(19)日本国特許庁(JP)　(12)特　許　公　報(B2)　(11)特許番号

特許第4565740号
(P4565740)

(45)発行日　平成22年10月20日(2010.10.20)　(24)登録日　平成22年8月13日(2010.8.13)

(51)Int.Cl.			F I		
H04Q	3/545	(2006.01)	H04Q	3/545	
H04M	3/00	(2006.01)	H04M	3/00	A
H04Q	3/00	(2006.01)	H04Q	3/00	

請求項の数 9　(全 46 頁)

(21)出願番号	特願2000-524941 (P2000-524941)
(86)(22)出願日	平成10年12月11日 (1998.12.11)
(65)公表番号	特表2001-526508 (P2001-526508A)
(43)公表日	平成13年12月18日 (2001.12.18)
(86)国際出願番号	PCT/US1998/026439
(87)国際公開番号	WO1999/030514
(87)国際公開日	平成11年6月17日 (1999.6.17)
審査請求日	平成17年12月5日 (2005.12.5)
審判番号	不服2008-12633 (P2008-12633/J1)
審判請求日	平成20年5月19日 (2008.5.19)
(31)優先権主張番号	60/069,576
(32)優先日	平成9年12月12日 (1997.12.12)
(33)優先権主張国	米国(US)

(73)特許権者　599072851
アルカテル・ユー・エス・エイ・ソーシング、エル・ピー
アメリカ合衆国、テキサス・７５０７５－５８１３、プレイノウ、コイツ・ロード・１０００
(74)代理人　100062007
弁理士　川口　義雄
(74)代理人　100114188
弁理士　小野　誠
(72)発明者　シヤー，マヘシユ，フイフス
アメリカ合衆国、テキサス・７５０２５、プレイノウ、ボーウイー・ドライブ・２６０８

最終頁に続く

(54)【発明の名称】ネットワークマネージメント

(57)【特許請求の範囲】

【請求項１】

通信機能を実施するアプリケーションプログラムと、アプリケーションプログラムをサポートするサイトにある少なくとも１つのノード上で動作するオペレーティングシステムとの間のソフトウェアインタフェースを提供し、さらに、アプリケーションプログラムと通信ネットワークとの間のインタフェースを形成する方法であって、

サービスからノードを除去し、サービスにノードを復元し、サービスからアプリケーションを除去し、かつサービスにアプリケーションを復元するように動作可能なネットワークプラットフォームマネージャ（１０４）を提供すること、

ノードを監視し、かつ故障ノードが回復するのを可能にするように動作可能なネットワークシステム保全マネージャ（１０６）を提供すること、

通信プラットフォームに結合されたホストとインタフェースするように動作可能な構成マネージャ（１０８）を提供すること、

ノード用の管理機能を提供するように動作可能なノードプラットフォームマネージャ（１１２）を提供すること、

ノードプラットフォームマネージャの指示でプロセスを開始および停止するように動作可能なサービスマネージャ（１３４）を提供すること、および

ノード間リンクを監視するように動作可能なノードシステム保全マネージャ（１３０）を提供すること

を含むことを特徴とする方法。

【請求項２】
さらに、
イベントを受信したいと望むクライアントプロセスを登録するように動作可能なイベントマネージャを提供すること、および
イベントを受信するように登録されたクライアントプロセスのためのインタフェースを提供するように動作可能なイベントレシーバを提供すること
を含む請求項１に記載の方法。
【請求項３】
さらに、日時機能を提供するように動作可能なタイママネージャを提供することを含む請求項１に記載の方法。
【請求項４】
さらに、
複数のノードにわたって発生する特定のイベントをカウントするように動作可能なペグカウンタプロセスを提供すること、
特定のイベントの期間を累積するように動作可能な時間計測プロセスを提供すること、
ノード上のカウンタデータを収集して、収集されたデータを記憶するように動作可能なデータ収集プロセスを提供すること
を含む請求項１に記載の方法。
【請求項５】
さらに、
ブートスクリプトを実行すること、
ブートスクリプトに従ってサービスマネージャを開始すること、
サービスマネージャによって、ノード用のノードプラットフォームマネージャを開始すること、
サービスマネージャによって、ノード用のＰＲＥ－ＭＩＮ構成エレメントを開始すること、
サービスマネージャによって、ノード用のＯＳ－ＭＩＮ構成エレメントを開始すること、および
ノード内のＯＳ－ＭＩＮ構成エレメントに応答してノードの状態をアップグレードすること
を含む請求項１に記載の方法。
【請求項６】
さらに、
構成可能エレメントの故障を監視して検出すること、
サービスマネージャに障害を通知すること、
サービスマネージャによって、構成可能エレメントに関するステータス変更を生成して、ノードシステム保全マネージャに通知を転送すること、
ノードシステム保全マネージャによって、ノードプラットフォームマネージャに通知を転送すること、
ノードプラットフォームマネージャによって、故障構成可能エレメントに応答してノードステータスを決定すること、および
ノードプラットフォームマネージャによって、ネットワークプラットフォームマネージャにノードステータス変更を通知すること
を含む請求項１に記載の方法。
【請求項７】
さらに、
ネットワークプラットフォームマネージャによって、故障構成可能エレメントを有するアプリケーションのステータス変更と、故障構成可能エレメントを有するアプリケーションを有するプロセッササービスグループのステータス変更とを決定すること、および
ステータス変更を構成マネージャに通知すること

を含む請求項６に記載の方法。
【請求項８】
さらに、構成マネージャによって、ノード、プロセッササービスグループ、またはアプリケーションのステータス変更をホストに転送すること
を含む請求項７に記載の方法。
【請求項９】
さらに、
アプリケーションによって、特定のイベントを受信するインタレストをイベントマネージャに登録すること、および
イベントレシーバによって、登録されたアプリケーションに特定のイベントを送ること
を含む請求項１に記載の方法。
【発明の詳細な説明】
【０００１】
発明の技術分野
本発明は、一般に通信の分野に関する。より詳細には、本発明は、通信プラットフォームシステムおよび方法に関する。
【０００２】
発明の概要
本発明の一態様では、通信プラットフォームが、通信機能を実施するアプリケーションプログラムと、アプリケーションプログラムをサポートするサイトにある少なくとも１つのノード上で動作するオペレーティングシステムとの間のインタフェースを形成し、さらに、アプリケーションプログラムと通信ネットワークとの間のインタフェースを形成する。この通信プラットフォームは、ノード間の構成、監視、および管理機能を提供するように動作可能なネットワーク管理プロセスと、ノードの初期設定、構成、監視、および管理機能を提供するように動作可能なノード管理プロセスと、所定のイベントに応答してタスクの初期設定、終了、および分散を提供するように動作可能なイベントプロセスと、アプリケーションプログラム開発用の複数のプログラミングツールのライブラリを提供するように動作可能な共通プロセスと、メッセージ取扱い機能を提供するように動作可能な通信プロセスと、オブジェクトベース通信用の分散データベースリポジトリを提供するように動作可能な分散オブジェクトプロセスとを含む。
【０００３】
本発明の他の態様では、通信機能を実施するアプリケーションプログラムと、アプリケーションプログラムをサポートするサイトにある少なくとも１つのノード上で動作するオペレーティングシステムとの間のソフトウェアインタフェースを提供し、さらに、アプリケーションプログラムと通信ネットワークとの間のインタフェースを形成する方法が提供されている。この方法は、ノード間の構成、監視、および管理機能を提供するように動作可能なネットワーク管理プロセスを供給すること、ノードの初期設定、構成、監視、および管理機能を提供するように動作可能なノード管理プロセスを供給すること、所定のイベントに応答して、タスクの初期設定、終了、および分散を提供するように動作可能なイベントプロセスを供給すること、アプリケーションプログラム開発用の複数のプログラミングツールのライブラリを提供するように動作可能な共通プロセスを供給すること、メッセージ取扱い機能を提供するように動作可能な通信プロセスを供給すること、およびオブジェクトベース通信用の分散データベースリポジトリを提供するように動作可能な分散オブジェクトプロセスを供給することを含む。
【０００４】
本発明の他の態様では、通信機能を実施するアプリケーションプログラムと、アプリケーションプログラムをサポートするサイトにある少なくとも１つのノード上で動作するオペレーティングシステムとの間のソフトウェアインタフェースを提供し、さらにアプリケーションプログラムと通信ネットワークとの間のインタフェースを形成する方法が提供される。この方法は、サービスからノードを除去し、サービスにノードを復元し、サービスか

らアプリケーションを除去し、かつサービスにアプリケーションを復元するように動作可能なネットワークプラットフォームマネージャを提供すること、ノードを監視し、かつ故障ノードが回復するのを可能にするように動作可能なネットワークシステム保全マネージャを提供すること、通信プラットフォームに結合されたホストとインタフェースするように動作可能な構成マネージャを提供すること、ノード用の管理機能を提供するように動作可能なノードプラットフォームマネージャを提供すること、ノードプラットフォームマネージャの指示でプロセスを開始および停止するように動作可能なサービスマネージャを提供すること、およびノード間リンクを監視するように動作可能なノードシステム保全マネージャを提供することを含む。

【０００５】

本発明をより詳しく理解するために、添付図面を参照することができる。

【０００６】

発明の詳細な説明

アーキテクチャ概説

本発明の通信プラットフォーム（ＴＰ）１０は、分散、スケーラブル、障害回復通信アプリケーション１２の開発および実行をサポートするように構成されたソフトウェアシステムである。通信プラットフォーム１０は、ＵＮＩＸなどのコンピューティング環境用に開発された独自のツールセットを提供する。これらのツールは、通信プラットフォーム開発および実行時パッケージによって提供される、インタフェース、ライブラリ、および実行可能機能のセットだけでなく、分散、スケーラブル、障害回復アプリケーションを構成して管理するのに必要な概念構成要素のセットも含む。

【０００７】

図１に示されるように、通信プラットフォーム１０は、３つの異なるソフトウェア層１４～１６からなる。層＃１は、通信プラットフォームアプリケーションプログラミングインタフェース（ＡＰＩ）層１４であり、層＃２は、通信プラットフォームサービス層１５であり、層＃３は、システムインタフェース層１６である。通信プラットフォームＡＰＩ層１４は、通信ミドルウェアサービスからなる、通信プラットフォームサービス層１５にアクセスするための通信方法を提供する。通信プラットフォームサービス層１５は、例えばＵＮＩＸベースの通信システム用に最も一般的に必要とされるミドルウェアサービスを提供するソフトウェア層である。システムインタフェース層１６は、オペレーティングシステム（ＯＳ）ＡＰＩとネットワークリンクとからなる。システムインタフェース層１６は、プロセスおよびスレッド管理、メモリ管理、タイマ、ファイルシステム、通信、ハードウェアデバイスへのインタフェース、および他のシステム構成要素の機能を定義する。通信プラットフォーム１０は、より高レベルのクライアントアプリケーション１２を、オペレーティングシステムおよびネットワークからデカップルされることを可能にする。通信プラットフォーム１０を使用することによって、アプリケーションに代わってワークを実施するオペレーティングシステムやネットワークなどの下位サービスの複雑な構造をマスターする必要なく、開発者がアプリケーションを書くことができる。

【０００８】

図２は、通信プラットフォーム１０に関連する概念構成要素のブロック図である。最小の概念構成要素は、構成可能エレメント（ＣＥ）３０である。構成可能エレメント３０は、通信プラットフォーム１０によって管理されるＵＮＩＸ実行可能プログラムの１つまたは複数のコピーとして、通信プラットフォーム１０により定義される。例えば、構成可能エレメントは、リンクプロセス、データベース、グラフィカルユーザインタフェース、タイミングプロセス、照会プロセス、誤り取扱い機能などであってよい。構成可能エレメント３０は、アプリケーションプログラムの基本的な構築ブロックである。通信プラットフォーム１０がアプリケーション開発者に提供する最も基本的なサービスは、構成可能エレメント３０を作成し、構成し、監視するためのサービスである。構成可能エレメント３０は、ノード初期設定中の特定の点で、開始されるように構成することができる。Ｕｎｉｘ実行可能な構成可能エレメント表現は、スケーラビリティまたは冗長度のために、複数回実

行することができる。適切なサービスを提供するために必要な構成可能エレメントのインスタンスの数のしきい値を構成することができ、プロセスが故障した場合に、通信プラットフォーム１０によってインスタンスを自動的に再スタートすべきかどうかも構成することができる。

【０００９】

構成可能エレメントの構成可能な属性は、構成可能エレメントがスタートするレベルであるＲｕｎＬｅｖｅｌ（実行レベル）を含む。ＲｕｎＬｅｖｅｌには、ＰＲＥ＿ＭＩＮ、ＯＳ＿ＭＩＮ、ＩＮ＿ＳＶＣ、およびＰＯＳＴ＿ＩＮ＿ＳＶＣが含まれる。ＰＲＥ＿ＭＩＮ実行レベルは、構成可能エレメントが、ブート時間にサービス管理サブシステムによって自動的に作成されることを指定する。ＰＲＥ＿ＭＩＮ構成可能エレメントは、プラットフォームマネージャサブシステムによって監視されない。ＯＳ＿ＭＩＮは、ノードがＯＳ＿ＭＩＮに移行しているときに、構成可能エレメントが作成されることを指定する。ＩＮ＿ＳＶＣは、ノードがＩＮ＿ＳＶＣに移行しているときに、構成可能エレメントが作成されることを指定する。ＰＯＳＴ＿ＩＮ＿ＳＶＣは、ノードがＩＮ＿ＳＶＣ状態に移行したときに構成可能エレメントが作成されることを指定する。構成可能な他の属性は、実行可能機能のコピーが何回実行されるかを指定するＮｕｍｂｅｒＯｆＩｎｓｔａｎｃｅｓ（インスタンスの数）である。ＩｎＳｅｒｖｉｃｅＴｈｒｅｓｈｏｌｄ（サービス中しきい値）は、構成可能エレメントの状態をＥＮＡＢＬＥＤ（イネーブル）にするために、ＮｕｍｂｅｒＯｆＩｎｓｔａｎｃｅｓの他に何回アップして実行する必要があるかを指定する構成可能な属性である。インスタンスの数がこのしきい値を下回る場合、構成可能エレメント全体または構成可能エレメントの全てのインスタンスが除去される。構成可能エレメントの他の属性は、構成可能エレメントに送られるハートビートメッセージのためのスケジュールを指定する、ＨｅａｒｔｂｅａｔＳｃｈｅｄｕｌｅ（ハートビートスケジュール）である。各構成可能エレメントはまた、構成可能エレメントに送られる監査メッセージのためのスケジュールを指定する、ＡｕｄｉｔＳｃｈｅｄｕｌｅ（監査スケジュール）を有する。

【００１０】

構成可能エレメントセット（ＣＥＳｅｔ）２６は、通信プラットフォーム１０により、１つまたは複数のノード２４上にまとめて配置されるように構成される構成可能エレメントのグループと定義される。構成可能エレメントセットは、分散可能な構成要素である。通信プラットフォーム１０は、構成可能エレメント２６を直接的に管理することはできないが、その作成および配置をサポートする。構成可能エレメントセット２６は、アプリケーション２８の分散可能な、かつ／または複写可能な構成要素と見ることができる。

【００１１】

アプリケーション２８は、分散プログラムの全ての構成可能エレメント３０を完全に定義する、構成可能エレメントセット２６のグループとして定義される。通信プラットフォーム１０は、サイト２０内部でアプリケーション２８を管理するためのソフトウェアを提供する。分散可能な構成要素に関してアプリケーションの構成を定義することにより、分散アプリケーション用のソフトウェアを、それが動作するハードウェアとは無関係に定義することが可能になる。アプリケーションの構成可能エレメントセットは、いくつかの時点でサイト２０のノード２４に配置される。そのとき、アプリケーション２８のスケールおよび障害回復が、各構成可能エレメントセットをサポートするために使用されるノードの数に基づいて定義される。

【００１２】

ノード２４は、通信プラットフォーム１０が動作するサポートされたオペレーティングシステムのインスタンスとして定義される。通信プラットフォーム１０は、ノード２４でのプロセスを管理するソフトウェアを提供する。ノード２４は、耐障害性または非耐障害性、単一または多重プロセッサであってよい。通信プラットフォーム１０は、オペレーティングシステムのサービスを使用し、通常それが動作するハードウェアとは無関係である。通信プラットフォームは、ノード２４に関する構成情報をほとんど必要としない。ノード

は、名前と、独自のデバイス識別子とを提供することによってシステム内に構成される。

【００１３】

ノード２４は、その内部で開始される構成可能エレメントの順序付けを記述する、通信プラットフォームによってサポートされた動作状態を有する。動作状態には、ＨＡＬＴＥＤ、ＰＲＥ＿ＭＩＮ、ＯＳ＿ＭＩＮ、ＩＮ＿ＳＶＣ、およびＰＯＳＴ＿ＩＮ＿ＳＶＣが含まれる。ＨＡＬＴＥＤノード状態は、ノードのオペレーティングシステムがシャットダウンされていることを示す。ＰＲＥ＿ＭＩＮ状態は、ＯＳ＿ＭＩＮ状態の構成可能エレメントが開始される前に、開始されている必要がある構成可能エレメントを開始するために使用される。通信プラットフォームはまず、ノードに関してＰＲＥ＿ＭＩＮで動作するように構成された全ての構成可能エレメントを開始し、次いですぐに、ＯＳ＿ＭＩＮ状態で動作するように構成された構成可能エレメントを実行しはじめる。ＰＲＥ＿ＭＩＮで動作するように構成された構成可能エレメントは、ノードの状態に直接変化をもたらさない。ＯＳ＿ＭＩＮノード状態は、ＯＳ＿ＭＩＮ実行レベル用に構成された全ての構成可能エレメントが開始されて、ノードをＯＳ＿ＭＩＮ状態にするよう調整する。ＯＳ＿ＭＩＮノード状態用に構成された全ての構成可能エレメントが、構成可能な実行レベル移行状態を達成すると、ノードがＯＳ＿ＭＩＮに移行したといわれる。ＯＳ＿ＭＩＮノード状態が達成されているとき、実行レベル移行状態を下回るように状態を変更した構成可能エレメントがあると、通信プラットフォームは、ノードをＨＡＬＴＥＤノード状態にダウングレードする。シャットダウンノードを自動的に回復してもよい。ＩＮ＿ＳＲＶノード状態は、ＩＮ＿ＳＲＶ実行レベル用に構成された構成可能エレメントを調整する。ＩＮ＿ＳＲＶノード状態用に構成された全ての構成可能エレメントが、その構成可能な実行レベル移行状態を達成すると、ノードがＩＮ＿ＳＲＶに移行される。ひとたびＩＮ＿ＳＲＶノード状態が達成されると、実行レベル移行状態を下回るように状態を変更した構成可能エレメントがあると、通信プラットフォームは、ノードをＯＳ＿ＭＩＮノード状態にダウングレードする。ノードダウングレードを手動で行わなかった場合は、ノードの自動回復を行うことができる。ＰＯＳＴ＿ＩＮ＿ＳＲＶノード状態は、ノードがＩＮ＿ＳＲＶに移行された後、すぐに開始される構成可能エレメントを構成するために使用される。ノードがＩＮ＿ＳＲＶを達成すると、通信プラットフォームが、各ＰＯＳＴ＿ＩＮ＿ＳＲＶ構成可能エレメントを作成する。ＰＯＳＴ＿ＩＮ＿ＳＲＶ構成可能エレメントに関する状態変更は、ノード状態に影響を及ぼさず、繰り返して開始および停止することができる。ＰＯＳＴ＿ＩＮ＿ＳＲＶ構成可能エレメントを停止するプロセスが、ノードをより下のノード状態にダウングレードさせることはない。

【００１４】

サイト２０は、通信プラットフォームによって、分散アプリケーションを配置することができるノードのグループとして定義される。通信プラットフォームは、サイト２０内部のノード２４を管理する、プラットフォームマネージャとして知られる通信プラットフォームアプリケーションを提供する。サイトは、少なくとも１つのノードで構成することができる。多重ノードサイトでは、プラットフォームマネージャアプリケーションは、ノードのうちの２つについて、アクティブ／待機分散アプリケーションとして動作することができる。単一ノードサイトでは、プラットフォームマネージャアプリケーションは、ユーザ定義アプリケーションと共にその単一ノード内を動作するが、待機ノードによって提供される障害取扱い能力をもたずに動作する。サイトの管理は、プラットフォームマネージャによって提供される。

【００１５】

プロセッササービスグループ（ＰＳＧ）２２は、特定の構成可能エレメントセット２６が、冗長度のために配置されるノードのグループとして定義される。通信プラットフォーム１０は、アプリケーション内部にあるプロセッササービスグループを管理するための、ソフトウェアアプリケーションを提供する。適切なレベルのサービスを提供するために、構成可能エレメントセットが動作する必要があるノードの数を、通信プラットフォームユーザが識別することを可能にすることによって、プロセッササービスグループが冗長度をサ

ポートする。ノード、またはノード上を動作する構成可能エレメントセットの状態が変化すると、通信プラットフォーム１０は、適切なレベルのサービスが維持されていることを検証し、または構成されたアプリケーションステータスを変更する。

【００１６】

図３は、ハードウェア構成要素上にマップされる通信プラットフォーム１０の概念構成要素を使用するシステム４０構成を示す図である。

【００１７】

ハードウェア構成の点から見ると、ノードは、クライアントまたはサーバとして作用することができる、ネットワーク（イーサネットなど）内部のコンピュータプロセッサである。各ノードは、ノード上を動作するオペレーティングシステムの単一のインスタンスを有する。ノード内部のプロセッサは、オペレーティングシステムに依存しているため、互いに独立して動作することができない。サイトにある各ノードは、プラットフォームマネージャまたはアプリケーションノードとして分類することができる。サイトは、ホストに接続された１つのノードまたはノードのグループからなっていてよい。プラットフォームマネージャノードは、冗長メートを有する。プラットフォームマネージャとそのメートは、アクティブ／待機モードまたはロード共有モードで動作することができる。

【００１８】

システム４０は８つのノードを有し、２つのプラットフォームマネージャノード（アクティブ４２と待機４３）と、６つのアプリケーションノード４４～４９とを含む。呼出しがかけられた時間、または時間依存ルーティングに基づいて、電話呼出しを取り扱うためのアプリケーション５０が、ノードにわたって配置される。アプリケーション５０の構成可能エレメントセット５２および５４は、時間依存ルーティング機能を供給する分散構成要素である。各構成可能エレメントセット５２および５４は、特定の時間区域に関するＵＮＩＸ実行可能プログラムまたは構成可能エレメントのソフトウェアプロセスを含む。図示されるように、アプリケーション５０は、単一のアプリケーションノード４４～４９に常駐する必要はない。構成可能エレメントセットを、異なるノードにマップすることが望ましい。これにより、構成可能エレメントセットが構成されるノードの数を増加させることによって、アプリケーションを基準化することが可能になる。

【００１９】

通信プラットフォーム内部アーキテクチャを、論理的区画分けおよび物理的区画分けの点から記述した。論理的区画分けは、図４に示されるように、通信プラットフォームを異なる機能領域に分ける。各機能領域は、組み合わさって１つの特定のシステム機能を提供する、クラスの結合性グループを含む。物理的区画分けは、システムのコンテキストの具体的なソフトウェアおよびハードウェア分解を記述する。通信プラットフォーム１０によって提供されるサービスは、アプリケーションサービス６０とコアサービス６２との２つのグループに区画分けすることができる。アプリケーションサービスは、情報および問題報告（ＩＰＲ）／警報６４、統計６５、辞書６６、グラフィカルユーザインタフェース（ＧＵＩ）６７、およびホスト保守シミュレータ（ＨＭＳ）を実施する、サービスを含むことができる。ＩＰＲ／警報サービス６４は、誤り状態および他の関連するシステム情報を、システムユーザに知らせるための標準機構を提供する。統計サービス６５は、全システム測定データにアクセスし、かつ収集されたデータに基づいて報告を生成するための方法を提供する。辞書サービス６６は、データ記憶装置（持続、共有、または個人）をサポートして、データにアクセスするように構成されたクラスを提供する。グラフィカルユーザインタフェース６７は、ＧＵＩアプリケーションを構築するための基本抽象を提供し、システムユーティリティ、およびシステム自体、例えばｘターム（ｘｔｅｒｍ）ウィンドウやオペレーティングシステムユーティリティプログラムにアクセスする。ホスト保守シミュレータサービス７５は、システム内部にノードが１つしかないとき、または接続するためのホストがないときに、通信プラットフォームとインタフェースする方法を提供する。プラットフォームの制御および操作が可能になるのは、ホストを介してである。

【００２０】

コアサービス６２は、ネットワーク管理６８、ノード管理６９、分散オブジェクト７０、通信７２、共通機能７３、およびイベント取扱い７４を実施するサービスを含むことができる。ネットワーク管理サービス６８は、ネットワークアクティビティ、例えばノードおよびネットワークレベル障害処理の構成を指図する。ノード管理サービス６９は、ノードレベルプロセス、例えばノードステータス報告およびリンク管理を指図する。分散オブジェクトサービス７０は、多重処理環境における、オブジェクトベースの通信用分散データベースリポジトリを提供する。通信サービス７２は、プラットフォームの外部のプロセス間リンクにわたってメッセージを取り扱うための機構を提供する。共通サービス７３は、通信プラットフォーム上、またはその内部で動作するように構成されるプロセスの迅速な開発を補助するために、プログラミングツールのライブラリを提供する。イベントサービス７４は、タスクに重要な特定のアクションを開始し、終了し、および／または分散する能力を提供する。

【００２１】

最低限、通信プラットフォームは、コアサービス全てを提供する。高レベルアプリケーションは、これらのサービスを使用して、より低いレベルの機能を実施する。

【００２２】

さらに、図５に通信プラットフォームサービス、およびその依存関係を示す。開発者は、通信プラットフォームアプリケーションプログラムインタフェース１４を介してコアサービスおよびアプリケーションサービス全てにアクセスする。また開発者は、必要があれば、オペレーションシステム、ネットワーク、および第三者ソフトウェア／ハードウェアにアクセスすることもできる。プロセス間オブジェクトベース通信は、通信サービス７２によって取り扱われる。ほとんどのコアサービスおよびアプリケーションサービスが、通信サービス７２および共通サービス７３に依存してそれぞれの機能を実施する。グラフィカルユーザインタフェースサービス６７は、通信サービス７２のみに依存する場合がある。図５の矢印は、サービス間の依存関係を示す。

【００２３】

図６は、アプリケーション層８０およびコア層８２を含む、通信プラットフォーム１０の物理的区画分けの図である。コアサービス６２を含むコア層８２は、通信プラットフォームのあらゆるインスタンスのために存在する。コア層８２は、通信プラットフォームＡＰＩ１４、プロセス間通信機構、イベント機構、およびプラットフォーム管理を含む。通信プラットフォームアプリケーション層８０は、垂直および水平の区画を有する。垂直方向では、各通信プラットフォームアプリケーションプロセスが、アプリケーションの主セット８４の一部、またはそうでないものとして分類される。非主セットプロセスは、主セットプロセスに依存する。水平方向では、通信プラットフォームアプリケーション８０は、必要に応じて、または任意に分類される。任意のアプリケーションには、ＩＰＲ／警報パッケージ８６、データ収集パッケージ８７、辞書管理システムパッケージ８８、およびホスト保守シミュレーションパッケージ８９が含まれる。

【００２４】

以下、通信プラットフォームサービスをより詳細に説明する。

【００２５】

ネットワーク管理サービス

ネットワーク管理サービス６８は、ネットワークエレメントの共通管理ビューを提供する。これは、サービスからサーバノードを除去する、サービスにサーバノードを復元する、サービスからアプリケーションを除去する、サービスからアプリケーションを復元する、アプリケーションをイネーブルまたはディスエーブルする、分散アプリケーションのステータスを保守する、サーバノード状態およびステータスを保守する、アプリケーションステータス変更を報告するなど、ネットワークエレメントノード上の高レベル動作を実施する役割をする。ネットワーク管理サービス６８は、ネットワークプラットフォームマネージャ（ＮｅｔＰＭ）、ネットワークシステム保全サブシステム（ＮｅｔＳＩ）、および構成マネージャ（ＣｏｎｆｉｇＭｇｒ）を含む。図７Ａは、アクティブプラットフォームマ

ネージャノード１００を、対応する、またはメートされた待機プラットフォームマネージャノード１０２と共に示すブロック図である。各プラットフォームマネージャノードは、ネットワークプラットフォームマネージャ１０４、ネットワークシステム保全サブシステム１０６、および構成マネージャ１０８を含む。プラットフォームマネージャネットワークテストドライバ１１０が、ネットワークレベルテストを提供する。

【００２６】

ネットワークプラットフォームマネージャ（ＮｅｔＰＭＭａｉｎ）

ネットワークプラットフォームマネージャのためのクラス名は、ＮｅｔＰＭである。ＮｅｔＰＭは、プラットフォームリソースの管理機能を提供する役割をする。プラットフォームは、テレフォンカードやクレジットカードの検証など、特定のサービスに関する処理能力を提供する、複数のノードまたはサーバからなる分散システムである。サーバによって提供されるサービスは、ノードに常駐する構成可能エレメントによって決定される。ＮｅｔＰＭは、プラットフォームに関連する全ての構成データを管理する。構成データは、サーバのＴＣＰ／ＩＰアドレスなどハードウェアに関する情報、サーバや照会ステータスなどステータス情報、アプリケーションタイプなどソフトウェア構成情報、ノード名、および個々の構成要素エレメントに関する情報を含む。

【００２７】

ＮｅｔＰＭは、以下の構成情報を保守する。これらの情報は、初期設定中にＮｅｔＰＭによって収集される。

【００２８】

・構成可能エレメント記述子情報

プラットフォームの各構成可能エレメントに関する構成情報を提供する。ＮｅｔＰＭは、様々なタイプの構成可能エレメントに関する情報を含むディスクファイルからこれらを検索する。

【００２９】

・アプリケーション情報

各アプリケーション（サービス）に関する構成情報を提供し、アプリケーションのステータスを計算する際に使用できる。ＮｅｔＰＭは、プラットフォーム内の全てのアプリケーションに関する情報を含むディスクファイルからこの情報を検索する。

【００３０】

・プロセッササービスグループ情報

プロセッササービスグループに関する構成情報を提供し、プロセッササービスグループステータスを計算する際に使用することができる（プロセッササービスグループは、同じアプリケーションを提供するプロセッサのグループ、すなわちＣＣＤ、ＣＣＬを表す）。ＮｅｔＰＭは、プラットフォーム内の全てのプロセッササービスグループに関する情報を含むディスクファイルからこれらを検索する。

【００３１】

・サーバ情報

プラットフォーム内の全てのサーバに関する特定の情報を提供する。ＮｅｔＰＭは、ＣｏｎｆｉｇＭｇｒにこの情報を要求して、そこから検索する。ＣｏｎｆｉｇＭｇｒはまず、プラットフォームマネージャノードに関するサーバ情報をＮｅｔＰＭに提供する。その後ＣｏｎｆｉｇＭｇｒが、現行サーバがアクティブプラットフォームマネージャであると判定した場合、プラットフォーム内の残りのサーバに関する情報を局所ＮｅｔＰＭに提供する。そうでない場合（待機プラットフォームマネージャ）は、ＮｅｔＰＭはそれらの情報を、ＣｏｎｆｉｇＭｇｒからでなく、メートから検索する。

【００３２】

これらの情報を収集している間に誤りが検出された場合、ＮｅｔＰＭは適切なＩＰＲを発行して、終了する。

【００３３】

ＮｅｔＰＭは、全ての構成データを管理するためにＮｅｔＭＡＰオブジェクトを使用する

。ＮｅｔＰＭはまた、持続辞書を使用して、プラットフォームマネージャリセットにわたってサーバステータス、照会ステータス、およびスケジュールされたアクション情報を保持する。この辞書を管理するために、ディスクファイル辞書オブジェクトが使用される。ＮｅｔＰＭは、２つのプラットフォームマネージャサーバ間の構成データの保全性を保守する役割をする。ＮｅｔＰＭは、持続辞書、データベース等化、および監査を使用して、データの保全性を保守する。

【００３４】

アプリケーションステータスは、プロセッササービスグループステータスに基づいて決定される。プロセッササービスグループステータスの決定には、以下の基準が使用される。

【００３５】

・ＰＳＧ＿ＤＩＳＡＢＬＥＤ

プロセッササービスグループ内の少なくとも設定数のサーバが、ディスエーブル状態にある。

【００３６】

・ＰＳＧ＿ＩＮＡＣＴＩＶＥ

各プロセッササービスグループ内の少なくとも１つのサーバが、待機状態にあり、どれもアクティブ状態でない。

【００３７】

・ＰＳＧ＿ＡＣＴＩＶＥ＿ＭＩＮＩＭＡＬ

プロセッササービスグループ内のいくつかのサーバのみが、アクティブ状態にある。

【００３８】

・ＰＳＧ＿ＡＣＴＩＶＥ

プロセッササービスグループ内の設定数のサーバが、アクティブ状態にある（注：この数は、ＰＳＧ＿ＡＣＴＩＶＥ＿ＭＩＮＩＭＡＬにおいてアクティブでなければならないサーバの数よりも大きくなる）。

【００３９】

アプリケーションステータスは、以下の基準を使用して導出することができる。

【００４０】

・ＡＰ＿ＤＩＳＡＢＬＥＤ

所与のアプリケーションに関して、少なくとも設定数のプロセッササービスグループが、ＰＳＧ＿ＤＩＳＡＢＬＥＤのステータスを有する。

【００４１】

・ＡＰ＿ＩＮＡＣＴＩＶＥ

所与のアプリケーションに関して、少なくとも１つのプロセッササービスグループが、ＰＳＧ＿ＩＮＡＣＴＩＶＥのステータスを有し、どのプロセッササービスグループも、ＰＳＧ＿ＡＣＴＩＶＥのステータスを有しない。

【００４２】

・ＡＰ＿ＡＣＴＩＶＥ＿ＭＩＮＩＭＡＬ

所与のアプリケーションに関して、設定数のプロセッササービスグループが、ＰＳＧ＿ＡＣＴＩＶＥ＿ＭＩＮＩＭＡＬのステータス、またはそれより高い（ＰＳＧ＿ＡＣＴＩＶＥ）ステータスを有する。

【００４３】

・ＡＰ＿ＡＣＴＩＶＥ＿ＰＡＲＴＩＡＬ

所与のアプリケーションに関して、設定数のプロセッササービスグループが、ＰＳＧ＿ＡＣＴＩＶＥ＿ＭＩＮＩＭＡＬのステータス、またはそれより高い（ＰＳＧ＿ＡＣＴＩＶＥ）ステータスを有する（注：ＡＰ＿ＡＣＴＩＶＥ＿ＰＡＲＴＩＡＬ状態に必要なプロセッササービスグループの数は、ＡＰ＿ＡＣＴＩＶＥ＿ＭＩＮＩＭＡＬにおけるプロセッササービスグループの必要数よりも大きい）。

【００４４】

・ＡＰ＿ＡＣＴＩＶＥ

所与のアプリケーションに関して、設定数のプロセッササービスグループが、ＰＳＧ＿ＡＣＴＩＶＥのステータスを有する（注：ＡＰ＿ＡＣＴＩＶＥ状態に必要なプロセッササービスグループの数は、ＡＰ＿ＡＣＴＩＶＥ＿ＰＡＲＴＩＡＬにおけるプロセッササービスグループの必要数よりも大きい）。

【００４５】

ＮｅｔＰＭは、各サーバノードに関するステータス変更のトラックを保持し、それらを得るとき、プロセッササービスグループのステータスを決定し、変更があった場合は、ノードに関する新しいアプリケーションステータスを決定し、この変更をＣｏｎｆｉｇＭｇｒに知らせる。

【００４６】

ＮｅｔＰＭは、アプリケーションステータスに関する送信請求された更新および自律的更新を提供する。自律的更新では、まず、アプリケーションプログラムが、ＮｅｔＰＭに機能を登録して、特定のアプリケーションタイプ（ＣＣＤまたはＣＣＬ）に関する更新を受け取る。ＮｅｔＰＭが、ＮｏｄｅＰＭからサーバまたは照会ステータスの変更を受け取ると、アプリケーションステータスが計算され、登録された機能が、古いアプリケーションステータスと新しいアプリケーションステータスで呼び出される。アプリケーションステータスは送信請求されることができ、その間、ＮｅｔＰＭは、要求するプロセスに、ＮｅｔＭＡＰに保存されたアプリケーションステータスの最新の計算値を返す。

【００４７】

ＮｅｔＰＭは、特に２つの別名オブジェクトの使用によって、ＮｅｔＰＭと通信することを望む他のプロセスに、２組のルーティングオプションを提供する。ＮｅｔＰＭは、局所および大域アクティブ待機オプションを提供する。局所オプションでは、全てのＮｅｔＰＭクライアント要求が、クライアントオブジェクトとして同じノードにあるＮｅｔＰＭサーバオブジェクトに送られる。大域アクティブ待機オプションでは、全てのＮｅｔＰＭクライアント要求が、大域的に（すなわち、場合によってはノード間で）使用可能なアクティブＮｅｔＰＭサーバオブジェクトに送られる。

【００４８】

ＮｅｔＰＭは、多数のサーバ構成データに関して、１組の読取機能および書込機能を提供する。これらには、スケジュールアクションデータ、プラットフォームマネージャアクティブステータスデータ、サーバステータスデータなどのための、読取機能／書込機能が含まれる。ＮｅｔＰＭは、構成可能エレメント記述データに対する直接読取／書込動作を提供しない。

【００４９】

ＮｅｔＰＭはまた、大部分のサーバ構成データを初期設定するための機能を提供する。この機能は、入力としてＳｅｒｖｅｒＩｎｆｏＭｓｇオブジェクトを予想する。

【００５０】

ＮｅｔＰＭは、特定の構成アクション（グレースフル中止、即時中止、グレースフルダウングレード、および復元）が特定のサーバ上で行われるようにする１組の機能を提供する。

【００５１】

ＮｅｔＰＭは、特定のサーバについてサーバステータスを変更することができる機能を提供する。

【００５２】

ＮｅｔＰＭは、特定のサーバ上での照会処理をイネーブルする、かつディスエーブルするための機能を提供する。

【００５３】

ＮｅｔＰＭは、サーバステータスを「報告」し、ステータス変更を照会する複数の機能を提供する。これらのルーチンは、ＮｅｔＭＡＰに新しいステータス情報を保存し、ＣｏｎｆｉｇＭｇｒソフトウェアにその変更を通知し、プラットフォーム内の全てのＮｏｄｅＰＭソフトウェアに、その変更をブロードキャストする。

【００５４】
ＮｅｔＰＭは、サーバネットワーク内部の時間同期をする役割もする。時間同期は、図７Ｂに示されるように、３つの主要部からなる。第１の部分は、アクティブプラットフォームマネージャ１００が、その局所時間をホストの時間と等しくする部分である。これは、ホスト（１１０）の時間を使用可能な形式に変換すること、およびプラットフォームマネージャノード１００および１０２上のＮｏｄｅＰＭ１１２に、ａｄｊｔｉｍｅ機能を実施するように知らせて、ホスト１１０と一致するようにクロックを調節することを含む。ＮｅｔＰＭ１０４はまた、時間メッセージを受け取ったときに、ホストチッカークラスに新しいホスト時間を知らせる。ついで、ｘｎｔｐプロセス１２０が、アプリケーションノード（１２１）の時間を、プラットフォームマネージャノード１００および１０２の時間と同期させる。各プラットフォームマネージャノード１００および１０２は、ｘｎｔｐマスタ時間源として構成される。アプリケーションノード１２１上のｘｎｔｐデーモンスレーブ１２２が、プラットフォームマネージャノード１００および１０２上のマスタｘｎｔｐデーモン１２０の１つを選択して、同期を保つ。最後に、非送信請求時間設定メッセージが、ホスト１１０から受け取られると、ネットワークの時間が、受け取られた時間と等しくなる。

【００５５】
最後に、ＮｅｔＰＭ１０４は、新たにブートされたノードに、プラットフォーム内の全てのサーバの持続サーバ構成データを提供する機能を提供する。ＮｅｔＰＭ１０４は、構成可能エレメントである。ＮｅｔＰＭ１０４は、ＮｏｄｅＰＭがＮｅｔＰＭの実行を制御するために必要とする、Ｒｅｍｏｖｅ（除去）、Ｒｅｓｔｏｒｅ（復元）、およびＧｅｔＳｔａｔｕｓ（ステータス取得）というカプセル化されていない操作を提供する。監査タイマが始動すると、ＮｅｔＰＭＴｉｍｅｒＨａｎｄｌｅｒが呼び出される。これが提供サービスループを打ち切り、ＮｅｔＰＭ機能ＳｅｔｔｉｍｅＴｏＶｅｒｉｆｙを呼び出して、監査を開始する。

【００５６】
ＮｅｔＰＭ１０４は、それ自体の制御のスレッドを伴うオブジェクトである。ＮｅｔＭＡＰリストを構築した後、要求を待つ無限ループに入る。サーバのサービスまたは照会ステータスの変更があるときはいつでも、ＮｅｔＰＭ１０４は、ＣｏｎｆｉｇＭｇｒ１０８に通知する。また、ＮｅｔＰＭ１０４は、これらのステータス変更を、プラットフォーム内の全てのＮｏｄｅＰＭ１１２に送る。ＮｅｔＰＭ１０４は、特定のＮｏｄｅＰＭ１１２に、照会処理をイネーブル、またはディスエーブルするように通知する。ＮｅｔＰＭ１０４は、サービスステータス同期機能を提供する。ＮｅｔＰＭ１０４は、プラットフォーム内のサーバに関するＩＰＵ情報を構築し、その情報を、ＢｏｏｔＮｏｔｉｆｙメンバ機能内の特定のＮｏｄｅＰＭ１１２に渡す。ＮｅｔＰＭは、サービスのダウングレードに対する全ての構成要求（すなわち、ＧｒａｃｅＤｏｗｎ、ＩｍｍｅｄＤｏｗｎ、ＧｒａｃｅＨａｌｔ、およびＩｍｍｅｄＨａｌｔ）において、サーバの所望の状態を、特定のＮｏｄｅＰＭ１１２に通知する。サーバ復元が要求されたとき、ＮｅｔＰＭ１０４は、いくつかのことを行う。まずＮｅｔＰＭ１０４は、特定のＮｏｄｅＰＭ１１２からサーバの現行ステータスを取得する。第２に、返されたステータスがサービス外／最小ソフトウェアである場合、関連するＮｏｄｅＳｐｅｃＩｎｆｏを、特定のＮｏｄｅＰＭ１１２に送る。第３にＮｅｔＰＭ１０４は、関連する構成可能エレメント記述子情報を、特定のＮｏｄｅＰＭ１１２に送る。最後に、ＮｅｔＰＭが特定のＮｏｄｅＰＭにサービスを復元するように知らせる。

【００５７】
ネットワークシステム保全（ＮｅｔＳＩＭａｉｎ）
ネットワークシステム保全（ＮｅｔＳＩ）サブシステム１０６は、ネットワークエレメントに関する監視および回復操作を提供する。これは、ネットワーク監視および回復を実施する役割をする。ネットワークシステム保全によって実施される操作には以下のものがある。

【００５８】
－　プラットフォームマネージャアクティブ／待機ステータス監視
－　ノード故障報告相関関係
－　故障ノード回復アクション
ネットワークシステム保全のクラス名は、ＮｅｔＳＩである。ＮｅｔＳＩ１０６は、プラットフォームマネージャに関するネットワークシステム保全性を管理する。ＮｅｔＳＩ１０６は、障害ノード上のＮｏｄｅＳＩから、サーバダウングレードおよび通信障害の通知を受け取る。ＮｅｔＳＩ１０６は、ＮｏｄｅＳＩによって与えられるデータに基づき、どのようなアクションをとるべきか決定する。ノードがダウングレードを示す場合、ＮｅｔＳＩは適切なアクションをとって、ネットワークレベルから所望のダウングレード状態にノードをダウングレードする。ノードが通信障害を示す場合は、ＮｅｔＳＩ１０６は前もって受け取ったデータから、（あるとしたら）どのノードが障害になっているか判定し、必要であれば、その障害ノードをダウングレードするようにアクションを行う。ＮｅｔＳＩがノードに対してダウングレードが必要であると判定したとき、ＮｅｔＳＩは、ダウングレードを行うために適切なＮｅｔＰＭ操作を呼び出す。ＮｅｔＳＩは、アクティブステータスの変更が必要な場合は、アクティブステータスを設定するために適切なＮｅｔＰＭ操作を呼び出す。ダウングレードを行うためにＮｅｔＰＭが呼び出された後、特定のノードに関してステータスが変更していることをＣｏｎｆｉｇＭｇｒに通知する。これにより、ノードがダウングレードされたという通知をホストがすぐに受けることが可能になる。次いで、ＮｅｔＳＩは、ステータス変更とその理由を示す、ネットワーク構成報告にエントリを書き込む。ＮｅｔＳＩは、ノードの現行状態に基づいて、正当なサービス状態にノードをダウングレードする。
【００５９】
ＮｅｔＳＩは、通信障害リストを含む。このリストは、受け取られた各通信障害報告の報告サーバノード名および問題サーバノード名を保持する。通信障害報告が受け取られると、リストは、問題ノードに関する他の報告をサーチされる。見つからなかった場合、障害情報がリストに追加される。ＮｅｔＳＩはまた、ダウンステータス情報リストも含む。ノードがサービス外であることをＮｏｄｅＰＭが示し、ノードが休止されていることをＮｅｔＰＭステータスが示さないとき、ダウンステータス情報エントリが、休止されたＩＰＵのノード名を伴って作成される。タイマが作成され、ダウンステータス情報がリストに追加される。後に、（タイマが切れる前に）ＮｏｄｅＰＭがそのノードに関してより高いステータスを示す場合、ダウンステータス情報エントリがリストから消去され、さらなるアクションは行われない。
【００６０】
ＮｅｔＳＩは、両ＰＭのステータス状態を定期的に監査する。無効状態が存在する場合、ＮｅｔＳＩは、アクティブステータスを是正状態に設定することによって、ステータスを是正することを試みる。他のプロセスも、プラットフォームマネージャステータス状態を監査するようＮｅｔＳＩに要求することができる。
【００６１】
ＮｅｔＳＩは、「両方に送る（ｓｅｎｄ　ｔｏ　ｂｏｔｈ）」ロード共有概念の下で動作する。プラットフォームマネージャノードが共に作動状態にある場合、各プラットフォームマネージャノードにおける各ＮｅｔＳＩプロセスが、ＮｏｄｅＳＩ要求を受け取る。各ＮｅｔＳＩプロセスは、プラットフォームのアクティブ／待機状態および障害サーバに基づいて、要求を取り扱うべきかどうか判定する。アクティブプラットフォームマネージャのＮｅｔＳＩプロセスは通常、必要とされるアクションを行い、一方、待機プラットフォームマネージャは情報を廃棄する。しかし、障害ノードがアクティブプラットフォームマネージャである場合、待機プラットフォームマネージャは、（有効であれば）それ自体をアクティブに設定し、要求アクションをとって、他のプラットフォームマネージャノードをダウングレードする。
【００６２】

ＮｅｔＳＩ操作が呼び出されるたびに、ＮｅｔＳＩはまず、アクティブプラットフォームマネージャであるか待機プラットフォームマネージャであるかを判定する。アクティブである場合、ターゲットノードがそれ自体であり、かつメートがサービス中であるときを除いて、ＮｅｔＳＩは、全ての状態に関する要求を処理する。待機状態の場合、ＮｅｔＳＩは、ターゲットノードがメートであるときを除いて、全ての状態に関する要求を廃棄する。

【００６３】

初期設定中、ＮｅｔＳＩは、ＮｏｄｅＰＭからメートのノード名、およびそれ自体のサーバおよびメートサーバのサーバ識別子を要求する。情報を要求する前、ＮｅｔＳＩは、ＮｏｄｅＰＭのステータスについてポーリングを行い、ＮｏｄｅＰＭが読み取られて提供されるまでは、ノード名およびサーバ識別子を要求しない。ＮｅｔＳＩは、この情報が適切に受け取られるまでは、サービスを提供するための準備をされない。

【００６４】

ＮｅｔＳＩは、コマンドラインパラメータＤＷＮ＿ＲＰＴ＿ＦＩＬＥを使用して、ネットワーク構成（ダウングレード）報告ファイル名の名前を得る。このパラメータが指定されない場合、どの報告エントリもダウングレードから作成されない。

【００６５】

図７Ｃおよび７Ｄを参照すると、ノード管理とネットワーク管理のプロセス相互作用が示される。定期監視モニタ（ＣｏｎＭｏｎ）は、アプリケーションノード１３６上を動作するオブジェクトのインスタンスである。ＣｏｎＭｏｎ１３２は、障害プロセスまたは故障構成可能エレメントを検出し、サービス管理プロセスプログラム１３４に通知する。サービス管理プロセス１３４は、構成可能エレメント故障によってプロセスがしきい値レベルを下回るかどうかを判定する。下回らない場合、サービス管理プロセス１３４は、構成可能エレメントを再始動する。しかし、構成可能エレメントがしきい値レベルを下回る場合は、サービス管理プロセス１３４は、構成可能エレメントステータス変更メッセージを生成して、その通知をＮｏｄｅＳＩ１３０に転送する。ＮｏｄｅＳＩは、構成可能エレメントステータス変更を、ＮｏｄｅＰＭ１１２に転送する。ＮｏｄｅＰＭ１１２は、構成可能ステータス変更がノードの実行レベルに影響を及ぼして、ノードのダウングレードを行う可能性があるかどうかを判定する。ノードが除去される場合、ＮｏｄｅＰＭ１１２は、サービス管理プロセス１３４に命令を提供して、ダウングレードされた状態を達成するのに必要な全ての構成可能エレメントを除去する。ＮｏｄｅＰＭ１３４は、ＮｅｔＰＭ１０４にノードステータス変更を通知した。ＮｅｔＰＭ１０４は、ノードステータス変更が、プロセッササービスグループおよびアプリケーションステータスに影響を及ぼすかどうかを判定するために計算を行う。ＮｅｔＰＭの計算はまた、ノードをサービス中から最小設定（ｍｉｎ－ｓｅｔ）へ移し、それを再び復元するなどの自動アクションを、ノードに対して行うべきかどうかを判定する。ノードが移される場合、ノードステータス変更は、ＮｅｔＰＭからＣｏｎｆｉｇＭｇｒ１０８に転送される。ＣｏｎｆｉｇＭｇｒは、ノード、プロセッササービスグループ、およびアプリケーションに関する状態変更をホスト１４０に通知する。これらの状態変更は、報告に表示または印刷することができる。

【００６６】

特に、各ＮｅｔＳＩは、ダウングレード要求を取り扱うべきかどうか判定する。要求を取り扱う場合、ターゲットサーバのステータスが検索される。ターゲットサーバがまだ休止されていない場合、サーバは、ＩＰＵステータスに基づいて、適切なステータスにダウングレードされる。ＩＰＵステータスがサービス外である場合、ＮｅｔＳＩは、ＮｅｔＰＭの即時休止操作を呼び出して、ターゲットノードを自動休止または手動休止する。ＩＰＵステータスがサービス外最小（ＯＳ－ＭＩＮ）である場合、ＮｅｔＳＩは、ＯＳ－ＭＩＮにターゲットノードをダウングレードするために、ＮｅｔＰＭの即時ダウングレード操作を呼び出す。ＩＰＵステータスがサービス中にディスエーブルされている場合、ＮｅｔＳＩは、ターゲットノードに関する照会ステータスをディスエーブルするように、ＮｅｔＰＭのディスエーブル照会操作を呼び出す。いずれにせよ、ターゲットノードがアクティブ

プラットフォームマネージャである場合、ＮｅｔｓＳＩがアクティブステータスを更新する。また、ターゲットノードが局所サイトの一部である場合、ＮｅｔＳＩは、ステータス変更が生じたことをＣｏｎｆｉｇＭｇｒを介してホストに知らせ、ターゲットサーバのプロセッササービスグループを回復すべきであると判定した場合は、プロセッササービスグループの回復を（ＣｏｎｆｉｇＭｇｒを介して）開始する。次いで、ＮｅｔＳＩは、ネットワーク構成報告ファイルにエントリを書き込み、ノードが障害を報告したためにステータス変更が生じたことを示す。

【００６７】

ＮｏｄｅＳＩは、２つのノード間で起こった通信障害をＮｅｔＳＩに知らせる。ＮｅｔＳＩは、以前の情報受信（があればそれ）に基づいて、障害を記憶するか、障害にアクションを起こす。各ＮｅｔＳＩは、報告ノードおよび問題ノードのステータスを判定する。いずれかのサーバが休止されている場合、データの保全性を保証することができないので、通信障害報告が廃棄される。どちらのサーバも休止されていない場合、通信障害リストが、問題ノードに関する他の報告をサーチされる。問題ノードに関する報告が見つからなかった場合、通信障害リストエントリが、サーバ情報と共にリストに追加される。問題ノードの他の報告が見つかり、他の報告サーバがそれを報告していた場合、その問題サーバが、ダウングレード処理を行うためにセットアップされる。サーバをダウングレードすべきかどうかについて決定が下されると、ＮｅｔＳＩは、（アクティブ状態、およびターゲットサーバがそれ自体であるか否かに基づいて）それを取り扱うべきかどうか判定する。ダウングレードを取り扱うべき場合、ＮｅｔＳＩは、ＮｅｔＰＭの即時休止操作を呼び出して、問題ノードを自動休止または手動休止する。休止すべきサーバがアクティブＰＭである場合は、それに応じて、ノードを休止する前に、アクティブステータスを更新する。また、ターゲットが局所サイトの一部である場合、ＮｅｔＳＩは、ステータス変更が生じていることを、ＣｏｎｆｉｇＭｇｒを介してホストに知らせ、ターゲットサーバのプロセッササービスグループを回復すべきであると判定された場合、（ＣｏｎｆｉｇＭｇｒを介して）プロセッササービスグループの回復を開始する。ＮｅｔＳＩはまた、ネットワーク構成報告ファイルにエントリを書き込み、通信障害により休止されたことを示す。

【００６８】

構成マネージャ（ＣｏｎｆｉｇＭｇｒ）

構成管理サブシステム（クラス名：ＣｏｎｆｉｇＭｇｒ）は、ＳＣＰホストとサーバ構成要素との間に制御インタフェースを提供する。サーバネットワーク上で行うことのできるすべての動作は、このインタフェースに定義される。構成管理サブシステムは、以下の特徴を与える。

【００６９】

－　ホストとサーバとの間の制御メッセージインタフェース
－　有効な動作に対する状態機械
－　要求によってネットワーク管理を駆動する
－　動作のタイミング／タイムアウトを制御する

ＣｏｎｆｉｇＭｇｒは、プラットフォームマネージャに対する、サーバ構成制御を管理する。ＣｏｎｆｉｇＭｇｒは、ＣＯＮＦＩＧＣＴＬ、ＭＡＩＮＴ、ＡＰＬＣＴＬ、ＲＯＵＴＩＮＧＣＴＬの論理リンク上で送信されたホストメッセージを受け取り、それぞれを、そのメッセージｉｄおよびタイプに基づいて処理する。ホストが応答または報告を送るように要求する場合、ＣｏｎｆｉｇＭｇｒは、必要な応答を決定し、必要な報告情報を取り出し、それをホストに送り返す。ＣｏｎｆｉｇＭｇｒは、以下のメッセージを扱う。

【００７０】

・ＡＰＰＬ＿ＳＴＡＴＵＳ＿ＭＳＧ
・ＡＳＰＥＣ＿ＭＳＧ
・ＣＯＮＦＩＧＵＲＥ＿ＳＥＲＶＥＲ＿ＭＳＧ
・ＰＳＧ＿ＩＮＦＯ＿ＭＳＧ
・ＰＳＧ＿ＳＴＡＴＵＳ＿ＭＳＧ

・ＱＵＥＲＹ＿ＰＲＯＣＥＳＳＩＮＧ＿ＭＳＧ
・ＲＥＳＥＴ＿ＳＥＲＶＥＲ＿ＭＳＧ
・ＲＯＵＴＩＮＧ＿ＩＮＦＯ＿ＭＳＧ
・ＳＣＨＥＤ＿ＡＣＴＩＯＮ＿ＣＴＬ＿ＭＳＧ
・ＳＥＲＶＥＲ＿ＩＮＦＯ＿ＭＳＧ
・ＳＥＲＶＥＲ＿ＳＴＡＴＵＳ＿ＭＳＧ
・ＴＥＳＴ＿ＳＥＲＶＥＲ＿ＭＳＧ
・ＴＩＭＥ＿ＭＳＧ
ＣｏｎｆｉｇＭｇｒはまた、ホストからサーバおよび時間の情報を取り出すための動作を、プラットフォームマネージャに提供する。ＣｏｎｆｉｇＭｇｒはまた、サーバのステータス変化をホストに通知する動作も提供する。ホスト指令メッセージの処理中、ＣｏｎｆｉｇＭｇｒは、ホストからの応答または特定のサーバからの状態変化を待機しなければならない時がある。ＣｏｎｆｉｇＭｇｒは、これらの待機に関して非ブロッキング原理を使用する。ＣｏｎｆｉｇＭｇｒは、停止してイベントの発生を待機するのではなく、所望の応答またはステータスを保留待ち行列（ＰｅｎｄｉｎｇＱｕｅｕｅ）上にセーブし、通常の、別のホストメッセージの処理を継続するか、またはクライアントにサービスを提供する。所望の応答またはステータスが発生すれば、適切な手続きが呼び出されて、ホスト指令メッセージの処理が再開される。指定の制限時間内に所望の応答が到着しない、あるいは所望のステータスが発生しない場合は、故障手続きを呼び出して、ホスト指令メッセージの処理をクリーンアップし、必要に応じてＩＰＲを発行する。

【００７１】
ホスト指令メッセージの処理に加え、ＣｏｎｆｉｇＭｇｒには、ステータス変化の発生時に、ホストに通知することも必要とされる。ＣｏｎｆｉｇＭｇｒは、ステータス変化を通知されると、ステータス保留待ち行列を検査して、それがステータス変化の発生を待機しているかどうかを判定する。待機している場合は、保留待ち行列成功動作が行われる。そうでない場合は、ＣｏｎｆｉｇＭｇｒは、ホストにサーバステータスメッセージを送る。ホスト応答メッセージの処理中、ＣｏｎｆｉｇＭｇｒは、ホスト応答保留待ち行列（ＨｏｓｔＰｅｎｄＱｕｅｕｅ）を検査して、それが応答を待機しているかどうかを判定する。待機している場合は、保留待ち行列成功動作が行われる。そうでない場合は、ＣｏｎｆｉｇＭｇｒは、ホストからのその応答メッセージを破棄する。プラットフォームマネージャノードがＯＳ－ＭＩＮ状態にブートされているとき、それはそのメートを監査し、そのメートのステータスを決定する。メートのプラットフォームマネージャノードが存在しない場合は、メートのステータスは自動的に休止に設定される。同様の監査は、サービスサーバノード（ＰＭ以外のノード）上でも行われ、それらのステータスが決定される。

【００７２】
ＣｏｎｆｉｇＭｇｒは、サブシステムが登録して特定のアプリケーションに対するルーティング情報を提供することのできる登録機能を有する。ホストがアプリケーションに関するルーティング情報を要求すると、ＣｏｎｆｉｇＭｇｒは、登録された適切なサブシステム（もし存在すれば）に、ルーティング情報の提供を要求する。

【００７３】
構成サーバメッセージ（ＣｏｎｆｉｇＳｅｒｖｅｒＭｓｇｓ）は、行われるサービス（すなわち休止、ダウングレード、復元、ブート）の性質のために、特別な処理を要求する。ホストメッセージが両方のプラットフォームマネージャサービスに送られるため、１つのプラットフォームマネージャノードだけがその要求を処理することが保証されるように配慮されなければならない。これには、プラットフォームマネージャノードおよびそのメートのサーバ状態を検査することが必要となる。プラットフォームマネージャノードのサーバ状態、ならびに、ＣｏｎｆｉｇＳｅｒｖｅｒ要求がプラットフォームマネージャノードへのものか、そのメートへのものか、それともサービスサーバへのものであるかとに基づいて、異なる動作がとられる。２つの有限状態機械（ＰＭＣｆｇＳｖｒＦＳＭおよびＳｖｃＣｆｇＳｖｒＦＳＭ）が、異なる状態によって駆動される全ての動作を管理する。

【００７４】
ＰＭＣｆｇＳｖｒＦＳＭは、プラットフォームマネージャアプリケーションサーバに対する復元、休止、再同期化、ダウングレード、ブートを扱う有限状態機械である。この機械は、要求がそれ自体へのものか、それともそのメートへのものかと、それ自体のステータスと、そのメートのステータスと、要求されたイベント（休止、ダウングレード、復元など）とに基づいて、その要求を処理する。検査されるプラットフォームマネージャサーバ状態は、Ｈａｌｔｅｄ（自動）、Ｈａｌｔｅｄ（手動）、ＸＯＳ－ＭＩＮ、ＡＯＳ－ＭＩＮ（自動）、ＭＯＳ－ＭＩＮ（手動）、Ｉｎ－Ｓｖｃである。Ｉｎ－Ｓｖｃの場合、アクティブ／待機ステータスが検査されて、サーバがアクティブであるか待機であるかが判定される。有効なイベントは、復元（Ｒｅｓｔｏｒｅ）、グレースフル休止（Ｇｒａｃｅｆｕｌ　Ｈａｌｔ）、即時休止（Ｉｍｍｅｄｉａｔｅ　Ｈａｌｔ）、グレースフルダウングレード（Ｇｒａｃｅｆｕｌ　Ｄｏｗｎｇｒａｄｅ）、即時ダウングレード（Ｉｍｍｅｄｉａｔｅ　Ｄｏｗｎｇｒａｄｅ）、グレースフルブート（Ｇｒａｃｅｆｕｌ　Ｂｏｏｔ）、即時ブート（Ｉｍｍｅｄｉａｔｅ　Ｂｏｏｔ）、ホスト再同期化（Ｈｏｓｔ　Ｒｅｓｙｎｃ）である。
【００７５】
イベントは、どのプラットフォームマネージャノードが要求を処理することになるかを決定するために重要である。復元が要求される場合、通常、復元されることになるプラットフォームマネージャノードが、その復元を処理する（すなわち、プラットフォームマネージャノードがそれ自体を復元する）。休止されているプラットフォームマネージャサーバの復元要求を処理する場合は、休止されたサーバのメートが（可能ならば）、拒否（Ｄｅｎｉａｌ）応答をホストに送り返す。休止、ダウングレード、またはブートがプラットフォームマネージャノードに要求されると、プラットフォームマネージャノードのメートは、休止されていない限りそれを処理する。メートが休止されているときは、プラットフォームマネージャノードがそれ自体で、その休止、ダウングレード、またはブートを処理する。休止、ダウングレード、またはブートの処理は、要求された動作を実際に行うこと、またはホストに拒否応答を送り返すことを伴う場合がある。休止、ダウングレード、またはブートの要求が拒否されなければ、ホストは、その動作が成功したとみなす。
【００７６】
プラットフォームマネージャノードがそれ自体でブートを処理しなければならないとき、プラットフォームマネージャノードは、ＮｅｔＰＭのＧｒａｃｅＨａｌｔまたはＩｍｍｅｄＨａｌｔ動作を（ブートタイプに基づいて）呼び出して、それ自体を休止状態にする。次いで、それが休止状態まで下げられているため、このノードに対する処理は完了する。（ホストが、そのサーバのリセットおよびブートを開始することになる。）休止、ダウングレード、またはブートが、最後のＩｎ－Ｓｅｒｖｉｃｅプラットフォームマネージャノードに要求されると、フォースフラグが検査される。フォースフラグが設定されていなければ、要求は、「ＤＥＮＩＥＤ－ＬＡＳＴ　ＡＭＰ」の応答と共に拒否される。フォースフラグが設定されていれば、休止、ダウングレード、またはブートが、最後のＩｎ－Ｓｅｒｖｉｃｅプラットフォームマネージャノード上で行われる。
【００７７】
プラットフォームマネージャノードに対してＨｏｓｔ　Ｒｅｓｙｎｃが要求されると、ターゲットプラットフォームマネージャサーバのメートは、休止されていない限りその要求を処理する。ターゲットプラットフォームマネージャサーバのメートが休止されている場合は、再同期化のためのプラットフォームマネージャノードがその要求を処理する。この要求の処理は、サーバステータスを、ＸＯＳ－ＭＩＮからＡＯＳ－ＭＩＮに、またはＭＯＳ－ＭＩＮに変更すること、あるいは、現在のステータスがＸＯＳ－ＭＩＮでない場合に、その要求を拒否することを伴う。
【００７８】
ＳｖｃＣＦｇＳｖｒＦＳＭは、サービスアプリケーションサーバに対する復元、休止、再同期化、ダウングレード、ブートを扱う有限状態機械である。この機械は、動作を行うプ

ラットフォームマネージャノードの状態、作用を受けるサービスサーバの状態、および要求されたイベント（休止、ダウングレード、復元など）に基づいて要求を処理する。検査されるサービス状態は、Ｈａｌｔｅｄ（自動）、Ｈａｌｔｅｄ（手動）、ＸＯＳ－ＭＩＮ、ＡＯＳ－ＭＩＮ（自動）、ＭＯＳ－ＭＩＮ（手動）、ＩｎＳｖｃである。有効なイベントは、復元、グレースフル休止、即時休止、グレースフルダウングレード、即時ダウングレード、グレースフルブート、即時ブート、ホスト再同期化である。

【００７９】

アクティブなプラットフォームマネージャノード（ＯＳ－ＭＩＮまたはＩｎ－Ｓｅｒｖｉｃｅ）は、サービスサーバに対する構成サーバ要求を処理する。復元、休止、再同期化、またはダウングレードは、１つのプラットフォームマネージャが少なくともＯＳ－ＭＩＮにある限り、サービスサーバ上で許可される。サービスサーバに対する復元は、少なくとも１つのプラットフォームマネージャがＩｎ－Ｓｅｒｖｉｃｅのときだけ許可される。どのプラットフォームマネージャノードもＩｎ－Ｓｅｒｖｉｃｅでない場合、アクティブなプラットフォームマネージャノードは、ＤＥＮＹ－ＡＭＰ　ｎｏｔ　Ｉｎ－Ｓｅｒｖｉｃｅ応答をホストに送り返す。休止、ダウングレード、またはブートの要求が拒否されなければ、ホストは、その動作が成功したとみなす。

【００８０】

休止、ダウングレード、またはブートが、アプリケーションの最後のＩｎ－Ｓｅｒｖｉｃｅノードに要求されると、フォースフラグが検査される。フォースフラグが設定されていなければ、要求は、「ＤＥＮＩＥＤ－ＬＡＳＴ　ＳＥＲＶＥＲ　ＩＮ　Ｐｒｏｃｅｓｓｏｒ　ｓｅｒｖｉｃｅ　ｇｒｏｕｐ　ＰＲＯＣＥＳＳＩＮＧ　ＱＵＥＲＩＥＳ」の応答と共に拒否される。フォースフラグが設定されていれば、休止、ダウングレード、またはブートが、アプリケーションの最後のＩｎ－Ｓｅｒｖｉｃｅノード上で行われる。

【００８１】

構成イベント（即時休止を除く）が処理されるときは常に、構成中（Ｕｎｄｅｒ　Ｃｏｎｆｉｇｕｒａｔｉｏｎ）フラグがチェックされる。構成中フラグが設定されていれば、要求は、「ＤＥＮＩＥＤ－ＳＥＲＶＥＲ　ＵＮＤＥＲ　ＣＯＮＦＩＧＵＲＡＴＩＯＮ」の応答と共に拒否される。ＣｏｎｆｉｇＭｒｇは、イベントの処理中に、構成中フラグを設定およびクリアする。その他のメッセージ（すなわちＳｅｒｖｅｒｉｎｆｏＭｓｇ、ＳｅｒｖｅｒＳｔａｔｕｓＭｓｇ、ＴｉｍｅＭｓｇなど）は、有限状態機械を必要としない。

【００８２】

復元要求が拒否されないとき、ＣｏｎｆｉｇＭｇｒは、サーバに対してＵｎｄｅｒＣｏｎｆｉｇフラグを設定し、ＣｏｎｆｉｇＳｅｒｖｅｒＭｓｇ「Ａｃｔｉｏｎ　ｉｎｉｔｉａｔｅｄ（動作開始）」ＲＥＳＰＯＮＳＥをホストに送り、サーバをＩｎ－Ｓｅｒｖｉｃｅに復元するために、ＮｅｔＰＭのＲｅｓｔｏｒｅＩＳＶ動作を呼び出す。次いでＣｏｎｆｉｇＭｇｒは、復元処理を延期し、サーバがＩｎ－Ｓｅｒｖｉｃｅになるように、サーバステータス保留待ち行列（Ｓｅｒｖｅｒ　Ｓｔａｔｕｓ　ＰｅｎｄｉｎｇＱｕｅｕｅ）エントリを設定する。復元処理は、サーバステータスがＩｎ－Ｓｅｒｖｉｃｅであるかまたはタイマが切れていることを、ＣｏｎｆｉｇＭｇｒが通知されるまで継続されない。ＣｏｎｆｉｇＭｇｒが、Ｉｎ－Ｓｅｒｖｉｃｅへのサーバステータス変化を通知されると、サーバ照会ステータスを検査することによって、復元処理が継続される。サーバの照会ステータスがＤＩＳＡＢＬＥＤ＿ＳＥＲＶＥＲ＿ＯＯＳであり、アクティブなサーバの数が、プロセッササービスグループのアクティブサーバ総数よりも少ない場合、ＣｏｎｆｉｇＭｇｒは、ＮｅｔＰＭのＥｎａｂｌｅＱｕｅｒｙ動作を呼び出して、サーバの照会ステータスをイネーブルにし、現在の照会ステータスをＰｅｎｄｉｎｇ（保留）に設定する。次いでＣｏｎｆｉｇＭｇｒは、サーバおよび照会ステータスの変化を通知するサーバステータスメッセージをホストに送る。サーバの照会ステータスをＥｎａｂｌｅｄにするために、照会ステータス保留待ち行列（ＱｕｅｒｙＳｔａｔｕｓ　ＰｅｎｄｉｎｇＱｕｅｕｅ）エントリが設定される。次いで処理は、照会ステータスがイネーブルにされるか、あるいはタイマが切れるまで延期される。照会ステータスがＥｎａｂｌｅｄに変化したことを、Ｃ

ｏｎｆｉｇＭｇｒが通知されると、復元処理は継続し、サーバステータスメッセージを送り、サーバに対する構成中フラグをクリアする。

【００８３】

サーバステータスがＩｎ－Ｓｅｒｖｉｃｅに変化する前にタイマが切れると、あるいは他のアプリケーションに対して要求したサーバ情報が受け取られないと、復元故障処理が開始される。故障処理は、グレースフルにサーバをＯＳ－ＭＩＮにダウングレードすること、ＩＰＲを発行すること、およびサーバに対する構成中フラグをクリアすることを含む。照会ステータスがＥｎａｂｌｅｄに変化する前にタイマが切れると、復元処理は継続し、照会ステータスをＤｉｓａｂｌｅｄに設定し、グレースフルにサーバをＯＳ－ＭＩＮにダウングレードし、サーバステータスメッセージを送り、ＩＰＲを発行し、サーバに対する構成中フラグをクリアする。

【００８４】

グレースフル休止要求が拒否されないとき、ＣｏｎｆｉｇＭｇｒは、サーバに対するＵｎｄｅｒＣｏｎｆｉｇフラグを設定し、ＣｏｎｆｉｇＳｅｒｖｅｒＭｓｇ「Ａｃｔｉｏｎ Ｉｎｉｔｉａｔｅｄ（動作開始）」ＲＥＳＰＯＮＳＥをホストに送り、サーバを休止させるために、ＮｅｔＰＭのＧｒａｃｅＨａｌｔ動作を呼び出す。ノードがまだ休止されていない場合、ＣｏｎｆｉｇＭｇｒは休止処理を延期し、サーバがＨａｌｔｅｄになるようにサーバステータス保留待ち行列エントリを設定する。次いでＣｏｎｆｉｇＭｇｒは、ホストによって休止が要求されたことを示すエントリをネットワーク構成報告に作成する。休止処理は、サーバステータスがＨａｌｔｅｄであるかまたはタイマが切れていることを、ＣｏｎｆｉｇＭｇｒが通知されるまで継続されない。休止状態へのサーバステータス変化をＣｏｎｆｉｇＭｇｒが通知されると、休止処理は継続し、サーバステータスメッセージを送り、サーバに対する構成中フラグをクリアする。サーバステータスがＨａｌｔｅｄに変化する前にタイマが切れると、休止故障処理が開始される。故障処理は、ＩＰＲを発行し、サーバに対する構成中フラグをクリアすることを伴う。

【００８５】

即時休止要求が拒否されないとき、ＣｏｎｆｉｇＭｇｒは、サーバに対するＵｎｄｅｒＣｏｎｆｉｇフラグを設定し、このサーバに対するすべての保留中のサーバステータス変化をステータス保留待ち行列から除去し、サーバを休止させるためにＮｅｔＰＭのＩｍｍｅｄＨａｌｔ動作を呼び出す。ノードがまだ休止されていなければ、ＣｏｎｆｉｇＭｇｒは、休止処理を延期し、サーバがＨａｌｔｅｄになるように、サーバステータス保留待ち行列エントリを設定する。次いでそれは、ホストによって休止が要求されたことを示すエントリを、ネットワーク構成報告に作成する。休止処理は、サーバステータスがＨａｌｔｅｄであるかまたはタイマが切れていることを、ＣｏｎｆｉｇＭｇｒが通知されるまで継続されない。休止状態へのサーバステータス変化をＣｏｎｆｉｇＭｇｒが通知されると（あるいは休止が発行されたときにノードがすでに休止されていると）、休止処理は継続し、サーバステータスメッセージを送り、ＣｏｎｆｉｇＳｅｒｖｅｒＭｇｒ「Ｓｕｃｃｅｓｓｆｕｌｌｙ Ｃｏｍｐｌｅｔｅｄ（首尾よく完了）」ＲＥＳＰＯＮＳＥをホストに送り、サーバに対する構成中フラグをクリアする。

【００８６】

サーバステータスがＨａｌｔｅｄに変化する前にタイマが切れると、休止故障処理が開始される。故障処理は、ＩＰＲを発行し、ＣｏｎｆｉｇＳｅｒｖｅｒＭｓｇ「Ａｃｔｉｏｎ Ｆａｉｌｅｄ（動作故障）」ＲＥＳＰＯＮＳＥをホストに送り、サーバに対する構成中フラグをクリアすることを含む。

【００８７】

グレースフルダウングレード要求が拒否されないとき、ＣｏｎｆｉｇＭｇｒは、サーバに対するＵｎｄｅｒＣｏｎｆｉｇフラグを設定し、ＣｏｎｆｉｇＳｅｒｖｅｒＭｓｇ「Ａｃｔｉｏｎ Ｉｎｉｔｉａｔｅｄ（動作開始）」ＲＥＳＰＯＮＳＥをホストに送り、サーバをダウングレードするためにＮｅｔＰＭのＧｒａｃｅＤｏｗｎ動作を呼び出す。ノードがまだ所望のダウングレード状態になければ、ＣｏｎｆｉｇＭｇｒは、ダウングレード処理

を延期し、サーバがＯＳ－ＭＩＮになるように、サーバステータス保留待ち行列エントリを設定する。次いでそれは、ホストによってダウングレードが要求されたことを示すエントリを、ネットワーク構成報告に作成する。ダウングレード処理は、サーバステータスがＯＳ－ＭＩＮであるかまたはタイマが切れていることを、ＣｏｎｆｉｇＭｇｒが通知されるまで継続されない。ＯＳ－ＭＩＮ状態へのサーバステータス変化をＣｏｎｆｉｇＭｇｒが通知されると（あるいはノードがすでにその状態であれば）、ダウングレード処理は継続し、サーバステータスメッセージを送り、サーバに対する構成中フラグをクリアする。サーバステータスがＯＳ－ＭＩＮ状態に変化する前にタイマが切れると、ダウングレード故障処理が開始される。故障処理は、ＩＰＲを発行し、サーバに対する構成中フラグをクリアすることを伴う。

【００８８】

即時ダウングレード要求が拒否されないとき、ＣｏｎｆｉｇＭｇｒは、サーバに対するＵｎｄｅｒＣｏｎｆｉｇフラグを設定し、サーバをダウングレードするためにＮｅｔＰＭのＩｍｍｅｄＤｏｗｎ動作を呼び出す。ノードがまだ所望のダウングレード状態になければ、ＣｏｎｆｉｇＭｇｒは、ダウングレード処理を延期し、サーバがＯＳ－ＭＩＮになるように、サーバステータス保留待ち行列エントリを設定する。次いでそれは、ホストによってダウングレードが要求されたことを示す、エントリをネットワーク構成報告に作成する。ダウングレード処理は、サーバステータスがＯＳ－ＭＩＮであるかまたはタイマが切れていることを、ＣｏｎｆｉｇＭｇｒが通知されるまで継続されない。ＯＳ－ＭＩＮ状態へサーバステータス変化をＣｏｎｆｉｇＭｇｒが通知されると（あるいはノードがすでにその状態であれば）、ダウングレード処理は継続し、サーバステータスメッセージを送り、ＣｏｎｆｉｇＳｅｒｖｅｒＭｓｇ「Ｓｕｃｃｅｓｓｆｕｌｌｙ Ｃｏｍｐｌｅｔｅｄ（首尾よく完了）」ＲＥＳＰＯＮＳＥをホストに送り、サーバに対する構成中フラグをクリアする。

【００８９】

ステータスがＯＳ－ＭＩＮ状態に変化する前にタイマが切れると、ダウングレード故障処理が開始される。故障処理は、ＩＰＲを発行し、ＣｏｎｆｉｇＳｅｒｖｅｒＭｓｇ「Ａｃｔｉｏｎ Ｆａｉｌｅｄ（動作故障）」ＲＥＳＰＯＮＳＥをホストに送り、サーバに対する構成中フラグをクリアすることを伴う。

【００９０】

グレースフルまたは即時ブート要求が拒否されないとき、ＣｏｎｆｉｇＭｇｒは、サーバに対するＵｎｄｅｒＣｏｎｆｉｇフラグを設定し、ＣｏｎｆｉｇＳｅｒｖｅｒＭｓｇ「Ａｃｔｉｏｎ Ｉｎｉｔｉａｔｅｄ（動作開始）」ＲＥＳＰＯＮＳＥをホストに送る。ＣｏｎｆｉｇＭｇｒは、サーバに対するサーバステータスを検査し、サーバが休止状態でなければ、ＮｅｔＰＭのＧｒａｃｅＨａｌｔまたはＩｍｍｅｄＨａｌｔ動作を呼び出す。休止動作が呼び出されると、処理は、サーバステータスが休止されているかまたはタイマが切れていることを、ＣｏｎｆｉｇＭｇｒが通知されるまで延期される。次いでＣｏｎｆｉｇＭｇｒは、ホストによってブートが要求されたことを示すエントリをネットワーク構成報告に作成する。

【００９１】

ＯＳ－ＭＩＮ状態へのサーバステータス変化をＣｏｎｆｉｇＭｇｒが通知されると（あるいはノードがすでにその状態であれば）、ダウングレード処理は継続し、サーバステータスメッセージを送り、ＣｏｎｆｉｇＳｅｒｖｅｒＭｓｇ「Ｓｕｃｃｅｓｓｆｕｌｌｙ Ｃｏｍｐｌｅｔｅｄ（首尾よく完了）」ＲＥＳＰＯＮＳＥをホストに送り、サーバに対する構成中フラグをクリアする。サーバステータスがＯＳ－ＭＩＮ状態に変化する前にタイマが切れると、ダウングレード故障処理が開始される。故障処理は、ＩＰＲを発行し、ＣｏｎｆｉｇＳｅｒｖｅｒＭｓｇ「Ａｃｔｉｏｎ Ｆａｉｌｅｄ（動作故障）」ＲＥＳＰＯＮＳＥをホストに送り、サーバに対する構成中フラグをクリアすることを伴う。

【００９２】

グレースフルまたは即時ブート要求が拒否されないとき、ＣｏｎｆｉｇＭｇｒは、サーバ

に対するＵｎｄｅｒＣｏｎｆｉｇフラグを設定し、ＣｏｎｆｉｇＳｅｒｖｅｒＭｓｇ「Ａｃｔｉｏｎ　Ｉｎｉｔｉａｔｅｄ（動作開始）」ＲＥＳＰＯＮＳＥをホストに送る。ＣｏｎｆｉｇＭｇｒは、サーバに対するサーバステータスを検査し、サーバが休止状態でなければ、ＮｅｔＰＭのＧｒａｃｅＨａｌｔまたはＩｍｍｅｄＨａｌｔ動作を呼び出す。休止動作が呼び出されると、処理は、サーバステータスが休止されているかまたはタイマが切れていることを、ＣｏｎｆｉｇＭｇｒが通知されるまで延期される。次いでＣｏｎｆｉｇＭｇｒは、ホストによってブートが要求されたことを示すエントリを、ネットワーク構成報告に作成する。

【００９３】

サーバが休止されているとＣｏｎｆｉｇＭｇｒが判定するとき、ＣｏｎｆｉｇＭｇｒは、ＲｅｓｅｔＳｅｒｖｅｒＭｓｇ要求をホストに送る。ＣｏｎｆｉｇＭｇｒは、ホスト応答待ち行列エントリを生成して、ホストからのＲｅｓｅｔＳｅｒｖｅｒＭｓｇ応答を待つ。次いで処理は、ＲＥＳＰＯＮＳＥが受け取られるまで、あるいはタイマが切れるまで延期される。ＲＥＳＰＯＮＳＥが受け取られると、ＣｏｎｆｉｇＭｇｒは、サーバステータス保留待ち行列エントリを設定して、サーバステータスがＯＳ－ＭＩＮになるのを待つ。タイマが切れる前にホストからの応答が受け取られなければ、ＩＰＲが発行され、構成中フラグがクリアされる。サーバステータスがＯＳ－ＭＩＮになると、ＣｏｎｆｉｇＭｇｒは、新しいサーバステータスを示すサーバステータスメッセージをホストに送り、構成中フラグをクリアする。サーバステータスがＯＳ－ＭＩＮになる前にタイマが切れると、ＣｏｎｆｉｇＭｇｒは、ＩＰＲを発行し、構成中フラグをクリアする。

【００９４】

ホスト再同期化要求が拒否されないとき、ＣｏｎｆｉｇＭｇｒは、サーバステータスがＸＯＳ＿ＭＩＮであるかどうかを判定する。そうである場合、ＮｅｔＰＭのＳｅｔＳｅｒｖｅｒＳｔａｔｕｓ動作が呼び出されて、サーバステータスが適切な自動／手動ＯＳ＿ＭＩＮ状態に設定され、新しいサーバステータスを示すサーバステータスメッセージが送られ、ＣｏｎｆｉｇＳｅｒｖｅｒＭｓｇ「Ｓｕｃｃｅｓｓｆｕｌ（成功）」ＲＥＳＰＯＮＳＥがホストに送られる。サーバステータスがＸＯＳ＿ＭＩＮでない場合、ＩＰＲが発行され、ＣｏｎｆｉｇＳｅｒｖｅｒＭｓｇ「Ａｃｔｉｏｎ　Ｆａｉｌｅｄ（動作故障）」ＲＥＳＰＯＮＳＥがホストに送られる。

【００９５】

アプリケーションステータスメッセージは、Ｉｎ－Ｓｅｒｖｉｃｅアクティブであるプラットフォームマネージャノードによって処理される。どのプラットフォームマネージャノードもＩｎ－Ｓｅｒｖｉｃｅでない場合、ＯＳ－ＭＩＮアクティブであるプラットフォームマネージャノードが、その要求を処理することになる。ＡｐｐｌＳｔａｔｕｓＭｓｇ　ＲＥＱＵＥＳＴタイプメッセージをホストから受け取ると、ＣｏｎｆｉｇＭｇｒは、アプリケーション照会ステータスを決定し、現在のアプリケーション照会ステータスと共に、ＡｐｐｌＳｔａｔｕｓＭｓｇ　Ｓ＿ＲＥＰＯＲＴをホストに送り返す。ＣｏｎｆｉｇＭｇｒは、サーバステータス変化が起こったとき、またはホスト構成サーバ要求の処理中に必要とされるとき、ＡｐｐｌＳｔａｔｕｓＭｓｇ　Ｕ＿ＲＥＰＯＲＴタイプメッセージをホストに送る。

【００９６】

ＣｏｎｆｉｇＭｇｒは、ＡｐｐｌｓＩｎｆｏ．ｄｅｓ記述子ファイル中の各アプリケーションに対する、ＡＳＰＥＣ　Ｄａｔａ　ＲＥＱＵＥＳＴメッセージをホストから受け取る。ＣｏｎｆｉｇＭｇｒは、ＮｅｔＰＭに照会して、そのアプリケーションに対する情報をＮｅｔＭＡＰから検索する。成功または故障を示す応答コードと共に、ＡＳＰＥＣ　Ｄａｔａを含む応答メッセージが、ホストに送り返される。無効なアプリケーションＩｄ、ＡＳＰＥＣ　Ｄａｔａ　ＲＥＱＵＥＳＴメッセージ以外のメッセージ、または要求以外のメッセージタイプがある場合、ＩＰＲが発行される。

【００９７】

プロセッササービスグループ情報メッセージは、Ｉｎ－Ｓｅｒｖｉｃｅアクティブである

プラットフォームマネージャノードによって処理される。どのプラットフォームマネージャノードもＩｎ－Ｓｅｒｖｉｃｅでない場合、ＯＳ－ＭＩＮアクティブであるプラットフォームマネージャノードが、その要求を処理することになる。

【００９８】

ＰＳＧＩｎｆｏＭｓｇ　ＲＥＱＵＥＳＴタイプメッセージをホストから受け取ると、ＣｏｎｆｉｇＭｇｒは、プロセッササービスグループ情報を決定し、プロセッササービスグループ情報と共に、ＰＳＧＩｎｆｏＭｓｇ　Ｓ＿ＲＥＰＯＲＴをホストに送り返す。

【００９９】

プロセッササービスグループステータスメッセージは、Ｉｎ－Ｓｅｒｖｉｃｅアクティブであるプラットフォームマネージャノードによって処理される。どのプラットフォームマネージャノードもＩｎ－Ｓｅｒｖｉｃｅでない場合、ＯＳ－ＭＩＮアクティブであるプラットフォームマネージャノードが、その要求を処理することになる。ＰＳＧＳｔａｔｕｓＭｓｇＲＥＱＵＥＳＴタイプメッセージをホストから受け取ると、ＣｏｎｆｉｇＭｇｒは、プロセッササービスグループ照会ステータスを決定し、現在のプロセッササービスグループ照会ステータスと共にＰＳＧＳｔａｔｕｓＭｓｇ　Ｓ＿ＲＥＰＯＲＴをホストに送り返す。ＣｏｎｆｉｇＭｇｒは、サーバステータス変化が起こったとき、またはホスト構成サーバ要求の処理中に要求があれば、ＰＳＧＳｔａｔｕｓＭｓｇ　Ｕ＿ＲＥＰＯＲＴタイプメッセージをホストに送る。

【０１００】

照会処理メッセージは、Ｉｎ－Ｓｅｒｖｉｃｅアクティブであるプラットフォームマネージャノードによって処理される。どのプラットフォームマネージャノードもＩｎ－Ｓｅｒｖｉｃｅでない場合、ＯＳ－ＭＩＮアクティブであるプラットフォームマネージャノードが、その要求を処理することになる。ＣｏｎｆｉｇＭｇｒは、ＱｕｅｒｙＰｒｏｃＭｓｇ　ＤＩＳＡＢＬＥ＿ＳＥＲＶＥＲ、ＤＩＳＡＢＬＥ＿ＳＥＲＶＥＲ＿ＦＯＲＣＥＤ、およびＥＮＡＢＬＥ＿ＳＥＲＶＥＲ要求タイプをホストから受け取る。このメッセージを処理する際、ＣｏｎｆｉｇＭｇｒは、ＥｎａｂｌｅＳｅｒｖｅｒ／ＤｉｓａｂｌｅＳｅｒｖｅｒ動作をＮｅｔＰＭから呼び出すことによって、ターゲットサーバに対する照会処理をイネーブル／ディスエーブルにすることを開始する。ＣｏｎｆｉｇＭｇｒは、サーバに対して照会ステータス保留待ち行列エントリを設定し、サーバに対する照会ステータスが所望の状態に変化するか、またはタイマが切れるまで、それ以上の処理を延期する。ＮｅｔＰＭは、ＣｏｎｆｉｇＭｇｒのＮｔｆｙＱｒｙＳｔａｔＣｈａｎｇｅ動作を呼び出すことによって、照会ステータスの変化をＣｏｎｆｉｇＭｇｒに通知する。ＣｏｎｆｉｇＭｇｒがこの動作を処理するとき、ＣｏｎｆｉｇＭｇｒは、サーバ照会ステータスの状態に対する照会ステータス保留待ち行列エントリを検査する。所望の照会ステータスを伴うエントリがあれば、適切な成功照会処理手続きが呼び出されて、ＱｕｅｒｙＰｒｏｃＭｓｇの処理が再開される。ＱｕｅｒｙＰｒｏｃＭｓｇに対する成功処理は、要求が成功であったことを示すＱｕｅｒｙＰｒｏｃＭｓｇ　ＲＥＳＰＯＮＳＥをホストに送り返すこと、および必要であればプラットフォームマネージャノードに対するアクティブステータスを変更することを伴う。

【０１０１】

サーバ照会ステータスが所望の状態になる前にタイマが切れると、適切な故障照会処理手続きが呼び出されて、ＱｕｅｒｙＰｒｏｃＭｓｇの処理が再開される。ＱｕｅｒｙＰｒｏｃＭｓｇの故障処理は、ＩＰＲの発行、および要求が故障したことを示すＱｕｅｒｙＰｒｏｃＭｓｇ　ＲＥＰＯＮＳＥをホストに送り返すことを伴う。

【０１０２】

ＣｏｎｆｉｇＭｇｒは、サーバのブート処理中に、ＲｅｓｅｔＳｅｒｖｅｒＭｓｇ　ＲＥＱＵＥＳＴタイプメッセージを送る。ホストがＰＭでないサーバに対するブートを要求するとき、ターゲットサーバが休止された後に、ＲｅｓｅｔＳｅｒｖｅｒＭｓｇ　ＲＥＱＵＥＳＴが送られる。次いで、ＣｏｎｆｉｇＭｇｒはブート処理を延期し、ＲｅｓｅｔＳｅｒｖｅｒＭｓｇ　ＲＥＳＰＯＮＳＥタイプメッセージに対してホスト応答保留待ち行列エ

ントリを設定する。ブート処理は、ＲＥＳＰＯＮＳＥが受け取られるまで、またはタイマが切れるまで、継続されない。ＣｏｎｆｉｇＭｇｒが、ＲｅｓｅｔＳｅｒｖｅｒＭｓｇ ＲＥＳＰＯＮＳＥタイプメッセージをホストから受け取ると、ＣｏｎｆｉｇＭｇｒは、ホスト応答保留待ち行列エントリ中のＲｅｓｔＳｅｒｖｅｒＭｓｇ ＲＥＳＰＯＮＳＥに対する、ホスト応答保留待ち行列中のＲｅｓｅｔＳｅｒｖｅｒＭｓｇＲＥＳＰＯＮＳＥに対するエントリがあるかどうかを検査する。ある場合は、適切な手続きが呼び出されて、ブート処理が完了する。

【０１０３】

ルーティング情報メッセージは、Ｉｎ－Ｓｅｒｖｉｃｅアクティブであるプラットフォームマネージャノードによって処理される。どのプラットフォームマネージャノードもＩｎ－Ｓｅｒｖｉｃｅでない場合、そのメッセージは破棄される。ＣｏｎｆｉｇＭｇｒは、ＲｏｕｔｉｎｇＩｎｆｏＭｓｇ ＲＥＱＵＥＳＴタイプメッセージをホストから受け取ると、要求が承認されたことを示すＲｏｕｔｉｎｇＩｎｆｏＭｓｇ ＲＥＳＰＯＮＳＥをホストに送り返し、ルーティング情報を検索することを試みる。ルーティング情報が検索されると、ＣｏｎｆｉｇＭｇｒは、ルーティング情報と共に、ＲｏｕｔｉｎｇＩｎｆｏＭｓｇ Ｓ＿ＲＥＰＯＲＴをホストに送り返す。ＣｏｎｆｉｇＭｇｒは、別のサブシステムからルーティング情報を送るよう要求があると、ＲｏｕｔｉｎｇＩｎｆｏＭｓｇ Ｕ＿ＲＥＰＯＲＴタイプメッセージをホストに送る。ルーティング情報の送信要求を別のサブシステムから受け取ると、ＣｏｎｆｉｇＭｇｒは、ルーティング保留待ち行列を検査して、ホストが情報を要求したかどうかを判定する。そうである場合は、ＣｏｎｆｉｇＭｇｒは、ルーティング情報と共に、ＲｏｕｔｉｎｇＩｎｆｏＭｇｒ Ｓ＿ＲＥＰＯＲＴをホストに送る。そうでない場合は、ＣｏｎｆｉｇＭｇｒは、ルーティング情報と共に、ＲｏｕｔｉｎｇＩｎｆｏＭｓｇ Ｕ＿ＲＥＰＯＲＴをホストに送る。ＣｏｎｆｉｇＭｇｒは、Ｕ＿ＲＥＰＯＲＴをホストに送った後、ＲｏｕｔｉｎｇＩｎｆｏＭｓｇ ＡＣＫ ＲＥＳＰＯＮＳＥを送ることによって、データの受信を承認するためにホストを待機する。制限時間内にＣｏｎｆｉｇＭｇｒが応答を得ない場合、ＣｏｎｆｉｇＭｇｒは、アプリケーションルーティング情報を再度送るように、適切なサブシステムに要求する（ホストにデータを再送するようにするため）。ＮＡＫ ＲＥＳＰＯＮＳＥがホストから受け取られる場合、ＣｏｎｆｉｇＭｇｒは、ホストからの故障応答コードを示すＩＰＲを発行する。

【０１０４】

計画的動作制御メッセージは、Ｉｎ－Ｓｅｒｖｉｃｅアクティブであるプラットフォームマネージャノードによって処理される。どのプラットフォームマネージャノードもＩｎ－Ｓｅｒｖｉｃｅでない場合、ＯＳ－ＭＩＮアクティブであるプラットフォームマネージャノードが、その要求を処理することになる。ＣｏｎｆｉｇＭｇｒは、ＳｃｈｅｄＡｃｔＣｔｌＭｓｇ ＳＥＴタイプのメッセージをホストから受け取ると、ＮｅｔＰＭのＳｅｔＳｃｈｅｄＡｃｔｉｏｎ動作を呼び出して、計画的動作（定期監視や包括的な監査など）を希望に応じてイネーブル／ディスエーブルにする。ＣｏｎｆｉｇＭｇｒは、設定が成功であったか否かを示す、ＳｃｈｅｄＡｃｔＣｔｌＭｓｇ ＲＥＳＰＯＮＳＥタイプをホストに送り返す。ＣｏｎｆｉｇＭｇｒは、クライアントがホスト時間情報を得るのに使用することのできる、ＧｅｔＳｃｈｅｄＡｃｔｉｏｎｓ動作を有する。この動作が呼び出されると、ＣｏｎｆｉｇＭｇｒは、ＳｃｈｅｄＡｃｔＣｔｌＭｓｇ ＲＥＱＵＥＳＴタイプメッセージをホストに送る。次いでＣｏｎｆｉｇＭｇｒは、ホストからの所望のＳｃｈｅｄＡｃｔＣｔｌＭｓｇ Ｓ＿ＲＥＰＯＲＴに対する、ホスト応答保留待ち行列エントリを設定する。次いで、（ＧｔＳｃｈｅｄＡｃｔｉｏｎｓの）処理は、Ｓ＿ＲＥＰＯＲＴが受け取られるまで、またはタイマが切れるまで延期される。計画的動作を受け取る前にタイマが切れた場合は、何の動作も行われない。ＣｏｎｆｉｇＭｇｒは、ＳｃｈｅｄＡｃｔＣｔｌＭｓｇ Ｓ＿ＲＥＰＯＲＴタイプメッセージをホストから受け取ると、ＳｃｈｅｄＡｃｔＣｔｌＭｓｇ Ｓ＿ＲＥＰＯＲＴに対するエントリが、ホスト応答保留待ち行列中にあるかどうかを検査する。ある場合は、ＣｏｎｆｉｇＭｇｒは、ＮｅｔＰＭのＳｅｔＳｃｈｅｄＡｃｔｉｏｎ動作を呼び出して、計画的動作を希望に応じてイネーブル／ディスエーブ

ルにする。

【０１０５】

サーバ情報メッセージは、Ｉｎ－Ｓｅｒｖｉｃｅアクティブであるプラットフォームマネージャノードによって処理される。どのプラットフォームマネージャノードもＩｎ－Ｓｅｒｖｉｃｅでない場合、ＯＳ－ＭＩＮアクティブであるプラットフォームマネージャノードが、その要求を処理することになる。ＣｏｎｆｉｇＭｇｒは、プラットフォームマネージャサーバの初期設定処理および復元処理中に、ＳｅｒｖｅＩｎｆｏＭｓｇ　ＲＥＱＵＥＳＴおよびＲＥＱＵＥＳＴ　ＡＬＬタイプメッセージをホストに送る。メッセージの送出後、ＣｏｎｆｉｇＭｇｒは、タスクの処理を延期し、ＳｅｒｖｅｒＩｎｆｏＭｓｇ　Ｓ＿ＲＥＰＯＲＴタイプ（および／または、ＲＥＱＵＥＳＴ　ＡＬＬが使用された場合は、ＣＯＭＰＬＥＴＥタイプ）に対する、ホスト応答保留待ち行列を設定する。初期設定処理および復元処理は、必要とされるサーバ情報が得られるまで、またはタイマが切れるまで継続されない。初期設定中、（情報が得られる前に）タイマが切れると、ＣｏｎｆｉｇＭｇｒは、情報が得られるまで、ＳｅｒｖｅｒＩｎｆｏＭｓｇ　ＲＥＱＵＥＳＴまたはＲＥＱＵＥＳＴ　ＡＬＬを再送する。プラットフォームマネージャサーバの復元中、（情報が得られる前に）タイマが切れると、ＣｏｎｆｉｇＭｇｒは、復元が故障したというＩＰＲを発行する。

【０１０６】

ＳｅｒｖｅｒＩｎｆｏＭｓｇ　Ｓ＿ＲＥＰＯＲＴおよびＣＯＭＰＬＥＴＥメッセージをホストから受け取ると、ＣｏｎｆｉｇＭｇｒは、ＳｅｒｖｅｒＩｎｆｏＭｓｇ　Ｓ＿ＲＥＰＯＲＴまたはＣＯＭＰＬＥＴＥに対するエントリが、ホスト応答保留待ち行列中にあるかどうかを検査する。ある場合は、適切な手続きが呼び出されて、初期設定処理または復元処理が完了する。ＳｅｒｖｅｒＩｎｆｏＭｓｇ　ＣＨＡＮＧＥタイプメッセージをホストから受け取ると、ＣｏｎｆｉｇＭｇｒは、それがサーバ情報ＣＨＡＮＧＥを処理するのに適切な状態であるかどうかを判定する。そうである場合は、ＣｏｎｆｉｇＭｇｒは、変更されたサーバ情報をＮｅｔＰＭに通知し、サーバ情報が首尾よく変更されたか否かを示すＳｅｒｖｅｒＩｎｆｏＭｓｇ　ＲＥＳＰＯＮＳＥタイプをホストに送り返す。

【０１０７】

サーバステータスメッセージは、Ｉｎ－Ｓｅｒｖｉｃｅアクティブであるプラットフォームマネージャノードによって処理される。どのプラットフォームマネージャノードもＩｎ－Ｓｅｒｖｉｃｅでない場合は、ＯＳ－ＭＩＮアクティブであるプラットフォームマネージャノードが、その要求を処理することになる。ＳｅｒｖｅｒＳｔａｔｕｓＭｓｇ　ＲＥＱＵＥＳＴタイプメッセージをホストから受け取ると、ＣｏｎｆｉｇＭｇｒは、サーバおよび照会ステータスの情報を得て、現在のステータス情報と共に、ＳｅｒｖｅｒＳｔａｔｕｓＭｓｇ　Ｓ＿ＲＥＰＯＲＴをホストに送り返す。サーバステータス変化が起こったとき、またはホスト構成サーバ要求の処理中に必要があれば、ＣｏｎｆｉｇＭｇｒは、ＳｅｒｖｅｒＳｔａｔｕｓＭｓｇ　Ｕ＿ＲＥＰＯＲＴタイプメッセージをホストに送る。

【０１０８】

テストサーバメッセージは、Ｉｎ－Ｓｅｒｖｉｃｅアクティブであるプラットフォームマネージャノードによって処理される。どのプラットフォームマネージャノードもＩｎ－Ｓｅｒｖｉｃｅでない場合は、ＯＳ－ＭＩＮアクティブであるプラットフォームマネージャノードが、その要求を処理することになる。ターゲットサーバがそれ自身であり、それ自身のメートのプラットフォームマネージャが休止されていない場合、このプラットフォームマネージャノードは要求を破棄し、他のプラットフォームマネージャがメッセージを処理することになる。ＣｏｎｆｉｇＭｇｒは、ＴｅｓｔＳｅｒｖｅｒＭｓｇ　ＲＥＱＵＥＳＴまたはＡＢＯＲＴタイプメッセージをＭＡＩＮＴ論理リンク上のホストから受け取ると、ターゲットサーバのステータスがＭＯＳ＿ＭＩＮであるかどうかを判定する。そうである場合、ＣｏｎｆｉｇＭｇｒは、ＴｅｓｔＳｅｒｖｅｒＭｓｇ　Ａｃｋｎｏｗｌｅｄｇｅ　ＲＥＳＰＯＮＳＥをホストに送り返す。以降に、ＣｏｎｆｉｇＭｇｒは、ＲＥＱＵＥＳＴが受け取られたかＡＢＯＲＴが受け取られたかに基づいて、適切なテストを開始または

中止することになる。ターゲットサーバがＭＯＳ－ＭＩＮでない場合、ＣｏｎｆｉｇＭｇｒは、ＴｅｓｔＳｅｒｖｅｒＭｓｇ Ｓｅｒｖｅｒ Ｎｏｔ ＭＯＳ－ＭＩＮ ＲＥＳＰＯＮＳＥをホストに送り返す。ターゲットサーバステータスが得られない場合、ＣｏｎｆｉｇＭｇｒは、ＴｅｓｔＳｅｒｖｅｒＭｓｇ Ｄｅｎｉｅｄ ＲＥＳＰＯＮＳＥをホストに送り返し、適切なＩＰＲを発行する。

【０１０９】

時間メッセージは、Ｉｎ－Ｓｅｒｖｉｃｅアクティブであるプラットフォームマネージャノードによって処理される。どのプラットフォームマネージャノードもＩｎ－Ｓｅｒｖｉｃｅでない場合は、ＯＳ－ＭＩＮアクティブであるプラットフォームマネージャノードが、その要求を処理することになる。ＴｉｍｅＭｓｇ ＳＥＴタイプメッセージをホストから受け取ると、ＣｏｎｆｉｇＭｇｒは、ＮｅｔＰＭのＳｅｔＴｉｍｅ動作を呼び出して、サーバネットワーク時間を適切な時間に設定し、設定が成功であったか否かを示すＴｉｍｅＭｓｇ ＲＥＳＰＯＮＳＥをホストに送り返す。ＣｏｎｆｉｇＭｇｒは、クライアントがホストの時間情報を得るために使用することのできるＧｅｔＴｉｍｅ動作を有する。この動作が呼び出されると、ＣｏｎｆｉｇＭｇｒは、ＴｉｍｅＭｓｇ ＲＥＱＵＥＳＴタイプメッセージをホストに送る。次いでＣｏｎｆｉｇＭｇｒは、ホストからの所望のＴｉｍｅＭｓｇ Ｓ＿ＲＥＰＯＲＴに対するホスト応答保留待ち行列エントリを設定する。次いで処理は、Ｓ＿ＲＥＰＯＲＴが受け取られるまで、またはタイマが切れるまで延期される。タイマ情報が受け取られる前にタイマが切れた場合は、何の動作も行われない。ＣｏｎｆｉｇＭｇｒは、ＴｉｍｅＭｓｇ Ｓ＿ＲＥＰＯＲＴタイプメッセージをホストから受け取ると、ホスト応答保留待ち行列中にＴｉｍｅＭｓｇ ＲＥＳＰＯＮＳＥに対するエントリがあるかどうかを検査する。ある場合は、ＮｅｔＰＭのＳｅｔＴｉｍｅ動作を呼び出して、サーバネットワーク時間を設定する。

【０１１０】

ノード管理サービス

ノードプラットフォームマネージャ（ＮｏｄｅｌＰＭＭＭａｉｎ）

ノード管理サブシステムは、単一のサーバノード内のプロセス管理を提供する。これは、サーバノード内のプロセスの開始／停止を担当して、特定の稼動レベルを維持する。ノード管理によってサポートされる稼動レベルは以下のものである。

【０１１１】

－ ＨＡＬＴＥＤ（どのソフトウェアも、ＯＳさえも稼動していない）

－ ＭＩＮ－ＳＥＴ（ＯＳ＋最低限必要なプラットフォームソフトウェア）

－ ＩＮＳＥＲＶＩＣｏｎｆｉｇｕｒａｂｌｅエレメント（ＭＩＮ－ＳＥＴ＋通常のソフトウェア）

ネットワーク管理は、特定のノードに対する所望の稼動レベルをノード管理に通知する。プロセス障害があった場合、ノード管理は障害を評価し、もし必要ならどの復旧動作が必要かを決定する。復旧動作は、障害を無視すること、ノードを次に低い稼動レベルに自動始動させて元の稼動レベルに戻すこと、およびシステムのシャットダウンを含む。

【０１１２】

ＮｏｄｅＰＭは、各サーバノードに対するシステム起動手続きの一部として働くことになる。その初期設定の一部として、ＮｏｄｅＰＭは、

・ＮｏｄｅＭＡＰオブジェクトのインスタンスを生成し、各サーバ上で構成される必要のある最低限の構成可能エレメントに関する構成情報を得た後で、サーバノードを最低限の動作状態（ＯＳ－ＭＩＮ）に引き上げる。この状態からは、サーバノードには、残りのプロセスを引き上げるなど最低限の機能のセットしか許可されない。各ノードのＮｏｄｅＭＡＰ中に提供される構成データは、各サーバノードの能力（プラットフォームマネージャ能力を伴うサーバノードに対する、照会処理能力を伴うサーバノード）を決定する。

【０１１３】

・ＮｅｔＰＭ要求を扱うＮｏｄｅＰＭサーバオブジェクトを生成して、同じサーバノード内で動作を行う。

【０１１４】
ＮｅｔＰＭ要求ごとに、ＮｏｄｅＰＭは、（そのサーバオブジェクトによって提供される動作を通して）以下の動作を行うことができる。
【０１１５】
・そのサーバノードを、最低限の動作状態（ＯＳ－ＭＩＮ）から完全な動作状態（ＩＮ－ＳＥＲＶＩＣｏｎｆｉｇｕｒａｂｌｅエレメント）に引き上げる（ＲｅｓｔｏｒｅＮｏｄｅ動作）。
【０１１６】
・そのサーバノードを完全な動作状態（ＩＮ－ＳＥＲＶＩＣｏｎｆｉｇｕｒａｂｌｅエレメント）から最低限の（ＯＳ－ＭＩＮ）または休止された（ＨＡＬＴ）動作状態に引き下げる（ＲｅｍｏｖｅＮｏｄｅ動作）。
【０１１７】
・そのサーバノード上の照会処理をイネーブル／ディスエーブルにする。
【０１１８】
・構成可能エレメントに関するステータス情報を提供する。
【０１１９】
ＮｏｄｅＰＭは、各ＩＰＵ上のどんなステータス変化も、ＮｅｔＰＭに自発的に報告する（ＮｏｄｅＰＭは、ＮｅｔＰＭから提供される動作を利用してステータス変化を報告する）。
【０１２０】
図８は、ノードに対するサービス状態の正当な移行を表す図である。全ての自動状態は他の自動状態に移行し、全ての手動状態は他の手動状態に移行することに留意されたい。手動状態から自動状態への正当な移行はない。このときＩＳＶ状態は、自動または手動の指示を有しない。状態は、ＩＮ－ＳＥＲＶＩＣＥ（ＩＳＶ）状態２００と他のいずれかの状態との間を移行することができる。図８で使用している頭字語は、以下のように復号される。
【０１２１】
【表１】

ISV 200	サービス中
OOSAM 202	自動最低限サービス停止中
OOSMM 204	手動最低限サービス停止中
OOSAN 206	自動サービス停止中－休止
OOSMN 208	手動サービス停止中－休止
ABOOT 210	自動ブート
MBOOT 212	手動ブート
ADOWN 214	自動ダウングレード
MDOWN 216	手動ダウングレード
AHALT 218	自動休止
MHALT 220	手動休止
AREST 222	自動復元
MREST 224	手動復元

ノードシステム保全性（ＮｏｄｅＳＩＭａｉｎ）
ノードシステム保全性サブシステム（クラス名ＮｏｄｅＳＩ）は、単一のサーバノード内における故障隔離および監視サービスを提供する。全てのプロセス障害は、このサブシステムによって記録され、復旧動作のためにノード管理に転送される。ノードシステム保全性は、以下の特徴を与える。
【０１２２】
－　受動的プロセス監視（信号の受止め）
－　ノード間通信の監視
－　局所故障報告
ＡＩＮプラットフォームのシステム保全性（ＳＩ）能力は、プラットフォームのサーバノードにわたって能力を提供するものと、単一のサーバノード内で能力を提供するものとに分類することができる。ＮｅｔＳＩは、プラットフォームレベルでシステム保全性能力を扱うが、ＮｏｄｅＳＩは、単一ノードレベルでシステム保全性を提供する。ＮｏｄｅＳＩは、プラットフォームのあらゆるサーバノードに常駐し、各構成可能エレメントに対するプロセスが、そのプロセスの故障ステータスを報告することができるための動作を提供する。これらの故障には、以下のものが含まれる。
【０１２３】
・各プロセス上で定期監視オブジェクトによって検出される故障
・ノード間通信障害
・ホストとサーバネットワークとの間の通信障害
・ＩＭサーバプロセスによって検出される故障
ＮｏｄｅＳＩはまた、ノード間の全ての接続に対するノード定期監視も行う。通信故障が検出される場合、ＮｏｄｅＳＩは、その通信故障をＮｅｔＳＩに通知する。報告される故障により、ＮｏｄｅＳＩは、ＩＰＲの発行およびノード状態のダウングレードを含めた適切な動作をとる（ＮｏｄｅＰＭと共同して）。
【０１２４】
ＮｏｄｅＳＩは、各サーバノード上のディスク利用を監視し、次いで、特定のファイルシステム上で使用される総容量があるしきい値を超えるとき、適切なＩＰＲを発行する。ＮｏｄｅＳＩによる他のオブジェクトとの通信は、ＤＯＭＥインタフェースを介して処理される。ＮｏｄｅＳＩは、構成内の全てのＩＰＵのリストをＮｏｄｅＰＭから得る。各ＩＰＵからの以下の情報を含む配列が設定される。
【０１２５】
・ＮｏｄｅＰＭから受け取ったＩＰＵ情報
・ＩＰＵステータス
・故障総数
・アライブメッセージ受領済みインジケータ
ステータスを他のＮｏｄｅＳＩと通信するために、ノード名よりも、このリストへの配列インデックスが使用される。というのは、文字列の比較は、速度および効率性の面からみてコストがかさむ可能性があるからである。したがって、構成内の各ノードが、同じオーダの同じＩＰＵリストを有することが重要である。
【０１２６】
ＮｏｄｅＳＩはＮｏｄｅＰＭに登録して、ノード状態の通知を受ける。ＮｏｄｅＳＩが他のＩＰＵに対するステータス変化の通知を受けると、ＮｏｄｅＳＩは、ＩＰＵ配列中のＩＰＵステータスを更新することになる。ステータス変化が休止状態への変化である場合、ＮｏｄｅＳＩは、故障総数およびアライブメッセージ受領済みインジケータをクリアすることになる。
【０１２７】
ＮｏｄｅＳＩは、その定期監視機能を扱うために２つのタイマを有する。
【０１２８】
・ＢｒｏａｄｃａｓｔＴｉｍｅｒ（ブロードキャストタイマ）－ＮｏｄｅＳＩがその視野

にある他のＮｏｄｅＳＩに「Ｉ’ｍ　ａｌｉｖｅ」メッセージをブロードキャストするようにするタイマ。

【０１２９】

・ＣｏｎＭｏｎＣｈｋＴｉｍｅｒ－適切な「Ｉ’ｍ　ａｌｉｖｅ」メッセージが全ての接続に対して時間間隔内に受け取られたかどうかを、ＮｏｄｅＳＩが判定するようにするタイマ。

【０１３０】

ノードがＯＳ－ＭＩＮであることをＮｏｄｅＳＩが通知されると、ＮｏｄｅＳＩは、その視野にある他のＮｏｄｅＳＩに「Ｉ’ｍ　ａｌｉｖｅ」メッセージをブロードキャストし始める。次いでＮｏｄｅＳＩは、ブロードキャストタイマをトリガする。ブロードキャストタイマが切れる際、ＮｏｄｅＳＩは、直ちに「Ｉ’ｍ　ａｌｉｖｅ」メッセージを再びブロードキャストし、ブロードキャストタイマを再トリガする。これは、継続している場合のあるどんなＮｏｄｅＳＩ処理にも割り込むことになる。

【０１３１】

ＮｏｄｅＳＩが「Ｉ’ｍ　ａｌｉｖｅ」メッセージを他のＮｏｄｅＳＩから受け取ると、ＮｏｄｅＳＩは、適切なＩＰＵ配列エントリのアライブメッセージ受領済みインジケータにマークする。

【０１３２】

ノードがＯＳ－ＭＩＮであることの通知をＮｏｄｅＳＩが受けると、ＮｏｄｅＳＩは、ＣｏｎＭｏｎＣｈｋＴｉｍｅｒをトリガする。ＣｏｎＭｏｎＣｈｋＴｉｍｅｒが切れる際、ＮｏｄｅＳＩは、ＣｏｍｍＦａｉｌＣｈｅｃｋ動作へのＤｏｍｅ呼出しをして、通信障害の検査を行い、タイマを再トリガする。ＮｏｄｅＳＩは、アライブメッセージをブロードキャストする優先度が与えられるのを保証するために、それ自体にＤＯＭＥ呼出しを使用している。

【０１３３】

通信障害の処理は、その配列内の各ＩＰＵを検査して、最後の検査以後にアライブメッセージが受け取られたかどうかを判定することを伴う。受け取られていれば、アライブメッセージ受領済みインジケータがクリアされる。メッセージが受け取られておらず、ＩＰＵステータスが休止されていなければ、そのノードに対する故障総数がインクリメントされることになる。そのＩＰＵに対する故障回数がその最大限に達していれば、ＮｏｄｅＳＩは、ＮｅｔＳＩに通信障害を報告する。

【０１３４】

故障総数の最大数は、キーワード「ＭＡＸ＿ＣＯＭＭ＿ＦＡＵＬＴＳ」の使用によって、コマンドラインから読み込むことのできる構成可能な値である。値が与えられない場合、故障総数のデフォルト数は２となる。また、コマンドライン中に与えられた値が２よりも小さい場合、最大数は２に設定される。

【０１３５】

アライブメッセージの各ブロードキャスト間の秒数は、キーワード「ＢＲＯＤＣＡＳＴ＿ＡＬＩＶＥ－ＳＥＣＳ」の使用によってコマンドラインから読み込むことのできる構成可能な値である。値が与えられない場合、ブロードキャスト間の秒のデフォルト数は１秒となる。コマンドライン中に与えられた値が１秒よりも小さい場合、秒数は１に設定される。

【０１３６】

各定期監視検査の間の秒数は、キーワード「ＣＯＮＭＯＮ＿ＣＨＫ＿ＳＥＣＳ」の使用によってコマンドラインから読み込むことのできる構成可能な値である。値が与えられない場合、検査間の秒のデフォルト数は２秒となる。コマンドライン中に与えられた値が２秒よりも小さい場合、秒数は２に設定される。

【０１３７】

ＮｏｄｅＳＩは、あらゆるノードの起動の一部として、他のプロセスの起動前に、ＮｏｄｅＰＭによって始動される。その初期設定の一部として、ＮｏｄｅＳＩは、ＮｏｄｅＳＩ

によって検出される故障の定義を含む記述子ファイル（Ｆａｕｌｔ．ｄｅｓ）を読み取り、これらの故障記録のリスト（ＦａｕｌｔＩｎｆｏＬｉｓｔ）を作成する。各故障記録（ＦａｕｌｔＩｎｆｏ）は、以下の部分を含む。

【０１３８】

・ＦａｕｌｔＩｄ－故障の識別

・ＦａｕｌｔＡｃｔＩｄ－報告される故障ごとに行われる動作

故障が受け取られるとき、ＮｏｄｅＳＩは、故障のＩＤを使用してそのリスト（ＦａｕｌｔＩｎｆｏＬｉｓｔ）中の故障記録を検索し、その故障に関連する動作を行うことになる。これらの動作は以下のことを含む場合がある。

【０１３９】

・適切なＩＰＲを発行する。

【０１４０】

・ＮｏｄｅＰＭプロセス上の破局的故障を検出した場合にノードを休止する。

【０１４１】

・構成可能エレメント上の自発的なステータス変化をＮｏｄｅＰＭに報告する。

【０１４２】

・通信障害をＮｏｄｅＰＭに、そしてまたＮｅｔＳＩに報告する。

【０１４３】

全ての故障（定期監視または他のプロセスから発するもの）は、ＮｏｄｅＳＩのＮｏｔｉｆｙＦａｕｌｔ動作を介して、各プロセスからＮｏｄｅＳＩに報告されることになる。ＮｏｄｅＳＩは、サーバノード上のディスク利用を常に追跡し、８０が使用された場合にＩＰＲを発行する。

【０１４４】

ＮｏｄｅＰＭインタフェース

ＮｏｄｅＳＩは、ＮｏｄｅＰＭから提供されるインタフェースを使用して、構成可能エレメントのステータスの自発的な変化（ＡｕｔｏＣｈｇＣＥＳｔａｔ（．．．））を報告する。ノードの状態に対する構成可能エレメントの影響により、ステータス変化は、ＮｏｄｅＰＭが以下の動作のいずれかを行うことを引き起こす場合がある。

【０１４５】

・ノードの状態をダウングレードする－この動作は、構成可能エレメントのステータス変化が、ノードの現在の動作状態に重大な影響を与えた場合に行われる。この動作を行う前に、ＮｏｄｅＰＭは、その意向をＮｅｔＳＩに通知し、タイマを始動させる。次いで、ＮｅｔＰＭからの要求時、またはタイムアウト時、ＮｏｄｅＰＭはノードの状態をダウングレードする。

【０１４６】

・通信障害を報告する－この動作は、構成可能エレメントのステータス変化がノード間通信障害を示した場合（ＴＣＰリンクがサービス停止となる）に行われる。この場合、ＮｏｄｅＰＭは通信障害をＮｅｔＳＩに通知し、通信を再び確立するよう試みる。

【０１４７】

ＮｅｔＳＩインタフェース

ＮｅｔＳＩは、以下の状態を報告するためにＮｏｄｅＳＩおよび／またはＮｏｄｅＰＭによって使用される動作を提供する。

【０１４８】

・ＩＰＵのステータスにおける自発的変化（ＤｏｗｎｇｒａｄｅＩＰＳｔａｔ（．．．））－この場合、ＮｅｔＳＩが、ＮｅｔＰＭを通してノードをダウングレードする（ノードがまだ休止されていなければＮｅｔＰＭにダウングレードを要求する）。

【０１４９】

・通信障害（ＣｏｍｍＦａｕｌｔＲｐｒｔ（．．．））－この場合、同じＩＰＵへの通信障害が他のＩＰＵから報告されると、ＮｅｔＳＩは、そのＩＰＵを故障中ＩＰＵとしてマークし、ＮｅｔＰＭを通してそれをダウングレードするよう試みる。

【０１５０】
定期監視インタフェース
各構成可能エレメントプロセスには、プロセス上で異なる信号を生成する異常な状態／イベントを検出して報告するために、定期監視オブジェクトのインスタンスを生成することが必要とされる。定期監視は、ＮｏｄｅＳＩのＮｏｔｉｆｙＦａｕｌｔ動作を介してこれらの状態を報告する。ＮｏｄｅＳＩに故障を通信するのに失敗した場合、定期監視は、そのインスタンス生成の時点で設定するオプションにより、ノードをＨＡＬＴにすることができる。
【０１５１】
メッセージハンドラ／論理リンクインタフェース
メッセージハンドラまたは論理リンク構成可能エレメントプロセスは、ＮｏｄｅＳＩの動作ＮｏｔｉｆｙＦａｕｌｔを利用して、ＤＮＩ／ＴＣＰリンク上の故障を報告する。
【０１５２】
サービスマネージャ（ＳＭＰｒｏｃｅｓｓ）
サービス管理サブシステムは、アプリケーションプロセスに対するプロセス制御を提供する。アプリケーションプロセスは、ノードがＩＮ　ＳＥＲＶＩＣＥ稼動レベルを達成した後でしか稼動されない。アプリケーションプロセスは、サーバノード上で個別に除去／復元、およびイネーブル／ディスエーブルすることができる。ネットワーク管理は、どのアプリケーションを除去、復元、イネーブル、ディスエーブルするかを、サービス管理に通知する。サービス管理によって与えられる特徴は、以下を含む。
【０１５３】
－　能動的プロセス監視（ハートビート、監査）
－　複数プロセスのインスタンスサポート
－　アプリケーションプロセスの状態管理
－　管理状態
－　動作状態
－　使用状態
－　アプリケーションプロセスの状態変化の通知
一貫した構成可能エレメントインタフェースを提示する通信プラットフォームナビゲータの特徴のために、ＮｏｄｅＰＭの代わりに、サービス管理がシステムの構成可能エレメントを始動させるように変更がなされている。こうすることにより、システム内の全てのプロセス（サービス管理を除く）は、サービス管理によって始動され、このとき各構成可能エレメントの特徴は、システム全体で同じである。通信プラットフォームナビゲータＧＵＩを生み出すために、通信プラットフォームシステムの一貫した計画が存在しなければならない。図９Ａは、ノード初期設定中に、通信プラットフォーム内のエンティティ間に存在する新たな関係を示す図である。構成可能エレメントが全てのサービス管理機能を利用できるようなるために、サービス管理インタフェースは、以下のことを必要とする。
【０１５４】
・全てのノード上で稼動する最初のものとして、ブートスクリプト２３０が生成される。ブートプログラム２３０が稼動するとき、このブートプログラムは、プラットフォームマネージャノード２３２を識別し、アクティブなプラットフォームマネージャノードのＴｃｌ記述子ファイル２３４を上書きコピーして、そのノードを立ち上げるのに使用する。ブートプログラムは、それが立ち上がる最初のプラットフォームマネージャノードであると決定する場合、既存のＴｃｌ記述子ファイル２３４を使用して稼動させる。
【０１５５】
・プラットフォームマネージャサブシステム、およびサービス管理サブシステム２３６は、前のバージョンのプラットフォームにおける構成可能エレメント２３８について異なる概念を有する。これら２つの概念は、これらの別々の機能が併合されて、１つの構成可能エレメントの概念にまとめられる。こうするために、プラットフォームマネージャサブシステムは、もはや構成可能エレメントの除去および復元をすることはなく、構成可能エレ

メントの除去および復元を希望するときに、サービス管理に通知することになる。このときサービス管理は、始動される最初の通信プラットフォームプログラムとなり、常にその初期設定の一部としてＮｏｄｅＰＭを始動させる。次いでＮｏｄｅＰＭは、前と同じように処理の開始および停止を制御するが、前のＲｅｍｏｖｅＣＥおよびＲｅｓｔｏｒｅＣＥ機能を通してでなく、サービス管理を通してのみ、これを行う。

【０１５６】

図９Ｂは、ＭＩＮ＿ＳＥＴ状態へのノード初期設定を示すメッセージ流れ図である。図９Ｃは、ＩＮ＿ＳＥＶＩＣＥ状態へのノード初期設定を示すメッセージ流れ図である。図９Ｄは、ＰＯＳＴ＿ＩＳＶ状態へのノード初期設定を示すメッセージ流れ図である。

【０１５７】

図１０は、ＳＭと構成可能エレメントとの間で使用されるメッセージプロトコルの概要を示す。サービス管理は、構成可能エレメントが、サービス管理インタフェース（ＳＭＩ）オブジェクトにリンクできない場合でも、構成可能エレメントを始動させることができるが、サービス管理が備える特徴の多くは利用できなくなる。

【０１５８】

イベントマネージャ（ｅｖｅｎｔｍａｎａｇｅｒｉｍｐｌ）

イベントマネージャサブシステムは、登録された１つまたは複数の相手に、包括的にイベント通知を発行することができる能力をユーザに提供する。システムには、複数のイベント（Ｍａｎａｇｅｒオブジェクトインスタンス）が存在してもよい。ノードレベルのイベント（Ｍａｎａｇｅｒ）は、全てのノードに存在する。また、１つのプロセスに専用のイベントに登録するための能力を、それに関心のある相手に提供するために、その他のイベント（Ｍａｎａｇｅｒインスタンス）が存在してもよい。ｅｖｅｎｔｍａｎａｇｅｒｉｍｐｌプログラムは、それが稼動し続けているモードに対して、イベント（Ｍａｎａｇｅｒオブジェクトインスタンス）を提供する。ノードに関連するイベントは、そのイベント（Ｍａｎａｇｅｒインスタンス）を通して発行される。特定のノード上のイベントに関心のあるユーザは、そのノード名をイベント（Ｍａｎａｇｅｒ名）に使用して、そのノードイベント（Ｍａｎａｇｅｒインスタンス）に結び付けることができる。プログラムもまた、その中にイベント（Ｍａｎａｇｅｒオブジェクト）を埋め込むことができる。これを行うプログラムの一例は、ＩｐｒＭｇｒＩｍｐｌプログラムである。ＩｐｒＭｇｒＩｍｐｌは、ＩｐｒＥｖｅｎｔＭｇｒと名付けられたイベント（Ｍａｎａｇｅｒ）を有する。ユーザはＩＰＲイベントの受取りを希望する。特定のイベントに関心のあるユーザは、特定のイベント（Ｍａｎａｇｅｒインスタンス）に登録して、そのイベント（Ｍａｎａｇｅｒインスタンス）を通してそのイベントを受け取ることもできる。イベント（Ｍａｎａｇｅｒ）は、登録された相手のリストを持続的に格納するわけではない。イベント（Ｍａｎａｇｅｒが）、過ぎ去ったイベント（Ｒｅｃｅｉｖｅｒ）にイベントを転送しようとしても、そのイベント（Ｒｅｃｅｉｖｅｒ）はリストから除去されている。

【０１５９】

図１１に、通信プラットフォームシステムにおけるイベント（Ｍａｎａｇｅｒ２５０）の２つの使用例を示す。ｅｖｅｎｔｍａｎａｇｅｒｉｍｐｌ２５２は、ノードイベント（Ｍａｎａｇｅｒオブジェクトインスタンス２５０）を含む。ＮｏｄｅＰＭＭａｉｎ通信プラットフォームプログラム２５４は、このイベント（Ｍａｎａｇｅｒ２５０）を使用して、ノードが状態を変更したときにイベントを発行する。次いで、アプリケーションプログラム２５６は、イベント（Ｒｅｃｅｉｖｅｒオブジェクト２６８）を生み出し、ＣＯＲＢＡオブジェクトリファレンスを「Ｎｏｄｅ１２３」イベント（Ｍａｎａｇｅｒ２５０）上の登録呼出しに渡す。ＮｏｄｅＰＭＭａｉｎ２５４が「Ｎｏｄｅ１２３」イベント（Ｍａｎａｇｅｒ２５０）上の通知を呼び出すことによってイベントを生成すると、このイベント（Ｍａｎａｇｅｒ２５０）は、このイベントを受け取るために登録した全てのイベント（Ｒｅｃｅｉｖｅｒオブジェクト２５８）を検出することになる。イベント（Ｍａｎａｇｅｒ２５０）は、アプリケーションプログラムがこのイベントに登録していることを知ると、そのイベント（Ｒｅｃｅｉｖｅｒオブジェクト２５８）上の通知方法を呼び出し、イベ

ント（Ｒｅｃｅｉｖｅｒオブジェクト２５８）は、通知方法がアプリケーションプログラム２５６中で呼び出されるようにすることになる。上記の例では、アプリケーションプログラム２５６はまた、ＩｐｒＭｇｒＩｍｐｌプログラム２６２中の「ＩｐｒＥｖｅｎｔＭｇｒ」イベント（Ｍａｎａｇｅｒ２６０）にも登録している。ＮｏｄｅＰＭＭＭａｉｎ２５４がＩｐｒＭｇｒＩｍｐｌインタフェースを使用してＩＰＲを発行すると、ＩｐｒＭｇｒＩｍｐｌプログラム２６２は、そのＩＰＲ上を検索し、検証し、「ＩｐｒＥｖｅｎｔＭｇｒ」イベント（Ｍａｎａｇｅｒ２６０）上の通知を呼び出す。これにより、そのイベント（Ｍａｎａｇｅｒ２５０）は、登録呼出し中で渡された、アプリケーションプログラム２５６中のイベント（Ｒｅｃｅｉｖｅｒ２６４）に、生成されたイベントを転送する。

【０１６０】

アプリケーションプログラム２５６は、ＩｐｒＭｇｒＩｍｐｌプログラムがしたのと同じように、それ自体の名称を有するそれ自体のイベント（Ｍａｎａｇｅｒ）を生成することができる。イベント（Ｍａｎａｇｅｒインスタンス）は、間違ったイベント（Ｍａｎａｇｅｒ）に対するイベントの生成を防止するため、またはユーザを間違ったイベント（Ｍａｎａｇｅｒ）への登録から隔離するのを補助するために、システムで固有の名称を有する必要がある。

【０１６１】

ＩＰＲ／ＡＬＡＲＭサービス

情報および問題報告（ＩＰＲ）サブシステムは、情報および問題報告を発行する能力を、システムにおける全てのプロセスに提供する。ＩＰＲは、システムのユーザに、誤り状況または他の関係するシステム情報について通知するのに使用される標準的な機構である。情報および問題報告サブシステムは、通信システムにおけるＩＰＲの収集を実施する。警報は、ＩＰＲに結合することのできる機構である。現在、警報サービスは利用できないが、将来公開される通信プラットフォームでは利用可能になるであろう。

【０１６２】

ＩＰＲサブシステムは、いくつかの特徴を備える。ＩＰＲサブシステムは、アクティブ／待機ＩＰＲサービス冗長性、登録された受信器にＩＰＲを転送する能力、ホストにＩＰＲを転送する能力、リアルタイムでＩＰＲを表示する能力、レガシーＰＡＣｏｎｆｉｇｕｒａｂｌｅエレメントＩＰＲインタフェースおよびＣＯＲＢＡ　ＩＰＲインタフェースとの後方互換性、ＩＰＲの検証にＩＰＲ辞書を使用する能力、ＩＰＲ辞書から発行されたＩＰＲに関する追加情報を提供する能力、およびＩＰＲ辞書中にＩＰＲを提供する能力を備える。

【０１６３】

図１２を参照すると、ＩｐｒＭｇｒＩｍｐｌプログラムは、通信プラットフォームサイトにおける全てのＩＰＲに対する収集ポイントである。このプログラムは、ＩｐｒＭｇｒＩｍｐｌ　ＣＯＲＢＡサーバオブジェクトを含む。ＩｐｒＭｇａｒＩｍｐｌオブジェクトは、アクティブ／待機の各プラットフォームマネージャノード上で稼動する。ＩｐｒＭｇｒＩｍｐｌが反応するアクティブ／待機状態は、通信プラットフォームマネージャノードのノードレベルのアクティブ／待機状態である。プラットフォームマネージャノードがアクティブ／待機状態を変更するとき、待機のＩｐｒＭｇｒＩｍｐｌオブジェクトは、そのインタフェースを公表せず、アクティブなＩｐｒＭｇｒＩｍｐｌオブジェクトは、そのＣＯＲＢＡインタフェースを公表することになる。こうすることにより、ＩｐｒＭｇｒとＩＰＲＣｌｉｅｎｔとのインターフェースの両方のクライアントユーザは、自身のＩＰＲをアクティブなＩｐｒＭｇｒＩｍｐｌオブジェクトに転送することになる。

【０１６４】

イベントマネージャサブシステムは、ＩＰＲを配布するためにＩＰＲサブシステム内で使用される。これにより、ＩＰＲを複数の宛先に転送することができる。イベントマネージャを使用することにより、インタフェースの変更をこうむることなく、追加のＩＰＲ特徴をシステムに容易に追加することができる。ＩＰＲサブシステムのイベントマネージャ機構は、現在、いくつかの既存のＩＰＲサービスを提供するために通信プラットフォーム内

で使用されている。リアルタイムＩＰＲ　ＧＵＩ２７０は、ＩＰＲが発生したときに、それらを表示する目的でＩＰＲを受け取るために登録する。Ｉｐｒ２ｈｏｓｔプログラム２７２は、ＩＰＲを受け取ってそれらをホストに転送するために、ＩＰＲサブシステムに登録する。ＩＰＲを受け取ってディスクに記録するために、ＩＰＲロガーも登録することができる。

【０１６５】

ｉｐｒ２ｈｏｓｔプログラム２７２は、ＩＰＲをホストに転送することを担当する。このプログラムは、ＩｐｒＭｇｒＩｍｐｌのイベントマネージャからＩＰＲを受け取り、引き続き転送するためにホストメッセージをフォーマットする。ホストに転送される全てのＩＰＲは、メッセージハンドラサブシステムを使用して、ＩＰＲ＿ＡＳＳＥＲＴ論理リンクを介してＩＰＲを転送する。

【０１６６】

ＩＰＲサブシステムは、２つの外部インタフェースを有する。すなわち、ＩＰＲクライアントインタフェース２７４およびＣＯＲＢＡ　ＩＰＲインタフェース２７６である。ＩＰＲクライアントインタフェース２７６は、以前に公開されたＰＡＣｏｎｆｉｇｕｒａｂｌｅエレメントとの後方互換性のために存在する。ＩＰＲクライアントインタフェース２７４から発行されたＩＰＲが、ＩＰＲクライアントコードによって変換されると、ＩＰＲは、ＩｐｒＭｇｒＩｍｐｌ　ＣＯＲＢＡインタフェースの使用によって発行され、アクティブなＩｐｒＭｇｒＩｍｐｌオブジェクトに送られる。このインタフェースは、旧来のＰＡＣｏｎｆｉｇｕｒａｂｌｅエレメントＩＰＲを、現在のＩＰＲサブシステム形式に変換するための入力として、ＬＯＣＩＰＲＤＢ．ＤＳＫ　ＩＰＲ辞書をまだ使用する。これには、ＩＰＲを発行するプログラムを有する各ノード上に、ＬＯＣＩＰＲＤＢ．ＤＳＫが常駐することが必要となる。ＬＯＣＩＰＲＤＢ．ＤＳＫ辞書は、以前に公開されたものでは、ＩＰＲがホストに転送される前に、ＩＰＲの検証を行うのに使用されていた。ＩＰＲをＬＯＣＩＰＲＤＢ．ＤＳＫ辞書に入力するためには、ＲｅｇｉｓｔｅｒＩＰＲユーティリティが使用される。データベースエントリ中のフィールドは、ＡＳＣＩＩキー（ＩＰＲテキスト）、ホストＩＰＲ番号、ＩＰＲ優先度、使用されるデータワード数、およびデータワード形式を含む。ＩＰＲＭｇｒをテストするために、ＩＰＲは、ｉｐｒ．ｉｎで定義されなければならず、これは、キーによる辞書に変換されることになる（ＲｅｇｉｓｔｅｒＩＰＲユーティリティを介して）。

【０１６７】

ＩｐｒＭｇｒＩｍｐｌインタフェースは、ＣＯＲＢＡ　ＩＤＬインタフェースである。ＩＰＲがこのインタフェースを使用して発行されると、ＩＰＲは、ＬＯＣＩＰＲＤＢ．ＤＳＫ辞書に入力される必要はない。ＩｐｒＭｇｒＩｍｐｌオブジェクトは、発行されたＩＰＲを受け取ると、そのＩＰＲ辞書中でそれを検索し、発行するＩＰＲイベントを構築する。ＩＰＲイベントは、ＩＰＲを発行したクライアントから渡された情報、およびＩＰＲ辞書からの情報を含む。ＩＰＲは、ＩＰＲの発行前に、ＩＰＲ辞書およびＭｅｇａＨｕｂホストのＩＰＲ辞書に追加されなければならない。ＩＰＲを、ＩｐｒＭｇｒＩｍｐｌ　ＩＰＲ辞書に追加するためには、ＩｐｒＤｒｉｖｅｒツールが使用される。ＩｐｒＭｇｒＩｍｐｌ　ＩＰＲ辞書内に場所を占めるように、ＶＡＸ　ＩＰＲファイルをＩｐｒＤｒｉｖｅｒで使用できる形式に変換するのを助けるために、再フォーマットおよび再フォーマット２スクリプトが存在する。

【０１６８】

図１３に、アプリケーションがＩＰＲを発行し、ＩＰＲマネージャがそれを処理し、イベントマネージャがそのＩＰＲを視覚表示用にＩＰＲ　ＧＵＩに送る筋書きを示す。

【０１６９】

１）ＩＰＲ　ＧＵＩが、ＩＰＲイベントマネージャに報告される全てのＩＰＲを受け取ることへのインタレストを登録する。

【０１７０】

２）アプリケーションがＩＰＲを発行する。

【０１７１】
３）ＩＰＲマネージャがイベントマネージャにＩＰＲを転送する。
【０１７２】
４）イベントマネージャがＩＰＲ　ＧＵＩにＩＰＲを配布する。
【０１７３】
図１４は、ＩＰＲビューＧＵＩのスクリーン印刷の一例である。ＩＰＲビューＧＵＩアプリケーションは、分割されたウィンドウにＩＰＲの表示を提供する。上部の枠には、カテゴリベースで表示されたコスト対時間によるＩＰＲの図表を示す。下部の枠は、ＩＰＲの従来の完全／簡略なテキストビューを表示する。サブカテゴリを見ることもでき、いくつかの表示カスタマイズが可能である。さらに、表示されるＩＰＲに対する、フィルタリングおよび強調も可能である。通信はＣＯＲＢＡを介して処理される。
【０１７４】
統計サービス
データ収集（ＤｃＭＰｒｏｃｅｓｓ、ＤｃＰｒｏｃｅｓｓ）
図１５を参照すると、データ収集サブシステム（ＤＣ）２９８は、ノード内のアプリケーションプログラムにトラフィック測定機能を提供する。これらの測定は、ペグカウンタ（Ｐｅｇｃｏｕｎｔｅｒ）クラスによって記録される総数、およびタイムメータ（ＴｉｍｅＭｅｔｅｒ）クラスによって記録される経過時間である。テストするペグカウンタ２９９は、共用記憶域３００およびセマフォを間接的にテストすることになる。クライアントプロセス３０１は、共用記憶域３００にペグし、データ収集２９８は、共用記憶域３００から収集してＤＣマスタ３０２に送る。３０分ごとに、データ収集２９８は、３０分あたりのペグカウンタスロット２９９をＤＣマスタ３０２（アクティブなプラットフォームマネージャノード内の）に送り、次いでデータ収集は、これらのスロットをゼロにする。アクティブなプラットフォームマネージャノード３０４は、待機プラットフォームマネージャノード３０６を更新する。
【０１７５】
図１６を参照すると、統計サービスまたはデータ収集サブシステム３２０は、プラットフォームのトラフィックの計量および測定能力を備える。このサブシステム３２０は、ペグカウンタ、タイムメータ、しきい値カウンタなどの統計測定の生成、収集、および報告と、収集および照会をサポートする。ペグカウンタ３２２およびタイムメータ３２４は、分散アプリケーションを介してサポートされるように示してある。データ収集サブシステム３２０によって与えられる特徴は、以下のものを含む。
【０１７６】
－　ペグカウンタ３２２およびタイムメータ３２４ＡＰＩサポート
－　複数のノードからの累積データの収集
－　統計をローカルで見るためのＧＵＩへの報告
－　報告カスタマイズ用にユーザによって定義される測定セット
しきい値カウンタ（ＴＣＳＳｅｒｖｅｒ）
しきい値カウンタサブシステムは、ｏｒｂｅｌｉｎｅＯＲＢの実行を使用して、オブジェクト要求ブローカ（ＯＲＢ）分散オブジェクトとして実行することができる。アプリケーションは、Ｏｒｂｅｌｉｎｅを介してプラットフォームマネージャノードに常駐するサーバオブジェクトに接続される。サーバは、分散オブジェクトメッセージング環境（ＤＯＭＥ）を介して、アプリケーションにカウンタしきい値交差を報告する。サーバオブジェクトは、しきい値カウンタサーバプロセス、ＴＣＳｅｒｖｅｒによって生成される。各ＴＣＳｅｒｖｅｒプロセスもまた、Ｏｒｂｅｌｉｎｅを介して遠隔ノード上でＴＣＳｅｒｖｅｒと通信し、したがって、カウンタはサイトにわたって同期化できる。ＴＣＳｅｒｖｅｒは、共通サービスライブラリ中に供給される永続性の辞書を、テンプレートクラスＲｅｐＳｈｍＤｉｃｔとして使用して、全てのカウンタを永続性の記憶域に保持する。
【０１７７】
図１７に、アプリケーションプロセス３４０とカウンタサーバプロセスとの間の通信パス

を示す。ＴＣＳｅｒｖｅｒプロセス３４２は、Ｏｒｂｅｌｉｎｅ３４４とＤＯＭＥ３４６の両方を介して、アプリケーションプロセス３４０と通信する。ＴＣＳｅｒｖｅｒプロセス３４２は、ｏｒｂｅｌｉｎｅ　ｉｍｐｌ＿ｉｓ＿ｒｅａｄｙループ内で稼動し、アプリケーションプロセス３４０と別のノード上のＴＣＳｅｒｖｅｒプロセス３４２のいずれかからのサービス要求を待機する。このＴＣＳｅｒｖｅｒプロセスは、ＤＯＭＥ　ＲｅｑＳｅｒｖ呼出しを行って、カウンタがそのしきい値に達したことをアプリケーションプロセス３４０に通知する。

【０１７８】

図１８を参照すると、しきい値カウンタサブシステム３６０ＡＰＩは、ｏｒｂｅｌｉｎｅ特有の実行部分をアプリケーションプログラマから隠している。その代わりに、サブシステムのクライアント側は、２つの層、すなわちＯＲＢ独立層３６２およびｏｒｂｅｌｉｎｅ依存層３６４で構成されることになる。サブシステムのｏｒｂｅｌｉｎｅ特有の実行は、アプリケーションプログラマより隠されているが、サブシステムの分散性質はそうではない。カウンタのインクリメントに必要とされる時間を最小限にするために、カウンタのインクリメントは、ＡＰＩ中でバッファされ、バッチでサーバに送られる。これは、アプリケーションが、ＡＰＩオブジェクト上の何らかの動作の成功または故障の通知を即時に受け取ることができないことを意味する。

【０１７９】

通信サービス

メッセージ処理（ＭｓｇＨｎｄｌ、ＬｉｎｋＸＸＸ）

図１９および２０に示すように、メッセージ処理サブシステム３７０は、メッセージベースのプロセッサ間通信サービスを提供する。一般に、サーバノード上のプロセス間における全てのプロセス間通信は、図２１に示す分散オブジェクトメッセージング環境（ＤＯＭＥ）３７２を介して実行される。ＤＯＭＥ３７２は、情報がノードの境界をわたって通信されなければならないとき、メッセージ処理サブシステム３７０を使用する。メッセージ処理サブシステム３７０はまた、ＳＣＰホストなど、サーバ以外の外部システムとの通信用に使用される。メッセージ処理サブシステム３７０は、以下の特徴を与える。

【０１８０】

－　複数プロトコル用の共通インタフェース
－　ＴＣＰ／ＩＰ　３７４
－　ＵＤＰ／ＩＰ　３７６
－　ＤＥＣＮＥＴ　３７８
－　同一の宛先を有する複数リンク用の単一アクセス識別子（論理リンクグループ名）
－　冗長リンク管理（スケーラビリティを向上させる）
－　リンク障害の復旧
－　非同期受信インタフェース

分散オブジェクトサービス

図２１を参照すると、ＤＯＭＥ３７２は、プロセス間クライアント／サーバ通信に使用されるクライアント／サーバインタフェースである。ＤＯＭＥ３７２は、サーバプロセス３８２がクライアントプロセス３８４によって使用されるオブジェクトおよびメンバ機能を登録することを可能にする、サーバインタフェース３８２を含む。ＤＯＭＥ３７２は、サーバ記述を記憶する共用記憶域データベース３８０と、他のノードからのサーバオブジェクト記述を維持するスタンドアロンのＤＯＭＥサービス（ＤＯＭＥＳｅｒｖｉｃｅｓ）プロセスとを含む。ＤＯＭＥ３７２はまた、ノードのＤＯＭＥデータベース中にあるどの登録済みサーバオブジェクトへのアクセスも提供する、クライアントインタフェース３８４も含む。

【０１８１】

プロセス間通信サブシステムは、主にＤＯＭＥで構成される。ＤＯＭＥは、サーバオブジェクトおよびその方法を登録する能力を、システム内の他のプロセスがそれらの方法を呼び出すことができるような形で、プロセスに提供する。ＤＯＭＥは、故障回復ソフトウェ

アの開発を補助する、多くの特別なルーティングオプションを含む、様々な登録およびアクセスのモードをサポートする。プロセス間通信サブシステムによって与えられる特徴は、以下のものを含む。

【０１８２】

－　ノードおよびサイトにわたる登録済みオブジェクト名の管理
－　優先順位のある要求の処理
－　アクティブ／待機オブジェクト要求のルーティング
－　共用オブジェクトのロード（Ｌｏａｄ　Ｓｈａｒｅｄ　Ｏｂｊｅｃｔ）要求のルーティング
－　オブジェクトのブロードキャスト（Ｂｒｏａｄｃａｓｔ　Ｏｂｊｅｃｔ）要求のルーティング
－　ブロック化／非ブロック化オブジェクト要求

共通サービス

共通ユーティリティサブシステムは、プラットフォーム層上でまたはその内部で稼動するように構成されるプロセスの迅速な開発を補助する、プログラミングツールのライブラリを提供する。共通ユーティリティサブシステムによって与えられる特徴は、以下のものを含む。

【０１８３】

－　コマンドラインオブジェクト
－　トレースオブジェクト
－　共用記憶域オブジェクト
－　セマファオブジェクト
－　キーによる辞書オブジェクト
－　リストオブジェクト
－　複製されたキーによる辞書オブジェクト
－　共用記憶域辞書オブジェクト
－　その他

ＤｂｇＴｒａｃｅオブジェクト

図２２を参照すると、ＤｂｇＴｒａｃｅ機能４００は、トレースバッファ、ファイル、および／または標準誤差にトレースメッセージを発行する能力を備える。トレースデータは、異なる２つの形式、すなわち標準印刷形式およびデータバッファダンプ形式で入力することができる。マスク４０２は、異なるレベルのメッセージをフィルタリングするのに使用することができる。各ＤｂｇＴｒａｃｅグループに対して３２の可能なマスクレベルが存在する。

【０１８４】

ＤｂｇＣｎｔｌインタフェース４０４は、ＤｂｇＴｒａｃｅオブジェクト４００に対する制御インタフェースである。これは、ユーザが、ＤｂｇＴｒａｃｅ機能４００の多くの異なる側面を指定することを可能にする。このインタフェースはまた、ユーザがＤｂｇＴｒａｃｅオブジェクト４００上で以下のことを行うのを可能にする。

【０１８５】

－　ＤｂｇＴｒａｃｅグループ４００に対するマスク４０２を設定／入手する。

【０１８６】

－　内部メッセージバッファ４１０のサイズを設定／入手する。

【０１８７】

－　存在するグループのリストを入手する。

【０１８８】

－　標準誤差への表示をオン／オフに切り替える。

【０１８９】

－　ファイルにトレースを１つづつダンピングするのをオン／オフに切り替える。

【０１９０】

－　トレースを上書きされる前にファイルからトレースをダンプする能力をイネーブル／ディスエーブルにする。

【０１９１】

ＤｂｇＤｉｓｋインタフェースは、全ての書込み要求上でどのファイルにトレースバッファ４１０が書き込まれることになるかを、ユーザが指定できるようにする。

【０１９２】

ＤｂｇＴｒａｃｅ機能４００は、それぞれが複数のグループのうちの１つに属することのできる、異なるＤｂｇＴｒａｃｅオブジェクト４００をユーザが生成することを可能にする。これにより、ユーザは、グループごとに固有のマスク値を有することができる。ＤｂｇＴｒａｃｅインタフェース４００を通して発行される全てのトレースは、内部メッセージバッファに記憶される。ユーザはまた、内部バッファに加えて標準誤差にもトレースを発行するかどうかも指定できる。

【０１９３】

トレースオブジェクト

トレースオブジェクトは、標準誤差にトレースメッセージを任意に発行する能力をユーザに提供する。ユーザがトレースを発行するとき、このトレースが出力されることになるトレースレベルを表すマスクが指定される。トレースインタフェースは、そのＵＮＩＸプロセス中のトレースの全てのインスタンスが、トレースメッセージを発行するか否かを決定するのに使用することになるマスクを、ユーザが指定できるようにする。トレースマスクは、８つの固有のマスク値をサポートすることができる。

【０１９４】

辞書管理システム

図２３を参照すると、辞書管理は、データの記憶およびアクセスをサポートするように構成されたクラスを提供する。辞書は、ディスク（永続的）またはメモリに記憶することができる。辞書はまた、私用（ローカルプロセスのみによって使用される）、または共用（複数のプロセスからアクセスできる）とすることもできる。これらの辞書の目的は、アプリケーションプログラムによって定義される。ＤｍｓＭａｓｔｅｒ４３０とＤｍｓＳｅｒｖｅｒ４３２との間の主要な相互作用は、ＤｍｓＭａｓｔｅｒ４３０が、アプリケーションから更新メッセージを受け取ったときに、ＤｍｓＳｅｒｖｅｒ４３２を更新することである。ＤｍｓＭａｓｔｅｒ４３０は、プラットフォームマネージャノード内でアクティブ／待機で稼動し、ＤｍｓＳｅｒｖｅｒ４３２は、全て（または１つのサブセット）のＩＰＵ内で稼動する。

【０１９５】

イベントサービス

イベントサービスは、タスクにとって重要な特定の出来事を、ゆるく結合されたプロセス間で生成および配布する能力を備える。イベントの一例は、入出力転送の完了である。イベントサービスは、ＣＯＲＢＡベースのプロセス間通信機能とすることができる。これは、オブジェクトによって動作が実行されることになる、標準的なＣＯＲＢＡ要求を使用する。これは、イベントマネージャ実行プログラムを通して達成される。

【０１９６】

オブジェクトに対して異なる２つの役割を定義することにより、通信はオブジェクト間で分断され、非同期通信が生成される。１つのオブジェクトは、新しいイベントを受け取って蓄積し、もう一方のオブジェクトは、これらの新しいイベントに転送される関心を登録する。これは、２つのＣＯＲＢＡクラス、すなわちイベントマネージャおよびイベントレシーバによって達成される。イベントマネージャは、新しいイベントを受け取るためのインタフェース定義言語（ＩＤＬ）インタフェースを提供する。イベントレシーバは、イベントを受け取ることに関心のあるクライアントのためのインタフェース定義言語インタフェースを提供する。

【０１９７】

ソフトウェアおよびハードウェア表現

図２４に、通信プラットフォームシステムのハードウェア図を示す。最上レベルでは、通信プラットフォームシステムは、１つまたは複数のサイト４４０を含む。サイト４４０内には、複数のノード４４２が存在する。

【０１９８】

ソフトウェア表現は、ソフトウェアの構成要素を共にグループ化することのできる階層構造である。図２５に、この階層構造を示す。最上レベルには、アプリケーション４５０が存在する。アプリケーション４５０は、１つまたは複数の構成可能エレメントセット４５２からなり、構成可能エレメントセットは、１つまたは複数の構成可能エレメント４５４からなる。複数のアプリケーション４５０を、システム内で定義することができる。システム内の全てのアプリケーション４５０は、システムのソフトウェア表現を構成する。

【０１９９】

ソフトウェアの、システムのハードウェア表現への動的マッピングを図２６に示すが、これは、どのようにアプリケーション４５０の各部分が、ノード４４２上に配置されるかを示している。サイト４４０はアプリケーション４５０を含む。アプリケーション４５０は、プロセッササービスグループ４５６を有する。プロセッササービスグループ４５６は、複数のノード４４２にまたがる。ノード４４２上には、構成可能エレメントセット４５２が配置される。構成可能エレメントセット４５２内には、構成可能エレメント４５４が常駐する。例えば、時間依存ルーティングアプリケーションのソフトウェア表現が、２つの構成可能エレメントセット、すなわち西海岸セット（ＷｅｓｔＣｏａｓｔＳｅｔ）および東海岸セット（ＥａｓｔＣｏａｓｔＳｅｔ）を有するとする。西海岸セット内では、時間依存ルーティングアプリケーションは、西海岸の呼出しを処理するように的を絞られたノード上で稼動する必要のある全てのプログラムを有することができる。これらのプログラムには、西海岸での処理用に特に構成されたデータベースプログラム、リンクプロセスなどを含めることができる。東海岸セット内では、時間依存ルーティングアプリケーションは、東海岸の呼出しを処理するように的を絞られたノード上で稼動する必要のある全てのプログラムを有することができる。次いで、時間依存ルーティングアプリケーションがサイトに割り当てられる。時間依存ルーティングアプリケーションを稼動させることになるノードは、プロセッササービスグループ内にグループ化される。次いで、アプリケーション用の構成可能エレメントセットは、時間依存ルーティングアプリケーションのプロセッササービスグループ内に配置されたノード上に配置される。

【０２００】

本発明のいくつかの実施形態およびその利点を詳細に述べたが、本発明の教示、添付の特許請求の範囲に述べる本発明の趣旨および範囲を逸脱することなく、変異、変更、代用、変換、修正、変形、および代替が、それに対して可能であることを理解されたい。

【図面の簡単な説明】

【図１】　本発明の一実施形態による通信プラットフォームアーキテクチャ層の概略ブロック図である。

【図２】　本発明の一実施形態による通信プラットフォーム概念構成要素の概略ブロック図である。

【図３】　本発明の一実施形態による通信プラットフォームの概念構成要素、およびそれらの間の関係のブロック図である。

【図４】　本発明の一実施形態による通信プラットフォームの論理区画分けの概略ブロック図である。

【図５】　本発明の一実施形態による通信プラットフォームサービス、およびその依存関係の概略ブロック図である。

【図６】　本発明の一実施形態による通信プラットフォームの物理的区画分けの概略ブロック図である。

【図７Ａ】　本発明の一実施形態によるＮｅｔＰＭのテストフローのブロック図である。

【図７Ｂ】　本発明の一実施形態によるＮｅｔＰＭの時間同期フローのブロック図である。

【図７Ｃ】　本発明の一実施形態によるネットワーク管理サービスとノード管理サービスとの間の障害検出および相互作用を示すブロック図である。
【図７Ｄ】　本発明の一実施形態によるコアサービス間の相互作用を示すブロック図である。
【図８】　本発明の一実施形態による通信プラットフォームノードの状態移行図である。
【図９Ａ】　本発明の一実施形態によるノードスタートアッププロセスの概略ブロック図である。
【図９Ｂ】　本発明の一実施形態によるノード初期設定プロセスのメッセージ流れ図である。
【図９Ｃ】　本発明の一実施形態によるノード初期設定プロセスのメッセージ流れ図である。
【図９Ｄ】　本発明の一実施形態によるノード初期設定プロセスのメッセージ流れ図である。
【図１０】　本発明の一実施形態によるサービス管理インタフェースプロトコルのメッセージ流れ図である。
【図１１】　本発明の一実施形態によるイベントマネージャ使用を示す概略ブロック図である。
【図１２】　本発明の一実施形態による情報および問題報告（ＩＰＲ）の概略流れ図である。
【図１３】　本発明の一実施形態によるＩＰＲ処理の概略流れ図である。
【図１４】　本発明の一実施形態による例示的ＩＰＲ閲覧グラフィカルユーザインタフェースである。
【図１５】　本発明の一実施形態によるデータ収集を示す概略ブロック図である。
【図１６】　本発明の一実施形態によるデータ収集サブシステムの概略ブロック図である。
【図１７】　本発明の一実施形態によるしきい値カウンタデータ通信パスの概略ブロック図である。
【図１８】　本発明の一実施形態によるしきい値カウンタサブシステムの概略ブロック図である。
【図１９】　本発明の一実施形態によるメッセージ取扱いサブシステムの概略ブロック図である。
【図２０】　本発明の一実施形態によるメッセージ取扱いテストの概略ブロック図である。
【図２１】　本発明の一実施形態による分散オブジェクトメッセージ環境の概略ブロック図である。
【図２２】　本発明の一実施形態による内部デバッグおよび追跡オブジェクト関係の概略ブロック図である。
【図２３】　本発明の一実施形態による辞書管理システムの概略ブロック図である。
【図２４】　本発明の一実施形態による通信プラットフォームのハードウェア表現の概略ブロック図である。
【図２５】　本発明の一実施形態による通信プラットフォームのソフトウェア表現の概略ブロック図である。
【図２６】　本発明の一実施形態による通信プラットフォームのハードウェア上へのソフトウェアの動的マッピング表現を示す概略ブロック図である。

【図１】

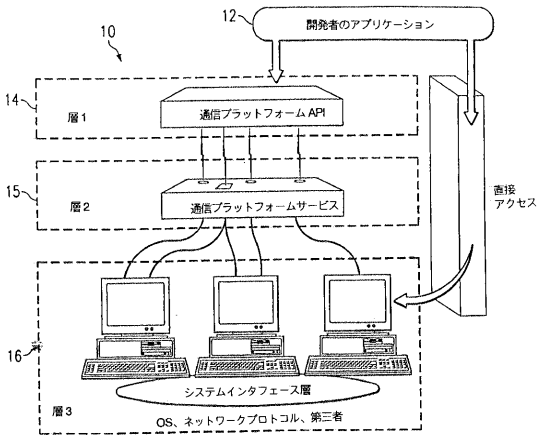


FIG. 1

【図２】

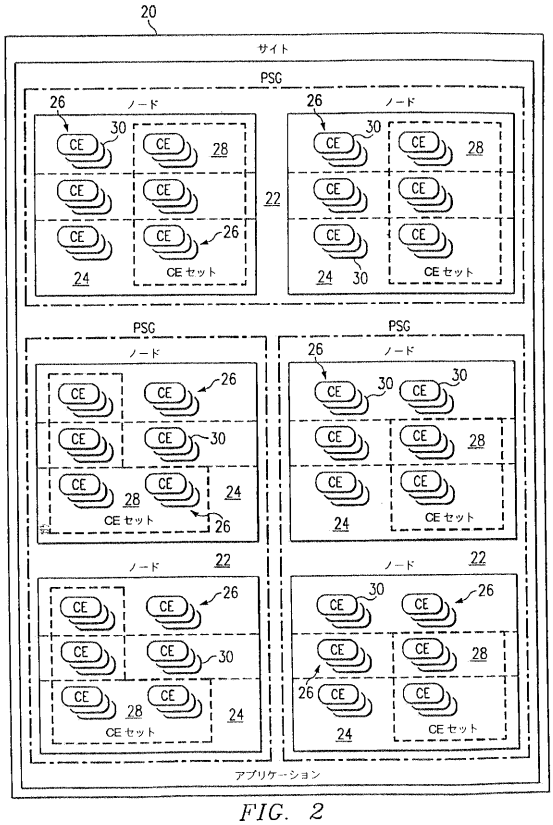


FIG. 2

【図３】

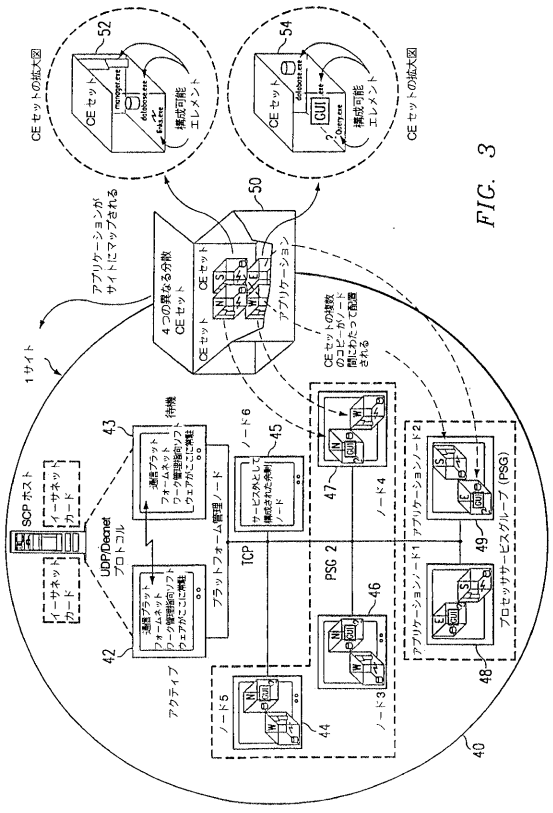


FIG. 3

【図４】

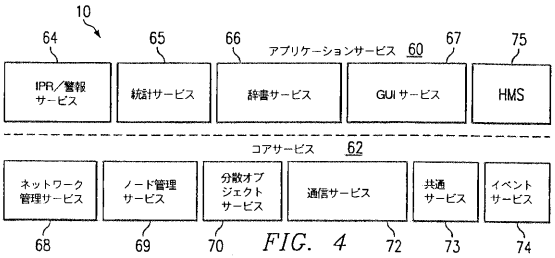


FIG. 4

【図５】

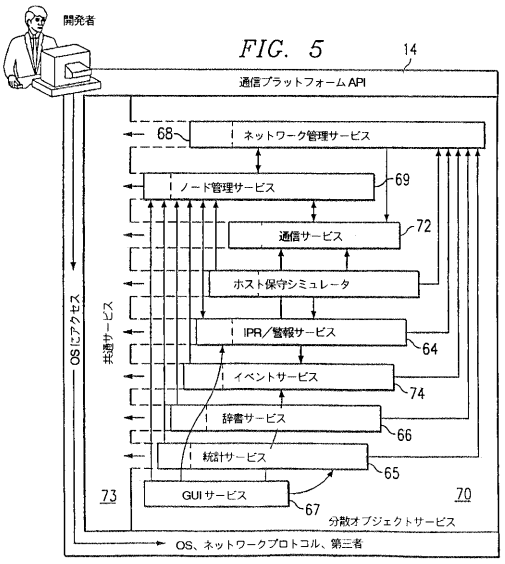


FIG. 5

【図６】

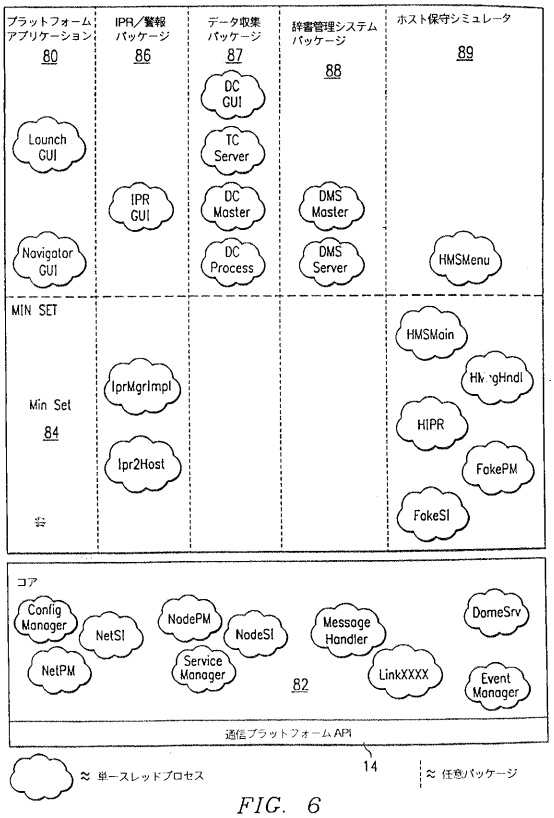


FIG. 6

【図７Ａ】

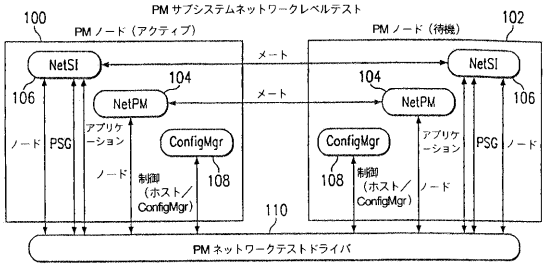


FIG. 7A

【図７Ｂ】

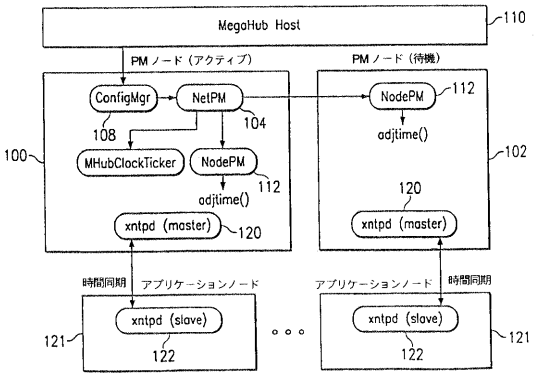


FIG. 7B

【図７Ｃ】

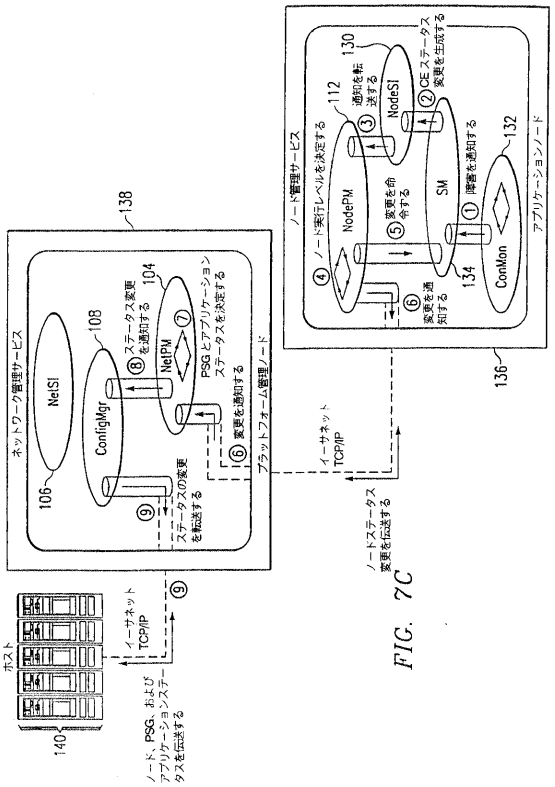


FIG. 7C

【図７Ｄ】

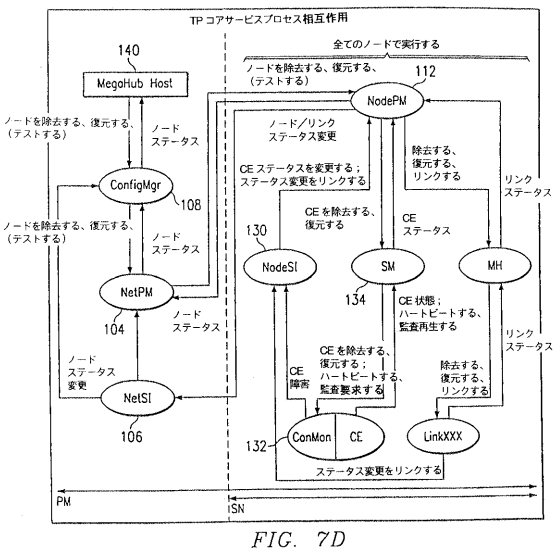


FIG. 7D

【図８】

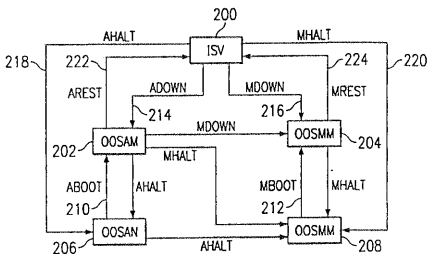


FIG. 8

【図９Ａ】

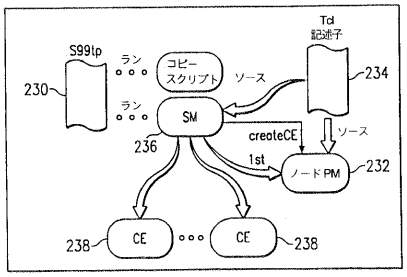


FIG. 9A

【図９Ｂ】

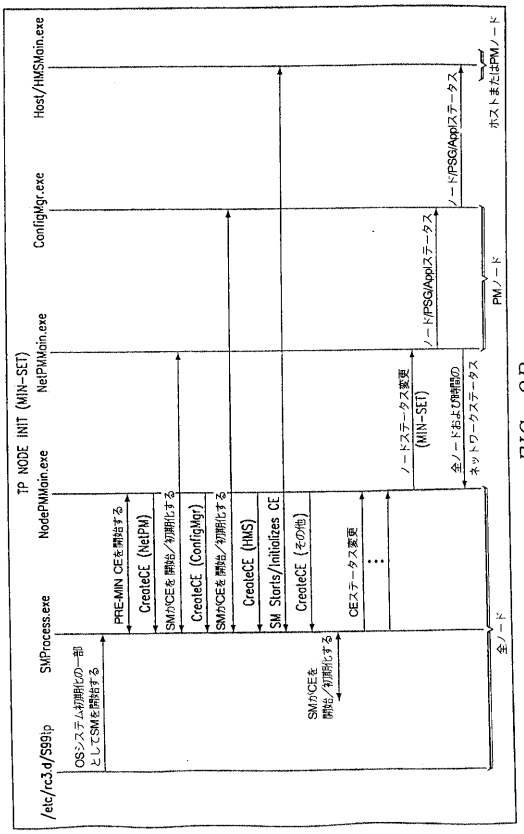


FIG. 9B

【図９Ｃ】

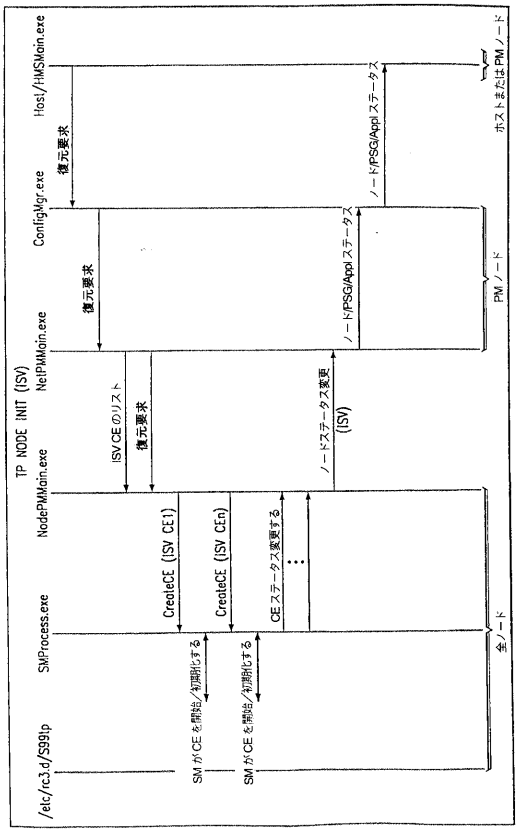


FIG. 9C

【図９Ｄ】

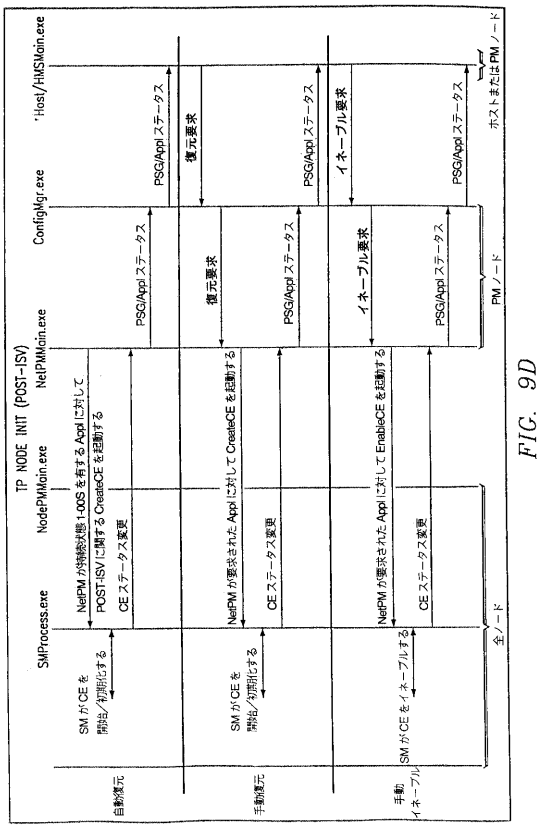
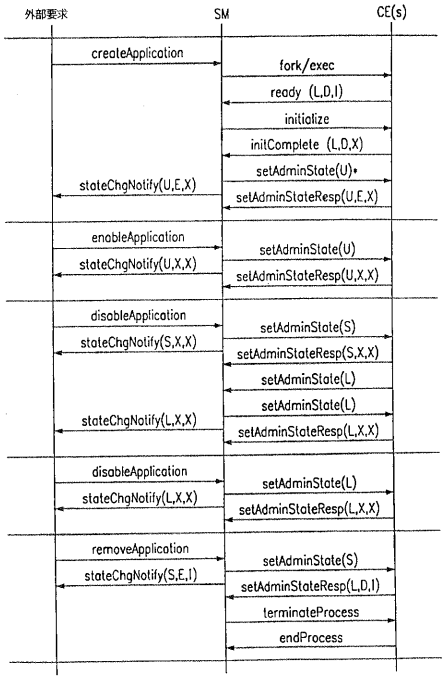


FIG. 9D

【図１０】

FIG. 10



* このメッセージは、自動ロック解除ステータスが AUTO_UNLOCKED に設定される場合にのみ送られる。

【図１１】

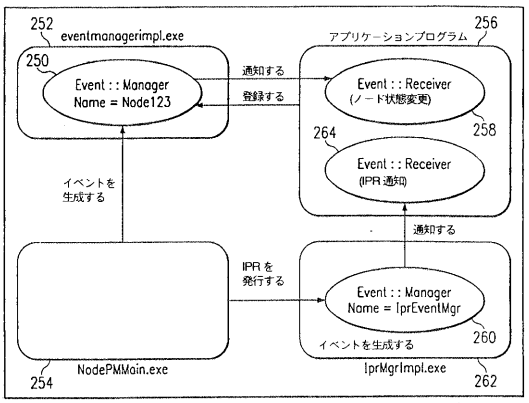
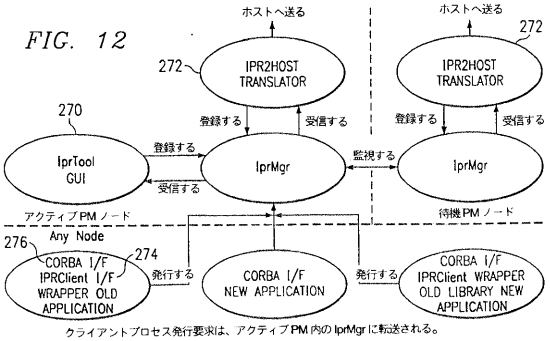


FIG. 11

【図１２】

FIG. 12



クライアントプロセス発行要求は、アクティブPM 内の IprMgr に転送される。

【図１３】

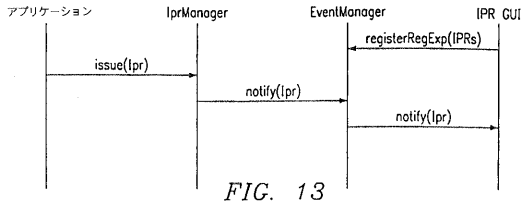


FIG. 13

【図１４】

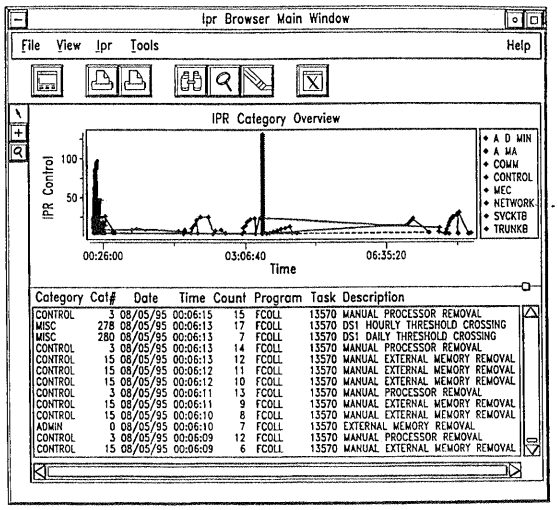


FIG. 14

【図１５】

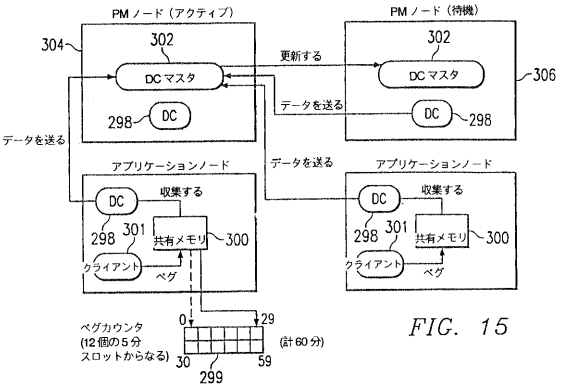


FIG. 15

【図１６】

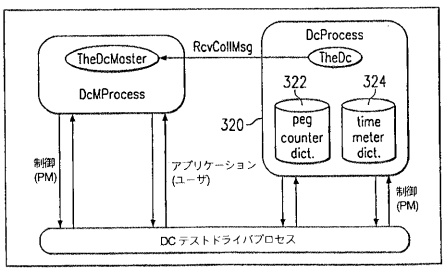


FIG. 16

【図１７】

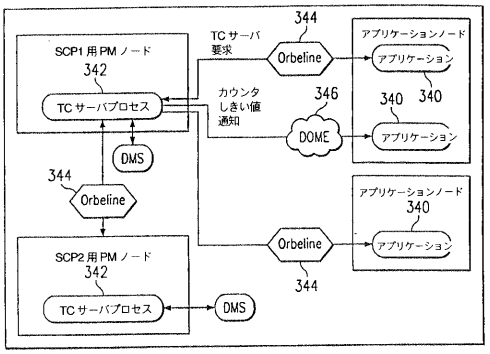


FIG. 17

【図１８】

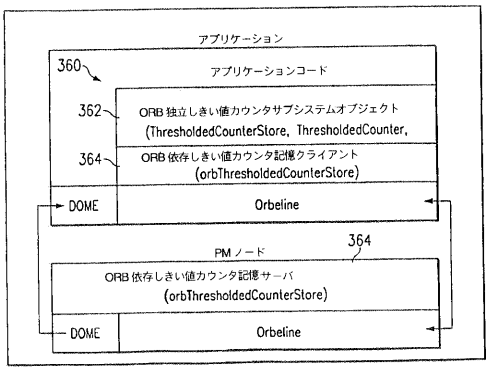


FIG. 18

【図１９】

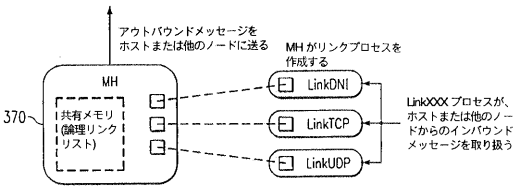


FIG. 19

【図２０】

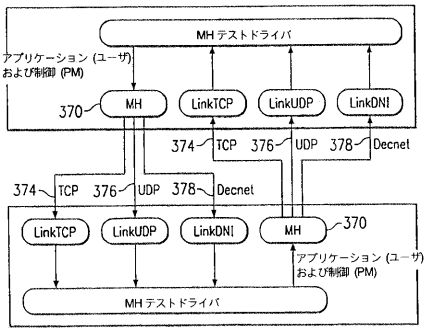


FIG. 20

【図２１】

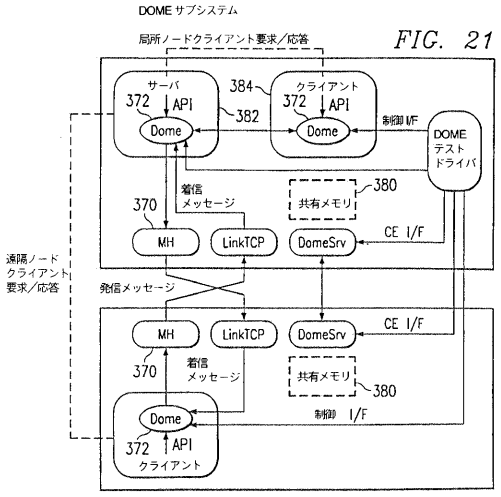


FIG. 21

【図２２】

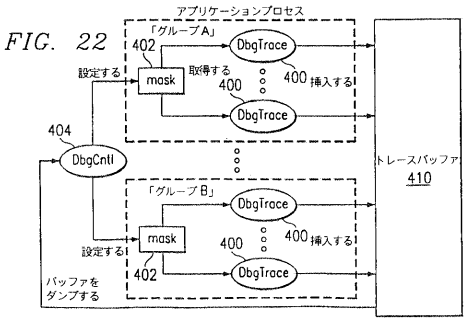


FIG. 22

【図２３】

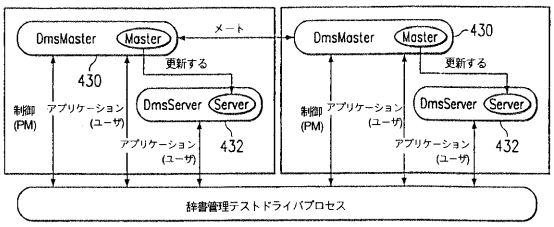


FIG. 23

【図２４】

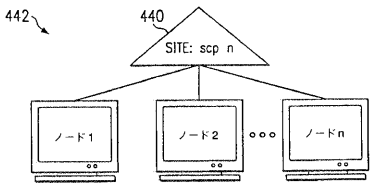


FIG. 24

【図２５】

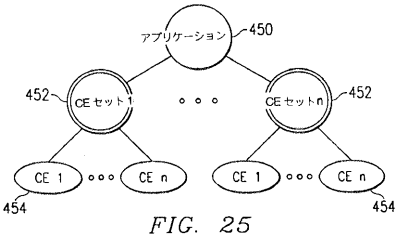


FIG. 25

【図２６】

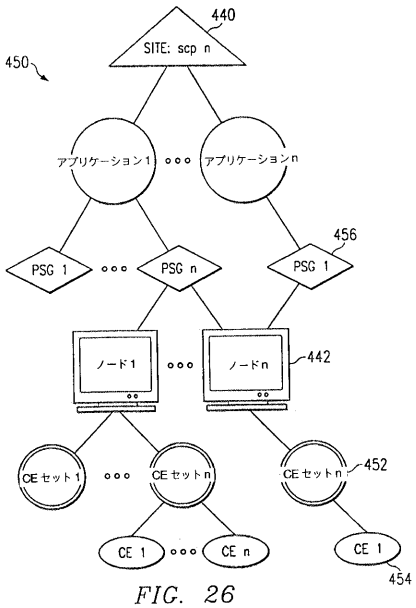

440
SITE: scp n
450
アプリケーション1
アプリケーションn
456
PSG 1
PSG n
PSG 1
ノード1
ノードn
442
CEセット1
CEセットn
CEセットn
452
CE 1
CE n
CE 1
454

FIG. 26

フロントページの続き

(72)発明者　マクダニエル，デイビツド・ダブリユ
アメリカ合衆国、テキサス・７５２１１、ダラス、タープレイ・アベニユー・１２０３
(72)発明者　バツテローニ，ジエームズ・アール
アメリカ合衆国、テキサス・７５０９８、ウエイリー、ウエスト・オーク・ストリート・４１３
(72)発明者　ジヤガーズ，ステイーブン・ビー
アメリカ合衆国、テキサス・７５００７、キヤロルトン、バークレイ・１５０８
(72)発明者　ワーライン，マーク・イー
アメリカ合衆国、テキサス・７５００２、アーリン、シカモア・クリーク・８１３

合議体
審判長　竹井　文雄
審判官　新川　圭二
審判官　猪瀬　隆広

(56)参考文献　井上友二，外４名，マルチメディアネットワークシリーズ　ネットワーク・アーキテクチャ　第１版、株式会社オーム社，平成６年６月５日，ｐ．２２，４５，５７，１０７，１３６，１４０，１４９，１５４，１６０
矢田浩二，外６名，ＡＴＭトランスポートネットワークのＯｐＳの構築　－ＴＮＭＳカーネルを用いたＯｐＳ開発－，電子情報通信学会技術研究報告，１９９３年１１月，９３巻，３３７号，１７－２２頁，ＣＳ９３－１３５
棚橋敦，外３名、ＡＴＭネットワークにおけるＴＭＮカスタマ網管理エージェントの実装，電子情報通信学会技術研究報告，１９９７年６月２３日，９７巻，１２７号，２５－３０頁，ＯＣＳ９７－１１

(58)調査した分野(Int.Cl.，ＤＢ名)
H04Q 3/00
H04M 3/00