

## 78

# ソフトウェア開発組織の効率的な品質改善に向けた プロセスデータの活用<sup>1</sup>

## 1. 概要

本編では、開発中の品質や生産性に関するデータ（以降、プロセスデータ）を使用して、ソフトウェア品質の良否に影響を及ぼす要因について分析した事例を紹介する。

ソフトウェア開発組織において、プロセス改善に取り組み、CMMI(Capability Maturity Model Integration)の成熟度レベルが向上した場合には、それに伴って品質や生産性などの数値結果が向上することが期待される。実際に、CMMI に基づく改善の効果として、コスト面でのROI改善、見積り精度向上、失敗率の低減など、多くの事例が報告されている。しかしながら、結果としての品質や生産性の良否に影響を及ぼす要因について、プロセスデータに着眼して効果を分析した研究事例は少ない。

当社は、SW-CMM および CMMI に基づくプロセス改善に継続的に取り組んでおり、現在社内には、成熟度レベル 1 から 5 までの多様な開発組織がある。また、ソフトウェア品質会計と呼ばれる定量的管理の仕組みを 30 年以上にわたって展開しており、成熟度レベルの低い組織であっても一定の定量的管理が行われている。

このような背景から、全社横断的に収集したプロセスデータ（異なる成熟度レベルの、複数組織のデータ）を使用することにより、成熟度レベル別の分析を実施し、ソフトウェア品質の良否に影響を及ぼす要因が成熟度レベルによってどのように変化するかについて考察した。

## 2. 取り組みの目的

一般的に、CMMI の成熟度レベルを 1 段階上げるためには、多大なる労力と期間を要する。成熟度レベルを上げるための施策は、組織固有の課題や状況によって異なる。しかし、それぞれの成熟度レベルにおいて、最終的なソフトウェア品質の良否に影響する要因を把握することができれば、品質向上に向けた改善活動を効率的に進めることができる。

当社では、ソフトウェア開発プロジェクトのプロセスデータを全社横断的に収集、蓄積している。これにより、全社的なプロセス実績ベースラインを示すことが可能であると同時に、全社共通の強み・弱みを分析することが可能となる。本事例では、開発現場でのマネジメントや改善に

---

<sup>1</sup> 事例提供：日本電気株式会社 柳田 礼子 氏

活用できる知見をフィードバックすることを目的として、各成熟度レベルにおけるソフトウェア品質の要否に影響を及ぼす要因について分析した。

### 3. 取り組みの内容

#### 3.1. 分析対象

分析対象としたプロジェクトの開発プロセスは V 字モデルに基づいており、当社の標準的な品質管理システムである品質会計<sup>2</sup>を適用している。品質会計の考え方の根本には、「バグは作りこまない、作りこんだバグはすばやく摘出する」という考え方がある。基本設計工程からコーディング工程までを上工程、単体テスト工程からシステムテスト工程までをテスト工程と呼んでおり、特に上工程のレビューでバグを摘出することを重視している。

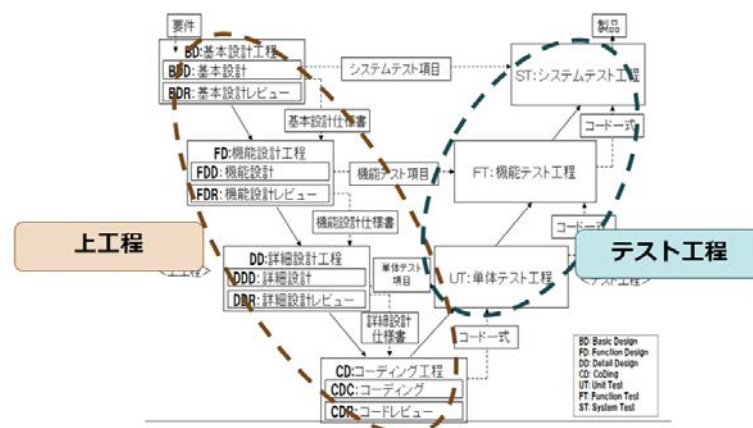


図 78-1 V 字モデル

分析対象としたプロジェクトは、2013 年度に開発を完了した 716 件である。プロジェクトのデータは「プロジェクト特性」情報も収集している。「プロジェクト特性」では、特定顧客向けのソフトウェア開発を「SI 系」、特定の顧客に特化しない汎用ソフトウェア開発（組込機器で動作するソフトウェアを含む）を「開発系」と定義している。

また、本分析では、プロジェクトの所属する組織の成熟度レベル（Maturity Level : ML）をプロジェクトに割り当てて、レベルの層別に使用している。これらにより層別したプロジェクトの件数は下表(表 78-1)のとおりである。

<sup>2</sup> 品質会計：高品質なソフトウェアを開発するための NEC グループの品質管理手法

表 78-1 データ件数

成熟度レベル(ML)	プロジェクト特性	プロジェクト数
ML1	SI 系	528
ML2	SI 系	
ML3	SI 系	
ML1	開発系	188
ML2	開発系	
ML3	開発系	
合計		716

ソフトウェア品質の良否については、出荷後に摘出されたバグ数を開発規模で割った、出荷後バグ密度で判断している。各組織であらかじめ定めている出荷後バグ密度の基準値と比較して、基準値以下の出荷後バグ密度のプロジェクトを「達成」、出荷後バグ密度が基準値を超えるプロジェクトを「未達」として分類した。「達成」プロジェクトの件数をプロジェクトの合計件数で割った値を「達成率」とし、各成熟度レベルの達成率を算出したところ、SI 系、開発系ともに、成熟度レベルが向上するにつれて達成率も向上するという結果が示された。

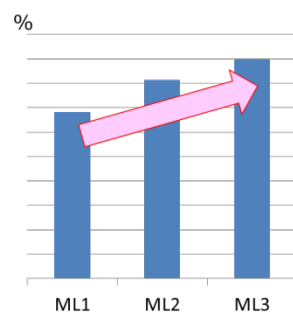


図 78-2 SI 系レベル別達成率

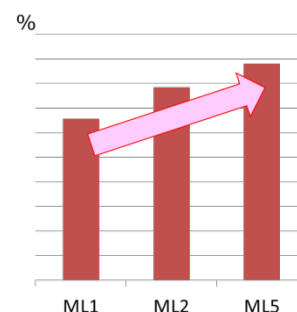


図 78-3 開発系レベル別達成率

いくつかの先行研究でも述べられているように、当社のデータでも、レベルが上がるほど達成率も高くなることがデータにより実証された。以降では、各レベルで、この達成率すなわちソフトウェア品質に影響を与える要因を詳細に分析する。同一プロジェクト特性のデータでレベル間の比較をするために、SI 系のデータのみを使用する、ただし、高成熟度レベルについては SI 系のデータがないため、ML5 のみは開発系のデータを使用している。

分析に使用したメトリクスは、品質会計制度で着目している主なメトリクスであり、欠損率が比較的低いものを選択した。各プロジェクトから収集して分析に使用したメトリクスは表 78-2 のとおりである。

表 78-2 分析に用いたメトリクス

No	メトリクス	単位	意味
1	開発規模（計画値）	ライン数	新規作成または変更予定の論理ソースコード行数（見積り段階の計画値）
2	開発規模	ライン数	新規作成または変更した論理ソースコード行数（実績値）
3	全工数	人時/KL	プロジェクトの全体工数を開発規模で割った値（いずれも実績値）
4	上工程工数	人時/KL	上工程の工数を開発規模で割った値（いずれも実績値）
5	レビュー工数	人時/KL	上工程のレビュー工数を開発規模で割った値（いずれも実績値）
6	テスト工程工数	人時/KL	テスト工程の工数を開発規模で割った値（いずれも実績値）
7	全バグ数	件/KL	開発中に摘出したバグ数を開発規模で割った値（いずれも実績値）
8	上工程バグ数	件/KL	上工程で摘出したバグ数を開発規模で割った値（いずれも実績値）
9	テスト工程バグ数	件/KL	テスト工程で摘出したバグ数を開発規模で割った値（いずれも実績値）
10	テスト項目数	件/KL	全テスト工程でのテスト項目数を開発規模で割った値（いずれも実績値）
11	上工程バグ摘出率	%	全バグ数に対する上工程摘出バグ数の比率（実績値）

### 3.2. 分析のステップ

分析のステップは、以下のとおりである。

- (1) 成熟度レベル毎に、出荷後バグ密度の基準値に対する「達成」/「未達」を目的変数としたロジスティック回帰分析を実施
- (2) 選択されたパラメータについて、選択された要因について深堀分析
  - (ア) 開発規模
  - (イ) テスト工程バグ数
  - (ウ) 上工程バグ摘出率、レビュー工数
  - (エ) レビュー工数、テスト工数、テスト工程バグ数
  - (オ) パラメータが選択されない要因

### 3.3. 分析結果

- (1) 成熟度レベル別のロジスティック回帰分析

各成熟度レベルにおいて、出荷後バグ密度の基準値に対する「達成」/「未達」を目的変数としてロジスティック回帰分析を実施した。分析に使用した変数は、表 78-2 の No1 を除く 10 個のメトリクスである。その結果、出荷後バグ基準達成状況に対する説明力が高いパラメータとして以下が選択された。

表 78-3 ロジスティック回帰分析で選択されたパラメータ

ML	パラメータ	係数	p 値
ML1	テスト工程バグ数/KL	- 0.170	0.064
	開発規模	- 6.07e-06	0.079
ML2	上工程バグ率	0.037	0.051
	レビュー工数/KL	- 0.021	0.086
	開発規模	- 7.15e-06	0.192
ML3	テスト工程工数/KL	- 0.012	0.023
	レビュー工数/KL	0.065	0.106
	テスト工程バグ数/KL	- 0.150	0.168
ML5	— (選択されず)	—	—

各成熟度レベルでは、出荷後バグ基準達成率も異なるが、その要因も異なるということが結果として示された。特に、成熟度レベル 1 および 2 では開発規模が選択され、係数がマイナスである。すなわち、成熟度レベルが低い場合は、開発規模が大きくなるとプロジェクトがコントロールできなくなり、品質リスクが高まると言える。

## (2) パラメータが選択された理由についての深堀分析

### (ア) 開発規模 (ML1、ML2)

開発規模が膨らむとプロジェクトの工数や工期へのインパクトがあると考えられるため、開発規模の計画値と実績値との差異を分析した。ML1～ML3 のレベルごとに比較すると、開発規模が計画値を超過するプロジェクトの割合は、成熟度レベルが向上するに伴い減少する傾向が確認された。特に ML1 から ML2 で大きく減少している。また、実績値が計画値を超過したプロジェクトの誤差範囲を、実績値を計画値で割った値として定義して、ML1～ML3 のレベルごとに比較した。20%以上の誤差、10%から 20%の誤差、10%未満の誤差で分類してそれぞれに分類されたプロジェクトの割合を算出したところ、誤差範囲が大きいプロジェクトの割合は、成熟度レベルが向上するにつれて減少する傾向が確認された。特に ML1 では、計画値を大きく超過する割合が高い。

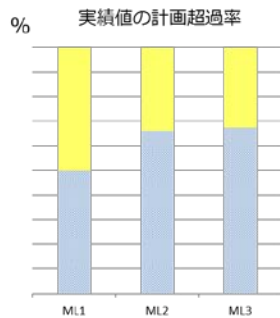


図 78-4 レベル別の計画超過率

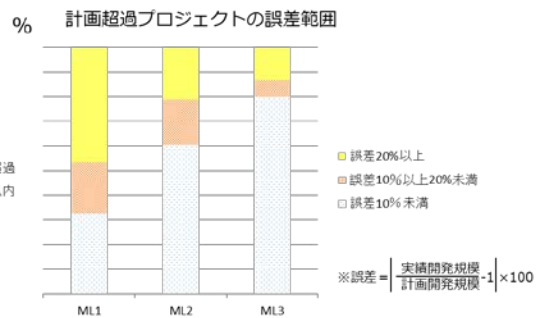


図 78-5 レベル別の誤差範囲

開発規模の変動要因は、要件変更など外的要因も複数考えられる。しかし、外的要因については各成熟度レベルで条件は同じであり、ML1 の組織だけに条件の悪いプロジェクトが多いわけではない。従って、ML1 の組織においては、プロジェクトのコントロールができておらず、見積りを大きく超過するケースが多いと考えられる。

#### (イ) テスト工程バグ数 (ML1)

ML1 ではテスト工程バグ数が選択されたため、テスト実施量との関係について、「達成」プロジェクトと「未達」プロジェクトの差異を確認した。

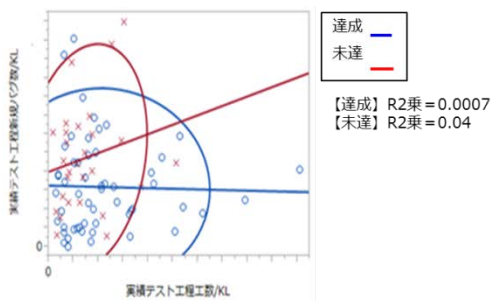


図 78-6 テスト工程工数とバグ数

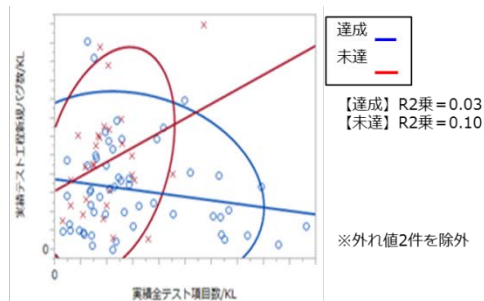


図 78-7 テスト項目数とバグ数

テスト工程の工数とバグ数、テスト項目数とバグ数の回帰分析では、いずれも「未達」の回帰直線が「達成」と比較して傾きが大きく、テスト実施量が増加した場合により多くのバグが摘出される。一方、「未達」の 90%確率楕円は、いずれのグラフでも「達成」と比較してグラフの左側に縦長に描かれている。すなわち、少ないテスト実施量で多くのバグが摘出されている。テスト開始時点に潜在バグが多く残存しているため、少ないテスト量で多くのバグが摘出されていると考えられる。「未達」は、上工程で品質確保されずテスト工程でバグが多く摘出されるうえ、テストも十分に実施されていない。

## (ウ) 上工程バグ摘出率、レビュー工数 (ML2)

ML2 では、レビュー工数と、レビューでのバグ摘出数に関する上工程バグ摘出率が選択された。上工程のレビュー工数とバグ数の散布図で確認したところ、「達成」と「未達」の 90% 確率楕円はほぼ同範囲であった。レビュー効率（レビュー工数に対するバグの摘出数）という観点では、ほぼ同等のレビューを実施していると考えられる。

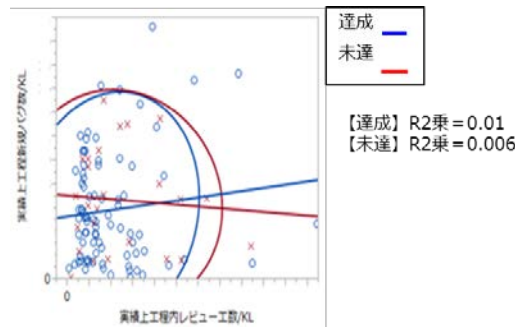


図 78-8 レビュー工数と上工程バグ数

ただし、中央値で比較すると、レビュー、テストの工数とバグ数について、「未達」は「達成」に比べて共に大きい値となっている。

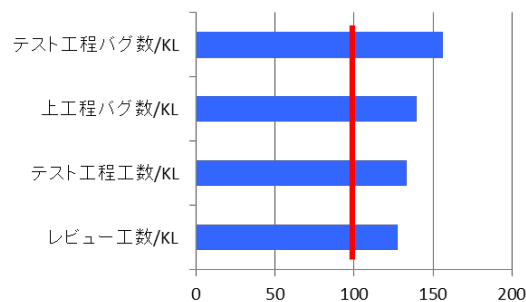


図 78-9 中央値比較（「達成」の中央値を 100 としたときの相対値）

「未達」では、実施量としては、レビューもテストも十分に実施しているが、バグも多く摘出されており、結果として「未達」となっていることから、バグを摘出しきれていないと考えられる。すなわち、レビュー品質が悪く、レビューで工数をかけてバグを多く摘出しても、テスト工程でもバグが多く摘出されて工数がかかっていると考えられる。

## (エ) レビュー工数、テスト工数、テスト工程バグ数 (ML3)

上工程工数とレビュー工数、レビュー工数と上工程バグ数の両散布図において、「未達」の 90% 確率楕円は、「達成」と比較すると、グラフの下側（縦軸の低い値）に位置しており一定のレビュー工数、一定のバグ数以下に分布している。

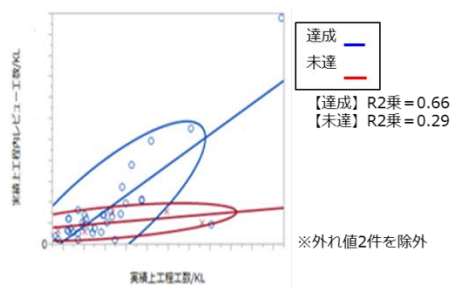


図 78-10 上工程工数とレビュー工数

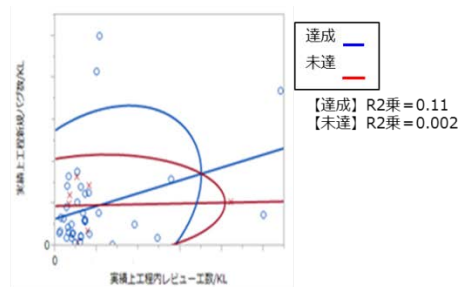


図 78-11 レビュー工数と上工程バグ数

通常は、設計の工数が多くなればレビュー工数も増え、レビュー工数が多ければバグも多く摘出されるが、「未達」はそのような結果になっていない。上工程工数が多くなってもレビュー工数は増えておらず、レビュー工数が多くなってもバグ数は増えていない。また、このような状況に対して適切な対応が取られていないと考えられる。すなわち、「未達」は、設計にかかる工数、摘出されるバグ数に応じたレビュー投入量の管理など、実績に応じたマネジメントが実施されていない。

テスト工程の各種メトリクスの中央値を「達成」と「未達」で比較すると、上工程できちんとマネジメントされなかった結果として、テスト工程に大きな差が出ていることが確認できる。

テスト工程の状況を詳細に分析するため、表 78-4 に示すメトリクスを用いて、工程別の値を比較する。なお、本分析では、UT～ST 工程のデータに欠損のないプロジェクトデータのみを使用する。

表 78-4 工程別分析に用いたメトリクス

メトリクス	単位	意味
工程別テスト工数/KL	人時/KL	UT, IT, ST 工程毎の、各テスト工程工数を開発規模で割った値
工程別テスト項目数/KL	件/KL	UT, IT, ST 工程毎の、各テスト工程のテスト項目数を開発規模で割った値
工程別テストバグ数/KL	件/KL	UT, IT, ST 工程毎の、各テスト工程で摘出したバグ数を開発規模で割った値

※UT：単体テスト IT：結合テスト ST：システムテスト

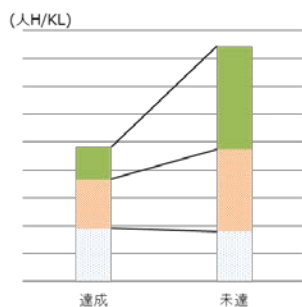


図 78-12 テスト工程工数

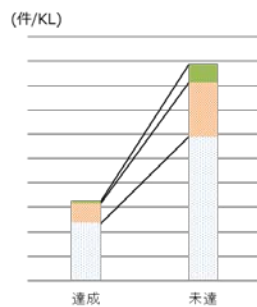


図 78-13 テスト工程バグ数

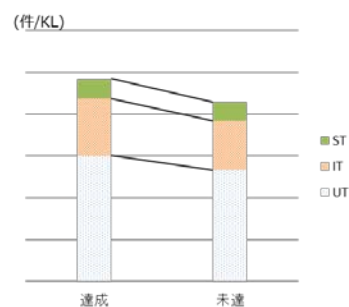


図 78-14 全テスト項目数



「達成」と「未達」の各種データを比較すると、特に ST 工程での差異が明らかである。「未達」の ST 工程の工数は「達成」の約 3 倍、バグ数は約 7 倍となっている。一方、ST 工程におけるテスト項目数には大きな差異はない。ほぼ同量のテストを実施しても、「未達」は ST 工程に至るまでに品質が確保されていないため、テストでバグが多く検出され、工数が多くかかっていると考えられる。

上工程の分析結果も考慮すると、「未達」は、開発中に実績を基にしたインプロセスのマネジメントが実施されず、ST 工程までバグが残存して ST 工程の工数が増大していると考えられる。

(オ) パラメータが選択されない理由 (ML5)

ML1~3 の分析で確認された「達成」と「未達」との傾向の違いについて、ML5 のデータで同様に比較分析を実施したところ、以下のいずれにおいても顕著な相違は見られなかった。

- レビュー工数、上工程バグ数、テスト工程工数、テスト工程バグ数

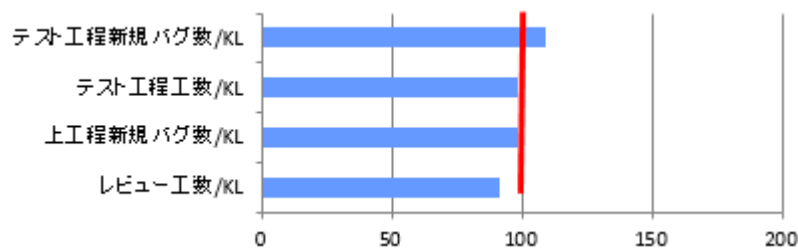


図 78-15 中央値比較（「達成」の中央値を 100 とした相対値）

- 上工程工数とレビュー工数、レビュー工数と上工程バグ数の散布図

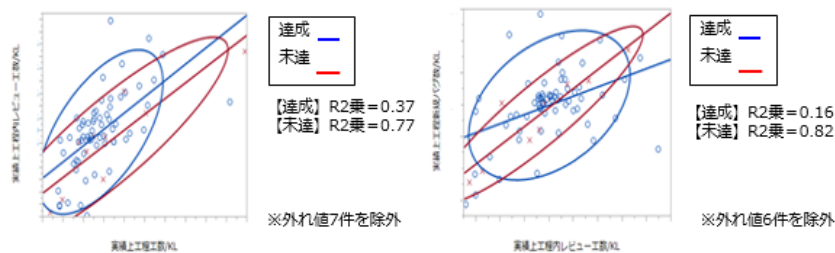


図 78-16 上工程工数とレビュー工数 図 78-17 レビュー工数と上工程バグ数

- テスト工程の工数、バグ数、テスト項目数（工程別）

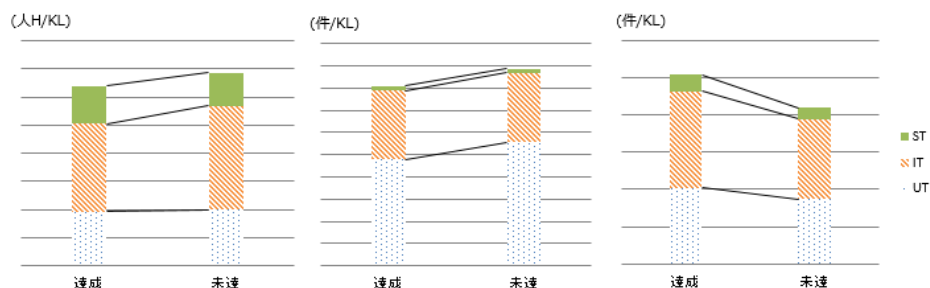


図 78-18 テスト工程工数 図 78-19 テスト工程バグ数 図 78-20 全テスト項目数

以上より、品質会計制度において収集している基本的なメトリクスでは「達成」と「未達」とでのプロセス上の相違点は見いだせなかった。これは、標準化が進んでいるため、より詳細なデータを確認しなければプロセス上の課題は見いだせない状況であると考えられる。なお、ML5 を達成している組織での出荷後バグ分析によると、「未達」となったプロジェクトの出荷後バグは、プロジェクト個別の細かな考慮漏れが影響しているとの結果が得られている。

### 3.4. 結果の考察

本分析により、ソフトウェア品質の良否に影響を及ぼす要因は、成熟度レベル毎に異なる結果となった。成熟度レベルの低い組織においては、開発規模が大きくなると品質を制御できなくなる傾向にあることを確認した。特にレベル1においては、上流の工程で品質が十分に確保されていないにもかかわらず、テストも十分に実施されていない場合に出荷後の品質を悪化させている。ソフトウェア開発の基本事項が徹底されていない状況が考えられる。レベル2においては、レビューやテストに十分に工数をかけていても、その実施内容に問題があり、十分に品質を確保できていない場合に出荷後の品質が悪くなる傾向がみられた。レベル3においては、プロセスデータの実績値を監視して適宜フィードバックするというタイムリーなマネジメントが機能していない場合に、出荷後の品質が悪くなる傾向があることがわかった。さらに、レベル5の組織においては、基準達成プロジェクトと基準未達プロジェクトとでプロセスデータの傾向について差異はほとんど見られず、安定したプロセスを保有していることがデータにより示された。

## 4. まとめ

今回の分析により、安定的に品質の良いソフトウェアを開発するためには組織のプロセス成熟度を向上させることが重要であるということがデータにより実証された。また、品質改善のために焦点を当てるべき改善のポイントは、各成熟度レベルで異なっていることが明らかになった。

た。さらに、効率的に品質改善を図るために焦点を当てるべきポイントは、CMMI の成熟度レベルで定義されたプロセス領域と密接に関連することが示された。

これらの知見をもとに、データ分析結果を基盤としてCMMIを活用することにより、成熟度レベルに応じた、効率的な品質改善活動を推進する所存である。

掲載されている会社名・製品名などは、各社の登録商標または商標です。

独立行政法人情報処理推進機構 技術本部 ソフトウェア高信頼化センター (IPA/SEC)