1C－8　ＳＴＭＥｐｊによるソフトウェア品質改善の推進（３）
－　静的解析で得られたデータの分析結果　－

小笠原秀人† 原田明宏‡ 杉山昭洋‡ 吉崎浩二‡
†株式会社東芝　研究開発センター　S&S研究所
‡株式会社トプコン　技術本部　S&S推進部

1. はじめに

(株)トプコンは(株)東芝の協力の下、1997 年 7 月からソフトウェアプロセスの改善と機械化のための活動を行ってきた。その中の活動アイテムの一つが、「静的解析によるソフトウェア品質改善活動」である。

本論では、この静的解析ツールによる品質改善活動をとおして得られたデータから以下の各項目について分析した結果を示す。

(1) 検出された警告メッセージ数と修正件数

検出された警告メッセージ数と修正件数を分析し、静的解析による効果を明確にする。また、C 言語によるプログラム開発において、どのような問題点が内在しており、今後、どのような改善が必要かを分析する。

(2) メトリクス値の傾向分析

定量的な評価のための元となるデータを得る。さらに、保守性やテスト容易性を考慮した標準的な値を検討する。

(3) 静的解析結果にもとづくレビュー効率

静的解析結果にもとづくレビュー効率を、人手によるコードレビューと比較して評価する。

2. プログラム静的解析とは

静的解析の機能には、コード解析、メトリクス計測、プログラム構造分析などがある。今回の活動で使用した静的解析ツール QAC の警告メッセージ例を図 1 に示す。

```
270:    if ( *sts = NULL ) {
                -

sample.c (270) ++WARNING++ : <=3= (3314) この制御式は代入式です。— 明示的に書いて、ゼロに対するテストをするように。
            .... ( 中略 ) ....
280:
```

図 1　QAC の警告メッセージ例

3. 分析対象としたデータ

分析対象としたデータの概要を、表 1 に示す。分析対象としたソフトウェアは、約 50 万 step の C ソースコードある。それぞれの対象製品は、結合テスト完了時などのタイミングで静的解析を行った。

その結果を、解析センター担当者のレビュー、共同レビュー、開発者のレビューという 3 段階でチェックした。

また、ツール解析時には、解析センターで分類した重要度の高い警告メッセージ（ランク A：16 種類、B：31 種類、C：7 種類。ランク A が一番重要度が高い）だけが検出されるように設定した。

表 1　分析対象としたデータの概要

製品分野	マイコンソフトウェア
開発言語	C 言語
特徴	シリーズ製品が多く、流用部分と新規開発部分が混在する。比較的少人数による開発。

4. 分析結果

4.1　警告メッセージの指摘件数と修正件数

指摘された件数と、レビューによって修正された件数を表 2 に示す。なお、すべての指摘された件についてはレビューで確認してある。その中で修正が必要と判断されたものに関しては、修正を加えてある。

表 2　警告メッセージの指摘件数と修正件数

項目＼ランク	A	B	C	全体
QAC による指摘件数	1322	16445	1489	19256
コードレビュー修正件数	331	1408	113	1852
指摘件数の検出密度(件/KLOC)	2.6	32.9	2.9	38.5
修正率(修正件数/指摘件数)	25%	8.6%	7.6%	9.6%

静的解析による指摘から、約 1800 件の修正が行われており、静的解析の有効性が明らかになった。

全体の修正率が約 10%になっているのは、以下のような理由のためである。例えば、冗長なキャストや、条件文の中で代入文が使われている場合に警告メッセージが生成されるが、意図的に行っている開発者もおり、このような場合には、指摘された警告は修正されず、“レビューの結果問題なし”と判断されている。

Promoting the software quality improvement activities by STMEpj (3)
Hideto Ogasawara†, Akihiro Harada‡, Akihiro Sugiyama‡,
Kouji Yoshizaki‡
†hideto@ssel.toshiba.co.jp
‡{aharada, akihiro_sugiyama, kouji_yoshizaki}@topcon.co.jp
†System&Software Research Laboratories, TOSHIBA Corporation
‡S&S Group, S&S Promoting Dept., TOPCON Corporation

次に、重要度ランク別に、内容を分析する。

ランクAの修正率は25%であり、重要度の高い警告がランクAに分類されていることがわかる。

ランクBには、変数の型の不一致や、プロトタイプ宣言を使っていないことによる警告があり、古くからのソースコードを利用している場合などは、指摘される警告数が増える傾向にある。

ランクCは、スタイルに関する警告とプログラムの複雑さによる警告が中心であり、保守性と強く関連する。各重要度ランク別ごとの代表的な警告例を表3に示す。

表3　各重要度ランク別ごとの代表的な警告例

重要度ランク	警告内容
A	Defaultスペルミス
	比較(==)と代入(=)のミス
	実数と0.0とのif文等価比較
B	条件分岐におけるその他の場合抜け
	大きい型から小さい型への代入
	未初期化変数の使用
C	旧プログラミングスタイル
	ネストの深さ

静的解析ツールの利用による結果として望ましい姿は、指摘件数が0、修正率100%である。これを実現するために、今後、教育の推進とガイドの徹底、および重要度ランクの定期的な見直しが重要となる。

4.2　メトリクス値の傾向分析

メトリクスは、ファイルベースメトリクス（コード行数、命令行数、コメント率、Halstead数）とモジュールベースメトリクス（サイクロマティック数、最大ネスト数、関数呼出数、命令行数）の2種類を計測した。

全体としては、各モジュールの命令行数の平均は約40行、サイクロマティック数の平均は約7であり、モジュール化を意識した開発が行われていることがわかった。また、命令行数と複雑さ（サイクロマティック数やネストの深さ）の相関が高いこともわかった。

ある製品の命令行数とサイクロマティック数の散布図を図2に示す。モジュール化を意識した開発はされているが、中には複雑な構造のモジュールも存在していることがわかる。複雑なモジュールに対しては、コードレビューを通して、モジュール分割の対象とした。

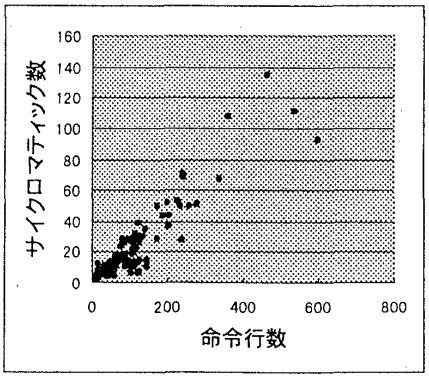

図2　命令行数とサイクロマテッィク数の散布図

構造の複雑なモジュールに問題が混入する可能性の高いことは、多くの文献[1]で述べられている。品質作り込み、およびテスト容易性や保守性を考え、今後、定期的にメトリクス値を基準値と比較してモニタリングし、構造が複雑になっているモジュールなどを早期にチェックする仕組みを確立することになった。

4.3　静的解析結果にもとづくレビュー効率

静的解析結果にもとづくレビュー効率の評価結果を表4に示す。手作業による最も効率的なレビュー速度は、150行/時と言われている[2]。この値と比較すると、今回の適用で得られたレビュー効率1900行/時は、少なく見積っても、人手でのコードレビューより約13倍の効率が得られている。

表4　静的解析結果にもどづくレビュー効率

項　目		データ
1製品当たりの規模		38KLOC
レビュー時間	静的解析センター	約10H
	共同	約2H
	開発担当者	約8H
レビュー効率（件/時間）		1.9KLOC

5.　まとめ

静的解析ツールは、全ソースコードを短時間で解析できるため、ソフトウェアの品質向上および安全性を検証するための技術として非常に有効である。今回の適用では、この有効性を示すことができた。

最後に、静的解析ツール導入によって得られた効果を以下に示す。

- センター活動の有効性と重要性を認識できた。
- ソフト生産技術の適用により、実機テスト以前の段階で品質を確認することの重要性を認識できた。
- レビューの重要性の認識を深めることができた。
- 開発者の陥りやすいエラーの傾向が把握できた。
- 開発者の陥りやすい事例をガイドブックとしてまとめ、教育に活用できた。
- メトリクス計測結果とコード品質の評価結果から、定量的な設計標準の重要性を認識できた。

参考文献

[1]LesHatton,"Safer C",McGRAW-HILL,1995.

[2]Thomas Liedtke,Christof Ebert:"On the Benefits of Reinforcing Code Inspection Activities", EuroSTAR '95.