

複数のオペレーティングシステム環境と ターゲットマシンリンクシステムの開発

三上 理*, 八木橋 信一*, 真鍋 和久**, 須堯 一志**

(*日本電気(株) C&Cシステムインタフェース技術本部, **日本電気技術情報システム開発(株))

本報告では、複数のオペレーティングシステムが共存し、かつ利用できる環境についての分類、検討を行い、その中の一例であるターゲットマシンリンクシステムについて述べる。このシステムは複数のオペレーティングシステムを単一のユーザインタフェースで利用でき、オペレーティングシステムへの依存度が小さいという特徴をもつ。ターゲットマシンリンクシステムの実現例として、NCI(New Command Interpreter)と呼ぶシステムを試作した。NCIはホストコンピュータ上のソフトウェア開発のため、ワークステーションとホストコンピュータを用いて連携作業を行うことを目的としたものである。連携作業はコーディング、実行、デバッグ工程の作業と限定し、その間、単一ユーザインタフェースでホストコンピュータとワークステーションの両方を操作できる。そのため、NCIはワークステーション、ホストコンピュータ間でのコマンド形式、コマンドレベルでのファイルアクセス形式の統一化を行う。また開発ターゲットとなるホストコンピュータへの適用性を高くするため、ホストコンピュータに依存する情報はシステムと独立に扱えるように工夫されている。

Development of a Target Machine Link Environment

Osamu MIKAMI*, Shin-ichi YAGIHASHI*, Kazuhisa MANABE** and Kazushi SUGYOU**

*C&C Systems Interface Engineering Laboratory, NEC Corporation,

**NEC Scientific Information System Development Corporation

14-22, Shibaura 4-chome, Minato-ku, Tokyo 108, Japan

Target Machine Link environment is one of multiple operating systems environments, which has characteristics of providing some facilities of non-native operating system, single man-machine interface and functions of multiple operating systems on many computers. A Target Machine Link environment called NCI(New Command Interpreter) system is developed. NCI system is designed for developing application software of host computer systems with the same commands as those of a personal workstation's operating system. A user can operate remote host computer systems as well as a local workstation with single man-machine interface. This system can be easily adapted to any host computers by replacing database which contains host computer dependent information.

1. はじめに

オペレーティングシステムの種類は、コンピュータの種類、用途によって様々である。代表的なものでは、ミニコンピュータ上のUNIX、パーソナルコンピュータ上のMS-DOSなどが挙げられる。通信技術の発達によって、このような多種多様なオペレーティングシステムを、手もとにあるコンピュータから利用することが可能になってきた。またオペレーティングシステムだけではなく、応用プログラムにも用途にしたがった様々な種類があり、1台のコンピュータ上で多数の応用プログラムを利用したいという要求も増加している。そのため、複数のオペレーティングシステムを1台のコンピュータ上に実現することも重要な課題となっている。

本稿では、このような複数のオペレーティングシステムを共存させ、利用できるような環境について、形態分類、検討を行う。またその分類の一形態であるターゲットマシンリンクシステムという利用環境について報告する。ターゲットマシンリンクシステムは、ユーザインタフェースを統一するオペレーティングシステムの共存形態であり、オペレーティングシステムへの依存性が小さく、既存のオペレーティングシステムに容易に適用できる特徴がある。さらにターゲットマシンリンクシステムの実現システムとして、NCI(New Command Interpreter)と呼ぶシステムを試作した。このシステムはワークステーションを用いて、ホストコンピュータ(以下、ホストと省略する)のソフトウェア開発を支援するシステムである。NCIによって、ワークステーションとホストのユーザインタフェースが統一され、かつ2つのコンピュータを連携させながら、開発作業を進めることができる。

2. 複数オペレーティングシステムの共存^[1,2,3,4,5,6]

2.1 目的

ユーザが複数のオペレーティングシステムを利用する目的は、大きく以下の2点が考えられる。

- (1) 他のコンピュータの機能を利用する。
- (2) 異なったオペレーティングシステム用の応用プログラムを1台のコンピュータで利用する。

(1)は、1台のコンピュータでは負担の大きな処理で、他のコンピュータを利用して負担を軽減させる、もしくは特定の処理に関して専用のコンピュータを設け、その機能を用いるといった目的である。一方、(2)の場合は、1台のコンピュータ上で種々の作業を行うため、多数の応用プログラムを実行させたいという目的である。

2.2 共存形態

共存形態の分類では、複数のオペレーティングシステムが物理的にどのように分散しているかと

いうこと、ユーザにどのような機能を提供するかということ、そしてユーザがどのようなインタフェースで操作できるかという3点に注目する。

まず、オペレーティングシステムの物理的分散について考える。その形態には次に示す3通りの場合がある。

- (1) 複数台のコンピュータ上で稼働しているオペレーティングシステムを利用する。
 - (a) 水平分散型---分散ファイル、リモートログインなど
 - (b) 垂直分散型(ホスト+パーソナルコンピュータなど)

---マイクロメインフレームリンク

- (2) 1台のコンピュータ上で複数のオペレーティングシステムを稼働させ、利用する。---マルチオペレーティングシステムなど

次に複数のオペレーティングシステムによって提供される機能について考える。特にネイティブでないオペレーティングシステム(以下、ゲストオペレーティングシステムと呼ぶ)によって提供される機能に注目する。

一般にオペレーティングシステムの提供する機能は以下の3つに分類できる。

- (1) ジョブ、プロセスの管理
- (2) データ(ファイル)の管理
- (3) 入出力(周辺機器)の管理

上記3種類の機能のうち、ゲストオペレーティングシステムがどの機能を提供するかに注目すると、次のような形態に分類できる。

① 3つの機能のうちいずれかを提供

- ①-1 ネイティブ機能+(1)
スーパーコンピュータとフロントエンドコンピュータの関係など
- ①-2 ネイティブ機能+(2)
分散ファイル、分散データベース
- ①-3 ネイティブ機能+(3)
リモートプリント

② 3つの機能をすべて提供

- ②-1 ゲスト機能のみ
マイクロメインフレームリンク
リモートログイン(マルチウィンドウ環境での利用を除く)
- ②-2 ネイティブとゲストの両機能
マルチオペレーティングシステム

次に操作方法についての分類を考える。まず考えられるのが、明示的な操作と非明示的な操作である。前者は、ゲストオペレーティングシステムをユーザに意識させ、そのオペレーティングシステムの方法で操作する。それに対し後者は、ユーザにゲストオペレーティングシステムを意識させず、統一的に操作できる方法である。

明示的な操作方法には、

- (1) ゲストオペレーティングシステムすべてをユーザに見せ、操作させる。たとえば、リ

モートログインやマルチオペレーティングシステムにこのタイプが多い。

(2) ゲストオペレーティングシステムのコマンドのみを投入する。たとえば、リモートバッチなど

(3) 部分的に指定を行う。たとえば、マシン名など。リモートプリントがこのタイプである。

などがある。

また非明示的な方法は、単一の操作方法を用いるが、その単一化の方法に

(1) どちらかのオペレーティングシステムの操作方法を用いる。

(2) まったく別の操作方法を用いる。マイクロメインフレームリンクがこの方法である。などがある。このような操作方は明示、非明示を共存させて用いることもある。

表2.1に代表的な共存システムをあげ、その形態分類を示す。図2.1には共存システムの構成を示

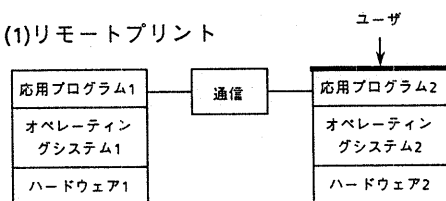
表2.1 オペレーティングシステムの共存システム

	分散処理 (リモートプリントなど)	分散処理 (分散ファイル)	分散処理 (リモートログインなど)
目的	機能分散	機能分散	機能分散
分散形態	水平 複数機械	水平 複数機械	水平 複数機械
提供機能	部分機能	部分機能	全体機能

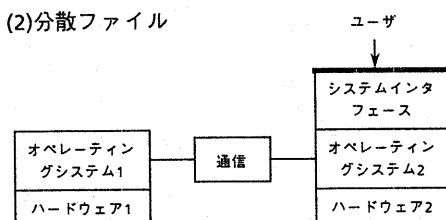
	マイクロメインフレームリンク	仮想機械 (マルチOSなど)	①ターゲットマシンリンク
目的	機能分散	アプリケーションの利用	機能分散
分散形態	垂直 ホスト, 端末	単一機械	水平 複数機械
提供機能	全体機能	全体機能	部分機能

す。図2.1中、(1)はリモートプリントなどにみられる構成でリモートコンピュータの単一機能を利用することを目的としたものである。ユーザからはローカルなコンピュータのアプリケーションインタフェースしかみることができない。(2)は分散ファイルなどの構成でユーザはローカルコンピュータのライブラリやコマンドインタフェースを通してリモートコンピュータの機能を利用できる。そのとき操作は非明示的なユーザインタフェースを介することが多い。(3)はリモートログインなどの構成で、両方のコンピュータのアプリケーションを各アプリケーションのユーザインタ

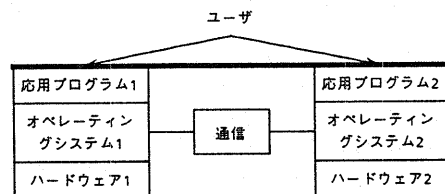
(1) リモートプリント



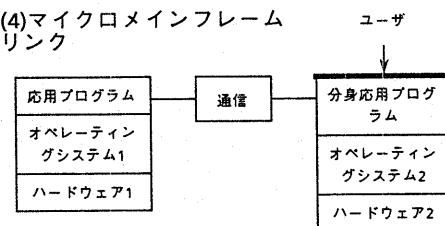
(2) 分散ファイル



(3) リモートログイン



(4) マイクロメインフレームリンク



(5) 仮想機械

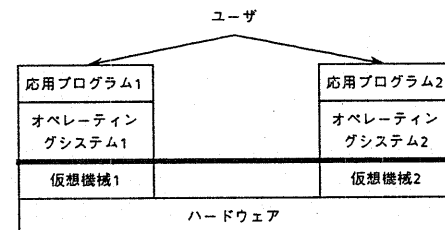


図2.1 共存システムの構成

フェースによって個別使用できる。(4)はマイクロメインフレームリンクの構成である。リモートコンピュータ(ホストコンピュータ)上のアプリケーションの分身的な役割をするアプリケーションをローカルコンピュータ上で利用する。したがってアプリケーションによって機能は限定される。垂直複数コンピュータの分散形態はこの構成が多い。(5)は

マルチオペレーティングシステムなどの構成で、仮想機械によってハードウェアレベルからのインタフェースを提供できる。

2.3 操作性を重視した共存方法

図2.2に示すようにオペレーティングシステムの

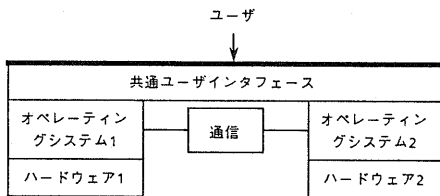


図2.2 共通利用型分散処理

上層にさらに共通ユーザインタフェースを提供する機能を加えた形態(共通利用型分散処理)を新たに考える。またその形態分類を表2.1の①に示す。この形態を用いると、ローカルとリモート両方のコンピュータ上のオペレーティングシステムが利用可能である。さらに、オペレーティングシステムと共通ユーザインタフェース間のインタフェースをうまく設計すれば、オペレーティングシステムに関係なく、かつ共通のユーザインタフェース(非明示的操作)を用いて両方のオペレーティングシステムを利用できる。したがって、異なるオペレーティングシステムを用いる場合にも、ユーザの負担が少なく、かつ再教育などの必要が少なくなるという利点がある。一方、ユーザインタフェースはオペレーティングシステムによって多種多様であり、統一インタフェースを提供する機能というのはかなり大がかりなものとなり、実用上問題である。しかし、利用目的や提供機能を限定することによって、この問題はある程度解決できると考える。以下このシステムの形態をターゲットマシンリンクと呼ぶ。

3. ターゲットマシンリンクシステム

3.1 目標^[10,11,12,13]

実現するターゲットマシンリンクシステムの用途はホスト用のソフトウェア開発と限定する。このシステムにより、ワークステーションとホスト間で連携作業を行いながら、ソフトウェア開発を進めることが可能である。すなわち、ユーザは図3.1に示すようにホストでの操作を意識せずに、単一の操作方法でソフトウェアの開発作業を行える。システムはすべてワークステーション上に実現し、ホストには一切手を加えない。したがって、ホストの種類に関係なく、このシステムは利用可能である。

連携作業中、ワークステーション上では、おもにソースプログラムを取り扱う作業、すなわち

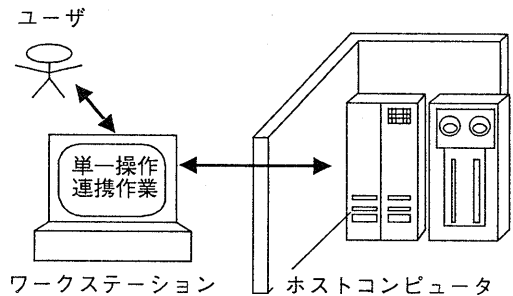


図3.1 ターゲットマシンリンクシステム

ファイルの作成や変更、管理など行う。ホスト上での作業は次のものである。

(1) コンパイラ、リンカなどを起動させる。

(2) プログラムを起動させる。

(3) 実行結果を画面やファイルに出力する。

このことより、作業は次の流れで行われる。

(1) ワークステーション上でのコーディング

(2) ホスト上でのコンパイルやリンク、実行

(3) 実行結果やコンパイル時のメッセージのワークステーションへの取り込みとそれを参照しながらのデバッグ作業、再び(2)へ

したがって、オペレーティングシステムの連携機能としてワークステーション上に実現するのは、次のとおりである。

(1) ジョブ、プロセスなどの起動、操作

ホストのオペレーティングシステムへコマンドを投入し、実行を行う。そのためコマンドの形式を統一する機能が必要である。

(2) ファイルのアクセス、操作、転送など

ファイルのアクセス、操作、転送もコマンドレベルで統一的なインタフェースを提供する。すなわちコマンドレベルでのファイルの共有や転送の機能である。

(3) 結果の出力

ホストからのメッセージをワークステーションの画面上や周辺装置(ディスク、プリンタ)に出力する機能である。

3.2 実現方式^[7,8,9,14]

ワークステーション上に試作したターゲットマシンリンクシステムをNCI(New Coammand Interpreter)と呼ぶ。NCIはユーザからのコマンドを解釈するインタプリタで、かつ前節で述べた(1)、(2)、(3)の機能を有している。ワークステーションのオペレーティングシステムをUNIXとし、ホストのオペレーティングシステムの種類は問わず、適用性を大きくする。

3.2.1 基本機能

NCIは、以下の3つの機能を持つ。

(1) コマンドの形式を統一する。

コマンドはワークステーション形式(UNIX形

式)で入力する。たとえばワークステーション上のファイルを消去するのも、ホスト上のファイルを消去するのも同じワークステーションのrmコマンドを用いることができる。そのため、ホストへ投入するコマンドはワークステーション形式からホスト形式へ変換を行う。たとえば、ホストのオペレーティングシステムとしてACOS4を例にとると、ファイルを消去するUNIXコマンド

```
% rm file名
はACOS4のコマンド
DELETE file名
```

というコマンドに変換される。

(2)ファイルの共有化を行う。

ファイルシステムはオペレーティングシステムによって異なる。たとえば、UNIXにおけるディレクトリ構造とACOS4におけるライブラリ構造を考える。

```
rm /usr/cmn/int/mikami/tmp/testfile
```

というUNIXコマンドの実行する場合、下線部のディレクトリによるファイルアクセスを

```
DELETE TMP(FILE) SL
```

というACOS4の下線部のライブラリによるファイルアクセス形式に変換する。

(3)ホストからのメッセージを画面上やファイルに出力する。

ホストと通信を行い、メッセージをワークステーションの画面に出力する。ファイルに取り込むために、UNIXのリダイレクションの機能を実現する。これはコマンド変換の機能を利用して実現する。

3.2.2 ホストとの結合

ホストのファイルシステムをUNIXのディレクトリに対応づけて管理する。対応づけられたディレクトリをホストディレクトリと呼ぶ。ホストディレクトリ以下のファイルは、NCIのシステム内で仮想的に管理し、そのファイルの実体はホスト上にある。

またユーザから入力されたコマンドはすべてワークステーション形式なので、ワークステーションに対するコマンドなのか、ホストに対するものなのか決定しなければならない。ホストディレクトリはファイル管理だけでなく、この決定にも利用できる。すなわちホストディレクトリ以下のファイル进行操作するコマンドはホストに対するコマンドであると決定できる。一方、ホストディレクトリ以下にないファイル进行操作するコマンドはワークステーションに対するコマンドであると判断する。ワークステーションのファイルが図3.2に示すようなディレクトリ構造である場合を例として示す。ホストディレクトリ/usr/cmn/ACOS(図中(1))以下のファイルを含む操作を行うならば、それはホストでの操作とみなす。つまり図中(2)のファイルtest.fに対して

```
% rm test.f
```

というコマンドを実行すればそれはホスト上の

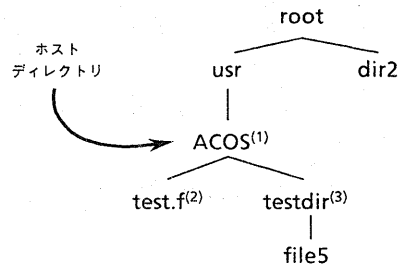


図3.2 NCIのファイル構造

ファイルを消去したことになる。同様に現在の作業ディレクトリが/usr/cmn/ACOS/testdirのとき(図中(3))、

```
% ls -la
```

というコマンドはホスト上のファイル名を表示するという操作に対応する。

3.2.3 機能構成

図3.3にNCIの機能構成を示す。NCIはNCI制御、

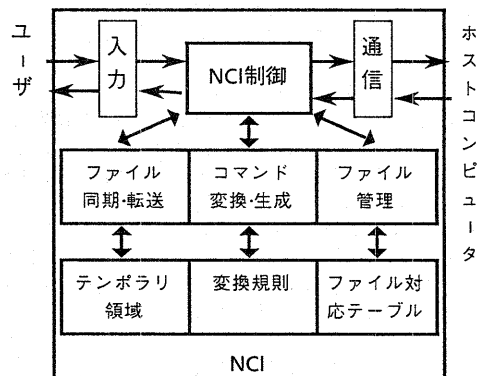


図3.3 NCIの機能構成

コマンド変換、ファイルシステム管理、及びファイル同期の各機能より構成されている。さらにユーザとのコネクションを実現する入力、及びホストとの通信を行う機能を持つ。図中の変換規則とファイル対応テーブルの部分はホストに依存する部分であり、システムとは独立に取り扱えるようにする。

3.2.4 各機能の説明

(1)NCI制御

入力されたコマンドをどちらのコンピュータに処理させるかを判断する。UNIXで処理するならば、そのままコマンドをシェルへ転送する。ホストで処理を行うならば、コマンド変換の処理をした後、ホストへ転送する。ホストからのメッセージを受けた場合には、そのままワークステーションの画面に出力する。

(2)コマンド変換・変換規則

コマンド変換を行う機能であり、**fix**^[15]と呼ぶフォーマット変換記述用の言語を用いて行う。例として先の

rm ファイル名

というコマンドを入力した場合を示す。ホストのファイル消去コマンドは

DELETE ファイル名

である。図3.4に**fix**による変換の記述を示す。**fix**は

原構文

(1)**rm** : RM file ;

(2)file : SFILE | OFILE | DIRECTORY ;

目的構文

arm(rm) :

(3)SetCommand("DELETE")

(4)SetCommand(GetHostFile(.file))

図3.4 コマンド変換記述

入力フォーマットのシンタックスを表現した原構文と変換や出力フォーマット間の対応関係を表した目的構文の2つの構文規則より構成される。原構文によって入力されたコマンドのシンタックスを解析し、目的構文によって対応するフォーマットに変換を行う。

rm ファイル名

というコマンドは規則(1)と(2)によって解析され、(3)によってまず**DELETE**というコマンドに変換される。そして(4)によってファイル名が変換され、

DELETE ファイル名

というコマンドが生成される。**fix**は、このような2つの構文規則を、c言語のプログラムに変換する。コマンド変換は、**fix**によって生成されたcプログラムによって行う(図3.5)。

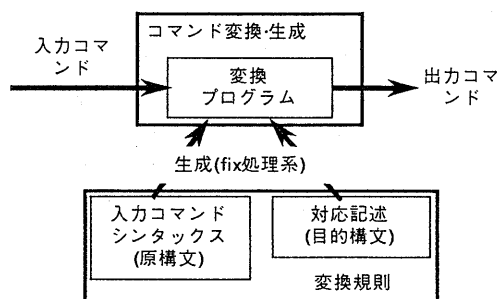


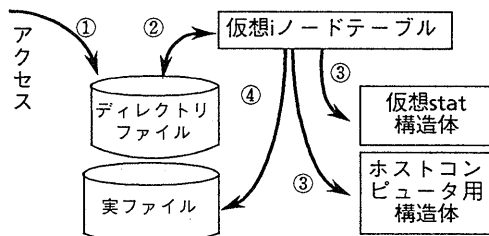
図3.5 コマンド変換機能

ここに示した方式のようにホスト依存の部分を機能的に分離すれば、保守が容易となり、種々のホストへの適用性が高くなる。

(3)仮想ファイル管理

ファイル管理はホストとワークステーション間でのファイル共有を実現する機能である。した

がってホストディレクトリ以下の仮想ファイルシステムを管理する。図3.6に示すように、仮想ファイルシステムは、UNIXファイルシステムと同様、仮想iノードテーブル及び仮想stat構造体よりなる。さらにファイル名変換等の処理が必要であり、ホストのファイルシステム用の情報を記述したホストコンピュータ用構造体を持つ(ファイル対応テーブル)。仮想ファイルのアクセス手順は以下



のとおりである(図3.6中の番号は以下の動作の番号に対応する)。

- ① /hostディレクトリのディレクトリファイルの中からsource.cを探索し、その仮想iノード番号を取り出す。
- ② 取り出した仮想iノード番号にしたがって仮想iノードテーブルを引く。その中から仮想stat構造体及びホストコンピュータ用構造体エントリを取り出す。
- ③ ②において取り出したエントリにしたがって仮想stat構造体及びホストコンピュータ用構造体を参照し、アクセス権などの情報をチェックする。アクセス可能であれば、ホストコンピュータ用構造体よりホストでのファイル名を取り出す。
- ④ ファイルがワークステーション上にあれば、実ファイルのアクセスを行う。
- ⑤ ファイルがワークステーション上になければ、ホストよりテンポラリ領域に転送してアクセスする。

(4)ファイル同期・転送

ファイル同期は、キャッシュメモリ的な領域(テンポラリ領域)を利用し、ファイルアクセスのたびにホストからの転送を行うことを避けるための機能である。すなわち一度ホストのファイルをアクセスすれば、ワークステーション上のテンポラリ領域にそのファイルを取り込み、それ以降はテンポラリ領域中のファイルにアクセスする。

この方法の場合、ホストとテンポラリ領域中のファイルの内容が異なる可能性がある。たとえば、エディタを用いてテンポラリ領域中のファイルをローカルに変更したような場合である。このため、ホストとテンポラリ領域中のファイルの内容が異なるか否かをチェックし、ホスト上のファイルと内容を統一する必要がある。

図3.7にファイル同期・転送の機能構成を示す。転

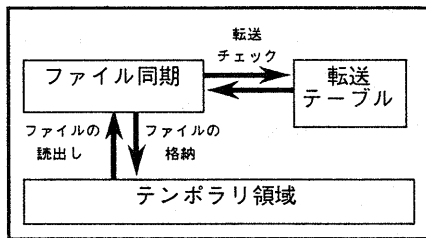


図3.7 ファイル同期・転送機能

送テーブルは、ワークステーション上でテンポラリ領域のファイルを変更したかしないかを記述するテーブルである。ワークステーション側でファイル内容を変更していれば、ホストでそのファイルを使用する時に転送しなければならない。逆にワークステーション上でファイルの内容を変更していなければ、ホストへの転送の必要はない。

ファイル転送はホスト上でファイルが必要な場

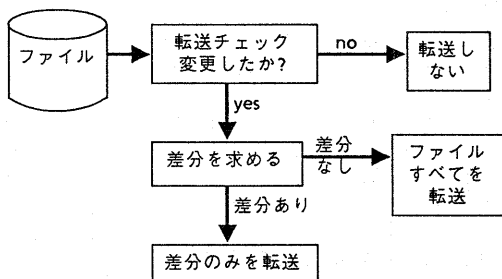


図3.8 ファイル転送

合にのみ行われる(オンデマンド転送)。たとえば、ソースプログラムをホストでコンパイルする場合などである。さらに転送する場合にも、なるべく無駄をさけるため、変更した箇所のみを転送する方式も実現している(ファイルの差分転送)。

ファイル転送のアルゴリズムを図3.10に示す。試作システムでは、ホストとワークステーション間を無手順回線で接続している。したがって、ファイル転送はホストのエディタを介してファイルにデータを入力する方式を採用している。差分転送はUNIXのdiffというコマンドの機能を用いて行っている。diffコマンドによって、変更前のファイルを変更後のファイルに一致させるedエディタコマンドが生成される。生成されたコマンドをコマンド変換機能によってホストのエディタコマンドに変換し、そのコマンドを転送する方法を用いている。

3.3 試作システム

以上の方針で、システムを試作した。試作に用いたコンピュータは次のとおりである。

(1)ホストコンピュータ

ACOS430、オペレーティングシステムはACOS4/AVP

(2)ワークステーション

SUN-3/160、オペレーティングシステムはUNIX4.2bsdおよびPC-9801VX41、オペレーティングシステムはPC-UX/V

図3.9にPC-9801VX上の試作システムでの実行結果の一部を示す。

3.4 今後の課題

現在のNCIの課題点を以下に示す。

(1) ホストとの通信方法

試作したシステムのコマンド実行時間を測定した結果、ホストとのインタラクション(コマンドやファイルの転送)を伴うコマンドが特に処理時間を要することがわかった。特にファイル転送には、たとえば5000バイトのファイルで1分以上も時間を要する。これは差分転送を用いることで、性能向上を実現できた。しかし、UNIXのrcpなどネットワーク上のプロトコルを用いた場合に比較するとかなりの差が見られる。このように高速通信を用いたホストとのインタラクションを導入しなければならない。

(2) テストのためのホスト用の応用プログラムの実行

現在システムでは、ホストからのメッセージは文字列に限られている。しかし、応用プログラムを実行させるには、グラフィックスなどの画面出力形式をサポートしなければならない。

(3) デバッグ機能のサポート

現在はサポートしていないデバッグを、どのようなユーザインタフェースでNCIに取り込むかという問題である。

(4) ワークステーションをネットワークで接続する。

このときのワークステーションとホスト間の、およびワークステーションとワークステーション間の連携機能について検討、実現する。

4. まとめ

本稿では、オペレーティングシステムの共存環境の分類、検討と、あらたな形態であるターゲットマシンリンクシステムについて述べた。形態分類には、物理的な分散、提供機能、ユーザインタフェースなどが考えられ、それぞれ共存環境の利用目的などにあったものを用いることが望ましい。またターゲットマシンリンクシステムには、複数のコンピュータ間のユーザインタフェースを統一し、かつ利用するオペレーティングシステムへの依存性が少ないという特徴がある。さらに

```

NCI)pwd
/usr/cmn/int/sugyou
NCI)cd ACOS          :ホストディレクトリへ移動
NCI)pwd
/usr/cmn/int/sugyou/ACOS
NCI)cp ../f77src/magic.f mag.f
NCI)ls -al
drwxr-xr-x k-sugyou int      52 Sep 1 11:47 .
drwxr-xr-x k-sugyou int      52 Sep 1 11:47 ..
-rw-r--r-- k-sugyou int      749 Sep 1 11:47 mag.f
NCI)cat mag.f          :ファイルをコピー
      DIMENSION M(100,100)
      INTEGER S1(100),S2(100),S3,S4

110  FORMAT(IH,102I6)
      STOP
      END
NCI)f77 mag.f          :ホスト上でコンパイルとリンク
FORTRAN 77 ENTERED
PROGRAM : MAG_F
NO ERROR
OBJECT PROGRAM PRODUCED
VER. R2.31 STLNK REV. 178 USER LM : AOUT REV.000
AN EXECUTABLE LOAD MODULE HAS BEEN PRODUCED
NO ERRORS DETECTED
NCI)ls
a.out
mag.f
NCI)a.out              :実行
      SIZE = 5

      17      24      1      8      15      65
      23      5      7      14     16      65
      4      6      13     20     22      65
      10     12     19     21      3      65
      11     18     25      2      9      65
      65     65     65     65     65     65
NCI)vi mag.f          :ファイルの変更
.....
NCI)f77 mag.f
FORTRAN 77 ENTERED
PROGRAM : MAG_F
.....
AN EXECUTABLE LOAD MODULE HAS BEEN PRODUCED
NO ERRORS DETECTED
NCI)a.out
      SIZE = 3

      8      1      6      15
      3      5      7      15
      4      9      2      15
      15     15     15     15
NCI)mkdir testd       :新しいディレクトリを作る
NCI)ls -l
-rw-r--r-- k-sugyou int      11 Sep 1 11:52 a.out
-rw-r--r-- k-sugyou int      749 Sep 1 11:49 mag.f
drwxr-xr-x k-sugyou int      52 Sep 1 11:53 testd
NCI)cd testd
NCI)pwd
/usr/cmn/int/sugyou/ACOS/testd
NCI)cp ../mag.f mag.f
NCI)f77 mag.f
FORTRAN 77 ENTERED
.....
NO ERRORS DETECTED
NCI)a.out > out.d      :ホストからの応答をファイルへおとす
NCI)ls -l
-rw-r--r-- k-sugyou int      11 Sep 1 11:54 a.out
-rw-r--r-- k-sugyou int      749 Sep 1 11:54 mag.f
-rw-r--r-- k-sugyou int      270 Sep 1 11:55 out.d
NCI)cat out.d
      SIZE = 3

      8      1      6      15
      3      5      7      15
      4      9      2      15
      15     15     15     15
NCI)

```

図3.9 実行結果

ターゲットマシンリンクシステムのひとつの実現例であるNCIの実現方式、試作、課題について報告した。NCIは

- (1) ホストのソフトウェア開発を利用目的としている。
- (2) ワークステーション形式のユーザインタフェースで作業を行える。
- (3) ホストとワークステーションとの連携作業を考慮し、そのための機能を実現している。
- (4) 対象としているホストへの依存性が小さい。

などの特徴をもつ。今後は、対象ホストを拡大して、さらに評価を進めたいと、問題点の解決を図っていく予定である。

参考文献

- [1] Bob Lyon and Gary Sager : "Overview of the SUN Network File System", Proc. of USENIX Conference & Exhibition, pp.117-124 (1985).
- [2] Systems Application Architecture, IBM, p.61, (1987).
- [3] 高木 他: 高機能インタフェースの現状と展望, マルチメディアと分散処理研究会資料, No.32, pp.1-6(1987).
- [4] 黒沢 他: 日本語PC/VM BOND; マイクロ・メインフレーム結合の実現例, マルチメディアと分散処理研究会資料, No.32, pp.7-14(1987).
- [5] 笹部 他: 日立におけるマイクロメインフレーム結合の設計思想, マルチメディアと分散処理研究会資料, No.32, pp.21-26(1987).
- [6] SunIPC User's Guide, Sun Microsystems (1986).
- [7] UNIX User's Manual Reference Guide(1986).
- [8] UNIX Programmer's Manual Reference Guide (1986).
- [9] ACOS-4/AVP PWSS コマンド説明書<非メニュー編>, NEC (1986).
- [10] 遠城: 異種OSにおけるUNIXインタフェースの実現法, 33回情処全大, pp.271-272(1986).
- [11] 前田 他: 複数OS動作環境での統合コマンド・インタフェースの一構成法, 33回情処全大, pp.317-318(1986).
- [12] 樽矢 他: WS連携の方式に関する一考察, 34回情処全大, pp.359-360 (1987).
- [13] BOEHM, B. W. : Software Engineering, IEEE TRANSACTION on Computers, Vol. c-25, No. 12 (1976).
- [14] 三上 他: マイクロメインフレームリンクによるソフトウェア開発環境の実現方式, 34回情処全大, pp.1171-1172(1987).
- [15] 伊藤, 大竹: プロトコル処理技術, NEC技報, Vol.40, No.1, pp.58-64 (1987).