Scenario-based Testing for Onboard Subsystem

Weihui Zhao[1], Chenling Li[1], Jidong Lv[2], Lei Yuan[1]

1 State Key Laboratory of Rail Traffic Control and Safety, Beijing Jiaotong University, Beijing, China
2 National Engineering Research Center of Rail Transportation Operation And Control System, Beijing Jiaotong University, Beijing, China
11120328@bjtu.edu.cn

Abstract - As a typical safety-critical system, the Onboard subsystem is a core subsystem in CTCS-3(Chinese Train Control System level 3), functional test is necessary to validate the conformance relation between the Onboard subsystem and its specification. Most of test cases are manually generated and can't be reused and leads to repeated works when the specification is changed. We introduced scenario-based method to improve the efficiency and quality of onboard equipment testing. The scenario-based hierarchy model of onboard subsystem was established according to the specification, and all definition coverage criteria were proposed with the all definition coverage observer. Then, an automatic tool chain was used to establish the onboard automata model and generate test cases for every scenario. Finally, a selection algorithm was given to choose a complete test sequence from the test cases, which was proven comprehensive to cover all running modes and efficient for onboard equipment testing.

Index Terms - scenario, test case, onboard subsystem, mode transition

1. Introduction

Test cases are the basis of the function testing and how to automatically generate test cases that satisfy the system specifications completely is the key issue of the test. Most of the traditional test cases are manually generated which can't be reused and leads to repeat works when the specification is changed. With the widespread of formal method, a lot of automatic test case generation technologies are introduced, and scenario-based testing has been gradually used. In scenario-based testing, scenario techniques[3] are applied to the test case expression, and the conversion tool will automatically generates test cases that can be run on a certain test platform and achieve the purpose of the test. Scenario-based test cases make the system easier to understand, and are widely used in the functional testing of the system.

The onboard system of CTCS-3 is responsible for the implementation of over-speed protection and safe distance between trains, and any fault can lead to huge human injury or wealth losing [1]. To ensure the correctness of functions in the onboard system, function testing is mainly used to test the conformance relation between the specification and the onboard equipment.

To improve the testing efficiency and quality of the onboard system, in this paper, a scenario- based test generation algorithm is given in order to generate a complete test suit of the function of the onboard system. We use automatic tools to generate test cases automatically based on scenario technique. Then, choosing algorithm is developed to achieve the test sequence covering all modes of onboard system. The complete model transition function test suit is derived on the basis of scenario-based hierarchy model, automata network model, and the observer theory.

2. Modeling of Onboard Subsystem

A. Scenario-based Hierarchy Model

Analysis and management of the operating scenarios in CTCS-3 specification are the basic work to get the model of onboard system. Operating scenes describe the running environment, and the train behaviour, the expectative running modes in related condition of certain scene. We classified all scenes in four groups in the modelling process, and made up the scenario tree as system level, objective level and property level. With this scenario technology, we established scenario tree model of onboard subsystem shown in Fig. 1.

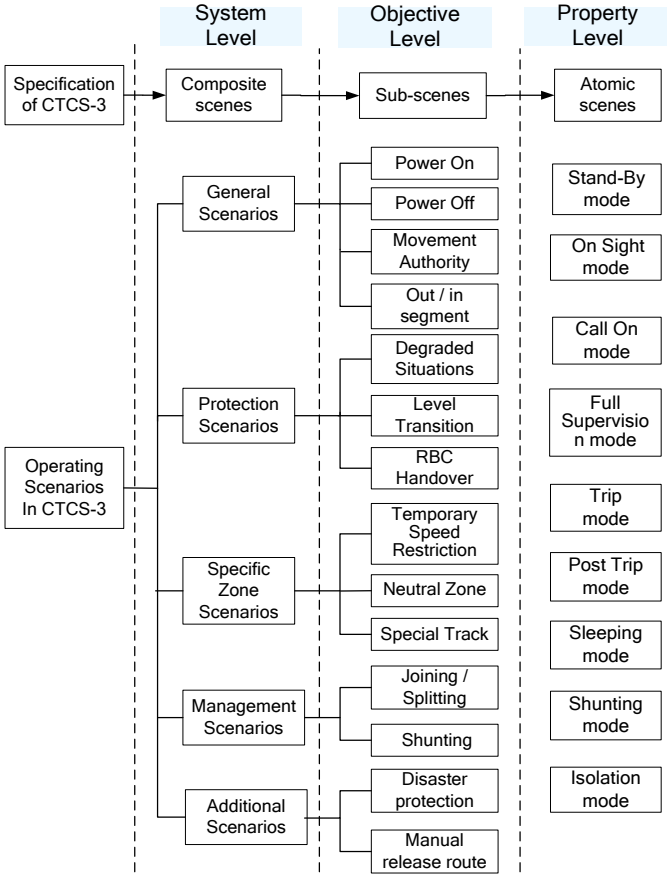


Fig. 1 Scenario-based Hierarchy Model of Onboard Subsystem

System level represents a function packet, reflecting the abstractive function set of the system. It consists of several operating scenarios, called composite scene; Objective level represents the specific function item and the related condition and operation, called sub-scene; Property level represents the minimum function of the system and can be tested independently, called atomic scene.

As in Fig. 1, the function scenarios are classified as general scenarios, protection scenarios, specific scenarios, management scenarios and additional scenarios in system level. In objective level, we match composite scene of CTCS-3 with the operating scenarios respectively as the idea in Reference [2]. The train operating scenarios are expressed by the train operating modes and their transitions, which are used as the atomic scenes. Each scenario contains one or more running modes, so the mode transition can be described with scenario. For example, mode SB to mode CO can be achieved in the scenario, guiding track departure instruction after power on.

B. Kernel-Environment Automata Model

The on-board equipment is a computer-based subsystem that supervises the movement of the train to which it belongs, on basis of information exchanged with the trackside sub-system [4,5]。 The interoperability requirements for the on-board equipment are related to the functionality and the data exchange data exchange between the on-board sub-system and the driver, the train, the specific transmission modules. The described structure is depicted in Figure 2.

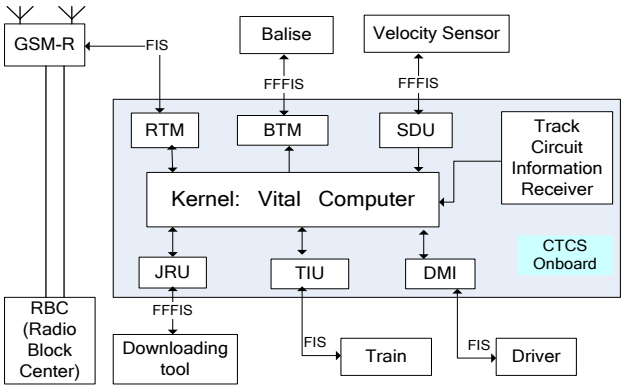


Fig. 2 Architectural Scheme of Onboard Subsystem

Automotive is suitable for modelling and analysis of train control system. Onboard equipments consist of vital computer, train, RBC, balise and driver. Based on the atomic scenes, the network automata model of the onboard subsystem is established with tool Uppaal, according to the mode transition table, symbolic description and transition conditions in CTCS-3 specification. The entire system network automaton model is divided as kernel part and environment part, Kernel(VC) ||Environment（Train, Driver, RBC, Balise）, named kernel-Environment model. Kernel automaton, train automaton, balise automaton, RBC automaton and driver automaton are parallel and they constitute the automata network. The location and edge is defined as the following symbols:

1）Location: <l，σ>, l is the currant location and σ is related variables of location l;

2）edge: e: <l, σ> → < l'，σ'>, location l will change to l' and variable set σ to σ' with certain condition.

3）'!' represents sending messages, and '?' represents receiving messages, realizing the message synchronization in the model to deliver information among the members.

For example, the Kernel automata model is shown in Fig. 3. We define the kernel part of onboard subsystem as $T_k = < S, S^0, A, X, I, E >$, according to the definition of automata, in which the set of locations is S = {SB, SH, FS, OS, CO, SL, TR, PT, IS}, covering all the running modes in CTCS-3. The edge stands for modes transition as the transition rule in chapter 4.4 of reference [4].

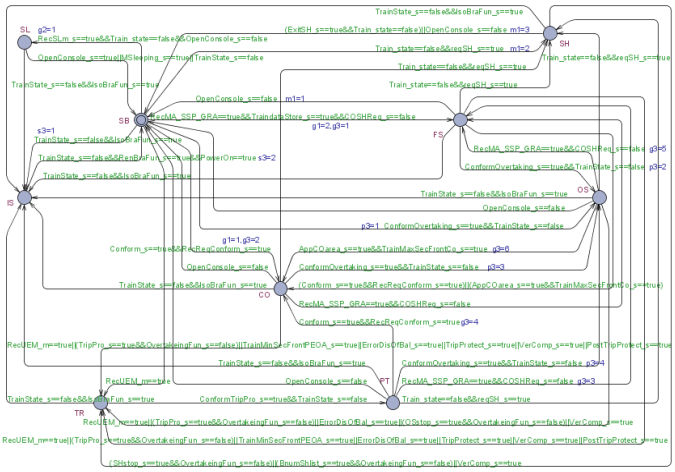


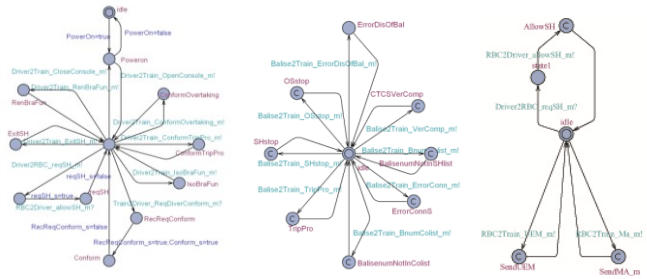
Fig. 3 Kernel Automaton

For the sub-scene level, operating scenarios are simulated in kernel automata according to the procedure described in chapter 3 of Reference [5]. The scenarios variables are defined with set G, P, S, M and A, depicted in Table 1.

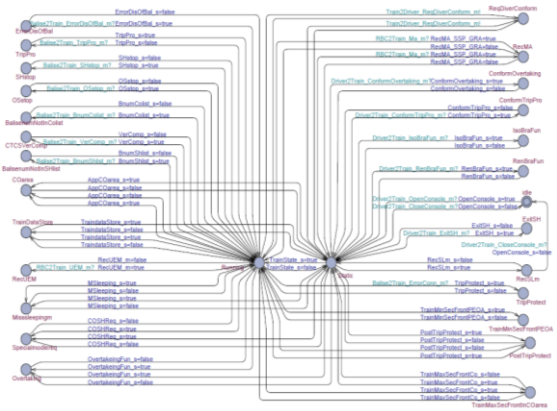
TABLE I Scenario Variables

Symbol	Operating Scenarios
g1, g2, g3, g4	Power On, Power Off, Movement Authority Out / in segment
p1, p2, p3	Level Transition, RBC Handover , Degraded Situation
s1, s2, s3	Temporary Speed Restriction, Pass Neutral Zone, Special Track
m1, m2	Joining / Splitting, Shunting
a1, a2	Disaster protection，Manual release route

Fig. 4 shows the four parts of environment automata, providing the mode transition conditions to Kernel automata.



a) Driver Automaton b) Balise Automaton c) RBC Automaton



d）Train Automaton

Fig. 4 Environment Automata

The information between onboard system and Train is train braking information and train state information, between onboard system and RBC is MA message and train location, between On-board system and Balise is train location and Hazard protection, between onboard system and Driver is the conform and choice of operation.

3. Test for Onboard Equipment

A. Automatic Generation of Test Cases

On the basis of Kernel-Environment model, the test cases can be generated automatically in automatic tool with the observer automata theory.

CoVer [7] is a tool for model-based testing of real-time systems, developed at Uppsala University since 2005. We use it to automatically generate test suites from the model. The tool finds test case from the initial location, and traverses all its possible successive locations with depth-first search method. As an automaton, observer [6] can be used to specify coverage criteria for test generation and monitor the location and edge in test trace. Whenever the coverage item has been covered, the observer location is called “accepting location”. The accepting set records all the test items matching the coverage criteria described by the observer and it will guide test cases generation as a configuration file in CoVer. The test case will be generated when a location is repeatedly traversed. In terms of Kernel- Environment automata and the test target, we proposed test coverage criteria and give the syntax description of all definition coverage observer as following:

Observer defineObs (varid X;) {
node defined (varid, edgeid);
rule start to defined(X,E) with def(X,E);
accepting defined;
}

The all definition coverage observer has an accepting location du (X; E), where X is a variable name, E is an edge on which X is defined, and only the defining edges are required to be covered. In the Kernel automaton, G, P, S, M are used to represent the scenarios of onboard subsystem, so X in the syntax above is the set {g1, g2, g3, g4, p1, p2, p3, s1, s2, s3, m1, m2}. Then, the test suit is automatically generated by tool CoVer. For example, the test cases for management scenarios are generated by searching symbol m1 and part of the outcome script is shown in Fig. 5. With the test cases, we can test the management scenarios as the generated trace: SB -> FS -> SB -> CO -> TR -> PT and SB -> SH -> SB -> CO -> TR -> PT, covering their related modes and mode transition, such as FS to SB, SB to SH and SH to SB that are included in shunning scenario.

```
===== Trace #1==================================
 defined<varid offset=70, edgeid VC.FS_to_idle_SB>
State:
( VC.idle_SB Driver.idle RBC.idle Train.idle Balise.idle )
Transitions:
  Driver.idle->Driver.Poweron {1,tau, PowerOn := 1 }
  Train.idle->Train.Static
  {1,Driver2Train_OpenConsole_m?,OpenConsole_s:=1}
  RBC.idle->RBC.SendMA_m { 1, RBC2Train_Ma_m!, 1 }
  Train.Static->Train.RecMA {1,RBC2Train_Ma_m?,RecMA_SSP_GRA:=1}
  RBC.SendMA_m->RBC.idle { 1, tau, 1 }
  VC.idle_SB->VC.FS
  {RecMA_SSP_GRA==1&&TraindataStore_s ==1&&COSHReq_s==0,tau,1}
State:
( VC.FS Driver._id18 RBC.idle Train.RecMA Balise.idle )
Transitions:
  Train.RecMA->Train.Static { 1, tau, RecMA_SSP_GRA := 0 }
  Driver._id18->Driver.Poweron{1,Driver2Train_CloseConsole_m!,1}
  Train.Static->Train.idle
  {1,Driver2Train_CloseConsole_m?, OpenConsole_s :=0}
  VC.FS->VC.idle_SB { OpenConsole_s == 0, tau, m1 := 1}
```

Fig. 5 Part of Test Cases Script for Joining Scene

B. Test Sequence for All Running Modes

The scenario trace is based on the operating scenes and described as test suit, which shows the mode transition. Selecting mode coverage test cases from the test suit in CoVer is pivotal to make up effective test sequence of onboard subsystem. Thus the problem of finding all mode test sequence is become to find the following trace that covers all modes in the accepting location set Q of observer. An algorithm was designed to get test sequence covering modes entirely:

```
Construct an trace TR;
i = 0; j = 0;
while(i < ScenarioNum) do
  Select test suit Ai : includes more modes;
  Max = TotalTrace(Ai);
  REPEAT
    if exist one mode in Ai.trj : qx ∉ Q then
       Add TestCase_Mode (Ai.trj.q, Q);
       Store TestCase_Edge (Ai.trj, TR);
    j = j+1;
    Find_nextTrace(Ai.trj);
  UNTIL  j = Max;
  i = i+1; j = 0;
RETURN  TR;
```

The programmed algorithm is described as following: for any certain scenario test case set A ,get its trace number MaxTrace, assume the accepting location set of observer is Q ,and our target sequence is TR. Define two variables i=0, j=0 as loop number.

(1) Select a test case A_i from the outcome script that covers modes as much as possible;

(2) If there is new mode in the test trace $A_i.tr_j$, add $A_i.tr_j$ to TR and tr.mode to Q;

(3) Make the pointer variable j = j + 1, and find next trace of A_i, (Find_nextTrace(tr_i));

(4) Execute step 2 and 3 until j= MaxTrace;

(5) Make i=i+1 and go on the same work for next scenario A_i, repeat step 2, 3 and 4;

(6) Return the target sequence TR.

The target sequence was generated from the test suit with the algorithm above, see in Table Ⅱ, covering all running modes in CTCS-3 specification connected by proper scenarios.

TABLE Ⅱ Test Sequence of All Mode Coverage

Case	Test Item
case 1	SB_to_CO_to_TR_to_PT_to_FS_to_SB_to_SL
case 2	SB_to_SH _to_ SB_to_CO _to_TR _to_PT
case 3	SB_to_IS _to_SB _to_CO _to_TR _to_PT
case 4	SB_to_FS _to_OS _to_ SB _to_SL

4. Conclusion

We use automated tools to get test cases automatically based on scenario technique and timed automata theory to improve the testing efficiency and quality, and then apply in onboard equipment testing. This method can avoid the weakness of manual generation, such as the test cases can't be reused and leads to repeated works when the specification is changed. Besides, scenario-based testing method for onboard equipment testing is proven very useful to test all modes of onboard system with proper operational scenes that connect all the modes automatically.

5. Acknowledgements

The research was supported by the project of National High-tech R&D Program of China: Safety certification and assessment technology of High-Speed railway signal system; the Fundamental Research Funds for the Central Universities: Research of formal design and development method in High-Speed train control system

6. References

[1] LV Jidong. Hierarchical Formal Modeling and Verification Train Control System [D]. Beijing: Beijing Jiaotong University, 2011. In Chinese.

[2] Beizer B. Black-Box Testing Technique for Functional Testing of Software and Systems, Wiley,New York, USA9,1995.

[3] Wang Shuai, Ji Yingdong, Yang Shiyuan. Scenario-based modeling method for CTCS-3 train control system [J] · Journal of The China Railway Society, 2011,33（9）.

[4] Yuan Lei, Lv Jidong. Model-Based Test Cases Generation for Onboard System [J] · Journal of The China Railway Society, in press.

[5] Department of railway ministry science and technology · CTCS-3 System Requirement Specification [M] · Beijing: China Railway Publishing House, 2009.

[6] Department of railway ministry science and technology · CTCS-3 System Overall Technical Program [M] · Beijing: China Railway Publishing House, 2009.

[7] Johan Blom, Anders HESSEL, PETTERSSON P. Specifying and Generating Test Cases Using Observer Automata[C] GABOWSKI J, NIELSEN B. Proceedings 4th International Workshop on Formal Approaches to Testing of Software 2004 (FATES,04), Volume 3395 of LNCS. Springer-Verlag, 2005: 125-139

[8] HESSEL A, PETTERSSON P. COVER—A Real-time Test Case Generation Tool [Z]. Accepted for the 3rd Workshop on Model-Based Testing 2007 (MBT07).