解　説

ソフトウェア・エンジニアリングの必要性*

水　野　幸　男**

1. ソフトウェアの動向

コンピュータの利用が多様化され，高度化されるにつれて，ソフトウェアは量的にも質的にもその重要性が高くなりつつある．例えば，過去 15 年間にコンピュータメーカが供給する標準的なスーパーバイザーとユーティリティプログラムのステップ数は 1000 倍以上にも増加し，もはや，ハードウェアのライフサイクルに対応して新しいソフトウェアシステムを同時に供給することは困難な状態になりつつある．一方，コンピュータユーザ側で開発されるアプリケーションソフトウェアは，コンピュータの設置台数の増加にともない指数関数的に増加している．コンピュータメーカにとってハードウェアの開発コストよりもソフトウェアの開発コストの方が実際に大きくなりつつある．今後，益々この傾向は強くなって行くものと思われる．

ユーザにおいても，コンピュータの利用領域を多様化し，拡大するためには，どうしても新しいソフトウェアを開発して行かねばならず，ソフトウェアの開発に対する投資は増大せざるを得ない．

このことはソフトウェアが従来の個人的，工芸的な産物ではなくなり，市場価値を持ち，市場流通性を有する近代的な製品になっていることを証明していると思われる．従って，ソフトウェアの開発方法は個人的，工芸的な段階から，より合理的な近代的な開発生産方法へ発展しなければならない時期に到達したといえよう．すなわち，ソフトウェアに対する高度な品質の確保と開発コストの低減，生産性の向上が強く要求される時代になったのである．

ソフトウェアは他の工業製品と異なり同一のものの繰返生産という生産形態をとらないので量産技術の適用によるコスト低減でなく，開発コストとそのメインテナンスコストの低減が重要になる．いくら開発コストを低減させても，その後のメインテナンスコストが大になるようでは意味がない．従って開発コストとメインテナンスコストの和が低減されねばならない．例えば，ある COBOL コンパイラの開発コストを 1 とすると，その 3 倍以上の費用をそのメインテナンスに投下した例がある．つまり，ソフトウェアは内部構造が複雑であり，ある部分を切り出してその部分だけを他に影響を与えずに変更することは困難な場合が多い特長を有している．

また，コンピュータの利用領域は，個別企業体のオペレーショナルな面，管理面，意志決定の面ばかりでなく社会システム，人命に直接関係する分野と拡大されており，ソフトウェアの信頼性は益々重要な問題になりつつある．ソフトウェアの信頼性は直接間接に開発コストに大きな影響をあたえている．ソフトウェアが工芸的な製品から近代的な製品となるためには，当然ソフトウェアの品質が保証されなければならない．この事はソフトウェアの生産過程において十分に品質が保証されるように品質保証の工程が明確化され実施される必要がある．

一般にソフトウェアが最も近代的な巨大な製品であるという認識が，開発生産担当者側も利用者側も未だ必ずしも十分でないように思われる．その結果，ソフトウェアの検査あるいは品質保証に対して軽視の傾向にあり，ソフトウェアの開発生産工程の期間の中に必要十分な期間と，開発投資が行われていないことがある．勿論，開発の初めには十分な検査，品質保証の期間と，マンパワーの投資を考えていたが，実際には開発がハードのトラブル等により遅れ，そのしわ寄せがソフトウェアの開発期間の短縮になり，特に，ソフトウェアの開発生産工程の最終段である検査品質保証の段階に影響が出てくる場合もあろう．

しかしながら，今後ソフトウェアはコンピュータの適用領域が拡大するにともなって，その品質は高度なものが要求されることは明らかであり，ソフトウェアの開発生産を品質保証の面から検討し，検査工程ばかりでなく，設計・製造の段階においても十分な考慮を

* Needs for Software Engineering by Yukio MIZUNO (Nippon Electric Co, Ltd. Basic Software Development Division).

** 日本電気(株)基本ソフトウェア開発本部

払う必要がある．特に，ソフトウェアの場合，ある部分の問題点が局所的なものではすまされず，その影響が他に及ぶことが多い．

以上のように，ソフトウェアの生産性の向上とその品質の向上は益々その重要性が高くなっているが，ソフトウェアの生産形態を鉱工業の発展過程と照し合せて見ると，現在のソフトウェアの生産方式は家内工業からやっとマニュファクチュア（工場手工業）形態に入ったかどうかという段階にあるように思われる．マニュファクチュア時代においては，労働手段が未だ道具であり，技術的には家内工業の域を脱していなかった．従って，その生産性は現在に比較して，著しく低く，品質もムラの多い状況であった．

現在，ソフトウェア開発の道具としては，紙，鉛筆，マシン，素朴な言語プロセッサといった極めて原始的な道具しかないように思われる．特に，新しいハードウェアのための OS を開発するような場合に使用できるソフトウェア開発の道具は極めて貧弱である．

従って，ソフトウェアの開発の生産性を向上させ，高水準の品質を有するソフトウェアを開発生産して行くためには，家内工業的な生産手段以上の高度な生産手段が必要となる．このソフトウェアの生産手段の高度化技術がソフトウェア・エンジニアリングと言う言葉で表されるのではなかろうか．

2. ソフトウェア開発生産手段の高度化と品質向上

今から数年前に，ソフトウェア開発の生産性向上の手段として PDOS (**P**rogram **D**evelopment **O**perating **S**ystem) を開発したことがある．このシステムはリモート処理機能，ファイルの集中管理機能，BPL (**B**asic **P**rogramming **L**anguage) 言語，ユーザの登録制度を有する RJE を中心にしたシステムであった．更に，その後，TSS 機能を充実させた SOFTWARE, FACTORY を実用化し（図-1 参照），現在利用している．これ等の利用経験を通じて，ソフトウェアの生産性の向上に対して，どのような事が実際に有効であったかを以下に整理してみよう．

（1） 会話形式処理

端末装置よりシステムの有する資源を有効に利用できる会話形式処理は，確かにソフトウェアの生産性を向上させることができる．

（2） 情報の共用機能と機密保護機能

情報（プログラム，データ等）の共用ができ，機密保護が論理的な情報単位で行い得ることは多くの開発者によってソフトウェアを開発する場合には重要である．

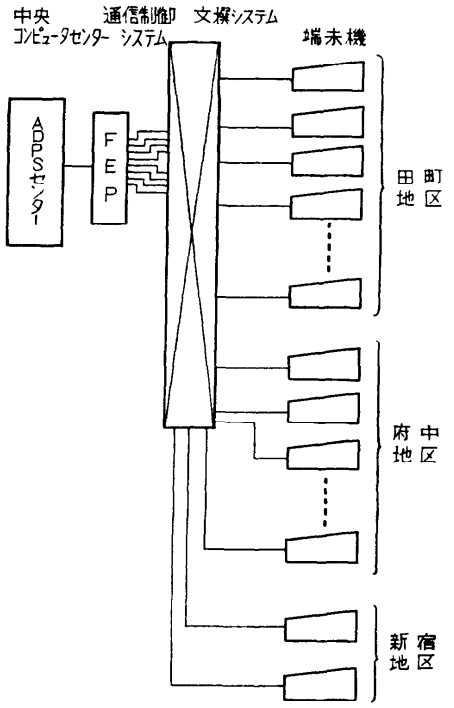

図-1 ソフトウェアファクトリ

（3） 組織化された情報の記憶と管理機能

論理的な情報単位が組織的に体系化され，オンライン的に利用可能であり，システムまたは開発管理者がシステムの資源を管理することができることが生産性の向上と管理の上で大切である．

（4） システム・インテグレーションに便利な機能

システムプログラムの開発，または大規模な応用プログラムを開発する場合には，各プログラムコンポーネントをインテグレートする作業が開発の段階において何回も発生する．この作業が簡単に短期間で行えることが生産性向上のために重要である．システム全体のシステムインテグレーション，あるいはローカルなインテグレーションが，ある期間内で何回行い得るかによって開発速度が決定されることもある．

（5） 信頼性の高いファイルシステム

開発段階においてしばしば経験するファイルの破壊は，モラールのダウンと開発工数の浪費になるので，バックアップ，リカバリー機能の充実が必要である．

（6） 高度な仮想記憶

記憶容量を意識せずにソフトウェア作成が行なえること，及び，複雑なファイルの使用法とかを憶えなくてもよいような高度な仮想記憶システムが利用できることはソフトウェアの生産性を向上させる．

（7） 適切な開発用言語とプログラム管理機能

ソフトウェア開発における開発用言語は，生産性の向上及び信頼性の点で極めて重要である．つまりコーディング，デバッグ効率とオブジェクトの能率といったインバースな関係にある両者のトレイドオフを十分考慮して，開発の対象となるソフトウェアごとに決める必要がある．

（8） ブートストラップ機能

ソフトウェアファクトリシステム上で開発されたソフトウェアを，ネイティブモード，つまり実マシン上で働かせるための確実なブートストラップ機能を準備すること，この作業は実際に思ったより多くの困難な問題，すなわちトランスファ上の問題を起すことがある．

（9） そ の 他

その他に，デバッギングツール，ダイナミックリンク機能，プロジェクト管理機能，連続運転機能等がソフトウェアファクトリとして重要である．実際に，このようなソフトウェアファクトリを利用することによって，従来の2〜5倍の能率をあげることができよう．

以上のように，ソフトウェアの生産性を向上させるためには，その生産手段に対する積極的な投資が必要になるであろう．ちょうど，家内工業からマニュファクチュアへ，更に工場工業へ発展するにつれて，その生産手段への投資が行われ，機械化されて生産性が向上したことと類似しているように思われる．勿論，鉱工業の場合には大量生産というソフトウェアの生産活動と異なった面があるが，生産性向上のための機械化，製造技術の高度化という意味では同様な発展過程をたどるように思える．従って，今後ソフトウェアファクトリシステムは，ソフトウェアの生産性向上のために是非必要な生産手段になると思われる．

次に，ソフトウェアファクトリと間接的に関連を有することで，ソフトウェアの生産性向上と品質向上の手段として重要と思われることについて述べる．まず第1は，システム記述用言語であるが，今後のシステム記述用言語は構造化プログラミングが可能な高級言語であり，また，その最適化の点より機械語とのリンク，レジスターの指定機能を有するものが必要になる

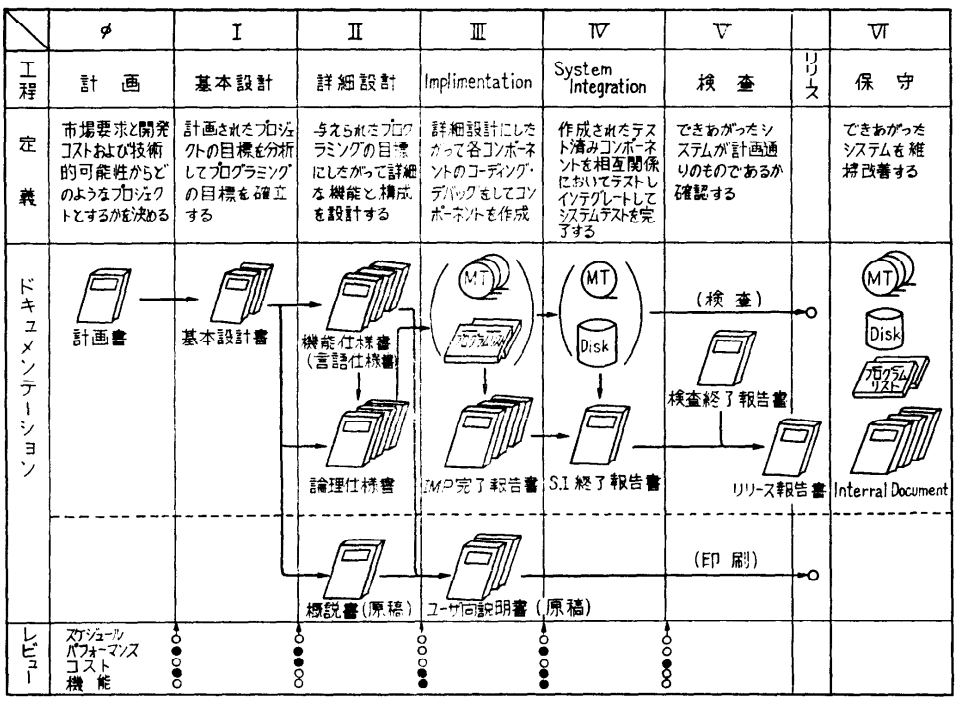

図-2　phase-plan 方式

であろう．更に，言語の拡張可能性も利用範囲の多様化に対処するために必要となる．プロセッサとしては，当然の事であるが，最適化オブジェクトの生成，診断機能の充実，デバッグ機能，高速コンパイル等が生産性向上のために益々重要になる．

第2として，構造化プログラミング手法の利用問題がある．この面においては，トップダウン・アプローチ，抽象資源の設定，セグメントの階層構造の導入，コーディングにおける GO TO 命令の減少，プログラムモジュール単位のリスティング，サブルーチンの入出口の単一化といった狭義の構造化プログラミングの段階にとどまることなく，IBM のチーフプログラム管理の例のように，マネージメント技術を含めた手法として定着させることが必要になるであろう．特に，今後の課題として，プログラムサイズの減少，例外処理の記述，構造化プログラミング支援用の小道具の準備等，その実用化技術の開発が大切であろう．

第3の問題として，ソフトウェア開発作業の標準化がある．一般の生産過程で，その昔，テーラーによって課業の標準化が行われ，その生産性が著しく向上したように，ソフトウェア開発作業においてその作成手順の標準化，ドキュメントの標準化はソフトウェアの生産性を向上させる手段となり得る．また，標準化によってケアレスミスを減少させ，信頼性をも向上させることができる．ソフトウェア開発の標準化はその開発工程の標準化とその各工程のアウトプットドキュメントの標準化が重要である．

第4は，開発工程管理と品質向上の問題である．ソフトウェアの開発管理は，ソフトウェアが物の形で見えにくいので他の管理に比較して困難である．このための管理手法として phase-plan 方式が効果的であると思う(**図-2** 前頁参照)．この方式はソフトウェア開発のためのチェックポイント管理方式であり，ソフトウェアシステムの品質，性能，コスト，開発スケジュールその他の各個別管理の基礎を確立する点にある．この方式は，開発作業の進行過程のみでなく，開発作業とその対象となるソフトウェアの両方を管理する手法であり，具体的には工程区分，ドキュメンテーション，評価の3項目を確立し，一歩一歩チェックしながら進む点に特徴がある．工程区分とは，開発作業を段階的に明確に区分することであり，例えば，計画，基本設計，詳細設計，製造，システム統合，検査というように区分し，その作業内容を規定し，担当部門，ドキュメント，評価をこの工程区分ごとに行う．

ドキュメンテーションの確立とは，ドキュメントを各工程の具体的なアウトプットとして見て行くことであり，当然ドキュメント自体の規定，運用規定(作成，発行，保管，更新の手続き)が含まれる．これにより各作業工程の作業内容，作業結果を統一化し，関係者相互に正確な情報を伝達することができ，スケジュールの把握，問題点の早期発見，品質向上が期待できる．最後の評価であるが，これは各工程の成果を評価し，次の工程へ前進してもよいかを決定するために行われる．内容としては，評価ポイントの設定，評価の時期，方法の決定，担当の決定，手続きの明確化である．ソフトウェア開発の場合に落ち入りやすい誤りとして先を急ぐあまり，次の工程へ進んでからまた逆戻りをしたり，極端な場合には最終工程へ入ってから再び初期工程へ戻ったりして開発が極端に遅れることがある．特に，性能評価が不十分なために逆戻りすることが多い．

第5の問題として，ソフトウェアの品質保証の問題がある．ソフトウェアの品質保証とは，ユーザの使用に先立って開発または修正されたソフトウェアが十分

表-1 phase-plan の3基本要素

	工程区分	Documentation	Review
意味	・ソフトウェア開発作業を段階的に区分すること	・工程の具象化をドキュメントによりおこなう	・各工程の成果を確認し，後続工程の見通しを明らかにする
必要性	・スケジュール的にも品質の面でも作業を円滑に進めるためには工程区分を明確にしなければならない	・各作業工程でなすべきことを具体的に表現するためにはドキュメンテーションが必要不可欠である	・各作業工程での作業の実行を効果的におこない，開発作業全体の成果を保障するために必要
効果	・開発の安全性を高めるようないろいろな管理手法の適用を可能とする	・正確な情報伝達が可能 ・スケジュールの正確な把握が可能 ・問題点の早期発見と品質の向上をうながす	・工程の流れを確実に把握，評価し着実な作業を保障する
内容	・工程の分割（定義と相互関係） ・各工程の作業内容の規定	・ドキュメントの種類（体系） ・開発工程とドキュメントの関係 ・ドキュメントの形式・内容の標準化 ・ドキュメントの運用規定	・Review Point の設定 ・Review 時期，方法 ・Review の運用規定

な品質を有しているかどうかを確認することであり，ソフトウェアが大規模，高度化するに従ってこの作業は重要になってきている．特に，ソフトウェア製品を使用者の立場にたって検査を行い，機能，性能が設計通り働くことを確認しなければならない．ユーザサイドでソフトウェアの事故によって発生する損害を極力未然に防止することが目的である．また最近のコンピュータシステムでは，品質保証の項目としてシステム運用における操作性が重要になってきており，この面の評価保証が必要である．更に，品質保証の対象として操作法説明書，ユーザ向説明書等も含むべきであり，これらが正確に利用しやすいように作成されているかどうかを検査する必要がある．

4. むすび

以上述べたように，コンピュータシステムの利用が多様化し，高度化するにつれ，ソフトウェアも多様化し，高度化し，新しい情報化時代の重要な製品となりつつある．従って，このソフトウェアを開発生産する手段も，生産性の向上と品質向上を目的とし，当然高度化されて行かねばならない．その技術がソフトウェア・エンジニアリングであろう．そのためには，ソフトウェアファクトリのような生産手段の高度化とその充実，さらにその運用を支援するための諸生産手段の開発が極めて重要である．

（昭和50年7月25日受付）