ソフトウェアとの付き合い

An Experience in Software Development

大筆　豊

OHFUDE Yutaka

鳥取環境大学紀要

第 9 号・第10号合併号　2012. 3 抜刷

Reprinted ftom

BULLETIN OF TOTTORI UNIVERSITY OF ENVIRONMENTAL STUDIES

Volumes 9 & 10　Mar. 2012

鳥取環境大学紀要
第９号・第10号合併号（2012.３）pp.１－９

ソフトウェアとの付き合い

An Experience in Software Development

大筆　豊

OHFUDE Yutaka

１．はじめに

鳥取環境大学を退職するにあたり、最終講義の内容を文章にせよとの紀要委員会の指示があり、以下のようにまとめてみた。私は何かの間違いで大学の教員になったが、研究者としての実績は殆どなく、研究についてではなく実践的なことしか書けず、また文章としても品位のないものになったと思うが、ある１人の老いたるプログラマの生き様が、後進に何かの役に立てばと願うばかりである。

２．プログラマ（子供の頃の憧れ）

もう60年も前、小学校の中学年の頃であったか、リーダースダイジェストという雑誌で、毎日頭をひねりパズルやゲームをしているような面白い職業が生まれたとの記事を読んだことがある。アメリカでコンピュータプログラマが正式の職業として認知された頃である。将棋や囲碁を覚えだしたころなので面白そうだな、できれば将来やってみたいな、と思っていた。

３．ノーベル賞狙いから技術者へ

理数が好きなので、湯川さんの跡をついでノーベル賞でも取ってやろうかなと、京都大学の理学部を受験したが、どういうわけか意見があわなかった。予備校では結構秀才であったが浪人してまでノーベル賞はないだろうと、工学部を目指すことにした。工学部の数理工学科に計算機工学講座があるのを知り、子供の頃の憧れもあり、そこに挑戦し、幸いなことに二浪しないですんだ。３回生のとき、演習授業で初めてプログラムを組んだ。課題は連立一次方程式の解を掃き出し法で求めるものであり、８進数の機械命令で200ステップくらいだったと思う。プログラムを組む楽しみを初めて知ったが、同時にデバッグ（誤りの修正）の辛さを思い知らされた。友人は楽々と答えを出すのに、何度やっても駄目で最後まで行き着かなかった。

４回生になり研究室配属のとき、もちろん、（日本のコンピュータの草分けの１人である萩原宏教授の）計算機講座を希望した。６人の枠に対して７人の希望者があり駄目かなと思っていると、先生が「全部まとめて面倒見るよ」とのことで助かった。修士課程にもその７人とも志願し、全員進学した。大学院の試験では、筆記試験の後、１人づつ、教授陣全員が並ぶ前での面接があり、萩原先生が「私の答案に名前を書いていない学生がいてね」と話された。当然私のことらしいので「しまった、今度こそ駄目か、就職活動をしないといかんな」と思っていたが、またも助けてもらったらしい。私のようにケアレスミスをするものはプログラマに向かない。工学部の大学院とは、上級技術者の養成所であると勘違いしていたが、研究者を育てるところらしく、私の肌には合わず、博士課程には行かず就職した。技術者として頑張るぞとの夢を見て。工学部の修士課程は技術者教育に徹すべきであると今でも思っているが、大学教員の多くは研究者を養成したがっているようだ。

４．恥ずかしながら研究員

計算機関連の企業ならどこでもよかったのであるが、同期の友人がまだ決めていない東芝に入れた。１ヶ月間の新人教育が終わるとき、配属先の希望を３つ書かされたが、もちろん研究所なんて想定外であり書かなかった。配属先の発表がある前日の日曜日に、本社に呼び出された。希望部門とミスマッチした20人ほどが呼び出されており、廊下で待っていた。１人づつ呼ばれ、私には「君は研究所を希望していないが、研究所でいいな」との脅迫である。会社対新人、力関係から言っていやとも言えず夢が破れる。それ以来東芝での研究所暮らしが28年も続いた。研究所にいる限りは研究者と言うらしいが、とてもそんな玉ではなく、場違いもいいところだが、ソフトウェア関連の研究グループの一員になった。人間万事塞翁が馬、何が、幸か不幸か、わからない。研究所にい

たからこそ、年を食ってから、社外の委員会活動や学会活動、大学との対応に駆り出され、結果として鳥取環境大学に誘われたのだから。

５．究極のバグラー

当時、研究所には TOSBAC-3400という計算機が鎮座していたが、主記憶容量は64Ｋ語（24ビットマシンだから192Ｋバイト）、今のパソコンと比べても千分の一くらいで玩具以下である。しばらくは、好き勝手に玉突きのシミュレーションなど何の役にも立たないプログラムを作っていた。プログラムを作るのは好きだから、幾らでも作るのだが、作っても思い通りに動かない。私の場合、30ステップに１回はバグ（誤り）を製造する。これは一種の才能であり、他人には負けない。1000ステップくらいのプログラムなら30個くらいのバグがあり、１個のバグを取るのに１回計算機で走らすこと（ラン）が必要とすると、30回のランが必要になる。ほかの人なら悠々数回のランですむのに情けない。究極のバグラー（バグ生産者）というところ。バグラーはバグラーなりに良い事がある。情けない、恥ずかしい、としょげるだけでなく、どうすればいいかとけなげにも考えることになるからである。

６．けなげな工夫１：トレーサ

６－１　トレーサ

プログラムは命令を順に書いていくが、その実行は条件により分岐したり繰り返したりするので、動的な動きを理解する必要がある。たとえば、気になる時点での変数の値をそこで止めて確認するなどができればよい。これをやるのにトレーサが便利であるが、１人がコンピュータを独占することになる。

６－２　紙の無駄遣いの疑似トレーサ

研究所にはコンピュータは１台しかなく、多くのプログラムを同時に実行するためバッチ方式で使用しており、複数のプログラムのカードをカードリーダで次々読ませ、次々に実行させ結果をプリンタで出力する。そこで私は次のような疑似トレーサを作った。分岐するときは分岐元と分岐先で、サブルーチンを呼び出すときは、呼び出し命令とサブルーチンの入り口・出口で、番地とすべてのレジスタの値をラインプリンタ１行120桁にほぼ一杯バタバタと出力させた。アセンブラによるプログラミングであったがマクロを旨く使って簡単にその処理を書けるように工夫をした。私がプログラムを実行させると、いつでもラインプリンタの何十枚、多いときには数百枚にわたりトレース情報がバタバタと出力される。バグがあるとき実行の順にトレース情報が出力されているので、どこを通ったか、変な値になっている部分があればどこかがすぐにわかるようになった。その結果、デバッグの効率が上がり、1000ステップのプログラムでも数回のランで完成するようになった。ただし、ラインプリンタの用紙は私が使うたびに減り方の早いこと。

６－３　人の成果をタダ食いする輩

私のやり方を見ていた同僚連中が、厚かましくも私のトレーサを無料でコピーして使っていた。今から思うと、せっかく使ってくれているので、使用前・使用後のデータをとり「効率的なデバッグの手法の実践例」くらいで論文にまとめられたと思う。悲しいかな、自分が研究者であるなんてこれっぽっちも思っていなかったので、データも取っていなかったし、論文なんか書いていない。

その後、使用するコンピュータが変わるたびにまずトレーサを作り、自分のためだけでなく同僚も使えるようにしていた。事業部のプログラマにも宣伝していれば使って貰っていたかもしれない。

７．けなげな工夫２：設計言語とデュアルコーディング

７－１　高水準言語による設計

当時、プログラムの設計はフローチャートで行うのが普通であった。フローチャートは２次元の図であり、きれいに書くのは面倒であるし、配置を工夫しないと読みにくい、設計とプログラムの対応が明確でないなど、私は好きではなかった。

研究所の先輩がソフトウェアを外注をするのに、仕様書として FORTRAN でプログラムを作り、それをアセンブラで書き直すようにプログラマに指示していた。FORTRAN では正しく動くのだが、メモリ容量を減らし実行速度を上げるため、アセンブラで書き直す必要があったからである。高水準言語で設計する手法の一種である。

７－２　自己流設計言語の発明

先輩のやり方に触発され、自分流の設計言語を発明し使っていた。たとえば「もし子供が男なら野球の選手にする、そうでなければバレリーナにする」という文を

```
if 子供　＝　男
　　then 野球の選手にする
　　else バレリーナにする
```

とすれば、一見プログラム風になる。条件分岐を「if～then～else」、繰り返しを「while～do」など ALGOL 風

に書き、処理の部分は日本語で書いていく方法で設計する。私なりに工夫したものではあったが、すでにほかでも使われており擬似言語というらしく私の発明ではない。こうすると設計記述が１次元になり、かつ、論理が明確になり、そのうえプログラミングとの対応がしやすくなる。

7－3　一意的なプログラミング

何度も試行錯誤しながら工夫していくと、同じ設計からほぼ一意的にプログラミングをすることが可能になった。一意的にプログラミングできることから、設計言語による設計をもとにプログラミングをしてから、何日か時間を置いてから、同じ設計からもう一度プログラミングを行い、前のプログラムと対比させる。違うところがあるとどちらかにバグがあるはずであるから、その部分を修正すればよい。私のように30ステップに１回はバグを作る名人でもこの操作を行うことにより「30×30＝900」であるので、1000ステップ程度のプログラムなら１回のランで完成することになる。実際には、設計そのものに誤りがある、あるいは２回とも同じところで同じ誤りをするなどがあり、そんなに甘くないのだが、飛躍的に早くデバッグできるようになった。これも私が編み出した手法であるが、同じような手法があり、デュアルコーディング（Dual Coding）と呼ばれていたことを後で知った。

7－4　社内でも使う人が出てきた

自慢じゃないがこの手法も論文にしていないが（論文になるという意識も全くなかったが）、セミナー業者の主催するプログラミング技術のセミナーで話したことがある。セミナーのために、20ページ位の予稿を書いたが、それを見た社内のソフトウェア生産技術関係の人が改めて印刷し、社内のいろんな部門に配布してくれたので、設計言語を使うこの手法を多くの人が使ってくれた。印税も貰っていないので、著作権侵害であるのは明らかであるが異議申し立てはしていない。役に立ってくれればそれでいい、というより我が幸せである。理学は真理の発見が主題であり、オリジナリティが重視されるが、工学では役に立つ事が目的であり、新規性は２の次であると思っている。新規性（珍奇性）を追求するあまり、糞の役にも立たない重箱の隅を突っつくような研究をする輩が多いが私の趣味に合わない。まあ好きにやっておくれやす。

8．ALGOL-N という言語のコンパイラ開発

8－1　コンパイラ開発の願望

入社して２年ほどは、国家プロジェクトの一環で、LSI 設計用の CAD のプログラムの開発を担当していた。それが終わったとき、東大出身の先輩から「これから何かやりたいことがあるの」と聞かれ「コンパイラを作りたい」と答えると、「ALGOL-N をやれば」と言われた。早速、東大のW教授のところに連れて行かれ、お話を聞くとともに修士課程のK氏（現早稲田大学教授）を紹介され、毎週１回研究所に来ていただきコンパイラの作り方を教えて戴くことになった（本音から言えば囲碁のプログラムを作りかったが、私の真面目な性格が災いし、さすがにそれは言えなかった）。

8－2　ALGOL-N

ALGOL-N というのは、欧州で開発された自己増殖言語 ALGOL68に対抗して日本の学者が集まり作られた言語である。自己増殖言語とは、言語仕様を使用者が定義できるようになっており、言語そのものを拡張できる言語で、一種のコンパイラコンパイラ（コンパイラを開発するためのコンパイラ）である。ALGOL-N のコンパイラは東大（K氏）や日立中央研究所などで作られていたが東芝でもやらないかといわれていたようだ。

私を含め社内で５人位、関連のソフトウェア会社から５人位、合計10人位のチームで開発することになった。入社して数年しかたっていない、バグだけは立派に製造できるが、コンパイラを作りたいという願望はあってもその技術もない若造に、何故作らせたのか不思議である。昔はのどかでよかったというしかない。普通のコンパイラも作ったことがないのに、いきなり自己増殖型言語とは荷が重い。

8－3　非終端記号ごとの再帰ルーチン

K氏に教えてもらった手法は、「非終端記号」ごとに再帰呼び出しのルーチンを作るトップダウン手法である。たとえば「私は日本人です」という文を、「文＝主語＋述語」、「主語＝名詞＋助詞、述語＝名詞＋助動詞」、「名詞＝私、日本人、助詞＝は、助動詞＝です」という風に分解すると、文の構造を文法的に説明できる。言語理論では「文、主語、述語、名詞、助詞、助動詞」を非終端記号、「私、は、日本人、です」は実際の文に現れるので終端記号という。非終端記号「文」、「主語」、「述語」、「名詞」などに対してそれぞれ再帰的なサブルーチンを作り、その中で解析したりオブジェクトコードを作成する手法である。

８−４　デバッグマシンを求めて放浪の旅

開発はTOSBAC−5600という当時の大型計算機で行った。研究所には１台だけ、社内にも本社の情報システム部門にあるくらいで、デバッグも容易ではない。デバッグやテストをするために、電気試験所（現在の電子技術総合研究所）や子会社の計算センターのマシンを使わせてもらいに放浪の旅をした。ある部分をテストしたいとき、その部分を開発した担当者と私が出かけて行ったが、他のメンバーからは、その間何をしたら良いかわからないと文句が出る。別の部分についても同様である。結局リーダである私が何時も席にいないことになる。チームプレーをしているときリーダが部分の開発に没頭するとチーム全体として機能しなくなるのを実感した。リーダは我慢してでも、全体を見渡すためにどっかりとデスクに座っているべきである。

８−５　開発完了と同時に没

10人のチームで約２年間かかりやっと完成した。完成したといっても、私が始めに設定した13個のテストプログラムを何とか実行できるようになっただけで、まだバグだらけであったと思う。それでも先輩が手配してくれて、そのコンパイラを情報システム部門の計算機に登録したので全社で使えるようになった。皆さんに使ってもらうには教育・宣伝をする必要があるなと思っているとき、突然電話がかかってきて「こんな下らないコンパイラを作ったのはお前か、システムから追い出すからな」との丁重なご挨拶。情報システム部門の責任者Ｙ氏で当時課長クラスの人だがそのときは面識がなかった。（東芝の計算機分野での実力者で、怖ろしいことに、その後20年以上にわたり何度もベロベロとかわいがられた。怖かったの何の）。

10人で２年がかりで開発したものがあっけなく没になってしまった。技術者１人雇うのに今なら1000万円かかる。研究所長に「5000万円（今なら２億円）の損害をかけてしまいました」と謝ると「だんない、だんない、教育費」との一言。正直助かった。

９．PL/40というコンパイラの開発

９−１　TLCS−12というマイクロコンピュータ

東芝では、米国Ford社のエンジンコントロール用に12ビットのマイクロコンピュータ（TLCS−12）を開発していた。インテルの４ビットマイコンが始めて市場に出だした頃で、世界でも最先端を走っていたと思う。ある事業部からTLCS−12用に言語を開発して欲しいという要望があり、PL360をベースに言語仕様を作り提案した。

A4で４〜５枚の手書きのもので、殆ど言語構文だけのようなちゃちなものである。研究所長から「開発するのに幾らかかるか」と問われ「人件費換算で3000万円掛かります」と答えた。もっと安く言い、開発させて貰えるようにすればよかったが、ALGOL−Nでは5000万円換算の金が掛かって結局使い物にならなかったので、正直に、安全をとり多めに言ったのだが「そんなに掛かるか」と没になった。

９−２　PL360

PL360はPascalなどを開発したスイス人Wirth（ヴィルト）がIBM360用に開発した言語で、

```
if R1 > 0
  then R2 = 1
  else  R2 = 0
```

のようにコンピュータ内部のレジスタも陽に記述する構造化アセンブラといった中水準言語である。私は「if〜fi」、「while〜elihw」、「when〜nehw」、「for〜rof」のように、先頭のキイワードを逆にしたものを最後に書き制御構造が明確になるように、余計なことを追加していたが。これは、ALGOL68で「if〜fi」、「do〜od」のように用いられた手法で、いつか使ってやろうと思っていたものである。

９−３　コンパイラの開発を即決で決断

しばらくしてＴ氏（後に徳島大教授）が、東芝も参加していたパターン大プロという国家プロジェクトの、東芝側のリーダとして事業部から研究所に戻ってこられた。TLCS−12用に作った先ほどの仕様書を持って行き、「こんなもん考えていました」と見せると、即座に「ミニコンTOSBAC−40用にコンパイラを半年で作れ。メンバーとして４人つける。文書は相互チェックをすること」との指示を出された。私はまだヒラであったがＴ氏は部長クラスである。今思うと物怖じもせず厚かましいことをやったものだ。言語の規模は違うがALGOL−Nでは10人で２年もかけたので、本当にやれるのか心配であったが「やります、そのかわり宣伝はお願いします」と返事した。開発させてもらえるだけ嬉しかった。つけられた４人は、男性２名（Ａ、Ｂ）、女性２名（Ｃ、Ｄ）の新人ばかり。要するに開発経験をつけさせて教育しろということらしい。

ALGOL−Nでは失敗したものの、曲がりなりにもコンパイラの作り方をマスタしていたので２〜３日のうちに、言語仕様をTOSBAC−40用に書き換え、コンパイラのアーキテクチャを決め、自分も含めて５人の分担を決

めた。私以外のメンバーは新人ばかりであったので、私自身は全体の３分の１を分担した。設計の仕方はもちろん ALGOL-N でやった非終端記号ルーチンを再帰的に呼ぶ出す方式で、設計の記述方法は、私流の設計言語である。

9－4　チーフプログラマチーム

開発は、結果的にはチーフプログラマチームの形式を取った。チーフプログラマチームというのは外科手術のチームにたとえられ、執刀する外科医がすべての責任と権限をもち、その他の人は、汗を拭いたり、メスを渡したり、心電計を見ているなど補助的な作業をするだけである。

４人の作成した文書（設計、プログラム、テスト仕様など）のすべてを私がチェックした。それに加えて男性（A，B）の２人、および女性（C，D）の２人は相互に文書をチェックし、私の作成した文書は修士を出たそれなりに出来る男性（A）にチェックしてもらった。自分の分担分の開発以外、メンバー全体の文書のチェック、テスト仕様の作成、マニュアルの作成などを行い、私の負荷は想像を絶した。当時会社の儲けが少なく、研究所に割り当てられる残業時間は少なかった。おまけに残業時間のほとんどは他のチームに配分され、私のチームには回ってこない。残業できないので定時になると書類を抱えて家に帰り、食事のあと午前２時、３時まで仕事をつづけた。会社で残業していれば遅くとも10時ごろには帰るのだが、家で仕事に熱中すると歯止めがなくなる。ALGOL-N の失敗もあったので、断崖絶壁に立たされたような気持ちである。もちろん土日にも家で仕事をしていた。

５月の連休の前、T氏が心配なのか、B君の設計をチェックし「何だこれは、書き直せ」と命令した。200ページほどの設計を、連休中B君は泣きながら書き直した。私もすでにチェックしており「おや？」とは思っていたが「まあいいか」と大目にみていた。結果としてB君は大きく成長したと思う。人を育てるときには厳しさも必要であると教えられたものだ。

開発の途中で、府中工場のソフトウェア生産技術担当の課長から「PL/40のコンパイラ開発を、設計言語でやっているそうだが資料を送って欲しい」との電話があり、A4で10枚程度の資料を送った。説明に来いとも言われなかったが、PL/40の普及の一助になったと思う。

9－5　宣伝活動は失敗か

約束どおり半年後にはコンパイラが完成し、手書きのマニュアルを作りT氏に報告すると、図書館に持ち込みタイプし印刷してくれた。ワープロもない時代で、自分の作成した文書が活字化されたのは始めての経験であり、それだけでも感激したものである。TOSBAC-40用だからPL/40と命名した。アセンブラで３万ステップくらいであったが、私が定義したマクロを駆使していたので、普通ならその倍は掛かっていたと思う。

T氏に「宣伝してください」というと、自分でやればとばかりに、社内の２部門を紹介して貰っただけである。一方はミニコンの基本ソフト部門で、部長のH氏以下20人位が会議室で聞いてくれた。１時間ほど説明して質疑の時間になると、H氏がおもむろに「研究所で（頼まれもしないのに）何でこんなものを作ったのか？」、「評価をしたのか？」など厳しく追及してくる。もう一方は重電技術研究所の技監M氏（後に京大教授）がたった一人で対応してくれ「そうでっか、そうでっか」と頷くだけで糠に釘。研究所長に「一方からはきつく言われるし、もう一方は糠に釘、やっぱりあきまへんわ」と報告すると悲しそうな顔をしていた。所長にしても研究所の成果が売れれば嬉しいし点数になる。

9－6　セルフバージョンの開発

一月ほどして、基本ソフト部隊の30歳位の女性のプログラマから、「PL/40を使ってみて」という報告が届いた。部長は厳しく言っていたが「まあ、遊んだれや」程度に試用させたらしい。生産性や品質向上についての数値は全くなく、A4で１枚だけの「結構面白い言語です」という報告。

また一月ほどすると、セルフバージョンを事業部で作るので今までの設計資料などすべて送れ、ついでに大筆も付けること、との司令がきた。セルフバージョンの開発にも私が指導（というより邪魔）をすることになった。

研究所で開発したコンパイラは、大型機でコンパイルしてオブジェクトコードを紙テープで出力するクロスバージョンである。セルフバージョンとはミニコン自身でコンパイルし、実行できるようにするものである。本音から言って、研究所で作ったものなんか危なくて使えない、製品にするには事業部で作り直すということであろう。

9－7　やっと自分の作品が使ってもらえるか

セルフバージョンがもうすぐ完成するという時期、東芝府中工場で「ソフトウェア生産性向上事例発表会」なるものがあった。私もH氏の隣で聴講していた。ある部門が「設計言語で設計し、それを元にアセンブラに変換

すると、生産性・品質とも大幅に向上した」との発表があった。設計言語による私の手法が伝わったのか独自にその部門で編み出したのかはわからないが。

それを聞いたH氏がフロアから「今まさに発表された技術を、コンパイラとして近く提供します、ぜひ使ってください」と話された。やっと私のコンパイラが社内で使ってもらえるのだと実感でき、涙があふれそうになった。

9－8　アセンブラからPL/40へ

それまで、ミニコンのプログラムは殆どアセンブラで開発され、実行効率を問題にしない一部のものだけFORTRANで開発されていた。PL/40が使えるようになってからTOSBAC-40あるいはその後継機種のアプリケーションのほぼすべてのプログラムがPL/40（あるいは同種の言語）で開発されるようになった。当時、ミニコンはシステム制御用という位置づけにあり、発電システム、配電システム、鉄鋼システム、化学プラント、水道システム、病院システム、交通システム、防衛システムなど社会のあらゆるシステムを制御するのに使われていた。PL/40はそれらのシステムを開発するのにほぼ全面的に使われるようになった。

PL/40は構造化アセンブラとしての位置づけにあり、機械依存の言語である。アセンブラでプログラミングをするよりはるかに生産性が向上するものの、コンピュータの機種が変わるたびに言語を変更する必要があり汎用性に欠ける。当時でも汎用言語としてのC言語はあったが、TOSBAC-40のように制御用目的では、メモリ効率や実行速度が悪すぎ、とても使い物にはならなかった。コンピュータの性能が上がり汎用性のあるC言語に移行するまでの10年ほどは、殆どのアプリケーションで使われるようになっていた。

9－9　普及活動

良い言語を提供すれば現場はすぐに喜んで使ってくれるかというと、そんな甘くはなく、宣伝・教育活動が必要になる。私のようにヒラレベルではどういうすべもないが、幸い、T氏が本社の技術管理部に話をつけ、全社レベルで推進をすることになり、その研修会を開くことになった。しかるべき地位の人がプロモートすると物事は動き出す。講師は私と社外の研修に慣れた教育のプロF氏の２人で、教材はもちろん自分で作成した。全部で20回くらい社内の各部門で講習を行った。その後も、F氏が何年か講習を続けていた。そういう努力も実り、普及するようになった。

10. 捕らぬ狸の皮算用

10－1　費用削減効果

府中工場だけで関連会社の人も含め１万人くらいの人がおり、ほかに日野工場や青梅工場、小向工場、柳町工場など多くの部門でミニコンを使うシステムを開発していた。控えめに言って関連のソフトウェア会社を含め2000人のプログラマが、C言語に移行するまで10年間以上は使っていた。生産性は、アセンブラに比べて確実に20％は向上した。20％くらいしか生産性の向上が見込まれないから、移行に伴う教育費用やトラブルがあり、すぐに元の技術や手法に戻ってしまう。いったん使い出した部門がアセンブラに戻らなかったのが、効果を実感した何よりの証拠である。

技術者１年間の費用を1000万円とする。PL/40を使うことによる削減効果は「2000人×10年×20％×1000万円/人・年＝400億円」と計算できる。この数字は、きちんとデータを取っておればもっと大きくなったと思う。

この言語を使うために多くの管理者や技術者が、苦労し･努力し･我慢したことは明らかであるが、一番頑張ったのは私である。ゴルフでは賞金総額の20％が優勝者に与えられることから、400億円の20％の80億円が私の貢献であると勝手に考える。まさに捕らぬ狸の皮算用。

10－2　本当の貢献者は

客観的に見ると、私の提案を即座に取り上げ開発を指示したT氏、そして基本ソフトとして開発を決断したH氏の方が貢献が大きかったかもしれないが、いずれにしても実際に金の動かないヴァーチャルな話であるからまあいいとしたい。最近では、青色発光ダイオードの発明者のように、企業の研究者や技術者が自分の貢献に見合った金額を主張して裁判で争う事例が多くなってきている。私もやったろかなんてことは夢にも思わない。アホかと馬鹿にされるだけであるから。しかし､運がよかったといえ、そしてそれがヴァーチャルな金額であるにせよ、そのうえ単なる自己満足に過ぎないにしても、少なくとも自分の一生分の給料は稼いだとの自負は持てただけ幸せである。いままでの人生で多くの人に世話になったが、なんと言っても最大の恩人はT氏でありH氏である。そして、これよりずっと後になるが、私をこの鳥取環境大学に誘っていただいた元副学長の都倉信樹先生（現大阪電通大学学長）との３人にはいまだに足を向けて寝たことがない（？）。

11. 調子に乗って

ミニコンで成功したので、調子に乗って上下の方向で

売り込みを始めた。上方とは汎用機 TOSBAC-5600であり、アプリケーションは FORTRAN などの汎用言語で書かれていたが、OS やコンパイラ、データベースなど基本ソフトを開発するためのシステム記述言語は相変わらずアセンブラが中心であった。Ada 言語を記述言語にしようという動きが NTT を中心にあったが旨く行かなかった。汎用機を開発している青梅工場にのこのこ出かけて行き、基本ソフトの部長に PL/40と同じような記述言語を作りませんかと持ちかけた。何度か説得していると検討してみるということになり、担当の技術者も決まり、技術的な打ち合わせを始めた。何度か打ち合わせているとき、突然、東芝は大型機からは撤退し NEC と共同開発する（建前はそうでも実質は NEC に引き取られる）事になった。もちろん記述言語の話は没になった。その何年か後、東芝の大型機の残党技術者がオフコン的な DP（Distributed Processor：分散処理プロセッサ）というコンピュータ開発したとき、PL/40もどきのシステム記述言語を作っていた。私とは直接関係がなく何の挨拶もなかったが、間接的な貢献はあったと思いたい。

下方とはマイクロコンピュータのことで、当時は組込み用のマイコンとして TLCS-4という 4 ビットマイコンがあり、計測機器などに使われていた。計測機器事業部と半導体事業部に売り込みをかけ、同じような言語 PL/4を開発した。私は言語仕様を決め方式設計までをやり、プログラミングは外部の業者に頼み開発が完了すると私の手から離れてしまった。しかし、あとで聞くと、PL/4も TLCS-4のアプリケーションには結構使われ、秘めたるベストセラーであったとのことである。

12. CASE ツールの開発

12-1　日本中が CASE 熱に浮かれていた

私の成功譚は主任クラスまでで、そのあとは、結果として殆ど役に立たないことばかりやっていた。課長から部長になった頃は主として CASE（Computer Aided Software Engineering）の開発に携わっていたが、歳を食い地位が上がるに従い自分自身で設計やプログラミングをすることはなく、部下に任せっきりにしたり、関連のソフトウェア会社に外注することだけが仕事の中心であった。社外の委員会や学会活動、大学との対応などはやっていたが自分でプログラムを作らなくなってからは充実感がない。

CASE とはソフトウェア開発をコンピュータツールの支援で行おうというもので、広義に言えば OS も言語もエディタもすべて CASE である。しかし、その当時流行したのは要求定義、設計、プログラミングやテストといったソフトウェア開発ライフサイクルのすべてを、特に上流工程をツールの支援で開発しようとするものである。私の所属していた東芝システム・ソフトウェア研究所（以下（S 研）と略）では IMAP（Integrated software MAnagement and Production support system）と銘打って、ソフトウェア開発のライフサイクルだけでなく、進捗管理、品質管理など開発管理全般にまで支援するツール群の開発を行ったが、結局一部のツールを除いて殆ど使われることがなく終わってしまった。私はこの仕事の No. 2 として戦犯の追及を受ける立場にあった。

CASE 開発のブームはコンピュータ関連企業の各社（N、H、F、I）でも行われていたし、国家プロジェクトとしてもΣ計画が1985年から始まり250億円の巨費をかけながら殆ど役に立たないまま終わっている。日本だけでなく欧米でも日本のブームに釣られてか CASE 開発がブームになっていた。欧州では、英、独、仏、伊などを巻き込んだ esprit プロジェクトと言っていたがいつのまにか消えてしまった。

12-2　技術者の能力を上げることこそ本筋

プロの料理人は良く切れる刺身包丁で美味しい刺身を造る。では、アマが同じ包丁を使えばプロと同様の刺身ができるか？もちろんできるわけがない。CASE ブームの根幹には、いいツールさえ作ればソフトウェアの生産性や品質が向上するはずであるという、共同幻想を抱いていたことがある。ハードの世界では、大型の建築機械や自動化された機械があれば土木建築や製造工場での生産性は飛躍的に向上する。同様、ソフトウェアを開発するための大規模なツールを開発すれば生産性は大幅に向上するに違いないとの願望があった。しかし、ソフトウェア開発はほとんどが、技術者による知的生産活動でありツールがあれば便利かもしれないが、本質的でない補助作業を支援するだけで、結局は技術者の能力に依存する。「紫式部がワープロを使っておれば、源氏物語級の小説を幾つも書いたに違いない」という命題は明らかに偽であろう。

12-3　目くそ鼻くその戦い

当時、日本のコンピュータ関連の企業では、AI（人工知能）もブームになっており、私のいた（S 研）でも多くの研究者が AI に携わっていた。当時の AI とは、今考えるとエキスパートシステムや、ファジー技術、ニューロ技術でアプリケーションを開発することであり、人間の知的活動を人工的に行う AI：人工知能なんて言葉は

おこがましく詐称もいいところである。たまたまその時点で使えそうになった断片的な技術に、いくつかの応用が見つかったレベルにすぎない。

ソフトウェア生産技術の部長になった頃、（Ｓ研）長が「ソフトウェア生産技術の研究開発にもAIを適用しなさい」と言いがかりを付けてきたので、売り言葉に買い言葉「AIのようなチャラチャラしたものに研究者を割けません」と毒づいた。CASEもAIも結局期待したほど成果を挙げられなかったので五十歩百歩であったが、論文を書くだけだったらAIのほうがカッコよく有利であったようだ。この時の所長との論争が実を結んだか、半年後には部長をはずされ、研究主幹という体のいい窓際族となりはてた。上に盾突くと得にならないとの教訓を得たがもう遅い。

13. 東芝情報システムでの４年間

（Ｓ研）長が大学の教授になるので退職したとき、次の所長は誰かなと私を含む部長級４～５人が顔を見合わせていた。この中の誰かに決まるのかなと思いながら。しかし、（Ｓ研）以外の部門の、部長クラスのわれわれより３歳くらい若い人が（Ｓ研）長として就任した。若い所長に年寄りの部長たちが仕えるのはお互い精神衛生に悪かろうという会社側の厚意により、当然、古い部長級のお掃除が始まる。事業部に行くもの、大学の教員になるもの、関連会社に行くものなどしばらくすると見事に誰もいなくなった。

私も関連会社の東芝情報システムに行くことになった。役員や部門長でなく技術本部長付という名称、要するに窓際から窓際へ移っただけ。窓の外に放り出されなかったことに幸せを見つけるべきか。

事業部門から関連会社に移る時は仕事（お土産）もつけて移るが通常であったが、私の場合そんなものもなく「研究所から何しに来たの、お土産を持ってきたか」と丁重なお迎えを受けた。本社からの移籍人事により社長、総務部長、経理部長など重要部署をすべて占められてしまい、プロパー（新人の時からその会社に入社した人）はいつも冷や飯を食われている。腹いせに、私のような地位がなくいじめても反動のない立場のものが不満のはけ口になる。

東芝情報システムでの４年間で仕事らしい仕事をしたのは2000年問題の責任者となったときぐらい。あとは、給料をもらうために毎日出社はしていたがほとんど役にたつ仕事をしたという実感がない。ヴァーチャルには一生分の給料を稼いだはずだが、リアルには毎月の給料が必要であるから。

新人へのプログラミング教育を担当していたとき、またもプログラミングをしたいという欲望に火が付き、いろんなパズルを解くプログラムを作っては気を紛らわせていた。そこでは60歳の定年待ちの毎日、定年後には駅の自転車置き場のおっちゃんになることを夢見て。転機が生まれたのは鳥取環境大学に誘われてから。

14. 鳥取環境大学での10年間

東芝にいるとき、歳を食い地位が上がってくるに従い、社外の委員会や学会活動など、会社にとっては直接役に立たない仕事が増えていた。そんな関係もあり、鳥取環境大学の開学のとき教員に誘われたのだと思う。研究所勤めは不本意であったが、教えることは好きであるので何が幸いするかわからない。おまけに、大学の教授なんて思いもかけない身に余る名誉な職に就けたのだから。

大学での講義課目は、「ソフトウェア設計」、「オブジェクト指向設計」、「プロジェクト管理」、「データベース管理技術」、「言語と処理系」など自分の専門に近いものだが、幾ら教育を重視する大学はといえ、教えるだけでは面白くなく、といって研究なんかできるわけはなく、何か面白いプログラムを作成しようと考えた。

ターゲットは前から作りたかった碁のプログラムにしたが、かなり規模が大きくなりそうで、私だけではしんどいのと、無料の労働力を確保できる機会ともとらえ、卒業研究で私の研究室に配属される学生を巻き込んだプロジェクトにしようと思いついた。

碁を知っている学生が少ないので、まず研究室の本棚に碁の漫画「ヒカルの碁」を全23巻揃えた。研究室に来るチュータグループの学生や研究室配属の学生に、まず漫画を読ませ、それにより碁に興味を持たせる。興味を持ったあと、私が手ほどきをして碁を覚えさせ、囲碁プログラム開発プロジェクトの一員に巻き込もうという水も漏らさない万全の体制である。学生たちは喜んで漫画を読み、９路盤からはじめる私の碁の手ほどきに付き合ったが、１～２回付き合うだけで、碁のルールも理解しないで逃げ出してしまった。結局、無料で漫画を読まれただけであり、食い逃げされたようなもの。人生とは辛いもの、またまた夢が潰える。

碁のプロジェクトはあきらめ、個人的には毎年のように、オセロのプログラムやパズルのプログラム、あるいはπの新しい計算公式を発見するためのプログラムなどを作って欲求不満を紛らわせていた。情報処理学会のゲーム情報処理研究会で発表するなど年甲斐にない活動もしていた。

学生の研究室への配属は３年からで、碁のことはあき

らめ、配属された学生には、毎年「何か面白いプログラム、たとえばゲームのプログラムを作りなさい。２年間で１万ステップ以上のプログラムを作り、卒業論文は50ページ以上書くこと」と指示した。小難しいご託宣はいいから、まずプログラミングを好きになり、プログラミングに浸りきれば、自分なりの流儀を編み出してくれるはずであるとの確信からである。この10年間、研究室の学生の殆どは、オセロ、将棋、シューティングゲーム、RPG、神経衰弱など種々雑多なゲームやパズルのプログラムを作って卒業していった。内容はともあれ、学生時代に１万ステップのプログラムを作りましたというと、企業では、「ムム、なかなかやるなと」評価されるとともに、就職にも有利ではないかと思ったからである。結果的には私の与えたノルマに達した学生は数人しかいなかったが。

15. おわりに

平成22年度、退職する私にとって最後の学位授与式で、学生に学位記を授与したあと、「大筆　豊」の呼び出しがあり、学科長から「所定の講義を行い卒業研究の指導を行った」として卒業証書を授与された。この文章は、はじめの予定よりだらだらと大幅に長くなっておりとても卒業論文といえない。

東芝で28年、東芝情報システムで４年、そして鳥取環境大学でちょうど10年、都合42年の勤務期間を通じて、ソフトウェア開発と関係してきた。研究であるかどうかは別にして、自らソフトウェアを開発し、その経験を通じて自分なりの工夫はしてきた。自分の好きなことを続けられたことは幸せである。

鳥取環境大学を卒業した後、またもや、役にも立たないプログラム作りに勤しむか、あるいはほかの道を見つけるか。プログラムを作るのは相当気力が必要で、もう無理であるかもしれない。

（2011年７月29日）