(19)日本国特許庁(JP)　(12)特　許　公　報(B2)　(11)特許番号
特許第6197692号
(P6197692)
(45)発行日　平成29年9月20日(2017.9.20)　(24)登録日　平成29年9月1日(2017.9.1)

(51)Int.Cl.　　F I
G06F 9/54 (2006.01)　　G06F 9/46 480B
H04L 12/70 (2013.01)　　H04L 12/70 D

請求項の数 3　(全 37 頁)

(21)出願番号　特願2014-36075(P2014-36075)
(22)出願日　平成26年2月26日(2014.2.26)
(65)公開番号　特開2015-162029(P2015-162029A)
(43)公開日　平成27年9月7日(2015.9.7)
審査請求日　平成28年8月4日(2016.8.4)

（出願人による申告）平成２５年度、総務省、「ネットワーク仮想化技術の研究開発」のうち、「仮想ネットワーク対応ノード技術」委託研究、産業技術力強化法第１９条の適用を受ける特許出願

(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　100113608
弁理士　平川　明
(74)代理人　100105407
弁理士　高田　大輔
(72)発明者　清水　翔
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　大桃　由紀雄

最終頁に続く

(54)【発明の名称】サーバ

(57)【特許請求の範囲】
【請求項１】
複数のコントローラがアプリケーションからの情報に基づいて複数のスイッチを制御するネットワークシステムに設けられるサーバであって、
スイッチ単位で設けられ、それぞれが自身に対応するスイッチに向けられた前記アプリケーションからの情報を格納する複数のキューと、
キューの指定情報と当該指定情報によって指定されたキューから読み出される情報を当該情報の送信先に該当するコントローラへ送信するための情報とを含む送信先情報を前記各コントローラから受信する受信部と、
前記複数のキューのそれぞれから読み出される情報を、前記送信先情報に従って前記各キューに対応するスイッチを制御する前記複数のコントローラの１つへ送信する送信部とを含むサーバ。
【請求項２】
前記受信部は、前記複数のコントローラの数の減少又は増加によって前記複数のスイッチの少なくとも１つを制御するコントローラが変更されたときに、当該変更された少なくとも１つのコントローラから前記送信先情報を受信し、
前記送信部は、前記少なくとも１つのコントローラから受信された前記送信先情報に従って、前記複数のスイッチの少なくとも１つに対応するキューから読み出される情報を前記少なくとも１つのコントローラへ送信する
請求項１に記載のサーバ。

【請求項３】
アプリケーションからの情報に基づいてスイッチを制御するとともに前記スイッチに関する情報であるスイッチ関連情報をアプリケーション向けに送信するコントローラを含むネットワークシステムに設けられるサーバであって、
アプリケーションによって提供される機能単位で設けられ、それぞれが自身に対応する機能と関連する前記スイッチ関連情報を格納する複数のキューと、
キューの指定情報と当該指定情報によって指定されたキューから読み出されるスイッチ関連情報を当該スイッチ関連情報の送信先に該当するアプリケーションへ送信するための情報とを含む送信先情報をアプリケーションから受信する受信部と、
前記複数のキューのそれぞれから読み出される前記スイッチ関連情報を、前記送信先情報に従って当該スイッチ関連情報に関連する機能を提供するアプリケーションへ送信する送信部と
を含むサーバ。
【発明の詳細な説明】
【技術分野】
【０００１】
本開示は、サーバに関する。
【背景技術】
【０００２】
近年、ネットワーク業界においてSoftware Defined Networking（ＳＤＮ）が注目されている。ＳＤＮでは、ネットワーク全体をコントローラと呼ばれるソフトウェアが集中的に制御、管理する。これによって、ネットワークのプログラマビリティを高め、制御の自動化を達成することを目的としている。
【０００３】
ＳＤＮでは、コントローラによるスイッチの集中制御モデルを採用している。すなわち、コントローラには、複数のスイッチが接続され、各スイッチの動作を制御プロトコルに従って制御する。このため、コントローラの性能が全体のボトルネックとなりやすい傾向がある。そこで、複数のコントローラを連携させることにより、例えば、物理構成は複数のサーバによる分散構成であるが論理的には集中制御となる分散コントローラを用いたネットワークシステム（分散コントローラシステム）によってスケーラビリティを向上させることが考えられている。
【０００４】
分散コントローラでは、スイッチの動作を制御するためのプロセスが、コントローラプロセスとアプリケーションプロセスとに分離され、両者がメッセージングシステムを介して情報をやりとりする。この種の分散コントローラは、コントローラプロセスとアプリケーションプロセスとがメッセージングシステムを介して疎結合される。このため、「疎結合型分散コントローラ」と呼ばれることがある。分散コントローラは、コントローラ及びアプリケーションのそれぞれを単独でスケールアウトさせることを可能とする。以下の説明において、コントローラプロセスを単にコントローラと表記し、アプリケーションプロセスを単にアプリケーションと表記することもある。
【０００５】
メッセージングシステムとしては、コントローラとアプリケーションとの間でやりとりされる情報を格納する複数のキューを備えるメッセージキューサーバ（ＭＱサーバ）が用いられる。ＭＱサーバにおいて、キューはプロセス毎に備えられる。上述したように、分散コントローラは、プロセスとしてコントローラとアプリケーションとを含む。このため、ＭＱサーバは、アプリケーション毎に作成されたキューと、コントローラ毎に作成されたキューとを含む。各キューには、対応するプロセス（アプリケーション又はコントローラ）を宛先とするメッセージが格納される。
【先行技術文献】
【特許文献】

【０００６】
【特許文献１】特開平７－２００４９４号公報
【発明の概要】
【発明が解決しようとする課題】
【０００７】
上述した分散コントローラ（関連技術）では、スケールアウト又はスケールインのため、或いは、障害の発生によって、アプリケーションの数やコントローラの数が増減することがある。また、分散コントローラでは、プロセス（コントローラ、アプリケーション）数の増加によって、システム全体の処理能力を向上させることが考えられている。このように、関連技術では、プロセス（コントローラ、アプリケーション）の数が増減することが想定されている。
【０００８】
しかしながら、関連技術では、上記したように、キューがプロセス単位で作成されている。このため、例えば、コントローラの数が増減すると、或るコントローラが自身に対応するキューに格納された情報だけでなく、自身以外のコントローラに対応するキューに格納された情報を得るという変則的な処理を行う状態となる。このように、関連技術では、コントローラの数の増減時に対応するために、キューからの情報の読み出し及びコントローラへの送信処理が複雑化する問題があった。
【０００９】
また、或る処理の負荷分散のために或る処理を行う複数のアプリケーションが存在する場合には、コントローラは、或る処理を実行可能なアプリケーションの数を把握して、上記複数のアプリケーションに対応する複数のキューに情報を分散配置する。このように、関連技術では、キューへの情報の格納にあたりキューからの情報の配信先の数を考慮するために、キューへの情報の格納処理が複雑化する問題があった。
【００１０】
以上のように、関連技術では、プロセス（コントローラ、アプリケーション）の増減に対応するために、キューからの情報の読み出し及び送信処理やキューへの情報の格納処理のようなキューに関わる処理が複雑になる問題があった。
【００１１】
本開示の目的は、コントローラ又はアプリケーションの増減を要因としてキューに関わる処理が複雑化することを抑制可能な技術を提供することにある。
【課題を解決するための手段】
【００１２】
本開示は、複数のコントローラがアプリケーションからの情報に基づいて複数のスイッチを制御するネットワークシステムに設けられるサーバであって、
スイッチ単位で設けられ、それぞれが自身に対応するスイッチに向けられた前記アプリケーションからの情報を格納する複数のキューと、
キューの指定情報と当該指定情報によって指定されたキューから読み出される情報を当該情報の送信先に該当するコントローラへ送信するための情報とを含む送信先情報を前記各コントローラから受信する受信部と、
前記複数のキューのそれぞれから読み出される情報を、前記送信先情報に従って前記各キューに対応するスイッチを制御する前記複数のコントローラの１つへ送信する送信部とを含むサーバである。
【発明の効果】
【００１３】
本開示によれば、キューに関わる処理がコントローラ又はアプリケーションの増減を要因として複雑化することを抑えることが可能となる。
【図面の簡単な説明】
【００１４】
【図１】図１は、関連技術における第１の問題の説明図であり、プロセス毎に作成された

キューと、コントローラと、スイッチとの関係を示す。
【図２】図２は、コントローラ＃１の障害発生時における様子を示す。
【図３】図３は、関連技術における第２の問題の説明図である。
【図４】図４は、複数のＭＱサーバを配置するときの例（関連技術）である。
【図５】図５は、図４に示した構成における問題点を示す。
【図６】図６は、分散コントローラが適用されるネットワークシステムの構成例を示す。
【図７】図７は、ＣＰＵがプログラムを実行することによって実現される分散コントローラ（疎結合型分散コントローラ）を用いたネットワークシステムを模式的に示す図である。
【図８】図８は、本実施形態に係るコントローラに対するキュー割り当て方法の説明図である。
【図９】図９は、本実施形態に係るコントローラに対するキュー割り当て方法の説明図である。
【図１０】図１０は、本実施形態に係るアプリケーションに対するキュー割り当て方法の説明図である。
【図１１】図１１は、本実施形態に係るアプリケーションに対するキュー割り当て方法の説明図である。
【図１２】図１２は、複数のＭＱサーバが用意された例を示す。
【図１３】図１３は、スイッチの構成例を示す。
【図１４】図１４は、コントローラの構成例を示す。
【図１５】図１５は、アプリケーションの構成例を示す。
【図１６】図１６は、レジストリの構成例を示す図である。
【図１７】図１７は、コントローラ情報格納部が有するテーブルのデータ構造例を示す。
【図１８】図１８は、スイッチ情報格納部が有するテーブルのデータ構造例を示す。
【図１９】図１９は、アプリケーション情報格納部が有するテーブルのデータ構造例を示す。
【図２０】図２０は、ＭＱサーバ情報格納部が有するテーブルのデータ構造例を示す。
【図２１】図２１は、ＭＱサーバの構成例を示す図である。
【図２２】図２２は、コントローラの起動時におけるコントローラの動作例を説明するフローチャートである。
【図２３】図２３は、コントローラの終了時における動作例を示すフローチャートである。
【図２４】図２４は、スイッチとの接続時におけるコントローラの動作例を示すフローチャートである。
【図２５】図２５は、スイッチとの切断時におけるコントローラの動作例を示すフローチャートである。
【図２６】図２６は、アプリケーションの起動時におけるアプリケーションの動作例を示すフローチャートである。
【図２７】図２７は、アプリケーションの終了時におけるアプリケーションの動作例を示すフローチャートである。
【図２８】図２８は、ＭＱサーバの起動時におけるＭＱサーバの動作例を示すフローチャートである。
【図２９】図２９は、ＭＱサーバの終了時におけるＭＱサーバの動作例を示すフローチャートである。
【図３０】図３０は、非同期イベント発生時におけるコントローラの動作例を示すフローチャートである。
【図３１】図３１は、ＭＱサーバからメッセージを受信したコントローラの動作例を示すフローチャートである。
【図３２】図３２は、ＭＱサーバからメッセージを受信したアプリケーションの動作例を示すフローチャートである。

【図３３】図３３は、アプリケーションが或るスイッチの制御を行う場合におけるアプリケーションの動作例を示すフローチャートである。
【図３４】図３４は、コントローラの数が増加した場合の動作例を示すフローチャートである。
【図３５】図３５は、コントローラが障害等によって減少した場合の動作例を示すフローチャートである。
【発明を実施するための形態】
【００１５】
以下、図面を参照して本発明の実施形態について説明する。実施形態の構成は例示であり、本発明は実施形態の構成に限定されない。
【００１６】
〔関連技術〕
本発明の実施形態を説明する前に、プロセス毎にキューが作成される分散コントローラを用いたネットワークシステム（関連技術）についての問題点を詳細に説明する。
【００１７】
＜＜第１の問題＞＞
第１の問題として、コントローラに障害が発生した際に生じる問題を説明する。分散コントローラシステム（分散コントローラ環境）では、各コントローラはネットワーク中の複数のスイッチの一部を制御し、全てのコントローラでネットワーク中の全てのスイッチの制御をカバーする。
【００１８】
すなわち、各スイッチにはマスタコントローラが存在し、通常時はマスタコントローラのみから制御を受ける。或るコントローラに障害が発生すると（コントローラの数が減少すると）、残りの正常なコントローラのうちの少なくとも１つが新たなマスタコントローラとなって或るコントローラが行っていた動作を継続する。
【００１９】
図１は、第１の問題の説明図であり、プロセス毎に作成されたキューと、コントローラと、スイッチとの関係を示す。図１では、スイッチ（ＳＷ）＃１，スイッチ（ＳＷ）＃２，スイッチ（ＳＷ）＃３のマスタコントローラとしてコントローラ（Controller）＃１が動作している。コントローラ＃２は、スイッチ（ＳＷ）＃４及びスイッチ（ＳＷ）＃５のマスタコントローラとして動作し、コントローラ＃３は、スイッチ（ＳＷ）＃６のマスタコントローラとして動作する。
【００２０】
ＭＱサーバは、コントローラプロセス（コントローラ＃１～＃３）毎に作成されたキューＱ１，キューＱ２，キューＱ３を有している。各キューＱ１～Ｑ３は、対応するコントローラがマスタコントローラとして制御しているスイッチを制御するためのメッセージを一時的に格納（蓄積）する。コントローラ＃１に対応するキューＱ１は、ＳＷ＃１～ＳＷ＃３向けのメッセージを格納する。コントローラ＃２に対応するキューＱ２は、ＳＷ＃４及びＳＷ＃５向けのメッセージを格納する。コントローラ＃３に対応するキューＱ３は、ＳＷ＃６向けのメッセージを格納する。
【００２１】
図１に示す状態で、コントローラ＃１に障害が発生した場合を考える。図２は、コントローラ＃１の障害発生時における様子を示す。この場合、ＳＷ＃１，ＳＷ＃２，ＳＷ＃３のそれぞれは、残りの正常なコントローラ＃２及びコントローラ＃３の一方を新たなマスタコントローラとして、マスタコントローラの配下に入る。例えば、ＳＷ＃１がコントローラ＃２の配下となり、ＳＷ＃２及びＳＷ＃３がコントローラ＃３の配下となったケースを仮定する。
【００２２】
図２において、キューＱ１には、コントローラ＃１が未処理のＳＷ＃１，ＳＷ＃２，ＳＷ＃３に関するメッセージが未処理の状態で格納されている。このため、コントローラ＃

２及びコントローラ＃３のそれぞれは、キューＱ１から自身が新たにマスタコントローラとして制御するスイッチに関するメッセージのみを選択的にキューＱ１から読み出す。

【００２３】

すなわち、コントローラ＃２は、キューＱ１からＳＷ＃１に関するメッセージを読み出してコントローラ＃２に送信することをＭＱサーバに依頼する。コントローラ＃３は、キューＱ１からＳＷ＃２及びＳＷ＃３に関するメッセージを読み出してコントローラ＃３に送信することをＭＱサーバに依頼する。

【００２４】

このように、通常時（障害発生前）では、各コントローラ＃１～＃３は、コントローラ自身に対応するキューＱ１～Ｑ３の何れかのみからメッセージの読み出すための動作（ＭＱサーバへの読み出し及び送信依頼）を行っている。これに対し、障害発生時（障害発生後）では、コントローラ＃２及びコントローラ＃３のそれぞれは、対応するキューＱ２又はＱ３からメッセージを読み出す動作だけでなく、他のコントローラ＃１（障害が発生したコントローラ）に対応するキューＱ１からメッセージを読み出す動作を行う。このように、障害が発生したコントローラの動作を引き継ぐコントローラの動作が、障害発生前の動作に比べて変則的な（複雑な）動作となっていた。

【００２５】

これに対し、既存のＭＱサーバの仕様では、既にキューに格納されたメッセージをそのメッセージ内容（例：メッセージの宛先）に応じて新たなマスタコントローラに送信する動作をサポートしていなかった。

【００２６】

＜＜第２の問題＞＞

第２の問題として、アプリケーション増加時におけるロードバランスへの対処に係る問題について説明する。ＭＱサーバがプロセス毎に作成されたキューを有する場合において、アプリケーションがスケールアウトのために複数のプロセスで動作している場合を考える。アプリケーションは、スケールアウトのために、同一の機能を提供する複数のプロセスから形成される場合がある。図３は、関連技術における第２の問題の説明図である。図３では、負荷分散（ロードバランス）による処理能力向上のために同一機能を提供する３つのアプリケーションプロセス（Ａｐｐ.＃１，Ａｐｐ.＃２，及びＡｐｐ.＃３，）が例示されている。

【００２７】

ＭＱサーバは、アプリケーション＃１（Ａｐｐ.＃１）に対応するキューＱＡ１と、アプリケーション＃２（Ａｐｐ.＃２）に対応するキューＱＡ２と、アプリケーション＃３（Ａｐｐ.＃３）に対応するキューＱＡ３とを有している。

【００２８】

処理能力向上のために同一機能を複数のプロセス（複数のアプリケーション＃１～＃３）で処理していることに鑑みると、各アプリケーション＃１～＃３に偏りなくメッセージを配信することが試行される。この場合、アプリケーション側にメッセージを送信するコントローラ（図３ではコントローラ＃１～＃３を例示）が、アプリケーションプロセス数を意識して、メッセージをキューＱＡ１，ＱＡ２，ＱＡ３の何れかに配置（格納）する。

【００２９】

さらに、スケールアウトのために、アプリケーションプロセス数が１つ増加した（各アプリケーション＃１～＃３と同一機能を有する図示しないアプリケーション＃４（Ａｐｐ.＃４）が追加された）場合を考える。この場合、ＭＱサーバには、アプリケーション＃４に対応するキューＱＡ４（図示せず）が作成される。すると、各コントローラ＃１～＃３は、ＭＱサーバにキューＱＡ４が作成された（アプリケーションプロセスが４つに増えた）ことを意識して、キューＱＡ１～ＱＡ４の何れかにメッセージを分散配置する。このように、各コントローラ＃１～＃３は、アプリケーションの数を含むアプリケーションの状態を意識してメッセージを複数のキューに分散配置する。従って、動作が複雑となる。

【００３０】

＜＜第３の問題＞＞

関連技術に係る分散コントローラシステムは、メッセージングシステムとして、単一のＭＱサーバを有する構成を採用している。この場合、メッセージングシステム自体のスケーラビリティがボトルネックになり分散コントローラシステムのスケーラビリティが制約される。

【００３１】

このため、メッセージングシステムのスケーラビリティを高めて、スケーラブルな分散コントローラシステムを実現することが考えられる。換言すれば、単一のＭＱサーバではなく、複数のＭＱサーバを用いてスケーラビリティが高められたメッセージングシステムを形成することが考えられる。

【００３２】

ここで、プロセス毎にキューを作成する方法が適用された分散コントローラにおいて、複数のＭＱサーバを用いる構成を考える。最も単純な方法としては、データベース（ＤＢ）技術において“シャーディング”と呼ばれているデータ分割手法に準じた方法が考えられる。具体的には、図４に示すように、１つのＭＱサーバ内に存在していたプロセス毎のキューを分割して複数のＭＱサーバに配置する。

【００３３】

図４に示す例では、ＭＱサーバ＃１は、コントローラ（ＣＮＴ）＃１～＃４並びにアプリケーション（ＡＰＰ）＃１及び＃２にそれぞれ対応する複数のキューを有している。これに対し、複数のキューの分割によって、コントローラ＃１及び＃２並びにアプリケーション＃１のキューがＭＱサーバ＃１に配置されている。一方、ＭＱサーバ＃２には、コントローラ＃３及び＃４並びにアプリケーション＃２のキューが配置されている。当該手法では、１つのＭＱサーバが有していたキューが、複数のＭＱサーバ（ＭＱサーバ群）の何れか１つに配置される。

【００３４】

図５は、図４に示した構成における問題点を示す。上記したキューの分割配置の手法では、メッセージの宛先が全プロセスに均等に分散している場合には、各ＭＱサーバに対する負荷が均等となることで、各ＭＱサーバを効率的に動作させることができる。

【００３５】

これに対し、図５に示すように、コントローラ＃１～＃４からアプリケーション＃１向けのメッセージが集中的に発行される場合では、ＭＱサーバ＃１に負荷が集中し、ＭＱサーバ＃２への負荷がない非効率な状態となる。換言すれば、上記した分割配置の手法では、或るプロセスに対応するキューがＭＱサーバ群の一つにしか配置されないため、メッセージの宛先に偏りが生じると、複数のＭＱサーバを設けた意義（負荷分散による効率化）が損なわれる虞があった。

【００３６】

以下に説明する実施形態に係る分散コントローラを用いたネットワークシステムは、上述した第１～第３の問題を解決する。

【００３７】

〔実施形態〕

以下、実施形態に係る分散コントローラを用いたネットワークシステムについて説明する。

【００３８】

＜ネットワークシステムの構成例＞

図６は、分散コントローラが適用されるネットワークシステムの構成例を示す。図６において、ネットワークシステムは、ネットワーク（ＮＷ）１を介して接続された複数のスイッチ２と、単数又は複数のサーバ３とを含む。

【００３９】

ネットワーク１は、例えば、インターネットやイントラネットに代表されるInternet Protocol （ＩＰ）ネットワーク、或いはLocal Area Network(ＬＡＮ)である。スイッチ２

は、例えば、ルータやレイヤ３スイッチである。スイッチ２は、レイヤ２スイッチやスイッチングＨＵＢを含み得る。スイッチ２は、物理的なスイッチ（実スイッチ）であっても、コンピュータ上で仮想的に生成される仮想スイッチであっても良い。

【００４０】

サーバ３は、プロセッサ及びメモリを含むコンピュータの一例である。サーバ３として、専用のサーバマシン，或いは、パーソナルコンピュータ（ＰＣ）やワークステーション（ＷＳ）のような汎用のコンピュータを適用することができる。

【００４１】

サーバ３は、バスＢを介して相互に接続されたCentral Processing Unit（ＣＰＵ）４と、メモリ５と、通信インタフェース（通信ＩＦ）６とを含み、通信ＩＦ６が物理回線を介してネットワーク１に接続されている。メモリ４は、不揮発性記憶媒体と、揮発性記憶媒体とを含む。ＣＰＵ４は、プロセッサ（制御装置）の一例である。

【００４２】

不揮発性記憶媒体は、例えば、Read Only Memory（ＲＯＭ），ハードディスク，Solid State Drive（ＳＳＤ），ＥＥＰＲＯＭ，フラッシュメモリなどであり、ＣＰＵ４によって実行されるプログラムや、プログラムの実行に際して使用されるデータを記憶する。揮発性記憶媒体は、例えばRandom Access Memory（ＲＡＭ）であり、ＣＰＵ４の作業領域として使用される。メモリ５は、「記憶装置」、「記憶媒体」の一例である。

【００４３】

通信ＩＦ６は、通信に係る信号変換、プロトコル変換を司る装置であり、例えば、ネットワークカード、或いはＬＡＮカードと呼ばれる通信機器が適用される。ＣＰＵ４は、メモリ５に記憶されたプログラムをロードして実行することによって、分散コントローラとして動作し、各スイッチ２の動作を制御する。

【００４４】

図７は、ＣＰＵ４がプログラムを実行することによって実現される分散コントローラ（疎結合型分散コントローラ）を用いたネットワークシステムを模式的に示す図である。分散コントローラを用いたネットワークシステムは、複数のスイッチ２と、複数のコントローラ１１と、単数又は複数のアプリケーション１２と、コーディネーションシステム１３、メッセージングシステム１５とを含む。

【００４５】

一つのコントローラ１１は、ネットワーク上の複数のスイッチ２の一部分の制御を行い、全てのコントローラプロセスでネットワーク上の全てのスイッチ２の制御をカバーする。アプリケーション１２（アプリケーションプロセス）は、コントローラ１１に対し、スイッチ２の制御に係る情報を与える。

【００４６】

コーディネーションシステム１３とメッセージングシステム１５は、アプリケーション１２とコントローラ１１との間における情報のやりとりを仲介するシステムであり、役割によってシステムが分けられている。

【００４７】

コーディネーションシステム１３は、主にアプリケーション１２やコントローラ１１のメンバーシップや状態などの分散コントローラ全体で共有すべき情報を格納する役割を担っている。コーディネーションシステム１３は、レジストリ１４を含み、レジストリ１４は、コントローラ１１，アプリケーション１２などに関する情報が格納され、他のプロセスは、レジストリ１４に格納された情報を参照することができる。

【００４８】

メッセージングシステム１５は、コントローラ１１とアプリケーション１２との間の通信（情報の送受信）を仲介するシステムである。分散コントローラでは、各コントローラ１１と各アプリケーション１２とが互いにメッセージ（情報の一例）を送り合うことでシステム全体の動作が行われる。

【００４９】

コントローラ１１は、自身がマスタコントローラとして割り当てられたスイッチ２（配下のスイッチ）を制御する。配下のスイッチ２は、イベント（例えば、Packet Inイベント）を発生させると、マスタコントローラに該当するコントローラ１１へイベントの発生を通知する。イベントの発生の通知を受信したコントローラ１１は、アプリケーション１２へ向けて、イベントの発生を示すメッセージをメッセージングシステム１５を介して通知する。イベント発生を示すメッセージは、「スイッチ関連情報」の一例である。

【００５０】

アプリケーション１２は、メッセージングシステム１５を介してイベントの発生を示すメッセージを受信し、アプリケーション１２自身の制御ロジックに従って、イベントに対する対応（処理）を決定する。この結果、アプリケーション１２は、該当のスイッチ２を制御する場合には、指定したスイッチ２を制御するためのメッセージをメッセージングシステム１５を介してコントローラ１１に送信する。

【００５１】

メッセージングシステム１５は、コントローラ１１とアプリケーション１２との間でやりとりされる情報が格納される複数のキューを備えたメッセージキューサーバ（ＭＱ）サーバ１６を含む。ＭＱサーバ１６は、スケールアウトによってその個数を増やすことができる。図７の例では、２つのＭＱサーバ１６が例示されている。ＭＱサーバ１６は、「サーバ」の一例である。

【００５２】

コントローラ１１，アプリケーション１２，コーディネーションシステム１３（レジストリ１４），メッセージングシステム１５（ＭＱサーバ１６）は、ＣＰＵ４がプログラムを実行することによって生じるエンティティである。コントローラ１１，アプリケーション１２は、例えば、オブジェクト指向プログラミングにおけるインスタンスである。

【００５３】

図７に示すような分散コントローラは、１つのサーバ３（物理サーバ：コンピュータ）によって生成されるようにしても良く、図６に示すような、２以上の物理サーバ（コンピュータ）間の連携処理によって生成されるようにしても良い。以下の説明は、説明を簡単にするため、各コントローラ１１，各アプリケーション１２，コーディネーションシステム１３（レジストリ１４），メッセージングシステム１５（ＭＱサーバ１６）が、単一のサーバ３上で生成されている場合を想定している。

【００５４】

＜第１～第３の問題に対する解決方法＞

次に、関連技術における第１～第３の問題を解決する方法（処理）について説明する。本実施形態では、第１及び第２の問題を解決するために、プロセス（コントローラ１１，アプリケーション１２）に対するキューの割り当て方法によって、動作の単純化を図る。以下、コントローラ１１及びアプリケーション１２のそれぞれに対するキューの割り当て方法について説明する。

【００５５】

＜＜コントローラに対するキュー割り当て＞＞

本実施形態では、スイッチ２毎にキューを作成し、各コントローラ１１は自身がマスタコントローラとして制御するスイッチ２用のキューを読み込む（ＭＱサーバ１６から対応するスイッチ２の情報を受信する）。

【００５６】

図８及び図９は、本実施形態に係るコントローラに対するキュー割り当て方法の説明図である。図８に示す例では、スイッチ２として、スイッチ（ＳＷ）＃１～＃６が示されており、コントローラ１１として、３つのコントローラ＃１～＃３が示されている。コントローラ（ＣＮＴ）＃１は、スイッチ（ＳＷ）＃１及びスイッチ＃２のマスタコントローラである。また、コントローラ（ＣＮＴ）＃２は、スイッチ（ＳＷ）＃３，スイッチ＃４及びスイッチ＃５のマスタコントローラである。また、コントローラ（ＣＮＴ）＃３は、スイッチ（ＳＷ）＃６のマスタコントローラである。

【００５７】
ＭＱサーバ１６は、スイッチ毎にキューＱを作成する。すなわち、ＭＱサーバ１６には、スイッチ＃１～＃６に対応する複数のキューＱが設けられている。各コントローラ＃１～＃３は、マスタコントローラとして制御するスイッチ２に対応するキューＱを読み込む（ＭＱサーバ１６からキューＱ内の情報（メッセージ）を受信する）。
【００５８】
コントローラ＃１，コントローラ＃２，コントローラ＃３のそれぞれは、自身の配下のスイッチ２に対応するキューＱから読み出されるメッセージが自身に送信されるようにするために、ＭＱサーバ１６にキューの指定を含む送信先情報を送る。送信先情報は、指定キューを示す情報であるキュー指定情報と、メッセージを所定の送信先（コントローラ１１）へ送るための情報である送信先指定情報（ネットワークアドレスなど）を含む。
【００５９】
ＭＱサーバ１６は、各コントローラ＃１～＃３からの送信先情報に基づいて、各キューＱの先頭から読み出されるメッセージを対応するコントローラ１１へ送信するための設定（コネクション設定）を行う。これによって、各キューＱからメッセージが読み出される毎に、当該メッセージの送信先の判定が行われることなく、予め設けられたコネクションを用いて、メッセージの転送が行われる。
【００６０】
このようにして、コントローラ＃１は、スイッチ＃１及び＃２に対応するキューＱに格納されたメッセージをＭＱサーバ１６から受信する。コントローラ＃２は、スイッチ＃３～＃５に対応するキューＱに格納されたメッセージをＭＱサーバ１６から受信する。コントローラ＃３は、スイッチ＃６に対応するキューＱに格納されたメッセージをＭＱサーバ１６から受信する。
【００６１】
なお、上述した事前のコネクション設定は必須の要件ではなく、メッセージの読み出し毎に送信先情報を参照して送信先判定を行う場合もあり得る。
【００６２】
図９は、図８に示した状態において、コントローラ＃３に障害が発生しコントローラ１１の数がコントローラ＃１及び＃２の２つに減少した場合の動作を示す。図９は、コントローラ＃３の障害によって、スイッチ＃６のマスタコントローラがコントローラ＃２に変更された例を示す。換言すれば、スイッチ＃６とコントローラ＃３との対応関係が、コントローラ＃３の障害（コントローラの減少）によって、スイッチ＃６とコントローラ＃２との対応関係に変更されている。
【００６３】
コントローラ＃２は、スイッチ＃６のマスタコントローラになったとき、スイッチ＃６に対応するキューＱをコントローラ＃２へ送信することを指示する送信先情報をＭＱサーバ１６に与え、ＭＱサーバ１６は、送信先情報に従ってキューＱ（スイッチ＃６）から読み出されるメッセージの送信先をコントローラ＃２に変更する。これによって、コントローラ＃２は、スイッチ＃６に対応するキューＱに格納されたメッセージ（情報）を読み込む（ＭＱサーバ１６から送信される情報を受信する）状態となる。
【００６４】
このように、各コントローラ＃１～＃３は、自身がマスタとなっているスイッチ２に対応するキューを指定した送信先情報をＭＱサーバ１６に送り、ＭＱサーバ１６に指定キューから読み出される情報を自身に送ることを依頼する。ＭＱサーバ１６は送信先情報を用いた依頼に応じて、指定キューからのメッセージを対応するコントローラに送る設定を行う。
【００６５】
コントローラ＃３の障害によって、コントローラ＃２がスイッチ＃６のマスタコントローラとなったときには、コントローラ＃２は、スイッチ＃６に対応するキューＱから読み出される情報が自身に送られるようにするための送信先情報をＭＱサーバ１６に送信し、

ＭＱサーバ１６が、該当キューＱに係る送信先を変更する設定を行う。

【００６６】

このように、本実施形態では、コントローラ１１は、自身が受信を所望する（自身がマスタとなっているスイッチ向けの）メッセージを格納するキューの指定と、送信先（コントローラ自身）の指定とを含む送信先情報を送る処理を行う。ＭＱサーバ１６は、送信先情報に基づき、指定されたキューに格納されたメッセージを指定された送信先に送る処理を行う。

【００６７】

上記のような、コントローラ１１及びＭＱサーバ１６の処理は、コントローラ１１の障害の前後において変わらない。従って、コントローラ１１及びＭＱサーバ１６の動作が単純化される。このように、本実施形態によれば、関連技術のようなキューＱからの情報の読み出し及び送信処理の複雑化が抑止される。

【００６８】

なお、図８及び図９は、コントローラ１１の数が減少した場合について説明したが、コントローラ１１が増加した場合も同様の動作となる。すなわち、コントローラ１１の動作の前後において、キューに関わる処理内容に変化はない。従って、処理の簡素化（複雑化の抑止）が図られる。

【００６９】

＜＜アプリケーションのキュー割り当て＞＞

本実施形態では、アプリケーション１２の機能（機能グループ）単位でキューを作成し、各アプリケーション１２は自身が属する機能グループに対応するキューを読み込む。図１０及び図１１は、本実施形態に係るアプリケーションに対するキュー割り当て方法の説明図である。

【００７０】

図１０に示す例では、アプリケーション１２の機能グループ（アプリケーション機能）の例として、パス計算，トポロジ発見，ＱｏＳ制御が定義されており、パス計算を行うアプリケーションプロセス群として、複数のアプリケーション１２（アプリケーション（ＡＰＰ）＃１～＃３）が設けられている例を示す。

【００７１】

この場合、ＭＱサーバ１６には、パス計算，トポロジ発見，ＱｏＳ制御のそれぞれに対応するキューＱＡが設けられる。各コントローラ１１（図１０ではコントローラ（ＣＮＴ）＃１～＃３を例示）は、書き込み先のキューの指定を含むメッセージをＭＱサーバ１６に送る。ＭＱサーバ１６は、指定されたキューにメッセージを書き込む。

【００７２】

各アプリケーション＃１～＃３は、上記したコントローラ１１と同様に、ＭＱサーバ１６に対して、キュー指定情報及び送信先指定情報を含む送信先情報を送る。ＭＱサーバ１６は、送信先情報を用いて指定されたキューＱＡから読み出される情報を指定された送信先へ送る設定を行う。このように、各キューＱＡから読み出される情報の送信先の設定方法は、コントローラ１１とアプリケーションとで同じである。

【００７３】

但し、図１０に示すアプリケーション＃１～＃３は、それぞれ同一の機能を提供する（同一の処理を行う）。このため、ＭＱサーバ１６は、パス計算に対応するキューＱＡから読み出した情報（メッセージ）を、所定のルールに従ってアプリケーション＃１～＃３の何れかに配信する。例えば、ＭＱサーバ１６は、ラウンドロビン方式や、負荷が小さいアプリケーションにメッセージを配信するといった様々なルールに従って、キューＱＡから読み出したメッセージを何れかのアプリケーション１２に分散的に配信する。

【００７４】

図１１は、図１０に示す状態からアプリケーションプロセス数（アプリケーション１２の数）が減少した場合の動作を示す。図１１に示す例では、アプリケーション＃３の消滅によって、アプリケーション１２の数が減少した例を示している。この場合、ＭＱサーバ

１６とアプリケーション＃３とのセッションが切断されるので、アプリケーション＃３へのキューＱＡ（パス計算）に格納された情報（メッセージ）の配信が行われなくなるだけである。残りのアプリケーション＃１及び＃２には、送信先情報に従った送信が行われる。

【００７５】

したがって、コントローラ１１（コントローラ＃１～＃３を例示）側の処理を見れば、アプリケーション＃３の消滅の前後（アプリケーション１２の数の減少の前後）において、処理に変化はない、すなわち、アプリケーション１２の減少に関わらず、各コントローラ＃１～＃３から送信されるパス計算に係るメッセージがパス計算に対応するキューＱＡに格納される処理が行われる。このように、本実施形態では、コントローラ１１は、アプリケーションの数の変化を把握しなくて良い。従って、キューに関わる処理（キューへの書き込み処理）が簡素化される（複雑化が抑止される）。

【００７６】

なお、図１０及び図１１に示した例では、アプリケーション１２の減少時の動作を説明したが、アプリケーション１２が増加しても、コントローラ１１（コントローラ＃１～＃３）からのパス計算に係るメッセージがキューＱＡに格納されるための動作（処理）が変化することはない。ＭＱサーバ１６が、追加（増加）に係るアプリケーション１２（図示せず）からの送信先情報を受けて、該当するキューＱＡ（パス計算）から読み出したメッセージを選択的に追加に係るアプリケーション１２へ配信するようになるだけである。

【００７７】

＜複数ＭＱサーバへのキューの割り当て＞

本実施形態では、第３の問題を解決するために、ＭＱサーバ１６間で担当するキューを分割するのではなく、全てのＭＱサーバ１６で同一のキュー構成を備える。すなわち、本実施形態では、キュー構成が同一の複数のＭＱサーバ１６が用意される。換言すれば、本実施形態では、複数のＭＱサーバ１６のそれぞれに、或るプロセス（コントローラ１１、アプリケーション１２）に提供されるメッセージを格納するキューが存在することになる。

【００７８】

図１２は、複数のＭＱサーバ１６が用意された例を示す。図１２に示す例では、コントローラ１１として、コントローラ＃１及び＃２が示されており、アプリケーション１２として、アプリケーション＃１及び＃２が示されている。スイッチ２として複数のスイッチ＃１～＃４が示されている。

【００７９】

スイッチ＃１及びスイッチ＃２のマスタコントローラはコントローラ＃１であり、スイッチ＃３及びスイッチ＃４のマスタコントローラはコントローラ＃２である。ＭＱサーバ１６として、同一のキュー構成を有する２つのＭＱサーバ＃１及びＭＱサーバ＃２が用意されている。各ＭＱサーバ１６には、各スイッチ＃１～＃４に対応するキューＱと、アプリケーション１２によって提供される機能グループ“パス計算”に対応するキューＱＡとが設けられている。

【００８０】

同一のキューＱ（ＱＡ）を有する２つのＭＱサーバ＃１及び＃２が設けられているため、情報（メッセージ）の書き込み側（コントローラ１１，アプリケーション１２）は、ＭＱサーバ＃１及び＃２のうちの一方を選択してメッセージをキューに書き込む（メッセージを送る）ことができる。

【００８１】

コントローラ１１及びアプリケーション１２は、ＭＱサーバ＃１及び＃２の双方に存在するキューＱ（キューＱＡ）から、並列的にメッセージを受け取ることができる。例えば、コントローラ＃１に注目すると、コントローラ＃１は、ＭＱサーバ＃１とＭＱサーバ＃２のキューＱに存在するスイッチ＃１，スイッチ＃２用のメッセージを並列的に受け取ることができる。

【００８２】
例として、スイッチ＃１用のメッセージは、アプリケーション＃１によってＭＱサーバ＃１及びＭＱサーバ＃２の一方に書き込まれ、スイッチ＃２用のメッセージは、アプリケーション＃１によってＭＱサーバ＃１及びＭＱサーバ＃２の一方に書き込まれる。
【００８３】
また、アプリケーション＃１に注目すると、アプリケーション＃１は、ＭＱサーバ＃１及びＭＱサーバ＃２の一方に存在するパス計算用のキューＱＡに格納されたメッセージを並列的に受け取ることができる。
【００８４】
上記動作によれば、情報（メッセージ）の書き込み側（送信側：コントローラ１１，アプリケーション１２）は、メッセージの送信先（書き込み先のキュー）を複数のＭＱサーバ１６からメッセージ毎に選択することができる。このため、適度に書き込み先のＭＱサーバ１６が分散するように、ＭＱサーバ１６の選択を行うことで、ＭＱサーバ１６が増えた分だけ書き込みメッセージ処理数を増加させることができる。
【００８５】
また、メッセージの読み出し側（受信側：コントローラ１１，アプリケーション１２）は、複数のＭＱサーバ１６からメッセージを並列的に受け取る。このため、読み出し側でも、ＭＱサーバ１６が増えた分だけ読み込みメッセージ処理数を増加させることができる。単純にキューを分割する方式（図４）と比較すると、特定のプロセス宛てのメッセージ数が多いという偏った状態であっても、本実施形態に係るキュー構成では全てのＭＱサーバ１６を有効に利用することができる。
【００８６】
このように、本実施形態に係る分散コントローラを用いたネットワークシステムによれば、プロセス増減時における動作の単純化（複雑化の抑止）を図ることができるとともに、複数のＭＱサーバ１６を用いることによるスケーラビリティ向上を図ることができる。
【００８７】
＜構成要素の詳細な説明＞
次に、上記した作用効果を奏する分散コントローラを形成するスイッチ２、コントローラ１１、アプリケーション１２、レジストリ１４、ＭＱサーバ１６の詳細について説明する。これまでの説明で理解されるように、スイッチ２，コントローラ１１，アプリケーション１２，ＭＱサーバ１６は、それぞれ複数個存在し得る。
【００８８】
＜＜スイッチ＞＞
スイッチ２は、所定の制御プロトコル（例えば、OpenFlow）に従って、マスタコントローラに該当するコントローラ１１により制御される。スイッチ２は、コントローラ１１と情報の交換を行う。すなわち、スイッチ２は、コントローラ１１から制御メッセージを受信し、制御メッセージに応じた動作を行う。一方、スイッチ２は、スイッチ２内で発生したイベントをコントローラに通知すべく、イベントを示す情報をコントローラ１１に送信する。
【００８９】
図１３は、スイッチ２の構成例を示す。スイッチ２は、制御プロトコル処理部２１と、パケット転送部２２とを含む。制御プロトコル処理部２１は、コントローラ１１から受信された制御メッセージに従った処理を行う。パケット転送部２２は、制御メッセージに従った処理の一環として、自身が受信したパケットを所定の方路（ポート）へ送出することによって、パケットの転送処理を行う。
【００９０】
＜＜コントローラ＞＞
コントローラ１１は、制御プロトコルを用いてスイッチ２を制御しつつ、スイッチ２で発生したイベントをアプリケーション１２にＭＱサーバ１６を介して通知する、また、コントローラ１１は、アプリケーション１２が送信したメッセージに従って、スイッチ２に

対する制御メッセージを発行する。また、コントローラ１１は、ＭＱサーバ１６とメッセージのやりとりを行う。

【００９１】

図１４は、コントローラの構成例を示す。コントローラ１１は、制御プロトコル処理部３１と、レジストリ関連部３２と、ＭＱ関連部３３とを含む。レジストリ関連部３２は、レジストリ１４と連携した処理を行う。レジストリ関連部３２は、レジストリ接続部３２１と、コントローラ情報取得・更新部３２２と、スイッチ情報取得・更新部３２３と、ＭＱサーバ情報取得部３２４と、アプリケーション情報取得部３２５とを含む。

【００９２】

レジストリ接続部３２１は、レジストリ１４に格納（記憶）された情報を参照・取得するためにコントローラ１１をレジストリ１４と接続する処理を司る。コントローラ情報取得・更新部３２２は、レジストリ１４に記憶されたコントローラ１１に係る情報（コントローラ情報）を取得したり、更新したりする処理を司る。

【００９３】

スイッチ情報取得・更新部３２３は、レジストリ１４に記憶されたスイッチ２に係る情報（スイッチ情報）を取得したり、更新したりする処理を司る。ＭＱサーバ情報取得部３２４は、レジストリ１４に記憶されたＭＱサーバ１６に係る情報（ＭＱサーバ情報）を取得する処理を司る。アプリケーション情報取得部３２５は、レジストリ１４に記憶されたアプリケーション１２に係る情報（アプリケーション情報）を取得する処理を司る。

【００９４】

ＭＱ関連部３３は、ＭＱサーバ１６と連携する処理を行う。ＭＱ関連部３３は、ＭＱサーバ接続部３３１と、キュー名算出部３３２と、キュー作成部３３３と、メッセージ受信部３３４と、メッセージ送信部３３５とを含む。

【００９５】

ＭＱサーバ接続部３３１は、ＭＱサーバ１６とのセッション確立及びセッション切断に係る処理を行う。キュー名算出部３３２は、ＭＱサーバ１６に備えられるキューの名称（識別子）であるキュー名を算出する処理を行う。キュー作成部３３３は、ＭＱサーバ１６にてキューＱを作成するための処理を司る。

【００９６】

メッセージ受信部３３４は、ＭＱサーバ１６から送信されるメッセージの受信処理を司る。メッセージ送信部３３５は、ＭＱサーバ１６へ送る（所定のキューＱＡに書き込まれる）メッセージの送信処理を司る。制御プロトコル処理部３１は、ＭＱサーバ１６から受信されるメッセージを用いて、配下のスイッチ２に対する制御メッセージを発行（生成）し、当該スイッチ２へ送る処理を行う。

【００９７】

＜＜アプリケーション＞＞

アプリケーション１２は、コントローラ１１からのメッセージ（例えば、或るスイッチ２でのイベント発生を示す）をＭＱサーバ１６を介して受信し、メッセージの内容に従って、自身の固有の制御ロジック４３に基づく処理（例えば、パス計算、トポロジ発見、ＱｏＳ制御など）を行う。

【００９８】

また、アプリケーション１２は、固有の制御ロジック４３の処理結果としてスイッチ２の制御を行う場合には、ＭＱサーバ１６を介してコントローラ１１にメッセージを送信する。さらに、アプリケーション１２は、ＭＱサーバ１６とメッセージのやりとりを行う。

【００９９】

図１５は、アプリケーションの構成例を示す図である。アプリケーション１２は、レジストリ関連部４１と、ＭＱ関連部４２と、上記した固有のロジック４３とを含む。レジストリ関連部４１は、レジストリ接続部４１１と、スイッチ情報取得４１２と、アプリケーション情報取得・更新部４１３と、ＭＱサーバ情報取得部４１４とを含む。

【０１００】

レジストリ接続部４１１は、レジストリ１４に格納（記憶）された情報を参照・取得するためにアプリケーション１２をレジストリ１４と接続する処理を司る。スイッチ情報取得部４１２は、レジストリ１４に記憶されたスイッチ２に係る情報（スイッチ情報）を取得する処理を司る。

【０１０１】

アプリケーション情報取得・更新部４１３は、レジストリ１４に記憶されたアプリケーション１２に係る情報（アプリケーション情報）を取得したり、更新したりする処理を司る。ＭＱサーバ情報取得部４１４は、レジストリ１４に記憶されたＭＱサーバ１６に係る情報（ＭＱサーバ情報）を取得する処理を司る。

【０１０２】

ＭＱ関連部４２は、ＭＱサーバ１６と連携する処理を行う。ＭＱ関連部４２は、ＭＱサーバ接続部４２１と、キュー名算出部４２２と、キュー作成部４２３と、メッセージ受信部４２４と、メッセージ送信部４２５とを含む。

【０１０３】

ＭＱサーバ接続部４２１は、ＭＱサーバ１６とのセッション確立及びセッション切断に係る処理を行う。キュー名算出部４２２は、ＭＱサーバ１６に備えられるキューの名称（識別子）であるキュー名を算出する処理を行う。キュー作成部４２３は、ＭＱサーバ１６にてキューＱＡを作成するための処理を司る。

【０１０４】

メッセージ受信部４２４は、ＭＱサーバ１６から送信されるメッセージの受信処理を司る。メッセージ送信部４２５は、ＭＱサーバ１６へ送る（所定のキューＱに書き込む）メッセージの送信処理を司る。

【０１０５】

＜＜レジストリ＞＞

図１６は、レジストリの構成例を示す図である。レジストリ１４は、コントローラ１１、アプリケーション１２、スイッチ２、ＭＱサーバ１６に関する情報を格納（記憶）する。レジストリ１４は、メモリ５上に作成される。レジストリ１４は、コントローラ情報格納部１４１と、スイッチ情報格納部１４２と、アプリケーション情報格納部１４３と、ＭＱサーバ情報格納部１４４と、情報変更検出部１４５と、情報変更通知部１４６とを含む。

【０１０６】

図１７は、コントローラ情報格納部１４１が有するテーブルのデータ構造例を示す。コントローラ情報格納部１４１は、コントローラ情報として、現在動作するコントローラ１１のＩＤ（コントローラＩＤ）毎に生成された１以上のレコード（エントリ）を有するリストを記憶（保持）している。

【０１０７】

レコードには、コントローラＩＤと関連づけられた、マスタコントローラとして管理するスイッチ２（配下のスイッチ２）のスイッチＩＤと、スレーブコントローラとして接続されている（セッションが確立されている）スイッチ２のスイッチＩＤとが記憶されている。また、各レコードに対応づけて、エラーフラグが管理される。コントローラ１１の障害が検知されたとき、エラーフラグはオンに設定される。

【０１０８】

図１８は、スイッチ情報格納部１４２が有するテーブルのデータ構造例を示す。スイッチ情報格納部１４２は、コントローラ１１と接続されているスイッチ２のスイッチＩＤ毎に生成された１以上のレコード（エントリ）を有するリスト（一覧）を記憶（保持）している。レコードには、スイッチＩＤと関連づけられた、マスタコントローラのコントローラＩＤと、スレーブコントローラのコントローラＩＤと、エラーフラグとが記憶される。エラーフラグは、対応するスイッチ２に障害が生じたときにオンとなる。

【０１０９】

図１９は、アプリケーション情報格納部１４３が有するテーブルのデータ構造例を示す

。アプリケーション情報格納部１４３は、現在動作するアプリケーション１２のＩＤ（アプリケーションＩＤ）毎に、当該アプリケーションＩＤを有するアプリケーション１２によって提供されるアプリケーション機能（アプリケーションプロセス群）の種別を示す識別子（機能グループＩＤ）と、エラーフラグとを記憶（保持）している。エラーフラグは、該当するアプリケーション１２の障害が検知されたときにオンとなる。

【０１１０】

上記のように、本実施形態では、アプリケーションプロセス群の識別子（機能グループＩＤ）と、アプリケーションプロセスの識別子（アプリケーションＩＤ）とを含む。機能グループＩＤは、或る機能をもたらすためのアプリケーションプロセス群を特定する識別子であり、アプリケーション機能を特定する識別子として使用される。アプリケーションプロセスの識別子はアプリケーション１２自身を他のアプリケーションから区別するための固有の識別子である。

【０１１１】

図２０は、ＭＱサーバ情報格納部１４４が有するテーブルのデータ構造例を示す。ＭＱサーバ情報格納部１４４には、現在動作しているＭＱサーバ１６のＩＰアドレス及びポート番号の一覧と、スイッチＩＤとキュー名との対応ルール（スイッチＩＤとキュー名との対応づけに係るルール）と、アプリケーション機能種別（機能グループＩＤ）とキュー名との対応ルール（機能グループＩＤとキュー名との対応づけに係るルール）と、エラーフラグとが格納される。エラーフラグは、障害（エラー）の生じたＭＱサーバ１６に対してセットされる。

【０１１２】

情報変更検出部１４５は、レジストリ１４に記憶されたコントローラ情報，スイッチ情報，アプリケーション情報，ＭＱサーバ情報が変更されたことを検出する処理を司る。情報変更通知部１４６は、情報変更検出部１４５にて検出された情報の変更をコントローラ１１，アプリケーション１２などに通知する処理を司る。

【０１１３】

＜＜ＭＱサーバ＞＞

図２１は、ＭＱサーバの構成例を示す図である。ＭＱサーバ１６は、メッセージングシステム１５の中核になるコンポーネントである。ＭＱサーバ１６は、メッセージ受信部１６１と、メッセージ送信部１６２と、メッセージ分配先選択部１６３と、キュー追加・削除部１６４とを含んでいる。また、ＭＱサーバ１６は、接続クライアント管理部１６５と、送信先情報（Subscribe情報）管理部１６６と、複数のキューＱ及びＱＡとを含む。

【０１１４】

メッセージ受信部１６１は、コントローラ１１やアプリケーション１２からの情報を受信し、対応するキューＱ又はＱＡに書き込む処理を司る。メッセージ送信部１６２は、送信先情報管理部１６６に記憶された送信先情報（送信先情報に基づく設定）に従ってキューＱ又はＱＡの先頭から読み出される情報（メッセージ）を送信先（コントローラ１１，アプリケーション１２）へ送信する処理を司る。メッセージ受信部１６１は受信部の一例であり、メッセージ送信部１６２は送信部の一例である。

【０１１５】

メッセージ分配先選択部１６３は、複数のアプリケーション１２の何れかにキューＱＡに格納されたメッセージを送信する場合に、所定のルールに従って、送信先のアプリケーション１２を決定する処理を司る。キュー追加・削除部１６４は、コントローラ１１やアプリケーション１２からの要求に応じて、キューの追加や削除を行う。

【０１１６】

接続クライアント管理部１６５は、現在セッションが確立されているクライアント（コントローラ１１，アプリケーション１２）とのセッション情報（クライアントのＩＰアドレス，ポート番号など）を記憶する。

【０１１７】

送信先情報管理部１６６は、各キューＱ及びＱＡに格納されたメッセージの送信先を示

す送信先情報を格納する。送信先情報は、例えば、コントローラ１１やアプリケーション１２からＭＱサーバ１６がメッセージ受信部１６１で受信するsubscribe（購読）コマンドに含まれるsubscribe情報である。subscribe情報は、例えば、キュー名及びセッション情報（送信先のＩＰアドレスなど）を含む。

【０１１８】

ここで、subscribe設定に関して説明する。subscribe設定は、コンシューマ（コントローラ１１，アプリケーション１２）のsubscribe宣言によって、コンシューマにより指定されたキューＱ（ＱＡ）に格納された情報（メッセージ）がコンシューマに届くようにする設定である。

【０１１９】

コントローラ１１及びアプリケーション１２は、或るキューＱ又はキューＱＡに格納された情報（メッセージ）の配信を所望する場合、subscribeコマンド（subscribe宣言）をＭＱサーバ１６に送る。subscribeコマンドは、上記したsubscribe情報を含んでいる。

【０１２０】

ＭＱサーバ１６は、subscribeコマンドを受け取ると、subscribe情報に含まれるキュー名を有するキューＱ又はキューＱＡをセッション情報で特定される宛先（送信先）へ送信するためのコネクション設定を行う。コネクション設定がなされると、該当するキューＱ（ＱＡ）の先頭から読み出される（プッシュされる）メッセージは、コネクションを通じて送信先へ送信される。このように、キューＱ及びキューＱＡに格納された情報（メッセージ）は、送信先情報に基づいて送信される。

【０１２１】

subscribeコマンドのやりとりは、ＭＱサーバ１６とコントローラ１１（アプリケーション１２）とのセッション確立時、或いはセッション確立後の適宜のタイミングで実施される。subscribeコマンド中のsubscribe情報が、上記した送信先情報に相当し、キュー名がキュー指定情報に相当する。セッション情報は、送信先指定情報（指定されたキューから読み出された情報を送信先に該当するコントローラ１１へ送信するための情報）に相当する。

【０１２２】

ＭＱサーバ１６が備える複数のキューとして、ＭＱサーバ１６は、複数のスイッチのそれぞれに対応する１又は２以上のキューＱと、アプリケーション機能（機能グループＩＤ）毎に設けられた１又は２以上のキューＱＡとを備える。

【０１２３】

各キューＱは、アプリケーション１２から送信された、各キューＱに対応するスイッチ２向けのメッセージを一時的に格納する。格納されたメッセージは、送信先情報を用いた送信先設定に従って、各キューＱに対応するスイッチ２を制御するコントローラ１１（マスタコントローラ）へ転送（配信）される。

【０１２４】

各キューＱＡは、コントローラ１１から送信された、各キューＱＡに対応するアプリケーション機能に係るメッセージを一時的に格納する。格納された情報は、送信先情報を用いた送信先設定に従って、各キューＱＡに対応するアプリケーション機能を提供するアプリケーション１２へ転送される。

【０１２５】

キューＱ又はキューＱＡに格納される情報（メッセージ）は、書き込み先のキュー名（キューの識別情報：書き込み先指定情報）を含んでいる。ＭＱサーバ１６は、書き込み対象のメッセージを受信すると、当該メッセージに付与された書き込み先情報（キュー名）を参照し、該当するキューＱ又はキューＱＡへの書き込み（格納）を行う。

【０１２６】

＜動作例＞

次に、分散コントローラにおける動作例について説明する。以下に説明するコントローラ１１，アプリケーション１２，ＭＱサーバ１６の動作は、ＣＰＵ４によるプログラムの

実行によってなされる。
【０１２７】
＜＜コントローラ起動時＞＞
図２２は、コントローラの起動時におけるコントローラ１１の動作例を説明するフローチャートである。０１において、コントローラ１１が起動すると、レジストリ接続部３２１がレジストリ１４との接続を行う。コントローラ１１のコントローラ情報取得・更新部３２２は、レジストリ１４のコントローラ情報格納部１４１にコントローラ１１自身のコントローラ情報を格納する。
【０１２８】
次の０２において、コントローラ１１のＭＱサーバ情報取得部３２４は、レジストリ１４のＭＱサーバ情報格納部１４４から、ＭＱサーバ情報として、動作中のＭＱサーバ１６のＩＰアドレス及びポート番号の一覧を取得する。
【０１２９】
次の０３において、コントローラ１１のＭＱサーバ情報取得部３２４は、レジストリ１４のＭＱサーバ情報格納部１４４からスイッチＩＤとキュー名との対応ルールを取得する。続いて、コントローラ１１のスイッチ情報取得・更新部３２３は、レジストリ１４のスイッチ情報格納部１４２に格納されたスイッチ情報を取得する。コントローラ１１のキュー名算出部３３２は、スイッチ情報を参照と対応ルールとを用いて、自身がマスタコントローラに該当するスイッチ２のスイッチＩＤから対応するキュー名を生成する。
【０１３０】
次の０４において、コントローラ１１のＭＱサーバ接続部３３１は、０２の処理で得られたＩＰアドレス及びポート番号を用いて、動作中の全てのＭＱサーバ１６とのセッションを確立する。
【０１３１】
次の０５において、コントローラ１１は、０３にて生成したキュー名を有するキューがＭＱサーバ１６に存在するかを確認する。当該確認は、ＭＱサーバ１６に対して問合せを行っても良く、レジストリ１４のＭＱサーバ情報に、キュー名とスイッチとの対応関係が保持されるようにして、当該対応関係を参照するようにしても良い。
【０１３２】
このとき、ＭＱサーバ１６において、０３で生成したキュー名を有するキューＱが存在しない場合には（０５，ＮＯ）、コントローラ１１のキュー作成部３３３は、生成したキュー名のキューをＭＱサーバ１６に依頼する（０６）。ＭＱサーバ１６のキュー追加・削除部１６４は、依頼に従ってキューＱを作成する。
【０１３３】
０７に処理が進んだ場合には、コントローラ１１が制御するスイッチ２に対応するキューＱに関して、subscribe設定が済んでいるか否かを判定する。subscribe設定が済んでいない場合には（０７，ＮＯ）、コントローラ１１は、subscribeコマンドをＭＱサーバ１６に送り、所定のキュー中のメッセージが自身に送信されるようにする（０８）。subscribe設定が終了すると、起動時における処理が終了する。なお、コントローラ１１の増加時におけるコントローラ１１の動作は、コントローラ起動時の動作に準じる。
【０１３４】
＜＜コントローラ終了時＞＞
次に、コントローラ１１の終了時におけるコントローラ１１の動作例を説明する。図２３は、コントローラの終了時における動作例を示すフローチャートである。
【０１３５】
１１において、コントローラ１１のレジストリ接続部３２１は、レジストリ１４との接続を行い、コントローラ情報取得・更新部３２２は、レジストリ１４のコントローラ情報格納部１４１からコントローラ１１自身のコントローラ情報を削除する。
【０１３６】
次の１２において、コントローラ１１は、配下のスイッチ２の全てとの接続を切断する

。次の１３において、コントローラ１１は、スイッチ情報取得・更新部３２３によって、レジストリ１４のスイッチ情報１４２を更新する（スイッチ２との切断を反映した内容に書き換える）。次の１４において、コントローラ１１は、全てのＭＱサーバ１６との接続（セッション）を切断する。そして、終了時の処理が終了する。

【０１３７】

＜＜スイッチとの接続時＞＞

次に、スイッチ２との接続時におけるコントローラ１１の動作例について説明する。図２４は、スイッチとの接続時におけるコントローラの動作例を示すフローチャートである。

【０１３８】

最初の２１において、コントローラ１１のレジストリ接続部３２１は、レジストリ１４に接続し、スイッチ情報取得・更新部３２３は、レジストリ１４のスイッチ情報格納部１４２から該当するスイッチ情報（図１８参照）を取得する。

【０１３９】

次の２２において、コントローラ１１のスイッチ情報取得・更新部３２３は、スイッチ情報を参照し、接続されたスイッチ２がまだどのコントローラ１１にも接続されていないか否かを判定する。スイッチ２が未接続の場合（２２，ＹＥＳ）には、自身と当該スイッチ２との接続を示すスイッチ情報を、レジストリ１４のスイッチ情報格納部１４２に格納する（２３）。そうでない場合には（２２，ＮＯ）、スイッチ情報取得・更新部３２３は、スイッチ情報格納部１４２に格納されている既存のスイッチ情報を更新する（２４）。すなわち、コントローラ１１は、当該スイッチ２との接続が他のコントローラ１１から自身に変更されたことを示す情報を書き込む）。

【０１４０】

次の２５において、コントローラ１１のキュー名算出部３３２は、既に説明した手法を用いて、接続されたスイッチ２のスイッチＩＤからキュー名を生成する。次の２６において、コントローラ１１は、２５で作成したキュー名を有するキューがＭＱサーバ１６に存在するかを確認する。

【０１４１】

このとき、キュー名に該当するキューがＭＱサーバ１６に存在しない場合には（２６，ＮＯ）、コントローラ１１のキュー作成部３３３は、生成したキュー名でのキュー作成を動作中の全てのＭＱサーバ１６に依頼する（２７）。ＭＱサーバ１６のキュー追加・削除部１６４は、依頼に従ってキューＱを作成する。

【０１４２】

次の２８で、コントローラ１１は、作成されたキューＱに対するsubscribe設定を行う。自身が制御するスイッチ２に対応する１以上のキューＱに関して、ＭＱサーバ１６がコントローラ１１からsubscribeコマンド（subscribe宣言）を受け取ると、ＭＱサーバ１６は、当該キューＱに格納されるメッセージに関して、宣言を行ったコントローラ１１に当該キューＱから読み出されるメッセージを送信する設定が行われる。なお、２８の処理（subscribe設定）は、コントローラ１１が接続されたスイッチ２のマスタコントローラとなる場合に行われ、スレーブとしての接続では行われない。

【０１４３】

＜＜スイッチとの切断時＞＞

次に、スイッチ２との切断時におけるコントローラの動作例について説明する。図２５は、スイッチ２との切断時におけるコントローラの動作例を示すフローチャートである。

【０１４４】

最初の３１において、コントローラ１１のレジストリ接続部３２１は、レジストリ１４との接続を行い、スイッチ情報取得・更新部３２３は、レジストリ１４のスイッチ情報格納部１４２から切断に係るスイッチ２のスイッチ情報を取得する。

【０１４５】

次の３２において、コントローラ１１のスイッチ情報取得・更新部３２３は、スイッチ

情報を参照し、スイッチ２が他のコントローラ１１と接続しているか否かを判定する。他のコントローラ１１と接続されている場合（３２，ＹＥＳ）には、３３において、既存のスイッチ情報を更新する（自身とスイッチ２との接続に係る情報のみを削除する）。

【０１４６】

スイッチ２が他のコントローラ１１と接続されていない場合（他に接続されたコントローラ１１が存在しない場合）には（３２，ＮＯ）、スイッチ情報取得・更新部３２３は、当該スイッチ２のスイッチ情報を削除する（３４）。さらに、コントローラ１１は、スイッチ２に対応するキューＱの削除を各ＭＱサーバ１６に依頼する（３５）。各ＭＱサーバ１６は、該当するキューＱを削除する。

【０１４７】

＜＜アプリケーション起動時＞＞

次に、アプリケーション１２の起動時におけるアプリケーション１２の動作例について説明する。図２６は、アプリケーションの起動時におけるアプリケーションの動作例を示すフローチャートである。

【０１４８】

最初の４１において、アプリケーション１２のレジストリ接続部４１１がレジストリ１４との接続を行い、アプリケーション情報取得・更新部４１３がアプリケーション情報格納部１４３に対し、アプリケーション１２自身のアプリケーション情報を格納する。

【０１４９】

次の４２において、アプリケーション１２のＭＱサーバ情報取得部４１４は、レジストリ１４のＭＱサーバ情報格納部１４４から、動作中のＭＱサーバ１６のＩＰアドレス及びポート番号を取得する。

【０１５０】

次の４３において、アプリケーション１２のＭＱサーバ情報取得部４１４は、レジストリ１４のＭＱサーバ情報格納部１４４から、アプリケーション機能種別（機能グループＩＤ）とキューＱＡとの対応ルールを取得する。キュー名算出部４２２は、アプリケーション１２自身が属するアプリケーション機能種別に対応するキュー名を生成する。

【０１５１】

次の４４において、アプリケーション１２は、ＭＱサーバ接続部４２１によって、４２で得たＩＰアドレス及びポート番号を用いて、動作中の全てのＭＱサーバ１６とのセッションを確立する。

【０１５２】

このとき、ＭＱサーバ１６において、４３で生成したキュー名を有するキューＱが存在しない場合には（４５，ＮＯ）、アプリケーション１２のキュー作成部４２３は、生成したキュー名のキューの作成をＭＱサーバ１６に依頼する。ＭＱサーバ１６のキュー追加・削除部１６４は、依頼に従ってキューＱＡを作成する。

【０１５３】

４７に処理が進んだ場合には、アプリケーション１２は、対応するキューＱＡに関して、subscribe設定が済んでいるか否かを判定する。subscribe設定が済んでいない場合には（４７，ＮＯ）、アプリケーション１２は、subscribeコマンドをＭＱサーバ１６に送り、所定のキューＱＡ中のメッセージが自身に送信されるようにする（４８）。subscribe設定が終了すると、起動時における処理が終了する。なお、アプリケーション１２の増加時におけるアプリケーション１２の動作は、上述したアプリケーション起動時の動作に準じる。

【０１５４】

＜＜アプリケーション終了時＞＞

次に、アプリケーション１２の終了時におけるアプリケーション１２の動作例について説明する。図２７は、アプリケーション１２の終了時におけるアプリケーション１２の動作例を示すフローチャートである。

【０１５５】

最初の５１において、アプリケーション１２のレジストリ接続部４１１がレジストリ１４との接続を行い、アプリケーション情報取得・更新部４１３が、レジストリ１４のアプリケーション情報格納部１４３から自身のアプリケーション情報を削除する。

【０１５６】

次の５２において、アプリケーション１２、全てのＭＱサーバ１６との接続（セッション）を切断する。その後、図２７の処理が終了する。

【０１５７】

＜＜ＭＱサーバ起動時＞＞

次に、ＭＱサーバ１６の起動時におけるＭＱサーバ１６の動作例について説明する。図２８は、ＭＱサーバ１６の起動時におけるＭＱサーバ１６の動作例を示すフローチャートである。５１において、ＭＱサーバ１６は、レジストリ１４との接続を行い、レジストリ１４のＭＱサーバ情報格納部１４４に自身のＭＱサーバ情報を格納する。

【０１５８】

＜＜ＭＱサーバ終了時＞＞

次に、ＭＱサーバ１６の終了時におけるＭＱサーバ１６の動作例について説明する。図２９は、ＭＱサーバ１６の終了時におけるＭＱサーバ１６の動作例を示すフローチャートである。５６において、ＭＱサーバ１６は、レジストリ１４との接続を行い、レジストリ１４のＭＱサーバ情報格納部１４４から自身のＭＱサーバ情報を削除する。

【０１５９】

＜＜スイッチの非同期イベントの発生時＞＞

或るスイッチ２において非同期イベントが発生した場合におけるコントローラ１１（或るスイッチ２のマスタコントローラ）における動作例について説明する。図３０は、非同期イベント発生時におけるコントローラの動作例を示すフローチャートである。

【０１６０】

最初の６１において、コントローラ１１の制御プロトコル処理部３１が、スイッチ２における非同期イベントの発生を検知する。例えば、制御プロトコル処理部３１は、スイッチ２からのイベント発生を示す通知を受領することで、イベント発生を検知することができる。

【０１６１】

次の６２において、コントローラ１１のレジストリ接続部３２１は、レジストリ１４への接続を行う。ＭＱサーバ情報取得部３２４は、ＭＱサーバ情報格納部１４４からＭＱサーバ情報を取得し、ＭＱサーバ１６のＩＰアドレス及びポート番号の一覧を得る。

【０１６２】

次の６３において、ＭＱサーバ情報取得部３２４は、レジストリ１４のＭＱサーバ情報格納部１４４からアプリケーション機能種別（機能グループＩＤ）とキュー名の対応ルールを取得する。次の６４において、コントローラ１１のキュー名算出部３２２は、非同期イベントに対応するアプリケーション機能種別（機能グループＩＤ）に対応するキュー名をＭＱサーバ１６から得られた対応ルールを用いて生成する。

【０１６３】

次の６５において、コントローラ１１は、６２で取得したＭＱサーバ１６のＩＰアドレス及びポート番号の一覧から、１つのＭＱサーバ１６を選択する。次の６６において、コントローラ１１は、非同期イベントの発生を対応するアプリケーション機能を提供するアプリケーション１２に通知するためのメッセージを作成する。そして、６７において、コントローラ１１のメッセージ送信部４２５は、６６で作成したメッセージを６５で選択したＭＱサーバ１６に送信する。メッセージは、当該メッセージに付与されたキュー名に基づいて、ＭＱサーバ１６に設けられたキューＱＡ（非同期イベントに対応するアプリケーション機能と対応づけられたキューＱＡ）に格納される。

【０１６４】

＜＜ＭＱサーバからのメッセージ受信時（コントローラ）＞＞

次に、ＭＱサーバ１６からメッセージを受信した際におけるコントローラ１１の動作例

について説明する。図３１は、ＭＱサーバ１６からメッセージを受信したコントローラ１１の動作例を示すフローチャートである。

【０１６５】

７１において、メッセージ受信部３３４がＭＱサーバ１６から送信されたメッセージを受信する。すると、制御プロトコル処理部３１が、受信したメッセージが格納されていたキューＱに対応するスイッチ２に対して制御メッセージを発行する（７２）。制御メッセージは、該当するスイッチ２に送られる。

【０１６６】

＜＜ＭＱサーバからのメッセージ受信時（アプリケーション）＞＞

次に、ＭＱサーバ１６からメッセージを受信した際におけるアプリケーション１２の動作例について説明する。図３２は、ＭＱサーバからメッセージを受信したアプリケーションの動作例を示すフローチャートである。

【０１６７】

７４において、メッセージ受信部４２４は、ＭＱサーバ１６から送信されたメッセージを受信する。すると、７５において、アプリケーション１２は、受信されたメッセージの内容を解釈し、解釈の結果に従って、固有の制御ロジック４３に応じた処理を実行する。

【０１６８】

＜＜アプリケーションからスイッチへの制御をする場合の動作例＞＞

図３３は、アプリケーション１２が或るスイッチ２の制御を行う場合におけるアプリケーション１２の動作例を示すフローチャートである。

【０１６９】

最初の８１において、アプリケーション１２のレジストリ接続部４１１は、レジストリ１４への接続処理を行い、スイッチ情報取得部４１２が、スイッチ情報格納部１４２からのスイッチ情報を用いて、制御を行うスイッチ２のスイッチＩＤを決定する。

【０１７０】

次の８２において、アプリケーション１２のＭＱサーバ情報取得部４１４は、レジストリ１４のＭＱサーバ情報格納部１４４からスイッチＩＤとキューとの対応ルールを得て、キュー名算出部４２２は、制御対象のスイッチ２のスイッチＩＤに対応するキュー名を求める。

【０１７１】

次の８３において、アプリケーション１２のＭＱサーバ情報取得部４１４は、ＭＱサーバ情報格納部１４４からＭＱサーバ情報を取得することで、ＭＱサーバ１６のＩＰアドレス及びポート番号の一覧を得る。

【０１７２】

次の８４において、アプリケーション１２は、８３で取得したＭＱサーバ１６のＩＰアドレス及びポート番号の一覧から、１つのＭＱサーバ１６を選択する。次の８５において、アプリケーション１２は、スイッチ２を制御するためのメッセージを作成する。

【０１７３】

次の８６において、アプリケーション１２は、作成したメッセージを、選択したＭＱサーバ１６に送信する。メッセージには、キュー名が含まれており、ＭＱサーバ１６は、キュー名を有するキューＱに受信されたメッセージを格納する。

【０１７４】

＜コントローラの数の変化時における動作＞

次に、コントローラ１１の数が増加又は減少した場合における動作例について説明する。

【０１７５】

＜＜コントローラ増加時＞＞

以下、新規のコントローラ１１の追加によって、コントローラ１１の数が増加した場合の動作例について説明する。追加時の動作は、コントローラ１１の起動時における動作と重複する部分を有する。図３４は、コントローラの数が増加した場合の動作例を示すフロ

ーチャートである。
【０１７６】
最初の９１では、　追加に係るコントローラ１１のレジストリ接続部３２１は、レジストリ１４への接続（アクセス）を行い、コントローラ１１自身のコントローラ情報（コントローラＩＤなど）を、レジストリ１４のコントローラ情報格納部１４１に格納（追加）する。
【０１７７】
次の９２では、追加に係るコントローラ１１が既存のコントローラ１１から或るスイッチ２についての制御を引き継ぐか否かを判定する。このとき、追加に係るコントローラ１１が既存のコントローラ１１から或るスイッチ２についての制御を引き継ぐ（或るスイッチ２のマスタコントローラとなる）場合には（９２，ＹＥＳ）、処理が９３に進み、そうでなければ（９２，ＮＯ）、処理が９４に進む。
【０１７８】
９３に処理が進んだ場合には、コントローラ１１のスイッチ情報取得・更新部３２３は、レジストリ１４のスイッチ情報格納部１４２にアクセスし、或るスイッチ２のマスタコントローラに係る情報を更新する。具体的には、或るスイッチ２のマスタコントローラが自身であり、既存のコントローラ１１が或るスイッチのスレーブコントローラであることを示す情報を格納する。その後、処理が９４に進む。
【０１７９】
９４に処理が進むと、追加に係るコントローラ１１のＭＱサーバ情報取得部３２４は、レジストリ１４のＭＱサーバ情報格納部１４４からＭＱサーバ情報を取得することで、接続先のＭＱサーバ１６のＩＰアドレス及びポート番号の一覧を得る。
【０１８０】
次の９５において、追加に係るコントローラ１１は、９４で得たＭＱサーバのＩＰアドレス及びポート番号の一覧を用いて各ＭＱサーバ１６とのセッションを確立する。次の９６において、コントローラ１１のキュー名算出部３３２は、自身がマスタコントローラとして制御するスイッチ２のスイッチＩＤと、レジストリ１４から得たスイッチＩＤとキュー名との対応ルールとを用いてキュー名を生成する。このとき、コントローラ１１が複数のスイッチ２に対する制御を行う場合には、各スイッチ２に対するキュー名が生成される。
【０１８１】
次の９７では、ＭＱサーバ１６において、９６で生成したキュー名を有するキューＱが存在するか否かが判定される。このとき、ＭＱサーバ１６において、手順３で生成したキュー名を有するキューＱが存在しない場合には（９７，ＮＯ）、コントローラ１１のキュー作成部３３３は、生成したキュー名のキューをＭＱサーバ１６に依頼する（９８）。ＭＱサーバ１６のキュー追加・削除部１６４は、依頼に従ってキューＱを作成する。
【０１８２】
次の９９では、コントローラ１１は、生成したキュー名を有するキューＱに格納されるメッセージのSubscribe（購読）を行う。すなわち、コントローラ１１は、購読情報（Subscribe情報）を含むSubscribeコマンドを生成し、ＭＱサーバ１６へ送る。購読情報は、メッセージの購読を所望するキュー名と、メッセージの送信先の情報（例えば、コントローラ１１とＭＱサーバ１６とのセッション情報）とを含む。
【０１８３】
ＭＱサーバ１６は、Subscribeコマンドを受け取ると、当該コマンド中の購読情報をキューＱに格納されるメッセージの送信先情報として、購読情報管理部１６６に格納する。ＭＱサーバ１６とコントローラ１１とは、該当するキューＱの先頭から読み出されるメッセージをコントローラ１１へ送信するコネクションの設定を行う。コネクションの設定によって、当該キューＱの先頭からメッセージが読み出されたとき、当該メッセージは、予め設定されたコネクションを用いて送信先のコントローラ１１へ送られる。
【０１８４】

このように、本実施形態では、コントローラ１１がSubscribeコマンド（購読情報：送信先情報の一例）をＭＱサーバ１６に送り、購読情報に基づくコネクションが設定される。これによって、コネクション設定後は、キューＱから読み出されるメッセージに関して個々の宛先判定が行われることなく、送信先の（Subscribe宣言を行った）コントローラ１１へメッセージが送信される。

【０１８５】

次の１００では、追加に係るコントローラ１１のアプリケーション情報取得部３２５は、レジストリ１４のアプリケーション情報格納部１４３に格納されているアプリケーションプロセス群（アプリケーション機能）の識別情報（機能グループＩＤ）を取得し、キュー名算出部３３２がアプリケーション機能に対応するキュー名を作成する。当該キュー名は、ＭＱサーバ１６に設けられるアプリケーション１２宛てのメッセージを格納するキューＱＡのキュー名であり、アプリケーション１２向けのメッセージをＭＱサーバ１６を介して対応するアプリケーション１２へ送るときに使用される。

【０１８６】

[レジストリの情報更新に係る処理]

上述のように、レジストリ１４では、コントローラ情報格納部１４１，スイッチ情報格納部１４２，アプリケーション情報格納部１４３，ＭＱサーバ情報格納部１４４に格納された情報が更新（変更）されると、当該変更が情報変更検出部１４５によって検出される。すると、情報が変更されたことが情報変更通知部１４６によって各コントローラ１１に通知される。

【０１８７】

典型的には、スイッチ２，コントローラ１１，アプリケーション１２，ＭＱサーバ１６の動作ないし処理が所定の監視機能によって監視されており、これらに障害（エラー）が発生した場合には、レジストリ１４における対応箇所のエラーフラグがオンになる。エラーフラグがオンになったことは、情報の更新として情報変更検出部１４５により検出され、当該更新を示す（所定箇所のエラー発生）が所定の通知先（コントローラ１１など）に通知される。このようにして、各コントローラ１１は、レジストリ１４に格納された情報の変更を検知することができ、変更（更新）の内容に応じて以下の動作を行う。

【０１８８】

[[自身がマスタコントローラとして管理するスイッチの情報が更新された場合]]

（１）スイッチ数の減少時

スイッチ情報の更新が通知された場合には、コントローラ１１は、レジストリのスイッチ情報格納部１４２に格納されたスイッチ情報を参照する。このとき、自身がマスタコントローラとなっているスイッチの情報が削除、或いはエラーフラグが設定されている（スイッチ２の数が減少している）場合には、コントローラ１１は、以下の動作を行う。

【０１８９】

すなわち、コントローラ１１は、減少に係るスイッチ２に対応するキューＱに対するSubscribe設定を解除するコマンドをＭＱサーバ１６に送る。これにより、該当のキューＱからのメッセージの送信が停止される。

【０１９０】

（２）スイッチ数の増加時

スイッチ情報の更新によってスイッチ２の数が増加している場合であって、且つ自身が当該スイッチ２のマスタコントローラに設定されている場合には、コントローラ１１は、増加に係るスイッチ２に対応するキューＱに対するSubscribe設定を行う。これによって、コントローラ１１は、当該キューＱから読み出されて送信されてくるメッセージを受信することができる。

【０１９１】

[[動作しているＭＱサーバの数が減少した場合]]

動作中のＭＱサーバ１６の数の減少（例えば、ＭＱサーバ１６の削除）によってレジストリ１４のＭＱサーバ情報が更新されている場合には、コントローラ１１は、減少に係る

ＭＱサーバとのセッションを切断する。

【０１９２】

[[動作しているＭＱサーバの数が増加した場合]]

動作中のＭＱサーバ１６の数の増加によってレジストリ１４のＭＱサーバ情報が更新されている場合には、コントローラ１１は、ＭＱサーバ情報を用いて、増加に係る各ＭＱサーバ１６に対する接続処理を行い、各ＭＱサーバ１６とのセッションを確立する。これによって、複数のＭＱサーバ１６から並列的にメッセージを受信可能となる。

【０１９３】

[[アプリケーションの増加・減少時]]

アプリケーションプロセス群（アプリケーション機能）の増加又は減少によって、レジストリ１４のアプリケーション情報が更新された場合には、コントローラ１１は、当該アプリケーション情報を用いて、コントローラ１１内部で保持しているアプリケーションプロセス群（機能グループＩＤ）に対応するキュー名を更新する。

【０１９４】

すなわち、各コントローラ１１は、コントローラ１１からアプリケーション１２向けにメッセージを送信する（キューＱＡにメッセージを格納させる）ために、メッセージの送信先の機能グループＩＤに対応するキュー名を保持している。アプリケーション機能増加の場合（キューＱＡ増加）には、コントローラ１１は、コントローラ１１内で保持するキューＱＡのキュー名（メッセージの送信先情報）に、増加に係るキューＱＡのキュー名を求めて追加し、アプリケーション機能減少（キューＱＡ削除）の場合には、該当するキュー名を削除する。

【０１９５】

[[配下スイッチからのイベント通知受信時]]

また、コントローラ１１自身がマスタコントローラとして管理するスイッチ２からイベントが通知された場合には、コントローラ１１は、以下の通りに動作する。すなわち、コントローラ１１は、確立済みの複数のＭＱサーバ１６とのセッションのうちから１つを選択する。コントローラ１１は、当該セッションを用いて、コントローラ１１内でキュー名が保持されているキューＱＡに、スイッチ２からのイベントが発生したことを示すメッセージを書き込む。

【０１９６】

また、コントローラ１１自身がSubscribeしているキューＱからメッセージが受信された場合には、コントローラ１１は、以下の通りに動作する。すなわち、受信されたメッセージの内容に応じて、制御プロトコル処理部３１が、受信されたメッセージの内容に応じた制御メッセージを発行し、該当するスイッチ２に送る。

【０１９７】

＜＜コントローラ減少時＞＞

次に、コントローラ１１が障害等によって減少した場合の動作例について説明する。図３５は、コントローラ１１が障害等によって減少した場合の動作例を示すフローチャートである。以下の動作例の説明において、図８に示すコントローラ＃１～＃３，スイッチ＃１～＃６，ＭＱサーバ１６の関係を想定する。

【０１９８】

複数のコントローラ１１のうちの１つ（コントローラ＃３）に障害が発生した場合、所定の監視機能によって、当該コントローラ＃３の障害を示す情報がレジストリ１４に書き込まれる。具体的には、レジストリ１４のコントローラ情報格納部１４１（図１７に示す内容を有している）において、コントローラ＃３に関してエラーフラグがセットされる。

【０１９９】

このようなエラーフラグのセットは、情報変更検出部１４５にてコントローラ情報の変更として検知され、情報変更通知部１４６によって、コントローラ情報の変更通知が残りのコントローラ１１（コントローラ＃１及び＃２）に通知される。

【０２００】

１１１において、各コントローラ＃１，＃２は、コントローラ情報の変更通知を受信すると、レジストリ１４との接続を行い、コントローラ情報格納部１４１の内容（図１７）を参照する。これにより、各コントローラ＃１，＃２は、コントローラ＃３に対して設定されたエラーフラグの参照によって、コントローラ＃３の障害を検知することができる（１３２）。

【０２０１】

次に、各コントローラ＃１，＃２は、自身がコントローラ＃３が行っていたスイッチ制御の引き継ぎを要するか否かを判定する（１１３）。引き継ぎを要する場合には（１１３，ＹＥＳ）、処理が１１４へ進み、引き継ぎを要しない場合には、図３５の処理が終了する。

【０２０２】

具体的には、コントローラ＃１は、コントローラ情報格納部１４１（図１７）の参照において、コントローラ＃１（自身）は、コントローラ＃３がマスタコントローラとして制御していたスイッチ＃６とスレーブ接続されていないため、引き継ぎ不要と判定し、処理を終了する。

【０２０３】

これに対し、コントローラ＃２は、コントローラ情報格納部１４１（図１７）の参照において、コントローラ＃２（自身）は、コントローラ＃３がマスタコントローラとして制御していたスイッチ＃６とスレーブ接続されているため、引き継ぎ要と判定する。

【０２０４】

なお、この例では、引き継ぎ対象のスイッチ２とスレーブ接続されたコントローラ１１がコントローラ＃２のみである例について説明している。もし、スレーブ接続された複数のコントローラ１１がある場合には、所定の優先順位や、コントローラ１１間のネゴシエーションによって、マスタ設定を引き継ぐコントローラ１１が決定される。

【０２０５】

コントローラ＃２は、次の１１４において、レジストリ１４のコントローラ情報格納部１４１の内容を更新する。例えば、コントローラ情報格納部１４１のコントローラ＃２のレコードにおいて、「マスタ」にスイッチ＃６を加え、「スレーブ」からスイッチ＃６を削除する。

【０２０６】

また、コントローラ＃２は、コントローラ＃３に係るレコードの「マスタ」から、スイッチ＃６を削除する。さらに、コントローラ＃２は、スイッチ情報格納部１４２（図１８参照）を参照し、スイッチ＃６のレコードに関して、「マスタ」を“ＣＮＴ＃２”に変更し、「スレーブ」から“ＣＮＴ＃２”を削除する更新を行う。

【０２０７】

次の１１５では、コントローラ＃２は、スイッチ＃６に対応するキューＱについてのsubscribe設定をＭＱサーバ１６に対して行う。これによって、スイッチ＃６向けのメッセージがＭＱサーバ１６からコントローラ＃２へ送られるようになる。

【０２０８】

次の１１６では、コントローラ＃２は、アプリケーション機能に対応するキュー名を、これまでに説明した手法（図３４の１００）と同様の手法で作成する。これにより、コントローラ＃２は、スイッチ＃６からイベント発生通知を受け取ったときに、当該イベントに対応するアプリケーション機能と対応するキュー名を割り出すことができる。但し、当該処理は、スイッチ＃６から実際にイベント発生通知を受信したときに行うようにしても良い。

【０２０９】

以上のように、コントローラ１１は、レジストリ１４に記憶されたスイッチ情報に同期して、subscribe設定を行うキューを増やしたり、減らしたりすることで、コントローラ１１の増減に対応することができる。

【０２１０】

なお、図３４に示した９４の処理と９５の処理は順序が逆であっても良い。また、図２２に示した０３の処理と０４の処理も順序が逆であっても良い。

【０２１１】

＜アプリケーションプロセス数の変化時の動作＞

＜＜アプリケーション増加時＞＞

アプリケーションプロセス（アプリケーション１２）が増加した場合アプリケーション１２の動作フローは、アプリケーション１２の起動時における動作（図２６）とほぼ同様であるので説明を省略する。

【０２１２】

アプリケーション１２は、レジストリ１４の情報変更通知を受け取ることで以下の動作を行うことができる。

【０２１３】

[[スイッチ情報の情報変更通知の受信]]

アプリケーション１２は、スイッチ情報の情報変更通知を受け取ると、レジストリ１４からスイッチ情報を取得する。このとき、スイッチ２が増減している場合には、アプリケーション１２は、ＭＱサーバ１６への送信用に自身が保持しているキュー名の更新処理を行う。すなわち、或るスイッチ２が削除されている場合には、アプリケーション１２は、当該スイッチ２に対応するキュー名を削除する。これに対し、或るスイッチ２が追加されている場合には、これまでに説明した手法で、追加に係るスイッチＩＤからキュー名を作成して保持する。

【０２１４】

[[ＭＱサーバの増減]]

動作中のＭＱサーバ１６の数が減少した（或るＭＱサーバ１６が削除された）場合には、アプリケーション１２は、当該ＭＱサーバ１６とのセッションを切断する。これに対し、ＭＱサーバ１６の数が増加した場合には、当該ＭＱサーバ１６に対する接続を行い、セッションを確立する。

【０２１５】

[[キューからのメッセージ受信時、スイッチ制御時]]

アプリケーション１２は、subscribeしているキューＱＡからのメッセージを受信した場合には、図３２に示した動作を行う。また、アプリケーション１２は、或るスイッチ２に対する制御を所望する場合、図３３に示す処理を行う。

【０２１６】

＜＜アプリケーション減少時＞＞

次に、アプリケーション１２の数が減少した場合の動作例について説明する。複数のアプリケーション１２のうちの１つが終了や削除によって消滅した場合には、当該アプリケーション１２とＭＱサーバ１６とのセッションが切断される。これによって、ＭＱサーバ１６は、当該アプリケーションへメッセージの配信を行わなくなる。

【０２１７】

以上の実施形態に関し、更に以下の付記を開示する。

（付記１）複数のコントローラがアプリケーションからの情報に基づいて複数のスイッチを制御するネットワークシステムに設けられるサーバであって、

スイッチ単位で設けられ、それぞれが自身に対応するスイッチに向けられた前記アプリケーションからの情報を格納する複数のキューと、

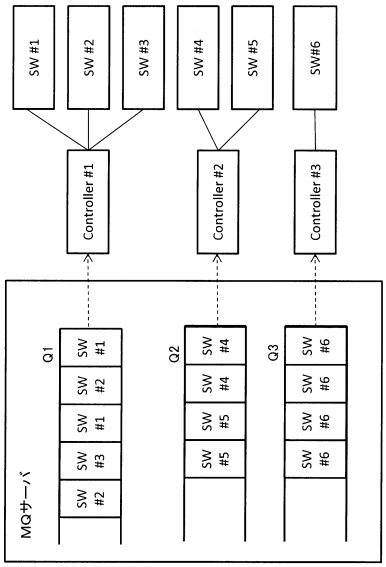
キューの指定情報と当該指定情報によって指定されたキューから読み出される情報を当該情報の送信先に該当するコントローラへ送信するための情報とを含む送信先情報を前記各コントローラから受信する受信部と、

前記複数のキューのそれぞれから読み出される情報を、前記送信先情報に従って前記各キューに対応するスイッチを制御する前記複数のコントローラの１つへ送信する送信部とを含むサーバ。（１）

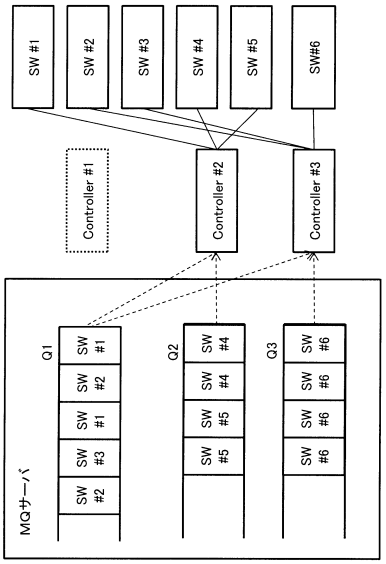
【０２１８】

（付記２）前記受信部は、前記複数のコントローラの数の減少又は増加によって前記複数のスイッチの少なくとも１つを制御するコントローラが変更されたときに、当該変更によって前記複数のスイッチの少なくとも１つを制御する少なくとも１つのコントローラから前記送信先情報を受信し、
前記送信部は、前記少なくとも１つのコントローラから受信された前記送信先情報に従って、前記複数のスイッチの少なくとも１つに対応するキューから読み出される情報を前記少なくとも１つのコントローラへ送信する
付記１に記載のサーバ。（２）
【０２１９】
（付記３）前記ネットワークシステムに複数個設けられ、それぞれが前記複数のキュー，前記受信部，及び前記送信部を含む
請求項１又は２に記載のサーバ。
【０２２０】
（付記４）アプリケーションからの情報に基づいてスイッチを制御するとともに前記スイッチに関する情報であるスイッチ関連情報をアプリケーション向けに送信するコントローラを含むネットワークシステムに設けられるサーバであって、
アプリケーションによって提供される機能単位で設けられ、それぞれが自身に対応する機能と関連する前記スイッチ関連情報を格納する複数のキューと、
キューの指定情報と当該指定情報によって指定されたキューから読み出されるスイッチ関連情報を当該スイッチ関連情報の送信先に該当するアプリケーションへ送信するための情報とを含む送信先情報をアプリケーションから受信する受信部と、
前記複数のキューのそれぞれから読み出される前記スイッチ関連情報を、前記送信先情報に従って当該スイッチ関連情報に関連する機能を提供するアプリケーションへ送信する送信部と
を含むサーバ。（３）
【０２２１】
（付記４） 前記コントローラは、キューの指定情報を含む前記スイッチ関連情報を前記サーバに送信し、前記スイッチ関連情報は前記キューの指定情報に従って、前記スイッチ関連情報と関連する前記機能に対応するキューに格納される
付記３に記載のサーバ。
【０２２２】
（付記５） 前記ネットワークシステムに複数個設けられ、それぞれが前記複数のキュー，前記受信部，及び前記送信部を含む
付記４に記載のサーバ。
【符号の説明】
【０２２３】
１・・・ネットワーク
２・・・スイッチ
３・・・サーバ
４・・・ＣＰＵ
５・・・メモリ
６・・・通信インタフェース
１１・・・コントローラ
１２・・・アプリケーション
１４・・・レジストリ
１６・・・メッセージキューサーバ（ＭＱサーバ）

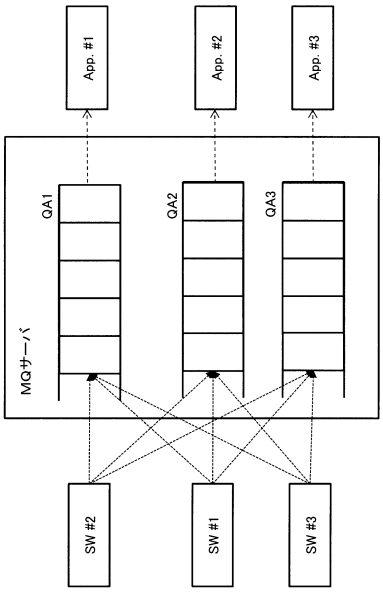
【図１】


SW #1
SW #2
SW #3
SW #4
SW #5
SW#6
Controller #1
Controller #2
Controller #3
MQサーバ
Q1
Q2
Q3

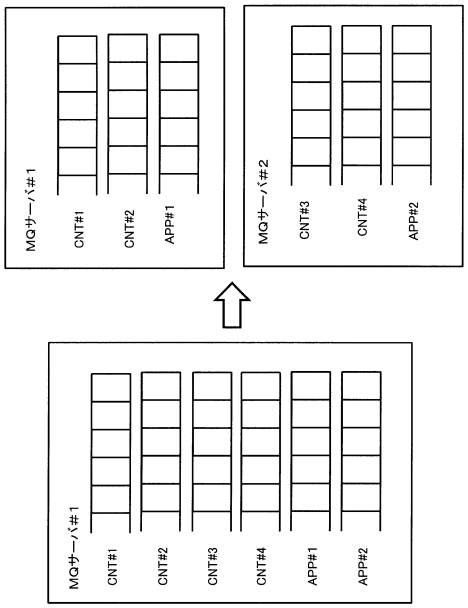
【図２】


SW #1
SW #2
SW #3
SW #4
SW #5
SW#6
Controller #1
Controller #2
Controller #3
MQサーバ
Q1
Q2
Q3

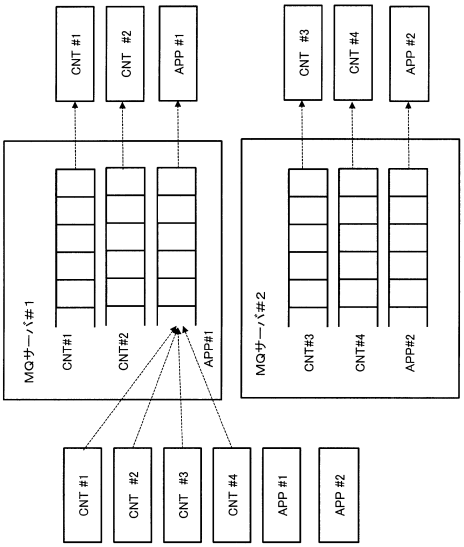
【図３】


App. #1
App. #2
App. #3
MQサーバ
QA1
QA2
QA3
SW #2
SW #1
SW #3

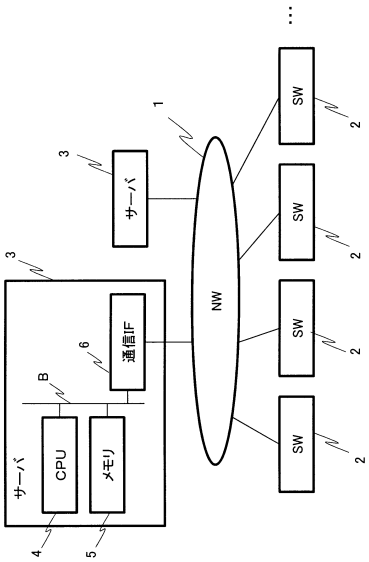
【図４】


MQサーバ#1
CNT#1
CNT#2
APP#1
MQサーバ#2
CNT#3
CNT#4
APP#2
MQサーバ#1
CNT#1
CNT#2
CNT#3
CNT#4
APP#1
APP#2

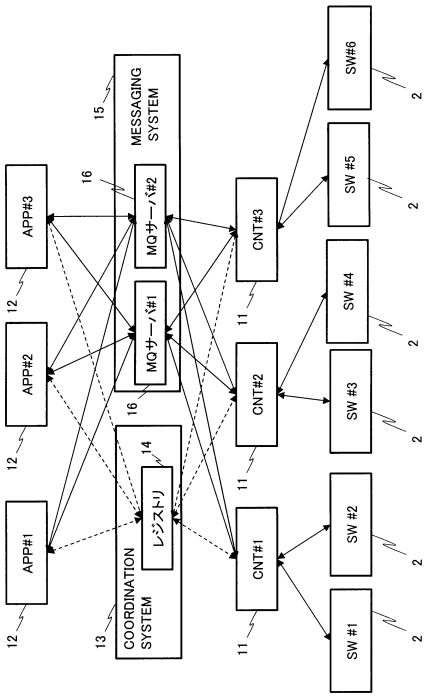
【図５】


CNT #1
CNT #2
APP #1
CNT #3
CNT #4
APP #2
MQサーバ#1
CNT#1
CNT#2
APP#1
MQサーバ#2
CNT#3
CNT#4
APP#2
CNT #1
CNT #2
CNT #3
CNT #4
APP #1
APP #2

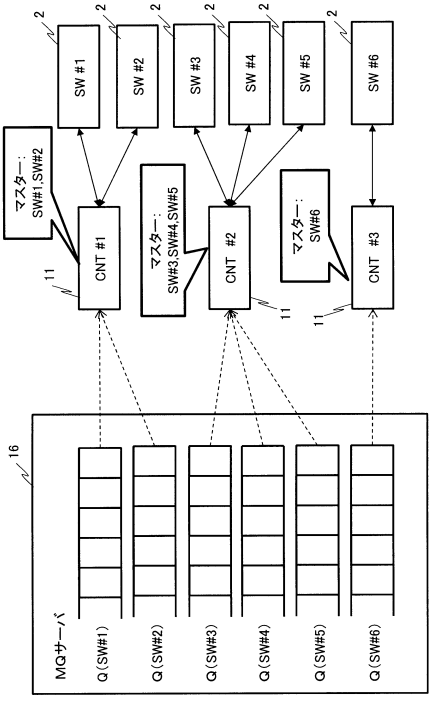
【図６】


1
NW
3
サーバ
3
サーバ
CPU
メモリ
B
6
通信IF
4
5
SW
SW
SW
SW
2
2
2
2
…

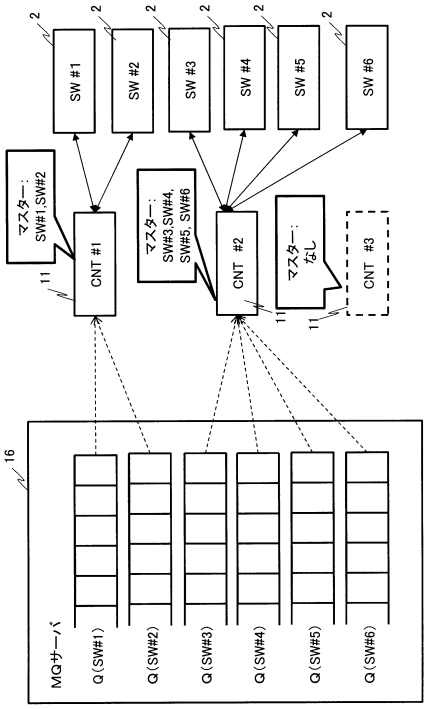
【図７】


15
MESSAGING SYSTEM
16
MQサーバ#2
MQサーバ#1
16
14
レジストリ
COORDINATION SYSTEM
13
APP#3
APP#2
APP#1
12
12
12
CNT#3
CNT#2
CNT#1
11
11
11
SW#6
SW #5
SW #4
SW #3
SW #2
SW #1
2
2
2
2
2
2

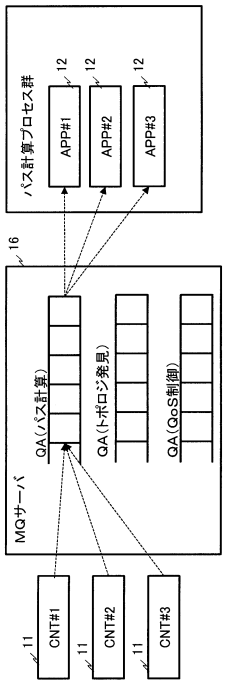
【図８】


2
SW #1
SW #2
SW #3
SW #4
SW #5
SW #6
マスター：SW#1,SW#2
マスター：SW#3,SW#4,SW#5
マスター：SW#6
11
CNT #1
CNT #2
CNT #3
16
MQサーバ
Q(SW#1)
Q(SW#2)
Q(SW#3)
Q(SW#4)
Q(SW#5)
Q(SW#6)

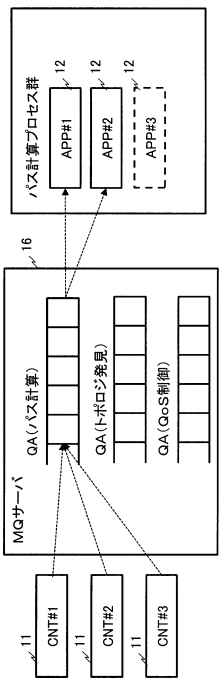
【図９】



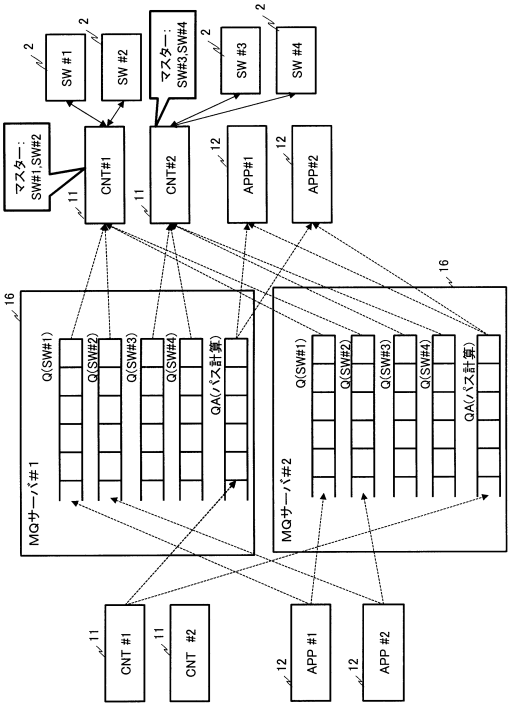
【図１０】



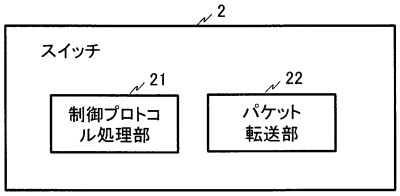
【図１１】



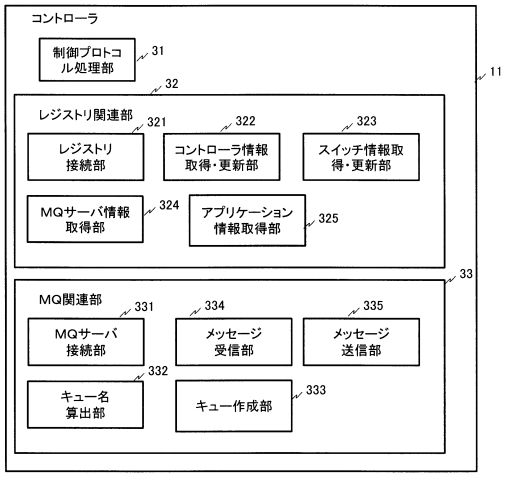
【図１２】



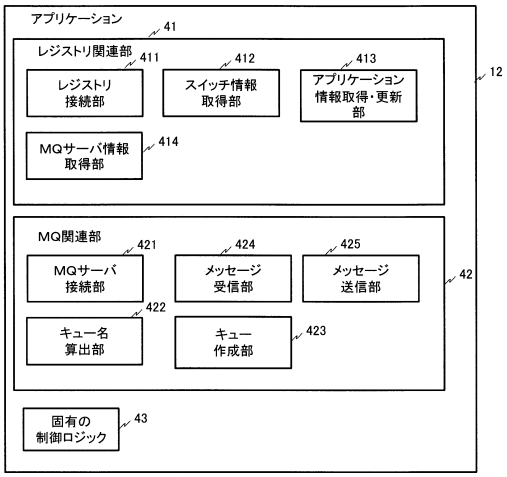
【図１３】



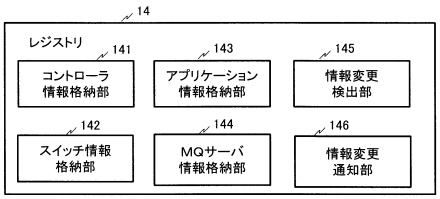
【図１４】



【図１５】



【図１６】



【図１７】

141

コントローラID	マスタ	スレーブ	ｴﾗｰﾌﾗｸﾞ
CNT＃1	SW#1, SW#2		
CNT＃2	SW#3, SW#4, SW#5	SW#6	
CNT＃3	SW#6		

【図１８】

142

スイッチID	マスタ	スレーブ	ｴﾗｰﾌﾗｸﾞ
SW＃1	CNT＃1		
SW＃2	CNT＃1		
SW＃3	CNT＃2		
SW＃4	CNT＃2		
SW＃5	CNT＃2		
SW＃6	CNT＃3	CNT＃2	

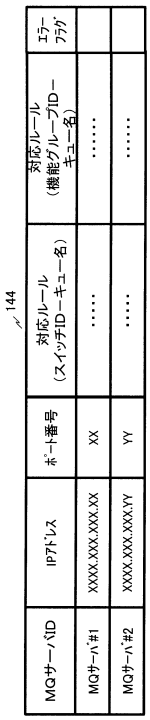
【図１９】

143

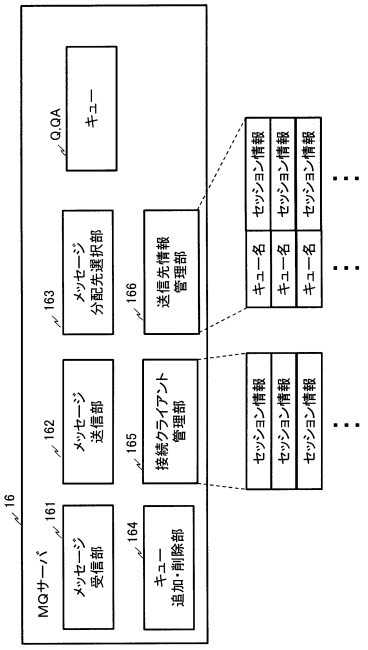
アプリケーションID	機能（機能ｸﾞﾙｰﾌﾟID）	ｴﾗｰﾌﾗｸﾞ
APP＃1	パス計算	
APP＃2	パス計算	
APP＃3	パス計算	
APP＃4	トポロジ発見	
APP＃5	トポロジ発見	
APP＃6	QoS制御	

⋮ ⋮ ⋮

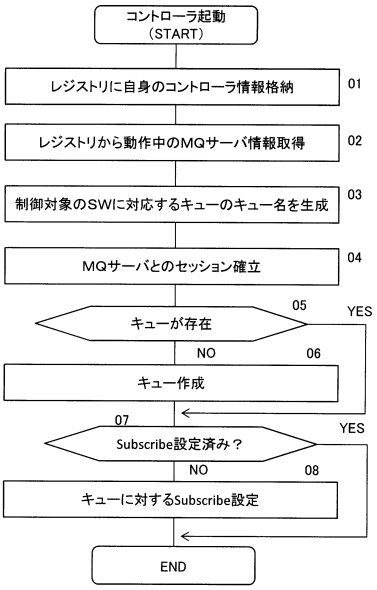
【図２０】



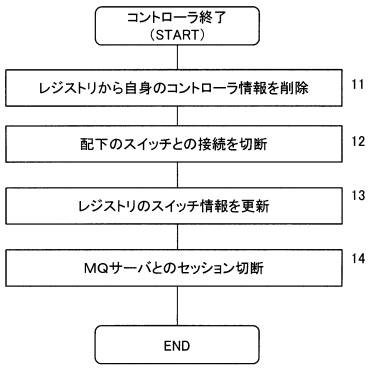
【図２１】



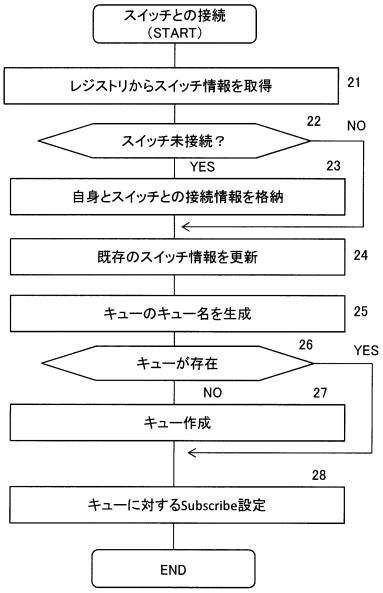
【図２２】



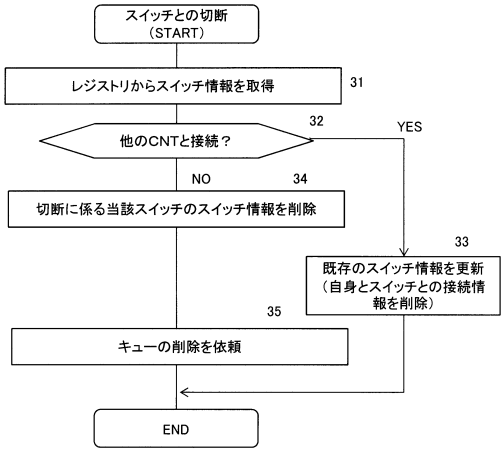
【図２３】



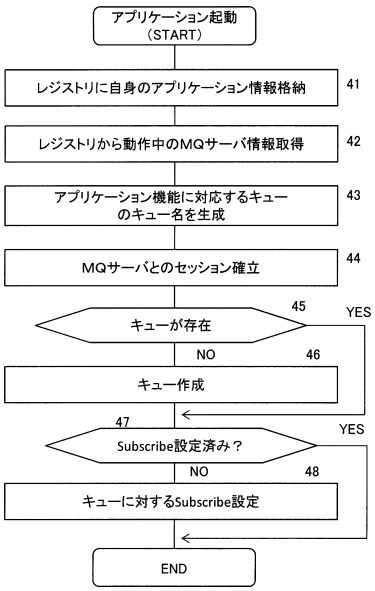
【図２４】



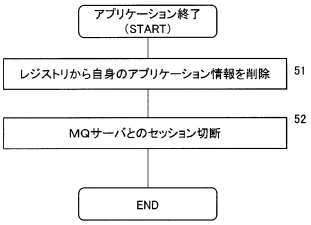
【図２５】



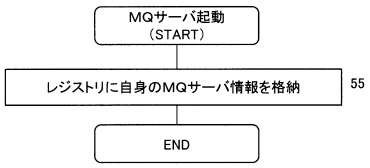
【図２６】



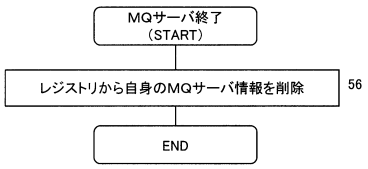
【図２７】



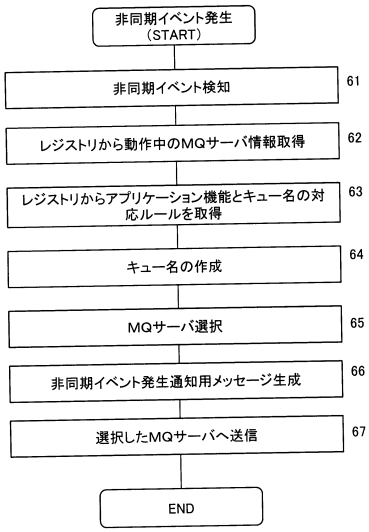
【図２８】



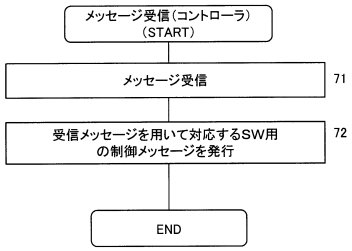
【図２９】



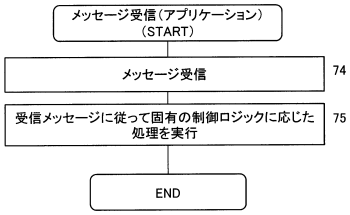
【図３０】



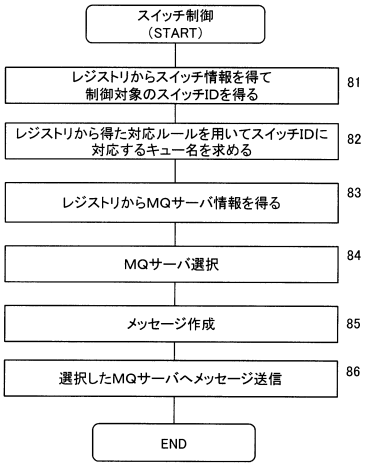
【図３１】



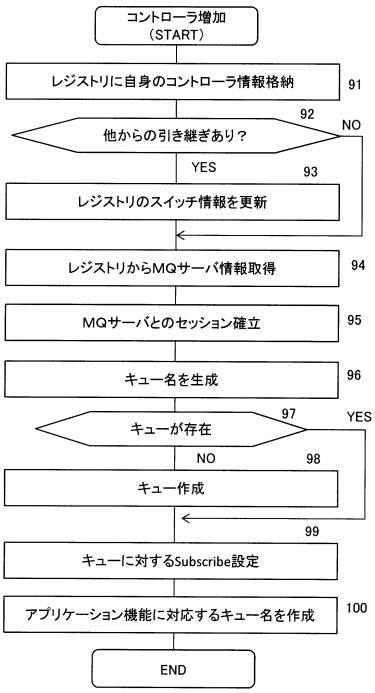
【図３２】



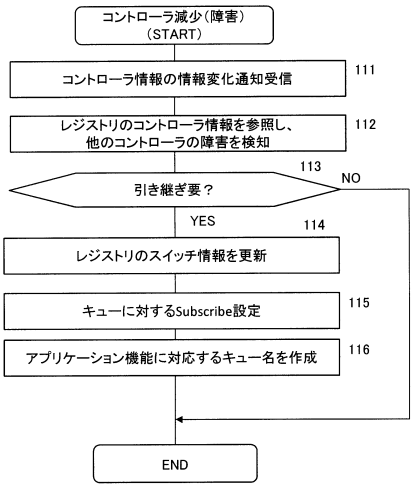
【図３３】



【図３４】



【図３５】


コントローラ減少（障害）
（START）
コントローラ情報の情報変化通知受信
111
レジストリのコントローラ情報を参照し、
他のコントローラの障害を検知
112
113
引き継ぎ要？
NO
YES
114
レジストリのスイッチ情報を更新
キューに対するSubscribe設定
115
アプリケーション機能に対応するキュー名を作成
116
END

フロントページの続き

(56)参考文献 国際公開第２０１３／１３３２２７（WO，Ａ１）
米国特許出願公開第２０１３／０１０３８１７（ＵＳ，Ａ１）
国際公開第２０１４／０１０７２３（WO，Ａ１）
米国特許出願公開第２００４／００９０９６４（ＵＳ，Ａ１）

(58)調査した分野(Int.Cl.，ＤＢ名)
Ｇ０６Ｆ　　９／５４
Ｈ０４Ｌ　１２／７０