ソフトウェア工学 138−5
（2002．7．11）

スケールポイント法による組込みソフトウェアの開発工数見積り

吉澤 圭介

(株) 島津製作所　ソフトウェア開発センター
E-mail: ksk@shimadzu.co.jp

組込みソフトウェア開発における開発規模の尺度（サイズメトリクス）としてスケールポイント法を考案した。スケールポイントとは入出力を行う機器の数・画面数・通信コマンド数という開発規模と関係の深い３つの指標から求めるものである。開発規模と開発工数には相関があることから、過去のプロジェクトの実績から規模と工数との関係式を統計的に求め、この式を用いて見積り対象の開発工数を予測することができる。本稿ではスケールポイント法による開発工数の見積りの有効性を一般的に知られているＬＯＣと比較することにより示す。さらにこの手法を実際のプロジェクトの開発工数見積りおよび計画立案に適用した事例について報告する。

Method to Estimate Development Effort for Embedded Software from Scale Points

Keisuke Yoshizawa

Software Development Center, Shimadzu Corporation
E-mail: ksk@shimadzu.co.jp

We devised Scale Point Method as size metrics in embedded software development. We can count Scale Points from 3 indices that is well-related with the development size, - the number of modules to be inputted and outputted, the number of screens and the number of communication commands. As there is correlation between size metrics and the development effort, we can estimate the development effort from the relation expression between size and effort that is calculated statistically from past projects' data. In this paper, we show the validity of the effort estimation from Scale Point Method by comparing with famous LOC. Furthermore we repot the case that we applied this method to the effort estimation and the planning of an actual project.

1. はじめに

ソフトウェア開発のプロジェクトの計画立案に際して開発工数の見積りが重要であるのは言うまでもない。正しい見積りによる計画無くしては、開発プロジェクトを成功に導くことは出来ない。例えば事実（データ）に基づく見積りを行わずＫＫＤ（勘と経験と度胸）にのみ頼った計画の場合、作業項目の抽出漏れや割り込み作業等による遅れが生じるのは避けられない。何の量的見積りもせずにいきなり時間が出てき

ても、そのような時間が約束されることはあり得ないからである。また見積り値の信憑性に乏しいため、納期やコストの制約を受けやすく、計画自体に無理がある場合が多い。そして遅れの原因が計画の立て方に問題があったのか、仕事の進め方に問題があったのか、計画時には想定不能な外的要因による問題発生があったのか真の要因をつかむことができないため、計画を立て直すこともできない。その結果、納期やコストに深刻な影響を与えるのは言うまでもなく、最終的には品質低下を招き、納期遅延のあげく不具合多発や性能不足で発売時期を逸したり顧客に迷惑をかけることになる。

これに対し事実（データ）に基づく見積りを行うことにより、見積り値は信憑性の高いものとなり、約束できる計画を立案することができる。また開発担当者が見積り値に納得して作業を行うため、担当者の納期に対する意識や開発に対するモチベーションも高い。このように開発工数の見積りが開発プロジェクトを成功に導くポイントであると言える。

一般に開発工数は後述するＬＯＣやファンクションポイント法などを用いて見積られている。しかしこれらの方法は一長一短があり、開発現場に見積りを浸透させるには難しい面がある。

当社ではＬＯＣやファンクションポイント法に代わる方法として、Windows ソフトウェアでは当社独自のコントロール法[1]を使用し、開発工数や不具合数の見積りに活用している。また組込み（マイコン）ソフトウェアではスケールポイント法という当社独自の手法を使用し、開発工数の見積りに活用している。本稿ではこのスケールポイント法について紹介する。

2. スケールポイント法による見積りモデル

2.1. 開発工数の見積り方法

一般に開発すべきソフトウェアの規模と開発工数とは図１ に示すように比例関係にあり、規模が大きくなればなるほど開発工数も大きくなる。従って開発工数を見積るためには、まずソフトウェアの規模（サイズメトリクス）を計測する必要がある。そして過去のプロジェクトの実績から規模と工数との関係式を統計的に求め、この式を用いて見積り対象の開発工数を予測する(図１)。

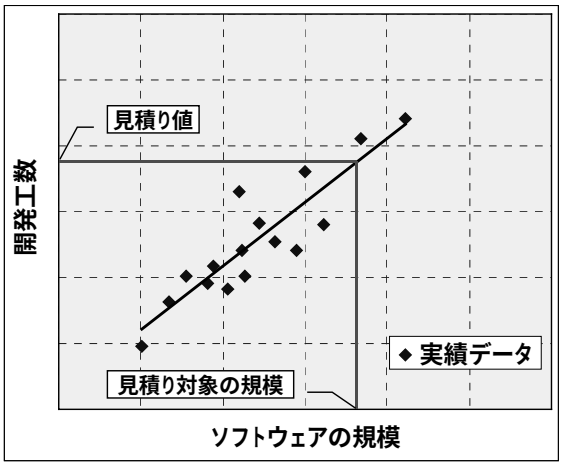

図1 ソフトウェアの規模と開発工数の関係

ソフトウェアのサイズメトリクスとしては、ＬＯＣ（Line Of Code、ソースコード行数）が一般的に知られている。しかしＬＯＣはソフトウェアが完成して初めて測れる数値であり、また開発言語に依存するという欠点もあることから、近年見積りのためのサイズメトリクスとしては適切でないという認識が高まっている。

機能仕様が固まった時点で算出可能なサイズメトリクスとして、ファンクションポイント法がある。ファンクションポイント法は、A.J.Albrecht が 1979 年に発表したサイズメトリクスであり、ソフトウェアが持つ機能に着目し、言語・実装に依存しない点に特徴がある。また IFPUG(International Function Point Users Group)の活動により、国際的業界スタンダードとして認められつつある。

ただしファンクションポイント法は、機能という漠然としたものを測定する手段であるため、

明確な定義がされているものではない。実際 IFPUG でもガイドラインや見解を示し判例を積み重ねることにより標準化を進めている。そのため、計測基準として客観性に欠けるとともに計測方法が複雑で分かりづらいという欠点を持つ。また元々銀行業務等のオンライン基幹システムなどを対象としており、判例も組込みソフトウェア にそぐわない、もしくは分かりづらいものである。

そこで当社ではファンクションポイントに代わるものとして、機能仕様が固まった時点で算出可能であり、機能の依存関係が深く計測が容易なスケールポイントを組込みソフトウェアのサイズメトリクスとして使用している。以下組込みソフトウェアにおいて、スケールポイントを計測することにより見積りを行う方法をスケールポイント法と呼ぶこととする。

2.2. スケールポイント

組込みソフトウェアの開発項目は大きく分けるとそのほとんどが以下の３つに分類される。

① 接続された入出力機器の制御
② 画面によるユーザと情報のやりとり
③ 通信による外部コンピュータとやりとり

これら３つに分類される項目の数は組込みソフトウェアの規模に関係すると容易に推定できる（図２ ）。

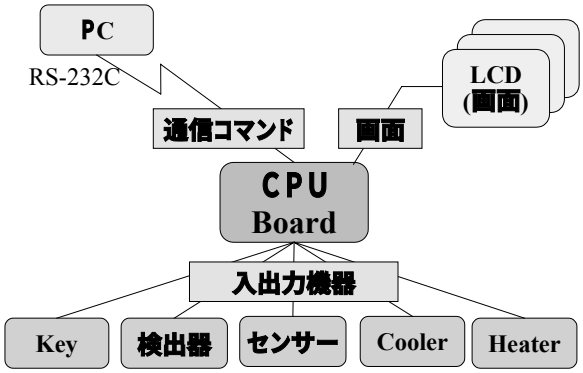

図2 組込みソフトの規模に関係するもの

そこでこれら３つに分類される作業項目の数を計測し、それに係数を乗じて足し合わせることで、組込みソフトウェアの規模（＝スケールポイント）を算出することを試みた。具体的には分類①の接続された入出力機器として入出力を行う 機器の個数、分類②の画面によるユーザと情報のやりとりとして画面の個数、そして分類③の通信による外部コンピュータとのやりとりとして通信コマンドの個数を単純・普通・複雑の３段階の複雑度に分けて数え、その複雑度に応じて係数を乗じ、それらを足し合わせることで算出する。

	単純	普通	複雑
入出力を行う機器	A_{11}	A_{12}	A_{13}
画面	A_{21}	A_{22}	A_{23}
通信コマンド	A_{31}	A_{32}	A_{33}

すなわち、先の表において、A_{mn} をプロジェクトにより決定されるそれぞれの指標の個数、k_{mn} を複雑度に応じた係数とすると、スケールポイントは次の式で算出される。

$$\text{スケールポイント} = \sum_{m,n} k_{mn} A_{mn}$$

以下､それぞれの指標について詳細に述べる。

[入出力機器数]

対象 CPU と接続していて入力や出力を行う機器の数。例えばキーユニット、検出器、センサー、クーラーおよびヒーターなどが挙げられる。また機器の複雑度の基準は以下の通りである。

単純：On/Off レベルのスイッチや LED など。
普通：一般の機器で On/Off レベルほど単純ではないもの。
複雑：ドライバに相当するものを新規開発するもの、複雑な制御が必要な機器など。

[画面数]

対象 CPU から出力される画面(帳票を含む)の数。複雑度のレベル分けは画面に含まれる内容の数による。表示されている項目を以下のよ

うに重みをつけて数える。

① 固定されている表示項目：0倍 (すなわちこの項目はカウントしない)
② 変化する可能性のある表示項目：1倍
③ 変更・設定できる項目やボタン・コンボボックスに相当するもの：2倍
④ 測定データや温度などリアルタイムに変化するもの：4倍

この重みをつけた表示項目の数を基に以下のようにレベル分けする。

単純：表示項目 6 個以下の画面。
普通：表示項目 7 個から 30 個までの画面。
複雑：表示項目 31 個以上の画面。

尚、帳票は画面として取り扱う。ただし帳票における表示項目は表示項目①の固定されているものか、②の変化する可能性のあるものしかないため、②の変化する可能性のある表示項目のみを数え、レベル分けを行う。

[通信コマンド数]

対象 CPU と外部との通信を行う際のコマンド数である。これに以下の基準でレベル分けを行う。

単純：1 往復以内で完了する通信コマンド
普通：2 往復以上必要な通信コマンド。例えば通信開始時のやりとり
複雑：やりとりが複雑な通信コマンド。例えば実試験におけるデータのやりとり

また、単にスルーするだけのコマンドなど同じ類のコマンドが続く場合はそれら全体で 1 つのコマンドとして数える。このとき同じ類のコマンドの数が 20 個以下であれば「単純」、21～100 個であれば「普通」、101 個以上であれば「複雑」に分類する。

2.3. 見積りモデル

過去実施した１６個の開発プロジェクトの実績データからスケールポイントと開発工数の相関図を作成すると図 3 に示すグラフが得られる。このグラフは両者の間の強い相関関係を示しているため、式(1)に示すようなモデル式を使用してスケールポイントと開発工数との関係を表すことができる。

$$y = ax + b \qquad (1)$$

ここで、ｙ：開発工数、ｘ：スケールポイント、ａ，ｂ：係数である。過去のプロジェクトの実績データを用いて回帰分析を行うことにより、各係数を求めることができる。こうして求められたモデル式は同種の開発プロジェクトの見積り式として使用することができる。

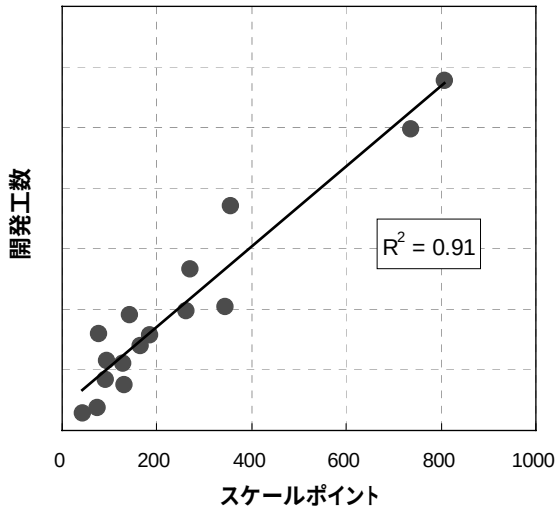

図3 開発工数とスケールポイントとの関係

3. ＬＯＣとの比較

開発工数の見積りに対するスケールポイント法の有効性を示すため、代表的な規模指標であるＬＯＣ（コード行数）と比較する。図 4 は図３で示した１６個の過去プロジェクトのうち、１４個のデータについて、ＬＯＣと開発工数との関係を示したグラフである。

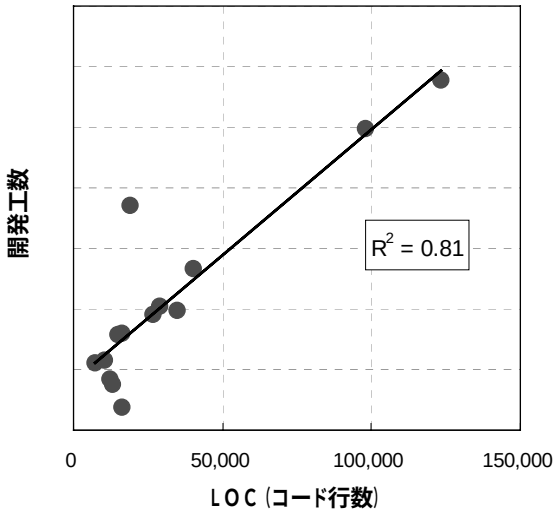

図4 開発工数とＬＯＣとの関係

このＬＯＣと開発工数のデータおよびスケールポイントと開発工数のデータについて相関分析を行い、両者の相関を比較する。この結果を表１に示す。

表１ 相関分析表

	相関係数	ｔ値	Ｐ値(両側)
ｽｹｰﾙﾎﾟｲﾝﾄ	0.956	-5.16	0.00012
ＬＯＣ	0.899	3.36	0.00513

t(0.975)=2.13 (自由度 15)

t(0.975)=2.16 (自由度 13)

自由度１５のｔ分布の危険率５％の両側検定の上側境界値 t(0.975)は 2.13 であり、|-5.16| より小さいため、開発工数とスケールポイントの間に相関は無いという帰無仮説は棄却域に入る。また自由度１３のｔ分布の危険率５％の両側検定の上側境界値 t(0.975)は 2.16 であり、3.36 より小さいため、開発工数とＬＯＣの間に相関は無いという帰無仮説も棄却域に入る。またＰ値<0.05 から両者は危険率５％で相関は有意であることが言える。そして相関係数はスケールポイント・ＬＯＣ共に１に近い。以上の結果よりスケールポイント・ＬＯＣ共に開発工数との相関は強く、強さは同等もしくはスケールポイントの方がやや強いため、両者共に開発工数の見積りに適していると言える。

一方実開発プロジェクトにおける見積りという観点で比較すると、ＬＯＣは開発を完了しないと計測できないのに対し、スケールポイントは開発仕様が固まった段階で算出することができる。さらに既存の製品に新たに機能を追加したり機能を削除するプロジェクトの場合、開発に関係したＬＯＣを測定することは困難であるのに対し、スケールポイントは追加・変更の仕様から算出することができる。このためスケールポイントはＬＯＣに比べて実開発プロジェクトにおける開発工数の見積りに適していると言える。

4. 見積りモデルで求めた値の調整

一般に開発工数見積りモデルでは、プロジェクトの規模に単純に係数を掛けるだけでなく、プロジェクトの属性に従って見積りを生成する。見積りに影響を与える属性には以下のようなものがあり、このほかにも様々な要因が考えられる。[2]

- プロジェクト要求の変化率
- 開発チームの経験
- 再利用可能な成果物の存在

現在１６個のプロジェクトデータが集まっており、プロジェクトの属性を考慮すると見積り値に影響があると思われるものが存在する。このためスケールポイント法では特に影響が強いと思われる以下の属性に従って見積りモデルで求めた値を調整している。

- プロジェクトの新規性
- 開発形態

プロジェクトの新規性とは過去のプロジェクトのバージョンアップなのか全く新規のプロジェクトなのかということである。ここで判定は新規性が高いか低いかの２段階で行う。また開発形態とは、スケールポイントで着目している画面と通信コマンドのある／なしの区別が全く

同じかということである。プロジェクトによっては画面や通信が全く無いものがあり、これらのある／なしの区別が異なると開発工数も異なるからである。

スケールポイント法における見積りモデルで求めた値の調整方法を以下に示す。

[手順1]

プロジェクトの新規性と開発形態が全く同一のプロジェクトを過去プロジェクトから抽出する。

[手順2]

抽出した過去プロジェクトの補正値を求める。補正値とは、過去プロジェクトのスケールポイントを見積りモデルに当てはめた値と実績値との差である。すなわちデータA～データCを抽出した過去プロジェクトデータとすると、補正値は以下の表2のように計算される。

表2 補正値の計算例

	データA	データB	データC
見積り値	1000	2000	2500
実績値	1050	2100	2440
補正値	+50	+100	-60

補正値の平均値：+30

[手順3]

補正値の平均値を見積りモデルで求めた値に加えた値を見積り値とする。また見積りモデルで求めた値に、補正値のうち最大の値を加えた値を上限値、最小の値を加えた値を下限値とする。すなわち表2に対し、見積り対象プロジェクトのスケールポイントを見積りモデルに当てはめた値を 1500 とすると、各値は以下のようになる。

見積り値：1500 ＋ 30 ＝1530
上限値 ：1500 ＋ 100 ＝1600
下限値 ：1500 － 60 ＝1440

以上の手順により見積り値および見積り値の上限値と下限値が求められる。特に見積り値の上限値と下限値を示すことにより、見積り値の幅が明確になり、次章で述べる見積り値の多方面からの検討が行いやすくなる。

5. 開発工数見積りおよび計画立案適用事例

スケールポイント法による開発工数の見積りは過去プロジェクトのデータから統計的に求められたいわゆるトップダウン見積りである。これに対し計画立案に欠かすことのできないＷＢＳを作成することによる見積りはボトムアップ見積りである。関係者全員が納得できる見積り値を基に約束できる計画を立案するためには、トップダウン・ボトムアップ両方向からの見積りが重要である。そこで当社ではＷＢＳの作成による見積りも行い、両者の差異を分析して見積り値を決定している。本章ではこれまで述べてきたスケールポイント法を、ＷＢＳおよび差異分析とともに、実際のプロジェクトの開発工数見積りおよび計画立案に適用した事例について報告する。

5.1. ＷＢＳ

ＷＢＳとは Working Breakdown Structure の略で、プロジェクトで実施する作業を細分化した結果である（図5）。

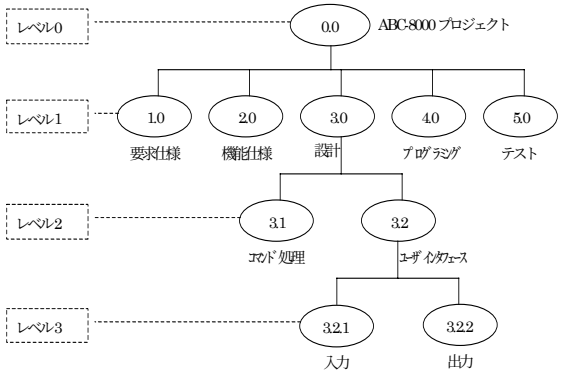

図5 WBS の例

ＷＢＳ完成後は、各作業項目の担当者を決定する。そして担当者により各作業項目の工数を見積り、その工数を積み上げることによりＷＢＳによる見積り値を決定する。

5.2. 差異分析

計画を立案する上で一番大切なことは約束できる計画を立てることであり、その為には開発関係者全員が見積り値に納得することが必要である。納得できない見積り値で計画を立てると開発者の計画を守るという意識が希薄になるからである。

見積り値に納得するためには、見積り結果をWBS による見積り値と比較し、その差の技術的背景を検討することにより、見積りの妥当性を確認する必要がある。

例えばWBSによる見積り値から35人月かかると思われるプロジェクトに対して、見積り式により 39.5 人月と算出された場合、まずその差の 4.5 人月がどこから生じているのかを検討する。良くある間違いが WBS の見積りにおける作業項目の抽出漏れである。テストの工数を考慮していなかったり、予期せぬ阻害要因による工数を含んでいない場合である。見積り式は過去の開発プロジェクトの平均値を算出すると言えるものですから、テスト工数はもちろんのこと、予期せぬ阻害要因による工数も（過去のプロジェクトでも予期せぬ阻害要因は発生したはずだから）含まれる。

逆に見積り式は平均値であるという性格から、注意しなければならない点もある。全く新しい装置を開発する場合や新しい技術を用いて開発する等、これまでのプロジェクトに無いリスクを見積り式は含まない。同様にこれまでに無い規模のプロジェクトにも見積り式は適用できない。見積り式には適用範囲があり、この範囲の中での関係を表したのが見積り式である。これを超える規模においても同じ関係が存在するかどうかを保証することはできないのである。

以上のように差異分析を行い見積りの妥当性を確認した上で関係者全員が納得できる最終的な見積り値を決める。

5.3. 適用事例

ある組込みソフトウェアのプロジェクトに対してスケールポイント法を使用して開発工数の見積りを行い、計画を立案した。

本プロジェクトに対してプロジェクトの新規性と開発形態が一致する過去プロジェクトは２つあった。このため本プロジェクトのスケールポイントを見積りモデルに当てはめた値に、抽出した２つの過去プロジェクトの補正値を使用して、見積り値および上限値・下限値を算出した。（図６）。

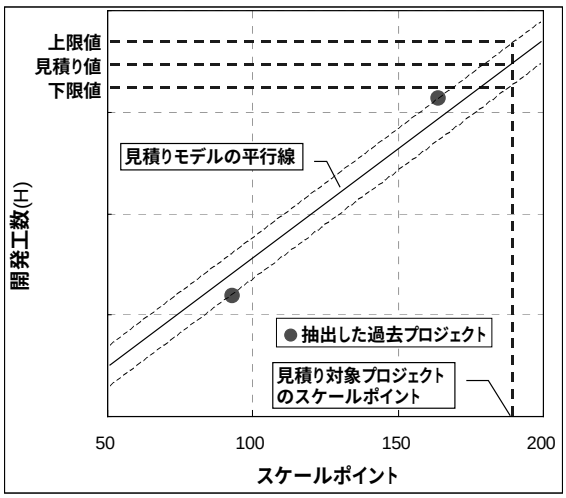

図6 過去の実績による開発工数の見積り

またスケールポイント法による見積りとは別に WBS を作成し、それぞれの作業項目の見積り値を積み上げることで開発工数の見積りを行った。そして両者の差異分析を行い、作業項目の抽出漏れや予期せぬ阻害要因について検討を行い、計画を立案した。

この結果、スケールポイント法による見積り値を１００％とすると、見積り結果および実績工数は以下のようになった。

	見積り値
スケールポイント法による見積り値	１００％
スケールポイント法による上限値	１０７％
スケールポイント法による下限値	９３％
WBS による見積り値	８５％
実績工数	９５％

これにより実績工数はスケールポイント法による見積り値の９５％という良い結果を得られた。

6. まとめ

本稿では組込みソフトウェアの規模を容易に測定する手法として当社独自のスケールポイント法を紹介し、代表的な規模指標であるＬＯＣと比較して開発工数見積りにおける有効性を示した。そして実際のプロジェクトにおいて開発工数の見積りおよび計画立案に活用している事例を示した。現在当社ではこのスケールポイント法を実際の組込みプロジェクトに適用し、実績を積み重ねるとともにノウハウを蓄積している。

スケールポイント法ではソフトウェアの規模を表す指標として組込みソフトウェアの開発項目の多くを占める入出力機器、画面、通信コマンドに着目した。この結果、ソフトウェアの規模指標としてはＬＯＣと同等であることが言え、開発工数の見積りについてはＬＯＣよりも適していることがわかった。このため組込みソフトウェア以外のソフトウェア、例えばクライアントサーバシステムにおいてもスケールポイントと同様の手法、すなわち開発項目の多くを占める内容に着目し、その数に適切な係数を掛けたものを利用して開発工数を見積ることができると思われる。

また現在スケールポイント法は開発工数の見積りのみに適用しているが、不具合数の見積りやプロジェクトの評価にも使用できるため、この方面でも活用していきたい。

参考文献

[1]「コントロール数による不具合数見積もりとその適用」橋口敏弘, 信学技報 Vol.99 No.683, p.9~16, 2000-3-14, 電子情報通信学会

[2]「ソフトウェア見積りのすべて」Capers Jones（著）富野壽（訳）, 共立出版, 2001

[3]「WBS の本質と現実的な活用方法」勝田祐輔, ユニシス技法 67 号, 2000

[4]「ファンクション・ポイント計測マニュアル リリース 4.1」JFPUG, 1999

[5]「見積もりの方法」真野俊樹, 誉田直美, 日科技連出版社, 1993

[6]「ソフトウェア機能性の計測」David Garmus･David Herron（著）阪田勇夫（訳）, トッパン, 1999

[7]「実践 ファンクションポイント法」児玉公信, 日本能力協会マネジメントセンター, 1999

[8]「回帰分析」久米均, 飯塚悦功, 岩波書店, 1987

[9]「ソフトウェア・メトリクス」Robert B. Grady（著）清水千博, 水野孝一（訳）, 日経 BP 社, 1990

[10]「ソフトウェアマネジメントモデル入門」山田茂, 高橋宗雄, 共立出版, 1993