ソフトウェア専門家を志す人のための基礎訓練科目

木村 泉† 大野浩之†

情報関係専門学科における．ある中級実験科目の設計とその実施経験について述べ，その種の科目のあるべき姿について論ずる．この科目の特徴は，(1)問題の意味そのものが明白でない，少なくとも平均的学部学生にとっては「得体の知れない」ような問題を取り上げ，(2)ソフトウェア開発の上流から下流に至るさまざまな問題点を体験を通じて総合的に理解させ，(3)報告書の書きかた，日誌のつけかた，キーボードのブラインドタッチなどのような基本的技能の強化を図り，(4)「終わりまでやらなければうちへ帰れない」状況を作り出すことによって，技術者としての自立心とスタミナを養うことにある．ことに(4)は，現代日本の社会状況においては重要，と考える．わが国の企業情報システム専門家の間では，大学における計算機科学教育の意義を疑問視する発言がしばしば聞かれる．この科目は，そういう声への一つの答えをなしていると信ずる．

A Laboratory Course Stressing Personal Growth for Future Software Professionals

Izumi Kimura† and Hiroyuki Ohno†

A middle-level laboratory course in an "information-oriented" department in a Japanese university is decribed, and its philosophy presented. Four points are raised: (1) it is built around a "fuzzy" problem, of which the definition itself is unclear; (2) an overall understanding by the students of the various aspects of software development is aimed at by excercises simulating real experiences; (3) supplementary training is given for writing a good technical paper, keeping journals successfully, and improving typing skills; and (4) the students' stamina is enhanced by creating a "no time limit" enviroment: no one, including the instructor, is allowed to go home until his or her job is done. The point (4) seems to be particularly important in the present Japanese social situation where some students, by being given too much parental protection, tend to remain childish. This course offers an answer to the tradition in Japan's information systems community to discount the value of the computer science education in universities.

1. はじめに

筆者らは，現在東京工業大学理学部情報科学科において，「情報科学実験第二」と称する科目を担当している．学部3年次前期開講の必修科目で，現在は2単位の実験科目として名目3時間（正味2時間半）の枠を週2回割り当てられている．実験科目としては，2年次前期の「計算機実験」(Pascal によるプログラミング)，2年次後期の「情報科学実験第一」(AI プログラミング的な内容，現在は Scheme によっている)に続くものであって，中級実験科目として位置づけることができる．カリキュラムの他の部分との釣り合いを考慮して，この科目には「ソフト屋の基礎訓練」とでもいうべき内容を持たせてある．その現状を，1991年度の経験を中心として報告し，情報関係専門学科における実験科目について筆者らの考えを述べ，ご批判を仰ぐ次第である．

この科目は，筆者の一人（木村）が途中何回かの中断をはさみながら約20年間にわたって担当してきたものであって，文献1)にその一端が示してある．その特徴としては次の(1)～(4)を挙げることができる．このうち，(4)は1990年度以降の新機軸である．

(1)この科目では，問題の意味そのものが明白でない，少なくとも平均的学部学生にとって「得体の知れない」ような問題を取り上げる．これによって，学生にありがちな「教師によって与えられた明確な問題をもとに，所定のプログラミング言語で，それらしく動くプログラムを作ればいいのだ．」という先入観を打ち破り，ソフトウェア開発の上流工程に目を向けさせる．

現実の問題は，内容がはっきりしていることはむし

† 東京工業大学理学部情報科学科
Department of Information Science, Tokyo Institute of Technology

ろ稀である．さまざまな解釈の余地を含んでいることが多い．また必ずしも決まり切った技法ですんなり解けるとは限らない．役に立つ技法なら何でも動員するぐらいの覚悟がないと間に合わないのがふつうである．だから「得体の知れない」，「こうも考えられる，ああも考えられる」ような課題を選ぶ．

もちろん大学の授業の枠の中でフルサイズの現実問題を取り上げることは無理であるから，規模を縮小した課題を取り上げ，模擬演習の中でエッセンスをとらえるように工夫する．これは，要求分析というものの意義を学部学生にわからせるための，ほとんど唯一の方法だと考える．

(2)またこの科目では，ソフトウェア開発の上流から下流に至るさまざまな問題点を，個別にではなく総合的に，体験を通じて理解させるように努める．たとえばシステム分析をおもな話題として取り上げている間も，モジュール分解の問題や効率の問題とのからみが生じたときには，それらを遠慮なく話題に含める．実際，よい設計家は上流工程を取り扱っているときでも，下流への目配りを失わないものだという[2]．なお，(1)で採用した問題を，(2)でもなるべくそのまま利用することによって，話題の流れに一貫性を持たせる．

(3)一方では報告書の書きかた，日誌のつけかた，キーボードのブラインドタッチなどのような基本的技能の重要性を強調し，それらに関する実習の機会をなるべく多く用意する．これらのことは，学習者本人が「なるほどこれは大切だ」と真に納得するまでは，なかなか身につくものではない．そこでわれわれの科目では，(1)，(2)の流れの中でこれらの技能の重要性に，おのずと気づかせるように工夫する．

(4)そして最後に，実験の実施に当たっては，「課題をやってしまうまではうちへ帰れない」ことをはっきりさせる．そのため，教師（2名）が代わり合って，原則として学生全員が目標の水準に達するまで，とことんつき合う．徹夜もいとわない．これは，技術者としての自立心とスタミナを養うこと，そしてひいては学生たちの人間的成長をうながすことが狙いである．このことは，現代日本の教育状況にあっては，ことのほか大切であると信ずる．

以下，まず科目の内容を具体的に説明し，そのあと上記の特徴について，さらにくわしく論ずる．

2. 現時点での科目内容

今期（1991年度前期）たまたま取り上げた話題は，次のとおりである．

要求分析，フォーマルな仕様，プログラム書法，日誌，キーボードのブラインドタッチ，文章作法（特に技術文書），レビューと査読報告，性能評価とチューニング，新しいシステムとの出会い，CLU言語[3]，データ抽象，プログラム設計，情報隠蔽

与えた問題文の主要部を**表1**に示す．配布した資料にはこのほか，補足説明，参考資料，および一部の課題についての「正解」が含まれている．表1の各課題には，それが上記の話題のうちのどれに対応しているかを【，】で括って書き添えておいた．

科目全体をつなぎ合わせる縦糸となっているのは，人名を権威ある国語辞典の配列規則に準拠して配列する，という問題（表1の課題1参照．以下「広辞苑問題」と記す）である．ただし注文主（問題文にいう「叔父さん」）は計算機に関する知識をほとんど持っていない，というところが工夫である．この問題をもとに，ソフトウェア開発のさまざまな側面を，学生にとって現実感を持てるような場面設定のもとで，シミュレーションすることができた．

課題8と課題9の間では，CLU言語[3]の手短な説明と簡単な実習をおこなった．その部分が上記の「新しいシステムとの出会い，CLU言語」というところに対応している．これらの課題は，前もってくわしく計画したものではなく，進行を見ながらその都度決めていったものである．

理工系の実験科目では，いくつかの課題をあらかじめ設定し，課題ごとにブース（実験設備）を用意し，学生にそれらのブースを巡回させるという方式がある程度常識になっているが，この科目では全員（1991年度は42名）に一斉に同じ課題を与えることにしている．われわれの科目は，技能を高め知識を確実にする，という実験科目の常識的目標に加えて，（結果的には）学生個人の人間的成長をうながすための科目という色彩を帯びている．その意味では一斉方式の方が，（計算資源の事情さえ許せば）学習効率がよいように思われる．

表1の課題の大部分は何となく謎めいており，「掘り下げ出すときりがない」性格を持っている．通常の意味での正解は存在しない．

にもかかわらずわれわれは，正解のない問題なりに

表 1 1991年度問題一覧（その1）
Table 1 Excercises in the 1991 Course (Continued.).

課題1【要求分析】
昔から大変世話になっている叔父さんとか、お父さんの会社のお得意先とかいった、ちょっと断わりにくい筋から相談が舞い込んだと思ってほしい。一応かなの振ってある人名簿がコンピュータに入っている。ただしそれはめちゃくちゃな順序に並んでいる。名簿に載っている人々を何回かにわけて招き、盛大なパーティーを開こうと考えている。毎回ごとの参加者名簿を作りたいのだが、その序列が問題である。

偉い順ということも考えられるが、それだときっと自分はあいつより偉いのにあいつの方が上席だ、などといいだす人が出てくる。だから名簿は厳密なアイウエオ順にして、そういう不平の出ようがないようにしたい。それをコンピュータでやってもらえまいか、という。また、使うアイウエオ順は、権威ある国語辞典（広辞苑か何か）の見出し項目の配列順序にぴったり合わせたい、という。そうしておけば、文句が出ても申し開きができるからだそうだ。

相手は気がせいている。2週間でできないか、といっている。

これまでに身につけた技術とセンスを駆使して、何とか相手を満足させてほしい。ただしコンピュータは、まだ使えないものとする。世の中の本物のプロのやりかたを見ていると、最初からコーディングをする人など誰もいない。最初は紙と鉛筆（またはワープロ）で仕事をする。諸君もそうやってみてほしい。

なおこの課題は諸君個人の腕試しであるから、今回は独力でやること。他人のプログラムを引き写すなどはもってのほかであり、失格となる。

課題2【フォーマルな仕様】
広辞苑の見出し語配列方式について調査し、見出し語配列プログラムを書くとすれば必要となるはずの知識を漏れなく抽出し、それを何らかの厳密な記法で書きあらわせ。次にそのどの部分は自動化でき、どの部分は（現在の技術では）人手による介入を必要とするか検討せよ。

課題3【プログラム書法、日誌】
前回解析した広辞苑の項目配列方式を、適当なプログラミング言語を用いて実現せよ。ただしたとえば3000件程度の人名をパソコン上で、待っていていらいらしない程度の時間で整列させるという状況を仮定する（実現自体はパソコンでなく、たとえば手近なワークステーションでやって構わないが、最終的にはパソコンに持ってゆけるように配慮する）。そして項目間の前後関係を判定するためのサブルーチン（手続き）と、（もし必要なら）データ項目を前処理するためのルーチンだけを作る。たとえばヴァからバへの変換は最初にまとめてやってしまい、前後関係の判定の際には一々ヴァのことを気にしないでもよいようにする、などの切りわけが考えられる。整列のアルゴリズム自身については当面は考えなくてよい。データが3000件あると、整列を浮上法（bubble sort）など使って「たらたら」やったのではずいぶんひどいことになる可能性があるが、その問題は一応前後関係判定アルゴリズムとは切り離して考えてよい。そこそこ速い整列アルゴリズムを使ったとき破滅的な結果にならない程度に賢い前後関係判定手続きができればよい。

諸君の作業の詳細は、日誌につけること。日誌は年月日、時分を付して作業の内容および気づいたことをどんどん書く。ただしその場で書かなくてはならない。また書いたことをあとでみだりに変更してはならない（変更するなら前の内容もわかるように、2本線で消して書きたす）。紙切れでなくノートに、鉛筆でなく簡単には消せない筆記用具（証券用のボールペン、万年筆など）で書く。ワープロで書くのもわるくはないが、ついあとで書きなおしたくなるので注意。いつ何という字を打ったかが自動的に記録されるような環境（たとえばＵＮＩＸのscriptによる）でない限り、ノートにボールペンの方がまさる。

諸君の前後判定手続きおよび前処理ルーチンは、なるべく問題の起こりそうな境界点においてテストすること。どのようなテストデータでテストし、その際どのような結果が出るはずであるかは事前に書き出し、書き出したノートを見ながら実際のテスト作業をすること。あとでこちらで用意したテストデータで再テストをしてもらうかもしれないので、そのつもりで。

課題4【キーボードのブラインドタッチ】
ＱＷＥＲＴＹ鍵盤配置を増田法を用いて練習せよ。最初に全体の配置を「指に覚えさせ」る練習をおこない、そのあとは1週間にわたって毎日３０分以上、実作業をする。作業内容は指定しないが、打つ内容として個人的興味があるものを選ぶとよい。打鍵状況（何を何分何秒打って出来高は何文字だったか、どんな間違いをしたか、など）は日誌に書き留めるほか、別に説明するnewscriptプログラムで記録すること。日誌のまとめ、文書ファイル、およびtypescriptファイルを提出する。日誌のまとめには積算練習時間数と１０分間当たりの出来高文字数の間の関係を示す、両対数表示のグラフを含めること。

課題5【続・プログラム書法】
a．コンピュータのプロにとって、プログラムをすっきりさせるためのノーハウは、ぜひ身につけておかなければならないものである。最小限のノーハウを知るために、カーニハンほか著「プログラム書法」をよく読んでほしい。気づいたことは日誌に書くこと。

b．別に示すプログラム（学生作品、一部改変）から、書法という見地から改善の余地がある場所を見つけ、「プログラム書法」の「規則」を参照しながら、それをどのように改善したらよいか述べよ。

課題6【文章作法】
実験の結果を簡にして要を得た報告書にまとめる能力は、理科系の人間にとっていくらあっても、ありすぎということのないものである。せっかくちょっとした実験（キーボードの習得に関する）をしたのだから、一つの練習として、その結果を論文っぽい形にまとめてみてほしい。

何か下敷きがないとやりにくいだろうと思うので、今回は電子情報通信学会の投稿規定を借用し、それに合うような形式のものを書いてもらうことにする。刷り上がり２ページの「ショートノート」を想定する。

もちろん今回の実験の結果をそのまま記述しただけでは、学会雑誌の論文として受理される見込みはない。とはいうものの、たとえ内容はそうだとしても、形式については文句をつけさせないぐらいのつもりでやってほしい。章立て、図（グラフ）の形式、文章のめりはりなどには特に注意すること。

課題7【レビューと査読報告】
課題５のｂで言及した学生作品を、書法の点から検討し、学会雑誌の投稿論文に見立てて査読報告書を書け。

課題8【性能評価とチューニング】
各自の整列プログラムを、プロファイラを利用してスピードアップしてみよ。たとえば3000件の整列が１秒、というあたりまでゆくだろうか。

課題9【ＣＬＵ言語、データ抽象】
広辞苑第２版の見出し配列順序を表現するクラスタを設計、製作、検査せよ。なおその際、基準とする広辞苑が第３版に変更になったとしても、必要な手なおしが一つのル
（続く）

何かしら正解らしいものが出てくるまでは，原則として「じゃあ次回までに考えておいで．」といわない．ただし問題選びは慎重にし，少なくとも何人かは，名目３時間の枠の中で正解らしきものに到達するように配慮する．枠の一つは昼まで，もう一つは午後３時からとなっていたので，昼までの枠では「時間切れ再試合」を認め，３時からの枠ではよりしつこく「ノー」という，というようにした．

各課題についての特記事項は次のとおりである．

課題１は「広辞苑問題」を，特に要求分析という側

表1 (その2)
Table 1 (Concluded.)

ーチンまたはクラスタに限定されるよう、工夫することが望ましい。

課題9 a【プログラム設計】
前回設計したクラスタを実現するために、どのような下請け機構（クラスタ、手続き、イテレータ）を用意すべきか、広辞苑第2版の凡例をよく読みなおした上で、前回の解答と同様のくわしさで書き出せ。

課題9 b【続・プログラム設計】
前回の課題（課題9 a）のようなものについて考えてみるための一つの方法は、課題の仕事をやるとしたらむずかしいところはどこか、と考えてみることである。そして、やりようがとっさに思いつかない事項について、「それはどうしたらいいのだろう」と自分にたずねる。たとえば次のような問いを立てる。

a．日本語文字（漢字、かななど）を表現するにはどうしたらよいか。振りがなの中で使われるかな文字についてはどうか。
b．日本語文字列はどう表現したらよいか。また振りがなとして使われるかな文字列についてはどうか。
c．かな文字列の前後関係を決める際には、濁音、半濁音のついた文字と清音の文字の間の前後関係は、どう扱ったらよいか。小さい「ぁぃぅぇぉゃゅょっ」とふつうの「あいうえおやゆよつ」の前後関係はどう扱うべきか。
d．読みが2文字以内の姓は親項目に立てない、というあの奇妙な規則はどう扱ったらよいか。漢字1文字から成る字音語は（たとえ読みが3文字以上であっても）親項目に立てない、というそれよりもっと奇妙な規則はどう扱ったらよいものか。
e．漢字1文字から成る同音の字音語は、画数の少ない順に並べるという規則はどう扱ったらよいか。
f．ヴァをバとして扱う、という話はどう扱ったらよいか。
g．長音の処理（「ハーモニー」は「はあもにい」として扱う、など）はどうやったらよいか。

そうやって問いを立てたら、次にはそれらの問いについて、1問当たり1個のクラスタで処理できないかな、と考えてみる。もちろん立てた問いを全部が全部、そのままクラスタにすればよい、というわけではない。相互に関連した問いは、まとめて処理しないとモジュール間に依存関係ができてしまう。また二つ以上の本質的に違った部分を含む問いに対しては、対応する二つ以上のクラスタを用意するようにしないと、モジュールの凝集性がそこなわれる。だから問いは、整理しながらやらなければならないが、この考えかたは発見の手法として、有用であることが多い。
問いa～gに対応するクラスタの骨格を作れ。木曜日には各クラスタを別人に作ってもらって、それが打ち合わせをなしでピチッと「はまる」かどうかテストしたいと思っている。そのつもりでがんばってください。

課題9 c【情報隠蔽】
3人を1班とし、2班を1チームとし、班の各人は1名は人名項目、1名は入出力（文字、文字列を含む）と全体の制御（start_up）、1名は整列を受け持つものとし、2班で話し合って各モジュールの仕様を定め、その上で各人が独立に各モジュールを作成し、それを持ち寄り、人名項目、入出力、整列の各モジュールから成る8種の組み合せが全部正しく動作することを確認せよ。人名項目の順序づけには、上記の分析にかかわらずJISコードにおけるかな文字（カタカナ部分）の並びかた（collating sequence）をそのまま利用してよい。

（終わり）

面から眺めた課題である．この課題では，問題を見たとたんに計算機に飛びつきたくなるのをどう我慢するか，というところを中心問題とした．現実の問題にぶつかったとき，即座にプログラムを書きはじめたい，という衝動を我慢できない学生は少なくない．そこで学生たちをコンピュータが使えない状態で演習室に閉じ込め，その状態で何ができるかを考えさせる．それでもなお，N88 Basic のプログラムを書きはじめた猛者があった．

次回に，プロならこんなことを考えただろう，と思われる事項の一覧表を示した．挙げたのは，

- 「一応かなの振ってある人名簿」という，その「一応」とはどういう意味か．
- 名簿が「コンピュータに入っている」とあるが，それはどういう形式か．叔父さんがコンピュータといっているのは，実は○○くん（ワープロ専用機）かもしれないぞ．
- そのコンピュータは叔父さんのところの所有物か．どこかのデータセンターと委託契約を結んでいて，データ一切を相手に握られているという可能性もある．こちらには電子記録は触らせてもらえないのかもしれない．
- 叔父さんのところには，コンピュータ要員がいるのか．いるとしたら，どの程度の技術力を持った人か．こちらが乗り込んで行ってばりばりやったら，彼らが気を悪くするという可能性はないか．
- 単にコンピュータでやることだけが大切なのか．それともコンピュータでやらなければ間に合わない規模なのか．
- 広辞苑式順序とは具体的にはどんなものか（ちょっとリサーチしておかなくちゃ）．
- そもそも叔父さんは，この件についてどのくらいの権限を持っているのか．また権限を持っているのだとしたら，本当に当方にやらせる気があるのか．それはどうやったら確認できるか．

など，計16項目であった．

課題2は，上記の広辞苑式順序に関する「リサーチ」である．状態遷移図を使う学生，疑似コードを書く学生，HCP（NTT 式構造流れ図）を描く学生などがあった．あとで，命題論理の式を書くというのもあるよ，といってやった．

課題3は，学生たちのプログラミング能力を見るために課した課題である．まず昼までの枠でできるところまでやったあと，続きは午後の枠でやる，ただし事

前にやっておけば早く帰れる，ということにした．使用言語は指定しなかったが，2年次前期に Pascal を習っている関係上，(UNIX 上，またはパソコン上の) Pascal を使った者が多かった．自宅のパソコンのCで下書きしてきた，などという者もあった．ここでは徹夜にはならなかったが，終電車がなくなって大学に泊まった学生がかなりあった．

なお日誌に時分を要求したのは，訓練のためである．日誌には必ず時分を書くべきものだ，と考えてそうしたわけではない．

課題4は，見かねて取り上げた予定外の課題である．学生たちにあまりにも一本指派が多すぎると気づいて取り上げた．右手と左手をピアニストのように交差させて，左手の置き場所よりもっと左寄りのキーを右手で打っている者さえあった．たまたまこの年度の学生は，2年生のときタイピングの練習をする機会を持たなかった模様である．なお，問題文で newscript プログラムといっているのは，UNIX の script コマンドを若干改変して，改行が打たれるごとにおおよその時刻を記録する，という機能をつけ加えたものである．

課題5も，プログラム書法の身についていない学生がかなりの数いたために取り上げた予定外の課題である．Pascal において，すべての変数が大域的に宣言され，n1 とか data2 とかいった変数名を与えられているという具合の，教師にとって読みようのないプログラムがかなりあった．参考書は，たまたま相当部数手元にあったので，それを希望者に貸し出した．あとで日誌を提出させて確かめたところ，実際にはあまりきちんと読んだ形跡のない者が多かった．本の読みかたがわかってない，という問題があったのかもしれない．

課題6は予想以上に適切な課題であった．文章作法に関する話題は例年必ず取り上げるが，学会雑誌の厳格な枠の中で書くというやりかたは，話がはっきりするのでことのほか有効である．またショートノート形式としたお蔭で，教師にとっても無理なく一息で読み，その場で批評する，ということができた．この課題をやらせてみて，学生たちがどれほど子どもっぽい「感想文」に毒されていたか痛感した．多くの学生が，この課題の終了時にはそれなりに大人びた，客観的な文章を書くようになった．この課題では，教師の側の成熟度が大きくものをいうことを痛感した．

課題7は，CLU 言語[3]の処理系（課題9以降で利用）の予定完成時期をにらんで，時間つなぎに取り上げた課題である．学会雑誌のための査読報告書の実例を，具体的な細部を隠して骨格のみ示し，それに倣って書くように勧めた．確かに有益な課題であったが，課題6と重複したきらいがある．

課題8（プログラムのチューニング）は，計算分野のコアコースのどこかでぜひやっておかなければならない課題である．ここで，3,000 件で1秒というのは，アイドル歌手の姓名から成る原ファイルを，広辞苑の複雑な配列方式に従ってワークステーション (SONY 製，CPU は R 3000，クロックは 25 MHz) で整列させることを想定している．時間は実時間である．すでに課題3の段階で，C言語のプログラムによって 0.7 秒という成績を出していた学生が1名あった．この課題で，はじめて本当に徹夜した者が何人かあらわれた．

課題9以下は CLU 言語の，できたての処理系（学生たちの先輩が作った．CLU をCに翻訳して処理する）を使って，データ抽象と情報隠蔽に関して学ぶ課題である．学生が内容をよく知っている題材を使いたいという意図から，引き続き「広辞苑問題」を用いたが，学生によっては「またか」と感じたものもあった模様である．

全体として，課題の選択に当たっては，どこがむずかしかったかが強い感情的体験を伴って印象に残るようなものを選ぶように心がけた．

取り上げたかったが，今回は主として時間的制約のために割愛した話題には，たとえば次のようなものがある．いずれも過去の，ある年度には取り上げたことのある話題である．

- インタビュー演習（課題1にいう「叔父さん」に会って，情報を引き出す）
- タイムプレッシャーのもとでの模擬システム構築（ワインバーグのアイディア[4]による）
- 公式的レビュー（レビューミーティングを開く）
- 既成の方法論に即した分析，設計（ちょっとやってみる程度）
- 英文和訳演習
- 状態遷移図とプログラムの間の等価変換（やってみると案外できない）

3.「徹夜がらみ」の意図と効果

「広辞苑問題」は，第1章で掲げたこの科目の特徴(1)～(3)を具体化する上で，大変有効であった．

実際2，3週間のうちに多くの学生たちが，課題1の「叔父さん」について，それがあたかも実在の人物であるかのような話しかたをするようになった．数週間後「広辞苑でやれ，というのは無理で損なやりかたなのだが，相手は2週間でやってくれなどといい出すぐらい素人だから，そこを納得してもらうのはものすごく大変だよ．」と話してみたところ，実在の「相手」に関するきわめてもっともな警告として，すんなりと受け入れられた．

またこの問題からは，ソフトウェア開発のさまざまな局面に関する多様な演習問題を，ごく自然に導き出すことができた．表1の課題9にあるとおり，広辞苑の第2版と第3版がかなり違った配列規則を用いている，という事実を利用して，仕様変更に関する経験をさせることができたのも大きなプラスであった．

しかし本文で特に強調したいのは，科目の特徴(4)，すなわち「徹夜がらみの実験」というところである．その狙いは次のようなところにある．

- 自分のことは自分の責任で解決するものだ，と覚悟すること．
- ものごとを最後までやり遂げるのに必要なスタミナを身につけること．
- 時間のやりくりができるようになること．そしてタイムプレッシャーのもとでも平気でいられるようになること．
- そしてひいては，一人前の大人になること．

これらが，プロの技術者にとってどのくらい大切なことかは，あえて説明するには及ばないであろう．これらについて，何らかの手を打つ必要がある，と気づいたのは次のような事情によってであった．

われわれの科目では，前々年度までは「フレックスタイム制」をとっていた．問題の説明は時間割どおりの時刻に集まってもらって実施するが，実際の作業は夕方でも夜でもいいから計算機が空いているときおやり，というようにいってあった．計算資源の関係上そうでないととても大変なことになる，という事情もあった．それでも「計算機は大っきらい」という少数の学生以外は，それなりによく勉強した．謎めいた問題は，計算機の好きな学生にとっては巨大な問題に，そうでない学生にとってはそこそこの問題に見える．筆者らの問題を巨大な問題と見て（ないし見誤って）われわれが予測した以上のことをやり遂げ，それによって大きく成長した学生が，毎年かなりの数にのぼった．

ところが近年は，段々そうでもなくなってきた．ほとんど全員が問題を矮小化してとらえ，最大限の手抜きで間に合わせる，というように変わってきたのである．その上学生間に，筆者らの学科は「楽学科」だ，とのうわさが流れはじめた．その時期は，入試に共通一次が取り入れられるようになったのとほぼ符合している．因果関係について決定的なことをいうには分析が不足であるが，問題を深読みしすぎる学生は多肢選択式試験では好成績を収めにくいはずであり，したがってそこに何らかの因果関係があっても決しておかしくはない．いずれにせよ，これでは卒業生を自信を持って世に送り出すことはできない．

そのようなことが起こった原因としては，次の二つの，互いに関連したことが考えられる．

青年がいつまでも自己陶酔的な少年，というところから抜け出せないというのは，(表面的に，または実質的に）豊かな社会にありがちなことである．われわれの社会には，「そのうち何とかなるだろう」と思ってだらだらと時を過ごし，「何とかならなかった」ときもその責任を自分で取ろうとせず，どこからか救いの神があらわれるのを期待する，という生活スタイルがじわじわと広がりつつある．社会が豊かであると，「何とかならなかった」場合でもそれほどひどく破局的なことにはならず，何となく「顔が立ってしまう」ので，ますますその傾向が助長される．最近はこの傾向が，大学生の間にまで浸透してきた．これが第1の要因である．

第2の要因は，われわれの社会に広まっている「代用特性志向」である．多肢選択テストで勝利を収めるために思考方法を変える，というのはその顕著な例である．いっそう顕著な例は，大学受験における偏差値であろう．本来は単なる目安（代用特性）にすぎない偏差値を最適化するために，多くの若者が目を血走らせている．いったん偏差値競争に勝って大学入学を果たした若者のかなりの部分が，究極の代用特性として次には「楽さ」を選択する．最小限の手間で何とか卒業することが最終目標であるかのように振る舞う．

このように楽さを一途に追い求める風潮が広まっているいま，従来のやりかたを無反省に繰り返したのでは社会的責任が果たせない．そこで基本的な筋書きはもとと同一ながら，適当なことをほどよくやる代わりに，無茶苦茶なことを元気よくやることにした次第である．最近実習用計算資源の事情がわずかながら好転して，そういうことが必ずしも不可能でなくなったの

は幸運であった．また学生数が絶望的に多いというほどではなかったのも，ありがたいことであった．

粋がって徹夜をさせるのがいい，というのではない．問題の性質によって，定刻で打ち切る，女子学生の通常の門限に間に合わせる，終電車には間に合うようにする，などを使いわけた．徹夜になった回もたまたまそうなっただけで，計画的に徹夜にしたわけではない．要は，場合によっては徹夜もありうる，という認識の方が大切である．そういう認識があってはじめて，学生たちは本気を出す．そしてそういう認識が生じうるためには，教師の側でも徹夜を当然と思っている必要がある．

このようなやりかたの最大の長所は，学生が自分で考えることを恐れなくなることである．ひいては自立心が育ち，「ガキっぽさ」が薄らぐ．ことに1991年度は，学期が終わりに近づくにつれて，多くの学生たちが目立って大人びた表情を示すようになった．特に次の経験は印象的であった．

この科目でも，はじめのうちは「今週は○○のリポートの締め切りだから，実験のリポートはなしにしてください．」などといってくる学生が多かった．「ふうん，それじゃ……」などと，プレッシャーを緩めることもしたが，回が進むにつれて徐々にそういう妥協を減らしていった．そして次回が最終回というところで，「次回は，うまくゆけば暗くなるまでにはできるはずだが，いままでの課題が完全に身についていなかったとすれば，もっと時間がかかるかもしれないような課題を出す．これはお祭りだから最後までつき合ってほしい．」と予告しておいた．

その課題とは表1の課題9cであって，6人ずつにわかれてのグループ課題である．この課題は，全員の作業が終わらないとグループとしての課題が終わらない仕組みになっていたこともあって，深夜になっても完了するグループがなかった．まあとは週末にでもおやり，と口まで出かかったが，あまりにも白熱した雰囲気なので進行を止めるのがはばかられた．結局翌朝8時半にストップをかけた（それまでに完了したのは2グループ）が，実はその「翌朝」が○○のリポートの締め切り日だったということが，学生の話でわかった．「あれれ，大変だね．」といったところ，「いや，こうなると思ったから前日にやっておきました．」との返事であった．ずいぶん大人になったものだと感動した．

ただしこういうやりかたをすると教師は，情報提供者ではなくむしろ学生たちの成長に立ち合う受容者，という立場に立つことになる．自分という人格自体が，最大の資源として酷使される[5]．肉体的はもとより精神的にもきつい，ということは認めざるをえない．

4. その他の考慮点，その後の経過

4.1 個人作業の強調

現在のところこの科目では，共同作業よりは個人作業を強調している．日誌をつけさせたのにも，一部そういう意図が含まれている．そうしているについては，次のような事情がある．

かつては，まったく共同作業のできない自閉的な学生がかなりいたので，仲間とのコミュニケーションをうながす課題を多く取り上げていた．ところが最近は，少数の気の合った仲間とうまくやることに異様な関心を持つ学生がふえてきた．特に1991年度はそれが行き過ぎて，あらゆる課題を一定の仲間うちで「つるんで」（しかもグループ外の誰かがアイディアを出すまで何もせずに待ち，それをすばやく盗んで）解き，同じリポートを一見各人でやったかのように偽装して提出するという，おそらくは6名から成るグループが発生し，苦慮した．アイディアを盗むこと，仲間で協力し合うことは，いずれもソフトウェア技術者にとって基本的な心得である．だがそれは，元気よくやるのでなければ意味がない．TA (Transactional Analysis—Eric Berne によって創始された心理療法の一流派)[6] の言葉でいえば，「自由な子ども」の自我状態ですべきことであって，「順応した子ども」の自我状態でしたのではつまらない．

4.2 遅刻，欠席

ゆっくりコーヒーを飲んで，あとからのっそり入ってくるなどという学生が，はじめのうちは稀ではなかった．授業に遅れてくるのは，コマーシャルがすんだ頃合を見て，テレビの野球中継のスイッチを入れる，というのと共通した心理のようだが，それは損だということをわからせる必要がある．心を鬼にして，出席をいやらしいほどやかましくした．もちろん教師としても，開始時刻を厳密に守ることが必要である．教師が遅れてきたのではサマにならない．

4.3 身体的ストローク

途中でこれは徹夜だなと気づいた回には，パック入りの紅茶とクッキーを少しばかり供する，というようなこともした．精神的にきついのを身体的なもの

(TA の言葉でいえばストローク[6]) で補償する，というのは大切なことである．

4.4 演技力の問題

扱う問題の性格上，事前に完全な模範解答を用意しておくことは不可能である．しかし何か模範解答めいたものを作ってみる，というところまでは不可能ではない．それをしておけば，課題に無理がないことを事前に確認できるので，自信を持って指導ができる．

だがその場合には，教師は「虚構を心から信ずる」という，近代リアリズム演劇でいわれる意味での，真の演技力を持っている必要がある．謎だ謎だといいながら，実は全部の答えを知っている教師は，たちまち学生を小手先で操ろうとする不誠実さを見破られることになろう．「いざとなれば答えを出す自信はあるが，自分も答えそのものは知らないのだ．」と心から信じ込み，学生の苦しみをともに苦しむことが必要である．そしてそれができるためには，よい健康状態と安定した精神状態を保つことが必要である．

4.5 IS への答え

わが国の企業情報システム（以下 IS と記す）専門家の間には，社会にとって大切なのは SE（システムエンジニア）だ，SE には CS（計算機科学）などいらない，大学で教えてもらってもしかたがない，と強く主張されるかたが少なくない．この科目はそういう主張への一つの回答をなしていると信ずる．

IS の構築に係わる SE の仕事が，社会にとって大切なものであることはいうまでもない．また SE の仕事が，CS の知識だけあればできるようなものでないことも事実である．それ以外にも必要なことはある．特に大切なのは対人的な力である．ワインバーグ[4]の表現を借りれば，技術革新の力に加えて，動機づけおよび組織化の力を持っていることが必要である．だが情報処理の根幹に当たる CS の基礎を身につけていない SE は，英語が読めない英文学者のようなものだといってよい．技術革新の力のまったくない SE に，きちんとした仕事ができるとは信じがたい．

この科目は，CS に関する知識，技能の問題を，そのような対人的な力，ないしさらにつきつめていえば人間的成長の問題と不可分なものとして取り扱っている．この科目は，IS 関係者の不満に答えるものとして，意義あるものと考えている．

4.6 負担の問題

教師，学生の双方にとって，この科目が大変きついものであることは否定できない．実際，すべての科目がのべつ幕なしにこのようにきついのでは，学習上かえって逆効果であろう．緩急の使いわけが必要である．このような科目は，入学から卒業までの4年間の間に，せいぜい2，3科目もあれば十分である．むしろその方が学生の印象に強く残り，学生の個人的成長をうながすというこの科目の狙いも，よりよく達成されるものと考える．

4.7 学生のレベルの問題

この種の科目を実施するに当たっては，学生の成長度を十分考慮することが必要である．本文で示した課題例は，時間内に一応の「完全答案」に達する学生が2，3人程度出る，というように選んだ．もし学生の基礎訓練が進んでいればもっとむずかしい問題を，遅れていればやさしい問題を選ぶ必要がある．さらに，困難にぶつかったとき降参してしまわないための強さがどのくらい身についているかも考慮しなければならない．場合によっては，上の「2，3人」というところは，「2，3割」とか「7，8割」とかに調整する必要があるかもしれない．

4.8 ほかの人たちの仕事

たとえば高橋延匡氏ら[7]の KJ 法など取り入れた仕事は，基本思想において共通するところが多い．また科目全体の構成について，ワインバーグ氏のワークショップ[4),5]から多くのヒントを得た．身体的ストロークを取り入れるというアイディアなども，同氏から得たものである．

4.9 次年度の経験

1992年度には文献1)の文書整形問題をおもな題材として採用し，最初から Pascal やCではなく CLU 言語を用い，ソフトウェアの設計段階に関する話題を多く取り上げた．これは前年度と同じことをするのでは芸がないと思ったのと，CLU 処理系がある程度しっかりでき上がっていたことによるが，内容がやや平板になったきらいもある．ただし1991年度には割愛した①正規表現に関する問題，②英文和訳演習，および③擬似システム構築問題を取り入れることができた．③は最終回に実施し，終了時にビールを出した．ほんのコップ1杯であったが，雰囲気は大いに盛り上がり，学生たちに達成感を味わせることができた．

なお「土曜閉庁」の影響等によって，1993年度以降この科目の単位数は2単位から2.5単位に増され，割り当て時間は名目6時間から5時間に削減されることになっている．

4.10 この科目では扱わなかった事項

はじめにいったように，この科目で何を扱うかはカリキュラムの他の部分との釣り合いを考慮して決めた．たとえばコーディング技法，デバッグ技法，およびテスト技法については，すでに基礎的なことを先行科目の中で学んでいる建て前なので，ここでは軽く触れる程度にとどめている（たとえば表1の課題3参照）．口頭発表の技術については，次の学年の「卒業研究」の中で大いにそれを磨くチャンスがあるので，あえて取り上げていない．また既成の分析，設計技法について深く学ばせることは避け，むしろ分析，設計といったことがなぜ重要かを身をもって体験させることに重点を置いた．既成技法についての具体的なことは，別にソフトウェア工学の科目（おそらくは大学院レベルの）を設けて，そこで取り上げることが望ましい．

5. 結　び

情報関係専門学科における，ある中級実験科目の設計とその実施経験について述べた．この科目の最大の特徴は，「徹夜もありうる」という態勢を作ることによって，学生の人間的成長をうながす，というところにある．

この科目設計に関して筆者らを鷹派だと評する同僚は多いが，それは事実とは少し違う．われわれは元気印派である．ぎゅうぎゅうと学生たちを絞め上げるのではなく，学生といっしょにわあわあ騒ぐ．人生を楽にうまく擦り抜けることを考えがちな多肢選択世代の学生諸君に捧げる解毒剤として，このぐらい大切なものはないのではないか，といささか自負している．

謝辞 無茶苦茶なことを元気にやる，という企てに嬉々として乗ってくれた（全員ではないが大多数の）学生諸君に感謝する．新しい CLU システムを後輩たちのために突貫工事で作り上げてくれた諸君（江原善君ほか）に感謝する．過去20年間にわたってこの科目を作り上げてゆく上で，辻　尚史，米沢明憲，久野　靖の各氏にお世話になった．感謝する．

ジェラルド・M・ワインバーグ氏からは，実に多くのインスピレーションを得た．深く感謝する．

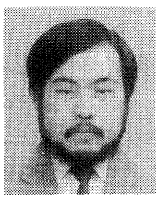
参考文献

1) Kimura, I.: On Teaching the Art of Compromising in the Development of External Specifications, *Journal of Information Processing*, Vol. 1, No. 1, pp. 33-41 (1978).
2) Curtis, B., Krasner, H. and Iscoe, N.: A Field Study of the Software Design Process for Large Systems, *Comm. ACM*, Vol. 13, No. 11, pp. 1268-1287 (1988).
3) Liskov, B. and Guttag, J.: *Abstraction and Specification in Program Development*, p. 469, MIT Press, Cambridge, Mass. (1986).
4) Weinberg, G. M.: *Becoming a Technical Leader*, Dorset House, New York (1986). 木村　泉訳：スーパーエンジニアへの道，p. 288, 共立出版，東京 (1991).
5) Weinberg, G. M.: *The Secrets of Consulting*, Dorset House, New York, 1985. 木村　泉訳：コンサルタントの秘密，p. 254, 共立出版，東京 (1990).
6) Stewart, I. and Joines, V.: *TA Today—A New Introduction to Transactional Analysis*, Lifespace Publishing, Nottingham, Chapel Hill (1987). 深沢道子（監訳）：TA TODAY — 最新・交流分析入門，p. 417，実務教育出版，東京 (1991).
7) 小谷善行，阿刀田央一，中森眞理雄，高橋延匡：情報工学系学科における実験・演習の一設計例，情報処理学会論文誌，Vol. 22, No. 5, pp. 402-410 (1981).

（平成5年1月20日受付）
（平成5年9月 8 日採録）

木村　泉（正会員）

1935年生．1960年東京大学理学部物理学科卒業．1965年同大学院博士課程退学．理学博士．東京大学助手，東京教育大学講師を経て，現在東京工業大学教授．計算機システムのヒューマンインタフェースに関する研究に従事，IEEE, ACM, 電子情報通信学会，ソフトウェア科学会，Human Factors Society ほか各会員．



大野　浩之（正会員）

1960年生．1984年東京工業大学工学部制御工学科卒業．1986年同大学院修士課程修了．1989年同大学院博士課程単位取得退学．博士（理学）同年共同電子株式会社技術部入社，同年東京工業大学理学部情報科学科助手．WIDE プロジェクトに参加し，大規模コンピュータネットワークの管理および運用に関する研究に従事．電子情報通信学会，日本ソフトウェア科学会，USENIX, Internet Society 各会員．