ARCHIVES OF TRANSPORT *ISSN (print):* *0866-9546*
Volume 54, Issue 2, 2020 ***e-ISSN (online):*** *2300-8830*
DOI: *10.5604/01.3001.0014.2730*

FUNCTIONAL SAFETY VERIFICATION OF TRAIN CONTROL PROCEDURE IN TRAIN-CENTRIC CBTC BY COLORED PETRI NET

Junting LIN[1], Qian XU[2]

[1, 2] *Lanzhou Jiaotong University, School of Automation and Electrical Engineering, Lanzhou, China*

Abstract:

Communication-based Train Control (CBTC) system is a widely-used signaling system. There is an increasing demand for innovating the traditional ground-centric architecture. With the application of train-train communication, object control and other advanced techniques, Train-centric CBTC (TcCBTC) system is expected to be the most promising tendency of train control system. The safe tracking interval would be reduced as well as the life-cycle costs. Formal methods play an essential role in the development of safety-critical systems, which provides an early integration of the verifiable design process. In the paper, the architecture design of TcCBTC is first analyzed. The official system specification of TcCBTC has not issued, so it takes efforts to perform the systematic summarization of the functional requirements. Secondly, we propose an integrated framework that combines the Colored Petri Net (CPN) models with the functional safety verification of the underlying systems. Functional safety depends on the logic accuracy and is a part of overall safety. The framework also specifies what kinds of functions, behaviors or properties need to be verified. The train control procedure of TcCBTC is regarded as the link among new functional modules, thus it is chosen as the modelling content. Thirdly, the scenarios and the color sets are prepared. Models are established with the novel design thought from top to bottom. Simulation and testing are implemented during the model establishment to discover the apparent errors. Lastly, the model checking by state space is performed. All possible states are checked in detail. Standard behavioral properties and other user-defined properties are verified by state space report and ASK-CTL (Computation Tree Logic) queries, respectively. Verification results reveal that the models are reasonable to depict the dynamic behaviors of train control procedure. The functional safety properties are satisfied and prepared for further drafting the system functional specification.

Keywords: *functional safety, Train-centric CBTC, train control procedure, formal methods, colored Petri net*

To cite this article:

Lin, J. T., Xu, Q., 2020. Functional safety verification of train control procedure in train-centric CBTC by colored petri net. Archives of Transport, 54(2), 43-58. DOI: https://doi.org/10.5604/01.3001.0014.2730

Contact:
1) linjt@mail.lzjtu.cn [https://orcid.org/0000-0002-5763-5256], 2) 973536969@qq.com [https://orcid.org/0000-0002-3374-4108] – corresponding author

Article is available in open access and licensed under a Creative Commons Attribution 4.0 International (CC BY 4.0)

1. Introduction

Train control systems are the brain of the modern metro signaling system to guarantee the trains operate safely, efficiently and automatically, which are gradually developing from Track-based Train Control (TBTC) to Communication-based Train Control (CBTC). Traditional CBTC system is limited to be ground-centric because the critical subsystems like Zone Controller (ZC) and Computer-based Interlocking (CBI) are located on the ground. However, the complex facilities and interfaces in the ground-centric mode make the life-cycle management costly (Gao, 2018). Furthermore, the indirect information exchanges of the "front train-ground-latter train" mode cause the unexpected transmission delay (Wang et al., 2019; Zhu et al., 2019). Due to the technology trends, especially Long Term Evolution for Metro (LTE-M), CBTC is expected to weaken the proportion of ground facilities and offers trains more information than in the past.

Studies on the Train-centric CBTC (TcCBTC) system represent a growing field. Via the train-centric communication, the project Shift^{2}Rail proposed the "virtually coupled trains" to make the trains operate much closer (Moreno, et al., 2015). "Urbalis Fluence"(2018) offered by Alstom was the first solution for TcCBTC, which was a milestone indicating that the train-centric communication was no longer simply regarded as a redundant backup solution but became an innovative train control system.

Central to the entire field of CBTC is always the concept of safety. Formal model-based approaches to perform safety analysis are well accepted in the development of safety-critical system. Functional safety, information safety and physical safety constitute the overall safety of the current system engineering. The paper is from the perspective of functional safety, which is part of the overall safety depending on functional physical units operating correctly in response to their inputs (EN 50126-1,2017). In the last few decades, there has been an increased emphasis on the formal methods to conduct the qualitative safety analysis of train control systems to locate possible hazards and identify proper precautions. A high-quality model by formal methods can systematically describe the system behavior in a mathematically precise and unambiguous manner (Chen et al., 2017; Wu et al., 2014). The work (Jansen, et al.,1998; Meyer, et al.,2000) introduced how to model the European Train Control System (ETCS) to reflect the architecture of the real system. Besides, the research project "OpenETCS"(2013) was launched to develop a framework integrated modelling, development, validation and testing for leveraging the cost-efficient and reliable implementation of ETCS.

The motivation of adopting colored Petri nets (CPN) as the means of description for this work is as following: CPN is a graphical and mathematical method of descriptions that are suitable in describing systems characterized as being concurrent, asynchronous distributed. With the hierarchical CPN models, it is possible to work with different levels of detail that is beneficial to model large-scale and complex systems. Besides, computer-aided tools are available for editing and verifying CPN models. Additionally, CPN has been applied to the reliability and safety engineering of train control system. Chen et.al (2012) dealt with formal and simulation-based verification of the safety communication protocol. Wu et.al (2018) proposed a top-down approach of scenario-based modelling CPNs to design the onboard subsystem of a satellite-based train control system. Song et.al (2019) verified the functional safety of Train to Train Distance Measurement System (TTDMS) system by establishing CPN models. Previous studies on modelling the ground-centric CBTC provide the feasibility for analyzing TcCBTC. However, there is very little published literature on modelling TcCBTC. More importantly, high-quality system models ought to be provided in terms of the completeness and the representing system properties verification, i.e., state, function, structure and behavior (Hörste, et al., 2013; Cheng et al.,2016). Therefore, how to ensure the quality of the TcCBTC model also faces challenges and deserves further research. The research gaps inspire us to perform CPN model-based functional safety verification for TcCBTC. Movement Authority (MA) is a distance information and allow the train to operate in certain path. The calculation of MA is shifting from the passive mode to the initiative mode. The train control procedure has changed very substantially. Train control procedure is chosen as the modelling content, which has the advantages that the functional modules are not analyzed in isolation with special consideration of the differences.

In the pages that follow, the paper is organized that Section 2 illustrates the system-level functional requirements structures and functional reallocation of

the TcCBTC as well as the introduction of CPN. In Section 3, how to model the train control procedure is present and their accuracy is proven by simulation and testing in modelling process. Formal verification of the established CPN models are implemented in Section 4. We conclude our work in Section 5.

2. Preliminaries

In this section, the architecture of TcCBTC is present. We analyze the demands for the functional modules and information exchanges of the subsystems. Moreover, CPN is the means of description for this work that we give a brief introduction. The framework of CPN models and functional safety verification is also proposed.

2.1. Architecture design of Train-centric CBTC

As depicted in Fig.1, the on-board facilities would be centric to take the responsibilities of ZC, CBI. The functional requirements are interpreted by geography.

(1) Ground facilities

*Intelligent Train Supervision Cen*ter (ITS) issues the operational schedule to trains and supervising the line state by communicating with OCs. ITS plays a similar role as the central Automatic Train Supervision (ATS) in the ground-centric CBTC. Train Manage Centre (TMC) located in ITS is responsible for the maintenance of database version and digital map version. TMC differentiates the train management function of ZC. When the trains are online, they send requests to check the database version. Furthermore, TMC receives the temporary speed restriction (TSR) information from ITS and provides the train with TSR commands in time;

Object Controllers(OCs) are designed to acquire the real-time information of the facilities in the trackside (switch, axle counter, virtual balise, etc.) or the station (Emergency Stop Button (ESB), Platform Screen Door (PSD)). OCs transform the state of trackside facilities to trains and ITS. OCs also execute the corresponding control commands from trains and ITS.

(2) Intelligent Vehicle On-Board Controller (IVOC)

On-board vital computer and the surrounding facilities in the ground-centric CBTC are called Vehicle On-board Controller (VOBC) while it is renamed as Intelligent Vehicle On-board Controller(IVOC) in TcCBTC. On-board Automatic Train Protection (ATP), Automatic Train Operation (ATO) are reserved (Hou, et al.2019). The train integrity, speed control and human-computer interaction are also indispensable. The high-precision train positioning technique is used for achieving the autonomous positioning.

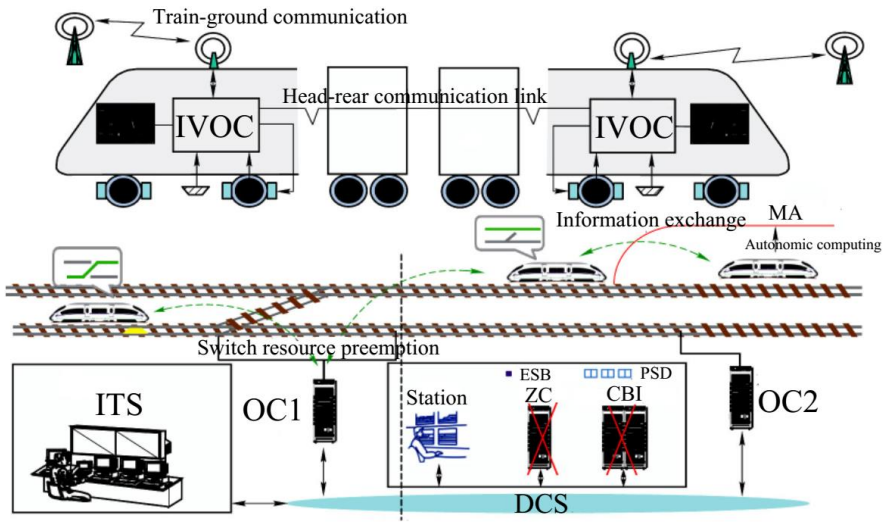

Fig. 1. Architecture design of TcCBTC

Digital map divides the lines into several logic sections. All subjects' locations are defined as the initial point of the logic sections, i.e., geographic coordinates are the addition of the link number and the offset. Three new functional modules are illustrated as follows.

- *Train-Train Communication Management (TTCM)* module is designed to assist trains to interact with ITS and adjacent trains within the communication range and then disconnected if the communication condition is changed. Front train identification is achieved by TTCM.
- Interlocking is one of the core concepts in the railway signalling field, which makes the mutual restraints among the signal controllers, switches and path. Ground-centric interlocking is transferred into on-board *Route Resource Management (RRM)* in TcCBTC. A recent study (Zheng, 2019) proposed that the line resources classification is the static resource (tracks in stations, section without a switch) and movable resources (switch and switch sections).
- The generation place of the Movement Authority (MA) is shifting from ZC to *Movement Authority Calculation Unit (MAU)*. MAU receives the obstacle information from TTCM and the safety path from RRM (Du, et al., 2017; Gao, 2018). As depicted in Fig.2, the functional modules are implemented by obtaining the corresponding information.

2.2. Concepts of colored Petri net

Places drawn as ellipses are used to represent the system state. *Places* are marked with some *tokens* (shown in a small green circle) and each token is attached with a data value called *the token color*. The tokens on a specific place constitute the *marking* of that place. The set of token colors depends on the places' type called *color set. Transitions* drawn as rectangular boxes represent the occurring events. When a transition occurs, it removes the tokens from its input places and adds tokens to its output places. Input *arcs* expression of a transition determines whether the transition is enabled or not and built from variables, constants, operators and functions written by CPN Meta Language (ML). The *binding* refers the variable bound to a value. Each step appears as a pair called a *binding element* consisting of a transition and the occurring binds of the transition. When all variables in the expression have been bound to values of the correct types, the expression ought to be evaluated. Formal definition of CPN can be found in the CPN guidebooks (Jenson et al., 2009).

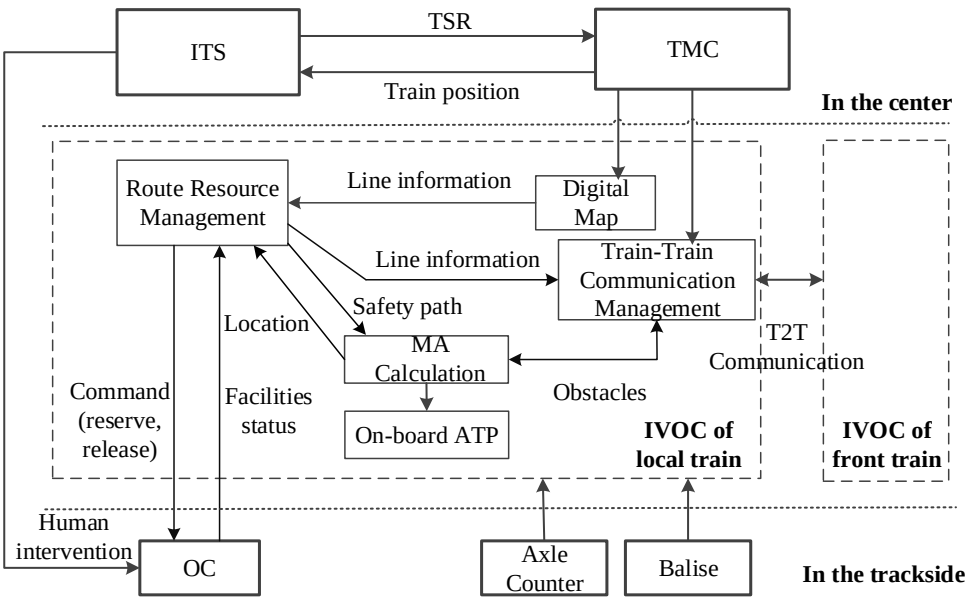

Fig. 2. Information interaction diagram among the subsystem

It is possible to work with different levels of detail and abstraction by specifying hierarchical models, where allow modules to have submodules represented by a substitution transition in its superior hierarchical level. A module exchanges tokens with its environment (i.e., other modules) through interfaces which are port places with rectangular port tags ("In," "Out" or "I/O"). The substitution tag contains the name of a submodule which is related to the substitution transition. Input places of substitution transitions are called input sockets, and the output places are called output sockets. CPN Tools is a tool for editing, simulating and analyzing.

2.3. A framework of CPN models and functional safety verification

The relations of the elements of CPN and the four system properties are illustrated as shown in Fig.3.

(1) *State* of a system is the system situation at a given moment that determines the possible action sequences in the next. In CPN models, states of *places* represent the state of modelled system state.
(2) A *function* is regarded as a specified purpose of the system or its inherent physical state transitions. Complete functions are implemented by sub-functions serving for the overall system. In CPN models, firing specific *transitions* can achieve a function of the modelled system;
(3) The *structure* is composed of a set of components that have close relations with themselves and the environment. In CPN models, System structure is represented by the CPN net structure with *places, transitions and arcs*;
(4) The *behavior* of a system is a set of possible actions and that the system occurs. The set of constraints are also taken into consideration. Behavior means the internal or an external interaction. In CPN models, the system behaviours are a set of possible actions and constraints. Behaviours cover the internal or external interactions specified by markings.

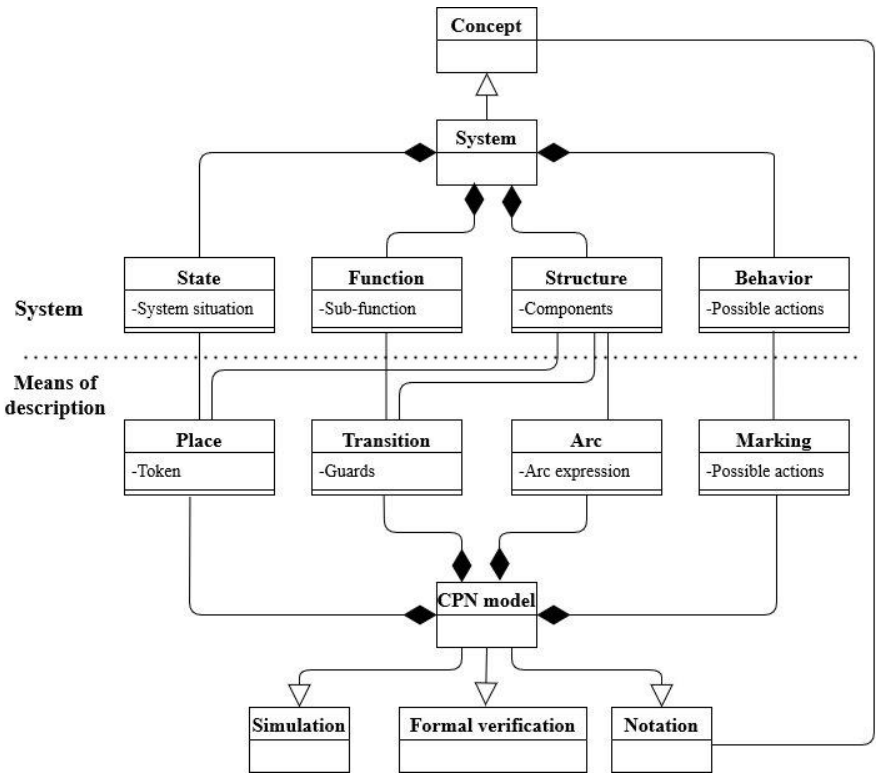

Fig. 3. CPN model and the modelled system

3. Modelling train control procedure with CPN

In this section, CPN models are established. The description of the train control procedure keeps consistent with Section 2.1 and is supposed to be the specific implementation process of IVOC modules.

3.1. Scenario assumption

Fig.4 and Fig.5 illustrate the line topology and trains' for further obstacle description. Taken as a case study, train2 is chosen as the local train, whose planning paths have overlapped area with trains3 or train4. End of Authority (EoA) determines the MA. Train3 would be identified as the front train of train 2. Trains2 and train3 continue to communicate in every tracking interval within communication range. Train4 is conducting the turn-back operation. Switch 1 is in reverse position, which doesn't conform to the former interlocking conditions for train2. Therefore, the EoA of train2 is before entering 1DG. In general, train4 operation range shouldn't reach 1DG. Considering the extreme cases, train4 may cross 1DG and enter 7G, so trains3 and train4 also keep real-time communication.

3.2. Description and models of train control procedure

3.2.1. Top page of CPN models

Before the model construction, the color set and the variables should be declared by keyword "closet" and "var" respectively. Apart from standard color sets, the user-defined color sets are shown in Fig.6. All subjects has the logic sections, i.e., geographic coordinates are the addition of the link number and the offset. *"Link"* and *"OFF"* define the type of link number and the offset. *"ID_T"* and "ID_O" represents Trains' IDs and obstacles' ID, respectively. List of trains' ID is bound to "LIST2_T". Common message is bound to "MSG". Sequence type is "NEXT". "DIR" represents the operation direction. "LOC" is the physical location whose list is bound to "LIST1_T". "STA_O" and "TYPE" is the state and the type of obstacle. "OBS1", "OBS2" and "OBS3" distinguish the different forms of the obstacles. List of "OBS3" is bound to "LIST3". "OC_IVOC" represents the message type from OC to IVOC. "MA" determines the final location authority. Variables types are consistent with places, thus variables are no longer repeated here.

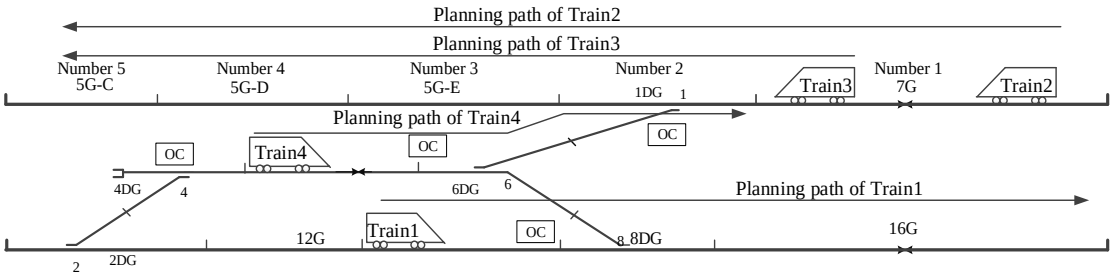

Fig. 4. Line topology describing the trains' original location

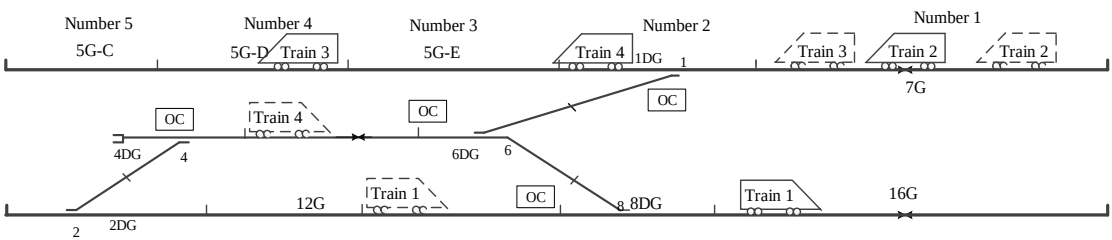

Fig. 5. Line topology describing the trains' location after a while

```
colset LINK = int;
colset OFF=int;
colset ID_T=index T with 1..5;
colset LIST2_T=list ID_T;
colset MSG=string;
colset NEXT=with n;
colset DIR=bool with(pos,neg);
colset ID_O=index O with 1..10;
colset LOC=product ID_T*LINK*OFF*DIR;
colset LIST1_T=list LOC;
colset STA_O=with acc|inacc|exist|notexist;
colset TYPE=with Switch|Section|FrontTrain;
colset OBS1=product ID_O*STA_O;
colset LIST_O=list OBS1;
colset OBS2=product ID_O*LINK*OFF*TYPE;
colset OBS3=product LINK*OFF*TYPE*STA_O*ID_O;
colset LIST3=list OBS3;
colset OC_IVOC=product ID_T*INT*LIST_O;
colset MA=product LINK*OFF;
```

Fig.6. Declarations of the color sets

A hierarchical CPN model of modelling the train control procedure is constructed and the top page is shown in Fig.7. Substitution *Train_sieving*, *Find_front_train*,*Obstacle Traversal* are modelled in detail. *Train_sieving* module describe the train sieving process that the local train checks whether there are the non-communication trains (the train has no location information reported to ITS) or not. *Find_front_train* module serves for front train identification and train-train (T2T) communication. *Obstacle Traversal* determines the MA and then MA produce the operating commands to allow the train to operate in certain path. Other process is similar to the ground-centric CBTC so they aren't unfolded in detail.

3.2.2. Description of Train_sieving module

Sieving is a necessary condition for updating to TcCBTC level and the necessary condition for MA calculation. CPN model of the sieving process is shown in Fig.8. All types of *places* are MSG and the model is interpreted as follows.

Firstly, obtain all "non-sieving" train position (represented by *transition getSieSt_unclear*) and the position of the section boundary point that the train matched (represented by transition *getAxleLoc*).Results can be checked from the "non-sieving clearly" train front end (represented by place *FrontPos_unclear*), "non-sieving clearly" train rear end (represented by place *AxleLoc_Front*), the section boundary point position that the train front and rear end matched (represented by transition *AxleLoc_Front*, *AxleLoc_Rear)*. Place *K1*, *K2* controls the actions

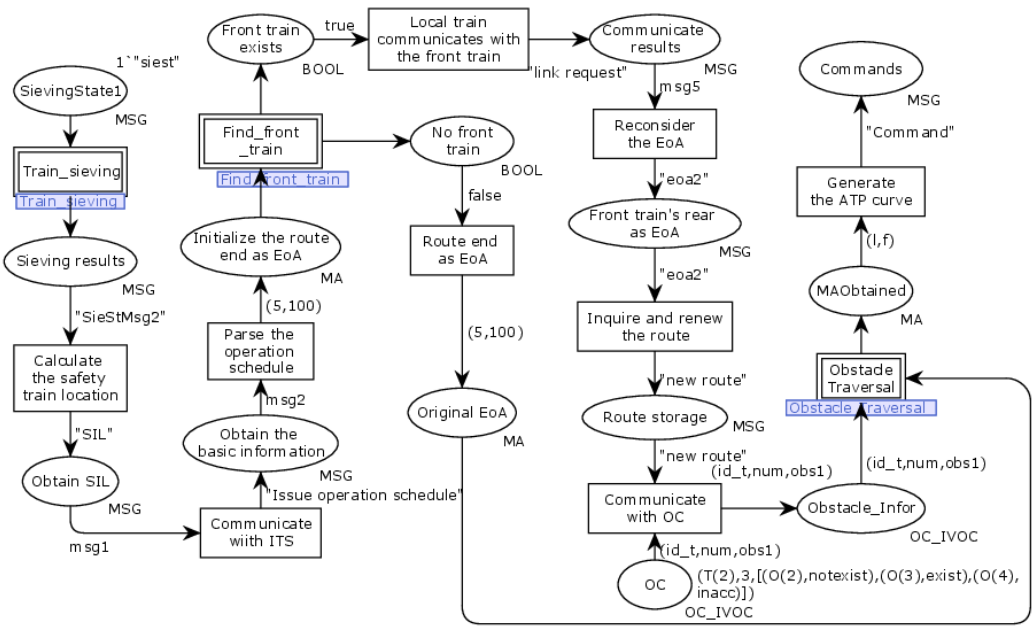

Fig.7. Top page of the train control procedure

can only happen once time. *N* is set to control the occurring sequence, i.e., "sieving the front end" takes precedence. Secondly, the "sieving distance" and the "minimum train length" are compared. Train end is verified by transition *Front Sieving*, *Rear Sieving*. At last, the front and rear end state after the front sieving process (represented by place *SieSt Front* and *SieSt Rear*) are integrated with the pre-sieving process "sieving cleaning" of the end information (represented by transition *SetSieSt*). Sieving status information (represented by place *Sieving Results*) is the output. By simulation, place *SieSt2* has token is *1`"SieStMsg2"* to indicate sieving process was completed. By simulation, place *SieSt2* has token is *1`"SieStMsg2"*, which indicates that the train was successfully sieved and there were no non-communication trains in front and behind.

3.2.3. Description of Find_front_train module

T2T communication is the main feature of TcCBTC. More importantly, the premise of T2T communication is that the front train can be identified correctly and timely. Three possible solutions are taken into account and the relative merits should be weighed to choose the most reasonable solution.

Solution 1: All online trains shall communicate with ITS so that ITS has the whole trains' information. The front train is determined by ITS and the local train keeps communication with a single front train. However, in this mode, ITS is required to cope with much data volume and afford greater safety responsibilities, which is equivalent to traditional ZC and against the principle of train-centric design.

Solution 2: The local train is supposed to communicate with the whole trains on the line to obtain their positions and identify the front train autonomously from them. However, the broadcast storms are prone to occur, i.e., a certain number of communication links cannot be met at the same time in a unit time.

Solution 3: All online trains' information is obtained by ITS as *Solution 1*. The local train communicates with the trains within the lines and determines the front train on its own. ITS, as an "intermediary" for the storage all trains' information, which meets is the concept of "weakening the functions in the ground facilities and realizing the trains' intelligence". Then the local train only keeps the communication with a single front train that would reduce the possibility of broadcast storms.

After the comparision, *Solution 3* is the most practical solution and the model is shown in Fig. 9. Transition *Check_infor* is used to judge whether the complete message is received or not. Then IVOC would combine the section's condition to search sections. Place *Section condition* and *Detecting results* are to model the obstacle detection. If the sections are occupied and trains are judged to be in front, the place *Front train exists* would hold the token; otherwise, the place *No front train* holds.

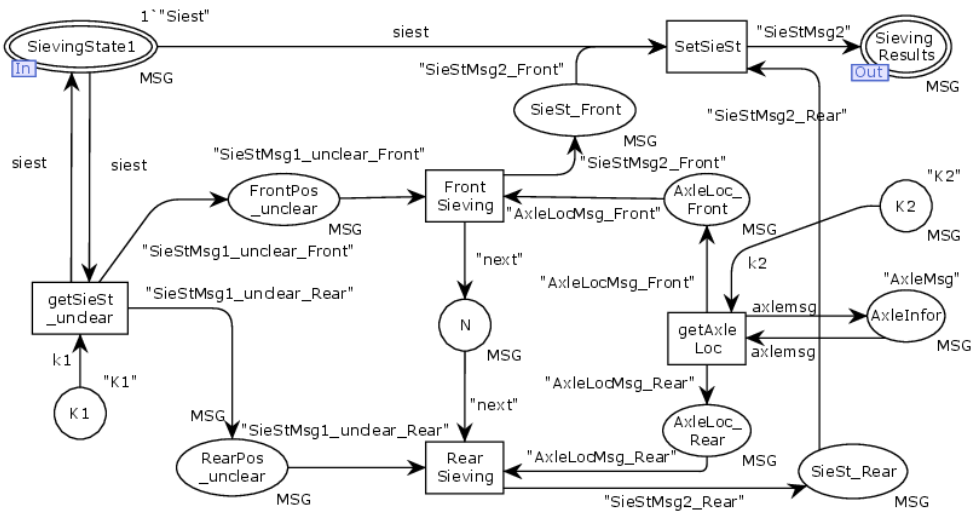

Fig.8. *Train_sieving* module

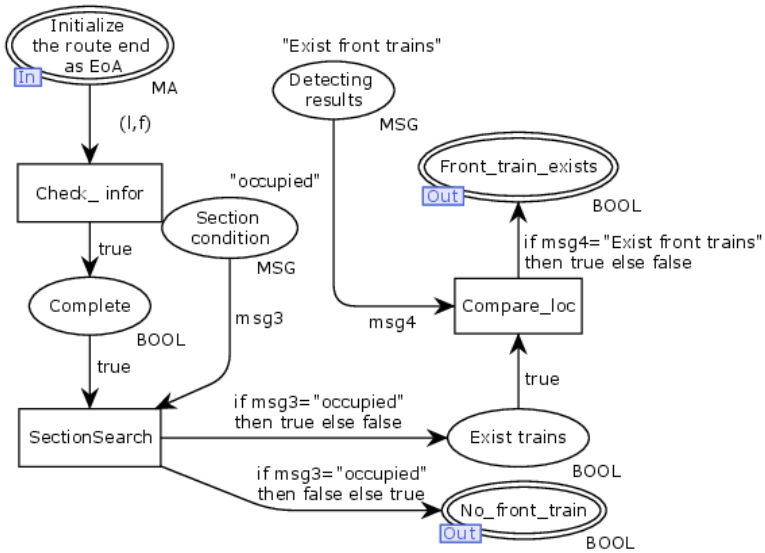

Fig. 9. *Find_front_train* module

3.2.4. Description of Obstacle Traversal module

Afterwards, T2T communication link would be established. Front trains' rear becomes the new EoA. Taken train 2 as an example, the EoA is set to be the rear location of train 4, i.e. (2, 1000). IVOC inquires and renews the route. Top page of *Obstacle Traversal* module are shown in Fig.10. Tokens on place *Obstacle_Infor* represents the obstacle information from OC and is assumed to be static for description despite it is dynamic in reality. Substitutions *Obstacle Process* and *Obstacle Traversal* represent how to process the obstacle information and traverse the obstacles. Word "late" obstacle is used to describe the last one in the obstacle list while "end" obstacle is used to be the EoA.

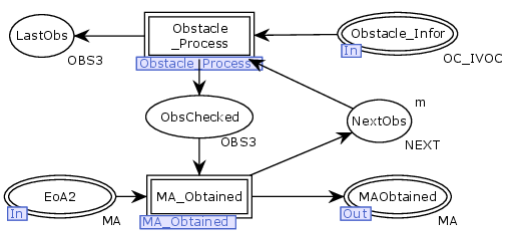

Fig. 10. Top page of *Obstacle Traversal module*

Arc inscription (t1, x, p1) includes the variable *t1* (train ID), *x* (obstacle ID) and *p1* (a list filled with obstacles) shown in Fig.11. Obstacles of train 2 are train 3, train 4, switch 1, represented by *O (2), O (3), O (4)*. Original marking of the *ObsInfor* is set to *(T (2), 4, [(O (2), notexist), (O (3), exist), (O (4), inacc)])*. Then tokens are delivered into the transition *InforDivided*. Then *x* is extracted into the place *OBSNum* and *p1* is extracted into the place *OBSList1* (the first obstacle list). MA is location information but OCs only provide the obstacle states [17]. Obstacle IDs (denoted as *ID_O*) are identifiers to search database. After the first conversion by transition *Conversion1)*, *p1* is turned into *p2*. Guard *"List.take(p1,x)"* means that *p2* is the first x elements of list *p1*. Afterwards, the tokens are delivered into the place *OBSArray* with color set OBS1. *Database* stores the obstacles information and its original marking is set to be *1`(O(2),4,500,FrontTrain)++1`(O(3),2,1000,FrontTrain)++ 1`(O(4),2,0,Switch)*.

By transition *CombinInfor*, the obstacles' information is acquired from the place *Database*. Obstacle list transfers to the second conversion (represented by the *Conversion2*). Then the output arc of the place *OBSList2* is *p4^^[(l,f,w,s,ob1)]* and the input arc is *p4*. Consequently, the obstacle list2 are resorted (represented by the transition *OBSSort)* and results are present on *OBSList3* (the sorted obstacle

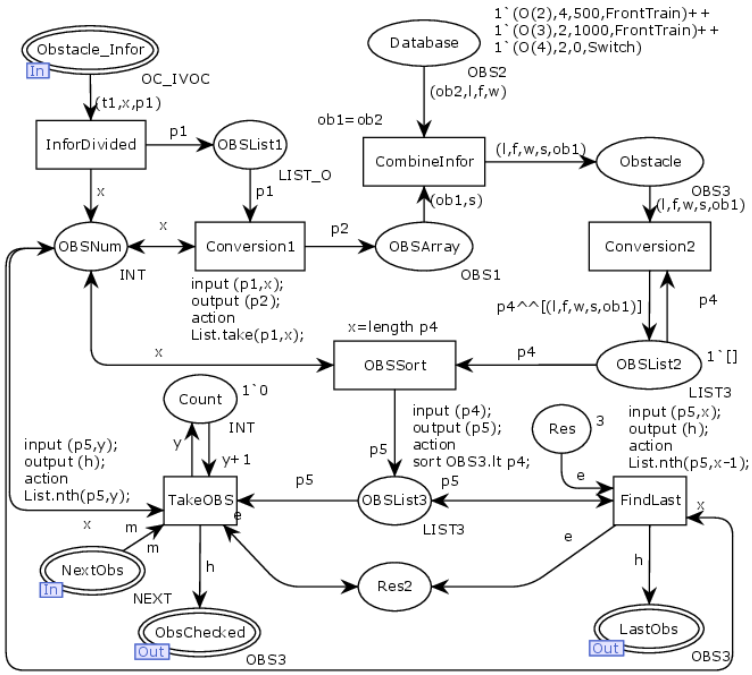

Fig. 11. *Obstacle_Process* module

list). Then select the obstacles to be traversed (transition *TakeOBS).* Through *ObsChecked*, tokens are delivered into the next level. Transition *FindLast* (to select the last obstacle in the list3), and the output is present on place *LastObstacle*.

Via simulation in CPN-Tools, the transmission process of the token are observed shown in Fig.12. Token on the *ObsChecked* is *1` (2,0, Switch, inacc, O (4))*, indicating that the nearest obstacle is *O(4)* and it is a switch with position against the train 2's interlocking requirements. *(2,0)* is 1DG with offset 0. Results are compared with scenarios in Fig.4, it manifests that the model completes the obstacle process by the "from near to far" principle.

The upper part in Fig.13 is to compare the obstacles to be checked (represented by the place *ObsChecked*) with the original MA to determine whether to withdraw or not. The previous MA(represented by the place *Original EoA*) is set to be (5,100). Tokens on the place *ObsChecked* are delivered into the transition *Compare1* to judge whether the MA change or not. If the obstacle to be checked meets the guard [l<>5, f<>100], it implies the MA changed and the place *MAChange's* token would be true. Besides, the token is through the transition Compare1 and delivered into the place *Obs1*.Then the obstacle's state is to judge meet the prerequisites by the RRM module or not. In addition, the judgement is separated in terms of obstacle types. If the token satisfies the guard [w=switch] of the transition *TypeJudge1*, it is proven that the obstacle is the switch, the tokens are delivered into the place *OBS3* and are judged whether the end obstacle or not. The same remark is also on the transition *StaJudge2* and the place *OBS4.* After simulation, the place *MAObtained* has the token 1`(2,0).

4. Formal Verification of the models

The flow of tokens can be observed in the network, which independently test different parts of the model or comprehensively test the model. For safety-critical system models, simulation is suitable for discovering obvious errors rather than proving that the error does not exist by simply within a limited number of steps. Formal verification is designed to ensure that the model is completely correct on the basis of their strict mathematical syntax and semantics (fig.14).

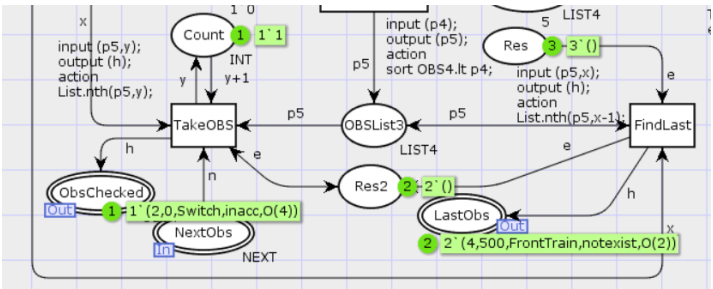

Fig. 12. Simulation results of *Obstacle_Process* module

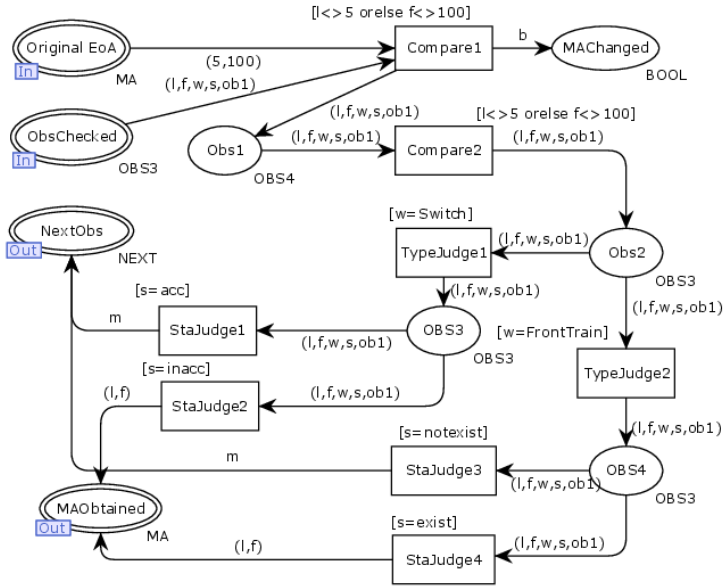

Fig. 13. *MA_Obtained* module

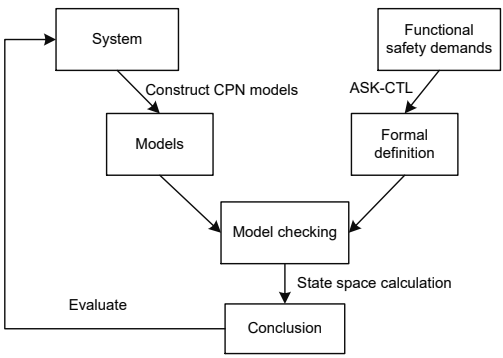

Fig. 14. Schematic diagram of the verification process

The properties to be verified are divided into the following two parts:
(1) Standard behavioral properties make sense for all CPN models, which are verified by state space report. State space report is consisted of statistics, boundedness properties, home properties, liveness properties and fairness properties. Dead marking, deadlock and live-lock should be avoided in a reasonable CPN model. Live-lock is a cycle where unlimited executions produce useless information. In general, the deadlock is due to self-lock terminals, death markings and model unplanned outages.
(2) Other properties of interest can be verified by ASK-CTL kit integrated in CPN-Tools. Computation Tree Logic (CTL) and Linear Temporal

Logic(LTL) are different branches of temporal logic. By shifting the requirements in natural languages into user-defined ASK-CTL queries, the functional safety of the underlying system can be verified. In fact, the standard behavioral properties can also verified by ASK-CTL but it takes more efforts than directly generating state space report.

4.1. Verification of the standard behavioral properties

As shown in Fig.15, statistics information covers the number of nodes and arcs in state space and Scc (Strongly connected component) graph. If two nodes are reachable mutually, they are divided into the same Scc. The state is "Full" with the means of the exhaustive calculation. In our work, the number of nodes and arcs in the state space is consistent with that of Scc, which indicates that there is no self-circle and the system model is considered to be no livelock.

```
Statistics
----------------------------------------
  State Space
     Nodes:   145
     Arcs:    252
     Secs:    0
     Status: Full
  Scc Graph
     Nodes:   145
     Arcs:    252
     Secs:    0
```

Fig. 15. Statistics parts of state space report

Boundedness properties specify the bounds for the number of tokens on each place and information about the possible token colors. Boundedness properties are always satisfied in the models so we skip to describe it in detail. Other properties information are shown in Fig.16. Home properties contains home markings that are marking reached from any reachable markings. In our work, the statement that the information is reasonable because the model is designed to follow the sequential execution rather than return back to initial marking repeatedly.

Fairness properties specify the transition is impartial if it occurs infinitely often in all infinite occurrence sequences. As mentioned, the model in our work has limited steps and ends up with obtaining braking commands, which explain that there are no infinite occurrence sequences.

```
Home Properties
----------------------------------------------------------------
  Home Markings
     None
Liveness Properties
----------------------------------------------------------------
  Dead Markings
     6 [145,144,141,140,135,...]
  Dead Transition Instances
     MA_Obtained'StaJudge1 1
     MA_Obtained'StaJudge3 1
     MA_Obtained'StaJudge4 1
     MA_Obtained'TypeJudge2 1
  Live Transition Instances
     None
Fairness Properties
----------------------------------------------------------------
     No infinite occurrence sequences.
```

Fig. 16. Home properties, liveness properties and fairness properties

Liveness properties provide the information of dead marking, dead transitions and live transition. Dead markings are the marking that the model terminates and there would be no transition can be fired. In our work, the model has six dead markings. Since the state space report doesn't show all dead markings, the ASK-CTL queries and evaluating results are shown in Fig.17. The marking 145 is found as the last nodes so it's the expected terminal marking. Other dead marking are also reasonable because there are separate place that are not integrated into the top page that cause the nodes towards two directions.

4.2 Verification of the specific properties

For our early design of the train control procedure, the properties that need to be verified could be identified based on a hazard analysis. Since the hazard analysis of the system model is out of the scope of this work, we directly present the properties of interest in the train control process model and classify them into the following three categories and the results are supposed to be true. Actually, there are many properties to further explore in the same way as examples here.

(1) Undesired properties are guaranteed to be avoided:

PROPERTY 1 (Fig. 18): Rear train sieving and the complete train sieving cannot take place at the same time;

PROPERTY 2 (Fig. 19): Identification results of the front train and no front train cannot take place at the same time.

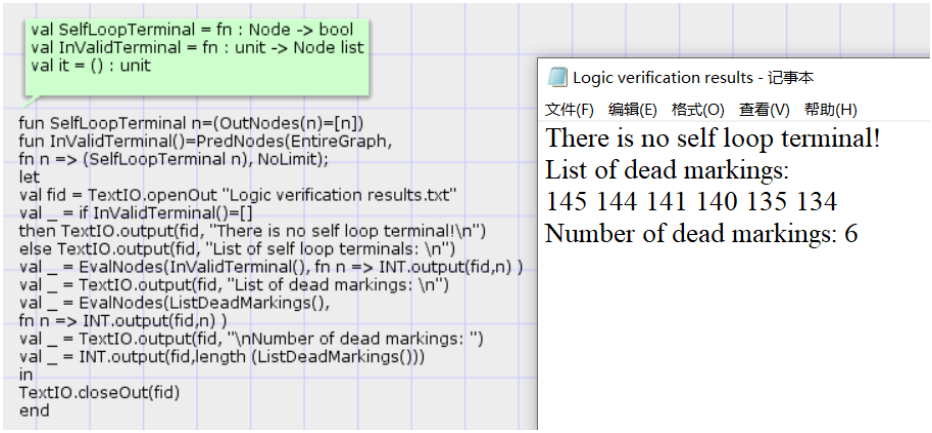

Fig. 17. ASK-CTL queries for checking the dead markings

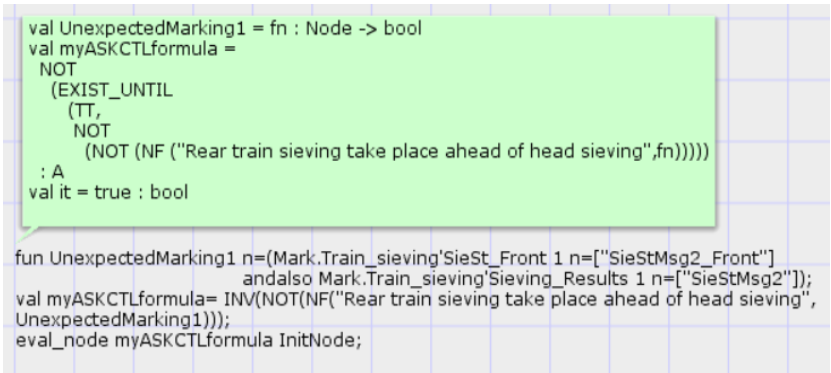

Fig. 18 ASK-CTL queries for PROPERTY 1

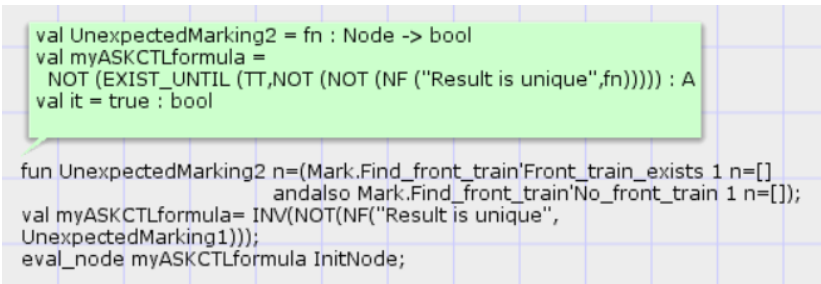

Fig. 19 ASK-CTL queries for PROPERTY 2

Due to the sieving sequence, place *SieSt_Front* and *Sieving_results* cannot hold the corresponding tokens at the same time. Operator *INV* is true if the argument is true for all reachable state, from the state we are at now. *NF* makes only sense to use as a state sub-formula. Its arguments are a string and a function which takes a state space node and returns a boolean. By state formula INV and node function NF, PROPERTY 1 is satisfied from the evaluation results in Fig.18. Likewise, results of PROPERTY 2 is shown in Fig.19.

(2) Desired properties are fulfilled:
PROPERTY 3 (Fig. 21): The last obstacle would be found after the obstacle process.
PROPERTY 4 (Fig. 22): The operation commands would be activated eventually, which means the train control procedure would take place accurately. The obstacle traversal is critical for calculating the final EoA. The input data of obstacles is unordered so it is feasible to reorder the obstacle and traverse the obstacles from near to far. The last obstacle indicates the farthest obstacle.If other obstacle are satisfied the interlocking conditions, the final EoA is the last obstacle. Thus, it is essential to verify whether the obstacle reordered or not. The results shown in Fig.21 are satisfied the design motivation. Likewise, the desirable verification results of PROPERTY 4 is shown in Fig.22.

5. Conclusion

Train-centric CBTC is the most promising tendency of CBTC. A considerable amount of work has been done on functional safety verification by formal methods. We differentiate our work here from the aforementioned literature by present the executable CPN models to abstract the train control procedure of TcCBTC in a systematical approach. Results from the simulation, state-space report and ASK-CTL queries are utilized for functional safety verification. Our work provides four contributions:
(1) Functional requirements of TcCBTC were sorted out for further model construction. Information interaction diagram among subsystems makes the requirements more clearly. The train control procedure is redesigned for TcCBTC. We choose the train control procedure of TcCBTC as the modelling content, which is regarded as a link among those functional modules.
(2) The procedures of model construction and simulation by CPN-Tools. An integrated framework is put forward to explain the relationship system properties and CPN models.
(3) Line topology is given for clarifying the specific obstacle information for trains in different planning path. Train 2 is enough typical for illustration. CPN's hierarchical features determine the feasibility of abstracting from different levels. We follow the classic top-down modelling approach. The top page and the modules distinguishing TcCBTC from CBTC are constructed with the model description.
(4) State space analysis was utilized to prove that there were no design deficiencies of the modelled system instead of simply finding error as simulation. Standard properties are deduced by the state space report. Other properties of interests are verified by

```
val ExpectedMarking1 = fn : Node -> bool
val myASKCTLformula =
  FORALL_UNTIL (TT,NF ("The last obstacle would be found",fn)) : A
val it = true : bool

fun ExpectedMarking1 n=(Mark.Obstacle_Process'LastObs 1 n=[]);
val myASKCTLformula= EV(NF("The last obstacle would be found",
ExpectedMarking));
eval_node myASKCTLformula InitNode;
```

Fig.21 ASK-CTL queries for PROPERTY 3

```
val ExpectedMarking2 = fn : Node -> bool
val myASKCTLformula =
  FORALL_UNTIL
    (TT,NF ("Train control procedure would take place accurately",fn)) : A
val it = true : bool

fun ExpectedMarking2 n=(Mark.TcCBTC'Commands 1 n=["commands"]);
val myASKCTLformula= EV(NF("Train control procedure would take place accurately",
ExpectedMarking));
eval_node myASKCTLformula InitNode;
```

Fig.22 ASK-CTL queries for PROPERTY 4

ASK-CTL queries, i.e., prove that certain desired properties are fulfilled or that certain undesired properties are guaranteed to be avoided. The properties shown in Section 4 are examples and actually there are more properties to further explore in the same way.

As a conclusion, our work aims to solve the lack of functional safety verification of TcCBTC. We construct the executable CPN models and perform detained simulation and verification. The models are reasonable to model the dynamic behaviors of the new train control process. Functional safety properties are satisfied. Using our process to evaluate and verify a system is easier to read and more reliable compared to executable code and mathematical methods. In our future work, more standardized system specifications assist us to modify our models. The code implementation and formal-model based hazard analysis would be carried out.

Acknowledgement

This paper is jointly funded by National Natural Science Foundation of China (No.61963023) and Scientific Research Project of Gansu University of Educational Department (No. 2018A-028).

Reference

[1] CENELEC. EN 50126-1:2017, Railway applications-The specification and demonstration of reliability, availability, maintainability and safety (RAMS) Part 1: Basic requirements and generic process.

[2] CHEN, L.J., TANG, T., et al. 2012.Verification of the safety communication protocol in train control system using colored Petri net. *Reliability Engineering and System Safety*,100:8–18.

[3] CHEN, M.S., BAO, Y.X., &SUN, H.Y., et al., 2017. Survey on formal method of trustworthy construction for communication-based train control systems. *Journal of Software*, 28(5):1183-1203.

[4] CHENG, R., ZHOU, J., &CHEN, D.,2016. Model-based verification method for solving the parameter uncertainty in the train control system. *Reliability Engineering and System Safety*,145:169–82.

[5] DU, H., SUN, G.J., &CHEN, Q., 2017. A new generation CBTC system without CI and ZC. *Urban Rapid Rail Transit*. 30(04):91-95.

[6] GAO, C.H, 2018.*Communication-based Train Control System.* China Railway Publishing House. Beijing,(Chapter 10).

[7] HORSTE, M. Z., HUNGAR, H.,& SCHNIEDER E., 2013, Modelling functionality of train control systems using Petri nets, *Towards a Formal Methods Body of Knowledge for Railway Control and Safety Systems*. Lyngby, Denmark, 46–50.

[8] HOU, T., GUO, Y.Y.,NIU, H.X., 2019,Research on speed control of high-speed train based on multi-point model.Archives of transport. 50(2), 35-46.

[9] INTERNATIONAL UNION OF RAILWAYS. UIC Homepage: OpenETCS, 2013. [checked on 2020-3-10] URL http://www.uic.org/spip.php?article2998.

[10] JANSEN, K., KRISTENSEN L. M. 2009. *Colored Petri Nets: Modelling and Validation of Concurrent Systems*. Springer.

[11] JANSEN, L., MEYER M., & SCHNIEDER E.,1998 Technical issues in modelling the ETCS using colored petri nets and the Design/CPN tools. *Proceedings of the workshop on practical use of colored Petri Nets and Design /CPN*, 103–115. Aarhus and Denmark.

[12] LI K., ZHANG X. S, Design and formal verification of safety communication protocol in STP. *Computer Engineering*, 2018, 44(12): 120-128.

[13] MEYER, M., PTOK, B., et.al. 2000, A case study for the automated system development: the satellite-based train control system. *Proceedings of IFAC Conference on Control Systems Design*,329–334. Bratislava and Slovak Republic.

[14] MORENO, J., RIERA, J. M., & DE HARO, L., et al, 2015. A survey on future railway radio communications services: Challenges and opportunities. *IEEE Communication Magazine*, 53(10), 62–68.

[15] SONG, H.F., SCHNIEDE, E., 2019.Development and Evaluation Procedure of the Train-Centric Communication-Based System. *IEEE Transaction on Vehicular Technology*,68(3).

[16] Urbalis Fluence. 2018. [checked on 2020-3-10].URL https://www.alstom.com/urbalis-cbtc-range-future-signalling-systems.

[17] WANG, X.X., LIU, L.J., et al, 2019. Enhancing Communication-Based Train Control Systems Through Train-to-Train Communications. *IEEE Transactions on Intelligent Transportation System*, 20(4),1544-1561.

[18] WU, D.H., 2014.*Verifiable design of a satellite-based train control system with petri nets* (Ph.D. thesis). Germany: Technische Universität Braunschweig.

[19] WU, D.H., SCHNIEDER, E., 2016, Scenario-based modeling of the on-board of a satellite-based train control system with colored petri net.*IEEE Transaction on Intelligent Transportation System*,17(11), 3045–3061.

[20] ZHENG, W., 2019. *Research on Modelling and Verification of Train-centric Interlocking Control* (Master thesis). China: Beijing Jiaotong University.

[21] ZHU, L., YAO, D.Y., &ZHAO, H.L., Reliability analysis of next-generation CBTC data communication systems. *IEEE Transactions on Vehicular Technology***,** 2019, 68(3), pp. 2024–2034.