Vol. 8, No. 2, pp. 31-36, December 2009

INTERNATIONAL JOURNAL OF SAFETY
www.kosos.or.kr

A Study of Software Coding Rules Inspection Tool for Railway Signaling Software Safety

Jong-Gyu Hwang* and Hyun-Jeong Jo

Department of Train Control & Communication Research, Korea Railroad Research Institute, Uiwang-city 437-050, Korea

(Received May 11, 2009; Accepted October 21, 2009)

Abstract : In accordance with the development of recent computer technology, railway signaling software became more complex for the intellectualization. Therefore the importance and dependency of railway signaling system on the computer software is getting more increased further, and the testing for the safety and reliability of railway signaling system software became more important. It is started to become influential as very important issue for the reliability and safety of vital embedded software like railway signaling system. The software coding which can have an effect on the safety at the coding level of software shall not be included preferentially, for the safety of software, and must be checked. This thesis suggested an automated testing tool for coding rules on this railway signaling system software, and presented its applied result for railway signaling system software. The testing items in the implemented tool had referred to the international standards in relation to the software for railway system and MISRA-C standards. This automated testing tool for railway signaling system can be utilized at the assessment stage for railway signaling system software also, and it is anticipated that it can be utilized usefully at the software development stage also

Key words: software testing tool, coding rules inspection, software safety

1. Introduction

Accordingly, the dependency of railway signaling system on the computer software is being increased rapidly according to the development of recent electronic and computer technology. Though magnitude and complexity of railway signaling system software is slower than the development speed of hardware, it is anticipated that the size will be bigger gradually and the complexity will be increased also. Railway signaling system software became more complex for the intellectualization and automation, the importance took by software in the railway signaling system is getting more increased. It is started to become influential as very important issue for the reliability and safety of vital embedded software like railway signaling system. The software coding which can have an effect on the safety at the coding level of software shall not be included preferentially, for the safety of software, and must be checked.

In the industry field other than railways, the coding rules and standards for embedded control source code in the automobile field which is one of the representative embedded systems are standardized in the form of MISRA-C. Like MISRA coding rules, the automobile field is applying internally authorized coding standards by making their lists, but in the aviation or railway field, the formalized coding rules or standards are not defined yet. Therefore, currently most of the railway system software is using MISRA-C as coding rules.

The software safety requirements necessary for railway signaling system are being standardized by IEC 61508-3 and IEC 6227 [1][2] like fig. 1, and the activities which must be performed by system development life cycle stage to secure the safety of vital software such as railway signaling system are being explained in this standard. Besides, according to this standard, the software for embedded system of railway signaling system is made to develop its source code to be suitable for proper coding standards and validate it also. As mentioned above there is not regulated software coding rules or standards for railway signaling system. Therefore the basic requirements which must be observed by railway software are being presented descriptively in these two standards. And also there is no automated

*Corresponding author: jghwang@krri.re.kr

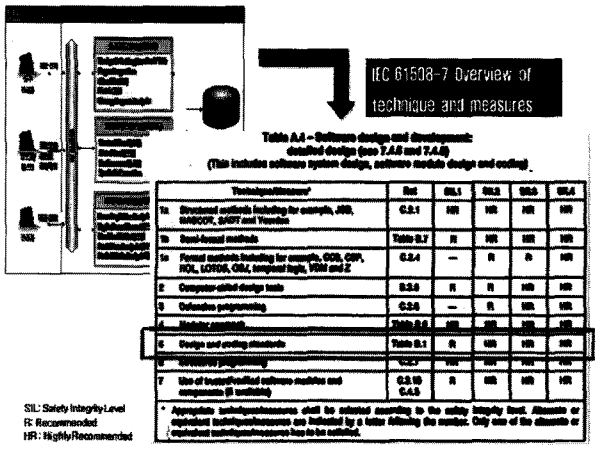

Fig. 1. Needs of Coding Rules Inspection for Software Safety by Int'l Std.

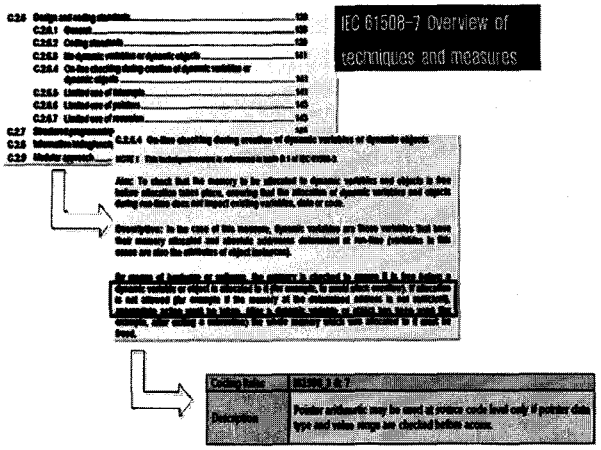

Fig. 2. Overview of Coding Rules Deduction for Railway Signaling System.

testing tool for coding rules to evaluate railway signaling system software yet in the overseas as well as in our country. Most of commercial software code inspection tools are made to inspect MISRA-C coding rules only [3], or made possible to inspect several rules designated randomly by some testing tool developing companies only. Therefore, the automated testing tool for coding rules for vital software such as railway signaling system has not developed yet [4][5][6].

This study developed an automated testing tool to inspect coding rules for testing of the railway signaling system software. For this purpose, the coding rules were drawn through IEC 61508-3 and IEC 62279 standards which are the standards in relation to the safety of railway system software. And the MISRA-C coding rules also applied to the developed testing tool. And we also applied developed automatic tool to real railway signaling system, which have developed in our country.

2. Coding Rules for Railway Signaling Software

Automated tool for inspection of coding rules made on IEC 61508 & IEC 62279 which are the international standards to be observed at time of software development for railway system as its basis and also MISRA-C coding standards which are being widely used in the automobile industry as the standard to be observed at the time of built-in software development by using C language. Especially, in case of IEC 61508 and IEC 62279, their levels are conceptional, comprehensive and they do not define any concrete design or coding rules. Therefore, this study drew and implemented the coding rules through analyzing relevant standards. MISRA-C which is the coding rules for automobile field is presenting total of 141 coding rules. Among the whole coding rules presented by this MISRA-C, the items possible to be automated were implemented in this development tool.

Fig. 1 shows the overview of coding rules deduction from IEC 61508 and IEC 62279. It is generally recommended the requirement for software coding for embedded software of railway signaling system in these two international standards. In this study we analysis the related comments and phrase in two standards and deduced coding rules like followed.

- Deducted coding rules from IEC 61508

IEC 61508 defines the life cycle of railway system and RAMS requirements according to it, and is consisted of total 7 parts. Among them, part 7 deals with the method to measure systems (HW/SW). The configuration of part 7 is as follows. Among them, the Annex C includes the explanation on the measurement of safety and technology of software, and the following rules were analyzed and drawn on this basis.

- MISRA-C Coding Rules

MISRA-C which is the coding rules for automobile field is presenting total of 141 coding rules. Among the whole coding rules presented by this MISRA-C, the items possible to be automated were implemented in this development tool. IEC 61508 defines the life cycle of railway system and RAMS requirements according to it, and is consisted of total 7 parts. Among them, part 7 deals with the method to measure systems (HW/SW). The configuration of part 7 is as follows. Among them, the Annex C includes the explanation on the measurement of safety and technology of software, and the following rules were analyzed and drawn on this basis.

Table 1. MISRA-C Coding Rules

MISRA-C item	Contents of rules
1.1	To check ISO 9989:1990
2.1	Whether assembly languages were encapsulated or not
2.2	Permits C comment style only
2.3	No use of nested comment (C style)
3.4	To inspect whether any non-standard #pragma directive was used
4.1	Permissible escape character
...	...

• Deducted coding rules from IEC 61508

IEC 61508 defines the life cycle of railway system and RAMS requirements according to it, and is consisted of total 7 parts. Among them, part 7 deals with the method to measure systems (HW/SW). The configuration of part 7 is as follows. Among them, the Annex C includes the explanation on the measurement of safety and technology of software, and the following rules were analyzed and drawn on this basis.

• Deducted coding rules from IEC 62279

IEC 62279 is consisted of total 17 chapters, and among them, the following rules were drawn from '10. Software design and implementation', '11. Software verification and testing', '14. Software assessment.'

Table 1. IEC 61508 Supported Rules

61508 items	Contents of rules (relevant sentence)	Detailed contents of rules
C.2.6.3	Prohibition function ("If dynamic variables or objects are not used, these faults are avoided.")	Dynamic memory variables/memory creation function are prohibited
C.2.6.4	Memory return ("the whole memory which was allocated to it must be freed.") Error handling routine ("if allocation is not allowed, appropriate action must be taken.")	To check if memory allocation/return function couple is matched
C.2.6.6	Limited used of Pointer	Pointer arithmetic may be used at source code level only if pointer data type and value range are checked before access.
...	...	...

Table 2. IEC 62279 Supported Rules

62279 items	Contents of rules (relevant sentence)	Detailed contents of rules
Table1.10 Design and Coding Standards	Prohibition function ("3. No Dynamic Objects 4. No Dynamic Variables")	Dynamic memory variables/memory creation function are prohibited
	...	...
	No use of certain kind of sentence ("7. No unconditional jumps")	No use of goto
...	...	...

3. Development of SW Inspection Tool

The automatic coding rules inspection tool was implemented by this study. Though there is no coding standard for railway signaling system software of railways yet, since this had drawn the software coding standard through analysis on the coding rule of automobile field which is being generally applied in the overseas and IEC standard in relation to the safety of railway system software.

The developed inspection tool for coding rules is the module which enables railway signaling system software to judge if it is suitable for the required rule to a certain extent by checking whether it observes coding rules required by relevant standards, etc. such as IEC 61508 and IEC 62279, and basically its purpose is to be utilized at the assessment stage for railway signaling system software, but the architecture was designed so that it can be utilized at the software development stage also. This section describes the architecture of inspection tool for the automated source code analysis tool of railway signaling system. Inspection Tool for coding rules is the module which makes railway signaling sys-

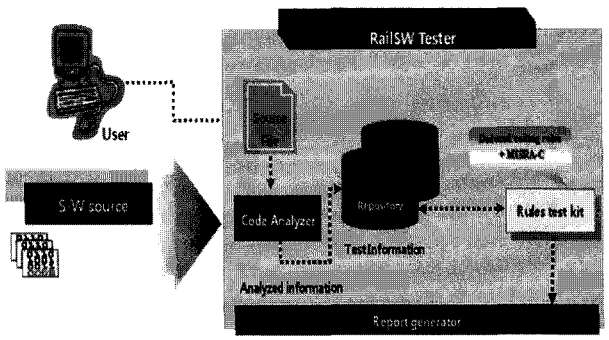

Fig. 3. Configuration of Developed Module.

tem software judge if it is suitable for the required rule to a certain extent by checking whether it observes coding rules required by relevant standards, etc.

Fig. 3 is the one showing the configuration diagram of developed inspection tool for railway signaling system software suggested through this study. As shown in the figure, it was designed to present results through report generator after analyzing whether the object source code input through prescribed coding rule test kit is suitable for the required coding standard by analyzing the input source code, if the source code targeted to be tested is being input into the automated testing tool. Since there is no prescribed coding rule for railway signaling system in the overseas as well as in our country yet, the functional architecture was designed with taking into consideration of extensibility so that it can inquire of or edit coding rules to be applied to the testing in functional module.

On the basis of the system and functional architecture, screen design and design contents of coding standard of the inspection tool for railway signaling system software explained so far, the automated testing tool was implemented. Fig. 4 shows the new rule creating window in the inspection module developed through this study. As explained in the previous section, the coding rules for railway signaling system software was not prescribed yet, and it is allowed to apply MISRA-C rules, otherwise to apply safety requirements of IEC 61508 or IEC 62279. Accordingly, this study drew and implemented coding rules from MISRA-C coding rules and IEC 61508 & IEC 62279.

And since there is no stipulated coding standard yet, it was made to create new rules by adopting or rejecting some parts from existing rule lists like fig. 4 so as to select properly at the testing process. That is, it was made to enable coding rules being required to the source code which becomes a testing target to be selected through coding rule setting windows as shown in the figure, and to enable the new rule set by project to be edited by using the rule set of coding rules which was database previously. Besides, the developed tool implemented the function possible to input new rules also in consideration of the extensibility for coding rules that would be prescribed in the future. Fig. 5 shows the window which performed actual testing with random source code as its target through developed inspection module. The whole violated rules are shown in the upper end of window in the figure, and if one of them was selected, it is made to point at the violated source code part like the window at the right lower end of figure. The left lower end of figure is the one showing the control flowchart of source code through analyzing those input source codes, and it was made to enable whole architectures of target source code for testing to

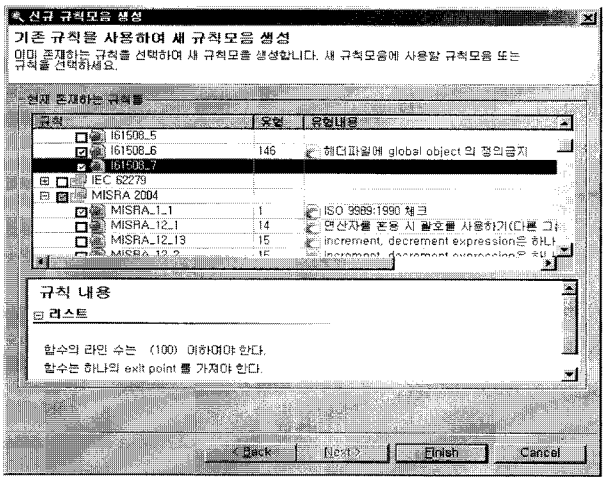

Fig. 4. Coding Rules Editing Windows.

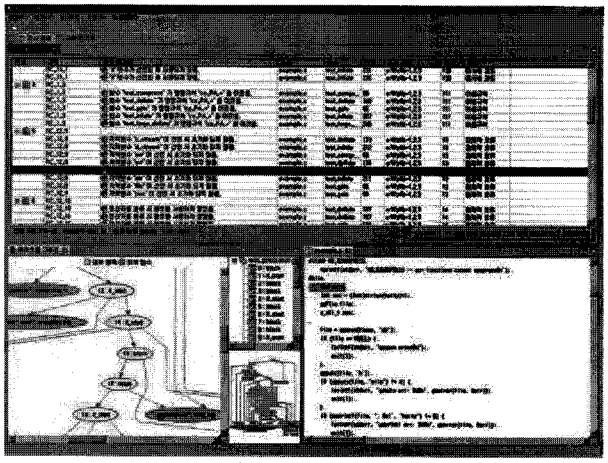

Fig. 5. Checking Results Windows.

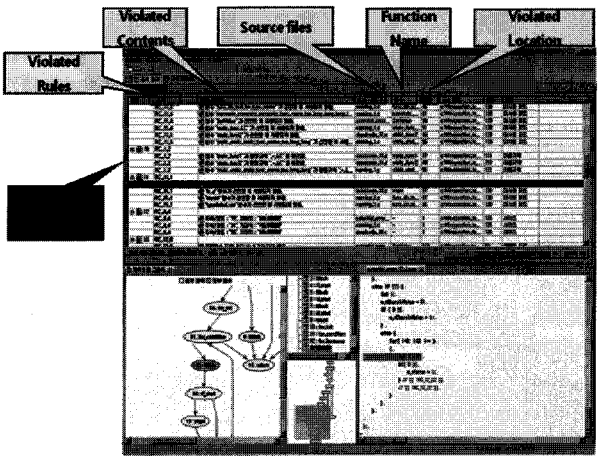

Fig. 6. Coding Rules Setting Windows.

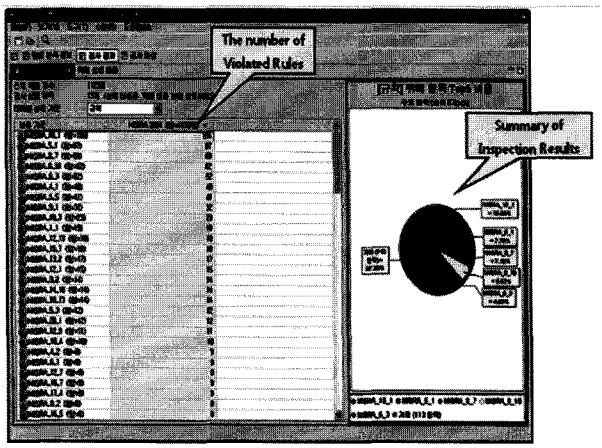

Fig. 7. The Summary Windows on Testing Results.

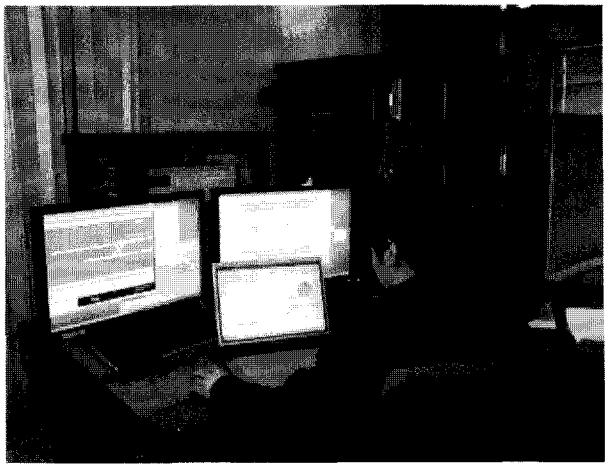

Fig. 8. Configuration of SW Inspection in Laboratory.

be checked. Due to these functions, this tool is expected to be utilized usefully even at the software debugging stage.

Fig. 6 shows the main window which performed actual testing with random source code as its target through developed inspection module. The whole violated rules are shown in the upper end of window in the figure, and if one of them was selected, it is made to point at the violated source code part like the window at the right lower end of figure. The left lower end of figure is the one showing the control flowchart of source code through analyzing those input source codes, and it was made to enable whole architectures of target source code for testing to be checked. Due to these functions, this tool is expected to be utilized usefully even at the software debugging stage. Fig. 7 shows the summary windows which performed actual testing with a source code through the developed inspection module. The number of violated rules in tested code is shown in left window and the right one shows the sum total of detection violated rules.

4. Application of Developed Inspection Tool

The developed software coding inspection tool was applied to real railway signaling system software to identify the efficiency of developed tool. Fig. 8 shows the configuration of software inspection in laboratory. Fig. 10 shows the two example results. The upper figure is the results that the declared variable was not used in the function, and the other is shows that the function was excessed the function's complexity limited value 10. The function complexity is generally considered major software metrics to indentify the safety of embedded software. If high software safety level is required,

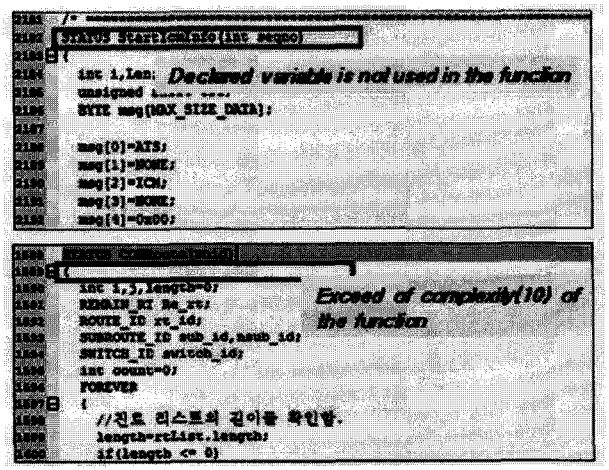

Fig. 9. Example of Test Results by Developed Inspection Tool.

the software complexity might excess 10. The lower of fig. 10 represents the function "CONRoute( )" is very complex, so that function is not suitable in aspect of software safety.

4. Conclusion

Recently, according to the development of computer technology, the dependency of railway signaling system on the computer software is being increased rapidly, and high level of safety and reliability in the railway signaling system software is required. There is not regulated software coding rules or standards for railway signaling system. Therefore the basic requirements which must be observed by railway software are being presented descriptively in related international standards.

The software safety requirements necessary for railway signaling system are being standardized by IEC 61508-3 and IEC 62279. First of all, the functional

design of developed tool for coding rules inspection testing on the software dedicated to the railway was explained, and the result of its implementation was shown concretely. In this paper the coding rules were drawn through IEC 61508-3 and IEC 62279 and developed the automatic inspection tool base deduced coding rules. Basically from the standpoint of assessor, the developed inspection may be utilized to check if the software had observed coding rules required to the target software at the software assessment stage, and it can be applied at the software development process. That is, if this automated inspection tool is used in debugging process at the software development stage, the source code suitable for the coding standard can be developed, and it is anticipated that the software having higher safety level can be secured through it.

References

[1] International Standard (1998), *Functional safety of electric/electronic/programmable electronic safety-related system*, IEC 61508

[2] International Standard (2002), *Railway Applications-software for Railway Control and Protection Systems*, IEC 62279

[3] MISRA Coding standard (2004), *MISRA-C Coding Standard*, MISRA (Motor Industry Software Reliability Association

[4] Robyn R. Lutz and Robert M. Woodhouse (1999), *Bidirectional Analysis for Certification of Safety-Critical Software*, Proceedings of 1st International Software Assurance Certification Conference, Dullers, Virginia.

[5] M. Fewstar, D. Graham (1999), *Software Testing Automation: Effective use of test execution tools*, ACM Press, Addison Wesley.

[6] J. D. Lawrence, *Software qualification in safety applications*, Reliability Engineering & System Safety, vol. 70, no. 2, pp. 167-184, 2000.