

個人用プログラミング環境

日本ビジネスオートメーション(株)
大野 旬郎

1. はじめに

ソフトウェアハウスの試験研究プロジェクトとして、掲題の“個人用プログラミング環境”があり、現在設計中であるので、とくに対話型システムの使い勝手に関連する課題を若干提示する。

本プロジェクトの背景は次のとおりである。

- (1) ソフトウェアハウスのプログラマーあるいはフリーランスのプログラマーがソフトウェアを出版物のように制作し普及したい。このとき、ソフトウェア制作は、単なる受請いのコード作成という中間生産物の生成ではなく、自主企画によるソフトウェア作成(Adaにおける package や UCSD Pascal における unit)が中心であって、マスマーケットへの配布やアフターサービスが従来にはない課題となる。
- (2) 開発組織は、東京・サンディエゴ・ベルサイユ・オックスフォード他に分散する。Telenetや Tymnet との接続が可能になっているから、トランスボーダーという設定はいまや非現実的ではない。
- (3) それぞれが小規模のソフトウェアハウスやフリーランスのプログラマーの弱い連合組織であるから、プログラミング環境に無制限の資源を投入することはできない。ハードウェアに対する初期投資は、フリーランスのプログラマーにとってせいぜい数100万円見当であろう。
- (4) ビジネスは、伝統的な汎用機用の Fortran コードや Cobol コードの開発・販売もあるが、むしろマイクロコンピュータベースの Pascal コードや Ada コードの制作・普及、関連教育、教材の出版に比重を置く。
相互利用する共通の技術文書等は、仏語や日本語でなく英語が中心になるのはやむをえないとしても、東京やベルサイユでは和文や仏文が用いられるのは明らかである。
- (5) 各地の機器構成は、当初は一樣でない。例えば、サンディエゴには Unix と UCSD Pascal システムがあり、欧州には X5-0 相当のシステムがある。3年後には、どちらにも Ada の核システムが稼動するはずである。東京にも従って何が相当するシステムが必要である。

2. フレームワーク

本プロジェクトの背景から、個人用プログラミング装置のフレームワークが次のように方向づけられる。

- (1) 従来の伝統的なコード作成をむしろ排しているところから、小規模ながら総合的な開発環境となろう。数1,000本ないし数100,000本というオーダーで制

作ソフトウェアを普及させるわけであるから、各地で共有するファイル(データベース)にある関連コードや文書のバージョン管理が1つの主要な技術課題となる。とくに、東京では和文の文書を扱わざるを得ない。

(2) (1)におけるデータベース中の情報伝達は、トランスポーターでは、バルクトランスミッションになる危険があり、ICASの従量制方式(55円/分, 165円/1,000字)を上手に活用する工夫が必要である。ICASの利用を極力さけるなり、他の有用な情報伝達媒体を併用するなり、地域別の作業分担を行うなどの試みがなされるべきである。

(3) 機器構成のうち望ましい要素は、次のとおりである。

- (a) CPU (16ビット)
- (b) RAM (1Mがバイト)
- (c) ディスク (1,000 Mがバイト)
- (d) プリンター (300字/秒)
- (e) 通信装置 (300/1,200 ボー)
- (f) 高解像度グラフィック装置
- (g) 和文ワードプロセッサ
- (h) VTR なしビデオディスク

このうち(a)~(d)のみなら数年後に150万円見当であるが、(e)~(h)については東京では選択して調達するが、共用するがしないと数100万円という初期投資の範囲では困難であろう。

なお、(h)は先進的技術の個人用習得には要となるものである。

(4) Unix上に構築されているUCSD Pascalシステムの開発保守環境(MSEF... Microcomputer Software Engineering Facility)やAdaのMASPE(Minimal Ada Program Support Environment)に相当するものが、個人用プログラミング環境には必要である。資源の制約を前提とすると、これらを数年後に汎用中大機上で実現することは、ほとんど不可能であろう。

独自に類似の開発保守環境を実現する見通しは、16ビットマイクロコンピュータ上のUnix風の基本ソフトウェアやMS-1プロジェクトからみて悪くないと判断される。

(5) 普及したいソフトウェアは、PascalやAdaで記述されるから、UCSD PascalのunitやAdaのpackageがソフトウェアの再利用や流通の単位になるであろう。意味的な注釈をどのようにspecificationに記述するかは当該コミュニティにおける1つの課題となるはずである。

予定されるAdaの処理系(256KB)はUCSD Pascalのそれ(48KB)より5倍以上大きいので、Adaの健全な修得にはPascalとは1桁違う困難が伴うと予想される。それ故、packageの例文集や手引書の類を十分用意する必要がある。ここに、(3)(h)の必要性が生じる。

(6) (1)~(5)の環境におけるソフトウェア体系は類似プロジェクトなどを参考にして構築され、現在その一部が設計されている。

開発方式は、1.(3)の制約があり、核となるシステムを開発し、逐次段階的に拡張する。

とくに、ツール群については急がない。類似システムなどを参考にして実績のあるものを少しずつ採り上げる。

3. 類似プロジェクト

本プロジェクトに関連する若干の類似システムに以下がある。

- (1) PWB/Unix (Programmers Work Bench/Unix)
- (2) XS-0, XS-1 (self-explanatory school computer)
- (3) MASPE (Minimal Ada Program Support Environment)
- (4) ASPE (Ada Program Support Environment)

これらの概要は、図 3.1 のとおりである。

図 3.1 類似プロジェクト

類似システム サブシステム	PWB/Unix	XS-0, XS-1	MASPE	ASPE
インターフェース	shell procedure QED (text editor) mailing function NROFF NEQN TBL TROFF EQN	central dialog control dialog machine structured oriented editor graphics system	command interpreter text editor terminal interface routine configuration manager	command interpreters Ada program editor configuration control system
データベース	file system SCCS (Source Code Control System)	file machine	library file administrator	data base document system project control system
ツール	C (translator) RATFOR (translator) YACC (translator) test drivers	pascal (translator) Modula-2 (translator)	Ada (translator) set-use static analyzer control flow static analyzer linkers loaders dynamic analysis tool	requirement specification tools design tools translators verification fault report system measurement tool

若干の補足を加えると、(1)は周知の大型機ソフトウェア開発支援用のプログラマースワークベンチ(作業台)であり、一部のソフトウェアハウスで利用されている。

(2)については、4.で述べる。

(3)はAdaのオペレーティング環境のミニマルサブセット、(4)は最終環境である。(3)、(4)の各サブシステムに対する米国国防総省の仕様(Stoneman)は、Unixの成果をよく導入しているようであるが、“理論も方法論もないところで、まともなオペレーティング環境などできるわけがない”という酷評(Jean Ichbiah)もある。

Stonemanは、とくに UnixやXDS-1のような対話性の良さを強調しているとは、むしろみられない趣きもある。現行の汎用中大機の“オペレーティングシステム”のような巨大複雑な代物になりかねない危険性に、仕様としてよく歯止めをかけているとはみなされない。

4. 若干の課題

類似プロジェクトとの関連で、本プロジェクトの設計上残されている若干の課題は、以下のとおりである。

(1) 通常の汎用中大機種のオペレーティングシステムの延長上で、ないし3.の機能が実現できるか。

(a) Unix との対比では、次の機能が効率よく実現できるか。

(i) 簡明なファイル体系(一様なレコード、ファイル概念など)

(ii) 簡明なファイルアクセス(7種の主要コマンド、copyなどのI/O)

(iii) shell procedure (簡明なコマンドとパラメータ、ユーザ定義など)

(iv) ソースコードコントロール

(v) パイプ

(vi) とくに、既存プログラムのshell procedureによる再利用

(b) できるとして、構想中の個人用プログラミング環境との価格上の比較はどうか。

(c) できないとすれば、順次新規の環境へ旧ソフトウェアを移行するほかないが、その手順をどうするか。

(2) プログラミング言語の発達を次の段階に区分する。

まず、Cobol や Fortran が提案され、試用され、現象的に使用実績が収集された実験的段階。

次に、アルゴリズムに基づく言語概念を導入した Algol 68 やまた PL/I のような言語が設計・使用された法則的接近の段階。

そして、プログラミング言語の目標や導入すべきパラダイムが追求され、Pascal や Ada のような言語が設計され普及する本質的な段階。

このとき、インターフェースやツールあるいはデータベースアクセスについての対話性(interactiveness)ないし人間機械系(man-machine interface)についての知見は、言語の発達史とのアナロジーではどこに位置づけられるか。例えば、次のような XDS-1 プロジェクトの知見はどうか。

- (a) 従来のシステムは次のような対話性についての欠点をもっている。
- (i) システムのステータス情報を過不足なく利用者にみせない。
 - (ii) 人間機械系のインターフェースであるコマンド名などの記法が不統一である。
 - (iii) ユニバーサルコマンド（例えば、途中で打ち切るなど）が不足している。
 - (iv) コマンド名が適切でなく、キーの配列・指定も不適切である。
 - (v) 表示装置の画面構成が、プリンターの延長上にある。
- (b) 利用者とシステムには、次の3つの次元があるから、利用者の種類に応じてこれを使いわけなければならない。

図4.1 利用者とシステム間の3つの次元

項目 次元	利用者の疑問・いらい	システムの応答・あいつち
(i)	Where am I ? What data is being affected ?	SITE (dataの近傍)
(ii)	What can I do ? What's active command ?	MODE (commandの近傍)
(iii)	How did I get here ? Where can I go ?	TRAIL (対話の経時)

- (c) 利用者を、システムの習熟度や立場によって、(i) casual user (ii) programmer (iii) instructor と区分けする。
- (d) 一般的に、有効な対話の方略は次のとおりである。
- (i) 利用者にしてできる限り主権を与える。
 - (ii) (b)(i)~(iii) にできる限り素速く応答する。画面上に常時 SITE を表示し、MODE や TRAIL がすぐわかるようにしておく。
 - (iii) 入力はできる限り、単一キーで済むようにする。
 - (iv) コマンド言語もできる限り、単純にする。
 - (v) システムから利用者への情報提示は、多少の冗長度があった方がよい。
 - (vi) 絵とくに動画は非常に有効だから、必要に応じて使用する。
 - (vii) 画面は一杯に有効に使う。
- (3) (2) のほかには有力な代替案がある。
- (4) 個人用プログラミング環境においては、プログラミングないし定型的なプログラミングとしてのコーディングの事前作業、事後作業が、通常のそれと比較してどれ程手間がかかるか。
- (5) 各工程で、(2) の方略はどう設定できるか。本プロジェクトでとくに考慮を要する要素は次のとおりである。
- (a) 通信機能の利用度
 - (b) package や unit の再利用
 - (c) 文書等の提示機能（和文を含む）
 - (d) 独習（CAI）機能（VTR の編集なども含む）

5. おわりに

本稿は、現行プロジェクトの途中経過の一端を素描しているに過ぎない。一応の区切りがついた時点で担当者による個人用プログラミング環境の成果が発表される予定である。

本プロジェクトの構想には、K. Bowles さん、J. Ichbia さん、J. Nieverget さんほかから建設的なコメントを頂いたので、謝意を表しておきたい。