報　告

パネル討論会

ソフトウェア工学 80 年代の課題

昭和 56 年前期第 22 回全国大会† 報告

パネリスト

國井　利泰[1]，清水　康男[2]，高村　眞司[3]

三巻　達夫[4]，吉村鐵太郎[5]，司会　大野　豊[6]

80 年代での期待と問題

大野　豊

本日のパネル討論は，現在最も研究開発の要請が強くまた問題点も多いソフトウェア工学について，その 80 年代の課題を討論するわけであるが，パネリストとしては，大学，公共的研究開発機関，メインフレームメーカ，ソフトウェアハウス，利用者のそれぞれの立場の方々にお願いした．ソフトウェア工学についてはその立場により，考え方も異なるが一様に大きな期待をよせているといってよいであろう．そこで，各パネリストのご意見をここにまとめておくべきかもしれないが，それは各パネリストの報告におまかせしてした方が良さそうなので，ここでは司会をするに当って司会者がどのような個人的見解をもっていたかを述べておくことにしよう．

ソフトウェア工学という言葉が使われだしてから，すでに 12 年経過しており，わが国でもようやくその研究・開発が盛んになってきた．ソフトウェア工学が目的とするソフトウェアの生産性向上，品質向上について，今まで多くのことが提案されてきたが，実用化されたものは必ずしも多くはない．研究として提案された新しい技法，技術の実際のソフトウェア開発への適用にはかなりのおくれが見られる．そこで，これからの課題は新しいアイデアの提案もさることながら，従来の研究成果を評価して，実用に結びつけることに努力を集中すべきである，という主張もなされている．

ソフトウェア工学では，今までに，ソフトウェアのライフサイクルという概念が認識され，その各段階，すなわち，要求分析，仕様，設計，プログラミング，テスト，保守において，表現の形式化や検証手法をもとによく定義された手順，技法が提案され，ツール化も行われた．ソフトウェアの問題は，最終的には，仕様からプログラムを自動生成するのでなければ解決されない，という主張もあり，その方向の試みも行われている．しかしながら，これら従来の研究・開発の実情から見て，ソフトウェアの問題はそう簡単ではないことが痛感され，おそらく，実際のソフトウェア開発の立場からすれば，革新的な特効薬を期待するより，良いと思われる提案を実用し，使いこなし，それらを地道に積み上げていく以外に方法はないであろう．

最近，ソフトウェア製品ということが強調されている．これはソフトウェアというものを工業製品と考えようということで，それは，近代的な工場で充分な品質管理の下に生産され，使用説明書や性能保証書付きでどこにでも売られる製品，という意味が込められているといえよう．これには従来の品質のよいソフトウェアを作りそして使う，という立場よりもっとはるかにきびしい条件がついており，ソフトウェア工学上より多くの問題を提起しているわけである．

そこで思いおこすのは，わが国の工業における生産技術である．今や各種工業生産品の生産技術は，長年の技術的努力の結果として世界の先頭に立った．ソフトウェアを工業製品と見なせるならば，同じような努力が結実しないであろうか，と考えるのは当然である．そのように見てくると 80 年代はわが国のソフトウェア界にとって大いに希望の持てることになるのであるが，果してどうであろうか．ここでそのためのいくつかの条件について考えてみよう．

よくいわれることであるが，わが国の生産技術が進歩したのは，日本人に革新的な創造性があったからではない．輸入された基本的アイデアに対して，小さな

† 日時　昭和 56 年 3 月 26 日，12：30～13：30
場所　学習院大学

1) 東京大学，2) 富士写真フイルム，3) 電電公社横須賀通研，
4) (株) 日立製作所，5) 管理工学研究所，6) 京都大学

創造的発想，改良的工夫を積み重ねることによって，ここまできたというわけである．そこには日本人の意識の均質性が大いにあずかっており，プロジェクトを構成するグループの各人のみならず，現場の作業者に至るまで一様に改良的発想を行い，それを積み重ねた結果であるといわれている．しかし，このようなことが成功した分野は多くの場合，すでに大きな革新的発想の時代は終り，技術の進歩が改良的になっていたものばかりである．そのような技術的社会的環境におかれたとき，日本人の改良的創造性が有効にはたらくことになる．

ソフトウェアの技術の現状がどうであるかを考えてみると，たしかになお現在大きな革新的発想が強く要求されており，その可能性もあるであろうが，一方国際会議などでの発表をみると種切れとまではいえないが，やや飽和状態にきているのではないか，と思わせるものがある．もしそうだとするならば，このような状況は日本的創造性を発揮するチャンスであり，ソフトウェア製品の生産技術を大いに推進することができるのではないか，という望みをいだかせる．今まで，ソフトウェア技術は米国に比べて 10 年以上もおくれていて，わが国の研究者，技術者のソフトウェア技術への貢献は非常に少ないといわれていた．この 80 年代にはそのような汚名をばん回することができるかもしれない．

しかしながら，そのためにはいくつかの検討すべきことがある．まず，自動車やカメラなどとソフトウェアというものとが，本質的に特に工業的生産という面から見てどう異なるのかを明確にしておかなければならない．ソフトウェアと同じように人間の知能的活動の集約であるLSIやマイクロプロセッサなどの生産技術が著しく成功を収めたことを考えると，その違いは何かが理解され，解決の道も見出せるかもしれない．

また，以上のようなソフトウェアの技術を推進するには，その研究開発体制の整備は急務であり，通産省をはじめ諸方面で諸施策が行われるようになってきているが，まだ必ずしも充分とはいえない．メインフレームのメーカでは各所とも着々研究開発体制を整えているようであるが，ソフトウェアハウスは組織的構造的に問題点が多いと聞いている．わが国の大学は今までソフトウェア工学の研究への寄与は多いとはいえなかった．研究者の数も少なく，また大学の研究組織がソフトウェアの研究に不整合な面もないではない，といわれている．

別の重要な問題として，ソフトウェアにかかわる人材の育成と確保がある．常々各方面でいわれているように，ソフトウェア開発の要求は年々急速に増大するのに，対応できる人材が揃わないので要求を満たすことができない状態が続いている．ソフトウェアの生産技術が要求に応えられるように発展していない段階で，さらに状況を複雑化したのは，マイクロコンピュータの工業化による急激なそのソフトウェア需要の増大があげられる．これらは，ソフトウェアの生産技術の開発にまで手を伸ばす余ゆうをさらにきびしいものにしている．

このほかにも多くの問題があるが，これらを解決しつつ，また解決するためにソフトウェア工学のより大きな進歩が望まれ，特にわが国の技術が進展する可能性のある重要な時期に現在きているのではないかと考えるわけである．幸いにして，「第6回ソフトウェア工学国際会議」を日本で行うという話がまとまり，昭和58年9月に東京で開催することができるようになった．以上のようなわが国の事情を考察すると，わが国での研究開発をさらに推進する丁度よい機会ではないかと考えられる．本国際会議は，今までその質の高さを誇っていた．ということであり，これからもそれを維持していきたいと強く要請されているので，関係者各位のすぐれた論文の投稿と多数方々の参加を期待している．

日本ソフトウェア立国の鍵“ソフトウェア製品”
—'80 年世界協調経済時代の工業的課題—

國井　利泰

日本は高度に教育され，勤勉かつ繊細な気質を持つ人的資源に恵まれている．地域的にもまとまった，協同作業に適した環境にある．工業製品の中でも，最も加工度が高く付加価値の多いもの，これはほぼ 100% 知識の塊であるソフトウェア製品である．世界の高度工業国の中でも天然資源の最も貧しいグループに属する日本にとって，このソフトウェア製品生産こそが最高に適した世界における工業分担である．

生産技術レベルでいえば，われわれ人類はこの新しい製品にたいして，1. 生産法，特に生産機械，2. 人的組織・管理法，3. 製品のライフ・スパン全体にわたるライフサイクル管理，4. 経済的側面，特にコスト管理の面などの重点項目に関してその一端を知りつつある段階である．この時期にあたって最重要事項は，日本の今後の世界工業分担を意識した独自の技術開発とそ

の蓄積，それに必要な体制の整備である．

日本の工業界・学会全体をカバーする公共的組織である情報処理学会がこのソフトウェア製品（ソフトウェア・プロダクトとも呼ばれる）の技術・組織両面に亘って早い時期に的確な活動体制を敷くことは大いに望ましい．そのための具体的ステップについて，活発な討議を開始したいと考える．

参考文献　國井利泰編「ソフトウェア・プロダクト工学」共立出版

ソフトウェア開発の生産性向上

清水　康男

ソフトウェア工学 80 年代の課題というテーマで討論をおひきうけしたわけであるが，パネリストの皆様はそれぞれこの分野での専門家として日夜研究開発に携わっておられまた造詣の深い方達ばかりですので，ソフトウェア工学そのものについての討論は皆様にお願いすることとし，私はその結果を利用するユーザの立場として常日頃考えていること，実施していることについて述べてみたい．

コンピュータ関係では最近ソフトウェア生産性向上が重要課題として採りあげられることが多くなってきた．テクノロジの急速な進歩により，ハードウェア費は全体の EDP 費用内で次第に下ってきており，一方ソフトウェア費は急激に増大してきており，当社においても，すでに両者相拮抗する状態にまでなってきた．

またコンピュータメーカの調査によれば，大半のユーザが平均 20. ほぼ2年分のプロジェクトを抱えており，その 2/3 が未着手の状態になっていると報告しており，この傾向は当社においてもまた然りである．日本 IBM の調査によれば適用業務プログラムの伸びは毎年 24% と見込まれている一方1人のプログラマの生産性は年率3%程度と報告されており，現状のままでは開発業務と開発能力との gap は益々増大すると予想される．

このような gap をいかに縮めるか，そのためにいかなる施策を講じるかが私共の当面する課題であるが，一方その根底にはいかにしてソフトウェア開発の労力を最小にするか，むしろ何も書かなくても必要な情報が生産できれば最高に理想の状態といえる．

前記の gap fil という観点に立ってのソフトウェアの生産性向上とは，開発量の増加を目指した一連の Activity の総合化努力にあるわけで，解決手段として開発保守そのものの生産性向上，開発保守の外注化，ソフトウェアパッケージの購入，社内各部門のコンピュータ化業務への動員等が考えられる．

最初のソフトウェア開発の生産性向上については専門メーカの技術開発に依存するところ大であるが，いろいろ技法を駆使して，たとえば富士通では3年で2倍という数字が報告されているし，また日本電気の SDMS (Software Development and Maintenane System) によれば開発スピードは3倍にも4倍にもなると報告されている．

極部的にはオンラインデータベースシステムを対象にした日立の CORAL (Custmer Oriented Application Program Development System)によればやはり同様な生産性向上が期待されるという．

これら諸々の生産性向上のツールについては提供され効果が期待できるならば大いに導入活用を図りたいと考える．

第2の外注要員の活用については，各社共すでにかなりの比率で依存しているものの国全体としても要員不足が深刻となっている現状，各注要員依存には自ずから限界があり，また生産性向上の本質的な解決手段にはなり得ない．

第3のソフトウェアパッケージの購入は，ソフトを作らなくて済む，書かなくても良いという根本理念からみても，さらに積極的に徹底して推進をはかるべき施策と考える．当社でもすでに MARK IV, ADBAS 等各種のソフトを購入活用しているが，今後とも効果期待のソフトが出現すれば，いくらでも購入し利用することに吝でない．その前提としていかにユーザニーズにマッチしたシステムソフト，アプリケーションソフトの流通の活発化が望まれる．1つのソフトを 10 社で使えば生産性は 10 倍ともいえるわけでソフトウェアの流通は我が国にとっても大きな課題として，通産省そのほかの機関により振興の動きが活発化してきた．54 年6月にソフトウェア流通促進センターが設立されたがこれらの機関の調査によればソフトウェアの流通を阻害している要因として，(1)ソフトは自ら作るものという考えがある，(2)製品の実態が不明である，(3)要求にあったソフトが見当らない，(4)品質が不明，(5)汎用性がない，(6)使いずらい，(7)需要が確実に把握されていない，(8)商品の認識が低い，(9)プロダクションシステムが確立されていない，(10)経営リスク等の問題でソフトウェアハウスが育たない，(11)適用コンピュータの標準化が遅れてい

る，(12)ソフトウェアの法的位置づけが低い，(13)要員の教育体制が整っていない，等の問題が指摘されている．ソフトウェアの流通に関しては，全般的に未成熟，未発達の分野である感は否めず，今後の関係部門の方々の努力が望まれるわけである．

最後に生産性向上の手段として社内エンドユーザ自身の動員が挙げられる．コンピュータ部門がユーザ部門の要請を受けて業務を開発推進する体制はすでにみたごとく限界飽和の状態にあり，これを解決する手段としてのエンドユーザの動員は自社内で自力で解決できる手段として最も有力でありかつ今後の方向でもあるのではないか，社内におけるコンピュータ活用の底辺を拡げる意味でも積極的に推進を図らねばならないと考える．これが従来のコンピュータ部門の大きな役割の一つともいえるわけで，エンドユーザ自身が直接コンピュータを利用できる，またそれを容易ならしめる仕組の構築，ハード，ソフトの提供と教育が必要．目下当社においても分散型コンピュータの社内各部門への配置，それに使用するソフトの提供と教育等を鋭意推進中であるが，ソフトウェアについても TSO による IBM の APL，そのほか第4世代の言語といわれる FOCUS とか SAS のような超簡易言語が徐々に出現しつつありユーザ動員の推進に拍車がかかってきつつある．もちろん今後共益々の超超簡易言語の出現が期待されるわけである．

時代は徐々にハードからソフトへ転換しつつある．ソフトを中心としてハードとか仕事の仕組そのものをそれに合せて変えるとか，この方が一層経済的といえる傾向が徐々に出現し増加していくのではなかろうか．

ソフトウェア資産の標準化（部品化）

高村　眞司

ソフトウェア工学の課題に対する多くの人達の意見，取組み方等は力点の違いはあっても大略方向は一致しているのではないかと思われる．

ソフトウェア技術は，これまでも多くの分野でそれぞれ研究・開発が進められ，著しい成果をあげてきているが，今後のコンピュータの利用動向を考えてみると，利用形態，応用分野の拡大速度がソフトウェア技術の開発速度を上回っている．ソフトウェアが対応しなければならない条件として，一つはハードウェア技術の進歩であり，もう一つは利用形態の多様化である．従来は利用形態が集中化指向にあり，より速く，より大きくという動向に対し，ハードおよびソフトが対応してきた．大規模化の動向は今後も続くが，最近では LSI 技術の進歩に伴いあらゆる分野にマイコンが導入されておりコンピュータの利用者層が専門家から大衆までに拡がり，コンピュータについての知識の幅は，さらに拡大する傾向にある．このような動向にあって，コンピュータの利用技術としてのソフトウェアについても，専門家用のより高度のテクノロジのほか，大衆用の利用技術の充足が必要である．

ここ 20 数年の間にコンピュータ関連の技術はいずれも3桁以上の改良が図られ，価格性能比も5年で4倍の速さで改善されており，今やハードよりソフトが高い時代になっていることは周知のとおりである．これに対し，ソフトウェア開発能力は，高々1桁程度の改善であって普及に対してソフトウェアの供給が追いつかない状態を招来している．ソフトウェア工学の急速な進展が望まれるゆえんである．

現在ソフトウェアの抱えている問題は，それぞれの分野，立場によって種々異なっている．すなわち開発者の立場でみれば，大規模化，資産増，多様化，大衆化等の動向があり，開発コストの増大，維持費の増大，さらに要員の確保（人材育成を含む）等の問題が，また，サービス提供者の立場からは，新サービス等の提供速度の遅延（ソフトウェア開発の長期化，既存資産との互換確保のための工期延伸 etc）等が，さらに利用者の立場からは，すべてがオーダメイドであること，流通するソフト製品が不足している，信頼性の保証がない等の問題が指摘されている．

前記の問題が生ずる要因を探ってみると，

① 機能の複合化……機能の膨張は無限に続く．

② システムの巨大化……量の多いことが質的変化をもたらす．

③ 資産の継承……既存資産に執着する．

④ 人的要素……能力のバラツキ，画一化の拒否反応，人為事故の増大

などが挙げられるが当面のニーズへの対応に追われて，これらへの対応が後手に回っているのが問題である．

このような状態から脱出するためにどうすればよいか，という命題については，特効薬は存在しないので，多くの方法により地道な努力の積み上げによるしかないが，努力の方向について多くの人達のベクトルが合っていなければならない．

ソフトウェアの生産性が今後の発展の鍵を握ること

は多くの人が言及しているが，実効を挙げるには，ソフトウェア製品についての分業が進まなければならない，すなわち，基本的には，ソフトウェアにも量産化効果を導入できる素地を作ることが，ベースになる．現状では，とかく，自分で作った方が早く，良いものができるという，プログラマの自負，生き甲斐に起因する要素があるが，独自性を発揮する分野と既存製品（部品）を活用すべき部分とを明確に区分し，商品についてはできるだけ枯れた部品を用いることを強制することが必要である．既存部分を使用するにはその機能，インタフェース条件等が明示され，かつ誰がみても読み誤りのないような表現がされている必要があり構造面の改良とともにドキュメント技法の洗練化も必要である．なお部品はカセット等に記録済みのプログラム製品として販売されることが望ましい．

ソフトウェアの開発にライフ・サイクルがあることはほぼ周知の事柄となったが，各工程を独立させ，異なる人がそれぞれの工程を分担することについては必ずしも，達成されていない．工程を分割しながら，同じ人が複数の工程を担当している．たとえば方式開発者が部品のレベルからシステム作りを行っているために工程間の引継ぎが曖昧にされ，結果として工業化が進まない原因となっている．工程を明確に切り，かつそれぞれの工程の改善を図る専門家を配置することがソフトウェア製品の製造工場化のために必須な条件である，この体制なしに設計技術，管理技術，あるいはテスト支援技術等を研究，試作してもそれらの成果が積み上らないで徒労に帰すことになる．

ソフトウェア工学の課題はこのような生産ラインを確立し，既存部品をこのラインに供給して行くような体制作りから改善して行かなければ効果は期待できない．現在ソフトウェア開発の生産性改善策としては，

1)　プログラミング言語の高級化
2)　プログラミング技法の高度化・標準化
3)　開発管理体制の強化
4)　問題解決手法の開発

などが挙げられている．このうち1),2)については，これまでの経験がある程度蓄積され，工程分割と分業化へ向けての基盤が形成されつつあるし，今後もソフトウェア従事者の増加とともに，さらに改善が図られることになろう．しかし，問題は 3)，4) であって，システム規模，ソフトウェア資産の巨大化，問題の複雑化傾向に対しては必ずしも見通しは良くない．これらに対しては個々の問題の単なる解決策ではなく，方式的・組織的な対応が必要であり，ハードウェア工場におけると同様の生産ラインと管理組織の確立が必要である．

方式的改善の一例としては，汎用に作られたソフトは，それを利用する1ユーザからみればきわめて冗長なものを使わされていることになる．ソフトの巨大化からの脱出のためにも，必要最小限の機能だけが選択組み合せ利用できる仕組みの確立が必要であり，これがプログラム製品化の促進にもつながる．また従来はハードウェア資源の有効利用を主目的にその共用化や多重制御技術の追求が進められてきたが，ソフト制御機構の単純化のためにはハード資源は犠牲にすることも必要となる．ソフトウェア制御機構の簡素化のためには，プログラム構造の階層化，抽象化は必須であるが，それらのインタフェースの標準化が重要な鍵を握ることになる．現在，接続条件の比較的粗な分野についてはインタフェースの標準化が進められているが，これをソフト部品について異機種間での互換がとれるまでに高めることが必要である．それを実現するのは当面高級言語レベルに期待されているが，それに加えて過去の履歴を蓄積し，学習による判断，管理などが自動的に行えるような管理システムの確立が望まれる．さらにこれらの推進のためには政策，行政的な強制力の働くことがある程度必要であり，今後，情報の保護，救済がさらに厳しい条件となることを考えると，ソフトウェアの複雑さ，巨大化は進む一方であり，その前に前述の種々の対策がとられねばならない．

ソフトウェアに期待される事項は，人工知能，知識ベースを初め，きわめて多くの事象があるが，目先のことに追われている比重が高すぎる．これらの工程を労働集約型から脱皮させることにより，ソフトウェアの研究開発が知識集約型へと転換が可能となりソフトウェア産業の発展につながることになろう．そのためにはソフトの生産ライン確立のための投資，補助を抑制しないことである．

ソフトウェア工学関連の技術

三巻　達夫

計算機の利用は大型計算機，ミニコン，制御用計算機，マイクロコンとあらゆる分野で拡大しつつあり，その局面もオンライン・ユースの高度なソフトウェアの開発が要求されている．マイクロコン応用ソフトウェアにしても，8 bit では数 Kstep のものが，16 bit になると高機能なるがゆえに，すぐ数 10 K step と1桁上の高機能なシステムの開発となる．当社における大

型計算機からマイクロコンに至る分野でのソフトウェア開発量はもちろん，増大しつつあるが，マイクロコンソフトウェアの増大量が最も大きい値を示している．80年代におけるソフトウェアの第1の問題は，「ソフトウェア生産性向上」である．一方大型機による銀行システム，座席予約システム等は，日常生活に深くかかわり合いのあるものとなり，いわゆる社会の公器ともいうべき性格であり，これらのシステムの長時間ダウンの影響は大きい．ハードウェアの信頼性の向上も著しく，ソフトウェアのバグ（応用プログラムやOS）によるシステムの停止を避けることが重要である．第2の問題はソフトウェアの質，すなわち「ソフトウェアの信頼性の向上」である．ソフトウェアの特質としては，実現方式の自由度が大きいこと，実態把握が困難なこと，人的依存の大きいことが挙げられる．同じ機能のプログラムを作成するにしても，巧妙に作った場合と，無駄に作った場合では，一見後者の方が大きな仕事をしたようにも見られる．また機能にしても，ハードウェアと異なり見えにくいので，過度の機能に設計されるなど実現方式の自由度が大きい．またソフトウェアは物の形として見え難いので，進捗管理なども困難であり，また当然のことながらソフトウェア作成の能力の個人差が大きい．これらのことから，第3の問題としては「ソフトウェアの管理性」が重要であり，またシステムの拡張，担当者の変更に対処するために，第4として「ソフトウェアの保守性」が重要である．

これらの問題に対処するため，いわゆるソフトウェア工学と称される領域で技術開発が進められている．ソフトウェアの開発フェーズを

- システム分析・計画
- システム設計
- ソフトウェア設計
- プログラミング
- テスト
- 運用・保守

と分けた場合，プログラミングとソフトウェア設計のフェーズでは，生産性向上のため高級言語の使用や，構造設計法の種々なツールが開発され，これらは実用レベルに入っている．現在の技術開発の主題は下記の分野と考えられる．

- システム分析・計画…システムの構造化手法
- システム設計……要求仕様定義・解析手法
- テスト……テスト項目作成支援

システム分析・計画フェーズでは，最近システムの規模が大きくなり，ユーザの中でも，システムを構築する部門とシステムを運用（実使用）する部門が異なり，両者の間のミスマッチングが生ずる恐れもある．これはもちろんメーカがユーザのシステムを構築する場合にも生じうる問題である．これは要求仕様が明確に規定できないことであり，また同じ言葉でも，システムの構築者と運用者との間で定義が異なることもあり，両者のコンセンサスをとることが必要である．この問題には漠然としたシステムの目的を明確化するシステム構造化の手法を利用することになる．具体的には，ある問題に対するキーワードを関係者より抽出し，これらのキーワード相互の相関関係の情報を計算機に入れ，計算機支援により，その問題の目的樹木を作成する．各種の方式が開発されているが，当社ではPPDS (Planning Procedure to Develop Systems) があり，システムの分析・計画に有用である．

システム設計のフェーズでは，ソフトウェアの機能仕様を明確化することが必要である．特にソフトウェアの機能はvisibleでないので，この機能を明確化することは，ソフトウェアの生産を近代化する上できわめて重要である．この分野ではPSL/PSAが有名であるが，より，リアルタイム用の機能仕様記述を可能とする方向の研究開発が，国外・国内で行われている．機能仕様記述と同時に，解析機能によりシステム設計の不具合を指摘し，設計ミスを早目に修正することが狙いである．仕様記述においても，言語形式と図形入力形式があり，特に図形化はソフトウェアの機能仕様をvisible，目に見える形にする意味で重要性は大きい．

テストフェーズでは，テスト実行範囲をチェックしたり，テストデータを与えたテストの自動化等は実用レベルにあり，最近ではソフトウェアの必要テスト項目を自動的に指示するような技術の開発が重要となってきている．ソフトウェアの機能の原因—結果グラフより，計算機が必要最小限のテストケースを判断するものである．このような手法を用いると，人間によるテストでは，ややもすると正常ケースのテストが重点となり，異常ケースのテストがおろそかになり勝ちであるが，機械化により，テストの完全性を期することができる．

このようなソフトウェア工学の種々な新しい技術の開発と，これらを一貫データベースのものに統合し，また機能分担インテリジェント化されたワークベンチを多用化することにより，ソフトウェア生産の近代工業化が図れるものと考えられる．

一方，マイクロコンソフトウェアに関しては，若干

異なるポイントにウェイトがあると思われる．ソフトウェアの規模も，大型機，制御用計算機とは異なるので，ソフトウェア工学的手法にしても，簡易なシステムの導入が先決である．マイクロコンソフトウェア開発支援システムは，当概機種を用いるレジデントシステム（自立型）と，大型計算機を利用するクロスシステムの2種がある．マイクロコンソフトウェアの開発人員が増加してくると，レジデントシステムの台数の増加が必要となり，クロスシステムの利用が重要となる．また，プログラムテストにおいても，大型計算機内にシミュレータを置き，大型計算機によるクロスのテストが効率向上の立場から有用である．外部デバイスとのインタフェースの多いプログラムは実機テストによらざるを得ないが，論理的な部分に関してはクロステストはメリットは大きく，また，構造設計フェーズをサポートするツールも一貫して使用できるような支援システムが有用と考えられる．

ソフトウェア業における技術移転の必要性

吉村鐵太郎

ソフトウェア業界の一員として，80年代におけるソフトウェア工学の役割についての私見を述べる．

ソフトウェア費用の上昇は，ソフトウェアの生産，購入，利用すべての立場において深刻な問題である．このままの勢いで計算機の応用が拡大すれば，多くの企業においてソフトウェア費用は遠からず経営を圧迫する一大要因となるだろう．そして大型ユーザやメインフレーマ等，ソフトウェアに大きく依存する企業は当然この対策をより真剣に考えるはずだ．

たとえばメインフレーマはハードウェアの売込みに伴うユーザサービスのため等に貴重な技術エネルギを消費することは極力さけ，ソフトウェアの信頼性向上と生産合理化の研究開発に向かって，資本投下と技術者の温存を一段と加速しはじめたのではなかろうか．そしてこれに類した傾向の直接間接の結果に，ソフトウェア業界が今日活況を呈している大きな原因の一つがあるように見受けられる．

この見方が正しければ，現在ソフトウェア業が主な取引先としている企業のうち相当数は，やがて強力なソフトウェア生産技術を駆使するようになり，外注の必要が大きく減る可能性は十分に考えられる．

その結果，ソフトウェア業界の発言力と地位は大幅に低下し，知識産業どころか一方的な買い手市場の中でいつまでたっても労働力提供業に甘んじなければならないかもしれない．

これを克服し知識集約産業として自立するためには，

- 競争力の高いソフトウェア製品をもつこと
- 高い技術水準に基づくコンサルテーションや教育の販売能力の蓄積
- 先進的なソフトウェア開発環境（日本語文書処理能力を含む）の自社保有．

などが急務である．

これらの目標の実現には，経営の全方位にわたる方策の展開が必要だが，技術的見地からすればプログラミング技術水準の向上を置いてほかにない．特に新しいソフトウェア技術や開発環境の現場への導入普及を急ぐべきである．正にこの文脈においてソフトウェア工学の進歩発達に期待する所が大きい．換言すればソフトウェア業における技術移転の促進が緊急に必要なのである．

プログラミングにおける素人芸からの脱却と工学的接近が叫ばれて久しいが，現場を見るとき，たとえばドキュメント基準などの形式的実施に終始して，属人的かつ断片的なスキルの集まりがほとんど唯一の技術的蓄積であり，最も基本的なクラフトマンシップすら常識になっていない．大学等におけるソフトウェア教育が理学部的見地に偏しているとの批判が一部にあるようだが，むしろ現実の水準は，工学部レベルどころか平均的に見て工業高校のそれにすら達していないことは，続発するソフトウェアトラブルを見ても明らかだ．

ソフトウェア技術の最近の成果のうち，欧米の産業界ですでに実用性についての実績やコンセンサスのあるものがすでに相当あるが，その多くはまだ日本のプログラミング現場にとって未知に等しい．

技術移転の第一歩として，まずすぐれたプログラミング技術，規範の導入から始めることが何をおいても必要だろう．参考までに米国におけるこの種の技術の採用例を示す．

- 抽象プログラム，段階的詳細化，数学的検証等のアプリケーションへの適用，(Mills, IBM FSD) IPT 等への言及は全く見られない．
- 抽象データ型の OS 設計への適用 (Belady, IBM Yorktown)
- Guarded Command 概念，抽象データ型の OS 設計への適用 (Honeywell)
- 艦載機 (A7E) 用ソフトウェアに関する形式的記述，抽象データ型，プロセス同期技術，Hoare モニタ，Parnas モジュール等の応用 (Parnas 他，米海軍技研)
- 抽象機械概念と段階的詳細化及び検証の OS 設計への応用 (Stanford Research Institute)