5Z－1

システム構成の柔軟性と高いパフォーマンスを同時に提供するオペレーティングシステム

小林 良岳　　前川 守

電気通信大学大学院 情報システム学研究科

1　はじめに

近年、技術の発達によって64ビットアドレス空間が使用可能になり、今までにはない広大なアドレス空間を対象としたオペレーティングシステムの構成が可能になった。また、オペレーティングシステムには、マルチメディア処理などの多様な要求に対応するため、従来のような固定されたポリシだけでなく、実行中に柔軟にシステムの構成を変更でき、さらに十分なパフォーマンス及び応答性能を引き出すことができる枠組が必要となっている。

本稿では、システムの構成を柔軟に変更できることを可能としつつ、多様な要求に十分なパフォーマンスを持って対応できるシステムを、64ビットのアドレス空間を有効に利用することで目指す。そこで、マイクロカーネルをベースにクライアントとサーバで構成されるシステムにおいて、サーバのコードを各クライアントのアドレス空間において共有し、クライアントのコンテキスト内で実行する枠組を提案する。

2　効率の良いサービスの実現

要求するサービスの効率を考えた時、カーネルをコールしたときの作業内容について考慮する必要がある。カーネルが行なう作業が誤りチェックなどのカーネルレベルで行う必要がないものであり、エラーが検出された時点で処理が終了するのであれば、カーネルへの動作モードの切替えは無駄になる。そこで、要求するサービスをどこで実現するかが問題となる。

Machで提供されているようなIPCによるサーバとクライアント間の通信は、その柔軟性を追求した結果、場合によっては必要としないようなチェックまでカーネルで行なうことがあり、効率が良いとはいえない。そこで、IPCの機構をよりカスタマイズすることによって効率をあげるということが考えられる。Kea [1] では、現在実行されているタスクが属するドメインから、他のドメインへのドメイン間コールを実現することにより、サービスに対しての効率の良い呼び出しを提供している。しかし、これら場合もカーネルをコールしチェックを行うオーバヘッド及びアドレス空間の切替えのオーバヘッドが生じる。

SPIN [2] や VINO [3] などのシステムでは、カーネル空間内にコードを取り込んで実行することによって、システムに柔軟に拡張を行うことが可能である。しかし、前述したような無駄な動作モードの切り替えが起こる可能性がある。

本稿で提唱するシステムではクライアントのアドレス空間においてサーバのコードを共有し、カーネルの介在なしにクライアントからサーバを手続き呼び出しのような手順で直接呼び出すことを可能にする。このことにより、メッセージ通信によるカーネルでのチェックをなくし、さらにサーバがカーネルをコールする割合が低い場合、効率の良いサービスが実現できると考えられる。

3　システムの概略

図1にシステムの概略を示す。図中右側のクライアントは、カーネルに対してサーバへの接続要求をしている模様をあらわし、左側のクライアントはサーバを呼び出している模様をあらわしている。

3.1　タスクとスレッド

タスクは、同一資源を共有する単位として考える。タスクは、複数のスレッドを含み、さらに1つ以上のサーバのコード及びデータが属することができる64ビットのアドレス空間を固有に保持している。タスクのアドレス空間には、あらかじめサーバの属する領域として広大な空間が確保されており、サーバのコード及びデータは、この領域内に順次配置されていく。サーバの配置はカーネルによって全てのタスクに対して一意に決定され、サーバの領域は全てのタスクで

An Operating System with High Flexibility and High Performance

Yoshitake Kobayashi and Mamoru Maekawa
Graduate School of Information Systems, University of Electro-Communications
1-5-1 Chofugaoka, Chofu-shi, Tokyo, 182, Japan

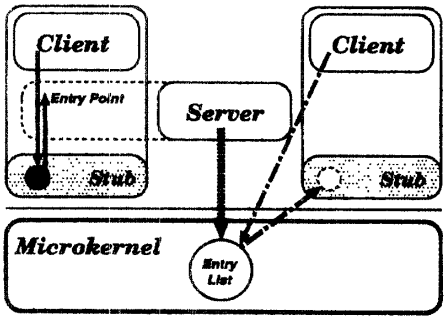

図 1: システムの概略

共有することが可能である。また、一度使われた空間は、サーバが終了した後でも使われることはない。

サーバとクライアント間でメッセージを渡す際、その空間はクライアント側のローカルな空間に作成されサーバと共有される。これにより、他のクライアントからの不正な読み出しからメッセージは保護される。

スレッドは、タスク内における１つの実行単位である。もし、あるスレッドがサーバを呼び出すと、スレッドのコンテクストを用いてサーバが実行される。

3.2　サーバのエントリの登録

サーバとして動作するタスクは、実行時にカーネルに対して自分自身が提供するサービスのエントリをカーネルに登録する。エントリには、エントリ名、アクセス権限が含まれる。サーバへのエントリは、１つであっても複数であっても良い。後に、カーネルはクライアントからのアクセス要求があった場合、この情報を基にアクセス権をチェックする。

3.3　サーバの呼び出し

まず、クライアントとして動作するタスクが実行されると、クライアント側から参照する名前及びサーバのエントリ名を指定し、サーバへの接続要求を行う。これを行わない限り、サーバが提供しているサービスを受けることはできない。

クライアント内には、サーバを呼び出すためのスタブが用意され、ここにカーネルはサーバに対するエントリを登録する。クライアントから接続要求があると、カーネルはまずアクセス権を調べ、チェックに成功するとサーバへのエントリがクライアントのスタブに登録される。一度チェックが行われ接続が成功すると、それ以降の呼び出しではアクセス権のチェックは行われず、スタブ内に登録されたエントリを利用した効率の良い呼び出しが可能となる。

サーバが呼び出されるとロックが行われ、他のクライアントからの呼び出しに対してブロックする。この、排他制御のポリシはサーバの実装に依存し、カーネルは介在しない。後にサーバのコードから復帰するときにロックは解除され、実行待ちのスレッドが存在した場合、再度スケジューリングが行なわれる。

3.4　サーバの置換

ユーザからの要求が変更されると、それに応じてサーバの置換が必要となる場合がある。この時、カーネルはサーバを置換する方法を提供する。カーネルは置換するサーバのエントリが、現在どのクライアントに対して登録されているかを調べ、それに対して全てあるいは選択的に接続を変更する。ただし、サーバが使用中である場合は置換を中断し、後に未使用状態の時に置換する。

4　まとめ

本稿では、マイクロカーネルをベースにしたシステムにおいて、広大なアドレス空間を前提とし、サーバのコードをクライアントの空間へ導入することを考えた。そして、クライアントからサーバへのコールを、すべてクライアントのコンテクストを用いた手続き呼び出しとすることで、効率の良い呼び出しを実現する方法について述べた。

今後の課題としては、カーネルをコールする割合について考え、ユーザ空間とカーネル空間のどちらかをユーザが選択してサーバのコードを実行できる統一されたインターフェイスを提供する必要があると考えられる。

参考文献

[1] A. C. Veitch and N. C. Hutchinson. Kea – A Dynamically Extensible and Configurable Operating System Kernel. In *Proceedings of the Third Conference on Configurable Distributed Systems*, 1996.

[2] B. Bershad et al. Extensibility, Safety and Performance in the SPIN Operating System. In *Proceedings of the 15th ACM Symposium on Operating System Principles (SOSP-15)*, pp. 267–284, 1996.

[3] C. Small and M. Seltzer. Structuring the kernel as a toolkit of extensible, reusable components. In *Proceedings of the 1995 International Workshop on Object Orientation in Operating Systems (IWOOOS '95)*, 1995.