(19)日本国特許庁(JP)　(12)特　許　公　報(B2)　(11)特許番号
特許第3739633号
(P3739633)
(45)発行日　平成18年1月25日(2006.1.25)　(24)登録日　平成17年11月11日(2005.11.11)

(51)Int.Cl.　　F I
G06F 9/54 (2006.01)　　G06F 9/46 480C
G06F 9/44 (2006.01)　　G06F 9/44 530M

請求項の数 25　(全 20 頁)

(21)出願番号　特願2000-158179(P2000-158179)
(22)出願日　平成12年5月29日(2000.5.29)
(65)公開番号　特開2001-14179(P2001-14179A)
(43)公開日　平成13年1月19日(2001.1.19)
審査請求日　平成13年8月21日(2001.8.21)
(31)優先権主張番号　09/322362
(32)優先日　平成11年5月28日(1999.5.28)
(33)優先権主張国　米国(US)

(73)特許権者　596092698
ルーセント　テクノロジーズ　インコーポレーテッド
アメリカ合衆国．０７９７４－０６３６　ニュージャーシィ，マレイ　ヒル，マウンテン　アヴェニュー　６００
(74)代理人　100064447
弁理士　岡部　正夫
(74)代理人　100085176
弁理士　加藤　伸晃
(74)代理人　100106703
弁理士　産形　和央
(74)代理人　100096943
弁理士　臼井　伸一

最終頁に続く

(54)【発明の名称】ソフトウェアモジュールの間で情報を交換するシステムと方法

(57)【特許請求の範囲】

【請求項１】

オブジェクト指向のソフトウエアプログラムのソフトウェアモジュール間で情報を交換するシステムであって、

対応するイベントクラスを有する１もしくは複数のトリガーイベントの発生について他のモジュールに通知するノーティファイヤーを含むソフトウエアモジュールを登録するための第１のルックアップテーブルと、対応するイベントクラスを有する１もしくは複数のトリガーイベントについての通知を受けるオブザーバーを含むソフトウエアモジュールを登録するための第２ルックアップテーブルとを備えるブローカーと、

該オブジェクト指向のソフトウエアプログラムにより生成され、第１のイベントクラスを有する所定の第１のトリガーイベントの発生を検出するように作られた第１のノーティファイヤーと、

第２のイベントクラスを有する所定のトリガーイベントの発生についての通知を受ける関心がある、該オブジェクト指向のソフトウエアプログラムにより生成される第１のオブザーバーとを備え、

該第１のノーティファイヤーが生成された場合には、該ブローカーは該第１のノーティファイヤーを該第１のルックアップテーブルに追加し、該第１のオブザーバーがサブスクライブメッセージを該ブローカーに送信した場合には該ブローカーは該第１のオブザーバーを該第２のルックアップテーブルに追加し、及び

該ブローカーは、該第１のルックアップテーブルと該第２のルックアップテーブルとを

比較し、該第１のイベントクラスと該第２のイベントクラスとが同一であることを検出すると、該ブローカーは該第１のノーティファイヤーに対して、該第１の所定のトリガーイベントの発生について直接に該第１のオブザーバーに知らせるように指示することを特徴とするシステム。

【請求項２】

第１のノーティファイヤー、第１のオブザーバー、前記ブローカーはそれぞれ、コンピュータのメモリーの別々のアドレス可能な位置に存在することを特徴とする請求項１記載のシステム。

【請求項３】

第１のノーティファイヤー、第１のオブザーバー、前記ブローカーはそれぞれ、１もしくは複数のコンピュータ上に存在することを特徴とする請求項２記載のシステム。

【請求項４】

前記ブローカーは、ノーティファイヤー－オブザーバー接続を登録する第３のルックアップテーブルを更に有し、前記ノーティファイヤー－オブザーバー接続の１つが、第１のイベントクラスと第２のイベントクラスが同じである場合に第３のルックアップテーブルに加えられることを特徴とする請求項１記載のシステム。

【請求項５】

第３のイベントクラスを有する所定のトリガーイベントの発生についての通知を受けることに関心のある第２のオブザーバーを更に有し、

第２のオブザーバーがサブスクライブメッセージを前記ブローカーに送った際に、前記ブローカーは第２のオブザーバーを第２のルックアップテーブルに加え、第１のイベントクラスと第３のイベントクラスが同じであることを前記ブローカーが検出した際、所定の第１のトリガーイベントの発生について第２のオブザーバーに直接通知するように前記ブローカーが第１のノーティファイヤーに指示することを特徴とする請求項１記載のシステム。

【請求項６】

第２のオブザーバーは、第５のイベントクラスを有する所定のトリガーイベントの発生についての通知を受けることに更に関心があり、第２のオブザーバーは第２のルックアップテーブルにそのように登録され、

第４のイベントクラスを有する所定の第２のトリガーイベントの発生を検出するように適合する第２のノーティファイヤーを更に有し、第２のノーティファイヤーが生成された際に前記ブローカーは第２のノーティファイヤーを第１のルックアップテーブルに加え、第４のイベントクラスと第５のイベントクラスが同じであることを前記ブローカーが検出した際に、所定の第２のトリガーイベントの発生について第２のオブザーバーに直接通知するように前記ブローカーが第２のノーティファイヤーに指示することを特徴とする請求項５記載のシステム。

【請求項７】

第４のイベントクラスを有する所定の第２のトリガーイベントの発生を検出するように適合する第２のノーティファイヤーを更に有し、

第２のノーティファイヤーが生成された際に前記ブローカーは第２のノーティファイヤーを第１ルックアップテーブルに加え、第２のイベントクラスと第４のイベントクラスが同じであることを前記ブローカーが検出した際に、所定の第２のトリガーイベントの発生について第１のオブザーバーに直接通知するように前記ブローカーが第２のノーティファイヤーに指示することを特徴とする請求項１記載のシステム。

【請求項８】

第２のノーティファイヤーは、第６のイベントクラスを有する所定の第３のトリガーイベントの発生を検出するように更に適合し、第２のノーティファイヤーは第１のルックアップテーブルにそのように登録され、

第３のイベントクラスを有する所定のトリガーイベントの発生についての通知を受けることに関心がある第２のオブザーバーを更に有し、第２のオブザーバーがサブスクライブ

メッセージを送った際に前記ブローカーは第２のオブザーバーを第２のルックアップテーブルに加え、第６のイベントクラスと第３のイベントクラスが同じであることを前記ブローカーが検出した際に所定の第３のトリガーイベントの発生について直接第２のオブザーバーに通知するように前記第２のブローカーが前記ノーティファイヤーに指示することを特徴とする請求項７記載のシステム。

【請求項９】

ソフトウェアモジュールを生成するように適合するクリエーターを更に有し、前記ブローカーが生成された際に前記ブローカーは前記クリエーターを第１のルックアップテーブルに加えることを特徴とする請求項１記載のシステム。

【請求項１０】

前記ブローカーが生成された際に前記ブローカーは前記ブローカーを第２のルックアップテーブルに加えることを特徴とする請求項１記載のシステム。

【請求項１１】

オブジェクト指向のソフトウエアプログラムのソフトウェアモジュール間で情報を交換する方法であって、

第１のルックアップテーブルと第２のルックアップテーブルを備えるブローカーを該オブジェクト指向のソフトウエアプログラムにより生成するステップと、

第１のイベントクラスを有する所定の第１のトリガーイベントの発生を検出するように適合する第１のノーティファイヤーを該オブジェクト指向のソフトウエアプログラムにより生成するステップと、

第１のノーティファイヤーのクリエートの際該ブローカーは第１のルックアップテーブルに第１のイベントクラスとともに第１のノーティファイヤーをリストに入れるステップと、

第２のイベントクラスを有する所定のトリガーイベントの発生についての通知を受けることに関心がある第１のオブザーバーを該オブジェクト指向のソフトウエアプログラムにより生成するステップと、　第１のオブザーバーが該ブローカーにサブスクライブした際に、該ブローカーは第２のルックアップテーブルに第２のイベントクラスとともに第１のオブザーバーをリストに入れるステップと、

該ブローカーは第１のルックアップテーブルを第２のルックアップテーブルと比較するステップと、

第１のイベントクラスと第２のイベントクラスが同じである場合に、所定の第１トリガーイベントの発生を第１のオブザーバーに直接通知するように該ブローカーが第１のノーティファイヤーに指示するステップとを有することを特徴とする方法。

【請求項１２】

前記ブローカーが生成された際に前記ブローカーが前記ブローカーを第２のルックアップテーブルに加えるステップを更に有することを特徴とする請求項１１記載の方法。

【請求項１３】

所定の第１トリガーイベントの発生を第１のノーティファイヤーが検出するステップと、　第１のイベントクラスと第２のイベントクラスが同じである場合に、所定の第１のトリガーイベントの発生について第１のノーティファイヤーが第１のオブザーバーに通知するステップとを有することを特徴とする請求項１１記載の方法。

【請求項１４】

ソフトウェアモジュールを生成するように作られたクリエーターを設けるステップと、

前記ブローカーが生成された際に前記ブローカーが前記クリエーターを第１のルックアップテーブルに加えるステップとを有することを特徴とする請求項１１記載の方法。

【請求項１５】

第１のノーティファイヤーを生成した後に、前記クリエーターが第１のノーティファイヤーをレジストリーに加えるステップと、

第１のオブザーバーを生成した後に前記クリエーターが第１のオブザーバーを前記レジストリーに加えるステップとを有することを特徴とする請求項１４記載の方法。

【請求項１６】
第１のイベントクラスと第２のイベントクラスが同じである場合に前記ブローカーがノーティファイヤー－オブザーバー接続を第３のルックアップテーブルに加えるステップを有することを特徴とする請求項１１記載の方法。
【請求項１７】
第３のイベントクラスを有する所定のトリガーイベントの発生についての通知を受ける関心のある第２のオブザーバーを生成し、
第２のオブザーバーが前記ブローカーにサブスクライブする際に、前記ブローカーは第２のルックアップテーブルにおいて第３のイベントクラスとともに第２のオブザーバーをリストに入れるステップと、
前記ブローカーが第１のルックアップテーブルを第２のルックアップテーブルと比較するステップと、
第１のイベントクラスと第３のイベントクラスが同じである場合に所定の第１のトリガーイベントの発生についての第２のオブザーバーに直接通知するように前記ブローカーが第１のノーティファイヤーに指示するステップとを有することを特徴とする請求項１１記載の方法。
【請求項１８】
第４のイベントクラスを有する所定の第２のトリガーイベントの発生を検出するように適合した第２のノーティファイヤーを前記ブローカーが生成するステップと、
第２のノーティファイヤーが生成された際に、前記ブローカーが第１のルックアップテーブルにて第２のノーティファイヤーを第４のイベントクラスとともにリストに入れるステップと、
前記ブローカーが第１のルックアップテーブルを第２のルックアップテーブルと比較するステップと、
第３のイベントクラスと第４のイベントクラスが同じである場合に、所定の第２のトリガーイベントの発生について第２のオブザーバーに直接通知するように前記ブローカーが第２のノーティファイヤーに指示するステップとを有することを特徴とする請求項１７記載の方法。
【請求項１９】
第４のイベントクラスを有する所定の第２のトリガーイベントの発生を検出するように適合した第２のノーティファイヤーを生成するステップと、
第２のノーティファイヤーが生成された際に前記ブローカーは第１のルックアップテーブルにて第２のノーティファイヤーを第４のイベントクラスとともにリストに入れるステップと、
前記ブローカーは第１のルックアップテーブルを第２のルックアップテーブルと比較するステップと、
第２のイベントクラスと第４のイベントクラスが同じである場合に、所定のトリガーイベントの発生について第１のオブザーバーに直接通知するように前記ブローカーが第２のノーティファイヤーに指示するステップとを有することを特徴とする請求項１１記載の方法。
【請求項２０】
各ノーティファイヤーがサービスしているかを判断するために前記ブローカーが第１のルックアップテーブルにリストされた各ノーティファイヤーに問い合わせるステップと、
サービスしていない各ノーティファイヤーを第１のルックアップテーブルから前記ブローカーが除去するステップと、
第１のルックアップテーブルから除去された各ノーティファイヤーについて前記ブローカーが全てのオブザーバーに知らせるステップとを有することを特徴とする請求項１１記載の方法。
【請求項２１】
サービスしていないノーティファイヤーとして第１のルックアップテーブルから除去さ

れた各ノーティファイヤーについて前記ブローカーが前記全てのオブザーバーに知らせるステップを有することを特徴とする請求項２０記載の方法。

【請求項２２】

各オブザーバーがサービスしているかどうかを第２のルックアップテーブルにリストされた各オブザーバーに前記ブローカーが問い合わせるステップと、

サービスしていない各オブザーバーを第２のルックアップテーブルから前記ブローカーが除去するステップと、

第２のルックアップテーブルから除去された各オブザーバーについて全てのノーティファイヤーに前記ブローカーが通知するステップとを有することを特徴とする請求項１１記載の方法。

【請求項２３】

第２のルックアップテーブルから除去された各オブザーバーと関連する全てのノーティファイヤーに対して各オブザーバーに通知することをやめるように前記ブローカーが指示するステップを有することを特徴とする請求項２２記載の方法。

【請求項２４】

第１のルックアップテーブルから第１のノーティファイヤーを除去するステップと、

前記除去するステップの後で、前記ブローカーが第１のルックアップテーブルと第２のルックアップテーブルを比較するステップと、

第１のイベントクラスと第２のイベントクラスが同じである場合に第１のノーティファイヤーがサービスしてないことを前記ブローカーが第１のオブザーバーに知らせるステップとを有することを特徴とする請求項１１記載の方法。

【請求項２５】

第２のルックアップテーブルから第１のオブザーバーを除去するステップと、

前記ブローカーが第１のルックアップテーブルを第２のルックアップテーブルと比較するステップと、

第１のイベントクラスと第２のイベントクラスが同じである場合に、所定の第１のトリガーイベントの発生について第１のオブザーバーに通知することをやめるように前記ブローカーが第１のノーティファイヤーに指示するステップとを有することを特徴とする請求項１１記載の方法。

【発明の詳細な説明】

【０００１】

【発明の属する技術分野】

本発明は、ソフトウェアモジュールの間で情報交換するシステムおよび方法に関し、特に、ノーティファイヤーと　オブザーバーの間のコールバックインターフェースを管理するようにブローカーサービスが提供されるようなシステムおよび方法に関する。

【０００２】

【従来の技術】

オブジェクト指向コンピュータプログラミング（ＯＯＰ）は、コンピュータアプリケーションにより行われる高レベルオペレーションを、オブジェクト、コンポーネントとして知られる離散的モジュールに変える。ＯＯＰの前には、コンピュータアプリケーションは通常、アプリケーションの高レベルオペレーションの際に実行される各タスクのプログラムコードを含んでいた。しかし、ＯＯＰによって、アプリケーションの高レベルオペレーションのオペレーティブ構造（フレームワーク）にとって最小限必要なプログラムコードのみを含むコンピュータアプリケーションを可能にした。コンピュータアプリケーションの高レベルオペレーションの際に行われる特定のタスクそれぞれは、必要な際にコンピュータアプリケーションのフレームワークへと「呼び出される」（コールされる）コンポーネントとして提供される。Microsoft Corporationが配布するコンポーネントオブジェクトモデル（ＣＯＭ）やObject Management Group により提唱されている顧問オブジェクトリクエストブローカーアーキティクチャー（ＣＯＲＢＡ）のようなコンポーネントベースのプログラミングプラットフォームがこのようなモジューラコンピュータアプリケーション

を設計するために開発されている。

【０００３】

一般に、オブジェクトは、「オブザーバー」または「ノーティファイヤー」、のいずれかとしてあると考えられ、これら両方はコンピュータメモリーの離散的アドレッサブル位置内に存在する。オブザーバーは、ノーティファイヤーによって所定のトリガーイベントが検出された際に何らかの所定のオペレーションを行う。例えば、通信ネットワークにおいて、オブザーバーは、自動化ボイスメールアプリケーションであることができる。これは電話呼びがボイスメールに配信されたときに記録メッセージを再生し入メッセージを記録する。ノーティファイヤーは、入呼びを検出するために電話線につながれ、その入呼びはオブザーバーに報告され、ボイスメールアプリケーションをトリガーされる。従って、所定のトリガーイベントの発生の際にノーティファイヤーコンポーネントからオブザーバーコンポーネントへとメッセージを送るシステムを提供することが望ましい。

【０００４】

トリガーイベントの発生をオブザーバーが知らされる既知の方法として、オブザーバーがノーティファイヤーを周期的にポール（poll）し、ノーティファイヤーのステータスを判断するものがある。例えば、ノーティファイヤーがメモリーフラッグを備え、これは論理「アップ」位置にあれば、ライン上の入呼びを検出し、その他の場合には論理「ダウン」位置にある。ノーティファイヤーをポールする際、フラッグが「アップ」位置にあるとオブザーバーが検出すると、それによりトリガーイベントの発生を検出し、自動化ボイスメールアプリケーションを開始する。しかしこの方法によると、オブザーバーは所定の間隔で頻繁にノーティファイヤーをポールしてフランクの位置を判断しなければならない。このようなポールの繰り返しはオペレーティング時間を相当に消費し、ノーティファイヤーがシステムから除去されるような場合などにオペレーティング問題を発生させてしまう。また、ノーティファイヤーのポールは、ほとんどの時間においては全く無意味なものである。なぜなら、フラッグのステータスは余り頻繁に変化しないからである。このように、ポーリングはシステム全体のパフォーマンスや効率を減らしてしまう。従って、ノーティファイヤーの不必要なポーリングや問い合わせを最小化したり排除するようなソフトウェアモジュールの間で情報を交換するシステムを提供することが望ましい。

【０００５】

トリガーイベントの発生をオブザーバーに通知する別の方法として、トリガーイベント発生の際にノーティファイヤーがメッセージをオブザーバーに瞬時に送るものがある。例えば、ＣＯＭのようなオブジェクトベースのフラットフォームは、オブザーバーとノーティファイヤーの間で「コールバックインターフェース」を確立し、これにより、トリガーイベントの発生の際に通知されることに関してオブザーバーの関心について前もってノーティファイヤーに対してオブザーバーがレジスターされることができる。オブザーバーは通常、以下のようにコールバックインターフェースを確立する手順を開始する。即ち、ノーティファイヤーを配置し、オブザーバーにとって関心事のトリガーイベントの発生をノーティファイヤーが検出するように適合しているかを判断するためにそのノーティファイヤーに問い合わせし、もし適合していれば、トリガーイベント発生についてオブザーバーに知らせるようにそのノーティファイヤーが適合しているかを判断する。もしそのように適合していれば、トリガーイベント発生の際に通知されるオブザーバーの関心についてオブザーバーはノーティファイヤーにレジスターする。

【０００６】

従来の方法に従ってコールバックインターフェースが確立されるプロセスは、残念ながら、オブザーバーとノーティファイヤーの間で幾つかの問い合わせやメッセージが交換されることを必要としてしまう。また、オブザーバーは、関心事のトリガーイベントをオブザーバーに対して検出するサーバー上のどこに存在するかあるいは存在するか否かを知らされない。従って、オブザーバーはシステムに存在するノーティファイヤーの通知サービスの利点を享受できるかも知れないことについて全く無知であるように動作する。これはノーティファイヤーの存在について知らないと言うことのみによって利点を享受していない

。コールバックインターフェースを確立することを望むオブザーバーに対して関心事の通知サービスを提供するノーティファイヤーを見つけるためには、関心事のイベントをオブザーバーに対してモニターするようなノーティファイヤーを見つけるためにシステム全体をオブザーバーが検索し、多くはオブザーバーに対し関心事のイベントをモニターしない、プロセスにおける全てのノーティファイヤーにオブザーバーがコンタクトすることを必要としてしまう。システム上で見つかる各ノーティファイヤーに対して検索し問い合わせすることはシステム全体のパフォーマンスや効率をかなり下げてしまうことは明らかである。従って、オブザーバーがノーティファイヤーとコールバックインターフェースを確立するのに最小限の検索や問い合わせしか必要としないような、オブザーバーとノーティファイヤーの間でコールバックインターフェースを確立するシステムの必要性がある。

【０００７】

このため、システム上のいずれのノーティファイヤーの位置およびノーティファイヤーがそれぞれモニターするイベントのクラスを含むルックアップテーブルを用いるディレクトリーサービスがシステムに提供されることがある。例えば、各ノーティファイヤーがクリエートされるに連れて、ノーティファイヤーの位置およびモニターするイベントのクラスを有する。そのノーティファイヤーに対するディレクトリーサービスルックアップテーブルにおいてエントリーがクリエートされる。特定のイベントクラスをモニターするノーティファイヤーを大きなシステムにおいて検索するよりも、オブザーバーは、ノーティファイヤーから通知を受ける関心を持った際に、ディレクトリーサービスルックアップテーブルを参照し、オブザーバーに対して関心事のイベントクラスをモニターする中にレジスターされたノーティファイヤーを見つける。例えば、上の例のラインモニターノーティファイヤーはクリエートされると、その位置を、電話ラインの未応答ステータスのモニタリングの参照とともにレジスターする。そして、ボイスメールオブザーバーは、ディレクトリーサービスの参照の際に、ラインモニターノーティファイヤーがそれに関心事のイベントクラスをモニターし、ラインモニターノーティファイヤーとコールバック関係を直接開始させることを決める。ディレクトリーサービスがノーティファイヤーに直接オブザーバーをポイントするので、オブザーバーはそれのためにシステムを検索することを必要とはされず、従って、システム全体のパフォーマンスや効率を改善することができる。しかし、オブザーバーが直接ノーティファイヤーにポイントすることが望ましくない場合もある。例えば、オブザーバーのような他のオブジェクトにノーティファイヤーの位置を知らせることは、セキュリティ問題を発生させてしまうこともある。また、ノーティファイヤーとコールバックインターフェースを確立するためにオブザーバーに対して必要とされる検索や問い合わせをディレクトリーサービスが削減できるがオブザーバーが新しいノーティファイヤーがクリエートされたかどうか判断するためにサービスにポールする必要性が発生する。

【０００８】

ディレクトリーサービスに対する代替サービスとしてイベントサービスを用いるものがある。これにおいて、以下のようにイベントサービスを通してメッセージングが配信される。即ち、ノーティファイヤーやオブザーバーがシステムにてクリエートされたりシステムから除去されるに連れて、ノーティファイヤーは除去されたオブザーバーにメッセージを送らないが、そのようにそのノーティファイヤーに指示することをできていない。また、イベントサービスを用いると、オブザーバーは所望のノーティファイヤーを見つけるためにポールする必要がなくなる。代わりに、イベントサービスがシステムにおける全てのノーティファイヤーからすべっての通知メッセージを受け、予め洗濯されたイベントクラスを有する受けメッセージについて関心をレジスターされているオブザーバーにメッセージをフォワードする。イベントサービスがシステムにてボトルネックを発生させてしまうことがよくあり、従って、システム全体のパフォーマンスや効率を下げてしまう。ストリーミングオーディオやビデオのようなイベントサービスを通過するデータが特別に密であるようなイベントサービスを用いると、システムパフォーマンスを特に減少させることができる。また、従来の構成全てはコンポーネントがサービスから除去されたりサービスに復

帰する際に対処しうるインターフェース回復手順が存在しない。

【０００９】

このように、コンポーネントおよびディレクトリーサービスやイベントサービスの潜在的に制限された形態の一部に対して必要とされる更なるアクションを必要とせずに、モジュールの間でメッセージが直接渡されるような、ソフトウェアモジュールの間で情報を交換するシステムの必要性がある。

【００１０】

【課題を解決するための手段】

本システムは、ソフトウェアモジュールの間で情報を交換するブローカー構成を備え、ノーティファイヤーは対応するイベントクラスを有する所定のトリガーイベントの発生を検出するように適合し、トリガーイベントの発生の通知を受けるのに関心があるオブザーバーを有する。本システムには、ブローカー、オブザーバー、ノーティファイヤーが含まれる。実施例において、クリエーターは、そのコンピュータのメモリーないしネットワークを通してつながれた複数のコンピュータのメモリーの離散的アドレッサブル位置にてソフトウェアモジュールをクリエートするクリエーターを用いて、ブローカー、オブザーバー、ノーティファイヤーを作る。ブローカーは、システムに存在する全てのノーティファイヤーやオブザーバーモジュールについてのアイデンティティや位置、そしてモジュールが関連するイベントクラスを備える１もしくは複数のルックアップテーブルを保持する。

【００１１】

新しいノーティファイヤーないしオブザーバーモジュールのクリエートの際、クリエーターはブローカーに対し、新しくクリエートされたことを知らせ、その新しいモジュールの位置にブローカーをポイントする。新しいモジュールにつながれると、ブローカーはモジュールに対してそれが通知サービスを提供するか（それがノーティファイヤーであるか）を判断するように問い合わせる。もしそうでなければ、ブローカーは、その新しくクリエートされたモジュールに関連して更なるアクションをしない。もしブローカーが新しくクリエートされたモジュールが通知サービスを提供することと判断すれば、ブローカーはノーティファイヤーのイベントクラスとともに、第１ルックアップテーブル（ノーティファイヤーのルックアップテーブル）にそのノーティファイヤーを加える。

【００１２】

システムに存在するオブザーバーは、そのオブザーバーがクリエーターによってクリエートされたかに関わらず、通知サービスにサブスクライブするためにブローカーを見つける。この通知サービスはシステムにおける１もしくは複数のノーティファイヤーによって提供され、オブザーバーに関心事のイベントクラスを知らせる。オブザーバーがブローカーを介してこのような通知サービスにサブスクライブすると、ブローカーはオブザーバーのイベントクラスとともに、第２ルックアップテーブル（オブザーバールックアップテーブル）にオブザーバーを加える。

【００１３】

ブローカーのルックアップテーブルにノーティファイヤーないしオブザーバーが加えられると、必ずブローカーはオブザーバールックアップテーブルとノーティファイヤールックアップテーブルを比較し、いずれかのノーティファイヤーやオブザーバー１つのイベントクラスを共有するかを判断する。ノーティファイヤーやオブザーバーがイベントクラスを共有すると、ブローカーが判断した場合には、ブローカーはトリガーイベントの発生の通知を直接オブザーバーにノーティファイヤーが送らせるように指示するメッセージをノーティファイヤーに送る。これによりブローカーはノーティファイヤーやオブザーバーの間のコールバックインターフェースを確立する補助をするが、コールバックインターフェースが確立された後のノーティファイヤーやオブザーバーの間のその後の通信には関わらない。新しいモジュールがクリエートされ、古いモジュールが除去されたり再びクリエートされるに連れて、ルックアップテーブルは、そのオブザーバーやノーティファイヤーの側に対してはいずれのアクションをも伴わずにコネクションインターフェースがクリエートされたり再びクリエートされたりするように更新される。これにより、ソフトウェアモジ

ュールのいわゆる「ホットスワッピング」およびコールインターフェースの自動復帰を可能にする。

【００１４】

本発明の一原理に従うと、ソフトウェアモジュールの間で情報を交換する本システムは、モジュールをレジスターする第１ルックアップテーブルを備えるブローカーと、モジュールをレジスターする第２ルックアップテーブルおよび対応するイベントクラスを有する１もしくは複数のトリガーイベントの発生を他のモジュールに知らせるノーティファイヤーと、対応するイベントクラスを有する１もしくは複数のトリガーイベントの通知を受けるオブザーバーとからなる。第１ノーティファイヤーは、第１イベントクラスを有する所定の第１トリガーイベントの発生を検出するように適合し、第１オブザーバーは、第２イベントクラスを有する所定のトリガーイベントの発生の通知を受信する関心がある。第１ノーティファイヤーがクリエートされるとブローカーは第１ルックアップテーブルに第１ノーティファイヤーを加え、第１オブザーバーがブローカーに対して所定のトリガーイベントの通知を受信することに関心があることを通知すると（ブローカーのサービスにサブスクライブすると）、ブローカーは第２ルックアップテーブルに第１オブザーバーを加える。ブローカーは、以下のように第１ルックアップテーブルを第２ルックアップテーブルと比較するように適合する。即ち、ブローカーによって第１イベントクラスと第２イベントクラスが同じであると検出すると、ブローカーは第１ノーティファイヤーに対して所定の第１トリガーイベントの発生を第１オブザーバーに直接通知することを指示する。

【００１５】

本発明の他の原理に従うと、本方法は、第１ルックアップテーブルと第２ルックアップテーブルを含むブローカーをクリエートし、第１イベントクラスを有する所定の第１トリガーイベントの発生を検出するように適合する第１ノーティファイヤーをクリエートする。ブローカーは第１ノーティファイヤーに対して第１ルックアップテーブルにおける第１イベントクラスをリストし、第２イベントクラスを有する所定のトリガーイベントの発生の通知を受信することに関心がある第１オブザーバーをクリエートし、ブローカーは第１オブザーバーに対して、第１オブザーバーがブローカーにその所定のトリガーイベントの発生を通知されることに関心があることを通知される場合に、第２ルックアップテーブルにおける第２イベントクラスをリストし、ブローカーは第１ルックアップテーブルを第２ルックアップテーブルと比較し、ブローカーは、第１ノーティファイヤーに対して、第１イベントクラスと第２イベントクラスが同じ場合に所定の第１トリガーイベントの発生を第１オブザーバーに直接通知するように指示する。

【００１６】

ソフトウェアモジュールの間で情報を交換するシステムおよび方法を提供することは本発明のフユチャーである。特に、ノーティファイヤーモジュールとオブザーバーモジュールの間の通信をブローカーするシステムおよび方法を提供する。

【００１７】

また、本発明は、オブザーバーによるノーティファイヤーの不必要なポールや問い合わせをなくしたようなソフトウェアモジュールの間で情報を交換するシステムや方法を提供する。

【００１８】

また、本発明は、オブザーバーモジュールとノーティファイヤーモジュールの間でコールバックインターフェースを確立するシステムを提供し、ここで、オブザーバーの側に対しての検索はノーティファイヤーとのコールバックインターフェースを確立するためにはオブザーバーにとっては必要がなくなるようにすることができる。

【００１９】

また、モジュールの間でメッセージが直接渡されるようなシステムを提供する。

【００２０】

また、堅牢性があり、ノーティファイヤーとオブザーバーの間のコールバックインターフェースの実装を容易にするシステムを提供する。

【００２１】
更に、ノーティファイヤーとオブザーバーの間の接続の回復能力が堅牢性があり、新しいノーティファイヤーモジュールがクリエートされたり現存するモジュールがサービスから除去されたり再びクリエートされた際に、コールバックインターフェースがブローカーによって自動的にクリエートされたり、再びクリエートされたり、回復されたりすることができる。
【００２２】
【発明の実施の形態】
図１において、本発明に従うソフトウェアモジュール、オブジェクト、コンポーネントなどの間の情報を交換するシステム１０を示した。システム１０は、クリエーター２０、ブローカー８０、少なくとも１つのノーティファイヤー４０、少なくとも１つのオブザーバー６０を備える。システム１０は１つのノーティファイヤー４０と１つのオブザーバー６０を有しているが、任意の数のノーティファイヤーやオブザーバーを用いてもよい。
【００２３】
ソフトウェアコンポーネント（クリエーター２０、ノーティファイヤー４０、オブザーバー６０、ブレーカー８０）は自己に含まれるモジュールとして示した。各モジュールは別々の仮想的アドレス空間に存在してもよく、あるいは全てのモジュールが共通の仮想的アドレス空間を共有したり、このような組み合わせにてモジュールを配置していてもよい。また、各モジュールは物理的に別々なコンピュータに存在しても、全てのモジュールが同じコンピュータに存在しても、そのような組み合わせで存在してもよい。モジュールがどのように分散するかに従って、モジュールは別のモジュールに以下のようにしてアクセスすることができる。即ち、直接メモリーアドレッシングによって（コールされたモジュールがコーリングモジュールとして存在するが別々の仮想的アドレス空間に存在する場合）、あるいはリモートプロシージャーコールによって（コールされたモジュールがコーリングモジュールとは別のコンピュータ上に存在する場合）である。本発明にとって必要条件ではないが、Microsoft Corporationによって開発され配布されているDistributed Component Object Model（ＤＣＯＭ）や、Common Object Request Broker Architecture（ＣＯＲＢＡ）のようなモジュールアクセスを提供する分散型オブジェクトベースプログラミングプラットフォームを用いると非常に便利である。なぜなら、コーリングモジュールがコールされるモジュールに対してその位置を知らずに、かつ、３つのアクセスモード（直接メモリーアドレッシング、プロセス間通信、リモートプロシージャーコール）の間を区別せずにアクセスすることを可能にするからである。
【００２４】
クリエーター２０は、ブローカー８０、ノーティファイヤー４０、オブザーバー６０のような他のオブジェクトをクリエートするように適合するソフトウェアコンポーネントであり、従って、何らかの高レベルオペレーションを行うようにオペレーティングシステム上にインストールされているソフトウェアアプリケーションなどによってシステム１０において固有（inherent）であることが好ましい。クリエーター２０がブローカー８０、ノーティファイヤー４０、オブザーバー６０をクリエートする責務があるので、ブローカー８０、ノーティファイヤー４０、オブザーバー６０がクリエートされる前に、クリエーター２０がシステム１０に存在する必要がある。しかし、ノーティファイヤー４０とともに他のシステム１０のノーティファイヤー、およびオブザーバー６０とともに他のシステム１０のオブザーバーは、クリエーター２０によってクリエートされる必要はなく、このようなコンポーネントをクリエートするように適合するシステム１０の他のコンポーネントによってクリエートされていてもよい。
【００２５】
一般に、システム１０はクリエーター２０によってコンストラクトされ、クリエーター２０はブロック１０１で示したようにコンピュータのメモリー内のアドレッサブル位置にてブローカー８０をクリエートする。クリエーター２０がブローカー８０をクリエートするのでブローカー８０の位置を知っておりブローカー８０の位置の記述を作る。これにより

、クリエーター２０がブローカー８０を直接後でアドレッシングすることを可能にする。例えば、ブローカー８０にメッセージや命令を送る。そして、クリエーター２０は別の位置にてブロック２０１で示すようにノーティファイヤー４０をクリエートし、メッセージ２０５をブローカー８０に送り、ブローカー８０に対してオブジェクトがクリエートされ新しくクリエートされたノーティファイヤー４０の位置に直接ブローカー８０をポイントすることを知らせ、これにより、ブローカー８０が直接ノーティファイヤー４０をアドレッシングすることを可能にする。また、ノーティファイヤー４０はノーティファイヤー４０によりモニターされる種類の情報に関連するイベントクラスを含む。その発生はそのイベントクラスに関心がありブローカーにサブスクライブしているシステム１０の任意のコンポーネントに提供される。例えば、ノーティファイヤー４０は１もしくは複数のイベントの発生をモニターし、関心あるコンポーネントによって必要とされるオペレーションを実行し、あるいは必要な情報を提供する。

【００２６】

そして、クリエーター２０はブロック３０１で示したように別の位置にオブザーバー６０をクリエートする。クリエーター２０はブローカー８０に対して更なるオブジェクトがクリエートされたことを知らせ、その新しくクリエートされたオブザーバー６０の位置にブローカー８０をポイントするメッセージ３０５を送る。これによりブローカー８０が直接オブザーバー６０をアドレッシングすることを可能にする。

【００２７】

システム１０にオブジェクトがクリエートするに従い、このようなクリエートについてブローカー８０は知らされ、ブローカー８０は各オブジェクトに問い合わせていずれかのイベントクラスの通知サービスをそれが提供するかどうかを判断する。即ち、そのクリエートされたオブジェクトがノーティファイヤーかどうかを判断する。もしブローカー８０が新しくクリエートされたオブジェクトがノーティファイヤーであると判断すれば、クリエーター２０はノーティファイヤー４０に更に問い合わせ、そこに含まれ、ブローカー８０がノーティファイヤー４０のアイデンティティとともに図２のノーティファイヤールックアップテーブル８４に加えるイベントクラスを決定する。

【００２８】

もしブローカー８０が、新しくクリエートされたオブジェクトが通知サービスを提供されないと判断すれば（新しくクリエートされたオブジェクトがオブザーバーである場合など）、その新しくクリエートされたオブジェクトに関してブローカー８０は更なるアクションを行わない。従って、ブローカー８０がノーティファイヤーではないオブジェクトがクリエートされたことを知らされること以外にオブザーバーの存在を知らされずに、オブザーバーがシステム１０にてクリエートされる。システム１０における１もしくは複数のノーティファイヤーによって提供される通知サービスにサブスクライブするためにオブザーバーがブローカー８０にコンタクトするときまで、ブローカーはオブザーバー（例えば、オブザーバー６０）に対してオブリビアス（oblivios）である状態に留まる。オブザーバー６０が通知されたいトリガーイベントのイベントクラスとともに図２のオブザーバールックアップテーブル８６にオブザーバー６０をブローカー８０が加えることを要求するメッセージ６０１をブローカー８０に送ることにより、オブザーバー６０は通知サービスにサブスクライブすると、そのオブザーバーの位置を知っているブローカー８０はノーティファイヤー４０に対してトリガーイベントの発生の通知全てをオブザーバー６０に送るように指示するメッセージ３０７を、同じイベントクラスを有するいずれかのノーティファイヤー（例えば、ノーティファイヤー４０）に送る。ノーティファイヤー４０がトリガーイベントの発生を検出すると、ノーティファイヤー４０はオブザーバー６０に直接メッセージ４０１を送る。

【００２９】

図２において、システム１０をより詳細に示してあり、通信システムを制御するコンピュータアプリケーションを示した。ただしシステム１０はコンピュータアプリケーションや通信システムの制御には限定されない。上述のように、クリエーター２０はシステム１０

にて固有（inherent）であるアドレッサブルオブジェクトであり、他のアドレッサブルオブジェクト、コンポーネント、モジュールなどのインスタンスをクリエートするように適合している。クリエーター２０は、所定の特定のオペレーションを行うように適合するクリエーター２０の１もしくは複数のサブプロセスを含む。この特定の操作には、ノーティファイヤー４０やオブザーバー６０のようなオブジェクトをクリエートするクリエーションサービス２２などがある。このクリエーションサービスは、ブローカー８０に対しオブジェクトがクリエートされた際には必ず通知するという点からイベントサービスとして機能する。またクリエーター２０はクリエーター２０がクリエートした全てのオブジェクトおよびクリエートしたオブジェクトの位置をリストするレジストリー２４を含む。

【００３０】

起動の際、クリエーター２０はブロック１０１で示したように、システム１０におけるアドレッサブル位置にてブローカー８０をクリエートし、ブローカー８０とブローカー８０の位置をレジストリー２４に加え、メッセージ１０３をレジストリー２４に送る。ブローカー８０がクリエートされるに従い、以下の３つのルックアップテーブルが定められる。即ち、ノーティファイヤールックアップテーブル８４、オブザーバールックアップテーブル８６、接続ルックアップテーブル８８である。ノーティファイヤールックアップテーブル８４はシステム１０にてクリエートされたノーティファイヤーすべておよびノーティファイヤーが報告するイベントのクラスをリストする。オブザーバールックアップテーブル８６はシステム１０におけるノーティファイヤーが提供する１もしくは複数の通知サービスにサブスクライブしているシステム１０においてクリエートされたすべてのオブザーバー、およびそれらオブザーバーが報告されることに関心があるトリガーイベントのイベントクラスをリストする。接続ルックアップテーブル８８は、同じイベントクラスを有するノーティファイヤールックアップテーブル８４、オブザーバールックアップテーブル８６にてリストされた全てのノーティファイヤーおよびオブザーバー、およびそのイベントクラスに対応するオブザーバーの位置をリストする。即ち、ブローカー８０によってセットアップ（ブローカー）された接続をリストする。

【００３１】

クリエートされるとすぐに、ブローカー８０はクリエーターに問い合わせ、そのクリエーターがノーティファイヤーでありそのイベントクラスがオブジェクトのクリエーションであることを判断する。そしてブローカー８０はノーティファイヤールックアップテーブル８４にクリエーター２０を加え、クリエーター２０に対応する「オブジェクトのクリエート」と記されたイベントクラスを参照する。このように、システム１０にてオブジェクトをクリエートしオブジェクトのクリエーションについて通知されることに関心がある全てのオブジェクトにその通知をするノーティファイヤーとしてクリエーター２０をブローカー８０は考慮する。そしてブローカー８０は自身をオブザーバールックアップテーブル８６に加え、その中の「オブジェクトのクリエート」と記されたイベントクラスを参照する。自身をオブザーバールックアップテーブル８６に加えた後、ブローカー８０はノーティファイヤールックアップテーブル８４をオブザーバールックアップテーブル８６と比較して、共通のイベントクラスのいずれかのインスタンスを見つける。「オブジェクトのクリエート」とラベルされたイベントクラスを共有するクリエーター２０とブローカー８０を見つけると、ブローカー８０はこの接続を接続ルックアップテーブル８８に加え（ブローカーし）、新しいオブジェクトがクリエートされた際に直接ブローカー８０に通知するようにクリエーター２０に指示するメッセージ１０９をクリエーター２０に送る。従って、ブローカー８０はそれ自身とクリエーター２０の間にコールバック関係を確立し、これにより、ブローカー８０は、システム１０にてクリエーター２０が新しいオブジェクトがクリエートされた際には必ず通知される。

【００３２】

図３において、クリエーター２０はブロック２０１に示したようにノーティファイヤー４０をクリエートし、所定のトリガーイベントの発生を検出し提供する。例えば、ノーティファイヤー４０は、「ラインモニター」とラベリングされ、ライン「Ｌ」のステータスを

モニターするように電話線「Ｌ」につながれ、そこに入呼びが存在する際に通知を与える。ユーザーの電話回線が未応答であった場合にそこから従来の電話交換オペレーションによって転送された後にのみ電話線「Ｌ」に入呼びが現れるようにされる。ノーティファイヤー４０のクリエートの際、クリエーションサービス２２はレジストリー２４に対しそこにノーティファイヤー４０を加えるメッセージ２０３を送り、ブローカー８０に対し、新しいオブジェクトのクリエーションを知らせ、新しくクリエートされたノーティファイヤー４０の位置に直接ブローカー８０をポイントするメッセージ２０５をブローカー８０に送る。ブローカー８０は、ノーティファイヤー４０がいずれかのイベントクラスの通知サービスを提供するかどうかを判断するメッセージ２０７をノーティファイヤー４０に送る。それが提供することを解ると、ブローカー８０は、ノーティファイヤー４０がブローカー８０にそのイベントクラスを開示することを要求する別のメッセージ２０９をノーティファイヤー４０に送る。ノーティファイヤー４０のイベントクラスを知ると、ブローカー８０はノーティファイヤールックアップテーブル８４におけるノーティファイヤー４０の次のノーティファイヤー４０のイベントクラスをリストする。この例において、ノーティファイヤー４０は「無応答」イベントクラスを含む。ブローカー８０はこれをノーティファイヤールックアップテーブル８４に加える。そして、ブローカー８０はオブザーバールックアップテーブル８６をスキャンし、そこに前にクリエートされリストされたいずれかのオブザーバーが「無応答」イベントクラスを含むかどうかを判断する。ブローカー８０が「無応答」イベントクラスを含むかどうかを判断する。ブローカー８０が「無応答」イベントクラスを有するオブザーバーを見つけることができないと、システム１０におけるオブジェクトからの更なる通信を待つ。

【００３３】

図４において、クリエーター２０は次に３０１で示したようにオブザーバー６０をクリエートする。これは例えば、「メッセージ記録」とラベリングすることができ、これは所定のトリガーイベントの発生について通知されることに関心がある。例えば、オブザーバー６０は、電話線「Ｌ」上に入呼びが存在することについて知らされることに関心があり、入電話呼びが配信された際に記録メッセージを再生したり入メッセージを記録したりする機能を行う自動化ボイスメール応答アプリケーションを開始させる。

【００３４】

オブザーバー６０のクリエートの際、クリエーションサービス２２はレジストリー２４に対し、それにオブザーバー６０を加えるメッセージ３０３を送り、ブローカー８０に対しシステム１０における新しいオブジェクトのクリエートを知らせ、新しくクリエートされたオブザーバー６０の位置に直接ブローカー８０をポイントするメッセージ３０３をブローカー８０に送る。次にブローカー８０はそのオブザーバー６０に、オブザーバー６０が通知サービスを提供するかどうかを判断するメッセージ３０７を送る。提供しないことを解ると、ブローカー８０はブローカー８０に関連して更なるアクションを行わず、システム１０におけるオブジェクトからの更なる通信を待つ。もし問い合わせの際にブローカー８０がオブザーバー６０はノーティファイヤーであり、通知サービスを提供すると判断すれば、ブローカー８０は上述のように進行する。

【００３５】

イベントクラスを共有するノーティファイヤーとオブザーバーの間の直接接続はブローカー８０によって折衝される。このブローカー８０は、システム１０に存在する全てのノーティファイヤーおよびそれらの対応するイベントクラスについて知っている。このブローカー８０は、オブザーバー６０およびそのイベントクラスについて知るようにはならず、重要なことに、所望のイベントクラスを有する通知サービスに加入するようにオブザーバー６０がブローカー８０にコンタクトするまで、オブザーバー６０およびそのイベントクラスをオブザーバールックアップテーブル８６に加えない。従って、オブザーバーはブローカー８０との通信を配置し開始するように適合しなければならない。

【００３６】

所与のイベントクラスを有する所定のトリガーイベントの発生についての通知を受けるこ

とにオブザーバー６０が関心を持つと、そのオブザーバー６０はブローカー８０を配置し予め選択されたイベントクラスを有する通知サービスに加入するメッセージ３０９をブローカー８０に送る。このことは単にブローカー８０がオブザーバー６０をオブザーバー６０のイベントクラスとともにオブザーバールックアップテーブル８６に加えることによって実現できる。これは、オブザーバー６０によってメッセージ３０９においてブローカー８０に開示される。

【００３７】

実施例において、オブザーバー６０は「無応答」イベントクラスを有するノーティファイヤーが提供する通知サービスにオブザーバー６０が加入する。ブローカー８０は、オブザーバー６０をオブザーバー６０の「無応答」イベントクラスへの参照とともにオブザーバールックアップテーブル８６に加える。そしてブローカー８０はノーティファイヤールックアップテーブル８４をスキャンして、そこに前にクリエートされリストされたいずれかのノーティファイヤーがマッチングする「無応答」イベントクラスを含むかどうかを判断する。そこにリストされたノーティファイヤー４０が「無応答」イベントクラスを有するものとして見つけると、ブローカー８０はオブザーバールックアップテーブル８６にノーティファイヤー４０－オブザーバー６０接続を加え、オブザーバー６０の位置を記録する。ブローカー８０はクリエーター２０によって開示されるに従ってノーティファイヤー４０の位置を知り、ブローカー８０はノーティファイヤー４０に対してオブザーバーの知っている位置にてオブザーバー６０へとトリガーイベントの発生の直接通知を送るようにノーティファイヤー４０に指示するメッセージ３０８をノーティファイヤー４０に送る。これにより、ブローカー８０はいずれかのコンポーネントが他方を検索する必要なしでノーティファイヤー４０とオブザーバー６０の間に直接のコールバック関係を確立する。ブローカー８０はこのようにしてノーティファイヤー４０とオブザーバー６０の間にコールバック関係を確立し、ノーティファイヤー４０とオブザーバー６０の間の更なるメッセージングは参加しない。

【００３８】

図５において、ノーティファイヤー４０（ここではオブザーバー６０に直接ポイントしている）は、所定のトリガーイベント４０３の発生を待つ。即ち、電話線「Ｌ」上の入呼びの存在がノーティファイヤー４０によってモニターされる。ノーティファイヤー４０がイベント４０３を検出すると、ノーティファイヤー４０はオブザーバー６０に対しボイスメールに電話呼びが配信された際に記録メッセージを再生しメッセージを記録するような処理のメッセージ４０１を送る。

【００３９】

図６において、システム１０において更なるオブジェクトがクリエートされており、これらは上述のようにシステム１０にて存在するノーティファイヤーとオブザーバーの間で前にブローカーされたコールバック関係に自動的にインディグレートされる。例えば、クリエーター２０はブロック５０１で示すように「点灯」とラベリングした新しいオブザーバー７０をクリエートする。これは、「メッセージ記録」オブザーバー６０と同様に、電話線「Ｌ」上で入呼びが存在する際に通知されることに関心があって、例えば、被呼ユーザーの電話番号上で「呼び待ち」ライトを点灯する。新しいオブザーバー７０のクリエートの際、クリエーションサービス２２は、クリエートされたオブジェクトレジストリー２４に対し、そこに新しいオブザーバー７０を加えるメッセージ５０３を送り、ブローカー８０に対しその新しいオブジェクトのクリエートを知らせ、その新しくクリエートされたオブザーバー７０の位置にブローカー８０をポイントするメッセージ５０５を送る。次にブローカー８０はその新しくクリエートされたオブジェクト（オブザーバー７０）に対し、その新しいオブジェクトが通知サービスを提供するかどうか判断するメッセージ５０７を送る。通知しないことが解ると、図のように、ブローカーはオブザーバー７０に関連して更なるアクションを行わない。

【００４０】

所与のイベントクラスを有する所定のトリガーイベントの発生の通知を受けることにオブ

ザーバー７０が関心を持つと、オブザーバー７０はブローカー８０を配置しブローカー８０に対してその所与のイベントクラスを有する通知サービスに加入するメッセージ５０９を送る。このことは単に、オブザーバー７０をオブザーバー７０のイベントクラスとともにオブザーバールックアップテーブル８６に加えるブローカー８０によって実現できる。これはオブザーバー７０によってメッセージ５０９にてブローカー８０へと開示される。図示したように、電話線「Ｌ」上に入呼びを受けた際に通知されて被呼ユーザーの電話上で「呼び待ち」ライトが点灯されることに関心がある。そしてオブザーバー７０は「無応答」イベントクラスを有する通知サービスにサブスクライブすることを望む。この情報はオブザーバールックアップテーブル８６に加えられる。

【００４１】

そしてブローカー８０はノーティファイヤールックアップテーブル８４をスキャンして、そこに前にクリエートされリフトされたいずれかのノーティファイヤーがマッチングする「無応答」イベントクラスを含むかどうかを判断する。含むと判断すると、ブローカー８０は接続ルックアップテーブル８８にノーティファイヤー４０－オブザーバー７０接続を加え、オブザーバー７０の位置を記録する。ブローカー８０には、クリエーター２０によって与えられるようにノーティファイヤー４０の位置が知らされ、ブローカー８０はノーティファイヤー４０に対しオブザーバー７０の知られている位置にて直接トリガーイベントの発生の通知をオブザーバー７０に送ることを指示するメッセージ５０８をノーティファイヤー４０に送る。これによりブローカー８０はいずれかのコンポーネントがお互い検索する必要なしで、また、ノーティファイヤー４０と現存する「メッセージ記録」オブザーバー６０の間の現存するコールバック関係を破壊することなしに、ノーティファイヤー４０とオブザーバー７０の間の直接のコールバック関係を確立される。このようにして、ノーティファイヤー４０とオブザーバー７０の間でコールバック関係が確立され、ブローカー８０はそれらの間の更なるメッセージングには参加しない。

【００４２】

更に図７において、ノーティファイヤー４０（ここではオブザーバー６０と新しいオブザーバー７０の両方に直接ポイントされている）は、所定のトリガーイベント４０３（例えば、電話線「Ｌ」上で入呼びが受けられること）の発生を待つ。電話線「Ｌ」上で入呼びが存在することをノーティファイヤー４０が検出すると、ノーティファイヤー４０はオブザーバー６０に対し、例えば、記録メッセージを発生し入メッセージを記録することの処理のメッセージ４０１を送り、また、オブザーバー７０に対し、例えば、被呼ユーザーの電話機上のライトをアクティベートするような処理のメッセージ５０５を送る。

【００４３】

オブザーバーはノーティファイヤーと直接コールバックインターフェースを折衝しないので、システムからノーティファイヤーを除いて、後にそのオブザーバーの知識なしで再びクリエートできる。これには例えば、ノーティファイヤーの故障や変更によるものがある。過去においてオブザーバーが１もしくは複数のノーティファイヤーとのコールバックインターフェースをコンタクトして折衝して、ノーティファイヤーからメッセージを受ける必要があったが、本発明に従うと、このような直接の折衝は行う必要性がなくなる。なぜなら、ブローカー８０によってノーティファイヤーとオブザーバーの間のコンタクトがセットアップされるからである。このことの利点の１つとして、ソフトウェアコンポーネント（実施例において、通信システムのノーティファイヤーとオブザーバー）のいわゆる「ホットスワッピング」可能になることがある。これらのコンポーネントはコールバックインターフェースを折衝するのにお互いを直接アドレッシングする必要がなく、ブローカー８０に対してそれらの存在を知らせるのみを必要とするからである。ノーティファイヤーの故障などによってシステム１０からノーティファイヤーがのぞかれると、それに対応するトリガーイベントの発生のモニターが中止され、オブザーバーにメッセージを送ることも中止される。オブザーバーは到達しない通知メッセージを待つことになるが、そのノーティファイヤーの再クリエートには、ノーティファイヤーとのコールバックインターフェースをオブザーバーが折衝する必要はない。

【００４４】
代わりに、新しいノーティファイヤーがクリエートされると、そのイベントクラスがブローカー８０のルックアップテーブルノーティファイヤールックアップテーブル８４、オブザーバールックアップテーブル８６、接続ルックアップテーブル８８に入り、オブザーバーへの接続が上述のようにオブザーバーの知識なしでブローカー８０によって動的に再クリエートされる。ブローカー８０は単にその再クリエートされたノーティファイヤーに対して適切なオブザーバーに通知メッセージを送ることをそのノーティファイヤーに指示するメッセージを送る。このようにシステム１０はノーティファイヤーの故障やサービスからの除去に対して自動的な回復を提供し、全ての適切なオブザーバーとのコールバックインターフェースについてその回復されるノーティファイヤーの折衝をなくすことを可能にする。

【００４５】
そしてシステム１０からノーティファイヤーを意図的に除去することができ、この場合、ノーティファイヤー４０はブローカー８０に対してノーティファイヤールックアップテーブル８４、オブザーバールックアップテーブル８６、接続ルックアップテーブル８８からそのノーティファイヤー４０を指示するように指示するメッセージをブローカー８０に送る。そのノーティファイヤーの再クリエートによって、上述のように、全ての関心のあるオブザーバーとのコールバックインターフェースを再び確立する。ノーティファイヤーが故障することによっても全ての壊滅的な故障や予期しない故障に対してクリエーターがこのようなパージング（purging）オペレーションを行うことが好ましい。同様に、オブザーバーは、ブローカー８０に対し、それらが除去された際に知らされるべきである。これには意図的な場合や故障に導く継続した劣化である場合がある。

【００４６】
システム１０からオブザーバーを意図的に除去することができる。この場合、オブザーバー６０（ないし「デクリエート」サービスを用いるクリエーター）はブローカー８０に対してノーティファイヤールックアップテーブル８４、オブザーバールックアップテーブル８６、接続ルックアップテーブル８８からオブザーバーを除去することを指示するメッセージを送る。ノーティファイヤールックアップテーブル８４、オブザーバールックアップテーブル８６、接続ルックアップテーブル８８をスキャンすると、ブローカー８０は、除去されたオブザーバーがいずれかのノーティファイヤーとの接続を形成するかを判断し、適切なノーティファイヤーに対して除去されたオブザーバーへの通知メッセージを送ることを中止することを指示するメッセージを送る。この方法により、前に加入していた通知サービスに対しオブザーバーは脱退する（unsubscribe）。オブザーバーの再クリエートによってノーティファイヤールックアップテーブル８４、オブザーバールックアップテーブル８６、接続ルックアップテーブル８８を更新し、これにより、対応するノーティファイヤーをその再クリエートされたオブザーバーの新しい位置にポイントする。オブザーバーの状態を問い合わせるようにノーティファイヤーを適合することができ、これにより、前にオブザーバーによって受けられていない全てのトリガーイベントがその再クリエートの際にそれに送られる。システム１０からのオブザーバーの意図しない除去（例えば、オブザーバーの故障によるもの）がオブザーバーの再クリエートの際に訂正され、この際にて、ルックアップテーブルが更新され、全てのノーティファイヤーが再クリエートされたオブザーバーの新しい位置に再び勧められる。また、新しくクリエートされたオブザーバーはそれらのクリエート時間までの全てのイベントについて知らされ、システム１０の現在のステータスについて知ることができるようにされる。このように、システム１０にてコンポーネントが除去されたり再インストールされたりする際には必ず、システム１０はソフトウェアはソフトウェアコンポーネントの動的回復および「ホットスワッピング」機能を与え、接続インターフェースの自動回復を与える。

【００４７】
ブローカー８０がオーディット機能を提供するようにすることができる。これにより、ブローカー８０は各ノーティファイヤーおよび各オブザーバーを周期的に問い合わせ、ノー

ティファイヤールックアップテーブル８４、オブザーバールックアップテーブル８６、接続ルックアップテーブル８８にてその位置が正しくリストされているかどうかを判断する。ノーティファイヤールックアップテーブル８４、オブザーバールックアップテーブル８６、接続ルックアップテーブル８８にリストされたコンポーネントがリストされた位置にもう存在しないとブローカー８０が判断すると、ブローカー８０はそのコンポーネントを対応するルックアップテーブルから自動的に除去し、その消滅（extinct）コンポーネントに関連するいずれのコンポーネントに対しても同様に指示する。例えば、クエリーによってオブザーバーがブローカー８０に対して知らせずにその位置から除去されたとブローカー８０が判断すれば、ブローカー８０はノーティファイヤールックアップテーブル８４、オブザーバールックアップテーブル８６、接続ルックアップテーブル８８からそのオブザーバーへのいずれの参照をも除去し、接続ルックアップテーブル８８にオブザーバーとの接続がリストされていたいずれのノーティファイヤーに対してもその除去されたオブザーバーへのメッセージを中止させるメッセージを送る。

【００４８】

上の説明は従来のメモリーアドレッシング技術を用いてメッセージをお互い直接送るコンポーネントに関連して説明したが、ノーティファイヤーとオブザーバーの間でコールバックインターフェースを確立するようないわゆる接続ポイントを用いるようなＣＯＭ、ＤＣＯＭ、ＣＯＲＢＡなどの従来のコンポーネントベースのプラットフォームを用いても実現することができる。接続ベースのプラットフォームでは、ブローカー８０は接続ポイントマネージャーと呼ぶことができる。

【図面の簡単な説明】

【図１】本発明の好ましい実施例に従い、ソフトウェアコンポーネントの間で情報を交換するシステムの図。

【図２】ブローカーをクリエートする第１ステップを示す図１のシステムの図。

【図３】ノーティファイヤーをクリエートする第２のステップを示す図１のシステムの図。

【図４】オブザーバーをクリエートするステップを示す図１のシステムの図。

【図５】オブザーバーを通知するステップを示す図１のシステムの図

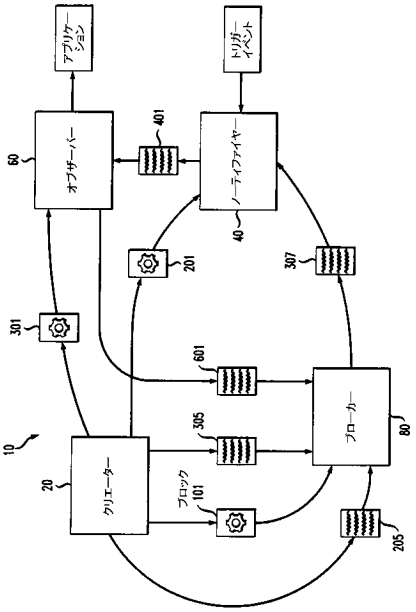
【図６】更なるオブザーバーをクリエートするステップを示す図１のシステムの図。

【図７】オブザーバーおよび更なるオブザーバーを通知するステップを示す図１のシステムの図。

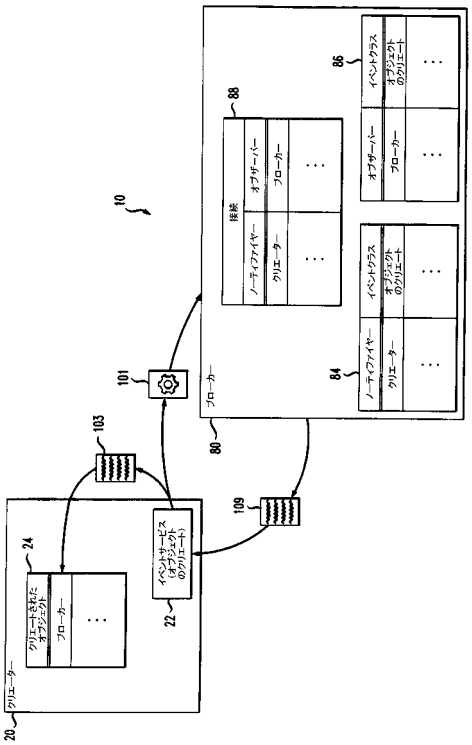
【符号の説明】

１０　システム
２０　クリエーター
２２　クリエーションサービス
２４　レジストリー
４０　ノーティファイヤー
６０、７０　オブザーバー
８０　ブローカー
８４　ノーティファイヤールックアップテーブル
８６　オブザーバールックアップテーブル
８８　接続ルックアップテーブル
１０１、２０１、３０１、５０１　ブロック
１０３、２０３、２０５、２０７、２０９、３０３、３０５、３０７、３０８、３０９、４０１、５０３、５０５、５０８、６０１　メッセージ
４０３　トリガーイベント

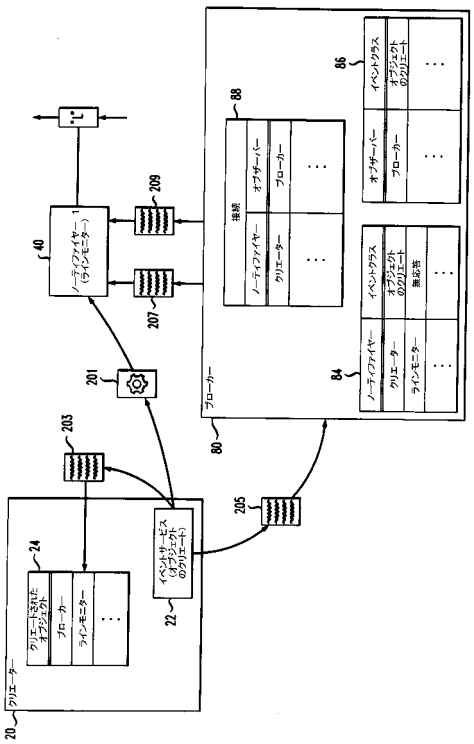
【図１】


アプリケーション
トリガーイベント
オブザーバー
ノーティファイヤー
クリエーター
ブローカー
ブロック
101
10
20
40
60
80
201
205
301
305
307
401
601

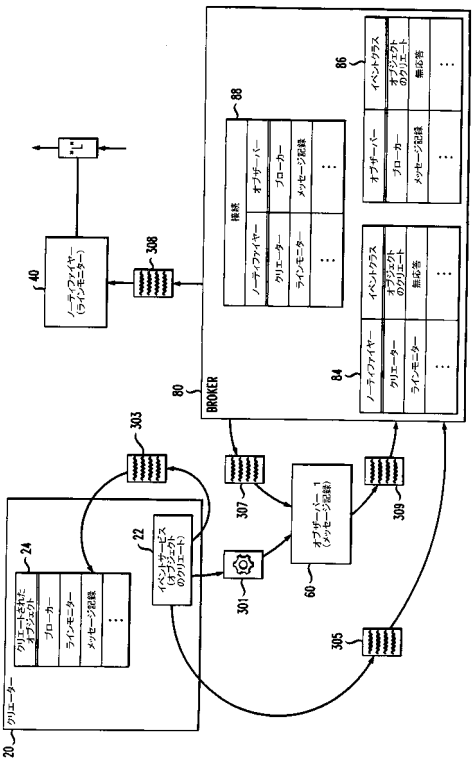
【図２】


10
20
80
84
86
88
101
103
109
ブローカー
クリエーター
ノーティファイヤー
イベントクラス
オブジェクトのクリエート
イベントサービス（オブジェクトのクリエート）
クリエートされたオブジェクト
接続
オブザーバー
クリエーター
ノーティファイヤー
ブローカー
22
24

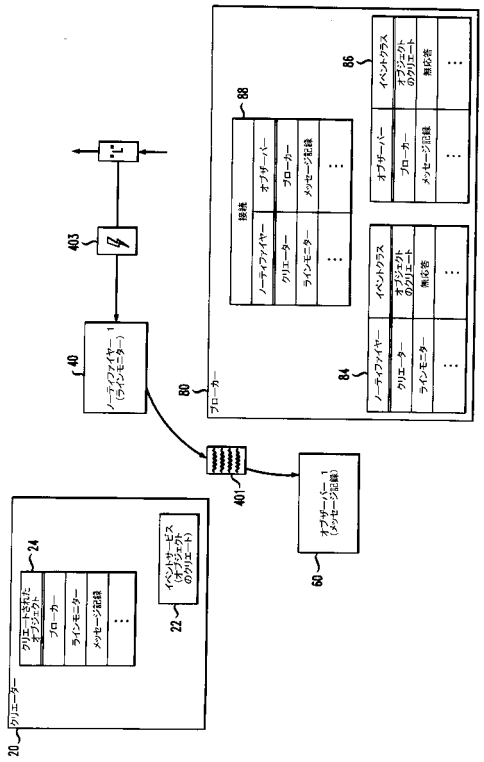
【図３】


20
22
24
40
80
84
86
88
201
203
205
207
209
ブローカー
クリエーター
ノーティファイヤー（ラインモニター）
イベントクラス
オブジェクトのクリエート
無応答
イベントサービス（オブジェクトのクリエート）
クリエートされたオブジェクト
接続
オブザーバー
クリエーター
ノーティファイヤー
ブローカー
ラインモニター

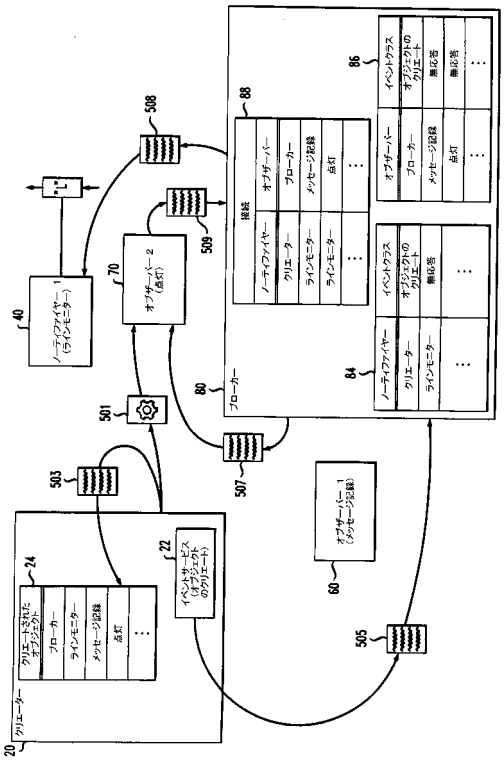
【図４】


20
22
24
40
60
80
84
86
88
301
303
305
307
308
309
BROKER
クリエーター
ノーティファイヤー（ラインモニター）
イベントクラス
オブジェクトのクリエート
無応答
メッセージ記録
イベントサービス（オブジェクトのクリエート）
クリエートされたオブジェクト
接続
オブザーバー
クリエーター
ノーティファイヤー
ブローカー
ラインモニター
オブザーバー（メッセージ記録）

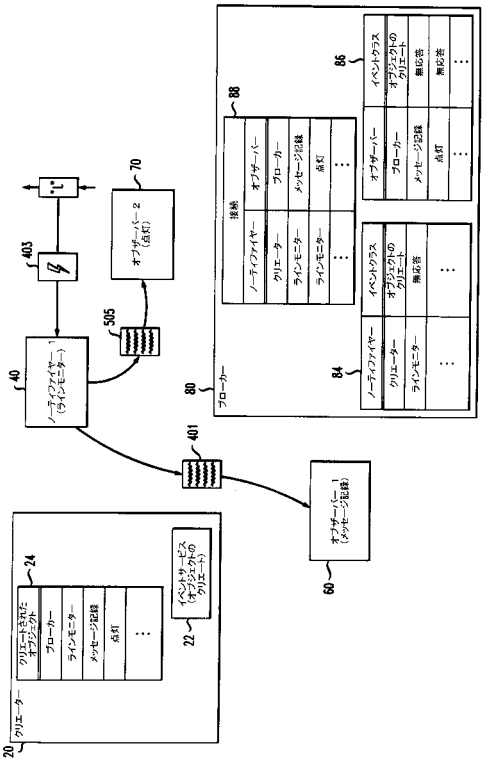
【図５】



【図６】



【図７】



フロントページの続き

(74)代理人 100091889
弁理士 藤野 育男
(74)代理人 100101498
弁理士 越智 隆夫
(74)代理人 100096688
弁理士 本宮 照久
(74)代理人 100102808
弁理士 高梨 憲通
(74)代理人 100104352
弁理士 朝日 伸光
(74)代理人 100107401
弁理士 高橋 誠一郎
(74)代理人 100106183
弁理士 吉澤 弘司
(74)代理人 100081053
弁理士 三俣 弘文
(72)発明者 マーク エー. コーエン
アメリカ合衆国、４３０６５ オハイオ、パウエル、ウッドランド ドライブ ２０８
(72)発明者 ジョン デビッド ロング
アメリカ合衆国、４３０８５ オハイオ、ワシントン、クロベンストン ドライブ １４６４

審査官 間野 裕一

(56)参考文献 特開昭６３－２２８３３５（ＪＰ，Ａ）
特開平０８－１２９５３６（ＪＰ，Ａ）
特表平１０－５００５０９（ＪＰ，Ａ）
西岡利博他，オブジェクトの交換を利用した分散サービス利用のためのフレームワーク，情報処理学会研究報告，社団法人情報処理学会，１９９６年 ８月２７日，第９６巻，第７９号，第４９－５４頁，９６－ＯＳ－７３－９

(58)調査した分野(Int.Cl.，ＤＢ名)
G06F 9/46-9/54
G06F 9/44