Development of process control software using software engineering techniques

Gregor Kandare

Jozef Stefan Institute, Jamova 39, 1000 Ljubljana, Slovenia

E-mail: gregor.kandare@ijs.si

Keywords: Process control software, Software Engineering, Automatic Code Generation, Programmable Logic Controllers

Abstract: Modern software systems are becoming increasingly complex and for that reason extremely difficult to develop and maintain. The size of the development team usually depends on the size and complexity of the software system developed in the project. Good coordination and organization of the team is necessary.

One of the main problems of software engineering is that it is a relatively new technical discipline, compared to well-established disciplines such as mechanical and electrical engineering. Plenty of software development projects are still performed in an ad-hoc manner and many of them yield bad and unreliable products or even fail to deliver a working software system. Another issue that makes software engineering techniques difficult to standardize and foment their use in a wider range of software development projects is the extremely rapid development of computer hardware on one hand, and development of programming languages and software development environments on the other hand. Nevertheless, in the last three decades, several software engineering modeling languages and computerized tools have been developed and successfully used. In the late sixties and throughout the seventies, procedural programming languages were dominant and thus the modeling languages were also designed to reflect the procedural point of view. In the mid-eighties object-oriented languages prevailed, which led to a development of a broad spectrum of object-oriented modeling languages and tools. Most of those modeling languages converged to the Unified Modeling Language (UML) [1], which is the one mostly used today.

In the domain of process control, the programmable logic controllers (PLC) are the devices used in almost every industrial control system. Their most important function is to supervise and control the discrete events that occur in the controlled system. The PLC are often used in very critical systems such as nuclear power plants etc., where a failure can cause significant material damage or can even threaten lives. Consequently, the demands for their reliability are very high. For this reason, the development of PLC hardware and programming languages (defined by the IEC 61131-3 standard) has been more conservative than the progress of common-use computers and programming languages.

Software engineering methods have not been widely used in the development of PLC software, probably due to the fact that the software developers were mostly electrical engineers. In the recent years, however, several approaches using software modeling techniques in PLC software development have been proposed in the literature.

In the article, a survey of the use of software engineering methods in process control software development is made. Furthermore, an example of a domain-specific (procedural control of continuous processes) software engineering methodology is presented. The

methodology has been successfully employed in several laboratory and industrial projects [2]. The formal description of syntax and semantics of the Procgraph modeling language, used by the methodology, are made. The formal model permits to build the rules for correctness and consistency checking of the models made with the language. Furthermore, a formal description of the language allows us to define the rules of model-to-code mapping. These rules are used as guidelines by the programmers that convert models from the analysis/design phase into IEC 61131-3 source code [3]. Furthermore, the mapping rules were implemented in a domain specific software modeling tool that can perform automatic source code generation. The use of automatic code generation significantly accelerates the software development process and reduces the number of coding errors. Additionally, automatically generated code is constructed from previously defined patterns and as such easier to maintain.

References

[1] Booch, G., J. Rumbaugh, I. Jacobson: *The Unified Modeling Language User Guide*, Addison Wesley, Boston, 1999.

[2] Kandare, G., G. Godena in S. Strmčnik: A New Approach to PLC Software Design, *ISA Transactions*, Vol. 42, No.2, str. 279-288, April 2003.

[3] Lewis, R.W.: *Programming industrial control systems using IEC 1131-3*, The Institute of Electrical Engineers, London, UK, 1998.