PRQA Programming Research

WHITE PAPER

「MISRA-C 準拠度チェックツールの比較調査」に関する概要

調査実施機関: TERA-Labs

TERA-Labs は、アントワープに所在地を置く
Karel de Grote University College の調査研究部門です

TERA-Labs(アントワープに所在地を置く Karel de Grote University College の調査研究部門)は、静的解析ツールの性能に関する調査結果をまとめたレポート「MISRA-C 準拠度チェックツールの比較調査」[Ref 1]の最終版を、2012 年 5 月 9 日に発行しました。

このホワイトペーパーは、以下に示す 2 つのセクションから構成されています。

- **セクション 1 :** TERA-Labs が発行した 80 ページにおよぶレポートの概要を示します。PRQA によって作成されたこの概要は、レポートに含まれている重要なコンテンツを直接引用する形で作成されており、*TERA-Labs においてレポートを作成した著者からも、公正かつ正確な要約であると認められています。*

- **セクション 2 :** TERA-Labs のレポートについて、PRQA 独自の見解を示します。

キーワード: MISRA, 静的コード解析, コーディング標準, コンプライアンス

セクション 1:レポートの概要

本セクションでは、80 ページにわたり詳細な記述が行われている TERA-Labs のレポート [Ref 1]を要約し、その概要を示します。

1.1 はじめに

TERA-Labs は、ベルギーのアントワープに所在地を置く Karel de Grote University College の新しい調査研究部門で、自動車から分散コンピューティングに至るまで、幅広い産業における組込みシステムの調査研究を行っています。本稿で取り上げる第三者比較調査(MISRA-C 準拠度チェックツールの比較調査)は、IWT (agentschap voor Innovatie door Wetenschap en Technologie, www.iwt.be)による資金の提供を受け、20 ヶ月を超える期間にわたり実施されました。その結果を受けて、予備調査報告書が 2010 年 10 月 6 日に、最終報告書が 2012 年の 5 月 9 日に提出されています。

1.2 調査対象範囲および目的

この調査は、複数の静的解析ツールにおいて MISRA-C:2004 コーディングルールのチェック能力を評価し、その結果を比較する目的で実施されました。TERA-Labs のエンジニアが作成した複数の MISRA-C:2004 コーディングルールに違反する記述が意図的に含まれているテストケース(“プローブ”)を複数のツールを用いて解析する形で行われ、それぞれのツールにおいて、テストケースに含まれている違反を検出する能力が調査されました。

1.3 調査対象製品

今回の調査対象製品には、MISRA-C 対応を明確に表明している以下の静的解析/コード解析ツールが、インターネット上の検索結果を通じて選ばれました。

1. Development Assistant for C (DAC) - RistanCASE
2. IAR embedded workbench
3. Klocwork Insight
4. LDRA Testbed
5. Parasoft C++test
6. QA·C – Programming Research/PRQA
7. PC-Lint - Gimpel
8. (Prevent - Coverity) *
9. Raincode **

* *Coverity* は、MISRA-C ルールの最小限のサブセットにしか対応していないことが判明したため、調査開始後に調査対象から外されました。そのため、このレポートで示されている調査結果に *Coverity* のデータは含まれていません。
** *Raincode* は、調査開始後に調査対象として追加されました。そのため、このレポートで示されている調査結果にも *Raincode* のデータが含まれています。

法的な要請により、TERA-Labs のレポート(最終版)では、すべてのベンダー名が匿名(Company 01～09)で記述されています。上述したリストにおけるツールの順番と、本稿に掲載されている表(またはグラフ)におけるツールの順番に関係はありません。なお、PRQA の製品(QA·C)は「Company 04」として記載されています。

1.4 調査基準

調査では、“ソフトクライテリア”と“ハードクライテリア”が用いられました。それぞれの基準の定義を以下に示します。

ソフトクライテリア

1. 既存の開発環境に統合しやすいか
2. ユーザビリティ: ツールが操作しやすいか
3. 拡張性: プログラムに機能を追加しやすいか
4. 違反を警告する際に出力されるメッセージの品質
5. コマンドライン機能 (自動解析用)
6. 追加機能 (例: メトリクスの計測など)

ハードクライテリア

7. 正確性: ツールによって検出されるコーディングルール違反は真の違反か?
8. 網羅性: コードに含まれているすべての違反を、ツールが検出できているか(未検出の違反が存在しないか)?

すべての MISRA-C ルールについてテストを行うことは現実的ではないため、今回の調査では、有識者により選ばれた“重要”かつ“代表的”な 11 個のルールに焦点を当て、調査を実施しています。

MISRA-C ルールは、以下に示す 3 つのグループに分類されます。

1. クラッシュ: ルールに違反した場合にソフトウェアがクラッシュするもの
2. 保守性: ルールを順守することで、コードの変更時にプログラマによって導入されるエラーが低減するもの
3. 移植性: ルールを適用することで、移植性が向上するもの

今回、調査の対象となる以下の 11 個のルールは、上述したルールの分類では“クラッシュ”または“保守性”のグループに分類されます。

MISRA-C ルール	ルールの説明
2.3	文字列　/* をコメント内で用いてはならない
8.12	外部結合をもつ配列を宣言するときは、明示的にサイズを記述するか、初期化によって暗黙的にサイズを定義しなければならない
9.1	すべての自動変数は、用いる前に値を代入しなければならない
11.1	関数ポインタは、汎整数型以外の任意の型との間で変換してはならない
12.4	シフト演算子の右側のオペランドの値は、0 以上、かつ、左側のオペランドの潜在型のビット幅未満でなければならない
14.7	関数では、関数の最後に唯一の出口がなくてはならない
15.2	空でない switch 節は、無条件 break 文で終了しなければならない
15.3	switch 文の最後の節は、default 節でなければならない
16.6	関数に渡される実引数の数は、仮引数の数と一致しなければならない

17.6	自動記憶域にあるオブジェクトのアドレスを、そのオブジェクトの存在が終了した後にまで持続期間をもつ他のオブジェクトに代入してはならない
19.10	＃又は＃＃のオペランドとして用いられている場合を除き、関数形式マクロの定義では、仮引数のそれぞれのインスタンスを括弧で囲まなければならない

1.5 調査結果 –ソフトクライテリア

TERA-Labs によるレポートでは、ソフトクライテリアに対する調査結果を 9 つの表を用いて解説しています。本稿では、それらの内容を以下の表に要約して示します。

基準	Comp 01	Comp 02	Comp 03	Comp 04 (PRQA)	Comp 05	Comp 06	Comp 08	Comp 09	最高点
コマンドラインからのチェック対象ルールの変更	0	2	2	3	3	0	0	0	**3**
GUI からのチェック対象ルールの変更	3	3	3	3	n/a	1	3	3	**3**
プロジェクトの解析と解析対象ファイルの除外	3	3	4	4	3	3	4	4	**4**
エラーおよび警告メッセージの品質	3	3	5	6	3	6	3	4	**6**
ツール連携	1	1	2	1	2	1	1	1	**3**
サポート可能な OSの種類	1	1	1	2	1	2	2	2	**2**
自動化	1	1	0	1	1	1	1	0	**1**
追加機能	3	1	3	2	1	3	4	1	**5**
技術サポート	0	3	2	3	2	3	3	3	**3**
合計点	**15**	**18**	**22**	**25**	**16**	**20**	**21**	**18**	**30**

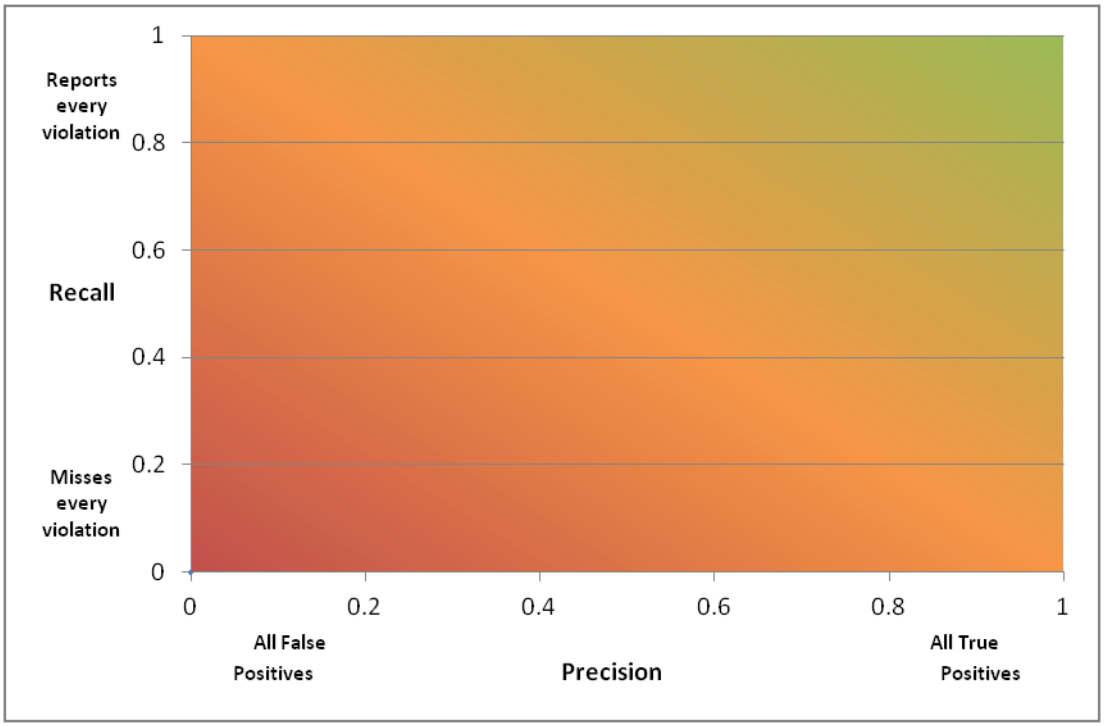
注: 実際の調査では、1 つの基準に対して“Yes”または“No”で判定可能な複数の設問が設けられ、それらの結果に応じて、各基準ごとに 1 つから 3 つの星付け評価が行われました。上表では、それぞれの基準に対して“Yes”と判定された設問および星付け評価の星 1 つにつき 1 点を加算し、その合計点を示しています。なお、表の右側の列に示されている“最高点”は、それぞれの基準において獲得されうる最も高い点数を示します。

1.6 調査結果 –ハードクライテリア

ハードクライテリアを用いた調査の結果からは、それぞれのツールが上述した 11 個の MIRSA-C ルールに対するルール違反を、どれだけ効果的に検出しているかがわかります。ツールごとの違反検出能力は、次の 2 つの観点から計測されます。

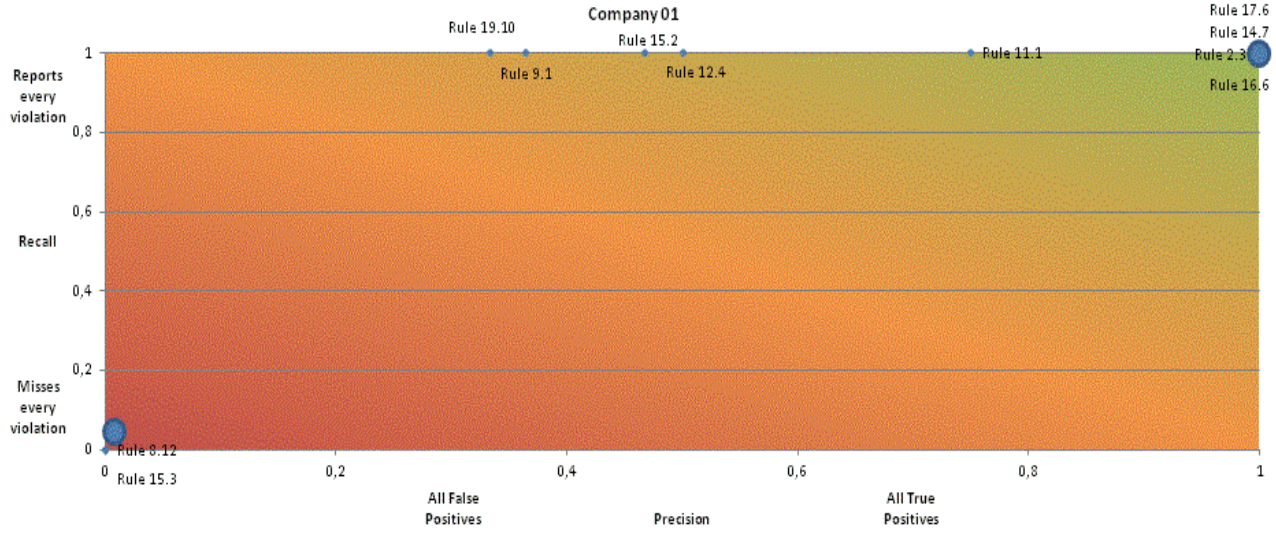
1) テストコードに含まれているルール違反を検出し、報告する能力
2) 誤検出(ノイズ)を報告しない(実際にはルール違反ではない記述を違反として報告しない)能力

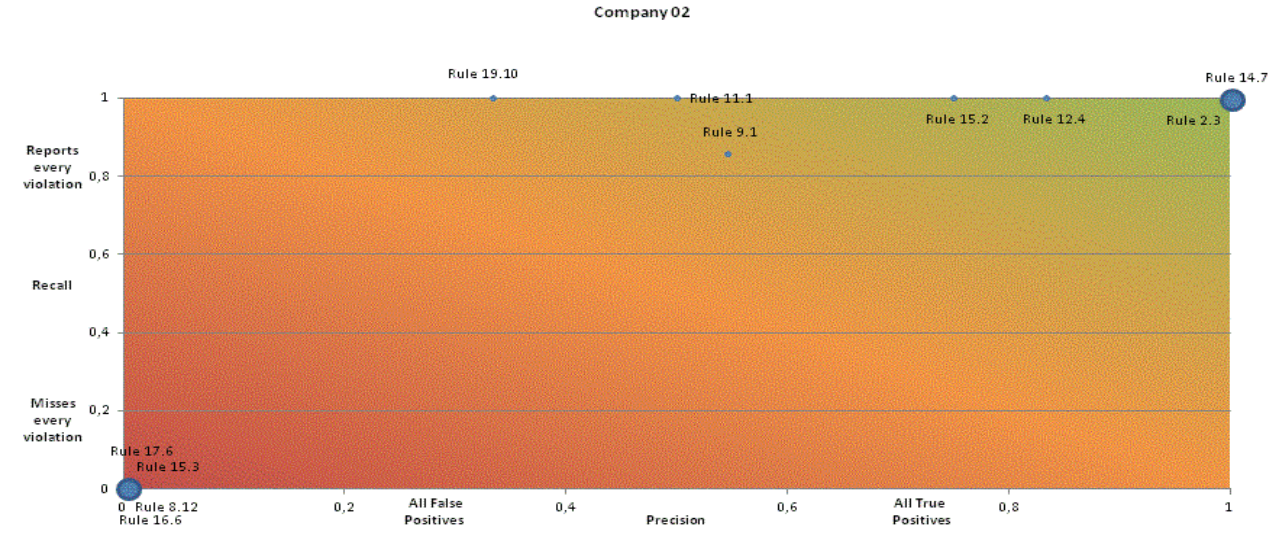
結果は、以下に掲載する一連のグラフにおいて、“検出能力(Recall)”および“検出精度(Precision)”として示されます。グラフ上の Y 軸(検出能力)は、ツールがルール違反を検出する能力を、X 軸(検出精度)は、ツールが誤検出を生成することなく解析する能力を表します。

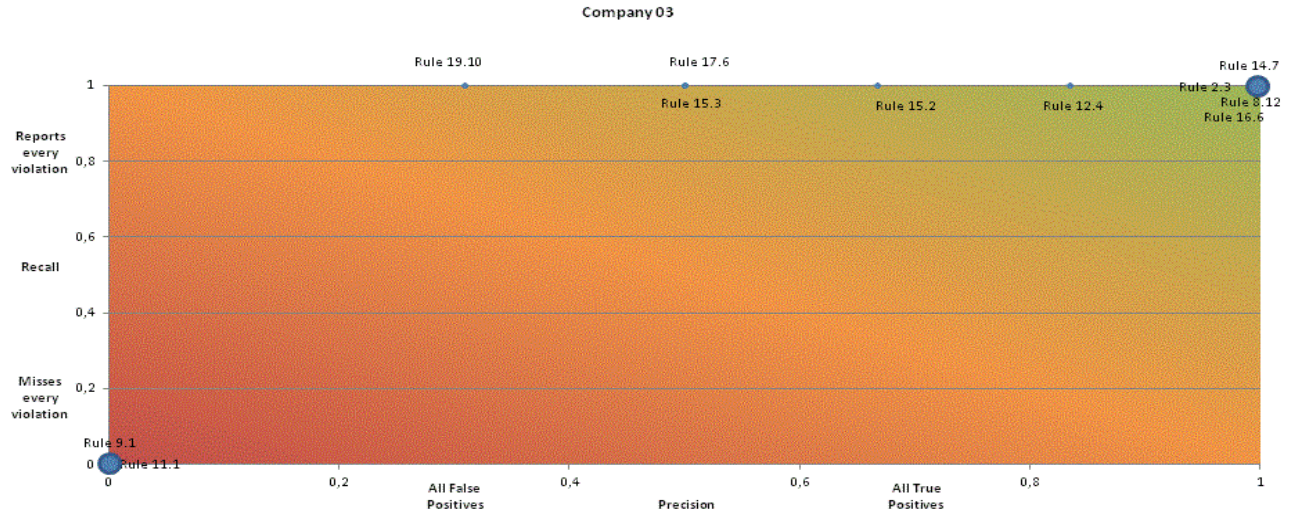


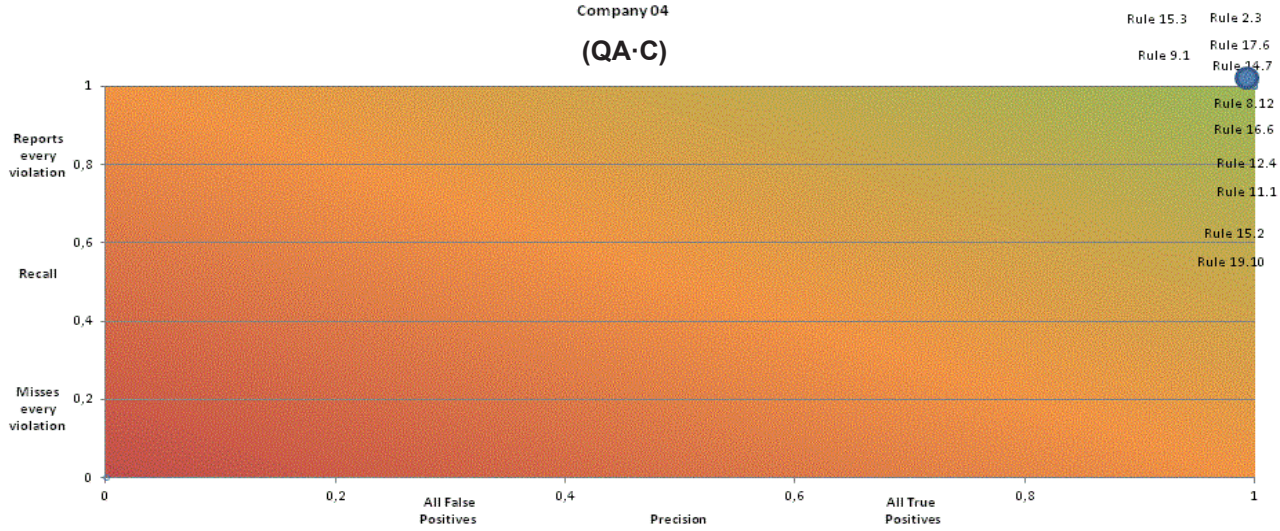
グラフ上では、優れた結果を残したツール(違反をより多く検出し、誤検出の発生率がより低かったツール)ほど、右上の位置に表示されます。つまり、表示される位置が左へ移動するほど、より多くの誤検出が発生したことを意味し、下へ移動するほど、検出された違反の数が少なかった(未検出の違反の数が多かった)ことを意味します。なお、製品名がグラフ上で最も左下の位置に表示された場合、その製品のパフォーマンスが調査対象製品の中で最も低かった(検出した違反の数が最も少なく、かつ、最も多くの誤検出を生成した)ことになります。

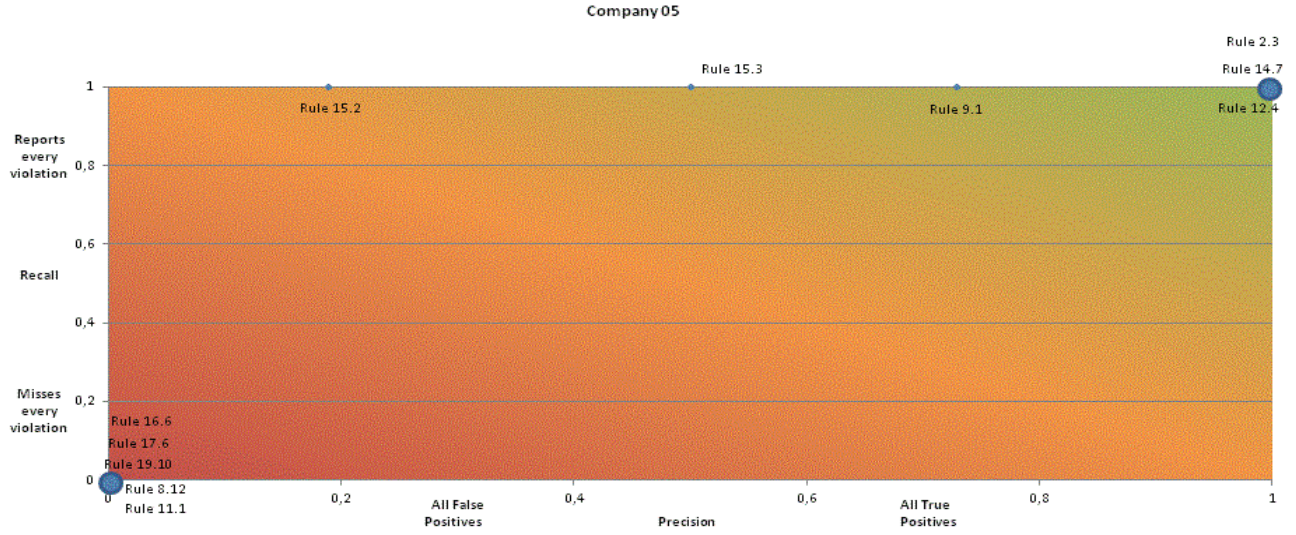
以下に、Coverity を除く 8 製品に対して実施された調査の結果を示します。(これらのグラフは、TERA-Labs によるレポートの付録 L～S に掲載されています。)

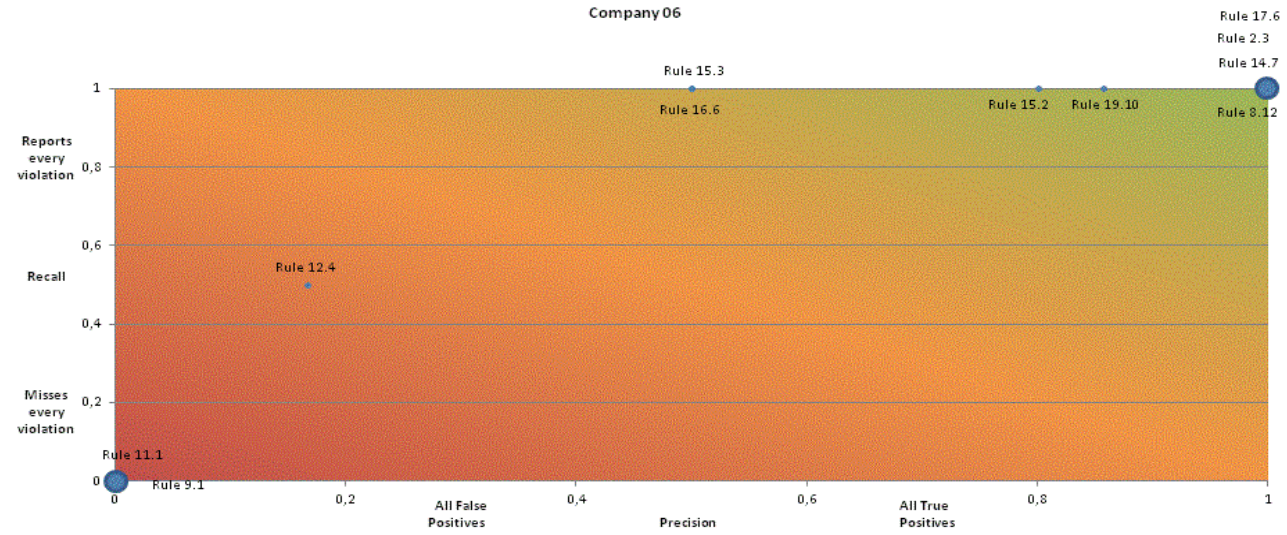

Company 01
Rule 19.10
Rule 15.2
Rule 17.6
Rule 14.7
Rule 2.3
Rule 16.6
Rule 11.1
Rule 9.1
Rule 12.4
Reports every violation
Recall
Misses every violation
1
0,8
0,6
0,4
0,2
0
Rule 8.12
Rule 15.3
0
0,2
0,4
0,6
0,8
1
All False Positives
Precision
All True Positives

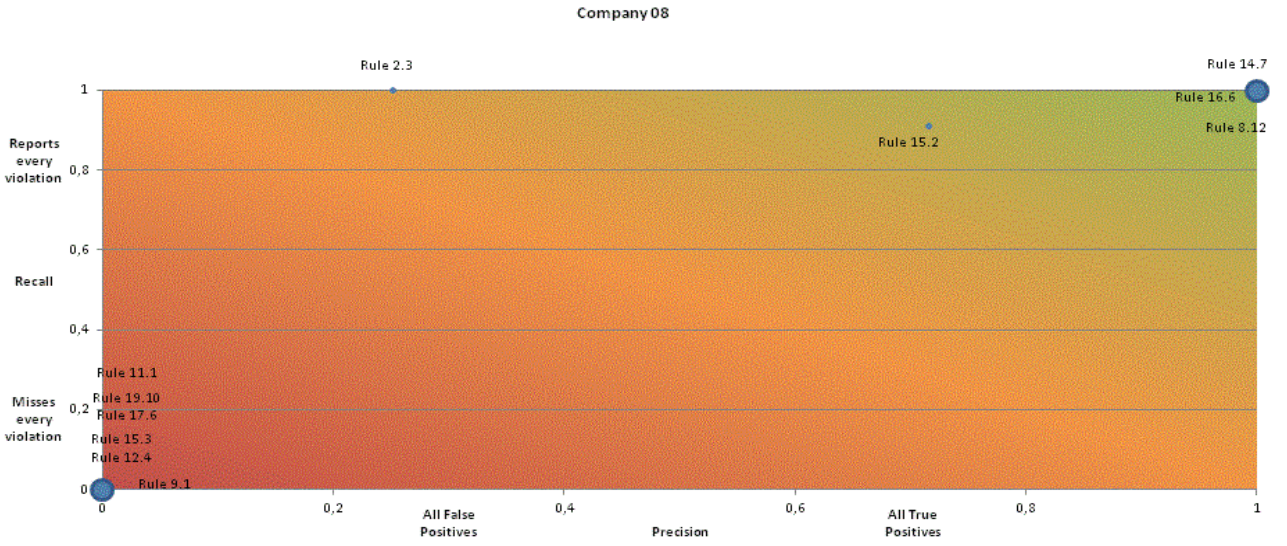

Company 02
Rule 19.10
Rule 14.7
Rule 11.1
Rule 15.2
Rule 12.4
Rule 2.3
Rule 9.1
Reports every violation
Recall
Misses every violation
1
0,8
0,6
0,4
0,2
0
Rule 17.6
Rule 15.3
Rule 8.12
Rule 16.6
0
0,2
0,4
0,6
0,8
1
All False Positives
Precision
All True Positives

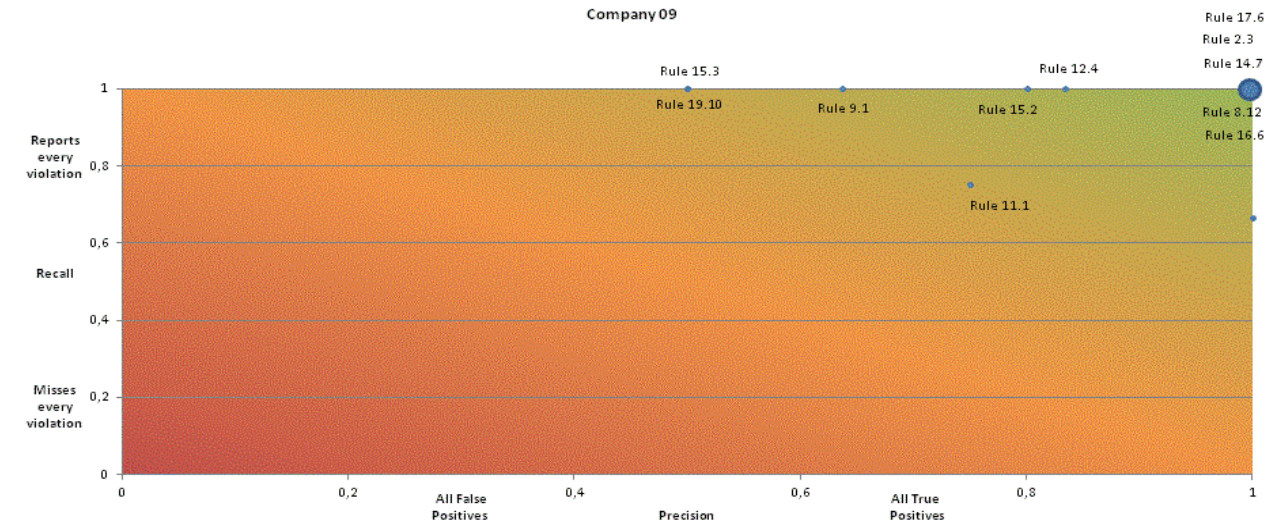

Company 03
Rule 19.10
Rule 17.6
Rule 14.7
Rule 2.3
Rule 15.3
Rule 15.2
Rule 12.4
Rule 8.12
Rule 16.6
Reports every violation
Recall
Misses every violation
Rule 9.1
Rule 11.1
0
0,2
0,4
0,6
0,8
1
All False Positives
Precision
All True Positives


Company 04
(QA·C)
Rule 15.3
Rule 2.3
Rule 9.1
Rule 17.6
Rule 14.7
Rule 8.12
Rule 16.6
Rule 12.4
Rule 11.1
Rule 15.2
Rule 19.10
Reports every violation
Recall
Misses every violation
0
0,2
0,4
0,6
0,8
1
All False Positives
Precision
All True Positives


Company 05
Rule 2.3
Rule 15.3
Rule 14.7
Rule 15.2
Rule 9.1
Rule 12.4
Reports every violation
Recall
Misses every violation
Rule 16.6
Rule 17.6
Rule 19.10
Rule 8.12
Rule 11.1
0
0,2
0,4
0,6
0,8
1
All False Positives
Precision
All True Positives


Company 06
Rule 17.6
Rule 2.3
Rule 14.7
Rule 15.3
Rule 16.6
Rule 15.2
Rule 19.10
Rule 8.12
Reports every violation
Recall
Misses every violation
Rule 12.4
Rule 11.1
Rule 9.1
1
0,8
0,6
0,4
0,2
0
0
0,2
0,4
0,6
0,8
1
All False Positives
Precision
All True Positives


Company 08
Rule 2.3
Rule 14.7
Rule 16.6
Rule 8.12
Rule 15.2
Reports every violation
Recall
Misses every violation
Rule 11.1
Rule 19.10
Rule 17.6
Rule 15.3
Rule 12.4
Rule 9.1
1
0,8
0,6
0,4
0,2
0
0
0,2
0,4
0,6
0,8
1
All False Positives
Precision
All True Positives


Company 09
Rule 17.6
Rule 2.3
Rule 14.7
Rule 15.3
Rule 12.4
Rule 19.10
Rule 9.1
Rule 15.2
Rule 8.12
Rule 16.6
Rule 11.1
Reports every violation
Recall
Misses every violation
1
0,8
0,6
0,4
0,2
0
0
0,2
0,4
0,6
0,8
1
All False Positives
Precision
All True Positives

1.7 TERA-Labs による結論

今回の調査における目的は、調査対象製品の分析およびその結果の提示であり、特定のツールを“最も優れたツール”として推奨することではありませんでした。

しかしながら、TERA-Labs のレポートには、調査結果から導き出された 6 つの結論が記されています。以下に、それらのポイントを要約します。

1. ツールごとに際立った違いが存在する。
    - GUI、コマンドライン、インストール手順、ライセンスの仕組みなど、様々な点において、ツールごとに大きな違いが存在する。
    - チェック可能なルールも、ツールごとに大きくことなる。
2. 多くの MISRA-C ルールに対して、非常に多くの違反メッセージが生成されてしまう(すべてのツールにおいて同様の傾向が見られる)。
3. 多数の誤検出が発生すると予想されていたが、調査ではこの予想に反して、多くの未検出(複数のツールが明らかなルール違反を検出できないなど)が確認された。
4. 一般的に、解析結果の品質はツールの価格に比例する。つまり、価格が高いツールほど、より多くのルール違反を、誤検出を伴わずに検出できる。
5. ほとんどのツールにおいて、注釈付きソースコード(ソースコード + 違反メッセージ + 行番号)を出力ファイルとして生成するためのオプションが提供されていない。そのため、既存の開発プロセスにツールを統合できない場合がある。
6. コードを完全に解析できなかった場合、いかなる解析結果も生成されなかった(すべてのツールにおいて同様)。これは、既存の大規模なコードベースを解析する場合などに、大きな問題となる。

セクション 2 : PRQA による見解およびコメント

このセクションでは、TERA-Lab のレポートに対する PRQA の見解およびコメントを示します。なお、本セクションの内容には、TERA-Lab のレポート作成者と PRQA の間で直接交わされた議論の内容が反映されています。

概要

今回の調査は、MISRA-C 準拠度チェックツールの性能を、第三者の立場から公平かつ公正に評価したものだと言えます。(レポートには、いくつかの誤りおよび不明確な記述が含まれていましたが、そのどちらも調査結果を大きく左右するものではありませんでした。)

以下に、TERA-Labs の研究チームによる声明を引用します。
「各社の販促資料によれば、今回調査を行ったすべてのツールが包括的な MIRSA-C 対応を謳っており、ツールごとの性能差は存在しないかのような印象を受ける。しかし、そのような認識は事実と大きく異なる」

調査には、1 人年以上におよぶ十分な時間と労力が費やされ、客観的な結論が導かれています。これは、ツールを選択するための調査として、営利企業ではまず実現することができないレベルの調査と言えるでしょう。

テストケース

今回の調査では、MISRA-C コーディング標準から選択された 11 個のルールに対してテストケースが作成されました。これらのルールは、いずれも MISRA-C コーディング標準を代表する重要なルールであり、その選択は妥当なものであったと思われます。

テストケースは、TERA-Labs によって作成された比較的単純なもの（複雑なコードや難解な論理、あるいは一般的でないエッジケースを含まないもの）が使用されました。そのため、ツールによっては、コーディング違反の基本的な例は検出できるものの、より大規模で複雑なコードベースが解析の対象となった場合に、その性能が大きく低下するものがあるかもしれません。

ハードクライテリアを用いた調査の結果について

今回の調査では、2 つの基準（ルール違反を検出する能力と誤検出を出力しない能力）を用いてツールの性能および有効性が評価されましたが、これらの基準の設定は非常に適切なものだった言えるでしょう。グラフに示された分析結果も、各ツールの性能を端的に表していたと評価できます。

また、QA·C はすべてのルール違反を誤検出を伴うことなく検出し、調査対象となったツールの中で最も高い評価を獲得することができました。

この結果からは、以下の 2 つの点を指摘することができます。

- 調査結果を示したグラフにおける X 軸(検出精度)は、ツールの“誤検出を生成しない能力”を表します。つまり、製品名の表示される位置が左へ移動するほど、より多くの誤検出が発生したことを意味します。誤検出が発生した場合、開発者は検出された警告の内容を評価したうえで原因を特定し、対応の是非を判断しなければならず、これらの作業を行うために余計な時間とコストがかかります。もちろん、小規模なサンプルコードに対して限られた数のルールを適用する場合、誤検出の発生は大きな問題ではありません。しかしながら、大規模なコードに対して多くのルールを適用する場合、誤検出の発生は、時間とコストの両面において深刻な影響を及ぼします。さらに、多くの誤検出を生成するツールは開発者から信用されず、利用される機会が減少することも十分に考えられます。

- 調査結果を示したグラフにおける Y 軸(検出能力)は、ツールの“ルール違反を検出する能力”を表します。つまり、製品名の表示される位置が下へ移動するほど、検出された違反の数が少なかった(未検出の違反の数が多かった)ことを意味します。検出能力の低さは、誤検出の発生よりも、潜在的に多くの危険を孕んでいます。そのため、検出能力の低いツールは、コーディング標準への準拠を達成するという静的解析ツール本来の目的にかなっていないと言えるでしょう。また、 今回の調査では、多くのツールにおいて事前の予想を上回る数の未検出項目が確認されました。しかしながら QA·C は、その他の製品が検出できなかった重要な問題の検出に成功しており、この点からも、QA·C の他社製品に対する優位性が証明されたと言えるでしょう。

今回の結果からの推測

TERA-Labs および PRQA の見解は、今回の調査結果から推測される次の 2 つの点において一致しています。

1) 静的にチェック可能な 133 個の MISRA-C ルールに対して今回と同様の調査を行った場合でも、それぞれのツールが示す相対的パフォーマンスは、今回の調査結果とほぼ変わらないことが推測される。

2) 今回の調査で示されたベンダー間の相対的な性能は、今回の調査と同様の手法によって MISRA-C++への準拠度チェック能力を調査する場合にも、同様の傾向を示すものと推測される。

投資収益率(ROI)

今回の調査の主な目的は、ツールの技術的な性能を評価することであり、それぞれのツールを使用した場合に必要となる作業時間およびコストに関しては、調査が行われませんでした。言うまでもなく、誤検出率の高いツールを使用した場合、誤検出率の低いツールを使用する場合に比べて、より多くの作業時間とコストがかかります。未検出に関しては、未検出となった違反を見つけるためだけでなく、未検出となった違反を検出するために新たなテスト手法を導入する必要が生じるため、膨大な作業時間とコストの増加が発生します。ツールごとに異なる作業時間およびコストを検証するには、費用対便益分析(CBA)および投資収益率(ROI)の分析が必要ですが、残念ながら今回の調査では触れられていません。しかしながら、作業時間およびコストが、ツールを選定する際の重要なポイントとなることは、認識しておく必要があります。

プロセス

PRQA が TERA-Labs の研究チームと行った議論では、それぞれのソフトウェア開発現場に合わせて設計された構造化された開発プロセスを導入することで、MISRA および MISRA 準拠度チェックツールをより有効に活用できることが確認されました。

コーディング標準

多くの開発チームは、MISRA への完全な準拠を目指してコードを開発していますが、一方で、未定義の動作などの重要な不具合を集中的に検出するため、また、より効果的なレビューを実現するための手段として、主要な MISRA ルール(またはその他のコーディング標準)のみを適用しているチームも存在します。後者の目的は“MISRA 準拠のコード”の開発ではなく、MISRA ルールを利用することにより“堅牢なコード”を開発することにあると言えるでしょう。

MISRA ルールを導入するための有効かつ一般的な方法は、主要なルールのサブセット(TERA-Labs による調査で選択された 11 個のルールなど)の適用から始めて、コーディング規約の重要性が開発チームのメンバーに浸透するに従い、適用するルールの数を増やしていく方法です。(PRQA のツールを利用した場合、MISRA ルールのサブセットの選択および、それぞれの企業に固有のルールのルールセットへの追加を、簡単に行うことができます。)

ルールがそれぞれの企業における内部的なものであれ、一般的な標準規格であれ、それぞれのルールへの準拠を効果的かつ自動的にテストできる静的解析ツールの役割は、十分に認識しておく必要があります。

確かな技術力に裏打ちされた PRQA の製品群

PRQA と MISRA は、20 年以上にわたり有力なパートナーとして歩んできました。そのパートナーシップの深さは、PRQA のコーディング標準を基に MISRA-C および MISRA-C++ガイドラインの主要なルールが作成された事実や、MISRA-C ワーキンググループの発足当初から現在に至るまで、PRQA の技術者がその中心的なメンバーとして活動を続けてきたことからもうかがい知ることができます。PRQA の静的解析ツールは、そのような両者のパートナーシップにおいて中心的な役割を担ってきた言語の専門家が、設計および開発を担当しています。

まとめ

TERA-Labs により実施された今回の調査は、MISRA-C 準拠度チェックツールの性能を、第三者の立場から公平かつ公正に評価したものでした。そのような調査において、QA·C は、他の製品が検出できなかった重要な問題を「誤検出をほとんど(あるいは 1 つも)伴うことなく 検出する」ことに成功し、調査対象となったツールの中で最も高い評価を獲得することができました。静的解析ツールの導入を検討している企業においては、今回の調査結果を踏まえたうえで、品質の低い(ルール違反の未検出および誤検出の発生率が高い)ツールを導入した場合の悪影響(コスト、作業時間、品質、投資収益率)について、改めて検討することをお奨めします。

参照文献:

Ref 1 : Marijn Temmerman, Hugo Van Hove, Kris Bellemans, “KRICODE RESEARCH REPORT I: COMPARATIVE STUDY OF MISRA-C COMPLIANCY CHECKING TOOLS”, Version 1.0, Final, May 9th 2012

PRQA への連絡先: **Email:** info@programmingresearch.com **Web サイト:** www.programmingresearch.com