LKM を用いたOSの動的適応機構に関する考察

5Q－2

杉野 洋一[1]　追川 修一[2]　徳田 英幸[1]

慶應義塾大学 環境情報学部[1]　カーネギメロン大学 計算機科学学部[2]

1　はじめに

小型のハードウェアが登場し，これに伴って，アプリケーション及び利用者の要求も変化した．一つは移動計算機環境への適応に対する要求であり，一つはNC(Network Computer)やPDA(Personal Data Assistant)，情報家電といった基盤ハードウェアの多様化への適応に対する要求である．しかし，これらの要求に応える為に複数の要求を予めOS内に保持する拡張は，OSの肥大化を招く．肥大化による拡張は，計算機の多様化の一例である組み込み型システムや，移動環境に適した携帯用小型計算機には適さない．移動計算機環境では，OSも動的に利用者の要求の変化に対応する必要がある．

LKM(Loadable Kernel Module)とは、脱着可能なカーネルの一部分であり，カーネルにロードされている時は主記憶に常駐し，アンロードされている時は主記憶から排除される．本稿では，LKM(Loadable Kernel Module)[1]を実際に適用してRT-Mach[2]を稼働させて，その有効性について述べる．

2　動的適応の必要性

動的適応とは計算機を停止しないで，利用者の要求の変化に対応したサービスを提供可能とする事である．計算機の小型化に伴って，計算機の携帯が可能となった．この様な計算機を利用する環境では，利用者の要求も動的に変化する．例えば，情報通信の帯域や電源管理の方針等が計算機の稼働中に変化する．PCMCIAカードはハードウェアで動的適応機能を提供する一例である．動的適応機能の実現の為には，ハードウェアだけでなくOSにおいてもこの機能が実装されている必要がある．

従来のモノリシックカーネルのOSでは必要となる機能は，予め全てOSの中に組み込まれていた．計算機の移動による利用者の要求の変化は，要求される全ての機能をカーネルが保持する事によって対応していた．豊富な主記憶，大容量の二次記憶，高速なCPU等，充分な資源のある計算機では，この様な，OSが全ての機能を内部的に保持する構造も可能である．しかし，資源が制限されている計算機では，要求の変化に対する全ての機能を保持する事は難しく，主記憶・二次記憶などの制限から不可能な事さえある．

又，要求される全ての機能をカーネルが保持する事は，OSを複雑化させ保守・拡張作業を困難にするなどの欠点がある．

移動計算機環境では，OSの提供するサービスも動的に変更可能である必要がある．動的変更を可能とし，同時にカーネルの主記憶占有量を制限する為には，OSにおいても，利用者の要求に従って必要なものを取り込み，不必要なものを取り外す動的再構成可能な機構が求められる．

3　静的再構成の必要性

静的再構成可能であるとは，OSの開発及び作成時に，特定の用途に適した任意の機能を容易に組み込む事が可能な事である．ハードウエア基盤の多様化により多数のOSを開発する必要が生じる．従って，これらのOSを効率良く開発する為の枠組が必要である．

ハードウェアの小型化は携帯可能な計算機と共に，多様なハードウェア基盤を作り出す．例えばIEEE1394[3]で想定されているように家電製品にもネットワーク機能が付加され，他の計算機と通信できるようになる．また，PDAの様な特殊なデバイスを持った計算機も開発されている．これらのシステム上では資源の制約からOSを利用しない事も考えられるが，効率的なプログラム開発を行う為には，OSによる資源の抽象化が必要である．

従来，OSの開発に於いてはそのフルセットを開発していたが，ブートストラップローダ，カーネルイメージのメモリ貼り付け等をOS開発の度に新規に作成する事は非効率である．これら物理機構依存部分も，LKMとして実装する事によりOSの開発時にその全てを開発する手間が省け，効率的なOS開発が可能となる[4]．

更に，資源に制限のある計算機では，タスク切り替えやアドレス空間の切り替えなどの負荷を避ける為に，従来，アプリケーションとして実装されて来たプログラムの一部が，OSと未分化な形で組み込まれる事が考えられる．これらのOSに於いてもアプリケーションの切り替えを実現する為には，静的再構成可能な機構が必要である．

4　再構成可能機構としてのLKMの特徴

LKMの特徴として次の2点が挙げられる．

1. ロードされている時は主記憶に常駐する．
2. アンロードされると主記憶から排除される．

1の特徴は静的再構成機構としてLKMが有用である事を示している．主記憶に常駐する事により，カーネルも構造化する事ができる．構造化の手法として，サーバによってサービスを切替えるもの(SS:Service by Server)[5]，ライブラリを利用するもの(SL:Service by Library)[6]などがある．これらはOSの機能の一部をサーバやライブラリとしてカーネルの外部に出しているが，LKMはこれらのOSのカーネルを更に部品化する事ができ，静的再構成を可能とする．

2の特徴は動的適応機構としてLKMが有効である事を示している．LKMはアンロードされると主記憶から排除される為，主記憶の容量に制限のある計算機でも利用する事ができる．

動的適応，静的再構成可能である為にはOSが構造化されている必要がある．OSの構造化の方法として，S-L，SS以外に，カーネル内に部品を追加・削除するもの(SK:Service by Kernel)がある．OSが構造化されていることにより，任意のサービスを組み込んだり外したりする事が可能となる．

SSではサーバはアプリケーションと同じ扱いなので高い安全性を確保する事ができる．反面，カーネル・サーバ・アプリケーションという構造上，アドレス空間の切替えやシステムコール等のオーバーヘッドが大きく，実行速度に問題が残る．

SLではOSが提供する機能の一部がライブラリとしてアプリケーションに付加されているので，実行速度に問題は無い．しかし，一定の主記憶常駐部は残り，カーネルが提供するサービスは固定されてしまう．これは静的再構成には適さない．

"Dynamic Adaptation using LKM"

Youichi Sugino, Oikawa Shuichi, Hideyuki Tokuda

[1]Faculty of Environmental Infromation, Keio University, 5322, Endo, Fujisawa-shi, Kanagawa, 252 Javan. [2]School of Computer Science, Carnegie Mellon University, Pittsburgh, PA 15213, USA.

	Size(bytes)
Deadline Monotonic	3477
Early Deadline First	3918
Fixed Priority	4806
Rate Monotonic	3813
Round Robin	2566

表 1 各モジュールの大きさ

SK は主記憶上に常駐すると言う性格上，実行速度に問題は無いが，主記憶を占有するという問題は残る．従来型の SK では動的適応をする為にはカーネルが全ての機能を保持していなければならない．

この SK に於ける問題を解決するのが LKM である．LKM を利用する事でカーネルを部品化し，脱着可能とする事ができる．LKM 化した場合，任意の部品の信頼性を特定する事はできない [7]．従って，信頼性に関しては，対象となるシステムの特性に合わせて別の機構が必要である．

LKM は SL，SS 等の API(Apprication Program Interface) とは別に SPI(System Program Interface) を持つ OS にとっても重要である．これらの OS では，アプリケーションに提供される API は，ライブラリやサーバによって実現される構造上，カーネルがサーバやライブラリに提供する SPI の抽象度は低いものに留まる．この様な機構を持った OS に於いても，任意の機能を持ったカーネルを容易に再構成可能な事は，SPI が固定されないという利点を持つ．

しかし，これら動的・静的再構成への要求に答える為には，FreeBSD 2.2.x で用いられている様な LKM では不十分である．つまり，OS の各機能を単に部品化するだけでなく，LKM の依存関係を整理し，ポリシーとメカニズムを分離し，LKM 間のインタフェイスを定義し，任意のサービスを提供可能なカーネルを再構築できるようにする必要がある．これにより，LKM を動的適応のみならず，静的再構成可能な機構として利用する事が可能となる．

5 スケジューリングモジュールへの応用

LKM が，実際に動的適応機構として有効である事を示す為，次世代マイクロカーネル研究プロジェクト [2] で開発されたマイクロカーネル (以下 RT-Mach) 上で新しいスケジューリングポリシーを LKM として実装した．RT-Mach 上での LKM は DKM という．

DKM 機構は，FreeBSD 2.2.x で用いられている LKM と比較して，DKM の依存関係が整理され，ポリシーとメカニズムが分離されている．従って DKM を容易に実装する事ができ，DKM のサイズも小さいものにとどめる事ができる．各ポリシーを実際にモジュール化した時のファイルシステム上の大きさを表 1 に示す．

RT-Mach 上の DKM 機構は，DKM サーバ及び DKM とで構成されている．DKM サーバはモジュール間の整合性を管理する．サーバはモジュールをロードする時，

1. ロードするモジュールを読み込み
2. それが正しいインターフェイスを提供しているか検査し
3. 外部シンボルとカーネルとのリンクをし
4. カーネル空間へモジュールを置く．

又，サーバは LKM 間の依存関係を把握し，ある DKM に先だって他の DKM が必要な場合にはそれをロードする役割も持っている．DKM はファイルとして二次記憶上に置かれており，関数とそれによって操作されるデータを持っている．

スケジューリングモジュールに特化したサーバの機能として，モジュールをロードした後，RT-Mach のスケジューリングポリシーをロードしたモジュールのものに変更する事がある．スケジューリングモジュールは単にメカニズムを提供するものであり，ポリシーを提供するものではない．従ってカーネルのポリシーをサーバが変更する必要がある．この概念図を図 1 に示す．

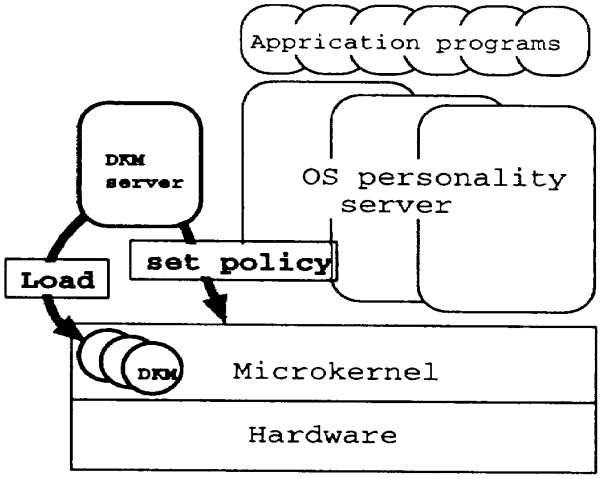

図 1: DKM 機構概念図

新しいポリシーはラウンドロビン方式を基礎に，優先順位によって実行時間が変化するというものである．このポリシーは，スレッドの切り替えが起こると

1. 実行待ちスレッドのキューの先頭からスレッドを取り出し
2. 優先順位に応じて実行時間を決定し
3. 実行時間だけ実行し
4. スレッドが実行時間内に終了しなかった場合には再びキューの最後尾に置く．

6 終りに

本稿は LKM 機構の有効性を確認する為に，スケジューリングモジュールを実装について述べた．DKM 機構では DKM 間のインタフェイスは定義されていないので，これを定義し，小型計算機や組み込み型システムなどへの適用を評価する必要がある．又，LKM 以外の他の手法 (SS, SL) に関しても定量的な評価が必要である．

今後は，他の手法の定量的な評価，及び組み込み型システム上での実装をする予定である．

参考文献

[1] Eric L. Hernes, "FreeBSD Device Driver Writer's Guide", http://www.freebsd.org/tutorials/ddwg/ddwg.html,1996

[2] 徳田 他: "MKng: 次世代マイクロカーネル研究プロジェクト概要", 第 55 回情報処理全国大会論文集, 1Z-2(1997).

[3] IEEE Std 1394-1995, High Performance Serial Bus, August 1996

[4] 盛合敏, 徳田英幸, "次世代 OS のためのマイクロカーネルトレイアーキテクチャ", 情報処理学会 OS 研究会,1997.

[5] David Golub, Randall Dean, Alessandro Forin, Richard Rashid, "Unix as an Application Program.", in *proceeding of the USENIX Summer Conference*, June 1990.

[6] Engler,D.R and Kaasheoek,M F.:Exterminate All Operating System Abstractions, in *proceedings of the 5th IEEE Workshop on Hot Topics in Operating Systems*(1995)

[7] 追川修一, 杉浦一徳, 西尾信彦, 徳田英幸, "マイクロカーネルにおけるカーネルモジュールのサポート", 第 53 回情報処理全国大会論文集,(1996)

[8] 追川修一, 喜多山卓郎, 徳田英幸, "スクリプト言語によるカーネルレベルでの動的適応", マルチメディア, 分散, 協調とモーバイル (DiCoMo) ワークショップ, 平成 9 年 7 月

[9] Tokuda, H., Nakajima, T., and Rao, P.: Real-Time Mach: Towards Predictable Real-Time System, in *proceedings of the USENIX 1990 Mach Workshop*(1990).