An Evaluation of Argument Patterns Based on Data Flow

Shuichiro Yamamoto

An Evaluation of Argument Patterns based on Data flow

Shuichiro Yamamoto

Strategy Office, Information and Communications Headquarters
Nagoya University
Nagoya, Japan
syamamoto@ acm.org

Abstract. In this paper, we will introduce some of the problem areas that software engineers are susceptible during the creation of assurance cases, based on the author's educational experience with assurance cases. To mitigate these problems, assurance case patterns are proposed based on Data flow diagrams that help engineers develop assurance cases by reusing those patterns. It is also shown an evaluation result of assurance case pattern application to develop an assurance case for a smart card application system.

Keywords: assurance case, argument pattern, data flow diagram, experimental evaluation, smart card application

1 Introduction

The safety case, the assurance case, and the dependability case are currently the focus of considerable attention for the purpose of verifying that systems are safe. Methods have thus been proposed for representing these using Goal Structuring Notation (GSN)[1][2][3]. However, in order to facilitate the creation of assurance cases by engineers during real-world system development, it is not enough to simply provide them with an editor. They also need a more concrete creation method for assurance cases that has been adapted to suit the system-development process and documentation.

Against this backdrop, a number of methods have been developed for safety cases and dependability cases as part of research in the field of assurance cases: For example, Kelly has proposed the following six-step method for GSN creation: (1) Identify the goals to be supported; (2) define the basis on which the goals are stated, (3) identify a strategy to support the goals, (4) define the basis on which the strategy is stated, (5) evaluate the strategies, and (6) identify the basic solution[1][2]. The Safety Case Development Manual[4] established by the European Organization for the Safety of Air Navigation identifies the establishment of contexts for safety cases as being extremely important. This manual also proposes a checklist for the review of safety cases.

In terms of the development process for a system that itself comprises multiple systems (i.e., a system of systems), a technique involving system analysis, goal elicita-

tion, identification of candidate design alternatives, and resolution of conflicts has been proposed for the creation of assurance cases in a structured fashion[5]. Meanwhile, methods for the decomposition of arguments as required when creating assurance cases have been arranged into categories such as architecture, functional, and set of attributes [6]. Goodenough, Lipson and others proposed a method to create Security Assurance case [7]. They described that the Common Criteria provides catalogs of standard Security Functional Requirements and Security Assurance Requirements. They decomposed Security case by focusing on the process, such as requirements, design, coding, and operation. The approach did not use the Security Target structure of the CC to describe Security case. Alexander, Hawkins and Kelly overviewed the state of the art on the Security Assurance cases [8]. They showed the practical aspects and benefits to describe Security case in relation to security target documents. However they did not provide any patterns to describe Security case using CC.

Kaneko, Yamamoto and Tanaka proposed a security countermeasure decision method using Assurance case and CC [9]. Their method is based on a goal oriented security requirements analysis [10-11]. Although the method showed a way to describe security case, it did not provide Security case graphical notations and the seamless relationship between security structure and security functional requirements. Yamamoto, Kaneko and Tanaka have proposed assurance case patterns based on security common criteria [12]. In addition, Yamamoto and Matsuno also evaluated the effectiveness of argument patterns for LAN device management system [13].

The diversity of these techniques is evidence of assurance-case creation methods being proposed on an individual basis for a range of different development processes and fields of application. However, in order that the assurance case may be used to verify that real-world systems are dependable, its specific correlation with the system development process and stage deliverables and its mode of use must be clear and consistent. In this regard, many improvements must still be made to today's methods for creating assurance cases.

This paper discusses the effectiveness of argument patterns of assurance case based on data flow diagram, DFD. Section 2 proposes argument patterns based on DFD. Section 3 shows the experiment of applying these argument patterns to assure dependability of a smartcard application. In section 4, we discuss the effectiveness of the proposed argument patterns. Section 5 concludes the paper and shows future work.

2 Argument Patterns based on DFD

We will look briefly at the DFD level decomposition pattern.

2.1 DFD Level Decomposition Pattern

An assurance case must be created for a system analyzed using data flow diagrams (DFD). The assumed condition for application of DFD level decomposition is the system being clearly definable using data flow diagrams.

Claims are decomposed based on the hierarchical levels of DFD. Data flow diagrams take the form of (1) a top-level context diagram that defines data flow between the system and external agents; (2) lower-level data flow diagrams that are hierarchically decomposed from the context diagram for individual process; (3) process specifications for processes that cannot be decomposed any further; and (4) data stores for retaining data. Accordingly, assurance cases for DFD levels can be created using the following procedure.

Step 1: The claim "The system is dependable" is decomposed in line with the process decomposition pattern and on the basis of the context diagram. Here, "Definition of data flow diagrams" is connected to this claim as an assumption.

Step 2: If processes can be decomposed;
An assurance case is created for each process in line with the process decomposition pattern. Because the assurance case will contain a parent node corresponding to the upper-level process, the process structure definition is connected to the parent node as an assumption node at this time.

Step 3: If processes cannot be decomposed, the following two options are available.

Step 3-1: When the process specification has been defined;
An assurance case is created for the process specification. Because the process specification will describe the corresponding processing, risk can be analyzed for each processing step at this time and evidence of countermeasures for the identified risks can be displayed.

Step 3-2: When the process specification has not been defined;
An undefined element is connected to the claim.

3 Evaluation of argument patterns

3.1 Design of experiment

The experiment was conducted to evaluate the effectiveness of assurance case patterns for the real smartcard based security application. Examinee is two engineers who have several years of experience in the smartcard system development. 4 hour course of assurance case education was provided to the examinee.

The examinees, then, developed the assurance case for the target system described below.

3.2 Overview of the target system

The Employee Attendance Management System (EAMS) consists of Server, Smartcard readers, Android terminals, Manager's PCs, and employee's PCs. These components are connected by intranet. The purpose of EAMS is to manage attendance in-

formation of employees with smartcards. Smartcards are monitored by readers that are controlled by Android terminals.

The Server gathers smartcard attendance information through Android terminals. Each terminal monitors smartcards of employees through readers in each location. The employees can register attendance schedule by user PCs in regional offices. Managers also execute employee management based on the attendance information of employees by Managers PCs.

3.3 Result of the experiment

The examinee developed the assurance case in 280 man hours during 2 month. There were 5 work items, learn D-Case, understand specification, develop DFD, analyze risk, and develop D-Case. D-Case is an abbreviation of Dependability Case that means assurance case for assuring dependability. D-Case was described by D-Case editor [14] that was originally developed by Tokyo University and enhanced currently by Nagoya University.

43% of time was used to describe D-Case diagram. 35% of time was also used to analyze risk of EAMS. 11% and 8% of total time were consumed to understand specification and develop DFD. 3% of the application time was used to learn D-Case. The pattern application time was included in D-Case development.

3.4 Examples of developed assurance case

Fig.1 shows the lower level assurance case example developed for the claim G_5 by using iteratively the DFD decomposition pattern according to the hierarchy of the DFD specifying EAMS.

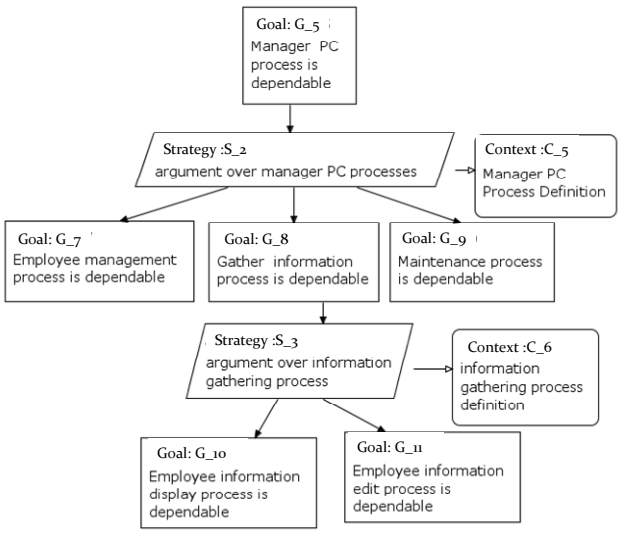

Fig. 1. Assurance case tree developed by using DFD decomposition pattern

3.5 Volume of the assurance case

There were 33 processes in DFD. There were 549 nodes of the assurance case in total. The ratio of claim and evidence are 46.6% and 26.4%. Although the number of context nodes was only 30, the numbers of risks corresponded to context nodes are 146.

4 Discussion

4.1 Effectiveness of argument patterns

As the examinee said, the architecture decomposition pattern was useful to analyze risk, although the decision to choose it from argument decomposition patterns needed time to understand appropriateness between the target system and argument patterns.

Pitfalls discussed in the paper [13] were not observed in the course of the experiment. This also showed the effectiveness of the argument patterns based on DFD. Without the knowledge of argument patterns, the examinee could not develop a large assurance case consists of 549 nodes in less than two weeks. Very little time, only 3%, was spent for learning D-Case, because the examinee studied the D-case development method from DFD very well. This showed the appropriateness of the proposed D-Case development method.

4.2 Limitation

The experimental evaluation treated only one application. It is necessary to show the effectiveness of the method by evaluating more number of applications. Patterns used in the experiment were also limited, although these were effectively applied to develop assurance cases. The developed assurance case is assuring the dependability of the target system as a product. The dependability of the development process of the target system is also necessary as mentioned in ISO 26262.

In addition, other argument patterns are needed to evaluate. For example, there are several number of argument patterns described in [6]. The applicability of these patterns for the EAMS can be investigated.

5 Conclusion

This paper introduced assurance case patterns for dealing with DFD. Evaluation of the pattern approach was also evaluated for assuring an employee attendance management system using smartcards. The experimental evaluation showed the effectiveness of the DFD based patterns of argument decomposition. The examinees developed assurance case contains more than 500 nodes systematically in two weeks, after learned assurance case introduction course and patterns in 4 hours.

Future work includes more experimental evaluation of the proposed approach, comparative analysis of different argument patterns, and consistency management

between assurance case and DFD. The author plans to develop and evaluate argument patterns for operation phase [15] in the future.

Acknowledgment

This research was partially supported by JSPS Research Project No. 24220001 and the DEOS (Dependable Operating Systems for Embedded Systems Aiming at Practical Applications) project [16].

References

1. Kelly, T.P., 1997, *A Six-Step Method for the Development of Goal Structures*, York Software Engineering
2. Kelly, T., 1998, *Arguing Safety, a Systematic Approach to Managing Safety Cases*, PhD thesis, Department of Computer Science, University of York
3. Jackson, D. et al, 2008, *Software for dependable systems–sufficient evidence?*, National Research Council
4. European Organisation for the Safety of Air Navigation, October 2006, *Safety Case Development Manual*, 2nd Edition, EUROCONTROL
5. Despotou, G., Kelly, T., 2007, *Design and Development of Dependability Case Architecture during System Development*, in proceedings of the 25th International System Safety Conference (ISSC), Baltimore, MD, USA. Proceedings by the System Safety Society
6. Bloomfield, R. and Bishop, P., February 2010, *Safety and assurance cases: Past, present and possible future–an Adelard perspective*, in proceedings of 18th Safety-Critical Systems Symposium
7. Goodenough, J., Lipson, H., and Weinstock, C., Arguing Security - Creating Security Assurance Cases, https://buildsecurityin.us-cert.gov/bsi/articles/knowledge/assurance/643-BSI.html(2007)
8. Alexander, T., Hawkins, R., and Kelly, T., Security Assurance Cases: Motivation and the State of the Art, CESG/TR/2011 (2011)
9. Kaneko, T., Yamamoto, S., Tanaka, H., Proposal on Countermeasure Decision Method Using Assurance Case And Common Criteria, ProMAC 2012(2012)
10. Kaneko, T., Yamamoto, S., Tanaka, H., SARM -- a spiral review method for security requirements based on Actor Relationship Matrix ,ProMAC2010, P1227-1238 (2010)
11. Kaneko, T., Yamamoto, S., Tanaka, H., Specification of Whole Steps for the Security Requirements Analysis Method (SARM)- From Requirement Analysis to Countermeasure Decision - ,ProMAC2011(2011)
12. Yamamoto, S., Kaneko, T., Tanaka, H., A Proposal on Security Case based on Common Criteria, ASIA ARES2013, K. Mustofa et al. (Eds.): ICT-EurAsia 2013, LNCS 7804, pp. 331−336
13. Yamamoto, S., Matsuno, Y., ment Patterns to Reduce Pitfalls of Applying Assurance Case, pp.12-17,Assure 2013
14. D-Case editor, http://www.dependable-os.net/tech/D-CaseEditor/
15. Shota Takama, Vaise Patu, Yutaka Matsuno, Shuichiro Yamamoto: A Proposal on a Method for Reviewing Operation Manuals of Supercomputer. ISSRE Workshops 2012: 305-306
16. DEOS, http://www.jst.go.jp/kisoken/crest/en/research_area/ongoing/area04-4.html