In Proceedings of the *8th International Conference on Telecommunication Systems — Modeling and Analysis*, pages 430–437, ATSMA, IFIP, 2000.

Formal Development of a Distributed Control System for Road Marking Machines[*]

Peter Herrmann, Manfred Noël

Universität Dortmund, FB Informatik, LS4, 44221 Dortmund, Germany
Email: Peter.Herrmann@cs.uni-dortmund.de | Manfred.Noel@t-online.de

Abstract

Formal specification and verification of large distributed systems, like computer controlled technical plants, is a rather demanding task. In spite of its complexity, a system should be described in a clear and easily understandable way. Moreover, one should be able to verify important system properties within an acceptable amount of time. The formal specification technique cTLA facilitates the development of complex system descriptions. A system specification is composed from much simpler behavior constraint descriptions each modeling only a certain part of a system component. The compositionality of cTLA and the modularity of the system specifications make proofs of system properties easier, too. Mostly, a system property is not fulfilled by the whole system but by the interaction of some system components. This is reflected in the cTLA proofs which are performed by partial, less complex subsystem models describing only relevant system parts.

In this article, we show the use of cTLA to specify and verify distributed hybrid technical systems by means of an application in practice. Due to a German state regulation, producers and users of road marking machines have to prove formally that the thickness of the applied road markings obey certain limits. The use of cTLA to specify a particular road marking machine as well as to perform the formal proof that this machine obeys the required limits are described.

[*]This work was funded by the German research foundation DFG

1 Introduction

Hybrid technical systems, i.e., continuous technical processes controlled by a discrete computer system, are special distributed systems. The interface between a discrete controller and a continuous process is provided by sensors and actors which are linked to the controller by means of a network. The network has to guarantee functional correctness as well as certain real time assumptions in order to facilitate a timely reaction of the controller on process state changes.

Besides of safety properties preventing serious system hazards, a hybrid technical system has also to fulfill functional requirements in order to guarantee the quality of the produced product. Since many hybrid systems form an economical and ecological risk, one should guarantee that a system fulfills the desired properties by means of formal methods. Due to the complexity of hybrid systems, however, formal specification and verification is a rather time- and cost-consuming task. Here, the specification technique cTLA [6, 15] proves useful since it makes specification and verification easier. The development of system specifications is facilitated by composition of smaller specification blocks each modeling only a single system component. Thus, the system structure can be modeled in a quite direct and open manner. Moreover, cTLA supports superposition, a special kind of compositionality, guaranteeing that properties fulfilled by a single component are also properties of a system containing this component [8]. Therefore, one can reduce the verification of system properties as well. Instead on the complex system model property proofs are based on subsystem models only. If a system is composed from components in a suitable way, these subsystem specifications are quite simple and the proof can be performed easily.

The structure of many distributed and hybrid systems, however, does not support this kind of structured verification since certain system properties are realized by a lot of cooperating system components and the subsystem specifications used for proofs tend to be large. Therefore, cTLA supports not only resource-oriented specifications where each physical component of a system is modeled by a separate specification, but also constraint-oriented specifications (cf. [17]). Here, system components are not specified as a monolith but are composed from constraint specifications each describing only a certain aspect of the component. For instance, an entity of a communication protocol is described by constraint specifications each modeling certain protocol mechanisms (e.g. sequence number handling, data repeat requests). Constraint-oriented specifications make the structured verification of system properties easier. The subsystem model used for property proofs consists only of specifications of the component constraints which realize a certain system property. For example, to prove that a communication protocol prevents the delivery of duplicated data, one only uses the sequence number handling constraints of the transmitter and receiver entities but not the data repeat requests.

cTLA is based on Leslie Lamport's specification technique "Temporal Logic of Actions" (TLA) [14], which was supplemented by a compositional process concept (cf. [12]). Processes encapsulate private system state variables. State transitions are specified by actions each describing a class of transitions. As in the formal description language LOTOS [11], interactions between processes are modeled by joint system actions. These are coupled from local process actions which have to occur simultaneously. Data transfer between processes is described by means of data parameters of actions. The formal semantics of cTLA specifications is defined by a mapping from cTLA specifications to equivalent TLA formulas. cTLA was successfully used to model and prove communication protocols (e.g. [9]).

In order to model real time assumptions of controllers and networks as well as continuous flows of technical systems, cTLA was supplemented by further syntactical constructs [8] which also correspond to similar TLA concepts [1, 13]. By real time constructs one can define that an action must not be enabled for a certain period of time without being executed. A special action facilitates the description of continuous flows by means of difference equations.

In [5], we showed that this extension of cTLA is useful to specify hybrid technical systems and to prove that certain safety properties are fulfilled. Here, we will outline the use of cTLA to verify functional properties of hybrid system control. As an example we use a road marking machine. In Germany, most road markings are applied to public and private roads by these kind of machines. The road markings consist of a mixture of white paint and glass beads which reflect headlights at night. The visibility at night depends highly on the thickness of the material applied to roads. If the thickness is too high, the glass beads sink completely into the paint and do not reflect headlights anymore. If the thickness is too low, the road markings do not hold the glass beads which are weared by tyres. Since the cost of the paint is between 40 and 60 % of the total marking cost and the thickness can hardly be measured afterwards, contractors often apply material with a lower thickness. From the year 2000 on, the government tries to prevent applying markings of a wrong thickness by a new regulation [3]. The contractors have to use an electronic control and record system guaranteeing that the thickness of road markings exceeds or falls below the desired value by less than 10%. Furthermore, the regulation lays down that the functional requirements of the electronic control system have to be proven formally. In this article, we will show the formal specification and verification of a control system developed in [16]. The components of the control system, the road marking machine, and the network linking both components were specified in cTLA. Moreover, we verified that the control system fulfills the regulation by means of a deduction proof in temporal logic.

In the remainder we will give a short introduction to cTLA and, in particular, to the constructs specifying real time and continuous flows. Afterwards, we will outline the road marking machine and the control system and introduce the corresponding compositional cTLA specification. Finally, we will sketch the formal proof that the control system guarantees the required thickness of road markings.

2 cTLA

While cTLA is used as a notation of canonical TLA formulas, its syntax is oriented at programming languages. Specifications are described by process types. The process instantiations specify either single system components resp. component constraints or systems composed from components. In figure 1 we outline a simple process modeling a link between a control unit

```
Safe simplex link between the controller and an actor resp. sensor
PROCESS Link (DataUnits : Any)
                    DataUnits : format of the transmitted data units
VARIABLES
  q : QUEUE OF DataUnits;  queue of data currently in transmission
INIT ≜ q = empty;  initial condition
ACTIONS
  SEND (d : DataUnits) ≜  send a data unit d
    q′ = append(d,q);
  DELIVER (d : DataUnits) ≜  deliver a data unit d
    q ≠ empty ∧ d = first(q) ∧ q′ = removefirst(q);
END
```

Figure 1: cTLA process type *Link*

and sensor resp. actor devices. The process header consists of the process name and generic process parameters (fi., `DataUnits`) which enable the definition of instances with different properties by a single process type. The state transition system is described in the process body. The states are modeled by variables (fi., `q`). The predicate `INIT` describes the set of initial states. Transitions are specified by so-called actions (fi., `SEND`) which are predicates on pairs of current and next states as well as parameters (fi., `d`). While variables referring to the current state are described by the pure variable identifiers (fi., `q`), the next state descriptors are marked by the symbol `'` (fi., `q'`). The set of process transitions corresponds to the disjunction of the process actions. Furthermore, transitions can be so-called stuttering steps where the current and next states are equal.

The example process type *Link* models a pure safety property (cf. [2]) since it tolerates that actions are never executed in spite of being enabled infinitely often. System progress is modeled by liveness properties containing fair actions. A fair action is forced to be executed eventually if it is often enabled (i.e., the local enabling condition is true and the process environment tolerates the execution as well). An action may be weak fair (described by the fairness assumption `WF: DELIVER`). Then it has eventually to be executed if it would be enabled incessantly otherwise. A strong fair action (described by `SF: SEND`) has to be executed even if it is sometimes disabled.

Fairness assumptions are not sufficient to model real time properties since they do not reflect exact minimum or maximum times, an action may be enabled without being executed. In cTLA, real time is realized according to the principle of virtual clocks (cf. [1]). A special real-valued state variable `now` represents the time. The action `tick` increments `now` in very small steps. Unlike other state variables which are private to exactly one process, the clock variable `now` can be read by all processes of a system. `now` forms the basis for defining real time and continuous properties. cTLA offers constructs modeling activity retarding and activity forcing constraints. In accordance with [1], one can define minimum waiting times and maximum reaction times for actions. Like the distinction between weak and strong fairness the waiting and reaction times may be volatile or persistent. The notation corresponds to fairness assumptions and consists of four parts. First, one determines by the keywords `V` or `P` if the modeled real time is volatile resp. persistent. At second, one defines the kind of real time constraint by the keywords `MIN TIME` (minimum waiting time) or `MAX TIME` (maximum reaction time). Finally, one gives the name of the constrained action and the period of time modeled. For instance, in the specification in figure 2 we model by the assumption `V MAX TIME: DELIVER time` that the action `DELIVER` has to be executed before it is incessantly enabled for longer than the period of time modeled by the process parameter `time`.

In order to specify hybrid systems we have to describe continuous behavior, too. cTLA processes model continuous flows by means of a special timed action `CONT` containing difference equations on real-valued state variables. Since `CONT` is linked to the clock action `tick`, the time steps of `CONT` are very small and the difference equations approximate continuous flows (cf. [13]). Continuous inputs and outputs are modeled by special action parameters. As an example we listed in figure 3 the process *Thickness* specifying the thickness of the material applied to the road. The variable `th` describes the thickness currently applied. The thickness depends on the velocity of the machine and on the flow of material applied to the road which are modeled by the input parameters `vi` and `fi` of action `CONT`. The current value of

```
Maximum time between two data deliveries
PROCESS MaxDeliverTime (time : Real)  time : period of time
ACTIONS
  DELIVER; deliver a data unit
V MAX TIME: DELIVER time
END
```

Figure 2: cTLA process type *MaxDeliverTime*

```
Material thickness applied to the road
PROCESS Thickness (lw : Real) lw : width of the marking
VARIABLES
  th : real; current material thickness
INIT ≜ th = 0;
ACTIONS
  CONT (INPUT vi, fi : Real;  vi : velocity of machine;
                              fi : flow of material
        OUTPUT to : Real) ≜   to : material thickness
  Continuous behavior
    to = th ∧
    th' = fi / (vi · (now'-now) · lw);
END
```

Figure 3: cTLA Process type *Thickness*

```
System consisting of a feed-back controller, road marking machine, links, and the road
PROCESS ControlledSystem (lw : Real;  lw : width of the marking
                          dt : Real)  dt : desired material thickness
PROCESSES
  V    : Velocity (-7, 7, 0.35, 0.7);  velocity of the machine
  F    : Flow (100000,1500);            flow of material applied to the road
  T    : Thickness (lw);                material thickness
  CF   : ControlFlow (0.377, dt);       control of machine depending on the flow
  CFR  : ExecutionTime (0.015);         real time constraint modeling maximum
                                        exec. time of the feed-forward controller
  CFW  : WaitingTime (0.005);           real time constraint modeling minimum
                                        exec. time of the feed-forward controller
  SF   : SensorFlow;                    sensor of the flow of material
  SFR  : ExecutionTime (0.015);         max. time for flow sensor reaction
  SFW  : WaitingTime (0.005);           min. time for flow sensor reaction
  VV   : Valve;                         valve controlling flow of material
  LSF  : Link (Real);                   link between flow sensor and controller
  LSFR : MaxDeliverTime (0.005);        maximum time to deliver data from
                                        the flow sensor to the controller
  LVV  : Link (Real);                   link between controller and valve
  LVVR : MaxDeliverTime (0.005);        maximum time to deliver data from
                                        the controller to the valve
  ...;
ACTIONS
  CONT (OUTPUT ... : Real) ≜ Continuous behavior
    V.CONT (; vo) ∧ F.CONT (; fo) ∧ T.CONT (vo, fo; to) ∧
    VV.CONT (; vvo) ∧ SF.CONT (fo ;) ∧ ...;
  READFLOW (fl : Real) ≜ Read flow in sensor and send flow to controller
    SF.READ (fl) ∧ SFR.SIGNAL ∧ SFW.SIGNAL ∧ LSF.SEND (fl) ∧ ...;
  RECEIVEFLOW (fl : Real) ≜ Receive flow at controller
    CF.READFLOW (fl) ∧ LSF.DELIVER (fl) ∧ LSFR.DELIVER ∧ ...;
  SENDSETTING (vs : Real) ≜ Calculate valve setting and send setting to the valve
    CF.SETVALVE (vs) ∧ CFR.SIGNAL ∧ CFW.SIGNAL ∧
    LVV.SEND (vs) ∧ ...;
  SETVALVE (vs : Real) ≜ Receive flow at valve and set valve
    LVV.DELIVER (vs) ∧ LVVR.DELIVER ∧ VV.SET (vs) ∧ ...;
  ...;
END
```

Figure 4: cTLA system type *ControlledSystem*

`th` is calculated by the difference equation `th' = fi / (vi` $\cdot$ `(now'-now)` $\cdot$ `lw)` where the difference `(now'-now)` specifies the time step modeled by `CONT`. The output parameter `to` exports the current material thickness to other processes.

Several constructs stating safety, fairness, real time, and continuous properties may be contained in the same process type specification. In order to provide a fine-grained constraint-oriented process structure, however, we recommend to use process types each concentrating only on one property of a single type.

Single process specifications can be composed to larger specifications modeling systems or subsystems. Each process encapsulates its local state which can be changed by actions of this process only. As in the formal description technique LOTOS [11], the coupling between processes is modeled by joint system actions which are coupled from synchronously executed process actions. Data transfer between processes is modeled by the action parameters since in a joint system action the parameters with the same name have to carry identical values. Each process corresponds to a system action either by exactly one process action or by a stuttering step. Thus, concurrency is modeled by interleaving.

The process *ControlledSystem* in figure 4 is an example of a cTLA system specification. The processes composed to the system are described in the section `PROCESSES`. Here, the process name (fi., `V`), the process type (fi., `Velocity`), and the parameter settings (fi., `(-7, 7, 0.35, 0.7)`) are listed. The coupling of process actions to joint system actions is specified in the section `ACTIONS`. For instance, the action `SETVALVE` models the setting of a valve. It is coupled from the process actions `DELIVER` of process `LVV` resp. `LVVR`, specifying constraints of the data link to the valve, and `SET` of process `VV`, describing the valve. The other processes participate to this system action by stuttering steps. The action parameter `vs` models the exchange of data (i.e., the desired setting of the valve) between the processes `LVV` and `VV`.

3 Road marking machine and controlled system

As described in the introduction, side and center markings of roads are applied by motorized road marking machines. These machines use one or two material containers with a capacity of 100 till 200 gallons which are pressurized by a compressor. By pipes the paint is lead to one or two spray-guns and applied to the surface. The machine is driven with a velocity

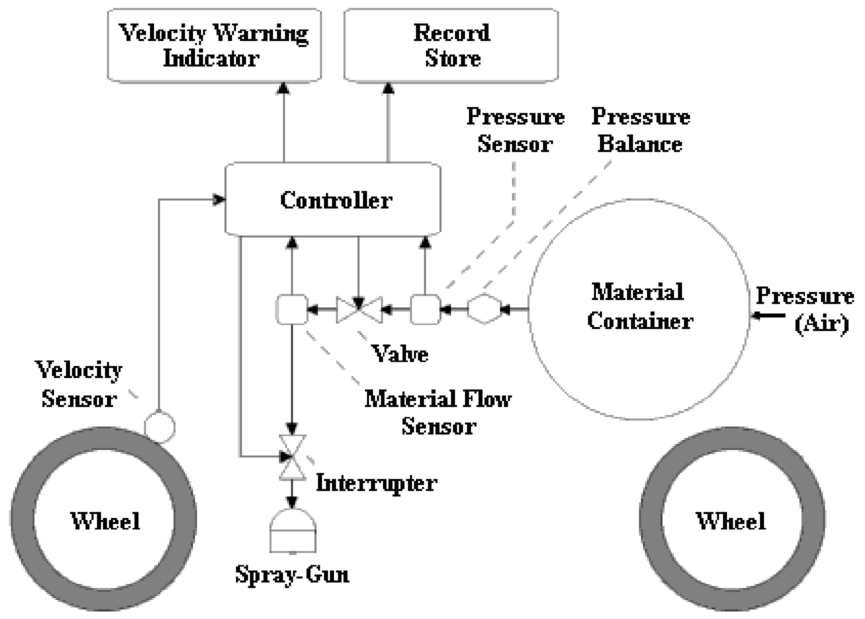

Figure 5: Road marking system

depending on the width and the thickness of the markings as well as on the material. Usually, the machine is run with 3 to 16 miles per hour. In order to guarantee correct segment lengths in the case of broken line markings, the application of material can be driven electronically.

Figure 5 outlines the road marking system used in our proof. The paint is piped from the material container through a pressure balance device, a pressure sensor, a valve, a material flow sensor, and an interrupter to the spray-guns. A feedback-controller controls the flow of paint by the valve. Self-triggered sensors gauge the current velocity, flow of material, and pressure of the machine. By the interrupter the flow of paint can be stopped immediately. It is switched by the machine operator as well as by the controller. The two actors and the three sensors are linked to the controller by a simple network. In order to make the control easier, the large pressure difference in the material container (3 to 6 Bar) is reduced by the pressure balance device reducing the pressure to a maximum of 4 Bar.

The main task of the controller is to guarantee that the material thickness does not exceed or fall below the limits specified by the state regulation. The material thickness depends on the flow of material, the line width, and the velocity. Based on the line width and the current velocity, the controller controls the material thickness by adjusting the flow of paint depending on the current flow and velocity. Testing and the formal verification sketched in section 5 proved that a simple proportional controller is sufficient to fulfill this task.

The controller closes the interrupter if the pressure in the material container is too low. Furthermore, the flow of paint is switched off if the operator drives too fast or too slow to guarantee the correct material thickness. Thus, the machine cannot be operated too fast in order to save paint. A short time, before the interrupter is closed due to a wrong velocity, the operator is alerted by a warning indicator. To inspect operations of the machine afterwards, the line width, velocity, flow of paint, and pressure are recorded.

4 Specification

The road marking machine and its environment are specified by the cTLA process type *ControlledSystem* outlined in figure 4. The process type is composed from various specifications modeling constraints of the continuous system flows, the actors, the sensors, the links, and the discrete controller. The continuous flows are specified by four cTLA process instances dealing with the velocity, pressure, flow, and material thickness. As an example we discuss the material thickness specified by process *T*. This process is an instance of the process *Thickness* listed in figure 3. The current thickness is modeled by the variable `th`. Since the process describes only continuous behavior, it uses `CONT` as the only action to describe state changes. The material thickness depends on the line width, the velocity, and the flow of material. Since the line width is a constant value during a marking process, it is modeled by the process parameter `lw`. In contrast, the velocity and flow of paint may change dynamically. Therefore, they are specified as input parameters `vi` and `fi` of the action `CONT`. These parameters are calculated by the processes `V` resp. `T`. The current thickness is exported by the output parameter `to` to other processes. The new value of the material thickness, which is described by the value of the variable `th` after executing `CONT` (`th'`), corresponds to the ratio of applied paint and the painted area. The area is the product of the line width and the distance run by the machine.

The distance corresponds to the product of the velocity and the duration `(now'-now)` modeled by `CONT`. Thus the material thickness is calculated as `th' = fi / (vi · (now'-now) · lw)`.

The controller influences the continuous behavior by the valve and the interrupter each specified by a cTLA process. Since the three sensors measuring the velocity, the pressure, and the flow of paint are self-triggered, they are modeled by three cTLA processes each. One process describes the reading and transmission of values while the other processes define real time assumptions. The second process guarantees that the sensors transmit data not after a certain amount of time while the third process models that data is send not too often in order to prevent jamming the link to the controller.

The controller is linked to each of the five actor and sensor devices. A link is described by two cTLA processes. The functional quality of service is modeled by instances of the process type *Link* outlined in figure 1. The link guarantees the delivery of transmitted data without reorderings, losses, duplicates, corruptions, and phantoms. Instances of the process type *MaxDeliverTime* describe the hard real time assumption that a data unit in transmission must be delivered not after a duration of `time` time units since the delivery of its predecessor.

The discrete controller fulfills two tasks. At first, the interrupter is closed if the pressure in the container is below a minimum value which is modeled by the process type *ControllerPressure*. At second, the controller controls the valve and the interrupter in order to guarantee a correct material thickness. We describe this control unit by two separate constraint processes. The process types *ControllerFlow* and *ControllerVelocity* model the control depending on the flow of paint resp. velocity. In *ControllerFlow*, the setting of the valve and the interrupter is calculated based on the value, received from the data flow sensor, under the assumption that the velocity did not change. The adjustment of these settings according to changes of velocity is modeled by *ControllerVelocity*. The adjusted settings are transmitted to the actor devices. Four processes model the real time assumptions of the controller. Two processes specify the maximum reaction time to calculate the settings for the valve resp. interrupter in order to guarantee on-time reaction on changes in the continuous system part. The other processes restrict the transmission of data in order to prevent late deliveries of obsolete setting commands due to jams in the links.

5 Verification

We will outline the proof that the road marking machined applies paint with a thickness of at most 10% below or above the desired value according to the state regulation. This property can be described by the TLA formula

$$I \triangleq (0,9 \cdot dt \leq T.th \leq 1,1 \cdot dt) \vee (I.open = 0)$$

Due to the formula either the current thickness which is modeled by the variable `th` of the process T (figure 3) differs at most by 10% from the desired value `dt` or the interrupter I is closed. We have to prove that the formula I is invariant within the system *ControlledSystem* which is modeled by the TLA formula[1]:

$$ControlledSystem \Rightarrow \Box I$$

Invariant proofs are carried out in two steps. At first, one verifies that I holds in the initial state of a system, i.e., it can be inferred from the initial conditions of the processes composed to *ControlledSystem*. At second, one proves that if I holds before an execution of a system action of *ControlledSystem*, it holds afterwards, too. Due to the compositionality of cTLA we do not need the complete description in order to prove I but a smaller subsystem consisting of the processes modeling the continuous flows, the interrupter, the link to the controller, the flow and velocity sensors, the links from these sensors, and the components of the controller influencing the interrupter.

Unfortunately, I is too weak to be verified directly. Instead, we have to prove a stronger invariant I_s implying I. Since we use a smaller subsystem, it was easy to understand the relevant aspects of the subsystem behavior. Therefore, it was not too hard to find I_s which is a conjunct on I and some properties fulfilled by *ControlledSystem*:

- The interrupter may be open only if the thickness of the applied paint is within the limit of 10%.
- The interrupter is already closed when the thickness falls below or exceeds the limits.

[1]The temporal operator $\Box$ defines that a predicate has to hold in every state reached by the modeled system.

The formula $Subsystem \Rightarrow \Box I_s$ could easily be verified by an invariant proof as described above. Since I_s implies I, the formula

$$Subsystem \Rightarrow \Box I$$

also holds.

The compositionality of cTLA guarantees that a safety property of a subsystem is also a property of a more comprehensive system which contains the processes of the subsystem. With respect to liveness and maximum reaction time real time properties, however, we have to check that the actions with fairness resp. real time assumptions are not blocked by other processes of the comprehensive system. In our example this proof is merely trivial since all processes of *ControlledSystem* which are not in the subsystem participate to the actions of the subsystem either by stuttering steps or by actions which are always enabled. Thus, also the formula $ControlledSystem \Rightarrow Subsystem$ holds and I is an invariant of *ControlledSystem*. Thus, the road marking systems fulfills the limitations of the regulation. By another proof we showed that, except for emergency stops, the valve control is sufficient to prevent switching off the interrupter due to velocity violations.

6 Conclusion

In this paper, we showed that cTLA is a useful means to specify and verify hybrid distributed systems. System specifications which are composed from component and constraint descriptions are usually easier to understand than monolithical specifications. The compositionality of cTLA also facilitates formal proofs since one mostly can use relatively small subsystems making the design of the proof structure as well as the detection of suitable invariants simpler. The example was specified and verified within a week.

cTLA also supports the reuse of specifications and proofs. So-called specification frameworks consist of libraries of process types and theorems. In [10] a specification framework was introduced which supports formal hazard analysis of hybrid systems. The framework contains a library of process types modeling certain components and constraints of hybrid chemical plants as vessels, pipes, sensors, actors, and discrete controller resp. network components. A specification is developed by instantiating and composing process types of this library. A second library contains process types modeling typical safety properties (fi., the pressure in a vessel must not exceed a certain value). In order to facilitate the formal proof that a system specification fulfills a safety property, a third library is provided as well. It contains theorems, already proven by the developers, which state that a certain hybrid subsystem realizes a safety property. The user can easily reduce the proof into proof steps which correspond directly to theorems of the framework. Moreover, a tool facilitates the selection of suitable theorems and the necessary consistency checks [4].

Another specification framework was successfully used in the field of high-speed communication protocols (cf. [7, 9]) where we could specify and verify complex transfer protocols like XTP [18] within three weeks. The work presented above, forms the basis for a third framework simplifying the formal specification and verification of the control of hybrid distributed systems.

References

[1] Abadi, M. and Lamport, L., "An old-fashioned recipe for real time", *ACM Transactions on Programming Languages and Systems*, vol. 16, no. 5, 1543-1571 (1994).

[2] Alpern, B. and Schneider, F. B., "Defining liveness", *Information Processing Letters*, vol. 21, 181-185 (1985).

[3] Fachgremium ZTV-M, "Zusätzliche Technische Vorschriften und Richtlinien für die Markierung von Straßen" (in German) (1999).

[4] Herrmann, P., Drögehorn, O., Geisselhardt, W., and Krumm, H., "Tool-supported formal verification of highspeed transfer protocol designs", in: **7th International Conference on Telecommunication Systems — Modeling and Analysis**, ATSMA, 531-541 (1999).

[5] Herrmann, P., Graw, G., and Krumm, H., "Compositional Specification and Structured Verification of Hybrid Systems in cTLA", in: **1st IEEE International Symposium on Object-oriented Real-time distributed Computing (ISORC98)**, IEEE Computer Society Press, 335-340 (1998).

[6] Herrmann, P. and Krumm, H., "Compositional Specification and Verification of High-Speed Transfer Protocols", in: **Protocol Speci-**

fication, Testing, and Verification XIV, eds. Vuong, S. T. and Chanson, S. T., IFIP, Chapman & Hall, 339-346 (1994).

[7] Herrmann, P. and Krumm, H., “Re-Usable Verification Elements for High-Speed Transfer Protocol Configurations”, in: **Protocol Specification, Testing, and Verification XV**, eds. Dembiński, P. and Średniawa, M., IFIP, Chapman & Hall, 171-186 (1995).

[8] Herrmann, P. and Krumm, H., “Specification of Hybrid Systems in cTLA+”, in: **5th International Workshop on Parallel & Distributed Real-Time Systems (WPDRTS'97)**, IEEE Computer Society Press, 212-216 (1997).

[9] Herrmann, P. and Krumm, H., “Modular Specification and Verification of XTP”, *Telecommunication Systems*, vol. 9, no. 2, 207-221 (1998). Also appeared in: **5th International Conference on Telecommunication Systems — Modeling and Analysis**, ATSMA, 477-486 (1997).

[10] Herrmann, P. and Krumm, H., “Formal Hazard Analysis of Hybrid Systems in cTLA”, in: **18th IEEE Symposium on Reliable Distributed Systems (SRDS'99)**, IEEE Computer Society Press, 68-77 (1999).

[11] ISO, “LOTOS: Language for the temporal ordering specification of observational behaviour”, **International Standard ISO/IS 8807 Edition** (1989).

[12] Kurki-Suonio, R., “Hybrid Models with Fairness and Distributed Clocks”, in: **Workshop on Theory of Hybrid Systems**, eds. Grossmann R. L., Nerode A., Ravn A., and Rischel H., Springer-Verlag (LNCS 736), 103-120 (1993).

[13] Lamport, L., “Hybrid Systems in TLA$^+$”, in: **Workshop on Theory of Hybrid Systems**, eds. Grossmann R. L., Nerode A., Ravn A., and Rischel H., Springer-Verlag (LNCS 736), 77-102 (1993).

[14] Lamport L., “The Temporal Logic of Actions”, *ACM Transactions on Programming Languages and Systems*, vol. 16, no. 3, 872-923 (1994).

[15] Mester, A. and Krumm, H., “Composition and Refinement Mapping based Construction of Distributed Applications”, in: **Workshop on Tools and Algorithms for the Construction and Analysis of Systems**, BRICS Notes Series, Denmark (1995).

[16] Noël, M., “Korrektheitssichernder Entwurf eines Reglers für Fahrbahnmarkierungsautomaten” (in German), Diploma Thesis, Universität Dortmund (1998).

[17] Vissers, C. A., Scollo. G., and van Sinderen, M., “Architecture and specification style in formal descriptions of distributed systems”, in: **Protocol Specification, Testing and Verification VIII**, eds. Agarwal S., and Sabnani K., IFIP, Elsevier, 189-204 (1988).

[18] XTP Forum, “XTP transport protocol specification, revision 4.0.”, Santa Barbara (1995).