1Q−6

Ｅｔｈｅｒｎｅｔ環境における端末エミュレーション方式

相澤宣一　山田一章　南部栄一†
(株)日立製作所ソフトウェア工場　†(株)日立情報システムズ

１．　はじめに

近年、日本でもＥｔｈｅｒｎｅｔが普及し、これに接続されたワークステーションを中心にＴＣＰ／ＩＰプロトコルをベースとした分散処理が急速に発達してきた。そして現在、パーソナルコンピュータからメインフレームまでの広い範囲でマルチベンダ共存の分散処理環境が実現している。

こうした中で、同じ分散処理環境から更にメインフレーム上のＴＣＰ／ＩＰプロトコルをベースとしない既存アプリケーションをも利用したいとのニーズが高まってきた。

そこで、Ｅｔｈｅｒｎｅｔ上のワークステーションからメインフレーム上の既存アプリケーションにアクセスする手段について検討した。本論文では、検討の経過と結果について報告する。

２．　端末エミュレーションの必要性

既存アプリケーションは、メインフレームの専用端末を前提としている。これには、次のような機能がある。

（１）漢字の表示、入力
（２）罫線の表示
（３）グラフィックの表示
（４）ファイル転送機能
（５）プリンタ等のデバイス制御

一方、Ｅｔｈｅｒｎｅｔ環境で標準的にサポートされている端末制御手順には、Ｔｅｌｎｅｔ[1,2]等がある。目的を果たすためには、専用端末の制御手順をこれに置き換えるか、どこかでプロトコル変換することが必要である。

ここで専用端末の制御手順は、画面制御のためのデータストリームと、それをやりとりするための通信手順から構成されており、通信手順を他の手順に置き換えることは可能だが、アプリケーションとのインタフェースは、専用端末用のデータストリームである必要がある。そこで、このデータストリームを組み立てたり解釈したりする処理を、どこかでエミュレートしなければならない。以上の理由から、エミュレータが必要となる。

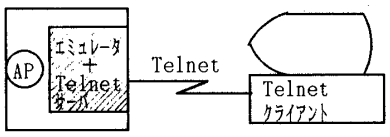
３．　エミュレーションの方式

エミュレーションには様々な方式が考えられるが、次の点が重要である。

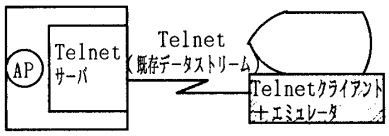
・　トータルの開発量が小さい。
・　どのワークステーションからでも利用できる。
・　メインフレームやネットワークに著しい負荷をかけない。

検討の結果、次の４つの形態を選び、比較することとした。

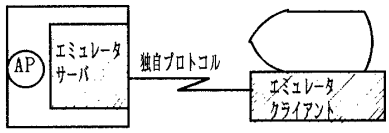
方式１：Ｔｅｌｎｅｔサーバ型[1)2)]
　メインフレーム側のＴｅｌｎｅｔサーバ上にエミュレータを作る方式。



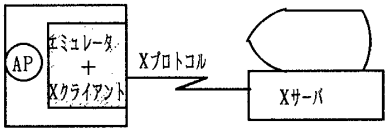
方式２：Ｔｅｌｎｅｔクライアント型[1)2)]
　ワークステーション側のＴｅｌｎｅｔクライアント上にエミュレータを作る方式。



方式３：サーバクライアント型
　既存端末と親和性のあるＴｅｌｎｅｔ相当の独自プロトコルを作り、そのプロトコルで通信するエミュレータプログラムを、メインフレーム側にサーバ、ワークステーション側にクライアントの　形で実装する方式。



方式４：Ｘ−Ｗｉｎｄｏｗ型[3)]
　メインフレーム上にＸクライアントとして、Ｘプロトコルと既存データストリームとのゲートウェイを設け、ワークステーション上のＸサーバと通信する方式。



４．　評価

各方式の比較を表１に示す。比較をした４つの方式は、エミュレータをメインフレーム側で実装するもの（方式１及び４）と、ワークステーション側で実装するもの（方式２及び３）に大別できる。両者の特徴を比較すると次のようになる。

＜メインフレーム側での実装＞
　エミュレータの開発はメインフレーム側のみでよく、またワークステーション側は業界標準プロトコルのみをサポートしていれば良いため、開発工数が小さく、オープン性にも優れている。
　反面、ワークステーションの各機種で共通に持っている機能しか使えないために、機能面では劣り、メインフレームの負荷も大きい。

Terminal Emulation Method for Ethernet Environment
Nobukazu AIZAWA[1], Kazuaki YAMADA[1], Eiichi NANBU[2]
1:Hitachi,Ltd. 2:Hitachi Information Systems,Ltd.

表1　エミュレーション方式の比較

比較項目／方式		Telnetサーバ型	Telnetクライアント型	サーバクライアント型	X-Window型
ネットワーク上のデータ		ANSI準拠端末シーケンス(VT100),JIS7,JIS8,EUC,DEC,シフトJIS 他(クライアントに依存)	エミュレーション対象機のデータストリーム及び文字コード	サーバとクライアントの間で定めたTelnet相当の独自プロトコル	Xプロトコル
対象となる端末の機種		Telnetクライアントを実装したすべての機種	エミュレータプログラムを実装した機種	同左	Xサーバを実装したすべての機種
機能	漢字	○	○	○	○
	罫線	×	○	○	○
	グラフィック	×	○	○	○
	ファイル転送	×	○	○	×
	プリンタ	×	○	○	×
オープン性		◎	○	△	◎
メインフレームのオーバヘッド		大	小	小	大
ネットワークのオーバヘッド		大	小	小	大
開発工数		小(メインフレーム側のみ)	中(WSの機種毎に開発が必要)	大(両側で開発が必要)	小(メインフレーム側のみ)
評価		機種依存性はないが、機能面の制限が大きい	機種依存性はあるが、機能的には充実している	開発工数が大きく、Telnetクライアント型に比べメリットが少ない	現状では稼働機種に制限がある

＜ワークステーション側での実装＞
　ワークステーションの個々の機種の持つ機能をフルに活かすことができ、メインフレームの負荷も小さい。
　反面、ワークステーション側で、機種毎にエミュレータの開発が必要なこと、専用端末の手順を実装しなければならないこと等のため、開発工数、オープン性にやや問題がある。

以上のように両者は相反する特長を持っており、用途に応じて使いわけるのが良いと考えられる。
また、方式（1）と（4）の比較では、プロトコルの普及状況からみて（1）の方がオープン性でやや優れており、方式（2）と（3）の比較では、開発工数からみて（2）の方がやや優れている。そこで、方式（1）及び（2）についてより詳細に評価した。

4．1　Ｔｅｌｎｅｔサーバ型
（1）特長
　ワークステーション側にはＴｅｌｎｅｔクライアントがあればよいので、機器の増設や機種の変更が容易である。
（2）課題
（a）性能と機能のトレードオフ
　メインフレームの専用端末が持つある種のフィールド機能（自動入力、非表示、数字チェックなど）をエミュレーションしようとすると、入力文字単位にクライアントからエミュレータにデータを転送し、サーバ側からエコーする必要があり、メインフレーム及びネットワーク上のトラフィック性能に大きな影響を及ぼすことが考えられる。このため、方式としては可能な機能も性能上の理由により、無効とせざるを得なくなる場合がある。
（b）文字コード体系の差の吸収
　パーソナルコンピュータ、ワークステーションでは、機種によりサポートしている文字コード体系が異なる。通貨記号の違い等マイナーなものを含めると、かなり多数の組合せの変換をエミュレータで実現しなければならない。
　これについては、ユーザオウンコード化する方式が考えられる。組合せは多数であっても、大部分のコード変換は簡単なアルゴリズムで実現できるため、この方式は有効であろう。

4．2　Ｔｅｌｎｅｔクライアント型
（1）特長
　エミュレータとして十分な機能を提供でき、ユーザインタフェース等のカスタマイズも容易である。
（2）課題
（a）汎用性
　エミュレータ開発の際に、ｓｏｃｋｅｔ，Ｘｌｉｂ等の業界標準インタフェースを使用することにより、異機種間の移植を容易にし、汎用性を持たせることができる。
（b）信号の転送方法
　データストリームは文字コードを含め、メインフレーム専用端末のものとなるが、データストリーム以外の信号（例えば割り込みキー）の転送に関して、メインフレーム・ワークステーション間で予め取決めておく必要がある。

5．　おわりに
本検討では、Ｔｅｌｎｅｔのサーバ側及びクライアント側でのエミュレーション方式について詳細に評価し、その利用分野を明確にした。
今後、サーバクライアント型及びＸ−Ｗｉｎｄｏｗ型のエミュレータについても、詳細な評価が必要であると考える。特にＸ−Ｗｉｎｄｏｗ型は、Ｘターミナルの普及とともに重要になってくるものと予想される。
尚、本検討に当って御協力頂いたニチメンデータシステム株式会社システムマネージャー、山田成美氏に深く感謝致します。

参考文献
1) Postel,J. and Reynolds,J.:Telnet Protocol Specification,RFC854,(1983.May)
2) Braden,R.:Requirements for Internet Hosts -- Application and Support,RFC1123,(Oct.1989)
3) Scheifler,R.:X Window System Protocol, Version 11:Alpha Update,RFC1013,(Apr.1987)