Hybrid Online Model-Based Testing for Communication-Based Train Control Systems

Wang, Yuemiao; Chen, Lei; Kirkwood, David; Fu, Peng; lv, jidong; Roberts, Clive

DOI:
10.1109/MITS.2018.2842230

Hybrid Online Model-Based Testing for Communication-Based Train Control Systems

Yuemiao Wang, Lei Chen*, AE, Dave Kirkwood, Peng Fu, Jidong Lv, Clive Roberts

***Abstract*—Communication-Based Train Control (CBTC) systems have been increasingly implemented on metro systems because of their characteristics, which result in safety and capacity improvements for metro operations. Automatic testing methods such as Model-Based Testing (MBT) have been applied to solve some particularly simple and ideal case studies. To bring automation to HIL testing, the authors apply MBT to HIL testing and present a novel hybrid online MBT testing platform that combines formal modelling methods with simulation. The theoretical methodology of the hybrid MBT, the platform architecture, and the testing results produced by the platform are described with a case study of a system under test (SUT) of a real Vehicle On-Board Controller (VOBC).**

Keywords—CBTC; Online testing; HIL testing; Hybrid Model-based testing;

I. INTRODUCTION

A Communication-Based Train Control (CBTC) system consists of three main subsystems, which are the Vehicle-On-Board Controller (VOBC), the Zone Controller (ZC) and the Data Communication System (DCS) [1]. Based on the cooperation between these three parts, the system realises a moving block operation, which provides higher resolution on train location and a higher capacity. On the other hand, the system operation safety largely depends on these subsystems. To ensure the safety of the whole system operation, it is essential to test these three subsystems [2].

Despite the evolution of testing technologies applied in railway systems, testing of the CBTC system is still difficult and time-consuming due to its characteristics of high complexity and non-determinism [3]. Automatic test generation technologies have been introduced into the field to satisfy the growing demand for quicker delivery with higher quality [4]. However, the application of current testing methods on the CBTC systems is usually based on manual test case generation and execution, which makes the testing accuracy, efficiency and expandability stay low [5]. Even worse, traditional testing methods which manually generate and execute the testing cases are not able to deal with non-deterministic systems. To address this problem and also to automate the testing process, Model Based Testing [6] (MBT) has been introduced into the field of railway systematic testing. The authors in [7] present a model based railway signaling system and verify its safety-critical specifications, exploring the possibilities of applying model-based technologies to railway system verification. The authors in [8] introduce automatic test case generation algorithms based on Coloured Petri- Net (CPN). They apply their algorithms on movement authority (MA) handover scenarios (where the MA is transferred from the current ZC to an adjacent ZC when the train is approaching the boundary of the two ZCs) to automatically generate testing cases with an adjustable coverage strategy. The presented research is generally based on assumptive or ideal scenarios, where the SUTs and their operation environment are relatively simple. However, real CBTC systems are usually far more complex. Dealing with the real system complexity, MBT becomes challenging and limited because of the exponentially increasing difficulties in modelling and test case generation from the model [9]. As a result, partition test of the SUTs becomes necessary to decrease the complexity of the system modelling. Hardware-In-the-Loop (HIL) testing is generally a straight forward way to realise this goal. Before all of the related components are developed, HIL testing provides a simulated environment for the SUTs where they can operate normally, as they do in a real working environment [10]. Therefore, the cost and damage risk of the testing can be reduced. With the characteristic of simulation, HIL testing can support various testing scenarios without large expense. Furthermore, HIL testing makes it possible to refine the SUTs into several functions which are easier to implement during testing.

Previous researchers have explored how to combine MBT and HIL testing to take advantage of both methods. J.S. Keranen T.D. Raty, implement MBT technologies in the HIL simulation environment to structure a prototype testing platform based on MBT in HIL testing [3]. The authors in [11] design a software MobiGUITAR to automate the test generation and execution process on mobile APPs with a combination of model-learning and model-based testing technologies. The authors in [12] define a formal definition for testing oracle strategy (guidance for test tools generating test cases with different emphasis) for software MBTs. The authors in [13] introduce Porantim, an approach providing a combination of different MBT techniques for software development. The presented researches focus on the MBT field or the HIL testing field. Some of them try to combine the two together to take both benefits from both sides. However, the embedded application concept presented by previous research limits the testing method application for CBTC system testing because the CBTC system is a hybrid system containing bi-direction communication between the VOBC and the ZC. It is a challenge for previous methods and tools to test such complexly integrated systems because many functions of those systems are related to continuous variables along with discrete commands. Formally modelling such systems is too risky to guarantee the model correctness

Correspond author: Lei Chen, AE

and to avoid model state explosion (the state space expands to infinite size which cannot be processed by computers).

In order to solve this problem, the authors introduce a novel hybrid MBT online testing platform to implement testing on subsystems in CBTC systems by the early development stage. The novelty of this paper is that the authors model the SUT by combining a formal method and simulation. Based on the hybrid model, an online MBT testing tool is applied to automatically generate test cases. The generated test cases are executed with the help of simulation so the continuous and discrete elements are isolated by the simulation and formal method. Different from the previous researchers, who suggest a straight combination of MBT and HIL testing, our hybrid MBT testing platform is compatible with a wider range of SUTs, not only the embedded systems. The design of the testing platform aims to explore whether the hybrid concept is a way to erase the limitation which occurs in the formal testing method application on an industrial level SUT. The paper is structured as follows: we firstly present the concept of conformance testing. Then we introduce the online MBT theory, which is a realisation approach to conformance testing. To break through the challenging exiting in the current online MBT techniques, our novel hybrid online MBT approach is elicited based on the theoretical derivation and combinational application of MBT and simulation techniques. A case study is designed based on a real SUT to evaluate the testing platform we build up. Finally, we draw a conclusion on our novel approach with analysis of the testing platform performance.

II. Methodology: Model-based Online Testing

A. Conformance Testing

Input/output (IO) conformance testing is used to determine whether a conformance relationship is satisfied between the SUT and its requirement specification under a specified testing environment [14]. It is widely applied on railway system testing because of its characteristic which is suitable for black box testing of real timed systems, such as railway systems [15]. Fig 1 shows a general structure of conformance testing utilising MBT testing tools [16].

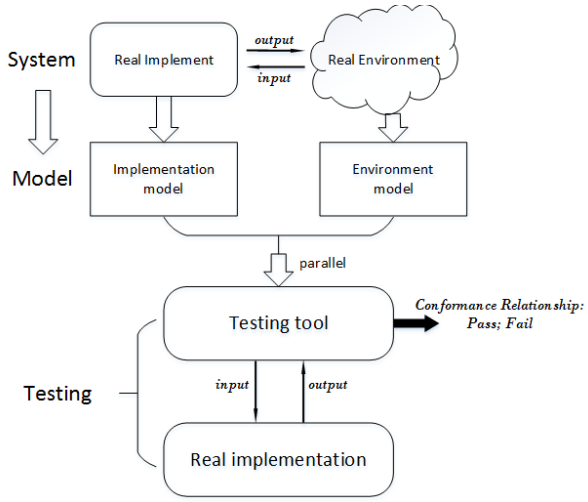

Fig 1 Schematic of conformance testing with MBT

Based on Fig 1, there are three key factors for the realisation of IO conformance testing with MBT, which are the oracle model [17], the MBT testing tool and the IO conformation relationship. In the remaining content of this chapter, these key factors are explained in detail and the complete methodology is presented.

B. Online MBT

The authors choose MBT to realise conformance testing because it provides automation for the testing on real timed systems. MBT testing can be classified into online and off-line testing. Online MBT generates testing cases automatically and executes the generated test case simultaneously, which means there is no complete testing case in online MBT [18, 19]. Due to the online feature, it is more suitable for testing non-deterministic SUTs in a real-time region because one input can be corresponding to several legal outputs[20]. For highly complex SUTs, the formal model size maybe too large to fully cover all possibilities by offline test generation. Online MBT is more applicable when testing such SUTs since the generated inputs are executed immediately without being stored in the computer memory. Different from online MBT, offline MBT firstly generates an entire test case and then executes the generated test case [21, 22]. Every input must correspond to an output in a certain condition, which means it is not eligible for testing non-deterministic SUTs. Due to the feature of offline test generation algorithm, the entire state space needs to be obtained to guarantee the coverage of the generated test cases, which means it can be challenging for offline MBT when testing complex SUTs of which the formal model size is too large for the maximum computational ability of the computer[23]. If the system has a very strong restriction on time, online MBT could become challenging because the simultaneous generation and execution processes may take a relatively long time [3, 24]. Since the SUT does not have very strong time restrictions and it is non-deterministic, online MBT is chosen to be the testing method.

To realise automatic testing by implementing online MBT, the SUT behaviour needs to be formally described so that it can be analysed by computers. Therefore, the modelling theory TIOTS is introduced to formally define the relations between the I/O actions and the state transitions based on the theory of Timed Automata (TA).

1) Timed I/O Transision Systems (TIOTS): The modelling theory

A TIOTS is an evolution of the Labelled Transition System (LTS) under descriptions of time-related constraints. The LTS is designed to present the system behaviour in the format of a state machine. An LTS generally consists of nodes and transitions between nodes. Actions can happen when valid transitions happen. Restraints are defined to specify which node or transition is valid. Definition 1 gives a formal description of an LTS:

Definition 1. An LTS S_B is a 4-tuple $(S, S0, A\tau, Tr)$ where S is a finite and non-empty set of states, and $S0$ is the set of initial states of S satisfying: $S0 \in S$ [25]. $A\tau$ is a set of actions containing observable action set A and internal action set τ, satisfying: $A\tau = A \cup \{\tau\}$; Tr is a set of transition relations that happens in S, satisfying: $Tr \subseteq S \times A\tau \times S$. The Fig 2 presents an example of LTS:

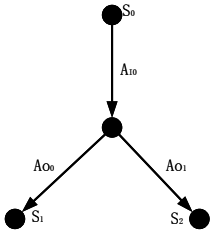

Fig 2 Modelling in the formal of LTS

Derived from classic LTS, I/O transition systems can be seen in Fig 2. Let $A = A_I \cup A_O, A_I \cap A_O = \Phi$. Happened in the system, A_I indicates external input actions and A_O indicates external output actions. With a set of time restraints placed on the system, a Timed I/O transition system (TIOTS) can be obtained.

Definition 2. A TIOTS S_T is a quintuple of $(S, S0, A\tau, Tr, X)$ where the $S, S0, Tr$ has the same meaning as in Definition 1, satisfying: $A_\tau = A \cup \tau = A_I \cup A_O \cup \tau$. X stands for time restraints of the system and it consists of a finite set of clock variables. Assume there exists a finite sequence of observable transitions: $s_0 \xrightarrow{a_0} s_1 \xrightarrow{a_1} s_2 \xrightarrow{a_2} \dots s_{k-1} \xrightarrow{a_{k-1}} s_k$, where $s_k \in S, a_{k-1} \in A$. Then an observable trace σ with timed restraints can be obtained: $\sigma = d_0 \cdot a_0 \cdot d_1 \cdot a_1 \cdot d_2 \cdot a_2 \cdot \dots d_{k-1} \cdot a_{k-1} \cdot d_k \cdot a_k$, where $d_k \in X$. Let s_0 and s_k are the initial state and final state of the assumed trace σ where satisfies: $s_t \overset{\sigma}{\Rightarrow} s_t', s_t \in S, s_t' \in S$, then the observable timed trace from every state can be written: $S_{trace} = \{\sigma \in (A \cup \{R \geq 0\})^*\}$, where * denotes an abstracted transition relationship and $R \geq 0$ denotes a non negative real number [19]. Let divide the S_{trace} into input actions $Input(s_t)$ and output actions $Output(s_t)$ which satisfy: $Input(s_t) = \{a \in A_I | s_t \overset{a}{\Rightarrow}\}$ $Output(s_t) = \{a \in A_O | s_t \overset{a}{\Rightarrow}\}$. Then the external input actions and output actions are defined formally [16, 19].

2) Timed I/O Conformance Relationship

To determine whether the SUT complies with the specification, the I/O conformance relationship needs to be formally defined. In black-box conformance testing, the testing objective is to determine the external I/O action conformance relationship against the specification which means internal actions inside the SUT are usually ignored [26]. Therefore, we extend the existed I/O Conformance Relationship (ioco) to a timed region so that it can be utilised in timed I/O transition systems [27]. The formal definition is shown below:

$$i \textbf{ rtiocoe } s = \forall \sigma \in Str(s, e).\ Out((i, e) \textbf{ AFTER } \sigma) \subseteq Out((s, e) \textbf{ AFTER } \sigma)$$

Where the **rtiocoe** denotes: after executing any possible timed IO trace σ under based the parallel of system specification s and environment specification e, the output generated from the implementation i with the environment e will always be the subset of the output generated from the specification s with environment e [16]. The **AFTER** σ presents all the achievable states specified by the specification environment (s, e) after executing a timed IO trace σ. If this equation holds, the implementation i is considered to conform to the specification s. Therefore, the conformance relationship is defined formally.

C. Hybrid MBT

A CBTC system is a part of a metro system and it is a real timed system with a series of time constraints relevant to safety. In order to ensure the functional correctness of a CBTC system, the functions need to be tested in the whole integrated metro system, which includes the vehicle controlled by the CBTC system and the network where the vehicle runs. We introduce our novel microscopic railway simulator, which can provide the necessary testing environment for the MBT testing. The simulator is written in JAVA at the University of Birmingham and is utilised to build up the kernel of a virtual railway laboratory at the Birmingham Centre for Railway Research and Education (BCRRE) [28]. By microscopic we mean that the simulator is built on detailed level where most of the components consisting of a railway control system are included, such as signals, axle counters, balises, etc. The microscopic railway simulation provides a simulated testing environment which is essential for the MBT. Rather than a real testing environment, the simulated environment reduces the testing costs significantly and makes it possible to test a subsystem before the whole integrated system is finished.

Based on the presented formal modelling method, we introduce our novel MBT method which combines the formal modelling method and simulation. The modelling methods for MBT in previous research model the SUT and HIL environment in a single modelling method. However,

when dealing with CBTC systems, these methods are limited because of the different system characteristics. In a CBTC system, the VOBC and ZC keep exchanging data via bi-direction communication with a desired period, which means some safety-critical functions cannot be realised individually by the VOBC or ZC. For example, the overspeed protection function requires the VOBC to judge whether the train's current speed is over the speed limit within a specified time. The involved speed limit is calculated by the VOBC based on the current MA which is calculated by the ZC and the temporary speed which is limit sent to the VOBC. In order to test such function which is realised collectively by several components in CBTC systems, the specification requirements should be divided into several sub-requirements targeting on each involved component. Some of these sub-requirements could be ineligible for formal modelling because they contain so many variables or calculation processes which make the possibility space become too large to implement model checking [29]. To address the challenge, we introduce our novel hybrid MBT method.

Definition 3. A TIOTS S_{IE} is a quintuple of (S, S_0, A_τ, T_r, X) where S, S_0, T_r, X are the same as defined before and $A_\tau = A_I \cup A_E \cup \tau$, where A_I stands for the internal observable actions and A_E stands for the external observable actions, which means $A_I = A_{I\_in} \cup A_{I\_out}$ and $A_E = A_{E\_in} \cup A_{E\_out}$. Let a finite sequence of observable transitions: $s_0 \xrightarrow{a_0} s_1 \xrightarrow{a_1} s_2 \xrightarrow{a_2} \dots s_{n-1} \xrightarrow{a_{n-1}} s_n$, where $s_n \in S$ and $a_n \in A_I \cup A_E$. Then the observable timed trace $\sum \subseteq \sigma_I \times \sigma_O$, where $\sigma_I \times \sigma_O = d_0 \cdot a_0^E \cdot d_1 \cdot a_0^I \cdot d_2 \cdot a_1^E \cdot d_3 \cdot a_1^I \cdot \dots d_N \cdot a_n^E \cdot d1 \cdot a_n^I$, where $d_N \in X$, $a_n^E \in A_E$, $a_n^I \in A_I$. Assume there is a transition, $s_{IE} \overset{\Sigma}{\Rightarrow} s'_{IE} \in S$, then the observable timed trace from every state can be written: $Tr(s_{IE}) = \{\sum \in (A \cup \{R \geq 0\})^*\}$, where * denotes an abstracted transition relationship [16]. Let $Input(s_{IE}) = \{a \in (A_{I\_in} \cup A_{E\_in}) | s_{IE} \overset{a}{\Rightarrow}\}$, $Output(s_{IE}) = \{a \in (A_{I\_out} \cup A_{E\_out}) | s_{IE} \overset{a}{\Rightarrow}\}$. Therefore, observable inputs and outputs on both internal and external layers are respectively .

Based on the introduced definitions, the modelling method can be illustrated as follows:

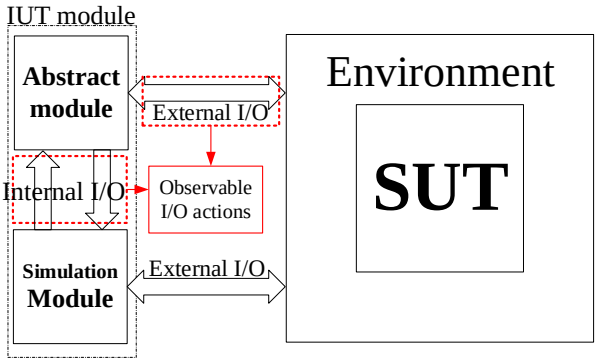

Fig 3 Schematic of hybrid modelling for hybrid online testing platform

As shown in Fig 3, all the observable inputs and outputs are put into a formal module and simulation module. The formal module directly interacts with the simulation module, which acts as a simulated environment for the HIL testing environment; in the simulation module, the whole SUT module is inserted into the HIL testing environment and interactions happen between the simulation module and the HIL environment. The modelling structure takes advantage of the two modelling methods in two modules so that the SUTs with hybrid characteristics can be modelled and tested. The formal module takes charge of checking the refined discrete specification and the simulation module is designed to deal with continuous variables and refine them into an acceptable format desired by the specification in the formal module. The limitation of formal methods can be overcome because the elements which may cause state explosion could be removed by the simulation module. However, the challenge is that the two layers bring us a new problem which needs to be solved. How to synchronise the behaviour which happens on the two layers is the key issue of the hybrid MBT method.

D. Hybrid Online Testing Platform

Based on the introduced hybrid MBT method, we present our novel hybrid online MBT platform in this chapter. Before starting to build the platform, the software tools applied to form the simulation and formal modules need to be determined. Based on *chapter C*, the microscopic railway simulator developed by the BCRRE group is eligible to provide a HIL environment for a CBTC system, and we have chosen it as a software tool for the simulation module. To build up the testing platform, the next step is to choose an appropriate MBT testing tool which is compatible with the simulator written in JAVA. We eventually chose UPPAAL-TRON as the MBT tool because it supports the online testing function, provides a non-commercial license and has a decent compatibility with the JAVA programme [30]. This chapter depicts the process of building up our novel hybrid online MBT platform.

1) Online MBT method with UPPAAL-TRON

To realise the online testing, we introduce an online MBT algorithm presented by **Algorithm 1** which can determine the conformance relationship between the SUT and the specification. Based on the reachability analysis according to the current states, valid inputs are generated by the algorithm and sent to the SUT. The outputs from the execution of inputs are collected and compared with the expected outputs from the specification [20]. If the observed output responding to the current input complies with the expected one from the specification, the algorithm will record it and move to the next set of states where a new set of inputs is available due to the updated reachable set of states[31]. In order to realise the algorithm, UPPAAL-TRON is adopted as the online MBT testing tool. The testing tool continuously repeats this process until an inconsistency is found and a "Fail" is returned, or no inconsistency is found when the testing time is up and a "Pass" is returned.

Algorithm 1.

Initial: $RS \coloneqq \{(s_0, e_0), clock = 0\}$

while $RS \neq \Phi$ and $clock \leq delay$

do *choose randomly:*

Action:

if *Input* $(RS) \neq \Phi$

$randomly\ choose\ a_I \in Input\ (RS)$

$send\ a_I\ to\ SUT$

$RS \coloneqq RS\ After\ a_I$

Delay:

$randomly\ choose\ d \in Delays(RS)$

$wait\ for\ d\ or\ actived\ by\ output\ a_O$ if $d' \leq d$

if a_O *arrives* ***when*** $d' \leq d$

then

$RS \coloneqq RS$ ***After*** a_O

if $a_O \notin Output(RS)$ ***then return*** $fail$

else $RS \coloneqq RS$ ***After*** a_O

else $RS \coloneqq RS$ ***After*** d

Restart:

$RS \coloneqq \{(s_0, e_0), clock = 0\}$

reset SUT ***if*** $RS = \Phi$

then return $fail$

else return *pass*

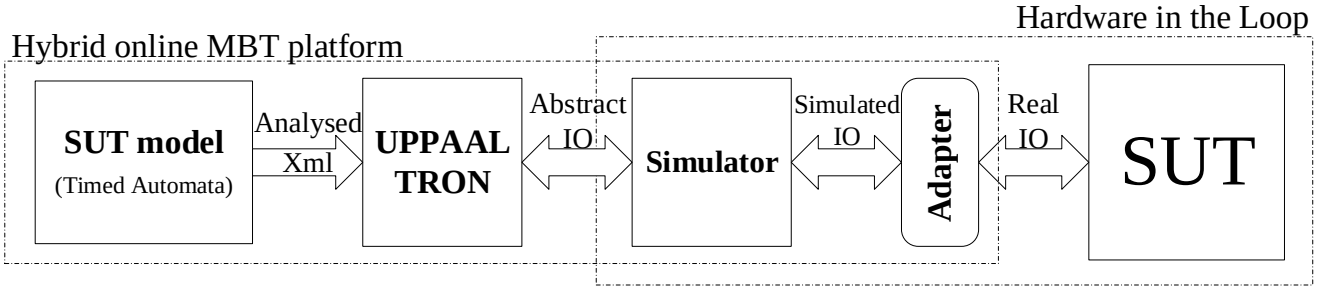

Fig 4 The configuration of the hybrid online MBT platform

Revealed by the pseudo code in **Algorithm 1,** the repeated processes are: ***Action*** where a timed input is randomly chosen from the set of valid inputs determined by the current reachable set and sent to the SUT; ***Delay*** where the testing tool waits for the output derived from the sent input and inspect whether the arrived output is the expected one satisfying the time constraints and legal output values; ***Reset*** when the reachable set is empty so that no input is available for testing.

2) Hybrid online MBT testing platform

Based on the chosen MBT tool and simulator, we developed a hybrid online MBT platform for system testing as shown in **Error! Reference source not found.**. The main components of the testing platform are integrated on a portable host PC which can be conveniently taken to the testing site. The adapter can be customised for different SUTs and integrated into the simulator after the SUT functions are well determined. The hybrid online MBT platform consists of four main components, the SUT model, the MBT tool UPPAAL-TRON, the microscopic railway simulator, and the adapter. The SUT model is written in the format of Timed Automata [32, 33] by UPPAAL, which is a modelling and verification tool based on the TA theory and it is the engine of UPPAAL-TRON. Discrete SUT behaviour is written in the model based on the specification requirement which is provided by the SUT manufacturer. The MBT tool UPPAAL-TRON is utilised to analyse the TA model and generate the test cases by extracting abstract inputs and outputs. The received outputs are compared with the expected outputs to determine whether the SUT behaviour complies with the specification requirements. The simulator models the functions that relate to continuous variables and provides a testing environment for the SUT. The inputs and outputs are translated into recognisable formats for the SUT and simulator via a customised adapter.

III. CASE STUDY

In this section, we design a case study to explain how we implement CBTC system testing on the hybrid online MBT platform. We determine the SUT to be the VOBC which is used as the on-board controller of Hefei Metro Line 1. We focus on the overspeed protection function of the VOBC and the objective is to determine whether the VOBC can protect the train from overspeed when a risky situation happens. The test environment is provided by the microscopic railway simulator which is developed by the BCRRE group. **Error! Reference source not found.** depicts the testing process of the testing platform. The testing platform consists of the TA model and simulation model which stands for the test oracle, the online MBT testing tool UPPAAL-TRON, the I/O handler which determines the input and output sequence, and the simulator which provides a general testing environment for the SUT.

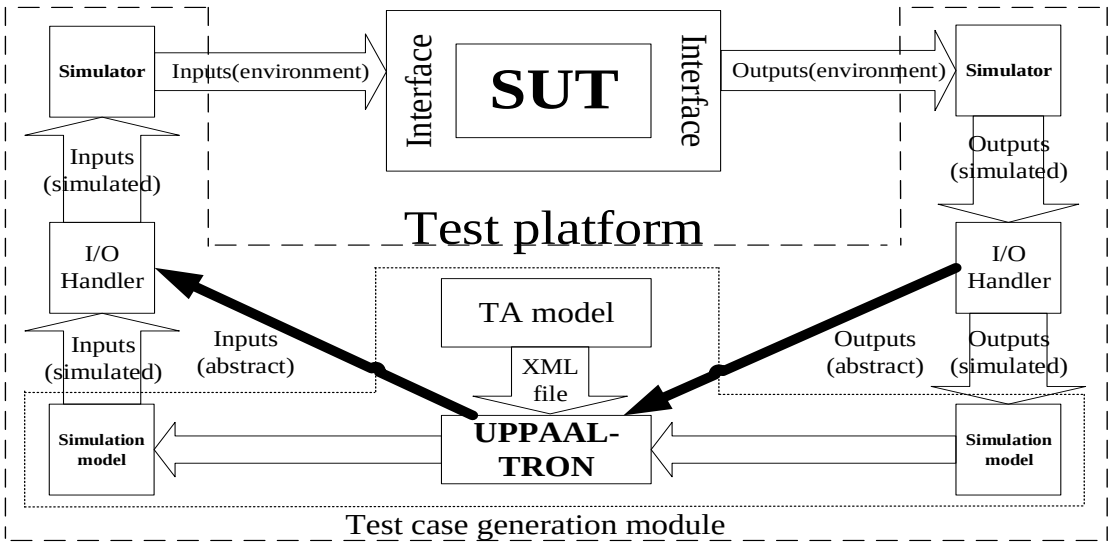

Fig 5 Data flow during the test process with the testing platform

A. *Components of the Hybrid MBT Platform*

1) *SUT (the VOBC)*

The VOBC we choose as the SUT is a real VOBC which can be installed on vehicles of Hefei Metro Line 1, which is designed to train trainee drivers before they drive real trains. Our goal is to test one of the VOBC functions, the overspeed protection function. This function is realised by the cooperation of hardware and software modules in the VOBC and involves a series of complex interactions between different components. We focus on the system level variables such as train speed and train position, but not the actual analogue signal sent from the speed sensor.

2) *TA model and UPPAAL-TRON*

We build a TA model based on refined specification of the SUT, the VOBC. The specification requires the VOBC to protect the train from overspeed by triggering the emergency brake (EB) when the train is about to overspeed. This specification is related to the VOBC, the ZC and the signalling and the train, which generate a series of different testing scenarios including different reasons for train overspeed. No matter what factor makes the train overspeed, the VOBC should always comply with the rule that the train current speed should never be higher than the train current speed limit. Based on the hybrid MBT theory introduced before, we refine the SUT specification into the following sub-specification:

a. The VOBC should receive the train current speed with a period of 200 ms.

b. The VOBC should receive the train MA with a period of 200 ms.

c. The VOBC should calculate a correct speed limit based on the received train MA.

d. In every period, the VOBC should compare the received train speed with the calculated speed limit. If the train speed is over the speed limit, the VOBC should trigger the EB within 1 second.

Since the refined specifications b and c contain relatively complex calculation processes and have a potential risk of increasing the TA model complexity, we model these two specifications by simulation and refine the specification for the TA model:

a. The VOBC should receive the train current speed with a period of 200 ms.

b. In every period, the VOBC should compare the received train speed with the calculated speed limit. If the train speed is over the speed limit, the VOBC should trigger the EB within 1 second.

The implementation model and the environment model of the SUT in UPPAAL are presented in Fig 6 and Fig 7. As introduced in the section II, circle nodes stand for states and arrows stand for transitions from one state to another. Along with transitions, input actions with '!' and output actions with '?' can happen. For example, the 'Depart!' in Fig 7 and the 'Depart?' in Fig 6 mean that the tester sends an departure command to the SUT and the SUT receives it when the conditions of both involved transitions are satisfied. Equations marked in blue are the variable manipulations happen with the corresponding transitions. The expressions marked in light yellow define uncertainties of the observed output values, which can be caused by the nondeterministic SUT behaviour or the uncertain communication delays between the testing platform and the SUT.

Correspond author: Lei Chen, AE

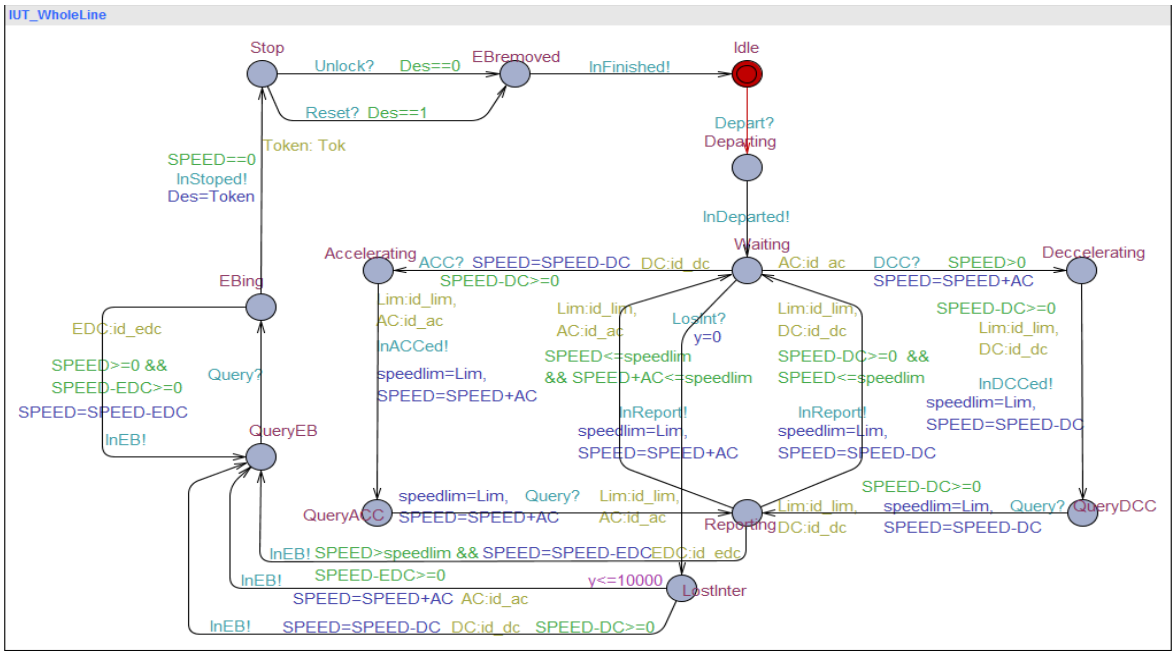

Fig 6 An example of the SUT implementation model written in UPPAAL

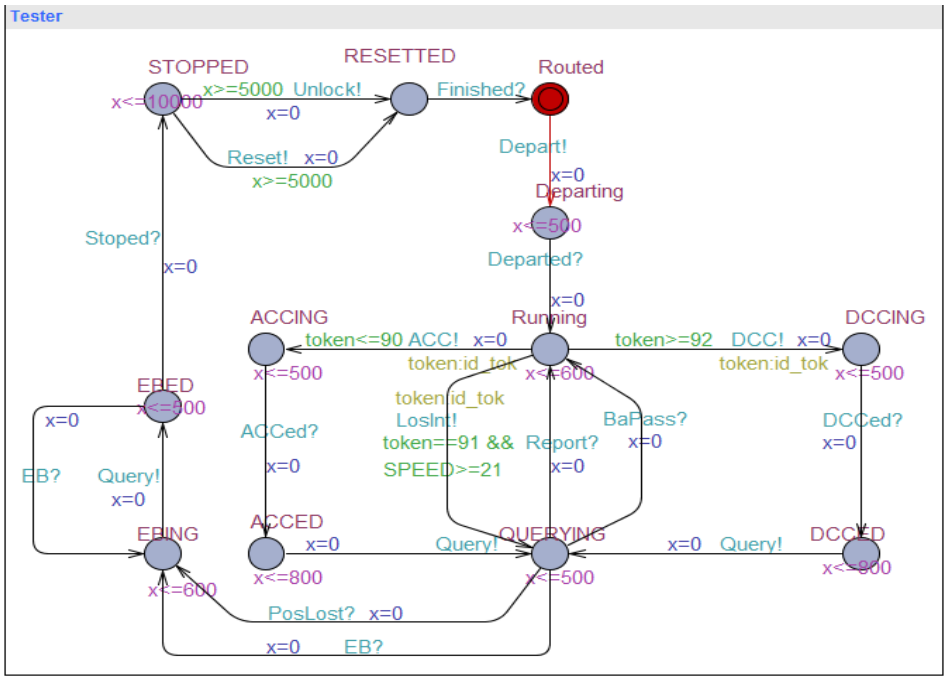

Fig 7 An example of the SUT environment model written in UPPAAL

To achieve the testing objective, we need to formulate the test case for train overspeed. There are lots of factors that can cause overspeed. In this paper, we only choose the most straightforward one. We integrate an inexperienced driver who randomly chooses to accelerate and decelerate. We define a token to specify the probability that the driver chooses accelerating and decelerating. In this model, the probability of accelerating is 93% and the probability of decelerating is 7%.

Correspond author: Lei Chen, AE

3) Simulation model

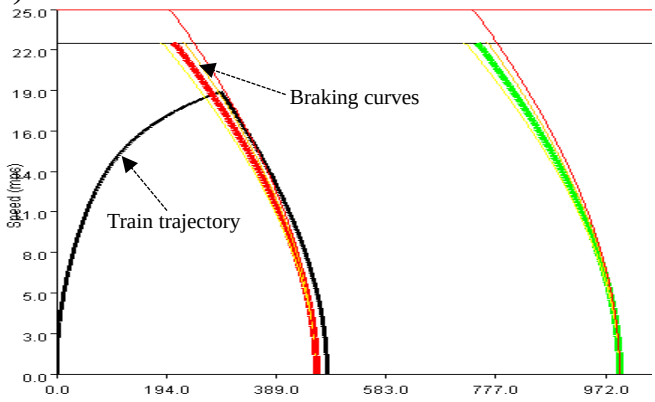

Fig 8 Overspeed scenario: exceeding the speed limit generated by MA

To reduce the complexity of the TA model, the simulation model is designed to calculate the speed limit of the train based on the received MA from the HIL environment. The VOBC should follow the minimum speed limit of the line speed limit and the speed limit generated by the MA. The two graphs in and Fig 9 show two different scenarios of overspeed, which are the overspeed caused by exceeding the speed limit generated by MA and the overspeed caused by exceeding the speed limit generated by line speed limit. Since the VOBC provider does not classify the braking into service braking or emergency braking, we name both two kinds of braking as emergency braking.

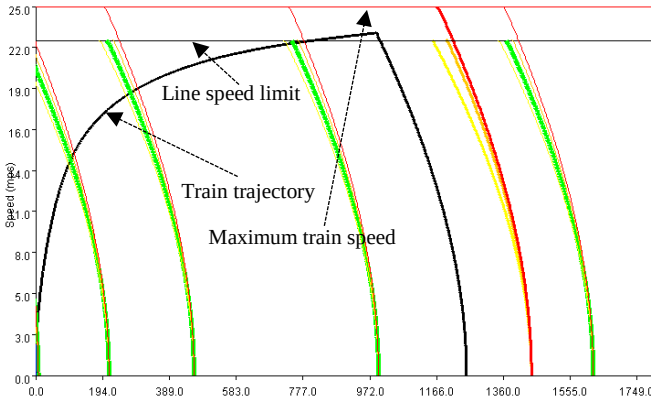

Fig 9 Overspeed scenario: exceeding the speed limit generated by line speed limit

As revealed by Fig 8 and Fig 9, the train trajectory, the line speed limit, the braking curves determined by the line speed limit or the train MA, and the maximum train speed limit are presented in the figures. After simulating the calculation of the speed limit based on the presented elements, the combination of the TA model and simulation model can be used to stand for the specification of the SUT.

4) I/O handler

Since the specification model consists of the TA model and the simulation model, two kinds of interaction happen in the testing process. The internal interaction happens between the TA model and the simulation model, and the external interaction happens between the specification model and the SUT. The correct sequence of I/O actions needs to be guaranteed to avoid conflict against each field. Therefore, the I/O handler is designed to manage the internal and external I/O sequence. As shown in Fig 10, three kinds of I/O channels are designed to realise the interactions between the specification model and the simulation environment. The internal channel is used for internal interactions between the UPPAAL-TRON and the simulation model, which in this case is the train current speed and speed limit. The external I/O channel 1 is used for the interactions between the UPPAAL-TRON and the simulation environment. In this case, it is used to pass the train control command, such as the acceleration and deceleration command. The external I/O channel 2 is used for the interactions between the simulation model and the simulation environment. In this case study, it is used to pass the train speed and the MA. To protect each channel from conflict against others, the I/O handler needs to determine what channel should be activated and no two channels can be activated at the same time.

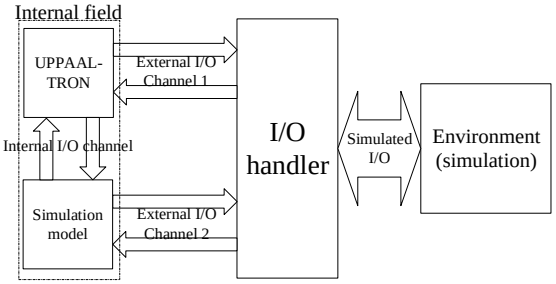

Fig 10 The operation principle of the I/O handler designed for the testing platform

5) Simulation environment

The simulation environment is provided by the microscopic railway simulator. In this environment, all the components essential for the system operation are simulated. During the testing process, the SUT VOBC is installed on a simulated train and the train is controlled by the inputs generated from the specification model. When the simulated train moves along the track, the simulator keeps updating the conditions of the whole network and sends related outputs back to the testing platform. The simulation environment merges all the components into an integrated platform and system level testing can be implemented based on that platform.

As can be seen in **Error! Reference source not found.**, which presents the testing environment utilised in the case study, the testing environment consists of four stations and one depot. During the testing, the train always departs from the depot which is on the right of the top line. The train runs along the track until it reaches the left end of the top line. The train's movement is controlled by the MBT tool. If the EB is triggered during the procedure, the whole system will be automatically reset after the train completely stops.

B. Testing Implementation

Traditional testing on CBTC systems is usually based on manually generated testing cases which only cover the

function under test in a single scenario. **Error! Reference source not found.** shows an example of the manually generated test case which is aimed at testing the overspeed protection function of a VOBC in a scenario of a 40 km/h line speed limit. To implement the test, the tester needs to follow the instructions specified by the test case and manually operate the SUT, which means that the testing can only cover a single sequence but not all the possibilities. Different from traditional testing, the hybrid online MBT platform implements the testing by randomly choosing a valid input and executing it automatically. After the input is successfully executed, the platform carries on choosing another one until all the valid inputs have finished executing. Therefore, the tester does not need to decide when or where to inject the fault to achieve the testing objectives.

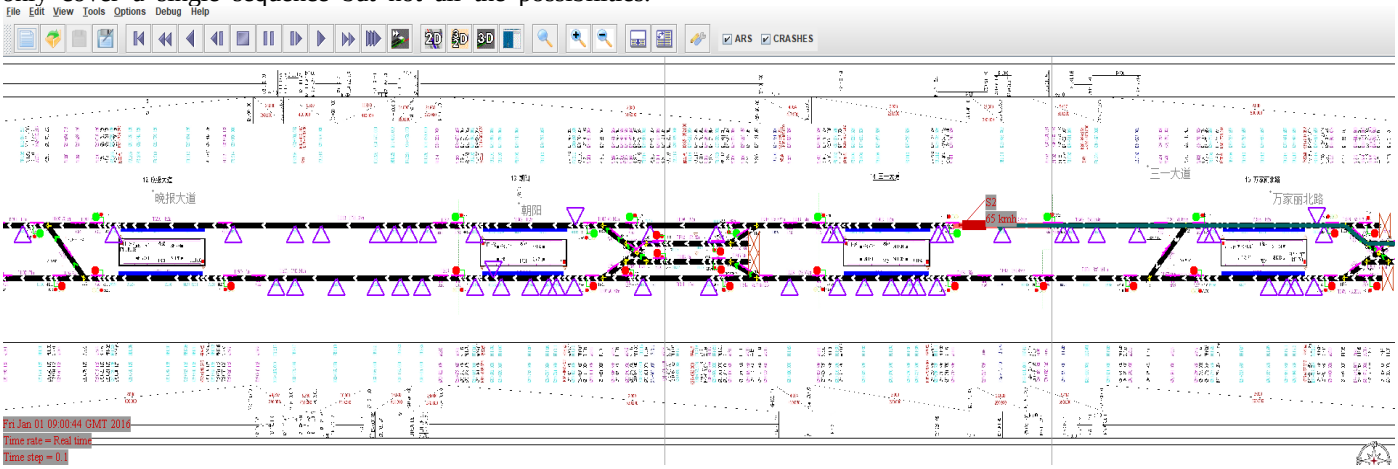

Fig 11 The overall outlines of the testing platform

Fundamental information		
	Content	**ID**
SUT	VOBC	
Description of test case	With line speed set to 40km/h, the driver ignores the line speed limitand accelerate the train with the maximum traction. When the train speed exceeds the line speed limit, observe whether the VOBC outputs EB to stop the train from overspeed.	
Initial condition	ATPM mode; wheel calibrated; EB unimplemented	
Testing objective	To check： ● When the train is overspeed, observe whether the VOBC triggers EB to stop the train	
Testing steps		
Method	To check DMI： ● Apply maximum traction when approaching the area of speed limit; ● The EB can be observed on the screen of the VOBC	
Restraints	The line speed limit is set to the 40 km/h	
Testing result	PASS	

Table 1: A real example of the testing case utilised in traditional testing of a CBTC system

C. Testing Results Analysis

After continuously testing for about 12 hours, we record the train behaviour while it is moving along the track. Since different driving strategies significantly influence the testing results, we adjust the driver's tendency to make the testing cover a greater area of the track. In **Error! Reference source not found.** which records a part of train movements during the test, the train driver has 7 percentages to decelerate the train. Revealed by Fig 12, the train firstly stops at the point around 2 kilometers and eventually stops at the point around 4.8 kilometers where it is closed to the end of the track.

We do not include the train position as a parameter in the testing process because the combination of train position and train speed will lead to state explosion in this scenario. The train speed exceeds the speed limit represented by the orange line because the real SUT has a 5 km/s tolerance for the overspeed train. During the whole testing procedure, no failure is found so that the testing is finally passed.

D. Mutation Testing

To prove our hybrid online MBT platform is capable of finding out errors in the SUT, we design a set of mutation testing on the testing platform [34]. To see whether the testing platform can find out the inconsistency between the SUT and the specification, the following mutation testing is implemented and the results are presented in Table 2.

The Table 2 indicates the results of the mutation testing. The mutation testing is recorded as passed if the testing platform detects the inserted errors and it is recorded as failed if it does not detect the inserted errors. Based on the results, only one mutation is failed. When we add a minor delay in the range of 10 ms to 100 ms, the SUT behaviour is slightly delayed by 10 ms to 100 ms and the testing platform cannot detect this amount of delay. When the delay is more than 100 ms, we find that the testing platform can steadily detect the delay and it sends out a failed verdict.

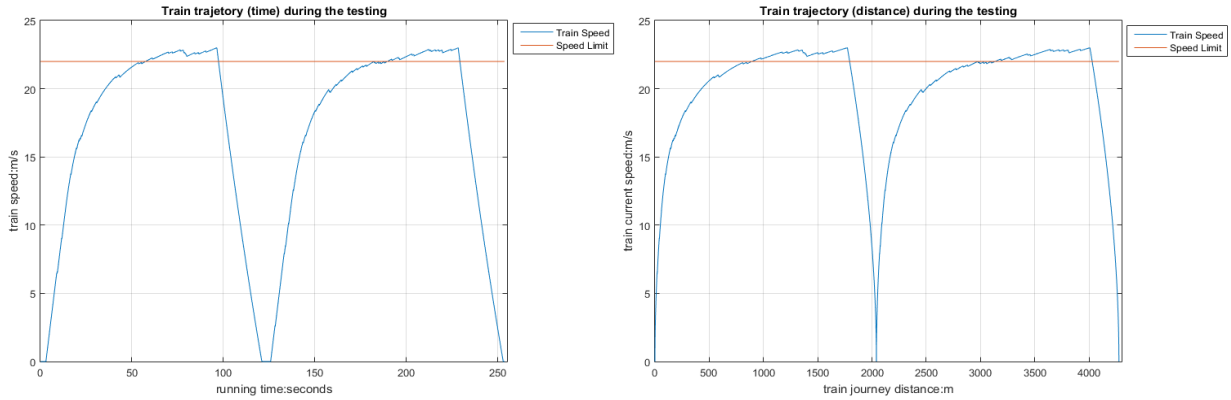

Fig 12 The train behaviour during the testing process: 93% acceleration

Mutation error	Error in the platform	Test results
Wrong output action	e.g. accelerate the train when the driver decelerates the train	Passed
Incorrect output value	e.g. make the train accelerate with an incorrect acceleration.	Passed
Minor delay (10ms~100ms)	e.g. insert a minor delay in the I/O channel	Failed
Major delay (>100ms)	e.g. insert a major delay in the I/O channel	Passed
Missing state	e.g. remove states “stop” (see Fig 6) in the simulation	Passed
Transition to wrong state	e.g. change the transition “Decelerating” to “queryACC” (see Fig 6)	Passed
Incorrect initial value	e.g. give the train a non-zero initial speed	Passed

Table 2 The summary of testing results for mutation testing

E. Coverage analysis

Coverage is a concept of off-line MBT testing which indicates how many traces, locations and variable values out of all the possible values have been covered by the generated test cases. Two key factors influence the coverage, the model complexity and the search depth. A complex model contains numerous possible states and usually leads to a low coverage. The search depth means the steps used to generate the test cases, where 1 step means 1 transition from one location to another one. More steps give the algorithm possibility to achieve more locations and thus more coverage. To determine the performance of our online MBT platform, we compare the coverage with the offline test generation algorithm under the same model. The offline testing coverage analysis is realised by a tool box integrated in UPPAAL and the online testing coverage is obtained from the testing log file. We extract the covered and uncovered traces or variable values from the log file then calculate the coverage by comparing them with the all possible set of traces or variable values. Based on the graph of the coverage against the search depth, the quantised performance of the online and offline testing methods can be obtained. Revealed from **Error! Reference source not found.** and **Error! Reference source not found.**, online testing performs worse than offline testing at the early stage of the search depth. However, with the increasing search depth, the online testing performance eventually surpasses the offline performance at crossing points, 1443 steps for trace coverage and 211 steps for variable coverage. The reason is that our online testing platform is not restricted by the formal model size when it generates the testing cases, but the offline test generation is very sensitive to the model size. When the model is relatively large and complex, offline testing cannot search as deep as online testing can because it needs to record all the pattern to achieve the optimised coverage. When the memory is used up, the coverage reaches its limitation. Our hybrid online MBT reduces the model size by introducing simulation modelling and generates possible input randomly without recording any pattern during the testing procedure. As a result, when offline coverage is blocked by the limitation of the computer's computational ability, our online MBT platform can still achieve 100% coverage for both trace and variable.

The online testing results indicate that our hybrid online MBT platform can detect errors from the SUT more efficiently than traditional testing methods. Since the test generation and execution are automatic, the time taken to finish a single test is significantly reduced. Better efficiency means that the online MBT platform can cover many more situations than traditional testing methods do. By introducing simulation into online testing, the platform manages to implement MBT in a HIL environment, which makes the testing results more convincing than the results from individual testing. Furthermore, our online MBT platform is more extendable and flexible than traditional testing methods. When modification is made on the SUT, the online MBT platform can easily cope with the changes by changing its model, whereas traditional testing methods have to rewrite the specification, test generation and execution. Based on the mutation testing results, the online MBT platform is able to find out most of the errors, except the minor delay error, because it generates and executes test cases simultaneously. As a result, some hard-real-time requirements could be challenging for online MBT [35]. For these hard-real-time requirements, the communication delay between each component and the computation time of the testing platform become nonnegligible. To deal with this kind of specification, communication delay and computation time may need to be taken into consideration by the specification model. A comparison between the online and offline coverages indicates that the online MBT platform can achieve 100% coverage test generation to guarantee that all of the possibilities inside the specification model can be taken into consideration and no potentially risky situations could be missed. Since the online MBT platform is less limited by the model complexity or the computational ability than offline testing, it is more eligible for testing complex or industrial-sized systems, such as CBTC systems.

IV. Conclusion

This paper presents a hybrid online MBT platform by explaining the hybrid online MBT theory and depicting the operation principle of the platform via a case study based on a real SUT. The testing platform is built with the combination of online MBT and our novel microscopic railway simulator. We evaluate the platform with a SUT which is a real VOBC used in metro systems and by use of mutation testing to determine whether the platform is eligible for testing on the CBTC system. It generally succeeds in detecting inconsistency between the SUT and the modelled specification. The testing performance is improved by the hybrid online MBT platform because it removes the manual elements from the testing process. With the benefit of model-based testing which makes the testing more efficient and accurate, the testing platform proves the possibilities to evolve railway system testing into the era of automation.

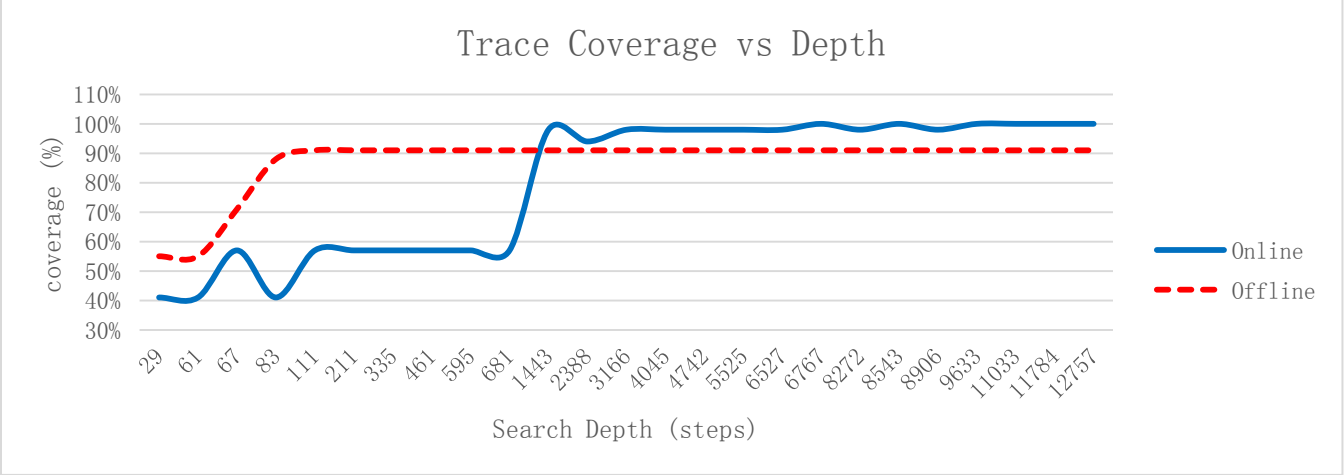

Fig 13 The trace coverage tendencies along with search depth

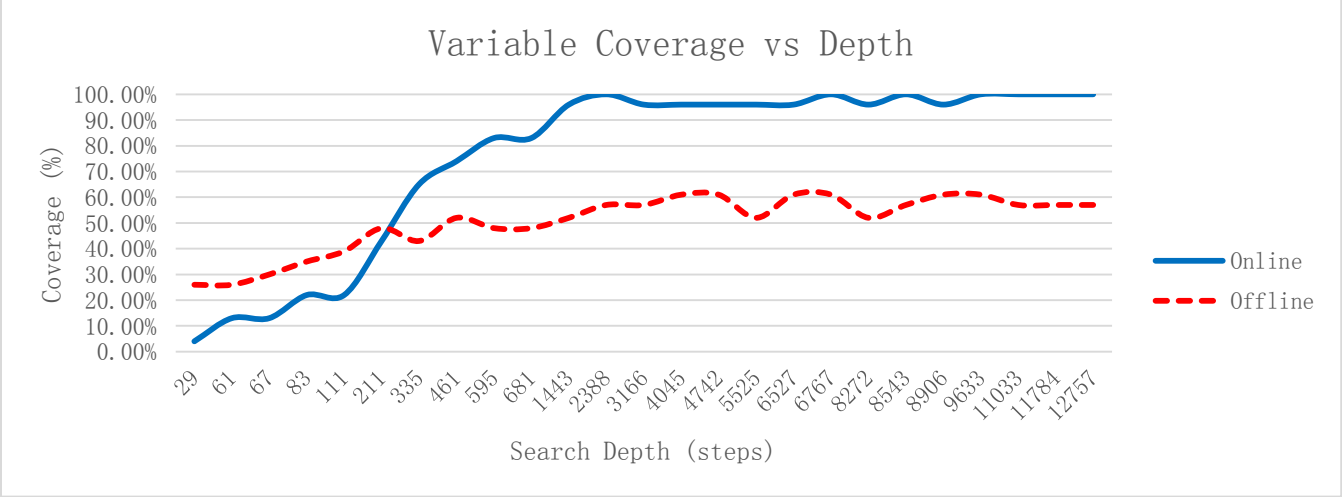

Fig 14 The variable coverage tendencies along with search depth

ACKNOWLEDGMENT

The authors would like to thank the support from Birmingham Centre for Railway Research and Education. The research work was also supported by Chinese Scholarship Council, the National Natural Science Foundation of China (U1434209) and the Fundamental Research Funds for the Central Universities (2016JBZ004).

REFERENCES

[1] IEEE, "IEEE Standard for Communications-Based Train Control (CBTC) Performance and Functional Requirements," ed: IEEE, 2004.

[2] D. Giannakopoulou, C. S. Pasareanu, and C. Blundell, "Assume-guarantee testing for software components," *IET Software*, vol. 2, pp. 547-562, 2008.

[3] J. S. Keranen and T. D. Raty, "Model-based testing of embedded systems in hardware in the loop environment," *IET Software*, vol. 6, pp. 364 - 376, 04 October 2012 2012.

[4] P. Samuel, R. Mall, and A. K. Bothra, "Automatic test case generation using unified modeling language (UML) state diagrams," *IET Software*, vol. 2, pp. 79-93, 2008.

[5] IEEE, "IEEE Recommended Practice for Functional Testing of a Communications-Based Train Control (CBTC) System," vol. 1474.4-2011, ed: IEEE, 2011.

[6] M. Utting and B. Legeard, *Practical model-based testing: a tools approach*. San Francisco: Morgan Kaufmann, 2007.

[7] E. Dincel, O. Eris, and S. Kurtulan. (2013, 23 October 2013) Automata-Based Railway Signaling and Interlocking System Design. *IEEE Antennas and Propagation Magazine*. 308 - 319.

[8] W. Zheng, C. Liang, R. Wang, and W. Kong, "Automated Test Approach Based on All Paths Covered Optimal Algorithm and Sequence Priority Selected Algorithm," *IEEE Transactions on Intelligent Transportation Systems*, vol. 15, pp. 2551-2560, 2014.

[9] J. Zander, I. Schieferdecker, and P. J. Mosterman, *Model-based testing for embedded systems*: CRC press, 2011.

[10] O. Tkachuk and M. B. Dwyer, "Environment generation for validating event-driven software using model checking," *IET Software*, vol. 4, pp. 194-209, 2010.

[11] D. Amalfitano, A. R. Fasolino, and P. Tramontana, "MobiGUITAR: Automated Model-Based Testing of Mobile Apps," *IEEE Software*, vol. 32, pp. 53 - 59, 10 April 2014 2015.

[12] N. Li and J. Offutt, "Test Oracle Strategies for Model-based Testing," *IEEE Transactions on Software Engineering*, vol. PP, pp. 1-1, 02 August 2016 2016.

[13] A. C. Dias-Neto and G. H. Travassos, "Supporting the Combined Selection of Model-Based Testing Techniques," *IEEE Transactions on Software Engineering*, vol. 40, pp. 1025 - 1041, 20 March 2014 2014.

[14] R. Cardell-Oliver, "Conformance Tests for Real-Time Systems with Timed Automata Specifications," *Formal Aspects of Computing*, pp. 350-371, 2000.

[15] B. Beizer and J.Wiley, "Black box testing: Techniques for functional testing of software and systems," *IEEE Software*, vol. 13, p. 98, 1996.

[16] W. Yuemiao, C. Lei, W. Jinwen, D. Kirkwood, X. Qian, J. Lv, *et al.*, "On-line conformance testing of the Communication-Based Train Control (CBTC)

system," in *2016 IEEE International Conference on Intelligent Rail Transportation (ICIRT)*, 2016, pp. 328-333.
[17] L. Padgham, Z. Zhang, J. Thangarajah, and T. Miller, "Model-Based Test Oracle Generation for Automated Unit Testing of Agent Systems," *IEEE Transactions on Software Engineering*, vol. 39, pp. 1230-1244, 2013.
[18] M. Mikucionis, K. G. Larsen, and B. Nielsen, "Online On-the-Fly Testing of Real-time Systems," *Basic Research in Computer Science*, p. 14, December 2003.
[19] Z. Xiaolin, L. Teng, L. Kaicheng, and L. Jidong, "Online Testing of Real-time Performance in High-speed Train Control System," presented at the IEEE 17th International Conference on Intelligent Transportation Systems (ITSC), Qingdao, China, 2014.
[20] M. Broy, B. Jonsson, J.-P. Katoen, and M. Leucker, *Model-Based Testing of Reactive Systems*: Springer, 1973.
[21] L. Jidong, W. Haifeng, L. Hongjie, Z. Lu, and T. Tao, "A model-based test case generation method for function testing of Train Control Systems," in *2016 IEEE International Conference on Intelligent Rail Transportation (ICIRT)*, 2016, pp. 334-346.
[22] S. Li, X. Chen, Y. Wang, and M. Sun, "A Framework for Off-Line Conformance Testing of Timed Connectors," presented at the International Symposium on Theoretical Aspects of Software Engineering, 2015.
[23] J. Lv, K. Li, G. Wei, T. Tang, C. Li, and W. Zhao, "Model-Based Test Cases Generation for Onboard System," *2013 IEEE Eleventh International Symposium on*, pp. 1-6, March 6-8 2013.
[24] G. Gay, S. Rayadurgam, and M. Heimdahl, "Automated Steering of Model-Based Test Oracles to Admit Real Program Behaviors," *IEEE Transactions on Software Engineering*, vol. PP, pp. 1 - 1, 05 October 2016 2016

[25] D. Xu, M. Kent, L. Thomas, T. Mouelhi, and Y. L. Traon, "Automated Model-Based Testing of Role-Based Access Control Using Predicate/Transition Nets," *IEEE Transactions on Computers*, vol. 64, pp. 2490-2505, 2015.
[26] J. CHEN, Y. REN, X. GAO, and Y. JIN, "The Research on Conformance Testing Platform of Numerical Substation," presented at the CICED2008, 2008.
[27] K. G. Larsen, M. Mikucionis, B. Nielsen, and A. Skou, "Testing Real-Time Embedded Software using UPPAAL-TRON: An Industrial Case Study," presented at the ACM International Conference On Embedded Software, Jersey City, NJ, USA, 2005.
[28] L. Dai, "Data for constructing experimental scenarios on testing the performance of rescheduling approaches," *Data in brief*, 2016.
[29] D. Angeletti, E. Giunchiglia, M. Narizzano, A. Puddu, and S. Sabina, "Using Bounded Model Checking for Coverage Analysis of Safety-Critical Software in an Industrial Setting," *Journal of Automated Reasoning*, vol. 45, pp. 397-414, December 2010 2010.
[30] C. Rutz and J. Schmaltz, "An Experience Report on an Industrial Case-Study about Timed Model-Based Testing with UPPAAL-TRON," presented at the Fourth International Conference on Software Testing, Verification and Validation Workshops, 2011.
[31] M. Mikucionis and E. Sasnauskaite, "On-the-fly Testing Using UPPAAL," Master, Department of Computer Science, Aalborg University, Fredrik Bajers VEJ 7B, 9220 AALBORG ØST, DENMARK, 2003.
[32] A.Anier and J.Vain, "Timed automata based provably correct robot control," presented at the 12th Biennial Baltic Electronics Conference, Tallinn, Estonia, 2010.
[33] D. K. Kaynar, N. Lynch, R. Segala, and F. Vaandrager, "Timed I/O automata: a mathematical framework for modeling and analyzing real-time systems," in *Real-Time Systems Symposium, 2003. RTSS 2003. 24th IEEE*, 2003, pp. 166-177.
[34] R. Baker and I. Habli, "An Empirical Evaluation of Mutation Testing for Improving the Test Quality of Safety-Critical Software," *IEEE Transactions on Software Engineering*, vol. 39, pp. 787-805, 11 September 2012 2013.
[35] I. Schieferdecker, "Model-Based Testing," *IEEE Software*, vol. 29, pp. 14-18, 2012.

THE AUTHORS

Yuemiao Wang received the B.E. degrees of Engineering from Fudan University, Shanghai, China and the University of Birmingham, Birmingham, the UK, in 2014. He is currently working toward the Ph.D. of railway engineering degree in the Birmingham Centre for Railway Research and Education, the University of Birmingham. His research interests include railway system testing, formal verification and testing, and CBTC system modelling.

Lei Chen received the B.Eng. degree in automation engineering from Beijing Jiaotong University, Beijing, China, in 2005, and the Ph.D. degree in railway traffic management from the University of Birmingham, Birmingham, U.K., in 2012. He is

currently a Birmingham Fellow for Railway Traffic Management with the Birmingham Centre for Railway Research and Education, University of Birmingham. His research interests include railway traffic management and control, railway safety critical system design, and railway simulation.

David Kirkwood received a BSc. in Computing Science in 2002 from Staffordshire University, UK. He then worked as a software developer in industry, specialising in Air Traffic control simulation. In 2015, he was awarded a PhD from the University of Birmingham and now works as a Research Fellow in the Birmingham Centre for Railway Research and Education at the University of Birmingham, specialising in the development and application of railway simulation software. He has participated in research projects related to railway traffic management, train control algorithms and hardware in the loop simulation and testing.

Peng Fu, senior engineer, graduated from Tongji University in major of communication engineering, works in STEDI and he is both the associate dean of the electrical branch of STEDI and project manager in Beijing. He mainly works for the communication signal, integrated monitoring, design of the operation control center and consulting in the field of tunnel engineering and rail transit.

Jidong Lv received the B.Eng. degree in Automation Engineering from Beijing Jiaotong University, China, in 2004, and the Ph.D. degree in Traffic Information Engineering and Control also from Beijing Jiaotong University, in 2011. He is now an associated professor with the National Engineering Research Centre of Rail Transportation Operation and Control Systems at Beijing Jiaotong University. His research interests include formal modeling and verification of hybrid systems, especially in real-time systems and model-based test case generation in high-speed railway train control systems.

Clive Roberts is Professor of Railway Systems at the University of Birmingham. Clive is Director of the Birmingham Centre for Railway Research and Education, which is the largest railway research group in Europe with just over 100 researchers. He works extensively with the railway industry and academia in Britain and overseas. He leads a broad portfolio of research aimed at improving the performance of railway systems, including a leading strategic partnership in the area of data integration with Network Rail. His main research interests lie in the areas of railway traffic management, condition monitoring, energy simulation and system integration.