インターフェイスの街角 (54) — Handheld PC の活用

増井俊之

UNIX ユーザーと Windows CE

いまや世の中は〝モバイル/ユビキタス時代〟といってもいいほどですから、UNIX ユーザーであれば、いつでもどこでも UNIX マシンを使って文章を書き、プログラムを作り、Web ページを眺め、メールをやりとりしたくなるでしょう。最近は、Linux や FreeBSD をインストールしたノート PC や Apple の iBook などを使えば、こうしたことも実現可能になりました。しかし、現在市販されているノート PC の多くは持ち運ぶには重すぎ、電池だけでは長時間は使えません。さらに、ハードディスクを内蔵しているため取扱いに注意が必要なので、気軽に持ち歩こうという気にはなれないのではないでしょうか。

一方、持ち運んで使う計算機のために、Microsoft は以前から Windows CE という OS を提唱し、これを載せた計算機が各メーカーから数多く発売されています。Windows CE はデスクトップ計算機の Windows と同じ操作性を保ちつつ、資源の少ない計算機でも使えることを目標としています。

あとで述べるように、Windows CE にはいくつものバージョンがありますが、このうち Handheld PC と呼ばれるタイプの計算機は、軽くてハードディスクを使用せず、キーボードが付いているという特徴があります。たとえば、Hewlett Packard の Jornada や NEC の Mobile Gear、シャープの Telios、NTT ドコモの sigmarion などのシリーズ製品が販売されています。

モバイル UNIX

Handheld PC の操作インターフェイスは Windows によく似ているため、Windows ユーザーにとってはたいへん使いやすくできています。しかし、当然のことながら Handheld PC には Emacs も、便利な UNIX コマンドも入っていないので、UNIX ユーザーにとってはそれほど魅力的ではありませんでした。そこで、Handheld PC のハードウェアを活用しつつ、Windows CE の代わりに UNIX を動かそうという試みがいくつかおこなわれています。

NetBSD/hpcmips プロジェクト[1]は、MIPS プロセッサを載せた Handheld PC マシンで NetBSD を動かそうというもので、現在までに Mobile Gear II や Telios、ビクターの InterLink などでの動作が確認されています。一方、Linux VR プロジェクト[2]でも、Windows CE マシン上で Linux を使うための開発がおこなわれています。

これらのモバイル UNIX システムでは UNIX の大部分のコマンドが使えますし、GCC で独自のアプリケーションを開発することもできるので、たいへん便利です。しかし、現状では以下のようにいろいろと制約があるようです。

- サスペンドがうまくいかない
  モバイル UNIX システムは、Handheld PC マシンのサスペンド機能にまだ対応していません。NetBSD では、電源ボタンで液晶画面を ON/OFF することができるので、1 日程度なら電池をもたせることはできます。しかし、そのまま放っておくと電池がなくなってしまい、再起動が必要になります。
- メモリカードの占有

1 http://www.jp.netbsd.org/ja/Ports/hpcmips/
2 http://pc1.peanuts.gr.jp/~kei/linux-vr.htm

これらのモバイル UNIX システムでは、メモリカード上に置いたファイルシステムやカーネルを使うことを前提としています。したがって、すくなくとも 1 つの CF (Compact Flash) カード /PC カードスロットが、ファイルシステムのために占有されてしまいます。カードスロットが複数ある機種なら、残った 1 つに通信カードなどを挿せばいいのですが、sigmarion のようにカードスロットが 1 つしかない機種では通信カードが利用できないため、用途がかなり限られてしまいます。

- インストールや起動が面倒

  残念ながら、現在のモバイル UNIX システムのインストールにはかなりの手間がかかります。さらに、リセットすると Windows CE が起動し、その後に UNIX をブートする方式になっているため、再起動には相当な時間がかかってしまいます。タッチパネルをもつ Handheld PC の場合には、リセットするたびにタッチパネルの位置調整が必要になるのも面倒です。
- GUI やマルチメディア・ライブラリが不十分

  X ウィンドウ・システムが動く機種もありますが、モバイル UNIX システムでは、画面描画やサウンドなど、マルチメディア関連のライブラリが不十分です。

また、当然ながら、Handheld PC に標準で付属しているブラウザやメールソフト (MUA) などのアプリケーションも使えません。

Handheld PC マシンに UNIX を移植している方々の努力には本当に頭が下がりますが、モバイル環境で Web ブラウザを使ったり、本格的に文章を書こうとすると、残念ながら機能不足の感はぬぐえないようです。

UNIX もどきで我慢する

1 台の計算機で Windows 環境と UNIX 環境を併用したい場合には、VMware などを用いて仮想的に複数のシステムを使うか、あるいは Cygwin のような〝UNIX もどき〟を無理やり Windows システムに導入して我慢する方法がよく利用されています。通常の PC であれば、VMware を使って PC UNIX と Windows を共存させることができます。しかし、Handheld PC ではそういうことができません。

一方、Windows で Cygwin を利用する場合のように、Windows CE の OS とアプリケーションは温存しつつ UNIX 的な環境を導入できれば、UNIX システムとしては不満は残るものの、とりあえず我慢して使えるかもしれません。Windows CE として利用しているかぎり、以下のような Windows CE の特徴はそのまま活かせるので、UNIX もどきで我慢するという選択肢も十分にありうるのではないでしょうか。

- サスペンド機能が有効になるので電池のもちがよい。
- リセットしても高速に立ち上がる。
- Internet Explorer などの Web ブラウザや付属の表計算ソフトなどが使える。
- サードパーティー製の数多くのソフトウェアが利用できる。
- PC カードスロットを有効に使える。

旧いバージョンの Windows CE にはコマンドシェルがなく、以前は Emacs も Perl もファイル処理ツールも皆無に等しい状態だったので、Windows CE を UNIX 的に使うのはほとんど不可能でした。しかし、最近はこういった状況もかなり改善されてきています。

たとえば、Handheld PC 上ではほぼ完全な GNU Emacs 20.7 が使えますし、いくつかのコマンドシェルの上では Perl や grep などの UNIX ツールを利用することもできます。また、あとで述べるように Microsoft が開発環境をフリーで提供しているため、必要なツールを Windows 上でクロスコンパイルすることも容易になりました。これらをうまく組み合わせれば、多少の不満は残るにせよ、UNIX ユーザーでも Windows CE を実用的に使えるのではないでしょうか。

Windows CE を UNIX ふうに使う

Emacs やコマンドシェル、Perl や grep などのツール、プログラム開発環境、TEX などが使えれば、Windows CE マシンといえどもかなり UNIX に近い感覚で使えます。フリーで公開されている各種のツールを導入することにより、かなり満足のいくレベルまで Windows CE を鍛えあげることができます。

キー配置の入替え

Windows CE を UNIX ふうに使うとすれば、Emacs やシェルを多用することになります。したがって、何はと

もあれ、最初にキーの配列をカスタマイズする必要があります。

大きなキーボードをもつ日本語版の Handheld PC マシンは、一般的な Windows マシンと同様、キーボードが JIS 配列になっています。一方、Jornada や sigmarion のような小さなマシンでは、A キーの左横に Tab キーが配置されていることがあります。私を含め、UNIX ユーザーの多くは PFU の Happy Hacking Keyboard のように Ctrl キーが A の左側にある US キー配列に慣れているので、キーボードの配列を変更したいと考えることが多いでしょう。

Handheld PC マシンのキー配列は、中村智史氏が開発した「Quack」というツール[3]を使えば変更できます。Quack では、JIS 配列を US 配列に変更したり、Caps Lock キーと Ctrl キーを入れ替えたり、Tab キーと Ctrl キーを入れ替えたりすることができます。

メモリカードの名前変更

日本語版 Windows CE マシンに CF などのメモリカードを挿入すると、半角カナの ｀メモリーカード´という名前のフォルダが自動的に作成されてしまいます。

シェルや Emacs で半角カナのディレクトリ名を指定するのはひどく厄介なので、適当な英数字の名前に変更しておくほうが無難です。英語版の Windows CE では、メモリカードは ｀Storage Card´という名前で認識されるので、この名前に設定しておけば安全だと思います。

メモリカードのフォルダ名は、レジストリの内容を以下のように修正することによって変更できます。

```
HKEY_LOCAL_MACHINE¥Drivers¥PCMCIA¥ATADisk
DLL="ATADISK.DLL"
Folder="Storage Card"
```

Jornada 680 の場合は、「Queer」というツール[4]を使って以下のように設定します。

```
HKEY_LOCAL_MACHINE¥¥Drivers¥¥PCMCIA¥¥ATADisk
Dll="queer.dll"
Queer="Storage Card"
Queer2="Storage Card2"
```

レジストリを編集するときは、TascalSoft で公開されている「Tascal RegEdit」[5]を利用すると便利です。

3 http://home.att.ne.jp/omega/snak/software/quack/
4 http://home.att.ne.jp/omega/snak/software/queer/

図 1　Emacs for Windows CE

Emacs for Windows CE

以前は、Handheld PC で動く GNU Emacs がなかったため、Emacs ふうのキー操作をもつエディタとしては「Ng for WinCE」[6]などの簡易版 Emacs を使うしかありませんでした。しかし、いまでは Rainer Keuchel 氏が GNU Emacs 20.7 を Windows CE に移植した Emacs for Windows CE が利用できます（図 1）。これは、下記の Web ページで公開されています。

- General Paranoyaxc Software
  http://www.rainer-keuchel.de/software.html

これをインストールすれば、Windows CE でも本格的な Emacs が使えるようになります。

Emacs for Windows CE の使い方は、Keuchel 氏の Web ページ[7]に書かれています。また Emacs for Windows CE で日本語を扱うための設定方法は、daisaku さんの Web ページ[8]に詳しく紹介されています。

以下では、これらのページの情報をもとに、インストールと設定方法について簡単に説明します。

ダウンロード

Emacs for Windows CE のバイナリ、DLL、Elisp (Emacs Lisp) ファイルは、前記の Keuchel 氏のページから入手できます。ただし、Elisp ファイルは最小限のものしか含まれていないため、NTEmacs[9]に含まれるファイルを適宜補う必要があります。

Keuchel 氏が移植している UNIX 関連のプログラムは、すべて **celib.dll** というライブラリを利用する仕組み

5 http://www2r.biglobe.ne.jp/~tascal/download/hpc/regedit.htm
6 http://tillanosoft.com/ce/ngj.html
7 http://www.rainer-keuchel.de/wince/emacs-wince.html
8 http://cwaweb.bai.ne.jp/~daisaku/ce_emacs.html
9 http://www.gnu.org/software/emacs/windows/ntemacs.html

図 2　日本語関連の設定 (.emacs)

```
(set-language-environment "Japanese")
(set-default-coding-systems 'euc-jp)
(set-selection-coding-system 'euc-jp)
(set-clipboard-coding-system 'japanese-shift-jis-dos)
(set-w32-system-coding-system 'japanese-shift-jis-dos)
(setq default-file-name-coding-system 'japanese-shift-jis)
```

図 3　NTANSI フォントを使うための設定 (.emacs)

```
(create-fontset-from-fontset-spec
 "-*-NetTerm ANSI-normal-r-*-*-14-*-*-*-c-*-fontset-null,
 japanese-jisx0208:-*-MS Gothic-normal-r-*-*-13-*-*-*-c-*-jisx0208-sjis,
 japanese-jisx0212:-*-MS Gothic-normal-r-*-*-13-*-*-*-c-*-jisx0208-sjis,
 katakana-jisx0201:-*-MS Gothic-normal-r-*-*-13-*-*-*-c-*-jisx0208-sjis,
 latin-jisx0201:-*-NetTerm ANSI-normal-r-*-*-14-*-*-*-c-*-iso8859-1,
 japanese-jisx0208-1978:-*-MS Gothic-normal-r-*-*-13-*-*-*-c-*-jisx0208-sjis")

(setq initial-frame-alist
      '((font . "fontset-null"))
      )
```

になっています。もちろん、Emacs for Windows CE も例外ではありません。したがって、このライブラリをあらかじめ ¥Windows ディレクトリに入れておく必要があります。

環境設定

Windows CE では環境変数は使えません。このため、Emacs for Windows CE では、ホーム・ディレクトリやディレクトリパスなどの値をレジストリから取得します。

さきほど紹介した Tascal RegEdit などのレジストリ・エディタを使い、以下のように設定します。

```
HKEY_LOCAL_MACHINE¥Environment
 COMPUTERNAME="Telios"
 EMACSDATA="¥Storage Card¥emacs¥etc"
 EMACSDIR="¥Storage Card¥emacs"
 EMACSDOC="¥Storage Card¥emacs¥etc"
 EMACSLOADPATH="¥Storage Card¥emacs¥lisp"
 EMACSPATH="¥Storage Card¥emacs¥bin"
 HOME="¥Storage Card¥masui"
 PATH="¥Program Files;¥Storage Card¥bin"
 SHELL="none"
 TEMP="¥tmp"
 TMP="¥tmp"
 TMPDIR="¥tmp"
 USERNAME="masui"
 UNIXROOTDIR="¥"
```

これは、`Storage Card´という名称でマウントするメモリカード上に Emacs をインストールした場合の例です。当然のことながら、必要とするすべての Elisp ファイルを内蔵メモリ上に置くことはできません。さらに、ハードリセットしたときは、内蔵メモリの内容はすべて消えてしまうので、このようにメモリカード上にインストールするほうが得策だと思います。

日本語表示

Emacs 20.x で正しく日本語を扱うには、文字コードとフォントセットを適切に定義する必要があります。そのために、たとえば .emacs で図 2 のように設定します。

NetTerm という端末ソフトと一緒に配布されている NTANSI フォント[10]を使い、図 3 のようにフォントの設定をおこなえば、漢字が ASCII 文字の 2 倍の幅となってきれいに表示されます。

日本語入力

残念ながら、いまのところ Emacs for Windows CE では Windows CE 標準の IME を使うことはできません。しかし、SKK[11]や POBox server[12]を利用すれば、日本語入力をおこなうことができます。

10 http://starbase.neosoft.com/~zkrr01/html/downloads.html
11 http://openlab.ring.gr.jp/skk/index-j.html
12 http://www.csl.sony.co.jp/person/masui/OpenPOBox/server/

図 4　cmd.exe

図 5　Console

図 6　w32console

コマンドシェル

H/PC Pro (H/PC 3.0) 以降の OS で動く Handheld PC 製品（Mobile Gear MC-R520 や Telios など以降の製品）であれば、標準で付属しているコマンドシェル (cmd.exe) 上で dir や type などの DOS 互換の組込みコマンドが実行できます。これらより前の世代の Windows CE マシンにはコマンドシェルがなかったため、独自に開発されたシェルが使われていました。

cmd.exe

cmd.exe（図 4）は、MS-DOS や、Windows 98 などの〝DOS プロンプト〟と同様の機能をもつコマンドシェルで、アイコンも〝MS-DOS〟という名前になっています。

cmd.exe には、MS-DOS 時代からの組込みコマンドが用意されています[13]。Cygwin の場合と同じように、cmd.exe 上で動く ls や grep などの UNIX ツールが欲しいところですが、比較的最近になって導入されたためか、そのようなツールはあまり使われていないようです。

Console

「Console」[14]は、山梨学院大学の伊藤栄一郎氏が 1997 年から開発しているコマンドシェルです（図 5）。英語版の初代 Windows CE 1.0 しかない時代からフリーで配布されていたという長い歴史をもっています。英語版 Windows CE でも日本語の表示を可能にするために、Console は漢字表示用の kctrl.dll というライブラリとともに利用するようになっています。

13 http://www.wince.ne.jp/frame.asp?/Review/Katsuo/HPCPro/hpc_dos.htm

14 http://www.oohito.com/

初期の Windows CE から使われていたため、Console 上で動く端末プログラムはたくさんあります。しかし、残念なことに、最近はこれらのプログラムのアップデートや新たなコマンドの追加はほとんどないようです。

w32console

w32console は、Emacs for Windows CE の開発者である Keuchel 氏が作成した比較的新しいコマンドシェルです（図 6）。

現在のところ、w32console ではパイプやリダイレクションは使えません。しかし、Emacs for Windows CE でも使われている専用ライブラリ (celib.dll) とヘッダファイル (celib.h) を利用すれば、UNIX の端末プログラムを容易に移植することができます。さらに、w32console 用に開発したプログラムでは、標準入出力を用いて Emacs for Windows CE と通信することもできます。

Keuchel 氏のページには、Perl や Python、ftp など、w32console 上で動く数多くの UNIX コマンドが公開されています。

■ ■ ■

ここまでに紹介したコマンドシェルのうち、どれか 1

つのために作成したプログラムを別のコマンドシェル上で実行すると、まずコマンドシェル・プログラムが起動し、その後に指定したプログラムが実行されます。たとえば、cmd.exe のプロンプトから w32console 用にコンパイルされたプログラムを実行すると、最初に w32console が起動してウィンドウが表示され、続いて指定したプログラムの実行結果が表示されます。Console 用にコンパイルされたプログラムを w32console から起動した場合も同様で、最初に Console が起動してウィンドウが表示され、その後に指定されたプログラムの実行結果が表示されます。

このため、それぞれのコマンドシェル用に作成されたプログラムを、別のコマンドシェル上で実行するのはあまりお勧めできません。

端末ソフト

Handheld PC には、ダイヤルアップ接続で別のマシンに直接ログインするための〝ターミナル〟プログラムが付属しています。しかし、最近は PPP を用いて TCP/IP による接続を確立し、それから Telnet で別マシンに接続する方法のほうが普通でしょう。Windows では、寺西 高氏の開発した「Tera Term」という端末ソフトウェアがひろく使われています[15]。Handheld PC では、Tera Term の Windows CE 版である「Pocket Tera Term」[16]が使えます。

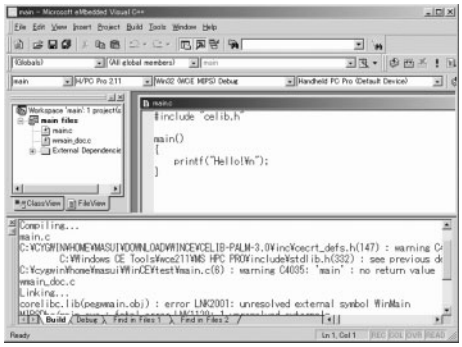
このほかに、w32console 上で動く telnet コマンドも利用できます。この telnet コマンドでは、ポート番号を指定した通信も可能です。

Windows CE のプログラム開発環境

C 言語で Windows CE のアプリケーションを開発するには、現在のところ Microsoft が提供している開発ツールを使う必要があります。

従来は、Microsoft の統合開発環境である「Microsoft Visual Studio」のアドオンとして Windows CE のコンパイラなどが提供されていましたが、これらをすべて購入すると 10 万円以上かかってしまうので、個人で Windows CE のプログラムを開発するのはかなり大変でした。

図 7　Microsoft eMbedded Tools



一方、Palm ではフリーの GCC でプログラム開発をおこなえますし、CodeWarrior のような統合開発環境も数万円で入手できます。こういったこともあって、個人の開発者は Palm のプログラム開発環境のほうに魅力を感じていたように思います。

こういった状況を打破しようと考えたのか、Microsoft は Windows CE のコンパイラと統合開発環境をすべて含む「Microsoft eMbedded Tools 3.0」をフリーで提供しています[17]。

この Microsoft eMbedded Tools は、Visual Studio とほぼ同じ操作インターフェイスをもつ高度な開発環境です（図 7）。

Visual Studio や eMbedded Tools は、エディタやプロジェクト管理など、あらゆるツールを含む GUI の統合開発環境です。ただし、コンパイラやリンカは独立したコマンドになっていますし、プロジェクトも UNIX のものとほぼ同じ Makefile で管理する方式になっています。したがって、端末ベースのプログラムであれば、あとで述べるように GUI をまったく使わずに開発作業がおこなえます。たとえば、Emacs でプログラムや Makefile を作成し、端末上でファイル操作やコンパイル操作をおこないながらプログラムを開発していくことができるわけです。

現在、Microsoft は開発環境を全面的に Visual Studio .NET に移行しつつありますが、それと並行して eMbedded Tools 4.0 という製品の提供も続けるのではないかといわれています。

15 昨年、ドイツで開かれたワークショップに参加したところ、そこに置かれていたすべての Windows マシンで Tera Term が使われていました。海外でも人気があるようです。

16 http://hp.vector.co.jp/authors/VA002416/ce/index_ce.html

17 http://www.microsoft.com/mobile/downloads/emvt30.asp

図 8　MIPS の CPU を使った Windows CE 2.11 システム用の Makefile

```
CC         = clmips.exe
OSVERSION  = WCE211
CEVERSION  = 211
PLATFORM   = MS HPC Pro
TARGETCPU  = mips
CESUBSYS   = windowsce,2.11 -machine:mips
MACHFLAGS  = -D _MT -D _DLL -D MIPS -D _MIPS_ -D mips -D _mips_

WCEROOT    = C:¥Windows CE Tools
INCLUDE=$(WCEROOT)/$(OSVERSION)/$(PLATFORM)/include
LIB=$(WCEROOT)/$(OSVERSION)/$(PLATFORM)/lib/$(TARGETCPU)
PATH=C:¥Program Files¥Microsoft eMbedded Tools¥EVC¥$(OSVERSION)¥BIN;$(PATH)

COMMONCFLAGS   = -nologo -I . -I .. -DPOCKET_SIZE -DPALM_SIZE $(MACHFLAGS)
COMMONCEDEFS   = -D_WINCE -DUNDER_CE=$(CEVERSION)
CFLAGS = $(COMMONCFLAGS) $(COMMONCEDEFS)
LDFLAGS  = -nologo  -subsystem:$(CESUBSYS)

test.exe: test.obj
        link $(LDFLAGS) -out:test.exe test.obj
```

端末プログラムの作成例

さきほど述べたように、eMbedded Tools を利用すれば、コマンドシェルの上で標準入出力を用いて動作するプログラムを簡単に作ることができます。

インクルード・ファイルやライブラリは、コマンドシェルごとに異なるものを使う必要があるので、それぞれについて個別に説明します。

cmd.exe 用のプログラム開発

cmd.exe 上の端末プログラムは、下記のようにごく簡単に書くことができます。

```
#include <windows.h>

WinMain(HINSTANCE hInstance,
        HINSTANCE hPrevInstance,
        LPWSTR    lpCmdLine,
        int       nCmdShow)
{
        printf("Hello!\n");
}
```

このプログラムのコンパイル/リンクには、eMbedded Tools に含まれるクロスコンパイラとリンカを使います。Telios など、MIPS の CPU を使った Windows CE 2.11 の Handheld PC マシンの場合には、図 8 のような Makefile を作成し、eMbedded Tools に付属の nmake コマンドを起動するとコンパイルとリンクが実行されます。

Console での端末プログラム開発

Console でも、端末プログラムは簡単に作れます。C の標準入出力ライブラリがすべて用意されているわけではありませんが、端末プログラムの作成には必要十分な機能をもつ conslib というライブラリをリンクして使用します。

```
#include "conslib.h"

TCHAR AppName[] = TEXT("Hello") ;

int Main(DWORD argc, LPTSTR argv[])
{
        Cputs(TEXT("Hello World")) ;
}
```

w32console での端末プログラム開発

w32console 上で動く端末プログラムも、コンソール入出力ライブラリをリンクすることにより、cmd.exe の場合とほぼ同様に作成することができます。

```
#include <celib.h>
main()
{
        printf("Hello!\n");
}
```

GUI プログラムの作成

Windows CE では、Windows の API のサブセットを使って GUI アプリケーションを作ることができます。

2001 年 10 月号で、Cygwin を用いて Windows の GUI アプリケーションを開発する方法を解説しました。そのときには Cygwin の GCC を前提としていましたが、これの代わりに eMbedded Tools 付属のクロスコンパイラを使えば、Cygwin 環境で Windows CE の GUI プログラムを作成することができます。

WinCE 上でのセルフ・プログラミング環境

残念ながら、いまのところ Windows CE マシン上で C 言語によるアプリケーション開発ができるセルフ開発環境は存在しないようです。しかし Perl や Python、Scheme、ML などの言語は Windows CE に対応していますし、もちろん Elisp も動きます。また、Windows CE 版の Squeak[18] もあるので、C で Windows API を使うことにこだわらなければ、Handheld PC でも十分にソフトウェア開発の楽しみを味わえると思います。

おわりに

今回紹介したような各種のシステムをうまく組み合わせれば、UNIX ユーザーでも、さほどの違和感なく Handheld PC を使えるのではないでしょうか。Web ブラウザやメールソフト、表計算ソフトなど、一般的によく使われるソフトウェアは標準で備わっていますから、使い方によってはモバイル環境でかなり重宝しそうです。

現時点での最大の問題は、Handheld PC を生産するメーカーがほとんどなくなりつつあることでしょう。スタイラスペンで操作する方式の Pocket PC については、各社とも積極的に新製品を出していますが、キーボード付きの Handheld PC はどうも元気がなく、最近は新製品がほとんど発表されていません。NEC は Mobile Gear シリーズの生産を完全に終了してしまいましたし、日立の Persona やビクターの InterLink、シャープの Telios などのシリーズも、このところ新たな製品はまったく発表されていません。多くの PC ショップの店頭では、NTT ドコモの sigmarion と HP の Jornada くらいしか見かけません。ノート PC の軽量化/低価格化に押されて、商品としての魅力が少なくなってきたためかもしれません。

さきほども書いたように、Handheld PC の発売当初はプログラム開発環境は高価で、Emacs や UNIX 互換ツールもほとんどなかったため、UNIX ユーザーにとってはほとんど魅力がありませんでした。ようやくそれらの問題が解決されつつあるのに、多くのメーカーが撤退する様子をみせているのはなんとも残念です。キーボード付きの小型軽量マシンは、文章の作成やメールの読み書き、Web ページの閲覧などの用途にかなりの需要があるのではないでしょうか。ぜひ、今後も製品展開を続けていってほしいものです。

Windows CE は 1997 年の発表からそれほど年月が経っていないにもかかわらず、Windows CE 1.0、Windows CE 2.0、Windows CE 3.0、Windows CE.NET とたびたびバージョンアップしています。これにともない、キーボード付きの機種の名称も H/PC 1.0、H/PC 2.0、H/PC 3.0 (H/PC Pro)、H/PC 2000 と変わってきました。スタイラスペンで操作する方式の機種も、P/PC 1.0、P/PC 1.2、PocketPC、PocketPC 2002 とバージョンアップを重ねています。このように数多くのバージョンがあることに加え、何種類もの CPU が使われているため、自分の持っている製品がどのバージョンなのかを把握するだけでも混乱しそうです。今回紹介したプログラムの多くは CPU や OS に依存することが多いので、導入時には注意してください。なお、Windows CE の歴史については WindowsCE Fan のページ[19]に詳しい解説があります。

UNIX ユーザーが Windows CE を活用する方法については亜蘭一人氏のページ[20]を、sigmarion 用のソフトウェアを開発する環境については米田 聡氏の記事[21]が参考になります。

（ますい・としゆき　ソニー CSL）

18 http://www.is.titech.ac.jp/~ohshima/squeak/WinCE/

19 http://www.wince.ne.jp/snap/ceSnapView.asp?PID=712

20 http://www.sparkling.gr.jp/aran/

21 http://pcweb.mycom.co.jp/news/2002/04/19/06.html