6V-4

分散ＯＳの機能分割についての一検討

松下　智*，高畑泰志**，猪瀬　博*，斉藤忠夫*
（*東京大学工学部，**三菱電機）

１．はじめに

近年，ビットマップディスプレイによるマルチウインドやマウスなど，高度なマンマシンインタフェイスを装備したワークステーション（ＷＳ）が注目されている．この一方では従来のＴＳＳの持つ，ファイル・デバイス・プロセッサ・メモリなどの共有のような利点も再認識されてきた．そこで，ＷＳにファイルの転送や共有の機能を持たせたシステムがさかんに研究されている．これらの多くは，独立性の高いＯＳを持ったＷＳの集合体である．これに対し，比較的新しい研究として，通信層に異機種ノードを結合してシステム全体を一つのＯＳで管理し，ＷＳの複合体上に一つの計算機を構成する方式がある．今回提案するシステム COSMOS/2 は，これをさらに押し進めＯＳ自体を個々に独立した論理モジュールの集合体とすることにより，物理構成に依存せず，自由な構成が可能な分散ＯＳの実現への指針を得た．

２．システムの概要

本システムは，アプリケーションレベルでＵＮＩＸと互換性を持った分散ＯＳをめざす．本システムはモジュールという論理機能単位を各ノードに任意個数配置した構成をとる．ここでは，モジュールをその機能に対応して，サーバ・ハンドラ・マネージャに分類した．モジュール間の通信は，そのモジュールが同一物理ノードにある場合でも，すべてオブジェクト間通信層と呼ばれる通信層を介して統一的に行なわれる．

３．論理モジュールによる構成

３－１　物理ノードの構成

図１にディスクを持つＷＳノードの構成例を示す．ユーザプロセスはプロセスサーバ(ＰＳ)というモジュール内で実行される．ファイルの管理はファイルサーバ(ＦＳ)が行い，ディスク・端末・ＬＡＮ装置(Ethernet)など実際の機器との入出力は，各機器に対し論理的に一つずつ置くデバイスハンドラ(ＤＨ)が行う．各モジュール間の通信は通信層を介してのみ行われる．

一つのノードに配置されたモジュール群は低レベルスケジューラでスケジューリングされる．これはＵＮＩＸのそれと異なり，固定優先度を持った簡単なものであり，ノード内の割り込み処理はすべてここで行われる．こうして，ハンドラ以外のモジュールは機器依存性がなくなる．またメモリ管理は，モジュールの要求に従いメモリハンドラが行う．通信層は，モジュール間の通信を統一的につかさどり，通信先のモジュールのロケーションをオブジェクトｉｄ(OID)より判定し，通信のためのＤＨを起動するためのサブルーチンである．

ユーザプロセスは，ＰＳ内のプロセスサーバカーネル(ＰＳＫ)でスケジューリングされる．ここでは，通常のＵＮＩＸのように，プロセス寿命などを考慮した比較的高度なスケジューリングが行なわれる．ＰＳＫはまた，ユーザプロセスから出されたシステムコールの実現のため，必要なモジュールと通信する，いわゆるプロトコルコンバータの役目を果たす．これが，おおよそ，従来のＵＮＩＸのカーネルに相当する．

３－２　モジュールの種類と機能

上記，a. ＰＳ，b. ＦＳ，c. ＤＨ以外のサーバの機能を以下に示す．ｄ～ｆはシステムの集中管理機能でありシステムで唯一存在する．

d. ファイルマネージャ(ＦＭ)　システム全体で一意に定められた広義のファイル名を，その指し示すファイルやデバイスのｉｄ(oid)に変換する．(ﾈｰﾑｻｰﾊﾞに相当)

e. プロセスマネージャ(ＰＭ)　プロセスのforkの管理や，execの際，配置すべきＰＳを決定したり（負荷分散の管理），シグナルの管理を行う．

f. ノードマネージャ(ＮＭ)　各ノードの動作監視やユーザ管理を行う．ここが，動的なブートを管理する．

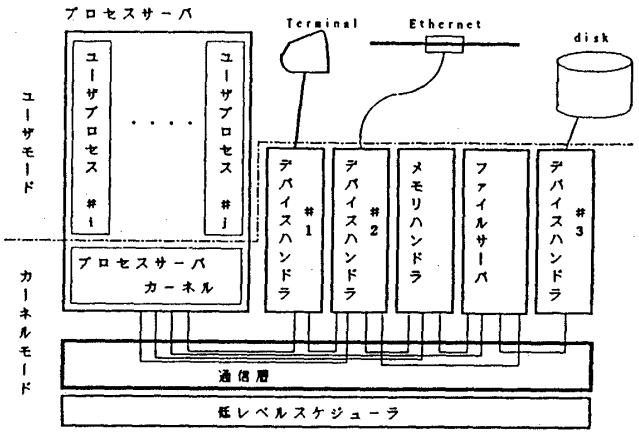

図１　ノード構成例

A Study of functional distribution in Distributed Operating Systems
Satoshi MATSUSHITA[1], Yasushi TAKAHATA[2], Hiroshi INOSE[1], Tadao SAITO[1]
1, University of TOKYO, 2, Mitsubishi Electric CORP.

g. パイプサーバ（ＰＰＳ） プロセス間通信に使われるＦＩＦＯ，すなわちパイプの入出力管理を行う．

h. ローカルファイルマネージャ（ＬＦＭ） ローカルディスク上でファイル名管理を行なう．

i. ゲートウェイサーバ(ＧＳ)， a～hより成るシステムを上位のネットワークなどに接続して，ツリー構成の大きなシステムにする際の接続部である．

4，オブジェクト間通信

4－1 オブジェクト識別子 (oid)

本システムでは，通信の対象を統一的にオブジェクトと呼び，システムでユニークな識別子 oid(Object ＩD)を与え管理する． oidは，モジュール識別子と，そこで管理される実体の識別子より構成される．本システムでは，実体を持たないマネージャやハンドラも統一的にオブジェクトとして扱う．

4－2 通信層の構成

通信にＬＡＮを用いた場合の通信層構成を図２に示す．ＬＬＣ(Logical Link Control)層では，パケットのアセンブリ，リアセンブリを行う．ＬＬＣ層以下はデバイスハンドラとして実装され，割り込みによるスケジューリングがなされる．ＩＯＣ(Inter Object Communication)層は oidからどのハンドラを用いてどのノードへ通信すべきか決定をする．この構成により，ＬＬＣ層以下を担当するハンドラを複数持たせ，ＬＡＮを多重化することや，システムの一部を共有メモリや並列バスを用いた構成にすることが可能になる．

5 ファイルシステムの構成

本システムでは，ＵＮＩＸと異なり，ディレクトリファイルのみを１つのディスクに格納し，ＦＭで集中管理する．こうして，ファイル名を oidに変換する際，ＦＭがいくつものＦＳを渡りあるく効率低下を防ぐ．この構成のファイルシステムは，ファイルアクセスが分散処理される，１つの大きなルートファイルシステムとみなせ，ＵＮＩＸと同様の方法でローカルディスクをmount, umountできる．ローカルファイルシステムはＵＮＩＸと同様の構成であり，ＬＦＭ(Local FM)およびＦＳで管理される．こうして，カレントディレクトリがローカルにある通常のファイル処理は，ネットワークを介さずＬＦＭのみで実行される．これを，図３に示す．

6 マウントによるシステムの動的拡張

前節で述べた構成により本システムは，システムとして動作していない時にも，ローカルファイルシステムをルートとし単一ノードで立ち上がりうる．その後，共用のＦＳ・ＦＭが立ち上がった時点で，ローカルなファイルシステムが自動的に共用ファイルシステムに mountされ，ファイル空間が拡張される．この点で，独立ＯＳを持つＷＳの集合体である，ホスト・ホストモデルの特徴を兼ね備えている．さらに，システムが，ＧＳを介し上位のネットワークとつながったツリー状の構成をしていれば，同様にしてファイル空間は動的に拡張される．この様子を図４に示す．この場合，上層のＦＳ，ＦＭなどの構成は下層のものと全く同じで良く，拡張性が高い．（プロセスに関しても同様に拡張できる．）

7．性能予測

μＶＡＸ-Ⅱ/4.2BSD のシステムコールの生起を実測し，これに従い，ＬＡＮに１０MbpsのEthernetを用いた構成について，簡単な評価を行った．

この結果，μＶＡＸ程度のノード用い，それら全てをフルに動作させた場合，８７ノードでＬＡＮのボトルネックを生ずる，これは，ＷＳを個人が占有する通常の形態では，５１０台のＷＳからなるシステムが構成可能なことを意味する．

また，ディスク容量から一つのＦＭで管理できるＦＳの数を求めると，８７となり，実用上充分である．

8．おわりに

資源の集中管理機能の採用により，ＴＳＳの持つ資源共有の容易さを持ち，かつ，構成ノード数の変化に応じて共有資源の量が動的に拡張されていく分散処理システムが実現される．また，このシステムでは，ＯＳを論理モジュールに細分化することで，必要に応じてＯＳ機能の分散をおこなう．さらに，これらの間の通信はオブジェクト間通信として統一的に扱い，高い柔軟性をもつ．

現在，サーバ間で重複してアクセスするデータを極力減らし，通信オーバヘッドを回避するための実現法，および，ブートの実現形態などについて検討中である．

＜参考文献＞

[1]橘爪宏達，"ネットワーク資源共有のための統合型ＮＯＳの研究"，学位論文，1983.

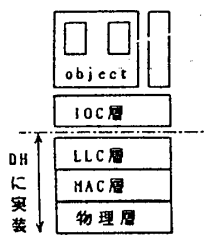

図２ 通信層の論理的構成

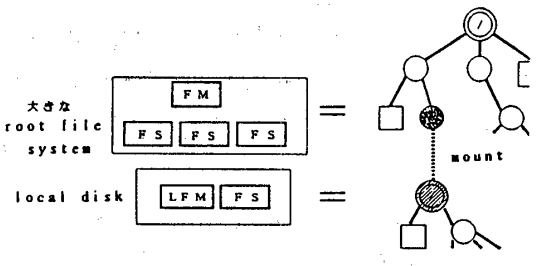

図３ ローカルファイルのマウント

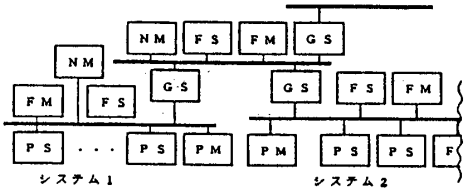

図４ 階層システム構成