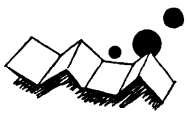
解　説



プロセス間通信機能指向ミニコンピュータ複合体 KOCOS のアーキテクチャ†

上林憲行††　相磯秀夫†††

1. 分散処理システムの交信方式による分類法と KOCOS の位置付け

近年，計算機システム技術の最重要課題としての認識が深まって来ている分散処理システム[2] (DPS: Distributed Processing System) は各応用分野におけるニーズの多様性を裏付けるようにそのシステム形態・構成技術は千差万別[1]である．しかし DPS の設計課題は基本的には物理的に分散配置される構成プロセッサ間の交信方式・交信媒体・その実現レベルの選択にある．その選択によって DPS の基本的性能や応用範囲は決まってしまう．DPS の交信方式による一分類法を**表-1** に示す．以下その簡単な補足説明を通して DPS のカテゴリにおける KOCOS の位置付けを明らかにする．

まず DPS はその基本交信機構の違いからグローバルアドレス空間システムとメッセージ交換システムに大別される．

(1) グローバルアドレス空間システム：この種のシステムでは各プロセッサがその命令レベルで任意のシステム構成メモリにアクセス可能なアドレス変換機構が用意されており，プロセッサ間交信はメモリ上の情報を介して間接的に遂行される．この方式は構成プロセッサ間に共有空間を提供する事を DPS 構成技術の基本としている．この型の DPS はさらに共有空間におけるメモリの物理的構成方式の違いから以下の3つに分類される．

(1)—a　共有メモリ型：構成プロセッサはプライベートメモリを所有せず，システム内には各プロセッサから独立した共有メモリのみが存在する．

表-1　交信方式による分散処理システムの分類法と KOCOS の位置づけ

	グローバルアドレス空間システム			メッセージ交換システム		
分類法	共有メモリ型	共有メモリ＋プライベートメモリ型	プライベートメモリ型	プロセッサ間通信型	プロセス間通信型	階層プロトコル型
システム例	C.mmp	Super IMP	Cm*	HXDP Tandem -16	KOCOS DCS	Kuipnet
交信スイッチの種類	クロスバスイッチ	バスカプラー	K.map	プロセッサ間通信バス	バスインタフェースユニット リングインタフェースユニット	IMP
並列処理レベル	命令レベル			プロセスレベル		ジョブレベル
交信機能の実現レベルの比重	ハードウェア			ファームウェア		ソフトウェア
プロセス間同期方法	共有変数型同期システム			メッセージ型同期システム		
プロセッサの地理的分散	地理的分散度大					
プロセッサ間交信頻度・単位	交信頻度			交信単位		
応用システム設計者の自由度・生産性	自由度			生産性		
システム分類	マルチプロセッサシステム			複合体		ローカルネットワーク

† The Architecture of Interprocess Communication Facility Oriented Minicomputer Complex, KOCOS by Noriyuki KAMIBAYASHI and Hideo AISO (Faculty of Engineering, Keio University).

†† 慶応義塾大学工学研究科　††† 慶応義塾大学工学部電気工学科

(1)—b 共有メモリ+プライベートメモリ型：構成プロセッサは原則としてプライベートメモリを所有し，プロセッサ間交信に必要な情報のみを物理的に独立した共有メモリに格納することによって，プログラムの局在性による並行処理と共有メモリを介した同期・通信の融通性の調和を図っている．

(1)—c プライベートメモリ型：独立した交信用共有メモリは存在しないがアドレス変換機構を介して任意のプライベートメモリへアクセス可能となっている．これは(1)—b に比べてさらにプロセッサの独立性・自律性（プログラムの局在性，システムの機能分散）を重視した構成である．

(1)型のシステムは構成プロセッサの地理的分散には不向きであり，共有空間を実現する各種のアドレス変換機構やアクセス競合防止機構が比較的高価となる割には，アクセス競合頻度によって性能の上限が決ってしまう．そのために並列に実行されるプログラムの局在性，機能分散・専用化に注目して，共有空間（スイッチ）へのアクセス頻度を最小限に押えようとするアプローチ((1)—b, c)が1つの潮流のように思われる．

(2) メッセージ交換システム（非グローバルアドレス空間）：メッセージ交換システムではシステム全体を制御可能なアドレス空間は存在しないが，構成プロセッサ間での直接的なメッセージ交換機構が用意され，交信は物理的にメッセージを交換することによって成立する．メッセージ送・受信機能を提供する事がこの型の DPS 構成技術の柱となる．この型はさらにメッセージ交換の主体の抽象度の違いにより以下の3つに分類される．

(2)—a プロセッサ間通信型：この型はプロセッサ間のハードウェア・メッセージ交換機構の設定を DPS 構成技術の骨子としている．これはプログラムの局在性，機能分散・専用化，地理的分散の視点から(1)—c 方式に比べてより直接的な交信機構を実現する事によってその適応分野を拡大するという目的を持っている．

(2)—b プロセス間通信型：これは実際のシステム運用上 (OS の機能構成) の視点から，プロセス（システムの論理制御単位）を交信主体としてプロセス間の通信・同期機能を体系化し，さらにその効率の良い実現のためのアーキテクチャ（ハードウェア，ファームウェア）を提供することを DPS 構成技術の主眼とするアプローチである．つまり DPS 環境におけるハードウェアとソフトウェアの接点技術として，プログラムの並行動作性，機能分散処理を実現するプロセス間通信機能に焦点を置いたものである．

(2)—c 階層プロトコル型：この型は一般にローカルな環境を支配する OS の存在を前提として基本的にはホスト間のプロトコル階層によって各プロセッサ間のメッセージ交換機能が実現される．ここでは各層のプロトコル設計と実現方式が DPS 構成技術の中心課題となる．

2. KOCOS のシステム構成と性能諸元

KOCOS (Keio-Oki's Complex System) は分散処理アーキテクチャ研究の一環として既存の異機種ミニコンピュータをプロセッサの変更なしに複合化するという制約条件下で開発された実験システムである．そして研究の焦点を次の2点に置いた．

(1) プロセス間通信機能 (Interprocess Communication Facility : IPCF)：複数のプロセッサにまたがる並行プロセス間の実行順序制御・メッセージ送受信機能の体系的方式．

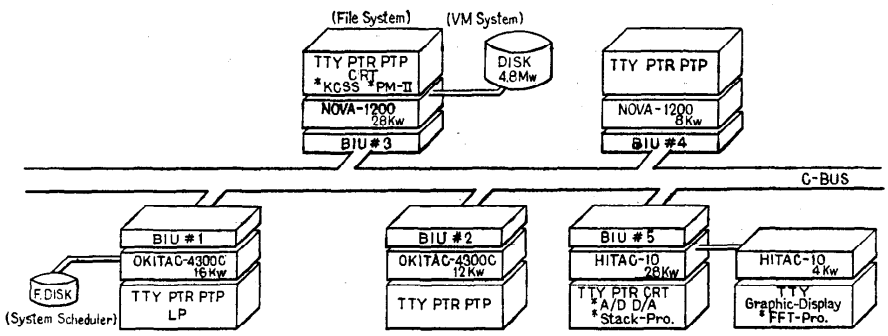


図-1 KOCOS のシステム構成図

(2) プロセス間通信機能の実現方式：コミュニケーションサブシステムの構成方式と効果的な機能分散化およびその中核をなす複合体向き高機能インタフェースプロセッサ.

この2つは DPS 指向 OS (OS 核) とシステム・アーキテクチャの接点となる重要課題である.

KOCOS のシステム・アーキテクチャ[4),7),10)], プロセス間通信機能[6),8),9)], システムソフトウェア[5),9)]については論文で詳細に報告した.

1° システム構成要素と性能諸元：KOCOS のシステム構成を図-1 に示す. 構成プロセッサは DMA 機能を持つミニコンピュータであり EP (Element Processor) と呼ばれる. 各 EP は 32 本のラインから構成される C-Bus (Common Bus) と呼ばれる双方向性の単一バスを介して接続される. BIU (Bus Interface Unit) はマイクロプロセッサや LSI を中心として構成され, EP と C-Bus 間の物理的な入出力インタフェースの機能に加えてプロセス間通信機能の物理的・論理的中枢機能がファームウェア化されている. DBC (Distributed Bus Controller) は各 BIU にあって分散論理による C-Bus の使用権競合制御を行う. 各 EP にはプロセスの動的制御を行う多重プログラミング核と OS の機能を実現するシステムプロセスから構成される LOS (Local Operating System) が存在する. システム構成要素の性能諸元を表-2 に示す.

2° システム動作概要：現在 KOCOS はハードウェア (BIU), ファームウェア (BIU 制御プログラム), ソフトウェア (LOS) の実装が完了しプロセス間通信機能の評価データ収集を通して実験システムとして使命を果している. また KOCOS のシステムアーキテクチャを適用したいくつかの実用レベルのシステムが開発 (BIU の出荷は約 30 台) されている. その代表的システム例の一つはオムニバックスの蓄積交換処理システムであり, 信頼性・システム再構成力向上のために OKITAC-4300 の複合体として実現されている. この例を始めとしていろいろな応用分野で KOCOS で研究・開発された DPS 構成技術の成果が還元されている.

3° 開発期間と人員：実験システムは本格的な方式設計の開始 (1974 年 9 月) から約 1 年半後にハードウェア実装が完了し, 1976 年 10 月には OS 核が完成し総合運転が可能となった. 開発期間と実質的な開発人員を表-3 に示す.

表-2 システム性能諸元

C-Bus 諸 元	
バ ス 長	20m (最大 200m)
ラ イ ン 数	32本 (双方向性)
エ ラ ー 率	10^{-11} 以下
転 送 能 力	約 250 k 語/秒
転 送 方 式	可変長ブロック転送 (最大 64 k 語)
転 送 幅	16 ビット並列
転送同期方式	ハンドシェイク方式 (1語単位)
ブロック形式	アドレス, 非アドレス型
BIU 諸 元	
マイクロプロセッサ動作速度	マイクロ命令サイクル 166 ns
マイクロプロセッサ演算幅	16 ビット (I3002×8)
マイクロ命令仕様	32 ビット×512 語 (ROM)
マクロ命令種類	問題向き命令 20 種類
マクロ命令実行時間	1 μs～ (命令によって任意)
マクロメモリ容量	512 語 (RAM)+1.5 k 語 (ROM)
BIU 制御プログラム	約 1 k 語 (16 ビット)
ハードウェアキュー	FIFO メモリ (18 ビット×40 語)×3
プロセス間通信能力	最大 16 プロセス対の同時サポート可
LOS 諸 元	
プロセススケジューラ	コード 246 語, データ 35 語
メ モ リ 管 理	コード 114 語, データ 14 語
プロセス間通信機能	コード 609 語, データ 81 語
システムバッファ	512 語
システムプロセス	約 1.5 k 語

表-3 開発期間と人員

開発・設計項目	期間・開発人員
方 式 設 計	5 ヵ月×3人
ハードウェア論理設計	5 ヵ月×6人
ハードウェア実装設計	3 ヵ月×4人
ハードウェア実装	3 ヵ月×3人
BIU ファームウェア設計	3 ヵ月×3人
BIU 総合ディバック	2 ヵ月×4人
BIU 開発支援システム設計・実装	3 ヵ月×3人
ミニコンピュータ基本ソフトウェア設計・実装	3 ヵ月×3人
LOS 設計・実装	10 ヵ月×3人
評価システムの設計・実装	1 ヵ月×2人
デモストレーションプログラム	1 ヵ月×2人
総合ディバック	2 ヵ月×3人

3. 分散処理指向プロセス間通信機能

3.1 プロセス間通信機能指向アーキテクチャ

ここは KOCOS の特徴である IPCF 指向アーキテクチャの基本概念と技術的課題を以下の項目に整理して述べる.

1° 高水準論理インタフェース：プロセスはプロセス協調を具現化するプロセス間通信基本命令 (IPCP: Interprocess Communication Primitive) を用いて直接 DPS 環境へアクセスできる. そのため①構成プロセッサにはリモート通信ソフトウェアが不要, ②異機種結合によるソフトウェア作成労力の軽減, ③構成プロセッサの相違をプロセス間通信機能で吸収, 等の利

点が得られた.

2° 並行プロセス制御環境：多重プログラミングOS設計上の基本命題は並行プロセスの制御であるが，DPS環境（交信プロセスの物理的分散）も同様に並行プロセス制御環境として把握される．DPS構成技術にこの概念を適応することによる利点は次の通りである．①DPS環境の仮想化（ローカル・リモート通信の論理インタフェースの統一），②構成プロセッサ上のOS核との整合性，③DPS上で展開される応用プログラム設計の容易性，④分散処理OS核の基本概念，

3° コミュニケーションサブシステムの高機能化：構成プロセッサが本来の処理に専念できるように，複合化に伴う処理（DPS環境アクセスの物理的・論理的制御）をコミュニケーション・サブシステムに効果的に機能分散しそのファームウェア化を図った．コミュニケーションサブシステムは交信の物理スイッチであるC-Busと複数の複合体向きインタフェースプロセッサ（BIU）から構成され，プロセス間通信機能の論理体系と相まってリモート通信の効率と構成プロセッサの稼動率の向上を実現している．

3.2 分散処理指向プロセス間通信機能

前節でDPS環境も究極的には並行プロセス制御環境と解釈可能であることを述べた．並行プロセス環境ではプロセス間の協調（Cooperation）によって目的の仕事は遂行される．それはDPS環境でもまったく同様である．ここではDPS環境におけるプロセス協調機構についてKOCOSの経験を踏まえて特にプロセス間通信基本命令を中心に考察する．

1° プロセス間通信基本命令の設定レベル：IPCPには①共有変数型（Shared variable type）と，②メッセージ型（Message System）が代表的である．前者は基本的には共有資源に対する相互排除操作であり，後者はプロセス間でメッセージ交信機構が用意され，交信メッセージによってプロセス協調が達成される．両者の方式について考察すると，環境操作の自由度の点からは①型が有利であるが，反面プログラム作成レベルが低くソフトウェアの複雑さが強いられる．一方②型はプログラムの作成レベルは改善されるがその汎用性，機能のバリエーションが保障されていないとプロセス協調のオーバヘッドを招く．物理環境の制約性の観点からは①型は共有空間が前提であるが，②型は一般にDPS環境の制約を受けず，むしろメッセージ通信はDPS環境に適している．

2° 分散処理指向プロセス間通信基本命令

DPS環境において効率の良いプロセス協調手段を提供するために必要な留意点を次に列挙する．

①能動プロセス，受動プロセス等の交信状態の差異，②交信プロセス間の作用の方向性における主従関係，③交信プロ間の認識度に応じた機構，④非同期処理環境を活かす機構，⑤プロセスの自律的・動的な挙動を実現する機構

KOCOSで実現されたIPCPについて簡単な特徴を述べることによってDPS指向IPCPについて考察する．

（1） プロセス間通信メッセージ（IPCM）：プロセス協調はリモート・ローカル通信の区別なくIPCMの交換によって統一的に処理される．IPCMはIPCPのキャリアとして役割を持っている．

（2） メッセージ通信：メッセージ転送機能としては，①論理リンク型（プロセス間で直接，論理リンクの確立が必要），②メイルボックス型（プロセス間で論理リンクが暗黙に成立している）の2つの型が必要である．また受信系が交信における主体となるような機構も不可欠である．

（3） イベント通信：DPS環境でのプロセス間の同期が効率よく実行されるようにメッセージ通信機能に加えてイベント通信機能が導入されている．これには，①システム・イベント型（相互のプロセス状態の遷移作用を起す），②プライベート・イベント型（相互の状態遷移作用でなく，単にイベントを交換する）が用意されている．特に後者は交信プロセスの自律性のある協調と環境に動的に応じられるように設定されたものである．

（4） Access Anyモード：DPS環境ではプロセスの性格によりあらかじめ交信プロセスが規定できない状況がある．その場合受動プロセス側ではSend AnyおよびReceive Anyのモードが必要である．

（5） IPCPの非同期制御：IPCPの制御は一般に，①完全同期方式（依頼した仕事が完了するまでプロセスはSuspend状態），②非同期方式（プロセスとモニタは非同期処理可能）の2つの方式があるが，DPS環境では②方式が非同期処理性の向上の観点から有効である．

（6） 交信環境の相違の吸収：ブロードキャスト転送やバッファ環境の相違を吸収するメッセージ転送終了遅延機構やデータチェイニング機構をコミュニケーションサブシステムに吸収している．

KOCOS で設定されたプロセス間通信基本命令を表-4 に，またその実効転送能力を表-5 に示す．

4. 分散処理システム開発の問題点と課題

4.1 分散処理システム開発の問題点

KOCOS の開発を通しての率直な感想は分散処理システム設計・開発の困難さである．その理由を要約すると次の通りである．

1° DPS 構築技術の多様性と複雑さ：DPS 構築技術は計算機システム技術（論理回路から OS まで）を網羅し，かつ相互の関連が密接な総合技術という性格を持っている．総合技術，例えばソフトウェア技術（OS の機能・構成，言語機能），フォームウェア技術（マイクロプロセッサ，マイクロプログラミング，機能デバイス），ハードウェア技術（プロセッサ間結合方式，データ転送方式，共有システム資源の競合防止機能）等であり，さらにシステムの通信機構やコミュニケーションサブシステムの構成・機能等は，上記の技術の効果的機能分担を考慮して設計する必要がある．

2° ディバックの困難さと開発支援システムの重要性：DPS では様々な角度から機能や論理の分散化が推進されるが，ディバックの段階では集中論理・逐次処理の環境に比べて多くの困難な問題を抱え込むことになる．例えば非同期動作ゆえに必要となる資源競合防止機構（ゲートからプロセスレベルまで各種）の動作チェックは，実際に競合状況（非常にクリティカルなタイミングで生じる）を設定することさえ至難の技である．また一般に DPS の構成プロセッサはソフトウェア・ツールが貧弱である．そのため動作が不完全な開発時にシステムとは独立にソフトウェア設計・検証が可能な開発支援システムが必要である．

具体的には KOCOS では以下の開発支援システム

表-4 KOCOS のプロセス間通信基本命令

		プロセス間通信基本命令	パラメータ	オプション B	D	P	相手プロセスへの状態遷移作用	交信の主従関係	Receive any or send any	非同期型制御
メッセージ転送	メイルボックス型	READ	Spn, Dpn, Ba, Bs, Toc	×	○	○	indirect	equal	×	○
		WRITE	Spn, Dpn, Ma, Ms, Toc	○	○	○	indirect	equal	×	○
	論理リンク型	FORCE READ	Spn, Dpn, Ba, Bs, Toc	×	○	○	direct	master	receive any	○
		READ ANSWER	Spn, *, Ma, Ms, Toc	×	○	○	indirect	slave	／	×
		FORCE WRITE	Spn, Dpn, Ma, Ms, Toc	○	○	○	direct	master	send any	○
		WRITE ANSWER	Spn, *, Ba, Bs, Toc	×	○	○	indirect	slave	／	×
		MASTER READ	Spn, Dpn, Ba, Bs, Ma	／	○	／	／	master	×	○
		MASTER WRITE	Spn, Dpn, Ma, Ms, Ba	／	○	／	／	master	×	○
プロセス協調	システムイベント型	STOP	Spn, Dpn, P1, P2, P3	／	／	／	direct	master	／	○
		START	Spn, Dpn, P1, P2, P3	／	／	／	direct	master	／	○
		SIGNAL	Spn, Dpn, Et, P1, P2	／	／	／	direct	master	×	○
		AWAIT	Spn, Dpn, Et, Ec, P1	／	／	／	／	slave	receive any	×
	プライベートイベント型	ADVANCE	Spn, Dpn, Et, P1, P2	／	／	／	／	equal	×	○
		OBSERVE	Spn, Dpn, Et, P1, P2	／	／	／	／	equal	receive any	○

B: ブロードキャスト転送　D: 離散データ転送　Spn: ソース・プロセス名　Dpn: ディスティネーション・プロセス名
Ma: メッセージ・アドレス　Ms: メッセージ・サイズ　Ba: バッファア・ドレス　Bs: バッファ・サイズ　Toc: タイムアウト・カウント
Et: イベント・タイプ　Ec: イベント・カウント　P: パラメータ　P: 部分転送

表-5 各プロセス間通信基本命令におけるリンク時間とデータ転送能力

プロセス間通信基本命令	LT (μs) リンク時間		データ転送能力 (kW/sec) n=4		n=16		n=64		n=256		n=1024		n=4096	
	L*	R*	L	R	L	R	L	R	L	R	L	R	L	R
WRITE	594	753	6.29	5.18	19.9	19.3	43.4	60.4	61.5	129	68.7	181	70.7	201
READ	652	1020	5.76	3.85	18.6	14.6	41.7	48.2	60.6	114	68.4	173	70.6	198
FWRITE-WANSWER	1060	2800	3.62	1.42	12.6	5.56	32.9	20.6	55.3	63.5	66.6	133	70.1	182
FREAD-RANSWER	1080	2330	3.57	1.70	12.4	6.65	32.7	24.3	55.1	71.9	66.5	141	70.1	186
MWRITE	134	652	22.7	5.96	46.5	22.0	63.0	66.7	69.1	136	70.8	184	71.3	202
MREAD	134	948	22.7	4.14	46.5	15.9	63.0	51.0	69.1	118	70.8	175	71.3	199
Control Type	180	610												

L*: ローカルプロセス間通信　R*: ローカルとリモートプロセス間通信　n: 転送語数

を作成した．①BIU 内蔵マイクロプロセッサ用サポートソフトウェア（準汎用マイクロ・マクロアセンブラ，各種ローダ，テスト・評価システム）：約 5k ステートメント数（主に Fortran）で Univac-1106 上に実装，②ミニコンピュータ用基本ソフトウェア（エディタ，ローダ，ユティリティルーチン，デバッガ）：約 3.5 K step 数（アセンブラ），③総合ディバック用テストプログラム：約 2K step 数．以上からも開発時におけるサポート・ソフトウェアの比重は大きいものであり，システム・ソフトウェアの効率良い作成・デバックが可能な環境を整備する重要性を痛感した．また①は OS 機能が豊富で強力な大型機上で実装したわけであるが，サポートシステム自体の開発，それを利用してのシステム・ソフトウェアの開発は，TSS モードで作業が可能で，かつソフトウェアの保守等でも大きな恩恵を授けられる大型機上で，整備する方が最終的には得策であると思う．

3° DPS 環境の利用技術：詳細な DPS 構成技術を知らない応用システム設計者にもその環境を十分に活用できる利用技術を提供することは DPS の命運を握る重要課題である．それは応用システムの生産性と環境操作の自由度(効率)という相反した要求を満足させるという困難な問題でもある．

具体的には分散処理システム環境の操作能力をどの程度の水準（アドレス空間，プロセッサ間通信，プロセス間通信，スパーバイザ機能，リモートジョブエントリ等）で応用システム設計者に対して開放すべきかという問題であり，次の点について考慮すべきである．①分散処理システム環境の操作・表現能力を備えた言語：KOCOS ではプロセス間通信基本命令を内包した言語の設計を試みたが，環境の特質や OS の機能との関連が強く専用的色彩が濃厚となり，言語のシステムからの独立性（汎用性）を保障するのは困難である，②分散処理指向 OS・ユーザインタフェース機能（資源管理，実行スケジューリング，メモリ・アロケーション）：システム・コマンド，スーパーバイザ機能の選択，以上は相互に密接な関連があり，それらを踏まえて利用技術を設定する必要がある．

4.2 分散処理システムの技術的課題[3)]

1° PMS レベル設計の標準サポート：現状では複合体を開発するのはまだまだ困難である．マイクロプロセッサやメモリは LSI によって設計・実装の容易性と信頼性の大幅な改善の恩恵を授けている．しかし DPS 構成技術の要であるスイッチ機構やインタフェース機構を従来通りディスクリートで設計する状況では，信頼性や設計・実装の容易性を損なう結果となり，実質的な PMS レベル設計は不可能である．PMS レベルでのシステム設計の容易性・自由度を高めるためには，①PMS レベル設計を意識したプロセッサアーキテクチャの確立，②スイッチ機構の LSI 化，③インタフェース機構（アドレス変換，プロセッサ間通信，相互排除，デッドロック解除）の LSI 化，が必要である．特に②③は複合体設計上，各種周辺インタフェースの LSI 化によりマイクロプロセッサシステムの設計が飛躍的に容易になったと同様の効果を期待できる．以上の論点を整理すると現状では DPS 構成技術において LSI の恩恵を享授しているのは一部分である．実質的な PMS レベル設計が可能な DPS の標準技法化を強く意識したマイクロプロセッサファミリの開発が必要である．

2° 分散処理指向システム核[11)]：一般に DPS はその成立の背景から応用依存性の高いシステム形態を要求される運命にある．そうした状況で様々なシステム形態に応じられる汎用の OS を提供するか，個々の応用ごとに専用の OS を設計するかの選択は難しい問題である．この問題に対して分散処理指向システム核(DPS 環境を設定するのに必要な最も基本的な機能＝論理的には並行プロセス制御環境の実現）という概念を導入する．著者らは計算機システムの機能階層と OS の設計構造に対する考察から計算機システム(DPS)を，①並行プロセス制御環境を実現するシステム核層，②プロセスによって機能表現される機能表現層との 2 層の明確な分離が可能であることを示した．ここでシステム核アーキテクチャは，OS の機能とは独立した多くの多重プログラミング OS に共通に見られる手続層であり，具体的にはプロセッサ仮想化，メモリ仮想化，プロセス間通信機能によってプロセスとプロセス協調を具現化している．また OS の機能も機能表現層においてシステムプロセスとして実現される．こうした概念は DPS 設計者（システム核設計：DPS 環境操作メカニズムの提供）と応用システム設計者（応用システムの機能・ポリシーをプロセスレベルで実現）の役割を明確化する一つの基準を示すものである．

最後に DPS 構成技術は総合的視野に立ってハードウェア，ファームウェア，ソフトウェアの効果的な機能分担を図り，高水準アーキテクチャ（分散処理システム核）レベルで対応すべきであると思われる．

謝辞 本研究およびシステムの実働に御指導，御助力頂いた沖電気工業株式会社松下温氏，西垣秀樹氏および関係各位に深謝致します．また，システムの設計・開発に多大の御貢献を頂いた竹山明氏（現，富士通研），徳田英幸氏（現，Waterloo 大学）に紙面を借りて感謝致します．

参考文献

1) 相磯秀夫：ミニコンピュータコンプレックス，計測と制御，Vol. 14, No. 8, pp. 35-45 (1975).
2) 相磯秀夫：機能分散型システム，情報処理，Vol. 18, No. 4, pp. 325-333 (1977).
3) 相磯秀夫：マルチマイクロプロセッサ，信学誌，Vol. 61, No. 10, pp. 1093-1094 (1977).
4) Aiso, H., Kamibayashi, N., et al.: A Minicomputer Complex-KOCOS, Proc. 4 th Data Communication Symposium (1975).
5) Aiso, H., Tokuda, H., Kamibayashi, N., et al.: A System Software for KOCOS, Proc. IFIP TC-2 Working Conference, pp. 41-51 (1975).
6) 上林，徳田，阿多，西垣，相磯：ミニコンピュータ複合システム KOCOS のプロセス間通信機能，信学論 (D), Vol. 61, No. 3, pp. 186-193 (1978).
7) 上林，竹山，西垣，相磯：ミニコンピュータ複合システム KOCOS の分散バス制御方式と知的インタフェース，信学論 (D), Vol. 61, No. 11, pp. 842-849 (1978).
8) Kamibayashi, N., Akatuka, H., Aiso, H., et al.: Distributed Processing Oriented Interprocess Communication Facility for KOCOS, Proc. 3 th UJCC, pp. 80～85 (1978).
9) 上林，徳田：ミニコンピュータ複合システム KOCOS のプロセス間通信基本命令とシステム・ニュークリアス，情報処理，投稿予定.
10) 松下，西垣：複合ミニコンピュータ・システム KOCOS，沖電気研究開発 104 号，Vol. 44, No. 1, pp. 65-73 (1977).
11) 上林，赤塚，小川名，多々良，相磯：分散処理向きシステム・ニュークリアスとアーキテクチャの一考察，情報処理学会，ソフトウェア工学研究会資料，78-7 (1978).

（昭和 53 年 12 月 21 日受付）