ソフトウェア工学 77−24
（１９９１．２．８）

工数予測モデルに基づく定量的工程管理の試み

坪井信男　真野俊樹　誉田直美　浦本啓司　小南はるみ　田中伸子　鈴置奈保美
日本電気株式会社　基本ソフトウェア開発本部

ＣＯＣＯＭＯモデルを参考に開発したソフトウェア開発工数予測モデルと、本モデルを用いた工程管理方式について述べる。過去の実績データの分析により、開発工数は開発規模に指数関数的に比例し、流用率、品質、開発グループのスキル（バグ摘出能力）、開発環境（ソフトウェア体系）といった要因により補正されることが判明し、我々はこの結果に基づいてモデル式を作成した。工程管理については、本モデルによる予測工数に基づき、開発に使用した工数が品質確保といった点から見て適正であるか、納期が守れるかといった２つの観点から、工数実績を評価する指標を定義し、これを用いて開発中に実績評価とフィードバックを行う手順を考案した。

Quantitative Process Management based on Cost Model

Nobuo Tsuboi　Toshiki Mano　Naomi Honda
Keishi Uramoto　Harumi Kominami　Nobuko Tanaka　Nahomi Suzuoki

Basic Software Development Division, NEC Corporation
14-22, Shibaura 4-Chome, Minato-ku, Tokyo, 108 Japan

This report introduces our cost model based on Boehm's COCOMO and quantitative process management system using this model. Our cost model was developed based on the analysis of our past data and was characterized by various development factors of the equation. Using this model, we examined process management system which used two quantitative measurements to evaluate actual data of manpower during development process from two points of view, that were whether actual manpower was adequate to expected quality and would be able to keep the planned delivery date.

1．はじめに

近年、ソフトウェアの重要性に対する認識が深まり、市場の高品質要求の声はますます高まっている。こうした状況下で、市場ニーズに応える高品質のソフトウェアを、タイムリーかつ効率良く開発しリリースするためには、適切な定量的根拠に基づく工程管理が必要不可欠な要素の一つとなってきている。

工程管理は、大きく分けて工数予測と実績評価・フィードバックの２つの技術で成り立っていると考えられる。これらの技術については、現在までにさまざまな試みがなされているものの、決定的な方式は確立されてはおらずまだまだ研究の余地があるといえる。我々の所属する部門においても、工程管理を行ってはいるものの、その方式はより一層の改善が必要とされていた。

本稿では、今回我々が開発した開発工数予測モデル[1]と、それに基づく工程管理方式について述べる。モデルに関しては Boehmの提案したCOCOMOモデル（COnstructive COst Model)[2]の考え方を参考として、過去の実績データを統計的手法により分析した結果から工数に影響を与える要因を抽出し、モデル式を作成した。また、工程管理は、納期に対する状況評価だけではなく、使用した工数が品質確保といった観点からみて適正であるかといった点からも評価を行う方式を考案した。

2．開発工数予測モデル

以下に我々が開発した開発工数予測モデルについて、モデルの基本形の検討、データ分析とモデル要因の決定、モデル式の作成・検証といった開発過程に従って説明する。

2．1　モデルの基本形

（1）モデル形式

COCOMOモデルでは、ソフトウェアの開発工数は、主に開発規模に指数関数的に比例し、これを種々の開発要因が補正するものとして定義しており、モデル式は以下の形式を取っている。

$$C = a \cdot \prod_{i=1}^{n} \alpha_i \cdot S^b$$

C：開発工数［人H］
S：開発規模［KL］
α_i：開発要因（コスト要因）
a，b：定数

ここで、当部門における開発規模（対数）と開発工数（対数）の関係（図2.1）を見てみると強い相関があることがわかった。すなわち当部門の開発においても、所要工数は主に開発規模に指数関数的に比例している。

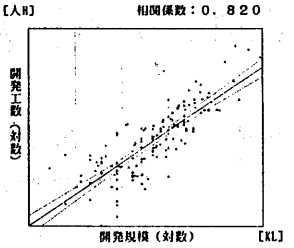

図2．1　開発規模と開発工数の相関

この事実により、我々はモデル開発に当たって、先に示したCOCOMOモデルの考え方を用いることとし、モデル式の形式はCOCOMOのそれに準ずることとした。

（2）上工程／テスト工程別モデルの設定

当部門では、開発作業を大きく上工程（設計～製造工程）とテスト工程の２つに分けている。また、上・テスト工程ともに、さらに複数の工程に分けられている。このような開発形態を取っていることから、きめ細かい分析とフォローを行うためには、モデルは工程別に設定することが望ましいと思われる。

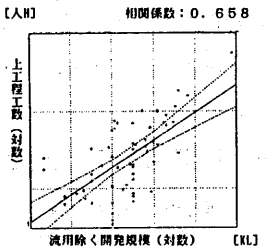

図2．2　流用除く規模と上工程工数の相関

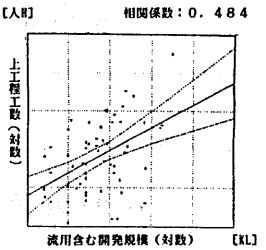

図2．3　流用含む規模と上工程工数の相関

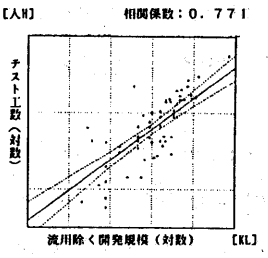

図2．4　流用除く規模とテスト工数の相関

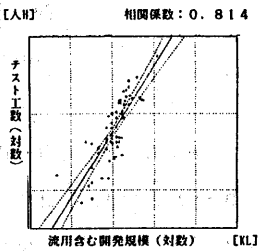

図2．5　流用含む規模とテスト工数の相関

そこで、今回は上工程とテスト工程の２つに分けてモデルを開発することとした。特に、上工程とテスト工程は、以下に示すように流用部分にかかる工数の影響の違いが大

きいため、このように分けることは有効である。

開発規模と開発工数の関係を、規模に流用部分を含めた場合と除いた場合に分けて見た場合、図2.2-2.5 に示すように、上工程では流用を除く規模が、また、テスト工程では流用を含む規模が、それぞれ他方に比べて相関の高いことがわかる。

この理由として、上工程は、設計および製造という作業の性格から、新規／改造部分にかける工数が流用部分にかけるそれに比べて大きいのに対し、テスト工程は、流用部分も新規／改造部分同様にテストを実施するので、各々にかける工数にさほど差異が無いためと考えられる。

2．2　開発要因

開発工数に影響を及ぼす要因として、大きく以下の３つのものが考えられる。

①開発するプログラムに関する要因

②開発する人間に関する要因

③開発環境に関する要因

そこで、各々の特徴を表すと思われる指標をいくつか設定し、過去の実績データを用いて、各指標の開発工数への影響を分析した。その結果、表2.1 に示す指標が開発工数と相関の高いことが判明したため、これらをモデルの要因として採用することとした。

表2．1　開発要因

開発要因	内　容
①開発するプログラムに関する要因	
・流用率	開発規模に対する流用規模の比率で表す。
・品質	当該プログラムの品質をＫＬ当たりのバグ数で表す。
②開発する人間に関する要因	
・バグ摘出能力（グループスキル）	過去のデータから、当該開発グループがバグを作り込んでから摘出するまでの時間の総和を、摘出バグ数で正規化した値とする。
③開発環境の要因	
・ソフトウェア体系	開発グループの特性（そのグループが開発している物の特性も含む）をソフトウェア体系で表し数量化理論Ⅰ類により分析して数量化する。

以下に、要因として採用したものについて、その考え方と分析結果を示す。

（1）流用率

ソフトウェア開発において高い生産性を確保しようとする場合、既存ソフトウェアの流用（部品使用を含む）は重要な施策の一つである。そこで、流用が工数に与える影響を調べるため、流用率（開発規模に対する流用規模の比率）と生産性（人時当たりの開発ステップ数）との関係を見てみた（図2.6 ）。

この図より、ソフトウェア開発における生産性は、流用率が７０％以上となると急激に向上することがわかる。このことにより、開発工数の予測においては、流用率は７０％を越えると急激に影響が強くなるような形でモデルに反映させるべきであろうと考えた。

図2.6 において、流用率をｘとした場合、生産性はx^3の曲線に最も良く当てはまる。従って、開発ステップ数当たりの開発工数と流用率の関係を見た場合、図2.7 に示すように　$1-x^3$　（ｘは流用率）の曲線で近似できることになるため、流用率はこの形で要因としてモデルに採用することとした。

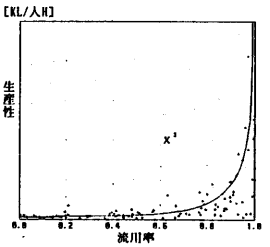

図2．6　流用率と生産性の関係

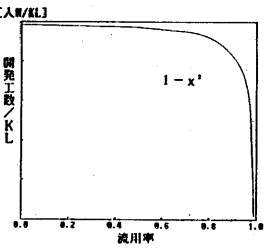

図2．7　流用率と開発工数／ＫＬの関係

（2）品質

一般的に品質の悪い（バグの多い）ソフトウェアは、十分な品質を確保するために必要となる工数は増大する。そこで、開発ステップ数当たりのレビュおよびテストの工数と、各々の摘出バグ数との関係を見てみると、正の相関があることがわかる（図2.8,2.9）。

なお、実際にモデルを利用して必要工数の予測を行う場合には、予測品質（開発計画時に予測した潜在バグ数）を用いることになる。

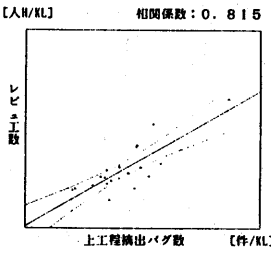

図2．8　上工程摘出バグとレビュ工数の相関

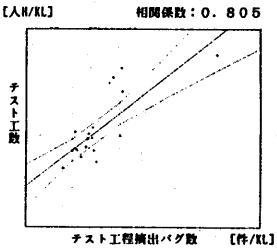

図2．9　テスト摘出バグとテスト工数の相関

（3）バグ摘出能力（開発グループのスキル）

プログラマのスキルはソフトウェア開発にかかる工数を左右する重要な要因と考えられるものの、現状ではこれを測定する方法が確立されているとはいえない。しかしながらいくつかの研究結果が発表されており、今回我々は、エラー寿命[3] の考え方を利用して、バグ摘出能力といった

指標を考えた。

ここでバグ摘出時間とは、バグが作り込まれてから摘出されるまでの時間を指す。当部門では、バグの作り込みと摘出は工程単位で管理しているため、具体的な値は、作り込み工程と摘出工程が同一の場合を１とし、摘出工程が一工程遅れるごとに値を１増やすこととした（図2.10）。

$$\text{バグ摘出能力} = \frac{\Sigma\ (\text{バグ摘出時間})}{\text{全摘出バグ数}}$$

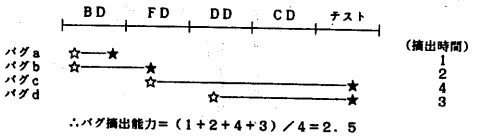

図2．10　バグ摘出能力の計算例

また、一つの機能を複数の担当（グループ）で開発していることが多いことから、バグ摘出能力の設定は開発グループ単位とした。

なお、バグ摘出能力と開発工数の関係を見てみると、本要因の値が大きいほど、開発工数が増大する傾向にあることがわかった。

（４）ソフトウェア体系（開発環境）

ソフトウェアの持つ性格や難易度といった特性に関しては、当部門で開発しているソフトウェアをカテゴリ化して決めた「ソフトウェア体系」を数量化理論Ｉ類を適用して要因に加えることでモデルに反映することとした。

表2.2　ソフトウェア体系（一部）

ID	カテゴリ
A	システム管理
B	運用管理
C	システム生成・保守
：	：

また、マシン環境や開発ツール類の状況などの開発環境は、開発グループによって特徴がみられる。この開発グループは、ほぼソフトウェア体系に沿って組織されているため、本要因は各グループが開発しているソフトウェアの特性と共に、開発環境を表しているといえる。

２．３　モデルの完成形とその有効性

以上の分析の結果得られたモデルは、以下の式で表される。

$$C = a \cdot (1 - X1^{3})^{b1} \cdot 10^{b2 \cdot X2} \cdot 10^{b3 \cdot X3} \cdot X4 \cdot S^{b0}$$

ただし、Ｃ：開発工数［人Ｈ］
Ｓ：開発規模［ＫＬ］
Ｘ1：流用率
Ｘ2：予測品質［件／ＫＬ］
Ｘ3：バグ摘出能力
Ｘ4：ソフトウェア体系別定数
a，bi（i=0,1,2,3）：定数

モデルの形式としては、上・テスト工程共にこれと同じであるが、実際のモデル式は上、テスト工程別に作成されている。

本モデルの有効性を検証するために、本モデル作成に用いたものとは別のリリースのデータを使用して、本モデルによる予測値と実績値の関係を調べてみた。その結果、図2.11に示すように、相関係数および直線ｙ＝ｘとの当てはまり具合い（予測と実績の一致する度合い）からみて、十分実用に耐えうる精度が確保できていると考えられる。

また、従来の見積方法による見積値と実績値の関係（図2.12）と比較しても、明らかに相関係数、当てはまり具合い共に、モデルを使用した場合の方が良い結果が得られている。

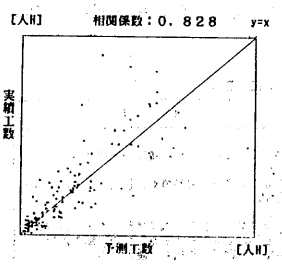

図2．11　モデルによる予測と実績

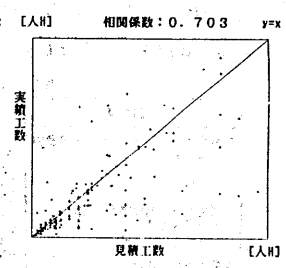

図2．12　従来の方法による見積と実績

３．工程管理方式

工程管理は、工数予測と実績評価・フィードバックの２つの技術が必要であることは、本稿の冒頭で述べた通りである。このうち予測技術に関しては前章で述べた。

我々は、開発計画時に開発工数予測モデルで工数を予測し、開発途中においては、開発に使用した工数が適正であるかと納期が守れるかといった２つの観点から、予測に基づいて工数実績を評価・フィードバックする方式を考案した。以下にその方式について説明する。

３．１　管理指標と評価方法

本工程管理方式で使用される指標は、次の２つである。

指標①：工数不適度

$$= \frac{|\text{実績工数累積値} - \text{予定工数累積値}|}{\text{予定工数累積値}}$$

指標②：納期達成必要工数

$$= \frac{\text{今後必要となる工数}}{\text{納期までの日数}}$$

これらの指標の意味、および評価方法は以下の通り。

（1）工数不適度

本指標は、単に工数の実績と予定の差を見るためだけではなく、品質確保という観点からみて適正な工数で開発が行われているかを見るためのものでもある。

モデルにより求めた工数は、あるソフトウェアを開発するのに必要かつ十分な工数（標準工数）である。従って、実績工数と予定工数の差が0以外の場合、値が正負にかかわらず、標準的な状況では起こらない何らかの問題があることを示している可能性を持つ。

実績工数が標準工数と大きくかけ離れている場合、品質に問題が発生していることが多いということを、我々は経験的に認識していた。つまり、実績＞＞予定の場合、工数の増大は予期しない問題の発生があったということを示している場合が多く、結果として品質劣化を招いており、逆に実績＜＜予定の場合は、標準レベル以上の品質を確保するのに必要な工数がかけられていないことを示す場合が多く、やはり品質は悪いと考えられるのである。

図3.1,3.2 は本指標と品質（開発ステップ数当たりの摘出バグ数）の関係を過去の実績データで分析した結果である。相関はあまり高くはないものの、明らかに本指標の値が大きい、つまり実績と予定の差が大きいものほど、品質が悪くなる傾向を示している。

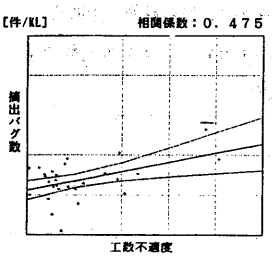

図3．1　工数不適度と品質（予測＜実績）

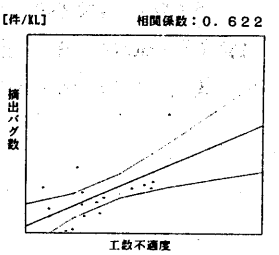

図3．2　工数不適度と品質（予測＞実績）

これをさらに詳細に分析したのが表3.1 である。品質比をみると、本指標が0．6以上になると急激に品質が悪くなることがわかる。

表3．1　工数適正度と品質の関係

工数適正度	品質比※	品　質
＜0．2	1．00	良↑
0．2≦ ＜0．4	1．09	｜
0．4≦ ＜0．6	1．15	｜
0．6≦	1．90	↓悪

※品質比～工数適正度＜0．2の開発物の平均バグ数／KL
＝1とした場合の、各層での平均バグ数／KL

よって、本指標≠0の場合は、現状分析を十分行って工数に差が生じた理由を明確にし、今後の計画に影響が出ないか吟味する必要がある。特に、本指標≧0．6の場合は品質面を重点的に確認する必要がある。

（2）納期達成必要工数

本指標は、納期の観点から工数を評価するために定義したものである。

工数不適度に問題が無い、つまり工数の実績と予定にそれほど差異が無い場合でも、納期までの期間で使用できる工数（保有工数）＜今後開発に必要な工数（予定工数）となってしまった場合、そのまま開発を進めると納期遅れといった事態を招く。そのため、工程管理においては、単に工数の予実差をみるだけではなく、納期といった観点からの評価も必要である。

本指標での評価は、計画時の本指標値と現時点での本指標値を比較することで行う。すなわち、

$$\frac{\text{今後の予定工数（計画時）}}{\text{今後の予定日数}} < \frac{\text{今後の予定工数（現状）}}{\text{実際の残り日数}}$$

となった場合、納期に間に合うようにするために、要員増強等、何らかの対策を講じる必要有りということになる。

この時、計画はモデルによる予定工数が保有工数内に収まるよう策定されている必要がある。また、本評価を行う場合、今後の予定工数（現状）の値は、現状を反映して見直しを行った値を使用する。

3．2　工程管理手順

（1）計画

開発計画立案時には、まず前述の開発工数予測モデルによって所要工数を予測し、これを予定工数とする。各要因の値は、今回の開発における計画値を使用するが、以下に示すものは、特にその算出方法を規定している。

- 品質：

  別途定義されている潜在バグ予測モデルを使用。

- バグ摘出能力：

当該グループの前回開発時の実績値を使用。

こうして求めた工数は、さらに各開発工程に分配する。分配に関しては、前回開発時の工程別実績工数比を標準比率とし、これに準じて行う。こうして各工程の工数が決定したら、保有工数を考慮して開発日程を決定する。

（2）開発中の実績評価とフィードバック

開発工程実施中は、適当なチェックポイントを設けて、3．1で示した方法で実績評価と分析を行い、その結果をフィードバックしていく。

図3.3 に工程実施途中の管理手順について示す。なお、モデル要因の値が、予定工数決定時から変化した場合は、随時、最新値を用いてモデルにより予定工数を再計算しておく必要がある。

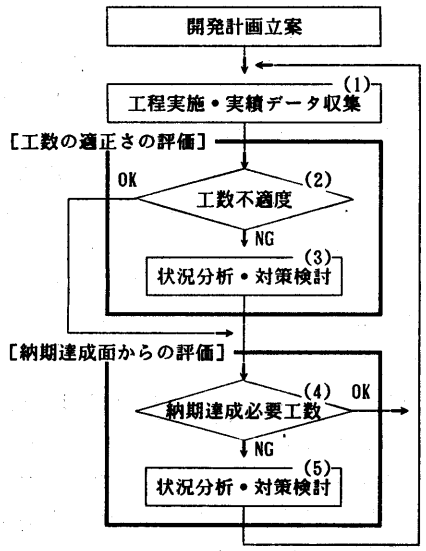

図3.3 工程管理手順

(1):工程実施中は常に実績データを収集しておき、定められたチェックポイントにきたら実績評価作業に入る。

(2):工数不適度≠０の場合はＮＧ。

(3):工数の予定と実績に差異が生じた理由を分析する。ここで、工数不適度≧０．６の場合は、品質面に関して特に十分な調査・分析が必要である。この分析結果をもとに、開発方式や要員配置の変更など、問題点を解決するための対策を検討すると共に、今後どのくらい工数がかかるのかを判断する。

(4):計画時の値＜現状の値ならばＮＧ。

(5):保有工数を見直し、現状の要員でカバーできる範囲か分析検討する。もしカバー不可能ならば、要員の増強や納期の調整（→保有工数の増加）や、開発機能や開発方式の見直し、要員配置の変更（→予定工数の見直し）等の対策を検討する。

（3）開発終了時

開発終了時には、実績工数の最終的な評価と、次回の開発へのフィードバックを行う。評価は、工数不適度による評価により、今回の開発の問題点を分析する。この分析結果および今回の実績データに基づいて、モデル式の見直し（係数の再設定等）や、次回開発時に問題点を解決するための対策の検討などを行う。

4．まとめ

以上述べてきたように、本方式によって定量的根拠に基づく工程管理を実施することができる。今後の課題としては、モデル要因をより適切なものとしてモデル精度の向上を図ることと、予定工数決定から担当者レベルの線表作成までのスケジューリング方式の検討があると考えている。

［参考文献］

[1] N.Honda,T.Mano,Y.Hirai:"Quality Assurance System Throughout Software Life Cycle", Proceedings of the 2nd Europian Conference on Software Quality Assurance (1990)

[2] B.W.Boehm:"Software Engineering Economics", Prentice-Hall (1981)

[3] 松本、井上、菊野、鳥居:"エラー寿命に基づくプログラマ性能の実験的評価ー大学環境におけるプログラム開発",電子通信学会論文誌 Vol.J71-D No.10 pp.1959-1965 (1988)

[4] 花田（編）:"ソフトウェアの計画と管理"(日科技連ソフトウェア品質管理シリーズ5), 日科技連 (1987)

[5] T.C.Jones(著),井上、荒川（訳）:"システム開発の生産性",マグロウヒル (1986)