情 報 シ ス テ ム 33－3
（1991. 3. 19）

拡張可能な端末エミュレータを用いたアプリケーション開発技法

土屋隆司　　長田弘康

鉄道総合技術研究所

拡張可能な端末エミュレータを用いてミニコンホストとパソコン端末の間の機能分散型アプリケーションを開発する手法を提案する。エミュレーションの対象としたのは、典型的な調歩同期式端末であるVT100である。採用した方法は、各アプリケーション固有の拡張制御シーケンスを使用して、ホスト側アプリケーションから端末側の手続きを直接呼び出すというものである。VT100の制御シーケンスの拡張は開発者が専用ツールを用いて容易に行えるようにした。本稿では、当端末エミュレータの実現方式とその応用事例について述べる。

A METHOD FOR DEVELOPING APPLICATION SYSTEMS USING AN EXTENSIBLE TERMINAL EMULATOR

Ryuji Tsuchiya　　Hiroyasu Osada

Railway Technical Research Institute

2-8-38 Hikari-cho Kokubunji-shi Tokyo Japan

In this paper, we propose a method for developing distributed applications, in which a VT100-based extensible terminal emulator is used to enable programs on host machines to invoke pre-defined local procedures through relatively simple interface. The terminal emulator we have developed is extensible in the sense that it can easily be reorganized, with the help of a tool, to identify application-dependent control sequences and associate each of them with a local procedure. We are going to give an account of the design and implementation of our terminal emulator and also present a brief overview of one of its applications.

1 はじめに

近年開発されている各種アプリケーションシステムには、元来ホストコンピュータに負わせていた処理の一部を端末であるパーソナルコンピュータに分散させているものが多く見かけられる。これには、ホストコンピュータの負荷の軽減、ホスト・端末間の通信量の削減、システムの応答性能の向上、あるいは、パーソナルコンピュータ上の各種ツールの活用等さまざまな目的が考えられる。本稿では、そのようなホスト・端末間機能分散型アプリケーションを、典型的な調歩同期式端末であるVT100のエミュレーション環境をベースに実現する手法について述べる。 今回採用した方法は、端末エミュレーション機能の拡張に基づいて、ホストコンピュータ主導型で、ホスト側アプリケーションと端末のローカル処理の結合を図るものである。

2 開発の背景

当研究所においては、ミニコン、パソコンをベースに鉄道分野における各種エキスパートシステムの開発を手掛けているが、ミニコンの提供するデータベース機能や知識処理機能(プロダクションシステム等)を使ったアプリケーションの中に、パソコンのグラフィックスを駆使した洗練されたマンマシンインタフェースを取り込みたい、という開発サイドの要望が高まってきた。汎用大型計算機とその端末の間には、各種ネットワークアーキテクチャーに基づく本格的分散環境が構築されてきているが、そのような有手順通信に基づく本格的プログラム間通信を調歩同期式通信が一般的であるミニコン・パソコン間に導入するのはコスト、開発作業量の面で必ずしも現実的ではない。そこで、我々は、あくまでも無手順通信の延長線上で、ミニコンホスト上で動作するアプリケーションにパソコンのローカル処理をスムーズに結合する手法を考案し，エキスパートシステム等のアプリケーション開発者に対して提供することにした。方式検討に際しては次のような点に特に留意した。

- ホスト処理、ローカル(パソコン)処理の区別をエンドユーザーに意識させないようなアプリケーションが実現できるようにする。
- ホスト・端末間を低速な通信回線や電話網経由で接続した場合にも実行効率の低下を招かないよう、通信のオーバーヘッドを極力小さくする。
- 端末システムを多数のフィールドへ配布する場合を考えて、低コストで実現する。
- アプリケーション開発者に極力負担をかけない単純なインタフェースにする。
- バイナリファイル、漢字ファイルを含めた透過的データ転送機能を実現する。
- パソコンに付加されている周辺機器や周辺ソフト(e.g. RAMディスク、日本語フロントエンドプロセッサ、各種デバイスドライバー等)と親和性があるものにする。

- パソコン上のグラフィック、マウス関連の既存開発ツールを取り込みやすくする。

具体的な実現方法としては、

1. 端末操作代行プログラムの実現

2. 端末エミュレーション機能の拡張

の2つを考慮した。**端末操作代行プログラム**は、通常端末オペレータがキーボードを用いて行う操作群(ホスト側プログラムの起動、データ転送、ファイル入出力等)の一部を端末側のプログラムで代行しようというものである。この方式では、端末(パソコン)側にシステムのメインプログラムが存在し、必要に応じてホストの機能を呼び出すという形態をとる。ホストはパソコンの処理を支援するサーバーとして位置づけられる。一方、**端末エミュレーション機能の拡張**による方法は、ホスト側アプリケーションから当該アプリケーション固有の制御シーケンスを端末へ送信することによりパソコン上の対応する手続きを直接起動するというものである。この場合、ユーザによって起動されたホスト側プログラムが当該アプリケーション全体の実行制御を行うことになる。両者を比較検討した結果、開発者に対して汎用的で一貫性のある枠組みを提供しやすいこと、システム全体をホスト側で管理できること等を鑑み、今回は端末エミュレーションの拡張による方法を採用することにした。 なお、開発にあたっては、既存の端末エミュレータの改修も考慮したが、それらのエミュレータは、もともとユーザーによる機能拡張を考慮に入れたものではないため、今回の目的には合致しないと判断された。そこで、ユーザーによる機能追加の可能性を初めから考慮に入れた端末エミュレータを独自に設計、開発することにした。

3 基本的な考え方

3.1 制御シーケンスの拡張

VT100端末の制御はホストが送信する一連の制御シーケンス群によって行われる。これらの制御シーケンスを受信した端末は、当該シーケンスに対応づけられている機能(カーソルの移動、画面クリア、画面スクロール等)を実行する。今回開発した拡張型端末エミュレータ(ete = extensible terminal emulator)は、ホストから受信した制御シーケンスの解析を、あらかじめ与えられた表形式データに基づいて行うことにより、認識できる制御シーケンスの追加や制御シーケンスと端末側機能の対応づけの変更を自由にできるようしたものである。 端末エミュレータに与えられる表形式データは、開発者が事前に定義した、制御シーケンスと端末側機能要素のマッピングデータを専用ツールを用いて、端末エミュレータが利用可能な形式に変換したものである。

3.2 表形式データに基づくエミュレータの実行制御

端末エミュレータは表形式データに含まれる制御シーケンスの構文情報に基づき、受信した制御シーケンスを認識し(identify)、対応する各処理への振り分けを行う。一連の処理の流れを図

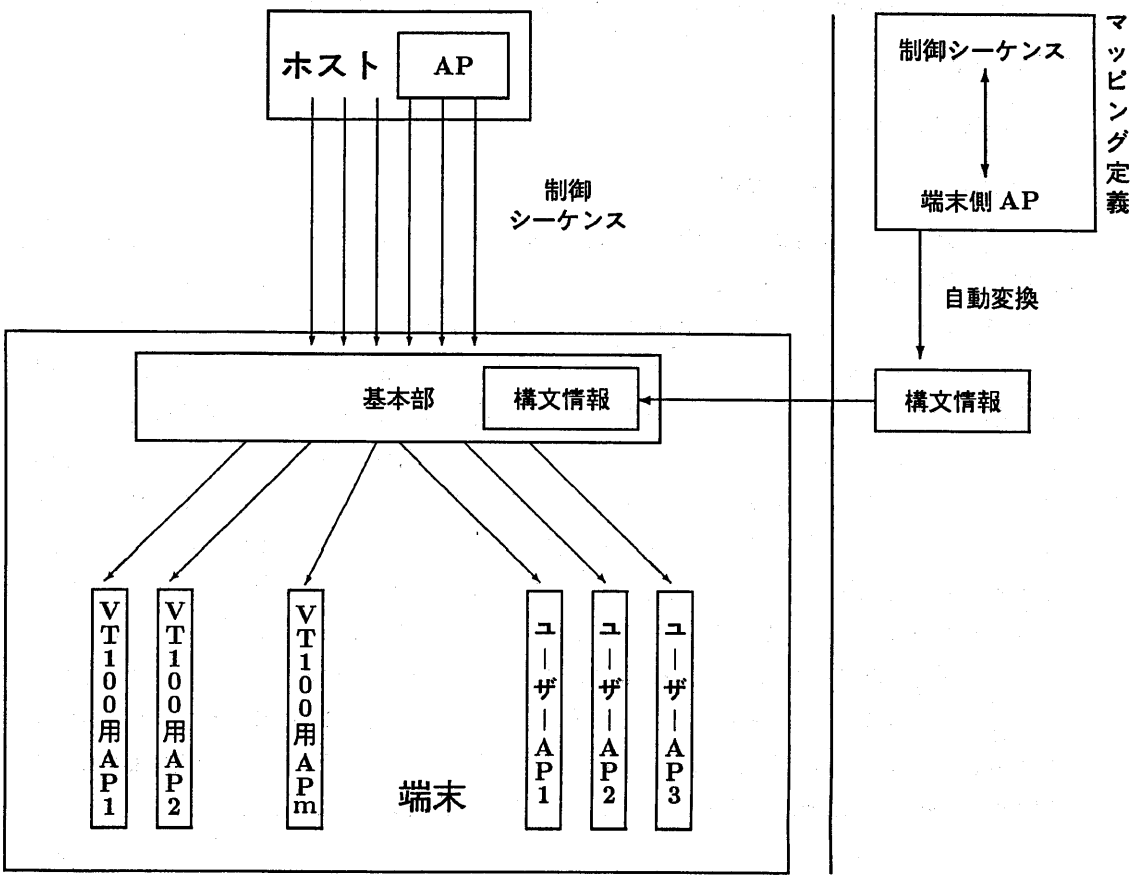

図 1: eteにおける端末エミュレーション処理の流れ

1に示す。 アプリケーションの開発者は，必要に応じて制御シーケンスを拡張し、対応するローカル処理を追加することにより、ホスト上のアプリケーションから端末であるパソコンのローカル処理を直接起動することが出来るようになる。

3.3 端末側手続きの組み込み

制御シーケンスによって、起動された処理の中では、パソコン固有の機能やパソコン上に蓄積されてきた各種開発用ツール類が利用できることは言うまでもない。 したがって、例えば、グラフィック表示機能を持ったパソコンを端末として使えば、ホスト側のグラフィック制御用ソフトを利用しなくともホスト側からパソコンのグラフィック機能を直接呼び出すことにより、高速なグラフィック表示が可能になる。 また、拡張制御シーケンスの一部を用いて、ホスト側アプリケーションと端末側アプリケーションの間の簡易データ交換機能を実現し、当エミュレータの標準機能とした。

4 ソフトウェア構成

当ソフトウェアは、ユーザアプリケーションの一部として動作する**端末エミュレータ本体**(もちろん、単独で通常の端末エミュレーターとして使用することもできる)と、エミュレートする制御シーケンス群から表形式構文情報(状態遷移テーブル)を自動作成する**構文データ作成ツール**の2つから構成される。端末エミュレータ本体のソフトウェア構成の概要を、図2に示す。端末エミュレータ本体は、キーボードやホスト回線(RS232C経由)からの入力情報を解析し、しかるべき処理へ振り分ける基本部、画面、キーボード、RS232C等の制御を行うハードウェア制御部、そしてデータ交換機能等を実現しているアプリケーション部から構成される。実装に使用した言語はC(及び部分的にアセンブラ)である。実際には、パソコン上で比較的ポピュラーな3つのC言語処理系を選び、それぞれに対して対応する版を作成した。これは当エミュレータに対しては、その開発の経緯から明らかなように、アプリケーション開発者による機能追加が想定されるので、各処理系に固有の機能やツール類も可能な限り取り込めるようにするためである。

一方、構文データ作成ツールは定義された制御シーケンスが所定の構文規則に則っているかどうかをチェックし、OKであれば、それを表形式の構文データに変換するものである。なお、作成された表形式データは、C言語のヘッダーファイルとしてそのまま使用できる形式であり、これをインクルードして、エミュレータ本体を再コンパイルし、さらに新規追加した制御シーケンスに対応する処理追加を行った部分と併せてリンクすることにより新しい端末エミュレータが生成される。この構文データ作成ツールは、LISPを用いて実装した。

5 アプリケーション作成の実際

5.1 端末エミュレータ拡張手順

端末エミュレータ ete を特定のアプリケーション向けに拡張する手順を以下に示す。

1. 制御シーケンスから端末機能要素(C言語の関数名)のマッピングデータを定義する(図3を参照)。なおVT100で使用する各制御シーケンスについては、すでに定義済みなので、アプリケーション用に新たに拡張する分だけ追加すればよい。

2. 構文データ作成ツールを用いて、当該マッピングデータをC言語で取り込み可能な表形式データに変換する。制御シーケンスが所定の構文規則に従っていない、或いは、すでに定義したVT100用のシーケンスと衝突するなどの場合はエラー表示をする。

3. マッピングデータの中に新たに定義したCの関数、及び関連する関数群を作成する。

4. 生成した表形式データと新規作成関数群を使って、エミュレータ本体を再作成(コンパイル、リンク)する。

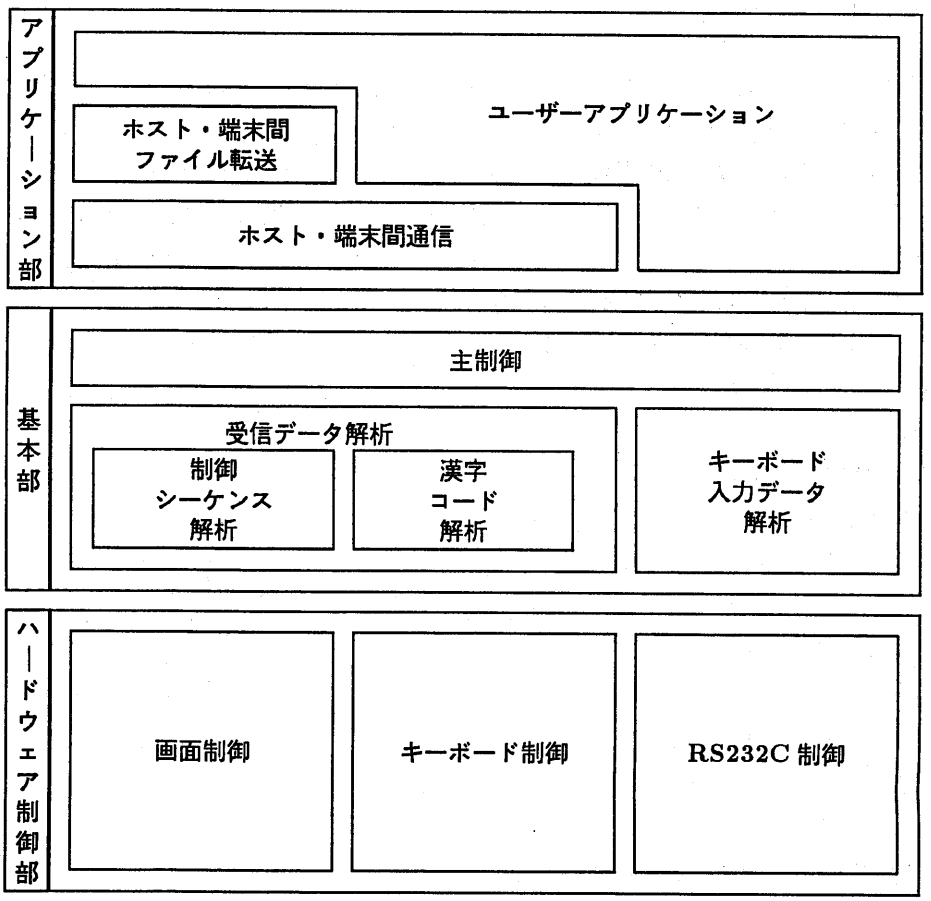

図 2: 端末エミュレータ ete のソフトウェア構成

```
("<ESC>[A" mv_crsr_up1  ｖｔ１００ カーソルを１行アップする。
("<ESC>[<NUM>B" mv_crsr_dwn ｖｔ１００ カーソルをｎ行アップする。
("<ESC>[C" mv_crsr_r1 ｖｔ１００  カーソルを１カラム右へ移動する。)
("<ESC>[<NUM>C" mv_crsr_r ｖｔ１００ カーソルをｎカラム右へ移動する。)
("<ESC>[D" mv_crsr_l1 ｖｔ１００ カーソルを１カラム左へ移動する。)
("<ESC>]<NUM>;<NUM>f" file_trans ファイル転送 テキストファイルの送信)
("<ESC>]m" msdos_command データ転送 ＭＳ／ＤＯＳコマンドの実行)
```

図 3: 制御シーケンスの定義例

```
[1] Escape Sequence ::= ESC INTER FINAL
    ESC ::= 1/b
    INTER ::= ε | INTER0 INTER
    INTER0 ::= 2/0 | 2/1 | 2/2 | ～ |2/f
    FINAL ::= 3/0 | 3/1 | ～ |7/e
[2] Control Sequence ::= CSI | PARA | FINAL
    CSI ::= ESC[ | ESC] | ESCp        ; Control Sequence Introducer
    ESC[ ::= 1/b 5/b
    ESC] ::= 1/b 5/d
    ESCp ::= 1/b 7/0
    PARA ::= SEL_PARA | NUM_PARA
    SEL_PARA ::= ε | DELIMIT SEL_PARA | NUM          ; 選択パラメータ
    NUM_PARA ::= ε | DELIMIT NUM_PARA | NUM NUM_PARA ; 数値パラメータ
    DELIMIT ::= 3/a | ～ |3/f
    NUM ::= 3/0 | ～ |3/9
    FINAL ::= 4/0 | 4/1 | ～ |7/e
          (ε は空記号列)
```

図 4: 使用できる制御シーケンス

定義可能な制御シーケンスは図4に示すEscape Sequence 及びControl Sequence である。これはANSI X-3.64の定義を拡張したものになっており、VT100で使用されている制御シーケンスはすべて含まれている。また Control Sequence には数値パラメータと選択パラメータの2種類のパラメータが使用できる。 eteの受信シーケンス解析処理は、選択パラメータの違いは認識するが、数値パラメータの違いは認識しない。すなわち選択パラメータの異なるコントロールシーケンスは別のシーケンスと解釈し、それぞれ別の処理（関数）へ振り分けるの対して、数値パラメータのみが異なるシーケンスは同一のシーケンスと見なされ、同一の処理(関数)へ振り分けられる。受信した数値パラメータは受信順にスタック上へ蓄積される。ユーザアプリケーションから当該スタックをアクセスするための関数が提供されている。

5.2 ホスト側アプリケーション

ホスト上で動作するアプリケーションから、パソコン上の関数を起動するには、所定の制御シーケンスを端末出力してやればよい。このように極めて単純なインタフェースであるため、ミニコンホスト上の各種プログラミング言語，ツール類からＯＳのコマンド言語に至るまで幅広い環境下で利用することが出来る。さらにホスト・端末間のデータ交換を支援するための基本的関数群

(ホスト、端末ともC言語で実現) を標準提供することにより、アプリケーション開発者の負担を軽減している。 なお、当エミュレータでは、この機能を用いることにより、ホスト主導型のファイル転送機能を比較的短期間で実現することが出来た。このファイル転送機能はホストOSのコマンドラインからファイル転送コマンドを投入することにより、ローカル側 (ファイルオープン、読みだし、書き込み等) の処理も自動起動するので、ローカル側、リモート側双方の設定が必要な Kermit 等のファイル転送に比べ、エンドユーザーとのインタフェースが簡単になるというメリットがある。

5.3 アプリケーション開発事例

端末エミュレータ ete を用いた応用事例として、鉄道工場における車両配置計画作成のためのエキスパートシステムシステムの例を紹介する。このシステムは、鉄道工場において入場車両の配置場所の決定をプロダクションシステムを用いて行うものであり、配置計画作成部にはミニコン上のプロダクションシステム言語 OPS5 を、データ入力、結果表示等のユーザーインタフェース部分はパソコンの C 言語を用いた垂直分散型システムの形態を取っている。このシステムでは、OPS5 のプログラムから随時、拡張制御シーケンスを端末へ出力することにより端末のマンマシンルーチンを起動している。また、端末とのデータのやり取りは、OPS5 から C 言語を介して、ete の標準提供するデータ交換機能を利用している。 なお、このシステムは後に、OPS5 の部分をパソコン上の OPS83 言語で書き変えることにより、パソコンスタンドアロンで動作するシステムとなった。その際にも、分散型システムの時に使用していたマンマシンプログラムはそのまま利用できたため、エンドユーザとのインタフェースを変更せずにすんだ。

6 終わりに

拡張可能な端末エミュレータを用いた簡易分散処理アプリケーションの開発技法について述べた。パソコンの性能向上に伴い、パソコンスタンドアロンで動作するアプリケーションへ重点が移りつつあるが、一方では、分散型システムへの需要も根強い。今後は、RPC(Remote Procedure Call) の実現など、より洗練されたインタフェースを目指していきたい。

参考文献

[1] ANSI Standard X-3.64 1979

[2] 村田、ワークステーションと汎用大型計算機との接続、前川、清水編集、「高機能ワークステーション」 1987

[3] 土屋、長田、 端末エミュレータの効率的実現 第 34 回情報処理学会全国大会 1988