4U-10　UNIXワークステーションにおけるネットワークターミナルシステム

唐住尚司　伊藤敏美　長友典昭
(株)東芝　青梅工場

1 はじめに

一般的に、ネットワークシステムを構築するうえで各資源の共有化をどこまで設計するかが大きな問題となる。

今回，著者らはUNIXワークステーションによるネットワークシステムに端末とCPUの共有化をねらったネットワークターミナルシステムを開発したのでその概要を報告する。

本稿では、UNIXワークステーションにおけるNTS（ネットワークターミナルシステム）の基本機能および実現方法について紹介する。

NTS（ネットワークターミナルシステム）は、次の2つの機能より構成されている。

1　NTS（ネットワークターミナルサービス）機能
ネットワーク上のリモート端末を自端末として収容するための機能である。本機能により，リモート端末から自システムのタイムシュアリングサービスを利用できる。

2　NTA（ネットワークターミナルアクセス）機能
自システムに接続されているローカル端末を、ネットワーク上の他システムの端末として使用するための機能である。本機能により、物理接続形態によらず必要に応じて1つの端末から複数のシステムを利用できる。

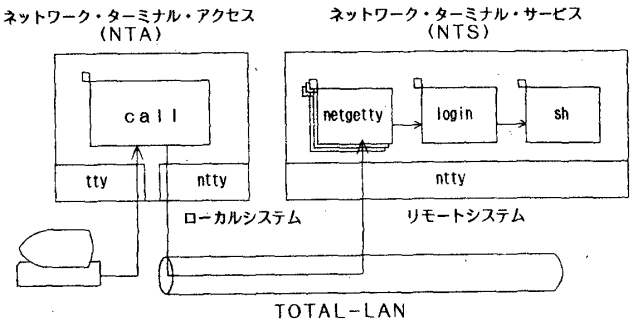

図1　ネットワーク・ターミナル・システム図

今回、NTS（ネットワークターミナルシステム）を構成するための要素としてつぎの3つを、TOTAL-LAN上に開発した。

ここで、TOTAL-LANとは、IEEE802．3準拠の10Mbps（ビット／秒）と高速なBUS型LANである。

（1）　ntty
TOTAL-LANへのアクセスを行うための仮想ttyデバイスである。

（2）　netgetty
ttyまたはnttyデバイスを介して、リモート端末を自システムの端末として接続するための回線属性の調整、端末属性の設定、およびlogin処理をするプロセスである。

（3）　call
ttyデバイスまたはnttyデバイスを介して、自端末を他システムの端末として使用するための無手順端末エミュレータである。

2 NTTY

nttyはTOTAL-LANにアクセスを行なう仮想デバイスである。nttyデバイスはTOTAL-LANにアクセスするためのプロトコルを全てドライバ内で実現している。nttyデバイスはttyデバイスと同様キャラクタ単位の入出力を基本としており、直前の1文字を消去するERASE文字処理、現ラインを消去するKILL文字処理などの文字編集機能や、端末から、かな漢字変換キーを押せばかな文字が漢字に変換されて端末の画面に表示されるかな漢字変換機能などを利用することができる。

nttyデバイスへのアクセスは、通常のopen、close、read、write、及びioctlシステムコールによって行われる。

これらのシステムコールは、ttyデバイスに対するそれと同様に行なう事が可能であり、アプリケーションプログラムはttyデバイスとnttyデバイスとの差異を特に意識する必要がない。

しかし、nttyデバイスを用いてTOTAL-LAN上のプロセスと能動的にコネクションを設定（発呼）する場合には、open時に接続相手の指定をする必要がある。

一方、受動的にコネクションを設定（着呼）する場合にはnttyドライバによってネットワークプロセス名が自動的に割り付けられる（端末グループの親プロセスの場合は“NTS”が、それ以外の場合はデバイス名がネットワークプロセス名となる）。

なお、このデフォルトのネットワークプロセス名はioctlで変更可能である。

＊UNIXはAT&Tが開発しライセンスしているOSです．

Network terminal system for UNIX work station
Shoji KARASUMI, Toshimi ITOH, Noriaki NAGATOMO
TOSHIBA Corp. OME Works

3 NETGETTY

ｎｅｔｇｅｔｔｙは、ＴＯＴＡＬ－ＬＡＮ上でリモート端末をローカルシステムの端末として収容するための回線属性の設定及び、端末形式を調整するプロセスである。

リモート端末を、公衆網、ＬＡＮなどでのネットワークを介して接続する場合、１つの物理／論理回線に異なる属性の端末が接続される可能性がある。

従来のｇｅｔｔｙでは、端末属性の設定はｌｏｇｉｎ処理後、ｓｈのＴＥＲＭ変数としてユーザーが設定しなければならない。このため、ｌｏｇｉｎ処理時にはどんな端末が接続されているか解らなかった。

これは、ｌｏｇｉｎ処理後、ただちに端末タイプに依存するプロセスが走る場合予期せぬ画面出力となるなどの不具合が生じる。

これに対して、ｎｅｔｇｅｔｔｙは接続してきたリモート端末に対してｌｏｇｉｎ名入力前に、端末タイプの問い合わせを行なう。ここで入力された端末タイプは、環境変数ＴＥＲＭにセットされｌｏｇｉｎプロセス、ｓｈと引きつがれていく。

つまり，ｌｏｇｉｎ処理時にはどんな端末が接続されているか認識出来、ｇｅｔｔｙの時の様な不具合はない。

4 CALL

ｃａｌｌは、自システムの端末をネットワーク上の他のシステムの端末とするコマンドである。

接続形態としては次ぎの形態がある。

- ◎ ＲＳ－２３２Ｃ直結ケーブルによる直接接続
- ◎ 音響カプラや公衆全二重モデムによる公衆接続
- ◎ ＬＡＮ接続

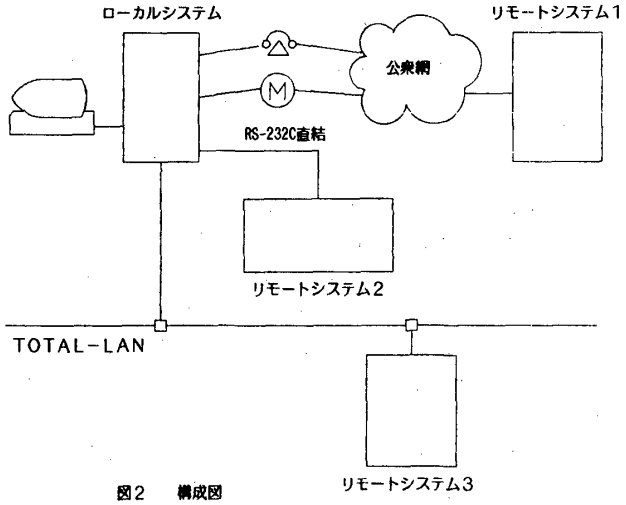

図2　構成図

従来のＵＮＩＸには、ｃａｌｌと同様に、自システムの端末を他のシステムの端末にするｃｕコマンドがある。

しかし、ｃａｌｌはこのｃｕとの大きな違いとしてｃｕにはない次の様な機能を持っている。

ｃａｌｌでは、リモートシステムとログオン／ログオフする為の手順や、ファイル転送を行なう為の手順を、外部ファイル＜手順ファイル＞に記述しておく事で、それらを自動的に実行出来る。

つまり、リモートシステムごとに、各種手順ファイルを作成しておくことにより、ＵＮＩＸ以外のシステムとも簡単に自動ログオン、自動ログオフ，あるいはファイル転送などが可能である。

ｃａｌｌは、引数として接続したいリモートシステム名を指定することで、回線属性などの情報をデータベースファイルより取り出してリモートシステムと接続する。

ｃａｌｌは、接続処理後、自端末のキーボード入力をリモートシステムに送信し、リモートシステムからのデータを画面に表示する。これらのデータの受け渡しの際、ビット単位の変換、漢字コードの変換、行末表現の変換、受信データ中のタイムフィラー文字の除去を自動的に行なう。

また、内部コマンドとして、会話型シェル内のコマンドを起動したり、ファイル送受信を行う事が出来る。

ｃａｌｌによる自動ログオン処理の例を以下に示す。 ここでは、自システム名をＷＳ０１、リモートシステム名をＷＳ０２とする。

◎ ログオン手順ファイルの記述例

```
TIMER   5
EXPECT  "terminal type ?" ¥r
SEND    $TERM¥r
EXPECT  "login:" ¥r ¥r
SEND    $RMTUSR¥r
IF      "Password:" $RMTPWD¥r
```

◎ 自動ログイン例

```
[WS01] who am i
karasumi
[WS01] call WS02
WS02
your terminal type ?rtux
WS02-login: itoh
Password:

  Welcome to UNIX world

[WS02] who am i
itoh
```

（下線部はオペレータ入力）

5 あとがき

本稿では、ＵＮＩＸワークステーション上に開発したネットワークターミナルシステムの概要について報告をおこなった。ＴＯＴＡＬ－ＬＡＮへの仮想端末プロトコルを実現するｎｔｔｙデバイスにより、ＬＡＮ経由でも従来のＵＮＩＸコマンドがそのまま使用可能になった。

（参考文献）

［１］島谷、生平、嶋本：“ＵＮＩＸワークステーションにおけるＴＯＴＡＬ－ＬＡＮ接続方式とその性能”
情報処理学会第３３回全国大会論文集、4U-9,1986

［２］中村、伊藤、長友：“ＵＮＩＸワークステーションにおけるネットワークファイルシステム”
情報処理学会第３３回全国大会論文集、3U-6,1986