ISSN: 2319-5967
ISO 9001:2008 Certified
International Journal of Engineering Science and Innovative Technology (IJESIT)
Volume 8, Issue 1, January 2019

Automated nested modeling of marine generator fault based on AltaRica

Meng Jie Wu, Bing Zhu, Zhen Li

Abstract*— As system size and complexity increase, system security modeling and analysis techniques are widely used in critical security systems. AltaRica is a high-level modeling language for security analysis. The AltaRica model can better reflect system functions and physical structure, eliminate the gap between traditional security analysis methods and system design, and improve the maintainability and reusability of security models. The article combines AltaRica with computer visualization modeling technology, develops a prototype of the visual hierarchical modeling tool supporting AltaRica language, and based on this tool prototype to carry out system security modeling and analysis work, the results show that the tool prototype can be very good Support AltaRica modeling language, the modeling process is correct, the system hierarchical structure is clear, and the user is convenient to use, which can effectively support the modeling and analysis of system security.

Keywords: **AltaRica; modeling; fault; development tool.**

I. INTRODUCTION

With the continuous development of computer and software technology, the consequences of security accidents caused by system failures are becoming more and more serious, and the security of systems is highly valued. System security modeling and analysis techniques are widely used in key security systems.

The AltaRica language is a modeling language for fault logic. The AltaRica model can truly reflect the structure of the system or the system operation mechanism, but the AltaRica code lacks a structural layering. Computer visual modeling technology can overcome this shortcoming, so that users can easily visualize the system through computer interface, the system structure is clearly described, the modeling process is clear, system modeling and system analysis process Combined into one, it facilitates the modification and maintenance of model and safety analysis, and improves the efficiency of safety work. As industrial systems become larger and larger, if only a single-layer system is used to describe the entire system, there will be too many model nodes at a single level, and the industrial system itself will be composed of many subsystems, some the system is composed of lower subsystems or nodes, so the system itself naturally forms a hierarchical structure. This first requires us to use hierarchical modeling techniques to describe the hierarchical structure of the system itself.

Based on the above, the article combines AltaRica with computer visualization modeling technology to develop a prototype of a visual hierarchical modeling tool supporting AltaRica language, and based on this tool prototype to carry out system security modeling and analysis.

II. FUNDAMENTAL PRINCIPLE

A. Fault logic modeling

Fault logic modeling methods are developed from traditional security analysis methods such as Fault Tree Analysis (FTA) and Failure Mode and Impact Analysis (FMEA), but overcome the traditional security analysis methods that are difficult to modify and reuse, and are not suitable for large-scale the disadvantages of complex systems. The fault logic modeling method fully defines the fault logic of each component, that is, fully explains how the deviation of each input of the component (ie, the input fault mode) and the component's own abnormality (self failure mode) combine to cause deviation of the component output behavior (output failure mode). . The fault logic of each component can be reused. When the input and output fault modes of different components are connected, the fault logic of the system is formed [1][2][3][4]

ISSN: 2319-5967
ISO 9001:2008 Certified
International Journal of Engineering Science and Innovative Technology (IJESIT)
Volume 8, Issue 1, January 2019

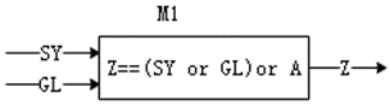

Fig.1. Simplified FPTN diagram

FPTN is a graphical notation of fault logic. Although it is a non-formal method, it uses a graphical language to visually and clearly describe the fault logic of a component or system. The graphical approach is simplified to form a simplified FPTN map as shown in Figure 1. The variable indicated by the left arrow of FPTN indicates the input failure mode of the module (faults SY and GL), and the arrow on the right side of the figure indicates the output failure mode of the module (fault X). The FPTN block diagram internally describes the module's own fault mode (such as fault A) and the logical relationship between the output fault mode and the input fault and the module's own fault [5][6].

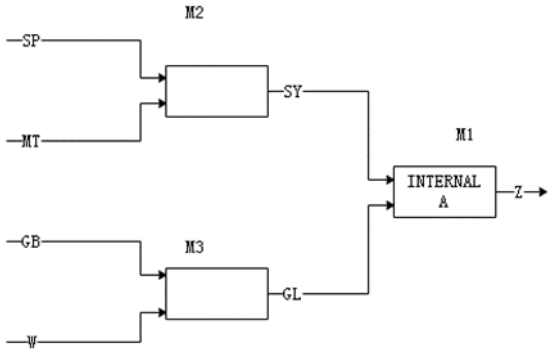

Fig.2. System FPTN diagram

According to the propagation relationship of faults, the output faults and input faults of different modules are connected to form the system FPTN diagram, as shown in Figure 2, Taking an output failure mode of the system (such as fault Z in Figure 2) as the top event, the cause of the fault is reversely searched according to the arrow of the FPTN graph, and finally the system input fault or the system itself fault that causes the system fault is determined. This fault tracing process is both a process of building a fault tree.

B. AltaRica Modeling Language

The AltaRica language was jointly developed by the computer science laboratory of Bordeaux in conjunction with other industrial sectors [7]. Since its appearance, it has received extensive attention and has been applied to a large number of practical projects. The AltaRica model uses nodes to describe the system. A node is a hierarchical description that can be made up of child nodes. A node that does not contain child nodes can represent a component (components are not decomposable), and a node that contains child nodes is used to represent a system.

a) AltaRica constituent elements

The semantics of the AltaRica [8] model is the Guarded Transition System (GTS). A GTS system consists of the following elements:

(1) State variables, used to describe the state of the system. These variables take values in finite fields (such as Boolean or enumerated character constants) or in infinite intervals (such as integers, floating-point numbers, or character constants).

(2) Flow variables, which are used to describe the transfer functions implemented by the system, namely the input and output of the system. Similar to a state variable, a stream variable takes a value within a domain.

(3) Assert, a set of constraints, used to describe the transfer function.

(4) Events, a series of events that may occur in the system.

(5) Conversion, marking describes the evolution of the system. The transformation is a triple <e, G, P>, which is defined as: e: G → P, where: e is the event of the marker conversion, G is a Boolean condition for the state and flow variables, called the whistle of the transformation, P Is the action performed on the new state calculation of the state variable. When the whistle is satisfied, the conversion e: G→P will trigger.

ISSN: 2319-5967
ISO 9001:2008 Certified
International Journal of Engineering Science and Innovative Technology (IJESIT)
Volume 8, Issue 1, January 2019

b) Component node

Each component node consists of three parts. The first part is used to declare different types of node elements, including states, flows, and events. The state is an internal variable that describes the operating mode (failure mode or normal mode) in which the component exists. The flow represents the input and output of the node; the second part describes the state transition mainly used for the node; the third part is a series of assertions that describe the logical relationship of the node.

c) System node

System nodes can be broken down into component nodes or continue to be broken down into lower-level system nodes. When describing a system node, you can first describe the component node in the system node in the way of Section 2.2.2, and then describe the component nodes contained in the node in the higher-level node.

C. Fault tree generation method based on AltaRica model

The AltaRica model can fully describe the system fault logic information, and the fault tree can be regarded as the refinement of the system fault logic. Therefore, before the AltaRica and fault tree transformation methods are clarified, the fault logic corresponding to the two models should be determined first, and then based on the same the fault logic compares the correspondence between the two models to determine the transformation method of the two models[9].

This paper first uses a simplified FPTN diagram to establish fault logic, then describes how the AltaRica data flow language describes the fault logic and the fault tree artificially formed based on the fault logic. Finally, the AltaRica code is compared with the manually drawn fault tree. Furthermore, the transformation relationship between the AltaRica language model and the fault tree is clearly defined for the same fault logic. Because AltaRica has different descriptions of the underlying components and the decomposable system, this paper is divided into the underlying components and systems when describing the AltaRica model and fault tree transformation relationship process.

a) Fault tree generation method for bottom component Altarica model

The underlying component is the component that cannot be further decomposed during the modeling process. It is the lowest unit that constitutes the system[10]. Figure 3 uses a simplified FPTN diagram to describe the fault logic of an underlying component, that is, the component M1 input failure modes are SY and GL (for the sake of simplicity, the fault mode contains only two values, normal and invalid), and the output failure mode is Z. At the same time, component M1 itself will also have failure mode A. When any of the SY and GL failure modes occur (ie, SY and GL are F, which can be expressed as SY.F, GL.F), component M1 will output fault Z, and at the same time, when component A fails, M1 Fault Z is also output.

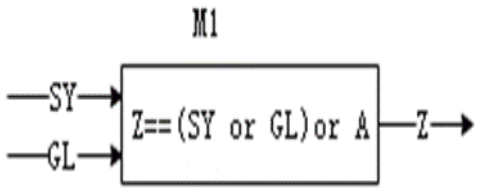

Fig. 3. Fault logic diagram

1) Fault tree

By selecting Z.F as the top event, the component failure logic can be transformed into the fault tree shown in Figure 4 below.

ISSN: 2319-5967
ISO 9001:2008 Certified
International Journal of Engineering Science and Innovative Technology (IJESIT)
Volume 8, Issue 1, January 2019

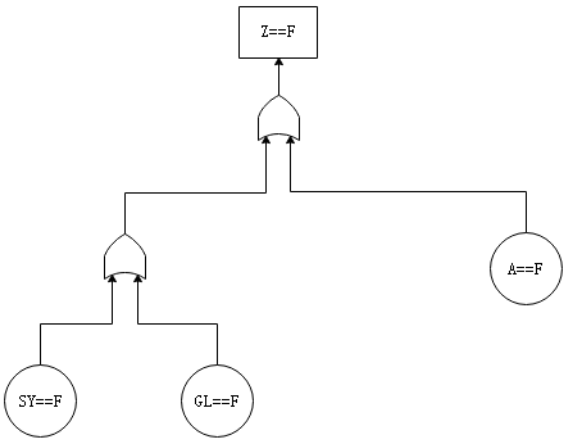

Fig 4. Fault Tree

As can be seen from Figure 4, the bottom event of the fault tree is formed by input faults and/or component failures.

b) Fault tree generation method for system Altarica model

A system is a component that can be further decomposed. The system can be composed of the underlying components or a lower-level system. Assume that the system is composed of components M1, M2, and M3. The fault logic of M2 and M3 is shown in Figure 5:

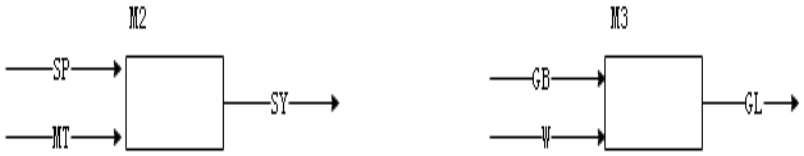

Fig 5 M2 and M3 fault logic diagram

Connect the corresponding input and output faults in the M1, M2 and M3 fault logic diagrams, and obtain the fault propagation logic inside the system System, as shown in Figure 6:

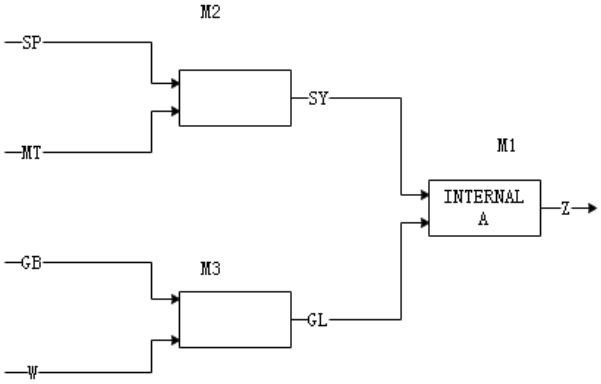

Fig 6. System fault logic diagram

1) Fault tree

According to the fault logic of the system System1, taking the fault Z.failed as the top event, the following fault tree can be generated:

ISSN: 2319-5967
ISO 9001:2008 Certified
International Journal of Engineering Science and Innovative Technology (IJESIT)
Volume 8, Issue 1, January 2019

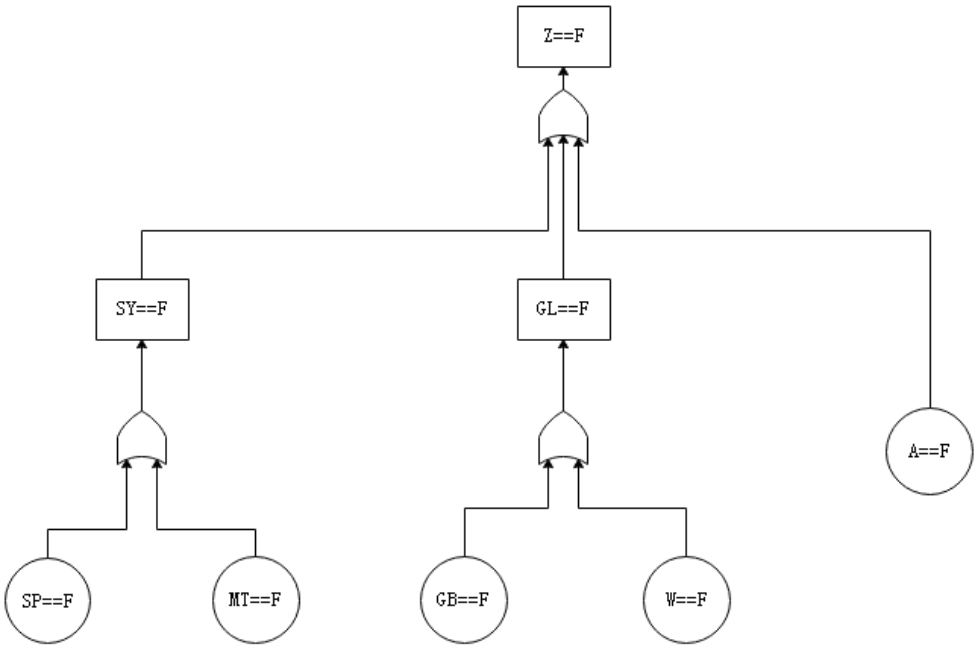

Fig 7. System fault tree

The top event of the system fault tree is usually set to the output fault of the system. As can be seen from Figure 7, the bottom event of the system fault tree is composed of system input faults and/or internal component faults.

III. INSTANCE VERIFICATION

1) Marine generator

As the heart of the ship, the generator of the ship plays a key role in the normal operation of the ship. Once the generator fails, it will lead to serious consequences. Therefore, it is very important to solve the problem of marine generator failure. The relevant staff must fully understand the cause of the generator failure, so as to eliminate the danger as soon as possible and ensure the safe operation of the ship. When the generator fails, the switch of the protection device will trip and the generator will stop running.

Generally speaking, there are two main reasons for the failure of the generator of the ship, namely internal causes and external causes. The internal cause is mainly caused by the aging of the internal circuit of the generator and the failure of the components. The external cause is mainly caused by the overload and undervoltage of the generator.

The generator is overloaded, that is, the load on the generator is too large to exceed the range that the generator can withstand, or the high-power electrical equipment undervoltage is operated without knowing the actual output power of the generator.

The under voltage of the generator is due to the failure of some of the generators in the generator set, or the voltage regulator has failed. By observing the meter reading, it is found that the generator voltage is decreasing.

2) Application

a) Class Modeling

The new class SY, SY consists of the input stream SP and MT, the output stream SY and the assertion. SP and MT respectively represent the non-human causes of the generator and the voltage loss caused by the human cause. The assertion indicates that the SP and the MT work together to generate the output stream SY. SP indicates that the generator has a loss of pressure, MT indicates a human cause, and the following table asserts that the input stream SY or the input stream MT has failed to produce a fault result SY

The new class SY, SY consists of the input stream SP and MT, the output stream SY and the assertion, SP indicates the generator voltage, MT indicates the human operation, and the assertion indicates that the SP and the MT work

ISSN: 2319-5967
ISO 9001:2008 Certified
International Journal of Engineering Science and Innovative Technology (IJESIT)
Volume 8, Issue 1, January 2019

together to generate the output stream SY. The following table asserts the input stream SY or the input stream MT fails to generate a fault result SY.

The new class GL, GL consists of the input streams GB and W, the output stream GL and the assertion, GB represents the generator load, W represents the erroneous operation, and the assertion indicates that the output stream GL is the result of the input stream GB and W or the gate action.

class name	state	input	output	assert
XT	A	SY、GL	Z	(SY==F) or (GL==F) or (A==F)
SY		SP、MT	SY	(SP==F)or(MT==F)
GL		GB、W	GL	(GB==F)or(W==F)

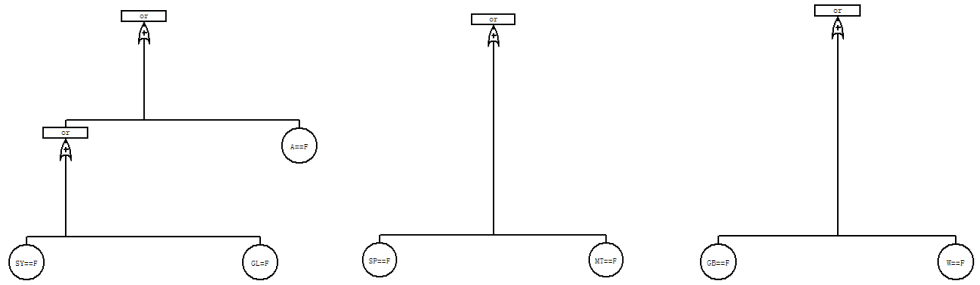

Fig 9. Class fault tree

b) System Nested Modeling

1) Identify top events and internal components

Set fault Z to the top event, and Z fault is output by class XT.

2) Establish a first-level fault tree

Since the fault tree Z is output by the component M1, a first-level fault tree is established according to the AltaRica code of the XT-like node. The input SY and GL of class XT are the outputs of class SY and class GL respectively. When faults A, SY and GL occur, fault Z will occur. The first level fault tree is shown in Figure 10 below:

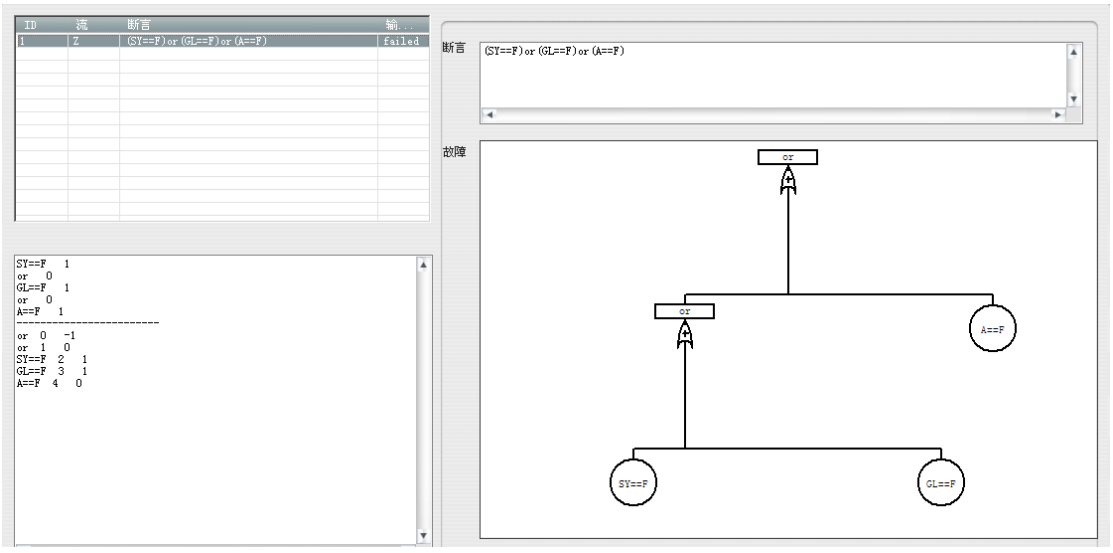

Fig 10. First level fault tree

3) Judging whether the bottom event of the first-level fault tree can be decomposed

The first level fault tree generates two bottom events SY.F and GL.F, and the input SY and GL of class XT are the outputs of class SY and class GL, respectively, and thus can be further decomposed.

The following describes the fault tree decomposition process by taking the fault Z as an example.

4) Determine the second-level fault tree top event and establish a secondary fault tree

With the fault Z as the top event, a second-level fault tree is established according to the input-output relationship of the class SY, as shown in Figure 11 below:

ISSN: 2319-5967
ISO 9001:2008 Certified
International Journal of Engineering Science and Innovative Technology (IJESIT)
Volume 8, Issue 1, January 2019

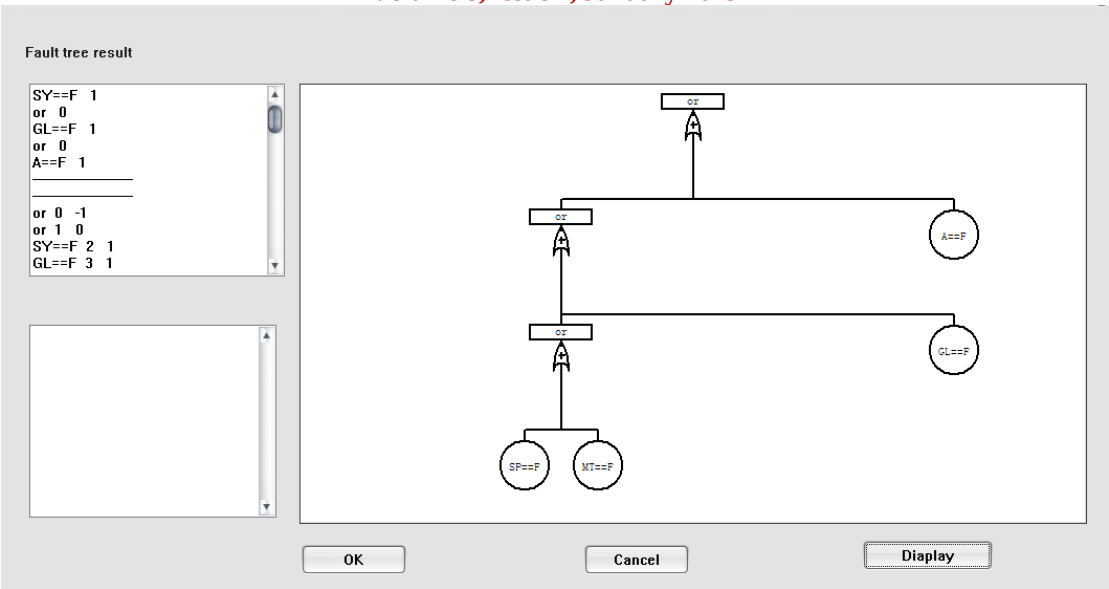

Fig 11 Second level fault tree

5) Judging whether the second-level fault tree can be decomposed

Repeat step 3). When the bottom event of the fault tree, its corresponding fault mode variable and the system input are not decomposable, the fault tree is established, as shown in Figure 12 below:

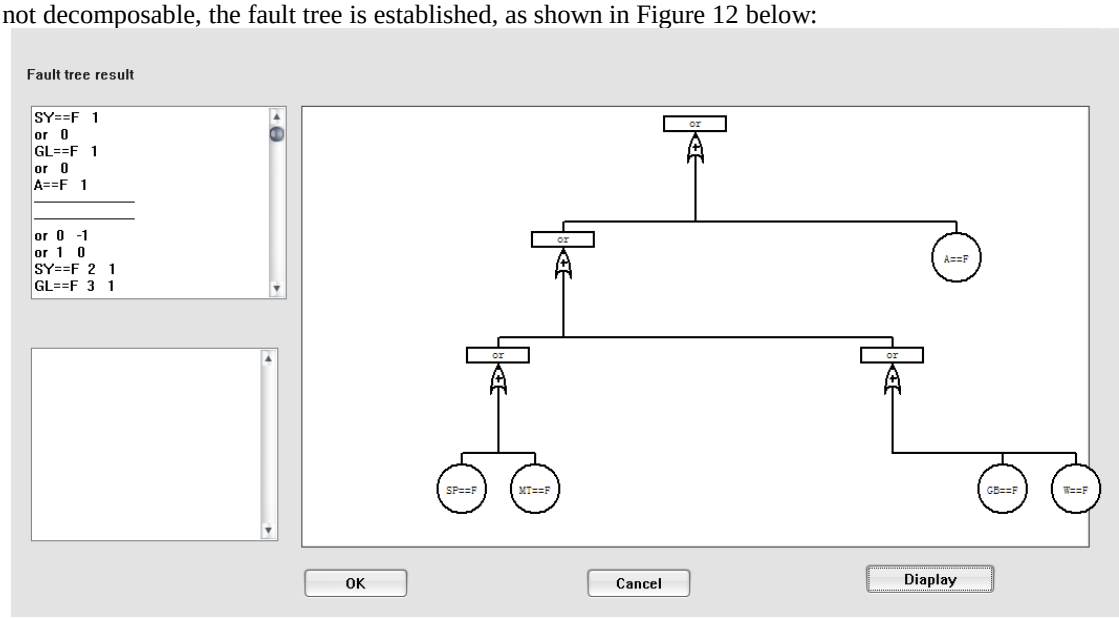

Fig 12. System fault tree

IV. CONCLUSION

This paper studies the hierarchical fault modeling and analysis based on AltaRica language, and develops a prototype of hierarchical fault modeling and analysis tools based on AltaRica language in Visual C++ 6.0 environment. Finally, an example application is carried out. The main work includes : Visualization of framework modeling; graphical interactive modeling; development of domain modeling, class modeling, system layered modeling through common actions such as mouse clicks, drag and drop; automatic mapping to generate Altarica code and other functions.

Through the above work, the article realizes interactive system modeling, the system structure description is clear, the system modeling process is clear, the modeling and analysis process is combined into one, the model

ISSN: 2319-5967
ISO 9001:2008 Certified
International Journal of Engineering Science and Innovative Technology (IJESIT)
Volume 8, Issue 1, January 2019

modification and analysis synchronization, with good user experience, at the same time Facilitate the modification and maintenance of models and safety analysis to improve the efficiency of safety work.

REFERENCES

[1] Xiaocheng Ge, Richard F. Paige, and John A. McDermid. Probabilistic Failure Propagation and Transformation Analysis. SAFECOMP 2009, LNCS 5775, pp. 215–228, 2009.

[2] G. Point and A. Rauzy. Altarica - constraint automata as a description language. European Journal on Automation, 1999. Special issue on the Modelling of Reactive Systems.

[3] A. Arnold, A. Griffault, G. Point, and A. Rauzy. The AltaRica formalism for describing concurrent systems. Fundamental Informaticae, 40:109–124, 2000.

AUTHOR BIOGRAPHY

Meng Jie Wu (1990-), Male, Master's Degree, Department of Electronics and Information, Jiangsu University of Science and Technology, The main research directions: system safety, system reliability.

Bing Zhu (1980-), Male, Master's Degree, senior engineer, shanghai merchant ship design and research institute, The main research direction is electrical Automation.

Zhen Li(1977-), Male, Associate Professor, Ph.D., Department of Electronics and Information, Jiangsu University of Science and Technology, Golden Ship Software Ltd ,The main research directions : system safety ,system reliability.