(19)日本国特許庁(JP)　(12)特　許　公　報(B2)　(11)特許番号
特許第3930743号
(P3930743)
(45)発行日　平成19年6月13日(2007.6.13)　(24)登録日　平成19年3月16日(2007.3.16)

(51)Int.Cl.		F I		
H04L 12/56	(2006.01)	H04L	12/56	G
H04L 12/28	(2006.01)	H04L	12/28	200Z

請求項の数 16　(全 17 頁)

(21)出願番号　特願2002-20221(P2002-20221)
(22)出願日　平成14年1月29日(2002.1.29)
(65)公開番号　特開2002-319963(P2002-319963A)
(43)公開日　平成14年10月31日(2002.10.31)
審査請求日　平成16年11月15日(2004.11.15)
(31)優先権主張番号　09/777,609
(32)優先日　平成13年2月6日(2001.2.6)
(33)優先権主張国　米国(US)

(73)特許権者　398038580
ヒューレット・パッカード・カンパニー
HEWLETT-PACKARD COMPANY
アメリカ合衆国カリフォルニア州パロアルト　ハノーバー・ストリート　3000
(74)代理人　100081721
弁理士　岡田　次生
(74)代理人　100105393
弁理士　伏見　直哉
(74)代理人　100111969
弁理士　平野　ゆかり

最終頁に続く

(54)【発明の名称】耐故障性プラットフォームにおいてネットワーク接続を提供する方法

(57)【特許請求の範囲】

【請求項１】

アクティブ状態のプロセス、スタンバイ状態のプロセス、スタンバイ状態のプロセスをアクティブ状態に移行させるための切り換え能力を有する耐故障性プラットフォームにおいてネットワーク接続を提供する方法であって、

該アクティブ状態のプロセスは、ネットワークアドレスを有し、該方法は、

該スタンバイ状態のプロセスをアクティブ状態へ移行させ、該スタンバイ状態のプロセスの移行中に該ネットワークアドレスを該スタンバイ状態のプロセスに転送し、

該スタンバイ状態のプロセスのアクティブ状態への移行に先立って、

該スタンバイ状態のプロセスに、該アクティブ状態のプロセスのネットワーク接続の状態データを複製するステップと、

該スタンバイ状態のプロセスについて、複製されたデータで更新された対応するスタンバイネットワーク接続を維持するステップと、

アクティブ状態への該スタンバイ状態のプロセスの移行中に、該アクティブ状態のプロセス内の該ネットワーク接続を、該ネットワーク上の該接続を閉じることなく非活動化し、該スタンバイ状態のプロセスにネットワークアドレスが転送されたら、該ネットワークアドレスで該対応するスタンバイネットワーク接続を活動化するステップと、を含み、

該移行したスタンバイ状態のプロセスが該ネットワークにおける接続を再開する必要がないようにする、耐故障性プラットフォームにおいてネットワーク接続を提供する方法。

【請求項２】

前記ネットワーク接続が伝送制御プロトコル（ＴＣＰ）接続、トランザクション処理（ＯＳＩ　ＴＰ）接続またはストリーム制御伝送プロトコル（ＳＣＴＰ）接続のいずれかである請求項１に記載の方法。

【請求項３】

前記アクティブプロセスの前記接続を監視するステップと、該接続のアイドル状態が識別されたことに応答して前記状態データの複製を起動するステップと、を含む、請求項１または２に記載の方法。

【請求項４】

接続の状態データを複製する前記ステップは、前記アクティブプロセスが前記接続を用いることができる間に実行される、請求項１から３のいずれか１項に記載の方法。

【請求項５】

それぞれ異なるネットワークアドレスを有するアクティブプロセスおよびスタンバイプロセスの複数の対を維持するステップを含む、請求項１から４のいずれかに記載の方法。

【請求項６】

前記ネットワークアドレスはＩＰアドレスである、請求項１から５のいずれかに記載の方法。

【請求項７】

前記ネットワーク接続がＴＣＰ接続である場合に、状態データを複製するステップは、ＴＣＰ状態情報を複製することを含む請求項２から６のいずれかに記載の方法。

【請求項８】

アクティブ状態のプロセス、スタンバイ状態のプロセス、および該スタンバイ状態のプロセスをアクティブ状態に移行させるための切り換え手段を有する耐故障性プラットフォームであって、

該アクティブ状態のプロセスは、ネットワークアドレスを有し、該切り換え手段は、スタンバイ状態のプロセスのアクティブ状態のプロセスへの移行の一部として、該ネットワークアドレスをスタンバイ状態のプロセスへ転送するように構成され、該プラットフォームは、

該アクティブ状態のプロセスに関連付けられた接続から状態データを抽出する第１の接続マネージャと、

該アクティブ状態のプロセスに関連付けられたネットワーク接続の状態データを該スタンバイ状態のプロセスに複製する複製マネージャと、

該スタンバイ状態のプロセスについて、複製されたデータで更新された対応するスタンバイネットワーク接続を維持する第２の接続マネージャとを備え、

該切り換え手段は、該スタンバイ状態のプロセスのアクティブ状態への移行に先立って、該アクティブ状態のプロセス内の該ネットワーク接続を、該ネットワーク上の該接続を閉じることなく非活動化するように構成され、ネットワークアドレスが該スタンバイ状態のプロセスに転送されたら、該ネットワークアドレスで該対応するスタンバイネットワーク接続を活動化するように構成されており、

該移行したスタンバイ状態のプロセスが該ネットワーク上で接続を再開する必要がないようにするプラットフォーム。

【請求項９】

前記ネットワーク接続が伝送制御プロトコル（ＴＣＰ）接続、トランザクション処理（ＯＳＩ　ＴＰ）接続またはストリーム制御伝送プロトコル（ＳＣＴＰ）接続のいずれかである請求項８に記載のプラットフォーム。

【請求項１０】

前記複製マネージャは、個別のソフトウエアモジュールの形をとる、請求項８または９に記載のプラットフォーム。

【請求項１１】

前記第１および／または第２の接続マネージャは、個別のソフトウエアモジュールの形をとる、請求項８から１０のいずれかに記載のプラットフォーム。

【請求項１２】
各プロセスは、命令セットを形成するアプリケーションソフトウエア層と、オペレーティングシステムソフトウエア層と、伝送制御プロトコルを適用することができるソフトウエア層とを提供し、
前記システムは、前記アクティブ接続の前記状態データを複製することができるプロセスを提供し、該プロセスは、少なくとも部分的に、前記アクティブプロセスの前記アプリケーション層によって実行される、請求項８から１１のいずれかに記載のプラットフォーム。
【請求項１３】
それぞれ異なるネットワークアドレスを有する、アクティブプロセスおよびスタンバイプロセスの複数の対を維持することを含む、請求項８から１２のいずれかに記載のプラットフォーム。
【請求項１４】
前記ネットワークアドレスはＩＰアドレスである、請求項８から１３のいずれかに記載のプラットフォーム。
r
【請求項１５】
前記ネットワーク接続がＴＣＰ接続である場合に、前記複製マネージャは、ＴＣＰ状態情報を含む状態データを複製するように構成されている請求項８から１４のいずれかに記載のプラットフォーム。
【請求項１６】
請求項８から１５のいずれかに記載の耐故障性プラットフォームであって、
ネットワーク接続は、関連付けられたプロセスが前記ネットワーク接続を閉じるか、または該関連付けられたプロセスが終わる度に、該ネットワーク接続が前記ネットワーク上で閉じられる第１の状態と、関連付けられたプロセスが前記ネットワーク接続を閉じるか、または該関連付けられたプロセスが終わるときに、該ネットワーク接続が該ネットワーク上で閉じられない第２の状態とをとることができ、
前記プラットフォームは、プログラム制御下で前記接続を前記第１の状態と前記第２の状態との間で切り換え、接続状態情報を抽出し、該接続状態情報を設定することを可能にするアプリケーションプログラミングインターフェースを含み、
前記接続状態情報は、ネットワークアドレスを前記スタンバイ接続に転送することにより、前記ネットワーク上で接続が再開されることを必要とすることなく、アクティブ状態のプロセスから複製された前記状態情報で更新され維持されたスタンバイネットワーク接続が、移行したスタンバイ状態のプロセスによって用いられることを可能にするような情報である耐故障性プラットフォーム。
【発明の詳細な説明】
【０００１】
【発明の属する技術分野】
本発明は耐故障性コンピューティングに関し、より具体的には、たとえばインターネットプロトコル（ＩＰ）ネットワークにおいて、耐故障性プラットフォームによって確立されるネットワーク接続を維持する耐故障性コンピューティングに関する。
【０００２】
【従来の技術】
インターネットがさらに普及し、一般的になってくると、それに応じて、毎日、終日稼動し続けなければならないミッション・クリティカルなアプリケーションに対応するために高い可用性を提供するインターネット装置の必要性も増す。ネットワーク内の１つの故障箇所、すなわちゲートウエイおよびファイアウォールのような構成要素は、高い信頼性をもって構成される必要があり、冗長な装置および種々のタイプのクラスタ化システムで強化されている。しかしながら、耐故障性のＴＣＰ／ＩＰベースのシステムでは、効率的なフェイルオーバ処理が実施されないと、メッセージトラフィックの損失、およびネットワ

ークセッションの再初期化といった問題が生じる。
【０００３】
たとえば、Ｈ．３２３ゲートキーパーのハードウエア故障に対処するために、同じソフトウエアを動作させる２つの冗長ゲートキーパーを採用することができる。それらのゲートキーパーはいずれも、全く同じように、ＩＰネットワークに接続され、同じデータを受信し、同じデータを生成する。
【０００４】
ゲートキーパーの一方に故障が生じた場合、故障の直前にはいずれのゲートキーパーにおいてもソフトウエア状態が同じであるので、何ら切り換えの問題を生じさせることなく、他方のゲートキーパーが単独で作業を継続する。しかしながら、このような装置はコストがかかり、いずれのシステムにおいてもソフトウエアが全く同じように動作するため、その解決策は完全に満足できるものではない。すなわち、ゲートキーパーの一方に現れたソフトウエア障害は、他方のゲートキーパーにも現れる。
【０００５】
米国特許第６，０７８，９５７号は、それぞれが自身の特定の接続を持つ１組のクラスタメンバを備えたクラスタアセンブリを提案する。各クラスタメンバは、別のクラスタメンバが動作不能になっていることを認識する手段と、動作不能となったクラスタメンバによって実施されていたタスクのうちのいくつかを再びバランスするための手段とを備える。動作不能になったクラスタメンバがクラスタマスターである場合には、他のクラスタメンバは直ちに、クラスタマスターのタスクを、別のクラスタメンバに再度割当てる。
【０００６】
【発明が解決しようとする課題】
あるクラスタメンバが故障した場合にその接続を移すために、各クラスタメンバは、他の各クラスタメンバに、そのクラスタメンバが責任を担っていた接続に関する、記憶された不可欠な状態情報を転送する。しかしながら、このシステムは、ＩＰネットワークと直接通信するアプリケーションを提供する耐故障性プロセスの問題には対処しない。したがって、上記特許は、クラスタ要素から他のクラスタ要素に接続が移るとき、接続される端末から見て、そのようなアプリケーションが依然として同じ段階のままであることを保証するための手段を提供しない。
【０００７】
本発明は主に、ＩＰエンドポイントとの接続を有するシステム、およびそのような接続を処理するための方法を提供することを目的とする。それにもかかわらず、ここに記載される技術は、たとえばＴＣＰ（伝送制御プロトコル）接続、ＯＳＩ　ＴＰ（トランザクション処理）接続、またはＳＣＴＰ（ストリーム制御伝送プロトコル）を用いるシステムにおける、任意のタイプの通信ネットワークまたは通信プロトコルに適用されることができることは理解されよう。
【０００８】
【課題を解決するための手段】
簡単に言うと、本発明は、アクティブ状態のプロセス、スタンバイ状態のプロセス、およびスタンバイ状態のプロセスをアクティブ状態に移行させるための切り換え能力とを有する耐故障性プラットフォームにおいて、ネットワーク接続を提供する方法を提供する。この方法は、アクティブプロセスからスタンバイプロセスに、アクティブプロセスのネットワーク接続の状態データを複製するステップと、スタンバイプロセスについて、該複製されたデータで更新された対応するスタンバイネットワーク接続を維持するステップと、スタンバイプロセスからアクティブ状態への移行中、ネットワーク上の接続を閉じる（クローズする）ことなく、アクティブシステムのネットワーク接続を非活動化するステップと、ネットワークアドレスをスタンバイプロセスに転送するステップと、該ネットワークアドレスで対応するスタンバイ接続をアクティブ状態にするステップとを含み、それにより、移行したスタンバイプロセスが、ネットワーク上の接続を再開する必要性をなくす。
【０００９】

このようにしてスタンバイ接続を維持することにより、ネットワーク接続のリモートエンドに対して高いトランスペアレンシー（透過性）を保持しつつ、フェイルオーバの達成を可能にする。

【００１０】

スタンバイプロセスのネットワークアドレスを活動化するステップは、スタンバイ接続が活動化される前に実行される。アクティブプロセスの接続を監視し、状態データの複製を起動するステップは、その接続のアイドル状態中に実行される。

【００１１】

接続の状態データを複製するステップは、アクティブプロセスがその接続を使用することができる間に実行されることができるようにするのが有利である。

【００１２】

第２の態様によれば、本発明は、アクティブ状態のプロセス、スタンバイ状態のプロセス、およびスタンバイ状態のプロセスをアクティブ状態に移行させるための切り換え手段とを有する耐故障性プラットフォームを提供する。このプラットフォームは、アクティブ状態に関連付けられた接続から状態データを抽出する第１の接続マネージャと、アクティブプロセスに関連付けられたネットワーク接続の状態データをスタンバイプロセスに複製する複製マネージャと、スタンバイプロセスについて、該複製されたデータで更新された対応するスタンバイネットワーク接続を維持する第２の接続マネージャとを備える。最後に、切り換え手段が、スタンバイプロセスのアクティブ状態への移行の一部として、ネットワーク上の接続を閉じることなくアクティブシステムのネットワーク接続を非活動化し、ネットワークアドレスをスタンバイプロセスに転送し、該ネットワークアドレスで対応するスタンバイ接続を活動化するように構成され、それにより、移行したスタンバイプロセスが、ネットワーク上で接続を再開する必要性をなくす。

【００１３】

好ましい実施形態では、複製マネージャ、および／または第１および／または第２の接続マネージャは、個別のソフトウエアモジュールの形態をとる。

【００１４】

本発明の第３の態様は、上記の汎用タイプの耐故障性プラットフォームを提供し、そのプラットフォームでは、ネットワーク接続は、関連するプロセスがその接続を閉じるか、またはそのプロセスが終わる度に、ネットワーク上でそれらの接続が閉じられる第１の状態と、関連するプロセスがその接続を閉じるか、またはそのプロセスが終了するときに、ネットワーク上でそれらの接続が閉じられない第２の状態とを有する。このプラットフォームは、プログラムの制御下で、接続が第１の状態と第２の状態との間で切り換えられるようにし、さらに、接続状態情報を抽出して、該接続状態情報の設定を可能にするアプリケーションプログラミングインターフェースを有する。接続状態情報は、ネットワークアドレスをそのスタンバイ接続に転送することにより、ネットワーク上で接続を再開することを必要とすることなく、アクティブプロセスから複製された状態情報で更新された、維持されるスタンバイネットワーク接続が、移行したスタンバイプロセスによって用いられることを可能にするような情報である。

【００１５】

【発明の実施の形態】

本発明の他の特徴、目的および利点は、本発明の好ましい実施形態の以下に記載される説明を通して、および図面を通して、当業者には明らかになるであろう。

【００１６】

図１を参照すると、接続１１０および２１０を介していずれもＩＰネットワーク３００に接続されることができる２つのハードウエア装置１００および２００を含む耐故障性コンピュータシステムが示される。耐故障性システムはたとえば、米国特許第５，９７８，９３３号に記載されるタイプの耐故障性プラットフォーム、またはヒューレット・パッカード社によって市販されるＯｐｅｎＣａｌｌＩＮプラットフォームに基づくことができる。そのようなシステムでは、故障検出のためのアプリケーション監視および故障に起因する

アクションは、構成要素、詳細には、当業者によく知られている高可用性（ＨＡ）コントローラ１０１、２０１によって管理される。図２は、ＨＡコントローラ１０１、２０１によって用いられる状態機械と、ＨＰ　ＯｐｅｎＣａｌｌ　ＩＮプラットフォームによって用いられるＨＡプロセスとを示す。

【００１７】

図２では、図示される状態のうちのいくつか、すなわちブート状態３００と、同期状態３１０と、活動化状態３２０と、停止状態３３０は、過渡（非常駐）状態である。そのプロセスは、最終的な安定状態に達する前の中間ステップとしてのみ、それらの状態を通過する。安定状態は、アクティブ状態３４０、ホットスタンバイ状態３５０、およびコールドスタンバイ状態３６０である。そのプロセスは、任意の状態でダウンすることがあるが、簡略化するために、ダウンへの状態遷移は図２には示されない。１つの装置上のＯｐｅｎＣａｌｌ耐故障性コントローラは、ピア装置２００上の対応するプロセスの状態を考慮に入れて、装置１００上で実行されるＨＡプロセスの状態遷移を調整する。

【００１８】

図１の装置１００および２００は、ローカルエリアネットワーク（ＬＡＮ）４００によって互いにリンクされ、それぞれ、その接続に最も近い層からその接続から最も離れた層への少なく４つのソフトウエア層、１２０～１５０および２２０～２５０、すなわちインターネットプロトコル（ＩＰ）層１２０、２２０、伝送制御プロトコル（ＴＣＰ）層１３０、２３０、オペレーティングシステム層１４０、２４０、およびアプリケーション層１５０、２５０を含む。

【００１９】

アプリケーション層１５０、２５０は、関連する特定用途のアプリケーションにしたがって種々のアプリケーションレベルのサービスを実行する。たとえば、１つのアプリケーションレベルの機能は、プリペイド通信に対するアプリケーションにおいて、通信時間のプリペイド度数を更新することができる。各装置１００および２００において、プロセスはアクティブ状態またはスタンバイ状態のいずれかであることができる。アクティブプロセスは、任意の特定の時間においてアプリケーションサービスを提供するプロセスである。スタンバイプロセスは、アクティブプロセスに障害が生じた場合に引き継ぎの役割を果たす。アクティブ装置１００のアプリケーション１５０はデータを処理し、一方、スタンバイ装置２００のアプリケーション２５０の状態は、それ自体がよく知られているやり方で、ローカルネットワーク４００を通して耐故障性コントローラによって更新される。

【００２０】

本システムでは、システムのうちの１つのシステムのＴＣＰ接続のみ、たとえば、装置１００のＴＣＰ接続１１０のみが任意のある時点でアクティブ状態にある。他のシステム上のＴＣＰ接続、たとえば２１０のＴＣＰ接続は、その接続がデータを全く受信も送信もできないように構成されるという意味で、スタンバイ接続である。したがって、スタンバイ装置２００は、物理的にはネットワークには接続されるが、インターネットプロトコルネットワーク３００からは見えない。

【００２１】

ここで、２つの装置間でのアプリケーション切り換え中に、確立されたＴＣＰ接続を保持するための技術について説明する。用いられる一般的なアプローチは、その接続を同期した状態にしておくために、アクティブ接続とスタンバイ接続との間でデータおよび状態情報を伝達するための複製機構をアクティブアプリケーション１５０に設けることである。言い換えると、アプリケーション１５０は、ＴＣＰコンテキストを、アクティブ装置１００からスタンバイ装置２００に移動させる。

【００２２】

以下の説明では、リモート側、すなわちエンドポイントが、装置１００と装置２００との間の切り換え後にその接続を再開する必要がなければ、ＴＣＰ接続は保持されるものと見なされる。ここで用いられる用語「アクティブＴＣＰ接続」、ここでは接続１１０は、パケットを送受信することのできる接続のことを指す。用語「スタンバイＴＣＰ接続」、こ

こでは接続２１０は、活動化されるまでパケットを全く送受信することができない接続のことを指す。詳細には、これは、その接続が開始または終了されるときに、この接続を提供しているＴＣＰスタック１３０、２３０によって、アプリケーションのためにパケットが送信されることはなく、そのＴＣＰ接続のリモートエンドに、生存しているパケットが全く送信されないことを意味する。

【００２３】

図３は、複製マネージャモジュール１６０、２６０および接続マネージャモジュール１７０、２７０を用いて、接続管理および複製機能を、アプリケーションコア１５０、２５０から分離する好ましい構造を示す。当然、他の実施形態では、接続マネージャ機能および複製マネージャ機能がアプリケーション自体に、またはオペレーティングシステムに組み込まれることができることは理解されよう。

【００２４】

接続マネージャ１７０、２７０は、接続１１０、２１０を開始し、構成し、その状態を抽出し、その状態を更新し、かつその接続を終了するために、アプリケーション１５０、２５０によって用いられる単一インターフェースである。その際、接続マネージャ１７０、２７０は、アプリケーション１５０、２５０のコアから接続保持の実施の細部を隠すことができ、ここに記載される接続保持機能を使用することを望む他のアプリケーションによって再使用されることができる。

【００２５】

また接続マネージャ１７０、２７０は、１つのアプリケーションのすべての保持される接続を管理するのに好ましい場所でもある。例として、接続マネージャは、すべてのアプリケーションコアモジュール１５０、２５０の内部の詳細に影響を及ぼすことなく、かつそれを知る必要なく、ある時点におけるすべての接続の終了、非活動化、または活動化を容易にする。

【００２６】

複製マネージャ１６０、２６０は、スタンバイにどのようにデータが送信されるかの詳細と、そのようなデータの肯定応答受信の複雑な手順とを隠すようにして、アプリケーションのための抽象複製サービスを提供することができる。

【００２７】

図４は、１つの接続が有することができる種々の状態と、アクティブ側１００およびスタンバイ側２００の両方において取り得る状態遷移とを示す状態図である。図４に示される状態および遷移は、アプリケーション１５０の視点からのＴＣＰ接続を表す。

【００２８】

その接続の各状態は、その状態に付随した以下の特性を有することができる。

【００２９】

Ｐ：ある状態において“Ｐ”があるとき、これは、その接続が保持された接続（ＴＣＰ接続保持拡張のうちのいずれにも対応しない標準的なＴＣＰ接続とは対照的に）であること示す。図４の３つのすべての状態は保持された状態であり、他のタイプの接続は示されていない。

【００３０】

Ａ：ある状態において“Ａ”があるとき、これは、その状態がアクティブな（活動中の）接続であることを示す。アクティブ状態はデータ転送を実行することができ、Ｆ／ＮＦフラグによって示されるように終了動作中に特定の挙動を持つ。

【００３１】

Ｓ：ある状態において“Ｓ”があるとき、これは、その状態がスタンバイ接続であることを示す。接続が閉じられるときであっても、そのような接続上ではパケットは送信されない。

【００３２】

Ｆ：ある状態において“Ｆ”があるとき、これは、ＴＣＰ接続がローカル的に閉じられるときには必ず、該ＴＣＰ接続はそのネットワーク上においてＴＣＰピアで終了されること

を示す。状態３７０の場合のように、Ａインジケータと組み合わせられるときには、その接続は標準的なＴＣＰ接続のように動作する。

【００３３】

ＮＦ：ある状態において“ＮＦ”があるとき、これは、アプリケーションによって明示的に要求される際、またはプロセス終了時に、ＴＣＰ接続がローカル的に閉じられるときには、該ＴＣＰ接続がピアＴＣＰで終了されないことを示す。そのローカルソケットは、暗黙のうちにパージされる。このオプションは、リモートＴＣＰピアが接続の終了を起動する場合には機能しない。その場合には、現在のオプション値が何であろうと、その接続は有効に終了される。状態３９０の場合のように、Ａインジケータと組み合わせられるときには、その接続は、同期した対応するスタンバイ接続を有するアクティブ接続である。

【００３４】

アクティブアプリケーション１５０が開始し、スタンバイアプリケーション２５０が開始され同期する前に、そのアプリケーションが開いた（開始した）すべての保持された接続がアクティブ状態（状態３７０）になる。アクティブアプリケーション１５０が終わった場合には、ＴＣＰは、ネットワーク上のリモート側への接続を閉じる。この挙動によって、ＴＣＰスタックは、リモート側への接続を閉じることができる。なぜなら、接続処理を引き継ぐためのスタンバイアプリケーションが存在しないためである。

【００３５】

以下により詳細に記載されることになる同期段階中に、スタンバイアプリケーションは、保持されたスタンバイ接続を形成し、アクティブ側から接続状態情報を複製する。同期段階が正常に完了すると、アクティブアプリケーションは、そのすべての保持された接続を状態３９０に移し、保持されたＴＣＰ接続が処理終了時に閉じられないようにする。これにより、切り換え後に、スタンバイアプリケーションがその接続処理を引き継ぐことができるようになる。アプリケーションが接続を閉じることを欲した場合には、そのアプリケーションはその接続を状態３７０に戻さなければならない。

【００３６】

スタンバイ接続は状態３８０にある。そのような接続が活動化されると、その接続は状態３７０に移される。新しいスタンバイシステムが再開され同期すると、そのような接続は状態３９０に移されることができる。

【００３７】

アクティブアプリケーションでは、ｓｏｃｋｅｔ（）、ｃｏｎｎｅｃｔ（）、ｂｉｎｄ（）、ｌｉｓｔｅｎ（）、ａｃｃｅｐｔ（）コールを含む標準的なソケットコールを用いて、ソケットが形成される。これが図５（ａ）に示される。アクティブアプリケーション１５０はアクティブ接続１１０を管理する。２つの装置間で複製される必要がある情報の大部分は、両方の装置１００および２００上に設けられるＴＣＰスタックモジュール１３０、２３０によって保持される。

【００３８】

その後、アクティブ側の接続に生じている変化は、その接続の持続時間にわたって、スタンバイ側に繰返し複製される。複製は、パケットが受信または送信される度に行われる必要はない。むしろ、その状態が安定している接続持続時間中の選択された時点においてのみ、この情報を複製することがより効率的である。その接続によってトラフィックが処理されていない（すなわち、その接続がアイドル状態である）とき、接続状態は安定している。待ち状態のアウトバウンドデータが存在せず、アプリケーションによって読み取られるのを待っている受信データが存在しない場合には、ＴＣＰ接続１１０はアイドル状態であると見なされる。そのようなアイドル接続は、ここに記載される技術を用いて保持されることができる。

【００３９】

アプリケーション１５０が、その接続状態が安定している（すなわち、その接続がアイドル状態である）と判断したとき、該アプリケーションは、ＴＣＰスタック１３０から各接続の状態を抽出し、それをスタンバイアプリケーション２５０に送信することにより、対

応するスタンバイ接続に接続１１０を複製することを決定することができる。スタンバイアプリケーション２５０はソケットを形成し、それをスタンバイソケットに構成し、そのソケットを、アクティブアプリケーションから受信された接続状態情報で更新する。最後に、スタンバイアプリケーションは、アクティブアプリケーションに肯定応答ＡＣＫを送信する。このプロセスが図５（ｂ）に示される。接続状態データは、アクティブ装置１００のアプリケーション層１５０によって、ＯＳ層１４０を介してＴＣＰスタック１３０から取得され、その後、ＬＡＮ４００を介して、スタンバイ装置２００のアプリケーション層２５０に送信される。

【００４０】

この期間中、アクティブアプリケーションは、スタンバイシステムからＡＣＫを受信するのを待つことはなく、接続は単に状態３７０においてアクティブ状態のままであることが好ましいことに留意されたい。ＡＣＫが受信されると、アクティブアプリケーションはソケットを構成し、それを図４の状態３９０に設定することにより、決してネットワーク上の接続を閉じないようにする。スタンバイ２００において接続の複製が失敗した場合に、これがアクティブ接続１１０に影響を及ぼさないことが許容可能とされる。その場合、アプリケーションは、スタンバイ肯定応答を待って妨害されることがないので、複製が非常に速く行われる。他の状況では、確立された接続１１０の保持を促進し、その接続がスタンバイシステム２００上で保持されることができた場合でもその接続が該アプリケーションによってのみ用いられることを確実にするために、ＡＣＫを待つことが好ましい場合がある。

【００４１】

アクティブアプリケーション１５０が接続１１０を閉じることを決めたとき、最初に、その接続を状態３７０に設定することにより、ネットワーク上においてその接続をピアで終了するようにその接続を構成し、その後に該接続を閉じなければならない。その後、この終了動作は、そのスタンバイ接続２１０を閉じるスタンバイアプリケーション２５０に複製される。このプロセスが図５（ｃ）に示される。

【００４２】

その接続がリモート側によって閉じられる場合には、ＴＣＰスタック１３０は、標準的な接続の場合と同様に動作する。アクティブアプリケーション１５０は終了動作をスタンバイアプリケーション２５０に複製し、その後、その接続を閉じる。これが図５（ｄ）に示される。

【００４３】

アクティブアプリケーション１５０が終わる場合には、スタンバイアプリケーション２５０がアクティブ状態になる前に、該アクティブアプリケーションに割り当てられたＩＰアドレスがスタンバイホストに移される。装置１１０は、１つのＩＰアドレスのみを有することが可能である。しかしながら、各アクティブアプリケーション１５０は、そのアプリケーションのプロセスによって用いられる、自身に専用のＩＰアドレスを有することが好ましい。そのアプリケーションのＩＰアドレスは、アクティブアプリケーション１５０を提供するアクティブ装置１００上でのみ有効である。切り換え中に、ＩＰアドレスは、古いアクティブ装置１００から新しいアクティブ装置２００に移される（すなわち、そのＩＰアドレスは古いアクティブ装置上で非活動化され、新しい装置上で活動化される）。そのＩＰアドレスは、任意の時点で１つの装置においてのみアクティブ状態である。

【００４４】

１つのＭＡＣアドレスから別のＭＡＣアドレスにＩＰアドレスを転送するための技術は当業者にはよく知られており、ＨＰ　Ｏｐｅｎｃａｌｌ製品は、これを行うためのＩＰアドレスマネージャ構成要素を含む。そのような技術の記載は、たとえば、米国特許第６，０４９，８２５号に見いだすことができる。

【００４５】

ＨＡプロセス状態機械は、切り換え中に、ＩＰアドレスが装置上でアクティブ状態になった後にのみ、スタンバイプロセスがアクティブ状態になり、ＩＰアドレスが装置上で非活

動化された後にのみ、アクティブプロセスがスタンバイ状態になるように、ＩＰアドレスの移動と同期する。これは、ＩＰアドレスマネージャとアプリケーションとの間で通信を行うことによって、または任意のスタンバイ接続を活動化する前にアプリケーション２００にＩＰアドレス状態を検査させることによって、確実に実行されることができる。標準的なＡＰＩコールを、ＩＰアドレス状態を検査するために利用することができることは理解されよう。

【００４６】

プロセス終了時に、カーネルは、すべてのそのファイル記述子を閉じる。すべての複製されたアクティブ接続が、ピアで接続を終了しないように設定されたので（状態３９０）、ＴＣＰスタック１３０は、その接続が閉じられていることを該接続のリモート側に示さない。

【００４７】

同時に、高可用性コントローラによって障害を通知されているスタンバイシステムは、その装置上でＩＰアドレスが活動化されるまで待機する。その後、その装置は、そのスタンバイ接続を活動化し、そのリスン（listen）接続を開始する。活動化の後、新しいスタンバイシステムが再開され同期するまで、その接続は状態３７０に留まらなければならない場合があることに留意されたい。これらのプロセスが図６（ａ）に示されており、好ましい実施形態では、それらのプロセスは、図２に示される活動化状態３２０中に行われる。

【００４８】

スタンバイソケットがアクティブ状態になると、ＴＣＰ接続は、そのＴＣＰピアと依然として同期しているか否かを判断する。最後の同期点以来アクティブＴＣＰ接続がデータを受信していた場合には、スタンバイＴＣＰは同期から外れるであろう。その場合、該ＴＣＰ接続は閉じられ、スタンバイアプリケーションによって再形成される。

【００４９】

手動切り換え（すなわち、プロセス終了によって引き起こされない切り換え）の場合、たとえば予防保守の場合、または、ソフトウエアあるいはハードウエアアップグレードを可能にするために、アクティブアプリケーション１００はそのすべての接続を非活動化し、スタンバイ状態になる。一方、スタンバイアプリケーション２００はすべての接続を活動化し、アクティブ状態になる。この手動切り換えは図６（ｂ）に示されており、好ましい実施形態では、そのプロセスは、図２に示される「停止」状態３３０において行われる。

【００５０】

リスン（ｌｉｓｔｅｎ）するために用いられるソケットは、それらが接続の一部を形成しないので保持される必要がないことに留意されたい。アクティブプロセスによって確立されるリスンソケットを処理するために、スタンバイプロセスは２つの方策のうちの１つの採用することができる。第１に、スタンバイプロセスは、アクティブプロセス状態に切り替わるときに、リスンソケットを再形成することができる。これは、ソケット機能シーケンス、ｓｏｃｋｅｔ（）、ｃｏｎｎｅｃｔ（）、ｂｉｎｄ（）、ｌｉｓｔｅｎ（）、ａｃｃｅｐｔ（）を実行することにより行うことができる。一般に、このアプローチは、最も安全で、かつ最も簡単であると考えられる。しかしながら、代替的に、スタンバイプロセスがリスンソケットを形成し、そのリスンソケットを任意のＩＰアドレス（ＩＰアドレス＝ＩＮＡＤＤＲ＿ＡＮＹ）および指定されたポート番号にバインドすることができる。これは、リスンソケットを再形成するステップを節約することになるが、スタンバイプロセスは、依然としてスタンバイモードにあるとき（その装置上でアクティブ状態にあるＩＰアドレスについて）、接続要求の可能性を制御しなければならないであろう。

【００５１】

ＴＣＰ接続１１０が、切り換えが生じた時点でアイドル状態でない場合には、その接続１１０は終了され、スタンバイアプリケーションによって再形成される必要がある。

【００５２】

以下により詳細に記載されるように、好ましい実施形態では、アプリケーションが接続特性を制御し、アクティブ側の状態を抽出し、スタンバイ側の状態を更新できるようにする

ために、拡張されたソケットアプリケーション・プログラミング・インターフェース（ＡＰＩ）が設けられる。このＡＰＩはたとえば、アプリケーションとは個別の、アプリケーションによってコールされる接続マネージャ１７０、２７０および複製マネージャ１６０、２６０のようなモジュールの形で、都合のいいように実装されることができる。図１に示されるＩＰ層１２０、２２０またはＴＣＰ層１３０、２３０を基点としたソフトウエア階層を参照すると、このＡＰＩは、アプリケーション１５０、２５０とオペレーティングシステム１４０、２４０との間に配置されることができる。各接続１１０の状態は、具体的にはＴＣＰデータとＯＳデータとを含む。

【００５３】

好ましい実施形態は、標準のＨＰ－ＵＸソケットおよび関連するコールへの追加からなる。具体的には、ｇｅｔｓｏｃｋｏｐｔ（）コールは、同様の接続／確立されたソケットを構築するのに必要とされるＴＣＰ状態情報を戻すために拡張される。これは、アクティブまたはスタンバイソケット上で実行されることができ、そのソケットにもその接続にも影響を及ぼさない読出し専用動作である。状態３７０から３９０への、または状態３９０から３７０へのソケットを設定するのに同じコールが用いられる。

【００５４】

状態情報の特質は、プラットフォームに応じて変化し、ネットワーク上で接続を再開することを必要とせずに、活動化されるスタンバイプロセスによって用いられるべき情報で、スタンバイソケットが更新されることを可能にするように選択される。

【００５５】

ｓｅｔｓｏｃｋｏｐｔ（）コールは、アクティブソケットから取得されるＴＣＰ状態情報でスタンバイソケットを再同期できるようにするために拡張される。これは、スタンバイソケットにおいてのみ行われることができ、すべての必要とされる層に影響を及ぼす。ｓｅｔｓｏｃｋｏｐｔ（）コールは、ｓｏｃｋｅｔ（）後にスタンバイソケット接続を形成するために実行される。これは、アクティブ接続をスタンバイ状態にする（非活動化する）ために、アクティブ接続上でコールされる。これは、スタンバイ接続をアクティブ状態にするために、スタンバイ接続上でコールされる。

【００５６】

図３に戻ると、好ましい実施形態では、接続マネージャ１７０は、１つの場所で、すべての開始している接続１１０と、その状態とを追跡する。接続マネージャ１７０は、同時に複数の接続１１０上で動作を実行することができ、たとえば、すべての接続を閉じ、すべての接続を複製し、すべての接続を活動化／非活動化することができる。接続マネージャ１７０は、それが複製されているか否かを示すために、接続１１０において特定のフラグを設定することができ、その際、接続テーブル上をただ単にループすることにより、未だ複製されていない接続を複製する。接続マネージャ１７０は、アプリケーション１５０によって用いられている接続の複製、活動化、抽出、または状態取得の間の長い期間の間に、アプリケーション１５０を妨害するのを避けるために、マルチスレッドであるのが好ましい。

【００５７】

アプリケーション１５０はＴＣＰ接続状態を抽出し、それを、スタンバイアプリケーション２５０に送信する。その後、スタンバイアプリケーション２５０は、状態データを、スタンバイ装置２００のＴＣＰ層２３０に送信する。

【００５８】

ここに記載されるソフトウエア障害フェイルオーバ技術は、個別のスタンバイ装置を用いて実装することに制限されない。むしろ、記載される方法は、同じ装置上のスタンバイアプリケーションに対するアプリケーションソフトウエアのフェイルオーバにも同様に適用することができる。アプリケーションモニタが、そのアプリケーション自体に障害があるものと判断することができる場合には、ローカルにフェイルオーバを行うことを選択することができる。

【００５９】

２つのプロセスを含み、各プロセスが同様のサービスアプリケーション命令セットを提供し、各プロセスがＩＰアドレスを有し、各プロセスがインターネットプロトコルエンドポイントを有する一連のアクティブ接続を提供することができるシステムが記載される。このシステムは、他のプロセスにおいてアクティブ接続を提供しない間に、１つのアクティブプロセスの一連のアクティブ接続を提供するための手段を備える。このシステムは、そのアクティブプロセスから他のアクティブプロセスに、アクティブ接続の状態データを含むデータと、アクティブプロセスのサービスアプリケーションの命令セットの状態データを含むデータとを複製するための手段を備える。アクティブ接続の状態データを含むデータは、更新されたスタンバイ接続を該他のプロセスが維持できるようにするためのものであり、アクティブプロセスのサービスアプリケーションの命令セットの状態データを含むデータは、該他のプロセスの命令セットが更新された状態に維持されるようにするためのものである。このシステムは、該アクティブプロセスが利用できなくなったとき、アクティブプロセスのＩＰアドレスを非活動化し、他のプロセスのＩＰアドレスを活動化するための手段を備える。

【００６０】

本発明の特定の実施形態が記載されてきたが、本発明は、そのような記載される特定の構成に限定されるべきではない。本発明は、特許請求の範囲によってのみ制限される。特許請求の範囲自体は、請求される本発明の周辺を指示することを意図しており、本明細書によって開示される例示的な実施形態のみを請求するものと解釈されるのではなく、言語そのものが許容するのと同じ広さで解釈されることを意図している。

【００６１】

本発明は、以下の実施態様を含む。

【００６２】

（１）アクティブ状態のプロセス、スタンバイ状態のプロセス、および前記スタンバイ状態のプロセスをアクティブ状態に移行させる切り換え能力を有する耐故障性プラットフォームにおいてネットワーク接続を提供する方法であって、
アクティブプロセスからスタンバイプロセスに、該アクティブプロセスのネットワーク接続の状態データを複製するステップと、
前記スタンバイプロセスについて、前記複製されたデータで更新された対応するスタンバイネットワーク接続を維持するステップと、
前記アクティブ状態への前記スタンバイプロセスの移行中に、前記アクティブシステム内の前記ネットワーク接続を、該ネットワーク上の該接続を閉じることなく非活動化し、前記スタンバイプロセスにネットワークアドレスを転送し、該ネットワークアドレスで前記対応するスタンバイ接続を活動化するステップと、を含み、
前記移行したスタンバイプロセスが前記ネットワーク接続を再開する必要がないようにする、耐故障性プラットフォームにおいてネットワーク接続を提供する方法。

【００６３】

（２）前記スタンバイプロセスの前記ネットワークアドレスを活動化するステップは、前記スタンバイ接続が活動化される前に実行される、上記（１）に記載の方法。
（３）前記アクティブプロセスの前記接続を監視するステップと、該接続のアイドル状態が識別されたことに応答して前記状態データの複製を起動するステップと、を含む、上記（１）または（２）に記載の方法。
（４）接続の状態データを複製する前記ステップは、前記アクティブプロセスが前記接続を用いることができる間に実行される、上記（３）に記載の方法。
（５）それぞれ異なるネットワークアドレスを有するアクティブプロセスおよびスタンバイプロセスの複数の対を維持するステップを含む、上記（１）から（４）のいずれかに記載の方法。
（６）前記ネットワークアドレスはＩＰアドレスである、上記（１）から（５）のいずれかに記載の方法。

【００６４】

（７）アクティブ状態のプロセス、スタンバイ状態のプロセス、および該スタンバイ状態のプロセスをアクティブ状態に移行させるための切り換え手段を有する耐故障性プラットフォームであって、
アクティブプロセスに関連付けられた接続から状態データを抽出する第１の接続マネージャと、
前記アクティブプロセスに関連付けられたネットワーク接続の状態データをスタンバイプロセスに複製する複製マネージャと、
前記スタンバイプロセスについて、前記複製されたデータで更新された対応するスタンバイネットワーク接続を維持する第２の接続マネージャとを備え、
前記切り換え手段は、前記スタンバイプロセスのアクティブ状態への移行の一部として、前記アクティブシステム内の前記ネットワーク接続を、該ネットワーク上の該接続を閉じることなく非活動化し、ネットワークアドレスを前記スタンバイプロセスに転送し、該ネットワークアドレスで前記対応するスタンバイ接続を活動化するように構成されており、前記移行したスタンバイプロセスが前記ネットワーク上で前記接続を再開する必要がないようにするプラットフォーム。

【００６５】

（８）前記複製マネージャは、個別のソフトウエアモジュールの形をとる、上記（７）に記載のプラットフォーム。
（９）前記第１および／または第２の接続マネージャは、個別のソフトウエアモジュールの形をとる、上記（７）または（８）に記載のプラットフォーム。
（１０）各プロセスは、命令セットを形成するアプリケーションソフトウエア層と、オペレーティングシステムソフトウエア層と、伝送制御プロトコルを適用することができるソフトウエア層とを提供し、
前記システムは、前記アクティブ接続の前記状態データを複製することができるプロセスを提供し、該プロセスは、少なくとも部分的に、前記アクティブプロセスの前記アプリケーション層によって実行される、上記（７）から（９）のいずれかに記載のプラットフォーム。
（１１）それぞれ異なるネットワークアドレスを有する、アクティブプロセスおよびスタンバイプロセスの複数の対を維持することを含む、上記（７）から（１０）のいずれかに記載のプラットフォーム。
（１２）前記ネットワークアドレスはＩＰアドレスである、上記（７）から（１１）のいずれかに記載のプラットフォーム。

【００６６】

（１３）アクティブ状態のプロセスをサポートする手段、スタンバイ状態のプロセスをサポートする手段、および該スタンバイ状態のプロセスをアクティブ状態に移行させるための切り換え能力を有する耐故障性プラットフォームであって、ネットワーク接続は、関連付けられたプロセスが前記ネットワーク接続を閉じるか、または該関連付けられたプロセスが終わる度に、該ネットワーク接続が前記ネットワーク上で閉じられる第１の状態と、関連付けられたプロセスが前記ネットワーク接続を閉じるか、または該関連付けられたプロセスが終わるときに、該ネットワーク接続が該ネットワーク上で閉じられない第２の状態とをとることができ、
前記プラットフォームは、プログラム制御下で前記接続を前記第１の状態と前記第２の状態との間で切り換え、接続状態情報を抽出し、該接続状態情報を設定することを可能にするアプリケーションプログラミングインターフェースを含み、前記接続状態情報は、ネットワークアドレスを前記スタンバイ接続に転送することにより、前記ネットワーク上で接続が再開されることを必要とすることなく、アクティブプロセスから複製された前記状態情報で更新され維持されたスタンバイネットワーク接続が、移行したスタンバイプロセスによって用いられることを可能にするような情報である耐故障性プラットフォーム。

【００６７】

（１４）アクティブ状態のプロセス、スタンバイ状態のプロセス、および該スタンバイ状

態のプロセスをアクティブ状態に移行させるための切り換え能力とを有する耐故障性プラットフォームにおいてネットワーク接続を提供するための方法であって、
アクティブプロセスからスタンバイプロセスに、該アクティブプロセスのネットワーク接続の状態データを複製するステップと、
前記スタンバイプロセスについて、前記複製されたデータで更新された対応するスタンバイネットワーク接続を維持するステップと、
前記スタンバイ状態のプロセスをアクティブ状態に移行させるステップとを含み、
前記スタンバイ状態のアクティブ状態への移行中に、前記アクティブシステム内の前記ネットワーク接続を、該ネットワーク上の該接続を閉じることなく非活動化し、ネットワークアドレスを前記スタンバイプロセスに転送し、該ネットワークアドレスで前記対応するスタンバイネットワーク接続を活動化し、該移行したスタンバイプロセスが前記ネットワーク上で前記接続を再開する必要がないようにする、方法。

【００６８】

（１５）アクティブ状態のプロセス、スタンバイ状態のプロセス、および該スタンバイ状態のプロセスをアクティブ状態に移行させるための切り換え能力とを有する耐故障性プラットフォームにおいてインターネットプロトコル（ＩＰ）ネットワーク接続を提供する方法であって、
アクティブプロセスからスタンバイプロセスに、該アクティブプロセスのネットワーク接続の状態データを複製するステップと、
前記スタンバイプロセスについて、前記複製されたデータで更新された対応するスタンバイネットワーク接続を維持するステップと、
前記スタンバイ状態のプロセスをアクティブ状態に移行させるステップとを含み、
前記スタンバイ状態のアクティブ状態への移行中に、前記アクティブシステム内の前記ネットワーク接続を、該ネットワーク上の該接続を閉じることなく非活動化し、ＩＰネットワークアドレスを前記スタンバイプロセスに転送し、該ＩＰネットワークアドレスで前記対応するスタンバイネットワーク接続を活動化し、前記移行したスタンバイプロセスが前記ネットワーク上で前記接続を再開する必要がないようにし、
前記スタンバイプロセスの前記ネットワークアドレスを活動化するステップは、前記スタンバイ接続が活動化される前に実行される、方法。

【００６９】

【発明の効果】

本発明によれば、ネットワーク接続のリモートエンドに対して高いトランスペアレンシー（透過性）が保持され、フェイルオーバが達成される。

【図面の簡単な説明】

【図１】本発明の一実施形態に従う、システムのアーキテクチャの概略図。

【図２】本発明の一実施形態に従う、システムにおける１つの装置の種々の状態を示す状態図。

【図３】本発明の一実施形態に従う、複製マネージャモジュールおよび接続マネージャモジュールを含むシステムの概略図。

【図４】本発明の一実施形態に従う、システム内の接続の種々の状態を示す状態図。

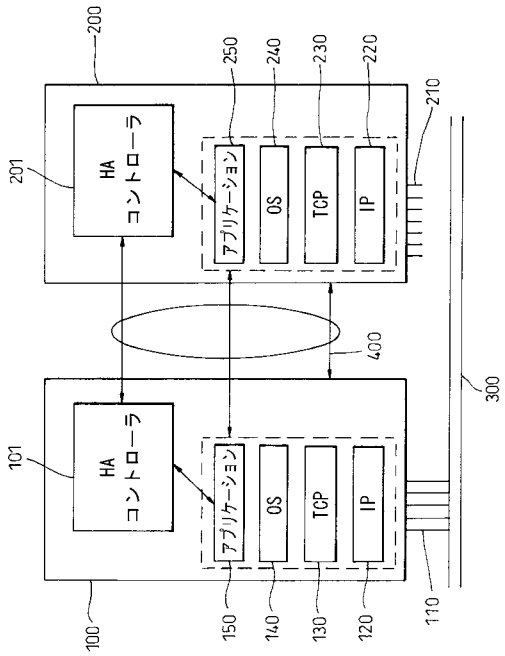
【図５】本発明の一実施形態に従う、アクティブ接続およびスタンバイ接続を設定するために用いられるプロセス、および閉じるために用いられるプロセスを示す図。

【図６】本発明の一実施形態に従う、プロセス終了時、またはアクティブシステムからスタンバイシステムへの接続の手動による移動時のプロセスを示す図。

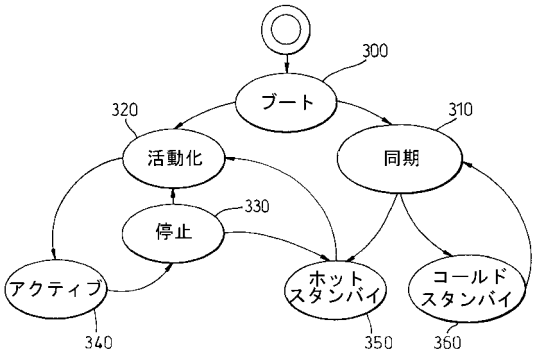
【符号の説明】

１００、２００　ハードウェア装置
１１０、２１０　接続
３００　　ネットワーク
１６０、２６０　複製マネージャ
１７０、２７０　接続マネージャ

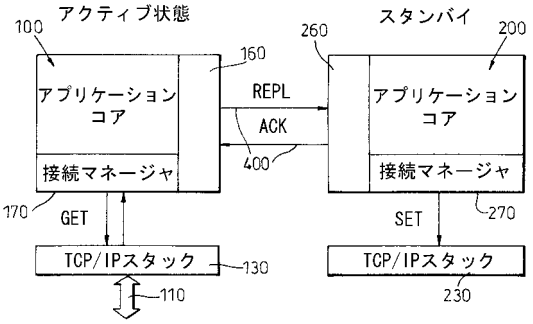
【図１】


200
250
240
230
220
210
201
HA
コントローラ
アプリケーション
OS
TCP
IP
400
300
101
HA
コントローラ
アプリケーション
OS
TCP
IP
100
150
140
130
120
110

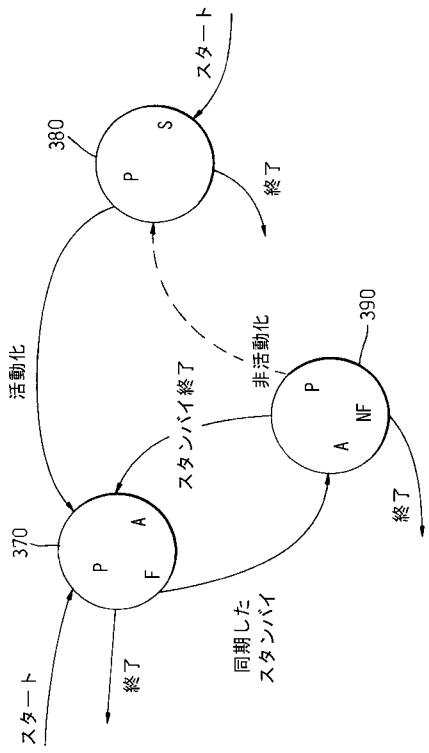
【図２】


300
ブート
320
活動化
310
同期
330
停止
アクティブ
340
ホット
スタンバイ
350
コールド
スタンバイ
360

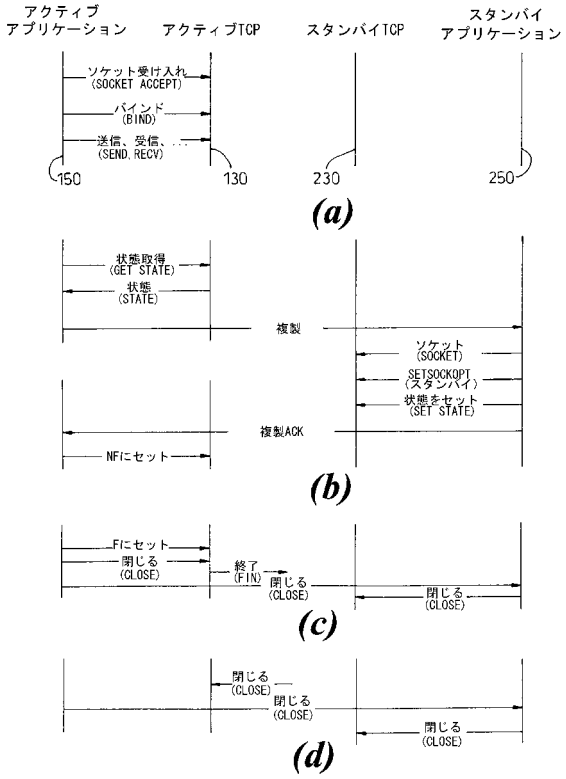
【図３】


アクティブ状態
スタンバイ
100
260
200
160
アプリケーション
コア
REPL
ACK
400
アプリケーション
コア
接続マネージャ
接続マネージャ
170
GET
SET
270
TCP/IPスタック
130
TCP/IPスタック
110
230

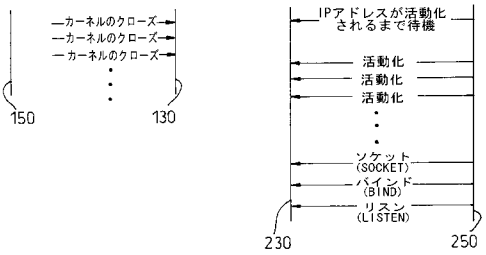
【図４】


スタート
380
P
S
終了
活動化
非活動化
390
スタンバイ終了
P
A
NF
370
P
A
F
終了
同期した
スタンバイ
スタート

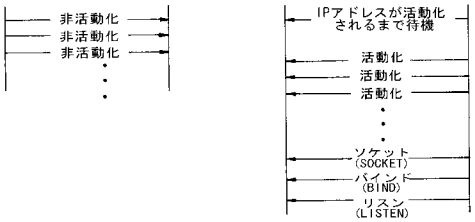
【図５】


アクティブ
アプリケーション
アクティブTCP
スタンバイTCP
スタンバイ
アプリケーション
ソケット受け入れ
(SOCKET ACCEPT)
バインド
(BIND)
送信、受信、...
(SEND, RECV)
150
130
230
250
(a)
状態取得
(GET STATE)
状態
(STATE)
複製
ソケット
(SOCKET)
SETSOCKOPT
(スタンバイ)
状態をセット
(SET STATE)
複製ACK
NFにセット
(b)
Fにセット
閉じる
(CLOSE)
終了
(FIN)
閉じる
(CLOSE)
閉じる
(CLOSE)
(c)
閉じる
(CLOSE)
閉じる
(CLOSE)
閉じる
(CLOSE)
(d)

【図６】


カーネルのクローズ
カーネルのクローズ
カーネルのクローズ
150
130
IPアドレスが活動化されるまで待機
活動化
活動化
活動化
ソケット (SOCKET)
バインド (BIND)
リスン (LISTEN)
230
250

(a)


非活動化
非活動化
非活動化
IPアドレスが活動化されるまで待機
活動化
活動化
活動化
ソケット (SOCKET)
バインド (BIND)
リスン (LISTEN)

(b)

フロントページの続き

(72)発明者 ジャック・ボウド
フランス国３８６６０サン・ヴァンサン・デュ・マルキュゼ、シャマン・ジュ・トーティエ
(72)発明者 ステラ・クゥォン
アメリカ合衆国９４０２２カリフォルニア州ロス・アルトス、マニュエル・ロード １４４７０
(72)発明者 アイザック・ウォン
アメリカ合衆国９５１５９カリフォルニア州サン・ノゼ、ピー・オー・ボックス ２６４５４
(72)発明者 デニス・ロジャー
フランス国３８７６０ヴァルス、リュト・ジュ・マルティネ・ダン・バ、レ・マージョエラ、ロト ２０
(72)発明者 アブデサター・サッシ
フランス国エフ－３８１００グルノーブル、シャマン・デュ・レグリス ７

審査官 石井 研一

(56)参考文献 国際公開第００／０６０４７２（WO，Ａ１）
特開平０８－０１６４２２（ＪＰ，Ａ）
国際公開第００／０３１９４２（WO，Ａ１）

(58)調査した分野(Int.Cl.，ＤＢ名)
H04L 12/56
H04L 12/28