情報システム構築のための「ソフトウェアエンジニアリング」の研究

牧　野　　勝*

Research of SFE (Software Engineering) for construction of information system

Masaru MAKINO

There are many thema in the field of software engineering. For example, "Software Requirements Specification", "Structured Software Design", "Object Oriented Programming", "SWQC (Software Quality Control", "CASE (Computer Aided Software Engineering)", "Open Down Sizing and Networking", "CSCW (Computer Supported Cooperative Works)", "AI (Artificial Intelligence)", "Multi-Media", "DTP (Desk Top Publishing)", "Heuman Interface Software", "Project Management", etc.

In this paper, I would like to write about the following problems (1) Engineering item of software engineering, (2) OOS (Object Oriented Software), (3) Engineering item of multi-media.

I think these two item, those are OOS and multi-media, should have great priority among thema of software engineering in these days.

まえがき

当論文では，情報システム構築のためのSFE（ソフトウェアエンジニアリング）について，その技術的項目を総合的体系的に研究し，特にオブジェクト指向ソフトウェア技術とマルチメディアに重点を置くこととする。何故なら，今ソフトウェアエンジニアリングは情報システム業界の構造的不況と共に困難な時期を迎えており，その困難な状況を打開するための二大技術テーマがオブジェクト指向とマルチメディア技術であると信じられるからである。

ソフトウェアエンジニアリングは明らかに曲り角に来ている。ソフトウェアは，かつて"ソフトウェア工場"の名の下に大量オートメーション方式の生産を目指して来た。しかし，それでは"ソフトウェア危機"が乗り切れないと思われるようになった。現在，オブジェクト指向によるソフトウェア技術がソフトウェア危機の救い主と唱えられている。また，マルチメディアの技術は

*経営工学科

ハードウェア（コンピュータおよび通信システム）およびソフトウェア共に有望な技術として期待されている。情報システムの関連技術がこの二大技術によって活況を呈し，ひいてはソフトウェアエンジニアリング全体が新しく生れ変れるよう期待したい。

1．システム設計からソフトウェア設計へ

情報システムを構築する場合，そのアプローチの段階として「計画」，「設計」，「製作」，「試験」の４つの段階が存在する。さらに，「設計」という段階は次のように分割できる。

<table>
<tr><td>段階
区分</td><td>組織部門</td><td>システム設計</td><td>ソフトウェア設計</td></tr>
<tr><td rowspan="2">基本設計
（外部設計）</td><td>利用部門</td><td rowspan="2">基本設計</td><td rowspan="2"></td></tr>
<tr><td>開発部門</td></tr>
<tr><td rowspan="2">詳細設計
（内部設計）</td><td>利用部門</td><td rowspan="2">詳細設計</td><td rowspan="2"></td></tr>
<tr><td>開発部門</td></tr>
<tr><td>技術者名</td><td></td><td>システムズエンジニア</td><td>ソフトウェアエンジニア</td></tr>
</table>

図１．情報システムの設計

情報システムの設計は，システム設計からソフトウェア設計へ段階を追って進められる。システム設計もソフトウェア設計も，共に“基本設計”と“詳細設計”に分れる。そして，図１でわかる通り組織部門としては，情報システムの利用部門と開発部門が存在し“システム設計”は利用部門も参画して進められるが“ソフトウェア設計”は開発部門（いわゆる情報システム部門やソフトウェア会社などの専門家集団）が主として行うことになる。ただし，利用部門のニーズは“要求分析”の段階で“要求仕様書”として作成されるので，ソフトウェア設計に反映される。

言うまでもなく，“設計”は物を作る場合非常に重要なステップである。設計にミスがあると，製作工程にマイナスの影響を与え，当該プロジェクトのスケジュールや納期や原価および製品の品質等あらゆる事象に悪影響を及ぼすことになる。

特に，ソフトウェア設計においては一般的な“設計”に加えて，品質設計や信頼性設計という後工程に関係する。即ち，ソフトウェアエンジニアリングの最終段階である“情報システムの試験”の段階で，“情報システムの設計”という初期段階での作成内容が大きく影響することになり最終段階でプロジェクト崩れを起しかねないという危険性が内在している。

目に見えないソフトウェアという製品を製作する場合，設計段階（特にソフトウェア設計の段階）で十分に試験項目および品質・信頼性の項目を盛り込んでおかねばならないのである。

2．ソフトウェアエンジニアリングの技術項目について

ソフトウェアエンジニアリングの技術的工学的な内容項目については，情報システム構築作業の段階別に(Ⅰ)ソフトウェア要求分析，(II)ソフトウェア設計，(III)ソフトウェア製作，(IV)ソフトウェア試験，および(V)プロジェクト管理に分けることができる。

以下，順を追ってこれらの項目について検討を進める。(表1，表2参照)

2．1　ソフトウェア要求分析

ソフトウェア要求分析を行うことはかなり難しい面が多い。何故なら相手にするのは“人間”であり，そして実現する場は“コンピュータという機械”だからである。

ソフトウェアに対する利用者からの要求としては，基本的な要求（ソフトウェアの機能，処理内容，入出力タイミング等)，品質・性能に関する要求（信頼性，処理速度，移植性等)，運用に関する要求（情報システムの使い勝手，便利さ等)，プロジェクト管理に関する要求（システム開発の納期，要員，経済性等）等々いろいろな要求がある。これらの要件を正確にわかり易く表現するためには“要求仕様書”を作成する必要があり，以下に述べる諸技術項目はこの要求仕様書を記述する方法論および記述ツール等である。なお，これらの方法やツールは最近ではほとんどの場合“コンピュータによる支援”（いわゆるCASE）としての実行が可能である。

表1．ソフトウェアエンジニアリング (1/2)

区　分	概　要	技術・方法論・技法等
(I) ソフトウェア 要求分析	ソフトウェア利用者のニーズを明確にして，その仕様書を効率良く作成する。	構造化分析 (Structured Analysis) SADT (Structured Analysis and Design Technique) SREM (Software Requirements Engineering Technology) フロー図表 DFD (Data Flow Diagram)，状態遷移図，ペトリネット，デイシジョン・テーブル，特性要因図 ERD (Entity Relation Diagram) 方法論 KJ法，BS法（ブレーン・ストーミング) CASE (Computer Aided Software Engineering)
(II) ソフトウェア 設　計	ソフトウェア要求分析を元に，ソフトウェアの仕様書を効率良く，正確に作成する。	フロー図1 DFD (Data Flow Diagram) ERD (Entity Relation Diagram) 構造化設計 (Structured Design) Demarco，Myers データ中心設計 (Data Oriented Design) ジャクソン法 (JSD，JSP) ワーニエ法（データ構造図，プログラム構造図，真理値表) フロー図2 JISフローチャート，HIPO，HCP，PAD，NS，YAC モジュール化設計 CASE (Computer Aided Software Engineering) AD/Cycle，SDEM，DOA，SPRINGAM etc. オブジェクト指向設計 (OOD)（注1） デザインレビュー (DR) オープン・ダウンサイジング&ネットワーキング
(III) ソフトウェア 製　作	ソフトウェア仕様書を元にプログラム・アルゴリズムを展開し，プログラミングを効率良く行う。	ソフトウェア・アルゴリズミング 構造化プログラミング (Structured Programming) モジュール化 オブジェクト指向プログラミング (Object Oriented Programming)（注1） CASE (Computer Aided Software Engineering) 効率化 ソフト部品化，ソフト再利用 ソフトウェア・モデリング マルチメディア（注2） 第N世代言語

(注1）第3章参照のこと。
(注2）第4章参照のこと。

(1)構造化分析（Structured Analysis）

構造化分析とは，利用者からのソフトウェア要求に関する内容の記述に当って，概要から詳細へ階層的に表現することである。フローチャート的な図表示方法で表現する。

構造化分析では，ERD（Entity　Relation　Diagram）を使用したSoft　Tech社のSADT（Structured Analysis and Design Technique），TRW社のSREM（Software Requirements Engineering Technology）等が有名である。また，国内メーカでも最近多くのCASEが作られ，利用されている。（文献5）

(2)フロー図表

DFD（Data Flow Diagram），状態遷移図，ペトリネット，ディシジョンテーブル，特性要因図，ERD（Entity Relation Diagram）等多くのフロー図表が使用される。（文献6）

(3)方法論

利用者のソフトウェアに対する要求をまとめる方法としてはKJ法やBS法（ブレーンストーミング）などがあり，さらに利用者側・専門技術者側（ソフト会社等）で調査分析のための標準的なドキュメンテーションが用意されている。またDR（デザインレビュ）も実施される。

2．2　ソフトウェア設計

ソフトウェアの要求分析に基づいて，ソフトウェアの仕様を確定させるのがソフトウェア設計である。ソフトウェアエンジニアリングにおいてはソフトウェア設計が最も重要な技術的工学的ステップであるといえよう。

(1)構造化設計（Structured Design）

T. DemarcoやG.J. Myersの構造化設計は，ソフトウェアの“モジュール化”の概念を重視した設計方式であり，かつてはソフトウェアエンジニアリングの中心的技術であった。ソフトウェアは情報やデータを処理する技術であり，“処理方法”とか“処理手順”とかを考える一つの方式が“構造化”である。構造化はソフトウェアの全体を建築物のように考えて，土台，柱，壁，屋根，外装，内装等の機能別に処理を構造化する方式である。

(2)データ中心設計

構造化設計はソフトウェアの機能のモジュール化に着目するのに対して，データ中心設計はソフトウェアの処理対象となる“データ”に着目する設計方式である。

データ中心設計はジャクソン法（M.A. Jackson）やワーニエ法（J.D. Warnier）が有名である。諸データの洗い出しや関連性に着目して“データモデル”を作成し構造部・操作部等から成るモデルやデータの辞書(DD)を作成する。ジャクソン法もワーニエ法もそれぞれ独自の記法(フロー図表等）が用いられている。（文献6）

(3)フロー図表

ソフトウェア設計は，機械設計と異り“目に見えないもの”すなわち情報やデータを相手にしているため，一歩間違えるととんでもないミスを生み出すことになる。従って，ビジブル化のためのいろいろな工夫をされており，その一手段としてフロー図表がある。これには，JISフロー

チャート，HIPO，HCP，PAD，NS，YAC，ジャクソン図等がある。

(4)オブジェクト指向設計

オブジェクト指向は，世の中の事象をそのままソフトウェアシステムに移し換えるモデル化が実現しやすい方法という事で最近急速に重要視されるようになった。オブジェクト指向の技術は分析にも設計にもプログラミングにも広く応用される。なお，オブジェクト指向については，第3章で論じることにしたい。

(5)その他

最近，オープン・ダウンサイジング＆ネットワーキングということがよく言われる。コンピュータは大型機であれば何でもできるという時代から，小型コンピュータ（パソコン，ワークステーション等）を通信ネットワークで接続して統合的効果をねらう方針転換が計られている。

2.3 ソフトウェアの製作と試験およびプロジェクト管理

(1)ソフトウェアの製作

ソフトウェアの製作は具体的にはプログラミング作業を進めることであるが，そのソフトウェアの形を作る基となるのは“ソフトウェア・アルゴリズム”である。アルゴリズムとは算法と訳され，いわゆる“処理の手順”と考えられる。処理の手順と言っても人間が行う処理の手順とコンピュータが行う処理の手順ではかなりの違いがある。コンピュータの処理では“繰り返し処理”が頻繁に行われる。しかし，人間の処理ではできる限り単純に行うことになる。

ソフトウェア・アルゴリズムの代表としては構造化プログラミング（Structured Programming）(E.W. Dijkstra)が有名であり，モジュール化やGOTO削減および3つのフローパターン（順次処理，分岐処理，繰り返し処理）などの技法が採用されている。

ソフトウェアのアルゴリズムとして最近，オブジェクト指向プログラミングが重視されるようになったことは上記のソフトウェア設計でも述べたが，これについては第3章で論じる。

表2．ソフトウェアエンジニアリング (2/2)

区　分	概　要	技術・方法論・技法等
(IV) ソフトウェア 試　験	ソフトウェア製作結果を元にプログラムの実行テスト（信頼性テスト）を効率的に行う。	ソフトウェア品質管理（SWQC） ウォークスルー，インスペクション バグ管理曲線（ゴンペルツ曲線） テスト方法 　ブラックボックス・テスト（同値分割法，原因結果グラフ） 　ホワイトボックス・テスト（命令網羅度，条件分岐網羅度） 　ビッグバン・テスト テスト手順 　単位テスト，結合テスト，システムテスト 　トップダウン・テスト，ボトムアップ・テスト テスト消化率 信頼性指標 　プロダクト指標，プロセス指標，主指標，補助指標 CASE（Computer Aided Software Engineering）
(V) プロジェクト 管　理	(I)ソフトウェア要求分析～(IV)ソフトウェア試験完了，システム移行運用開始までのPLAN-DO-CHECK-ACTIONを行う。	スケジューリング・モデル 　ウォータフォール・モデル，プロトタイピング・モデル，スパイラル・モデル 　トップダウン・アプローチ，ボトムアップ・アプローチ プロジェクト計画 工程管理 　日程，負荷，進捗，納期 等 見積管理，外注管理，品質管理，原価管理，生産性評価，組織管理，設備管理（グループウェア等を含む），ドキュメント＆成果物管理 CASE（Computer Aided Software Engineering）

(2)ソフトウェアの試験

ソフトウェアは目に見えない物であるが故に，その品質が特に重視されている。注意しているにもかかわらずバグの伴うソフトウェアが氾濫して世間を混乱させている。ソフトウェア試験の効率化と信頼性向上のために，ソフトウェア品質管理(SWQC)，ウォークスルー，インスペクション，バグ管理曲線(ゴンペルツ曲線)，各種信頼性指標，テスト方法，テスト手順等が実施されている。

(3)プロジェクト管理

ソフトウェアエンジニアリングとプロジェクト管理の関係は一枚の紙の表（おもて）と裏との関係に例えられる。ソフトウェアエンジニアリングの効果を発揮するためにはプロジェクト管理の力を借りる必要がある。プロジェクト管理については別の機会に論じることとする。

3．オブジェクト指向ソフトウェア技術

オブジェクト指向ソフトウェア技術は，ソフトウェアエンジニアリングの最大の技術として脚光を浴びている。オブジェクトとは“物”とか“目的”とか訳され，オブジェクト指向技術を用いれば，我々の周囲の現実世界をソフトウェアの中にそのまま移し換えられる（写像される)。

また，オブジェクト指向技術によれば図2にあるような多くの技術によって，プログラミングの効率化や便利さを実現することができる。

オブジェクト指向とは世の中にある“物”をそのまゝソフトウェア化することである。物と言っても非常に広い意味を持っており，動物，人間，自動車，建物，日用生活品等“目に見える物”から，家族，会社等の組織体や社会現象物理現象等“目に見えない物”まで何でも対象となる。

オブジェクト指向ソフトウェア技術としては，図2にあるように，オブジェクト，クラス，メッセージ，メソッド，パブリック，プライベート，インスタンス……と多くの技法やコマンドがあり，カプセル化，データ抽象化などの“オブジェクト指向技術のねらい”が重要となる。

カプセル化とはソフトウェアの中に処理手続き（メソッドと

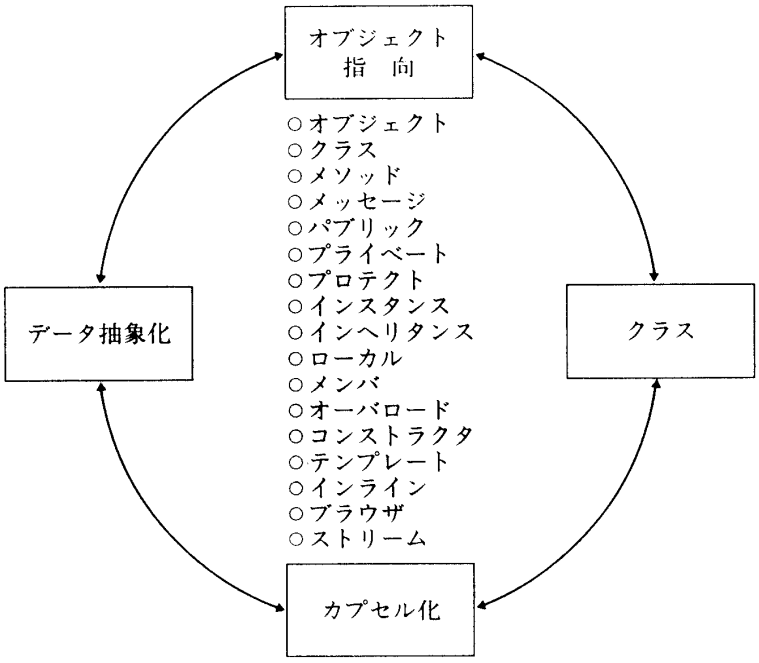

図2．オブジェクト指向ソフトウェア技術

呼ぶ）とデータを合せ持つ方式であり，データはパブリックとプライベートに分かれ，パブリックなデータのみが別のオブジェクト（クラス・オブジェクト，インスタンス・オブジェクトなど）とアクセス可能となる。

オブジェクト間のアクセスはメッセージを渡すことによって実現するので，メッセージを受けた側のオブジェクトの処理内容（すなわちメソッド）に関与することができない。すなわち，オブジェクトの独立性が保証され，ソフトウェアの混乱を防ぐことができる。

さらに，オブジェクト指向技術はソフトウェア・パラダイムのみならず，オブジェクト指向分析，オブジェクト指向設計，オブジェクト指向システム開発，オブジェクト指向データベースなど多くの情報システム化活動に関して“オブジェクト指向”の概念と用語が適用されている。

ここで，オブジェクト指向によるプログラム例を挙げておく。(表3)オブジェクトはクラスとして型表現され，さらには実体としてのオブジェクトとしてインスタンスが使用される。

クラスというデータ型はさらにサブクラスに分けることができ，上位の特性（ネイチャーやアトリビュートなど）は下位のクラスが継承（インヘリタンス）するのでプログラムの規模縮小化に役立っている。AI言語CESPはProlog＋オブジェクトであり，プログラム処理手続き（いわゆるメソッド)の部分はPrologによる論理式が使用される。(：で始まる文がメソッド)CESPのアトリビュートはC＋＋のパブリック，CESPのコンポーネントはC＋＋のプライベートに対応している。

オブジェクト指向プログラム言語のもう一つの例として，ウインドウ上で使用されるVisual BASICを挙げておく。この言語にはクラスの型概念はないが，これを補うイベント，メソッド，オブジェクト，プロパティと呼ばれるコマンド群がビジュアルな形で用意されており，プログラ

表3．オブジェクト指向言語によるプログラム例

AI言語CESPの例(1)（三菱電機㈱教育テキストより）

```
class family has
instance
: father (Obj, X, Y) :-father (X, Y) ;
: mother (Obj, X, Y) :-mother (X, Y) ;
local
  family (taro, hanako, [yoshio, yoshiko] ) ;
  family (kenichi, hiroko, [kenta] ) ;
  member (X, [X|Ch] ) ;
  member (X, [Z|Ch] ) :-member (X, Ch) ;
  father (X, Y) :-family (Y, _, Ch), member (X, Ch) ;
  mother (X, Y) :-family (_, Y, Ch), member (X, Ch) ;
  end.
```

AI言語CESPの例(2)（同左テキストより）

```
class  ダークホース  has
nature  馬
attribute
  name  :=  ダークホース
  sex   :=  雄
  place  :=  太郎君の牧場
  : get_name (Class, Class! name) ;
  : get_sex (Class, Class! sex) ;
  : get_place (Class, Class! place) ;
end.
```

C＋＋言語の例（TurboC＋＋マニュアルより，「オンライン辞書」宣言部分）

```
//単語定義のクラス
# include<string. h>
const int Maxmeans = 5 ;
class Definition
{
  char * word ;                    //定義される単語
  char * meanings [Maxmeans] ; //複数の意味あり
  int nmeanings ;
public :
  void put_word (char * ) ;
  char * get_word (char * s) {return strcpy (s, word) ;} ;
  void add_meaning (char * )
  char * get_meaning (int, char * ) ;
} ;
```

ムのプロトタイピング（試行）がやり易い便利なオブジェクト指向言語である。（図３）

４．マルチメディアのソフトウェア技術項目について

今，マルチメディアが政治・経済・技術を動かしている。政治分野ではアメリカのクリントン大統領・ゴア副大統領による“情報スーパーハイウェイ構想”であり，2015年までに全米に光ファイバー等による情報の超ハイウェイを実現するというもの。わが国も同じ構想を持っている。

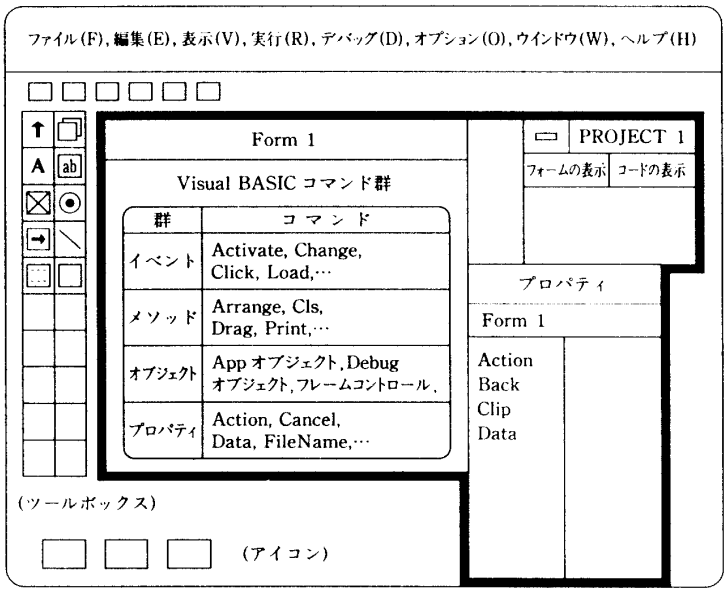

図３．ウインドウ上の Visual BASIC（太線枠内）

経済分野では，政治政策から来る財政の投融資による関連業界の活性化であり，情報通信システム業界，コンピュータ業界，ソフトウェア業界および経済関連業界の大きな期待がある。

マルチメディアのソフトウェア技術としては表４に挙げた項目が実現しつつある。

マルチメディアの実体としては，表４にある各種表現メディア（画像，映像，音声，文字…）のほかに，伝達メディア（新聞，雑誌，単行本などのペーパーメディア，テレビ，ラジオ，電話，FAX 等の有線・無線通信放送メディア等々いわゆるマスメディア）がある。そして，これらを統合的に処理する技術として，コンピュータによるハイパーメディア（超メディア）が生まれた。

マルチメディア技術は新しい情報化を推進し，社会生活を大きく変革するものと期待される。

表４．マルチメディアのソフトウェア技術項目

メディア	技　術　項　目（例）
（I） 画　像 （図形）	CG（２次元，３次元，アニメーション），画像圧縮，画像符号化，コンピュータ・アート，CAD，GUI，GKS（CG 標準化），画像データベース，カラー技術（スキャナ等），テクスチャ・マッピング，DTP，電子印刷，画像通信，広帯域 B ISDN，ATM，光技術，動画ホログラフィ，リモートセンシング，シミュレーション，ビデオゲーム，VR（バーチャルリアリティ），AI・ファジィ・ニューロシステム，地図情報システム（含，ナビゲータ）
（II） 映　像	人工現実感(RWC)，CT スキャナ，パターン認識，双方向 CATV，光ファイバー，テレビ会議，ハイビジョン，衛星放送，モルフ，ロボット，VR(バーチャル・リアリティ)，遠隔教育システム，電子印刷，ビデオメール，ビジュアル・マルチ・テレホン，AI・ファジィ・ニューロシステム
（III） 音　声 （音楽）	音声認識，音声生成，自動翻訳電話，情報家電，ADPCM(Adaptive Differential PCM)，圧縮技術，ヒューマンインタフェース，知的エージェント，カラオケ，コンピュータ・ミュージック，ディジタル・シンセサイザ(MIDI 規約)，知的対話，自然言語処理，AI・ファジィ・ニューロシステム
（IV） 文　字 （テキスト）	文字認識，ハイパーテキスト，DTP，電子印刷，CTS（コンピュータ組版），電子本（CR ROM 等），DB（各種媒体），AI 翻訳，AI・ファジィ・ニューロシステム
（V） 業界応用 等	コンピュータ・マッピング，医療，印刷，アート，AI，アパレル産業 CG（型紙，織物），建築・都市計画，通信，放送，機械 CAD，知能ロボット，娯楽，行政，教育，新聞・雑誌，図書館，出版等
備　　考	マルチメディアは各メディアの統合化利用を目指している。

あとがき

ソフトウェアエンジニアリングは，プログラミングを効率的に行うという初期のねらいから，オブジェクト指向技術やマルチメディア技術など新しいソフトウェア技術を採り込んで，大変革を遂げようとしている。ソフトウェアが変ると共にソフトウェアエンジニアリングが変るのは当然と言えよう。当論文では，要求分析，設計，製作，試験などのソフトウェアエンジニアリング項目に加えてオブジェクト指向とマルチメディアの技術について新側面に光を当てて論じた。

参考文献

1）牧野　勝：「システムズエンジニア育成のためのソフトウェア工学およびシステム設計工学に関する統合的研究について」，PP. 125－PP. 128，日本ソフトウェア科学会第11回大会，1994.
2）牧野　勝：「情報システム構築のための“システムズエンジニアリング”の研究」，PP. 221－PP. 228，福井工業大学研究紀要第24号（第一部），1994.
3）牧野　勝：「情報システム構築のための“システム設計論”の研究」，PP. 243－PP. 250，福井工業大学研究紀要第23号（第一部），1993.
4）佐藤　真・牧野　勝：「SEプロジェクト成功の鍵」，日科技連出版，1990.
5）原田　実　監修：「CASEのすべて」，オーム社，1991.
6）花田収税　編：「ソフトウェア仕様化と設計」，日科技連出版，1986.
7）Bertrand Meyer：「オブジェクト指向入門」，アスキー出版局，1990.
8）所・松岡・垂水：「オブジェクト指向コンピューティング」，岩波書店，1993.
9）Association for Computing Machinery：「COMMUNICATIONS OF THE ACM (Hypermedia), Vol 37, Number 2」, ACM, 1994.
10）本位田真一・山城明宏：「オブジェクト指向システム開発」，日経BP社，1993.
11）佐藤　真・牧野　勝：「実戦型SE育成の鍵」，日科技連出版，1992.

（平成6年10月24日受理）