Information Technology and Computer Science

Proceedings
International Conference, ITCS 2012
Porto, Portugal, July 2012
A Performance Model for Hypercube Based NoC with Fully Adaptive Routing Algorithm..........................68
   Jin Liu, Xiaofeng Wang, Hongmin Ren, Jin Wang, Jeong-Uk Kim

Knowledge Modeling Guideline for Reentrant Features in SAGE ..................76
   Jeong Ah Kim, InSook Cho, BinGu Shim, Min Hee Choi, SunTae Kim,
   Sun Ah Kim, Yongho Kim, Chang Hee Lee, Heeseong Yun
   and Dongkyu Seo

Test Case Extraction for Intelligent Power Switch of Heterogeneous Vehicles...82
   Dong Ho Kim, Hyun Seung Son, Woo Yeol Kim,
   and Robert Young Chul Kim

A Study on Requirement Extraction on Use Case Approach ......................89
   Bo Kyung Park, and Robert Young Chul Kim

Query Language for Business Process Framework based
on Closed Architecture ........................................................................94
   Chae Yun Seo, and Robert Young Chul Kim

Verification of Requirements Extraction and Prioritization
using Use Case Points.............................................................................100
   So Young Moon, Bo Kyung Park, and Robert Young Chul Kim

Automatic Test Case Generation using Multiple Condition Control
Flow Graph..............................................................................................105
   Hyun Seung Son, Woo Yeol Kim, Jae Seung Kim,
   and Robert Young Chul Kim

A Multi-agent based Facility Maintenance Planning and Monitoring
System: A Case Study...............................................................................110
   JaeHoon Lee, MyungSoo Lee, SangHoon Lee, SeGhok Oh,
   and JoongSoon Jang

The Archiving Method for Records of Public Sector’s Facebook Page..........117
   Yun-Young Hwang, In-Ho Jang, Seung-Jun Cha and Kyu-Chul Lee

A Study on Architecture of a Cyber Checkpoint Model in the Cloud
Computing Environment.........................................................................123
   Min-Woo Park, Jung-Ho Eom, Sung-Hwan Kim, Nam-Uk Kim,
   and Tai-Myoung Chung
Test Case Extraction
for Intelligent Power Switch of Heterogeneous Vehicles

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

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 dedicated 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.

![Model Driven Development and Test Process](image)

**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-specific 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.

![Use Case Diagram](image)

**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.

![Junction Box](image)

**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

<table>
<thead>
<tr>
<th>Function</th>
<th>Name</th>
<th>Description</th>
<th>Status</th>
</tr>
</thead>
<tbody>
<tr>
<td>Input</td>
<td>Hardware</td>
<td>O_SNSR2 SENSOR2 IPS</td>
<td>ON/OFF</td>
</tr>
<tr>
<td></td>
<td>Hardware</td>
<td>L_CFanLoSig COOLING_FAN_LOW SIGNAL</td>
<td>ON/OFF</td>
</tr>
<tr>
<td></td>
<td>Hardware</td>
<td>L_CFanHiSig COOLING_FAN_HIGH SIGNAL</td>
<td>ON/OFF</td>
</tr>
<tr>
<td>Logic</td>
<td>M_CFanLo</td>
<td>MCU COOLING_FAN_LOW SIGNAL</td>
<td>ON/OFF</td>
</tr>
<tr>
<td>Logic</td>
<td>CS_CFanLo</td>
<td>COOLING_FAN_LOW SIGNAL</td>
<td>Variable</td>
</tr>
<tr>
<td>Logic</td>
<td>M_CFanHi</td>
<td>MCU COOLING_FAN_HIGH SIGNAL</td>
<td>ON/OFF</td>
</tr>
<tr>
<td>Logic</td>
<td>CS_CFanHi</td>
<td>COOLING_FAN_HIGH SIGNAL</td>
<td>Variable</td>
</tr>
<tr>
<td>Output</td>
<td>Logic</td>
<td>O_CFanLo COOLING_FAN_LOW</td>
<td>ON/OFF</td>
</tr>
<tr>
<td>Logic</td>
<td>O_CFanHi</td>
<td>COOLING_FAN_HIGH</td>
<td>ON/OFF</td>
</tr>
</tbody>
</table>

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.

<table>
<thead>
<tr>
<th>List Num</th>
<th>Value</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>11</td>
<td>FAN_ON</td>
<td>input</td>
</tr>
<tr>
<td>12</td>
<td>FAN LOW or HIGH ON</td>
<td>input</td>
</tr>
<tr>
<td>13</td>
<td>FAN LOW and HIGH OFF</td>
<td>input</td>
</tr>
<tr>
<td>14</td>
<td>LOW_Temperature</td>
<td>input</td>
</tr>
<tr>
<td>15</td>
<td>Over_LOW_Temperature</td>
<td>input</td>
</tr>
<tr>
<td>16</td>
<td>_LC FanLoSig=On</td>
<td>input</td>
</tr>
<tr>
<td>17</td>
<td>_LC FanHiSig=On</td>
<td>input</td>
</tr>
<tr>
<td>18</td>
<td>_LC FanLoSig=OFF</td>
<td>input</td>
</tr>
<tr>
<td>19</td>
<td>_LC FanHiSig=OFF</td>
<td>input</td>
</tr>
<tr>
<td>01</td>
<td>FAN_OFF</td>
<td>output</td>
</tr>
<tr>
<td>02</td>
<td>O_COOLING_FAN_LOW</td>
<td>output</td>
</tr>
<tr>
<td>03</td>
<td>O_COOLING_FAN_LOW</td>
<td>output</td>
</tr>
<tr>
<td>C1</td>
<td>12 &amp; 13 &amp; 14 &amp; 15 = 01</td>
<td>condition</td>
</tr>
<tr>
<td>C2</td>
<td>12 &amp; 14 = 02</td>
<td>condition</td>
</tr>
<tr>
<td>C3</td>
<td>13 &amp; 15 &amp; 16 = 01</td>
<td>condition</td>
</tr>
<tr>
<td>C4</td>
<td>12 &amp; 16 = 03</td>
<td>condition</td>
</tr>
<tr>
<td>C5</td>
<td>13 &amp; 15 = 01</td>
<td>condition</td>
</tr>
<tr>
<td>C6</td>
<td>=</td>
<td>condition</td>
</tr>
<tr>
<td>C7</td>
<td>O_COOLING_FAN_LOW</td>
<td>condition</td>
</tr>
</tbody>
</table>

**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.
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.

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