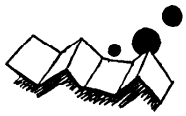
解　説



情報処理システム設計用ツールの現状と将来方向†
——ビジネスアプリケーションを中心とする——

東　基　衛††

1. はじめに

コンピュータサイエンスの進歩につれて，コンピュータアプリケーションシステムは，ますます複雑になり大規模化してきている．また，その適用分野も非常に重要な分野に多く使われるようになり，ソフトウェアの信頼性の欠陥は，利用者のビジネスに致命的打撃を与える場合も少なくない．このような状況においては，アプリケーションソフトウェアの設計技術は特に重要である．

またコンピュータのハードウェアのコストパフォーマンスの向上，人件費の上昇によるソフトウェアコストの相対的上昇によりソフトウェア開発の効率向上が重要な課題となってきている．一般にソフトウェア開発コストの30ないし40%が設計にかかるといわれる[1),2)]．しかも残りの製造とテストの効率は設計の良否に強く依存することを考えると，開発効率向上のためにも設計の重要性は高い．

ソフトウェアの設計を正しく，かつ効率良く行うためには，幅の広い知識，経験および創造性といった本質的に属人的な事が重要なことはいうまでもないが理論，方法論，事例ならびにツールが役に立つ．

設計作業は，頭脳労働に強く依存する創造活動である．一般的には，創造的頭脳労働はコンピュータなどの機械に頼ることは困難であるとされている．しかし，設計作業も細かく調べれば，計算やデータの整理などのように，非創造的な機械化の可能な作業の部分が意外に多いことに気がつく．また機械だけがツールではない．むしろ創造的頭脳労働に役立つ，あるとちょっと便利という小道具こそツールという言葉がふさわしいといえよう．

本文ではツールを広義に解釈して，ビジネスアプリケーションシステム／ソフトウェアの設計に役立ついろいろなツールを利用目的および形態面から捉え，その現状の整理と将来方向の展望を試みたものである．

2. ビジネスアプリケーションの特長と設計ツール

ビジネスアプリケーションソフトウェアは，ビジネス活動における情報処理のためのソフトウェアである．設計ツールを考えるためには，まず何を設計するのかを考えなければならない．そこでビジネスアプリケーションソフトウェアの特長を考えてみる．

企業あるいはその他の組織の活動の中の意味のある任意のひとまとまりの活動，たとえば生産，販売，在庫管理をビジネスシステムと考える．ビジネスシステムはひとつのマンマシンシステムである．ビジネスシステムを，最適化するプログラムを含むコンピュータの利用技術がビジネスアプリケーションソフトウェアである．ビジネスシステムの中の機械系（コンピュータ）の要素を集めた集合をビジネスアプリケーションシステムという．(以下単にシステムという場合，これを意味する．) このように考えると，ビジネスアプリケーションシステムは，ビジネス活動にコンピュータの利用を計画しシステム（ソフトウェア）を開発してゆく時の単位となる．

ビジネスアプリケーションシステムは次のような特長をもっている．

① 経営環境の変化が激しく，それに合わせてシステム要求の変化も激しい．
② 数多くの比較的小規模かつ単純なプログラムから構成される．
③ いくつかのプログラムがまとまってファイル(データベース)およびJCLを介して，一連のジョブとして実行される．
④ 多くのジョブが集ってシステムを構成している．
⑤ プログラムの扱うデータ項目は，ジョブ間に渡

† Current and Future Information Processing System Design Tool by Motoei AZUMA (Application Programs Department, EDP Marketing Support Division, Nippon Electric Co., Ltd.).

†† 日本電気(株)情報処理営業支援本部

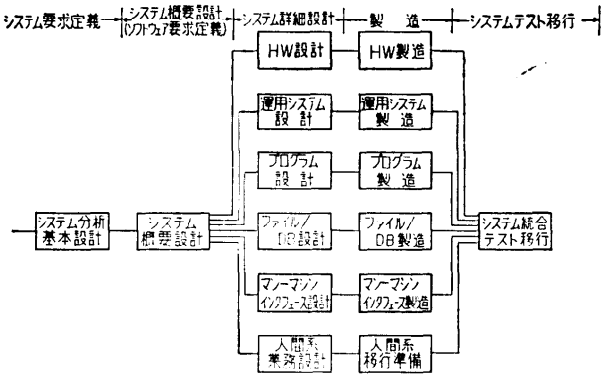


図-1 ビジネスアプリケーションシステム開発手順

って互いに複雑な関係をもつ.

⑥ ハードウェアやディスクパックなどのソースはいくつかのシステムで共有される.

このような特長から，ビジネスアプリケーションの場合には，単にプログラムを開発するだけでは所期の成果を得ることはできない．したがって，ビジネスアプリケーションのソフトウェアは，プログラムの他に，人間の業務（特にコンピュータ化による影響），マンマシンインタフェース，ファイル（データベース），および運用システムを包含するシステムとして設計されなければならない．

次に設計の手順について考えてみよう．ビジネスアプリケーションシステムの設計作業は，システム基本設計，システム概要設計およびシステム詳細設計の三段階に分けられる．

システム基本設計は，システムの要求仕様を定義するための段階である．この段階ではシステムをどのようなコンポネントを用いてどのような方法で実現するかなどという実現の方法はブラックボックスにして，システムの目的，機能，性能，信頼性，環境とのインタフェース条件，その他の制約条件などが定義される．従って，プログラムという概念はまだない．

システム概要設計は，システムの内部の論理的な設計を行い，論理的な単位を物理的なコンポネントに分割し割りあてる．次に，各コンポネントの要求仕様を定義する．つまり，ソフトウェア要求定義段階に相当する．

システム詳細設計は，システム内部の各コンポネントの物理的な設計を行う段階である．つまりここでは，プログラムのようなコンポネントを実際に製造するための設計を行う．この関係を，図-1に示す．

これまで述べてきたようにビジネスアプリケーションシステム（ソフトウェア）の設計は，オペレーティングシステムのような大規模ソフトウェアとは，その対象としての作業項目にも，また手順にも相違が見られる．したがって，ツールも，それが自動設計までゆかないにしても，システム化されればされる程，ビジネスアプリケーションに適したものでなければならない．

3. 設計時の頭脳活動とツール

ツールは道具，工具，機械などと訳されている．したがって，設計用ツールとは設計作業を助ける道具ということになる．

人間の能力は，トレーニングによって伸ばしてゆくことができるが，いくらトレーニングしても越すことの困難な能力の限界というものが存在する．たとえば，100 m を走る速さとしては，10 秒がひとつの限界と見られる．しかし，マラソンのような長距離になると，まだトレーニング次第で記録が短縮される余地が大きい．このような場合には，正しい走り方，つまり方法論の研究が役に立つと考えられる．能力の限界をさらに大幅に超すためには，道具が必要である．人間の速さの能力の限界を越すための道具としては，自転車やスケートが，また力の限界を越すための道具として，テコが発明された．自動車やクレーンのように動力を用いることによって，これらの能力の限界は一層高められる．

頭脳活動の能力の限界についても，同様のことが考えられる．たとえば人間が一度に記憶できる数字は，普通の人では 10 桁位が限界である．一分間に計算できる加減算の数にも限界があるが，これはそろばんというツールの利用で高められる．考えを正確に伝える場合の正確さや一度に伝えることの可能な人数にも，限界がある．このような人間の頭脳活動の能力の限界を越すための最も基本的なツールは，紙と鉛筆である．

紙と鉛筆があれば，日常的な頭脳活動の大部分はなんとか処理してゆくことができる．しかし能率という点から考えると，字を書いたり，画を描いたりする時の手の速さで制約されることが明らかになってくる．これを設計作業について考えると，設計作業を分担したり，要求者と仕様の打合せをしたり，また製造者に仕様を示したりしなければならないので，単に設計者本人に理解されるように書くだけでなく，他人にも理

解できるように標準化された様式できれいに書かなければならない．また，設計は通常一回の作業でピタリときまるものではないので，書き直しが必然的に多くなる．したがって，書く速さによる頭脳活動の制限は，かなり厳しいといわねばならない．こうした観点にたってみると，設計作業においてもツールの研究の範囲はかなり広いことがわかる．

次に，設計作業にはどのような要素作業があるかを考え，それらを行うための人間の能力の限界をどのようなツールがあれば越し得るかを考えてみよう．

1900年頃アメリカで，ギルブレス(F. B. G. Gilbreth)は物を生産する時の最も良い作業方法を見つけ，またその標準的な時間を定めるために，作業を，“見る”，“手をのばす”，“はこぶ”，などという要素作業に分けることを考えついた．そして個々の作業に記号を定め，自分の名を逆にしてサーブリック (Therblig) と名づけた．その後 WF (Work Factor) MTM (Method Time Measurement) などの研究へと発展した．これらは PTS (Predetermined Time Standards Systems) といわれ，現在でも経営工学 (IE: Industrial Engineering) の中で重要な意味を持っている．この考え方は，その後事務作業の中にも試みられた[3]．

ここでこれらの話を持ち出して，設計作業の標準時間をきめようなどというつもりは今のところ毛頭ない．しかし設計作業を IE 的なセンスで要素作業に分け個々の要素作業がどのような条件の時効率が上がるかということを考えることは，良いツールを考えるうえで役に立つと思う．

一般に，システム設計など主として頭を使う仕事は，次のような要素作業に分けられる．

① 気がつく：無意識下に視覚，聴覚などの感覚を通して脳に伝えられた情報を意識下として確認することをいう．思いつく，発見するなども含まれる．

② 調査する：ここでは作業に必要な情報を意識下に認識することを意味する．読む，聴く，探す，観察するなどの他，思い出すも含める．

③ 理解する：情報の価値を認識することをいう．

④ 創造する：脳の内部で新らしい情報を発生することをいう．それ以前に記憶されている情報と直接の関数的または論理的関係はない．仮説をたてるとか，代替案を考えることが該当する．

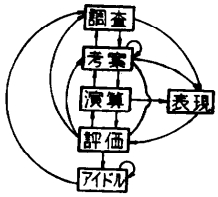


図-2　設計の要素作業間の状態遷移

⑤ 演算する：情報の関数的または論理的演算を行う．

⑥ 整理する：情報を並べたり，分析又は合成あるいはその他の関連づけを行う．ここでは頭の中での操作のみを意味する．記憶もこの中に含める．

⑦ 表現する：頭の中で整理された情報を文字や記号によって紙の上に書く．

⑧ 評価する：情報を比較し，相対的に価値づける．

⑨ 決断する：意思決定を行う．

⑩ 伝達する：意識下にある情報を他人に伝える目的をもって紙の上に書かれたものを呈示し，もしくは，口頭で伝える．口頭の場合表現と同一になる．

これらの要素作業の間の状態の遷移は大変複雑な形となる．そこで①と②を調査，③，④を考案，⑤，⑥を演算，⑦，⑩を表現および⑧，⑨を評価とすると図-2のような状態遷移図で表わされる．

このように考えると設計のような頭脳労働の中でも真に頭の中で行われる作業は考察のみで，その他はツールの活用によって効率と質を向上させ得るといえよう．しかも，考察そのものも，その他の作業の質の向上により，効率と質の向上が期待される．例えば調査から表現へのパスは，フォームシートなどのツールによって始めて可能となる．これらの要素作業は，次のようなツールによって効率を上げることができる．

- 調査支援ツール：事例集，情報検索システム，マイクロフィルム装置，KJ ラベル，フォームシートなど．
- 演算支援ツール：コンピュータ，電子卓上計算機など．
- 表現支援ツール：フォームシート，テンプレート，ドキュメンテーションエイドなど．
- 評価支援ツール：デシジョンテーブル，アナライザーなど．

4. 設計用ツールの現状と分類

ソフトウェア工学の中でも設計用ツールは比較的遅れた分野であると思われる．特にまとまった文献となると少ない．しかし実際には，設計作業のいろいろな局面において，大小さまざまのツールが用いられている．これらの多くは体系化されることはなく，職場内に限定されて使われている．従ってそれらを工学的見地から整理し体系化することは，今後の研究のために意義があると思う．ここでは設計用ツールを，設計作業のために開発された専用のツールだけでなく，設計作業に使うと便利な一般のツールを含めて考えることにする．こうした見地から見たツールの分類およびその現状は次のようである．

（1） 事務用品および小道具

設計作業は，その多くは机上の頭脳作業であるから，事務作業やハードウェアの設計作業と共通した性格をもっている．したがってそれらの分野で使われるツールは，設計の役に立つ．例えば次のようなものがある．

- 一般事務用品：定規，ナイフ，紙などがある．これらはソフトウェア作業という見地から十分な研究がされているとはいい難い．特に汎用の記録用紙はソフトウェア設計の基盤となるもので，その紙質，ケイなどは生産性に重大な影響をもつ．
- テンプレート：プログラム用のフローチャートはあまり書かれなくなる傾向があるが，設計段階ではプロセスフローチャートや，システムのハードウェア構成図などに広く用いられている．個々の記号は JIS 化されているが，コーナーの形状や，断面，紙の大きさとケイ線の間隔と関連した記号の大きさなど研究の余地がある．
- ラベル：最近 KJ 法を始めとして，いろいろな発想法や整理術が実用化されてきているが，これらに適したラベルやカード台紙，キャビネットなど今後ソフトウェア向きのものを期待したい．

（2） データおよび基準値表

設計作業，特に性能評価などでは，それ程厳密な有効数字は要求されず，3桁もあれば十分なことが多い．また公式にあてはめて計算することが少なくない．このような場合には計算図表のようなものが役に立つ．例えば次のようなものがあるが，コンピュータを活用しやすい分野だけに今後余り発展は期待できない．

- ソートタイムテーブル：テープベースシステムが主流だった頃活用された．
- 回線網のトラフィック計算表：これも最近はコンピュータにとって替わられてきている．

（3） フォームシート

フォーム（様式）の一定なシートをあらかじめ印刷しておき，空欄を埋めてゆくことで文書を作成する．文書の標準化と効率化が同時に達成できる．履歴書を始めとするビジネスフォームが，人事管理など広範囲に利用されて効果を上げている．現実的なツールとして，システム要求定義や設計にも用いられている．今後コンピュータ利用による自動設計が実用化されてゆくとしても，ハンディな利点があるので存在価値が減ることはないだろう．例として以下のようなものがある．これらはいずれも方法論と組み合わせて提供されているが，その提供形態は，一般図書，教育コース，専門のパッケージなどいろいろである．

- Hand Book of Data Processing Management: Brandon Systems Press から出版され Auerbach に引き継がれた図書である[4]．
- ADS (Accurately Defined Systems): NCR 社で開発された．5種類のシートを用意している[5]．
- PRIDE (Profitable Information by Design): アメリカ MBA 社の Milt Bryce により開発された．15種類のワークシートを提供している[6]．
- SPECTRUM 1: 米国の J. Toellner & Associates により開発された．シートの種類や内容は不明である[7]．
- STEPS (Standard Technology & Engineering for Programming Support): 日本電気で開発された．全工程で136の[8)~10)]ワークドキュメントに26種類のフォームシートが提供される．

（4） スタンドアロン ソフトウェアツール

一般にただソフトウェアツールといった場合には，これを指す場合が多い．設計段階の個々の特定の作業を支援する目的で，独立に用いられる．デバッグ，テスト，保守工程に比べて，設計用のものは全般に少ない．しかし，その体系が整理され，流通する環境が整備されれば，大いに発展が期待される分野である．現在のところ必要に応じてチーム内で作られるという性格が強いため，特定の名前を挙げることは困難である

が例えば次のようなものがある.

- データ設計のためのアクセス頻度調査用
- メッセージバッファ最適化用
- 入出力 CRT 画面設計用例えば SCREEN[11]
- ドキュメンテーション支援用例えば Flowcharter[12]
- パフォーマンス予測用例えば SCERT
- 回網設計用例えば NETS[13]

(5) 汎用アプリケーションプログラム

汎用アプリケーションプログラムの中には設計作業の支援に役に立つものがある. これらは一般によく知られているので具体的製品名は省略するが, 例えば次のようなものが利用できる.

- 性能設計用として GPSS タイプのシミュレータ
- 設計情報提供用として, 特にファクトリトリーバルの可能な情報検索システム
- 同様にデータベース管理システム

(6) 設計言語およびその関連ツールシステム

ソフトウェア工学の関係者で設計ツールといえば, これを指すといっても過言でない程関心は高い. 一般的にシステムまたはソフトウェアの仕様を記述するための言語がきめられ, その言語で記述された仕様を, 設計者に見易いように表現を変えて出力するドキュメンテーションエイドと設計の妥当性の評価を助けるためのアナライザから成る. その言語のタイプによって, 大体次の系統に分けられる.

① 設計用の新らしい言語を定める場合

ミシガン大学の ISDOS の PSL/PSA[14], ケースウェスタン大学の SODA[5], TRW 社の SREM の RSL/REVS[15] などがある. また日本では, 通産省の援助で開発を行っている協同システム開発(株)の CPL-A, CPL-B, 富士通(株)の SDSS, (株)日立の PPDS, 日本電気(株)中研の SDMS などが現在開発中である.

② 高級言語タイプの場合

これは COBOL, FORTRAN などの在来言語を拡張する場合とそれと同等水準以上の新らしい言語を開発する場合がある. このような高級プログラミング言語は, 製造段階支援ツールであるという考えもある. しかし高級言語の水準がどんどん高くなっている現在, 設計言語との区別をつけるのはむつかしい. 勿論, 目的プログラムへの翻訳可能性をもって区別することも可能である. しかし, 一度 PL/I や COBOL などの言語を介するにしても, 翻訳可能性を確保することは設計言語の目標である. したがって, ここでは従来のコンパイラ言語より高い水準にあること, および, 翻訳以外にドキュメンテーションエイドなど機能を備えていることという条件を満す言語は, 設計支援ツールと考える. この場合は, 主として, 詳細設計(プログラム製造設計) 支援ツールとなる. 例としては PASCAL, ADR の METACOBOL[16], 日本電気(株)の COBOL/S[17] などがある.

③ 簡易言語又はパラメータ言語の場合

これにも, 直接翻訳するタイプと, プリプロセッサタイプがある. いずれも, 一般ユーザまたはエンドユーザが定型的業務処理プログラムを簡便に作るために用いられる. その狙いは, プログラムの要求仕様がきまれば, プログラムの設計工程を, 経ずに直接製造することである. 例としては, RPG, (株)日立の HELP, 日本電気(株)の APG, 構造計画研究所の GENASYS[27], JASS グループの JASPOL[28] などがある.

(7) 事務用機械

事務用の機械類にも, 設計の効率を高めるツールとして有効なものが多く見逃せない. 勿論ほとんどのシステム技術者は良く使いこなしており, 今更いうまでもないことではあるが.

- カーボン複写機: ゼロックスを始めとする白黒のコピーは, メンディングテープ, カッターと共に用いて有力な設計ツールとなる.
- ファックス: 遠隔地の利用者との打合せなどに威力を発揮する.
- 電子卓上計算機: 現在はポケット型のものが主流となっている. 操作の簡便さでは, TSS や RJE が普及してきたといっても, ストアードプログラム型コンピュータをはるかにしのぐため, 設計用ツールとしての存在価値はゆるがない.
- タイプライタ: 欧米では常識となっているが, 日本では漢字の問題が解決しないため普及は困難である.

(8) インテリジェント事務用機械

マイクロコンピュータの急速な進歩により, 従来の事務機と組み合わせて今後大いに発展が見込まれる分野である. 設計用のツール, いわばインテリジェントデザインエイド (IDA) は, 今後ソフトウェア工学の関係者には興味深い課題となるだろう. まだ, この分野は例が少ないが, 次のようなものが考えられる.

- ワードプロセッサ: 米国では既にかなりの普及をみており, ひとつの市場を形成している. 一度利用してみるとその便利さは, 従来のタイプライタ

とは比較にならないことがわかる．しかし，漢字，ひらがな，カタカナ，ALPHABET を使う日本で実用化するにはまだ時間が必要である．

・マイクロフィルム検索機：設計作業の中でも利用者に近いところほど言語などフォーマルな記述がなじまない．この性質が急になくなることは考えられないので，設計言語に良いものができたとしても従来形式の文書はなくならないだろう．マイクロフィルムは，COM と合わせて発達すると思うが，ひんぱんに修正を繰り返す設計工程で利用が進むとは考えられない．むしろ保守工程向きだろう．

5. STEPS の概念

設計ツールの今後の方向を考えるためには，コンピュータサイエンスの今後の方向と現在良く使われているツールの分析が必要である．そこでそのツールとして，ISDOS と STEPS をとり上げてみることにする．ISDOS の概要は他の項に譲るとして，ここでは STEPS の概念にふれてみる．

STEPS には現在 STEPS-2 と STEPS-4 があるが，これらの違いは対象とするアプリケーションシステムの規模が異なるだけで基本的には同じ概念に基づいている．STEPS はシステム開発および運用に関する諸々の標準の総称であり，システム開発標準とプログラミング標準から構成されている．

STEPS のシステム開発標準は，問題の認識から始まるシステム分析段階から，システムの完成，移行までの全工程の手順，作業方法，フォームシート，ワークドキュメントおよびドキュメンテーションを一貫した思想と技術のもとで体系化したものである．

手順は概念的には **図-3** のような5つのレベルで定義されている．フェーズは，開発推進に関する意志決定の区切であり，アクティビティは日程管理上の単位，また，ワークセットは技術上の単位となる．STEPS は

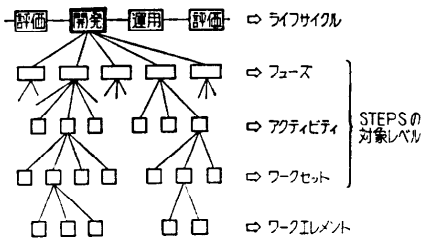


図-3 STEPS の手順のレベル

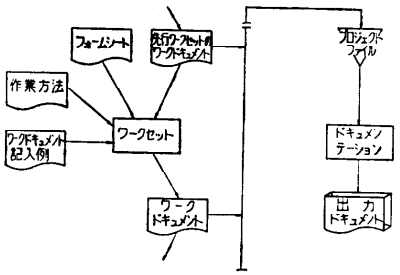


図-4 STEPS システム開発標準の概念

この三つのレベルごとに手順を標準化する．ワークエレメントは前述の要素作業に相当する．

作業方法，ワークドキュメントおよびフォームシートは，このワークセット単位で定められている．これらの関係は **図-4** に示す通りである．ここで作業方法とはワークセットの作業の進め方，記号の標準，ヒントなどである．使用可能なツールもここで示される．ワークドキュメントは，ワークセットの作業結果を表現したものである．フォームシートはこのワークドキュメントごとに標準的な様式を印刷したもので，ドキュメントの標準化と作業の効率化を狙っている．仕様書や説明書などはこのワークドキュメントを選択し編集するだけで得られ，この作業をドキュメンテーションという．

現在，STEPS-4 の場合，フェーズ数は5，アクティビティ数は34，ワークセット数は119である．

プログラミング標準は，定型的なビジネスアプリケーションのプログラムを，照合，更新，作表，問合せ(オンライン) などのパターンに分け，パターンごとに標準的なプログラム仕様，プログラム構造，および標準 COBOL ソースコーディングを定めたものである．また，各パターンに共通なデータタイプごとのデータ名のつけ方やコーディング規則をも，標準として定めている．このため，プログラム設計・製造の手間を省くことのできるツールである．

この STEPS を開発するにあたり，前提とした条件は次の通りである．

(1) システム工学やソフトウェア工学の背景としての理論と整合がとれていること．

(2) 一貫した方法論を開発し，それと一体となっていること．

(3) 開発の全工程を一貫した概念で網羅し，作業の連続性を保つこと．

(4) 手順のフレームワークを設け，個々の作業は HOW より WHAT をはっきりさせること．

(5) 開発対象としてのシステム，およびソフトウェアの概念を明らかにすること．

(6) 汎用性の高い抽象的なものより，対象をビジネスアプリケーションにしぼって具体的なものを作ること．

(7) 全体を通して使うことは勿論，個々のワークセット単位で，独立に使っても役立つこと．

6. 設計ツールの将来方向

設計ツールの将来方向を考える場合，前述の(1)～(8)までの分類のいずれも重要であることはいうまでもない．しかし，ソフトウェア工学の対象として見た場合には，(3)のフォームシート，(4)のスタンドアロンソフトウェアツール，(6)の設計言語およびその関連ツールシステムならびに(8)のインテリジェント事務用機械が重要である．そこでこれらが将来どのような方向に進むか，考えてみたい．

フォームシートは，今後ますます洗練されて，ソフトウェアの種類ごとに適したものが使用されてゆくことと思う．今後の分散システムやデータベースシステムの普及，および簡易言語や高水準言語の発達と合わせて考えると，プログラムの設計からシステムの設計およびプロジェクト管理用に重点が移行してゆくであろう．しかし，マイコンや端末，特にグラフや漢字の扱えるワークステーションが発展して，コストダウンが実現すると，フォームシートの利点がやがて失われ，かなりの部分は機械化されてゆく．

次にスタンドアロンソフトウェアであるが，これもフォームシートと同様，プロジェクト内で必要な都度簡単なものを機動的に作ってゆくという本質は失われることはないと思う．特にワークステーションの発展により，この傾向は促進される．しかしコストダウンの要請は，一般の企業での恣意的なツール開発の制約となるため，汎用のツールとしてシステム化されたものへの要求が高まるだろう．

設計言語関連ツールシステムは，現在のところ最も進歩した狙いを持っている．しかしそれが日本で普及するためには，日本と欧米の文化の差に注目して，日本的設計言語関連ツールシステムを開発してゆかなければなるまい．それには，次の日本文化の特長に注目したい．

① 漢字を使う

② タイプライタを日常使う習慣がない．

③ 日本は均質民族であるため，文書による厳密なコミュニケーションの習慣に欠ける．

最後にインテリジェント事務用機械であるが，これはマイコンの発達とコストダウンにより，設計言語関連ツールシステムと大差のない機能を提供するようになるであろう．

このように考えてくると，設計ツールの将来方向は，スタンドアロン型，端末型あるいは直結型のいずれかにより，会話的に設計を進めるデザイナーズワークステーションのようなものになるであろう．このようなツールの提供する機能と特長は，現在米国でよく使われているISDOS，および日本で良く使われているSTEPSを分析すると予想できる．

設計ツールとして見たSTEPSの長所は，ISDOSと比較して次のようなことが考えられる．

- 日本語がそのまま使用できる．
- 頭の中でパターンとして考えたものがそのまま記述できる．
- 設備が不要でハンディである．
- 方法論と一体となっている．
- 分析からテストおよび移行までの一貫性を持つ．
- 事例が提供される．
- 機種に関係なく使用できる．

一方，STEPSと比較したISDOSの長所は次のようなものである．

- ドキュメンテーションが自動化されており，手書きによる速度の制約から解放される．
- 設計の修正，変更の場合，その部分のみ修正すればクリーンなドキュメントが得られる．
- 解析が半ば自動化されており，人間の手間が省け信頼性の向上が図れる．

これらから得られるツールの機能要求条件は，次のようなものである．

- 漢字およびグラフの扱いうる会話的オペレーション．
- 設計情報および事例の検索と表示
- 作業手順のガイド
- 作業のチェックリスト
- ドキュメンテーションエイド
- アナライザ
- ソースプログラムとの連続性
- 運用システムとのインタフェース
- 管理データの蓄積とレポート

7. おわりに

われわれ技術者は，とかく技術的興味という強い誘惑に惑わされて，本来の目的を見失いがちである．そこで本文では，設計用ツールの本来の目的に立ち戻ってその体系化を試みた．次に，現在良く使われているツールを分類し，その機能や特長を述べた．勿論ここに取りあげたものがそのすべてではない．専門の研究者ではないので，見るべき文献での見落しがあるかも知れない．筆者が重点を置いたのは，一般のコンピュータユーザの眼の届く範囲にどのようなツールがあるか，それをどう評価するか，更に今後どのようなツールが求められているか，ということである．いわばこれは今後の設計ツールの研究・開発への問題提起でもある．したがって，これが更に検討されて，実際に開発されるまでには，かなりの研究と醗酵の時間を要すると思う．

末筆ながら本文をまとめるにあたり助言や協力をいただいた，篠澤昭二，藤野喜一，岩元莞二，紙谷進の各氏に感謝の意を表する．

参考文献

1) Wolverton, Ray W.: The Cost of Developing Large-Scale Software, Quantitative Management: Software Cost Estimating, Compsac 77, p. 156.
2) Aron, J. D.: Estimating resources for large programming systems, 同上, p. 257.
3) S. A バーン他，村松林太郎訳，事務作業と標準時間，日本能率協会，p. 300 (1965).
4) Rubin, Martin L.: Hand Book of Data Processing Management, Vol. 1～6.
5) Ramamoorthy, C. V. 宮本勲訳：ソフトウェア要求と仕様化技術，bit，8臨時増刊，pp. 63-119 (1978).
6) 松平和也：構造的データベースアプローチ PRIDE，ビジネスコミュニケーション，Vol. 15, No. 3, pp. 43-48.
7) SPECTRUM-1, J Joellner & Associates (小冊子).
8) STEPS-2 概要説明書，日本電気.
9) 水野幸男，東 基衛：コンピュータソフトウェアの標準化，日本経済新聞社，p. 864 (1977).
10) 永野幸男，東 基衛：管理者のためのソフトウェアの標準化，コンピュータレポート，1978・7～1979・1 (連載).
11) ACOS-2 リモート処理管理，サービスプログラム説明書，日本電気.
12) COBOL FLOWCHARTER 利用者マニュアル，情報処理振興事業協会.
13) Teshigawara, Y. A: Communication Network Design Tool-NETS, Proceedings COMPCON 78, pp. 158-165.
14) Teichroew, D. 他，山本裕三訳：PSL/PSA 情報処理システムの要求定義解析・文書化システム，bit，8臨時増刊，pp. 152-176 (1978).
15) 宮本 勲訳：解説 TRW 社の SREM, bit, 8臨時増刊，pp. 177-209 (1978).
16) Meta COBOL User Guide 他: Applied Data Research.
17) COBOL/S ドキュメンテーションエイド説明書，日本電気.

(昭和54年2月26日受付)