Automatic Code Generation from SDL to a Functional Programming Language for Safety Critical Systems

László Gombos

Safety critical system is a system where human safety is dependent upon the correct operation of the system. Although safety critical systems have been in use for many years, the development of safety critical software is still a relatively new and immature subject. Our approach is to use functional languages for high reliable softwares, which allows higher abstraction level in the implementation and allows greatly improved modularisation. Functional programming languages provide two new kinds of modularisation technique - higher order functions and lazy evaluation.

We have choosen Clean as a target functional language. It is a strongly typed language with support for unique types. Unique arguments (such as the environment of the system) allows destructive updates preserving the referential transparency and makes it possible to improve the efficiency of the program execution.

Clean provides an extensible library (Object I/O library) to create interactive applications by composition of concurent, interleaved communication processes. Besides, the library has support for abstract devices such as timer and receiver objects.

The Specification and Description Language (SDL) is a standardized Formal Description Technique (FDT) for the specification of discrete reactive systems. SDL models the system as a number of concurent, communicating process instances interchanging signal instances with each other and with the environment of the system. Each process instance is an extended state finite state machine. The communication is based on asynchronous passing of parametral signal instances from one sender to one receiver. Time-dependencies can be modelled by means of timer instances and with the help of the Time and Duration predefined data types.

SDL is widely used for specifying and implementing huge and complicated telecommunication systems and protocols. The existing code generation tools, however, generate mostly C code, which is poorly suitable for safety critical systems.

In our project, we define some auxiliary Clean functions and semantically map SDL systems to Clean programs. This paper describes a code generator mechanism from SDL/PR to Clean. SDL objects, such as channels, process types and signals are defined on a high level of abstraction using algebraic datatypes. Procedure, process type and signal specialisation is discussed in details, and a mechanism is shown how to model unreliable system components with the help of spontaneous transitions.

References

[1] CCITT, *Recomendation Z.100, Specification and Description Language SDL*, 1993.

[2] Peter Acten: *A Tutorial to the Clean Object I/O Library - Version 1.2*, University of Nijmegen, 2000.

[3] Zoltán Horváth, Peter Achten, Tamás Kozsik, Rinus Plasmeijer: *Verification of the Temporal Properties of Dynamic Clean Processes*, In: Koopman, P., Clack. C. eds. Proceedings of the 11th International workshop on the Implementation of Functional Languages, IFL'99, Lochem, The Netherlands, September 7-10, 1999, pp. 203-218.