Research on Real-Time Testing Modeling Based on RTCM Approach

Ling Li, Ji Wu, Chao Liu and Haiyan Yang

Software Engineering Institute Beihang University Beijing, China

***Abstract*—Test case is used for a particular destination and prepared in a set of test inputs, conditions and expected results in order to test whether a feature is to meet a specific demand [1]. In industrial applications, the description of the test cases is usually in the form of documents, these documents are usually short of a standard format [2]. The other hand, for real time systems, the test case description is more complex. In our work, with the investigation research on the real-time system in industrial field, and the study of the related standard and literature, we extracted out the key elements of the real-time system, constructed the domain model for the important concepts, and then based on the Restricted test case modeling (RTCM) [3], we proposed a real-time test case description method, named as the Real-time Restricted test case modeling (RT-RTCM). RTCM is a test case modeling approach which contains a templates and a group of restriction rules. Based on this approach, we define three templates and proposed new restriction rules, which can support and standardize the description of the real-time test case, while through on the constraints of the natural language, it can also effectively reduce the ambiguity of the test case description. And based on the description method, we proposed an approach about the real-time test cases generation and choice, this method is a combination of the branch coverage criteria of the test flows, the boundary coverage criteria of the test data and the status coverage criteria of the system resource. And in the end of this paper, we modeled the single elevator system, to prove that RT-RTCM is easy to learn, use and understand, and can be an effective technique for real-time test cases modeling.**

Keywords-real-time; test; test case modeling; test generation

I. Introduction

With the rapid development of real-time systems, the testing for real-time systems is becoming more and more important. In order to test the real-time system adequately and accurately, there have been some modeling methods, but it is not mature enough compared with the common functional test modeling methods [4]. This is exactly the necessity and background of our study.

The test object of this study is real-time system. Real-time systems usually have the following characteristic:

(1) Time constraints. The task of real-time system usually has some time constraints which mainly include three aspects. First, the constraints on the point of time, it requires a certain task or operation must occur at a certain time. Second, the time interval constraints, a task must be completed in a certain period of time. The third one is the time sequence constraints, that several steps must be completed in accordance with a certain time sequence.

(2) Concurrency. Real-time system tasks are usually with concurrency, that is, each processing function includes a number of real-time concurrent tasks, and is preemptive scheduled by the operating system according to the priority. The tasks send data to the system, then suspend and wait for the system to response the information. After receiving the feedback information, the tasks lift the suspension and handle the feedback. And this is a complete operating cycle.

(3) Reliability. Most real-time systems require high reliability. Reliability means that the system can work or avoid losses even in the worst case. In some important real-time applications, any unreliable factor or failure of the computer, or over-time response in real-time system, may cause unpredictable serious consequences.

(4) Interaction with the external environment. Real-time systems usually run in some certain environment, the external environment is an integral part of real-time system [5]. Computer subsystem is generally a control system which must respond to external requests within a specified period of time, and external environment is often the controlled subsystem, these two parts interact and form a complete real-time system.

These features above increase the technically challenging of real-time system testing, especially in the test case construction and test output verification, etc., which are the focus of this study.

Thus, the research objectives in this paper are to analyze the characteristics of the real-time system and the problems of real-time test in the industry, extract the important concepts, firstly. And then design a set of templates which can support describing the real-time test requirements. And then based on the templates, define the test cases generation and choice method, with the purpose of covering enough test paths, test data, and system status, and reduce the number of test sets as much as possible at the same time.

The rest of this paper is organized as follows: Section 2, construct the real-time test cases description templates; Section 3, do research on the generation method of real-time test case according to the coverage criteria; Section 4, case studies, illustrate the correctness and effectiveness of research approach through specific cases; Section 5, the conclusion of this paper.

II. Design of a real-time test case description template

Test case is a set of test inputs, execution conditions, and expected results that are prepared for a particular goal in order to test a program path or to verify if a particular requirement has been met or not. A test case usually has one set of inputs and one expected output [6]. The essence of software testing is to confirm a set of test cases for the content to be tested, and it is one of the ways to quantify the test as well. A test case contains information such as the test input, the prerequisites (the environment that existed before the test case was executed), and the expected output. And the test activity is a process as following, first of all, we create the necessary prerequisites, provide the test case input, observe the output, then compare the output with the expected output to determine whether the test passed or not.

In this paper, we select a test case modeling approach named Restricted Test Case Modeling (RTCM) approach. And the RTCM is based on another modeling approach called Restricted Use Case Modeling (RUCM) [7]. They both have a template and a set of restriction rules, which can help reduce imprecision and incompleteness in Use Case Specification and Test Case Specification and reduce ambiguity of models. These two methods have proved easy to apply and can achieve better understandability of cases [8]. By doing research on the real-time system testing method, this paper models the key concepts of real-time system testing by combining the standard and literature research. And based on the RTCM approach, propose a new approach named Real-Time Restricted Test Case Modeling (RT-RTCM) to support the description of real-time test cases, which can reduce the ambiguity of the test case description and retain the advantages of the natural language that is easy to understand and easy to use relying on the test specification template and the restriction rules.

A. *The Description of Real-Time Testing Requirements*

Specification is a common technique to describe testing requirements; each case usually corresponds to a specification, which contains basic information, a basic flow of events and multiple alternative flows as well as a few test assertions [9]. In this study, on the basis of the test case specifications, we presented an improved real-time testing specification description method, which contains a test case description template to support describing real-time testing requirements, and contains a set of restrictive rules, which limit the use of natural language. The structure of the template is shown in the following table, it consists of five parts:

TABLE I. THE TEMPLATE OF REAL-TIME TEST REQUIREMENT DESCRIPTION

Name	The name of the test case	
Brief Description	The description of the content	
Tester	The tester of the test case	
Dependecy	The Dependency of other test case	
Period	The period of the test case	
ImportDataDic	Import a data dictionary	
ImportResource	Import a resource configuration	
Baisc Flow	The main test flows	
	Steps	Test sequences
	Test Oracle	Post condition
Specific alternative flow	The specific alternative test flows	
	RFS	Referenced flow step
	Steps	Test sequences
	Test Oracle	Post condition
Bounded alternative flow	The Bounded alternative test flows	
	RFS	Referenced flow step
	Steps	Test sequences
	Test Oracle	Post condition
Global alternative flow	The Global alternative test flows	
	RFS	Referenced flow step
	Steps	Test sequences
	Test Oracle	Post condition
Oracle Verification Flows	The Oracle Verification Flows	
	RFS	Referenced flow step
	Steps	Test sequences
	Test Oracle	Post condition
Timepoint Verification Flows	The Time-point Verification Flows	
	RFS	Referenced flow step
	Steps	Test sequences
	Test Oracle	Post condition
Duration Verification Flows	The Duration Verification Flows	
	RFS	Referenced flow step
	Steps	Test sequences
	Test Oracle	Post condition
Timing sequences Verification Flows	The sequences Verification Flows	
	RFS	Referenced flow step
	Steps	Test sequences
	Test Oracle	Post condition
Timing Constraints		
Name	**RFS**	**Constraints**

1) Basic information. Including the name of the test case (Name) that usually corresponds to the name of the function; brief description (Brief Description) that describe the main behavior and test target; test participants (Tester), for real-time system, test participants are usually contains of three types, human tester, external system and timer; the other test case that relies on (Dependency), it contains of other child function or other function that has interactive relationship; Period

information (Period), in real-time system many functions are periodic, the field is used to describe the periodic characteristics, period usually has a specific value; data dictionary (ImportDataDic), Data dictionary describe the data information of the test case, mainly contains input parameter and time constraints information, the define of the data dictionary will be introduced in the next section; resources configuration (ImportResource), resources configuration describe the system status, mainly including CPU status, memory status, IO status and network, the resources configuration will be introduced in another section below.

2) Basic test flows. Basic test flows are a series of normal behavior. Each function corresponds to a basic test flow usually, and the flows describe the behavior sequences when the function executes normally.

3) Alternative test flows. Alternative test flows represent the different possibility of the behaviors, there are often one or more alternative test flows. Each one is another possibility of the basic flows, it will execute in the specific condition. In this work, alternative test flows are divided into specific test flow, bounded test flows and global test flows.

4) Test oracle flows. For every test case, there is an output at least. The output may be a data or an action. The test oracle flows are used to judge whether the actual output is the same as the expected output. For the real-time system, besides the functional output, the time constraints are also need to be verified. As described earlier, for the time constrains of the real-time in our work, we focus mainly on the time-point、time duration and time sequences, in the test oracle flows, they execute the time-point verification flows, time duration verification flows and time sequences flows respectively.

5) Timing Constraints. Describe the time-related constraint rules. The RFS keyword points to a test step and constraint describe the specific timing constraint by a general expression.

B. *The Template of the Data Dictionary*

There are always large amounts of data transmission in the real-time system; meanwhile, there are also a lot of data transmission in the process of interaction with other equipment and systems [10]. For the non-real-time systems, the input of the test case usually includes the input data or an event only, but for the real-time systems, the input also includes the time elements, the time elements maybe a time-point, time duration or even an event with time.

In our work, we introduced a data dictionary to describe the data in the process of test. The template is shown in TABLE II.

Each test case is responded to one or more data dictionaries. The name of the data dictionary is the value of the ImportDataDic field in the real-time test case template. There is a point to be highlighted, that each line in the Data Definition represents a sentence of data definition, and the data name is the name of the data. The data type has three different values, numeric type, time-point type and time duration type. And the scope of the value represents the domain of the data.

TABLE II. THE TEMPLATE OF THE DATA DICTIONARY

Name	The name of the data dictionary		
Descrption	The brief description of the data dictionary		
Dependency	The dependency on other dictionaries		
Data Defination	**DataName**	**Data Type**	**Value Range**

C. *The Template of the Resource Configuration*

The situation that there is a lot of interaction with the external environment is an important feature of the real-time systems; the external environment usually has important implications for the performance of real time [11]. Different combinations of system resources represents the status of the environment in which the system under test. Through the combination of the resource, we can construct test under the different combination of load. Meanwhile, the worst case combination of resources can also test the reliability of the real- time system.

TABLE III. THE TEMPLATE OF THE RESOURCE CONFIGURATION

Name	The name of the resource configuration	
Disk	**capacity**	The capacity of the disk
	Status	The occupancy status of the disk
Memory	**capacity**	The capacity of the disk of the disk
	Status	The occupancy status
CPU	**kernel number**	The kernel number of the CPU
	frequency	The frequency of the CPU
	type	The type of the CPU
	status	The occupancy status of the CPU

Tester has to fill in the specific resource information in the resource configuration template. The information includes the resource attribute, such as the capacity of the memory and disk, the frequency and kernel number of the CPU and it also includes the resource status that needed to test, such as the consumption of the CPU. And one thing to be stressed is that the resource configuration template is customizable, the users can add new option according to their needs.

D. *The Restriction Rules*

RTCM restricts the use of natural language by restriction rules [3]. There are two types of restriction rules: one is to restrict the expression of the natural language, the other is to use keywords to describe the specific action or process. The main purpose of the definition of restriction rules is to reduce the ambiguity of natural language, while retaining the advantages of easy to use and understand.

Aimed at the key points of real-time testing, this paper introduced new keywords.

TABLE IV. THE TEMPLATE OF THE DATA DICTIONARY

Keyword	Description
CREATE TIMER	Create a timer
START TIMER	Start a timer
PAUSE TIMER	Pause a timer
RESET TIMER	Reset a timer
READ TIMER	Read time from timer
STOP TIMER	Stop timer
SEND INPUT	A system send data to SUT
COLLECT INPUT	SUT collect data from another system
DELIVER OUTPUT	SUT deliver data to another system
VIA COMMUNICATION	Transfer data by a specific media
AT/BEFORE/AFTER	Describe the time point relation
WITHEN	Describe the time duration

III. GENERATION AND CHOICE OF THE REAL-TIMETEST CASES

In this work, based on the method of the real-time test case description, we proposed a method about the real-time test cases generation and choice, this method is a combination of the branch coverage criteria of the test flows, the boundary coverage criteria of the test data and the status coverage criteria of the system resource [11].

1) Testers have to mark the important test steps.

2) According to the original rule of branch coverage, the test scenario is generated based on the basic test flow and the branch test flow in the test description template. A test scenario is an instance of a use case that represents the execution path of a use case. In our study, the basic test flow corresponds to the first test scenario, and the combination of each basic test flow and the branch test flow corresponds to one test scenario, respectively. At the same time, if the test scenario contains user annotations, then mark the scene as an important test scenario [12].

3) Introduce the test data and resource status, and carry out the combination test.

a) Capture parameters from each step of the test case, including common data types and time types. According to the data definition in the data dictionary, find the value range of the parameters, respectively, and then take the correct value, error value, and the border value to generate a parameter value table.

b) The same as the parameters, for each option of the resource configuration, according to the user's configuration, take the boundary value, and the out-of-range value, and add then to the above-mentioned parameter value selection table.

c) Theoretically, in accordance with the principle of single variable control, we can generate test cases using the parameters and resources introduced above. However, in practice, we should create a minimal set of test cases to cover them, to avoid the waste of resources in the testing process [13]. In our study, we use the method of depth-first search for test case selection. When creating the first test case parameter combination, select any value of the first parameter, then select any value of the second parameter, and then select any of the third parameter value, and so on, until the last parameter, thus generating the first test case parameter combination. Next create a second test case parameter combination, the method is as this, in order to generate the second test case parameter combination, for every parameter, choose the value which the first test case is not used. And so on until all the optional values are covered.

4) Carry out testing according to the test cases. For different test scenario with different importance, carry out different testing ways. For the non-critical test path, select a group of test data to test, and for critical scenarios, each test data has to be tested.

5) Verify the test output [14]. According to the test case description of the Oracle Verification Flows, Time verification Flows, check whether the actual output and the expected output is consistent, and whether the time constraints are met.

IV. EVALUATION

In order to illustrate the effectiveness of the method, in our work, we choose a single elevator system (SE) case as verification. The single elevator system consists of a vertical lift and corresponding equipment on each floor. Elevator interior includes an open button and a close button, as well as floor buttons. There is also a screen used to display the current floor of the elevator. Outside the elevator doors of each floor, there is a request up button and a request down button, as well as an up down indicator light.

The real-time requirements of the single elevator system are as follows:

1) If the passenger presses the door open button (or automatically detects an obstacle) during the closing process, the elevator door must stop the closing process within 0.15 s and restart.

2) In the process of running, if the lift sensor needs to be parked on the current floor after the floor sensor signal is updated, the elevator engine must start to decelerate within 0.5 seconds and stop the elevator completely within 3 seconds.

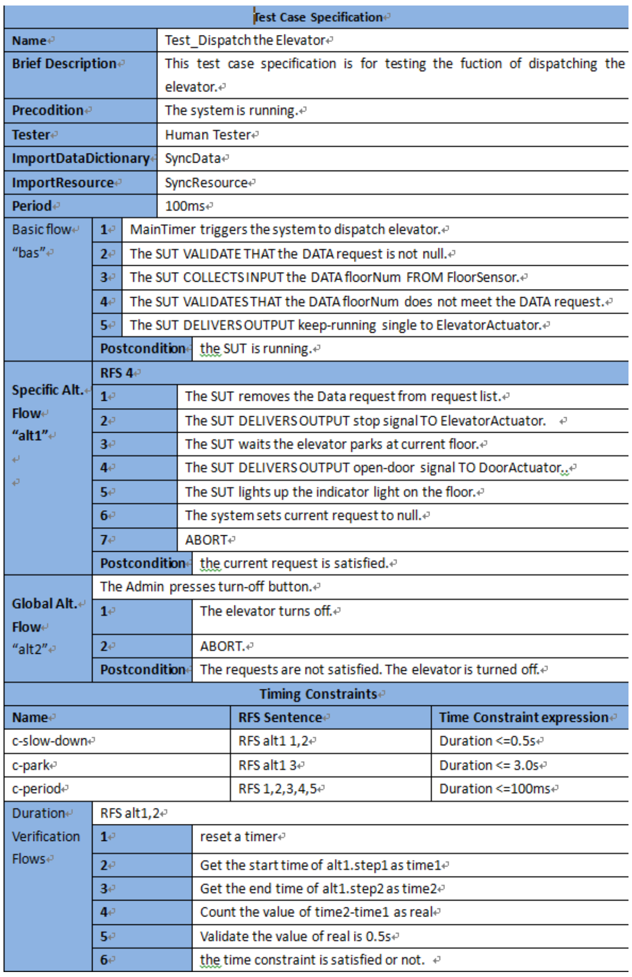
3) If the elevator is overloaded, the alarm must sound within 0.5 s and the elevator door must stop within 0.15 s.

4) After the passenger presses the floor request button or the up and down request button, the system should store the request in the request list within 0.4 s.

5) The system operates in a periodic task with a period of 100ms.

And we choose the elevator scheduling function to illustrate in this paper.

1) First, describe the elevator test case by using the real-time test requirement template.



Test Case Specification			
Name	Test_Dispatch the Elevator		
Brief Description	This test case specification is for testing the fuction of dispatching the elevator.		
Precodition	The system is running.		
Tester	Human Tester		
ImportDataDictionary	SyncData		
ImportResource	SyncResource		
Period	100ms		
Basic flow "bas"	**1**	MainTimer triggers the system to dispatch elevator.	
	2	The SUT VALIDATE THAT the DATA request is not null.	
	3	The SUT COLLECTS INPUT the DATA floorNum FROM FloorSensor.	
	4	The SUT VALIDATES THAT the DATA floorNum does not meet the DATA request.	
	5	The SUT DELIVERS OUTPUT keep-running single to ElevatorActuator.	
	Postcondition	the SUT is running.	
Specific Alt. Flow "alt1"	**RFS 4**		
	1	The SUT removes the Data request from request list.	
	2	The SUT DELIVERS OUTPUT stop signal TO ElevatorActuator.	
	3	The SUT waits the elevator parks at current floor.	
	4	The SUT DELIVERS OUTPUT open-door signal TO DoorActuator..	
	5	The SUT lights up the indicator light on the floor.	
	6	The system sets current request to null.	
	7	ABORT	
	Postcondition	the current request is satisfied.	
Global Alt. Flow "alt2"	The Admin presses turn-off button.		
	1	The elevator turns off.	
	2	ABORT.	
	Postcondition	The requests are not satisfied. The elevator is turned off.	
Timing Constraints			
Name	**RFS Sentence**	**Time Constraint expression**	
c-slow-down	RFS alt1 1,2	Duration <=0.5s	
c-park	RFS alt1 3	Duration <= 3.0s	
c-period	RFS 1,2,3,4,5	Duration <=100ms	
Duration Verification Flows	RFS alt1,2		
	1	reset a timer	
	2	Get the start time of alt1.step1 as time1	
	3	Get the end time of alt1.step2 as time2	
	4	Count the value of time2-time1 as real	
	5	Validate the value of real is 0.5s	
	6	the time constraint is satisfied or not.	

FIGURE I. DESCRIPTION OF THE ELEVATOR SCHEDULING TEST CASE

2) Define the parameters involved in the test case using the data dictionary, and the parameters are specified in the test case template using the "The Data" keyword.

TABLE V. THE DATA DICTIONARY OF THE TEST CASE

Name	SyncData		
Descrption	describe the data in the test case of Test_Dispatch the Elevator		
Dependency	None		
Data Defination	**DataName**	**Data Type**	**Value Range**
	Request	Numeric	[1,10]
	FloorNum	Numeric	[1,10]

3) Configure the resource status using the resource configuration template.

TABLE VI. THE RESOURCE CONFIGURATION OF THE TEST CASE

Name	SyncResource	
Disk	**capacity**	1T
	Status	[30%,90%]
Memory	**capacity**	4G
	Status	[20%,90%]
CPU	**kernel number**	4
	frequency	3.00MHz
	type	Intel
	status	[20%,90%]

4) Mark the important steps of the test case. The above steps complete the description of the test requirements, and then we have to mark the important steps firstly. Through analyzing the flows in the test case, we marked bas.step5 and alt1.step4 as the key step.

5) Generate the abstract test scenario. According to the criterion of conditional coverage, this function module has one basic flow and two branch flow. Therefore, three test scenarios are generated, named bas, alt1 and alt2. Because there both have a step marked as key step in bas and alt1, the two scenarios are critical scenarios.

6) Combine the test data and environmental status, and list the optional parameters in the table below.

TABLE VII. THE OPTIONAL PARAMETERS TABLE

Request	**FloorNum**	**Disk**	**CPU**	**Memory**
1	**1**	**30%**	**30%**	**30%**
5	**6**	**50%**	**60%**	**80%**
10	**10**	**90%**	**90%**	**90%**
11	**12**	**99%**	**92%**	**100%**

Depending on the depth-first criterion, four random combinations of parameters are generated.

7) Use the parameters' combination obtained in 6) to configure the system environment and assign values to the parameters, and then test the elevator scheduling function according to the test scenarios generated in 5). Select a combination randomly to test the alt2, and all the parameters combinations have to be test in the test scenario of bas and alt1.

8) Verify the test results. We have to verify whether the real output data is consistent with the expected value and whether the three time constraints are met.

In summary, in the elevator scheduling function, 9 test cases were tested in total.

In the whole case of the single elevator system, six cases were built, and the test cases were generated according to the above steps. The specific data is shown in the following table.

TABLE VIII. THE SINGLE ELEVATOR SYSTEM

Name	**Test scenarios**	**Main scenarios**	**Data combinations**	**Test Cases**
Turn on Elevator	2	1	4	5
Turn off Elevator	2	1	4	5
Dispatch the Elevator	3	2	4	9
Open Elevator Door	2	0	4	2
Go To a Floor	3	2	4	9
Go Down a Floor	3	2	4	9

The case study implied that the RT-RTCM method can accurately and comprehensively describe the test cases, and can effectively avoid the ambiguity of natural language. At the same time, it can automatically generate test cases according to the test strategy, and reduce the workload of testing.

V. CONCLUSION

By analyzing the characteristics of real-time system and combining with the problems of real-time system testing in practice, this paper proposed a model-based real-time test modeling method. Firstly, this paper defined three templates to describe the real-time test cases, the parameters involved in the test cases, and the system environment in which the test cases are located. And then, based on the description template, this paper proposed a test case generation and selection method. This method references the path coverage criteria, the test data coverage criteria and the system status coverage criteria, so that the generated test cases can cover all the test paths and test data. And by marking the key paths, dominating the test data based on the depth-first method, the test case set is selected to reduce the workload of the test.

At the end of the paper, the method proposed in this paper is illustrated and verified by modeling the single elevator system. The results of the case analysis show that the method can effectively capture the test cases of real-time systems, and is easy to learn, understand and use.

ACKNOWLEDGMENT

This work is supported by the Technology and Industry for National Defense under Grant No.JSZL2014601B008.

REFERENCES

[1] ISO/IEC/IEEE 29119 -1: Software and systems engineering- Software testing Part 1: Concepts and definitions[S]. ISO/IEC/IEEE, 2013.

[2] DO178B/C: Software Considerations in Airborne Systems and Equiment Certification[S]. RTCA, 1982-2012.

[3] Zhang, M.,Yue, T., and Ali S.: A Keyword and Restricted NL Based TCS Language for Auto-mated Testing. Simula Research Laboratory, Technical Report (2014-01).

[4] The UML Profile for MARTE: Modeling and Analysis of Real-Time and Embedded Systems[EB/OL]. http://www.omgmarte.org

[5] L Lavazza, G Quaroni, M Ventureli. Combining UML and Formal Notations for Modeling Real-Time Systems[A]. ProcJoint 8th ESEC and 9th ACM SIGSOFT FSE[C].2001. 196-206

[6] Beizer B. Black-Box: Testing: Techniques for Functional Testing of Software and Systems, Wiley, New York, USA, 1995.

[7] Yue, T., Briand, L. C. and Labiche, Y.: Facilitating the Transition from Use Case Models to Anal-ysis Models: Approach and Experiments. Transactions on Software Engineering and Methodol-ogy (TOSEM), vol. 22(2011)

[8] Zhang,G.,Yue,T., and Ali,S.: Modeling Crisis Management System with the Restricted Use Case Modeling Approach, In: Comparing Modeling Approaches(CMA) Workshop at ACM/IEEE 16th International Conference on Model Driven Engineering Languages and Systems(MODELS), ed. By Jeff Gray, ACM/IEEE(2013).

[9] Yue Tao, Briand L.C., Labiche Y. A use case modeling approach to facilitate the transition towards analysis models: Concepts and empirical evaluation[M]. Model Driven Engineering Languages and Systems. Springer Berlin Heidelberg, 2009: 484-498

[10] Weyuker E, Goradia T, Singh A. Automatically generating test data from a Boolean specification[J]. Software Engineering, IEEE Transactions on, 1994, 20(5): 353-363.

[11] ETSI ES 202 782: Methods for Testing and Specification(MTS); The Testing and Test Control Notation version 3; TTCN-3 Language Extensions: TTCN Performance and Real Time Testing[S]. ETSI, 2010,07

[12] Donat M R. Automating formal specification-based testing[M]//TAPSOFT'97: Theory and Practice of Software Development. Springer Berlin Heidelberg, 1997: 833-847.

[13] Hall P A V. Relationship between specifications and testing. Information and Software Technology, 1991, 33(1):47-52

[14] Cabral G, Tamai T. Requirement-based testing through formal methods[J]. Proceedings of TESTCOM-FATES, 2008