A MDE-Based Approach to the Safety Verification of Extended SysML Activity Diagram

Chuanlin Huang[1*], Zhiqiu Huang[1], Jun Hu[1], Zhipeng Wu[1], Siqi Wang[1]
[1] College of Computer Science and Technology, Nanjing University of Aeronautics and Astronautics, Jiangsu, China.

* Corresponding author. Tel.:18609828687; email: huangchuanlin@nuaa.edu.cn
Manuscript submitted February 24, 2014; accepted August 23, 2014.

Abstract: Safety verification of real-time embedded systems is a complex and hot issue. This paper proposes a SysML/MARTE activity diagram (SMAD), which is extended from SysML activity diagram (SAD) with non-functional MARTE semantics, for the describing of the real-time embedded systems behaviors. To carry out the safety verification, we transform the SMAD into timed automata. The processes of the model transformation and formal verification are as follows: first, building the meta-models of SMAD and timed automata, which are based on MDE; second, achieving the semantic and structures mapping, which can complete the model transformation; third, input the CTL specification into model checker UPPAAL for the verification. Finally, we construct an instance to illustrate the validity of the approach.

Key words: Safety verification, SysML activity diagram, MARTE, model transformation.

1. Introduction

There are lots of complex dynamic behaviors in embedded systems，which have been used in the industrial control systems for highlighting the safety of aerospace systems [1], high-speed rail, nuclear power and so on. In these areas, higher safety requirements should be guaranteed [2]. Once the software goes wrong, it may bring immeasurable losses to life and property. As software system becomes increasingly complex and the scale is getting bigger, how to design real-time embedded systems, which have high quality, reliability and can be verified, is a hotspot issue in academia and industry.

Nowadays, UML (United Modeling Language) [3] has been the recognized industry standard modeling language and has been widely used, but modeling real-time embedded systems using UML have some difficulties, such as the lack of consistency [4], poor interoperability and poor modeling ability for system projects. Modeling dynamic behaviors using UML activity diagram (AD) also have some problems, such as, difficulties in safety verification.

SysML (Systems Modeling Language) [5], [6] is standard modeling language in system engineering application, supporting analysis, design, verification and validation for a complex system in detail. MARTE (Modeling and Analysis of Real-Time Embedded Systems) [7] is standard modeling language for real-time embedded systems, providing expression syntax for time and algebra, and supporting the non-functional attributes modeling. We should transfer these UML or SysML models into other formal models.

Traditionally, the transformation is the format of ad-hoc [8] which builds the special transformation for special models. This transformation has questions of semantic and syntax mingling with each other and needs to be rebuilt when adding new elements. For example, Bernardi, S. [9] proposes the transformation between UML sequence diagram and Petri net. Nianhua Yang [10] proposes the transformation between the UML model marked MARTE's information and colored Petri net. Yu Zhou [11] proposes a transformation between UML state machine diagrams along with the time-related modeling elements of MARTE and extended hierarchical timed automata [12], [13]. Jagadish Suryadevara and Yin Ling [14] propose the transformation between CCSL (Clock Constraint Specification Language) of MARTE and timed automata. Jagadish Suryadevara and Cristina Seceleanu [15] propose a technique for transforming MARTE/CCSL mode behaviors into timed automata. These lay particular stress on CCSL. Samir Ouchani [16] proposes a mapping between a composition of SAD and the input language of the probabilistic symbolic model checker. Yosr Jarraya [17] proposes a transformation between SAD and discrete-time Markov chains. These lay particular stress on the analysis of probability.

To solve these problems, the model transformation based on MDE is proposed. Frederic Jouault [18] proposes how the ATL achieves the automatic model transformation. Tian Zhang [19] proposes the transformation between MARTE model and FIACRE model using ATL in AMMA; Yaping Liu [20] proposes the method of real-time system transformation based on meta-model [21]. MingJi [22] achieves the transformation between MARTE model and priced timed automata. Lixia Ji [23] achieves the transformation between UML model and timed automata. Xiaopu Huang [24] proposes the transformation between SysML state diagram and timed automata. The above research work segregates the transformation of semantic and syntax, and can reuse the transformation.

However, modeling using UML and interoperability are not robustness, and the description of embedded systems using AD is incomplete. We use SMAD model to describe embedded systems. The reason why we use SMAD is that SysML can describe systems better and SAD extends some properties which can describe systems more completely. Another reason is that introduction of MARTE clock can provide the expression of time which is important in real-time systems. This paper proposes a method transforming the SMAD into timed automata. Because AD is also a kind of special state diagram, activity is also a special activity state and timed automata is based on state, the activities and transitions of SAD have a semantic-mapping relation with the states and edges of timed automata. This paper proposes a method of model transformation and safety verification using the UPPAAL [25]. First, we use the SMAD to describe the behaviors of the real-time embedded systems. Second, we build the meta-models of the SMAD model and timed automata model based on MDE. Third, we formulate the semantic mapping rules between meta-models and give the transformation algorithms of structures. Finally, we input CTL specification into UPPAAL for the safety verification.

The rest of the paper is organized as follows: in Section II, we propose a short overview over SAD, MARTE and the verification framework; in Section III, we build the meta-models of the SMAD model and timed automata model based on MDE, then we formulate the semantic mapping rules between meta-models and give the transformation algorithms of structures; in Section IV, we achieve the safety verification of SMAD using the control system of telephone; in Section V, we have a conclusion of our work.

2. SysML and MARTE

This section describes the SysML and MARTE briefly, puts emphasis on the concepts of SAD and MARTE clock and gives the verification framework.

2.1. SysML and SAD

SysML based on model-driven is the system engineering standard modeling language. SysML supports the description of analysis, design, verification and validation, for hardware, software, personnel and information. It is based on UML2.0, reusing the state machine package and timing package, expanding the activity package and auxiliary package and adding requirement package, parameter package and distribution package [5]. Modeling systems engineering using UML have some questions such as the lack of rigorous semantic, in adequate function description, low reusability and poor interoperability. SysML can remedy these defects.

SAD is an extended from UML AD, including the probability, control as data, control material flow or continuous energy flow[5]. All of these can describe the real-time embedded systems more completely. In the UML, the control can only make the action start while the control can also make the action end in SysML. SysML supports the control operator, which is a logic operator and can generate a control value from input to output. It also supports the limits of the entity flow rate including material, energy, information's continuous stream and discrete streams. SysML extends the object node and class diagram symbol for activity, demonstrates an associated semantics combination among the activities and defines the consistency rules between activity and class diagram. SysML introduces the concept of probability which is used to represent a possibility to leave the decision point or transition be fired. The output parameter sets can also use it to describe the possibility of an output.

AD is also a kind of special state diagrams, activity is also a special activity state, and the transition between activities is not needed to be fired. AD is an important way to describe the system behavior, and SAD can describe the embedded systems that exist a large number of behaviors better: first, using the control as data can control activity's beginning and ending, and control activity in accordance with the direction of reliable security; second, control operator can not only use the precise activity's input and output, thereby improving the controllability of the system activity and safety, but also provide a formal convenient expression for the AD; third, for the actual system activity, various factors influence the activity, so we are not certain that the activity occurs in accordance with the certain order, and the introduction of probability one of safety attributes can improve the description of the activity. However, SAD does not have enough data types and it lacks of the syntax expressions of time and algebra. As a result, it can't model system with sufficient non-functional property [19].

2.2. MARTE and Clock

MARTE is a UML profile published by the OMG (Object Management Group) in 2007, replacing the scheduling performance and timing modeling profile UML-SPT(UML profile for Schedulability, Performance and Time). UML and SysML lack of time and algebraic expression syntax, and there are not enough data types supporting system modeling and other non-functional properties in the non-functional attributes modeling. MARTE are mainly based on three parts (i.e. base, design and analysis), which can be seen in Fig. 1. The base model covers the embedded real-time systems concepts, such as Time, NFP (Non-Functional Properties), GRM (Generic Resource Modeling) and Alloc (Allocation Modeling). Design model is for concurrent and real-time activity on the behavior modeling, such as GCM (Generic Component Model), HLAM (High-Level Application Modeling), SRM (Software Resource Modeling) and HRM (Hardware Resource Modeling). Analysis model can be used to encapsulate analysis system performance and reliability modeling elements, such as GQAM (Generic Quantitative Analysis Modeling), SAM (Schedulability Analysis Modeling) and PAM (Performance Analysis Modeling) [7].

For the real-time embedded systems, the time is essential. This paper uses the time package in the base part. Time structure consists of the time-base, multi-time base, time and structural relationship of time. The basic elements constituting the time structure is time-base, which is a set of ordered points. The channel of time is mainly made up of clock. In MARTE, the clock is an element of a model, and is the channel contacting with time structures. Clock is the element that the most often used to access the time structures and has the ability to bind a specific action or reference to the clock individual, including the Chronometric Time and Logic Time. The clock can be logic or physical, or both. Fig. 2 is Chronometric clock used by this paper.

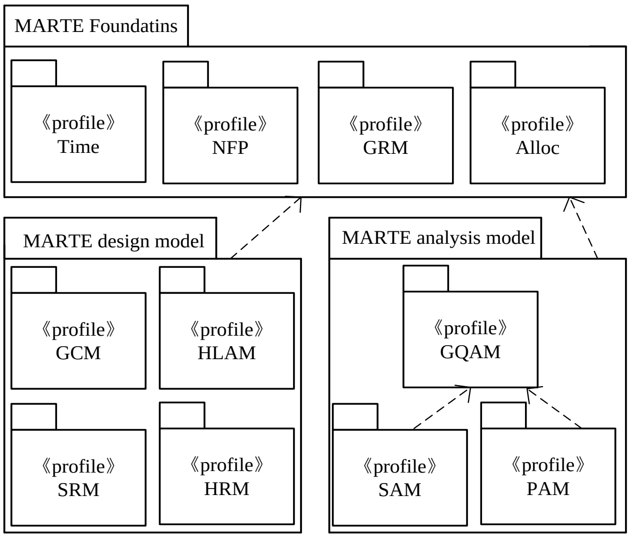


Fig. 1. Structure of MARTE packages.

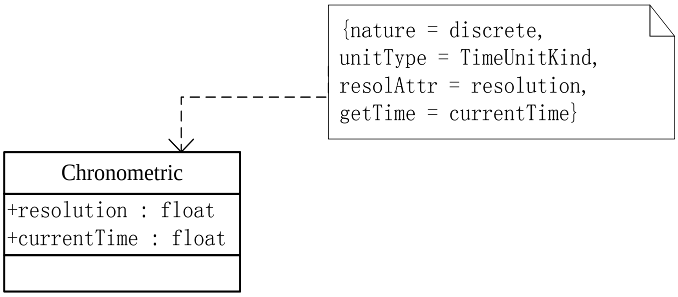


Fig. 2. Clock model.

2.3. The Safety Verification Framework of SAD Combined with MARTE Semantic

From the foregoing information, SAD and MARTE clock can compensate for the defects of modeling the functional descriptions and safety attributes using UML, but the semi-formal SMAD is difficult for safety verification. To solve this problem, this paper proposes a safety verification method transforming SMAD to timed automata. Its framework is shown in Fig. 3.

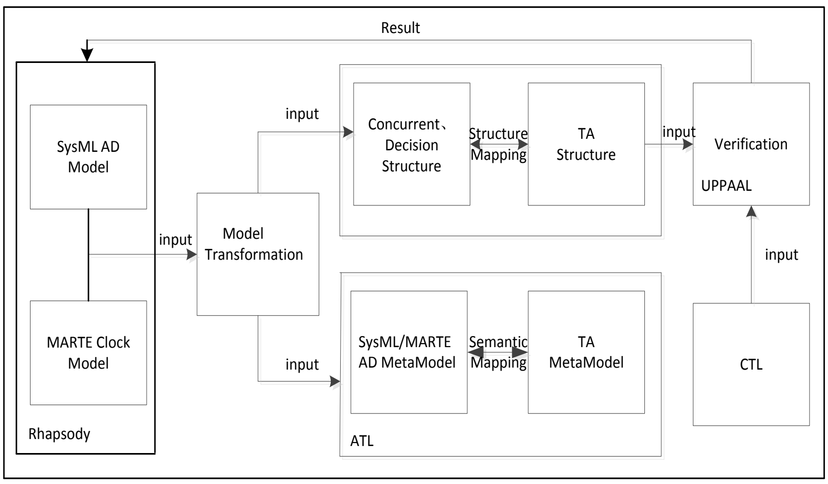


Fig. 3. Safety verification framework of SMAD.

As is shown in Fig. 3, first, we extract the activity behavior, the time constraints, probability, control and other non-functional properties of real-time embedded systems applications and build the model of SMAD using Rhapsody [26].Second, we build meta-models of SMAD and timed automata. Third, we build semantic mapping between them, and construct the structures transformation algorithms. Then, the model of SMAD can be transformed into timed automata model. Finally, after read by UPPAAL, we can have a verification and analysis on the safety using TCTL [27] and PCTL [28]. The results are fed back to the SMAD model. The transformation process is achieved in the AMMA platform. UPPAAL is the model checker, supporting the verification of time and probability [29]. Rhapsody is developed by IBM for the modeling of real-time embedded systems.

3. The Semantic Mapping between SMAD and Timed Automata

In this section, we build the meta-models of SMAD and timed automata, and have a detail analysis for the automatic semantic mapping process between meta-models. Then, we give the transformation algorithms about concurrent and decision structures.

3.1. The Meta-Model of SMAD

According to the definition of the SMAD in the SysML and MARTE standard documents and the behaviors requirements in the real-time embedded systems, we build the meta-model of SMAD, which is shown in the Fig. 4 SMAD consists of Activity Node and Activity Edge which are two major components. There is an Activity Edge between Activity Nodes. When meeting the guard in decision structure, the next activity which is pointed by the Activity Edge will be fired. Each active node can derive from other nodes, such as the Decision Node, the Initial Node and Action. Action which can derive from types of behavior has constraints. In this paper, we use the type of CallO peration Action. There are guard conditions (such as Value Specification) on the decision node which can trigger the transition. The Region covers all behaviors and attributes of activities, so the attribute of the Region is inherited by all activities. This paper puts the concept of clock into the on which is one attribute of Region, so each element of the SMAD can be labeled by the time constraint.

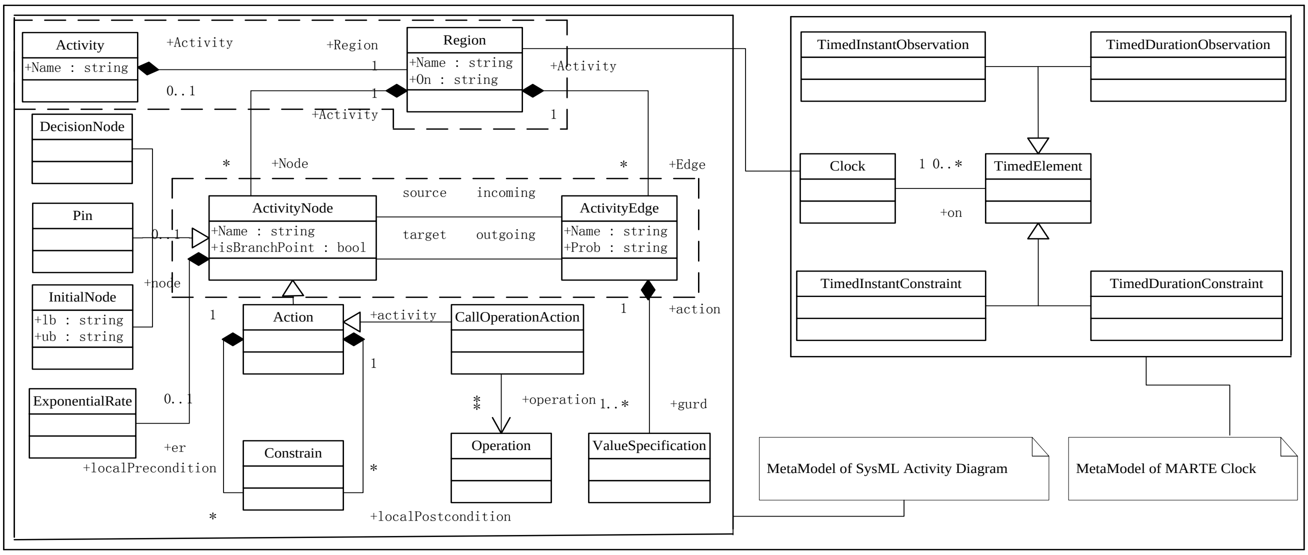


Fig. 4. Meta-model of SMAD.

Control as data, probability and other new concepts are introduced in SMAD. In the UML AD, the control can only make an activity begin, while in SysML, the control can also be performed to make an end of the activity. The control as data is represented by the value of input and output. Pin, one of SAD's elements, is used to represent the type of control. Expression of the transition composed by name and type of Pin, is the input and output of the data. Exponential rate representing exponential probability distribution is a probability expression in SysML is Branch Point is used to indicate whether the state is a probability of a branch. Prob indicates the probability of an activity node which could be happened. The introduction of the probability can improve the description of the activity diagram, and is helpful for the safety verification.

However, SysML does not have the syntax's description of the time and algebra, providing only a simple time which inadequately describes the non-functional properties of embedded real-time systems. MARTE offers a rich way to support the formal expression of non-functional properties of embedded real-time systems, such as Logic time and Chronometric time. Logical time is used to define the number of events. Chronometric time is used to describe the physical time. They can supply clocks for different needs and define the clock constraints to restrain the behaviors of the system [20]. Logic time and Chronometric time solve the problems of how the systems rely more clocks and how the clock restrains the system time. In this paper, we simplify the meta-model of the time and reserve the needed time elements in the modeling of the SMAD. It is shown in the Fig. 4.

Clock represents the access time. Timed Element is an abstract of timing concept and it integrates time element and the set of non-empty clocks together. Timed Install Observati-

On represents the given example and Timed Duration Observation is the given interval of clocks. Timed Instant Constraint binds with the occurrence of events, which is associated with a predicate expressions. Timed Duration Constraint binds with the execution of event, which is associated with a role used in the period expression.

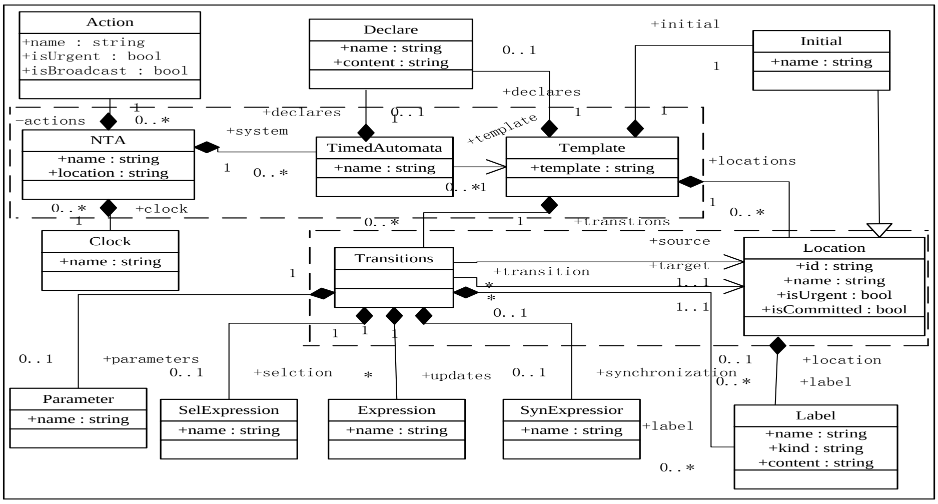


Fig. 5. Meta-model of timed automata.

3.2. The Meta-Model of Timed Automata

A timed automata is a four-tuple <N,I0,E,I>, where N represents a set of finite locations , I0 is the initial location , E represents the set of edges and I is a variant on the location. The core modeling language in UPPAAL is network of timed automata (NTA). NTA is a concurrent combination of timed automata A1, ... An.

In this paper, the model checker is UPPAAL which is based on the timed automata. According to the definition of timed automata and safety requirements, we build the meta-model of timed automata which is shown in Fig. 5. NTA includes timed automata, common Action set, Declare and sets of Clock. Each timed automata has Location and Transition. Transition defines a number of guard conditions including parameter expression, Boolean expressions SelExpression and Label. SelExpression means the control. SynExpressor represents channel associated expression. Location derives from the Initial Location and combines of the Label.

As is shown in Fig. 5, Initial marks the initial position. Label is a clock constraint expression which can be used as the invariant of Location or the constraints of Transition. We add a Boolean expression SelExpression that represents the control. We combine Label for Location and Transition，then use Probability and Exponential rate in the Kind of Label.

3.3. The Semantic Mapping Rules between the Meta-Models of SMAD and Timed Automata

In MDE, the semantic and abstract syntax of the language are defined through meta-model. For example, UML semantics are defined in the meta-model of M1 layer through its MOF. Assuming S is the meta-model of SMAD, T is the meta-model of timed automata and relationship $\oint$ defines a mapping from S to T , then the semantics of timed automata can be represented by relation $\oint$ (S) = T. $\oint$ is defined by a set of the mapping rules (S2T for short). For each mapping rule $\oint$ (s) = t, s means one or more SMAD meta-model and t is a timed automata meta-model.

S2T mapping rules are defined separately from the basic types of structure, behavior, and time constraints. Each aspect contains a set ofS2T rules.

In the basic types of structural mapping, SMAD basic data types include Real, Integer, Boolean, String and Data Time etc, while UPPAAL only supports Integer, String and Boolean. Based on the semantics, Integer, String and Boolean are mapped directly. Data Time is mapped to Integer. Real value before the decimal point is mapped to Integer. The description of S2T rules based on basic data types can be seen in the Table 1.

Table 1. Basic Types of Structure Mapping between SMAD and Timed Automata

$\oint$: Basic Types of Structure Mapping rules
$\oint$ (Integer) |= Integer
$\oint$ (String) |=String
$\oint$ (Boolean) |= Boolean
$\oint$ (DataTime) |= Interger
$\oint$ (Real) |=Interger

In behavioral mapping, an Activity is mapped to a timed automata template. Region is mapped to timed automata. Action has the same semantic with Transition, so they map each. Node of SMAD and Location of timed automata describe state, so they map each. Pin, as the type of SMAD control, is mapped to boolean selection expression SelExpressionintimed automata.isBranchPoint showing weather the active state is a probability branch maps to the Label a branch intimed automata. Pro and Exponentialrate attributes in SMAD indicating the probability of anactive state map to the probability and Exponentialrate in timed automata. In summary, the semantic behavioral mapping rules between the meta-models of SMAD and timed automata are shown in Table 2.

In time constraint mapping rules, TimeInstantConstraint is a boolean expression and is mapped directly to BoolExpression. TimeDurationConstraint represents association of two events and time interval which needs a local clock, and is mapped to the variable of clock in Label. TimedInstantObservation and TimedDurationObservation are mapped to the clock in timed automata and set to different time constraints by observing the different properties. On which is one attribute of Region in SMAD is behalf of the clock model and is mapped directly to the clock in timed automata. In summary, the time constraints mapping rules between the meta-models of MARTE clock and timed automata are shown in Table 3.

Table 2. Behaviors Mapping between Activity Diagram and Timed Automata

∮ : Behavior Mapping Rules
∮ (Activity) |= Template
∮ (Region) |= TA
∮ (Constraint) |= BoolExpression
∮ (ActivityNode) |= Location
∮ (Action) |= Transition
∮ (InitialNode) |= Initial
∮ (CallOperationAction) |=SynExpression
∮ (Operation) |= SynExpression
∮ (Pin) |= SelExpression
∮ (ValueSpecification) |= BoolExpression
∮ (Pro) |= Probability
∮ (Exponentialrate) |= Exponentialrate
∮ (isBranchPoint) |= Label

Table 3. Time Constraints Mapping between MARTE and Timed Automata

∮ : Time Constraints Mapping Rules
∮ (TimedInstantObservation) |= Lable.Clock
∮ (TimedDurationObservation) |= Lable.Clock
∮ (TimedInstantConstraint) |= BoolExpression
∮ (TimedDurationConstraint) |= Lable.Clock
∮ (Region.On) |= Clock

3.4. The Structure Transformation between SMAD and Timed Automata

We have introduced the semantic mapping process between meta-models of SMAD and timed automata. However, the concurrent structure in SMAD can't be expressed by timed automata which have no concurrent structure. Network of time automata can deal with the concurrent structure. Timed automata have no decision structure. As a result, we should transform the SMAD with concurrent and decision structures to timed automata. The concurrent structure of SMAD is shown in Fig. 6 and the decision structure is shown in Fig. 7.

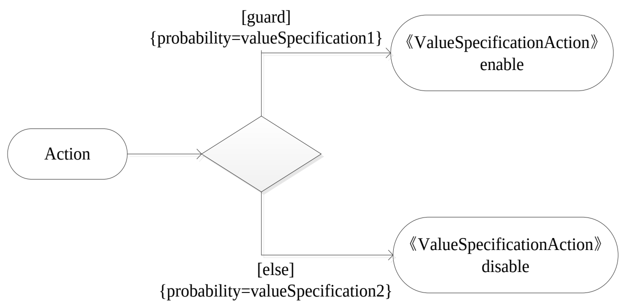


Fig. 6. Concurrent structure of SMAD.

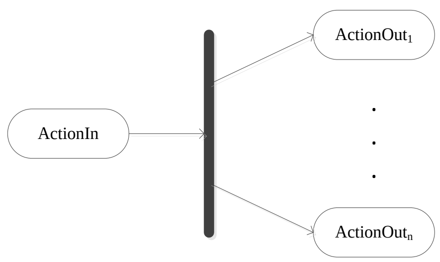


Fig. 7. Decision structure of SMAD.

The transformation process between SMAD's concurrent structure and timed automata is shown in Algorithm1.

```
Algorithm1：The transformation of concurrent structure
Input: SAD = (actionin, actionout, guard, transition, fork, lable).
Output: timed automata = (actionIn, actionOut, actionin, actionout,
clock, go,edge, guard, lable).
Function: transform SAD into NTA
Begin
If actioninis over
add actionIn, actionOut;
add go?   edge from actionOut to actionIn;
If actioninhas signal
   signal is the synchronization;
Else
   go is the synchronization
edge.lable= actionin1==0||…|| actioninn==0;
If actionin has lable
    add lable to actionOut;
End if;
add edge fromactionIn toactionOut;
guard. actionin1==1&&…..&&guard. actioninn==1&&clock==0;
add edge from actionIn to actionOut;
guard.lable=lable;
guard.actionin1==0..guard.actioninn==0,guard.actionout1==11..
guard. Actionoutn==1;
end if
End
```

According to this algorithm, the result of concurrent structure transformation from SMAD to timed automata is shown in Fig. 8.

The transformation process between SMAD's decision structure andtimed automata is shown in Algorithm 2.

```
Algorithm2: The transformation of decision structure
Input: decisionNode=(actionin, actionout, guard, transition,lable).
Output: timed automata=(stateIn,stateOut,edge,guard,lable).
Function: transform the decision structure into timed autoamta
Begin
If guard is meet
Add edge from stateIn to stateOut1;
Edge.guare=transition.lable;
Edge.lable=transiton.lablel;
Edge.action=valueSpecificationAction;
Else
Add edge from stateIn to stateOut2;
Edge.guare=transition.lable;
Edge.lable=transiton.lablel;
Edge.action=valueSpecificationAction;
End
```

According to this algorithm, the result of decision structure transformation from SMAD to timed automata is shown in Fig. 9.

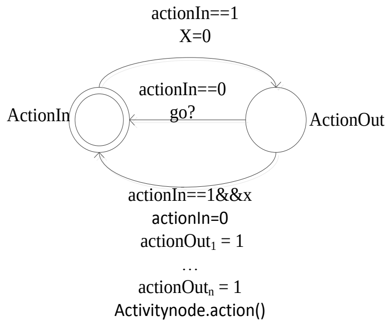


Fig. 8. Concurrent structure in timed automata

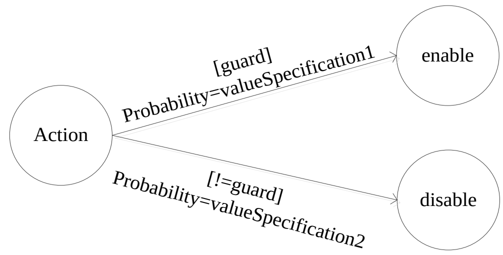


Fig. 9. Decision structure in timed automata.

4. Case Study

In this section, we verify the safety of a telephone control system(TS) using SMAD. Control systems restrict the behavior of the system via some designated control mechanisms, which provide a more generic framework to integrate the automaton and grammar representations of control in supervisory control and regulated rewriting, and is related to system safety issues [30].

4.1. Question

When a user dials a telephone number, the telephone system works as follows: TS will come into the connect activity.In this activity, TS will decide whether the time is beyond 30s.If yes, it will end, else if it receives a conflict, the call is finished too, else if it receives an answer, it will come into counting activity and checking activity. In these activities, the TS counts the calling time and checks the signal. If the signal is lost, the keep activity will keep searching the signal for 10s. If it gets the signal, the call will go on, else the call is finished. In order to guarantee the calling process is stable, every action can be executed when the time is more than 2s.

4.2. Building Model of SMAD

We use the Rhapsody to build the SMAD model for TS. Rhapsody developed by IBM supports the modeling for real-time system's hardware and software. According to the question's description, we build the SMAD of TS which is shown in Fig. 10.

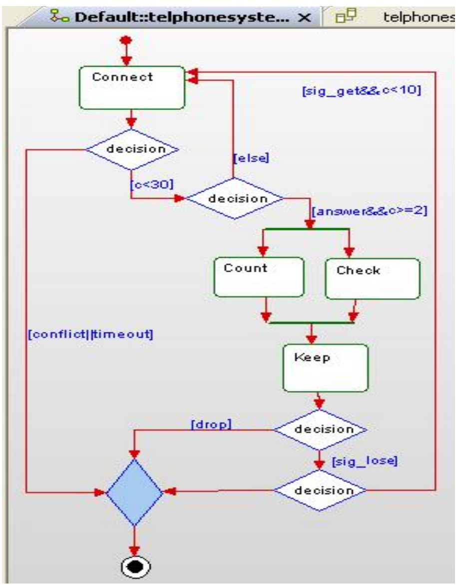


Fig. 10. Clock instance in telephone system.

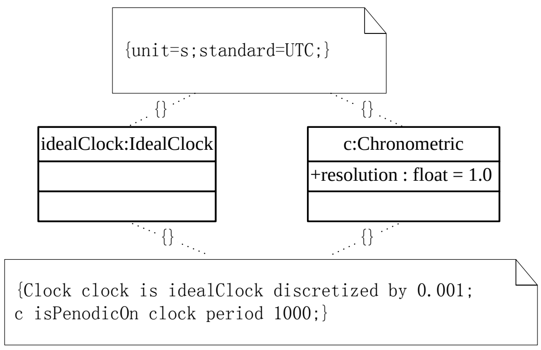


Fig. 11. Activity diagram of telephone system.

The TS SMAD has a concurrent structure and four decision structures. Time is an important attribute in the TS. We should use a clock to count the time when the TS comes into activities of count and check. We have shown the Chronometric Clock in Fig. 2. In this place, the which is one instance of Chronometric is achieved by time constrain language on the ideal Clock which is one instance of IdealClock. IdealClock represents the continuous and physical clock. We get clock by 0.001 discretization of idealClock. Then, we get c by sampling once every 1000 cycles of clock, which is shown in Fig. 11.

4.3. The Transformation Result and Verification

The transformation is divided into two steps: first, we make the semantic transformation on the meta-models in ATL which can map the meta-model elements between SMAD and timed automata; then, according to the transformation algorithms, we transform the concurrent and decision structures in ATL project. The detailed processes can be shown in [18], [19]. The meta-model of SMAD is shown in Fig. 12 and the meta-model can be shown in Fig. 13.

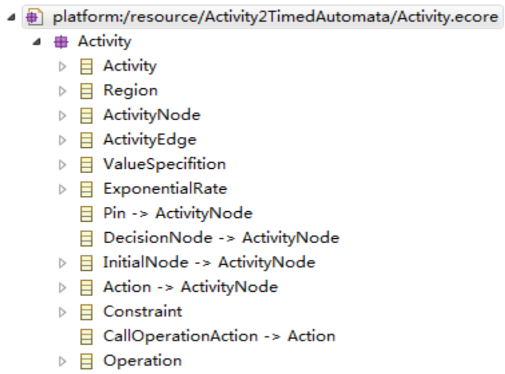


Fig. 12. Meta-model of SMAD in ATL.

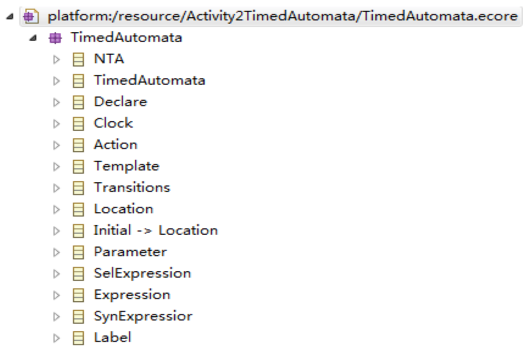


Fig. 13. Meta-model of timed automata in ATL.

After semantic mapping and structures transformation, we transform the TS into NTA, which is consisted of many timed automat as and idle. Each timed automata contains two locations: disabled and firing. The result is shown in Fig. 14.

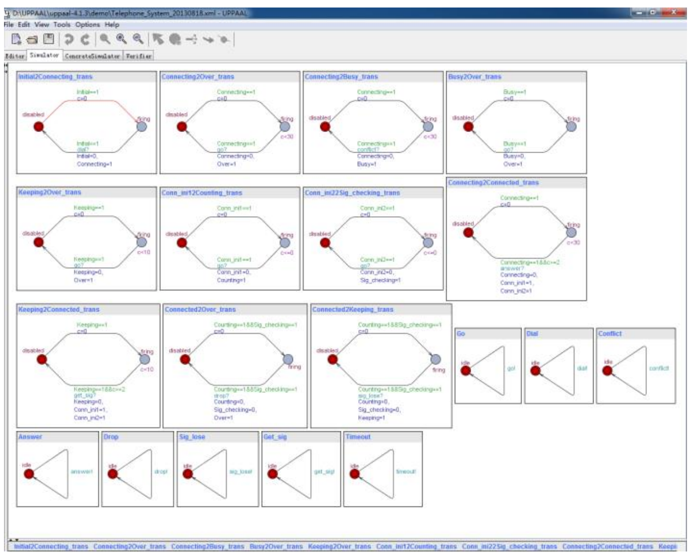


Fig. 14. Telephone system net of timed automata in UPPAAL.

In UPPAAL, we can use the CTL to verify the safety and liveness. The results are shown in Fig. 15.

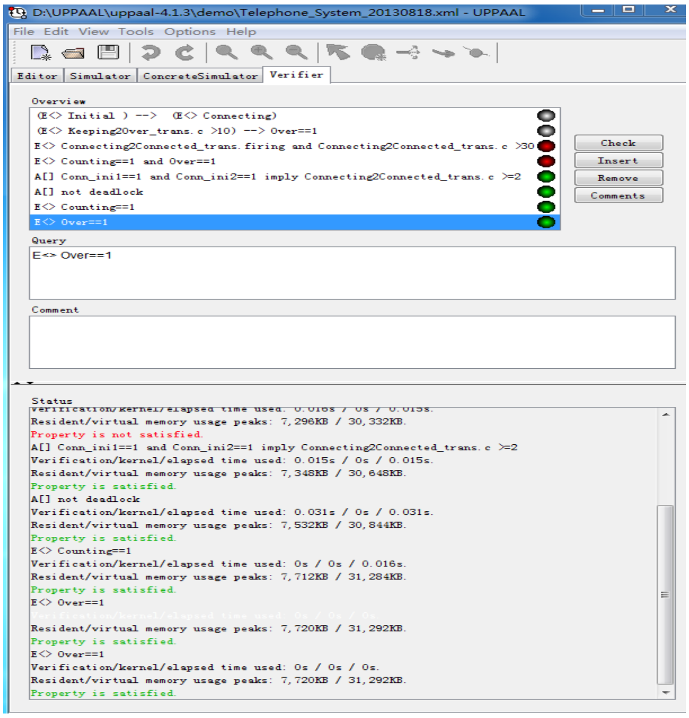


Fig. 15. Result of verification.

The samples of safety and liveness's verification:

①A[] not deadlock

Description: the TS will not be deadlock

Verification result: property satisfies;

②A[] Con_ini1==1 and Con_ini2==1 imply Connecting 2Connected_trans.c>2

Description: the time constructing a connection is more than 2s

Verification result: property satisfies;

③E[] Counting==1 and Over==1

Description result: call is finished but counter goes on

Verification result: property doesn't satisfy;

④E[] Connecting2Connected_trans.firing and Connecting 2Connected_trans.c>30

Description: the call time is more than 30s

Verification result: property doesn't satisfy;

⑤E[] Counting==1

Description: when a connection is connected, counter starts to count

Verification result: property satisfies.

5. Conclusion

Aiming at questions of real-time embedded systems, we use an extended SAD combining with semantic information of MARTE to describe the dynamic behaviors of real-time embedded systems in critical-safety applications. We research on the model transformation and formal safety verification of the system design. We build the meta-models of SMAD and timed automata. Then, we build the semantic mapping rules between meta-models and construct the transformation algorithms of concurrent and decision structures. We design a model transformation and safety verification framework of SMAD, which achieves the transformation and verifies the safety of the SMAD. Compared with the model monitoring [31] which is another popular approach for safety verification, and compared with the ad-hoc which is the traditional transformation, this method is based on MDE and some tools. It is better to be achieved automatically. Compared with the transformation based on MDE, this paper uses the SysML and MARTE to model the embedded systems, which has better interoperability and completeness.

The future work we will do is as follows: the state explosion problem may be a serious problem for checking large systems and we may use the approach proposed in [32]. This paper uses some new attributes of SAD and we will introduce other attributes of SAD's into the design of real-time embedded systems, such as flow.We may introduce the resource and Schedulability into the design of real-time embedded systems.

Acknowledgement

This work was supported by the National Natural Science Foundation of China (Grant No. 61272083 and No. 61170043) and China Postdoctoral Science Foundation of China (Grant No. 20110491411).

Reference

[1] Chen, Z., & Gu, Y. (2014). Model checking aircraft controller software: A case study. *Software-Practice & Experience*. Wiley.

[2] He, H., Zhong, Y., & Zhong, G. (2004, April). Methodlgy for complex real-time embedded systems modeling and design. *Journal of Chinese Computer Systems*, *26(4)*, 716-720,

[3] OMG. Unified modeling language: Super structure v2.0. Retrieved 2005 from, http://www.omg.org/docs/formal/05-07-04.pdf.

[4] Chen, Z., & Motet, G. (2009). A language-theoretic view on guidelines and consistency rules of UML. *Proceedings of the 5th European Conference on Model Driven Architecture Foundations and Applications (ECMDA-FA 2009), Lecture Notes in Computer Science*, vol. 5562, (pp. 66-81). Springer.

[5] OMG. Systems modeling language.v1.2. Retrieved 2008 from http://www.omg.org/spec/SysML/1.2.

[6] Soares, M. S., & Vrancken, J. (2008). Model-driven user requirements specification using SysML. *Journal of Software*, *3(6)*.

[7] OMG. UML profile for MARTE.beta2. Retrieved 2008 from http://www.omg.Or/cgi-bin/doc?ptc/2008-06-08.

[8] Chen, Z., Zhang, D., & Zhu, R. (2013). A review of automated formal verification of ad-hoc routing protocols for wireless sensor networks. *Sensor Letters*, *11(5)*, 752-764.

[9] Bernardi, S., Donatelli, S., & Mersegue, J. (2002). From UML sequence diagrams and state charts to analyzable petri net model. *Proceedings of 3rd International Workshop on Software Performance* (pp. 35-45).

[10] Yang, N. (2010). Modeling and analyzing trustworthy embedded software in model driven architecture. East China University of Science and Technology.

[11] Zhou, Y., Baresi, L., & Rossi, M. (2013). Towards a formal semantic for UML/MARTE state machine based on hierarchical timed automata. *Journal of Computer Science and Technology*, *29(1)*, 188-202,

[12] Knapp, A., Merz, S., & Rauh, C. (2002). Model checking timed UML state machines and collaborations. *Proceedings of the 7th International Symposium on Formal Techniques in Real-Time and Fault-Tolerant Systems* (pp. 395-414).

[13] Chen, Z. (2011). On the generative power of ω-grammars and ω-automata. *Fundamenta Informaticae*,

111(2), 119-145.

[14] Suryadevara, J., & Ling, Y. (2012). Timed automata modeling of CCSL constraints. *First International Workshop on Formal Techniques for Safety-Critical Systems*, 152-156.

[15] Suryadevara, J., Seceleanu, C., & Mallet, F. (2013). Verifying MARTE/CCSL model behaviors using UPPAAL. *Software Engineering and Formal Methods*. Berlin Heidelberg: Spring.

[16] Ouchani, S., Mohamed, O. A., & Debbabi, M. (2014). A formal verification framework for SysML activity diagrams. *Expert Systems with Applications*, *41(6)*, 2713-2728.

[17] Jarraya, Y., Soeanu, A., & Debbabi, M. (2007). Automatic verification and performance analysis of time constrained SysML activity diagrams. *Engineering of Computer-Based Systems*, 515-522.

[18] Jouault, F., Bzivin, J., & Kurtev, I. (2006). TCS: A DSL for the specification of textual concrete syntaxes in model engineering. *Proceedings of the 5th Generative Programming and Component Engineering(GPCE)* (pp. 249-254).

[19] Zhang, T., Jouault, F., & Attiogbe, C. (2009). MDE-based model to fiacre model. *Journal of Software*, *20(2)*, 214-233.

[20] Liu, Y., Huang, Z., & Zhu, Y. (2010). Research on modeling transformation method of real-time system based on meta modeling. *Journal of Chinese Computer Systems*, *31(11)*, 2145-2153.

[21] Chen, Z., & Motet, G. (2009). Modeling system safety requirements using input/output constraint meta-automata. *Proceedings of the 4th International Conference on Systems (ICONS 2009)* (pp. 228-233). IEEE Computer Society.

[22] Ji, M., Huang, Z., & Zhu, Y. (2011). MDA-based method on resource modeling and model transformation of real-time software. *Computer Science*, *38(8)*, 137-141.

[23] Ji, L., & Ma, J. (2013). Research on model transformation and model checking of UML based on timed automata. *Journal of Zhengzhou University*, 45(1).

[24] Huang, X., Sun, Q., & Li., J. (2013). MDE based verification of SysML state machine diagram by UPPAAL. *Trustworthy Computing and Services*, Berlin Heidelberg: Springer.

[25] Behrmann, G., David, A., & Larsen, K. G. (2004). A tutorial on UPPAAL. *Proceedings of the 4th International School on Formal Methods for the Design of Computer, Communication, and Software Systems (SFMRT'04)*, pp. 200-236.

[26] Gery, E., Harel, D., & Palachi, E. (2002). Rhapsody: A complete life-cycle Model-based development system. Berlin Heidelberg: Integrated Formal Methods,

[27] Liu, H. (2012). Real-time analysis and verification of real-time cyber physical systems. Guangdong University of Technology.

[28] Jurdzi´nski, M., Laroussinie F., and Sproston. J. (2007). Model checking probabilistic timed automata with one or two clocks. *Tools and Algorithms for the Construction and Analysis of Systems*. Springer Berlin Heidelberg.

[29] Pronk, C. (2012). Model checking, the technoloy and the To-os[C]. *2012 International Conference on System Engineering and Technology*.

[30] Chen, Z., & Motet, G. (2009). System safety requirements as control structures. *Proceedings of the 33rd Annual IEEE International Computer Software and Applications Conference (COMPSAC 2009)* (pp. 324-331). IEEE Computer Society.

[31] Chen, Z, & Motet, G. (2010). Towards better support for the evolution of safety requirements via the model monitoring approach. *Proceedings of the ACM/IEEE 32nd International Conference on Software Engineering (ICSE 2010)* (pp. 219-222).

[32] Chen, Z., & Motet, G. (2010). Nevertrace claims for model checking. *Proceedings of the 17th International SPIN Workshop on Model Checking of Software (SPIN 2010), Lecture Notes in Computer*

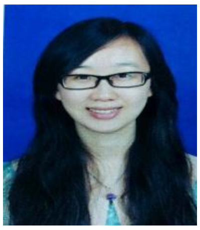
Science, vol. 6349, pp. 162-179. Springer

Chuanlin Huang was born in 1988 and he received the B.S. degree in computer science and technology. Now he is a master candidate at College of Computer Science and Technology, Nanjing University of Aeronautics and Astronautics, Jiangsu, China. His research interests include software engineering, formal methods.

Zhiqiu Huang was born in 1965. He received his Ph.D. degree in computer science from Nanjing University of Aeronautics and Astronautics in 1999. Now he is a professor and Ph.D. supervisor at College of Computer Science and Technology, Nanjing University of Aeronautics and Astronautics. His research interests include software engineering, formal methods, cloud computing and privacy.

Jun Hu was born in 1973. He received his Ph.D. degree in computer science from Nanjing University. Now he is an associate professor at College of Computer Science and Technology, Nanjing University of Aeronautics and Astronautics. His research interests include software engineering, formal methods.

Zhipeng Wu was born in 1989 and received the B.S. degree in computer science and technology. Now he is a master candidate at College of Computer Science and Technology, Nanjing University of Aeronautics and Astronautics, Jiangsu, China. His research interests include software engineering, formal methods.



Siqi Wang was born in 1990 and received the B.S. degree in computer science and technology. Now he is a master candidate at College of Computer Science and Technology, Nanjing University of Aeronautics and Astronautics, Jiangsu, China. His research interests include software engineering, formal methods.