報　告

パネル討論会

ソフトウェア・クライシス

昭和60年前期第30回全国大会†報告

パネリスト
伊藤　正雄[1]，大石　東作[2]，久保　宏志[3]
鈴木　　弘[4]，**司会**　川合　英俊[5]

総合司会　「ソフトウェア・クライシス」という題で，パネル討論を始めます．

司会の川合さんは，31年に北海道大学理学部物理学科を卒業されて，気象庁に入られ，その後現在の電子技術総合研究所に出向され，以来電子計算機を研究されております．

特にこの間，大型計算機の設計，TSS，コンピュータ・グラフィックス，コンピュータの性能測定あるいはネットワーク，についての研究に従事されておりまして，昭和56年にIPA，情報処理振興事業協会の技術センターの所長になっておられます．また，58年に東大から工学博士の学位を授与されており，ソフトウェアの開発の研究に，従事されてこられた方でございます．

よろしくお願いいたします．

川合　たくさんの方に集まっていただきまして，どうもありがとうございます．せっかくですから，実りの多いパネルにするため，活発に討論されるようにお願いいたします．パネラの方も，忌憚のないご意見をはかれて，業界，学界に新しい風を吹き込んでいただくように，張り切ってやっていただきたい．

最初に，1人10分ぐらいずつ，考えていることをご紹介していただきまして，そのあと皆さんと一緒にしばらく討論し，後半また，5分ぐらいずつパネラに話していただく，というふうにして進めていきたい．

まず，私が口火を切りまして，数分，ソフトウェア・クライシスとは，どういうことを考えているのか，ということと，なぜいま壇上にいらっしゃるような4人の先生方にお願いしたか，ということを，お話しいたします（**図-1**）．

ソフトウェアについて新しい考え方でアプローチするとき，3つのことを考えるのがいい．第一に，ソフトウェアの中身ですが，プログラムとそれに関連したドキュメントを合わせて，ソフトウェアというふうにとりあえず考えておきたい．そのほかに，まだ紙に書き表していない頭の中にあるものも入れてこのことばを使う場合がありますが，それはあまり考えないほうがいい．

第二に，プログラムは，機械が理解する情報ですが，じつは，人間の世界に対しても，かなり意味をもっているという面があります．

技術系の方がここに多いので，皆さんは情報の形式的なことばかり相手にして暮らしているわけですが，きょうは少し広げて，人間の社会にどういう意味をもっているか，ということまで考えてみる．

第三に，そういう生産財がどういうふうに世の中で産み出され，流通しているかということでございます．つくるときには，どういう問題があるのか，できたものを流通させたり，使ったりするときに，どういうことがあるのかということも考えなければなりません．実は，つくる前に，どういうものをつくるかを考える場面がございまして，そこには大変深刻な問題がある．

次に，クライシスとは，非常に問題が多いということです．その問題を解かなくちゃいけないんだけれども，うまく解けないということで，なんとかしないといかん，で困ったことだなというのが，クライシスということばの意味でございます．ですから，どんな問題があるのかというのが1つの観点です．解けないというけれど，ほんとうか，ほんとうに解かなくてはいけないのか，ということを考えてみる必要があると思

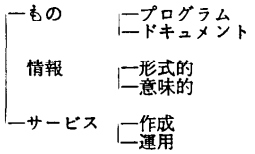


図-1　ソフトウェア

† 日時　昭和60年3月13日，12：30～14：45
場所　工学院大学〔A会場〕

1) 新日鉄，2) 電総研，3) 富士通，4) 構造計画研究所，5) IPA

┌多くの問題がある
└うまく解けない
⟶ 困ったものだ

1950年代
ソフトは修正しにくい
⟶ FORTRAN

1960年代
巨大ソフトは作りにくい
⟶ ソフト工学

1980年代
社会的信用
新しい需要
⟶ 認知工学？

図-2 クライシス

います（図-2）.

振り返ってみると，50年代に，ソフトウェアは修正しにくくて困るという話がありました．このときには，まだクライシスということばはありませんでしたけれども，非常に深刻に悩んだわけです．ところが，幸いなことに，コンパイラをつくれば随分楽になる，という発明があって，56年ですが，FORTRAN が発明され，随分状況が変わりました．大変な階段を登ったわけです．

同じ50年代の中ごろに，もう2つほど，実は大変な発明がございました．

1つは，Chomsky が生成文法という提言をし，言語理論でかなり進歩がありました．この考え方は，まだ全部は解決しておりませんが，情報処理技術に今では相当取り込まれている．

それからもう1つ，大脳生理学で，アミノ酸を検出する技術が非常に発達してきたことです．いまは，頭の中の神経細胞の中を，どういうふうにホルモンが通っていくか，顕微鏡的にもわかるようになっています．そのちょっと前に，DNA の大変な発見があって，遺伝子工学ができたことは，皆さんよくご存じだと思います．大脳生理学的な知見を情報処理技術に取り入れることは，いまのところまだ全然行われておりません．

それから，60年代に OS をつくったら，なかなかできなくて困ったという事件がございます．このときには，ソフトウェア・クライシスということばが初めて出ました．そのころ発明された，ソフトウェア工学というものの評価は分かれていて，さっぱりメリットがないというのと，いやここ20年で相当よくなっているんだという説とがあります．この辺，先生方がどういうふうにお考えになっているかも，興味深いところです．

さて，今またなぜ新しいクライシスかについて，私は，2つのことをいっておきたい．

1つは，ソフトウェアが，社会的信用の問題になっているということです．バグがあると，人が死ぬ危険性が出てきた．プライバシが侵される心配がある．社会に受け入れられるためには，質的に工夫しないといかんということが1つ．

もう1つは，新しい型の需要ができていること．つまり，従来は1つの企業の中で，人手をコンピュータに置き換えていたんですけれども，最近は，コンピュータを使って新しいサービスを産み，新しい産業を起こし，企業間が連携をとるという意味で，随分変わってきている．それらに対して，対応ができないことが，新しいクライシスの特徴だと思います．

方法論的な発明はいまのところあまり徴候がない．わずか知識工学とか，認知工学とか（このことばは，まだ定着していない）ができて，そのバリアを越すかもしれないと思って話題を提供いたしました．

これからどうしたらいいのか，全然考えられないわけではない．1番目に，情報産業に携わっている人間自体が，情報化を進めること．「弱い連携」と書きましたのは，「強い連携」が，契約に基づく支配関係の約束であるのに対してです．もっと気軽な，責任をともなわない情報の交換を促進しないといかんということが，指摘したいところでございます．それで，きょうはいろんな世界の人に，登場していただいたわけです（図-3）.

まず，情報産業ではなくて，別の産業から，新日鉄の伊藤さん．

伊藤さんは，情報システム部長をなさっておりまして，制御系のシステムの場合の問題とか，事務処理系のシステムの場合の問題とかを，比べて考えられる位置にいらっしゃる方です．

それから，学会関係，研究会関係では，ソフトウェア工学でどのぐらいうまくやってたのか，これから先どうか，ということについては，電総研でソフトウェ

情報産業の情報化
⟶ 再利用と自動化
弱い連携を深める
（業界，学界，他の業界）

産業の現代化（近代化ではない）
⟶ 異業種間の融合

需要は供給者が発掘

図-3 解決へのアプローチ

ア・エンジニアリング関係の研究をなさっている大石さんに，来ていただきました．

ソフトウェア業界，情報産業界の状況も非常に大事なことでして，人が集まらないとか，給料が安くて困るとか，いろんなことがありますので，少し丁寧に，2人にしました．まずメーカを代表して富士通の久保さんは，ソフトウェアの部長さんをなさっており，その品質の管理だとか，人を集める話とか，出荷したものの状況をどうしたらいいか，などの見解をお持ちです．

それから，鈴木さんには，ソフトウェア業界を代表していただきました．ソフトウェアをつくる，あるいは運用することを専業にしている企業が日本に3000から4000社ぐらいある．鈴木さんに，情報処理産業の観点から，この問題にアプローチしてもらいたいと思います．

2番目に，ほかの産業がどうなるかということでございます．産業の近代化とは，大量生産して物を安く売って，市場を制覇していくことなんですが，これからは，そういういき方ではないんだろう．現代的な産業は，異業種間と融合して，新しい価値をつくっていくという方向に進まないといけないと思います．

3番目に，従来のソフトウェアは，需要があるからそれを聞き出して，コンピュータに置き換えて供給していくというアプローチを長い間とってきた．しかし，いまになってみると，発注者，需要者自身がどういう要求をもっているのか，知らないでいる面がある．だから，供給者が協力して，需要をつくっていかないといけない．需要とは，契約に基づいて金を支払うという意思を表明することですが，そこまでプッシュしてもっていかないといけない．

伊藤 ソフトウェア・クライシスというものが，現実にどういう形で存在しているのか，議論のあるところでしょうが，私ども製造業でのコンピュータ利用は，製造ラインの自動化を含む総合生産管理の分野と，オフィスを中心とする情報システムの分野とに大別しており，それぞれ特長のあるシステムを構成しており，したがってソフトウェアの構造や開発維持についても多面的な問題を持っておりますので，まず私どもの感じております問題点を若干提起させて頂きます．

一般に企業の管理システムは，**図-4**（情報システムの全体概念）のように3階層で整理されることが行われております．

鉄鋼業の場合，右下の部分が製鉄所であり，その他は本社部門の業務であります．

鉄鋼業は比較的早い時期からコンピュータの利用が開始された業界でありますが，現在のところ，大手の

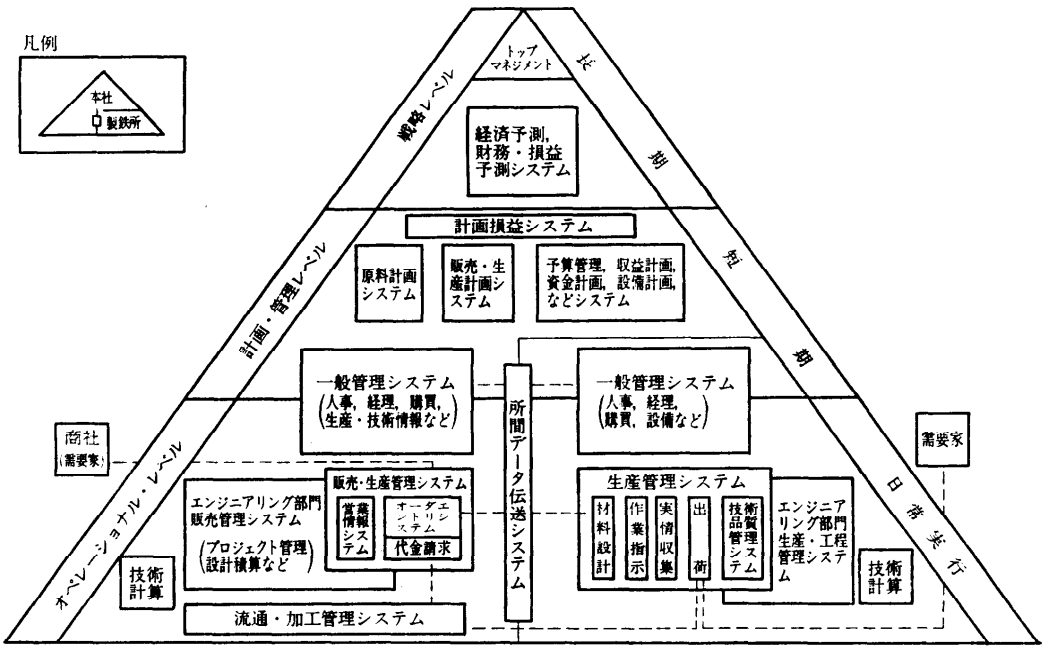


図-4 情報システムの全体概念図

各社は平均して約3000万ステップのプログラムを保有しているといわれておりますが，ほぼ 70～80% は生産管理システムであります．しかも無人化を志向し，かつ複雑高度化する品質・工程管理への対応はますます強まってきており，今後とも年間 10% 程度の増加が必要ではないかと言われております．

図-5 に鉄鋼業の生産販売管理の特長を整理してみました．全量受注生産で，板1枚といえども注文書にしたがって1品ごとの品質・納期に応じて造られます．したがって，受注から生産に至る情報の処理も膨大なデータのハンドリングを必要とします．また生産設備も，溶鉱炉や製鋼炉・圧延設備などいずれもきわめて高速大容量であり，かつ各工程別生産を保証するとともに，さらに全工程を通して品質的にも，工程管理的にも最適化を実現するためのいわゆる総合一貫生産管理システムのコントロール下にあります．

（1） 受注生産方式
（2） 流通面における商社機能の活用
（3） 製鉄所を中心とした総合一貫管理体制
（4） 原料から最後成品に至るまでの工程が長くかつ多様
（5） 各製造工程及びそれを結ぶ輸送工程が各々バッチプロセスかつ全体がブレークダウン構造でタイトな一貫構造
（6） 24時間ノンストップ操業

図-5　鉄鋼業の生産販売管理構造面の特長

図-6 は，組立ラインと比較した鉄鋼のブレークダウン構造の特長を示したもので，製造工程でのトラッキングと異常処理の必要性がおわかり頂けると思います．

図-7 は圧延工場を例に，生産管理システムのコンピュータ・ハイアラキを示したものであります．工場の各工程はプロセスコンピュータの下で，自動化を達成しております．

このためには，与えられた生産前提に対し，1品ごとに正確なトラッキングを行い，それと同調して高精度の品質工程情報に基づく即時判断と異常例外の処理が不可欠であります．したがって自動化ラインの生産管理は巨大な制御システムであり，設備と技術と品種別製造条件に対応したデータの処理，判断処理ロジックで構成されております．このため，ソフトウェアは個別対応型で大型のものになっております．さらに製造ラインに必要かつ十分な生産条件を与えるための生産計画・スケジューリングも，各工程の特性と全工程の総合的な最適化に応じ，複雑高度なものを必要と致します．すなわち，自動化を志向する生産管理システム

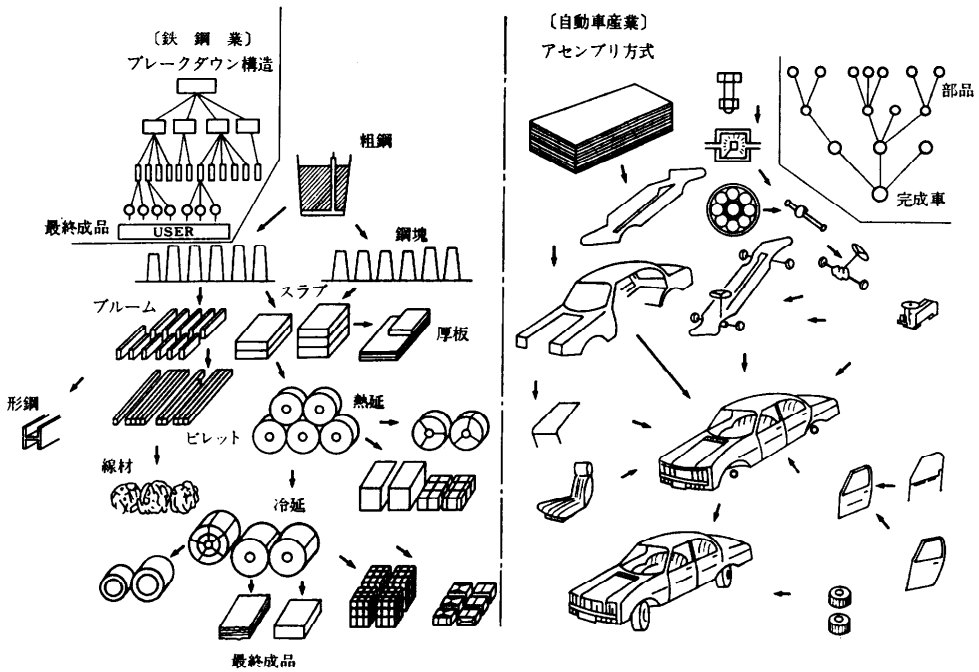


図-6　鉄鋼業の生産管理の特徴

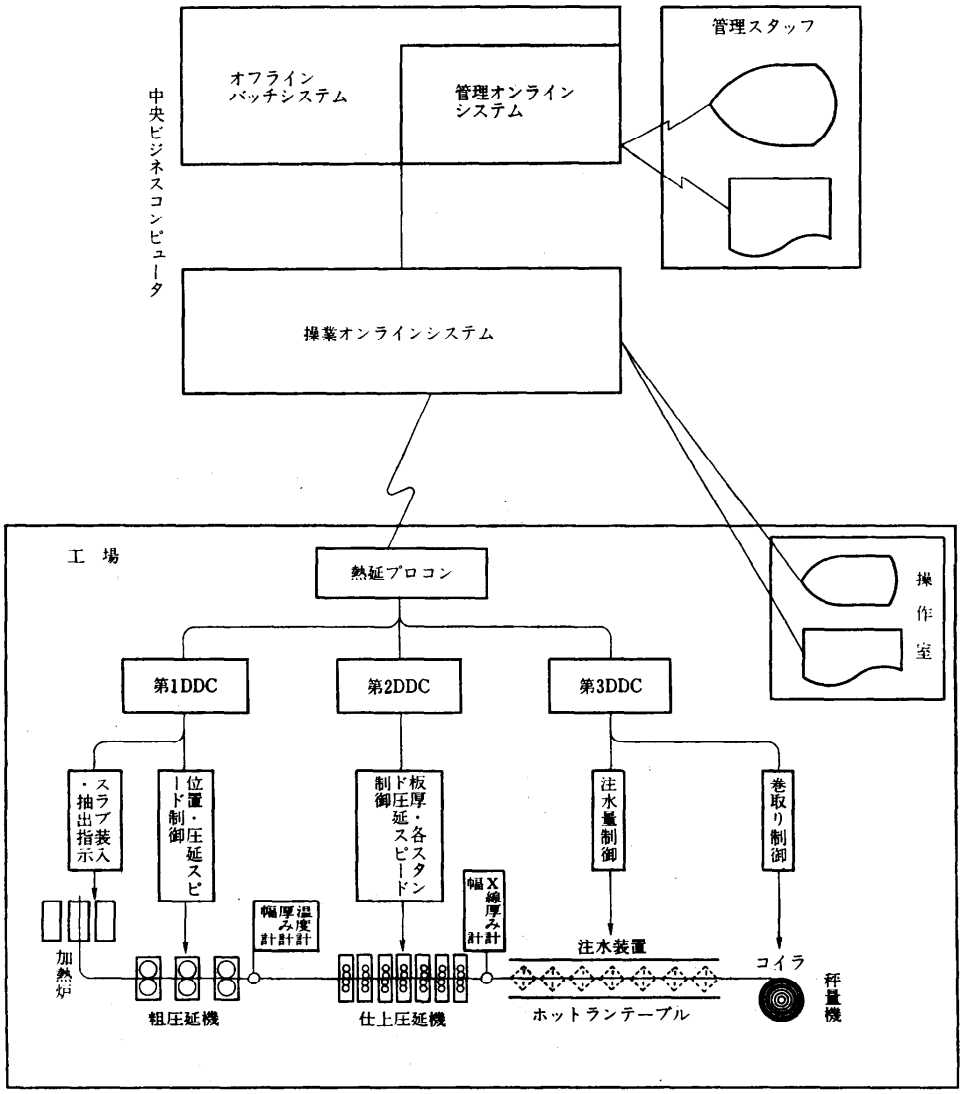


図-7 コンピュータ・ハイアラキ

は，設備，製造技術，品種別製造特性などに応じ，しかも品質や工程の安定度にかかわっているため，ソフトウェアの開発維持は現在のところ，オーソドックスな方法によらざるを得ない状況にあります．

最近の鉄鋼業の連続化自動化は，さらに図-8 のようなレベルに達しております．溶鉱炉から圧延工場に至るまで完全に連続化されております．このため，熱エネルギでも，歩留りの面でも，世界最高レベルの成果を得ているわけですが，このためには前述の総合一貫生産管理システムが，製鉄所全体をコントロールしなければなりません．したがってアプリケーションソフトウェアもまた冒頭述べたような膨大なものとなって参ったものであります．またこのようなシステムは，開発はもちろんのこと生産環境の変化に柔軟に対応するため維持運用がより重要であります．

このような生産管理システムの特性に対し，オフィス・システムは現在開発中でありますが，人に対する情報提供を主たる目的とするものであり，現在各方面で研究開発されつつあるデータ・ベースシステムや統合的な OA システムの適用できる分野であろうと考

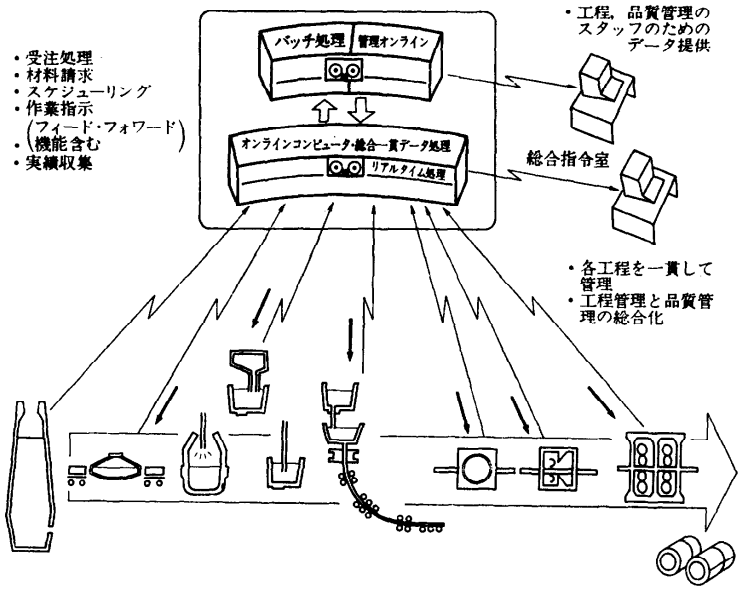


図-8　工程の連続化

えております．

鉄鋼業を例としてコンピュータシステムの利用環境を概観したが，アプリケーション・ソフトウェアの開発維持に関する問題を簡単に申し上げると，第一に今まで申し上げた，生産管理を中心とするオペレーショナル・レベルのシステムは，自動化システムを含めて今後とも継続することであります．したがって現行の開発維持の方式・生産性を続ける限り，要員での危機的状態は一層深刻になってくると思われます．また要員面でソフトウェアの開発が減速されるとすると，今後企業の経営展開に与える影響も重大であります．第二にシステムの質的な問題があります．複雑膨大なシステムの質的管理は，一般にソフトウェア生産技術上の問題に止まらず，生産製造管理上の問題の解き方いかんによって，歩留りや品質保証レベルなどが左右されることにもなります．よいソフトウェアとは何かという問題意識であります．また同じ問題を解く場合，システム・エンジニアのいかんによってステップ数が大きな違いになる場合もあります．

第三に人材の育成確保の問題があります．ユーザにとってソフトウェア開発維持に必要な人材には，二つの重要な専門性を必要とします．すなわちコンピュータシステム技術に関するものと，ユーザ業務の仕組みに関するものであります．システムが高度になりますとこの両面はますます精細複雑となり，それなどを統合してアプリケーション・システムとして組み上げる技術者の育成確保はきわめて重大な問題であろうかと存じます．また問題の解き方につきましても，最近知識工学的方法が研究されておりますように，斬新なアプローチが必要になってくるものと思いますが，そのような可能性を持った人材のニーズが高まってきております．

最後に，汎用アプリケーション・プログラムの流通に関してであります．私どものシステムの大部分は，前述の特性をもっておりますが汎用パッケージの活用も当然積極的に進める必要があります．流通するソフトウェアの条件，及びソフトウェアが流通するための市場の条件はもっと具体的に検討する必要があるように思います．

高橋(東芝)　よいソフトウェアとはという話の中で，技術の良い人が作るとステップ数は短くなるのではないかということでしたが，そのときのソフトウェアの価値と技術の価値をどのように評価しているのか．

伊藤　ソフトウェアとしての表現が良いかどうかということと，ソフトウェアの生み出す価値はどうかという面があると思いますが，明確な基準をもっているわけではない．ただ人間のシンキング・プロセスをそのままプログラムするような技術では問題があり，ソ

フトウェア開発技術の良否が，莫大なコストの差を生ずるため，重大な問題だと思っている．

鈴木　きょうは，ソフトウェア産業に携わる方は，多くはないと思いますが，業界の立場から，「ソフトウェア・クライシス」という問題をどう考えているか，述べさせていただきます．

論文集に書いてございますが，需要の質的な変化が，急激であり，それに対する供給側の態勢としては，これまでとは違った態勢，考え方をとらなければいけないというのが，結論でございます．

それでは，ソフトウェア・クライシスがあるという論拠を展開している人たちは，大体次のような線に沿っているのだと思います（**図-9**）．

つまり，需給のギャップが拡大している一方，価格は上がっていく．するとソフトウェアが入手しにくくなるので，コンピュータの普及が鈍化する．もう少しおおげさにいう人は，それによって世の中の発展に支障をきたすと．

こういっているのだろうと思います．

ところで，世の中のデータをいろいろ分析してみると，クライシス存在論の展開を裏づけるものがない．ソフトウェア業界売上高は年 26% という高率で急激に伸びている．それだけソフトウェアの需要が伸びてるのだといわれるが，そうではない．価格が高くなるといわれるけれども，まったく高騰などはしていないという状況なのです．

このことについては，日本興業銀行の吉田さんがよく分析されて，レポートを出されています．少し内容を紹介させてもらいますと……需給のボリュームではなく質的構造は，おそらく5年なり10年先を考えてみますと，相当程度変わっていくのではないかと思います．

まず，ハードウェアのコスト・パフォーマンスは年率大体 15% ぐらいよくなっており，ほぼそれに見合って日本のコンピュータ・ハードウェアの売上げが伸びている（**図-11**）．

ソフトウェア産業の伸びが 25% になっているということはソフトウェアがそれだけつくられる必要があるのだというように，誤解されている．したがってこの25と15の差が非常に問題だといわれているのです．

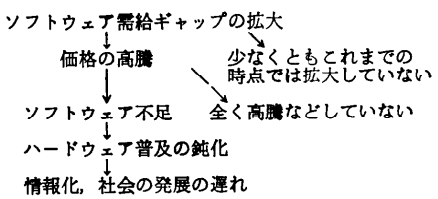


図-9　ソフトウェア・クライシスはあり得るのか

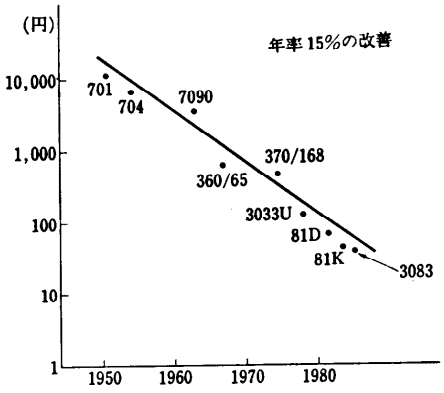


図-10　コスト・パフォーマンスの推移：価格1秒当り実行命令数

ハードウェアの伸び	15%
(対)	
ソフトウェア産業の伸び	25%
ソフトウェア生産量の伸び	15%
ソフトウェア技術者総数	9%/年
生産性上昇率	5%/年
コンピュータ設置台数の伸び	21%/年

図-11　日本興業銀行調査（吉田素紀）

実は，どうもそうではないのではないかというのが，先程のレポートの主張です．ソフトウェア技術者の総数に注目してみます．これはこの業界だけではございません．伊藤さんのところのような一般企業のEDP 部門にもソフトウェア技術者はいるわけで，ここには大体現在15万人ぐらいの方がいるといわれています．そういう人たちもあわせまして，その数は，年9%ぐらいしか伸びていない．

つくられるソフトウェアの量は，つくるソフトウェアの技術者の数に依存すると考えれば，あとは生産性の上昇がどの程度かということになります．せいぜい 5% ぐらいしか伸びていないであろうというのが大方の意見の一致するところです．とすると，これをかけましても実際のソフトウェアの生産量は 15% 程度だということで，ハードウェアの伸びの 15% と大体一致しているということになるわけです．

ましてや，コンピュータの設置台数あたりでいきますと，設置台数の伸びは20数%ですから，むしろ逆になっているはずです．したがって一台当りのソフトウェアの生産量はむしろ少なくなっている．

以上は興業銀行の調査が，これに私なりの分析結果

を加えてみます．それは先ほどありました価格の上昇の問題です．これもまた，どうも信憑性が薄い．

これは，決して私ども業界が不満を述べているわけではございませんで，客観的な数字を出してみました．

つまり，技術者が非常に不足している．不足しているなら，当然引く手あまたなのだから，その待遇は相当改善されているはずということに，世の中の常識としてはなるのですが，どうもそうはなっていない．

つい最近出ましたコンピュータ白書（**図-12**）を見ますと，たとえば，この業界でソフトウェアの生産に従事している人たちの平均給与は，全産業の平均給与より低いのです．

この業界と言いましたのは，情報サービス業として，ソフトウェア産業のほかに，いわゆる情報処理サービス，さらに，ソフトウェア・プロダクトを販売する．あるいはデータベースを提供する，そういったものを全部含めてしまいましたが，大変なことに，2割近く低いのです．

ソフトウェアの価格は上昇していない
あれほど技術者の不足が叫ばれていながら…

月額平均給与　（千円）

	プログラマ	SE
全産業	192.5 (28.8)	243.2 (33.7)
情報サービス業	164.4 (26.4)	210.7 (31.1)

図-12　コンピュータ白書（1984～85）

大きく変化する
↑
- 米国の市場規模：76年より急速に拡大
- 日本は6～7年の差
- マイクロ・コンピュータの普及
  - ソフトウェア・プロダクト
  - システム

需要者主導の価格形成
⇒性能，需給バランスに見合った価格形成

図-13　これからの需要構造は？

- ソフトウェア技術者
- 生産の工学：software engineering
- 製品・技術の流通・移転
- ソフトウェア作りを減らすハードウェアの活用

⇑

政府，	学界，	産業界
↙	↓	↘
真にソフトウェア産業の育成を	さらに一段の研究努力を	最も責任が重い経営者の意識改革

↙

ソフトウェア・インダストリ・クライシス

図-14　変化する需要への対応

このデータから見ると，ソフトウェアの価格が上がっているというのは，どうもおかしい．部分的には上昇しているのでしょうが，上がっているけれども経営者がみんなもっていってしまうということなのかも知れませんが…．

さてそれでは，そのままの状態がいつまでも続くのかというと，そうではなさそうです．

なぜ，変わると思うかという理由の最大なものは，やはりマイクロコンピュータの問題だと思います．

これは，コンピュータの利用者の側にとってもいろいろな問題があると思いますが，少なくとも私ども専業者の立場から見ましても，このマイクロコンピュータあるいはパーソナルコンピュータの普及は，ソフトウェア・プロダクトの普及という点で，大きなインパクトを与えつつあります．

それから，もう1つは，システムです．このシステムというのは，たとえばシステム・ハウスというときに使うシステムという意味でございまして，そういうビジネスの仕方が大きく成長していくという点で大きなインパクトを与えつつあります．

さらに，米国の市場規模の伸びなどをみてますと，76年あるいは7年ぐらいから，急速に拡大しているのが，はっきりしているわけです（**図-13**）．大体日本の場合5～7年ぐらいの差で米国のあとを追っている．

これは，いろいろなデータからみてかなり確かなことです．76年の6～7年後，つまり，去年あるいは一昨年あたりから需要構造の変化が始まっているといえます．

論文集に書いておきましたが，そういう構造の変化がわれわれ専業者あるいは専門技術者にどういう影響を与えるのかということが重要な問題になってまいります（**図-14**）．

この変化への対応について私なりに整理してみますと，ここにあげました4項目ぐらいを指摘しておきたいと思います．1つは，技術者のレベルアップです．この技術者というのは，広い意味で促えて研究レベルにある人たちも含めさせていただきました．

それから，ソフトウェア・エンジニアリングの発展に，期待といいますか，もっと進んでほしいと思う．ここ5年から10年をみてみても，あるいはソフトウェア・エンジニアリングということばが生れてきたあたりからみても，どれだけの貢献があったかとなりますと，大石さんには叱られるかもしれないのですが，疑問を持ちます．それほど大きな貢献はしていない．

一段の飛躍を望みたいところです.

3つめは,プロダクトの流通,あるいは技術の移転をとにかく盛んにするということが,1つの大きなこの需要構造の変化への対応策だと思います.

最後は,ハードウェアの活用という問題です.非常に低いレベルでいえば,ソフトウェア技術者が,端末をもっと使うということですが,もっと積極的にはソフトウェアづくりを減らすという意味でのハードウェアの活用を考えるべきではないかと思います.

以上のような策を講ずるためにも,やはり基本的には,政府,学界,あるいは産業界,こういう人たちが一体となってとにかく本気になってこの問題を解決する努力をしないと,だめなのではないかと思います.

たとえば政府にしても,一方では,ソフトウェア産業はこれからの日本の基幹産業なんだというふうにいいつつ,他方では,官庁の中でのソフトウェアづくりがどのような仕掛で行われているか考えてみると,ほんとうにそういうことを考えてやっているとは,思えない面がかなりあるような気がします.

また,IPA も必ずしもそういうふうにまっしぐらに向かって進んでいるともいえない気がいたします.

それから,学界でございます.この情報処理学会もその役割を担うものの1つだと思うのですが,さらに一段の研究努力をお願いしたい.教育という面でも,非常に私どもとしては期待しております.

しかしながら,やはりこういうクライシスといいますか,問題を引き起こしている責任がもっとも重いのは,残念ながらこの業界,ソフトウェア産業界であろうと思います.

この業界は,先ほどのように,とにかく年 20% 程度の伸びをしている.これはある意味では,この業界にとって不幸なことでもあるのです.つまり,ぬるま湯の状況が,ずっとつくられてきているわけです.したがいまして,経営の立場としては,ほとんど何もせずに伸びていくことができるわけです.

これは,非常にまずいことであります.ただここに来て,先ほど来申し上げておりますように,需要の構造が大きく変わりつつあり,この業界の経営者の意識改革が必要になってきていると思います.

ですから,私はソフトウェア・クライシスというのは,むしろ,ソフトウェア・インダストリ・クライシスだというふうに思っておるわけです.

このような,いろいろな手だてがとられなければならないのですが,具体的なこととして,最近ちょっと私が気になっておりますことが,2つあります.1つは以上のような状況に対処するいろいろな試みの1つとして,この4月から,SIGMA プロジェクトというのが始まると聞いております(図-15).

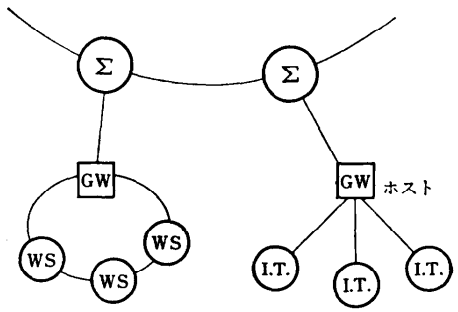
これには,私どもの業界は非常に期待をしております.このプロジェクトの中味はまだはっきりしていないようですが,その一つの要は,とにかく日本全国に,ソフトウェアに関連する人達がアクセスできるネットワークを張ってしまおうということのようです.これは,ソフトウェアの生産性の向上に大いに寄与すると思います.

つまり,情報ないし技術の移転が,このネットワークによって促進される.そういう意味では,ソフトウェア作りへのインパクトが大きいので,ぜひこのプロジェクトを,うまい方向に進めていただきたいと思います.

それから,もう1つ.私はソフトウェア業界の団体であります情報サービス産業協会の技術委員会のメンバになっております.そこの人たちの間で,ソフトウェア技術者のレベルの向上について真剣に考えないといけないという気運が出ており,そのために技術者のプロフェッショナル・ソサイエティをつくろうか,という話題があります(図-16).

これはなにも,情報処理学会に対抗してやろうということでは,決してないのでソフトウェア作りの現場に携わるプロフェッショナルたちが,自分達自らが進んで向上努力をしなければならない,ということがそもそもの発端です.

先ほど川合さんのお話にもありましたように,ソフトウェアがいろいろな社会的な信用の場にもち出され



いろいろなレベルでのソフトウェアに関連する組織の NETWORK

インパクトは大きい

情報・技術の移転

図-15 SIGMA プロジェクト

てきつつある．そういう中で，それをつくる技術者の責任，あるいは逆に，社会からの技術者の認知，そういうことが重要になってきていると思います．

外国に行ったときなど，見知らぬ人に飛行機の中で，「あなたは何をやっていますか」あるいは「何屋さんですか」という質問をされる．しかし，即座に「ソフトウェア技術者です」とこう答えたいのですが，なかなかいいきれない．言っても通じないといいますか，そういうような状況が，いまだにあると思います．ソフトウェア技術者はもっと社会に認知される必要があり，それだけの責任を果たさなければならない．

そういうことがこのソフトウェアの問題を解決する1つの策になると思っております．

久保　ソフトウェア・クライシスは，ないんじゃないかという話を，鈴木さんがされたんですが，あるとすれば，それをどんなふうに捉え，どんなふうに考えていったらよいか．考えようによってはなにかいい知恵が出てくるかもしれない，と考えて，1つの提案をしたい．

かなり有名になった話で通産省がいってることです．いま40万人ぐらいソフトウェア技術者がいて，それが1990年，昭和65年には，160万人ぐらい必要になる．ところが，供給のほうは，100万人ぐらいしかいかない．したがって，60万人足りなくなるといういい方をされているわけです．

ソフトウェア危機というのは，こういうふうにみるんだという，この見方に同調することから始めたいと思います．

もしも需給ギャップがあるとすれば，それはどうして起きてきているのであろうか．需要構造と供給構造の両方に，何か不健全さがあるにもかかわらず，それを放置してきた結果として，あるいは，改善の契機にしうるようなチャンスは，いっぱいあったにもかかわらず，それを活用することを怠ってきた結果として，需給ギャップが起こっていると，そんなふうに疑ってみるわけです．

こう疑ってかかると原因らしいことが，いろいろと浮びあがってきます．供給構造については，人の数と1人当たりの生産性の積で供給が決まっていると単純に

ソフトウェア技術者の向上努力
社会的責任
社会への認知
　JISA/STC
　SEA 構想

図-16　プロフェッショナル・ソサイエティの確立

危機問題
需要>>ソフト人口×生産性
⇩
需要>ソフト人口×生産性

需要構造
- ソフトに頼りすぎる
- 流通機構が未発達である
- 情報公開の秩序ができていない
- マルチベンダシステムをいやがる
- 標準化が遅れている
- 再利用技術が未熟である
- 小さく作れるものを大きく作っている

供給構造
- エンドユーザを巻きこめていない
- 中央に住む日本人男性（青年）にこだわっている
- 英・数・国を実学として教えていない
- OA・FA・HA のために働かせ，SA のために働かせていない
- ソフトウェア開発のための設備投資に抵抗を示す
- 現場体験を技術向上に役立てていない
- 産業界は学界を有効に活用していない

図-17　ソフトウェア危機とその対策

考えて，それぞれについて見ていきます（図-17の上）．

需要構造のほうですが，ハードウェアにやらせればいいことを，ハードウェアでやるとハードウェアの原価が高くなるからというので，ソフトウェアに押しつけることが行われている可能性がある．

アメリカのソフトウェア・ハウスがつくったいいソフトウェアがあるのに，日本の国内にサポートするエージェントがないがために，使いたいんだけど使えなくて，似たものをつくっている可能性がある．

ソース・コードが開示されていれば，それを読んでちょっと手直しして自分に都合のいい，自分のほしいソフトウェアがつくれるのにソース・コードの開示がなされていないがために，全部つくることが行われている可能性がある．

あるシステムつまり，お客さんの問題を解決するためのシステム・ソリューションを組み立てるときに，サードベンタのものも組み合わせてオファできると，ちゃんとした答えになるのだけれども，そういうことを受け入れてもらえない．つまりシングルベンダ・システムを希望されるために，つくらなくてもいいものをわざわざつくらなければいけない羽目になっている可能性がある．

たとえば日本語のコードがもしも統一されていれば，変換なんてことは必要ないのに，それがそろってないがために，変換なんていう需要を生み出している面もある．

アプリケーションをある OS のもとでつくるとき

に，その OS が，そのうちに別の OS にグレードアップされることを見越して，OS 独立なつくりにしておけばいいものを，そういうことを配慮せずに，眼の前の OS に最適なつくりにする．その結果，機種更新システムグレードアップのときに大きな費用を発生させている可能性がある．

エンドユーザの要求を聞いてアプリケーションのスペックを決めるとき，十分時間をかけてスペックをきれいに決めればいいのにそういう努力をあまりしないで，つくることを急いでとにかくつくってしまう．その結果，小さくつくれるものを大きくつくっていることもおきている可能性がある．

以上のようなことを，需要の側にある需要ギャップをもたらしている原因として，あげたくなります（図-17 の中）．

もう一方の供給構造のほうに眼を移してまず人のほうをみてみます．

DP 部門の人たちは，仕事は全部自分がするのではなく，一番要求を知ってるエンドユーザにつくってもらいたいと思ってるはずです．そういうことをいうためには，エンドユーザでも使えるようなツールを供給しなければならないことも知っている．ところがそれができない．その結果，エンドユーザを招き入れて戦力を増やすことができないでいる可能性がある．

東京で発生した仕事は，東京の人がつくらなければいけないというふうな思い込みみたいなものがあって，一生懸命この東京で人集めをして，足りない，足りないといっている．あまり集める努力をしないでそういうことをいっているのではないか．供給構造を決めている，もう一つのソフトウェア問題の生産性のほうに話を移します．

学校で基礎学科として英・数・国を教わって社会に出てくるわけですけれども，社会に出てきて役に立つような英・数・国を教えてくれるように，学校の先生方にお願いをしていない．その結果として，人間の可能性を引き出すことをうまくやれていない．

ソフトウェアに従事している人たちは，やれ OA だ，FA だ，HA だと，いわば他人のために一生懸命働くばかりで，SA つまり自分たちのソフトウェアのオートメーションのためには，あまり働いていない．経営者のほうも，自分の部下のソフトウェア技術者たちをハッピィにすることよりも，お客さんのほうばかりを向いている．

同じ設計でも，ハードウェアの設計のためであれば，1台何千万円もする機械を技術者に与えるのに，ソフトウェアの設計者のためには，せいぜい100万円ぐらいの端末を与えて，しかもそれを数人でシェアさせる．このことは，生産性の向上を妨げている．

現場で毎日仕事をしていれば，技術を引き上げるためのヒントだとか，あるいは引き上げるために解決しなければならない問題にひんぱんに出会います．そういう体験を技術を引き上げるという目的のために，うまく使っていない．いい方をかえますと，ソフトウェア工学というのは，経験工学あるいは現場工学と呼びたくなるような要素を，たくさんもっているにもかかわらず，そういう視点からのアプローチをあまりやっていない．

生産性を高めるのは，われわれ産業界にいる技術者の仕事ですが，学会にうまく問題を提起して，学会から現場に役に立つようなコントリビューションを引き出すという努力を，あまりしていない．このような原因らしいことをあげたくなります（図-17 の下）．

需給ギャップをもたらしている原因を需給両面からあげた上で，一体どういうことが需給ギャップ解決のために世の中で具体的に行われているかを調べてみました．

その中から，鈴木さんの話とできるだけ重ならないようにいくつかあげてみます．

ハードウェア機器メーカにしろソフトウェア・ベンダにしろ，IBM の製品と共存したり，あるいは IBM の製品とつなぐために，どうしても IBM の情報が必要になることはよくあります．

そういう情報を IBM から EC の諸国の会社に提供する取り決めがかわされたという報道が去年の8月にありました．情報公開の努力が，IBM と EC の2者によって行われています．

通産省が中心になって，コンピュータ業界とソフトウェア業界も一緒になって行っていることにソフトウェア権法づくりがあります．これも，ソフトウェア開

- 情報公開の秩序づくり
  EC と IBM の契約，ソフトウェア権法，日米交渉
- 標準化
  ISO/TC 97/SC 7, UNIX のための日本語コード，ビジネスプロトコル統一
- 品質保証
  品質表示制度，システム監査基準，高信頼化技術の JIS 化
- 地方進出・海外進出・女性の戦力化
  労働者派遣事業問題の立法化，雇用平等法

図-18 ソフトウェア危機対策

発者の権利を守ることによって，公開を促していく努力の例です．ソフトウェアを法的に保護する法にはいろいろな案があって，必ずしも日本とアメリカでは合っていないようですが，意見の対立を調整するために当局同士の交渉も行われています．

これらの事実をまとめていえば，情報公開を促すことによって，よけいなソフトウェアの需要を抑制しようという動きとみることができる．

去年の ISO 理事会は，標準化の迅速化のために，ファーストラックという手続きを導入していますし，TC 97/SC 7 は，その守備範囲をソフトウェア工学全域にまで拡大し，検討を加速化するために，いままで年1回しか会議を開いていなかったのを，間に1回入れるというような動きもしています．実際，ミュンヘンで2月にありました．

ATT が Unix を日本で普及させたければ日本語のサポートは欠かせない．そのためには，各界の同意が得られるような仕様にしなければいけない．ということで，ATT が日本語サポートの標準仕様を定める作業を行っているという話が伝わっています．

産構審の答申もあって，通産省はビジネス・プロトコルを統一するためのプロジェクトを起こそうとします．

こういったことは，標準化を早目，早目に実施することによって，変換なんていう無駄な需要を発生しないように抑制しようという動きであるとみることができます．

おととしの暮れのやはり産構審の情報産業部会の答申の中に入っていたものですが，世の中に不良品質のソフトウェアが出回らないように，品質の表示制度の提言がありました．

システムの信頼性とか安全性，あるいは経済性といったものを，外部の監査人によって評価をすることを促すために，通産省はシステム監査基準をこの1月に公表しました．

高信頼化技術の JIS 化のために，通産省は JIS 原案をつくって，この1月に公表しました．

これらは，悪い品質のソフトウェアが使われることによって発生する明らかな無駄な需要を抑制するための対策であるとみることができます．

地方進出とか海外進出とか，あるいは女性の戦力化によってソフトウェア技術者の数をふやすことは，いまかなりお互い競い合ってそういう動きをしていると思います．それが行き過ぎないようにという配慮もあってであろうと思いますが，労働者派遣事業問題の立法化だとか，あるいは雇用平等法のような動きもあります．

いくつかの動きを例示的に紹介しましたが申し上げたいのは，とにかくいろいろな危機対策はうたれていることです（図-18）．

にもかかわらず，こうやってパネル討論をもっているのは，そこで打たれている対策が，重点の置き方，方向性，あるいは対策に伴って発生する費用の負担などについて，必ずしも関係者の間でコンセンサスができていないからではなかろうかと思うわけです．

いま1つ1つの対策について，綿密に検討しないで何かを申し上げるとすれば，こういった問題に関心ある人たちが集まって，自由に意見交換するような広場をつくるのがいいように思います．

「ソフトウェア危機問題フォーラム」とでも呼ぶようなものをひとつつくってはどうでしょうか．

それには，とにかくいろいろな人たちが入らなければ意味がありません．人の集団に偏りがあると，出てくる意見に偏りがどうしても出てきます．ですから，法曹界の人たちもジャーナリストもはいってもらったほうがいいでしょうし，ともすれば，外資系の人たちを除外することが行われますが外資系の人たちにも外国人にもはいってもらう．諸官庁の人たちにもどんどんはいっていただいて，学問されている方も理学，工学だけでなくて，人文科学，社会科学の人たちにも入っていただく．そんなような集団にしていくのがいいのではないか．そうすれば何か知恵が出てくるのではないかという期待があります．

自由な発想から知恵を生み出すためには集団は，だれの支配も受けてはなりません．完全にお互い自由な個人の集団であるべきであろうと思います．

1つのヒントとして，戦前戦中の理化学研究所や京大の人文科学研究所のことを想い出します．産業技術史学会というのを，人文科学研究所にいた人々だと

- 産・官・学・法・ジャーナリズムのすべてから
- 産はメーカ・ソフトハウス・システムハウス・ユーザのすべてから，外資系も，外国人も
- 官は通産・郵政だけでなく労働・外務も，外国政府の人も
- 学は理学・工学だけでなく人文・社会科学の人々も
- 誰の支配もうけない自由な個人の集り

ヒント：理化学研究所
京大人文科学研究所
産業技術史学会

図-19 ソフトウェア危機問題フォーラム

か，あるいは今いる人々，あるいは出入りしていた人たちが中心になって最近つくったようなんですが，それらを勉強すれば，知的生産性をもたらす行動の知恵が出てくるのではなかろうか．そんなことを考えました（図-19）．

大石　私は，「ソフトウェア危機とソフトウェア工学」ということで話をしてみたい．

この2つは，不思議なことに，ほぼ同じころに，使われ始めたのではないかと思います．

川合さんから指摘がありましたように，すでに15年あるいは20年という年月がたっていますが，いまだ両方ともいわれている．かつ，久保さんから，ソフトウェア工学は一体何をやっておるんだ，ちっとも問題は解決しておらんではないかともいわれますが，それには理由があります（図-20，ハードウェアの急速な進歩に伴って）．

ソフトウェアに対する要求自身が，昔に比べればものすごく高度化，あるいは巨大化しているわけで，それに対してソフトウェア工学のほうからの十分な解答が出ていない状況だと思われます．

いまだ，手工業とはいいませんけれども，頭脳工業として，基本的には個人の頭の働きに大部分を頼っているようなソフトウェア生産では，結局生産性が低いとか，それに伴ってバックログが大量発生するとか，巨大なソフトウェアの維持管理が困難だと，そういう問題は依然として残っています．

ソフトウェア工学はある程度のことは解決したし，これからもソフトウェア工学という方向で解決法を示したいと思います．

現在，ソフトウェア工学の到達点として異存のないのは，高水準プログラム言語及びその処理系の完備と，一般化でしょう．それから，文書編集系，いわゆるエディタに関連するツールというものが，現場での最強の武器になっている（図-21）．

システムの実現段階においては高水準のプログラム言語は，最大の成果であるわけで FORTRAN の1956年の開発．それ以来綿々とこの方向で，いまだ研究を続けてるわけです．

最近，実用的な言語として米国国防総省の唱えてお

ソフトウェアの高度化，多量化，巨大化
手（頭脳）工業的なソフトウェア生産
　生産性が低い
　バックログの大量発生
　巨大なソフトの維持・更新の困難性

図-20　ソフトウェア危機とは何か？

ります Ada あるいは最近日本でも非常に盛んになってます論理型言語，オブジェクト指向型言語というものがあります．こういう言語の研究は，非常に重要ではあるわけですが，いままでの高水準言語の延長線上ということで考えているんだとソフトウェア危機は解決しないという気がします．

もう1つ別種の達成点ですけれども，これは文章編集とか，最近のワークステーションにみられるような多重画面管理技術の成熟ということがありますから，この面からも，ソフトウェアのつくりやすさは，増してきてる．

ソフトウェア作成の初期段階である要求仕様の定義技法が，すでに開発されている．いわゆるソフトウェア・ライフサイクルの概念の確立と，これにのっとったソフトウェアの生産支援システムの開発です．

しかし，こういうライフサイクルの概念ですと，要求仕様の定義あるいは設計，コーディングというふうに，各段階を非常に固定的に考えるあまり，ユーザからの仕様が初期の段階では決定しがたいこととあいまって，非常に使い勝手の悪いようなシステムが出てくる可能性がある．

それに対して，最近では，またラピッド・プロトタイピングという考え方を導入して，使いやすいソフトウェアがつくられるような方向も考えられています．

さらに別の考え方としては，自動プログラミングとか，あるいは形式的な方法があります．これらの方法自身は，いまだ研究段階にあり実際に解けるところは，トイ・プログラムの域を脱していないと考えられます．

それでは，この先，ソフトウェア工学によって，どうソフトウェア危機を解決していくのかというと，いくつかの方法が考えられる（図-22）．

1つは，要求仕様と，実現の段階とが，個々に分か

ソフトウェアの作成現場での最強の武器
　高水準プログラム言語処理系
　文書編集系（エディタなど）
● 要求仕様定義技法
　　ソフトウェア・ライフサイクルの概念
　　同上支援システム
● システムの実現段階
　　高水準プログラム言語
　　　Ada
　　　論理型言語
　　　オブジェクト指向言語
● 文書編集や多重画面管理技術の成熟
● 自動プログラミングや形式的方法

図-21　ソフトウェア工学の到達点

れた技術によって，それぞれ担われていたわけですが，これからは，技術を統合していかなければならない．そうすれば，現場での最強の武器は，コンパイラとエディタということはなくなり仕様段階もうまく含まれるようになります．

この方向では，現在も研究が進んでいまして，本大会でもいくつかの方法が発表されています．まだ普及が進んでいないのですが，これからは非常に可能性のある方向だと考えます．

それから，ソフトウェアの再利用が非常に重要だと考えています．

ただ，通常の生産におけるハードウェアの部品の利用と，アナロジで考えられていることが多いのですが，これには問題があります．

部品の場合は，いつまでもその部品が最終製品の中に残るわけですけれども，そういう形態でソフトウェアの部品を使ったとすると，余計な機能というものが最終製品であるソフトウェアに残ってしまうことになり，効率上望ましくありません．利用者も受け入れないでしょう．通常の部品という考え方をソフトウェアに直接あてはめるのは危険であると思います．

したがって，単にソフトウェア部品という格好で蓄積するのではなくて，現存するソフトウェア自身あるいはソフトウェアの設計法を抽象化して蓄積しなければなりません．

逆に抽象化されたもの自身は，そのままでは使えませんから，再利用するために，具体化の技法の開発が必要になります．この辺を着目することによって，従来の高水準言語でプログラムを書く段階からは，一段抜け出るのではないかと考えています．

また，先ほど伊藤さんからもご指摘がありましたように，知識工学的手法というものをソフトウェア工学と融合させる必要がある．ただし，ここで1つむずかしいのは，ソフトウェア工学における知識工学的手法は何かということです．何を知識としてのせればいいかというのは，これからの検討課題だが，抽象化技法あるいは具体化技法と密接に結びついてくると考えられます．

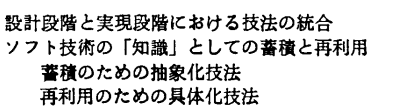


図-22 危機解決のためのソフトウェア工学

それから，再利用に関連してパッケージ・ソフトウェアというものの利用も進めていかなければなりません．

この辺は，業務の標準化ということともからみますが，こういうソフトウェアが普及する要因は，日本においてもあると思います．

さらに，パッケージ・ソフトウェアの概念と，先ほどいいました具体化技法を結びつけますことによって，実際の各カストマの要求に合ったようなものもつくられていく可能性があります．

もう1つ，パーソナル・コンピュータのほうから提案されてきた簡易言語が，1つ大きな役割を果すのではないか，いや，現実に果してます．

たとえば，表計算システムでは米国の VISICALC や我が国の PIPS があります．これらはプログラムを手続き記述という形態で作成するのではないので，事実上ソフトウェアの生産性はあがります．従来のソフトウェア工学の立場でものを考えていた者にとっては，アッという側面を持っています．

別の観点からすると，ソフトウェアのみならずハードウェアを含めた計算機の専門家の教育があります．すでにわれわれの持っている知識を，教育という場でいかに継承していくかです．知識工学的な方法を含めて，有効な方法が開発されれば，不必要なプログラムがどんどん作成されるという弊害も，教育の面からも防げると考えています．

川合 これから討論，質問に入っていきたい．まず短い質問のある方，短いというのは，回答が短いという意味ですが，どうぞ．

(少し間)

たくさんの話が出ました．パネリストの方には，この際どうしてもいっておきたいことを最後に，2分間ぐらいずつ言っていただきます．それまで約30分ぐらいは皆さんと一緒に話を進めていきたい．回答は，あとでまとめてやるようにします．

古宮 (IPA) 久保さん及び大石さんのおっしゃられたことは，非常によくわかります．

大石さんに質問します．

ソフトウェア工学の要求定義に関して汎用的なものがない．それから，要求定義とソフトウェアの自動生成とつながったようなものが，まだあまりない．その辺のところをどうお考えになっていらっしゃるか．これから先を考えてですね．

村田(富士通) コメントを少し申し上げたい．まず大きなほうからいきます．4人の話しにまったく欠如していると思うのは，消費者の観点で，遺憾だと思います．たとえば，エレベータのプログラムは，大学の3年生ぐらいの学生が，アルバイトでつくっているという話もあります．高信頼設計なんかしていないという懸念も大いにあるわけです．産業革命以来，人類が使ってきた機械に比べて，ソフトウェア（計算機も含む）の特徴は複雑であることなので，あらゆる動作を予知することが情報工学的にできるのかが問題です．ソフトウェアなんていう得体のしれないものに，われわれの運命を託すのは，非常に不安であるというのがソフトウェア・クライシスから想像する消費者の気持じゃないか．

それから，いままでのパネリストのご発言の中で，久保さんが包括的に具体的にご指摘になって，大変有益だったと思いますが，対策の中で，ちょっと触れてほしかったなと思うのは，通産省の推進している Interoperability の話をいれてほしかった．

それから，英・数・国を実学として教えてないというのをもう少し具体的に堀り下げていただきたかったと思う．

大きなソフトウェア・プロジェクトでは，設計にかかわる情報は，多分ドキュメンテーションを通して，上から下へ，下から上へと人に伝達するんでしょうけども，最近の情報技術者はハッカー的志向の人が多くてキーボードをたたけばなんかつくるんだけども，それを文章に書かせると全然書けないという世代が，生まれつつあると思う．国語とはそういう文章作成の実技でしょうか．

それから，意外と，数学のもっている論理的な側面は，最近は逆に後退しているんじゃないかと．これは初等教育においての話ですけど，という懸念がありますので．一松先生もいらっしゃるようですからご意見をいただければありがたいと思います．

それから，大石先生のお話ですが，あげられた題目には，奇異な面は1つもなかった．ただ，ほんとうのポイントは，どこなんでしょうか．これは質問です．

川合 最後の質問は，今後の見通しについてのポイントはどこか，ということですね．

一松(京大・数理解析研究所長) 教育のほうに携わっている人間として，耳の痛いこともいわれましたが，弁解は，今日は必要ないと思いますので，ちょっとコメント及び質問をしたい．

特に大石先生に伺いたい．教育の問題が非常に大事だということを皆様もおっしゃっているが，どういうふうにして実現するか．あえて申しますと，大学で計算機を教えてる先生は，私も含めて，頭が古いんでありまして，10年，20年前のコンピュータしか知らない方が非常に多い．

そういう人に教えてもらうと，かえってマイナスになるんで，そういう大学教授をいかにして再教育するかという問題，これはもう不可能なんですね．じつはそれは考えます．

それから，英・数・国に関しては，情報処理学会が前に大学に対してアンケートを出したことがありまして，非常にはっきり出ている．現代のアメリカのマニュアルの読解力，それから，日本語がちゃんと書けることという，これが学生に非常に要望されている．

数学も，コンピュータに直結した数学をやれと私もいっているんですが，ただ，そういうような要求をどうやって先生方に納得させるか，ということが非常に問題でございまして，これをへたにいうと，かえってつむじ曲げちゃうんで，逆効果なんです．

そういうことが必要であるということを，先生方に説得するノウハウを，これは私自身も責任がありますが，少し考えていただきたい．

もう1つ申しますと，もちろんフォーラム．非常に大事でございます．設備投資が大事だという話も，大分前から聞いています．

要するに，ソフトウェアをつくる工場は，アトリエである．椅子とか机は，非常に大事な設備投資なんだということは，随分前から聞いてるんですが，一向に改善されていないのは，なぜであろうか．そういうことはどういうところに行ったら実現するんであろうかと，考えているわけでございます．

半分コメント，半分質問でございますが，よろしくお願いいたします．

川合 再教育されないといけないということは，大学の先生方はおわかりになっていらっしゃるんですか．(笑)

一松 それが一番問題ですね．それを着眼させることが．

川合 それじゃ回答の方を一通りやって，また皆さんのご意見を伺いたいと思います．私の近くからいきましょう．

大石 先ほど出ました，要求仕様定義技術と実現技術の統合ですが，要求仕様に関しては，すでに10年

前から研究が行われ，実際に支援システムがつくられています．しかし，これが実現段階と全然結びついていない．そのため，設計用のドキュメントができても，そこからの設計作業はまた人間でやらなければならず実際には，ほとんど使われていないのが現状だと思います．これから先のことについては実現段階における各種の蓄積が非常に重要で，これが切り札になると考えています．私自身もその方向で研究を進めています．さらに10年先を見通して，ソフトウェア危機を解決する切り札は何かといわれても，そういうものがポンと出てくるような気はしません．現実の1つ1つのソフトウェア技術の積み重ねが重要です．その意味でいうと，いままでのソフトウェアは，あまりにもワンショットでつくられ再利用されることがなかった．

ソフトウェアというのは，人間の知恵を出し切ってつくったものですから，これは，長く使わない手はない．ハードウェアが変わったから，いままでのソフトウェアが使えないなんていうのは，非常に困る．

まして，OSのバージョンが変わって，いままでのソフトウェアが使えないというのは，言語道断であるわけで私はユーザの立場からもソフトウェアの作成法を研究している立場からも，声を大きくしていいたい．

久保 たくさんのことを申し上げた中で，意見として言ったのはなにかフォーラムのようなものをつくりましょう，ということだけです．そのほかは，そういう意見もあるらしいということをいっただけですが，お答えをしてみます．

まず，消費者への配慮という問題です．私のご紹介した品質表示の制度にはそういうファクタが当然入っているはずですし，システム監査基準も，監査結果の公表ということになりますと，それは消費者への配慮が入っていることになると思います．高信頼システムの JIS 原案の中には明らかに，フォールトトレランスというものを，きちんと定着させていこうという思想が入っております．SC 7 の中でも，パソコンソフトウェアのパッケージングを優先的に取り上げています．ですから，いろいろな意味で，消費者のための活動は行われていると思います．

それが消費者からみて，ほんとうに十分かどうかというのは，いろんな意見を出し合って解決していくしかないのではなかろうかと思います．

つぎの英・数・国の問題は，いま一松先生が解説してくださった通りですが，一つだけ付け加えます．私が学校で勉強したころも国語は全然役に立たないことしかやらなかった．数学では答案を書かせることをやらされました．そこで作文の練習をしていたように思いますが，いまは，数学までが作文の練習機会を奪っているという感じをもっております．

そんなことを，社会に出てくる技術者を預かっている私たちが，小・中学校，高校の先生たちと話し合う機会があったかというと，1回もない．ですから，そんな話し合いも意味のあることではなかろうかということで，要約して，フォーラムをつくりませんかと提案したわけです．

通産省の施策の中に入っておりますインタオペラビリティの実現ということですが，ビジネスプロトコルの統一が考えられているし，企業間でシェアできるデータベースを築くことも考えられています．鈴木さんの紹介された SIGMA というのは，完全にインタオペラブルでなければ，役に立ちません．全国にネットワークを張って，ソフトウェアに従事する人が，みんなで使おうというわけですから，もともと基本条件として，インタオペラビリティは入ってると私は思っています．

鈴木 設備投資の話というのは，前々からいわれているのに，依然としてなってないじゃないかというお話なのですが，私どもの業界に関していいますと，私はやはり基本的には，使える，あるいは，投資に価する設備がなかったということではないかなと思っています．

ごく最近，UNIX を初めとするジュネリック OS のようなものが出てきて，初めて投資に価する設備が可能になったのではないかと思います．つまり，ソフトウェア業者の要求するあらゆる局面に対応できるような設備は，5年ぐらい昔には考えられなかったのです．最近ようやく投資のメリットのある本当に使える環境が自分たちで持てるようになってきたということではないかと思います．

川合 どうもありがとうございました．

私もひとこと教育のことをいいたい．私は，きょうここに大学の先生を呼ばなかった．クライシスの問題で，教育のことは非常に重要です．アメリカのある研究者が，ソフトウェアは，最も重要なことは技術移転であるという点で他の生産物と違うといっている．技術移転をする究極の目標は人を再教育することで，それがレクチャではだめなんだという．

レクチャというのは，すでにできあがった概念なり，知識なりを解説する，ことばに直して解説すると

いうふうにやるもんだから，どっちかというと，オーソライズされたことを扱うわけです．

そうでなくて，実際にわれわれが必要なのは，ごく生々しい，肌で感じたような感想であって，そういうものを心から心へ伝えるような，インフォーマルなコミュニケーションを進めることが伴わないとだめなんだと，こういうわけです．

実際，私も，ものを話しするための教育は，全然受けていない．学校の作文も，心情を表現することを求めているんであって，論理的に理詰で何か順々に自分の考えを述べるとか，人を説得するとか，そういうことはやれといわれたこともない．どちらかというと，人前でうるさいことをいうなと，こういう雰囲気なわけです．

それが非常によくない．アメリカの優秀な人は，何か聞くと，すぐ，「それに対しては重要なことは3つある」といういい方をする．そのときは，3つは何かということは考えないでいうらしいんだけど(笑)．

向こうはそういいながら，次々と考えていくわけなんですが，そういう話し方をどこで習得したのか聞くと，高校時代先生から，何度もいい直しをされていわされた．私はあの先生を非常に尊敬していますと，こういうことになる．

日本の高校教育では，もう一度いい直しなさい，なんていわないで「生意気なことをいうな」と威圧する．攻撃的な関係になるわけです．

そうでなくて，建設的な，仲のいい関係，愉快な関係にもっていくことを学ぶチャンスが，日本人には少ないと思う．唯一のチャンスは，恋愛するときで，学習して覚えていく．日本の場合には，社会に入ってから，会社の中でやるということになっている．ですから，ソフトウェア業なんかで派遣されていて，会社から離れているというようなときには，仲のよい対立関係を学習するチャンスがない．

これは，非常にゆゆしい問題で，いいかえれば，幼少時代に栄養が偏るというような，一種の病気をつくっているんじゃないかとさえ私は思います．

ソフトウェアのことを解決するためには，ソフトウェアそのもの，あるいはソフトウェアをいじくる道具と同時に，人間側ももう少し変わらないといけない．

玉木(ICOT の WG)　ソフトウェアをつくる上で，人間的な要素は大事だというのは納得しますし，教育が大事だというのもわかる．ライティングする能力が，人間一番知的な作業のベースにあるというのもわかる．いまのソフトウェア技術者の文章がひどいというのは認める．

ストラクチャされた文章が書けるようになり，だんだん数学的な思考もある程度上がってきたときに，われわれ情報処理研究者はソフトウェア生産者をサポートするようなツールだとか，エンバイロメントを提供できるのか．ある程度教育されても，われわれが提供できるツールを使うには，もっと利口にならないとだめだというんじゃ教育しきれないと思う．

アメリカ人がプレゼンテーションがうまい，ライティングもうまい，そういう教科テキストも，たくさん昔からあったというのなら，アメリカにはソフトウェア・クライシスがないのか，または，日本ほどひどくないのか．

ソフトウェア生産現場の人が，いまのアメリカ人のレベルに教育され，ライティング，プレゼンテーションがうまくなってきたときに，それでもまだだめなんですか，と聞きたい．

どこまで教育すれば，われわれ情報処理研究者や，ソフトウェア開発者が，それをサポートするソフトウェアツール，コンピュータ環境を提供できると思っていわれているのか，それをお聞きしたい．

甘田(筑波大学)　川合さん，久保さんにご質問したい．

私も教育者のはしくれでございますが，学校の教育というのは，社会に出て役に立つための素材を学生に与えるところだと思っております．ですから，ソフトウェアを書くのに具合のいい，たとえば国語を教えるといわれても，それはちょっとおかしいんじゃないか．

アメリカの学生は，川合さんがおっしゃったように，英語を話すことが非常にうまい，日本人は下手である．しかし，逆に，高校を卒業したレベルで数学の能力からというと，日本人のほうがずっと高いということも事実です．

そういうふうに，お互いに弱点もあり，長所もあるという者が，ある中でやっていくんですから，素材を与えて，社会に出て変わった環境なり，世の中の動きなりに適応できるものをつくるほうが，いい．

その辺に対してのご意見を承りたい．

西岡(横河北辰電機)　私は，ソフトウェアの仕事をやって，10年ぐらいたっている．生産性を上げたいという場合に，開発環境1つつくるのにも2，3年の年月がかかり，さらに2，3年たつと，古くなってしま

うという可能性が出てくる．10年ぐらいはもつような開発環境をつくることは一体可能なのか，大石さんにお聞きしたい．

古宮（IPA） 質問というよりも，意見をいわせていただきます．

汎用的な要求定義支援を中心としたエンジニアリングが必要だと思う．ソフトウェアの機能の客観的な表現が，まだない．それが再利用とか，生産性を上げるとかの障害になっている．

もう1つは，客観的な機能の表現方式が確立されれば，それをベースにしたソフトウェアの自動合成技術．それから，ソース公開がされていないため手直しをする機会を与えられないということもある．

それに対して，たとえば，通産省がやっているシグマ計画，のようなものに登録して，ソフトウェアの著作権が保障されるならば，もっと公開の機会に恵まれる．それが国家的なレベルで行われていけばいい．保守技術にも期待をかけたい．

会場より 私は，何が危機かという質問が，よくわからない．

伊藤さんの，ユーザとして満足なシステムができない，やろうと思えば高くつく，という話しは非常によくわかります．産業界が，コンピュータ事業をこういう形で進めているということ自体に責任がある．

鈴木さんのように，ギャップはなく，いいところでバランスが取れているというお話はその通りです．ソフトウェア業界は，ギャップがあればあるほど，儲かるわけでございますから，これはソフトウェアをやっていらっしゃる方については，大変だ，大変だといいながら，やはり満足な状態であろうと思います．

ステップ数が多いプログラムは下手なんだから金を払うのがいけない．

きょう，もしハードウェア担当の人がいたとすれば，「ハードウェアは，よいソフトウェアのサポートがないから売れないんだと，それがソフトウェア・クライシスである」というかもしれない．

ソフトウェアの効率が悪いが，学校から出た人をそのまま採用して，ポカンとユーザなりメーカに投げ込んでいるので，効率の悪さを教育にもっていくのは，筋違いじゃないかと思います．

全体としては，システムエンジニアが，怠慢である．詰まるとすると，この辺からソフトウェアのために何か抜本的に考えていかなければならない．

発展，発達それぞれしておりますので，それを全体的にみるのは非常に困難ですけれども，自由討議でもなんでもされて，抜本的なものの見方をされるがいい．

質問です．危機に対してどこに重点があるか，それを司会の川合さんに質問したい．

川合 もう10分ぐらいでやめたいので伊藤さんから順々に，これだけはいっておきたいこと，あるいは10年なり20年なり，先のことまで考えると，これだけが最も重要だと思うことを一言ずつお話願いたい．

伊藤 冒頭問題提起を主にお話申し上げましたが，ソフトウェアへの対応が困難になりつつあるのは，ユーザのみでなく，メーカもソフトウェア産業も同様ではないかと思います．また，対応の仕方についても，それぞれの業界が，個別に対応するだけでは解決は困難ではなかろうかと存じます．したがってメーカやソフトウェア業界の方々と私どもユーザはさらに一層の協業が重要ではないかと思います．

ユーザにとりましても，たとえばソフトウェア開発維持の業務管理体制が十分近代化されているのか，あるいは必要なシステム技術基盤が整備ができているのかなど反省すべき点が数多く存在しており，標準化・手順化などの観点からも指摘すべき問題が残されております．

一方，現在主流を占めているコンピュータのハード・ソフトウェア開発維持の面からみて，改善すべき点や一層の開発をお願いすべき問題もまた多いと思います．また，ソフトウェア産業の方々に開発業務の大きな部分を期待せざるを得ない現在，ユーザの期待する技術レベル，特に設計レベルでのより一層の強化と，製作レベルでの工業的方法へのアプローチが必要と思われます．

学界の方々も含め，メーカ，ソフトウェア産業そして私どもユーザのより一層の緊密な協業が望まれるところであります．

鈴木 私たち自身の業界も含めまして，要員の派遣という問題，これをなんとかやめたい．われわれ自身もやめなければいけませんし，ぜひお願いしたいのは，受ける側もやめていただきたい．

これは，いろいろなことに影響を与えております．

先ほど来あります技術の問題，あるいは，技術者のレベルアップの問題，あるいは環境の整備の問題もそうです．大体その会社にいなければ，環境の整備などほとんどできないのに等しいのです．管理の問題もあります．いろいろな問題の1つの源は，ここにあるの

ではないかと思います.

久保 危機があるかどうかには，あまり関心がありません．そう思ったほうが，生産的な活動ができるんではないだろうかという考え方です.

その1つとして，教育の見直しという声も出てくるでしょうし，ソフトウェアの技術者たちが怠慢ではないかという意見も出てくるだろうと思います.

ですから，ギャップの犯人を教育にもっていこうという発想ではありません．われわれにも反省すべき点があって，その1つの例として，いままでの体験をうまく工学化することを怠っているのではないか，ということを申し上げました．きょうは多分学界の方が多数いらっしゃるかと思いますのでその点について，もう1度提案のかたちで繰り返させていただきます.

いいたいのは，「ソフトウェア工学に対して経験工学としてアプローチするのはいかがでしょうか」ということですが，もうちょっと具体的に申し上げます.

いろいろな開発の仕事が行われているわけですが，それをいわば表現するファクトデータの取り方・集め方・ためこみ方について，何か基準をつくって，それをソフトウェア工学の1つのインフラストチャにしておくことはやってよいと思われます.

そういったファクトデータをためこんだ博物館を持つことも，ソフトウェア工学のインフラストラクチャになり得るはずです.

これらのことを学界の人たちが積極的に推進するためには，研究者を産業界の現場の中に，1年とか2年とか派遣されるのがいいのではないでしょうか.

これが私の提案です.

大石 経験を工学的にするのには私も賛成です．それでこそ，ソフトウェアあるいは設計の再利用が，現実的なものになる.

ソフトウェアの作成環境についてですが，紙テープという時代は，さすが終わりをつげましたので，ちゃんとした物が使える時代になってきました．そういう意味では，ワークステーションもどんどん普及していくだろうと思います．実際にはお金の問題がありますので，現実的なところとしては，パーソナルコンピュータをうまく使ってメインフレームとの間を埋めていくのが1つの手と考えております.

川合 フロアからたくさん意見をいただきまして，どうもありがとうございました.

問題がいくつも残っておりまして，ここで私がスパスパ答えると格好いいんですが，とてもむずかしくてそういうわけにはいきませんので，勘弁していただきます(笑).

最後に，私自身が一番大事だと思っていることをひとこと申し上げておきます.

アメリカ人並みにしゃべればクライシスは解決するのではなくて，なるべく論理的にかいつまんで話しをすることに，お互いに修練することが，ソフトウェアを上手につくったり，うまく使ったりすることと関係があるということを，よく認識しておかなければならないのです．人間は，脳の前のほうで物を考えているんですが，駆動しているのはかなり後ろだということがわかっています．快感神経が脳全体に広く広がっている．つまり愉快にならないと，頭も働かないことは，経験上も生理学的にもわかってきている.

こんなことは，素直に受け入れて，コンピュータで知能を扱おうとしたら情欲の駆動はどういうふうにして実現したらいいかということに発展させて，考えていかなければならない.

前脳の神経はさやがついてきますが，20才，30才ぐらいまではどんどん増えていきます．皆さんもこれからますます勉強すれば，もっと頭がよくなると，こうお考えになって，お互いに励みたい.

私のところで．日本語の辞書をつくる仕事に国語学者が参加していますが，彼らは，「そのことばの意味は，1年考えます」と軽くいう．それではいつまでも完成しないことになる．私達の見解は，そのぐらい，立場や経験が違うと差があるわけだから，お互いの立場の相違を認め合いながら共存していこう．そこでまた新しい仕事をみつけていくというふうにお互いに働いていけば，だんだん建設的な関係が強くなって，ソフトウェア・クライシスは解決の道へ進んでいくと思います.

要するに，たくさん問題があって，それぞれ細かいところで工夫して解決の努力をしなければならない.

一見漸進的な方法が，じつは一番革新的な方法なのです.

どうも長い間，ありがとうございました．(拍手)

総合司会 先ほど，このソフトウェアに携わる者の弱い連携を深めることが大事であると川合さんがいわれましたけれども，本日は5時半から懇親会を予定しておりますのでご案内します．どうもありがとうございました．(拍手)