9-2018

Challenges in learning UML: From the perspective of diagrammatic representation and reasoning

Z. SHEN

S. TAN

Keng SIAU
Singapore Management University, klsiau@smu.edu.sg

Communications of the Association for Information Systems

Research Paper DOI: 10.17705/1CAIS.04330 ISSN: 1529-3181

Challenges in Learning Unified Modeling Language: From the Perspective of Diagrammatic Representation and Reasoning

Zixing Shen
College of Business and Information Systems
Dakota State University
zixing.shen@dsu.edu

Songxin Tan
Department of EECS
South Dakota State University

Keng Siau
Department of Business and Information Technology
Missouri University of Science and Technology

Abstract:

Unified modeling language (UML) is widely taught in the information systems (IS) curriculum. To understand UML in IS education, this paper reports on an empirical study that taps into students' learning of UML. The study uses a concept-mapping technique to identify the challenges in learning UML notational elements. It reveals that some technical properties of UML diagrammatic representation, coupled with students' cognitive attributes, hinder both perceptual and conceptual processes involved in searching, recognizing, and inferring visual information, which creates learning barriers. This paper also discusses how to facilitate perceptual and conceptual processes in instruction to overcome learning challenges. The study provides valuable insights for the IS educators, the UML academic community, and practitioners.

Keywords: UML, Diagrammatic Representation, Diagrammatic Reasoning, Learning Challenges, Concept Mapping.

This manuscript underwent peer review. It was received 10/04/2017 and was with the authors for 3 months for 2 revisions. The Associate Editor chose to remain anonymous.

1 Introduction

System analysis and design (SA&D) is a core component of the information systems (IS) curriculum (Topi et al., 2010, 2017). As the SA&D field has changed from the structured to object-oriented (OO) methodology (Nerur, Slinkman, & Mahapatra, 2005), OO methodology has become an integral part of IS education (Satzinger, Batra, & Topi, 2007). Of the various OO techniques, unified modeling language (UML) has emerged as the dominant modeling approach and become an essential part of any IS professional's toolset. To maintain their relevance, IS programs recognize the need to teach OO methodology. Many of them offer courses on UML or incorporate UML into their existing SA&D courses.

In an effort to understand UML in the context of IS education, we report on an empirical study on learning UML. Specifically, we use a cognitive technique to tap into students' challenges in learning UML notational elements. We interpret the findings through the lens of diagrammatic representation and diagrammatic reasoning and offer insights into how to enhance teaching UML.

The paper proceeds as follows: in Section 2, we discuss diagrammatic representation and diagrammatic reasoning that learning a visual language such as UML involves. In Section 3, we describe the cognitive method we used in our empirical investigation. In Section 4, we present our empirical study and detail our research procedures. In Section 5, we present our findings, which we discuss in Section 6. Finally, in Section 7, we discuss the study's contributions and limitations and conclude the paper.

2 Theoretical Background

As a visual language, UML uses diagrams to represent information on a two-dimensional (2D) surface to visualize, specify, construct, and document systems (Object Management Group, 2015). Diagrams, therefore, play a critical and essential role for UML. In this section, we describe diagrammatic representation and diagrammatic reasoning, the two integral areas for effective diagrams.

2.1 Diagrammatic Representation

Diagrammatic representation concerns how to represent information in diagrams. At its core, diagrammatic representation has visual notations, also known as graphical notations and diagramming notations, that comprise a set of graphical symbols (visual vocabulary), a set of compositional rules (visual grammar), and the meaning of each symbol (visual semantics) (Moody, 2009). Graphical symbols make up the visual vocabulary that represents information perceptually in diagrams. The vocabulary of a visual language should be large enough to cover all the concepts to be communicated, and its users should have the same understanding of it.

Computational rules represent the visual grammar that defines the relationships between various graphical symbols to make them meaningful. A diagram will lose its informative value if it does not follow computational rules. Different visual languages have different visual grammars. The visual vocabulary and visual grammar together form the visual syntax of a visual language. Visual syntax and visual semantics work together to deliver and make sense of the visual information embedded in diagrams.

2.2 Diagrammatic Reasoning

Diagrammatic reasoning studies how human beings search, recognize, and infer visual information in diagrams. Different visual notations may represent the same information, but different representations for the same information may not be equally effective for diagrammatic reasoning. Empirical studies show that diagrams are cognitively effective only when they facilitate both perceptual and conceptual processes involved in diagrammatic reasoning (Kim, Hahn, & Hahn, 2000; Hahn & Kim, 1999).

In perceptual processes (seeing), human beings sense the things in the outside world and recognize relevant information. Automatic, very fast, and mostly executed in parallel, perceptual processes produce automatic and intuitive reactions to directly observable features of the items such as shape and direction. In conceptual processes (understanding), human beings infer and derive new information from the information obtained in perceptual processes. Effortful, relatively slow, and mostly executed in sequence, conceptual processes require attention, focus, deliberation, and analysis to form abstractions of the things in the outside world.

To enable effective perceptual processing, visual notations need to have certain features such as perceptual discriminability and semantic transparency. Perceptual discriminability refers to how easily and accurately individuals can visually differentiate visual notations from each other (Moody, 2009). The visual distinction between different visual notations enables individuals to more quickly search for, recognize, and recall information and, therefore, can reduce cognitive load in perceptual processes. To maximize discriminability, individuals should be able to automatically and in parallel detect the differences of graphical symbols. Individuals need to accurately visually discriminate between visual notations to accurately interpret diagrams.

Semantic transparency stipulates that a visual notation's meaning (semantic) should be clear (transparent) from its appearance alone (Caire, Genon, Heymans, & Moody, 2013). While perceptual discriminability requires different visual notations to have different appearances, semantic transparency demands the appearance of a visual notation to provide cues to its meaning. The more natural and intuitive the association between a visual notation's appearance and meaning is, the more semantically transparent the visual notation. Semantically transparent visual notations reduce cognitive load because individuals can directly perceive or easily learn their meanings (e.g., a trash can icon to represent deleted items or the color red to suggest danger, respectively) (Petre, 1995).

To support effective conceptual processing, visual notations need to have such properties as construct clarity and manageable complexity to facilitate inductive inference. Construct clarity refers to one to one correspondence between the graphical symbols and their referent concepts (Goodman, 1968). When a one-to-one correspondence between the constructs and symbols does not exist, ontological discrepancies occur (Wand & Weber, 1993). Four possible ontological discrepancies include symbol deficiency, overload, redundancy, and excess (see Table 1). These ontological discrepancies add cognitive load and slow down conceptual processing of diagrammatic information. For example, symbol redundancy places a burden on choosing which symbol to use and to remember multiple representations of the same construct. Symbol overload leads to ambiguity and the potential for misinterpretation. Symbol excess adds visual clutter to diagrams and confounds the interpretation of diagrams. In contrast, construct clarity maximizes precision, expressiveness, and parsimony and, therefore, is highly desirable.

Table 1. Ontological Discrepancies

	Matching between	
	Semantic construct(s)	**Graphical symbol(s)**
Symbol deficiency	One or Many	None
Symbol overload	Many	One
Symbol redundancy	One	Many
Symbol excess	None	One or Many

Individuals can comprehend only a limited number of visual notations at a time. When they exceed this cognitive limit, they experience cognitive overload and their comprehension degrades rapidly. Visual notations should be able to represent information without overloading the human mind. To optimize conceptual processing, a visual language like UML should have some mechanisms to manage graphical complexity. One such mechanism is graphical complexity, which the number of graphical symbols and their various combinations in a visual language measures (Nordbotten & Crosby, 1999); that is, the size of the visual vocabulary. Ideally, the visual vocabulary should be adequate enough to cover necessary constructs but succinct enough to prevent cognitive overload.

3 Methodology

Cognitive mapping extracts statements from individuals about subjectively meaningful concepts and relations in particular areas of interest and then describes these concepts and relations in diagrammatical layout (Swan, 1997). Researchers have widely used it in many areas such as strategic management (Johnson & Lipp, 2007), project management (Robertson & Williams, 2006), and education (Ihlenfeldt, 1981). It has also proven valuable for the SA&D field in which researchers have used it to complement and augment the methods for information requirements analysis (Montazemi & Conrath, 1986) and to understand software operation support expertise (Nelson, Nadkarni, Narayanan, & Ghods, 2000). Because

it taps into human cognitive processes and can discover latent characteristics, we used the cognitive mapping method to solicit students' perceived difficulties in learning UML.

Commonly used cognitive mapping approaches include causal mapping, mind mapping, and concept mapping (Siau & Tan, 2005). We did not know the challenges in learning UML, so we did not have the "central idea" that one needs for mind mapping (Buzan, 1993). While causal mapping represents the antecedent-consequence relation between two concepts (Eden, Ackermann, & Cropper, 1992), other types of relationships among concepts also exist, such as association. We used the concept-mapping approach, which shows concepts and relationships between concepts in a graphical representation called a concept map (Novak, 1993). Concept mapping helps one to generate ideas, communicate complex ideas, and assess understanding and diagnose misunderstanding (Jonassen, Beissner, & Yacci, 1993). As such, it can reveal the challenges in learning UML and the relationships among them.

We used the group concept-mapping technique that Trochim (1989), Kane and Trochim (2007, and Trochim and McLinden (2017) have developed. We chose this technique because researchers have considerably used it in many fields such as social services (Galvin, 1989) and healthcare (Valentine, 1989), and produces reliable results according to generally recognized standard for acceptable reliability levels (Trochim, 1993). Also, it provides clear procedures for collecting data and allows one to systematically statistically analyze it. We conducted our study to identify challenges in learning UML from the student perspective. We then analyzed the student-perceived learning challenges through the lens of diagrammatic representation and reasoning and identified instructional strategies to facilitate both perceptual and conceptual processes involved in the challenges in learning UML.

4 Procedures

We conducted our concept-mapping study in six steps as Trochim (1989), Kane and Trochim (2007), and Trochim and McLinden (2017) specify. In this section, we describe how we performed each step in detail.

4.1 Preparation

In the first step of a concept-mapping project, one needs to select participants and develop the focus statement based on the research question. Senior and graduate students who had completed a design object-oriented system course at a large public university in the United States served as our study participants. The university's college of business administration offered the course to students who majored or minored in MIS. The lecture-based course focused on basic concepts and processes in designing OO systems with UML. Taught in a traditional classroom setting, the course introduced UML's fundamental semantics, syntax, and diagramming techniques. In the 16-week course, students did in-class exercises, quizzes, and exams and finished a semester-long SA&D project using UML. Thirty-two students participated in our study.

Given UML's complexity (with its multiple diagrams and numerous modeling techniques), the course exposed students only to UML's fundamentals. Thus, instead of seeking students' input on difficulties in learning UML in general, we focused on their challenges in learning UML notational elements (i.e., the building block of UML diagrams). UML notational elements refer to UML graphical symbols and their combinations (Booch, Rumbaugh, & Jacobson, 2005). UML has four kinds of graphical symbols: icons, 2D symbols, paths, and strings.

- An icon refers to a graphical figure of a fixed size and shape, and it does not expand to hold contents. The diamond shape exemplifies an icon.
- A 2D symbol has variable height and width and can expand to hold other things, such as a list of strings or other symbols. Rectangles, ovals, and circles exemplify a 2D symbol.
- A path refers to a sequence of line segments that have attached endpoints. The lines that attach other graphical symbols at both ends exemplify a path.
- A string basically refers to a character sequence that may use some suitable character sets and may exist as a singular element of a symbol or a component of a symbol, such as an element in a list, a label attached to a symbol or path, or as a standalone element in a diagram. The name of a class exemplifies a string.

One can use the four kinds of graphical symbols alone and in combination to form notational elements, which, in turn, one can map to construct various diagrams. Figure 1 illustrates UML graphical symbols in a class diagram.

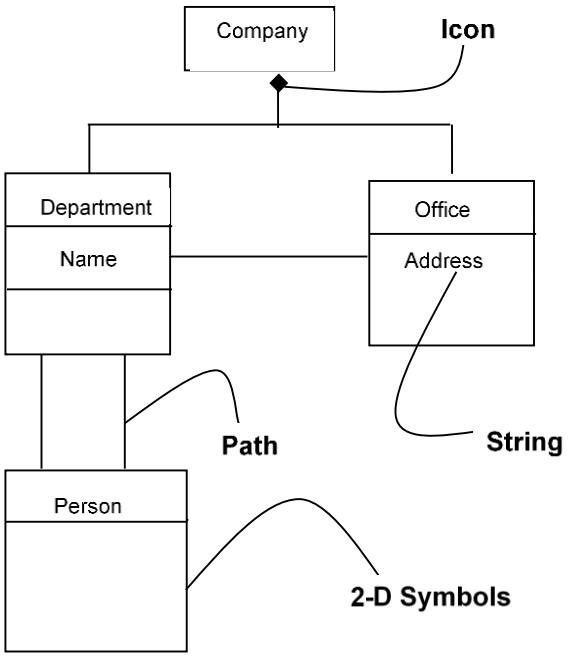

Figure 1. Illustration of UML Graphical Symbols

We were interested in the challenges that students encountered in learning UML notational elements. Accordingly, we developed a focus statement operationalized as: generate statements (short phrases or sentences) that describes specific problems, difficulties, and/or concerns in drawing and interpreting UML notational elements.

4.2 Generating Statements

In this step, participants brainstorm to generate as many statements relevant to the focus statements as possible. We showed the focus statement to the participants, and they worked individually in generating statements. They generated 112 statements in total. We then examined the 112 statements to eliminate irrelevancies, redundancies, and inaccuracies. For example, statements such as “The software used to draw UML is not very user friendly” concerned the usability of the supporting tools and did not pertain to our focus statement. We considered statements like these to be irrelevant and removed them.

We combined statements that addressed the same idea. For example, statements such as “For the generalization, the direct style and the tree style are almost the same” and “Direct style and tree style depicted for super- and sub-classes could cause drawing difficulty” both expressed the concern that the concept of generalization has two graphical styles. We collapsed these statements into one statement: “In generalization, the two styles (i.e., the direct and tree style) should not be allowed. Having only one style would make it easier.”. In addition, we checked the accuracy and validity of each statement. For example, we removed statements such as “Distinction between squares and rectangles—all seem to be rectangles” because the statement itself was incorrect—UML diagrams do not use squares at all.

As a result, 39 statements remained. A UML expert who served as an instructor of the class and was not associated with this study evaluated the statement list. He validated that each statement represented one unique learning challenge that pertained to learning UML notational elements and that each one read well/clearly. We used these 39 statements (see the Appendix) for further analysis.

4.3 Structuring Statements

In this step, participants sort the brainstormed statements and rate each statement according to a Likert-type scale. In our study, we first showed the 39 statements in a randomized order to every participant. We asked them to go through the statements and answered their questions (if they had any) about the statements before they sorted and rated each statement individually. We then instructed the participants to sort the statements into piles in a way that made sense to them following three sorting rules: 1) they could place each statement in only one pile, 2) they could not place all statements into one single pile, and 3) each pile needed to have more than one statement. After sorting statements into piles, the participants rated each statement on a five-point Likert type scale (1: relatively less important, 2: somewhat important, 3: moderately important, 4: very important, and 5: extremely important) in terms of each one's relative importance to learning UML notational elements.

4.4 Concept-mapping Analysis

In this step, one conducts several analyses to represent the statements and construct four concept maps (i.e., the point map, the cluster map, the point rating map, and the cluster rating map). In our study, we first constructed two types of similarity matrices: an individual similarity matrix for each participant and a total similarity matrix. In constructing the individual similarity matrix, for any two statements i and j, we placed a "1" in the individual similarity matrix if a sorter placed the two statements in the same pile; otherwise, we entered a "0". We obtained the total similarity matrix by summing the individual matrices. Any cell in the total similarity matrix could assume an integer value between 0 and 32 (the total number of sorters), and the value indicates the number of sorters who placed the i and j pair in the same pile. Figure 2 shows the total similarity matrix.

State-ment	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	37	38	39
1	32	14	4	19	3	13	16	8	9	8	9	13	7	7	5	17	9	2	13	16	7	14	3	16	8	9	4	16	7	5	4	12	24	5	4	10	17	4	7
2	14	32	5	22	4	14	10	11	6	11	14	11	12	7	6	8	17	5	8	8	11	10	5	8	13	18	5	10	1	0	4	9	11	3	6	8	12	4	14
3	4	5	32	3	30	4	2	2	2	13	2	5	1	3	3	3	2	10	1	0	1	1	9	1	1	0	30	1	2	2	0	10	3	2	3	2	1	1	1
4	19	22	3	32	2	18	14	9	5	7	10	17	9	9	5	12	13	2	14	14	8	14	4	14	10	14	3	11	3	2	4	7	16	4	3	10	13	2	11
5	3	4	30	2	32	5	2	1	2	12	0	5	0	4	3	2	0	12	0	0	0	1	8	2	0	0	29	1	2	3	0	9	3	3	4	3	1	2	0
6	13	14	4	18	5	32	8	5	7	7	6	11	5	11	3	8	7	7	7	8	4	7	6	9	6	8	3	7	5	4	4	8	11	4	6	7	8	2	6
7	16	10	2	14	2	8	32	7	3	4	8	10	8	4	4	11	9	5	20	18	7	25	4	15	11	10	3	12	2	0	15	9	19	14	2	20	13	5	9
8	8	11	2	9	1	5	7	32	0	4	26	11	27	1	5	12	21	2	13	11	28	7	2	10	22	20	0	14	1	1	2	2	7	2	2	8	9	3	22
9	9	6	2	5	2	7	3	0	32	8	2	6	2	9	7	5	2	4	4	6	1	7	5	5	3	3	4	6	7	7	7	5	7	2	9	4	10	9	2
10	8	11	13	7	12	7	4	4	8	32	5	6	3	8	8	5	3	10	4	3	3	5	9	4	2	3	13	4	4	2	3	14	4	3	9	5	10	1	3
11	9	14	2	10	0	6	8	26	2	5	32	7	27	1	6	8	24	3	10	8	28	7	3	9	24	21	2	11	0	0	2	5	9	1	3	7	10	2	23
12	13	11	5	17	5	11	10	11	6	6	7	32	9	10	7	14	5	1	15	14	9	11	2	12	6	7	4	15	7	6	8	7	11	6	3	10	13	6	7
13	7	12	1	9	0	5	8	27	2	3	27	9	32	2	6	10	23	3	12	10	28	7	2	9	26	22	1	13	0	0	3	3	6	1	3	9	10	2	24
15	7	7	3	9	4	11	4	1	9	8	1	10	2	32	10	7	2	5	7	9	1	5	4	9	2	2	2	4	7	8	6	10	7	8	10	7	6	6	2
14	5	6	3	5	3	3	4	5	7	8	6	7	6	10	32	6	5	6	8	7	5	6	6	9	4	4	2	6	14	16	13	9	6	11	4	6	6	13	5
16	17	8	3	12	2	8	11	12	5	5	8	14	10	7	6	32	6	3	20	18	10	10	2	19	7	6	1	16	11	12	3	11	14	3	4	8	12	5	9
17	9	17	2	13	0	7	9	21	2	3	24	5	23	2	5	6	32	2	9	7	23	8	3	8	27	28	1	9	0	0	2	4	10	2	3	7	8	3	28
18	2	5	10	2	12	7	5	2	4	10	3	1	3	5	6	3	2	32	6	5	3	6	24	7	2	2	10	1	2	4	6	10	2	6	4	10	4	5	2
19	13	8	1	14	0	7	20	13	4	4	10	15	12	7	8	20	9	6	32	25	13	20	7	22	9	9	0	16	6	6	12	8	15	12	3	18	13	6	10
20	16	8	0	14	0	8	18	11	6	3	8	14	10	9	7	18	7	5	25	32	10	19	5	25	8	8	0	15	8	8	11	11	18	9	5	15	14	6	9
21	7	11	1	8	0	4	7	28	1	3	28	9	28	1	5	10	23	3	13	10	32	7	3	9	23	22	0	14	0	0	3	3	7	1	3	9	10	2	23
22	14	10	1	14	1	7	25	7	7	5	7	11	7	5	6	10	8	6	20	19	7	32	7	16	9	9	1	10	4	3	17	7	16	12	3	18	13	6	9
23	3	5	9	4	8	6	4	2	5	9	3	2	2	4	6	2	3	24	7	5	3	7	32	6	1	3	10	4	4	3	6	7	1	4	3	7	3	4	1
24	16	8	1	14	2	9	15	10	5	4	9	12	9	9	9	19	8	7	22	25	9	16	6	32	9	9	1	13	10	11	8	12	16	8	5	13	14	5	10
25	8	13	1	10	0	6	11	22	3	2	24	6	26	2	4	7	27	2	9	8	23	9	1	9	32	25	1	9	0	0	3	3	9	2	3	8	10	3	28
26	9	18	0	14	0	8	10	20	3	3	21	7	22	2	4	6	28	2	9	8	22	9	3	9	25	32	1	11	1	0	3	3	9	2	2	8	11	3	26
27	4	5	30	3	29	3	3	0	4	13	2	4	1	2	2	1	1	10	0	0	0	1	10	1	1	1	32	2	3	2	0	10	3	2	3	2	2	1	0
28	16	10	1	11	1	7	12	14	6	4	11	15	13	4	6	16	9	1	16	15	14	10	4	13	9	11	2	32	7	5	6	7	15	7	4	11	14	7	10
29	7	1	2	3	2	5	2	1	7	4	0	7	0	7	14	11	0	2	6	8	0	4	4	10	0	1	3	7	32	26	15	8	5	12	5	3	5	15	1
30	5	0	2	2	3	4	0	1	7	2	0	6	0	8	16	12	0	4	6	8	0	3	3	11	0	0	2	5	26	32	14	8	4	12	5	3	2	16	1
31	4	4	0	4	0	4	15	2	7	3	2	8	3	6	13	3	2	6	12	11	3	17	6	8	3	3	0	6	15	14	32	4	5	20	6	16	8	16	4
32	12	9	10	7	9	8	9	2	5	14	5	7	3	10	9	11	4	10	8	11	3	7	7	12	3	3	10	7	8	8	4	32	11	6	5	8	8	4	3
33	24	11	3	16	3	11	19	7	7	4	9	11	6	7	6	14	10	2	15	18	7	16	1	16	9	9	3	15	5	4	5	11	32	7	5	11	15	5	9
34	5	3	2	4	3	4	14	2	2	3	1	6	1	8	11	3	2	6	12	9	1	12	4	8	2	2	2	7	12	12	20	6	7	32	5	18	4	17	1
35	4	6	3	3	4	6	2	2	9	9	3	3	3	10	4	4	3	4	3	5	3	3	3	5	3	2	3	4	5	5	6	5	5	5	32	8	6	7	4
36	10	8	2	10	3	7	20	8	4	5	7	10	9	7	6	8	7	10	18	15	9	18	7	13	8	8	2	11	3	3	16	8	11	18	8	32	10	8	7
37	17	12	1	13	1	8	13	9	10	10	10	13	10	6	6	12	8	4	13	14	10	13	3	14	10	11	2	14	5	2	8	8	15	4	6	10	32	5	12
38	4	4	1	2	2	2	5	3	9	1	2	6	2	6	13	5	3	5	6	6	2	6	4	5	3	3	1	7	15	16	16	4	5	17	7	8	5	32	3
39	7	14	1	11	0	6	9	22	2	3	23	7	24	2	5	9	28	2	10	9	23	9	1	10	28	26	0	10	1	1	4	3	9	1	4	7	12	3	32

Figure 2. Total Similarity Matrix

Next, we conducted a multi-dimensional scaling (MDS) analysis on the total similarity matrix in SPSS. This SPSS analysis yielded the point map (i.e., the 2D configuration of the 39 statements; see Figure 3). In the point map, a point represented a statement. SPSS configured the relationships among the statements using the criterion that statements that piled together should appear closer in 2D space and that those that piled together less frequently should appear further apart.

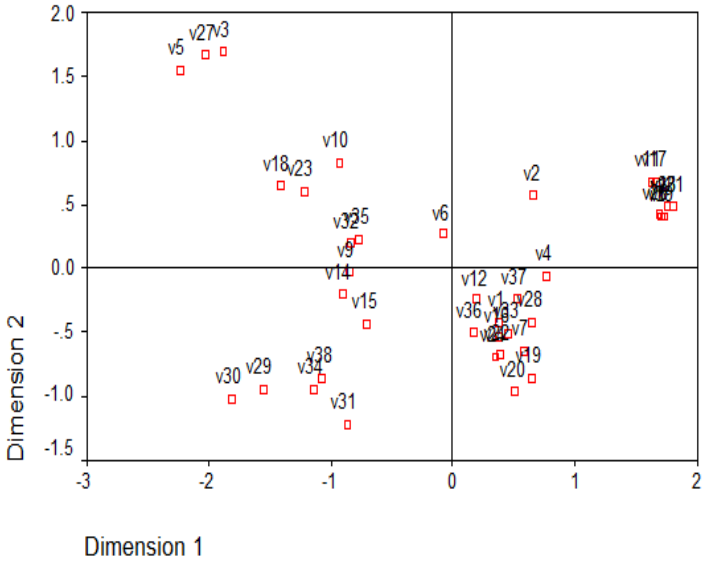

Figure 3. SPSS 2D Configuration (Point Map)

We then conducted a hierarchical cluster analysis to group individual statements on the point map into clusters of statements (the cluster map). The key operative rule for a cluster analysis is to maintain the integrity of the multidimensional scaling and partition the MDS configuration into non-overlapping clusters in a 2D space (Trochim, 1989; Kane & Trochim, 2007; Trochim & McLinden, 2017). We used Ward's algorithm because it generally gives more sensible and interpretable solutions than other approaches (e.g., single linkage, centroid) (Kane & Trochim, 2007). The algorithm begins by considering each statement to be its own cluster. It then combines two clusters until it has combined them all into a single cluster at the end. Figure 4 shows the dendrogram graph that the SPSS hierarchical cluster analysis generated.

Subsequently, we had to decide how many clusters we should group the statements into for the final solution. The point map (see Figure 3) suggests six clusters. First, a cluster with statements 3, 5, and 27 appeared in the upper-left corner of the point map. The dendrogram analysis merged these three statements in the first combination round. A cluster with statements 8, 11, 13, 17, 21, 25, 26, and 39 appeared on the very right side of the point map. The dendrogram analysis merged these eight statements in the second combination round. Based on the convergence of the point map and the dendrogram graph, we classified statements 3, 5, and 27 into one cluster and statements 8, 11, 13, 17, 21, 25, 26, and 39 into another.

Statements 1, 4, 7, 12, 16, 19, 20, 22, 24, 28, 33, 36, and 37 grouped together at the center of the point map. The dendrogram analysis combined these 13 statements, along with statements 2 and 6, into one cluster after four combination rounds. From carefully reading all the 15 statements, we found that statement 2 regarding the stereotype fit well with statements 1, 4, 7, 12, 16, 19, 20, 22, 24, 28, 33, 36, and 37 because they all pertained to 2D symbols, while statement 6 regarding actor notation (i.e., an icon) did not. So we decided to include statement 2 in and exclude statement 6 from the cluster that had statements 1, 4, 7, 12, 16, 19, 20, 22, 24, 28, 33, 36, and 37.

Statements 9, 30, 31, 34, and 38 appeared quite closely together in the left lower part in the point map. The dendrogram analysis merged these five statements, along with statement 15, into one cluster after three combination rounds. Statement 15, however, appeared in the point map closer to the block with the statements 9, 14, 32, and 35. Considering that statement 15 described using two notations for the interface concept, we classified it with statements 29, 30, 31, 34, and 38 because they all dealt with the issue of symbol redundancy.

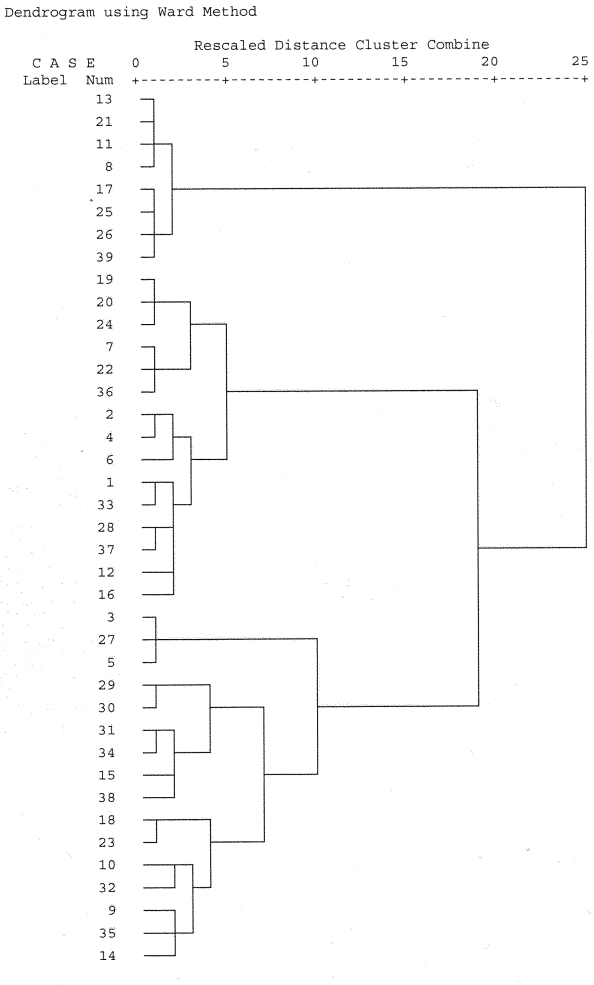

Figure 4. SPSS Dendrogram Graph in Cluster Analysis

The point map seems to suggest two clusters for the remaining statements: one with the statements 10, 18, and 23 and the other with the statements 9, 14, 32, and 35. Statements 18 and 23 both concerned the multiplicity concept in the class diagram and statement 10 concerned the lifeline concept in the sequence diagram. The dendrogram analysis did not merge these three statements (i.e., 18, 23, and 10) until it combined them into one cluster with statements 9, 14, 32, and 35 after four combination rounds, though they appeared together in the point map. Similarly, the dendrogram analysis did not merge statements 9, 14, 32, and 35 until it combined them into one cluster with statement 10, but they appeared close to each other in the point map. So we decided to group these seven statements into one cluster. We also put statement 6 into this cluster for two reasons: 1) because it appeared more closely to this cluster than to any other clusters in the point map and 2) because, like the other seven statements in this cluster, it raised one issue that did not belong to any other clusters.

In sum, based on the dendrogram graph, the point map, and on the sensemaking criterion, we came up with five clusters. Figure 5 presents the cluster map that shows how we grouped the 39 statements.

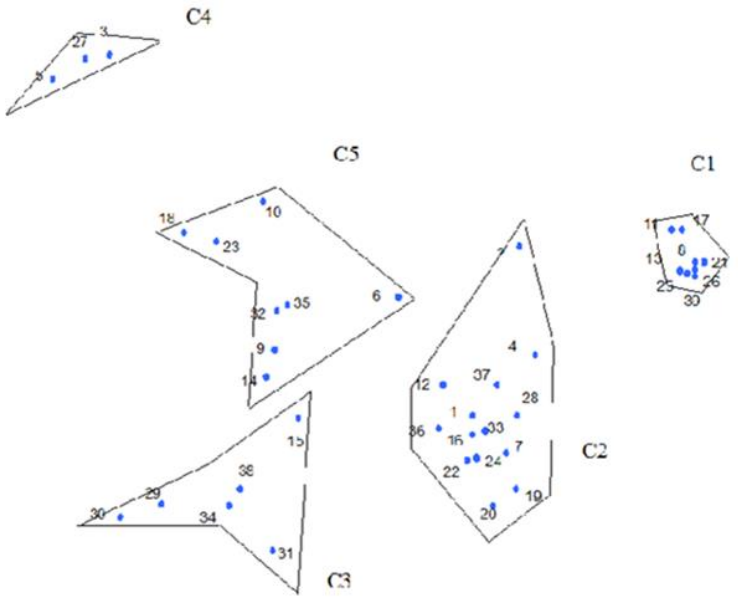

Figure 5. Cluster Map

Finally, we calculated the average rating for each statement (see the Appendix) and added each one to the point map to produce the point rating map (Figure 6). We calculated the rating for each cluster by averaging the sum of the average ratings of the statements in the cluster. We then added the average rating for each cluster to the cluster map to produce the cluster rating map (Figure 7).

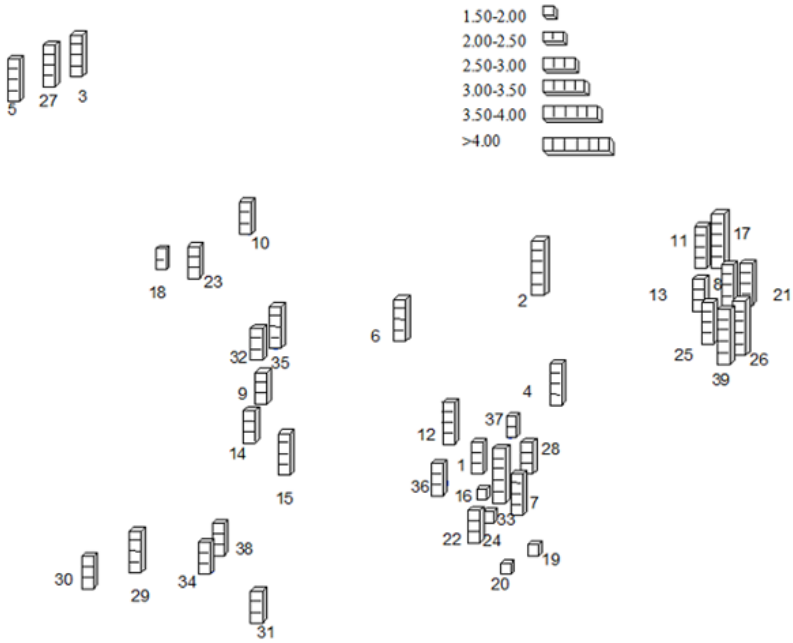

Figure 6. Point Rating Map

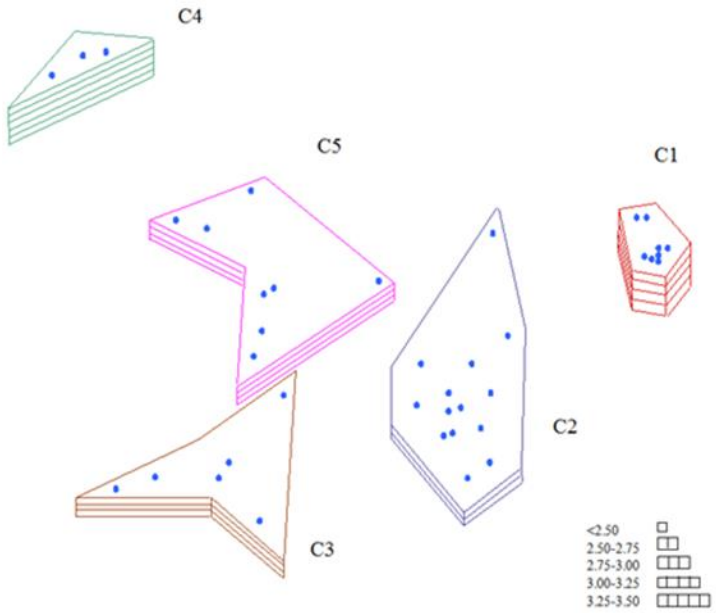

Figure 7. Cluster Rating Map

4.5 Interpreting Concept Maps

In this step, participants validate and name the clusters. In our study, we showed each participant individually the clusters and asked each one to come up with a phrase or short sentence to describe each cluster. Based on the names that the participants provided, we finalized the name for each cluster. The UML expert who independently validated the statement list in the second step (see Section 4.2) went through the cluster list and analyzed the cluster names that the participants provided. He agreed that the cluster names were valid. Table 2 summarizes the cluster information.

Table 2. Summary of Clusters

Cluster and name	Statements in cluster	Cluster rating	Statement rating range
1) Similarity in notational elements	Eight statements: 8, 11, 13, 17, 21, 25, 26, and 39	3.24	2.58 – 3.53
2) Ambiguity in notational elements	14 statements: 1, 2, 4, 7, 12, 16, 19, 20, 22, 24, 28, 33, 36, and 37	2.66	1.97 – 3.53
3) Inconsistency in notational elements	Six statements: 15, 29, 30, 31, 34, and 38	2.93	2.81 – 3.22
4) Confusing numbering in communication diagram	Three statements: 3, 5, and 7	3.30	3.28 – 3.31
5) Other problems	Eight statements: 6, 9, 10, 14, 18, 23, 32, and 35	2.77	2.44 – 3.41

4.6 Utilizing Concept Maps

In this step, one interprets and explains the concept maps using relevant theories and frameworks. In the next two sections, we present the findings of our study and discuss how we interpret and use the concept maps in the light of diagrammatic representation and diagrammatic reasoning, respectively.

5 Findings

The concept maps we constructed identify five clusters of challenges in learning UML notational elements. In this section, we report the learning challenges in each cluster.

5.1 Cluster 1: Similarity in Notational Elements

This cluster indicates that some UML notational elements do not differ enough from one another (i.e., they look too similar to each other at a glance). The eight statements in this cluster concerned lines (e.g., dashed and solid lines), arrowheads (e.g., stick, empty, and filled arrowheads), and various combinations of different lines and arrowhead styles (see Figure 8 for illustration). Such minute differences impose perceptual challenges in distinguishing between graphical symbols and in seeing and linking relevant visual items, which lowers the perceptual discriminability for accurate visual discrimination in perceptual processes.

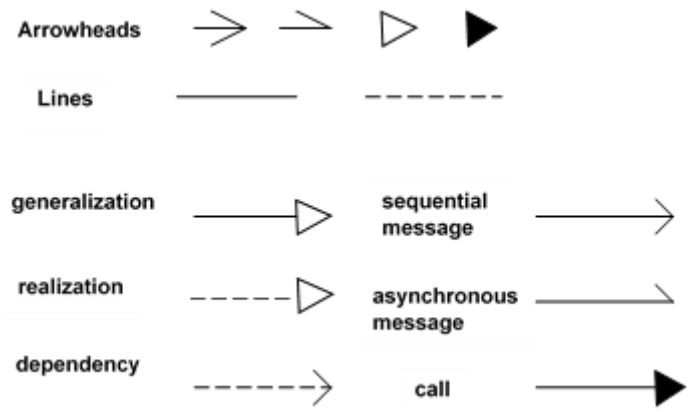

Figure 8. Examples of Similarity in Notational Elements

The cluster had a cluster rating of 3.24, which indicates that low perceptual discriminability represents an important learning challenge for students. Seven statements (39, 17, 26, 25, 21, 8, and 11) had high ratings over 3, and one statement (13) had a rating of 2.58. Statement 13 concerned the differentiation between dependency and realization—the two specialized relationships. We can attribute statement 13's lower rating to the practice that the name of the dependency relation goes with the dashed line and stick arrowhead to represent a dependency relationship. Using a string along with 2D symbols helps differentiate dependency from realization. Overall, the participants found the subtle differences in notations for different semantic concepts confusing.

5.2 Cluster 2: Ambiguity in Notational Elements

This cluster identifies learning challenges related to using one 2D symbol to represent two or more semantic concepts. For example, rectangles represent classes in the class diagram and objects in the object diagram. Rectangles also describe different concepts in the same diagram. For example, rounded rectangles represent activity states and action states in the activity diagram. Ovals describe base use cases, extend use cases, and include use cases in the use case diagram. Using one graphical symbol to represent two or more semantic constructs shows the ontological discrepancy of symbol overload (see Table 1). Such a lack of construct clarity slows down conceptual processing of visual information and, thus, contributes to learning challenges.

The cluster had 14 statements—the largest of all clusters. It also had the lowest cluster rating of 2.66. However, this cluster did have statements with ratings over 3. For example, statements 33, 2, and 7 had ratings of 3.53, 3.50, and 3.34, respectively. A close look reveals the statements' ratings ranged widely from 1.69 to 3.53. Further examination shows that statements with ratings higher than 3 (i.e., statements 33, 2, 7, and 4) concerned the use of same or similar notations to represent semantically related notions (e.g., excluding and included stereotypes in statement 2; classes and objects in statement 7), whereas statements with a rating below 2 (i.e., statements 16, 19, 20, and 24) concerned semantically distinct notions (e.g., components and nodes in statement 16; notes and classes in statement 19). In other words, symbol ambiguity poses significantly more challenges when it involves semantic constructs that individuals cannot easily distinguish from one another.

5.3 Cluster 3: Inconsistency in Notational Elements

The six statements in this cluster show that more than one graphical symbol represent some concepts. For example, icons such as the computer and cube both represent a server. Some 2D symbols such as rectangle and circle both represent an interface. Two styles can represent generalization: direct and tree. Using more than one graphical symbol to represent one semantic construct shows the ontological discrepancy of symbol redundancy. The cluster's cluster rating of 2.93 indicates the symbol-redundancy issue is not that serious. Nevertheless, the cluster did contain statements with a rating over 3 (e.g., 3.22 for statement 29). Statements 29, 15, and 31 had ratings above the cluster rating and concerned components, interfaces, and active classes, respectively. The other three statements in this cluster had lower ratings and concerned servers (statement 30), classes (statement 34), and generalization (statement 38). The concepts involved in higher rating statements were more abstract by definition and less used in the UML course than those in lower rating statements.

5.4 Cluster 4: Confusing Numbering in Communication Diagrams

With only three statements, this cluster was the smallest one. However, it had the greatest importance with the highest cluster rating of 3.30. In fact, all three statements in this cluster had a rating above 3. They represent the challenges with communication diagrams from different perspectives. Statement 3 concerned the difficulty in following the numbering, statement 5 concerned the difficulty in differentiating sequence number from other similar notations, and statement 27 concerned the difficulty in retrieving information from the sequence number. All three statements point out that discrimination that relies on only textual characteristics (i.e. strings), sometimes in combination with contexts, makes it extremely difficult to automatically and in parallel detect the differences between notational elements. That is, this cluster reveals the issue of low perceptual discriminability due to the extensive use of strings in communication diagrams.

5.5 Cluster 5: Other Problems

This cluster included the problems, difficulties, or concerns that the other clusters did not cover. The eight statements concerned strings such as multiplicity (statements 18 and 23), icons such as stickman (statement 6), and 2D symbols such as object lifelines (statement 10) and circles (statement 14). The cluster's cluster rating of 2.77 indicates these concerns had moderate importance. Statements 35 and 6 had ratings above 3. Statement 35, with a rating of 3.41, the highest in this cluster, interestingly observes the lack of linkage between the various diagrams, which highlights the issue with complexity management. Statement 6 shows that using a stick man to represent a non-human actor is counterintuitive, which relates to the lack of semantic transparency.

6 Discussion

In this section, we explain what our findings explain about why students have challenges in learning UML notational elements and discuss how to help overcome these challenges in teaching.

6.1 Learning Challenges

UML diagrammatic representations should facilitate diagrammatic reasoning. They need to be easy to recognize and understand and support accurate model interpretation in a timely manner. We identified 39 challenges in learning UML. These learning challenges show how the issues with diagrammatic representation of UML notational elements hinder perceptual and conceptual processes for effective diagrammatic reasoning, which Table 3 summarizes.

The first cluster shows that UML notational elements with low perceptual discriminability impede students' ability to readily recognize graphical symbols and see and link relevant visual items. Unlike experts who can make much finer distinctions with experience (Britton & Jones, 1999), students (novices) cannot quickly and accurately differentiate UML overly similar notational elements. Because such individuals need to make a more conscious effort and to memorize elements, perceptual processes slow down and learning challenges arise.

The fourth cluster shows similar perceptual challenges in that students found textual coding with strings difficult to understand. Though more expressive semantically, textual encoding is less semantically transparent and, therefore, less perceptually effective than graphical encoding (Nelson et al., 2000).

Communication diagram's sole reliance on strings for discrimination makes it difficult for novices to instantly detect the difference. Instead, they need to perform serial searches to process perceptual cues in the textual format. Because they need to expend more time and effort on searching for and inferring information, students feel that it is difficult to learn message passing in the communication diagram.

Table 3. Summary of Diagrammatic Representation Issues

Issues in diagrammatic representation	Hindrance to diagrammatic reasoning
• Notational similarity (cluster 1)	• Low perceptual discriminability slows down perceptual processing.
• Notational ambiguity (cluster 2)	• Ontological discrepancy (i.e., symbol overload) impedes conceptual processing.
• Notational inconsistency (cluster 3)	• Ontological discrepancy (i.e., symbol redundancy) impedes conceptual processing.
• Reliance on strings for message passing in Communication Diagram (cluster 4)	• Lack of semantic transparency slows down perceptual processing.
• Problems with stickman, multiplicity, and lack of linkage among diagrams (cluster 5)	• Lack of semantic transparency (e.g., the use of stickman to represent non-human actor) slows down perceptual processing. • Lack of complexity-management mechanism impedes conceptual processing.

The lack of semantic transparency in perceptual processes also creates learning barriers, which some statements in the fifth cluster indicate. Statement 6, for example, points out that a stickman (an icon) represents non-human actors in practice. Using a stick figure for a non-human actor is counterintuitive because it breaks the natural correspondence between a stick figure and a person. Such usage may not be problematic to seasoned users who have experience with this type of representation. But, for untrained eyes, it adds cognitive load in inferring the meaning of non-human actor from the appearance of a stick figure alone.

Some technical properties of UML notational elements also impede conceptual processes. UML notational elements lack construct clarity since we found ontological discrepancies in the second and third clusters. Symbol overload, which we found in the second cluster, increases ambiguity, and symbol redundancy, which we found in the third cluster, adds to the complexity of UML. Ambiguous and complex graphical symbols affect novices more than experts who have developed the long-term memory that automates diagrammatic interpretation (Blackenship & Dansereau, 2000). Research has also shown that novices have to more consciously remember symbols' meaning (Winn, 1993). Because novices need to more significantly pay attention to and deliberate on choosing which symbol to use, ontological discrepancies negatively affect the speed, ease, and accuracy in conceptual processes, which becomes an obstacle to learning.

In addition, the fifth cluster shows that the lack of complexity management challenges cognitive processes. For example, statement 35 indicates that UML lacks an explicit mechanism to simplify navigation and transition between diagrams (i.e., perceptual integration) and to help assemble information from separate diagrams into a coherent mental representation of the system (i.e., conceptual integration). This shortcoming places additional cognitive demands on individuals—especially novices who lack the training and experience in searching for, recognizing, and inferring information from various diagrams and in assembling such information to form a coherent mental representation.

In short, our group concept-mapping study shows that both the technical properties of UML notational elements and the cognitive attributes of students obstruct one from learning UML. The concept maps also show that learning challenges are not equally distanced to each other. The second, third, and fifth clusters were very close to each other at the center of the cluster map, while the first and fourth were spread out (i.e., one on the far right side and the other on the far left corner of the cluster map). Carefully examining the statements in the five clusters shows that the second, third, and fifth dealt with concepts such as class, object, action, activity, use case, and interfaces and that the first and fourth focused more on relationships and interactions between things. That is, we can separate the learning challenges into two categories: one related to static aspects (i.e., structures) and the other to dynamic aspects (i.e., behaviors). The issues with the structures and behaviors impede conceptual processing and perceptual processing of information, respectively.

Finally, the cluster rating map indicates that individuals find learning behavioral aspects more challenging than learning structural aspects, which we may explain in terms of the criticality of perceptual processing in visual languages. Research has shown that diagrams' major cognitive advantage lies in their ability to shift some of the processing burden from conceptual processing to perceptual processing (Larkin & Simon, 1987; Petre, 1995). Our study discovers that UML notational elements that represent behaviors relate primarily to perceptual processes, whereas those that represent structures relate primarily to conceptual processes. Therefore, it makes sense that the first and fourth clusters had higher ratings on importance than the second, third, and fifth because the latter capitalize on the advantage of the power of diagrammatic representation.

6.2 Instructional Implications

First and foremost, we identify the 39 learning challenges that instructors need to address when teaching UML to improve learning outcomes. Our study suggests that instructors should prioritize the dynamic aspect of UML diagrammatic representation and, in particular, relationships and message passing. Specifically, instructors should emphasize how to deal with low perceptual discriminability in the combinations of 2D shape, paths, and strings. When teaching the static aspect, instructors should focus more on addressing the issues of ontological discrepancy and symbol overload in particular. While a variety of learning strategies (e.g., review session, study guide, and tutorials) can help students cope with the learning challenges, we focus on the pedagogical strategies that can facilitate perceptual and conceptual processes involved in learning behavioral and structural aspects of UML representations. Table 4 summarizes the teaching strategies that one can use to address specific learning challenges.

Familiarity with graphical symbols can help individuals search for, recognize, and infer information from perceptual cues in graphical symbols and, thus, facilitate perceptual processes (Briton & Jones, 1999) in learning behavioral aspects of UML notational elements. Therefore, instructors need to familiarize students with the graphical symbols that are difficult to recognize, counterintuitive to infer, and need serial searches. They can do so by having more practices and drills in the form of both in-class exercises and homework assignments. Based on our findings, students need to work on message passing and relationships to overcome challenges in learning the behavioral aspects of UML notational elements. As such, instructors need to allocate time for exercises in those areas both in and outside the classroom to enhance students' visual perception capability.

Table 4. Summary of Teaching Strategies

Learning challenges	Coping strategies in teaching
Perceptual challenges	
Low discriminability	• Assign more in-class exercises and homework assignments to familiarize students with notations, particularly in relationships and message passing. • Provide students with the descriptions and graphical displays of similar notations. • Require students to compare and outline differential features of similar notations.
Lack of semantic transparency	• Drill students more on notations that lack semantic transparency (e.g., the use of a stickman for non-human actors) to build and reinforce the connection between a graphical symbol and its semantic construct.
Conceptual challenges	
Symbol overload	• Lecture more on the semantic constructs that similar notations represent, such as relationships (generalization, dependency, and realization) and messages (sequential and asynchronous messages). • Conduct in-class mini-cases with the real-life scenarios so students can better comprehend the semantic constructs.
Symbol redundancy	• Summarize the semantic constructs with two or more graphical representations (e.g., server, interface, and generalization) with graphical illustration.
Complexity management	• Assign homework that focuses on linking and integrating information from different diagrams throughout the class.

Moreover, instructors should provide students with documents that include both the descriptions and graphical displays of the perceptually similar graphical symbols. They can also have students compare and outline the differential features of similar notational elements. Such activities can reinforce students' visual memories, which helps them to more easily recall and more quickly process perceptual information.

While untrained users find the lack of symbol discriminability and semantic transparency problematic, conceptual confusion can also cause challenges. Research has demonstrated that prior knowledge informs users about what to look for and how to organize information found in a diagram and, thus, improves their performance in searching for and recognizing relevant visual information (Winn, 1993). Students may not have knowledge about message passing. They may not firmly grasp the conceptual differences between a call and an asynchronous message and between generalization and realization. This lack of comprehension about the concepts that challenging graphical symbols represent can adversely affect perceptual processes and contribute to challenges in learning.

Similarly, the difficulties with symbols' meaning will burden conceptual processes. Construct overload for semantically related notations (e.g., excluding and included stereotypes) is more difficult to overcome than those for semantically distinct notations (e.g., component and node). Symbol inconsistencies for abstract concepts (e.g., interface) are more challenging than for concrete concepts (e.g., server). To alleviate learning challenges that arise from conceptual confusion, we recommend that instructors prioritize teaching visual semantics that students experience difficulty in comprehending. Instructors should provide more lectures on the cognitively challenging concepts (e.g., interfaces, active classes, and message passing) and the differences between concepts with close relations in meaning (e.g., stereotypes and relationships). We also recommend providing students with the opportunity to practice with real-life scenarios that pertain to the concepts under study. Instructors can approach such scenarios by using mini-cases (Sirias, 2005) in the class. If instructors use concrete examples that students can relate to, students can come to better understand the concepts, which, in turn, will help them to differentiate and comprehend the graphical symbols that represent those concepts.

Finally, we propose that instructors set homework that focuses on linking and integrating information from different diagrams to enhance conceptual integration. Instructors should assign this type of homework as soon as they introduce the two diagrams. As students progress through the course, instructors can decrease the frequency of homework assignments but increase their difficulty because it is harder and harder to navigate through multiple diagrams. Using time and repetition in this way to space learning can help students develop long-term memories that help them automatically process information.

7 Conclusions

Over the last two decades, UML has gained tremendous popularity. It is now commonly practiced in OO SA&D. College graduates with UML knowledge and skills are in demand, and the IS curriculum now generally covers UML. Because the IS curriculum widely teaches UML, more systematic research needs to evaluate teaching and learning of UML in IS education.

7.1 Contributions

In this study, we use a cognitive technique to investigate students' challenges in learning UML. The concept maps we constructed provide empirical evidence of the difficulties that novices experience in learning UML. We present concrete recommendations to address the learning challenges. Therefore, we provide valuable insights into and recommendations on teaching UML. Our study also demonstrates the usefulness of participatory cognitive approach in developing the shared understanding between instructors and students on difficult course materials. The concept-mapping technique enables one to meaningfully and efficiently assess and represent learning challenges, and other IS courses can use it to enhance teaching and facilitate learning.

Our study also has significance for the UML academic community. It identifies the cognitive weaknesses and deficiencies in UML from the perspective of inexperienced users and offers theoretical understanding of where and why UML notational elements hinder UML's diagrammatic reasoning. This knowledge can assist in UML diagrammatic representation's continuous evolution and further development to enhance UML's effectiveness. Finally, our study benefits UML practitioners, too. Student subjects in our study neared the end of their university education and were about to enter industry. As such, they represent entry-level practitioners. Thus, we can generalize our results to junior UML professionals. Our study sheds light on UML training that entry-level professionals may need.

7.2 Limitations and Future Research

Like any study, ours has limitations. First, we did not look into the more advanced aspects of UML diagrammatic representation. Instead, we focused on the UML basics that the participants had prior exposure to and revealed the challenges they faced in learning the fundamentals of UML notation. Future research needs to examine the UML notation that represents more advanced topics to understand the challenges individuals face in learning more complex and complicated areas of UML. In addition, since we used students as research participants, our findings have limited generalizability. We cannot generalize our findings to expert users. Research needs to replicate our study with UML experts to more comprehensively understand the challenges that individuals face in learning UML diagrams. Finally, the three sorting rules we used in structuring the statements step might have affected our study's results. The fact that we did not incorporate the "do not group dissimilar statements together into one pile" rule may have resulted in the fifth cluster. Future concept-mapping studies should consider and use important sorting rules that might influence the structuring process to resolve this methodological issue.

In conclusion, our concept-mapping study advances our empirical knowledge and theoretical understanding about the challenges that individuals face in learning UML. We hope that our study stimulates more empirical studies in UML in the context of IS education. Both educators and practitioners will find significance in work that focuses on more deeply understanding and appreciating UML in the IS curriculum.

References

Blankenship, J., & Dansereau, D. F. (2000). The effect of animated node-link displays on information recall. *Journal of Experimental Education, 68*(4), 293-308.

Booch, G., Rumbaugh, J., & Jacobson, I. (2005). *The unified modeling language user guide* (2nd Ed.). Upper Saddle River, NJ: Addison-Wesley.

Britton, C., & Jones, S. (1999). The untrained eye: How languages for software specification support understanding by untrained users. *Human Computer Interaction, 14*(1), 191-244.

Buzan, T. (1993). *The mind map book*. London, UK: BBC Books.

Caire, P., Genon, N., Heymans, P., & Moody, D. L. (2013). Visual notation design 2.0: Towards user comprehensible requirements engineering notations. In *Proceedings of 21st IEEE International Conference on Requirements Engineering*.

Eden, C., Ackermann, F., & Cropper, S. (1992). The analysis of cause maps. *Journal of Management Studies, 29*(3), 309-324.

Galvin, P. E. (1989). Concept mapping for planning and evaluation of a big brother/big sister program. *Evaluation and Program Planning, 12*(1), 53-58.

Goodman, N. (1968). *Languages of art: An approach to a theory of symbols*. New York: Bobbs-Merrill Co.

Hahn, J., & Kim, J. (1999). Why are some diagrams easier to work with? Effects of diagrammatic representation on the cognitive integration process of system analysis and design. *ACM Transactions on Computer-Human Interaction, 6*(3), 181-213.

Ihlenfeldt, W. A. (1981). Using cognitive mapping to capitalize on learning strengths. *Training and Development Journal, 35*(5), 98- 102.

Johnson, R. D., & Lipp, A. (2007). Cognitive mapping: A process to support strategic planning in an academic department. *Group Decision and Negotiation, 16*(1), 43-60.

Jonassen, D. H., Beissner, K., & Yacci, M. A. (1993). *Structural knowledge: Techniques for conveying, assessing, and acquiring structural knowledge*. Hillsdale, NJ: Lawrence Erlbaum Assoc.

Kane, M., & Trochim, W. M. K. (2007). *Concept mapping for planning and evaluation*. Thousand Oaks, CA: Sage.

Kim, J., Hahn, J., & Hahn, H. (2000). How do we understand a system with (so) many diagrams? Cognitive integration processes in diagrammatic reasoning. *Information Systems Research*, *11*(3), 284-303.

Larkin, J., & Simon, H. (1987). Why a diagram is (sometimes) worth ten thousand words? *Cognitive Science, 11*(1), 65-100.

Montazemi, A. R., & Conrath, D. W. (1986). The use of cognitive mapping for information requirements analysis. *MIS Quarterly, 10*(1), 45- 56.

Moody, D. L. (2009). The “physics” of notations: Toward a scientific basis for constructing visual notations in software engineering. *IEEE Transactions on Software Engineering, 35*(6), 756-779.

Nelson, K. M., Nadkarni, S., Narayanan, V. K., & Ghods, M. (2000). Understanding software operations support expertise: A revealed causal mapping approach. *MIS Quarterly*, *24*(3), 475-507.

Nerur, S. P., Slinkman, C. W., & Mahapatra, R. (2005). Special theme of research in information systems analysis and design—III teaching systems analysis and design: A case for the object oriented approach. *Communications of the Association for Information Systems, 16*, 848-860.

Nordbotten, J. C., & Crosby, M. E. (1999). The effect of graphic style on data model interpretation. *Information Systems Journal, 9*(2), 139-155.

Novak, D. J. (1993). How do we learn our lesson? Taking students through the process. *Science Teacher, 60*(3), 50-55.

Object Management Group. (2015). Documents associated with unified modeling language (UML) version 2.5. Retrieved from http://www.omg.org/spec/UML/2.5/

Petre, M. (1995). Why looking isn't always seeing: Readership skills and graphical programming. *Communications of ACM*, *38*(6), 33-44.

Robertson, S., & Williams, T. (2006). Understanding project failure: Using cognitive mapping in an insurance project. *Project Management Journal, 37*(4), 55-71.

Satzinger, J., Batra, D., & Topi, H. (2007). Analysis and design the IS curriculum: Taking it to the next level. *Communications of the Association for Information Systems, 20*, 483-496.

Siau, K., & Tan, X. (2005). Technical communications in information systems development: The use of cognitive mapping. *IEEE Transactions of Professional Communication, 48*(3), 269-284.

Sirias, D. (2005). Combining cooperative learning and conflict resolution techniques to teach information systems. *Journal of Education for Business, 80*(3), 153-158.

Swan, J. (1997). Using cognitive mapping in management research: Decisions about technical innovation. *British Journal of Management, 8*(2), 183-198.

Topi, H., Valacich, J. S., Wright, R. T., Kaiser, K., Nunamaker, J. F., Jr., Sipior, J. C., & de Vreede, G. J. (2010). IS 2010: Curriculum guidelines for undergraduate degree programs in information systems. *Communications of the Association for Information Systems, 26*, 359-428.

Topi, H., Karsten, H., Brown, S. A., Carvalho, J. A., Donnellan, B., Shen, J., Tan, B. C. Y., & Thouin, M. F. (2017). MSIS 2016 global competency model for graduate degree programs in information systems. *Communications of the Association for Information Systems, 40*, MSIS-i-MSIS-107.

Trochim, W. (1989). An introduction to concept mapping for planning and evaluation. *Evaluation and Program Planning, 12*(1), 1-16.

Trochim, W. (1993). The reliability of concept mapping. In *Proceedings of* the *Annual Conference of the American Evaluation Association.*

Trochim, W. M. K., & McLinden, D. (2017). Introduction to a special issue on concept mapping. *Evaluation and Program Planning*, *60*, 166-175.

Valentine, K. (1989). Contributions to the theory of care. *Evaluation and Program Planning, 12*(1), 17-24.

Wand, Y., & Weber, R. (1993). On the ontological expressiveness of information systems analysis and design grammars. *Journal of Information Systems, 3*(4), 217-237.

Winn, W. D. (1993). An account of how readers search for information in diagrams. *Contemporary Educational Psychology, 18*, 162-185.

Appendix

Table A1. List of Statements

Statement no.	Statement	Average rating
1	The nested/composite state has the same notation as the regular state. It is confusing to have the same notation for both cases.	2.50
2	The notations for the excluding and included stereotypes are confusing. Both use the same arrow-notation but the direction of the arrow is different.	3.50
3	The numbering notation in the collaboration diagram is difficult to follow.	3.31
4	The notations for base use case, include use case, and extend use case are all the same. It is difficult to determine which use case is the base use case in an extend or include relationship without additional symbols.	3.09
5	In the communication diagram, the format for the sequence number can be easily misunderstood as part of the operation or vice versa.	3.31
6	The notation for an actor is a stick man. However, other information systems outside the domain can be actors too and these systems are not human! So, the notation of a stick man for actor can be misleading.	3.16
7	The notations for class and object are the same. Since a class contains multiple objects, we should have a different notation for class (e.g., double-rectangle as a notation for classes).	3.34
8	The notations for sequential message, asynchronous message, and call are too similar.	3.09
9	Activity diagram is a special case of state-chart diagram but one cannot use a branch in a state-chart diagram.	2.78
10	The notation for lifelines of objects in the sequence diagram is confusing, and for most objects the lifeline is continuous.	2.50
11	The notations for generalization and realization are easily confused - both use a hollow arrowhead.	3.03
12	Collaboration could have a different shape, different from that of the use case.	3.16
13	Dependency and realization should differ more in their notations.	2.58
14	The notation for interface could be a little bigger so that instead of writing the interface name outside, we can write it inside the interface notation.	2.53
15	There are two notations for interface – the explicit style (as a rectangle) and the implicit style (as a circle).	3.09
16	Component and node notations are pretty hard to differentiate.	1.97
17	A dashed line with an arrow at the end is used to represent both dependency and extend/include relationship.	3.53
18	Multiplicity of an association's role is difficult to follow -- is there a better notation than using 0 … *?	2.44
19	The notations for note and class are too similar.	1.69
20	The notations for states in state-chart diagram and classes in class diagram are too similar.	1.72
21	Different styles of arrow heads are used in UML. It is very hard to differentiate one from the other. More obvious notation difference should be there other than the filled or hollow arrowhead.	3.22
22	The notation for association class is the same as that of regular class.	2.94
23	Multiplicity notation (e.g., 1…*) can sometimes clutter up a diagram (i.e., making it messier).	2.84
24	Classes, notes, packages, and components all have a rectangular base for their notations.	1.91
25	The dashed line with arrow is used to represent several different things (dependency, realization).	3.47
26	Extend and include relationships use the same notation -- an arrow with dotted line.	3.50

Table A1. List of Statements

27	For communication diagrams, the notation for sequence number—thread names, and iteration expressions—can become confusing and jumbled. Not easy to decipher information.	3.28
28	The composition and aggregation symbols should be more different from each other.	2.50
29	The notation for component in a component diagram is not standardized (e.g., a database can be represented using different notations – cylinder, rectangle with tabs, etc.).	3.22
30	The notation for node in a deployment diagram is not standardized (e.g., a serve can be represented using a computer symbol or using a cube).	2.63
31	The notation for an active class is represented differently in different books.	3.00
32	The notation for template parameters is confusing.	2.50
33	The same rounded rectangle is used to represent activity state and action state in the activity diagram.	3.53
34	There are different notations for classes. A more consistent structure should be followed for drawing the boxes. For example, (a) is how classes are drawn, but if one does not have any methods or responsibility it is drawn like (b). Keep the number of compartments consistent even if some compartments are empty.	2.81
35	There should be notations to specify “linkages” among various diagrams. For example, actors are used in sequence and collaboration diagram to provide “correspondence” to actors in the use case diagram. More notations can be used to link up other diagrams.	3.41
36	The naming of classes and objects has to be more different. There has to be a way to differentiate classes from objects other than a letter followed by a colon.	2.88
37	The notations for join and fork are the same (using the synchronization bar).	2.47
38	In generalization, the two styles (i.e., the direct and tree style) should not be allowed.	2.84
39	Stick arrow is used in combination with different line styles to represent many different things.	3.53

About the Authors

Zixing Shen is an Associate Professor of Management at Dakota State University. She earned her PhD in Management from Case Western Reserve University. Her research in information modeling, the organizational impacts of IT, and IT-driven innovation has been published in *European Journal of Information Systems*, *IEEE Transactions of Engineering Management*, and *Communications of ACM*, among others, and presented at conferences such as Academy of Management Annual Meeting, International Conference on Information Systems (ICIS), and Hawaii International Conference on Systems Science (HICSS).

Songxin Tan is an Associate Professor of Electrical Engineering at South Dakota State University. He earned his PhD in Electrical Engineering from University of Nebraska—Lincoln. His research interests include lidar design, lidar remote sensing, digital imaging processing and 3D sensing and machine vision. His research work has been published in such journals as *Applied Optics* and *Optical Engineering*, presented at various conferences including the IEEE Geoscience and Remote Sensing Symposium and SPIE Optics and Phonics.

Keng Siau is Chair of the Department of Business and Information Technology (BIT) at the Missouri University of Science and Technology. She is Editor-in-Chief of the *Journal of Database Management* and served as the North America Regional Editor of the Requirements Engineering journal from 2010-2016. He also served as the Vice President of Education for the Association for Information Systems and served as the AIS representative on the Board of Partnership for Advancing Computing Education (PACE) from July 2011-June 2014 (i.e., three-year term). Professor Siau has more than 250 academic publications. According to Google Scholar, he has more than 10,000 citation counts. His h-index and i10-index, according to Google Scholar, are 50 and 122 respectively. She is consistently ranked as one of the top information systems researchers in the world based on h-index and productivity rate. She has received numerous teaching, research, service, and leadership awards including the prestigious International Federation for Information Processing (IFIP) Outstanding Service Award in 2006, IBM Faculty Awards in 2006 and 2008, and IBM Faculty Innovation Award in 2010.