Embedded Software Reliability Testing and Its Practice

Yumei Wu
Dept. of System Engineering, Beihang University
Beijing, China
wuyumei@buaa.edu.cn

Shuanqi Wang
Dept. of System Engineering, Beihang University
Beijing, China
wangshuanqi@163.com

Zhengwei Yu
Dept. of System Engineering, Beihang University
Beijing, China
buaayuchard@hotmail.com

***Abstract*—Software reliability testing application framework (SRTAF) for the embedded software system is proposed in this paper. The structure and characteristics of SRTAF are described in detail based on the test process. Simulation models and data models as the main parts of the framework are presented, and how the models work and integrate with the test platform (in our previous work) are also explained. Data generation mechanism, data transmission mechanism and data behavior control mechanism are summarized and analyzed for clarifying the kernel of the framework. A case has been carried out for an avionics system using the SRTAF. It shows that the model driven based SRTAF can be effectively used to implement reliability testing for the embedded software system.**

Keywords- embedded software;software reliability testing; software simulation models

I. INTRODUCTION

Embedded systems are now widely used in defense, aeronautics, aerospace and many sectors of industry. Embedded software, as an essential part of the systems, is of major importance. It is remarked that the causes of more disasters and accidents lie in unreliable functioning of the software. So it is urgent and important to do software reliability testing to assure the reliability of the software especially the embedded software. But as the software is becoming more complex and the constraint of time and economic resources it is difficult to implement the reliability testing for the embedded software.

The remaining of this article is organized by follows: Section 2 discusses characteristics of the embedded software and the differences between the embedded software testing and the ordinary software testing. Section 3 describes the structure and nature of the SRTAF in detail. In section 4, we discuss the kernel of the SRTAF. In section 5, we introduce a case to illustrate the usability and effectiveness of the SRTAF. In section 6, we conclude and discuss the future work.

II. EMBEDDED SOFTWARE AND ITS RELIABILITY TESTING

Until now there is no explicit and uniform definition for the embedded software. Embedded software can be considered as which can make response to the random external input events in time and process the events as soon as possible (within the time limit). The software is usually complex and is broadly used in some important avionic control systems. The characteristics of the embedded software are real-time and embedded which are the differences between the embedded software and the ordinary software. The real-time characteristic means the input and behavior of the software has rigorous time constraint, for instance, input is usually classified by time, and they are periodical input, instant input or input when events occur. The response of the software to the input is generally within a very short time, and the time is generally calculated by millisecond. The embedded characteristic shows that the software is interacted with some related systems and specific working environments through I/O interfaces, and the software is "embedded" into a larger system. It can work normally and correctly only under this circumstance.

On the other hand we know that reliability testing requires that the operating environment should be as real as possible, and the testing itself can not destroy the integrity of the embedded system. But if we construct the operating environment with real interacted systems the cost would be too high to afford and the time would be too long as the required interacted systems can not be in field in most circumstances. Even if the above conditions are satisfied that is to say we use all the real interacted systems to do the testing, the danger risk should be considered because the reliability testing has the possibility of destroying the real interacted systems.

All these determine that the embedded software reliability testing should be automatic, real-time, closed-loop and non-invasive. So we cannot follow the general testing procedures for ordinary software to do the embedded software reliability testing.

In this paper, we present the software reliability testing application framework (SRTAF) that provides an approach for the embedded software reliability testing. It is a model driven based simulation approach and helps us analyze the input space, generate test data, organize the test, collect test result and give the reliability evaluation to complete a whole and effective reliability test.

III. THE DESCRIPTION OF THE MODEL DRIVEN BASED SRTAF

A complete reliability test is composed of three main stages: test preparation, test run and test result analysis. The structure of the SRTAF and the relationships among these three stages are illustrated (shown in Fig. 1) and described in detail by the following:

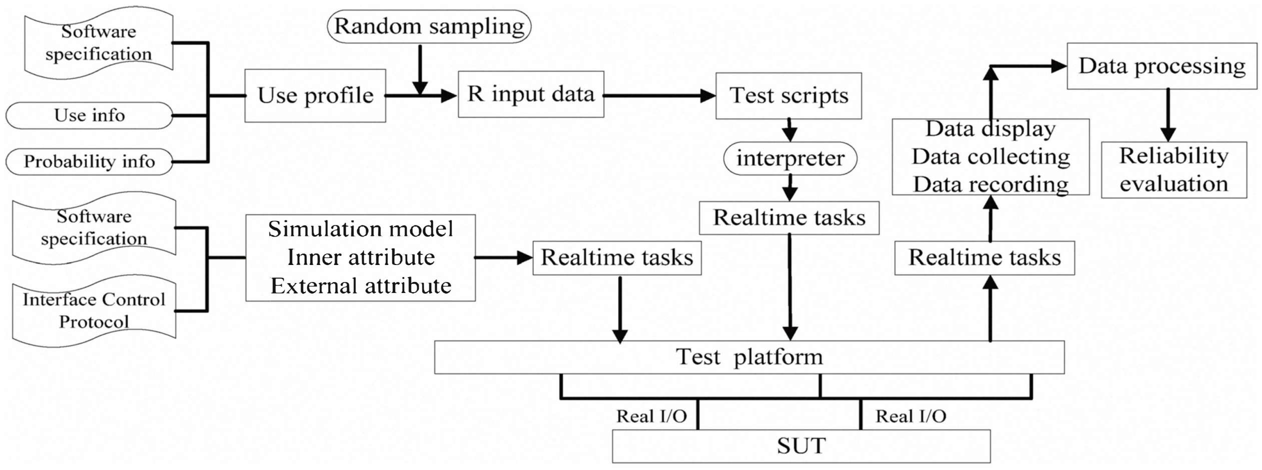

Figure 1. Diagram of software reliability testing application framework (SRTAF)

A. *Test Preparation*

During the test preparation, first we construct the models of the systems interacted with the embedded software under test (SUT) and the operating environment of the SUT according to the software specification. The model is called the simulation model, and it is characterized by the inner and external attributes: (1) The inner attribute refers to the arithmetic inside the model which is mainly decided by the software specification; the external attribute mainly decided by interface control protocol defines the types of the I/O interfaces and the physical information such as baud rate and voltage; the bus channel number, sub-address, and data length of MIL-STD-1553B. It also ensures the model satisfies the time constraints such as sequential, concurrent or feedback relations. (2) The inner attribute (the arithmetic of the model) can be simplified sometimes to ensure the real-time of the simulation model. The validity of the I/O interfaces and scheduling is guaranteed by the external attribute of the simulation model. (3) Message response is one of the important characteristics of simulation models and it supports getting the result data from the SUT and processing them in order to generate new data online which are used as new input to the SUT. Only by this the SUT can run normally and accurately. So message response is a kind of information feedback mechanism.

Another important work is to construct the operation profile (Musa, J.D. 1998) or use profile according to the software operation information or use information, and then set up the sampling principle. A large amount of reliability test input data which compose the test input data set can be generated based on it. For the embedded software the reliability test input data usually cannot be input manually because manual input is too slow for testing the embedded software. So we use the data model to simulate the data input and transformation behavior, that is to say, Data models are used to describe how the input data (test cases) are generated and how they are transmitted.

B. *Test Run*

During the performance of the test, the simulation models are compiled and the related information is downloaded to the test platform (Ruan, L. et al. 2000, Liu, B 2000a,b) When the test starts, the simulation models generate data according to the inner attributes, at the same time receive and send data according to the external attributes automatically. The data are transmitted with the SUT through real I/O interfaces in the test platform.

The test cases are composed of two parts: the first one is data which involved data types, data value or value range, and the variance rule; the second is time and behavior constraints which specify when the data should be sent, where it is to be sent or where we receive the data. We use test scripts to express the test cases, and these test scripts are converted to real time tasks arranged by time through the interpreter when the test begins. Data models are also in charge of real-time data display and data collection. They are very important and useful: data display can help monitor the test process, find and solve the problem; data collection provides data with time label for reliability estimation and decides the accuracy of reliability estimation to some extent if the result data are collected completely and exactly.

The test platform in our previous work (Liu, B. 2000a, b) supports the mechanisms of simulation model registration compilation and the test script real-time explanation.

In this phase we focus on the scheduling and intercommunication among the simulation models, data models and the SUT. It depends on the scheduling of our test platform. After the models have been compiled, they are downloaded to the test platform and run as three different kinds of tasks scheduled by the task scheduler (Fig. 2). They are periodic tasks, aperiodic tasks and sporadic tasks.

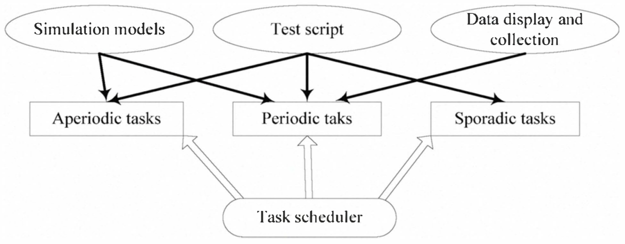

Figure 2. General scheduling diagram

- Periodic task: receiving and sending data among the simulation models and real-time calculation in simulation models can be deemed as periodic tasks. And the task period is usually the working period of the models. Data display and data collection tasks are also periodic tasks and their period can be set by customer requirement.
- Aperiodic task: the realization of message response in simulation models can be regarded as performance of aperiodic tasks, and this kind of tasks is usually initiated by other tasks. Test script tasks are aperiodic tasks, as test script imitates how the SUT is used.
- Sporadic task: we use sporadic tasks to implement sending test scripts online. It is a very flexible mechanism. When the test is running testers can make temporary scripts which will be added to the data model, and the data will be sent to the SUT immediately.

C. *Test Result Analysis*

Test result analysis is added to the framework as a component. The test result data have been recorded and stored in the database during the test process. After collecting the test result data we preprocess the data according to the requirement and make comparison with the expectation. For the software reliability testing we make the reliability evaluation based on the data with time label. There are a lot of software tools such as SRAT (developed by our lab members) can be used to process the test result and give the reliability estimation.

The SRTAF is in comprehensive support of the three major phases of the embedded software reliability testing: (1) The framework can provide the test data generation and transmission mechanism to satisfy the time and scheduling requirements of the embedded software reliability testing. (2) The framework is qualified with the ability of modeling the related systems or equipments accurately and the operating environment interacted with the embedded SUT. (3) The framework is integrated very well with our test platform. Model construction, model compilation, downloading and running, and tasks (generated from models) scheduling are regulated and guaranteed in the framework in order to ensure the integrality, efficiency and sufficiency of the reliability test process.

IV. THE KERNEL OF THE FRAMEWORK SRTAF

From the above we can know that the SRTAF is a kind of model driven framework. Simulation models and data models are the important parts involved in the SRTAF. The characteristics of these two kinds of models can both be described by the following three mechanisms. They are data generation mechanism, data transmission mechanism and data behavior control mechanism. Fig. 3 shows the relationships among them.

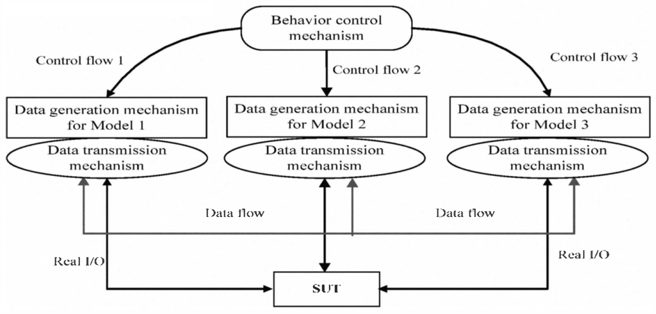
Behavior control mechanism
Control flow 1
Control flow 2
Control flow 3
Data generation mechanism for Model 1
Data generation mechanism for Model 2
Data generation mechanism for Model 3
Data transmission mechanism
Data transmission mechanism
Data transmission mechanism
Data flow
Data flow
Real I/O
Real I/O
SUT

Figure 3. The running diagram of models

A. *Data Generation Mechanism:*

For simulation models data are generated from the inner attributes of the simulation model. Real arithmetic of the interacted systems should be used in the simulation model in order to improve its fidelity and credibility. For data models data are generated based on the software use profile, and by the random sampling theory in mathematics we can get the input data of the embedded software in reliability testing.

B. *Data Transmission Mechanism:*

The data interaction between the SUT and the test platform is realized by the data transmission mechanism. External attributes of simulation models have specified the data transmission mechanism, such as the types of the I/O interfaces. Message response of simulation models can be guaranteed by transmission mechanism. So as one of the characteristics of the simulation models the closed-loop feedback mechanism is able to ensure the realness of the data transmission. Data models are like containers which not only contain the input data but also process the data for data display during test running and at the same time collect the result data online.

C. *Data Behavior Control Mechanism:*

It determines the sender and receiver of the data and makes time constraints for the input data variable. The model registration mechanism is provided in the framework: the behavior of the models is established during test preparation, and when the test begins the test platform will be informed of the model behavior and make scheduling according to the mechanism which has been registered. For simulation models behavior control mechanism is based on the real performance of the interacted systems, and for data models it is based on the test scripts which are transferred from the input data and describe the input behavior. The test process is dynamically controlled by the behavior control mechanism according to the current status of the SUT and the running information that the models have registered.

Therefore we can see that the mechanisms of data generation, data transmission and data behavior control can guarantee that the simulation models and data models run correctly and accurately in the test platform: the input data can be input to the SUT by the test scripts which satisfy the requirement; the data needed for the normal running of the test process can be dynamically generated from the simulation models; real-time communication between the SUT and interacted systems or between every two interacted systems can be satisfied. The good performance of the reliability testing for the embedded software can be implemented.

V. APPLICATION

A case was carried out for an INS (Inertial Navigation System) using the SRTAF. In this section we would describe the test procedures of the INS under the instruction of the SRTAF in order to have a good understanding of how the SRTAF is used for embedded software reliability testing.

INS is a kind of typical avionics equipment with the embedded real-time software. It is interacted with many avionics equipments, such as MC (Mission Computer), ADC (Air Data Computer), DCMP (Display Control and Management Processor), and CNI (Communication and Navigation Identification). Figure 4 shows the interface and data transmission relationships between the simulation models in the testing.

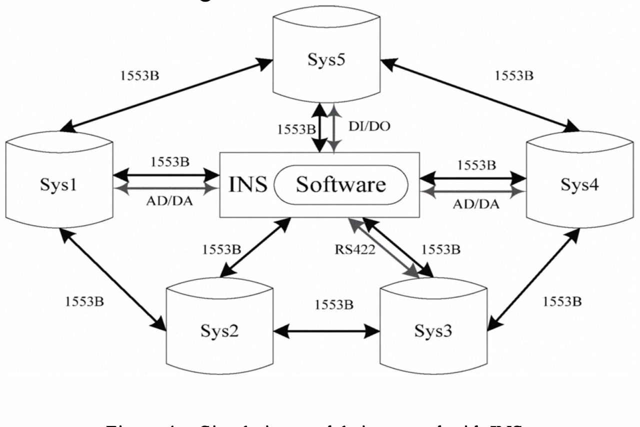

Figure 4. Simulation models interacted with INS

Test preparation: we constructed the simulation models including inner attributes and external attributes of Sys1, Sys2, Sys3, Sys4 and Sys5 under the instruction of the software specification and interface control protocol. At the same time we generated test cases based on the use profile which we made after we had conducted an investigation among the software users.

Test run: test cases had been converted to test scripts using the script language, some types of the test cases can be transformed to test scripts automatically by our test platform. When the test initialized the simulation models were compiled and downloaded to the test platform, at the same time test scripts were converted to tasks. And all these run as three kinds of tasks which were scheduled by the mechanisms that the models had registered in the test platform.

Test data analysis and software reliability evaluation: test data had been recorded and stored during the test process. Data display of data models can help us monitor the whole test process and find the problems in time in order to make the test result analysis and reliability evaluation more accurate and reasonable. The data collected in the process were proved complete, correct and effective based on the analysis especially of the data with time label. The SRAT tool was used to evaluate the reliability of the INS .

In this work we constructed some complex models and some types of interfaces such as MIL-STD-1553B, AD/DA, DI/DO, RS422. By monitoring the performance of the whole test process the SRTAF was successfully used, and a lot of sequential, logical and interface faults had been found, reflecting excellent usability, expandability and validating the correctness and effectiveness of the SRTAF put forward in this paper.

VI. CONCLUSION

From all the above we know that the framework SRTAF presented provides a new, efficient and effective approach for the embedded software reliability testing with the advantages of the model driven approach.

Simulation models and data models, as the key parts of the framework, their characteristics and running mechanisms have been proposed and analyzed. With these we can guarantee the real-time and reality of the test process, and the flexibility of the models also makes the framework have strong expandability. Of course this approach which saves much time is also cost effective.

Data model can also be used in boundary or abnormal function testing, stress testing and some other types of testing as we change the data generation mechanism of the models. And more work should be done to improve the performance of the SRTAF and make it generally used in testing field in the future. Now reliability test cases are generated only dependent on the operation or use information, input space analysis such as equivalence partitioning which can be implemented by data models may provide for us a better way to decrease the number of test cases, delete ineffective test cases, accelerate the test process and improve the test efficiency. Therefore we can prospect that the framework would be broadly applied in other types of testing for the embedded software.

REFERENCES

[1] Musa, J.D. 1998. Software reliability engineering. New York: McGraw-Hill.

[2] Ruan, L. Liu B. Chen, X.S. 2000. Software reliability testing environment. test control technology. 19(2): 71-76.

[3] Liu, B. 2000a. Study on the embedded software reliability simulation testing environment, PhD Dissertation. Beijing University of Aeronautics and Astronautics.

[4] Liu, B. Gao, X.P. Lu, M.Y. Ruan, L. 2000b. Study on the embedded software reliability simulation testing system. Journal of Beijing University of Aeronautics and Astronautics. 26(4): 59-63.