Applying SDL Specifications and Tools to the Verification of Procedures*

Wenhui Zhang

Institute for Energy Technology, P.O.Box 173, N-1751 Halden, Norway
Wenhui.Zhang@hrp.no

Abstract. Verification of operating procedures (that is, specifications of manual control actions) by model checking has been discussed in [16]. The modelling language Promela and the model checker Spin were used in that report. In order to be able to apply model checking in a wider scope, modelling languages with graphical interface and verification tools used in industrial context are preferable (for example, to facilitate collaboration with process experts). In this paper, we discuss how to use SDL to model systems consisting of operating procedures and the controlled processes. Verification of procedures against correctness specifications is done by using the tool SDT. We conclude the paper with a short discussion of the integration of formal verification with the procedure design process.

1 Introduction

Operating procedures are documents telling operators what to do in various situations. They may be used in normal operation situations or for bringing a system from an unsafe state to a safe state. The main contents of a procedure (an operating procedure is sometimes referred to as a *procedure* for short, and for clarity, a procedure used as a subroutine in SDL is referred to as an *SDL-procedure*) are sequences of instructions. An operator could for instance be instructed to read some relevant information, to perform a sequence of actions, to check a system state and then go to a suitable instruction according to the system state, and to wait for a condition. Instructions may be grouped into steps and sub-steps.

Operating procedures are widely used in process industries including the nuclear power industry. The correctness of such procedures is of great importance to the safe operation of nuclear power plants. Traditional methods for the verification and validation of procedures are reviews and simulation. By means of reviews, the designer or members of a review team walk through the document to check errors, violation of standards or other problems. As the number of execution paths of a procedure could be large, this technique may not be practical

* The research described in this paper was carried out at the OECD Halden Reactor Project at the Institute for Energy Technology, Norway. The author thanks G. Dahll, T. Sivertsen and K. Stølen for their helpful suggestions and comments. The author also thanks anonymous referees for their constructive criticisms that helped in improving this paper.

R. Reed and J. Reed (Eds.): SDL 2001, LNCS 2078, pp. 421–437, 2001.

for large procedures. Simulation can be compared with software testing with a limited number of executions (in the environment provided by a simulator). As simulations are time-consuming and the number of simulations is limited, this approach may not be able to provide the desirable level of confidence on the correctness of procedures.

In the recent years, procedures have been computerised in order to enhance safety and reduce operator stress [13,7,10]. Computerised procedures provide a better opportunity for formal verification as they are written in formal or semi-formal languages with better structure and formality. Earlier work on using formal methods for verification of operating procedures includes investigating techniques based on algebraic specifications and related theorem proving tools [14,15] and on techniques based on model checking [16]. In the work reported in [16], the modelling language Promela and the model checking tool Spin [8,9] were used. In this paper, we present an approach to modelling and verification of operating procedures by using SDL.

Motivation: The main motivation of investigating how to use SDL for the verification of operating procedures is that SDL is a modelling language with graphical interface and verification tools used in industrial context. For the purpose of verification of procedures in a wider scope, a collaboration between formal methods experts and process experts (with knowledge on how procedures and the controlled systems work) is necessary, and an intuitive graphical syntax instead of a textual mathematical syntax seems to be crucial.

SDL specifications: SDL [2,4] is a specification and description language standardised as ITU (International Telecommunication Union) recommendation Z.100. The basis for description of behaviour is communicating extended state machines that are represented by processes. Communication is represented by signals and can take place between processes, or between processes and the environments of the system model. Some aspects of communication between processes are closely related to the description of system structure. An extended state machine consists of a number of states and a number of transitions connecting the states. One of the states is designated the initial state.

In SDL, each feature has a graphical syntax and a textual syntax. The sequential components are called processes. Processes can be composed into blocks, blocks can be composed into larger blocks and form systems. Systems may communicate with external environments. SDL supports object-oriented notions such as classes, objects encapsulation and inheritance.

A tool that can be used to create SDL descriptions and to verify properties of such descriptions is SDT [2,3]. There are different ways for validating a system description. Properties of a description can be specified by using user-defined rules, assertions, message sequence charts and observes processes, and verified accordingly. An example of user-defined rules is

Exists P:Proc | (P->var=1)

which is true for all system states where there exists a process P of type "Proc" with a variable "var" that is equal to 1 (comparing with CTL [5], one may