International Journal of Smart Computing and Artificial Intelligence
International Institute of Applied Informatics
2019, Vol. 3, No. 1, 35 – 54

Proposal of Application Method of O-DA Template

Hiroyuki Utsunomiya [*], Nobuhide Kobayashi [†],
Kaori Hayashi [†], Shuji Morisaki [‡], Shuichiro Yamamoto [‡]

Abstract

O-DA (Open Dependability through Assuredness) is a dependable Enterprise Architecture frame-work for assuring open systems which evolve continuously after launching. O-DA is standardized by The Open Group. Although an O-DA template was proposed to support O-DA application, it has not been introduced for industry sectors. This paper proposes an application method of O-DA template. It also evaluates the effectiveness of the O-DA template for an actual embedded software design project and social game software Implementation project. The result shows 90% or more of the O-DA tem-plate can be reused for the application.

Keywords: O-DA, TOGAF, ArchiMate, Assurance case, Industrial case study

1 Introduction

Future systems usually might be going to cooperate with various adaptive devices such as IoT (In-ternet of Things) devices. As a result, it will be difficult to statically define the boundary of the as-suring target system. Those systems are called Open System.

In recent years, cars have also come to communicate with the outside, and the system has diversi-fied. It is difficult to establish a boundary that guarantees the embedded system, and the embedded system is also an Open System. Because of this situation, it is becoming very difficult to validate software design of embedded systems.

This paper proposes an application method of the O-DA template to build a software design vali-dation service in an embedded software development company. Then the method is applied to eval-uate the applicability of the O-DA template.

[*] Graduate School of Informatics Nagoya University, Aichi, Japan
[†] DENSO CREATE INC., Aichi, Japan
[‡] Graduate School of Informatics Nagoya University, Aichi, Japan

2 Terminology

Terminology used in this paper is shown in the Table 1.

Table 1: Terminology

TERM	DEFINITION
AA	Application Architecture
AADM	Assured Architecture Development Method
ABACE	Architecture Based Assurance Case Engineering
ADM	Architecture Development Method
AP	Application
ArchiMate	The ArchiMate is a modeling language of graphic representation similar to UML. The ArchiMate is designed to represent business architecture, application architecture, technology architecture.
Assurance case	A document for discussing the safety of the system on the basis of the test results and verifying results as evidence and guaranteeing or confidence to the system certifier and users etc.
AV	Architecture Vision
BA	Business Architecture
BG	Background
DEOS process	The DEOS process is an integrated iterative process containing the change accommodation cycle and the failure response cycle.
DIO	Domain Independent Ontology
DSO	Domain Specific Ontology
FABACE	Formal ABACE
feature model	In software development, a feature model is a compact representation of all the products of the Software Product Line in terms of "features".
Formal method	A method to describe system requirements, designs, etc. Using a mathematically strictly meaningful language, and to provide a mechanism for logically inferring whether the system meets user requirements or the like.
GSN	Goal Structuring Notation
O-DA	Open Dependability through Assuredness
O-DM	Open Dependency Modeling
OODA	Observe, Orient, Decide and Act
SPRME	Subject, Property, Risk, Measure and Evidence
TA	Technology Architecture
target use case	A use case to be analyzed.
TOGAF	The Open Group Architecture Framework
TrA	Transition Architecture
variability point	Indicate each feature between different systems.
variable element	The values that each system can take on features common to different systems.
viewpoint model	A viewpoint is a model that expresses an enterprise architecture, each handling a particular interest in a particular type of stakeholder.

3 Knowledge configuration of O-DA Framework

The Open Group standardized the O-DA (Open Dependability through Assuredness) [1] as the framework for assuring Open System dependability [2]. The O-DA standard is based on TOGAF (The Open Group Architecture Framework) [3] and it outlines the set of valuable knowledge for mitigating risk associated with dependability of complex interoperable systems based on assurance cases. The assurance case is used to assure the target of evaluation based on claims, strategies, context, and evidences. The goal structuring notation (GSN) is used to describe assurance cases [7][8][9][10]. Figure 1 outlines knowledge configuration of O-DA framework. The O-DA framework is decomposed by AADM (Assured Architecture Development Method) corresponds to Architecture Development Method (ADM) of TOGAF. ADM consists of the following phases.

- Phase P: Preliminary activities are achieved to develop enterprise architecture.
- Phase A: Architecture vision is defined.

- Phase B: Business architecture is developed.
- Phase C: Information system architecture is developed.
- Phase D: Technology architecture is developed.
- Phase E: Opportunity and solutions are clarified to realize the enterprise architecture to integrate business, information, and technology architecture.
- Phase F: Transition architecture is defined to achieve the target architecture from the baseline architecture.
- Phase G: Implementation and governance activities are achieved for the target enterprise architecture.
- Phase H: Change management of the realized target enterprise architecture is controlled.

In case of AADM, assurance cases are used to build consensus among stakeholders to ensure dependability of the target enterprise architecture in the course of ADM phases. The enterprise architecture can be assured of developing assurance cases in all the phases of AADM. The O-DA application knowledge provides Architecture based assurance case engineering (ABACE) [11][12], FABACE (Formal ABACE), Assurance case review method [15], SPRME (Subject, Property, Risk, Measure, Evidence) method, Assurance case capability index, and O-DA template [13]. FABACE provides a method to develop evidence by formal methods such as B [20], and Event-B [21][22]. The O-DA application knowledge utilizes elementary knowledge, such as, TOGAF, ADM, ArchiMate, Assurance case, and Formal methods.

ABACE provides a systematic method to develop assurance cases based on enterprise architecture. FABACE provides a method to develop evidence of assurance cases by using formal methods. SPRME method also provides a systematic method to develop assurance cases based on the definition of the subject architecture, dependable property, risks to thereat the property, measures to mitigate the risk, and the evidence for the measures. As the way of introducing an O-DA to the actual development project, O-DA template was proposed to show the typical use case for the quality evaluation service by describing AADM phases in detail.

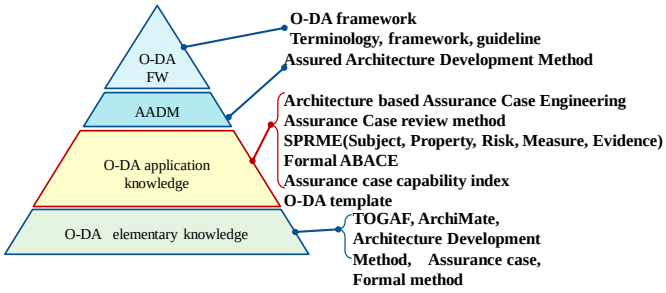

Figure 1: Knowledge configuration of O-DA Framework

4 Related work

The Open Group Real Time & Embedded Systems Forum focuses on standards for high assurance, secure dependable and complete systems [1]. At the heart of this O-DA (Open Dependability through Assuredness) standard, there is the concept of modeling dependencies, building assurance cases, and achieving agreement on accountability in the event of actual or potential failures. Assurance cases are necessary to assure architectures of dependable systems [1][2]. Assurance cases are used to show the validity of claims by evidence. GSN (Goal Structuring Notation) was also used to describe assurance cases [7][8][9][10]. The DEOS process was proposed to manage dependability of complex systems by using dependability cases [1][2]. The dependability case is an assurance case for assuring dependability. The DEOS process [2] is an integrated iterative process containing the change accommodation cycle and the failure response cycle.

O-DA will benefit organizations relying on complex systems to avoid or mitigate the impact of failure of those systems. O-DA includes the DEOS process mentioned before. The Change Accommodation Cycle and the Failure Response Cycle that together provide a framework for these critical processes. O-DA brings together and builds on The Open Group vision of Boundaryless Information Flow. These concepts include O-DM (Open Dependency Modeling) and Risk Taxonomy of The Open Group Security Forum, and Architecture models of The Open Group ArchiMate® Forum [4]. ArchiMate can be to describe enterprise architecture models [5][6]. Approaches to assure architecture were proposed by using ArchiMate [11][12]. The O-DA template [13] has been proposed to clearly define the relationship between O-DA and ArchiMate concepts.

Perroud and Inversini proposed the Enterprise Architecture Patterns, EAP, as practical solutions for IT-Architecture problems [14]. Although EAP showed 3 businesses, 5 supports, and 5 infrastructure patterns, no pattern to integrate all the architecture layers was considered. The O-DA template can be considered as the pattern of EA Patterns, because it contains all EA artifacts through ADM processes.

Variable elements of the O-DA template are necessary to manage to reuse. Feature model provides a useful structural constructs for managing variable and instance elements [23][24] [27]. Feature model is also compatible to an ontology which is useful for systematization of terms (contains in-stance elements) [25] [26].

Nwokeji et.al. [28] proposed a table to describe key concepts based on the meta model of EA change drivers. Kattenstroth [29] assessed TOGAF, ArchiMate and MEMO from the requirements for managing EA evolution. Antunes et.al. [30] proposed a hierarchical EA model analysis method based on ontology. The hierarchy consists of Domain Independent Ontology (DIO) and Domain Specific Ontology (DSO). ArchiMate is used to describe DIO. Durham et.al. [31] proposed an EA ontology learning process based on OODA (Observe, Orient, Decide, Act) loop [32].

5 O-DA Template

This Chapter explains the O-DA template proposed in [13]. The O-DA template is described compliant with TOGAF, and it is expected to apply to various case studies. The contents of O-DA template are background, architecture vision, business architecture, application architecture, technology architecture, and transition architecture. The O-DA template is defined by 1514 terms in total.

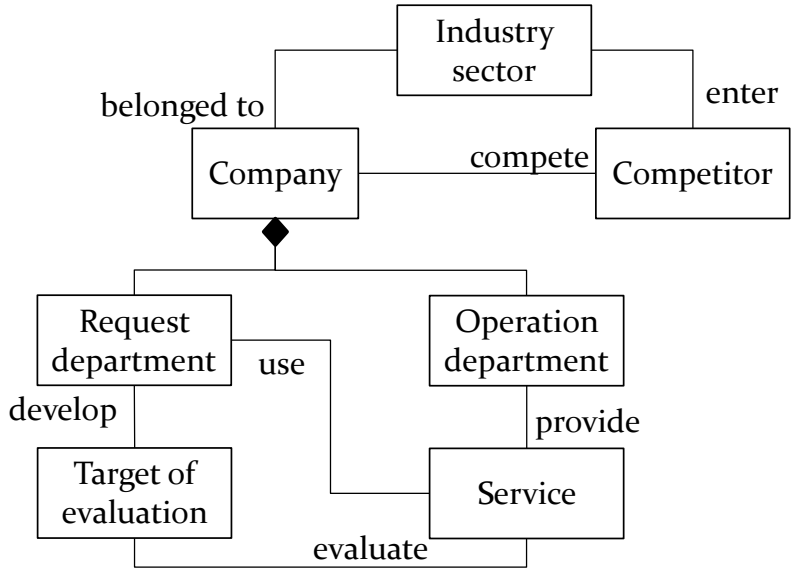

Figure 2: Meta model of generic terms in O-DA template

There are 7 generic terms to represent the concept of the business case described in O-DA template. The generic terms are “Company,” “Request department,” “Service operation department,” “Industry sector,” “Competitor,” “Service,” and “Target of Evaluation.” The meta model of the generic terms in O-DA template is shown in Figure 2. In this meta model, a company belongs to an industry sector. Some competitor enters into the industrial sector from other industry. A request department of the company develops a target of evaluation. An operation department provides a service to evaluate the target of evaluation.

The O-DA template consists of six sections in a format conforming to TOGAF [3]. The following explains each section of the O-DA template.

5.1 Background (BG)

The BG section shows the necessary elements to explain the background of introducing a service.

- Organization information (Organization name, Industry sector)
- External threats for organization
- Internal weakness of organization
- The function of the organization realized by introducing a service.

5.2 Architecture Vision (AV)

The AV section describes the necessary elements of the target architecture.

- The effect of introducing the service
- Comparison between before and after the service introduction
- Actors necessary to introducing the service
- Service implementation policy and execution procedures

5.3 Business Architecture (BA)

BA describes the baseline and target business architecture. It also analyzes the differences between baseline and target enterprise architectures. The architecture quality assurance process is defined based on assurance cases.

- Service process and their relationship are defined
- Trigger initiates the service
- Actors and roles are identified for the service

5.4 Application Architecture (AA)

The baseline and target application architecture are defined. The difference between baseline and target application architecture is also analyzed. The functions and information are clarified for the application architecture risk measures based on assurance cases.

- Constituents of application architectures
- Input and output information and their relationship of each application element
- Functions for application elements
- Relationship between application elements and actors

5.5 Technology Architecture (TA)

The baseline and target technology architecture are defined. The difference between baseline and target technology architecture is also analyzed. The information infrastructure environment is clarified for the technology architecture risk measures based on assurance cases.

- Devices, software and their relationship to realize the technology architecture
- Roles for each constituent

5.6 Transition Architecture (TrA)

The execution plan and transition architecture are defined based on the architecture road map. The architecture road map includes phases that define transitional architectures to realize the target architecture. The cost effective analysis and risk analysis are executed to refine the transition architec-ture execution plan.

- Architecture road map to realize the target architecture from baseline architecture
- Phases constitute the road map
- Artifacts of BA, AA, and TA for each phase
- The return on investment and risk for each phase

6 Application Method of O-DA Template

The application method of the O-DA template is developed to introduce the template. The constituents of the O-DA template application method includes the application steps to adopt the template for the specific target enterprise environment and the feature model manages variable terms. As the selection of terms may be cumbersome to introduce the template, we introduced the feature model to help engineers select candidate terms in variability points of the template.

6.1 O-DA template application steps

The application method of the O-DA template includes the following steps. In order to be able to deal with every sector, expandability is given in Step 6.

1. Identify the target use case for the O-DA template application.
2. Extract variable element from O-DA template (See Chapter 5) and define those relations as the introductory viewpoint model shown in Figure 3.
3. Comparative Meta-model of O-DA template and the target use case.
4. Replace terms in O-DA template with the corresponding terms in the target use case for where the Meta - model is common.
5. Replace terms after rebuilding the structure of the O-DA template for terms that are not common to those of the Meta-model.
6. Add sentences if necessary.
7. Manage candidate terms efficiently by using the feature model for replacing variable elements in step 4 and step 5. Chapter 6.2 explains the usage of the feature model.

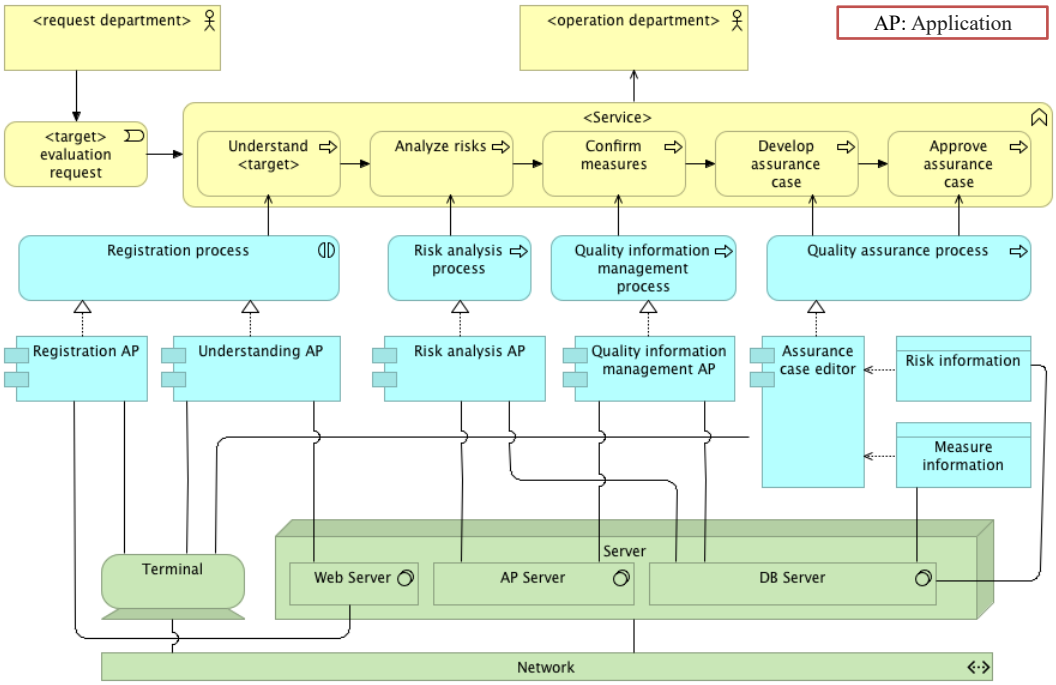

Figure 3: Introductory viewpoint model for BA, AA and TA in ArchiMate

6.2 Variable element management in O-DA template

Variable element management can support efficient application of O-DA template for various departments. It stores terms corresponds to variable element using a feature model. Figure 4 shows the feature model for O-DA template. The nodes under each alternative relation are concrete terms when an O-DA template is applied to the target service shown in Chapter 7. Other nodes are defined based on the Meta model shown in Figure 2. The following shows the usage of this feature model.

1. Identify a lower mandatory element in the feature model corresponds to a replace term of an actual use case.
2. Select an appropriate term from sub elements of an identified mandatory element.
3. Replace a term in O-DA template with a selected term.
4. Add a term of an actual use case as a reusable element under an alternative relation if an appropriate term is not found in step 2.

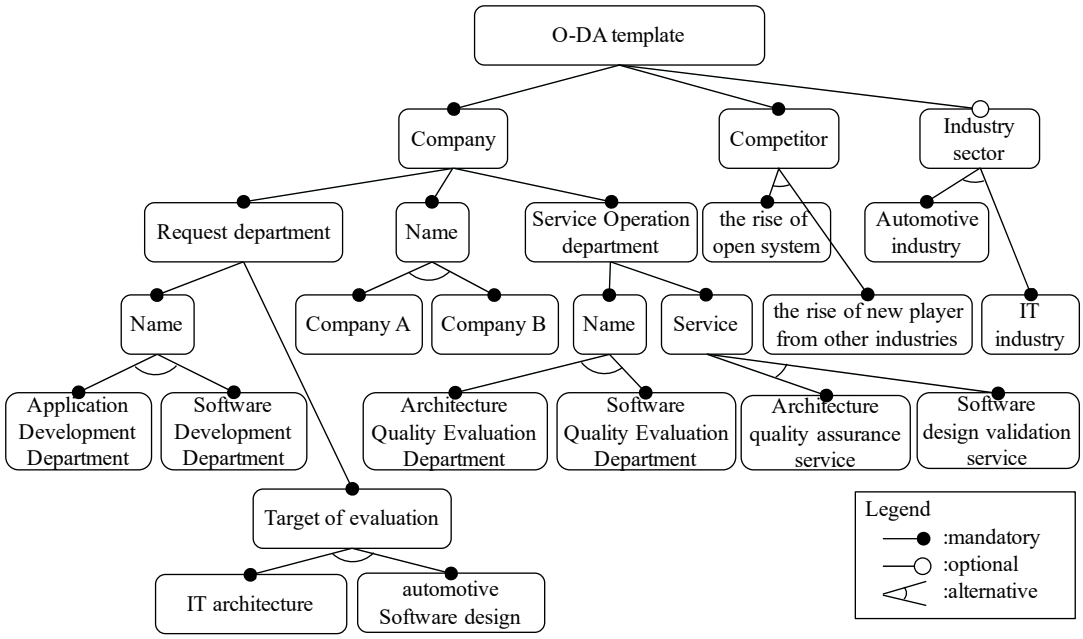

Figure 4: Feature model for O-DA template

7. Case Study

7.1 Software design validation service in Automotive sector

In this case, create the developer case examples of software design verification service that solves the problems of the organization like Table 2. O-DA approach is introduced to the software design validation service by using the O-DA template. Table 2 describes the target software design validation service. Industry sector of target service is Automotive. External threat is that rapid increase of new business players from external industry into the automotive sector. Internal issue is that organizational total productivity is low, because quality of projects is assured individually. Assurance target is software design quality of each project. Actors are Software development division as the requester, and Software design quality evaluation division as a service provider.

Table 2: Description of Automotive software

Category	Explanation
Industry	Automotive sector
An external threat	Rapid increase that new business players come into the automotive sector from external industry
Internal issues	As quality is assured for individual projects, organizational total productivity is low
Assurance target	Software design quality
Actors	-Software development division as requester -Software design quality evaluation division as service provider

Figure 5 shows the background goal model in ArchiMate for the target software design assurance service. Figure 5 shows the task of the company's operations department that must evaluate the quality of a system. The tasks are the following three.

- Evaluation knowledge is not shared in a department
- Affiliated industry faces competition with competitors
- Need to train staffs who have little experience

The following activities are necessary to solve the tasks.

- Maximize reuse of existing assets
- Keeping cost competitiveness
- Introducing evaluation service using IT

In addition, the evaluation knowledge within the department needs to be shared by processes and tools.

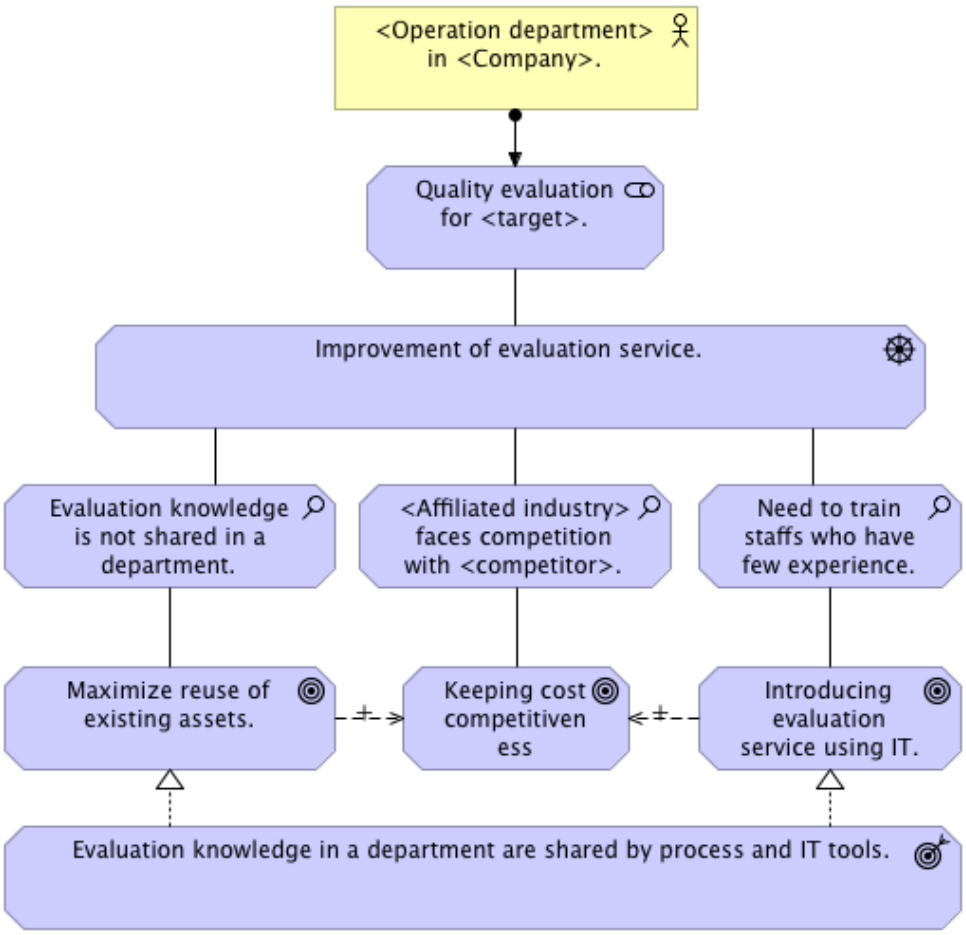

Figure 5: ArchiMate model for the background of the software design validation service

And Figure 6 shows the example of Transition architecture of system in Figure 3. Figure 6 defines the implementation order of elements through three phases about the introductory viewpoint model shown in Figure 3. It shows where the elements are aligned step by step, this is the Transition Architecture.

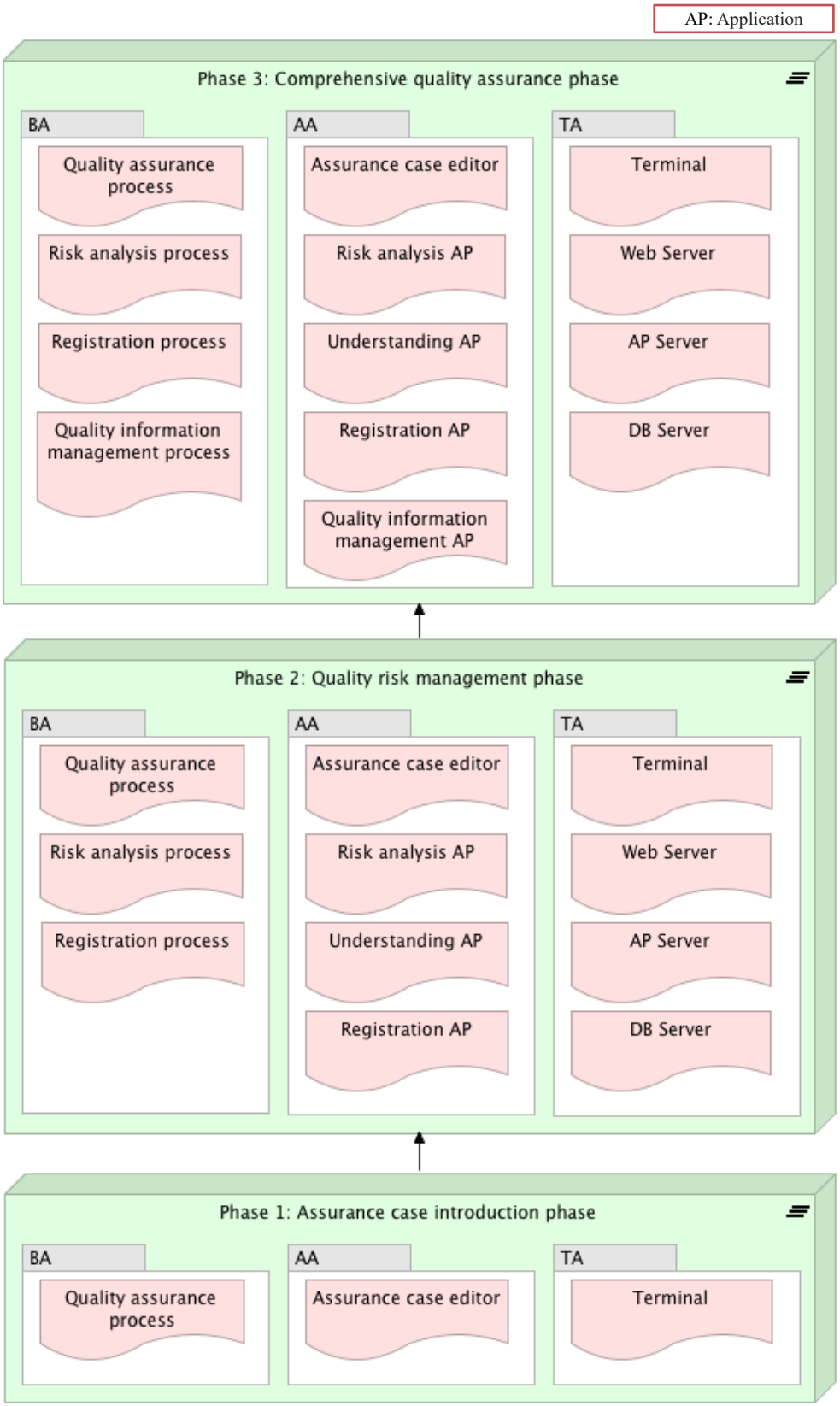

Figure 6: Transition Architecture of Automotive software

Result of the first case

The O-DA template was used to describe the software design validation service mentioned above according to the method proposed in the Chapter 6. Table 3 shows the variable points between the O-DA template and the software design validation service and the respective variable elements. These numbers are the number of replaced words, or the number of added words, determined by Step 4-6 of Section 6.1. The category of variability points of the case are 7 generic terms mentioned Figure 2.

By replacing terms only related to the seven variability points, the O-DA artifacts for the software design validation service have been developed efficiently. The difference of the O-DA template and the developed O-DA artifact is only 97 terms. This showed that 93.6% ((1514-97) /1514) of the O-DA template terms was reused to develop O-DA artifacts of the software design validation service. This shows the reuse ratio of the O-DA template was 93.6%. Moreover, in the course of replacement, sentence structures of the O-DA template were not necessary to change.

Table 3: Number of replaced terms

Variables	O-DA template terms	Software design validation service terms	Number of replaced terms in each phase						Total
			BG	AV	BA	AA	TA	TrA	
Company	Company A	Automotive Company	3						3
Request department	Application Development Department	Software Development Department		5	1	3			9
Service operation department	Architecture Quality Evaluation Department	Software Quality Evaluation Department		8	3	1			12
Industry sector	IT industry	Automotive industry	1						1
Competitor	the rise of open system	the rise of new players from other industries	1						1
Service	Architecture quality assurance service	Software design validation service	1	1	2		1		5
	AQAS	SDVS		1					1
Target of Evaluation	IT architecture	automotive software design	1	1					2
	architecture	software design	4	11	6	18	7	13	59
	IT systems	automotive systems	2						2
	architecture components	software design components	1						1
	system architecture design	software design	1						1
Total number of occurrences			15	27	12	22	8	13	97

Table 4 shows the application workload of the O-DA template for the software design validation service. The following seven steps are the steps described in section 6.1.

Table 4: Work Load

Steps	Contents	Hours
1	The target use case is defined outside of the O-DA template	0.0
2	Define Introductory model in ArchiMate for the software design validation service	4.0
3	Compare O-DA template with software design validation service by using introductory model	0.5
4	Replace terms of O-DA template into those of the corresponding software design validation service	2.0
5	Revise the structure of template sentences for the un-corresponding parts and then replace the Software design validation terms	0.0
6	Add necessary sentences	0.0
7	Revision of feature model and manage of feature model	0.0

The application workload was only 6.5 hours in total. As an important point that could shorten the time it was step 4, and 93.6% of the terms could be reused. It can be thought that it took several days if the developer could not reuse the term and implemented a new software design. However, the developer had already taken the training of TOGAF and ArchiMate and also had knowledge of the feature model.

The step1 was executed outside of the template application. In step 2, introductory model of the Software design validation service was developed in ArchiMate. In step 3, introductory models of O-DA template and Software design validation service were compared to detect the difference. In step 4, the different terms detected in step 3 were replaced with O-DA template and the resulted artifacts were used as O-DA artifacts for the Software design validation service. The step 5-7 was not necessary for the Software design validation service, because the O-DA template was clearly aligned to the Software design validation service.

7.2 Software implementation validation service in social game sector

In the next case, the developer create case examples of software implementation verification service that solves the problems of the organization like Table 5. O-DA approach is introduced to the software implementation validation service by using the O-DA template. Table 5 describes the target software implementation validation service. Industry sector of target service is Social game. External threat is that rapid increase of new business players from external industry into the social game sector. Internal issue is that developers are short. Assurance target is software implementation quality of each project. Actors are Software development division as the requester, and Software implementation quality evaluation division as a tester.

In addition, social games are played by multiple people simultaneously using each device. Connection from multiple devices, and multiple device platforms must be supported.

Table 5: Description of Social game software

Category	Explanation
Industry	Social game sector
External threat	Rapid increase that new business players come into Social game sector from external industry
Internal issues	Developers are short
Assurance target	Software implementation quality
Actors	-Software development division as requester -Software implementation quality evaluation division as tester
Technology	-Multiple device simultaneous play -Multiple devices

In this case, terms are different, but the structure of the background goal model is equivalent to Figure 5. For introductory viewpoint model, we need to add one application to the configuration in Figure 3. It is an application for verifying multiple devices, multiple simultaneous access. In relation to this, in Step 6, add sentences that are required for BG and AA of the O-DA template.

Result of the second case

Figure 6 shows the variable points between the O-DA template and the software implementation validation service and the respective variable elements.

In this case, it was necessary to replace 97 terms and add sentences consisting of 52 terms. The O-DA template reuse rate was 90.5% ((1514-97) /(1514+52)). In addition, although there was a sentence addition, there was no need to change the structure of the O-DA template.

Table 6: Number of replaced terms

Variables	O-DA template terms	Software implementation validation service terms	Number of replaced terms in each phase						Total
			BG	AV	BA	AA	TA	TrA	
Company	Company A	Social game Company	3						3
Request department	Application Development Department	Social game Development Department		5	1	3			9
Service operation department	Architecture Quality Evaluation Department	Software Test Department		8	3	1			12
Industry sector	IT industry	Social game industry	1						1
Competitor	the rise of open system	the rise of new entry from other industries	1						1
Service	Architecture quality assurance service	Software implementation validation service	1	1	2		1		5
	AQAS	SIVS		1					1
Target of Evaluation	IT architecture	social game implementation	1	1					2
	architecture	implementation	4	11	6	18	7	13	59
	IT systems	social game systems	2						2
	architecture components	software implementation	1						1
	system architecture design	system implementation	1						1
Technology	None (Add sentences by Step 6)	-Multiple device simultaneous play -Multiple devices	24 (Sentences)			28 (Sentences)			52
Total number of occurrences			39	27	12	50	8	13	149

Table 7 shows the application workload of the O-DA template for the software implementation validation service. The following 1-7 steps are the steps described in section 6.1, step 0 has been added because acquisition of peripheral knowledge was necessary.

Table 7 Work Load

Steps	Contents	Hours
0	Understanding O-DA template (including reference time of TOGAF and ArchiMate)	3.0
1	The target use case is defined outside of the O-DA template	0.0
2	Define Introductory model in ArchiMate for the software implementation validation service	3.0
3	Compare O-DA template with software implementation validation service by using introductory model	0.5
4	Replace terms of O-DA template into those of the corresponding software implementation validation service	2.0
5	Revise the structure of template sentences for the un-corresponding parts and then replace the Software implementation validation terms	0.0
6	Add necessary sentences	0.5
7	Revision of feature model and manage of feature model	0.5

The application workload was 9.5 hours in total. Here again, as an important point that could shorten the time it became step 4, and 90.5% of terms could be reused. In this time, the developer who has no knowledge was in charge, so needed time to understand the O-DA template as Step 0.

The step1 was executed outside of the template application. In step 2, introductory model of the target service was developed in ArchiMate. In step 3, introductory models of O-DA template and target service were compared to detect the difference. In step 4, the different terms detected in step 3 were replaced with O-DA template and the resulted artifacts were used as O-DA artifacts for the target service. The step 5 was not necessary for the target service, because the O-DA template was clearly aligned to the target service. In step 6, the necessary sentences for verifying multiple devices and multiple simultaneous connections have been added. In step 7, the variable point extracted this time was fed back to the feature model.

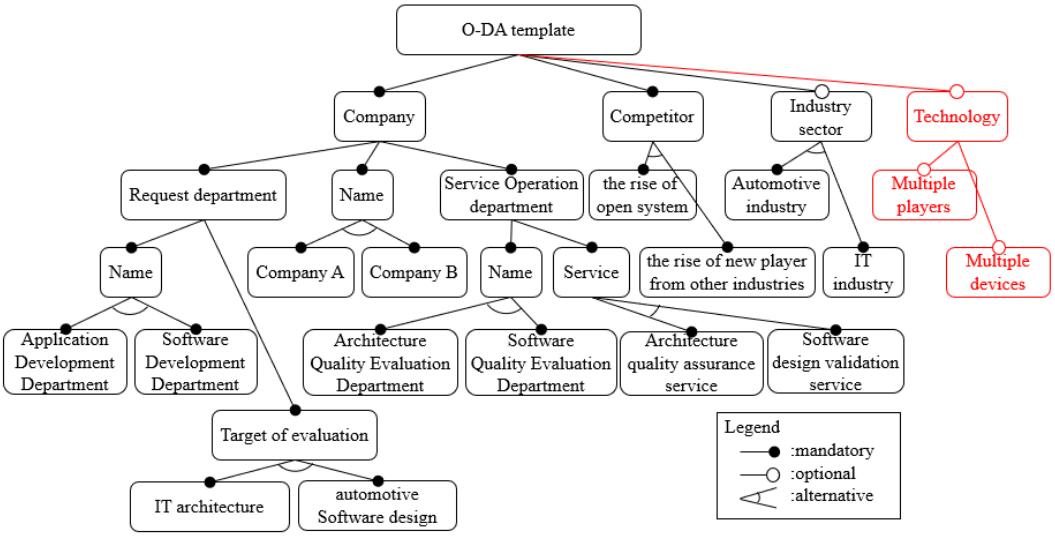

Figure 7: Feature model for social game sector

8 Discussion

8.1 Generality

In two cases, step 5 was not necessary for the application of the O-DA template. This showed the generality of the O-DA template for the quality assurance related service sectors. This is because the meta model of O-DA template is simple enough to describe the typical service sector of evaluation as shown in Figure 2. The same results were obtained even if applied to another evaluation system in the another Embedded sector.

8.2 Efficiency

The O-DA template is adopted within 10 hours for the target service. It seems that it took several days if developers did not use O-DA template and implemented new software design. This indicates that the proposed method (especially Step 4) is efficient. This shows the efficiency of the proposed steps in Chapter 6. Although the most time-consuming task was to develop an introductory model in step 2, this step will be eliminated subsequent O-DA template applications of the similar target services.

Except, when an operator who does not have the necessary knowledge, such as ArchiMate or a feature model is in charge, additional knowledge acquisition time is required. However, this time is unnecessary by education within the organization.

8.3 Novelty

This paper shows the following novelty.

This paper shows the efficiency and reusability of the O-DA template quantitatively for the first time, and it is new as it shows qualitatively the effect and generality for the first time. In Chapter 6, we introduced the feature model for clarifying and managing variable points of O-DA template for the first time. The feature model helps engineers select the variability within the O-DA template. Using the feature model improves the adoption process of O-DA templates.

Industry application of O-DA template has never been done, but it was expected to be applied to various case studies. This paper showed that the O-DA template can be practically applied to improve the evaluation process of automotive software development company and social game development company. This paper is the first to clarify the effectiveness of the O-DA template for the evaluation department of Japanese automobile companies and game companies.

8.4 Limitation

The case showed positive results as above, but the O-DA template is only applied to two similar services. In order to evaluate the effectiveness of the O-DA template, we need to prepare more different types of applications. Further cases by more developers without relating knowledge are also necessary.

9 Conclusion

In this paper, an application method of the O-DA template was proposed. The case to evaluate the effectiveness of the O-DA template has been executed by applying the proposed method for the automotive sector and social game sector. The result shows the reusability of the O-DA template was 93.6% and 90.5% for the application case. 90% or more of the template was reused without modification. In cases with two sectors, it was only the replacement of 97 terms at the most and the addition of sentences consisting of 52 terms.

For example, the names of the company, departments of the organization, service, target of assurance, and architectural artifacts of the O-DA template were changed to develop a target system development plan based on baseline architecture. Therefore, the workload of developing the planning document was within 10 hours. This showed the effectiveness and reusability of the O-DA application.

It has been demonstrated that it can cover the services of two embedded software verification departments. It is also empirically considered that other embedded software is probably established. In other words, We think that we verified that the software validation service could be logged by O-DA. This is based on the fact that it covers the verification service process and that the verification service process is standard.

Future work includes another application of O-DA template for improvement, knowledge integration with other safety critical knowledge [16][17], and consideration on quality assurance with weight [18][19]. Additionally, it is necessary to evaluate the effectiveness of the feature model managing variable elements in the O-DA template.

Acknowledgment

Authors thanks to members of The Open Group Real Time & Embedded Systems Forum for their continued support and helpful comments.

References

[1] The Open Group, “Dependability through Assuredness (O-DA) Framework,” November 2013.

[2] Mario Tokoro., “Open Systems Dependability,” CRC Press, May 2015.

[3] Andrew Josey., “TOGAF Version 9.1-A Pocket Guide,” Van Haren Publishing, 2011.

[4] Andrew Josey., “ArchiMate 2.1-A Pocket Guide,” Van Haren, 2013.

[5] Marc lankhorst et al., “Enterprise Architecture at Work -- Modeling Communication and Analysis, ” Third Edition, Springer, 2013.

[6] Gerben Wierda, “Mastering ArchiMate – A Serious Introduction to the ArchiMate Enterprise Architecture Modeling Language,” Edition II, The Netherlands Published by R&a, 2014.

[7] Tim Kelly, “A Six-Step Method for the Development of Goal Structures,” York Software Engineering, 1997.

[8] Tim Kelly, Jhon McDermid, "Safety Case Construction and Reuse using Patterns," University of York, 1997.

[9] Tim Kelly, "Arguing Safety, a Systematic Approach to Managing Safety Cases," PhD Thesis, Department of Computer Science, University of York, 1998.

[10] Tim Kelly and Robert Weaver, "The goal structuring notation–a safety argument notation," In Proceedings of the dependable systems and networks 2004 workshop on assurance cases, 2004.

[11] Shuichiro Yamamoto., "An approach to assure Dependability through ArchiMate," In International Conference on Computer Safety, Reliability, and Security, pp. 50–61. Springer, 2015.

[12] Shuichiro Yamamoto and Nobuhide Kobayashi., "Mobile Security Assurance through ArchiMate," In The 2016 International Symposium on Mobile Internet Security, October 2016.

[13] Shuichiro Yamamoto and Shuji Morisaki., "A case study on architecture quality assurance service using O-DA," In Conference on ENTERprise In- formation Systems 2016, September 2016.

[14] Thierry Perroud and Reto Inversini, "Enterprise Architecture Patterns—Practical Solutions for Recuring IT-Architecture Problems," Springer, 2013.

[15] Shuichiro Yamamoto, Shuji Morisaki, "A System Theoretic Assurance Case Review," ICCSE 2016, 11th International Conference on Computer Science & Education (ICCSE), pp. 992 - 996, DOI: 10.1109/ICCSE.2016.7581719

[16] Shuichiro Yamamoto., "A Knowledge Integration Approach of Safety-critical Software Development and Operation based on the Method Architecture," In Procedia - Procedia Computer Science, pp. 1718–1727. Elsevier Masson SAS, 2014.

[17] Shuichiro Yamamoto., "A Systematic Knowledge Education Approach for Safety-Critical System Development," Procedia - Procedia Computer Science, Vol. 60, pp. 960–967, 2015.

[18] Shuichiro Yamamoto, "An approach for evaluating softgoals using weight," ASIAARES 2015, pp. 203-212, 2015

[19] Nobuhide Kobayashi, Shuji Morisaki, Noritoshi Atsumi, and Shuichiro Yamamoto, "Quantitative Non Functional Requirements evaluation using softgoal weight," Journal of Internet Services and Information Security (JISIS), volume: 6, number: 1, pp37-46, 2016.

[20] Abrial, Jean-Raymond. The B-Book: Assigning Programs to Meanings. Cambridge University Press, 1996.

[21] Event-B, http://www.event-b.org/index.html. Home of Event-B and the Rodin Platform. 2008.

[22] Imed Abbassi, Mourad Kmimech, Nejib Ben Hadj-Alouane and Walid Gaaloul, "Modeling and Verifying the Transactional and QoS-aware Services Composition Using Event-B," IEEE 23rd International WETICE Conference, pp. 313 – 318, 2014

[23] Kang, Kyo C., et al. "Feature-oriented domain analysis (FODA) feasibility study." No. CMU/SEI-90-TR-21. Carnegie-Mellon Univ Pittsburgh Pa Software Engineering Inst, 1990.

[24] Systems and Software Variability Management. Concepts, Tools and Experiences. Bearbeitet von. Rafael Capilla, Jan Bosch, Kyo-Chul Kang. 1. Auflage 2013. Buch. XIV, 317 S. Gebunden.

[25] W3C. OWL web ontology language. Document available at http://www.w3.org/TR/owl-features/.

[26] Czarnecki, Krzysztof, et al. "Feature models are views on ontologies." Software Product Line Conference, 2006 10th International. IEEE, 2006.

[27] Systems and Software Variability Management. Concepts, Tools and Experiences. Bearbeitet von. Rafael Capilla, Jan Bosch, Kyo-Chul Kang. 1. Auflage 2013. Buch. XIV, 317 S. Gebunden.

[28] Nwokeji, J. C., Clark, T., Barn, B., Kulkarni, V., A conceptual framework for enterprise agility, proc. of 30th Annual ACM Symposium on Applied Computing, pp. 1242-1244, 2015

[29] Kattenstroth, H., DSMLs for Enterprise Architecture Management-Review of Selected Approaches, DSM'12, pp.39-44, 2012

[30] Antunes, G., Bakhshandeh, M., Mayer, R., Ontology-Based Enterprise Architecture Model Analysis, SAC'14, pp.1420-1422, 2014

[31] Durham, J., McLauchlan, L., Yuster, R., Enabling A Common and Consistent Enterprise-Wide Terminology:An Initial Assessment of Available Tools, International Conference on Web Intelligence and Intelligent Agent Technology, pp. 544-548, 2008

[32] Schechtman, G., Capt. USAF, "Manipulating the OODA loop: the overlooked role of information resource management in information warfare," (Thesis), Logistics and Acquisition Management of the Air Force Institute of Technology Air University Air Education and Training Command, Dec. 1996.