What Method is to Achieve the Purposes of Systems-of-Systems Having Different Boundaries as System-of-Systems for Each Organization?

Nobuyuki KOBAYASHI\textsuperscript{a}, Mikiko NAKADA\textsuperscript{b}, Seiko SHIRASAKA\textsuperscript{b}

\textsuperscript{a}The System Design and Management Research Institute of Graduate School of System Design and Management, Keio University, 4-1-1 Hiyoshi, Kohoku-ku, Yokohama-shi, Kanagawa 223-8526 Japan
\textsuperscript{b}Graduate School of System Design and Management, Keio University, 4-1-1 Hiyoshi, Kohoku-ku, Yokohama-shi, Kanagawa 223-8526 Japan

Abstract

In an enterprise, designing system-of-systems (SoS) is required to assume and clarify each organization-specific viewpoint for the SoS, in order to solve problems within their budgets and benefit. The previous study suggests that designing the IT system as to achieve different purposes depending on the various positions is difficult. To solve the above problems includes two difficulties which must solve at least. One difficulty is that if the organizations do not have enough conversation among organizations, the organizations cannot reduce the range of what we do now know the ‘unknown unknowns’. Another difficulty is that to achieve two (or more) purposes at the same time. Therefore, the purpose of this study is to reduce misunderstanding at the time of SoS design in different positions that have each different purpose, in order simultaneously to achieve the purposes of SoSs which have different boundaries defining as SoS for each organization. We will propose a design method which assumes the purpose of this study, includes steps to make it that can commonize design by two positions. To evaluate the proposed method, we actually applied the proposed method to the processes for the order for sales and the maintenance for the product in the manufacturing industry.

Keywords: System-of-Systems, SoS Design, design method.

1. Introduction

In an enterprise, when designing an IT system, it is necessary to design to achieve different purposes for each organization, assuming each organization-specific viewpoint for the system-of-systems (SoS) \cite{1}. Moer \cite{1} describes that a SoS is an assemblage of components which individually may be regarded as systems. One of the reasons is that companies which do not consider or clarify managed objects (IT system) as either a monolithic system or a SoS are likely to face problems in their analyses of cost and benefit relationships \cite{2}. In other words, designing SoS is required to assume and clarify each organization-specific viewpoint for the SoS, in order to solve problems within their budgets and benefit. On the other hand, unfortunately, the previous study \cite{3} showed that there are multiple positions that grasp the managed object (IT system) differently which increases the complexity (misunderstanding probability) exponentially. Therefore, the previous study suggests that designing the IT system as to achieve different purposes depending on the various positions is difficult. To solve the above problems includes two difficulties which must solve at least. One difficulty is that if the organizations do not have enough conversation among organizations, the organizations cannot reduce the range of what we do now know the ‘unknown unknowns’ \cite{4}. Another difficulty is that to achieve two (or more) purposes at the same time \cite{5} \cite{6} \cite{7} \cite{8}. Therefore, the purpose of this study is to reduce misunderstanding at the time of SoS design in different positions that have each different purpose, in order simultaneously to achieve the purposes of SoSs which have different boundaries defining as SoS for each organization.

In order to evaluate to have achieved the purpose of this study, it deals with a relationship between business processes and IT systems for two positions (sales department and parts department) that have each different purpose. If two positions use one or more IT systems in common by two different positions, it is difficult to assume each other position-specific viewpoint for each SoS which is established a combination of the above common IT systems and each position’s specific IT systems. Thus, each position cannot understand the difference between how to use and boundaries of each SoS which is established a combination of the above common IT systems and each position’s specific IT systems. The reason why each position cannot understand the difference between how to use and boundaries of each SoS is that organizations have become complex socio-technical systems of heterogeneous systems (e.g., system of systems) that pursue their own—sometimes conflicting—goals, behaviors, functions, and requirements \cite{9}. The behavior executed by such local autonomous entities may, however, conflict with the system-wide goals \cite{9}. Despite these occasionally conflicting behaviors and goals, enterprises should have the ability to cope with emerging threats and to adapt to turbulent environments, while satisfying stakeholders’ needs \cite{9}. Therefore, we used the above situation which will lead the stakeholders (sales department and the parts department and systems engineers and designers) to misunderstand or not consider other departments in this

*Corresponding author: n-kobayashi5@a6.keio.jp

Received: April 12, 2021
Accepted: October 5, 2021
case, in designing the SoS for achieving two or more purposes simultaneously.

In the above situation, we propose a design method which assumes the purpose of this study, includes steps to make it that can commonize design by two positions. Specifically, the first step designs A system (a monolithic system) to be used in one position (sales department) (Called “system design in the first position”). Next, we design an “improved A system” that is handled in another one position (parts department), based on A system, using Systems Engineering (Called “system design in the second position”). At this time, when the function is added or deleted by the system design in the second position, the designer understands that the system design in the first position (sales department) is not optimal, the designer redesigns the system design in the first position (sales department) again [3]. The designer understands that by designing the system for the second position (parts department), the system design for the first position (sales department) was not optimal as a result of trying to achieve the two goals of the first and second positions at the same time. In addition, as a result of the functional difference being reflected, that is because there is a possibility that the system design in the first position (sales department) is broken. This might lead to cause an operation failure not assumed at the time of the system design in the first position by the added or removed function. This approach is a similar approach to category creation when comparing multiple sentences using open coding of qualitative data analysis [10] [11] [12] [13]. If we will qualitatively achieve mutual understanding related to managed objects, each time a change occurs them, it is necessary to check again until the function does not change from both the system design in the first position (sales department) is broken. That is because the systems designers have defined the range that does not affect the function used by the sales department within the imagination of the systems designers. Future program modifications include legal changes and marketing changes. As a result, it is necessary to consider eliminating the act of simply adding functions in order to avoid an increase in cost in the future. Therefore, the proposed method has usefulness for avoiding an increase in cost in the future. The evaluation method of this study is to confirm “agreement between different organizations, which have independent business processes, and completion of design” using “IT systems used by them for different purposes” in a company. In order to evaluate the proposed method, we are using the processes for the order for sales and the maintenance for the product in the manufacturing industry.

Next, the novelty of this study describes in detail. The previous study [2] proposed an assurance case description method to reduce misunderstanding caused by the difference of grasping the objects managed in various departments as a monolithic system or a SoS. The previous study [2] has shown a way to express the results after obtaining an agreement, but we have not clarified how to eliminate the misunderstanding specifically. Therefore, the previous study [2] is not research that focuses on how to design based on the premise of misunderstanding. The previous study [14] shows how use case diagrams, use case specifications, sequence diagrams, interface class diagrams and input output (I/O) entity class diagrams can be used to model the SoS, SoS constituent systems, and the associated capabilities at both the SoS and constituent system level. Cherfa et al. [15] proposed the creation of a strong link between the SoS analysis stage and the architecture stage in the SoS wave model [16] that is dedicated to the acknowledged SoS, in order to address misunderstanding among stakeholders in the two stages for one mission which is one purpose. Therefore, the previous study is not research that focuses on how to design based on the premise of misunderstanding among stakeholders who have each different purpose when they use business processes and IT systems. The previous study [17] introduces mKAOS which a way of defining what are the objectives of the SoS, as follows. mKAOS which is a Domain Specific Language (DSL) for mission modeling in the context of SoS was developed to tackle the problem of
properly representing missions in this context \[18\]. mKAOS \[19\] extends KAOS \[20\], a widely known language for Goal-oriented modeling, and it provides an associated design tool that allows the graphical modeling of the so-called Mission Models \[19\]. The other previous study \[21\] proposes a process to define, model and validate missions of SoSs in reference architectures using mKAOS. This process explains how to identify missions, designate responsibilities, allocate operational and communicational capabilities to abstract constituent systems, identify exchanged data entities, and determine emergent behaviors. Another previous study \[22\] proposes a method to verify SoS models based on mission-related properties, using the mission modeling language mKAOS and the DynBLTL formalism. Therefore, the above previous studies use mKAOS describing a SoS after deciding systems as SoS, do not focus on how to design based on the premise of misunderstanding among stakeholders who have each different purpose when they use business processes and IT systems. In category creation for comparing multiple objects using open coding of qualitative data analysis \[10\] \[11\] \[12\] \[13\], the analyst analyzes until categories do not increase after defining a certain viewpoint for creating categories. The action of the analyst is the recognition alignment of evaluations for multiple objects is performed by one person in order to compare multiple objects. A method to ensure the validity and the reliability of analysis is to confirm the above result with a third party who are experts or experimenters. However, open coding handled in previous study is a method used to evaluate one purpose and does not assume that two or more different purposes are evaluated. In other words, open coding is not a task of matching recognition for objects with different purposes such as SoS. From the above, the novelty of this study are two points. One is how to design objects with two or more different purposes on the premise of misunderstanding. Another one is to have clarified how the range that organizations can actually control and the range that organizations are responsible for management different positions are different depending on whether they are SoS of partially overlapping relationship or inclusive relationship. In other words, this study will contribute to finding and reducing unmatched between the range that organizations can actually control and the range that organizations are responsible for management.

We describe the remaining structure of this paper as below. Chapter 2 shows the concept of previous studies used in the design method. Chapter 3 describes the proposed method. Chapter 4 presents the evaluation method. Chapter 5 shows the evaluation results. Chapter 6 describes the discussion. Chapter 7 presents conclusions and future research topics.

2. Previous Studies

Five principal features of a “system of systems” \[23\] \[24\] is as follows:

- Operational Independence. Any system that is part of an SoS is independent and is able to operate serviceably if the SoS is disassembled.
- Managerial Independence. Despite collaborating with the other members of the SoS, the individual systems are self-governing and individually managed so that they “not only can operate independently, they do operate independently.”
- Geographic Distribution. The parties collaborating in an SoS are distributed over a large geographic extent. Although the geographic extent is defined vaguely, it is stressed that the collaborating systems can only exchange information and not considerable quantities of mass or energy.
- Evolutionary Development. An SoS’s existence and development are evolutionary in the sense that objectives and functionality can be under constant change, as they can be added, modified, or removed with experience. Thus, an SoS never appears completely formed.
- Emergent Behaviour. Through the collaboration between the systems in an SoS a synergism is reached in which the system behaviour fulfils a purpose that cannot be achieved by, or attributed to, any of the individual systems.

The previous study \[2\] states for reducing the misunderstandings we therefore first need to set the positions relevant to the misunderstanding, and then set the managed object either as a monolithic system or a SoS. Steps needed to reduce the misunderstanding are as follows:

Step 1: Set position(s) relevant to the misunderstanding.
Step 2: Set the managed object(s) as a monolithic system or a SoS.
Step 3: Describe the managed object(s) as a monolithic system or as a SoS for each position set in Step 1.
Step 4: Check if there are any gaps relevant to the position(s) set in steps 1.
Step 5: Eliminate all the gaps among position(s) set in Step 1.

With the above steps, the positions can eliminate the difference to view the managed object(s).

The previous study \[3\] used information system design as an example, and when changes were discovered in the application layer (a lower abstract concept), the procedure for returning to the business process layer (a highly abstract concept) and making corrections was shown in Figure 1.

In other words, when there are two parties with different purposes as in this study, it is necessary to confirm that the changes that occurred in the low abstract concept with each other, so that the highly abstract concept may be achieved.

Next, open coding \[12\] for performing two comparisons is a method that performs as follows:

Step 1: View the free answers for CASE 1, set the viewpoint for Affinity Diagram grouping (Step 2). It was set in the study as “usefulness of CASE 1,” to show for what purpose CASE 1 is useful.
Step 2: Look for, from the aforementioned viewpoint, the descriptions for CASE 1 that seem to be related, and sort them into groups.
Figure 1 The procedure of change from state 1 to state 3 [3].

Step 3: Write titles for each group that summarizes the essence of the group, at a slightly higher level of abstraction (called "Open coding results of CASE 1" in the study).

Step 4: Compare with "Open coding results of CASE 1" the free answers for CASE 2.

Step 5: For CASE 2, look for, from the aforementioned viewpoint, the descriptions that seem to be related, and sort them into groups under the same title as CASE 1, or new groups for different descriptions (called "Open coding results of CASE 2" in the study).

Step 6: Write titles for the groups newly made for CASE 2.

Step 7: Make a table of "Open coding results of CASE 1" and "Open coding results of CASE 2" to highlight the differences. The study ensured the validity of the analysis by having one researcher specializing in qualitative research methods and another specializing in assurance cases review the analysis results. The above is open coding for performing two comparisons.

3. A design method to achieve two purposes within the SoS simultaneously

To achieve two purposes within the SoS simultaneously, it is necessary to consider two system designs for each purpose. The function range that the system has, may differ between the system design in the first position (A) and the system design in the second position (B). In other words, if the physical range is affected, the designers also consider the physical range. It is necessary to consider and define ranges for each hierarchy (functional layer, physical layer). In Figure 2, the designers divide according to the three relationships: Inclusive relationship, partly overlapping relationship and non-overlapping relationship.

Pattern 1 is two systems with an inclusive relationship. (This is SoS.)

Pattern 2 is two systems with a partially overlapping relationship. (This is SoS.)

Pattern 3 is two systems with a non-overlapping relationship. (This is two monolithic systems.)

If there is no overlap in the object (pattern 3), it will be excluded from this study.

This study proposes a design method for two systems, which are the minimum units of SoS, as below. In this study, the system design in the first position is called A system design, and the system design in the second position is called B system design.

Step 1): Design the A system from the first position.

Step 2): Design the B system from the second position on the premise of the A system design of step 1). (Designers determine the range that can commonize design in both A system range and B system range and the range that cannot commonize design in both A system range and B system range.)

Step 3): Design the A system from the first position on the premise of the B system design of step 2).

Step 4): Design the B system from the second position on the premise of the A system design of step 3).

Designers repeat above steps 3) to 4) until both the first position and the second position have confirmed the range that cannot commonize design in both A system range and B system range.

As shown in the above design method, even if the steps 1) and 2) are carried out from either step, if the design of 1) is the A design, the latter design will perform the B design on the premise of the A design. Furthermore, at the step of 3), the designers consider the second A design on the premise of the B design of 2). Further, in the step 4), the designers consider the second B design on the premise of the A design of 3). The reason is that the two systems overlap, making each other a constraint. The steps 3) and 4) are repeated, and the designers agree with the design upon when there are no additional functions. In this way, design premised on SoS is possible. According to Kobayashi et al. [3], this procedure is the same as the procedure making corrections revising the business process layer (a higher abstract concept) when changes are discovered in the application layer (a lower abstract concept) in information system design. The difference from the previous study [3] is that the two system designs start with the state that the two systems do not exist both.

4. Evaluation Method

The evaluation in this study is to confirm completing both business process design and IT system design for two organizations by agreeing between two organizations which have independent business processes, using IT system used for each purpose in two organizations. To evaluate the proposed method, we are using the processes for the order for sales and the maintenance for the product.
in the manufacturing industry. Two organizations for evaluating the proposed method are sales department and parts department. Therefore, by checking the sales department and parts department using the proposed method, we evaluate the usefulness of confirming that the design changes of two departments do not affect the functions of two departments (the range that cannot be commonized). Concretely, this study used Business Process Modeling Notation (BPMN) as a visualization method of system design to evaluate the proposed method. Figure 3 shows the relationship among procedures of the proposed design method and the evaluation procedures of the proposed design method. Based on the procedures in Figure 3, this study was performed in the order of 1)-1, 1)-2, 2)-1, 2)-2, 3)-1, 3)-2, 4)-1 and 4)-2. This study evaluated by 13 participants from 7 organizations in Table 1.

The business processes used for the evaluation are both an accepting order process for sales and a maintenance process for the product in the manufacturing industry. The accepting order process and the maintenance process use Web Electronic Data Interchange (Web-EDI). Next, this study explains why the evaluation object is SoS. This study shows below that both partially overlapping relationship and inclusive relationship in Figure 2 are SoS. In Figure 4, in the case of partially overlapping relationship, the object that works only in the range that cannot be commonized is the object that also works as a monolithic system. As a result, the SoS for the partially overlapping relationship are SoS, which is a combination of a range that cannot be commonized and a range that can be commonized. Inclusive relationship in Figure 4 shows that the range that can be commonized between the sales department and the parts department is the range that works as a monolithic system of the sales department. Additionally, the range that cannot be commonized between the sales department and the parts department is the range that works as a monolithic system of the parts department. The reason is that each department operates independently, so it is not possible to manage the range that is commonized. As well as the partially overlapping relationship, the SoS for the inclusive relationship are SoS, which is a combination of a range that cannot be commonized and a range that can be commonized. From the above, this study has clarified how the range that organizations can actually control and the range that organizations are responsible for management different positions are different depending on whether they are SoS of partially overlapping relationship or inclusive relationship.

Next, we show whether the partially overlapping relationship and the inclusive relationship in Figure 4 satisfy the condition of SoS. C, D, E, F, G and H in Figure 4 corresponds in Figure 2. We show the situation where the evaluation object is a partially overlapping relationship. From the perspective of operational independence, C and D in Figure 4 operate in a decomposed state. From the perspective of managerial independence, C in Figure 4 is managed by the Sales department and D in Figure 4 is managed by the parts department. In other words, using the architecture description method for open SoS [25], we can explain that C works as an external object (X) of D (system of interests), and D works as an external object (X) of C (system of interests). From the perspective of geographic distribution, C and D in Figure 4 are distributed and managed in a cloud environment. Furthermore, C and D can only exchange information. From the perspective of evolutionary development, the range defined as a monolithic system may evolve for each department, so C and D evolve individually.

| Procedure 1)-1: Designers design business processes and IT systems with the viewpoint of the process of X organization. | Procedure 1)-2: Using the system design results, stakeholders confirm the “process and IT system” that cannot accept the quality from the viewpoint of the process of X organization. |
| Procedure 2)-1: Using the system design and confirmation results of procedure 1), Designers perform system design of business processes and IT systems from the viewpoint of the process of Y organization. | Procedure 2)-2: Using the system design results, stakeholders confirm the “process and IT system” that cannot accept the quality from the viewpoint of the process of Y organization. |
| Procedure 3)-1: Using the system design and confirmation results of procedure 2), Designers perform system design of business processes and IT systems from the viewpoint of the process of X organization. | Procedure 3)-2: using the system design results, stakeholders confirm the “process and IT system” that cannot accept the quality from the viewpoint of the process of X organization. |
| Procedure 4)-1: Using the system design and confirmation results of procedure 3), Designers perform system design of business processes and IT systems from the viewpoint of the process of X organization. | Procedure 4)-2: Using the system design results, stakeholders confirm the “process and IT system” that cannot accept the quality from the viewpoint of the process of Y organization. |

Designers repeats from Procedure 3) to Procedure 4) until there is no change in the system design result.

Figure 3 The relationship among procedures of the proposed design method and the evaluation procedures of the proposed design method.

Vol. 13 No. 1 (2021)
Table 1 Age and organizations of participants.

| Participant | Age  | Organization |
|-------------|------|--------------|
| A           | 30’s | Designer     |
| B           | 30’s | Designer     |
| C           | 30’s | Designer     |
| D           | 50’s | Sales        |
| E           | 30’s | Sales        |
| F           | 30’s | Sales        |
| G           | 40’s | Parts        |
| H           | 40’s | Other1       |
| I           | 40’s | Other1       |
| J           | 40’s | Other2       |
| K           | 30’s | Other3       |
| L           | 20’s | Other4       |
| M           | 50’s | Other4       |

From the perspective of Emergent Behavior, only the E in Figure 4 achieves the purpose of SoS, but the operation of the E is not always assured. From the above, since the definition of SoS is satisfied, if the evaluation object of this study is the partially overlapping relationship, the evaluation object of this study is SoS.

We show the situation where the evaluation object is an inclusive relationship. From the perspective of operational independence, F and G in Figure 4 operate in a decomposed state. From the perspective of managerial independence, F in Figure 4 is managed by the Sales department and G in Figure 4 is managed by the parts department. In other words, using the architecture description method for open SoS [25], we can explain that G works as an external object (X) of F (system of interests), and F works as an external object (X) of G (system of interests). From the perspective of geographic distribution, F and G in Figure 4 are distributed and managed in a cloud environment. Furthermore, F and G can only exchange information. From the perspective of evolutionary development, the range defined as a monolithic system may evolve for each department, so F and G evolve individually. From the perspective of Emergent Behavior, only the H in Figure 4 achieves the purpose of SoS, but the operation of the H is not always assured. From the above, since the definition of SoS is satisfied, if the evaluation object of this study is the inclusive relationship, the evaluation object of this study is SoS.

Next, the changes in the deliverables created in the design procedure are in the order of 1), 2), 3) and 4) in Figure 5. Figure 5 shows both the range that can commonize and the range that cannot commonize in business processes and IT systems between sales department and parts department. We will explain the order of Figure 5 using Figure 3 as follows.

Procedure 1)-1: Designers design business processes and IT systems with the viewpoint of the accepting order process (System design result of Procedure 1)). At this point, this study confirms the business process without any IT system constraints.

Procedure 1)-2: Using the above system design results, stakeholders confirm the “process and IT system” that cannot accept the quality from the viewpoint of the accepting order process.

Procedure 2)-1: Using the system design and confirmation results of procedure 1), Designers perform system design of business processes and IT systems from the viewpoint of the maintenance process. At the same time, designers determine the range that can commonize within the system design results of Procedure 1).

Procedure 2)-2: Using the above system design results, stakeholders confirm the “process and IT system” that cannot accept the quality from the viewpoint of the maintenance process.

Procedure 3)-1: Using the system design and confirmation results of procedure 2), Designers perform system design of business processes and IT systems from the viewpoint of the accepting order process. At the same time, designers determine the range that can commonize within the system design results of Procedure 2).

Procedure 3)-2: Using the above system design results, stakeholders confirm the “process and IT system” that cannot accept the quality from the viewpoint of the maintenance process.

Procedure 4)-1: Using the system design and confirmation results of procedure 3), Designers perform system design of business processes and IT systems from the viewpoint of the maintenance process. At the same time, designers determine the range that can commonize within the system design results of Procedure 3).

Procedure 4)-2: Using the above system design results, stakeholders confirm the “process and IT system” that cannot accept the quality from the viewpoint of the maintenance process.

**Figure 4 The relationship of departments to manage and kinds of systems.**

International Journal of Japan Association for Management Systems

- 20 -
Designers repeat from Procedure 3) to Procedure 4) until there is no change in the system design result.

From the above, based on the evaluation procedure 1)-2, 2)-2, 3)-2 and 4)-2, this study evaluates the proposed design method.

5. Evaluation Results

As a result of the evaluation based on the evaluation procedure, we show the design results of the business process and the design results of the IT system in Figure 5. We show the results in procedures. For the system design in the first position used in procedure 1), the designers considered the system design that fulfills the requirements of the sales department as shown in Figure 6. The designers confirmed with the sales department and decided on the system design of the state shown in Figure 6 in procedure 1).

In procedure 2), the parts department decided that "it is impossible without actually disassembling". As a result, the designers made a decision described in Figure 7 for the system design in procedure 2). In other words, regarding order entry, the designer has determined that it is difficult to integrate product orders and parts orders. In addition, the designer has decided that orders using Web-EDI will be handled as separate processes in the IT system. At this time, a difference occurred as an input item in the product order entry. (This difference is referred to as "items that have appended due to parts sales.")

In procedure 3), the stakeholders (the sales department) agreed to the items that have appended due to parts sales (changes to the IT system). Therefore, since the agreement was obtained, there is no change in the system design between procedure 2 and procedure 3 in Figure 5.

In procedure 4), the stakeholders (the parts department) agreed to the items that have appended due to parts sales (changes to the IT system). Therefore, since the agreement was obtained, there is no change in the system design between procedure 3 and procedure 4 in Figure 5. From the above, Figure 7 shows the final system design between the sales department and the parts department. Therefore, the order process for sales and the maintenance process for the product were SoS corresponding to pattern 2 in Figure 2, two systems with a partially overlapping relationship.

6. Discussion

The evaluation of this study confirms "agreement between different organizations, which have independent business processes, and completion of design" using "IT systems used by them for different purposes" in a company.

Actually, to achieve purposes of both an order process for sales and a maintenance process for the product, a design result of considering two system designs for each purpose was the design premised SoS. In fact, since the evaluation object is a combination of the range that cannot be commonized and the range that can be commonized, the evaluation object was SoS of the partially overlapped relationship. Additionally, we confirmed that stakeholders agreed using the proposed design method in this study.

Concretely, at the stage of procedure 2) shown in the evaluation results, the parts department decided that "it is impossible without actually disassembling". Additionally, in procedure 3) of the proposed method, we were able to confirm the "the range that does not affect the function used by the sales department" compared to the conventional method by checking the system design to the parts department. Therefore, we suggested that we could confirm the usefulness of the proposed method.
As a result, we consider that the system design achieved by changing to the business process and IT system in Figure 7. The reason is that if the system design realized Figure 5, the system design realizes only the purpose of an order process for sales. It is difficult that the system design in Figure 6 realizes the purpose of a maintenance process for the product. If the judgment of system design is always considered independent by organizations (or purpose), the previous study [2] indicates that it is difficult to cope with the increase in management costs as a result of too many managed objects. In other words, it is necessary to reconsider the business process and IT system for each organization at the timing when "it is impossible without actually disassembling" is determined in each procedure. The reason is that the business process and IT system which realize the purposes of each organization can change. Therefore, if the organizations do not reconsider both the business process and IT system that are Systems-of-Systems with multiple purposes, it is difficult to realize purposes that cannot agree among organizations. In other words, the organization unknowing each organization's purposes that are not agreed among organizations misunderstands the operation of both the business process and the IT system, it thus is difficult to eliminate the misunderstanding.

We show the limitation of this study as follows:
After performing the proposed method, each organization can treat the final system in Figure 7 as a SoS in assuming that the range that commonized design does not change. In other words, designers and systems engineers designed the interface between each organization to target the business process and the IT system in this study. The limitation of this study is that the proposed method has difficulty changing the interface itself of the final system completed as a SoS, in the future. The reason is that each organization individually continues to change the final system which uses as a SoS, through evolutionary development processes which is one of the definitions of a SoS. Therefore, future research topic needs how to change the interface within a SoS after completing the final system as a SoS.

The limitation of this study is that we do not indicate an effective procedure depending on the type of SoS [26]. In addition, although this study does not show the procedure changing the process and IT system after a certain period of time, we consider the procedure to be the same. In other words, organizations need to agree to the business process and the IT system that realize the purposes of each organization while confirming different organizations' purposes. In figure 4, this study has clarified how the range that organizations can actually control and the range that organizations are responsible for management different positions are different depending on whether they are SoS of partially overlapping relationship or inclusive relationship. Therefore, future research needs a method of changing the organization so that a recognition difference does not occur depending on whether it is a partially overlapping relationship or an inclusive relationship. For example, in the case of an inclusive relationship, if another organization is included in the inclusive organization, the object treated as SoS is manageable in the inclusive organization and the object can be treated as a monolithic

---

International Journal of Japan Association for Management Systems

- 22 -
Having Different Boundaries as System-of-Systems for Each Organization?

7. Conclusions

The purpose of this study was to reduce misunderstanding at the time of SoS design in different positions that have different purposes, in order to achieve the purposes of SoSs which have different boundaries defining as SoS for each organization. We proposed a design method which assumes the purpose of this study, includes steps to make it that can monimize design by two positions. This study confirmed "agreement between different organizations, which have independent business processes, and completion of design" using IT systems used by them for different purposes" in a company. As a result, we suggested that we could confirm the usefulness of the proposed method. Additionally, according to discussion in this study, the organization unknowing each organization's purposes that are not agreed among organizations misunderstands the operation of both the business process and the IT system, thus, it is difficult to eliminate the misunderstanding. Therefore, the organizations need to reconsider both the business process and IT system that are SoS with multiple purposes, in order to realize purposes that could not agree among organizations.

Future research topics are as follows:
- Evaluation of this study does not target performance. Therefore, in order to evaluate the performance, it is necessary to confirm the next system design after actual implementation and speed verification.
- We have not indicated how to change the interface within a SoS after completing the final system as a SoS.
- We do not indicate an effective procedure depending on the type of SoS [26].
- This study does not show the procedure changing the process and IT system after a certain period of time.
- We need to confirm the results of comparison with other conventional methods and the quality and reliability of the system designed by the proposed method.

Acknowledgment

Nobuyuki Kobayashi, the first author of this research, is supported by Andgate Inc. (https://www.andgate.co.jp/).

References

[1] M. W. Maier: Architecting Principles for Systems of Systems, Systems Engineering, 1 (4) (1998), 267-284.
[2] N. Kobayashi, A. Nakamoto, M. Kawase, F. Sussan, M. Ioki, and S. Shirasaka: Managing a monolithic system or a System-of-Systems? An assurance case approach to reach intra-organizational consensus, proceedings 2018 7th International Congress on Advanced Applied Informatics (IAAI-AAI 2018), (2018), 688-693.
[3] N. Kobayashi, M. Kawase, and S. Shirasaka: A Proposal for an Assurance Case Description Method for Business Assignment Using Information Systems, Journal of Japan Association for Management Systems, 34 (3) (2018), 321-328.
[4] G. Bammer: Disciplining interdisciplinarity: Integration and implementation sciences for researching complex real-world problems, Canberra: ANU E-PR, (2013).
[5] A. K. Nandakamuran: Stackelberg's Optimization with Microstructure and Homogenization, Differential Equations and Dynamical Systems, 6 (1) (1998), 113-123.
[6] R. Xiao, Y. Wang, H. Pan, and F. Wu: Automatic Video Summarization by Spatio-temporal Analysis and Non-trivial Repeating Pattern Detection, 2008 Congress on Image and Signal Processing, (2008), 555-559.
[7] J. Li, E. Gong, Z. Sun, W. Liu, and H. Xie: AeroMTP: A fountain code-based multipath transport protocol for airborne networks, Chinese Journal of Aeronautics, 28 (4) (2015) 1147-1162.
[8] S. M. Kim: A Multilevel Analysis of the Relationship between Physical Education Requirements and Student Academic Achievement in High School, Thesis for Doctor of Philosophy in the Graduate School of the University of Maryland, (2015).
[9] R. Benthuis, M. E. Iacob, P. Havinga: A Design of the Resilient Enterprise: A Reference Architecture for Emergent Behaviors Control, Sensors (Basel). 20 (22) (2020) 6672.
[10] N. Kobayashi, A. Nakamoto, M. Kawase, F. Sussan, and S. Shirasaka: What Model(s) of Assurance Cases Will Increase the Feasibility of Accomplishing Both Vision and Strategy?, Review of Integrative Business and Economics Research, 7 (2) (2018), 1-17.
[11] N. Kobayashi, M. Kawase, and S. Shirasaka: A Proposal for an Assurance Case Description Method -Aiming to Tackle Challenges in Work-Related Communication-, Journal of Japan Association for Management Systems, 33 (2) (2016), 91-107.
[12] N. Kobayashi, A. Nakamoto, M. Kawase, and S. Shirasaka: Comparison of Two Quantitative Evaluation Methods for Assurance Cases, International Journal of Japan Association for Management Systems, 8 (1) (2016), 27-34.
[13] N. Kobayashi, M. Kawase, and S. Shirasaka: A Proposal of Assurance Case Description Method for Sharing a Company’s Vision, Journal of Japan
Association for Management Systems, 34 (1) (2017), 85-94

[14] J. Lane, and T. Bohn.: Using SySML to Evolve Systems of Systems. Los Angeles, CA, USA: USC CSSE Technical Report, USC-CSSE-2010-506, (2010).

[15] I. Cherfa, N. Belloir, S. Sadou, R. Fleurquin, D. Bennouar: Systems of systems: From mission definition to architecture description, Systems Engineering, 22 (6) (2019), 437-454

[16] J. Dahmann, G. Rebovich, J Lane, R. Lowry, K. Baldwin: An implementers' view of systems engineering for systems of systems, IEEE International Systems Conference, Montreal, Canada, (2011), 212-217.

[17] E. Silva, and T. Batista: Formal modeling systems-of-systems missions with mkaos, Proceedings of the 33rd ACM/SIGAPP Symposium On Applied Computing, (2018).

[18] E. Silva, E. Cavalcante, T. Batista, F. Oquendo, F. Delicato, and P. Pires: On the Characterization of Missions of Systems-of-Systems, Proceeding of the 2nd International Workshop on Software Engineering for Systems-of-Systems, (2014).

[19] E. Silva, T. Batista, and E. Cavalcante: A mission-oriented tool for system-of-systems modeling, Proceedings of the 3rd IEEE/ACM International Workshop on Software Engineering for Systems-of-System (SESoS’15), (2015).

[20] A. Dardenne, A. v. Lamsweerde, and S. Fickas: Goal-directed requirements acquisition, Science of Computer Programming, 20 (1) (1993), 3 – 50.

[21] L. Garcés, and E. Y. Nakagawa: A process to establish, model and validate missions of systems-of-systems in reference architectures, Proceedings of the Symposium on Applied Computing (SAC ‘17). ACM, New York, (2017), 1765–1772.

[22] E. Silva, T. Batista, and F. Oquendo: On the verification of mission-related properties in software-intensive systems-of-systems architectural design, Science of Computer Programming, 192 (15) (2020), 1-24.

[23] M. W. Maier: Architecting principles for systems-of-systems. In INCOSE 1996 6th Annual International Symposium of the International Council on Systems Engineering. INCOSE., (1996).

[24] C. B. Nielsen, P. G. Larsen, J. Fitzgerald, J. Woodcock, J. Peleska: Systems of systems engineering: basic concepts, model-based techniques, and research directions, ACM Computing Surveys, 48 (2) (2015), 1–41.

[25] N. Kobayashi, H. Fijino, S. Shirasaka: Architecture Description Method for Open Systems-of-Systems to Reduce Misunderstanding the Scopes of Managed Objects, International Journal of Japan Association for Management Systems, 12 (1) (2020), 43-50.

[26] Department of Defence: Systems Engineering Guide for Systems of Systems, Technical Report, Department of Defence, (2008). Available at http://www.acq.osd.mil/se/docs/SE-Guide-for-SoS.pdf.