

プロジェクトプログラミングについて

本吉 由紀夫
プロジェクトフォーミュラ

大槻 繁
(株) エクイティ・リサーチ

ソフトウェア開発プロジェクトの成否を左右する要因としてのプロジェクト管理を適切に実施するには、IEEE Std. 1058 Standard for Software Project Management Plans などに基づく、適切なプロジェクト管理計画を作成しプロジェクト管理を実施する必要があるが、その作成コストや修正コストが多であることが問題である。プロジェクトプログラミング技法は、プロジェクトポートフォリオを利用して適切なプロジェクト管理計画を、シミュレーション技術やパターン技術により機械的に作成し、修正量を低減することにより、プロジェクト管理計画書の作成・修正コストの低減さらにプロジェクトコストの低減を図ると同時にプロジェクトの成功確率向上を図るものである。

Introduction of Project Programming Technology

Yukio Motoyoshi
ProjectFormula

Shigeru Otsuki
Equity Research, Co.Ltd.

The key to success of software development project is to develop a well designed Software Project Management Plan, typically based on the IEEE Std. 1058 Standard for Software Project Management Plans (SPMP). The problem is the cost of development and maintenance of the SPMP. Introducing Project Programming Technology into the development of SPMP, they can generate a well designed SPMP on the fly based on the project-portfolio which is made automatically by the technologies such as simulation, software patterns and process technology, which can decrease the SPMP development/maintenance cost and reduce the total project cost as well.

はじめに

ソフトウェア開発作業は、ビジネスニーズや市場環境の変化に伴い、高品質・短納期・低コスト化が危急の課題となっている。この課題に向けて、過去種々の理論や技法を開発し適用することにより、生産性や品質を向上してきた。しかしソフトウェアは他の生産財と異なり、その生産工程における人間の関与を排除することは不可能であり、工程を管理することが非常に重要かつ困難である。その結果、平均50%のプロジェクトはデスマーチ状態に陥ることになる[1]。一部の統計では、納期・予算・機能において期待通りに出荷されたプログラムは、発注量の2%にすぎなかった[2]。デスマーチを避けるにはピープルウェア（人材）が大事であるとする意見もある[3]が、現状を見るとほとんどのソフトウェア企業は、そのような人材を確保できない。eXtreme Programming などの新しい技法においても、計画のフェーズにおいては経験に基づき考えることが多く、計算による定量化はない[4]。

一方、企業のグローバル化に伴い、ソフトウェア開発チームの構成までグローバル化しプロジェクトがバーチャル化[5]しており、単純なピープルウェア議論は通用しなくなっている。このように、プロジェクトマネジメントは危機的状況にあるにもかかわらず、現実には即したプロセス制御・開発環境は、求められているのに存在しない。

プロジェクトを成功させるには、プロジェクトの計画・管理がプロジェクトコストに占める割合として、全体の20%以上が必要[2]とされている。それにも関わらず、計画・管理の効果が判りづらいため、開発技法やプロセスに執着してソフトウェア開発作業の生産性向上によるコスト削減を図る場合が多い。しかしながら、生産性向上によるコスト削減が限界に近い昨今、プロジェクト管理計画立案と管理を適切に行い、プロジェクトを成功へ導く技法が必須といえる。

プロジェクトプログラミング技法は、このプロジェクト管理計画立案およびプロジェクト管理における計画精度の向上、リスク管理の適正化、計画や管理の工数削減によりプロジェクトの成功確率の向上を狙いとした技法として提唱するものである。SWEBOOK(Software Engineering Body of Knowledge)の知識体系を、プロセスプログラミング理論[6]、パターン理論[7]、およびポートフォリオ理論[8][9]により定式化するものであり、以下の特徴を備えている。

- ・プロジェクト計画立案の初期段階において、各種技法やプロセスを組み合わせたシミュレーションを行い、最適な計画を選択できるようにする
- ・モニタリングはパイナリトラッキングにより実施し、リスクを検知したら再びシミュレーションを行うことにより、プロジェクト計画を修正する

1. プロジェクト管理計画とプロジェクトプログラミング

1.1 プロジェクトプロセスモデル

図1に典型的なプロジェクトのプロセスモデルを示す。

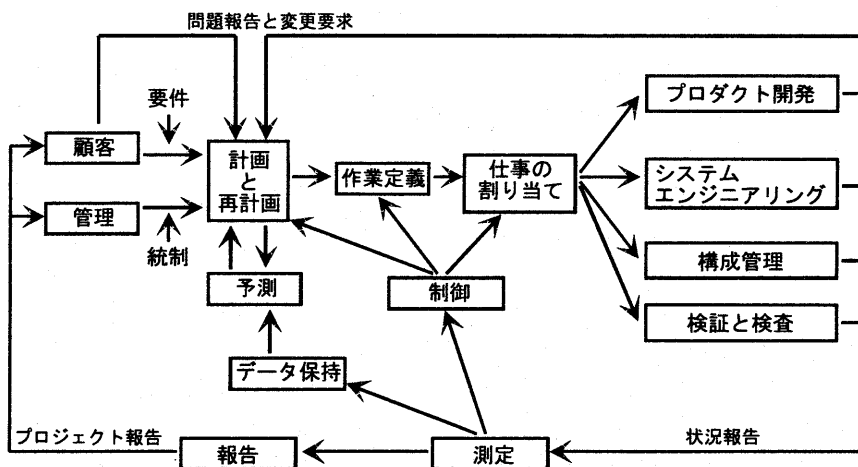


図1. ソフトウェア開発プロジェクトのプロセスモデル

ソフトウェアプロセスとして様々なモデルが提唱されているが、いずれも、繰り返しの仕方とそれに伴う成果物の規模が異なる程度で、ほぼこのプロセスモデルに対応付けが可能である。

従来からのCASEツールは、このプロセスモデルにおいて主として、予測・データ保持、仕事の割当、

プロダクト開発、システムエンジニアリング、構成管理、検証・検査、測定、を対象としてきている。

これらツールは、それぞれの対象に対しては十分に機能するものであるが、上流からの入力に誤りがあってもそれを検知するすべが無い。象徴的なのは、この根元とでも言える部分が計画（再計画）と作業定義である。即ち、各種のツールが適切に機能するかどうかは計画段階にかかっていると看做しても過言ではない。顧客の要求を適切に取り込むための技法やツールが必要とも言えるが、現実には計画段階で適切な技法を用いていれば適切に計画ができる。

前述の各種の問題点は、結局各種ツールの入力となる計画段階の出力情報が適切ではないため、それ以降の予測から測定までの精度が向上しないことを意味する。

この様に、プロジェクト管理の計画・再計画の段階が、開発プロジェクトの成否を左右するが、この部分について効果的な技法として、プロジェクトプログラミング技法を紹介する。

1. 2 プロジェクト管理計画

(1) プロジェクト管理計画のアウトライン

表1に、IEEE1058.1 プロジェクト管理計画のガイドラインについて、アウトラインと平均的な作成規模を示す。

表1において、目次項目はガイドラインにおいて提示されている項目であり、作成ページ数は、いくつかのプロジェクトの経験から導いた平均的な最終作成ページ数である。プロジェクトの規模は10人で12ヶ月程度のものである。また、更新頻度は、計画書の更新が必要となる頻度を+++（週単位程度）、++（月単位程度）、+（3～6ヶ月単位程度）で示す。作成ページで“-”としているのは、単独では1ページ未満でも構わない部分である。

これから判るように、プロジェクト管理計画書の作成は、300ページ超もの容量となり、短期間で作成できるものではない。さらに作業単位(Work Activity)の記述は、1週間程度の作業量となるまで分解することが望ましいとされているが、その意味するところは、構造設計や詳細設計ができていくことが必要であり、計画段階とすることを考えると、矛盾する。そこで、実際には最初は大きめの製品機能または開発プロセスに基づくトップレベルの作業単位（5～10個程度）を記述し、プロジェクトの進捗に伴い、記述を追加していくことになる。

重要な点は、プロジェクト管理計画においては、この例では300ページ相当分の情報量を必要としていることである。この情報が不足している場合は、十分なモニタリングやリスク管理を実施できなくなる。

このように、プロジェクト管理計画を実施することは、プロジェクトマネージャにとっては膨大な負荷になる。さらにプロジェクトの進捗特に設計段階での情報を的確に反映していくという継続的な作業がついて回る。繰り返し型の開発プロセスにおいては、これが頻繁に発生することになり、単なる負荷といえず、このために本来の進捗測定作業や報告作業に支障を来しかねない。プロジェクトマネージャの負荷が高すぎれば、リスク管理のための再計画などが不十分になってしまう。

即ち、プロジェクト管理計画として十分な情報量を作成・維持するための負荷は、可能な限り低減できることが望ましいと言える。

(2) プロジェクト管理計画の価値

前の例では、最終的な作業単位の記述は約500個にのぼる。これは単純に、10人に1年間の作業を毎週割り当てると、これだけ必要になるからである。現実には5人～10人程度のチームで1ヶ月から6ヶ月程度のプロジェクトが多いと考えられる。

それでも、5人月で20作業単位、60人月で240作業単位の記述が必要となる。作業単位を1週間よりも小さくすると、作業単位は簡単に増加する。しかしこの作業単位を記述することは、ソフトウェア開発において何ら成果物となるものではない。それにもかかわらず、この様に膨大な工数をかけてプロジェクト管理計画を作成し、プロジェクト管理を実施する必要性が本当にあるのだろうか。

表1. IEEE1058.1プロジェクト管理計画ガイドライン（目次）と作成ページ数および再利用性

目次項目	作成ページ数	算出法	更新頻度	再利用性	内容	
Title Page	1					
Signature Page	1					
Change History	1					
Preface	1					
Table of Contents						
List of Figures	1					
List of Tables	1					
1 Overview						
1.1 Project Summary						
1.1.1 Purpose, Scope, and Objectives	2		++	低	プロジェクト毎に記述	
1.1.2 Assumptions and Constraints	1		++	低		
1.1.3 Project Deliverables	1		++	低		
1.1.4 Schedule and Budget Summary	1		++	低		
1.2 Evolution of the Plan	1		+	低		
2 References	1					
3 Definitions	1					
4 Project Organization						
4.1 External Interfaces	1		+	高	組織依存性が高く、変化が少ない	
4.2 Internal Structure	1		+	高		
4.3 Roles and Responsibilities	1		+	高		
5 Managerial Process Plans						
5.1 Start-Up Plan						
5.1.1 Estimation Plan	-		+	高	組織依存性が高いが、チームが変化すると内容も変更が必要	
5.1.2 Staffing Plan	1		+	中		
5.1.3 Resource Acquisition Plan	-		+	高		
5.1.4 Project Staff Training Plan	1		+	中		
5.2 Work Plan						
5.2.1 Work Activities	250	Man*Week/2	+++	低	プロジェクト毎に記述	
5.2.2 Schedule Allocation	12	Month	++	低		
5.2.3 Resource Allocation	22	Man, Month	++	低		
5.2.4 Budget Allocation	1			低		
5.3 Control Plan						
5.3.1 Requirements Control Plan	-		+++	低	プロジェクト毎に記述	
5.3.2 Schedule Control Plan	1		+++	低		
5.3.3 Budget Control Plan	-		+	低		
5.3.4 Quality Control Plan	1		+	低		
5.3.5 Reporting Plan	3	D/W/M		中	組織依存性が高いが、チームが変化したり、対象知識が少ない場合などには内容も変更が必要	
5.3.6 Metrics Collection Plan	1		+	中		
5.4 Risk Management Plan	2		+++	中		
5.5 Closeout Plan	1			中		
6 Technical Process Plans						
6.1 Process Model	1		++	中	組織依存性が高いが、対象知識が少ない場合などには内容も変更が必要	
6.2 Methods, Tools, and Techniques	3		+	中		
6.3 Infrastructure Plan	3		+	中		
6.4 Product Acceptance Plan	1			高	組織依存性が高く、変化が少ない	
7 Supporting Process Plans						
7.1 Configuration Management Plan	1		+	高		
7.2 Verification and Validation Plan	1			中		
7.3 Documentation Plan	2			高		
7.4 Quality Assurance Plan	1			高		
7.5 Reviews and Audits	1		+	高		
7.6 Problem Resolution Plan	1			中		
7.7 Subcontractor Management Plan	1			高		
7.8 Process Improvement Plan	1			中		
8 Additional Plans	1					
Annexes	10					
Index	5					
合計	346					

現実にはその必要性がないと判断されて、これだけのプロジェクト管理計画を作成しないことが多いと考えられる。特に ISO などの認証などに無縁の開発グループでは、このような部分に工数を投入することは労多くして益少なしと思われても仕方がない。米国においても、この様に徹底したプロジェクト管理計画書を作成する必要があるのは主として政府系のプロジェクトが中心である。これは、情報公開を含めた、投資の最適性を判定する材料として利用されるためである。米国の一般企業では、昨今 Extreme Programming によるプロジェクト運営が活発化しつつあり、直接の成果物とならないドキュメントの作成にかける工数の低減が図られている。

では、プロジェクト管理計画書の価値は一体何か。

上記のように公共的なプロジェクトにおける場合以外にも、100人月程度を超える場合、従来型のプロジェクト運営を行う方が確実であり、そのために最も効果的な測定技法として、バイナリトラッキングがある。これにより、作業単位が消化されていくことを確実に把握でき、遅延が発生した場合、即座にリスク対応の手続きに着手できる。バイナリトラッキングを実施するためには、逆にプロジェクト管理計画書において作業が明確に定義されている必要がある。

また、プロジェクト投資の最適性の判定が必要な場合である。プロジェクト運営が適切になされているかどうか、投資が利益を生むのかどうかを、ソフトウェアが稼動する前に判定できる材料は、プロジェクト管理計画書しかない。投機的にソフトウェア開発を行うのであれば、このような計画書は不要であるが、経営判断を伴う投資となるソフトウェア開発においては、不可欠のドキュメントであり、その価値は、投資金額あるいはそこから得られるであろう利益そのものに匹敵すると考えるべきである。

(3) プロジェクト管理計画の作成コスト

プロジェクト管理計画を作成するにはどれだけの工数が必要になるかが、その後の直接開発工数を如何に確保するかという点で重要になる。表1からも判るように、再利用性の高い項目もあり、毎回作業対象となるのは、主として概要部分と作業単位の洗い出しと作業定義の部分である。プロセスモデルで、顧客の要求から開発機能の洗い出しと見積もり、そして作業単位への展開が、プロジェクト管理計画を作成する上での主なコスト要因となっている。表1の規模の開発において、初期のプロジェクト管理計画は、50ページ相当の情報量になる。これを開発の進捗に伴い改訂していくのが従来型のアプローチである。そこで、リスクが発生せず、改訂作業そのものを管理コストとした場合、初期のプロジェクト管理計画の作成コストは、プロジェクトコスト全体の3～5%程度が妥当であると考えられる。

1. 3 プロジェクトプログラミング技法の位置づけ

従来は、プロジェクト管理計画書の初期版を作成した後、開発の進捗に伴う改訂のコストが避けられない。改訂のタイミングが開発プロセスに依存するとは言え、計画・管理コストが全体コストの20%以上という事実には変りはない。

プロジェクトプログラミングは、初期の計画段階で新たにプロジェクトのパターンやソフトウェアのアーキテクチャパターンや開発プロセスパターンとシミュレーションツールを導入することにより、機械的に、最終版に近いプロジェクト管理計画書を作成するための技法である。これにより、投資価値をプロジェクトの初期段階で確実に把握することを可能とし、計画・管理のコストを低減して開発そのものや、トラッキングとリスク管理中心に振り向けることにより、プロジェクトの成功確率の向上を狙う。

2. プロジェクトプログラミング

2.1 基本的な考え方

図2にプロジェクトプログラミング技法を用いたプロジェクトポートフォリオとプロジェクト管理計画書（SPMP）作成の基本的な流れを示す。

プロジェクトプログラミングでは、従来静的なリスク管理に終始しがちであったプロジェクト管理において、計画のフェーズで時系列のリスク変動に着目し、複数のプロジェクト計画のリスク比較をしながらリスクを極小化する基本プラン・リスク対応プランの組み合わせ（プロジェクトポートフォリオ）を作成する。プロジェクトの進捗に伴い EV(Earned Value)や VAR(Value at Risk)の計算を行えばリスクの発生を早期に検知できるとともに、アクション（リスク対応計画）を即座に適用可能となり、プロジェクトの成功率や損失を検証可能とするものである。また、予定外のリスクが発生した場合、発生時点から目標（プロジェクト完了）までの確率過程を計算（ソルバー）し、実施可能なアクションを見いだしたり、致命的リスク（予定通りのプロジェクト完了は不可能）であると判れば、例えば、期間を固定して要件の削減やリソースの追加など変動させられるパラメタを指定して確率過程を計算することにより、最善のアクションを導出するものである。

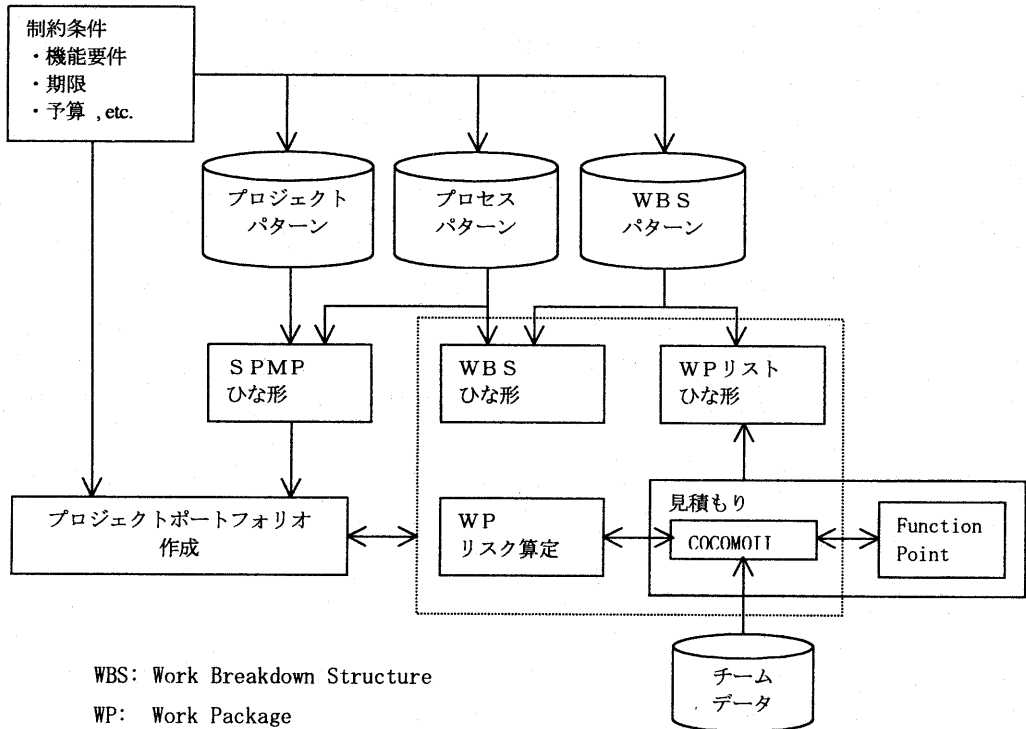


図2. プロジェクトプログラミング技法の基本的な考え方

図2において、成果物はSPMPを含むいくつかのプロジェクトポートフォリオである。プロジェクトポートフォリオは、幾通りかのWBSに基づき生成されたWP毎のリスク算定を元に、プロジェクトリスクを計算し、コストに対する投資効果をまとめ上げたものである。

制約条件を元に、プロセスやWBSのパターンから候補となるインスタンスを生成し、WP単位の見積

もり、積算、リスク算定を行う。見積もり技法としては、COCOMOII[10]ならびに Function Point[11]を用いる。プロジェクトポートフォリオとしては、例えば、初期のリスクを低くするプロセス・WBS や初期のリスクは高くとも、中盤以降のリスクを低く抑えたプロセス・WBS、などが作成できる。プロジェクトマネージャは、プロジェクトポートフォリオを参照して、いずれかのシナリオを選択する。

プロジェクトはバイナリトラッキングを行い、ある WP で遅延が発生した場合、別のプロジェクトポートフォリオのシナリオを選択し直していくことにより、リスクへの対応が即時に簡単に実施できる。適切なシナリオが無い場合には、元のシナリオにおいて、遅延 WP 以降のシナリオ（プロセス、WBS）の再作成を行うことにより、新たなシナリオを選択できる。それでも制約条件を満たすことができない場合、期限・予算・機能（WP）の中の2つのパラメータを固定して、1つを変動させて解を導く（ソルバー）ことができる。この様に、プロジェクトプログラミングでは以下の技術を利用する。

(1) プロジェクトポートフォリオの作成技術

従来、プロジェクト計画は、フレームワークに基づきプロジェクトマネージャの経験により実施されることが多い。このため、プロジェクトマネージャの養成は困難であり人材不足となっている。

そこで、金融工学的なリスク算定の技術を適用し、WBS の作成過程において、WP 毎のリスクと価値のバランスを取るとともに全体リスクを極小化できるように、プロセスと機能と工数を算出する。これらのデータは、機能・コスト・納期のパラメータに基づきパターンとして再利用可能とすることにより、初級マネージャでも精度の高い分析・予測を可能とする。

(2) プロジェクトシミュレーション技術

また、経験に基づく計画作成の欠点として、習熟度情報やパターン情報が欠落（不足）している場合に大きな誤差が発生してもそれを予測できない点にある。この誤差を経験則で扱ってリスク管理により対応する計画（コンティンジェンシープラン）を立案することが従来の方法であるが、そもそも欠落している情報がきちんと特定できていない場合が多く、結果的に大幅なプランの修正が必要になる。また、後に計画の大規模な修正を行うことは WP の大幅な修正が発生し多大な工数を要するため、詳細な修正は不可能に近いのが現状である。プロジェクトプログラミングでは、リスクは後にコントロールするのではなく、事前にアセスメントを徹底して行いリスクを極小化しておくこと（プロジェクト管理計画精度の向上）を支援する。このために、欠落している情報について習熟度レベル別等の各種の仮説を設定してプロジェクトプランを生成し、そのスケールを容易に計算でき、それらの結果を比較・検討可能とし（可視化）、リスク評価を行うシミュレーション技術を導入する。

(3) プロジェクトパターン技術

上記の計画作成技術を適用できるように、リスクを極小化できたプロジェクトとプロセスをカスタマイズしたり、パターン化し再利用する技術である。

このプロジェクトパターン技術により、さまざまな状況に於けるプロジェクトの制御方法を、勘や経験に頼らずに学び、活用することができる。

(4) リスクトラッキング技術

プロジェクト管理計画の精度が向上すると、プロジェクトのモニタリングの精度を向上し、リスクの早期検知とシミュレーションにより得た複数の仮説との比較を行うことが可能となる。また、現実合うような逆演算（ソルバー）技術により、途中から最小のコストでプランを修正するシミュレーションを行うことができる。これらにより、リスクの制御が容易にかつ最小の工数で実現可能となり、リアルタイム性の高いリスク追跡と修正のためのアクションプランの立案が可能となる。

プロジェクトプログラミングにより、新たなソフトウェア開発プロジェクトにおけるプランニングの大まかなステップは次のようになる。

- ①既知のプロジェクトパターンから最適と思われるものを検索する。
- ②新プロジェクトに合わせてチーム情報などを見直す。
- ③要求上の制約条件を設定する。
- ④ソフトウェアの目標価値を計算する。
- ⑤リスクの目標範囲を指定する。
- ⑥制約・価値・リスクからWBSを計算（修正）する。
- ⑦目標に到達できないWPを中心に、②に戻って制約を満たすような局所最適解を計算する。
- ⑧いくつかのプロジェクトプランの候補を作成したらその組み合わせの中から最適なプロジェクトポートフォリオを作成する。

2. 2 プロジェクトプログラミングツール

これら一連の計算を行い、生成したプロジェクトポートフォリオを、プロジェクト管理ソフトウェアへ出力するツールを現在開発中である。

おわりに

プロジェクトプログラミング技法をツールとともに採用することにより、多くの場合、プロジェクト管理計画書の作成工数を1/10程度まで劇的に削減できると考えている。また、作成されたプロジェクトポートフォリオは、経営的な判断の材料ともなりうるものであり、開発投資の適正化を図ることができる。さらに事前にリスクアセスメントができていますのでプロジェクトの運営コストが劇的に削減し、万一リスクが発生しても、対応が容易になるものと期待している。

参考文献：

- [1] E.Yourdon :Death March -The Complete Software Developer's Guide to Surviving "Mission Impossible" Projects, Prentice-Hall, 1997. 邦題「デスマーチ」, 松原/山浦 訳, トップラン.
- [2] D.W.Karolak :Software Engineering Risk Management, IEEE, 1995.
- [3] T.DeMarco,T.Lister :Peopleware, DorsetHouse, 1987. 邦題「ピープルウェア」, 日立ソフト生産性研究会訳, 日経BP
- [4] K.Beck,M.Fowler :Planning Extreme Programming, Addison-Wesley Longman, 2000.
- [5] D.W.Karolak :Global Software Development, IEEE, 1998.
- [6] L.Osterweil :Software Processes are Software too, 9th ICSE, 1987.
- [7] E.Gamma, R.Helm, R.Johnson, J.Vlissides: Design Patterns, Addison-Wesley, 1995.
- [8] L.Black, et. al. :New Directions in Portfolio Assessment:Reflective Practice, Critical Theory, and Large-Scale Scoring, Boyton/Cook, 1994.
- [9] P.Best :Implementing Value at Risk, John Wiley & Sons, 1999.
- [10] B.Boehm:Software Engineering Economics, Prentice-Hall, 1981.
- [11] C.Jones :Estimating Software Costs, McGraw Hill, 1998.