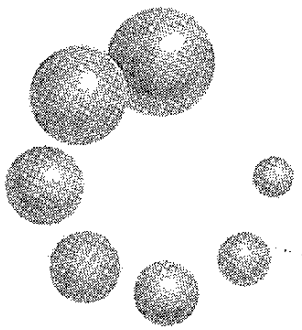
'76情報化国際講演・討論会

新世代を迎えた情報処理技術

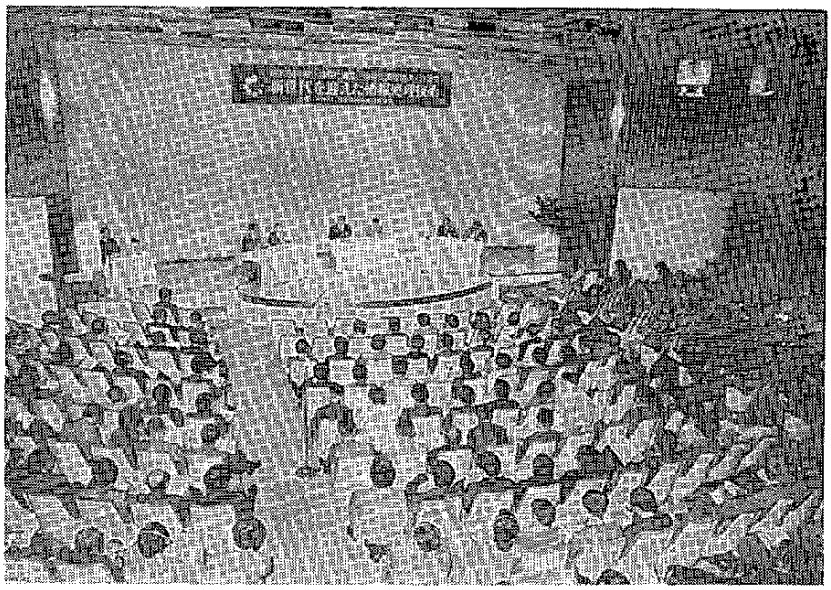
会 議 録



JIPDEC

財団法人　日本情報処理開発協会

この資料は，日本自転車振興会から競輪収益の一部である機械工業振興資金の補助を受けて昭和５１年度に実施した「情報化国際講演・討論会」の成果をとりまとめたものであります。



国際講演・討論会

新世代を迎えた情報処理技術

会　議　録

昭和51年度情報化週間

情報化国際講演・討論会

新世代を迎えた情報処理技術

日時　昭和51年9月30日(木)・10月1日(金)

会場　経団連ホール

主催　日本情報処理開発協会

後援　通商産業省
経済団体連合会
経済同友会
日本経営者団体連盟
日本商工会議所

協賛　EDPユーザー団体連合会
関西情報センター
情報処理振興事業協会
ソフトウエア産業振興協会
日本情報センター協会
日本電子工業振興協会
日本電信電話公社
日本公認会計士協会
日本内部監査協会

会議録目次

開会のあいさつ

植　村　甲午郎[1)]

本日は，御多忙の中を通産省御当局初め多数の方々の御出席をいただきまして，ここに昭和５１年度の情報化週間の行事として，第５回情報化国際講演・討論会を開催できますことは，主催者といたしましてまことに感謝にたえない次第でございます。また，ソルトマン博士，ウェバー博士並びにダイクストラ博士におかれては，本討論会に御参加いただきまして，遠路はるばる来日いただき，また国内の学識経験者の方々にも多数御参加を得まして，まことにありがとうございます。さらに，この討論会の開催にあたって，通商産業省初め多数の関係団体の御後援，御協賛を賜りましたことをあつく御礼を申し上げます。

私どもは過去４カ年にわたって，情報化週間の行事といたしまして，そのときどきにマッチいたしましたテーマを取り上げ，国際講演・討論会を開催してまいりましたが，おかげをもちまして年々多くの方々多数御参加を得まして多大の成果をおさめ，情報化週間中における主要な行事の一つとして定着してまいった次第でありますが，これはまことに喜ばしい限りでございます。

本年は，過去４カ年の成果を踏まえて，また本年４月に情報関係３団体が大同団結いたしまして，財団法人日本情報処理開発協会として新発足以来最初の情報化国際講演・討論会でございますので，「新世代を迎えた情報処理技術」というテーマのもとに，システム監査，分散処理，構造化プログラミングの諸問題を取り上げることといたしました。

皆様御承知のとおり，わが国におけるコンピュータの設置台数はすでに３万台を超えまして，米国に次いで世界第２位に位置していますとともに，その利用の面におきましても，当初の民間企業等の産業面から，次いで行政面へ，さらに最近においては交通，教育，環境，医療等の社会生活面へと着実な広がりを見せておりまして，いまやコンピュータはわれわれ国民の日常生活と切り離しては考えられないという時代になってまいりました。しかも今後わが国は海外諸国との友好と協調のもとに，高度の福祉，文化国家を目ざす以上，政治，経済，文化等のあらゆる分野においてコンピュータを中心とする情報化の進展がさらに期待されるわけでございます。

しかし，他方，わが国の経済は，過去の高成長時代から安定成長時代へと大きな転換を見せております。したがって，この際コンピュータの利用につきましても，ハードウェアソフトウェアの両面にわたる技術の著しい進歩を踏まえて，新たな角度からの点検と発想の転換とが必要ではなかろうかと考える次第でございます。このような観点から，このたびの「新世代を迎えた情報処理技術」をテーマとして，システム監査，分散処理及び構造

---

1)　日本情報処理開発協会会長

化プログラミングの諸問題を取り上げた次第でございます。講師の方々並びに本日御参集の皆様方がこれらの問題につきまして真摯率直な討論を行われて有益な成果を上げられますことを期待いたしかつ確信いたす次第でございます。

以上はなはだ簡単でございますが，私の開会のごあいさつといたします。

祝　　辞

通商産業大臣　　河　本　敏　夫

情報化国際講演・討論会の開催にあたり，一言ごあいさつ申し上げます。

わが国経済は，戦後急速な発展を遂げましたが，近年に至り，わが国は環境の保護，資源エネルギーの安定確保とその有効利用等の深刻な問題に直面しております。わが国はこうした情勢に適応し，経済社会の発展を図るためには，省資源，高付加価値型産業を育成し，産業構造の高度化を図ることが必要でありますが，特にその中核としての情報化の推進と情報産業の振興は強く望まれるところであります。

他方，国民のニーズがますます多様化し，高度化する現在，豊かな，活力ある福祉社会を築くためには，産業活動面のみならず，医療，教育，交通といった社会面，生活面への情報化の拡大が必要であると思われます。

こうした中にあって，情報産業は本年４月をもって完全自由化に踏み切り，技術力，資金力等の面で圧倒的優位に立つ外国企業といわばはだかの競争をしいられることとなったわけでありますが，わが国情報産業がこの競争にうちかって，自立と安定を確保することは，今後のわが国の情報化及び情報産業の健全な発展のかぎとなるものであります。通商産業省といたしましても，このような認識に立って，医療情報システム，生活映像情報システム等の社会システムの開発，次世代電子計算機用の超ＬＳＩ開発，ソフトウェア生産技術開発などを推進しております。こうしたときに，情報化に関する正しい理解と認識を深めるため，「新世代を迎えた情報処理技術」と題して本講演会が開催され，世界的視野に立ってコンピュータの利用に関する新たな着想と技術研究について，講演及び討論が行われますことは，まことに時宜を得たものであると思います。本講演・討論会において示唆に富む数々の貴重な意見が提出されることを期待いたしまして，私のあいさつといたします。

基調講演

基調講演

稲葉秀三[1)]

ことしの国際シンポジウムのテーマは，「新世代を迎えた情報処理技術」でございまして，私のようなエコノミストが果たしてそういうことにつきまして十分な問題提起ができるかどうかということにつきまして内心じくじたるものがございますけれども，私なりにひとつ問題を出させていただきまして，皆さま方の御注意を喚起をさせていただきたいと思います。

最近，日本経済の将来の見通しや産業構造のあり方，また産業の中でも私たちに関係の深い情報産業の将来像，また国際的な情報網の結成などをめぐりまして，重要な問題がいろいろとこの日本で登場をしているのでございます。私はここで経済と情報化に関しまして，私どもにとりまして共通をした，また今後も継続して検討し，推進していかねばならない2，3の問題を指摘をさせていただきたいと思います。そして情報ネットワークの創造に関する各方面のコンセンサスの形成と，これに伴うもろもろの体制の整備という点につきまして関係者の御注意を喚起をさせていただきたいと思うのでございます。

さて，話題の第1といたしまして，これからの日本経済の環境変化と情報産業のあり方という問題を取り上げてみたいと思います。その手がかりとして次の三つのことを私は申し上げたいのであります。

その一つとして，この5月に政府から公表されました昭和50年代前期経済計画というものがございます。また同じくことしの8月に発表されました通産省の産業構造の長期ビジョンというものがございます。その二つの手がかりといたしましては，この春に改定されました政府の電子計算機利用高度化計画というものがございます。さらに7月に発表されました産業構造審議会の昭和60年度におきまするわが国の情報化及び情報産業の計量予測というものがございます。そして同じくその三つといたしましては，いわゆる情報産業自由化の最終段階が過ぎまして，ことしは新たに日本に上陸をいたしました，または上陸しようとしている外国資本によるところの情報処理サービス業の問題が登場してきております。そしてヨーロッパでは，このような問題が着実に進んでいるように思われますが，これらの国際的な情報ネットワークの問題に対してどのように対処をしていかねばならないかと，このような問題が出てきていると思うのであります。この三つの手がかりをもとにして，私は次のような問題提示をさせていただきたいと思います。

私は経済審議会の委員であり，またその5カ年計画を取りまとめました産業分科会の会長といたしまして，政府のこの経済計画の作成にお手伝いをしてまいったのでございます

1) 経済評論家

が，５０年代前期経済計画というのはすでに皆様方も御承知のように今後おおむね６％前後のいわゆる安定経済成長路線というものを骨子として組み立てられているものでございます。この計画は，環境の変化といたしまして資源問題や世界各国経済の相互依存関係の進展などを挙げるとともに，社会的消費の充実とそれに対する国民意識の変化といったような点を考慮をしているのでございます。ことに重要なものといたしましては，この計画が経済成長率の低下に伴う雇用問題，物価問題，財政問題と並びまして，経済の活力と申しますか，企業活動や企業体質の強化をいかに図るべきかということに対しまして問いを投げかけている，このように考える次第でございます。

先ほど申し上げました手がかりの一の中のもう一つの問題として，産業構造審議会の方の産業構造の長期ビジョンでは，わが国の産業が知識集約化，付加価値増大の目標に向かって進まなければならないということを，これは従来から主張されていることでございますけれども，今回のビジョン報告では，５０年代前期経済計画の趣旨というものを受け継ぎまして，わが国の産業構造がよりそのような方向に変化をしていかねばならない要因といたしまして，経済成長率の鈍化とかコスト構造の変化でありますとか国際経済環境の変化などを指摘をしているのでございます。しかし私は，これにつけ加えまして，産業，企業を通ずる情報処理機能やシステムの高度化というものが産業構造そのものを知識集約化の目標に向かって変化させるものである。このような点を強調したい次第でございます。そしてまさにこの高度情報化戦略こそが，５０年代前期経済計画の投げかけているところの経済の活力とか企業活動でありますとか体質の変化につながるものであるということを皆様方とともに確認をしたいと思っているのでございます。

それではわが国の情報処理システムというものが，形の上の規模としてどのようにこれまで展開をしてきたか，これから展開をしていくのであるか，この問いに答えているのが先ほど２番目の手がかりとして申しました電子計算機利用高度化計画と産構審の６０年度におきまする情報化及び情報産業の計量予測であると言えましょう。これらの報告によりますと，先ほど植村会長さんがお話しになったのでございますけれども，今１９７６年３月にわが国では３６,４００台，約２兆３,０００億円の汎用コンピュータが設置をされていると推定されているのでございます。これが５年後の昭和５５年，１９８０年度末には７５,０００台と５兆５,０００億円，また６０年度末，１９８５年度末には１０万７,０００台，７兆５,０００億円に達するであろうということが計量予測をされている次第でございます。実はこのような予測の値というものは経済の他の分野ではほとんど考えられないような高い成長率というものを意味しているものだということを私は申し上げたいのでございます。コンピュータ設置金額の５１年から５５年に至る平均の伸び率は年率１９％でありました。これは一種の名目的な金額によるインデクスであると言えましょう。５０年度前期経済計画の同じ期間におきまするＧＮＰの成長率は実質６％，名目成長率に換算をいたしまして13.3％ということになっております。そこで５０年から５５年，つまり１９７５年から１９８０年に至りまする名目経済成長率に対しまするコンピュータの設置金額伸び率の弾性値というものを計算をいたしますと，1.43大略1.5と，こういうことになる次第でございます。そのことは名目ＧＮＰが１％ふえればコンピュータの設置金額は１.５％ふえる。このような形になるということでございます。このように急速度に増加をしていくといたしますと，コンピュータの実質実働規模は１９８０年度で５兆円を超えるということになります。また1985

年つまり昭和６０年では７兆円を超える，このように予想されるのでございます。

ところで，このような巨大な情報が処理システムとして活動していく，このようなことになる場合において，一体どのような形や方法で利用されるのであろうか，ここがやはり大きな問題だと言わねばならないと思います。後で若干この問題に触れてみたいと思いますけれども，最初に申し上げた第３番目の手がかりというものを皆様方にお考え願いたいと思います。

ことし情報産業の完全自由化を迎えたというのがわが日本でございます。現在の既存の合弁企業に続きまして，新しい外資系のオンライン情報処理サービス業者の進出というものがいま話題になっておりますが，これも自由化に基づいて出てきたものと言えるのでございましょう。そしてその意味するところは，国際的な情報処理ネットワークの中にわが国も組み込まれていくであろう。いな組み込まれていかなければならざるを得なくなってきた，こういうことではないかと思う次第でございます。私の思いますのに，このような現象を狭いナショナリズムの目で受け取るということは必ずしも当を得ていないと思うのでございます。と同時に，国際ネットワークが好むと好まざるとにかかわらず形成されていくとすれば，その中でわが国の主体的立場というものをどのようにしていくのか，いかねばならないのかということこそが必要であると思う次第でございます。

一方，わが国の最近の状況を見ておりますと，ヨーロッパを中心とする銀行間の情報ネットワークでございますいわゆるSWIFTにわが国の銀行が加入をされるという動きが伝えられております。また国際ネットワークとしての気象情報網にわが国はすでに加盟をしているのでございます。科学技術情報システムといたしましては，ヨーロッパにおいてユーロネット計画がスタートをしていると伝えられておりますが，アメリカのALOHAシステムと日本の大学との連絡も計画され，推進されているという次第でございます。証券業界では，海外データベースとの提携やオフラインではございますけれども，国際的な中央銀行同士の合同データベースの設置，このような動きも注目されているところでございます。このような多種多様な国際情報ネットワークの形成の話題は，情報を有効に，高度に利用する技術方式，組織，制度等について改めて大きな基本的な問題を提起をしているかのように私には思われるのでございます。

さて，私はこれまで最近の話題を借りながら，わが国の経済のゆくえと産業構造の変化を促すものとしての高度情報化戦略について触れてきた次第でございます。また予想される膨大なコンピュータの投資と国際ネットワークの発達にも目を注いでいかねばならない，このように思う次第でございますけれども，余りこの方面の専門家と言われるような人間ではございません。しかし私がここで皆さんに強調したいのは，このような巨大な情報処理メカニズム自体がいまや大きな変化，大きな転換を迎えようとしているのではなかろうかということでございます。これを産業構造の変化になぞらえまして，情報化構造の変化と仮に呼ぶといたしますと，その変化をもたらす大きな誘因を考えていかねばならない。また，これにいかに対処すべきかということをそれこそ多角的に検討を加えていかねばならない。そして今後どのようにわれわれは進んでいかねばならないかということを，それこそ政府と民間の双方でやっていく，真剣に推進をしていくということをやっていかねばならないときが来たのではなかろうかと考える次第でございます。それは社会や産業や企業のニードがどのように変わっていくのか，これに対して情報産業はハードとソフトの面でどう対応していくのか，情報ネットワークを今後１０年，２０年の先を考えて日本でど

のように整備をしていくのか，そしてその際官と民の推進方策をどのように分担し合うのか，またこれらと関連をいたしまして標準化や言語の統一やソフト技術の整備というものをどのように進めていくのか，海外とのネットワークその他の関係をどうしていくのか，人員の養成，訓練，他の部門からの再配置をどうしていくのか，またさきにも申し上げましたように日本で一番早いテンポで，他の産業部門よりも大きく情報産業というものを伸ばしていくとした場合に，どうしても国と民間で膨大な資金や投資というものをその方面に集中をしていかねばなりませんが，これをどのように調達していくのか，さらには有効に投資を活用しようとした場合，どういう方策を推進すべきか，具体的な措置をとるべきか，これらについてもっともっと画期的な推進措置をとらねばならない。それなしにはさらに大きなたくましい前進ができないのではないか，このように考える次第でございます。ここではこれらにつきまして私の見解を申し上げるという時間的な余裕はございません。また私の見解そのものもやはり問題だ，このように思っております。

そこで，ただ単にこういう問題を指摘をするということにとどめさせていただきたいと思う次第でございますが，ただ，ここで私がエコノミストとして強調しておきたいのは，情報産業というものはシステム産業中のシステム産業であるということであります。遺憾ながら過去の歩んできた道は，大きな成果を上げたということは否定しませんけれども，余りシステム産業にふさわしい発展方法ではなかったのではないか，このように考える次第でございます。今後の大きな発展のためには，ここで一度ゆっくりと立ちどまって周囲をよく見渡して，そして再出発や再軌道を歩んでいくというための再検討を政府と民間の双方でしていく必要があるのではなかろうか，このように私は個人としては申し上げたい次第でございます。

そこで，その中の一つに投資の問題があると私は思っております。私はこれまた通産省の産構審の産業資金部会の委員の１人でございますが，この情報化のための産業資金を通産省はもっと大きく検討をしていかねばならぬし，ぜひともしていただきたいと思う次第でございます。と申しますのは，その一つといたしまして，過去におきましてこの部面におきまして二重，三重の投資のダブリがあった，このように私は思っている次第でございます。もちろんすべての盾には両面がございます。このような二重投資，三重投資があったために，わが国は何と申しましても自前で情報化を推進してくる，情報産業を大きく伸ばしていくということができた，こういう見方も当然成り立つと思っております。また日本にはバイタリティというものがございますので，バイタリティがあるためにこのようなことが起こり，そして伸びてきたのだ，このような見解も十分成立すると思いますし，さらには技術面からの事情というものもあったと思う次第でございます。ですけれども，これからの大きな躍進というものを考えますと，いまのままであってよいとは言えないのではなかろうか，このように考える次第であります。最近そちらにつきましてまた新しい方向に向かっての改善の体制というものが，徐々ではございますけれども，政府と民間の中で進んでいるということははなはだ喜ばしいことだと思っておりますが，もう一つユーザーサイドにおきまして，産業として，また個々の企業として情報関連リソースの有効利用につきまして今後厳しい方策が必要ではなかろうかと考える次第でございます。初めにも申し上げましたように経済成長率の低下を前提としますときに，経済の活力，企業活動，体質の強化のためには，高度な情報化の推進こそがそのきめ手であると思っております。情報化を志向する投資はさらに拡大されねば

ならないのでございますが，マクロ的にもミクロ的にもむだな重複した投資を許容する，このような経済環境に私たちは直面をしていく可能性はきわめて少ない。しかもここで大きな量的，質的躍進を図っていかねばならないといたしますと，やはりこのような点が重要ではなかろうかと思う次第でございます。

次に，私は国際間の問題にちょっと触れてみたいと思います。国際的にはすでにリソース・シェアリングまたはユーティリティ・シェアリングという形での多国間情報システムというものが出現しつつあります。また今後ももっと出現してくるであろうということが予想される次第でございます。このような要請・現実から導かれますところの情報化構造の変化の一つの重要な形は，多くの情報ネットワークの形成が必要であるということでありましょう。情報ネットワークという場合，そこには情報の伝達，交換，処理あるいは情報提供，情報検索といった数多くのシステムが考えられるわけでございます。またネットワークというものは，コンピュータと各種のノード，ターミナルそして通信回線が結合したものを指すということは言うまでもないことでございます。さらに，ネットワークを流れる情報というものはメッセージ，データ，画像，映像を含むと考えられるのでございます。しかしこのような情報ネットワークの形成を必要とする認識と自覚というものは比較的先進的な一部の企業に限られているように思われます。まだまだ一般のものとして結実してこない，このように思われるのであります。確かにネットワークを創造することは，技術的な面から見ましても，見かけのコストの面から見ましても，制度的な面から見ましても困難な問題が多いのでありますが，今後特に広範な中小企業の情報化を進めるに際し，個々の事業体がそれぞれ自己完結的なシステムを個々に保有する場合，その経済性には明らかに疑問がある，このように思う次第でございます。産業ごとのアップストリームからダウンストリームをつなぐネットワークは比較的形成しやすいのでございますけれども，現在ではまだ困難な産業横断的ネットワーク，地域的ネットワーク，社会機能に応じたネットワークなどが，多くの参加者により形成されるということができておりませんが，これが可能となるということが必要だと思っております。そして個々の参加者の負担コストを軽減をするということが必要であり，またそのような方向に進んでいかねばならないと思う次第でございます。このような多様な共同情報システムの必要性についての社会各層の認識と実現への共同行動が開始されますということを，またされんということをここに強く私は期待したいのでございます。同時に，このコンセンサスを高度な情報化戦略の大きな炎とするために，国としての十分な支援が必要である，このように考える次第でございます。これらにつきまして肝要と思われます多くの施策のうちで，私は次の二つのものだけを取り出してみたいと思い，そしてそれにつきまして各方面の方の御検討をお願いしたいと思う次第でございます。

その一つは，自由に共同情報システムが形成されることのできるような制度面の前進ということでございます。現在わが国におきましては，ネットワーク形成に関しある程度の制約がございます。その制約の根拠，理由というものは十分理解できるところでございますけれども，すでに国際情報ネットワークとの接触が現実のものとなっているときに，目を新しく開きまして，開放された体制というものを私たちは考えねばならない。わが国独自のネットワーク確立を図らなければならないとともに，国際的にわが国の主体性を失うおそれなしにこういう点に向かってどのように前進をしていくのか，このようなことをぜひとも考え，推進をしていかねばならないと思う次第でございます。この点につきまして，

私は建設的な論議というものがもっと活発に各界，各層で起こってくるということを希望し，期待する次第でございます。

その二つは，資金的なバックアップというものが重点的になされねばならないということであります。具体的には共同情報ネットワーク樹立の障壁の一つであるネットワーク関連ソフトウェアの開発にもっと十分な資金が供与されることが望ましいと思う次第でございます。これも最近の話題の一つでございますけれども，西ドイツ政府の発表によりますと，西ドイツでは1976年から1979年にかけまして総額15億7,500万マルク，日本のお金に換算をいたしますと1,900億円の予算で情報処理システムの助成を図るということでございます。しかもハードウェアとソフトウエアに同額を振りあてるのだということが新しい計画になっているということでございます。さて，これに対しましてわが国では，ハードウェアの育成にはこれまで相当な金額の資金が投ぜられてきましたけれども，公的なソフトウェア単独の開発資金はまだまだ不十分でございまして，年間50億円にも満たない，こういったような規模でございます。

開発テーマといたしましては，電算機共同利用高度計画の中でネットワークに関連するものも十分盛り上げられてはおりますけれども，さらに実行にあたりまして重点的なネットワーク，ソフトウェア優先を考慮していただきたいと思う次第でございます。いずれにいたしましても，比喩的に申せば，私どもはかつて大艦巨砲主義の国であった。そのような時代からタスクフォース優位の時代への変遷を痛切な思いで体験をした国民でございました。情報化戦略が浸透いたしまして，先見の明と速度に富んだ実行力でシステム手法がみずからの手で生かされていくということこそが，またそういう気魂に立って今後展開をしていくということが私どもに与えられました問題を今後有効に推進をしていくのではなかろうか。またそのような過去を再検討しながら，新しい飛躍と産業構造の転換ということの中でいま一度情報産業を新しく見直してみる，推進をしてみる，こういうことが必要であるということを指摘いたしまして，私に与えられました時間と任務を終わりたいと思う次第でございます。御清聴ありがとうございました。

セッションⅠ　　システム監査と管理者の責任

新しい分野として台頭しつつあるコンピュータ監査

要　旨

最近，コンピュータ・システム監査が大きくクローズ・アップされてきている。その根底には，コンピュータを有効に活用するため，いかにマネージするか，過失・事故・不正をいかにして防止するか，コンピュータ会計の監査，プライバシーの保護をいかにするか，等種々の問題がある。ここでは，これらの諸問題を有機的に把握し解決するために，米国の実情を参考としながら，わが国企業における，システム監査体制のあり方，実施の方法等について，実務サイドから具体的な方法論を追求するものである。

新しい分野として台頭しつつあるコンピュータ監査

ロイ・ソルトマン[1)]

コンピュータ監査というのは新しい言葉でありまして，コンピュータ・システムの効率的かつ効果的な運営を確認するためのものです。コンピュータ監査という概念の確立とその方法の開発は連邦標準局のコンピュータ科学技術研究所の大きな関心事であります。

コンピュータ監査はきわめて重要な活動になってきています。これから私が話を進めていくにあたり，コンピュータ監査と通常の財務，会計監査とを区別して申し上げたいと思います。この区別を理解することはきわめて重要です。そしてこの区別については，順次明確にしていきたいと思います。

民間であれ，政府であれ，製造業であれ，サービス業であれ，すべての組織はコンピュータを利用しており，そしてその円滑な活動のためにコンピュータに依存しています。コンピュータは技術革新によってだんだんパワフルになってきましたし，そのためにコンピュータに与えられる機能は拡大し，深くなってきています。多くの組織では，その活動や財務面での最も基本的なデータはコンピュータ・システムの中に入っており，それによって業務は，処理されています。これらの組織におきましてはコンピュータ・システムのアウトプットをその組織活動を適正に維持していくための意思決定に利用しています。というのは，意思決定のための基本的なデータのほとんどがコンピュータからしか得られない状態になっているからです。さらにコンピュータ・システムが組織の意思決定においてますます重要になってくるにつれて，その組織はコンピュータ処理に合うようにマネージメント・プロシーデュアを合わせなければなりません。具体的なデータ・フォマットやメディアに対するコンピュータの要件，またデータ・エントリーの方法，高速データ通信の要件をまず考慮して，オフィスの機能とそれから個人の役割を決めていかなければなりません。

こういった状況のもとでコンピュータ・システムが意図したとおり機能できない場合，また逆に意図しなかった機能をやってしまうことは，組織にとって壊滅的な打撃を与え，コストもきわめて高くなり，また個人に対しても害を与えることがあります。しかし前向きの組織であるならば，コンピュータ・システムを設置することによって，組織の効率が上がり，制度的また社会的な逆作用がなければシステムの設置をためらうことはありません。コンピュータの設置を決定する場合には，その遂行能力を明確にしておくことが必要です。これを確認するための手続をコンピュータ監査と言うことができると思います。

そもそもコンピュータ監査は学際的な概念であります。組織的な活動としての監査はもともと会計監査から始まっており，内部監査，

（Roy G・Saltman）
1) 米国商務省標準局コンピュータ科学技術研究所 プログラム マネージャー

外部監査両方から成っております。内部監査はコンピュータ監査に最も関係のあるもので，経営活動を再検討するための組織内の独立した評価活動であるとされています。ほかのコントロールを測定し，その効果を評価する機能がマネジリアン・コントロールです。内部監査人は，経営上役に立つすべてのビジネス活動に関心を持っています。その活動としては，まず第一にオペレーティング・コントロールの評価と検討，二番目は，すでに確立された政策，計画，手続等がどの程度受け入れられるものであるかの評価，三番目は会社の資産があらゆる種類の損失から保護されているか，四番目は，組織内で作成したマネージメント・データの信頼性を確認すること，5番目は与えられた責任を遂行する質を評価することであります。

監査は，事業活動の高度化につれてますます複雑になってきています。小企業または初期の段階では，監査人はたとえば手で株や現金を数えることができたわけです。つまり資産を直接取り扱っておりました。しかしながら業務が増大し，複雑化するにつれて，その金融資産を物理的に保有するのではなくて，象徴的に保有するようになってきました。たとえば従業員に対して現金ではなく，小切手で支払いをしたり，信用状，預金入金票，クレジット引出券等が使われ始めました。こうして監査人は金融データを書いた紙きれを取り扱うようになり，資産を有形な形ではなく，記録として扱うことになったわけです。このような場合には，記録を処理する方法が監査人にとって基本的に重要なポイントになります。

コンピュータを利用しての情報処理の自動化は，監査人に新しい問題を提起しました。というのは多くの紙のレコードはなくなってコンピュータ処理によるインプットとアウトプットになったわけです。多くのデータはディスクやテープ等磁気媒体の中に入っております。以前事務員によっておこなわれていた内部統制は，現在ソフトウェアすなわちコンピュータ・プログラムによってなされています。監査人はただその業務の記述を検討するだけでは，オペレーションを検討できません。同時にコンピュータ・システムの論理構造とパーフォマンスをチェックしなければなりません。このためには全く新しい技術が必要になってくるわけでありまして，会計監査の訓練を受けただけでは十分に対応することはできなくなっています。

またコンピュータ・プログラムによっては，監査人を助けコンピュータ・オペレーションを分析し，コンピュータ・ベース・ファイルを検討できるようになっています。これをオーディット・パッケージと呼んでいます。ファイルから情報をコピーし，データをまとめ，それからファイルの比較をし，リストをつくるわけです。その目的は，コンピュータを利用して監査人を助け，監査の機能を自動化することであります。

しかしたとえ有用であっても，これらのプログラム・パッケージにはその限界があります。まず第1に，同じコンピュータを使ってデータをつくり，そして監査するわけですから，監査対象になるデータにエラーがある場合にはそのシステムに問題があるわけで，同じシステムに入っているオーディット・パッケージも同じエラーを持っており，それをなかなか識別することができないわけです。

二番目に，こういうパッケージは会計・データにはいいが，会計データ以外にはうまくいきません。今日ではだんだんと会計データ以外のデータも組織の資産評価に含めなければならなくなり，われわれは会計データ以外のデータをもっと重要視しなければなりません。

三番目に，これらのパッケージは静態だけ，つまりある特定の時点におけるデータだけを考えているわけで，過去のオペレーションに

ついてはまとめて入っています。ですから，もっとダイナミックな条件を取り扱うということが必要になり，データ・フローの効率，速度，また大量のデータについても処理し，適切で，正確なアウトプットを出したい訳です。正しいダイナミックなオペレーションという問題は，リアルタイムのノンファイナンシャル・システムの特徴であり，こういうシステムについてこそ現在のオーディット・パッケージは関心を持っています。

四番目に，これらのオーディット・パッケージはファイルをつくるコンピュータ・プログラムの論理構造を検討することはできず，ファイルの中にあるデータだけに限られます。

さらに内部監査人にとって複雑なのは，象徴的なトランザクションについて目に見えない処理が行われることです。コンピュータをマルチプログラミング化することが可能です。

これは幾つかの種類の業務が同時に同じコンピュータを利用することです。マルチプログラミングを利用すれば，時間を分割することによって，すなわちタイムシェアリングによって，コンピュータ機器を有効に使用することができます。つまりコンピュータのＣＰＵをさまざまな業務に分けるわけです。このようなマルチプログラミングの手法はほとんどの大企業において利用されています。マルチプログラミングのためには，オペレーティング・システムと呼ばれる複雑なコンピュータ・プログラムを使ってＣＰＵへの仕事の流れをコントロールし，そしてその時間の利用を管理します。オペレーティング・システムにより，コンピュータはそれぞれのプログラムをかわるがわる処理します。そしてそれぞれのアクティビティの優先度に基づいて計算をするわけです。

オペレーティング・システムについて監査人は十分に関心を持たなければ，事業の効果的な運営管理はできません。オペレーティング・システムは管理の中枢なのでそれによりコンピュータを利用するすべての業務に接近することができます。ある意味でマルチプログラミングのオペレーティング・システムという概念は，監査人の基本信条である業務の分離に矛盾することがあります。この業務の分離という概念によって別々の人が違った角度から業務を管理することになり，それによって不正を起こす機会を少なくしようとしています。オペレーティング・システムはすべての業務を管理しているのですからオペレーティング・システムを管理することは，すべてのアプリケーションプログラムを管理することになります。

コンピュータ科学者は，オペレーティング・システムが不正な目的を持ってコンピュータに近づく者に対して最大の弱点であることを認識しています。そしてこの点を証明しようとしてコンピュータ科学者たちは電子的またプログラミング・コントロールを使ってシミュレーションをしようとしています。このオペレーティング・システムに侵入するという点は非常に重要な点であり，コンピュータ・システムが不安であるという点もここに出てくるわけです。

監査人の業務は，オペレーティング・システムの複雑さによって決して簡単になることはありません。巨大なコンピュータ・システムは，一人の人が理解するには複雑過ぎます。しかし先に述べたようにオペレーティング・システムはコンピュータを利用するすべてのアプリケーションを管理する鍵になるわけです。

さらに監査人の仕事は，マルチプログラムの下で多くのリモートターミナルを利用することによって複雑になります。

コンピュータへの入力にターミナルを使用することで事務量や，データ伝送の時間も減少し，さらに初期の段階で入力することによって転記エラーも減少させることができます。

ターミナルからデータをエントリーするプログラムはどのようなものとなるでしょう。そしてこれらのプログラムはシステム・マネジャーが意図したことができるだろうか，また以前は事務員がやっていたコントロールの代替ができるのだろうか，また良心的な人間にかわってエラーを発見できるのだろうか，こういった質問に答えるには，プログラムがこういったことができるかを検討し，また特別なテストをすることによってその有効性を確認する必要があります。しかし，このようなテストを作成したり，その結果を評価するにはコンピュータ技術の専門知識が必要です。さらにターミナルを通して，コンピュータのオペレーティング・システムへの侵入や，セキュリティが害されることがしばしばあります。ターミナルを通してオペレーティング・システムへの侵入をふせぐシステムの作成，またそのシステムの評価には多くのコンピュータの専門知識が必要です。

それではここで，いままで組織においてコンピュータの使用がどのように変わってきたかを見ることによって，コンピュータのパーフォマンスのインテグリティとそれから監査の問題に対する視点が変わってきたことを見るべきだと思います。

1940年代後半に計算機が使われたのは，科学的な計算でした。当時のコンピュータはコスト・パーフォマンスで考えると，現在のものに比べてかなり高価でした。当時コンピュータはオフラインで使われ，データ・フローの統合的な一部をなさないような形でしか組織とは結びついていませんでした。当時の主要な関心事はハードウェアの信頼性と記憶容量の問題でした。1950年代にかなりの業務にコンピュータが使われるようになってからでも，やはり主な関心事は引き続きハードウェアの信頼性と記憶容量の問題でした。さらにつけ加えてコストの観点から，スピードとスループットの問題も問題視されるようになったわけです。ハードウェアの信頼性の問題は，1950年代後半から1960年代初頭に真空管にかわってソリッドステートのトランジスタ回路が使われるようになってから，問題はかなり軽減しました。また記憶容量の問題もやはり磁気テープや磁気ディスク等の磁気媒体を基礎とした周辺装置が多様化し安く手に入るようになってからかなり削減しました。コンピュータがオフラインで使われるときは，スループットに対する時間的な圧力は少なくなります。というのは，人間の介入しない形で直接コンピュータのアウトプットに依存して自動化された意思決定が行われていないからです。人が介入すれば，実際に決定を下す前に非常に明らかなエラーであれば，それを取り除くことができるからです。

1960年代の中頃から，コンピュータのインラインおよびリアルタイムでの使用が急速に増加してきました。

インラインないしはリアルタイムのアプリケーションは，コンピュータがその組織の機能にとって不可欠な部分をなし，また実際に意思決定に使われるデータがコンピュータを通過し，コンピュータによって操作されるわけです。オフラインの状況ですと，組織のオペレーションはもし必要とあれば，コンピュータがなくても機能を続けることができます。しかしインラインになると，もうコンピュータは組織の一部になっており，コンピュータに対する問いかけ，コンピュータの与える答え自身がもうすでに組織の情報の流れの一部をなしているわけで，ちょうど人が必要なのと同じように，インラインでは，組織にとってコンピュータが欠くことのできないものになるわけです。こういう形でコンピュータを組織のトップが使うようになったのは，コンピュータのコスト・パーフォマンスが上がり，信頼性が上がったためで，これは大量生産のハイスピードの集積ソリッド回路の開発によるものです。そしてあらゆる組織でインライ

ン，リアルタイムのアプリケーションが現在では行われています。たとえば電力会社などの場合，コンピュータがエネルギーの利用に関する負荷のファクターの報告を得て，自動的に発電能力とネットワーク内のエネルギーの配分をリアルタイムで変えることができます。化学工場においても，コンピュータはリアルタイムで製品の品質・成分に基づいて原料の流量を調整し工程の管理をすることができます。また小売店では販売を記録し自動的に請求書を送ることができますし，製造工場では原料の使用状況を調べ，業者に対し在庫を補充するように自動的に通知を出すことができます。アメリカ連邦政府において，たとえば連邦航空局の管理している航空管制にもリアルタイムで使われていますし，国税庁，社会保障局などでも使われていまして，市民の何十億ドルにものぼる送金・計算を自動的に行っているのです。

ここで注意しなければならないのはコンピュータがこういう形で利用されるようになると，つまり一たん設置されてしまうと，もう再びマニュアルのオペレーションに戻ることは不可能だということです。これは組織の業務がコンピュータの能力に合わせてつくられるからです。コンピュータは高速，大量，正確に業務を処理できますがもし故障を起こした場合に，人々がそれに取って代わる事は不可能です。たとえば，航空管制の場合，航空機の頻繁な発着は，管制官が電子的にとらえたフライト，インフォメーション，および自動的にうつしだされるディスプレイ等コンピュータの助けをかりることによって可能となるのであってこれを人手で行うことは不可能です。もしコンピュータが故障した場合には，このような頻繁な発着は，非常に危険です。こういったことから一つの事実がはっきりわかります。もしコンピュータ・システムのオペレーションが正しく行なわれないと，組織は大混乱に陥る可能性があるということです。正しいオペレーションは，ソフトウェアに大きく依存しています。つまりデータ処理を行なうすべてのプログラムに依存するわけです。たとえば故障の可能性を少なくするためにハードウェアを二重化することはできましょう。しかしソフトウェアの場合，二重化することはもちろん可能ではありますが，そうしたことは何も意味がありません。というのは，ソフトウェアの問題はすでに存在はしているが，まだ発見されていない論理的エラーであり，デザイン上の誤りであるからです。ソフトウェアは，人々がデザインし，プログラムし，テストし，設置するのですから，コンピュータ・システムが正しく運営されるかいなかは，ハードウェアではなくまさにソフトウェアをデザインした人間にかかってくる訳です。これは非常に重要な意味を持っています。

人件費に対するハードウェアのコストは過去２０年間急激に変わってきました。現在人件費は全システムコストの９０％以上を占めるほど多額になっています。２０年前はこの比は全く反対でありました。当時のハードウェアのスピード，信頼性，スループットを高めようという努力によってコストが大幅に削減されました。現在は労働力のコストの方が大きいわけですから，コストを大幅に削減するためには，人々の生産性を最大限にしなければなりません。

人間にかかわるコストがこのように高いのですから，プログラムおよびソフトウェアを組織的に開発する手法を緊急に見つけ出さなければなりません。２０年前からほぼ今日に到るまで，こういった活動は重要でしたが進んで取り扱われてはきませんでした。プログラミングとそれに関連した業務は，論理的な検証を経ない，または標準的な手法を使えない手工芸のようなものになっていました。しかしいまやコンピュータが一つの組織のなかでリアルタイムで運用される必須のものとなったのですから，ソフトウェアの完全性が非

常に重要となり，ソフトウェア・エンジニアリングというものこそが開発，研究にあたっても非常に重要な役割りを占めるようになったわけです。

そこで，コンピュータ・システムの実行の完全性を確保するという懸念を示したのは，連邦標準局の報告書の３１と４１です。これはコンピュータのセキュリティに関する問題でした。また同じようにアメリカの議会の監査を行なっている団体である会計検査院も同じような懸念を示し，二つ報告書を出しています。題名は，「連邦政府内におけるコンピュータによる自動的な意思決定の管理の改善の必要性」と題するものと，「連邦プログラムにおけるコンピュータに関連した犯罪」という報告書でした。

前者の報告書が示していたのは，正しくない決定を自動的に行った例で，こういったエラーが起こるのは二つの理由があります。ソフトウェアとデータの誤りです。誤まった意思決定を下す原因となったソフトウェアの問題はソフトウェアが要求通りつくられていない場合，データ入力の際に必要なチェック，コントロールをしなかった場合等におこり，データ・エラーは，不完全，不正確また古いデータ等を用いた場合におこります。

また会計検査院は，二番目の報告書のなかで組織の資産がいろいろな誤用の危険性にさらされることから，データ・プロセッシングの危険性とを打ち出しています。この報告書によると，コンピュータにより業務がほんの少数の人の手に握られることになり，先ほど申し上げたオペレーティング・システムと同じような問題になるわけです。この報告書はコンピュータ関連のシステムを評価・検討することができる十分な訓練を経た監査人による系統的な内部からの見なおしとシステムがオペレーションに移った時と同様に，それ以前のデザインの時から監査人が関与することを求めています。

損害は必ずしも金銭的なものだけではありません。たとえば個人の秘密といったものも組織の資産になるわけですから，そういった秘密が暴露された場合には，大きな損害をこうむるわけです。

報告書が最も重要だと指摘しているのは，専門家の努力を傾注し，コンピュータ・システムが与えられた仕事を十分に行ない，しかも与えられていない仕事は行なっていないということを確認しなければならないということです。この場合，ただ単にコンピュータ・オペレーションだけではなく，その前のデザイン，開発，コンピュータ・システムの入手，設置ということにも目を向けなければなりません。というのは，オペレーションが正しく効率的に行なわれるためには，それ以前の作業が十分に行われることが必要になるからです。

確かに監査人は直接そのような業務を行なうためのチームのメンバーという形では責任を担わないかもしれませんが，少なくとも経営陣に対する助言者として情報処理が正しくまた効果的に行なわれているかを確認すべきです。また監査人は，組織の資産を守るという役割りを担っているわけですから，彼はコンピュータ・システムのデザイン，開発，取得，設置が最も有効な形で行なわれていることを確認しなければなりません。それには，これらの業務に関する専門知識とそれがどのように行なわれるべきかを知る事が必要です。

よいシステムをつくり出す一つの重要な条件は，情報処理システムの設計者と実際にそのシステムを使用する人々の間の協力とコミュニケーションであります。もし両者の合意が得られないと，結果として設備を有効に利用できなかったり，無駄な演算をしたり，組織の要求に不適当といったまずいデザインになってしまう可能性があります。監査人は，すべてのユーザーの要求を満足させる合意に達したかどうか確認しなければなりません。

もし両者の合意がない場合やシステムが目的にかなわない場合は組織の資産を浪費することとなります。

確かにこういった意見の一致が見られないことは，いろいろな利害が対立することからあり得るでしょう。一つの組織が一つの目的を達成しようと思った場合にもそれぞれ異なった目的と機能を持った部門ができるので，それぞれの部門ごとに独特の物の見方を持っておりますし，その部門の役割，またそこで働く人々の個性を反映した反応の仕方をするようになります。主に会計士が構成している会計部門と研究員が構成している研究部門という異なる部門ではそれぞれの責任範囲と機能という観点から全体的な目的を見ようとするでしょう。またこれらの両部門ではエンジニアリングとか販売とか生産の部門とはまた違った見方でその状況を捉えようとするでしょう。こういった見方の違いというものは当然正当化されるものです。

というのは，今日の組織の中では機能の専門化が必要だからです。したがって組織内の情報処理機能の拡張をはかる場合，特にそれが総括的な経営情報システムの開発を目的とするとそれは組織のなかに混乱をひきおこすことになります。これらの混乱は単に情報処理そのものに関連しているかもしれないし，組織全体の政策とか資源配分に問題があって，それが新しい経営情報システムが導入されるにあたって表面化するということが十分あり得るわけであります。というのは，そのシステムが利害の対立を含むような内容のデータをもたらすことになるからです。

数年前にアメリカにおきまして完全な経営情報システムということがさかんに言われました。しかし時間がたつとともに経営情報システムは組織にとって完全な形では，つくり得ないのだということがわかりました。また「ハーバード・ビジネス・レビュー誌」の中でいろいろな論文が書かれ，その問題を取り上げました。ただ，こういった研究の結果が事前に評価されなかったということは非常に残念だと思います。つまり人間の業務とコンピュータの業務を統合するという問題が大企業の経営者には十分理解されていなかったのは非常に残念なことでした。

監査人は，こういった対立の存在を認識し，基本的な情報処理については合意が得られるように努力しなければいけません。たとえばこういった問題では，処理された情報を一体組織の階層の中でどこにリポートするか，また情報処理施設は異なるユーザー間でどのように配分されるか，どのユーザーに優先権を与えるか，情報処理のコストはどこがどう負担するかという問題があるわけです。したがってユーザーの協力と参加を得るためには，ユーザーの機能を縮小したり，省いたりするようなことがあってはなりません。仕事を充実させることこそが，協力を得る一番いい方法かもしれません。この人間とそれからオートメ化という問題について話をしたわけでありますが，この点は特に重要なポイントだと思います。

ユーザーとの間に合意とコミュニケーションが確立されますと，次はシステムの詳細と要求を明確化しなければなりません。コンピュータ・システムの機能についての詳細で明確な仕様書の重要性はいくら強張してもしすぎることはありません。つまりその仕様書がハードウェア選択の基準，プログラマ，アナリストの仕事の決定，またコンピュータ・システムの効果測定の基準ともなるからです。またコンピュータ・システムが実際に仕様書どおりにいっているかを確認するためのテストも考えておかねばなりません。つまり仕様書は業務を成功に導く土台になるわけです。

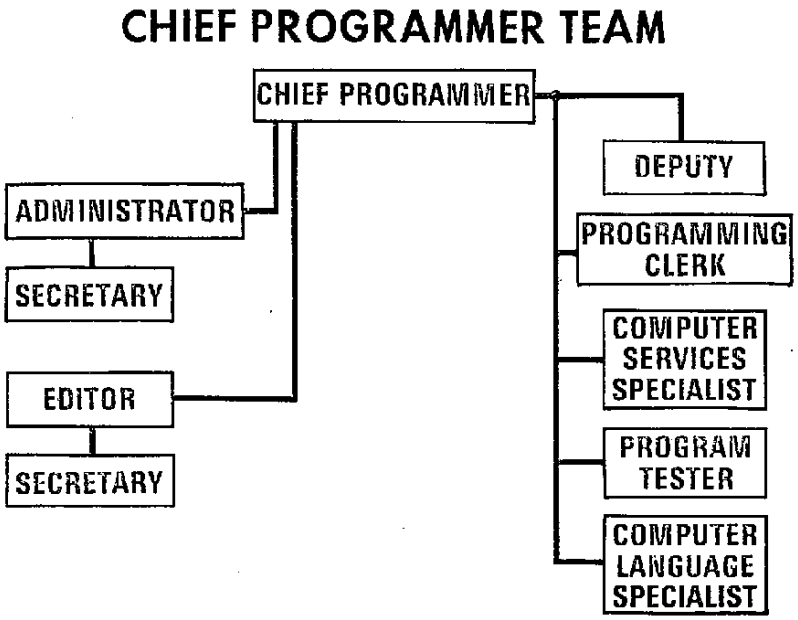
いかなる組織においても重要なステップは，コンピュータ・ハードウェアを入手することです。これは財政的な問題のように思われますが，この種の決定をするためには技術的な

専門知識も必要です。たとえばコンピュータを購入するか，リースをするかというのは，そのシステムが後で時代おくれになってしまうかどうかも考えなければなりませんし，またコンピュータを社内で使うのか，サービスビューローにするのかというのは，システムをサポートできる人員によるわけです。

また，さまざまなコンピュータの能力を比較する場合，しばしばベンチマーキング（benchmarking）の手法がとられます。これはある種のスタンダード・オペレーションをやってスループットを比較するわけで，連邦標準局の連邦情報処理標準規格42号がベンチマーキング用のガイドラインです。またもう一つの比較はシミュレーションによって可能であり，機械の数学的なモデルをつくって，いろいろな条件のもとでシミュレーションを行なうわけです。監査人はこれらの技術について知識を持ち，情報処理設備への資産の配分の意思決定の理解，確認ができなければなりません。

もう一つは，コンピュータ・プログラム開発のための組織です。エラーを含まない効率的なソフトウェアを生産するワーク・チーム作成の問題点は，多くの創造力に富んだコンピュータ科学者のエネルギーを必要とすることにあります。というのは，ソフトウェア・エンジニアリングとトータル・システム・コストの間に，より緊密な関係があるからです。

最初にソフトウェア・エンジニアリングについてお話しますが，これがダイクストラさんの講演と重ならないとよいと思っています。しかしわれわれ三人の講演には多くの関連する点があります。というのは，ソフトウェア・エンジニアリングがよい場合には，その結果として資源を節約することができるわけで，これはコンピュータ監査にも非常に大きな意味を持っています。ですからそういう観点に立ってコンピュータ・ソフトウェア・エンジニアリングのさまざまな側面を検討することができます。



ソフトウェアの効率的生産のためのチーフ・プログラマー・チームを使うという概念は1971年最初に発表されました。この計画によると，プログラマー10人のチームを編成し，それをチーフ・プログラマーが指導するわけです。この図1を見ますと，チーフ・プログラマーを中心にしてプログラマー・チームがどのように構成されているかがわかります。ほかにデピュティ・チーフと1人のコンピュータ・サービス専門家，プログラム・テスター1人，コンピュータ言語の専門家1人，エディター1人，ドキュメンター1人，アドミニストレーター1人，事務員3人です。このチームは個人の仕事から共同作業にプログラミングを変換させるもので，分業という概念（製造業では非常に大切であります。）をコンピュータ・プログラムの製造に拡張したものです。その目的は，訓練された人員を最も効果的，生産的に利用するという考え方です。

仕事の量によっては，プログラミングの作業にいくつかのプログラマー・チームが必要な場合もあります。その場合少なくとも相互に関係を持たなければならない人の数はかなり減ってきます。何か業務をしようとする場合，コミュニケーションをする人の数が算術的に減ればコミュニケーションの問題は幾何級数的に減ってきます。ですから，プログラマー・チームをつくることによって相互の交流の数を減らすことができ，この場合にはチーフ・プログラマーがグループの監督を担当するわけです。

監査人はこれらの組織概念を理解しなければ，コンピュータ・システム設計がもっとも効率的にデザインされているかを，またほかの人々の作業の質をも適切に評価することができないわけです。

ときおり監査人はジョブ・ディスクリプションを検討します。ジョブ・ディスクリプションというのは非常に重要な書類であり，監査人はそれに基づいて意図されたとおりに仕事が行なわれているかを確認するのです。チーフ・プログラマー・チームをよく理解することはソフトウェア開発組織を検討する上に欠くことができません。ジョブ・ディスクリプションを検討する際にチーフ・プログラマー・チームの働きをよく理解しておく必要があるからです。

監査人が理解しなければならないもう一つの重要な概念は，ソフトウェアを開発する際の時間と努力の配分で，それはプランニング，コーディング，サブシステム・テスティング及びトータル・システム・テスティングです。この間の時間と努力の配分をどうするかを理解することが重要なポイントです。

米国のフレデリック・ブルックス・ジュニアという有名なコンピュータ科学者はこれを以下のように分けています。

プランニング$\frac{1}{3}$
コーディング$\frac{1}{6}$
サブシステム・テスティング$\frac{1}{4}$
フルシステム・テスティング$\frac{1}{4}$

ここでわかるように，$\frac{1}{3}$がプランニングに，各種のテストには$\frac{1}{2}$の時間が使われております。唯一の生産活動であるコーディングには全体の時間の$\frac{1}{6}$しか与えられていません。このようにプログラミングは非常に変わった製造活動ですが，十分なタイムとコストを与えられておらず，後で改良したり，エラーを訂正する必要が出てくるわけです。ですから，よいプロジェクト・マネジメントの場合には，十分な時間と費用の正確な見積もりをしなければならず，健全な財政と経済的な資源配分がなされなければなりません。生産物のコストに関する技術的決定と資源をどこに，どのように配分するかという決定には密接な関連があります。マネジメントの助言者として監査人はソフトウェアの開発の時間的な要素を認識しなければなりません。

他のソフトウェア・エンジニアリングの技

術も最近広く使われるようになってきました。それらのなかの，HIPO（Hierarchy plus Input—Process—Output）という技術はもともとドキュメンテーションの道具として開発されたものです。

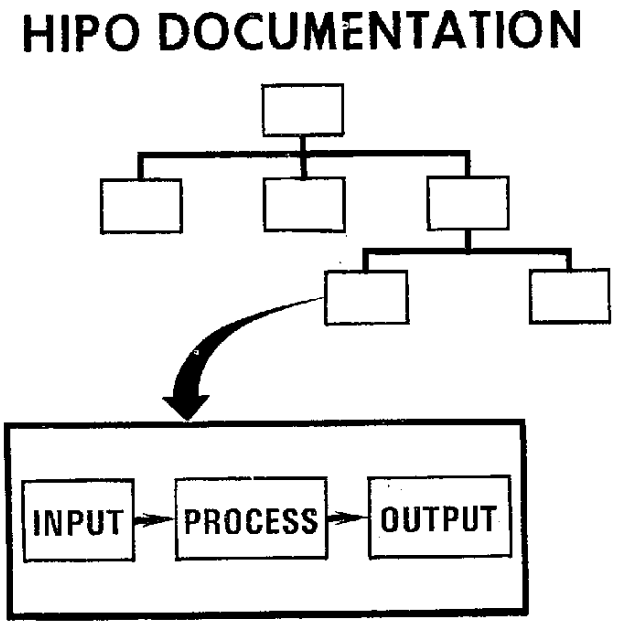


図2は，二つの基本的な成分から成っています。ひとつは1枚のハイアラーキー・チャート，これはそれぞれの機能がどのようにコンピュータ化されるかをあらわしており，いくつかのサブファンクションに細分化されています。他のひとつはインプット・プロセス・アウトプット・チャートで，それはそれぞれの階級組織におけるサブファンクションの入出力を表現しています。このHIPOという概念はコンピュータ化する機能を明確，論理的に表現することができ，ハイポによる設計戦略を利用することによって分析が容易になります。HIPOは，コンピュータ・プログラム監査の際，システムの把握を容易にし，監査人の業務を簡単にする可能性があります。

もう一つ重要なのはプログラムのドキュメンテーションです。HIPOの技術を利用してプログラム開発のセルフ・ドキュメンティング・フォームができます。その意味で二重の目的を達することができ，監査人とオペレーションと両面において貢献できるわけです。

コンピュータ・システムを開発するにあたり，監査人は積極的にソース・データのコントロールの設計，またオペレーティング・コントロールの設計にも関与するべきです。データのコントロールには監査証跡を残すことやデータの信頼性を保証することも含まれます。監査証跡は，必要な場合にはデータの動きを再構築し，データ・フローを検証するわけです。監査証跡を維持するためには，データ編成，伝送，獲得時でのコントロールがあります。

また，データの信頼性に対するものとしては，入力，処理，出力のコントロールがあり

ます。入力のコントロールはデータの確認を含み，コンピュータ・エディティングともいわれます。これは妥当性の基準をもうけてデータをチェックすることで，たとえば不正確なもの，道理に合わないもの，不完全なもの等を取り除くことです。データの妥当性を確認することによって正確度を期し，データ編成と承認の手続が行なわれます。そしてそれぞれのデータ要素の高度なテストを行なうことができ，人手で行なうよりも非常に有効になります。

処理のコントロールについては，リスタートとリカバリーの必要を考慮しておかなければなりません。これらは，ハードウェアやソフトウェアが正常に働かなくなってしまった時に，データを失ってしまったり，再度長たらしい，プログラムを実行するのをふせぐためのものです。リスタートとリカバリーはたとえば人間の命にかかわる航空管制のようなリアル・タイムのアプリケーションにおいて特に重要です。このような場合には，「フェイルソフト」コントロールが必要となってきます。「フェイルソフト」とは故障を表面化させずに，全体の能力が落ちても稼動を続ける方式のことです。

オペレーティング・コントロールとは，システムに対し認められていないアクセスを防いだり，公認されていなかったり，また乱用によってすでにつかわれているデータやプログラムが修正されるのを防ぐことです。このようなコントロールは権限を分離することによってコンピュータの「オペレーティング・システム」を保護しています。コンピュータのオペレーションとオペレーティング・システムの修正は二つの別個の業務にしなければなりません。というのは，このプログラムは非常に微妙なものだからです。またオペレーティング・コントロールはターミナルからのコンピュータに対する，アクセスにも関係がありユーザーはターミナルから自分に認められたデータ，プログラムにしかアクセスできないようにしなければなりません。

コンピュータ関連の犯罪を防止し，探知することの重要性はいくら強調してもし過ぎることはありません。多くのコンピュータ関連の犯罪をアメリカで分析をしてみると，ユーザーがデータ処理の初歩的な知識しか持っていない例が多く，高度な知識を持った例はほとんどありませんでした。これは大変興味深いことで，コンピュータ科学者は，オペレーティング・システムに対する侵入に関心を持ち，特に高度なユーザーが侵入する可能性を恐れているわけです。もちろんこれらの点にも注意を払わなければなりません。しかし実際にアメリカで起こったコンピュータ関連の犯罪を分析してみると，オペレーティング・システムに侵入することすら知らないような初歩的な知識しか持たない人が多かったわけであります。ですから，多くのコンピュータ犯罪というのは現在のコントロールとリスク分析によって防止することができます。リスク分析は潜在的脅威を分析するもので，その脅威は事故によって，たとえば洪水とか火災とかその他の天災によって，または意図的に破壊されるという可能性も含みます。

興味深いのはスタンフォード・リサーチ・インスティテュートがやった有名な「コンピュータの乱用」と題される報告であります。報告書のなかには物理的な乱用の例としてたとえば鉄砲やピストルでコンピュータに発砲した例もありました。われわれはこのような幼稚な介入も電気的な高度な介入同様防止するように努めなければなりません。

監査人の仕事は潜在的な危険や損失についての情報をマネジメントに与え欠陥を是正する代替案を提出することです。また監査人は物理的なセキュリティと技術的なセキュリティを認識しておかなければなりません。これらの技術は急速に進歩しつつあり，そのなかには声紋分析や自動指紋分析も含まれます。

また，監査人が関与しなければならないもう一つの大きな問題は開発され，修正されたコンピュータ・システムのテストです。それは現在大規模な研究が行なわれているコンピュータ科学の分野であります。重要な問題は，いろいろなファクター・バリューの組み合わせによってコンピュータ・プログラムが影響を受けると，テストできないということです。というのは，その組み合わせが非常に大きいので，コンピュータですら取り扱うことができないからです。監査人は十分な資源をシステム・テスティングに対して割り当てなければなりません。そしてエラーのないシステムをつくり上げるよう十分慎重に努力しなければなりません。テストは金銭的なロスを起こさないためにも重要なポイントですし，また秘密のデータを扱うものやプロセス制御においては人権や安全にもかかわるので非常に重要であります。

コンピュータ・システム，またはプログラムが意図されたとおりの機能を正確に実行しているかどうか検討するにはまず最初に仕様書の見なおしが必要です。残念ながら仕様書は余り正確に書いてありませんし，いろいろな可能な条件を全部カバーしていません。また実際の設計が仕様書に合わないということもあります。開発の過程においていろいろな変化が起こり，それを完全に文章化していないこともあるわけです。

また，テストを行うにあたって考慮しなければならないのは，性能です。性能とはコンピュータ・システムの能力のことで，異常な条件のもとでも必要なスピードと負荷にたえられなければなりません。約９ケ月前アメリカにおいて航空機の自動座席予約システムが一時に多くの飛行スケジュールがとりやめになったため負荷が高くなり過ぎてダウンしてしまいました。このような高い負荷は切符を持った人々が彼らの予約を変更しようと殺到したためにおきたもので，通信回線上でデータがが失われてしまいました。

その結果フライトに対する予約が多過ぎたりして多くの乗客が迷惑を受けたわけです。この程度の異常な負荷はもちろん最初からデザインする必要はなかったのですがただもし異常に高い負荷があった場合にはデータが失われることがないようシステムをデザインする際考慮しなければなりません。

アメリカには，いわゆるマーフィ法というのがあり，コンピュータのシステム設計の際何かがうまくいかなくなった場合を考慮しなければならないことになっています。もちろん事前にあらゆる不測の事態を予測することは不可能です。しかし，最善を尽くして予測し得る限りの緊急事態に備えなければなりません。テストの際には「フェイルソフト」のデザインとリカバリーの能力を十分考慮しなければなりません。

テストはプログラムの設計，開発後同様，修正後にも行なうべきです。複雑なシステムではプログラムが互いに理解しがたいように影響し合います。テストを行なうことによってのみ修正がシステムに悪い影響を与えたかどうかがわかるのです。ウェーバーさんのお話によると，システム間の相互作用を下げるためにも，小さなコンピュータを用いての分散処理は有効な方法だという事です。

テストを行なう戦略にも監査人は参加すべきであり，どういったデータを使うか，どういった種類のテストを何回行なうかということにも注意しなければなりません。テストを行なう際には，個々のプログラムと同時に，システム全体にも関心を払うべきです。

コンピュータ科学者たちは，プログラムの構造をチェックするという概念を確立し，そのための分析プログラムを開発しています。これは，プログラムの論理構造を検討し，それぞれのブランチ・ポイントに影響する条件等を明確化します。またトレース・プログラムというものを使って，余り使われていない

コードとか冗長なまたは非経済的コードを発見することもできます。

システム・テスティングという概念は，複雑なエレクトロニクスや宇宙関係の機器をチェックするために，技術者たちが用いている技術を検討する際に開発されたものです。これらの技術は本質的にはシミュレーションの形を使い，実際に現場で使われるようなデータをシステムに入力するのです。設置前にシステムの十分納得のいくシミュレーションを行なわなければそのシステムに対して完全な信頼性を抱くことはできないと思います。シミュレーションのプロセスは，実際の環境のかわりにドライバー・プログラムの助けをかりることがしばしばあります。シミュレーションは現在の監査手法たとえばテスト・デッキングとかインテグレーテッド・テスト・ファシリティとかの拡張であります。

ここでテストの問題に関して申し上げたいのですが，これは機能の差に依存しているということです。コンピュータ科学者，プログラムを開発する人々は，プログラムを何回もテストし，実際に使おうと思うものをオフラインでテストします。つまり正確さをテストするわけです。ただ実際にユーザーの環境でどうなるかというテストはしません。というのは，開発する側であって，ユーザーではないからです。しかしユーザーにはユーザーなりの使用上の問題があり，コンピュータ科学者は，いわば実験室の中ではテストを行なうけれども，実際の現場でのユーザーの状況ではどうなるかということをテストできないわけです。ユーザーの直面するであろういろいろな細かい問題について開発する側が知らないままにその要因を入れられないということから，開発する人とそれからユーザーというものが分離しているという事実を十分考える必要があると思います。それによって初めてコンピュータの真の正しい開発と利用というものを結びつけ，本当に統一された形の作業をデザインから仕様書まで，われわれはオペレーションの実際の使用に至るまで確保することが必要だと思います。

さて，今までのところコンピュータ・システムの組織内部の問題にのみ触れてきました。しかし対外的な問題はどうでしょうか。コンピュータを設置したり，コンピュータを変えたりした場合には，その組織の顧客に対し，また他団体に対してどういう影響を与えるでしょうか。

監査の文献の中にこの問題をほとんど見ることができません。しかしながら実際のところ組織が成功するか，失敗するかは，対外的な関係においてであります。財であれ，サービスであれ組織の提供するものは，第三者によって判断されます。したがって対外関係がもっとも重要なわけです。そして対外関係というものを会計士はグッドウイル（のれん）というふうに呼んでいます。グッドウイルこそは監査人が保護しなければならない財産であります。

現代社会においては個人的関係は非常にまれであり，多くの関係は非個人的でありデータの伝達をもって処理されます。これらのデータはデジタルでコンピュータがつくり出したものです。民間部門ではこういったデータは会計的なものですが，公的な部門ではいろいろ統計の表になることが多いわけです。

この対外関係について監査の文献の中にはほとんど書かれていないということは非常に驚くべきことです。というのは，新聞でコンピュータがうまく働かずに起こったさまざまの悲劇についての報道があるからです。

おもしろい話があります。ある店の顧客が０ドル０セントの請求書を何回も受け取りました。そしてしばらく後には，もし払わないとクレジットの特典を取り消すというふうにおどかされました。この顧客は０ドル０セントの小切手を切って店に送ったら，その後そういう手紙は来なくなったという事です。

しかし次のような話もあります。米国では，多くの州において，盗難車をレポートするために警察はコンピュータ・ネットワークを利用しています。システムに対するアクセスはパトカーに設置されたターミナルを通して行なうことができます。ある地方の道路で車がとめられました。というのは，そのライセンス・プレートが盗難車のものだったからです。そしてドライバーは警察官の命令で車からおりてきました。しかし彼がちょっと変な動きをしたので警察官は彼が武器を持っていると思ってその人を撃ってしまいました。しかしこの車は実は盗難車ではなくて，後でデータが間違っていたということがわかったわけです。

いま，一つはおもしろいものと，一つ非常に悲劇的な例を挙げましたが，これは共通して外部の者に対する影響を十分考慮に入れていなかったことによります。こういった問題に対しては，「自動化の人間的側面」というふうに言いますが，私はこれは監査の人間的側面というふうに言いたいと思います。というのは，コンピュータ監査人はこういった問題を認識し，それを経営陣に報告し，是正を提案する立場にあるからです。

この対外的な関係における問題は，システムの設計者がデータの内部的な利用者のみを考えていたことから起こったことが多く，実際外部にいる人，外にいるユーザーは，組織内部で彼の意見を代弁してくれる人はいないわけです。またシステムの設計の際，そのシステムが外部と互いに影響しあうにもかかわらず外部の者を十分考えてはおらず実際のところ外部の者は一体どういうふうにしたら自分の不満を訴えることができるか，またこれを直してほしいと通告することができるかわからないわけです。そういったコミュニケーションの方法はまだ公式には確立しておりません。また組織が非常に大きくなってしまうと，一たんエラーが起こってしまった場合，それを恒久的に是正するのは，内部の複雑な事情によって非常に困難です。場合によっては１人の部外者が報告したエラーが是正されるかもしれません。しかし組織内部で恒久的にこういったエラーを防ぐ方法が打ち出されなかったために，何回も繰り返されるということはあり得るわけです。確かに監査人が最も道理にかなった仕事をすれば部外者の問題も解決できるかもしれません。しかしそれ以上に監査人がシステムの設計・運営にあたって人間関係に多くの注意を払うことによって問題を早期に発見し，担当のマネージャーに恒久的にシステムの手なおしをさせていくべきです。

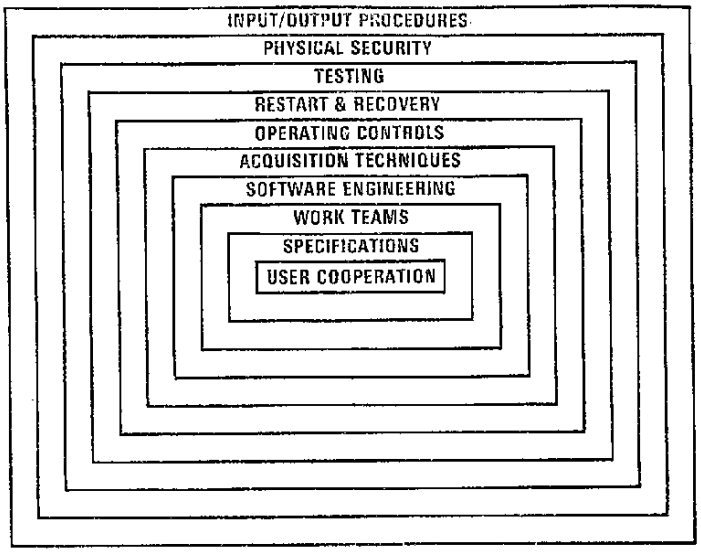
まとめてみますと，コンピュータ・オーディティングは，組織のすべての情報処理業務の管理と関連してきます。

図３は，監査人のいろいろな関心事を示しています。これは情報処理のすべての局面を含んでいます。たとえば，仕様書，設計，開発，ハードウェア獲得，設置，そして運用等です。またコンピュータ監査は，ソフトウェア・エンジニアリングに関心を示さねばなりません。というのは事業費の管理また事業組織と関連してくるからです。また資産の保護という観点からコンピュータ・セキュリティに，エラーをおこさないためにシステムのテストに大きな関心を示さなければなりません。

このようにコンピュータ監査はコンピュータとコンピュータ・システム運用についての多大の専門知識を必要とし，会計に関する知識も付随的に必要になってきます。これは特に会計的でないシステムやリアル・タイムのプロセス・コントロールの場合明らかです。これらのシステムは監査されねばなりません。というのはそれが効率的であることは同様に重要であるからです。そして監査にあたっては会計的なシステムの場合と同じような手法が使われるべきです。

コンピュータ監査は知的にも非常に深く広

AUDITING CONCERNS



い内容をもっておりそのアプリケーションは非常に重要です。それは一つの学問分野として呼ばれるに足るものであると考えられます。しかし現在のところ一つの学問分野として教えられてはいません。コンピュータの監査人が知らなければいけないことは，１カ所で修得できず，大学もコンピュータ監査人という形で学生を卒業させてはいません。コンピュータ監査が学問の一分野として確立されれば，理論とアプリケーションの両方の知識が必要になるでしょう。そしてそれは，専門的なコースを経て十分訓練を積み，その道の専門家にも認められるような専門家によってなされるべきです。現在のところ次のような方向に移るように思われます。まず最初にたとえば専門団体，政府またはその双方がその資格能力に一定の基準を設け，コンピュータ監査人の活動によって公益を守ることに関心を示すことが考えられます。

米国においては内部監査人協会がＥＤＰ監査委員会を設け，そこにおいてコンピュータ監査の基準を決定しようとしています。ただ，申し上げたように，コンピュータ監査人がするべき作業とその内容・方法についてはまだまだ定義づけが十分進んでいないと思います。また大学の方では知識を一堂のもとにまとめて研究を行ない，同時にコンピュータ監査人としての能力を十分つけられるような一つの教育過程を確立するようになると思います。

これは特にアメリカでは起こり得る傾向だと思います。というのは，経済社会の構造の中でいわゆる「知識労働」が占める割合が非常に大きくなっているからです。科学者，エンジニア，法律家，研究者また統計専門家，数学家そして教育家といったような人々が必要な知識を提供して，それによって社会の意思決定が行なわれるわけです。決定を行なう際，情報は判断の基礎となります。よき代替案を考えだすために，情報を収集し，転送し処理し報告せねばなりません。また情報に関

して行なわれるオペレーションには当然コストがかかりますから，情報はコストを支払う人の財産であるということになります。つまり情報は価値を持った有形の財として考えられ，それが一体どういう形の媒体に記録されているかは無関係なわけです。この情報の有形資産概念というのはアメリカではまだ新しいもので，現在も開発中であります。アメリカに有名な判例があります。1928年にオームステッドと米国が争ったもので，この裁判ではプライバシーが争点になりました。そして最高裁の判事であるブライダイスはマイノリティの意見として，「プライバシーは文明人の最も必要とするものである。」と述べました。しかしながらこれは有形な資産ではないという理由で認められませんでした。すなわち1928年当時情報は評価できる資産とは認められませんでした。しかしこの概念も変わってきており，1974年にプライバシー法案が通り，プライバシーというのは米国民がすべて権利として保有しているものであるというふうに言ったわけです。したがって情報を評価することができるという事はアメリカにおいて非常に大きな関心を引くようになりました。

ほかにもアメリカで法律ができ，情報の正しさの問題に関しては，フェアクレジット・リポーティング法，フェアクレジット・ビリング法の二つが出ました。これは過去数年内に制定されたものであり，個人に関して得られた情報の正しさに関するものです。コンピュータのオペレーションが盛んになったということから，アメリカの議会がこういった法律を通過させるようになったわけです。

ですから，情報管理という分野において，コンピュータの監査人は非常に重要な役割りを果たすようになると思います。そういう意味で，近い将来システムの正確性，極秘性の維持を含めこれらのシステムを誰かが確認することを要請されるかもしれません。そのような保証をする人をコンピュータ監査人と言えると思います。そして同時にコンピュータ監査人が受けるべき訓練の内容，知識，その職業につくための条件等の設定とこの目的を実行し資格を付与する組織も必要となるでしょう。私は今，現在の監査人のことを言っているのではなくコンピュータ監査人のことを言っているのですが，今のところまだそのような人は存在しません。現在のところ財務以外のパブリック・データを監査しなければいけないという要件はアメリカではありません。したがってだれがそのような保証を実際にしていくかということは議論の余地があります。しかしながら将来においてはそういった可能性も当然出てくると思います。1933年に企業の財務諸表の正しさに関する懸念が示されその年に法律が通り，その結果公認会計士が証券市場に上場されている会社の財務諸表の公正さについて証明をしなければいけないということになったわけです。

現在，個人情報やリアルタイムの公益事業を扱う情報システムに関する懸念が示されています。財務のみでなく非財務のシステムを扱うコンピュータ監査人が必要になってきておりますが，その方法論や専門分野というものがまだ十分に確立されていません。したがって，監査人とコンピュータ科学者がこういったチャレンジにこたえてこの部門を確立していくことが必要だろうと思います。

セッションⅠ　＜パネルディスカッション＞

システム監査と管理者の責任

システム監査と管理者の責任

パネリスト　石田　甫[1)]
上野英夫[2)]
島川聖明[3)]
西野嘉一郎[4)]
ロイ・ソルトマン[5)]
コーディネータ　堤　佳辰[6)]

堤　「システム監査と管理者の責任」こういったテーマをいただきました。日本の汎用コンピュータの普及状況は，本年度初現在で3万台を超えております。これは設置金額にしてざっと2兆円になります。これはアメリカの6万5,000台，金額にして300億ドルに比べ，設置台数で約半分ですが，世界で2位の普及度であります。銀行，保険，証券，商社，行政そういったオンライン化も進んでいます。さらに産業の各部門に電算機のシステムとしての利用が普及し，浸透し，定着しつつあります。

そういった意味で，これだけの投資を有効に使わなければならない。しかもコンピュータは，今日の経済，産業の中枢であり，その背後には数十万のコンピュータ関係者，そしてこれが日本の経済，企業運営の中枢神経として，日本のGNPそのものに作用しているわけであります。そういった意味で，これらのシステムは，今後さらに普及していくわけであります。大体過去6年間で6倍に設置台数はふえてきていますが，汎用だけで見て，今後の進捗状況を見ますと，おそらく昭和60年度末には約7万5,000セット，金額にしても大変大きくなるわけでありまして，今後の伸び率を考えた場合にも，今後の情報化社会の中枢となるこれらのシステムが有効に機能しているか，さらに安定に機能しているか，さらに企業内あるいは社会の信頼性というものを十分かち得る域に達しているか，こういったところにもおそらくシステム監査の基本的な考え方があるだろうと思います。

システム監査への要請ということを考えますと，一つはまず会計監査という立場からのシステム監査という考えがあると思います。第2の要請としましては，経営監査こういった立場からのシステム監査，さらに第3の視点からこれは社会監査としたいわけでありますが，社会に対する信頼性の保持という意味での社会監査，こういった三つの流れがあるかと思います。きょうは講師の方々専門の方をお招きしてありますので，こういう意味で

1）宇部興産常任監査役
2）日本公認会計士協会常務理事
3）富士銀行常務取締役
4）日本内部監査協会会長
5）米国商務省標準局コンピュータ科学技術研究所プログラムマネージャー
6）日本経済新聞社論説委員

御存分に問題を討議していただきたいと思います。

その前に，この情報開発協会で昨年からシステム監査委員会というものを設けて作業を進めております。これは５０年度に始まった作業ですが，過去１年余にわたる成果をことしの３月に中間報告書という形でまとめております。それを御参考までに御紹介して，今後のパネルディスカッションの参考点にしたいと存じます。

システム監査という場合には，「システム監査とは。」ということが一つのポイントになるわけですが，この報告書では，システム監査というものはマネジメント（管理）それからセキュリティ（保安）ですが，それに会計システムそれからリアルタイム，オンライン・システム，個人データなどの監査をも含む総合概念である，こういった非常に広いとらえ方をしております。

さらに「システム監査の対象とは何か。」ということですが，これは企画，開発，実働の全段階，すなわちコンピュータの運用ばかりではなく，前後の入出力のプロセスも含めた全体がシステム監査の対象になるであろうと考えております。

さらにこのシステム監査の実施基準が重要な問題になるわけですが，これには一般基準としては準拠性，採算性，適時性，生産性こういったものが挙げられます。さらに品質基準として安全性，信頼性，機密性，こういったものが各業務レベルに要求されることを考えております。

さらにチェックポイントを設定しなければならないわけですが，非常に大きなシステムですから，どこを重点的にチェックするか，チェックポイントの設定が重要になるわけですが，これは機能面ではコントロール，セキュリティ，プライバシー保護というものが重要である。さらに管理面では承認，標準化，ドキュメンテーション，スケジューリングというものがチェックポイントになるであろうという考え方をしております。これに従っておそらくチェック・リストというものを設定しなければならなくなる。このマニュアルに従ってシステム監査というものを進めていくわけですが，そのためにはおそらくシステム監査人，専門要員というものを養成することが必要になるであろう。

システム監査には，専門知識の監査技術さらに監査技法というものが必要ですから，こういった在来の監査体制にこだわらずに，企業内部あるいはその外にシステム監査人というものを養成していく必要がある。

こういったことで，ちょうど同じ頃アメリカでもこういった研究がスタートしております。期せずして日米共同でこういった研究を進めているわけです。そういった意味で，きょう講演してくださいましたソルトマンさんが参加しておられることは大変意義があることと存じます。

それではさっそく各部の方々からお話をしていただきたいと思います。

まず西野先生から，よろしくお願いします。

西野　私，内部監査協会の会長，同時にこの６月から大蔵省に設けられた公認会計士審査会の会長もやっております。

また，私は長年芝浦製作所の社長，会長などをやってまいりまして，いま相談役の位置にございますので，経営の直接責任から解除されておりますが，言うならば内部監査の立場，公認会計士の立場，ユーザーとしての管理者の立場という三つを持っておるものでございます。不幸にしてコンピュータに対する知識が，もう年寄りですので，ございません。これから申し上げることが果たして当を得ているかどうかということについて，かえってお恥かしいのであります。

先日ある方から，アメリカの２００年祭でエール大学の総長が講演をした，その中で私は非常に興味のある言葉を聞いたのです。それ

はアメリカの200年の最初の100年は政治の時代であった。次の100年は 経済の時代であった。第3番目はユースフルネスの時代である。こういうことを言って，これからの100年はユースフルの時代だ。つまり効率というか，利用というか，資源も少なくなるし，人間も高齢化するし，いろいろな意味において世界じゅうが一つの変化をしてくる。そこにおいて最も大事なことはユースフルネス，つまり効率の世界が必要だということを述べたのを聞いたのです。

私は，このコンピュータの問題はまさにこの第三のエージ，有効に使う，人間あるいは知能，あるいは世界の有限な資源をユースフルネスにどう使っていくかということの一つの非常に大きなツールである。同時にそれに伴うところのソフトウェアの開発というものがこれから行われてくるのだということを強く感じております。

皆様も御承知のとおり，日本の経済はこの20年間非常な高度経済成長をしております。エネルギーの問題一つとりましても,昭和30年にはわずか石油は1,000万キロリットルの輸入しかなかったのです。ところが御承知のとおり50年には3億キロリットルと30倍に増加しております。さらに通産省のエネルギー需給部会におきましては60年には6億キロリットルになるということでありまして，日本の経済は今後，いま堤さんからお話がありましたように，非常に大きなコンピュータの導入というものが出てきて，いまの倍ぐらいにすぐなるというお話でございます。なぜかと言うと，その裏にある日本の経済が非常に膨張していくということです。鉄鋼の問題をとりましても，現在日本の粗鋼生産能力は1億5,000万キロトンあるといわれております。いま大体1億1,000万キロトンしか生産しておりませんが，一番最大の49年には1億3,000万キロトンが生産されたのであります。しかもこれは輸出をのけまして，日本国民が1人当たり800キロの鉄鋼を使っております。そのときの統計によりますと，アメリカでさえ700キロしか使っていない。アメリカよりもよけい日本人が鉄鋼を使った年が49年，石油ショックの前年であります。

これだけ大きく日本経済というものが拡張していったということで，それをどう消化するかということについて，問題は結局コンピュータの利用というものが非常に大きな力をなしてきたのじゃないかと私は思うのであります。建築にしても，大きな設備を設計するにしても，また膨大な資料を瞬間的に処理する能力も，まさにこのコンピュータがあったからこそ日本経済がこの10年なり20年なりに高度経済成長をなしとげ，また今後も6%の経済成長でも，先ほど見たようにエネルギーはこの10年間で現在の石油消費量が倍ぐらい使わないと成長しないということでありますから，そうしたものの計算あるいはそれに対する経営の意思決定，設備の問題は非常に大きくなるのではないかと思います。

私がコンピュータを最初に見ましたのは，1955年にアメリカへ生産性本部のコスト・コントロールのチームの団長で渡米したときであります。これは終戦直後でしたが，ふしぎなところで，アメリカの南のテキサス州にサンアントニオという町がありますが，そこの郊外にテキサスのパブリックの研究所がございました。そこで非常な大型のコンピュータを初めて見たのです。そこで詳しい説明を聞きました。そして同時にそこではORという技術を導入し，コンピュータによっていろいろな経営者の意思決定，生産の方向，マーケティング，いろいろなものをやってユーザーに提供するのだという話を聞かされたので，これは大変なものができたなということで日本へ帰ってきた。1955年というと20年前ですが，もうアメリカではすでにコンピュータが多少使われていた時代でありまして，その他の企業を回りましたが，それほどコン

ピュータが利用されている時代ではございませんでした。

同時に，講演を聞き，ある１冊の書物を買ってきました。皆さんもお読みになったと思いますが，ＧＥの中興をなしたコーディナーという人の「ニューフロンティア・オア・プロフェッショナル・マネジャー」という書物です。この中で，これからの企業経営者の三つのファンクションというものを挙げております。

一つは長期計画を立てる能力，つまりこれから世界の経済は非常な大きな変化をしていく。人口もふえていくし，資源も枯渇してくる。そういうものをどう取り上げて長期計画を企業が進めていくかというこの能力がなければいけない。第２の問題としては，あらゆる情報を収集し，これを一般に伝達する機能，その能力がなくてはいけない。それから第三のプロフェッションとしてのファンクションとしては，株主，消費者，従業員あるいは第四のセクターというか，一般の大衆の人たちのモチベーションをいかにうまくつかまえ，企業のニーズに合わせていくか，これがプロフェッショナル・マネージャーのこれからの機能だ，フロンティアなんだということを書いておられることを，読んで非常に感動したのであります。

これらのことは，その当時はコーディナーが考えていたかどうかわかりませんが，結局今日のコンピュータの発達によって初めてこういうことが可能になったのだ。つまりコンピュータの機能は皆さんも御承知のとおりに経営者の意思決定をする一つの資料として，これを非常に短期間にいまコーディナーが言ったような三つのファンクションをコンピュータがすることと，それからもう一つは日本経済も非常に大きくなり，アメリカ経済もそうでありますが，非常に拡大されたこの業務量を普通の計算機器によってはとても消化できないものでもコンピュータを使えば消化できるということであります。

ところが，日本の現在数はなるほど３万セットですが，大変な金額になっていま日本にコンピュータが入ってまいりました。どうもトップマネジメントというか，最高管理者はいまのような大きな経済の中にコンピュータの占める位置，製鉄が世界一の鉄をつくるようになったのもコンピュータリゼーションのおかげであるということがわかっているのですけれども，最高経営者はコンピュータに対する知識が非常に少ない。そういうことから，コンピュータの導入に対しては比較的冷淡ではないか。そういうものを日本においては現在非常に有効に使っている大企業その他はございますし，また銀行その他においてはオンラインなども導入され非常に有効に使われております。一方においては，一つのムード的な導入というか，そういう問題があると私は思います。言うならば，これからのコンピュータの効率的な適用の死命を制するものは，何といいましても経営管理者自身の現在及び将来のニーズそのものがどこにあるかということを理解するかどうかにあるのだと思います。単に導入するということだけが問題の解決になるのじゃない。それに対してはやはり経営管理者，社長と申しますか，きわめて最高のトップは，そのこと自体の技術的な，テクニカルな知識がなくても，それに対してどう運用できるかという，先ほど申しましたような非常に複雑な情報を収集して，これをどう伝達し，それをどう経営の意思決定に持っていくか，また膨大な企業内におけるデータをどう短期間に処理していくかということが非常に大事なことであります。したがってコンピュータを導入するにおいて，その目的は何かということをわれわれはまず第一に考えなければならない。企業の大小によりましてどんなことでもできるような大型のコンピュータを導入するのは大変な金も要りますし，リースをするにしても莫大なリース料が要り

ます。

したがって，やはり企業は，まずコンピュータ導入の主なる目的は何かをまず決定する必要がある。顧客へのサービスのために適当な情報の入手をするのにコンピュータを導入するのか，あるいは膨大な事務量を削減して，事務費を削減するために，コンピュータを導入するのか，それぞれのいろいろな用途があると思いますが，それをまずわれわれは決定することが必要だと思います。

そのためには，経営管理者はやはりトップとよく相談して，その目的は何かということをよく話し合って導入しなければいけない。御承知のとおりコンピュータも，いろいろな用途のものができております。また周辺装置も非常にふえておりますので，きわめて価格の安いものでもその目的は達せられることが多々あると思います。

それから第二は，まずコンピュータという資源を利用する上において優先順位を明確に示すことが必要だと思います。優先順位を導入の時期で決定するべきでありますが，初めこういうことのためにするのだという優先順位を決めましても，世の中の状態が変わってまいりますので，その時代時代にやはりこれを適当に変えていかなければならぬ場合もございますが，やはりその場合において機器の取りかえあるいは周辺装置をどう入れかえるかという問題も検討する必要があります。

第三に，いま申しました目的や方針や優先順位が現在これが妥当にちゃんとなっているかどうかを反省する，考え直すことが必要ではないかと思います。したがいまして，いま最高経営者がコンピュータを取得するための意思決定をする場合に参画をするとかあるいはＥＤＰに対する方針の確立とか，新しいコンピュータの適応業務の導入に関する優先順位の決定とかあるいはコンピュータに関する長期計画の設定とかＥＤＰ部門の発展の監視などいろいろな問題がございますが，そういう問題に一番入りやすいのは，何といっても内部監査に属しているものでないかと思うのであります。

内部監査人は，そういう意味においてトップと，いわゆる一番最高経営者とよく相談する，その相談相手になることが必要であると思います。先ほどソルトマンさんから非常に詳細なアメリカの現状そして監査人との関係についてお話がございましたが，私も大変感心しました。

結局，日本でこの間日本情報開発協会が日本における何万台かのコンピュータに対してどう利用されているか調査したことについて，きわめて悲観的な調査の現状が報告されていることを私は読んだことがありますが，数はふえておりますが，アメリカのような企業の意思決定，あるいは経営全般に対する利用度はまだ非常に薄いのではないか。ただ工賃の計算を従業員がふえたからするとかあるいは購買事務がふえたからこれを消化するとかあるいは生産管理の一部に使うのだとかきわめて縦割りのコンピュータ利用が一般の皆さんの会社でなされているのではないかという感じがするのであります。つまり総合経営管理というか，総合管理の中においてコンピュータが使われているかどうかということに対して私は非常に疑いを持っています。もちろん日本の経済がここまで大きくなりましたし，人件費も西独に次いでのレベルにまでなってまいりました。先日日経連で調査いたしますと，日本の１人当たりの生産単位当たりの労働工賃，（労働コスト）はアメリカと大体同じくらいになりました。ドイツよりも高くなっております。これぐらい人件費が上がりましたから，そういう部門を徹底的に洗い直して，コンピュータによって人件費の節約を図っていくことがここで必要になってきたと思うのであります。

それの相談相手になり，最も効率的なコンピュータの利用を図らしめるものは，内部監

査の役に当たっている内部監査人の仕事だと私は思うのでありますが，不幸にして日本の内部監査をやってきた人たちはコンピュータに対する知識が非常に少ない，あるいは皆無だということも言えると思います。従来一般の企業内部監査のみをやってきた。後に公認会計士の方のお話もあると思いますが，公認会計士の方もコンピュータに対しては，どうもあれはむずかしい，おれたちはわからないのだ，どうも敬遠するという傾向があります。したがいまして，私が会長をしております当日本内部監査協会ではこういうことに着目しまして，４２年にこれからのＥＤＰ監査というものがどうしても必要になるから，内部監査人にしてコンピュータに対する知識を修養しなければならぬということで，ＥＤＰ監査研修コースというものを設置して，内部監査人にコンピュータの教育をすることをスタートしました。さらに４７年にはＥＤＰ監査技法開発委員会を発足いたしまして，さらにコンピュータ・システムに対する監査をいかに進めるかということに着目しまして，チームをつくり，研究会をつくりまして，４９年の９月にはコンピュータ監査実務ガイドというものをつくって，当協会としていささかなりとも内部監査人がコンピュータ監査をシステム監査上できるようなことに努力をしようじゃないかということで，内部で専門家を呼び会社，企業の中でも大企業あるいは外国の合弁会社の中にもそういう専門家がおりますから，来ていただいてやってきたのであります。

しかし，何といっても日本のコンピュータの利用を私なりに解釈しますと，縦割りと申しましょうか，先ほど申しましたように経理部門とかあるいは生産部門とか資材部門とか総合的なコンピュータの利用というものが日本の場合においてまだ非常に不十分だ。これは何といっても非常に高価なものであります。リースをするにしても，非常に多額の費用をリース会社に払わなければならない。日本の場合におきましては，それを横断的な機能としてコンピュータの効率を図っていかなければならぬというのが今日的な問題であると思うのであります。

したがいまして，ここでわれわれ内部監査サイドから一番考えますことは，そういう非常に大事なこと，つまり経営者の意思決定をする，それをできるだけ短期間に意思決定をする資料，データを，いまコーディナーの言葉で言うならば，情報を収集し，これを伝達するというプロフェッショナルのマネジャーの機能を助けるものが内部監査人であり，そうであるならばそれはどうしてもコンピュータによらなければならない。その場合においては，コンピュータの機能なり，その知識なり，それのデータをつくるプログラマーなりの一連の知識がないことにはだめであります。したがって今日内部監査人として一番急務は，この人たちをどう育成していくかだと思います。このためには，私ははなはだ失礼でありますが，内部監査に従来属している中堅の人では，なかなか急に入りにくいと思います。したがってわれわれとして，いま内部監査協会として考えておりますことは，できるならば最初からコンピュータの機械のプログマーになったり，あるいはそれを動かしている知識のある方々を，内部配置を変えて，その人たちにコンピュータの知識がある人をまず最初に配置転換して，そしてその内部監査の方に回してもらう，トップの組織がえによって変えてもらう，その人を今度逆に経営管理なりあるいは総合管理なり会計管理なり生産管理という勉強をさせるということの方がより有効でないか，私はそう考えております。

要は，これから内部監査人の能力不足をどうやって解決していくのか，そして先ほどありました監査の要点である採算性の中心とか安全性の中心とか信頼性の中心とかあるいはプライバシーの保護とかというような問題がたくさんありますが，そういう問題にいくプ

レパレーションを内部監査人としての知識を養成するということにこれから最大の努力をしなければならぬ。つまり内部監査人の養成に力を入れていかなければならないのではないか，こういうふうに私は思っております。

いま内部監査人のサイドからということになれば，結論的には内部監査人がコンピュータに対する知識を持つ。それをどう育成していくか，その育成の方法としては，私は内部監査の人を育成することも一つの方法ですが，一番早道は，できるだけコンピュータの教育を受けた若い人たちを内部監査人の中に入れて，内部監査人としてインターナル・オーディターの育成をして，そしてその人たちに今度逆に経営管理なり，総合管理なり，生産管理なりそれなりのことを教える方がむしろ早道じゃないか，こういうことを考えております。

大変まとまりのないことを申しましたが，日本は先ほど申しましたように非常に大きな経済成長をしております。今後も成長していかなければならない。そこには人件費も世界でアメリカに次いでの高いものになってきたし，生産単位当たりの人件費もアメリカに次ぐところまで来たのですから，できるだけ少ない人間で効率的なものを生かす。それは先ほどエール大学の総長が言った，これからのエージはユースフルネスだ。つまり効率を上げることなんだ。これに最大のエージが来るのだということは，結局コンピュータ時代が来るということになる。それに対しては，内部監査人としてはできるだけそういうものが有効に使われているかどうか，機器の問題，いろいろな問題の監査をする前に，その育成をすることが日本の現在の最大の急務ではないかと思っておるわけであります。

堤　西野先生には，経営トップの立場，さらに内部監査人の立場からシステム監査の重要性を御指摘いただきました。特に日本のコンピュータ・システムは量はよいが，質はどうか，そういった意味では非常に厳しい愛のむちをいただいたわけです。

次は，ブラックボックスになりがちなシステムに対しまして，公認会計士の立場から上野先生にお願いします。

上野　西野先生のように広い立場からのお話ではございませんで，公認会計士という狭い側面から意見を申し上げます。

私はコンピュータの専門家ではありません。先ほど西野先生もおっしゃったように，公認会計士はややコンピュータに弱いという面が一般的だと思われます。

先ほど申し上げましたように，私は外部監査人である公認会計士という立場から，意見を申し述べる次第です。

趣旨は，内部統制制度の一環としてシステム監査人制度の整備，確立を望むということです。これは私ども日本のケースとして申し上げるものです。またアメリカについては別のやり方があるようですし，そういういろいろな考え方もあるようです。

前の講師の方からお話がありましたように，コンピュータを用いました情報処理システムの急激な発展，それから社会一般への普及はまことにめざましいものがあります。御高承のように，主要産業，企業はもちろん，中央，地方　官公庁，各種の学校，団体などにおいてそれぞれ業務処理に使用されているほか，各種の公共料金の計算や銀行のオンライン処理による預金及び出納事務などを通じてＥＤＰシステムは国民大衆の日常生活と密接な関係を持つに至っています。したがいまして，もしこれらのＥＤＰシステムの信頼性が失われ，重大なエラーが発生した場合に，その被害は大変なことになります。一企業，一機関のみにとどまらず，社会の広範囲の人たちにはかり知れない影響を及ぼす可能性がございます。

この問題，つまりＥＤＰシステムの信頼性の問題は，われわれ公認会計士にとっても重

大な関心を持たざるを得ない問題なのです。公認会計士は，企業が発表する財務諸表の適正性についての意見を表明するのがその職務でございまして，その意見表明のための合理的な基礎として監査証拠を求めて監査手続を実施しています。この監査手続の実施にあたりまして，ＥＤＰシステムが大きな影響を与えているのです。

その第一は会計記録のあとづけを行って，会計記録の正確性を確かめるための監査証跡が，いままでの手作業のときのように直接目に見える形のものから，コンピュータの場合には，コンピュータを用いなければ読み取ることができないものに変わったことでございまして，また事前に監査証跡を保存しておくような措置を講じておかない限り消滅をしてしまうという問題があります。公認会計士は，監査証跡を追跡いたしますことによって合理的な監査証拠を求め，それによって意見を表明するのですから，ＥＤＰシステムにあっては，従来のような事後の監査だけでは監査証拠を求めることができないという大変大きな問題となるわけです。と言いますのは，システムの段階からいろいろタッチをしなければならないという，事前監査という意味が含まれてくるわけです。

第二点は，公認会計士の行う財務諸表監査は，企業において，健全にして良好なる内部統制が存在していることがその前提になっていまして，その前提の上に立って公認会計士は示唆により監査証拠を求めることになっています。それでＥＤＰシステムにおきましてもこのような内部統制の存在が保証されているかどうかという点ですが，御承知のようにＥＤＰシステムの企業における導入は，一般的に言いまして事務の機械化という考え方から出発しております。これは人間の使うところのそろばんをただコンピュータに置きかえるという認識がその基盤にありまして，ＥＤＰシステムの特性とか事務処理の態様に与える影響について余り深く検討されずにその導入が行われた傾向があるように考えられます。それは先ほど西野先生も別の意味で，簡単に導入したのではないかというようなことをちょっと触れられましたが，たとえば手作業のシステムにおける相互牽制の機能がＥＤＰシステムではどのように変化し，どのような弱点を起こしているかという検討をＥＤＰシステムの導入の前に十分に行ったという企業はほとんどないのではなかろうかとわれわれは考えております。そしてそのＥＤＰシステムが高度化され，複雑化された現在においても，大勢というのはおそらくこういう傾向の延長線上にあるのではないかと感ぜられるのです。このことは，コンピュータは正確無比であり，絶対間違いを起こさないという観念から出発しているところにその原因があると思うのです。

ところが今日になってみますと，ＥＤＰシステムには人間の要因による影響がさまざまな形であらわれることがはっきりしてまいったのです。この人間の原因による影響，これが非常に大きいのです。コンピュータ自体はばか正直と言ってもいいくらいの正確性と高速性をもって業務を処理するすぐれた能力がありますが，このコンピュータを使っての業務処理の体系をつくり，データ処理の方法，内容を指示し，操作するのは，やはり人間が行うのです。そのＥＤＰシステムにかかわる人間にもしも不正な意図があれば，どのようなこともできるのですから人間の原因によって非常に影響が大きいということでございます。

アメリカにおきましてエクィティ・ファンディング社事件というのがございました。これは，２０億ドルつまり6,000億円に上るところの一大虚偽粉飾事件があったのですが，このほかにも多くの不正事例が表面化しております。われわれもこれを対岸の火災と見ているわけにはいかないのでございます。不正

の意図はなくても，たとえば不注意とか錯覚とかあるいは要員の疲労などによってＥＤＰシステムに携わる人間がミスをおかせば，当然ＥＤＰシステムにもエラーが生ずるのでございます。

われわれが痛感いたしておりますことは，このようなエラーや不正を防止するには，ＥＤＰシステムにおける内部統制の整備と充実をしていただくということ，そのことによってＥＤＰシステムの信頼性を高めていくこと以外にないのではないかということでございます。内部統制とはちょっと大上段ですが，誤りや不正を合理的にかつ迅速に発見し，防止する機能を経営組織の中に組み込んであることを指しているわけですが，ＥＤＰシステムには，ＥＤＰシステム特有のコントロール機能を備えていなければならないのです。これはいろいろなコントロール組織がございますが，これは後で何らか触れることがあるかもしれません。

内部統制の見地からＥＤＰシステムに必要な機能が備わっており，かつ十分な運用が行われているかということをシステム設計の段階から常時継続的に観察し，チェックを行い，あわせてＥＤＰシステムの採算性，生産性等についても同時に調査し，これらの結果を経営者に対して報告し，助言する役割りの存在が企業の内部においてどうしても必要であると考えるのでございます。これらの役割りを担当する人たちが，私どもといたしましてはシステム監査人ということですが，われわれといたしましては，このシステム監査人制度の整備，確立というものがＥＤＰシステムの信頼性保証のために欠くことのできないものであると思うのです。これはアメリカ等におきましては若干異なる考え方もあるようですが，私どもはさように考えておる次第でございます。

それから責任の問題ですが，ＥＤＰシステムに関する責任というものは，第一義的には企業の経営者にあるのでして，もしコンピュータによる誤りや不正事件が発生したとしますと，経営者がその責任を問われることになるわけですから，この基本的責任を全うするためにもＥＤＰシステムの信頼性を確保すること，すなわちＥＤＰシステムの内部統制を整備，確立することを真剣にお考えいただきまして，システム監査人制度を具体化なさることを，各企業の実情に応じて早急な実現の方向で御検討いただき，一日も早い整備，確立を心から望むものでございます。

どうもお願いのような意見でございますが，以上申し述べた次第でございます。

堤　公認会計士の立場から大変明快な問題の提起をしていただきました。

次は，巨大なシステムを運用しておられ，また大衆との接点も多い島川先生にユーザーの立場から問題の提起をしていただきます。

島川　私は銀行の中におきまして全般的な経営計画を立案しております企画部を担当しておりまして，企画部は全般的な経営計画と，さらに収益というものも担当しております。それからシステム開発部と申しまして，本日のテーマに非常に関係のございますコンピュータを中心といたしました開発部門を担当し，かたがた銀行のオペレーション部門のコントロールというものの所管をいたします事務管理部，さらにはコンピュータの運営，管理の基本的な責任を持っておりますオペレーション・センターを担当しておるわけでございます。別にこれを担当しているからと申しまして，私が非常に詳しいというわけではございませんで，これからお話し申し上げますことも大変表面的なことなのですが，しばらくの間御清聴いただきたいと考えます。

私はシステム監査，特にコンピュータ監査とトップ・マネジメントの責任についてお話をしたいと思います。

私はここでシステム監査，特にコンピュータ監査というふうに言葉づかいをやや意識的

に分けてございますが，銀行の場合はコンピュータ・システム以外のシステムがたくさんございますので，やや言葉づかいには神経質になっているわけでございます。それはともかく，私の話は銀行という特殊な分野，それも富士銀行というきわめて限られた分野から見た考え方で，しかも多分に私自身の個人的見解が入っておるということをあらかじめお断わりを申し上げた上でお話を進めてみたいと考えます。

まず第一に強調いたしたいことは，おそらくどの銀行もそうであろうと思いますが，富士銀行におきましてトップ・マネジメントの主要な関心事の一つは，早急にコンピュータ監査システムを企業内にビルトインし，これを充実していかなければならないということでございます。

この理由は，申し上げるまでもなく銀行の膨大な営業システムあるいはこれをコントロールする幾つかの管理システムのほとんどすべてがコンピュータによって支えられているのでありまして，そこでは銀行自身の資産，負債のすべてのアクティビティあるいは顧客と銀行との間の債権，債務のすべてのアクティビティというものがコンピュータによって処理されていることはもちろんでございまして，さらには経営の意思決定のためのサポート資料もまたコンピュータによって確保，提供されているのでございます。言うなれば，今日ではコンピュータ・システムは経営管理システムの中枢的存在となっておると申しても過言ではないのでございます。

ところで，銀行経営のよって立つ基盤と申しますものは，顧客や株主，社会からの信頼性をかち得て初めてこの基盤が強固なものになり得るという，言うなれば伝統的な経営原則というものは，今日でもいささかも変わりはないのでございます。むしろ今日の銀行をめぐる諸情勢すなわち取引の大衆化の進展ですとか安定成長下の取引先の内容の管理，あるいはまた高まる銀行批判の社会風潮，こういう中でトップ・マネジメントとしては，いろいろな営業システムないしは管理システムにつきまして，従来にも増して高度の信頼度を確保しておくことが経営的に絶対に必要になってきつつあるわけです。このことはまた換言いたしますと，これらのシステムを支え，あるいはこれらのシステムを可能ならしめておりますコンピュータ・システムの信頼度が，銀行への信頼度そのものに重大な関連を持つことを意味するわけでありまして，トップ・マネジメントといたしましてもこのことを深く認識せざるを得ないのであります。そういう意味で，まさに今日的な問題だということができると思います。

このコンピュータ・システムについての信頼度を確立すること，すなわちコンピュータ監査の重要性についてのトップ・マネジメントの認識は，実はただいま申し上げたところから始まっているのでありまして，言葉づかいには多少恐縮なところもあるわけですが，決して監査役制度の強化とか公認会計士制度の導入があって防衛的にコンピュータ監査というものの問題を取り上げたものではないのでございます。

もとより監査役制度あるいは公認会計士制度は，コンピュータ監査の重要性を加速するものであることは当然ですが，いま申し上げましたようにコンピュータ監査は何よりもトップ・マネジメントが先ほど申し上げました伝統的な経営原則と今日の客観情勢の中で問題を主体的に取り上げているのだということを私はまずもって強調したいのです。

このようなトップ・マネジメントの主体的な認識によって取り上げられつつあるコンピュータ監査で，具体的にそれでは何が問題になるかと申しますと，私は二つあると考えております。

その一つは，いかなる体制，組織でコンピュータ監査を行うかということが第一点。い

かなる点に重点を置きながら，いかなる手法で監査を進めていくかということが第二の点でございます。私見によりますと，第二の点は今後相当の期間をかけて，試行的な態度を持ちながら開発，定着させていくべきものであるというふうに結論的には考えておりますので，トップ・マネジメントとしてまず着手すべき点は，内部にいかなる組織，体制をつくるべきかという問題から始めるべきだと考えます。

以上のような観点から，富士銀行内でのコンピュータ監査体制の現状と考え方について述べてみたいと思います。

当行におきましてコンピュータ監査と関係の深いファンクションを持つものは幾つかございまして，第一はシステム開発部門，第二はオペレーション・センター，第三は営業店，第四は検査部，その次が監査役，それから公認会計士，これらの諸機能がコンピュータ・システムの監査と深いかかわり合いを持つわけでありまして，内部的なコンピュータ監査と組織というものがこれらのファンクションといかなる関係を持つかが組織上の問題となるわけです。申し上げるまでもございませんが，およそ体制ですとか組織というものを考える場合には二つの原則，すなわち組織論から見て妥当なものであること，及びそれらがプラクチカルなものであること，この二つの言ってみれば組織論の原則の調和を求めながら組織をつくることが必要でございます。

以上の考え方に基づきまして，当行の内部におきますコンピュータ監査組織がつくられておるわけですが，具体的にはシステム開発部門にコンピュータ監査のための専門スタッフを数名発令し，独立の部門として部長に直属させているということがまず申し上げられると思います。同時に彼らは検査部兼務の発令となっておるということが特徴でございます。さらにこれらのスタッフ全員はいずれもコンピュータの経験者でございまして，プランナーないしはプログラマーあるいはまたオペレーターといたしましても，いずれも１０年以上の経験を有しております非常にすぐれた，かつベテランということでございます。

私はここで申し上げたいのは，およそ監査というものは本来第三者的立場から行うことが望ましい，かつそれが原則であるということも承知しているわけでありますし，おそらく皆様方も本来監査の対象でもあるべきシステム開発部に監査スタッフを発令しておるというのは理解にお苦しみになるかもしれません。しかし私どもの場合，新しい監査スタッフにとりまして与えられた命令，すなわちコンピュータ監査システム確立のために必要な諸情報，必要な協力ということが何よりも大事なのでありまして，そのためにはシステム開発部門にいることが一番よいのだというふうにきわめて現実的な考慮をいたしまして発令したわけであります。言うなれば先ほどもちょっと御紹介申し上げましたように，これらのスタッフは１０年以上のコンピュータの経験者でありますから，昔の同僚，友人がたくさんいる部門内に独立のグループとしてまず働き出した。こういうことでございます。したがいまして，必要な諸情報，必要な協力がきわめて得られやすい，こういう状況にあるわけでございます。ただ，あくまで第三者的ということが私は必要な条件であろうと思いますので，検査部の発令を同時にしておるわけでございます。またそうした制約の中でシステム開発部門の中では独立のグループとして位置づけをし，部長の直属にしておる，その程度のくふうはしておる，こういうことでございます。

そこで大事なことは，こうした現在のコンピュータ監査体制と将来の姿との関係でございますが，私見では，将来的にこれらのスタッフが十分成長したとしましても，全員検査部に移籍すべきであるとは考えておりません。これが私どもの特徴の一つでございます。す

なわちおそらく富士銀行におきましては，将来の姿といたしましてコンピュータ監査体制は二元的存在，すなわちシステム開発部門内に独立したグループとして存在しておる，と同時に，検査部にもまた数名の監査スタッフが存在することになるはずであるというふうに私は考えております。将来ともシステム開発部門内に独立した監査スタッフを置くというのは，以下申し上げる理由によるからでございます。

すなわちシステム開発部門というのは，絶えず新しいシステムを生産し，既存のシステムをレビューし，改善する責任を帯びておる部でございます。しかも皆様方おそらく同意見だと思いますけれども，開発というものは単に新しいものを考え出すとか言い出すということではなくて，みずからの責任において監査にたえ得るものをつくり出すのが，真に開発という名前にふさわしい職責の果たし方であるというのが私の考えであります。また実際問題としても，常時システムの改善，くふう，変化が行われるところにある程度監査スタッフがいなければ，真にタイミングのよい連絡協調は期しがたいと考えるのです。これが開発部門の中に監査グループを将来とも置く基本的な理由でございます。

もちろんこの部門の中ではできるだけ独立したグループにすることが望ましく，新しいシステムを生産する場合には，部門の長は必ず監査グループの分析，評価を求めることが必要です。また場合によってはこれは多分に私見でございますが，部門の長は企業にとりましてきわめて重要なシステムについては，監査部門に直接ソフトを作成せしめる場合があってもよいということを申し上げておきたいと思います。これは銀行業務の特殊性によるかもしれませんが，一般的な業務開発グループではなくて，全体のシステムの中で監査グループに直接プログラムをつくらせる，そういうケースがあり得ていいのだと考えているわけでございます。

次に，検査部に置かれるべきコンピュータ監査スタッフは，システム開発部門で育てられましたスタッフの一部がやがて移籍され，さらには若干増員されるはずでございます。彼らの仕事は，言うなればルーティン化されました監査手法によりまして具体的にコンピュータ・システムの運営，管理について中核的な存在になっておりますオペレーション・センターを監査するということが主要任務だと考えております。

なお，当行では，いわゆる内部監査役に対しましては，検査部が緊密に協力し，必要な資料を提出するということが規定で定められておりますから，検査部によるコンピュータ監査は監査役のためのコンピュータ監査ということにもなるわけでございます。また言うまでもなく検査部は頭取に直属をいたしまして，全く独立のスタッフ部門であることは申し上げるまでもございません。

検査部のコンピュータ監査スタッフの仕事の中には，先ほど申し上げましたシステム開発部門の監査スタッフの仕事がトップ・マネジメントの期待どおり部門内で厳格に運営され，所期のとおりワークしているかどうかというものを監査する仕事も含まれるということは当然でございます。

次に，当行の場合，200を超える多数の支店すなわちライン部門があるわけですが，これらの支店は端末器を利用する上での，あるいは異例ケースのための監査システムがあらかじめ与えられておりまして，さらには幾つかの自己監査システムが義務づけられておりますので，一般的に検査部の検査対象ではありますが，いわゆる私がここで申し上げておりますコンピュータの監査グループの監査対象とは考える必要はない。むしろ異常なケースを営業店が発見した場合には，逆に監査グループないしは検査部に新しい情報を提供する義務がある，こういうふうに考えておりま

す。

このようにシステム開発部門，オペレーション・センター，検査部，営業店等コンピュータ・システムに関連のある部門とコンピュータ監査体制との関係について述べたわけですが，要約しますと，検査部という独立の部門にコンピュータ監査スタッフを置くだけではなくて，システム開発部門にも独立したグループを置きまして，二元的組織にするというのが私の考えでございます。

なお，公認会計士の監査とコンピュータ監査との関係もきわめて重要な問題でありますが，私は銀行内でまずトップ・マネジメントが主体的に充実したコンピュータ監査体制を内部的に確立していくことによりまして，公認会計士の方々との間の信頼関係は一層緊密なものとなり，問題の解決を容易にするはずだということを申し上げておくにとどめたいと思います。

次の問題は，トップ・マネジメントはコンピュータ監査グループに重点的な監査内容といたしまして何を期待するかということでございます。およそトップ・マネジメントは，コンピュータ・システム全般につきましては常に効率性，顧客ニーズの満足度，安全性という観点から重大関心を持っているわけでありますが，コンピュータ監査との関係におきましては特に安全性に置くべきであると私は考えております。当然のことですが，このことは他の項目の重要性が経営的に見てより低いことを意味するものでは毛頭ございません。むしろ富士銀行におきましては次のような措置が講じられているから，重点を安全性にしぼられるというふうにおとりいただきたいと思います。すなわち効率性ということにつきましては，システム開発部門内に通常の業務開発グループとは別に独立のグループとしましてソフト，ハード，回線等システム全般の効率化に専念するグループを置いておりまして，さらに権威ある外部の専門コンサルタントに効率化のためのアドバイスを定例的に求めておりまして，トップ・マネジメントがみずからそのアドバイス，意見を聞くことにしておるのであります。もちろん行内のコンピュータ監査グループからも効率化についての助言，意見は期待もし，歓迎もするわけでありますけれども，主たる任務は安全性ということで見てほしい。すなわち効率化その他は言うなれば助言ないしはコンサルテーションの範囲というふうに考えておるわけでございます。

次に，顧客のニーズの満足度という点につきましても，これもまた別部門にサービス監査室というものを設けて，銀行全般のシステム，コンピュータ・システムも含めまして顧客からの苦情，不満，助言を直接，間接に聞き，そのつど必要な措置を講ずることとしております。そしてトップに遅滞なくその情報が提供されるような仕組みにしておるのであります。これもまた行内的なコンピュータ監査グループの主要任務と位置づける必要がない一つの理由でございます。したがいましてトップ・マネジメントはコンピュータ監査にあたって安全性に重点を置くという理由がおわかりいただけたと思います。もちろん言葉の使い方でございますから，先ほど私が申し上げました効率専門のグループとかサービス監査室のグループ，こういうものを全部ひっくるめてシステム監査であるという言葉の使い方もあると思いますけれども，ここでは私はコンピュータ監査というものをやや狭義に，厳格に解釈をしておる立場に立っているものでございます。

以上のようなことで，安全性ということに重点を置くべきことがおわかりいただけたと思いますが，なおこの機会に富士銀行のコンピュータ・システムのアウトラインを一言だけ申し上げておきますと，オンライン・システムによりまして全店取り扱い可能な預金システムをやっております。またキャッシュ・

ディスペンサーによります広範囲な払い出し，あるいは営業時間外の払い出しサービス，公共料金を初めとする各種の自動支払いの制度，為替の全国銀行ベースのオンライン・システム，その他外国為替，貸し出し，バランスシート，損益計算書等のオンライン処理など広範，多岐にわたりましてコンピュータで処理しておりまして，１日の平均事務量は預金だけで１２０万～１３０万件。為替で約３０万件，しかもこれらが行内の端末器だけでも3,000台弱ということでございます。したがいましてトップ・マネジメントといたしましては全体のシステムの安全性にこの面からも最重点を置くということがおわかりいただけると思います。

もちろん安全性のための監査手法というものも多様でございまして，オペレーション部門の組織，規則，建物管理，データの保管管理，ハードウェア，ソフトウェアの安全性等監査手法の開発は今後の努力にまつべきものが多いわけでありますが，トップ・マネジメントといたしましては，これらの監査グループの実効ある努力を大いに期待しているわけです。しかし担当者によりますと，たとえばソフトの安全性についてはかなりの期間かかるということも聞いておりますので，私どもといたしましては組織や規則あるいはセンターの運営，オペレーションの管理等々できるものからまずやっていって，長い時間をかけて進みたいと思っております。

私どもといたしては，基本的な認識がトップ・マネジメントが主体的にこの問題の重要性を認識するというところから始まっておりますので，コンピュータ監査にかかります体力，時間というものについては彼らに協力を惜しむべきではないと考えておるわけでございます。

以上，当行のコンピュータ監査を頭におきながら，一般の書物その他に出ておりますこととかなり言葉の使い方あるいは問題の処理の仕方というものを特殊性を持たせながら私の考え方を大胆に申し上げたわけでございますが，これも皆様からの御批判を率直にいただきたいというのが趣旨でございます。いずれにいたしましても，トップ・マネジメントが現在の社会風潮，銀行業務の膨大化とコンピュータリゼーションという中で主体的に問題を取り上げることこそ，トップ・マネジメントの経営責任を果たすゆえんであることを強調したつもりでございます。

堤　それでは商法におきましても大変強化されております常任監査役の問題などございますが，そういった監査役の立場からシステム監査についての御意見を石田先生からお願いいたします。

石田　私は監査役の立場から，システム監査をどういうふうに考えるかということを三点ほど申し上げたいと思います。

第一は，先ほどからお話が出ておりますけれども，システム監査を一体どう考えるか，それから第二点は，監査役監査とコンピュータ・システムあるいはＥＤＰシステムがどうかかわり合うか，それから第三点は，監査役の監査の場合の監査基準は一体何であるのか，この三点を申し上げたいと思います。

本日のテーマでありますシステム監査は，先ほどソルトマンさんその他の先生方からお話がございましたように，大変広い内容を含んでおります。公認会計士の方がお話しになりました内部統制組織というのも，英語に引き直せばインターナル・コントロール・システムの監査になるわけです。また島川さんからお話がありました営業システムだとか人事システムあるいは購買システムというような縦割りにいたしました業務の監査もあり得るわけでございますし，これから触れてまいります会計監査というのはいわば会計システムの監査になるかと思います。

本日のテーマは，もちろんＥＤＰあるいはコンピュータ・システムの監査ですが，これ

をＥＤＰのシステムに限定いたしましても，なお監査を考える場合に二つの限定が必要になるように考えます。先ほどソルトマンさんがお話しになりましたＥＤＰシステム，最後の図面に出てまいりましたが，監査する側面は非常にたくさんあるわけでございます。そのどの側面を監査するのか，あるいはＥＤＰのどの属性に着目をするのかというのが第一でございます。それから第二は，判断あるいは評価を行いますのに一体どういう基準を用いるかということでございます。

最初に第二の基準の方から申し上げますが，監査を行うにあたりましては，事の当否，よしあしというのは何か一つの基準に照らして判断がされておるわけです。たとえば公認会計士の方がなさいます会計監査におきましては，一般に受け入れられた会計原則が基準になっておりまして，その会計原則に対して会計処理がどの程度一体適合しているのかを証拠によって調べるという仕組みになっているようでございます。ところがこの基準というものの立て方は，決して一とおりではございません。先ほど島川さんから幾つかの点を指摘されましたが私なりに分類をしてみますと，三つに一応分けられるかと思います。

一つは，制度として確立しておる基準でございます。その中は，たとえば法令，定款，株主総会の決議に従って行われておるかどうか，これは一般に適法性の基準と言われております。それから第二番目は，会計原則あるいは公正なる会計慣行に従って会計処理が行われているかどうかという場合の一つの基準でございまして，これが一般に公認会計士の方々が適正性の基準と呼んでおられるものでございます。それから第三番目は，社内規定あるいは諸手続につきましてどれほどそれに準拠しておるか，内部統制の問題が一つだと思いますが，いわば社内規定の準拠性あるいは遵守性を基準として取り上げるという立場があるかと思います。適法性の基準だとか適正性の基準はいわば法制制度として確立しておる基準でございます。

それから大きい分類の第二は，経済性の基準でして，これも先ほどからお話が出ておりますように，経済性だとか効率性だとか合目的性というふうな基準でございます。実はこれは先ほど第一番目に申し上げました確立された基準とはちょっと認めがたいわけでございまして，ケースにより，あるいは人によりましてその判断基準が異なっておるように考えております。

それから大きい分類の第三番目は，これは仮につけた名前ですが，社会性の基準というふうなものが考えられるかと思います。会社の社会的な責任ということが言われておりますが，その社会性が何であるかにつきましては的確な説明はできませんが，そしてまた社会的にそのコンセンサスが得られた定義というものはないように存じますが，いわば一種の社会責任性というふうなものが一つの判断基準になる場合があるように思います。先ほどの島川さんのお話で申し上げますと，たとえば銀行の顧客のニーズというふうなものは，あるいはこのカテゴリーの中に入るかと考えております。

いま申し上げました基準のどれを選ぶのだということを頭に置きながら，ＥＤＰＳの一体どの側面を監査の対象にするのかという第一の問題に移ります。

本日のテーマは，システム監査という頭にたとえばＥＤＰだとか営業システムだとか頭に冠詞あるいは接頭語なしのテーマになっておりますが，接頭語をつけましたシステム，それも先ほど引き合いに出しましたＥＤＰ会計のシステムを取り上げてみることにいたします。この場合，会計監査という場合のその対象はＥＤＰのアプリケーション，その中の一つでございます会計の流れあるいは会計の処理のその運行面，あるいはそのＥＤＰの働きに着目をした監査であるというふうに考え

ております。その働きに対して信頼が置けるか，あるいは正確であるかの判断を，要するに判断基準を何に求めているかと申し上げますと，会計上の基準あるいは言いかえれば商法の帳簿計算規程あるいは会計原則に従ったようなアウトプットをＥＤＰＳがつくり出しているかどうかということを問題にしておるのだというふうに考えられるわけでございます。しかしＥＤＰ会計システムの監査と申し上げましても，別の考え方もあるように思います。

それで，会計の機械化をいまから始める，あるいはすでに始めたけれども，一体これで人手から機械に移すこと自体を検討しようという態度でございます。コンピュータの多数ございますアプリケーションの中で会計の機械化を選択したことの効果あるいはその機械原価の見地から，言いかえますと先ほど申し上げた経済性の基準からＥＤＰ会計士等の監査をすることも可能でございます。もちろんこの検討の過程におきましては，先ほど申し上げたいわゆるＥＤＰ会計監査と言われる場合の正確性あるいは信頼性という要因も考慮されるはずですが，二つの視点というのは異なっているように考えられるわけです。

ついでございますが，かつてすでに十何年前でございますか，コンピュータ会計をやること，少なくとも財務会計のコンピュータ化をやることは意味があるとかないとかという議論がかなり行われたことがありますけれども，これも会計の機械化という一つの事象，事柄をつかまえまして，それを経済性の基準で判断をした一つのケースだと考えております。

次に，ＥＤＰからつくり出されるアウトプットですが，そのアウトプットあるいは情報を対象にした監査も考えられるわけです。コンピュータのアウトプットが果たして情報の要求あるいはタイミングだとか頻度だとかあるいはさかのぼりまして先ほど西野先生のお話がございましたように情報の基本的な用途，たとえば意思決定に一体役に立っているかどうかということを検討しようというそういう検討の考え方もあるわけでして，ユーザーあるいはＥＤＰ部門で行われますアウトプットの点検運動というようなものが行われるかと思いますが，それも一例であろうかと思います。

もう一つ別な例を挙げますと，これは先ほどからお話が出ております企業財産の保全という観点から，いわゆるセキュリティの監査と言われるものがあるわけです。

以上のように，ＥＤＰの監査と申し上げましても，要するに対象をどの側面に着目するか，その側面が大変多様でございますし，それから仮に基準は三つ挙げましたけれども，その基準も実は多岐にわたっておると考えております。私は三つ挙げましたが，先ほどの島川さんのお話で言えば，安全性の基準というものも重要なものとしてつけ加える必要があると思います。いずれもこれは経営全般にとりまして，あるいは経営管理にとって大変有用な，あるいは欠くことのできない管理，統制手段であることはもちろんでございますが，その全部が一体監査役監査の対象になるかどうかということになりますと，これは私自身は疑問に存じております。新商法におきましては，監査役は業務監査を行うということになっており，この業務の解釈次第では，先ほど申し上げました，あるいはソルトマンさんがおっしゃったすべてが監査役監査の対象になるという見方ができるかもしれません。

そこで，話を多少基本的なことに戻して，監査役監査と一体コンピュータあるいはＥＤＰとはどういうふうにかかわり合うかという点を考えてみたいと思います。

監査役の職務は，商法２７４条で，「監査役ハ取締役ノ職務ノ執行ヲ監査ス」と包括的に規定されております。そしてこの職務には，取締役会ないし取締役の行うすべての行為を

含むことになっているわけでございます。このわずか１７文字の包括的な規定は大変抽象的で，漠然としており，何をしてよろしいかこれ自体ではわからないわけでございますが，商法は他の条文でもって監査の個別的な指示を与えております。

この指示を監査の対象別に分けて見ますと，一つは，取締役の行います行為そのものでございまして，取締役の法令，定款違反行為と不正行為とを取り上げております。法令，定款に違反する重大な事実あるいは不正行為はたとえば監査報告書に記載しろ，あるいは法令，定款違反行為で著しい損害を及ぼすときは行為の差止請求をしろというようなことを言っております。実際問題としてはまず起こりそうもない事柄でありますけれども，ＥＤＰに関連した取締役自身の，あるいは部下を指揮しての違法または不正行為ということも全然考え方として考えられないわけではございません。しかし仮にこの不正行為が行われたとしても，結果は会計事実としてあらわれるのが普通でして，これから後で申し上げます書類の方でございますが，会計報告の中におのずから表現されると考えられます。

第二番目の対象の分類は書類でございます。株主総会に提出します書類，議案で違法な事実もしくは著しく不当な事項を認めたなら株主総会で意見を報告しろと言っております。この株主総会に提出いたします書類，議案でございますが，この中で問題になりますのは，ほとんど書類，しかも計算書類でございます。計算書類はいわば決算書類でございますが，御存じのように貸借対照表，損益計算書，営業報告書それから利益配当の議案四つを毎期提出するようになっております。ＥＤＰに直接関係いたしますのは前の三つ，つまり貸借対照表，損益計算書，営業報告書でございます。貸借対照表，損益計算書は法令，定款に従い，会社の財産，損益の状況を正しく示しているかどうか，あるいはこれをつくるにもとになりました会計帳簿について記載漏れはないか，あるいは不実の記載はないか，あるいは貸借対照表，損益計算書と帳簿とが一致しているか，つまり正しい会計報告であるかどうかを見ろというふうに言っているわけでございます。この一連の会計プロセスの全部もしくはいずれかのステップにＥＤＰがからんでいる，あるいは貸借対照表，損益計算書がＥＤＰからのアウトプットだといたしますと，一応監査役の監査とかかわり合いがあるということになるわけでございます。

報告書の書類の中のもう一つのものは営業報告書です。これの内容が真実であるかどうかを報告しろと言っているわけですが，営業報告書に記載されております営業の概況等を示します数字の中で，もしＥＤＰＳからのアウトプットがあるといたしますと，その限りではこのコンピュータ・システムとかかわり合いがあるわけでございますが，一般にはこの数字は会計報告に含まれるのが普通だと存じますので，会計報告と突き合わせをすればその成否はチェックできるわけでございます。その意味では，間接的にはやはり会計の問題としてとらえられるのではないかというふうに考えるわけでございます。

さっきもちょっと触れましたように，監査役の業務監査の範囲につきましてはいろいろ解釈がされておりまして，いろいろな意見が出ているのですが，商法が明文をもって規定しておりますその監査役の職務というのは大体以上に要約できるかと思います。

説明申し上げましたとおり，この商法が言っておる文脈に従って監査をいたします限り，監査役の監査はおのずから限定をされ，ＥＤＰシステムのすべてのフェーズあるいは側面ではございませんで，ＥＤＰ会計にしぼられてくると言わざるを得ないように考えるわけです。もっとも商法で要求している会計監査というのは，貸借対照表，損益計算書とシステムがつくり出すアウトプットあるいは会計

報告に対して監査をしろと言っておるわけでして，システム自体を監査しろと言っているわけではございません。しかし申し上げるまでもなく，そのアウトプットの成否を検証するためには，これをつくり出すプロセスあるいはメカニズム自体も調査の対象にせざるを得ないわけでして，製品の品質はインプットの質と同時に，プロセスあるいは加工の良否によって決まるわけでして，会計プロセスの中でどれだけＥＤＰ化されておるかＥＤＰ化の割合の大小はあるにいたしましても，ＥＤＰ化されておる限りでは，そのシステムそれ自体がやはり監査役監査の対象になるというふうに考えておるわけでございます。

そういたしまして，対象をいま申し上げたわけですが，もう一つの基準について申し上げれば，そのシステムが適法な会計報告をつくり出す仕組みになっているかどうかを問うことになっているわけでございまして，間接的に申し上げれば，上野先生が御指摘になりましたように信頼性を監査することになるかと考えるわけです。

以上私の申し上げたい結論でございますが，なおもう一つつけ加えておきますと，このシステム監査ですが，監査役の立場で言えば，ＥＤＰ会計の監査の方法をどうするのか，要するにＥＤＰ会計のシステム監査の方法はどういうふうにすればよいかというのが次の検討点になるわけでございます。

監査役の実務にとりまして一番解明を要する点でございますけれども，申し上げようとしておるのは，その前に手作業であろうと，ＥＤＰ会計であろうと，監査役の行う監査というのは一体どういう基準によるのかということです。商法では，業務監査は別として，会計監査については資本金の大小にかかわりなく監査役が行うことになっております。ただ，資本金５億円以上の会社では会計監査人の監査も受けることになっており，その場合，監査役は会計監査人の監査を評定して，不満足であれば，商法の言葉で言えば，相当と認めなければということでございますが，自分で監査しなさいというふうに言っております。その場合に相当であるかどうかを評定するためには，監査役が自分で監査してみなければわからないじゃないかという考え方もございます。

いずれにしても，ここで大変重要な問題が出てまいります。監査役の会計監査のあり方あるいは方法は，会計監査人つまり公認会計士がやっておられましたのと，同一の基準に基づいてやるべきものであるのかどうか，つまり公認会計士の方がよりどころとしておられます監査基準あるいは監査実施準則というものがございますが，一体監査役もそれに基づいてやらなければならないのかどうかという問題でございます。この一点，監査役にとっては大変重要な問題でありますけれども，どうもあまり明示的な解釈というのは下されていないように思います。仮に公認会計士がおやりになります監査基準準則によるといたしますと，監査役のＥＤＰ会計の立場とそれから公認会計士のＥＤＰ会計の監査の方法はやはり同じ文脈に従って組み立てられることになるわけでございまして，その意味では監査役それから公認会計士共通の問題となると考えますので，この点一点だけ指摘をしておきたいと考えるわけです。

結論は，簡単なことを大変回りくどく申し上げて恐縮でございますが，一応監査役の立場から見たシステム監査というものについて私見の大要を申し上げたわけです。

堤　これで日本側パネリストの発言は一応終わったわけでございますが，ここでソルトマンさんにコメントをお願いしたいと思います。

ソルトマン　ただいま４人の方が発表されましたが，それぞれの分野で専門家の方々であり，そしてきわめて興味深い点を指摘されたと思います。

ＥＤＰ監査の概念について皆さんが指摘されましたなかで，私の観点からこれらの話のうち特に重要だと思うことを繰り返し私のコメントにかえたいと思います。以下の点が皆様方の指摘の中で重要だったと感じました。

まず第一番目にコンピュータを長期計画に利用するという点があったわけでありまして，このコンピュータのアプリケーションはさらに改善しなければならないということがあったと思います。米国におきましては，政府がこの数年間コンピュータを利用し，そしてコンピュータにベースを置いたモデルを利用してきたわけでございます。米国のさまざまな省庁におきましてコンピュータ・ベースのモデルを大幅に利用してまいりました。その結果，投資の額が大幅になりまして，ゼネラル・アカウンティング・オフィスはモデルの利用について報告書を作成するに至ったわけであります。会計監査院の結論は，もっとモデルの入手，モデルの開発，そして効果的なモデルの利用についてもっと努力をしなければいけないという結論でありました。というのは，会計検査院はその結論として，モデルが投資しただけの成果を挙げていないということを言っているわけであります。

ですから，コンピュータのモデル化ということは，コンピュータの利用としては非常にすぐれておりますけれども，もっとマネジメントのコントロールが必要であるという結論であります。この分野におきましては，コンピュータ監査人が非常に効果を上げることができるわけであります。そしてモデルに対するマネジメント・プランニングを再検討し，そして少なくとも投資に対する見返り率をある程度のモデルに対する投資からどの程度得られるかというガイドラインをつくるべきだと思います。そしてこれは監査後に必要になってくるプロセスになるわけです。そういうガイドラインを大きな組織において提供すべきだと思います。特にモデルを利用しようとしておる組織に対してガイドラインを与えるべきだと思います。

またデータとしてはたとえばどの程度の時間をモデルにかけるべきか，そしてモデルがもはや効果的に利用されていないということがわかったときに，どの時点で廃止すべきかということであります。モデルの利用は，実際行われたことよりも，これこれのことが可能であるという期待の方が強かったというふうに思うわけでありまして，これについてはさらに慎重に検討することが必要だと考えます。

もう一つ私の頭の中に浮かんでまいりましたのは，コンピュータ監査とシステム監査の間に違いがあるのかという問題であります。この意味で，システムというのはいろいろな見方ができると思うのですが，たとえばシステムというのは，セールス・システムとかパーチェス・システムとかという考えを持つことができるし，そしてまた会社の業務の一つの側面というふうに考えることができるわけであります。

また，もう一つのシステムの考え方としては情報システム，つまり情報が入ってきて，そしてシステムが情報から出ていくところまで情報システムと考えることができるわけであります。また少し違う点としてコンピュータ監査を考えることができるわけですけれども，これはコンピュータとの関連を維持して，コンピュータ化されない情報システムについては除外をするという考え方であります。ですから，いまのところまだコンピュータ監査とシステム監査が同じかどうかという点について十分な定義がないわけでありまして，また監査人がどういうシステムを取り扱っているかという定義もないわけです。これに対して私は答えは持っておりませんが，いまのところいろいろな方向に進展をして，できるだけ可能性と合理性の高い方向に行くということであると思います。

また，指摘された中でもう一つ重要な点は，二人のスピーカーが指摘されたと思いますが人々の考え方におけるクレディビリティ（確実性）という言葉を一人の方が使われ，もう一人は信頼の失墜とおっしゃいましたけれども，結局概念は同じだというふうに思うわけであります。効果的なコンピュータもしくは非能率的なコンピュータの利用による信頼性の増大ないしは低下ということが非常に重要なポイントだと思います。

米国におきましては，いままでコンピュータ監査に対する力強い動きがあったわけでありまして，私個人もまたいろいろな研究を行い，コンピュータ化された投票方法という点についての報告書を出したわけであります。米国では，いくつかの地方都市におきましてはパンチカードを使って投票をしているわけでありまして，そしてパンチカードをコンピュータの中に入れて投票を計算することになっております。それでときにはそういうシステムがうまく動かないことがあるわけでありまして，そしてそういうシステムに対する大衆の信頼感が失墜しているわけであります。そこで標準局の報告は米国の連邦選挙委員会に出したわけでありますけれども，その報告の中で，これらのシステムがどの程度うまくいったかという監査報告を出し，そして内部コントロールを改善するための提言を出したわけであります。これはノンファイナンシャル・システムのよい例でありまして，監査の技術をもっとこういった分野にも利用していかなければいけないという一つの例だと思います。大衆のクレディビリティとコンフィデンスということはコンピュータのベースになるべきものだと思います。

また，もう一点二人の方がおっしゃったテーマに関連してくるわけですが，外部の監査は，内部監査に対する信頼性に依拠するということであります。外部の監査人は，まず内部監査のコントロールの効果性を知りたいと思うわけでありまして，それにのっとって外部監査が行われるわけであります。

もう一つ指摘された点として，監査人というのはシステムの開発とインスペクションという二重の役割りを担っているという点であります。この問題について，私ごく最近も考えたわけでありまして，この問題について内部監査の独立性という報告がアメリカ・インスティチュート・オブ・インターナル・オーディターズから出されたわけであります。アメリカ人の監査人の意見というのは，インスペクション・デパートメントはあらゆる意味において独立性を保持しなければいけないという点であります。これが問題になるのですけれども，インスペクション・デパートメントの中にしか監査についての専門知識がない場合，そしてまた内部コントロールとそれからシステム開発に関係する場合に問題が出てくるわけであります。アメリカでもこういった考え方はよくないというふうに現在では考えられております。つまりシステムの開発に関する監査に関する専門知識というのは，そのコントロールの効果を評価するような監査のエキスパーティズとは分離しなければいけない，その両者をそれぞれ独立させておかなければいけないというのが最近のアメリカの考え方であります。

それから最後のポイントとして指摘されたのですけれども，ＥＤＰのアウトプットの適正さは，処理の質に依存をするという点であります。評価するというよりは，皆様方がおっしゃったことを振りかえってみまして，日本のマネジメントは監査という問題については，アメリカのマネジメントも同じように手さぐり状態にあるのではないかという感じを抱いたわけでございます。その意味で，このシンポジウムは私にとって非常に貴重であります。というのは，このシンポジウムを通じてこのコンピュータ監査についての日本の考え方がよく理解できるわけでありまして，私

のコメントも皆様方に少しでもお役に立てばと思います。

堤　それでは第3セッションの講師であるダイクストラさんから特にコメントをしたいということでございますので，そのコメントをいただくことにいたします。

ダイクストラ　オランダのダイクストラでございます。ドルも円も私が考える上にあたっての単位になっておりませんけれども，三つの点についてコメントしたいと思います。

第一は，フレデリック・ブルックス・ジュニアという人が出した数字に関してソルトマンさんが報告なさいましたけれども，それについて述べたいと思います。つまり，開発の時間は$\frac{1}{3}$がプランニングで，$\frac{1}{6}$がコーディングで，$\frac{1}{2}$はテストに充てなければいけないということでありました。これに関して申し上げたいのは，このブルックス氏の経験というのはもう10年ほど前のものでございまして，それからもう一つプロジェクトとしては非常に失敗ということで悪名高かったプロジェクトから得た数字であるということを申しげたいと思います。

むしろ現代のより成功をおさめているような形のソフトウェアのプロジェクトの場合にどういう時間の配分になるかということを参考までに申し上げたいと思いますけれども，全部の時間中の75％はプランニングに向けられております。それから20％がコーディング，残りの5％がベリフィケーション，予想され，意図されたようにすべてがうまくいくかどうかというベリーが5％ですから，ブルックスのとは大分違います。

二番目に申し上げたい点は，マネジメントとコンピュータとの関係についてであります。スライドを見せていただきましたが，いわゆるチーフ・プログラマー・チームの構造を見せていただきました。またHIPOというシステムを示すスライドも見せていただきました。こういったスライドでマネジメントの努力を示しているわけでありますが，ソフトウェアの開発を改善するマネジメントの努力を示そうとしているのでありますが，確かにマネジメントが悪い場合には，どんなプロジェクトでも完全に破壊されてしまいますけれども，同時に覚えておかなければいけないのは，幾らマネジメントがよかったからといって，技術的な欠点があった場合には，決してそれを相殺することはできないのだということであります。たとえばデザイン・チームに幾ら有能な人を配したからといって，もしたとえば飛行機の設計について何も知らない，技術的な背景のない人を置いてしまえば，飛行機は決して飛ぶことはできません。それがチーフ・プログラマー・チームに当てはまると思うのであります。

スライドを覚えておられるでしょうか。スペシャリストが1人チームの中に入っておりました。つまりプログラム・ランゲージのスペシャリストも入っておりましたけれども，こういう人が入ってきますと，それだけでチームが複雑になってしまいますので，その複雑さをほぐすためにもう1人専門家が要るぐらいでありましょう。

それから最後に申し上げたいのは，コンピュータの経営者に対する利用でありますが，実際には影響の方向は逆ではないかというふうな気がするわけであります。いままで成功をおさめてきたコンピュータのアプリケーションというのは簿記でありますが，簿記というのはもう実によく理解され，確立されている事務上の活動であります。ほかにコンピュータのアプリケーションが成功を見た例としては科学技術的な計算でありますけれども，この場合にも覚えておかなければいけないのは，第二番目のアプリケーションのクラスというものの背景には，たとえば数学，物理，また機械学に対する多大な知識が背景にあったということであります。

それに比べますと，もしあるマネジャーが

自分の活動領域においてコンピュータを利用することによって何らかの益を得ようというふうに願う場合には，確かにいわゆるマネジメント・サイエンスという言葉がよく使われてはおりますけれども，いま申し上げたたとえば数学，物理，機械学等々の分野に比べますと，科学としてのマネジメントというのは実態としては存在していないと私は思います。これは単なる神話の段階を出ないわけでありまして，まだ実態としては存在していないと思います。

そこで私が警告を発したいのは，余りに過大な非現実的な期待をコンピュータに対して抱いてはならないということです。つまりコンピュータを使えば問題が解決できるのだということを余り過大に期待してはいけないと思います。

堤　何か反論することがございましょうか。

ソルトマン　まず最初に，プランニング，コーディング，テスティングの時間の配分の問題でありますが，ダイクストラ先生の数字は，コーディングが２０％でありますが，これはブルックスの$\frac{1}{6}$というのと大体同じだというふうに思います。そういう意味では，プランニングとテスティングとの間のトレードオフの問題になるのですが，多分両方とも数字は正しいのだと思います。つまりどういうシステムに関して話しているかということによって両方とも成り立つと思うのです。大型のリアルタイムで本当にテストにおいて５％しか必要ないというのは，ちょっと信じられません。しかしながらたとえばアプリケーションのプログラムでダイクストラ先生がおっしゃったような種類のもの，つまりたとえばよく理解されているのは，簿記におけるようなアプリケーションにおいてはテスト時間５％ぐらいで多分十分なものだろうと思います。

そこで一つ意見一致できないのですけれども，マネジメントというのは科学ではない，ということには賛成できません。確かにもしエンジニアリングがなければ飛行機が空を飛ばないのも確かでありますけれども，マネジメントというのは，その役割りとしてエンジニアが必要とするような資源を組織したり，また提供したりすることができるというふうに思うのです。確かに余り過大な期待を，たとえば数学的なモデルの適用から期待してはいけないということはわかりますけれども，だからといってあらゆる種類の予測であるとかそれから傾向分析であるとか趨勢分析であるとか，また合理的な形での意思決定が全く無意味であるというふうには私は思いません。

堤　フロアーから質問をいただいております。講師の皆様方がシステム監査人制度の確立ということについても異論がなくそれを望んでおられるようでありますけれども，具体的にどう考えたらよいか。たとえば単に企業に担当者を置けばよいというのか，あるいは国家的な制度を設けて，システム監査の要件や手続までもスタンダード的なものにする必要があるか，現段階で考えられる方向を示してほしいということでございます。このあたりをパネルの先生方に御意見を承りたいと思いますが，島川先生いかがでしょうか。

島川　私は，先ほどから申し上げておりますように，自分の銀行の場合を考えるだけで精一ぱいでありまして，実は本来いかにあるべきかということについては，端的に言って余り興味がないわけです。少なくともどういう方向になろうと，まずは私は経営者は主体的に問題を把握して，まず事業で育てていくことだ。仮に将来国家的に何とか制度になるにしても，事業で育てられた人というものでないと，専門知識その他から豊かな一流の監査人というものにはなかなかなれないのじゃないか。そういう意味で，各事業がまずそれぞれ育てて，そしてそういう方というのは事業にいつまでもいるのか，あるいは独立したいのか，こういう問題にもなるのでしょうから，私は現実的にはまず事業で育てるという

ことが初めて，後は先ほどのソルトマンさんの言葉づかいによれば，おのずからいま結論を出さなくても合理的な方向に進むのじゃないか，かように考えます。

堤　ほかの方，いかがでございましょうか。

石田　私はさっき監査の基準の点に触れて，話が中途になっておりましたけれども，公認会計士と監査役が同一の立場で会計の監査をやるといたしますと，公認会計士のよりどころになっております監査基準あるいは監査実施準則というものがあるわけでございます。監査基準の中に監査をするのには，内部統制システムを評価し，それによってサンプリングの範囲を決めるということになっておりまして，公認会計士協会でお出しになりました内部統制の質問書というのは，実はその問題でございます。

あわせて，これを検討すべき問題は，ＥＤＰシステムに対する内部統制の質問はそれはそれで結構ですが，その場合の監査実施準則というものが一体どういうふうに変わるのか，要するに実施準則の文言そのものはそれでよろしいかと思いますけれども，文言の実態，実施準則を見ますと，たとえば証拠だとかあるいは計上するだとかというふうないろいろな文言があるわけでございますが，たとえば会計の計上の仕方それ自体がすでにＥＤＰ会計では変っておる，あるいは内部証拠は，先ほどから御説明がありましたようにインビジブルなものになっておることになりますと，文言は同じでも，実態は変わっておりますので，内部統制質問書の精神が同じでも，ＥＤＰ会計に対する質問書ができましたと同じように実施準則に対してそれなりのやはり別な一つの準則というか，ガイドラインをつくるべきであろう。これは公認会計士と同時に，われわれ監査役の共通の問題であるというふうに考えております。

堤　いま石田先生から御意見がありました公認会計士協会の内部統制についての質問書の問題でございますが，ちょうどここに公認会計士の井上先生が見えておられますので，ちょっと御意見を賜りたいと思います。

井上　昨年ＥＤＰに関します内部統制の質問書を公表いたしまして，その後商法監査に備えましてもう一度見直そうじゃないかというところから，９月１４日ようやく改定案が脱稿いたしました。それに伴いまして，実は御指摘の監査基準なり，これはＥＤＰシステムに対する監査基準でございますが，ＥＤＰシステムに対する監査基準並びに監査手続書を同日付で公表いたしております。まだ皆さん方のお手元には届いておらないだろうと思いますが，私，昨日ゲラを見てまいったので，多分近いうちに公表されると思います。

考え方といたしましては，ＥＤＰシステムの監査に関しましては，このような概念を持っているわけであります。

まず，ＥＤＰシステムにおける取引記録のデータ処理の有効性と妥当性を検討し，会計記録の適正性の程度を確かめるために，ＥＤＰシステムに組み込まれた内部統制の信頼性の程度を評価するのだ，こういう表現であるわけでございます。と申しますことは，財務諸表監査は財務諸表全体の適正性を監査するわけであります。それに対しましてＥＤＰシステム監査はその中間的な監査でございます。と申しますのは，ＥＤＰシステムが行っておりますのは，取引記録を中心といたしました会計記録でございます。したがいまして，まずデータ処理過程におきますところの取引記録の有効性あるいは適正性，それを確かめようじゃないか。そしてその結果として会計記録の適正性の程度が確かめられるのではないか。しかしながら，ＥＤＰシステムは非常に多くの内部統制組織が入っております。と申しますのは，プログラムにチェック機構が多く組み込まれていることによっても明らかだと思います。したがいまして，ＥＤＰシステム自体の信頼性をまず評価することが大切だ。

したがってわれわれはＥＤＰシステムの内部統制を評価するという言葉ではございますけれども，結論的にはそれとＥＤＰシステムの信頼性とは同義語になるのではないか，こう考えまして，ＥＤＰシステムの信頼性，すなわちＥＤＰシステムの内部統制を確かめることがＥＤＰシステムの監査だ。最終の目標は会計記録の適正性を確かめるのだ，財務諸表との関係は中間監査である，そういうふうに考えております。

したがって，監査手続にありましては，二つに分けまして，前半がまずＥＤＰシステムの内部統制の遵守性，信頼性の遵守性，定められた手続に対してどの程度の離反をしているか，あるいは逸脱をしているのはどの程度であるかというふうな監査を実施しまして，しかる後に会計記録の中で信頼性が高ければ示唆の範囲を非常に少なくいたしまして，会計記録そのものの確証性，立証性を持とうではないか。そのためには外部証拠を求めてやるのだ，こういうような構造の監査基準書と監査手続書を公表するような次第でございます。

以上まことに簡単でございますけれども，申し上げさせていただきました。

上野　ちょっと協会の立場から補足申し上げますと，ただいま井上先生が申し上げましたＥＤＰシステムの監査基準及び監査手続試案というものでございますが，これは９月14日に会計士協会の理事会で決定したものでございまして，これはあくまでも協会独自のものでございまして，したがって監査にあたって使っていくという道具でございます。これが慣行化されて，その上でその間にはいろいろ手直しもあろうかと思いますが，その上でただいま石田先生からお話があった監査実施準則等とのかかわり合い，その辺が大蔵当局などとも若干その先において詰めをしなければならないものです。そういう意味で作成し，発表したものでございます。

堤　それではまとめに入りたいと思いますが，西野先生から一言ずつまとめのお言葉を賜りたいと思います。

西野　大体皆さんの意見が大変よく出ておりまして，私は何を申し上げることもございませんが，先ほどちょっと結論的にはやはりコンピュータ監査のシステム監査と申しますか，こういうもののきわめて概略的なガイドブックというか，そういうものをどこかのところで共同で，あるいは一ところではなくて，たとえば公認会計士協会とか内部監査協会とかあるいはその他の機関で一応標準的なものをまとめてみることは，日本の今後のシステム監査を普及するのに，またそれを実行する上において非常に重要である。

もう一つ，先ほどもありましたが，この内部統制，公認会計士協会の出しておるこういう問題と同時に，大蔵当局との税務の問題があるのです。そういう問題も信頼性の問題が確立しませんと，なかなかこのコンピュータのＥＤＰ監査が税務の問題でごちゃごちゃする問題が非常に多くなる。監査をやはり信頼性を確立するために，また同時にさらにもっと言うならばいま問題になっておりますし，商法においてもこうしたコンピュータにおけるデータを帳簿として認めさせるという問題もまだ確立しておりません。

こういう問題を，これから日本としてはやらなければならない問題はたくさん残っているわけでありますから，そういう問題の積み上げをするためには，まず一番先にいま申しましたような実務のガイドラインというようなものをぜひつくっていく。これは合同でチームをつくるとか各機関で一緒になってつくるとかということが必要ではないかを思っております。

上野　先ほど私どもの立場からの意見としまして，内部システム監査人の制度を確立していただきたいということを申し上げたわけでございますが，それはただいまお話にちょ

っと出ました監査基準の中にもうたわれているわけでございますけれども，やはり会計士監査はある一定時点の監査といいますか，どちらかといいますと内部システム監査人のように常時継続的にそのシステムの欠陥があるかどうかとかそういうことについてのチェックをするというわけにまいりませんので，どうしても常時継続的に監査を行っていただけるシステム監査人がおられて，それで内部統制の信頼性を維持させていただくということが私どもにとっては非常に必要かつありがたいわけです。そういう意味で先ほどからいろいろと申し上げておるわけですが，先ほどちょっと触れました私ども協会としては，井上先生からもありました電子計算機を使用した会計組織に対する内部統制の質問書を今度の監査基準を発表するのに伴いまして，それにつれて改定を行い，またその監査基準によっていわゆるアメリカであったような大きな不正事件ということが起こらないように，早くこれをとりあえず発表いたしまして，企業の方でシステム監査人制度を確立していただきたい，これが一番重要であるということで急いで発表したわけでございますので，何分ひとつよろしくお願い申し上げたいと思います。

島川　私は，まとめというよりも，感想の一つなんですけれども，先ほど上野先生からお話がありましたコンピュータ時代になって手作業時代のシステム，牽制システムのメリットを見忘れてはいないか，こういう話がありまして，非常に感銘を受けたわけです。と申しますのは，私どもの方で現在非常に大がかりなリアルタイム・システムを導入しておりますけれども，それと同時にリアルタイム・システムになって，大規模になって初めてインプットの重要性というものを保全するためにマニュアルで即座にインプット・データをチェックする，そういう制度を採用したわけです。巨大な投資をし，そして省力化するという目的のためにしたエレクトロニクスの原理というものの世界に，逆に従来よりもインプット・データの保全を確保するという意味で，インプットした直後直ちに専門の人がチェックする人間が要る。こういうマニュアルとエレクトロニクス（あるいはコンピュータ・システム）のマージングというか，そういうことで自分としてももやもやしたものがあったのですけれども，これもシステムの安全性とか保全という意味からは大事なことなんだ，御指摘をいただきながら，改めて私どもの方の銀行のシステムというものの根拠と申しますか，そういうものを得たような気がします。

それから監査基準につきましては，いま諸先生からお話が出ておることに私は大綱において賛成でございます。相互に協力してつくっていきたいと思います。乱用，悪用，不正というものをなくすために監査基準とかそういうものが非常に必要なわけでありますから，私どもの方としましても，内部的にも幾つかの部門に分けましてマニュアルをつくっております。ドキュメンテーションとか監査プログラムとかいろいろなものが今後出てまいりますが，何と申しましても銀行の場合一つ問題になりますのは，リアルタイムで日中動いておりますので，これを途中で監査する手法があるかどうか。こういうことにつきましてはなお将来の手法の開発に待つべきでございます。いかに証跡が必要だと言っても，リアルタイムの途中ということになると，かなり現在の手法ではむずかしい面があるのではないか。一日の終わりとかそういうことになりますと，これはまた問題も違ってくる，こういう感じがいたします。

要は，コンピュータ・システムの安全性とか信頼性を確保するには，結局企業の安全性とか安定性とか信頼度を確保し，自由主義経済を守るのだ。そういう意味では，公認会計士の方とあるいは内部監査役の方あるいはわれわれも共通の認識ではないか。この基盤に

立つ限り，問題はおのずから解決されていくはずである，かように考えます。

石田　私の先ほどの質問に対しては，井上先生から御返事いただきました。

私自身が興味を持っておりますことは，内部統制の必要性あるいはシステム監査の必要性，これは皆さんの意見の一致するところでありますが，具体的に方法をどうするのか。要するにコンピュータ会計の具体的なテクノロジーあるいは方法論をやはり検討すべきではないかというふうに考えております。ソルトマンさんのお話でも，コーディファイされたディシプリンなどは将来のことだという，大変むずかしい問題だと思いますが，方法論それ自体はやはり着実な検討が必要であろうと考えております。

ソルトマン　私はもう十分に時間をいただいたと思いますので，特につけ加えることはありませんので，この機会を与えていただいたことに対してお礼だけ申し上げます。

堤　もう時間も過ぎましたが，システム監査とは，日本が完全な情報化社会に移行するためには避けて通ることのできない重要な関門でありますし，また新しい挑戦課題であります。これに有効に立ち向かうのにはやはり各関係者からの相互協力，チームワークというものが不可欠であります。その意味で，システム監査とは，それ自体はシステマティックな監査でなければならないということを本日改めて痛感した次第であります。

セッションⅡ　集中処理から分散処理へ

経営面から見た分散処理システム

要　旨

コンピュータの利用技術は，ハードウエアおよびソフトウエア機能の高度化，データ・コミュニケーション技術の進展，ターミナル・サイドのインテリジェンスの向上に伴い，処理形態は，集中型から分散型へと急激に移行しつつある。またデータ・ベースも分散化への傾向にあり，銀行をはじめ金融機関，諸企業等における重複型データ・ベースもクローズアップされ始めている。ここでは，今後の分散データ処理システム発展の方向，解決すべき問題点，分散処理がユーザーに与えるインパクト等について論じる。

経営面から見た分散処理システム

アラン・ウエバー[1)]

ニューヨークの最大の銀行として，世界では二番目に大きい銀行として知られておりますシティバンク，もとはファースト・ナショナル・シティバンクとして知られておりましたが，ことしの二月以降，シティバンクと名前を変更しております。内外に５万人の行員を擁し，国内に３００の支店，海外１０１カ国において５００の支店並びに関連銀行を擁しております。

私は国内金融機関のサービス部の長で，責任者でありますが，仲介業者並びにコレスポンデント銀行，保険会社，市，並びに州政府に対する銀行サービスの処理業務の補助機能を提供しております。

毎日の処理業務でありますが，内外顧客銀行間の１００億ドルの振りかえ，１８億ドルの小切手の処理，それは１５０万から２００万点の処理ということになります。９００件のデータ処理業務，１億5,０００万ドル相当の請求書の送付業務，税支払い業務の処理を，大企業，政府機関にかわって行っております。生産管理業務もやっておりまして，商品の企画，開発，販売，サービスなどを手がけております。

さて，分散処理システムの管理についてお話ししてみたいと思います。

最近，経営管理者は，実務業務におけるコンピュータ資源の解禁を目指して努力してまいりました。そして業務分野では成功しておりますが，単にコンピュータを使うというだけでは十分ではありません。コンピュータを資源として管理しなければならないと同時に，その利用によって利益を享受できるようにしなければならないのです。

データ処理のコストの高騰並びに複雑化によって多くの銀行並びに他の機関は，伝統的な集中処理方式から，最近では非集中処理方式もしくは分散処理方式へと転換しつつあります。

そこでシティバンクは，新しいマネジメント・アプローチを通信及びコンピュータ使用に対して採用しております。これを「非集中的データ処理」と私どもでは呼んでおります。これはビジネス・システムの外郭に処理機を持ってまいりまして，ソースでデータを収集処理するという考え方であります。これは，過去に行われてきたこと，また今日の大半のビジネスで見られるやり方とは全く違う構想であります。

シティバンクにおきましては，大型のセントラルコンピュータを排除することを考えました。大型のセントラルコンピュータには，管理上の問題，成長上の問題，ユーザー開発能力の欠如，周辺施設など，内在する問題を抱えているからであります。非集中的データ処理と「分散処理」とは次のひとつの理由により異なります。つまり非集中的データ処理はすべての処理業務をユーザーすなわちわれわれの場合にはビジネス・マネージャにまかせるという点であります。ビジネス・トラン

(Alan J. Weber)
1) シティーバンク副社長

ザクションは源泉で，把握，編集，検証されるだけでなく，同じところで顧客のために処理されます。われわれが利用する唯一の中央コンピュータは，社内の会計，経営情報システムを，分散データ収集システムから入力を受けて処理するものです。

この最新の技術を反映した新しいコンピュータの哲学を採用するには，業務環境の再編成が必要です。すでに再編成された部門もありますが，今後さらに再編成しなければならない分野も残っております。

シティバンクのアプローチを説明するに当たり，まず基本的な経営理論，当行の目標，また目標達成のための各段階，当行の業務管理の枠組みを構成する主要要素についてお話ししてみたいと思います。最後のものは，バック・オフィスといままで呼ばれていたものです。まず第一にここ数年来の動向，第二にシティバンクの位置づけを過去を振り返ってみて，私どもが現在の状態に達するのにどのような過程をたどったか。第三に，今後どのような方向に行こうとしているのか。個人または企業顧客のトランザクション処理業務の目標をどのように考えているかをお話ししてみたいと思います。最後に，どのような方法でわれわれは目標を達成しようとしているのか。ミニまたはマイクロのハードウエアを用いた非集中的処理アプローチが標準的な取引処理業務においてうまく動いていることを理解していただきたいと思います。

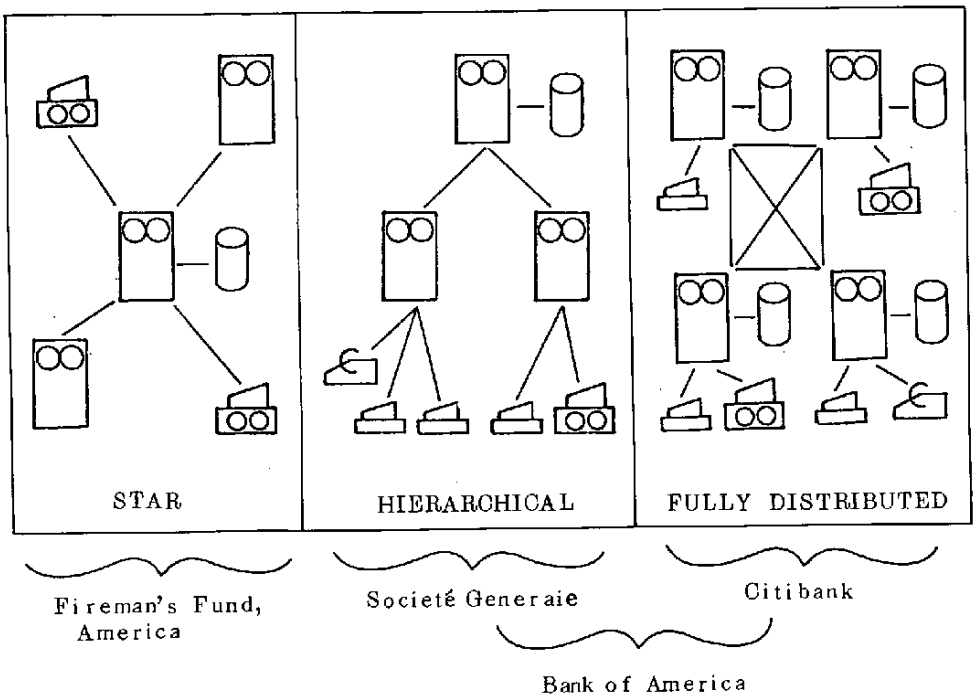
コンピュータ利用の当初から，総合，統合的なシステムは利用，形態の主流でありました。初期のコンピュータは比較的大型で，コストもかさんでおりましたので，オペレーティング・システムの効率の向上ということに視点が置かれ次にＣＰＵが高価だったためアプリケーション，ファイルの統合等による規模の経済が追求されました。しかし，ここ１０年間コンピュータを使ってきてわかったことは，大型のシステムは，問題もまた大型であるということです。ＣＰＵの有効性を最大にしようとする努力がなされるにつれて，オペレーティング・システムは非常に複雑になってまいりました。中央コンピュータにますます多くのアプリケーションがロードされますと，ダウンタイムであるとか遅れた業務入力を吸収する余裕がなくなってしまいます。中央コンピュータ・システムが経営者にとって悪夢になってしまったわけです。

そこで，まず大型アプリケーション・システムをサブ・システムに分ける努力がなされ，次にモジュールにシステム化する努力がなされ，最後に分散型の処理方式という方向に向かったわけであります。ここ３～４年間分散処理システムが開発されつつありますが本当の意味での分散処理システムへの接近には各種の方々があります。最初にまず近年開発中のシステムの総括的な定義，それから，異なるビジネス部門においてどのような動向があるかを次に話してみたいと思います。

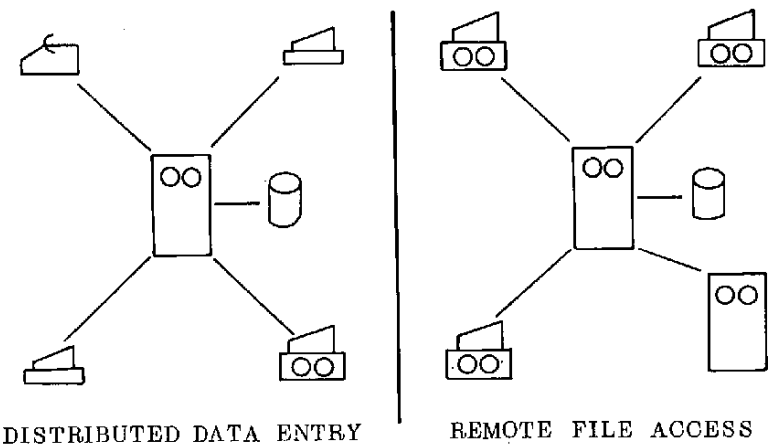
分散処理システムというのは，データファイルが中央集中型であっても，非集中型であってもよいわけでありますが，複数の処理機を持っているということが一つの定義になっております。分散ネットワークのデザインには三つのアプローチがありまして，スターネットワーク，階層ネットワーク，それから完全に分散されたネットワークがあります。これは対象とする業務構造の違いを反映していると思われます。

スターネットワークは中央のホスト・コンピュータとグローバルデータベースから成ります。この中央のコンピュータは，ターミナルまたはワークステーションその他のコンピュータからデータ・インプットを受けそれに答えを送ってシステム全体に対する監督管理を行っております。最近，スターネットワークで一番使われている方法は，データエントリーの機能を分散することであります。インテリジェント・ターミ

DISTRIBUTED NETWORK TYPES
COMPANY EXAMPLES



STAR NETWORK TYPES



ナルとかワークステーションが，入力の検証機能を一部果たしております。これは，従来の統合システム，総合システムから自然な形で派生してきたものでその幅広い開発が期待されます。

それから，リモート・ファイル・アクセス・システムは，スター・システムのもう一つの使用例でありますがこれは今日のオンライン・システムに非常に近いものであります。リモート・ターミナルがセントラルシステムに結合されておりまして，セントラルシステムにほとんどのデータが貯蔵されているわけであります。ターミナルが一部処理を行い，そして論理編集の大半を行っております。リモート・ファイル・アクセスの例はファイアマンズ・ファンドというサンフランシスコに本社を持つ保険会社について話す時に多くの例をあげられると思います。

次に階層ネットワークでありますが，名のごとく階層になっており，それぞれのレベルでそれぞれの仕事を行っております。まず一番上に中央コンピュータがありまして，グローバル・データベースを持っております。中間レベルのプロセッサーは，低レベルのコンピュータからのトランザクションをまとめて中央コンピュータに伝送いたします。逆に，中央コンピュータからの応答を低レベルに送るということも行います。リモートファイル・アクセス・システムと同様に，低レベルのコンピュータは，一部処理業務を行い，論理編集の大半を行います。

この階層アプローチの例は，フランスの銀行，ソシエテ・ゼネラルに例を見ることができます。

それから完全に分散されたネットワークにおいては，中央コンピュータは存在しません。そのかわりに複数のコンピュータがありまして，それぞれデータベースを持っております。これは直接通信方式によって相互に結合されており，特定のアプリケーションによりまして，このデータベースは特定のプロセッサーに対して，ローカル・データベースとしてサービスすることもできますし，ネットワークに対してグローバル・データベースになることもできます。これについてはシティバンクの例をあとでお話ししてみたいと思います。

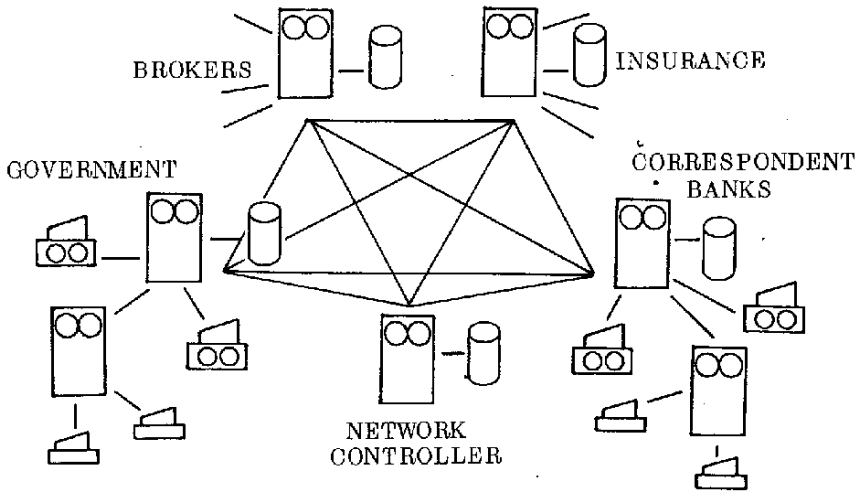
プロセッサー間の直接コミュニケーションは，完全に分散されたネットワークでは必要な要件でありますが，ネットワークが非常に大きくなりますと，コストもかさみますし，複雑なものになります。この解決策は現在研究中のARPANETに見ることができます。

ARPA，これは国防省の先行研究プロジェクト局でありますが，コンピュータ間の高速伝送の通信ネットワークの開発を現在主管しております。シティバンクは，その非集中型のプロセッサーをリンクされるため，その種のネットワークについて研究中であります。

このような分散ネットワークのノードにとりつけられる各種の機器についてお話しすることは意義があると思います。まずコンピュータが挙げられます。これはホスト・コンピュータと呼んでもいいと思います。それはローカルターミナルをサポートすることにもなります。それから，大型，中型，小型のメモリー，インテリジェント・ターミナル。これはキーボード・ターミナルやバッチ・ターミナル，グラフィック・ターミナルなどを含みます。それから，ワークステーションと呼ばれるものがありまして，これは小型のビジネスコンピュータが持つ多くの特徴を持つものですが，特定のアプリケーションに順応されたものであります。弾力性，簡便さ，信頼性という点においては事務機に似ていると考えられます。ネットワークのノードは，処理能力とデータ記憶能力を持っております。

では，ソシエテ・ゼネラル銀行，それから保険会社であるファイアマンズ・ファンド，それから銀行であるバンク・オブ・アメリカ

CITIBANK'S ARPA–LIKE NETWORK



TYPES OF EQUIPMENT

TYPE	Keyboard	Special Data Entry Devices	Processing Logic	Memory	Local Data Base	Global Data Base
Computer	X		X	X	X	X
Terminal	X					
Intelligent Terminal	X		X	X		
Work Station	X	X	X		X	

CAPABILITY

の例をとってお話ししてみたいと思います。

ソシエテ・ゼネラルは，パリに本社を持つ国営銀行でありますが，フランスで三番目に大きい銀行で，2,000ほどの支店を持っております。現在のところ，二つのデータ処理センターを持っております。一つはパリ，もう一つはマルセイユの北部にありますが，各支店からのデータが毎晩この二つのセンターに送られてきます。ここで処理されまして，報告その他の応答が各支店に翌朝の8時までに送り届けられます。このドキュメントの輸送に当たりましては，鉄道，バス・それからモーターバイクなどが使われております。

1972年に，この銀行は，次代のデータ処理はどうあるべきかということを検討いたしました。トランザクションの量も急速にふえておりましたし，既存のシステムでは，78年には能力いっぱいになるということが予想されました。1980年には，1秒当たり100のトランザクションに量がふえるであろうということも予測されました。その当時発表されておりましたコンピュータの能力というものを考えますと，1～2のセンターではこれだけのトランザクションの量を処理できないということがわかりました。高い信頼性を保ちつつ速やかに処理することは，とうてい1～2のセンターではできないということで，分散型の処理システムについて検討を始めました。1973年には，その経営管理者を説得し，分散処理システムを採用することを一つの目標といたしました。

この銀行がとっております分散処理システムは非常に興味深いものであります。それは，階層システムであります。トップにパリ・センターがまずあり，次のレベルでは毎日のバッチ処理の30％をおこないパリ・センターのバックアップをしております。これがマルセイユ・センターであります。その次のレベルには，グループまたは地方センターと呼ばれるものがありまして，これは小さな銀行に相当するものであり，各グループが平均10支店を受け持っております。各支店は，ターミナル・コントローラーと平均5ターミナルを持っております。全システムには大体1万ターミナルほどがありますが，このターミナルはノン・インテリジェントのCRTタイプのユニットでありまして，プリンターがついております。第一レベルの制御で，アプリケーションとコントロール・ロジックがターミナル・コントローラーにありまして，ほかのコントローラーではグループ・プロセッサーにそれがあります。

機器の投資のうちの大半が支店機器への投資でありますので，ノンインテリジェント・ターミナルを採用することを決定いたしました。そしてターミナル・コントローラーからインテリジェントが供給されるわけであります。この経済性は，グループ設置がオペレーターを要しないということであります。システムはディスク，ドラム，コア・メモリーなどで構成されています。200ものグループ・プロセッサーがあって，それぞれにオペレーション並びにプログラミング・スタッフを置くということは，非常に大きな支出となります。銀行は，この方式について慎重に調査を行い，この階層型アプローチに決定しました。

この分散処理方式は，ソシエテ・ゼネラルにとっては画期的な変化を意味するということで，注意深く，漸次その移行を行うということが決定されました。この分散システムによって，ワークロードの増大とともに，システムを大きくすることも可能であり，トランザクションの量の増大に伴ってシステムを調整することもできます。それから，新しい支店，グループを開設するのに弾力性を持たせることができるという訳です。このシステムは，階層型の分散処理が増務能力を容易に追加できるという利点があることを示しております。3年のスケジュールが立てられ，1980年にはすべてのノードがオペレーションに移

されるという目標が立てられております。

それから，アメリカの七大保険会社の一つであるサンフランシスコに本社を持つファイアマンズ・ファンドの例も，分散処理型を採用した一つの例であります。これはリモートデータ・エントリーによる分散システムということが言えましょう。1972年にこのファイアマンズ・ファンドは，まずインテリジェント・ターミナルを6カ所の処理センターに設置するということで，第一歩をとりました。支店から上がってまいります保険証券のコピーをもとに，オペレーターはCRTターミナルの空欄にデータを打ち込むわけであります。オンライン・データ・エントリーによりまして，パンチ工程を省略でき，さらに即座に入力の検証が可能になったわけであります。

昨年，ファイアマンズ・ファンドは，さらに分散処理システムを整備する上で各支店にミニコンピュータを設置しました。その利点としては，データ処理のスピードアップができ，また代理店の手書きの申込書からじかに入力ができますから，その料率を計算する必要もなければタイプする必要もなくなったわけです。そしてその情報をバッチ方式で本社へ伝送し，本社で料率計算され，コード化されて，処理されます。保険証券の情報は支店に送られ，新しい証書が自動タイプで印刷されます。

76年には，各支店にローカル・ファイルを設置することが計画されております。そしてオンライン・ベースで中央のシステムに結合し，変更や保険金請求業務を補佐するために，中央データ・ベースに問い合わせできるように配慮されております。各支店ではひき続き保険金支払い事務を処理することになるでしょう。

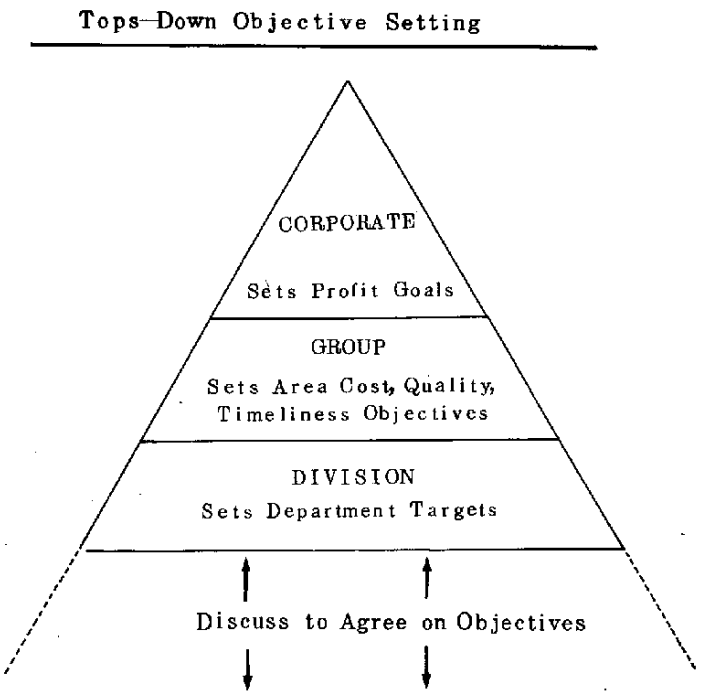
ファイアマンズ・ファンドは，この分散処理方式に向かって，ミニコンピュータを設置し，ローカルファイルを支店に設置する準備を進めているわけでありますが，現在のところ，ローカル・ポリシー・ホールダーデータを記憶することまでの計画はできておりません。

次は，1,100万の口座を持つバンク・オブ・アメリカの例でありますが，これは口座情報にオンラインでアクセスが各支店のターミナルで可能になる方式であります。このシステムは，分散的，階層的アプローチで，データベースを分割し，業務を何台かのコンピュータからなるミニコンベースのモジュールによって分散処理します。

バンク・オブ・アメリカのデータ処理センターはサンフランシスコとロサンゼルスの二ケ所にあり，カリフォルニア州内に1,000以上の支店があります。各センターでは半分ずつその処理を分担しており，1日当たりそれぞれ300万のトランザクションの処理ということになります。現在のところ，各支店からのデータはその日の午後遅くセンターに集められ，ファイルの更新と問い合わせに対する応答は，各センターともIBM370-195 1台と多数の同168によりバッチモードで処理しております。結果のプリント出力は，翌朝早く各支店に届けられるようになりますが，システムが完了いたしますと，現在のメッセンジャー・システムは必要なくなります。各支店のビデオ・テラー・ターミナルによりプログラマブル・コントロール・ユニットを介して，マルチドロップ回線によりセンターと結合されます。ネットワーク全体のターミナル総数は6,000台でありまして，コントローラーの総数は1,000台，大体半々の割合で二つのセンターが分担しております。

これらの会社におきましては分散処理システムをとっているわけでありますが，シティバンクにおきましても独自のものを開発しておりますので，御紹介いたします。

シティバンク型の管理というのは，トップダウン型の管理と呼ばれるものであります。このことについて，われわれの企業としての



哲学を少々御紹介したいと思いますが，この言葉の説明としましては，上位レベルで設定されました目標を解釈いたしまして，自分のレベルでの自分の目標を設定するということになります。トップレベルの管理者は，ある種の数字を要求いたします。たとえば1株当たりの収益率の増加ということで，われわれの場合には年率15％ということになっております。それがグループ管理者の方に伝えられまして，それが具体的な目標にされるわけです。たとえば，コストを一定にしてサービスを向上させるというようなことであります。それから，部門の管理者にこれが送られまして，それを組織体制並びに管理手段と関連させて，その目的を達成するわけです。また，各オペレーション単位でもその結果と処理目標を設定しまして，そのレベルでの目標とするわけであります。

このようなゴールを達成することは必ずしもやさしいことではございませんが，特にわれわれの競争的な環境にありまして，次のような手段によりまして成功を獲得するわけであります。たとえば，

○少なくとも同率の成長をとげている産業に関与すること。
○現在の業務でのマーケット・シェアの拡大。
○新製品の提供。
○コストの低減。
○価格のひき上げ。
○または，これらのすべてまたはそのうちの幾つかの組み合わせ。

よき時代には，収入の増加がコストの上昇を上回り，利益向上が達成できまして，そのためにコスト低減を強調することはむずかしかったわけです。そこでわれわれとしましては，収入曲線の上昇率がコスト曲線上昇率を上回ることが望ましいわけです。われわれは営業費を削減することによって，生産性をあげなければならないのです。この場合に，業務の各分野のなかで常に費用効果が問題だっ

たのは，大型のマルチプログラム，マルチ処理のコンピュータ・システムでありました。つまり大型のコンピュータ・センターでは「大きいほどよい」という誤った考え方がはびこっていましたが，その間違った考え方がわれわれを非集中型データ処理へ移行させるのに重要な役割りをはたしました。

第一に，大型ビジネス計算システムは高度の業務知識を必要とします。そのような高度の業務知識をもったものを，システム設計にあたるプログラマー，アナリスト，システム・マネージャーに求めるのは通常困難です。彼らは技術者であって常にこのような包括的な視野を持っているとはかぎらないからです。

第二に，大型コンピュータの規模とその複雑性はそのシステム開発努力と大きな関係があります。多くのプロジェクトはその設計がすでに実情にあわなくなってから完成することがしばしばあります。つまりプロジェクトの完成までにあまりに時間がかかりすぎるわけです。

第三番目には，いろいろな環境的な要件で，たとえば電源の問題，空調の問題，機密装置の問題等で，大型のコンピュータは，費用効果的にもあまり好ましくありません。

最後に，これは大事なことですが，ますますビジネス・マネジャーとコンピュータ専門家との間のコミュニケーション・ギャップが大きくなってきているということです。業務の問題をコンピュータ・システムの問題に変換するのはますます困難になりつつあります。というのはビジネス・マネージャーとコンピュータ専門家は同じ「言葉」で話さないからです。彼らの考え方の違いの故に仕事の優先順位をめぐってしばしば衝突がおきるのです。先程トップダウン・マネジメントと申し上げましたが，シティバンクでは現在の銀行業務から目標とするそれへの移行方法としてトータル方式を採用しました。ここではその順序が大切であります。

1.今日のわれわれの状態の正しい評価
2.明日達成したいと望む目標の設定
3.今日の成果をもたらした過程の正しい認識
4.明日の目標を達成するための手段の検討

われわれは今日の手続から明日の成果をうまく引きつげるよう注意しなければなりません。よりよい成果は確固とした手続きなしには達成されないということを覚えておかなければなりません。

そこで，われわれは，このような目標を管理者の行動へどのように結びつけていったか，どのようにして現在の位置に達したかを時系列的にさかのぼって，ニューヨークでの達成事項を考えてみますと，1970年代の初めに，銀行の上級管理者が銀行全体の業務のベースを分析しまして，業務費用の上昇が年率過去15年間15％であるということがわかりました。サービスはそれほど向上しないのに事務の滞貨は多く何らかの改善を考えなければなりませんでした。

業務量とコスト増大に伴いまして，シティバンクは大規模な統合を業務にもたらすことになりました。そこで，オペレーション・グループをウォール・ストリートに設置し，国内のオペレーション・ベースをそこに置いたわけです。そのときもまた大型コンピュータのマネジメントを集中化することにいたしました。そして，それをすべてのオペレーション業務が分有することにいたしました。これは普通の大型のユーザーと何の変わるところもありませんでした。われわれは大型コンピュータのいろいろな神話を信じていたわけです。後から考えてみても，自分達のコンピュータへの接近の方法をこころえた人々の数，ハードウェアのセールスや，外部のコンサルタントから得られた情報を考慮すると，当時のわれわれの集中方式の接近は適当であったと思われるのです。

ところが，管理者の主な関心事は，処理機能に関する彼の責任であり，顧客に対して充

分注意を払わなかった訳です。そして多くの問題はそれを処理する人々をふやすという方法で行なわれたので機構が非常に膨大になってしまい，管理不可能のレベルまで達してしまいました。

このような悪循環が改善を見ません間に，1970年の終わりごろになりまして，それを変える計画が立てられました。1971年に業務の再学習をし，業務全体を分解し，分析し，より効果的に再組み立てをするような計画がなされました。

そしてその第一のステップとして，新しく，変容する銀行の状態に通じ，技術の変化についていける経営チームを編成することでした。

第二のステップとしまして，財務管理システムの整備をいたしました。ここでのかぎとなりますのも，トップダウン型マネジメント目標の設定でありまして，計画展開とか実施とか報告は底辺の管理者がするようになりました。またもうひとつは，業務の成果を事前に正確に予測することです。これらの要素をささえる経営情報システムはすでに実施されています。

次のステップとしましては，適当な品質と時機の管理であります。顧客サービスの質を測定する規準が設定されました。日々の問い合わせ，調査，苦情同様，タイミング，待ち時間，またはこれらを1日ごとに分析する管理システムが完成しました。

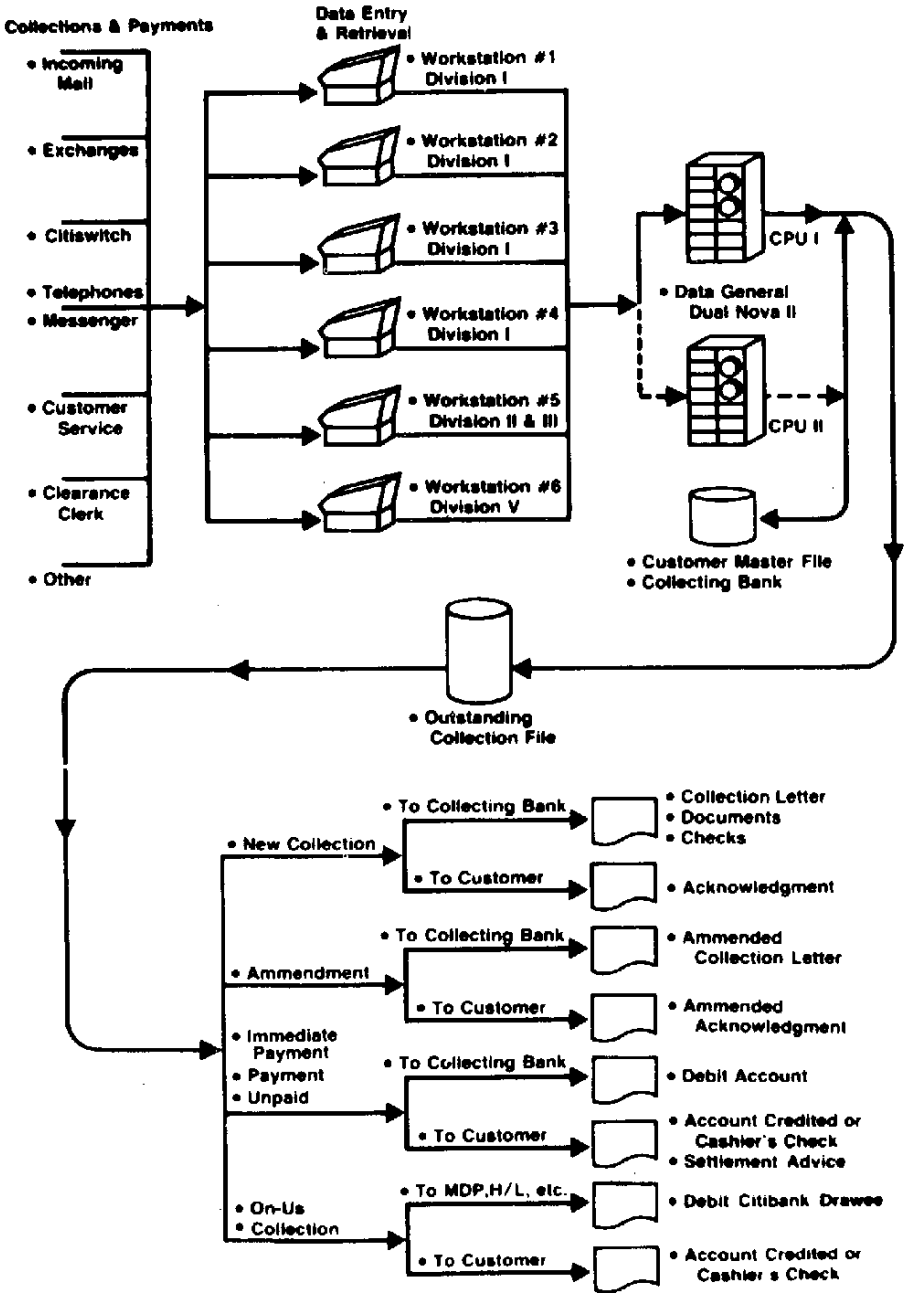
このようなプランは1971年の初頭に立てられまして，これをPCS，パーフォーマンス・クライテリア・システムと呼んだわけであります。これは，製品またはサービスとか処理方法，組織，レイアウトなどを統合いたしまして，非常に簡潔な非集中型のプロダクト・チャンネルに統合したわけであります。処理ラインは専用化されまして，特定の製品と結びつけられ，各ラインには製品処理のすべての必要な機能が与えられました。

1974年に生産現場のコントロールができた後，市場のセグメントに合わせたサービス機構をつくりました。このプロジェクト・プランによりまして，市場セグメントに合わせた別個の業務ベース，つまり多国籍企業グループ，国際銀行グループ，国内銀行グループ，そしてニューヨークの消費者銀行グループというふうに分けられたわけであります。このマーケット・セグメントに合わせた組織の編成には次のような利点があります。一つは，特定の市場に関連した処理の専門知識が得られるということ，第二には，新製品を迅速につくり出すことができる，第三には，個人，法人の顧客に対して，個別的に顧客に合わせたサービスができるということ，それから，銀行グループに対するより柔軟なサポートの態勢ができる，それから，オペレーティング・ベースと銀行業務を結びつけることができるということです。

1975年，シティバンクは50台のミニコンピュータを設置しまして，大きく組織変更しました。このミニコンピュータによりまして，各プロセス・マネジャーにそのプロセスの環境の完全なコントロールができるようにいたしました。他の多くの生産プロセスも自動化いたしまして，それによってプロセスがデータセンターから独立するようにいたしました。

私のディビジョンでも，集金部にこのようなシステムが設置されました。この集金機能といいますのは，コルレス銀行にとりましては，銀行が提供する重要なサービスの一つと考えられております。顧客が輸出出荷に関する証書，たとえば小切手，手形，証書等外国銀行より振り出されるとこれは集金部で扱われます。こうしたものはすぐ通常の方法で処理できない性質のものでありまして証書は，支払い請求のため海外に送付されますが，集金には二つの段階があります。第一に集金の準備がなされます。すなわち証書は外国のコレクティング・バンクに送付されねばなりま

COLLECTION OPERATIONS
AUTOMATED SYSTEM FLOW



せん。そして第二にその銀行が支払いに応じる訳です。これが完了しますと顧客は，資金の回収ができるのです。このような集金部の処理量ですが，これは年に１６万件でありまして，金額にしまして２４億ドルに上っております。

１９６３年に，ＦＯＢまたは外国向け手形システム（Foreign Outword Bills system），というものがつくられました。その基礎としましては，エレクトロ・メカニカル，電子機械型データ入力装置を持ち，ペーパーテープ，バッチ処理で中央のデータ・センターのＩＢＭで行われたものです。当時としては非常に高度なシステムでしたが，だんだん時がたつにつれ周囲のビジネス環境が変わるにつれて，このようなプロセスの能力は陳腐化してしまいました。当時の処理は直線型生産モードでなされておりまして，ソフトウエアはＩＢＭの１４０１用に書かれておりましたので，これを改善して，分散型ネットワークを組むのが困難でありました。

そこで，ワーク・ステーションの概念についてお話ししましたが，分散型の自動化には，ワーク・ステーションの概念が必要であります。各ワークステーションは，明確に識別された顧客グループに合わせられまして，その処理をコントロールするために設定されております。これは処理と顧客を接近させます。このため，エラーをその発生場所ですぐに訂正することができるようになりましたし，顧客の要求を満たすことができるようになったのです。

ここで非常におもしろいことは，そのこと自体が，非常にむずかしいトレーニングの問題を含んでいたのです。つまり，たった一つの生産方法しか知らないアセンブリー・ラインの人々は全体の機能を扱うことがむずかしく，その複雑さにすぐついていくのは困難だったのです。しかし，最終的には仕事のすべての面に責任を持つことになったので自分の仕事を充実したものとすることができました。

分散処理システムを採用したもうひとつの理由はデータセンターから独立するためでした。以前は，バッチ処理，企業会計，ＭＩＳリポートのため集金業務はデータセンターによっておこなわれていました。そのチャンネルと独立を確保するためこれらの機能を自動ミニコンピュータ・システムに結合しました。

そして，達成された目標でありますが，集金業務と企業システムとの直接のインターフェイスができまして，また新しい分散企業会計システムを通じて，そのようなことができましたし，プロダクト・パフォーマンスがカスタマー・ベースで改善されましたし，また人件費，ハードウエア・コストが下がりました。

この実施にあたって一番大きなステップは既存の製品を一掃することでした。これは，顧客との直接のインターフェイスにより，特別なインストラクションが標準化されまして，フォーマット化されたインプットが達成されるようになりました。既存のシステムを一掃した後にわれわれは，次にどのシステムを採用するか検討することになりました。

そして集金業務は現在二台のＮＯＶＡⅡミニコンプロセッサーで，データエントリーはＣＲＴターミナルによって行なわれています。アプリケーション・プログラムと，不完全なデータは，ミニコンのディスクの中にたくわえられます。現存の構成は，ハードウエア的にあるいはテープを介して他の機関のシステムと結合することができるので最終的に，バンク・アドバイス（報告書）などができるような柔軟性を組み込むことができました。これによって，他のサービス処理の分野とのインターフエイスが必要がなくなり，集金部門は完全に独立した処理機関となりました。

もう一つの例としては，１９７５年に，ブローカー業務の顧客が自分のオフィスのＣＲＴを使って取引を開始することができるようにしました。これは電話線を使いまして，トラ

ンザクションを処理コンピュータにインプットするもので顧客は，自分の口座の状態を直接問い合わせることができます。

この場合のハードウエアの構成ですが，二つのデータセンターがありまして，各データセンターにはＰＤＰ11／70，ディスク・ドライブ，磁気テープ・ドライブ，プリンター，コンソールが置いてありました。

このブローカー・バンク・サービス部は，仲介業をサポートする製品としまして，信用貸付，一日の貸付・担保代替，保証小切手，振替業務などを提供しています。

この仲介業のトランザクションの中には，貸付の見返りとしての担保業とか，またローンに対する見返りの保証業務などが含まれるわけです。このシステム実施以前には，シティバンクのサービス部では必要な書類を準備して，メッセンジャーを通じて銀行に回すことが必要でした。

ところが大商いの場合になりますと，ブローカーは証券を自分の欲しいだけ必要とします。つまりブローカーとしての正常なビジネス，自分の有利になる商いをする必要があったわけです。ところが，ブローカーの貸付とは，歴史的に見て，ペーパーの発行でありまして，データの転送が多く，また多くの人と，ペーパーを必要としました。ここで最大の問題となるのはエラーです。エラーによって貴重な時間を無駄にしてしまいますし，金銭的にも損失となります。

新しいシステム，つまりブローカー・ダイレクト・インプット・システムをつくることによりまして，このようなエラーの可能性を少なくし，ペーパーベースの処理の冗長性を少なくし，また必要な資金がリアルタイムに入手できるようになりました。

このオンライン応答システムの大きな利点としましては，実時間でのブローカーの状態を知ることができるということです。現在のところ，われわれ以外のシステムでは，リアルタイムでのポジションを知ることは不可能です。せいぜい昨日または午前中の状態しか知ることはできません。

このようなリアルタイムのサービス以外に，ブローカーは過去の自己の口座の情報も知ることができます。たとえば，貸付金の副担保の投資がどのくらいできるかということ，また額面金額によりまして，それぞれのローンのセキュリティがどのくらいあるかということ，こうした情報によってブローカーは取引が容易にできるようになります。

このシステムの開発によりまして，処理は時間単位から分単位になり，顧客にとってもわれわれにとっても大幅な費用の節減ができるようになりました。これは，分散処理方式の採用によって，銀行の後方事務を顧客の特別なニーズによりうまく答えられるようにした良い例だといえます。

このブローカーのダイレクト・インプット・システムは，業界初の電子式銀行業務であり，マーケッティングの面では，シティバンクのサービスをブローカー業に売り込む確実な利点となったわけであります。また内部的にはエラーの可能性を少なくし，また一つのステーションで顧客の銀行に対するすべての要求を扱うことができるようになりました。

また，消費者に対しても，同じようなシステム，たとえば支店，または営業所とかスーパーの店頭にこのようなシステムを置くことによりまして，顧客との取引をターミナルにキー・インすることができるようになりました。そしてその処理は，現場に置かれたトランザクション処理コンピュータによりまして，地方の支店ではおそらくマイクロプロセッサーによって処理されます。この遠隔地のマイクロプロセッサーは，会計機能と，各社の報告書を作成するため，会計とＭＩＳデータを回線を通じて送ります。

これによって，支店と処理設備間の冗長性を少なくしました。またデータ・エントリー・

ポイントを，支店，自動車販売店，旅行代理店等そのサービスを必要とする顧客のそばに移動することになりました。

これ以後の5年間の計画は７１年から７６年までの経験にそったものとなるでしょう。現在われわれは，情報処理の知識，論理と，行員の技術力の程度を心得ています。こうした処理ステップは，多くのサービスに共通であることを知りました。そのため，プログラムをモジュールとして，多様なアプリケーションに繰り返して使用できるものを書くことができるのに気がつきました。ＯＳのソフトウエアに合わせた形のものとの組み合わせで，スタンダードなトランザクション処理を可能としました。

この方法の採用によって二つの大きな結果が期待されております。一つは，短時間で新しい処理システムをつくれること。いままで２年間ぐらいかかっていたものを，６カ月から９カ月で完成することができる。また二つ目は，問題をよりよく解決することができることです。

小型コンピュータは，ユーザーと処理との距離を縮めるという利点があります。また標準化の努力によりまして，トランザクションの処理と情報源，つまりユーザーのもとで処理を行うことが可能になりました。ミニコンピュータは，費用，効果的にも非常に強力なビジネス・ツールとして，われわれの要求を満たしてくれるものでありますが，マイクロプロセッサーの登場によりまして，処理の非集中化が加速されることが期待されております。

このように安価なマイクロプロセッサーにより非集中化が可能となり，われわれは顧客により接近することが可能になりました。まだ夢のような部分もありますが，すでにマイクロプロセッサーを利用して小切手処理システムが完成されております。まだ大型の機能性には欠けるところがありますけれども，これだけでも非常に興味深いものであります。近い将来，マイクロプロセッサーが，現在の電卓と同じように，銀行内で常時使用することができるようになるであろうと考えられております。ありがとうございました。

セッションⅡ　＜パネルディスカッション＞

集中処理から分散処理へ

集中処理から分散処理へ

パネリスト　高山精造[1)]
塚本和孝[2)]
柳井朗人[3)]
山本欣子[4)]
アラン・ウエバー[5)]
コーディネータ　小沢暢夫[6)]

小沢　ただいまから，「集中処理から分散処理へ」につきまして，パネル討論を始めたいと存じます。4人のパネリストの方から各自10分ないし15分ぐらいの持ち時間で御意見をお願いしたいと思っています。先ほどウェバーさんのレクチャーがございましたので，これにのっとって話を進めたいと思いますが，パネリストの方たちにウェバーさんへの質問がございましたら，その質問を皮切りに御意見を述べていただいても結構でございます。

では，一般ユーザーの立場から，高山さんからお話をお願いいたします。

高山　トップバッターで，まことにむずかしい立場でございますが，私は技術的な面は後続の先生方にお願いすることにして，本日は実務家として指名された私らしく，分散処理の具体的事例の一，二を述べながら，問題提起を，してみたいと思います。先程のウェーバーさんの話にもありましたが，私は常にコンピュータシステムはハードやソフトのいわゆる技術の進歩に追従して変化させるものではなく，その経営が求めるニーズによりシステムを展開してゆくべきだと考えています。ですから，分散か集中かの問題にも単にコンピュータ自体を分散するかしないかという技術的判断よりむしろ目的別にその必要性を考えるべきではないかと思います。データ・エントリー，すなわち人間と機械の結合点がいかにあるべきか又，データの利用面からプロセスとファイルが如何にあるべきか。もう一つは，システムとその開発の部分を集中すべきか，分散すべきかという考え方が必要ではなかろうかと思います。

そこでまず第一に申し上げる事例は，1968年，当社において総てのデータ・エントリーを現場に分散した一つの例であります。

当社では現在，ビル内約2百数カ所に分散した設備でデータ・エントリーをしていますが，その設備は決してウェバーさんのお話のような超近代的なものばかりではありません。最も古いものはパンチカード方式ですが，当社が分散した当時日本では，パンチ集中処理が常識であった時代ですから，相当思いきっ

1）伊藤忠商事情報システム室長
2）電電公社データ本部第4データ部長
3）電通国際情報サービス常務取締役
4）日本情報処理開発協会開発部長
5）シティバンク副社長
6）エム・エス・ケー・システムズ代表取締役

た方式を採ったものだと思っています。9年たった今日でも，月間約300万枚のカードを分散入力していますし又，900万枚に及ぶターンアラウンドカードファイルを事務所内に分散しています。方式は古いものですがこの分散は本当によかったと思っておりますし又，だれもこれをやめたいとは思っていません。

ではデータエントリーの分散がどのような効果をもたらしたかということを述べてみたいと思います。

一つは，日本特有のタイプの原稿や伝票が不要となったこと，すなわち重複作業がなくなったことです。

第二には，処理要員と設備コストが半減し，大幅なコストダウンが図られたことです。

三番目には，慣れた職場の担当者がみずから日常業務と並行処理するために，事前に誤りが防止でき時間的な制約がなくなると同時に，誤りに対する責任が明確になったこと，入れたいものはすぐ入るという準即時化が実現できたこと，むずかしい人事管理が容易になったこと，最大の成果は，機械システムへのユーザーの直接参加意識が徹底できたことで，今日のオンライン・データ・エントリーにまで及んだということができると思います。

しかしこの分散型データエントリーシステムにもいくつかの問題はありました。まず非常に小さくて見にくい。インターの文字をディスプレイに表現できないか。カードはインプットに時間がかかるので，磁気テープやディスケットに置きかえられないか。データの持ち込み時刻がばらばらになるがこれを統制するため離れたセンターで自動的に取り上げられないか。さらには，ターンアラウンド・データや定型的なデータが画面で反復つかえないか。多くのカードファイルを自動的に検索出来ないか，さらには，セントラル・ファイルとデータ・エントリーの端末のローカルファイルとの間にやりとりを自動化できないだろうか。さらにまた，エントリーのためにセンターの大型プロセッサーの負担を大きくしなくて済ませないだろうかといったことが今日迄の課題でありましたが，幸い，今では相当なファイルの能力を持ち，インプット・バリデーション機能を持ったインテリジェント・ターミナルあるいはキー・ツー・ディスク等が実現し，多くの望みが解決できるようになりました。われわれはいまその大部分をこれらに切りかえ，完全なディストリービューテッド・データ・エントリー・システムを具体化したいと考えているわけであります。

第二の具体例は，ファイルとプロセス，いわゆるコンピュータ自身の分散例であります。

当社の経営会議は，1970 年の始めコンピュータシステムに関する今後の方針として「大艦巨砲主義から艦隊主義へ」という方向を決めたのであります。すなわち，コンピュータは，適用業務別に，場所別に，複数設備を持つことを方針としたのです。当時はまだ，大きなことはよいことだ，より大きくなることはより安くなることだと主張されていた時代ですから思いきったことを決めたことになります。7年たちました今日，東京と大阪に全く同じコンフィギュレーションの設備を分散配置し同一階層構造を持つデータベースシステムを運営し又，この二つのセンターとともに，グループセンターとして，名古屋，ニューヨーク・シドニー・ロンドン，香港に本社と同一システムを運営出来る中型プロセッサーを持ち，各地域には端末を配置いたしまして，HIERARCHICALネットワーク方式による企業情報の国際ネットワークを着々と展開中であります。さらにまた，国際メッセージ・スイッチングにも専用のプロセッサーを持つと共にデータ・エントリー専用のミニコンを各事業所で持つ方向を進めております。

では，我々は何故このように「大艦巨砲主義から艦隊主義」に切りかえたのかということを述べてみましょう。

まず一つは，大量のトランザクション・デ

ータをセンターに伝送するために要する時間の問題です。わが国では時差を持っておりません。時差のある国では，データを伝送する時間は時差によって解決することが多くあります。しかしわが国ではいかに早い伝送スピードを持ちましても，伝送の時間というものはマイナスの時間帯になってしまうからです。

第二には，センターに日々データを集める理由が見出せなかったからです。確かにプロセスの集中はコストダウンとなります。またシステムの統一も容易で管理も一元化できます。またデータベースも一元化することが可能であります。しかし，データベースを最もひんぱんに利用するマネージメントやビジネス・オペレータたちの存在場所が問題なのです。当然トップマネージメントは本社に位置しますが経営の管理方式が中央集権的でなく事業部制がとられ部署別に分権化された企業では部門のマネージメントやビジネスオペレータが最大のデータベースユーザーなのです。即ち最も多くの情報を求めている人たちは，その情報が発生する現場すなわち，それぞれの事業所にいるということです。これを，わざわざ時間と金をかけてセンターで集中処理するよりはデータベースを分散管理し，中央が必要なもののみをそのつど集めた方がよいと考えたからです。更に電送のコストダウンよりコンピュータのコストダウンの方がはやいとよんだからです。私達の分散はシティバンクの様な大がかりなものではありませんが幾つかの効果を生みました。まず一つは，複雑多岐にわたる取引形態を持ち，激しい環境の変化にたえず対処しなければならない商社のマネージメント達のわがままな要望にこたえられるようなシステムが出来ました。我々の 情報システムは小廻りがきき，レスポンスの速さが必要ですが，プロセッサーが利用者に接近しておればおる程この望みがかなえやすくなると思います。又，ユーザーの直接参加が可能となり，システムが自然に組織にとけ込むようになりました。第二には，通信コストに比べ大巾に低下したプロセッサーの処理能力をフルに利用できかつ大量のトランザクションデータの送受にむだな時間を費やさなくて済んだことです。さらには，最新の設備でやるシステムのみが最善だという考え方ではなく，古いシステムでも良いシステムは，その容量が不足したり，あるいは採算に合わなくなるまでは，有効なシステムとしていく覚悟ができコンバージョンに要するシステムエンジニアを新システムの開発にかけられるようになったことや，データベースの構造が大きくなり，複雑になる程，事故を起こしやすいという問題がカバーできたこと，最後には，完全なシステムの保全体制がとれたということであります。このように艦隊主義をとった東阪分散配置方式は多くの効果を上げることが出来ました。又，北米20都市に端末を分散したニューヨークのリモートファイル・アクセス・システムも逐次その成果を上げています。一方欧州ではローカル・プロセッサーをロンドンに置き，ニューヨークの設備をセントラルプロセッサーとして 時差を利用した階層型ネットワークシステムを試行して参りました。しかしこのシステムは，ニューヨークとロンドンの時差を有効に利用することは出来ましたが最近になって中央コンピュータの使用料と端末のコストと，それから伝送にかける費用の合計よりロンドンに専用のプロセッサーを置いた方が安くなるようになってしまいました。この点大きいものほど安あがりという考え方は，時々考えなおさなければいけないという教訓になったように私は思うわけであります。

私は，分散化の傾向は必然的なものだと思います。その理由は，何といいましても，情報の利用者，つまりユーザーが，より適切な時期に，より適切なものを求めるからだと思います。第二の理由は，その方がコスト安の場合が多くなったからであろうと思います。

第三の理由は，メーカーがこれらの機器を大量生産して，よりよいものをより安く，しかもより多く販売しようと努力しているからであります。われわれは，これをそのままうのみにするのではなく，その目的に合った，方式を選ぶべきだと思います。コンピュータ化の目的は機械化ではなく合理化にあるのです。本当に分散が合理化を生むものであるならば，勇気をふるって分散化すべきではなかろうかと考える次第であります。

小沢　分散処理にネットワークを外しては考えられませんので，電電公社の塚本さんからひとつ御意見を承りたいと思います。

塚本　集中処理から分散処理についてというふうなお話ですけれども，まず，集中と分散という二つの相反する命題ですが，これは非常に古くて新しい命題であって，何もコンピュータに限ってこういう事実があるわけではない。二，三の例を挙げてみますと，私どもがサービスを提供しております電話交換機あるいは加入電信の交換機といったものの集中，分散というものが，身近な例としてまずございます。この交換機というものは，一定の地域ごとに，電話の加入者の分布と，その伝送路あるいは線路，そういったもののコストとの関連，それから一日に何回ぐらい電話を利用するか，これはトラフィックといっておりますけれども，こういうことの経済的なバランスによって交換機の設置を決めていく。したがって，集中と分散というものを全国的に地理的な分布に応じて配置していくということになります。大都市には幾つかの電話局，さらに東京のような大都市になると何十という電話局があり，さらに巨大な高層ビルとか，そういうビルになると，その昔の電話局並みのビル内の専用交換機，こういうものがさらに置かれている。これらすべて，いまのトラフィックと伝送路というような経済性によって，それぞれ集中，分散というものが適宜配置されているわけであります。

二つ目の事例では，これは余り極端な例ではないかもしれませんが，昔，エネルギーの電力問題として水主火従ということがあって，要するに水力のあるところに発電所を設けて，あとはそれに送配電の技術というものがいろいろサポートするということであったのですが，送配電技術が非常に発展した現在では，あるいはまた火主水従と申しますか，最近は原子力ということにもなってまいりましたけれども，水力というものはかなり従になってまいりました。そうすると，火力発電というもののディストリビューションというのは送配電というものの技術との調和で，理想的な広域分散，こういう配置が可能になってきている。将来，太陽エネルギーという問題を議論されていく場合には，逆に個別の発電をして，それを，必要な場合には発電所の方へ送り返すといいますか，そういうふうなアイデアも一部言われているようなことでありまして，現在はやはり集中型の方に片寄ってはいますが，火主水従から，かなり分散型のエネルギーの供給という形になっている。

三番目の例では，動力の分散，集中ということがあります。

昔の動力というものは，非常に大きなモーターで，ベルトで伝達したわけでございます。これは，戦時中の勤労動員あたりで行かれた方は，ベルトで旋盤を駆動したといった例はおそらく御存じだと思います。現在そういう機械を見ることは全くない。すべての工作機械には必要なところにふんだんにモーターを使う。それは，そういう小型のモーターというものが，非常に安い価格でふんだんに供給されるという事実があったからであります。

もう一つの例としては，この動力では，鉄道における機関車と電車，こういう関係においても感ぜられます。蒸気機関から電化ということになってまいりました。しかしながら，蒸気機関はすでになくなったわけでありますけれども，電気機関車自体も，すでにほとん

ど貨物の専門の機関車になっている，そういう事実であります。新幹線等は，分散型の動力なくしては実現し得なかった。これもやはり動力の分散といういい例であります。

なぜそういう例をいままで申し上げたかといいますと，その分散を支える一つの素地なり，そういう技術というものの発展なくしては，やはり分散化ということはなかなかむずかしい面もあったという事例が，ここにいろいろあったということであります。これは，動力，エネルギーなんかが非常にいい例であります。

四番目に，今日ここで議論するデータ処理の集中処理と分散処理ということがあります。この見方は，いまから十数年前に集中処理というのは，ＰＣＳなりＥＤＰＳのいわゆるバッチのデータ処理という形が，集中処理という概念であったわけであります。ところが，古い言葉でありますが，ＩＤＰ，インテグレーテッド・データ・プロセシングという言葉がございまして，これが一種のデータ・エントリーでございますね，それの分散的な集中処理ということで出てまいりました。あるいはデータ・ギャザリングというような言葉が出てまいりました。しかしその当時は，やはり分散処理というツールはなかなか出てまいらなかったわけでありますが，いまから７～８年前ですか，ミニコンというものがこの世にあらわれて，この数年の間に急速な進展を示すに至って，分散処理というものが大幅にクローズアップしてきた。そのために，この２～３年間にいわゆる大型の電子計算機というものが，グロッシュの法則によって，そのパフォーマンスが倍になればその価格は二分の一乗である，何割かアップするというようなルールがあるわけですが，それがなかなかのらなくなってきたということが言われております。これは，ミニコン，あるいはつい最近大きくまた普及してまいりましたマイクロ・プロセッサー，こういうものの技術の進展によって大きな変貌を遂げてきた。要するに，これは分散処理というものを実現するツールが開発され，経済的にも，いまのグロッシュが崩れるということがいわれるように，実現したために，いよいよかねてからの念願であった分散処理が実現してきたのだ，道が開けてきたのだ，こういうふうに考えております。

ところで，非常にこういうものが発展する段階では，大型機もそろそろ要らなくなるんじゃないかというような極端な議論もいろいろされるわけでありますけれども，そうはなかなかならないであろう。この点で，やはり集中処理というものとの，一つのそれぞれの持ち分といいますか，存在意義というものを考えなければいけない。

集中処理と分散処理というものの特性を非常に大ざっぱに眺めてみますと，まず，集中と分散と分けまして，ファイルの問題を考えますと，集中型のファイルというのでは，国鉄のみどりの窓口で使われているような予約型のファイルといったようなもの，あるいは旅行サービスの予約といったものは，インテグレーテッド・データ・プロセシングと同じようなデータ・エントリーを広めるだけでありまして，ファイルはやはり集中化せざるを得ない。あるいはオーダー・エントリーとかで製造管理を行うといった場合にも，自動車とか，いろいろな機械工業とかありますが，そういう場合にも，ファイルというものの一元化ということは非常に重要であります。

さらには，情報検索というようなデータベース，こういったようなものについては，非常に値段もかかりますし，ファイル・メンテナンスというものも大変でありますから，集中型の方に属するかと思います。

一方，分散型のファイルというのはどうかといいますと，顧客ファイル，こういったものは非常に地理的な，営業的な分散ということになりますので，これは分散型の方の形になるであろう。あるいは会社内，企業内の事

務管理用の人事とか経理とか金とか，そういう分野のファイルというのはかなり分散型になるんではないか。

一方，集中型の処理という点では，これは非常に大ざっぱで恐縮なんですが，グロッシュの法則が成り立つならば，そういうものは集中型処理であろう。あるいは，アメリカにおけるイリアックⅣに見られるような非常に高度なプロセス機能というものが大型機によってこそなされるという場合には，これは集中型ということになるんではないだろうか。あるいはまたライブラリーといったようなものについては，これは値段にもよりますが，集中化が適してくるんではないか。それから分散型の処理では，これは逆でありまして，グロッシュの法則が成り立たないもの。成り立つか成り立たないかというのは，実は，一つは価格政策の問題と，一つは処理の内容ということによって変わってくるのではないかというふうに，最近見られております。したがって，グロッシュの法則が成り立たないようなものは，やはり分散処理ということに向いてくるんではないか。たとえば単純な反復処理とか，ミニコンがカバーするいろいろな事務計算の処理の分野，それほど大容量でないファイルの処理といったものは，分散型に非常に適する。

こういうことによって，集中処理と分散処理というものがいろいろなかっこうで組み合わされてくるわけでありますが，一つは大型の集中処理，たとえばＴＳＳサービスとか，あるいは予約型のファイルのサービス，それから二番目には，大型プラス小型のネットワーク処理，これはバンキングなどのようなものであります。小型と申しましたけれども，これは，最近インテリジェント・ターミナルというのが盛んに利用されつつあります。こういうものがだんだんファイル機能等を充足して集中化した大型のデータベースと，それから各店舗ごとの小型のインテリジェント・ターミナルという形で，だんだんネットワーク型になってくるだろう。

それから三番目は，大型の分散処理といったような形で，これは官庁とか大企業のようなものになるんではないか。もともとこれは組織上あるいはその企業の性格上，行政的な面もありまして，どうしても集中化するのはメリットがないというような場合には，大型，中型というものをそれぞれの管轄に分けて処理するという形が今後もとられるのではないか。

それから四番目には，大型ネットワーク処理として，ＡＲＰＡとか，それからこれは米国の航空管制の例でありますが，ＮＡＳステージＡというのがあります。これは20の管制センターに大型を置いてあります。各管制管轄区ごとに大型を置いていく。これを完全にネットワーク化して，分担処理する。

五番目には，小型の分散処理として，オフィス・コンピュータというものを適宜置いていく。これは，一つの事業所の中に何か所置いてもかまわないというぐらいの徹底した小型の分散処理という形。あるいは小型のネットワークの処理というものが考えられます。

こういう形でいろんな組み合わせというものがありますが，ネットワークの構造として見ますと，これは一般的に言うのは階層構造という形，これは電話交換機なんかの形に似ておりますが，そういう形が多いし，あるいはまたメッシュ型の構造というものも考えられます。

いずれにしても，交換という機能は何かといいますと，それはルートを決めるということであります。ですから，そういう機能をみんなネットワークの中に組み込んでいく。そういう面で従来，現在の回線のサービスでありますのが，専用線あるいは特定通信回線というものがある。これは月決めという形でありますが，いま一つは公衆回線，つまり普通の電話回線を使う。これはトラフィック，つ

まりメッセージの量に応じて使っていく。しかしながら，こういう面で見た場合に，一つはまだまだディジタルというベースに完全にはなり切っていないわけであります。それともう一つは，電話交換というものが発信者がお金を払う。センターでお金を払うというふうには必ずしもなっていない。あるいは1対1，ポイント・ツー・ポイントの接続というものを前提としている。それから，ディジタル通信あるいはスピードにある程度の制約がある。こういったようなことから，現在DDXとわれわれは申しておりますが，ディジタル・データ交換網の新規サービスというものを実験し，近いうちにそれを提供しようということで考えております。

集中と分散というものは，いま申しましたようないろんな歴史的な経緯あるいは新しい技術革新ということによって，その道が開け，現在は一つの発展期でございますから，今後集中，分散というものが，その利用形態，目的に応じて，非常に効率的な使われ方というふうになるであろう。私どもとしても，それに見合うような回線のサービスということを現在いろいろ研究し，やっている段階でございます。

小沢　ありがとうございました。次は，世界的ネットワークをもって仕事をされておられる柳井さんから，ひとつ御意見を承りたいと思います。

柳井　私の方，現在，グローバル・ネットワークというものがございまして，国際間のデータ処理のサービスをやっておるわけなんですが，その前に，いわゆる遠隔情報処理サービスの発展過程と，きょうのウェバーさんの話を比較しながら考えてみたいと思います。普通日本でも，タイムシェアリング・サービスという形で，IBM，GE系，あるいは電電公社系と，この数年間にそういう形が出てきたわけです。よく考えてみますと，こういうサービスは，サービスが発生する初期の段階から，情報の分散の処理のために，顧客にとっては分散した情報処理という形で業務形態が発達してきたわけです。やっている内容は，技術の計算あるいは問題解決型，シュミレーションの問題を解く。それはたとえばウェバーさんでいえば，スター型のディストリービュートされた形のセンターを使うということで，これが商売として成り立ってきたわけです。現在でも，そういうスター型にネットを張りながら，そのネットの先端に端末を置くことによって，多くの人が，不特定多数の人が計算機の能力を利用する。商売としてもたくさんの人が使うからコスト安く提供し，商売としてもそれが収入となるということで発達してきたわけです。これは集中型であるかどうであるかというのは，見方によって違うわけなんですが，業務上では，それは分散タイプとして業務を考えてきたわけです。

そういう問題解決型だとか，大きな技術計算を中心にしてユーザーが端末を使って，必要なときに計算するサービスというものが起こって，それが徐々に情報処理データ・プロセスの方に要望がふえてきたときに，少しずつ問題が出てまいりまして，最初は端末に機能がなくて，ただスター型のネットワークを利用して中央のファイルにデータを蓄えながらデータの集中をして処理をしているというタイプのサービスというのが，アメリカにも幾つか出てまいりました。あるいはウェバーさんの銀行のように非常に巨大な銀行，そういうようなところでは，そういうものを採用するということは，あるいはコスト的に全体の利用度からいって得策ではないと思われますので，実際アメリカでそういう遠隔情報処理サービスを利用してデータ・プロセスするという場合には，自社でそういうシステムを持つよりも，そういうサービスを利用した方が，コスト的に見合うという顧客がそういうものを使っております。これは，アメリカでここ4～5年の間に非常に成長しております。

年間成長率が40％以上確実に伸びてきております。

ここで必然的な顧客の要望として次第に，端末自身をインテリジェンス性を持たさなければ，どうしても回線上にあらゆるデータを中央に送ることに対するコストというものが問題になってきて，顧客の中では，何とか端末側にインテリジェンス性を持たせる必要があるんじゃないかという要望も出てきておりまして，ここ１～２年前ぐらいから要望されてきておりますのは，端末側にインテリジェンス性を持たせて，そういうサービスへの，中央へのアクセスのスピードを上げてくる。1200ボー，2400，4800，9,600と，徐々に顧客からのセンターのコンピュータ・パワーを利用するスピードを上げてきております。

ここで，ウェバーさんの御説明なさったような第二の段階，階層別の構造を持ったような遠隔情報処理サービスということが，ここ１，２年アメリカでも出てきております。われわれが現在日本でサービスしておりますマークⅢにおいても，これはＧＥだけか，あるいは一般に通ずるかどうかわかりませんが，われわれはそれをインター・プロセシングと称しております。顧客側のＣＰＵと情報処理サービスのＣＰＵを結んでしまうというようなサービスの要望が出てきておりまして，徐々にその要望にこたえつつある形です。

この際も，塚本さんからもお話ございましたけれども，顧客の要望というのは，ほとんどメインのファイルにデータをコンソリデートするというような要望のためにサービスを利用する形態が多いようです。わが国ではまだわれわれサービスしておりませんが，やがてここ１～２年にそういうものが電電公社も，ほかのＩＢＭもＧＥも全部考えていくのは，商売上当然の方向だと思います。情報処理の場合，完全に分散してしまうということが企業ではなかなかできなくて，セントラル・ファイルというものに企業としてのコーポレート・データというものを集めるというは，どうしても情報処理の形では必要なんです。これは，その次にウェバーさんがおっしゃった完全なディストリビュートされた，それがミニコンであろうが，大型計算機であろうが，そういうような形においても何らかの形でどこかでファイルにデータを集約するということは，情報処理の全体から，データプロセスの場合，無視することはできないと思いますので，こういう情報処理サービスというものは，全体として顧客が自分でメンテナンスするには，非常にコスト，それから人，そういうような問題で，そういうサービス機関を使って，そこが専門的に大きな構造のファイルを自由に使えるようなものを提供するという形で，私は情報処理サービス業というものは向かっていくものだと思います。いずれも，そういうものを使ったコストが安いのか，あるいは自分で全部完全にインハウスで設計して，それを運用するということが安いのかということは，やはり効率とコストの問題で決まってくると思います。現在顧客というのは非常に賢明でありますから，みずから選択していくものだと思っております。

もう一つ，世界あるいは最近の大企業は，グローバルに営業所が点在をしておりまして，そこで起こった情報処理というものをローカルで処理するという方向へ向かっている形が現在あるわけなんですが，やはりそのグローバルを一つの固まりとして情報を集約するということは，企業活動の方向としては絶対的であろうと思うんです。現在，各国別に業務をやっておりましても，全体の総合の本社においてデータを集めて，そこで処理をする，そういうような問題が幾つかあると思います。

この際に一つ問題になりますのは，そういうグローバルのネットをインハウスで持つということ，自社だけで持つということは，非常にコスト的にむずかしい問題あるいは運用上むずかしい問題がございまして，現在のと

ころは，世界じゅう見渡しましても，自企業で，飛行機の予約とか座席予約とか，そういうリアルタイムの処理を必要とするものは，どうしてもインハウスで整えているケースがございますが，それ以外の場合は，コスト上の問題から，どうしてもどこかのネットを利用するということが起こってきておりますし，将来とも，そういうものがなくなって，全部インハウスで世界じゅうネットを結ぶというようなことは，なかなかできないのではないかと想像しております。

われわれの方のネットは，やはり共同使用するということによってコストの低減を図っておりまして，グローバルに，先ほど言いましたスター型のディストリビュートした形か，あるいは階層別のネットを張っていくという形をとっております。先月でしたか，新聞紙上をにぎわしましたＣＤＣにしても，タイムシェアといいますか，このような大きな情報処理サービス業務においても，やはりグローバルにネットを張らないと，業務上非常に問題を生じるというようなことに，米国の遠隔情報処理サービスというのはなってきております。これは，一に企業活動がグローバルになりまして，グローバルの個々の処理された情報を集約するということから起こっていると思います。

もう一つ，端末の側なんですが，こういう遠隔情報処理サービスにおいても，端末にインテリジェンス性を持たす，あるいは小スケールで情報分散処理をするということを無視するわけにいかないので，それをどういう形で既存のサービスに集約していくかということが問題になるわけですが，全体の方向としましては，できるだけ吸収していくタイプになってきております。

このインテリジェンス・ターミナルというのは，世界の情勢を見ておりましても，最近，マイクロ・プロセッサーその他が手軽に扱えるようになって，急速に種類がふえてきております。特殊性のデータ・エントリーの端末が，日本でも毎月新型が出るというような形で，こういう分野においては，案外アメリカよりも，自動車産業の輸出が伸びている形も，こういうインテリジェンス性の端末というのは，日本の現在の産業のレベルからして，非常に取り扱いやすくて，しかもトップレベルに出てくるのも時間の問題だと思っております。そういう意味での，そういうものを利用した分散処理というのは，日本で非常にユニークなものが出てくるでしょう。

ただし，後でお話になりましたようなＡＲＰＡ型のネットワークを自分でデザインして運用していくのには，まだ技術上非常にＯＳの問題，その他プロトコル，いろんな問題で，まだ研究課題で，私企業でそれを自分で完全に運用される，ウェバーさんのような形で出てくるのは，わが国でも少し先のことと思いますし，当然そちらの方向に行くと思います。そういうことになった場合の遠隔情報処理サービスは，なおそういう形であっても，集約されたファイルを提供するというような形では存続していくものだと思います。

特に最近の動きで，外部の情報を集約したデータベースというものは，こういうサービスが提供する大きな分野でございまして，現在アメリカでも，そういう外部情報を遠隔情報処理サービスが提供しているケースは非常に多うございまして，日本でも，東京証券取引所の株価のデータが，即刻ではありませんが，その日終了いたしますとマークⅢに入っておりまして，それが利用者が利用できる。又，最近ですが，シカゴの商品相場がマークⅢにその日の終わりに入りまして，世界じゅうが情報を利用できるというようなこともありますし，為替相場も，ロンドン，ニューヨークという為替相場が徐々に中央の情報処理サービス業のファイルに集約されて，それを世界じゅうが使う，こういう形で利用度が高まってきております。こういうような形で，

遠隔情報処理サービスとしても，将来ＡＲＰＡネットワークのようなコンピュータが全部つながって処理をされるにしても，やはり外部情報を提供する，あるいは自社でそこまでコストをかけられない人のための分散データ処理と，集約に関する道具を提供するというようなことについては，将来とも変わっていかない，こういうぐあいに思っております。一応私の方は，[illegible]ている情報処理サービスが，こういう分散処理が発達した後も，サービスというものは存続するということを御説明したいと思います。

小沢　今度は，ウェバーさんの言われるような分散ネットワークを研究されておられる山本さんに御意見を承りたいと思います。

山本　今回このパネラーとして出席しなければいけないということになりましたときに，まず，分散処理システムというのは一体どういう定義なのか，分散処理システムはどういう形で実現され，とらえられているのだろうかということが気になりまして，たまたま二，三日前，手元に来た新しい雑誌が分散処理システム特集号のような格好になっておりましたので，読んでみました。これはいくつかのメーカーの方々が，それぞれの分散処理システムについての紹介をしておられるわけです。それを一通り見てびっくりしたわけですが，非常にいろいろな解釈があるということです。

一つは，先ほどウェバーさんの話にもありました，分散処理と非集中処理の問題がからんでくるわけですが，ディストリビューテッド・システム（分散処理）というのは，ディセントラライズド・システム（非集中処理）とセントラライズド・システム（集中処理）の両方の長所を統合したものであるという意見が一つありました。

もう一つの意見は，非集中処理は処理の分散の方式が固定的であるのに対して，分散処理はダイナミックに処理の分散の度合いとか方法を変化し得るのだというような話も出ているわけです。

またネットワークとしての形からいうと，集中型分散システム，それから分散型分散システムなどというのがどうもあるらしい。それから先ほどウェバーさんのお話で，階層型分散システムというものもあり，ネットワークの形としてもこれまたいろいろあるのだということです。

もう一つの別の意見としては，いわゆるホスト・プロセッサーとフロント・エンド・プロセッサーと，その先に幾つかのディストリビュートされたプロセッサーというものがばらまかれている。それらの中にあるおのおののソフトウェアがはっきりとモジュール化されていて，たとえばそれぞれのＯＳというのは一体どういうところをシェアリングして分担しているのか，通信制御用のプログラムがそれぞれあるわけですが，それがどういうように分担されているのか，業務処理のプログラムはどことどこで何の処理を受け持つかという役割というものがはっきり分散され，すなわち，分担されているのが分散処理システムなのだ。これがソフトウェアの分散ということであろうかと思います。

そうかと思いますと，もう少し変った意見としては，一つのコンピュータ・システムの中で，非常に高速に演算処理をやるようなパイプライン・プロセッサーのようなものが一つあり，また，ファイルをいろいろマネージするためのファイル・プロセッサーというものもあります。ＩＯを処理するＩＯプロセッサーというものもある。それからコミュニケーションのための通信制御用のプロセッサーがある。

こういった機能別のプロセッサーがたくさん存在していまして，そういうものが，たとえばバス制御機構というようなものを介して相互にデータを転送し合いながら一つの処理をしていくのだというような，いわゆる複合プロセッサー，これを機能分散型という意味

で分散処理システムと呼ぶのだというような定義もあります。

これの発展型といたしましては，おのおののプロセッサーがある程度地理的に離れたところに存在するという形も将来は考えられるかと思うのですが，このように分散処理システムという名前から連想されるものは，実はいろいろあるわけです。そこで，実際には分散処理システムとはどういうものかというような定義するのは愚かな話であって，やはりそれぞれの企業あるいはそれぞれの目的によって自分に合った分散処理のやり方を考えればよろしいので，統一した定義などというものはむしろないのがあたりまえだという感想を持ったわけですが，ただ，その雑誌の中で，二つばかりの意見は私はどうもいまだに反対なのですが，その一つは，処理は分散しているけれどもコントロールはあくまでも集中しているのだという意見と，分散システムとはいえ，データ・ベースの集中化とその集中管理というものは基本的な概念なのだというような意見です。これらについてはいささか私は異論を持たざるを得なかったわけであります。

そもそも分散システムの発生というのは，先ほど来いろいろ話が出ておりますけれども素子の集積化技術というものが非常に進歩してきましたために，ハードウェアのコストが下がった，ミニ・コンピュータが非常に容易に手に入るようになった，しかしながら一方通信のコストはそれほどまだ低下はしていないということで，分散した小型システムとか，インテリジェント端末などの設置のメリットがますます増大した結果，分散処理システムというものが出てきたという話が基本的にあるようですけれども，メモリーというようなものに関しましても，単価はどんどん安くなってきておりますが，いわゆるオーバーヘッドのコストがある程度下がりさえすれば，データ・ベースの分散化も当然問題になってくるし，メリットも出てくるのではないかと思います。しかしながら，現在のところ，分散型データ・ベースというのは，コントロール上非常にむずかしい問題を持っておりまして，そういうところから先ほどのようなデータ・ベースはあくまでも集中するのだというような意見もあるいは出てきたのかもしれないと思います。

いままでの方々は，かなり大きな立場からお話しいただいたのですが，私は，多少細かい技術的なところをつつく結果になるかと思いますけれども，分散型データ・ベースあるいは分散型ファイルというような問題で若干技術的な問題を話させていただきたいと思います。

分散型データ・ベースは，大ざっぱに分けると二つに分類されるであろうと思われます。一つは，すでに存在しているいろいろなデータ・ベースをお互いにシェアリングしようという形のリソース・シェアリングの立場から来た分散型データ・ベースが一つあると思います。

もう一つは，単一のデータ・ベースであっても，これは新たにつくる場合，目的に応じて地理的または機能的にいろいろなところに分散してデータを配置する，これが企業内あるいは専用の分散型ネットワークの形と考えられます。

どちらかといいますと，きょうの話は後者の方に主として関係があると思いますので，そちらを中心に，またある程度両者に共通した問題点も含めて若干あげてみたいと思います。

一つはデータの最適配置という問題であります。配置を決める要因としては，データの蓄積のコスト，いわゆるファイルのメモリーの容量との関係が一つあります。

それからそのファイルをどこかに送るということから考えますと，伝送のコストが当然一つの要素となります。また一体どのくらい

アクセスされるのかという，アクセスの頻度の問題がもう一つあります。それから一体アクセスがどういうところで発生するかというアクセスの発生場所という問題がありまして，こういうものが当然基本的な要素と考えられるわけですけれども，それ以外に，最近は安全性のためにコピーを持つということで，コピーの必要性と，一体そのコピーをどのくらいどこに配置しておいたらいいだろうかという問題，それからアクセスされる内容が問い合わせなのか，あるいは更新なのかという区別，要するに，ファイルを読むだけなのか，ファイルに書くということも伴うのかというような要素，それから更新の頻度，またその更新をやるのに適した場所がどこなのか，さらにプログラム・ファイルとデータ・ファイルとの関係として，プログラムのあるところにデータを運んでいって処理した方がいいのか，データのあるところにプログラムを運んでいってそこで処理させてしまった方がいいのか，そういう問題も最近では出てきているわけでありまして，こういうファクターを入れて非線形の数学モデルをつくった例は，最近いろいろ出ているわけです。これは一般解として求めようとしますと，複雑な話になるわけですけれども，企業内システムなどの場合は具体的なアプリケーションに即して幾つかのファクターはしぼって考えることが可能だと思われます。

いろいろの方の経験を伺いますと，ローカルのみで必要であるというデータは予想以上に多いのだ，そこでそもそも分散型処理システムというのが発生したのだということですがむしろ問題なのは，これに関連しまして，ファイルのディレクトリーをどういう形で持つのがいいかということが問題になってくるのではないかと思われます。ディレクトリーをすべて一カ所に集中するのはやはりまずいのではないか。ローカルなものはローカルにディレクトリーを持たせる，コントロール自身もやはりある程度分散させることが必要になるのではないかと思いますがローカルなディレクトリーを設置した場所を何らかの形でロジカルなループをつくり，ファイルを探しやすくするような方法あるいはディレクトリーのデュプリケーションを行うというようなことも考える必要があるのではないかと思います。

もう一つの問題は，ファイルを共用する場合のアクセスの制御法とか，いわゆるデッドロックを回避するという問題があるわけで，二つ以上のプロセスが同時に同一のファイルのリソースにアクセスをいたしますと，そこでデッドロックが起こる可能性があるというのが常識になっているわけですが，デッドロックを防止する方法，これは一般のデータ・ベースでもいまいろいろ問題になっておりますが，最善の方法はやはり原則的にはデッドロックが発生する可能性があるような使用法を禁止すればいいというのがまず基本的な考え方になると思います。たとえばシェアード（Shared）と，エクスクルーシブ（Exclusive）と二つのアクセスの方法がデータ・ベースには一般にありますけれども，当然のことですが，一つのファイルへのエクスクルーシブな使用は，同時には一つのプロセスにしか許さないという方針が最も簡単な解決法であるかと思います。

それから現在とられているアプローチとしましては，あるプロセスが実行を開始する場合には，そのプロセスに必要なファイルを全部登録させまして，現在それがアクセス可能であるかどうかという事を調べ，可能であれば初めてそのプロセスが動き出すことが許されるというような方法がとられているわけですが，特に分散型データ・ベースの場合には，ほかのサイトにおけるリモート・ファイル・アクセスに対する制御を一体どこで行ったらよいのかというのがむしろ問題になると思いますが，これも先ほど申しましたディレクト

リーの問題と同じように，やはり制御が分散される必要があるのではないかと思われます。

もう一つの問題がいわゆる仮想データ・ベースという概念でありまして，ちょっと先走った話になるかと思いますが，ネットワーク内に分散しているすべてのデータを，ユーザーから見た場合にはいわゆるロジカルには一元的なものとみなして使えるという機能であります。

すなわち，ユーザーは，それぞれのデータ・ファイルが物理的にどこに置かれているか，そのようなことは何も意識しないでよろしい，すべて標準化されたアクセス法あるいは操作言語を使って一つの非常に大きな仮想データ・ベースというのが存在しているのだという感覚で使ってよろしいというような考え方であります。

現在のこういう方法に対するアプローチとしましては，従来のジョブ・コントロール・ランゲージを含みました標準的データ・ベース処理言語というようなものを設定いたしまして，ユーザーはその言語で自由に操作をおこない各サイトにおきまして，物理的あるいはロジカルな対応にコンバートするというのが，現在の可能な方法として考えられ，方々で実験はされております。ただ，これに伴いまして，いわゆる物理的な問題だけではなくて，むしろデータの中身のセマンティックスの問題に対する変換というのがあるわけです。たとえば同じものをこのデータ・ベースではこう呼んでいるけれども，別のデータ・ベースでは違う名前で呼んでいるということがあるわけです。そこで，インフォメーション・リトリーバルにおけるシソーラスを変換するのに相当するような問題が出てくるのではないかと思います。

例えばＡＲＰＡネットワーク形式のような汎用のコンピュータ・ネットワークで仮想データ・ベースというものを汎用的な概念で考えて処理しようとしますと，これは大変な話になるわけですがただ企業内の分散システムとか専用のネットワークにおきましては，データ・ベースの共有の具体的なニーズはおそらくかなりしぼられているのではないか。したがって，標準的な操作言語をつくるというところまではいかなくとも，論理レベルあるいは物理レベルのデータ構造をあらかじめ統一する可能性もあるかと思われますので，そういう意味ではかなり実用性の高い仮想データ・ベースの設計も可能なのではないかというような気がいたします。

小沢　どうもありがとうございました。

皆さん，時間をきっちり守っていただいたので，若干余裕がありますので，高山さん，ひとつまだ残しがあるようですから……。

高山　先ほど途中でしたのであと少し時間をいただきます。

私は先程，合理的なものは思いきって分散すべきではなかろうかとの意見を述べましたがこの分散化にはいくつかの前提条件があります。

まず一つは，システムの統制と汎用化という前提条件です。このことは，メインテナンスの合理性，運営の合理性，開発の合理性，展開と管理の合理性から必要なのです。統制と汎用化のための方法についてその１～２の具体例をあげてみましょう。その一つはデータ・ベース指向だと思います。データ・ベースつまり同一構造のファイルを持ったシステムの場合，分散しても一つのシステム体系として維持管理することが出来ます。当社では大阪，東京はもとより，名古屋，ニューヨーク，シドニー，香港，ロンドンの各地で同一ファイル構造のデータ・ベースシステムを維持してきました。その結果，非常に有利であったという点が証明できます。

もう一つは小さな具体例ですが，オンライン画面の汎用化であります。私は，よくコクヨの帳簿をたとえシステムの汎用化を進めさせていますがコクヨの帳簿は，どこで買っ

ても同じデザインです。ただ，ヘディングを商品名にすれば在庫台帳にもなるだろうし，ヘディングを得意先にすれば得意先元帳になるがごとく，オンラインのシステムも業務毎に単独に設計するのではなくて，一つの汎用的な画面をつくり，これをいろいろのシステムで共用していけば，システム開発も容易でシステムが分散してもコントロールしやすいことは申すまでもありません。

第二の問題は，業務分担に基準を設けることです。これがなければ分散化は幾らでも深く広くなってしまうと思います。たとえばデータ・エントリーのみだとか，現場におけるオペレーショナルの業務だけにこれを制限するのも一つの方法です。

第三は，受け持ち範囲の制限です。どこのどの機能を，どの組織を，どの場所をということを限定する必要があろうかと思います。第二と第三が無視されると分散は又，集中化となってしまうでしょう。

第四には，専門のＳＥやオペレーターが不要なシステムであるということが条件ではなかろうかと思います。当社の場合，現地の人でもちょっと教えればそのままやってくれるというシステム設計をしています。

第五には，開発や改善へユーザーの直接参加をルールづける必要があると思います。

最後には，センター・ファイルとローカル・ファイルの分担とダブリの明確化も一つの条件です。特にダブリファイルは常に一致していることが常識ですが分散の場合ファイルには時間的にずれがあり不一致の状態が前提条件となる場合も少なくありません。

この他分散化をするためには，いろいろ解決しなければならない問題も少なくないと思います。たとえばより大きくより新しいハードやソフトを使うことを夢みたシステムエンジニア達に分散化に必要な小型のシステム開発にも興味をもたす教育が必要でしょう。又，ユーザーが，いわゆるコンピュータ・アレルギーを起こさないような配慮も必要になってくると思います。

よく申すことですが，「コンピュータ」という表現は「乗り物」という総称と同じだと思います。自動車，電車，飛行機という表現なればその目的もわかりやすくアレルギーも起こらないのですがすべての機械にコンピュータという名前がつけられるために，非常に小さな自転車のようなコンピュータも大型と間違えられていることが，原因になるのではなかろうかと思うのです。分散型設備には，テレビや電話機と同様に気安い表現が必要ではなかろうかと思います。

私はコンピュータシステムはユーザーの満足度により評価されるものと信じています。ユーザにコンピュータシステムを満足させるためには親しみを持たせることが必要です。そのためには常にユーザー・オリエンテッドな配慮が必要で開発の分野にはユーザーが直接参画するという方向をとるべきです。

当社では前述の如くデータエントリーとプロセスアンドファイルは分散主義です。しかしシステムとその開発は統一的でかつ集中化を維持しています。これは先程も述べた通り開発と運営管理の合理化のためですが集中的な開発はユーザーの親しみを欠きやすいものです。そこで当社では開発は人間系システムと機械系システムに分断し，人間系システムは総てユーザーが主体でかつ機械系に優先さすことを条件としています。人間系システムとは，組織規程，制度，基準，手続といったものですがかつてこれらは機械系システムができてから手がつけられる場合が少くありませんでした。コンピュータのプロは必ずしも，人間系システムのプロではありません。会計，生産，販売等それぞれの専門家つまり人間系のプロと機械系のプロが一つのシステム開発を分業協力して始めてユーザーオリエンテッドのシステムが生まれるものと信じています。

私は，「集中から分散へ」ではなくて，「

分散と集中」ではないかと思います。すなわち，それぞれその企業の環境の特性と経営の体系に合致しかつ真の合理化を追求できるような方式を選ぶべきだと思うのです。

確かに，ミニコンの性能は上がり，コストは安くなりました。しかし「集中から分散へ」と叫ばれるとかつてのＭＩＳやＩＤＰブームの如く分散化ブームとなりそうに思います。分散はよいのだからどんどん分散してしまえということになりますと，過ちを犯すことになるのではないかと思います。分散処理はどんな機能だけを分散すべきか，どのように分散すべきか，機能，階層のオペレーションをどのようにコントロールするのか，データ・ベースの統合性をいかに確保していくのかということが大切なのではなかろうかと思います。経営の管理方式やその対象業務の特性を無視して分散すべきではないと思います。

小沢　以上，お話をお聞きしたのですが，高山さんが主張しておられるのを一言にして言えば，エンド・ユーザーの要請にこたえるために分散処理をしたいのだ，とにかくエンド・ユーザーなくしてわれわれの存在はないのではないかということを言っておられたと思います。

次に，塚本さんは，先ほど高山さんが後半に述べられたように，分散と集中の調和を図るべきだという意見でなかったかと思います。

柳井さんの御意見としては，遠隔情報処理サービスにおいても，端末にインテリジェントを設けながら，やはりエンド・ユーザーの要請にもこたえ，なおかつ，タイムシェアリングの仕事をやっていくような傾向があらわれつつあるというようなことが話されたと思います。

また，山本さんは，分散処理におけるファイルあるいはデータ・ベースの技術的諸問題についていろいろ提起がなされたと思います。

最後にウェバーさんは，マネジメントの要請によって分散処理が行われた。あるいはビジネス環境の変化による要請，とくにおのおのの業務特性に応じて分散処理をしていこうということでないかと思います。

それから一つはＯＳの複雑化によって部門のマネジメントが非常にむずかしくなってきた。そういうところからも分散処理が望ましいのだというような御意見でなかったかと思います。

そこで，この主催者の側から，コーディネーターも意見を言っていいということですから，私は私なりの意見をこれから述べさせていただきます。

私は，どちらかというと，分散と集中は調和がとれなければならぬということに異論はありませんが，データ・エントリーの面から分散処理にならざるを得ないのでないかということを申し上げたいと思うのです。

それならば，データ・エントリーの理想像はどんなものか，それを照らし合わせていけばわかってくるのではないか。理想像のまず第一に挙げたいのは，ターンアラウンド・タイムを短縮することが一つの理想像だと思います。

もう一つの理想像は，ソース・データ・エントリーを行うのだ，この二つに照らし合わせていけば，分散処理データ・エントリーをとらざるを得ないのではないか，こう考えております。いわゆるキー・エントリー・システムの時代を考えればいいわけですね。原票の作成，キー・エントリー，バッチ・トータルの算出，プルーフ・リストとバッチ・トータルの照合とかコンピュータへの入力，入力後のエラーリストの作成，そしてまたエラーの修正のための再エントリー，問題が与えられてから答えが出るまでのターンアラウンド・タイムがいかにもかかりすぎた。コンピュータ・ルームにとっては，これは問題にしなくても済むわけです。いわゆるキーパンチャーがミスしようが，第一線の原票の記入がミスしようが，とにかくエラーが出たら，もう

一回入れてくださいと済ましておられるわけです。しかし，企業全般から考えれば，このターンアラウンド・タイムの短縮は非常に重要な問題でないかと思うのです。

そこで，とにかくターンアラウンド・タイムを短縮できるシステムはないかという，この理想像を追っていこうということです。

その次はソース・データ・エントリーです。データの発生源でデータをとらえるのが理想だと思うのです。ということは，仕事の内容を最も知っておるのはデータの発生源である第一線の人々です。ですから，不注意でおかすミスでない限り，ありえない誤りをおかすおそれは，まずないとみるべきでしょう。そのことは正確なデータが捕捉できることにつながります。

ここで，私，強調したいのは，現在最もソース・データ・エントリーの理想にかなうものは，銀行におけるキャッシュ・ディスペンサーだと思います。これはコンピュータとの会話型であり，即座に答がえられるので，ターンアラウンド・タイムは，感覚的にはほとんどかからないのみでなく，預金者が預金を引出すには，自己の利害関係がからむから，まちがいないように，自分のコードを慎重にキー・インする。テラーズ・マシンも銀行におけるソース・データ発生源でデータをとらえるものであるがこれは預金者とシステムの間に行員がはいるので，ソース・データ・エントリーという観点からみれば，キャッシュディスペンサーにはかなわない。

それならば，キー・エントリー・システムの沿革を述べながら，分散処理データ・エントリーに到達する道をたどってみたい。

まず，1968 年にユニバックがバッファードキーパンチを発表いたしました。この後IBMも129 などを発表するのですが，いずれにしても，このバッファード・キーパンチは，キーエントリー・システムのキーパンチの形では最高のものに持っていったと思うのです。それは記憶機構をもって，いわゆる逐次穿孔でなしに一斉穿孔をやったための自覚ミスの修正とかスキップ速度，複写のスピード・アップとか，いろいろ機能が充実して一応キーパンチの最高に来たと思うのです。これを受け継いできてカードが磁気テープに変わったのは，キー・ツー・テープだと思うのです。ですからキー・ツー・テープはバッファード・キーパンチの特性はみんな持っています。

次にはテープの代わりにディスケットにしたのがキー・ツー・ディスケット，これはIBMが開発したわけですが，このような順序をたどってきて，この次にインテリジェント・キーエントリー・システムが生まれてくるのです。

まず1970年にはコンピュータ・ターミナル社がインテリジェント・ターミナルを一応発表，さらに71年にはモホークがキー・ツー・ディスクを発表しています。74年にはIBMがインテリジェント・キー・ツー・ディスケットを発表しています。だんだんインテリジェントをつけるようになった。このインテリジェントがつくということは，クリーン・データをコンピュータに送ることができるわけです。データのチェックあるいは編集を端末で行うことができる。これは何につながるかというと，ターンアラウンド・タイムの短縮につながるということですね。いわゆるエラー・リストに出るものが少くなる。それだけターンアラウンド・タイムの短縮につながる，こういうふうに，ソース・データ・エントリーからターンアラウンド短縮と，発展してきました。スタンドアローンのターミナルにインテリジェントが個々ついたらもったいないという考えから，一カ所に集めてターミナル・コントロール・ユニットにし，これに端末をぶら下げたのがキー・ツー・ディスクです。したがって，このインテリジェント・ターミナルは結局ターンアラウンド・タイム

の短縮につながったのです。しかし，このキー・ツー・ディスクは，もう一つの理想であるソース・データ・エントリー・システムにはほど遠いわけです。つまり，集中してデータをとらえるということです。現場の第一線からとらえるものではない。そこでソース・データ・エントリー・システムの理想からはずれる。そこで，ソース・データ・エントリーの理想にかなえるものはないか。1954年にヒラーが世界で初めてのデータ収集システムを開発，その後ＩＢＭが357を発表，今年になってからもＩＢＭは，5230を発表しております。いわゆるデータ収集装置です。これはソース・データ・エントリーの理想にかなったものです。

次に，1965年にＣＲＴターミナルがコマーシャルベースに乗って以来，多くのターミナルメーカ現われ，オンライン・ターミナル・エントリー・システムを多彩なものにした。

そこでまずデータ収集ターミナルについて見ますと，これは先ほど高山さんが言われたように，第一線の人にデータ・エントリーのトレーニングを行うことなくしてできる機械です。そういう理想的なものが一応できたけれども，しかし，まだこれはターンアラウンド・タイムの短縮にはつながらない。エラーがあった場合には，もう一回入力しなければならない。そういうように，非常にいい面があると同時に，欠点があって，理想像に近づきにくい。

そこでこの次に出たのは，オンライン・ターミナル・エントリー・システムである。汎用ターミナルにはＣＲＴディスプレーや，キーボード・プリンターが用いられる。コンピュータと会話型の入力ができる。キーボードでデータを入力すれば，いま入れたデータがＣＲＴディスプレーに表示され，目で確認できる。コンピュータではプログラム・チェックによって，入力データをチェックする。ロジックにひっかかったものであれば，訂正方法をディスプレイに表示する。再びキーボードから訂正データを入力する。このようなことからターン・アラウンド・タイムが大幅に短縮される。しかも第一線にＣＲＴターミナルを置くことができるので，ソース・データ・エントリーの理想にもかなうというわけです。そうすると，やっとここで理想に近づいた。しかしここに一つの難点が私はあると思うのです。既存の適用業務を今度オンライン・ターミナル・エントリーシステムに切りかえようというときに問題が起こるわけです。いままでのはバッチでやっておったわけです。バッチを効率的に動かすためには，ファイルはシークェンシャルファイルにしている。これをオンラインに切りかえるとすれば，ランダム・ファイルに書きかえなければならない。いわゆるファイルを書きかえるということは，システムを全く新しく組成することと同じであるわけです。おいそれと簡単にはいかない。われわれのところでいいますと，基幹的なシステムが40万ステップくらいの大きさですが，これをすぐにデータ・エントリーをオンライン・ターミナル・エントリー・システムに切りかえようなんてとてもできないわけです。

それならばどうか。次に何があらわれたかというと，分散処理データ・エントリー・システムなるものがあらわれてきたことによってこの救いができる。いわゆる本体のシステムとプログラムをいじらなくとも，ローカル処理はローカル処理をさせて，ホストに必要なものだけをホストに送る。ファイルはローカルでも持つし，ローカルで持つということは，業務特性に応じた処理ができるということです。コーポレート・データも要るとすれば，あとバッチで回線料金の一番安い８時以降にホスト・コンピュータにデータを送ることができます。

そしてデータ・エントリーの面から考えると，分散処理に向かわざるを得ないのではなかろうかというのが私の意見でございます。

それから次に，いままで私たちがお話したことに対して，ウェバーさんから御意見を承りたいと思うのです。

ウェバー　幾つかお話してみたい点がございます。

その一つは，非集中型と分散型の違いということでございますが，非集中式というのはマネジメントの用語でございます。分散型というのは，プロセスから見た用語だと思います。非集中型という場合には，これは分散型プラスということだろうと思います。完全に非集中化されたネットワーク・ミニコンなりマイクロ・コンピュータなどを使った場合には，これはフリー・ディストリビューテッドと定義的にはなるだろうと思います。

ただ，ファイルとプロセシングはミニ・コンピュータにローカルの場所で受け持たせることになるだろうと思います。ファイルのコンソリデーションが必要であるということを指摘されたパネリストがございました。私どもシティバンクではそれをやってはおりません。

その考え方としては，ファイルを集中しないという考え方でございます。先ほど銀行のマーケット・ラインをどういうふうに分けているかということをお話ししましたけれども既存の集中ファイルを6つから8つのローカル・ファイルに分けてしまったわけであります。国際顧客，消費者顧客，ニューヨークの顧客，それから多国籍顧客というふうに，6つ～8つに分けてしまったわけであります。6つ～8つのこのファイルは，ミニ・コンピュータのところに置かれることになります。ですから集中ファイル，中央ファイルというものは全く持たないことになるわけであります。この中央ファイルを小さいものに分けようというのが私どもの考え方で，現場のユーザーのところに持っていこうという考え方であります。95～99%までユーザーが直接アクセスできるように持っていこうということであります。プロセシングを非集中してミニ・コンピュータにファイルを持っていけばいいわけでありまして，集中ファイルを持つ必要はないわけであります。

集中でやっている演算でありますが，これは社内の会計だけであります。これは階層システムのトップのところでやっているわけでありますが，底辺のところではもうミニコンになっております。それぞれの銀行グループにおいて，それぞれのレベルで，それぞれのミニ・コンピュータを使ってファイルを処理しているわけであります。6～7のミニ・コンピュータがありまして，それぞれグループに一つずつあるわけでありますが，テープを中央のミニ・コンピュータに送ると，これはサマリー・レベルの情報処理を行っておりまして，会社の会計情報に近い処理をやっており，そこが違うわけであります。私どもの考え方としては，集中させるものは何も持たないということなのであります。

それから分散型処理への方向というものは技術の進歩の方向と同じ方向にあるわけであります。確かに，会社の中にはアプリケーションによりましてはコンピュータ・リソースを集中させているところがあります。たとえばＩＢＭのメインフレームなどを使っているという会社もないわけではないのでありますしかし，そういったところでも，向こう数年の間にはやはり方向転換せざるを得ないのではないかと思います。集中型のコンピュータというものは，私どもの体験ではかなりの高レベルの理解がなければ運用することはできない。そういった理解を持っている会社というものは，非常に少なかったということですから，余り成功裡に運用できなかったわけであります。そういったシステムの開発期間というものは，2～3倍はかかったということであります。特にミニ・コンピュータを非集中式で開発する期間の2～3倍は，大きなものを使うとかかるわけであります。

それからこの環境装置もかなり必要であるということが，この中央コンピュータの一つの特徴であります。そのメインテナンスも大切でありますし，ワイアリングのために床を上げなければいけないとか，ちりを払っておかなければならないとか，かなりメインテナンスが大変なわけであります。ウォールストリートの私どもの事務室ではミニ・コンピュータを床の上にじかに置いてあります。土足で歩く人たちのそばに置いてあるわけです。必要であるならば，空調機をディスクのすぐそばに置いても別に問題がないということでございまして，床をわざわざ上げる必要もない。そういった意味で，環境整備が必要ないということはかなりのコスト・ダウンにつながるわけであります。この環境整備の問題もミニコンとセントラルではかなり違うということであります。

もう一つ，私，指摘したい点でありますが，通常かなりのコミュニケーション・ギャップがあるわけであります。システム・アプリケーション・プログラマーとビジネス・マネジメントとの間に，普通の場合，コミュニケーション・ギャップというものはかなり大きいわけであります。

システムをつくる場合に，銀行の場合ですが，一つの仕様にのっとってシステムがつくられるわけであります。システム・アナリストの設計に従ってそれがつくられるわけでありますが，ユーザーの立場からではなかなか使いにくいということがありまして，ユーザーは古い在来の方法でコンピュータを50％使ってあとはマニュアルでやるという方式に帰ってしまうということもあったわけであります。しかし，この種の問題も，ミニ・コンピュータを採用することによって，それから一線のビジネス・マネジャーが直接アクセスすることによって解決されているわけであります。

それから皆様方の方で何か私の話に対して御質問があれば喜んで答えさせていただきたいと思いますが……。

小沢　皆様方からこの壇上におられる諸先生方に対して御質問をお受けしたいと思います。なかなか質問が出にくいようでしたら，ウェバーさんの方から皆様方へ何か質問していただいたらいかがでしょう。

ウェバー　パネリストの方のコメントを伺っておりまして，ここにいらっしゃる方が全体に将来分散型にいくということでは合意を見ていると思います。しかし，ちょっと戸惑っているわけですけれども，具体的なケースとして日本の企業が分散処理をどのようにしているかということは私は知らないので，はっきりと理解できないところがありまして，その意味では具体的な質問ができない状態です。

ただ，私の興味のある点ですが，いろいろな違ったタイプのコンピュータを日本で使っている場合，どういうタイプのものを使っていらっしゃるか，ミニ・コンピュータを使う場合の一つの問題は，ミニ・コンピュータの会社，企業が非常に急速な発展を遂げておりまして，しかもいろいろな問題を含んでおります。つまり，それに対するマニュファクチャリング・サポートがないという点，需要が大き過ぎて供給が間に合わないということです。これが日本での問題で，アメリカでの問題で，特にＩＢＭなどの問題でもありますけれども，ただ，ＩＢＭもまだはっきりとミニ・コンピュータの方に努力を集中しているわけではありませんが，ミニ・コンピュータ，マイクロ・コンピュータの部分では需要の大き過ぎるということが問題になっております。

そこで日本ではどのようなエクイプメントを持っておるか，どのようなサポートがあるかということを知りたいと思います。

こうした問題が私の関心事でございます。

小沢　この点について，山本さん，いかがでしょう。

山本　先日ある海外の資料を見ましたところ，分散システムがアメリカでは非常に発達してきたため，ミニコンあるいは周辺プロセッサーを販売している会社のメインテナンス・コストがどうも大分増大してきたという話が出ていたわけです。分散システムは当然システムが地理的に分散しておりますので，メインテナンスの人があちこち出張旅費をかけて歩かなければならないということが起きたためだという話が出ているわけですが，私の感じでは，日本での現状は集中化指向の企業内の大きなシステムが一通り整備し終わった時期で，早いところはここ1～2年，あるいは2～3年後に現在のシステムがオーバーフローし始める。ここで本格的な分散処理の時代に日本でもなるのではないかという気がしているわけです。

いまのウェバーさんの質問には一寸はづれることになるかと思いますが，私は国産コンピュータしか使った経験がありませんので，その範囲内での経験から申しますと，正直申しまして，大型機に比べるとミニコンのメインテナンス，サポートというのは非常に劣っていると言わざるを得ないという気がいたします。今後本格的なミニコン時代になった場合には，サポート体制を少し基本的に考え直していただかないと，どうもうまくないのではないかという気が，私個人としてはいたします。

小沢　ありがとうございます。

ウェバーさんの質問に対し，会場からひとつだれかお答え願えないでしょうか。――そうすると，山本さんのお答えでよろしゅうございますか。

柳井　山本さんのお話はきょう現在のお話ですけれども，私，将来の日本のこういうミニコン業界あるいは製造業というものが，先ほどインテリジェント・ターミナルで日本製の自動車が世界で非常に技術レベルも高いと同じように，やがていま日本でメインフレームをつくっている会社が，全力をあげ出すころには，やはり相当リードするのではないかという気を持っております。輸出も，おそらくこの分野が世界の中心的なものを日本のミニコンが押えるような時期が案外早く来るのではないかというような気がします。

いま日本のメインフレームを製造している大きなメーカーが，ミニコンもやっているケースが多いのだと思うのですが，アメリカではそういうケースは非常に少ないのですね。端末でもそうですが，そう大きなメーカーが全力を注入するというケースじゃないのですが，日本では何でもやりますから，そういう点では非常に優秀なものが出てくるというぐあいに信じているのです。大きなメインフレームをやっている会社が全力をあげ出しますと，メインテナンスその他も相当レベルよくなるという感じを私自身はいま持っております。

山本　ちょっとつけ加えたいと思いますけれども，おそらく将来のメインテナンスのあり方というのは，ミニコン・ベースではその場でものを直すということではなくて，さっと取りかえてしまう。直すのはゆっくり会社や工場へ持って帰って直すという形が普通の形になるのではないかと思います。

小沢　ありがとうございます。

そういたしましたら，話を元に戻しまして，皆様方から諸先生に御質問をお受けしたいと思います。――どうぞ。

質問　小沢さんに……。

私自身は将来分散型システムに移行すると思っているのですが，いま仮にある集中システム，処理能力が無限でメモリーも無限，それからＩＯも非常にたくさんあるというシステムと，分散システムと比べた場合，信頼性についてもコストについてもほとんどの場合集中型システムの方が有利じゃないかということになると思うのです。

なぜ分散型システムが将来発展するかとい

うことを考えてみますと，私自身には考えがあるのですが，小沢先生は先ほどターンアラウンド・タイムが非常に早くなることと，ソース・データ・エントリー・システムということで現場の人間が直接入力できて誤りの少ないデータを入れることができるということをおっしゃったのですが，先ほどのそういう無限大のシステムを考えてみた場合，結局一点を除いて集中型システムの方が有利なんです。それはどういうことかと申しますと，先ほどいろいろな方がおっしゃったように，ファイルが各地に分散していることだと思うのです。それゆえに集中型システムでは処理できないことを分散システムでやるということになる。そういう業務からの要求が結局は将来の分散型システムへの移行を促しているのじゃないか，私自身はこう考えるのですが……。

小沢　私も同感です。分散処理，データ・エントリー・システムそのものは，ローカル処理ができる，それからファイルをローカルに持つということが非常に特色です。ローカルのニーズにこたえるためには，やはり分散処理をとらざるを得ないだろう，私自身はこう思っています。

たとえば在庫管理システムでも，現場ですぐ在庫が幾ら出るというのがわかれば，すぐ対処できますよね。そのかわり，在庫が動いたという会計情報に関しては，バッチでホストへあとで送ればいわゆる会計情報の集中化はできますし，ですから現場のニーズにこたえる。これは高山さんも盛んに強調していましたし，私自身もそう考えます。

いままで現場のニーズをある程度集中することによって，標準化することで処理した。これは一番効率的だ。しかし第一線のニーズ必ずしも受け入れられなかった。これを第一線のニーズに答えるためにはやはり分散処理がいいのではないか，私はこう考えております。

質問　先ほど山本先生がおっしゃったのですが，公社の通信コストが高いために，ある程度インテリジェント・ターミナルの発展が促されるということで，将来広域的，グローバルになりますと，ＡＲＰＡみたいに処理形態が各地で違う，コードも違うということになりますと，かなり二次的な問題が出てくると思うのですね。変換の問題とか，ファイルの転送とか。そういう面で，それほどばら色ではないような気がしますけれども，これは質問じゃないのですが，そんな気がしているのですけれども，いかがでしょうか，山本さん。

小沢　まず山本さんからどうぞ。

山本　いわゆるリソース・シェアリング・コンピュータ・ネットワークと，いまここで議論されている分散システムとは，現時点ではやはり違いがあるのではないかという気がいたします。技術的には共通の問題もありまた将来の方向としては，かなり似てくると考えてもいいかと思いますが。ＡＲＰＡネットワークで代表されるリソース・シェアリング・システムは，基本的にはすでにあるハードウェア，ソフトウェア，データ・ベースというリソースをほぼそのままできるだけ多くの人がシェアリングして使おうということが，その基本的発想になるわけですし，もう一つは，汎用性ということが大きなポイントになっております。いろいろな使い方をされるだろうけれども，それが全部応じられるように汎用的な機能を持っていなければいけないということです。そういう意味では，大きな機能，複雑な機能，汎用化された機能，既存のものを利用する，それをなるべく生かしていこうという機能，そういうことからの発想できたものだと思います。

一方企業内あるいはここで議論されている分散システムというものは，もう少し目的がしぼられているし，はっきりしているということで，ある意味ではもう少しやりやすいし，より目的に合ったコスト効率の高い方法をと

ることが可能ではないかということで，私自身は区別して考えております。

小沢　ウェバーさん，どうぞ。

ウェバー　ひとつ私も同意したい点でありますが，将来はそれほど明るいものでないかもしれないということであります。コミュニケーションに関してでありますが，このARPAの開発もまだ実験的な段階でありまして，今日ではまだ実用的な応用を見ることができないわけであります。銀行での問題は，ミニ・コンピュータのリンクをどう行うかということであります。ゼネラル・オートメーション，それからインプット・デックなどのゼネラルなものもございますが，かなり問題もないわけではないわけで，機能的な立場からいえば，それぞれをいろいろな機能をさせて集中処理からはずしていくという形であります。コミュニケーション・ネットワークを使って実用的なネットワークができた段階でそれをつけていくという二段構えを考えております。

小沢　柳井さん，どうぞ。

柳井　ウェバーさんに御質問があるのですが，ミニコンでのARPA型のネットワークで，現在ウェバーさんのやられているシステムあるいはこれから実行されるシステムの通信コストは，全体の経費のうちどのくらいの比重を占めているものか，知りたいのですが……。

ウェバー　いますぐお答えできないのです。というのはコミュニケーションの機器の方ができていないわけでありますから，そのコストについてお答えするところまで来ていないわけであります。

現在のところでは，ミニ・コンピュータが物理的に大体同じようなところに置かれているわけでありますから，ミニ・コンピュータ・ボックスの間にハンド・オフするハードワイア・インターフェースを設けるという場合もあるわけであります。ハードワイアのネットワークを使うということであるならば，コストもかなり高いものにつくだろうと思いますが，そこまで来ていないので，直接的お答えにならないと思います。

小沢　山本さん，どうぞ。

山本　ウェバーさんに御質問したいのでございますけれども，分散システムというものは中央センターのCPUとかメモリーあるいは要員などのコストを減らすという要素があるということですが，逆に今度は分散して，ミニ・コンピュータといえどもたくさんのコンピュータが存在することになるわけですね。そうしますと，要員のコストおよび教育の問題がでてくると思います。たとえば日本の場合など分散システムを実施したとすると，それぞれの分散したサイドでは要員がせいぜい2～3名で済む程度の分散でないと，コスト・パフォーマンスが悪いのではないかというような意見も聞かれるわけであります。そのあたり，分散システムに関する要員の問題が，どの程度までがスケール・メリットであるか，そのようなことで何か御意見がございましたらお伺いしたいと思います。

ウェバー　私ども，銀行業界で特に数年前に発見したことでありますが，かなりペーパー集中的であるということを感じたわけであります。

たとえば私どもの銀行のニューヨークでは一晩300万ほどのペーパーを処理します。小切手に一枚一枚マイクロ・コードをつけなければデータをとらえることはできないということだったわけですが，このような銀行の労働の内容としては，データのインプットとバリデーションが非常に多かったわけであります。

それからバック・エンドの会計経理処理といったことが非常に多かったわけであります。インテリジェント・ターミナル，それからコントローラー，ローカルのミニ・プロセッサーを使うことによって，データ・エントリーのエフィシェンシーをかなり上げることがで

きたわけです。キーパンチしないでいいということで，テープに直接やる。オンラインでバリデーションをできるということがかなりメリットだったわけであります。コンピュータに入れられるまでに，データも正しいかどうか検証することができますし，フロント・エンドで正しいデータを入れるときには会計経理の方の作業もかなり減らすことはできたということがわかったのであります。

ですからこのシステムの前面と後方のところでかなりコストを下げることができた，一番労働の集中していたところで下げることができたわけであります。労働量とハードウェアのところでのコストをどこで分岐するかという点でありますけれども，環境問題ということもあるわけで，労働力をたくさん抱えるということでは，たとえば組合の問題，ストの問題というようなこともありますから，1対1でトレード・オフをすることは必ずしもできないわけであります。3人の人にかわってミニ・コンピュータ1台を持ってくればいいという簡単な答えは出てこないわけであります。フリンジ・ベネフィット，賃金とか，ストがあったら困るとかないとかいうことも考えていかなければなりませんから，それほど簡単にトレード・オフできないわけでありますが，できるところ，ハードウェアを1対1で今日相殺できなくとも，将来長い目で見ればかなりコストを回収できるという考え方でやるわけです。

小沢　次に，何か御質問がございましたらどうぞ。

質問　ちょっと分散型かあるいは集中型かという議論で，私自身疑問があるのです。現在非常に小さな規模ですが，インテリジェント・ターミナルを全国に20台ほど置きまして，公衆電話で夜バッチでデータを送ってくるわけです。これはもともと分散化を目指してやっているのですが，分散化しようと思いましてデータ・バリデーションをやりますのに，現在のインテリジェント・ターミナル，かなりの規模を持っているのですが，データのチェックすらできない。といいますのは，もしやろうと思いましたら，全部部品票を持っていないといけない。何十万点になってくるわけです。ですから結局はインテリジェントを持っていても，デイリーに集めたやつを夜バッチに持ってきて，本社で高速処理をしてエラー・データを夜の間に端末に返してやる，そういう処理をとらざるを得ないのが現状なんです。

システム的には分散型というのは非常にむずかしいと思うのですが，あと幸い端末の方がインテリジェントが余っていますので，各ロケーションにおいて部品のシステムとか全然違うシステムを処理しているわけですね。九州で部品処理しても，北海道では全然別のプログラムを開発して，それを処理しているというような形で，分散というのは非常に望ましいと思うのですが，非常にやりにくいということを感じます。

もう一つ，非常に将来的に疑問になってくるのですが，たとえば部品票にしても商品マスターにしても，どんどん追加するロジックがあるわけです。ところがドロップするロジックが非常に欠けている。ですから分散化していきましても，ミニ・コンピュータでできると言いながら，事実上大型コンピュータを支店とか販売拠点に置かざるを得ないのではないのではなかろうか。そうしますと，オペレーターが当然要る。ですからせいぜい女性でオペレーションできるくらいのミニ・コンピュータではとてもできないのではなかろうか，こういう心配を持っておるわけです。

この辺について先生方のお考えを聞きたいと思います。

小沢　ウェバーさん，どうぞ。

ウェバー　ちょっとビジネスの問題をはっきり理解しないではコメントすることがむずかしいわけですけれども，いまの第一の点に

ついてですが，インテリジェント・ターミナルのかわりに多分バリデーションの問題をいまおっしゃっていたと思いますけれども，何十万点というパーツの問題，バリデーションの問題ですけれども，ミニ・コンピュータがターミナルに近く置かれたときには，そのターミナルにインテリジェンスが必要かどうかDEC 11/70のミニ・コンピュータのデイスクストレージですけれども，それで十分にディスクリプターを何万か入れることができると思いますし，また，何万かのバリデーションがそこでできるかと思います。ただし，分散処理のときにはファンクションをブレークダウンしなければならないわけで，それをマルチプロセシング・マシンでするような形になるわけです。

たとえばフロント・エンドでターミナルが一つのファンクションをするだけの，たとえばバリデーションのファンクションをさせるだけのものにして，そのインフォーメーションをバリデーションしてからほかのコンピュータに送って次のファンクションをさせる，つまり，ファンクションの分散化をするわけです。必ずしもそれは中央の大型のコンピュータではなく，ミニ・コンピュータでできると思います。そのミニ・コンピュータができるということは，ほかのコンピュータができないことをネットワークの中で別なミニ・コンピュータがやっていくことができるということです。

小沢　よろしゅうございますか。答えになっておりますか。

質問　いや，かなり違うのです。（笑）

質問　SRIからの者でございますが，ウェバーさんに御質問させていただきたいと思います。

いまのウェバーさんへの質問ですけれども，リミテーションの違い，つまりOSのリミテーションですけれども，日本の銀行とアメリカの銀行との比較をしまして，日本の大きな銀行は大型のCPUを使っており，3000とか4000，5000のターミナルを日本じゅうに配置しているような，それが典型的な日本の銀行のシステムになっております。

ところがアメリカの銀行の場合には，ネットワークで分散型でミニ・コンピュータを使っていらっしゃるということですけれども，質問の趣旨ですが，ソフトウェア・バックアップの制限的要素，リミテーションの問題ですけれども，その問題をシティバンクの方で解決なさったかどうかということを聞いていたのだと思います。

ウェバー　ちょっと誤解があったのかもしれませんけれども，ミニ・コンピュータを現在使うときの問題ですが，オペレーティング・システム，ソフトウェアが完全にメインフレームと同じようには発展してないというところでございます。というのは，歴史が浅くて数年しかミニ・コンピュータが市場になかったということ，ところがメインフレームの場合には40年，50年の歴史を持っていますので，そこが違うために問題があると思います。

一つの具体的な問題ですけれども，ミニ・コンがちがうとオペレーティング・システムも違うということで，一つでアプリケーションをつくると，ほかのミニコンにそのアプリケーションを使うことができないということで，われわれはそれをわれわれ自身のRアンドDで解決しようとしております。つまり，アディアムとかハネウエルとかを，またそこからパッケージで買うこともできるでしょうが，われわれ自身のRアンドDをしております。

これがむずかしいのは，まだまだ開発の段階にあるから困難が多いということです。しかしシティバンクは，ニューヨークでは現在唯一のミニ・コンピュータ・ネットワークを持っている銀行であります。そのほかの銀行は，まだ様子を見ているというか，われわれがどう扱うか，情勢を見ているのではないか

と思います。われわれは，この技術の面では，特に銀行の面では，アメリカでパイオニア的な役割を果たしていると言えると思います。

それから幾つかの場合にアプリケーションをミニ・コンピュータ用にわれわれが開発したような例もあります。会社用のものですけれども，これはマイクロ・イメージ・テクノロジーと言われるものですが，そのときのわれわれ直面した問題は，サービスをカスタマーに提供するという問題がありました。つまり，必要な記録を蓄積して調査をするのに十分な技術がなかった。というのは，いろいろなプロセシング・サイトにいろいろな市にわたってこのレコードがあったために，それを集めることがむずかしいために，MDS，マイクロ・データ・システムというものをつくりまして，一日に13,000のドキュメントを集めてマイクロフィルム化しまして，そのマイクロフィルムの大きさですが，このくらいの大きさで350万くらいのレコードが1つの中に入っているような，そしてマイクロフィルムのテクノロジーを利用してミニ・コンピュータに使ったわけですけれども，この350万のレコードに4秒くらいの間でアクセスができるというようなテクノロジーです。

これは，しかもロケーションは50くらいのターミナルにわたってできるような場合で，このマイクロフィルム・テクノロジーを開発しましたけれども，それに対しての簡単なアプリケーションを持たない会社のオペレーション・システム，またアプリケーション・システムをつくるときのわれわれの改善のためのヘルプを与えて指導いたしまして，そのようにミニ・コンピュータのOSをつくるための開発の援助をしております。しかし，あと数年たたないと，多くのビジネスで新しいこのミニ・コンピュータの技術に到達することはできないのではないかと考えております。

小沢　どうぞ，あちらの方の方。

質問　ウェバーさんに伺いたいのですが，企業を機能的な単位に分けられて，そのおのおのに計算組織を帰属させているということなんですけれども，ひとつ，つい数カ月前の「データメーション」に載っていたことですが，ミニというのは出発点はミニだけれども，いつの間にか非常に大きくフルに最大限度まで拡張されたマキシ並みになる傾向があるということなんですけれども，計算機一般に，計算機自身が自己増殖を始めるという傾向は否定できないと思います。

もう一つ，現在のソフトウエアの技術でいえば，何か仕事をしたいユーザーがしたいと思うことをすぐ計算機に載せることはできなくて，何らか計算機の専門家がそれをトランスレートするという仕事はしなければいけない，つまり，計算機には人が要るということも現実だと思いますが，そういうファンクショナルな分割された組織が自分たちのやりたいことを極限追求すれば，人の方もどんどんふえていくだろうということが予想されるのですが，そういう機械と人とを分けてそれが増殖した結果，やはり集中しておいた方がトータルに見て安かったのではないかということになる心配はないのかということが一つ。

それからもしそういうことを懸念されているとすれば，そのためにどのようなコントロールの手順を考えておられるかということを伺いたい。

ウェバー　幾つかの点をただいま挙げられたと思いますけれども，前にも申しましたとおり，私ども，幾つかの要件があると考えるわけです。特にビジネス的にこれを応用する場合に，ミニ・コンピュータにするか，集中型を採用するか，分散型を採用するかというときに，まずビジネスのトレード・オフ，要件が何であるかということを確認することが必要なわけであります。

いろいろなアプリケーションでも，まだまだビジネス的にうまくいくという傾向が言えるわけであります。ニューヨークの大手の銀

行ですから，証券の書きかえ業務などがあります。これはＩＢＭの360でされていたわけでありますが，この処理の性格から，インプットは日中いままで行われていたわけであります。バッチ・プロセシングの方は夜行って，データを更新して記録は翌朝出されるという形になっておりました。それからエラー・コレクションは次の日に延ばされる。エラー・コレクションが非常に大きくなりますと，２日間なければ証券の振りかえもできないということだったわけであります。

そこで，インター・データ・ミニ・コンピュータ・システムというものをつくったわけであります。オンラインでインプットのバリデーションを行う。その日にインプットされたものは，その午後処理または少くとも夕方には処理してしまおう，そして記録の方も，報告書の方もその晩には出てくるということで１日で処理できるということはかなりメリットがあったわけであります。

しかし，オンラインでインプットのバリデーションすることの危険性，むずかしさということは，６００人中１２５人を削ることができ，125人分の賃金の節約部分でも新しいハードウェアを購入する費用が十分出たわけでありますが，ただ，このミニ・コンピュータにはかなりオペレーターの数が必要であるというところに気がつかなかったわけであります。銀行のほかの業務に中央データ処理をやっておりますから，中央施設に働いているスタッフも当然そこにいるわけでありますから，ミニ・コンピュータを採用する場合に２つのシフトが必要になったわけであります。そこで20人新たに採用しなければならなかったということでございます。

ですから，新しいシステムをつくる場合，そういった落とし穴があるわけでありますが，大きいところから見れば総体的にはコスト整備になったと思います。

ミニ・コンピュータには，確かに長短両方の見方があると思います。分散か集中かという考え方も当然あるわけであります。われわれはすべてを非集中でやろうということに決めまして，ミニ・コンピュータをすべてに使うのだという方針をまず決定いたしまして，そこからやったわけであります。ビジネス的にこれはよくないということが実証されるまでわれわれはこの方針を徹底するのだということを決めたわけであります。

いま御指摘のトレードオフも一つでありますが，オペレーターの数をふやさなければならないということも，その一つの危険といえば危険だろうと思います。

質問　日本システミックスの上原と申しますが，ミスター・ウェバーにお願いしたいと思います。

先ほど集中されているデータはサマリー・レベルの会計データであるというお話がございましたが，トップマネジメントが通常必要とされるデータというのは，その会計のデータ・ベースの方からほとんどの場合は供給されるものなのかどうか。

それから，もしそうでないとしますと，分散されたデータから情報が必要だという場合には，どういうふうにそれをつなげているのか。それはつなげ方として，何かシステムとしてつなげているか，人間がここにその情報があるだろうということで探すのかという点。

もう一点は，集中されたといいますか，会計データに対するアクセスといいますか，処理ですが，バッチ方式のみ可能なのか，常時データに対してアクセス可能になっているか，つまり，トップが何か必要であるといったときに，即そこにアクセスできるようになっているのかという点についてお答えいただきたいと思いますが……。

ウェバー　サマリーの情報は，特にトップのマネジメントが必要と要求したときに出されるものであります。ユーザーのトップ・マネジメントが，たとえばデイリー・ベースで

必要とされるものをセントラル・コンピュータからサマリー・インフォメーションとして出すわけです。

しかし，多くの場合，もしもっと細かいところまで必要な場合には，そしてその細かいところがミニ・コンピュータにサマリー・レベル・データとしてない場合に，その次の階層のレベルに下がっていきまして，そこでデータがあるかどうかということを次のレベルのコンピュータに問い合わせるわけです。

つまり，必要とされているのはリンクでありまして，サマリー・レポートがあれば，それに対して細かいものをＣＲＴスクリーンまたはディスプレイなどで，そのサマリー・レポートを出したらコンピュータからそのようなものをとるということですけれども，まだそのような集中化されたものがセントラル・コンピュータから出てくるということはできておりませんので，リンクというものがミニ・コンピュータ間にまだできておりません。その場合にはテープの手渡し，ハンド・オフということになっております。

たとえばウォールストリートの方にターミナルがありますので，そこの方にいきまして，そのビルの中にミニ・コンピュータがありまして，そこにテープがあるわけですけれども，そのテープを持ってニューヨークの北のアップタウンの方に行きまして，そこでまたサマリーを出すためにそこのコンピュータに入れるということをするわけです。

ですから，銀行の頭取がもっと細かいところを欲しい場合には，ダウンタウンにあるようなミニ・コンピュータに行ってそれを集めてくるということが必要になるわけです。

第二番の質問ですけれども，インフォメーションをどのようなバッチ・プロセスをしているかということですけれども，これはオンラインでバリデーションをしておりまして，夜にバッチ・モードでサマライズしております。

しかし分散ネットワークの場合には，それによってバッチ・プロセスの時間を夜に持ち越して必要なものを日中にオンラインでするということが一つのネットワークの利点だと思います。このようなネットワークが完成しましたら，先ほどのようにテープをダウンタウンからアップタウンに持っていくことも必要なくなりますし，バッチ・プロセスも全然必要なくなるということも考えられております。

質問　二番目の質問は，会計データを処理するマシンの情報の取り出しメインテナンスとか処理ではなくて，取り出しの方が常時取り出し可能か，バッチ処理によらなければいけないかというような点でございますが，もう一度お願いします。

ウェバー　はい，済みませんでした。

いまわかったと思いますけれども，バッチで夜にできた報告書ですけれども，これはオンライン・レポートではないわけです。したがって，現在仕事を続けているところですけれども，レポート・オンラインにするということは，この次のステップになります。現在，オンラインのレポートが必要ではないというふうな理解に立ってバッチでやっております。シニアのマネジメントはサマリーのデータが必要であって，つまり，前日のものをその次の日の午前中に得て，それによってアップデートをするということであります。しかし，もし必要であれば，オンラインでアップデートするようなレポートを次のステップとして考えたいと思っております。

小沢　あと３分ぐらいありますが，その時間でおさまるような御質問ありませんか。御質問がないようでしたら，大体時間がまいりましたので，これで終らせていただきます。

ウェバーさん初めパネリストの皆さん方，どうもいろいろありがとうございました。

また，御清聴いただきまして，皆さんありがとうございました。

セッションⅢ　プログラミング：手工芸から科学へ

要旨

今世紀の一大技術革命のひとつと言われるコンピュータが登場してすでに２０有余年が経過した。この間にハードウェアの急速な進歩に比較して，ソフトウェアの工学としての進展の遅れが問題になっている。ストラクチャード・プログラミングは，この問題解決の鍵として，この分野に一大革新をもたらし，新しい世代のソフトウェア技術として，今や大きな話題を呼んでいる。このストラクチャード・プログラミングが，わが国のソフトウェア技術者に与える影響は極めて大であり，工芸から工学への大きな飛躍が期待される。

プログラミング：手工芸から科学へ

エヅガル・ダイクストラ[1)]

通訳の方々にあまり負担を掛けたくないので，日本語でお話したいと思います。多少ヨーロッパなまりの日本語になるかもしれませんけれども，私の日本語を御理解いただければと思っております。

マネジメント，ユーザー，インテェリジェント・ターミナル，パラレル・プロセシング，システム，ネットワーク，データ・ストラクチャー，システムズ・アナリスト（ＳＡとも申します），エディティング，マン・マシン・インターフェイス，多重のハイアラーキー，データ・ベース，ファイル，レコード，タイム・シェアリング，リソース，コンセプト，フローチャート，ディバッグ，ヒューリスティック，ベンチマーク，コスト・パフォーマンス，チェックポイント，ポイント・オブ・セール，キー・ツー・ディスク，トレード・オフ，バッチ・プロセシング，オンライン，オフライン，ファンクション，コントロール，アーキテクチャー，インタラクティブ，プログラム・メインテナンス。これらの言葉の中で，３つ最悪なのがありますが，それはユーザーとインテリジェント・ターミナルとプログラム・メインテナンスといった言葉であります。

何年も前に，数学をやるスタイルというのは各国によって異なっているということに気がつきました。たとえば１９世紀のドイツの物理学者でありますボルツマンは，非常に長たらしい，醜く乱雑な計算を黒板に書くということで知られておりました。あるとき彼の学生が彼の数学の式は非常に醜いと不平を言いました。するとボルツマンは非常に怒って学生にどなり返しました。「そんなことは私の知ったことじゃない。」そして，「エレガンスというのはくつ屋とか洋服屋が考えればいいことだ。」といったのであります。丁度同じ頃に，英国においてはジョージ・ブールが――彼の名前はブール代数という名称で栄光に輝いていますが――思考の法則ということを研究し，本を書いております。その本の象徴的ロジックという章の中で彼は突如中断をいたしまして，議論の美ということを取り上げ，議論が美しく，愛らしいことがいかに大切であるかについて，まるまる１ページを費やしているのです。この両者の対照はきわめて大きいというふうに言わねばなりません。後になりまして，もっと国際的な接触を私が持つようになりましてから，これらの違いがきわめて深刻であるということに私は気がつきました。それはただ単に数学のやり方のスタイルの違いというだけではなくて，人々のコンピュータ科学やプログラム等々のやり方にも直接に反映していることに気づいたわけであります。英国の数学が原則としてきわめてエレガントなものであるのに対し，ドイツの数学はどちらかというと大時代がかっているという傾向があり，私の印象ではソ連の数

1）（Edsger W·Dijkstra）
アイントホーベン工科大学教授

学にはさらにその傾向が強いように思われます。このように数学の役割りに対する認識にきわめて大きな差異があることから，一方にはボルツマン学派的な人々がおりまして，その仕事は結果さえよければよく，そして，手段は結果によって正当化されるという考え方をするわけですけれども，私はそういうのは余り賛成できかねます。

さて，1960年代の後半になりまして「ストラクチャード・プログラミング」という言葉を私はつくり出しましたが，その際に私は二つの重大な間違いを犯しました。間違いのひとつは商標登録をしなかったということであり，その結果としてＩＢＭがその言葉を全く恥しらずにも盗むことができたわけであります。もう一つは，定義を与えなかったということであります。それに対する私の唯一の言いわけは，このストラクチャード・プラグラミングという言葉が野火のようにこんなに広がるとは私には予想しえなかったということでございます。ストラクチャード・プログラミングに関するものとして知られているレポートにしても，果して世間に配布する意図をもって書いたかどうか自分でもはっきり判りません。これらは自分自身のために書いた文章の形をとっており，私としては私が神経症にかかった時の回復期に正気を取り戻すために書いたと言った方がよいのであります。とかくするうちに明らかに，たった一つの言葉だけが人々の目にとまるところとなり，最も濫用された流行語のひとつとなったわけでありますが，私自身は確かもう４年間ほどこの言葉を使っておりません。

今後出てくる流行語として，マネジメント・オブ・コンプレクシティ（複雑さの管理）とザ・レベル・オブ・アブストラクション（抽象の水準）があげられます。あらかじめ皆さんに警告しておきましょう。

なぜ私がこのストラクチャード・プログラミングという言葉を定義しなかったかと申しますと，その時点におきましてはこの言葉が一体どういう意味を持つようになるか判らなかったからであります。それよりむしろ目標を達成する以前に，われわれが何を目指しているかを示唆するために，その言葉が必要だったのです。その当時いろんな理由から，われわれの多くは，プログラミングという作業は大幅に変化するであろうということは知っておりましたし，その変化の方向についても確信をもっておりましたが，どの程度の速度で変化が起こるかについてはわかりませんでした。しかしながら，その発展の方向はわかっていましたので，それに対する言葉を探し求めて，ストラクチャード・プログラミングという言葉を使えばその目的に即応することができると考えたわけであります。

６０年代の中葉になりまして，ソフトウェアの危機とかソフトウェアの失敗とかいわける事象を観察するにいたったのですが，そうしたことが起こっても別に驚くには当りませんでした。そうした事態が起こることは前もって予測されておりましたし，その原因もごく単純なものだったからであります。その10年の間にコンピュータのスピードは1,000倍ぐらいになり，その結果としてプログラミングの能力がコンピュータの発展についていけないであろうことが明らかだったからです。また，同時にマシンが非常に複雑なものになって来ました。古い純粋にシクェンシャルなマシンは，多重の記憶装置とか非同期に動く周辺装置によって置き換えられ，記憶装置も一レベルであったものが多重レベルに変わっていったのです。そして，こうしたことがマシンの複雑性を更に高めたので，当時われわれが用いていたプログラミングの手法では十分に対応できないことは極めて明らかだったのです。しかしながら，当時のプログラミング慣行では不適当なのだということを世界全体が認識しないことには，何んらかの手を打つなどということは不可能でした。自分たち

のやり方が完全にいいと信じ込んでいる人たちに対してそのやり方を改善しろと言っても無駄だからです。その意味で，1968年10月にドイツのガーミッシュでNATOのソフトウェア工学に対する会議が開かれたとき私は非常に感激をおぼえました。ついに私が長年待ちかねていたことがその会議において起こったからであります。ついにその時点に至りまして組織のかなり上層に属する多くの人々がプログラミングに関する限り危機の状態であるということを告白したわけであります。つまり公開の場でソフトウェアの失敗を認めたということはきわめて画期的なことであり，少なくともついに条件が変わり，この事態に対処できるようになったということを示唆するものだったのです。大変おもしろかったのでよく覚えております。全体として会議の出席者たちは互いに他の出席者たちが率直に認めたということに印象を受け，また事態の深刻さも十分認識されたわけであります。会議の最終日に，非常に大きなトラブルを抱えておりますある一人の男がおりました。この人はほかの多くの者と同じように，国に帰って会議の報告を提出しなければならないという立場にあったわけであります。彼はIBMの代表でありまして，会社の言葉だけを使っては，failureという言葉を説明できないということで，どうやって報告書を書いたものだろうかと悩んでおりました。ともかく，そのときにわれわれは何か画期的なことが起こったということを認識したわけであります。

また，もう一つプログラム活動の再検討をわれわれに迫るものとして，新しいプログラム言語を設計しようと試みていた多数の国際委員会の運命がありました。たとえばIFIPのワーキング・グループ2・1が後にALGOL 68となったものを設計しようとしていましたし，SHAREの委員会がPL／Iをつくろうとしておりました。こうした言語設計にたずさわっていた多くの人々が自分達のやっている作業に非常な不満を持つにいたりました。もともと新しいプログラム言語の設計は難しくて委員会組織ではとても手に負えないのではないかと考えられてはいましたが，それだけでは説明になりません。本当の理由は，明らかにわれわれがプログラミングという仕事の性格を十分に理解していなかったというところにあったのです。それぞれの道具がもっている最も重要な側面は，その利用について訓練を受けた人の習慣に対し，それが大きな影響を与えるというところにあります。これはあらゆる道具について言えることですが特にプログラム言語についてよく当てはまることであります。何故なら，プログラム言語の利用は，利用すること自体が直接利用者の思考過程に影響を与えるからであります。さて，PL／IとALGOL 68の設計がなぜ共に行き詰まってしまったかと申しますと，過去10年間，つまりFORTRANとかALGOL 60などが設計されてから後，プログラム言語設計過程の中でのプログラマーの性格付けが余り変わっていなかったからであります。われわれがその当時頭に画いていたプログラマーは，FORTRANとかALGOL 60の設計をやった時と依然として同じような未熟練プログラマーだったのです。すなわち，物理学者とかエンジニアなどで，ときたまコンピュータを使う人達であり，午後いっぱいかかって3ページのプログラムを書くといったユーザーを対象にしていたのです。われわれがプログラマーに対して画いていたイメージが進化していなかったので，きわめてすぐれたプログラム言語を設計することはできなかったわけであります。その時点でわれわれは認識を新たにしたのです。すなわち，われわれは物理学者とかエンジニアがどのようにALGOL 60とかFORTRANを使って片手間にプログラムを書くかということについてはおぼろげながらも理解してはいたが，真にプロのプログラマーが30ページとか

100 ページのプログラムをどうやって作成するかについてはほんの少しの知識しか持ち合わせていないことに気付いたのです。ですから，よりよいプログラム言語を設計するためには，われわれはまず何よりも先に真のプロのプログラマーの理想像を創造する必要があることを発見したのです。プロのプログラマーになるためにはどのような知的な訓練が必要か？　そのようなプロのプログラマーは正当な要求として，どのような特性をプログラム言語，すなわち，彼が用いねばならない正式の道具に要求できるものであろうか？

こうした問題の全てがまだ未解決のままに残っていたのであります。そこで，こういった問題に対する答えを探し求めるに当って，新しい科学の分野が台頭してまいりました。そしてそれにも名前がつけられさえしました。すなわちプログラミング方法論と呼ばれるものであります。

プログラミング方法論というのは最初は極めてあいまいな活動でした。しかし当時ひとつだけはっきりしていたことは，５０年代から６０年代を通じてコンピュータが依然として未だ非常に高価であったことから，プログラムの実行効率に非常に大きな注意が払われていたということであります。そこでわれわれの多くはプログラムの実行効率をあまりにも強調しすぎることが，麻痺的作用を及ぼしていると感じ，その結果，より定量的に評価し難い面，例えばプログラムの優雅さとか美しさとかいったものにかえって関心を抱くにいたったのです。

私はプログラミング方法論の初期を，あいまいさと散漫さといった言葉で呼んでいます。

1964年にピーター・ナウアーが，1967年にボブ・フロイドが，そして1968年にトニー・ホアが書いた論文が既に出版されており，こうした３つの論文が，プログラムの正しさをどのように証明するかを取りあげていたにも拘わらず，最初の頃にはわれわれは残酷で非妥協的な表現である「プログラムの正しさを証明する」といった言葉は用いませんでした。未だ機が熟していなかったからです。少くともその後１年位は彼等の論文は棚の上に放置されたままであり，学界の注目は浴びたものの実務家にはあまり影響を与えることはありませんでした。当時の状態というのは，同じ問題に関して小さなプログラムを幾つか作って，その３つか４つを目の前に置いて数時間も眺めたあげく，三種類のプログラムの中「自分の一番好きなものはどれか」，そして次に「それは何故か」といったことを考えるといった状況にあったので，こうした活動には余り多くの共感を集めえなかったのであります。「現実の世界のプログラマー」すなわち，実務家の人々はその目的を理解せず，数学者とか論理家といった理論家の人々も，深さがないといって，その意義を認めませんでした。また当時は，プログラミング方法論に関心を持っていた人々が，――それぞれ理由があったのですが――プログラミングのスタイルにあまりにも心を奪われすぎていると批判されていました。

しかしながら，私はあらゆる種類のプログラムについての小さな実験を繰り返しながら暗中模索にあった，こうした幼年期が絶対に必要であったことを指摘しておきたいと思います。何故なら当時プログラミングは，未だ直感的な工芸の域を出ていなかったからでありまして，もちろんもっとしっかりしたものにして行かなければならないし，またそれがやがては形式にかなった学問の一分野にまで進化して行く可能性があるとは薄々は感じられておりました。もちろん形式を導入する手順としては，まず形式が必要であるとの確信がなければなりませんし，又形式化によって何を達成するのか，その目的は何かを明確に把握しておかねばなりません。ところでこれは大変よい効果をもたらしました。何故ならこうした様々なプログラムの実験の結果，わ

れわれは既に当時用いられていたプログラム言語から余分なものを殆んど取り去ってしまっていたからです。われわれは優雅さを追求するためにやったのですが，その結果としては，フォーマルなテクニックが大規模に適用され始めた頃には，正式な理論にまとめ上げる価値のあるものだけが残っていた程度にまで，剪定が行われたと云うわけです。

さて，プログラミング方法論の幼年期における美的関心について一言申上げますと，当時われわれは観賞するために美しいプログラムを書くと云うことを始めており，実際にわれわれは，あたかも数学者達が数世紀にわたって証明の美を論じてきたのと同じように，プログラムの美について論じ合ったものでした。その結果，プログラマーの世界と数学者の世界との間に情緒的なつながりが出来たと云うわけでありまして，こう云ったことが後の発展の基本となったのであります。もっともそれ以前の，自動計算が始まった初期の頃，つまり４０年代にはそうしたつながりがあるにはあったのですが，しかし６０年代の中頃になるとこの二つの世界は分離して行ったのです。

以上，私はプログラミング方法論の幼年期をめぐる情緒的そして知的な環境を概観して来ました。と云うのも，そうすることがその後に起ったことの意義を，あまり技術的な詳細にふれることなく，皆様方にご理解いただく上で役立つと考えたからであります。

ひとつ覚えておいていただきたいのは，10年前にこうした動きが始まった頃には，プログラムは専らといってよい程，そのプログラムがコンピュータにより実行された時に，一体何が起るかという観点から考察されていたということです。計算効率に対する関心が計算機の歴史と結びついて，深い影響を与えていたというわけでありまして，このことは人々がプログラムを評価する場合に，そのプログラムのコントロールの下で行われる可能性のあるあらゆる種類の計算を考慮しなければならないということを意味します。その結果として，当時のプログラムに意味論的な定義付けをする努力の全ては，常に今日のいわゆる「操作的定義（operational definition）」であったのです。

６０年代の初頭における問題は――プログラム言語の構文（syntax）を定義するのに用いることができる技術はあったものの意味論の定義付け，つまり，プログラムが実行された後の正味の結果の解明は非公式にしか行われなかったということであります。(このことについては，ピーター・ナウアーがALGOL６０のレポートの中で彼のすばらしい英語で述べています。）その後の形式化の努力というと，６０年代の残りの期間においては，インタプリティブ・オートマトンといった方向へと漂って行ったのであります。

しかしながら，進歩するためには，プログラマーによって書かれたプログラム・テキストと，そのプログラムのコントロールの下で行われる可能性のある一群の計算との間の緊密な関係を切り離すか，少くとも緩和してやることが絶対に必要だったのです。もちろんわれわれは，コンピュータがプログラムを実行したときに何が起るかを少々忘れるためには，それなりの訓練が必要であることを知っていました。何故ならば，われわれがそれをあまり意識しすぎると，その当時のハードウェアの好ましくない特性が，プログラム言語に影響を与え，しいてはわれわれの思考にまで悪影響を及ぼす危険性が高かったからです。

もう一つの理由は――これは後になって，初めて気付いたことですが――とかくプログラマーは彼の主要関心事として，まず第一に「プログラムが正しい」と云うことを求めます。第二に実行のためのコストが許容できるようなものであること，つまり計算時間からいっても，またメモリー利用からいっても効率が高いといったことを求めるものですが，

われわれは後になってこの二つの異なる関心事を区別して，別々にひとつひとつ取り扱うことが必要であると認識するにいたったのであります。すなわち，プログラムの正しさと，プログラムの効率は両方とも極めて重要であり，かつ極めて難しい問題であるが故に，プログラマーがこうした二つの関心事を完全に切り離すことを許容することは非常に大切なのであります。これについては後にもう少しふれてみたいと思います。

同時に二つの進展がみられた結果，プログラムと，行なわれる可能性のある一群の計算とのつながりを緩和することが可能になりました。その一つは，プログラミング方法論の幼年期において，われわれは極めて多くの実験を行ないましたが，その全ては未だにプログラム言語として実現されてはおらなかった試験的言語によった実験であったことです。すなわち，こうした活動自体がプログラムと実際の実行との間のつながりをある程度まで緩めていたわけであります。第二は，ナウアー，フロイド，ホアーなどによって開発されていたフォーマルなテクニックがその独自のやり方でプログラム・テキストの検討を可能にするものであるとの発見が行なわれたことです。もともと，こうした公理的な定義は，プログラムの実行時に起ると想定されるものによってインスパイアされたわけですが，ひとたびそうした正式な描写が行われた以上，今度は正式な定義としてそれ自体を独立に取り扱ってもよいわけでして，プログラム・テキストを数学の対象と見なし，当座のあいだは，それがコンピュータによる実行の対象になることを忘れてしまうのです。このような転換が起ったわけですが，これを別の言葉で表現すると，初期の頃には，われわれはプログラムの目的はコンピュータにインストラクションを与えることにあると思っていたのに，いつの間にか物事がひっくり返って，われわれはコンピュータの目的はプログラムを実行することにあると考えるようになってしまったのです。

「アンダースタンダブル（理解できる）」とか「クリア（明せき）」とか「リーダブル（読める）」といったあいまいな，情緒的な言葉から「正しさの証明」といった非妥協的で残酷な考え方への転換は，緩慢でそして苦痛に満ちたものでしたが，これにはいくつかの理由がありました。理由の一つは，前にも申しましたように多くのプログラマー達はプログラミングを手工業として習ったわけでして，命題算に対する機敏さといったものを持ち合わせていないので，教育，訓練を受け，新しいフォーマリズムを知的資産としなければならないのですが，これには長い時間が掛ります。このような成長の側面を知ることは大変大切です。

もう一つは――これは全く偶然の出来事とは思いますが――スカンジナビアの雑誌を通じて発表されたナウアーの論文は，その後に発表されたR・W・ロイドの論文ほど注目を集めなかったのであります。そこで，プログラムの正しさを証明するというアイディアが，たちまちにして機械的なベリフィケーションの問題と結びつけられて考えられるようになったのはフロイドの責任だと思います。この数年間プログラミングの方法論は，人工頭脳研究活動の底に根強く横たわる，迷信，つまり，困難な仕事はすべて退屈なので，それを機械にまかせた方がよいという迷信によって，成熟を見ないままに葬り去られる危険にさらされてきたのです。

（時間がありませんので少々飛ばしますが）初期の頃の正しさの証明をめぐるわれわれの経験は，気がめいらんばかりのものでした。証明はだらだらと長く，醜くいものでした。数は少なかったのですが，美しいプログラムもあり，その正しさの証明も同じように美しくなりそうに思える例がいくつかあったのは幸運でした。しばらくするうちに，ある人々は，

プログラムの正しさを証明することに失望しましたが，その他の人々は証明の努力を続けました。そして何故初期の正しさの証明をめぐる努力がかんばしくなかったか理由がだんだん判ってきました。

まず第一に，通常の数学の場合と異なって，定理がまだなくて，公理から出発しなければならなかったことがあげられます。しかしその中に，非常に数は少ないが，非常に一般的な定理をいくつかのプログラム構成について発見し，それらの利用について経験を積むようになるにつれて，これらの定理についての知識のお陰で，全てが公理から引き出されていた頃よりも，正しさの証明がはるかに簡潔なものになりました。

正しさの証明が改善された第二の理由は，

これは主としてトニー・ホアーのイニシアチブによるところが大きいのでありますが

われわれがより簡素でよりシステマティックなプログラム言語の設計にはじめから取り掛ったことであります。

第三に，証明が改善された理由に挙げられるのは——これはあとから考えてみると別に驚くにあたらないことですが——プログラムの正しさを証明するためには，二つのことが必要だということの発見です。片や，プログラム言語そのものとその構成物に関する形式的な理論が必要でありもう一方には，計算の対象物に関する形式的理論が必要なのです。例をテキストの中にあげておきましたが，コンパイラーの一部としてparser（構文解析プログラム）を書いた場合に，例えば文字列についての演算といったそのプログラムに必要な関連オペレーションの良い正式の定義を持っているだけでは十分ではない。その他にＢＮＦ－文法の助けをかりて，定義付けられた言語に関するある程度の理論が必要なのです。最初の間は，証明義務のこうした二つの別個の側面が十分明確に区別されていなかったために面倒が起ったのです。

しかし，最大の改善は，われわれが「事後的（ａ ｐｏｓｔｅｒｉｏｒｉ）」な検証にのみ注意を向けていることを止めた時になされました。事後的な証明は元々，ロバート・フロイドの考え方でした。つまり「任意のプログラムを取り上げ，そのプログラムが正しいということの証明を試みよ」というものでした。これがあまりにもいやな仕事であったもので，彼の弟子のジェームス・キングといった人が機械化を試みるに至ったのであります。

次に発見されたことは，同じ問題に関していくつかの異なったプログラムが書けますが，あるプログラムは他のプログラムよりも遙かに正しさが証明しやすいということでした。このことに気付いてからというものは，個々の問題について，正しさの証明がうまくできるように注意を集中しようということになりました。プログラマーの仕事が大きく変ったというわけです。すなわち，プログラマーが単に正しいプログラムを書くというだけではもはや十分ではなく，プログラマーはプログラムの正しさが証明しうるような形のプログラムを書くことも要求されるようになったのです。こういうと皆さん方は，正しさを実証できるようなプログラムをどうしたら書けるのだろうと考えられるでしょうが，そうするためにはプログラムとその正しさの証明を同時に開発する以外にないのです。このようなことを始めますと，つまりプログラムとその正しさの証明を同時に開発するようになりますと，その途端に状況は一変します。何故かというと，実際には，正しさの証明の開発が，プログラムよりも少しばかり先行するからです。そして，今や新しい段階に達したのです。つまり，まず正しさの証明を設計し，出来上った正しさの証明からプログラムを引き出すといってもよいでしょう。今やわれわれは，ある種の正当性をもって，プログラムを形式的に誘導するための計算法（ｃａｌｃｕｌｕｓ）の存在を口にすることができないというわけ

です。

これは非常に激烈な変化をもたらしたものなので，もう一度別のいい方で表現してみたいと思います。すなわち，あなた方が設計しなくてはならないのは，「似合いの一組み（matching-pair）」なのです。つまり，一方はプログラムであり，もう一方は，そのプログラムを正当化する正しさの証明から成る一組みです。ところが実際には，最初にプログラムが与えられて，それにふさわしい正しさの証明を推し出すのは，非常に難しいのですが，逆に，正しさの証明から始めて，それからプログラムを引き出す方は些事に近いといってよいでしょう。繰り返しお話したのは，この変化の意味を考えていただきたいからです。この変化は激烈な社会的影響を及ぼすと考えています。何故なら，この変化は伝統的な労働分業体制に大きな変更を余儀なくするからであります。

１０年前，つまり私がいま申し上げたこと自体が未だ十分理解されていなかった頃に，いわゆるソフトウェア生産という伝統が既に確立されておりました。そして，プログラマーの任務はプログラムを製作することだとされていましたが，これは大きな誤解であったと思います。

ともかくその結果，自動車とか，テレビ・セットそれに洗濯機等々，より伝統的な生産工程と非常に類似しているとの仮定に基づいてソフトウェア生産の管理が組織化されたのです。そしてこのようなソフトウェア生産に関する認識が６０年代に非常に大きな影響を与えたのです。

第一に，生産ラインの労働者にプログラマーをなぞらえたことから，「１カ月に作成するコードの行数」をプログラマーの生産性の尺度にすることが管理者によって受け入れられてしまったわけですが，これは最悪の尺度であります。何故ならこのような尺度を採用することは，ばかげたコードの作成を奨励することになるからです。のみならず，これは誤解に基づいた尺度であります。すなわち，プログラマーの任務は，プログラムを作成することではなく，問題を解決することであります。そして，彼が書くプログラムは製品ではなく，目的のための手段にすぎません。そこでプログラマーによって作成されたインストラクションの数を云々すべきものではなく，目的達成のために「必要だった」あるいは「用いられた」インストラクションの数として問題とすべきでしょう。

第二に，通常の生産ラインになぞらえて，ソフトウェアのマネージャーは安い労働力を用いることにより，ソフトウェアの生産コストを切り下げようと試みました。つまり，一定の労働環境の下で教育程度の低い人を使おうというわけですから，その結果が悲劇的になるということは目に見えていました。

第三に，生産工場と同じように考えたがために，更に大きな過ちが犯されました。すなわち，ソフトウェアの品質管理を行なう独立したグループがソフトウェア会社内に導入されたのです。つまり，あたかも電球か何かの品質管理のように，実際にコード生産に当たるグループとは別のグループがコード生産担当のプログラマー・グループの仕事をあとから検証したのです。

前にも申し上げたように，ある特定の問題に関して，二通りの正しいプログラムを書くことができ，その一方の正しいプログラムを正しいと証明することは容易ですが，もう一方については多分正しいのでしょうが，それが正しいと証明することは不可能です。何故なら，あまりにも複雑だからです。ですから，先ほど申し上げたように，プログラミング・グループの仕事は，正しいプログラムの生産にとどまらず，自らその正しさを証明できるような形で生産をするということです。というのも，いわゆる品質管理グループによる品質管理などというのは全く不可能であり，無

意味なものなのです。いわゆる品質管理グループが実際にやろうとしたのは，こうしたプログラムのテストであったわけで，一体これがどの位ばかげたことであったかを示す例として次のことをあげておきましょう。

名前を言うのは差しひかえますが，ある大手のアメリカ企業内で，FORTRAN のトランスレータの開発に当たっていたグループがありました。このトランスレータが完成したとき，このコンパイラーは，品質管理グループへ送られ，２０のテスト・ケースにかけられて，沢山のバグがあることが判りました。そこで２０のテスト・ケース毎のバグを付記してプログラミング・グループに「直して呉れ」と送り返してやりました。コンパイラー作成グループはバグを取り除いて，再びコンパイラーを品質管理グループへ回しました。そこで品質管理グループが，別の２０のテスト・プログラムを使って沢山のエラーを発見して，またプログラマーへ送り返してやったのです。プログラマーはかんかんに怒り，「標準を変えたりしたら，われわれは正しいプログラムなんか作れっこないじゃないか」と叫んだということであります。これではまるでこのFORTRAN コンパイラの目的はこの２０のプログラムを処理することにあったのだと言わんばかりです。これは信じ難いかも知れませんが，本当にあったことなのです。

ソフトウェアの生産にみられた伝統的な分業に関してはこの位にしておきますが，ただこういった伝統は，誤った仮定に基づいているものであり，打破されるべきであり，近代のソフトウェア開発の手法が遙かに効率的であることから，こうした伝統はいずれは打破される運命にあります。将来，いつの日にか過去を振り返って見た場合に，今日の伝統的なソフトウェア生産のやり方が，科学的な手法を必要としていた分野に，手工芸的手法が適用された結果であると判ることでしょう。

手工芸から科学への移行の意義を考えてみたいと思います。中世のヨーロッパの絵画の世界において遠近法が発見されたときに，何が起こったか想い起こして下さい。それ以前の時代におきましては，画家というものは手工芸家であって，長い経験をもった優れた画家であったなら，遠近法を知らなくても比率正しく物体を画くことができました。ただ彼が物の形を正確に写しとれるかどうかは１回１回やってみなければ成功するかどうか判らなかったのです。しかし，その中に新しい画家の世代が突然と現われました。この人達はアルブレヒド・デューラーの直弟子，もしくは彼の孫弟子だったわけですが，透視画法の法則を知っていて，それに基づいて絵画を描きあげたのです。そして次の世代の画家達にとっては，遠近法は実験段階を既に脱し，その意味では遠近法を用いることには何の危険も伴いませんでした。次の世代の画家達はどうやって画けばよいか知っていたし，そして，単に手法を知っていたばかりか，彼等は彼等が手法を知っているということを知っていたのです。また，手法についてはどう説明したらよいかも，またその次の世代にどうやって教えればよいかをも知っていたのです。

そのとき，それまで直感的に勘に頼って行なわれていた絵画芸術の一部分がはじめて形式的処理を受け入れるようになったのです。プログラムに関しても，われわれは今丁度同じような過渡期を経験しつつあるのです。そしてこれは十分理解しうることでありますが，こうした手工芸が科学的学問に取って代わられようとする場合には何時でも，古い手工芸家達，つまりギルドの古いメンバー達は非常に自分達が脅威にさらされていると感じるのです。

先ほど私は急激な社会的インパクトを与えると言いましたが，何故かというと現在，５０万から１００万ものプログラミングに携わっている手工芸家がいるわけで，そのほとんどの人達にとって，これからでも自分達は改め

て科学的な態度を取りうるのだと考えることは全く非現実的なのでありますから，こうした人達にとって，プログラミング技法にみられる最近の進展は，深刻な脅威を与えるものであり，また逆に，こういう人達が存在することは，新しいプログラミング手法を広く広めて行くのに非常な障害となっているのであります。こうした新・旧の対立にかんがみて，古いプログラミングの伝統が何時，何処で，どのようにして，大規模に打ち破られはじめるか，予測するのは非常に困難であります。ですから私なりの意見をいくつか申し上げるにとどめて，後はみなさんの推測にお任せしたいと思います。

8年前の1968年に，ソフトウェア危機が初めて公に認められたわけでありますが，当時ピッツバーグのカーネギー・メロン大学の教授をしておられ，現在ではエール大学で教鞭をとっておられるアレン・パーリス教授が正しさの証明という考え方を「牧歌的である」として排斥してしまいましたし，英国のニューキャッスル大学のブライアン・ランドルフ教授も，同じ会議の席上，正しさの証明ということを考えただけでも，「頭の中でシャックリ」が起こりそうだと言いました。

4年後の1972年におきましても，プログラミングは非常に難しいから，科学的に訓練されたプログラマーが必要であるという提案が依然として排撃されるのを私は目撃しました。当時の「平均的なプログラマー」の知的能力を考え合わせた場合，この提案は全く馬鹿げていると排撃されたのです。

それから更に3年後の昨年のことですが，4月にソフトウェアの信頼性に関する国際会議がロスアンゼルスで開催されました。会議のテーマからして形式的手法というものが主要な役割を果たし，1,000人以上の参加者がありました。

そのまた1年後，つまり今年のことですが，アメリカ政府に対するある勧告の草案を私は見たのでありますが，それは「歴史上の偶然のためにたまたま，アメリカは世界の他の国々と比べて現行のソフトウェア慣行を維持して行こうとする慣性が強いし，国内に既得権の問題をもっている」との警告を発しております。この文書はアメリカ政府に対する提言であったわけですが，アメリカがさらされている深刻な危機を指摘しているのです。つまり，今後10年以内にプログラミングに関して柔軟性をもち，旧式の技法について大きな既得権を持たないが故に近代的なプログラミング手法をより早く採用しうる，ひとつないし複数の国々によってアメリカがリードされてしまうであろうというのです。

以上，私なりの観測を申しあげたわけですが，一体何時，何処で，どのようにして古い伝統の打破が行なわれるかはみな様方の想像におまかせしたいと思います。

コンピュータのプログラミングに成功するためには，数学的な内容を含んだ相当量の科学的な教育が必要であるといった結論は，前にも申し上げたように，ギルドのメンバーには歓迎されません。彼等はそれを拒否するのみならず，コンピュータを普通の人の手に取り戻すことが最も望ましい目標であるかのようなふん囲気を醸成しようとしており，また，そうした可能性を前提として話をしている訳で，それについて議論しようともしません。（可能でないことを論証する方が遙かにやさしいから当然かもしれません。）また彼等は，ユーザーが教育を受けなくても済むように，難しいことは機械にやらせることを提案したあらゆる種類の人工頭脳に関するプロジェクトに，資金が流れ込むような環境を作りあげているのです。しかし，私はみな様方に警告しておきますが，こうしたプロジェクトに資金がついていることは，こうしたプロジェクトに意義があることを示すものであると解釈してはなりません。むしろこうしたプロジェクトに資金が出されているという事実が，そ

れをめぐる政治的環境を如実に物語っているのであります。

「自然言語」とか「ノレッジ・ベース」等々のこうしたプロジェクトの全てが失敗に終わるであろうと私は確信しております。そして，それらのプロジェクトが野心的であればあるほどその失敗はみじめなものとなりましょう。したがいまして，私はこうしたプロジェクトが馬鹿げたものであり，科学の一領域として教えるに価するプログラミングにとっては，むしろ無害な後衛的な行為であると考えており，気の毒だとさえ思っています。

そうは言うものの，こうしたプロジェクトが多くの害を与える可能性はやはりあります。というのは，将来開発されるシステムがあたかも非常に高度なものであり，それが減少しつつあるハードウェア・コストとあいまって経済的にも魅力的なものであり，次の世代に対する教育の義務を放棄してしまってもよいといったような印象を抱かせるような偽りの約束をすることにあります。このような誘惑のわなに陥ってしまうことは,われわれが70年代に犯しうる最大の文化的失策であることは申すまでもありません。

私はときどき，ある意味では既に害が及ぼされているのではないかと思うことがあります。つまり，特にコンピュータ・プログラムの正しさの証明というものは，本質的に長ったらしく，単調で，退屈であり，面白くない，しかもエラーを起こしがちであるから，プログラムの検証は機械化すべきであるとの民間伝承が広く流布されています。しかし，こういった前提は間違っており，プログラムの正しさの証明も，他の数学分野と同じように美しく，魅力的で，しかも説得力に富んだものになり得るし，またそうでなければならないのです。しかし，それとは全く正反対の風説が，機械的検証システムのための広告宣伝活動を通じてたえず喧伝されているのであります。人工頭脳に関するプロジェクトのほとんどのものの最も顕著な特色は，これらのプロジェクトをサポートするために必要だとして大々的な宣伝活動が行なわれているところにあるようですが，これではわれわれが不信感をもったり，疑問の念を抱いても当然なのではないでしょうか。

セッション Ⅲ ＜パネルディスカッション＞

プログラミング：手工芸から科学へ

プログラミング：手工芸から科学へ

パネリスト　上条史彦[1)]
岸田孝一[2)]
藤村礼吉[3)]
水野幸男[4)]
エドガー・ダイクストラ[5)]
コーディネータ　森口繁一[6)]

森口　それでは前セッションと同じような手順で進めたいと思います。

一番初めの上条さんは情報処理振興事業協会におられるわけですから，先ほどのダイクストラさんのお話の中に出てきたギルドをたくさん相手にしていらっしゃる立場におられます。御本人はいまやオブソリートになった，ないしはなろうとしているプログラマーの人だと言っておられますが，まず上条さんお願いします。

上条　本日は世界的に高名なダイクストラ先生と同席させていただきまして，つたない意見を述べさせていただけるのを大変光栄に存じております。

先生がおっしゃっておりましたプログラムに関する正確さという問題，それから効率という問題を2つの大きな目標と考え，中でも特に正確さの保証を先生が言っておられましたように制作工程において行うべきだとする新しい考え方を取り上げてみますと，これは共鳴する点こそたくさんございますが，異論を差しはさむ余地はどうもあまりなさそうでございます。といいましても，全然先生のお考えに異論を唱えないというのでは全く討論にもなりませんので，私は私の立場から理論と現実とのギャップといった問題，あるいはそういう側面を取り上げまして，実務といった立場から実際の場で理論を実現していくむずかしさというものについて述べてみたいと思うわけです。

さて，現在私の手元で委託しておりますプロジェクトの数は，契約という数で数えますと年間約40前後になります。成果物の数，プログラムの数として考えますと20件ぐらいになります。それぞれのプログラムの大きさは，これは先ほどダイクストラ先生からステップ数云々という話が出ましたが，私どもの方ではステップ数ではなくてお金で数えておりまして，一件当たり平均5,000万円から1億円前後と考えていただいてよろしいのではないかと思います。

直接私の手元で開発作業を実施しているわけではございませんので，私の立場をかいつ

1）情報処理振興事業協会開発振興部長
2）ソフトウエア・リサーチ・アソシエイツ専務取締役
3）中央電算研究所常務取締役
4）日本電気基本ソフトウエア開発本部長
5）アイントホーベン工科大学教授
6）東京大学教授

まんで申し上げますと，マニュファクチャラーというよりは，注文主といいますか，ちょうど造船関係，船の関係で言いますとシップオーナーに当たる感じでございます。したがいまして，一般的に行なわれておりますプログラムの外注とはちょっと違う面が出てまいっておりますのでその制約を申し上げておきますと，まず第一に私どもでつくっておりますプログラムにつきましては多数の利用者を予定している開発である，これはあくまでも予定であるということ。それから，私ども自身はでき上がった成果物を使用する立場には全くないということでございます。三番目に，使っておりますお金が国の一般会計であるために，たとえば納期が非常に厳格に限られているという，通常のシステム開発では見受けられないような変わった制約があるわけでございます。

さて，こういう仕事をしておりまして，過去何年かの経験を通じて言えますことの一つに，プログラムの制作という事業におきます人間臭さ，これはプログラムそのものも非常に人間臭い面がございますが，そういう点がございます。これはプログラムそれ自身が独立の品物ではなくて，ハードウエアと一方においては密接な関連を保たなければならないし，また他の面におきましては社会制度あるいは人間の諸活動と直結しているということから来る問題だと思います。たとえば事務処理業務の中で最も早くから電算化されております給与計算を考えてみますと，その中には税制の問題でありますとか，あるいは従業員福祉の問題といったものが実にいろいろ入っております。非常に生臭い問題をたくさん含んでおるわけでございます。したがいまして，それをプログラム化しようといたしますと，これは単に計算のアルゴリズムを命令群に変換するといったような形だけではなく，その中に人間的な活動の写像としての特性を持ち込まないといけないわけです。どういう特性かというと，非常に目立ちますが時間的な変動要因，すなわち非常に不安定であるということです。アルゴリズム自身が非常に不安定であるということ。それから利用の局面，利用の場面，すなわち会社とか，そういったところによってその条件が非常に変化する，すなわち多様性の問題があるということです。この不安定性と多様性といいますのは，私どもがやっておりますパッケージ・プログラムのビジネスの泣きどころであるとともに，プログラミングの技術の面では理論を実用化するという上で最大の問題点になることだと考えております。

妙な言葉を持ち出して恐縮でございますが，かつて私がプログラマーでありましたころ，ビューティフルなプログラムという言葉が使われていた時代がございます。このビューティフルなプログラムというのは先生がおっしゃいましたビューティフルとはややニュアンスが違いまして，その中にはかなりよい意味でのスマートさといいますかスマートネスといいますか，そういうものが含まれているような気がいたしますが，そういうビューティフルなプログラムというものが実は私どもの現在のパッケージド・ビジネスにとっては非常に重要な要素であるわけです。先生のおっしゃるようにコレクトでありかつエフィシェントであるということは必要条件でございますけれども，それに加えて先ほどの人間臭さにいかに対処していくかという面も重要でして，多少時代が変わりましてもあるいは環境が変わりましても，たとえばデータのテーブルを拡充するといったような簡単な処置によってそのプログラムの寿命を延ばせるという柔軟性が要求されるわけでございます。昔のことを考えてみますと，そういうプログラムを非常にビューティフルにつくるプログラマーとそうでないプログラマーの差というのは非常に開いておりました。

その原因をいろんな人が考えておりました

が，その中の一つにプログラマー適性といったものが持ち出されておったわけで，プログラマー適性というようなものを持ち出しますと，おまえは先ほどの森口先生の御紹介にありますギルドのメンバーなのかということになるのじゃないかと思いますけれども，私は何もニュー・プログラミング・テクノロジーにさおを差そうというのではありません。むしろ現在私が抱えております問題の多くはこのニュー・プログラム・テクノロジーによって解決可能なものであるということには，先ほど申し上げましたように何の異論も持たないわけです。しかしながら，パッケージ・ビジネスに限って申し上げますと，ビジネス上必要なビューティフルなパッケージというものは先生がおっしゃったエスタブリッシュド・ワーク・ルールだけではつくれないのではないかと思うのです。すなわちワーク・ルールをエスタブリッシュということは非常に重要でございましたが，これはあくまで全体的な水準，平和的な水準を持ち上げるという面に即効的でありまして，それに加えて私どもの商売の場合には何らかの才能のひらめきといいますか適性といいますか，そういうものをつけ加える必要があるような気がするわけです。一般的に職人芸と言われますのは才能と努力の結合だ，人によって才能の割合と努力の割合は変わっておりますけれども，おおむね努力の方が大きいというのが通説であるようです。しかしビューティフルネスというのはどうも一旦われわれが忘れ去ろうとしております職人的要素といいますかクラフトをつくるときの要因といいますか，そういうようなものに一脈通ずる点があるという気がするわけでございます。

先生のおっしゃいますように，プログラミングをクラフトの世界からサイエンスの世界に引き出すということが現在の非常に大きな問題でございますし，それを行うために大きなブレークスルーを行わなくちゃならないと思います。それに対しまして現在の状況は，やはり大方のプログラム開発は個人的経験の集積と言えます手法を抜け出していないわけです。これを一挙に新しい世界に持ち込むというのは容易でございませんで，最近のような人件費の上昇でありますとかあるいはコード・プロダクションの伸び率がプログラマーの人口に合わないというような問題が一つのブレークスルーを助ける要因になると思いますけれども，それにしましてもここしばらくの間は現状が続くという，どちらかというとやや悲観的な考え方を持っております。しかし，いずれは洋服に見られますように，コスト・パフォーマンスの面からレディーメードの洋服というものがもてはやされる。プログラミングにしましてもそういうルールでつくられたプログラムというものがもてはやされるようになっていくと思います。けれども，一方においては，多少オーダーメードの世界というものが残っていくと思います。

そういうような点をいろいろ考えてみますと，どうもこのプログラム作成というビジネスの中にはいろいろなものがある，非常に多様性に富む世界であるというふうに見ることができるのではないかと思います。

簡単にそれを仕分けしてみますと，一度限りの使い捨て型のプログラムというのが，これはコードの量だけで考えますと非常に大きな割合で世の中に存在するわけでして，先生はこれをノンプロフェッショナルなプログラマーが行う分野に入れておられますけれども，プロフェッショナルなプログラマーが行う作業の中にも大量にあるわけです。これに対しましては，われわれは使いやすいランゲージを開発するということであるとか，あるいは新しい製造のシステムを開発するというようなことで対処すべきではないかと思います。

それから二番目に，一つの電算機ユーザーがもっぱら専用で使うプログラムというのがございまして，これも非常に大きな比率を占

める分野です。これは現在いわゆる内製，社内だけでつくるという行き方と，外注，外の製造能力を使うという行き方と，大体大ざっぱに2つに分けられております。内製品といたしましては，通常その会社にプログラマーを雇いまして，自社の社員といたしまして大ぜいのプログラマーを組織化し，そしてそこで自社開発の形をとるわけでございます。これに対する近代化の対策は，私は治具工具の整備という点にあると思います。現在のように毎回プログラムを開発するにあたってゼロからスタートするということをなくして，治具工具のたぐいを整備することによりまして，ある程度のものはすでに既存の土台からつくっていく，そうすることによって経済的な問題を今後避けていけるのではないかと思うわけです。

ユーザーのプログラムの外注分野につきましては，これは大ざっぱに考えまして，先ほどの洋服の世界ではございませんけれども，オーダーメードのケースとレディーメードの，つまりパッケージを使うケースとが現在考えられます。

オーダーメードの世界につきましては，これは実際につくりますのはプロフェッショナルなプログラマーでございますから，この分野におきましては先生のおっしゃるようなサイエンティフィック・ディシプリンを十分に活用するような体制を，これは何らかの形でつくっていく必要があると思います。

それからもう一つはレディーメードの分野でございますが，これは一度つくれば後はコピーするだけというのが本来のものでして，それにつきましては品質が非常に問題になります関係から依然としてクラフト・ワークが続くのではないかというのが私の考え方でございます。

現在，世の中ではその中間的な分野も考えられておりまして，一種のイージーオーダーといいますか，たとえば途中まで部品をつくっておき，そしてその部品を組み合わせてオーダーメードのプログラムを仕立てるという話も進んでおります。これは現存するシステムで実際実用になっているものは非常に少ないあるいはほとんどないと思われますので，今後の可能性の探究にゆだねられるべき分野ではないかと考えております。

最後に，三つ目のカテゴリーといたしましては，普遍的に利用されるソフトウエアの一群があります。これはオペレーティング・システムのパーツであるとか，あるいはランゲージ・プロセッサーであるとか，あるいはユーティリティー関係のツールであるとか，先ほど申し上げました治具工具もこの中に入ると思いますが，これらは先生が今回の議論にあたって対象とされておられたのではないかと思う分野でございまして，最もサイエンティフィックなディシプリンが取り入れられやすいわけですし，製造環境も大きなメーカーの工場等でございますので，非常にそういう革新に耐え得る体制になっているのではないかと愚考しております。

以上，パッケージ・ビジネスを中心にいたしまして私のつたない経験を申し述べたわけでございますけれども，このソフトウエアの世界というのはあまりに守備範囲が広いためになかなか一義的な議論というのはしにくいという感じを実はこの議論を終えて私自身持ったという結論で話を終わらせていただきたいと思います。

森口　豊富な経験に基づいていろいろな側面を吟味し，そして恐らく後ほどいろいろと議論がはずむような種を出してくださったと思います。

その次は，ソフトウエア・リサーチ・アソシエイツの専務取締役をしていらっしゃる岸田孝一さんですが，岸田さんは私の想像するところでは先ほどの言葉でギルドと呼ばれるような人たちを指揮して仕事をしているんだろうと思います。しかし，同時にまたダイク

ストラさんの言っておられることに前々から関心を持っておられたようで，たとえばハンブル・プログラマーという有名なチューリング・レクチャーがありますが，その内容をいち早く雑誌に紹介したのも岸田さんであると聞いております。では岸田さん，どうぞ。

岸田　私がきょうここでお話しする立場は二つの面を持っておりまして，私自身その矛盾をどう解決したらいいかと常々悩んでいるのですけれども，一つは私自身も十数年プログラミングを自分の仕事にしておりまして，そういった一人のプログラマーとしての立場と，もう一つはいま森口先生のお話しになりましたようなプログラマー・ギルドの一つである小さなソフトウエア会社のテクニカル・マネジメントを担当している人間としての立場と，二つの立場でお話を申し上げたいと思います。

まず第一に，プログラマーとしての私は，例のソフトウエア・クライシスなんかの議論が始まりました1960年代の半ばごろからプログラミングの方法論については私なりに強い関心を持っておりまして，ダイクストラ教授の仕事も早くからいろいろな論文を読んでおりました。そういった意味できょうの教授のお話には私一プログラマーとしてはほぼ全面的に賛成の意を表したいと思います。もっとも，若干のニュアンスは違いますが，その違いは多分二人のひげの生え方の違い程度ではないかと思います。どんな点が違うかと言えば，私自身はプログラムのコレクトネス・プルーフということについてはあまり関心を持っておりませんで，むしろプログラムの構造の美しさといいますかその美学といいますか，そういった方面に私の関心はより強く向けられております。その原因が何かということについていろいろ考えてみますと，恐らくその一つには，私は25歳ぐらいからプログラマーを始めたのですが，それ以前，プログラマーになる前に絵かきになろうといろいろ勉強していたことがありまして，そうした言ってみれば職人的なプログラマーとして，絵を描くかわりに，抽象絵画をかわりにフローチャートやプログラムのストラクチャー・チャートを一種のデッサンのようなつもりで書いていたというあたりが多分そういった私の関心の違いの原因ではないかというふうに思います。

先ほどのダイクストラ教授のお話の中に，プログラミングにおけるプロフェッショナルのプログラマーの仕事と，アマチュア・プログラマーの仕事の違いについての議論がございましたけれども，私自身はその問題については大体次のように考えております。よくコンピュータというのは問題解決の道具である，プログラム・ソルビングのための道具であるというふうに言われるのですけれども，確かにアマチュアのプログラマーといいますかあるいはエンド・ユーザーといいますか，そういった方々はコンピュータによって解くべき何らかの問題を自分のところに抱えているわけで，その問題を解くために道具としてコンピュータというものが自分の前にある。したがって，アマチュアあるいはエンド・ユーザーにとっては，プログラムが仕様書どおりに正しく動作して，望むアウトプットがきちんと出てくるということが最大の関心事だろうと思います。

一方，プロフェッショナル・プログラマーの立場はどうかと言いますと，彼は自分自身に解くべき問題を別に持っているわけではないわけです。つまりだれかほかの人が解くべき問題を持っていて，その手助けをし，そのためのプログラムを彼はつくっているわけです。したがって，そのプログラムが動いて，そのプログラムによってどんなアウトプットが出てくるかということは，実はプログラマーの関心の外にあることであるというふうに思います。外にあるということは決してアウトプットがでたらめであってもいいということではなく，むしろプロフェッショナルなプ

ログラマーにとりプログラムが正しく動作してユーザーの望む正しい結果が出てくるということは仕事の目標ではなくて前提条件である。プロである以上，自分のプログラムが正しく動いて当然であると考えるべきだろうと思います。

ではプログラマーの仕事の目標は一体何か考えた場合，一つの問題を解いて正しい計算結果を求めるためのプログラムにはいろいろな書き方があるわけで，同じアウトプットを出すにもどのようなプロセスを使ってアウトプットを出すのがその問題に一番ふさわしいか，そういったことを考えるのがプログラマーの仕事であると思います。したがって，そういったプログラマーの努力の結果は，いわば美しい計算プロセスをもたらすような美しいプログラムの構造という形で具体化されるものであるというふうに考えます。

言いかえれば，その議論をさらに進めますと，プロフェッショナルなプログラマーにとっては与えられた個々のプログラミングの問題というのは常に単なるエグザンプルでありまして，もっとも世の中のプログラムというのは常にそういった具体的な問題を解くプログラムとしてしか存在し得ないのでそれは仕方がないわけですが，プロフェッショナル・プログラマーはそういった数々のエグザンプルのプログラムを解くことによってより純粋な，より抽象的なプログラムの構造の美しさを求めるというふうな形で仕事をするんじゃないかというふうに思います。もちろんそうした美しい構造というのは，そのプログラマー自身の頭の中での理論的に美しい思考プロセスといったものの反映であるというふうに言えるだろうと思います。それはちょうどシステム・エンジニアとかシステム・アナリストとかいった人々の仕事が，具体的には特定の在庫管理システムであるとかあるいは何々システムをつくり出すことであるわけでありますけれども，彼らの直接の関心がそういった特定のシステムよりは，むしろシステムをつくる過程を通じてより一般的なシステム的な思考方法というものを追求することにだんだん向けられていくというのと同じようなことであるだろうと思います。

私がダイクストラ教授のきょうのお話に基本的に賛成をしたいと考えておりますのは，教授の言うフォーマルなディシプリンといったものが，そういった言ってみれば具体的なプログラミングのテクニックとかノーハウとかいうものではなくて，もっとより根本的な一種の技術的な哲学，フィロソヒーのようなものではないかというふうに考え，そういったフィロソヒーがこれまでわれわれの仕事の分野で欠けていた，それを早急に確立する必要があるという意味で全面的に賛成の意を表したいと思うのです。

ただ，そういうプログラムのストラクチャーの分析や設計に非常に形式的な科学的な方法論というものを使っていく場合に一つわれわれが注意しなければいけない点があると思います。それはちょうど私が昔絵の勉強をしておりましたときに，ドイツの抽象画家でパウル・クレーという人がおりますけれども，クレーという人は画家としても非常にすばらしい作品を残した20世紀で最高のアーチストの一人であると私は考えますが，彼はまた教育者としても有名でありまして，第二次大戦前のドイツにありましたバウハウスという造形学校でデザインの教授をしておりました。その彼の講義録の中の言葉として，やはりデザインの分野でもそういったデザインに関する，造形に関する科学的な方法論というものの必要性をクレーは説いて，このような精密な科学的な方法論がわれわれに対してもたらす効果といいますかメリットというのは未知の大きなエックス，つまりアンノウン・ビッグ・エックスを包囲することであって，それだけである。そのエックスそのものに直接アタックするのは直感力の助けを借りざるを得

ない。つまりどのように精密な科学的な方法論を打ち立てようと，最終的には芸術家にとっては芸術的な直感力が必要であるということを述べております。プログラミングの世界においても同じようなことが多分言えるのじゃないだろうか。いかに科学的な方法論，プログラム設計の方法論ができようと，最終的にはそのプログラムを書くプログラマーの芸術的な直感力というものが作品であるプログラムの品質にかなり重要な本質的な貢献をすることになるだろうというふうに思います。

そこで問題は，そういった直感力が一体どのような方法によって教育あるいはトレーニングできるかということですが，そのへんについて私自身もどうしたらいいか回答はまだ自分自身得ておりません。

それからまた，それ以前の問題としましても，先ほどダイクストラ教授がプログラミングにおけるサイエンティフェック・ディシプリンといったものの必要性を説かれましたけれども，それは確かにそのとおりですが，そういったサイエンティフェック・ディシプリンというのは，恐らくトレーニングなりエジュケーションが非常にむずかしいものになるだろう。というのは，いまのプログラマー・ギルドを構成している大多数のアベレージ・プログラマーにとってはそういったことの習得は非常にむずかしいものに違いない。もしむずかしくなければギルドからの反発はないわけでありますから。したがってそれは確かに教授の言われるように一つの社会的な問題でありますけれども，社会的な問題を提起した以上はそれを何らかの意味で社会的に解決することをわれわれは考えなければいけないんじゃないかというふうに思います。その辺を教授に後でお聞きしたいと思うのですが，もしかしたら教授のお答えは幾つか精神病院をつくればいいじゃないかということになるかもしれませんが，その辺についても私はまだ自分自身で納得のいく回答は得ておりません。

先ほど申しました私のもう1つの立場であるソフトウエア会社の経営者としての立場で次に少し補足を申し上げておきますと，私自身現在ソフトウエア会社でそういうプログラマー・ギルドの一部分を指導しているわけでありますが，そういう実務的な観点から言いますと，現在私が指導している大多数の，中には非常にすぐれたプログラマーもおりますが，大多数の平均的な知的能力を持ったプログラマーたちに対して，こういう新しいプログラミング方法論をどのように伝えていったらいいかということが非常に大きな問題としてございます。これはまだ未解決ではありますけれども，先ほど上条さんが少し言われましたように，そういった決して優秀なプログラマーではないけれども平均的な力を持った現在のプログラミングの職人たちに使えるような，それ向きの新しい方法論といいますかあるいはその方法論を具体化したプログラミング・システムなりソフトウエア・パッケージというものができるだろうという楽観的な予想を私個人としては立てております。

それから，最後にもう一つ申し上げておきたいのは，現実のソフトウエアの世界において発生し，われわれソフトウエア・ハウスの人間が一番悩んでいる多くの問題は，ソフトウエアに関する多くのトラブルがプログラミング方法論を論ずる以前の時点で発生するということであります。それはきょうの話題とは若干ずれるわけでありますけれども，プログラムのスペシフィケーションの問題でありまして，ほとんどのソフトウエア・トラブルというのは私どもがソフトウエア・ハウスとして仕事をする場合にクライエントとプログラマーとの間のソフトウエア・スペシフィケーションに関する打ち合わせのミスといいますかお互いの誤解といいますか，そういったものによって発生している。したがって，現在私がソフトウエア・ハウスにいて非常に痛

切に解決を迫られている問題を解くためには，プログラミングの方法論以前にそういったプログラム・スペシフィケーションの作成プロセスをどのように形式化するかという問題が非常に大きな問題としてございます。この点について教授がどんなふうにお考えになっているか，後でまたお聞きしてみたいというふうに考えます。

以上です。

森口　ありがとうございました。岸田さんの芸術家風な風貌の由来がわかるようなお話がありました。なお，重要な問題点を提起されました。

次の藤村礼吉さんですが，私はこの中央電算研究所というのをよく知らなかったものですから御本人にお伺いしましたら，FACOMのディーラーでありかつ計算サービスセンターの業務が主なんだそうです。しかし，きょうここにおられます一つの理由は，昨年でしたか，ＩＰＴの実情調査のために渡米した団体のリーダーであられたということのようです。このＩＰＴというのはインプルーブド・プログラミング・テクノロジーの略と承知しておりますが，これは先ほどダイクストラさんが包括的にリファーされた対象に該当するのではないかと考えております。藤村さん，どうぞ。

藤村　中央電算の藤村でございます。

昨年の11月でございましたか，私ども計算センターの協議会がございまして，そのセンター協議会の中でソフトウエア関係の部門で一つの大きな話題になりましたのが，ダイクストラ教授が御提案された「GO TO レスは有効か」という形から発展しましたストラクチャード・プログラムということについての新しい技法がいわゆるユーザー・サイドにとって非常にコスト低減につながる，それと生産性の向上につながるといったことで，こういう話からそれの実態調査に行こうということになり11月に参りまして，7～8社の企業の実態を見てきたわけでございます。そのときにきょうのセッションでダイクストラさんがいわゆるストラクチャード・プログラムについて一つの言語とシステムについての国際登録をされなかったということに大きな損失があった。それは大きな誤りであったというふうにおっしゃっておられましたけれども，実際に現地で使っておられる実情を見ますと，ダイクストラさんがお嘆きになることも非常によくわかる，それほど非常な成果をおさめている会社があるということと並行しまして，まだ非常に反論も多いということがアメリカの実情の中の1つの大きなポイントだということがわかったわけでございます。そういった背景の中でわれわれがいわゆるユーザー・オリエントという立場からこのストラクチャード・プログラムをダイクストラ先生の思想的なお話をもとにして現実に使っていくためにはどういう点がポイントになるか，それから現実の実態がどうか，最後に日本でそれを適用するための一つの問題とわれわれとしての考え方を私なりにまとめて御報告させていただきたいというふうに考えております。

システムの開発手順の確立ということが，アメリカの場合には，実施に当たりまして非常に大きなポイントとして言われまして，1973年の段階ではいわゆるモジュラー化構造という形の方式がストラクチャード・デザインそのものとして確立されていないことによる非常に大きな失敗論というのが一つ提起されておりました。いわゆるマクドナルド・ダグラスのオートメーション・カンパニー，ここでも1973年にストラクチャード・プログラムの導入に対して一度失敗をし，1974年の初めに再度挑戦をして成功されたという形の報告が挙がっております。これらに起因する大きな問題は，先ほど岸田さんがお話しになりましたように，いわゆるプログラミングの前のスペシフィケーションの確立ということに対する大きな組織的な変化が大前提に

ならざるを得ないために，ソフトウエアの信頼度と生産性の問題がこのストラクチャード・プログラムでは非常に大きくクローズアップされてくるという形でございます。その信頼度につきましては，いろいろな努力により，基本的には5倍から10倍以上の向上が期待できそうであるということが調査の結果で判明してきた。生産性の向上そのものにつきましては，大体1.4倍から1.6倍ぐらいと非常にばらつきが多うございます。これらはストラクチャード・プログラムの手法を導入してもしなくても，生産性そのものについていいものと悪いものはシステム自体にあるという形がございます。そういった背景から考えますと，大体それの平均の差は約5倍ぐらい生産性が向上するというような話もございますし，私どもで実際にあるトライアルをやってみた段階では，生産性の向上は約50％平均というところが日本の実情ではないだろうかというふうに考えておるわけでございます。これらもまだテスト・サンプルの状態だけでございますのではっきりとした生産性の数字は申し上げられないというのが現状の実態でございます。

ニューヨークにソフトウエアのコンサルタント会社でディーボルトという会社がございますが，そのディーボルト自体の予測でも，いわゆるＩＢＭが指向しておりますインプルーブメント・プログラム・テクニックとしてのＩＰＴの採用によって発見されるエラーの減少という形での信頼度は，最終的には10倍から100倍の向上で，コストの節減そのものは大体25％から50％どまりという見方をしているようでございます。この時点でのコスト節減のコストというのは，ひとつにはプログラマーのマンパワー・コスト，またひとつにはコンピュータのマシン・フィーでございます。

それから，アメリカのＨ・Ｄ・ミルズ氏はこの方式に対して，まず質の向上から着手するという提案をされておられます。ストラクチャード・プログラムの導入そのものに当たって，初めから生産性の向上ということを期待したようなスケジュールを組むこと自体非常に危険である。生産そのものは自動的に上がってくる。むしろ質の着実な向上を望んでそれなりのスケジューリング設定を行うという体制が望ましい。こういうことをミルズ博士がおっしゃっておられる。先ほどダイクストラ博士からのお話もありましたように，単純なコーディング・ミスそのものは単体テストですぐ見つかるということがございますが，基本的にシステム分析の段階で考え違いをしておる。これが先ほど岸田さんからもお話がございましたようにユーザーの受け入れテストのときまで発見されずに後で見つかるエラーとなり，開発初期の段階でミスの原因となっている。このために非常にやり直し作業が大きくなるわけでございます。

1973年の3月の「ＤＡＴＡＭＡＴＩＯＮ」誌に載っておりますベームの論文によりますと，ソフトウエアに費やされる努力はアナリシス・デザインで大体45％，コーディングあるいは単体テストで20％，インストレーションとしての総合テストで35％であるということです。そういった一つの体系を見ますと，きのうのセッションの中で一部アナリシス・デザイン70％というお話がございましたけれども，プロジェクトの内容によっては，むしろコーディングそのものの技法よりも，その時点におけるインスペクションの体制の確立と精密化を厳密に行うということがいわゆるストラクチャード・プログラミングとしての基本的な受入体制の大きな前提にならざるを得ないというふうに痛感しておるわけでございます。したがって，そのための方式としては開発の初期段階でエラーが埋め込まれないように万全のチェック体制をとるという形でのウォークスルーという一つのやり方，これがストラクチャード・プログラムでは非

常に大きな特徴とされておりまして，現実にウォークスルーということに対応するやり方では，現状のプログラミングのいわゆるアーキテクチャー方式でのプログラムの体制であったとしても，エラー減少率は非常に高まるということが証明されておりますし，現実に私どもの方ではそういった試行をできるポイントから着手しておるというのが実態でございます。

それと，複雑な形態のものをできるだけ単純化して理解しやすくしていく，それによって複雑なものの誤りを少なくしていくという形がいわゆるダイクストラ教授のおっしゃる入口一つ，出口一つというＩＰＴの一つの大きな基本原則に立っておるんではないだろうかというふうに考えておるわけであります。先ほどそれに対するプログラムそのものの証明というお話がございましたが，つくったその内容自体が正しいと確認しながらつくれることがストラクチャード・プログラムの考え方ではないだろうかと考えております。

システム分析上のモジュール化そのものの意味でございますが，ソフティックというソフトウエア・ハウスのドクター・ブラケットという方が，システム分析と設計の手法について基本的にはまず分割して征服せよというふうにおっしゃっております。分割して征服するということについては，いわゆるモジュール化の意義そのものを理解して，それに基づいた一貫した考え方を身につけないとモジュール化そのものの利点が引き出せないということが背景ではないだろうか。

われわれサイドで解釈いたしますと，短く簡単なプログラム化のねらいのためには，一つは読みやすくわかりやすい方式でなければいけない。

二番目には，マネージしやすいパッテーションであるということ。

三番目には，正しいかどうかチェックがしやすいということでなければいけないというような考え方になるわけでございます。

これらも下手に分割すると非常に複雑性を増したり，あるいは無用のやり直しをするというような結果になりますので，そのための手法としてHIPO上ではいわゆるスタブルの使用が非常に限定されておるというような背景になっているようでございます。

ラリーコンサンティンの言によりますと，プログラムをうまく分割する心配り，これを頭に入れておく必要がある，このことをストラクチャー・デザインというふうに定義づけております。この考え方はいろいろな教育カリキュラムの中に現実に取り入れられておりまして，一つのセグメントのリスト結果そのものがいわゆるドキュメントとして一ページの中におさまるという形が最終の理想であるというふうに言われております。つまり，プログラム・ステートメントという形では１モジュールのバックがいわゆる50行またはそれ以下をプログラムのテキストにおさめられるように体系化することというような言われ方をしております。このようにストラクチャード・デザインの機能モジュールに分割した後でHIPOにまとめていくという形でございますが，ヒューズ・エアクラフトという電子機器の製造会社で教えてくれました事例では，その前にでき上がりましたさまざまな数百のモジュール，そういう構造を人間の視覚に訴えて目で判断できるように体系をドキュメント化することに非常に重点を置いておられる。これらを別称としまして，ブルー・プリントとかウォール・プリントとか，いわゆる壁に張りつけて，それらのモジュール化構造のフローを見て全体の形態に誤りがあるかどうかという形の論議に非常に時間を費やすということでございます。

現在のプログラム過程でいきますと，われわれがやっておるやり方はウォール・プリント方式では決してございませんで，内部仕様書という形でのいわゆる文章表現に頼る仕様

書上の取り決めが非常に多い。ここに外部ユーザーとそれを受ける段階での基本的なスペシフィケーションでの誤りが文章の表現下に隠されてしまうということがおこります。そこでそれらを図示化し，相関をあらわして最終的なモジュール化構造に焼き直していくというのがこのヒューズでやっておる一つの大きな例のようでございます。

いわゆる部分的に最適化にならないようによく全体を見るということと，エラー処理のモジュールの位置，これらを決定していく。この設計どおりにつくれば間違いなくプログラムが働くという形の確信を持つまでレビュー行為が何回となく繰り返されていく，これがいわゆるストラクチャード・プログラムのストラクチャード・デザインに課せられた非常に大きなポイントではないだろうかと考えております。基本的な仕事はあくまでこれは人間の頭脳で行うというのがたてまえになっておりまして，そのモジュール分けそのものにはやはり非常に豊富な経験が必要である。したがって強い洞察力が必要とされることになってまいります。

完全なシステム設計を行うことがいわゆるモジュール化に当たっての非常に大きな要件で，その上の段階に立ってHIPO化を行なっていくというような形態がございます。このHIPOにつきましても，何をしたいかあるいは何を欲するかというレベルでその持つ機能を書くということがたてまえでありまして，いかにしてその機能を実現するかというようなプロセスを書くのでは決してないということがございます。これらが現実にわれわれが実行しようとする段階で非常に誤解があるところではないかというふうに感じております。ブラケット博士もいわゆるモジュール・レベルでの仕様書を非常に重視しておりまして，この仕様書のコピーからだれがモジュールをつくっても全く同じ形のもの，つまりプラグコンパティブルという形のモジュールがつくれるだけの情報を織り込むようにというアドバイスが与えられております。

それに関連しました形でシュード・コードというのがございますが，ヒューズ・エアクラフトそのものは先ほどのドクター・ブラケットのようにかたいことは一切言っていない。むしろアナリストの性格に任せて，HIPOの段階でいきなりシュード・コードを書いてもいいというようなことをおっしゃっておられる。シュード・コードそのものは厳密に言いますとストラクチャード・コードではなくて，ひら文の箇条書きでストラクチャード・コードの形を取り入れてもよいと言われておりますが，プロセスの記述にやや近いものであると理解しております。プログラミング言語の前に必ずシュード・コードを書かせて，フローチャートのかわりにするという形が基本的にフローチャートレスという言葉で言われているんではないだろうかということでございます。特にこの段階では英語あるいはヨーロッパ関係の国の言葉を話す人々にとってみてこれが一番読みやすいということは何回となくいろいろな会社を訪問した段階で言われておりますが，これは文法のせいだという一つの結論を得て帰ってまいりました。

たとえばジス・イズ・ザ・ブック・ザット・ヒー・ボート・イエスタディー，これは彼がきのう買った本だという文章があったといたしますと，これをデータ定義に見立ててストラクチャード展開をいたしますと，これは本だというのが一つ出てまいります。それは彼が買ったというのが第二段階，きのうというのが第三段階，こういう形の記述がいわゆるシュード・コードで書かれていく。これは日本的に見て非常に感じを感じる形でして，いわゆる言葉の，言語そのものの発生の形態からしてシュード・コードが最適であるとい

うような考え方もある一方，日本流に焼き直した形でのシュード・コード方式というのはこの文法の違いゆえに定義づけそのものに不自然な面がある。この辺から日本の場合にはアウトライン・メソッドというような方式かあるいはカードベース方式を利用するとか，よい方法を考えていくべきではないだろうかということをコードの問題として痛感した次第でございます。

このプログラムの作成につきましては，トップダウン方向の原則ということがたてまえでございます。もっともこれには，こんなものをまず先に完成させていくというカーディスアウトの考え方が要求されております。ステート・アーミー・インシュアランスの保険会社では本社の機構でプログラムをつくりすべての関連会社に配付するというようなやり方をとっておりまして，プログラムの難易度は比較的その面では集中化ということでそろっております。ただし，開発する量が非常に多い場合，これらにつきましてはHIPOまではトップダウンで設計する，ただしプログラムの作成はボトムアップで行うといういわゆるIPT思想とは非常に違った概念でやっておられるということも大きな特徴であろうと思います。

これらに伴ういわゆるプログラマー共通の問題がございまして，これは新しい技術を導入する際の問題でアメリカでも非常に苦労しておられるようです。冒頭に申し上げましたように，いわゆるコンピュータ・アーキテクチャーを希望する人々，これらがまだかなりたくさんおられるということです。この新しい構造化理論に伴うプログラム・テクニックに対しての教育方法としては，まずキーマンを養成するというやり方が第一義でございまして，そのキーマンそのものからチーフ・プログラマー・チームの結成ということにつなげていくような考え方，それと新人をまず先に教育してしまうという形でございます。最も抵抗の強い中間層のプログラマーに技術修得をする必要性が両面から上がってくるような感覚から認識をさせていくというようなサンドイッチ方式の教育という形が非常に各所で行われておるということが非常に目につくのであります。したがって，こういうストラクチャー・デザインの教育の重視ということに対してアナリスト単位では5日間のセミナーを受け，その後である程度の一定期間を置いてさらに5日間の教育をさせる，その教育の中でHIPO，ストラクチャー・コーディング，こういう教育をやるというような体制がとられているのが一般的のようでございます。

この教育と並行しまして，チーム編成の問題が重なってまいりますが，チーフ・プログラマー・チームというようなチームで，公式に大体三通りのやり方があるということがわかりました。

一つは，このプロジェクト・チームを編成して，仕事が終わると解散するというプロジェクト単位のチーム編成の方式。

二番目は，ある程度のメンバーを固定化していく，そのチームそのものに次々とプロジェクトを投入していくというようなやり方。

三番目は，いわゆる設計そのものはいつも設計だけの完全ないわゆる横割りスタイルという形，この三つのパターンがあるようでございます。

これらにつきましても，一つの方式に長く固執することによる結果，欠点が累積するという傾向がありまして，これらの三つのパターンを幾つか変化させるというやり方でやっておるのが最も成功に近いというような報告が多くございました。

問題となりますのは，こういうチーフ・プログラマー・チームとしてのチーフが，ユーザーとのいわゆる技術的なコンタクトあるいはチーム管理あるいは困難な部分のコーディングという形に対して非常にスーパーマン的な役割りを果たさなければいけない。一番の

問題は，そういう適格者がその企業の中に非常に少ないということでございます。したがって，そういうチーム制をしておるという結果非常に大きな効果を上げておられるところ，その反面リスクを非常に大きくしょっておられるというところもあるようです。このチーフ・プログラマー・チームそのものは，H・D・ミルズ氏がIBMにおられた時FSD部門というのがございまして，そこにチーフ・プログラマー・チームを試みたということからスタートしたようでございます。そのFSD部門ではベテラン・プログラマーが管理者になっておって，実際にはプログラムが書けないという立場のところが非常に多い，そこで病院では外科手術については必ず病院の主管プロフェッサーが執刀するというところからこのチーフ・プログラマー・チーム制を思いついた，その結果試行してみた，それがうまくいったということがまずはしりとなっておるということのようでございます。こういうチーフ・プログラマーという形をとったチームの中でも，そのチームとしてやっておられるところあるいはそのレベルに応じてグループ複数制でやっておられるところ等，その企業体制に応じて幾つかのやり方があるのではないだろうか。

そういった評価の中で最も大きな成果を上げたというふうに感じておりましたのがいわゆるライブラリアンの設置であるということでございます。プログラム・ライブラリアンというのは，プログラマーが雑務から解放される，いわゆるプログラマー関係の事務というのがございまして，インターフェイスのコンパイル・リストであるとか，あるいはコントロール・フェース・カードであるとか，あるいはプログラム・シートそのものの登録等ですが，ライブラリアンではチームがオープンになってそれが管理できるという形で，特に主観が入らないという点では非常にその進捗度合い掌握の関係から成果が上がっておるんではないだろうかということでございます。このブラケット博士は必ずしもこのライブラリアンに対する意向としてあまり賛意を示しておられなかったようでありますが，基本的には6人のプログラマーを採用するという形で1チームをつくるよりも，5人のプログラマーに1名のライブラリアンの方がはるかに効率としては上がるということを強調されておったということが非常に耳新しい。さらにそのライブラリアンはアメリカにおいてはほとんど女性が占めておるということも一つの特徴ではないだろうか。

それと，エゴレス・チーム，これはこのIPTの応用として上がってまいりました方式でございますが，中央のサポート部門からいろいろなプロジェクト・チームを派遣するようなやり方，これがライブラリアンの体制の一つとして考えられておりまして，基本的にはチーム全体でコーディングを設計するというのがたてまえでございます。コーディングの設計をチーム・メンバーの前でそのプログラマーが説明をする，それで理解と納得をする，いわゆるチーム全体でそのプログラムに対する解析を行う。だれが書くかということはそれを発表した後別個に決めるというやり方がエゴレス・チームの一つの大きな特徴でございます。プログラマーが自分の書いたプログラムを他人から批判されるということは，基本的に父親が自分の息子から批判されることと同じような感情を持つということでして，その結果自己防衛とかあるいは自閉症という形の形態が生まれてまいりました。そういう結果から技術向上の進歩が妨げられるということになります。その最終的な結果，チームとしてのコミュニケーションも悪くなる。こういう結果になりますので，最近ではそのエゴレス方式というのが非常に活発に活用されておるというのも一つの大きな特徴である。いずれにしましても，これらは社風とか要員の習熟度あるいは主要業務，こういう性格を

吟味した形でその会社，企業に合わせた形でチーム編成が必要ではないだろうかということでございます。

それと，いわゆるウォークスルーとしての見直しということ，これがどの企業でも非常に重視されておる。非公式な仲間内の集まりというようなやり方でのミーティング方式でございまして，つくった方がその仕事を見てもらうという形でお互いに説明をし納得をしてサインをしてもらった上で，サイン後に次のステップに移るというようなやり方でございますが，こういう方式そのものをステップ・バイ・ステップでやっておるというケースと同時に，チームでやらなくても，たとえば自分のつくったプログラムを他人が見て，それでサインをしてゴーを与えるというやり方であれば，それだけで40％以上のミスがカバーできるというようなケースがございます。基本的には本人以外の者に目を通させることによってレビュー行為を正確にさせるという形でございます。

これらのソフトウエアの開発の結果で得られるものは完成したプログラムそのものに関連したドキュメントの整備でございまして，それだけのものを書く労力をいずれにしてもマンパワーを投入して書かざるを得ない。そういったあたりがきのうのセッションで75％とかあるいはアメリカの統計によると45％というような数字であらわれてきておるんではないだろうかというふうに感じておるわけです。こういうソフトウエアの開発業務につきましては，広い理念的な一つの体制と同時にその裏づけが必要である。特に企業内では人間の性質とかあるいは心理ということの組み込みを考えた上でストラクチャード・プログラムというものの体制と取り組んでいく。つまり外面の形態あるいはそういう一つのマニュアルの形態にとらわれないで，企業なりに消化していくことがこの新しい技法に対応するチャレンジとして最も重要なことではないだろうかというふうに痛感をして帰ってきたわけでございます。

以上で報告を終わらせていただきます。

森口　プログラムの信頼性が10倍になる，そして生産費は半分になるというようなことを聞くとなかなかいいなあと思うわけです。しかしいろいろウォークスルーですとか，ウォール・プリントですとか，何々レスというような言葉がたくさん出てきますと，ああそれならやれそうだという気がするうまい話をたくさんしてくださったわけです。それにしてもダイクストラさんが「ストラクチャード・プログラミング」という言葉の特許を取っておられたらずいぶん使用料がかせげたんじゃないかと思われるような節もあります。

ところで，次の水野さんは大ぜいの職人を抱えたギルドの親玉とみずから称しておられます。どうぞ。

水野　いま御紹介がありましたように，私は大ぜいのギルドを抱えているあるいはギルドにしてしまった1人かもしれませんが，現在大型のオペレーティング・システムの開発を行っております。それで，きょうのダイクストラさんのお話の中で，今後のプログラムとして正しいプログラム，そしてエフィシェントなプログラム，そういったお話があったわけでございますが，これらについて私の考えを若干述べさせていただきたいと思います。

まず，よいプログラムというのはどういう条件を備えているかということですが，これはまずコレクトネス，それから保守，それから修理，修正がやさしい，それから明快であること，それから低コストでつくっていかなければならないこと，移行しやすいこと，それから人間に関連するいろんな要素を考えて使いやすくなくてはいけない。それから最後に，ＯＳその他基本的なソフトウエアとして要求されてきていることは，いわゆるＯＳの第一段階の機能としてリソースのマネジメントのレベルから，ソフトウエアのユーザーズ

・プログラムあるいはデペンデント・プログラムの開発が容易であるという領域から，さらに最近では壊そうとしても壊れないトレーディング・システム，そういったことが一つのよい条件であると思います。もっともこれは非常に感覚的な話で絶対的な判定基準をつけることはむずかしい問題でございますが，もしよいプログラムをそういうような条件で考えていきましたときに今後やっていかなければならない問題としては，よいプログラム構造を持たせるためのメソドロジーで，これはきょうダイクストラ先生が話された領域の問題でございます。

第二番目は，ギルドがサイエンティストの組合といいますか，クラフトからサイエンスの領域に入っていくためには，一つの考え方として，これも過去の生産工程の発展過程と対比して考えてみますと，家内工業時代，それからマニュファクチャー時代，それらの時代においてはツールがなかった。つまりソフトウエアの場合にもソフトウエアを開発するツールのもっといいものを私のギルド・グループに与えることができたならば，ギルドからエンジニアリングの領域へ進められる一歩になるのではないだろうか。表現しやすい，特によいプログラムが満足すべき性質あるいは構造を表現しやすい言語，それからディバッグ・テストあるいはドキュメンテーション，そういったものの支援ツールを開発していくことがギルド・グループから少くともエンジニアリング・グループに発展させるための必要条件じゃないかと思っております。

それから，一つの大きなＯＳを開発していくときにいわゆるブラックボックス・ユーセージといいますかパッケージ化，これの考え方を導入し，ソフトウエアの開発費用，それから信頼性，この両面を上げていく。

それから，フォーマル・ディスクリプションの問題でございますが，プログラマーは隣の席にいてもその間のコミニュケーションがうまくいかない。それから，隣同士で話が通じているかと思うと，最後のインテグレーション・フェーズにおいてインターフェース・ミスが大きくなってきている。情報のエクスチェンジでのあいまいさや誤解，そういったもののエラーの発生を防ぐためにフォーマル・ディスクリプションの問題を取り入れていく必要があるのではないかと考えております。

それから，テスト，ベリフィケーション，こういった問題についてですが，実は私のソフトウエア開発部門の中に検査部というのがございまして，コントロールではなくてクォリティー・エバリュエーションをやっております。そのグループがクォリティー・エバリュエーションするのには，湯水のように電算機を使って，それで24時間テストしていざ出していくとバッグが多い。このことから違ったディビジョンででき上がったものをテストするということの不経済さを非常に常々感じておりまして，新しい構造のプログラミングに対してどういう有効なテスト方法があるか，これが重要なんじゃないかと思います。

それで，こういったフォーマルなディスプリン，これは非常に理想的なお考えですし，われわれの将来の方向を示唆する方向だろうと思いますが，現実にはまだまだ多くのギルドがいる。それで時間の連続性の点からこれをどういうふうに改善していくかということが重要で，あしたからすぐ精神革命が起こって，あるいは技術的なテクノロジーが一遍によくなっていくというわけにもいかないと思います。

それで，ここで一つおもしろい経験をお話しいたしますが，現在存在しているプログラマー，これをギルド・グループとしますと，そのグループにこういった新しいソフトウエアのエンジニアリングの話をしましても浸透スピードが非常におそいのです。いわゆる新入社員，新しく大学のサイエンスあるいはエンジニアリングのコースをとってきて入って

きた人にソフトウエア・エンジニアリングの教育をしますと非常に早くそれになじんでくる。ですから，ギルド組合のメンバーになってしまう前に早くこの新しい方式を教え，それをツールとして活用していけるようにしていく必要があるんじゃないかと思います。

以上，簡単でございますが私の意見といたしたいと思います。

森口　やはりその近代化といいますか科学への道を歩むためには道具が大切だというポイントを中心にしていろいろな経験に基づく意見を述べられたわけです。

ここでちょっと合いの手として，ダイクストラさんに関することを二〜三御紹介申し上げます。

先ほどの休憩時間に木村泉さんから名前の読み方をぜひ御本人に伺ってくれと言われましたので，その答えをまず申し上げます。

ここにかたかなで書いてありますこの読み方は大体よろしいのですけれども，困難ですができるだけオランダ語の発音に近づけるとしますと，まずエドガーの2番目のトにてんてんのかわりにツにてんてんを打つとよろしいと思います。これはもちろん新かなづかいではスにてんてんになりますが，それではぶち壊しになりますからツにてんてん。それから，ガーの音引きの棒をルにかえていただく，そうすると近くなると思います。なお，ダイはダイとディの間くらいらしいのですが，むしろこのままにしておいた方がいいかと思います。それでできるだけうまく読んでみますが，後でひょっとすると御本人から訂正されるかもしれません。エツガル・W・ダイクストラ。（笑）

第二点は，いま主としてどこで時間を過ごしておられるかということ，これは夕べ伺ったのですが，1週間に1回アイントホーベン工科大学で講義をするそうであります。それ以外の日はバローズ社のオランダ本部に勤務しておられるそうでありまして，それは自宅であるということでございました。

その自宅の所在地は，きのういただいた名刺によりますと，ニューネンという名前の村であります。これは日本式に言えばアイントホーベン市の郊外にあるわけでして，特にアイントホーベン工科大学までは自転車で20分の距離という話であります。自転車はちなみにオランダでは非常にはやっていて，1人当たりの自転車の台数は1よりちょっと多いらしいです。なぜそういうことがあるのかと伺ってみましたら，御本人の家庭は5人家族だそうですが自転車が6台あって，それは5人が1台ずつ自分の自転車を持っているほかにタンデムの自転車が1台あるんだそうです。

まあその辺のところがダイクストラさんの紹介の追加でございます。

さて，これからいままでに挙がった問題点についてダイクストラさんの御意見を伺いたいわけです。問題はいろいろ挙がっていると思いますが，大別しますと，まずは職人方式から科学的な方式に切りかえるに当たっていろいろな道具が必要ではないか，その道具としてはどんなものがあり得るかというような方向。

第二は，人間の教育の問題が大変重要な問題として浮かび上がっております。既存の人たちは果たしてこういう転換に向けて教育できるのかどうか。いずれにしても新人を正しく教育することが一番肝心であるということについては全く異論がないようでありますが，それについてもその方法その他いろいろな問題があろうかと思います。

それではダイクストラさん，どうぞ。

ダイクストラ　家族における自転車の数の問題は，犬が2匹いるのを申し上げるのを忘れまして，それを足しますとちょっとパーセンテージが変わってくるかと思います。

さて，少しこの席をかりまして科学的理論の点について「ふえん」をしたいと思います。誤解を避けるためにそうしたいと思います。

私はフォーマル・カルキュラスという言葉を使ってプログラムのデリベーションにしようとしたわけであります。それがオートマティズムもしくはアルゴリズムになるということを意図したわけではありません。そして，それは大量の発明を除外するものではなくて発明を必要とするものであります。そして，フォーマルな手法，テクニックを適切に応用いたしますと，その結果としてプログラムがスペックに合うという保証を得ることができます。先ほど画家の例を引いたわけでありますけれども，まさにそういった例はいまでも適応できるわけでありまして，デューラーによって遠近法の法則が確立されて，新しい画家たちはその法則を知っていたけれども芸術の仕事はなくならなかったわけであります。そして，知的な人々はその画法によって画家を知ることができたわけでありまして，全く同じことが美しいプログラムについても言えるわけであります。多くの同僚を私は持っておりますけれども，プログラムにたとえサインをつけなくても，プログラムを見ればこれはだれのプログラムだということはよくわかるわけでありまして，これはたとえば数学の教科書を見ますと，初めて読んだだけでもこの教科書はだれの本であるかということがすぐわかるのと全く同じであります。事実，フォーマルなテクニックの開発の終わりとしてその最終的な目標は一体どの程度までプログラムの活動を削減してフォーマルなルーチン化をすることができるかということを知ることであります。そして，どの程度までオリジナリティーとかいい趣味とか発明とか，そんな種類のものが必要になるかということを知ることだと思います。

ツール，道具の問題が指摘されました。プログラミングをサポートするための道具という問題が出たわけでありますが，数学者は一体どんな道具を持っているわけでしょうか。いい頭と鉛筆と紙だけあればいいんじゃないでしょうか。一体どういう道具を使って音楽の作曲家は作曲するでしょうか。これもまた鉛筆と紙とすぐれた音楽的な能力，音感といったようなものでも持っていればいいと思います。

提起された問題の中に，普通のごく平凡なプログラマーをどうやったら教育できるかという問題があったわけでありますけれども，その質問に対する答えをしてみたいと思います。そのために逆に質問を問い返したいと思います。それでは本当に普通の数学者はどうやったら教育できるかという質問で答えにかえてみたいのです。こういうことを言っておりますが，私はまじめなんです。私たちは，どうやればいいかということはわかっていると思います。また，同時にあまりよくない数学者を教育しようとするのは全くのむだであるということも十分承知しているわけで，これは教師にとってまた生徒にとっても不快な作業以外の何ものでもないと思います。また，すぐれた数学者を教育訓練するということは全く喜びになるわけであります。ですから，普通のプログラマーをどうやって教育したらいいかという質問をされますと，私はその平均的な，普通のという言葉を抜かして考えたいわけでありまして，プログラマーをどうやって教育訓練するかというのは数学者を教育するのと全く同じだというふうに考えております。少なくとも3000年の経験を通してこういうことが言えるような気がするわけでございます。

次に，多くのパネリストの方が，形はいろいろ違いましたけれども，少なくとも英語に訳してヒューマン・ファクターという言葉が出たわけでありまして，ヒューマン・ファクターというのが日本では非常に人気を集めているという気がしたわけでございます。

さて，11年前だったと思いますけれども，私はニューヨークで講演をしたわけでございます。「人間活動としてのプログラミング」

という講演をＩＦＩＰでしたわけでございます。２年前にもまたもう一つの講演をいたしましたけれども，これは「プログラミングの教育」つまり「思考の教育」という題で講演をしたわけでございます。

人間活動についてはこのぐらいにしておきまして，もう一つのポイントは，いろいろな方々のコメントの中で，メカニズムが確立することと，それから要求に合わせてそのメカニズムをデザインすることの間の明確な弁別が必ずしもできていなかったような気がするわけであります。分析のフェーズとアナリシスのフェーズを経てコーディングに行くということを言いまして，それが努力の 75 %ぐらいを占めるというふうに考えているわけであります。この75%の一部は明らかに正確にどういうリクワイアメントがあるかということを発見し，そしてそれをフォーミュレートする，つまり方式化，形式化する必要があるわけでありまして，デザイナーの満足のいくようなものを見出していかなければいけないわけであります。

きょうの午後もこういうコメントが出たと思うのですけれども，プログラミングのむずかしさの一つは，スペシフィケーションが非常にあいまいであるというところに起因するといったことが指摘されました。それを額面どおり受け取りますと，そのコメントは全くナンセンスであります。というのは，そのスペックがあいまいであればあるほどそれを満足させることが簡単であるからであります。もしそのスペシフィケーションが全く空疎なものであれば完全にすべてを満足できるわけでありまして，またスペックが矛盾していれば簡単に満足できます。どうせ矛盾しているんだったならば何にも満足させることができないのですからプログラムをつくらなければいいわけでありまして，その要件に合わせるのはごく簡単なわけで大した問題ではありません。プログラムが大変だというときは，そのスペックが非常に厳しく明確に書かれていて，そして全く矛盾を含んでいないというような場合なのであります。

普通のユーザーとか普通のプログラマーの話はいたしませんけれども，普通のカスタマーの場合はその考え方，話し方，また物の書き方におきまして，コンピュータが必要とするような正確さになれていないわけであります。ですから，プログラマーがほかの人にサービスをするに当たって気がつくと思うのですけれども，何日間，何週間，またときには何カ月間かかってやっとカスタマーに話をして，その結果としてカスタマーが一体何を求めているかがやっとわかるわけであります。そして，プログラマーとカスタマーの関係を最もよくするためには，プログラマー自身がスペックが完全に明確であって，そしてカスタマーに対して３～４回もしくは５回ぐらい説明して，実際にカスタマーの求めることがこれだということがわかったときしかプログラムにはしないということにすることがいいと思います。そういう観点から申しますと，私は何回もこのことを言っているのですけれども，そしていろんな方々からそのコメントが確認されておりますけれども，最も能力のあるプログラマーの重要な資産というのは，母国語もしくはできればもう少し数カ国語を完全にマスターしていることでありまして，母国語を例外的と言ってよいほど完全に理解をするということが絶対必要条件であると考えております。

それから次に，ＩＰＴ，ＨＩＰＯ，デボ，ミルズ，ラリー・コンスタンチン等についていろいろな話が出ました。私の印象では，これらのコメントはマネジャーのしたものであるという気がいたします。マネジャーとの関係で私が問題に感じるのは，彼らは決してあいまいであることと抽象的であることの基本的な違いを理解しようとしないということであります。抽象ということは新しい理論のレ

ベルであって，完全に正確さを意味するものであるということを彼らは理解しないようであります。私が了解します限り，IPT，HIPO及びその他というのは，あまりそれについて云々しない方がいいと私は考えます。これらのドキュメントを読んでみましたが，彼らは非常に皮相的なことしか言っていないと思います。まじめにモジュールに対するリクワイアメントがあるということであれば，少なくともたとえば50のプログラム・ステップであるべきであるとか1ページであるべきであるとかというふうに言っているわけですけれども，アメリカ人がそういうことをやっておるのは知っておりますけれども，そういうことを真剣に提案するということは完全に自分の目前にあるタスクというのを誤解しているということだと思いますし，これは全くばかげたことでありまして，数学的な証明は1ページ以内におさめるべしというのと全く同じ程度のばかさかげんだと思います。そして，すべての彫刻は彫刻家によってつくられるべきであって5ポンド以上になってはいけないというようなばかげたステートメントと全く同じだと思います。

最も恐るべきトラブルとして考えられますのは，たとえばミルズとかディボルトとかコンスタンチンのような活動は，あたかも彼らが引き続きこういう皮相な物の考え方，話し方で十分だというような主張をしているわけですけれども，実際はそうではないわけでありまして，これはあたかも詐欺のようなものだという気がするわけであります。それらの問題の幅があまりにも大きいわけでありまして，そしてまた，IBMがつくっているような非常にスロッピーな言語を使ってやろうとしているわけであります。換言いたしますと，たとえば鈍いおので鉛筆を削ろうとすると，たとえ10本の鈍いおのを持ってきてすら1本の鉛筆を鋭くとがらせることはできないということが言えると思います。

はじめはこのくらいでいいと思います。

森口　さて，パネルの皆さんから何かただいまのダイクストラさんのコメントに対して御意見のある方はいらっしゃいませんか。

上条　御意見はほとんど尽くされておるような感じでございます。

最初に申し上げましたように，私は決して立場を違えるものでもございませんので特に反論というようなことは考えておりません。ただツールに関しましては数学の証明をするのにツールが必要であるかどうかという議論に持っていかれてしまったのですが，アクションからスタートしていけばこれは大変な手数がかかるわけでございます。それに対してセオレムがあればそれから先はわりあい簡単になるというような逆の引用をさせていただきますと，逆例証をさせていただきますと，そういうタイプのセオレムに対応するようなツールはあってもいいんじゃないかというふうに思うわけです。

それからもう一つ，教育の問題，特にリトレーニングの問題に関しましても，リトレーニングを心底からやるというのには限界があるわけでございますし，また必ずしも許される話だとは思いません。むしろこれも何らかのツール的なものを使いまして，ある意味でその人の思考の範囲を規則化することによって疑似的にサイエンティフィック・ディシプリンの中に封じ込めることができるんじゃないかなというような感じも持つわけでございます。

以上，その二点を反論というわけではございませんけれども申し上げたかったわけでございます。

森口　ほかにどなたか……

岸田　私もいまのツールに関してちょっと上条さんの意見の補足を申し上げたいと思いますが，確かにダイクストラさんの言うように，真の意味でプログラマーである人々にとっては，数学者にとって紙と鉛筆だけあれば

十分なように，やはり同じような単純なツールだけで十分であるだろうと思います。けれども，現在のいわゆる社会的なソシアルな問題として，そういったプログラマーとしての真の有能性を持っていないかなり多数の人々がコンピュータにアクセスしてソフトウエアの制作に関係しているということがあります。そういったアマチュアあるいはセミプロフェショナルな人々のために，彼らの仕事をコンピュータ側でアシストするようなサポート・ツールというものは，理論的にではなくて社会的にかなりの存在意義があるというふうに私は考えます。もっともその辺は恐らくダイクストラさんの関心の外にあるだろうと思いますが。

それから，プログラマーの教育について非常に明快な御意見をいただきましたが，それは確かにそのとおりでありまして，私自身も過去数年間プログラミング・メソドロジーについて若いプログラマーの教育にタッチしてまいりましたけれども，そうした教育の効果というのは，プログラマーになれる人間となれない人間をふるい分けるということではないかと思います。したがって，こういうことを言ってしまっていいのかどうかわかりませんけれども，人間が教育を受けてプログラマーになるというよりは，本来生まれつきプログラマーになれない人間とが分かれているんじゃないかというふうな感じさえ私は持っております。そういった意味では，いわゆるプログラマーでない人間をプログラマーとして教育するのは非常にむだであるという御意見は確かにそのとおりだと思いますが，これもやはり社会的な側面としてはそういった人間も現実にプログラミングの仕事に携っている以上，彼らが何らかの形でプログラミングの仕事をスムーズにできるような形での教育方法というものが考えられるべきであろうというふうに私は思います。

藤村　先ほどのお話の中で2点，一つはツールの関係がございましたけれども，いわゆるセッションという一つの段階でのツールをソシアルな面で実際運用をしていくといった場合の一つのお答えとは基本的な形，背景から違いがあるんではないだろうかというふうに思います。

それと，教育ということについては，基本的には紙と鉛筆ということがあたりまえでございます。それから，国情によっても違ってまいりますし，当然それに対するスタンダードな過程があってしかるべきだろう。それらについて，詳細なお答えをいただけないことは，これはセッションの段階では当然でむしろわれわれがその背景を受けとめてやるべきではないだろうかということが私の感じておるポイントでございます。

それから，最後に，ラリー・コンスタンチンとかいろいろなお話がございましたときに反対御意見がございましたけれども，実際にそれらが成果を上げておるということを大きな一つの正当な評価として受けとめて，それらの背景とやり方とをこの思想とつなげた形でいわゆる技術開発ということを進めていきたいというふうに考えております。

水野　一つはツールの問題でございますが，ツールを与えることによってより効果的な生産性が向上でき，正しいエフィシェントなプログラムができる，そういったことは岸田さんからも御指摘ありましたように，現実の問題として多くのプログラマーと名づけられている人がいる場合に，これらの人がすべて適性を持っているかどうか，数学者としてでもすべてそれが本当の数学者ばかりというわけではなく，いろいろな目的に応じて社会的な活動をしているわけですが，そういうエグジステンシーも存在している立場からは，やはりツールというものが重要なのじゃないかと私も思います。

以上です。

森口　どうでしょうダイクストラさん，会

場からの質問を直接受けてよろしいですか，それとも何かこの時点でコメントありますか。

ダイクストラ　ちょっと短いコメントをしたいと思います。

科学的な問題と社会的な問題を混同するべきではないと思います。もちろん両者とも存在していることは認めますけれども，それを同時に取り扱うのは避けようではないですか。もちろん何千，何万という人がプログラミングという職業がむずかし過ぎるにもかかわらず魅惑を受けてその職業につこうとしてきたわけであります。これはある意味では非常に悲劇的なことでありますがただ私の言えることはそれは私の責任ではないということです。別に私がそうしたわけではないのですから。多分時間だけが問題を解決してくれると思います。というのは，そうした人たちは永久に生き続けるわけではありまんから。

森口　それではフロアからいろいろな御意見，御質問を出していただくことをダイクストラ先生も望んでおられますので，どうぞお願いします。マイクが行くまで待っていただきたいのですが，どうぞどなたか。

質問　外部仕様の決め方についてはすでにお話がございましたけれども，もう一声伺いたいと思います。

お書きになった御本など拝見いたしますと，一応スペシフィケーションができた上でプログラムを組み立てていくことについて大変おもしろいお話がたくさん出ている。そしてまた，きょうもそういうお話があったんだと思います。ところで実は外部仕様というものは決めるまでに現実の問題がぼやっとありまして，たとえばオペレーティング・システム一つつくるというようなときでもそうだと思いますが，その外部仕様をどこでどういうふうに線を引いて切り出してくるか，そこの選び方がうまくいくかまずくいくかでプログラミングのむずかしさは大変違うものだと思っております。そのために設計上の意思決定というようなものをいっぱいしなければならない。ところで，10年ほど昔にお書きになりましたザ・マルチプログラミング・システムの論文など拝見いたしますと，そこのところを非常にすばらしく上手に切り出していらっしゃると思います。

それで，問題をわりにはっきりさせるためにこういう形で御質問してみたいのですが，大学の環境の中でそういう問題の切り出し方という点についてお弟子さんを教育なさるとしたらどういう教育の仕方をなさるか，どういう例題をお使いになるか，そういう点について一言伺いたいと思います。

ダイクストラ　そうですね，最初にまず彼らに対して10分ほどスピーチをして，なぜ私がプログラミングというのが数学の中でも最もむずかしい分野であるか，というのはこれがエンジニアリングの中で最もむずかしい分野であるからというふうに説明いたします。プログラミングは非常にむずかしいということを10分言った後で，次の1時間使いましてカード・ボード・ゲームというものについて話をいたします。ユークリッド幾何学によるところの数学的計算の問題について触れるでありましょう。ユークリッドの数学というものは結局のところは，そうですね，簡単なものに関して4時間ぐらいかけて話をいたします。ユークリッドのアルゴリズムを使ってノンデタミナンシーの問題を指摘することもできましょうし，またアルゴリズムを使って数字の役割りとバリアント・ファンクションとの関係の問題について，また同時にターミネーション・プルーフをプログラムによって，いろいろ別個のものを使うことによってどういうふうな結果を得ることができるかというふうに使いますし。またこれを使ってユニバリアント・リレーションというものをリビティタティブなリレーションに関して見つける際のホーリスティックスについて触れるというふうなことをすると思います。

例を使う場合には非常に慎重に選択したものを使います。しかも非常にシンプルなものを前半は使うでしょう。できたらそういう方法は私としてはやりたくないのであります。つまり問題というのは，もし例を示しませんと学生は一体何の話か全くわからないということになります。しかし例を示しますと学生というのはそれが単なる例でしかないということを忘れてしまうのです。ここが問題です。たとえば最大の例というのは，クラフトのサブストゥーミングトゥリーというものを考えるのに使いまして，あまり頭のよくないばかな学生の一人がそれを聞きまして，クラフト理論というものがコンピュータ科学の最も主要な問題点であるとどういうわけか誤解してしまったくらいであります。同じような経験を，私は講義をしばらく前にしたことがありますが，その講義の中でフォーマル・トリートメントというものと大型のプログラムに関してのフォーマル・トリートメントの経験をお話ししたのでありますけれども，リカーシブ・パーサーの問題についても話をしたわけですが，しかし聴衆の一部はパーシングについて私が話していたんだというふうに考えた人がいるわけであります。

お答えになりましたでしょうか。

質問　私は民間の計算機を使う企業で働いているものですが，先ほどのスペシフィケーション，いわゆる外部スペシフィケーションという言い方をされた方もおられましたが，その問題についてもう一度コメントをいただきたいのですが，ダイクストラ先生の答えはとにかくスペックがちゃんとできるまでプログラミングにかからないことだということであったのですが，実はスペシフィケーションをつくるという作業自身も何らかのステップ・ブライズ・リファインメントといいますかそういうアプローチを必要とする作業なのではないかと思うわけで，したがって，プログラムの機能の定義といいますかスペシフィケーションをつくる作業と，それからそれを実現するための作業，それはどちらもステップ・ブライズ・リファインメントというようなアプローチが必要だとして，それがうまく一致しないというところに非常にスペシフィケーションをつくるのが大変だというふうに騒いでいる原因があるのではないかと思います。

先ほどのそのスペックができるまでプログラムにかからないことが賢明だというようなお話について一つ私考えることですが，たとえばある種のプログラムでエラー・チェックをしようというときに，プログラムを書いていて初めてこのフラグがこうなっていたらどこへ行ったらいいのか，飛び先がないからということで初めてどういうエラー処理をすべきかということに気づくかを考えなければいけないということに気づくことが，私自身こういうプログミング・スタイルということを学ぶ前にはあったわけです。それは初めのうちは私もスペシフィケーションを完全に決めてなかったのが悪いんだというふうに考えていたわけですけれども，どうも最近そういうことが事の本質なのではないか。やはりスペシフィケーション自身もだんだんとしかはっきりしてこないものではないのか。プログラミング作業といいますかプログラムのロジックを決めていく作業とともにスペックも決まってくるというのが理想なのではないかという気が一方ではするのですが，その点についてコメントをいただけたら幸いに存じます。

もう一つ，先ほど教育の話が出ていますけれども，私もACMに載った有名なGO TOレス・ステートメント云々というコレスポンデンスを見たときよりも，やはりもう少しやや詳しい本を読んだときの方が少しはわかったような気がしたわけなんですけれども，そういう意味でたとえばハネウエル・ブルのブラニエさんがとっておられる方法あるいはアメリカのACM，計算機学会でマイケル・ジャクソンという方がやっておられるというプ

ログラム設計法のコース，そういうような教育法といいますか，それは特定の事務処理，いわゆるEDPSの世界を対象としたいわば特殊な分野について有効な方法論であるかもしれませんけれども，そういうような教育法といいますか方法論についてどのようなお考えをお持ちか，もし関心がおありでしたらコメントをいただきたいと思います。

ダイクストラ　ハネウエルのケースについて私は存じません。マイケル・ジャクソンについては何回か指摘されるのを聞いたことがあります。マイケル・ジャクソンが教えていることは，結局のところシークエンシャル・ファイルをCOBOLでいかにプロセスするかということになると思います。マイケル・ジャクソンの講義を1回だけ聞いたことがありますが，なかなかいいものだったと思います。彼の言ったことは全部よかったと思いますし，COBOLのプログラムはCOBOLを使ってファイルを処理しなければいけないものもたくさんあるわけでありますから，彼の講義が役に立つマーケットはたくさんあると思います。まあオーケーという感じです。彼は非常に注意深く説明をしているものがありますけれども，それは平均的なプログラマーにとって受け入れ可能なターミノロジーを説明しております。普通のレギューなエクスプレッションの説明をしているわけで，それは結局はディーという形で示しておりまして，いかにしてよいプログラムというのはシークエンシャルなプログラムのストラクチャーにおいてインプット・ファイルのシンタックスをいかにしてリフレクトするかというようなことを言っているわけでありますから，COBOLに関して，ファイルのプロセスに関して興味がある場合には御参考になれば非常によいものだと思います。

それから，その前におっしゃった外部仕様の問題でありますけれども，時間の75％が過ぎてしまわない限り，つまり時間が75％たつとやっとそのときになってだんだんとデザインを始めるわけであり，また仕様を決定する段階がやっと訪れるわけであります。つまり75％の時間が経過した時点でデザインが完了した段階になるわけでありますけれども，コーディングというのはできるだけ長く延ばしておかなければいけません。できるだけ遅くまで延期しておかなければいけないわけで，私自身の経験では，もしプログラムを開発し，それに関してエッセーを書くとしたならば，つまりどういう形でプログラムをデベロップし，なぜそうしたかということについて私がエッセーを書くとしたら，コレクトネスのバリデーションと，それからリジェクトのルタナティブとか，そういったことについて触れることになりましょう。そういったエッセーというのは実際に生のコードでコンピネートされるものの10倍ぐらいになると思います。これがどうも自然の法則であるようでありまして，十分にアノテートされたプログラム，つまりアノテーションというのがプログラム自体の9倍とか10倍ぐらいになるのはどうも自然の法則のようであります。つまりこれが意味するのは，使用されているプログラミング言語というものが非常にコンパクトな形で最終的なプログラムというものを説明できないということでありまして，つまり最終的なロー・プログラムというものが知的な活動を非常にコンパクトな形でまとめているということになるわけです。

お答えになりましたでしょうか。

森口　他かに御質問，御意見がないでしょうか。時間はまだ数分あります。

質問　さっき上条先生と岸田先生が言われた社会問題云々というのはどうもはっきり答えがわからなかったのですけれども，やはりアベレージの人間を使って，実際にあそこにも書いてありますように「情報化週間」とかなんとか，日本は情報化するらしいのですけれども，そういう環境においてやはり非常に

優秀な数学者だけの世界ではやっていけないと思うのですけれども，そこら辺が先ほどのダイクストラ先生の答えではちょっと私には明快にわからなかったのでもう1回お答え願いたいということと，それから藤村さんがおっしゃったIBMのIPT，それがとにかく成果を上げているらしい。私は本当のことを知りませんけれども，それに対してもコメントがなかったように思うのですけれども，それももう一回聞きたいのです。

以上です。

ダイクストラ　成功した例を私は知りませんとまず申し上げた上で申しますけれども，つまりプログラムにおけるドキュメンテーションをプロモートした結果，アメリカでそれほど進歩が見られたというふうにも思いませんし，コンピュータ科学に関しましてはアメリカは後進国であるということを忘れてはいけません。アメリカにおいては改善の余地は非常にたくさんあると私は思います。若干の改善はなされたようでありますけれども，比較をしてしまうといけないかもしれませんけれども，数学者であればだれもセオレムなくしてプルーフをドキュメンテーションしようとは思わないでありましょう。HIPO，インプット・プロセス，アウトプットというふうな言い方をしますけれども，これをなし遂げたのは，もしモジュールがあった場合には，マネージャブルな形で与えられたスペシフィケーションをモジュール化できるということ以外の何ものも意味できないわけで，それが上に出てくるかどうか，ショアアップするかどうかということはささいな問題なのでありますけれども，なぜかアメリカにおきましてはアクロにも出まして，コンピュータ科学にもう一つのフォー・レター・ワードという言葉は悪い意味もありますので通訳に失礼かもしれませんが，おわかりいただけましたでしょうか。

森口　それでは時間が来ましたのでこの辺でまとめにしたいと思います。

ずいぶんいろいろな話題が論じられたので一口にまとめるのは大変むずかしいですが，私が感じたところでは，一つの確信的な考えというものを正しく理解することはなかなかむずかしいことだということであります。実際これを理解したと思って，そしてそれがまた有用な行動の根拠になったからと言って，それですっかりわかったんだというふうに考えるのはどうも表面的過ぎるようであります。だから，もちろんそれぞれの理解において行動に移すことはきわめて好ましいことでありますが，しかしまだまだ考えるべきこと，努力すべきことがあるんだということを忘れてはならない，それが大体きょうのダイクストラさんのお話の気分であったと思います。

それで，私はわが国が後進国と言われたアメリカの後をさらにもう数年おくれてやっていくべきであるのか，あるいはこの辺でそろそろそういう姿勢を改めて，幸いにしてまだプログラミング人口等においてやや害が少ない状況にあるということ，それから大変国民がみんなまじめであって，そして一生懸命数学の教育もよく受けているというようなことを考えますと，わが国はこの辺でこの革新を推進する先頭に立つ国の1つになることも望めないわけでもなさそうではないかという気がいたします。それについて非常に大事なことは，何かこういうストラクチャード・プログラミングというような言葉の一面だけを理解して皮相な解釈に陥って，そこで進歩がとまってしまうということがないようにすることではないかと思います。

そういう非常に有益な刺激を与えてくださったダイクストラさん，それからそのダイクストラさんからまた痛烈な言葉を引き出す機縁を提供してくださったパネルのメンバーにお礼の拍手を贈りたいと思います。

ま　と　め

ま　と　め

斎　藤　　有[1)]

私，副会長の斎藤でございます。きのうから2日間にわたって催しましたこの国際シンポジウムも，おかげさまで予定のプログラムを順調に消化しまして，最後に私が「まとめ」という予定になっておりますけれども，皆様けさから大変お疲れのことと思いますし，時間もございませんので，それに大体私にはこの性格の違う三つのセッションをまとめる能力もあまりないので，主催者としてのお礼と今後の取り組みについてのお願いをかねまして閉会のごあいさつを簡単に申し上げて，このシンポジウムを終わらせていただきたいと存じます。

さて，昨日の植村会長の「あいさつ」と，それから稲葉講師の「基調講演」でも触れられましたが，皆様御承知のとおり，わが国の経済はいままでの高度成長から安定成長へ転換しつつありますし，産業構造につきましても，いわゆる知識集約化，集約型とかあるいは高付加価値型というのに内容の転換を求められているわけであります。このような条件の変化に伴いまして，わが国の情報処理分野においても多くの反省が試みられておりますし，たくさんの課題を抱えておることは，これも皆様十分御承知のことと存じます。

今回のシンポジウムにおきましては，これらの背景のもとに三つの課題，すなわちきのうからのコンピュータ・システムのオーディティングの問題，それから情報の分散処理の問題，それからただいまのプログラミングの問題，この3つの問題をこのシンポジウムで取り上げました理由は，このようないまさっき申しましたような背景で，わが国の情報処理でこれらの問題が非常に緊急な重要な課題であるという認識に立ってこれらのテーマが取り上げられている次第でございます。

幸いにしまして，アメリカとオランダからそれぞれの問題についてパイオニアでいらっしゃいます，それから国際的にも大きなリーダーシップをとっていらっしゃるお三方をお迎えできましたことは，主催者として非常に幸いに存ずる次第であります。お三方ともそれぞれの分野できわめて興味深い貴重な内容のお話を承り，また国内のこれらの道の先導的な役割りを果たしておられるパネラーの皆様には熱心に御討議をいただきまして，皆様も得るところが多かっただろうと思う次第であります。私どもはこれらのディスカッションの成果をもとにして，今後どう取り組んでいくかというようなことを検討しなければならぬと思っておる次第でございます。

はるばるアメリカとオランダからいらしてくだすったお三方に対して心からのお礼を申し上げますとともに，お忙しい最中にパネルの討論会に御参加いただきましたパネラーの皆様にも厚く感謝の意を表したいと存じます。なお，フロアの皆様には2日間にわたってきわめて熱心にこの討論会に御参加いただきま

1）日本情報処理開発協会副会長

して，これまた主催者として心から御礼を申し上げる次第でございます。願わくばこれらの二日間にわたる討論の成果が将来のわが国の情報化の発展に幾らかでも役立つことを私どもは念願している次第でございます。

なお，当情報処理開発協会といたしましては，今後どういうふうにこれらの問題に取り組んでいくかということがまたわれわれにとっての課題であるわけでありますが，情報処理開発協会はいろいろ情報処理に関する調査研究とか開発とか，あるいは要員の養成とか，いろんな問題と取り組んでいるわけでありますが，今後も皆様の御指導，御協力を得まして，これからこの三つの問題にいかに取り組んでいくかというようなことを真剣に検討してまいりたいと存ずる次第でございます。

予定のプログラム全部を終わるに当たりまして重ねてお礼を申し上げまして，本日のシンポジウムを終わらせていただきます。ありがとうございました。

この会議録は当日の速記録をもとにとりまとめたものであります。

昭和５２年　３　月発行

発行所　財団法人　日本情報処理開発協会
東京都港区芝公園３－５－８
機械振興会館内
ＴＥＬ　(434) 8211 (代表)

印刷所　㈱イフ・アドバタイジング
ＴＥＬ　(262) 2984