(19)日本国特許庁(JP)　(12)特　許　公　報(B2)　(11)特許番号
特許第4000718号
(P4000718)
(45)発行日　平成19年10月31日(2007.10.31)　(24)登録日　平成19年8月24日(2007.8.24)

(51)Int.Cl.　F I
G06F 9/54 (2006.01)　G06F 9/46 480Z

請求項の数 12 (全 16 頁)

(21)出願番号　特願平11-177755
(22)出願日　平成11年6月24日(1999.6.24)
(65)公開番号　特開2001-5652(P2001-5652A)
(43)公開日　平成13年1月12日(2001.1.12)
審査請求日　平成16年3月26日(2004.3.26)

(73)特許権者　000005108
株式会社日立製作所
東京都千代田区丸の内一丁目６番６号
(74)代理人　100100310
弁理士　井上　学
(72)発明者　鮫嶋　茂稔
神奈川県川崎市麻生区王禅寺１０９９番地
株式会社日立製作所　システム開発研究所内
(72)発明者　河野　克己
神奈川県川崎市麻生区王禅寺１０９９番地
株式会社日立製作所　システム開発研究所内

最終頁に続く

(54)【発明の名称】プログラム間バインド方法及び分散処理システム

(57)【特許請求の範囲】
【請求項１】
伝送媒体に接続された複数の処理装置からなり、各々の前記複数の処理装置は、該複数の処理のそれぞれを実行するためのプログラムを記憶し、互いのデータ交換により前記複数の処理を行う分散処理システムによるプログラム間バインド方法において、
前記プログラムの開発時に入力された情報に基づき、前記プログラムが更新または送信または受信する、前記プログラム間で共有するデータに関する共有データ情報を抽出するステップと、
該共有データ情報を用いてプログラム間で交換するデータの構造を作成し、作成された前記データ構造を前記複数の処理装置で共有するステップと、
該作成された前記データ構造に従ってプログラム間で交換するデータを送受信するステップと、を有し、
前記抽出するステップは、
前記共有データとして、前記プログラムの開発時に入力された情報に含まれるプログラムソースコードから変数を抽出し、
前記共有データ情報として、前記プログラムの開発時に入力された情報に含まれる変数定義情報から、前記変数のデータ名とデータ型とを抽出し、
前記共有データ情報として、前記プログラムの開発時に入力された情報から、前記変数を書き込むプログラム及びそのプログラムが格納される処理装置と、前記変数を読み込むプログラム及びそのプログラムが格納される処理装置と、前記変数に対する読み書きの反

映方法を示す共有タイプとを抽出し、
さらに、前記共有データ情報として、前記共有データの組み合わせや前記プログラム間でのデータ交換のタイミングに応じて、前記プログラム間でデータ交換する場合の入出力単位毎の識別子を決定し、
前記共有データの組み合わせは、前記変数を書き込むプログラムが格納される処理装置毎、前記変数を書き込むプログラムとそのプログラムが格納される処理装置、前記共有データ情報の抽出の対象とされたプログラムグループと前記変数を書き込むプログラムとそのプログラムが格納される処理装置、前記共有データ情報の抽出の対象とされたプログラムグループと前記変数を書き込むプログラムとそのプログラムが格納される処理装置と前記変数を読み込むプログラムとそのプログラムが格納される処理装置の組み合わせを含み、
前記データを送受信するステップにおける送信タイミングは、周期、共有データの更新時、プログラム終了時、のいずれかであることを特徴とするプログラム間バインド方法。
【請求項２】
請求項１に記載のプログラム間バインド方法において、
前記抽出するステップは、
前記変数に対する出力があるか否かを判定し。
前記変数に対する出力がある場合は前記変数を書き込むプログラム及びそのプログラムが格納される処理装置と前記共有タイプとを抽出し、
前記変数に対する出力がない場合は前記変数を読み込むプログラム及びそのプログラムが格納される処理装置を抽出することを特徴とするプログラム間バインド方法。
【請求項３】
請求項１に記載のプログラム間バインド方法において、
前記プログラム間で交換する前記データ構造は、共有変数名、共有変数属性、手続き名、手続き引数、手続き引数の属性のいずれかを用いて作成されることを特徴とするプログラム間バインド方法。
【請求項４】
請求項１に記載のプログラム間バインド方法において、
前記プログラム間で交換する前記データ構造は、全ての前記プログラム、前記複数の処理を行う１つまたは複数の前記プログラム、特定の１つまたは複数の前記処理装置に属する前記プログラムのいずれかより抽出することを特徴とするプログラム間バインド方法。
【請求項５】
請求項１から３のいずれか一に記載のプログラム間バインド方法において、
前記プログラム間で交換する前記データ構造は、１つの共有データ、送信側のプログラムの書込みデータ、送信側の処理装置の書込みデータ、送信側のプログラムが書込みかつ受信側のプログラムが読込むデータのいずれかを１つに纏めたものであることを特徴とするプログラム間バインド方法。
【請求項６】
請求項１から３のいずれか一に記載のプログラム間バインド方法において、さらに、
各々の前記プログラムのもつ共有変数のうち値を共有する共有変数を設定するステップを備えることを特徴とするプログラム間バインド方法。
【請求項７】
請求項１、２又は６に記載のプログラム間バインド方法において、
前記変数は、グローバル変数であることを特徴とするプログラム間バインド方法。
【請求項８】
請求項１、２、６又は７に記載のプログラム間バインド方法において、
前記プログラム間で交換する前記データ構造は、前記データ名と前記データ型と前記識別子を含むことを特徴とするプログラム間バインド方法。
【請求項９】
請求項８に記載のプログラム間バインド方法において、

前記プログラム間で交換する前記データ構造は、さらに、処理プログラム実行環境から得られたデータ値とデータサイズを含むことを特徴とするプログラム間バインド方法。

【請求項１０】

請求項１に記載のプログラム間バインド方法において、

前記抽出するステップは、前記プログラムから他処理を呼び出す個所を抽出し、前記共有データとして前記プログラムから前記他処理の呼び出しにおける引数と、前記他処理からの戻り値を抽出することを特徴とするプログラム間バインド方法。

【請求項１１】

伝送媒体に接続された複数の処理装置からなり、各々の前記複数の処理装置は、該複数の処理のそれぞれを実行するためのプログラムを記憶し、互いのデータ交換により前記複数の処理を行う分散処理システムにおいて、

前記プログラムの開発時に入力された情報に基づき、前記プログラムが更新または送信または受信する、前記プログラム間で共有するデータに関する共有データ情報を抽出する手段と、

該共有データ情報を用いてプログラム間で交換するデータの構造を作成し、作成された前記データ構造を前記複数の処理装置のうちの他の処理装置と共有する手段と、

該作成された前記データ構造に従ってプログラム間で交換するデータを送受信する手段と、を有し、

前記抽出する手段は、

前記共有データとして、前記プログラムの開発時に入力された情報に含まれるプログラムソースコードから変数を抽出する手段と、

前記共有データ情報として、前記プログラムの開発時に入力された情報に含まれる変数定義情報から、前記変数のデータ名とデータ型とを抽出する手段と、

前記共有データ情報として、前記プログラムの開発時に入力された情報から、前記変数を書き込むプログラム及びそのプログラムが格納される処理装置と、前記変数を読み込むプログラム及びそのプログラムが格納される処理装置と、前記変数に対する読み書きの反映方法を示す共有タイプとを抽出する手段と、

さらに、前記共有データ情報として、前記共有データの組み合わせや前記プログラム間でのデータ交換のタイミングに応じて、前記プログラム間でデータ交換する場合の入出力単位毎の識別子を決定する手段とを有し、

前記共有データの組み合わせは、前記変数を書き込むプログラムが格納される処理装置毎、前記変数を書き込むプログラムとそのプログラムが格納される処理装置、前記共有データ情報の抽出の対象とされたプログラムグループと前記変数を書き込むプログラムとそのプログラムが格納される処理装置、前記共有データ情報の抽出の対象とされたプログラムグループと前記変数を書き込むプログラムとそのプログラムが格納される処理装置と前記変数を読み込むプログラムとそのプログラムが格納される処理装置の組み合わせを含み、

前記データを送受信するステップにおける送信タイミングは、周期、共有データの更新時、プログラム終了時、のいずれかであることを特徴とする分散処理システム。

【請求項１２】

請求項１１に記載の分散処理システムにおいて、

前記抽出する手段は、

前記変数に対する出力があるか否かを判定する手段と、

前記変数に対する出力がある場合は前記変数を書き込むプログラム及びそのプログラムが格納される処理装置と前記共有タイプとを抽出する手段と、

前記変数に対する出力がない場合は前記変数を読み込むプログラム及びそのプログラムが格納される処理装置を抽出する手段とを有することを特徴とする分散処理システム。

【発明の詳細な説明】

【発明の属する技術分野】

本発明は、複数のプログラムにより一連の処理が行われる分散処理システムのプログラム

間の連携方法に関し、特にプログラム開発時に入力された外部インタフェースに関する情報を用いて、プログラム間でデータ交換を行う場合に好適に適合しうるプログラム間バインド方法に関するものである。

【従来の技術】

分散処理システムにおけるプログラム間のバインド方式に関しては、メッセージ交換によるものがあり、例えば日経コンピュータ（1995年9月4日）P.P.126 - 139「メッセージ連携システム」にあるようなメッセージ連携ミドルウェアが分散処理装置間でのメッセージ送受信を行う。これは分散処理システムを構成するプログラム間で交換されるデータを、ミドルウェアが仲介役となって交換することで、プログラム間を非同期に連携させ、プログラムの開発や運用を柔軟にするものである。ここで、メッセージデータの構成は、アプリケーション間で個別に取り決めて設計を行い、プログラムでは、専用のAPIを用いてメッセージの交換を行う。

他にプログラム間を連携させる方法としては、例えば「Inside CORBA－CORBAとそのシステム開発への応用」(ISBN4-7561-2015-6)にあるようにオブジェクト指向プログラム間のデータ送受信プラットフォームなどもある。これによると、受信側プログラム（オブジェクト）の手続きをインタフェースとして規定し、該インタフェースを呼び出す形式でプログラム間でメッセージデータを交換し、プログラム間での連携を可能としている。

また、メッセージデータの送信を、アプリケーションからの指示と独立して行う方法として、周期的に指定領域のデータを送信する転写メモリ方式などがあり、例えば「日立　ＨＩＤＩＣ　Ｓ１０αシリーズ マニュアル」などにある。この方法では、転写メモリ上のアドレスを指定してデータの読み書きを行うことで、プログラム間でのデータ共有を可能としている。

【発明が解決しようとする課題】

分散処理システム内のプログラム間をメッセージデータの交換により連携させる方法は、プログラム間を疎結合化し、運用を容易にすることで普及している。一方で、このメッセージデータの設計は、連携するプログラム間で取り決めが必要である。このためプログラム間を連携させるためには、開発時に交換するデータ構成を合せるか、または一方のプログラムに合せて連携する他方のプログラムの送信または受信するデータ構成を変更する必要があり、開発時、改造時に工数を要していた。プログラム間で共有するデータは、各々のプログラムでの処理に用いるデータであるにも関わらず、再度メッセージとしての設計が必要であり、プログラマにとって設計の手間が必要であった。

また、設計したメッセージに対して、プログラムで渡すデータの設定を誤ったり、プログラムの変更によりメッセージとプログラムでの設定のバージョンが異なる場合があり、トラブルを引き起こす可能性があった。

さらに、過去開発したシステムや他人の開発したシステムにおいては、システムの改造を行う場合、メッセージ化される情報やメッセージとして受け取る情報が何かを調査する必要がある。このために設計ドキュメントを検索したり、ソースコードを詳細に調査する必要があるため大変手間がかかってしまう。

本発明は上記問題点を解決し、プログラム作成時に入力した情報を用いて、プログラム間で送受信されるデータの設計と送受信を行い、処理装置に分散されたプログラム間のバインドを容易に、かつ互いにインタフェースが異なる場合でも改造なく連携を可能とするものである。

【課題を解決するための手段】

本発明のプログラム間バインド方法は、プログラム間で交換するデータを検出するために、プログラム開発情報を用いて、プログラム間で共有するデータに関する情報を抽出する手段と、データの更新を行うプログラムと該データを利用するプログラム間でインタフェースが異なる場合でもデータを共有できるよう、前記手段で抽出した情報を用いてメッセージデータを構成する手段と、

該構造に従ってプログラム間で交換するデータを送受信する手段とを有するものである。

これによりプログラムと別にデータの構成、送受信が行われるため、プログラム作成と別

にメッセージ設計を行うことなく、またプログラムを改造することなく、インタフェースの異なるプログラム間で情報共有することを可能とする。

【発明の実施の形態】

以下に、本発明の実施の形態を詳細に説明する。特に、プログラム間で交換するデータの抽出方法と、共通交換形式データの構成方法及び送信方法に関して以下の例について説明する。

（１）グローバル変数を用いて共有する場合の、該変数抽出による共通交換形式データの作成と送信方法

（２）分散オブジェクトやリモート・プロシージャ・コール（Remote Procedure Call）などの処理プログラムインタフェースを用いて共有する場合の、該処理呼出し引数と戻り値抽出による共通交換形式データの作成と送信方法

（３）ツールによる設定方法

図１は、本発明を適用したシステムの構成例である。伝送媒体101を介して互いにデータの授受を行なう処理装置111～113と、外部との入出力インタフェースを有する外部入出力装置121～123から構成されている。処理装置111～113には、端末121～123が接続されている。ここで、外部入出力装置とは、ディスプレイやキーボード、タッチパネル等のマンマシンインタフェースを介して処理装置上で実行されるプログラムの制御やプログラムの出力参照を行ったり、制御対象機械との間の制御信号入出力を行う機能を有している。ただしこれは必須ではなく、外部入出力装置を持たない処理装置もある。また処理装置111は伝送媒体102を介して処理装置114とデータの授受を行なう。このように各処理装置は複数の伝送媒体を接続可能であり、処理装置111が仲介することで処理装置114と処理装置112, 113がデータの授受を行なうこともできる。

図２は、処理装置111～114の内任意の処理装置におけるソフトウェア構成を示す図である。データ送受信処理201は、自処理装置内、または他処理装置との間でプログラム間の共通交換形式データの送受信を管理するプログラムである。処理プログラム実行環境202は、処理装置内の処理を行うプログラム211の実行や起動、停止、データ入出力などを管理する。データ送受信処理201と処理プログラム実行環境202は互いにデータの交換や、処理の起動・終了といったイベントの通知を行い、他処理装置との送受信データと処理プログラム使用データの変換を行う。他処理装置とのデータの送受信方法については、TCP/IPのようなコネクションベースの方法を用いてもよいし、特開昭５６－１１１３５３号に示されたようなブロードキャスト方式を用いてもよい。

処理プログラムは、プログラム開発処理221で開発され、ソースプログラムなどの開発情報は、処理プログラム開発情報ファイル241に格納される。共有データ情報抽出処理231は、データ送受信処理201で送受信する共通交換形式データの構成とタイミングを決定するための処理で、プログラム開発情報241を元に、処理プログラムで用いられているデータの名前や構造といった情報を抽出し、共有データ情報テーブル242に格納する。

（実施例１）

図３は、本発明の第１の実施例における、共有データ情報テーブル242のテーブル構成例を示す図である。共有データ情報テーブル242の各レコードは、データ名311、データ型312、送信元処理装置・プログラム313、受信側装置・プログラム314、共有タイプ315、共通交換形式識別子316から構成される。データ名311は各共有データを識別するための名称であり、プログラムで用いられている変数名を用いることができる。データ型312は、各データの型を示し、文字列型や整数/実数型といった種別や、サイズなどの情報を示す。送信元処理装置・プログラム313は、共有データに対して書き込みを行うプログラム、及び該プログラムの格納される処理装置を示す。受信側処理装置・プログラム314は、該共有データを読み込むプログラム及び該プログラムの格納される処理装置を示す。共有タイプ315は、共有データに対する読み書き反映方法のタイプを示す。プログラムから随時更新される共有変数である「Flow」型や、処理の呼出し又は戻り値のように通知タイミングの明示される「Batch」型といった情報が格納される。共通交換形式識別子316は、他プログラムとデータ交換する場合の入出力単位毎の識別子である。

レコード321の共有データ例は、データ名「IOState」、データ型が構造体「IOState」、該データへの書込みは処理装置「114」のプログラム「IOControl」から行われ、読込みは不特定の複数処理装置・プログラムから行われる、すなわち読み書き型「Flow」型のデータの例が示されている。共通交換形式識別子316は「IOControl」で、他レコードに示されたデータ項目のうち同一の識別子「IOControl」を持つデータ項目と共に送受信される。
レコード322の共有データ例は、データ名「Mode」、データ型「BOOL」すなわちブール型、該データへの書込みは処理装置「114」のプログラム「IOControl」及び処理装置「111」の「StateSet」から行われ、読込みは処理装置「114」のプログラム「IOControl」から行われるデータの例が示されている。
図４は、図３にて示した共有データ情報を抽出する元となるプログラムソースコードの例であり、複数のプログラム間で共有されている変数の例を示している。プログラム331「IOControl」において、変数「IOState」の構成要素に、コード341、342にて書込みを行っている。また、プログラム332「IOMonitor」では、コード351にて変数「IOState」の構成要素を画面に表示する処理を行っている。このようなコードから、図５にて説明するステップで共有データ「IOState」を抽出することができる。グローバル変数定義333には、各処理プログラム間で共有されるグローバル変数の定義が記述され、コード361に共有データ「IOState」の定義が、コード362に共有データ「Mode」の定義が行われている。
図５は、図３にて示した共有データ情報を抽出する処理の流れを示す図である。本実施形態における共有データ情報の抽出は、プログラムを選択し（ステップ371）、グローバル変数を抽出する（ステップ372）。本処理で選択するプログラムは、システムにおける全プログラムを用いてもよいし、特定の一連の処理を行うプログラムグループのみを選択させてもよい。次に抽出したグローバル変数に対する入出力があるかどうか検索し（ステップ373）、出力がある場合は（ステップ374）、共有データ情報テーブルを検索し、同一のデータが定義されていない場合はデータ名及びデータ型の追加を行い（ステップ375）、該グローバル変数のレコードに処理プログラム及び該プログラムの実行される処理装置、共有タイプ「Flow」を設定する（ステップ376）。出力がない場合は、共有データ情報テーブルを検索し、同一のデータが定義されていない場合はデータ名及びデータ型の追加を行い、受信側装置・プログラムを設定する（ステップ377）。
ここで、ステップ375及びステップ377において、同一のデータかどうかの判定には、データ名を用いることも、またはデータに付記されたコメントなどの属性を用いることもできる。これらのステップの後、共通交換形式識別子を設定する（ステップ378）。共通交換形式識別子の設定は、共有データの組合せやプログラム間での交換タイミングに着目して行うことができる。共有データの組合せは、送信元装置毎や、送信元装置・プログラム名、選択したプログラムグループと送信元装置・プログラム名の組合せ、選択したプログラムグループと送信元装置・プログラム名と受信側装置・プログラム名の組合せなどである。特定のプログラムのみを選択させることで、共有させるグローバル変数のスコープを制限することが可能である。また共有データ１つ毎に着目してもよい。プログラム間での交換タイミングは、データ送受信処理201で用いる、周期、更新イベント、プログラム終了処理検出イベントなどである。
全てのプログラムをチェックした場合は終了し、未チェックプログラムが存在する場合はステップ373より再度処理を行う。
ここで示した共有データ情報の抽出は、プログラム開発処理221で共有変数の入出力管理まで行っている場合は、ステップ373を省略しても容易に抽出可能である。また共有変数がポインタ型である場合も、該ポインタの指す型をコンパイラなどにより抽出することで、共有するデータの型に関する情報を抽出することでデータ型の情報を得ることができる。
なお本処理で作成した共有データ情報テーブルの内容は、処理装置間で共有し、処理装置間で不一致が発生しないようにする。これには、例えば特願平８－249611に示された方法などを用いることができる。
図６は、第1の実施例におけるデータ送受信処理201の内、共有データ情報抽出処理231で

作成した共通交換形式データを周期的に送信する場合の流れを示す図である。データ送信処理は、送信周期イベントを検出する処理（ステップ401）と、送信データを検出する共有データ情報テーブル検索処理（ステップ402）、送信するデータの値を取得する処理（ステップ403）、及びデータ送信処理（ステップ404）から構成される。共有データ情報テーブル検索処理402は、周期イベント検出処理401から呼び出され、共有データ情報テーブル242を検索して送信すべきデータを抽出する。例えば、共有データ情報テーブル242のフィールド313「送信元処理装置・プログラム」を検索し、自処理装置内の処理プログラムが更新する共有データを抽出する。ここで抽出されたデータを用いて、共有データ値取得処理403は、処理プログラム実行環境202より該共有データの値を取得し、共通交換形式形成・送信404で送信データを形成して送信する。

図７は、図６にて説明した処理で用いる送信メッセージフォーマット例である。通信処理で用いる通信ヘッダ511、共通交換形式識別子512、及び共有データ513の列で構成される。共通交換形式識別子512は、例えば本例では送信周期と送信元処理装置などに着目した共有データ構成毎に、共有データ情報抽出処理231にて一意に指定する。共有データ513は、各データ名522、データ型523、データ値524、及びこれらのサイズ合計を示すサイズ521で構成される。

なお、本例ではデータの送信処理の流れを示したが、該データを受信した処理装置は、受信データの共通交換形式識別子を用いて共通交換形式を判断し、共有データ情報テーブル242の受信側装置・プログラムフィールド314を用いて自処理装置が受信側装置となっている共有データ項目を抽出し、受信データからこれを抽出した後処理プログラム実行環境202に渡して、処理装置内のプログラムが利用できるよう展開する。

このような機構を用いることで、複数の処理装置間、処理プログラム間で、共有データの構成やタイミングを意識することなくデータを共有することができる。

図８は、第1の実施例におけるデータ送受信処理201の内、共有データ情報抽出処理231で作成した共通交換形式データを更新時にFIFO形式で送信する場合の流れを示す図である。データ送信処理は、共有データ更新イベント検出処理（ステップ601）と、送信する共有データ値の取得、及びメッセージ形成・送信処理（ステップ411）から構成される。共有データ更新イベント検出処理601は、処理プログラムによる共有データ更新イベントを、更新した処理プログラム名と共に処理プログラム実行環境202から受け取り、ステップ411を呼び出す。または、周期的に共有データの値変化を監視することにより共有データ更新を検出してもよい。ステップ411においては、第１の実施例同様に更新された共有データの値を取得し、メッセージ形成・送信処理404でメッセージデータを形成して送信する。

送信するメッセージデータの構成例を、図９に示す。（a）図は、本実施例において、１つの共有データを１メッセージ化する場合のメッセージフォーマット例である。図７にて示した第１の実施例におけるメッセージフォーマットと同様の通信ヘッダ511、共通交換形式識別子711、共有データサイズ712、共有データ名713、共有データ型714、共有データ値715で構成される。共有データ715には、データ送受信処理201が呼び出されるトリガとなった更新データの値が格納される。（b）図は、更新された共有データを相乗りさせて送信する場合のメッセージフォーマット例である。図７にて示した第１の実施例におけるメッセージフォーマットと同様の通信ヘッダ511、共通交換形式データ721の列で構成される。共通交換形式データ721の構成は、（a）図にて示した通信ヘッダ以外の部分の構成と同様である。本フォーマットを用いる場合は、メッセージ形成・送信処理404において、更新された共有データのフォーマット721を作成し、蓄積する。メッセージサイズを越えた時点、または一定タイムアウト時間経過などのイベントにより、蓄積された部分をメッセージとして送信する。本例においても、該メッセージを受信した処理装置は、受信メッセージを解釈して共有データ項目毎に展開し、処理プログラム実行環境202に渡して、処理装置内のプログラムが利用できるよう展開する。

このような機構を用いることで、処理装置、処理プログラム間で、メッセージの構成やタイミングを意識することなく、更新されたデータのみを他処理装置やプログラムに通知・反映することができる。

図１０は、第1の実施例におけるデータ送受信処理201の内、共有データ情報抽出処理231で作成した共通交換形式データを、プログラムの実行状況により送信する場合の流れを示す図である。データ送信処理は、プログラム処理終了検出処理（ステップ801）と、送信する共有データの検索、値の取得、及びメッセージ形成・送信処理（ステップ411）から構成される。プログラム処理終了検出処理801は、処理プログラムの終了や、周期的に起動される処理プログラムの終了などのイベントを、処理プログラム名と共に処理プログラム実行環境202から受け取り、ステップ411を呼び出す。ステップ411においては、第１の実施例同様に更新された共有データの値を取得し、共通交換形式データ形成・送信処理404で送信データを形成して送信する。ここで、送信データとしては、例えばイベントの発生元プログラムが関係するデータを送信する。このとき、共有データ情報テーブル検索処理402において、共有データ情報テーブル242のフィールド313「送信元処理装置・プログラム」を検索し、処理終了した処理プログラムが送信元となっている共通交換形式データを抽出し、以降のステップ403、ステップ404でデータ生成・送信を行う。

（実施例２）

本発明の第２の実施例においては、分散オブジェクトやリモート・プロシージャ・コール（Remote Procedure Call）などのように、プログラムに明示的に他処理の呼出しが記述される場合の、共通交換形式データの作成と送信例について説明する。

図１１は、第２の実施例における共有データ情報テーブルの設定例を示す図である。プログラム「IOControl」からプログラム「IOServer」の処理「StateCheck」を呼び出す場合の引数となる変数「a」の情報がレコード901に、変数「ptr」の情報がレコード902にそれぞれ格納されている。また、処理「StateCheck」からの戻り値の情報がレコード903に格納されている。ここで、戻り値のデータ名は呼び出し処理名を用いて定義されている。

図１２は、図１１にて示した共有データ情報を抽出するプログラムソースコードの例であり、処理呼出しによるデータ共有の場合の例を示している。プログラム911「IOControl」において、コード922で他処理「StateCheck」を呼び出している。このとき渡すデータは「a」、「ptr」であるが、特に「ptr」についてはコード921においてデータ型が定義されている。また、プログラム912「IOServer」では、コード923にて処理「StateCheck」を記述しており、コード924で値を戻している。

図１３は、図１１にて示した共有データ情報を抽出する処理の流れを示す図である。本実施例における共有データ情報の抽出は、プログラムを選択し（ステップ951）、該プログラム内で他処理を呼び出している個所を抽出する（ステップ952）。次に該呼出しにおける引数や、引数がポインタ型である場合は該ポインタの指すデータ、及び呼出し処理からの戻り値を抽出し（ステップ953）、共有データ情報テーブルに引数、戻り値、及び呼び出し元または呼び出し先処理を格納する（ステップ954）。全てのプログラムをチェックした場合は終了し、未チェックプログラムが存在する場合はステップ951より再度処理を行う。共通交換形式データの作成は、第1の実施例同様に行うこともできるが、呼出ししている処理や呼び出されている処理、これらの組み合わせ、などに着目して設定することも可能である。

このような処理を用いて、図１２において示した例のプログラムより、共有データとなる「a」、「ptr」、及び「ptr」の指すデータ実体の構成要素を抽出することができる。

図１４は、第３の実施例におけるメッセージ送受信処理の流れを示す図である。本実施例におけるデータ送信処理は、処理プログラム実行環境202から処理の呼出しイベントを受け取り（ステップ1001）、共有データ情報テーブルを検索する（ステップ1002）。ステップ1002においては、呼び出される処理の引数となっている変数を検出し、さらに該処理に戻り値があるかどうかを検出する。ここで、戻り値は明示的に処理の戻り値が存在する場合に加え、本例で示したようにポインタ型変数を渡して、呼出し先処理においてデータ更新される場合も含む。その後引数となっている変数の値を取得し（ステップ1003）、共通交換形式データを形成して送信する（ステップ1004）。メッセージ送信後は、呼び出される処理に戻り値が存在したかを判断し（ステップ1005）、存在しなかった場合は終了する。存在した場合は、戻り値を受信するイベントを待つ（ステップ1006）。受信処理によっ

て戻り値を受信した場合、ステップ1006の待ちが解除されて、処理が終了する。
図１５は、本実施例におけるメッセージフォーマット例である。第１、第２の実施例同様に通信制御のための通信ヘッダ511を先頭に、共通交換形式識別子1101を設ける。（a）図は呼び出しメッセージフォーマットを示す。呼出しに用いる引数となる共有変数「a」、「ptr」の情報は、それぞれ共有データ部1102、1103に格納される。特に、ポインタ型である共有データ「ptr」については、受け取り側で該ポインタの指すデータ値そのものが必要となってくるため、「ptr」の指すデータ構造「struct IOHandle」の値一式を送信するものとし、データ名1112、データ型1113に続き、データ構造「struct IOHandle」の値一式を1114に格納し、これらのサイズ合計を1111に格納する。
（b）図は、応答メッセージフォーマットを示す。共通交換形式識別子1121と、応答データとして戻り値1122、及び呼出し先処理「StateCheck」にて更新された可能性のある共有データ「ptr」が1123に格納されて返送される。
このような機構を用いることで、処理プログラムの外部インタフェースが手続きの呼出しの形式で記述されている場合にも、プログラム開発情報を用いてプログラム間をバインドすることが可能である。
図１６は、本実施例において、オブジェクト指向に基づいて作成されたプログラム間をバインドする場合の共有データ情報抽出処理の流れを示す図である。本実施例における共有データ情報の抽出は、プログラムを選択し（ステップ1201）、該プログラム内で呼び出されているオブジェクトを抽出する（ステップ1202）。次に該オブジェクトに対して参照、または書き込みされているオブジェクト属性を抽出し（ステップ1203）、該オブジェクトの呼び出されているメソッドと、メソッド呼出しにおける引数や、引数がポインタ型である場合は該ポインタの指すデータ、及び呼出し処理からの戻り値を抽出する（ステップ1204）。ステップ1205においては、共有データ情報テーブルに、ステップ1203で抽出された属性を共有データとして呼び出し元/先処理と共に格納し、ステップ1204で抽出された引数、戻り値、及び呼び出し元/先処理を格納する。全てのプログラムをチェックした場合は終了し、未チェックプログラムが存在する場合はステップ1201より再度処理を行う。
このような処理により、オブジェクト指向プログラム間のバインドも、プログラム開発情報を用いて、プログラム開発時に呼び出し相手や呼出し元を意識しなくとも可能となる。また、既に開発したプログラム間を、プログラムを改造することなく連携させることが可能となる。
また、本実施例ではバインドするプログラムの送信側と受信側のインタフェースの対応に手続き名を用いたが、引数や戻り値名などを用いることで、片方のプログラムがグローバル変数の形式でデータを共有する場合でもプログラム間をバインドすることが可能である。ここで、送信側がグローバル変数の形式で、受信側が手続きの形式でデータ共有する場合は、受信側の必要とする手続きの引数が非同期に発生するが、例えば特願平９－75529号にあるような方法を用いることでデータを統合して受信側を呼び出すことが可能である。

（実施例３）
本発明の第３の実施例においては、本実施形態の第1の実施例及び第2の実施例で作成した共有データ情報を用いて、共通交換形式データを変更・設定するツールの例を説明する。
図１７は、本実施例における共通交換形式データを変更・設定するツールの例である。画面例1301には、一連の処理を行うプログラムグループである「実績管理」1311、該プログラムグループ「実績管理」に含まれる装置111の送信側プログラム「製造実績」1312、これらに着目して共有データ情報抽出処理231により作成された共通交換形式データ「実績管理・製造実績」1313、及び構成共有データ項目である「BCD」1314、「処理結果」1315、「作業者」1316が表示されている。また、プログラム「製造実績」とバインドされるプログラムとして、処理装置114の「実績監視」、及びバインドに使用される共有データ項目として「BCD」1322、「処理結果」1323が示されている。このような画面より、例えばリンク1331を解除することで、共有データ情報テーブル242の該当するレコードの送信元処理装置・プログラムまたは受信側処理装置・プログラムのフィールドを変更し、プログ

ラム間のバインドを解除することができる。
画面例1302は、リンク1371を新規作成することでプログラム間をバインドした例を示している。画面1302には、一連の処理を行うプログラムグループである「実績管理」1311、該プログラムグループ「実績管理」に含まれる装置111の送信側プログラム「設備状態」1341、共通交換形式データ「設備状態」1342、及び構成共有データ項目である「State」1343が表示されている。また、「設備状態」1342を介してバインドされた処理装置114のプログラム「状態監視」1351、及び共有データ項目である「状態」1361が示されている。このような画面を持つツールを用いることで、個別に作成されたプログラムで、かつ共有するデータ名やコメントなどの属性が異なる場合でも、プログラムを改造することなくバインドすることが可能となる。

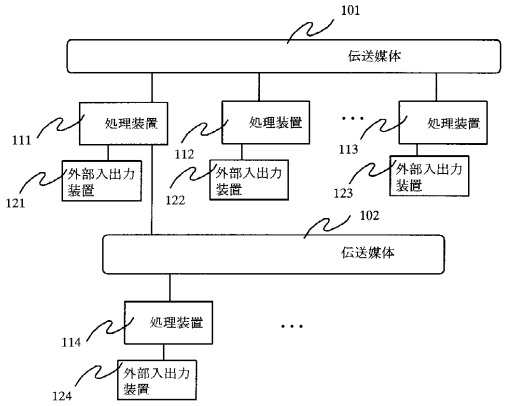
【発明の効果】
プログラム開発時に入力されるデータやプログラム実行に関する情報などのプログラム開発情報を利用することで、プログラム間で交換するデータを設計し、またプログラム実行時に設計した交換データを送受信することで、個別に開発されたプログラムをバインドすることが可能となる。このため開発者は内部処理作成に専念することができると共に、プログラム開発・実行後に連携させたくなったプログラム間も、プログラムを改造することなくバインドすることが可能となる。

【図面の簡単な説明】
【図１】実施形態の分散処理システムのハードウェア構成図である。
【図２】実施形態のプログラム間バインド方法のソフトウェア構成図である。
【図３】実施形態の第１の実施例における共有データ情報テーブル242のデータ構成を示す図である。
【図４】実施形態の第１の実施例におけるプログラム例と共有データ例を示す図である。
【図５】実施形態の第１の実施例における共通交換形式データ抽出処理の流れを示す図である。
【図６】実施形態の第１の実施例におけるデータ送信処理の流れの例を示す図である。
【図７】実施形態の第１の実施例におけるメッセージフォーマット例を示す図である。
【図８】実施形態の第１の実施例におけるデータ送信処理の流れの例を示す図である。
【図９】実施形態の第１の実施例におけるメッセージフォーマット例を示す図である。
【図１０】実施形態の第１の実施例におけるデータ送信処理の流れの例を示す図である。
【図１１】実施形態の第２の実施例における共有データ情報テーブル242の設定例を示す図である。
【図１２】実施形態の第２の実施例におけるプログラム例と共有データ例を示す図である。
【図１３】実施形態の第２の実施例における共有データ情報抽出処理の流れの例を示す図である。
【図１４】実施形態の第２の実施例におけるデータ送信処理の流れを示す図である。
【図１５】実施形態の第２の実施例におけるメッセージフォーマット例を示す図である。
【図１６】実施形態の第２の実施例における共有データ情報抽出処理の流れの例を示す図である。
【図１７】実施形態の第３の実施例における共通交換形式データを変更・設定するツールの例を示す図である。

【符号の説明】
111～114…処理装置、201…メッセージ送受信処理、202…処理プログラム実行環境、211…処理プログラム、221…プログラム開発処理、231…共有データ情報抽出処理、241…処理プログラム開発情報ファイル、242…共有データ情報テーブル。

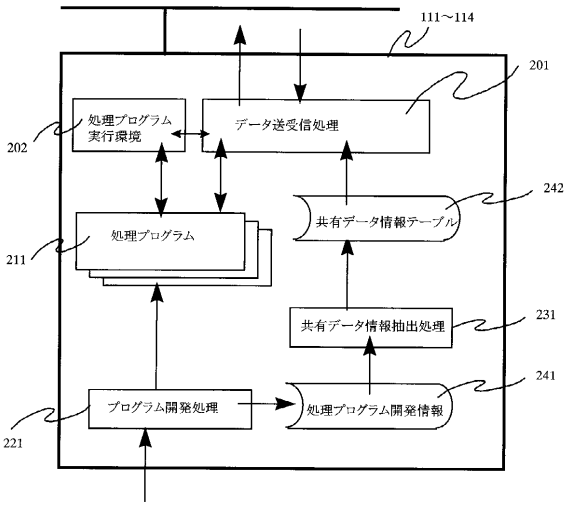
【図１】

図1



【図２】

図2



【図３】

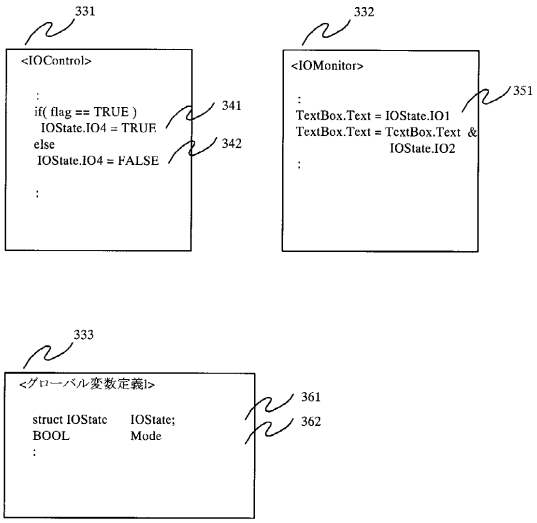
図3

311 データ名	312 データ型	313 送信元装置・プログラム	314 受信側装置・プログラム	315 共有タイプ	316 共通交換形式識別子
IOState	struct IOState	114・IOControl	*	Flow	IOControl
Mode	BOOL	114・IOControl 111・StateSet	114・IOControl	Flow	IOControl

321
322

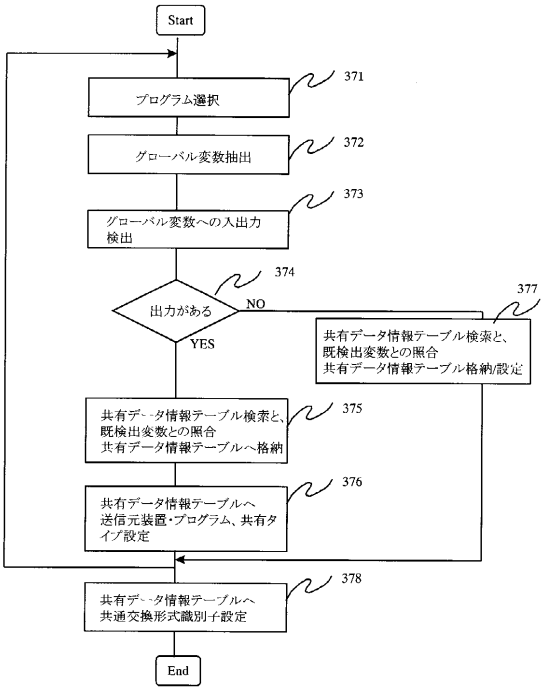
【図４】

図 4



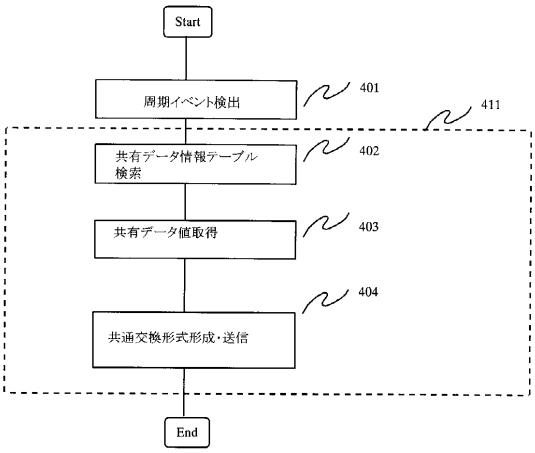
【図５】

図 5



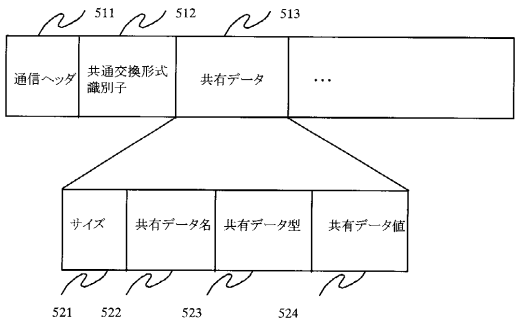
【図６】

図 6



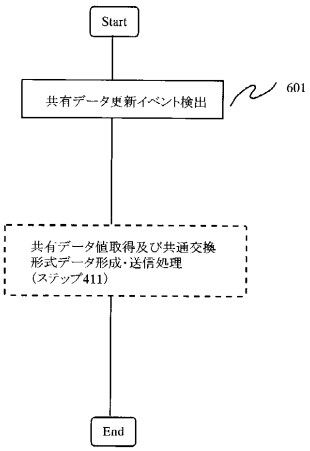
【図７】

図 7



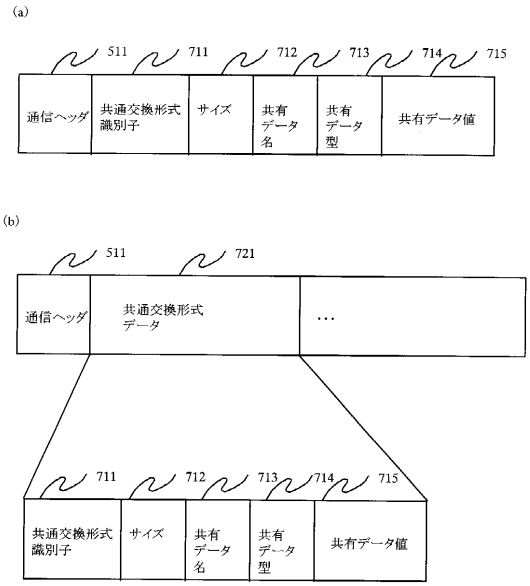
【図８】

図 8



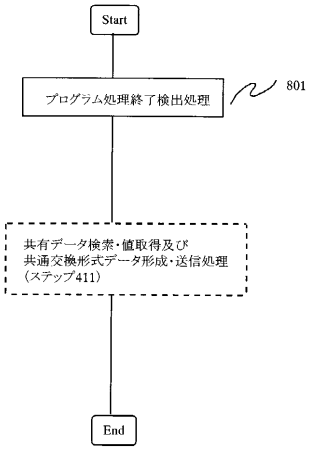
【図９】

図 9



【図１０】

図 10



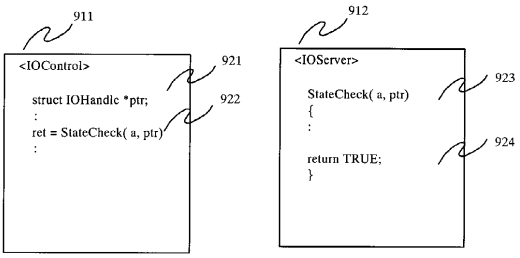
【図１１】

図11

311 データ名	312 データ型	313 送信元装置・プログラム	314 受信側装置・プログラム	315 共有タイプ	316 共通交換形式識別子	
a	integer16	114・IOControl:StateCheck	112・IOServer:StateCheck	Batch	IOControl・StateCheck	901
ptr	struct IOHandle *	114・IOControl:StateCheck	112・IOServer:StateCheck	Batch	IOControl・StateCheck	902
112・IOServer:StateCheck	BOOL	112・IOServer	114・IOControl	Batch	112・IOServer:StateCheck	903

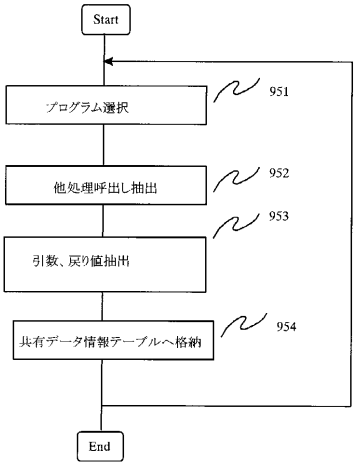
【図１２】

図 12



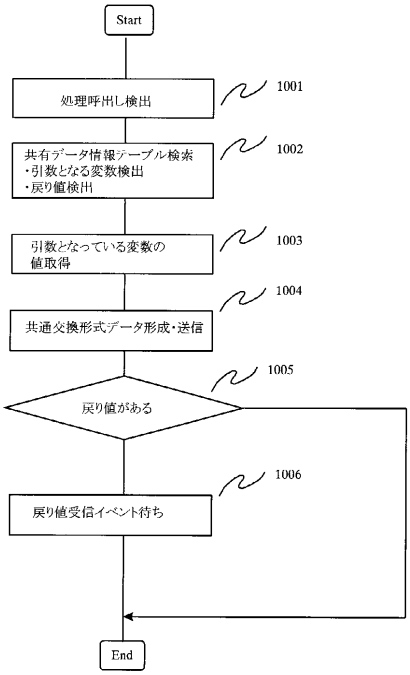
【図１３】

図 13



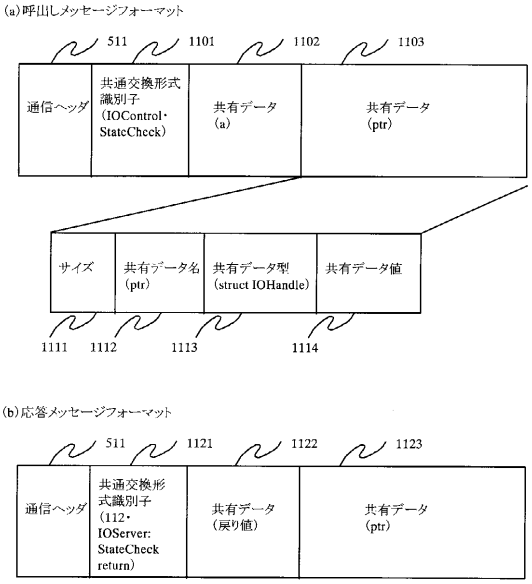
【図１４】

図 14



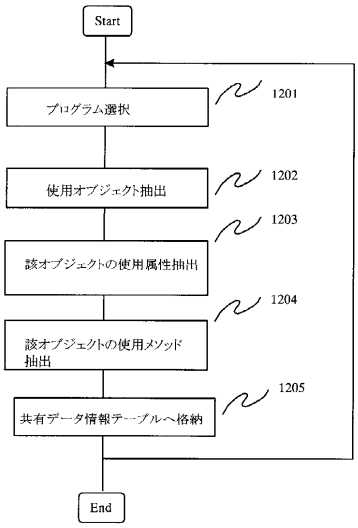
【図１５】

図 15



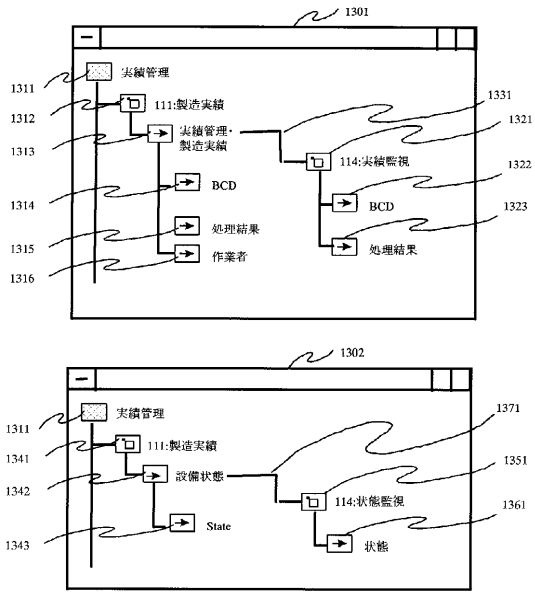
【図１６】

図 16



【図１７】

図 17


1301
実績管理
1311
111:製造実績
1312
実績管理・
製造実績
1313
BCD
1314
処理結果
1315
作業者
1316
1331
1321
114:実績監視
1322
BCD
1323
処理結果
1302
実績管理
1311
111:製造実績
1341
設備状態
1342
State
1343
1371
1351
114:状態監視
1361
状態

フロントページの続き

(72)発明者　西島　英児
神奈川県川崎市麻生区王禅寺１０９９番地　株式会社日立製作所　システム開発研究所内
(72)発明者　中村　智明
東京都千代田区神田駿河台四丁目６番地　株式会社日立製作所　新事業推進本部内

審査官　殿川　雅也

(56)参考文献　特開平０８－１２３６９９（ＪＰ，Ａ）
特開平０９－２４５００３（ＪＰ，Ａ）
特開昭６３－１０９５３９（ＪＰ，Ａ）
特開平０９－０８１３８３（ＪＰ，Ａ）

(58)調査した分野(Int.Cl.，ＤＢ名)
G06F 9/46 - 9/54