第１分科会グループＢ

開発現場にうれしさのわかるメトリクス有効活用の提案、事例研究と実践
～プロジェクトを成功させるために問題を先取りしよう！～

A “Profitable” Method and Practice of Applying Metrics to Development Site

-Let's find problems earlier for successful projects! -

主査	三浦 邦彦	矢崎総業㈱		
副主査	藤巻 昇	㈱ 東芝		
リーダ	山内 一資	㈱ デンソー		
	天野 佑太	㈱ 日立メディコ	山下 憲男	㈱ NTT データ三洋システム
	中村 光治	オリンパス㈱	関川 信吉	㈱ アルゴ２１
	鈴木 武志	日本電気通信システム㈱	髙津 修一	㈱ 山武

1. 研究概要

ソフトウェア開発プロジェクトにおいては、生産されるソフトウェアそのものや、プロジェクトに携わるメンバの活動状況など、目に見えにくいものが多い。それが原因となり、開発プロセスやプロダクトの品質低下、コスト超過、スケジュール遅延などの様々な問題が起こるだけでなく、そういった問題の発見の遅れがプロジェクトを混乱に陥れている。

このような目に見えにくい活動やソフトウェアをメトリクス等によって「見える化」し、問題の早期発見・改善を行うために、様々な取り組みがおこなわれている。しかし、こうした活動をスタッフ部門が主導して推進しても開発現場には根付かず、思うように成果が上がらないという悩みを抱えるプロジェクトが多い。

そこで当分科会では、開発現場に根付きやすい、開発現場にとって「うれしい」メトリクスの活用方法を提案し、これらを実際の開発現場に適用して事例研究をおこなった。

Abstract

In a software development project, there are many factors that cannot be observed easily, such as the software itself and the activity of project members. This decreases the quality of the development process and product, balloons the development cost, and causes schedule delays and other problems. Moreover, a delay in finding these problems eventually corrupts the project.

There are a number of methods available that let you “statistically visualize” those factors by using software metrics, and find and fix problems in early stages. However many projects are facing the issue that these activities do not become ingrained nor accomplished on the development site even when staff members (supporting divisions) attempt them.

We present a “profitable” method to apply metrics for the development site and practice on the site.

2. テーマ選定の理由

今回、第１分科会には品質保証部門の立場で参画したメンバ、また、開発現場から参画しているメンバが、それぞれのソフトウェアメトリクスに対する思いを持って集まった。品質保証部門は、どうやったら開発現場が効率よくデータを収集し活用できるか、日々、頭を痛めている。一方、開発現場では、忙しい開発の合間を縫ってデータを収集するが、どう役に立つかわからない、また、品質保証部門からの要求でデータを集めているにすぎない、と思っている。両部門ともに納得して、うれしさを共有しプロジェクトを成功させるには、どうしたらよいか日々悩んでいる。

そこで、我々第１分科会のメンバは、開発現場に喜ばれる、「うれしさ」のわかるメトリクスの活用方法と事例研究を考えることをテーマとして選択した。

3. 活動目標

次のことを当グループの活動目標とした。

1) 現場の「うれしさ」を考察し、「うれしさ」の定義を考える
2) 現場に「うれしさ」を提供するための、メトリクスの活用方法を研究する
3) 現場が「うれしい」メトリクスの活用方法と実践事例を提示する

4. 活動内容

この１年の活動内容を「表-1　活動経過」に示す。

表-1　活動経過

日程	活　動　内　容
平成 19 年 4 月 21 日	研究会メンバ自己紹介と検討課題洗い出し、グループ分け。
平成 19 年 6 月 1 日	各社の課題に関して意見交換を実施。 グループリーダを決定。
平成 19 年 7 月 12 日 ～13 日	方向性決定、論文骨子作成。
平成 19 年 8 月 31 日 （臨時会）	自社のメトリクスについて意見交換。現場に役立つメトリクスについて議論。
平成 19 年 9 月 21 日	収集しているデータ・項目について持ち寄りと骨子に基づいた論文フレーム決め、中心点・定義決め、事例案決め。
平成 19 年 10 月 10 日 （臨時会）	同上、論文作成。
平成 19 年 11 月 16 日	論文読み合わせ。１回目。
平成 19 年 11 月 30 日 （臨時会）	論文読み合わせ。２回目。
平成 19 年 12 月 14 日	論文内容精査１回目。
平成 20 年 1 月 11 日	論文内容精査２回目。
平成 20 年 1 月 25 日 （臨時会）	論文内容精査３回目。
平成 20 年 1 月 25 日 ～2 月 5 日	メンバによる論文全体のレビュー、および精査。

5. 研究成果および考察

我々の研究として、「プロジェクトを成功させるために問題を先取りしよう」の想いと開発現場の「うれしさ」をあわせて定義した。各メンバの事例を研究して、開発現場の「うれしさ」にマッチするメトリクスを開発現場に試行した。以下、その成果と考察について説明する。

5.1. 「うれしさ」の考察と定義（「うれしさ」への想い）

プロジェクトは品質、費用、納期を計画通りに達成することを目標に、開発現場は日々の活動を行っている。しかし、プロジェクトの実行においては目標達成を阻害する様々な事態が発生する。

開発現場にとっての「うれしさ」とは、このような状況に陥ることなく、プロジェクトが成功することであると考えた。

成功するプロジェクトに参画することにより、開発メンバは、

- お客さん（次工程の担当者）が喜ぶ
- 自分が成長できる
- 会社の収益が上がる
- 達成感が得られる（スケジュール通り完了することにより）

といった開発者が本来感じるであろう「うれしさ」を得ることができるのである。

プロジェクトメンバがうれしさを感じるためには、プロジェクトメンバが次の認識を持つことが必要である。

- 合意形成ができる。
- 情報共有ができる。
- 危機感の共有ができる。
- 納得（して仕事が）できる。
- 感動できる。

これらの認識を得るためには、現場が「うれしい」メトリクスの提供が必要と考えた。

5.2. 現場が「うれしい」メトリクスの定義

現場が「うれしい」メトリクスを考える上で、現場が「うれしくない」メトリクスとは、

- データを取るだけで、何もフィードバックが無い
- 改善につながらない
- 個人攻撃につながる
- データを採るのが大変

と考えられ、逆説的に考えると「うれしい」メトリクスとは、

- 有効なフィードバックがある
- 改善につながる
- 個人のスキルアップにつながる
- 簡単に採れる

というように定義できる。このように定義すると、「うれしい」メトリクスとは長短期的に次へつながるメトリクスであることがわかる。

5.3. 現場に「うれしさ」のわかるメトリクスと適用タイミングの考察

現場に「うれしさ」のわかるメトリクスについて、必要な時に必要な情報を得るために、メトリクスの選択指標と適用タイミングについて、分けて考察した。

「うれしさ」を提供するためのメトリクスを考える上で、我々はまず、その選択指標について検討した。

（1）メトリクスの選択指標

そのメトリクスを使う上で「うれしいこと」、「うれしいことの実施行動」を紐付けることで必要なメトリクスを抽出できると考えた。

うれしいこと【目的】	選択指標【目標】
・有効なフィードバックがある ・改善につながる ・個人のスキルアップにつながる ・簡単に採れる	・進行中プロジェクトがうまくいく ・次プロジェクトがうまくいく ・ノウハウが向上する ・無駄な時間が削減できる

次に、適用タイミングについて説明する。

（2） 適用タイミング

「うれしさ」のわかるメトリクスを適用するタイミングは、今動いているプロジェクトへのリアルタイムな状況へのフィードバックアプローチと次プロジェクトへ教訓を活かすためのフロントローディングアプローチの２つがある。

すなわち、フィードバックアプローチは当事者の活動を主に置き、フロントローディングアプローチは、プロジェクト活動をまたいだ組織活動とみることができる。

a) 今進行しているプロジェクトへのフィードバックアプローチ

すでに進行しているプロジェクトに対し、その時その時に手を打っておくことは、そのプロジェクトへの影響を最小限にすることのみならず、並行に動いている他のプロジェクトへ被害を及ぼさないためにも大切なことである。そのためには、リアルタイムに状況を把握し手を打つことが必要となる。

リアルタイムに手を打つためには、進行中のプロジェクトに対して、いかにわかりやすく、見る人の負荷を少なくし判断をし易くするかが大事である。

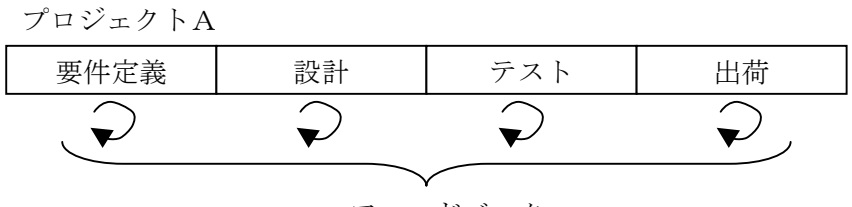


図１　進行中プロジェクトへのフィードバックアプローチ
（当事者活動）

b) 次プロジェクトへのフロントローディングアプローチ

プロジェクトは、新規開発、前プロジェクトの派生開発、保守などいくつかの種類があるが、元となったプロジェクトの良い点、反省点をプロジェクトの始まりに省みて、こ

れから始まるプロジェクト開発へ活かすことがよりよい効率的･効果的な開発といえる。フロントローディングに手を打つためには､前プロジェクトのＱＣＤの結果のみならず、前プロジェクトの始まりから終わりまで、顧客との仕様確認のやりとり、設計、テスト結果、制約条件などを振り返り、前プロジェクトとの相違や制約を検討し計画を練ることが大事である。

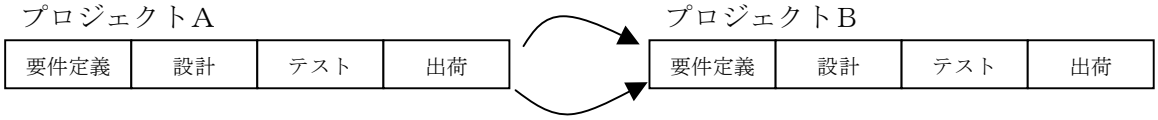


図２　次のプロジェクトへのフロントローディングアプローチ
（プロジェクトをまたいだ組織活動）

以上の「メトリクスの選択指標」と「適用タイミング」を踏まえた上で、メトリクスの例と活用の仕方について説明する。

5.4. 「うれしさ」を提供するメトリクスと活用の仕方

（1）「うれしさ」のわかるメトリクス一覧

前述の適用タイミングとメトリクスの選択指標からメトリクスの一覧（付表１）を下記にまとめた。この表は、「うれしさ」を感じるメトリクス、適用タイミング、対象領域と前述の「うれしい事」を関連付けさせたものである。この表と添付する事例を参考にすることで開発現場にとってうれしいメトリクス抽出のヒントになることを期待する。

（2）メトリクスの効果的な活用方法（＝メトリクス方程式）

メトリクス一覧にあるメトリクスは、１つだけで見たのではその効果や「うれしさ」がわかりにくい場合がある。この場合、複数のメトリクスを使って課題となっている箇所を複眼的な視点で見ることで、うれしさを得ることができる。

ある領域のメトリクスを決めて、合否判定に使うときは、着眼点の違うメトリクスを複数使うことで網羅性をカバーし、問題がなければ次のステップに移行する。

例えば、ソフト設計工程レビューを、①前回からの変更点、②仕様書とソフトウェアの網羅性、③第３者における指摘内容・件数　④テスト項目といったメトリクスを掛け合わせることで、以下のような複数の着眼点で見ることができ、網羅性が上がる。上記①②③④はそれぞれ、

【①要求のモレ】×【②機能上のヌケ】×【③制約条件】×【④機能・性能上のモレ】

このように「**網羅性＝①×②×③×④**」というメトリクスの方程式を組むことで「うれしさ」の相乗効果が期待できると考える。次に、これらのメトリクスを有効に活用していくための考慮ポイントについて整理した。

また､メトリクスは時間的な概念として､それを使う間(ま)に合わせて使う事が大事である。例えば、フィードバックの時は、瞬間的な平均化していないメトリクスを使い、フロントローディングの時は、平均化したり特異点を示したりして傾向を見るなどの工夫がいる。

5.5. 「うれしさ」を現場に浸透させ、継続実施していくための考慮ポイント

「うれしさ」を継続するためには、メトリクスや複眼的な方程式を考えるだけでなく、それを継続的に実施していく工夫が必要である。

改善は、問題を「見える化」し、その問題の真因が何かを「言える化」、そして真因を見つけたら対処方法を考えて、問題を是正し効果を把握する「直せる化」のプロセスを踏んでいく。[2]

しかし、それだけでは現場に浸透・継続させていくことは難しく、浸透・継続していくためには、前述の『「見える化」・「言える化」・「直せる化」』に加えて、このプロセスの恩恵を受ける相手側(設計者自身や顧客)が「うれしい」と思わなくてはならない。すなわち、「見える化／言える化／直せる化」に加えて、それを享受する相手側(関係者)の「うれしい化」の考え方が必要となる(下図：「うれしい化」の構造)。

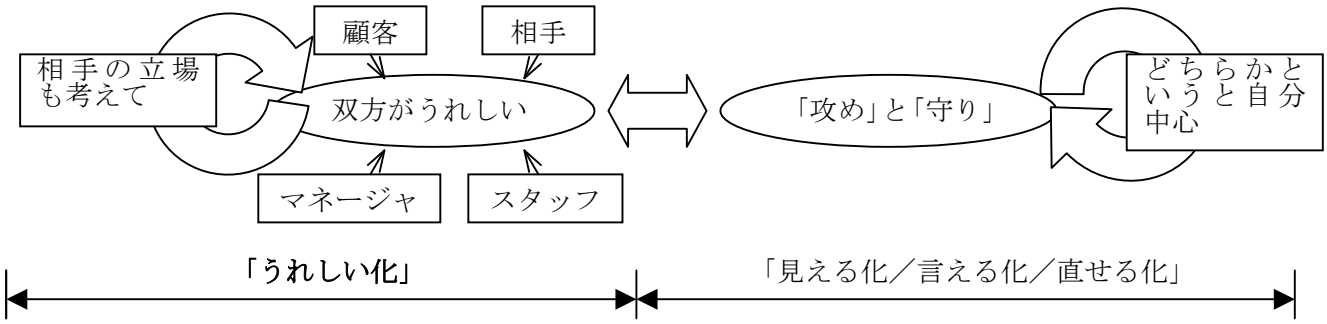


図３　「うれしい化」の構造

「うれしい化」は、「ビジネスゴール」に直結する。例えば、他社より先行して製品を出荷できた。開発費を低く抑えることができた、不具合のない製品が出荷できた、次のプロジェクトに繋げられた、などである。このように相手のことも考えることが「うれしさ」を継続、増大させるのである。

5.6. 活用事例の研究と改善工夫

題名：「見える化」が“現場のため”と思えるメトリクス収集

事例：

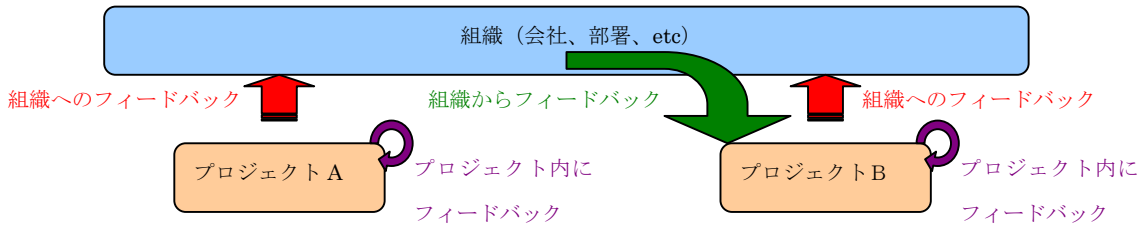


図４：組織とプロジェクトにおける成果(情報)のフィードバック

【活動】

自社の改善活動において、数ヶ月間にわたり問題・課題の抽出を行い、改善の対象として以下の２つを選定した。

①レビュー

②トラブル分析・再発防止

これらは自社のメンバおよび開発部隊の興味(逆に言うと危機感)が大きかったものである。

【計画】

改善を実施するタイミングとして、プロジェクト内で効果が出るメトリクス計測（作りこみフェーズ）と、プロジェクト終了後に他のプロジェクトに対して効果が出るメトリクス計測（再発防止フェーズ）の２つのタイミングに分けて考えることとした。(上記 図４参照)

メトリクス計測を“それぞれ別々に”収集するのでは、現場の負担が大きい。

よって、

効果の場所	メトリクス計測と利用の粒度
プロジェクト内	**その場で使えるメトリクス** ⇒即効性があり現場が“うれしい”
プロジェクト間(組織)	**その場で使えるメトリクスの累積 ＋ 観点の変更** ⇒知識・経験が累積され組織が“うれしい”

で対応し、１回のメトリクス計測で、見方、考え方を変えて、２回(それ以上)利用できれば、“メトリクス計測のための負荷増大感”を軽減することができると考えた。

今回の改善対象では、

■レビューを例にとれば、

短期（その場）	**レビュー指摘件数**から見るプロジェクトの傾向把握 ⇒バグの偏在性から、バグの早期発見
長期（再発防止など）	**レビュー指摘件数**の推移から見るプロジェクトの傾向把握 ⇒レビューチェックリストの拡充

■分析を例にとれば、

短期（その場）	当該プロジェクトの**問題発生の傾向把握** ⇒犯しやすい間違いの発生防止
長期（再発防止など）	**問題発生の傾向把握**の累積による再発防止対策立案 ⇒類似プロジェクトのモデル分解による品質予測

などの効果が期待される。

【実践するための工夫】

メトリクス計測と利用のタイミングを変えることで、“メトリクス計測や見える化は現場(自分たち)のためにならない”というプロジェクト担当者の疑念を払拭できるし、メトリクス計測結果を活用するタイミングを“分けて”考えたことで、現場用と組織用２つのデータの切り分けが楽になり、“現場が(に)うれしい”が実感できるのではないかと考える。

6. 目標達成度の評価

本研究を終えて、目標達成度は以下の通りである。

1) 目標１：現場の「うれしさ」を考察し、「うれしさ」の定義を考える

現場が「うれしい」とは何かを考察し、定義することで、メンバ間で「うれしさ」に関する認識を共有することができた。また、これにより、次のステップにスムーズに移行できた。

2) 目標２：現場に「うれしさ」を提供するための、メトリクスの活用方法を研究する
現場が「うれしい」定義に従って、どのようなメトリクスが提供できるかを考察し、その事例とメトリクスの活用方法についてもまとめることができた。

3) 目標３：現場が「うれしい」メトリクスの活用方法と実践事例を提示する
「うれしさ」を現場に浸透させるための考慮ポイントを議論し、「うれしい化の構造」としてまとめることができた。また、メンバが各社で試行中の「うれしい」メトリクス事例について、提示することができた。

7. 反省と今後の課題

今回の活動について反省点をまとめた。

良かった点は、いろいろな会社のメンバと、さまざまな角度から前向きな議論を行うことができ、内容の濃い分科会活動とすることができた。

反省すべき点は、メトリクスの有効活用を考察し実際に現場に適用することはできたが、「後工程のバグが減った」「〇〇の工数が削減できた」等、目に見える、具体的な成果を上げるまでにはいたらなかった。

・有効なフィードバックがある
・改善につながる
・個人のスキルアップにつながる
・簡単に採れる

などの“うれしい”を見える化することはまだ途中段階である。

プロジェクトをうまく運営するには、プロジェクトで計測されたメトリクスを、組織が受け取り、それを他のプロジェクト（未来のプロジェクト）に生かす仕組みを整備することが重要である。

メトリクスが一人歩きせず、“現場が(に)うれしい”を得られるようにするには、プロジェクト内、プロジェクトから組織へ、組織からプロジェクトへの フィードバックの連携が大事だと考える。

今回の研究成果を現場に浸透させ、継続実施していくことにより、プロジェクトを成功に導くプロセス改善につなげていきたい。

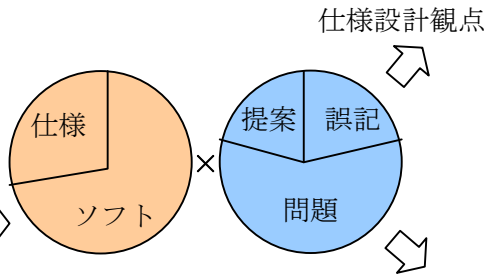
8. 参考文献

[1] 情報処理推進機構(IPA)，ソフトウェア・エンジニアリング・センター ：IT プロジェクトの「見える化」下流工程編，日経 BP 社，2006
[2] 第 22 年度ソフトウェア品質研究会第１分科会グループ，ソフトウェアプロセス評価・改善報告書，2007
[3] Lawrence H.Putnam and Ware Myers ： 初めて学ぶソフトウェアメトリクス，日経ＢＰ社，2005
[4] 遠藤功 ： 見える化－強い企業をつくる「見える」仕組み，東洋経済新報社，2005
[5] 今里健一郎・高木美作恵 ： 改善を見える化する技術，日科技連，2007
[6] Karl E.Wiegers ：ピアレビュー，日経 BP 社，2004

付録１．仕様書設計とソフト設計で交わされる「質問と回答」における分類指標

「質問＆回答書」のメトリクスを分析しその傾向や内容から設計にとっての「うれしさ」を抽出する。

質問	分類A		分類B			回答
	仕様	ソフト	問題	誤記	提案	
○分解能	○			○		分解能□に変更
○○単位変換		○		○		変換誤り
△方法→○方法	○				○	
：						
○○演算式		○	○			演算誤り



定性的な設計観点　　定量的な傾向　　定性的な設計観点　　複数メトリクスの掛け算

解説

ソフトウェアを作成する工程において、インプット情報である仕様書を作成する側とその仕様書をもとにソフトウェアを作成する側の間で交わされる「質問」が膨大な数にのぼった。設計の質を上げるべく「質問＆回答書」に分類Aと分類Bというメトリクスを設け、そのメトリクスの傾向を見える化し分析することで設計において「少し気をつければなくせる(誤記等)」、「十分に注意しないとなくせない(想定不足)」や「あいまいな内容」という設計の重要性や設計の観点を抽出する。これらの項目は、上図のように複数のメトリクスを掛け合わせ、分析することで抽出する。

実践効果と実践するための工夫

- その場でグラフとして「問題」・「誤記」・「提案」の傾向が見えるようにすることで「誤記」などケアレスミスをなくしていくことができた(その場でわかる「うれしさ」)。また、「問題」も顕在化させてその場で早く手が打てるようになった。
- 傾向が見えることで、例えば「誤記」が増えてきたら潜在的に重要な問題が潜んでいるのではないかという気づきを促すことができた(副作用的なうれしさ)。

また、このようにグラフ化し傾向を１回のプロジェクト毎に見えるように工夫して、その場で設計へフィードバックしたり、複数回のプロジェクトの傾向をまとめて見えるように工夫することで、次回プロジェクトへ活かすためのフロントローディングな改善に繋げたりすることができる。

付録２．各工程での摘出バグ数

No.	要求番号	機能仕様書		コード				
		仕様書名	摘出件数	モジュール名	摘出件数			
			レビュー		テスト			
					レビュー	単体	結合	システム
1	R00001	○○○機能		Class A				
				Class B				
				Class C				
				Class D				
				Class E				
2	R00002	○○○機能		Class F				
				Class G				
				Class H				
3	R00003	○○○機能		Class A				
				Class I				
				Class J				

解説

開発時やフィールドで発見されるバグを、より前工程で発見されるようにプロセスを改善していきたい。その為には、発見されたバグが本来どの工程で発見されるべきであったかを１件ずつ検討していく必要がある。しかし、現状の開発現場では、目の前の仕事をこなすのに精一杯で、「終わった開発を振り返ろう」という提案は、素直には受け入れられ難い。そこで、まずはその取掛りとして下表のようなメトリクスを採取し、この表から、特異な点(後工程に行くほどバグが増えているようなモジュール等)について議論を行い、現場の開発者に“改善が必要かも”という認識を広めていくことにした。

実践効果と実践するための工夫

プロセス改善の取り掛かりとして、改善の必要性を現場の開発者と共有するという意味において効果があった。また、この表が、コードの変更際どの部分をテストするかを判断するために使える等、前向きな意見も上がった。

今後は、摘出バグ数に発見された工程に応じて重み付けを行って（発見が遅くなるほど点数が大きくなる)、有益な情報が得られるか検証したい。

また、今回は比較的小規模な開発での試みであったため、手作業で表の作成を行ったが、規模の大きな開発時にこのような表を随時更新していくことは困難であるため、自動化を行っていく必要がある。

付録３．プロジェクト管理指標

プロジェクトの成功／失敗の予測のため、「プロジェクト管理指標」を用いる。プロジェクト管理指標は、以下の式で算出する。この場合の管理指標とは、プロジェクトの充実度を意味する。

プロジェクト管理指標 ＝ 計画充実度　×　進捗管理充実度　×　リスク管理充実度

それぞれの要素「計画充実度」、「進捗管理充実度」、「リスク管理充実度」は以下の通り算出する。

指標項目	指標項目	評価ﾀｲﾐﾝｸﾞ	評価点	
計画充実度	プロジェクトの目的が明確か？	計画作成時	xx 点	計 yy 点
	プロジェクトの概要が記載されているか？ ・終了時期 ・主なマイルストーン	計画作成時	xx 点	
	開発方針が明確か？	計画作成時	xx 点	
	開発体制が明確か？	計画作成時	xx 点	
	品質指標が明確か？	計画作成時	xx 点	
	マスタースケジュールがあるか？	計画作成時	xx 点	
	規模の見積りがされているか？	計画作成時	xx 点	
	リソースの見積りがされているか？	計画作成時	xx 点	
	開発環境の見積りがされているか？	計画作成時	xx 点	
	計画は適時見直されているか？	PJ 進行中	xx 点	
進捗管理充実度	進捗管理が計画されているか？ ・進捗管理のタイミング ・レポートライン ・報告内容 ・是正処置方針	計画作成時	xx 点	計 yy 点
	進捗管理が計画通り行われているか？	PJ 進行中	xx 点	
	是正処置が適切に行われているか？	PJ 進行中	xx 点	
ﾘｽｸ管理充実度	リスク管理計画がされているか？ ・リスク項目の抽出 ・リスクの評価 ・リスクへの対応策	計画作成時	xx 点	計 yy 点
	リスクが定期的に追跡されているか？	PJ 進行中	xx 点	
	リスクの再評価が定期的に行われているか？	PJ 進行中	xx 点	

解説

終了したプロジェクトの実績として、上記の指標を記録する。成功プロジェクト、および失敗プロジェクトを評価し、指標の傾向をバックデータとして蓄積する。

新規プロジェクトでは、定期的に上記指標を評価し、バックデータと比較することで、プロジェクト管理の状況を把握し、必要があれば是正処置を行う。

実践効果と実践するための工夫

現在進行中の計画の充実度について評価することができ、プロジェクト計画の充実度が「見える化」できた。試行中のため、最終的な成果は今後の活動による。

活用する上で、組織ごとに、評価する項目、および項目毎の点数について検討し、評価値が実態を反映するよう工夫する必要がある。

付録４．レビュー管理指標

レビューの指摘状況から成果物（基本設計書、詳細設計書）の品質評価を行う。また、レビュー品質についての評価も行う。

No.	日付	人数	開始時間	終了時間	工数人時	原因分類						指摘密度
						単純ミス	仕様ミス	設計ミス	標準化違反	問合せ	合計	件数／工数
1	1月25日	3	16:00	18:00	6.0	4	0	1	3	4	12	2.0
2	1月26日	5	17:00	18:00	5.0	0	0	2	1	0	3	0.6
3	1月27日	4	13:30	16:00	10.0	0	0	9	0	0	9	0.9
4	1月28日	4	11:20	12:20	4.0	4	2	10	2	0	18	4.5
5	1月29日	3	14:40	15:00	1.0	1	0	0	0	1	2	2.0
6	1月30日	3	14:20	14:40	1.0	1	1	1	0	2	5	5.0

原因分類

単純ミス　：誤字、脱字
仕様ミス　：前工程の設計書での曖昧表現、設計書内の矛盾
設計ミス　：現肯定の設計書での曖昧表現、設計書内の矛盾
標準化違反：用語の不統一など
問合せ　　：前工程の設計者への問合せ

解説

レビューを実施したときは、議事録にて参加者、開催日時、指摘項目の原因分類を行い記録として残す。議事録からの自動集計により上記のメトリクスを取れる仕組みにしておく。

評価を行うためには指摘密度（件数／レビュー人時）、（件数／規模）などの指標が必要である。

原因分類では次のことが判断できる。

- 単純ミス、標準化違反が多いときは、設計書のセルフチェックを徹底させる。
- 仕様ミス、問合せが多いときは、前工程の設計書を再度評価する。
- 設計ミスが追いときは、設計者の理解度を評価する。

実践効果と実践するための工夫

メトリクスの集計までは行えるが、数字から品質の評価を行い、対応策を立案するのに苦労していた。また、メトリクスでの評価は工程の初期から実施しないと有効な対策を実施できない。

評価を行うためには必ず指標が必要になる。現場、組織として指標を持っていないときは、現場として実績を積上げて指標を作るとの考えではなく、一般に知られている指標を持ってきて評価を行うのも一つの方法と考える。メトリクスを取るときは、漠然と集計を行い分析、評価を行うのでは難しい作業になってしまう。メトリクスを取る目的を明確にして、仮説を立てておくことも必要である。

- 一度のレビューで指摘件数が XX 件を超えたときは、レビューを中断する。
- 指摘密度がある値の範囲外では原因の追究を行う。
- 単純ミス、標準化違反が XX 件を超えたときは、セルフチェックの徹底度を確認する。

付録５．基準値を設けた品質管理

各工程で、“レビュー問題検出数”、“テスト項目数”、“バグ検出数”など「基準値」を設けて、ソフトウェア品質を定量的に管理する。

■モジュール別、テスト数、バグ数の集計表

○○システム

（単体試験）

No.	サブシステム	ID	処理名	試験規模	単体試験							単体分析評価		単体分析評価コメント
					机上計画	机上実績	試験密度	試験評価	単体バグ件数	バグ検出密度	バグ評価	総合評価	単体バグ収束率	
01	A00	A001	Aマスターメンテ											
		A002	Bマスターメンテ											
		A003	Cマスターメンテ											
		A004	A画面入力											
		A005	B画面入力											
		A006	C画面入力											
		A007	Aバッチ処理											
		A008	Bバッチ処理											
		A009	Cバッチ処理											

■品質管理シート

＊＊＊品質管理シート＊＊＊

顧客名:　　システム名:　　部門:

工数		人月	システム規模		Kステップ

	進捗率(%)	レビュー実施回数				問題検出数		未解決問題数
		社内		顧客		計画数	実績数	
		計画回数	実績回数	計画回数	実績回数			
要件定義								
外部基本設計								
内部基本設計								
詳細設計								

	進捗率(%)	テスト項目数			バグ検出数			未解決バグ数
		計画数	設定数	実績数	計画数	実績数	計画比(%)	
プログラム製造/検収（機能テストまで）								
結合テスト								
総合テスト								
本稼動後								

<基準値>

-レビュー問題検出数-			
■要件定義	0.5	／ページ	
■外部基本設計	0.3	／ページ	
■内部基本設計	0.2	／ページ	
■詳細設計	0.1	／ページ	
	-テスト項目数-	-バグ検出数-	
■機能テストまで	100	10	／Kステップ
■結合テスト	50	5	／Kステップ
■総合テスト	5	0.1	／Kステップ

解説：

品質のつくりこみ工程（要件定義～詳細設計）では、レビュー数、問題検出数、品質を確認する工程では、テスト項目数、バグ検出数をメトリクスとして、モジュール単位で集計する。

過去のプロジェクトから、レビュー問題検出数、テスト項目数、バグ検出数の基準値を設け、その数値との乖離を見ることにより、品質管理していく。

実践効果と実践するための工夫：

モジュール単位でメトリクスを取っておくことによって、１モジュールあたりの品質管理ができる。今まで、品質管理の“基準値”がなくて品質を“見える化”できていなかったが、テストの目安ができるなど、開発現場にとって“うれしい”メトリクス表となった。

また、システムとして問題が出たときに、バグの多かったモジュールが原因であることが多い。トラブルの原因究明時に、非常に役立つメトリクスであると考える。

付表１．開発現場に「うれしさ」のわかる「メトリクス」と「うれしさ事例」

No.	「うれしさのわかる」メトリクス	概要	適用タイミング（フロントローディング／フィードバック）	適用領域	うれしいことの実施行動				うれしさの事例	事例No.
					進行中プロジェクトがうまくいく	次プロジェクトがうまくいく	ノウハウが向上する	無駄時間が削減できる		
1	前回からの変更点	前回の仕様との変更点を洗い出す	フロントローディング	レビュー		○		○	仕様変更の把握度合いがわかる。	本文
2	仕様書とソフトウェアの網羅性	仕様書の内容のソフトウェアへの反映度合いを確認する	フロントローディング	レビュー		○		○	ソフトウェアの完成度合いがわかる。	
3	指摘内容	プロジェクト内のレビューにおける指摘内容を分析する	フィードバック	レビュー	○		○	○	プロジェクト内で、指摘内容が安易な間違いや深く考えを要する内容は設計が甘いことがわかる。また、指摘内容を把握することで設計の勘所がわかる。	
4	テスト項目	テスト項目と仕様を対比させることで仕様の抜け・誤りを抽出する	フィードバック	レビュー	○		○	○	仕様・ソフトウェア作成時にテスト項目も考えることで仕様の漏れや誤りを見つけることができる。	
5	指摘件数	プロジェクト内で、指摘件数の多い成果物、または箇所は要注意だとわかる	フロントローディング	レビュー	○		○		成果物をテストする前に、不適合が除去できる。	本文事例（5.6項）
6	指摘件数の推移	成果物の成熟度の評価ができる	フロントローディング／フィードバック	レビュー	○	○	○		成果物の残存不適合数の予測ができる。	
7	指摘件数のプロジェクト毎集計	成果物に対する指摘件数の予測ができ、レビューを効率化できる	フィードバック	レビュー		○	○	○	プロジェクト、成果物に対して、レビュー時の指摘件数の予測ができ、どれだけレビュー時間が必要かがわかり、効率化できる。	
8	トラブル分析	プロジェクト内で、不適合の発生原因、作りこみ原因を分析する	フロントローディング	テスト後	○		○		20対80の法則から、不適合が多く含まれそうな部分を予測できる。	
9	分析結果の集計	不適合の発生傾向を予測する	フィードバック	テスト後		○	○	○	プロジェクト、成果物に対して、トラブル混入の傾向がわかり、レビューのチェックリストに結果を反映できる。	
10	仕様の誤り	仕様／ソフトウェアのどちらの誤りであるか区別する	フロントローディング／フィードバック	レビュー	○	○	○		仕様／ソフトウェアの誤りを区別することでどちらの工程に課題があるのかを判断することができる。	付録１
11	ソフトウェアの誤り									
12	誤記	ケアレスミス指摘、解釈などの誤り確認、別の方法の提案などによる設計品質向上を分けることでその成果物の質がわかる	フロントローディング／フィードバック	レビュー	○	○	○		誤り度合いがわかることにより要求の解釈不足なのかケアレスミスなのかがわかる。また、誤記が多い場合は見直し時間の不足も推測できる。提案があるときはそれがノウハウに繋がることや考え不足が推測できる。	
13	問題									
14	提案									
15	各工程での摘出バグ件数	レビュー、テストの各工程で摘出されたバグ数を抽出する	フロントローディング	テスト後		○			工程別のバグ数の傾向から、テスト、レビュープロセスの改善点を見出せる。	付録２
16	プロジェクト管理指標	プロジェクト管理の状況を指標として評価できる	フィードバック	プロジェクト管理	○		○		プロジェクト管理の状況をチェックすることで、プロジェクト管理に起因する混乱を防ぐことができる。	付録３
17	指摘密度（件数／人時）	レビューの品質として評価する	フィードバック	レビュー	○		○		指摘件数が少ないときは、レビューが正しく運用されていない。メンバーが適正でない。	付録４
18	指摘密度（件数／規模）	規模当りの件数を評価する	フィードバック	レビュー	○		○		成果物の品質を評価する。	
19	原因分類（誤記、標準化違反）	設計書のケアレスミスを把握する	フィードバック	レビュー	○		○		仕事の質、理解度を評価する。	
20	原因分類（仕様ミス、問合せ）	背経書の完成度を把握する	フロントローディング	レビュー	○		○		前工程の終了基準を評価する。 前工程で本来行うべき事を行っていない。	
21	プロジェクト管理指標	テスト項目数、バグ検出数、など基準値を設けて、ソフトウェアの品質を定量的に管理する。	フロントローディング／フィードバック	プロジェクト管理	○	○	○		上流工程から、下流工程まで定量的にソフトウェアの品質管理ができる。	付録５