3L-07

分散向け OS-Plan9 を用いた分散パイプシステム

中原　健志†　　紫合　治†
東京電機大学　情報環境学部†

１．はじめに

近年，分散処理環境が身近になりつつある．分散処理を実現する手法として，プログラム中にそれぞれのプロセス間通信に関する特殊な記述を行い実装することが多い．しかしこの手法では，プログラム構造の複雑さ，習得の難しさ，言語の制限などの問題点がある．ソフトウェア工学においてプログラム記述の複雑さ，制限を減らすことは重要な課題である．そこで本研究では，UNIX におけるパイプの概念を，コンピュータ内からネットワークを介した他のコンピュータへ拡張することを考えた．その中で分散システム向けに開発された Operating System（以下，OS と略）「Plan9」 [1]のアクセス透過性と位置透過性の機能を用いて実現する事を検討した．

２．分散向け OS-Plan9 について

Plan9 はベル研究所によって開発された分散向け OS である．特徴として，Plan9 ではネットワークを含むほぼ全てのコンピュータ資源をファイル資源に抽象化した．これにより様々なコンピュータ資源へのアクセスをファイル入出力で実現し，アクセス透過性を実現している．

また，名前空間を自由に編成可能であり，ネットワークを介した他のコンピュータ資源を自分の名前空間の好きな場所へつなげることが可能である．これにより位置透過性を実現している．

また，名前空間はプロセス毎に独立しており，編成した名前空間をほかのプロセスは見ることが出来ない．そのため編成による他のプロセスへの影響を無くし，かつ高いセキュリティ性を実現している．上記の特徴から分散システムの構築に適している．

Distributed Pipe System based on Distributed OS Plan9
†Takeshi Nakahara and † Osamu Shigo
School of Information Environment, Tokyo Denki University

３．システム概要

システム全体は端末（Term），認証サーバ（Auth srv），CPU サーバ（CPU srv），ファイルサーバ（File srv）の４つの機能から構成されている（図１）．このなかで「CPU サーバ」は計算処理機能のみを提供するサーバである．利用者は端末から CPU サーバへログインして計算処理を行わせることが可能である．この際，CPU サーバのファイルツリーに端末のファイルツリーがマウントされる（図２）．これにより CPU サーバ上で端末のファイルへ容易にアクセスできる．

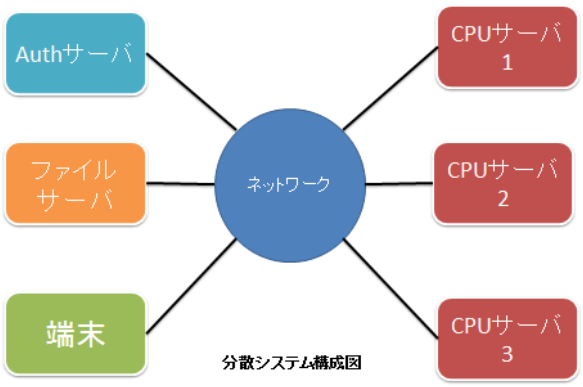

図１：システム構成図

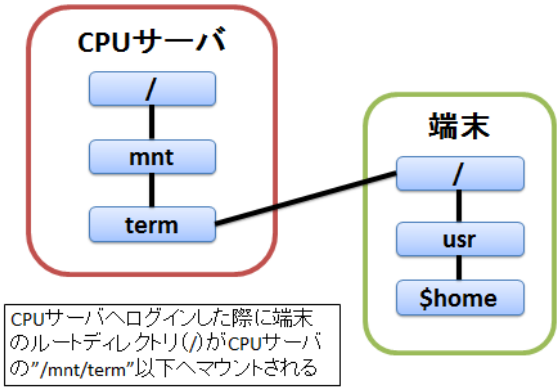

図２：CPU サーバと端末間のファイルツリー

４．分散パイプシステムの構成

本研究では分散処理の記述に UNIX で使われているパイプを応用したものを検討した．これはパイプを用いることでプログラミング言語による縛りをなくす，また記述が単純かつ分かり易く，習得が容易であると考えられたためである．

利用者は下記の記号を用いてプロセスの流れを記述する．

<並列シェル上の記号>

- | ：端末で実行
- || ：CPU サーバ上で実行
- **[ ]** ：接続されている全ての CPU サーバ上で指定されたプロセスが実行される
- **n[ ]** ：n 台の CPU サーバ上で指定されたプロセスが複数個同時に実行される

５．複数台の CPU サーバでの処理

同一のアプリケーションを複数台の CPU サーバで並列実行を行う．標準入力の改行毎にそれぞれの CPU サーバへ入力されたデータをラウンドロビンで渡していく．それぞれの CPU サーバで処理された結果は処理に使われている名前付きパイプの出力をパイプシステムが監視し，自動的にまとめられ，一つに出力される．結果をまとめる方法に「データの到着順（ [ ] ）」、「ラウンドロビン（ [ ]r ）」，「ソートマージ（ [ ]m ）」の３つを想定している．

６．処理の流れ

利用者は予め端末の特定のファイルツリー下に利用可能な CPU サーバのファイルツリーをマウントし，そのディレクトリを環境変数に登録する．利用者の記述内容を元に必要な数の名前付きパイプを生成する．また，それぞれ CPU サーバのホームディレクトリを端末のホームディレクトリへ名前空間を変更する．これにより CPU サーバ上で端末と同じディレクトリの読み書きが可能になる（図３）．そしてそれぞれのプロセスの入出力を名前付きパイプのものへ変更を行う（図４）．

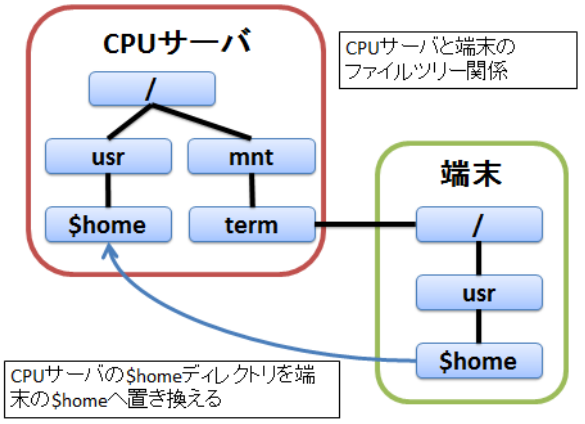

図３：CPU サーバと端末のファイルツリー関係

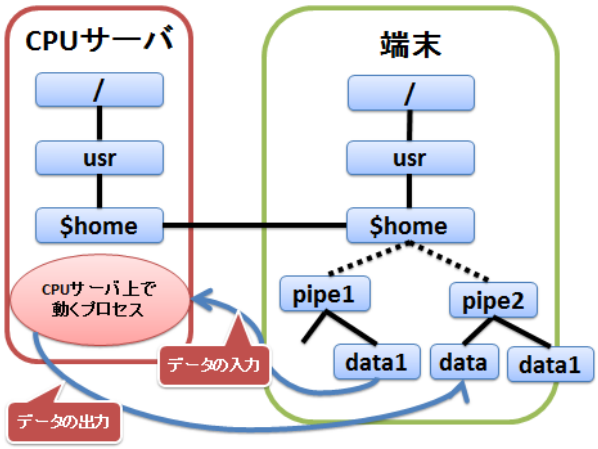

図４：CPU サーバ上で動くプロセスのデータ入出力

７．処理の例

単語の出現頻度表を作成する例

dps ‘word < in.txt | [sort]m | count || sort -r | report > out.txt’

プロセス「word」は端末上で実行され，文章を単語に分解・出力する．プロセス「sort」は複数台の CPU サーバ上で並列実行され，入力された単語を整列して，パイプへ出力する．パイプシステムはそれぞれのサーバからの出力を監視・ソートマージして一つにまとめ出力する．プロセス「count」は端末上で実行され，ソートされた単語の出現数を数え，「単語」と共に出力する．プロセス「sort –r」は１台の CPU サーバ上で実行され，入力された「出現数」と「単語」を出現数を元に降順に並べ，出力する．プロセス「report」は端末上で実行され，「単語」と「出現数」を整形して出力する．そして整形された出現頻度表は「out.txt」のファイルへ出力される．

８．まとめ

本研究では分散処理を実現する手法として UNIX などにおけるパイプの概念を応用した．その中で OS：Plan9 の持つ２つの透過性が本研究の目的を実装するのに適していることを示した．

現在，それぞれの CPU サーバ上で一つのプロセスを処理するところまでの実装を行った．今後は複数台の CPU サーバで同一のタスクを実行したときに複数の入出力を一つの入出力へまとめるプログラムの作成，また CPU サーバの負荷に応じて自動的に適切なサーバを選択する機能の実装，評価する．

参考文献

[1]： Alcatel-Lucent. Plan 9 from Bell Labs Fourth Edition. http://plan9.bell-labs.com/plan9/