Software redundancy design for a Human-Machine Interface in railway vehicles

G. Zheng[1] & J. Chen[1,2]
[1]*Institute of Software, Chinese Academy of Sciences, China*
[2]*Graduate University of Chinese Academy of Sciences, China*

Abstract

The Human-Machine Interface (HMI), which displays the real-time status of electrical systems, interacts with the driver or operator, and collects and reports system fault information, is an important device in railway vehicles. The HMI is a critical component of the control and diagnosis system in the railway vehicle, thus the reliability of the HMI software affects the reliability and safety of the whole railway vehicle. Therefore, it is necessary to design the HMI software with high reliability for railway vehicles so as to ensure the reliability, stability and safety of the railway vehicle operation. This paper analyzes the HMI software function requirements, which include information display, the human-machine interaction, and communication. A kind of redundancy mechanism is proposed, which employs two structural redundancy methods: N-version programming and recovery blocks. The HMI software is divided into the information display module, the human-machine interaction module and the communication module, and each module is made up of some components. Based on the analysis of the reliability requirement, complexity, and the implementation cost for each component in the HMI software modules, the corresponding redundancy design mechanism is proposed, which consider the tradeoff between the reliability and the cost. In order to evaluate the reliability of the designed redundancy mechanism, a scenario-based reliability analysis method is used to calculate the reliability of the HMI software, which constructs five scenarios and employs the component dependency graph to compute the reliability. The reliability of the HMI software after redundancy design is compared with that before the redundancy design.

Keywords: human-machine interface, reliability, software fault tolerance, redundancy design, reliability analysis.

1 Introduction

A Human-Machine Interface (HMI) in the driver cab is an important device for a railway driver to interact with the railway vehicle, and also an integral part of the vehicle control system. During the vehicle operation, the driver can monitor the state of the vehicle in real time and send control messages to ensure safety. As a whole, the HMI executes operation states information display, human-machine interaction and communication with other electrical devices in the vehicle. In order to ensure the safety and stability of the railway vehicle operation, it is necessary to design highly reliable software in the human-machine interface for railway vehicles.

At present, there are mainly the following methods for the software reliability: error avoidance, error detection and correction, and fault tolerance. Error avoidance employs the standardization design and coding process to reduce software errors. Error detection [1] discovers software errors by setting checkpoints in the software program, moreover, some techniques are used to isolate and correct the errors [2]. Note that it is very difficult to completely ensure software does not have any errors. Fault tolerance [3] is currently a valid technique to improve the reliability of the computer software, which can detect faults automatically and execute the corresponding fault tolerance program. The structural redundancy technique is used commonly in software fault tolerance, which generally includes N-version programming (NVP) and the recovery block technique (RCB) [4]. In the N-version programming technique, N software versions (N>1) are developed independently and work simultaneously after being installed in the same environment, where N versions accept the same input, and the final output is selected from the N outputs by a majority voting algorithm. In the recovery block technique, several recovery blocks are developed for the same software function, where each recovery block accepts the same input and gives an output, and the output is the input of the acceptance test unit. If the output passes the test, the software continues to run, else the software environment is restored, and then the other recovery blocks repeat the above process until a valid result is accepted or there are no other recovery blocks.

Considering the functions and the safety requirements of the HMI, several different redundancy mechanisms are employed to improve the software reliability. When developing the HMI software, it is necessary to select different redundancy mechanisms, e.g., N-version programming or recovery block technique, for different function components in terms of the software complexity and cost to implement structural redundancy. After completing the software redundancy design, the reliability of the software is evaluated. In this paper, a scenario-based analysis technique [5] is employed to evaluate the reliability result of a component-based application in the HMI software.

The rest of the paper is organized as follows. The components function of the HMI software is introduced in section 2, and the redundancy design is presented in section 3; next, the reliability evaluation of the HMI software is given based on a scenario and, finally, the conclusion is drawn in section 5.

2 Functions of HMI software in railway vehicles

The HMI is connected to the electrical systems over the vehicle communication bus, as shown in the Fig. 1. The HMI monitors the operation status of the vehicle, displays the fault diagnosis results and receives the driver inputs and gives the associate responses. The HMI software functions are given as follows.

(1) Information display: displays the status of electrical systems, faults diagnosis results and fault recovery information;

(2) Human-machine interaction: gives responses to the operation of the driver on the HMI screen to transit the interfaces, input data, and send control instructions;

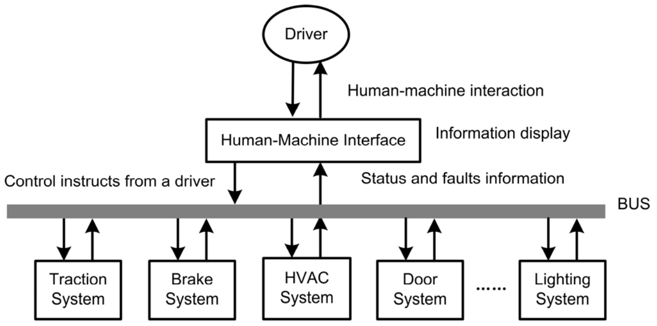

Figure 1: Connection between the HMI and other electrical systems.

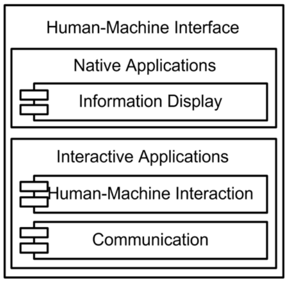

Figure 2: Structure of HMI the software.

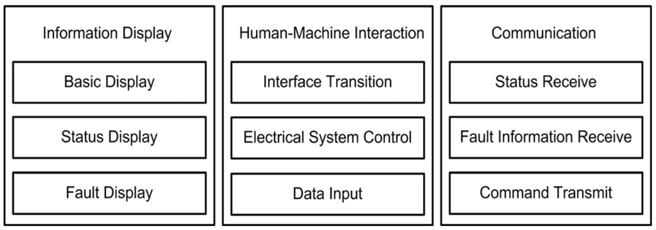

Figure 3: Sub-functions of the HMI software.

(3) Communication: communicates with control units in the vehicle, input/output devices and other electrical equipments to transmit message data and state data over the vehicle bus.

The structure of the HMI software is shown in the Fig. 2 in terms of the HMI function. Each function is divided into some sub-functions so as to implement the redundancy design, please see Fig. 3.

The sub-functions of the information display are presented as follows:

(1) Basic display: displays interface name, line, date, time, vehicle number and other simple information.
(2) Status display: displays status information of the corresponding electrical system according to the requirement of the driver.
(3) Fault display: displays fault information when an electrical system error happens.

The sub-functions of the human-machine interaction are presented as follows:

(1) Interface transition: implements the corresponding interface transition when the driver presses the transition key.
(2) Electrical system control: calls the communication module to send control commands when the driver presses the control key.
(3) Data input: responds to input information from the driver, such as vehicle number, driver number, system time.

The sub-functions of the communication are presented as follows:

(1) Status receive: receives real-time status information of all electrical systems.
(2) Fault information receive: receives the fault information and then transfers it to the appropriate processing.
(3) Command transmit: sends control commands to the appropriate electrical system.

3 Software redundancy design for a human-machine interface

3.1 Analysis on the compromise between costs and reliability

The HMI software is divided into three modules corresponding to the functions, where each module is made up of some components. The structure of a component is more compact than the one of a module and contains some similar features, which can employ the same redundancy design technique. Therefore, the redundancy design of the HMI software is based on the structure of the components.

In terms of the functions of the human-machine interface software in railway vehicles, two structural redundancy methods, N-version programming and recovery blocks, are employed to improve the reliability. When designing the software redundancy, the reliability requirements, the complexity of the various function components and implementation methods for structural redundancy are considered.

(1) Reliability requirements

High reliability is demanded on some components that have a direct effect on the safe and stable operation of the train, and some assistant function components demand relatively low reliability. According to the standard EN50126 [6], the whole HMI software has a reliability requirement.

(2) Complexity of components

A simple structure component can achieve a high reliability before the redundancy, so there is no need to implement redundancy design on it. Meanwhile a complex component requires redundancy design to improve its reliability. The complexity of a component is considered to determine the need for redundancy design.

(3) Implementation costs

N-version programming requires N teams to complete the same function component, at the same time; recovery block technique requires several blocks with the same function, which leads to more development cost. N-version programming makes a selection among several outputs. Numerical value may facilitate carries on the selection, while some display functions are unable to make selection. An acceptance test unit in recovery block technique is used to test result. The result of a number of function components cannot be used for testing, so that these components cannot use the recovery block technique. Because some acceptance units are difficult to write, a compromise should carry on between the reliability and the cost.

3.2 Software redundancy design

The software redundancy design for three modules, i.e., information display, human-machine interaction and communication, is presented as follows.

(1) Information Display

The information display module includes three components: basic display, status display, and fault display.

Because of the simple structure, basic display component can easily obtain high reliability. The operation of the vehicle will not be influenced even if the basic information is displayed inaccurately, so high reliability is not demanded on this component. This component does not need redundancy design.

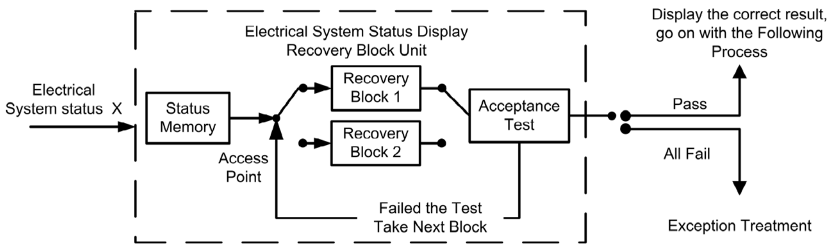
Electrical System Status Display
Recovery Block Unit
Electrical System status X
Status Memory
Access Point
Recovery Block 1
Recovery Block 2
Acceptance Test
Failed the Test Take Next Block
Display the correct result, go on with the Following Process
Pass
All Fail
Exception Treatment

Figure 4: Redundancy design structure of the status display component.

Status display component receives the electrical system status, processes and displays the status information on the screen. Since this component has many kinds of status to process, and each kind of status has a variety of forms, this component is highly complex. Whether the status of electrical systems is good or not affects the operation of the railway vehicle, the HMI can reflect the operation status of electrical systems in real time; therefore, the recovery block technique is employed in this component. Fig. 4 shows the redundancy design structure:

According to the process showed in fig. 4, before entering the recovery bock, current electrical status should be stored by memory unit. For the first time, the primary recovery block is chosen to figure out a numerical result. If the acceptance test unit accepts the result, the result is displayed and the following process is executed. If the acceptance test unit does not accept the result, the component returns the access point of recovery blocks and chooses another recovery block. If the results do not pass the acceptance test, this component throws an exception and executes the exception treatment, which means the component has broken down.

The fault display component has a high reliability requirement as it shows the operation status of the electrical systems. This information reminds the driver to response to a fault. Before the new fault is diagnosed, the component can query the diagnosed faults. The 3-version programming (3VP) is employed to improve the reliability of the component. Three versions receive the same error number as their own inputs and figure out the display result. The final result is obtained based on the majority voting algorithm [7], which is chosen among the results of three versions. Fig. 5 shows the redundancy structure with three versions.

(2) Human-Machine Interaction

The human-machine interaction module consists of three components: interface transition, electrical systems control, and data input.

The information cannot be displayed if the interface transition component breaks down, which could result in the entire vehicle out of control. Therefore, the interface transition component is required with high reliability. However, the number of the interfaces, which interfaces in the HMI device can switch to, is limit, thus the component should be designed with low complexity so that the reliability requirements can be satisfied easily. The cost to multiple recovery blocks is low because the structure of the interface transition component is not complex. The algorithm of acceptance test unit is described as follows: 1) read

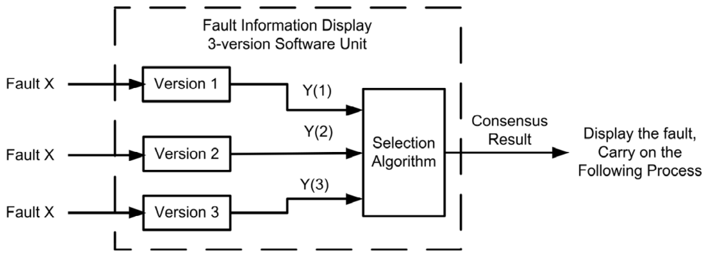

Figure 5: Redundancy structure of the fault information display component.

the name of the interface which is switched to; 2) judge whether the name is the same as the one the pressed soft key corresponds to; 3) If two names are not the same, the program return to the access point and choose another recovery block. Note that the algorithm is easy to implement.

It is so important for the driver to control the electrical systems via the HMI screen, which is relative with the reliability and stability of the vehicle operation, and even the comfortable level of the passenger. Therefore, the reliability requirement for the electrical system control component is so high. Meanwhile, the complexity of this component is high because there are so many electrical systems to control. This component employs 3-version programming to improve the reliability of this component. The process is the same as the one of the fault display component. When the driver presses the soft key to control the electrical system, three versions receive the same input, and each version sends a control instruction to the electrical system. The selection unit in the component receives these three control instruction, decides which instruction is sent.

If the data input component fails, the input, *e.g.*, the vehicle number, is not consistent with the last saved results, but the inconsistence has little effect on the vehicle operation, thus, the reliability requirement of this component is low. Meanwhile, the complexity of the component is not high because the data, which the driver can enter, is so limited. Therefore, it is not necessary to implement redundancy design.

(3) Communication

The information display module consists of three components: status receive, fault information receive, command transmit.

Status receive component is very similar to fault information receive component. They both receive important information which directly reflects the

Table 1: Redundancy design of the HMI function components.

Function Module	Component	Reliability Requirement	Complexity	cost of redundancy design	Redundancy design
Information Display	Basic Display	low	low	--	--
	Status Display	high	high	high	RCB
	Fault Display	high	medium	medium	3VP
Human-Machine Interaction	Interface Transition	high	medium	medium	RCB
	Electrical System Control	high	high	high	3VP
	Data Input	low	medium	--	--
Commun-ication	Status Receive	high	high	medium	RCB
	Fault Information Receive	high	high	medium	3VP
	Command Transmit	high	medium	medium	RCB

operation status of the vehicle .Because of the large number of electrical systems and devices, they both have much information to deal with. Thus, the reliability requirements and complexity of these two components are high. 3-version programming is employed in these two components to avoid data loss and ensure accuracy of the information.

The failure of the command transmit component can cause that electrical systems are out of control, which is so dangerous. This component has medium complexity because the process of information transmission is not complex. A recovery block technique is employed to ensure that control command is transmitted normally.

Table 1 shows the components redundancy design on the HMI software.

4 Software reliability analysis

The effect of redundancy design is measured by means of assessment of HMI software reliability. The reliability is estimated using Scenario-Based Reliability Analysis (SBRA) [5], which is specific for component-based software whose analysis is strictly based on execution scenarios. In this paper, the reliability of HMI software after redundancy design is compared with the one before redundancy design.

SBRA consists of three steps:

(1) Usage of scenarios to analyze the dynamic behaviour of HMI software, construct a sequence diagram to each scenario.

(2) Calculate some parameters using the sequence diagrams, construct a component dependency graph (CDG) using these parameters.

(3) After constructing the CDG model, use an algorithm [5] to analyze the reliability of HMI software.

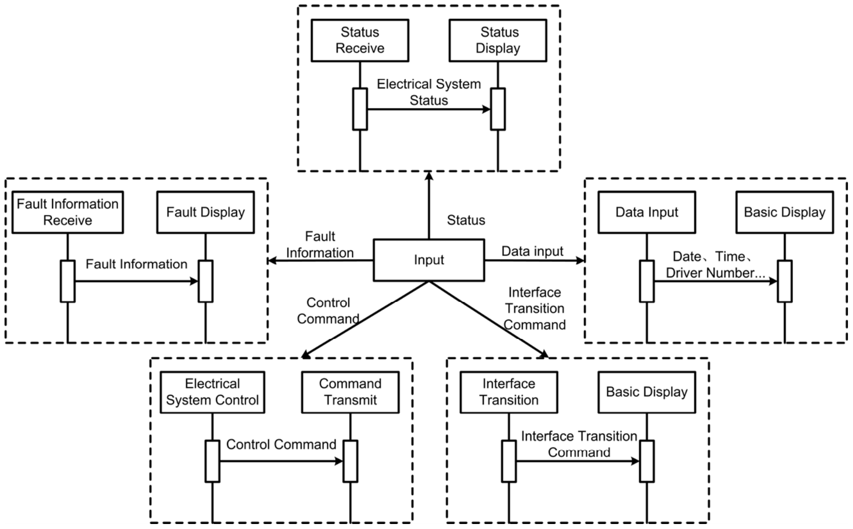

Figure 6: Scenarios of the HMI software.

Table 2: Parameters of each scenario.

Scenario Name	Probability of a Scenario (PS_k)	Average Execution Time of a Scenario (EC_i)
Status display	0.85	3
Fault display	0.03	3
Switch	0.05	11
Input	0.02	2
Control	0.05	3

4.1 Construction of scenarios

There are two types of input can stimulate HMI. One is the data receiving from the other system: electrical system status and fault information; the other is the driver input: interface switching input, data input, and electrical system control command. Based on these inputs, five scenarios can be constructed to describe the interactions between components. Fig. 6 shows the five scenarios.

4.2 Component dependency graph construction

Let S_k be an element of the application scenarios set S , $k=1,...,|S|$, where $|S|$ is the number of the set S. The probability of the kth scenarios PS_k is calculated based on the implementation of HMI software, the probabilities of execution of the 5 scenarios are listed in the following table.

Let RC_i be reliability of the ith component in the HMI software, i=1,…,9. RC_i is calculated based on the one of a single version/ recovery block. Suppose that the reliability of each version is r, the reliability of a component that has implemented 3-version programming is calculated by eqn (1).

$$RC_i = r^3 + 3 * r^2 * (1-r) \tag{1}$$

Suppose that the reliability of each recovery block is r_b, the reliability of acceptance test unit is r_a , the reliability RC_j of a component implementing recovery block technique is calculated by eqn (2):

$$RC_j = r_b * r_a + (1-r_b) * r_b * r_a \tag{2}$$

where j=1,…,9.

The reliability result of each component is shown in table 3.

Let RT_{ij} be a reliability estimate of a transition from component C_i to component C_j. In order to simplify the analysis, supposes the reliability of the interface is 1. Let EC_i be the average execution time of the ith component, i=1,…,9,

$$EC_i = \sum_{k=1}^{|S|} PS_k \cdot Time(C_i) \tag{3}$$

where PS_k is the execution probability of the kth scenario, $k = 1,...,|S|$, $Time(C_i)$ is the execution time of component C_i. The average execution time is shown in table 3.

Table 3: Parameters of each component.

Component Name	Component Reliability before redundancy design	Component Reliability after redundancy design	the average execution time of each component EC_i
Basic Display	0.95	0.95	0.52
Status Display	0.9	0.98	0.85
Fault Display	0.92	0.982	0.03
Interface Transition	0.92	0.984	0.05
Electrical System Control	0.9	0.972	0.05
Data Input	0.92	0.92	0.02
Status Receive	0.9	0.972	1.7
Fault Information Receive	0.9	0.972	0.06
Command Transmit	0.92	0.984	0.1

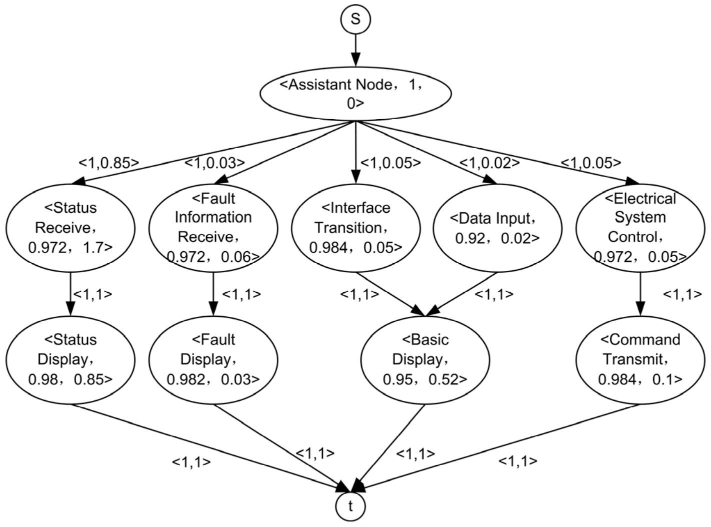
S
<Assistant Node，1，0>
<1,0.85>
<1,0.03>
<1,0.05>
<1,0.02>
<1,0.05>
<Status Receive，0.972，1.7>
<Fault Information Receive，0.972，0.06>
<Interface Transition，0.984，0.05>
<Data Input，0.92，0.02>
<Electrical System Control，0.972，0.05>
<1,1>
<Status Display，0.98，0.85>
<Fault Display，0.982，0.03>
<Basic Display，0.95，0.52>
<Command Transmit，0.984，0.1>
t

Figure 7: CDG of HMI software.

Let AE_{appl} be the average execution time of the HMI software,

$$AE_{\text{appl}} = \sum_{k=1}^{|S|} PS_k \cdot Time(S_k) \tag{4}$$

where $Time(Sk)$ is the average execution time of scenario S_k .

Based on eqn (4), the average execution time of the HMI software is 3.38.

The transition probability between components RT_{ij} is obtained based on the analysis of each scenario. The component dependency graph of the HMI software is shown in Fig. 7.

4.3 Reliability analysis

Based on the scenario-based reliability analysis algorithm [5], the construction process of reliability is shown in Fig. 8.

The reliability of the HMI software after redundancy design is 0.95035352.

Follow the above steps, the reliability of HMI software before redundancy design is 0.81572.

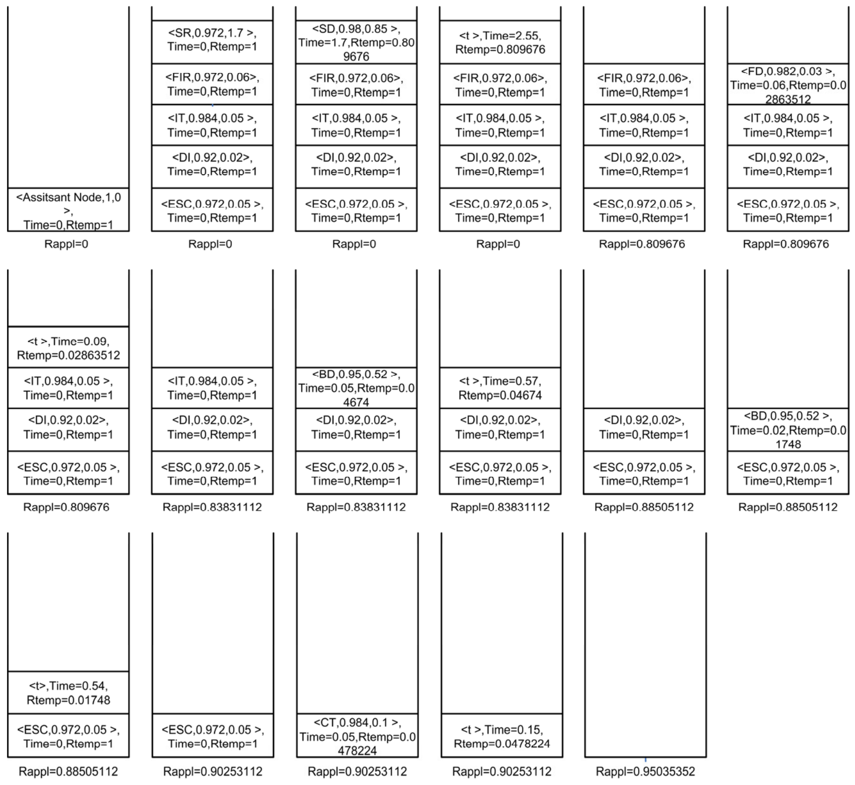
<Assitsant Node,1,0 >, Time=0,Rtemp=1
Rappl=0
<SR,0.972,1.7 >, Time=0,Rtemp=1
<FIR,0.972,0.06>, Time=0,Rtemp=1
<IT,0.984,0.05 >, Time=0,Rtemp=1
<DI,0.92,0.02>, Time=0,Rtemp=1
<ESC,0.972,0.05 >, Time=0,Rtemp=1
Rappl=0
<SD,0.98,0.85 >, Time=1.7,Rtemp=0.809676
<FIR,0.972,0.06>, Time=0,Rtemp=1
<IT,0.984,0.05 >, Time=0,Rtemp=1
<DI,0.92,0.02>, Time=0,Rtemp=1
<ESC,0.972,0.05 >, Time=0,Rtemp=1
Rappl=0
<t >,Time=2.55, Rtemp=0.809676
<FIR,0.972,0.06>, Time=0,Rtemp=1
<IT,0.984,0.05 >, Time=0,Rtemp=1
<DI,0.92,0.02>, Time=0,Rtemp=1
<ESC,0.972,0.05 >, Time=0,Rtemp=1
Rappl=0
<FIR,0.972,0.06>, Time=0,Rtemp=1
<IT,0.984,0.05 >, Time=0,Rtemp=1
<DI,0.92,0.02>, Time=0,Rtemp=1
<ESC,0.972,0.05 >, Time=0,Rtemp=1
Rappl=0.809676
<FD,0.982,0.03 >, Time=0.06,Rtemp=0.02863512
<IT,0.984,0.05 >, Time=0,Rtemp=1
<DI,0.92,0.02>, Time=0,Rtemp=1
<ESC,0.972,0.05 >, Time=0,Rtemp=1
Rappl=0.809676
<t >,Time=0.09, Rtemp=0.02863512
<IT,0.984,0.05 >, Time=0,Rtemp=1
<DI,0.92,0.02>, Time=0,Rtemp=1
<ESC,0.972,0.05 >, Time=0,Rtemp=1
Rappl=0.809676
<IT,0.984,0.05 >, Time=0,Rtemp=1
<DI,0.92,0.02>, Time=0,Rtemp=1
<ESC,0.972,0.05 >, Time=0,Rtemp=1
Rappl=0.83831112
<BD,0.95,0.52 >, Time=0.05,Rtemp=0.04674
<DI,0.92,0.02>, Time=0,Rtemp=1
<ESC,0.972,0.05 >, Time=0,Rtemp=1
Rappl=0.83831112
<t >,Time=0.57, Rtemp=0.04674
<DI,0.92,0.02>, Time=0,Rtemp=1
<ESC,0.972,0.05 >, Time=0,Rtemp=1
Rappl=0.83831112
<DI,0.92,0.02>, Time=0,Rtemp=1
<ESC,0.972,0.05 >, Time=0,Rtemp=1
Rappl=0.88505112
<BD,0.95,0.52 >, Time=0.02,Rtemp=0.01748
<ESC,0.972,0.05 >, Time=0,Rtemp=1
Rappl=0.88505112
<t>,Time=0.54, Rtemp=0.01748
<ESC,0.972,0.05 >, Time=0,Rtemp=1
Rappl=0.88505112
<ESC,0.972,0.05 >, Time=0,Rtemp=1
Rappl=0.90253112
<CT,0.984,0.1 >, Time=0.05,Rtemp=0.0478224
Rappl=0.90253112
<t >,Time=0.15, Rtemp=0.0478224
Rappl=0.90253112
Rappl=0.95035352

Figure 8: Construction progress of HMI software reliability.

5 Conclusion

The redundancy design on the various function components in the HMI software is proposed in this paper based on the N-version programming and recovery block techniques, and the HMI software reliability is analyzed by employing the SBRA method. The result shows that the kind of redundancy design can improve software reliability effectively. Note that only the N-version programming and the recovery block techniques are considered in this paper, the other fault tolerance techniques, such as the N self-checking programming and retry block, will be introduced in order to access higher reliability and reduce the cost in the future work.

References

[1] Cobb, P.R., Lennon, C.J. & Long, K.J., *System and method for software error early detection and data capture,* US Patent: 5119377, June, 2, 1992.

[2] Moon, T.K., Error Correction Coding: Mathematical Methods and Algorithms, John Wiley & Sons, New Jersey, 2005.

[3] Lyu, M.R., *Handbook of software reliability engineering*, McGraw-Hill, Inc., NJ, USA, 1996.

[4] Pham, H., *System Software Reliability,* Springer-Verlag New York, 2006.

[5] Yacoub, S., Cukic, B. & Ammar, H., *A Scenario-Based Analysis for Component-Based Software*, IEEE Trans. Reliability, vol.53, no.4, pp. 465-480, 2004.

[6] CENELEC EN50126, Railway Applications: The specification and demonstration of Reliability, Availability, Maintainability and Safety (RAMS), 1999.

[7] Goseva-Popstojanova, K. & Grnarov, A., *N-Version Programming with Majority Voting Decision: Dependability Modeling and Evaluation*, Microprocessing and Microprogramming, Vol.38, No.1-5, pp.811-818, 1993.