「ソフトウェア工学」の講義を担当して

随　筆

首藤　勝*

Giving a Lecture on Software Engineering

Key Words：Software Engineering, Information Science, Lecture.

はじめに

ソフトウェア工学に関して大阪大学基礎工学部で行った講義について紹介することが本文の目的であるが，初めにソフトウェア工学の由来に触れておく．

コンピュータが社会の隅々にまで普及して，日常生活のあらゆる場面で陰に陽に我々を支えているという時代が実現したが，そのコンピュータを使用者の意図通りに働かせるソフトウェアは，（機械加工，組み立てなどでなく）人間の思考の直接の産物として作られる．このことがソフトウェアを工業製品として効率よく作り出す上で妨げとなってきた．ハードウェア分野では生産性が目覚ましく向上したがソフトウェア分野での生産性向上は遅々としてしか進んでいないなどと指摘され続けている．

コンピュータ（いわゆる組み込みマイコンを含めて）が広く普及する程ソフトウェアの開発需要が高まって，その需要の伸びに比べて肝心の開発要員の供給が甚だしく不足してしまう．この差は拡大の一途であり，さらに1件ごとのソフトウェアの複雑さの増大が加わって，やがてソフトウェアの開発供給が不可能になる．これが「ソフトウェア危機」と指摘される事象である．通産省などでこのソフトウェア技術者の不足を何度か予測し，160万人とか90万人とかの数字をはじき出して，技術者育成の強化を訴えていることは知る人も多いであろう．

この問題に対処するための方策として，ソフトウェアを合理的に設計制作する手法がこれまでに種々のアプローチで研究され，大規模なソフトウェアを出来るだけ工業的な方法で作ろう，また開発組織で取り組む課題を明確化し解決法を見出そう，という努力が重ねられてきた．現在ではかなりの局面で成果が見られている．

大学の情報科学分野でこのソフトウェア工学の教育をどの様に展開するかについても，各方面で検討が続けられてきた．その結果が情報処理学会の検討した「コンピュータサイエンス教育カリキュラム」にも採り入れられている．

講義科目としてのソフトウェア工学

大学の情報科学関連学科ではソフトウェア技術の教育を基礎理論，実技の両面から実施している．実技に関しては，プログラミング技術の訓練を，幾つかの水準の言語（人間が用いる数式表現に近いのが高水準，機械構造に近い暗号的表現によるのが低水準）を用いて，低学年次から多くの訓練を受けさせているのが普通である．3年生にもなると，学生は問題が与えられればプログラムを書けるという意味ではほぼ一

*Masaru SUDO
1934年5月7日生
1957年大阪大学工学部通信工学科卒業
現在，大阪工業大学情報科学部，教授，工学博士，情報システム工学
1998年3月大阪大学基礎工学部停年退官
TEL 0720−66−5384
FAX 0720−66−8380
E-Mail sudo@ij.oit.ac.jp

人前となっている.

しかしながら，学生が一人で或る期間に課題解答として書くプログラムの規模はたかが知れている．産業界で実用に供される大規模ソフトウェアと比べてあまりにも差が大きく，それだけの訓練ではソフトウェア作りで重要な「どの様な構成に作るか」,「どの様に記述文書を整えるか」,「どの様に作業段取りを決めるか」,さらに,「そもそもそのソフトウェアに持たせる機能・性能をどの様に決めて記述するか」,という課題にはとても手が届かない．この様なソフトウェアを生産する技術が如何に大切か，どういう技術手法が開発され実用化されて来たか，について見識を持たせたいという観点からは，プログラミング技術の教育だけで済まさずに何とかこのソフトウェア生産技術あるいはソフトウェア工学についても学ばせたい.

前置きが長くなったが，大阪大学基礎工学部の情報工学科でこの分野を対象とする授業科目「ソフトウェア構成論」が設定され，企業出身で多少の経験を持つ筆者が平成2年度から担当することになった.

しかし，このソフトウェア工学を講義科目として捉えると，プログラム技術などの具体的教科と比べて，どの様な態度でソフトウェアを作り上げるかという方法論の類であるので，折角の講義が単なるお話に終わってしまうのでないか，という心配が予見され，これをどの様にして実効ある授業科目にするかという問題に突き当った.

それを考え悩むうち，筆者自身の会社入社の頃を思い出し，一つのヒントを得た．昭和30年代初めの頃，電機業界では品質管理あるいは生産技術を工場のラインに適用する動きが顕著であった．当時は新入社員にもその専攻の者が居り，本社生産技術部などのいわゆるスタッフ部門に所属して，学校で生産の実戦を経験しているわけでもないのにスタッフチームの一員として各工場に出掛けてはその生産体制，品質管理体制の改善に取り組んでいたのである．結構自信に満ちた働きぶりが印象的であった．このことをソフトウェア生産技術に適用し，方法論であっても一通り修得させておけば，少なくともソフトウェア開発でこれらが重要であるという意識を持ち，現場で課題に遭遇して実戦に適用することは出来るであろうし，技術者としての成長の加速に役立つであろうと，楽観的に考えることにした.

講義の内容と方法

(1)情報工学科の入門カリキュラム(受講者の先修条件)

この授業科目「ソフトウェア構成論」は情報工学科3年後期に割り付けられている．幸い基礎工学部は全学共通教育機構と同じ豊中キャンパスにあって，共通教育課程の学生に対し専門基礎の科目を修得させることが比較的容易である．実際，情報工学科の学生は1年次からプログラミングの訓練を徹底して受けている．従って3年次ともなればプログラミングとはどんなものかを体験的に知っている．またプログラム設計という科目で良いプログラムとはどんなものか，どういう技術でそれを作るかについても教え込まれている.

この様な土台の上に立つならば，ソフトウェア工学の授業の展開が楽になる.

(2)ソフトウェア構成論で採り上げる範囲

ソフトウェア工学の範囲は随分広いのであるが，3年次という対象を考えると，それ以前にプログラム造りの技術面の修得は済ませており，また専門的にソフトウェア工学を突っ込んで勉強するのはそれを自己の専攻に選ぶ者は4年次の卒業研究から，それ以外の者は大学院に進んだ段階でよいだろうと判断される．このことから，この科目ではソフトウェア生産技術全般についての基本的な知識習得に加えて，ソフトウェア開発工程の初期の段階，通例的に上流工程と呼ばれる部分(要求仕様定義，ソフトウェア設計など)の重要性を認識してそこで用いられる諸方法を修得させることに絞ってよいと判断した.

(3)教材の選定

新しい分野であり，範囲が広いだけに，適切な教科書が見付かり難い．一人の著者の執筆では殆どの場合，著者の専攻からの距離に依って内容に粗密の差が出やすく，教科書として用い

るには不満が大きい．調査の結果，電子情報通信学会刊行の「ソフトウェア生産技術」を選定した．この本は著作時点は少々古いがNTTとNECのソフトウェア生産技術拠点の長が共同で執筆しており，配下の技術陣が開発活動を通じて積み上げた豊富なデータが随所に引用されているなど，他書に見られない良さがある．また内容構成として，ソフトウェア生産技術の各段階ごとの方法解説を行った後に，一章を設けて一つの実例についてそれらの段階を通して具体展開して見せるという形を採っており，これが読者の理解に役立つと判断される．

教科書では言い尽くせない部分，さらに実社会の動向やマスコミ報道に出る表現の評価など，教官として学生に是非伝えたい事柄も結構あるので，これらについては資料を随時配布することにした．例えば大規模ソフトウェア開発の問題については，現在我々を取り巻く社会の諸面でどの程度の規模のソフトウェアがどのような形で存在するのかを具体的に示す資料，また，新聞報道に現れるソフトウェア生産性3倍の向上などの表現の真偽，理解の仕方を説く資料，などで始める．さらに，ソフトウェアを含むシステム全体がどのような工程(段取り)で作り上げられるのか，ソフトウェア工程はハードウェアに習って論じられることが多いがそれはソフトウェア開発の真の姿なのか，また，ソフトウェア生産性向上が歴史的にどのような発達を見せたのか，その各々の技術実用化の意義をどう評価するか，などなど，話し出すと際限ないが，こういう実社会の問題を出来るだけ要領よく伝えることに努力を注いだ．これらの資料は筆者が体験に基づいて作成した解説資料の他に新聞雑誌記事の切り抜きを随時加えた．

(4)講義と課題の組み合わせ

講義の冒頭で，現実社会で我々がどの程度にソフトウェアの利用をしているのかを，種々の例で示し，ソフトウェアのボリュームがもたらす問題についての理解させることに務めた．

そこから先は，概ね教科書に従って，ソフトウェア開発の各段階，すなわち要求定義，ソフトウェア設計，プログラミング，デバッギング，システムテスト，運用と保守，と，いわゆるウォーターフォールモデルに従って，各段階で立ちはだかる問題とその解決手法について解説を行う．特に上流工程である要求定義とシステム設計について重点的に説明をする．ソフトウェア工学分野では他分野で考案実用されている技術手法を貪欲に採り入れてきたこともこの段階で理解させる．

教科書の構成意図を出来るだけ活用し，各段階の方法解説のあとに後章の具体例展開の対応部分を読んで理解するやり方で，単なる方法の解説でなく具体的なイメージを保ちながら学習出来るよう配慮した．

講義の進展に伴って，レポート課題を出すのであるが，3年ほどの内に次の形に落ち着いた．

［課題1］システムの要求仕様を書いてみる．
授業の冒頭で出題し，その時点で学生が持っている知識で解答させる．題材としては駅の自動改札機構，商品の自動販売機等を採り上げる．

［課題2］要求仕様分析段階で問題の把握のためのツールとして使われる特性要因図を具体的なテーマについて作成させる．頭で知るだけでなく手を動かして理解することを狙う．

［課題3］簡単なシステム(課題1で採り上げたものと同じ)について，要求仕様分析およびシステム設計を課題とし，5人程度のグループ作業として実施させる．グループ作業を体験させることと併せて，課題1との対比で合理的開発手法の意義を理解させることを狙う．

(5)演習問題を配布，自習させた

前述のレポート課題に加えて，全部で30題程の演習問題を与え，講義の内容をより具体的に理解させるよう務めた．

演習問題としては，講義内容と直結した知識の整理と応用の考察に関する試問を主としたが，その他に，筆者自身が執筆した雑誌の巻頭言などからソフトウェア工学についての主張を引用し，それを対象にディベートを行わせる問題など，自分の意見を整理して述べる訓練を意識したものも用意した．

(6)ソフトウェア原価など経済面についても

講義に加えた．

工業として物を生産する上で原価についての基礎的な話を是非聞かせておきたい．ソフトウェアの場合に原価はどのように把握されるのか，原価と価格の関係はどうなっているのかという事業行動のメカニズムについても，学生の生活とそんなにかけ離れた他の世界の問題ではないことを理解させたい．例えば日常的に使用するウィンドウズ95が何故あのような価格設定になるのか，ソフト会社でアルバイトをして時給基準で得られる額と製品ソフトに付けられる値段との間にはどのような関係があるかなど，技術者であっても知っていて欲しいので，この程度のことには触れることにした．

ここまで来ると，価格決定の仕組み，さらに減価償却など，経済の分野の用語，概念にも多少は踏み込むことになった．

また，ソフトウェア技術分野で活動する上で重要なポイントである技術標準の意義，さらに工業所有権および知的所有権に関する諸問題について，入門的知識を与えることも付け加えた．

評価および受講生の反響

以上のような講義を8年間にわたり，多少内容を変えながら続けてきたが，現時点で振り返ってみた感想，並びに学生のレポートからの反響の抜粋を次に列挙しよう．

(1)実践課題が有効であった．特に課題3のグループ作業は，受講者にとって体験効果が大であったようだ．苦労してまとめたレポートでそれが伺えるし，またそのレポートに書かせた感想で殆どの受講生がその体験効果に触れていることからも手応えを感じた．

(2)他学科受講生との比較で情報工学科の低学年からの訓練の有効性が実証された．

この科目は情報工学科以外の学生にも受講を許しており，当初は大勢の他学科受講生が来て，総人数が100人を超える年もあった．科目の名称からソフトウェア関連の比較的楽な科目という観測があったのかも知れない．しかしながら，かなりの量のプログラム作りの訓練を経験して初めて理解できる内容が結構あるため，この科目だけを目指してきた者には少々無理があったようだ．試験の答案でどれだけポイントを押さえて解答しているかを評価すると，土台の訓練の無い者は点数が取れない．結果として他学科からの受講生は軒並みに落第する状況となった．中にはある学科から25人受講に来て合格は1名のみ，それも同情採点の結果，という年があった．この様なことがあって後，先修条件の吟味を厳格にすることにし，他学科受講生については履修希望者を面接して実質的にプログラミングあるいはプログラム設計についての基礎事項を修得しているかどうかをチェックし，授業の効果を保つ策をとった．他学科からの受講生がすべて出来が悪いわけではなく，このチェックをパスした上で授業そのものにもよく追随し立派な成績を得た者も居ることを付言しておく．

(3)プログラミング実践体験が受講効果を高めた例もある．

情報工学科の学生にはアルバイトとしてソフト会社でプログラミングの実戦をやっている者が少なからず居る．学校での教科で与えられる宿題と比べて大規模なソフトウェアを限られた時間で仕上げることは相当な苦行であると思われる．この様な経験を積んだ学生がレポートの中で，このソフトウェア工学分野の技術の意義を改めて感じた等の感想を書いているが，まさにその通りであったろう．実社会に出て体験する苦労とそれによって身に付ける見識を先取りしたことにもなっていると，教官としてもある種複雑な感想を持った．

学生がアルバイトに打ち込むことの是非については議論のあるところであるが，アルバイトをしないとこのソフトウェア工学の科目の意義が解らないというのは本来的でない．ソフトウェアに関する実験科目で効果的にこれに近い実戦的体験をさせることなど，今後のカリキュラム充実の上での検討項目であろう．

(4)大規模ソフト開発に伴う諸問題に迫れたかという点では，この半年間の講義では，まだその入口に留まった感は拭えず，不満が残る．けれども，冒頭に述べた生産技術専攻者と同様に，一通りの知識を修得し問題に取り組む見識を持って企業活動に入って行く，という境遇での強みは持たせ得たと考える．

おわりに

以上がここ数年間に亘って実施してきたソフトウェア工学の講義の概要である．プログラミング技術の教育訓練に加えてソフトウェア生産技術の概要を修得させる努力を幾分か汲み取って頂けるだろうか．

ソフトウェア工学の主題の一つが，計量し難い対象に如何に迫り定量的に評価して改善の方策を立てるかであるが，この科目の履修を通じてその基礎的事項は習得させ得るであろう．更にその先については大学院で「ソフトウェア計画構成論」という科目で，コストモデル等を主題とした講義により学習させてきた．

余談だが，計測し難いものに迫る努力を説いたガリレオの言葉を「ホルモンの鶏冠単位」の例(薬剤投与により立ち上がる鶏冠の投影面積増大を計測する)とともにご教示下さった阪大工学部通信工学科の青柳健次先生の講義を思い出しながら本科目の講義を行った．無線通信工学だったかと思うが，定かでない．当時先生は「卒業生達が講義内容は忘れたがホルモンの話だけは覚えていますと言う」等と苦笑しておられた．筆者もその輩であるらしい．