Available online at www.sciencedirect.com

ScienceDirect

Electronic Notes in Theoretical Computer Science 309 (2014) 51–62

Electronic Notes in Theoretical Computer Science

www.elsevier.com/locate/entcs

Modeling and Verification of RBC Handover Protocol

Kai Yang [1,2]

Institute of Computing Theory and Technology, and ISN Lab
Xidian University
Xi'an, P.R.China

Zhenhua Duan*, Cong Tian[3]

Institute of Computing Theory and Technology, and ISN Lab
Xidian University
Xi'an, P.R.China

Abstract

Radio Block Center (RBC) is a core part of the Chinese Train Control System level 3 (CTCS-3) which is a protocol for safety critical systems. The correctness of the RBC handover protocol is one of the most important factors affecting the safety of systems. Hence, it is of great importance to ensure the correctness of the protocol. To this end, some formal methods have been used to model and verify the protocol. In this paper, we use Modeling, Simulation and Verification Language (MSVL) as the formal language to model and verify the protocol. The result shows that the behavior of RBC handover is consistent with the specification.

Keywords: CTCS, RBC Handover, modeling, verification

1 Introduction

CTCS-3 is a protocol for control train systems in Chinese railway area which is in developing and required to be safety critical. The system requirement specification of CTCS-3 is written in natural language, inevitably containing ambiguities and defects. Ambiguities and defects in requirements specifications may lead to failure in the whole system development. The failure could be fatal in case of safety critical

[1] This research is supported by the National Program on the Key Basic Research Project of China (973 Program) under Grant No. 2010CB328102, and the National Natural Science Foundation of China under Grant Nos. 61133001, 61272117, 61272118, 61202038, 91218301, 61322202 and 61373043. * Corresponding author.

[2] Email: kaiyang@stu.xidian.edu.cn

[3] Email: zhhduan,ctian@mail.xidian.edu.cn

http://dx.doi.org/10.1016/j.entcs.2014.12.005

systems. Therefore, high-quality specifications are required, which should satisfy the quality properties, such as correctness, completeness, consistency, and traceability [16].

In order to describe the requirement more formally, the Unified Modeling Language (UML) is often used to model the system requirement specification. As a visual modeling language, UML has some advantages in system modeling and has become a popular modeling language since its inception. To make use of the characteristics of UML, numerous tools (Rational Rose, Argo UML, Rhapsody etc.) that exist in the market have to be used to support its functions. However, unfortunately, none of them guarantees specification correctness [1]. Therefore, formal verification methods have been adopted to verify whether or not a system satisfies the desired properties. In fact, many other studies have been done in formal analysis of RBC handover with different approaches. In [12], the authors model and verify the RBC handover of European Train Control System level 2 (ETCS-2) with Differential Dynamic Logic. According to the RBC handover protocol, the authors recapitulate the behavior of RBC handover briefly and build a formal model which successfully describes the behavior of the RBC handover. Furthermore, the collision avoidance and derailment avoidance requirements they propose are both fulfilled through the final verification. In [2], RBC of ETCS-2 is formally modeled and validated based on message sequence charts. Using Stochastic Petri Nets, [15] establishes the failure probability models of the handover conduct for ETCS-2 with two different train configurations, and offers the analysis of the influence of different train velocities and RBC overlap on the successful probability of RBC handover. In [11], Faber et al. use the formal language CSP-OZ-DC to describe the specification of RBC of ETCS-2, and give a model checking and reasoning verification method after translating the model into PEA.

Projection Temporal Logic (PTL) [3,4,10] is a useful formalism for system verification. MSVL [7,8,5] is an executable subset of PTL and can be used to model, simulate and verify concurrent systems. To do so, a system is modeled as an MSVL program and a property of the system is specified by a Propositional Projection Temporal Logic (PPTL) formula. Thus, whether or not the system satisfies the property can be checked by means of model checking with the same logic framework [6,9,17]. In addition, asynchronous communication techniques have been implemented in MSVL, hence, MSVL can be employed to model an asynchronous distributed system [14]. In this paper, we use MSVL as the formal language to model the RBC handover protocol and verify some properties of the protocol.

The remainder of this paper is constructed as follows. In section 2, the MSV toolkit for MSVL is briefly introduced. The RBC handover protocol is described in section 3. Section 4 and 5 dedicate to modeling, simulation and verification of the protocol. Finally, we give conclusion in section 6.

2 MSV Toolkit

The language MSVL is a subset of Projection Temporal Logic with framing technique [8]. A toolkit named MSV has been developed and the structure of the toolkit is shown in figure 1. As its name shows, the MSV toolkit can work in three modes: modeling, simulation and verification. With the simulation mode, the values of variables in the program at each state are evaluated and outputted while with the modeling mode, the model of the program is given. A user can input a property described by a PPTL formula with the verification mode and the MSV toolkit will check whether the system satisfies the property or not automatically. If the property is not satisfied, all counter-examples will be given by the MSV toolkit and the system designer can modify the system according to the counter-examples. The verification process is shown in figure 2. In addition, the formal definitions of process structures, channel structures and communication commands are presented in [14], which enable us to model and verify concurrent systems with asynchronous communication.

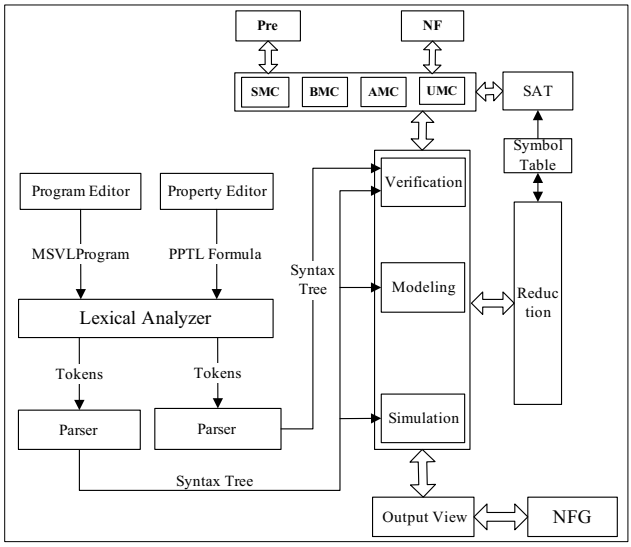

Fig. 1. The Structure of MSV Toolkit

3 RBC Handover Protocol for CTCS-3

3.1 The structure of CTCS-3

Before introducing the RBC handover protocol, we first make a brief introduction of the structure of CTCS-3. CTCS-3 mainly consists of three subsystems, as shown in figure 3, the on-board subsystem, the ground subsystem and the network communication subsystem.

As a core ground subsystem of CTCS-3, the main task of RBC is to guarantee the safety and efficiency of trains and it is a control system designed based on the

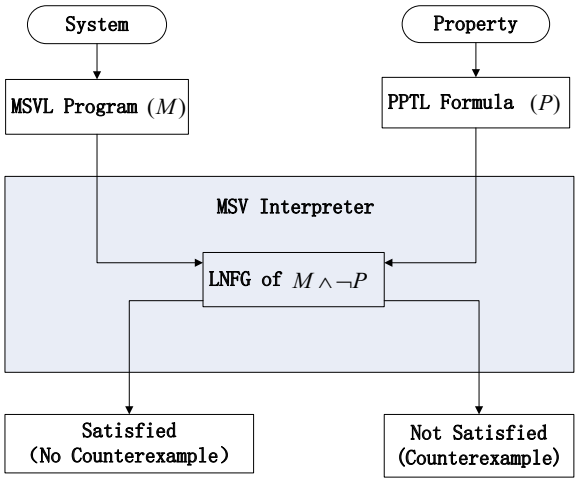

Fig. 2. Verification Process of MSV Toolkit

principle of fail-safe in railway signaling systems. RBC uses the information from other subsystems to compute real-time Movement Authority (MA) and forwards the MA to the train controlled by it.

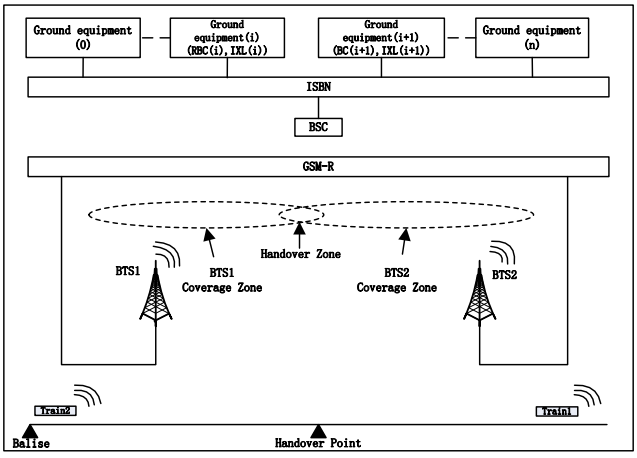

Fig. 3. The Structure of CTCS-3

The RBC subsystem includes communication modules (wireless information process module and ground information process module), control modules (train control order and train movement interval management) and the MA computing module [13].

3.2 RBC Handover Protocol

Due to the limitation of the control capacity of one RBC, multiple RBCs are placed along a railway. When a train arrives at the border of the control area of the current RBC, the control of the train should be handed over to the next RBC

and this procedure is called RBC handover, as shown in figure 4. Under normal circumstances, the procedure is completed automatically and driver's intervention is not needed. Two cases are included in the conduct of RBC handover: 1. RBC

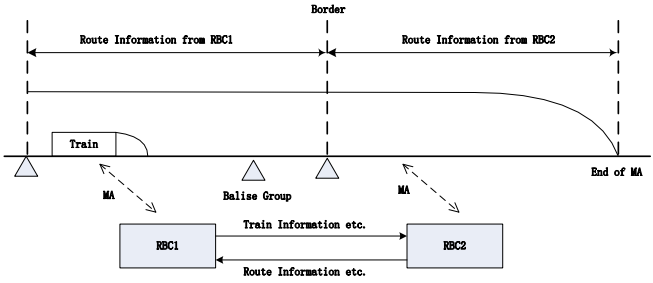

Fig. 4. RBC Handover

hands over with two normal radio stations; 2. RBC hands over with only one normal radio station. The handing over procedures in both cases are different. We focus on the second case here. We use RBC1 to represent the handover RBC and RBC2 the takeover RBC.

The procedure of RBC handover with one normal radio station is as follows:

1) After a train passes the level transition announcement balise group (LTA) for RBC transition, the train will send a position report to RBC1;

2) When RBC1 receives the position report from the train, it will send the RBC transition command (message packet 131) to the train and the route requesting information to RBC2;

3) If there is a route available in the region of RBC2, RBC2 will send Hybrid Movement Authority to RBC1, or it will send message "NoMA" to RBC1;

4) According to the information from RBC2, RBC1 sends extended message "Hybrid MA" or "NoMA" to the train;

5) The train sends a position report to RBC1 after the front end of the train (maximum safe front end) passed the transition border;

6) RBC1 forwards the position report to RBC2;

7) RBC2 sends takeover information to RBC1 after receiving the position report from RBC1;

8) The train sends a position report to RBC1 after the rear end of the train passed the transition border;

9) Based on the position report provided by the train, when RBC1 detects that the rear end of the train (minimum safe rear end) has passed the RBC1/RBC2 transition border, the train will be commanded to close the communication session with RBC1;

10) After the train received the disconnection order from RBC1, it will disconnect the communication with RBC1. The train starts to call RBC2 based on the previous command from RBC1. If the call is successful, the train sends initialization information (M155) to RBC2, RBC2 sends communication version information (M32) to the train, and the train sends communication establishing information

(M159) to RBC2. Then, the communication is established; and

11) RBC2 generates MA and sends the MA to the train. Thus, the transition from RBC1 to RBC2 is completed.

The narrative text description above is very redundant and not intuitive, but detailed. It dedicates on supporting and interpreting the graphic and tabular form in the system requirement specification.

We model the "RBC handover" process using UML sequence diagram. UML sequence diagrams are behavioral diagrams which is a simple, expressive, intuitive, graphical and standardized notation used to specify interactions among system entities in many different situations. Along with class diagrams and use case diagrams, the sequence diagram is the most popular diagram of the UML. The expressiveness of the sequence diagram ensures the consistency between the model and specification of a system [16].

The UML sequence diagram modeling approach maps the interacting entities and the event occurrences of the scenarios to the participants (called objects in UML1.x) and the messages of the sequence diagrams. The modeling steps are listed as follows:

(1) List all the entities (including system components) interacting in the scenario;

(2) Find out all the event occurrences and the associations among the entities of the scenario;

(3) Map the basic elements (including the entities, event occurrences and the associations) of the scenario to the sequence diagram; and

(4) Check the consistency between the scenario and the sequence diagram.

Following the listed steps, we construct the sequence diagram model of the "RBC handover" scenario as shown in figure 5. The meanings of messages in figure 5 are indicated in table 1.

4 Simulation and Modeling

A UML sequence diagram of the procedure of RBC handover is given in the previous section. In this section, we translate the sequence diagram into an MSVL program. There are three main components in the procedure of RBC handover: TRAIN, RBC1 and RBC2. They are independently distributed components running in parallel. Corresponding to each component, we define a process. The three processes are parallel and there is a communication channel between each two of them. The state of a train will change according to the messages it receives. The result of simulation under the normal circumstance (there is no message loss) is shown in figure 6. The meaning of the variables in the program is shown in table 2. The value of the variable "FinishHandover" is equal to 1 at the last state, which means that the procedure of RBC handover is completed successfully.

We run the program with MSVL toolkit under modeling model and all execution paths are shown in figure 7. Due to the fact that the paths are too long to completely show in the figure, only part of the paths is shown. The two paths in figure 7 represent two cases respectively: 1. the procedure of RBC handover completed

Table 1
Meanings of Messages in Figure 5

Message	Meaning
RBCComd	RBC handover command
Req&Train_Infor	Route requesting information
Call	Communication initializing infromation
AnswerCall	Communication version information
CommBuilt	Communication built infromation
AnswerRBC	Route information
HybridMA	Hybrid MA on the border
Train_position	The position of the train
arriveRN	Front end of the train arrives at the border
Ann	Enter into the region of the RBC2
Infor	Takeover information
MA	Movement authority
passRN	Rear end of the train passed the border
Close_Comm	Terminate the communication

Table 2
Meanings of Variables in the Program

Variable	Meaning
MessageTrain	Messages the train received
MessageRBC1	Messages RBC1 received
MessageRBC2	Messages RBC2 received
level	Control level of the train
SupervisedByRBC1	The train is supervised by RBC1
SupervisedByRBC2	The train is supervised by RBC2
CommBuilt	Communication with RBC2 is builted
FinishHandover	The process of RBC handover is completed successfully

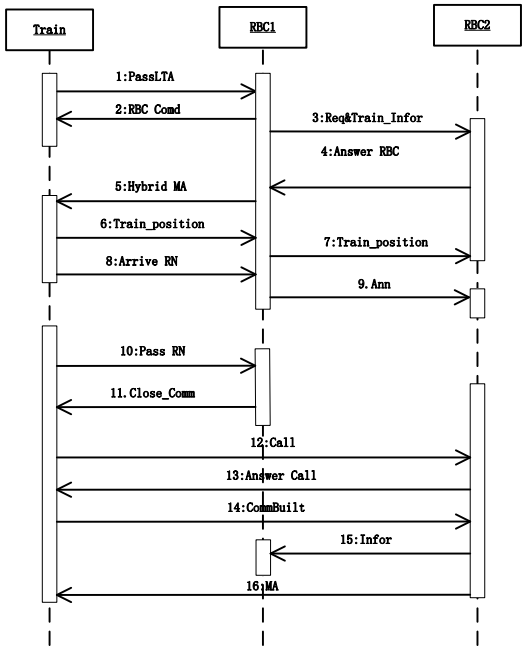

Fig. 5. The Sequence Diagram of RBC Handover

Fig. 6. Simulation Result

successfully; 2. there is no MA in the region of RBC2 and the train control level changes to level 2 finally.

5 Verification

The verification process of the properties specified in the system requirement specification is given in this section. Before verifying the properties, we need to specify them by PPTL formulas. Then the verification can be done automatically by means

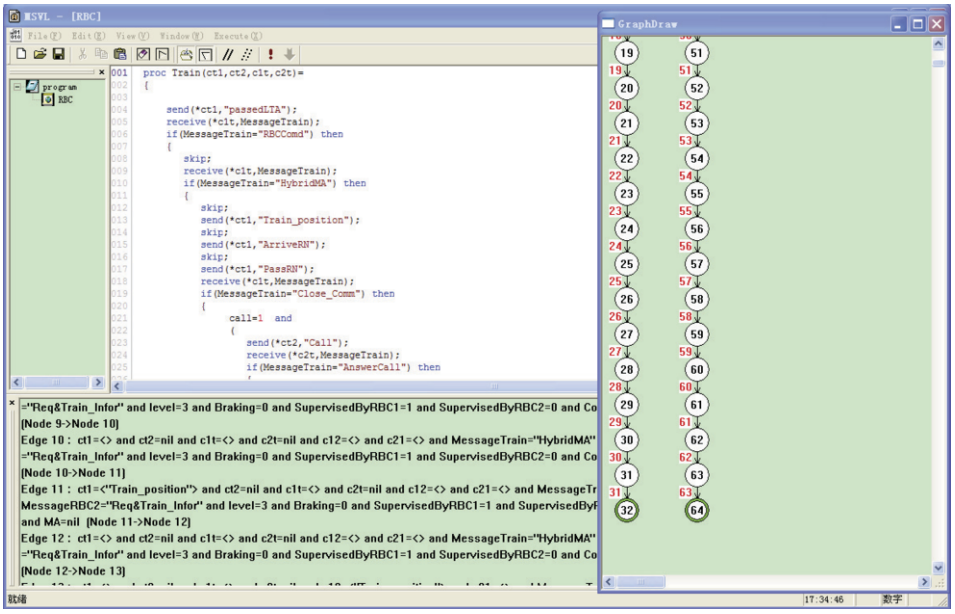

Fig. 7. Modeling Result

of model checking with the MSV toolkit. Here, we only verify three properties.

Property 1: Finally, the procedure of RBC handover completes successfully, or the control level of the train changes to level 2.

define p: FinishHandover=1;
define q: level=2;
fin(p or q)

Property 2: The train receives message "Close_Comm" from RBC1 after sending the message "PassRN" to RBC1.

define p: MessageRBC1="PassRN";
define q: MessageTrain="Close_Comm";
p and next(true);q

The verification results show that the above 2 properties are satisfied but only the verification result of Property 1 is shown in figure 8.

Now, we verify the 3rd property. The result is shown in figure 9. As we can see from the result, property 3 is not satisfied. The result means that the train neither under the control of RBC1 nor RBC2 during a period of time in the handover process. After an analysis of the requirement specification, we find that when the train receives the message "Close_Comm" from RBC1, the communication with RBC1 will be disconnected (time 1). Then the train starts to call RBC2 until the communication with RBC2 is built (time 2). Between time 1 and time 2 the train is neither supervised by RBC1 nor RBC2. Hence, the verification result of property 3 is consistent with the requirement specification.

Property 3: The train is always under the control of RBC1 or RBC2.

define p: SupervisedByRBC1=1;

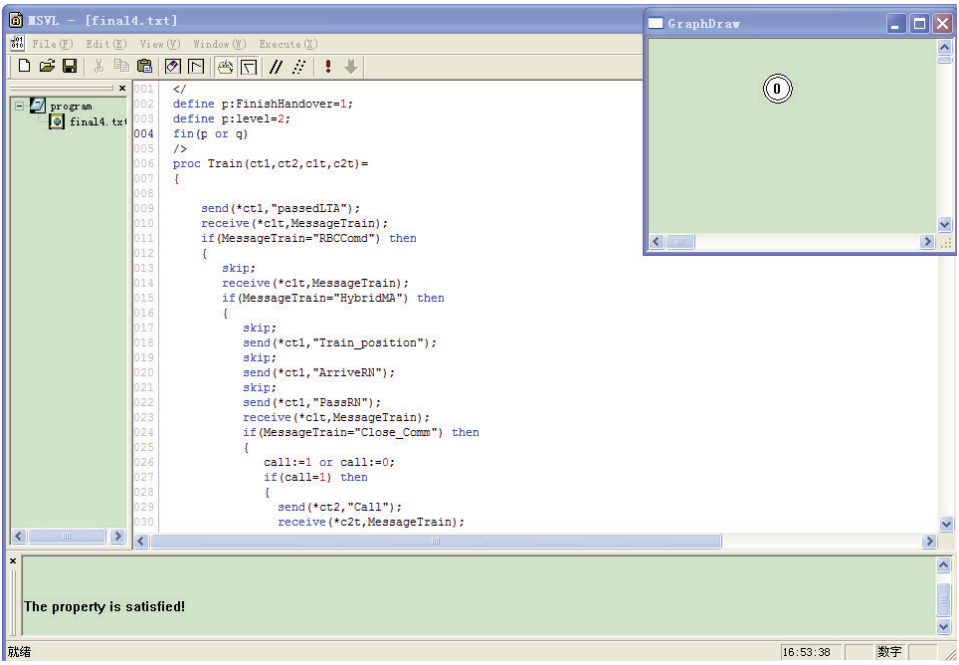

Fig. 8. Verification Result of Property 1

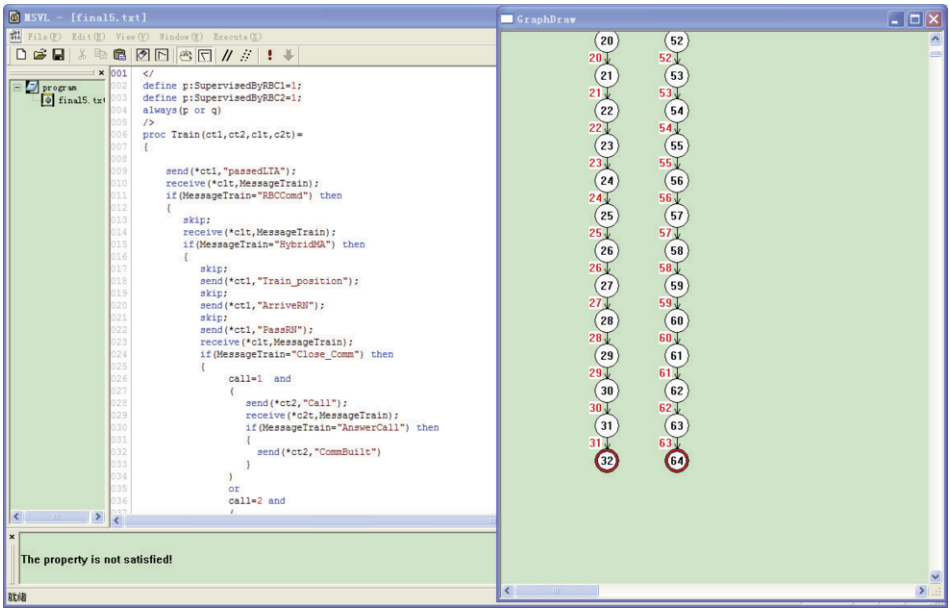

Fig. 9. Verification Result of Property 3

define q: SupervisedByRBC2=1;
always(p or q)

6 Conclusion

In this paper, we analyse the RBC handover protocol for CTCS-3. The UML sequence diagram, a semi-formal description of the handover process, is given according to the requirement specification. Then, we translate the semi-formal description to an MSVL program. In this way, not only the simulation and modeling processes can be done with the MSV toolkit but also the more important process, i.e., the verification of the protocol, can be conducted. However, the translation from the UML sequence diagram to the MSVL program is done manually. Therefore, it is difficult to ensure consistency between the two formalisms. To improve this, an automated translation method should be studied. Further, qualitative time in the model is also needed to take into account in the future.

References

[1] Beato, M. E., M. Barrio-Solórzano, C. E. Cuesta and P. de la Fuente, *Uml automatic verification tool with formal methods*, Electronic Notes in Theoretical Computer Science **127** (2005), pp. 3–16.

[2] Chiappini, A., A. Cimatti, C. Porzia, G. Rotondo, R. Sebastiani, P. Traverso and A. Villafiorita, *Formal specification and development of a safety-critical train management system*, in: *Computer Safety, Reliability and Security* (1999), pp. 410–419.

[3] Duan, Z., "An extended interval temporal logic and a framing technique for temporal logic programming." Ph.D. thesis, University of Newcastle upon Tyne (1996).

[4] Duan, Z., "Temporal logic and temporal logic programming," Science Press, 2005.

[5] Duan, Z., M. Koutny and C. Holt, *Projection in temporal logic programming*, in: *Logic Programming and Automated Reasoning*, Springer, 1994, pp. 333–344, (A technical report No. 452, University of Newcastle Upon Type, 10, 1993, is available).

[6] Duan, Z. and C. Tian, *A unified model checking approach with projection temporal logic*, in: *Formal Methods and Software Engineering* (2008), pp. 167–186.

[7] Duan, Z., X. Yang and M. Koutny, *Semantics of framed temporal logic programs*, in: *Logic Programming* (2005), pp. 356–370.

[8] Duan, Z., X. Yang and M. Koutny, *Framed temporal logic programming*, Science of Computer Programming **70** (2008), pp. 31–61.

[9] Duan, Z., N. Zhang and M. Koutny, *A complete proof system for propositional projection temporal logic*, Theoretical Computer Science **497** (2013), pp. 84 – 107.

[10] Duan, Z.-H. and M. Koutny, *A framed temporal logic programming language*, Journal of Computer Science and Technology **19** (2004), pp. 341–351.

[11] Faber, J., S. Jacobs and V. Sofronie-Stokkermans, *Verifying csp-oz-dc specifications with complex data types and timing parameters*, in: *Integrated Formal Methods*, Springer, 2007, pp. 233–252.

[12] Liu, Y., T. Tang, J. Liu, L. Zhao and T. Xu, *Formal modeling and verification of rbc handover of etcs using differential dynamic logic*, in: *Autonomous Decentralized Systems (ISADS), 2011 10th International Symposium on*, IEEE, 2011, pp. 67–72.

[13] Lv, J. and T. Tang, *Formal modeling and analysis of rbc subsystem in ctcs level 3 using uppaal*, in: *CSIE 2009, 2009 WRI World Congress on Computer Science and Information Engineering, March 31 - April 2, 2009, Los Angeles, California, USA*, IEEE Computer Society, 2009, pp. 49–53.

[14] Mo, D., X. Wang and Z. Duan, *Asynchronous communication in msvl*, in: *Formal Methods and Software Engineering* (2011), pp. 82–97.

[15] NIU, R., Y. CAO and T. TANG, *Formal modeling and analysis of rbc handover protocol for etcs level 2 using stochastic petri nets*, Journal of the china railway society **4** (2009), pp. 52–58.

[16] Tang, W., B. Ning, T. Xu and L. Zhao, *Scenario-based modeling and verification for ctcs-3 system requirement specification*, in: *Computer Engineering and Technology (ICCET), 2010 2nd International Conference on*, IEEE, 2010, pp. 400–403.

[17] Zhang, N., Z. Duan, C. Tian and D. Du, *A formal proof of the deadline driven scheduler in pptl axiomatic system*, Theoretical Computer Science (2013).
URL http://dx.doi.org/10.1016/j.tcs.2013.12.014