SPI Japan 2009　10月6日

プログラムソースコードの高精度な品質評価

鷲崎 弘宜（早稲田大学 / 国立情報学研究所）
washizaki@waseda.jp　http://www.washi.cs.waseda.ac.jp/

本成果は下記の皆様との産学連携により実現されました。
波木 理恵子（オージス総研）
田邉 浩之（オージス総研）
小池 利和（ヤマハ株式会社）

とあるC言語ソースコード

```
char buffer[128];
char *cmd, *arg;
int monitor_period,mic_threshold, show_mic_value, len, size,
i, counter, finish, c;
void shrine_task(int invalid_param)
{
  init_shrine_counter();
  start_shrine(0);
  return 0;
}
```

グローバル変数の多用

戻り値未チェック

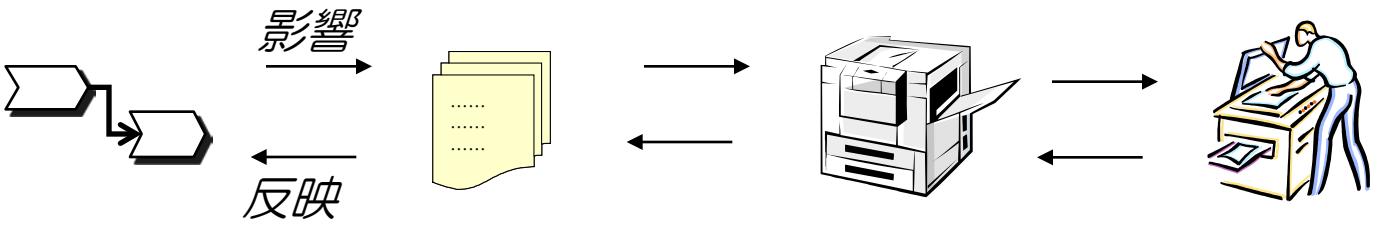
```
int start_shrine()
{
  while (1) {
    if( show_mic_value == 1 ) syslog( LOG_NOTICE
"mic_value
    if( (
      coun
      if (                                    = 0; }
```

関数内の複雑な分岐

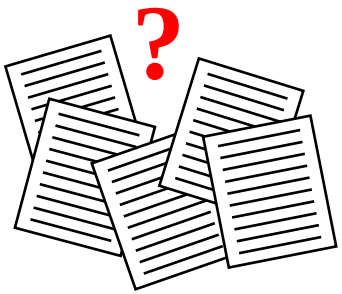
動くには動くけれど、きちんと書けてない。
どこにどのような問題があるのだろう？
何にどれぐらい影響するのだろう？

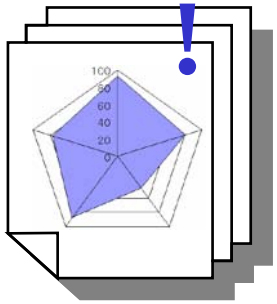
ソースコード品質の「見える化」

- ソースコードの品質重要
  - システムの性能、開発/保守コストに影響
  - 静的解析の欠陥除去率 87%（ ＞ レビュー、テスト）



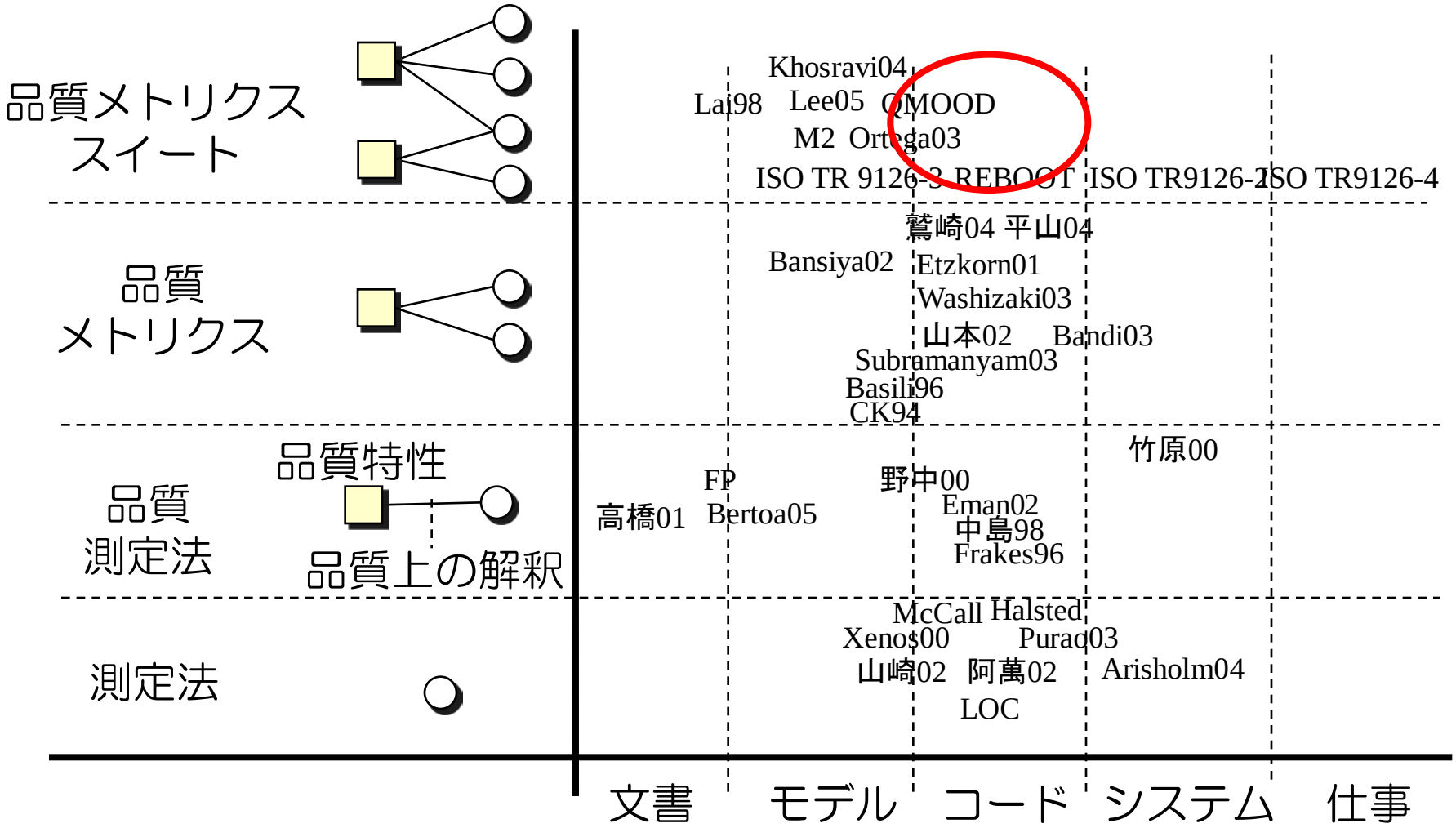
- 静的解析のベタな結果では、品質は「見えない」
  - 簡単に、具体的に、同じように、体系的に把握したい
  - 問題特定に利用: どのモジュールに問題？ どの品質に影響?
  - 問題解消に利用: 改善したい品質は？ どこから先に？





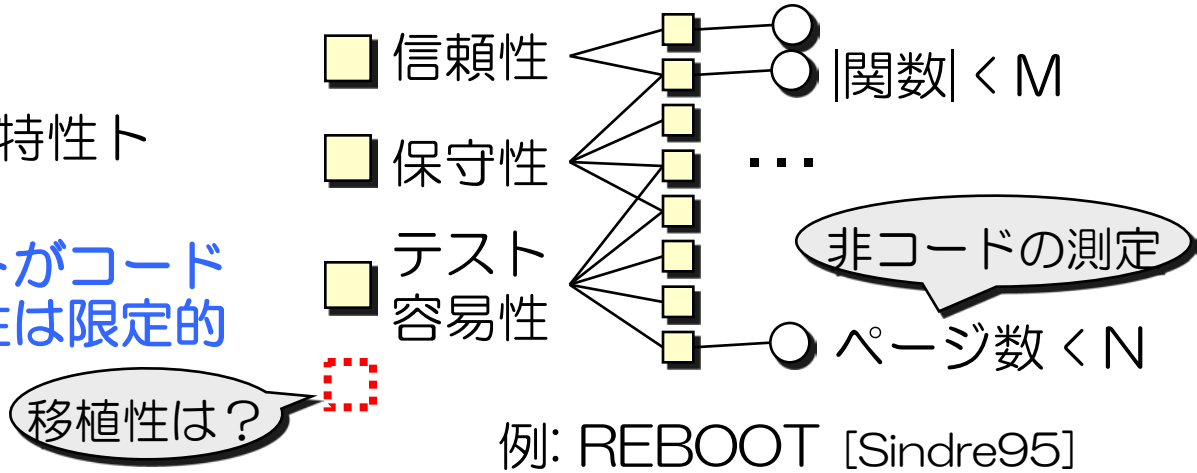
従来の品質測定の取り組み

- これまでに多数の品質測定法が提案されている
- しかし「色々あるが十分に活用されていない」[小笠原04



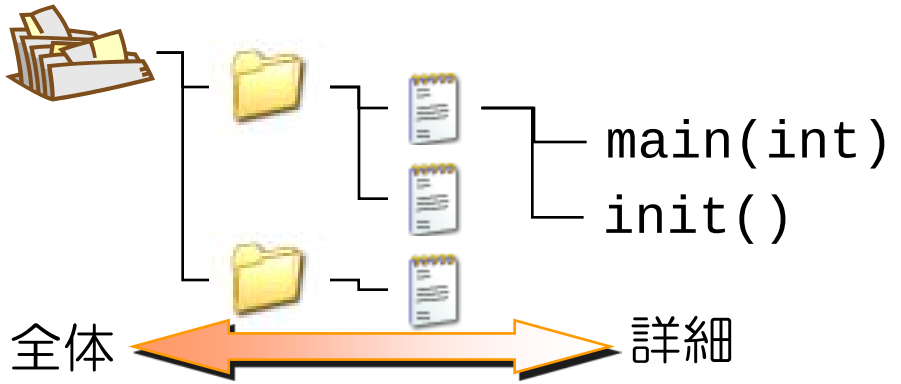
問題の本質

- 網羅性の低さ
  - ISO9126の複数品質特性トレードオフをみたい
  - しかし、既存スイートがコードで測定可能な品質特性は限定的



例: REBOOT [Sindre95]

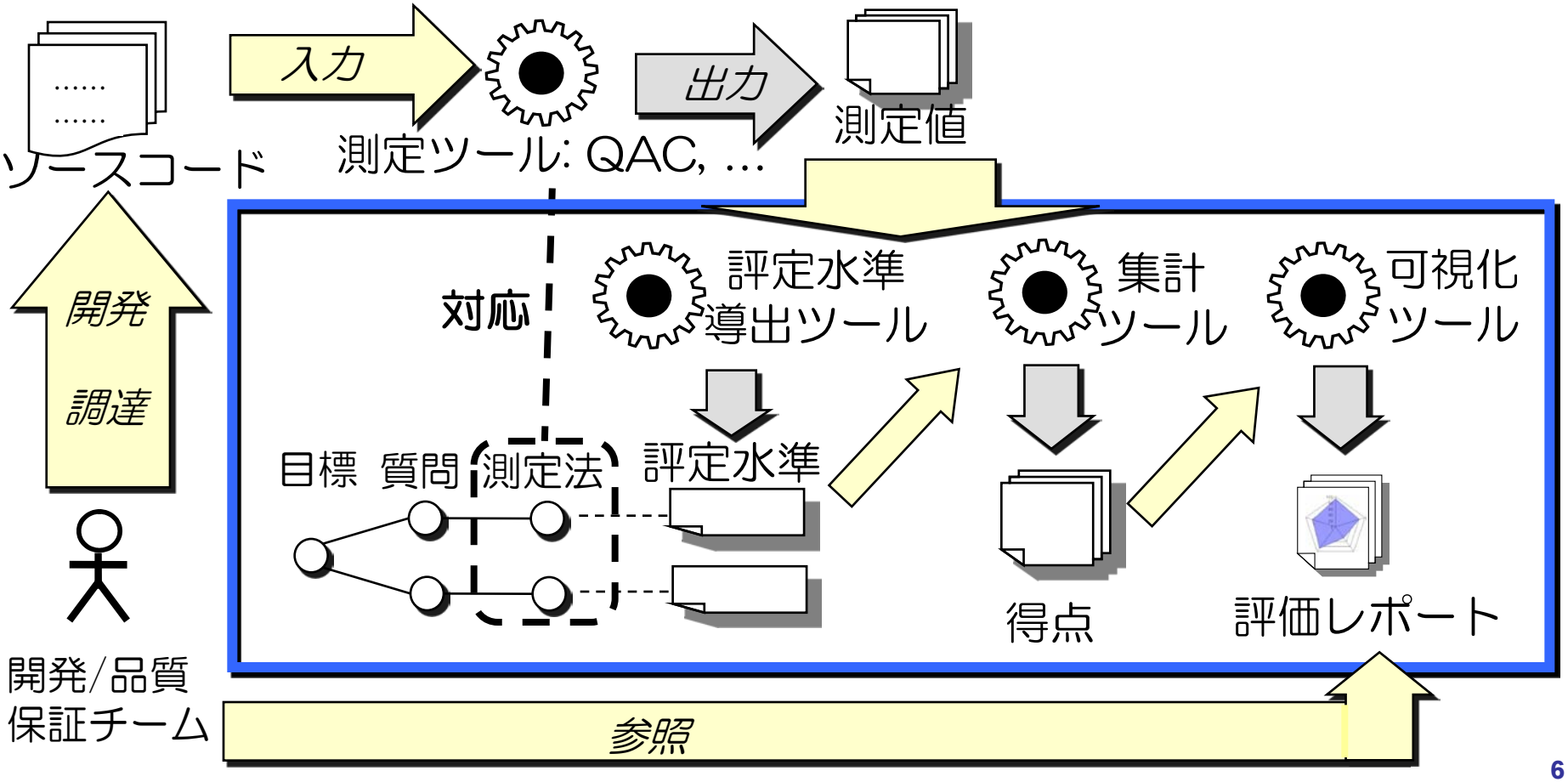
- 分解/総合評価能力の欠如
  - 全体から詳細まで一貫して測定/評価したい
  - しかし、部分まで一貫して扱う具体的コード対象スイートなし



- 評定水準導出の非簡便さ
  - 一般的な「閾値」がほしい
  - しかし、従来は被利用状況や定性的評価など追加情報が必要

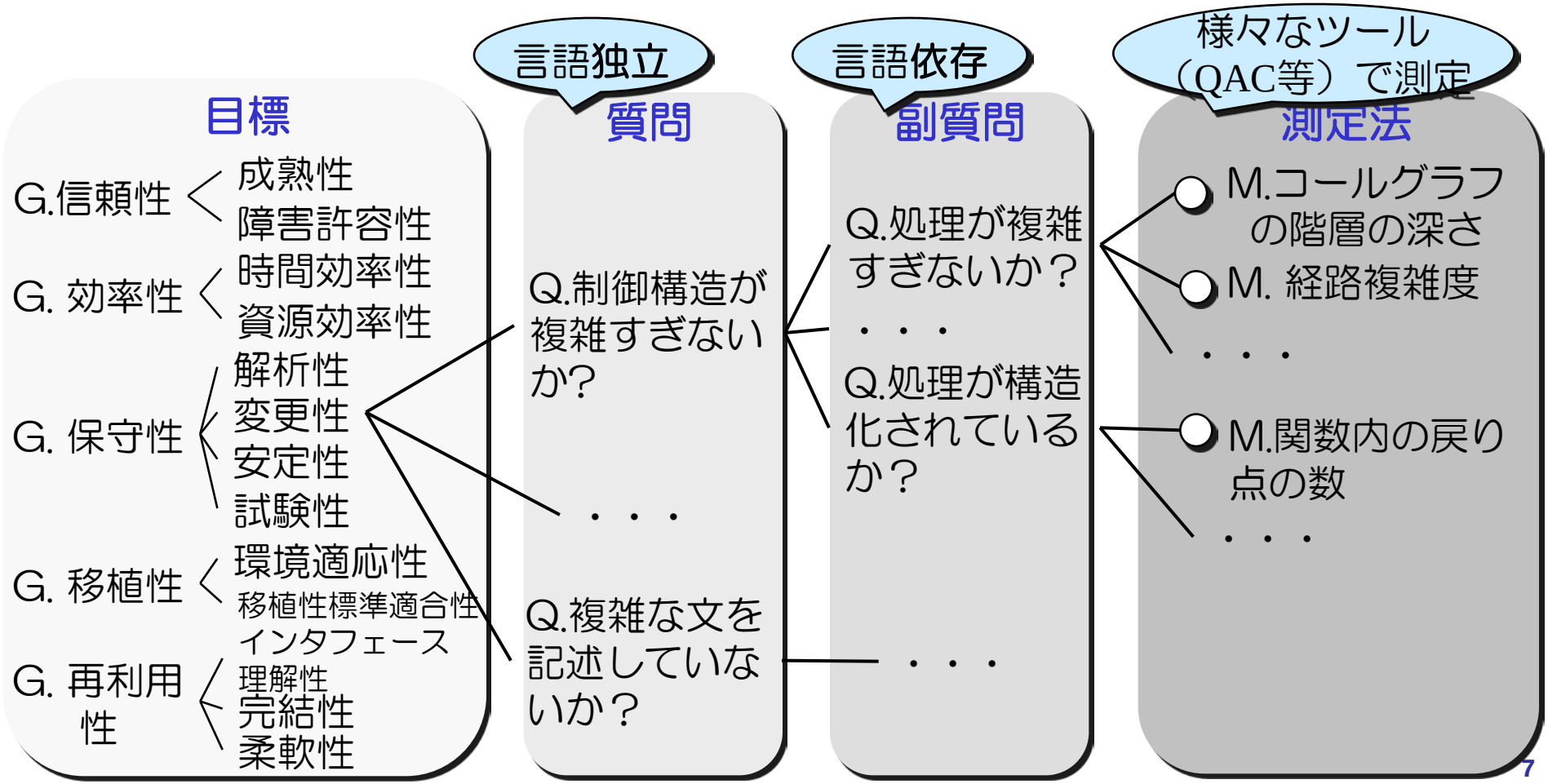
高精度品質測定/評価枠組み

- ISO9126ベース品質モデル　⇒「網羅性の低さ」改善
- 測定値正規化/集計、可視化　⇒「分解/総合評価能力の欠如」解決
- 180超の測定による統計的導出 ⇒「評定水準の非簡便さ」解決
- 測定対象: C，C++ソースコード
- **ツール『Adqua』無償提供開始**



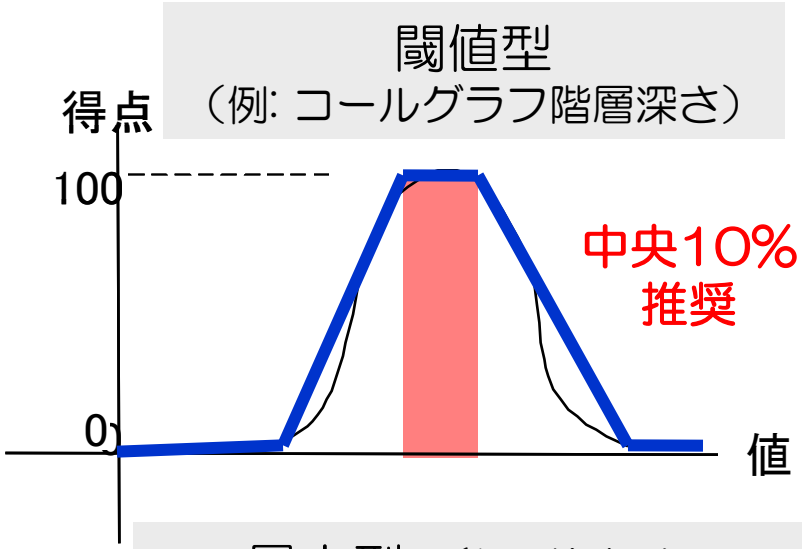
スイート

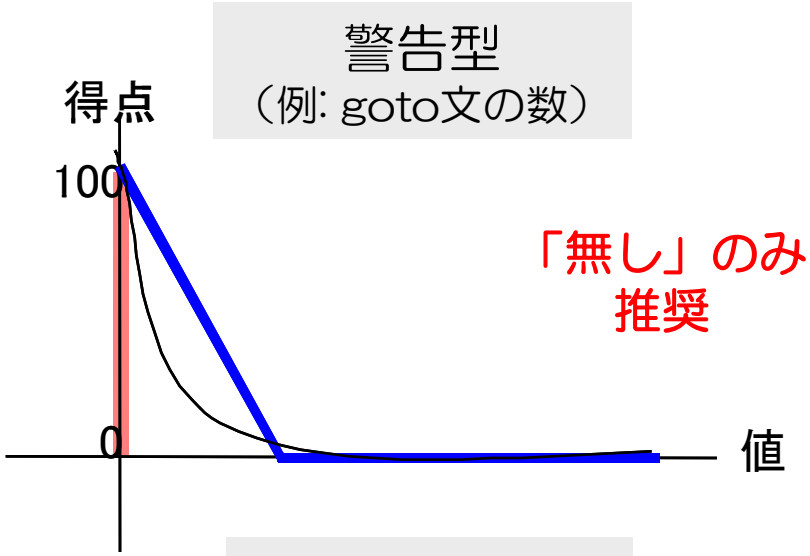
- ISO9126ベースの網羅的品質モデル
- GQM法による段階的マッピング
  - 複数品質改善従事者によるレビュー/修正の繰り返し（4年超）
  - 43の質問、123の副質問、348(C言語)・433(C++)の測定法

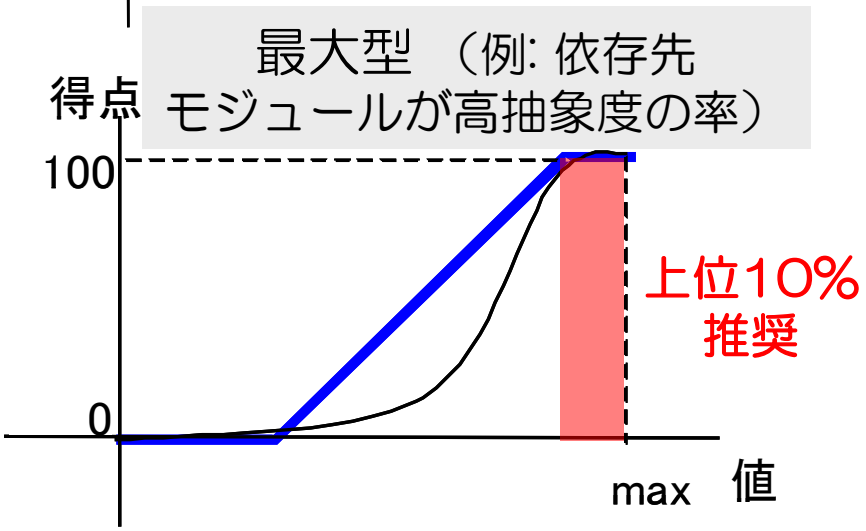


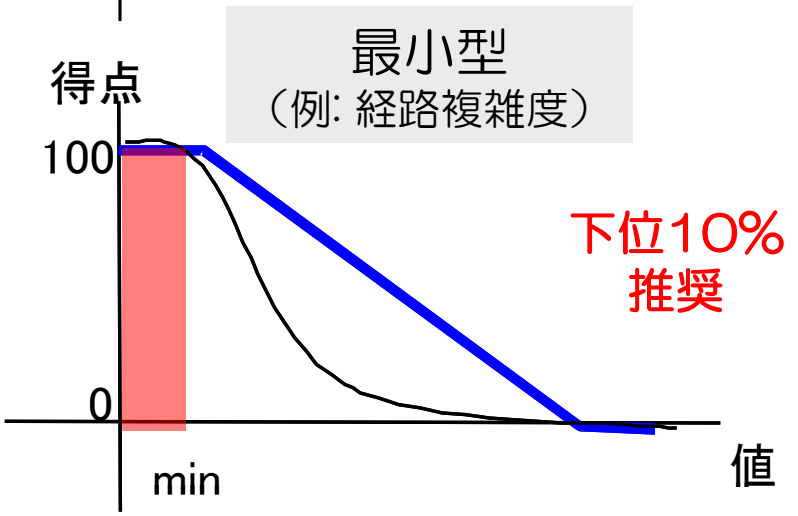
1～3: 評定水準の導出と得点化

- 1. C/C++の合計145プロジェクトより測定値分布取得
- 2. 分布に基づき4（8）種への分類、必要に応じて対数変換
- 3. グラフに基づく測定値の得点化





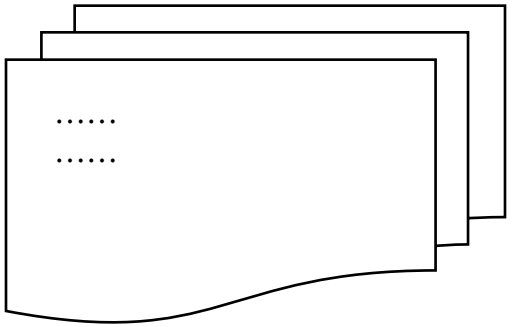




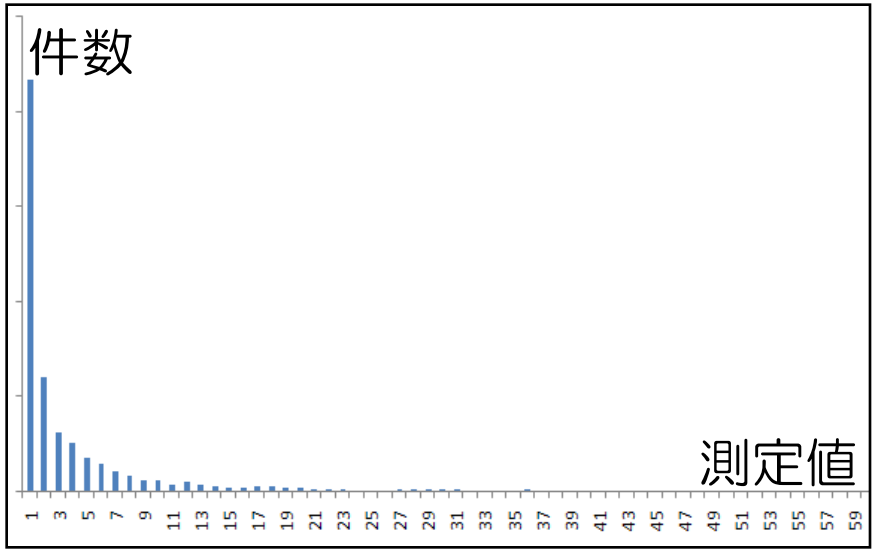
例: 関数の経路複雑度（サイクロマティック数）

- 種別: 最小型
- 評定水準: 1 推奨、2~9 許容

80超のプロジェクト

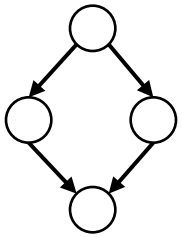


1. 測定値分布の取得



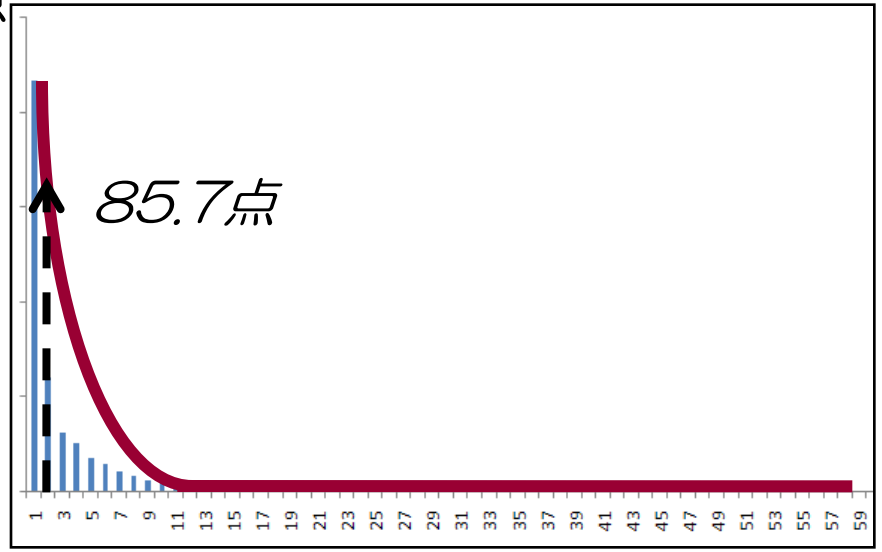
2. 得点グラフ化

```
void foo(int p){
 if (p < LIMIT){
  bar();
 } else {
  baz();
 }
}
```



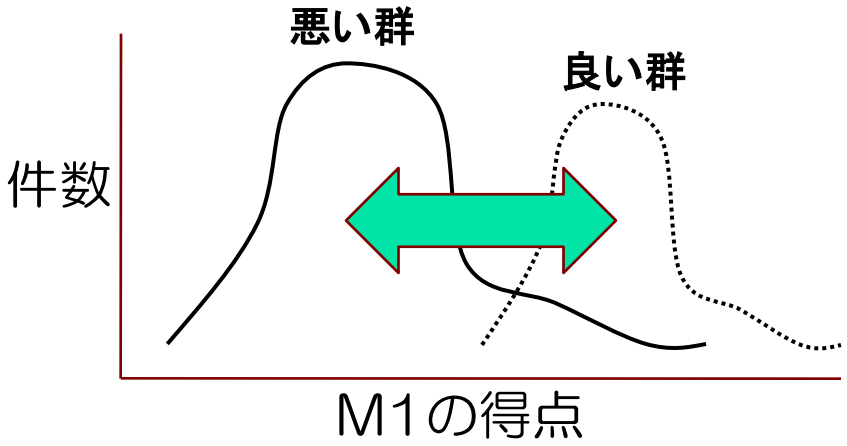
＝2

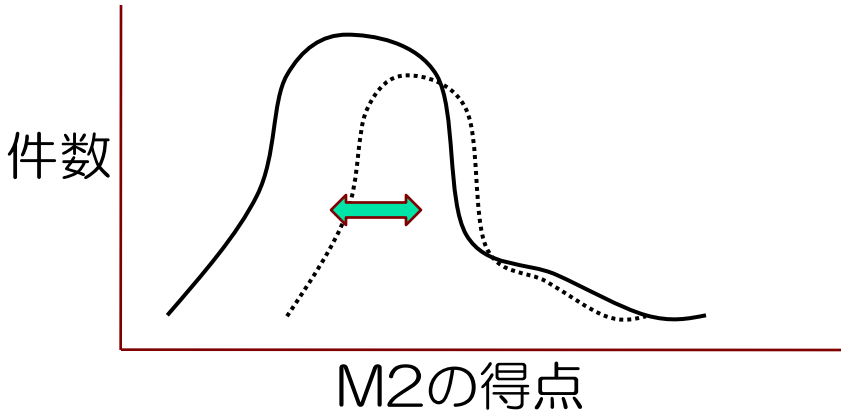
3. 測定と得点



4. 測定法の重要さによる重みづけ

- 品質測定に効くものとそうでないものを区別
  - 4-1. 典型的な一定規模のプロジェクト群を「良い群」「悪い群」に定性的に分類
  - 4-2. （良い群の平均点 － 悪い群の平均点）を算出し差が大きいほど強く重みづけ



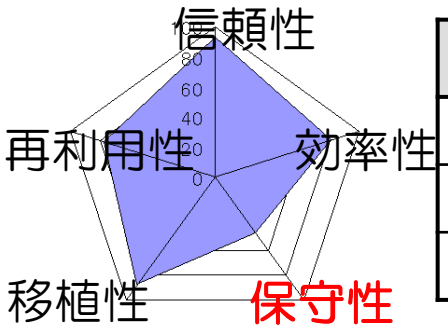


測定法	M1.経路複雑度	M2. コールグラフの階層の深さ
重み ※	1.0	0.5
得点例	80	80
重みづけ後の得点例	80	40

※掲載した重みは説明のための例であり実際の測定結果は異なります

5~6. 規模による重みづけと集約

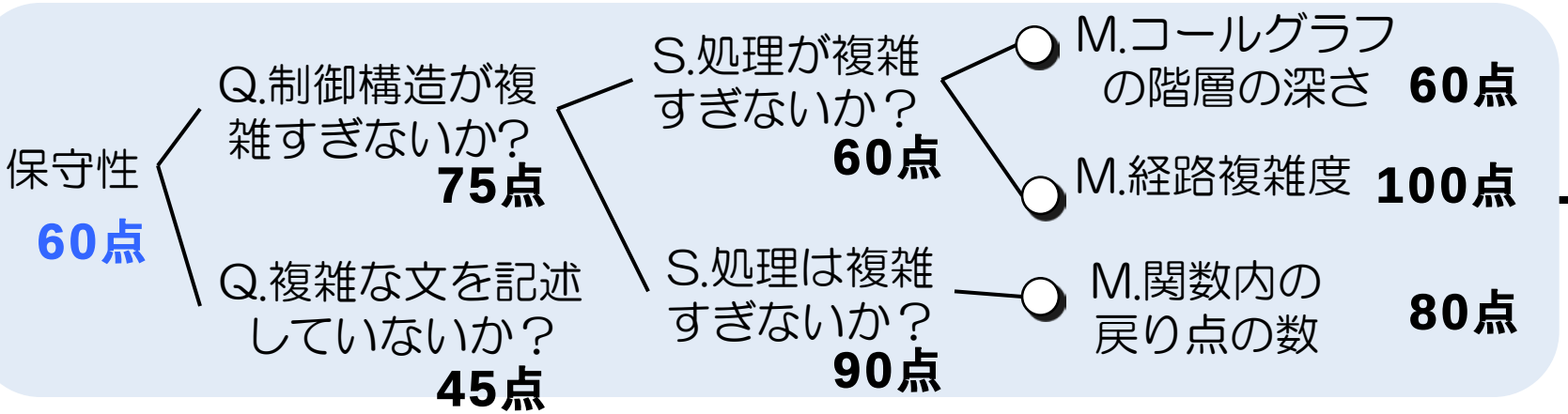
- 5. 規模の大きい要素の得点を相対的に重く
  - 規模: 行数、ステートメント数
- 6. 得点の集約: 品質特性単位と要素単位
  - 特性: スイートに基づく集約
  - 要素: 関数→ファイル→ ディレクトリ/名前空間→システム



ディレクトリ	保守性
System	83点
Application	91点
Device	69点

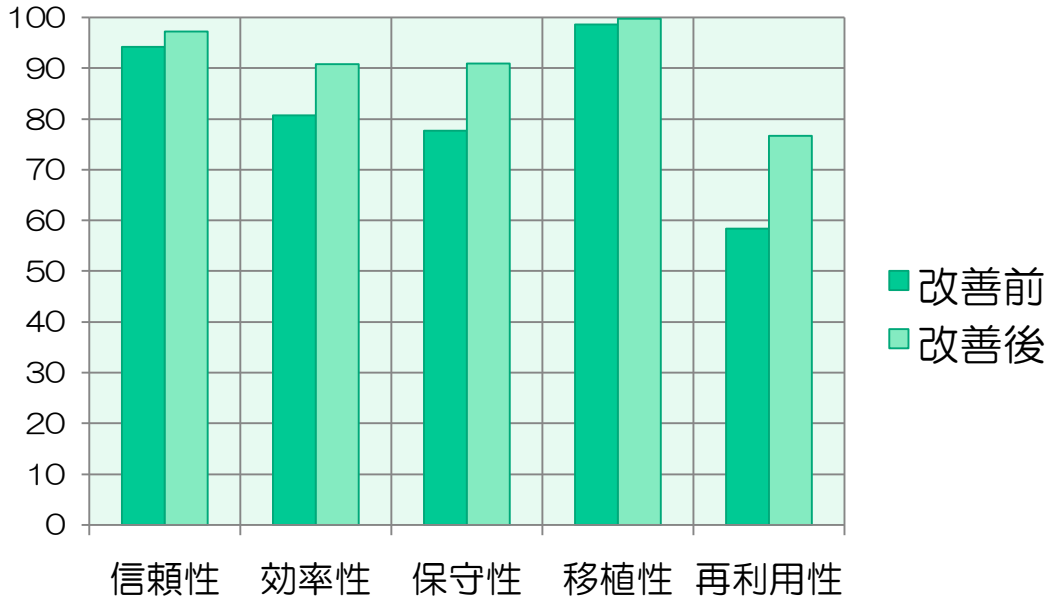
ファイル	保守性
lopanel.c	72点
motor.c	66点

関数	保守性
main()	79点
init()	80点
task()	45点
start()	60点



実験的評価1: サンプル

- 神棚プログラム、1,200[ELOC]程度
- 効率性、保守性、再利用性の顕著な向上捕捉
  - 関数の複雑さ低減などを測定結果として得た



改善前

```
extern int mic_threshold;
extern int show_mic_value;
extern int monitor_period;
void main(void)
{
 MY_ADCSR.BYTE = 0x31;
 while(!MY_ADCSR.BIT.ADF);  ···
 while (1) {  ···
  for(i=0;i<64;i++)buffer[i]='¥0';
```

改善後

グローバル変数抑制

適度な関数分割

```
int main()
{
 start_microphone();
 init_switch();
 init_motor();
 mic_task();
 while (1) { ···
  char buffer[BUFFER_SIZE];
  read_command(buffer,BUFFER_SIZE);
```

実験的評価2: 実プログラム

- 車載/プリンタ組込みプログラム3つ改善前後
  - 規模: 7,500～111,500[ELOC]
  - 定性評価: 改善前開発者のアンケート回答
  - 定量評価: 品質評価枠組みによる評価
- 4特性について定性と同様の向上確認、有効性確認

  ※ 06年度版の結果であり09年現在はさらなる改善が図られています

定性的評価
（改善前→後）

対象	信頼性	効率性	保守性	移植性	再利用性
S1	92→92	80→83	75→95	69→100	92→100
S2	59→79	67→71	54→78	76→ 88	60→ 83
S3	→92	→78	→75	→ 88	→ 83

定量的評価
（改善前→後）

対象	信頼性	効率性	保守性	移植性	再利用性
S1	79→83	96→92	80→88	87→86	80→92
S2	88→99	99→96	74→89	94→96	0→95
S3	85→90	96→86	67→75	77→82	0→ 0

適用事例: 派生開発における活用

- 組込みソフトウェアの実際の派生開発
  - 動機: 流用部分の品質が開発に大きく影響
  - 方法: 過去7機種分について、開発・保守の実績データと、品質得点の相関性を分析
  - 結果: 保守性、信頼性について高い相関性あり。品質評価枠組みの妥当性が明らかに。

※詳細は当日ご紹介予定

[出典] 小池利和:ソースコードの品質を直接、詳細に測る方法, SQiPシンポジウム2009

適用事例：業界の品質傾向

140を超える測定実績に基づく傾向

- (1) 品質特性別の得点傾向
- (2) 製品分野別の得点傾向

※詳細は当日ご紹介予定

まとめ

- 高精度ソースコード品質評価に成功
  - スイートの4年以上レビュー実績、ISO9126ベース網羅
  - 145プロジェクト測定実績に基づく評定水準導出
  - 測定法の重要さやプログラム規模を加味し得点化
  - 高い学会評価: 情報処理学会SES論文賞、研究賞
- 活用にあたって
  - 5特性、348(C言語)・433(C++)測定法
  - C/C++言語＆組込み: そのまま
  - 他言語、他ドメイン: 部分を再利用可

サービス展開、展望

- 品質評価ツール「Adqua」（アドクア）の無償提供開始
  - オージス総研により提供中
  - オージス総研、早稲田大学、ヤマハの共同研究で開発
  - http://www.ogis-ri.co.jp/news/g-01-00000170.html
- 品質診断の実サービスの提供中
  - オージス総研により提供中
  - http://www.ogis-ri.co.jp/solution/a-041-00000160.html
- 今後の展望
  - 測定値収集による継続的改善
  - 他の様々な言語への展開
  - 分野ごとの傾向公開

発表資料、記事

- 主要論文
  - 鷲崎弘宜, 小池利和, 波木理恵子, 田邉浩之, ″C言語プログラムソースコードの再利用性測定法とその評価″, ソフトウェアテストシンポジウム JaSST'09 Tokyo, 2009.
  - 鷲崎弘宜, 波木理恵子, 福岡呂之, 原田洋子, 渡辺博之, ″プログラムソースコードのための実用的な品質評価枠組み″, 情報処理学会論文誌, Vol.48, No.8, pp.2637-2650, 2007.
  - H. Washizaki, R. Namiki, T. Fukuoka, Y. Harada and H. Watanabe, ″A Framework for Measuring and Evaluating Program Source Code Quality″, 8th International Conference on Product Focused Software Development and Process Improvement (PROFES 2007)
- 記事
  - 日経エレクトロニクス『組み込みソフト: ソフトの品質を可視化するツール 無償提供が始まる』2009年4月6日号