SEAMAIL

Monthly Newsletter from
Software Engineers Association

Volume 1, Number 7 | July 1986

目　次

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．１，　Ｎｏ．７　　昭和６１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）
〒１６６　東京都杉並区高円寺南１－５－４　高円寺サンハイツ４０４
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

前号は，いろいろな事情で，５－６月合併号ということになってしまいました．多くの会員の方から，心配や激励のお電話をいただき，ありがとうございました．なんとかがんばって，毎月１回の発行を続けていきたいと思います．

誌上フォーラム

今回の誌上フォーラムは，「ソフトウェア産業論」をテーマに，４人の方に原稿をお願いしました．

ソフトウェア・ビジネスといっても，きわめて幅が広く，その現状や将来についての意見は，ＳＥＡ会員のあいだでも，それぞれのおかれた環境によって，さまざまに異なるでしょう．単眼的な視野では，これからのきびしい状況に，とうてい対処できなくなるものと予想されます．

４人の会員の方々の意見を，みなさんは，どう読まれたでしょうか．反論や意見をどしどし編集部（ＳＥＡ東高円寺事務所）あてにおよせください．

ＳＥＡ設立記念Ｆｏｒｕｍ

ＳＥＡ設立記念フォーラムは，５月１９日（月）に，東京・農林年金会館パストラルで，「ソフトウェア技術者に期待する」という基調テーマで，基調講演，パネル討論，情報交換パーティーという３部構成で行われました．

ＳＥＡの正式設立は，昨年１２月に行われていますが，今回イベントは，その活動を広く一般にお披露目しようということで，企画されたものです．

午後の部の基調講演およびパネル討論には，延べ１４０名の参加者を得ました．ＳＥＡならではの豪華メンバーで行われたその講演と討論の記録を編集部でまとめました．これからも，こうしたイベントに参加できなかった会員へのサービスの意味で，編集部の力がおよぶかぎり，記録をのせるようにします．

なお，今後企画されているイベントの一覧は，巻末３６ページに掲載してあります．

会員からの投稿

今号は，一般からの投稿がなく，編集スッタフの中の渡辺さんに「ＣＯＢＯＬ８５の使い心地」と，藤野さんにＡＩ／ＫＥに関する文献紹介を書いていただきました．ＳＥＡＭＡＩＬは，みなさんの雑誌です．エッセイ，書評，情報紹介，技術メモなど，なんでもけっこうです．どんどん原稿をおよせください．

原稿の在庫はほとんどありません．そのため，いつも自転車操業の綱渡りです．みなさんの積極的なご協力をお願いします．

分科会からの成果報告

ＳＩＧＡＩおよびＳＩＧＳＬＰが正式発足するなど，分科会の活動がだんだん活発になってきています．今回は，教育分科会から河村さんに，管理分科会から芝原さんに，それぞれ研究会での発表内容をまとめていただきました．地方会員，また分科会に参加していない会員への情報提供を目的として，今後もこうした小論文をできるだけ掲載していくつもりです．

代表幹事からの御挨拶

先日の総会で代表幹事に選出された鈴木弘さんから，就任の御挨拶を今月号にいただく予定でしたが，残念ながら間に合いませんでした．編集部としては，設立記念フォーラムでの開会のことばにちょっと手を加えた程度でよいと考えていたのですが，鈴木さんのほうは，新たに書きおろしたいという御意向で，締切の関係から次号にゆずることにしました．

８月号の発行は，いまのところ，８月第１週を目標にしており，すでに一部編集作業にはいっています．

内容は，９月第１週に予定されている，「秋のセミナー・ウィーク」の講師２１人の方の「ポジション・ステートメント」を特集します．

また，秋のセミナー・ウィークの各セッションを聞いて，その討論をまとめる役割の臨時編集ボランティアを募集しています．もちろん，セミナーには無料で参加できます．希望者は，ＳＥＡ事務局までご連絡ください．セミナーの御案内パンフレットは，７月末か８月初めに別途お手元におとどけしますので，よろしく．

SEA 設立記念フォーラム

基調講演：知のマイスターの時代に向って

大野　豊
京都大学
日本ソフトウェア科学会理事長

この原稿は，去る５月１９日（月）に東京・農林年金会館パストラルで行われた設立記念フォーラムの基調講演の内容を編集部でまとめたものです．

１．はじめに

マイスターとは，中世から近代へ激動していた，１４～１６世紀のヨーロッパにあった社会的な制度であり，家内工業で物を作っていた職人の親方制度における，親方（師匠）をさしている．印刷術などが発明されていたが，近代工業化される前の時代であって，商人が権力を握りはじめており，親方も，社会的にはかなり高い地位と財力をもっていた．

２．コンピュータ初期のプログラミング

コンピュータが出現した初期のころ，つまり，第二次世界大戦が終るころは，プログラミングといえば，かなり高度な研究開発活動とみなされていた時代であった．

そのころは，プログラミングに書き上げるということは，与えられた問題を正しく分析し，さらにその内容をコンピュータの言語で表現するという，２つの能力があって，はじめて可能であると考えられていた．つまり，問題分野とコンピュータの両方に関する深い知識と，分析力が必要とされる知的活動であり，当時は，一流の研究者がこぞって，みずから，問題のプログラミングを行なっていた．

３．大規模工業化時代とプログラマ

コンピュータが発展・普及するにつれて，事務処理などの応用分野への利用が多くなり，研究から日常業務へ密着した方面へ広がっていった．範囲が広がり，研究的でなくなると，専門的にプログラムを作る仕事をする，プログラマという職業が登場した．

解決する問題も新しいものでなく，問題の解き方も日常業務化されて，プログラミングから研究的色彩がうすれてきた．コンピュータの発展にともなう社会の工業化の原動力はコンピュータであり，大規模工業化により，大量生産が行なわれて来た．そこでは，多くのものを効率よく生産する手段としてコンピュータが使われた．それによって大規模プラントも実現し，さらにコンピュータも大きなものになった．当然，プログラマの数も増加し，また，プログラマに対する需要もふえてきた．

しかし残念ながら，日本の社会だけの傾向ではないが，筋力作業，事務作業にみられるように，人手によって，きめのこまかい作業をすることは，あまり高級な仕事とはみなされない，という社会の慣習的な通念があるようである．

この時代になって，プログラミングというような，きめこまかい作業は，専門的ではあるが，労務的になってきており，プログラミングは，高度な仕事ではないと思われるようになってきた．大勢の人が，日常業務的な作業をくりかえしてやっていくと，そう思われるのはある程度，やむえないのではないかというのは，ソフトウェアに関係する者にとっては，とても残念なことである．

４．脱工業化，高度情報化の時代

ところが，１９７０年代に，社会の推移が一転して，脱工業化とか高度情報化といわれる時代になると，大量生産による効率化もさることながら，大量消費から多品質少量生産へという産業構造，つまり，量から質への生活構造の変化がおこり，ユーザ・ニーズの多様化，個別化が重視されるようになってきた．

物を作るときの設計は，ユーザ側に立った設計者が，要求を引き出すようにと変化してきた．この要求を分析し，設計を行う仕事は，コンピュータによる全面的自動化がむずかしい，きわめて人間的かつ知的な活動である．

こうした多様化，個別化は，当然，ソフトウェアにも必要となった（ある時期から，プログラムの概念を拡張して，ソフトウェアという用語が使われだした）．依然として，大量のソフトウェアの需要が続いている一方で，多様化，個別化の要求から，特色ある新しいソフトウェアが求められるようになった．したがって，大きなシステムを作るというだけでなく，個々のユーザのニーズに

応じた，きめこまかなシステムに対応するソフトウェアを作っていく必要がある．

５．高度なソフトウェアの要請

通産省の「情報処理のこれから進むべきソフトウェアのあり方」という調査の中で，マシン・インタフェイスの高度化，相互運用性の確保，分散処理システム，知識情報処理システム，画像処理システム，分散化・パーソナル化データベース，高度セキュリティと信頼性等々，高度なソフトウェアに対して，いろいろな要求が存在することが報告されている．これらは技術的にもむずかしく高度であるばかりでなく，ユーザのニーズをきめ細かくみていかないと，できないようなソフトウェアである．

このように，情報ネットワーク，分散化が情報システムとそのソフトウェアを集積化し，高度化する時代になって，ソフトウェア技術者は，こうした高度化されたソフトウェアの要求に対して，ルーチン処理的なプログラムから，ユーザの多様化，個別化した要求に答える知的活動へと駆り立てられる時代になってくる．したがって，質のよい人材がおおいに要求される．

６．知の職匠の道具と仕事場は

このような知的活動は，ソフトウェア，ハードウェアにかかわらず，分析，計画，設計において主要なものであり，人間の専門的知識と能力を必要とする高度に知的な仕事であって，単純なプログラム・コーディングとは質的な違いがある．それは，知の職匠，あるいはマイスターというべきひとたちの仕事である．

むかしの職人は，いい物を作るには，いい道具を作った．共通な道具もあるが，それぞれの親方が工夫していい道具を作った．やはり，職匠は自らの腕，頭の他に，すぐれた道具と仕事場が必要である．

知の職匠にふさわしい，知的働きを助ける道具と仕事場は何か．すぐれたワークステーションと情報ネットワークがそれである．物を設計する職匠にはエンジニアリング・ワークステーション，ソフトウェアを設計する職匠にはソフトウェア・ワークステーション，医の職匠すなわち医者にはメディカル・ワークステーションといったものが，これから知の職匠の働く場として必要であり，それらは，またお互いに接続されて，ひとつの情報ネットワークを形成している．

これは，もともとシグマが意図していたようなものであり，仕事場としてのワークステーションには，社会の共通な知識，共通なレベルとして，共通ツールが中に入っていると，そういうものを使って，それぞれの親方が，自分自身の工夫したツールをのせて，物を作るということになっていく．

共用のツールは，社会の基盤的水準をあらわすもので，だれでもが使える．しかし，より高度に使用するためには，各自，自己の知識と能力に応じて，独自のツールを用意しなければならない．職匠は，自らの腕前とすぐれた道具と仕事場によって社会的地位を確立する．人によっては，いい道具，仕事場を求め，一つの組織に縛られることなく，独立した立場をとることにもなるであろう．現在もその傾向が見られるように，社会的流動性をもつようになるであろう．

７．知のマイスターの時代に向けて

自分の経験をつんで，腕を磨くことも必要だが，また，もっと大切なことは，新しい研究開発の動向を，常時フォローしていくことである．

ソフトウェア工学が生まれて１５年たったが，その成果の現場への技術移転がなく，現場で成果が使われないとよくいわれる．そういうことがないよう，自らが研究開発をしていなくても，新しい研究開発を，知識としてツールとして，自らのワークステーションに取り込む必要がある．また，情報ネットワークを通して，これらの情報のサービスを受けることもできよう．

ここに，ソフトウェアの基礎研究を行なう研究者グループと産業界との，密接な連携の必要性がある．また，研究者にとっても実務面の職匠たちの経験と成果の情報は，次の研究への大きなきっかけとなるはずである．

このような，職匠あるいはマイスターの社会がくるかは，これからの問題であるが，これまでにみてきたように，すでにそのあらわれは，あちこちにみられる．

知のマイスターだけで，世の中が成り立つわけではないが，そういう人たちが，ひじょうに重要な役割をする世の中になるのではないかと思う．しかしながら，ソフトウェア技術者が，自らを高める努力をおこなわないかぎり，マイスターとなる資格はないし，社会的地位はあがっていかない．仕事場やＳＥＡのような場を通じて，腕と道具を使っていい仕事をすることによって，プログラマの社会的地位はあがっていく．

これからも，ＳＥＡと日本ソフトウェア科学会とが協調・協力をして，ソフトウェアの諸問題を解決していきたいと考えています．

（文責・編集部）

SEA設立記念フォーラム

パネル討論

「情報社会の立役者たるべきソフトウェア技術者の役割」

コーディネータ：辻岡　健（情報処理振興事業協会）
パネリスト：戸田　保一（野村コンピュータ・サービス）
鳥居　宏次（大阪大学）
淵　一博（新世代コンピュータ技術開発機構研究所）
松崎　稔（日経コンピュータ）

１．はじめに

辻岡：　今日は，ＳＥＡが昨年の１２月２０日に，正式に設立されてから，ちょうど６ヶ月になります．さきほど，大野先生がおしゃったように，新しいタイプの専門家集団として，これから発展していっていただきたいと思います．今日はその設立を記念して，この世界の第１線で活躍していらしゃる先生がたに，お集りいただきました．

野村コンピュータ・サービスの戸田専務には業界の代表として，大阪大学の鳥居先生には学界代表として，ＩＣＯＴの淵先生には研究者代表として，また，日経コンピュータ編集長の松崎さんには編集者というよりはコンピュータを愛するひとりとして，それぞれお話をしていただければとおもいます．

さて，今日のテーマは非常にむずかしい話題ですが，私なりに解釈させていただいて，まず，パネリストのみなさんに，社会の情報化はどうなるか，その中でソフトウェア技術者は何を期待されているか，ということを話していただき，そのあとで，その期待に答えるためにこれから何をなすべきかを，時間の許すかぎり，みなさんで討論したいと思います．

２．情報産業化のうねり

戸田：　私どもの会社は，過去２０年間，主として野村証券の仕事をしてきています．そういった背景でお話をさせていただきます．

私どもの社員は約１０００人（男子７００人）で，ほとんどがソフトウェア技術者です．その他に５００－６００人の協力会社のエンジニアのかたにもお手伝いしていただいていますが，ひとことでいって，ソフトウェア技術者に対するニーズは，まさに爆発的だというのが，実感です．

よくいわれていることですが，ハードの費用は，相対的に低くなっています．実例をいいますと，野村証券では，いま第３次オンラインの開発を進めています．４１－４５年の第１次では，概算の総費用は６０億（ソフトは１５％）ぐらいでした．それが，５１－５５年の第２次では２００億（ソフトは２５％）にふくらみ，現在の第３次では，約７００億以上（ソフトは３５％以上）になるだろうと予想されています．インフレを考えると，伸びはそうでもないのですが，ともかくソフトの占める割合が増えています．

さらに，現在約１２０万ステップのソースコードがありますが，これを２－３次にむけて，プログラム開発規模以上に，高度化された内容にしていかなくてはならない．こうした現状を考えると，ソフトウェアが，企業戦略を左右しかねないのではないか，という点では，危具の念すら感じています．

さらに私どもでは，ＰＯＳのサービスもしていますが，先日，ある顧客の全国の端末からのデータ件数が，１，１００万件をこえまして，産業の情報化がここまできたかという，感慨をもちました．そして，情報の産業化のこのうねりは，もうとまらないとも感じています．これを支えると期待されるのが，ソフトウェア技術者であると思います．

３．教育と資質

鳥居：　大学に移って２年たちましたが，町工場の親分みたいだと感じていまして，というのは，ほっておくと学生は何にもしませんでして，それが一緒にうごきますと，実に際限なくやってくれます．

最近，ベンチャー・ビジネスに入りたいという学生が

でてきまして，話をきいてみたんですが，１０年間つぶれなければ，おもしろい会社にいきたいということでした．では，その１０年間とはなんなのかと考えました．

情報化社会とカッコイイことをいいながら，プログラムの量産をして，労働集約型になってしまうのではないか．一方で知識集約とかいっているが，残業でおいまくられているともききます．このままの１０年ではと，危具の念を持ったわけです．

そこで，学生をどう教育したらいいかを考え，いろいろやってみたんですが，正直いって，教育できないのではないかとも考えています．といいますのは，学校の講義，演習のなかで実験をしまして，これが大学のいいところなんですが，情報処理にでていました，「酒屋の問題」を，２年生の後半に，他人にプログラムを組ませるための設計をやらせました．この時期には，すこし大きい問題ではあるのですが，メチャメチャに書くのもいれば，データ構造まで書く学生もいるという結果がでました．これは，学生の資質の問題ではないかと思えます．

このことは，学生の中でも，労働集約になっているのではなかろうか．そこで，もうすこし実験をしまして，時間をかければ，虫のいないプログラムができるのか，ということを，３年生にコンパイラの作成のさいにやってみました．ようするに，データ構造を決定したり，アルゴリズムをつくるという，プログラム作成能力を，バグの数でみてみようと思ったわけです．虫が取れたとき，その虫がどの時点で入ったのか，虫がどれだけの期間あったのかをみて，虫を作りこまない，虫をはやくとりのぞく能力をみてみようとしたわけです．

もちろん十分なデータ量ではないと思いますが，問題から概念を取り出すのは，たんにプログラムをつくるということとは関係ないのではないか．ソフトウェアに関するＩＱ，つまりＳＩＱがあるというのは，概念形成能力ではなかろうかと考えています．このへんに資質があるのではなかろうか，ということです．

では，このへんの能力をどうあげるのか，短時間で要求をとらえる能力を，どう訓練するかということを考えれば，プログラミングをすることは，そう必要ないのでは，と考えています．

４．応用の時代に向かって

松崎：　日経は，広く話題を扱っていて，編集方針がないんではないかといわれていますが，今日はそれを離れ，私自身がどう考えているかを話たいと思います．

「社会の情報化はどう向かうか」ということですが，結論をたんてきにいうと，８０年代後半は，応用の時代である．東大の国井先生との対談でも話たのですが，エンド・ユーザが力をもっているエンド・ユーザの時代より，システムを応用していく時代ではないかと思います．つまり，８０年代後半から９０年近辺では，システムというのは，サービスそのものではないかと思います．

私は，１０数年間コンピュータの世界に携わってきているのですが，情報化の流れを，４つのレベルでみています．最初のレベルが，６０年までのもので，ハードウェア・プロダクト，つまりエレクトロニクス製品の時代であった．この時代では，そこでの商品価値は，その製品をどう付加価値があるようにつくるか，ということでして，学問に裏打ちされた技術が，製品をきめるポイントだった．

そのあと，７０年の終わりまでが，システム・プロダクト，つまり，ソフトの時代といえます．ここでは，それまでのエレクトニクス製品とは違って，ハードとソフトをあわせて製品として動くわけでして，とくに７０年以降，急速に進展してきている．ここでの大きな特徴は，ハードウェアだけでなく，ソフトウェアが必要になっているという点です．

システムの第３のレベルとして考えているのは，システムそのもの，製品としてのコンピュータ・システムを，ユーザのシステムとしてつくりあげる，ということです．ここでなされることは，コンピュータにでできることはコンピュータにさせる，という発想になっていることです．つまり，サービスそのものではく，サービスをおこなうことを支援するようにしようという，合理化の考えかたが特徴である．

そして，第４のレベルが，さきほどいいました，８０年代後半は，応用の時代である．

こうした時代のながれをみてきますと，ソフトウェア技術者にどのような課題を提起するか，ということですが，ＳＥＡが昨年発足していますが，そのなかでの目標のひとつに，学問と実践技術のギャップを埋めること，というのがあったと思いますが，いままさにそのことが必要とされています．そして，ソフトウェアでもうひとつ問題になっていることは，過去の遺産をどうするか，ということです．

ＳＥＡが，このギャップを埋めて，かつ，過去の遺産の問題を解決してくれたらと思います．

５．ハードからソフト，そしてＡＩ

渕： 松崎さんが，情報技術の流れというか，情報化社会のながれを歴史的に展望されたのですが，そこで感じたのですが，どうも私は，本当の技術ができる前に失業してきたということです．

私がコンピュータと付き合いはじめたのは，電総研で，コンピュータの設計をしたときです．そのとき，数千ゲートのものを設計して，自分では複雑なものを作ったと思っていたのですが，数年後には，何万ゲートのコンピュータができ，さらには数年後のは，数十万ゲートものができている．私なんかは，数十万ゲートのものはどう設計していいのか判らん，ということです．

１９６０年代の後半は，ＯＳが大切だろうということで，チームを作って，複雑なものを作ったと思ったのですが，それも数年すると，はるかに規模の大きなものがでてきている．

そこで，その先はなにかということで，５－６年前から，情報処理技術は，知識情報処理という特性づけの方向ですすむのではないか，そういう方向ですすまなくてはならないのではないか，といっていた．その頃は，知識情報処理とか人工知能（ＡＩ）というのは，学者の道楽でしかないといわれていたのですが，どうも大きくなるのではないかと思っていました．

そんなわけで，第５世代のようなプロジェクトをやっているのですが，最初に，知識情報処理という方向が大切であるとするなら，まず小さいものでもいいから，エキスパート・システムを何十と作って，有効性をださないといけないのではないか，ＩＣＯＴの計画にはそのへんが希薄であると批判を受けたのですが，しかし，それはそう簡単に信じてはいけないと思いまして，国のプロジェクトとしてふさわしいものは，すぐに役立たないような部分であるということでやってきたわけです．幸いに，ここ１－２年，エキスパート・システムというものが，小規模ですが，現在の技術で実用化というほどではないにしても，作られてきている．

私個人史から，ハードからソフトという大きな流れがありましたが，個人史とはべつに，こうした動きの結果として，ＳＥＡのような団体ができたと思うのです．そのなかには，ソフトウェアの高度化とか，未来の展望もあると思うのですが，我々のプロジェクトは，９０年代のことをめざしているわけでして，できるだけ，８０年代のことはしちゃいけないと思っているわけですが，世の中としては，８０年代を生きのびていかないと，次の時代がないわけですから，そのへんを期待したいとも思います．

しかしいずれにしても，８０年代の後半に，エキスパート・システム，人工知能という方向性がでてきているのは，非常に重要視していいのではないかと思います．人工知能というと，かっては，まじめなソフトウェア技術者から，ひんしゅくをかっていたのですが，そう毛嫌いする必要はないのではないかと，だんだんわかってきたのではないかと思います．

最近のＡＩブームで私が感じるのは，このブームを支えているのは，普通でいうユーザ畑にいる人たちで，技術動向をみながら，非常によく勉強しているようです．一部には，ＡＩはブームが先走っていて，限界があると，警告を発する必要があるといわれていますが，現実は堅実で，そうしたことを承知していて，ＡＩ的技術をとりこもうとしているようにみられます．

今日のような状況は，数年前には予測されなっかたと思うのです．また，これから数年先は予測がつかないと思いますが，しかし，大きな流れはなにかというと，ソフトウェアの技術は，ひとところにとどまらない，ということだと思います．

辻岡： 先程設定した将来の方向ということについては，量とか質の面に関するご意見は，だいたい均一しているのではないかと思いますが，問題は，ソフトウェア技術者にどのような命題が，今日から未来にかけて，課せられるかということを議論していきたいとおもいます．

先程の鳥居先生の話のなかにありましたが，ソフトウェアの作成能力というものが，旧来的な労働集約型でいっていいのか，そうでなく，別の技法が今後より重要視されてくるのか，そしてこれからのソフトウェア需要をどうカバーすればよいのか，という点について，まずは，鳥居先生からおうかがいしたいと思います．

６．足元を固めてから

鳥居： 私どもは，物事にたいして，シンセシスとアナリシスと２つあると思ってまして，どうもアカデッミックなとこにいますと，アナリシスのほうが論文の生産性がいいということもありまして，アナリシスにかたよりがちなのですが，現実をもっと，きちんと見直さなくてはいけないと思っています．

一方では，Ｌｉｓｐマシンで，デバッキングのエキスパート・システムを考えてはいますが，どうも現実をみ

てみると，まだ足元がぐらついているのではないかと思ったりしてまして，９０年代以降はそれなりの人にまかせて，私たしたちは，もっときっちりしたことをやらなくてはと思うのですが．

辻岡： では，このへんで会場から質問を受けたいと思います．

７．２１世紀のソフトウェア技術者とは

佐原（野村コンピュータ・サービス）： 淵先生におうかがいしたいのですが，２１世紀のソフトウェア技術者はどうなるかをお聞きしたいのですが．といいますのは，今の小学生の９０％は，なんらかの形でコンピュータに触れているといわれてまして，彼等が担う２１世紀はどうなるのかということです．

淵： 本当にどうなっているか，ということは誰にもわからないことだとおもいます．今の小中学生は，日常的にコンピュータに触っていますが，これが１０数年前ですと大人がやっていたことを，楽々とやっている．その子供たちがＳＥＡに入るかは疑問だと思います．これはいい意味でして，それだけの力をもっているけれども，別の分野にいく子供たちが増えてくると思います．

現在のレベルでのソフトウェア技術者というのは，２１世紀になるとソフトウェア技術者とはいわれなくなると思います．つまり，今の技術レベル程度は，皆ができるという時代です．では，２１世紀のプロ中のプロ，つまりプロのソフトウェア技術者というのは何かと考えますと，かってのソフトウェアの蓄積である遺産や，最先端の研究の技術のことも知っていて仕事ができる，というものではないかと思います．

現在のレベルよりさらに高いレベルでの，本当のプロフェッショナルなソフトウェア技術者はなんであるか，ということは，ＳＥＡのひとつの仕事かな，と思うのですが．

８．技術者とマネージメント

鈴木（構造計画研究所）： 戸田さんにちょっと話題をかえてお聞きしたいのですが，企業のなかでの技術者とマネージメントの問題についてです．通常の企業ですと，比較的スムーズに技術者がマネージメントの世界に入っていけるようですが，そうした意味で，ソフトウェア技術者とマネージメントの世界とのつなぎは，積極的に展開を求めるべきなのか，それとも自然に育つのを待つほうがいいのかをうかがえればと思います．

戸田： ソフトウェア技術者のマネージメントというのは，正直いって，私どもも毎日頭を悩ましています，というより，たえず考えている．そして，やはりむずかしいことが多いんで悩んでいる，という問題なんですが，さきほど淵先生のお話にもありましたけど，プロフェショナルなソフトウェア技術者はどうあらねばならないのか，２１世紀まで待たなくても，今現在どうならなくてはならないかを，実は社内で，技術者にいつもなげかけている問題なんです．

よくいわれるように，コンピュータのソフトウェアの世界は，プロとアマの区別をどうやってするのだろうか，何をもってプロとするのか，非常にむずかしい．私はこれに対して，正解はないが，だからこそたえず意識して，すこしでも解決の方向をみいだしていかなくてはならないと思っている．

９．技術者はどうあらねばならないか

私自身，いまでもシステム・エンジニアだと思っていますが，設計者をめざすかぎり，コンピュータの技術者には定年はないと思っています．むしろ年輪があると信じて疑っていない．ただ，一年たったからひとつ年輪が増えるとういことはなく，そこには厳しさと，自己管理の目標がないといけないと思っていまして，これを私の指導原理としています．

ただ，ソフトウェア技術者の中には，非常にユニークな個性を持った人が，他の世界よりは多くみられるのではないかと思います．その人たちには，集団で指導するのでなくて，自分自身でソフトウェア技術者として，何をどうやっていくのかを，本人自身ができないと，プロになりそこなってしまうといっています．こうしたことをしないと，１０年，１５年コンピュータをやってきて，何となくメシを食っているということになってしまう．

いま，世の中では，２０万，３０万のソフトウェア技術者が不足しているといわれていますが，本当に必要な２０万人というのは，なにがしかを持っていて，そして年輪をきざんでいく人であろうと思います．

辻岡： 時間もなくなりまして，予定していた討議がすべてできなかったようですが，このへんで終わりにしたいと思います，普段おいそがしいかたにこうして集まっていただき，有意義な討議ができたのではないかと思います．最後に，パネリストのかたがたに感謝の意味で，盛大に拍手をお願いします．（拍手）

（レポータ：野辺良一）

ソフトウェア産業の現状と将来

静野　哲雄
評論家

１．はじめに

この地球上に，これまで数多くの産業が生まれ，育ってきたことは衆知のとおりである．その発展の推移を見ると，石炭，鉄鋼，電気製品，自動車ｅｔｃ，多少，形態が異なる面があったとしても，マクロ的な見地からすれば，成長するまでの時間の差はあれど，その経路は類似することが多い．

例えば，自動車産業で見てみよう．はじめ鋭い感性を持ち，そのうえ，豊かな知性に富んだ，フロンティア精神に溢れた研究者が，「必要は発明の母」の諺のたとえではないが，発明した．これを先見性のある事業家が企業をおこし，オモチャのような機械を発明した．これを先見性のある事業家が企業をおこし，大量生産に入ると，これが知的研究開発型から，たちまち労働集約型の体質に変わってしまう．ソフトウェア産業は，いまやまさにこの状況かにおかれているのが現状だ．

２．ソフトウェア産業の現状

各企業とも人が不足している．優秀な人材がほしい．こういう環境がなにを生むか？経営者は，効率化，合理化に走り出す，機械化，生産部門の強化が，それぞれの会社にとって必要不可決な条件になってくるわけだ．鉛筆，紙，消しゴムなど，これらの道具を使って，すべての作業を手で行っていた時代から，端末で開発するような環境へ変わっていくわけである．

そうなるとそれぞれの企業とも，知的な研究開発部門，生産工場，営業販売といった各事業部が明確になってくる．ソフトウェア産業界において，本当の意味での淘汰が始まるわけだ．現在は，そのスタート時点にいるといってもよいであろう．もちろん，このように時代が変化すれば，プログラマ，ＳＥなどソフトウェア技術者に対する会社の要求も当然のごとく変わってくる．

ソフトウェア技術者諸君よ！いままさに自分の最終ターゲットを見つめ直す時ではないか．いままで，どのようなフィールドで仕事をしてきたのか，そのスタンスに誤りがなかったか，どのような仕事をしてきたのか，機械化が進むとそれがどう変わるのか．ソフトウェア開発工程についての伝統的な考え方まで一新してしまうようなものではないか．

３．将来展望

ソフトウェア技術は，他の産業の技術と違うよと多くの人は言う．しかし，「違うよ」という言葉の意味を真剣に考えた人が，ソフトウェア技術者の中にいるだろうか．どのような産業も，それが成熟していくと，研究・開発・設計といった人間の感性，独創性，創造力を要する部門と，標準化・規格化された部品，パーツを使い製造するライン，生産工場を持つのが普通である．

「他の技術とは違うよ」という人の立場で考えてみると，このようになると思う．つまり，ソフトウェアというのは狭義にとらえ，モジュール化，パーツ化しようとしても，部品にはなりにくい，作文技術ににているところがある．

しかし，私はこれはまったく間違った言い分ではないかと思う．言語があって文法的なルールがあり，もちろんビジネス業務にも規則がある．それならば，むしろ大きな問題があるとすれば，ユーザの要求が個々に違うことと，言語，ルールがいくつもあることにある．それとても，この業務処理にはどのようなソフトウェア，プログラム・パッケージを使ったらよいのか，その評価能力がソフトウェア技術者にあれば，標準化，規格化はできる部分がかなり多くあると思う．

つまり，これからのソフトウェア技術者に要求されることは，大きくわけると２つある．

１つは，どのような業務処理に使うソフトを製造しているのかがハッキリとわかる人で，しかも，その業務処理に対しユーザは，どのような処理手順と結果を依頼しているのか，理解できるだけのノウハウ，知性を持っていること．そして２つめには，そのソフトウェアを作るのに，ツールとしてのプログラムをどれだけ用意すればよいのか，評価能力と独自設計が必要なところが幾つあるのか，これらが完全に理解できる人ということになろう．

いまや，ソフトウェア技術者は，極端に言えば研究・開発・設計者になるか，職人，工員になりたがるか，どちらを選ぶのか問われているのである．

ソフトウェア産業キキ論

北条　正顕
シスプラン

１．はじめに

ソフトウェア産業と聞けば，ソフトウェア危機という言葉を思い出してしまう．堅苦しい産業論は隣り近所のページに譲ることにし，なぜ危機なのか，どうすればＳＥＡ会員が幸せになり，かつ危機を救えるのかを考えてみたい．

２．奇奇なる危機

ソフトウェア技術者，つまりソフト屋さんの絶対人数が足りないと云われて久しい．われわれソフト屋業界だけではなく，エンド・ユーザ内のソフト屋も含めれば，百万の単位の人間は存在すると思えるのだが，バックログは増える一方である．こなせる仕事量が要求される仕事量を大幅に下回ってしまい，このままの状態では全国民をプログラマにしてもこなしきれなくなる．さて，困ったということらしい．

なぜ仕事が溜ってしまうのか．それは，納期を守るソフト屋が皆無であるからだ．納期を単なる目標としか考えていないようだ．１ヶ月遅れなど良い方でひどいのになると年の単位だそうである．この遅れが重なり，次々に仕事が溜ってしまう．危機の根元はここからはじまっている．各々の仕事に予定（見積り）をかなり上回る工数をかけているのではないらしい．１つの仕事単位で工数をみれば予定とさほど変わらない．しかし，積もり重なってしまうのだ（土木・建築の世界では１日の遅れは受注金額の約２００分の１のペナルティになるそうだ．われわれソフトの世界はアマイものである）．

遅れが遅れであるうちはまだよいが，次の仕事を失敗させる要因となってしまう．次に予定されていた仕事の一番重要な最初の行程が，前の仕事の遅れ（追い込み）でおろそかになってしまう．この無理が後々の工程にひびき失敗するのである．

この遅れと失敗がソフトウェア危機の実体なのだ．

３．ソフトウェア危機を救うには

この危機を，お上では「シグマ」計画なるもので，生産性を上げて対処しようとしているらしい．人員の増加に頼らずに開発環境を整備・充実させて生産性を上げようと云う意図は大賛成であり，できあがったおりには，ぜひとも利用させて頂きたく思っている．

しかし，もっと金のかからぬよい方法がある．それは，現在のソフト屋の約６割にお引取りを願って，残りの４割だけにする．国鉄で考えている以上の人員整理である．

ＳＥやプログラマの質の差は，優秀な者とそうでない者とでは，４倍とも３０倍ともいわれているが，いずれにせよ，かなりの差があることは，万人に認められている事実である．しかし，われわれソフト屋の世界でも年功序列の給与体系（これは組織体系につながる）が，依然として守られている．質の高い人もそうでない人も，入社何年でいくらと決められておりほとんど差がない．同期の人の倍の仕事をしても（残業時間のことではアリマセン！），決して倍はおろか１．５倍ももらえない．また，年上の技術者の倍の仕事をしても，年寄りには追い付くことはできないようになっている．

このような体系がキッチリ守られ続けると，良いソフト屋の背中の「ヤリ貝」はどんどん縮んでいく．ただでさえ他人の倍の仕事をこなせるのだから，他人と仕事量を合わせ，１日にほんの１～２時間しかまともに頭を使わなくなる．そうしていれば自然と技術力は低下し，ただの人になってしまう．

また，質の差が大きい者どうしがチームを組んだときも，よくないことが起こる．質の低い方の者が技術力を上げればよいのだが，反対に高い方が低い方に合わせてしまったりする．こうなると生産物の質も問題になり，踏んだり蹴ったりである．

だから，質のよい優秀なソフト屋４割を残し，かれらだけで仕事をしてもらうのである．ソフト屋の総数は減っても全体の仕事量は変わらないから，給与は上がる．優秀な者だけでチームを組むので出来上りは速く質も高い．おまけに，競い合うことにより全体的規模で技術力が向上する．

少数は精鋭になりうるのだ．人員が増え過ぎておかしくなった例はいっぱいある．ソフトウェア業界ももう一度見直す必要がある．

雑観論的「産業論」

佐野　美知夫
システム工学

１．はじめに

ソフトウェア産業論について何か書くように，という依頼をＳＥＡＭＡＩＬ編集部の芝原さんから受けたのは，会社をかわってから，しばらくたったころであった．その依頼を受けてから，原稿のことはほとんど完全に忘れていた．今，改めてこのテーマについて考えてみると，何を書いていいのかまったく途方にくれてしまうところである．退職した理由にしても，決してソフトウェア産業論的考察のもとに判断したわけではない．どこにでもある私的な理由にすぎなかったからである．ここは居直って，私的理由を最大限に重視した立場からの業界雑観を書くことに決心した．

２．まず重要なこと

雑観論的「産業論」でまず第一に重要なことは，いかなる産業もその隆盛を未来永却に存続させることはできない，それぞれがライフサイクルを持つ，ということである．たとえば，鉄・アルミ等，過去の一大産業群は，大きな雇用を誇り経済をリードしてきたが，今や昔の面影はない．

もちろんソフトウェアの業界も，この原則を逃れえるものではない．たしかに，ソフトウェア業界は今や隆盛の一途を歩んでいる．しかし，このようなときにこそ，自らの将来を展望して，その立場をチェックすることは，大切なことではないだろうか．間違っても，鉄鋼業界のサルマネをして，「ソフトウェアは国家なり」等と口走って有頂天になってはならないだろう．その時から転落がはじまるハズであるから．

ここでのテーマは，「ソフトウェア業界は，いつ，どのような形で飽和状態に入るのであろうか」である．しかもこのテーマに対して，かならずしもお互いに論理的なつながりはないが，興味あるいくつかの項目をとりあげて，論をまとめてみたい．

３．時代の流れ

一番目の項目は，産業社会のソフト化という言葉で表わせる時代の流れである．重厚長大型産業から，軽薄短小型産業への流れである．この表現法は，対象となる製品・商品の特性をとらえたものの見方である．

しかしながら，ソフトウェアを業とするものは，この軽薄短小の中に入っているとも思えない．われわれの作る製品は，ゲームソフトに見られるごとく軽薄ではあっても，決して軽くも，薄くも，短くも，小さくもない不可視なものであるからである．しかも，軽薄短小型産業を裏から支えている立場にあるてんで，損をしている面もある．いわば裏方産業といえる．そして重要なことは，軽薄短小型産業の次にいるのは，具体的な「物」にこだわらない新しい産業群である（いわば裏方産業的特性の強い産業）ということである．

今年の経済企画庁の発表によれば，紀元２０００年には全雇用人口の３分の２が第３次産業に就業することになる，とのことである．このことは重要な意味を持っていそうである．「生産」という言葉が農業だけを意味していた時代，その後工業も生産と認められ，更にこれからはそれらの生産自体がソフト化（知的化）される時代にきつつあるということである．そしてそのような新しい形の産業が，膨大な雇用人口をかかえ，名実共に一大産業群を構成しようとしているのである．

因みにソフトウェア業に限ってみても，米国のシンクタンクＡＤＬ社の研究員の報告にあるように，このままソフトウェアの膨脹がつづくなら，紀元２０２５年には全世界の人々がプログラマにならなければならなくなる，とさえいわれているほどなのである．

４．ソフトウェアの未来はバラ色か

そのような成長力を秘めたソフトウェアを業とする業界の未来は本当にバラ色なのか？またそれだけの潜在力を業界は持っているのだろうか？

ソフトウェア業界の有力な経営基盤の１つに，技術力や技術者のスキルがあげられている．しかしながら，ソフトウェアの技術は属人的性格が強く，とても個々人の技術を組織として結果・統合していけそうにも思えないし，また現実的にも組織化された高度な技術を持った企業をまだしらない．

「技術者集団」という営業用のコピーをよく目にすることがあるが，その実態は，組織化された集団というよりは，単なる技術者の集まりである場合が多いのではな

いか．ソフトウェア業が誇るのは，技術者達の技術的成果物であるよりは，技能者の動員力であったりする．いわゆるソフトウェアの技術は，組織になじまない側面も持っている．

顧客から頼まれたプログラムは公的な性格を持つものに違いないが，それを作るプログラマの考えは外からは測りしれないし，またその成果たるプログラムも会社・組織として上司が査閲・承認するわけでもない．その結果，プログラミングはかなり私的な面をもつものとなってしまっている．

このような状況では，ダグラス・マグレガーのＹ理論（人間を性善説的にとらえ動機づけていく）にたよらざるをえないのが常態であるが，実際にはプログラマの管理は困難となり，かくして業界では企業の分裂・離合集散が多発することになる．

プログラマ達は，伝統的な組織の縄縛から解放され，さらには，場合によっては自身の受注単金を推定でき，その利益配分がわかるようなケースも出てくるようになり，次第に伝統的サラリーマン・カルチュアから逸脱しながちなものになる．プログラマの人達の中に自由業的感覚をもつ人達が割合多いのもこの為である．

５．仲間意識－ＳＥＡ

ＳＥＡという協会は，そのような意識を背景に生まれたもので，シンボル的意識を有しているように思われる．ＳＥＡは，会社組織・企業内組合のいづれかにも関係を持たない自由人の集まりである．「われわれ」「うち」意識の強い日本の組織風土の中では，きわめてめずらしい組織といえよう．

そのようなＳＥＡで，ソフトウェア産業論を試みようとすることの真意は何なのだろうか？ここに到って，議論の方向づけにとまどいを感せずにはいられない．ＪＩＳＡの会員としてなら，ソフトウェア産業論を展開すべき視点はかなり限定的なものとして決まってしまうだろう．１プログラマとして業界をみたときには，産業論など考えたくもないテーマである．というのは，私は「産業」といったときには伝統的な鉄鉱とか自動車産業といった，大きな裾野をもった富士山型の業界を思い浮かべるだけの頭しか有していなかったからである．

無数のソフトウェア業が林立し，陣取戦よろしく人集め競争に狂奔している姿をみれば，誰だってソフトウェア産業論を考えてみようとは思わなくなるだろう．まるで戦国時代の領土争いに精を出す欲得丸出しの企業群を思い浮かべるだけの私の見方は，余りにも浅薄に過ぎるのだろうか？いやいや，戦国時代の武士も大変であったはずである．

多くの武士は，武術の研さん・部下の掌握・管理に頭を悩ませていたに違いない．また，領主の力量によっていつ浮草のように流されるのか分からない不安定な立場にあったことであろう．なんとなく，現在のわれわれの立場と，一脈相通じるものがあるように思われる．

そのような武士達が，その仲間意識から横断的な（領国にとらわれない）組織を作ったとしら，（ちょうど明治維新時代の日本全国の志士群のように）領主達にとっては大変迷惑なことかもしれない．

ＳＥＡは領主に団交を迫らない（労働組合ではないのたから）．しかしながら，皆が集まるのだから何かがあるに違いない，と不安に思うにちがいない．その通り何かがあるのです．何か？答えは一つ，「時代の流れ」なのである．

少なくとも日本では（学会は別として），ＳＥＡは数少くない横断的組織の一つであることは明らかである．日本のサラリーマン・カルチュアにあいた小さな風穴である．そしてこのことはＳＥＡの会員自身が認識しなければならない重要な点である．

６．ソフトウェアの拡大・需要

ソフトウェア業界の中の，最大の経営資源であるプログラマ世界の特徴的な事項は一応理解できた．しかしながら，ソフトウェア人口が増えて，ＳＥＡのような仲間組織が生まれたことを再確認してみたところで，ソフトウェア産業の未来はまだ見えてこない．やはりその需要のゆくえが重要な要素になってくるであろう．

ソフトウェアの需要はいろいろな分野にまで拡がっており，今後もどこまで拡がるかは説明に窮するところがある．人間社会のあらゆる領域に拡がっていくという言い方しかできない．人間が考える存在である限り，ソフトウェアは要求される，人間の考えが及ぶ範囲のあらゆる領域に入っていくことであろう．というのは，ソフトウェアは「考え」であり，「考え」がソフトウェア産業の基盤となっているからである．

もちろん，人間は「考え」だけで生きられるわけではない．たとえば「食」に当たる産業も必要である．しかし，それらの「食」の産業（たとえば農業・鉄）も今や産業としては成熟し飽和状態に入っている状況にある．同様に現在，成長盛りのソフトウェアもいつかはそうな

るとみなければならないだろう．

それがいつかは，潜在需要・技術革新等のファクタによって決まることである．私は，需要面だけからみると，ＡＤＬ社の報告の数字を考えて（根拠はないが），少なくとも紀元２０２５年までは，年率２０％の成長は期待できるのではないかと夢想した．

７．ソフトウェアの生産技術

一方，ソフトウェアの生産技術そのものにも，技術革新がないわけではなかろう．２０２５年まで現在のソフトウェア生産技術が生き延びているとは考えられない．革新的な技術は，かならず実現されるに違いない．

それは，ＡＩ技術の適用によるものかもしれないし，関数型言語・論理型言語等々の技術によるものかもしれない．あるいは，直接操作法等のインタフェース面の技術によるものかもしれない．それ以外の技術によるものかもしれない．

いづれにしろ，ソフトウェアの生産性が飛躍的に向上するような時機がまもなく来るのではないかと思われる．そのための「核」になるような技術の芽は出ているように思われる．

ソフトウェアを作る技術は，人間の頭脳とコンピュータの頭脳との和に依存している．人間の頭脳は優秀であるが，その能力には人によるバラツキがあり，１日に８時間位しか正常動作が期待できない．コンピュータの方は，ソフトウェアの生産といった面では，能力的には低く人間の補助的作業を部分的に行っているだけである．ただし，間違うということはほとんどなく２４時間労働に耐えられる特徴がある．

ソフトウェア産業の資源は，この２つでありほかには何もなさそうである．人間の能力を向上させるには教育とか経験とかが考えられどこでも実践されている．が，その効果は常に限定的であり大幅な改善をすることは困難そうである．

一方，コンピュータの方は，その潜在能力は大きく，今後はとくにその能力の活用が大事である．そのための方向が，前記のいろいろな「核」となる技術で示唆されるものであろうと思われるわけである．いずれにしろ，これも今後の人間の頭脳の成果を持つより他にはないわけである．その解決策が実現できるまでは，ソフトウェア産業の膨脹は続くはずである．

新しい技術によってソフトウェアの生産性が進展すれば，ソフトウェア産業の飽和状態は急速に進むであろう．もちろん，それでわれわれの仕事が直ちになってしまうわけではなく，産業としての成熟を深めつつ，成長率の鈍化が顕著になり，やがてはゼロ成長へ達するということであろう．

多くの企業は，独自の分野を自力で開拓していくことができる．当然，プログラマにもそれに対する適応力が要求されることになろう．

このような経過を通して，ソフトウェア業界は次第に衰退の途をたどっていくことであろう．これがソフトウェア業のライフサイクルである．１プログラマにとって，自らの業界の衰退までを書くことはしのびない（まことに「諸行無常」という気がしてくる）．したがって，この話はここで止めることにする．

８．最後に

以上のような経緯で展開していくであろうと思われるソフトウェア業界も，「産業」としてみた場合には，伝統的産業とはかなりイメージが違うものではなかろうか．

先にソフトウェア業界の現状を戦国時代の様相にたとえたが，この状態は今後も続き，巨大な独占的ソフトウェア企業は依存しえないと考えられる．むしろ，企業組織よりも，ＳＥＡのような任意団体組織の重要性が高まってくるのが時代の流れというものであろう．数多い企業群の中で，その地位を安定・向上させ中立的立場でその職業的・社会的貢献を果たせるのは，このような組織しかないであろう．しかしながら，ＳＥＡの会員は，企業人としての身分をも有するから，企業側の利害との調和も要求されることになり，その道はかならずしも平坦ではないかもしれない．

一方，企業人としてのプログラマは，来るべきソフトウェア生産技術の技術革新に備えて，自己の将来像を明確に描いておかなければならないであろう．さもないと，このように雇用人口の多い業界で一端人手が余り始めたら，たちまちのうちに不況感が強まるのが労働集約型産業のパターンであるから，中高年（あるいはそれ以前）になって，余剰人員扱いを受け路頭に迷うことになりかねないからである（オドカシではない．社会の進むテンポは意外な程速いものだから）．

ソフトウェア業界についての雑観的産業論を展開してきたが，まだまだ議論すべき項目は多い．企業規模，ソフトウェアの経済性，等．それらの項目は，別途議論を展開していく必要があろう．読者からの反論やコメントを期待したい．

辺境の地からのソフトウェア産業論

熊谷　章
パナファコム

１．何のために働くのか

パナファコムというコンピュータ産業の辺境にいる者が，ソフトウェア産業論という大テーマについて論ずるのは，ミスキャストにちがいない．しかし，辺境の地からのメッセージとして受け取ってもらえれば幸いである．

まず日頃，ソフトウェアで働く者として感じている辺りからはじめてみたい．

ソフトウェアの稼ぎ人は何のために働くのか．答えはきわめて明瞭である．金のためだ．いや，最近では自分のため（自己の実現欲を満足させるため）に働く．

ソフトウェア産業の稼ぎ高はどのように計算されるかといえば，３つの方法に大別される．１つは人月計算であり，いま１つは作成されたプログラムサイズに依存する方法である．残ったもう１つの方法は，人月やプログラムサイズに依存せず，機能そのもので売買価格の決まるものである．

最後の機能単位で売買する方式は，パソコンにより活性化された流通ソフトウェアに多くみられる．その代表例が，ワープロ，各種表計算，ゲーム等である．この流通ソフトウェアの価格は市場価格であり，小生のような門外漢のよく知らぬ部分であり口出しできない．第三者から見れば需要と供給のバランスが成立する適正価格のように思える．ようするに良いソフトは売れるのである．

これに反して汎用マシン，ミニコン，オフコン等で多くみうけられるソフトウェア開発の方法が，人月計算やプログラム・サイズ方法を採っている．理由はごく簡単で，この二つ以外に，定量的に開発したソフトウェアを計る手段がないのである．

この方式はおかしな現象をもたらしている．結論をいえばこうである．ソフトウェア産業で儲けようとすれば，あまり腕のよくないプログラマを沢山雇い，よい商売相手をみつければよい，ということである．

なぜなら，ソフトウェアの売り上げは前述のように，所要した人月（工数）とプログラム・サイズに線形に依存しているからである．この例を示してみよう．

２．例による考察

ある仕事Ｘを完成するのに，腕のよいプログラマＡが６ケ月で３Ｋステップのプログラムを作成したとする．一方，腕のよくないプログラムＢは同じ仕事Ｘを完成するのに，１０ケ月で６Ｋステップのプログラムを作成したとする．

一人月の単価を１００万円，１ステップの単価を２０００円とすれば，Ａ氏の稼ぎは，６００万であり，Ｂ氏の稼ぎは，１０００万円か１２００万円となる．稼ぎ高からすれば，Ｂ氏の方がＡ氏よりはるかに優れている．しかも，Ｂ氏は長時間かけるため，仕事を長持ちさせることができる．マネジャとしてはＢ氏はなにも欠点のない神様にみえてしまう．

多かれ少なかれ，人間のなせる技は，この程度なのかもしれないが，なにか変であり，気持悪い．こう思えるのは，小生だけではあるまい．

この珍現象を支えているソフトウェア産業の構造は，なんだろうか．その特性として考えられるのは，やってみなければ分からない部分が多すぎることである．施工前に精確に見積り，それに基いた作成など不可能なのである．したがって，最初は粗い見積りをし，それに基いて施工し，出来上がった時点で補正するしかない．

結論からいえば，出来高払い制度になる．出来高払いの許容度がマネジャにとって，よい商売相手になるか否かのバロメータになる．かくして，嘘のような真実のソフトウェアの稼ぎ高がきまる．

３．アートとインダストリ

先日，トヨタ自動車の工場を見学する機会にめぐまれた．インダストリの極みをみるような思いをした．

人間の作業の限界と思えるような速い速度で，ベルトコンベア上を車が動き，並んでいる作業者は，次々と自分専用のツールと部品で各自の役割を果たす．作業者は交代制度を採っており，誰がやっても同じような精確さと品質で車はでき上がる．

感慨は二つあった．一つは，「ソフトウェア・インダストリは，絶対こんな工場にならないなあ」，という思いと，「とてもこのような作業は自分にはできない」，という実感である．

車は，目的とする物，道具，部品のすべてが目に見え

る．したがって，それらの物に関するエンジニアを作業者は，直接それらの物に触って学び習得することができる．

これに対し，ソフトウェアではすべてが目に見えない．手にとり感触を通して学習することは不可である．したがって，自動車の作成工程では途中で作業者が代っても，対象物を見るだけで，何をどうすればよいか即座に判断できる．ソフトウェアではこうはいかない．作業者が何をやっていて，どこまでできあがっているかは，即座にはわからない．規模が大きいプログラムになると半永久的にわからない場合もある．

社内のハード部門の人は，われわれソフト部門の人々に対してよく次のようなことをいう．ソフトは目にみえないからまるでわからない．ただわかるのは，ソフトウェアそのものではなくて，ソフトウェアを作っている人間しかみえない．だから，よいソフトウェアか否かを判断する基本は，そのソフトウェアを作成している人しかない．よいソフトウェアとは，よいエンジニアか否かである，というのである．ソフトウェアを外部からみればもっともな話である．

こうして考えてみれば，ソフトウェアを部品化し，工場で自動車のように組み立てて，製品を作り出すのは至難の技にみえる．ソフトウェアはまだまだインダストリではなく，むしろ，個人の能力に依存したアートといったほうがよいように思える．

４．プログラマと作家

比較として適切でないかもしれないが，プログラム作成は，まるで作家が物を書くようにみえる．そのとき使用する言語は，アプリケーションによってことなるが，それは，言葉で意味を表わす作業そのものであり，作家のレトリックはソフトウェアの技法とほとんど同じである．

もともとアートには，専門家の技という意味があり，ソフトウェアは正にアートの感じが強い．そうすれば，もともとアートのものを，あたかもインダストリとして無理矢理実践するところに無理があり，前節の神話が生まれるのではなかろうか．

ソフトウェアは，個人の能力に依存するアートなのだ，という悟りを開いた途端，ソフトウェア産業の運はひらけてくる．

５．ソフトウェアとハードウェア二元論

デカルトが人間の二元論を唱えて以来，人間は二元論に悩み続けてきた．それは，肉体と精神の分離についてである．コンピュータも同じことがいえる．二元論に毒されている人間の創るものが，肉体と精神という二元論に基いているのは当然すぎるわけである．

哀しいかなソフトウェアとハードウェアは運命共同体であるが，別々の途を歩んできた．人間と同様にソフトウェアは我思う故に我有り，に気付かなければこの二元論から脱却できない．その根本理念は，意識は存在を規定するということだろう．

コンピュータにおける二元論とは，具体的に何を指すのか，きわめて原始的レベルでは，ハードウェア命令とコンピュータ言語の関係がある．ブール代数をベースとするチューリング・マシンであるハードウェアは，汎用的であることが証明されている．用途を決めるのはソフトウェアである．そのソフトウェアでは，用途を決めるアプリケーションは，それを記述するコンピュータ言語にかなり依存する．

ハードウェアとコンピュータ言語間には，常にかなりの距離があり，それを埋めるのが大変である．この距離が二元論そのものである．

もっと社会的な意味での二元論はなにか．これは，コンピュータ・メーカとソフトウェア・ハウスである．ここでも，先にみたハードウェアとソフトウェアと同じ関係が成立している．コンピュータ・メーカの作り出すコンピュータ・システムは，その上にアプリケーションを作成すれば，何にでも使用できるはずだ，というメーカ側の意見がある．

一方，使いにくいコンピュータを作り出すから，ソフトウェアを作るのが大変で，ヤル気も起きなくなってしまう．もう少しアプリケーションのことを考えて，ましなコンピュータを作ってもらいたい．しかし，そんなことをいっても，外国の真似ばかりしているていたらくでは何もできないであろう，というのがソフトハウス側の意見であろう．

この両者の関係は，ソフトの稼ぎという神話も中間に介在しているから，きわめて不健康な形で成立している．メーカベッタリ型，メーカ非依存型とあまりにも極端に走りすぎている．

ハードウェアとソフトウェアは両方共にかなり成長した．もうそろそろ二元論を克服すべき時代ではなかろうか．この二元論を止揚すれば，そこには現在のコンピュータ産業とはことなった世界がひらけるにちがいない．

アートとしてのソフトウェアが確立され，ソフトの稼ぎの神話が消え，ハードとソフトの生き別れがなくなり，コンピュータ・メーカとソフトハウスは信頼で結ばれることになる．

６．全体と部分

人間は系統的に発展する動物である．したがって，自分たちだけをすべてと考えて価値を決めてはいけない．現代の様相に，人類六千年の反映をみなければ盲である．つまり，永遠に続いている時間軸に展開されている現象が全体であり，この現在のわれわれの世代が部分に当る．

全体の中における部分の意味を，われわれは，コンピュータという仕事を通して直観できなければならない．われわれは，現在に命を賭けるしかないが，その賭けかたは部分の意味を洞察し，われわれ人間自身のため，系統的に発展している方向にむかねばならない．利益のためだけに命は賭けられない．

コンピュータ・システムを作成するときでも，この全体と部分の考え方が重要である．誰もが全体になれるわけがない．みんな部分である．部分が他の部分をよく理解したときに，はじめて自分自身の部分も活きてくる．要するに，人間に対して，人類に対して，自分に対して，他人に対して，愛と信頼がなければならない．その上で，部分達が寄り集まってコンピュータという物造りを通して，一つのカルチャを生み出さなければならない．存在している物を単に使うだけではいけない．

新しいコンピュータをハードウェアからソフトウェアまで自らの手で作らなければ，そこにカルチャーは生まれてこない．

７．世代交代

ソフトウェア産業は，パソコンの出現により大きなうねりを生じた．それは，一つの世代の到来であり，質的に変化しはじめているソフト産業の兆しである．今やその加速度は誰にも止められない．

われわれは，人類が連綿とやってきた世代交代を通して，ソフトウェア産業そのものを質的に変化させる役割をおっている．このとき重要なことは，「当たり前のことに気付き，当たり前のことを実行する」ことであると感じている．ソフトウェア産業論がソフトウェア精神論になってしまった．辺境の地に住む人種の妄想はこの程度かとお許し願いたい．

今後の誌上フォーラム

特定の論争テーマを選んで，会員相互が意見を交換しあう誌上フォーラムの，９月以降に予定している論争テーマは以下の通りです：

★「情報処理試験について」

毎年おこなわれている情報処理技術者について，賛成／反対，試験の内容や方法の改善，合格者へのフォローアップなど，さまざまな角度から論議してみたいと考えています．

★「パソコン・ソフトをめぐって」

権利保護，不法コピー，プロテクトにからむ技術的なことがら，価格と性能の対比，品質保証などなど，パソコン・ソフト特有の問題点を討論しようと企画です．

★「女性技術者から（へ）の提言」

ＳＥＡの会員のなかでは，まだ５％程度ですが，最近どの職場でも，女性技術者の比率はふえてきています．そこで，ソフトウェアの世界における女性技術者の問題を広く考えてみたいと思います．男女いずれの立場から論じていただいてもかまいません．

これらのテーマへの読者からの投稿をお待ちしています．原稿はＳＥＡＭＡＩＬ仕上がりで１ページ（約２千字）で，編集部（ＳＥＡ事務所）まで，どしどしお寄せください．

また，これまでに掲載してきた誌上フォーラムの論文に対する反論や意見もお待ちしています．

ＣＯＢＯＬ８５の使い心地

渡辺　雄一
電力計算センター

１．はじめに

昨年暮に，アメリカでＣｏｂｏｌに関する規格が正式に確定した（８５ＡＮＳ規格）ことにともなって，いちはやく，国内のメイン・フレーマもこれに対応した（富士通では８６年１月より出荷開始）．私の勤務先では富士通の大型汎用機を中心に使用している関係で，富士通製品の情報にかたよりがちではあるが，ＣＯＢＯＬに関しては，多くの方が関心を持っておられると思うので，何かの参考になればと思い，ＣＯＢＯＬ８５（正確にはＦＡＣＯＭ　ＯＳ０４／Ｆ４　ＭＳＰ　ＣＯＢＯＬ８５）の使い心地を紹介したい．

２．コンバージョン

ＣＯＢＯＬ８５はＪＩＳ－ＣＯＢＯＬ（７４ＡＮＳ）の上位互換を基本的に保っている．しかし多くのユーザは，それ以前の規格（６８ＡＮＳ）でのソフトウェア資産も多く保持していると思われる．

富士通のコンパイラでは，とくに６８ＡＮＳや旧ＪＩＳ規格をサポートしていないので，多くのユーザが，コンバージョンを本格的に実施する必要に迫られている（正確には，コンパイラのオプションで言語規格を選択できるが，言語規格の混用は当然できないので，コンバージョンの必要性がある）．メーカでもこのような事態に，かなり前向きの姿勢で取り組むものと思われる．

具体的には，

・ＣＯＢＯＬ８５サポート・センタの設置

・ソース・プログラム変換コンバータの提供

が発表されている．

なおＣＯＢＯＬは，基本的に上位互換を保っているので，よほどのことがない限り問題は少ないと思う（これと比較すると，ＦＯＲＴＲＡＮは悲惨の一言につきる）．

また今回のＣＯＢＯＬ規格改訂にともなって，メーカがコンバージョンに躍起になっているもうひとつの原因は（多分富士通のコンパイラだけの問題だろうが），オブジェクト・モジュールおよびロード・モジュルの互換性が，今までのＣＯＢＯＬで作成されたものとの間に，保証されていないことにある．たとえば，プログラムＭＡＩＮから，サブルーチンＳＵＢ１を呼びだしている場合に，ＭＡＩＮもＳＵＢ１も，同じコンパイラで作成されている必要がある．これの一つの理由は，拡張空間（３１ビット・アドレッシング）サポート機能にＣＯＢＯＬも対応するためと思われる．

筆者がいままでコンバージョンしたさいの，非互換項目を以下にしめす．

・ID-DIVISION（省略形を認めない）
・SKIPn文，EJECT文
・COPY-SUPPRES句
・TIME-OF-DATE
・READY TRACE，RESET TRACE
・FETCH（予約語の変更）

等々

３．ＣＯＢＯＬ８５の新規格

他人に説明できるほど，充分な知識や経験が私にはないが，試用した範囲での新規格，新機能についてのべてみたい（具体的な規格仕様については，参考文献の〔１〕〔３〕を参照されたい）．

(1)データ項目の部分参照

某いわく，「やっとＣＯＢＯＬも，まともな高級言語の仲間入りをできるようになった」．また，某いわく，「誰かがあのＣＯＢＯＬに，また生きのびる要因を作ってしまった」．

〔記述例〕

```
move " ABC" to A1(1 : 3).
```

(2)外部参照機能

ＥＸＴＥＲＮＡＬデータ（ＦｏｒｔｒａｎのＣＯＭＭＯＮに相当），ＥＸＴＥＲＮＡＬファイル（プログラム間でのファイルの共用）が可能となる．

(3)プログラムの入れ子

原始プログラムに入れ子を認めることで，外部参照の機能と組み合せにより，原始プログラムの部品化に大きな可能性がみえてきた．

〔概念図〕

```
IDENTIFICATION DIVISION.
PROGRAM-ID.  MAIN.
IDENTIFICATION DIVISION.
```

```
PROGRAM-ID.  SUB1.
END-PROGRAM SUB1.
IDENTIFICATION DIVISION.
PROGRAM-ID.  SUB2.
IDENTIFICATION DIVISION.
PROGRAM-ID.  SUB3.
END-PROGRAM SUB3.
END-PROGRAM SUB2.
END-PROGRAM MAIN.
```

ただし，いままでどうりに，ＣＯＰＹ命令の入れ子は許されていないので，やはりプログラムの部品化等では苦しい面も残っている．また再帰的な呼出しも許されていない．

(4)整構造プログラミング

〔記述例〕

```
PERFORM varying i from 1 by 1
            until i > 10
            WHIT TEST AFTER
    move " test" to A(i)
END-PERFORM.
```

これは，いままでのＦＡＣＯＭのコンパイラでは実現されていた．

(5)データの初期化の為の命令（ＩＮＩＴＩＡＬＩＺＥ）

この命令により，データの初期化のための不要な手続きが大幅に削減される．とくに，データ項目の多いファイルを多く使用しているプログラムには有難い．

(6)作業域（Ｗｏｒｋｉｎｇ　Ｓｔｏｒａｇｅ）の大きさの制限の縮小

ちょっとしたＥｄｉｔｏｒなどを，ＣＯＢＯＬで作成しようとすると，すぐに上限（１３１０７１バイト）にひっかかってしまって，これまでいろいろと苦労した．新規格では，なんと，２１４７４８３６７バイトまで拡張されたので，へんなプログラミング・テクニックを用いなくても済むようになった（これはＣＯＢＯＬの文法の問題ではなく，単にコンパイラ側の問題かもしれない）．

また，ＣＯＢＯＬ８５の文法書には，今後の改訂を意識して，不適当な命令には”廃要素”という明確な記述がなされている（廃要素とは，次回の改訂時には廃棄されることを意味している）．このような配慮は，利用者からみて，とても好ましい．

〔主な廃要素〕

ＡＵＴＨＯＲ段落，ＤＡＴＥ－ＣＯＭＰＩＬＥＤ段落，ＤＡＴＡＲＥＣＯＲＤ句，ＧＯＴＯ命令の特殊な書き方，ＳＴＯＰ定数命令，デバック機能（デバック文は残る），区分化機能，等々である．

４．コンパイラの改良点と使い心地（問題点）

今回の私が使用した範囲では，コンパイラの性能等の調査結果を，公表できるほど十分にではないが，気付いた範囲でのコンパイラの使い心地を述べてみたい．

また，各項目に，◎，○，×で，自分なりの評価もしてみた．

(1)コンパイル時間の短縮（◎）

今度のコンパイラでは，作業域をディスク上から仮想記憶上に移した（ディスクにとるワークはＳＹＳＵＴ１のみでＳＹＳＵＴ２～４は不要となった）ので，基本的に，ディスクとのＩ／Ｏがなくなった．

また，いままでＣＯＢＯＬの拡張仕様として用意していた日本語処理，データベース処理（ＡＩＭ／ＮＤＢ，ＲＤＢ）も，ＣＯＢＯＬの文法書に吸収されたこともあり，ＣＰＵ時間，ＥＲＲＡＰＳＥＤ時間がともに短縮した（ＣＰＵ時間で平均で１５～２０％の短縮）．

(2)コンパイル時のリージョンの増大（×）

上に述べたように，作業域の多くをディスクから仮想記憶上に移したために，多くのリージョンを必要とする（ソースのステップ数に比例）．

自分の保有するソース（１０００～２０００ステップ）で，２０００Ｋほど必要とする．また，ＡＩＭ／ＲＤＢの親言語とインタフェース（ＡＱＬを使用してアクセスする）をとる場合は，ＡＱＬの解釈のために，作業域としてさらに２０００Ｋほど必要になる（注：ＡＱＬとはＩＢＭのＳＱＬと同等の言語）．

(3)コンパイル・リストの改良（○）

いままでのコンパイル・リストは，ソース・リストとクロス・リファレンス・リストが別々であったが，今度はそれらがいっしょになった（ＩＢＭのコンパイラでは実現されていたが，それよりもみやすい）．

また，ＩＦ文等による分岐の深さや，ＯＢＪＥＣＴコードのアドレス等もあわせて表示されるようになった（図１を参照）．

(4)実行時の統計情報の充実（○）

いままで，使用できて便利であったデバックの手法に，ＴＲＡＣＥ命令の利用があった．しかし今度の，ＣＯＢ

ＯＬ８５では，この命令が使用できなくなった．そのかわりに，ソース・プログラムをいじるのではなくて，コンパイラにＣＯＵＮＴオプションが用意された．

ＣＯＵＮＴオプションを使用した場合と，そうでない場合では，コンパイル時および実行時ともに，約２倍の処理時間がかかる（図２を参照）．

(5)ブロック化因数の指定が不要（○）

いままでのコンパイラでは，プログラム中で正しくＢＬＯＣＫ－ＣＯＮＴＡＩＮＳを記述しなければ，データセットにアクセスできなかったが，ＣＯＢＯＬ８５では，ＢＬＯＣＫ－ＣＯＮＴＡＩＮＳを書かなかった場合は，実行時に実際にアクセスしたデータセットのブロック長を自動的に判断してくれる（こんなことは，Ｆｏｒｔｒａｎ等の他の言語では，あたりまえだった）．また，ＬＡＢＥＬ－ＲＥＣＯＲＤ句は注釈として扱われる．

(6)初期障害（×）

世の中に広く利用されている言語の処理系を，真っ先に使う機会は，そうたびたびあるものではない．今回のＣＯＢＯＬ８５の試用は，そのような意味で貴重な経験であった．

しかし，そこでの感想はまず第一に，「なんでこんなに，バグがあるのか？」ということである．自分が新規に発見した障害が２件，文法書の記述不足によるトラブルが１件あった．いずれも早急にメーカに対応してもらえたが，やはり製品出荷時点での十分な検査を望みたい．

(7)運用まわり

センタの運用という立場からは，まずＬＰＡ（ＬｉｎｋＰａｃｋ Ａｒｅａ）への（一部の）モジュールの常駐が可能になった（現在筆者のところでは，並行運用中なので，これについてはテストしていない）．

また，予約語セット・テーブルの確認，取捨選択が比較的容易なようにテーラリング，ミラーリングと称する機能が付加された．

５．関連製品の動向

ＣＯＢＯＬの改訂にともなって，関連製品も改訂が進んでいる．が，動作環境として，ＣＯＢＯＬ８５との整合をとる必要があるものと，言語仕様の改訂としてＣＯＢＯＬ８５の影響を受けるものの２通りがある．

前者については，半年たった現在でだいたい整備されてきているが，後者については，あと１年ぐらいかかりそうである．たとえば，ＳＩＭＰＬＥ／ＬＩＮＤＡ（ＣＯＢＯＬのデータ定義部のテスタ）は”ＰＡＣＫＥＤ”を許さない（改訂版は８７年に出荷予定）．

また，プレ・コンパイラは（メーカ製品にとどまらず），本質的な見直しが必要となろう．十分な機能を持っていない製品は，最近のＴＳＳ環境等の充実からみて，プログラムの生産性を落とすし，また，移行を阻害する原因にもなりかねないのではないか？

逆に，ＣＯＢＯＬ８５で恩恵を受けたものもある．たとえば，ドキュメント・ジェネレータのＣＯＢＯＬ８５－ＤＦ（旧ＰＡＧＥ，ＳＮＯＴＥ）では，拡張言語仕様がなくなって，ひとつの処理系に吸収されたため，その機能が実務レベルとして期待できるものにまで高まってきた．

６．終わりに

「いまさらＣＯＢＯＬなんて」という意見が多いことは，筆者自身も，十分ではないかも知れないが，理解しているつもりである．しかし，ＣＯＢＯＬ以外”実用”になる言語が存在しないのも事実だと思う．いまＡｄａやＣを使って，事務アプリケーションを大型汎用機の上で作成することは，まだ半年から１年以上先のことだと私には思われる．

この拙い雑文は，ＳＥＡでの（例えば，健全な言語の在り方みたいな）議論の引き金になることを期待して書いた．会員諸兄の御意見を拝聴したい．

この原稿を書いているときに，たまたま情報処理学会の言語処理研究会で，ＣＯＢＯＬ８５に関する発表〔３〕があった．こちらは識者による立派な内容なので，ぜひ一読されたい．

末筆ながら，ＣＯＢＯＬ８５の試用の機会を与えて下さった（財）電力中央研究所の関係諸氏，ならびに日頃から御協力頂いている（株）富士通の関係者に感謝します．

【参考資料】

〔１〕富士通：ＦＡＣＯＭＯＳ04 ＣＯＢＯＬ８５文法書Ｖ１１用（７０ＳＰ－５８０１－１）

〔２〕富士通：ＦＡＣＯＭＯＳ04 ＣＯＢＯＬ８５使用手引書Ｖ１１用（７０ＳＰ－５８１１－１）

〔３〕植村俊亮：ＣＯＢＯＬ８５－ＣＯＢＯＬ言語の進化，標準化および認証，プログラミング言語６－３，情報処理学会（１９８６．５．２３）

```
1   LINE NST ADDR      SEQNO  A   B

     39                003900/*****************************************************************
     40                004000*                                                               *
     41                004100*                        データ部                               *
     42                004200*                                                               *
     43                004300*--1-B--+----2----+----3----+----4----+----5----+----6----+----7-
     44                004400 DATA                  DIVISION.
     45                004500*********************************************************
     46                004600*                      作業場所節                         *
     47                004700*********************************************************
     48                004800  WORKING-STORAGE     SECTION.
     49                004900
     50   000002B8     005000 01  メッセージ                PIC     X(100).              74S,77S,80R
     51                005100
     52                005200*********************************************************
     53                005300*                      連絡節                             *
     54                005400*********************************************************
     55                005500  LINKAGE             SECTION.
     56                005600                                                           69R
     57   ********     005700 01  パラメタ.                                             62D,72R,76R
     58   ********     005800     03  パラメタ長              PIC     S9(4) COMP.       76R
     59   ********     005900     03  パラメタ内容.
     60                006000       05  FILLER             PIC     X(1)
     61                006100                              OCCURS    0  TO 100  TIMES
     62                006200                              DEPENDING ON  パラメタ長.    58
     63                006300


     64                006400/*****************************************************************
     65                006500*                                                               *
     66                006600*                        手続き部                               *
     67                006700*                                                               *
     68                006800*--1-B--+----2----+----3----+----4----+----5----+----6----+----7-
     69                006900 PROCEDURE       DIVISION   USING  パラメタ.                57
     70                007000
     71                007100 コンソールメッセージ        SECTION.
     72   00000406     007200     IF     ( パラメタ長 = 0 )  THEN                        58
     73 1 00000418     007300        MOVE  "*****     PAUSE     *****     PAUSE     *****"
     74                007400          TO  メッセージ                                   50
     75                007500     ELSE
     76 1 0000043C     007600        MOVE  パラメタ内容(1:パラメタ長)                    59,58
     77                007700          TO  メッセージ                                   50
     78                007800     END-IF.
     79                007900
     80   0000049C     008000     DISPLAY  メッセージ  UPON  CONSOLE.                    50
     81                008100
     82   000004BC     008200     STOP     "+++++     JOB-STEP  PAUSE   +++++".
     83                008300
     84                008400 END PROGRAM   PAUSE.
```

図1

```
COBOL85 COUNT INFORMATION(END OF RUN UNIT)                                    PAGE   1

  STATEMENT EXECUTION COUNT    PROGRAM-NAME : PAUSE

     STATEMENT
       NUMBER          PROCEDURE-NAME/VERB-ID          EXECUTION     PERCENTAGE
                                                         COUNT           (%)
   ------------  ------------------------------  ---------------  --------
           69  PROCEDURE DIVISION   PAUSE
           71  コンソールメッセージ
           72      IF                                        1   25.0000
           73      MOVE                                      0    0.0000
           76      MOVE                                      1   25.0000
           80      DISPLAY                                   1   25.0000
           82      STOP                                      1   25.0000
   ------------  ------------------------------  ---------------  --------
                                                             4
```

```
COBOL85 COUNT INFORMATION(END OF RUN UNIT)                                    PAGE   2

  VERB EXECUTION COUNT         PROGRAM-NAME : PAUSE

                                                    PERCENTAGE                    PERCENTAGE
       VERB-ID          ACTIVE VERB     TOTAL VERB       (%)   EXECUTION COUNT       (%)
   -----------------  ---------------  ---------- --------  ---------------  --------
   DISPLAY                         1           1 100.0000                 1   25.0000
   IF                              1           1 100.0000                 1   25.0000
   MOVE                            1           2  50.0000                 1   25.0000
   STOP                            1           1 100.0000                 1   25.0000
   -----------------  ---------------  ---------- --------  ---------------  --------
                                   4           5  80.0000                 4
```

図2

新入社員教育について

河村　一樹
日本電子専門学校

１．はじめに

教育分科会（ＳＩＧＥＤＵ）第１回の検討テーマは，新入社員教育となった．新入社員として，どのような技能と特徴をもたせるかは，現在のところ各社各様である．というのも，社内のシステム開発における標準化制度や開発用ターゲット・マシンの整備環境，あるいは社員のキャリア・パス（技術移転も含める）の設定方法などは，各社の状況によって異なってくるからである．したがって，教育目標のテーマをしぼって設定することが，むずかしいという面も生じよう．

本稿では，討論会での発表に補筆，加筆したものである．討論会に参加されなかった会員の方々の，ご意見をお聞きしたい（ＳＥＡＭＡＩＬへの投稿，または，教育分科会への参加をお待ちしています）．

２．職種の限定

さて，情報処理技術者の職種というものを大きく分けてみると，ハードウェア指向およびソフトウェア指向というものになる．一般的には，ハードウェア指向の技術者には，ＣＥ（カストマー・エンジニア），ＯＰ（オペレータ）等の職種が設定される．

一方，ソフトウェア指向の技術者としては，ＳＰ（システム・プランナー），ＳＡ（システム・アナリスト），ＳＥ（システム・エンジニア），ＰＧ（プログラマ），ＣＤ（コーダ）等の職種が設定さている．それらの中で，現場の人材不足が最も顕著化している職種といえば，SP，SA，SE，PGといえる．というのも，コンピュータの導入と普及が進み，適用業務の世界でＥＤＰＳ化が進展するにしたがい，ソフトウェア開発に対するニーズが非常に高まってきたことがあげられる．

このようなことにより，現状では，ますますソフトウェア技術者が不足している傾向にある．しかし，ＳＰ，ＳＡ，ＳＥという職種には，多大な経験の蓄積と高度な技能の応用が要求されることになる．通常は，何年も現場で適用業務を経験しながら，除々にそれらの職種に近づいていくことになる．したがって，短期間に，しかも技術として身につけるまで，高度な技能レベルでの教育をおこなうことは，新人教育という範囲からは困難なものになる．このようなことにより，新人教育としての人材育成のターゲットとしては，プログラマという職種に限定することにする．

３．教育目標の設定

以上のような前提条件のもとに，具体的な教育目標を設定することにする．教育目標を明確に定義することによって，教育内容についての方針が設定されることになり，より効果的な教育が実践できるようになるからである．

その教育目標とは，「即戦力としてのプログラマ養成」ということになる．ここでいうプログラマの具体的なイメージは，次のようになる．「与えられたプログラマの外部仕様をもとに，独力でアルゴリズムの設計，および指定された言語によってプログラミングおこなえること．かつ，デバッグをおこない，最終的には高品質のプログラムを定められた期間内に完成することができる人材」である．

高品質のプログラムには，以下のような条件項目が含まれている必要がある．

(1)要求仕様どうりであること
(2)潜在的バグがより少ないこと
(3)理解容易性（検査や保守がしやすく，構造化されている）であること
(4)運用（処理効率が高く，オペレーションが容易）がしやすいこと
(5)高品質のドキュメント（プログラム内容と整合性があり，わかりやすく，標準化されている）があること

このような条件を満足せずに「ただ動けばいい」式のスパゲッティ・プログラムが作れるというだけでは，即戦力のプログラマとはいいがたい．

４．即戦力としてのプログラマ

これらの条件まで充分考慮したうえで，品質のよいプログラムを作成することが，「プロとしての」プログラマといえる．そして，こういう人材こそが即戦力として，位置づけられることになる．そうでなければ，ただたんに低品質のプログラムが大量に生産されることになり，結果として保守工程の作業が増大することになる．これ

ではかえって，全体的な生産性の低下をまねくことになる．

なお，ここでいっているプログラマとは，高水準言語（COBOL，FORTRAN，PL／Iなど），あるいはアセンブリ言語を用いることを前提条件としている．したがって，設計するアルゴリズムは，プログラム言語のロジック形態に対応したものとなる．

上述した即戦力としてプログラマに要求される能力としては，以下のようなものがあげられる．

・プロセス設計能力（ただし，この能力はシニア・プログラマの方に近い）
・ファイル設計能力
・プログラム設計能力
・プログラミング能力
・デバッグ能力
・コンピュータ基礎理解能力

それでは，これらの能力を育成するためには，どのようなカリキュラム構成とすべきかについて，以下から述べてみることにする．

５．基本的なカリキュラム構成

基本的なカリキュラム構成としては，コンピュータ基礎教育（知識習得中心）から，コンピュータ専門教育（技術習得中心）へとステップ・アップしていく形態となる．

前者の知識習得の範囲としては，情報処理技術者第２種試験の午前に出題される内容が中心になる．具体的には，コンピュータのハードウェア知識，ソフトウェア知識，そして関連知識（簿記，経営，数学，英語等）などがあげられる．これらの知識を習得することによって，コンピュータ本体のしくみや構成，そして実際に使用するための手順や方法がわかることになる．

一方，後者の技能育成とは，体得して蓄積していくものである．具体的には，机上作業に依存するものと，マシン作業に依存するものとがある．机上で育成される技能とは，与えられた外部仕様をもとに，独力でアルゴリズム設計（内部仕様の完成），プログラム言語への変換，机上デバックができる能力ということになる．

マシン作業で育成される技能とは，プログラム・テスト環境の設定（領域確保，ファイル生成，ダンプ・リスト取得，プログラム実行等）およびマシン・デバックができる能力ということになる．これらの能力を習得するためには，何度も実際におこなってみて，経験を積んでいく必要があるといえる．

これらのカリキュラムを具体的な科目として設定する．コンピュータ基礎科目としては，ハードウェア概論，ソフトウェア概論，数学，英語，簿記，経営などがあげられる．そして，これらの科目をカリキュラム体系としてタイム・チャート化すると，以下のようなカリキュラム事例となる．

６．カリキュラム事例の問題点とその対応

このカリキュラム事例で問題となる点がいくつかあるので列挙し，その対応についても考えてみる．

(1) 本来目に見えない抽象的なものを，どのように理解させるか．

具体的には，ハードウェアの機構やオペレーティング・システムの機能，そしてプログラムのロジック概念などがある．

これに対しては，パソコンによるシュミレーション・プログラムの実行やビデオ教材を用いたメディア教育が効果的である（たとえば，当校ですすめている「MEDIA－LINKS」の概念適用）．

(2)個々人の技量に対応した学習をすすめるにはどうすればよいか．

知識習得の部分に関しては，全面的にCAI（さらにはCMI）を活用する．また，個人別評価データベースの構築も実施する．

(3)アルゴリズム構築の訓練をどうすればよいか

これは，もっともプログラマの資質として要求される技能である．

パソコンからフローチャート入力による自動変換ツールを活用する．これによって，机上での思考に対する実践的フィードバックがはかれる．また，他人の作成した既存のプログラムを読ませ，メンテナンスをさせる．こうして読解力を養成する．

(4)カリキュラム構成として，プログラム設計とプログラム言語のどちらを先におこなうべきか．

あくまで設計重視の姿勢をとる．ソフトウェア・エンジニアリング的アプローチをとり，設計の重要性を認識させる．なお，設計という抽象度の高い概念の教育を支援するものとして(3)を活用する．

(5)プログラム設計において，ソフトウェア・エンジニアリング的発想をどのようにとりこむか．

プログラム設計を２つに分割する．プログラム設計ではフローチャート中心にしておこなう．これによって，

アルゴリズムの理解を深める．ただし，フローチャートを用いるために，作成されるスパゲッティ・プログラムの欠点を実体験させたあと，構造化プログラミングを中心としたプログラム設計IIを実施する．このときは，構造化チャート（ＰＡＤなど）を用いる．

(5)教育効果をどのように測定するか

国家試験合否状況やＯＪＴによる実績評価などをおこなう．ＯＪＴによる実績評価の基準としては，次のような事項とする．

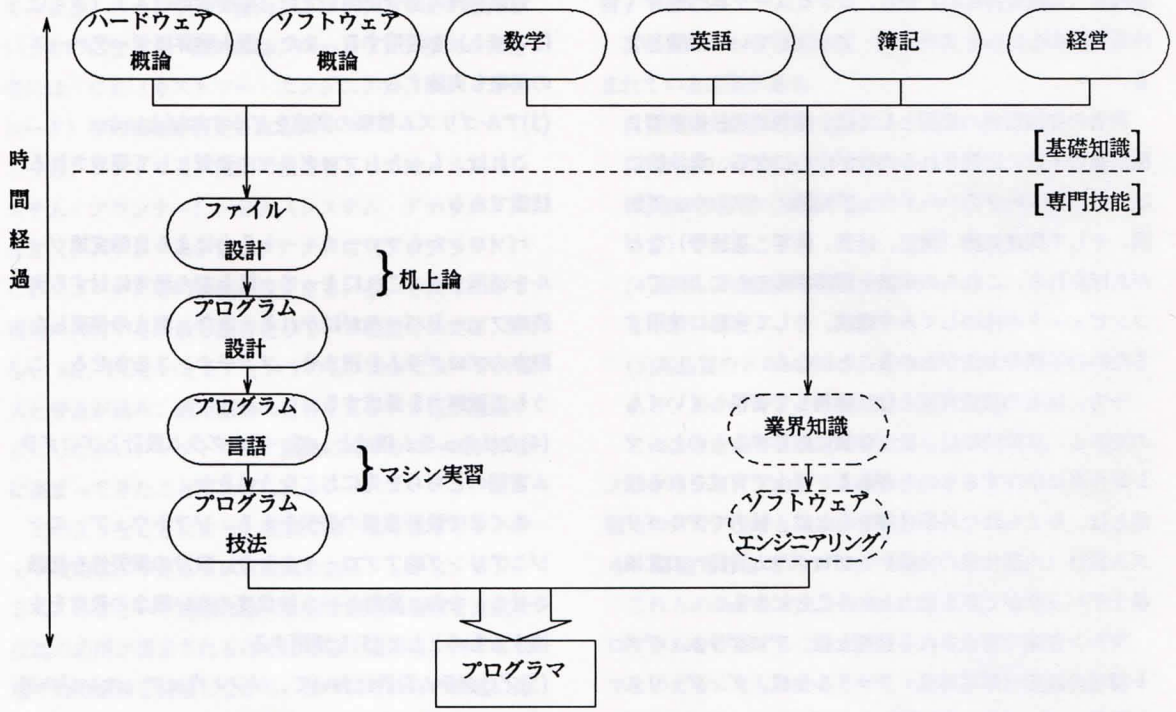
開発工数（設計期間，コーディング作業期間，デバック期間の総合計），デバック工程におけるマシン資源の使用状態（実行回数，ＣＰＵ使用量，入出力使用量），完成したプログラムの潜在的バグ発生状況（バグ発生数，修正工数），完成したプログラムの品質状態（外部仕様条件との適合度合，ソース・プログラムの理解容易性，プログラムの稼働効率），ドキュメントの品質状態（社内標準との適合度合，ソース・プログラムとの整合性，内容の信頼性度合）

６．今後の課題

以上，新入社員教育として，プログラマの育成を中心に検討してきた．明確な教育目標にもとずく体系だてられたカリキュラム構成は，学ぶ者にとっても，コンピュータについての勉学に対する方向づけができるとともに，効果的な教育経験を積み重ねていくことができる．そして，カリキュラムの中に，視聴覚教育を積極的にとりいれ，教育工学的アプローチをはかっていくことも，これからの教育には必要とされる．

今までの形骸化された新人教育（先輩１人に全員の新人をつけ，講義形式ですすめる形態）に対して，教育を工学的システムとしてとらえ直さなければならない．より多くの教材や適確な教授方法まで含めた新人の技術的な側面での教育方法論をノウハウとして蓄積していかなければならない．

会社にとって人材は人財であるはずである．有能なプログラマを育成していくことは，人財の蓄積にも結びつくことになる．有能なプログラマは，ＯＪＴによってより有能なＳＥへと変貌していくはずである．そのためにも，有能なプログラマをどのように教育していくか，いろいろな事例を含め研究していく必要がある．



CALL FOR PAPERS

9TH INTERNATIONAL CONFERENCE ON SOFTWARE ENGINEERING

FORMALIZING AND AUTOMATING THE SOFTWARE PROCESS

MONTEREY, CALIFORNIA, USA 30 March-2 April 1987

CHAIRMAN
William E. Riddle
software design & analysis, inc.
PO Box 3521
Boulder, CO 80303 USA

PROGRAM CHAIRMEN
Robert Balzer
Information Sciences Institute
4676 Admiralty Way
Marina del Rey, CA 90292 USA

Kouichi Kishida
Software Research Associates, Inc.
1-1-1, Hirakawa-cho, Chiyoda-ku
Tokyo 102 JAPAN

TOOLS FAIR CHAIRMEN
Larry E. Druffel
Rational
1501 Salado Drive
Mountain View, CA 94043 USA

Jack C. Wileden
Computer and Information Sciences
University of Massachusetts
Amherst, MASS 01003 USA

TUTORIALS CHAIRMAN
Richard Fairley
Wang Institute
Tyng Road
Tyngsboro, MASS 01879 USA

LOCAL ARRANGEMENTS CHAIRMAN
William M. Murray
General Dynamics Corporation
12101 Woodcrest Executive Drive
St. Louis, MO 63141 USA

Theme: Over the last decade, two important myths have gradually been discarded: that the waterfall model describes the software lifecycle and that code is the only product of that cycle. Significant systems arise by a evolutionary process rather than by implementing a carefully constructed specification in a single sequence of development phases. Code defines execution but provides an inadequate basis for understanding, and hence, maintaining and/or evolving, the implemented system.

These insights have shifted attention toward paradigms which recognize software development as an iterative design (development) process and techniques for recording the process itself to document and understand the resulting implementation. Formalizing the intellectual activity provides the basis for a new generation of development tools which automate and/or effectively support portions of the design process.

Objectives: The main objective of the conference will be presentation and discussion of progress in formalizing and automating the software process through provision of better models, methods, and tools, and identification of critical issues to be addressed. A secondary objective is to broaden participation beyond traditional Software Engineering to include the Artificial Intelligence and Database technologies needed to model, store, manipulate, access, reason about, and automate portions of the software process. The final objective is to provide wider access to the best work in this expanded field by republishing outstanding papers, summarizing important specialized workshops, and presenting summaries of recent advances in particular areas.

Original Papers: Authors are encouraged to submit papers that directly address the formalization and automation of the software process. Theoretical papers should indicate applicability. Papers about new developments should separate achievement from plans and evaluate usage. Papers reporting practical experience should include sound empirical evidence. Authors should be advised that conference time allocated to the newly incorporated workshop summaries, "Recent Advances In", and republished outstanding papers, will substantially reduce the number of accepted papers.

Tools Fair: A Software Tools Fair will be held in parallel with the conference to provide conference attendees with information about current software tools. Both experimental and commercial software will be demonstrated. In addition, the conference will include a special, separate track featuring presentation and demonstration of tools selected by the tools fair committee. Those interested in exhibiting in the tools fair, and especially authors interested in presenting a paper describing practice and experience with a particular tool in conjunction with a demonstration, should contact one of the Tools Fair chairmen.

Submission of Papers: Four copies (in English) should be submitted by 1 September 1986 to either Program Chair. Papers should be no longer than 6000 words. Full-page figures should be counted as 300 words. The paper should include a short abstract and a list of keywords indicating subject classification. Notification of acceptance will be sent by 1 December 1986. Camera-ready copy of the final version will be due 1 February 1987.

Further Information: For further information and/or a copy of the advance program when available, write to 9ICSE, c/o IEEE Computer Society, 1730 Massachusetts Ave., N.W., Washington, D.C. 20036 USA.

IMPORTANT DATES

Submission Deadline:	**September 1, 1986**
Acceptance Notification:	**December 1, 1986**
Final Versions Due:	**February 1, 1987**

SPONSORED BY:

ACM SIGSOFT

IEEE COMPUTER SOCIETY

IEEE THE INSTITUTE OF ELECTRICAL AND ELECTRONICS ENGINEERS, INC.

コスト管理の諸問題について

芝原　雄二
沖ソフトウェア

１．はじめに

５月９日（金）に行なわれた管理分科会（ＳＩＧＭＡＮ）の第２回研究会の報告をします．

当日の話題は，コスト管理ということで，ＪＳＤ（協同システム開発）で実施している調査研究を紹介し，そこで収集されたコスト・データを示しました．そのあとで，コスト管理ということで，参加者がどう考え，どう悩んでいるか等について意見を述べてもらいました．

本稿では，コスト管理の諸問題をどのように考えていくか，ということに論点を絞って，出席いただいた方々の意見をとりまぜながら，まとめました．

コスト管理の問題は，ここでとりあげたような問題だけではありません．会員のかたのご意見，経験を管理分科会にお寄せいただけると幸いです．

２．提示したコスト・データの概要

提示したコスト・データの詳細については，参考文献１を参照していただきたい（ＪＳＤの形見氏－０３－５０３－４９８１に問い合わせすることで，入手が可能です）．

３．質疑・応答

Q1. 受注・契約形態で見積り方法も変わるのでは？

受注・契約形態は各社各様であり，そうしたことからも，見積り方法も変わってくる．ここでは，受注形態としてどのような形態があり，契約形態にはどのような形態があるのか，ＪＳＤの報告書（参考文献１）からピックアップしてみる．

(1)受注形態

受注形態は，主に以下のようになっている．

・詳細設計からコーディング＆単体テストまで
・基本設計からシステムテストまで
・要求仕様から受入れテスト＆試運転まで

しかし，受注形態の比率は各社各様であり，発注者から作業内容が，明確に示されないことも少なくない．

(2)契約形態

契約形態としては，次のような形態が主に利用されている．

・定額契約方式
定額による契約で変動を認めない方式．
・精算方式
公共企業の研究機関などで使用されており，概算契約で作業を開始し，事後に超過（不足）額を一定の比率で当事者間で分配する方式．
・概算契約方式
作業量，作業範囲等が確定しにくい場合，まず上限を定めた概算契約を行い，作業開始後，確定してから詳細見積りをし，全額調整する．

この他にも，政策面から様々な契約形態がある（参考文献１参照）．

参加者の方で，「フェイズごとの契約を行っており，そのフェイズごとにコストを見積もることにしている．コストの折り合いがつかない場合は，契約中止となる」という実例としての話がありました．

また，エンド・ユーザである参加者からは，「契約形態として精算方式を望んでいるが，最近，この方式が開発会社の方からなくなりつつある．私どもとしては大変残念である．なぜならば，われわれは，ソフト開発には，こういうツールも作成する必要があるのだ，というようなノウハウの提供も望んでおり，そのノウハウにｐａｙする気持ちがある」という話もありました．

Q2. 見積り時，生産性を左右するものとして，どのような要因を考えるべきか？

これだけ，という要因は現在のところみあたらないようである．しかし，様々な要因が考えられている．コスト見積りのため，各社各様にデータ収集フォーマットを作成している．そうしたフォーマットから，どのような要因が必要か読みとることができる．ここでは，ＴＲＷ社ＢｏｅｈｍのＣＯＣＯＭＯであげられている要因を示すことにする（各社のデータ収集フォーマットについては，参考文献１を見てもらいたい．５社の事例が示してある）．

・プロジェクトの形態
・ソフトウェアに要求される信頼性の度合
・プログラム・サイズに対するデータベースの相対的規模

・ソフトウェアの複雑さ
・完成時のシステムの実行時間の制約
・完成時のシステムのメモリの制約
・対象システムの開発に使用するハード・ソフトの変更頻度
・対象システムの開発時におけるターンアラウンド時間
・分析チームの能力
・対象アプリケーションのシステム化経験
・プログラマ・チームの能力
・開発環境での作業経験
・プログラム言語の経験
・近代的プログラミング手法の使用と経験
・ソフトウェア開発に使われるツールのレベル
・プロジェクトへ要求された開発期間

ここに，ＣＯＣＯＭＯの要因を上げたが，「あまりにも要因が多すぎて，実際に，データを現場で収集するとき，現場の人間が，答えてくれない懸念がある」と言った意見も聞かれた．

この問題に関して，参加者の方々がとっている実例をあげてみる．

・コストの見積り時，マシン，言語，再利用率，システム経験を考えている．
・言語経験は，開発体制が設計とそれ以降の作業に分業化されている社内データを見ると，生産性に直接関係はみられない．
・客先での仕事が多いので，ユーザ環境，客先体制と提供仕様，個人のSKILL を考えている．

このように各社各様である一番の要因は，人とソフトウェア開発環境の違いによるものであろう．

Q3. コスト管理でのコスト・モデルの使いかたは？

ソフトウェア開発プロジェクトの巨大化，人件費の高騰の対応，生産における効率化，ソフトウェア生産の実態の把握，見積りの合理化といった，様々な要求が管理の面でわれわれに求められている．

これらコスト管理に関連する問題への１つの解決アプローチとして，生産性の尺度の設定，計測およびデータ収集の方法，分析評価方法を示しているコスト・モデルがあげられる．

コスト・モデルとは，推定された規模にシステムの持つ難易度，開発環境，ヒューマン・ファクター等の，生産性を左右する諸要因を掛け合せて，工数やコストを算出することである．ただし，この生産性を左右する要因も，ソフトウェア開発環境等の変化によって複雑に変化している．このため，これまでに多くのコスト・モデルが提案されている．

コスト・モデルは，あくまでもコスト見積り時の意志決定の１つの情報とすべきである．ソフトウェア開発環境は変化するものであるから，それらの開発情報を収集しながら，あるコスト・モデルを長期的にみつめていき，今後のコスト管理に利用するべき体制も必要である．

Q4. コストを見積る際の基礎となる規模は，どうして測るのか？

参加者の中で，次のような経験を話してくれた．「われわれが外注に仕事を委託した時，われわれが見積ったステップ数より，外注の見積りの方が，はるかによくあてはまった．そこで理由を聞いてみると，ＫＫＤ（経験，勘，度胸）だという」

これには参加者一同，うなずいてしまった．たしかに，ＫＫＤに基づいているのが実状であろう．

つまり，要求仕様書等を基に，機能の分析を行ない，さらに全体の機能を分割し，規模の推定可能なレベルまでモジュール分けし，モジュールの機能，言語などを参考にして，見積り者のプログラミングスタイル，開発環境での標準的なプログラミング規約等を考え合わせながら，過去の経験や勘を基に算出しているのが実状であろう．

Q5. ＫＫＤから脱却するためには？

Ｑ４のような討議のあと，こうした議論に発展することは，ＳＩＧＭＡＮが，たんなる世間話の場ではないことの証だと思うが，ＫＫＤに頼っていてはいけないとは，誰しもそう考えていよう．

これに対しては，類似システムの実績との比較によるコストの算出，つまり，完了プロジェクトの実績を技術分野，業績などの作業内容別に分類して蓄積しておき，このデータを参考にして算出する．このデータの蓄積こそが，ＫＫＤから脱却していくための１つの道ではなかろうか．社内での蓄積にとどまらず，社内データの社外への公開により，この業界のデータが蓄積され，データに語らせる時代がくる事を望む．

参考文献１：
「ソフトウェア・コスト・モデルの定量的評価」情報処理工学に関する調査研究　（社）情報サービス産業協会
昭和５８，５９，６０年度

環境構築のための自己啓発

道正　一郎
協同システム開発

１．はじめに

近年，ソフトウェア業界で「環境」という言葉を，頻繁に耳にするようになった．「環境」は，「ソフトウェア開発者が作業を通して利用する有形・無形の資源すべてである」とか，「ある方法論を基に，ソフトウェア開発の工程全体を支援するための自動化ツール群を有機的に統合化したもの」と，定義されている．

しかし，いぜんとしてこの言葉は，私自身にとって，漠然としたものであった．そうしたおり，ＳＥＡ　Ｆｏｒｕｍ　Ａｐｒｉｌ　’８６に参加する機会があり，「環境」という言葉を，具体的に，かつ実践的にどうとらえるかを，より身近なものとすることができた．

２．環境構築への出発点

このフォーラムを聞いて，次のような実験例を思いだした．

「空腹のカマスを水槽に入れ，中央に透明なガラス板を入れる．片方にはカマスの餌として小魚を入れると，カマスはその小魚を見るや，”食べよう”と突進する．しかしカマスはガラス板にぶつかって，もとの位置の方にひきかえす．また食べようと突進して，ぶつかる．これをくりかすうち，カマスはバッタリともう小魚の方へ行かなくなる．そこで，今度は水槽のガラス板をとりのける．すると小魚は，カマスの近くを泳ぎまわる．もうガラス板がないので，本来なら，小魚をカマスが食べることができるわけだが，カマスは食べようとは突進しない．そして翌朝，カマスは飢えのため水の上に死んで浮いていた」．

カマスは，自分の何回かの経験で，小魚を食べようとすると，痛い目にあったことから食べようとしなくなった．しかも，環境が変わったことにも気がつかず，自分の経験から学んだことに固執したといえよう．

このことは，まさに，自分自身についてもいえるように思えた．日常の業務に追われ，経営者または管理者から与えられた環境に対して，なんら疑問を抱かず，その枠内でいかに効率よく業務を遂行するかのみを考えてきた．つまり，自ら，現状に固執し，閉鎖状態をもたらし，問題の存在に気づかなくなってしまっていたようである．

わが国のソフトウェア技術者は，一説では約４０万人と聞く．他方ＳＥＡの会員は，残念ながら，まだその０．１％に満たない．

そこで単純に考えると，残りの９９．９％のうち，少なからず私と同じような状況にある人が存在するのではないかと思われる．そうであれば，技術者に課せられた緊急の課題である環境構築のためには，個々人の自己啓発が出発点となるのではないだろうか．実際，長岡のワークショップでも，「環境は自分自身のためのものであり，環境は与えられるものではなく自分で構築していくもの」と意見が一致していたようである．

３．ＳＥＡからソフトウェア業界全体へ

基調講演からは，社会的・経済的・技術的な側面から，ラディカルな変化が予想される，ということを再認識し，パネル・ディスカッションからは，実際の開発現場からの，生々しい多くの問題点によって，パネラー諸氏の関心事がどのあたりにあるかを思い図ることができた．

また，環境構築に向けてのひとつの事例として，ソフトウェア開発環境に定評のある，幕末の黒船にたとえられたＵＮＩＸの使用経験も，示唆にとんだものであった（反面，大艦巨砲のターゲット環境で長い間仕事をしてきていると，なかなかＵＮＩＸの導入には踏み切れない，という意見もある）．

しかし，一般的にはこれまでの伝統を打破し，蓄積された資産を捨て，改革を行うには，リスクを含んでいるため難しさがあり，費用もかかることはゆがめない．

だからといって，こうしたワークショップやフォーラムでの議論が架空の議論で終わってはならない．ＳＥＡでのこうした議論を，ソフトウェア業界全体に浸透させることが望まれる．それは，参加したひとりひとりの課題であろう．

４．おわりに

このフォーラムから，私自身に，日頃抱えているさまざまな問題意識の提起がなされた．このことはいいかえると，現状では，「ワカル」という作業がなされた．この作業から今後は，どのように「カワル」という作業に移行すべきかを，考えていく必要性が感じられる．

書評

人工知能・知識工学に関する文献紹介

人工知能，あるいは知識工学の応用を目指している者にとって，参考になると思われる文献を紹介する．

最新・最近の話題を載せているものを紹介したかったのだが，実際には，ほとんどの文献が，内容的には，1983年までの技術的水準と動向が主体となっているため，ごく最近の話題については，残念ながら適当なものがない．読者のかたで，そうしたものを知ってらしたら，ＳＥＡＭＡＩＬの編集部に「書評」として投稿していただけたらと思う．

こうした裏には，現在のいわゆるＡＩ関連のビジネスが，実用的なシステムとして応用可能なのは，1983年までにだいたい確立された，「浅い知識」についての研究成果であり，したがって，それ以後のものについては，まだ研究段階のものでしかないと推測できる．最近盛んにいわれている，「深い知識」の取り扱い，あるいは「常識」の取り扱いなどは，論文としては出ていても，１冊の本にできるほどの内容をもっているとはいい難いようである．これらのいわゆる「難しい問題」についての参考書・教科書がでるには，今少し時間が必要だろう（３～４年はかかかりそうにみえるのだが）．

したがって，ここに紹介するものも背景となっている理論・技術的側面は，８３年までに擁立されたものである．ただし，ビジネスとしての動向には，比較的新しい話題も若干は含まれている．

(1)「BUILDING EXPERT SYSTEMS」, Frederick Hayes-Roth, Donald A. Waterman, and Douglas B. Lenet, 1983, Addison-Wesley

（「エキスパート・システム」，ＡＩＵＥＯ訳，産業図書，１９８５，３９００円）

この本については，いろいろな書評欄等で採り挙げられているので，いまさらといった感もなきにしもあらずなのだが，とにかく，ＡＩに関した仕事をしている者には，読んでおかなくてはならない（？）一冊といえる，ということで，あえて取り上げてみた．

巻頭にもあるが，ＥＳ（エキスパート・システム）について，これまでチャンとした一冊の本としてまとまったものがなかったので，ワークショップを催して，本書を編集したとある．実際，著者の顔ぶれを見ただけでも，そうそうたるメンバーが名を連ねている．したがって，良くも悪くも，この１冊でＥＳについて，ほとんどのことがわかるように書かれている．

５部１０章と，１つの付録から成る本書は，第１部がＥＳの基本概念，第２部がＥＳのアーキテクチャ，第３部がＥＳの評価，第４部がケース・スタディとして，「流出問題」を各ツールでシステム化する場合を取り上げて，各ツールを解説している．第５部がその「流出問題」自身の解説，そして付録に「流出問題」についての各ツールのインタラクションの模様がでている．

どの部のどの章からでも読みはじめられるようになっている反面，章間での内容の重複がやや目立つが，質的には，かなり詳しく解説されている．とくに，第６章，第９章のツール解説では，「流出問題」を各ＥＳツールをもちいて，知識ベース化する際の，シンタックスと推論エンジンの振舞いが示されており，実際にＥＳを構築しようとする者には，格好の「外部仕様」の模範がここに読み取れる．

ＥＳ構築を目指す人，研究者向きといえよう．

(2)「EXPERT SYSTEM: ARTIFITIAL INTELLIGENCE IN BUSINESS」, Paul Harmon and David King, 1985, 283p, WILEY PRESS

これは副題をみてわかるように，本書は，ビジネスとして，ＡＩを応用する立場を主眼として書かれているが，内容は万遍なく一通りのＡＩに関する話題を説明している．本としてのボリュウムも手ごろな割には，よくまとまっているといえよう．ＥＳの初期のシステムから，最近のものまでが手際良く整理され，その特徴，購入価格，適用分野等が，一目でわかるようになっているのは便利である．いわばハンドブック的な体裁になっていて，ＡＩについての初心者にも，充分理解できる内容となっている．

第一章では，まずＭＹＣＩＮの概略が示される．続いて，人の問題解決におけるアプローチ（例題にはナンと，ドイル卿の「赤毛同盟」の冒頭が引用されている），知識表現の方法，推論の手法，そしてＡＩの基本的知識が，各々に解説される．そして再度，ＭＹＣＩＮを採りあげ，

詳細を述べている．

第二章は，ＡＩの言語，商用ツール・システムが解説される．ＡＲＴ，ＫＥＥ，ＯＰＳ５，ＬＯＯＰＳ等が採りあげられている．

第三章は，ＥＳの構築について述べているのだが，残念ながら，ここで述べている内容程度では，とても実際にＥＳを構築するにはおぼつかない．参考といった程度である．

第４章がこの文献の本領といってよく，ＥＳの市場について，多角的な検討をすることによって，今後の方向について述べている．

全体としてみてみて，コンパクトでよくまとまっており，ＡＩ・ＥＳの入門，あるいは概要を知りたいといった人には，オススメの品である．

(3)「ARTIFICIAL INTELLIGENCE」, Elaine Rich, 1983, McGraw-Hill
（「人工知能１，２」　広田薫／宮村勲　訳，マグロウヒル，１９８４）

ＡＩについて，理論的側面から書かれているのが，本書の特徴である．ＡＩ，あるいは知識工学の背景となっている様々な理論・手法を，どのような問題に対処しようとしているのか，解決へのアプローチのしかた，数学的または科学的な裏付け，現状での成果・限度，今後の課題，といった点から詳述してある．

もともと，本書は，大学の教科書として，専門課程よりは一般課程の学生向けにＥ．Ｒｉｃｈ女史が書き下ろしたものであり，人工知能，知識工学の基礎を，わかりやすくしめしたものである．したがって，それほど深い数学的素養を必要としなくても，内容が充分理解できるように，配慮された記述がされている．

前半は主として，「検索(SEARCH)」について，さまざまな手法と理論が示される．いわゆる人工知能・知識工学で，現在対処が可能な「浅い知識」の扱いに関する問題は，ほとんど「検索」処理の問題として考えられていることがわかる．

つまり，知識が記号として格納されている状態では，知識処理は，その格納されている知識をいかにに速く捜しだすか，という問題と等価であるとみてよく，知識の獲得（記号化），あるいは既存の知識から全く新しい知識の合成，そして「常識」のようなあい昧模糊としたものの扱い，といった記号化以前の「難しい問題」は，現状のＡＩでは対処できないことも理解される．

それゆえに，知識工学のこれからの問題も，「より高度な検索手法の開発」と「知識の記号化」の２つが主要な分野となるであろうことは論を待たない．

本書でも後半では，このへんの問題をを取り上げている．本著の著者であるＥ．Ｒｉｃｈ女史は，すでにＭＣＣに移っており，ＭＣＣの「常識」を扱うことを目指した，ＣＹＣプロジェクトが開始されていること等からも，いよいよ「深い知識」の取り扱いに向かって，ＡＩ・ＫＥの研究は，第２ラウンドに入ったようである．

(4)「COMMON LISP THE LANGUAGE」, Guy L. Steele Jr., et al., 1984, Digital Press

いわずと知れた，ＧＬＳ（グリスと読む）の「コモン・リスプ」言語仕様書である．手元に実際に動くコモン・リスプ処理系がないと，ちょっと面白くないかもしれないが，リスプを使おうとする者には，必読の書といってよいだろう．

また，他の言語処理系開発に携わる者にとっても，本書にもられているコモン・リスプの機能仕様，たとえば，パッケージ，レキシカル・スコープ，ＩＭＰＯＲＴ／ＥＸＰＯＲＴ，等などの概念を理解することは，おおいに参考になると思う．本書を勧める由縁である．

コモン・リスプには，まだまだいろいろな問題はあるが，ネットワーク上のＣＣＳでの話合いから，これだけの物を作りだしてしまう米国の研究者の環境，能力が，なんとも羨ましく思えるのは私だけであろうか．

（藤野晃延）

お知らせ

ＳＥＡＭＡＩＬの１号，２号，３号，４号，５－６号の在庫が若干あります．ご希望のかたは，１冊につき７００円（送料込み）の切手を同封のうえ，事務局までお申しこみください．

幹事会報告

昭和６１年度第１回：６月１７日（火）
会　場：機械振興会館
時　間：１８：３０－２０：００
議題：

１．６１年度幹事会役委員

５月１７日の年次総会で承認された，６１年度の幹事会役員が紹介された．この，役員一覧は，本ＳＥＡＭＡＩＬの裏表紙に載せられている．

２．会員状況

会員数は，６月１７日時点においては，正会員３７０名，賛助会員１０社であった．

１２月からの正会員の増加は，月平均３０名になっているが，最近やや増加の伸びがにぶってきている．

３．会員更新状況

２００名いるボランティア会員の更新状況は，６月１７日では，約５０パーセントであった．

更新率が低いのは，更新の意志がないためより，更新の手続き（銀行へ行って振り込む）が，面倒でしていないケースがおおいようにおもえる．そのため，６月末に，個人宛に，更新の案内をだすことにする．また，メイル・ラベルに更新時期を追加することは，５－６合併号のＳＥＡＭＡＩＬからおこなっている．

４．設立記念フォーラム

５月１９日（月）の午後，東京農林年金会館パストラルでおこなわれた，ＳＥＡ設立記念フォーラム「ソフトウェア技術者に期待する」の収支概算が杉田セミナー委員長から報告され，若干の黒字になりそうなので，事務局にプールされることになった．

このフォーラムの基調講演とパネル討論の記録は，今号のＳＥＡＭＡＩＬに特集されている．

５．イベント

今後のイベントとして，「実践的再利用技術を探る」を基調テーマとして，７月１１日に，ＳＥＡ　Ｆｏｒｕｍ　Ｊｕｌｙ’８６が予定されている．また，７月の毎火曜日の夜，ＬＩＳＰの講習会が行われる．このふたつのイベントは，再利用分科会，ＡＩ分科会が協力して行われる．

９月に行われる「秋のセミナー・ウィーク」は，準備が進められており，新しい方式が，講師陣にも好評であることが報告された．

ＪＩＳＡと共催で行われる，盛岡「若手の会ワークショップ」も準備がすすめられていることが報告された．

情報化月間（１０月）にあわせて，１０月１－２日に，国際フォーラムが企画されている．このフォーラムのメイン・スピーカーとして，カーネギーメロン大学のハーバーマン教授を予定しており，現在交渉中である．

１０月２２－２４日に，日本ユニックス・ユーザ会の主催，ＳＥＡ後援，ｂｉｔ協賛で，「ＵＮＩＸワークショップ８６　ｉｎ　横浜」の計画がすすんでいる．詳細は，これからである．

これらイベントは，ＳＥＡとしての活動の活性化と，事務局の財政基盤確保のためにも，是非とも成功させたい．

６．分科会・支部報告

ＡＩ分科会が，ようやく発足することになった．また，法的保護部会も，７月１日に第１回会合が開催される予定であることも報告された．

７．会員の獲得

会員の増加が，６１年度計画からすると，伸びがにぶいようである．６１年度では，会員数１，０００人を予定している．そのためには，今後月平均５０人の新規加入が必要となる．経済的基盤，および活動の活性化のためにも，正会員，賛助会員を増やすための活動を行う必要がある．賛助会員用パンフレットの作成，会員獲得期間の設定など，具体的に動くことになった．これは，水谷幹事，針谷幹事を中心におこなわれる．

８．次回会合

日時：７月２４日（木）18:30 - 21:00
会場：機械振興会館　地下３階Ｂ３－１

（野辺良一）

ＡＩ分科会(SIGAI)

１．第１回月例会

第１回月例会を，６月２４日（火）に，新潟からの２名（遠いところごくろうさまでした，これからもよろしく）を含めて１９名の参加者によって開催しました．

まずは参加者の自己紹介を，ＡＩというえたいのしれないものとの関わりあい含めて行いました．

そのあとに，白井さんから，今後のＳＩＧＡＩの活動方向を探る意味から，「ＡＩ技術のテクノロージ・マップ」ということで，知識工学の動向を簡単に話てもらいました．この発表の内容は，近々，ＳＥＡＭＡＩＬ誌上で紹介したいとおもいます．

２．今後の運営について

白井さんの話のあとに，今後の運営について，参加者全員で討論を行い，以下のように取り決めた．

(1)開催日・場所

開催は月例会として，毎月第２木曜日の１９：００からとし，会場は原則として「機械振興会館」とする．

(2)世話人・会計

世話人として，以下の方々があたる．世話人は，月例会の運営にあたり，講演会，事例発表，勉強会とうの月毎の企画にあたる．

藤野晃延（７月），山本一成（８月），高田佳彦（９月），坂下秀（１０月），広川昭八（１１月），白井豊（１２月），安倍昭敬（１月），横山憲一（２月）

また，会計は梅林信之があたる．

第２回月例会

日時：７月１０日（木）１９：００～２１：００
場所：機械振興会館　６階６１号室
話題：事例発表「ロジック・プログラミングについて」
報告者：西山　聡（ＭＲＩ）
会費：1,000円程度（会場費として）

第３回月例会

日時：８月７日（木）１９：００～２１：００
場所：機械振興会館　６階６１号室
話題：未定

(3)会費

会費はとくに徴集しないが，月例会の会場費として，参加者にその都度５００－１０００円を集める．

(4)連絡先

連絡先および問い合わせは，協同システム開発(03-503-4981)の野辺までどうぞ．

（野辺良一）

管理分科会(SIGMAN)

第３回研究会は，講師の都合により，発表を次回の４回にスライドし，３名による「管理問題夜話」になりました．

現在管理分科会に登録しているかたは，マネージャの人が多く，なにかと忙しいようです．プロジェクト管理に関心をもっている，若い方々の参加を待っています．

スライドした今後の研究会の予定は，以下のとうりです．

また，管理分科会への連絡，問い合わせは，下記にどうぞ．

連絡先：協同システム開発(03-503-4981)
芝原雄二

第４回会合

日時：７月１１日（金）１９：００～２１：００
場所：ＳＥＡ高円寺事務所
話題：デザインレビューに関するデータの紹介
報告者：前島　仁
会費：1,000円程度

第５回会合

日時：８月８日（金）１９：００～２１：００
場所：ＳＥＡ高円寺事務所
話題：プロジェクト管理上の問題について
報告者：土崎直樹

教育分科会（SIGEDU）

1．第3回分科会報告

6月24日（火）に，8名の参加を得て第3回会合を開催しました．テーマは「新技術の導入—実践的技術移転の試み」として，シーイーシーの大浦洋一さんから話していただきました．マイコン・ソフトウェア開発における新しい技術移転をシグマ・プロジェクトの成果を取り入れて，ワークステーションを導入し開発環境を整備しようという提案を，参加者で検討しました．経営者への具体的な提案書の書き方や，データ収集の方法，説得の仕方，メーカが実施しているモニター制度への直接参加する有利な方法など活発に議論が出て有益な討論になったようです．その他，派遣社員の意識の話題，定着率の低さ，要員に対しての期待や不満など広範囲に議論が飛び，楽しい討論集会になりました．

今回の発表と討論を参考にして，大浦さんがまとめたうえ，ＳＥＡＭＡＩＬに掲載する予定です．

2．次回（第4回）分科会の予定

・日時：7月30日（火）18：30−20：00
・場所：機械振興会館：6S−1（6F）
・参加費：1000円（会場費として）
・発表者：中園　順三（富士通ビー・エス・シー）
　テーマ：中堅技術者教育
・内容：ソフトウェア中堅技術者を，どのように養成し，教育するか．参加者とともに効果的な教育の方策を探る．
・参加資格：参加自由
・連絡先：杉田　義明（03—234—2611）
　　中園　順三（03−501−4181）
・その他：参加される方は特に連絡は不用です．直接会場にお越し下さい．

3．教育フォーラムの予告

ソフトウェア技術者の新入社員教育を考える半日のフォーラムを計画しています．主として今年度教育を担当されたスタッフの方々に集まっていただき，カリキュラムの反省や今年の新入社員の傾向等の情報交換をし，参加される方々の今後の計画づくりに生かそうと狙うものです．このフォーラムに関して何か発表したい方，あるいは意見のある方は事務局または幹事まで連絡を取って下さるようお待ちしております．

（杉田義明）

再利用分科会(SIGREUSE)

1．活動報告

(1)第3回編集会議−5月7日（9名）
日経コンピュータ副編集長の上村氏を囲んでの討論会．

(2)第3回研究会−5月14日（8名）
世間に存在する再利用ツールのサーベイについて

(3)第3回討論会−5月28日（8名）
Ｆｒｅｅｍａｎの論文に従った，ソフトウェア情報の構造に関する検討

(4)第4回研究会研究会−6月11日（10名）
各個人の研究テーマ案を以下のように決定をした．
・再利用と4GL，VHLL
・データ抽象化言語と再利用
・再利用の実際システム
・ソフトウェア再利用とＡＩ
・ソフトウェア部品の現状
・再利用とメトリックス
・フォーマル・スペックと再利用
・再利用ツールの調査
・設計情報の再利用

(5)第4回討論会6月25日（12名）
フォーマルスペックとソフトウェア再利用．

2．今後の予定

(1)第5回討論会（第4水）
日時：7月23日19:00-21:00
場所：機械振興会館　B3−2
テーマ：4GLとソフトウェア再利用

(2)第6回研究会（第2水）
日時：8月13日　19:00-21:00
場所：機械振興会館　B3−9
テーマ：各自のテーマに基づく意見交換

(3)第6回討論会（第4水）
日時：8月27日　19:00-21:00
場所：機械振興会館　B3−9
テーマ：部品化に基づくソフトウェア再利用システムを構築し，全社レベルで実際に使用しているという，実例の発表．

（参加はすべて自由）

連絡先：青島（234-2611），阿部（444-3211），大西（503-4981），村井（433-8171）

（村井進）

法的保護分科会(SIGSLP)

1．第1回分科会

ＪＩＳＡの権利保護部会時代のメンバー１７名に，新規メンバー２名を加えた１９名でＳＩＧＳＬＰが発足しました．メンバー構成は，ソフト業界において法的保護のオーソリティの１人であるソフトウェア流通促進センター長の石原氏，ソフトウェアと法律の両方に明るい弁護士の佐野稔先生をはじめ，各社の経営者，管理者がほとんどという”プログラマの成れの果て集団”はたまた”おジン・パワー集団”的色彩になっています．

第１回目の分科会は７月１日（火）１８：３０から，「ＵＮＩＸのライセンスをめぐる諸問題」というテーマで，参加者全員が好き勝手ことをいう自由討議形式で行われました．テーマの討議にあたっては，ＡＴ&Ｔと某社のＵＮＩＸ「Ｓｏｕｒｃｅ」と「Ｓｕｂｌｉｃｅｎｓｅ Ａｇｒｅｅｍｅｎｔ」を参考にしました．

佐野先生から「バイナリ・ソフトウェアの使用（ｔｏ ｕｓｅ）とは，どういう意味なのか？実行するのであればｔｏ ｅｘｅｃｕｔｅだけでもよさそうなのに」という”ソフトウェア使用”という定義についての新しい問題提起があったり，はたまた，シグマ・センターにおけるＵＮＩＸセンター化はできるのかとか，「Ｓｏｕｒｃｅ Ａｇｒｅｅｍｅｔ」にはＬｉｃｅｎｓｅｅに有利になる盲点があるのが発見されたり，大変活発な，かつ有意義な討議が２時間続きました．

なお，この「Ｓｏｕｒｃｅ Ａｇｒｅｅｍｅｎｔ」の盲点についてお知りになりたいかたは，ＳＩＧＳＬＰ分科会にご参加下さい．

2．次回分科会

ＡＤＡＰＳＯ（アメリカの業界団体）がまとめたＰｒｏｐｏｓａｌ ｆｏｒ Ｓｏｆｔｗａｒｅ Ａｕｔｈｏｒｉｚａｔｉｏｎ Ｓｙｓｔｅｍ Ｓｔａｎｄａｒｄｓ（全８４ページ）を材料に，ソフトウェアのキイやプロトコルの標準化について，フリー・ディスカッションをする予定です．

日時：８月５日（火）18：30-20：30

場所：ソフトウェア流通促進センター会議室

会費：１，０００円（お茶代）

参加資格：ＳＥＡの会員で法的保護に興味のある人（新規に参加を希望されるかたは分科会世話人・能登までご連絡下さい）

ＴＥＬ03−239−5472（ＳＲＡ）

3．ついでに

・ＳＩＧＳＬＰのＳＬＰはＳｏｆｔｗａｒｅ Ｌｅｇａｌ Ｐｒｏｔｅｃｉｏｎの略です．

・今後ＳＩＧＳＬＰは，毎月第１火曜日に開催する予定です．なので，別称「一火会」です．

環境分科会(SIGENV)

1．第６回月例会

第６回の月例会を，６月１８日（水）に機械振興会館で行った．

テーマは，「ドキュメントの再利用支援環境」であった．この発表内容と，そのあとの討論をまとめて，ＳＥＡＭＡＩＬ誌上に掲載する予定です．

2．次回以降の月例研究会の予定

第７回月例会

日時：昭和６１年７月１６日（水）19:00−21:00

場所：機械振興会館　地下３階２号室

テーマ：バロース社のＬＩＮＣの使い心地

その他

世話人（連絡先）：引地信之（03-234-2611−ＳＲＡ）

第８回月例会

日時：昭和６１年８月２０日（水）19:00−21:00

場所：機械振興会館　地下３階２号室

テーマ：未定

関西だより

１．第４回研究会

去る５月３１日，恒例のＳＥＡ関西第４回研究会を開催しました．今回のテーマと講演者は次の通りです．

(1)シグマ・センターについて（久保宏志）

シグマ・シリーズの第３弾として，今回はその中核となるべきシグマ・センターについて，最新の話題を提供していただきました．話の内容については，特に関西風にアレンジしていただき，とても東京では聞けないような裏話を交え，楽しくお話いただきました．

(2)ＵＮＩＸの画面エディタについて（坂井誠）

ＵＮＩＸ環境における，もっとも有名な画面エディタであるｖｉについて，実際に利用した立場からその使いごこちを報告していただきました．最初の１年は非常に使いずらい，２年目はなるほど使えるな，３年目はもう手放せない，という感じでした．

(3)プログラム言語はなにがよいか

コーディネータ：盛田政敏

現在，利用されているプログラム言語について，参加者の方々を交えて討論をおこないました．

洋服のデザインの流行に対応させ，モード，ファション，スタイル，フォーマルとうに分類すれば，それぞれの言語は一体どの辺りに位置するのであろうか．また，市場占有率や成長率からみた，各プログラム言語の位置付けはどのような状況であろうかという，大胆な観点からの討議が行わました．会場からも活発な意見がでて，非常に楽しい討議となりました．

２．ＳＥＡよもやま話

今回からの新企画として，ＳＥＡの活動状況や，ソフトウェアに関する最近の話題について，気軽に話し合うことにしました．第１回目として次のような話題を取り上げました．

(1)ＳＥＡ設立総会，設立記念フォーラムの報告

報告者：盛田政敏

５月１７日に行われたＳＥＡの設立総会，５月１９日のＳＥＡ設立記念フォーラムの様子を報告しました．

(2)ＣＩＳＡ（情報監査人認定試験）の受験体験談

報告者：臼井義美

４月１９日に行われたＣＩＳＡ試験の受験失敗を話ました．皆さんに簡単な例題に取り組んでいただき，自信を深めて，来年のＣＩＳＡ試験にトライしていただくことにしました．

(3)長岡ワークショップ参加報告

報告者：久保宏志

雪の長岡で行われたＳＥＡのワークショップについて，そこで起こった出来事を，生々しく報告しました．

３．ＳＥＡ関西月例会

今回より，テーマを決めて，月例会を開催することになりました．６月は，ネットワークについてのソフトウェア談義を行いました．主な話題は，「ｊｕｎｅｔの接続に死ぬほど苦労した」こと，「ｊｕｎｅｔとはそもそも何であるか」などでしたが，「ＳＥＡ－ＮＥＴは一体どうなとるんじゃ」，「何で回線使用料が高いのか」，「何で回線を使わななければ通信できんのか」，「鯨は海のなかで１万キロも離れて通信できるらし」，「そんな騒々しい所でどうして寝のか」などと，だんだん訳のわからん論議に発展し，予定時間を大幅に超過してしまいました．

４．次回月例会

次回は，「ダメプログラムの見分け方」というテーマで行いますが，一体どうなることやら．興味ある方はどうぞご参加ください．

（臼井義美）

月例会および関西支部に興味のあるかたは，下記までお問い合わせ下さい．

臼井義美（日本電子計算06-448-6021），盛田政敏（神戸コンピュータサービス078-391-8291）

ＵＮＩＸワークショップ 86 in 横浜

討論参加者募集

日本ＵＮＩＸユーザ会（jus）では，今秋１０月２２日（水）－２４日（金）の３日間，「ＵＮＩＸ活用の現状と将来動向をさぐる」というタイトルのワークショップを，ホリデイ・イン横浜（神奈川県横浜市）で開催します．後援は，ソフトウェア技術者協会（SEA）．なお，bit 誌が協賛し，ワークショップ・レポートは来春同誌増刊号としてまとめられる予定です．

企画・運営スタッフは，以下の各氏（敬称略）：

実行委員長：石田晴久（東大）
プログラム委員長：岸田孝一（ＳＲＡ）　斎藤信男（慶応大）
プログラム委員：佐原伸（ＮＣＣ）　杉田義明（ＳＲＡ）　多田好克（電通大）　長谷部紀元（図書館情報大）
平塚芳隆（富士通）　深瀬弘恭（アスキー）　藤林信也（日電）　村井純（東工大）

このワークショップは，ＵＮＩＸに関するホットな話題をめぐっての連続パネル討論（各３時間）の形式で運営されます．いま予定されているパネル・セッションのテーマは，以下の通り：

Ａ．ソフトウェア開発環境

ＵＮＩＸをベースとするソフトウェア開発・保守・管理支援の現状（どんなツールを使ってどんな仕事をしているか？）と，これからの環境改善の動向について，パイオニア的ユーザの経験にもとづいて討論する．

Ｂ．ネットワークと分散環境

ＬＡＮおよびパーソナル・ワークステーションに関するハード・ソフト両面の技術動向や，それらを利用した分散型の開発環境／アプリケーション環境の構築や利用の具体的なケースについて討論する．

Ｃ．オフィスとＵＮＩＸ

ＵＮＩＸのＯＡ分野への応用の現状と今後の発展の可能性について，日本語化，文書処理，グラフィクス，電子メイルその他の日常的ユティリティなど関連する技術要素の動向も含めて討論する．

Ｄ．ハッカーたちのＢＯＦ

ＵＮＩＸの上で楽しくハッキングしながら仕事をこなしてゆくやり方のいろいろについて，またそうした気狂いたちの面倒をみる上でのノウハウについて，インフォーマルかつ過激に語りあう．

Ｅ．ＵＮＩＸの将来

ＡＴＴ版およびバークレイ版の今後は？　パソコンＵＮＩＸはどうなる？　シグマ・プロジェクトの行方は？　ＵＮＩＸに代わるＯＳはあらわれるか？　標準化の是非論，ソフトウェアの流通や保護をめぐる問題点など，ＵＮＩＸの将来について幅広く討論する．

ついては，これらの討論にパネリストまたはディスカッサントとして参加される方々を広く公募します．希望者は，対象セッション名およびそのテーマに関する御自身の主張／問題意識等をＡ４版１－２ページにまとめたポジション・ステートメント（できればワープロまたはコンピュータ出力で）を，７月末日までに２人のプログラム委員長のいずれか宛にお送りください．８月上旬に開かれるプログラム委員会で審査のうえ，結果は８月中に応募者全員に通知します．

ポジション・ステートメント送付先：

岸田孝一　〒102 千代田区平河町 1-1-1
（株）ソフトウェア・リサーチ・アソシエイツ
または
斎藤信男　〒223 横浜市港北区日吉 3-14-1
慶応義塾大学　理工学部　数理科学科

論 文 募 集

ソフトウェア・シンポジウム '87

1987 年 6 月 4 - 5 日　◇　農林年金会館（東京・虎ノ門）

共　催：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７回目を迎えます．産業界の現場で，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，内容も年々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，急激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，ヴァラエ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・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
- 現実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
- 中堅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
- ネットワークのセキュリティ
- 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

応募要領

応募論文の要旨を２千字－３千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判２－３ページ)を**10部**コピーして，

1986 年 11 月 30 日までに

２人のプログラム委員長（高田佳彦または林香）のいずれか宛にお送りください．

別途編成するプログラム委員会で応募論文の審査を行い，結果を 1987 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（予稿集にのせるためのカメラ・レディ原稿）を1987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，なに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
岡田正志
日本電気ソフトウェア
生産技術部
〒108 港区高輪 2 - 17 - 11
TEL 03 - 444 - 3211

プログラム委員長
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

ツール展示委員長
岸田孝一
ソフトウェア・リサーチ・アソシエイツ
専務取締役

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

ＳＥＡこれからの活動予定

－１９８６－

◆７月(July)

１０日（木）

ＡＩ分科会第２回月例会（機械振興会館）

１１日（金）

ＳＥＡ Ｆｏｒｕｍ Ｊｕｌｙ（青年会議所会館）

「実践的再利用技術を探る」

管理分科会第４回（ＪＳＤ）

１６日（水）

環境分科会第６回月例会（機械振興会館）

３０日（水）教育分科会第３回会合（機械振興会館）

◆８月(August)

５日（火）

法的保護分科会第２回会合

（ソフトウェア流通促進センター）

７日（木）

ＡＩ分科会第３回月例会（機械振興会館）

１４日（水）

環境分科会第７回月例会（機械振興会館）

３０日（土）教育分科会第４回会合（機械振興会館）

◆９月(September)

３日～６日

秋のセミナー・ウィーク（機械振興会館）

１０日～１３日

夏のプログラミング・ワークショップ（岩手県・盛岡）

１１日（木）

ＡＩ分科会第４回月例会（機械振興会館）

１８日（水）

環境分科会第８回月例会（機械振興会館）

◆１０月(October)

１日～２日

国際フォーラム（東京・東条会館）

９日（木）

ＡＩ分科会第５回月例会（機械振興会館）

１５日（水）

環境分科会第９回月例会（機械振興会館）

２２日～２４日ＵＮＩＸワークショップ８６ ｉｎ 横浜

（ホリデイ・イン 横浜）

ＳＥＡ会員状況

昭和６１年６月３０日現在の入会会員状況は以下の通りです．

正会員－－３７２名

賛助会員－－１０社

	勤務地域	居住地域
岩手 ＝	2	2
新潟 ＝	5	5
栃木 ＝	1	2
茨城 ＝	2	2
埼玉 ＝	3	31
千葉 ＝	8	31
東京 ＝	256	150
神奈川＝	20	70
長野 ＝	5	5
富山 ＝	1	1
静岡 ＝	3	3
岐阜 ＝	2	3
愛知 ＝	6	5
滋賀 ＝	0	1
京都 ＝	4	8
大阪 ＝	37	29
奈良 ＝	0	2
兵庫 ＝	11	13
長崎 ＝	1	1
鹿児島＝	1	1

会員の事務管理は，パソコンを使用しておこなっています．住所，会社部署等の変更があったときは，事務所へ早い時期にお知らせください．

ソフトウェア技術者協会入会申込書（正会員）

（フリガナ）
氏　名：＿＿＿＿＿＿＿＿＿＿＿＿＿＿＿＿

勤務先名：＿＿＿＿＿＿＿＿＿＿＿＿＿＿＿＿＿＿＿＿＿＿＿＿＿＿

勤務先住所：〒(　　)＿＿＿＿＿＿＿＿＿＿＿＿＿＿＿＿＿＿＿＿

勤務先TEL：＿＿＿＿＿＿＿＿＿＿＿＿

自宅住所：〒(　　)＿＿＿＿＿＿＿＿＿＿＿＿＿＿＿＿＿＿＿＿

自宅TEL：＿＿＿＿＿＿＿＿＿＿＿＿

連絡先（どちらかにチェックしてください）　　□勤務先 □自宅

年令 ＿＿ 才　　性別（男・女）　　血液型（A・O・B・AB）

会費：　入会金３千円　＋　年会費（入会より１ヶ年分*）７千円　＝　１万円

ソフトウェア技術者協会入会申込書（賛助会員）

会社名：＿＿＿＿＿＿＿＿＿＿＿＿
（フリガナ）
代表者：＿＿＿＿＿＿＿＿＿＿＿＿

住所：〒(　　) ＿＿＿＿＿＿＿＿＿＿＿＿＿＿＿＿＿＿＿＿

電話：＿＿＿＿＿＿＿＿＿＿＿＿

連絡担当者：
（フリガナ）
氏名：＿＿＿＿＿＿＿＿＿＿＿＿

所属：＿＿＿＿＿＿＿＿＿＿＿＿

賛助会費：＿＿＿＿口（１口　５万円）　　ただし入会より１ヶ年分*

＜申込書送付先＞：〒166　東京都杉並区高円寺南1-5-4 高円寺サンハイツ404
ソフトウェア技術者協会

＜会費振込先＞：三菱銀行本店公務部
普通預金口座 No. 0004830
口座名：ソフトウェア技術者協会
なるべく添付の振込用紙をお使い下さい。

*：　61年2月26日の幹事会での細則決定による

ソフトウェア技術者協会
〒166 東京都杉並区高円寺南1-5-4 高円寺サンハイツ404
TEL. 03-312-3256