

SDL based protocol engineering and visualization for education: ISDN Q.931 case study

Wolfgang Kellerer¹, Achim Autenrieth, Andreas Iselt
Lehrstuhl für Kommunikationsnetze, Technische Universität München
Arcisstr. 21, 80290 München
{kellerer, autenrieth, iselt}@ei.tum.de

Keywords

SDL, MSC, educational case study, specification, simulation, visualization, ISDN Q.931 protocol

Abstract

In this paper an educational case study is presented, that shows the combined use of SDL for system design and a graphical environment for simulation visualization. This case study is embedded in an SDL laboratory course, that has been developed at the Institute for Communication Networks of the Technical University of Munich. After an brief overview over the laboratory course, the paper presents the case study, which actually is the final and largest exercise of the course. The example is based on the SDL-description of the ISDN signaling protocol Q.931. The well-known functionality of the protocol makes it a good example for the use of SDL in protocol design. The case study also deals with the integration of SDL-defined systems in an external environment, in this case a Tcl/Tk-based graphical user interface. The paper concludes with a report on the experiences gained from the laboratory course and the case study.

1 Introduction

SDL (Specification and Description Language [Z.100]) is recommended by the ITU-T for the specification and description of telecommunication systems and is especially well suited for the design of communication protocols. At the Institute for Communication Networks of the Technical University of Munich SDL is playing a major role for the engineering of large telecommunication systems. Experience and expertise in the use of SDL in various fields of telecommunications has been gained during the last years. For several recent projects SDL has been chosen as a description technique. This has especially been encouraged by the increasing availability of SDL-based CASE tools.

Within several research projects at the authors' institute, SDL specifications have been used as a basis for prototyping as well as for the simulation and evaluation of communication systems. In some latest research projects SDL played a major role in the evaluation and

¹ This work was partially supported by the Bayerische Forschungsstiftung.

simulation of distributed network protocols [IsAu97]. Prototyping has been experienced in two respects. In one approach generated C-code has been implemented on microcontroller target systems [KIR96]. In another approach an SDL simulator has been integrated into a real-time prototyping platform for ISDN-DECT-PBXes [VK+98].

The growing importance of SDL in systems engineering triggered the introduction of a SDL laboratory course. This offers our students the possibility to obtain practical experience with a modern, formal system engineering technique. After the course, interested students may apply their knowledge and get further experience in diploma theses where SDL is applied to problems in many research subjects of the institute. The course is currently open to 20 students each half-year term. The language of the course is german, but for future international study programs at our university an english version of the course material is currently prepared.

In this paper first the educational context of the SDL course is presented and an overview over the course is given. Then, one exercise is described in detail. It is a practical example composed of the specification, simulation and visualization of the Q.931 ISDN signaling protocol. A report on the gained experiences is given in the final section.

2 SDL laboratory course

The Institute for Communication Networks at the Technical University of Munich offers a laboratory course to bring the benefits of SDL usage in systems and protocol engineering to the students.

The course is structured in an introductory lecture, which is offered during one afternoon at the beginning of every semester and about 6 additional lessons with practical exercises. Since SDL is not a major topic in mandatory courses so far, the students only have a vague knowledge about the characteristics of SDL. Thus the lecture at the beginning of the course gives in 2 hours a short overview over SDL, its properties and the basic constructs. Furthermore, these topics are also covered in an additional tutorial booklet, that has been prepared for this course [KeAu97]. During the 6 practical lessons, the students work autonomously on several exercises, which teach the language SDL as well as the use of the design tool. All the practical work is done using SDT (SDL Design Tool, TELELOGIC, Sweden) [SDTa], which is a widely used commercial CASE tool for system development using SDL. The exercises are done by the students on their own, guided by the tutorial booklet. Student tutors, which have already absolved the course, answer questions via email and post common problems on the course's intranet-web-pages. For further questions there are special tutorial hours to help students with specific problems on site.

2.1 Introductory lecture

The introductory class includes an overview over formal description techniques and benefits of SDL usage within system development. The SDL concepts concerning the behavior, hierarchical structuring, the communication using signals and the data concept are explained. The basic SDL-symbols are introduced, including an outlook on object-orientation in SDL-92 ([OFM+94]). An overview over Message Sequence Charts (MSC [Z.120]) concludes the theoretic explanations. The lecture is completed by a practical demonstration of an SDL example on the workstations.

2.2 Practical exercises

The educational objectives for the practical part include:

- Modeling behavior: SDL process and state transition
- Modeling structure: SDL system, blocks and channels
- Communication mechanism: signal consumption and timer mechanism
- Parallel processes and process instantiation
- Variable declaration and data processing in tasks using abstract data types
- Object orientation in SDL-92: block types and gates
- Message Sequence Charts

These objectives are met step by step by examples from protocol specification. For the specification, design, simulation and validation the CASE tool SDT is used. First steps with the tool are guided by SDT's "Getting Started" manual [SDTb]. To teach the SDL basics, we have chosen a simple stop-and-wait protocol as a practical example. The exercise deals with a scenario for data transmission over a connectionless channel. The participants have to specify the processes for an acknowledged data transmission. In the next exercise the protocol has to be enhanced by a timer mechanism to prevent dead-locks by signal loss. The unreliable transport media is represented by a separate block. Thus a transmission failure can be simulated within the simulation. The last step is to move to connection oriented data transmission. That means, that a connection establishment procedure has to be built into the system. Thereby, dynamic process creation and management is shown. For these exercises MSCs are given to the students as requirements specifications for the system behavior.

For teaching object oriented mechanisms we have chosen the token ring recovery protocol [802.5]. Given the description of the token ring recovery mechanism, the students have to specify a simulation system to show the correct functionality of the protocol. The system consists of four equal stations in a ring LAN. Each station is an instance of one block type, which contains the specification of the recovery protocol.

The last exercise in the laboratory course deals with the simulation of a larger system, the integration of an SDL-described system in an external environment and the visualization using a Tcl/Tk based graphical user interface, developed at the institute. This exercise will be described in more detail in the section 3. Figure 1 shows the sequence of teaching units in the laboratory course.

Lesson 1	Lesson 2	Lesson 3	Lesson 4	Lesson 5	Lesson 6	Lesson 7	Lesson 8	Lesson 9
Intro- ductory lecture	Tutorial 'Getting started'	Small state machine example	Stop & wait protocol I	Stop & wait protocol II	Token ring protocol I	Token ring protocol II	ISDN Q.931 I	ISDN Q.931 II
Theory	Basic tool handling	States, Signals	System structure	Comm. & Data concept	Larger example, Object orientation		Larger example, Integration in ext. Environment, Visualization	

Figure 1: Teaching units of the laboratory course

A final exam concludes the laboratory course. Only students who have solved all exercises correctly are allowed to take this exam. As a proof for solved exercises MSCs have to be sent to the supervising tutor via email showing the correct behavior of the simulated systems.

3 ISDN protocol case study: From specification to visualization

The last exercise of the lab course is a case study of the Q.931 signaling protocol. It is a larger example, showing the use of SDL in complex systems. The students learn how a SDL-based system can be integrated in an external environment. Furthermore, the students get to know a graphical user interface for visualizations, which has been developed at the institute. Besides that, the exercise is also useful to teach the functionality of the Q.931 protocol itself.

3.1 ISDN signaling protocol Q.931

ITU-T Q.931 [Q.931] defines the user-network protocol for the ISDN signaling channel (D-channel) on layer 3 of the OSI reference model. The Q.931 protocol was chosen, because it is a well defined and standardized signaling protocol and is well known by the students from other lectures. In addition to that parts of the Q.931 recommendation already use SDL syntax for the illustration of the protocol definition. Since this exercise's main target is to show specification, simulation and visualization of a sample telecommunication protocol, we concentrate on the aspects concerning basic layer 3 messaging to be realized by the students and skipped other aspects and details of the protocol.

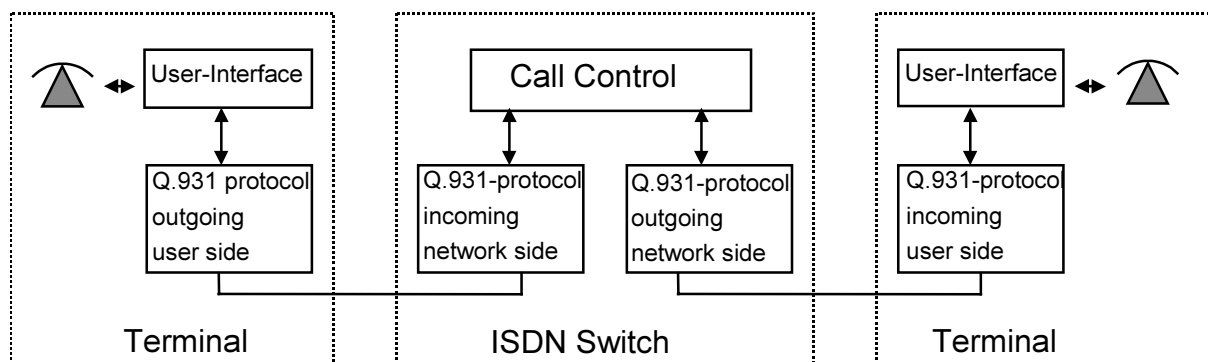


Figure 2: Q.931 layer 3 signaling protocol

Q.931 uses call modeling to control ISDN communication. In this way the relationship between two partners (terminal or switch) in an ISDN call is divided into two sides, an outgoing side and a incoming side. Accordingly, four finite state machines are defined for the protocol specification (figure 2): Outgoing user side, incoming user side, outgoing network side and incoming network side.

The exercise for the students is divided into three parts. First they have to complete given specifications of the four finite state machines which define the Q.931 signaling protocol on the user and network side. For an easy comparison of the students' results the system structure and the start transitions of each process are predefined. The next step is to define a precise specification of the call control within the switch. In the final step the user side has to be connected to the graphical user interface to allow input into the SDL simulation via the GUI

and to visualize the results to the user. In that way the normal hardware user interface (telephone set) is replaced by the GUI.

3.2 SDL specification of the Q.931 system

As already mentioned, the user and network side of the Q.931 system is modeled in the SDL system specification. Figure 3 shows the structure of the SDL system specification. Since the Q.931 exercise is the final exercise and the students already have some experience in defining SDL systems, the system structure and signal definitions are given. Only the process specifications of the User and Network processes (printed in bold) have to be completed. The incoming side and the outgoing side of the user and network processes are defined within the same process specification. The *Process CallControl* must be specified completely by the students.

At the user side multiple terminals may be connected to the ISDN switch. This is realized using the type concepts of SDL-92 and creating multiple instances of the specified *Process Type User*. Since the dialing information in the chosen system definition consists of a single number, a maximum number of 10 terminals (corresponding to 10 user processes) can be connected to the system.

Only a single ISDN switch is assumed on the network side. This resembles for example the system model of a private ISDN PBX. For each user process one corresponding network process is created in the *Block Q_931_Network*.

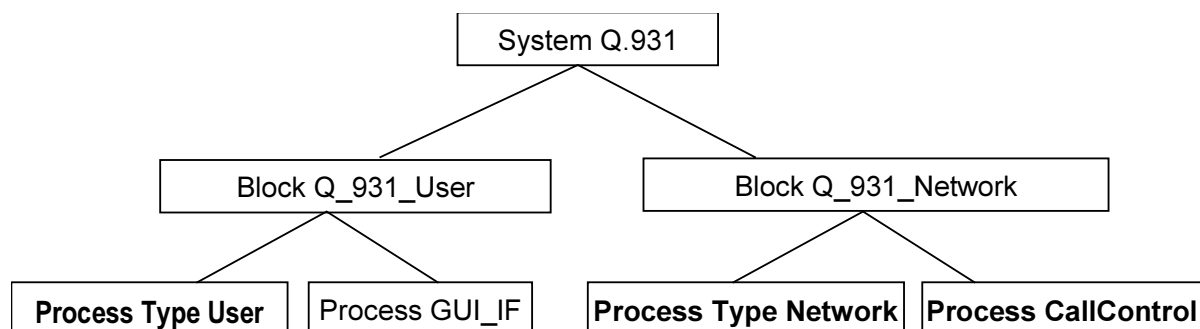


Figure 3: The SDL structure of the Q.931 system

The *Process CallControl* is the main process the students have to specify. In this process, the basic call processing (call setup and clearance) and the call management have to be defined. In this way the Call Control has to realize protocol processing between the network side Q.931 protocol instances. For the management of the different calls a data base has to be set up and maintained, which stores call relationships and addresses. That means to specify a table containing the PIDs and the call context of the Q.931 processes. Besides this basic call functionality, supplementary features like “suspend” and “resume” or “conferencing” are foreseen to be implemented by the students optionally.

The Process GUI_IF realizes the interconnection with the graphical user interface. This process is completely defined and no specification needs to be done by the students.

For the requirements specification selected pages of the ITU-T Q.931 recommendation are provided to the students. In addition to that an MSC, which illustrates the messages between

the different protocol instances, is part of the definition of the requirements. The MSC is shown in figure 4.

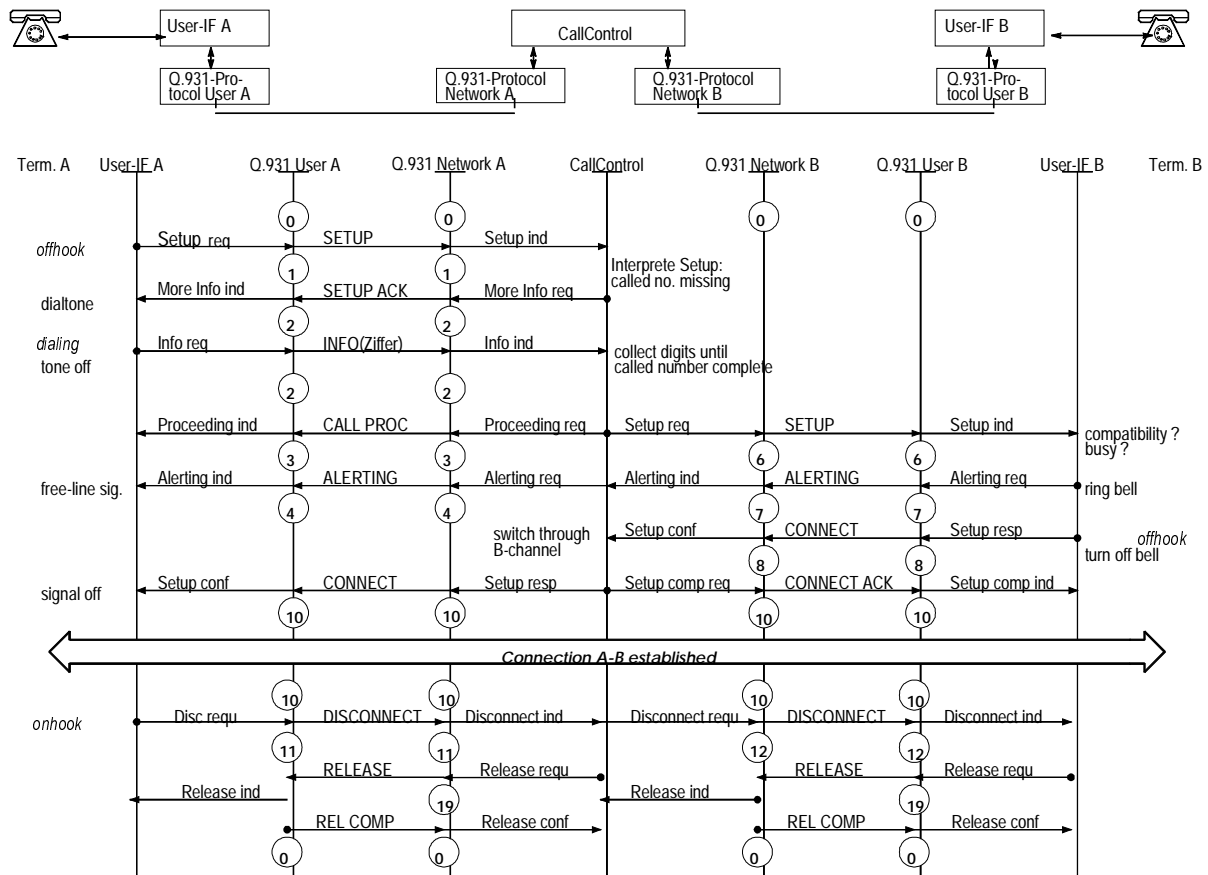


Figure 4: Q.931 message sequence chart

3.3 Graphical User Interface

For the Q.931 system a user interface has been developed using Tcl/Tk as implementation language. Tcl/Tk is an interpreted language and consists of two parts, the Tcl interpreter and the Tk graphical library [Ous94]. The Tcl interpreter is a general purpose interpreter similar to an UNIX C-shell interpreter. Tcl offers a set of well known command constructs as “if then else” or “while”. The use of procedures allows a clear structuring of the system. To support graphical representations the Tk library is provided. This library contains many powerful functions to create graphical objects like buttons, text and geometric objects. The communication with other applications, which in our case is the SDL-simulation, can be realized via socket connections.

For the socket connection of the SDT tool set to the Tcl/Tk user interface a communication agent (‘sdttcl’) is used, which has been developed at the Institute for Communication Networks (figure 5). The agent realizes a bi-directional socket server between the Tcl/Tk GUI and the ‘Postmaster’, a communication agent provided by SDT [SDTa].

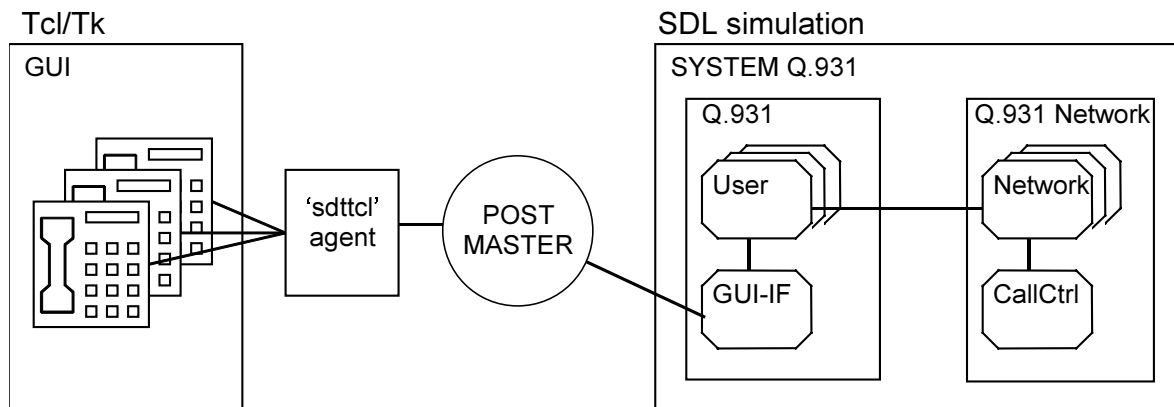


Figure 5: Connection of the graphical user interface to the SDL system

For the laboratory course the GUI and the communication agent are provided to the students completely finished. They serve mainly as an example to show how an SDL-specified system can be integrated in an external environment. Furthermore, the students learn how the graphical user interface works, which is used in several projects at the institute.

3.4 The final system

When the students have completed their system specification, they can syntactically and semantically analyze the system and compile it to generate an executable application. The last step of the exercise is then to connect this executable to the graphical user interface. For this purpose, the SDL system executable, the 'sdtctl' agent and the graphical user interface have to be invoked.

Using the graphical terminals, the simulation of the Q.931 protocol can be triggered. For example, it is possible to place a call by simply clicking on the handset and then dialing the destination number. When the Q.931 system is correctly specified, the message 'Alerting' is displayed at the calling side and a bell is shown at the called side (figure 6). That way, the complete call setup and clearing can be simulated. In addition to the reactions of the graphical user interface, a message sequence chart (MSC) can also be generated automatically by the SDT tool.

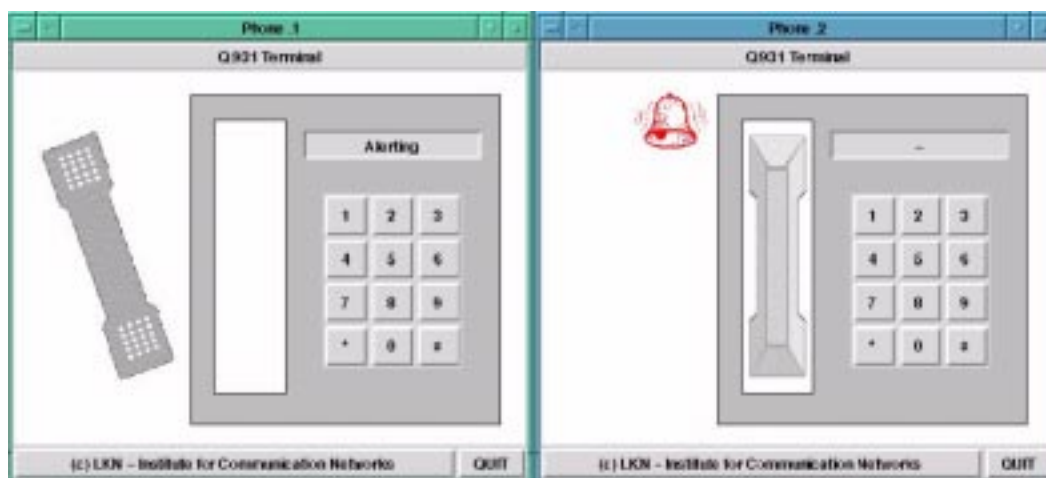


Figure 6: Graphical user interface of the Q.931 system

To test the correct behavior of the system, the students have to generate a MSC of multiple successful call setups and clearances. Only with multiple calls the correct processing of the CallControl can be verified.

Besides the integration of an SDL system in its environment, this last exercise shows the students an example for a large SDL system. Hereby they are taught, how to work within a predefined framework. This is especially important for projects, with several students working on one system, like in diploma theses, which are part of research projects.

4 Experiences and conclusion

The application of SDL to protocol engineering produces a number of insights in the functional system design process as well as in the protocols themselves concerning educational aspects. Since formal description languages are currently not part of the mandatory lectures for students of 'Electrical Engineering and Information Technology' at the Technical University of Munich, the laboratory course on systems engineering using SDL is a valuable extension of the curriculum.

The structure of the presented lab course, composed of an introductory lecture combined with practical exercises which are solved by the students on their own, has worked out very well for the teaching staff and is highly appreciated by the students. The tutoring system and the validation of the exercises using email gives the students the freedom to organize their time to work on the exercises according to their personal preferences. In the final exam the theoretical understanding of SDL as well as the capability to solve a given problem using SDL specifications can be tested.

The graphical representation of SDL makes this language well suited for a laboratory course and for student projects. Working with commercial CASE tools brings "life" into standards and protocols known from lectures, since the protocols can easily be simulated. Very soon, the students can start with the specification of small to moderately sized systems. The integrated environment simplifies the specification, verification and simulation of the systems.

The hierarchical structure of SDL demands a top down design within the system development. We experienced that this way of thinking is in contrast to the trial and error programming students are often using with conventional implementation languages. Thus SDL guides them to make the structuring of their systems into blocks and processes the first task of the system specification and to spend more time on the elaboration of the system structure.

The well defined interfaces and inherent good documentation of the SDL systems allow even the definition of large systems in student projects. Several students can work in a team at the same project, or a project may be continued by another student with only a small period of vocational adjustment. This has proven to be very advantageous in several research projects carried out at the authors' institute, which otherwise hardly would have been possible.

Therefore both, the students and the institute, benefit from the lab course. The experience also shows that partners from industries very much appreciate the students' knowledge in SDL and are offering interesting jobs in this field. Currently, other european universities (University of Braunschweig, University of Gent) are also preparing SDL lab courses, based on the course material presented in this paper.

5 References

- [OFM+94] A. Olsen, O. Færgemand, B. Møller-Pedersen, R. Reed, J.R.W. Smith. *Systems Engineering Using SDL-92*. Elsevier Science B.V., Amsterdam, 1994.
- [Z.100] ITU-T Recommendation Z.100. *Specification and Description Language (SDL)*, ITU, 1994.
- [Z.120] ITU-T Recommendation Z.120. *Message Sequence Chart (MSC)*. ITU, 1993.
- [SDTa] SDT 3.2 documentation, Telelogic AB Malmö, Sweden, 1997.
- [SDTb] *Getting Started With SDT 3.2*, Telelogic AB, Malmö, Sweden, 1997.
- [Q.931] ITU-T Recommendation Q.931. *Digital Subscriber Signalling System No. 1 (DSS1), ISDN User-Network Interface Layer 3, Specification For Basic Call Control*. ITU, 1993.
- [Ous94] J. Ousterhoud. *TCL and the TK Toolkit*. Addison-Wesley, 1994.
- [802.5] IEEE 802.5 - ISO/IEC 8802-5:1995 (Token Ring)
- [KIR96] W. Kellerer, A. Iselt, R. Riek. *Using SDL for the Specification, Simulation and Implementation of an Advanced OSI Data-Link Protocol on an embedded Microcontroller System*. In Gotzhein R., Bredereke J. (Edts.): *Formal Description Techniques IX: Theory, application and tools (Proceedings of FORTE/PSTV'96)*, Chapman&Hall, 1996, pp. 419-434.
- [VK+98] H.-J. Vögel, W. Kellerer, S. Karg, M. Kober, A. Beckert, G. Einfalt. *SDL-based prototyping of ISDN-DECT-PBX switching software*. In proceedings of 1st Workshop of the SDL Forum Society on SDL and MSC, Berlin, 1998.
- [IsAu97] A. Iselt, A. Autenrieth. *An SDL-based platform for the simulation of communication networks using dynamic block instantiations*. In proceedings of SDL'97 Conference, 1997.
- [KeAu97] W. Kellerer, A. Autenrieth. *Praktikum Systementwicklung mit SDL, Unterlagen und Aufgaben (course material), Version 1.21*. Lehrstuhl für Kommunikationsnetze, Technische Universität München, 1997.