SEAMAIL

Monthly Newsletter from
Software Engineers Association

Volume 2, Number 7 July 1987

目　次

編集部から		1
春のセミナー・ウィーク報告・その2		2
電子出版技術の動向	大野俊治／田中慎一郎	2
AIパラダイムとソフトウェア開発	大木幹雄／村井　進	6
新形態プログラミング	二木厚吉／中川　中	10
創造的開発のための管理	土井利忠／野辺良一	12
プログラミング能力の定量的評価	鳥居宏次／芝原雄二	18
第9回ICSE特集		21
ICSEの感想	鳥居　宏次	21
9th ICSEに参加して	玉井　哲雄	22
ICSE印象記	落水　浩一郎	26
論文発表を終えて	野村　敏次	29
西海岸ＣＡＩ事情	寺嶋　祐一	31
ＣＡＩ分科会活動報告	大木　幹雄	33
名古屋支部	岩田　康	35
会員状況		36

ソフトウェア技術者協会（ＳＥＡ）は，ソフトウェア・エンジニアの，ソフトウェア・エンジニアによる，ソフトウェア・エンジニアのための団体であり，これまでに日本になかった新しいタイプのプロフェッショナル・ソサイエティたることを目指して，１９８５年１２月２０日に設立されました．

現在のソフトウェア技術が抱える最大の課題は，ソフトウェア・エンジニアリング研究の最前線（ステイト・オブ・アート）と，その実践状況（ステイト・オブ・プラクティス）との間に横たわる大きなギャップを埋めることだといわれています．ソフトウェア技術の特徴は，他の工学諸分野の技術にくらべて属人性がきわめて強い点にあります．したがって，そうしたテクノロジー・トランスファの成否の鍵は，研究者や技術者が，既存の社会組織の壁を越えて，相互の交流を効果的に行うためのメカニズムが確立できるか否かにかかっています．ＳＥＡは，ソフトウェア・ハウス，計算センタ，システム・ハウス，コンピュータ・メーカ，一般ユーザ，大学，研究所など，さまざまな職場で働く人々が，技術的・人間的交流を行うための自由な＜場＞であることをを目指しています．

ＳＥＡの具体的な活動としては，特定のテーマに関する研究分科会（ＳＩＧ）や地方支部の運営，月刊機関誌（ＳＥＡＭＡＩＬ）の発行，各種のセミナー，ワークショップ，シンポジウムなどのイベントの開催，既存の学会や業界団体の活動への協力，また，さなざまな国際交流の促進等があげられます．

なおＳＥＡは，個人参加を原則とする専門家団体です．その運営は，つねに中立かつ技術オリエンテッドな視点に立って行われ，特定の企業や組織あるいは業界の利益を代表することはありません．

ＳＥＡＭＡＩＬ　Ｖｏｌ．２，　Ｎｏ．７　昭和６２年７月１日発行

編集人　岸田孝一

発行人　ソフトウェア技術者協会（ＳＥＡ）

〒１０２　東京都千代田区隼町２－１２　藤和半蔵門コーブビル５０５

印刷所　サンビルト印刷株式会社　〒１６２　東京都新宿区築地町８番地

定価　５００円

編集部から

秋の深まりとともに

前号（５－６合併号）が皆さんの手元に届いたのは残暑の頃でしたから，本来ならこの号は，秋の初め頃には発行しなくてはいけなかったのですが，編集スタッフのパワー不足から，「ともかく冬にずれこまないように」というのが，精一杯の努力目標になってしまいました．

９月末の幹事会で，ある幹事から，次号の発行予定について質問があったとき，「いま２冊同時に編集しているから，２週間後には２冊同時に届くよ」と編集長が背伸びをして答えたのですが，「あまり期待はしてないけど，頑張ってください」と逆に慰められるといった今日この頃です．

春のセミナー

３月のセミナー・ウィークのレポートを５セッション分まとめて掲載しました．４号で２セッション報告していますので，合計７セッションをカバーしたことになります．残りの５セッションは，レポータがギブアップしてしまいましたので，ひとまずこれでレポートを終了します．ただし，東工大の片山先生の「属性文法」の講義内容は，先生ご自身が bit 誌の９月号に解説されているので，興味ある方はそちらをご覧ください．

第９回ＩＣＳＥ特集

この春に開かれた第９回ＩＣＳＥに参加された方々の手によるエッセイ風のレポートです．ＳＥＡが日本を代表する協賛団体であったことから，今回の会議には，全部で３０名の会員が参加されました．

プログラム委員の鳥居先生，玉井さん，論文発表者の野村さん，ツール・フェアに出品された落水先生，以上４人の方から，バラエティに富む原稿をいただきました（夏前には，もう原稿がそろっていたのに，掲載がおくれてゴメンナサイ！）．残念ながら，プログラム委員長の大役を務められた岸田編集長の原稿は間に合いませんでした（聞くところによれば，ソフトウェア科学会の機関誌への寄稿も約束していて，そちらもまだできていないとか）．いずれまた別の機会に，思い出話でも書いていただきましょう．

ちなみに，第１０回はシンガポールで，来年の４月中旬に開催されます．第６回の東京以来のアジアでの開催になります．協賛団体としてのＳＥＡでは，また，グループ・ツアーを企画する予定ですので，皆さんも今から準備（来年度海外出張予算の手当，スケジュール調整等）を心掛けておいてください．

分科会から

ＣＡＩ分科会からのレポートを，キックオフ案内以来，初めて掲載しました．最近，分科会からの寄稿がほとんどありません．それぞれ活発に活動しているという噂だけは聞くのですが，どうか皆さんふるってレポートを書いてください．よろしく．

会員からの投稿

ＩＣＳＥとほぼ同じ時期にカリフォルニアを訪問された寺嶋さんからの「西海岸ＣＡＩ事情」です．ＣＡＩ分科会のレポートと併せてお読みください．

支部から

中日ドラゴンズの成績は竜頭蛇尾に終わってしまいましたが，名古屋支部は順調に運営され，会員も着実にふえつつあります．他の地区の方々，長野や名古屋をモデル・ケースとして，それぞれの地元に支部を設立してください．いつの日か，１０月の情報処理月間に日本全国縦断フォーラムを開催したいものです．

会員状況

１０月末には，１２００人にとどきそうです．最近の傾向は，支部が設立された地方での大幅増加を含めて，地方での会員が増えていることです．居住地域は，北は北海道（稚内）から南は沖縄までと広がっています．季節の便りなどを編集部宛（ＳＥＡ事務局内）にお寄せください．また，年令層も２０代が増えています．

次号予告

今月，編集長がまたしても海外へ出かけてしまい，原稿整理のアンカーが不在になるために，いつとは確約できませんが，１１月初旬には発行すべく努力します（記：勝手に編集長代理を買って出た野辺良一）．

SEA 春のセミナー・ウィーク： Session B1

電子出版技術の動向

講 師： 大野 俊 治（アスキー）
報告者： 田中 慎一郎（日本タイムシェア）

１．はじめに

本セミナーは，テーマがテーマだけにというべきか，テキストの出来がすばらしい．セミナーの後半でも紹介されているＴＥＸを用いて作成されており，出力も480 dpi のＬＢＰにより行われているために，印字が非常に美しい（ちなみにＳＥＡＭＡＩＬは ditroff を用いて，300 dpi のＬＢＰ出力である）．このため，本テーマに関心のある方は，このテキストを一読されることをおすすめする．テキストそのものが，具体事例にもなっている，というわけなのである．

さて，講義内容はというと，前半が電子出版技術の一般的動向についてであり，後半は文書処理システムとしてＴＥＸをとりあげて，その概要を紹介するというものであった．ＴＥＸは，スタンフォード大学の Donald E. Knuth 教授が開発した組版システムであるが，私がここでこれを簡単にまとめることには無理があるため，文末にテキスト中の参考文献を掲載することで御容赦願いたい．ここでは，主にセミナーの前半部分について，私がＳＥＡＭＡＩＬの編集に若干拘った経験をまじえて紹介させていただく．

２．電子出版技術とは

「電子出版技術」という言葉は，多方面の分野に渡る内容を包括した広義の意味を持つ言葉であるということで，講義では，これを出版物の編集／制作と流通の２つの分野から促えた場合の要素について述べられた．

２．１ 編集／制作

出版物の制作者側から見ると，電子出版技術に対する一番の関心事は，「編集／印刷の工程を電子化することにより，いかに生産効率の向上と経費の節減を図るか」にあるという．これはさらに次の５つの要素に分けることができる．

(1) 入力： まず原稿を作成する段階から文書をマシン・リーダブルな形態をとることが重要である．これは，この後の工程を電子化する上でも大きな意味を持つ．

悲しいかな私は，今回初めて手書きでＳＥＡＭＡＩＬの原稿を書くはめにおちいったが，はっきり言ってこれは，「２度手間」以外の何物でもない．ワープロおよびネットワークの発達等により，著者が直接マシン・リーダブルな原稿を提供するのが一般化しつつあるということだが，私も早くなんとかしたいと思っているところである（今回この原稿を入力してくれた人，ご苦労さまでした）．

(2) 編集： 編集者の主な仕事は，執筆者からの原稿を校正したり，版組の段階で必要となる書体指定などの指示を与えることである．講師の所属しているアスキー社では，この段階もフロッピー上で行えるようになっているという．

ＳＥＡＭＡＩＬでも，原稿が執著者と編集者の間を行き来することは，通常ほとんどなく，ここから先はマシン上で行われている．つまり，入力は一部の図や表を除いて全てコンピュータに登録されているので，エディタをつかって，roff，tbl，pic，enq 等のサブコマンドを，原稿の修正を行ないながら埋め込んでいくことになる．

(3) 組版： 実際に活字（写植）で文字を組む作業である．この段階を電子化することにより，植字の際の誤りをなくしたり，組み上がりを予めＣＲＴ画面，あるいは普通紙の上で確認できるようになり，日程の短縮と経費の節約が可能となる．

実際こうしたやり方でなくては，つまり，活字を組むというような作業でやっていたのなら，（船便批判はあるものの）ＳＥＡＭＡＩＬを発行しつづけるということは無理であったし，これからもそうである．言い替えれば，電子出版技術の普及によって，ＳＥＡＭＡＩＬが遅ればせながら発行できている，ということになる．但し，プリビュー（画面上で印刷イメージを出力して確認を行なう）ができるということと，誤植等が全くなくなるということは，必ずしも一致しないようではある（？！）．

(4) 割付： 組み上がった文字を，きめられた大きさのページ上に図版とともに，バランスよく配置する作業である．この段階では，表や図の作成を支援する機能が求められる．

ＳＥＡＭＡＩＬでは，図や表のスペースを空けて文字だけをまず割付て，印刷の段階で合成することと， tbl，picなどのコマンドを使って，表や図を文字と一緒に割り付けている．

(5) 印刷： 実際の出版において，一番経費のかかるのは，製版と印刷である．この段階を電子化することにより，小量の出版物でも必要量だけ印刷することが可能になり，経費の節減ができ，最終的な出版物の価格を安くすることができるという．

これら各段階は，それぞれ専門の業者により処理されており，次段階へのインタフェイスがいかにスムーズにいくかが問題となっていた．従来の出版では，これは「紙」により中介されていたのだが，電子化によりマシン・リーダブルな媒体や，通信にとって代わられようとしている．

さて，出版物の編集／制作には上のような要素があるわけだが，従来の「紙」主体の作業から現在までには，２つの「革命」があったという．すなわち「ワープロ革命」である．

（１）第１次ワープロ革命

オアシスに代表されるような，ワープロが世に登場したことによる変化である．この段階では，

- ワープロによる入稿
- ワープロによる修正
- ワープロによる原稿保存

が行われるようになった．

これは「革命」と呼ぶにふさわしい画期的現象で，それまでの原稿用紙を介して，活字をひろい，版をおこすという世界からうってかわって，写植屋自体がフロッピーを受付けて，それを入力とする世界へと大きく変化したのである．但し，この段階はまだ本文のみ対象であり，単なる「入力作業の省力化」の段階でもあった．

（２）第２次ワープロ革命

松，一太郎などの良質のパソコンソフトが登場したことによる変化である．単なるワード・プロセッサではなく，ものがコンピュータになったために，

・各種ワープロ間での媒体，フォーマット変換
・ワープロから写植システムへの媒体，フォーマット変換
・簡単な組版の指定

などが可能になったのである．

原稿用紙をフロッピーにおきかえただけ，という第１次革命と較べて，組版の作業までワープロ上で行えることにより，原稿作成から印刷までの手続きが，大幅に変更されつつある．この段階は「指定作業の効率化」が起っている段階であるといえよう．

２．２ 流通

さて，電子出版技術というものを，流通の視点から見るとどうであろうか．最近では出版物＝印刷物という概念は通用しなくなってきている．見近な例でいえば，パソコン用のソフトウェアなどは，明らかに出版物としての形態と流通径路を使用して販売されている．こうした出版物（？！）の最近の流通媒体を簡単にまとめてみる．

a) フロッピー・ディスク

先に述べたように，ソフトウェアの媒体としてよく用いられている．

b) カセット・ブック

語りの入ったテープと本とが組み合わされているようなもので，少し前からよく見かけるようになっているが，最近では書店でも専用コーナーなどができていて，文芸，ハウツー，ノンフィクションと分野が広がっていて，サラリーマンが通勤の行き帰りにウォークマンで聴いているとか．

c) ＣＤ

今話題の，といったところだが，製品化／標準化はこれからで，出版界では先陣争いのために一時的にＣＤフィードが起きているといったところ．現状では，特定用途向けの限定機能を提供しているにすぎない．

CD-I（CD Interactive）
CD-V（CD Visual）
VDI（Digital Video Interactive）

d) パソコン通信，コンピュータ・ネットワーク

不特定多数に対して情報を提供するという意味から促えれば，データベース・サービスやパソコン・ネットワークも電子的な出版の媒体であり，流通経路であると考えることができる．また，単に個人がコンピュータにアクセスして，という使い方だけでなく，逆に情報提供者から集めたものをまとめて紙にした上で，ユーザに配布するといったサービスも行われはじめている．

3．技術的背景

ワープロ革命ということで，原稿入力の電子化については触れたが，電子出版技術全体から見れば，それを紙にする技術が伴わなくては無意味である．そうしたことから，電子出版を可能とした技術的な背景についてまとめられた．

a) パーソナル・コンピュータ／ワークステーション

やはりパソコンやワークステーションの発達が，非常に重要な要素であるという．従来まで，ミニコン以上のシステムでしか利用できなかった文書処理システムが，机の上にのってしまうまでになってきたのである．

これは単に，ハードウェアの向上により小システムでも充分な処理能力が達成されるようになった，という意味だけでなく，この結果，文書処理システムに関する広い市場が形成されたことにも注目する必要がある．つまり，従来の電子組版システムは，専用ハードウェアを用いた専門家向けのものであったのに対し，より幅広いユーザを対象としたシステムが作られるようになってきたわけである．

b) レーザ・プリンタ

写植機に較べればまだまだとはいえ，出力装置として，高速／高解像度のレーザ・プリンタが，小型／軽量化され，低価格で入手できるようになったことも大きな意味を持つ．

現在一般的に使用されているレーザ・プリンタは，200～400 dpi （dot per inch ：この出力が300 dpi）の解像度のものであるが，写植機は，2000～5000 dpi 程度の解像度はあることになるので，一般の書籍にこれを用いるのはまだ無理である．しかし，コストの差は非常に大きく，簡易的な出版物や組版確認のための試し刷りに手軽に用いることができるようになったという点でその意義は大きい．また文字だけではなく，イメージやグラフィックスの出力も行えるため，これらを含んだページ・イメージの印刷が行えるという特徴もある．1000 dpi 程度のものまでは実現可能ということなので，今後ますます美しい印刷物が手軽にできるようになりそうである．

c) 補助記憶装置

文書処理を行う上では，次のような各種のデータやイメージを保存するために，大容量の記憶媒体と，データ圧縮技法が必要とされる．

- フォント
- 文書の書式，マクロ
- 図，写真
- 文書の原稿
- 組版システムの出力
- 印刷装置への出力イメージ

そのため，電子出版システムを利用する上では，大容量のハードディスクが利用できることが重要な要素となっているのが，最近では，ＣＤ－ＲＯＭや光ディスクなども注目されている．

d) ビットマップ・ディスプレイ

電子出版システムが，従来の印刷システムと大きく異なる点の一つは，最終的な印刷イメージを前もってＣＲＴ上でプリビューできることである．特にインタラクティブなシステムでは，ＣＲＴ上で作成した文書のイメージがそのまま印刷物として得られるために，ＷＹＳＩＷＹＧ（What you see is What you get）という言葉が用いられたりする．

ここで面白いのは，ＣＲＴの解像度は，印字装置に較べれば劣るために，正確にはＷＹＳ＜ＷＹＧとなっていた．そのため，バッチ方式のシステムを皮肉って，"What you get is more than what you see."という言葉が用いられていたという話である．最近では，300 dpi のＡ４イメージがそのまま表示できる専用ディスプレイも登場し，また，ディスプレイ・プロセッサの発達により，イメージやグラフィックスも高速で描画することができるようになってきているというが，まださほど一般的ではないのではなかろうか．

ちなみにＳＥＡＭＡＩＬの世界は，完全にＷＹＳ＜ＷＹＧであり，確認はほとんどレーザ・プリンタにたよっているのが現状である．

e) デジタル・フォント

最終的な印刷物のできは，文字そのもののデザインによるところも大きい．このため出力装置の特性や解像度に応じてフォントを生成するためのシステムが作られ，販売／利用されているという．

また，フォントのデータ容量を少なくするための圧縮技術等についても研究がさかんである．これについては後でまとめることにする．

4．電子出版システムのクラス分け

電子出版システムを，簡単にクラス分けしてみる．

a) ワード・プロセッサ

非常に低レベルのものではあるが，電子出版技術とし

てはなかなか重要な位置を占めるものといえる．

主用途：手紙，原稿，社内文書など

分　量：小量，小部数

機　能：簡単な組版処理まで（句読点の禁版処理程度）

装　置：ドット・プリンタ（200 dpi程度）

b) デスク・トップ・パブリシング

現在最も話題となっているもので，ＤＴＰなどとも呼ばれている．文章だけでなく，図や写真などもとり込めるようになっていることが多い．代表的なシステムは，Apple社のMacintogshとLaserWriterの組合せである．

主用途：ニュースレター，マニュアル，スライドなど

分　量：小量（100ページくらいまで）小部数

機　能：基本的な組版処理

装　置：スキャナ，レーザ・プリンタ（200～400dpi）

c) 専用システム（ＣＴＳ）

専用システムといってもその幅は広く，b) との中間的なものも多数存在する．ある程度分量のあるものを対象とし，やや特殊な組版処理（数式処理など）ができるようになっている．

主用途：広告，新聞，書籍など

分　量：大量，大部数

機　能：複雑，高度な組版処理

装　置：写植機，製版機（1000 dpi 以上）

５．最近の話題

a) ＰＤＬ（Page Description Language）の普及

ＰＤＬは，組版のための制御情報ではなく，組版の結果のページ・イメージを記述し，出力装置への印刷指示情報として利用するためのものである．例えば，ページのどの部分に，どの書体のどの文字があるとか，どの点からどの点に線を引くとかを，コマンドで指示させることになる．ページを構成する要素にのみ着目していればいいため，使用する文書処理系や出力装置に依存しないように言語を設計できる．この結果，ユーザ・レベルから見ると出力装置の解像度の違いなどが全く影響しないシステムとなる．つまり今のワープロ・ソフトだと，２４ドットのプリンタだったら大丈夫だが，１６ドットのプリンタでは出力できない，などというのが普通であるが，ＰＤＬで記述されたものは，相手が 200 dpi ならば 200 dpi なりに出力してくれるようになる．

ＰＤＬとしては，Adobe Syatem の PostScript，Imagen の DDL，Xerox の Interpress などが有名である．

b) デジタル・フォントの普及と新しい圧縮／展開，技術の発達

従来どのようにフォントを持っていたかというと，一番多いのがドットで持つというもので，ある程度大きくなってくると，ベクターで持つようになっていた．

ドットで持つとそのままＣＲＴに出力できるという利点はあるが，非常にデータ容量が大きなものになってしまう（何種類か字体を持とうとすると，何メガという領域が必要となってしまう）．これに対し，ベクター・フォントで持てば，その点は解消されるが，今度はそれを展開するのに時間がかかってしまうという欠点がある．

このような圧縮技法／展開技法も電子出版技術として非常に重要なものであるため，その研究もさかんである．例えば，平均５０バイト程度に漢字フォントを圧縮したフォント屋が現われたり，高速展開のために，専用のＬＳＩやボードが登場したりしている．現代はフォント屋がチップについても考える時代になってしまったということである．

６．おわりに

この後，後半ではＴＥＸの話をされたわけであるが，最初に述べたとおり，簡単にまとめることなどとても無理なので，テキストの参考文献を引用してこれに変えたいと考える．ノートが不充分だったため，わからない所を自分で適当に埋めたり，カットしたり，テキストから無断引用したりで大野氏にはまことに申しわけないが，本稿の主旨を御理解の上，御容赦願いたい．

［参考文献］

(1) Lawrence A.Beck Report from ANSI X3V1. TUGboat, 7(3):132, 1986.

(2) 藤井啓文，ＴＥＸ－欧文誌を速く安く出版する可能性－について，日本物理学会誌，４１（１２）：９５７，１９８６

(3) Donald E.Knuth. The TEX book, Addison-Weasley, Reading, Mass, 1984.

(4) Leslie Lamport. A Document Preparation System - JATEX. Addison-Weseley, Reading, Mass., 1986.

(5) 山内長承，来住伸子，ＴＥＸ文書清書システム，コンピュータ・ソフトウェア，４（１）；４４，１９８７

(6) D.Lucarella. TEX Document Retrieval. PROTEXT I, Proceeding of the First International Conference on Text Proceesing Systems, Boole Press, Dublin, Ireland, 1984.

SEA春のセミナー・ウィーク：　Session C1

ＡＩ パラダイムとソフトウェア開発

講　師：　大木　幹夫（日本電子計算）
報告者：　村井　進　（テクニカル・ライター）

１．はじめに

ＡＩとソフトウェア工学の融合が話題になって久しいが，まだ具体的な成果は現れていないようである．その大きな問題点は，開発成果物に対する知識と開発プロセスに関する知識の両方を，いかに表現し利用したらよいかが明確になっていないことであろう．

当セッションでは，ソフトウェアの部品化および再利用のための環境構築について，知識工学の立場から研究を続けてきた大木さんの経験を踏まえて，ＡＩパラダイムのソフトウェア開発現場への応用について，具体例を交えた解説が行われた．

大木さんが最も強調したかったのは，「ソフトウェア開発プロセスを決定し，記述するのにどのようにＡＩを応用するか」という点だったようである．したがって，解説も大きく分けて，「ソフトウェア・プロセス，特に設計プロセスやプログラミング・プロセスを決定する要因は何か．そして，プロセスをフォーマルに記述し，実行する場合の問題点は何か」ということと，「ソフトウェア開発のためのエキスパート・システムとは何か．具体的にはどのような例があるか．そしてその中では，ソフトウェア開発知識はどのように表現されているか」という２つに分かれていたと思う．

２．ソフトウェア開発プロセス

ソフトウェア開発プロセスについては，プロセス自体をモデル化する問題と，プロセスに影響を及ぼす人間的側面および組織的側面について話が進められた．

２．１　ソフトウェア開発プロセスのモデル化

（１）　設計過程のモデル化

設計プロセスをモデル化するためには何が必要なのだろうか．モジュール分割作業の分析を例にとって考えてみる．たとえば，設計過程でモジュール分割を行うアプローチとしては，

- トップダウン・アプローチ
- ボトムアップ・アプローチ
- 最重要コンポーネント最優先アプローチ

が考えられる．ところが実作業では，これらのアプローチのうちのどれか１つを徹底して行えばいいというものではなく，普通は次のような折衷案的なアプローチがとられる．

- トップレベルから２－３層までは，トップダウン・アプローチで分析する．
- ２－３層以降は，トップダウン・アプローチとボトムアップ・アプローチを併用する．
- 同一レベルの分析では，最重要コンポーネント最優先アプローチを採用する．

そして，この分割規則に従ってモジュール分割する際の基準としては，次のようなものが考えられる．

- データの意味的従属性による分割
- 制御変数の関連分析による分割
- 意味ネットワークによる分割

それならば，これらの分割規則や分割基準に従って作業をガイドする設計支援システムを構築し，それを使ってモジュール分割を行えば，いかにもにもうまくいきそうである．しかし，実際にはそれだけではうまくいかない．なぜなら，これらの規則や基準に関する具体的な知識を，どのような場面でどういう具合いに使用するかは，やはり人間の個別の知識に基づく判断にまかされた問題だからである．したがって，これらの判断を上手に行えるエキスパートの知識を定式化して，エキスパート・システムに組み込むことなくしては，この設計支援システムは役に立たないのである．

プロセスという立場からみれば，モジュール分割プロセスを抽象的に記述することはできるが，実際に具体的な問題について，どういう順序で分割を行ったらいいかを記述するのは，非常に難しいわけである．

（２）プログラミング過程のモデル化

次に，プログラミング・プロセスを形式的な記述をし，

そのプロセスを実際に実行する場合の問題点を探り出してみる．次のプログラムは，仮想的な言語でプログラミング・プロセスを記述したものである．

```
type pipe: array[0..1] of
    record
    spec : Que of Specification
    func : array[0..m] of function;
    need-arrange : boolean;
  end
procedre PROGRAM_BUILDER
          (module-no:integer, A : pipe );
begin
    n,i   : integer,
    B     : pipe,
    retry : boolean;
    acquire-specification(A[module-no]);
    confirm-environment(A[module-no]);
    while NOT decomposition-completed do
    begin
     n := n + 1;
     decomposition-to-module
        (A[module-no].spec,B[n].spec,B[n].func);
    end while;
    for i := 1 to n do
    begin
     if module-already-present(B[i].func) then
       apply-corresponded-module(B[i].func);
     else
     if need-to-new-module(B[i].func) then
       begin
         process PROGRAM-BUILDER(i,B);
         describe-module-call;
     end;
    end for;
    describe-own-function(A[module-no].func);
    if need-arrange
          then modfication-determinate(A,B);
    if all process of B = completed then
      while retry do
        begin
          retry := false;
          for i = 1 to n do
            if B[i].need-arrange := false;
            process PROGRAM-BUILDER(i,B);
            describe-module-call;
            retry := true;
        end if;
    enf while;
end procedure;
```

これはいま流行のプロセス・プログラミングの一種である．このプロセス・プログラムの細かい解釈は，読者におまかせする．このプログラミング・モデルから，次の作業がクリティカルなものとして抽出される．

（a）confirm-environment
仕様の内容に，どのように優先度をつけて整理するか．

（b）decomposition-to-module
どのように仕様から機能分割するか．

（c）aplly-corresponded-module
どのように機能にあった既存モジュールを選択し，適用していくか．

（d）modification-determination
相互に調整の必要なモジュールをどう決定するか．

（e）プログラム機能の検証をどのように行うか．

これらの知識については，ほとんど形式化が行われていない．プロセスは記述できるが，そのプロセスを実際に走らせた場合の分岐点における判断方法が決っていないからである．したがって，現状でこのプログラミング・プロセスを支援する方法としては，ベテラン・プログラマの持つ知識を知識ベース化して利用することが最も現実的だろうか．

ここでちょっと飛躍するが，以上の述べてきたことから，ソフトウェア開発プロセスをＡＩで支援することに関して，結論として以下のことを指摘することができる．

- 開発プロセスにおける各フェーズ／モジュールをアクター（Actor）として扱うと，人間のソフトウェア開発作業とうまく対応がつく．
- 開発プロセス中に生じるコミュニケーション，モジュール間のインタフェースのとり方として，メッセージ・パッシングの考え方が自然である．
- モジュールの分割基準として，データ間の依存関係がかなり利用できる．

・開発戦略や制約条件とマッチしなかった場合のフィードバックの方法は，開発者の経験則に依存している．

２．２　人間的特性・組織と開発プロセスとの関係

（１）　人間的特性との関係

次に，ソフトウェア開発プロセスと人間的特性の関係について考えてみる．大木さんの説によれば，技術者の設計行動パターンには大きく分けて７つあり，各々のパターン毎に設計方法論に対する評価のパターンがかなり異なっている．その７つとは，

・積極的な専門家指向型
・なりゆき・自己中心型
・気配り・無難型
・体制的変革指向型
・保守的・馬車馬型
・体力依存・粗製乱造型
・独善的職人仕事型

である．この７つの型の人について，変更管理性，記述規則の厳密性，記述内容の詳細化レベル，機能や制御フローの分かりやすさ，入出力ファイル記述の完全性を比較すると，非常に大きな差が出るそうである．この部分の詳細については省略するとして，重要なことは，従来設計方法論の適用を考える上で重要とされてきたプロジェクト要員数や工期，対象分野などは，そうおおきな役割を持たないということである．むしろ，仕事に対する積極性や専門的な技術水準などが，設計方法論の適用対象を決める上で重要なキーとなっている．

（２）組織との関係

ＡＩ（特にオブジェクト指向概念）を開発プロセスのモデル化に適用することについて，要員をオブジェクト指向言語のオブジェクトと見なし，組織全体と人とのかかわり合いを類比させることを考える．以下に，具体的な組織内での人の行動を例にとって，オブジェクト指向概念との対応を示す．

（ａ）一般にソフトウェアは人について回る．そこで組織変更の都度，連絡先を確認する社員名簿が必要となる．

→　オブジェクトに対するメッセージの宛先名簿の必要性＝システム・ブラウザ

（ｂ）意志統一のために会議を頻繁に開くが，儲っている会社ほどその頻度が低いのは，クラスの構成（モジュール分割）がいいからである．

→　クラスの継承関係をうまく利用して，メソッドが容易に利用できる構成．

（ｃ）会議の場では，理屈より声の大きい人が主導権を握る．この場合，優先度付け規約が必要．

→　全体を見て並列的に処理するような，オブジェクトの優先度付けを行うメタ・レベルのコントローラが必要．

（ｄ）階層的な組織でも，フレキシブルで外的変化に対応しやすい組織ほど，コミュニケーションの階層が短くなる．すなわち，権限以上による分散型の意志決定が行われている．

→　メソッドの継承は，あまり上位レベルまで行かない方がいい．

（ｅ）階層構造的なライン部門の外に，いろいろなスタッフ部門の援助のもとに業務が行われる．

→　クラスの多重継承＝1つのオブジェクトが，いろいろなクラスから性質を継承する．

３．ソフトウェア開発知識の表現方法

一般にＡＩとは，「人間が，与えられた外部状況や情報から，どのような認識，推論，評価を行い，結論や判断を導き出すかのプロセスをモデル化し，コンピュータ上でそのプロセスを再現することに」重点をおいたものである．それに対してソフトウェア開発環境とは，「人間のソフトウェア開発作業を改善（効率と信頼性の向上）するために，開発プロセスを自在に変化させる仕組み作り」と言える．実際に，開発環境を構築することは，開発担当者の能力や技術の発達によって刻々と変化してゆく開発プロセスの進化に，永続的に対処する仕組み作りに他ならない．

したがって，「開発プロセスにおいて，プロジェクトの組織形態や人間の思考過程が環境とどのように係わっているかをモデル化すること」が，ＡＩおよび開発環境構築の双方で目下必要とされることといえる．すなわち，それは人間の認識，判断，評価，および協同作業等と環境との関わり具合いをモデル化することである．

このような認識のもとに，１章ではソフトウェア開発プロセスについて各種の分析を行ってきた．引き続いて，実際の講演ではＡＩツールの一般的な特徴を基に，ソフトウェア開発プロセスを支援する可能性について話が進められた．

ＡＩツールの特徴としては，次のようなものがあげられる．

（ａ）アプリケーション開発方式

・部分的に正しいプロトタイプから段階的に成長させる

（ｂ）従来の計算モデルと異なる特徴

・記号による推論：記号の背後にある意味を操作
・非決定論的な駆動：プロトタイピングが容易．ルールの追加・変更が容易
・性質の継承：ルールの蓄積・保守が容易

これに基づいて，エキスパート・システムの適用しやすい工程をあげると次のようになる．

（ａ）あいまいなものの関係を発見的に整理してゆく必要のある工程．

・あいまいな要求内容のモデル化
・プロトタイピングとインタープリタによる要求内容の検証
・ユーザ・インタフェースの設計

（ｂ）経験的な知識が蓄積され，かつ形式的に表現できる工程

・ソフトウェア部品の再利用とプログラム合成
・仕様定義プロセスのコンサルテーション

このような観点から，CSRL（Conceptual Structured Representation Language），DSPL（Design Structures and Plans Language），Draco，Objtalkなどを例に取り，ソフトウェア開発プロセスを支援する知識の表現方法について具体的な解説が行われた．詳細は省略するが，中心となったのは，General Taskの考え方，知識のフレーム表現とパラメータ記述部の類比，およびオブジェクト指向型の知識表現である．

最後に，オブジェクト指向言語に焦点を当て，それをソフトウェア環境，特に部品再利用のための環境に利用するためにはどのような拡張が必要かを示すと，以下のようになる．

（ａ）メソッドの機能拡張

・メソッドの条件付き活性化や不活性化
・位相の異なるオブジェクト（重ね合わせができるオブジェクト）の支援

（ｂ）階層分類化について

・データ，制御の階層的分類化
・制約条件，評価の階層的分類化
・階層分類プロセスへの部品のあてはめ

（ｃ）メッセージ機能の拡張

・カラード・メッセージ
・インヘリタンスの関連付を担うメッセージ
・メソッドを生成するメッセージ

（ｄ）部品合成モニター機能

・視点の役割と動的な変更
・階層的な構成部品と全体を関連づける黒板

（ｅ）全体と部分の関係から，抽象と具象の関係への体系化

本来は，本章で知識の表現方法とプロセスの関係についてもっと詳しく書くべきだと思う．しかし，レポータは知識工学の専門家ではないので，そのあたりは御容赦頂きたいと思う．

４．質疑応答

Ｑ： ＡＩとソフトウェア開発の融合という意味では，今後どのように進展してゆくと思われるのか．

Ａ： これは，今後の戦略上大きな問題だと思う．具体的にはＡＩは，トップダウン方式とボトムアップ方式を結合する役割を果たすのがよいと思う．たとえば，ＯＳに近いような機能とCSRLなどの一部を，通常のルーチン・コールに置き換えることなどを考えている．また，Interlispの世界のように，ＡＩ機能を核として非常に迅速なプロトタイピングを支援するという方向も面白いと思う．ＡＩ技術自体がどう進展するかということに関しては，確実なことはいえないと思う．

SEA春のセミナー・ウィーク
セッション C2

新形態プログラミング

講　師：　二木　厚吉（電総研）
報告者：　中川　中　（SRA）

1．はじめに

本セッションでは，最近注目を浴びている新しいプログラミングのパラダイム，すなわち，仕様記述の段階から実行可能な形式的言語を利用し，幾つかの変換を施すことにより最終的なプログラムを作り出す，という方式の概略が述べられた．

以下では，このパラダイムの利点を簡単にみた後，それを実現するために必要な２種類の技術，実行可能仕様とプログラム変換について講義の内容を要約し，終わりに仕様記述言語の発展に一つの方向を示唆するＯＢＪ２に触れる．

2．実行可能仕様＋変換

この新しいプログラミング・パラダイムの利点は，次のように纏めることができる．

- 仕様自身をテストできる．
- 仕様のライブラリ化，知識ベース化ができる．
- プログラム開発過程が（半）自動化される．

このうち，開発の自動化に関しては，過大な期待を持つべきではない．というのは，仕様の本来の目的を無視し，単なるプロトタイプ・キット，ないしプログラミング言語の追求に走りがちになるからである．仕様の支援という面では，実行の可能性は，解析ないし検証機能の強化という側面から眺められるべきである．

3．実行可能仕様記述言語

仕様記述言語には，実現（実行）性，形式性，構築性，モデル性などが望まれる．これらの条件すべてを満たすような言語の設計は困難であるが，計算モデルの発展とともに，幾つかの有望な言語が現れはじめている．それらの言語の有効性は，仕様記述としての宣言的な意味，あるいはモデル性，という観点，実行するための操作的な意味づけという観点，の両面から把握すべきである．

（１）関数モデル

モデルとしては写像，操作面では帰納的関数，ラムダ計算，コムビネータなどの理論が整備されており，HOPE，Mirandaなどが代表例である．

（２）論理モデル

述語論理に基づくモデル性と，推論規則を用いる機械的な証明という操作性を基礎とする．Prolog，ＥqＬogなどが有名である．

（３）代数モデル

始代数，終代数などによるモデル化と，項書換えシステムによる実現とを基盤とし，HISP，OBJ，ASLなどの代表例がある．

（４）オブジェクト・モデル

オブジェクトとメッセージにより，記述対象を自然な形でモデル化する．Smalltalk，Orient 84などがある．

これらの言語が台頭してきた背景には，プログラミング言語が命令型から宣言型へ，手続き指向からオブジェクト指向へ，また逐次集中型から並行分散型へ，と推移してきた事情がある．

それぞれの言語には利点・欠点が相半ばしており，いまの時点では優劣をつけ難い．例えば，並行処理のように時間の概念を必要とする問題領域では，関数や述語論理によるモデル化は難しい．反面，様相論理やオブジェクト・モデルはそうした領域を記述しやすいが，操作上の意味付けが難しい．

4．変換法

プログラム変換には，以下のような方法が提案されてきた．

（１）コムパイリング法

仕様記述言語を超高級プログラミング言語と考えると，従来のコムパイラと同じような仕組みで効率的に実行で

きるコードを生成できる可能性がある．あるいは，宣言的な記述を直接実行するようなマシンの実現もありうる．

（2）部分計算法

汎用的なプログラムを目的に応じて特殊化し，実行・オブジェクト効率を上げる方法で，パラメータの固定化やインタプリタからのコムパイラの生成などの研究が進められている．

（3）展開／畳み込み法

コード変換の規則をあらかじめ定めておき，それに従ってプログラムを機械的に変換して効率を上げる方法で，関数の展開，演算子の性質を利用した書換え，関数定義を用いての畳み込み，の手順を踏むことからこの呼称がある．再帰呼び出しのあるプログラムに有効であり，関数型や論理型のプログラムでの適用例がある．

（4）カタログ法

局所的に適用可能な変換戦略を蓄えておき，カズイスティッシュに変換を進める方法で，（3）のような一般性はない．新たな関数を導入することによる tail recursion 形への変換法などが知られている．

（5）パラメータ化法

上記の各変換法がアルゴリズム・レベルのそれであるのに対し，この技法は，予めモジュール単位でパラメータ化を行い，具体的なデータの種類や演算子の解釈をあとであたえられるようにするもので，プログラムの再利用／保守に飛躍的な貢献をする可能性がある．ＯＢＪ２はこれを object， theory, view という３種類のモジュールを提供することで実現している．

これらの変換法は補完的な役割を果たし，状況と必要に応じて使い分けられるべきである．

５．ＯＢＪ２

ＯＢＪ２は代数モデルに基づく言語の代表の一つであり，一般的な言語設計の点でも著しい特徴がみられる．それらを含めＯＢＪ２の特色を列挙する．

（1）ソート

ＯＢＪ２は抽象データ型をソート集合とその上の演算によって実現している．しかも，ソート間には半順序をつけることができ，ソート間の集合論上の関係，エラーの意味付けなどが自然でかつ厳密に扱える．

（2）パラメータ

すでに述べたように，ＯＢＪ２はルールを定義して，それをさまざまな用途に用いることができる．たとえば，整列を行う汎用のモジュールを用意して，それを整数の昇順の整列，降順の整列，文字列の辞書順の整列，逆順の整列，文の整列，等々に利用できる．しかも，パラメータが満たすべき性質を特定することもできる．

（3）混合記法

ＯＢＪ２では演算の記法を自由に定義できる．たとえば，+(X, Y), + X Y, X + Y, X Y +, のいずれの書き方も好みに応じて採用できる．

（4）項書換え

ＯＢＪ２の実行エンジンは項書換えシステムであるが，それぞれの演算子には交換則，結合則，idempotency, などを指定でき，書換えはそれらの性質を利用しつつ行われる．

（5）モジュール表現

（2）のようにしてパラメータ化されたモジュールを組み合わせることにより，簡潔な記法で階層化された複雑なモジュールを記述できる．

現在は書換え規則の満たすべき性質のチェック，たとえば停止性，合流性などの確認，あるいは実パラメータの整合性の検証，といった静的な機能にやや不十分な点がある．勿論，それらは将来に向けての課題であるが，Ｋ＝Ｂ　アルゴリズムや定理証明系にはいまのところ効率上問題があり，単純に解決するわけにはいかない．

SEA 春のセミナー・ウィーク：Session C3

創造的開発のための管理

講　師：　土井　利忠（ソニー）
報告者：　野辺　良一（ＳＲＡ）

１．はじめに

このレポートは，ワークステーション NEWS の開発責任者である土井さんの講演をまとめたものです．執筆にあたっては，管理される側のエンジニアの立場で，講師のお話に対して自由にコメント（『』で囲んである）を追加させてもらいました．当然ながら，以下の記事は，レポータの耳からはいって（聞きのがしがある），頭を通過して（偏見や独断が入る），書き下ろした（表現不足がある）ものであることを念頭に置いて，お読みください．

２．経験から

お手元にお配りした本は，ＣＤやそれに関連するディジタル・オーディオ機器開発の裏話を書いたものです．私が最近手がけたワークステーションの開発は，以前関係したディジタル・オーディオの開発と，非常に状況が似ています．今日のセミナーのタイトルは，「創造的開発のための管理」ということになっていますが，そんな管理の手法があるなら，私も聞きたい（笑い）．それで，少し方向を変えて，「開発プロジェクトを成功させるためにはどうしたらいいか」をお話してみたいと思います．

『人づてによる人物評では，かなり過激でユニークな方だと聞いていたが，最初の印象は，温厚な紳士に思えた．話す調子は丁寧であるが，強烈な自信に裏付けされているようであった．このへんの様子を伝えられないと，誤解される恐れがある．読者の方々の善意のフィルタに期待したい．

テキストの替りに配布された本は，「ＣＤはこう生まれ未来をこう変える」（天外伺朗著、ダイヤモンド社刊　１５００円）であった．自分の年齢が気になりだした管理者の方々にぜひご一読をおすすめしたい』

２．１　いくつかのプロジェクトから

まず始めにＣＤの話をします．ＣＤは別名をＤＡＤともいい，７７年末にいくつかのメーカがプロトタイプを発表しています．実際の開発が始まったのは，７９年の９月から，フィリップスとソニーとの共同作業でした．実際の開発は約７ケ月弱で終わっています．８０年の３月末には，現在と全く同じのフォーマットが決まりました．その時ソニーで開発に携わったのは，私を含めて３名，フィリップス側には３０名程いましたが，フォーマットの９割方は，ソニー側で決めたものです．つまり，ＣＤはきわめて小人数かつ短期間で開発されたものです．

次に，最近新聞をさわがせているＤＡＴですが，この開発には約３年もの時間がかかっています．８１社が参加したＤＡＴ懇談会という組織があり，そこでフォーマットその他を決めたのですが，ソニー１社だけでも，担当技術者の人数は５０名を越えています．つまり，ものすごい時間とお金がかかっているわけです．しかし，出来上がった結果はといえば，なんでもかんでも盛り込んだフォーマットになっており，決してよいものとはいえません．

商品を設計するときに，ユーザが必要とするだろういろいろな機能を考えるわけですが，そういった機能をなんでも取り込むことは簡単ですが，どうやって必要最小限にしぼり込むかが，非常にむずかしい．

よくいわれる「委員会シンドローム」，つまり多勢からいろんな意見が出て，それらの妥協案をまとめるということになると，非常に時間がかかり，また，結果としては，いまのＤＡＴのように，なんでもかんでも盛り込んでしまって，わけがわからないものになってしまう．

NEWS ワークステーションの場合は，ちょうどＣＤとほぼ同様でして，開発には，わずか７ケ月しかかかりませんでした．エンジニアは１０名だけです．それから製造準備に入って，昨年１０月のデータショウで発表し，１月から出荷を開始したわけです．

データショウでは，プロトタイプを何台かならべて，来場者にさわってもらっていました．そうすると，みなさんそれぞれ大変意地悪いソフトを持ってきて，ベンチマーク・テストしたり，システムを落とそうと試みたり，

いろいろなことをやられていました．

だれが本当のユーザで，だれが競合他社の人か，全然わかりません．しかし，最後の日に某コンピュータ・メーカの方が，名刺を持ってきて，「実は自分もこういうマシンを作りたかったのだけれども，うちではとても作らしてもらえない．上の方から降りてくる企画は，チンプンカンプンのものばかりだ．ソニーという会社がうらやましい」といわれた．

私は，その時に「ソニーでも普通の組織でできたわけではなく，社内ベンチャーというものを作ってやってきた．通常の組織でやっていたら，プロトタイプができる以前につぶされていただろう」と答えました．

２．２　大企業シンドローム

いわゆる大企業シンドロームは，構成員がサラリーマン化してしまうことによって起こります．サラリーマンの定義はむずかしいのですが，自分の地位の保全が主な行動目標になるというのが，特徴です．「休まず，遅れず，働かず」という行動パターンが，よく見られます．働かずといいましても，会社を休むわけではなく，結構残業なんかもするんですが，ベクトルの方向として，なるべく世の中に逆らわない，会社のなかでも体制に逆らわない．その分，人間関係を非常に重視するようになる．

デシジョンをトップがしたがる，なんでもトップが決めたがる，これが大きな弊害だと思います．トップは忙しいときもあれば，暇なときもありますが，いずれにしても充分な情報を得られない立場にある．不十分な情報でデシジョンするのは非常にむずかしい．したがって，デシジョンが形骸化・セレモニー化してしまう．

もう一つは，下から見るとデシジョン待ちになってしまいます．トップが決めないと何も動かない，したがって，トップが決定するまでポカンと待っている．世の中の動きは激しいですから，待っているうちに勝機を失ってしまうということがあります．

トップが決定したがりますから，回りはトップにいろんなプレゼンテーションをするんですね．で，プレゼンテーション合戦がおきるわけです．しかし，トップからみますと自分からデシジョンをするということに対して喜びを感じていますから，デシジョン権を下におろすことはなかなかしようとはしない．こうした現象は，大企業になればなるほどあります．したがって派閥争いが活発化してきます．これが大企業の一番具合悪いことですね．

人間関係からいうと，親分子分の関係が主流になってきます．親分子分で物事が推し量れることになりまして，これはどこの会社でもあると思いますが，トップのまわりをイエス・マンが取り囲むようになります．トップにとって，イエス・マンは心地好い訳でして，そういう人達を割とよく扱ってしまう．白アリが家を食い荒らすように，イエス・マンが大企業を食い荒らす結果になる．派閥抗争が発生すると，世の中で敵（競合他社）に勝つことよりも，まず社内で勝たないといけない．社外との競争は，今年負けても来年勝てばいいが，社内の隣の事業部に今年負けたら将来がなくなる（笑い），といったぐあいです．

『わたしの経験では，白アリ＝イエス・マンは，小さな会社や官庁の外郭団体等にも大勢いるようです．大企業での派閥争いは，最近ではコミックの材料にもなっています（例えば，「課長島耕作」）』

こうした状況になると，トップのデシジョンはほとんど狂ってしまう．デシジョンは，一定の戦略にもとづいて，１つ１つ積み上げていかなくてはならないのに，情報不足，派閥間の競合あるいは妥協，トップ自身の気まぐれなどによって，事態の本質から外れたものになってしまう．多くの人々が困った困ったといいながら，集団で破滅に向かっていきます．しかし，大企業のすごいところは，それでもつぶれないことですね（笑い）．

NEWS の開発にさいして，私が打ち出した方針は，「長」がつく人間がデシジョンをする時代は終わった，研究開発プロジェクトでは年功序列は有害無益である，デシジョンできる人間がデシジョンをする，ということでした．２４時間そのことを考えていられる人間がいい．ただし，戦略的デシジョンに関しては，なるべく１人に集中する．これは別にトップである必要はなくて，そのチームの中の誰か１人に集中するということです．

ここで，誰がデシジョンすべきかを決めるのは，トップないしはマネージャの役目だと思います．技術的デシジョンは，１人で何もかもというのは無理ですから，担当を分担することになります．したがって，それぞれのデシジョン・メーカは，全体の方向性を正しく把握している必要があります．

このような提案をしたところ，それでは NEWS の開発は社内ベンチャーでやりなさい，ということになりました．社内ベンチャーとは何かといえば，

・トップは金を出すが口は出さない．
・ただし使えるお金及び累積赤字の総額は決められている（歯止めとして）．
・会社のオーバヘッドは負担しない．ただし，間接部門の援助は受ける．

実は，こうした虫のいい制度がそれまでにあったわけでなくて，この時に決めたものです（笑い）．

３．開発プロジェクトを成功させるには

３．１　その項目

・人材を登用する（絶対に必要なこと）．
・専門性・人間性を持ったよいチームを作る．
・マネージャが邪魔をしない（仕事をしないということではない）．
・チームに燃えてもらう．燃えないチームはダメ．
・周囲の冷たい目や妨害にめげない．めげないためにはマネージャの力が必要．
・技術プロジェクトの場合には，開発プロジェクトが成功した後の戦略を考えておく
・時の流れを読む．
・短期間で開発する必要がある．皆（小人数）が熱気があるうちの短期間でやることで成功するケースが多い．研究は生鮮食料品で，腐らないうちに料理しないといけない．

３．２　人材を登用する

人材とは何か？　それは以下のようなセンスをもった人をいう．

・金が稼げないといけない
・全体を見通せるセンス
・直感が効く
・時代を先取りする
・強力な達成意欲がある
・問題解決能力は当然必要だが，それ以上に問題提示能力が必要
・物事に対して感激する人間性
・自分の技術を向上させる新しい技術にチャレンジしていく意欲

こうして上げていくと，そんな人材なんかいないという人がいますが，そうじゃなくて人材は必ずいます．ただ通常は隠れている．大企業シンドロームになっている会社（普通の組織）でいうと，こうした人材は不良社員になっています．ですから不良社員をみたら人材と思え（笑い），といってもいいくらいで，多くの場合不良社員の中から人材がみつかります．

そういう人達は，どこにでもいると思います．たまたま才能を表にだしているということもありますが，そんなことをすると組織に使われて，すでに消耗してしまっていることが多い．ほとんど表に出している人材は活用できない．ですから，新たなプロジェクトを始めようと思ったら，組織の中で疎まれて不良社員に化けている人材こそ連れて来ないといけない．人材を登用するということは，隠れている人材を見付けるか，あるいはまだその能力に比べて低い評価の人材に，責任ある仕事につけるということです．

そして，仕事がスタートしたら，どんな障害にもめげずに，プロジェクトをすすめてゆく気構えが必要です．新しいプロジェクトに，マイナスの要因を数えたらいくらでもあります．失敗したときのいい訳をいおうと思えば，いくらでもいえる．しかし，本当の人材は，そうしたマイナス要因をすべてプラスに変えていきます．いうのは簡単ですが，それができるためには，ある種の才能が必要なのです．

『これはなかなか含蓄がある言葉としてわたしには受け取れた．組織の中での評価というのは，一体だれが下しているのだろうか？　日本の教育は団体行動をよしとしており，ある枠組の中からでると，異端とみられる．その教育成果が会社という団体でも十二分に反映している．個性を尊重するのがいいのか，というといちがいにはいえない．これはどうも文化論になってしまうので，別の機会にでも考えてみたい．

また，障害にめげないというのも重要ではないだろうか．なぜなら，世の中うまくいくことよりも，そうでないことの方が多いようである．だとしたら，いちいちめげていたら何もできなくなってしまう．めげても，次の日は元気，というのがいいようだ．気分転換のうまさ，というのも才能の一種だろう』

感激することが必要です．社内で NEWS のプロトタイプをみせたとき，何人かの若いエンジニアを除いて，ほとんどすべての人が無反応でした．にもかかわらず，私が製品開発を進めようと決意したのは，日本の代表的な Unix ユーザの方々が，試作の初期の段階のマシンを見て，「これはすごい．商品化されたら絶対うちで買うから」といって，はげましてくれました．こういう感

激が必要です．一流の人物というのはまず感激そして，その感激を交換しながら，お互いに夢を語れるんですね．一方，感激できない人は，人の夢をつぶすことで人生を送っている（笑い）．

子どものうちはなににでも感激できるのですが，大人になるに従ってものごとに感激できなくなってくるようです．感激できない人を技術オジンといいます（笑い）．

管理者が，人間に対するセンスティビティを失ったら，もうおしまいです．成功するためには，基本的には任せるべき人に仕事を任せ，任せるべきでない人には仕事を任せない，ということがいえます．それを決めるのはマネージャの仕事・役割で，これは非常に重要だと思います．

『感激する心というのはたしかに大切です．ある年令に達すると，ただ感激しにくくなるだけでなく，それを表現することが恥ずかしい，みっともないと思うようになる．しかし，歳をとると涙もろくなるというの現象は，感激とは関係がないようです』

３．３　よいチームを作るためには

よいチームとは何か？　よいチームではリーダとメンバの区別が年功序列でなく，自然になんとなく決ってしまう．ですから，このメンバでチームを作ったら，この人間がリーダシップをとってくれると，最初から予想がつくようなチーム構成をした方がうまくいくようです．

特殊な才能を持った人間は，チームに１～２人いればいいのですが，その他のメンバもある一定以上のレベルが必要です．たとえば，戦略をたてる力がなくても戦略を理解できる，感激する心，頭が柔らかさ（非常に重要），積極性，好奇心などです．

専門分野が偏ってしまうのは当然よくありませんが，プロジェクトで必要とされる専門家が，すべて必要かというとそうではなくて，足りないものはみんなの力で何とか補えるものです．そのためにも，専門分野が適度に分散していることが必要です．

会議でいつも真剣に議論したりとか，感情的なもつれが尾を引いていたりといった調子のプロジェクトは，あまりうまくいったためしがありません．あまり力まないほうがよく，楽観的なほうがよい．あまりにそうした傾向が強過ぎてもダメですが，悲観的な人間では，絶対プロジェクトは成功しません．

また，過激でないとプロジェクトはうまくいきません．技術プロジェクトですから，世の中に対して，ある程度何かを主張するわけです．主張ができるためには，ある程度過激でなくてはいけない．ただし，あまり過激すぎるといけません．目をつりあげてものをいうのは，ダメです．ちょっとおかしないいかたですが，「余裕を持った過激集団」が理想でしょう．

３．４　マネジャは仕事の邪魔をしない

技術開発プロジェクトの場合，いい人材が登用できたら，マネージャはかれらの仕事を邪魔をしてはいけません．せっかくの人材が，余計な口出しのせいで燃えなくなってしまいます．主役はメンバであって，マネージャでありません．ただ，報告は受けるようにします．そして，負け戦の時は経験豊かなマネジャが，敵の正面に出ていくようにします．

『戦いにおいて一番難しいのは，歴史上の例を出すまでもなく，古今東西でも撤退戦ということになっている．最低限の被害で切り抜けるるのが腕の見せどころとなる』

３．５　チームを燃えさせる

人間は命令では燃えません．すぐれた人材であればあるほど，つまり内部に高いエネルギーを持っている人ほど，外からの命令では燃えないのです．逆にいうと，命令を受けてひょいひょいやるような人は，それほどの人材ではない．だから先程いったように，通常の組織では埋もれてしまう．

NEWSを開発した人たちは，売れるものを作ろうなどとは考えていなくて，自分たちにとってほしいと思うやつを作っただけなのです．社内のVAXがパンク状態にあり，それを逃れるためにはこんなマシンが欲しいというのが，かれらにとっての第１のモティベーションでした．一般に，技術者は，好きなマシンを作っていいよというかたちで，知的に挑発すると燃えるんですね．そのためには，かれらが心のなかで何を考えているかを知る必要がある．最初にビジネス・プランがあってプロジェクトが始まるわけでなく，エンジニアたちが内心やりたいと思っている核心を突いて，こんなテーマがあるがむずかしそうだネとそそのかす．それをビジネスに持っていくのは，もちろんマネージャの仕事ですね．

阿呆のいないチームにすることも必要です．ここでの阿呆とは，ふつうの組織で高く評価されている常識人の

ことで，このような人間がいると，せっかくチーム内に燃え上がった火が消えてしまいます．感情的トラブルがある２人を同じチームに入れてはいけない．抗争は，感情的なものから起きることが多く，これがあるとチームは燃えませんね．感情的な抗争がでてきたら，どっちか１人を外してあげないとダメです．これもまたマネージャの仕事で，早く察知してあげる必要があります．

技術開発プロジェクトは，ある程度の情熱がなければできません．燃えないチームに長いこといると，どんなにいい素質を持っていてもエンジニアはダメになる．逆に燃えるチームにいると人材はますます磨きがかかってきます．外からの干渉がはいってくると，チームの火は消えてしまうので，そうした雑音を排除するのも，マネージャの役目です．なるべく，チームが自由に働けるようにしてやらないといけません．

３．６　周囲の冷たい目や障害にめげない

研究開発のプロジェクトに対して周囲の目は，一般に冷たいものだということを最初から覚悟しておくと，マネージャは，比較的冷静に毎日を過ごせるんじゃないかと思います．なぜ周囲が冷たいかといいますと，技術オジンがいるからなんですね（笑い）．技術オジンとはなにものかを定義をする前に，中年の定義をします．中年とは：

- 絶対に食い物を残さない（笑い）．
- トイレにはいると，ちゃんとしたポジションに立たずにチャックをあける．終わるとチャックをしめながら歩き出す（笑い）．その話を聞いて，私も品川駅のトイレで観察してみたのですが，本当にそうなんですね（笑い）

こういった人物像です．

次にオジンの定義ですが：

- 可愛い女の子を前にしても感激しない

『オジンといわれないために，レポータの推薦する可愛い女の子をひとりあげてみますと，この講演が行なわれた時点では，後久美でしょう．まさか，究極の美少女を未だに知らない人がいるとは思えませんが，ＳＥＡ会員の平均年齢はかなりたかいので，もしかしたら知らないかもいしれません．知らないあなた！感激をしなかったら間違いなくオジンですよ』

さて，そこでいよいよ技術オジンの条件はといえば，次の通です：

- 社内の新技術には感激しない
- 自分は新技術を製造しない
- 他社の新技術には過剰に反応する

世の中のほとんどの技術管理者が，こういった技術オジンとして分類できると考えられます．したがって，社内で新しい技術をやっている人間が，組織のなかで虐げられ，阻害されてしまうのは当りまえです．人間の目には，一般的に身近なものが悪く見えて，遠いものほどよく見えるということが，よくいわれています．これは日本固有のことではなく，アメリカでも，ディジタル・オーディオの時にいやというほど経験しました．ですから，世の中で，自然に新技術が認められるということは，絶対にありません．技術者はつねに戦って，戦いつづけて，自分の考えを世の中に広めていくしかないのです．

３．７　歴史が証明してくれる

私は，自分ではそんなつもりはないのですが，会社のなかでよくケンカ（もちろん技術上の）をします．日本の会社では「和」というものを大切にする習慣があるので，よく批判されます．みなさんも，御自分の周囲を見ればおわかりになると思いますが，世の中にインパクトを与えるようないい仕事をしている人というのは，人の和の中に生きているというより，むしろ，自分の意見を押し通しているのではないかと思います．周囲の冷たい目にめげて，障害を打ち破る力を失っては，いくら技術的に優れていても，だめだと思います．

小さなベンチャー企業でしたら別ですが，ほとんどの日本の企業では，多かれ少なかれ大企業シンドロームにかかっています．人間の行動や思想を誰が評価するかというと，それは歴史の役割です．自分のやったことが，技術の歴史の中でどういった評価を受けるか，それを第一義に考えて行動することが非常に重要だと思います．こういった考え方をもっていれば，大企業シンドロームの中でもいい仕事をできるでしょう．周りの（部長や社長）の評価を気にしてしまいますと，自分が大企業シンドロームの片棒をかついでしまうことになります．

こういった生き方がいいかどうかはわかりません．ある意味で人に逆らってやっていくわけですから，心のよりどころみたいなものが必要なわけです．そんな時に，業界全体の発展だとか，歴史の評価ということを考えるということが必要なのです．

4．質疑応答

Q： 人材は埋もれているといいましたが，人材を見分ける方法があれば．

A： 一般的は方法はないのでないかと思います．ただ職場で接しているだけだと判らないですね．どこか別のところでリラックスして接すると，これは人材だなとピンとくることがあります．不良社員になっていますので，職場でだけみていると，不良社員そのものですから判りませんね．職場では心を開いて自分の考えは述べないですね．

NEWSの時でいえば，コンピュータ関連のゼミを社内でやりましたが，その中で中心になった人物をピックアップするということをやりましたね．ですから通常の業務以外で接する必要があると思いますね．

Q： 戦略のいかし方がありましたが，いい戦略の立て方があれば．

A： 戦略というと紙にかいた戦略と思うかもしてませんが，あらゆることが時間と共に周辺の環境が動いていくわけでして，ポイントが多くの場合言葉で表現できない．戦略とは何かということを言葉で表現できないことが，言葉で表現できないと同じです．強いていえば，直感とひらめきではないかと思います．紙にかいた戦略というのは，動きを反映できませんから，どうしても表面上のものになってしまうと思いますね．

Q： 人材を集めるという話しがありましたが，そのなかで人材でない人を外さないと，チームが燃えないということがあったと思います．しかし，通常の会社でいきますと，会社から与えられた人間で構成するということで行ないます．ですから，人材でない人をチームから外すことをしますと，結局その課の中で面倒をみないといけないんですね．人材を集めるということと，いらなくなった人をどうするかということを，お聞きしたいのですが．

A： 一つの仕事をやるときに，それが開発プロジェクトだとしますと，会社が人をあてがってきて，その人を運営しないといけないということになりますと，その会社はあきらかに大企業シンドロームに陥っています．つまり，それでは開発プロジェクトはできないと思いますね．

会社があてがうということは，人事の担当者が見て，つまり面接とかペーパの出来とかといったことであてがってくるわけです．人事の人がプロジェクトの内容が判るわけがないんでして，本当の意味での開発プロジェクトはこなせないんじゃないかとお思いますね．もうちょっとルーチン・ワーク的な仕事でしたらいいんでしょうが．

私はあてがわれた人でプロジェクトを運営したことがないので，あまり適切なことはいえないかも知れないのですが，外された人をどうするかというのは，人事課の人が考えることであって，開発プロジェクトの人間が悩む必要はないんじゃないか，と思います．

Q： チームを燃えさせるために，マネージャがキャッチフレーズを作るということはされたのでしょうか？

A： いや，キャッチフレーズなんかでは，エンジニアは燃えないと思いますね．ただ，キャッチフレーズは必要です．というのは，何か事にあたって判断するときに，自分たちが何をやっているか，ということを常に意識しておくためです．別のいい方をすれば，何らかの大義名分が絶対に必要だと思います．ただ大義名分ではエンジニアは燃えない．大義名分の必要性を話し出すと，それだけで１時間以上かかってしまうので，これはまた別の機会にしたいと思います．

5．終りにかえて

かなり過激な話しがいくつかあったと思いますが，それが過激に聞こえるのは，同じように考えているけれど，大企業シンドロームに浸っているために，正しいことを正しいといえなくなってしまっているのではないだろうか．人事の人には（またはそれに類する人に）プロジェクトの中身が判らない．従って，判らない人の判断を組織の判断として受け入れるのは，プロジェクトをまずくするものだ，というのは実に正鵠を得た発言だとおもう．

冒頭でも述べたように，わたしはマネージャではないので，土井さんの話しから，プロジェクトを成功させるための人材の発掘法より，自分の能力を利用してくれるマネージャの探し方をどうしたらいいか，いくらかのヒントをつかめたような気がしました．

SEA 春のセミナー・ウィーク
セッション C4

プログラミング能力の定量的評価

講　師：　鳥居　宏次（大阪大学）
報告者：　芝原　雄二（沖ソフトウェア）

１．はじめに

セミナーは，突然，α波の話しからはじまった．

ある学生がいうには，講師の鳥居先生の声には，α波が交じっているという．どういうことかといえば，講義の内容がはっきり聞き取れず，眠気をもよおす．これは，先生の声に眠気をもよおさせるα波が交じっているためだという．

こんな冗談から始まった昼食後のセッションではあったが，私にはα波の影響はまったくなかった．講義の内容が非常に興味深かかったためであるが，それだけに読者の方々にその内容が伝わるかが心配でもある．幸いなことに，このレポートの中でも触れているが，講義で話されたことのいくつかは論文で発表されているので，是非そちらを読んでいただきたい．

これは余談だが，セミナーが終ったあとに事務局の方に聞いたのだが，今回のセミナー・ウィークでこのセッションの参加者が一番多かったそうである．世間の関心度のあらわれといえようか．

２．各種の実験

（１）概念把握能力の実験

この実験は，設計能力に焦点をあてており，学生が，プログラミング課題をどのように分析し把握できるかを，初期基本設計のでき具合と，ソースコードの良し悪しの関係からみようとしている．

課題は，昭和５９年９月号の情報処理学会誌に掲載された酒屋の在庫管理の問題である．披験者は，構造化プログラミング，プログラム書法の講義と演習を終えている情報工学科の２年生，３７名である．

まず，１時間で課題の初期基本設計を行ない，時間を経て見直しを行なった後に，２週間をかけて Pascal でプログラミングを行う，というものである．この結果の詳細については，情報処理学会論文集（６１年前半）に論文としてまとめられているのでそちらを見ていただければと思う．

この実験で私が興味深いと思ったことは，「はじめよければ終りよし」という一応の結果がでたということである．つまり，初期基本設計をしっかりやっている人は，バグもなく，ソースコードの評価点もたかく，工数も少なくなっている．おもしろいのは，バグを出している人が，初期基本設計での評価点で中間点に位置する人達であり，可もなく不可でもないといった人達であるということである．初期基本設計で評価点の低い人達は，作業時間をさいているが（つまり工数をかけているが），バグは出していないということである．

われわれのソフトウェア開発現場でも，基本設計段階であいまいにしておいたものが，おうおうにしてバグの原因になっている．設計段階で，わかっていること，わかっていないことをしっかり区別しておくことによって，わからなかったことに対しては，あとで時間をかけて解決できるものである．もっとも納期などの時間に追われて，おろそかにしてしまうという，毎度の失敗があるが．

（２）複合設計法の採用による実験

モジュールに分解する過程と，最終ソースコードにおいて，どのような差異が認められるかという実験である．第１段階では，設計法に関する知識を与えない状態で自己流にプログラムを作成し，第２段階では，各種の設計法を教授した後で，複合設計法でプログラムを作成する．

披験者は，構造化プログラミングやプログラム書法程度の講義と演習を終えている情報工学科の１３名の２年生である．この内７名に関しては，複合設計法を用いて作成したとは考えられないもの，作成したが解析困難なもの，動作確認でバグが発見されたものであり，サンプルからは除外している．また，ここでも Pascal を用いているため，内部手続きはモジュールとして取り扱っている．そのため，モジュール強度，モジュール間結合度という評価基準を採用している．この詳細についても論

文にまとめられているので，情報処理学会論文集（６１年前半）を見ていただければと思う．

私として興味深かったことは，行数およびモジュールの分割数では設計法の方が多くなっていることと，実行速度と作業工数では差がないことであった．ただし，この工数のなかには，設計法を理解するための時間が含まれていると思われると述べられていた．つまり，技法を収得していれば，作業工数は少なくなるということである．

ここで問題定義をしてみると，こうした技法を用いて作成したものが，はたして本当に他人にとって見やすいものであり，メンテナンスをしやすいかということである．すくなくとも，その技法を知らない人達にとっては，その技法を収得するのに時間がかかるし，まして正しく理解しているという保証はない．これは技法なり方法論を否定しているのでなく，ソフトウェアの開発現場では，技術レベルが極端に違っていることが非常に多いのである．エンジニアだけでなく，エンド・ユーザ，営業マンといったさまざまな人が，実際のプロジェクトには拘ってくる．技術移転といった場合，単にエンジニア間だけのこととするのは，問題の解決にはなりえないのではないかと思う．

（3）プログラム設計過程での定量化

ここでの実験は，ある学生が設計したものを，他の学生がコーディングをするというものである．この実験は，われわれがソフトウェア開発の現場で，実際に行なっている作業方法に類似したものであり，その結果は非常に興味深いものがあった．この詳細についても，情報処理学会論文集（６２年前半）を見ていただければと思う．

この実験で得られた結果のひとつとして，モジュール結合度に関する評価の高いプログラムを作成したチームのものは，バグが少ないということである．その理由としては，モジュール間のデータの受渡しを，設計過程で明確にしておくことがあげられている．つまり，与えられた仕様書の質によって，プログラムの質が大きく依存するという傾向があることになる．しかし，与えられた設計書が悪いと評価されたものであっても，コーダにモジュール化の能力がある人があたると，バグの少ないプログラムができる，という例外もあった．

（4）実験結果のまとめ

これら一連の実験を通して，プログラム作成能力を評価する尺度の提案をおこなっている．

設計に誤りがなく，コーディング，デバックを効率よくおこなえることは，プログラム作成能力のひとつとして考えることができ，このプログラム作成能力は，プログラムの作成過程で作り込まれたエラーを調べることによって定量化できると考えられる．そして，それを評価する尺度が以下のように提案されている．

$$\frac{1}{C}=\frac{\Sigma TE}{F(p)}$$

TE：エラーＥの寿命
p：与えられた問題の難しさ
F：正規化関数
C：プログラム作成能力

３．評価尺度の適用実験

設定した評価尺度を，実際のプログラム作成過程において適用した実験結果が次に示された．この実験は，C, Pascal または PL/I のサブセットのコンパイラ（約2000行）を，C, Pascal を使用して作成させて，プログラム作成能力「Ｃ」を求める実験である．

この実験の結果として，いくつかの関係が示された．まず，「Ｃ」の値の高い学生は，設計，コーディング，タイプイン，テスト，デバックという作業工程を明確に分けており，設計，コーディングを机上で充分に行なっているために，端末使用時間が非常に短い．これに対して，「Ｃ」の値の小さい学生は，コンパイラに対する理解度が低く，段階的なコンパイルとテストを行なっていない．

プログラム作成能力「Ｃ」をみる実験として，異なるプログラムを作成させ，その個人の「Ｃ」を求める実験もしているが，この実験の結果はほとんど差がでなかったそうである．

これらの実験ですべてを見ようとしているのではなく，講師の鳥居先生も，この程度のデータ量では結論を出すものでなく，とりあえず報告を，ということであった．

プログラム作成能力「Ｃ」に関しては，当日のテキストに詳しく述べられているので，興味のあるかたは，ＳＥＡの事務所に問い合わせれば，在庫があるかぎり（実費で）分けてくれるはずである．

さて，このいくつかの実験を聴いて，私が思ったことがいくつかある．そのひとつとして，プログラム作成能力「C」が，今後確実に使えるメドをできるだけはやく作ってくれたら，ということである．これによって，プログラマの評価のときに，言語経験何年などというほとんどあてにならない評価基準を無視することができそうだからである．

そしてもうひとつ思ったことは，自分も学生時代にこういう実験に拘れたら，ということであった．たとえ披験者のひとりであったとしても，ソフトウェア工学の研究に関係できることの意義は大きいと思う．われわれが開発現場での諸問題を解決しようとするとき，こうしたソフトウェア工学によるアプローチが採用できない理由の1つに，どうやっていいのか見当がつかないということがある．そうした意味においても，今回の講演は極めて有意義なものであった．

4．私感として

以下にこのセッションをレポートするにあたって，私なりに考えたことをまとめてみたいと思う．

ソフトウェア工学の権威であり，COCOMOの生みの親でもあるB. Bohemによると，生産性コストに影響を与える人間的要因として，次のものをあげている．

- 分析チームの能力
- 対象アプリケーション・システムの経験
- プログラム能力
- 開発環境での作業経験
- プログラミング言語の経験

さらに，能力が高いほどエラー発生が少なく，また経験があるほどエラーが少ないといっている．このことは，エラーの発生率が低いほど，プログラミング能力が優れているということになり，鳥居先生の実験とも共通するものがあるようである．

今回紹介されたいくつかの実験は，個人の能力評価に関するものであったが，チームで行なう作業が多い現状において，個人能力とチームでの能力ということも実験していただけたらと思う．このチームでの能力ということに関して，G.Weinbergは，「プログラマの能力が同じ場合には，3人のプログラマのチームでは，1人のプログラマの作業量の2倍しかできない」と述べている（もっとも，おなじ能力のプログラマが3人もそろうことというのはあるのだろうか）．

人間はさまざまな経験を得ることによって，その能力が向上するという．また，経験は教育によって与えることもできるという．しかし，おなじ事を教えたつもりでも，いざ仕事になると，教えた通りをできるかといえば，そんなことが少ないことは，多くの方が経験していることではなかろうか．

鳥居先生の講義の中で，概念把握能力の実験において，評価点の高い学生はあらためて教育の必要がないほどその能力があるとおっしゃていたが，彼らはどこでその能力を得たのだろうか．自ら学んだのだろうか．かって山本五十六は，「やってみせ，説いてきかせて，させてみて，誉めてやらせねば人は動かじ」といったという．実に人はさまざまである．

5．終わりにかえて

ソフトウェアの世界で過ごしていると，定量化できない様々なことにぶつかる．これは現状の研究段階ではできなくて，将来的にはできるようになるは判らないが，プログラミング能力に関していえば，定量化ができるのではないかと思っている．

人間的要因によって大きく左右されるという属人性があるからこそソフトウェアであり，その宿命は，おそらく当分はなくならないのではなかろうか．しかし，この属人性が定量化の大きな壁となっているが，自分のやっている仕事を証明するためにも，尺度ということに，もっと真剣に取り組むべきではなかろうか．

講義内容をかなり自分の趣味に近付けてレポートにしてしまったが，意見等があれば，管理分科会でのテーマにしてみたいので，出席してみてください．また，出席できない方で意見があれば，ＳＥＡＭＡＩＬ誌上に発表してください．さっそく管理分科会でとりあげますので．

ICSE の感想

鳥居　宏次
大阪大学

すでに昨年秋のプログラム委員会に参加して，発表される論文の見当はついていた．また，東京で第６回（１９７９年）が開催されて以来，プログラム委員会なるものに連続携わり，第８回のロンドン大会以外は，本番にも参加してきた．こういう個人的な「慣れ」による余裕がでてきたのも原因だろうが，今回の会議では，多くの日本からの仲間も含めて知った顔も多く，比較的のんびり参加でき，それなりに納得できる会合だった．

日本人として，岸田さんがプログラム・チェアマンという大役を見事に果たされたことに敬意の念を持ちつつも，日本人の１人として誇りを感じている．その理由はプレナリー・セッションでの司会を無難にこなしたり，最優秀論文者に渡す表彰状が間に合わなくて，白紙の表彰状をそしらぬ顔で渡したということではなく，３日目の ICSE の世界での重鎮たちの「顔見世パネル」での活躍に感心したのだ．格好ばかりつけるのではなく，控え目でいて，それでいて面白い話は，態度のデカイなまじっかな「毛唐」なんかくそくらえといった印象で，大いに気分をよくさせていただけた．

少し真面目な話題に移ろう．プレナリー・セッションについては，特に１日目は儀式的になりやすい雰囲気なのに３者３様の議論で興味を引いた．本来ならば，Osterweil 流の，ソフトウェア開発工程を program として記述しようとする提案を，具体的には procedural な記述で説明すること，および process programing としてのねらいは同じでも Lehman 流の「初めにモデルありき」にもとづいてやろうとすること，の２人の考えを聞かされることになると思っていた．そこに３人目として，フロアーから，「小生は第１回目の ICSE に当たる会合の General Chaiman をやったものだが」といって，Harlan Mills 先生が立ち上がった．数理的接近とでもいうのか，最近の ICSE 自身が formal な取り扱いに欠けているというような事をいいだした． Osterweil 先生は Over-Mathematical になることを懸念していると反発をしていた．３人ともいいたいことを言ったのでそれで十分なのだろう．しかし，それぞれ正しい事をいっているので敬意を表して聞いていた．（Mills 先生は翌日の Measurement の Session-3でも Discussants として自己流の長い話をした．ストレスがたまっていて，いわゆるガス抜きのチャンスが必要なのだろうと想像はできた）．

後日談として，Osterweil 先生が来日され，話を聞く会機に恵まれたが，彼の Process Programming に対して英語圏の人達の反応が割れていたとのことだった．Lehman 先生も「seductive but, ...」といういいかただったとのことで， Osterweil 先生も少し浮かぬ顔をしていた．

あと１つ，たまたま（SEA の代表選手？）野村さんの論文の Session を司会したので，その様子について，Best paper awards が２件に与えられたが，その１件がこの session の最後４人目で， parallel に走る system を prototype として実現できる環境を作ったという発表だった．１人目の発表が４つの設計方法論の比較についてなので，質問も多く興味を持つものも多かった．その間に野村さんのを含めて２件の論文が発表されたわけで，聴衆は４００人を超えていた（司会者は暇なものですることがない．それでいて会場はよく見える）．野村さんの発表は，日本での調査結果を論文にまとめたものだが，国内の調査結果もこういう形で外部発表されれば，内外を通じて利用の度合いも上がろう．

今回の ICSE での論文の受理された率が，全体でも，日本人のだけでも約９分の１だから，会議のレベルの高さと日本からの質の高さにも感心すべきなのだろう．日本からは，２人の司会者はともかくとして，パネリストとして４人も活躍し，参加者が B747 ジェット１機分（？）ということで大いに頑張ったといえるだろう．次回以降もこの勢いを続けるために，遠慮せず自分達の成果を論文の形で出される習慣を期待したい．

9th ICSE に参加して

玉井　哲雄
三菱総合研究所

１．はじめに

第９回ソフトウェア工学国際会議（略称 ICSE）は，３月３０日－４月２日に米国 California 州 Monterey で開かれた．雨季の明けた California は快適な気候で，海岸線の美しい Monterey をさらに魅力的にしていた．それも手伝ってか，参加者は１２００人を数える盛況で，チュートリアルに６００人もの参加があったと，主催者側は喜んでいた．日本からの参加も多く，おそらく１００人近くに達しと思われる．

今回の実行委員長は，W.Riddle，プログラム委員長は R.Balzer とわが岸田孝一氏であった．プログラム委員は他に３３名の多きを数えた．うち日本からは，阪大の鳥居先生，慶大の斎藤（信）先生，さらに筆者もその末席に連なった．

２．基調テーマ

今回のプログラムは，基調となるテーマを明確に打ちだしたところに大きな特徴がある．そのテーマとは，「ソフトウェア・プロセスの形式化と自動化」である．

技術セッションの初日（３／３１）に，Balzer がその趣旨を説明した．いわく，「われわれは６０年代，７０年代にいわれたソフトウェア危機をすでに解決した．すなわち，期限に合わせて定められた機能のソフトウェアを作ることは，できるようになった」．

ここで会場から思わず笑いがもれる．

「しかしソフトウェアの保守と進化の過程でのソフトウェア危機は，まだ解決されていない．そのために，ソフトウェアを従来のように製品（プロダクト）としてとらえるのではなく，プロセスとしてとらえることが，ますます重要になる．プログラム自身を保守するのではなく，プロセスを保守することにより，柔軟な機能拡張や再利用ができる．ウォータフォール・モデルは死んだ．数学的な方法論から設計方法論へ，すなわち多人数による長期間の開発に向いた方法への，転換が行われなければならない」．

まさに，Balzer らしいアジテーションだった．

プログラムの構成の上では，さらに次のような特徴があった：

- このテーマの趣旨に沿って，セッションをプロセス，形式化，自動化の３つの枠にまとめたこと．
- 関連する技術として，人工知能とデータベースをとりあげたこと．
- 新規の論文だけでなく，すでに行われたワークショップの報告や，特定分野最新動向といったセッションも設けたこと．
- 投稿論文は，プログラム委員の少なくともひとりが推薦するという条件をつけ，厳選したこと．

この基調テーマを強調する形で最初に行われたのが，L.Osterweil の「ソフトウェア・プロセスもまたソフトウェアだ（"Software Process Are Software Too"）」と題する招待講演．筆者の知る限り，これまでの ICSE の招待講演は，外部の分野から講演者を文字通り招いて行われるのが普通で，会議の基調講演という性格のものはあまり例がなかったのではないかと思う．一昨年の第８回（London）では，まったくの外部ではないが，ソフトウェア工学とは分野を異にする D.Scott が招待講演を行った．San Diego（第５回）では N.Wirth の講演もあったが，一方でノーベル賞受賞者による分子生物学の話もあった．

さて，Osterweil の講演だが，その要点を示せば次のようになろう．

ソフトウェア・プロセスにとって，プロセスの手順をなるべく厳密に記述すること，そしてその記述を個々のケースに適用して実行できるようにすることが，重要である．このためには，プログラミング技術を応用し，プロセスの記述もソフトウェアとして扱えるようにすればよい．Osterweil はこの考え方に基づいたソフトウェア開発のパラダイムを提唱し，さらにテスティングの簡単なプロセスについて，プロセス・プログラミングの例を示した．

この講演に対し，M.Lehman がコメンテータとして意見を述べた．「ソフトウェア・プロセスのうちで，ア

ルゴリズム的な記述に向くステップはごく一部である．多くのプロセスは予測しがたい性質をもち，そこに人間の工夫が要請される．プロセス・プログラミングという概念は，ソフトウェア・プロセスが決定論的（deterministic）であるかのような誤解を与える」という，かなりもっともな批判である．

Osterweil は，「プロセス・プログラミングの例として，現在は手続き的な記述をとっているが，これが最善と主張している訳ではない．ルールなどによる宣言的な記述が向く場合もあろう」などと答えた．

会場からの質問を受けたところ，最初に立ったのが H.Mills であった．彼の質問は，いきなり「この会場にチューリング賞受賞者がいたら手を挙げてほしい」という唐突な問いから始まった．該当者がいないことを確認したうえで，「ＩＣＳＥからチューリング賞クラスの知的指導者がなぜか排除されてきている．当初は Dijkstra, Hoare, Gries 等がいたのに，なんらかの社会的プロセスによって，この会議は反知的な方向に進んできてしまったのではないか．もうかれらから学ぶことはないのか」という強烈なパンチを浴びせたのである．会場にたちまち緊張が走る．

Osterweil が多少の動揺を見せながらも，さすがにこうした議論に慣れたアメリカ人らしく，弁じたてた．しかし，Mills の批判は会議の運営に対してむけられたものだから，かみあうはずもない．もともと Osterweil が答える必要はなかったのである．Mills も，これは主催者にいっているのだ，というようなことを付け加えた．とにかくこのやりとりから，米国のソフトウェアの学界にも，こみいった勢力争いがありそうなことが，想像された．

３．形式的仕様言語

続いて一般セッションの報告に入る．まず，セッション２「形式的仕様言語」に出た．最初の発表は，E.Berliner と P.Zave の共同論文．Zave が開発した，とくに実時間分散処理の記述に向く動作的な（operational）仕様記述言語 PAISLey を，海中光ケーブル通信の制御システムの要求記述という実際的な問題に適用した，その体験報告である．筆者は去年の１０月に今回の会議への投稿論文を実に１６編査読させられたが，この論文はその中の１編で，なかでは一番いいと思い，ただひとつ合格点を与えた．技術移転というあまり論文にしにくいテーマであるが，プレゼンテーションが非常にうまい．

ただし，実時間システムの記述に向くという PAISLey の特徴が活かされた対象でないことから（この点についても発表者は誠実に報告したが），技術的にはやや物足りない感もあった．後で Zave に，このプロジェクトの結果から PAISLey の言語仕様を見直す必要は起こらなかったかと聞いたら，問題点はあったが PAISLey にはもうずいぶん長くかかずりあっているので，いいかげんに次のことをやりたいため，言語をいじるようなことをしていない，といっていた．

同セッションの次の発表は，電総研の二木さんたちの論文だった．二木さんは１９８４年に SRI に客員研究員として行かれ，J.Goguen 等と一緒に仕事をされたが，この論文は，電総研と SRI の共同によるその後の研究成果である代数的仕様記述システム OBJ2 に組み入れられたパラメータ化プログラミングの考え方を述べたもの．この考え方を，簡単な例で示せば次のようになる．

ソートという概念を考えよう．OBJ2 では，これを object として定義できる．ソートの対象となるデータは，もっと一般的には全順序の性質を持てばよい．全順序集合は，theory として定義ができる．ソートに対し，全順序集合はパラメータとして働く．たとえば整数を全順序集合としてとり，ソートのパラメータをこれで実体化（instantiate）すれば，整数のソート・モジュールが得られる．整数を全順序集合とみなすことを正当化する仕組みとして，OBJ2 には view という構成要素がある．

次は，D.E.Perry（ベル研）による "Software Interconnection Models"．これは，優秀論文として表彰された２編のうちの１つである．

内容は大きく２つの部分に分かれる．前半は，ソフトウェアの育成（evolution の試し訳．「進化」ではピンとこないだろう）モデルとして，きわめて一般性の高い相互結合（interconnection）モデル（IM）なるものを提唱する．既存のいくつかのモデルを，ＩＭをさらに分類した二つのモデル，すなわち単位（unit）ＩＭと統語的（syntactic）ＩＭとして，説明する．後半では，第３のＩＭとして意味的（semantic）ＩＭと呼ぶものを提出し，その具体的な方法を示す．

こういう概念的な整理は日本人にはあまり得意でないが，欧米では評価されるようだ．

4．ソフトウェア・プロセスの実験的研究

こんなペースで書いていては長くなるばかりなので，以降は，かなりはしょって紹介する．１日目のセッションで他に比較的面白かったのは，「ソフトウェア・プロセスの実験的研究」と名づけられたパネルである．座長は MCC の W.Curtis，パネリストは同じく MCC の H.Krasner，Yale 大学の D.Littman，Stanford 大学兼 Xerox Parc の J.Tang の３人．パネルといっても，各パネリストによる発表が中心で，一般のセッションとあまり変わらない．

Krasner の発表は，実験的なソフトウェア・プロセスのあり方を探るために，大規模システム開発プロジェクトの事例を集め，実験的な分析を行なった結果の報告である．事例として MCC のスポンサー会社の１９のプロジェクトをとり，インタビュー調査を行なった．その結果として，従来のプロセス・モデルでは重要視されていないチーム・ワーク，インフォーマルなコミュニケーションを通じての設計状況に関する共通認識，学習，交渉，顧客との意思疎通，といったプロセスの重要性が明らかになったとしている．

Tang の発表は，グループによる概念設計の過程を観察することにより，どのような要因が意思決定に寄与し，設計作業の進行につながるかについての，調査報告である．ジェスチャがよく使われていること，黒板のようなものより共通の作業机のほうが効率的であること，などの話が印象に残っている．また，Littman の話は，スペース・シャトル関係のソフトウェア開発を題材に，そこで行なわれた査閲（inspection）をビデオに撮って分析した結果の報告であった．

こういうパネルをみると，米国でもグループによる意思決定プロセスに注目しだしているという印象を受ける．日本では，要求分析のフェーズでよく行なわれる合宿討議などで，こういった問題をかなり取り上げてきているところが多いのではないだろうか．Krasnerとは，翌日，水族館（！）で開かれたパーティでいろいろ話をする機会があった．MCC はいわば寄り合い所帯だから，とくにチームワークといった問題意識が強いかもしれないが，かれの話では，米国一般にそういう点への注目が高まっているという．

5．形式的な方法の実践的適用について

形式主義（formalism）を扱ったセッションがかなりあったので，それらをまとめて紹介しよう．筆者が聞いたのは，以下の３つである：

- 招待講演：D.Bjorner，「ソフトウェア開発における形式主義の利用：プログラムとメタプログラム」
- セッション 12（パネル）：「変換技法の最先端」（座長：W.Scherlis，パネリスト：D.Bjorner，M.Feather）
- セッション 15（パネル）：「形式的仕様を日常的に用いること」（座長：P.Zave，パネリスト：D.Good，C.Jones，M.Meller-Smith，W.Scacchi，I.Sorensen）

他に，V.Stenning（Imperial Software Technology）の招待講演「環境の役割（プログラムでは「ソフトウェア・プロセスの自動化」という題になっていたもの）」，およびD.Barstow のチュートリアル的な発表の後に，S.Fickas（プログラムでは C.Williams になっていたが交代）と B.Boehm がコメントするような形式のパネル「人工知能とソフトウェア工学」も関連が深い．

Bjorner の講演や Zave のパネルから感じたことは，形式主義の実践的利用については，米国よりヨーロッパの方が進んでいるという印象である．Bjorner や Jones のウィーン技法（Vienna Software Development Method，略して VDM）が当人達から紹介されたこともあり，米国では理論と実践が分離しているのに，VDM は，米国や日本ではまだなじみがうすいだろう．

Scherisのパネルは，概念論に終始して面白くなかった．Scheris とは，後に打ち上げパーティでゆっくり話しをする機会があった．彼と Scott の推論プログラミング（inferencial programming）に関する論文は，日本でよく読まれていることをいい，その後進展があるかどうか尋ねた．Scherlis によれば，大いに進んでいるという．数編の論文をその後書いているから送るといっていたが，未だに届いていない．

Barstow の話しもよく整理されているが，あまり新味はなかった．Boehm が，今のＡＩブームを多少皮肉って，現在はやりのシステム開発手順をフローチャートで示した．それによると，まず問題が難しいかと聞き，易しければ従来型のシステム開発を行なう．難しければ，ＡＩでやることにして Lisp マシンを買ったり，ＡＩセミナーに人をやったりしたうえで，知識工学アプローチによる開発を試みる．うまくいけばそれでよし，だめなら従来の開発に戻るというものである．

6．日本からの発表

二木さんからの発表についてはすでに述べたが，他にも日本人の発表がいくつかあった．寺本さん（日本電気）は，セッション８「信頼できるソフトウェアの作成プロセス」というパネルで，「ソフトウェア開発の各プロセスの担当者が，そこで発生の予想されるエラーの数を見積り，前のプロセスから持ち越している想定されるエラーの数を加え，そこから実際にそのプロセスで発見したエラーの数を差し引いて，それを次のプロセスに渡す．最終結果と見積りのずれを各担当者にフィードバックして，見積りの精度を上げるとともに，エラー発見の率を高める」というきわめて実践的な品質管理の方法と実践結果を発表した．このパネルの他の二人（N.Levenson と P.Neuman）が，安全なソフトウェアを作ることがいかに難しいかという観念論ばかり述べて，聴衆を辟易させていたのとは対照的でよかった．

論文としては，宮崎・村上両氏（富士通）の「誤差に基づくソフトウェア計量」，野村さん（日本電子計算）の「日本におけるソフトウェア工学ツールの利用状況」があった．パネルには，「既存の環境のサンプリング」に松本さん（東芝）が，「将来の環境動向」に横井さん（ＩＣＯＴ）が，「ソフトウェア・プロセスの改良アプローチ」に岸田さん（ＳＲＡ）が参加した．

7．最終パネル

会議をしめくくるパネルは，Balzer もいったようにテーマとして異質で全体のプログラムからはみだした，知的所有権保護の問題を扱うものであった．演題を，「個人および国際的所有権：問題はいかに深刻か」という．個人と国際をならべたところが不自然であるが，実は前半と後半に分かれていて，前半で３人のパナリストが個人の所有権の問題を論じ，後半ではやはり３人のパネリストが国際問題を論じた．前半の座長を勤めた女性弁護士（名前が判らない）は，なかなかものの分かった人のようだった．「現在，ソフトウェアの著作権に関連した裁判例は多いが，裁判官がソフトウェアについてよく知らないため，ソフトウェア業を保護するつもりで，かえってソフトウェアの流通や標準化の障害となり業界に不利益をもたらすような裁判が出されている．今のうちに，ソフトウェア業界が正しい主張をしておく必要がある」といった指摘をしていた．また，聴衆からの質問のなかに，「東南アジアで著作権を保護していない国があるが，そういうところの海賊版封じにうまい対策があるか？」というのがあったが，それに答える前に，１９世紀の米国は著作権を認めておらず，ヨーロッパのあらゆる本を複製して利用していたという歴史的事実に注意を喚起したことも印象的であった．かなりレベラルな人らしい．

後半の座長は，最近まで SRI の幹部だった D.Brandin である．ACM の会長を勤めたこともあり，日本の先端技術比較のレポートを出しているので，読まれた人も多いだろう．現在は，Strategic Technologys という会社をやっているらしい．後半の議論は，時節がら，やや日本批判の色彩を帯びた．聴衆からヨーロッパの人が一人立って，議論がアメリカ側に偏している穏やかに抗議したが，日本人は自分を含めてとくに発言しなかったのは，やはり反省すべきであろう．

8．ツール・フェア

ツールおよび本の展示が併設されたばかりでなく，新しい試みとして展示中のツールの説明をするセッションが，一般セッションと同時に開催された．

商用ツールに目立つ傾向は，ソフトウェア開発用 CAD と称するような，図式を用いた設計開発支援システムの数が多かったことである．たとえば，Cadre の Technology Teamwork，Chen & Associates の ER-Designer I.G.L. の SPECIF-X，Interactive Development Environments の Software through Picture，Reasoning Systems の Refine など．

日本から，慶応大学，静岡大学，ＳＲＡが参加．いずれもソニーのワークステーションを使って，UNIX 上のツールをみせた．

9．まとめ

ICSE に限らずソフトウェア工学の会議に出るたびに思うことは，ソフトウェア工学は難しいということである．計算機科学とソフトウェア開発の現場の間にあって，技術的にも面白く実践上も役に立つ仕事をすることは，なかなか大変である．今回も，全体に概念的な話が多く，もう少し技術的な問題に特化するか，あるいは実際的な話題に徹するかした方がよいのではなか，と思うことが多かった．

ICSE 印象記

落水 浩一郎
静岡大学

1.はじめに

今回の ICSE には，筆者もツール展示に出品の機会を得て，最近の研究動向の調査もかねて出席した．会議はかなりの盛況であり，内容も充実していたように思う．以下，筆者自身が参加し，興味を覚えた部分を中心に，印象を報告させていただく．

2．概要と特徴

(1) 会議構成上の特徴

- 採録された論文の数を減らした：採択論文数は 22 編，採択率は 11.5 % であった．
- ソフトウェア・ツールの重視：並列 3 トラックの内の１つが，展示ツールのデモンストレーションを伴なう説明にあてられた．
- パネル討論の重視．最近のワークショップのまとめや，ある分野の世界レベルの専門家達の意見交換を目的としたパネル討論のセッションが多かった．セッション数の割合は，論文発表 8，ツール発表 7，パネル討論 12，合計 27であった．
- データベース分野，人工知能分野との積極的交流．

(2) プログラム構成，会議運営上の特徴

- テーマの明確な設定：「ソフトウェアプロセスの形式化と自動化」を取り上げ，プロセス，形式化，自動化を扱う３つの論理トラックを設定した．
- 反応者（respondent）方式の導入：基調講演等の直後に，講演者と同等の専門性を有する人間が，講演者の主張や見解に対して，その問題点を指摘したり，別の角度から光をあてたりして，議論のポイントを浮彫りにする．聴衆にとって，質問の代弁者や問題の理解を助ける役割を果し，効果的であった．

3．チュートリアル

会議前日（３月３０日）の朝９時から夕方の５時まで，並列５テーマのセミナーがあった．

(1) ソフトウェア工学のための DB 技術
(2) ソフトウェア開発環境
(3) 形式仕様と証明
(4) 知識ベース・プログラミング環境
(5) ソフトウェアの再利用

筆者は (4) に参加した．内容は，ＡＩ技術や知識ベースシステムの適用分野についてであり，結論としては，

- Programming-in the-small に対しては，かなりの研究成果があり，生産性向上に関して，２～３年後に実用性が確認され，今世紀末までに広く普及する．
- Programming-in the-large の問題はなかなかむずかしい．研究課題がまだ残されており，９０年代なかばまでに実用性が確認されるものはなく，普及は次の世紀にまたがる．

ということであった．

4．プロセス・プログラミング

会議初日に，「ソフトウェア・プロセスもまたソフトウェアである」と題して，Colorado 大学の L.Osterweil 教授による基調講演があった．

ソフトウェア・プロセスモデル，プロセス・プログラミング等の概念は，８０年代に開発環境を研究していたグループの間から生まれてきたものであり，Arcadia プロジェクトの推進者の１人 Osterweil 教授は，その中心に位置している．「プロセス・プログラミング」とは，その名の示す通り，ソフトウェア開発・進化に内在する諸活動を明示的に書き下そうとする試みであり，定義・設計・実現・テストのすべての開発活動をその対象とする．このようなパラダイム追求の意義は，以下の点にある：

(1) 従来，経験や直観の問題とされてきた，あるプロダクトから別のプロダクトを生成していく問題解決の過程を「もの」として取り扱うことができ，上流工程に対する自動化支援環境の基本論理構造として採用すれば，その実現への手がかりを与えうる．
(2) それに基づいて，プロジェクト管理も，きめこまかく実施できる．

(3) チーム内協同作業が円滑になる.

(4) 再利用の問題に対して，プロセスの再利用（設計上の意志決定の再利用）という攻め口を与えうる.

これに対し，反応者の Lehman 教授（英国のImperial College)は，プロセス・プログラミングが実現できる領域は，きわめて限定された分野にすぎず，その中でしか効果的でないと反論した. Osterweil 氏は当面，domain-specific なプロセス・モデルの構築を試行しており，以下の研究目標を示した：

(1) ソフトウェア・プロセスそのものに対する実験的研究（観測とモデル化）.

(2) プロセス・プログラミング言語の設計：型定義，データ集約化機構，制御構造（選択，反復，並行制御），有効範囲とアクセス規則.

(3) プロセス・プログラミング支援環境のプロトタイプ開発.

(4) 対応するソフトウェア・メトリクスの導入.

(1) に関しては，セッション 5 で W.Curtis 氏が，大規模プロジェクトにおける観測結果をまとめ，source-conceptualizer の存在等，なかなか実感のある結果を報告しており，(2) に関しては，セッション 1 で繰り返し構造に関する討論の結果が報告されている.

操作型アプローチの方は，研究室内における基礎研究，プロトタイプ開発の段階を終え，実用への足固めが試行されはじめている様子がセッション 2 の最初の発表からうかがえ，興味深かった.

5. AI とソフトウェア工学

2日目，筆者は，自分たちが出展したツールの説明に追われて，本会議にはほとんど出席していない．ときどき，展示説明の暇を盗んで会議をのぞいたなかで，セッション 11 で，D.Barstow が，AI 関連技術を応用したプログラミング環境の発展が，ソフトウェア開発のコストに与える影響を，数字をあげて予想していたのが，興味を引いた.

(1) Programinng-in-the-small における自動プログラミングの効果

活動	現在の比率	ファクタ	結果
仕様	.10	2.0	.20
分解	.20	0.1	.02
実現	.20	0.1	.02
最適化	.15	0.1	.01
テスト	.25	0.0	.00
妥当性確認	.10	1.0	.10
全体	1.00		.35

(2) Programinng-in-the-large における自動プログラミングの効果

活動	現在の比率	ファクタ	結果
要求分析	.05	1.0	.05
設計	.10	1.0	.10
コーディング	.15	0.3	.05
統合化	.10	0.5	.05
保守	.60	0.8	.50
全体	1.00		.75

(3) Programinng-in-the-large における設計履歴モデルの効果

活動	現在の比率	ファクタ	結果
要求分析	.05	1.0	.05
設計	.10	0.5	.05
コーディング	.15	1.0	.15
統合化	.10	1.0	.10
保守	.60	0.5	.30
全体	1.00		.65

Programinng-in-the-large の場合，上記 (2) と (3) の相乗効果で，全体の開発工数が現在の 40% に削減されるという予想である．しかし，それが可能になるためには，Domain Knowledge の表現，Component Interfaces や Design History，Collaborative Work などのモデル化について，一層の研究が進められなければならない.

6. 環境と定量化

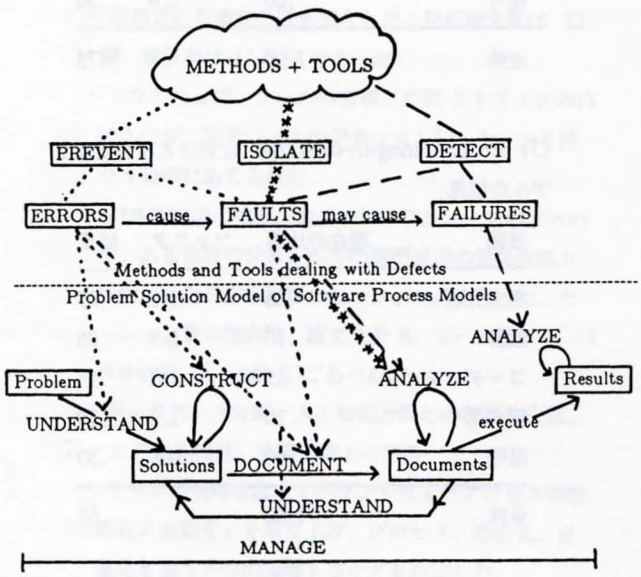
セッション 19 で発表された V.Basili の論文は，プロジェクトが採用するソフトウェア・プロセスとそれを支援する技法・ツールの効果を定量的に把握し，ソフトウェア・プロセスの評価，適切な技法・

ツールの選択を容易にしようという NASA/SEL での研究の紹介として，おもしろかった．

具体的には，技法・ツールを縦軸にとり，プロジェクトの良否を表わす尺度を横軸にとって，技法やツールがプロジェクト品質に与える効果を格子点に置く評価法を用いる．横軸の尺度としては，Defect が用いられる．Defect は，

- Error： 与えられた情報を理解する，問題解決法を考える，手段やツールの利用法を理解する等の，人間の思考過程で発生する誤り．
- Fault： Error の顕在化．
- Failure： ユーザ要求との不一致

の３種類に分解される．さらに，この３つを，下図に示す問題解決モデルと対応させて，以下のスキーマに分類する．



Errorスキーマ1：開発活動の時間的推移
Errorスキーマ2：誤解や誤用のレベル
Faultスキーマ1：Error スキーマ１と同分類
Faultスキーマ2：Fault の型
Faultスキーマ3：プロダクト視点
Failureスキーマ1：Error スキーマ１の部分
Failureスキーマ2：動作状況

上記スキーマを横軸にとり，たとえば機能テスト，構造テスト，コード読解，構文エデイタ，共通データの追跡参照などを縦軸にとって，技法やツールが Error/Fault/Failure を防止・孤立・検出しうる能力を５段階評価値で表わして，格子点に記入する．

このような評価手段を用いて，プロジェクトごとに，技法・ツールの選択，実施，事後解析と評価の実験を繰り返しつつ，データベースの内容を蓄積していく．プロダクト品質の評価を目的とする従来のプロジェクト・データベースとはちがって，プロセスの品質を評価するためのアプローチとして，非常に参考になる研究だと感じられた．

８．おわりに

とくに印象に残った点をまとめると以下の通りである．

(1) 操作型パラダイム，変換パラダイムを支援するツール群が出そろいつつある．とくに，操作型パラダイムに基づく研究はフィールドテストの段階に入りつつある．
(2) ＡＩブームの中で，知識工学をソフトウェア工学に適用する研究グループが冷静な研究上の見通しを持ちはじめた．
(3) プロセス・モデルのパラダイムを中心にして，従来の方法では攻めきれなかった「設計」支援の問題に対するアタックが始まった．
(4) ツール展示は，グラフィクエディタ，ワークステーション，マルチウィンドウのオンパレードであった．

アメリカに来る度に，いつもながら感じるのは，研究層の底の深さと広がりである．新しい考え方や概念が出現した時に，組織的・系統的に関連研究グループが結成され，アイデアやデータを出しあえる体制がある．すぐ役に立つ既成のものを見つけて手軽に導入しようという日本人の傾向とは対照的な，ともに新しいものを築きあげていこうとする研究開発環境は魅力にあふれている．

その意味で，たとえば，プロセス・モデルやプロセスプログラミングにしても，短兵急にその即効性を議論し，評価を決めつけるのは，好ましくないように思われる．

論文発表を終えて

野村　敏次
日本電子計算

１．はじめに

去る３月３０日より４月２日の４日間にわたり，第９回ＩＣＳＥがアメリカ西海岸のモントレイで開催された．幸いにして，私の論文 "Use of Software Engineering Tools in Japan"が，世界各国から寄せられた論文の中から，競争率約９倍という難関（？）を突破して採用され，会議でプレゼンテーションをする機会を得た．世界で最も権威ある，ソフトウェア工学の国際会議に，初めて出した論文が通るとは思わなかったし，まして，何百人もの外国人の専門家の前で英語でプレゼンテーションするなど，夢にも思っていなかったので，正直なところ嬉しさ半分／悩み半分であった．

２月の初め，私とちがって英語に強い何人かのＳＥＡ会員の方々に多大な御協力を頂いて，プロシーディングに収録する最終論文を仕上げ，事務局に送った後は，発表に使用するスライドを作っただけで，プレゼンテーションの練習を何もしなかったことは，我ながらＢ型に近いＡ型であるとあきれている．

会議の内容については，他の参加者の方々が詳細に報告していると思うので，私としては，発表者の心情を率直に書いてみたい．

２．発表前日

私の発表は，４日間ある会議の最終日である４月２日の，朝一番のセッションであった．まず，これが気分を重くしていた．こういうものは，早いうちにに終わらせて，後はのんびりと人の発表を聞いているのが理想的だが，世の中なかなか自分の思う通りにはいかないものである．それでも，最初の２日間は，発表のことはあまり気にしないで，人の発表を聞いていられたのだが，さすが前日になると人の発表内容を聞くというより，発表のやり方や喋り方だけに注意を注いでいたといっていい．明日は，こうして／ああしなければならないといったことを，一生懸命考えていたのである．

会議会場のモントレー・コンベンション・センタの２階には，発表者の練習用の部屋が用意されていて，発表者はそこでいつでも自由に練習ができるようになっていた．私が練習に行った発表前日の午後は，コーヒーを飲みながら発表の準備をしている人が１人いただけで，気持ちの悪いほど静かだった．部屋は２つに分かれていて，１つの部屋には，ＯＨＰが２台とスライドが１台用意されていた．隣の部屋は，いくつつかのテーブルと椅子が用意されていて，コーヒーも準備されていた．すでに部屋のなかにいた人と挨拶を交わした後，さっそくスライドにフィルムをセットして練習をしようとしたが，何しろ初めてのことで，うまくセットできない．明日の本番で失敗しないようにするためには，この機械の扱い方を覚えておかねばならないと，何回か練習を繰り返した．これでスライド操作に関する心配はなくなった．プレゼンテーションの練習を２－３回繰り返したところで，いまさら練習してみても始まらないという，怠け心がまたしても頭をもたげ，何とかなるだろうという安易な気持ちで部屋を出た．

この日は夜７時から，Monterey Bay Aquarium でレセプションが開かれた．魚の泳ぐ姿を楽しみながらののんびりとした歓談の場も，フッと心のどこかに明日の発表のことが思い出され，口や顔には出せないが気分は重く，心から楽しめなかったのは残念だった．

３．発表当日

(1) Before Presentation

いよいよ，待ちに待った（？）発表の当日である．朝７時半からセッション・チェアマンを囲んでの朝食会があった．チェアマンの鳥居先生，午後のセッションのパネラーである岸田さんと３人で朝食会の会場に向かった．会場に向かう途中の私はかなり緊張していたと思われる．鳥居先生が，自分も最初の国際会議の発表の前は，緊張していて食事もとれなかった，と話して下さった．多分，私の緊張をほぐすための優しい心遣いであったと思う．

朝食をとる会場では，各セッション単位に丸テーブルに分かれるようになっていて，既にかなりの人が集まっていた．今回の会議のジェネラル・チェアマンの

W.E.Riddle，カリフォルニア大学の A,Wassereman，イギリスは Imperial College の M.M.Lehman，コロラド大学の L.Osterweil，TRW 社の B.Boehm，Wang Institute の R.Fariley 等々，世界のソフトウェア工学の権威者が顔を揃えていた．私にとっての救いは，セッション・チェアマンが鳥居先生であるということだった．何故なら，発表はともかくとして，質問で立ち往生してしまうことを最も心配していたからである．

朝食は，簡単ながらバイキング形式で，飲み物／食べ物を取ってきて，自分のテーブルで打ち合わせをしながら食べるようになっていた．パン２切れ，ジュース，コーヒー２杯，果物を食べたら，もう何か胸が一杯で入らない（それだけ食べれたら，充分だという影の声もあったが）．

発表の順序，発表時間の確認，チェアマンが紹介する発表者の経歴の再確認等を行って朝食会は終わった．時間はすでに８時を回っていて，準備のため急いで会議場に向う．会議場の発表者の席に着くと，事務局の人が使用するのはＯＨＰかスライドかを聞きにきた．スライドを使用する旨伝え，準備をしてもらう．その間に，昨日練習したフィルムのセットを行う．セットしたスライド・フィルムを使ってピント合わせが終わると，事務局の人がやってきて，ワイアレスの操作盤の使用方法を親切に説明してくれた．そうこうするうちに，開始時間近くなっていたが，この時にはもう気分も落ち着き，会場を眺め回す余裕も出てきていた．

会議は１階の２会場（Sierra 1&2）と，２階の１会場（Steinbeck Forum：モントレイはノーベル文学章作家ジョン・スタインベックゆかりの地である）の３会場でパラレルに行われ，さらにコンベンション・センタと繋がっているダブルツリー・ホテルの１階でツール・フェアとブック・フェアが行われていた．私たちのセッションは，この中の” Sierra 2” で行われた．満席になれば７００－８００人は入る会場の２／３程度は埋まっていたように記憶している．

定刻の８時半になり，鳥居先生の流暢な英語でセッションは開始された．セッションの発表者は４人で，私はその３番目の発表であったが，私以外の発表者はすべてアメリカの大学の先生であった．因みに最初の発表は，"A Comparison of our Design Methods for Real-Time System" というテーマで Furman 大学のコンピュータ・サイエンスの先生が発表された．２番目は，"Knowledge-Based Software DesignUsing Design Schemas"というテーマで，Illinois 大学のコンピュータ・サイエンスの先生が発表された．私の後の４番目は，"A System for Parallel Proguramming"というテーマで，Ohio 州立大学のコンピュータ及び情報サイエンスの先生が発表された．これらの人達と一緒では，日本人の私の発表がヘタなのは当たり前と考えたら，一層気分が落ち着いて，じっくりと他の発表を聞くことが出来た．

(2) After Presentation

何とか私の発表も無事に終わった．そして，セッションが終了し，資料を片付けている私の周りに５－６人の人達が寄ってきた．ソフトウェアのエラーに関する調査資料はないか？　という質問をした人，名刺に"Mr. Nomura! Excellent Presentation Thank you" と書いた名刺を渡して握手を求めた人，ＯＨＰのコピーが欲しいというＨＰ社の人，貴方の発表に非常に興味を持ったので，類似の調査を論文にして投稿して欲しいと言ってきたイギリスの Butterworth Scientific 社の女性記者等々であった．自分の発表がどのようなものであったかは，本人にはわからないが，少なくとも終わった後こうして何人かの人が寄ってきて話し掛けてくれたことは，それなりに内容を理解してもらえたものと思っている．

そんな安心感があってか，午後のパネル・セッションで岸田さんが得意の（？）「企業内遊園地」の話をして，会場を笑いの渦に巻き込んでいるの聞いて，一緒に心から笑うことができた．

４．終わりに

終わってみればどうということもないが，発表が終わるまでの待つ時間は，大変嫌なものである．同時に，もっと英語の勉強をしなくてはならないと，いつも思うのであるが，喉元過ぎれば何とやらのたとえのごとく，進歩のない私である．ただ，私のようなものにもできたのだから，これからは，もっと若い方々がどしどし論文を書き，国際舞台に飛び出していって欲しいと切に願う次第である．

西海岸 CAI 事情

寺嶋 祐一
日本電気ソフトウェア

1．はじめに

おもしろいたとえ話しがある．教育をダム作りにたとえるのである．ダムを作る．初めは完全でないから，ところどころに穴がある．水も少しはもれている．アメリカの教育は，そんなことはおかまいなしに，穴だらけのダムを作り続ける．一方，日本の教育は，もちろん，穴を埋め埋め進んでいるのである．

４月初旬，カリフォルニア州パロアルトを訪問した．イースターを真近にひかえ，彩りもあざやかな大小の卵が，街のそこここにあふれていた．以下は，イースター・ラビットにエスコートされて見聞してきたことがらの，私見聞録とでもいうべき，コンピュータ教育（ＣＡＩ注**）よもやま話しである．

2．コンピュータ教育のこれまで

コンピュータ教育は，かれこれ２０年前にスタートしている．しかしかなり長い期間行なわれているが，本質的にはほとんど進展していない分野のひとつである．過去2度にわたってＣＡＩブームと呼ばれた時代があったが，いずれも実を結ぶにはいたっていない．

今は，第３次ブームの途上にあるといってよい．若干，年代上のずれはあるものの，発展過程の大筋は日本もアメリカもさほどの違いはない．最近についていうならば，アメリカでは，１９８１年頃から５年がかりで続けられてきた，学校など教育現場へのパソコン導入がそろそろ終わりにさしかかっているのに対し，日本では，ようやく昨年から学校へのパソコン導入が本格化しはじめたところである．現在は，およそ５年の遅れで推移しているといえる．

注**：日本でいうＣＡＩ（Computer Assisted Instruction）とアメリカでいうＣＡＩとは，意味合いが異なる．日本では，特に断らない限り，ＣＡＩといえば広くコンピュータを使用した教育全般を指すのが普通である．アメリカでＣＡＩといえば，この分野の初期の頃のコンピュータを使ったドリル型学習という狭義の意味にとられがちである．アメリカにおいては，日本でいうＣＡＩに相当する語として，コンピュータ教育（Computer Learnibg）という言葉があり，現在ではＣＡＩよりもこちらの方が一般的となっている．

3．アメリカの経験

ここ５年間に，アメリカの政府，学校，企業が行なってきたことを見ておくことは，今後の日本の行方を考える上で興味深いと思われる．アメリカの現場の教師や関係者の話しによれば，この５年間のうち，初めの２年間は「失敗の２年間」であり，後の３年間はその反省にたった「再出発の３年間」であったという．この間，政府はお金を，学校は人と場所を，そして企業はコンピュータをそれぞれ提供し，官民共同でコンピュータ教育の発展に貢献してきたわけである．ただ，残念なことに，これら３者をとりまとめる体制が十分ではなく，思ったほどの成果には至らなかったようだ．

「何よりも，まず，アップルのコンピュータが欲しかったんだ」とある先生はいう．この言葉が示すように，初めはコンピュータの導入が先行し，それからコンピュータを使った教育や教材を考えた．そして，このアプローチは２年で失敗した．導入の前に，教育のどの場面で，何のためにコンピュータを使うのかを明らかにしてからスタートすることを忘れていたのが，その失敗の原因である．

最初のうちこそ，自らコンピュータ教育用の教材を作成していた教師も，よい教材を作ることの難しさと，実際に教材作りに充てる時間のなさから，自分で教材を作成することは諦めざるをえなかった．この問題は，教師と教材作成専門家の分業体制の確立，ということで一応の決着がついた．しかし，優れた教材をいかに作るか，という本質的な問題は，依然として未解決のままである．

また，コンピュータを教師や生徒により身近なものにするため，さまざまなツールが工夫され，インタフェイスも改善された．今はといえば，それぞれには優れたものもではじめた．コンピュータとツール類と教材という一連の道具立てを，実際の教育の中で，どのような形で利用していくかを，各教師が模索している段階である．

4．日本の実情

一方，日本の実情を眺めると，文部省の予算政策等により，１９８６年頃から多くの学校でコンピュータの導

入が開始されている．また，文部省・通産省共管の財団法人教育開発センターが設立され，活動も活発化している．その意味で，日本における第３次ＣＡＩブームのスタートといえる．しかし，初年度の様子は，やはり「ハードウェア導入」が先行しており，かつて失敗を経験したアメリカの初年度と酷似していて，気がかりなところである．

コンピュータ教育の分野で，アメリカのパワーを見せつけられるのは，教材の豊富さとその価格の安さである．現在の日本の体制・構造から，これに比肩する教材群が生まれるかどうかは見当がつかない．しかし，それだけの量を誇るアメリカにおいても，実際に使えそうな優れた教材となるとまだまだ少ないという．インパクトが大きい質のよい教材を作ることが難しいという点に限っていえば，彼我の事情は同じである．

いわゆる「使える」教材であるためには，面白くエキサイテングな題材であることが必要であるが，それだけで十分というわけではない．人間の思考プロセスや理解の過程に刺激を与える要素が，なんらかの形で含まれていることが肝要である．そのためには，教育者のノウハウだけでなく，社会学や心理学などのいろいろな分野の人達とともに，可能性を探っていかなければならない．

５．コンピュータ教育の３つの問題

アメリカの教育界は，コンピュータ教育について３つの問題を抱えているといわれる．第１はシステムの問題，第２は教師の問題，そして第３はリーダシップの問題である．

第１は，教師が仕様に値すると考えるパワフルなシステムと教材がまだ存在しないことである．試験的に使うには興味深いものであっても，従来の教育に自然に取り込めるようになるには，コンピュータとソフトウェアの双方に，さらなる工夫と洗練が必要である．

第２は，教師の技術レベルや考え方が技術の進歩に合わせて切り替ていけないという問題である．現場の教師はいつも新しい技術から置き去りにされ，知識を更新するチャンスにすら恵まれていないのが普通である．そのために，生徒に与えるプログラムや教材の使い方がわからず，生徒に先行されてしまうことさえある．さらに深刻なのは，システムを使った際に生徒が示す反応を，教師は見ることはできるが，その活かしかたについては分からないといった問題である．これでは，コンピュータの導入は混乱を増すばかりであろう．

第３は，教育現場の閉鎖性の問題である．教育界が外界に開かれていないため，外界の技術の良し悪しを正しく評価する力がない．また，世の動向と技術革新を取り込みながら進むには，投資も必要である．これらのことについての理解とリーダシップに欠けている．というより不在なのである．問題は十分認識されているであろうが，さて実際に事を起すとなると，担い手がいないというのが実情であろう．

これら３つの問題は，やがて日本のものともなるであろう．

６．何がかわるか？

コンピュータを導入したからといって，教育の本質が変わるわけではないであろう．また，もちろんこれは大方の教師の見方でもある．しかし，モチベーションの与え方や指導法の点で変わる可能性はある．コンピュータ教育において，特に教師は一方的に何かを教えるというよりは，学習の促進者（インストラクタあるいはアドバイザ）としての色彩を強めてゆくであろう．

結果として，コンピュータ教育が受け入れられて軌道にのる可能性もあるし，逆に教育の現場へのコンピュータ導入は，結局のところ見合わせるということになるかもしれない．まだ，結論はでていない．

多くの新しい活動がそうであるように，コンピュータ教育もまた，かっての先駆者達の夢とボランタリエナジーによって，これまでの流れを刻んできた．アメリカでは，今，その夢がさめかけている．再考のときであるという．およそ５年の距離をおいて追従している日本が，アメリカと必ずしも同じ道を歩むとは限らない．が，そのようになると考える方が，そのようにはならないと考えるより自然であろう．夢が消えないうちに，そのときまでに，何ができるであろうか．

ハサミやノリや定規やコンパスを使って楽しい模型工作をするように，優れたシステムやソフトウェアを使って新しい教育の可能性を探りたい．コンピュータを特別扱いすることなく，文化や芸術を語るがごとくソフトウェアを語れる，そんな世代を我々の手で育てていきたいものである．

企業内 CAI の実例

大木 幹雄
日本電子計算

1．はじめに

CAI 分科会（SIGCAI）の第 3 回の月例会を，以下のように行なった．

開催日時：7 月 21 日（火）18:30-21:00
開催場所：日本 DEC 教育センター
（池袋サンシャイン 60）
スピーカ：日本 DEC 教育本部教育部
小川，後藤，小澤各氏
テーマ：日本 DEC における企業内 CAI
出席者：12 名

この月例会の内容を，わたしなりにまとめて報告をしたいと思う．

2．内容

CAI を企業内教育に利用しておられることで有名な日本 DEC をお訪ねして，デモンストレーションを含めて，日本 DEC で行なっている企業内 CAI（Computer Assisted Instruction）の実例を説明していただいた．

(1) 概要

当日デモンストレーション対象のシステムおよびソフトウェアの概要説明をまずしていただいた．

IVIS（Interactive Video Information System），DECtalk，VMSを初めて学ぶ人のためのCAIシステム，および MTS（Management Training System）についての概要説明が，SIGCAI の世話人の 1 人である小川さんからあった．

(a) IVIS

このシステムは，対話を行なうさいに，双方向の会話を可能にしたビデオ教育用システムであり，ハードウェアは，Profesiona l350 と呼ばれる本体と，タッチ・センサー付きのカラーTV，ビデオ・ディスクプレイヤー，キーボード，FD，その他周辺機器から構成されている．

IVIS の特徴は，"The Producer" と呼ばれる教育ソフトウェアの開発支援ツール（実際には特別のプログラム言語）が用意されていて，動的なコースウェア・フローの制御や画面制御，問題や成績表作成などが，容易に行えることである．画面制御は，通常のプログラム言語のレベルと同程度に，細かく行なえる．残念ながら，この特別なプログラム言語の詳細については，知ることができなかった．「容易に」というのが，本当にどの程度かが重要になってくると思う．ともかく，利用者というのは面倒なことがあると，それだけでそのシステムを受け入れないということを，筆者自身もいやというほど経験している．

現在，日本 DEC では，IVIS それ自体の販売活動はしておらず，もっぱら社内用として用いている．そのひとつの理由として，ビデオ・ディスクのプレスだけでも 80 万円程度はするので，1 教材当り 50 セット以上利用する企業でないとペイしない，という事情があるようだ．しかし，DEC 社内の要員教育には使われている．それは，全世界に散らばっている要員を，1 箇所に集めて教育する際の交通費，宿泊代等を考えれば安い，ということである．1 教材当りの 50 台という数値は，その経験を基にしたものであることから，かなり信頼性のある数字になっている．

(b) DECtalk

英文テキストを入力として，音声合成をするハードウェア／ソフトウェア一体のシステムである．日本語の発音は英語なまりになるが，英語の発音はほぼネイティブなものと区別がつけにくい．男女，子どもと複数の声質の選択ができるようになっている．販売価格は約 100 万円で，主に輸出関連企業に販売している．

この種の類似品として，Macの世界では，Macintalk，Smoothtalker がある．

(c) VMS 入門 CAI

日本 DEC で販売しているハードウェアのマイクロ VAX からVAX8800までの基本 OS である VMS の体得を目指したもので，本物の VMS を実際に用いながら，VMS や EVE（エディタ）の学習を行う．この方法は，基本的には，BASIC を学習するのに，BASIC 用 CAI ソフトの裏で，実際に BASIC を動かし，シュミレートする考え方と同じである．

このツールは，前述した "The Producer" を用いて，米国 DEC で開発されたものであるが，モジュール分割が非常に巧みに行なわれていることから，日本語化に対する変更が簡単に行なえるようになっている．

(d) MTS

管理者教育用のインテリジェント CAI である．実際のデモで使用して，初めて膨大なものであることが実感できるシステムであった．管理者の精神コンサルティングも行なってくれるが，日本とアメリカの文化の違いなどはどうなるのだろうか？

(2) デモンストレーション

(a) IVIS，DECtalk のデモ

後藤氏により IVIS，DECtalk のデモが行なわれた．

当日参加した参加者により意見が異なろうが，正直にいって，単に英文のテキスト（発声に関する制御文字など一切含まない）を読んで，DECtalk の音声合成の自然さには驚嘆させられた．ただし，日本語の発音は，かなり苦労してテキストを作っても，バタ臭い発音になる．これは仕方がないとは思うが，日本で本格的な商売をしようと思ったら，英語程度の自然さがなくては無理ではないだろうか．

(b) VMS 入門 CAI，MTS のデモ

後藤氏，小澤さんによる VMS 入門 CAI と MTS のデモを，2 グループに分かれて見学をした．

VMS の CAI は，実際に CAI システムの指示に従って，VMS のコマンドを入力すると，それが VMS に渡され，その結果が CAI の画面に戻されてくる．画面表示は，キャラクタ・ディスプレイを用いているため，多少ぎこちないが，本番として走っている VMS を CAI として用い，結果を CAI ソフトに返すという仕掛には，VMS 側でもそれなりの工夫を必要としたのではなかろうか．

マネジメント訓練システムである MTS は，機能が膨大過ぎるため，一部の機能であるマネージャの精神分析に関するデモを，実際に試してみた．かなりハイレベルの英語が問いとして表示されるため，わたし程度の英語力では，その意味を理解するだけでも困難である．そこで，適当に応えたところ，コンピュータから出された最終診断は，「あなたは働きすぎだ．休養を取った方がよい」と出力された．「フーム，このシステムは実に正しい」，と 1 人で納得をしてしまった．

しかし，やはり英語での入出力は，わたしを含めて，平均的な日本人の英語力では，かなりむずかしい，というのが実感であった．

(3) まとめの質疑応答

最後に，デモの結果を踏まえて，CAI をビジネスとして考えた場合，つまり CAI は儲るか，等の質疑応答を行った．

日本 DEC の場合，教育本部は独立採算であるが，社内教育も社内売りで売上を経常しているため，収支はとれているとのことであった．顧客教育が見込めるからいえることかもしれない．

CAI はもうかるか？，という質問に関しては，ソフトウェア・ハウスとメーカーの立場の違いを感じた．

3．筆者雑感

小川さん，後藤氏，小澤さんの計 3 名で，夜遅くまでマシンのセット，デモンストレーション等も含め，実に丁重に対応して頂きました．まずは，紙面を借りて，小川さんをはじめとする日本 DEC の方々に感謝致します．

日本 DEC の教育本部は，スタッフが 8 名で，使用しているハードウェアが VAX11/785，マイクロ VAX，VAX8800 というぜいたくさです．こうしたことからも，会社全体での教育に関する意気込みが感じられた．これからの企業は，中堅社員の再教育を初めとして，企業内教育に力を注いでおかないと，日進月歩の技術の変化に取り残されてしまうのではなかろうか．そうした意味でも，企業内教育に CAI を取り込んで行くというのは，重要なアプローチではないかと思う．

サンシャイン36階の夜景を見ながらの月例会は，日頃，地面を這って仕事している者にとって，久しぶりに潤いをもたらしてくれた．

企業内 CAI を如何に行うかを実際に見学することは，100 冊の本を読むより有益です．SIGCAI メンバーのみならず，SEA 会員各位の一層の参加を期待します．お近くに参加を希望される方がおられたら，是非ご参加ください．

また，SIGCAI で見学したい企業，あるいはデモしてよい方をご存じでしたら，世話人までご一報下さい．実現のために最大限の努力をしたいと思います．

名古屋支部紹介

岩田　康
ソフトウェア・リサーチ・アソシエイツ

１．はじめに（気が付いてみたら世話人だった）

私がＳＥＡに入会したのは今年の２月頃ですが，それから３ヶ月もたたないうちに，ＳＥＡ本部から「名古屋支部発足準備ミーティング」と題した案内が電子メイルで送られてきました．その文中に，設立準備ボランティアとして申し出ている人の先頭に，私の名前があったので，あまり深くも考えずにとりあえずキック・オフだけでもしようか，ということで会員に呼びかけたところ，予想以上に名古屋支部をつくろうという有志が集まり，引くに引けなくなってしまいました．

というわけで，今のところ主に，月例会の司会と各種案内状の作成，送付，および会計といった，緻密で責任感の強いＢ型人間にピッタリの役割を仰せつかっています．こうした事務的雑務は面倒なものですが，これはやってみれば結構楽しいものです．なによりも仕事だけでは知り合えなかった多くの人達と，さまざまな情報交換と人間交流ができるからです．

２．今までの活動

発足ミーティングを６月２９日に開いてから，今までに，月例会を２回とセミナーを１回行っています．

■　月例会

月例会は，原則として毎月末の金曜日夜６：３０から８：３０まで，世話人の会社の会議室を無料で借りて行っています．出席者は毎回大体１０人前後で，特にテーマを決めず，技術的話題の他に，支部運営等についてサロン風にワイワイやっています．その後は有志で，といってもほとんど全員で，２次会へなだれ込むパターンが定着しています．

参加費は，会員５００円，非会員１０００円で，泡のたつ飲物付きです．来るものは拒まずの精神でやっていますので，会員のかたは周りの方を誘って，自由に参加してください．

■　セミナー

名古屋近辺のＳＥＡ会員を増やして，支部活動を活発にするために，月１回のペースでセミナーを開催しようと思っています．まず第１回目は，８月２９日にＳＥＡ本部と大阪支部から講演者の応援を得て，このところ活動が停滞ぎみのｊｕｓ東海と共催で，電子出版等をテーマに行ないましたが，準備不足の第１回目としては，３０数人が集まり，収支共にまあまあだったと思います．

３．今後の活動

■　月例会

しばらくは今までどおり，ワイガヤサロンでいこうと思っています．つまり，技術交流より人間交流を主体にやっていこうというものです（横浜支部が月例会後の飲み食い会で評判ということですから）．

■　セミナー

セミナーの参加費は貴重な支部運営資金となるので，今後も月一回ぐらいのペースで開催したいと思っています．第二回目は，一回目の電子出版にからめ，出版会社の人に講師をお願いして『外国向けマニュアル作りの１０年間を振り返って』と題して，電子出版の実践編として１０月３日に行いました．

■　分科会

前出のアンケートで，分科会の世話人をやってもよいという人が，４名程名乗りを挙げてくれたので，これからも月例会やセミナーの席で希望者を募り，徐々に発足，活動していきたいと思います．興味のある方は世話人までご連絡ください．

４．名称，ロゴ，ｅｔｃ

今のところ，とりあえずＳＥＡ名古屋としてありますが，ｊｕｓ東海と合わせてＳＥＡ東海にした方がいいとか，ＹＤＯＣみたいなカッコいいのにしようとか，会員からいろいろ勝手な意見がでています．ロゴ・マークは，いいアイデアがあるのですが，ｐｉｃを使って書くとハマってしまいそうなデザインなので，どうしようか思案中です．

５．世話人

当面以下のものがつとめます．次回のセミナーのテーマ／開催日時とか，分科会のこととか，その他諸々支部の運営に関してご遠慮なくお問い合わせ下さい．

岩田　康（ＳＲＡ名古屋営業所　tel：052-04-5411），鈴木　智（日本電子計算　tel：052-01-5100），西村　享（名古屋工業大学　電気情報科　tel：052-32-1111）

ＳＥＡ会員状況（昭和 62 年 9 月 24 日現在）

正会員	1155名（8月25日から86名増）
賛助会員	31社（8月25日から3社増）

＜正会員の勤務地および居住地域分布＞

	勤務地域	居住地域
北海道＝	9	9
山形　＝	1	1
宮城　＝	3	2
岩手　＝	6	6
福島　＝	1	1
秋田　＝	1	1
新潟　＝	6	6
栃木　＝	6	5
群馬　＝	1	1
茨城　＝	7	9
埼玉　＝	16	72
千葉　＝	15	73
東京　＝	667	415
神奈川＝	85	212
長野　＝	31	31
富山　＝	3	3
石川　＝	2	2
静岡　＝	15	13
岐阜　＝	2	6
愛知　＝	35	29
和歌山＝	2	1
三重　＝	1	2
滋賀　＝	2	3
京都　＝	14	19
大阪　＝	129	102
奈良　＝	3	7
兵庫　＝	28	44
愛媛　＝	2	2
徳島　＝	3	3
岡山　＝	1	1
広島　＝	4	4
福岡　＝	12	12
熊本　＝	12	12
大分　＝	4	4
佐賀　＝	1	1
宮崎　＝	1	1
鹿児島＝	3	3
沖縄　＝	1	1

＜男女分布＞

男　＝	1092
女　＝	63

＜年齢分布＞

20以下　＝	0
20_24　＝	54
25_29　＝	272
30_34　＝	315
35_39　＝	282
40_44　＝	135
45_49　＝	53
50_54　＝	20
55_59　＝	10
60以上　＝	7

＜血液型分布＞

A型　＝	466
O型　＝	322
B型　＝	251
AB型＝	117

＜賛助会員会社名＞

ＩＮ情報センター，旭化成工業，ＳＢＣソフトウェア，エムテイシー，サン・ビルト印刷，情報システムサーブス，ジェーエムエーシステムズ，セントラル・コンピュータ・サービス，ソフトウェア・リサーチ・アソシエイツ，ニッポンダイナミックシステムズ，マイクロキャビン，ＰＦＵ，リクルート，リコーシステム開発，近畿日本ツーリスト，構造計画研究所，神戸コンピューターサービス，経調，辻システム計画事務所，東海クリエイト，東電ソフトウェア，日進ソフトウェア，日本システム，日本システムサイエンス，日本能率コンサルタント，日立製作所，富士ゼロックス情報システム，富士通，富士通ビジネスシステム

（アイウエオ順）

論 文 募 集

ソフトウェア・シンポジウム'88

1988 年 6 月 8-9 日(予定) ◇ 農林年金会館(東京・虎ノ門)

共 催:(社)情報サービス産業協会(JISA)
ソフトウェア技術者協会(SEA)

はやいもので、ソフトウェア・シンポジウムも第8回目を迎えます。 産業界の最前線で、日頃忙しくソフトウェアの開発・保守・運用・管理に従事しているエンジニアたちが、それぞれの仕事のなかから得た経験や知識、アイデア、あるいは意見を交換しあうための貴重な場として、多くの方々の賛同を頂き、内容もますます充実してまいりました。

ここ1-2年、ネットワークの整備やワークステーションの普及、あるいは各種先進的技法/ツールの実用化など、ソフトウェア業界をとりまく状況は急速に変わりつつあります。今回のシンポジウムでは、こうした新しい社会的・技術的環境への対応を発表や討論の主題として、従来にもまして魅力的なプログラムを組みたいと考えております。

実践的技術の経験、先進的技術の試行結果、基礎研究の成果等に関して、ヴァライティに富む多数の論文がよせられることを期待いたします。

募集論文テーマ

特に限定はしていませんが、たとえば、次のような話題がセッションーテーマの候補として考えられます:

- ソフトウェア開発におけるAI関連技術の応用
- 実際のプロジェクト支援環境における問題点
- 新しい開発/保守方法論の実践とその評価
- ユニークなアプリケーション開発の事例
- 技術者教育のケース・スタディ
- ドキュメンテーションの機械化/合理化
- ネットワークのセキュリティ/維持/管理
- プロジェクトにおける技術移転
- LANを用いた分散型環境の構築・利用
- 先進的ソフトウェア・ツールの試作・実験
- ソフトウェア・プロダクトの流通と権利保護
- 定量的マネジメント・各種モデルの適用と評価

等々。

応募要領

応募論文の概要を4千字~6千字程度にまとめたもの(ワープロまたはコンピュータ出力でA4版4~5ページ)のコピー10部を、

1987 年 11 月 30 日までに

2人のプログラム委員長(林 香または藤野 晃延)のいずれか宛にお送りください。

プログラム委員会で応募論文の審査を行い、結果を1988年1月末までに、応募者全員にお知らせします。

審査を通過した方々には、発表用の最終論文(予稿集に載せるためのカメラ・レディ原稿)を1988年3月末までに仕上げていただくことになります。

その他、パネル討論のテーマや招待講演、チュートリアルなどに関して、何かアイデアがありましたら、実行委員長までご連絡ください。

シンポジウム・スタッフ

実行委員長
高田 佳彦
日本電子計算
科学技術事業部
〒103 中央区日本橋兜町 6-7
Tel. 03-668-6171

プログラム委員長
林 香
ソフトウェア・リサーチ・アソシエイツ
環境開発本部
〒102 千代田区平河町 1-1-1
Tel. 03-234-2611

藤野 晃延
富士ゼロックス情報システム
技術部
〒160 新宿区西新宿 3-16-6 西新宿水野ビル
Tel. 03-378-8010

ソフトウェア技術者協会
〒102 東京都千代田区隼町2-12 藤和半蔵門コープビル505
TEL.03-234-9455　FAX.03-234-9454