An Integrated Development Tool for a safety application using FBD language

Young-Jun Lee, Jang-Soo Lee, Dong-Young Lee
Korean Atomic Energy Research Institute
yjlee426@kaeri.re.kr; jslee@kaeri.re.kr; dylee2@kaeri.re.kr

1. Introduction

Regarding digitalizing the Nuclear Instrumentation and Control Systems, the application program responsible for the safety functions of Nuclear I&C Systems shall ensure the robustness of the safety function through development, testing, and validation roles for a life cycle process during software development. The importance of software in nuclear systems increases continuously. The integrated engineering tools to develop, test, and validate safety application programs require increasingly more complex parts among a number of components within nuclear digital I&C systems. This paper introduces the integrated engineering tool (SafeCASE-PLC) developed by our project. The SafeCASE-PLC is a kind of software engineering tool to develop, test, and validate the nuclear application program performed in an automatic controller.

2. The Structure of the integrated engineering tool

In this section, we explain the overview of the SafeCASE-PLC. The structure of the SafeCASE-PLC consists of four parts. A platform based on the Eclipse environment, an FBD-Developer used for developing the application of the PLC (Programmable Logic Controller) system, an FBD-Tester used for direct testing of the software without C language conversion and finally the FBD-Verifier which can verify and validate the software. We describe the modules briefly.

2.1 The platform

The platform for the SafeCASE-PLC tool is used based on the JAVA Eclipse Environments. The Eclipse is an open platform and supports any framework to merge different individual tools to an integrated representative single tool. It includes the user interface to connect resources between the one tool and the other tools continuously. The Eclipse is able to be used as any framework for a specific environment that is able to integrate the separated sub-systems in accordance with defining the sub-systems as several groups hierarchically. These frameworks have some hierarchic structures. Due to any frameworks only being able to be integrated with unique architectural interfaces, an application can be created only when merging the frameworks to be requested by the application. The reason for selecting the Eclipse platform to integrate several tools is because the Eclipse features cover the requirements and elements needed to integrate the tools effectively.

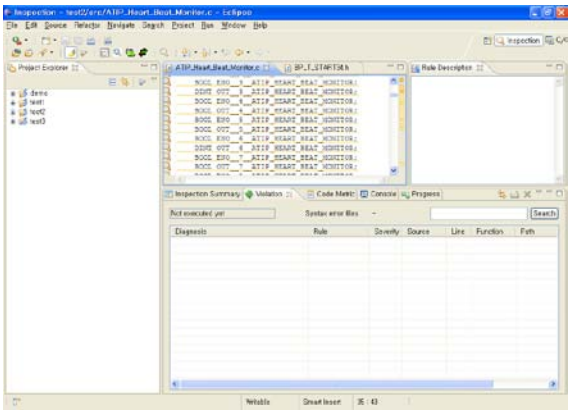

Fig.1. The perspective view of Eclipse env.

2.2 The FBD-Developer

The FBD-Developer is the software that creates a new program operated in the SafeCASE-PLC. The application software performed in Safety Grade PLC systems has to do the same work repeatedly every scan time using deterministic methods. The application software acquires some data from other equipment, calculates outputs with acquired data, and some predicted actions are performed using the calculated data. The FBD-Developer should have the robustness because of generating application programs related to the safety features.

2.3 The FBD-Tester

FBD is one of the standard PLC programming languages. FBD is widely used because of its graphical notations and usefulness in implementing applications where a high degree of data flow exists among components. FBD defines system behavior in terms of flow of signals among function blocks. A collection of function blocks is wired together in the manner of a circuit diagram. First of all, the FBD program differs from previous programs made by well-known structural languages such as C++ in regard to structure and formal type. Thus, we need to define the distinguishing features of the Unit, Module, and System used in FBD language. The definitions are suggested in the paper [2].

The FBD-Tester is implemented using definitions regarding system-specific FBD language.

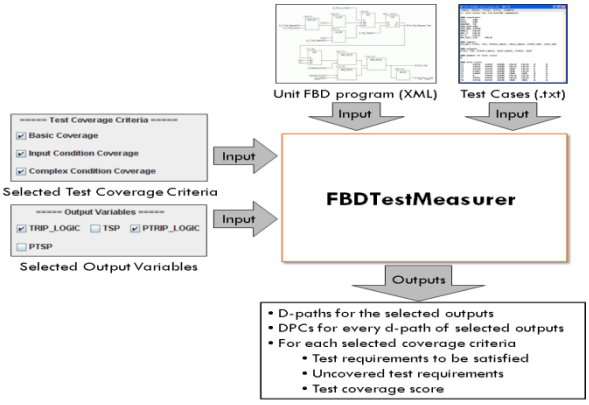

Fig.2. The structure of the FBD-Tester

2.4 The FBD-Verifier

The FBD-Verifier is a sub tool which operates the verification activity in the SafeCASE-PLC. The application developed and tested by specific sub-tools of the SafeCASE-PLC is finally verified as using the FBD-Verifier. Generally, a formal verification means checking whether design specifications satisfy the requirement specifications. If the no design specifications satisfy the requirement specifications, the verification tools make some counter-examples and can traces cases of inconsistency between the design and requirement [3].

The FBD-Verifier translates the FBD program to Verilog Program operated with the same means. Verilog is a hardware modeling language and can be used as an input of previous formal verification tools. Thus, it can be efficiently used to perform the formal verification activities. After translating the FBD programs into the Verilog programs, it performs the equivalence check and the model check using a formal verification tool like a VIS and SMV. The equivalence check of a VIS can prove the equality of the FBD programs and Verilog programs. The model checking function of an SMV can also prove whether the FBD program satisfies the important verification attribute mathematically[3].

3. Conclusions

It is essential and meaningful to newly develop an integrated engineering tool that can develop application software for a man-machine interface system used within the nuclear industry. However, this development strategy has low efficiency due to time limitation and high costs. Instead of undertaking the entire life cycle development process, it is more efficient to adopt optimized tools after integrating a commercial product that is able to develop some safety applications.

We developed the integrated engineering tool (SafeCASE-PLC). The SafeCASE-PLC is a kind of software engineering tool to develop, test, and validate the nuclear application programs performed by automatic controller. The platform for the SafeCASE-PLC tool is based on the JAVA Eclipse Environments. The Eclipse is an open platform and supports any framework to merge diverse, separate tools to make an integrated representative single tool. Existing engineering tools can translate the FBD programs into the C-code programs, but there are continuous debates about inconsistencies between the original FBD program and the translated C program. In order to solve these problems, we also researched the methods that can test directly the application programs generated by using FBDs languages. The newly generated application program can be directly tested without converting to C language, and it is also possible to test the application programs indirectly. The two testing results from the C tester and the FBD tester can be utilized for ensuring a quality feature of the translator which can convert the FBD application to C language code. The SafeCASE-PLC includes a utility that can verify some applications formally. The utility translates the FBD program into the Verilog Program. It performs the equivalence check and the model check using formal verification tools such as VIS and SMV. The equivalence check of a VIS can prove the equality of the FBD program and Verilog program. The model checking function of an SMV can also prove whether the FBD program satisfies the verification attribute mathematically.

REFERENCES

[1] IEC (International Electrotechnical Commission), International standard for programmable controllers: Programming languages: Part3 (IEC 61131-3), 1993.

[2] State-of-the-Art Report for the testing and formal verification methods for FBD program, KAERI/AR-901-2011..

[3] Junbeom Yoo, Jong-Hoon Lee, “FBDtoVerilog: A Vendor-Independent Translation from FBDs into Verilog Programs”, The Twenty-Third International Conference on Software Engineering and Knowledge Engineering (SEKE2011).

[4] Eunkyoung Jee, Seungjae Jeon, Sungdeok Cha, Kwanyong Koh, Junbeom Yoo, Geeyong Park, and Poonghyun Seong, “FBDVerifier: Interactive and Visual Analysis of Counter-example in Formal Verification of Function Block Diagram,” Journal of Research and Practice in Information Technology, Vol.42, No.3, pp.255-272, August, 2010.

[5] Eunkyoung Jee, Junbeom Yoo, and Sungdeok Cha, “Control and Data Flow Testing on Function Block Diagram” SAFECOMP 2005, LNCS 3688, pp. 67-80, 2005.

[6] A.Mader, “A Classification of PLC Models and Applications”, In Proc. WODES 2000: 5th Workshop on Discrete Event Systems, August 21-23, Gent, Belgium, 2000.