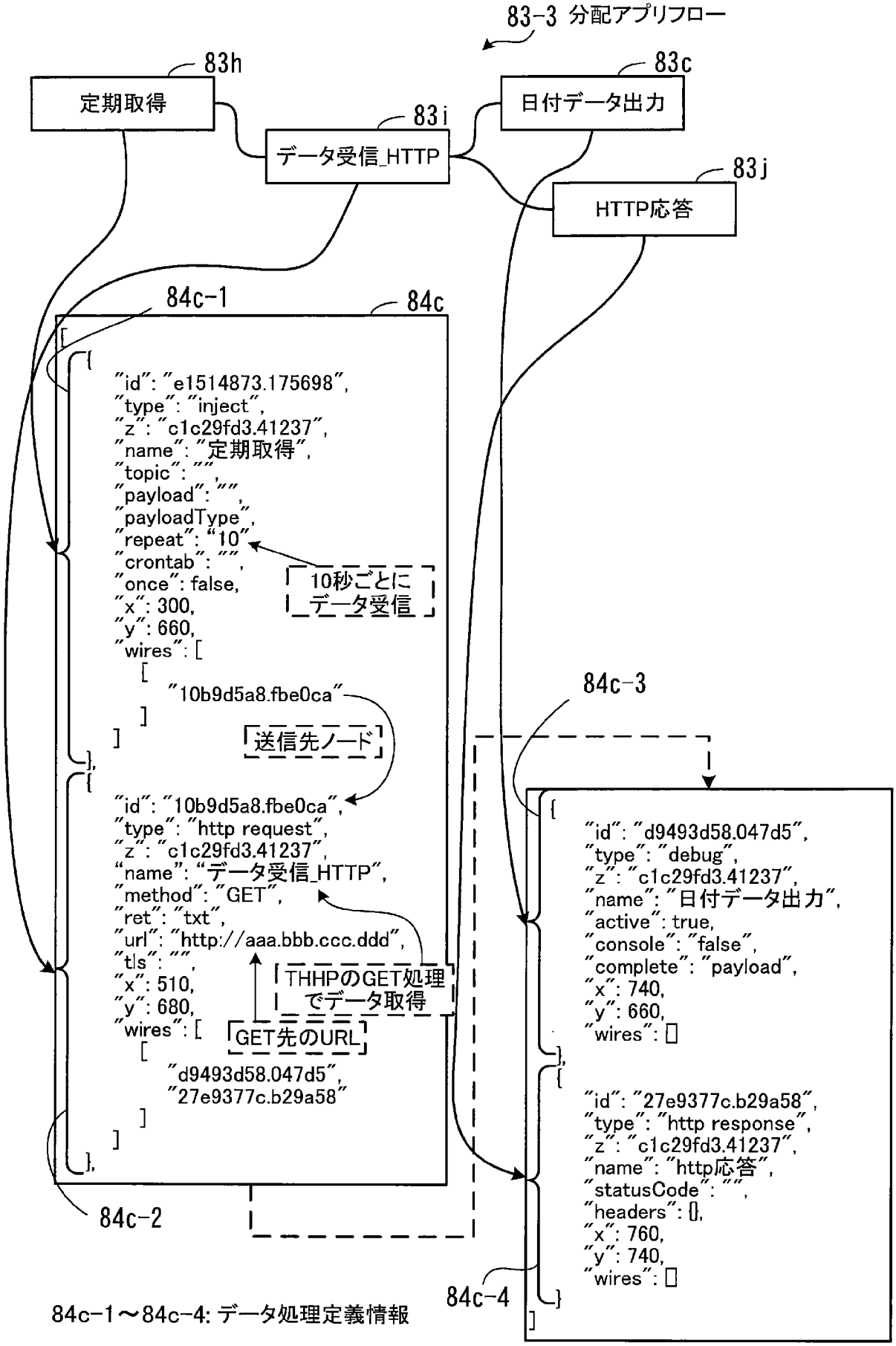
[図19]


83a
日付データ生成
83-1 分配アプリフロー
83d
データ送信_MQTT
84a 分配アプリフローデータ
データ処理
定義情報
84a-1
データ処理
定義情報
84a-2
送信先ノード
MQTTトピック
ブローカの
詳細情報指定
[
{
"id": "b12e7d38.6c58f",
"type": "inject",
"z": "11ba778.437c589",
"name": "日付データ生成",
"topic": "",
"payload": "",
"payloadType": "date",
"repeat": "",
"crontab": "",
"once": false,
"x": 227,
"y": 220,
"wires": [
[
"dc596be5.2f6d68"
]
]
},
{
"id": "dc596be5.2f6d68",
"type": "mqtt out",
"z": "11ba778.437c589",
"name": "データ送信_MQTT",
"topic": "dev1todev2",
"qos": "",
"retain": "",
"broker": "a16d3187.15fd1",
"x": 479,
"y": 280,
"wires": []
},
{
"id": "a16d3187.15fd1",
"type": "mqtt-broker",
"z": "",
"broker": "localhost",
"port": "1883",
"clientid": "",
"usetls": false,
"compatmode": true,
"keepalive": "60",
"cleansession": true,
"willTopic": "",
"willQos": "0",
"willPayload": "",
"birthTopic": "",
"birthQos": "0",
"birthPayload": ""
}
]

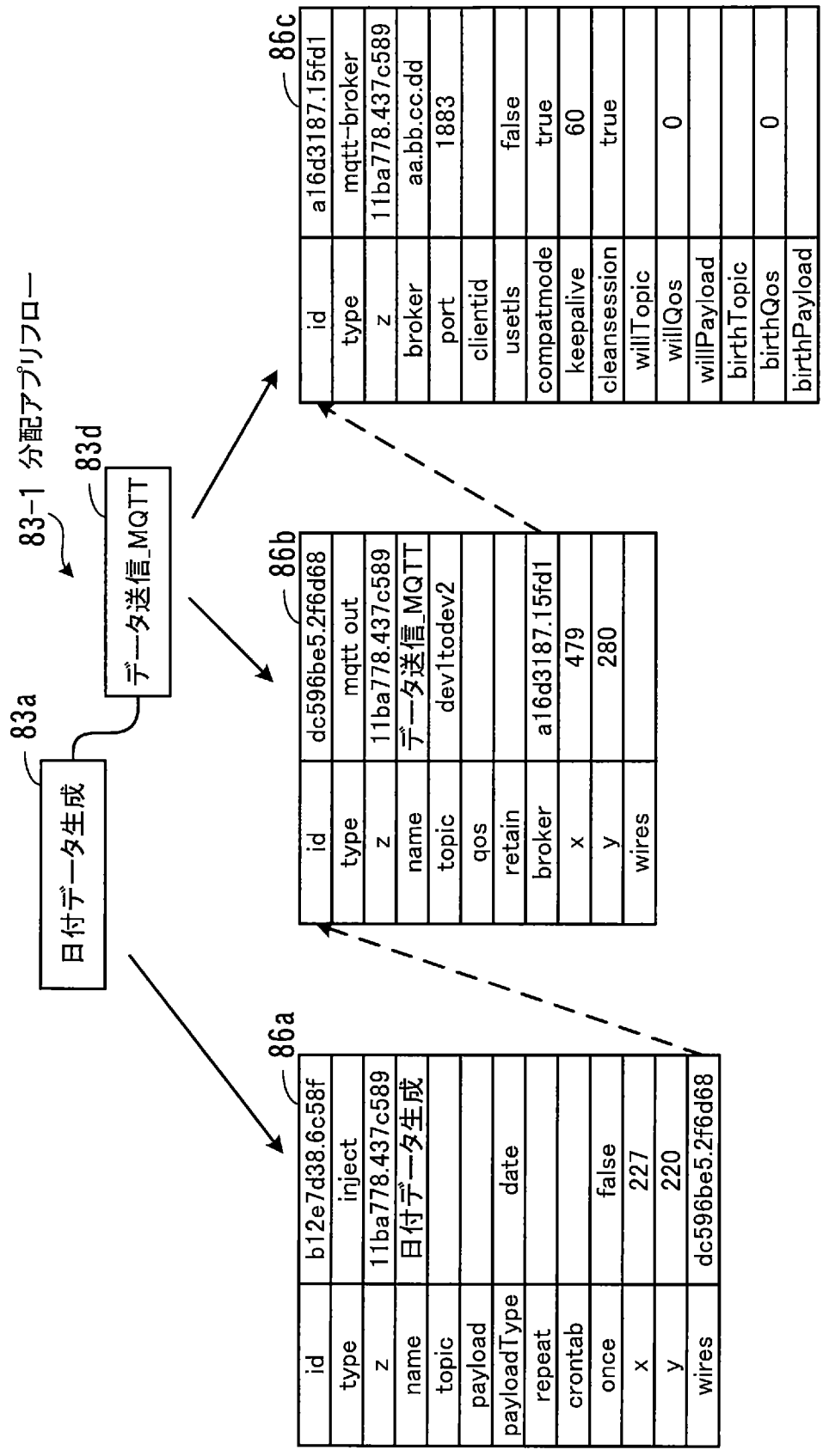
[図20]


83e
データ受信_MQTT
83-2 分配アプリフロー
83b
日付データ整形
83f
データ送信_HTTP
83g
HTTP応答
84b-1
84b
MQTTトピック
"id": "1feb6ab1.36c415",
"type": "mqtt in",
"z": "c1c29fd3.41237",
"name": "データ受信_MQTT",
"topic": "dev1todev2",
"qos": "2",
"broker": "a16d3187.15fd1",
"x": 170,
"y": 440,
"wires": [
"64a8ad53.6a3984"
]
ブローカの詳細情報指定
送信先ノード
"id": "64a8ad53.6a3984",
"type": "function",
"z": "c1c29fd3.41237",
"name": "日付データ整形",
"func": "var date = new Date(JSON.parse(msg.payload));¥nmsg.payload = date.toString();¥nreturn msg;",
"outputs": 1,
"noerr": 0,
"x": 420,
"y": 500,
"wires": [
"948044e4.ed2d18"
]
送信先ノード
"id": "a16d3187.15fd1",
"type": "mqtt-broker",
"z": "",
"broker": "localhost",
"port": "1883",
"clientid": "",
"usetls": false,
"compatmode": true,
"keepalive": "60",
"cleansession": true,
"willTopic": "",
"willQos": "0",
"willPayload": "",
"birthTopic": "",
"birthQos": "0",
"birthPayload": ""
HTTPのPUT処理でデータ送信
"id": "948044e4.ed2d18",
"type": "http request",
"z": "c1c29fd3.41237",
"name": "データ送信_HTTP",
"method": "PUT",
"ret": "txt",
"url": "http://aaa.bbb.ccc.ddd",
"tls": "",
"x": 670,
"y": 540,
"wires": [
"cb70656.2c98b98"
]
PUT先
送信先ノード
"id": "cb70656.2c98b98",
"type": "http response",
"z": "c1c29fd3.41237",
"name": "http応答",
"statusCode": "",
"headers": {},
"x": 860,
"y": 560,
"wires": []
84b-2
84b-3
84b-4
84b-1～84b-4: データ処理定義情報

[図21]


83-3 分配アプリフロー
83h
定期取得
83i
データ受信_HTTP
83c
日付データ出力
83j
HTTP応答
84c-1
84c
"id": "e1514873.175698",
"type": "inject",
"z": "c1c29fd3.41237",
"name": "定期取得",
"topic": "",
"payload": "",
"payloadType",
"repeat": "10",
"crontab": "",
"once": false,
"x": 300,
"y": 660,
"wires": [
"10b9d5a8.fbe0ca"
10秒ごとに
データ受信
送信先ノード
"id": "10b9d5a8.fbe0ca",
"type": "http request",
"z": "c1c29fd3.41237",
"name": "データ受信_HTTP",
"method": "GET",
"ret": "txt",
"url": "http://aaa.bbb.ccc.ddd",
"tls": "",
"x": 510,
"y": 680,
"wires": [
"d9493d58.047d5",
"27e9377c.b29a58"
THHPのGET処理
でデータ取得
GET先のURL
84c-2
84c-3
"id": "d9493d58.047d5",
"type": "debug",
"z": "c1c29fd3.41237",
"name": "日付データ出力",
"active": true,
"console": "false",
"complete": "payload",
"x": 740,
"y": 660,
"wires": []
"id": "27e9377c.b29a58",
"type": "http response",
"z": "c1c29fd3.41237",
"name": "http応答",
"statusCode": "",
"headers": {},
"x": 760,
"y": 740,
"wires": []
84c-4
84c-1～84c-4: データ処理定義情報

[図22]


83-1 分配アプリフロー
83a
日付データ生成
83d
データ送信_MQTT
86a
id b12e7d38.6c58f
type inject
z 11ba778.437c589
name 日付データ生成
topic
payload
payloadType date
repeat
crontab
once false
x 227
y 220
wires dc596be5.2f6d68
86b
id dc596be5.2f6d68
type mqtt out
z 11ba778.437c589
name データ送信_MQTT
topic dev1todev2
qos
retain
broker a16d3187.15fd1
x 479
y 280
wires
86c
id a16d3187.15fd1
type mqtt-broker
z 11ba778.437c589
broker aa.bb.cc.dd
port 1883
clientid
usetls false
compatmode true
keepalive 60
cleansession true
willTopic
willQos 0
willPayload
birthTopic
birthQos 0
birthPayload

INTERNATIONAL SEARCH REPORT

International application No.
PCT/JP2018/000795

A. CLASSIFICATION OF SUBJECT MATTER
Int. Cl. G06F8/00(2018.01)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. G06F8/00, G06F8/34

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 1971-2018
Registered utility model specifications of Japan 1996-2018
Published registered utility model applications of Japan 1994-2018

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.
X Y	BLACKSTOCK, Michael and LEA, Rodger, Toward a Distributed Data Flow Platform for the Web of Things (Distributed Node-RED), Proceedings of the 5th International Workshop on Web of Things (WoT 2014), ACM, 08 October 2014, pp. 34-39, ISBN978-1-4503-3066-4	1-3, 6-7, 9-11 4-5, 8
Y	JP 6-332828 A (HITACHI, LTD.) 02 December 1994, paragraphs [0024]-[0037], [0052]-[0054] (Family: none)	4-5, 8

☒ 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	Date of mailing of the international search report
22.02.2018	13.03.2018
Name and mailing address of the ISA/ Japan Patent Office 3-4-3, Kasumigaseki, Chiyoda-ku, Tokyo 100-8915, Japan	Authorized officer Telephone No.

Form PCT/ISA/210 (second sheet) (January 2015)

INTERNATIONAL SEARCH REPORT

International application No.
PCT/JP2018/000795

C (Continuation). DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
Y	田坂和之,今井尚樹,茂木信二,磯村学,井戸上彰, デジタル家電ネットワークにおける制御メッセージの経路制御方式, マルチメディア,分散,協調とモバイル(DICOMO2008)シンポジウム論文集[CD-ROM], 02 July 2008, vol. 2008, no. 1, pp. 1692-1699, ISSN 1882-0840 (TASAKA, Kazuyuki, IMAI, Naoki, MOTEGI, Shinji, ISOMURA, Manabu, IDOUE, Akira. Routing Control Method for Control Messages in the Digital Appliance Network. Multimedia, Distributed, Cooperative, and Mobile Symposium (DICOMO2008) [CD-ROM].)	8
A	JP 6-332689 A (HITACHI, LTD.) 02 December 1994, abstract, paragraphs [0115], [0116], fig. 34, 35 (Family: none)	1-11
A	JP 2000-305763 A (NEC CORP.) 02 November 2000, abstract, paragraphs [0036]-[0045] (Family: none)	1-11
A	JP 2015-222517 A (SEIKO EPSON CORP.) 10 December 2015, paragraphs [0021]-[0032], fig. 2 (Family: none)	1-11

国際調査報告

国際出願番号　ＰＣＴ／ＪＰ２０１８／０００７９５

Ａ．　発明の属する分野の分類（国際特許分類（ＩＰＣ））

Int.Cl.　G06F8/00(2018.01)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（ＩＰＣ））

Int.Cl.　G06F8/00, G06F8/34

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	１９７１－２０１８年
日本国実用新案登録公報	１９９６－２０１８年
日本国登録実用新案公報	１９９４－２０１８年

国際調査で使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
X Y	BLACKSTOCK, Michael and LEA, Rodger, Toward a Distributed Data Flow Platform for the Web of Things (Distributed Node-RED), Proceedings of the 5th International Workshop on Web of Things (WoT 2014), ACM, 2014.10.08, p.34-39, ISBN978-1-4503-3066-4	1-3, 6-7, 9-11 4-5, 8
Y	JP 6-332828 A（株式会社日立製作所）1994.12.02, 段落[0024]-[0037], [0052]-[0054]（ファミリーなし）	4-5, 8

☑ 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、国際出願日以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	国際調査報告の発送日
２２．０２．２０１８	１３．０３．２０１８

国際調査機関の名称及びあて先
日本国特許庁（ＩＳＡ／ＪＰ）
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）　５Ｂ　９２８８
坂庭　剛史
電話番号　０３－３５８１－１１０１　内線　３５４５

Ｃ（続き）．　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
Y	田坂和之、今井尚樹、茂木信二、磯村 学、井戸上 彰，デジタル家電ネットワークにおける制御メッセージの経路制御方式，マルチメディア，分散，協調とモバイル（ＤＩＣＯＭＯ２００８）シンポジウム論文集［ＣＤ－ＲＯＭ］，2008.07.02，Ｖｏｌ．２００８，Ｎｏ．１，ｐ．１６９２－１６９９，ISSN 1882-0840	8
A	JP 6-332689 A（株式会社日立製作所）1994.12.02, 要約，段落[0115]-[0116]，図34-35（ファミリーなし）	1-11
A	JP 2000-305763 A（日本電気株式会社）2000.11.02, 要約，段落[0036]-[0045]（ファミリーなし）	1-11
A	JP 2015-222517 A（セイコーエプソン株式会社）2015.12.10, 段落[0021]-[0032]，図２（ファミリーなし）	1-11