Test Case Extraction for Intelligent Power Switch of Heterogeneous Vehicles

Dong Ho Kim[1], Hyun Seung Son[1], Woo Yeol Kim[1], and Robert Young Chul Kim[1]

[1] Dept. of CIC(Computer and Information Communication), Hongik University, Sejong Campus, 339-701, Korea
{ray,son,john,bob}@selab.hongik.ac.kr

Abstract. Modern car-related industry has required a rapid development of a car. Although many functions may be added to meet various requirements, its development time should be shorter. As a result, traditional development methods cannot satisfy the test requirements which not only have various functions, but also take a longer time on performing the test. Therefore, we propose to adopt the existing Model Driven Architecture (MDA) to be used for testing the various platforms. In this paper, we apply this test method to a car IPS software test, and extract test cases using this method.

Keywords: Testing, Model based Test, Heterogeneous Platform, MDA(Model Driven Architecture), IPS(Intelligent Power Supply)

1 Introduction

Currently, the car industry requires IT-related functions and the size of the software has increased. Because of this, a new development method is needed, which is different from the one being currently used. To solve this problem, a platform-based development has introduced to the car industry. However, existing platform-based development method has not matured yet due to its relatively short history [1]. Therefore, it is also true that the development speed for many delicated functions cannot satisfy the consumer's needs to apply the state-of-the-art technology. Moreover, since a car is related with a person's life, we cannot simply focus on rapid development like other traditional testing techniques.

For this reason, we introduce and apply the Model Driven Architecture (MDA) based on a software engineering model and the V-model used in the traditional embedded software development [2,3].

In this paper, we apply the previously developed MDA-based test process to the car IPS software test. Through the Use Case based test case method, a test case for car IPS is created. Also we will show each deliverable stage and test case extraction process.

This paper is organized as follows: Chapter 2 describes related work of the MDA based development method; Chapter 3 discusses the test case extraction process and in Chapter 4, a case study is explained; and Chapter 5 provides conclusion and further research.

2 Related Work

An existing embedded software development process develops software by modeling, prototype, and production in this exact order [4]. This method while repetitive and agile can be used is suitable for developing a new system. However, this method has its drawbacks in which it can be only applied to a single system. The reason for this drawback is that the software already has dependency on the hardware when developing the model and prototype. The existing methods have problems with heterogeneous systems development and reuse of the software. The MDA method can solve this hardware dependency problem [2,3]. However, this method requires more reinforcement for safety-related parts like a car. We proposed a combination of a parallel testing process with an existing embedded software development process. This is to allow the development and its testing process to be performed simultaneously.

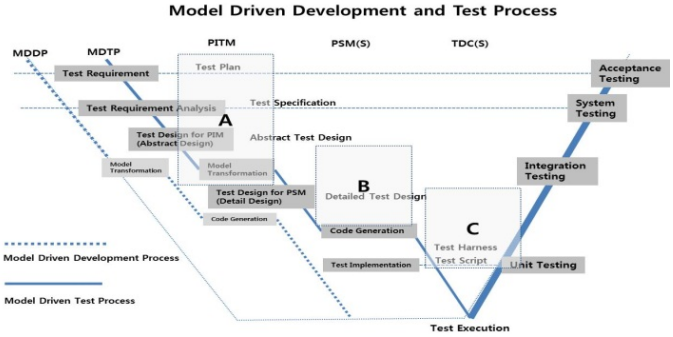

Fig. 1. Model-based development and test process

Figure 1 shows an overall flow of the model-based development and test process. The idea of the Model Driven Architecture (MDA), which helps a development over many various platforms in existing software engineering, is combined with embedded development for its development and test over a heterogeneous embedded environment.

Each embedded development and test environments are all very different. The MDA-based development process is to define a platform-independent model, and then creates a platform-specified model by transforming the model through an automatic tool. Finally it creates codes based on this model for each embedded target. However, since it lacks testing definitions and process, a test process corresponding to each development process was defined respectively [5,6,7,8] .

3. Use case based Test case Extraction Method

The Use case based test case extraction method is to create a test case through sequential processes from the Use Case diagrams. Decision factor is extracted from

the Use case specification which is one of the Use Case diagrams, and a decision table is created with this decision factor and a final test case is generated.

The Use case based test case extraction method is performed as shown in Figure 2. First, a decision factor (input value, output value and condition value) is identified. At this time, input values are list number, value and type. List number is a sequence number for the identified factor, and the value describes the identified factor. A type is categorized into input, output and condition. Secondly, based on this, a test case is created. The test case extraction method is to list decision factors such as input, condition, and output of this sequence, and to create a test case by applying coverage.

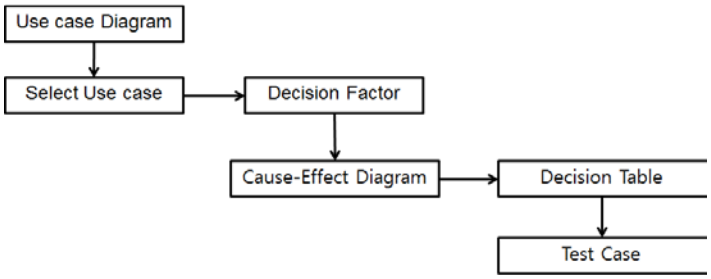

Fig. 2. Use case based test case extraction method

4 Case Study

A car does not have a power distribution problem while operating with a battery. However, after startup there may be a problem of stable power distribution. The junction box inside a standard car is placed to solve this problem. Figure 3 shows the real junction box. The junction box distributes stable power to 26 electronic parts such as wiper, car engine and cooling fan. The junction box has different functional characteristics depending on the car parts and the nations of the supplier.

Fig. 3. Junction Box

For a model-based development and test case, we have an example of a cooling fan which functions to cool the engine of the car. Depending on temperature, a cooling fan has simple adjustment features such as LOW, HIGH or OFF. If a cooling fan is not working as expected, it could cause serious problems to the car. Table 1 summarizes the signals of input or output from a cooling fan. The model is created based on these signals and the state diagram in Figure 4.

Table 1. Signal list of the COOLING_FAN module

Function		Name	Description	Status
Input	Hardwire	O_SNSR2	SENSOR2 IPS	ON / OFF
	Hardwire	L_CFanLoSig	COOLING_FAN_LOW SIGNAL	ON / OFF
	Hardwire	L_CFanHiSig	COOLING_FAN_HIGH SIGNAL	ON / OFF
Internal	Logic	M_ CFanLo	MCU COOLING_FAN_LOW SIGNAL	ON / OFF
	Logic	CS_ CFanLo	COOLING_FAN_LOW SIGNAL	Variable
	Logic	M_ CFanHi	MCU COOLING_FAN_ HIGH SIGNAL	ON / OFF
	Logic	CS_ CFanHi	COOLING_FAN_ HIGH SIGNAL	Variable
Output	Logic	O_ CFanLo	COOLING_FAN_LOW	ON / OFF
	Logic	O_ CFanHi	COOLING_FAN_HIGH	ON / OFF

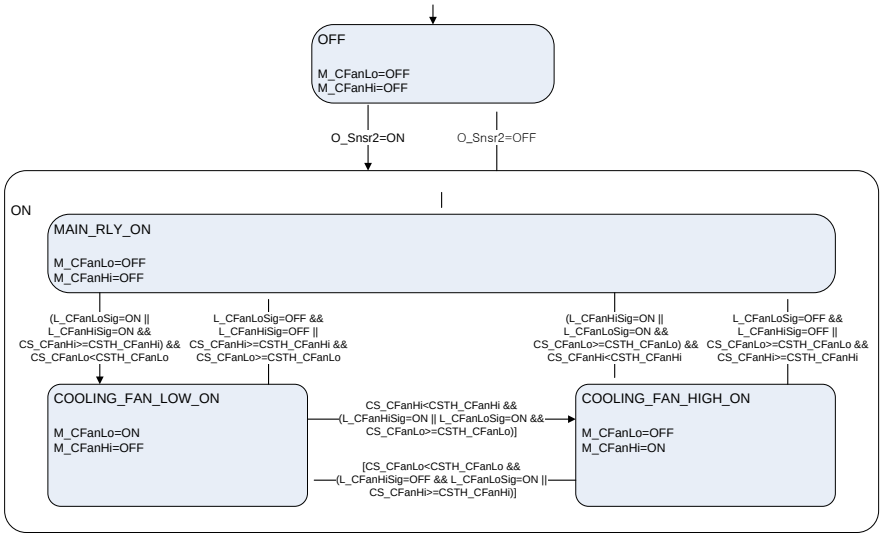

Fig. 4. Cooling Fan Module's State Diagram

Using the previously developed, Use Case modeling is performed based on a state diagram and input/output signals for development and test.

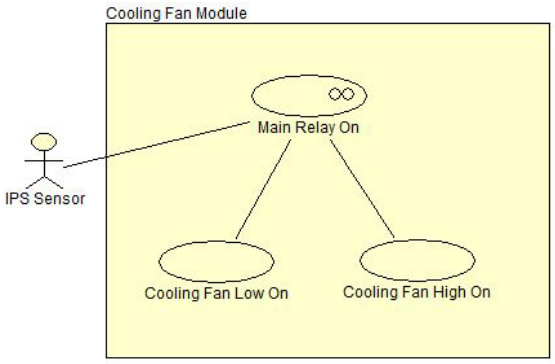

Fig. 5. Cooling Fan Module's Use Case Diagram

When the modeling is performed using the Use Case diagram in Figure 5, in order to use the Decision Factor, detailed explanations of the system need be described on the tool. As shown in Figure 6, the main relay starts when a signal is entered and after the temperature sensor compares the temperature it starts the cooling pan.

List Num	Value	Type
I1	FAN_ON	input
I2	FAN LOW or HIGH ON	input
I3	FAN LOW and HIGH OFF	input
I4	LOW_Temperature	input
I5	Under_LOW_Temperature	input
I6	Over_LOW_Temperature	input
I7	L_CFanLoSig=On	input
I8	L_CFanHiSig=On	input
I9	L_CFanLoSig=OFF	input
I10	L_CFanHiSig=OFF	input
O1	FAN_OFF	output
O2	O_COOLING_FAN_LOW	output
O3	O_COOLING_FAN_HIGH	output
C1	I1 & I5 &I9 & I10= O1	condition
C2	I2 & I4 = O2	condition
C3	I3 || I5 & I6 = O1	condition
C4	I2 & I6 = O3	condition
C5	I3 & I5 = O1	condition
C6	=	condition
C7	O_COOLING_FAN_LOW	condition

Fig. 6. Decision Factor.

Transformation between models is required to automatically generate a test case, and the CED (Cause Effect Diagram) is used to transform between models as shown in Figure 7.

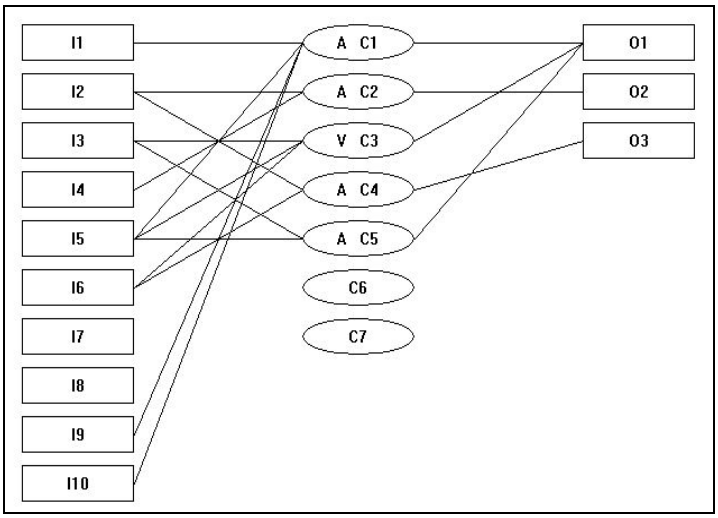

Fig. 7. Cause Effect Diagram.

Based on the automatically generated CED, Decision Factor is created as a tool as shown in Figure 8, and a test case is created as shown in Figure 9.

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
Causes:																
I1-FAN_ON	F	T	F	T	F	T	F	T	F	T	F	T	F	T	F	T
I2-FAN LOW or HIGH ON																
I3-FAN LOW and HIGH OFF																
I4-LOW_Temperature																
I5-Under_LOW_Temperature	F	F	T	T	F	F	T	T	F	F	T	T	F	F	T	T
I6-Over_LOW_Temperature																
I7-L_CFanLoSig=On																
I8-L_CFanHiSig=On																
I9-L_CFanLoSig=OFF	F	F	F	F	T	T	T	T	F	F	F	F	T	T	T	T
I10-L_CFanHiSig=OFF	F	F	F	F	F	F	F	F	T	T	T	T	T	T	T	T
Effects:																
O1-FAN_OFF	F	F	F	F	F	F	F	F	F	F	F	F	F	F	F	T
O2-O_COOLING_FAN_LOW																
O3-O_COOLING_FAN_HIGH																

Fig. 8. Decision Factorj

When the existing white box method is used, the number of test cases comes out to 86. It is 72, when our proposed method is used.

No	Post Condition	Test Condition	Expectation Result
TC 1	I1-FAN_ON=F, I5-Under_LOW_Temperature=F, I9-L_CFanLoSig=OFF=F, I10-L_CFanHiSig=OFF=F	and	O1-FAN_OFF=F
TC 2	I1-FAN_ON=T, I5-Under_LOW_Temperature=F, I9-L_CFanLoSig=OFF=F, I10-L_CFanHiSig=OFF=F	and	O1-FAN_OFF=F
TC 3	I1-FAN_ON=F, I5-Under_LOW_Temperature=T, I9-L_CFanLoSig=OFF=F, I10-L_CFanHiSig=OFF=F	and	O1-FAN_OFF=F
TC 4	I1-FAN_ON=T, I5-Under_LOW_Temperature=T, I9-L_CFanLoSig=OFF=F, I10-L_CFanHiSig=OFF=F	and	O1-FAN_OFF=F
TC 5	I1-FAN_ON=F, I5-Under_LOW_Temperature=F, I9-L_CFanLoSig=OFF=T, I10-L_CFanHiSig=OFF=F	and	O1-FAN_OFF=F
TC 6	I1-FAN_ON=T, I5-Under_LOW_Temperature=F, I9-L_CFanLoSig=OFF=T, I10-L_CFanHiSig=OFF=F	and	O1-FAN_OFF=F
TC 7	I1-FAN_ON=F, I5-Under_LOW_Temperature=T, I9-L_CFanLoSig=OFF=T, I10-L_CFanHiSig=OFF=F	and	O1-FAN_OFF=F
TC 8	I1-FAN_ON=T, I5-Under_LOW_Temperature=T, I9-L_CFanLoSig=OFF=T, I10-L_CFanHiSig=OFF=F	and	O1-FAN_OFF=F

Fig. 8. Test case

5 Conclusion

Excess time and cost is wasted in order to develop Software which has the same functions over various platforms. Also test costs for the software increases. To solve this problem, we proposed the Model Driven Architecture (MDA) based test method and tools which were used over heterogeneous embedded system's test. In this paper, we apply our proposed method to the car IPS software. We also compare test cases of using the existing white box method and using our proposed method. As a result, we extract a test case with a similar level compared to one using the existing method.

Our future work will study further on automatically extracted test case levels and coverage.

Acknowledgments. This research was supported by Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Education, Science and Technology (2012-0001845) and the Ministry of Education, Science Technology (MEST) and National Research Foundation of Korea(NRF) through the Human Resource Training Project for Regional Innovation.

Reference

1. AUTOSAR, AUTOSAR Technical Overview 3.0, http://www.autosar.org
2. A. Kleppe, J. Warmer, W. Bast: MDA Explained : The Model Driven Architecture: Practice and Promise. Addison-Wisley (2003)
3. Dong ho Kim, Woo Yeol Kim, Hyun Seung Son, R. Young Chul Kim, "A Study on Embedded Test Process based on Model Driven Architecture for Heterogeneous Embedded Testing". Korea Information Processing Society (2012)
4. John Wakins: Testing IT. Cambridge (2010)
5. B. Brockman, E. Notenboom: Embedded Software Testing. Addison Wesley (2003)
6. Woo Yeol Kim, Hyun Seung Son, Y. B. Park, B. H. Park, C. R. Carlson, R. Young Chul Kim, "Automatic MDA (Model Driven Architecture) Transformations for Heterogeneous Embedded Systems". SERP'08, LV Nevada, USA (2008)
7. Woo Yeol Kim, Hyun Seung Son, Robert Young Chul Kim, "A Study on UML Model convergence Using Model Transformation Technique for Heterogeneous Smart Phone Application". Software Engineering, Business Continuity, and Education, CCIS 257 (2011)
8. Dong ho Kim, Woo Yeol Kim, Hyun Seung Son, Hyung-Mook Lee, HeaKyung Seong, R. Young Chul Kim, "Test Case Extraction based on Use Case Approach for Heterogeneous Embedded System". The 1st YEllow Sea Interntional Conference on ubiquitous Computing (2011)