読書案内　ソフトウェア工学

中所 武司

1 はじめに

電子計算機ソフトウェアはその利用形態の複雑化と多様化のために大規模化の一途をたどっている．そのため1970年頃から大規模ソフトウェアの信頼性と生産性向上のための技法の確立が急務とされた．そして，70年代にはソフトウェア開発現場で直面している問題を解決するための「ソフトウェア工学」という研究分野が確立し，今日に至っている．

そこで本文ではこれまでの主要な研究成果をその関連文献の紹介という形でまとめた．文献は入門書的な単行本や解説論文を主とするが，主要な研究論文も含める．

2 総　論

本文では便宜上ソフトウェアのライフサイクルを要求定義，設計，プログラミング，テスト，保守の5項目に分けて扱うが境界は必ずしも明確ではない．この全体を概観するための手頃な解説書としてB. W. Boehm[1]やM. V. Zelkowitz[2]の論文がある．特に前者のハードウェアとソフトウェアのコスト比のグラフやライフサイクルのモデル，後者のソフトウェア開発コストの各フェーズ比のグラフなどはよく引用される．単行本では宮本の力作[3]がある．これは70年代の主要な研究成果を集大成した内容になっており，事典代りになる．

3 要求定義

要求定義は顧客の要求を完全な仕様にまとめるものである．そのため，開発すべきシステムの概念的モデルを構築し，その稼動環境とのインターフェースを明確にする必要がある．その技法は多様であるが，概説書としては野木の[4]が手際良くまとめてある．この分野の先駆的システムであるPSL/PSA，SREM，SADTなどの詳細は国井らの[5]にまとめて紹介されている．70年代後半はこのような要求定義技法の研究の活発化と共にソフトウェア工学が広く普及し始めた．

Takeshi Chusho，日立製作所システム開発研究所．
1984年4月11日受付．

しかしながら，これらの技法では定義誤りがシステムテストの段階まで検出されないとか，開発未経験のシステムの要求定義が難しいなどの欠点がある．そこで80年代には要求定義段階で実行可能なモデルを構築し，仕様の正当性や完全性を早期に確認する技法が研究されている．この技法はoperational approachと呼ばれ，P. Zaveの[6]でわかり易く述べられている．

4 設　計

設計は要求定義の概念的モデルを計算機で実行可能なモデルに変換する作業であり，その時の注目点によって種々の技法がある．概説書としては前掲の[4]の他にG. D. Berglandの[7]が例が詳しく，わかり易い．代表的技法としては，データフローに注目した複合設計法[8]，データ構造に注目したJackson法[9]がある．なお，プログラミング方法論と関連の深いモジュール化技法については次節に譲る．

5 プログラミング

プログラミングは設計書の実行可能モデルを計算機入力可能なプログラミング言語で記述する過程である．ここでは70年代にプログラムの作成容易性よりも理解容易性が重要であるという視点から種々の方法論やそのための言語が開発された．その概要は鳥居らの[10]によくまとめられている．代表的な技法である構造化プログラミングや段階的詳細化法についてはE. W. Dijkstraの本[11]がある．最近のオブジェクト指向型プログラミングについては米澤の[12]に詳しい．

なお，従来の流れ図等のプログラム図式の高級化については二村の[13]によくまとめられている．また，プログラム開発には言語と処理系の他に支援系の充実が不可欠という考えから最近注目されている特定言語向きプログラミング環境(構造エディタを含む)については[14]が参考になろう．

6 テスト

プログラムのテストは前段階までの要求定義，設計，

プログラミングの検証を逆順に行うもので，モジュールテスト，結合テスト，システムテスト等がある．その各各でテスト技法は異なるが，一般的にプログラムの実行を伴うか否かで動的テストと静的テストに分けられる．前者はテストデータ作成方法が重要であり，機能仕様に基づく機能テストとプログラムの内部構造に基づく構造テストがある．この分野の全体的概説はG. J. Myersの[15]，動的テストに関しては[16]がある．

7 保　守

保守とは，システムの運用開始後に検出された誤りの修正や機能拡張のためのプログラム変更作業である．大規模ソフトウェアでは保守費用が開発費用を上回り，比率は増大している．しかしながら保守作業の多様性のために技術の進歩は最も遅れている．この分野の概説としては三浦らの[17]がある．

8 おわりに

以上，解説的な文献を幾つか紹介した．しかしながら，ソフトウェア工学の対象分野は広く，また人間的要因の影響を受け易いため，多種多様の技法が存在する．その中で真に役に立つ技術を見出すためには現状の問題認識が不可欠である．その意味では，大規模ソフトウェア開発プロジェクトへの参加が最良の教科書と言える．

更に重要な点として，現状の問題に対する対症療法的改良技法には限度があり，今後は理論的基礎を持った実用技術が望まれる．

謝辞　要求定義および設計技法について御教示頂いた日立製作所システム開発研究所の野木兼六主任研究員に感謝する．

(総論)

[1] Boehm, B. W.: Software Engineering, *IEEE Trans. Comput.*, C-25, No. 12(1976), pp. 1226-1241.

[2] Zelkowitz, M. V.: Perspectives on Software Engineering, *Comput. Surv.*, Vol. 10, No. 2(1978), pp. 197-216. 邦訳，日経エレクトロニクス，1979年1月22日号，pp. 146-170および1979年2月5日号，pp. 119-135.

[3] 宮本勲: ソフトウェアエンジニアリング，TBS出版会，1982.

(要求定義)

[4] 野木兼六: ライフサイクルI—構想から設計まで—，第6回ソフトウェア工学国際会議資料，情報処理学会，1982，pp. 1-25.

[5] 国井利泰(監修): ソフトウェア工学(要求仕様技術)，bit, Vol. 10, No. 10(1978).

[6] Zave, P.: The Operational Versus the Conventional Approach to Software Development, *Comm. ACM*, Vol. 27, No. 2(1984), pp. 104-118.

(設計)

[7] Bergland, G. D.: A Guided Tour of Program Design Methodologies, *Computer*, Vol. 14, No. 10 (1981), pp. 13-37.

[8] Myers, G. J.: *Composite／Structured Design*, Van Nostrand Reinhold, 1978. 邦訳(国友義久，伊藤武夫)，ソフトウェアの複合／構造化設計，近代科学社，1979.

[9] 峰尾欽二: プログラミング方法論(ジャクソン法)，情報処理，Vol. 23, No. 11(1982), pp. 1063-1073.

(プログラミング)

[10] 鳥居宏次，二木厚吉，真野芳久: プログラミング方法論の展望，情報処理，Vol. 20, No. 1(1979), pp. 22-43.

[11] Dijkstra, E. W.: Notes on Structured Programming, *Structured Programming*, Academic Press, 1972, pp. 1-82. 邦訳(野下浩平)，構造化プログラミング，サイエンス社，1975.

[12] 米澤明憲: オブジェクト指向型プログラミングについて，コンピュータソフトウェア，Vol. 1, No. 1 (1984), pp. 29-41.

[13] 二村良彦: 構造化プログラム図式，同上，pp. 64-77.

[14] 中所武司: プログラミング言語とその会話型支援環境，情報処理，Vol. 24, No. 9(1983), pp. 715-726.

(テスト)

[15] Myers, G. J.: *The Art of Software Testing*, John Wiley & Sons, 1979. 邦訳(長尾真，松尾正信)，ソフトウェア・テストの技法，近代科学社，1980.

[16] 中所武司，ソフトウェアのテスト技法，情報処理，Vol. 24, No. 7(1983), pp. 842-852.

(保守)

[17] 三浦一成，富田正治，尾関一平: ソフトウェア保守技術，bit, Vol. 14, No. 11(1982), pp. 131-206.