

Facilitating the Transition from Use Case Models to Analysis Models: Approach and Experiments

Tao Yue ^{φ§}, Lionel C. Briand ^φ, and Yvan Labiche [§]

^φ Simula Research Laboratory &
University of Oslo,
P.O. Box 134, Lysaker, Norway
{tao, briand}@simula.no

[§] Carleton University
Software Quality Engineering Lab,
1125 Colonel By Drive
Ottawa, ON K1S 5B6, Canada
labiche@sce.carleton.ca

ABSTRACT

Use case modeling (UCMod), including use case diagrams and use case specifications (UCSs), is commonly applied to structure and document requirements. Use case specifications are usually structured, unrestricted textual documents complying with a certain use case template. However, because UCMs remain essentially textual, ambiguity is inevitably introduced. In this paper, we propose a UCM approach, called Restricted Use Case Modeling (RUCM), which is composed of a set of well-defined restriction rules and a modified use case template. The goal is two-fold: (1) restrict the way users can document use case specifications in order to reduce ambiguity and (2) facilitate automated analysis in order to provide tool support to derive initial analysis models, which in UML are typically composed of class diagrams, interaction diagrams, and possibly other types of diagrams and constraints. The derivation of analysis models from use case models is currently poorly supported by tools and the process is mostly manual.

Though the proposed restriction rules and template are based on a clear rationale, two main questions need to be investigated. Do users find them too restrictive or impractical in certain situations? Second, do the rules and template have a positive, significant impact on the quality of the constructed analysis models? To investigate these questions, we performed and report on controlled experiments, which evaluate the restriction rules and use case template in terms of whether they are easy to apply while developing use case

models and facilitate understanding of use case specifications, and whether they help obtain higher quality analysis models in terms of correctness, completeness, and redundancy. Results show that, the restriction rules are overall easy to apply and that RUCM results into significant improvements over traditional approaches (i.e., with standard templates, without restrictions) in terms of class correctness and class diagram completeness, message correctness and sequence diagram completeness, and understandability of use case specifications.

Keywords: Use Case; Use Case Modeling; Use Case Template; Restriction Rules; Analysis Model; Class Diagram; Sequence Diagram; Controlled Experiment.

Table of Contents

ABSTRACT	1
1 INTRODUCTION	8
2 RELATED WORK	10
2.1 Use case template	10
2.2 Restriction rules.....	12
2.3 Empirical evaluation	14
3 USE CASE MODELING APPROACH (RUCM).....	15
3.1 Use case template	15
3.2 Restriction rules.....	18
3.3 Example	21
4 EXPERIMENT PLANNING	23
4.1 Experiment Definition.....	23
4.2 Context Selection and Subjects.....	24
4.3 Hypotheses Formulation.....	25
4.4 Experiment Design.....	26
4.4.1 Experiment 1	26
4.4.2 Experiment Replication (Experiment 2)	28
4.5 Instrumentation.....	29
4.5.1 Task 1.....	29
4.5.2 Task 2.....	31
4.6 Evaluation measurement and data collection	36
4.6.1 Error Rate	36
4.6.2 Understandability, Applicability and Restrictiveness	39
4.6.3 Quality of analysis class diagram (CD)	39
4.6.4 Quality of analysis sequence diagram (SD) and consistency of analysis class and sequence diagrams (CS)	42
4.6.5 Correctness of responses to the comprehension questionnaires (QC).....	44
5 EXPERIMENT RESULTS AND ANALYSIS	45
5.1 Usage of restriction rules	46
5.1.1 Error rate	46
5.1.2 Understandability.....	48
5.1.3 Applicability	49
5.1.4 Restrictiveness.....	49
5.1.5 Correlation Analysis.....	50
5.2 Quality of analysis models	51
5.2.1 Quality of analysis class diagram (CD)	51
5.2.2 Quality of analysis sequence diagram (SD)	61
5.2.3 Consistency between analysis class and sequence diagrams (CS)	69
5.2.4 Correctness of responses to the comprehension questionnaire	70
6 THREATS TO VALIDITY.....	71
7 CONCLUSION	73

REFERENCES	74
APPENDIX A RESTRICTION RULES	77
APPENDIX B EXPERIMENT 1- COMPREHENSION QUESTIONNAIRE	81
APPENDIX C EXPERIMENT 2 - COMPREHENSION QUESTIONNAIRE (CPD) ..	84
APPENDIX D EXPERIMENT 2 - COMPREHENSION QUESTIONNAIRE (VS)	90
APPENDIX E PRE- AND POST- LAB QUESTIONNAIRES	98
APPENDIX F EXPERIMENT DATA	106

Table of Figures

Figure 1 Defining use case model and deriving analysis model activities	9
Figure 2 Error rates of restriction rules (when $\geq 5\%$)	47
Figure 3 Understandability of the restriction rules (when $< 90\%$)	48
Figure 4 High applicability of the restriction rules (when $< 90\%$)	49
Figure 5 High restrictiveness of the restriction rules (when $> 20\%$)	50
Figure 6 Distributions of Class Completeness - Labs 1&2	53
Figure 7 Distributions of CD Completeness - Labs 1&2	53
Figure 8 Distributions of Class Correctness - Labs 1&2	54
Figure 9 Distributions of Redundancy - Labs 1&2	54
Figure 10 Distributions of Class Completeness - Lab 2.1	55
Figure 11 Distributions of CD Completeness - Lab 2.1	55
Figure 12 Distributions of Class Correctness - Lab 2.1	55
Figure 13 Distributions of Redundancy - Lab 2.1	56
Figure 14 Distributions of Class Completeness - Lab 2.3	56
Figure 15 Distributions of CD Completeness - Lab 2.3	56
Figure 16 Distributions of Class Correctness - Lab 2.3	57
Figure 17 Distributions of Redundancy - Lab 2.3	57
Figure 18 Distributions of Message Completeness - Lab 2.2	63
Figure 19 Distributions of SD Completeness - Lab 2.2	63
Figure 20 Distributions of Message Correctness - Lab 2.2	63
Figure 21 Distributions of SD Correctness - Lab 2.2	64
Figure 22 Distributions of BCE Consistency - Lab 2.2	64
Figure 23 Distributions of Message Completeness - Lab 2.4	64
Figure 24 Distributions of SD Completeness - Lab 2.4	65
Figure 25 Distributions of Message Correctness - Lab 2.4	65
Figure 26 Distributions of SD Correctness - Lab 2.4	65
Figure 27 Distributions of BCE Consistency - Lab 2.4	66
Figure 28 Least-square means for ANOVA interaction analysis between <i>Method</i> and <i>System</i> for CD - Lab 2.1	113
Figure 29 Least-square means for ANOVA interaction analysis between <i>Method</i> and <i>System</i> for CD - Lab 2.3	114
Figure 30 Least-square means for ANOVA interaction analysis between <i>Method</i> and <i>Order</i> for CD - Lab 2.1 and Lab 2.3	115
Figure 31 Least-square means for ANOVA interaction analysis between <i>Method</i> and <i>System</i> for SD - Lab 2.2	116
Figure 32 Least-square means for ANOVA interaction analysis between <i>Method</i> and <i>System</i> for SD - Lab 2.4	117
Figure 33 Least-square means for ANOVA interaction analysis between <i>Method</i> and <i>Order</i> for SD - Lab 2.2 and Lab 2.4	118

Table of Figures

Table 1 Use case template.....	18
Table 2 Restriction rules (R1-R7).....	19
Table 3 Restriction rules (R8-R16).....	19
Table 4 Restriction rules (R17-R26).....	21
Table 5 Use case Withdraw Fund (part 1)	22
Table 6 Goal 1	23
Table 7 Goal 2.....	24
Table 8 Hypotheses – Experiment 2	26
Table 9 Participant groups and tasks – Experiment 1	27
Table 10 Participant groups and tasks – Experiment 2.....	29
Table 11 Comprehension questionnaire of Task 1	31
Table 12 Classification and examples of comprehension questions.....	35
Table 13 Question distributions of the comprehension questionnaires for Task 2.....	35
Table 14 Measures used to derive Error Rate.....	37
Table 15 Error Rate measurements (R1–R16).....	38
Table 16 Error Rate measurements (R17–R26).....	38
Table 17 Measures used to derive CD	40
Table 18 Quality measures for a class	41
Table 19 Quality measures for a class diagram	41
Table 20 Measures used to derive SD.....	43
Table 21 Quality measures for a sequence diagram	44
Table 22 Collected and analyzed data points – Experiment 2	46
Table 23 Spearman nonparametric correlation analysis – correlation table	50
Table 24 two tailed <i>t</i> -test and Wilcoxon test– CD.....	59
Table 25 Summary of <i>t</i> -test results of CD	60
Table 26 Summary of ANOVA tests - CD.....	61
Table 27 one tailed <i>t</i> -test and Wilcoxon test– SD.....	67
Table 28 Summary of <i>t</i> -test results of SD.....	69
Table 29 Summary of ANOVA tests - SD.....	69
Table 30 Descriptive statistics of CS – lab 2.2 and lab 2.4	70
Table 31 One way <i>t</i> -test – CS – lab 2.2 and lab 2.4.....	70
Table 32 Descriptive statistics of QC – Experiment 1, Experiment 2: lab 2.2, and lab 2.3	70
Table 33 <i>t</i> -test – QC – Experiment 1, Experiment 2: lab 2.2, and lab 2.3.....	70
Table 34 Restrictions (part 1)	77
Table 35 Restrictions (part 2)	78
Table 36 Restrictions (part 3)	79
Table 37 Restrictions (part 4)	80
Table 38 Experiment 1 – Comprehension Questionnaire – Part 1 (R1-R16)	82
Table 39 Experiment 1 – Comprehension Questionnaire – Part 2 (R17-R26)	83
Table 40 Error rates of the restriction rules	106

Table 41 Understandability of the restriction rules.....	106
Table 42 Applicability of the restriction rules (frequencies).....	106
Table 43 Restrictiveness of the restriction rules (frequencies).....	107
Table 44 Descriptive statistics of measure CD – Lab 2.1 and 2.3.....	108
Table 45 Descriptive statistics of measure SD – Lab 2.2 and 2.4.....	109
Table 46 ANOVA – Interaction between <i>Method</i> (UCM_R vs. UCM_UR) and <i>System</i> (CPD vs. VS) for CD – Lab 2.1 and Lab 2.3.....	110
Table 47 ANOVA – Interaction between <i>Method</i> (UCM_R vs. UCM_UR) and <i>Order</i> (2.1 vs. 2.3) on CD – Lab 2.1 and Lab 2.3.....	110
Table 48 ANOVA – Interaction between <i>Method</i> (UCM_R vs. UCM_UR) and <i>System</i> (CPD vs. VS) for SD – Experiment 2.....	111
Table 49 ANOVA – Interaction between <i>Method</i> (UCM_R vs. UCM_UR) and <i>Order</i> (2.2 vs. 2.4) on SD – Lab 2.2 and Lab 2.4.....	112

1 INTRODUCTION

Use case modeling (UCMod), including use case diagrams and use case textual specifications, is commonly applied to structure and document requirements [9, 13, 15]. Use Case Specifications (UCS) are usually textual documents complying with a use case template that, though helping the reading and reviewing of use cases, inevitably contains ambiguities. In this paper, we propose a set of restriction rules and a use case template, which are based in part on the results of a thorough systematic literature review [31]. The goal is to restrict the way users can document UCSs in order to reduce ambiguity and facilitate the manual or automated construction of initial analysis models, which in the context of a Unified Modeling Language (UML) [19] based development are typically at least composed of class and interaction diagrams, and possibly other types of diagrams and constraints. Our use case modeling approach is denoted as Restricted Use Case Modeling (RUCM).

The restriction rules and the use case template we specify should be applied during the requirements elicitation phase of a use case-driven software development (e.g., [5]) to produce, to the maximum extent possible, precise and unambiguous use case models. This is modeled as the first activity, activity `Define Use Case Model`, of the activity diagram in Figure 1. Our use case modeling approach (RUCM) takes in input the use case diagram, the restriction rules, and the use case template. A use case diagram in UML [19], is used to represent relationships among actors and use cases; the restriction rules we defined are applied to restrict the way users can write UCSs; the use case template is a means to structure UCSs. With a RUCM use case model, an initial analysis model can then be derived as denoted by the activity `Derive Analysis Model`. This step is usually manually performed by system analysts but can also be partially automated, especially if inputted use case models are expressed in a not too ambiguous form.

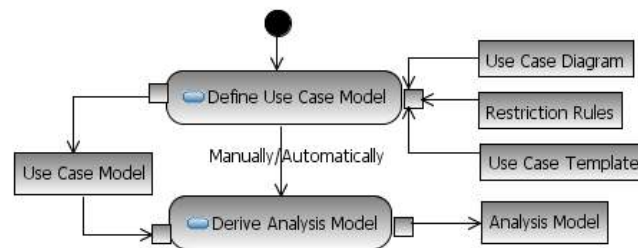


Figure 1 Defining use case model and deriving analysis model activities

Though the restriction rules we proposed are based on a clear rationale, users might find them too restrictive or impractical in certain situations. They must therefore be experimentally evaluated in order to assess whether (1) they are easy to apply while developing use case models, (2) they help the designer generate higher quality analysis models than what can be generated when they are not used, and (3) they can help automatically deriving analysis models from use case models. This paper investigates the first two questions through controlled experiments with fully-trained, 4th year undergraduate students. The third question is addressed in [30, 33] where an approach called aToucan is proposed to automatically derive an analysis model including class, sequence and activity diagrams, from a RUCM use case model.

This paper is an extension of reference [32], where we first described RUCM and reported on an initial controlled experiment to evaluate its applicability and impact on the quality of manually derived analysis models. The extensions are the following: 1) The first experiment [32] was not able to evaluate the impact of RUCM on manually derived sequence diagrams because participants did not have enough time to generate these diagrams. Therefore, we replicated the experiment, allowing more time to participants to create sequence diagrams. Replication also allowed us to have more data points; 2) Due to space limitation of the conference paper [32], the design and analysis of the initial experiment were partially reported. In this paper, we report them in details.

The rest of the paper is organized as follows. The related work is reported in Section 2. Section 3 discusses RUCM: the use case template and the restriction rules. The experimental evaluation of RUCM is presented in Section 4 (experiment planning),

Section 5 (experiment results and analysis), and Section 6 (threats to validity). Last we conclude in Section 7.

2 RELATED WORK

Several streams of research relate to our work: use case templates (Section 2.1), restriction rules (Section 2.2), and experimental evaluations of use case modeling approaches (Section 2.3).

2.1 Use case template

The concept of use case has been first introduced by Ivar Jacobson in 1986 to capture functional requirements [12]. Since then, use cases have been widely accepted and use case modeling techniques have evolved and matured. The concept of use case is part of the UML (though UML does not support use case specification), which provides a use case diagram to specify relations between use cases and between use cases and actors. As the importance of use cases (requirements) is increasingly recognized, use case modeling is more than a requirements specification technique; it drives the whole software development process as most activities (e.g., analysis, design, and test) start from use cases. This is referred to as use case-driven software development [14].

It is a common practice to follow a use case template to structure UCSs, thereby helping their writing, reading and reviewing. Various use case templates (e.g., [7, 14-17]) have been suggested in the literature to satisfy different application contexts and purposes. They share common fields such as: *use case name*, brief overall *description*, *precondition*, *postcondition*, *basic flow*, and *alternative flows*. The template suggested in [7] introduces additional fields such as *scope*, *level* (level of abstraction) and *stakeholders and interests*. The Rational Unified Process (RUP) [15] suggests a template with six fields: *use case name*, *flow of events* (basic flow, alternative flows), *special requirements* (e.g., performance and security requirements), *preconditions*, *postconditions*, and *extension points*. Kulak and Guiney [16] suggest a template that includes an *exception paths* field:

exception paths are distinguished from alternative flows since they are paths taken when errors occur. Both Cockburn and Larman [7, 17] mention that use cases can be written at different levels of details: brief, casual, and fully dressed. The templates they propose are similar and composed of most of the fields that have been proposed in the literature and that we mentioned previously. In addition, a template for describing actors is proposed in [2], which consists of three sections: *actor name*, *description*, and *examples*.

In addition to capturing requirements, use cases can also facilitate the automated derivation of an initial analysis model – one of our goals. The systematic literature review [31] we conducted to examine techniques that transform textual requirements into analysis models revealed that six approaches require use cases. The approach proposed in [18] relies on the RUP use case template [15]. The use case template proposed in [25] contains eight fields: title (i.e., use case name), primary actor, participants (i.e., secondary actors), goal (i.e., brief description), precondition, postcondition, steps (i.e., basic flow steps), and alternatives. An enriched use case template is proposed in [11], which is composed of three sections: use case summary, basic flows, and alternative flows. The use case summary section is further divided into four subsections: use case name, actors, cross-reference, and an overview of the use case purpose (may be used to describe non-functional requirements). The cross-reference section is used to link the use case to high-level requirements. A three-column structure (i.e., general, actor/system communication, and system response) is introduced in this use case template to structure the steps of flow of events. The general column describes general activities that are not supposed to be implemented by the system but can help users better understand the use case. Steps in the actor/system communication column specify actions performed by actors when interacting with the system. The system response column represents reactions of the system, including changes of state.

The use case template we propose in this paper (Section 3.1) contains fields similar to those encountered in conventional use case templates but with a few variations on the structure of the flow of events. The motivation is to support the automated generation of

analysis models and to further reduce ambiguity. Our objective is to propose a new use case template that not only complies as much as possible with conventional use case templates but also facilitates the process of deriving analysis models. Therefore, we made the following decisions: (1) We included fields commonly encountered in most templates: use case name, brief description, primary actor, secondary actor, precondition, postcondition, basic and alternative flows. (2) Some of the fields (e.g., *scope*, *level*, and *special requirements*) proposed in the literature to capture requirements were excluded since they do not help deriving analysis models. (They could easily be added though.) (3) We excluded the fields that, on the one hand may increase the precision of UCSs but, on the other hand require that the designer provide much more information than what is actually needed for our purpose. In other words, we believe that the additional precision does not warrant the additional cost, and that these fields do not bring clear advantages with respect to our objectives. For example, the semantics of the three-column steps modeling style proposed in [11] (discussed above) can actually be automatically derived from UCSs by grammatically analyzing each sentence; therefore we made a design decision not to include this style into our use case template. (4) Six interaction types (five from [7], one we newly propose in this work) are suggested to describe action steps of flow of events. (5) Differing from most of existing use case templates that suggest having one postcondition for one use case, our template enforces that each flow of events (both basic flow and alternative flows) of a UCS contains its own postcondition: the postcondition of the use case is simply the disjunction of all those postconditions.

2.2 Restriction rules

In [31], we summarize and classify the restriction rules (also called writing guidelines) applied in [24-26, 28], which propose transformations from requirements to analysis models. In this paper, we propose a total of 26 restriction rules on the use of Natural Language (NL) to document UCSs that complies with our use case template. Some of these restrictions are the results of the systematic review [31] we conducted (see further details in Section 3.2); others are heuristics suggested in the literature on writing use

cases (e.g., [1, 4, 7]); last some of them are derived from our experience in writing UCSs when attempting to reduce ambiguity and facilitating automated transition to analysis models. None of the related work we looked at relies on a set of rules as complete as the one we suggest.

Existing guidelines are recommended, based on practitioners' experience in writing UCSs, to reduce ambiguities of UCSs or to facilitate the process of deriving analysis models from them. We reused some of them, excluded others, recommended new ones, and classified all the rules. For example, we excluded the rules that constrain grammatical sentence structures, such as only allowing certain types of structures such as subject-verb-object (e.g., [1, 8]), because these structures can be automatically obtained by grammatically analyzing each sentence using a NL parser. We also excluded the rules that put excessive constraints on wording. For example, one such rule suggests using "is a kind of", "is specification of" or "is generalization of" to describe inheritance between the subject and object of a sentence.

Additionally, we explicitly describe why each of our restriction rules is needed either to reduce ambiguities or facilitate the process of deriving analysis models, a crucial piece of information that is often omitted in the literature. We also indicate how and where to apply (Section 3.2) each of our restriction rules, another piece of information often left out by most papers on the topic. Several rules we newly propose in this work are based on our experience with several NL parsers (e.g., [27]) and are proposed because sentences with certain structures cannot be correctly parsed. These rules can also help reduce ambiguity of UCSs and therefore help to manually derive analysis models from them. Furthermore, as opposed to many related works, our restriction rules are integrated with our use case template together as a comprehensive solution for use case modeling: several of our restriction rules refer to some of the features of our use case template (Section 3.2).

2.3 Empirical evaluation

Some empirical studies have been conducted to evaluate the impact of applying restriction rules on the quality of UCSs. Achour et al. [1] investigated the effectiveness of the CREWS rules [1] in terms of the completeness and structuredness of UCSs: UCSs were evaluated by comparing them to “expert” UCSs developed by the authors of the paper. The experiment results show that the application of the rules produced more complete and better structured UCSs. Phalp et al. [22] conducted an empirical study to compare two sets of writing rules: the CREWS rules and CR rules [8] (leaner than the CREWS rules). The overall quality of UCSs was evaluated based on seven quality factors, referred as the ‘7Cs of communicability’ [21]: coverage (a use case should contain all the required information), cogent (a sentence should follow a logical path), coherent (sentences should be all connected by, for example, repeating a noun), consistent abstraction, consistent structure, consistent grammar, and consideration of alternatives. The experiment results from [22] show that the leaner set of rules (the CR rules) results in less learning overhead than the CREWS rules and performs at least as well as the CREWS rules, in terms of the overall quality of produced UCSs. Anda et al. [2] conducted a similar experiment to compare three different sets of guidelines: minimal guidelines (guidelines on identifying actors and use cases), template guidelines (a commonly applied actor and use case template based on templates proposed in [7, 16, 23]), style guidelines (modified version of the CREWS rules, focusing on the documentation of the flow of events of each use case). UCSs were evaluated in terms of their understandability, usefulness, and quality. The experiment results show that the template guidelines led to the highest understandability, usefulness, and overall quality, and that the style guidelines performed better than the minimal guidelines. The authors also suggest that combining the style guidelines with the template guidelines might further improve quality attributes of UCSs to compare with independently applying the template or style guidelines. This is exactly the case of our RUCM.

All these experiments evaluated restriction rules as a whole; none of them evaluated each rule individually. We however evaluate each of our restriction rules both individually and as a whole. By doing so, we are able to tell which rule(s) are difficult to apply and therefore where extra focus and significant practical exercises during training should be given. It is also worth noticing that all of them assess the quality of UCSs against some quality criteria (e.g., understandability, structuredness, completeness), rather than test the ability of subjects to extract information from UCSs such as deriving analysis models from UCSs. In this paper, we report on two controlled experiments, which evaluate the impact of our restriction rules and use case template both on the understandability of UCSs and the quality of analysis models manually generated from them in terms of correctness, completeness, and redundancy.

3 USE CASE MODELING APPROACH (RUCM)

In the context of UML-based development, a use case model, modeling the functional requirements of a system, is composed of a UML use case diagram and a set of structured, textual UCSs. Each use case of a use case diagram is described in a textual UCS complying with a certain use case template. We specify a new use case template (Section 3.1) that merges several aspects of existing templates. We also propose a set of restrictions to the use of plain language for documenting UCSs (Section 3.2). An example of applying our restriction rules and the template is provided in Section 3.3.

3.1 Use case template

Our use case template has eleven first-level fields (first column of Table 1). The last four fields are decomposed into second-level fields (second column of the last four rows). The last column of each row explains the corresponding field(s). There is no need to further discuss the first seven fields since they are straightforward and commonly encountered in many templates. Below we focus the discussion on the *Basic Flow* field and the three types of *Alternative Flows*: specific, global, and bounded alternative flows.

A *basic flow* describes a main successful path. It often does not include any condition or branching [17]. It is recommended to describe separately the conditions and branching in alternative flows. A basic flow is composed of a sequence of steps and a postcondition. Each UCS can only have one basic flow. The action steps can be one of the following five interactions, which are reused from [7] except for the fifth:

1. Primary actor \rightarrow system: the primary actor sends a request and data to the system.
2. System \rightarrow system: the system validates a request and data.
3. System \rightarrow system: the system alters its internal state (e.g., recording or modifying something).
4. System \rightarrow primary actor: the system replies to the primary actor with a result.
5. System \rightarrow secondary actor: the system sends requests to a secondary actor.

All steps are numbered sequentially. This implies that each step is completed before the next one is started. If there is a need to express conditions, iterations, or concurrency, then specific keywords, specified as restriction rules in Section 3.2, should be applied.

Alternative flows describe all the other scenarios or branches, both success and failure. An alternative flow always depends on a condition occurring in a specific step in a flow of reference, referred to as *reference flow*, and that reference flow is either the basic flow or an alternative flow itself. The branching condition is specified in the reference flow by following restriction rules (R20 and R22—Section 3.2). We refer to steps specifying such conditions as *condition steps* and the other steps as *action steps*. Similarly to the basic flow, an alternative flow is composed of a sequence of numbered steps. We classify alternative flows into three types: specific, global, and bounded alternative flows. This classification is adapted from [4]. A *specific alternative flow* is an alternative flow that refers to a specific step in the reference flow. A *bounded alternative flow* is an alternative flow that refers to more than one step in the reference flow—consecutive steps or not. A *global alternative flow* (called *general alternative flow* in [4]) is an alternative flow that refers to any step in the reference flow. Distinguishing different types of alternative flows makes interactions between the reference flow and its alternative flows much clearer. For

specific and bounded alternative flows, a RFS (Reference flow Step) section, specified as rule R19 (Section 3.2), is used to specify one or more (reference flow) step numbers. Whether and where the flow of an alternative flow merges back to the reference flow or terminates the use case must be specified as the last step of the alternative flow. Similarly to the branching condition, merging and termination are specified by following restriction rules (R24 and R25—Section 3.2). By doing so, we can avoid potential ambiguity in UCSs, caused by unclear specification of interactions between the basic flow and its corresponding alternative flows. Each alternative flow must have a postcondition (enforced by restriction rule R26—Section 3.2).

It is usual to provide a postcondition describing a constraint that must be true when a use case terminates. If the use case contains alternative flows, then the postcondition of the use case should describe not only what must be true when the basic flow terminates but also what must be true when each alternative flow terminates. The branching condition to each alternative flow is then necessarily part of the postcondition (to distinguish the different results). In such a case, the postcondition can become complex and the branching condition for each alternative flow is redundantly described (both in the steps of flows and the postcondition), which therefore increases the risk of ambiguity in UCSs. Our template enforces that each flow (both basic flow and alternative flows) of a UCS contains its own postcondition and therefore avoids such ambiguity.

Table 1 Use case template

Use Case Name	The name of the use case. It usually starts with a verb.	
Brief Description	Summarizes the use case in a short paragraph.	
Precondition	What should be true before the use case is executed.	
Primary Actor	The actor which initiates the use case.	
Secondary Actors	Other actors the system relies on to accomplish the services of the use case.	
Dependency	Include and extend relationships to other use cases.	
Generalization	Generalization relationships to other use cases.	
Basic Flow	Specifies the main successful path, also called “happy path”.	
	Steps (numbered)	Flow of events.
	Postcondition	What should be true after the basic flow executes.
Specific Alternative Flows	Applies to one specific step of the basic flow.	
	RFS	A reference flow step number where flow branches from.
	Steps (numbered)	Flow of events.
Global Alternative Flows	Applies to all the steps of the basic flow.	
	Steps (numbered)	Flow of events.
	Postcondition	What should be true after the alternative flow executes.
Bounded Alternative Flows	Applies to more than one step of the basic flow, but not all of them.	
	RFS	A list of reference flow steps where flow branches from.
	Steps (numbered)	Flow of events.
	Postcondition	What should be true after the alternative flow executes.

3.2 Restriction rules

The restriction rules are classified into two groups: restrictions on the use of natural language, and restrictions enforcing the use of specific keywords for specifying control structures. The first group of restrictions is further divided into two categories according to their location of application (see below). Each restriction rule is assigned a unique number.

Seven restriction rules (R1-R7) constrain the use of natural language (Table 2): the table explains why they are needed to reduce ambiguity. Notice that these rules apply only to action steps; they do not apply to condition steps or preconditions or postconditions. The other nine restriction rules (Table 3) apply to all sentences in a UCS: action steps, condition steps, preconditions, postconditions, and sentences in the brief description. Rules R8-R10 and R16 are to reduce ambiguity of UCSs; the remaining rules specifically facilitate automated generation of analysis models, though they can also help reduce

ambiguity. These two sets of restrictions are thought to be good practice for writing clear and concise UCSs (e.g., [4, 7, 23]) except for R13 and R15. We include these two rules because we perceived that negative adverbs, negative adjectives, and participle phrases are very difficult to parse for natural language parsers. R9 requires using words consistently to document UCSs. A common approach to do so is to use a domain model and glossary (e.g., [17], [5]) as a basis to write UCSs.

Table 2 Restriction rules (R1-R7)

#	Description	Explanation
R1	The subject of a sentence in basic and alternative flows should be the system or an actor.	Enforce describing flows of events correctly. These rules conform to our use case template (the five interactions).
R2	Describe the flow of events sequentially.	
R3	Actor-to-actor interactions are not allowed.	
R4	Describe one action per sentence. (Avoid compound predicates.)	Otherwise it is hard to decide the sequence of multiple actions in a sentence.
R5	Use present tense only.	Enforce describing what the system does, rather than what it will do or what it has done.
R6	Use active voice rather than passive voice.	Enforce explicitly showing the subject and/or object(s) of a sentence.
R7	Clearly describe the interaction between the system and actors without omitting its sender and receiver.	

Table 3 Restriction rules (R8-R16)

#	Description	Explanation
R8	Use declarative sentence only. "Is the system idle?" is a non-declarative sentence.	Commonly required for writing UCSs.
R9	Use words in a consistent way.	Keep one term to describe one thing.
R10	Don't use modal verbs (e.g., <i>might</i>)	Modal verbs and adverbs usually indicate uncertainty; Instead, metrics should be used if possible.
R11	Avoid adverbs (e.g., <i>very</i>).	
R12	Use simple sentences only. A simple sentence must contain only one subject and one predicate.	Facilitate automated natural language parsing and reduce ambiguity.
R13	Don't use negative adverb and adjective (e.g., <i>hardly, never</i>), but it is allowed to use <i>not</i> or <i>no</i> .	
R14	Don't use pronouns (e.g. <i>he, this</i>)	
R15	Don't use participle phrases as adverbial modifier. For example, the italic-font part of the sentence "ATM is idle, <i>displaying a Welcome message</i> ", is a participle phrase.	
R16	Use "the system" to refer to the system under design consistently.	Keep one term to describe the system; therefore reduce ambiguity.

The remaining ten restriction rules (Table 4) specify control structures, except R26 that specifies that each basic flow and alternative flow should have its own postcondition. R17 and R18 specify keywords to describe use case dependencies *include* and *extend*. R19 specifies keyword RFS, which is used in a specific (or bounded) alternative flow to refer to a step number (or a set of step numbers) of a reference flow that this alternative flow branches from.

Rules R20-R23 specify the keywords used to specify conditional logic sentences (IF-THEN-ELSE-ELSEIF-ENDIF), concurrency sentences (MEANWHILE), condition checking sentences (VALIDATES THAT), and iteration sentences (DO-UNTIL), respectively. The keyword IF-THEN-ELSE-ELSEIF-ENDIF can be used in three different ways (these are specified as a grammar): 1) IF-THEN-ENDIF (appears in one flow only), 2) IF-THEN-ELSE-ENDIF (everything is in one flow, or IF-THEN in one flow and ELSE in an alternative flow), and 3) IF-THEN-ELSEIF-THEN-ELSE-ENDIF (everything is in one flow, or IF-THEN in a flow and ELSEIF-THEN-ELSE-ENDIF in an alternative flow). Keyword VALIDATES THAT (R22) means that the condition is evaluated by the system and must be true to proceed to the next step. This rule also requires an alternative flow describing what happens when the validation fails (the condition does not hold). Rules R20 and R22 are two complex rules, when compared with the others of the same rule set, because both of them require that UCS designers look at multiple steps in two different flows: the basic flow and an alternative flow.

R24 and R25 specify keywords ABORT and RESUME STEP to describe an exceptional exit action and when an alternative flow goes back to its corresponding basic flow, respectively. These two rules also specify that an alternative flow ends either with ABORT or RESUME STEP, which means that the last step of the alternative flow should clearly specify whether the flow returns back to the basic flow and where (using keywords RESUME STEP followed by a returning step number) or terminates (using keyword ABORT).

R17-R21 and R23 have been proposed in the literature and we reused them with some variation. We add R22, R24 and R25 for the purpose of making the whole set of restrictions as complete as possible so that flows of events and interactions between the basic flow and the alternatives can be clearly and concisely specified. Applying this set of rules facilitates automated NL processing (e.g., correctly parse sentences with our specified keywords) and generating of analysis models, especially sequence diagrams which also helps reducing ambiguity of UCSs [30, 33]. The detailed description of all the 26 restriction rules is provided in Appendix A.

Table 4 Restriction rules (R17-R26)

#	Description	#	Description
R17	INCLUDE USE CASE	R22	VALIDATE THAT
R18	EXTENDED BY USE CASE	R23	DO-UNTIL
R19	RFS	R24	ABORT
R20	IF-THEN-ELSE-ELSEIF-ENDIF	R25	RESUME STEP
R21	MEANWHILE	R26	Each basic flow and alternative flow should have its own postconditions.

3.3 Example

An example of use case descriptions documented with our use case template and restriction rules are presented in Table 5. The original design of the use case descriptions is from [10]. We rewrote them by applying RUCM. As show in Table 5, the use case `Withdraw Fund` contains one basic flow, one specific alternative flow, one bounded alternative flow, and one global alternative flow. The specific and bounded alternative flows correspond to four basic flow steps containing the keyword `VALIDATES THAT`. Notice that this is just one possible specification of the UCS applying our template and restrictions; it is possible to have different but equivalent specifications: for example, using the keyword `IF-THEN-ELSE-ELSEIF-ENDIF` instead of `VALIDATES THAT`.

Table 5 Use case Withdraw Fund (part 1)

Use Case Name	Withdraw Fund	
Brief Description	ATM customer withdraws a specific amount of funds from a valid bank account.	
Precondition	The system is idle. The system is displaying a Welcome message.	
Primary Actor	ATM customer	
Secondary Actors	None	
Dependency	INCLUDE USE CASE Validate PIN.	
Generalization	None	
Basic Flow	Steps	
	1	INCLUDE USE CASE Validate PIN.
	2	ATM customer selects Withdrawal.
	3	ATM customer enters the withdrawal amount.
	4	ATM customer selects the account number.
	5	The system VALIDATES THAT the account number is valid.
	6	The system VALIDATES THAT ATM customer has enough funds in the account.
	7	The system VALIDATES THAT the withdrawal amount does not exceed the daily limit of the account.
	8	The system VALIDATES THAT the ATM has enough funds.
	9	The system dispenses the cash amount.
	10	The system prints a receipt showing transaction number, transaction type, amount withdrawn, and account balance.
	11	The system ejects the ATM card.
	12	The system displays Welcome message.
	Postcondition	ATM customer funds have been withdrawn.
Specific Alternative Flows	BFS 5-7	
	1	The system displays an apology message MEANWHILE the system ejects the ATM card.
	2	The system shuts down.
	3	ABORT.
	Postcondition	ATM customer funds have not been withdrawn. The system is shut down.
Global Alternative Flows	IF ATM customer enters Cancel THEN	
	1	The system cancels the transaction MEANWHILE the system ejects the ATM card.
	2	RESUME STEP Basic Flow 12
	ENDIF	
	Postcondition	ATM customer funds have not been withdrawn. The system is idle. The system is displaying a Welcome message.
Bounded Alternative Flows	BFS 8	
	1	The system displays an apology message MEANWHILE the system ejects the ATM card.
	2	ABORT.
	Postcondition	ATM customer funds have not been withdrawn. The system is displaying an apology message.

4 EXPERIMENT PLANNING

In this section, we follow the experiment reporting template proposed in [29]. All aspects of the experiments we conducted to assess our use case template and restriction rules are described and justified.

4.1 Experiment Definition

We are interested in the applicability of the restriction rules, combined with the use case template we proposed. We refer to a use case model with UCSs that follow our restriction rules and template as a *restricted* use case model. We are also interested in the impact of a restricted use case model on the quality of analysis models that are manually derived from it (Figure 1), for instance by following standard guidelines for building analysis models (e.g., [5]). Indeed, if the restriction rules actually reduce ambiguity, then such models should exhibit higher quality.

The experiment objectives are formulated as Goal-Question-Metric (GQM) goals [3] as shown in Table 6 and Table 7. The evaluation of Goal 1 is a necessary pre-requisite to the investigation of Goal 2: we need to ensure that the restriction rules can be applied at a reasonable level of correctness. If the result of the experiment for Goal 1 shows that the restriction rules are applicable, then we can go further and investigate whether these RUCM has an impact on the quality of manually generated analysis models (class and sequence diagrams in our experiment).

Table 6 Goal 1

<i>Analyze</i>	the restriction rules
<i>for the purpose of</i>	characterizing
<i>with respect to</i>	the applicability of each restriction rule
<i>from the point of view of</i>	the requirements engineer
<i>in the context of</i>	4 th year undergraduate students defining use case models

Table 7 Goal 2

<i>Analyze for the purpose of with respect to from the point of view of in the context of</i>	the restriction rules and the use case template of RUCM Evaluating their impact on quality of derived analysis models the system analyst 4 th year undergraduate students manually deriving analysis models from use case models
---	--

Regarding Goal 1, the applicability of each restriction rule is characterized in terms of *Error Rate*, *Understandability*, *Applicability*, and *Restrictiveness*. The experiment for Goal 2 has one independent variable, namely *Method*, with two treatments corresponding to the usage or not of our restriction rules and proposed template. The alternative was to use one of the well-known use case templates and no restriction rules in the textual description of UCSs. The experiment for Goal 2 has four dependent variables, namely the quality (*Correctness*, *Completeness*, and *Redundancy*) of analysis class diagrams, the quality (*Correctness*, *Completeness*, *Redundancy*, and *Consistency* to the Boundary-Control-Entity principle [6]) of analysis sequence diagrams, the consistency between analysis class and sequence diagrams, and the correctness of responses to a comprehension questionnaire designed for this experiment.

4.2 Context Selection and Subjects

The context of both experiments is a fourth year Software Engineering course at Carleton University, Ottawa, Canada. Since this course is the one but last software engineering course in the curriculum, it is an opportunity for the students to use the skills they have acquired in at least three previous courses regarding requirements elicitation and object-oriented analysis and design. The subjects selected were 34 (the first experiment) and 39 (the second experiment) students registered in the course in 2008 and 2009, respectively.

A presentation was first provided to the students before each experiment, focusing on the restriction rules and the use case template, and how they fit into the requirements elicitation and specification phase. An assignment was designed to train the students on how to apply the restriction rules and the use case template before the experiments. The

result of the assignment was used to group the participants into two blocks and therefore ensure better homogeneity across the two groups involved in the experiment (Section 4.4) for the first experiment. In the second experiment, the participants were grouped into four blocks according to the result of the first two labs of this course, one of which was about identifying inconsistency between a use case model and its corresponding UML analysis models and the other was designed for the students to practice metamodeling using the UML class diagram notations.

Two applications, namely a Video Store (VS) system and a Car Parts Dealer (CPD) system are used as experimental materials. These two systems are of similar complexity in terms of the number of use cases in their use case models, the number of classes in their class diagrams, and the complexity of each UCS. Experimental materials must necessarily be of limited complexity since we have to consider whether the participants are able to finish the prescribed tasks, being described in Section 4.4, within two (Experiment 1) and four (Experiment 2) 3-hour long laboratory sessions.

The experiments were part of a series of compulsory laboratory exercises that were part of the course curriculum. After the experiments, the participants were asked to sign a consent form to indicate whether they allowed their laboratory results to be used for research purposes. The experiment plan had been reviewed and received clearance through the Carleton's Research Ethics Committee before collecting the consent forms. Participants who participated in the experiments signed a consent form to confirm their agreement on our using of the collected data for research purposes.

4.3 Hypotheses Formulation

In this section, we only formulate experimental hypotheses for our second research goal as the first one exclusively focuses on characterizing the ease with which developers can apply the restriction rules and does not involve a comparison. The experiment for Goal 2 (Experiment 2) has one independent variable *Method*, with two treatments: *UCM_R* and *UCM_UR*, respectively denoting the use or not of RUCM, and four dependent variables

CD , SD , CS , and QC , respectively denoting the quality of analysis class diagrams, the quality of analysis sequence diagrams, the consistency between analysis class and sequence diagrams, and the correctness of responses to a comprehension questionnaire.

Based on the above variables, we can formulate the following null hypothesis (H_0) to be tested for each dependent variable for Goal 2: there is no significant difference between UCM_R (RUCM) and UCM_UR (with the standard template and unrestricted) in terms of CD , SD , CS , and QC . The alternative hypotheses (H_a) is then two-tailed and stated as: UCM_R results in different quality or consistency of analysis models, or different correctness of responses to the comprehension questionnaire when compared to UCM_UR. Formal hypotheses are provided in Table 8.

Table 8 Hypotheses – Experiment 2

Dependent Variable	Null Hypothesis	Alternative Hypothesis
Quality of analysis class diagram	$H_0: CD(UCM_R) = CD(UCM_UR)$	$H_a: CD(UCM_R) \neq CD(UCM_UR)$
Quality of analysis sequence diagram	$H_0: SD(UCM_R) = SD(UCM_UR)$	$H_a: SD(UCM_R) \neq SD(UCM_UR)$
Consistency between analysis class and sequence diagrams	$H_0: CS(UCM_R) = CS(UCM_UR)$	$H_a: CS(UCM_R) \neq CS(UCM_UR)$
Correct response rate of the comprehension questionnaire	$H_0: QC(UCM_R) = QC(UCM_UR)$	$H_a: QC(UCM_R) \neq QC(UCM_UR)$

4.4 Experiment Design

In this section, we report the experiment design of the first experiment in Section 4.4.1, followed by the rationale for conducting a replication (the second experiment) and its experiment design (Section 4.4.2).

4.4.1 Experiment 1

The participants were asked to perform two tasks over two laboratories (3 hours each). Task 1 (activity `Define Use Case Model` in Figure 1) involved writing UCSs by applying RUCM (Section 3). In this task, the use case diagrams of the two systems, partially filled UCSs, and a comprehension questionnaire designed for evaluating the restrictions rules, were provided as input documents to the participants. Task 2 (activity

Derive Analysis Model in Figure 1) involved the construction of analysis models from two types of use case models, one of which applied RUCM (UCM_R) whereas the other only applied a standard template without restrictions (UCM_UR).

As stated previously, an assignment was designed to train the participants to apply RUCM. The assignment was essentially similar to Task 1 except that a different system was used and the participants were not monitored while they did their assignment. Individual feedbacks were given to each participant and a solution of the assignment was also provided before the experiment was conducted. Based on the grades of the assignment preceding the experiment, we defined the following three blocks: grades B to A+ (15 participants), grades B- to F (13 participants), and absent (ABS) (6 participants). As shown in Table 9, the participants were then divided into two groups: A and B. Each of the two groups was then randomly assigned participants from the three blocks in nearly identical proportions.

In Lab 1, the participants in group A were asked to complete UCSs of the VS system by applying RUCM, whereas the participants in group B did the same task on the CPD system. In Lab 2, we further divided the participants of group A into groups A1 and A2, so that the participants in A1 could derive class and sequence diagrams from the UCM_R use case model for the CPD system, while the participants in A2 did the same from the UCM_UR use case model. The same strategy was followed for group B. When assigning participants to sub-groups we followed the same blocking strategy as the one used to create groups A and B. Note that we used different systems for the two labs for each group of participants to limit learning effects that would otherwise constitute a threat to validity. For example, group A used the VS system in Lab 1 but the CPD system in Lab 2. In total, 26 data points were obtained for Task 1 and Task 2 (14 data points for treatment UCM_R and 12 for treatment UCM_UR), respectively.

Table 9 Participant groups and tasks – Experiment 1

Lab	Task	Group A		Group B		Obtained data points
		Group A1	Group A2	Group B1	Group B2	
1	Defining UCSs	VS		CPD		26

2	Deriving analysis models	CPD in UCM_R	CPD in UCM_UR	VS in UCM_UR	VS in UCM_R	UCM_R	UCM_UR
						14	12

4.4.2 Experiment Replication (Experiment 2)

In Lab 2 of Experiment 1, the participants were asked to derive both class and sequence diagrams. However, most of the participants were not able to finish sequence diagrams due to time constraints and therefore we were not able to evaluate the impact of RUCM on the quality of manually created analysis sequence diagrams. Besides, no statistically significant differences were observed for analysis class diagrams in terms of their completeness and redundancy in Experiment 1 and we thought it was perhaps also due to the time constraints. Therefore, we replicated Task 2 to 1) evaluate the impact of RUCM on the quality of analysis sequence diagrams and 2) to see whether significant differences between two treatments can be identified in terms of completeness and redundancy of generated analysis class diagrams if more time is given to participants. Task 2 was split into two tasks in Experiment 2: one was to construct an analysis class diagram (Task A) and the other was to construct analysis sequence diagrams (Task B). The participants in Experiment 2 were asked to perform Task A and Task B over two consecutive laboratories (3 hours each).

Based on the average grades of two laboratories preceding the experiment, we defined the following three blocks: grades A+ (11 participants), grades A and A- (12 participants), grades B+ – C+ (9 participants), and grades C – D- (7 participants). As shown in Table 10, the participants were then divided into four groups: G1, G2, G3, and G4. Each of the four groups was then randomly assigned participants from the four blocks in nearly identical proportions.

In Lab 1, the participants in group G1 were asked to derive analysis class diagrams from the use case model with restrictions for the CPD system, while the participants in group G2 did the same from the use case model without restrictions. The same strategy was followed for groups G3 and G4. In Lab 2, the participants were provided the same use case model as the one they were assigned in Lab 1 but they were asked to derive analysis

sequence diagrams and at the same time keep consistency between these sequence diagrams and the class diagrams they designed in Lab 1. The participants were also told to refine their class diagrams designed in Lab 1 whenever necessary. At the end of the lab, the participants were asked to submit their original class diagram from Lab 1, refined class diagram, and sequence diagrams from Lab 2. In Lab 3 and Lab 4, each participant group performed Task A and Task B with a different system and a different treatment. As a result, each group executed different combinations of treatment and system and therefore learning effects that would constitute a threat to validity were limited.

Table 10 Participant groups and tasks – Experiment 2

Lab	Task	G1	G2	G3	G4
1	Deriving class diagram (Task A)	CPD in UCM_R	CPD in UCM_UR	VS in UCM_R	VS in UCM_UR
2	Deriving sequence diagram (Task B)	CPD in UCM_R	CPD in UCM_UR	VS in UCM_R	VS in UCM_UR
3	Deriving class diagram (Task A)	VS in UCM_UR	VS in UCM_R	CPD in UCM_UR	CPD in UCM_R
4	Deriving sequence diagram (Task B)	VS in UCM_UR	VS in UCM_R	CPD in UCM_UR	CPD in UCM_R

4.5 Instrumentation

The instruments of an experiment are classified into three types: experiment objects, guidelines and measurement instruments [29]. In this section we discuss our experiment instruments by conforming to this classification.

4.5.1 Task 1

Use cases are specified by applying RUCM (Section 3.1). Inputs are the use case diagrams of the two systems, the corresponding UCSs, and a comprehension questionnaire designed to capture the participants' subjective opinions on the restrictions rules. These input documents are further discussed below. The UCSs completed by the participants and their responses to the comprehension questionnaire were collected and used to evaluate the application of each restriction rule.

4.5.1.1 Experiment objects

For each system (CPD and VS), the experiment objects are composed of a use case diagram, a set of UCSs, and a system description. These two systems were originally designed as the lab materials for the Software Engineering course (4th year undergraduate course) and have been used for several years. Each document contains a textual system description, a use case diagram along with UCSs following a standard template [5], a class diagram, and sequence diagrams for a subset of the use cases. The use case diagrams and the textual system descriptions are reused as experiment objects without making any changes. Since Task 1 requires that the participants document UCSs by applying RUCM, we provided the participants the partially filled UCSs complying with our use case template. The UCSs were partially filled and contain descriptions for only the following fields of the template (Section 3): *Use Case Name*, *Brief Description*, *Primary Actor*, *Secondary Actor*, *Dependency*, and *Generalization*. The rationale was to provide an overview of high level requirements (e.g., thanks to the brief description), ensure UCSs were consistent with the use case diagram, and let the participants focus on defining flows of events, which is the most complex part of UCSs and on which most of our restriction rules apply.

4.5.1.2 Experiment guidelines

A lab description was provided to the participants at the beginning of each lab, describing the list of documents provided, the task of the lab, and the submission guidelines. The participants were asked to complete five UCSs during the three-hour lab. The lab description also made it clear that the restriction rules had to be applied and this was a very important component of the evaluation of lab results.

4.5.1.3 Measurement instruments

An evaluation questionnaire was designed for the participants to characterize each restriction rule according to three measures: *Understandability*, *Applicability*, and *Restrictiveness*. Three questions or statements were designed to capture these three

measures on an appropriate scale (Table 11). The first question is a “yes/no” question to capture whether the participants perceived they were able to understand each restriction rule and were able to properly apply them at the time when the lab task was performed. The second statement is evaluated on a four-point Likert scale [20] question, which requires the participants to rate each restriction rule according to the extent to which they perceive it to be straightforward to apply. The third statement is also defined on a four-point Likert scale. It is used to capture the perceived restrictiveness of each restriction rule. The complete questionnaire is provided in Appendix B for reference.

Table 11 Comprehension questionnaire of Task 1

Measure	Question/Statement	Scale
<i>Understandability</i>	I understood the restriction rule and was able to properly apply it. (Question asked for each rule.)	Yes / No
<i>Applicability</i>	The restriction rule is straightforward to apply. (Question asked for each rule.)	4-point Likert scale: Completely agree, Generally agree, Generally disagree, Completely disagree
<i>Restrictiveness</i>	The restriction rule was too restrictive. (Question asked for each rule.)	4-point Likert scale: Completely agree, Generally agree, Generally disagree, Completely disagree

4.5.2 Task 2

Task 2 consists in deriving analysis models from two types of use case models: One use case model documented in RUCM (UCM_R) (e.g., groups A1 and B1 in Experiment 1—Section 4.4) whereas the other only uses a standard template [5] (UCM_UR) (e.g., groups A2 and B2). Notice that the participants were equally trained to understand our use case template and the standard template. With the provided use case models, in Experiment 1, the participants were asked to manually derive an analysis model (class and sequence diagrams) and also answer a comprehension questionnaire to assess their understanding of the use cases. In Experiment 2, the participants were asked to derive a class diagram for two different systems in Lab 1 and Lab 3 (Task A), and derive corresponding sequence diagrams and answer a comprehension questionnaire to assess their understanding of the use cases in Lab 2 and Lab 4 (Task B).

The standard template [5] of UCM_UR that we used as a comparison baseline to document UCSs shares common fields with our template: use case name, brief description, precondition, primary actor, secondary actors, dependency, and generalization. However, it is missing the following characteristics (which are specified as restriction rules in RUCM):

- It does not require that each flow of events (both basic flow and alternative flow) of a UCS contains its own postcondition.
- It does not distinguish different types of alternative flows.
- It does not require that the action steps of a UCS match one of the five interaction patterns we specified in Section 3.1.
- It does not enforce using any keyword to clearly specify interactions between the basic flow and its corresponding alternative flows, contrary to our template.

4.5.2.1 Experiment objects

The CPD and VS systems come in two versions for Task 2: they contain the same use case diagram but have different UCSs (with or without restrictions). Both sets of UCSs were specifically created for the experiment and carefully reviewed by the authors to ensure that they both contain the same amount of information though through different forms so that the participants have identical chances of creating identical analysis models from them.

4.5.2.2 Experiment guidelines

As for Task 1 (Section 4.5.1.2), in Task 2, we also provided the participants a lab description at the beginning of the lab, describing the list of documents provided, the task of the lab, and the submission guidelines. With the use case models as input documents, the participants were asked to design a class diagram. We made it clear in the lab description (Experiment 1: Lab 2, Experiment 2: Lab 1 and Lab 3) that the participants should, based on the use case description, assign meaningful names for each class,

attribute, and operation, and apply the well-established *Entity/Boundary/Control* stereotype classification [6] for each class. The participants were also asked to complete a comprehension questionnaire during the lab (Experiment 1: Lab 2, Experiment 2: Lab 2 and Lab 4), which was designed to evaluate how well they were able to understand the flow of events of each UCS. In Experiment 1, Lab 2, the participants were also asked to derive sequence diagrams for two selected use cases; however most of the participants were not able to derive (complete) diagrams due to time constraints, and we therefore decided not to analyze them. In Experiment 2, deriving sequence diagrams (Task B) was completed in Lab 2 and Lab 4. The participants working on the CPD (VS) system were asked to derive three (two) sequence diagrams for three (two) selected use cases. Participants worked on different numbers of use cases to ensure balanced tasks given the differences of the complexity of UCSs: the three CPD use cases had the similar overall complexity as the two VS use cases. We measured the complexity of a use case by simply calculating the total number of condition and action sentences contained the use case specification.

4.5.2.3 Measurement instruments

A comprehension questionnaire was designed for each system to quickly evaluate, in a repeatable and unbiased way, the extent to which a participant understood the main body (flows of events) of each UCS. To avoid introducing any bias, we ensured comprehension questions were answerable by both the participants using the restricted and unrestricted use case models.

Questions in the comprehension questionnaires are grouped per use case so that the questionnaires can be easily browsed. The complete questionnaires for the two systems are respectively provided in Appendix C and Appendix D. Each question covers either one of the following aspects of UCSs: action sequences, object responsibilities, conditions triggering actions, and general comprehension. An example question for each aspect is provided in Table 12. Each multiple-choice question includes two choices, namely “Not specified in the UCS” and “Other answer”, so that questions are at the same

time closed and open, thereby allowing us to collect complete information. (For instance, some ambiguities in UCSs may prevent the participants from selecting one of the available choices, in which case they can use either one of these two specific choices.) For each question, the participants were also asked to indicate where they found relevant information to answer the question (location). These two special choices, along with the location question, should also significantly reduce the chances that participants randomly select a choice. During data collection, we checked consistency between the selected choice and the location answer: no inconsistency was observed. For each use case, a general comprehension question was asked to determine whether the participants identified any ambiguity in the corresponding UCS. The participants' responses to this question were carefully verified to determine the validity of the data collected, but were not directly used to test our experimental hypotheses.

All multiple-choice questions contain one and only one correct answer, except for one question in the comprehension questionnaire of the VS system, where two choices are partially correct and together make up a complete, correct answer to the question. Therefore, when grading answers to this question we considered both choices as correct and we also considered "other" as correct if the correct explanation were provided. If a response to a multiple-choice question was "Not specified in the UCS", we assumed that there either exists an ambiguity in the UCS, which further leads to a low quality analysis model, or the participants were not able to correctly answer the question. Such an answer was therefore considered to be incorrect. If the response to a multiple-choice question is "Other", then the response was carefully checked to see whether it was equivalent to the correct answer.

Table 12 Classification and examples of comprehension questions

Category	Example question
Action sequences	When the part number is unknown, Sales provides an alternative part number to the system. Then what happens? 1. The system requests DBMS to check whether this alternative part number is known or unknown. 2. The system orders the part with the alternative part number for the customer. 3. The use case terminates. 4. Not specified in the use case specification. 5. Other answers: Where did you get the information from?
Object responsibilities	Who is responsible to create a pending order if the customer does not accept the alternative order provided by the system? 1. The system 2. Customer 3. Sales 4. DBMS 5. Receiving&Shipping 6. Accounting 7. Not specified in the use case specification. 8. Other answers: Where did you get the information from?
Conditions	Under which condition is use case Video Overdue invoked? (If you cannot find the answer directly from the use case specification, please indicate it. If you find the answer in the use case specification, please indicate the place.)
General comprehension question	Did you identify any place(s) in the use case specification, which causes any confusion for you to understand the use case, for example, in terms of the sequence of the steps of the basic flow or the alternative flow(s), the subjects of actions? If your answer is “yes”, please list the places and provide a brief explanation.

Questionnaire characteristics for the two systems are summarized in Table 13. In total, fifteen questions contribute to the measurement of the comprehension questionnaire for the CPD system; while twenty-five questions contribute to that of the VS system. The difference in the numbers of the questions is simply due to the different numbers of UCSs of the two systems.

Table 13 Question distributions of the comprehension questionnaires for Task 2

	CPD	VS
# of multiple-choice questions	15	23
# of non multiple-choice questions	0	2
# of general questions (not used to test our experimental hypotheses)	6	5
Total number of questions used to test our experimental hypotheses	15	25

4.6 Evaluation measurement and data collection

The goal of Task 1 is to evaluate each restriction rule based on four measures: *Error Rate*, *Understandability*, *Applicability*, and *Restrictiveness*. The goal of Task 2 is to evaluate the impact of either our restriction rules and template or a standard template on the quality of manually derived analysis models, with two dependent variables: the quality of class diagrams (abbreviated as CD), sequence diagrams (abbreviated as SD), the consistency of class and sequence diagrams (CS), and the correctness of responses to the comprehension questionnaires (abbreviated as QC). All these measures and their data collection are described next.

4.6.1 Error Rate

The *Error Rate* of a restriction rule represents the rate of improper applications of the rule, on the scale from 0 to 1. Error rates are presented in Table 15 (first 16 rules) and Table 16 (last 10 rules), and are based on raw data as measured according to metrics in Table 14. The first column of Table 15 and Table 16 lists the restriction rule number. The second column describes the formulas used to calculate the *Error Rate* of each rule. In Table 14, the first column provides the names of the metrics and the second column specifies the metrics or their calculations. The variable $N_{m,r}$ specific to the formulas (Table 16, Column 2) for rule r (one of the rules R17-R25) is defined in Table 16, Column 3.

Table 14 Measures used to derive Error Rate

Measure	Specification (a single UCS)
N_{v_r}	# of violations of a specific restriction rule r
N_{missed_r}	# of instances where the keyword (defined as the restriction rule r) should be applied, but is not
N_{ASteps}	Total number of action steps (sentences)
N_{Cond}	Total number of condition sentences: precondition, postcondition, IF/ELSEIF condition, VALIDATES THAT condition, and UNTIL condition
$N_{Include}$	Total number of INCLUDE USE CASE sentences
N_{Extend}	Total number of EXTENDED BY USE CASE sentences
$N_{AltFlows}$	Total number of alternative flows
N_{NP}	Total number of noun phrases
N_{IF}	Total number of IF-THEN-ELSE-ELSEIF-ENDIF conditional logic sentences
N_{ABORT}	Total number of ABORT sentences
N_{RESUME}	Total number of RESUME STEP # sentences
$N_{MEANWHILE}$	Total number of MEANWHILE sentences
N_{VALTS}	Total number of VALIDATES THAT sentences
$N_{DO-UNTIL}$	Total number of DO-UNTIL sentences
N_{RFS}	Total number of the places where the keyword RFS is applied
N_{Spe}	$=N_{ABORT}+N_{RESUME}$
N_{Dep}	$=N_{Include}+N_{Extend}$
$N_{NormalA}$	$=N_{ASteps}-N_{Spe}-N_{Dep}$
$N_{NormalS}$	$=N_{NormalA}+N_{Cond}$
N_{AllS}	$=N_{ASteps}+N_{Cond}-N_{VALTS}$

The *Error Rate* of each restriction rule equals (Table 15) the number of violations of the rule r (N_{v_r}) divided by the total number of steps where the rule could be applied, i.e., either $N_{NormalA}$, N_{ASteps} , N_{NP} , or $N_{NormalS}$. For example, R1 is specified as “The subjects of sentences in basic and alternative flows should be the system or actors”. It puts restriction on action steps, except those describing use case relationships (i.e., those that use keywords INCLUDE and EXTEND) and those specifying what happens at the end of an alternative flow (i.e., those that use keywords ABORT and RESUME). Therefore the error rate of R1 for a UCS equals $N_{v_1}/N_{NormalA}$, where N_{v_1} is the number of violations of R1 in the UCS and $N_{NormalA}$ (Table 14) is the total number of action steps (N_{ASteps}) of the UCS that do not contain keywords ABORT, RESUME, INCLUDE USE CASE and EXTENDED BY USE CASE (we subtract N_{Spe} and N_{Dep} from N_{ASteps} in Table 14). Some restriction rules require two error rate measures. For example (Table 16), the error rate of R17 is calculated as the average of N_{missed_17}/N_{AllS} and $N_{m_17}/N_{Include}$, where N_{missed_17}/N_{AllS} captures the rate of absence of keyword INCLUDE USE CASE (specified in R17), i.e., it

should be used but it is not, whereas $N_{m_17}/N_{Include}$ captures the rate of erroneous occurrences (the keyword is used, but either in a wrong situation, without the included use case name, or with an incorrect use case name, or should not be applied at all).

Table 15 Error Rate measurements (R1–R16)

#	Measure	#	Measure	#	Measure	#	Measure
R1	$N_{v_1}/N_{NormalA}$	R5	$N_{v_5}/N_{NormalA}$	R9	N_{v_9}/N_{NP}	R13	$N_{v_13}/N_{NormalS}$
R2	N_{v_2}/N_{ASteps}	R6	$N_{v_6}/N_{NormalA}$	R10	$N_{v_10}/N_{NormalS}$	R14	N_{v_14}/N_{NP}
R3	$N_{v_3}/N_{NormalA}$	R7	$N_{v_7}/N_{NormalA}$	R11	$N_{v_11}/N_{NormalS}$	R15	$N_{v_15}/N_{NormalS}$
R4	N_{v_4}/N_{ASteps}	R8	$N_{v_8}/N_{NormalS}$	R12	$N_{v_12}/N_{NormalS}$	R16	N_{v_16}/N_{NP}

Table 16 Error Rate measurements (R17–R26)

#	Measure	Description (N_{m_r})—see Table 4 for the corresponding rules/keywords
R17	$(N_{missed_17}/N_{AllS} + N_{m_17}/N_{Include})/2$	# of instances where the keyword is applied, but either in a wrong situation, without the included use case name, or with incorrect use case name, or should not be applied at all.
R18	$(N_{missed_18}/N_{AllS} + N_{m_18}/N_{Extend})/2$	# of instances where the keyword is applied, but either in a wrong situation, without the extended use case name, or with incorrect use case name.
R19	$(N_{missed_19}/N_{AltFlows} + N_{m_19}/N_{RFS})/2$	# of instances where the keyword is applied, but either in a wrong situation, without the basic flow step number followed, or with a wrong basic flow step number.
R20	N_{m_20}/N_{IF}	# of instances where the keyword is incorrectly applied, such as incorrect grammar.
R21	$(N_{missed_21}/N_{AllS} + N_{m_21}/N_{MEANWHILE})/2$	# of instances where the keyword is applied, but in a wrong situation, or the action steps connected by the keyword are not appropriate (e.g., non-action steps).
R22	N_{m_22}/N_{VALTS}	# of instances where the alternative case is not described in its corresponding alternative flow, the condition sentence is not a complete sentence, or there is no condition sentence followed by the keyword.
R23	$(N_{missed_23}/N_{AllS} + N_{m_23}/N_{DO-UNTIL})/2$	# of instances where the keyword is applied, but in a wrong situation, or the grammar is not followed: e.g., missing condition.
R24	$(N_{missed_24}/N_{AllS} + N_{m_24}/N_{ABORT})/2$	# of instances where the keyword is applied in a wrong place: e.g., not the last step of an alternative flow.
R25	$(N_{missed_25}/N_{AllS} + N_{m_25}/N_{RESUME})/2$	# of instances where the keyword is applied, but not in an alternative flow, or the keyword is applied in an alternative flow, but not in the last step of an alternative flow.
R26	$N_{v_26}/(N_{AltFlows}+1)$	$N_{AltFlows}+1$: # of the flows of events of a UCS.

For each UCS, the error rate of a specific restriction rule r ($ErrorRate_r$) is:

$$ErrorRate_r = \frac{\sum_{s=1}^{|S|} \left(\sum_{u_s=1}^{|U_s|} ErrorRate_{u_s} \right)}{U \times |S|}$$

where S is the set of participants (s is an index identifying each participant), U_s is the set of UCSs created by participant s (u_s is an index identifying each UCSs created by participant s), U denotes the total number of UCSs written by all the participant.

4.6.2 Understandability, Applicability and Restrictiveness

Understandability is one of the three subjective measures used to assess the restriction rules and is based on responses to the first (yes/no) question of the comprehension questionnaire of Task 1 (Section 4.5.1.3). This measure, normalized on the scale from 0 to 1, is the ratio of “yes” in all collected responses. The other two subjective measures, *Applicability* and *Restrictiveness*, are measured by using the participant responses to the second and third questions of the comprehension questionnaire of Task 1 (Section 4.5.1.3), respectively. Recall that both questions are answered on four-point Likert scales, from 1 (Completely disagree) to 4 (Completely agree). Our analysis will focus on comparing average scores across all participant responses.

4.6.3 Quality of analysis class diagram (CD)

The quality of an analysis class diagram is evaluated from three aspects: *Correctness*, *Completeness*, and *Redundancy*. The reference class diagrams, used as the basis to evaluate class diagrams designed by the participants, are the original class diagrams of the two case study systems we used in Task 2 (Section 4.5.1.1). Data are collected for the first five measures in Table 17 from the reference class diagrams (e.g., number of classes in each reference class diagram (N_{class})); while data are also collected for the last 10 measures in Table 17 from the class diagram of each participant. All these data are then used to compute the measures of *Completeness*, *Correctness* and *Redundancy* of a participant class diagram according to the formulas presented in Table 19. The completeness of a class diagram ($R_{CDcompl}$) is computed as the average of Class

Completeness (R_{clp1}), Association Completeness (R_{clp2}) and Generalization Completeness (R_{clp3}) since we consider classes, associations and generalizations to be the three most important (structural) aspects of a class diagram. The class diagram correctness ($R_{CDcorrect}$) is determined by the correctness of classes (R_{Ci})—computed as the average, over the complete class diagram, of the class measures of *Completeness* (R_{Ccompl}) and *Correctness* ($R_{Ccorrect}$) (Table 18)—and associations. The class diagram redundancy (R_{CDre}) is computed as the ratio of redundant classes (N_{re}) over all the classes of a participant’s class diagram (N_{class}).

Table 17 Measures used to derive CD

#	Measure	Specification
1	N_{class}	# of classes in the reference model
2	N_{asso}	# of associations in the reference model
3	N_{gen}	# of generalizations in the reference model
4	N_{attr}	# of attributes in entity classes in the reference model
5	N_{opr}	# of operations of control or boundary classes in the reference model
6	N_{cp1}	# of missing attributes of a class
7	N_{cp2}	# of missing operations of a class
8	N_{clp1}	# of missing classes in a class diagram
9	N_{clp2}	# of missing associations in a class diagram
10	N_{clp3}	# of missing generalizations in a class diagram
11	N_c	# of matching classes in a class diagram
12	N_{eclass}	# of classes in a class diagram
13	N_{clr2}	# of incorrect associations of a class diagram
14	N_a	# of matching associations between matching class diagrams
15	N_{re}	# of extra classes that are redundant, excluding equivalent model elements

For each class of the reference class diagram of a case study system, we look for a class with the same name in a participant class diagram. If such a matching class is found, then it is evaluated according to the quality measures for a class (Table 18); otherwise, we keep looking for a design equivalent³ to the reference class in the participant class diagram. If no such equivalent design exists, then we consider the reference class as missing and therefore the participant diagram as incomplete. When all the reference classes have been looked at, we obtain three outputs: 1) A set of matching classes which are evaluated by using the quality measures for a class (Table 18); 2) A set of equivalent

³ An equivalent design may contain one or more model elements, which could be attributes, multiple classes connected by associations, etc.

designs, which are not measured because this would require either a subjective measurement or a large number of specific measures. Besides not many such equivalent designs have been found and not measuring them does not really impact the measurement of CD; 3) A set of reference classes, missing in the participant class diagram. A procedure similar to this identification of matching classes, missing classes, and equivalent class designs is also applied to identify matching/missing/equivalent attributes, operations, associations, and generalizations. Each participant class diagram is evaluated by applying the quality measures for a class diagram (Table 19).

Table 18 Quality measures for a class

Category	Measure	Formula
Completeness (R_{Ccomplt})	Missing stereotype (R _{cp1})	Entity classes: $R_{Ccomplt} = 1 - (R_{cp1} + R_{cp2})/2$ Boundary classes: $R_{Ccomplt} = 1 - (R_{cp1} + R_{cp3})/2$
	Missing attributes $R_{cp2} = N_{cp1} / N_{atr}$	
	Missing operations $R_{cp3} = N_{cp2} / N_{opr}$	
Correctness (R_{Ccorrect})	Incorrectly named (R _{cr1})	Entity classes: $R_{Ccorrect} = 1 - (R_{cr1} + R_{cr2} + R_{cr3} + R_{cr4} + R_{cr5} + R_{cr6})/6$ Boundary classes: $R_{Ccorrect} = 1 - (R_{cr1} + R_{cr2} + R_{cr3} + R_{cr5} + R_{cr6})/5$ If the entity class under evaluation does not contain any attributes, then: $R_{Ccorrect} = 1 - (R_{cr1} + R_{cr2} + R_{cr3} + R_{cr5})/4$ If the boundary class under evaluation does not contain any operations, then: $R_{Ccorrect} = 1 - (R_{cr1} + R_{cr2} + R_{cr3} + R_{cr6})/4$
	Incorrectly stereotyped (R _{cr2})	
	Incorrectly assigned "abstract" (R _{cr3})	
	Does not represent one and only one logical concept (R _{cr4})	
	Not given a cohesive set of responsibilities (R _{cr5})	
	Does not represent the intended meaning of the class (R _{cr6})	
Class i (R_{ci})	$R_{ci} = (R_{Ccomplt} + R_{Ccorrect})/2$	

Table 19 Quality measures for a class diagram

Category	Measure	Formula
Completeness (R_{CDcomplt})	Class Completeness $R_{clp1} = 1 - N_{clp1} / N_{class}$	$R_{CDcomplt} = (R_{clp1} + R_{clp2} + R_{clp3})/3$
	Association Completeness $R_{clp2} = 1 - N_{clp2} / N_{asso}$	
	Generalization Completeness $R_{clp3} = 1 - N_{clp3} / N_{gen}$	
Correctness (R_{CDcorrect})	Class Correctness $R_{clr1} = 1 - \sum_{i=1}^{N_c} R_{ci} / N_c$	$R_{CDcorrect} = (R_{clr1} + R_{clr2})/2$
	Association Correctness	

	$R_{ctr2} = N_{ctr2} / N_a$	
Redundancy (R_{CDre})	$R_{CDre} = N_{re} / N_{eclass}$	

4.6.4 Quality of analysis sequence diagram (SD) and consistency of analysis class and sequence diagrams (CS)

Five measures evaluate the quality of a sequence diagram (SD): *SD Completeness*, *SD Correctness*, *SD Redundancy*, *BCE Consistency* (consistency with the Boundary/Control/Entity principle), and *SDCD Consistency*. They are defined (Table 21) based on simpler measures (Table 20). To evaluate the quality of sequence diagrams produced by participants we need reference sequence diagrams. Unfortunately we do not find sequence diagrams for the CPD and VS systems in reference textbooks or other resources. We only possess sequence diagrams manually created by the authors of this article. However, we can use sequence diagrams generated by the CASE tool aToucan [30], which automatically generates analysis class and sequence diagrams from RUCM requirements, since these diagrams have been shown to be of high quality, the quality of renowned textbooks [30]. Data are collected from the reference sequence diagrams to compute the first four measures in Table 20; while data are collected from the sequence diagrams being evaluated for the last 14 measures.

As opposed to the first four aspects, which compare a participant's sequence diagram to a reference sequence diagram, *SDCD Consistency* ($R_{conSDCD}$) measures the consistency between a participant sequence diagram and this participant's class diagram. Any violation of the following three rules contributes to $N_{inconSDCD-t}$, which determines *SDCD Consistency* (Table 21):

- Each object in a sequence diagram must be an instance of a class in the class diagram;
- Each operation, attribute in a sequence diagram must appear in the class diagram and must be consistent with it (e.g., an operation in a message must be declared in the class that is the type of the target lifeline);

- Two objects can communicate with messages provided there is an association between their respective classes.

As shown in Table 21, the *completeness* of a sequence diagram ($R_{SDcompl}$) is computed as the average of the *Message Completeness* (R_{conMsg}), *IU (InteractionUse) Completeness* (R_{conIU}), and *CF (CombinedFragment) Completeness* (R_{conCF}). The consistency of a sequence diagram ($R_{SDcorrect}$) is determined by the consistency of messages (R_{corMsg}), interaction uses (R_{corIU}), combined fragments (R_{corCF}), and message ordering (R_{corSeq}) since these are considered to be important aspects in a sequence diagram. The redundancy of a sequence diagram (R_{SDre}) is computed as the ratio of redundant messages ($N_{reMsg-t}$) over all the messages of a student's sequence diagram (N_{msg-t}). Measure *BCE Consistency* (R_{conBCE}) evaluates whether a student's sequence diagram conforms to the Boundary-Control-Entity (BCE) design principle of analysis sequence diagrams [6]. Any violation of the following three rules is considered to be a violation of the BCE principle and therefore contributes to measure N_{vBCE-t} (Table 20):

- A message is sent from a Boundary object to an Entity object.
- A message is sent from an Entity object to a Control object.
- A message is sent from an Entity object to a Boundary object.

Table 20 Measures used to derive SD

#	Measure	Specification
1	N_{msg-r}	# of messages in the reference sequence diagram
2	$N_{lifeline-r}$	# of lifelines in the reference sequence diagram
3	N_{IU-r}	# of interaction uses in the reference sequence diagram
4	N_{CF-r}	# of combined fragments in the reference sequence diagram
5	N_{msg-t}	# of messages in a tested sequence diagram
6	$N_{lifeline-t}$	# of lifelines in a tested sequence diagram
7	N_{IU-t}	# of interaction uses in a tested sequence diagram
8	N_{CF-t}	# of combined fragments in a tested (participant) sequence diagram
9	N_{mMsg-t}	# of missing messages in a tested sequence diagram
10	N_{mIU-t}	# of missing interaction uses in a tested sequence diagram
11	N_{mCF-t}	# of missing combined fragments in a tested sequence diagram
12	N_{iMsg-t}	# of occurrences of incorrect messages (i.e., incorrect message name, incorrect lifelines, wrong direction, and/or incorrect message types) in a tested sequence diagram
13	N_{iIU-t}	# of occurrences of incorrectly-applied interaction uses in a tested sequence diagram
14	N_{iCF-t}	# of occurrences of incorrectly-applied combined fragments 1. InteractionOperand is not provided or a wrong operand is used.

		2. Condition is not provided or a wrong condition is used. 3. Lifelines are not correctly covered. 4. Wrong messages are covered.
15	N_{iSeq-t}	# of occurrences of incorrect sequences of messages, interaction uses, or combined fragments.
16	N_{vBCE-t}	# of violations of the Boundary-Control-Entity (BCE) principle
17	$N_{reMsg-t}$	# of extra messages that are redundant; the semantics of these messages have been modeled by other elements
18	$N_{inconSDCD-t}$	# of occurrences of inconsistency between a tested sequence diagram and its corresponding class diagram

Table 21 Quality measures for a sequence diagram

Category	Measure	Formula
Completeness ($R_{SDcomplt}$)	Message Completeness $R_{conMsg} = 1 - \frac{N_{mMsg-t}}{N_{msg-r} - N_{uMsg-r} - N_{aaMsg-r}}$	$R_{SDcomplt} = \frac{R_{conMsg} + R_{conIU} + R_{conCF}}{3}$
	IU Completeness $R_{conIU} = 1 - N_{mIU-t} / (N_{IU-t})$	
	CF Completeness $R_{conCF} = 1 - N_{mCF-t} / (N_{CF-t})$	
Correctness ($R_{SDcorrect}$)	Message Correctness $R_{corMsg} = 1 - \frac{N_{iMsg-t} + N_{aaMsg-t}}{N_{msg-t}}$	$R_{SDcorrect} = \frac{R_{corMsg} + R_{corIU} + R_{corCF} + R_{corSeq}}{4}$
	IU Correctness $R_{corIU} = 1 - N_{iIU-t} / (N_{IU-t})$	
	CF Correctness $R_{corCF} = 1 - N_{iCF-t} / (N_{CF-t})$	
	Message Sequence Correctness $R_{corSeq} = 1 - N_{iSeq-t} / N_{msg-t}$	
Redundancy (R_{SDre})	$R_{SDre} = N_{reMsg-t} / N_{msg-t}$	
BCE Consistency (R_{conBCE})	$R_{conBCE} = 1 - N_{vBCE-t} / N_{mMsg-t}$	
SDCD Consistency ($R_{conSDCD}$)	$R_{conSDCD} = 1 - N_{inconSDCD-t} / (N_{msg-t} + N_{lifeline-t})$	

4.6.5 Correctness of responses to the comprehension questionnaires (QC)

As discussed in Section 4.5.2.3, data about the correctness of responses to the questions of the comprehension questionnaires of Task 2 (QC), are used to evaluate the understandability of UCSs, which is normalized between 0 and 1:

For the CPD system: $QC_{CPD} = \text{number of correct responses} / 15$

For the VS system: $QC_{VS} = \text{number of correct responses} / 25$

where the denominators are the total numbers of questions in each system questionnaire.

5 EXPERIMENT RESULTS AND ANALYSIS

Recall that our first goal (Section 4.1) is to assess each restriction rule with respect to four measures: *Error Rate*, *Understandability*, *Applicability*, and *Restrictiveness*, whereas the second goal (Section 4.1) is to evaluate the impact of the restriction rules and use case template on the quality of manually derived analysis models, with four dependent variables: *CD*, *SD*, *CS*, and *QC*, respectively denoting the quality and consistency of analysis class and sequence diagrams manually generated by participants and the correctness of participants responses to a comprehension questionnaire.

In Experiment 1, among the collected 26 data points for Task 2 (Table 9), one participant's result was not included in the analysis because it was very incomplete (no class diagram was derived), thus suggesting he had not complied with instructions. One participant missed the lab for Task 2 and performed the task at home in an uncontrolled manner. We excluded this data point as well. We also excluded a data point from a participant who spent significantly more than the planned three hours (3 hours 40 mins) to finish Task 2. As a result, a total of 23 data points were used for analyzing Task 2 (14 data points for treatment UCM_R and 9 for treatment UCM_UR). Though the two systems used for the experiment may lead to different results, the number of observations does not allow us to perform a separate analysis for each of them. We, however, counter-balance their possible effect by ensuring a similar proportion of observations coming from each system, for each of the tasks.

The collected and used data points in Experiment 2 are provided in Table 22. As it is often the case in controlled experiments, unforeseen, difficult to control events lead to the exclusion of invalid data. Some observations in each lab and group were left out of the analysis because of one of the following reasons. 1) Three participants (one in G2 and two in G4) did not want to sign the consent form and therefore their data points were excluded; 2) One participant's result (Lab 2-G1) was very incomplete (no meaningful messages were derived) suggesting he did not follow instructions; 3) Seven participants' results (one in Lab 2-G1, two in Lab 2-G2, one in Lab 2-G3, one in Lab 2-G4, one in Lab

4-G1, and one in Lab 4-G2) contain only one or two sequence diagrams (two or three sequence diagrams are required); 4) Four participants (two in Lab 1-G1, one in Lab 1-G4, and two in Lab 3-G4) spent significantly less than the planned three hours (less than 2 hours) to finish Task A; 5) Two participants in G1 and G4 missed Lab 3 and Lab 2, respectively, and performed the task at home in an uncontrolled manner; 6) One participant (Lab 3-G3) derived a class diagram which is extremely similar to the reference class diagram (e.g., identical class, attribute, and operation names); 7) One participant (G3) derived a single sequence diagram for three use cases of the CPD system in Lab 4.

Table 22 Collected and analyzed data points – Experiment 2

Lab	Collected data points				Analyzed data points			
	G1	G2	G3	G4	G1	G2	G3	G4
1	10	10	9	10	8	9	9	7
2	10	10	9	8	8	7	8	6
3	10	9	9	10	9	9	8	6
4	9	9	10	8	8	8	9	6

5.1 Usage of restriction rules

Each restriction rule is evaluated by four measures: *Error Rate*, *Understandability*, *Applicability*, and *Restrictiveness* (Section 4.6). The last three measure participants' subjective opinions on rules; while *Error Rate* objectively evaluates errors made by participants when applying each rule to UCSs during Experiment 1, Task 1. In this section, experiment results are first discussed individually in terms of each measure. Then, we synthesize these observations and try to identify correlations between the measures. This analysis is motivated by the expectation that a rule that is easy to understand and apply should be less restrictive and have a lower error rate. Last, we provide suggestions regarding future training for applying restriction rules.

5.1.1 Error rate

Twelve of the 26 restriction rules have an error rate above or equal to 5%: Figure 2. Six other rules have nearly no errors. All 26 error rates are provided in Appendix E, Table 40

for reference. Notice that the highest error rate is 25% (R22), and that all but two rules (R20 and R22) have an error rate under 15%. If we now look more closely at R20 and R22, we notice that R20 and R22 are more complex than the others (Section 3.2). R20 contains three different cases, which specify how keywords are used to describe conditional logic sentences (IF-THEN-ELSE-ELSEIF-ENDIF) within a flow or across a reference flow and an alternative flow (e.g., IF-THEN appears in the reference flow while ELSE-ENDIF appears in the alternative flow). We observed that the most frequently occurring errors regarding R20 involve not applying or incorrectly applying the required keywords (e.g., missing ELSE in alternative flow). R22 is also a composite and complex restriction rule, which not only specifies the usage of keyword VALIDATES THAT, but also indicates that the alternative case of the condition checking sentence containing this keyword must be described in an alternative flow.

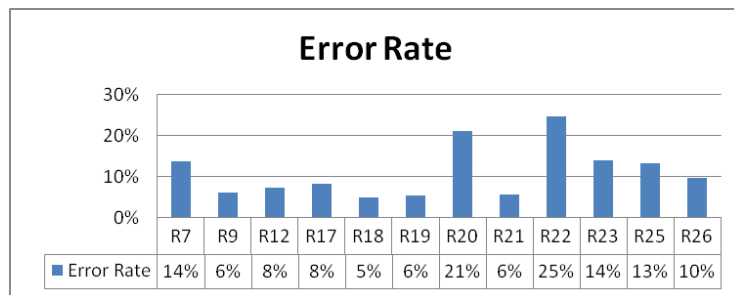


Figure 2 Error rates of restriction rules (when $\geq 5\%$)

From a more general standpoint, we can also observe from Figure 2 that most of the restrictions on the use of natural language (R1-R16) have error rates lower than the restrictions on the use of control structures specified as keywords (R17-R25). Appropriate tool support (part of our future work) can very likely be used to enforce the proper usage of keywords specified in the latter set of rules, thus potentially reducing their error rates. We also believe that more extensive training, perhaps specifically focused on error-prone rules, could further reduce error rates.

5.1.2 Understandability

Recall that *Understandability* of a restriction rule reflects the ease of understanding the rule according to the subjective opinions of the participants. Figure 3 presents the 11 (out of 26) rules that score below 90%. All 26 *Understandability* scores are provided in Appendix E, Table 41 for reference. The lowest score is 65% (R10 and R15). It is worth mentioning that R8, R10, R13, and R15 rely on concepts from the natural language domain (e.g., “declarative sentence”, “modal verb”, “negatively adverb”, and “participle phrase”), which could probably explain why these four restriction rules have relatively low *Understandability* (from 65% to 75%). It is our experience that computer engineering participants in Canada have in general a limited understanding of grammatical and natural language concepts, an issue that may also extend to many IT professionals. This suggests that in the future we probably need to put more efforts on these rules during training. Another possible explanation could be that the motivation and rationale of these restriction rules is not clear to the participants since some of them (e.g., R12-R15) are designed to facilitate automated natural language processing, an aspect of the study that the participants were not informed about.

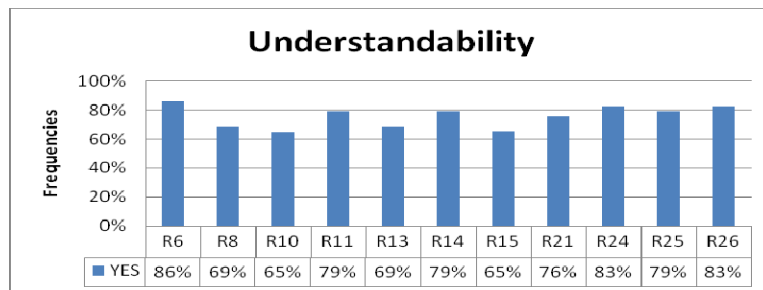


Figure 3 Understandability of the restriction rules (when <90%)

In general, we can see that the participants have received sufficient training before the experiment so that they were able to understand most of the restriction rules to a sufficient degree. Therefore, we are confident that the participants who were given use case models with restrictions in Task 2 are capable to understand the use case models and derive analysis models from them.

5.1.3 Applicability

Recall that *Applicability* is measured on a four-point Likert scale, from 1 (Completely disagree) to 4 (Completely agree). The frequencies of the participants’ responses on this scale are analyzed. Figure 4 presents the percentage of responses with “agree” scores on *Applicability*, showing only rules with a percentage below 90% (12 out of 26 restriction rules). The scores for the 26 restriction rules are provided in Appendix E, Table 42, for reference.

Figure 4 shows that most of the participants (80% and above) agree that most of the restriction rules (21 rules) are easy to apply, except for rules R8, R10, R11, R13 and R15, which receive less than 80% “agree” responses. Notice that these rules also receive relatively low *Understandability* scores (Figure 3). This probably suggests that these rules are relatively difficult to understand and, as a result, they are also relatively hard to apply. As we have discussed in Section 5.1.2, better training and/or tool support could help improve these scores.

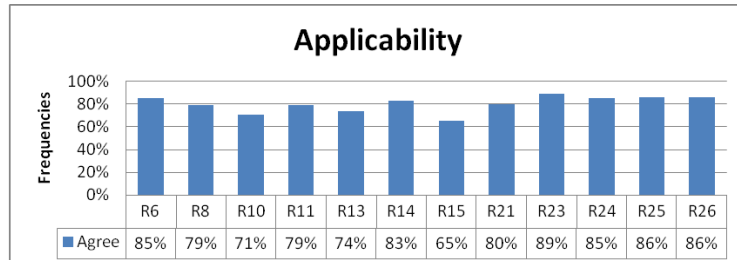


Figure 4 High applicability of the restriction rules (when < 90%)

5.1.4 Restrictiveness

Restrictiveness is also measured on a four-point Likert scale from 1 (Completely disagree) to 4 (Completely agree). Figure 5 presents the percentage of responses with “agree” scores on *Restrictiveness*, showing only the 10 (out of 26) rules receiving scores above 20%. The scores for all the 26 restriction rules are provided in Appendix E, Table 43, for reference. As shown in Figure 5, most of the participants (80% and above) agree that more than half of the restriction rules (16) are not restrictive. Though the 10 rules shown

in Figure 5 all received a score above 20%, all scores are below 40% and the remaining 16 rules were below this threshold. This is not bad considering that the participants applied the restriction rules for the first time and were not informed of the rationale of the restriction rules when the experiment was conducted. In other words, if the participants knew what the rules were for, they would consider them less restrictive.

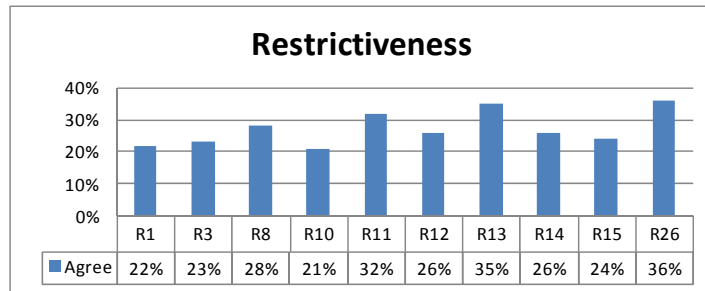


Figure 5 High restrictiveness of the restriction rules (when > 20%)

5.1.5 Correlation Analysis

Let us analyze the correlations between the measures of *Understandability*, *Restrictiveness*, *Applicability*, and *Error Rate*. We expect that a rule that is easy to understand and apply should be less restrictive and have a lower error rate. Spearman nonparametric correlations between each pair of the four measures are presented in Table 23. The measures of *Understandability*, *Applicability*, and *Restrictiveness* are strongly related to each other (significant correlations are identified), which means that most rules with high *Understandability* consistently have high *Applicability* and low *Restrictiveness*. However, no significant correlations between *Error Rate* and the other three measures can be identified. One possible explanation is that the participants were consistent when answering questionnaires but some of them did not gain enough insight to precisely answer those questionnaires according to their own experience of using the restriction rules: some of the participants that perceived some rules to be understandable, easy to apply and not restrictive committed mistakes when applying them while others did not.

Table 23 Spearman nonparametric correlation analysis – correlation table

Measure	By Measure	Spearman ρ	Prob> ρ
---------	------------	-----------------	---------------

Low Restrictiveness	Understandability	0.5420	0.0042
High Applicability	Understandability	0.7894	<0.0001
High Applicability	Low Restrictiveness	0.5972	0.0013
Error Rate	Understandability	0.2930	0.1463
Error Rate	Low Restrictiveness	0.2262	0.2665
Error Rate	High Applicability	0.1299	0.5272

5.2 Quality of analysis models

As we have discussed in Section 4.3, Goal 2 involves one independent variable (*Method*) with two treatments, *UCM_R* and *UCM_UR*, respectively denoting the use or not of RUCM, and four dependent variables *CD*, *SD*, *CS*, and *QC*, respectively denoting the quality of analysis class diagrams, the quality of analysis sequence diagrams, the consistency between analysis class and sequence diagrams, and the correctness of responses to a comprehension questionnaire (Sections 4.6.3 and 4.6.4). In this section, we discuss the impact of the independent variable *Method* on the dependent variable *CD* (Section 5.2.1), *SD* (Section 5.2.2), *CS* (Section 5.2.3), and *QC* (Section 5.2.4) and also investigate the possible interactions between *Method* and two factors (*System* and *Order*) on these four dependent variables.

5.2.1 Quality of analysis class diagram (CD)

This section discusses the impact of *Method* (two treatments: *UCM_R* vs. *UCM_UR*) on *CD* (the quality of analysis class diagrams), which is determined by several measures (Section 4.6.3). Next, we investigate possible interactions between *Method* and a number of factors on *CD*: *System* (*CPD* vs. *VS*) and *Order* (2.1⁴ vs. 2.3). The descriptive statistics of all the experiments regarding *CD* are provided in Table 44, Appendix E for reference.

One first observation is that all means for *Association Completeness*, for all experiments, are below 0.25. This indicates that less than 25% of the associations in the reference class diagrams were derived by the participants. (Recall that the reference class diagrams of

⁴ In the rest of the paper, we use 2.1, 2.2, 2.3, and 2.4 to denote the first, second, third and fourth labs of Experiment 2, respectively.

CPD and VS contain 15 and 10 associations, respectively.) Furthermore, many participants were not even able to derive any of the reference associations. In such situations, *Association Correctness*, which is defined as the ratio of correctly identified associations to all identified associations and also contributes to the calculation of *CD Correctness*, is then undefined due to a division by zero. Therefore, we exclude *Association Correctness* and *CD Correctness* from our analysis.

In the rest of the section, we first report univariate analysis results for *CD* on the following four measures: *Class Completeness*, *CD Completeness*, *Class Correctness*, and *Redundancy*. Then interaction effects between *Method* and *System* and between *Method* and *Order* are discussed in Section 5.2.1.2. In Section 5.2.1.3, we summarize the analysis results.

5.2.1.1 Univariate analysis

Figure 6 to Figure 9 show, for the four quality measures of *CD*, the mean diamonds plots of Experiment 1 and Experiment 2. This visual representation helps compare the means of the different measures for treatments UCM_R and UCM_UR. A mean diamond depicts the sample mean and 95% confidence interval. The line across each diamond pair represents the group mean and the vertical span of each diamond represents the 95% confidence interval. Notice that though the systems used for the experiment might lead to different results, the number of observations does not allow us to perform a separate analysis for each of them in Experiment 1. Additionally, for the purpose of comparison, the mean diamonds of the four measures in Experiment 2 (without differentiating systems and labs) are also provided in these figures.

As shown in these figures, the participants applying UCM_R consistently performed better than the participants applying UCM_UR in terms of all the four measures in both Experiment 1 and Experiment 2. The results of a two-tailed *t*-test to assess the statistical significance, for each system, of the difference between the two treatments are discussed below.

By comparing the global means for the two experiments, we can notice that the results of Experiment 2 are overall better than those of Experiment 1 regarding *Class Completeness*, *CD Completeness*, and *Class Correctness*. However it is the opposite for *Redundancy*; Experiment 2 yielded higher mean values than Experiment 1. As discussed in Section 4.4, participants in Experiment 2 were given twice as much time to derive class diagrams and we were expecting to obtain more complete class diagrams and, as a result, more redundant classes.

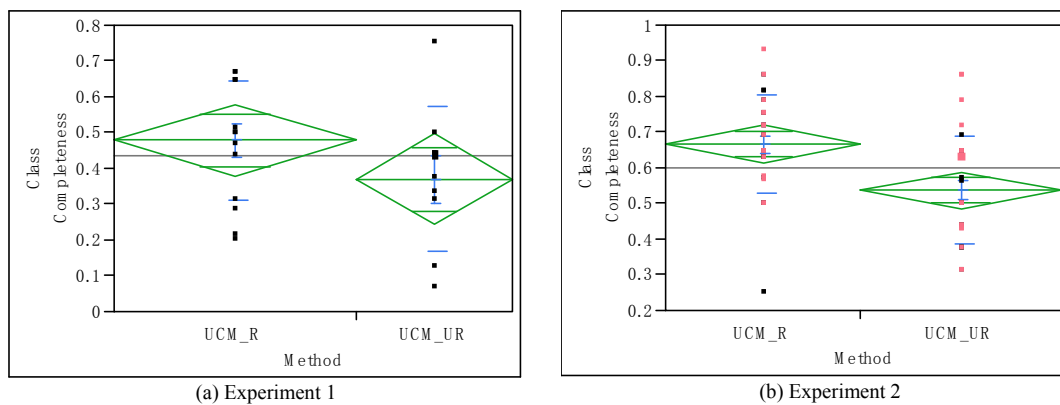


Figure 6 Distributions of Class Completeness - Labs 1&2

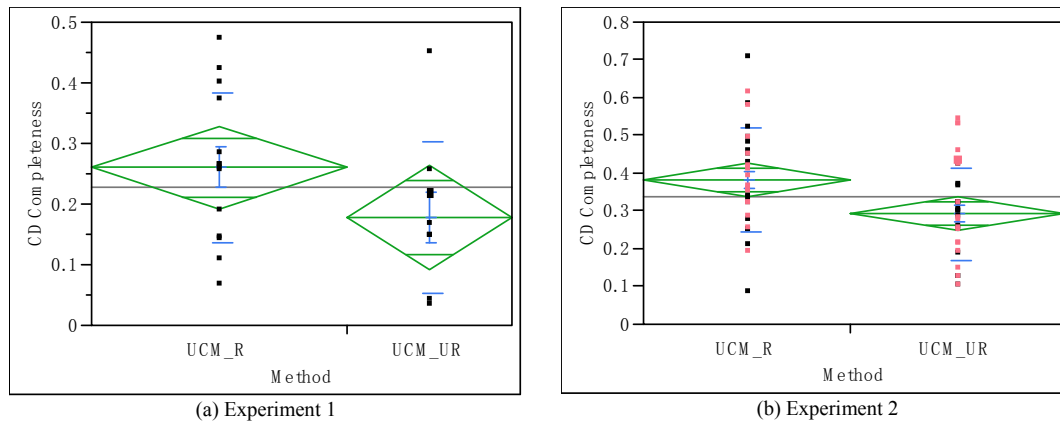


Figure 7 Distributions of CD Completeness - Labs 1&2

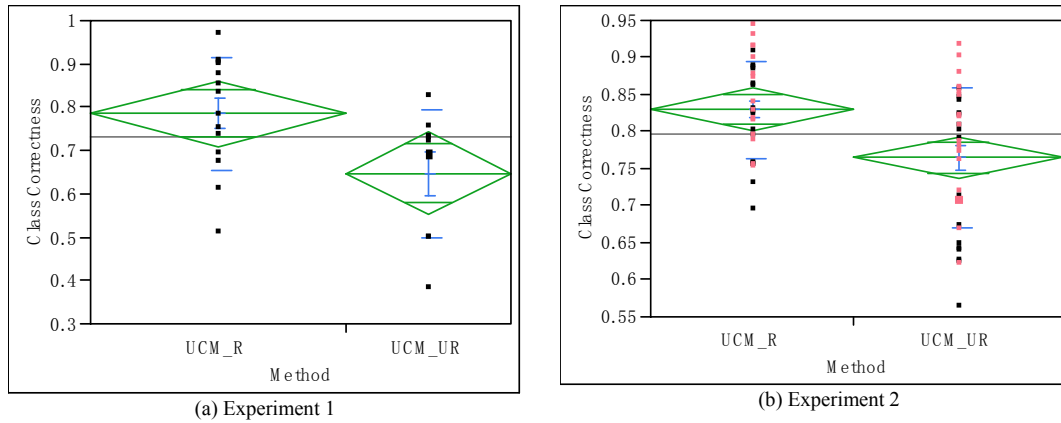


Figure 8 Distributions of Class Correctness - Labs 1&2

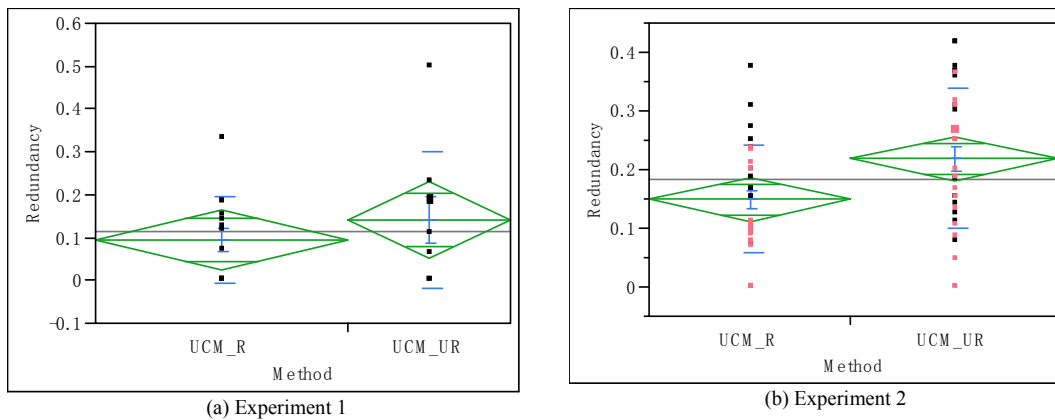


Figure 9 Distributions of Redundancy - Labs 1&2

Figure 10 to Figure 13 show mean diamonds of the four measures of lab 2.1. Figure 14 to Figure 17 show mean diamonds of the four measures of lab 2.3. In each of these figures, the mean diamonds for each system are provided for the purpose of comparison.

We notice from these figures that the participants applying UCM_R performed better than those applying UCM_UR, for both systems in both labs 2.1 and 2.3. Also, the mean values of the VS system are consistently higher (*Class Completeness*, *CD Completeness* and *Class Correctness*) or lower (*Redundancy*) than those of the CPD system. Therefore we conducted a two-way ANOVA analysis (Section 5.2.1.2) to assess the impact of *System* (CPD vs. VS) on *CD*.

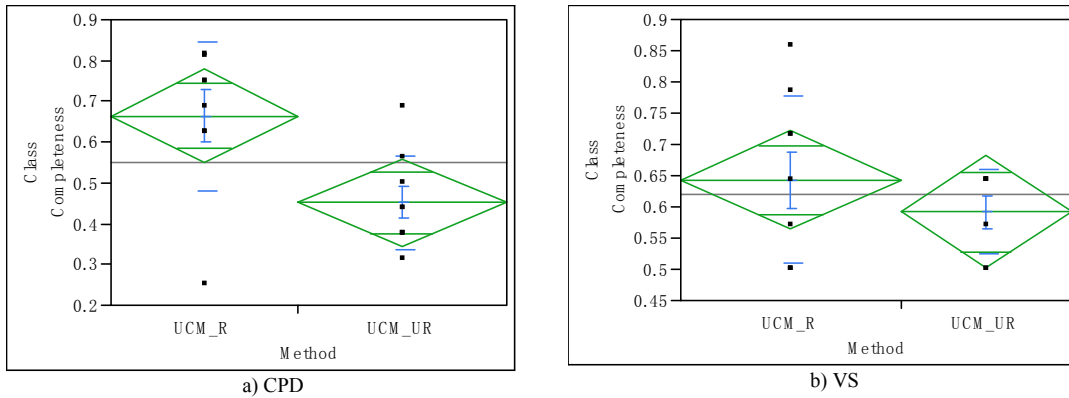


Figure 10 Distributions of Class Completeness - Lab 2.1

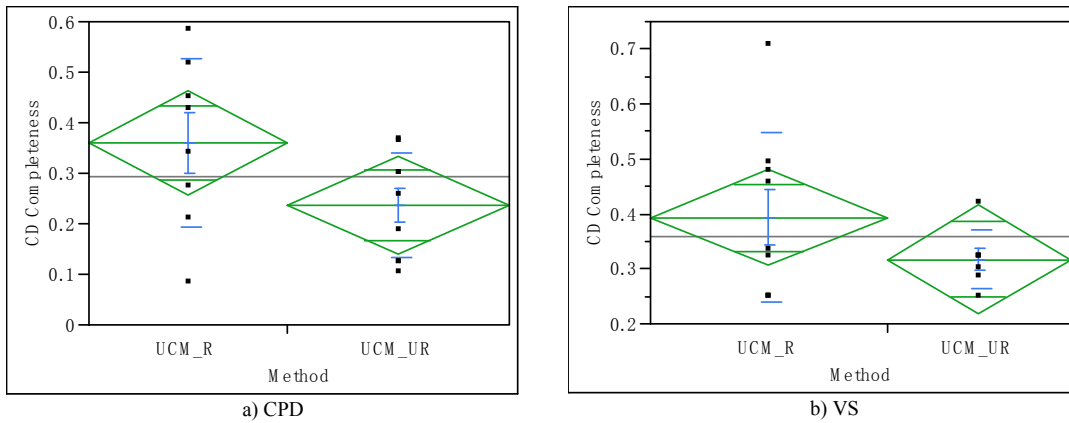


Figure 11 Distributions of CD Completeness - Lab 2.1

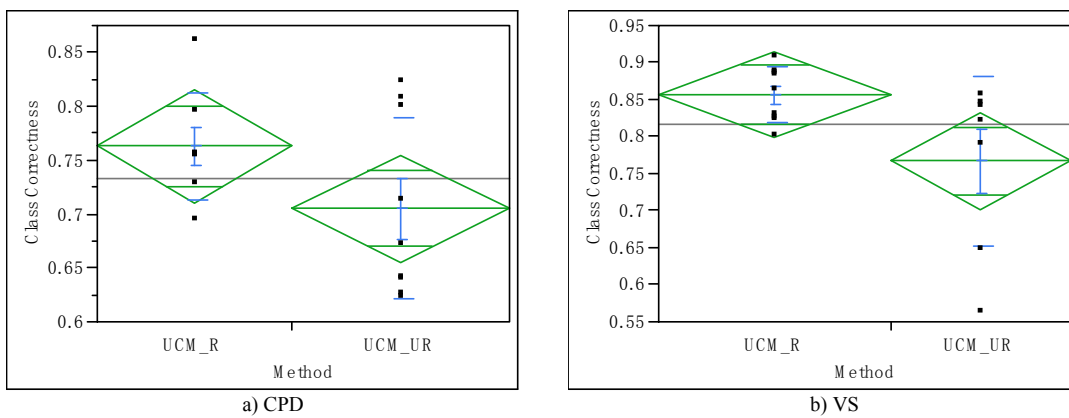


Figure 12 Distributions of Class Correctness - Lab 2.1

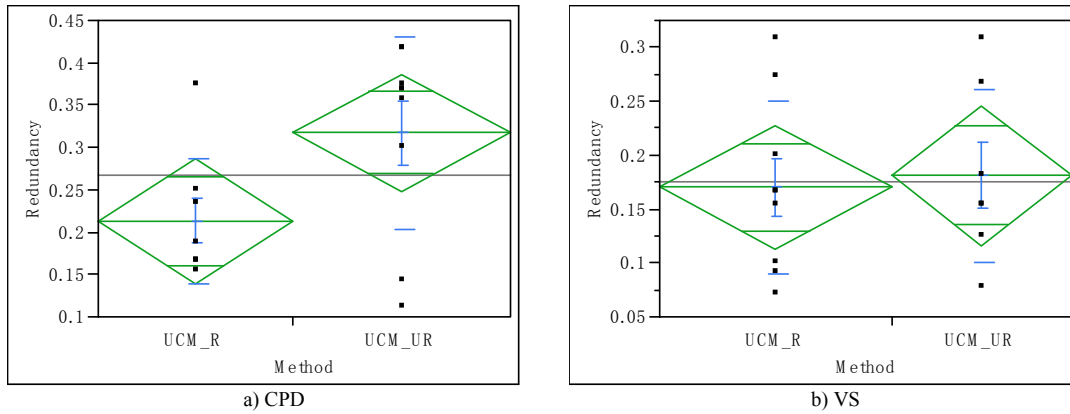


Figure 13 Distributions of Redundancy - Lab 2.1

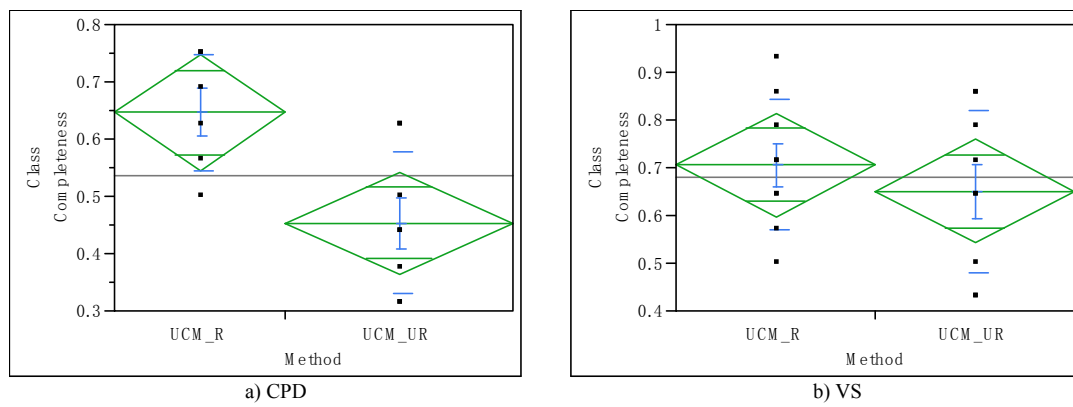


Figure 14 Distributions of Class Completeness - Lab 2.3

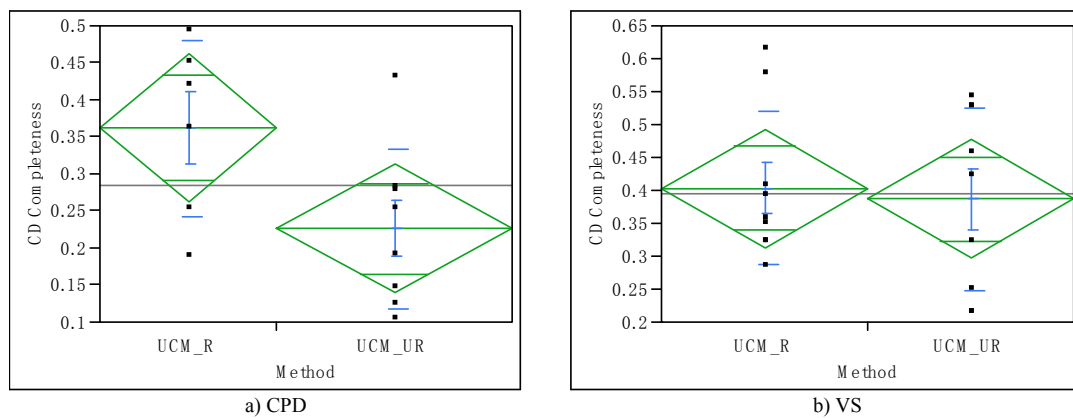


Figure 15 Distributions of CD Completeness - Lab 2.3

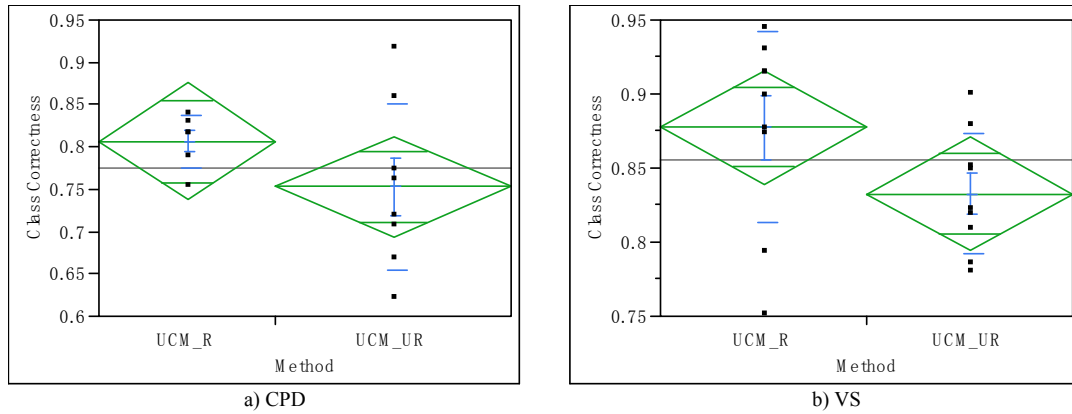


Figure 16 Distributions of Class Correctness - Lab 2.3

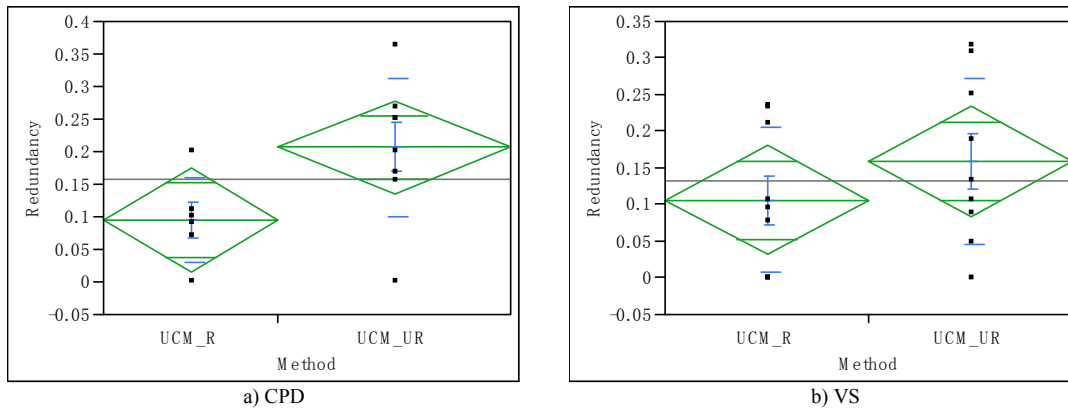


Figure 17 Distributions of Redundancy - Lab 2.3

Table 24 reports on the results of a two-tailed *t*-test to assess the statistical significance of differences of the four CD quality measures between UCM_R and UCM_UR. Equivalent non-parametric tests results (Wilcoxon rank sum test [29]) are also reported to confirm the validity of *t*-test results. Each row reports on each *CD* quality measure for each system (CPD or VS) or the two systems together (CPD+VS) for Experiment 1 (due to the small number of observations). Columns show the mean differences, degree of freedom (DF), the calculated *t*-value, and the corresponding *p*-value of the two-tailed *t*-test and corresponding probability for the non-parametric test.

Though the differences are not significant, the participants using UCM_R performed slightly better in terms of *Class Completeness*, *CD Completeness*, and *Redundancy* than

those using UCM_UR. However, there is a statistically significant difference regarding *Class Correctness*: UCM_R yielded significantly higher quality class diagrams than UCM_UR.

In lab 2.1, UCM_R are better in terms of all four measures than UCM_UR. There are statistically significant differences regarding *Class Completeness* and *Class Correctness*. There is, however, a conflict between the results of the *t*-test and the non-parametric Wilcoxon test regarding *CD Completeness*. Various statistically-significant differences are observed for CPD only *t*-tests and no statistically-significant differences are observed for VS only *t*-tests, which indicates that it is necessary to perform a two-way ANOVA test (Section 5.2.1.2) to assess interaction effects between *System* (CPD vs. VS) and *Method* (UCM_R vs. UCM_UR).

In lab 2.3, again, UCM_R outperformed UCM_UR in terms of all the four measures. Statistically significant differences are observed in terms of *Class Completeness*, *Class Correctness*, and *Redundancy*, which is not consistent with the results of lab 2.1. This inconsistency is due perhaps to interaction effects between *Order* (2.1 vs. 2.3) and *Method* (UCM_R vs. UCM_UR). A two-way ANOVA test (Section 5.2.1.2) was performed to examine the impact of lab order on the dependent variable *CD*.

Table 24 two tailed *t*-test and Wilcoxon test– CD

Exp	System	Measures	<i>t</i> -test				Wilcoxon test
			Mean difference (UCM_R – UCM_UR)	DF	t-value	p-value	Prob> Z
1	CPD+VS	Class Completeness	0.108	15	1.334	0.2022	0.1848
		CD Completeness	0.082	17	1.552	0.139	0.1564
		Class Correctness	0.138	16	2.281	0.037	0.0298
		Redundancy	-0.048	12	-0.792	0.4436	0.5372
2.1	CPD	Class Completeness	0.213	12	2.827	0.0157	0.0203
		CD Completeness	0.124	11	1.813	0.0961	0.1356
		Class Correctness	0.058	13	1.757	0.1022	0.1937
		Redundancy	-0.105	14	-2.269	0.0398	0.1472
	VS	Class Completeness	0.051	12	0.99	0.3411	0.4775
		CD Completeness	0.076	10	1.384	0.1956	0.39
		Class Correctness	0.09	7	1.986	0.0875	0.0567
		Redundancy	-0.011	13	0.27	0.7914	0.9576
	CPD+VS	Class Completeness	0.14	30	2.933	0.0064	0.0066
		CD Completeness	0.106	26	2.392	0.0242	0.0582
		Class Correctness	0.08	25	2.73	0.0114	0.021
		Redundancy	-0.068	26	-1.904	0.0682	0.2122
2.3	CPD	Class Completeness	0.193	12	3.188	0.008	0.0382
		CD Completeness	0.136	10	2.203	0.0514	0.0814
		Class Correctness	0.054	9	1.476	0.0175	0.22
		Redundancy	-0.111	12	-2.423	0.0326	0.0446
	VS	Class Completeness	0.056	15	0.768	0.454	0.532
		CD Completeness	0.017	16	0.277	0.7854	0.7904
		Class Correctness	0.045	13	1.765	0.1002	0.0934
		Redundancy	-0.053	16	-1.059	0.3056	0.2866
	CPD+VS	Class Completeness	0.124	29	2.326	0.0274	0.0344
		CD Completeness	0.076	30	1.634	0.1128	0.1214
		Class Correctness	0.054	30	2.112	0.0432	0.0892
		Redundancy	-0.08	30	-2.323	0.0272	0.0338

5.2.1.2 Interaction effects

The relationship between the dependent variable *CD* and *Method* may be affected by a number of interaction factors. A Two-way Analysis of Variance (ANOVA) was performed to test both the interaction between *Method* and *Order* and between *Method* and *System* for Experiment 2: Appendix F, Table 46.

For lab 2.1, the results show significant main effects of *Method* on all the four measures (in favor of UCM_R), significant main effects of *System* on *Class Correctness* and *Redundancy* (in favor of VS) and no significant interaction effects on any of the four

measures. For lab 2.3, the results show significant main effects of *Method* on *Class Completeness*, *Class Correctness*, and *Redundancy* (in favor of UCM_R). Significant main effects of *System* are observed in lab 2.3 for *Class Completeness*, *CD Completeness*, and *Class Correctness* (in favor of VS). No significant interaction effects are identified in lab 2.3 for any of the four measures. Least-square means plots are used to visualize the main and interaction effects of *Method* and *System* in lab 2.1 and lab 2.3: Appendix F, Figure 28 and Figure 29.

Lab order was the second factor for which we studied the interaction effect on *CD* to account for learning effects. We assume that participants would tend to perform better on lab 2.3 than lab 2.1, regardless of other factors. We performed a two-way ANOVA analysis to assess the impact of lab order on the four measures of CD: Appendix F, Table 47. Statistically significant differences are found for the main effects of *Order* in terms of *Class Correctness* and *Redundancy* (in favor of Lab 2.3), but no significant effect was observed for the interaction of *Method* and *Order*. We also observed that the participants performed better in Lab 2.3 than in Lab 2.1 in terms of all the four measures (least-square means plots are provided in Appendix F, Figure 30 for reference).

5.2.1.3 Summary

The participants applying UCM_R consistently performed better than the participants applying UCM_UR in terms of all four-quality measures in both Experiment 1 and Experiment 2. Statistically significant differences (in favor of UCM_R) are summarized in Table 25, when combining observations from the two systems to gain statistical power. “Yes” means statistically significant difference is identified.

Table 25 Summary of *t*-test results of CD

Experiment	Class Completeness	CD Completeness	Class Correctness	Redundancy
1	No	No	Yes	No
2.1	Yes	Yes	Yes	No
2.3	Yes	No	Yes	Yes

We also observed that the participants in Experiment 2 outperformed the participants in Experiment 1 for both treatments with respect to *Class Completeness*, *CD Completeness*, and *Class Correctness*, but not *Redundancy*. Recall that the participants in Experiment 2 were given more time to derive class diagrams and we were expecting to obtain more complete and correct class diagrams and, as a result, more redundant classes (Section 4.4).

A Two-way ANOVA was performed to test both the interaction between *Method* and *System* and between *Method* and *Order* for Experiment 2. No significant interaction effects were identified for any of the four measures. However, significant main effects of *System* and *Order* were found on various measures. Results are summarized in Table 26. Participants performed better on VS than on CPD perhaps because the use case model of VS contains five use cases while the use case model of CPD has six use cases, the latter thus requiring more time to understand.

Table 26 Summary of ANOVA tests - CD

Main effects	Experiment	Class Completeness	CD Completeness	Class Correctness	Redundancy
<i>System</i> (in favor of VS)	2.1	No	No	Yes	Yes
	2.3	Yes	Yes	Yes	No
<i>Order</i> (in favor of Lab 2.3)		No	No	Yes	Yes

5.2.2 Quality of analysis sequence diagram (SD)

In this section, we discuss the impact of *Method* (with two treatments *UCM_R* vs. *UCM_UR*) on the dependent variable *SD*, which is determined by different measures (Section 4.6.4). Next, we investigate the possible interactions between *Method* and possible interaction factors: *System* (*CPD* vs. *VS*) and *Order* (2.2 vs. 2.4). Descriptive statistics for *SD* are provided in Table 45, Appendix E for reference.

Recall, from Section 4.6.4, that a sequence diagram is evaluated in terms of a set of measures: *Completeness*, *Correctness*, *Redundancy*, and *BCE Consistency*. The completeness of a sequence diagram is calculated as the average of the completeness of the messages, interaction uses, and combined fragments contained in the sequence

diagram. The correctness of the sequence diagram is evaluated as the average of the correctness of the messages, interaction uses, and combined fragments of the sequence diagram. In the rest of the analysis, we report on a selected set of these measures: *Message Completeness*, *SD Completeness*, *Message Correctness*, *SD Correctness*, and *BCE Consistency*. The rationale is that *Message* is one of the most important elements of sequence diagrams and *SD Completeness* and *SD Correctness* indicate the overall completeness and correctness of a sequence diagram, including the completeness and correctness of its interaction uses and combined fragments. Therefore it is not necessary to report on the completeness and correctness of interaction uses and combined fragments separately. We also exclude *Redundancy* from the analysis of sequence diagrams as we observed very few sequence diagrams that contained redundant messages.

In the rest of the section, we first report univariate analysis results for *SD* (Section 5.2.2.1) on the selected five measures discussed above. Then interaction effects between *Method* and *System* and between *Method* and *Order* are discussed in Section 5.2.2.2. In Section 5.2.2.3, we summarize and discuss the analysis results.

5.2.2.1 Univariate analysis

We performed univariate analysis to assess the isolated effect of *Method* on the five selected quality measures for *SD*: *Message Completeness*, *SD Completeness*, *Message Correctness*, *SD Correctness*, and *BCE Consistency*. Figure 18 to Figure 22 show mean diamonds for these measures in lab 2.2. Mean diamonds for all measures in lab 2.4 are shown in Figure 23 to Figure 27. In each of these figures, results are shown for each system to help comparisons.

We notice from these figures that UCM_R are better than UCM_UR in terms of most of the measures and for both systems in both lab 2.2 and lab 2.4.

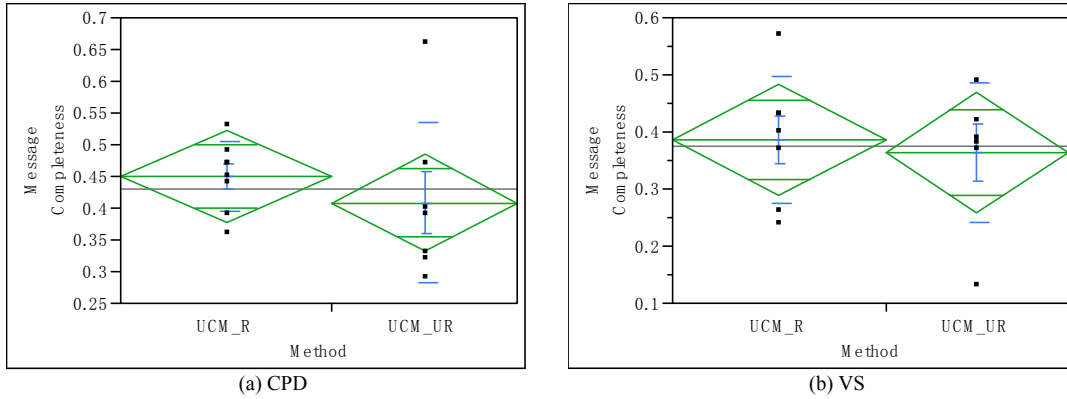


Figure 18 Distributions of Message Completeness - Lab 2.2

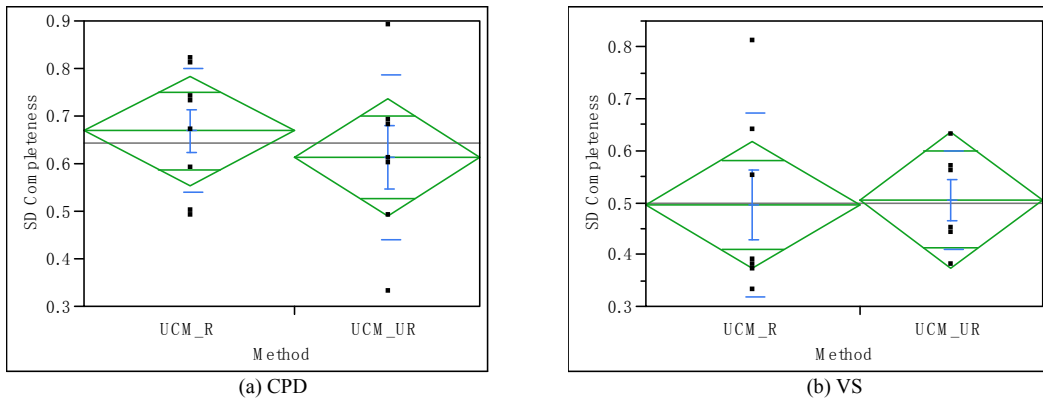


Figure 19 Distributions of SD Completeness - Lab 2.2

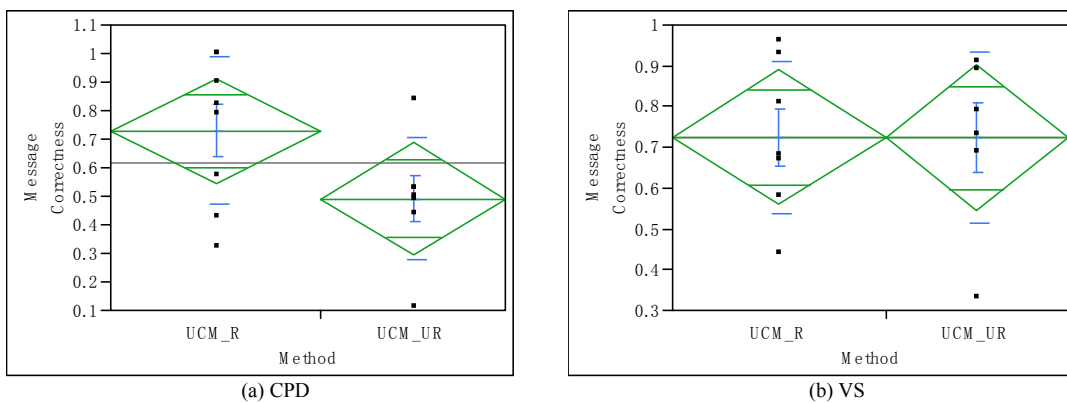


Figure 20 Distributions of Message Correctness - Lab 2.2

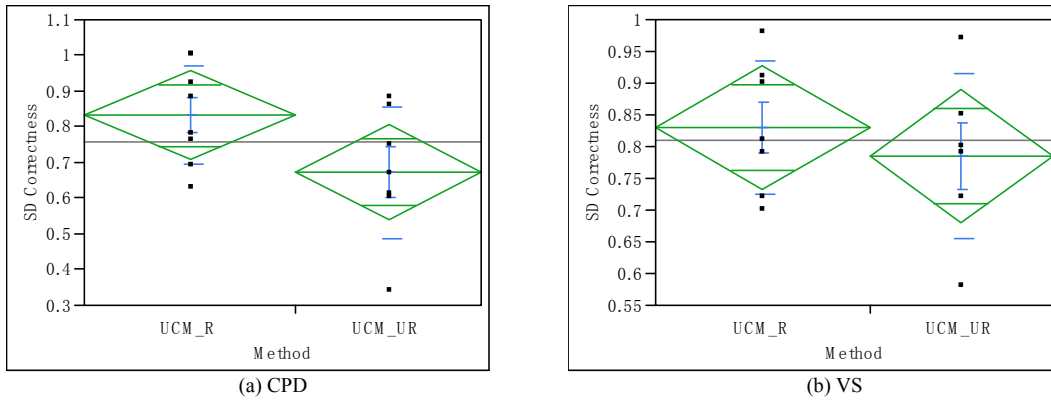


Figure 21 Distributions of SD Correctness - Lab 2.2

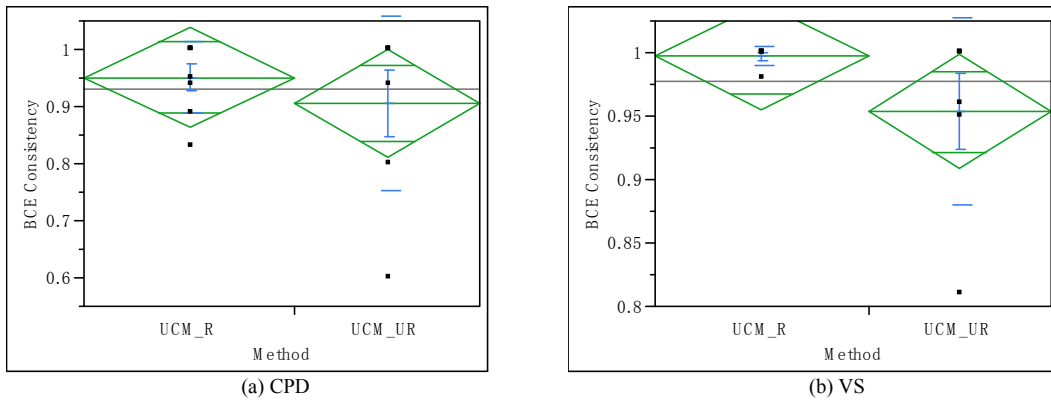


Figure 22 Distributions of BCE Consistency - Lab 2.2

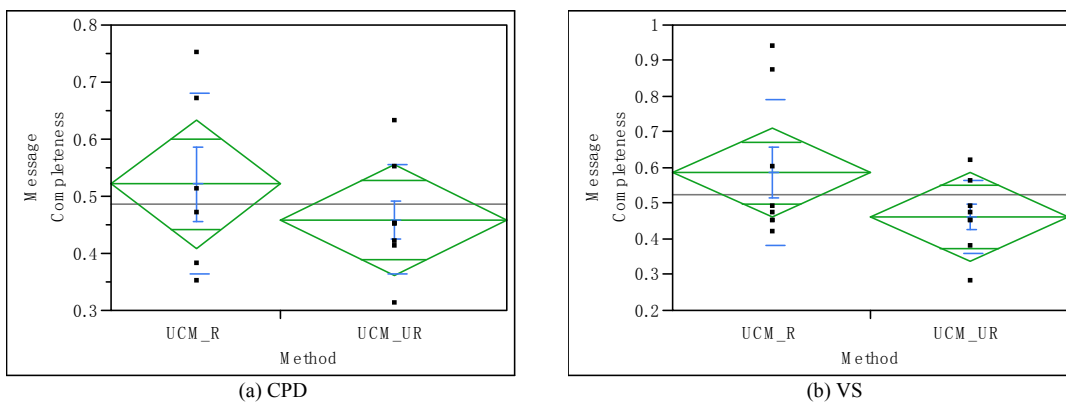


Figure 23 Distributions of Message Completeness - Lab 2.4

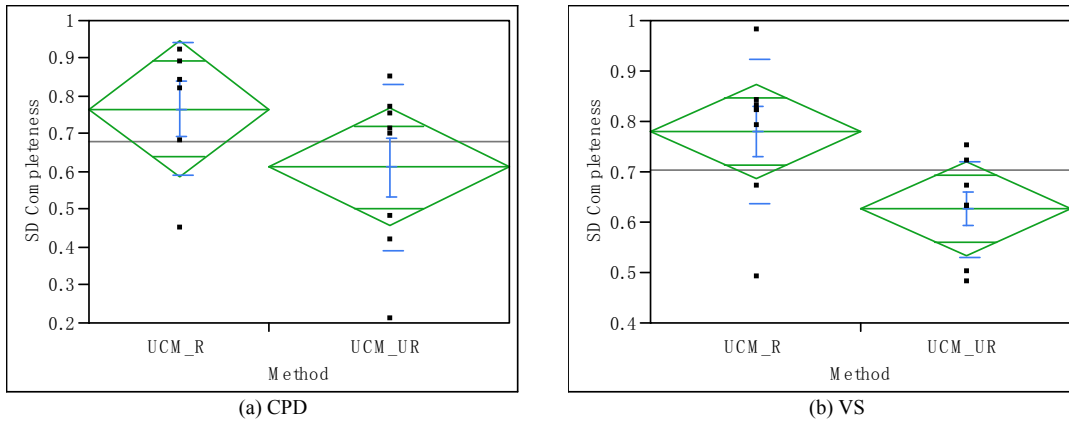


Figure 24 Distributions of SD Completeness - Lab 2.4

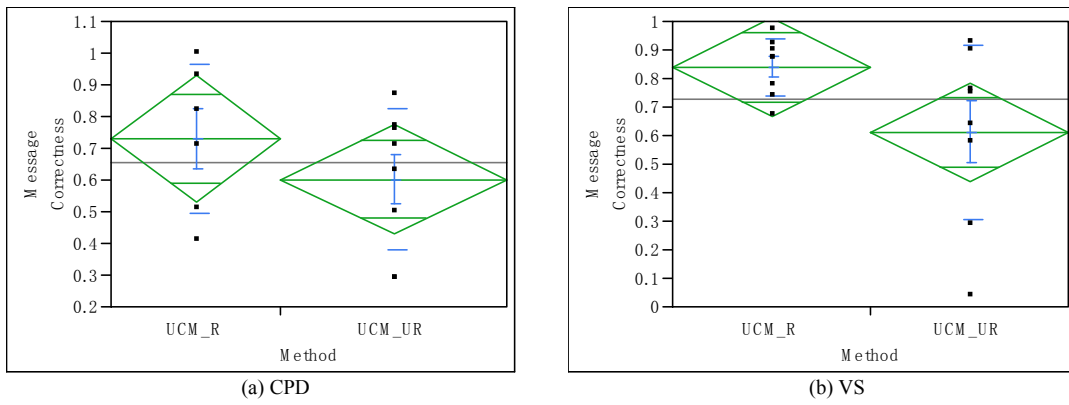


Figure 25 Distributions of Message Correctness - Lab 2.4

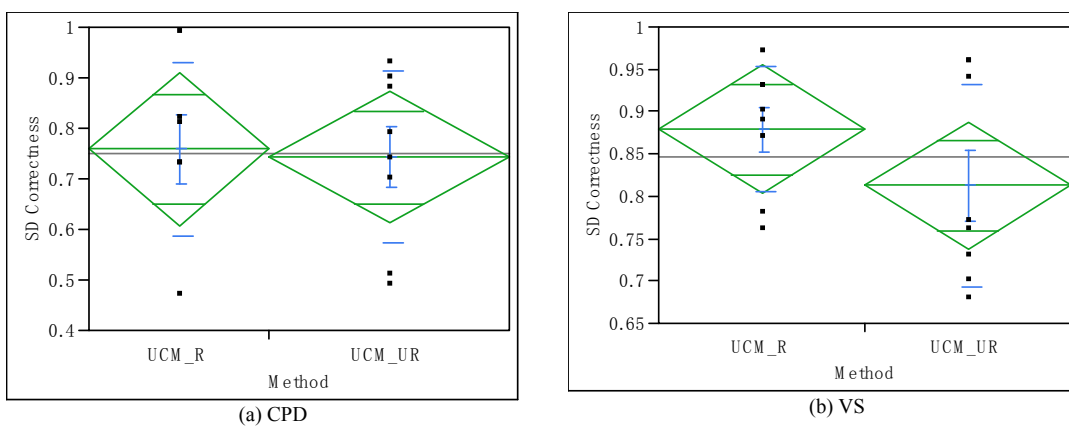


Figure 26 Distributions of SD Correctness - Lab 2.4

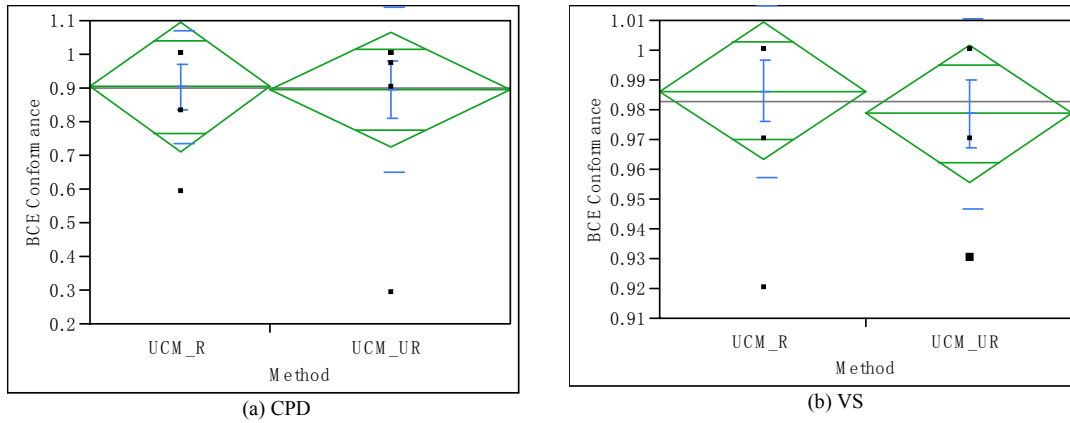


Figure 27 Distributions of BCE Consistency - Lab 2.4

Table 27 reports on the results of a two-tailed t -test for each system to assess the statistical significance of the difference for the five selected quality measures between UCM_R and UCM_UR. Non-parametric tests (Wilcoxon rank sum test [29]) were also performed to double-check the validity of t -test results.

In lab 2.2, there is no statistically significant difference observed. In lab 2.4, statistically significant differences regarding *SD Completeness* and *Message Correctness* are found based on the CPD+VS data points.

Table 27 one tailed *t*-test and Wilcoxon test– SD

Lab	System	<i>t</i> -test					Wilcoxon test
		Measures	Mean difference (UCM_R – UCM_UR)	DF	t-value	p-value	Prob> Z
2.2	CPD	Message Completeness	0.041	8	0.8061	0.4438	0.201
		SD Completeness	0.056	11	0.6955	0.5012	0.5624
		Message Correctness	0.237	13	1.9451	0.0738	0.2948
		SD Correctness	0.156	11	1.872	0.0878	0.0636
		BCE Consistency	0.046	8	0.7322	0.4856	0.9
	VS	Message Completeness	0.022	10	0.3419	0.7392	0.7746
		SD Completeness	-0.009	9	-0.1193	0.9076	0.5672
		Message Correctness	0.001	10	0.0085	0.9834	0.9432
		SD Correctness	0.045	10	0.6804	0.5124	0.6162
		BCE Consistency	0.044	5	1.45	0.2056	0.158
	CPD+VS	Message Completeness	0.032	22	0.792	0.437	0.2584
		SD Completeness	0.025	26	0.4094	0.6856	0.7952
		Message Correctness	0.128	25	1.4766	0.1524	0.1814
		SD Correctness	0.107	22	1.9275	0.0672	0.076
		BCE Consistency	0.045	16	1.247	0.2306	0.3742
2.4	CPD	Message Completeness	0.063	8	0.8596	0.416	0.5736
		SD Completeness	0.155	12	1.4738	0.1666	0.1752
		Message Correctness	0.128	11	1.0345	0.324	0.3006
		SD Correctness	0.016	11	0.1726	0.8662	1
		BCE Consistency	0.008	12	0.0751	0.9514	1
	VS	Message Completeness	0.124	10	1.5246	0.1572	0.3678
		SD Completeness	0.154	12	2.5168	0.0268	0.0204
		Message Correctness	0.229	8	2.0051	0.0778	0.1146
		SD Correctness	0.066	12	1.3277	0.2098	0.3436
		BCE Consistency	0.008	14	0.495	0.6286	0.7012
	CPD+VS	Message Completeness	0.098	19	1.798	0.088	0.1546
		SD Completeness	0.156	28	2.7023	0.0116	0.0076
		Message Correctness	0.186	26	2.3462	0.0268	0.0304
		SD Correctness	0.05	28	0.97	0.3404	0.3078
		BCE Consistency	0.014	26	0.259	0.7978	0.7433

5.2.2.2 Interaction effects

We performed a two-way Analysis of Variance (ANOVA) to assess possible interactions between *Method* and *Order* and between *Method* and *System*. ANOVA results showing the main and interaction effects of *Method* and *System* on the five *SD* quality measures are reported in Table 48 (Appendix F). For lab 2.2, the results show a significant main effect only for *System* on *SD Correctness*. For lab 2.4, the results show significant main

effects of *Method* on *SD Completeness* and *Message Correctness*. No significant interaction effects are identified in both experiments.

We also use LSMeans plots (Appendix F, Figure 31 and Figure 32) to visualize the main and interaction effects of *Method* and *System* in lab 2.2 and lab 2.4. Results confirm that the participants with treatment UCM_R performed better than those with treatment UCM_UR. Unlike *CD* in lab 2.1 and lab 2.3 where the participants working on VS performed better than the participants working on CPD for all four quality measures (Section 5.2.1.2), we don't observe the same phenomenon for *SD* for lab 2.2 though we do in Lab 2.4 where significant differences between VS and CPD are observed for all the five measures.

To account for learning effects, the interaction effect of lab order on *SD* was also studied. We assume that participants would tend to perform better on lab 2.4 than lab 2.2, regardless of other factors. We performed a two-way ANOVA analysis to assess the impact of lab order on the five *SD* quality measures and the results are given in Table 49 (Appendix F). Corresponding LS Means plots are provided in Appendix F, Figure 33 to help visualizing the results. The results show a statistically significant impact of lab order through interactions with *Method* on *Message Completeness*, *SD Completeness*, *Message Correctness*, and *SD Correctness*, but not on *BCE Consistency*. The most plausible explanation is that participants improved their UML sequence diagram modeling skills from experience in lab 2.2 and therefore performed better in lab 2.4. This tentative explanation is confirmed by the participants when filling pre-lab questionnaires (Appendix E), reporting on how well they felt they mastered UML sequence diagram modeling.

5.2.2.3 Summary

The participants applying UCM_R consistently performed better than participants applying UCM_UR in terms of most of the five measures in Experiment 2. Statistically

significant differences when combining observations from both systems are summarized in Table 28.

Table 28 Summary of *t*-test results of SD

Experiment	Message Completeness	SD Completeness	Message Correctness	SD Correctness	BCE Consistency
2.2	No	No	No	No	No
2.4	No	Yes	Yes	No	No

A two-way ANOVA was performed to test both the interaction between *Method* and *System* and between *Method* and *Order* for Experiment 2. No significant interaction effects were identified for any of the four measures. However, significant main effects of *System* and *Order* were found on various measures. The results are summarized in Table 26. Except for *BCE Consistency*, a statistically significant main effect of *Order* was identified for the other four measures in lab 2.4. This is because the participants, building on their experience in lab 2.2, performed much better in lab 2.4. This explanation is also confirmed from the responses of the participants to the pre-lab questionnaires in the two labs.

Table 29 Summary of ANOVA tests - SD

Main effects	Exp.	Message Completeness	SD Completeness	Message Correctness	SD Correctness	BCE Consistency
<i>System</i> (in favor of CPD)	2.2	No	Yes	No	No	No
	2.4	No	No	No	No	No
<i>Order</i> (in favor of lab 2.4)		Yes	Yes	Yes	Yes	No

5.2.3 Consistency between analysis class and sequence diagrams (CS)

The variable *CS* denotes the consistency between analysis class and sequence diagrams. It is the focus of one of the measures evaluating sequence diagrams: *SDCD Consistency* (Table 21). The descriptive statistics of the measure are presented in Table 30. Table 31 presents a summary of the statistical *t*-test results for *CS*. The results do not show statistically significant differences between the two treatments. This is reasonable because UCM_R does not in any way help designers keep consistency between analysis

class and sequence diagrams. Non-parametric test results are not very different from the t -test results and are therefore not presented.

Table 30 Descriptive statistics of CS – lab 2.2 and lab 2.4

Methods	Lab 2.2		Lab 2.4	
	Mean	Size	Mean	Size
UCM_R	0.804	16	0.881	14
UCM_UR	0.812	14	0.848	16
All Methods	0.808	30	0.865	30

Table 31 One way t -test – CS – lab 2.2 and lab 2.4

Experiment	Mean difference (UCM_R – UCM_UR)	DF	t-value	p-value
2.2	-0.008	27	-0.149	0.4413
2.4	0.033	28	0.535	0.2986

5.2.4 Correctness of responses to the comprehension questionnaire

The dependent variable QC denotes the correctness of responses from a comprehension questionnaire (Sections 4.6.3 and 4.6.4). In this section, we report on two-tailed t -test results using the factor *Method*. The descriptive statistics of the measure are presented in Table 32 and the t -test results are given in Table 33.

The t -test result shows a significant difference between the two treatments in the expected direction for all the experiments, thus indicating an increased understanding due to restriction rules and the template. Non-parametric test results are not very different from the t -test results and are therefore not presented.

Table 32 Descriptive statistics of QC – Experiment 1, Experiment 2: lab 2.2, and lab 2.3

Methods	Experiment 1		Lab 2.2		Lab 2.3	
	Mean	Size	Mean	Size	Mean	Size
UCM_R	0.913	12	0.816	18	0.852	16
UCM_UR	0.527	8	0.545	15	0.494	18
All Methods	0.72	20	0.693	33	0.662	34

Table 33 t -test – QC – Experiment 1, Experiment 2: lab 2.2, and lab 2.3

Experiment	Mean difference (UCM_R – UCM_UR)	DF	t-value	p-value
------------	-------------------------------------	----	---------	---------

1	0.387	8	5.189	<0.0008
2.2	0.271	29	4.4	<0.0002
2.3	0.356	26	8.4	<0.0002

6 THREATS TO VALIDITY

Two main threats to external validity are related to our experiment, and are typical to controlled experiments in artificial settings and within time constraints: 1) are the subjects representative of software professionals? 2) Is the experiment material representative of industrial practice, in terms of the size of the systems we used?

Regarding issue 1), recall that in Task 1 the participants designed use case models by applying the restriction rules and the use case template we proposed. This task is usually performed by requirements engineers during the requirements elicitation phase of a typical software development lifecycle. Given the state of practice in most of the software industry, either participants or professional requirements engineers will likely require training. The participants of our experiment are 4th year software and computer engineering students who have received extensive training in use case modeling in previous courses. In addition, they were given a one-and-half-hour lecture and an assignment specifically focusing on how to apply the restriction rules and the use case template. In our context, the main difference between students and professional requirements engineers, is that the latter could have more experience on designing use case models, they could be more familiar with the domain, and thus we assume that they would probably apply more effectively the restriction rules and the use case template than students given the same amount of training and time to perform the task. Thus, we believe that professional requirements engineers would be able to further benefit from the restriction rules and template, and thus provide a more positive opinion on the rules' understandability, applicability, and restrictiveness. As for Task 2, the students derived analysis models from the use case models, with or without restrictions and template. This task is usually performed by system analysts. Again, our 4th year software and computer engineering students have received extensive training on software modeling with UML,

through several courses, and this is more than what we have observed in most software development environments.

As for issue 2) above, the scale of the systems is not likely to have a significant impact on the results of the experiment for Task 1. Indeed, this task does not require an overall understanding of the systems as the use case diagrams of the two systems were provided to the students as part of the experiment material. The students were only asked to write some UCSs by applying the restriction rules and the template. Due to time constraints (two three-hour laboratories), it was anyway not feasible to consider larger scale systems (with more UCSs) for Task 2.

Construct validity is related to our measurement instruments: the two comprehension questionnaires used respectively for the two tasks. The comprehension questionnaire for Task 1 (Section 4.5.1.3) was not involved in any comparison so it does not bear on construct validity. The questions of the comprehension questionnaire for Task 2 (Section 4.5.2.3) were designed to be answerable from the use case models with or without restrictions; therefore introducing no bias for any of the treatments. The same quality measures are used to measure the students class and sequence diagrams derived from the use case models with or without restrictions and therefore no bias is introduced as well when evaluating these diagrams.

Three students presented problems related to internal validity. One of them missed the lab for Task 2 and had to perform the task at home; another spent 3 hours 40 mins on Task 2; the other produced a very incomplete result (no class diagram was derived). These three data points were excluded from the analysis. The participants belonging to different groups were monitored to ensure they would not access each other's documents during the entire lab duration. By doing so, we also limited the threat on the internal validity of the experiments.

7 CONCLUSION

Use case modeling is one of the most common practices for capturing functional requirements. However, use case specifications (UCSs) are essentially textual documents and therefore ambiguity is inevitably introduced. To facilitate the transition towards analysis models, the UCSs are expected to be the least ambiguous possible, especially when the goal is to provide automated support for the generation of analysis models. In this paper, we propose a use case modeling approach, referred to as RUCM, which is composed of 26 well-defined restriction rules and a use case template. Its purpose is to restrict the use of natural language when users document UCSs in order to reduce ambiguity and also facilitate automated transition towards analysis models.

Two controlled experiments have been conducted, in the context of a fourth year Software Engineering course, to evaluate whether RUCM is easy to apply while developing use case models and whether it helps obtain higher quality analysis models. Each restriction rule is evaluated in terms of its understandability, applicability, restrictiveness, and error rate. The experiment results indicate that our 26 restriction rules are easy to apply and with focused training on the rules receiving higher error rates and appropriate tool support (e.g., enforcing the usage of keyword and the definition of compulsory alternative flows), error rates can be expected to further decrease. This forms our future work. Based on these results, we are therefore confident that trained engineers are capable of properly applying our restriction rules and template and obtain UCSs from which to derive analysis models.

Another goal of the controlled experiments was to evaluate whether RUCM helps derive higher quality analysis models, by comparing it to a common use case modeling approach that does not put restrictions on natural language. The quality of analysis class and sequence diagrams is evaluated from the viewpoints of their correctness, completeness, and redundancy. The results show that RUCM results in significant improvements regarding the class correctness of derived class diagrams consistently in all the experiments, but not their completeness and redundancy. The results also show that

RUCM leads to significant improvement on the diagram completeness and message correctness of derived sequence diagrams during the second lab of the second experiment. Furthermore, our approach resulted in a large improvement in term of comprehension of the use case models as measured by a carefully designed questionnaire.

To the best of our knowledge, this paper is the first controlled experiment that evaluates the applicability, both individually and as a whole, of restriction rules used to document UCSs and that also evaluates the impact of these rules and template on the quality of generated analysis class and sequence diagrams. The measures we have defined to characterize restriction rules and evaluate the quality of analysis class diagrams can be reused for similar experiments to be conducted in the future.

REFERENCES

- [1] Achour C. B., Rolland C., Maiden N. A. M. and Souveyet C., “Guiding use case authoring: Results of an empirical study,” *Proc. RE'99*, pp. 36-43, 1999.
- [2] Anda B., Sjoberg D. and Jorgensen M., “Quality and understandability of use case models,” *Proc. ECOOP 2001*, pp. 402-428, 2001.
- [3] Basili V. R., Caldiera G. and Rombach H. D., “The goal question metric approach,” *Encyclopedia of Software Engineering*, vol. 1, pp. 528-532, 1994.
- [4] Bittner K. and Spence I., *Use Case Modeling*, Addison-Wesley Boston, 2002.
- [5] Bruegge B. and Dutoit A. H., *Object-Oriented Software Engineering Using UML, Patterns, and Java*, Prentice Hall, 2nd Edition, 2004.
- [6] Bruegge B. and Dutoit A. H., *Object-Oriented Software Engineering Using UML, Patterns, and Java*, Prentice Hall, 3rd Edition, 2009.
- [7] Cockburn A., *Writing effective use cases*, Addison-Wesley Boston, 2001.
- [8] Cox K., *Heuristics for use case descriptions*, PhD Thesis, Bournemouth University, UK, 2002
- [9] Dobing B. and Parsons J., “How UML is used,” *Com. of the ACM*, vol. 49 (5), pp. 109-113, 2006.

- [10] Gomaa H., *Designing Concurrent, Distributed, and Real-Time Applications with UML*, Addison-Wesley, 2000.
- [11] Insfrán E., Pastor O. and Wieringa R., “Requirements Engineering-Based Conceptual Modelling,” *Requirements Engineering*, vol. 7 (2), pp. 61-72, 2002.
- [12] Ivar J., “Object-oriented development in an industrial environment,” *Proc. OOPSLA*, 1987.
- [13] Jacobson I., “Use cases - yesterday, today, and tomorrow,” *Software and Systems Modeling*, vol. 3 (3), pp. 210-220, 2004.
- [14] Jacobson I., Christerson M., Jonsson P. and Overgaard G., *Object-oriented software engineering: a use case driven approach*, Addison-Wesley, 1992.
- [15] Kruchten P., *The Rational Unified Process: An Introduction*, Addison-Wesley, 2003.
- [16] Kulak D. and Guiney E., *Use cases: requirements in context*, ACM Press, 2000.
- [17] Larman C., *Applying UML and Patterns*, Prentice-Hall, 3rd Edition, 2004.
- [18] Liu D., *Automating Transition from Use Cases to Class Model*, Thesis, University of Calgary, Department of Electrical and Computer Engineering, 2003
- [19] OMG, “UML 2.0 Superstructure Specification,” Object Management Group, <http://www.omg.org/spec/UML/2.0/>, 2005.
- [20] Oppenheim A. N., *Questionnaire design, interviewing, and attitude measurement*, Pinter Pub Ltd, 1992.
- [21] Phalp K. T., Vincent J. and Cox K., “Assessing the quality of use case descriptions,” *Software Quality Journal*, vol. 15 (1), pp. 69-97, 2007.
- [22] Phalp K. T., Vincent J. and Cox K., “Improving the quality of use case descriptions: empirical assessment of writing guidelines,” *Software Quality Journal*, vol. 15 (4), pp. 383-399, 2007.
- [23] Schneider G. and Winters J. P., *Applying use cases: a practical guide*, Object Technology, Addison-Wesley, 1998.
- [24] Śmiałek M., Bojarski J., Nowakowski W., Ambroziewicz A. and Straszak T., “Complementary Use Case Scenario Representations Based on Domain Vocabularies,” *Proc. MoDELS*, 2007.

- [25] Somé S. S., “Supporting use case based requirements engineering,” *Information and Software Technology*, vol. 48 (1), pp. 43-58, 2006.
- [26] Subramaniam K., Liu D., Far B. H. and Eberlein A., “UCDA: Use Case Driven Development Assistant Tool for Class Model Generation,” *Proc. SEKE'04*, 2004.
- [27] The Stanford Parser version 1.6: The Stanford Natural Language Processing Group, <http://nlp.stanford.edu/software/lex-parser.shtml>.
- [28] Wahono R. S. and Far B. H., “A framework for object identification and refinement process in object-oriented analysis and design,” *Proc. Cognitive Informatics*, pp. 351-360, 2002.
- [29] Wohlin C. and Wesslen A., *Experimentation in Software Engineering: An Introduction*, Springer, 2000.
- [30] Yue T., Briand L. C. and Labiche Y., “Automatically Deriving a UML Analysis Model from a Use Case Model,” Carleton University, Technical Report SCE-09-09, 2009.
- [31] Yue T., Briand L. C. and Labiche Y., “A Systematic Review of Transformation Methodologies between User Requirements and Analysis Models,” Carleton University, Technical Report SCE-09-03, 2009.
- [32] Yue T., Briand L. C. and Labiche Y., “A Use Case Modeling Approach to Facilitate the Transition Towards Analysis Models: Concepts and Empirical Evaluation,” *Proc. MODELS2009*, 5795/2009, pp. 484-498, 2009.
- [33] Yue T., Briand L. C. and Labiche Y., “An Automated Approach to Transform Use Cases into Activity Diagrams,” *Proc. 6th ECMFA*, LNCS 6138, pp. 337-353, 2010.

Appendix A Restriction rules

Table 34 Restrictions (part 1)

#	Description	Example		Location to apply
		RIGHT	WRONG	
1	The subjects of sentences in basic and alternative flows should be the system or actors.	<i>The system</i> ejects the ATM card	<i>The card</i> has been ejected	Apply to action steps of basic and alternative flows of events. Don't apply to conditions: <ul style="list-style-type: none"> • IF conditions • ELSEIF conditions • VALIDATES THAT conditions • DO-UNTIL conditions
2	Describe the flow of events sequentially.	1. The system dispenses the cash amount. 2. The system ejects the ATM card.	1. The system ejects the ATM card. 2. The system dispenses the cash amount.	
3	Actor-to-actor interactions are not allowed.	The customer inserts the ATM card into the card reader.	The customer gives the teller the ATM card.	
4	Describe one action per sentence. (Avoid compound predicates.)	the system cancels the transaction MEANWHILE the system ejects the card. or 1. the system cancels the transaction. 2. the system ejects the card	...the system cancels the transaction and ejects the card.	
5	Use present tense only.	The system <i>ejects</i> the card.	The system <i>ejected</i> the card.	
6	Use active voice rather than passive voice in flow of events	The system <i>ejects</i> the card.	The card <i>is ejected</i> .	
7	Clearly describe the interaction between the system and actors.	ATM customer enters PIN number <i>to the system</i> .	Customer enters PIN.	

Table 35 Restrictions (part 2)

#	Description	Example	
		RIGHT	WRONG
8	Use declarative sentence only.	The system ejects the card.	Ejects the card.
9	Use words in a consistent way.	<i>ATM customer</i> inserts the ATM card...	<i>Customer</i> inserts the ATM card...
10	Don't use modal verbs (e.g., <i>might, could, should</i>).	The system ejects the card.	The system <i>might</i> eject the card.
11	Avoid adverbs, such as <i>very, more, rather, and the like</i> .	The system ejects the card.	The system <i>very likely</i> ejects the card.
12	Use simple sentences only. A simple sentence must contain only one subject and one predicate. No compound subject and predicate are allowed. This restriction does not apply to the sentences using keywords: IF-THEN-ELSE-ELSEIF pairs, DO-UNTIL, MEANWHILE, and VALIDATES THAT.	<ol style="list-style-type: none"> The system displays ATM customer accounts. The system prompts ATM customer for ... 	System displays customer accounts <i>and</i> prompts customer for transaction type...
13	Don't use negatively adverb or adjective such as <i>hardly</i> and <i>never</i> . But it is allowed to use <i>not</i> or <i>no</i> .	The PIN number <i>has not</i> been validated	The PIN number <i>has never been</i> validated.
14	Don't use pronouns such as <i>he, his, him, himself, this, that, everyone, one, another, each, everything, many, it, etc.</i>	... <i>the system</i> reads the card number.	... <i>it</i> reads the card number.
15	Don't use participle phrases as adverbial modifier.	The system is idle. The system is displaying a Welcome message.	ATM is idle, displaying a Welcome message.
16	Use "the system" to refer to the system under design consistently.	the system	"ATM" or "The ATM system"

Table 36 Restrictions (part 3)

#	Description	Grammar	Explanation	Example	
				RIGHT	WRONG
17	Use keywords INCLUDE USE CASE to describe the include dependencies with other use cases.	INCLUDE USE CASE <included use case name>	The keywords can be used in basic and step alternative flows.	INCLUDE USE CASE Validate PIN	Include Validate PIN abstract use case.
18	Use keywords EXTENDED BY USE CASE to refer to the extended use case.	EXTENDED BY USE CASE <extending use case> or <specific use case> Appears either in the action part of a THEN or ELSE in the main flow, or as an action in an alternative flow.	The keywords can be used in basic and alternative flows.	EXTENDED BY USE CASE CreateIncident	Use case CreateIncident extends the current use case.
19	Use keyword RFS in a specific (or bounded) alternative flow to refer to a step number (or a lower bound step number and an upper bound step number) of a basic flow step that this alternative flow corresponds to.	RFS <basic flow step #> (specific alternative flow) RFS <set of step numbers> (specific alternative flow) Not required notation for global alternative flow.	One specific or bounded alternative flow must correspond to exactly one or more than one basic flow steps.	RFS 5 ... RFS 5-7, 10, 14 ...	
20	Use pairs of keywords of IF, THEN, ELSE, ELSEIF, and ENDIF to describe conditional logic sentences.	IF <condition> THEN <steps> ENDIF	Appears in one flow only.	IF the system recognizes the ATM card, THEN the system reads the ATM card number.	If the system recognizes the card, it reads the card number.
		IF <condition> THEN <steps> – ELSE <steps> ENDIF	IF-THEN is used in basic flows; while ELSE-ENDIF is used in alternative flows (see RFS). Or everything is used in only one flow.		
		IF <condition> THEN <steps> – ELSEIF <condition> THEN <steps> ENDIF	IF-THEN is used in basic flows; while ELSEIF-THEN-ENDIF is used in alternative flows (see RFS). Or everything is used in only one flow.		

Table 37 Restrictions (part 4)

#	Description	Grammar	Explanation	Example	
				RIGHT	WRONG
21	Use keyword MEANWHILE to describe concurrency.	<action> MEANWHILE <action>	It implies that the sentence before keyword MEANWHILE and the sentence after the keyword occur concurrently.	...the system cancels the transaction MEANWHILE the system ejects the card.	...the system cancels the transaction and ejects the card.
22	Use keywords VALIDATES THAT to describe condition checking sentences. VALIDATES THAT means that the condition is evaluated and must be true to proceed to the next step.	VALIDATES THAT <condition>	The alternative case (the condition does not hold) must be described in its corresponding alternative flow (RFS).	...the system VALIDATES THAT the user- entered PIN...	... the system checks whether the user-entered PIN...
23	Use keywords DO and UNTIL to describe iteration.	DO <steps> UNTIL <condition >	Following keyword DO is a sequence of steps. Following keyword UNTIL is a loop ending condition.	1. DO 2. action1 3. action2 4. UNTIL condition	
24	Use keyword ABORT to describe an exceptionally exit action. An alternative flow ends either with ABORT or RESUME STEP.	ABORT	Used in alternative flows, iterative, and conditional logic sentences. It means the ending of a use case.		
25	Use keywords RESUME STEP to describe the situation where an alternative flow goes back to its corresponding basic flow. An alternative flow ends either with ABORT or RESUME STEP.	RESUME STEP <basic flow step #>	Used in alternative flows.	RESUME STEP 5	
26	Each basic flow and alternative flow should have their own postconditions.	Refer to restriction # 19.			

Appendix B Experiment 1- Comprehension Questionnaire

Please put a (√) in the corresponding column.

You are strongly encouraged to refer to the list of restrictions you were provided with.
(The restriction numbers in the following tables match the numbers provided in the list of restriction specifications).

Table 38 Experiment 1 – Comprehension Questionnaire – Part 1 (R1-R16)

#	Restriction	I understood the restriction rule and was able to properly apply it.		The restriction rule is straightforward to apply.				Did you feel the restriction rule was too restrictive?			
		Yes	No	Completely agree	Generally agree	Generally disagree	Completely disagree	Completely agree	Generally agree	Generally disagree	Completely disagree
1	The subjects of sentences in basic and alternative flows should be the system or actors.										
2	Describe the flow of events sequentially.										
3	Actor-to-actor interactions are not allowed.										
4	Describe one action per sentence.										
5	Use present tense only.										
6	Use active voice rather than passive voice.										
7	Clearly describe the interaction between the system and actors.										
8	Use declarative sentence only.										
9	Use words in a consistent way.										
10	Don't use modal verbs.										
11	Avoid adverbs.										
12	Use simple sentence only.										
13	Don't use negatively adverb and adjective.										
14	Don't use pronouns.										
15	Don't use participle phrases as adverbial modifier.										
16	Use "the system" to describe the system under design consistently.										

Table 39 Experiment 1 – Comprehension Questionnaire – Part 2 (R17-R26)

#	Restriction	I understood the restriction rule and was able to properly apply it.		The restriction rule is straightforward to apply.				Did you feel the restriction rule was too restrictive?			
		Yes	No	Completely agree	Generally agree	Generally disagree	Completely disagree	Completely agree	Generally agree	Generally disagree	Completely disagree
17	INCLUDE USE CASE										
18	EXTENDED BY USE CASE										
19	RFS <basic flow step #>										
20	IF-THEN-ELSE-ENDIF (or ELSEIF-THEN-ENDIF)										
21	MEANWHILE										
22	VALIDATES THAT										
23	DO-UNTIL										
24	ABORT										
25	RESUME STEP										
26	Each basic flow and alternative flow should have their own postconditions.										

Appendix C Experiment 2 – Comprehension Questionnaire (CPD)

Instruction:

1. Please answer all questions, which are organized use case by use case.
2. For each multiple-choice question, please:
 - Select only one choice for each question, except those indicated with three asterisk (***)
Please circle your answer.
 - Justify your choice by further answering the question: “where did you get the information from”. Please clearly indicate the location (e.g., Basic flow – Step 2, Precondition, Postcondition) where you found information in the use case description to answer the multiple-choice question.
 - Some questions may have a choice: “Other answer”. If your answer is not in the provided choices, you may select this choice and provide your answer. Please also answer the question: “where did you get the information”.
3. A special question is asked for each use case. The following is the question:

*Did you identify any place(s) in the use case specification, which cause any confusion for you to understand the use case, for example, in terms of the sequence of the steps of basic flow or alternative flows, and/or the subjects of actions? **YES or NO.***

If your answer is “yes”, please list the places and provide a brief explanation.

If you identify any confusion about the use case, please circle YES and describe the confusion.
4. Please make sure that your handwriting is recognizable.

Use case CreateCusotmerOrder

1. Which of the following scenarios is correct in terms of creating a new customer order?
 - a. Sales creates a new customer order without interacting with the system.
 - b. Sales requests the system to create a new customer order by providing the system the customer data.
 - c. The customer requests the system to create a new order for him/her. Then the system informs Sales the new created customer order.
 - d. The system creates a new customer order and informs Sales and the customer about the newly created customer order.
 - e. Not specified in the use case specification.
 - f. Other answers:

Where did you get the information from?

2. Which of the following scenarios is correct in terms of creating a part order for the customer?
 - a. Sales creates a part order indicated by the customer without interacting with the system.
 - b. Sales requests the system to create a part order.
 - c. The customer requests the system to create a new part order. Then the system informs Sales to create a new part order for the customer.
 - d. The system creates a part order and informs Sales and the customer about the newly created part order.
 - e. Not specified in the use case specification.
 - f. Other answers:

Where did you get the information from?

3. How is the newly created customer order saved?
 - a. The system saves the customer order by itself.
 - b. Sales keeps the customer order information.
 - c. The customer keeps his/her order information.
 - d. The system asks DBMS to save the customer order.
 - e. Not specified in the use case specification.
 - f. Other answers:

Where did you get the information from?

4. Did you identify any place(s) in the use case specification, which cause any confusion for you to understand the use case specification, for example, in terms of the sequence of the steps of the basic flow or the alternative flows, the subjects of actions?

YES or NO.

If your answer is “yes”, please list the places and provide a brief explanation.

Use case OrderPart

5. How does Sales know whether the part number given by the customer is unknown or not?
- Sales has all the records of the part numbers, so he/she can figure out.
 - Sales directly asks DBMS to validate the part number.
 - Sales asks the system to validate the part number. Then the system retrieves the information from DBMS.
 - Not specified in the use case.
 - Other answers:

Where did you get the information from?

6. How does the system know whether there is enough stock for the customer part order or not?
- The system asks Sales for the information.
 - The system asks the customer for the information.
 - The system asks DBMS for the information.
 - The DBMS tells the system spontaneously.
 - Not specified in the use case.
 - Other answers:

Where did you get the information from?

7. When the part number is unknown, Sales provides an alternative part number to the system, then what happens?
- The system requests DBMS to check whether this alternative part number is known or unknown.
 - The system orders the part with the alternative part order for the customer.
 - The use case terminates.
 - Not specified in the use case.
 - Other answers:

Where did you get the information from?

8. Did you identify any place(s) in the use case specification, which cause any confusion for you to understand the use case, for example, in terms of the sequence of the steps of the basic flow or the alternative flows, the subjects of actions?

YES or NO.

If your answer is “yes”, please list the places and provide a brief explanation.

Use case InsufficientStock

9. How does the system know whether the customer accepts the alternative part?
- Sales asks the customer whether an alternative part would do. Then Sales informs the system the customer’s decision.
 - The customer tells the system his/her decision directly.

- c. The system does not need to know whether the customer accepts the alternative parts.
- d. Not specified in the use case
- e. Other answers:

Where did you get the information from?

10. Where is the part order information saved?

- a. The system
- b. Customer
- c. Sales
- d. DBMS
- e. Receiving&Shipping
- f. Accounting
- g. Not specified in the use case
- h. Other answers:

Where did you get the information from?

11. Who is responsible to create a pending order if the customer does not accept the alternative order provided by the system?

- a. The system
- b. Customer
- c. Sales
- d. DBMS
- e. Receiving&Shipping
- f. Accounting
- g. Not specified in the use case
- h. Other answers:

Where did you get the information from?

12. Where is the created pending order information saved if the customer does not accept the alternative order provided by the system?

- a. The system
- b. Customer
- c. Sales
- d. DBMS
- e. Receiving&Shipping
- f. Accounting
- g. Not specified in the use case
- h. Other answers:

Where did you get the information from?

13. Did you identify any place(s) in the use case specification, which cause any confusion for you to understand the use case, for example, in terms of the sequence of the steps

of the basic flow or the alternative flows, the subjects of actions?

YES or NO.

If your answer is “yes”, please list the places and provide a brief explanation.

Use case CreateVendorOrder

14. Who is responsible to create a vendor order for the vendor selected by Sales, who has less number of discrepancies in the vendor’s previous shipments?
- The system
 - Customer
 - Sales
 - DBMS
 - Receiving&Shipping
 - Accounting
 - Not specified in the use case
 - Other answers:

Where did you get the information from?

15. Where is the created vendor order stored?
- The system
 - Customer
 - Sales
 - DBMS
 - Receiving&Shipping
 - Accounting
 - Not specified in the use case
 - Other answers:

Where did you get the information from?

16. Did you identify any place(s) in the use case specification, which cause any confusion for you to understand the use case, for example, in terms of the sequence of the steps of the basic flow or the alternative flows, the subjects of actions?
- YES or NO.**

If your answer is “yes”, please list the places and provide a brief explanation.

Use case CompletePendingOrder

17. How does Sales obtain a pending order that can be completed when there are new arrivals from vendors?
- Sales asks customers.
 - Sales asks DBMS.
 - Sales asks the system. Then the system asks DBMS.
 - Sales maintains a record about the pending order.

- e Not specified in the use case
- f Other answers:

Where did you get the information from?

18. Where is the part order stored?

- a. The system
- b. Customer
- c. Sales
- d. DBMS
- e. Receiving&Shipping
- f. Accounting
- g. Not specified in the use case
- h. Other answers:

Where did you get the information from?

19. Did you identify any place(s) in the use case specification, which cause any confusion for you to understand the use case, for example, in terms of the sequence of the steps of the basic flow or the alternative flows, the subjects of actions?

YES or NO.

If your answer is “yes”, please list the places and provide a brief explanation.

Use case CreatePreventiveOrder

20. Where are the statistics on past orders stored?

- a. The system
- b. Customer
- c. Sales
- d. DBMS
- e. Receiving&Shipping
- f. Accounting
- g. Not specified in the use case
- h. Other answers:

Where did you get the information from?

21. Did you identify any place(s) in the use case specification, which cause any confusion for you to understand the use case, for example, in terms of the sequence of the steps of the basic flow or the alternative flows, the subjects of actions?

YES or NO.

If your answer is “yes”, please list the places and provide a brief explanation.

Appendix D Experiment 2 - Comprehension Questionnaire (VS)

Instruction:

5. Please answer all questions, which are organized use case by use case.
6. For each multiple-choice question, please:
 - Select only one choice for each question, except those indicated with three asterisk (***)
Please circle your answer.
 - Justify your choice by further answering the question: “where did you get the information from”. Please clearly indicate the location (e.g., Basic flow – Step 2, Precondition, Postcondition) where you found information in the use case to answer the multiple-choice question.
 - Some questions may have a choice: “Other answer”. If your answer is not in the provided choices, you may select this choice and provide your answer. Please also answer the question: “where did you get the information from”.
7. A special question is asked for each use case. The following is the question.

*Did you identify any place(s) in the use case specification, which cause any confusion for you to understand the use case, for example, in terms of the sequence of the steps of basic flow or alternative flows, and/or the subjects of actions? **YES or NO.***

If your answer is “yes”, please list the places and provide a brief explanation.

If you identify any confusion about the use case, please circle YES and describe the confusion.
8. Please make sure that your handwriting is recognizable.

Use case Rent Video

1. Where is the information of whether a member has borrowing privileges stored?

- a. Administrator
- b. Employee
- c. The system itself.
- d. DBMS
- e. CCS
- f. Read Card Strip
- g. Not specified in the use case.
- h. Other answers:

Where did you get the information from?

2. Which one of the following statements is correct?

- a. First the system identifies the video copy from DBMS. Then the system updates the member's account information to reflect the additional rental.
- b. First the system updates the member's account information to reflect the additional rental. Then the system identifies the video copy from DBMS.
- c. The system identifies the video copy from DBMS, and concurrently the system updates the member's account information to reflect the additional rental.

Where did you get the information from?

3. How does the system checks whether a video copy is reserved by a member?

- a. The information is stored in the system itself.
- b. The system gets the information by asking the employee.
- c. The system requests the information from DBMS.
- d. Not specified in the use case.
- e. Other answers:

Where did you get the information from?

4. ***What information should have been saved into DBMS after the rental is completed?
(you may want to select more than one choices)

- a. The updated account information of the member.
- b. The return date.
- c. The member's name.
- d. The member's identification number.
- e. Not specified in the use case.
- f. Other answers:

Where did you get the information from?

5. How does the employee know that the rental is confirmed?

- a. The employee gets the confirmation from the member.
- b. The employee gets the confirmation from the system.

- c. The employee gets the confirmation from DBMS.
- d. The employee gets the confirmation from another employee.
- e. Not specified in the use case.
- f. Other answers:

Where did you get the information from?

6. If the video copy is reserved by the member, then the reservation becomes fulfilled. What following action should the system take?
- a. The system exits.
 - b. The system sends the employee a confirmation.
 - c. The system communicates with DBMS to save the updated member's account information.
 - d. Not specified in the use case.
 - e. Other answers:

Where did you get the information from?

7. Did you identify any place(s) in the use case specification, which cause any confusion for you to understand the use case, for example, in terms of the sequence of the steps of the basic flow or the alternative flows, the subjects of actions?
YES or NO.

If your answer is "yes", please list the places and provide a brief explanation.

Use case Return Video

8. Which one of the following statements is correct?
- a. First the system identifies the video copy from DBMS. Then the system checks whether the video copy is overdue.
 - b. First the system checks whether the video copy is overdue. Then the system identifies the video copy from DBMS.
 - c. The system identifies the video copy from DBMS, and concurrently the system checks whether the video copy is overdue.

Where did you get the information from?

9. How does the employee know that the video copy return is confirmed?
- a. The employee gets the confirmation from the member.
 - b. The employee gets the confirmation from the system.
 - c. The employee gets the confirmation from DBMS.
 - d. The employee gets the confirmation from another employee.
 - e. Not specified in the use case.
 - f. Other answers:

Where did you get the information from?

10. Where is the information regarding whether the title of the returning video copy is on hold (i.e., there is at least one reservation) by other members stored?
- The system itself
 - The employee
 - The member
 - The DBMS
 - The CCS
 - Not specified in the use case.
 - Other answers:

Where did you get the information from?

11. How does the system update the reservation status of a video copy from “on hold” to “outstanding”?
- The system itself updates the reservation status
 - Through the employee
 - Through the member
 - Through DBMS
 - Through CCS
 - Not specified in the use case.
 - Other answers:

Where did you get the information from?

12. In which condition, the use case Video Overdue is invoked? (If you cannot find the answer directly from the use case description, please indicate it.) (if you find the answer in the use case description, please indicate the place.)

13. Did you identify any place(s) in the use case specification, which cause any confusion for you to understand the use case, for example, in terms of the sequence of the steps of the basic flow or the alternative flows, the subjects of actions?

YES or NO.

If your answer is “yes”, please list the places and provide a brief explanation.

Use case Video Overdue

14. Which one of the following statements regarding calculating the fine for the overdue video copy is correct?
- First, CCS sends the current date and time to the system. Then the system calculates the fine for the overdue video copy.
 - First, the system requests CCS to calculate the fine for overdue video copy. Then CCS returns the amount of fine to the system.
 - First, the system requests CCS to provide the current date and time. Then the system calculates the fine of the overdue video copy according to the current date and time returned by CCS.

- d. First, the system calculates the fine for the overdue video copy. Then the system requests the current date and time from CCS.

Where did you get the information from?

15. Who is responsible for requesting Printer to print the receipt for the fine?

- e. DBMS
- f. Employee
- g. The system
- h. The member (or customer)
- i. CCS
- j. Printer itself
- k. Not specified in the use case.
- l. Other answers:

Where did you get the information from?

16. What information should be stored into DBMS when the payment for the overdue video copy has been processed? (If you cannot find the answer directly from the use case description, please indicate it.) (if you find the answer in the use case description, please indicate the place.)

17. How does the employee know that the overdue video copy is available for further rental after the fine has been processed?

- a. The employee gets the confirmation from the member.
- b. The employee gets the confirmation from the system.
- c. The employee gets the confirmation from DBMS.
- d. The employee gets the confirmation from CCS.
- e. Not specified in the use case.
- f. Other answers:

Where did you get the information from?

18. How are the member's privileges suspended, when the member does not pay the fine?

- a. The employee suspends the member's privilege by telling the member the fact that his/her privileges are suspended.
- b. The employee informs the system that the member does not pay the fine. The system suspends the member's privileges and also informs DBMS to update the member's account information.
- c. Not specified in the use case.
- d. Other answers:

Where did you get the information from?

19. Did you identify any place(s) in the use case specification, which cause any confusion for you to understand the use case, for example, in terms of the sequence of the steps of the basic flow or the alternative flows, the subjects of actions?

YES or NO.

If your answer is "yes", please list the places and provide a brief explanation.

Use case Reserve Video

20. How does the system know whether a member has borrowing privileges before it reserves a video copy to the member?
- The information is stored in the system directly.
 - Inquiry DBMS.
 - The employee informs the system after asking the member either through email or telephone.
 - Not specified in the use case.
 - Other answers:

Where did you get the information from?

21. After the member who wants to make a reservation on a video title is identified, how does the system get the information regarding the title to be reserved?
- The information is stored in the system directly.
 - Inquiry DBMS.
 - The employee informs the system after asking the member.
 - The member tells the system directly.
 - Not specified in the use case.
 - Other answers:

Where did you get the information from?

22. Which one of the following statements is correct, when there is a copy available for the title that the member wants to reserve?
- First, the system creates an outstanding reservation for the title. Then the information of the outstanding reservation is stored into DBMS. At last, the reservation number is communicated to the member through employee.
 - First, DBMS creates an outstanding reservation for the title. Then DBMS informs the system that the reservation has been created along with the reservation number. At last, the reservation number is communicated to the member through employee.
 - First, the system creates an outstanding reservation for the title. Then the reservation number is communicated to the member through employee.
 - First, DBMS creates an outstanding reservation for the title. Then the reservation number is communicated to the member through employee.

Where did you get the information from?

23. What does the system do when it validates that the member has no borrowing privileges?
- The system asks for information regarding the title to be reserved.
 - The system exits directly.
 - The system informs the employee that the member has no borrowing privilege and then exits.

- d. They system makes a reservation for the member and informs employee about the created reservation.
- e. Not specified in the use case.
- f. Other answers:

Where did you get the information from?

24. What does the system do when it validates that the title the member requested to reserve does not exist?
- a. The system checks whether there are copies available for this title.
 - b. The system exits directly.
 - c. The system informs the employee that the title does not exist and then exits.
 - d. They system makes a reservation for the member and informs employee about the created reservation.
 - e. Not specified in the use case.
 - f. Other answers:

Where did you get the information from?

25. Did you identify any place(s) in the use case specification, which cause any confusion for you to understand the use case, for example, in terms of the sequence of the steps of the basic flow or the alternative flows, the subjects of actions?
YES or NO.

If your answer is “yes”, please list the places and provide a brief explanation.

Use case Check Database

26. Where is the information of rentals and overdue video copies stored?
- a. The information is stored in the system directly.
 - b. Members of the store
 - c. DBMS
 - d. Employee
 - e. CCS
 - f. Administrator
 - g. Email System
 - h. Not specified in the use case.
 - i. Other answers:

Where did you get the information from?

27. Where is the information of expired reservations stored?
- a. The information is stored in the system directly.
 - b. Members of the store
 - c. DBMS
 - d. Employee

- e. CCS
- f. Administrator
- g. Email System
- h. Not specified in the use case.
- i. Other answers:

Where did you get the information from?

28. If there is no overdue video copies in rentals identified, what action(s) should the system take next?

- a. The system exits.
- b. The system continues to check whether there are any expired reservations.
- c. The system waits for employee's response.
- d. Not specified in the use case.
- e. Other answers:

Where did you get the information from?

29. If there is no expired reservation identified, what action(s) should the system take next?

- a. The system exits.
- b. The system continues to identify the titles generating insufficient income.
- c. The system waits for employee's response.
- d. Not specified in the use case.
- e. Other answers:

Where did you get the information from?

30. Did you identify any place(s) in the use case specification, which cause any confusion for you to understand the use case, for example, in terms of the sequence of the steps of the basic flow or the alternative flows, the subjects of actions?

YES or NO.

If your answer is "yes", please list the places and provide a brief explanation.

Appendix E Pre- and post- lab questionnaires

Pre-Lab Questionnaire (Experiment 1.1)

Levels of agreement:

1 – Completely agree

2 – Generally agree

3 – Generally disagree

4 – Completely disagree

Questions:

	1	2	3	4
• I have received extensive teaching on use case modeling.	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
• I have substantial experience with use case modeling.	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
• I am confident I can fully understand a use case diagram.	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
• I am confident I can accurately apply the use case template discussed in class.	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
• I fully understand the restrictions that apply to use case descriptions.	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
• I am confident I can accurately apply the restrictions.	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

Post-Lab Questionnaire (Experiment 1.1)

Levels of agreement:

1 – Completely agree

2 – Generally agree

3 – Generally disagree

4 – Completely disagree

Questions:

	1	2	3	4
1. The instructions and objectives of the lab were perfectly clear to me.	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
2. I fully understood the use case diagram I was provided.	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
3. I was entirely comfortable when applying the use case template.	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
4. I was entirely comfortable when applying the restrictions.	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
5. I gained valuable experience by applying the restrictions and in the future I will be in a better position to perform such tasks.	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
6. Did you have enough time to finish the task?			YES <input type="checkbox"/>	NO <input type="checkbox"/>
If not, please answer the following sub-questions:				
• I spent too much time on understanding the case study system.	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
• I spent too much time on understanding the restrictions.	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
• I spent too much time on understanding the use case template.	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

Pre-Lab Questionnaire (Experiment 1.2)

Levels of agreement:

1 – Completely agree

2 – Generally agree

3 – Generally disagree

4 – Completely disagree

Questions:

	1	2	3	4
• I have good knowledge on UML modeling (class and sequence diagrams).	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
• I have substantial experience in building analysis models from use cases.	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
• I am confident I can accurately apply the stereotypes <<Entity>> <<Boundary>>, and <<Control>> to classify classes.	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
• I fully understand how to keep the consistency between class and sequence diagrams.	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
• I am confident I can correctly use MS-Visio to draw class diagrams.	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
• I am confident I can correctly use MS-Visio to draw sequence diagrams.	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

Post-Lab Questionnaire (Experiment 1.2)

Levels of agreement:

1 – Completely agree

2 – Generally agree

3 – Generally disagree

4 – Completely disagree

Questions:

	1	2	3	4
• The instructions and objectives of the lab were perfectly clear to me.	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
• I had plenty of time to finish the lab.	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
• I fully understood the use case descriptions I was provided.	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
• The sequence of the basic flow steps is very clear to me.	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
• The sequences of the alternative flow steps are very clear to me.	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
• The alternative flows clearly correspond to the basic flow.	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
• The use case template is very easy to apply.	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
• It was extremely hard for me to design the class diagram.	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
• It was extremely hard for me to design the sequence diagrams.	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
• It was extremely hard for me to keep the consistency between the class and the sequence diagrams.	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
• It is extremely hard for me use MS-Visio to draw class diagram.	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
• It is extremely hard for me use MS-Visio to draw sequence diagrams.	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

Pre-Lab Questionnaire (Lab 2.1 and 2.3)

Levels of agreement:

1 – Completely agree

2 – Generally agree

3 – Generally disagree

4 – Completely disagree

Questions:

	1	2	3	4
• I have good knowledge on UML class diagram modeling.	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
• I have substantial experience in building class diagrams from use cases.	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
• I am confident I can accurately apply the stereotypes <<Entity>> <<Boundary>>, and <<Control>> to classify classes.	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
• I am confident I can correctly use MS-Visio to draw class diagrams.	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

Post-Lab Questionnaire (Lab 2.1 and 2.3)

Levels of agreement:

1 – Completely agree

2 – Generally agree

3 – Generally disagree

4 – Completely disagree

Questions:

	1	2	3	4
• The instructions and objectives of the lab were perfectly clear to me.	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
• I had plenty of time to finish the lab.	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
• I fully understood the use case descriptions I was provided.	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
• The sequence of the basic flow steps is very clear to me.	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
• The sequences of the alternative flow steps are very clear to me.	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
• The alternative flows clearly correspond to the basic flow.	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
• It is extremely hard for me to use MS-Visio to draw class diagram.	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
• It was extremely hard for me to design the class diagram. Why?	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

Pre-Lab Questionnaire (Lab 2.2 and 2.4)

Levels of agreement:

1 – Completely agree

2 – Generally agree

3 – Generally disagree

4 – Completely disagree

Questions:

	1	2	3	4
• I have good knowledge on UML sequence diagram modeling.	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
• I have substantial experience in building sequence diagrams from use cases.	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
• I fully understand how to keep the consistency between class and sequence diagrams.	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
• I am confident I can correctly use MS-Visio to draw sequence diagrams.	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

Post-Lab Questionnaire (Lab 2.2 and 2.4)

Levels of agreement:

1 – Completely agree

2 – Generally agree

3 – Generally disagree

4 – Completely disagree

Questions:

	1	2	3	4
• The instructions and objectives of the lab were perfectly clear to me.	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
• I had plenty of time to finish the lab.	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
• I fully understood the use case descriptions I was provided.	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
• The sequence of the basic flow steps is very clear to me.	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
• The sequences of the alternative flow steps are very clear to me.	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
• The alternative flows clearly correspond to the basic flow.	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
• It is extremely hard for me use MS-Visio to draw sequence diagrams.	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
• It was extremely hard for me to design the sequence diagrams. Why?	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

Appendix F Experiment data

Table 40 Error rates of the restriction rules

Measure	CPD (%)	VS (%)	Avg. (%)	Error rate (%)	Measure	CPD (%)	VS (%)	Avg. (%)	Error rate (%)
R1	1.3	1.93	1.62	2	R17-V	1.1	1	1.05	8
R2	0.87	1	0.94	1	R17-M	28.6	3	15.8	
R3	0.6	0	0.3	0	R18-V	1.4	0.3	0.85	5
R4	0.3	0	0.15	0	R18-M	18	0	9.00	
R5	0	0	0	0	R19-V	0	0	0.00	6
R6	1.5	1.6	1.55	2	R19-M	9.5	12.6	11.05	
R7	13.5	13.9	13.7	14	R20	29.5	12.8	21.15	21
R8	0.1	0.9	0.5	1	R21-V	0	0.2	0.10	6
R9	3.6	9	6.3	6	R21-M	22.2	0	11.10	
R10	0.52	1	0.76	1	R22-M1	10.4	35.6	23.00	25
R11	0.46	1	0.73	1	R22-M2	31.2	22	26.60	
R12	9	6	7.5	8	R23-V	0	0.5	0.25	14
R13	0	0	0	0	R23-M	50	5.6	27.80	
R14	0.72	2	1.36	1	R24-V	3.2	1.6	2.40	3
R15	0.1	0	0.05	0	R24-M	6.7	0	3.35	
R16	0	0	0	0	R25-V	0.8	0	0.40	13
					R25-M	41.7	11	26.35	
					R26	9.9	9.8	9.85	10

Table 41 Understandability of the restriction rules

Restriction rule	R1	R2	R3	R4	R5	R6	R7	R8	R9	R10	R11	R12	R13
Understandability (%)	90	100	97	90	90	86	90	69	97	65	79	90	69
Restriction rule	R14	R15	R16	R17	R18	R19	R20	R21	R22	R23	R24	R25	R26
Understandability (%)	79	65	100	90	100	90	76	100	100	83	79	83	

Table 42 Applicability of the restriction rules (frequencies)

Level	R1	R2	R3	R4	R5	R6	R7	R8	R9	R10	R11	R12	R13
1	0	0	0	0	0	0	4%	0	0	7%	7%	4%	15%
2	0	0	10%	0	3%	15%	4%	21%	7%	22%	14%	0	11%
1+2	0	0	10%	0	3%	15%	8%	21%	7%	29%	21%	4%	26%
3	42%	26%	38%	38%	35%	33%	55%	43%	46%	32%	36%	33%	30%
4	58%	74%	52%	62%	62%	52%	38%	36%	47%	39%	43%	63%	44%
3+4	100%	100%	90%	100%	97%	85%	92%	79%	93%	71%	79%	96%	74%

Level	R14	R15	R16	R17	R18	R19	R20	R21	R22	R23	R24	R25	R26
1	7%	14%	0	0	0	0	0	0	0	0	7%	4%	7%
2	10%	21%	0	0	0	4%	7%	20%	7%	11%	7%	11%	7%
1+2	17%	35%	0	0	0	4%	7%	20%	7%	11%	14%	15%	14%
3	38%	36%	29%	36%	33%	32%	36%	16%	32%	25%	22%	22%	39%
4	45%	29%	71%	64%	67%	64%	57%	64%	61%	64%	63%	64%	47%
3+4	83%	65%	100%	100%	100%	96%	93%	80%	93%	89%	85%	86%	86%

Table 43 Restrictiveness of the restriction rules (frequencies)

Level	R1	R2	R3	R4	R5	R6	R7	R8	R9	R10	R11	R12	R13
1	35%	52%	43%	57%	43%	33%	39%	25%	38%	41%	30%	29%	31%
2	43%	37%	34%	27%	37%	49%	44%	47%	47%	38%	38%	46%	34%
1+2	78%	89%	77%	84%	80%	82%	83%	72%	84%	79%	68%	74%	65%
3	19%	11%	14%	17%	17%	12%	14%	28%	16%	21%	32%	20%	29%
4	3%	0	9%	9%	3%	6%	3%	0	0	0	0	6%	6%
3+4	21%	11%	23%	26%	20%	18%	17%	28%	16%	21%	32%	26%	35%
Level	R14	R15	R16	R17	R18	R19	R20	R21	R22	R23	R24	R25	R26
1	29%	31%	47%	34%	35%	40%	31%	36%	31%	32%	34%	36%	32%
2	46%	45%	41%	54%	53%	49%	49%	45%	56%	50%	54%	50%	32%
1+2	74%	76%	88%	88%	88%	88%	80%	82%	86%	82%	88%	86%	64%
3	23%	24%	9%	9%	9%	9%	11%	9%	6%	3%	3%	6%	23%
4	3%	0	3%	3%	3%	3%	9%	9%	8%	15%	9%	8%	13%
3+4	26%	24%	12%	12%	12%	12%	20%	18%	14%	18%	12%	14%	36%

Table 44 Descriptive statistics of measure CD – Lab 2.1 and 2.3

Exp	System	Methods	Class Completeness			Association Completeness			CD Completeness			Class Correctness			CD Correctness			Redundancy		
			Mean	Std Dev	Size	Mean	Std Dev	Size	Mean	Std Dev	Size	Mean	Std Dev	Size	Mean	Std Dev	Size	Mean	Std Dev	Size
1	CPD+VS	UCM_R	0.48	0.17	14	0.11	0.11	14	0.26	0.12	14	0.79	0.13	14	0.88	0.07	14	0.09	0.03	14
		UCM_UR	0.37	0.2	9	0.05	0.07	9	0.18	0.12	9	0.65	0.15	9	0.81	0.07	9	0.14	0.04	9
2.1	CPD	UCM_R	0.66	0.18	8	0.24	0.21	8	0.36	0.06	8	0.76	0.05	8	0.62	0.15	8	0.21	0.07	8
		UCM_UR	0.45	0.12	9	0.08	0.11	9	0.24	0.03	9	0.71	0.08	9	0.64	0.18	9	0.32	0.11	9
	VS	UCM_R	0.64	0.13	9	0.14	0.22	9	0.39	0.15	9	0.86	0.04	9	0.67	0.24	9	0.17	0.08	9
		UCM_UR	0.59	0.07	7	0.04	0.08	7	0.32	0.05	7	0.77	0.11	7	0.73	0.17	7	0.18	0.08	7
	CPD+VS	UCM_R	0.65	0.15	17	0.19	0.22	17	0.38	0.16	17	0.81	0.06	17	0.65	0.2	17	0.19	0.08	17
		UCM_UR	0.51	0.12	16	0.06	0.1	16	0.27	0.09	16	0.76	0.1	16	0.68	0.18	16	0.26	0.12	16
2.3	CPD	UCM_R	0.65	0.1	6	0.21	0.09	6	0.36	0.12	6	0.81	0.03	6	0.51	0.19	6	0.1	0.06	6
		UCM_UR	0.45	0.12	8	0.1	0.11	8	0.23	0.11	8	0.75	0.1	8	0.6	0.16	8	0.2	0.1	8
	VS	UCM_R	0.7	0.14	9	0.1	0.13	9	0.4	0.12	9	0.88	0.06	9	0.83	0.13	9	0.11	0.1	9
		UCM_UR	0.65	0.17	9	0.12	0.12	9	0.39	0.14	9	0.83	0.04	9	0.76	0.17	9	0.16	0.11	9
	CPD+VS	UCM_R	0.68	0.12	15	0.14	0.13	15	0.37	0.11	15	0.85	0.06	15	0.7	0.22	15	0.1	0.08	15
		UCM_UR	0.56	0.18	17	0.11	0.11	17	0.31	0.15	17	0.8	0.08	17	0.68	0.18	17	0.18	0.11	17

Table 45 Descriptive statistics of measure SD – Lab 2.2 and 2.4

Exp	System	Methods	Message Completeness			SD Completeness			Message Correctness			SD Correctness			BCE Consistency		
			Mean	Std Dev	Size	Mean	Std Dev	Size	Mean	Std Dev	Size	Mean	Std Dev	Size	Mean	Std Dev	Size
2.2	CPD	UCM_R	0.45	0.05	8	0.67	0.13	8	0.73	0.26	8	0.83	0.14	8	0.95	0.06	8
		UCM_UR	0.41	0.13	7	0.61	0.17	7	0.49	0.21	7	0.67	0.18	7	0.91	0.15	7
	VS	UCM_R	0.39	0.11	7	0.5	0.18	7	0.72	0.18	7	0.83	0.1	7	1	0.01	7
		UCM_UR	0.36	0.12	6	0.51	0.1	6	0.72	0.21	6	0.79	0.13	6	0.95	0.07	6
	CPD+VS	UCM_R	0.42	0.09	15	0.59	0.17	15	0.73	0.06	15	0.83	0.12	15	0.97	0.05	15
		UCM_UR	0.39	0.12	13	0.56	0.15	13	0.6	0.07	13	0.72	0.17	13	0.93	0.12	13
2.4	CPD	UCM_R	0.52	0.16	6	0.77	0.18	6	0.73	0.23	6	0.76	0.17	6	0.9	0.07	6
		UCM_UR	0.46	0.1	8	0.61	0.22	8	0.6	0.22	8	0.74	0.17	8	0.9	0.09	8
	VS	UCM_R	0.59	0.2	8	0.78	0.14	8	0.84	0.1	8	0.88	0.07	8	0.99	0.03	8
		UCM_UR	0.46	0.1	8	0.63	0.1	8	0.61	0.31	8	0.81	0.12	8	0.98	0.03	8
	CPD+VS	UCM_R	0.56	0.18	14	0.77	0.15	14	0.79	0.17	14	0.83	0.13	14	0.95	0.11	14
		UCM_UR	0.46	0.1	16	0.62	0.16	16	0.61	0.23	16	0.78	0.15	16	0.94	0.18	16

Table 46 ANOVA – Interaction between *Method* (UCM_R vs. UCM_UR) and *System* (CPD vs. VS) for CD – Lab 2.1 and Lab 2.3

Exp.	Measures	Factor	DF	Parameter estimate	Sum of Square	F Ratio	Prob > F
2.1	Class Completeness	Method	1	0.0659	0.1419	8.0427	0.0082
		System	1	-0.0299	0.0292	1.6534	0.2087
		Method * System	1	0.0405	0.0535	3.0308	0.0923
	CD Completeness	Method	1	0.05	0.0816	4.8629	0.0355
		System	1	-0.0286	0.0267	1.5893	0.2175
		Method * System	1	0.0119	0.0046	0.2742	0.6045
	Class Correctness	Method	1	0.0368	0.0442	7.877	0.0089
		System	1	-0.0388	0.0491	8.7481	0.0061
		Method * System	1	-0.0079	0.0021	0.3664	0.5497
	Redundancy	Method	1	-0.0289	0.0272	3.3946	0.0757
		System	1	0.0448	0.0655	8.1658	0.0078
		Method * System	1	-0.0234	0.0179	2.2286	0.1463
2.3	Class Completeness	Method	1	0.0621	0.1199	6.2663	0.0184
		System	1	-0.0645	0.1297	6.7772	0.0146
		Method * System	1	0.0343	0.0366	1.9125	0.1776
	CD Completeness	Method	1	0.0381	0.0451	3.0662	0.0909
		System	1	-0.0505	0.0794	5.3932	0.0277
		Method * System	1	0.0297	0.0275	1.8709	0.1823
	Class Correctness	Method	1	0.0248	0.0191	4.5373	0.0421
		System	1	-0.0376	0.0441	10.4934	0.0031
		Method * System	1	0.0023	0.0002	0.0405	0.8419
	Redundancy	Method	1	-0.0411	0.0526	5.245	0.0297
		System	1	0.0089	0.0025	0.2475	0.6227
		Method * System	1	-0.0146	0.0066	0.6592	0.4237

Table 47 ANOVA – Interaction between *Method* (UCM_R vs. UCM_UR) and *Order* (2.1 vs. 2.3) on CD – Lab 2.1 and Lab 2.3

Measures	Factor	DF	Parameter estimate	Sum of Square	F Ratio	Prob > F
Class Completeness	Method	1	0.0661	0.2833	13.2282	0.0006
	Order	1	-0.0186	0.0225	1.0499	0.3096
	Method * Order	1	0.0039	0.001	0.0465	0.8300
CD Completeness	Method	1	0.0454	0.1336	7.8038	0.007
	Order	1	-0.012	0.0093	0.5415	0.4646
	Method * Order	1	0.0076	0.0038	0.2195	0.6411
Class Correctness	Method	1	0.0336	0.0732	11.83	0.0011
	Order	1	-0.025	0.0406	6.5542	0.013
	Method * Order	1	0.0065	0.0027	0.4432	0.5081
Redundancy	Method	1	-0.0368	0.0878	8.8676	0.0042
	Order	1	0.0412	0.1098	11.0882	0.0015
	Method * Order	1	0.003	0.0006	0.8062	0.8062

Table 48 ANOVA – Interaction between Method (UCM_R vs. UCM_UR) and System (CPD vs. VS) for SD – Experiment 2

Exp.	Measures	Factor	DF	Parameter estimate	Sum of Square	F Ratio	Prob > F
2.2	Message Completeness	Method	1	0.016	0.007	0.6358	0.4331
		System	1	0.027	0.021	1.8732	0.1838
		Method * System	1	0.005	0.001	0.8139	0.8139
	SD Completeness	Method	1	0.012	0.004	0.1682	0.6854
		System	1	0.07	0.137	6.1079	0.0209
		Method * System	1	0.016	0.007	0.3289	0.5717
	Sequence Correctness	Method	1	0.06	0.098	2.0042	0.1689
		System	1	-0.057	0.09	1.8261	0.1892
		Method * System	1	0.059	0.097	1.9723	0.173
	SD Correctness	Method	1	0.051	0.073	3.5595	0.0714
		System	1	-0.027	0.021	1.0218	0.3222
		Method * System	1	0.029	0.023	1.1171	0.3011
BCE Consistency	Method	1	0.022	0.014	1.6849	0.2066	
	System	1	-0.023	0.015	1.8457	0.1869	
	Method * System	1	0.0004	<0.0001	0.0006	0.9802	
2.4	Message Completeness	Method	1	0.047	0.064	2.9904	0.0956
		System	1	-0.017	0.009	0.4007	0.5322
		Method * System	1	-0.015	0.007	0.3176	0.5779
	SD Completeness	Method	1	0.077	0.176	6.5768	0.0165
		System	1	-0.007	0.001	0.0552	0.816
		Method * System	1	0.0004	0.000005	0.0002	0.9891
	Message Correctness	Method	1	0.089	0.234	4.5339	0.0429
		System	1	-0.03	0.026	0.5038	0.4842
		Method * System	1	-0.025	0.019	0.3662	0.5503
	SD Correctness	Method	1	0.021	0.012	0.6667	0.4216
		System	1	-0.048	0.067	3.5879	0.0694
		Method * System	1	-0.013	0.005	0.2515	0.6202
	BCE Consistency	Method	1	0.004	0.001	0.0207	0.8866
		System	1	-0.051	0.051	2.2964	0.1417
		Method * System	1	<0.0001	<0.0001	0.0001	0.9904

Table 49 ANOVA – Interaction between Method (UCM_R vs. UCM_UR) and Order (2.2 vs. 2.4) on SD – Lab 2.2 and Lab 2.4

Measures	Factor	DF	Parameter estimate	Sum of Square	F Ratio	Prob > F
Message Completeness	Method	1	0.025	0.037	0.8815	0.352
	Order	1	-0.222	2.83	67.5539	<.0001
	Method * Order	1	-0.009	0.005	0.1153	0.7355
SD Completeness	Method	1	0.053	0.16	4.1777	0.0458
	Order	1	-0.062	0.223	5.8071	0.0194
	Method * Order	1	-0.04	0.093	2.4363	0.1244
Message Correctness	Method	1	0.044	0.114	3.2296	0.0779
	Order	1	-0.07	0.282	7.9774	0.0066
	Method * Order	1	0.02	0.022	0.6302	0.4308
SD Correctness	Method	1	0.03	0.052	2.4339	0.1246
	Order	1	-0.083	0.396	18.4816	<.0001
	Method * Order	1	0.023	0.031	1.4446	0.2346
BCE Consistency	Method	1	0.022	0.028	2.0863	0.1544
	Order	1	0.024	0.033	2.4192	0.1257
	Method * Order	1	0.0003	<0.0001	0.0005	0.983

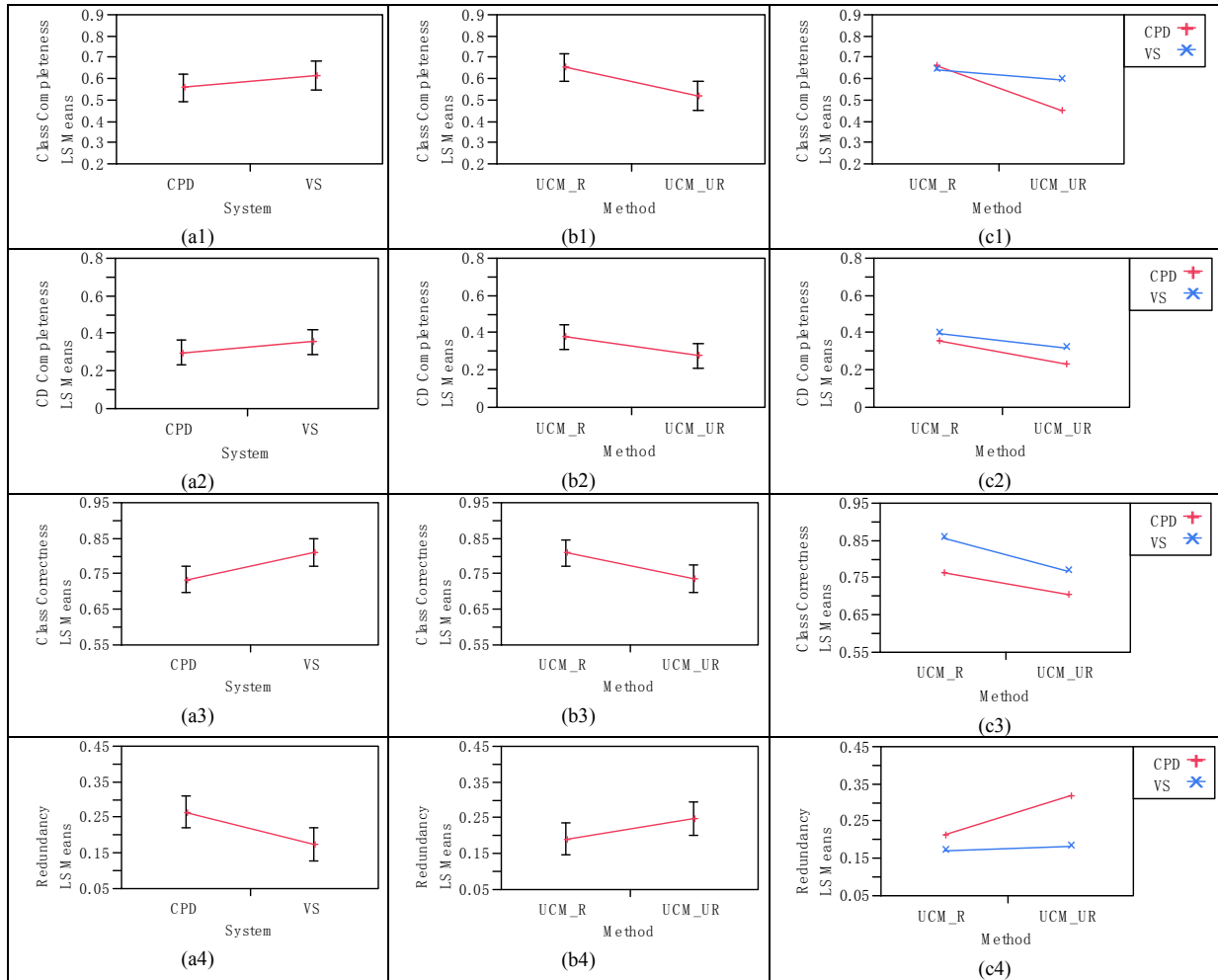


Figure 28 Least-square means for ANOVA interaction analysis between *Method* and *System* for CD - Lab 2.1

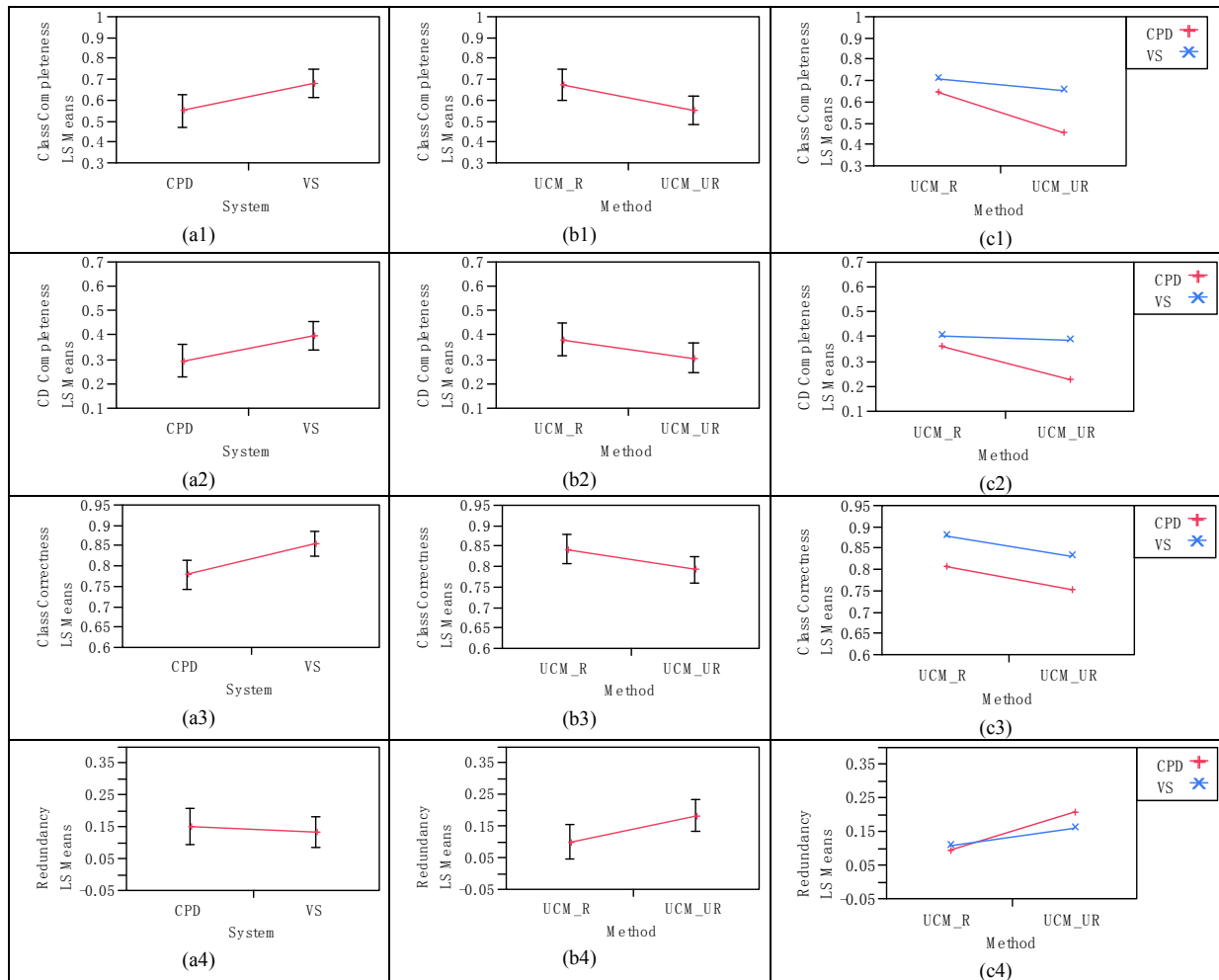


Figure 29 Least-square means for ANOVA interaction analysis between *Method* and *System* for CD - Lab 2.3

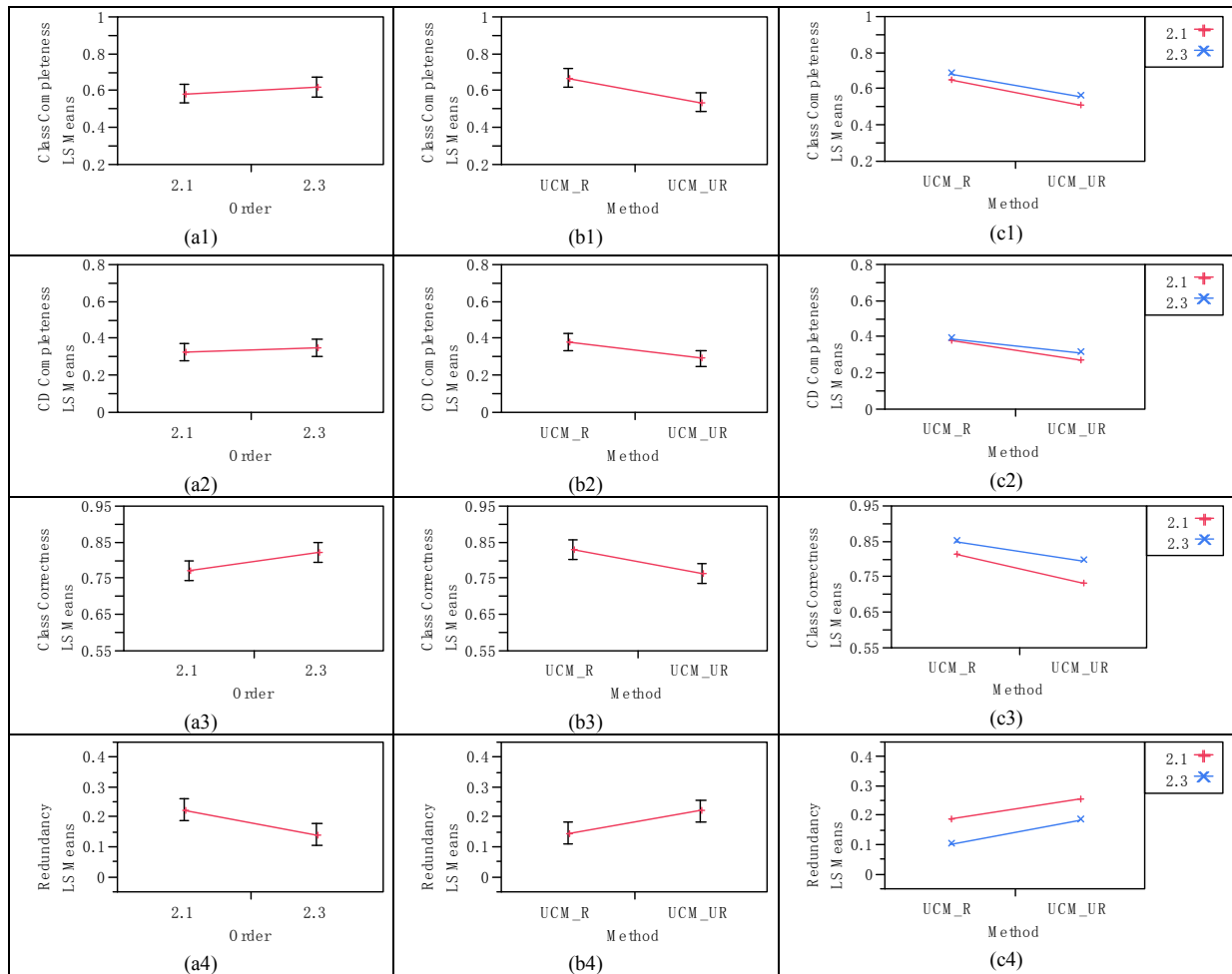


Figure 30 Least-square means for ANOVA interaction analysis between Method and Order for CD - Lab 2.1 and Lab 2.3

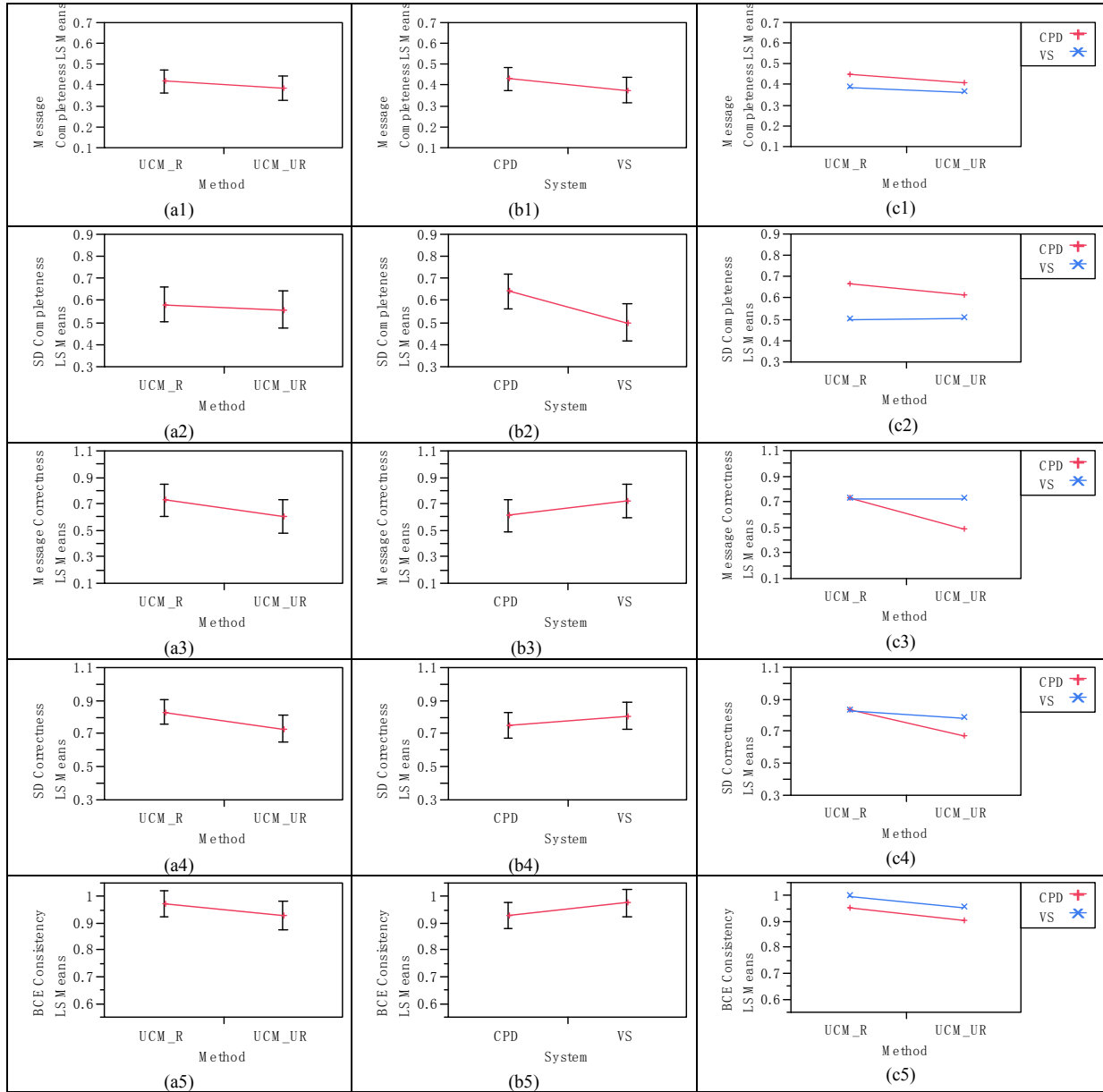


Figure 31 Least-square means for ANOVA interaction analysis between *Method* and *System* for SD - Lab 2.2

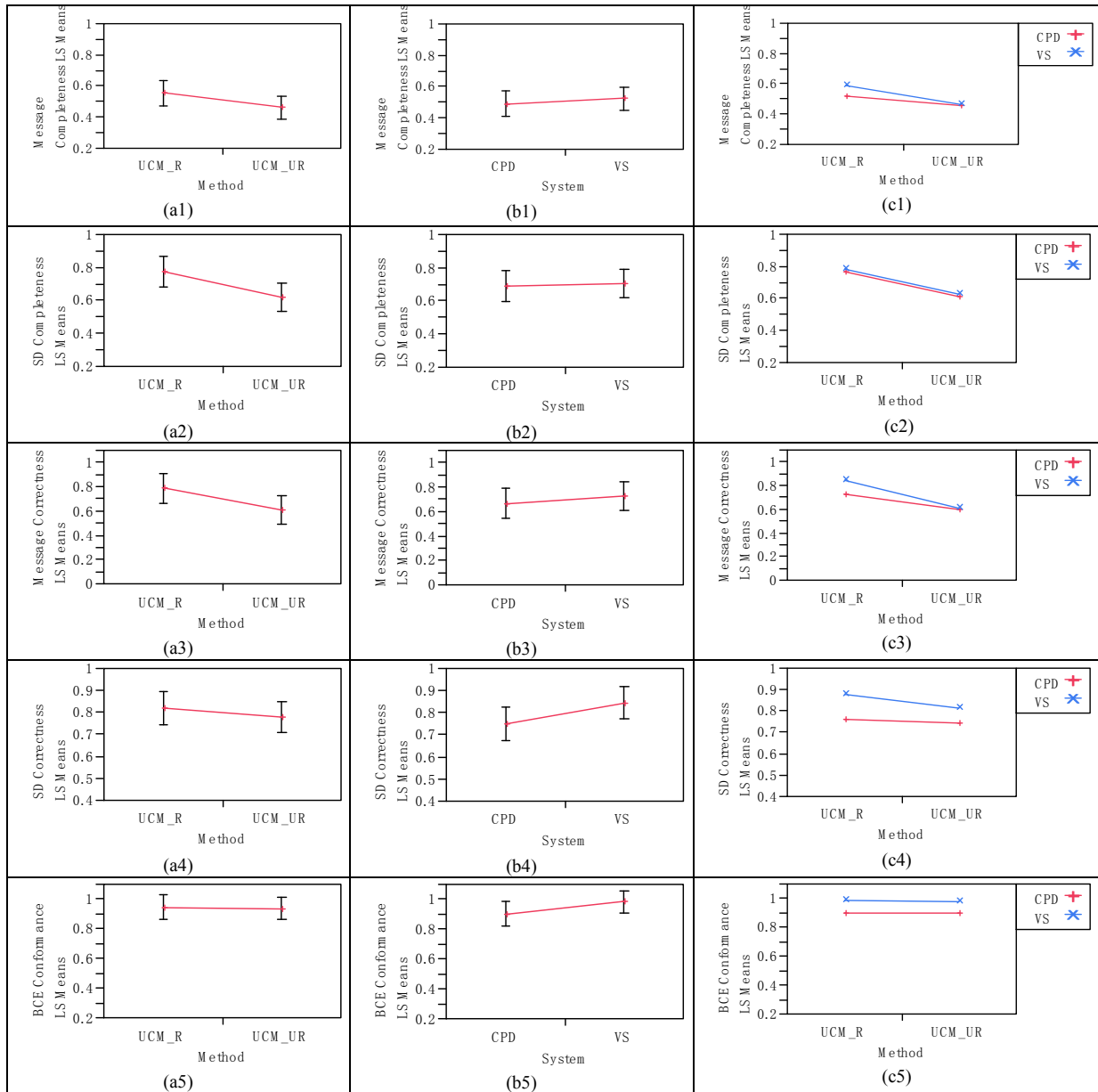


Figure 32 Least-square means for ANOVA interaction analysis between *Method* and *System* for SD - Lab 2.4

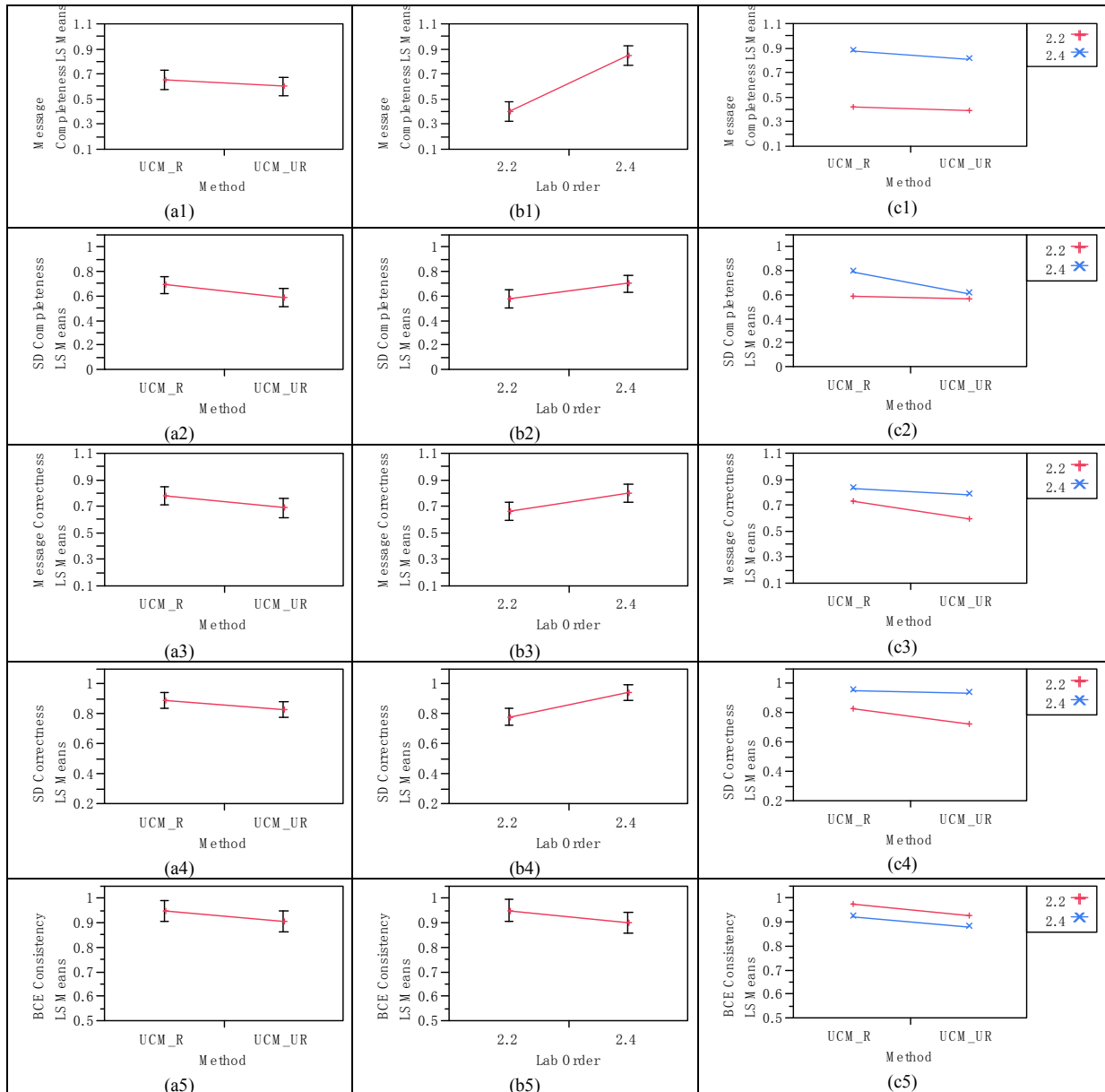


Figure 33 Least-square means for ANOVA interaction analysis between *Method* and *Order* for SD - Lab 2.2 and Lab 2.4