



atollic



atollic ab

Science Park Jönköping  
Djuterigatan 7  
SE-553 18 Jönköping  
Sweden

atollic inc.

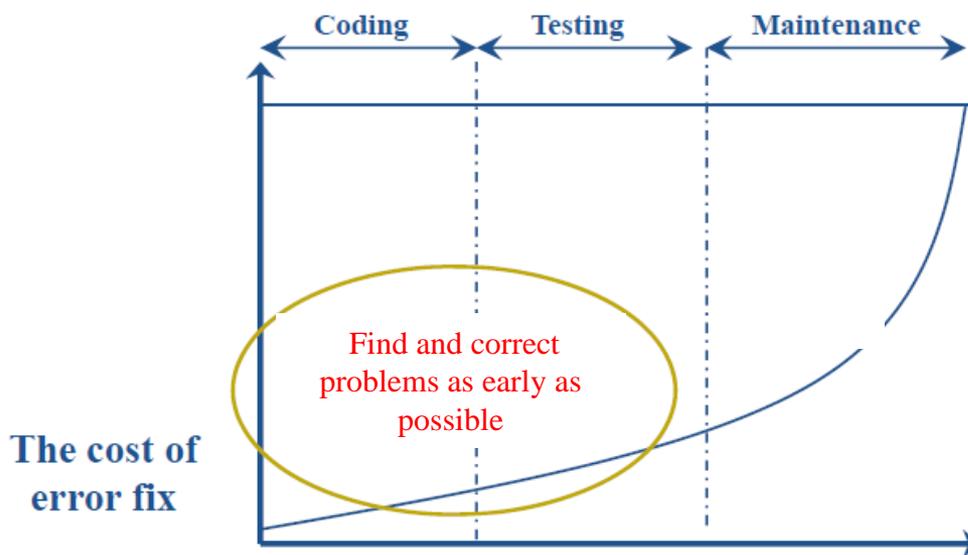
115 Route 46  
Building F, Suite 1000  
Parsippany  
N.J. 070504  
USA

[www.atollic.com](http://www.atollic.com)

# Atollic TrueINSPECTOR®

*Improve software quality with static source code inspection!*

# Software errors are more expensive to find & fix later



By finding bugs earlier, you reduce cost, development time and secure your company/product reputation



# MISRA<sup>®</sup> -C:2004

Reduce errors by limiting developer flexibility



<b>What is it?</b>	A coding standard developed by the automotive industry for improving reliability, safety, maintainability and portability of safety-critical systems.
<b>Rule classifications</b>	The rules can be classified as either being Required or only Advisory to use.
<b>141 rules in 21 rule groups</b>	Rule grouping for Environment, Language extensions, Documentation, Character sets, Identifiers, Types, Constants, Declarations and definitions, Initialization, Arithmetic type conversions, Pointer type conversions, Expressions, Control statement expressions, Control flow, Switch statements, Functions, Pointers and arrays, Structures and unions, Preprocessing directives, Standard libraries and Runtime failures.

# Code metrics

Reduce errors by limiting code complexity



<b>How is it gathered?</b>	Code metrics is gathered by performing a static analysis of the source code.
<b>What is the purpose?</b>	Information on code complexity in different parts of the program gives hints on where the code needs to be simplified, thus reducing risk of errors and improving maintainability.
<b>What is the benefit?</b>	By simplifying complex sections of code, the likelihood of errors are reduced. This results in reduced costs, improved maintainability and higher software quality.



# Atollic TrueINSPECTOR<sup>®</sup>

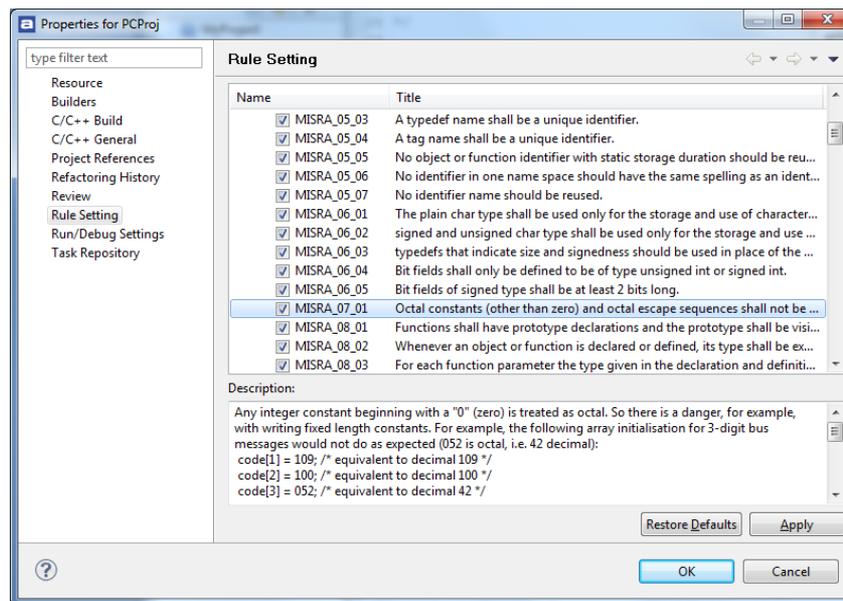
A tool for professional source code analysis

The screenshot displays the Atollic TrueINSPECTOR interface. The main window shows a C source code file with several lines of code, including comments and function calls. A yellow box highlights a comment: "Multiple markers at this line". The right-hand pane shows a "Rule Description" for MISRA\_05\_07, which states: "No identifier name should be reused." Below this, a "Bad Example" is provided: "struct ar\_speed { uint8\_t speed; /\* knots \*/ } \*x; struct spd\_speed { uint8\_t speed; /\* mph \*/ /\* Not Compliant - speed is in different units \*/ } \*y; x->speed = y->speed;". The bottom-left pane shows an "Inspection Summary" table with 4082 total violations. The bottom-right pane shows a "[Rule] Distribution violations (%)" pie chart.

Rule ID	Count	Rule title
MISRA_17_01	45	Pointer arithmetic shall only be applied to pointers that address an array or array element.
MISRA_12_01	34	Limited dependence should be placed on C's operator precedence rules in expressions.
MISRA_08_01	31	Functions shall have prototype declarations and the prototype shall be visible at both ...
MISRA_14_07	30	A function shall have a single point of exit at the end of the function.
MISRA_16_07	25	A pointer parameter in a function prototype should be declared as pointer to const if ...
MISRA_12_08	22	The right-hand operand of a shift operator shall lie between zero and one less than th...
MISRA_11_03	19	A cast should not be performed between a pointer type and an integral type.
MISRA_14_10	18	All if ... else if constructs shall be terminated with an else clause.
MISRA_06_07	17	Objects shall be defined at block scope if they are only accessed from within a single f...
MISRA_10_07	15	A function should be used in preference to a function-like macro.
MISRA_05_06	12	No identifier in one name space should have the same spelling as an identifier in anot...
MISRA_10_17	11	All #else, #elif and #endif preprocessor directives shall reside in the same file as the #if...
MISRA_14_03	10	A null statement shall only occur on a line by itself.
MISRA_11_04	8	A cast should not be performed between a pointer to object type and a different point...
MISRA_12_04	8	The right-hand operand of a logical && or    operator shall not contain side effects.
MISRA_12_13	7	The increment (++) and decrement (--) operators should not be mixed with other ope...
MISRA_14_08	6	The statement forming the body of a switch, while, do ... while or for statement shall ...

- Analyzes source code
- Validates the source code against the MISRA<sup>®</sup>-C:2004 coding standard
- Present code metrics (such as cyclomatic value of code complexity)
- Detects potential bugs and helps improve software quality

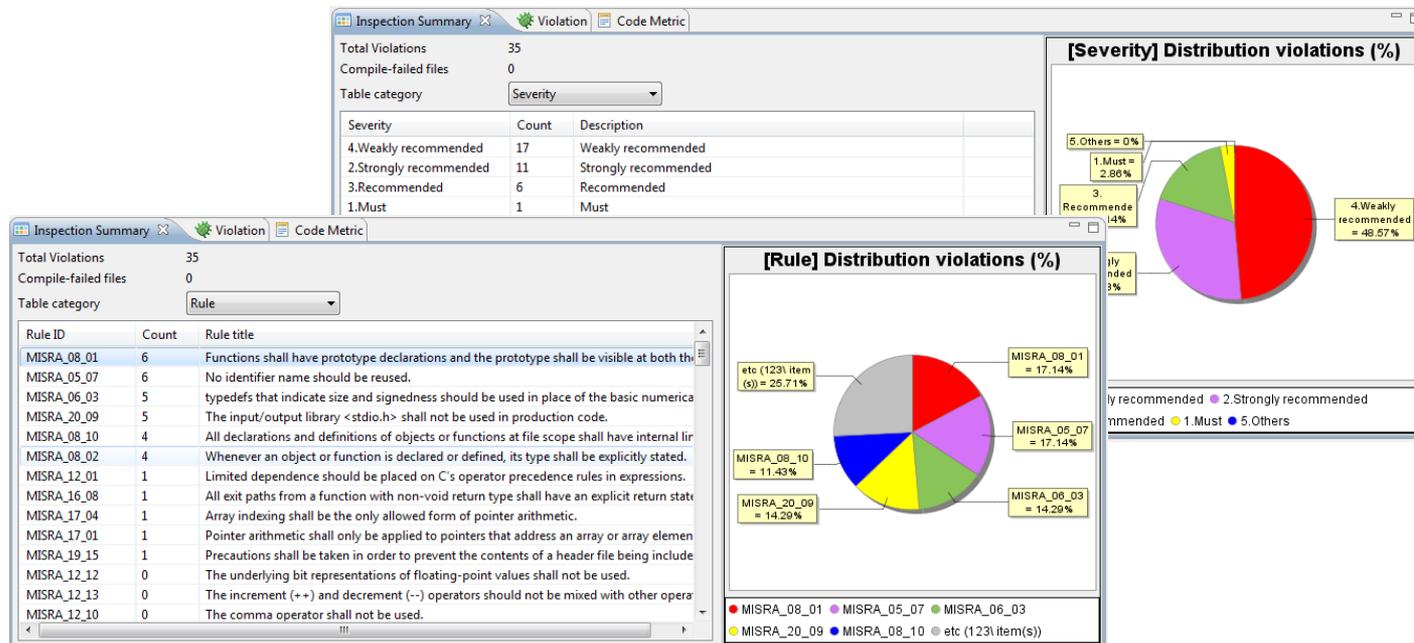
# Selecting rules



- Atollic TrueINSPECTOR® allows individual selection of rules in a formal rule database
- Each rule is annotated with a clear explanation of what it does, in addition to its formal name and title



# Inspection summary



- Summary display of Rules, Files and Severity results
- Pie charts gives excellent overview
- Text reports gives detailed information
- Different sorting options

# Rule violations

Diagnosis	Rule	Severity	Source	Line	Fun
MISRA_05_07 (6 items)					
An identifier 'i' is not unique identifier.(declared at[line:35, file:C:\Users\Magnus\Atollic\TrueSTU	MISRA_05_07	★★	main.c	61	add
An identifier 'i' is not unique identifier.(declared at[line:35, file:C:\Users\Magnus\Atollic\TrueSTU	MISRA_05_07	★★	main.c	79	sub
An identifier 'i' is not unique identifier.(declared at[line:35, file:C:\Users\Magnus\Atollic\TrueSTU	MISRA_05_07	★★	main.c	91	mul
An identifier 'xx' is	ueST MISRA_05_07	★★	main.c	59	add
An identifier 'xx' is	ueST MISRA_05_07	★★	main.c	77	sub
An identifier 'xx' is	ueST MISRA_05_07	★★	main.c	89	mul
MISRA_06_03 (5 items)					
MISRA_08_01 (6 items)					
MISRA_08_02 (4 items)					
A variable type is n	MISRA_08_02	★★★	main.c	55	calc
A variable type is n	MISRA_08_02	★★★	main.c	59	add
A variable type is n	MISRA_08_02	★★★	main.c	77	sub
A variable type is n	MISRA_08_02	★★★	main.c	89	mul
MISRA_08_10 (4 items)					
MISRA_12_01 (1 item)					

**Export View Content**

File name: MyProject

Use default location

Location: C:\MISRA\Reports

File Extension

- DOC (.doc)
- HTML (.html)
- PDF (.pdf)
- PPT (.ppt)
- XLS (.xls)

Select All Deselect All

Export Cancel

**RuleExplanation**

➤RULE : No identifier name should be reused.

➤DESCRIPTION : Regardless of scope, no identifier should be re-used across any files in the system. This rule incorporates the provisions of Rules 5.2, 5.3, 5.4, 5.5 and 5.6.

Where an identifier name is used in a header file, and that header file is included in multiple source files, this rule is not violated. The use of a rigorous naming convention can support the implementation of this rule.

-----

1. All identifiers shall be unique identifier.

▶Bad Example◀

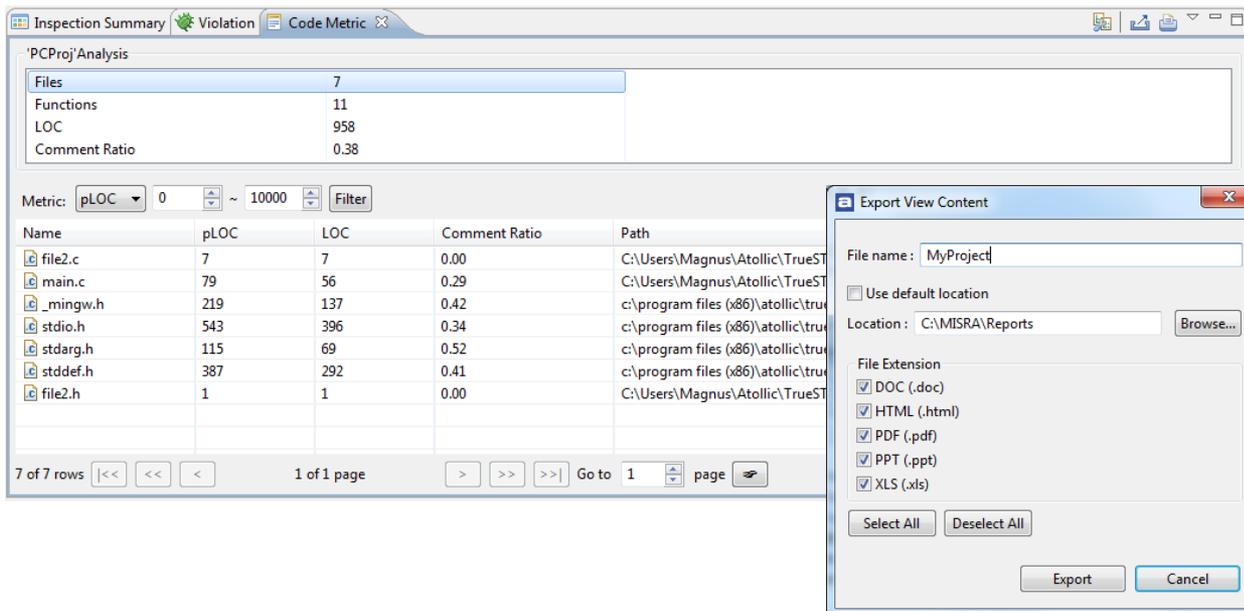
```
struct air_speed {
    uint16_t speed; /* knots */
} * x;
struct gnd_speed {
    uint16_t speed; /* mph */ /* Not Compliant - speed is in different units */
} * y;
x->speed = y->speed;
```

▶Good Example◀

```
struct air_speed {
    uint16_t a_speed; /* knots */
```

- List inspection violations per rule
- Explanation of rule details, the problem, and suggested solution
- Navigation links for every violation (jump to offending line in editor)
- Searching/filtering
- Export reports to MS-Word, MS-Excel, MS-PowerPoint, HTML and PDF

# Code metrics



The screenshot displays a software interface for code metrics analysis. The main window, titled 'PCProj' Analysis, shows a summary of metrics:

Metric	Value
Files	7
Functions	11
LOC	958
Comment Ratio	0.38

Below the summary, a detailed table lists individual files with their respective metrics. The 'Metric' is set to 'pLOC' with a range from 0 to 10000. The detailed table is as follows:

Name	pLOC	LOC	Comment Ratio	Path
file2.c	7	7	0.00	C:\Users\Magnus\Atollic\TrueST
main.c	79	56	0.29	C:\Users\Magnus\Atollic\TrueST
_mingw.h	219	137	0.42	c:\program files (x86)\atollic\tru
stdio.h	543	396	0.34	c:\program files (x86)\atollic\tru
stdarg.h	115	69	0.52	c:\program files (x86)\atollic\tru
stddef.h	387	292	0.41	c:\program files (x86)\atollic\tru
file2.h	1	1	0.00	C:\Users\Magnus\Atollic\TrueST

An 'Export View Content' dialog box is open, showing the following options:

- File name: MyProject
- Use default location
- Location: C:\MISRA\Reports
- File Extension:
  - DOC (.doc)
  - HTML (.html)
  - PDF (.pdf)
  - PPT (.ppt)
  - XLS (.xls)
- 
- 

- Display of source code metrics
- Number of files, functions, lines and comment line ratio
- Cyclomatic value of code complexity
- Export reports to MS-Word®, MS-Excel®, MS-PowerPoint®, HTML and PDF



atollic™

# DEMO

Improve your software quality with  
Atollic TrueINSPECTOR®!

[www.atollic.com](http://www.atollic.com)

*"Embedded passion"*



# Contact Us

atollic<sup>TM</sup>

- **Italy**

**Fenway Embedded Systems**

Via Don Giovanni Minzoni, 31

20010 Arluno (MI) - Italy

Tel. +39 02 97310120

*Email:* sales@fenwayembedded.com

*Web:* www.fenwayembedded.com



- **Headquarter**

**Atollic AB**

Science Park Jönköping

Gjuterigatan 7

SE-553 18 Jönköping – Sweden

*Email:* sales@atollic.com

*Web:* www.atollic.com

atollic ab

Science Park Jönköping

Gjuterigatan 7

SE-553 18 Jönköping

Sweden

atollic inc.

115 Route 46

Building F, Suite 1000

Parsippany

NJ, 070504

USA

[www.atollic.com](http://www.atollic.com)

