ソフトウェア工学　137－3
（２００２．　５．　１０）

Eタイプソフトウェアのライフサイクル・コスティングと利益計画に関する一考察

河野充央
広島県立大学経営学部
mkohno@bus.hiroshima-pu.ac.jp

ソフトウェアには，その原価計算(原価管理)や価格決定に関して，不明確な部分が多く存在する。ソフトウェアという生産物が持つ属性は，工業生産品と比較した場合かなり独特ともいえる個性的要因に支配されている。例えば，工業生産品は，一応の完成品が市場(ユーザー)に提供されるが，ソフトウェアの場合には，多くのケースにおいて，完成品としてソフトウェアが市場に提供されるという意識をユーザーもサプライヤーも有していない。しかし，工業生産品とのこのような乖離を少しでも乗り越えることが，管理会計との接点を見出す上で重要な視点であると考える。

本稿では，進化という概念によって識別されるEタイプソフトウェアに関して，その属性から導き出されたLehman's Lawに焦点をあて，このような法則性が存在する場合，これがいかなる点で管理会計と結びつく可能性があるかということ，そしてソフトウェアの原価管理や利益計画の可能性がどのような形で考察可能であるかということに言及した。

A study of lifecycle costing and profit planning for E-type software

Michio Kohno
Hiroshima Prefectural University
School of Business
mkohno@bus.hiroshima-pu.ac.jp

An indefinite portion exists in software mostly about the cost accounting (cost management) and price determination. For the attribute which is inherent in software, the quite individual factor lies in comparison with industrial products. For example, although a market (user) is provided with temporary finished goods, as for the industrial products, in the case of software, in many cases, neither a user nor a supplier is considered that a market is provided with software as finished goods. However, I think that it is a viewpoint with important when managing the software process by management accounting easing difference with industrial products. This paper focused on Lehman's Law drawn from the attribute about E type software which continues evolving. And when such a law existed, it considered at what point this may be connected with management accounting, and whether the cost control and the profit planning of software are possible.

Ⅰ．ソフトウェアとソフトウェア原価計算の特質

本稿で議論の対象とするソフトウェアとは，いわゆるプログラム(特に，E タイプと分類されるもの)であるが，この知的生産物を，工業製品と対比した場合，いくつかの特徴を見出せる[櫻井 1992, pp.35-42]。

(1)ソフトウェアは工業製品のような有形物ではなく無形の情報である。
(2)インプットとアウトプットの関係，すなわち資源の投入とその成果との関係が比例的ではない。
(3)ソフトウェアの開発では工業製品の原価計算で重要な要素となる材料費がほとんど発生しない。
(4)ソフトウェアは，工業製品のように大量生産されるものではなく個別生産されるものである。

(5)研究開発活動や建設工事とのアナロジーが存在する。

では，工業製品に比して上記のような特徴を有するソフトウェアは，現行「原価計算基準」の適用にどの程度の対応が可能であろうか。この点に関して，次のような観点からの分析がある[櫻井1992, pp.46-50]。

(1)ソフトウェア原価に固有の特質

①原価財の投入量(原価)とソフトウェア製品の産出量との相関関係が工業生産物ほど明確ではない。

②ソフトウェアの仕掛品は不可視であり，ソフトウェアの仕掛品を原価との関係で管理することは難しい。

③ソフトウェア原価の大部分は人件費からなり，ハード原価計算にとって一般に妥当する直接材料費，直接労務費，製造間接費という原価分類は，ソフトウェア原価計算の実態にそぐわない。

すなわち，これらの特質から，ソフトウェア業に「原価計算基準」をそのまま適用することは困難であるとの結論が導かれる。

(2)ソフトウェア原価計算の目的

原価計算の目的は，各時代の要請にもとづき変遷することはありうるものの，ソフトウェア原価計算の目的は，一般的に，原価管理，予算編成および利益計画等の経営管理目的あるいは価格計算目的に重点がおかれている。

(3)総原価算定の必要性

原価管理および価格計算目的を達成するためには，製品原価の計算よりも総原価の算定が重要な役割を果たす。

Ⅱ．メンテナンスと原価計算

1.メンテナンスの種類

製品ライフサイクルを考察の対象としたとき，ソフトウェアほどその重要性が問われる製品はないだろう。環境や技術の進展に影響を受けつつも，大規模な組み込み型ソフトウェアシステムでは，そのライフサイクルにおいて，開発コストの 2～4 倍を要するメンテナンスコストが発生するといわれている[Sommerville, 1995, p.666]。

Sommerville は，メンテナンスとして次の 3 種類をあげている[Sommerville, 1995, p.660]。

①適応化保守(adaptive maintenance)，②完全化保守(perfective maintenance)，③修正的保守(corrective maintenance)

Lehman は，プログラムの進化(evolution)を分析する際に，いわゆるメンテナンスに対して，①メジャー・メインストリーム・リリース(major mainstream releases)，②マイナー・メインストリーム・リリース(minor mainstream releases，③エラー・コレクション・リリース(error correction releases)という名称をあげている[Lehman,1997,p.6]。ここで詳述することはできないものの，これらは，Sommerville の提示した上記 3 種類と，その定義に関し大きく異なる見解ではないと思える。

ソフトウェアのメンテナンスに対して，フィードバックコントロールにもとづく進化の法則が存在するということを総合的なメトリック研究から導いた Lehman の所論を考慮すると，ソフトウェア原価計算研究の中心も必然的に，システムの開発導入後(いわゆる，進化のプロセス)に向けられるべきであろう。そこでメンテナンスという概念に対する考察も十分に行われる必要がある。

2.Lehman によるメンテナンスの解釈と Lehman's Law

メンテナンスに関して Lehman は次のような考えを示している[Lehman,2002,p.4]。

プログラムもしくはソフトウェアシステムのバージョンアップ、リリース、アップグレードといったことには、欠陥の改正や削除といった変化が含まれる。あるいは代替的操作環境の準備、および、システム機能や性能や品質などの改善や拡張を実施することの双方もしくは一方が含まれる。それらは、一般に、リリースプロセスという名前でユーザーに提供される。

基本的には、これらの変化はすべて、ステークホルダーにとっては、ある意味で、プログラムに加えられる改善であると解釈される。さもなければ、そういったことへの着手は行われない。そして、このプロ

セスは、広い意味でプログラムのメンテナンスであると考えられる。

一般に、メンテナンスという言葉は、ソフトウェアのコンテクストにおいては、何か不適当なこと(実は、これは誤った解釈である)のように認識されてきた。他のエリアで使用されている言葉に従うなら、メンテナンスとは、一般に、改正するために、着たり剥ぎ取ったりする活動である。あるいは、いわゆる人為的結果として開発されてきたものの中の悪い部分を正す活動である。メンテナンスの本来の目的は、オリジナルの状態、もとの清純な状態にできる限り近づくように対象物を戻すことを意味する。

しかしながら、ソフトウェアにはその目的は当てはまらないのである。ソフトウェアの質はそれ自体で物理的に悪化するというものではない。ユーザーの感じるソフトウェアの品質の悪化というのは、例えば、もともとソフトウェアを購入したときの目的に対して現在の目的が変化したところから生じるものである。あるいは、操作ドメインの特質の変化とか、技術の進展とか、競合製品の出現といったことから生じるのである。このような形での悪化は、論理的仮説であることが多い。論理的仮説というのは、外部の状況次第では仮説のまま終わることが多い。しかしながら、ソフトウェアの世界では、これは、暗黙的にも開示的にも、多分に仮説では終わらないのである。

したがって、こういったコンテクストにおいて、メンテナンスという言葉は、もっと適切に、ソフトウェアの本質を反映した形で理解されなければならない。そして、ステークホルダーやユーザーがきちんとした判断で行動できるように把握されなければならない。 要するに、ソフトウェアは、その仮説を本質的に正当なものとして保持しつつ進化していくのである。進化がおこなわれ、その時々の現実世界とマッチしていくのである。

進化は，ソフトウェア環境のあらゆる側面で観察されるが，Eタイププログラムは，次に示す8つの法則に支配された成長プロセスを辿る[Lehman,1997,p.3]。

① 継続的変化(Continuous Change)…Eタイプ・プログラムは、ユーザーの満足が徐々に喪失していかない限り、絶えず環境に適合していくことが必要である。

② 複雑性増大(Increasing Complexity)…現実世界で生じる問題のレベルが、維持されるかもしくは減少しない限り、Eタイプ・プログラムは進化につれて、複雑性を増大させる。

③ 自己規制(Self Regulation)…Eタイプ・プログラムの進化のプロセスは、自己規制的に推移する。

④ 組織的安定性保存(Conservation of Organizational Stability)…Eタイプ・プログラムの進化のプロセスにおいて、有効活動率（増加的努力;incremental effort）は、グローバルな観点から、平均的な値で推移する。

⑤ 親和性保存(Conservation of Familiarity)…進化するEタイプ・プログラムのライフタイムにおいて、連続したリリースの平均的内容は統計的に不変である。

⑥ 継続的成長(Continuing Growth)…Eタイプ・プログラムの機能的内容は、そのライフタイムの間、ユーザーの満足を維持するために、絶えず増加していかなければならない。

⑦ 品質後退(Declining Quality)…Eタイプ・プログラムは、厳しくメンテナンスを行い、変化する操作環境に適応していかなければ品質の後退を招く。

⑧ フィードバック・システム(Feedback System)…Eタイプのプログラミング・プロセスは、マルチレベル、多重ループ、フィードバック・システムを構成する。何らかの合理的根拠にもとづき、重要な改善をするには、そのように対処することが求められる。

3.メンテナンスをサポートする原価計算の例

「保守の原価に製品原価性を付与すべきか否かは，税務上の処理との関係から問題の多いところであるが，製品原価性を持たせるか否かに関わりなく，原価計算は重要な意味を持つ。それゆえ，保守のためにも適切な原価計算制度を適用して，原価管理や採算管理に役立てていく必要があろう」[櫻井，1992，p.58]。

メンテナンスに，原価計算を適用している例として，発生単位別に集計するユーザー別原価計算がある。ユーザー別の原価集計のために，次に示す表のようなユーザー別収支計算表が作表される。

年　月分

付番	ユーザー別 項目			センター計	Aユーザー	Bユーザー	Cユーザー		委託業務費
1	売　上　高								
2	保守原価	直接材料費							
3		部品費							
4		直接旅費							
5		直接運賃							
6		外注加工費							
7		委託業務費							
8	計								
9	保守原価率								
10	貢献利益								
11	貢献利益率								
12	本社共通費								
13	総原価								
14	売上利益								
15	売上利益率								

図表-1　ユーザー別収支計算表
(出典：櫻井，1992，p.59)

Ⅲ．Eタイプソフトウェアに対する利益計画の方法

1.ソフトウェアの進化に対するLehmanの解釈

Lehmanは，プログラムの進化に関する考え方をかなり柔軟に展開しているように考えられる[Lehman,2002,pp.5-13]。

たとえば，システム開発のプロセスで繰り返される，フィードバックに基づく諸経験が，開発プロセス自体の進化を生むし，新たな開発ツールの開発も進化の源泉でといえるようである。また，メンテナンスという概念は，元に戻すということであるが，進化をベースとするソフトウェアプロセスにはこのようなメンテナンスの概念はあてはまらない。また，進化の概念には，スパイラル型やインクリメンタル型で開発されるシステムのプロセスも包含される可能性がある。

さらには，Eタイプソフトウェアの進化は，その対象となるアプリケーションの進化を誘発するし，プロセスモデル自体も進化する。

したがって，Eタイプソフトウェアの範疇には，アプリケーションの要求を常に受け止めながら，変貌していくすべてのソフトウェアとして，次のようなものを広く考察の対象としているように思われる。

①メインフレームのOSおよびアプリケーションソフト(受注型ソフト)

②パソコン用のOSやアプリケーションソフト(見込み生産型ソフト)

また，進化のプロセスのモデル化にあたっても，LST(Lehman,Stenning and Turski)パラダイム(Inverse Square Model)のみならず，線型モデル等複数の対応モデルを考慮することを示唆している。

また，開発と進化のプロセスとの間には，次のような関係が想定できる。すなわち，ライフサイクルコスティングを考慮の対象とした場合，開発に十分な投資を下したものは，進化のプロセスでの負担が軽減し，逆に開発に手をかけなかった場合には，進化のプロセスで多大なコストが発生すると予想できる。つまり，開発コストと進化のプロセスで発生するコストとの間には，トレードオフの関係が生じるのではなかろうか。

品質原価計算(quality costing)における予防原価と失敗原価との関係のようなものである。

2.見込み生産型Eタイプソフトウェアの利益計画に関しての私案

(1)原価企画(target costing)の定義［門田,1994,p.8］

原価企画とは，新製品開発段階における全社的利益管理を意味し，顧客の要求を満たす品質をもった製品を企画し，中長期利益計画で必要とされる目標利益を所与の市場環境条件のなかで達成するために，新製品の目標原価（目標投資額を含む）を決定し，要求品質・納期を満たしながら，目標原価を製品の設計上で達成するようにとりはからう全社的活動である．

原価企画システムとは，このような原価企画プロセスをサポートする全社的マネジメントシステムである．原価企画は，企業の利益管理のプロセスと一体化しており，①中長期の総合利益計画から始まる．そこでは，個別構造計画として，個別新製品開発計画，販売計画，設備投資計画，要員計画などとともに，各種製品別の目標利益を設定する．この目標利益は，原価企画活動で達成すべき目標になる．

ついで原価企画は，次の2つのプロセスをもつことになる．②顧客のニーズを満たす商品を具体的に企画し，その新商品の目標利益から目標原価を導くプロセスと，③その目標原価を設計部でVE的に達成することを，実際の原価見積りによって確認していくプロセスとである．

(2)自動車メーカーの個別新製品ライフサイクル計画と中長期利益計画

上記原価企画は，自動車メーカー等で活用される原価低減の技法である。もともと工業製品とソフトウェアとは全く属性が異なることから，生産手法のみならず生産現場も全く異なる。しかしながら，いくつかの共通点を考慮すれば，Eタイプソフトウェアの特性に原価企画の手法を結びつけることも可能であると考える。そこで，一例として，見込み生産型Eタイプソフトウェアを対象として，その利益計画に対し若干の考察を加えたいと思う。

自動車の原価企画では，その第1ステップは個別新製品ライフサイクル計画である。その内容は，次の通りである［門田,1994,p.11］。

個別新製品ライフサイクル計画とは，車種別の担当プロダクトマネジャーが各車種のライフサイクルを計画（開発開始年度，量産開始年度および次のモデルチェンジ年度に関する計画設定）し，その車種に関する設計・試作・生産準備の人的能力，新規設備費・試作費・開発費等の所要費用を見積り，車種別のモデルライフ（量産開始から次のモデルチェンジにいたるまでの期間）に関する利益計画案を樹立するプロセスのことである。

そして，第2ステップは中長期利益計画と総合新製品計画である［門田,1994,p.12］。

中長期利益計画とは，向こう3年ないし5年にわたる全社的な毎期の利益計画と資金計画をたてることである。この3~5年利益計画をたてるプロセスのなかで，個々の製品ラインのモデルライフにわたる利益目標も決定され，また経営の基本構造を決めるようなさまざまな個別構造計画が利益計画と資金計画の観点から相互に調整される。個別構造計画としては，車種別の新製品開発計画や販売計画，設備投資計画，人員計画，資金調達計画などがある。

総合新製品計画とは，車種（車系）別の「個別新製品ライフサイクル計画」が長期利益計画のプロセスのなかで総合調整されたものである。それは各車種のモデルライフを年次に割り付けて，それぞれの車のモデルチェンジやマイナーチェンジなどを全車種について総合したものである。

(3) 見込み生産型Eタイプソフトウェアの中長期利益計画

①ISMとリリース工数予測モデル

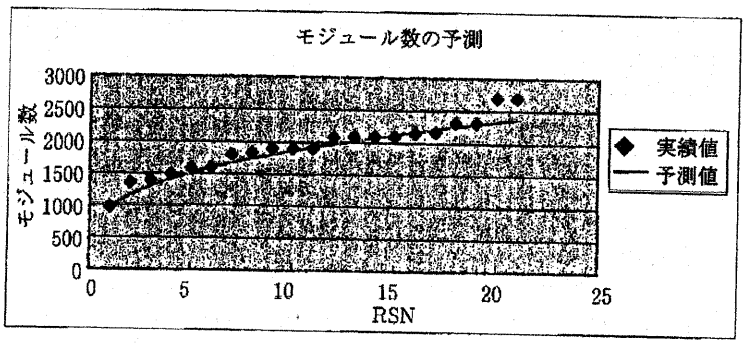
上記内容から判断できることは，このプロセスが，リリースを繰り返しながら(いわゆるバージョンアップをしながら)市場に提供され続ける見込み生産型アプリケーションソフトとかなり符合する部分があるのではないかということである。特に，パソコン用のパッケージソフトを念頭に置けば，モデルチェンジをしながら市場でシェアを維持していく，自動車産業のあり方と同様の姿をそこに見出すことができる。

Lehman's Lawを前提に，Inverse Square Model(ISM)[Turski,1996,p.600]を使い，図表-2のデータを

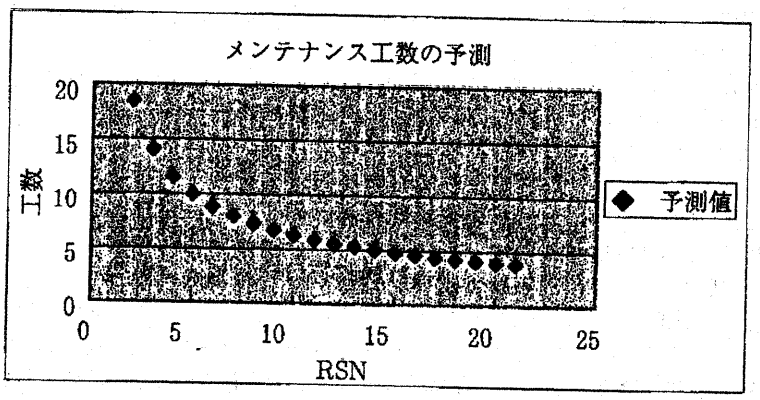
もとに，モジュール数の予測し，さらにこれをベースにリリース工数の予測値を設定したものが，図表-3および図表-4である。これは，メンテナンス(リリース)工数予測モデルである。また，図表-5はこの分析に利用したExcelのワークシートである。

RSN	Size in Modules	Release ID	RSN	Size in Modules	Release ID
1	977	1.0	12	2087	5.0A
2	1344	2.0A	13	2091	5.0B
3	1390	2.0B	14	2095	5.0C
4	1492	2.0C	15	2101	5.0D
5	1581	2.0D	16	2151	5.0E
6	1595	2.0E	17	2167	5.0F
7	1800	3.0A	18	2312	6.0A
8	1832	3.0B	19	2315	6.0B
9	1897	4.0A	20	2696	7.0A
10	1897	4.0B	21	2699	7.8B
11	1902	4.0C			

図表-2　The FW Data Set



図表-3



図表-4

RSN	ME	Theoretica	CT	Modules	theoretica	1/s^2	Sigma	E	average	E =	206046744.3
	1			977	977	1.048E-06				S_1 =	977
	2	18.472105	0.184721	1344	1157.472	5.536E-07	1.601E-06	229196804	229196804.3	SDT=	100
	3	14.01765	0.140176	1390	1294.425	5.176E-07	2.119E-06	194920226	212058515.1		
	4	11.550604	0.115506	1492	1407.274	4.492E-07	2.568E-06	200542133	208219721.2		
	5	9.9464721	0.099465	1581	1504.451	4.001E-07	2.968E-06	203496559	207038930.6		
	6	8.8047894	0.088048	1595	1590.474	3.931E-07	3.361E-06	183863614	202403867.4		
	7	7.9432684	0.079433	1800	1668.08	3.086E-07	3.67E-06	224261129	206046744.3		
	8	7.2658525	0.072659	1832	1739.067	2.98E-07	3.968E-06	215485579	207395149.2		
	9	6.7166674	0.067167	1897	1804.689	2.779E-07	4.246E-06	216691457	208557187.7		
	10	6.2607968	0.062608	1897	1865.857	2.779E-07	4.524E-06	203379944	207981938.4		
	11	5.875197	0.058752	1902	1923.258	2.764E-07	4.8E-06	192709165	206454661.1		
	12	5.5439916	0.05544	2087	1977.423	2.296E-07	5.03E-06	220694793	207749218.6		
	13	5.2558539	0.052559	2091	2028.772	2.287E-07	5.258E-06	211856188	208091466.1		
	14	5.0024672	0.050025	2095	2077.646	2.278E-07	5.486E-06	203786826	207760339.9		
	15	4.777575	0.047776	2101	2124.323	2.265E-07	5.713E-06	196755751	206974297.8		
	16	4.5763723	0.045764	2151	2169.034	2.161E-07	5.929E-06	198016508	206377111.8		
	17	4.3951024	0.043951	2167	2211.975	2.13E-07	6.142E-06	193755825	205588281.3		
	18	4.2307819	0.042308	2312	2253.309	1.871E-07	6.329E-06	210939478	205903057.6		
	19	4.0810091	0.04081	2315	2293.181	1.866E-07	6.515E-06	205358863	205872824.6		
	20	3.9438266	0.039438	2696	2331.712	1.376E-07	6.653E-06	258379474	208636332.5		
	21	3.817622	0.038176	2699	2369.01	1.373E-07	6.79E-06	253597748	210884403.2		

図表-5

Turski は，6 回のリリースにおいて変動したモジュール数，したがって RSN(Release Sequence Number)1からRSN7までの各RSNにおけるシステムのモジュール数からパラーメータEを決定すれば，ISM 関数によって，残りの全リリースにおけるモジュール数を予測できることを示した[Turski,1996,p.600]。ちなみに誤差は10%未満である。

ここで例示したのは英国FW 社の財務処理システムであるが，これは100 以上のコンピュータにインストールされ現在稼動中のソフトウェアであり，数年にわたってリリースが繰り返されてきた[Lehman,1997,p.5]。

E タイプソフトウェアのフィードバックドライブ現象が，商業的な政策を超えて存在する普遍的なものであるとの解釈には，無条件に賛同しかねる部分が多々あるものと思われるが，これを一つの指針，一つの可能性として提示するならば，マイナーおよびメジャーのモデルチェンジに関わる開発費の予算を算定する際の根拠としてこれを活用することには，一定の評価が与えられるのではないかと思われる。もちろん，大前提としてLehman's Law が肯定されていなければならない。

②原価企画の方法

[ステップ1]

ある汎用パッケージソフトウェアを新商品として開発する際に，次の点をシステム分析および設計段階で検討する。

- 商品の開発規模(総モジュール数)を策定
- 6回のリリース計画を立て，それぞれにおけるモジュール数を算定

[ステップ2]

- ISMにより，商品のライフサイクルを把握
- 長期的なリリースの目標工数を算定
- 予定賃率×目標工数により目標固定費を算定

[ステップ3]

- 総原価にもとづく中長期総合利益計画の設定

[ステップ4]

- 販売計画，要員計画，個別利益計画
- 顧客のニーズにあった商品の企画

[ステップ5]

- 製品開発と原価企画実績評価

[参考文献]

1.櫻井通晴他著『ソフトウェア原価計算』白桃書房，1992年。

2.門田安弘『価格競争力をつける原価企画と原価改善の技法』東洋経済，1994年。

3.Sommerville, I., Software Engineering (5th edn.), Addison-Wesley Publishing Company Inc., 1995.

4.Lehman,M.M., Characteristics of S- ,E- and P-type Application and Software, Prepr., Second FEAST Workshop, Imp. Col., 24-25 Oct., 1994.

5.Lehman,M.M. and Ramil, J.F., Evolution in Software and Related Areas (This paper represents a broadening and revision of an article to appear in the Encyclopedia of Software Engineering, 2nd edition, edited by J Marciniak), 2002.

6.Lehman, M.M., Program Life Cycles and Law of Software Evolution, Proc. IEEE Spec. Iss. on Softw. Eng., Sept. 1980.

7.Lehman,M.M., Ramil, J.F. and Wernick, P.D., Metrics of Software Evolution — The Nineties View, Paper presented at the Fourth International Symposium on Software Metrics, Metrics'97, Albuquerque, New Mexico, Nov. 5-7th, 1997.

8.Turski, W., Reference Model for Smooth Growth of Software Systems, IEEE Transactions on Software Engineering, vol.22, N.8., 1996.