Electronic Preprint for Journal of Information Processing Vol.23 No.3

Regular Paper

Applying Multidimensional Scaling for Responsibility Distribution between Objects ——Linking Conceptual Activity Model and Class Diagram

Akio Ida[1,a) Shigeo Kaneda[2]

Received: August 16, 2014, Accepted: February 4, 2015

Abstract: In the field of systems development, linking models are expected for each process to improve understandability and productivity. For example, when Soft Systems Methodology is applied to the uppermost process in a stream and an Object-Oriented Approach is applied to the upstream and midstream processes, if the conceptual activity model of the former and the class diagram of the latter can be linked, then we can build systems that are intuitively simple to understand, easily maintained, and reusable. This paper considers a system's root definition and proposes an approach for spatial arrangement through multidimensional scaling of responsibilities extracted from a conceptual activity model. We tested our proposed approach on several samples and obtained a spatial responsibility arrangement that is quite reasonable. We believe this approach has the potential to bridge artifacts from the uppermost stream, including conceptual activity models and midstream artifacts represented by class diagrams.

Keywords: Soft Systems Methodology, conceptual model, Object-Oriented Approach, responsibility for knowledge, responsibility for behavior, Multidimensional Scaling

1. Introduction

There is a growing consensus among those in the field of systems development about the requirements for increasing diversification and complexity, and therefore, model- and component-based development processes are employed to facilitate prompt and flexible implementation. For instance, since a prototype can be a process where Soft Systems Methodology (SSM) is employed in the uppermost process to achieve a consensus among the parties involved on objectives, an Object-Oriented Approach (OOA) is applied in the analysis and design stage to achieve them. By linking the obtained models using SSM in the uppermost stream, the OOA models are expected to significantly contribute to achieving those objectives.

However, currently no method links artifacts from such uppermost processes as SSM conceptual activity models and those from such upper- to midstream processes as OOA class diagrams.

Conversely, multidimensional scaling (MDS), which was developed in the field of psychology, is proving to be extremely useful for obtaining the geometrical representation of potential data structures. It has been recently used in many areas, such as marketing and behavioral science, and has also attracted significant attention in the big data boom. A previous study proposed MDS to evaluate OOA structural patterns [1]; however, no examples have demonstrated its use for linking SSM and OOA models.

In this paper, we extract both knowledge and behavioral responsibilities from the root definition of SSM relevant systems, and propose a draft standard for easy quantification of the dependency levels between them. We propose an approach that links SSM and OOA models by analysis using a non-metric MDS, which is a responsibility relationship matrix, and the spatial arrangement for the responsibility distribution of objects.

The remainder of this paper is organized as follows. Related works are discussed in Section 2. In Section 3, our proposed approach is explained by an example. In Section 4, we add consideration on reading and using our proposed approach's output. Section 5 reports the application results of our proposed approach with a different example. Finally, we state our conclusions in Section 6.

2. Overview of Related Research

2.1 Soft Systems Methodology

In situations where a problem is not clearly defined, the SSM process obtains modifications of the opinions of stakeholders' (i.e., people with different viewpoints or value systems allow each other's viewpoints or value systems to coexist during extensive debate) leading to problem resolution [2], [3].

The following are the outlines of SSM's 7 stages: The 1st stage identifies the problem situation. The 2nd expresses it. The 3rd defines the relevant systems. The 4th is a conceptual modeling of the defined systems. The 5th compares the models and the real

[1] Doshisha University, Kyotanabe, Kyoto 610–0394, Japan
[2] Graduate School of Science and Engineering, Doshisha University, Kyotanabe, Kyoto 610–0394, Japan
[a)] ideaworks@kcn.jp

world. The 6th identifies desirable and feasible changes. The 7th stage takes action to improve the problem situation.

In the 4th stage, a conceptual activity model [*1] (conceptual model: elemental activities required to achieve a purpose) is developed from the root definition of a relevant system (purposeful human activity system) that is selected from multiple candidates. In this paper, the authors assume that the system development process (such as the Rational Unified Process) is applied to implement the 5th and subsequent stages. The obtained conceptual activity models are linked to subsequent models of the 5th stage.

2.2 Object-Oriented Approach and Responsibility Distribution Problem

In OOA, responsibility is encapsulated for the knowledge necessary to satisfy the responsibility for the behavior required from a certain object. In this manner, it contributes to building systems that are easy to maintain, reuse, and scale by reducing the dependency (coupling) on other objects while increasing the objects' strength as modules.

Therefore, object identification and appropriate responsibility distribution are critical, and many rules and guidelines have been proposed, e.g., BCE by Jacobson [4], Wirfs-Brock's role stereotypes [5], Larman's GRASP [6], and Martin's SOLID principles [7]. However, these examples only provide general standards or guidelines for responsibility separation, without offering any tips about individual problems.

Some small-scale examples exist. Consider the example of developing a billiards game simulator through OOA. Determining such a simulator with just the above guidelines is difficult because the following questions must be answered. Under what class of responsibility should ball-collision detection be considered? How should the post-collision energy exchange be viewed?

One problem with OOA is that the suitability of the responsibility distribution is only evaluated after the structural and behavioral modeling of a use-case realization. We believe that it is possible to outline the responsibility distribution to objects at an earlier stage.

2.3 Multidimensional Scaling and Spatial Responsibility Distribution

MDS can be broadly divided into two types: metric and non-metric. A metric MDS is a technique for interval scale data that geometrically represents the structure hidden in the data in the space of a few dimensions. A non-metric MDS expands metric MDS by also considering ordinal scale data [8].

To clearly understand MDS, consider a map of the Japanese island of Hokkaido. The distances between its cities can be simply obtained using a ruler. If only the distance relations between the cities are given, however, placing the cities on a plane and reconstructing the map is difficult.

MDS is a technique that analytically solves such problems. **Table 1** summarizes the measured distances between eight cities in Hokkaido, and **Fig. 1** plots the data in an X-Y plane by applying MDS. If these data are superimposed on a blank map, the positional relationships are approximately identical to those between the actual cities on the map. In this example, although the directions also match the actual map, that result is merely coincidental.

Table 1 Direct distances (km) between cities in Hokkaido [9].

	Sapporo	Asahikawa	Wakkanai	Kushiro	Obihiro	Muroran	Hakodate	Otaru
Sapporo								
Asahikawa	115							
Wakkanai	274	202						
Kushiro	249	188	358					
Obihiro	152	118	313	98				
Muroran	88	200	360	290	195			
Hakodate	152	263	413	330	240	64		
Otaru	30	130	266	277	180	98	159	

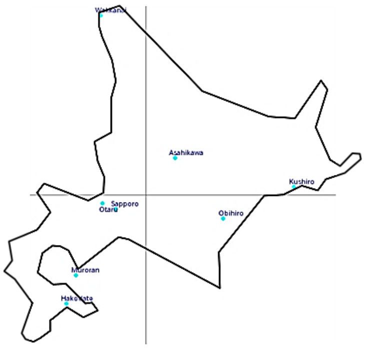

Fig. 1 Blank map overlaid with spatial arrangement using MDS (after 24 iterations, converged stress value = 0.0042).

Table 2 Stress values and goodness of fit evaluation.

Stress value	Evaluation
0.2	poor
0.1	fair
0.05	good
0.025	excellent
0	perfect

MDS is not used to visualize structures that are already known; it obtains distance relationship data (both acceptable similarity and dissimilarity data) for pairs of objects and arranges them in a space with a minimum number of dimensions (up to three is preferable for readability).

J.B. Kruskal's method [8], a type of non- metric MDS, uses the stress values as the goodness of the fit index for spatial arrangements. Stress values broadly correspond to the goodness of the fit evaluation, as outlined in **Table 2** [8], [9].

3. Proposed Approach

Based on these situations and outcomes, this paper proposes an approach that uses MDS to link SSM and OOA models. We assume that an object-oriented development process (such as the Rational Unified Process) is applied later to the implementation of SSM's 5th stage [*2].

We propose a draft standard to extract the responsibilities for knowledge (RFKs) and responsibilities for behavior (RFBs) from each node of the conceptual activity model that is required to realize the root definition of a relevant system. We also propose easy quantification of their dependency levels.

In our proposed approach, RFBs are defined as the required behaviors to realize the root definition of the relevant system, and

[*1] In this paper, we use "conceptual activity model" to distinguish it from the "conceptual model" in the object-oriented approach.

[*2] Activities that already exist in the traditional SSM is not included in this proposal.

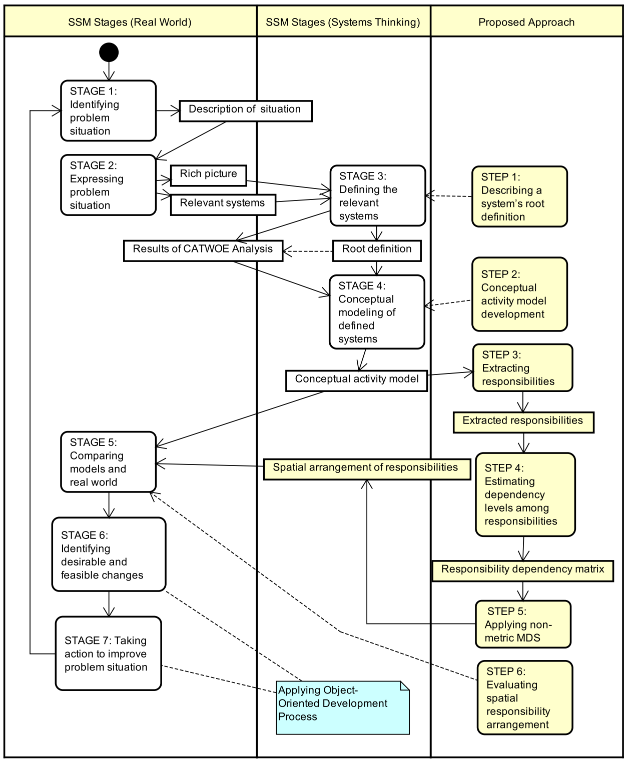
SSM Stages (Real World)
SSM Stages (Systems Thinking)
Proposed Approach
STAGE 1: Identifying problem situation
Description of situation
STAGE 2: Expressing problem situation
Rich picture
Relevant systems
STAGE 3: Defining the relevant systems
STEP 1: Describing a system's root definition
Results of CATWOE Analysis
Root definition
STAGE 4: Conceptual modeling of defined systems
STEP 2: Conceptual activity model development
Conceptual activity model
STEP 3: Extracting responsibilities
Extracted responsibilities
STAGE 5: Comparing models and real world
Spatial arrangement of responsibilities
STEP 4: Estimating dependency levels among responsibilities
STAGE 6: Identifying desirable and feasible changes
Responsibility dependency matrix
STEP 5: Applying non-metric MDS
STAGE 7: Taking action to improve problem situation
STEP 6: Evaluating spatial responsibility arrangement
Applying Object-Oriented Development Process

Fig. 2 Overview of SSM and our proposed approach.

RFKs are the required knowledge for achieving the RFBs. In each node of the conceptual activity model, a RFB takes the form of an action verb and is a candidate for the operation for an object. A RFK takes the form of a countable noun or a noun phrase, and is a candidate for an object or an attribute.

Using a non-metric MDS, we analyze the responsibility dependency level matrix and attempt the spatial arrangement of responsibilities.

Figure 2 shows an overview of our proposed approach. Step 1 extends the activities in SSM's 3rd stage. Step 2 extends the work in its 4th stage. Steps 3, 4, and 5 are additional steps to the legacy SSM. Step 6 adds viewpoints from the spatial arrangement of the responsibilities.

As shown in Fig. 2, our proposed approach extracts responsibilities from the conceptual activity model that was developed from SSM's root definition. The dependency levels between the responsibilities are obtained and a spatial responsibility arrangement is acquired. The approach then uses this result as a reference for subsequent SSM stages and OOA class diagram generation. We explain the process of our proposed approach using some examples below.

3.1 STEP 1: Describing a System's Root Definition

System requirements basically define "when," "by whom," and "what for (why)," a system is used. For instance, in a use case model, "when" corresponds to the business event, "who" to the actors, and "what for" to the use case. On the other hand, in SSM a representation in the following format is termed a root definition of a relevant system: "a system to do X, by Y, to achieve Z." Representing "thought activity" using a root definition is important to guarantee recoverability, which is an academic standard in SSM [2].

Our proposed approach adds information regarding "who" (W) to the SSM root definition as the format to define requirements in SSM. This "who" (W) is the customer defined in the CATWOE analysis of SSM's 3rd stage and uses the following format: "a system where W does X by Y to achieve Z." Adding W facilitates the extraction of each node of the conceptual activity model as an action sentence by clarifying the subject as the actor. The following are examples of root definitions used in our proposed approach:

Example 1: *"A system where parents who are considering enrolling their child in a tutoring academy (W) do a comparative investigation (X) by searching for tutoring academies, subjects, locations, tuition fees, special features, and tutors (Y) to select an academy (Z)."*

Example 2: *"A system where a billiards enthusiast (W) learns how to strike a ball (X) by simulating the game on a computer (Y) to improve her skill (Z)."*

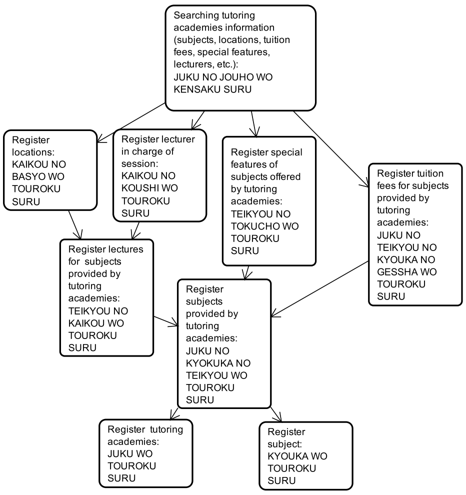
Searching tutoring academies information (subjects, locations, tuition fees, special features, lecturers, etc.): JUKU NO JOUHO WO KENSAKU SURU
Register locations: KAIKOU NO BASYO WO TOUROKU SURU
Register lecturer in charge of session: KAIKOU NO KOUSHI WO TOUROKU SURU
Register special features of subjects offered by tutoring academies: TEIKYOU NO TOKUCHO WO TOUROKU SURU
Register tuition fees for subjects provided by tutoring academies: JUKU NO TEIKYOU NO KYOUKA NO GESSHA WO TOUROKU SURU
Register lectures for subjects provided by tutoring academies: TEIKYOU NO KAIKOU WO TOUROKU SURU
Register subjects provided by tutoring academies: JUKU NO KYOKUKA NO TEIKYOU WO TOUROKU SURU
Register tutoring academies: JUKU WO TOUROKU SURU
Register subject: KYOUKA WO TOUROKU SURU

Fig. 3 Conceptual activity model for Example 1 obtained in this step developed from root definition.

3.2 STEP 2: Development into a Conceptual Activity Model

A relevant system's root definition encapsulates its functional requirements. However, because it contains insufficient information, the responsibilities required by the system components cannot be extracted from such a simple definition. Therefore, we developed a conceptual activity model from the root definition in modeling sessions between these involved persons, e.g., those who request or provide system development.

A conceptual activity model (CM) expresses the activities to achieve the means (Y) which expressed in the root definition. The number of nodes of a CM is recommended approximately 7 ± 2. Each node of the CM expresses a predicate, its subject is W (but not written). The node format is action verb (V) + a countable noun (O) or V + noun phrase (O). In Japanese, the format is "O を V する." If relations of "owner-possession" are recognized between the countable nouns included in the noun phrase, you must constitute it by connecting those countable nouns with Japanese particle 'の'.

Figure 3 is an example output of the conceptual activity model

Table 3 Extracted RFKs and RFBs for Example 1.

Extracted RFKs	Extracted RFBs
Tutoring academy	Register tutoring academy
Subject	Register subject
Provision by academy of subjects	Register provision
Tuition fee for provision	Register tuition fees
Special features of provision	Register special features
Session of provisions	Register session
Lecturer in charge of session	Register session lecturer in charge
Location of session	Register session location
	Search tutoring academy information

for example 1, which is developed from the root definition in this step.

3.3 STEP 3: Extracting Responsibilities from the Conceptual Activity Model

We extracted countable nouns and verbs from each node of the CM as RFKs and RFBs, respectively. This is considerably easier than extracting objects from a Japanese language requirement specification. This is because a conceptual activity model follows the format, "V (action verb) + O (countable noun)," unlike the unformatted requirement descriptions in Japanese, and because of the nature of Japanese, which contains many existential clauses (not action clauses) [10].

In previous research, we clarified the correlation between English word classes and class diagram elements [11]. Class or attribute names are countable nouns, and operation names are action verbs. Even if it is an action verb, however, the thing that should record the action's result assumes it a class.

It is difficult to judge quickly whether the countable nouns in predicates are class or attribute names. However, below in Section 3.6, we apply MDS and consider the spatial arrangements of responsibilities. Therefore determining whether they are classes or attributes, is not necessary at this step. Here we simply regard countable nouns and action verbs as RFKs and RFBs, respectively.

Table 3 shows the extracted RFKs and RFBs in this step.

3.4 STEP 4: Estimating Dependency Levels among Responsibilities

Since the present approach uses MDS at a later step, the dependency levels among all responsibilities were extracted in Section 3.3 because the object, attribute, and operation candidates collectively need to be quantified.

If the dependency levels are estimated manually, a very convenient method is desired. We chose the following standard for quantification based on a trial and error process. Since the values only have to satisfy an ordinal scale depending on the strength of the dependency, integral values 0–4 (five levels) were used. However, because 4 (strongest) denotes the dependency level between identical responsibilities, it is used as the default value for diagonal elements. 0 signifies an undetermined relationship. Since we do not display diagonal elements and 0, they are not featured in the tables below. The aim of this quantification is to obtain ordinal scale data that is satisfied among all responsibilities for the following quantification standards. In order to achieve this goal, we assume dependency levels, among RFKs and RFKs, RFBs and RFKs, RFBs and RFBs, which must have an ordinal scale level.

Table 4 Quantification standard for dependency levels between RFKs.

Dependency level	Meaning	Example
3	Both are linked by particle 'の' expressing affiliation or possession and there is one affiliation or possessor	KAIKOU *NO* TANTOU KOUSHI Lecturer in charge **of** session
2	Both are linked by particle 'の' expressing affiliation or possession but there are two affiliations or possessors	JYUKU *NO* KYOUKA *NO* TEIKYOU Provision **by** tutoring academy **of** subjects (can be split into provision by tutoring academies, provision of subjects)
1	Both are linked by particle 'の' expressing affiliation or possession but there are three affiliations or possessors	SHYOHIN *NO* SOUKO *NO* KIMATSU *NO* ZAIKO Stock **of** products, stock **in** warehouse, stock **at** end-of-period
0	None of the above	

Table 5 Obtained dependency levels among RFKs for Example 1.

Knowledge \ Knowledge	Tutoring academy	Subject	Provision by academy of subjects	Tuition fee for provision	Special features of provision	Session of provisions	Lecturer in charge of session	Location of session
Tutoring academy								
Subject								
Provision by academy of subjects	2	2						
Tuition fee for provision			3					
Special features of provision			3					
Session of provisions			3					
Session lecturer in charge						3		
Location of session						3		

3.4.1 Quantification of Dependency Levels among RFKs

This quantification standard is based on the idea that when a couple of RFKs (countable nouns) have "owner-possession" relation then they are depending on each other. We think such RFKs should be placed closer to each other. Actually, in a class diagram, when RFK A owns RFK B, RFK B is modelled as an attribute of RFK A. Also, when RFK A owns RFK B, the relation between RFK A and RFK B is modelled as a "has_a" association. Therefore we evaluate that there is dependency between RFK A and RFK B.

We focused on a Japanese case particle 'の' in the quantification to estimate dependency levels between RFKs. 'の' frequently expresses an affiliational or possessional relationship between two countable nouns at the predicate in the nodes of Japanese written in conceptual activity models. In this case the dependency levels between RFKs (i.e., countable nouns) linked with these particles can be considered strong. In other words, the dependency level between two countable nouns connected 'の' can be quantified at a relatively strong-level.

For example, to realize *Register lecturer in charge of session* a lecturer has to be determined before the session is offered. To realize *Register subjects provided by tutoring academies* the provided subjects, the tutoring academies and the subjects have to be previously registered.

Table 4 shows the standard that we employed, and **Table 5** shows the results obtained for the dependency levels between RFKs based on it.

3.4.2 Quantification of Dependency Levels among RFBs and RFKs

This quantification standard is based on the idea that verbs that manage the lifecycles of certain nouns should be placed closer together than other verbs that do not manage their lifecycles. These responsibility dependency levels are obtained through CRUD, which is an acronym for Create, Read, Update, and Delete, the four basic functions in persistent object software handling. A *Create* or *Delete* controls the lifecycle of its target; therefore it is assigned the second highest score (i.e., 3). A *Read* refers only to

Table 6 Quantification standard for dependency levels among RFBs and RFKs.

Dependency level	Meaning	Example
C or D=3	Said responsibility for behavior *creates* or *deletes* said responsibility for knowledge	Between *Register subject* and *Subject* (a *Subject* is created when the *Subject* is registered)
U=2	Said responsibility for behavior *updates* said responsibility for knowledge	Between *Register session location* and *Session* (when a *Session location* is created, the *Session* is updated)
R=1	Said responsibility for behavior *reads* said responsibility for knowledge	Between *Register provision* and *Tutoring academy, Register provision* and *Subject* (the *Tutoring academy* and the *Subject* are readed, when a *Provision* is created)
0	None of the above	

Table 7 Obtained relationship levels between RFBs and RFKs for Example 1.

Behavior \ Knowledge	Tutoring academy	Subject	Provision by academy of subjects	Tuition fee for provision	Special features of provision	Session of provisions	Lecturer in charge of session	Location of session
Register tutoring academy	C							
Register subject		C						
Register provision	R	R	C					
Register tuition fees	R	R	U	C				
Register special features	R	R	U		C			
Register session	R	R	R			C		
Register session lecturer in charge	R	R	R			U	C	
Register session location	R	R	R			U		C
Search tutoring academy information	R	R	R	R	R	R	R	R

its target's attribute value; therefore it is assigned the second lowest score (i.e., 1). An *Update* changes its target's attribute value; therefore it is assigned the middle score (i.e., 2).

For example, *Register subject* has a side effect that creates a new knowledge *subject*. In addition, *Register location* has a side effect that creates new knowledge *location*, and updates the existing knowledge *Provision by academy of subjects*.

Table 6 shows our employed standard, and **Table 7** shows the results of its application to Example 1.

3.4.3 Quantification of Dependency Levels among RFBs

This quantification standard is based on the idea that verbs that directly use it for certain verbs should be placed closer than verbs that do not use it or use it indirectly.

We obtained the quantification of the dependency levels between RFBs from the connection statuses between activities in the conceptual activity model. For example, in Fig. 2, the node, *Register subjects provided by tutoring academies* directly uses *Register tutoring academies* and *Register subject*.

The dependency levels between the RFBs obtained by this quantification standard do not contribute to a better spatial arrangement, but at least they are not detrimental. This quantification standard gives a value that exceeds 0 for the pair of action verbs that are directly linked with an arrow. Of course, as we delve further, we might find a dependency level that exceeds 1, even between action verbs noted in the nodes of the conceptual activity model that are linked indirectly by arrows. Furthermore, we believe that a dependency is needed from the caller node to the called node but not in the reverse direction. This quantification standard includes the above consideration. This explains why the scored value is not 3, even though two nodes are linked directly in the conceptual activity model.

Table 8 shows the employed standard, and **Tables 9** and **10** show the dependency levels between the RFBs obtained from Example 1.

Table 8 Quantification standard for dependency levels among RFBs.

Dependency level	Meaning	Example
1	One responsibility for behavior and another responsibility for behavior are connected by arrows in the conceptual activity model	Between *Register subjects provided by tutoring academy* and *Register tutoring academy*
0	None of the above	

Table 9 Obtained dependency levels among RFBs for Example 1.

Behavior \ Behavior	Register tutoring academy	Register subject	Register provision	Register tuition fees	Register special features	Register session	Register session lecturer in charge	Register session location	Search tutoring academy information
Register tutoring academy									
Register subject									
Register provision	1	1							
Register tuition fees			1						
Register special features			1						
Register session			1						
Register session lecturer in charge						1			
Register session location						1			
Search tutoring academy information				1	1		1	1	

3.5 STEP 5: Applying Non-metric MDS

Once the dependency levels between the responsibilities have been quantified, and the lower triangular matrix has been obtained from the square matrix, MDS is applied. Since the values for the responsibility dependency levels used in this approach are on an ordinal scale, a non-metric MDS must be selected. The Dependency levels are regarded as approximations between responsibilities.

Various methods exist for the application of MDS, such as the use of statistical packages including SPSS or open source products including the R language for statistical analysis. However, we decided to use a manually written program based on the implementation of Kruskal's method in Ref. [12]. As the contents are well known in the case of a manually written program, this will be useful for the incorporation of MDS into a conceptual activity model editor in future studies.

Processing the data of Table 9 with MDS obtains the spatial responsibilities arrangement in **Fig. 4**.

3.6 STEP 6: Evaluating Spatial Responsibility Arrangement

Figure 4 clearly shows that *Tutoring academy* and *Subject* are separated and that responsibilities *Tuition fees for provision* and *Special features of provision* are exactly in the middle of the imaginary line that connects the two classes. In Fig. 4 RFKs are displayed with upper or under lines.

The distance between the responsibilities in the spatial arrangement obtained by our proposed method is not a difference in kilometers, like the spatial arrangement of cities; rather, it is the difference in the dependency level between responsibilities. Closely arranged responsibilities have a strong relationship with each other and should be placed in the same class when they are encapsulated. If they are not placed in the same class, perhaps the degree of coupling to each other shows very high multiple classes. On the other hand, separately arranged responsibilities have a weak dependency relationship, suggesting that they need not be placed in the same class when they are encapsulated.

Information of the spatial arrangement is effective as a clue for the distribution of the responsibilities to the objects, which is an important matter in the Object-Oriented Approach.

Figure 5 suggests the following conclusion: *Provision by*

Table 10 Obtained dependency levels between responsibilities for Example 1 (Overall view for Tables 5, 7, and 9).

Responsibilities \ Responsibilities	Tutoring academy	Subject	Provision by academy of subjects	Tuition fee for provision	Special features of provision	Session of provisions	Lecturer in charge of session	Location of session	Register tutoring academy	Register subject	Register provision	Register tuition fees	Register special features	Register session	Register session lecturer in charge	Register session location	Search tutoring academy information
Tutoring academy																	
Subject																	
Provision by academy of subjects	2	2															
Tuition fee for provision			3														
Special features of provision			3														
Session of provisions			3														
Lecturer in charge of session						3											
Location of session						3											
Register tutoring academy	3																
Register subject		3															
Register provision	1	1	3						1	1							
Register tuition fees	1	1	2	3							1						
Register special features	1	1	2		3						1						
Register session	1	1	1			3					1						
Register session lecturer in charge	1	1	1			2	3				0			1			
Register session location	1	1	1			2		3			0			1			
Search tutoring academy information	1	1	1	1	1	1	1	1			0	1	1		1	1	

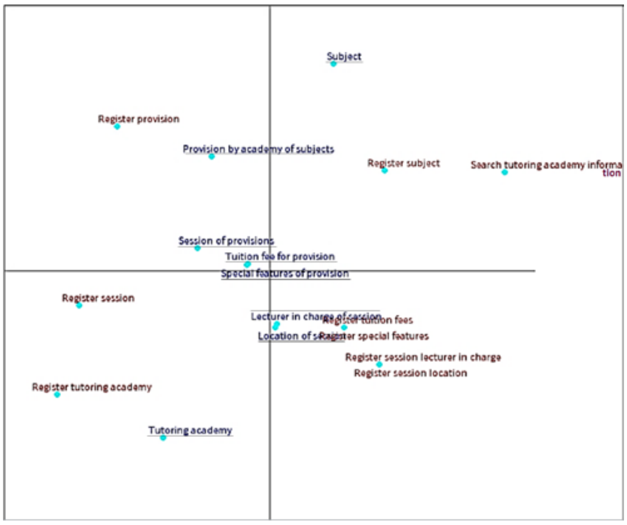

Fig. 4 Spatial responsibilities arrangement for Example 1, obtained by MDS (after 44 iterations, converged stress value = 0.188).

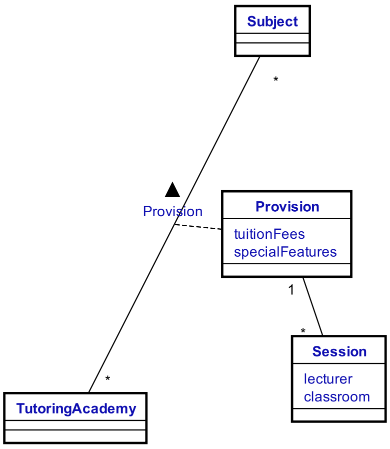

Fig. 5 Manually created class diagram for Example 1 (Inclined notation shows contrast with Fig. 4).

academy of subjects has nearly equal dependence between *Tutoring academy* and *Subject*. The dependence between *Session of provision* and *Provision by academy of subject* is stronger than *Session of provision* and *Tutoring academy*, and *Session of provision* and *Subject*. *Tuition fee for provision* and *Special features of provision* have a strong dependence to *Provision by academy of subjects*. Therefore, these three RFKs can be encapsulated into the same class. Strong dependence is also recognized between *Lecturer in charge of session* and *Location of session*. Therefore, these two RFKs can be encapsulated into the same class, which also has a strong dependency to *Session of provision*.

Although the stress value is high at 0.188 and the spatial arrangement of the RFBs is not so good, we successfully obtained a RFK structure that closely resembles a manually created class diagram (Fig. 5). The spatial arrangement of the RFBs is not very good because the quantification standard of the dependence level among RFBs is not substantial.

4. Discussion

4.1 Instructions for Reading Obtained Spatial Arrangement

We outline some considerations regarding the interpretation of the spatial arrangement of responsibilities. Because no relations are shown in the spatial arrangement, reading requires the support of connecting lines between the responsibilities. These connections are determined from the system's root definitions. A conceptual activity model is an "objective → means" graph developed for determining the root definition (objective) of the relevant system, and therefore, the RFKs (objects or attributes) entered on each node are essential to achieve the objective.

If maintainability, understandability, and reusability are ignored, a functional system can be designed even if they are united in one class. Thus, direct or indirect connections originally exist between all RFKs. According to the MDS output results, if the responsibility groups that are assembled in the vicinity are extracted as classes, then one must assume the existence of connections between them.

In software engineering, the distribution of responsibilities is always designed based on the perspective that "if elements with intrinsically strong connections are massed together, they are stable against future remodeling." The MDS output supports this

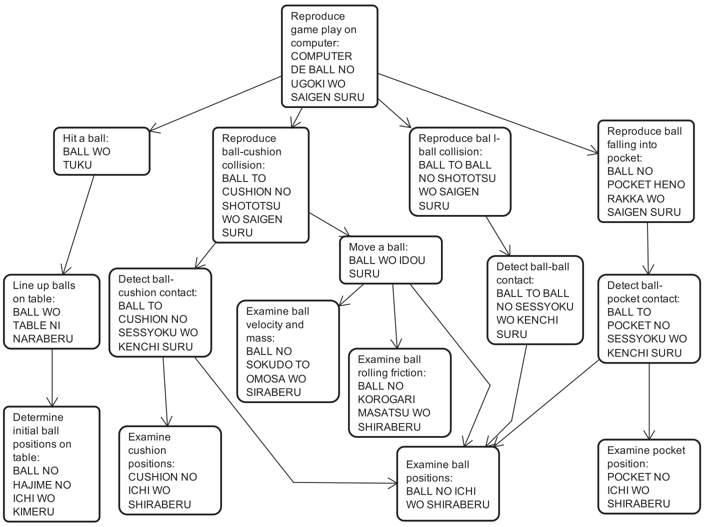
Reproduce game play on computer: COMPUTER DE BALL NO UGOKI WO SAIGEN SURU
Hit a ball: BALL WO TUKU
Reproduce ball-cushion collision: BALL TO CUSHION NO SHOTOTSU WO SAIGEN SURU
Reproduce bal l-ball collision: BALL TO BALL NO SHOTOTSU WO SAIGEN SURU
Reproduce ball falling into pocket: BALL NO POCKET HENO RAKKA WO SAIGEN SURU
Move a ball: BALL WO IDOU SURU
Line up balls on table: BALL WO TABLE NI NARABERU
Detect ball-cushion contact: BALL TO CUSHION NO SESSYOKU WO KENCHI SURU
Examine ball velocity and mass: BALL NO SOKUDO TO OMOSA WO SIRABERU
Detect ball-ball contact: BALL TO BALL NO SESSYOKU WO KENCHI SURU
Detect ball-pocket contact: BALL TO POCKET NO SESSYOKU WO KENCHI SURU
Examine ball rolling friction: BALL NO KOROGARI MASATSU WO SHIRABERU
Determine initial ball positions on table: BALL NO HAJIME NO ICHI WO KIMERU
Examine cushion positions: CUSHION NO ICHI WO SHIRABERU
Examine ball positions: BALL NO ICHI WO SHIRABERU
Examine pocket position: POCKET NO ICHI WO SHIRABERU

Fig. 6 Conceptual activity model for Example 2.

viewpoint.

4.2 Discussion On Using Spatial Arrangement

SSM's conceptual activity model is a verb-oriented model. On the other hand, our proposed spatial arrangement is a noun oriented model. They are relations that supplement each other.

We can use SSM's 5th stage, as a basis for the discussion of potential changes to real world problem situations expressed in the 2nd stage. The comparison allows the examination of differences between the current real world situation and the models of relevant systems and facilitates debate about why these differences exist [3]. The comparison focuses on possible better ways of carrying out activities or identifying new knowledge.

In this paper, we assume that a system development process (such as the Rational Unified Process) is applied to the implementation of the 5th and subsequent stages, to implement actions that are designed to fulfil changes identified as both desirable and feasible. This is itself a new problem situation that requires the design or redesign of relevant systems. Having identified what must be done, the above models are now a hard system problem, and can be tackled using a traditional engineering-based approach.

5. Application to a Different Example

We applied our proposed approach to the second example given in Section 3.1 and confirmed a change in the stress value when the number of responsibilities was increased slightly. **Figure 6** shows the development of the conceptual activity model from the root definition results.

Table 10 was obtained by extracting the responsibilities from the conceptual activity model and estimating and arranging the various responsibility dependency levels. **Figure 7** was obtained by applying MDS to the data in **Table 11**. In Fig. 7 the RFKs are displayed with upper or underlined. The stress value improved to 0.092, which probably reflects the increased number of responsibilities and a greater volume of information.

Note that in Fig. 7, the *Rolling friction of ball* is closer to *Table* than *Balls on Table*, the *Detect ball-ball contact* responsibility is closer to *Table*, and the *Reproduce ball-ball collision* responsibil-

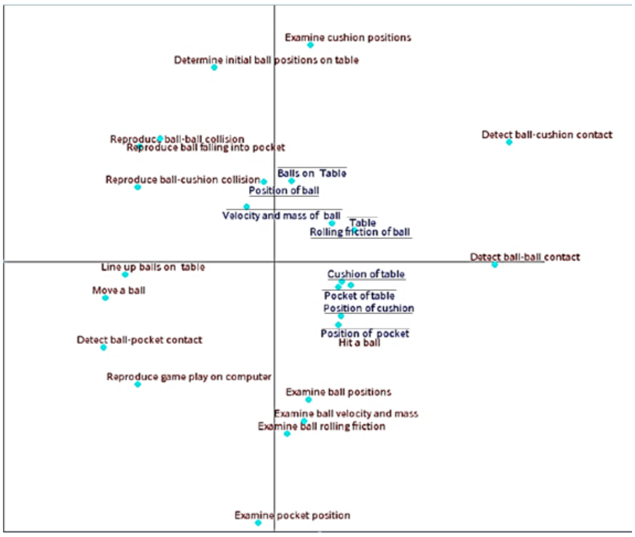

Fig. 7 Spatial responsibility arrangement for Example 2 (after 73 iterations, converged stress value = 0.092).

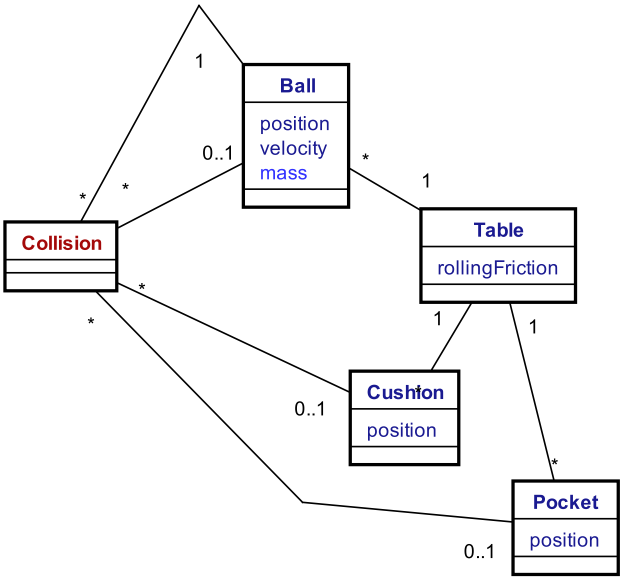

Fig. 8 Class diagram based on spatial arrangement for Example 2.

Table 11 Obtained dependency levels between responsibilities for Example 2.

Responsibilities \ Responsibilities	Table	Cushion of table	Pocket of table	Balls on Table	Position of cushion	Position of pocket	Position of ball	Velocity and mass of ball	Rolling friction of ball	Determine initial ball positions on table	Examine cushion positions	Examine ball positions	Examine pocket position	Examine ball velocity and mass	Examine ball rolling friction	Line up balls on table	Detect ball–cushion contact	Move a ball	Detect ball–ball contact	Detect ball–pocket contact	Hit a ball	Reproduce ball–cushion collision	Reproduc e ball–ball collision	Reproduce ball falling into pocket	Reproduce game play on computer
Table																									
Cushion of table	3																								
Pocket of table	3																								
Balls on Table	3	2	2																						
Position of cushion	2	3																							
Position of pocket	2		3																						
Position of ball	2			3																					
Velocity and mass of ball				3																					
Rolling friction of ball				3																					
Determine initial ball positions on table	1			2			2	2																	
Examine cushion positions	1	1			1																				
Examine ball positions				1			1																		
Examine pocket position			1			1																			
Examine ball velocity and mass				1				1																	
Examine ball rolling friction				1					1																
Line up balls on table	1			1			2	2		1															
Detect ball–cushion contact		1		1	1		1																		
Move a ball		1	1	1	1	1	1	2	1			1		1	1										
Detect ball–ball contact				1			1					1													
Detect ball–pocket contact			1	1		1	1					1	1												
Hit a ball				2			2	2								1									
Reproduce ball–cushion collision		1			1		2	2			1	1													
Reproduce bal l–ball collision				2			2	2	1																
Reproduce ball falling into pocket			1	2		1	2	2												1					
Reproduce game play on computer	1	1	1	1	1	1	1	1	1												1	1	1	1	

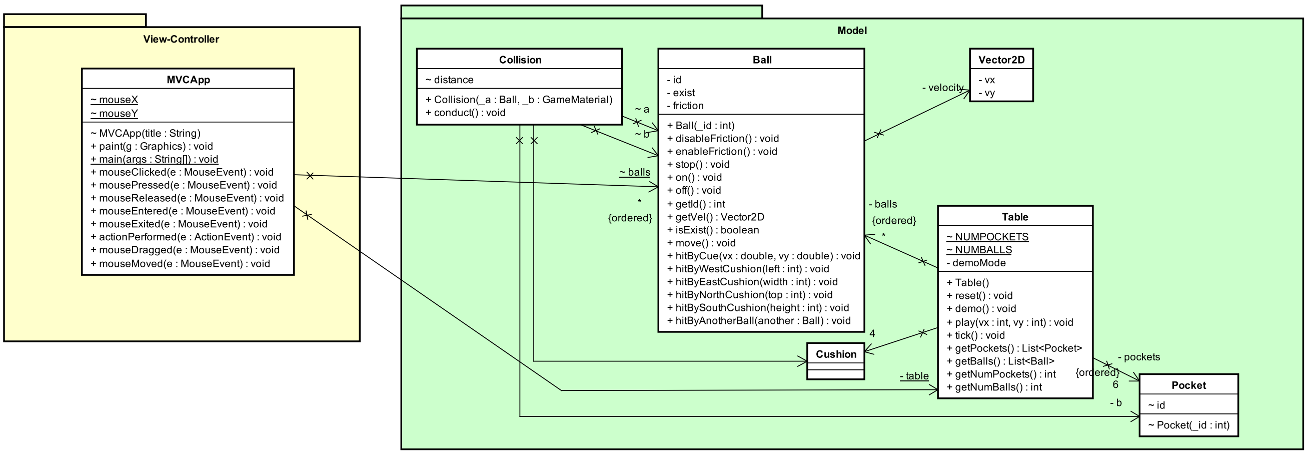
View-Controller
MVCApp
~ mouseX
~ mouseY
~ MVCApp(title : String)
+ paint(g : Graphics) : void
+ main(args : String[]) : void
+ mouseClicked(e : MouseEvent) : void
+ mousePressed(e : MouseEvent) : void
+ mouseReleased(e : MouseEvent) : void
+ mouseEntered(e : MouseEvent) : void
+ mouseExited(e : MouseEvent) : void
+ actionPerformed(e : ActionEvent) : void
+ mouseDragged(e : MouseEvent) : void
+ mouseMoved(e : MouseEvent) : void
Model
Collision
~ distance
+ Collision(_a : Ball, _b : GameMaterial)
+ conduct() : void
~ a
~ b
~ balls
*
{ordered}
Ball
- id
- exist
- friction
+ Ball(_id : int)
+ disableFriction() : void
+ enableFriction() : void
+ stop() : void
+ on() : void
+ off() : void
+ getId() : int
+ getVel() : Vector2D
+ isExist() : boolean
+ move() : void
+ hitByCue(vx : double, vy : double) : void
+ hitByWestCushion(left : int) : void
+ hitByEastCushion(width : int) : void
+ hitByNorthCushion(top : int) : void
+ hitBySouthCushion(height : int) : void
+ hitByAnotherBall(another : Ball) : void
- velocity
Vector2D
- vx
- vy
- balls
{ordered}
*
Table
~ NUMPOCKETS
~ NUMBALLS
- demoMode
+ Table()
+ reset() : void
+ demo() : void
+ play(vx : int, vy : int) : void
+ tick() : void
+ getPockets() : List<Pocket>
+ getBalls() : List<Ball>
+ getNumPockets() : int
+ getNumBalls() : int
4
Cushion
- table
- pockets
{ordered}
6
- b
Pocket
~ id
~ Pocket(_id : int)

Fig. 9 Design level class diagram for Example 2 (only major packages and classes excerpted).

ity is closer to *Balls on Table*. The class diagram in **Fig. 8** was created based on the spatial positions in Fig. 7. Although this is somewhat subjective, the information was very useful when creating Fig. 8.

Furthermore, we considered the classes shown in Fig. 8 as a Model part in the Model-View-Controller architecture pattern [13] and made a class diagram as a design model by adding View and Controller classes to them (**Fig. 9**).

The subsequent implementation trial using Java produced a favorable outcome. In particular, the separation of the contact detection and the collision recurrence in the model of the analysis stage, made possible by the arrangement of the MDS, reduced the responsibility of the ball class and made the subsequent detailed design and implementation work easier.

It is not clear from the experimental results whether object placement based on the standard outlined in the present proposal is truly appropriate in the context of the functional requirements, but we believe that it is at least consistent with the Object Oriented paradigm, which encapsulates responsibilities with the objective of achieving high cohesion and low coupling.

Therefore, we consider this approach to be a potential method to bridge artifacts from the uppermost stream such as conceptual activity models and midstream artifacts represented by class diagrams. Furthermore, our proposed method to connect SSM artifacts with the OOA responsibility distribution design exhibits originality.

6. Conclusion

This paper employs Soft Systems Methodology (SSM) as an uppermost stream methodology and Object-Oriented Approach (OOA) as a mid- to upstream methodology to extract a conceptual activity model from a system root definition to quantify the dependency levels between extracted responsibilities and to use the spatial arrangement obtained through Multi-dimensional Scaling (MDS) as a reference to create a class diagram.

Although we generally locate objects in an object-oriented manner, it is also rational to identify their responsibilities, arrange them in a space, create responsibility groups based on their dependency levels, and consider them from this perspective. Our proposed method also exhibits originality by connecting SSM artifacts with the OOA responsibility distribution design.

However, because manual implementation of our proposed approach is extremely complex, future work will implement a conceptual activity model editor with an embedded MDS and automate its process from responsibility extraction to spatial arrangement. Automation can generate more detailed responsibility dependency levels. For instance, the number of arrows making their way through a specific node to another node in a conceptual activity model could color the responsibility dependency levels.

If our approach is implemented, it may be possible to realize an intelligent modeling tool that proposes class diagrams. Our future work will continue along these lines.

References

[1] Iijima, T. and Nagata, M.: Analysis and evaluation of structural object-oriented design patterns by clustering and multi-dimensional scaling methods, Technical-report, SIG-KBSE, IEICE of Japan, KBSE95-2, pp.9–16 (1995). (in Japanese)

[2] Checkland, P. and Scholes, J.: *Soft Systems Methodology in Action*, John Wiley & Sons (1999).

[3] Wilson, B.: *Systems: Concepts, Methodologies, and Applications*, John Wiley & Sons (1999).

[4] Jacobson, I., Jonsson, P., Christerson, M. and Övergaard, G.: *Object Oriented Software Engineering: A Use Case Driven Approach*, Addison-Wesley Professional (1992).

[5] Wirfs-Brock, R. and McKean, A.: *Object Design: Roles, Responsibilities, and Collaborations* (*Addison-Wesley Object Technology Series*), Addison-Wesley (2002).

[6] Larman, C.: *Applying UML and Patterns: An Introduction to Object-Oriented Analysis and Design and Iterative Development*, Prentice Hall (2004).

[7] Martin, R.C.: *Agile Software Development, Principles, Patterns, and Practices*, Pearson Education (2011).

[8] Kruskal, J.B.: Nonmetric Multidimensional Scaling: A Numerical Method, *Psychometrica*, Vol.29, pp.115–129 (1964).

[9] Nakamura, N.: *Methods for Multidimensional Data Analysis - Learning Data Sciences in* R, Kyoritsu Shuppan (2009). (in Japanese)

[10] Fujisawa, Y., Okada, Y., Ichinose, K. and Kaneda, S.: Proposal of operation analysis method by using Soft Systems Methodology and Conceptual Data Modeling, IPSJ SIG Technical Report, 2010-IS-111(6), pp.1–8 (2010). (in Japanese)

[11] Kaneda, S., Ida, A. and Sakai, T.: Guidelines for Class Diagram Design based on English Sentence Patterns and Functional Dependency, Technical Report, SIG-KBSE, IEICE of Japan (2014). (in Japanese)

[12] Sugimoto, F.: Multi-dimensional Scaling, *Information Science*, pp.125–130 (1983). (in Japanese)

[13] Buschmann, F., Rohnert, H., Stal, M., Meunier, R. and Sommerlad, P.: *Pattern-Oriented Software Architecture, A System of Patterns*, John Wiley & Sons (1996).

Akio Ida was born in Nara, Japan. He graduated from Doshisha University in 1984. He worked for several years as a bank clerk and as a system engineer. He joined the Osaka Gas Information System Research Institute (OGIS-RI) in 1989. He has held education and consulting positions in connection with object-oriented analysis and design. He is currently president of the IDA company and a lecturer at Doshisha University. He is a member of IPSJ and IEICE of Japan.

Shigeo Kaneda was born in Hyogo, Japan and received a B.Eng. Degree in electrical and electronic engineering from Kyoto University in 1974 a M.Eng. degree in electrical and electronic engineering from the Graduate School of Kyoto University in 1976, and a D.Eng. degree in engineering from Kyoto University in 1983. He joined the Musashino Electrical Communication Laboratories, Nippon Telegraph and Telephone Public Corporation (NTT) in 1976. He became a professor at the Graduate School of Policy and Management, Doshisha University, Kyoto, Japan in 1997. He is currently a professor of the Graduate School of Science and Engineering, Doshisha University. He is a member of IPSJ, IEICE of Japan, and IEEE.