A Multi-layered Statechart Diagram Including Mitigation Behavior

Hoijin Yoon,

Hyupsung University, Kyunggi, South Korea

Summary
Safety critical system such as Unmanned Aerial Vehicle, Medical system, Railroad control system and so on includes software in it. It is called Embedded software. Model based testing is a recommended solution for testing embedded software. It needs a specific model that it designs test cases from, and its testing quality totally depends on the model's quality. Unlike the traditional software, the embedded software is expected to control risky situation. It means that the mitigation of risk is necessary. Therefore a model should include mitigation information for testing mitigation behaviors of embedded software. This paper proposes a diagram including both of normal behaviors and their mitigation behaviors. The mitigation behavior supports failsafe behavior, and Model based testing using the multi-layered statechart diagram results in failtsafe testing, which checks if the system mitigates risk well and doesn't go to failures at all.

Key words:
Model-based testing, Embedded software, Failsafe testing, Mitigation, Safety Critical System.

1. Introduction

Nowadays, software goes into a device, which is used in our daily life, instead of a general computer. The system is called Embedded system and the software is called Embedded software. An embedded system [1, 2] is a system built for dedicated control functions. Embedded software [2] is the software running on an embedded system. Embedded systems have become increasingly sophisticated and their software content has grown rapidly in the last few years. The requirements that must be fulfilled while developing embedded software are complex in comparison to standard software. Moreover, it extends to work in safety critical systems such as Medical devices, Automotive systems, and Unmanned Aerial Vehicle (UAV).
According to the study in [3] 50% of embedded systems development projects are months behind schedule and only 44% of designs meet 20% of functionality and performance expectations. This happens despite the fact that approximately 50% of total development effort is spent on testing [3, 4]. The testing in embedded systems especially in safety critical systems is heavier than in traditional computer based applications.
Model based Testing (MBT) allows tests to be linked directly to the SUT requirements, makes readability, understandability and maintainability of tests easier. It helps to ensure a repeatable and scientific basis for testing and it may give good coverage of all the behaviors of the SUT [5]. Finally, it is a way to reduce the efforts and cost for testing [6].
This paper proposes a model for MBT, which represents Embedded software such as an autopilot software of UAV. Unlike traditional modeling diagrams, our model covers failsafe behaviors. It mean that MBT using the failsafe behavior model tests if the system manages failure-causing situations and handles their mitigations rightly.
In Section 2, we introduce some issues of MBT. Section 3 explains what this paper proposes. It is a multi-layered state diagram that represents both of normal behaviors and failsafe behaviors at the same time. We conclude this paper with mentioning a usage of the multi-layered state diagram in UAV.

2. Related Works

2.1 Model based Development

The development process of embedded systems usually occurs on at least three different levels. First a model of the system is built. It simulates the required system behavior and usually represents an abstraction of the system. When the model is revealed to be correct, code is generated from the model. This is the software level. Eventually, hardware including the software is the product of the development. The reason for building those intermediate levels is the fact, that it is much cheaper and faster to modify a model than to change the final product. The entire process is called model-based development (MBD).
The multiple V-model [7, 8], based on the traditional V-Modell®, takes this phenomenon into account. The V-Modell is a guideline for the planning and execution of development projects, which takes into account the whole life cycle of the system. The V-Modell defines the results that have to be prepared in a project and describes the concrete approaches that are used to achieve these results [9]. In the multiple V-model, each specification level (e.g.,

Manuscript received April 5, 2015
Manuscript revised April 20, 2015

model, software, final product) follows a complete V-development cycle, including design, build, and test activities as shown in Fig.1. The essence of the multiple V-model is that different physical representations of the same system on different abstraction levels are developed, aiming at the same final functionality. Then, the complete functionality can be tested on those different platforms.

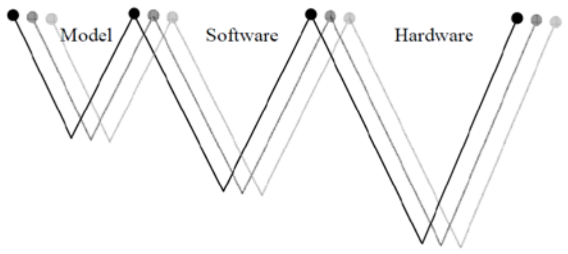

Fig. 1 Multijple V-Model

Since certain detailed technical properties cannot be tested very well on the model, they must be tested on the prototype instead. Testing the various SUT representations often requires specific techniques and a specific test environment. Therefore, a clear relation between the multiple V-model and the different test environments exists.

2.2 MBT

Model-based testing is testing in which the entire test specification is derived in whole or in part from both the system requirements and a model that describe selected functional aspects of the SUT. In this context, the term entire test specification covers the abstract test scenarios substantiated with the concrete sets of test data and the expected SUT outputs. It is organized in a set of test cases [10].

The introduction of MBD led to the development of modeling technologies. Consequently, executable high-level models can be obtained. The selection of a modeling technology is very dependent on the type of system being modeled and the task for which the model is being constructed [11].

One thing we need to remember is that the models generated through development phases are not good sources of MBT. Development models could contain defects, and to use models from development activities could transfer defects of development models to testing activity. It is recommended that test models should be made separately from development phases, moreover by an independent testing team.

3. Multi-layered State Diagram

UAV is running based on autopilot programs. This paper proposes a specific diagram that covers normal behavior model and failsafe behavior model together. We explain the diagram with a case of Drone's autopilot. Drone is one kind of UAV and it is very popular since its price is low and its operation is easy to learn. Also many industry fields interest in adopting Drone to their business models.

3.1 Autopilot Implementation Environment in Drone

Drone flies following waypoints. Users set up waypoints that they want Drone moves to. A waypoint consists of its longitude, latitude, and altitude. A special tool helps this setting up process. The tool is named Mission Planner [12]. Fig.2 is a screen shot of Mission Planner. Mission Planner shows the map of the home location, which is also set up by users. A sequence of waypoints are written in a file and it will be sent to Drone's special device. Pixhawk [13] is one kind of the devices.

Fig.3 shows two pictures; one is a quadcopter and the other is Pixhawk that is mounted on the quadcopter as expressed in the red circle and arrow. Pixhawk is autopilot device mounted on X8 manufactured by 3D Robotics. We have X8 and used it as a sample case of autopilot implementation. Finally, Pixhawk includes autopilot waypoints and some other settings of peripheral devices of Drone.

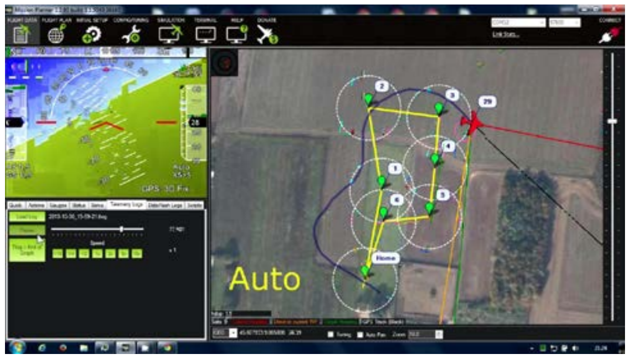

Fig. 2 Mission Planner

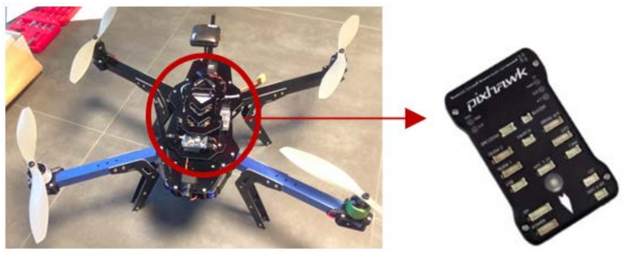

Fig. 3 3D Robotics quadcopter and its Pixhawk

3.2 Normal behaviors and Failsafe behaviors

In modeling software, there are three separate viewpoints; Functional modeling, Behavioral modeling, and Structural modeling. We focus on Behavioral modeling since the failsafe mitigation would fall on Behavioral modeling. Normally, Behavioral modeling is to draw Statechart diagram, Sequence diagram, or Collaboration diagram of Unified Modeling Language (UML) [14]. Behavioral model is built for each function of Functional model. The normal behavior is what users or clients expect a system to do.

Safety critical system, such as Drones, Medical system, Railroad system and so on, should control unexpected behaviors because the unexpected behaviors could result in a very serious happening like killing human. Therefore, the unexpected behaviors need to be modeled and tested. According to Risk Management Process, developers implement mitigation strategies for expected risks. In Drone, developers set up mitigation strategies in a menu called “Failsafe options” of Mission Planner. They set up a mitigation action for a certain risky condition. For example, the copter should land if the battery level is below 10.5V. This behavior is called Failsafe behavior.

3.3 Multi-layered Statechart diagram

Unlike traditional behavioral modeling, we add another layer that covers Failsafe behaviors. The failsafe behavior is on risk mitigation, and safety critical systems are supposed to have mitigation strategies for the expected risks. In case of Drone, the battery failure is a main cause of crashes. The mitigation is set up through Mission Planner, and the mitigation strategy for the battery failure is embedded in autopilot programs that is running on Pixhawk.

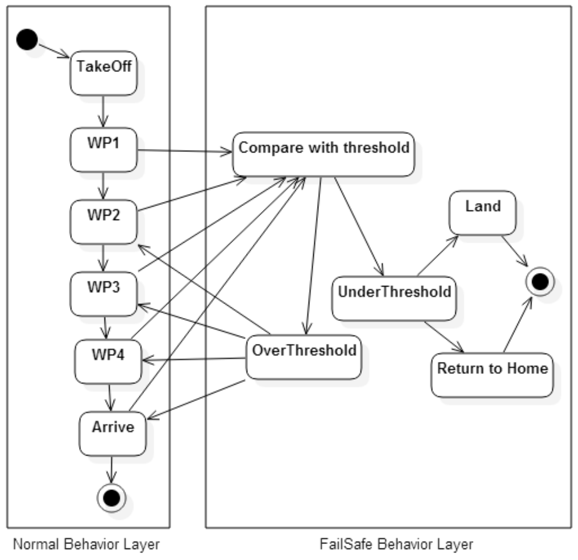

Fig. 4 Multi-layered statechart diagram for battery failure mitigation

This mitigation modeling is done separately from behavioral modeling. Of course, the testing would be done separately. Now we need to remind the fact that failures could happen during normal behaviors. It means that we should test if the system mitigates risk well and doesn’t go to failures at all. It is Failsafe Testing. The failsafe testing focus on the flow of going from normal behavior states to failsafe behavior states. That is why we need to combine normal behaviors and failsafe behaviors in one model.

Fig.4 is the proposed diagram, multi-layered statechart diagram. It is about a sample mitigation, which orders Drone to “Land” or “Return to home” if the battery voltage level is under the threshold. The default threshold value is 10.5V. In each waypoint state, it compares the current voltage with the threshold. If the current value is over the threshold, it would go to the next waypoint safely. If the voltage is under threshold, it would do the predefined mitigation action. Land or Return to Home could be one.

4. Conclusions

MBT is a recommended solution for testing Embedded software. Recently, software is embedded in safety critical systems such as Railroad system, Aircraft, Drone, Medical devices and so on. It means that Embedded software testing is important as much as the safety issue is getting serious.

This paper proposed a multi-layered statechart diagram that includes normal behaviors and failsafe behaviors together. Traditional statechart diagram is used in Behavioral modeling, and the behavioral modeling builds diagrams that shows clients or users’ requirements. And MBT apply test criteria to the modeling diagram and design test cases. As mentioned in Section 2, MBT’s result totally depends on which models are used. Our multi-layered statechart diagram has MBT design test cases that check if the system mitigates risk well and doesn’t go to failures at all.

Suppose MBT applies “all-path criterion” to Fig.4 diagram. One of the test cases is <TakeOff, WP1, WP2, WP3, Compare with threshold, UnderThreshold, Land> and its expected output is <no-failure>. It tests if the system goes to Compare with threshold from WP3, and if it goes to Land in UnderThreshold condition.

Finally, our diagram supports Failsafe testing through MBT. The model using this diagram would be more sophisticated if mitigation patterns are applies in weaving normal behavioral layers and failsafe behavior layers.

Acknowledgments

This research was supported by Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Science, ICT & Future Planning (NRF-2014R1A1A3051827).

References

[1] Broy, M., von der Beeck, M., Krüger, I.: Softbed: Problemanalyse für ein Großverbundprojekt, “Systemtechnik Automobil – Software für eingebettete Systeme”. Ausarbeitung für das BMBF. 1998 (in German).
[2] Lazić, Lj., Velašević, D.: Applying simulation and design of experiments to the embedded software testing process. In Software Testing, Verification & Reliability, Volume 14, Issue 4, Pages: 257 – 282, ISSN: 0960-0833. John Wiley and Sons Ltd. Chichester, UK, UK, 2004.
[3] Encontre, V.: Testing embedded systems: Do you have the GuTs for it?. IBM, 2003. http://www-128.ibm.com/developerworks/rational/library/459.html [04/18/2008].
[4] Helmerich, A., Koch, N. and Mandel, L., Braun, P., Dornbusch, P., Gruler, A., Keil, P., Leisibach, R., Romberg, J., Schätz, B., Wild, T. Wimmel, G.: Study of Worldwide Trends and R&D Programmes in Embedded Systems in View of Maximising the Impact of a Technology Platform in the Area, Final Report for the European Commission, Brussels Belgium, 2005.
[5] Utting M. Model-Based Testing. In Proceedings of the Workshop on Verified Software:Theory, Tools, and Experiments VSTTE 2005. 2005.
[6] Pretschner, A., Prenninger, W., Wagner, S., Kühnel, C., Baumgartner, M., Sostawa, B., Zölch, R., Stauner, T.: One evaluation of model-based testing and its automation. In Proceedings of the 27th International Conference on Software Engineering, St. Louis, MO, U.S.A., Pages: 392 – 401, ISBN: 1-59593-963-2. ACM New York, NY, USA, 2005.
[7] Brökman, B., Notenboom, E.: Testing Embedded Software. ISBN: 978-0-3211-5986-1. Addison-Wesley International, 2002.
[8] Schäuffele, J., Zurawka, T.: Automotive Software Engineering, ISBN: 3528110406. Vieweg, 2006.
[9] V-Modell® XT, version 1.2.1, 2006, ftp://ftp.tuclausthal.de/pub/institute/informatik/v-modell-t/Releases/1.2.1/Documentation/VModell-XT-Complete.pdf [07/05/08].
[10] Justyna Zander, Ina Schieferdecker, Pieter J. Mosterman, Model-Based Testing for Embedded Systems, CRC press, 2011
[11] Mosterman, P. J.: Hybrid dynamic systems: a hybrid bond graph modeling paradigm and its application in diagnosis, PhD thesis, Faculty of the Graduate School of Vanderbilt University, Electrical Engineering. 1997.
[12] Mission Planner Home, http://planner.ardupilot.com/
[13] “Pixhawk Autopilot,” http://pixhawk.org/modules/pixhawk
[14] Alan Dennis, Barbara Haley Wixom, David Tegarden, Systems Analysis and Design with UML: An Object-oriented Approach, Wiley, 2010

Hoijin Yoon received the B.S., M.S. and Ph.D in Computer Science from Ewha Womans University in 1993, 1998, and 2004 respectively. During 2004-2007, she stayed in Ewha Womans University as a full time lecturer. She has been working as a faculty at Hyupsung Univeristy since 2007. Her research interests are Software Testing and Safety Critical System.