

(19) 日本国特許庁 (JP)

(12) 特 許 公 報 (B2)

(11) 特許番号

特許第5097620号
(P5097620)

(45) 発行日 平成24年12月12日 (2012.12.12)

(24) 登録日 平成24年9月28日 (2012.9.28)

(51) Int.Cl.	F I
HO 4W 48/18 (2009.01)	HO 4 Q 7/00 4 1 5
HO 4W 88/06 (2009.01)	HO 4 Q 7/00 6 5 3
HO 4W 40/12 (2009.01)	HO 4 Q 7/00 3 4 9
HO 4W 48/16 (2009.01)	HO 4 Q 7/00 4 0 5

請求項の数 5 (全 23 頁)

(21) 出願番号	特願2008-145266 (P2008-145266)	(73) 特許権者	000005108
(22) 出願日	平成20年6月3日 (2008.6.3)		株式会社日立製作所
(65) 公開番号	特開2009-296084 (P2009-296084A)		東京都千代田区丸の内一丁目6番6号
(43) 公開日	平成21年12月17日 (2009.12.17)	(74) 代理人	100100310
審査請求日	平成22年8月11日 (2010.8.11)		弁理士 井上 学
		(74) 代理人	100098660
			弁理士 戸田 裕二
		(72) 発明者	門田 和也
			神奈川県川崎市麻生区王禅寺1099番地
			株式会社日立製作所システム開発研究所
			内
		(72) 発明者	佐藤 弘起
			神奈川県川崎市麻生区王禅寺1099番地
			株式会社日立製作所システム開発研究所
			内

最終頁に続く

(54) 【発明の名称】 マルチパス通信システム

(57) 【特許請求の範囲】

【請求項1】

ネットワークを介して接続される端末とサーバとを含んで構成されるマルチパス通信システムであって、

前記端末は、それぞれが異なるネットワークアドレスが割り当てられている複数のネットワークインターフェース装置を備え、

前記サーバは、一つのネットワークインターフェース装置を備え、

前記端末は、プロセッサにより実行される、オペレーティングシステムと、アプリケーションプログラムと、前記複数のネットワークインターフェース装置用のデバイスドライバと、前記オペレーティングシステムと前記デバイスドライバとを接続する仮想インターフェースプログラムと、を備え、

前記オペレーティングシステムは、前記アプリケーションプログラムが扱うデータと、前記ネットワーク上での送受信に適したIPパケットとの変換を行うIP層処理プログラムを備え、

前記仮想インターフェースプログラムは、前記IP層処理プログラムと前記デバイスドライバとの間でやり取りされるIPパケットの中継処理と、前記IP層処理プログラムに対する一つのデバイスドライバとしての、一つのIP層処理インターフェースと、前記デバイスドライバに対するデバイスドライバインターフェースと、セッション毎に、前記一つのIP層処理インターフェースから受信した複数のIPパケットについて、前記複数のネットワークインターフェース装置のいずれかから送信されるように振り分け処理を行い、前記デバイ

10

20

スドライバへ送信する処理と、前記複数のネットワークインターフェース装置が受信したIPパケットを、前記一つのIP層処理インターフェースから前記IP層処理プログラムへ送信する処理と、を実現し、

前記端末と前記サーバは、各々が備えるプロセッサにより実行されるマルチパス制御プログラムを備え、

前記端末と前記サーバの前記マルチパス制御プログラムは、互いに、通信相手となる装置がマルチパス通信に対応しているか、および/または、前記複数のネットワークアドレスを利用するマルチパス通信が可能かを調べるためのメッセージ交換を行い、

通信相手装置毎に、セッションが確立されていないセッション未確立の状態、通信相手装置に対しセッションの確立を開始しているがまだ完了していないセッション確立中の状態、通信相手装置との間で、セッション確立済の状態、通信相手装置との間に通信パスが存在しない、または、通信相手装置がマルチパス通信に対応していない状態である不可の状態、を管理し、セッション未確立の状態または、セッション確立済みの状態で所定の時間が経過した場合に、前記サーバの前記マルチパス制御プログラムを通信相手とするメッセージ交換を行い、パス追加処理またはパス削除処理を行い、

前記デバイスドライバを用いて、前記サーバの前記マルチパス制御プログラムへのセッション開始要求メッセージの送信と、前記サーバの前記マルチパス制御プログラムからのセッション開始応答メッセージの受信と、を行い、ラウンドトリップタイムの計算と、を行い、最もラウンドトリップタイムの小さいパスを利用するセッションをプライマリパスとする

ことを特徴とするマルチパス通信システム。

【請求項 2】

請求項 1 に記載のマルチパス通信システムであって、

前記マルチパス制御プログラムは、さらに、前記複数のネットワークインターフェース装置への振り分け率を算出し、

前記仮想インターフェースプログラムは、

セッション毎に、前記一つのIP層処理インターフェースから受信した複数のIPパケットを、算出された前記振り分け率に従い、前記複数のネットワークインターフェース装置への前記振り分け処理を行うことを特徴とするマルチパス通信システム。

【請求項 3】

請求項 2 に記載のマルチパス通信システムであって、

前記仮想インターフェースプログラムは、

UDPパケットのペイロード部分に前記IP層処理インターフェースから受信したIPパケットを格納することにより、カプセル化を行うことを特徴とするマルチパス通信システム。

【請求項 4】

請求項 2 に記載のマルチパス通信システムであって、

前記端末の前記マルチパス制御プログラムは、

通信相手となる装置が備える前記仮想インターフェースプログラムに割り当てられている前記ネットワークアドレスと、

パスごとに当該端末内で一意に定めるパスIDと、

自装置と通信相手となる装置とが備えるネットワークインターフェース装置に割り当てられているネットワークアドレスと、

前記複数のパスへの前記振り分け率と、を含むパス管理テーブルを管理することを特徴とするマルチパス通信システム。

【請求項 5】

請求項 4 に記載のマルチパス通信システムであって、

前記端末と前記サーバの前記仮想インターフェースは、前記デバイスドライバが受信したデータパケットが、前記カプセル化が行われたものか否かを調べ、

前記カプセル化が行われていれば、前記IPパケットを取り出し、

前記カプセル化が行われていなければ、前記データパケットの宛先を、受信した当該端

10

20

30

40

50

末もしくは当該サーバを前記ネットワークで一意に特定する前記ネットワークアドレスに変換することを特徴とするマルチパス通信システム。

【発明の詳細な説明】

【技術分野】

【0001】

本発明は、複数のネットワークに接続されている通信装置において、複数の通信パスを利用する通信方法に関する。

【背景技術】

【0002】

無線ネットワーク技術の発展に伴い、通信装置において携帯電話、PHS、無線LANなど複数のネットワークを並行して利用することが可能となってきた。しかしながら、移動を前提としたモバイル端末のような通信装置において、複数のネットワークに接続し通信先との間に複数の通信パスが存在していても、アプリケーション層で使用する通信パスは、利用者が選択したいいずれかのネットワークを経由する一つの通信パスに限られていた。これに対し、複数の通信パスを活用して通信を行うマルチパス通信を行っても、アプリケーション層には従来どおり一つの通信として提供するために、トランスポート層で複数の通信をまとめる技術がある（例えば特許文献1参照）。

【0003】

【特許文献1】特開2001-60956号公報

【発明の開示】

【発明が解決しようとする課題】

【0004】

特許文献1に記載の技術により、複数のネットワークに接続している通信装置において、通信先との通信において複数の通信パスを使用してもアプリケーション層には一つの通信として通信することが可能となる。しかしながら、トランスポート層で各々の通信を束ねるため、オペレーティングシステムに組み込まれているTCP (Transmission Control Protocol) やUDP (User Datagram Protocol) といったトランスポート層のプログラムを変更する必要がある、また、トランスポート層を使用しないICMP (Internet Control Message Protocol) といった通信が考慮されていない。

【0005】

また、通信装置が移動することが考慮されておらず、接続しているネットワークを変更し通信を行っているIPアドレスが変更された場合のアプリケーション層の通信継続方法について考慮されていない。

【課題を解決するための手段】

【0006】

本発明は、トランスポート層の変更を行うことなく、複数の通信パスを併用するマルチパス通信を可能とする通信装置を提供する。

【0007】

より具体的には、トランスポート層の変更を行うことなく、アプリケーション層が、マルチパス通信を一つの通信として処理可能な通信装置を提供する。

【0008】

また、通信装置が移動するなどして接続するネットワークが変更となっても、アプリケーション層が、継続して通信可能な通信装置を提供する。

【0009】

本発明は、より具体的には、送信側、受信側ともに通信装置において、複数のネットワークインターフェース装置とそれらのデバイスドライバとを、IP層処理部には1つに見せる仮想インターフェースを備え、アプリケーション層によるデータパケットの送受信は仮想インターフェースを経由して行う。

【0010】

更に具体的には、それぞれが異なるネットワークアドレスが割り当てられている複数の

10

20

30

40

50

ネットワークインターフェース装置を備える端末は、

プロセッサにより実行される、オペレーティングシステムと、アプリケーションプログラムと、複数のネットワークインターフェース装置用のデバイスドライバと、オペレーティングシステムとデバイスドライバとを接続する仮想インターフェースプログラムと、を備え、

オペレーティングシステムは、アプリケーションプログラムが扱うデータと、ネットワーク上での送受信に適したIPパケットとの変換を行うIP層処理プログラムを備え、

仮想インターフェースを実現するプログラムは、

IP層処理プログラムとデバイスドライバとの間でやり取りされるIPパケットの中継処理と、

10

IP層処理プログラムに対する一つのデバイスドライバとしてのIP層処理インターフェースと、

デバイスドライバに対するデバイスドライバインターフェースと、

セッション毎に、一つのIP層処理インターフェースから受信した複数のIPパケットについて、複数のネットワークインターフェース装置のいずれかから送信されるように振り分け処理を行い、デバイスドライバへ送信する処理と、

複数のネットワークインターフェース装置が受信したIPパケットを、一つのIP層インターフェースからIP層処理プログラムへ送信する処理と、を実現する。

【0011】

さらに、端末とサーバは、各々が備えるプロセッサにより実行されるマルチパス制御プログラムを備え、

20

端末とサーバのマルチパス制御プログラムは、互いに、通信相手となる装置がマルチパス通信に対応しているか、および/または、複数のネットワークアドレスを利用するマルチパス通信が可能かを調べるためのメッセージ交換を行う。

【0012】

すなわち、送信時、仮想インターフェースは、IP層処理部からのデータパケットを、複数のネットワークインターフェース装置のいずれかから送信されるように振り分け、受信時は複数のネットワークインターフェース装置で受信したデータパケットを、一つのネットワークインターフェース装置で受信したようにまとめてIP層処理部に渡す。これにより、IP層処理部以上に変更を加えなくても、一つの通信セッションに関して、複数の通信パスを使用しアプリケーションには一つの通信として提供することが可能となる。

30

【0013】

また、通信先の通信装置がマルチパス通信に対応しているかどうか、および/または、複数の通信パスのうちどの通信パスで通信することが可能かを調べるために、アプリケーションの通信とは別に通信装置間でマルチパス通信可能かを各パスで調べるメッセージや、接続しているネットワークの変更を伝達するメッセージを交換する。このようなメッセージ交換により状態情報を共有することで、IPアドレスが変更された場合でもアプリケーション層の通信を継続することが可能となる。

【発明の効果】

【0014】

40

本発明によれば、トランスポート層の変更を行うことなく、複数の通信パスを使用したマルチパス通信を行うことが可能になる。

【0015】

また、通信装置が移動するなどして接続するネットワークが変更となっても、マルチパス通信を継続することが可能となる。

【発明を実施するための最良の形態】

【0016】

以下、本発明の実施形態を説明する。本実施形態では、複数のネットワークに接続する、移動によりIPアドレスが変わる可能のある通信装置を端末107と呼び、IPアドレスが固定的な通信装置をサーバ108と呼ぶ。

50

【0017】

図1は本実施形態におけるシステム構成と、そのプログラム構成を例示している。

【0018】

端末107にはアプリケーションプログラムとしてクライアントアプリケーション102とマルチパス制御117が搭載されており、端末107を制御するためのオペレーティングシステム121には、アプリケーションプログラムから出されたデータにTCP (Transmission Control Protocol) もしくはUDP (User Datagram Protocol) といったトランスポートプロトコルの処理を行うトランスポート層処理部103と、データをIPパケット化するIP層処理部119が搭載されている。

【0019】

また、端末107は、物理的にネットワークに接続するネットワークインターフェース装置を3つ備え、さらに、それぞれのネットワークインターフェース装置の制御を行うデバイスドライバを備える。すなわち、デバイスドライバA109はネットワークA112 (例えば携帯電話ネットワーク)、デバイスドライバB110はネットワークB113 (例えばPHSネットワーク)、デバイスドライバC111はネットワークC114 (例えば無線LANネットワーク)、への接続を制御する。これらのネットワークA112、B113、C114は、ネットワークD101 (例えばインターネット) に接続されている。また、デバイスドライバ109には、ネットワークA112のIPアドレスI/F Aが、デバイスドライバ110には、ネットワークB113のIPアドレスI/F Bが、デバイスドライバ111には、ネットワークC114のIPアドレスI/F Cが、それぞれ割り当てられているものとする。

【0020】

なお、ネットワークインターフェース装置数は3つに限定されず、複数であれば本発明は適用できる。また、一つのデバイスドライバで、上述のように一つのネットワークインターフェース装置を制御してもよいし、同じ通信媒体を扱う複数のネットワークインターフェース装置を制御しても良いし、異なる通信媒体を扱う複数のネットワークインターフェース装置を制御しても良い。

【0021】

本実施形態では、実際にネットワークに接続されているネットワークインターフェース装置を制御するデバイスドライバA109、B110、C111と、IP処理部119と、を接続するプログラムを仮想インターフェース104として実装する。仮想インターフェース104には端末107を一意に特定可能なIPアドレスであるコグニティブIP Aが割り当てる。コグニティブIP Aは端末107が接続しているネットワークに属している必要はない。IPアドレスI/F A、IPアドレスI/F B、IPアドレスI/F Cは、ネットワークに属するIPアドレスであり、ネットワークの割り当てポリシー (例えばDHCP) で決まるのでネットワークに接続する度に変更になる可能性があるが、コグニティブIPはネットワークの接続に関係なく固定 (少なくとも通信を行っている間に変更されない) アドレスである。以降、仮想インターフェース104に割り当てられているIPアドレスをコグニティブIPアドレスと呼ぶ。

【0022】

サーバ108にはアプリケーションプログラムとしてサーバアプリケーション105と、マルチパス制御118と、端末107と同様のオペレーティングシステム122が搭載されており、オペレーティングシステム122はトランスポート層処理部116、IP層処理部120を搭載している。サーバ108は物理的なネットワークD101 (例えばインターネット) に接続する一つのネットワークインターフェース装置と、そのデバイスドライバD115と備える。また、デバイスドライバD115にはIPアドレスI/F Dが割り当てられているとする。

【0023】

上記端末107、サーバ108は、CPU (プロセッサとも称する) とメモリ、固定記憶装置、入出力インターフェース、を有する一般的なコンピュータを用いて実現することができる。さらに、以下に説明する、端末107、サーバ108が実現する機能は、CPUが固定記憶装置またはメモリに格納されているプログラムを実行することにより、上記端末107、サーバ108に具現化される。

【0024】

各プログラムは、あらかじめ、上記コンピュータ内の記憶装置に格納されていても良いし、必要なときに、入出力インターフェースと上記コンピュータが利用可能な媒体を介して、他の装置から上記記憶部に導入されてもよい。媒体とは、たとえば、入出力インターフェースに着脱可能な記憶媒体、または通信媒体（すなわちネットワークまたはネットワークを伝搬する搬送波やデジタル信号）を指す。なお、以下の実施形態では、便宜上、プログラムを実行主体として説明する。

【0025】

サーバ108も端末107と同様に、デバイスドライバ115とIP処理部120とを接続するデバイスドライバプログラムを仮想インターフェース106として実装する。仮想インターフェース106には、デバイスドライバD115に割り当てられているIPアドレスI/F Dと同じアドレスが、コグニティブIP Bとして割り当てられているものとする。

10

【0026】

ネットワークA112、ネットワークB113、ネットワークC114もネットワークD101と接続しており、端末107とサーバ108の間には、ネットワークA112もしくはネットワークB113もしくはネットワークC114を経由する3つの通信パスが存在する。端末107が移動する場合、デバイスドライバに割り当てられているIPアドレスが変更になったり、ネットワークと切断されたりすることがある。

【0027】

端末107内のクライアントアプリケーション102と、サーバ108内のサーバアプリケーション105とがデータの交換を行うことで通信を行う。クライアントアプリケーション102からサーバアプリケーション105に向けデータを送信することで、通信が開始することを前提としているが、通信開始後はサーバアプリケーション105からクライアントアプリケーション102へもデータが送られる。

20

【0028】

クライアントアプリケーション102から送信されたデータはTCP、UDPといったトランスポートプロトコルを使用する場合トランスポート層処理部103に渡される。クライアントアプリケーション102がトランスポートプロトコルを使用しないICMP（Internet Control Message Protocol）のような通信を使用する場合は、トランスポート層処理部103を経由せずIP層処理部119に送られる。

30

【0029】

本実施形態ではトランスポートプロトコルを使用した通信を説明するが、本発明は、トランスポートプロトコルを使用しない場合でも、トランスポート層処理部103での処理を行わない、という違いを考慮すれば適用可能であり、トランスポートプロトコルを使用した通信に限定するものではない。トランスポート層処理部103で処理されたデータはIP層処理部119に渡され、IPパケット化される。本実施形態ではクライアントアプリケーション102もしくはサーバアプリケーション105からのデータをIPパケット化したものをデータパケットと呼ぶ。

【0030】

データパケットは仮想インターフェース104に送られ、仮想インターフェース104にてどの通信パスを使って送信を行うか決定され、実際にネットワークを流すことの出来るようにデータパケットのカプセル化が行われる。どの通信パスを使って送信するかは、端末107に、通信パス毎の送信レートを設定しておき、使用可能な通信パスの送信レートに従った割合で決定される。カプセル化されたデータパケットはネットワークを経由しサーバ108へと送られる。

40

【0031】

カプセル化されたデータパケットは、サーバ108のデバイスドライバD115で受信され、仮想インターフェース106へと渡される。仮想インターフェース106ではカプセル化を解き（デカプセル化）、データパケットを取り出す。データパケットはトランスポート層処理部116へと渡され、処理された後、サーバアプリケーション105へ渡されることによって

50

通信が行われる。サーバアプリケーション105からクライアントアプリケーション102へのデータパケットの送信も、同様に行われる。

【0032】

クライアントアプリケーション102から最初のデータが送信される通信開始時において、どのネットワークを経由して通信先であるサーバ108と通信が可能であるか、また、マルチパス通信が可能であるかが不明である。よって、端末107のマルチパス制御117とサーバ108のマルチパス制御118のプログラムが通信を行い、端末107とサーバ108に間のマルチパスセッションの確立可否を調べ、可の場合にマルチパスセッションを確立する。仮想インターフェース104と仮想インターフェース106はマルチパスセッションの状態を把握し、データパケットの送信先ネットワークの決定、送信先ネットワークも合わせたカプセル化を行う。

10

【0033】

端末107のマルチパス制御117は図2の状態遷移フローに従い状態を管理し、サーバ108のマルチパス制御118は図3の状態遷移フローに従い状態を管理する。

【0034】

マルチパスセッションとは、通信を行う通信装置間でマルチパス通信の開始から終了、すなわち、通信装置間でマルチパス通信を行える状態を指す。また、パスとは、通信装置間において、パケットを送受信可能な状態にあるネットワークを指す。複数あるパスのうち、いずれか一つ以上のパスで通信が可能であり、マルチパス通信に対応していることが判明していればマルチパスセッションが確立しているとする。

20

【0035】

マルチパスセッションの状態として、通信相手毎に「未確立」、「確立中」、「確立済」、「不可」の4つの状態を定義する。「未確立」はセッションの確立が行われていない状態であり、通信相手との間に通信パスが存在するか、マルチパス通信に対応しているかが不明な状態（パス管理テーブルには載っていないが、パケットを送受信可能なパスが存在する可能性がある）である。「確立中」は、通信相手に対しセッションの確立を開始しているがまだ完了していない状態であり、通信パスが存在するか、マルチパス通信に対応しているかは不明な状態である。「確立済」は通信相手とマルチパスセッションの確立が完了しており、通信パスが存在し、マルチパス通信に対応している状態である。「不可」は通信相手との間に通信パスが存在しない、もしくは、マルチパス通信に対応していない状態である。なお、パスが存在していても、アプリケーションからの通信がなくなり、所定の時間が経過するとセッションの終了と判断する。また、マルチパスセッションを構成するすべてのパスが存在しなくなった場合にも、終了処理は行わずにセッション終了と判断する。

30

【0036】

上記マルチパスセッションの状態は、図4に例示するセッション管理テーブル400に保持される。列401には通信相手のIPアドレスを記載する。IPアドレスとしてクライアント102が送信要求をするアドレスを記載する。このアドレスは、マルチパス通信に対応している装置、たとえば端末107もしくはサーバ108であればコグニティブIPアドレスとなるが、対応していない通信装置であれば通信相手のデバイスドライバに割り当てられているアドレスとなる。列402にはセッション状態を記載し、列403にはセッション残り時間を記載する。セッション残り時間は確立済であれば最後にデータパケットの送信を行ってから、あらかじめ規定されているセッション保持時間までの残り時間を記載する。セッション状態が不可であれば、あらかじめ規定されているセッション再確立時間までの残り時間を記載する。

40

【0037】

行404に通信相手がサーバA 108である例を記載している。セッション状態は確立済でセッション残り時間は30秒である。行405には通信相手がサーバB 108である場合の例を記載している。セッション状態は不可でセッション再確立時間までは20秒である。

【0038】

50

図2は、端末107のマルチパス制御117の状態遷移フローを説明する。

【0039】

セッション状態は未確立201とするステップ201から始まる。通信開始要求が来るまでステップ202で待機し、通信開始要求が来るとステップ203に進みセッション管理テーブルに当該行を追加し、セッション状態を確立中とし、ステップ204でセッション開始処理を行う。セッション開始処理は、デバイスドライバごとに並列に行う。セッション開始処理の対象となるデバイスドライバは、端末107に実装されており、ネットワークへの接続を制御しているデバイスドライバすべて（図1では、ネットワークA112～C114への接続を制御するデバイスドライバA～C）とする。

【0040】

セッション開始処理では端末107のマルチパス制御117とサーバ108のマルチパス制御118の間でメッセージと呼ぶパケットを交換する。本実施形態ではすべてのメッセージはUDPパケットであるとする。メッセージのポート番号はあらかじめ定められたポート番号を使用するものとする。セッション開始処理では端末107からセッション開始要求メッセージ700をサーバ108に対し送信し、セッション開始要求メッセージ700を受信したサーバ108のマルチパス制御118はセッション開始応答メッセージ900を端末107に返信する。端末107のマルチパス制御117は、セッション開始応答メッセージ900を受信すると、セッション開始要求メッセージ700の送信からセッション開始応答メッセージ900の受信までの時間であるラウンドトリップタイム（以下RTTと記す）を計算し、サーバ108に対しセッション確立メッセージ1000を送信する。サーバ108がセッション確立メッセージ1000を受信すると、セッション開始シーケンスは終了である。

【0041】

端末107のマルチパス制御117のセッション開始処理フローを図5に示す。クライアントアプリケーション102から通信開始要求が来ると、ステップ501で、端末107に装着されているデバイスドライバのうち、セッション開始処理を行っていないデバイスドライバを指定し、セッション開始要求メッセージ700を送信させる。

【0042】

本実施形態では、ネットワークに接続しているすべてのデバイスドライバに対して、セッション開始処理を行うが、かならずしもすべてのデバイスドライバに対して、セッション開始処理を行わせる必要はなく、あらかじめ指定されたデバイスドライバに対してセッション開始処理を行わせてもよい。ステップ501が終了すると、ステップ502に進み、セッション開始応答メッセージ900の受信待ちタイマーをセットする。

【0043】

ここで、図7に、セッション開始要求メッセージ700のパケットフォーマットを例示する。701にメッセージタイプを格納し、702にコグニティブIPアドレスを格納する。703に通信パスごとに端末107内で一意に定めるパスIDを格納し、704には送信に用いるデバイスドライバに従った送信レートを格納する。705には端末107によるセッション開始要求メッセージ700の送信時刻を格納する。送信レート704はマルチパス制御117が保持する図8に示す送信レート設定テーブル800に従い格納する。

【0044】

送信レート設定テーブル800に設定されている値は、マルチパス制御117が起動するときにユーザによりあらかじめ設定されている送信レート設定ファイルを読み込むことにより設定する。送信レート設定テーブル800の801にはデバイスドライバ名を記載し、802には送信レートを記載する。

【0045】

次に、図9に、セッション開始応答メッセージ900のパケットフォーマットを例示する。901にはメッセージタイプとして「セッション開始応答メッセージ」を格納し、902にはサーバ108のコグニティブIPアドレスを格納する。903にはセッション開始要求メッセージ700のパスID703に格納されていたパスIDが格納されている。904には送信時刻1として、セッション開始要求メッセージ700の送信時刻列705に格納されていた端末107からの送信時刻

を格納する。905には送信時刻2として、サーバ108のセッション開始応答メッセージ900の送信時刻が格納されている。

【0046】

図5の説明に戻り、ステップ503ではサーバ108からセッション開始応答メッセージ900を受信したかチェックを行い、受信していない場合はステップ504に進む。ステップ504ではセッション開始応答メッセージ受信待ちタイマーの時間が経過したかチェックを行い、経過していればステップ505に進む。ステップ505では本セッション開始シーケンスの中でセッション開始メッセージの送信回数があらかじめ定められた一定回数以下であるかのチェックを行い、一定回数を越えていけばステップ508に進みセッション開始処理を失敗とし終了する。一定回数を越えていなければステップ506にてセッション開始要求メッセージ700を再送させる。ステップ507ではステップ502と同様にセッション開始応答メッセージ900の受信待ちタイマーをセットし、ステップ503に戻る。ステップ503でセッション開始応答メッセージ900を受信すると、ステップ509に進み、セッション確立メッセージ1000を送信させるデバイスドライバを、セッション開始処理を行っているデバイスドライバに指定し、送信させる。

【0047】

ここで、図10に、セッション確立メッセージ1000のパケットフォーマットを例示する。1001のメッセージタイプにセッション確立メッセージ1000を格納し、1002に端末107のコグニティブIPアドレスを格納する。1003にパスIDを格納し、1004にサーバ108からのセッション開始応答メッセージ900の送信時刻(2)905に格納されていた時刻を格納する。

【0048】

図5の説明に戻り、ステップ510ではサーバ108からのセッション開始応答メッセージ900の送信時刻(1)904に格納されている時刻と端末107の現在時刻を比較することで、セッション開始要求メッセージ700の送信からセッション開始応答メッセージ900の受信にかかった時間を計算し、RTTを計算する。セッション開始応答メッセージ900を受信したサーバ108とのパスは、マルチパス通信が可能なパスとして、ステップ511で、パス管理テーブル1100に追加する。

【0049】

ここで、図11に、端末107のパス管理テーブル1100を例示する。列1101には宛先としてサーバ108のコグニティブIPアドレスを記載する。列1102にはパスIDを格納する。列1103には送信先の物理IPアドレスを記載する。列1104には端末107の送信元アドレスを記載する。列1105にはパスに対する振り分け率を記載する。振り分け率は、パス管理テーブル1100に記載され使用されている通信パスの送信レート802に基づいて計算する。例えば、デバイスドライバAを使用しているパスの振り分け率は以下の式に従って計算できる。

【0050】

「デバイスドライバAへの振り分け率」 = $100 \times \frac{\text{「デバイスドライバAの送信レート」}}{\text{「パスすべての送信レートの和」}}$

図8に例示された送信レートによれば、I/F Aの送信レートは60であり、パスすべての送信レートの和はI/F Aの送信レート60、I/F Bの送信レート30、I/F Cの送信レート10の和であるので100である。よって、I/F Aへの振り分け率は $100 \times 60 / 100 = 60\%$ となる。

【0051】

列1105にはサーバ108とのRTTを記載する。列1107は各パスが、RTTが最短なプライマリパスであるかどうかを記載する。プライマリパスである場合は1、プライマリパスでない場合は0を記載する。列1108にはパス確認タイマーの残り時間を記載する。各行に、パスに関する情報を記載する。例として、セッション開始処理を行っているのが端末107のデバイスドライバAAであり、通信先がサーバA108である場合の行1109を説明する。列1101にサーバA108を記載し、パスIDが例えば1だったとすると列1102にパスID1を記載する。サーバA108のデバイスドライバDのアドレスはサーバ108からのセッション開始応答メッセージ900を含んだパケットのIPヘッダーの送信元アドレスを調べることによりわかる。本実施形態ではサーバ108のコグニティブIPアドレスとデバイスドライバDのIPアドレスは同じで

あるとしているので、サーバA108のデバイスドライバDのIPアドレスもサーバA108である。IPヘッダーの送信元アドレスより取得したサーバA108のデバイスドライバDのアドレスを列1103に記載し、列1104に端末107の送信元のアドレスとしてI/F Aを記載する。ステップ510で計算したRTTは列1106に格納する。列1107には列1109で示されるパスがサーバA108に対するプライマリパスであるかどうかを格納するが、ステップ511の時点ではまだ不明であるので0を記載する。列1108にはパス確認処理を行うまでのタイマーとしてパス確認タイマーの残り時間を記載する。パス確認タイマーの時間はあらかじめ定められた値（例えば30秒）を記載する。

【0052】

図5の説明に戻り、ステップ512ではステップ511で追加したパスがサーバA108に対するパスのうち、もっともRTTが短いかどうかを検索し、もっともRTTが短ければ、ステップ513でプライマリパスに設定し列1107に1を記載する。RTTが最も短くなければ何もしない。以上の処理を持ってセッション開始処理は成功とし（ステップ514）、処理を終了する。

【0053】

図2の説明に戻り、ステップ204でセッション開始処理が終了すると、ステップ205でセッション開始処理が成功したかのチェックを行い、デバイスドライバごとに行ったセッション開始処理のうち、いずれかの処理で成功すればステップ209へ、すべてのセッション開始処理が失敗すればステップ206に進む。失敗した場合、ステップ206にてセッション管理テーブルの当該行のセッション状態を不可とし、ステップ207でセッション再確立タイマーをセッション管理テーブルの当該行に記載する。再確立タイマーの時間はあらかじめ定められている値を記載すればよい。ステップ208でセッション再確立タイマーの時間が経過すると、ステップ201に戻りセッション状態を未確立とする。セッション開始処理が成功すると、ステップ209でセッション管理テーブルの当該行のセッション状態を確立済とし、イベントの発生まで待つ。発生待ちを行うイベントには、パス確認タイマーの時間が経過するステップ211、新しいネットワークへ接続されるなどでパスが追加されるステップ212、ネットワークとの切断が検知されパスが削除されるステップ213がある。ステップ211で、パス管理テーブル1100に記載されているパス確認タイマーが経過すると、ステップ217で、経過したパスに対しパス確認処理を行う。

【0054】

パス確認処理（ステップ217）では、端末107のマルチパス制御117からサーバ108に対しパス確認要求メッセージ1200を送信させる。パス確認要求メッセージ1200を受信したサーバ108のマルチパス制御118はパス応答メッセージを端末107に送信させる。パス確認応答メッセージを受信した端末107のマルチパス制御117はパス確認通知メッセージをサーバ108に送信させる。メッセージを送信するデバイスドライバはパス確認を行うデバイスドライバである。

【0055】

図12に、パス確認要求メッセージ1200のパケットフォーマットを例示する。列1201にメッセージタイプとしてパス確認要求メッセージ1200を格納し、1202に端末107のコグニティブIPアドレスを格納する。1203にパス確認を行うパスIDを格納し、1204にパス確認要求メッセージ1200の送信時刻を格納する。次に、パス確認応答メッセージのパケットフォーマット例は図9に示すフォーマットと同じである。901のメッセージタイプにパス確認応答メッセージを格納し、902にサーバ108のコグニティブIPアドレスを格納する。903にパス確認要求メッセージ1200の1203に格納されていたパスIDを格納する。送信時刻(1) 904にパス確認要求メッセージ1200の1204に格納されていた送信時刻を格納し、送信時刻(2) 905にパス確認応答メッセージの送信時刻を格納する。パス確認通知メッセージのパケットフォーマット例は図10に示すパケットフォーマットと同じである。メッセージタイプ1001にパス確認通知メッセージを格納し、端末107の1002に端末107のコグニティブIPアドレスを格納する。1003にパス確認を行ったパスIDを格納し1004にパス確認応答メッセージ1307の1005に格納されていたパス確認応答メッセージの送信時刻を格納する。

【0056】

端末107のマルチパス制御117によるパス確認処理のフローチャートを図13に示す。ステップ1301でパス確認要求メッセージ1200を送信すると、ステップ1302でパス確認応答メッセージの受信待ちタイマーをセットする。ステップ1303で、パス確認応答メッセージを受信したか否かチェックを行い、パス確認応答メッセージを受信していればステップ1310へ、受信していなければステップ1304へ進む。ステップ1304ではパス確認応答メッセージ受信タイマーが経過したかチェックを行い、経過していなければステップ1303に戻る。経過していればステップ1305で、パス確認要求メッセージ1200の送信回数があらかじめ定められている一定回数以下かチェックを行い、一定回数以下でなければ、パスが切断されているのでステップ1308でパス管理テーブル1100の当該行を削除する。ステップ1309で他にもパスが残っていれば、最もRTTの短い別のパスをプライマリパスとして再設定し処理を終了する。一定回数以下であれば、ステップ1306でパス確認要求メッセージ1200を再送し、ステップ1307でパス確認応答待ちタイマーをステップ1302と同様に設定し、ステップ1303に戻る。

10

【0057】

ステップ1303でパス確認応答メッセージを受信した場合、ステップ1310でサーバ108に向けパス確認通知メッセージを送信し、ステップ1311で、パス確認応答メッセージの送信時刻(1) 1004と現在時刻の比較によりRTTを計算し、当該パス管理テーブル1100の列のRTTを更新する。ステップ1312ではパス管理テーブル1100の当該行のパス確認タイマーを再設定する。ステップ1313で、サーバ108との間でパス確認処理を行ったパスのRTTがもっとも短いかチェックを行い、もっとも短ければステップ1314でプライマリパスに設定する。他に短いパスがあればなにもしない。以上の処理をもって、ステップ1315でパス確認処理は成功とし処理を終了する。

20

【0058】

図2の説明に戻り、ステップ217でのパス確認処理の結果をステップ218でチェックし、パス切断であればステップ215に進む。パス切断でなければステップ209に戻る。ステップ210で発生したイベントがパス追加ステップ212であれば、ステップ216で、追加されたパスによるセッション開始処理を行う。パスが追加される場合には、端末107に新たにデバイスドライバとネットワークインターフェース装置とが装着された場合と、デバイスドライバとネットワークインターフェース装置がこれまで接続していなかったネットワークに接続した場合が考えられる。

30

【0059】

ステップ210で発生したイベントがパス削除ステップ213であれば、ステップ214でパス削除処理を行う。パス削除とは、端末107に装着されていたデバイスドライバが取り外されたり、接続していたネットワークが無線であれば圏外になったりなどしてネットワークが切断されたことを検知した場合に行う処理である。パス削除処理では、パス削除を検知した端末107のマルチパス制御117はパス削除要求メッセージをサーバ108にプライマリパスを用いて送信する。パス削除要求メッセージを受信したサーバ108のマルチパス制御118はパス削除応答メッセージを端末107に返信する。経路削除処理で使用するメッセージはプライマリパスとなっているデバイスドライバを送信先として指定し送信する。

40

【0060】

図14に、パス削除要求メッセージとパス削除応答メッセージ1400のパケットフォーマットを例示する。メッセージタイプ1401にそれぞれ、「パス削除要求メッセージ」または、「パス削除応答メッセージ」を格納する。コグニティブIPアドレス1402にはパス削除要求メッセージであれば端末107のコグニティブIPアドレスをパス削除応答メッセージであればサーバ108のコグニティブIPアドレスを格納し、パスID1403には削除を行うパスのIDを格納する。

【0061】

図15に、端末107のマルチパス制御117のパス削除処理フローチャートを例示する。ステップ1501でパス管理テーブル1100から当該パスの行を削除し、ステップ1502で同一通信先に別パスが存在するかのチェックを行う。別パスが存在していなければ、処理を終了する

50

。別パスが存在した場合、ステップ1503で、削除を行ったパスがプライマリパスであるかチェックし、プライマリパスでなければステップ1505に進む。プライマリパスであればステップ1504で、残っているパスでもっともRTTの短いパスをプライマリパスに設定する。

【0062】

ステップ1505ではパス削除要求メッセージ1400をプライマリパスより送信を行う。送信を行う送信先は、パス管理テーブル1100の宛先に記載されている通信先の中で削除したパスで通信可能であった通信先すべてに対して行う。ステップ1506でパス削除応答メッセージ1400の受信待ちタイマーをセットする。タイマーの時間は予め定められている値とする。ステップ1507ではパス削除応答メッセージ1400を受信したかチェックを行い、受信すれば処理を終了する。

10

【0063】

受信していなければ、ステップ1508でパス削除応答メッセージ受信待ちタイマーが経過したかチェックを行い、経過していなければステップ1507に戻る。経過していればステップ1509で、パス削除要求の送信回数があらかじめ定められた一定数以下かのチェックを行い、一定数以下でなければ処理を終了する。一定数以下であれば、ステップ1510でパス削除要求メッセージ1400の再送を行い、ステップ1511で、パス削除応答メッセージ受信待ちタイマーをステップ1506と同様にセットしステップ1507に戻る。以上が端末107のパス削除処理である。

【0064】

図2の説明に戻り、パス削除処理ステップ214を終了すると、ステップ215でパス管理テーブル1100をチェックし、セッション管理テーブル400においてセッション状態が確立済である通信先に対し、パスが存在しているかのチェックを行う。パスが存在していなければ、セッション管理テーブル400から当該行を削除し、ステップ201に戻る。パスが存在していれば、ステップ209に戻る。

20

【0065】

ステップ210で発生したイベントがステップ211～ステップ213のいずれでもない場合、セッション管理テーブル400のセッション残り時間が経過もしくはセッション終了要求メッセージを受信したイベントであり、ステップ219でセッション終了処理を行う。セッション終了処理では、端末107のマルチパス制御117は、サーバ108に対しセッション終了要求メッセージを送信する。サーバ108のマルチパス制御118はセッション終了要求メッセージを受信するとセッション終了応答メッセージを端末107に返信する。端末107のマルチパス制御117は、セッション終了応答メッセージを受信すると、セッション終了確認メッセージ1600をサーバ108に送信する。セッション終了処理で使用するメッセージは、プライマリパスとなっているデバイスドライバを送信先として指定し送信する。

30

【0066】

図16に、セッション終了確認メッセージのパケットフォーマットを例示する。メッセージタイプ1601に各メッセージのタイプを格納し、コグニティブIPアドレス1602にメッセージの送信元のコグニティブIPアドレスを格納する。

【0067】

次に、図17に、端末107のマルチパス制御117とサーバ108のマルチパス制御118のセッション終了処理のフローチャートを例示する。ステップ1701で、セッション終了タイマーの経過による終了処理の開始か、セッション終了要求メッセージ1600の受信による開始かをチェックする。セッション終了要求受信であれば、ステップ1710に進む。セッション終了タイマーの経過であれば、ステップ1702に進み、セッション終了要求メッセージ1600をプライマリパスより送信する。ステップ1703で、セッション終了応答メッセージ受信タイマーをあらかじめ定められた時間にセットする。

40

【0068】

ステップ1704で、セッション終了応答メッセージを受信したかチェックを行い、受信していればステップ1709に進む。受信していなければステップ1705に進みセッション終了応答メッセージ受信待ちタイマーが経過したかチェックする。経過していなければ、ステッ

50

プ1704に戻り、経過していなければステップ1706で、セッション終了要求メッセージの送信回数があらかじめ定められた一定回数以下かのチェックを行う。送信回数が一定回数以下でないなら、ステップ1716に進む。一定回数以下であればステップ1707で、セッション終了要求メッセージを再送し、ステップ1708で、セッション終了応答メッセージ受信待ちタイマーをステップ1703と同様にセットしステップ1704に戻る。ステップ1704でセッション終了応答メッセージを受信すると、ステップ1709でセッション終了確認メッセージ1600をプライマリパスより送信し、ステップ1716に進む。

【0069】

ステップ1701でセッション終了要求を受信した場合、ステップ1710で、セッション終了応答メッセージをプライマリパスよりセッション終了要求を送信してきた送信元に対し送信する。次に、ステップ1711で、セッション終了確認メッセージ1600の受信待ちタイマーをあらかじめ定められている値にセットする。ステップ1712で、セッション終了確認メッセージ1600を受信したかチェックを行い、受信していればステップ1716に進む。受信していなければ、ステップ1713に進み、セッション終了確認メッセージ受信待ちタイマーが経過したかチェックする。経過していれば、ステップ1716に進む。

10

【0070】

経過していなければ、ステップ1714でセッション終了要求メッセージが再送されて来ないかチェックを行い、再送されて来なければ、ステップ1702に戻る。再送してきていれば、ステップ1715でセッション終了応答の再送を行い、ステップ1702に戻る。ステップ1706では経管理テーブルよりセッション終了処理を行った通信先を宛先とするパスすべてを削除し、ステップ1717でセッション管理テーブルより当該通信先のセッションに関する行を削除し、処理を終了する。

20

【0071】

図2の説明に戻り、ステップ219でセッション終了処理が終了すると、ステップ201に戻る。

【0072】

以上の状態遷移フローチャートに従い、端末107のマルチパス制御117は通信先サーバ108のマルチパス制御118とのセッション状態を管理することが出来る。

【0073】

次に、図3を用いて、サーバ108のマルチパス制御118の状態遷移フローを説明する。

30

【0074】

セッション状態を未確立とするステップ301から始め、セッション開始要求が来るまでステップ302で待機し、セッション開始要求が来るとステップ303に進みセッション管理テーブルに当該行を追加し、セッション状態を確立中とし、ステップ304でセッション開始処理を行う。セッション開始処理は、複数並行して行ってもよい。

【0075】

図6に、サーバ108のマルチパス制御118のセッション開始処理のフローチャートを例示する。ステップ601で端末107よりセッション開始要求メッセージ700を受信すると、ステップ602に進み、セッション開始応答メッセージ900を送信する。セッション開始応答メッセージの宛先は、セッション開始要求メッセージの送信元アドレスとする。ステップ603ではセッション確立メッセージ1000の受信待ちタイマーをあらかじめ定められた値にセットし、ステップ604でセッション確立メッセージ1000を受信したかどうかのチェックを行う。受信していればステップ610に進む。受信していなければ、ステップ605に進みセッション確立メッセージ受信待ちタイマーが経過していないかチェックを行う。

40

【0076】

経過していなければステップ604に戻る。経過していれば、ステップ606に進みセッション開始応答メッセージ900の送信回数があらかじめ定められた一定回数以下かのチェックを行う。一定回数以下でなければ、セッション開始処理は失敗とし処理を終了する。一定回数以下であれば、ステップ607でセッション開始応答メッセージ900の再送を行う。ステップ608ではステップ603と同様に、セッション確立メッセージ受信待ちタイマーのセット

50

を行い、ステップ604に戻る。ステップ610では、セッション確立メッセージ1000に格納されているセッション開始応答メッセージ900の送信時刻と現在時刻との比較を行いRTTの計算を行う。ステップ611ではパス管理テーブル1100にセッション開始要求メッセージ700の送信元へのパスを追加する。

【0077】

サーバ108のパス管理テーブル1100も図11に示す端末107のパス管理テーブル1100と同じであるが、列1108にはあらかじめ定められているパス確認の受信待ちタイマーを記載する。パス確認の受信待ちタイマーは、図5のステップ511で記載する端末107のパス確認タイマーの3倍といったように端末107のパス確認タイマーより長めに設定しておく必要がある。また、列1105の振り分け率の計算に使用する各パスの送信レートは図7に示すセッション開始要求メッセージ700に格納されている送信レート704を使用する。ステップ612では、ステップ611で追加したパスがもっともRTTの短いパスであるかチェックを行い、最もRTTの短いパスであればステップ613でプライマリパスに設定し、最もRTTの短いパスでなければ、プライマリパスとはせず、ステップ614に進みセッション開始処理は成功とし処理を終了する。

10

【0078】

図3の説明に戻り、ステップ305にてセッション開始処理の結果をチェックし、すべてのセッション開始処理が失敗となればステップ306に進みセッション管理テーブルのセッション状態を不可とし、図2のステップ207と同様にセッション再確立タイマーを設定する。ステップ308にてセッション再確立タイマーが経過したかチェックを行い、経過すればステップ301に戻る。セッション開始処理のうちひとつでも成功していれば、ステップ309でセッション管理テーブルのセッション状態を確立済とし、ステップ310でイベントの派生を待つ。ステップ310で発生するイベントにはパス確認要求メッセージ1200の受信ステップ311、パス確認受信タイマーの経過ステップ312、パス追加要求メッセージの受信ステップ313、パス削除要求メッセージ1400の受信ステップ314のいずれかである。発生したイベントがパス確認要求メッセージ1200の受信ステップ311であれば、ステップ318に進みパス確認処理を行う。

20

【0079】

図18に、サーバ108のマルチパス制御118のパス確認処理のフローチャートを例示する。ステップ1801でパス確認要求メッセージ1200を受信すると、ステップ1802でパス確認応答メッセージを要求メッセージの送信元に返信する。ステップ1803でパス確認通知メッセージの受信待ちタイマーをあらかじめ定められた値にセットする。ステップ1804でパス確認通知メッセージを受信したかチェックし、受信していればステップ1811に進む。受信していなければステップ1805に進む。ステップ1805ではパス確認通知メッセージ受信待ちタイマーが経過したかチェックを行い、経過してればステップ1809に進みパス切断とし処理を終了する。パス確認通知メッセージ受信待ちタイマーが経過していなければ、ステップ1806で、パス確認要求メッセージ1200が再送されてきていないかチェックを行い、再送を受信していなければステップ1804に戻る。

30

【0080】

受信していれば、ステップ1807でパス確認応答メッセージを再送し、ステップ1808で、ステップ1803と同様にパス確認通知メッセージの受信待ちタイマーをセットしステップ1804に戻る。パス確認通知メッセージを受信したら、ステップ1810で、パス確認通知メッセージに格納されているパス確認応答メッセージの送信時刻と現在時刻を比較しパスのRTTを計算する。ステップ1811ではパス管理テーブル1100のパス確認受信タイマーを再設定する。ステップ1812で、パス確認処理を行ったパスがもっともRTTの短いパスがチェックし、もっとも短いパスであればステップ1813でプライマリパスに設定をし、パス確認処理は成功とし処理を終了する。

40

【0081】

図3の説明に戻り、ステップ319でパスが切断されたかのチェックを行い、パスが切断されていなければステップ309に戻る。パスが切断されていればステップ316に進む。ステッ

50

プ310で発生したイベントがパス確認受信タイマーの経過ステップ312であれば、ステップ316に進みパス削除処理を行う。

【0082】

図19に、サーバ108のパス削除処理のフローチャートを例示する。ステップ1901でパス削除要求メッセージ1400による受信であるかチェックを行い、パス削除要求メッセージ1400の受信であれば、ステップ1902で要求メッセージの送信元にパス削除応答メッセージ1400を返信し、ステップ1903に進む。パス削除要求メッセージ1400の受信でなく、パス確認処理によるパス切断、もしくはパス確認受信タイマーの経過であれば、ステップ1902を実行せずにステップ1903に進む。ステップ1903ではパス管理テーブル1100より当該パスを削除し、ステップ1904に進む。ステップ1904では削除を行ったパスと同一の宛先に別パスがあるかチェックを行い、なければ処理を終了する。別パスがあれば、ステップ1905に進み削除したパスがプライマリパスであったかチェックを行い、プライマリパスでなければ処理を終了する。プライマリパスであったならば、ステップ1906に進み、別パスの中で最もRTTの短いパスをプライマリパスとした後、処理を終了する。

10

【0083】

図3の説明に戻り、パス削除処理の結果、通信可能なパスが残っているかステップ317でチェックを行いパスが残っていれば、ステップ309に戻る。残っていなければ、セッション管理テーブルより当該セッションを削除し、ステップ301に戻る。ステップ310で発生したイベントが、パス追加要求メッセージの受信ステップ313であった場合、ステップ315に進みセッション開始処理を行った後、ステップ309に戻る。ステップ310で発生したイベントがパス削除要求メッセージ1400の受信ステップ314であれば、ステップ316に進む。ステップ310で発生したイベントがステップ311～ステップ314のいずれでもない場合、セッション管理テーブルに記載されたセッション残り時間が経過もしくはセッション終了要求メッセージを受信したイベントであり、ステップ320でセッション終了処理を行う。セッションの終了処理は図17に示すフローチャートと同じである。

20

【0084】

以上の状態遷移フローに従うことで、サーバ108のマルチパス制御118は通信先端末107のマルチパス制御117とのセッション状態管理を行うことが出来る。

【0085】

端末107のクライアントアプリケーション102とサーバ108のサーバアプリケーション105間の通信のデータパケットの送信処理はマルチパス通信のセッション状態に従って行われる。データパケットは仮想インターフェース104もしくは106に渡され、送信処理が行われる。

30

【0086】

図20に、仮想インターフェース104または106による、データパケットとメッセージの送信処理のフローチャートを例示する。ステップ2001でパケットの送信要求を受けると、ステップ2013でメッセージかデータパケットか判別する。メッセージであるかどうかはポート番号により判別可能である。メッセージであれば、送信に用いるデバイスドライバをマルチパス制御117または118が指定したデバイスドライバに決定し、ステップ2008に進む。

【0087】

メッセージでなければ、マルチパス制御117または118が管理しているセッション管理テーブルを参照し、データパケットの宛先とマッチする通信相手を検索する。検索が終了すると、ステップ2002に進む。ステップ2002で、検索の結果、マッチする通信相手がなければセッション状態は未確立であるとしステップ2003に進み、データパケットを破棄する。ステップ2005で、仮想インターフェース104または106はマルチパス制御117または118にデータパケットの宛先に対する通信開始要求を出し、処理を終了する。ステップ2002で、マッチした通信相手のセッション状態が確立済であれば、ステップ2006に進み、以降の振り分け処理を行う。

40

【0088】

振り分け処理として、ステップ2006ではマルチパス制御117または118が管理するパス管

50

理テーブル1100を参照し、宛先にマッチするパスの振り分け率に従い、送信を行うデバイスドライバを決定し、ステップ2007に進む。ステップ2007ではセッション管理テーブルの当該行のセッション残り時間を予め定められた値まで延長し、ステップ2008に進む。ステップ2008ではデータパケットを、送信を行うデバイスドライバから送信するためのカプセル化を行う。本実施形態では、端末107にグローバルIPアドレスが割り当てられていない場合に使用されるNAPT技術に対応するためUDPパケットのペイロード部分にIP層処理部でデータパケット化されたパケットをIPヘッダーごと格納することによるカプセル化を行う。NAPT技術に関してはIETFより公開されている技術文書RFC2663 (<http://www.ietf.org/rfc/rfc2663.txt>)に記載されている。カプセル化後データパケットのIPヘッダーには、デバイスドライバから送信するためのIPヘッダーを従来どおりに格納する。IPヘッダーに格納される送信元アドレスはデバイスドライバに従う。カプセル化後パケットのUDPヘッダーに格納される宛先ポート番号は本通信用に決まった値を用いてもよい。カプセル化後パケットのペイロード部分に格納されている、IP層処理部から受信したデータパケットのIPヘッダーは、仮想インターフェースを送信元とした場合のIPヘッダーであり、宛先アドレスとして宛先のコグニティブIPアドレス、送信元アドレスとして、自装置に割り当てられているコグニティブIPアドレスを格納する。

10

【0089】

データパケットのカプセル化が終了し、振り分け処理が終了すると、ステップ2009に進みデバイスドライバからネットワークに対し送信を行う。送信を行う際のルーティングは端末107もしくはサーバ108のオペレーティングシステムが保持するルーティングテーブルに従い送信を行い、処理を終了する。

20

【0090】

ステップ2005で宛先のセッション状態が確立済でなければ、ステップ2010に進みセッション状態が不可でないかチェックを行う。セッション状態が不可であれば、宛先はマルチパス通信に対応していないノードであるので、仮想インターフェース104または106に渡されたパケットをカプセル化せずに、そのままデバイスドライバから送信を行う。送信を行うデバイスドライバは、端末107もしくはサーバ108のオペレーティングシステムが保持するルーティングテーブルに従って決定される。ステップ2010で宛先のセッション状態が不可でなければ、ステップ2012に進みパケットを破棄して処理を終了する。

【0091】

30

図21に、端末107とサーバ108におけるデータパケットとメッセージの受信処理のフローチャートを例示する。ステップ2101においてデバイスドライバで受信されたパケットはすべて仮想インターフェース104または106に処理を引き渡すと、ステップ2106でメッセージであるかどうかの判定を行い、メッセージであればステップ2105、メッセージでなければステップ2102に進む。メッセージであるかどうかはポート番号により判別可能である。ステップ2102ではデータパケットがマルチパス通信用にカプセル化されているかチェックを行い、カプセル化パケットであればステップ2103に進みデータパケットのデカプセル化を行う。

【0092】

デカプセル化とは、2601のIPヘッダーと2602のUDPヘッダーを取り除き、送信元の仮想インターフェース104または106から送信されたデータパケットとして取り出す処理のことである。カプセル化パケットでなければ、ステップ2104に進み、データパケットの宛先を受信した端末107もしくはサーバ108のコグニティブIPアドレスに変換を行う。変換を行うことにより、マルチパス通信に対応していないノードからのデータパケットもコグニティブIPアドレス宛に送信されているとして処理することが可能となる。処理が行われたデータパケットは、ステップ2105で、端末107もしくはサーバ108のオペレーティングシステムに処理が引き渡され従来と同じ受信処理が行われた後、アプリケーションへと渡される。

40

【0093】

以上により、複数の通信パスを使用したマルチパス通信をトランスポート層処理部の変更を行うことなくアプリケーション層に一つの通信として提供することが可能になる。ま

50

た、通信装置が移動するなどして接続するネットワークが変更となってもアプリケーション層のプログラムは通信を継続することが可能となる。

【図面の簡単な説明】

【0094】

【図1】本実施形態のシステム構成の一例である。

【図2】端末107のマルチパス制御117の状態遷移フロー例である。

【図3】サーバ108のマルチパス制御117の状態遷移フロー例である。

【図4】セッションの管理テーブルの一例である。

【図5】端末107のマルチパス制御117のセッション開始処理フロー例である。

【図6】サーバ108のマルチパス制御117のセッション開始処理フロー例である。

10

【図7】セッション開始要求メッセージ700のパケットフォーマット例である。

【図8】送信レート設定テーブル800の一例である。

【図9】セッション開始応答メッセージ900のパケットフォーマット例である。

【図10】セッション確立メッセージ1000のパケットフォーマット例である。

【図11】パス管理テーブル1100の例である。

【図12】パス確認要求メッセージ1200のパケットフォーマット例である。

【図13】端末107のマルチパス制御117のパス確認処理フロー例である。

【図14】パス削除要求メッセージ、パス削除応答メッセージのパケットフォーマットの例である。

【図15】端末107のマルチパス制御117のパス削除処理フロー例である。

20

【図16】セッション終了処理で使用するメッセージのパケットフォーマット例である。

【図17】サーバ108のマルチパス制御117のパス削除処理フロー例である。

【図18】サーバ108のマルチパス制御117のパス確認処理フロー例である。

【図19】サーバ108のマルチパス制御117のパス削除処理フロー例である。

【図20】仮想インターフェースのデータパケット送信フロー例である

【図21】仮想インターフェースのデータパケット受信フロー例である

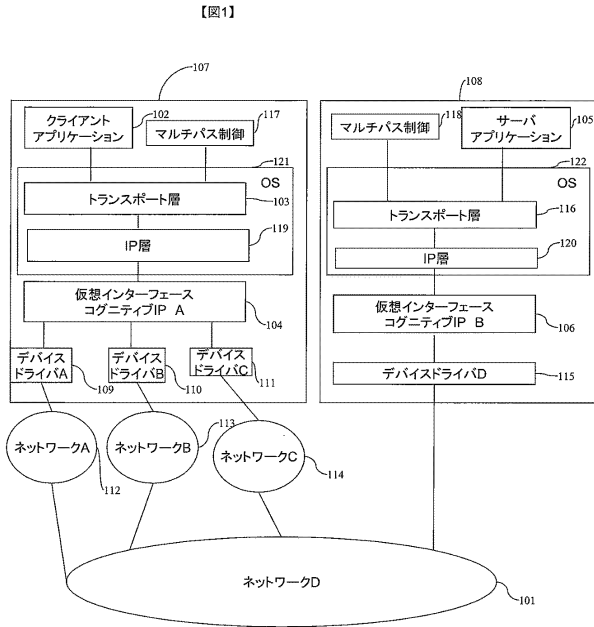
【符号の説明】

【0095】

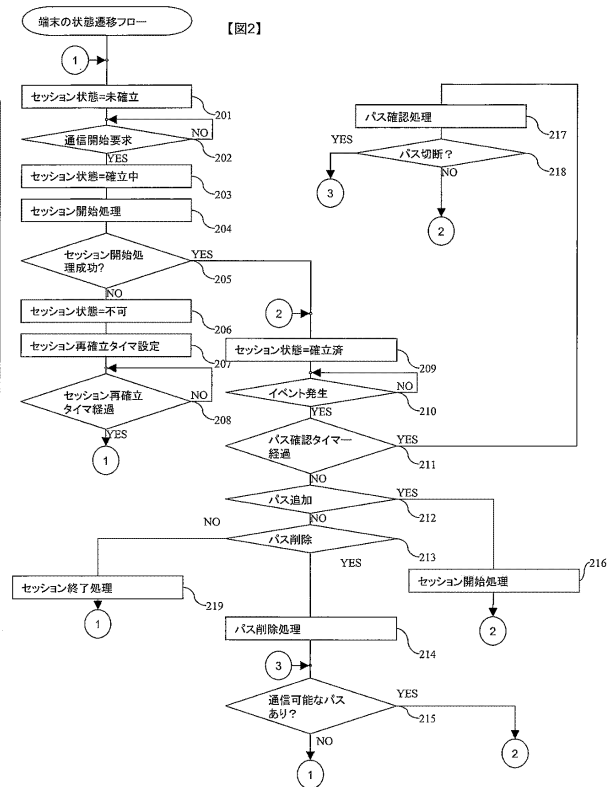
107：端末，108：サーバ，400：セッション管理テーブル，700：セッション管理開始要求メッセージ，900：セッション開始応答メッセージ，1000：セッション確立メッセージ，1100：パス管理テーブル，1200：パス確認要求メッセージ，1400：パス削除要求メッセージ/パス削除応答メッセージ，1600：セッション終了確認メッセージ。

30

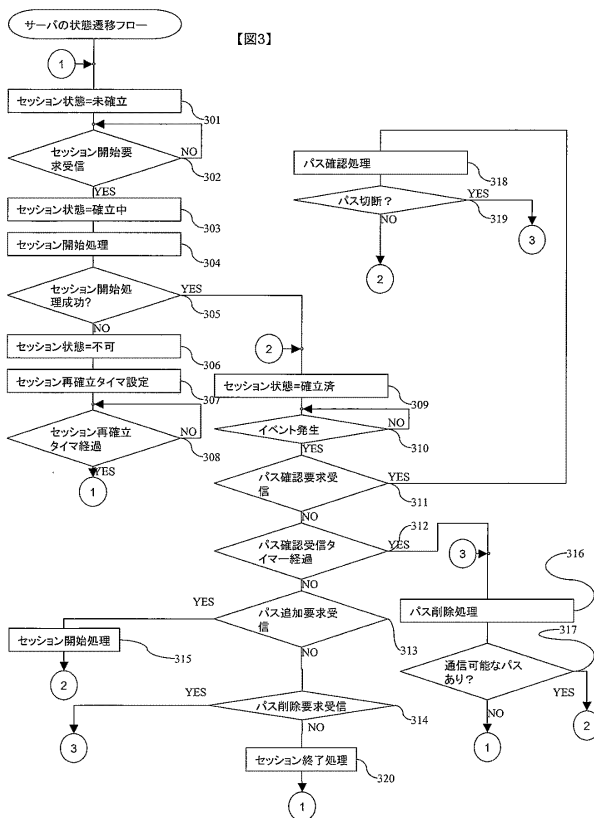
【図 1】



【図 2】



【図 3】

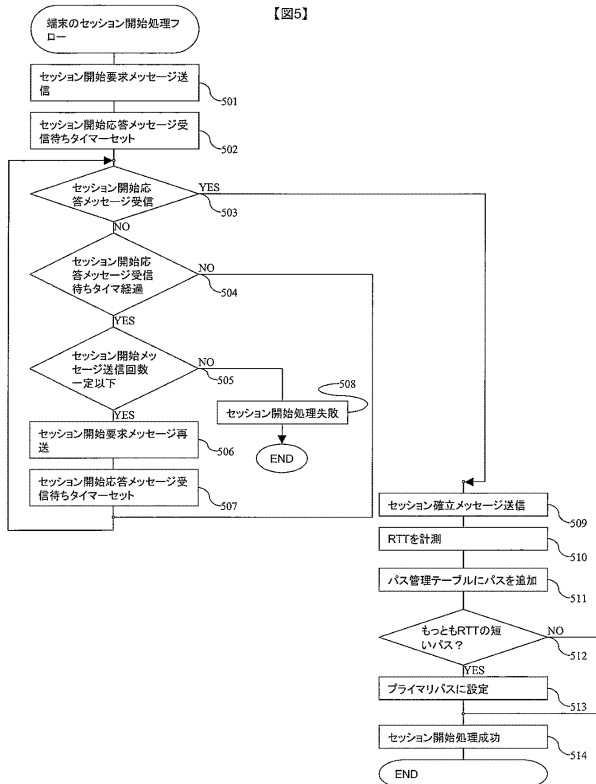


【図 4】

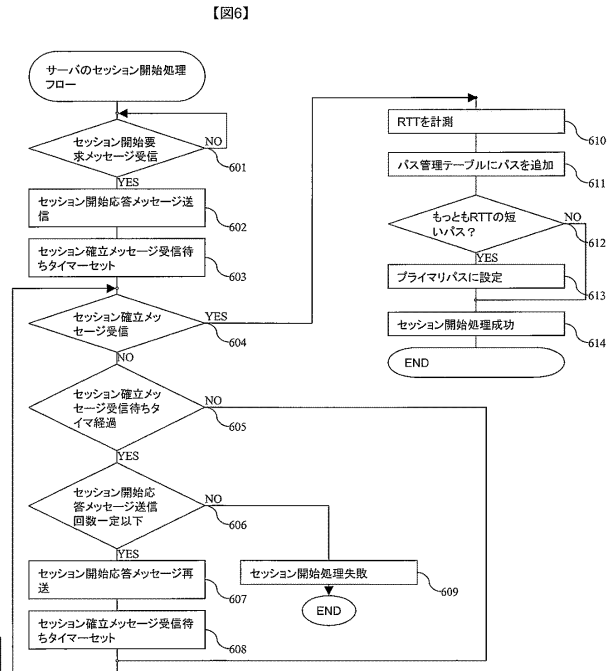
【図4】 セッション管理テーブル 400

401	402	403
通信相手	セッション状態	セッション残り時間
404 サーバA	確立済	30
405 サーバB	不可	20
406 ...	...	...

【図 5】



【図 6】



【図 7】

【図7】

セッション開始要求メッセージ 700

701	702	703	704	705
メッセージタイプ	コグニティブIPアドレス	バスID	送信レート	送信時刻

【図 8】

【図8】

送信レート設定テーブル 800

801	802
物理インターフェース	送信レート
デバイスドライバA	60
デバイスドライバB	30
デバイスドライバC	10

【図 1 1】

【図11】

バス管理テーブル 1100

	1101	1102	1103	1104	1105	1106	1107	1108
	宛先	バスID	送信先	送信元	振り分け率	RTT	プライマリ	バス確認 タイマー
1109	サーバA	1	サーバA	I/F A	60	20	0	30
1110		2	サーバA	I/F B	30	10	1	30
1111		3	サーバA	I/F C	10	30	0	30
1112	サーバB	1	サーバB	I/F A	100	10	1	10
	...	...	...	...	...	...	...	...

【図 1 2】

【図12】

バス確認要求メッセージ1200

1201	1202	1203	1204
メッセージタイプ	コグニティブIPアドレス	バスID	バス確認要求送信時刻

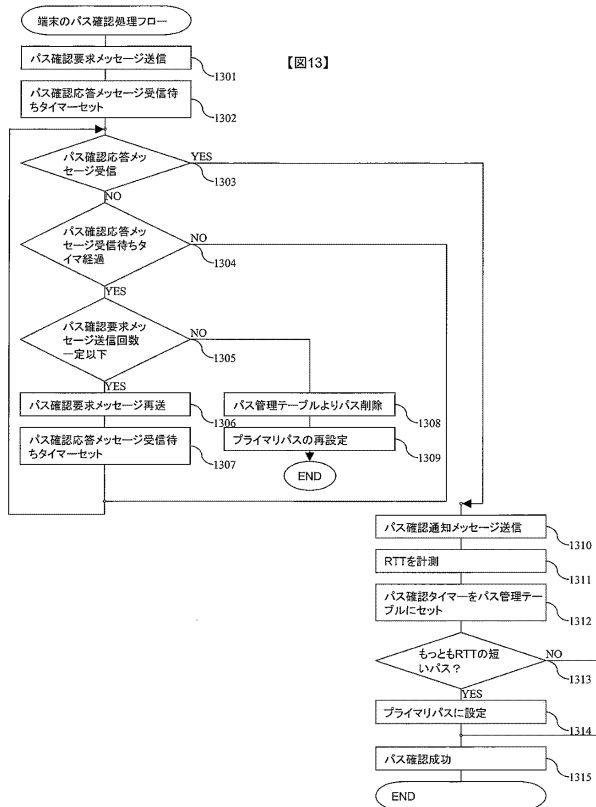
【図 1 0】

【図10】

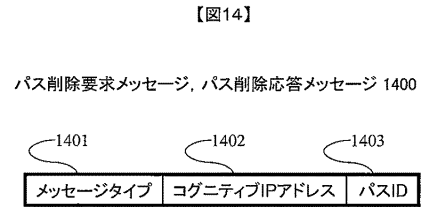
セッション確立メッセージ 1000

1001	1002	1003	1004
メッセージタイプ	コグニティブIPアドレス	バスID	送信時刻

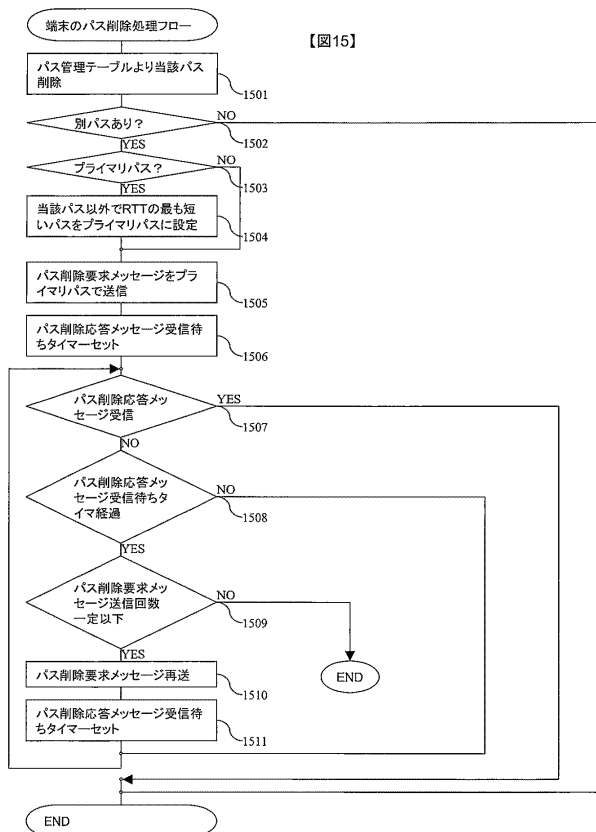
【図 13】



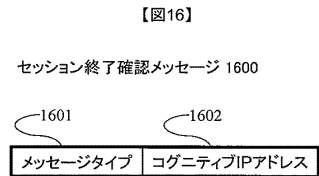
【図 14】



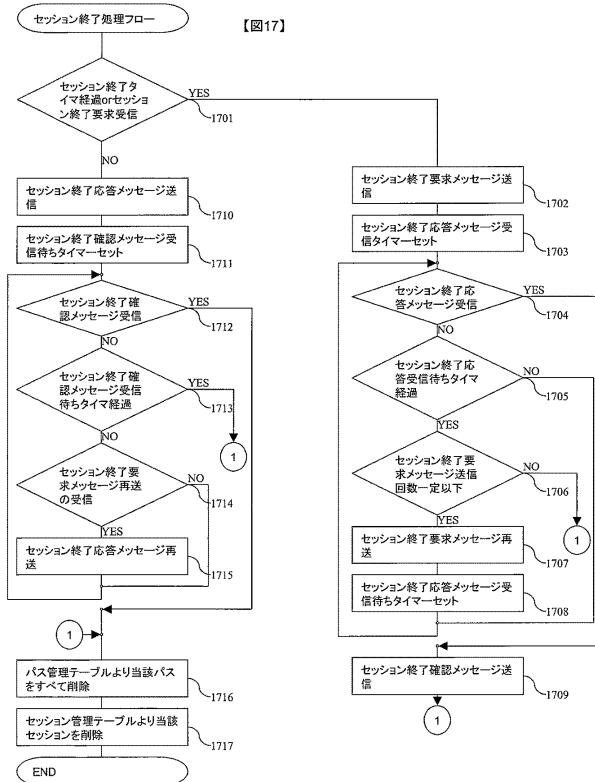
【図 15】



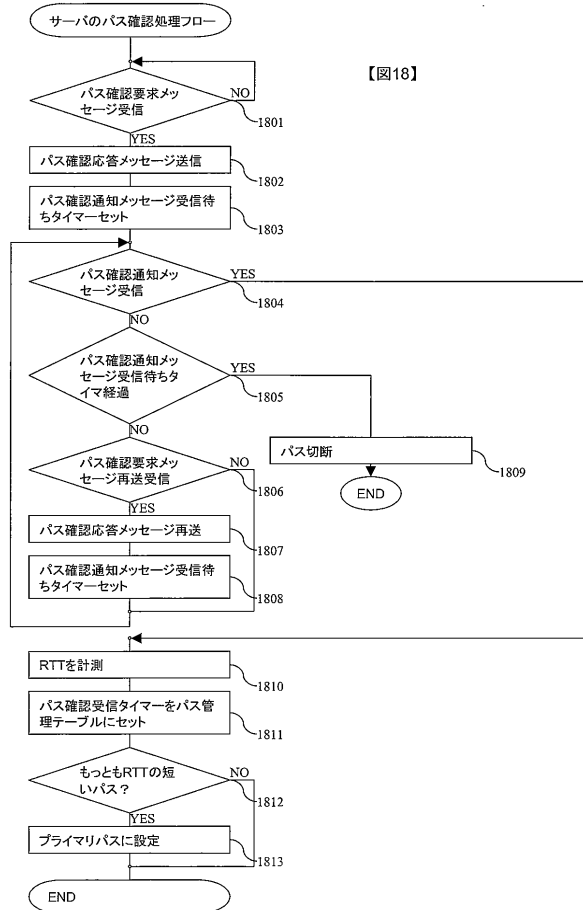
【図 16】



【図 17】

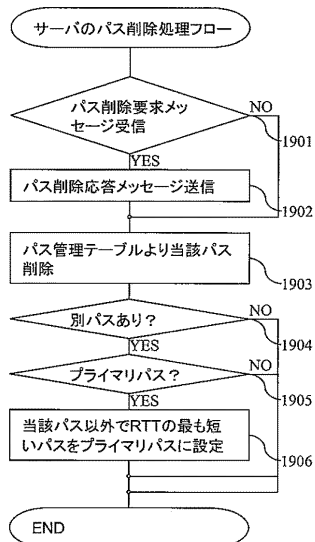


【図 18】



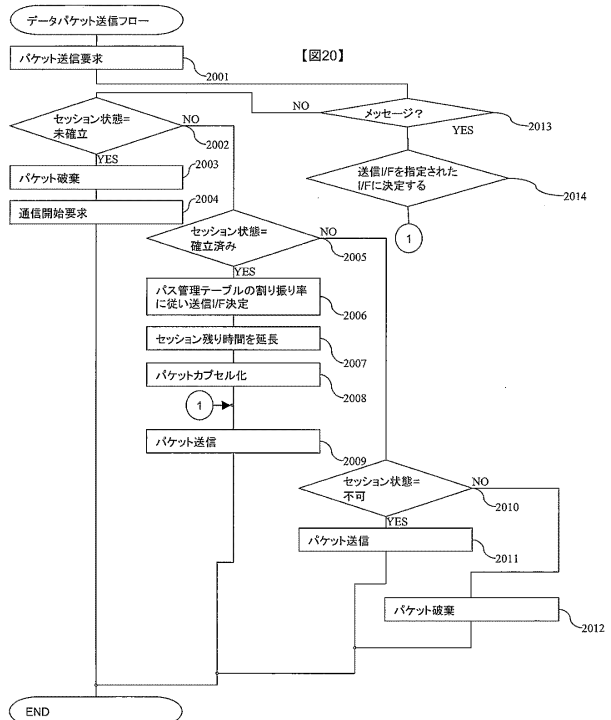
【図 19】

【図19】



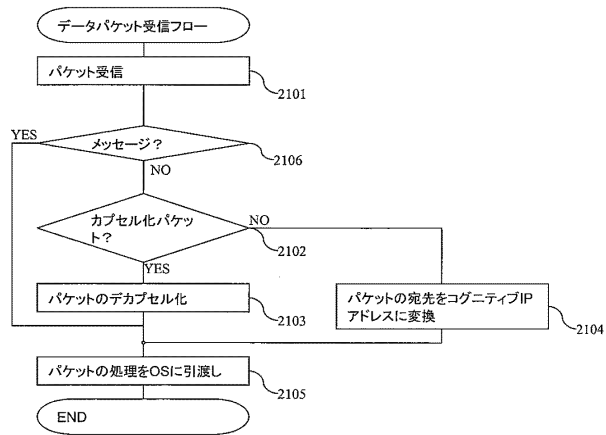
【図 20】

【図20】



【図 21】

【図21】



フロントページの続き

(72)発明者 藤城 孝宏

神奈川県川崎市麻生区王禅寺1099番地 株式会社日立製作所システム開発研究所内

(72)発明者 高谷 幸宏

神奈川県川崎市麻生区王禅寺1099番地 株式会社日立製作所システム開発研究所内

審査官 山中 実

(56)参考文献 特表2007-538422 (JP, A)

特表2007-529127 (JP, A)

特開2007-088857 (JP, A)

特開2004-364343 (JP, A)

特開平11-145990 (JP, A)

(58)調査した分野(Int.Cl., DB名)

H04B 7/24- 7/26

H04W 4/00-99/00