Enhancing the ESIM (Embedded Systems Improving Method) by Combining Information Flow Diagram with Analysis Matrix for Efficient Analysis of Unexpected Obstacles in Embedded Software

著者	Shinyashiki Yasufumi, Mise Toshiro, Hashimoto Masaaki, Katamine Keiichi, Ubayashi Naoyasu, Nakatani Takako
journal or publication title	Software Engineering Conference, 2007. APSEC 2007. 14th Asia-Pacific
page range	326-333
year	2008-01-07
URL	http://hdl.handle.net/10228/00006295

doi: info:doi/10.1109/ASPEC.2007.81

Enhancing the ESIM (Embedded Systems Improving Method) by Combining Information Flow Diagram with Analysis Matrix for Efficient Analysis of Unexpected Obstacles in Embedded Software

Yasufumi Shinyashiki
Matsushita Electric Works, Ltd.
yasufumi@mail.mew.co.jp
Masaaki Hashimoto
Kyusyu Institute of Technology
hasimoto@ai.kyutech.ac.jp
Naoyasu Ubayashi
Kyusyu Institute of Technology
ubayashi@ai.kyutech.ac.jp

Toshiro Mise
Matsushita Electric Works, Ltd.
mise@mail.mew.co.jp
Keiichi Katamine
Kyusyu Institute of Technology
katamine@ci.kyutech.ac.jp
Takako Nakatani
University of Tsukuba
nakatani@gssm.otsuka.tsukuba.ac.jp

Abstract

In order to improve the quality of embedded software, this paper proposes an enhancement to the ESIM (Embedded Systems Improving Method) by combining an IFD (Information Flow Diagram) with an Analysis Matrix to analyze unexpected obstacles in the software. These obstacles are difficult to predict in the software specification. Recently, embedded systems have become larger and more complicated. Theoretically therefore, the development cycle of these systems should be longer. On the contrary, in practice the cycle has been shortened. This trend in industry has resulted in the oversight of unexpected obstacles, and consequently affected the quality of embedded software. In order to prevent the oversight of unexpected obstacles, we have already proposed two methods for requirements analysis: the ESIM using an Analysis Matrix and a method that uses an IFD. In order to improve the efficiency of unexpected obstacle analysis at reasonable cost, we now enhance the ESIM by combining an IFD with an Analysis Matrix. The enhancement is studied from the following three viewpoints. First, a conceptual model comprising both the Analysis Matrix and IFD is defined. Then, a requirements analysis procedure is proposed, that uses both the Analysis Matrix and IFD, and assigns each specific role to either an expert or non-expert engineer. Finally, to confirm the effectiveness of this enhancement, we carry out a description experiment using an IFD.

1. Introduction

Embedded systems are used by various users who are unaware of the existence of the systems in different environments [1], [2]. Furthermore, these systems are required to provide a safe, reliable service over a long period. As a result, 70% or more of the source code of embedded systems is generally allocated to exception handling. It is expected that as embedded systems grow larger in scale and become more complicated [3], the development cycle would lengthen. On the contrary, in reality this development cycle has actually shortened, with the result that it has become more difficult to take into account all the exception conditions. Furthermore, software engineers are required to have knowledge not only of the software, but also of devices, users and environments to be able to recognize exception conditions in the embedded software [4]. Occasionally however, software engineers do overlook exception conditions expected at times. In fact, the failure of products occurs mainly because exception conditions have not been foreseen and a re-design of the software becomes necessary. This means that we can expect to improve the quality and productivity of the embedded software by reflecting the exception conditions more accurately in the specification.

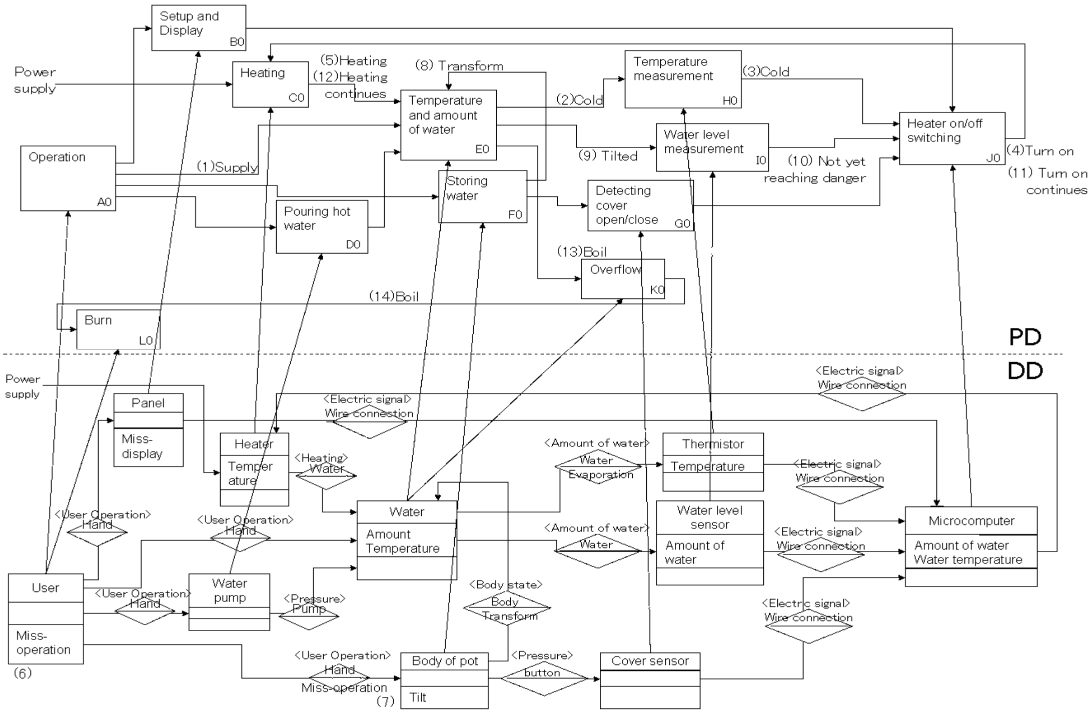

Fig. 1. Example of an IFD

We call these exception conditions “unexpected obstacles” and have studied requirements analysis methods to prevent overlooking unexpected obstacles in embedded systems [5], [6], [7], [8]. In this paper, the term “expected specifications” specifies the usual system behavior described in the software operation manual and explicitly defined at the start of the architectural design, whereas “unexpected obstacle specifications” are concerned with any deviations from the usual behavior. Examples of such deviations are fading and failure of system hardware, incorrect operation and overload caused by system users, and temperature or radio noise in the natural environment. Of course, the unexpected obstacle specifications should also be explicitly defined in the system specifications. However, they are referred to as "unexpected obstacle specifications" throughout the entire system development process, in order to distinguish them from the expected specifications, and because they are sometimes left undefined.

We have studied two analysis methods for unexpected obstacles by formalizing a method that is tacitly used by experts in embedded software development. We have already proposed using an Information Flow Diagram (IFD) in one of the methods [6]. In the other method, which we call ESIM, we proposed the use of an Analysis Matrix [7]. Compared with other existing analysis methods [4], [9], [10], [11], [12], [13], these methods incorporate both top-down and bottom-up analysis approaches to prevent overlooking unexpected obstacles. Until now, it has been assumed that these two methods are used separately. However, if we combine them to form a new method, we can expect the cost-effectiveness of analyzing unexpected obstacles accruing to each of the two methods according to the experiments. In this paper, we therefore propose an enhancement to ESIM by combining an IFD with an Analysis Matrix.

In Section 2, we describe briefly an IFD, the ESIM and the requirements for the enhancement. In Section 3, we propose the enhancement to the ESIM. Section 4 discusses the enhancement. Our conclusions are presented in Section 5.

2. IFD, ESIM and requirements

We first give a brief description of an IFD and the ESIM. Then, we specify the requirements for the enhancement.

state event	The kettle is in an empty state.	The kettle is in the filled with water state.	The water level sensor is in the state of sensing the level.	The kettle is not in the filled with water state and the heater is on.	The water level sensor sensing the level, but the real water level is out of recognition, The kettle is not in the filled with water state and the heater is on, The kettle is tripped.
Supply water.	Turning on the heater	The water may run over	Reaching the filled with water level	Reaching the filled with water level	Before receiving this event, the kettle may burn the user
Water level sensor turns off the heater.	SAFE	Stop boiling water.	SAFE	SAFE	SAFE
Something tips the kettle.	SAFE	The water may run over	Cannot recognize the real water level	The water level reaches the filled with water level	-----
A user touched the kettle.	SAFE	SAFE	SAFE	SAFE	The kettle may burn the user

Fig. 2. Example of an Analysis Matrix

2.1. An IFD

An IFD is composed of two diagrams, the Process Diagram (PD) and the Device Diagram (DD), and the connections between these diagrams. The PD represents the processes in the embedded system and the information flow between them. The DD represents the devices in the embedded system, the objects, such as users, in the operational environment and the communication carriers between them. Fig. 1 shows an example of an IFD. The upper and lower parts are separated by a dotted line, and specify the PD and DD respectively.

When we analyze unexpected obstacles with an IFD, we can logically trace the causal relation of the failure from the causes to the final result in the IFD. For example, a failure scenario is outlined by the numbered sequence (1), (2), (3), etc. in the figure. Because of the traceability, non-expert engineers can analyze unexpected obstacles under the leadership of expert engineers. However, much information about devices, carriers, processes and information flows needs to be specified in the IFD.

2.2. ESIM and an Analysis Matrix

The ESIM is composed of two phases: device failure extracting phase and failure scenario constructing phase. In the former phase, by applying guide words, we assume unexpected phenomena to be deviations from the component functions. Then, we extract component device failures from the unexpected phenomena by applying FTA. In the latter phase, we extract unexpected states or events by analyzing the unexpected phenomena, and add these states or events to the Analysis Matrix, an example of which is given in Fig. 2. New unexpected phenomena may occur because of the added states or events. This procedure is repeated until no new unexpected phenomena are discovered, and we can then construct the failure scenarios that are often overlooked by the software engineers during the requirements analysis and design phases.

ESIM has two characteristics. One is that we can find unexpected states and events which are overlooked by the software design engineers. The other is that we extend the Analysis Matrix from which we can easily obtain all the combinations of states and events. However, only expert engineers can use the ESIM in practice since the granularity of the information contained in the Analysis Matrix is coarser than that in the IFD. However, the ESIM requires less information to be specified than for the IFD.

2.3. Requirements

By experimenting with the ESIM and an IFD, we obtained the above-mentioned results, which show that the characteristics of each method are contradictory. With regard to engineer skills, only expert engineers can use the ESIM, whereas analysis by non-expert engineers using an IFD only requires leadership from expert engineers. Regarding the quantity of information to be specified, an IFD needs more

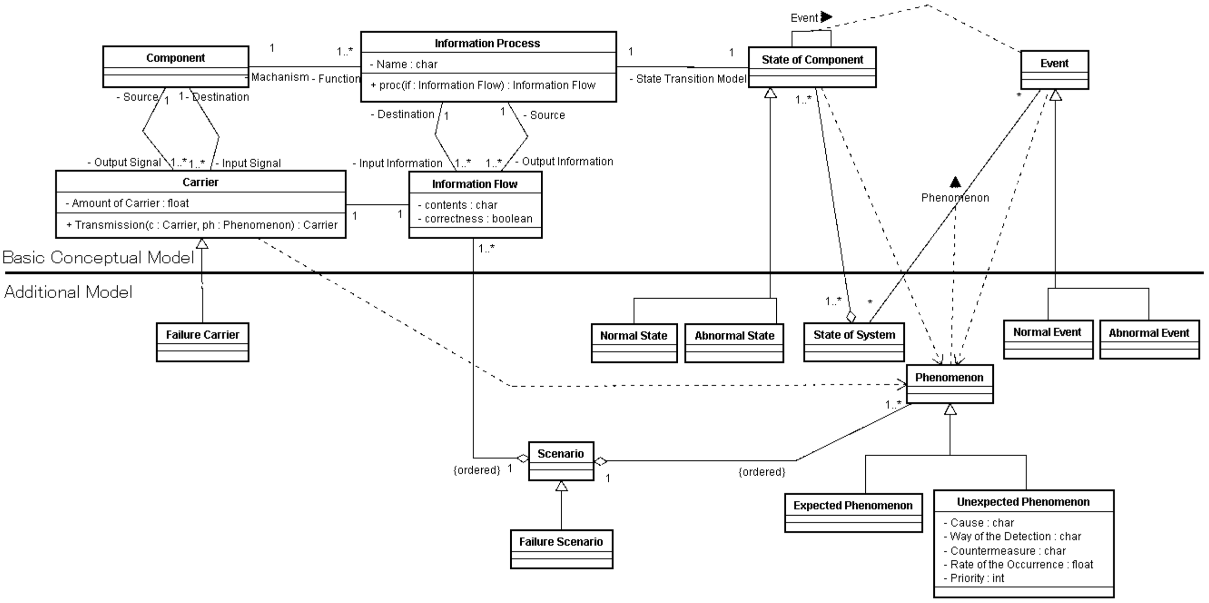

Fig. 3. Conceptual Model of Requirement Analysis Objects

information than the ESIM. This means, of course, that the ESIM is more efficient than an IFD in analyzing unexpected obstacles although relying on expert engineers is more costly than using non-expert engineers.

In practice, it is difficult for a project manager to find several expert engineers. On the other hand, it is not difficult to obtain non-expert engineers. Therefore, we propose a cost-effective analysis method operable by a team of expert and non-expert engineers, by combining an IFD with the ESIM. In this method, expert engineers lead non-expert engineers by using the ESIM and IFD respectively.

3. Enhancing ESIM by combining IFD with Analysis Matrix

In this section, we propose an enhancement to the ESIM. We start with a conceptual model of an embedded system, which we call the “Basic Conceptual Model”. We then add to this basic model the concepts that expert engineers use for understanding embedded systems, and call this the “Additional Model”. Thereafter, we propose an enhancement to the ESIM by combining an IFD with an Analysis Matrix in order to satisfy the requirements specified in Section 2.3.

3.1. Conceptual Model of Requirement Analysis Objects

Fig. 3 shows the conceptual model of requirement analysis objects when expert engineers analyze unexpected obstacles in embedded systems. This model is separated into two parts by the horizontal line. The upper part of the model shows the basic conceptual model. These concepts are considered by the engineers during the design phase of the expected specifications. However, when the engineers design unexpected obstacle specifications, they introduce new concepts. The lower part of the model shows the additional model. The concepts in the additional model have the following meanings:

- **Normal State (Expected State)** means a state that is defined in the expected specifications.
- **Abnormal State (Unexpected State)** means a state that is not defined in the expected specifications. Thus the Abnormal States consist, not only of fatal failures such as out of component works, but also of irregular states, such as processing overloads.
- **Normal Event (Expected Event)** means an event that is defined in the expected specifications.
- **Abnormal Event (Unexpected Event)** means an event that is not defined in the expected specifications. For example, it may be classified as a normal event when a motor transmits power to another component. However, assume that the motor makes a noise at the same time. If software engineers do not take this into consideration, it is classified as an abnormal event.

- **Failure Carrier** specifies a carrier that has an irregular value because of the effects of some phenomenon. If a carrier of this kind reaches a destination component, the information process of the component will not understand the original meaning. An example of a failure carrier is an irregular voltage signal caused by a short in the signal line.
- **State of the System** addresses the entire state, including environment. We note that this definition is different from the general definition of the state of the system, which is usually defined as the Cartesian product of the states of all components at a particular moment. In our definition, however, it is defined as the Cartesian product of the states of those components on which the analysis is focused.
- **Phenomenon** addresses the phenomenon that occurs by combining the states of the system, even as defined above. The occurrence of the phenomenon means that there is a transition of the states of some components and/or sending events. This is shown as a dependency in the conceptual model of the requirement analysis objects.
- **Expected Phenomenon** addresses a phenomenon judged not to become a failure or to cause a failure in the system or environment.
- **Unexpected Phenomenon** addresses a phenomenon that is a failure in the system or environment or the cause of such a failure.
- **Scenario** means an ordered information flow or ordered phenomena and has meaning, which is given by a constraint.
- **Failure Scenario** addresses a scenario that results in failure.

Expert engineers analyze unexpected obstacles using either a top-down analysis like FTA [9], a bottom-up analysis like FMEA [10] or deflection analysis like HAZOP [11]. We now show how each of these analyses is carried out on the conceptual model.

FTA is used to analyze the causes of the failure specified at the root of a tree; it proceeds from the root to the leaves in a stepwise manner. In the conceptual model, we show two top-down analyses like FTA. One of these is an analysis method in which engineers specify unexpected phenomena at the root of a tree and arrive at the causes by moving from the results toward the causes using a state transition model. The other is an analysis method in which engineers specify an information flow which assigns failure at the root of a tree and arrives at the causes by moving from the destination toward the sources in an information communication model.

FMEA is used to analyze the failures that occur as a result of problems with the components in a bottom-up manner. In the conceptual model, we show two bottom-up analyses similar to FMEA. One of these is an analysis method where engineers specify abnormal states of a component, instead of the problems with the component, at the root of the tree and then analyze the failures that occur as a result of combining with other states of the component and/or events using a state transition model. The second is an analysis method where engineers specify abnormal states of the component or failure carrier at the root of a tree and analyze the failures that occur as a result of combining with other information in an information communication model.

HAZOP was originally used for analyzing the safety of plants, and assumes deviational phenomena that have a negative impact on the plant. In the conceptual model, the analysis process relies on the analysis of deviational phenomena as in HAZOP. Engineers expect the failure of carriers, abnormal states of components, and abnormal states caused by the deviation of carrier, state of component or event.

3.2. Combining an IFD with an Analysis Matrix

In this section, we discuss combining an IFD with an Analysis Matrix, which requires two issues to be considered. The first of these involves matching the concepts of the ESIM and an IFD to avoid repetition of concepts used in both the IFD and ESIM. This includes three topics. First, we explain how to describe a failure scenario. In an IFD, a failure scenario is depicted by information flow. On the other hand, in the ESIM, a failure scenario is described by phenomena. To enable non-expert engineers to deal with failure scenarios, we describe a failure scenario using information flow. Next, we explain the relationship between devices in an IFD and components in ESIM. We treat these concepts as one in the enhanced ESIM because they represent the same thing. Finally, we explain the relationship between the set of concepts, carriers and information

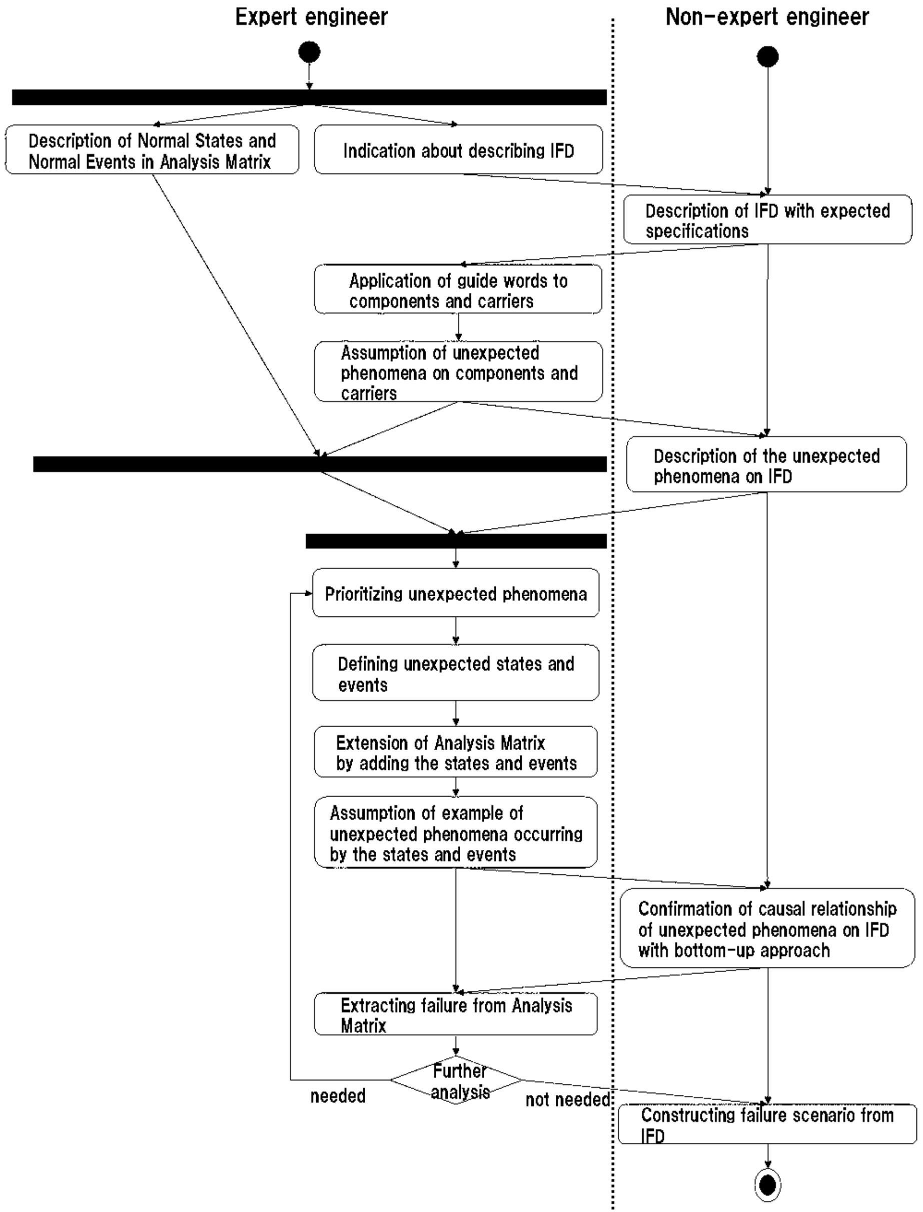
Expert engineer
Non-expert engineer
Description of Normal States and Normal Events in Analysis Matrix
Indication about describing IFD
Description of IFD with expected specifications
Application of guide words to components and carriers
Assumption of unexpected phenomena on components and carriers
Description of the unexpected phenomena on IFD
Prioritizing unexpected phenomena
Defining unexpected states and events
Extension of Analysis Matrix by adding the states and events
Assumption of example of unexpected phenomena occurring by the states and events
Confirmation of causal relationship of unexpected phenomena on IFD with bottom-up approach
Extracting failure from Analysis Matrix
Further analysis
needed
not needed
Constructing failure scenario from IFD

Fig. 4. A procedure in the enhanced ESIM

flow in an IFD and the communication contents in the ESIM. To enable non-expert engineers to deal with failure scenarios, we choose the solution that allows engineers to describe a more detailed failure scenario. A deviation of communication content in the ESIM is the same as recognizing a wrong meaning at a destination component because of deviation of the carrier in an IFD. Therefore, we use the concept of information flow and carrier in the enhanced ESIM.

The second consideration is the analysis procedure and role assignment. Fig. 4 shows an analysis procedure in the enhanced ESIM using an IFD and an Analysis Matrix. To allow the analysis to be led by an expert engineer, this procedure is based on the ESIM procedure. When the analysis process reaches the end of this procedure, we obtain failure scenarios and feedback for software engineers. Furthermore, we assign the roles for each process to expert or non-expert engineers as shown in Fig. 4. This assignment is based on the requirements shown in Section 2.3.

3.3. Application example

We show an example of the description of a failure scenario using an IFD and Analysis Matrix. The failure scenario is constructed from the sample specifications for an electric kettle. The kettle is equipped with a water sensor that senses the water level. If the sensor detects the filled water state, it turns off the heating element to prevent the boiling water from overflowing. When a user supplies water to the kettle, a thermistor senses the falling temperature, and this event turns on the heating element, which in time causes the kettle to enter the boiling water state. However, there is the possibility of an unexpected event happening, such as something tipping the kettle. Depending on the angle of the kettle, the filled water sensor may not recognize that the actual water level has reached the danger point, and the heating element remains in the state of "boiling". Ultimately, the boiling water overflows and may burn the user.

In Fig. 1, we indicate the failure scenario by attaching numerical comments, e.g. (1), (2), (3), etc.. In Fig. 2, we show the same failure scenario. If we compare both descriptions, we find that the ESIM omits the detail of the failure scenario. The detail corresponding to (2), (3), (5), (7), (8), (9), (11), (12) and (13) in Fig. 1 is not present in Fig. 2.

4. Discussion

This section discusses the enhancement to the ESIM by combining an IFD and an Analysis Matrix.

4.1. Effect of enhancing the ESIM

To confirm the effects of this enhancement, we devised a description experiment using an IFD and based on the results of the ESIM. The target embedded system is implemented as a new product in the C programming language. During the development of this product, an expert engineer applied the ESIM to analyze failure scenarios. The expert engineer referred the requirements specification of the product for analysis. Applying the ESIM resulted in the expert engineer finding 44 failure scenarios in 32 hours. We gave 9 of these failure scenarios to testers to describe using IFDs. The testers were 3 students who had no development experience of the embedded software. In the experiment, the testers were allowed to question us freely. We recorded their questions, and analyzed the content. Ultimately, the testers were able to describe each of the scenarios using an IFD. The results of our analysis are as follows:

- The method of mapping from a description of a failure scenario given in natural language to an IFD is useful for non-expert engineers.
- It was difficult for the non-expert engineers to distinguish the control information and input information in a PD.
- Some of the non-expert engineers were not sufficiently able to distinguish between information process and information flow.

To assist testers with respect to the above-mentioned points, an expert engineer was available to support them in all the non-expert engineer's processes specified in Fig. 4. We expect that the effect of an actual application of this method in companies would result in the expert engineer's load being reduced since the non-expert engineers would be more knowledgeable about embedded software than the testers in this experiment.

4.2. Future work

In this paper, we have proposed an enhancement to the ESIM by combining an IFD and an Analysis Matrix and have predicted the effect of this enhancement. In the future, we aim to confirm the effectiveness of the enhanced ESIM quantitatively by experimenting with real applications. Thereafter, we aim to reduce the cost of application of the enhanced ESIM by formalizing the analysis processes which are done by expert engineers in this paper. There are two future works. One of these is the formalization of the analysis method by applying qualitative reasoning with concepts of states provided by an Analysis Matrix and constraints given by an IFD. This formalization will lead to the formalization of knowledge about unexpected obstacles owned by expert engineers. Furthermore, the development of a knowledge base about unexpected obstacles will also be undertaken. The knowledge base is expected to be connected with CASE tools. Other future work will focus on the character of an IFD as a directed graph. Based on this, we will study the application of the techniques of graph analysis to analyze loss, connections and loops which provide feedback [5].

5. Conclusion

This paper has proposed an enhancement to the ESIM by combining an IFD and an Analysis Matrix. In this enhancement, we considered the following three issues: establishing the conceptual model, defining the analysis procedure, and assigning the roles of expert and non-expert engineers. Furthermore, we have predicted the effectiveness of this enhancement by carrying out a description experiment using an IFD.

In the future, we aim to confirm the effectiveness of the enhanced ESIM quantitatively by experimenting with real applications. We will also study the formalization of the analysis method by applying qualitative reasoning theory.

Acknowledgment

The authors wish to thank Mr.Tanabe, Mr.Tanimoto, Mr.Inoue and Mr.Kubo in the Hashimoto Lab. for their help with the experiments in this research.

References

[1] T.Nakatani, T.Mise, Y.Shinyashiki, K.Katamine, N.Ubayashi, and M.Hashimoto, "Toward defining a system boundary based on real world analysis", *Proc. of the FOSE2005*, Japan Society for Software Science and Technology, Kindai Kagaku Sha, 2005, pp. 221-226 (in Japanese).

[2] T.Sumi, M.Hirayama, and N.Ubayashi, "Analysis of the external environment for embedded systems," *IPSJ SIG Technical Reports*, 2004-SE-146, 2004, pp. 33-40 (in Japanese).

[3] Ministry of Economy, Trade and Industry, editor, *Report of actual field survey of embedded software 2005 Edition*, Ministry of Economy, Trade and Industry, 2005 (in Japanese).

[4] R.Crook, D.Ince, L.Lin, and B.Nuseibeh, "Security Requirements Engineering: When Anti-Requirements Hit the Fan", *Proc. of the 10th Anniversary Joint IEEE International Requirements Engineering Conference (RE'02)*, 2002, pp. 203-205.

[5] H.Hatanaka, Y.Shinyashiki, T.Mise, H.Kametani, M.Hashimoto, N.Ubayashi, K.Katamine, and T.Nakatani, "An Analysis of Information Flow Graph based on Conceptual Model of Exceptions in Embedded Software", *TECHNICAL REPORT OF IEICE 104-431*, 2004, pp. 19-24.

[6] H.Kametani, Y.Shinyashiki, T.Mise, M.Hashimoto, N.Ubayashi, K.Katamine, and T.Nakatani, "Information Flow Diagram and Analysis Method for Unexpected Obstacle Specification of Embedded Software", *Proc. of the Knowledge-Based Software Engineering (JCKBSE'06)*, 2006, pp. 115-124.

[7] T.Mise, M.Hashimoto, K.Katamine, Y.Shinyashiki, N.Ubayashi, and T.Nakatani, "A Method for Extracting Unexpected Scenarios of Embedded Systems", *Proc. of the Knowledge-Based Software Engineering (JCKBSE'06)*, 2006, pp. 41-50.

[8] Y.Shinyashiki, T.Mise, M.Hashimoto, K.Katamine, N.Ubayashi, T.Nakatani, "A Requirements Analysis Method of Unexpected Obstacles in Embedded Software by Applying Information Flow Diagram", *IPSJ Special Issue on embedded system*, 2007(To be appeared)(In Japanese).

[9] N.G.Leveson, "Fault Tree Analysis", *Safeware: System Safety and Computers*, Addison-Wesley, 1995, pp. 317-326.

[10] N.G.Leveson, "Failure Modes and Effects Analysis", Safeware: System Safety and Computers, Addison-Wesley, 1995, pp. 341-344.

[11] N.G.Leveson, "HaZards and Operability Analysis", *Safeware: System Safety and Computers*, Addison-Wesley, 1995, pp. 335-341.

[12] I.Alexander, "Misuse cases, use cases with hostile intent", *IEEE Software*, vol.20, no.1, 2003, pp. 55-66.

[13] A.V.Lamsweerde, E.Letier, "Handling Obstacles in Goal-Oriented Requirements Engineering", *IEEE Transactions on Software Engineering*, vol.26, no.10, 2000, pp. 978-1005.