(19)日本国特許庁(JP)　(12)特　許　公　報(B2)　(11)特許番号
特許第6248523号
(P6248523)
(45)発行日　平成29年12月20日(2017.12.20)　(24)登録日　平成29年12月1日(2017.12.1)

(51)Int.Cl.　F I
G06F 9/50 (2006.01)　G06F 9/46 465Z

請求項の数 8 (全 34 頁)

(21)出願番号　特願2013-209824(P2013-209824)
(22)出願日　平成25年10月7日(2013.10.7)
(65)公開番号　特開2015-75803(P2015-75803A)
(43)公開日　平成27年4月20日(2015.4.20)
審査請求日　平成28年6月6日(2016.6.6)

(73)特許権者　000005223
富士通株式会社
神奈川県川崎市中原区上小田中４丁目１番１号
(74)代理人　100092152
弁理士　服部　毅巖
(72)発明者　今村　信貴
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

審査官　漆原　孝治

最終頁に続く

(54)【発明の名称】データ処理管理方法、情報処理装置およびデータ処理管理プログラム

(57)【特許請求の範囲】
【請求項１】
第１のコンピュータおよび第２のコンピュータを有するシステムのデータ処理管理方法において、
前記第１のコンピュータが、
一連のイベントを示すパターンであって記憶部に記憶された前記パターンに対応する処理を、前記第１のコンピュータにより受信した到着済イベントに応じて実行し、前記到着済イベントの情報を前記記憶部に記録し、
前記処理の割当てを前記第２のコンピュータに変更する指示を受信し、
前記処理に対する前記到着済イベントを示す進捗情報の送信準備中に、前記パターンに属する第１のイベントを受信し、
前記進捗情報および前記第１のイベントを含む送信データを生成し、
前記進捗情報に代えて、前記送信データを前記第２のコンピュータに送信する、
データ処理管理方法。
【請求項２】
前記第２のコンピュータが、前記送信データが到着する前に前記処理に対応付けられた第２のイベントを受信すると、前記第２のイベントを保持する、請求項１記載のデータ処理管理方法。
【請求項３】
前記第２のコンピュータが、前記送信データを受信した際に、前記第２のイベントより

も前に発生し前記処理に対応付けられているが前記第２のコンピュータに未到着である第３のイベントがあっても、前記進捗情報および前記第２のイベントを用いて前記処理を実行する、請求項２記載のデータ処理管理方法。
【請求項４】
前記第２のコンピュータが、前記送信データを受信すると、前記進捗情報および前記第１のイベントを用いて前記処理を実行し、その後、前記第２のイベントを用いて前記処理を実行する、請求項２または３記載のデータ処理管理方法。
【請求項５】
前記第１のコンピュータが、前記進捗情報と前記第１のイベントとを連続して送信する、請求項１乃至４の何れか１項に記載のデータ処理管理方法。
【請求項６】
前記第１のコンピュータが、前記指示を受信すると前記処理の実行を停止し、
前記第２のコンピュータが、前記進捗情報および前記第１のイベントを用いて前記処理を再開する、請求項１乃至５の何れか１項に記載のデータ処理管理方法。
【請求項７】
自装置で実行される処理に対応する一連のイベントのパターンを記憶する記憶部と、
前記パターンに対応する前記処理を、受信した到着済イベントに応じて実行し、前記到着済イベントの情報を前記記憶部に記録し、
前記処理の割当てを他の情報処理装置に変更する指示を受信し、
前記処理に対する前記到着済イベントを示す進捗情報の送信準備中に、前記パターンに属する第１のイベントを受信し、
前記進捗情報および前記第１のイベントを含む送信データを生成し、
前記進捗情報に代えて、前記送信データを前記他の情報処理装置に送信する、演算部と、
を有する情報処理装置。
【請求項８】
コンピュータに、
一連のイベントを示すパターンであって記憶部に記憶された前記パターンに対応する処理を、前記コンピュータにより受信した到着済イベントに応じて実行し、前記到着済イベントの情報を前記記憶部に記録し、
前記処理の割当てを他のコンピュータに変更する指示を受信し、
前記処理に対する前記到着済イベントを示す進捗情報の送信準備中に、前記パターンに属する第１のイベントを受信し、
前記進捗情報および前記第１のイベントを含む送信データを生成し、
前記進捗情報に代えて、前記送信データを前記他のコンピュータに送信する、
処理を実行させるデータ処理管理プログラム。

【発明の詳細な説明】
【技術分野】
【０００１】
本発明はデータ処理管理方法、情報処理装置およびデータ処理管理プログラムに関する。
【背景技術】
【０００２】
複数のマシンを含む分散処理システムを用いてデータ処理を行うことがある。マシンには、物理的なコンピュータ（物理マシンや物理ホストということがある）や物理マシン上で動作する仮想的なコンピュータ（仮想マシンや論理ホストということがある）が含まれ得る。例えば、分散処理システムを用いて複合イベント処理（ＣＥＰ：Complex Event Processing）を行うことがある。種々の装置により発行された複数のイベントを複数のマシンで並列に処理することで、複数のイベントを高速に処理し得る。

【０００３】
分散処理システムでは、マシン毎の処理の割当てを変更することでマシン毎の負荷の平準化を図ることがある。そこで、処理の割当て変更を行う種々の方法が考えられている。例えば、第１の計算機で実行中のプロセスを第２の計算機に移動させる際に、複製した当該プロセスを第２の計算機に送信しながら、第１の計算機で当該プロセスの実行を継続することで、プロセス移動中のプロセス停止時間を短く抑える提案がある。
【０００４】
また、自プロセッサで動作しているタスクを他のプロセッサに移動させる際、当該タスクを移動先に送り出す送り出しタスクを実行し、送り出す最中に移動対象タスクに対する割り込み要求が発生すると、送り出しタスクにより割り込み処理を起動する提案もある。
【０００５】
更に、処理負荷が所定値を超える仮想マシンが検出された場合、複合イベント処理の関連度に基づいて、仮想マシンそれぞれに複合イベント処理を分散させる提案もある。
【先行技術文献】
【特許文献】
【０００６】
【特許文献１】特開２００４－７８４６５号公報
【特許文献２】特開２０１０－２７２０７６号公報
【特許文献３】特開２０１２－７９２４２号公報
【発明の概要】
【発明が解決しようとする課題】
【０００７】
データ処理の割当てを変更する際に、変更前の処理経過を変更後も引き継ぎたいことがある。例えば、複数のイベントに対する処理を行う場合に、そのうち一部のイベントが発生済なら、一部のイベントの発生済の状態を割当て変更後も維持したいことがある。そこで、割当て変更前の第１のマシンから割当て変更後の第２のマシンに処理の進捗を提供し、第２のマシンに処理を途中から引き継がせることが考えられる。ところが、第１のマシンから第２のマシンへの進捗情報の送信中にも、割当て変更対象の処理に用いられるデータが第１のマシンに入力されることがある。このとき、当該データの扱いが問題となる。
【０００８】
例えば、上記提案のように、入力されたデータを第１のマシン側で継続して処理することが考えられる。しかし、当該データが第１のマシンに到着し続けると第１のマシンで処理を終えられなくなり、第２のマシンに当該処理を開始させるまでに時間がかかり得る。一方、第１のマシンで当該処理を中断し、進捗情報の送信完了後に、第１のマシンに入力されたデータを第２のマシンに送信し、処理を再開させることも考えられる。しかし、進捗情報の送信完了を待ち、更に当該データを次に送信できるタイミングまで待っていると、第２のマシンへの当該データの到着が遅れ、第２のマシンでの処理再開が遅延し得る。
【０００９】
１つの側面では、本発明は、割当て変更先での処理開始までの遅延を短縮できるデータ処理管理方法、情報処理装置およびデータ処理管理プログラムを提供することを目的とする。
【課題を解決するための手段】
【００１０】
１つの態様では、第１のコンピュータおよび第２のコンピュータを有するシステムのデータ処理管理方法が提供される。このデータ処理管理方法では、第１のコンピュータが、一連のイベントを示すパターンであって記憶部に記憶されたパターンに対応する処理を、第１のコンピュータにより受信した到着済イベントに応じて実行し、到着済イベントの情報を記憶部に記録し、処理の割当てを第２のコンピュータに変更する指示を受信し、処理に対する到着済イベントを示す進捗情報の送信準備中に、パターンに属する第１のイベントを受信し、進捗情報および第１のイベントを含む送信データを生成し、進捗情報に代え

て、送信データを第２のコンピュータに送信する。
【００１１】
また、１つの態様では、情報処理装置が提供される。この情報処理装置は、記憶部と演算部とを有する。記憶部は、自装置で実行される処理に対応する一連のイベントのパターンを記憶する。演算部は、パターンに対応する処理を、受信した到着済イベントに応じて実行し、到着済イベントの情報を記憶部に記録し、処理の割当てを他の情報処理装置に変更する指示を受信し、処理に対する到着済イベントを示す進捗情報の送信準備中に、パターンに属する第１のイベントを受信し、進捗情報および第１のイベントを含む送信データを生成し、進捗情報に代えて、送信データを他の情報処理装置に送信する。
【００１２】
また、１つの態様では、データ処理管理プログラムが提供される。このデータ処理管理プログラムは、コンピュータに、一連のイベントを示すパターンであって記憶部に記憶されたパターンに対応する処理を、コンピュータにより受信した到着済イベントに応じて実行し、到着済イベントの情報を記憶部に記録し、処理の割当てを他のコンピュータに変更する指示を受信し、処理に対する到着済イベントを示す進捗情報の送信準備中に、パターンに属する第１のイベントを受信し、進捗情報および第１のイベントを含む送信データを生成し、進捗情報に代えて、送信データを他のコンピュータに送信する、処理を実行させる。
【発明の効果】
【００１３】
１つの側面では、割当て変更先での処理開始までの遅延を短縮できる。
【図面の簡単な説明】
【００１４】
【図１】第１の実施の形態の分散処理システムを示す図である。
【図２】第２の実施の形態の分散処理システムを示す図である。
【図３】ノードのハードウェア例を示す図である。
【図４】クエリの割当て変更例を示す図である。
【図５】分散処理システムのソフトウェア例を示す図である。
【図６】分散処理システムのソフトウェア例（続き）を示す図である。
【図７】イベントの例を示す図である。
【図８】クエリの例を示す図である。
【図９】クエリ状態の例を示す図である。
【図１０】配置表の例（その１）を示す図である。
【図１１】配置表の例（その２）を示す図である。
【図１２】配置表の例（その３）を示す図である。
【図１３】送信データ管理構造体の例を示す図である。
【図１４】送信リストの例を示す図である。
【図１５】クエリ状態送信管理の例を示すフローチャートである。
【図１６】イベント受信管理（変更元）の例を示すフローチャートである。
【図１７】クエリ状態受信管理の例を示すフローチャートである。
【図１８】イベント受信管理（変更先）の例を示すフローチャートである。
【図１９】イベント送信管理の例を示すフローチャートである。
【図２０】クエリ状態の送信例（その１）を示す図である。
【図２１】クエリ状態の送信例（その２）を示す図である。
【図２２】クエリ割当て変更の例（その１）を示すシーケンス図である。
【図２３】クエリ割当て変更の比較例（その１）を示すシーケンス図である。
【図２４】クエリ割当て変更の例（その２）を示すシーケンス図である。
【図２５】クエリ割当て変更の比較例（その２）を示すシーケンス図である。
【図２６】分散処理システムの他の例を示す図である。
【発明を実施するための形態】

【００１５】
以下、本実施の形態を図面を参照して説明する。
［第１の実施の形態］
図１は、第１の実施の形態の分散処理システムを示す図である。第１の実施の形態の分散処理システムはマシン１，２を含む複数のマシンを有する。複数のマシンはネットワークに接続され、相互に通信可能である。この分散処理システムでは、複数の処理を複数のマシンにより分散して実行する。
【００１６】
ここで、第１の実施の形態ではマシン１，２として物理マシンを想定する。ただし、マシン１，２は仮想マシンでもよい。例えば、マシン１，２は異なる物理マシン上で動作する仮想マシンでもよいし、１台の物理マシン（記憶装置や演算処理を行う装置などを備えたコンピュータシステム）上で動作する仮想マシンでもよい。
【００１７】
マシン１は、記憶部１ａおよび演算部１ｂを有する。マシン２は、記憶部２ａおよび演算部２ｂを有する。記憶部１ａ，２ａは、ＲＡＭ（Random Access Memory）などの揮発性記憶装置でもよいし、ＨＤＤ（Hard Disk Drive）やフラッシュメモリなどの不揮発性記憶装置でもよい。演算部１ｂ，２ｂは、ＣＰＵ（Central Processing Unit）、ＤＳＰ（Digital Signal Processor）、ＡＳＩＣ（Application Specific Integrated Circuit）、ＦＰＧＡ（Field Programmable Gate Array）などを含み得る。演算部１ｂ，２ｂはプログラムを実行するプロセッサであってもよい。ここでいう「プロセッサ」には、複数のプロセッサの集合（マルチプロセッサ）も含まれ得る。
【００１８】
記憶部１ａは、マシン１に割当てられた処理毎の進捗情報を記憶する。進捗情報３は、そのうちのある処理の進捗情報である。例えば、第１の実施の形態の分散処理システムが、複数の入力データに対して所定の処理を実行するシステムなら、記憶部１ａは、複数の入力データのうち一部が到着済であることを進捗情報３として記憶する。一例として、ＣＥＰを実行するシステムが考えられる。その場合、記憶部１ａは、一連のイベントのうちの一部のイベントが到着済であることを進捗情報３として記憶することが考えられる。
【００１９】
演算部１ｂは、マシン１に割当てられた複数の処理を実行する。演算部１ｂは、各処理の実行状況を進捗情報として記録し、記憶部１ａに格納する。例えば、演算部１ｂは、複数の入力データに対して当該処理を実行する。例えば、演算部１ｂは、複数の入力データの何れかを受信するたびに、当該入力データが到着済である（当該入力データを処理済である）ことを進捗情報３に記録してもよい。
【００２０】
演算部１ｂは、マシン１で実行される処理の割当てをマシン２に変更する指示（変更指示４）を受信すると、当該指示で指定された処理に対応する進捗情報３をマシン２に送信する。例えば、演算部１ｂは、処理の割当てを管理する所定の装置から変更指示４を受け付けてもよい。あるいは、演算部１ｂは、マシン１に接続された所定の入力デバイスを用いたユーザの操作入力により、変更指示４を受け付けてもよい。
【００２１】
ここで、各処理を実行するためのプログラムは、例えば記憶部１ａ，２ａに予め格納される。例えば、演算部１ｂ，２ｂは、ある処理が自身に割当てられた際に、ＨＤＤなどに格納された当該処理に対応するプログラムをＲＡＭに格納し、実行することで、当該処理をデータ入力待ちの状態とすることができる。
【００２２】
演算部１ｂは、割当て変更対象の処理の進捗情報３をマシン２に送信中に、当該処理に用いられるデータ５を受信すると、進捗情報３にデータ５を追加してマシン２に送信する。情報の送信中の期間は、ネットワーク上へ当該情報を送出するための準備（送信準備）期間を含む。例えば、準備期間はシリアライズやバッファリングなどに要する期間を含む

。シリアライズは、進捗情報３をネットワークの転送形式に変換する処理である。バッファリングは、送信する情報を所定量まで蓄積する処理である。マシン１からマシン２へ進捗情報３を送信中の間、進捗情報３に対応する処理の実行は中断される。

【００２３】

例えば、演算部１ｂは、変更指示４を受信すると、記憶部１ａに記憶された進捗情報３のシリアライズやバッファリングを行い、進捗情報３を含む送信データを生成する。演算部１ｂは、当該送信データの生成中にデータ５を受信すると、当該送信データにデータ５を含めた送信データ６を生成する。送信データ６には、進捗情報３およびデータ５以外の情報が含まれてもよい。演算部１ｂは、送信データ６をマシン２に送信する。

【００２４】

マシン２は、送信データ６を受信すると、送信データ６から進捗情報３およびデータ５を取得する。マシン２は、進捗情報３およびデータ５を用いて、マシン１からマシン２に割当て変更された処理を再開する。

【００２５】

第１の実施の形態の分散処理システムによれば、マシン１により、マシン１で実行される処理の割当てをマシン２に変更する変更指示４が受信される（ステップＳ１）。マシン１により、当該処理の進捗情報３のマシン２への送信中に、当該処理に用いられるデータ５が受信される（ステップＳ２）。すると、マシン１により、進捗情報３にデータ５が追加されてマシン２に送信される（ステップＳ３）。

【００２６】

これにより、割当て変更先（上記の例ではマシン２）での処理開始までの遅延を短縮できる。ここで、例えば、進捗情報３の送信中、割当て変更対象の処理をマシン１で継続することも考えられる。しかし、当該処理に対してデータ５を含む複数のデータがマシン１に入力され続けることがある。すると、マシン１での処理を終えられなくなり、マシン２への当該処理の割当て変更が完了するまでに時間がかかる。また、マシン１が高負荷であるために処理の割当て変更を行いたいにも関わらず、マシン１が高負荷である状況が継続してしまう。しかも、マシン１の負荷が高い程、このような状況が発生する可能性は高い。更に、マシン１で処理が行われると、進捗情報３にも更新による差分が生じる。このため、マシン１は、マシン２への進捗情報３の差分の提供も継続することになり、ネットワーク帯域を消費してしまうという問題もある。

【００２７】

一方、マシン１による割当て変更対象の処理を中断し、進捗情報３の送信完了後にデータ５をマシン２に送信して、マシン２により当該処理を再開させることも考えられる。しかし、情報の送信にはシリアライズやバッファリングなどの送信側での処理や、デシリアライズなどの受信側での処理を要し、時間がかかる。このため、進捗情報３の送信完了を待ち、更にデータ５を次に送信できるタイミングまで待っていると、マシン２へのデータ５の到着が遅れ、マシン２での処理再開が遅延し得る。

【００２８】

そこで、マシン１は、割当て変更対象の処理の進捗情報３のマシン２への送信中に、当該処理に用いられるデータ５を受信すると、進捗情報３にデータ５を追加してマシン２に送信する。例えば、マシン１へデータ５を含む複数のデータが継続的にマシン１に入力されても、進捗情報３とともに各データ（あるいは、そのうちの一部）をマシン２へ提供し得る。マシン２は、進捗情報３とともにデータ５を取得できるので、進捗情報３およびデータ５を用いて当該処理を迅速に開始できる。また、マシン１で当該処理を継続せずに済むので、進捗情報３の提供を継続して行わなくてもよい。よって、マシン１で処理を継続する場合よりも、マシン１の負荷を迅速に低減でき、また、ネットワーク帯域の利用量も低減できる。

【００２９】

更に、進捗情報３とともにデータ５をマシン２に提供できるので、進捗情報３の送信完了を待機し、更に、次の送信タイミングまでデータ５の送信を待機しなくてよい。このた

め、マシン２へデータ５が到着するまでの時間を短縮化でき、マシン２で割当て変更対象の処理を再開するまでの遅延を短縮できる。

【００３０】

［第２の実施の形態］

図２は、第２の実施の形態の分散処理システムを示す図である。第２の実施の形態の分散処理システムは、ノード１００，２００，３００および管理ノード４００を含む。ノード１００，２００，３００および管理ノード４００は、ネットワーク１０に接続されている。ネットワーク１０は、例えばＬＡＮ（Local Area Network）である。

【００３１】

ネットワーク１０は、ネットワーク２０に接続されている。ネットワーク２０は、ＷＡＮ（Wide Area Network）やインターネットなどの広域ネットワークでもよい。ネットワーク２０には、スマートシティ２１、物流センサ２２、気象衛星２３、携帯装置２４およびスマートセンサ２５が無線または有線で接続されている。ネットワーク２０には、これらの装置以外にもイベントを発行する各種の装置が接続され得る。第２の実施の形態の分散処理システムは、ノード１００，２００，３００を用いてＣＥＰを実行する。

【００３２】

ノード１００，２００，３００は、イベントを処理するサーバコンピュータである。ノード１００，２００，３００は、スマートシティ２１、物流センサ２２、気象衛星２３、携帯装置２４およびスマートセンサ２５などにより発行された種々のイベントを並列に処理する。ノード１００，２００，３００は、複数のイベントの処理結果（新たなイベント）に対する所定の処理を行うこともある。

【００３３】

ノード１００，２００，３００は、ＣＥＰにより次のような機能を実現することが考えられる。例えば、ネットワーク２０に接続されたスマートシティ２１やスマートセンサ２５から取得された消費電力の情報を基にして、スマートシティ２１の省電力化を制御する。また、ネットワーク２０に接続された各種装置から取得した交通状況の情報を基にして、携帯装置２４を所持するユーザや車などの状況に合わせた適切なナビゲーションを携帯装置２４に提供する。また、気象衛星２３やレーダーなどから取得されたイベントを基にして、天気予報を携帯装置２４に提供する。更に、家屋への侵入有無の報告、家族（子供やお年寄り）の所在確認の報告などを行う。ノード１００，２００，３００は、その他にも種々の情報をユーザに提供し得る。

【００３４】

ここで、ノード１００，２００，３００により、イベントに応じて実行される処理を以下ではクエリと称する。クエリの内容を記述したプログラムは、ノード１００，２００，３００に予め与えられる。当該プログラムを、イベントに応じた処理を記述したルールまたはルール情報ということもできる。また、以下の説明では、ノード１００，２００，３００それぞれを指す場合に、各ノードということがある。

【００３５】

管理ノード４００は、各クエリによるイベント処理を、何れのノードに担当させるかの割当て（クエリの割当て）を行うサーバコンピュータである。例えば、管理ノード４００は、ノード１００，２００，３００の負荷を分散する。管理ノード４００は、ノード１００，２００，３００の負荷に応じてクエリの割当て変更を行うことで、ノード１００，２００，３００の負荷の平準化を図る。

【００３６】

図３は、ノードのハードウェア例を示す図である。ノード１００は、プロセッサ１０１、ＲＡＭ１０２、ＨＤＤ１０３、通信部１０４、画像信号処理部１０５、入力信号処理部１０６、ディスクドライブ１０７および機器接続部１０８を有する。各ユニットがノード１００のバスに接続されている。ノード２００，３００および管理ノード４００もノード１００と同様のユニットを用いて実現できる。

【００３７】

プロセッサ１０１は、ノード１００の情報処理を制御する。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、ＤＳＰ、ＡＳＩＣまたはＦＰＧＡなどである。プロセッサ１０１は、ＣＰＵ、ＤＳＰ、ＡＳＩＣ、ＦＰＧＡなどのうちの２以上の要素の組み合わせであってもよい。

【００３８】

ＲＡＭ１０２は、ノード１００の主記憶装置である。ＲＡＭ１０２は、プロセッサ１０１に実行させるＯＳ（Operating System）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を一時的に記憶する。また、ＲＡＭ１０２は、プロセッサ１０１による処理に用いる各種データを記憶する。

【００３９】

ＨＤＤ１０３は、ノード１００の補助記憶装置である。ＨＤＤ１０３は、内蔵した磁気ディスクに対して、磁気的にデータの書き込みおよび読み出しを行う。ＨＤＤ１０３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。ノード１００は、フラッシュメモリやＳＳＤ（Solid State Drive）などの他の種類の補助記憶装置を備えてもよく、複数の補助記憶装置を備えてもよい。

【００４０】

通信部１０４は、ネットワーク１０を介して他のコンピュータと通信を行えるインタフェースである。通信部１０４は、有線インタフェースでもよいし、無線インタフェースでもよい。

【００４１】

画像信号処理部１０５は、プロセッサ１０１からの命令に従って、ノード１００に接続されたディスプレイ１１に画像を出力する。ディスプレイ１１としては、ＣＲＴ（Cathode Ray Tube）ディスプレイや液晶ディスプレイなどを用いることができる。

【００４２】

入力信号処理部１０６は、ノード１００に接続された入力デバイス１２から入力信号を取得し、プロセッサ１０１に出力する。入力デバイス１２としては、例えば、マウスやタッチパネルなどのポインティングデバイス、キーボードなどを用いることができる。

【００４３】

ディスクドライブ１０７は、レーザ光などを利用して、光ディスク１３に記録されたプログラムやデータを読み取る駆動装置である。光ディスク１３として、例えば、ＤＶＤ（Digital Versatile Disc）、ＤＶＤ－ＲＡＭ、ＣＤ－ＲＯＭ（Compact Disc Read Only Memory）、ＣＤ－Ｒ（Recordable）／ＲＷ（ReWritable）などを使用できる。ディスクドライブ１０７は、例えば、プロセッサ１０１からの命令に従って、光ディスク１３から読み取ったプログラムやデータをＲＡＭ１０２またはＨＤＤ１０３に格納する。

【００４４】

機器接続部１０８は、ノード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、機器接続部１０８にはメモリ装置１４やリーダライタ装置１５を接続できる。メモリ装置１４は、機器接続部１０８との通信機能を搭載した記録媒体である。リーダライタ装置１５は、メモリカード１６へのデータの書き込み、またはメモリカード１６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１６は、カード型の記録媒体である。機器接続部１０８は、例えば、プロセッサ１０１からの命令に従って、メモリ装置１４またはメモリカード１６から読み取ったプログラムやデータをＲＡＭ１０２またはＨＤＤ１０３に格納する。

【００４５】

図４は、クエリの割当て変更例を示す図である。ノード１００は、ＣＥＰエンジンＥ１を有する。ノード２００は、ＣＥＰエンジンＥ２を有する。ノード３００は、ＣＥＰエンジンＥ３を有する。ＣＥＰエンジンＥ１，Ｅ２，Ｅ３はＣＥＰを実行する。例えば、ＣＥＰエンジンＥ１，Ｅ２，Ｅ３は、各ノードが備えるＲＡＭに記憶されたプログラムを各ノードが備えるプロセッサが実行することで実現される。ＣＥＰエンジンＥ１，Ｅ２，Ｅ３は、各ノードが備える専用のハードウェアで実現されてもよい。

【００４６】
例えば、ＣＥＰエンジンＥ１には、ノード１００が担当するクエリに応じたイベントが入力される。ＣＥＰエンジンＥ１は当該クエリの結果として新たなイベントを生成し、出力する。出力されたイベントは、他ノードに送信されたり、ネットワーク２０に接続された各種の装置に送信されたりする。これにより、ＣＥＰエンジンＥ１は、他ノードに別のイベント処理を行わせたり、ネットワーク２０に接続された装置を制御したりする。ＣＥＰエンジンＥ２，Ｅ３もＣＥＰエンジンＥ１と同様である。
【００４７】
ここで、管理ノード４００は、ノード１００，２００，３００の負荷を監視する。例えば、ノード２００の負荷がノード１００よりも高く、ノード３００の負荷がノード１００よりも低い場合が考えられる。この場合、管理ノード４００は、ノード２００に割当てられた何れかのクエリをノード３００に割当てると決定する。ノード２００の負荷を軽減でき、かつ、ノード１００，２００，３００の負荷の平準化を図れるからである。管理ノード４００は、ノード３００の負荷が比較的高まり、ノード２００の負荷が比較的低くなったのであれば、ノード３００に割当てられた何れかのクエリをノード２００に割当てると決定することもできる。
【００４８】
管理ノード４００は、決定した割当て変更をノード１００，２００，３００に指示する。ノード１００，２００，３００は当該指示に応じて、クエリの割当て変更を行う。例えば、あるクエリの割当てがノード２００からノード３００へ変更になった場合、ノード１００，２００，３００は、当該クエリの担当ノードをノード３００に更新する。このように、第２の実施の形態では、スケーラブルなシステムを実現し得る。例えば、ノードの追加やノードの削減に対して、クエリの割当てを柔軟化できる。
【００４９】
ここで、クエリは、複数のイベントの到着状況に応じた状態（クエリ状態）をもつ。第２の実施の形態では、クエリの割当て変更前後でクエリ状態を維持する。具体的には、割当て変更元のノードは、割当て変更先のノードへ、割当て変更対象のクエリのクエリ状態を提供することで、割当て変更先のノードへクエリの実行を引き継ぐ。ここで、クエリ状態は、第１の実施の形態の進捗情報の一例である。
【００５０】
図５は、分散処理システムのソフトウェア例を示す図である。ノード１００は、クエリ記憶部１１０、管理情報記憶部１２０、クエリ実行管理部１３０、クエリ状態送信管理部１４０、クエリ状態受信管理部１５０、イベント送信管理部１６０、イベント受信管理部１７０および通信部１８０を有する。クエリ記憶部１１０および管理情報記憶部１２０は、ＲＡＭ１０２やＨＤＤ１０３に確保した記憶領域として実現できる。
【００５１】
クエリ実行管理部１３０、クエリ状態送信管理部１４０、クエリ状態受信管理部１５０、イベント送信管理部１６０、イベント受信管理部１７０および通信部１８０は、プロセッサ１０１が実行するソフトウェアのモジュールとして実現できる。また、各部はＣＥＰエンジンＥ１の機能の一部でもよい（ノード２００，３００も同様）。
【００５２】
クエリ記憶部１１０は、クエリおよびクエリ状態を記憶する。クエリ記憶部１１０は、分散処理システムで用いられる全てのクエリを予め記憶する。また、クエリ記憶部１１０は、クエリ毎の現在のクエリ状態を記憶する。クエリ記憶部１１０は、イベントに含まれるストリーム名とそのイベントを処理するクエリの識別情報（クエリ名）との対応関係の情報を予め記憶する。
【００５３】
管理情報記憶部１２０は、配置表および送信リストを記憶する。配置表は、各ノードへのクエリの割当て状況を示す。また、配置表は、クエリと当該クエリの現在のクエリ状態との対応を示す。送信リストは、送信対象のデータを格納したコンテナである。送信リス

トは、宛先毎に用意される。
【００５４】
クエリ実行管理部１３０は、入力されたイベントに応じたクエリの実行を管理する。クエリ実行管理部１３０は、クエリの実行に伴って、クエリ記憶部１１０に記憶された当該クエリのクエリ状態を更新する。
【００５５】
クエリ状態送信管理部１４０は、自ノード（ここでは、ノード１００）に割当てられたクエリの他ノードへの割当て変更指示を管理ノード４００から受信する。すると、クエリ状態送信管理部１４０は、クエリ記憶部１１０から当該クエリのクエリ状態を取得する。クエリ状態送信管理部１４０は、取得したクエリ状態をシリアライズし、宛先ノードへの送信リストに追加する。シリアライズとは、ネットワーク１０に送出可能なデータ形式へ情報を変換する処理である。クエリ状態送信管理部１４０は、送信リストのデータサイズが所定サイズに達するまでバッファリングし、当該送信リストに含まれるデータを宛先ノードへ送信する。
【００５６】
クエリ状態受信管理部１５０は、他ノードから自ノードへ割当て変更されたクエリのクエリ状態を、当該他ノードから受信する。クエリ状態受信管理部１５０は、割当て変更されたクエリとクエリ状態との対応を管理情報記憶部１２０に記憶された配置表に登録する。後述するように、クエリ状態受信管理部１５０が他ノードから受信するクエリ状態には、当該クエリに対するイベントが付加されていることもある。その場合、クエリ状態受信管理部１５０は当該イベントを用いたクエリ実行をクエリ実行管理部１３０に依頼する。
【００５７】
イベント送信管理部１６０は、イベントの送信管理を行う。具体的には、イベント送信管理部１６０は、他ノードが担当するクエリのイベントを当該他ノードへ送信するため、当該イベントのシリアライズや送信リストへの登録を行う。イベント送信管理部１６０は、送信リストに含まれるイベントを当該他ノードへ送信する。
【００５８】
イベント受信管理部１７０は、イベントの受信管理を行う。具体的には、イベント受信管理部１７０は、次に示す（１）～（４）の各場合に応じた処理を実行する。
（１）取得したイベントが自ノードに割当てられたクエリに対応するものである場合。この場合、イベント受信管理部１７０は、当該イベントを用いたクエリ実行をクエリ実行管理部１３０に依頼する。
【００５９】
（２）取得したイベントが自ノードから他ノードへの割当て変更対象のクエリに対応するものであり、当該クエリのクエリ状態が自ノードから他ノードへ送信中として管理されている場合。この場合、イベント受信管理部１７０は、当該クエリ状態の送信リストに当該イベントを追加する。
【００６０】
（３）取得したイベントが他ノードから自ノードへの割当て変更対象のクエリに対応するものであり、当該クエリのクエリ状態が他ノードから自ノードへ送信中として管理されている場合。この場合、イベント受信管理部１７０は、当該クエリに対する待ちイベントとして、管理情報記憶部１２０に記憶された配置表に当該イベントを登録する。
【００６１】
（４）取得したイベントが他ノードに割当てられたクエリに対応するものである場合。この場合、イベント受信管理部１７０は当該他ノードに当該イベントを転送する。
イベント受信管理部１７０は、管理情報記憶部１２０に記憶された配置表に基づいて上記（１）～（４）の場合を判断できる。なお、上記（２）、（３）において「クエリ状態が送信中である」と管理される期間には、クエリ状態をネットワーク上へ送出するための準備（送信準備）期間（シリアライズやバッファリングなどを行う期間）を含む。
【００６２】

通信部１８０は、ノード２００，３００、管理ノード４００およびネットワーク２０に接続された各種の装置との間の通信を行う。上述したクエリ状態送信管理部１４０、クエリ状態受信管理部１５０、イベント送信管理部１６０およびイベント受信管理部１７０と他の装置との間のデータの送受信は、通信部１８０を介して行われる。

【００６３】

管理ノード４００は、管理情報記憶部４１０および配置制御部４２０を有する。管理情報記憶部４１０は、管理ノード４００が備えるＲＡＭやＨＤＤなどに確保した記憶領域として実現できる。配置制御部４２０は、管理ノード４００が備えるプロセッサが実行するソフトウェアのモジュールとして実現できる。

【００６４】

管理情報記憶部４１０は、配置表を記憶する。配置制御部４２０は、ノード１００，２００，３００の負荷を監視する。配置制御部４２０は、ノード１００，２００，３００の負荷に応じて、各ノードに対するクエリの割当てを変更することで、各ノードの負荷の平準化を図る。配置制御部４２０は、クエリの割当て変更に伴って、管理情報記憶部４１０に記憶された配置表の更新も行う。

【００６５】

配置制御部４２０は、クエリの割当てを変更する際、ノード１００，２００，３００の全てに、その旨を指示する。この指示は、割当て変更対象のクエリ、変更元のノードおよび変更先のノードを示す識別情報を含む。また、この指示は、各ノードが保持する配置表の更新指示も含む。クエリの割当て変更の指示は、ユーザにより、管理ノード４００に接続された所定の入力デバイスを用いて入力されてもよい。その場合、管理ノード４００は、ユーザによる操作入力を契機としてクエリの割当て変更を各ノードに指示する。

【００６６】

図６は、分散処理システムのソフトウェア例（続き）を示す図である。ノード２００は、クエリ記憶部２１０、管理情報記憶部２２０、クエリ実行管理部２３０、クエリ状態送信管理部２４０、クエリ状態受信管理部２５０、イベント送信管理部２６０、イベント受信管理部２７０および通信部２８０を有する。

【００６７】

ノード３００は、クエリ記憶部３１０、管理情報記憶部３２０、クエリ実行管理部３３０、クエリ状態送信管理部３４０、クエリ状態受信管理部３５０、イベント送信管理部３６０、イベント受信管理部３７０および通信部３８０を有する。

【００６８】

クエリ記憶部２１０，３１０および管理情報記憶部２２０，３２０は、ノード２００，３００が備えるＲＡＭやＨＤＤなどに確保した記憶領域として実現できる。クエリ実行管理部２３０，３３０、クエリ状態送信管理部２４０，３４０、クエリ状態受信管理部２５０，３５０、イベント送信管理部２６０，３６０、イベント受信管理部２７０，３７０および通信部２８０，３８０は、ノード２００，３００が備えるプロセッサが実行するソフトウェアのモジュールとして実現できる。ここで、ノード２００，３００が備える機能は、ノード１００が備える同名の機能と同様であるため、説明を省略する。

【００６９】

図７は、イベントの例を示す図である。イベントＸは、イベントのフォーマットを例示している。イベントＸは、イベント種別、ストリーム名および内容の項目を含む。イベント種別の項目にはイベントの種別が登録される。ストリーム名の項目には、当該イベントに対応するストリームの識別情報が登録される。内容の項目には、イベントの内容を示す情報が登録される。

【００７０】

イベントＸ１は、イベントＸの一例である。例えば、イベントＸ１には、イベント種別が“Ｐ”、ストリーム名が“ＩｎｐｕｔＳｔｒｅａｍ”、内容が“１０００Ｗ”という情報が格納されている。これは、当該イベント種別が“Ｐ”であり、電力に関する情報であることを示す。また、イベントＸ１に対応するストリーム名が“ＩｎｐｕｔＳｔｒｅａｍ

”であることを示す。また、イベントＸ１の内容が“１０００Ｗ”の消費電力の検出であることを示す。

【００７１】

図８は、クエリの例を示す図である。クエリ１１１は、クエリ記憶部１１０に格納される。クエリ１１１は、Ｅｓｐｅｒと呼ばれるＥＰＬ（Event Processing Language）を想定した記述例である。クエリ１１１では、“Ａ”、“Ｂ”、“Ｃ”という３種のストリームの順番でイベントが入力された場合に、データ（イベント）をデータストリームに出力する。各クエリでは、このように種々のイベントに対する条件を定めることができる。ここで、以下の説明では、ストリーム名“Ａ”であるイベントを、イベント“Ａ”、イベントＡのように表記することがある。

【００７２】

図９は、クエリ状態の例を示す図である。クエリ状態１１２は、クエリ１１１のクエリ状態の一例である。クエリ状態１１２は、クエリ記憶部１１０に格納される。クエリ状態１１２は、クエリ１１１に対して、イベントＡ，Ｂが到着済だが、イベントＣが未到着である（イベントＣの到着待ちの状態である）ことを示している。ノード１００，２００，３００は、クエリ毎の現在のクエリ状態を保持する。クエリとクエリ状態との対応は、配置表により管理される。

【００７３】

図１０は、配置表の例（その１）を示す図である。配置表１２１は、管理情報記憶部１２０に格納される。図１０の例では、テーブル形式で配置表１２１を表したが任意のデータ構造を利用できる。例えば、配置表１２１は、クエリ名をｋｅｙとしたＨａｓｈＭａｐなどでもよい。配置表１２１は、クエリ名、配置ノード名、状況、クエリ状態への参照、ロックおよび待ちイベントの項目を含む。

【００７４】

クエリ名の項目には、クエリの識別情報が登録される。配置ノード名の項目には、当該クエリを割当てられた（配置された）ノード（担当ノード）の名称が登録される。状況の項目には、当該クエリの現在の状況が登録される。

【００７５】

クエリの状況は、次のような状態を含む。（１）稼働中の状態（担当ノードで当該クエリを実行可能である状態）。（２）他ノードに移動中の状態（クエリの割当て変更に伴い、変更元のノードから変更先の担当ノードへクエリ状態を送信中である状態）。（３）他ノードに移動済の状態（変更元のノードから変更先の担当ノードへクエリ状態の送信を終えた状態）。

【００７６】

クエリ状態への参照の項目には、クエリ状態へのポインタが登録される。ただし、自ノードで当該クエリ状態を管理していない場合には、クエリ状態への参照の項目には“ノード内に無し”という情報が登録される。

【００７７】

ロックの項目には、クエリ状態のロックの有無が登録される。待ちイベントの項目には、待ちイベントが登録される。待ちイベントは、割当て変更前の担当ノードから自ノードへ、割当て変更対象のクエリのクエリ状態の送信が完了する前に、当該クエリに対して自ノードが取得したイベントである。

【００７８】

例えば、配置表１２１には、クエリ名が“Ｑｕｅｒｙ１”、配置ノード名が“ノード１００”、状況が“ノード２００へ移動中”、クエリ状態への参照が“＆Ｑｕｅｒｙ１Ｓｔａｔｅ”、ロックが“無”、待ちイベントが“ｎｕｌｌ”という情報が登録されている。

【００７９】

これは、“Ｑｕｅｒｙ１”で示されるクエリがノード１００に現在割当てられていること、ノード１００からノード２００へ当該クエリを割当て変更中であることを示す。また、当該クエリのクエリ状態がポインタ“＆Ｑｕｅｒｙ１Ｓｔａｔｅ”で示される状態であ

ること、クエリ状態がロックされていないこと、当該クエリに対する待ちイベントが存在しないことを示す。

【００８０】

図１１は、配置表の例（その２）を示す図である。配置表２２１は、管理情報記憶部２２０に格納される。配置表２２１は、配置表１２１と同じタイミングの登録内容を例示している。配置表２２１に含まれる項目は、配置表１２１と同様であるため、説明を省略する。

【００８１】

例えば、配置表２２１では、“Ｑｕｅｒｙ１”で示されるクエリに対して、クエリ状態への参照が“ノード内に無し”であり、待ちイベントが“α５，α６”である点が、配置表１２１と異なる。これは、当該クエリは、現在割当て変更中（クエリ状態をノード２００へ移動中）であり、自ノード（ノード２００）では、クエリ状態を保持していないことを示す。また、当該クエリに対して、既にイベント“α５，α６”を待ちイベントとして取得済であることを示す。

【００８２】

また、配置表２２１では、“Ｑｕｅｒｙ８”で示されるクエリに対して、クエリ状態への参照“＆Ｑｕｅｒｙ８Ｓｔａｔｅ”が登録されている点が配置表１２１と異なる。これは、当該クエリの担当ノードがノード２００であり、ノード２００で当該クエリのクエリ状態を保持しているからである。

【００８３】

図１２は、配置表の例（その３）を示す図である。配置表３２１は、管理情報記憶部３２０に格納される。配置表３２１は、配置表１２１，２２１と同じタイミングの登録内容を例示している。配置表３２１に含まれる項目は、配置表１２１と同様であるため、説明を省略する。

【００８４】

例えば、配置表３２１では、“Ｑｕｅｒｙ１”で示されるクエリに対して、配置ノード名が“ノード２００”、状況が“稼働中”である点が配置表１２１，２２１と異なる。これは、ノード３００がクエリ名“Ｑｕｅｒｙ１”のクエリの割当て変更に伴うクエリ状態の送受信に関与しないためである。すなわち、各ノードは、管理ノード４００からクエリの割当て変更の指示を受け、自ノードが当該変更に伴うクエリ状態の送受信に関与しないと判断すると、直ちに配置ノード名の変更を行ってよい。

【００８５】

また、配置表３２１では、“Ｑｕｅｒｙ１０”で示されるクエリに対して、クエリ状態への参照“＆Ｑｕｅｒｙ１０Ｓｔａｔｅ”が登録されている点が配置表１２１，２２１と異なる。これは、当該クエリの担当ノードがノード３００であり、ノード３００で当該クエリのクエリ状態を保持しているからである。

【００８６】

なお、管理ノード４００も配置表１２１，２２１，３２１と同様に配置表を保持する。管理ノード４００の配置表では、各ノードに対する各クエリの最新の割当て状況が登録される。ただし、管理ノード４００が保持する配置表では、クエリ状態への参照、ロックおよび待ちイベントを管理しなくてよい。

【００８７】

図１３は、送信データ管理構造体の例を示す図である。送信データ管理構造体Ｄは、送信リストにおいて１つのクエリのクエリ状態を格納するための構造体である。ここで、送信リストとして、双方向リストを想定したデータ構造を例示するが、双方向リストでなくてもよい（例えば、単方向リストでもよい）。送信データ管理構造体Ｄは、Ｆｏｒｗａｒｄ、Ｂａｃｋｗａｒｄ、Ｑｕｅｒｙ　ＳｔａｔｅおよびＥｖｅｎｔｓの項目を含む。

【００８８】

Ｆｏｒｗａｒｄは、連結された次の送信データ管理構造体を示すポインタ（＆ＳｅｎｄＢｕｆＳｔｒｕｃｔｕｒｅ）である。Ｂａｃｋｗａｒｄは、連結された前の送信データ管

理構造体を示すポインタ（＆ＳｅｎｄＢｕｆＳｔｒｕｃｔｕｒｅ）である。ＱｕｅｒｙＳｔａｔｅは、送信対象のクエリ状態を示すポインタ（＆ＱｕｅｒｙＳｔａｔｅ）である。Ｅｖｅｎｔｓは、イベントを格納した配列を示すポインタ（＆Ｅｖｅｎｔｓ［］）である。

【００８９】

図１４は、送信リストの例を示す図である。送信リスト１２２は、管理情報記憶部１２０に格納される。送信リスト１２２は、ノード１００からノード２００への情報の送信に用いられる。送信リスト１２２は、第１の実施の形態の送信データ６の一例である。送信リストは、宛先毎に作成される。ノード１００からノード２００以外の他ノードへ情報を送信する際、ノード１００は他の送信リストを作成する。

【００９０】

送信リスト１２２は、送信データ管理構造体Ｄ（リスト要素ということがある）を複数連結可能な双方向リストである。送信リスト１２２は、リスト要素１２２ａ，１２２ｂ，１２２ｃを含む。リスト要素１２２ａは、送信リスト１２２の先頭（Ｈｅａｄ）である。また、リスト要素１２２ａは、送信リスト１２２がロックされているか否かを示す情報（例えば、フラグ）を含む（Ｌｏｃｋの項目に設定される）。

【００９１】

リスト要素１２２ｂは、リスト要素１２２ａの次のリスト要素である。リスト要素１２２ｃは、リスト要素１２２ｂの次のリスト要素である。リスト要素１２２ｂ，１２２ｃは、Ｆｏｒｗａｒｄ、Ｂａｃｋｗａｒｄ、Ｑｕｅｒｙ　Ｓｔａｔｅ、Ｅｖｅｎｔｓの項目を含む。具体的な設定内容は次の通りである。

【００９２】

リスト要素１２２ｂには、次の情報が設定されている。Ｆｏｒｗａｒｄには、リスト要素１２２ｃへのリンク（Ｆｏｒｗａｒｄリンク）を示すポインタが設定される。Ｂａｃｋｗａｒｄには、リスト要素１２２ａへのリンク（Ｂａｃｋｗａｒｄリンク）を示すポインタが設定される。Ｑｕｅｒｙ　Ｓｔａｔｅには、“Ｑｕｅｒｙ１”で示されるクエリのクエリ状態（Ｑｕｅｒｙ１Ｓｔａｔｅ）を示すポインタが設定される。当該クエリ状態は、“Ｑｕｅｒｙ１”で示されるクエリのクエリ状態である旨の情報も含む。Ｅｖｅｎｔｓには、当該クエリに対して受信したイベントの配列（［α１，α２，α３］）を示すポインタが設定される。

【００９３】

リスト要素１２２ｃには、次の情報が設定されている。Ｆｏｒｗａｒｄは、次のリスト要素が存在していないため、設定無し（ｎｕｌｌ）となる。Ｂａｃｋｗａｒｄには、リスト要素１２２ｂへのリンクを示すポインタが設定される。Ｑｕｅｒｙ　Ｓｔａｔｅには、“Ｑｕｅｒｙ１３”で示されるクエリのクエリ状態（Ｑｕｅｒｙ１３Ｓｔａｔｅ）を示すポインタが設定される。当該クエリ状態は、“Ｑｕｅｒｙ１３”で示されるクエリのクエリ状態である旨の情報も含む。Ｅｖｅｎｔｓは、当該クエリに対して受信したイベントが存在していないため、設定無し（ｎｕｌｌ）となる。

【００９４】

次に、第２の実施の形態のクエリの割当て変更時の処理手順を説明する。以下の説明では、ノード１００からノード２００へクエリの割当てを変更する場合を想定する。ただし、他ノード間でクエリの割当てを変更する場合も同様の手順となる。

【００９５】

図１５は、クエリ状態送信管理の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。以下の手順は、ノード１００からノード２００へクエリの割当てを変更する場合のノード１００の手順である。

【００９６】

（Ｓ１１）クエリ状態送信管理部１４０は、ノード１００が担当するクエリ（例えば、クエリ名“Ｑｕｅｒｙ１”のクエリ）をノード２００へ割当て変更する指示を管理ノード４００から受信する。クエリ状態送信管理部１４０は、配置表１２１を操作して、変更対

象のクエリのクエリ状態をロックする（ロック“無”から“有”に変更する）。
【００９７】
（Ｓ１２）クエリ状態送信管理部１４０は、配置表１２１を操作して、当該クエリに対応するエントリの状況の項目を“稼働中”から“ノード２００へ移動中”に変更する（割当て変更先のノード２００も同様の設定を行う）。以降、割当て変更が完了し、ノード２００で当該クエリの実行が再開されるまで、当該クエリの実行は中断される。
【００９８】
（Ｓ１３）クエリ状態送信管理部１４０は、送信データ管理構造体Ｄを生成する。
（Ｓ１４）クエリ状態送信管理部１４０は、配置表１２１を参照して、当該クエリに対応するエントリのクエリ状態への参照の項目に設定されたポインタを用いて、クエリ状態を取得する。クエリ状態送信管理部１４０は、配置表１２１を操作して、当該エントリのクエリ状態への参照の項目を、ステップＳ１３で生成した送信データ管理構造体Ｄへのポインタに変更する。
【００９９】
（Ｓ１５）クエリ状態送信管理部１４０は、配置表１２１を操作して、変更対象のクエリのクエリ状態をアンロックする（ロック“有”から“無”に変更する）。
（Ｓ１６）クエリ状態送信管理部１４０は、ステップＳ１４で取得したクエリ状態のシリアライズを実行する。
【０１００】
（Ｓ１７）クエリ状態送信管理部１４０は、シリアライズ済のクエリ状態をステップＳ１３で生成した送信データ管理構造体Ｄに登録する。具体的には、クエリ状態送信管理部１４０は、当該クエリ状態へのポインタを送信データ管理構造体Ｄに登録する。
【０１０１】
（Ｓ１８）クエリ状態送信管理部１４０は、送信リスト１２２をロックする。
（Ｓ１９）クエリ状態送信管理部１４０は、送信データ管理構造体Ｄ（クエリ状態登録済）を送信リスト１２２に連結する。
【０１０２】
（Ｓ２０）クエリ状態送信管理部１４０は、送信リスト１２２の合計のデータサイズが閾値以上であるか否かを判定する。閾値以上である場合、処理をステップＳ２１に進める。閾値未満である場合、処理をステップＳ２４に進める。なお、当該閾値には、例えばユーザにより、通信環境に応じた任意の値が設定され得る。
【０１０３】
（Ｓ２１）クエリ状態送信管理部１４０は、配置表１２１を操作して、送信リスト１２２に登録されているクエリ状態を全てロックする。配置表１２１において、状況の項目が“ノード２００へ移動中”と設定されているエントリのクエリ状態が、送信リスト１２２に登録されているクエリ状態である。
【０１０４】
（Ｓ２２）クエリ状態送信管理部１４０は、送信リスト１２２に登録されているクエリ状態をノード２００宛で、ネットワーク１０に送出する。クエリ状態送信管理部１４０は、配置表１２１を操作して、配置ノード名の項目を（“ノード１００”から）“ノード２００”に、当該クエリ状態に対応するエントリの状況の項目を“移動済”に、クエリ状態への参照の項目を“ノード内に無し”に変更する。クエリ状態送信管理部１４０は、配置表１２１を操作して、当該エントリのクエリ状態をアンロックする。
【０１０５】
（Ｓ２３）クエリ状態送信管理部１４０は、送信リスト１２２を参照して、ステップＳ２２で送信対象としたクエリ状態にイベントが付属していれば、当該イベントもノード２００に送信する。なお、ステップＳ２２，Ｓ２３は、送信リスト１２２に登録されたクエリ状態毎に実行される。送信リスト１２２に登録されたクエリ状態が複数であれば、クエリ状態送信管理部１４０は、クエリ状態毎にステップＳ２２，Ｓ２３を繰り返す。
【０１０６】

（Ｓ２４）クエリ状態送信管理部１４０は、送信リスト１２２をアンロックする。
このように、ノード１００は、割当て変更対象のクエリのクエリ状態をノード２００に送信する際に、当該クエリに対するイベントが送信リスト１２２に含まれていれば、クエリ状態とともに、そのイベントも送信する。

【０１０７】
その後、例えば、クエリ状態送信管理部１４０は管理ノード４００からクエリの割当て変更が完了した旨の通知を受け付ける。例えば、クエリ状態送信管理部１４０は、当該通知を受けると、配置表１２１の対応するクエリの状況の項目を（“移動済”から）“稼働中”に変更する。

【０１０８】
また、ステップＳ１２において、前述のようにクエリ状態の送受信に関与しないノード３００では、ステップＳ１１の指示に対して、配置表３２１の配置ノード名の項目を“ノード２００”に変更してよい。管理ノード４００でも同様に、割当て変更指示を行うとともに管理ノード４００が保持する配置表の配置ノード名の項目を“ノード２００”に変更してよい。

【０１０９】
図１６は、イベント受信管理（変更元）の例を示すフローチャートである。以下、図１６に示す処理をステップ番号に沿って説明する。ノード１００を例示するが、他ノードでも同様の手順となる。

【０１１０】
（Ｓ３１）イベント受信管理部１７０は、イベントを取得する。イベントの発行元は、ネットワーク２０に接続された何れかの装置でもよいし、ノード１００，２００，３００の何れかでもよい。イベント受信管理部１７０は、取得したイベントのデシリアライズを実行する（イベントの発行元がノード１００ならデシリアライズを行わなくてもよい）。

【０１１１】
（Ｓ３２）イベント受信管理部１７０は、当該イベントに含まれるストリーム名から、クエリの識別情報（クエリ名）を取得する。クエリ記憶部１１０は、ストリーム名と当該ストリーム名のイベントを処理するクエリのクエリ名との対応関係の情報を記憶している。よって、イベント受信管理部１７０は、当該情報を参照することで、ストリーム名からクエリ名を取得できる。イベント受信管理部１７０は、クエリ名をキーに配置表１２１から当該イベントに対応するエントリを検索する。

【０１１２】
（Ｓ３３）イベント受信管理部１７０は、配置表１２１を操作して、当該エントリのクエリ状態をロックする。
（Ｓ３４）イベント受信管理部１７０は、配置表１２１を参照して、当該エントリで示されるクエリを実行可能であるか否かを判定する。クエリを実行可能である場合、処理をステップＳ３５に進める。クエリを実行可能でない場合、処理をステップＳ３６に進める。クエリを実行可能である場合とは、当該クエリを自ノードが担当しており、状況が“稼働中”である場合である。クエリを実行可能でない場合とは、当該クエリを自ノードが担当していない場合や、当該クエリを自ノードが担当しているが状況が“稼働中”でない場合である。

【０１１３】
（Ｓ３５）クエリ実行管理部１３０は、取得したイベントを用いてクエリを実行し、実行結果に応じて当該クエリのクエリ状態を変更する。クエリ実行管理部１３０は、配置表１２１を操作して、当該クエリのクエリ状態をアンロックする。そして、処理を終了する。

【０１１４】
（Ｓ３６）イベント受信管理部１７０は、配置表１２１を参照して、取得したイベントに対応するクエリのクエリ状態がバッファリング中であるか否かを判定する。クエリ状態がバッファリング中である場合、処理をステップＳ３７に進める。クエリ状態がバッファ

リング中でない場合、処理をステップＳ４２に進める。クエリ状態がバッファリング中であるか否かは、当該クエリに対応する配置表１２１のエントリの状況の項目を参照することで判定できる。当該クエリを自ノードが担当しており、状況が“他ノード（例えば、ノード２００）に移動中”であれば、当該クエリ状態はバッファリング中である。状況が“他ノードに移動中”以外であれば、当該クエリ状態はバッファリング中でない。

【０１１５】

（Ｓ３７）イベント受信管理部１７０は、取得したイベントのシリアライズを実行する。

（Ｓ３８）イベント受信管理部１７０は、送信リストのうち、当該イベントに対応するクエリのクエリ状態を登録した送信データ管理構造体Ｄ（リスト要素）のリンクをロックする。

【０１１６】

（Ｓ３９）イベント受信管理部１７０は、当該送信データ管理構造体Ｄに、取得したイベントを登録する（イベント連結）。

（Ｓ４０）イベント受信管理部１７０は、当該送信データ管理構造体Ｄのリンクをアンロックする。

【０１１７】

（Ｓ４１）イベント受信管理部１７０は、配置表１２１を操作して、着目するクエリのクエリ状態をアンロックする。そして、処理を終了する。

（Ｓ４２）イベント受信管理部１７０は、配置表１２１を操作して、着目するクエリのクエリ状態をアンロックする。

【０１１８】

（Ｓ４３）イベント受信管理部１７０は、取得したイベントをイベント送信管理部１６０に送信させる。イベント送信管理部１６０による処理の詳細については後述する。

このように、イベント受信管理部１７０は、イベントを取得した際に、当該イベントを処理するクエリのクエリ状態が送信中の状態であれば、送信リスト内の当該クエリ状態を格納したリスト要素に、そのイベントを登録する（ステップＳ３７～Ｓ４０）。ここで、例えばｇａｔｈｅｒと呼ばれるＡＰＩ（Application Programming Interface）を用いることで、上述したクエリ状態送信管理部１４０やイベント受信管理部１７０の送信リストの作成処理を低コストで実現できる。

【０１１９】

また、自ノードが担当するクエリに対するイベントでクエリ状態が送信中の状態でなければ、通常通りクエリを実行する（ステップＳ３５）。更に、他ノードが担当するクエリに対するイベントを取得した場合には、イベント送信管理部１６０により、他ノードに当該イベントを送信させる（ステップＳ４３）。

【０１２０】

図１７は、クエリ状態受信管理の例を示すフローチャートである。以下、図１７に示す処理をステップ番号に沿って説明する。以下の手順は、ノード１００からノード２００へクエリの割当てを変更する場合のノード２００の手順である。なお、ノード２００は、ステップＳ５１よりも前にノード１００からノード２００へのクエリの割当て変更指示を管理ノード４００から受信している。ノード２００は、配置表２２１において、当該変更指示で指定されたクエリの状況を“ノード２００へ移動中”と設定している。

【０１２１】

（Ｓ５１）クエリ状態受信管理部２５０は、ノード１００から割当て変更対象のクエリ（例えば、クエリ名“Ｑｕｅｒｙ１”のクエリ）のクエリ状態を受信する。クエリ状態受信管理部２５０は、受信したクエリ状態のデシリアライズを実行し、クエリ記憶部２１０に格納する。

【０１２２】

（Ｓ５２）クエリ状態受信管理部２５０は、受信したクエリ状態にイベントが付属しているか否かを判定する。イベントが付属している場合、処理をステップＳ５３に進める。

イベントが付属していない場合、処理をステップＳ５５に進める。

【０１２３】

（Ｓ５３）クエリ状態受信管理部２５０は、付属しているイベント（例えば、イベントα１，α２，α３）のデシリアライズを実行し、クエリ実行管理部２３０に出力する。

（Ｓ５４）クエリ実行管理部２３０は、取得したイベントおよびクエリ状態により該当のクエリを実行し、クエリ記憶部２１０に記憶されたクエリ状態を変更する。

【０１２４】

（Ｓ５５）クエリ状態受信管理部２５０は、管理情報記憶部２２０に記憶された配置表２２１を参照して、当該クエリのエントリを検索する。

（Ｓ５６）クエリ状態受信管理部２５０は、当該エントリをロックする。

【０１２５】

（Ｓ５７）クエリ状態受信管理部２５０は、当該エントリのクエリ状態への参照の項目に、クエリ記憶部２１０に記憶されたクエリ状態へのポインタを登録する。当該ポインタで示されるクエリ状態は、ステップＳ５４を実行していなければ、ステップＳ５１で取得したクエリ状態である。一方、ステップＳ５４を実行していれば、ステップＳ５４の実行結果に応じたクエリ状態である。

【０１２６】

（Ｓ５８）クエリ状態受信管理部２５０は、当該エントリに待ちイベントがあるか否かを判定する。待ちイベントがある場合、処理をステップＳ５９に進める。待ちイベントがない場合、処理をステップＳ６０に進める。

【０１２７】

（Ｓ５９）クエリ実行管理部２３０は、待ちイベント（例えば、イベントα５，α６）および現在のクエリ状態により該当のクエリを実行し、クエリ状態を変更する。

（Ｓ６０）クエリ状態受信管理部２５０は、配置表２２１を操作して、ステップＳ５５で検索されたエントリの配置ノード名の項目を（“ノード１００”から）“ノード２００”に、状況の項目を“移動済”に変更する。

【０１２８】

（Ｓ６１）クエリ状態受信管理部２５０は、当該エントリをアンロックする。

このように、クエリ状態受信管理部２５０は、受信したクエリ状態にイベントが付属していれば、当該イベントおよびクエリ状態を用いてクエリを実行し、クエリ状態を更新する。更に、クエリ状態受信管理部２５０は、当該クエリに対して待ちイベントが存在する場合には、待ちイベントを用いて当該クエリを実行し、クエリ状態を更新する。

【０１２９】

このとき、待ちイベントよりも前に発生し当該クエリに用いられるがノード２００に未到着である未到着イベントが存在することもある（後述する）。この場合でも、ノード２００によるクエリ状態および待ちイベントを用いたクエリの実行を許容する。

【０１３０】

なお、ステップＳ５１の後の何れかのタイミングで、クエリ状態受信管理部２５０は、クエリ状態を適切に受信した旨を管理ノード４００に通知する。例えば、クエリ状態受信管理部２５０は、当該通知に対する管理ノード４００からの応答（割当て変更完了の通知）を受けると、配置表２２１の当該クエリの状況の項目を“稼働中”に変更する（ステップＳ６０よりも前に“稼働中”になっていればステップＳ６０では“移動済”に変更しなくてよい）。

【０１３１】

図１８は、イベント受信管理（変更先）の例を示すフローチャートである。以下、図１８に示す処理をステップ番号に沿って説明する。ノード２００を例示するが、他ノードでも同様の手順となる。

【０１３２】

（Ｓ７１）イベント受信管理部２７０は、イベントを取得する。イベントの発行元は、ネットワーク２０に接続された何れかの装置でもよいし、ノード１００，２００，３００

でもよい。イベント受信管理部２７０は、取得したイベントのデシリアライズを実行する（イベントの発行元がノード２００ならデシリアライズを行わなくてもよい）。

【０１３３】

（Ｓ７２）イベント受信管理部２７０は、当該イベントに含まれるストリーム名から、クエリの識別情報（クエリ名）を取得する。クエリ記憶部２１０は、ストリーム名と当該ストリーム名のイベントを処理するクエリのクエリ名との対応関係の情報を記憶している。よって、イベント受信管理部２７０は、当該情報を参照することで、ストリーム名からクエリ名を取得できる。イベント受信管理部２７０は、クエリ名をキーに配置表２２１から当該イベントに対応するエントリを検索する。

【０１３４】

（Ｓ７３）イベント受信管理部２７０は、配置表２２１を操作して、当該クエリのクエリ状態をロックする。

（Ｓ７４）イベント受信管理部２７０は、配置表２２１を参照して、当該エントリのクエリを実行可能であるか否かを判定する。クエリを実行可能である場合、処理をステップＳ７５に進める。クエリを実行可能でない場合、処理をステップＳ７６に進める。クエリを実行可能である場合とは、当該クエリを自ノードが担当しており、状況が“稼働中”または“移動済”である場合である。クエリを実行可能でない場合とは、当該クエリを自ノードが担当していない場合や、当該クエリを自ノードが担当しているが状況が“稼働中”または“移動済”でない場合である。

【０１３５】

（Ｓ７５）クエリ実行管理部２３０は、取得したイベントを用いてクエリを実行し、クエリ記憶部２１０に記憶された当該クエリのクエリ状態を変更する。クエリ実行管理部２３０は、配置表２２１を操作して、クエリ状態をアンロックする。そして、処理を終了する。

【０１３６】

（Ｓ７６）イベント受信管理部２７０は、配置表２２１を参照して、取得したイベントに対応するクエリについて、クエリ状態の到着待ちであるか否かを判定する。クエリ状態の到着待ちである場合、処理をステップＳ７７に進める。クエリ状態の到着待ちでない場合、処理をステップＳ７９に進める。クエリ状態の到着待ちである場合とは、当該クエリ状態のエントリの状況の項目に“自ノード（ここでは、ノード２００）へ移動中”と設定されている場合である。クエリ状態の到着待ちでない場合とは、当該クエリ状態のエントリの状況の項目に“自ノードへ移動中”以外の情報が設定されている場合である。

【０１３７】

（Ｓ７７）イベント受信管理部２７０は、配置表２２１の当該エントリの待ちイベントの項目に、取得したイベント（例えば、イベントα５やイベントα６）を登録する。

（Ｓ７８）イベント受信管理部２７０は、配置表２２１を操作して、当該エントリのクエリ状態をアンロックする。そして、処理を終了する。

【０１３８】

（Ｓ７９）イベント受信管理部２７０は、配置表２２１を操作して、取得したイベントに対応するクエリのクエリ状態をアンロックする。

（Ｓ８０）イベント受信管理部２７０は、取得したイベントをイベント送信管理部２６０に送信させる。

【０１３９】

このように、イベント受信管理部２７０は、クエリ状態の到着待ちであるクエリに対するイベントを取得すると、当該イベントを待ちイベントとして配置表２２１に登録する（ステップＳ７６，Ｓ７７）。それ以外のイベントで自ノードが担当するクエリに対するイベントを取得した場合には、通常通りクエリを実行する（ステップＳ７５）。また、他ノードが担当するクエリに対するイベントを取得した場合には、イベント送信管理部２６０により、他ノードに当該イベントを送信させる（ステップＳ８０）。

【０１４０】

次に、イベント送信管理部１６０，２６０，３６０によるイベント送信管理の手順を説明する。以下では、イベント送信管理部１６０を例示するが、イベント送信管理部２６０，３６０による手順も同様である。

【０１４１】

図１９は、イベント送信管理の例を示すフローチャートである。以下、図１９に示す処理をステップ番号に沿って説明する。

（Ｓ８１）イベント送信管理部１６０は、イベントを取得する。イベントの発行元は、ネットワーク２０に接続された何れかの装置でもよいし、ノード１００，２００，３００の何れかでもよい。前述のように、イベント送信管理部１６０は、イベント受信管理部１７０から他ノードが担当するクエリのイベントを取得することもある。イベント送信管理部１６０は、イベントのシリアライズを実行する。

【０１４２】

（Ｓ８２）イベント送信管理部１６０は、配置表１２１を参照して、当該イベントの送信先のノードを特定する。なお、図１６のステップＳ３２や図１８のステップＳ７２と同様に、イベント送信管理部１６０は、イベントに含まれるストリーム名から、当該イベントを処理するクエリのクエリ名を特定できる。当該クエリ名のクエリを担当するノードが当該イベントの送信先のノードである。イベント送信管理部１６０は、当該送信先に対応する送信リストをロックする。

【０１４３】

（Ｓ８３）イベント送信管理部１６０は、シリアライズ済のイベントを当該送信リストに追加する。

（Ｓ８４）イベント送信管理部１６０は、当該送信リストの合計のデータサイズが閾値を超えたか否かを判定する。閾値以上である場合、処理をステップＳ８５に進める。閾値未満である場合、処理をステップＳ８６に進める。なお、当該閾値には、例えばユーザにより、通信環境に応じた任意の値が設定され得る。

【０１４４】

（Ｓ８５）イベント送信管理部１６０は、当該送信リスト内のイベントを送信先のノードを宛先として、ネットワーク１０に送出する。このとき、他のイベントなどのデータが当該送信リストに格納されていれば、そのデータも送出する。

【０１４５】

（Ｓ８６）イベント送信管理部１６０は、当該送信リストをアンロックする。

このように、イベント送信管理部１６０は、担当ノードが自ノードでないクエリに対するイベントを取得すると、当該イベントを担当ノードに送信する。

【０１４６】

図２０は、クエリ状態の送信例（その１）を示す図である。図２０では、ノード１００からノード２００へ、クエリ名“Ｑｕｅｒｙ１”のクエリの割当てを変更する場合を例示している。ノード１００は、当該クエリのノード２００への割当て変更指示を管理ノード４００から受信すると、当該クエリのクエリ状態のシリアライズを実行し、送信リスト１２２に追加する。ノード１００において、当該クエリ状態は送信中として管理される。ノード１００は、送信リスト１２２が所定サイズに達するまで、送信対象のデータを送信リスト１２２に追加する（バッファリング）。

【０１４７】

（１）ノード１００は、バッファリングの間に、イベントα１，α２，α３を順に取得する。ノード１００は、イベントα１，α２，α３が割当て変更対象のクエリに対するイベントであることを特定する。

【０１４８】

（２）ノード１００は、イベントα１，α２，α３のシリアライズを実行し、送信リスト１２２内の当該クエリのリスト要素に、順番に追加する。

図２１は、クエリ状態の送信例（その２）を示す図である。図２１では、図２０の後の処理を例示している。なお、前述のイベントα１，α２，α３に加えて、イベントα４，

α５，α６，α７もクエリ名“Ｑｕｅｒｙ１”のクエリで用いられるイベントである。

【０１４９】

（３）ノード１００は、送信リスト１２２のデータサイズが閾値に達すると、送信リスト１２２の内容を、ノード２００を宛先としてネットワーク１０に送出する。これにより、クエリ名“Ｑｕｅｒｙ１”のクエリのクエリ状態とともに、イベントα１，α２，α３も送出される。

【０１５０】

（４）次に、ノード１００はノード３００からイベントα４を取得する。このような状況が発生するのは、割当て変更指示がノード３００に到着する以前に、ノード３００がイベントα４をネットワーク１０に送出する場合である。ノード３００の配置表３２１ではイベントα４を送出するタイミングで、イベントα４に対するクエリの担当ノードがノード１００のままであったためである。ノード１００はイベントα４を、イベントα１，α２，α３よりも後のタイミングでノード２００に送信することになる。

【０１５１】

（５）ノード１００により送出された送信リスト１２２の内容がノード２００に到着する前に、ノード２００はイベントα５，α６を取得する。この場合、ノード２００は、イベントα５，α６を待ちイベントとして保持する。

【０１５２】

（６）ノード２００は、ノード３００からイベントα７を取得する。このような状況が発生するのは、割当て変更指示がノード３００に到着した後にノード３００がイベントα７をネットワーク１０に送出する場合である（ノード３００の配置表３２１ではイベントα７に対するクエリの担当ノードがノード２００に変更されている）。

【０１５３】

上記の例では、ノード２００は、ノード１００からクエリ状態およびイベントα１，α２，α３を受信すると、当該クエリ状態およびイベントα１，α２，α３を用いてクエリ名“Ｑｕｅｒｙ１”のクエリを実行する。その後、ノード２００は、待ちイベントであるイベントα５，α６を用いて当該クエリを実行する。更に、ノード２００は、イベントα４，α７をノード２００に到着した順に用いて、当該クエリを実行する。

【０１５４】

このように、イベントα１，α２，α３の処理をクエリ状態とともにノード２００に送信する。このため、当該クエリ状態を取得したノード２００は、イベントα１，α２，α３を用いて、直ちにクエリ名“Ｑｕｅｒｙ１”のクエリを実行できる。すなわち、イベントα１，α２，α３を当該クエリ状態とともにノード２００に送信しない場合よりも、ノード２００によるクエリの実行再開を短縮化できる。

【０１５５】

ここで、第２の実施の形態の方法を用いると、例えば、イベントα１，α２，α３，α５，α６，α４，α７をこの順番で用いて、クエリが実行されることもある。一方、クエリによってはイベントの発生順を重視するものもある。例えば、イベントα１，α２，α３，α４，α５，α６，α７がこの順番に発生したなら、当該発生順で各イベントを用いてクエリを実行したいこともある（例えば、イベントの時間差検出を厳密に行う科学的計測の場合など）。そこで、イベントの発生順を厳密に維持してクエリ実行するか、イベントの発生順と異なる順番で各イベントを用いたクエリ実行を許容するかを指定する情報を、各クエリに含めてもよい。

【０１５６】

例えば、イベントの発生順と異なる順番でのクエリ実行を許容するクエリには、その旨を示すアノテーション（注釈を示すメタデータ。例えば、“＠ＸＸＸ”などの文字列。）を当該クエリ内に記述可能とすることが考えられる。例えば、クエリ内に当該アノテーションが記述されている場合には、上記の方法により、クエリ状態にイベントを追加して送信可能とする。クエリ内に当該アノテーションが記述されていない場合には、例えば、クエリ状態の送信が完了するまで、各ノードは対象のクエリに対する全てのイベントの担当

ノードへの送信を保留し、クエリ状態の送信完了後に発生順で担当ノードに各イベントを処理させる。

【０１５７】

このようにすれば、クエリ状態とともにイベントを送信してクエリ実行する方法と、イベントの発生順を厳守してクエリ実行する方法とを両立できる。特に、イベントの発生順に拘らずにイベント処理を行えるようにすれば、未到着のイベント（上記の例ではイベントα４）を待たずに、待ちイベント（上記の例ではイベントα５，α６）のイベント処理を行える。このため、割当て変更後の担当ノードによるイベント処理を高速に開始できる。

【０１５８】

また、クエリ状態にイベントが付属しているときは、付属する当該イベントを先に利用し、待ちイベントをその後で用いることで、（発生順を入れ替えた入力順でのイベント処理を許容するものの）イベント発生順をある程度保ってクエリ実行できる。クエリ状態に付属するイベントの方が、待ちイベントよりも早いタイミングで生成されたものである可能性が高いからである。イベント間の発生タイミングの時間差として、比較的小さな時間差のイベント間の入れ替わりを許容するが、比較的大きな時間差のイベント間の入れ替わりを許容しない場合に有用である。

【０１５９】

なお、図２１の（４）の例において、送信リスト１２２に含まれる全情報のネットワーク１０への送出が完了する前に、イベントα４の到着や送信準備が間に合えば、当該全情報の中にイベントα４の情報を割り込ませて送出してもよい。例えば、他の情報に代えて、イベントα４を優先して送出してもよい。すると、ノード１００は、クエリ状態とともにイベントα１，α２，α３，α４をノード２００に提供できる。

【０１６０】

図２２は、クエリ割当て変更の例（その１）を示すシーケンス図である。以下、図２２に示す処理をステップ番号に沿って説明する。

（ＳＴ１１）管理ノード４００は、ノード１００からノード２００へのクエリの割当て変更をノード１００，２００，３００に指示する。当該指示は、この段階ではノード１００，２００，３００に未だ到着していない。

【０１６１】

（ＳＴ１２）ノード３００は、新たなイベントを取得する（ノード３００でイベント発生）。当該イベントは、割当て変更対象のクエリに対するイベントである。

（ＳＴ１３）ノード３００は、配置表３２１を参照し、当該クエリの担当ノードであるノード１００を特定する。ノード３００は、ノード１００へ宛てて、取得したイベントを送出する。当該イベントは、この段階ではノード１００に未だ到着していない。

【０１６２】

（ＳＴ１４）ノード１００，２００，３００は、ステップＳＴ１１の指示を受信する。ノード１００，２００は、配置表１２１，２２１を操作して、当該クエリの状況を“ノード２００へ移動中”とする。ノード３００は、配置表３２１の当該クエリの担当ノード名を“ノード２００”とする。ノード１００は、指示されたクエリのクエリ状態のシリアライズを実行し、当該クエリ状態を登録したリスト要素を送信リスト１２２に追加する。その後、ノード１００は、ノード３００から当該クエリに対するイベントを受信する。ノード１００は、当該イベントのシリアライズを実行し、上記クエリ状態のリスト要素に追加する。

【０１６３】

（ＳＴ１５）ノード１００は、送信リスト１２２のデータサイズが閾値以上になるまでバッファリングする。

（ＳＴ１６）ノード１００は、送信リスト１２２のデータサイズが閾値以上になると、送信リスト１２２の内容をノード２００へ宛てて送出する。このとき、ノード１００は、ステップＳＴ１４で送信リスト１２２に登録したクエリ状態も送出する。

【０１６４】
（ＳＴ１７）ノード１００は、ステップＳＴ１４で当該クエリ状態に付加したイベントも送出する。
（ＳＴ１８）ノード２００は、ノード１００からクエリ状態を受信すると、当該クエリ状態のデシリアライズを実行する。ノード２００は、ノード１００からクエリ状態に付属するイベントを受信すると、当該イベントのデシリアライズを実行する。
【０１６５】
（ＳＴ１９）ノード２００は、クエリ状態の移動が完了した旨を管理ノード４００に通知する。
（ＳＴ２０）ノード２００は、ステップＳＴ１８で取得したクエリ状態およびイベントを用いて、ノード１００からノード２００へ割当て変更されたクエリを実行する。
【０１６６】
このように、クエリの割当て変更指示がノード３００に到達する前に、ノード３００からノード１００へ当該クエリに対するイベントが送出され得る。この場合、新たな担当ノードであるノード２００が当該イベントを用いて割当て変更対象のクエリを実行できるまでの遅延時間（イベント転送レイテンシと呼ぶ）は、ステップＳＴ１２からステップＳＴ１８の完了までとなる。
【０１６７】
なお、管理ノード４００は、ステップＳＴ１９の後、クエリの割当て変更が完了した旨をノード１００，２００，３００に通知してもよい。
図２３は、クエリ割当て変更の比較例（その１）を示すシーケンス図である。以下、図２３に示す処理をステップ番号に沿って説明する。図２３では、第２の実施の形態の方法を用いない場合を想定して図２２に対する比較例を説明する。図２３の説明では、便宜的に第２の実施の形態と同じ符号を用いて各ノードを示す。
【０１６８】
（ＳＴ２１）管理ノード４００は、ノード１００からノード２００へのクエリの割当て変更をノード１００，２００，３００に指示する。当該指示は、この段階ではノード１００，２００，３００に未だ到着していない。
【０１６９】
（ＳＴ２２）ノード３００は、新たなイベントを取得する（ノード３００でイベント発生）。当該イベントは、割当て変更対象のクエリに対するイベントである。
（ＳＴ２３）ノード３００は、当該クエリの担当ノードをノード１００と特定する。ノード３００は、ノード１００へ宛てて、取得したイベントを送出する。当該イベントは、この段階ではノード１００に未だ到着していない。
【０１７０】
（ＳＴ２４）ノード１００，２００，３００は、ステップＳＴ２１の指示を受信する。ノード１００は、指示されたクエリのクエリ状態のシリアライズを実行し、送信データに追加する。また、ノード１００は、ノード３００から当該クエリに対するイベントを受信する。
【０１７１】
（ＳＴ２５）ノード１００は、送信データのデータサイズが閾値以上になるまでバッファリングする。
（ＳＴ２６）ノード１００は、送信データのデータサイズが閾値以上になると、送信データの内容をノード２００へ宛てて送出する。当該送信データは、割当て変更対象のクエリのクエリ状態を含む。一方、当該送信データは、ステップＳＴ２４で受信したイベントを含んでいない。
【０１７２】
（ＳＴ２７）ノード２００は、ノード１００からクエリ状態を受信すると、当該クエリ状態のデシリアライズを実行する。
（ＳＴ２８）ノード２００は、クエリ状態の移動が完了した旨を管理ノード４００に通

知する。

【０１７３】

（ＳＴ２９）管理ノード４００は、クエリの割当て変更が完了した旨をノード１００，２００，３００に通知する。ノード１００，２００，３００は、当該通知を受信する。

（ＳＴ３０）ノード１００は、ノード３００から受信していたイベントを、割当て変更後の新たな担当ノードであるノード２００に送信する（当該イベントに対してもシリアライズやバッファリングを行う）。ノード２００は、ノード１００からイベントを受信すると、当該イベントのデシリアライズを実行する。

【０１７４】

（ＳＴ３１）ノード２００は、ステップＳＴ２７で取得したクエリ状態およびステップＳＴ３０で取得したイベントを用いて、ノード１００からノード２００へ割当て変更されたクエリを実行する。

【０１７５】

上記比較例の場合、ノード３００で発生したイベントをノード２００が受信完了するまでのイベント転送レイテンシは、ステップＳＴ２２からステップＳＴ３０の完了までとなる。比較例では、図２２のイベント転送レイテンシに比べて、ステップＳＴ２８～ＳＴ３０の時間が余計にかかる。特に、ステップＳＴ３０では、図示を省略しているが、ステップＳＴ２４～ＳＴ２６と同様にロック待ち、バッファリング、転送、デシリアライズ、登録、アンロック待ちという複数段階のフェーズを含み、これら処理のレイテンシも含まれることになる。

【０１７６】

一方、図２２の場合は、ステップＳＴ２８～ＳＴ３０の時間分の遅延を削減できる。よって、比較例に比べて、割当て変更対象のクエリをノード２００により実行再開できるまでの遅延を短縮できる。

【０１７７】

なお、ステップＳＴ２４，ＳＴ２５で送信データのデータサイズが閾値に達していなければ、ステップＳＴ２６でノード３００から到着したイベントをクエリ状態とともにノード２００に送信できる可能性もある。ただし、比較例の場合では、送信データを単にバッファリングするのみである。すなわち、第２の実施の形態の方法のように、送信データ管理構造体Ｄを用いてクエリ状態とイベントとを一塊として管理していない。したがって、ノード１００からノード２００へクエリ状態とイベントとが関連性なく送信されることになる。この場合、ノード２００では、受信したデータの中からクエリ状態と当該クエリ状態に関係するイベントとを検索する演算コストや遅延も生じ得る。この演算コストや遅延も改善の余地がある。

【０１７８】

一方、図２２の場合は、送信データ管理構造体Ｄを用いてクエリ状態とイベントとを一塊で管理する。これにより、クエリ状態とイベントとをノード１００から連続して送信可能とする。ノード２００では、受信データのデシリアイズの際に、クエリ状態の直後にイベントを抽出すれば、当該イベントを直前のクエリ状態に付属するイベントと判断できる。すなわち、クエリ状態とイベントとをノード２００により効率的に取得させ、割当て変更対象のクエリの実行再開をより短縮できる（上記の演算コストや遅延を改善し得る）。このように、クエリ状態とイベントとをノード１００から連続して送信する方が、より好ましい。

【０１７９】

図２４は、クエリ割当て変更の例（その２）を示すシーケンス図である。以下、図２４に示す処理をステップ番号に沿って説明する。

（ＳＴ４１）管理ノード４００は、ノード１００からノード２００へのクエリの割当て変更をノード１００，２００，３００に指示する。

【０１８０】

（ＳＴ４２）ノード１００，２００，３００は、ステップＳＴ４１の指示を受信する。

ノード１００，２００は、配置表１２１，２２１を操作して、当該クエリの状況を“ノード２００へ移動中”とする。ノード３００は、配置表３２１の当該クエリの担当ノード名を“ノード２００”とする。ノード１００は、指示されたクエリのクエリ状態のシリアライズを実行し、当該クエリ状態を送信リスト１２２に追加する。

【０１８１】

（ＳＴ４３）ノード３００は、新たなイベントを取得する（ノード３００でイベント発生）。当該イベントは、割当て変更対象のクエリに対するイベントである。

（ＳＴ４４）ノード３００は、配置表３２１を参照し、当該クエリの担当ノードであるノード２００を特定する。ノード３００は、ノード２００へ宛てて、取得したイベントを送出する。その後、ノード２００は当該イベントを受信し、受信したイベントのデシリアライズを行う。ノード２００は、割当て変更対象のクエリの待ちイベントとして、当該イベントを配置表２２１に登録する。ノード２００の配置表２２１では、ノード３００から受信したイベントに対するクエリの状況が“自ノード（ここでは、ノード２００）へ移動中”となっているためである。

【０１８２】

（ＳＴ４５）ノード１００は、送信リスト１２２のデータサイズが閾値以上になるまでバッファリングする。

（ＳＴ４６）ノード１００は、送信リスト１２２のデータサイズが閾値以上になると、送信リスト１２２の内容をノード２００へ宛てて送出する。

【０１８３】

（ＳＴ４７）ノード２００は、ノード１００からクエリ状態を受信すると、当該クエリ状態のデシリアライズを実行する。

（ＳＴ４８）ノード２００は、クエリ状態の移動が完了した旨を管理ノード４００に通知する。

【０１８４】

（ＳＴ４９）ノード２００は、ステップＳＴ４６で取得したクエリ状態およびステップＳＴ４４で取得したイベント（待ちイベント）を用いて、ノード１００からノード２００へ割当て変更されたクエリを実行する。

【０１８５】

このように、クエリの割当て変更指示がノード３００に到達した直後に、ノード３００からノード２００へ当該クエリに対するイベントが送出され得る。この場合、イベント転送レイテンシは、ステップＳＴ４３からステップＳＴ４７の完了までとなる。

【０１８６】

なお、管理ノード４００は、ステップＳＴ４８の後、クエリの割当て変更が完了した旨をノード１００，２００，３００に通知してもよい。

図２５は、クエリ割当て変更の比較例（その２）を示すシーケンス図である。以下、図２５に示す処理をステップ番号に沿って説明する。図２５では、第２の実施の形態の方法を用いない場合を想定して図２４に対する比較例を説明する。図２５の説明では、便宜的に第２の実施の形態と同じ符号を用いて各ノードを示す。

【０１８７】

（ＳＴ５１）管理ノード４００は、ノード１００からノード２００へのクエリの割当て変更をノード１００，２００，３００に指示する。

（ＳＴ５２）ノード１００，２００，３００は、ステップＳＴ５１の指示を受信する。ノード１００は、指示されたクエリのクエリ状態のシリアライズを実行し、送信データに追加する。

【０１８８】

（ＳＴ５３）ノード３００は、新たなイベントを取得する（ノード３００でイベント発生）。当該イベントは、割当て変更対象のクエリに対するイベントである。ノード３００は、当該イベントが割当て変更対象のクエリに対するものであることを検出すると、当該イベントの担当ノードへの提供を保留する。

【０１８９】
（ＳＴ５４）ノード１００は、送信データのデータサイズが閾値以上になるまでバッファリングする。
（ＳＴ５５）ノード１００は、送信データのデータサイズが閾値以上になると、送信データの内容をノード２００へ宛てて送出する。当該送信データには、クエリ状態が含まれる。
【０１９０】
（ＳＴ５６）ノード２００は、ノード１００からクエリ状態を受信すると、当該クエリ状態のデシリアライズを実行する。
（ＳＴ５７）ノード２００は、クエリ状態の移動が完了した旨を管理ノード４００に通知する。
【０１９１】
（ＳＴ５８）管理ノード４００は、クエリの割当て変更が完了した旨をノード１００，２００，３００に通知する。ノード１００，２００，３００は、当該通知を受信する。
（ＳＴ５９）ノード３００は、ステップＳＴ５３で取得したイベントの送信先をノード２００と特定する。ノード３００は、当該イベントをノード２００に送信する（当該イベントに対してもシリアライズやバッファリングを行う）。ノード２００は、ノード３００からイベントを受信すると、当該イベントのデシリアライズを実行する。
【０１９２】
（ＳＴ６０）ノード２００は、ステップＳＴ５６で取得したクエリ状態およびステップＳＴ５９で取得したイベントを用いて、ノード１００からノード２００へ割当て変更されたクエリを実行する。
【０１９３】
上記比較例の場合、ノード３００で発生したイベントをノード２００が受信完了するまでのイベント転送レイテンシは、ステップＳＴ５３からステップＳＴ５９の完了までとなる。比較例では、図２４のイベント転送レイテンシに比べて、ステップＳＴ５７～ＳＴ５９の時間が余計にかかる。特に、ステップＳＴ５９では、図示を省略しているが、ステップＳＴ５２～ＳＴ５６と同様にロック待ち、バッファリング、転送、デシリアライズ、登録、アンロック待ちという複数段階のフェーズを含み、これら処理のレイテンシも含まれることになる。
【０１９４】
一方、図２４の場合はステップＳＴ４４において、ノード３００からノード２００へのイベントの送信を許容し、ノード２００では当該イベントを待ちイベントとして管理する。このため、ステップＳＴ５７～ＳＴ５９の時間分の遅延を削減できる。すなわち、比較例に比べて、割当て変更対象のクエリをノード２００により実行再開できるまでの遅延を短縮できる。
【０１９５】
図２６は、分散処理システムの他の例を示す図である。図２では、サーバコンピュータ（物理マシン）を用いてノード１００，２００，３００を実現する例を示した。一方、仮想マシンを用いて各ノードを実現してもよい。例えば、サーバ５１，５２，５３をネットワーク１０に接続する。ここで、図２６ではネットワーク２０および他の装置の図示を省略している。
【０１９６】
例えば、サーバ５１，５２，５３は、ハイパーバイザや仮想マシンモニタなどと呼ばれる管理用のソフトウェアを実行し、サーバ５１，５２，５３上の複数の仮想マシンに、サーバ５１，５２，５３が備えるＣＰＵやＲＡＭなどのリソースを割当てる。例えば、サーバ５１は、仮想マシン１００ａ，２００ａ，３００ａを有する。
【０１９７】
仮想マシン１００ａ，２００ａ，３００ａを、分散処理を行うノードとして用いることができる。例えば、仮想マシン１００ａは、ノード１００と同一の機能を実現できる。仮

想マシン２００ａは、ノード２００と同一の機能を実現できる。仮想マシン３００ａは、ノード３００と同一の機能を実現できる。なお、物理マシンと仮想マシンとが分散処理を行うノードとして混在してもよい。このように、仮想マシン１００ａ，２００ａ，３００ａを用いて分散処理を行う場合も、クエリの割当て変更先で当該クエリ実行を再開するまでの遅延を短縮できる。

【０１９８】

すなわち、ノード１００，２００，３００は、第１の実施の形態のマシン（物理マシン）の一例である。仮想マシン１００ａ，２００ａ，３００ａは、第１の実施の形態のマシン（仮想マシン）の一例である。

【０１９９】

また、上記の例では、クエリ毎に担当ノードを割当てるものとした。一方、クエリおよび当該クエリに対するイベントに含まれる所定のキーに応じて、クエリの割当てを行うことも考えられる。同じクエリでも、キーに応じて担当ノードを別個にしたいこともあるからである。具体的には、複数の地域のうちの何れの地域で発生したイベントかを示す情報（キー）が、当該イベントに含まれる場合に、当該キーに応じて担当ノードを決定することが考えられる。その場合、配置表１２１，２２１，３２１では、クエリとキーとの組に応じて、配置ノード名を管理することができる。その場合でも、上記の説明において「クエリとキーとの組」に対して「クエリ名」が与えられると考えれば、上記と同様の方法を適用できる。

【０２００】

なお、第１の実施の形態の情報処理は、演算部１ｂにプログラムを実行させることで実現できる。また、第２の実施の形態の情報処理は、プロセッサ１０１にプログラムを実行させることで実現できる。プログラムは、コンピュータ読み取り可能な記録媒体（例えば、光ディスク１３、メモリ装置１４およびメモリカード１６など）に記録できる。

【０２０１】

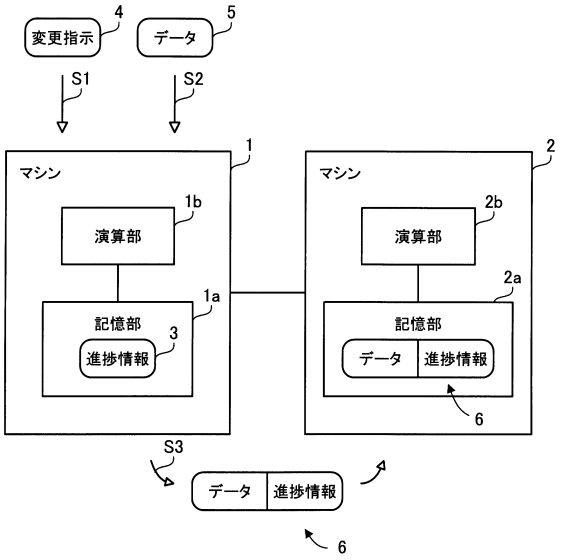
例えば、プログラムを記録した記録媒体を配布することで、プログラムを流通させることができる。また、プログラムを他のコンピュータに格納しておき、ネットワーク経由でプログラムを配布してもよい。コンピュータは、例えば、記録媒体に記録されたプログラムまたは他のコンピュータから受信したプログラムを、ＲＡＭ１０２やＨＤＤ１０３などの記憶装置に格納し（インストールし）、当該記憶装置からプログラムを読み込んで実行してもよい。

【符号の説明】

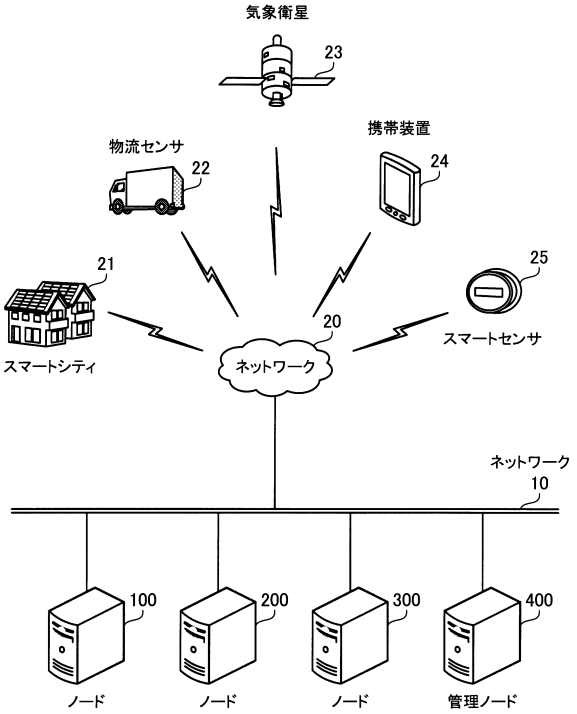
【０２０２】

１，２　マシン
１ａ，２ａ　記憶部
１ｂ，２ｂ　演算部
３　進捗情報
４　変更指示
５　データ
６　送信データ

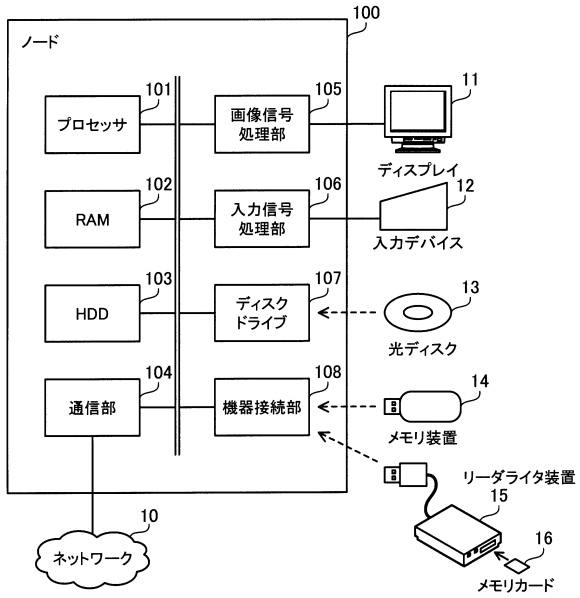
【図１】


変更指示
4
データ
5
S1
S2
マシン
1
演算部
1b
記憶部
1a
進捗情報
3
マシン
2
演算部
2b
記憶部
2a
データ
進捗情報
6
S3
データ
進捗情報
6

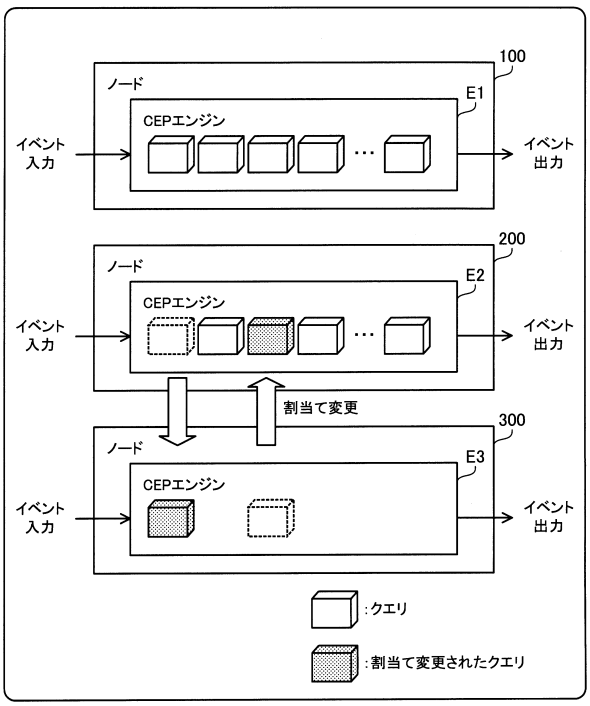
【図２】


気象衛星
23
物流センサ
22
携帯装置
24
21
25
スマートシティ
20
ネットワーク
スマートセンサ
ネットワーク
10
100
200
300
400
ノード
ノード
ノード
管理ノード

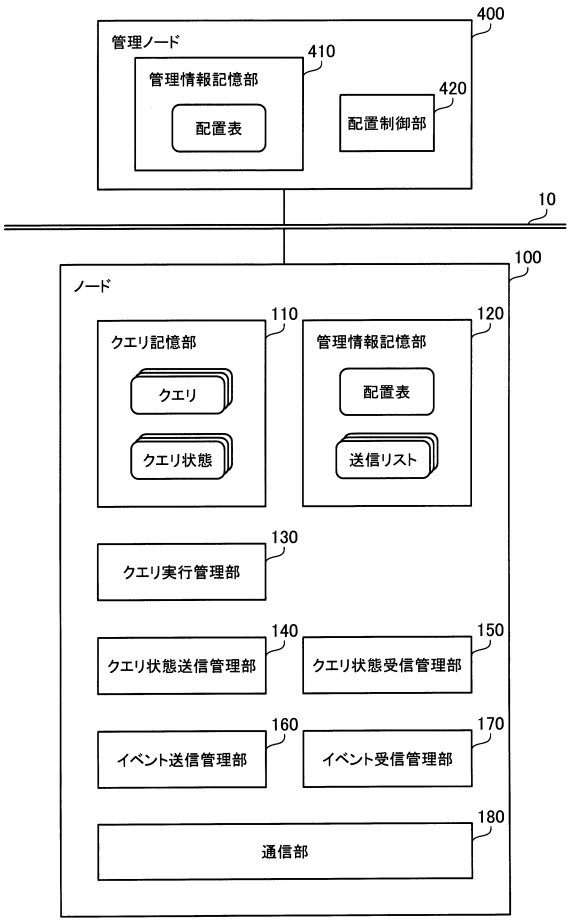
【図３】


ノード
100
プロセッサ
101
RAM
102
HDD
103
通信部
104
画像信号処理部
105
入力信号処理部
106
ディスクドライブ
107
機器接続部
108
11
ディスプレイ
12
入力デバイス
13
光ディスク
14
メモリ装置
リーダライタ装置
15
16
メモリカード
ネットワーク
10

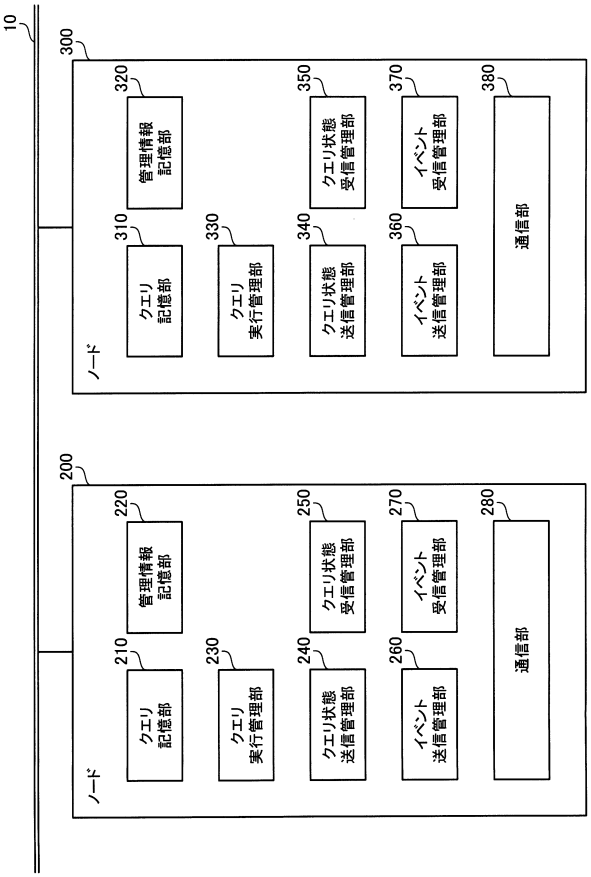
【図４】


ノード
100
E1
CEPエンジン
イベント入力
イベント出力
ノード
200
E2
CEPエンジン
イベント入力
イベント出力
割当て変更
ノード
300
E3
CEPエンジン
イベント入力
イベント出力
:クエリ
:割当て変更されたクエリ

【図５】



【図６】



【図７】

イベントの例

イベント種別	ストリーム名	内容
P	InputStream	1000W

X

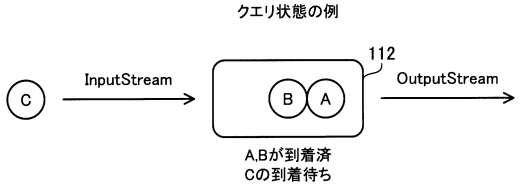
X1

【図８】

クエリの例 111

```
1 insert into OutputStream
2 select *
3 from pattern [ every A=InputStream -> B=InputStream -> C=InputStream ]
```

【図９】



【図１０】

121

配置表					
クエリ名	配置ノード名	状況	クエリ状態への参照	ロック	待ちイベント
Query1	ノード100	ノード200へ移動中	&Query1State	無	null
Query8	ノード200	稼働中	ノード内に無し	無	null
Query10	ノード300	稼働中	ノード内に無し	無	null
…	…	…	…	…	…

【図１１】

221

配置表					
クエリ名	配置ノード名	状況	クエリ状態への参照	ロック	待ちイベント
Query1	ノード100	ノード200へ移動中	ノード内に無し	無	α5,α6
Query8	ノード200	稼働中	&Query8State	無	null
Query10	ノード300	稼働中	ノード内に無し	無	null
…	…	…	…	…	…

【図１２】

321

配置表					
クエリ名	配置ノード名	状況	クエリ状態への参照	ロック	待ちイベント
Query1	ノード200	稼働中	ノード内に無し	無	null
Query8	ノード200	稼働中	ノード内に無し	無	null
Query10	ノード300	稼働中	&Query10State	無	null
…	…	…	…	…	…

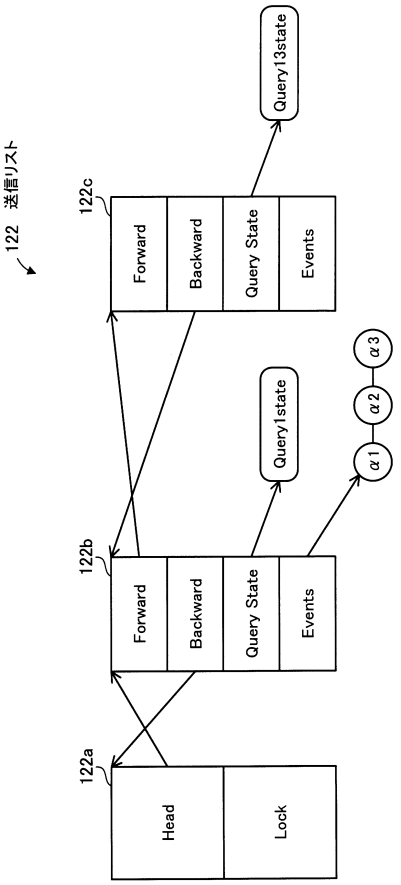
【図１３】

D

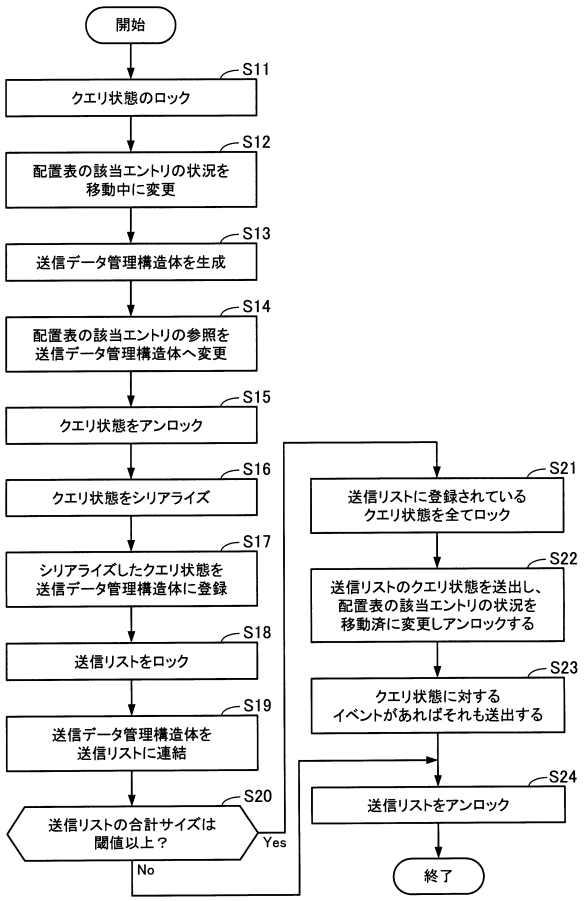
送信データ管理構造体の例

Forward	&SendBufStructure
Backward	&SendBufStructure
Query State	&QueryState
Events	&Events [ ]

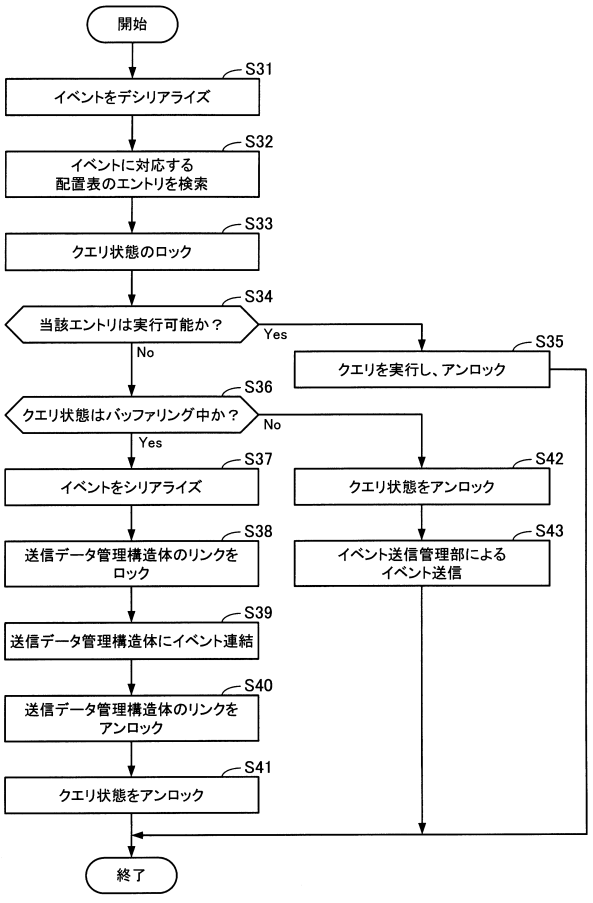
【図１４】



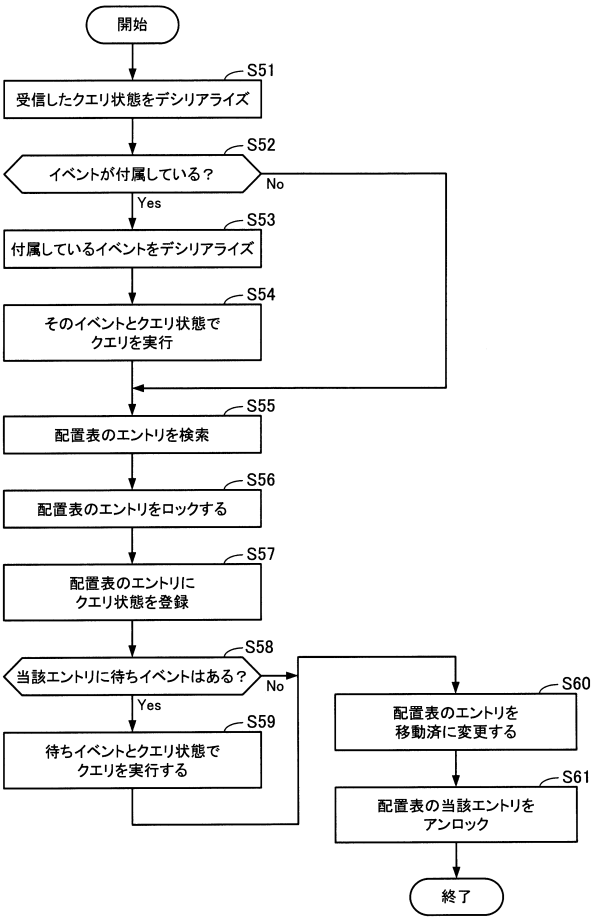
【図１５】



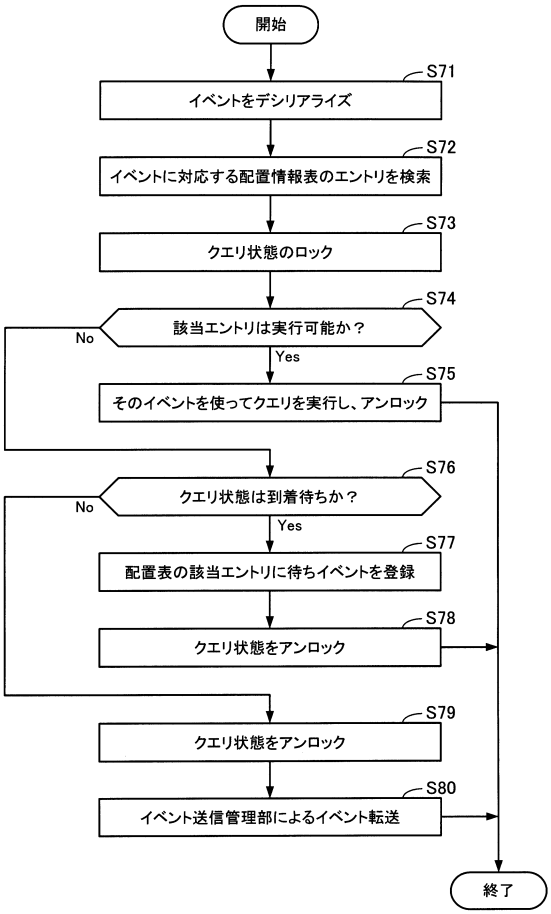
【図１６】



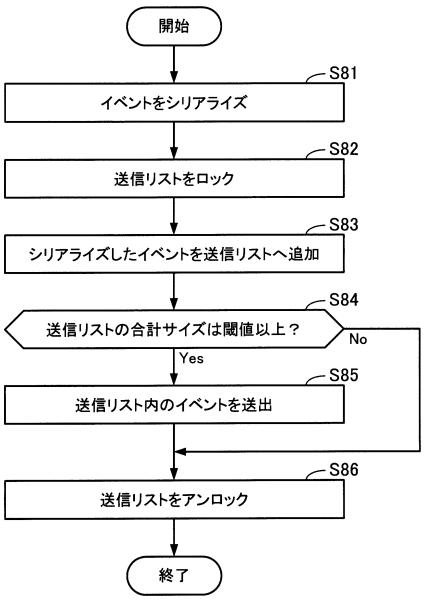
【図１７】



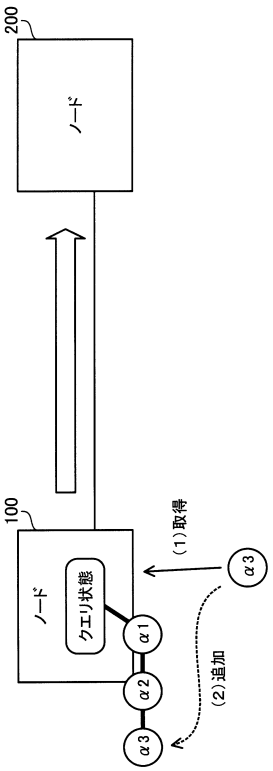
【図１８】



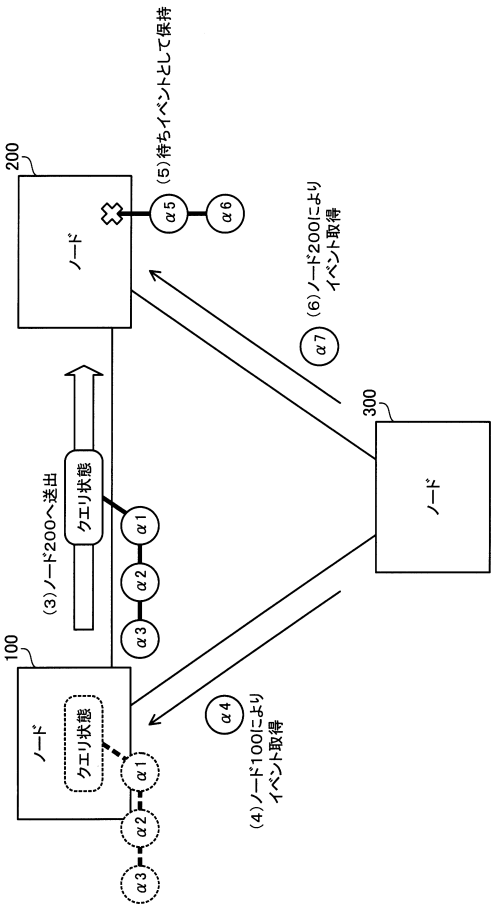
【図１９】



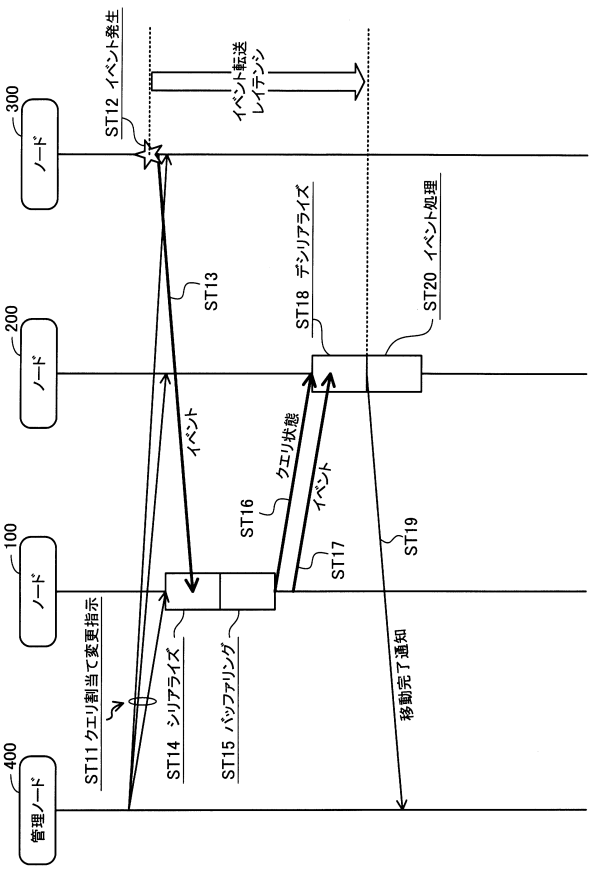
【図２０】



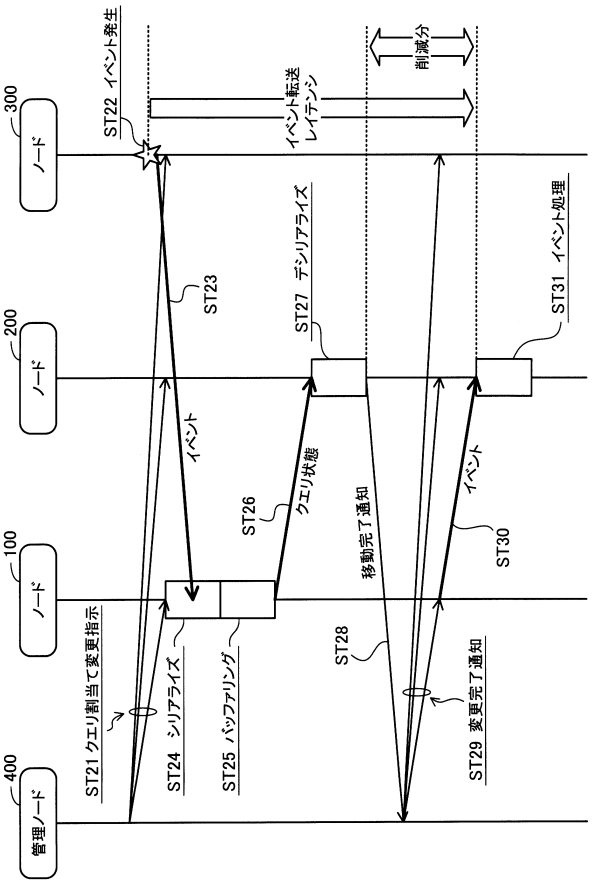
【図２１】



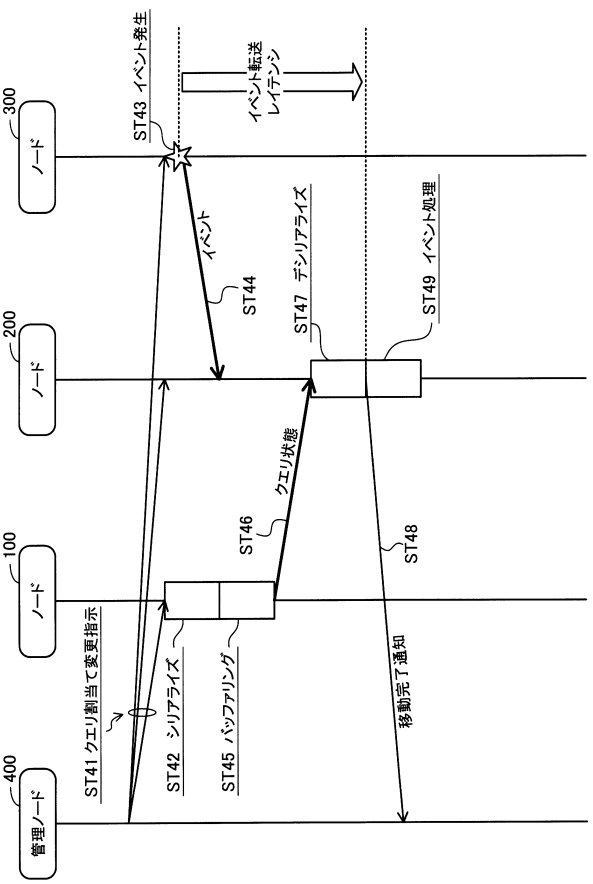
【図２２】



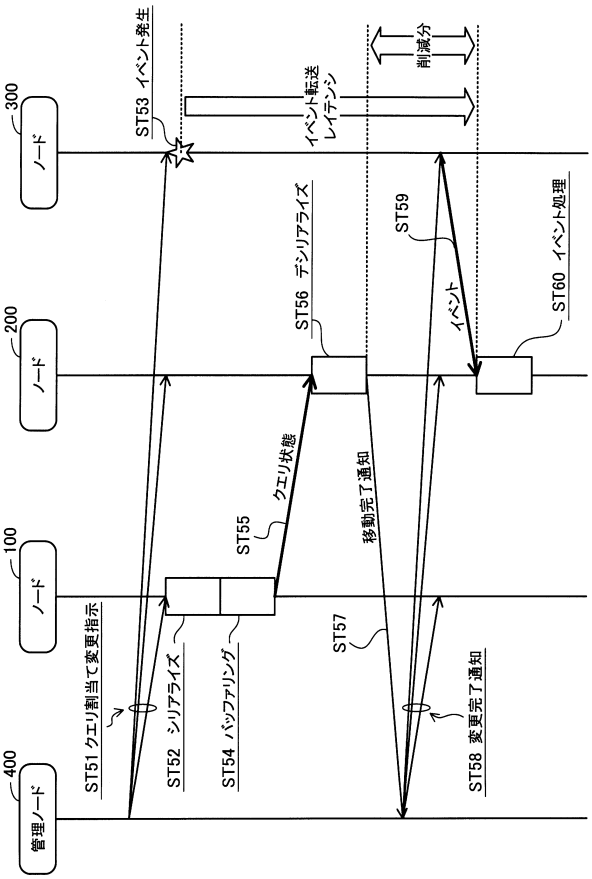
【図２３】



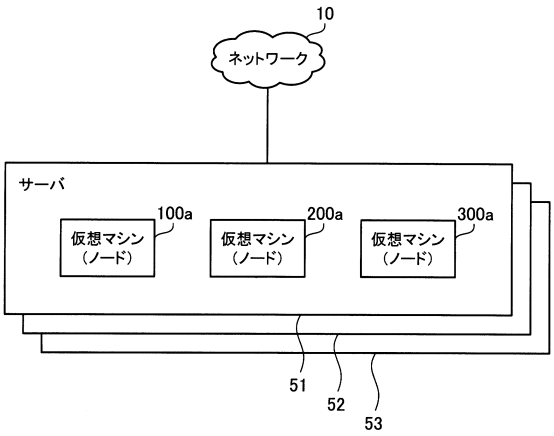
【図２４】



【図２５】


300
ノード
200
ノード
100
ノード
400
管理ノード
ST53 イベント発生
イベント転送
レイテンシ
削減分
ST59
ST56 デシリアライズ
イベント
ST60 イベント処理
ST55
クエリ状態
移動完了通知
ST57
ST51 クエリ割当て変更指示
ST52 シリアライズ
ST54 バッファリング
ST58 変更完了通知

【図２６】


10
ネットワーク
サーバ
100a
仮想マシン
（ノード）
200a
仮想マシン
（ノード）
300a
仮想マシン
（ノード）
51
52
53

フロントページの続き

(56)参考文献 特開２０１２－２４８１６９（ＪＰ，Ａ）
高橋 雅彦，データセンタ環境に適したＴＣＰ無切断プロセスマイグレーションの実現（２００４－ＯＳ－９６），情報処理学会研究報告，日本，社団法人情報処理学会，２００４年 6月１８日，第2004巻,第63号，pp.29-36

(58)調査した分野(Int.Cl.，ＤＢ名)
Ｇ０６Ｆ 9／５０