U.D.C. 681.32.06:519.683.2

プログラムの木構造化図面“PAD”

Tree Structured Program Logic Diagram ``PAD''

二村良彦* *Yoshihiko Futamura*
川合敏雄** *Toshio Kawai*

ＰＡＤは，日立製作所の研究所で開発したプログラム論理図の記述法である。プログラムの論理は，本質的に木構造として記述できることに注目し，この木を目に見える形に書いた図面がＰＡＤである。ＰＡＤにより，だれでも論理構造がすっきり分かるので，論理ミスが設計段階で少なくなる上，製造検査段階でも有用である。したがって，フローチャートの代わりにＰＡＤを用いるとプログラム開発の管理が可能となり，プログラムの生産性，信頼性を大幅に向上させることができる。

昭和54年５月に発表以来，日立製作所の内外で2,000名以上の人々に対しＰＡＤの講習会，講演会などを行ない，現在ＰＡＤユーザーは急速に増加中である。

本稿では，ＰＡＤの見やすさを中心に，ＰＡＤの効果などについて概説する。

1 緒　　言

ハードウェアの設計，製造，検査などに携わってきた人が，ソフトウェアの開発を見て驚きかつ怪しむのは，下記の点についてであろう。すなわち，ソフトウェアの開発では，

(1) 設計審査の方法が確立していないこと。
(2) 部品(モジュール又はサブプログラム)が標準化されていないこと。
(3) 検査手順が確立していないこと。

もっとも，人類がコンピュータのプログラムを作るようになってから，まだ40年に満たないのである。「人類が数千年の開発経験をもつハードウェアと比較すれば，プログラムの開発方法が劣るのは当然だ。」と主張するソフトウェアの専門家もいよう。しかし，ソフトウェア開発での上記の三つの問題は，開発経験の差によるのではなくて，「ソフトウェアは人間の目に見えない」ことに起因している。**表1**から分かるように，ハードウェアと比べるとソフトウェアは，その働き(機能)も仕組(論理)も人間の目に見にくいのである。そこで，ソフトウェアをだれの目にも見え，かつ分かるようにしようとして提案したものが，ＰＡＤ(Problem Analysis Diagram)と呼ばれるプログラムの論理図面である。

ＰＡＤは，２次元木構造をした図面であり，従来からプログラムの論理(すなわち，仕組)の記述に用いられてきたフローチャートや設計用言語などと比べると，**表2**に示すような特長がある[1)]。しかし，ＰＡＤの最大の特長はその「見やすさ」にある。

本稿では，ＰＡＤによるプログラム開発の原理，ＰＡＤが見やすいわけ，ＰＡＤの有効性評価，ＰＡＤツールとその効果，ＰＡＤの普及活動状況などについて述べる。

表1　ハードウェア，ソフトウェアの見やすさの比較　ソフトウェア開発の最大の問題点は，ソフトウェアが「見えない」点にあった。

	働　　き	仕　　組
ハードウェア	格好を見れば，およその見当がつく。	見える。
ソフトウェア	格好（カードの束，磁気テープなど）からは不明。	見えない。

表2　プログラムの論理を記述する手段としてのフローチャート(ＦＣ)，構造化プログラム用言語，シュードコードなど(ＰＬ)及びＰＡＤの比較　ＰＡＤは図形なので，コンピュータ処理はＰＬよりもしにくいが，ＰＡＤのサポートプログラムの開発により，機械処理は容易になる。

項番	比　較　項　目	ＦＣ	ＰＬ	ＰＡＤ
1	論理の見やすさ	×	△	○
2	構造的プログラムを書く規律	×	△	○
3	初期テストのしやすさ	×	×	○
4	コーディングのしやすさ	×	○	△
5	データ構造の図式的記述の可能性	×	×	○
6	機械処理のしやすさ	×	○	△
7	所用スペースの少なさ	×	○	△

注：略語説明など
ＦＣ(フローチャート)
ＰＬ(プログラムランゲージ)
ＰＡＤ(問題分析図)
各比較項目につき，良い順に○，△，×で示した。

2 ＰＡＤによるプログラム開発の原理

ＰＡＤによるプログラム開発手法の核心は，頭の中でもやもやはっきりしない問題解決手続を，コンピュータにかかるように明確化する次のような手順である(**図1**)。

(1) もやもや部分の時系列的な前後関係
(2) 中心的な繰返し手続の開始条件，終了条件など
(3) 中心的な条件判定の選択条件

などを明確にすることによって，一つの大きなもやもや部分を細分化された幾つかのもやもや部分に分割する。そして，このもやもや部分がなくなるまでこの分割を繰り返す。

上記三つの細分化規則により，どのようなプログラムでも記述できるというのが，構造化プログラミングの原理である。以下に例題によりＰＡＤの記述例を示すが，ＰＡＤに現われ

* 日立製作所中央研究所　** 日立製作所中央研究所 工学博士

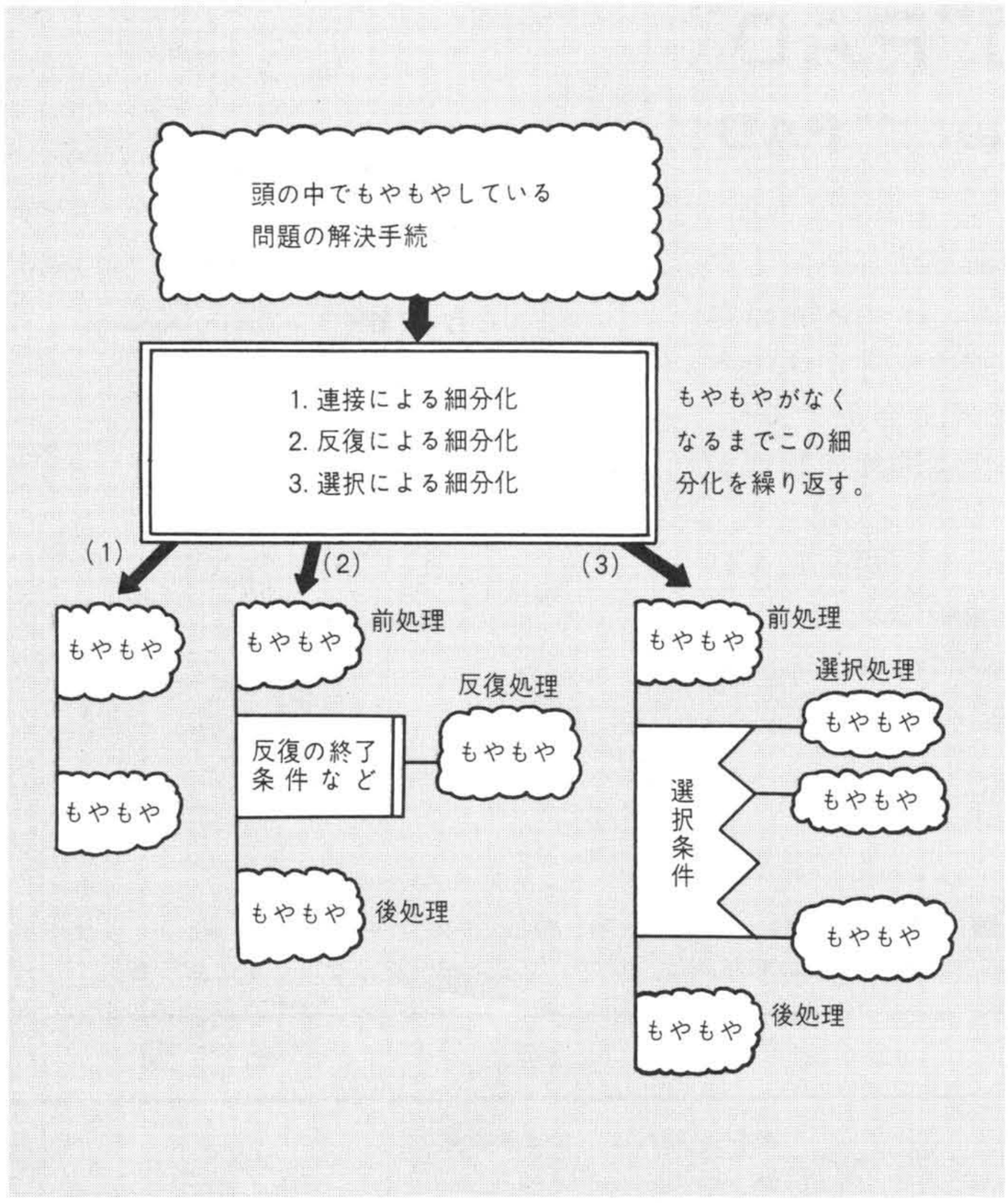


図1 問題分析図(PAD)によるプログラム開発の原理 これは，いわゆるトップダウンプログラム開発法とか段階的改良法を，より明確に述べたものである。

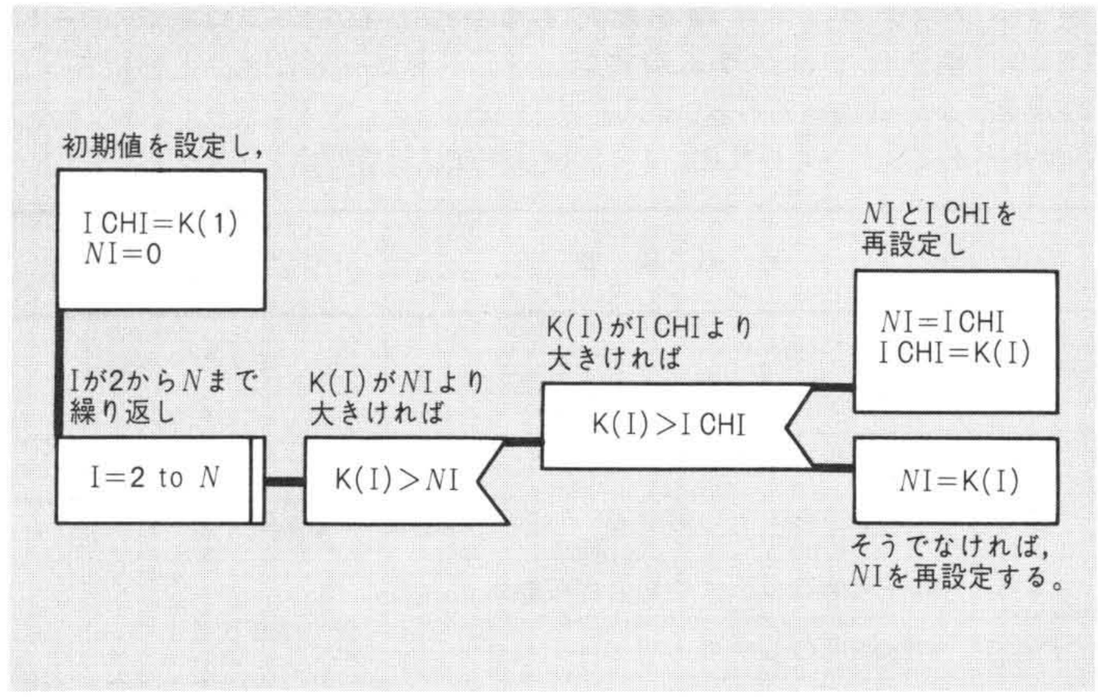


図2 ICHI-*NI*問題のPAD *N*個のデータK(I)，…，K(*N*)の中から，最大値と次値を探す問題である。

る記号は，反復を表わす□，選択を表わす◇及び処理を表わす▭の三つだけであることに注意されたい。これら三つの記号が下と右に直線で結ばれているだけである。矢印が上下，左右，斜めへと入り組んで現われるフローチャートと比べると，ＰＡＤは大変すっきりした図面ということができる。

〔例題〕配列K(1)，K(2)，…，K(*N*)に*N*個の相異なる正数が入っている。このうちの最大の数(ICHI)と2番目に大きな数(*NI*)を求めるＰＡＤを作成せよ。ただし，$N \geqq 2$とする。

この例題に対するＰＡＤを**図2**に示す。同図をたどって読むと，整った日本語としても読むことができる。

3 PADの見やすい理由

プログラムの論理構造の見やすさについて，フローチャート，設計用言語及びＰＡＤの比較を行なってみると，次に述べるようになる。

フローチャートでは，処理のまとまり，実行順序，反復及び判定の深さなどのプログラムの論理構造を，視覚的に表現しない麻のような図が書けてしまう[2]。

一方，言語は今にして思えば本来2次元的である論理構造を習慣的に1次元的に表現するものであった。すなわち，論理構造を視覚に訴えるという視点は重視されなかった。「百聞は一見にしかず。」といわれるとおり，人間が物事を理解する場合に，パターン認識能力が果たす役割は大きい。ダイクストラらにより提唱されてきた構造化プログラミングの思想は，「プログラムの論理を分かりやすく，すっきりと書こう。なるべく単純明快な構造をもったプログラムを書こう。」ということであった。構造化プログラミングにより確かにプログラムの論理は以前よりも読みやすくなった。しかし，論理構造が言葉で書かれている以上は，プログラムを理解する際に人間の論理的思考能力だけに頼らざるを得ない。人間のパターン認識能力を活用するためには，プログラムの論理構造を目に見えるようにする必要がある。構造化プログラミングにより，すっきりと整理されたプログラムの論理構造を，目に見えるようにし，プログラムを理解する際に人間のパターン認識能力も活用できるようにしたものがＰＡＤであるといえる(**図3**)。そして，ＰＡＤでは構造的プログラムしか書けない。プログラマを構造的プログラマに再教育することが，自然のうちに達成できるのである。

4 PADの有効性評価

プログラムの生産技術の効果の測定には，時間がかかるものである。例えば，ダイクストラが構造化プログラミングを提唱してから，その効果を多くの人に納得させる定量的データが公表されるまでには，5年以上の歳月を要している。ＰＡＤの有効性の定量的評価は今後の課題であるが，日立製作所はとりあえず，次のような主観的な評価を行なった。

新しい物が世に受け入れられるためには，その物がもつ「フィーリングの良さ」が大切である。そこで，「ＰＡＤを使ってプログラムを作成すると，それ以前と比べてどれだけプログラムの生産性が向上したと思うか。」という生産性向上感を，

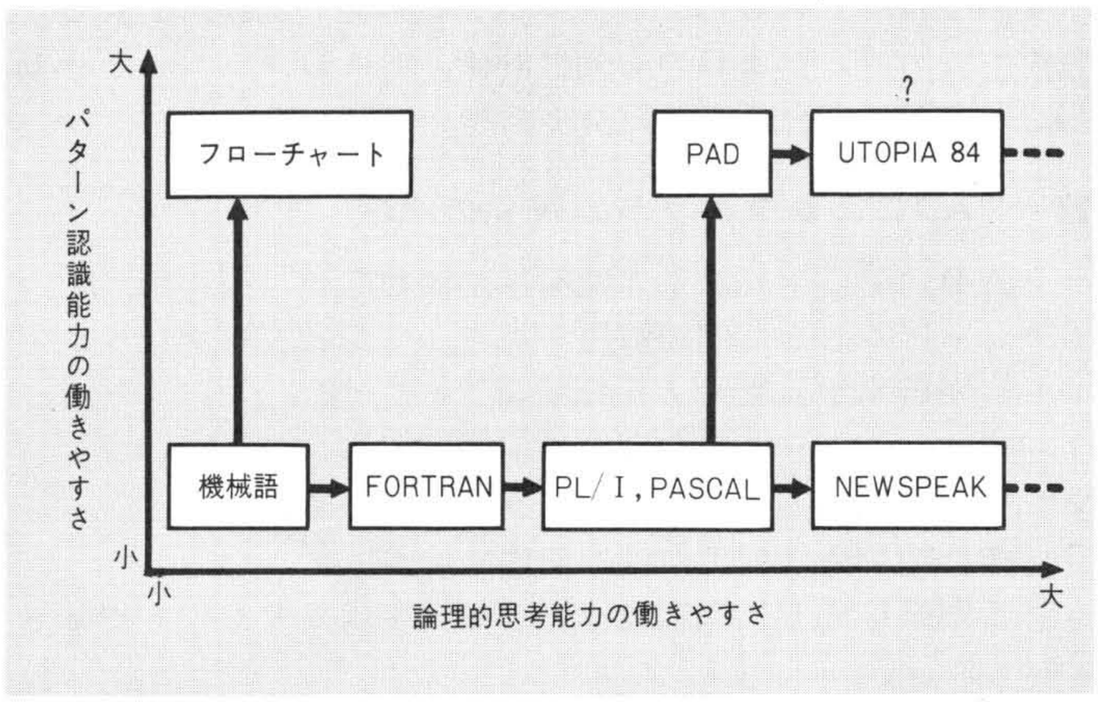


図3 人間のパターン認識能力に訴えるPAD プログラム論理の理解にも，人間のパターン認識能力を活用すべきである。

表3　PADによる生産性向上感の評価　PADの使用により，プログラムの生産性は，数割から数倍向上する。

開発フェーズ	評価		
	平均	最高	最低
機能設計	1.5	2	1
論理設計	1.6	3	1
テストケース分析	1.9	2.5	1.1
コーディング	2.0	3	1.3
デバッグ	2.6	3	1
修正・変更	1.4	2	1

6チーム(12名)から2週間ごとに3箇月間報告してもらった。作成したプログラムの種類は，オペレーティングシステムのスケジューラ，テキストエディタ，図形処理プログラム，数値計算ライブラリなどの一部である(使用言語は，PL/I，FORTRAN，アセンブラなど)。**表3**に，プログラム開発の各フェーズでの生産性向上感をまとめて示す。同表中，3という数は，「在来手法〔HIPO(Hierarchy plus Input Processing and Output)，フローチャートなど〕を用いた構造化プログラム技法で作業したのに比べて3倍ぐらい生産性が向上したと感じた。」ということを意味する。

表3のデータ量は多くないが，ＰＡＤ講習会の受講者やその他のＰＡＤユーザーの感想は，大体この数値と合致している。

5　PADの発展方向

ＰＡＤは，プログラム論理の見やすい記述法とトップダウンに，プログラムを設計するための枠組みを与えた。これだけでも，**表3**に示したような効果がある。しかし，ＰＡＤの効果を更に増大するために，ＰＡＤ作成技法及びＰＡＤツールの整備が将来の課題として考えられよう。

(1)　ＰＡＤ作成技法の整備

ＰＡＤを作成するということは，仕様書を満足すべきプログラムの論理を書くことである。この論理は，はじめ，プログラマの頭の中でもやもやしている。そのもやもやを徐々に細分化，かつ明確化し，最後にはコンピュータにかかるようにするのがＰＡＤ作成の過程である(**図1**)。ある種の手順を決め，その手順に従ってプログラマが作業すると，自動的にＰＡＤが作成できるという「ＰＡＤ作成の手順化」により，プログラムの信頼性と生産性は更に向上することが期待できる。

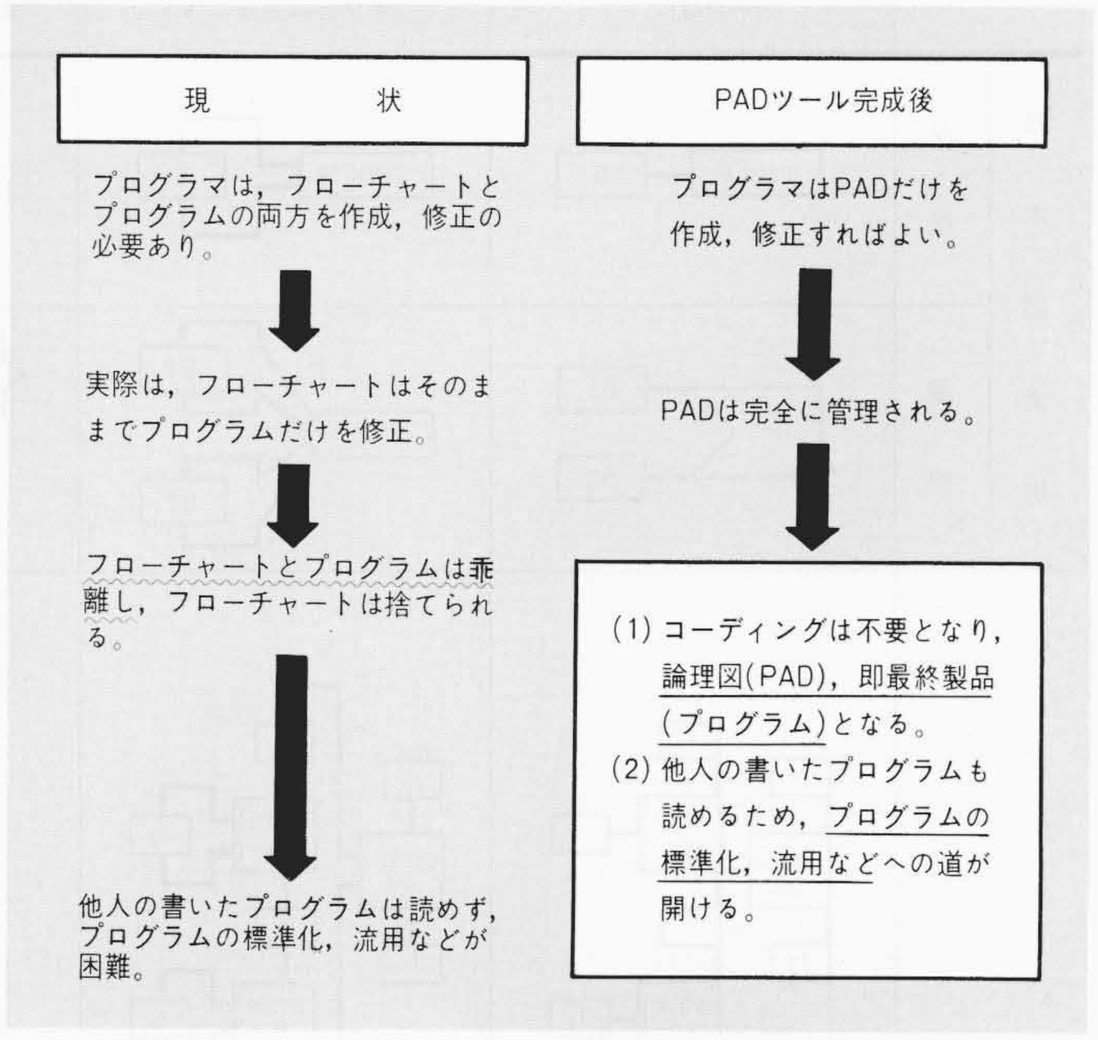
(2)　ＰＡＤツールの整備

ＰＡＤは図形であるが，フローチャートなどと比べると単純なので，コンピュータへ入力したり，処理したり，出力したりすることが容易にできる。ＰＡＤをコンピュータに入力し，ＰＡＤの変更，清書などの管理をコンピュータに行なわせ(ＰＡＤエディタ)，コンピュータに入力したＰＡＤから自動的にプログラムを作成する(ＰＡＤコンパイラ)。

このようなＰＡＤツールの整備により，ＰＡＤ以降の作業が人手を介することなく自動化され，ＰＡＤが最終文書となる。従来，フローチャート時代にはプログラムと論理図面の乖離という問題があり，プログラムリストが最終文書であった。

ＰＡＤツールなしに，ＰＡＤをフローチャートの代替として使う場合には，ＰＡＤとプログラムの間に人手作業を介するために両者の乖離は起こり得る。このときに乖離を生じさせないためには，一種の「しつけ」が必要であるが，ＰＡＤツールが整備されればその「しつけ」も不要となり，**図4**に示したＰＡＤの効果が完全に発揮されると期待される。

図4　PADツール(エディタ，コンパイラ)の効果　PADツールの開発は，日立製作所の所管研究所が中心となって進めている。



6　PAD以外の木図面

プログラムの論理的な構造を，最初に木図面で記述したのはだれだろうか。文献などで知る限りでは，それはワーニエのようである。しかし，ワーニエの方法はフローチャートに代わり得るほどには，完全な論理を記述していなかった[1)]。ワーニエ法での論理的不足を補い，拡張したものがＰＡＤであり，ワーニエ オア図である[1)]。

一方，フローチャートから出発して木図面に至ろうという努力は，国の内外で多数行なわれている[1)]。そのうちの幾つかを**表4**に示した。

7　PAD普及活動

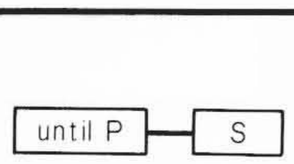
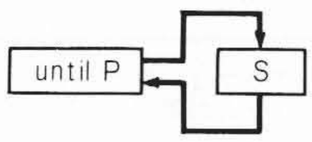
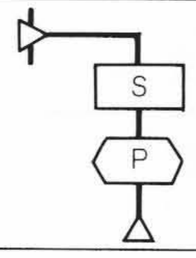
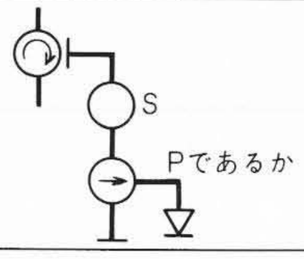
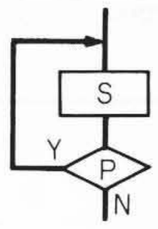
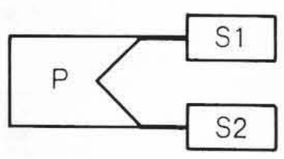
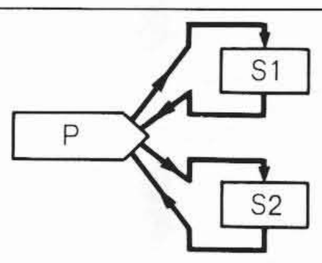
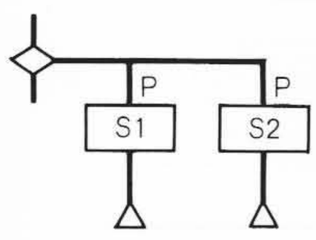
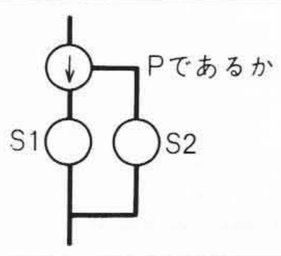
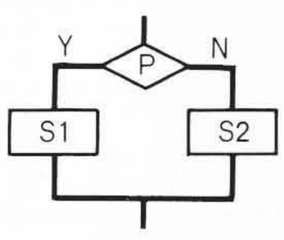
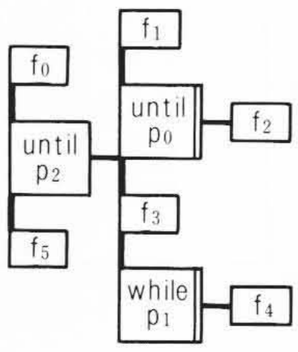
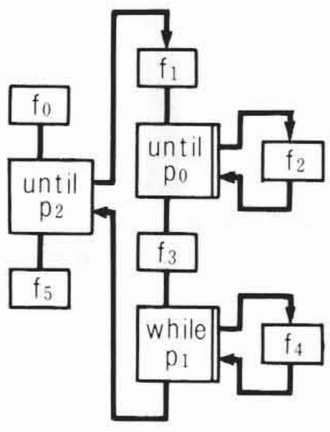
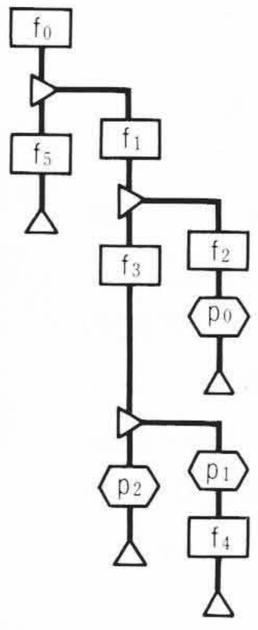
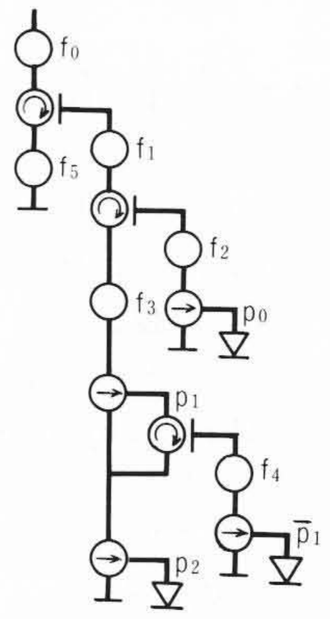
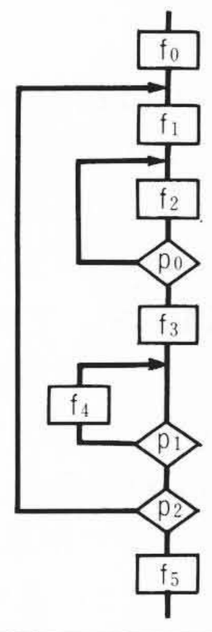
ＰＡＤにより，プログラムの論理構造を木構造として目に見えるようにすることができた。ＰＡＤを使ってプログラムを開発する様子を見ていると，プログラムの論理構造がしだいに見えてくるようになり，英語の“develop”の意味がもつ，“make visible”という感じを受ける。この感じがＰＡＤの身上であり，フローチャートやプログラム言語にないＰＡＤの特長と考えている。ＰＡＤの大きな効果を要約すると，**図5**に示すようになる。

本稿ではＰＡＤの概要について説明したが，詳細については文献[1)~3)]を参照されたい。なお，ＰＡＤ講習会としては，日立京浜工業専門学院で「ＰＡＤ専門講座」(4日間コース)を社内向けに行ない，日刊工業新聞社事業局で「ＰＡＤセミナー」(2日間コース)を社外向けに開催している。

ＰＡＤの表記法は単純なので，ＰＡＤ講習会は，従来のフローチャートによる考え方をＰＡＤ的な考え方に切り替えることに重点を置き，演習中心で行なっている。講習会の最後の時間に受講者の意見を聞くと，6割以上の人が「ＰＡＤのほうがフローチャートよりも良い。」という意見を述べている。

「どうしてＰＡＤのほうが見やすいか。」という問題は，本

表4 我が国で考案されたプログラムの木図面の例 PAD以外の図面では，フローチャートの名残をとどめているものが多い。

		PAD	木フローチャート	木構造化フローチャート	HCPチャート	フローチャート
基本図式例	反復					
	選択					
記述例						

注：略語説明 HCP(Hierachical Compact)

稿で定性的には議論したが，「ＰＡＤにより実際にどれだけ生産性が向上したか。」という定量的なデータを提供するほどのものはまだ集まっていない。ＰＡＤの定量的評価は，今後の課題である。

「ＰＡＤは良いから，組織として採用しよう。」ということになれば，在来方式から移行するための教育の問題，ドキュメント形式の問題，フローチャートとの並用の問題など，運用上の問題が当然発生する。また，ＰＡＤコンパイラ，エディタなどのツールも整備したくなる。これらの問題の解決には，若干時間がかかる。

自分の所属する組織が，ＰＡＤを正式に採用するしないにかかわらず，プログラムの生産性を向上させる個人的武器として，プログラマはＰＡＤを使うことができる。例えば，仕事の都合上，フローチャートの作成を要求されるならば，ＰＡＤ→フローチャート変換は機械的に行なえる。ＰＡＤプログラマの生産性の良さが周囲から認められれば，自然にＰＡＤは浸透するであろう。

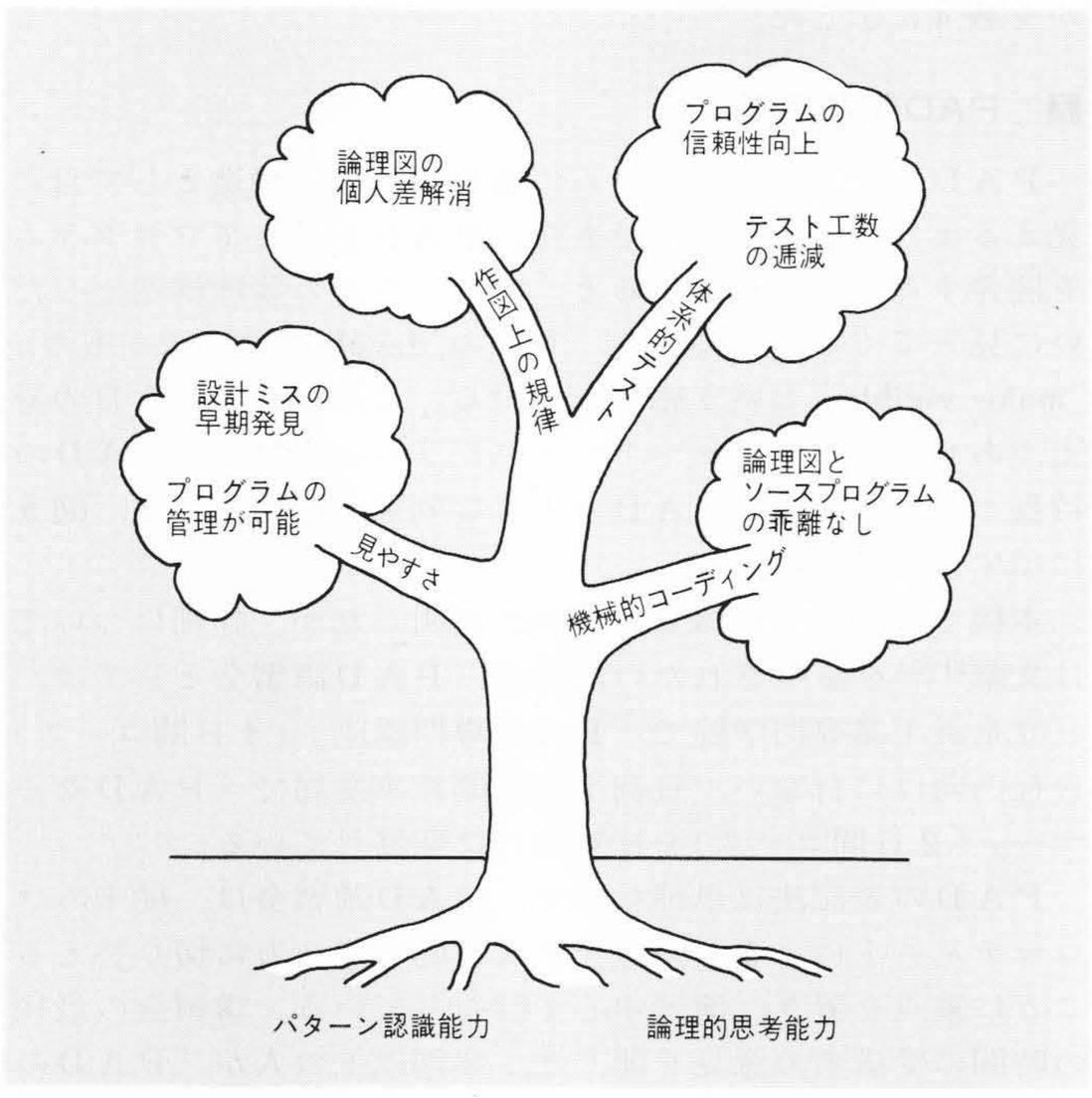


図5 PADの効果 PADの利用により，ソフトウェアの開発も，ハードウェアの開発同様，その過程が目に見えるようになる。

8 結 言

日立製作所はＰＡＤを開発し，その普及を図ってきた。科学と違い，「見やすさ」は数字では表わせないから，どれだけ理解が得られるか分からないままに，体験だけを述べてきた。幸い，ソフトウェア開発の辛苦を実際に経験している多くの人々が，ＰＡＤの良さに深く共感し，自然に構造的プログラマに変身した。今後もＰＡＤが一人でも多くのプログラマに役立てることができれば幸いとするところである。

参考文献

1) 二村，外：ＰＡＤによるプログラムの開発，bit，3月号，4月号及び5月号，共立出版(昭-55)
2) 二村，外：ＰＡＤによるプログラムの論理設計，学習コンピュータ，12月号，24～29(昭-54)
3) 二村，外：ＰＡＤ(Problem Analysis Diagram)によるプログラムの設計及び作成，情報処理学会論文誌，21巻，4号，259～267(昭55-7)