A General Approach for the Specification of Real-Time Systems with SDL*

Ralf Münzenberger, Frank Slomka, Matthias Dörfel, and Richard Hofmann

University of Erlangen-Nuremberg
Department of Computer Networks and Communication Systems
Martensstr. 3, 91058 Erlangen, Germany
{rfmuenze, slomka, doerfel, rhofmann}@informatik.uni-erlangen.de

Abstract. In contrast to protocols of the network or transport layer the protocols for medium access have to consider the timing behavior of the communication medium. Although SDL is a widely used language for the specification of communication systems, in most cases time critical parts are not considered. In this paper, a design pattern is discussed that allows the specification of time critical functionality sucg as multiplexers or Quality-of-Service (QoS) schedulers. In many applications such services are running in a synchronous manner with the communication medium. A notation for timing aspects is needed for the specification of this behavior which itself is only possible in a sensible way with a formal model of time. Clocks are used to define the term real-time in a formal way, leading to the specification of timing constraints, for example sending data packets in deterministic time intervals within a communication system. In a case study from the mobile communication area, the design pattern was used to specify the MAC-Layer including time critical parts.

1 Introduction

The specification of large, distributed communication systems is a well known application area of the specification and description language (SDL). SDL is mainly designed for the development of communication protocols. Many language constructs support the requirements needed by protocol development, such as non-deterministic choices, sending and receiving signals asynchronously and the specification of protocol timers.

Nearly all protocols specified with SDL are at layer 3 of the OSI reference model or above. Layer 2 functionality such as medium access is often implemented in a mixed hardware/ software architecture to achieve the performance

* This work has been funded by the Deutsche Forschungsgemeinschaft (DFG) under grant HE 1408/4-3 as a part of the program Rapid Prototyping of Embedded Control Systems with Real-Time Constraints. We thank Lennard Kerber for his critical comments and ideas; Günther Peitz and Georg Sandhaus of Tenovis for their support during the development of the DECT MAC-Layer and providing the Tenovis DECT transceiver modules; additionally Kai Lampka for his work on the specification of the DECT MAC-Layer.

R. Reed and J. Reed (Eds.): SDL 2001, LNCS 2078, pp. 203–222, 2001.

goals. The separate consideration of hardware and software has led to the use of different design methodologies for the two parts. Especially, the separated consideration often leads to errors during the integration phase. To improve the design process, an integrated specification language is needed where all aspects can be formulated in an adequate way. Because of the wide usage of SDL at the system level, it is desirable to use SDL also for the hardware/software parts of the system. To reach this goal, SDL needs a new time model which has to enable the definition of clocks, the specification of timing constraints and the specification of synchronous behavior.

The current time model of SDL allows the use of timers as a way of defining a timing behavior. Unfortunately the value given to a timer has no unit. This leads to additional informal specifications given in comments or in separate documents to determine which unit and resolution to use for the respective timers. For formal verification and integrated hardware/ software code generation it is important that this information is an integral part of the system specification. Even the reception of signals sent by expired timers is asynchronous because the consumption of the timer signal has no relation to its expiration time. This is in contradiction to the former required deterministic and synchronous behavior of real-time systems.

The usage of the construct *now* as an alternative to timers is not sufficient to implement deterministic real-time behavior because *now* only accesses the system clock variable which has an implementation dependent behavior. The impossibility to specify the behavior of the system clock in a formal and code generator independent way clearly rules it out. The only valid assumption about *now* is the relation $now_n <= now_{n+1}$ in two consecutive statements in the same transition. Neither the size of a time step nor the linearity of the clock is defined by this.

Without a formal model of time, it is not possible to specify all aspects of real-time systems non-ambiguously. Currently, only the functional behavior can be specified, or, in other words, only a system. Due to the aforementioned reasons a formal model of clocks and timing constraints was developed. The timing constraints are subdivided into two different aspects: First, a timing requirement describes a requirement to the system that has to be validated only during the design and the test phase of the system. Second, a timing condition describes a requirement which has to be augmented repeatedly during normal execution of the system and which triggers specified actions depending on validity.

Based on this insight, a design pattern was developed to specify real-time systems. It enables the designer to specify synchronous reaction to medium access components, deterministic real-time behavior and layer 2 functionality as in packet multiplexers or schedulers. It can also be used to build a complex system, such as an IP based switch, by connecting different instances of the same pattern. This paper focusses on the specification of real-time systems with SDL. In [15] strategies for hardware/software partitioning and implementation of such systems are shown.