社団法人 情報処理学会 研究報告
IPSJ SIG Technical Report

2006－OS－103 (10)
2006／7／31

適応性と堅牢性をあわせ持つ *AnT* オペレーティングシステム

谷口秀夫、乃村能成、田端利宏
安達俊光、野村裕佑、梅本昌典、仁科匡人

岡山大学 大学院自然科学研究科

マイクロプロセッサや入出力ハードウェアの進歩には目覚しいものがある。さらに、通信路の伝送速度の向上も著しい。また、様々な場面で計算機が必要となり、提供するサービス種別も飛躍的に増大している。このような背景から、これらハードウェアの機能や性能を有効に利用でき、さらに多様なサービスの提供を支える基盤ソフトウェアが必要になっている。そこで、適応性と堅牢性をあわせ持つ *AnT* オペレーティングシステム(An operating system with adaptability and toughness)を開発している。ここでは、*AnT* オペレーティングシステムの設計方針と特徴的な機能について説明する。

Design for *AnT* Operating System

Hideo TANIGUCHI, Yoshinari NOMURA, Toshihiro TABATA,
Toshimitsu ADACHI, Yusuke NOMURA, Masanori UMEMOTO and Tadato NISINA

Graduate School of Natural Science and Technology, Okayama University

There is a remarkable thing for progress of a microprocessor and input-output hardware. Furthermore, improvement of transmission speed of a channel is remarkable, too. In addition, a computer is necessary in various scenes, and service classification to offer increases drastically. It is demanded that base software can use a function and performance of these hardware effectively. Therefore it was started development the *AnT* operating system (An operating system with adaptability and toughness) to have both adaptability and solidity. This article explains a design policy of the *AnT* operating system and a characteristic function.

1．まえがき

マイクロプロセッサや入出力ハードウェアの進歩には目覚しいものがある。さらに、通信路の伝送速度の向上も著しい。また、様々な場面で計算機が必要となり、提供するサービス種別も飛躍的に増大している。このよう

な背景から、これらハードウェアの機能や性能を有効に利用でき、さらに多様なサービスの提供を支える基盤ソフトウェアが必要になっている。

そこで、***Tender*** オペレーティングシステムの研究開発[1][2]の経験を生かし、以下を可能にする ***AnT*** オペレーティングシステム（An operating system with adaptability and toughness）を開発している。

(1) ハードウェアとサービスを様々に組み合わせたシステムを管理制御できる

(2) ネットワークを高い通信効率とセキュリティで利用できる

(3) ソフトウェア開発をオープン化できる

ここでは、***AnT*** オペレーティングシステムの設計方針と特徴的な機能について述べる。

2．適応性と堅牢性および開発機能項目

2．1　適応性と堅牢性

AnT オペレーティングシステムは、計算機システムの開発者と利用者の両者に「使いやすい」かつ「壊れにくい」という特徴を有する基盤ソフトウェアであることを目指す。

開発者にとって「使いやすい」ためには、新たなサービス（機能）の追加つまり新たなプログラムの追加を容易にできる必要がある。特に、組込みシステムの場合、新たなドライバプログラムの組込みが頻繁に発生する。このため、ドライバプログラムの新規作成の容易化だけではなく既存プログラムの流用ができれば好ましい。利用者にとって「使いやすい」ためには、高性能であるとともに、利用したいサービスを簡単に利用できる必要がある。したがって、両者の要求を満足するには、基盤ソフトウェアは高い適応性を有することが求められる。

そこで、高い適応性を有する基盤ソフトウェアの構成法が必要になる。具体的には、サービス（機能）の組込みや取外しが簡単なプログラム構成法、および利用者の利用履歴を考慮したプログラム実行制御法が必要である。また、計算機システムの性能低下の大きな要因であるメモリ間データ複写をゼロにする高速なプログラム間データ通信機構も必要である。

開発者にとって「壊れにくい」ためには、開発中のプログラムの暴走によりシステム全体が壊れることを防ぐ必要がある。利用者にとって「壊れにくい」ためには、いろいろなサービス（機能）を使っても相互干渉によって不都合な動作にならないような高い信頼性とともに、高いセキュリティが必要である。したがって、両者の要求を満足するには、基盤ソフトウェアは高い堅牢性を有することが求められる。

そこで、高い堅牢性を有する基盤ソフトウェアの構成法が必要になる。具体的には、組み込んだプログラムに不具合が発生しても他のプログラムやシステム全体に悪影響を与えない仕組み、およびネットワークを介したセキュリティ低下を防ぐ仕組みが必要である。

2．2　開発機能項目

適応性と堅牢性を確保するために、いくつかの機能項目を開発する。適応性と堅牢性が求める検討項目と開発する主な機能項目の関係を図１に示し、各機能項目について以下に説明する。

(A) プロセスのモード変更機能[3]

プロセスのプログラム走行モード（ユーザ／スーパバイザ）を自由に変更できる機能

(B) 適応制御機能[4]

システムの要求に合わせてプログラムの構成を適応させる機能

(C) ゼロ複写プログラム間通信機能[5]

メモリ間のデータ複写を行わないでプログラム間の通信を可能にする機能

(D) ドライバプロセス機能[6]

ドライバプログラムをプロセスとして動作させる機能

(E) 複数 OS 共存制御機能[7]

１台の計算機上に二つのOSを走行させる機能

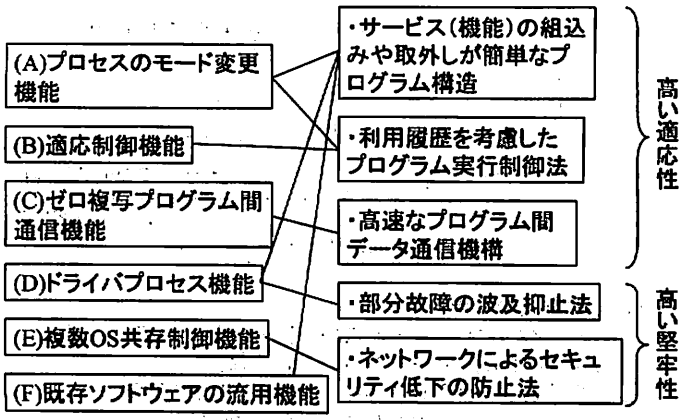


図1 主要機能項目と検討項目の関係

さらに、Linux の既存のサービス AP やドライバのプログラムを利用できるために、以下のものがある

(F) 既存ソフトウェアの流用機能

3．設計方針

プログラムの設計においては、以下の 3 つを基本方針とする。

(1) 適応型構造

システムの環境に合わせて発展するために、環境を学習し、環境に適応できる機能やプログラム構造を有する。このため、多くのシステムの共通な機能を内コアとして用意し、システムの環境に適応した機能を外コアとして発展させることにより、1 つのコアから環境ごとに変貌し無限個のオペレーティングシステム（OS）が誕生する。

(2) ソフトウェアの公開

開発したソフトウェアの普及を促進するため、ソフトウェアは公開する。このため、第 3 者のプログラム構造の理解を支援するため、複雑な機能や構造（例えば、ハッシュ構造）は採用しない。

(3) 開発工数の削減

OS そのものの開発を重視するため、開発の言語や環境は UNIX を利用する。また、次々に登場する新しい入出力ハードウェアを速やかに利用できるようにするために、入出力制御ソフトウェア（ドライバ）のユーザモードでの走行（プロセスとしての走行）を可能にし、既存ドライバの移植方法を確立する。

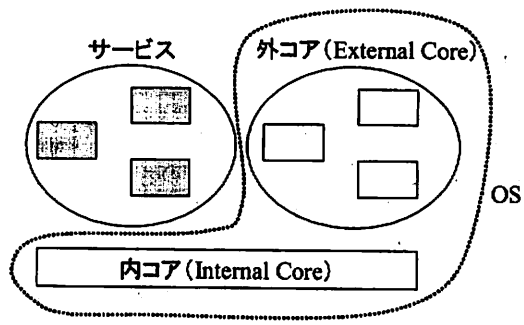
4．適応型構造

4．1 基本構造

プログラムは、OS とサービスからなる。OS は、内コアとプロセスとして動作する外コアからなる。サービスは、プロセスからなる。この様子を図 2 に示す。

内コアは、最小のシステムの動作を保証するプログラム部分である．外コアは、適応したシステムに必須なプログラム部分であり、動的に再構成な構造を有する。

サービスは、サービスを提供するプログラム部分である。



(1)OS：内コア＋外コア
・内コア：最小システムの動作を保証する部分
・外コア：適応したシステムに必須な部分(動的再構成構造)
(2)サービス：サービスを提供する部分

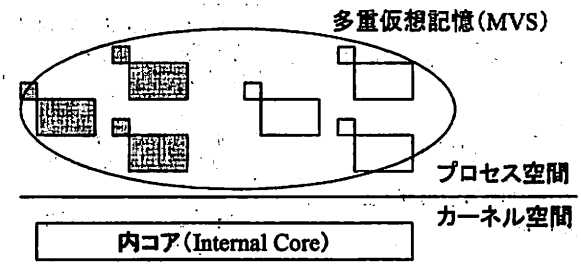
図2 基本構造

4．2　信頼性の確保

プログラム動作の信頼性を確保するため、記憶空間の分離と後作成プログラムの外コア実行を行う。この様子を図３に示す。

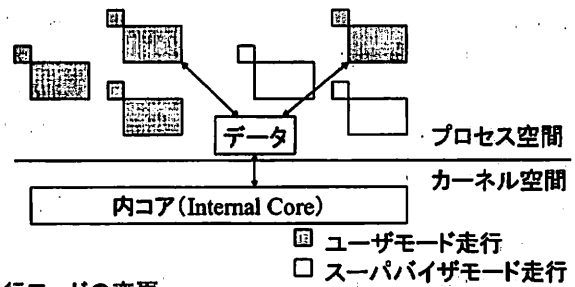
内コアはカーネル空間で実行し、サービスと外コアはプロセス空間で実行する。これにより、内コアを保護する。また、サービスと外コアの間および外コア間を保護するため、プロセス空間は多重仮想記憶とする。

ドライバなどのようなプログラムは、新たな入出力ハードウェアの登場と共に順次開発される。このため、OSの開発当初ではなく、後から開発されたプログラムを後作成プログラムとして区別する。多くの場合、ソフトウェアの開発においては、開発初期に開発されたプログラムに比べ、後作成プログラムは低品質な場合が多い。また、開発者によりソフトウェアの品質が大きく異なることは言うまでもない。したがって、開発時期の違いやドライバ開発元の違いにより、その信頼性（ソフトウェア品質）には大きな差が生じてしまう。このため、異なる信頼性を有するプログラムの共存制御が必要である。そこで、ドライバなどのようなプログラムは外コアとして実行し、多重仮想記憶による相互保護を行う。



(1)記憶空間の分離
　(A)内コアと外コア・サービスの分離——>内コアの保護
　(B)外コア間の分離——>外コア間の保護
(2)後作成のプログラム(ドライバなど)の外コア実行
　——>異信頼性プログラムの共存制御

図3　信頼性の確保



(1)走行モードの変更
　・必要に応じて、外コアをスーパバイザモードで実行
(2)ゼロコピーによるプログラム間通信
　・プログラム間のデータ授受をメモリマッピングで提供

図4　高性能化

4．3　高性能化

プログラム実行におけるオーバヘッドを削減し、プログラムの処理性能を向上させるため、走行モードの変更とゼロコピーによるプログラム間通信を行う。この様子を図４に示し、以下に説明する。

信頼性が高くかつ頻繁利用される外コアのプログラムは、スーパバイザモードで実行する。スーパバイザモードで実行することにより、外コアのプロセスからの内コア呼び出しを直接行える。つまり、例外を発生させることなく、関数呼び出しのようにジャンプ命令のように呼び出すことができる。これにより、外コアと内コアの連携を高速化でき、ドライバ処理の高性能化を実現する。

サービス提供を適切に行うためには、サービスを提供するプロセスとドライバプロセス

の高速な連携が必要である。具体的には、両プロセス間での高速なデータ授受（高速IPC）が必要である。さらに、プロセスと内コア間の高速なデータ授受も必要である。そこで、プログラム間のデータ授受をメモリマッピングで提供する。これにより、データ授受できるデータの基本単位は、MMUハードウェアのページに制限されるものの、高速な通信が可能になる。

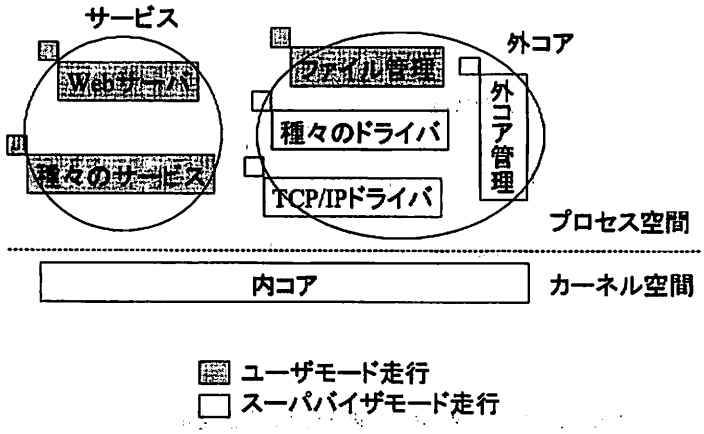


図5 ソフトウェア構造

5．特徴的な機能

5．1 機能概要

本OSは、特徴的な機能として、走行モード変更、ゼロコピー通信、および適応制御の機能を提供する。

走行モード変更機能は、外コアの走行モードを動的かつ自由に変更できる機能である。ゼロコピー通信機能は、プログラム間のデータ授受をゼロコピーで行える機能である。適応制御機能は、システムの環境に合わせてソフトウェアの機能や構成を自動的に変更する機能である。

本OSのソフトウェア構造の概観を図5に示し、特徴的な機能について以降で説明する。

5．2 走行モード変更

プロセスの高速な実行を可能にするため、走行モード変更機能がある。本機能は、他プロセス空間プログラムの走行モードを、ユーザモードからスーパバイザモードへ変更、あるいはスーパバイザモードをユーザモードへ変更する機能である。つまり、変更の指示により、プロセス空間プログラムの走行モードを変更し、変更されたプログラムと内コアの連携方式を変更する。

プロセス空間プログラムがユーザモード走行の場合、内コア呼び出しは例外を利用し、内コアからの呼び出しはプロセス割込みを利用する。これに対し、プロセス空間プログラムがスーパバイザモード走行の場合、内コア呼び出しは関数呼び出しであり、内コアからの呼び出しも同様である。したがって、プロセス空間プログラムをスーパバイザモード走行させることにより、プロセス空間プログラムと内コア間の連携オーバヘッドを大きく削減できる。

5．3 ゼロコピー通信

プロセスと内コアの間あるいはプロセス間の通信を高速化するため、ゼロコピーでのデータ授受機能がある。このためには、大きく以下の2つの機能がある。

(1) ページ単位による領域の確保と解放
(2) 2仮想空間の間での領域の貼り替え

プロセスと内コアの間あるいはプロセス間で授受を行うデータについては、ページ単位で管理される領域にデータを格納して、通信を行う。この領域をコア間通信データ域（IC域：Inter-core Communication Area）と名付ける。IC域は、内コアにより管理される。仮想空間の様子を図6に示す。なお、仮想空間の構成については、6章で詳しく説明する。

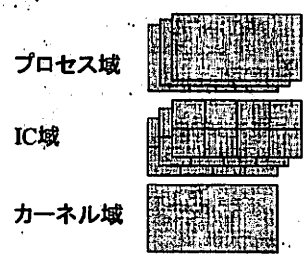


図6 ゼロコピー通信の機能を支える仮想空間

プロセスと内コアの間あるいはプロセス間でのデータ授受において、データの複写を避けるため、IC域に格納されたデータは、仮想空間のマッピング表を書き換えることによりデータを授受する。これをIC域の貼り替えと名付ける。IC域はページ単位であるため、このような貼り替えが可能である。

したがって、プロセスが内コアの機能を呼び出す際に渡すデータは、プロセスの仮想空間にあるIC域を内コアの仮想空間に貼り替える。一方、内コアがプロセスに実行結果を返却する際に渡すデータは、内コアの仮想空間にあるIC域をプロセスの仮想空間に貼り替える。また、プロセス間通信を行う場合、送信プロセスの仮想空間にあるIC域を受信プロセスの仮想空間に貼り替える。

つまり、通信する2つのプログラム間でのデータ授受を2仮想空間の間でのIC域の貼り替えにより実現し、ゼロコピー通信を実現する。

5．4 適応制御

新しいハードウェアやサービスに即座に対応できるため、適応制御機能がある。この機能は、大きく以下のものからなる。

(1) 新しいハードウェアやサービスの検知：新しいハードウェアの組み込みを監視し検出する。また、新しいサービスの要求を受け付ける。

(2) 必要な外コアの起動：新しいハードウェアの検知や新しいサービスの要求に基づき、必要なプログラムを起動する。

(3) 過去のプログラム利用履歴の管理：起動したプログラムについて、起動時刻や終了時刻を監視把握し、プログラムの利用履歴情報の獲得と管理を行う。この情報をプログラム利用履歴情報と名付ける。

(4) システム適応制御：プログラム利用履歴情報に基づき、頻繁に使われるプログラムについては、プログラムの常駐化、さらにスーパバイザモードへの走行モード変更を行う。一方、利用頻度が低下したプログラムについては、ユーザモードへの走行モード変更、さらにプログラムの終了指示を行う。これらの処理は、システム立ち上げの際だけではなく、定期的に行う。

(5) 異常プログラムの管理：プログラムの異常により終了した（させられた）プログラムについては、その情報を収集し、プログラム利用履歴情報に登録する。

6．仮想記憶

ここでは、仮想空間の構成について説明する。実メモリ空間と仮想空間の対応づけを工夫することにより、TLBミスの低減やゼロコピー通信の機能を支援している。図7に仮想空間の構成を示し、以下に説明する。

＜実メモリ空間＞

（1）カーネルの実行モジュールは、手続き部（Text部）のみからなり、データ部やBSS部を持たない。これにより、カーネルプログラムのROM化を容易にしている。領域管理用および各モジュール用領域は、カーネル初期化時に動的に確保され、カーネルのデータ部として利用される。

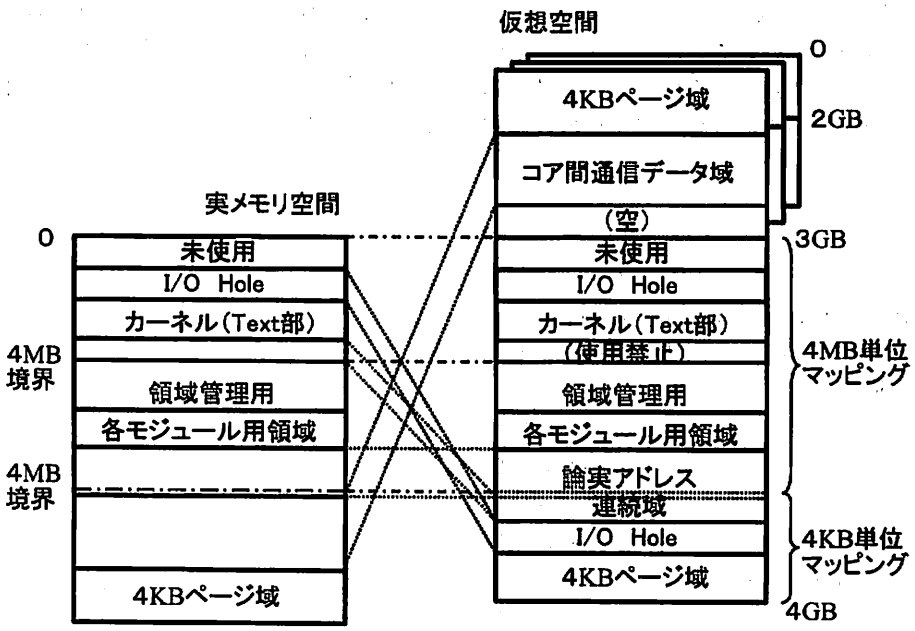


図7 仮想空間の構成

＜仮想メモリ空間（全容）＞

（2）0から3GBをプロセス空間、3GB以降をカーネル空間とし、プロセス空間は多重仮想記憶とする。これにより、プロセス空間上で走行するプログラム間の保護を可能にする。

＜プロセス空間＞

（3）0から2GBのプロセス空間は、サービスや外コアのプログラムが利用する空間であり、4KB単位でマッピングされている。これに対し、2GBから3GBのプロセス空間は、サービスや外コアのプログラムが相互のデータ通信に利用する空間であり、IC域（ICA）である。ICAは、4KB単位でマッピングされるものの、実アドレスと論理（仮想）アドレスがすべて連続しており、論実アドレスの変換を容易にしている。また、ICAは実メモリ常駐としており、入出力バッファとして利用時のページアウト対象外処理を不要にしている。

＜カーネル空間＞

（4）3GB以降のカーネル空間は、手続き部（Text部）を含む4MBの整数倍の空間を参照のみ、データ部を含む4MBの整数倍の空間を参照更新可能とし、各空間を4MB単位でマッピングすることにより、TLBの占有を抑制しTLBミスの低減を図っている。さらに、TLBからフラッシュされない設定も行っている。なお、この部分は、実アドレスと論理アドレスがすべて連続しており、論実アドレスの変換を容易にしている。

（5）データ部を含む4MBの整数倍の空間の老番地から以降の部分に論実アドレス連続域（SMA：Straight Mapping Area）がある。この部分は、実アドレスと論理アドレスが連続しており、論実アドレスの変換を容易にしている。SMAについては、仮想空間を構成するマッピング表などの利用を想定している。なお、手続き部（Text部）を含む4MBの整数倍の空間には、SMAの一部がマッピングされる形になる。これは4MB単位のマッピングの弊害であるが、この部分は参照のみの空間であるため、大きな問題はないと考える。

（6）I/O Holeは、参照更新可能である必要があるため、SMAに続く位置にマッピングしている。

（7）4KBページ域（PGA：PaGe Area）は、内コアが4KB単位でマッピングされる領域（ただし、実アドレスと論理アドレスは4KB単位で非連続である）を利用するための領域である。

7．おわりに

AnT オペレーティングシステム（An operating system with adaptability and toughness）について、設計方針、機能、および適応型構造について述べた。

設計方針として、適応型構造、ソフトウェアの公開、および開発工数の削減がある。機能として、走行モード変更、ゼロコピー通信、および適応制御の機能を提供する。適応型構造として、内コア、外コア、およびサービスの3層構造を有し、信頼性の確保と高性能化を実現する。

本研究の一部は、基盤研究（B）「適応性と頑健性を有する基盤ソフトウェアのカーネル開発」（課題番号：18300010）による。

＜参考文献＞

[1] 谷口秀夫, “***Tender*** オペレーティングシステムの開発”, 共立出版, bit, Vol. 32, No. 7, pp. 8-12, July 2000.

[2] http://www.swlab.it.okayama-u.ac.jp/lab/tani/research/tender-j.html

[3] 横山和俊, 乃村能成, 谷口秀夫, 丸山勝巳, “応用プログラムの走行モード変更機構”, 情処研報, 2004-OS-95, pp. 25-32, 2004.

[4] 仁科匡人, 谷口秀夫, “プログラムの動作を利用形態に適応させる適応制御機能”, 情処研報, 2006-OS-103, (掲載予定) 2006.

[5] 梅本昌典, 田端利宏, 乃村能成, 谷口秀夫, “***AnT*** における高速なプロセス間通信の実現”, FIT2006, (掲載予定) 2006.

[6] 野村裕佑, 乃村能成, 横山和俊, 谷口秀夫, 丸山勝巳, “走行モード変更機構を利用したデバイスドライバの実現,” 情処研報, 2006-OS-101, vol. 2006, No. 15, pp. 25-31, Feb. 2006.

[7] 田渕正樹, 伊藤健一, 乃村能成, 谷口秀夫, “二つの Linux を共存走行させる機能の設計と評価”, 信学論 (D-I), Vol. J88-D-I, No. 2, pp. 251-262, 2005.