New Proposal on the Development of Machine Protection Functions for ITER Diagnostics Control*)

Tsuyoshi YAMAMOTO[1,2)], Eiichi YATSUKA[1)], Takaki HATAE[1)], Kazuya OTA[2)], Yasunori HASHIMOTO[2)], Kitaru NAKAMURA[2)], Tatsuo SUGIE[3)], Masaki TAKEUCHI[1)], Sin-iti KITAZAWA[1)], Hiroaki OGAWA[1)], Yasunori KAWANO[1)] and Kiyoshi ITAMI[1)]

[1)]*Japan Atomic Energy Agency, Naka-shi, Ibaraki 311-0193, Japan*[**)]
[2)]*Japan Expert Clone Corporation, Shinjuku-ku, Tokyo 160-0022, Japan*
[3)]*Nippon Advanced Technology Co. Ltd., Tokai-mura, Ibaraki 319-1112, Japan*

(Received 30 November 2015 / Accepted 25 August 2016)

There is a need to develop ITER instrumentation and control (I&C) systems with high reliabilities. Interlock systems that activate machine protection functions are implemented on robust wired-logic systems such as programmable logic controllers (PLCs). We herein propose a software tool that generates program code templates for the control systems using PLC logic. This tool decreases careless mistakes by developers and increases reliability of the program codes. A large-scale engineering database has been implemented in the ITER project. To derive useful information from this database, we propose adding semantic data to it using the Resource Description Framework format. In our novel proposal for the ITER diagnostic control system, a guide words generator that analyzes the engineering data by inference is applied to the hazard and operability study. We validated the methods proposed in this paper by applying them to the preliminary design for the I&C system of the ITER edge Thomson scattering system.

Keywords: ITER, machine protection, inference, program code template, HAZOP

DOI: 10.1585/pfr.11.2405114

1. Introduction

ITER is an international experimental nuclear fusion device with challenging technological requirements [1]. Machine protection is an important function for maintaining high reliability and safe operation in large experimental facilities such as ITER. Therefore, the ITER instrumentation and control (I&C) system requires high reliability. The ITER I&C system comprises the control system, interlock system, and safety system [2]. The interlock and safety systems are implemented on wired-logic systems and are responsible for machine protection and safety, respectively. The control system, which is a software-based system, controls and monitors the plant. In this paper, we focus on consistency between the designs of the control and interlock systems.

The interlock system, which performs protective actions for the ITER plant systems, is implemented with robust controllers such as programmable logic controllers (PLCs). However, the control system is implemented with industrial computers. To ensure interoperability between the interlock and control systems, we propose a method to convert an interlock logic described by the PLC code into another programming language for the control system. As an example implementation, we applied this method to the design of the ITER edge Thomson scattering (ETS) system, which is being developed in Japan. ETS is a diagnostic system for measuring electron densities and temperature profiles of ITER plasmas using a high-energy laser [3]. The ETS system must protect the vacuum window and other devices from laser hazards. Therefore, our method can be validated through ETS development.

A large-scale engineering database has been implemented in the ITER project [4, 5]. This database has the potential to produce useful information for the design of the I&C system [6, 7]. We developed a new method to add semantics data to the engineering database via information technology. We validated this method by applying it to the hazard and operability (HAZOP) study [8], a risk analysis approach being used in the development of the ETS system.

In this paper, we propose a source code generation tool and the application of the engineering database to the development of a highly reliable I&C system for the ITER.

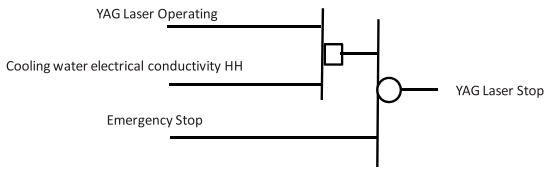
2. Automatic Generation of the Control System Software

Automatic code generation has become a major development approach in recent years. This approach decreases

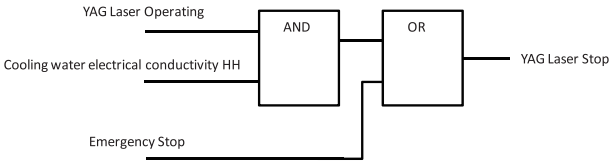
author's e-mail: yamamoto.tsuyoshi@qst.go.jp
*) This article is based on the presentation at the 25th International Toki Conference (ITC25).
**) Current affiliation: National Institutes for Quantum and Radiological Science and Technology

careless mistakes by developers and increases reusability and reliability of software. We applied automatic code generation to the development of the ITER I&C diagnostic system. We developed a software tool to generate a template of the program codes for the control system.

The interlock system should be reliable. Interlock systems have previously been implemented as hardwired circuits such as relay circuits. With advances in microcomputer technology, such hardwired circuits have been replaced with programmable logic controllers (PLCs). Machine protection logic can be described using an interlock block diagram (IBD) (Fig. 1 (A)). A PLC can be programmed using six different programming languages defined in international standard IEC 61131-3 [9]. In the case of functional block diagram (FBD), one of the six languages, machine protection logic can be expressed as



(A) An example of interlock block diagram.



(B) An example of functional block diagram.

Fig. 1 Examples of an interlock block diagram (A) and a functional block diagram (B). Both diagrams have similar expressions but differ in their symbols.

the circuit diagram in Fig. 1 (B). We therefore selected FBD for the PLC programming language. FBD can be described in the extensible markup language (XML) format [10]. PLCopen, an international organization active in industrial control, has defined an XML format for FBD. We used the XML format for FBD to generate the software codes from FBD because XML is easy to handle using various programming languages and because we have already demonstrated the use of the XML format for PLC development [11].

Figure 2 shows the procedure used to generate the program code template. Interlock logic (Fig. 2 (A)) designed by a process engineer is implemented in a PLC using FBD (Fig. 2 (B)). In most cases, a program code in the control system is developed by a software engineer working independently of the process engineer. The program code template generation tool converts the FBD code to the program code. In the ITER project, the experimental physics and industrial control system (EPICS) [12] is recommended as a software infrastructure for the development of control systems. Input or output signals used for the control systems are managed as a "record" of the database in the EPICS environment. Input or output for controlling a device can be used to read a value from or write a value to the record, respectively. In addition, complex control logic can be implemented to link several records (Fig. 2 (C)). The program code template generation tool generates the definition of the records and links between the records automatically.

We developed the program code template generation tool using the Python language. The data described in the XML format for FBD are seperated into pieces that can be processed by the computer program using the subroutine called "XML parser," which is integrated into the Python language. The program code generation tool analyzes the XML file and stores relationship data between function blocks to internal memory. The tool generates the

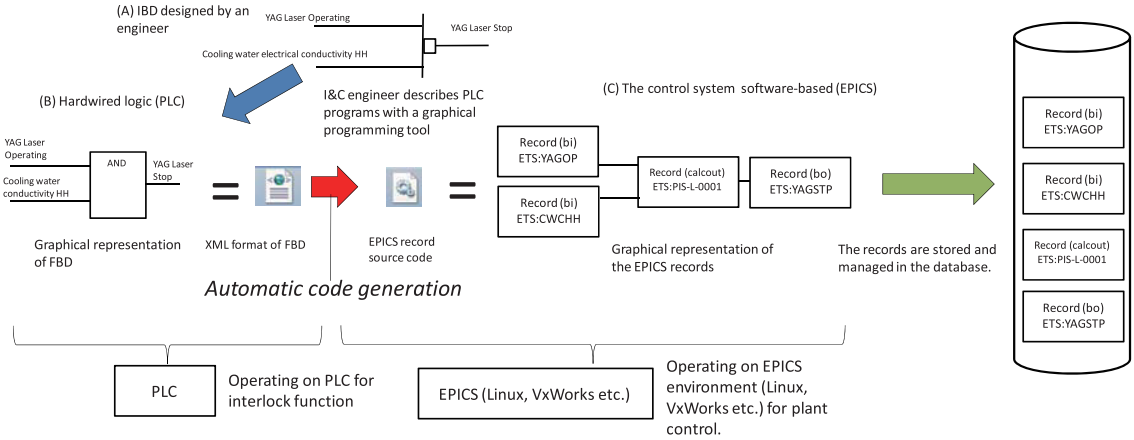


Fig. 2 (A) IBD is used in the design phase. (B) Control system engineer describes PLC logic based on IBD. PLC logic formatted in XML format can be converted into other software languages, including EPICS source code. (C) EPICS codes are executed in a major industrial computer. The ITER Organization recommends EPICS.

EPICS source codes, which describe the definition of the records and the links of the records, on the basis of relationship data analyzed by the XML parser. The EPICS source codes generated by the tool can be executed in the EPICS environment on various operating systems, including Linux and VxWorks.

3. Proposed Application of the Information Technologies for Design of Machine Protection

3.1 Engineering data analysis

In the ITER project, design activities over several years produce a large volume of engineering data. The engineering data are stored in the engineering database. Currently, the engineering data are used for managing components [5]. We believe that the engineering database has the potential to generate useful design information. To enable efficient use of the engineering database, meta-data, which explain the engineering data, must be added to the database. We previously proposed that meta-data for the engineering database be described using the resource description framework (RDF) [13]. RDF is a simple set of three data (i.e., subject, predicate, and object) based on predicate logic. Most database management systems can store RDF data.

A graphical representation of the RDF is shown in Fig. 3. A set of subject, predicate, and object is called a Triple. The Triples shown in Fig. 3 indicate that Pipe number 0001 transmits cooling water and that cooling water is electrically conducting. Notably, the vocabulary, known as *ontology* in information science, should be defined in advance.

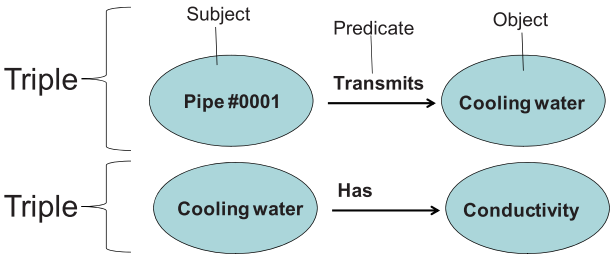


Fig. 3 An example of a graphical representation of RDF. This example consists of two expressions. One means component "Pipe #0001" is used for transferring cooling water. The other means cooling water has electrical conductivity as a property.

3.2 Application to ETS and validation of the results

We defined simple predicates for diagnostic systems to validate our concept. The predicates were applied to preliminary design the laser system for ETS [14]. A schematic of the process flow diagram of the laser system is shown in Fig. 4. Two YAG laser systems are cooled by chilled water. The beams from the YAG lasers are combined and injected into a plasma. The beam is terminated by the beam dump installed in the vacuum vessel [3]. A ruby laser is also connected to the beam combiner for calibration.

Functional modeling of a plant based on ontological engineering was studied by Kitamura *et al.* [6, 7]. We applied three points of view—physical relationship, function, and attribute—to describe the RDF for the process flow diagram according to the previous studies. The predicate "connect" is a typical predicate used to represent a physical relationship. The function expresses the behavior of a component. Predicates of the function used in the RDF for the process flow diagram are listed in Table 1. Temperature, flow rate, and electrical conductivity are examples of attributes. A portion of the RDF for the process diagram is shown in Fig. 5.

The logical conclusion derived from one or more statements is called an *inference*. Figure 6 shows a syllogism, which is a logical argument. The first term, "All electrical conductivity of cooling water must be monitored," is

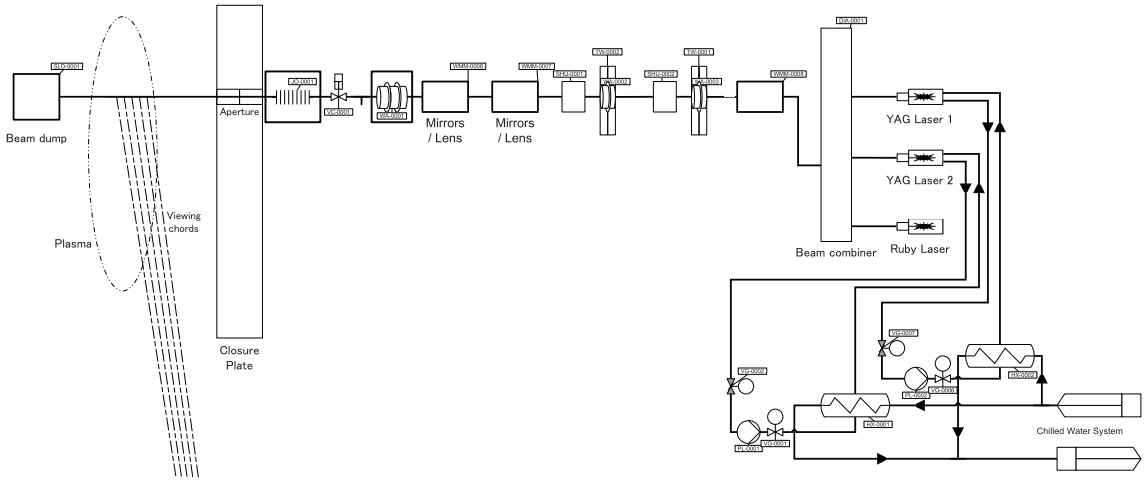


Fig. 4 Preliminary process diagram of a laser system for the ITER ETS.

Table 1 Predicates of the function used in RDF for the process diagram.

Transmits
Supplies
Provides
Retrieves
Generates
Injects
Dumps

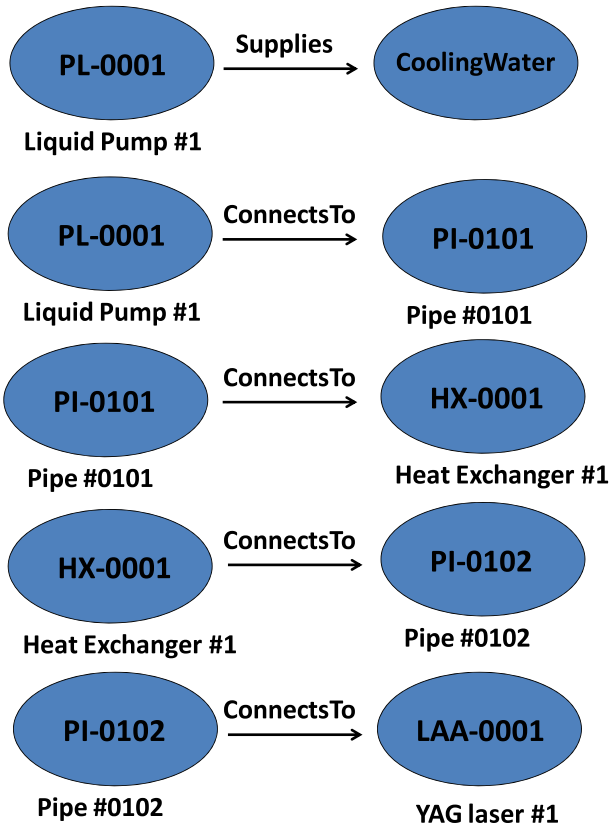


Fig. 5 A part of the RDF for the process diagram of a preliminary ITER ETS laser system design. This figure shows liquid flow from the chilled water pump to the YAG laser.

a major premise. The second term, "Some pipes transmit cooling water" is a minor premise. The logical conclusion "Cooling water must be monitored at the pipe" is derived from the major and minor premises.

We applied the inference rules to the RDF for the process flow diagram of the laser system for ETS. The software tool cwm [15] was used for this inference. Table 2 shows inference rules and results. The inference rules included "If conductivity of cooling water exceeds the limit, related components must be stopped." The cwm tool generated the inference results listed in Table 2. The results also included "Controller triggers digital outputs addressed C3S0 and C3S1" (the digital outputs are signals to stop the device). We note that the ETS RDF data are just a set of Triples. We concluded that the inference derives meaningful results.

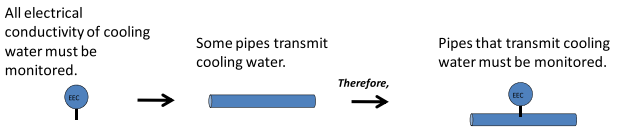


Fig. 6 An example of an inference. This example shows a simple syllogism. General (first) and specific (second) statements are mentioned; a conclusion is then derived.

3.3 Risk analysis by the HAZOP

HAZOP is a method used to identify hazards and risks in the system [8]. In HAZOP, guide words, which indicate potential risks, must be determined in advance according to analysis of the precedents. A hazard analyst investigates the likelihood of the risks based on guide words assigned to the process.

As discussed in Section 3.1, piping, instrumentation, and other plant characteristics can be described in the RDF files. We proposed a guide-word generator using inference rules. Guide words are associated with the characteristics of the process (e.g., flow, pressure, and temperature). Therefore, an inference rule that describes the relationship between process types and guide words is needed to generate guide words for the system.

Our generating method has the following advantages. 1) Because guide words are generated in accordance with predefined rules, they are generated systematically, reducing the number of missed guide words. 2) Knowledge of experts can be reused because the knowledge is described in the inference rules. 3) The quality of HAZOP assessment using generated guide words is easily maintained.

We also applied our method to the RDF data for the ETS laser system mentioned in Section 3.1 and compared the results with our hazard analysis performed in 2014. The results of this comparison are reported in Table 3. The generated guide words were equivalent to our hazard analysis. The results of this comparison indicate that our method produces adequate guide words. We plan to apply this method to risk analysis for other diagnostic systems procured by Japan.

4. Summary

We proposed a tool for the control system to generate program code templates written by the EPICS source basing on the FBD of the interlock system. We designed

Table 2 Rules and results of inferences.

Description of inference	Inference rules	Inference results	Description of results
If conductivity of cooling water exceeds the limit, - Components, which are connected to cooling water line must be stopped; - To stop the component, digital output assigned to the component shall be triggered.	@forAll:x,:y,:z . {:x :ConnectsTo:y .:y : ConnectsTo :z. } => {:x :ConnectsTo :z .} . @forAll:l,:m,:n . {:l :ConnectsTo :m .:m :Transmits:CoolingWater.:l :IsStoppedBy :n .} => {:Controller :Triggers :n . } .	<ets://Controller> <ets://Triggers> <ets://PSStop-C3S0>, <ets://PSStop-C3S1> .	Controller triggers digital output C3S0 and C3S1.

Table 3 Comparison of hazards between analyses performed in 2014 and generated guide words.

No.	Hazard analyzed in 2014	Generated guide word
1	Conductivity of cooling water exceeds limits	High conductivity at cooling water pipes
2	Temperature of cooling water exceeds limits	High/low temperature at cooling water pipes
3	Water flow of cooling water exceeds limits	High/low flow at cooling water pipes
4	Failure of chiller	Heat exchanger failure
5	Laser beam stray	YAG laser 1 stray beam; YAG laser 2 stray beam; Ruby laser stray beam; Beam combiner stray beam
6	Deterioration of beam profile	YAG laser 1 bad profile; YAG laser 2 bad profile; Ruby laser bad profile; Beam combiner bad profile
7	Detection of 1st window scattered light	Window assembly failure
8	Detection of water leak	Water leak at pipes

RDF meta-data to automatically produce useful information from the engineering database for design of the machine protection function. We also proposed a new HAZOP guide-words generator to generate guide words from the engineering database. These proposals will be applied to the development of ITER I&C diagnostic systems.

Acknowledgments

The authors would like to thank Dr. S. Simrock and staff of the ITER Organization for fruitful discussions about ITER control system and design methodology.

Disclaimer

The views and opinions expressed herein do not necessarily reflect those of the ITER Organization.

[1] S. Kitazawa *et al.*, Fusion Eng. Des. **89**, 88 (2014).
[2] A. Wallander *et al.*, Fusion Eng. Des. **85**, 529 (2010).
[3] E. Yatsuka *et al.*, J. Instrumentation **8**, C12001 (2013).
[4] L. Abadie *et al.*, Fusion Eng. Des. **87**, 2213 (2012).
[5] J. Sonara *et al.*, Fusion Eng. Des. **96-97**, 957 (2015).
[6] Y. Kitamura *et al.*, Trans. Jpn. Soc. Artificial Intelligence **17**, 61 (2002).
[7] Y. Kitamura *et al.*, Trans. Jpn. Soc. Artificial Intelligence **17**, 73 (2002).
[8] T. Kletz, *Hazop & Hazan*: *Identifying and Assessing Process Industry Hazards*, 4th edition (Institute of Chemical Engineers, Rugby UK, 1999).
[9] IEC 61131-3: 2013. Programmable controllers–Part 3: Programming languages.
[10] Extensible Markup Language (XML), https://www.w3.org/XML/
[11] T. Yamamoto *et al.*, Fusion Eng. Des. **96-97**, 1012 (2015).
[12] Experimental Physics and Industrial Control System, http://www.aps.anl.gov/epics/
[13] Resource Description Framework (RDF), http://www.w3.org/RDF/
[14] T. Hatae *et al.*, Rev. Sci. Instrum. **83**, 10E344 (2012).
[15] cwm, http://www.w3.org/2000/10/swap/doc/cwm.html