FUJIFILM
Value from Innovation

メトリクスでソースコードの美しさは判断出来るのか？

富士フイルムソフトウェア株式会社

アジェンダ

FUJIFILM

- コード品質管理活動(Code Quality Management =CQM)のご紹介
- (内部品質判断に関する)メトリクスと閾値の分析のご紹介
- 今後の展望 ～展開・発展・願望～

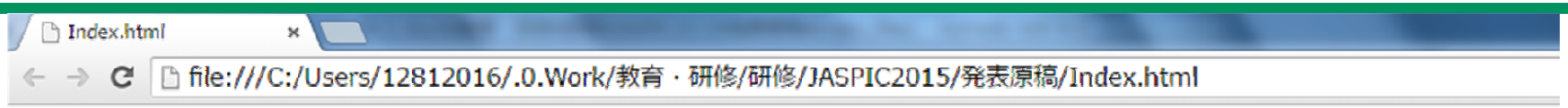
コード品質管理活動
Code Quality Management

CQM

のご紹介

コード品質管理活動CQMのご紹介

FUJIFILM



CQM

Code Quality Management Report

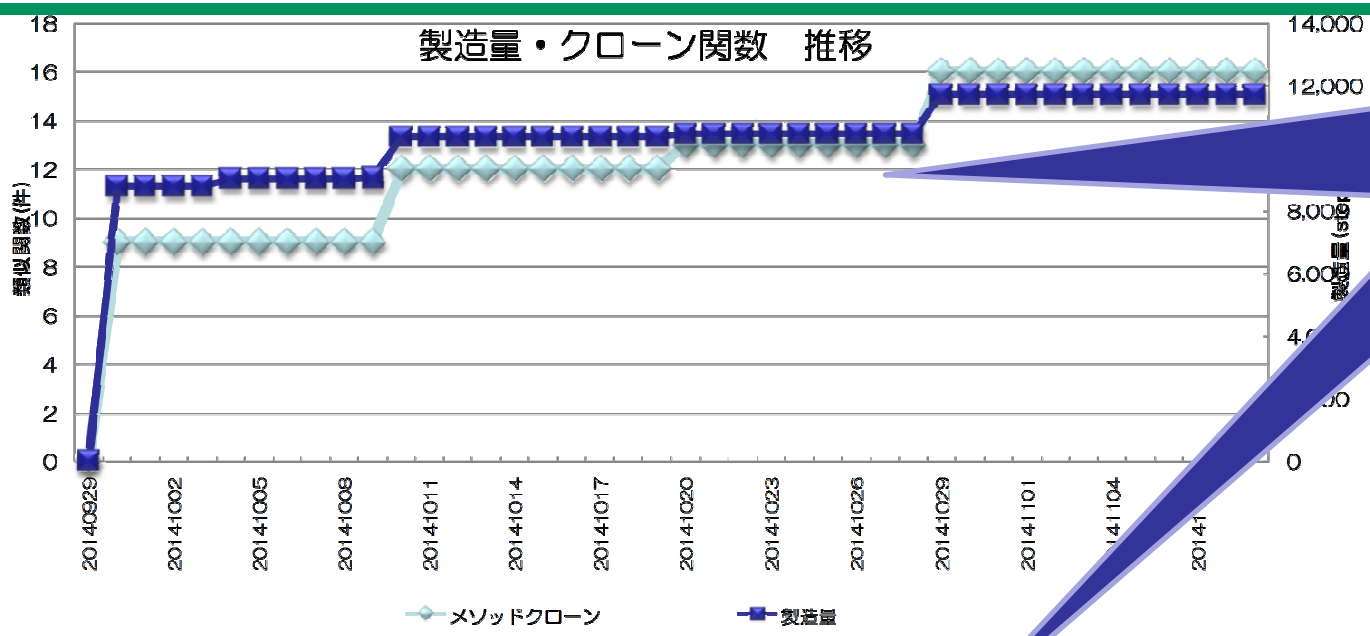
集計日付　YYYYMMDD

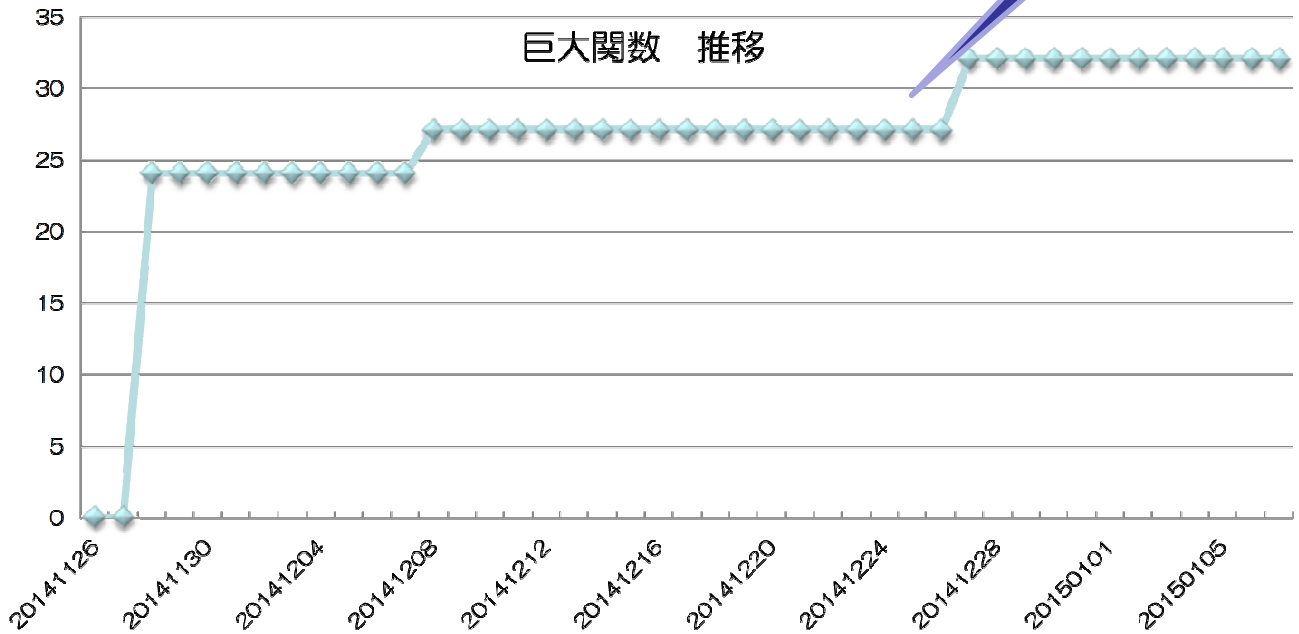
サブシステム	製造量 合計(step)	違反 合計(件)	クローン関数(件)					巨大関数(件)				
			小計	CPP	CS	Java	JS	小計	CPP	CS	Java	JS
SubSystem A	2,575	7	6			6		0			0	
SubSystem B	3,974	7	6				6	1				1
SubSystem C		100	10	10				2	2			
SubSystem D	1,213	0	0			0		0			0	
SubSystem E	632	2	2			2		0			0	
SubSystem F	1,530	11	10			10		0			0	
SubSystem G	186	7	2			2		0			0	
SubSystem H	0	0	0	0				0	0			

毎朝のソースメトリクス違反を報告するCQMレポートによって、
製造のリアルタイムでソースの内部品質を制御・維持・管理する。

あるプロジェクトでのCQM推移

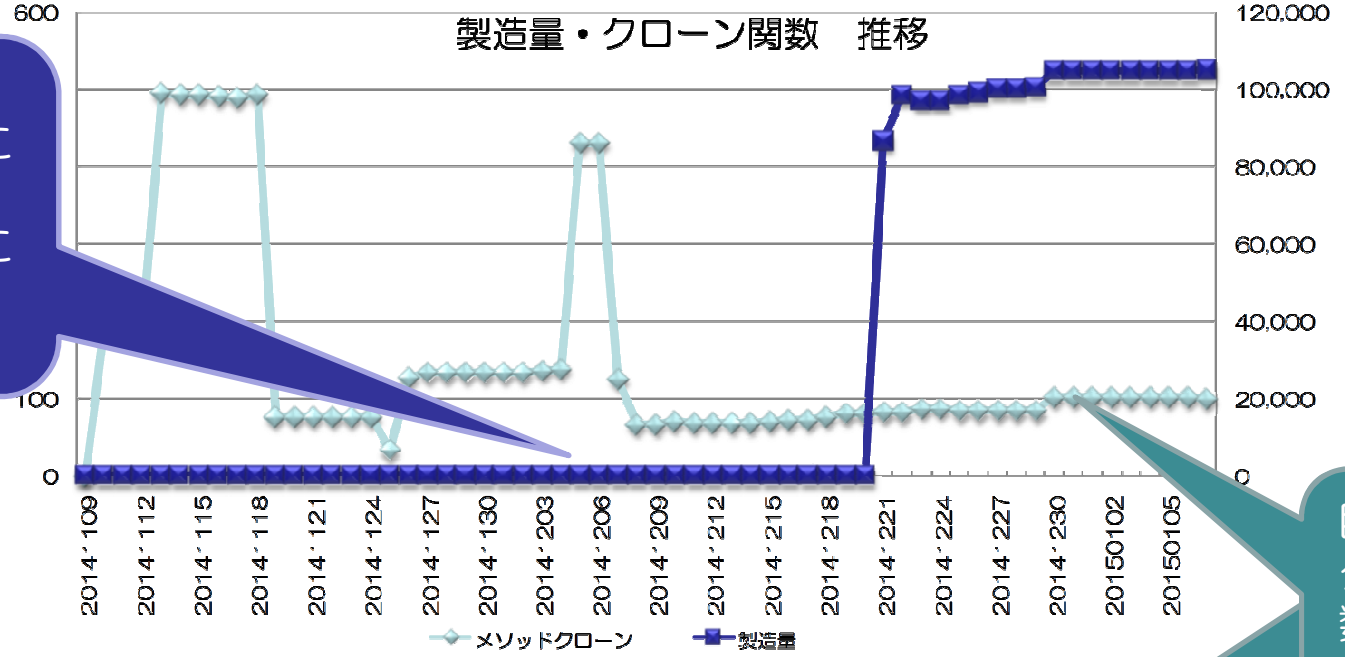
FUJIFILM



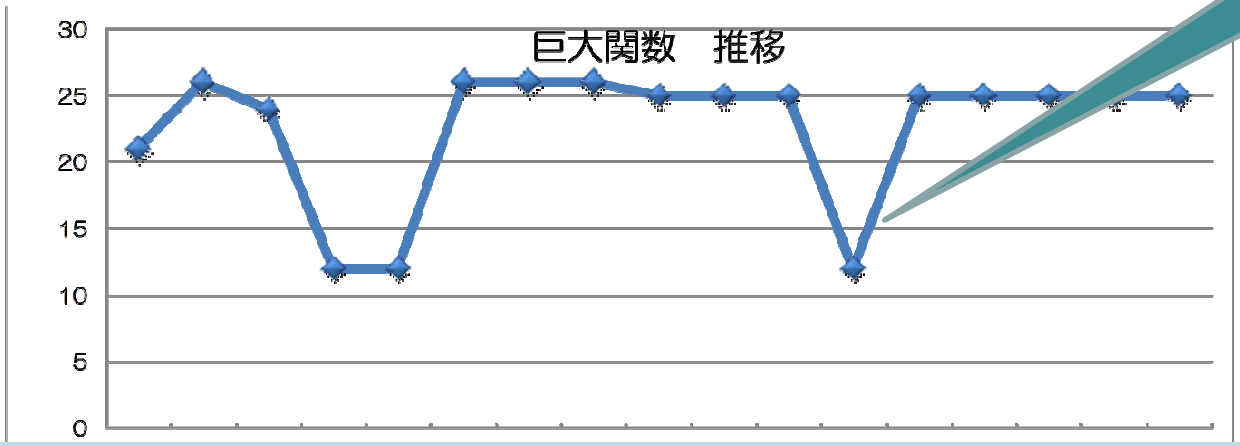


製造量、クローン、巨大関数の成長がシンクロしている

また、あるプロジェクトでのCQM推移



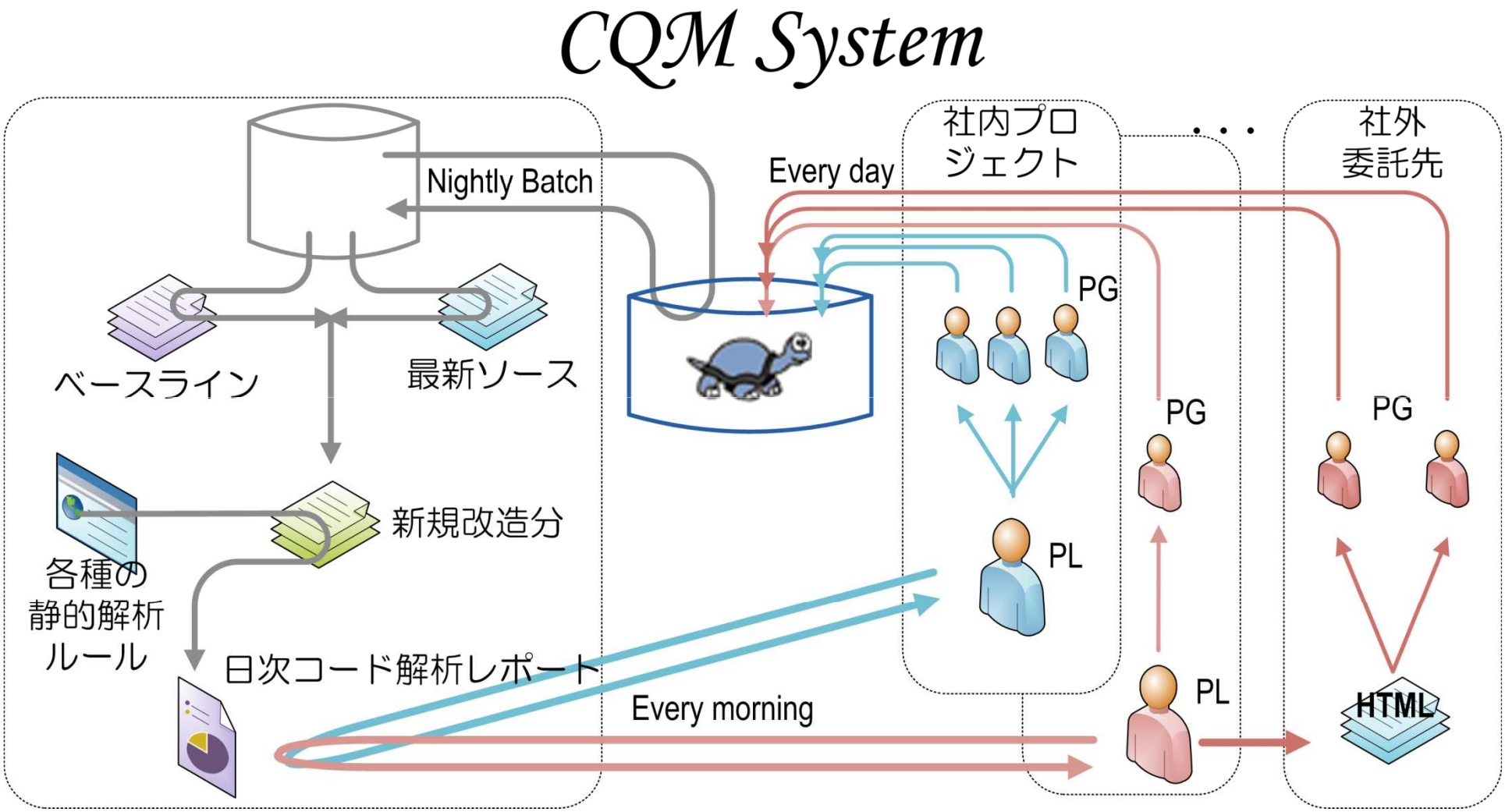
ソース管理に問題があり、データ収集に梃子摺った



巨大関数、クローン関数、はいずれも低水準で問題ない

いずれにせよ、一目瞭然に内部品質の変化が把握可能

システムを提供し、納品条件化



これにより、社内・外注全てで、製造の最初から最後まで、
ソースの内部品質をデイリーで監視、制御が可能

(内部品質判断に関する)
メトリクスと閾値の分析
のご紹介

(内部品質判断に関する)

メトリクスと閾値の分析のご紹介

FUJIFILM

- 現状
  - ＣＱＭにて以下のメトリクスを運用中
    - 巨大関数(100ステップ以上)が0件を目標
    - クローン関数(30行以上)が0件を目標
- 問題
  - それぞれの閾値の妥当性が未検証
  - これ以外のメトリクスが未検証
- 背景
  - 巨大関数、クローン関数およびそれらの閾値は、過去のコード検証サービスから経験的に妥当であったため、実質的な現場での品質向上を優先して採用・運用に入った
- 課題
  - 内部品質の判断基準は定量的に確立されていないため、閾値やメトリクスは理論的に特定出来ず、経験的な手法で特定する必要がある

課題に対する対策

FUJIFILM

対策

- 経験的な手法で特定するため、尺度の網羅性、データの客観性、評価の適正さを確保する。

網羅性の確保

・一般的な閾値とメトリクスだけでは網羅性が不十分な可能性があるため、なるべく多種類で集計した。

※ 複数の閾値と二次的な導出指標を組み合わせ、計266種類の指標で集計

客観性の確保

・社内データだけでは客観性を欠く危険性が有るので、社外の実績あるOSSのデータを併せて収集した。

※ 社内データ：13　OSSデータ：9　で集計

適正な評価

・閾値の妥当性/メトリクスの有効性の評価は、いくらでも匙加減で恣意的になるため、意図的に画一的な評価を実施した。

※ 巨大関数、クローンに関する閾値/メトリクスから絞り込み、それで説明が不足する場合は、その点を補完するメトリクスを更に特定して行った。
※ 集計結果は、数字の操作は行わず、一律の基準(+-1σ)で判断し、経験的な内部品質の良し悪しの納得感と適合するかどうかで評価した。

対策の具体的な集計手順

社内データの集計

- 社内の過去のプロジェクトのデータを収集する。
- メトリクスに複数の閾値を設定し、更に二次的な導出指標も含めて集計する。
- 内部品質の5段階主観評価を行う。
- 主観評価と相関するメトリクスを抽出する。

OSSデータの集計

- 社外の実績あるOSSのデータを収集する。
- メトリクスに複数の閾値を設定し、更に二次的な導出指標も含めて集計する。
- 内部品質の5段階主観評価を行う。
- 主観評価と相関するメトリクスを抽出する。

両者の突合

- 上記の両者を突き合わせ、双方に共通して相関するメトリクスを抽出する。
- 品質の判断基準を一律に+-1σとし評価する。
- 主観評価と整合するかを、巨大関数、クローンに関するメトリクスから絞り込み、それで説明が不足する場合は、その点を補完するメトリクスを更に特定する。

社内でもOSSでも使えるメトリクスは何か？

集計結果

：低品質判断基準より悪い

：高品質判断基準より良い

FUJIFILM

項目	低品質判断基準 +1σ	高品質判断基準 -1σ	Project A	Project B	Project C	Project D	Project E	Project F	Project G	Project H	ProjectI	Project J	Project K	Project L	Project M
			C/C++	C/C++	C++	C#	C++	C++	C++	C/C++	C++	C++	Java	Java,JS	Java
			408K	3.9M	994K	1.3M	619K	1.3M	3.1M	2.9M	870K	916K	420K	1.7M	530K
主観評価(5段階)			1	1	1	1	1	1	2	3	3	3	4	5	5
メソッド実効行数>100個数比率	2.45%	0.08%	1.13%	2.15%	3.77%	0.87%	1.81%	3.55%	0.63%	1.13%	0.67%	0.51%	0.20%	0.07%	0.00%
メソッドクローン個数比率	8.51%	1.89%	5.70%	8.65%	9.77%	2.73%	9.10%	8.89%	3.88%	8.88%	2.12%	1.96%	4.15%	1.72%	0.09%

• 集計結果から、巨大関数、クローンに関するメトリクスから絞り込み、低品質/高品質の判断基準(+-1σ)で区分したところ、以上のように主観評価ともなじむメトリクスを抽出することが出来た。

巨大関数とクローンの運用の妥当性は検証できた。

結論(その1)

- ＣＱＭにて運用中の以下のメトリクスの閾値は妥当である
  - 巨大関数(100ステップ以上)が0件を目標
  - クローン関数(30行以上)が0件を目標

残課題（その1）

：低品質判断基準より悪い
：高品質判断基準より良い

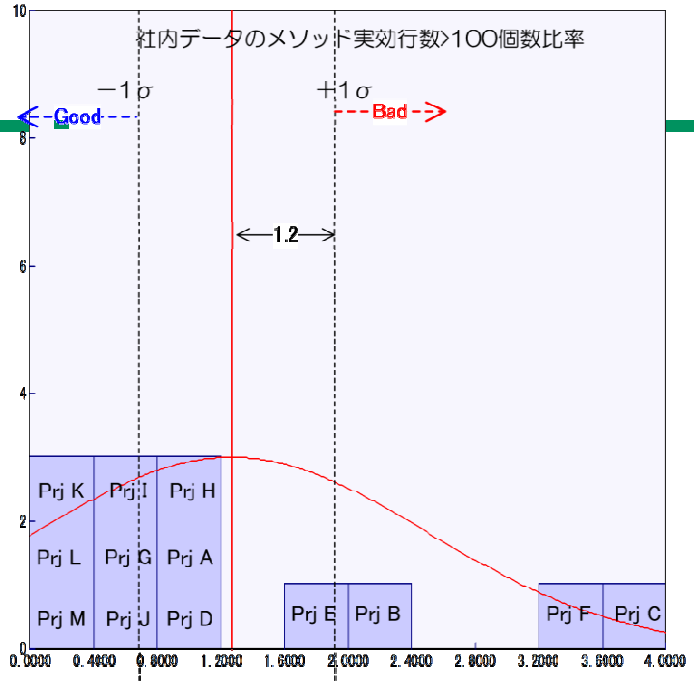
FUJIFILM

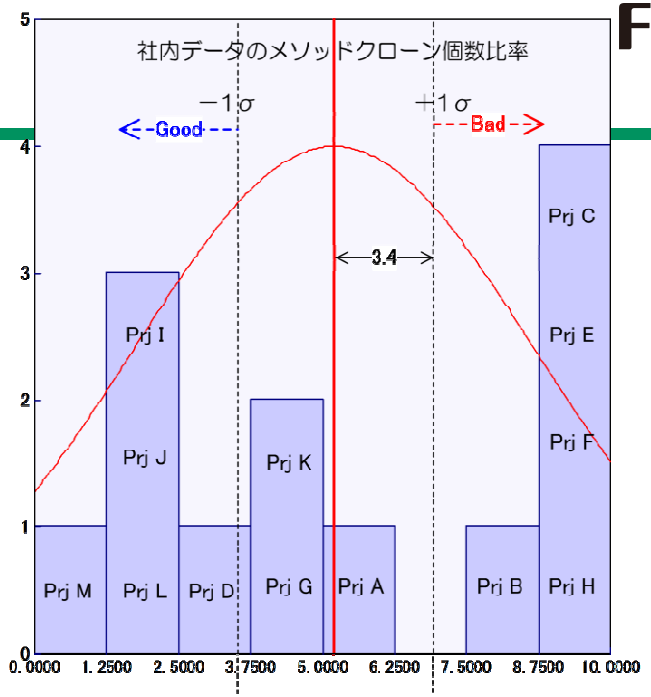
ただ、それだけでは説明できない課題も残った。

項目	低品質判断基準	高品質判断基準	Project A	Project B	Project C	Project D	Project E	Project F	Project G	Project H	ProjectI	Project J	Project K	Project L	Project M
			C/C++	C/C++	C++	C#	C++	C++	C++	C/C++	C++	C++	Java	Java,JS	Java
			408K	3.9M	994K	1.3M	619K	1.3M	3.1M	2.9M	870K	916K	420K	1.7M	530K
主観評価(5段階)			1	1	1	1	1	1	2	3	3	3	4	5	5
メソッド実効行数>100個数比率	2.45%	0.08%	1.13%	2.15%	3.77%	0.87%	1.81%	3.55%	0.63%	1.13%	0.67%	0.51%	0.20%	0.07%	0.00%
メソッドクローン個数比率	8.51%	1.89%	5.70%	8.65%	9.77%	2.73%	9.10%	8.89%	3.88%	8.88%	2.12%	1.96%	4.15%	1.72%	0.09%

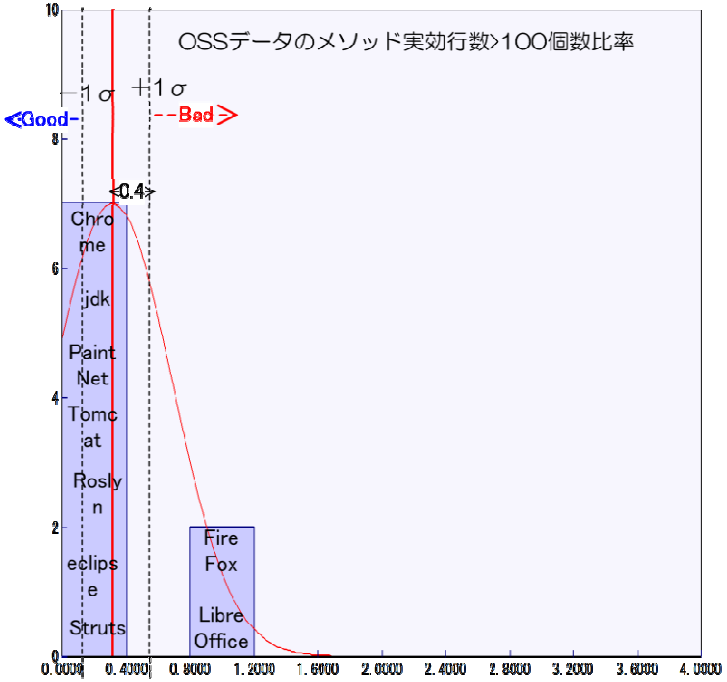
- 主観評価では低品質なのに、メトリクスの評価ではすり抜けるものがある。
- これを上手く説明できるメトリクスが、どうこねくり回しても社内データからは導けない。

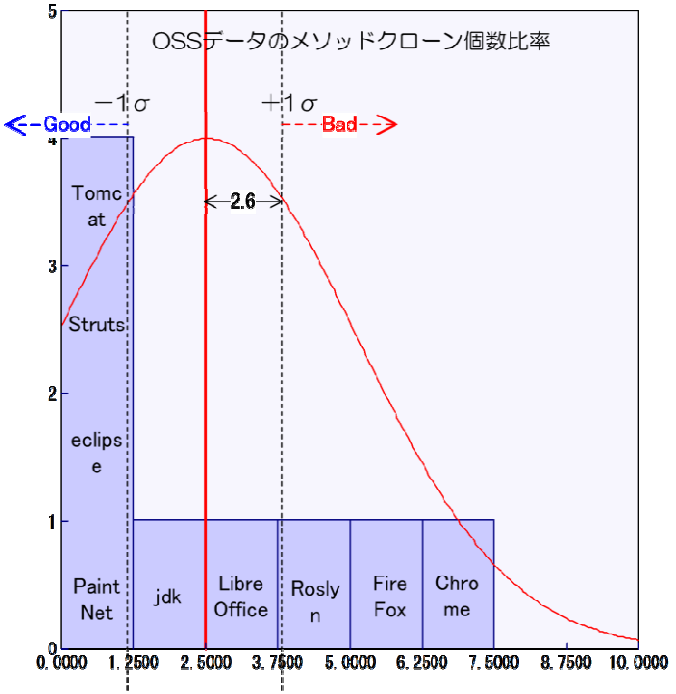
社内データ





OSSデータ





社内データの偏り（表）

：低品質判断基準より悪い
：高品質判断基準より良い

FUJIFILM

	項目	低品質判断基準	高品質判断基準	Project A	Project B	Project C	Project D	Project E	Project F	Project G	Project H	Project I	Project J	Project K	Project L	Project M
				C/C++	C/C++	C++	C#	C++	C++	C++	C/C++	C++	C++	Java	J,JS	Java
				408K	3.9M	994K	1.3M	619K	1.3M	3.1M	2.9M	870K	916K	420K	1.7M	530K
	主観評価			1	1	1	1	1	1	2	3	3	3	4	5	5
CQM	メソッド実効行数>100個数比率	2.45%	0.08%	1.13%	2.15%	3.77%	0.87%	1.81%	3.55%	0.63%	1.13%	0.67%	0.51%	0.20%	0.07%	0.00%
CQM	メソッドクローン個数比率	8.51%	1.89%	5.70%	8.65%	9.77%	2.73%	9.10%	8.89%	3.88%	8.88%	2.12%	1.96%	4.15%	1.72%	0.09%
肥大化	メソッド総行数150規模比率	35.00%	11.36%	27.85%	36.52%	40.12%	36.73%	27.30%	37.14%	12.09%	14.19%	18.06%	16.31%	7.49%	24.55%	2.99%
肥大化	メソッドLOC120規模比率	29.23%	9.02%	23.68%	29.65%	36.24%	28.67%	21.09%	33.29%	11.41%	12.69%	14.95%	11.41%	7.29%	15.77%	2.47%
肥大化	メソッド実効行数>100規模比率	28.08%	6.87%	17.25%	29.54%	35.80%	26.53%	22.18%	30.40%	10.48%	14.67%	17.29%	13.18%	7.72%	2.19%	0.00%
複雑度	メソッド複雑度>30規模比率	19.17%	3.01%	5.96%	18.51%	26.88%	5.17%	15.30%	23.27%	7.42%	9.39%	15.37%	12.18%	4.24%	0.53%	0.00%
複雑度	メソッド複雑度>20規模比率	28.91%	6.36%	14.35%	28.51%	37.92%	10.69%	23.03%	35.84%	13.60%	14.77%	22.11%	18.77%	8.03%	1.29%	0.37%
複雑度	メソッド複雑度>10規模比率	50.55%	16.56%	35.10%	49.07%	58.58%	24.99%	49.53%	58.24%	31.33%	32.60%	37.63%	32.97%	15.83%	6.27%	4.12%
ネスト	メソッド最大ネスト>2規模比率	38.00%	17.67%	41.65%	38.65%	34.62%	37.38%	22.54%	36.07%	40.12%	21.04%	23.05%	21.56%	19.74%	15.92%	9.50%
ネスト	メソッド最大ネスト>5規模比率	7.05%	1.47%	8.79%	6.37%	6.88%	8.90%	2.28%	4.25%	6.80%	2.01%	1.92%	1.78%	2.38%	2.57%	0.46%

・本来、巨大関数とクローンのメトリクスで、低いと判断されてもおかしくないものが、社内データの分布が悪い方に偏っているために判断をすり抜けたが、肥大化や複雑度はかなり高かった。
・ネストのメトリクスが複雑度の特徴を捉え、品質判断を補完したとも考えられるが、それは既に納得感とはかけ離れた数値となってしまう。

OSSデータの偏り

：低品質判断基準より悪い
：高品質判断基準より良い

FUJIFILM

本当に、社内データの分布が悪い方に偏っているかどうかを検証するために
OSSで同様の指標がどうなっているのかを見てみる。

	項目	低品質判断基準	高品質判断基準	FireFox	Libre Office	roslyn	chrome	jdk	Tomcat	eclipse	PaintNet	Struts
				C/C++	C++	C#	C	Java	Java	Java	C#	Java
				3.8M	2.9M	560K	3.2M	1.3M	220K	380K	67K	144K
	主観評価			1	1	2	2	4	4	5	5	5
CQM	メソッド実効行数>100個数比率	0.67%	-0.04%	1.06%	0.85%	0.08%	0.28%	0.21%	0.10%	0.08%	0.14%	0.04%
	メソッドクローン個数比率	4.99%	0.04%	5.81%	2.68%	3.77%	7.15%	2.37%	0.29%	0.20%	0.13%	0.23%
肥大化	メソッド総行数150規模比率	14.00%	3.30%	17.22%	18.47%	7.70%	6.60%	8.38%	4.45%	2.90%	9.11%	3.01%
	メソッドLOC120規模比率	13.21%	3.63%	13.94%	18.40%	8.36%	5.42%	10.04%	5.31%	3.08%	8.00%	3.28%
	メソッド実効行数>100規模比率	14.47%	1.78%	20.63%	18.12%	4.33%	7.56%	5.80%	4.26%	2.52%	8.13%	1.80%
複雑度	メソッド複雑度>30規模比率	13.05%	0.12%	19.05%	17.04%	1.55%	7.04%	5.86%	4.48%	1.51%	0.53%	2.21%
	メソッド複雑度>20規模比率	19.38%	1.58%	27.60%	24.79%	3.17%	10.92%	9.79%	8.20%	2.98%	2.93%	3.96%
	メソッド複雑度>10規模比率	34.01%	7.37%	45.47%	41.80%	8.11%	21.59%	21.39%	18.98%	7.30%	10.90%	10.69%
ネスト	メソッド最大ネスト>2規模比率	38.70%	13.32%	46.12%	48.85%	11.74%	20.59%	29.52%	25.67%	12.86%	22.32%	16.44%
	メソッド最大ネスト>5規模比率	7.31%	0.33%	8.09%	11.91%	0.95%	2.26%	3.41%	2.94%	1.60%	1.64%	1.59%

同じようにすり抜ける場合もあるが、データの分布としては納得感がある。
ためしにOSSの判断基準を社内データに当て嵌めると→次ページ

OSS基準での社内データの判断

：低品質判断基準より悪い

：高品質判断基準より良い

FUJIFILM

OSSの基準で社内データを判断すると以下の通り。

	項目	低品質判断基準	高品質判断基準	Project A	Project B	Project C	Project D	Project E	Project F	Project tG	Project H	Project tI	Project tJ	Project tK	Project tL	Project tM
				C/C++	C/C++	C++	C#	C++	C++	C++	C/C++	C++	C++	Java	J,JS	Java
				408K	3.9M	994K	1.3M	619K	1.3M	3.1M	2.9M	870K	916K	420K	1.7M	530K
	主観評価			1	1	1	1	1	1	2	3	3	3	4	5	5
CQM	メソッド実効行数>100個数比率	0.67%	-0.04%	1.13%	2.15%	3.77%	0.87%	1.81%	3.55%	0.63%	1.13%	0.67%	0.51%	0.20%	0.07%	0.00%
CQM	メソッドクローン個数比率	4.99%	0.04%	5.70%	8.65%	9.77%	2.73%	9.10%	8.89%	3.88%	8.88%	2.12%	1.96%	4.15%	1.72%	0.09%
肥大化	メソッド総行数150規模比率	14.00%	3.30%	27.85%	36.52%	40.12%	36.73%	27.30%	37.14%	12.09%	14.19%	18.06%	16.31%	7.49%	24.55%	2.99%
肥大化	メソッドLOC120規模比率	13.21%	3.63%	23.68%	29.65%	36.24%	28.67%	21.09%	33.29%	11.41%	12.69%	14.95%	11.41%	7.29%	15.77%	2.47%
肥大化	メソッド実効行数>100規模比率	14.47%	1.78%	17.25%	29.54%	35.80%	26.53%	22.18%	30.40%	10.48%	14.67%	17.29%	13.18%	7.72%	2.19%	0.00%
複雑度	メソッド複雑度>30規模比率	13.05%	0.12%	5.96%	18.51%	26.88%	5.17%	15.30%	23.27%	7.42%	9.39%	15.37%	12.18%	4.24%	0.53%	0.00%
複雑度	メソッド複雑度>20規模比率	19.38%	1.58%	14.35%	28.51%	37.92%	10.69%	23.03%	35.84%	13.60%	14.77%	22.11%	18.77%	8.03%	1.29%	0.37%
複雑度	メソッド複雑度>10規模比率	34.01%	7.37%	35.10%	49.07%	58.58%	24.99%	49.53%	58.24%	31.33%	32.60%	37.63%	32.97%	15.83%	6.27%	4.12%
ネスト	メソッド最大ネスト>2個数比率	38.70%	13.32%	41.65%	38.65%	34.62%	37.38%	22.54%	36.07%	40.12%	21.04%	23.05%	21.56%	19.74%	15.92%	9.50%
ネスト	メソッド最大ネスト>5個数比率	7.31%	0.33%	8.79%	6.37%	6.88%	8.90%	2.28%	4.25%	6.80%	2.01%	1.92%	1.78%	2.38%	2.57%	0.46%

悪いものは悪いと判断でき、基準としてはこちらを採用するべき。

結論(その2)

- ＣＱＭにて運用中の以下のメトリクスの閾値は妥当である
  - 巨大関数(100ステップ以上)が0件を目標
  - クローン関数(30行以上)が0件を目標
- ただ、品質判断の基準は偏りのひどい社内データではなくOSSが妥当である
  - OSSでの判断基準を採用する

今後の展望

～展開・発展・願望～

今後の展開と発展

- (展開)メソッド設計向上の継続
  - CQMの狙いに間違いはなかったので更に推進する
    - CQM管理の定着推進し、各部の独立度を向上させる
    - CQMマスター制度を確立し、メソッド設計能力を向上させる
- (発展)細かな単位での主観評価の収集
  - 今回の分析は製品全体の主観評価とメトリクスの整合性で、完成後しか評価できない
    - 日々のコーディングの品質を制御するためには、より細かい単位(ファイル、クラス、メソッド)での主観評価データが必要

→これがあれば有効な指標を統計的に割り出せると考える

ソースレビューで教師データを収集し、
ソースコードのビッグデータ解析

主観評価の例

良い(Good)、悪い(Bad)、普通(Average)の三段階評価(CQM Score)。

Type	悪い(Bad)	普通(Average)	良い(Good)
メソッド	似たコーディングが何度も出てくる。	似たコードがあってもそれほどでもない。	殆ど似たコードは出てこない。
	兎に角メソッドが長い。	長くても3~4スクロールくらいで収まる。	殆ど1~2スクロールに収まっている。
	引数、一時変数、returnが多い。	引数、一時変数、returnは多くても3~5程度である。	引数はオブジェクト化され、一時変数は殆どなく、returnは殆ど一つ。
	コメントがやたらと多い。	難しい処理に適切なコメントが付与されている。	適度なコメントと、更になくとも処理内容が理解できる。
	マジックナンバーが平然とある。	マジックナンバーがたまにある。	マジックナンバーがない。
クラス	兎に角クラスが大きい。	大きくても~2000行くらいで収まる。	殆ど500行程度に収まっている。
	変数、メソッドがやたらと多い。	変数<10、 メソッド<20程度。	一目で見渡せる程度に収まっている。

ご興味のある方、是非一緒にやりませんか。

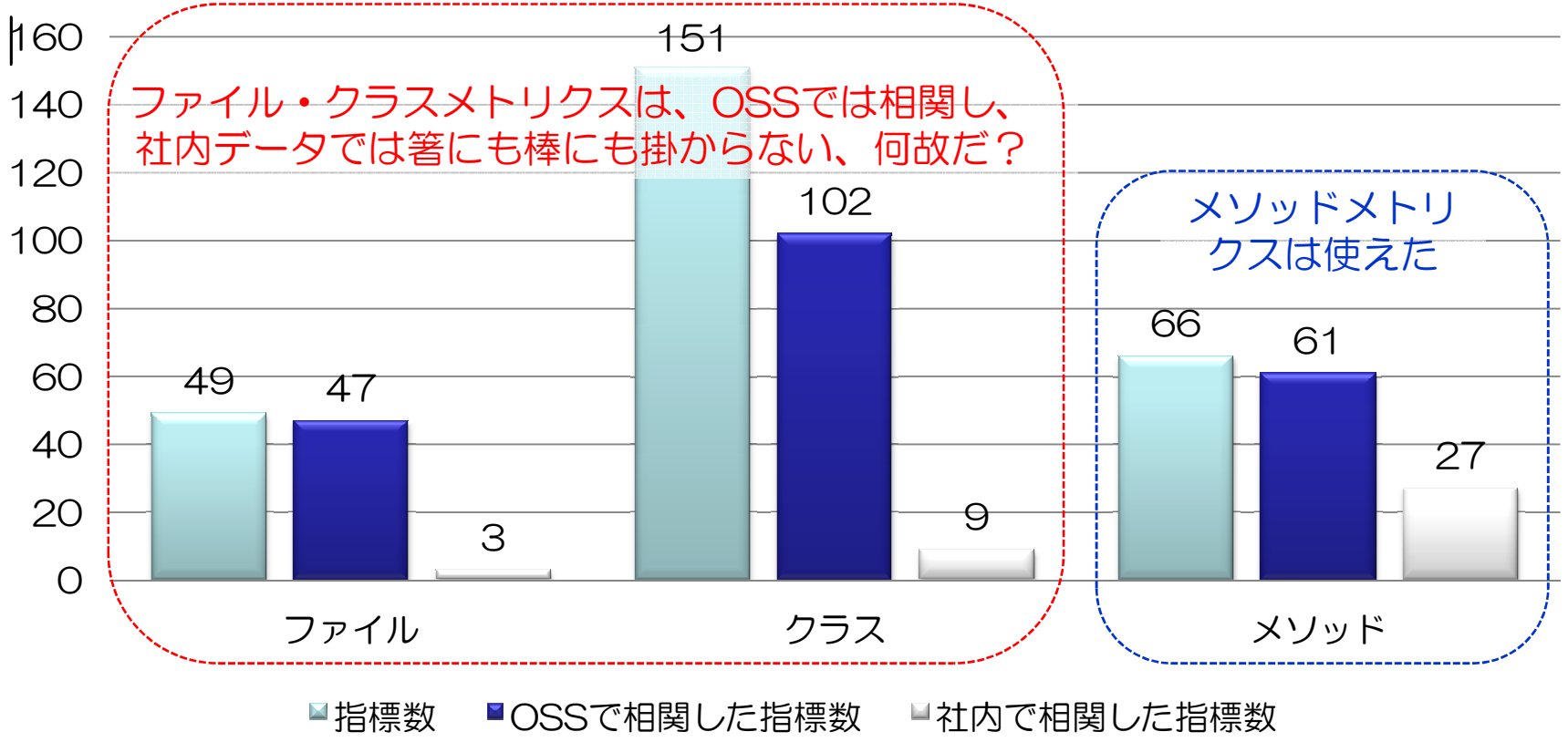
ソースコードはビッグデータだ！！！

一人でやるのは辛い
(><)

Appendix

ファイル・クラスメトリクスの現状

社内データ、OSSデータで、それぞれ相関した指標数は以下の通り。



・メソッドメトリクスは社内データでも使えた。

・しかし、クラスやファイルのメトリクスは、OSSでは使えそうなのに、社内データでは殆ど使えない。→何故だろう？

OSSとの違い（１）

FUJIFILM

変数名	社内Prj	OSS
CBO>100個数	-0.08	-0.756
CBO>100個数比率	-0.076	-0.738
CBO>100規模	-0.286	-0.714
CBO>100規模比率	-0.237	-0.738
CBO>50個数	-0.212	-0.934
CBO>50個数比率	-0.24	-0.767
CBO>50規模	-0.397	-0.799
CBO>50規模比率	-0.311	-0.586
CBO>20個数	-0.204	-0.755
CBO>20個数比率	-0.103	-0.285
CBO>20規模	-0.374	-0.814
CBO>20規模比率	-0.208	-0.216
CBO>5個数	-0.069	-0.77
CBO>5個数比率	0.233	-0.504
CBO>5規模	-0.36	-0.844
CBO>5規模比率	-0.315	-0.078

OSSは比率に
関係なく、
個数
規模
といった
絶対値に
相関している。

変数名	社内Prj	OSS
LCM>90個数	0.017	-0.498
LCM>90個数比率	0.34	0.138
LCM>90規模	-0.378	-0.809
LCM>90規模比率	0.315	0.081
LCM>70個数	0.002	-0.73
LCM>70個数比率	0.408	-0.038
LCM>70規模	-0.38	-0.828
LCM>70規模比率	-0.203	-0.004
LCM>50個数	0	-0.748
LCM>50個数比率	0.488	-0.216
LCM>50規模	-0.328	-0.826
LCM>50規模比率	0.334	-0.026
LCM>30個数	0.008	-0.764
LCM>30個数比率	0.566	-0.323
LCM>30規模	-0.324	-0.826
LCM>30規模比率	0.467	-0.001

・OSSでは、悪いものが絶対的に多ければ、比率に関係なく全体の評価も悪いという、考えてみれば非常に全うな相関をしている。
　つまり、大柄な人も小柄な人も、熱が高ければ風邪、と言う判断。

社内OODデータをOSSの上位と入れ替えた場合

FUJIFILM

項目	FireFox	Libre Office	roslyn	chrome	jdk	Tomcat	eclipse	PaintNet	Struts
	C/C++	C++	C#	C	Java	Java	Java	C#	Java
	3.8M	2.9M	560K	3.2M	1.3M	220K	380K	67K	144K
主観評価	1	1	2	2	3	4	5	5	5

新目のフレームワークを導入しているOODのPrjならば、ファイル・クラスメトリクスとの相関もあると想定し、OSSの上位に入れ替えて集計をトライ

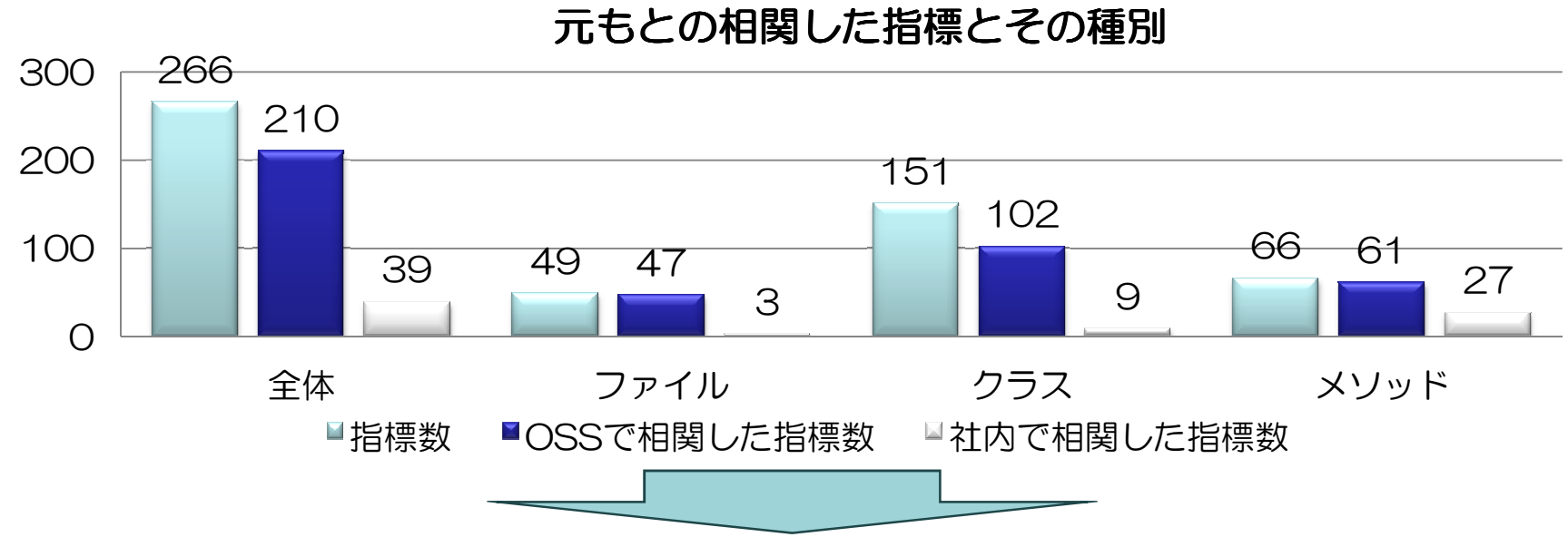
上位４，５を入れ替え

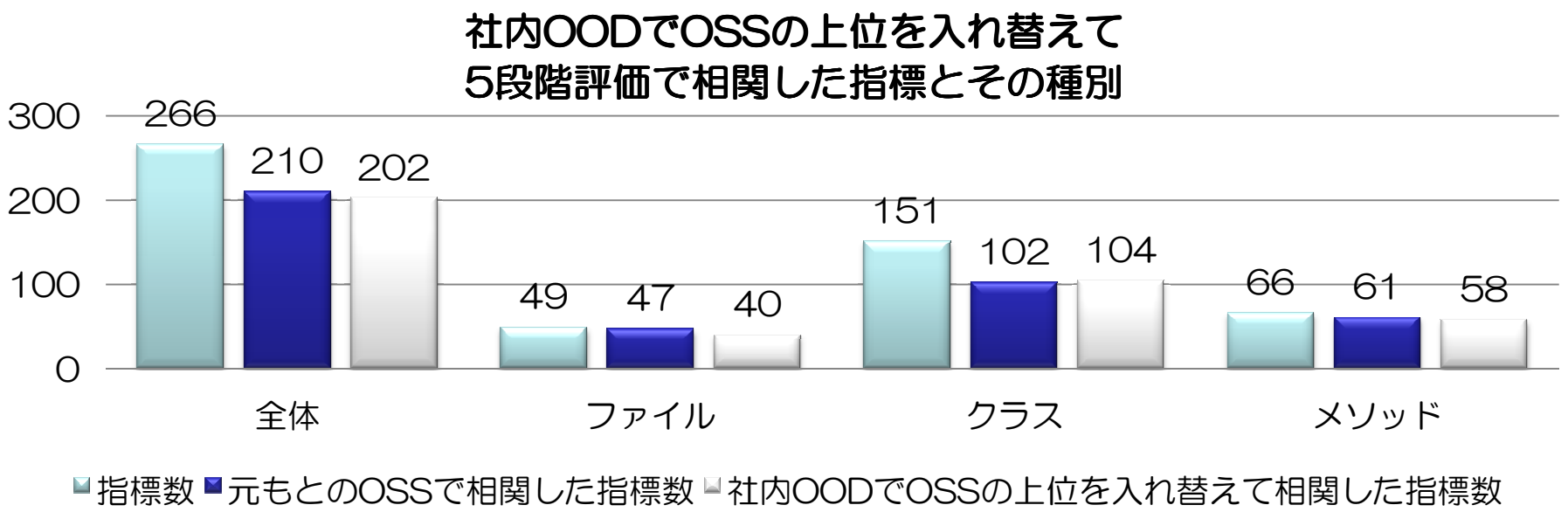
ProjectK	ProjectN	ProjectO	ProjectP	ProjectL	ProjectM
Java	C#	Java	C#	J,JS	Java
107K	30K	54K	254K	680K	57K
4	4	4	4	5	5

集計に新規追加

項目	FireFox	Libre Office	roslyn	chrome	jdk	Project K	Project N	Project O	ProjectP	ProjectL	Project M
	C/C++	C++	C#	C	Java	Java	C#	Java	C#	J,JS	Java
	3.8M	2.9M	560K	3.2M	1.3M	107K	30K	54K	254K	680K	57K
主観評価	1	1	2	2	3	4	4	4	4	5	5

社内OODデータをOSSの上位と入れ替えた場合





社内OODデータをOSSの上位と入れ替えた場合

変数名	社内OOD	OSS
CBO>100個数	-0.778	-0.756
CBO>100個数比率	-0.755	-0.738
CBO>100規模	-0.741	-0.714
CBO>100規模比率	-0.809	-0.738
CBO>50個数	-0.913	-0.934
CBO>50個数比率	-0.731	-0.767
CBO>50規模	-0.815	-0.799
CBO>50規模比率	-0.779	-0.586
CBO>20個数	-0.785	-0.755
CBO>20個数比率	-0.498	-0.285
CBO>20規模	-0.836	-0.814
CBO>20規模比率	-0.688	-0.216
CBO>5個数	-0.803	-0.77
CBO>5個数比率	-0.518	-0.504
CBO>5規模	-0.856	-0.844
CBO>5規模比率	-0.73	-0.078

社内OODであれば
OSS同様
比率ではなく、
個数
規模
といった
絶対値に
相関している。

変数名	社内OOD	OSS
LCM>90個数	-0.614	-0.498
LCM>90個数比率	0.34	0.138
LCM>90規模	-0.813	-0.809
LCM>90規模比率	0.477	0.081
LCM>70個数	-0.774	-0.73
LCM>70個数比率	0.395	-0.038
LCM>70規模	-0.835	-0.828
LCM>70規模比率	-0.043	-0.004
LCM>50個数	-0.781	-0.748
LCM>50個数比率	0.299	-0.216
LCM>50規模	-0.827	-0.826
LCM>50規模比率	0.155	-0.026
LCM>30個数	-0.792	-0.764
LCM>30個数比率	0.198	-0.323
LCM>30規模	-0.831	-0.826
LCM>30規模比率	0.163	-0.001

・社内OODなら、悪いものが絶対的に多ければ、比率に関係なく全体の評価も悪いという、考えてみれば非常に全うな相関をしている。
　つまり、大柄な人も小柄な人も、熱が高ければ風邪、と言う判断。

古い社内データをOSSの下位と入れ替えた場合

項目	FireFox	Libre Office	roslyn	chrome	jdk	Tomcat	eclipse	PaintNet	Struts
	C/C++	C++	C#	C	Java	Java	Java	C#	Java
	3.8M	2.9M	560K	3.2M	1.3M	220K	380K	67K	144K
主観評価	1	1	2	2	3	4	5	5	5

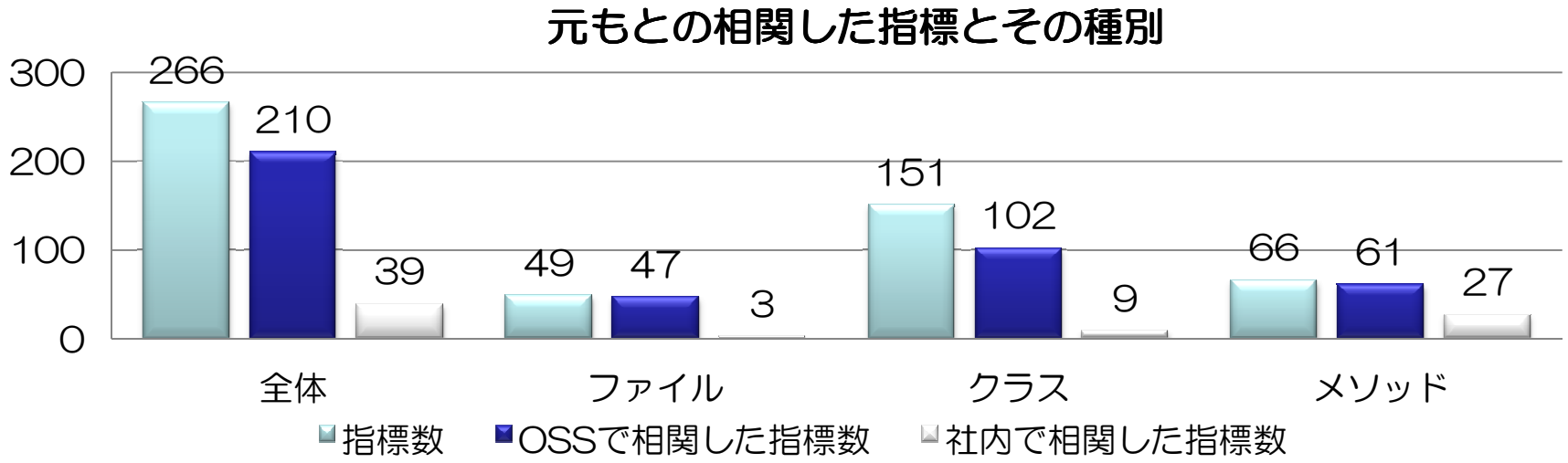
上位1，2を入れ替え

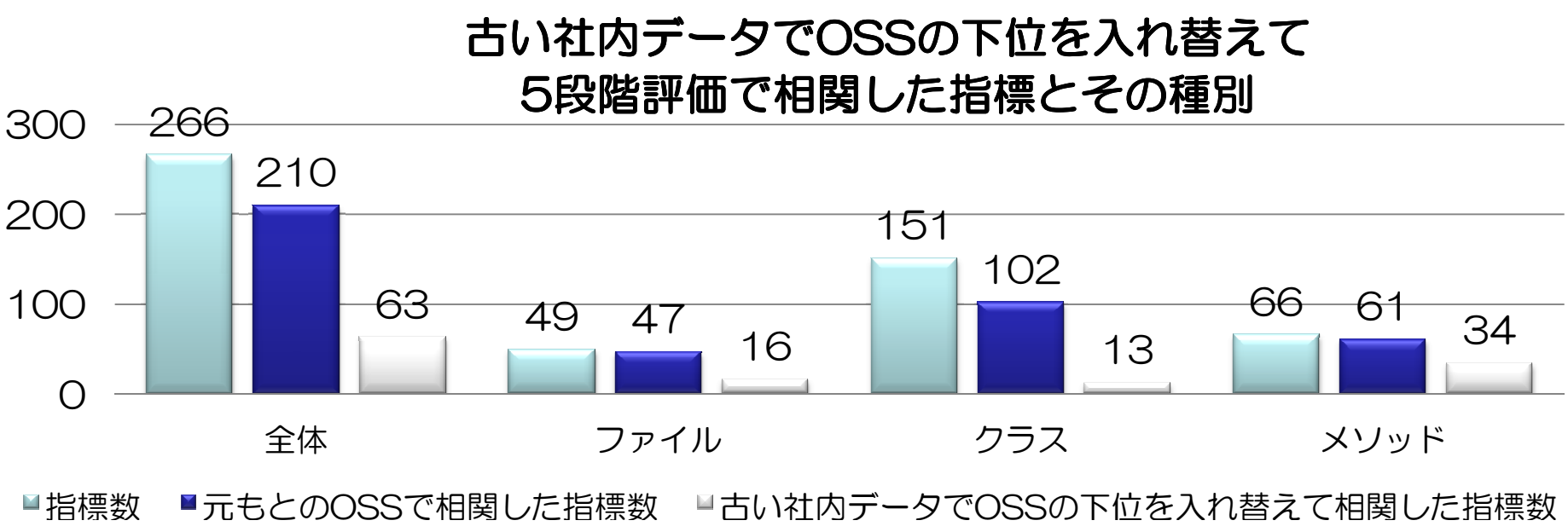
古い社内データをOSSの下位データと入れ替えたら、OSSデータも相関しなくなるのかを検証

Project A	Project B	Project C	Project D	Project E	Project F	ProjectG
C/C++	C/C++	C++	C#	C++	C++	C++
408K	3.9M	994K	1.3M	619K	1.3M	3.1M
1	1	1	1	1	1	2

項目	Project A	Project B	Project C	Project D	Project E	Project F	Project G	jdk	Tomcat	eclipse	Paint Net	Struts
	C/C++	C/C++	C++	C#	C++	C++	C++	Java	Java	Java	C#	Java
	408K	3.9M	994K	1.3M	619K	1.3M	3.1M	1.3M	220K	380K	67K	144K
主観評価	1	1	1	1	1	1	2	3	4	5	5	5

古い社内データをOSSの下位と入れ替えた場合





古い社内データをOSSの下位と入れ替えた場合

変数名	古い社内データ	OSS
CBO>100個数	-0.022	-0.756
CBO>100個数比率	-0.226	-0.738
CBO>100規模	-0.254	-0.714
CBO>100規模比率	-0.212	-0.738
CBO>50個数	0.055	-0.934
CBO>50個数比率	-0.269	-0.767
CBO>50規模	-0.213	-0.799
CBO>50規模比率	-0.127	-0.586
CBO>20個数	0.177	-0.755
CBO>20個数比率	0.047	-0.285
CBO>20規模	-0.064	-0.814
CBO>20規模比率	0.233	-0.216
CBO>5個数	0.217	-0.77
CBO>5個数比率	0.184	-0.504
CBO>5規模	-0.097	-0.844
CBO>5規模比率	0.43	-0.078

古い社内データを使用すると、上位がOSSでもファイル、クラスメトリクスとは相関しない。

変数名	古い社内データ	OSS
LCM>90個数	0.49	-0.498
LCM>90個数比率	0.144	0.138
LCM>90規模	-0.175	-0.809
LCM>90規模比率	-0.067	0.081
LCM>70個数	0.277	-0.73
LCM>70個数比率	-0.253	-0.038
LCM>70規模	-0.176	-0.828
LCM>70規模比率	-0.351	-0.004
LCM>50個数	0.237	-0.748
LCM>50個数比率	-0.391	-0.216
LCM>50規模	-0.168	-0.826
LCM>50規模比率	-0.206	-0.026
LCM>30個数	0.221	-0.764
LCM>30個数比率	-0.406	-0.323
LCM>30規模	-0.154	-0.826
LCM>30規模比率	-0.018	-0.001

・古い社内データを使用すると、ファイル・クラスメトリクスは殆ど相関しないので、品質判断には使用できない。


FUJIFILM
Value from Innovation