2P−1　分散並列OS「Orion」の試作 システムの概要

岩嵜正明，吉澤聡，千葉寛之，宇都宮直樹，薗田浩二，山内雅彦
(株)日立製作所　中央研究所

1．はじめに

我々は分散環境上で並列処理機能を提供する分散並列オペレーティング・システムOrionの研究を進めている。最近の高性能RISCプロセッサや光ファイバの出現は、ネットワーク環境上での並列処理の可能性を示唆している。しかし、その実現に向けては以下の様な問題点の指摘がある。

- ●既存のsocket, RPC等のプロセス間通信のAPI（Application Program Interface）は、並列処理を指向した設計とはなっていない。
- ●汎用のOSでは、通信処理のオーバヘッドが大きく並列処理には向かない。

我々は分散並列処理の実現に向けて、二つの方向から取り組んでいる。第一は、ノード数やネットワーク性能に対して拡張性を持った並列処理指向APIの設計である。第二は、並列処理に適した通信プロトコルやその実装技術の検討である。現在、Unixワークステーション上でのOrionシステムのプロトタイプ開発[1-3]とその試用評価[4]、及び高速通信方式の検討[5]を行なっている。

本稿では、このプロトタイプで実現した機能、及びシステム構成の概要について説明する。

2．設計方針

現時点では、Orionシステムは科学技術系のデータ分割型並列処理を題材に、以下の前提で設計している。

- ●ハードウェア的にはバイトオーダの相違等の問題がないホモジニアスな分散メモリ・アーキテクチャである。
- ●アプリケーションは前処理部分と並列演算部分とに分離でき、前処理部分でプロセス数（並列度）が決定でき、並列演算部分での動的なプロセスの生成や移動等は発生しない。

具体的にはOrionシステムが提供するAPIの仕様を以下の方針で設計及び実装している。

- ●ノード数やネットワーク・トポロジが変わっても同一のプログラム・コードが利用できるAPI仕様とする。
- ●ノード間を渡るプロセス間通信は全て明示的なメッセージ通信によって行なうAPI仕様とする。
- ●全ノードで同一のプログラム・コードを実行するSPMD（Single Program Multiple Data stream）方式の並列処理に適合するAPI仕様とする。

3．機能概要

Orionシステムは上記の方針を具体化するために、位置透過性を備えるメッセージ通信機能を提供する。Orionシステムが提供するプロセス間通信のAPIは、ポートを用いたメッセージ通信に基づいており、最上位層では通信相手をグローバルなポート名だけで指定できる位置透過性を実現している。

この他にもOrionシステムは、他ノード上でプロセスを遠隔起動する機能を始めとして、並列処理に必要な種々の基本機能をC言語関数ライブラリとして提供する。表1にプロトタイプの並列処理ライブラリが提供する機能の一覧を示す。表1に示す各機能とシステム構成の関係とを図1に示す。図1の斜線を施した細長い長方形に付けた番号が、表1の右端の欄の番号に対応する。

表1．Orionシステムの機能

	分類		提供機能の内容	#
1	リモートプロセス制御	プロセス起動	自／他ノード上での子プロセス生成機能	7
		プロセス終了	子プロセス終了の親プロセスへの通知機能	
		プロセス同期	子プロセス終了を待つ機能	
		シグナル発行	親子プロセス間でのシグナル通知機能	
2	プロセス間通信	ポート生成・消去	ローカルポート／グローバルポートの生成と消去機能	
		ローカル通信	ローカルポートを用いるノード内プロセス間通信	6
		ロケーション指定の通信	ロケーションを指定したプロセス間通信機能	5
		OIDを用いる通信	OID（システム定義名）を用いた位置透過な通信機能	3
		UDNを用いる通信	UDN（ユーザ定義名）を用いた位置透過な通信機能	1
3	ネームサービス	OIDの生成	全ノードに渡って一意なOIDの生成	4
		OIDの登録・削除・検索	OID／位置情報対のロケーションテーブルへの登録など	
		UDNの登録・削除・検索	UDN／OID対のロケーションテーブルへの登録など	2
		一貫性制御	分散化したロケーションテーブルの一貫性保持（内部機能）	
4	Combuf通信		Combuf通信のAPIをエミュレーションにより実現	

Prototyping of Distributed-Parallel Operating System "Orion"
- System's Overview -

Masaaki IWASAKI, Satoshi YOSHIZAWA, Hiroyuki CHIBA, Naoki UTSUNOMIYA, Kouji SONODA, Masahiko YAMAUCHI
Central Research Laboratory, Hitachi, Ltd.

表１中のCombuf通信は、ユーザ空間とカーネル空間の間でのデータコピーを無くす高速ノード間通信インタフェースである。プロトタイプではそのAPIのみをエミュレートしている。

尚、Orionシステムは、NFSやX windowシステム等の既存のRPCに基づく分散処理機能と共存できる。

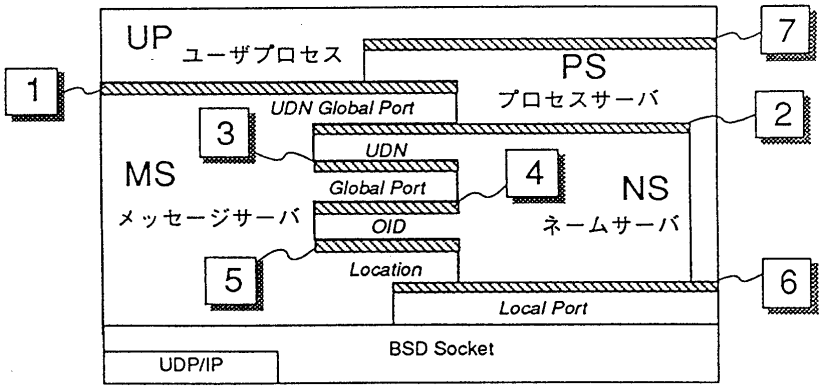

図１．Orionシステムの構成

４．構成概要

Orionシステムは、集中的な管理ノードを設けず、下記のサーバを対称に各ノードに配置する分散型構成を採用している。Orionシステムでは、ノード間を渡る一元的管理が必要なプロセス名やポート名等については、各ノードに常駐するネームサーバどうしがメッセージを交換して、システム全体に渡る一貫性を保証している。

- ●メッセージサーバ（MS）…通信相手のプロセスが自ノード内に存在するか、あるいは他ノード上に存在するかによって、メッセージを配送する。
- ●ネームサーバ（NS）…通信相手のプロセスのロケーション（位置情報）を管理する分散型ネームサーバ。名前（ポート名）で通信相手を問い合わせると、そのロケーションを返す。
- ●プロセスサーバ（PS）…他ノードからのプロセス起動要求を受け取り、自ノード内にプロセスを生成する等のプロセス制御機能を提供する。

プロトタイプでは、上記の他にCombuf通信のAPIを模擬するためのCombuf APIエミュレータもサーバとして実装している。

これらの各サーバは、システムプロセスとして常駐するサーバ本体部分、及びサーバ本体部分へのインタフェース機能を提供するライブラリ部分で構成する。このライブラリ部分はアプリケーション・プログラムにリンクされ、関数コール形式で渡される引き数をメッセージ形式に変換し、サーバのポートへ送信する機能を提供する。

図１に示す様にメッセージサーバとネームサーバは相互に依存する構造となっている。ネームサーバの下位層は、メッセージサーバが提供する通信相手の位置を明示指定する通信機能（５）を用いて、大域的にネームサーバ間の一貫性を維持し、OID（Object IDentifier）による位置透過なネーミング機能（４）を提供する。

メッセージサーバの上位層は、ネームサーバが提供するOIDによるネーミング機能（４）を用いて、位置透過なグローバル・ポートを用いたプロセス間通信機能（３）を実現する。ネームサーバはOIDによる位置透過なネーミング機能（４）の上位に、さらにUDN（User Defined Name）による位置透過なネーミング機能（２）を実現する。通常、ユーザ・アプリケーションは、このUDNを用いて通信相手のグローバル・ポートを指定するAPI（１）を利用する。

プロセスサーバは上述のメッセージサーバが提供するプロセス間通信機能を使って他ノード上のプロセスサーバと通信する。ノード内の各サーバどうしは、ローカル・ポートを用いたプロセス間通信機能（６）によって相互に通信する。また、ユーザプロセスとサーバとのノード内通信もローカル・ポートを用いて実現している。

プロトタイプは、標準的なUnixシステムコール及びライブラリのみを用いて実装しており、X/Open Portability Giudeの仕様に従うシステム上へは、容易に移植できる。ノード間のプロセス間通信はBSD系のSocketを用いて実装しており、UDP/IPプロトコルを使用している。

５．おわりに

プロトタイプ開発によって、ネットワーク接続されたUnixワークステーションを用いて、並列処理指向APIを検証できる環境が実現できた。今後、Orionシステムが提供する位置透過なメッセージ通信機能を基盤に、よりプログラミングが容易な並列処理環境の実現に向けて研究を進める予定である。

参考文献

[1] 宇都宮, 他, 分散並列OS「Orion」の試作－並列計算におけるネームサーバの課題とその解決, 情報処理学会第45回全国大会予稿集, 2P-02, 1992.

[2] 山内, 他, 分散並列OS「Orion」の試作－プロセスサーバの実装, 情報処理学会第45回全国大会予稿集, 2P-03, 1992

[3] 薗田, 他, 分散並列OS「Orion」の試作－メッセージサーバの実装, 情報処理学会第45回全国大会予稿集, 2P-04, 1992

[4] 吉澤, 他, 分散並列OS「Orion」の試作－システムの性能評価, 情報処理学会第45回全国大会予稿集, 2P-05, 1992

[5] 千葉, 他, 分散並列OS「Orion」の試作－高速通信機能の検討, 情報処理学会第45回全国大会予稿集, 2P-06, 1992.