Information and Software Technology 131 (2021) 106484

Contents lists available at ScienceDirect

Information and Software Technology

journal homepage: www.elsevier.com/locate/infsof

On deriving conceptual models from user requirements: An empirical study

Fabiano Dalpiaz [*],[a], Patrizia Gieske [a], Arnon Sturm [b]

[a] Utrecht University, the Netherlands
[b] Ben-Gurion University of the Negev, Israel

ARTICLE INFO

Keywords:
Requirements engineering
Conceptual modeling
Use cases
User stories
Derivation process

ABSTRACT

Context: There are numerous textual notations and techniques that can be used in requirements engineering. Currently, practitioners make a choice without having scientific evidence regarding their suitability for given tasks. This uninformed choice may affect task performance. **Objective:** In this research, we investigate the adequacy of two well-known notations: use cases and user stories, as a starting point for the manual derivation of a structural conceptual model that represents the domain of the system. We also examine other factors that may affect the performance of this task. **Methods:** This work relies on two experiments. The first is a controlled classroom experiment. The second one is a quasi-experiment, conducted over multiple weeks, that aims at evaluating the quality of the derived conceptual model in light of the notation used, the adopted derivation process, and the complexity of the system to be. We measure quality in terms of validity and completeness of the conceptual model. **Results:** The results of the controlled experiment indicate that, for deriving conceptual models, user stories fit better than use cases. Yet, the second experiment indicates that the quality of the derived conceptual models is affected mainly by the derivation process and by the complexity of the case rather than the notation used. **Contribution:** We present evidence that the task of deriving a conceptual model is affected significantly by additional factors other than requirements notations. Furthermore, we propose implications and hypotheses that pave the way for further studies that compare alternative notations for the same task as well as for other tasks. Practitioners may use our findings to analyze the factors that affect the quality of the conceptual model when choosing a requirements notation and an elicitation technique that best fit their needs.

1. Introduction

Over the years, researchers and practitioners have proposed numerous notations for expressing requirements for software systems, ranging from sentences in natural language [1], graphical and semi-formal models [2,3], to formal languages [4,5]. Within this landscape, free or templatic notations are very common [6], with examples such as the Easy Approach to Requirements Syntax (EARS) [7], UML use cases, and user stories [8]. The adequacy of the notation depends on the type of system, the application domain, and the granularity of the requirements. Unfortunately, the research community has largely overlooked the contextual adequacy of these notations, thereby offering no evidence for practitioners on the choice of an effective notation.

The selection of a notation also depends on other factors, such as the role of the specification within the software team. More informal notations are better suitable for high-level specifications that foster communication among stakeholders; this is also a reason why user stories became so prevalent in agile development [9]. More constrained notations may be useful for analytical tasks such as finding inconsistencies [10], or when the requirements are a starting point for model-driven development [11].

We focus on the task of refining high-level, user requirements[1] through the construction of a conceptual model that represents the major entities, and relationships that are referred to in the requirements [11–13]. A conceptual model [14] can be seen as a collection of statements, often represented via a graphical model [15], that specify a given domain. Specifically, we are concerned with static/structural conceptual models that represent the domain entities and their relationships. Such conceptual models can be employed in requirements engineering to (i) provide an overview for team members to understand the product domain [16,17]; (ii) identify quasi-synonyms that may lead to misunderstandings [18]; (iii) support model-driven engineering [11,19]; and (iv) analyze certain quality aspects such as security and privacy [20].

* Corresponding author.
E-mail address: f.dalpiaz@uu.nl (F. Dalpiaz).

[1] By user requirement, we refer to a requirement expressed from the perspective of a user.

https://doi.org/10.1016/j.infsof.2020.106484
Received 15 July 2020; Received in revised form 2 November 2020; Accepted 6 November 2020
Available online 12 November 2020

We study the manual derivation of a conceptual model (in particular, in the form of a UML class diagram) that represents the main concepts in a collection of high-level requirements. These models have shown to be a useful learning tool for new employees who recently joined a development team [17], for representing the domain in which the system will be deployed [16], and for supporting the transition to later phases in object-oriented software development [21,22].

We investigate the fitness of two mainstream notations in terms of how well they support analysts in manually extracting conceptual models. Our main research question in this paper is as follows: ***MRQ.*** *How does the choice of a requirements notation affect the quality of derived static conceptual models?*

The secondary research question we examine in this paper is as follows: ***SRQ.*** *Are there other factors that affect the quality of derived static conceptual models? And to what extent?*

The user requirements notations that we choose are use cases (UC) and user stories (US). The former are chosen because they are part of the UML and, despite their suitability to express requirements has been challenged [23], they are widely adopted in the software industry [6]. The latter are chosen because they are by far the most popular notation in projects that follow agile development methods like Scrum and Kanban [9,24].

We used a mixed approach to answers the research questions. First, we performed a controlled experiment [25] that we report in the first part of this paper. Subjects were briefed to individually derive UML class diagrams starting from high-level requirements for two systems using either notation (US and UC). By defining a gold standard conceptual model, we measure the completeness and validity of the subjects' derived models. Furthermore, we evaluate the preference of the subjects in extracting models from either notation.

Next, we performed a second, longer, quasi-experiment in which the subjects practiced requirements elicitation through interviews, created a requirements specification using either use cases or user stories, and then derived static conceptual models based on their own specifications. We measure the quality of the conceptual models in light of expert-generated conceptual models that were built for each subject based on that subject's specification. In this experiment, we further emphasize the analysis of additional factors other than the notation that may affect the quality of the derived conceptual model (as stated in SRQ).

The results of the first experiment indicate that, in a course where object orientation is explained in detail, user stories seem to be preferred for the task at hand. Besides such preference, the quality (in terms of completeness and validity) of the derived models tends to be higher with user stories. The second experiment, instead, suggests that the quality of the models mainly depends on the derivation process or the complexity of the case rather than the notation. Among our findings, in cases where the subjects applied a disciplined derivation process, the alignment with the expert-generated model was higher.

Besides providing initial evidence on the factors that affect the derivation of conceptual models from requirements, our results aim to inspire other research on the effectiveness of alternative requirements notations and requirement engineering techniques for different requirements-related tasks. By doing so, we hope the research community can provide scientifically grounded guidance for practitioners to choose their notations and techniques.

The rest of the paper is organized as follows. In Section 2, we set the background for this study and review related works. In Section 3, we shortly present the controlled experiment, which is fully described in a previous paper [25]. In Section 4, we elaborate on the quasi-experiment, whereas in Section 5, we interpret and discuss the results of both studies. We conclude and set plans for future research in Section 6.

2. Background and related work

We present the baseline of our work: use cases and user stories, in Section 2.1. Then, we discuss related literature in Section 2.2.

2.1. Research baseline

As part of the Unified Modeling Language (UML) [26], use cases were set as a means for expressing requirements that describe the interaction between a user and a system. Although typically used for expressing functional requirements, use cases have been extended to make them suitable for representing quality aspects such as security and privacy [20,27]. A use case defines scenarios in which actors and the system interact. A use case is specified following a textual template. The set of use cases is organized and presented using a use case diagram.

In our work, we focus on a simple template notation adapted from Larman's book [28], which is based on the widely used notations by Cockburn [29] and Kruchten [30]. Listing 1 illustrates the notation on the requirements for the Planning Poker game website from the Scrum Alliance, showing how a moderator sets up a game among estimators. Note that we focus on use case scenarios and not the use case diagrams that only provide an overview of the system to be.

User stories are another widespread notation [9,24], especially in projects that adhere to agile software development [8]. User stories are simple descriptions of a feature written from the perspective of the stakeholder who wants such a feature. Multiple templates exist for representing user stories [31], among which the Connextra format is the most common [9]: *As a* <*role*>, *I want* <*action*>, *so that* <*benefit*>. The "so that" part, despite its importance in providing the rationale for a user story [32], is often omitted in practice. We consider user stories that are formulated using the Connextra template, and we group related user stories into epics. See Listing 2 for some examples regarding the Planning Poker website.

UC1. Set a Game
Primary Actor: Moderator
Main Success Scenario (or Basic Flow):
1. Create a new game by entering a name and an optional description
2. The system records the game parameters
3. Set the estimation policy
4. The system stores the estimation policy
5. Invite up to 15 estimators to participate
6. The system sends invitations and add estimators to the game

Listing 1. A use case for the Planning Poker game website.

Epic. Set a Game
US1: As a moderator, I want to create a new game by entering a name and an optional description, so that I can start inviting estimators.
US2: As a moderator, I want to invite estimators, so that we can start the game.
US3: As a moderator, I want to have the "estimate" field filled in automatically if all estimators show the same card, so that I can accept it more quickly.
US4: As a moderator, I want to enter the agreed-upon estimate, so that we can move on to the next item when we agree.

Listing 2. Some user stories for the Planning Poker game website.

2.2. Related work

After providing an overview of conceptual modeling (Section 2.2.1), we discuss the derivation of conceptual models from use cases (Section 2.2.2), from user stories (Section 2.2.3), the automated extraction of conceptual models from textual notations (Section 2.2.4), and existing experiments that compare multiple notations (Section 2.2.5).

2.2.1. Conceptual modeling

Conceptual modeling emerged in the mid-1970s as an important activity of database and information system design; pioneering works were Abrial's semantic model [33] and Chen's entity-relationship model [34]. Conceptual models, as argued by Wand and Weber [35], are mostly *graphic models* that represent both static phenomena and dynamic phenomena in *some domain*. Mylopoulos [15] claims that conceptual models *formally* describe some aspects of the physical and social world around us for *purposes of understanding and communication*.

Lindland et al. [14] characterize the notion of *quality* in conceptual models and they distinguish between three aspects of quality: (i) *syntactic quality* concerns how well a model employs a given language; (ii) *semantic quality* is about how well the model portrays the modeled domain; and (iii) *pragmatic quality* regards how well the model corresponds to its audience interpretation.

In this paper, we focus on graphical conceptual models that denote a domain in structural terms (entities and relationships) and that can be used to facilitate the understanding of a domain among stakeholders in requirements engineering. Furthermore, in our measurements of model quality, we limit our attention to semantic quality.

2.2.2. From use cases to conceptual models

Insfrn et al. [12] propose a process that assists in the refinement of high-level requirements into lower-level models that can be automatically mapped to code. Part of their approach is the creation of use cases to facilitate the transition from natural language statements to executable models.

Yue et al. [36] observe that informal use case specifications may contain vagueness and ambiguity that hamper the derivation of precise UML models, including class and sequence diagrams. To overcome this problem, they propose a restricted version of the use case template. The approach was found easy to apply by practitioners and led to improvements in terms of class validity and class diagram completeness.

Anda and Sjberg [37] describe two object-oriented modeling techniques: (i) a *derivation technique* in which a class diagram is directly derived from use cases, (ii) a *validation technique* in which an initial class diagram is created from the textual requirements independently of the use cases and subsequently validated and improved by applying the use case model. Both techniques were empirically compared in two separate experiments. An experiment with students indicated that the designed model represented as a class diagram was more complete, although less structured when applying the validation technique. In an experiment with experts, the differences were not significant.

Fortuna et al. [38] propose an integrated requirements model called *info case*. As a specialization of use case models, info cases are intended for deriving domain models with higher consistency and increased usability. In a comparative experiment involving six subjects, the model transformation from use case models to domain models (i.e., conceptual models) is compared to the model transformation from info case models to domain models. The results indicate that info cases can reduce the heterogeneity among several domain models derived from the same use case model.

2.2.3. From user stories to conceptual models

Fewer methods exist that derive conceptual models from user stories. Lucassen et al. [17] propose an automated approach based on the Visual Narrator tool for extracting conceptual models from a set of user stories. Their work relies on and adapts natural language processing heuristics from the literature. The resulting models show good precision and recall, also thanks to the syntactic constraints imposed by user stories, although perfect accuracy is not possible due to the large variety of linguistic patterns that natural language allows for.

Wautelet et al. [39] introduce a process for transforming a collection of user stories into a use case diagram by using the granularity information obtained through tagging the user stories. Instead of comparing notations, they focus on the joint use of two notations, one textual and one diagrammatic.

The same research group [40] proposed one of the few studies on the construction of diagrams from user stories. In particular, they investigate the construction of a goal-oriented model (a rationale tree) that links the who, what, and why dimensions of a user story. Their research shows differences depending on the modeler background and other factors. While their work is highly related, we focus on a less demanding task, which focuses only on the what dimension: the derivation of a static conceptual model.

Trkman et al. [41] point out how user stories, being defined independently, fail to highlight execution and integration dependencies. As a solution, they propose the use of a different type of business process models to associate user stories with activities and, thus, to facilitate the discovery of dependencies by following the control flow in the business process model.

2.2.4. Automated extraction of conceptual models

The extraction of conceptual models from natural language description requirements is one of the four types of NLP tools described by Berry et al. [42] and a long-standing research thread.

Saeki et al. described a method where verbs and nouns are automatically extracted from natural language, thereby assisting a requirements engineer in the derivation of a formal specification of a system [43]. NL-OOPS [44] is an early tool that implements Saeki's idea. CM-builder [45] improves over the early results of NL-OOPS and extracts candidate attributes, entities, and relationships with 66% precision and 73% recall. CIRCE [46] goes further by supporting the generation of many types of models. Arora et al. [47] combine existing state-of-the-art heuristics for domain model extraction. They apply their approach to four industrial requirements documents. According to expert evaluation, their implementation achieves validity between 74% and 100%.

Some studies evaluated tooling against humans. The aToucan tool [22] generates high-quality class diagrams from restricted use case models in comparison to diagrams created by experts, managing to consistently outperform fourth-year software engineering students in terms of completeness, consistency, and redundancy. However, this result is facilitated by restricting the use cases language, which becomes a controlled language. Similarly, the tool presented in [48] outperforms novice human modelers in generating conceptual models from natural language requirements. Again, this result depends on setting constraints to the notation that is used to express the requirements.

Although these tools can perform as good as novice human modelers when the syntax is sufficiently restricted, we take a different perspective, in which we do not wish to over constrain the humans in the specification of the requirements.

2.2.5. Experimental studies across notations

To the best of our knowledge, there are no experimental studies that compare the effectiveness of requirements notations, in particular, use cases and user stories. The closest work to ours empirically assesses two different *derivation techniques* on the quality of conceptual models that are derived from requirements specifications [49]. They compare a text-based derivation technique based on the OO-Method [50] against a communication-based derivation technique. No differences between the techniques are observed regarding the resulting model validity. Nevertheless, the results show an interaction between the modeling competence of the subjects and the derivation techniques. Significantly higher completeness for the conceptual models derived with the communication-based technique can be observed in the high competency group. In contrast, no significant difference was found in the average competency group. Unlike their work, we do not provide the subjects with rigid guidelines for deriving the models.

Apart from that study, the closest works to ours regard the comparison of (graphical) notations used in information systems design. Ottensooser et al. [51] compare the Business Process Modeling Notation (BPMN) against textual use cases in interpreting business process descriptions. Their experiment shows that BPMN adds value only with trained readers. Cardoso et al. [52] conduct an experiment that shows how the adequacy of modeling languages depends on how structured a business process. Hoisl et al. [53] compare three notations (textual, semi-structured, diagrammatic) for expressing scenario-based model tests; their experimental results show a preference toward natural language-based notations.

The relative paucity of empirical studies investigating the impact of the choice of requirements notations and other factors on the quality of static conceptual models motivates this experimental study.

3. The controlled experiment

We conducted a controlled experiment through which we analyze the quality of static conceptual models derived from equivalent requirements expressed using two requirements notations: use cases and user stories. We measure model quality via *validity* and *completeness* with respect to a gold standard solution. Furthermore, we collect and compare the preference of the subjects with respect to the use of the two notations for various tasks.

The starting point of our research is the following null hypothesis, in which we do not assume a difference exists between the notations:

H_0: user stories and use cases are equally good for the derivation of a static conceptual model.

3.1. Experiment design

We describe the variables and their measurements, the subjects, and the tasks.

Independent Variables The first variable is the notation (IV1) according to which the requirements are specified. It has two possible values: User Stories (US) and Use Cases (UC). The second independent variable is the case study used (IV2). It has two possible values: Data Hub (DH) and Planning Poker (PP) [54]. DH is the specification for the web interface of a platform for collecting, organizing, sharing, and finding data sets. PP are the requirements for the first version of the *planningpoker.com* website, an online platform for estimating user stories using the Planning Poker technique.

Dependent Variables There are two dependent variables, taken from conceptual modeling research [14,49], that we use for measuring the quality of a generated conceptual model. These variables are specified by comparing the elements in the *subject solution* (the conceptual model derived by a subject) against the *gold standard solution* :

- *Validity (DV1)*: the ratio between the number of elements in the subject solution that also exist in the gold standard (true positives) and the true positives plus the number of elements in the subject's solution that do not exist within the gold standard solution (false positives). In information retrieval terms, validity equates to precision.

$$Validity = \frac{|True\ Positives|}{|True\ Positives| + |False\ Positives|}$$

- *Completeness (DV2)*: the ratio between the number of elements in the subject solution that also exist in the gold standard (true positives) and the number of elements in the gold standard (true positives + false negatives). In information retrieval terms, completeness equates to recall.

$$Completeness = \frac{|True\ Positives|}{|True\ Positives| + |False\ Negatives|}$$

While measuring completeness and validity, we refer to various ways of counting the elements of a conceptual model:

- Number of entities, i.e., classes;
- Number of relationships between classes;
- Total: number of entities + number of relationships.

Since relationships can only be identified when the connected entities are identified, we introduce an *adjusted* version of validity and completeness for the relationships, which calculates completeness and validity with respect to those relationships in the gold standard among the entities that the subject has identified. For example, if the gold standard has entities A, B, C with relationships R1(A,B), R2(B,C) and R3 (A,C), but the subject has identified only A and C, then only R3 is considered for computing validity and completeness in the *adjusted* version.

Subjects We involved third-year students taking the course on *Object-Oriented Analysis and Design* at Ben-Gurion University of the Negev. The course teaches how to analyze, design, and implement software based on the object-oriented paradigm. In the course, the students-subjects learned the notion of modeling and, in particular, class diagrams. The instructor of the course was the third author of this paper. They learned user stories and use cases for specifying requirements as part of the development process. They also practiced class diagrams, use cases, and user stories through homework assignments, in which they achieved good results, indicating that they understood the notations well.

Task We designed the experiment so that each subject would experience the derivation of a conceptual model from both notations. For that purpose, we designed two forms (available online [55]), in which we alternate the treatment and the case study.

The form has 4 parts: (1) a pre-task questionnaire that checks the subjects' background and knowledge; (2) the first task, in which subjects receive the requirements of the Data Hub application, specified either in use cases or user stories, and were asked to derive a conceptual model; (3) the second task, in which subjects receive the requirements of the Planning Poker application, specified either in use cases or user stories, and were asked to derive a conceptual model; (4) questions about the subjects' perception of the two notations and their usefulness. We asked the subjects to derive a conceptual model that would serve as a domain model for the backbone of the system to be developed (as taught in the course).

Execution Before executing the experiment, we did a pilot with the course teaching assistant so to confirm the task readability and comprehension and the clarity of requirements (both for the user stories and the use cases). The experiment took place in a dedicated time slot of two hours. All subjects, however, finished in roughly 1 h. The assignment of the groups (i.e., the forms) to the 118 subjects was done randomly. The distribution of groups was as follows:

- Form A: DH with user stories and PP with use cases: 57 subjects;
- Form B: DH with use cases and PP with user stories: 61 subjects.

3.2. Experiment results

Prior to the data analysis, we compared the subjects in both groups and found them with similar competencies, though there was a difference in their GPAs. We ran a series of analyses over the results (all materials are online [55]). In analyzing the results of the completeness and validity of the conceptual models, we performed an ANOVA test [56] and found out that the interaction between the case study (IV2) and the notation (IV1), concerning the adjusted total validity and completeness, is statistically significant. This interaction probably occurred due to the complexity differences between the two case studies. We thus analyze each case study separately.

Table 1 a and b present the results of the DH and PP case studies, respectively. For the user stories and the use cases columns, we report arithmetic mean and standard deviation for the related metric. Bold numbers indicate the best results, whereas the statistically significant differences (applying T-test) are highlighted using one or two bullets, to denote $p < .05$ and $p < .01$, respectively. We also denote the effect size of *intermediate range* via one star, when Hedges' $g \geq 0.5$ [57].

In all metrics in both case studies, the conceptual models derived from the set of user stories outperform the conceptual models derived from the set of use cases (see Table 1a and b). For the DH case study, the difference was statistically significant in the case of the relationships for all the adjusted metrics as well as for the total validity. Furthermore, the effect sizes for DH indicate a *medium effect* for many metrics (all those with $g \geq 0.5$).

Based on the results, we can conclude that for the Data Hub application, we can reject the H_0 hypotheses on the equality of both notations in the quality of the derived conceptual model, in particular with respect to relationship validity and completeness, total validity and all adjusted metrics. For the other metrics, in the DH case, introducing user stories, resulted in better conceptual models. For the other metrics (entity completeness and validity, total completeness, as well as all metrics for the PP case), we cannot reject the H_0 hypotheses: the notation seems to not impact the quality of the derived conceptual model.

Table 1
Results from the first experiment. Legend: $^{\bullet}$ denotes significance with $p < .05$; $^{\bullet\bullet}$ indicates significance with $p < .01$; * denotes a medium effect: $g \geq 0.5$.

(a) Data Hub

	User Stories		Use Cases		Sig.	Effect
	$\bar{x}$	σ	$\bar{x}$	σ	p	size r
Entity Completeness	**0.73**	0.13	0.70	0.14	0.258	0.222
Entity Validity	**0.66**	0.14	0.61	0.12	0.089	0.384
Relation Completeness	**0.38**	0.15	0.34	0.12	0.047$^{\bullet}$	0.296
Relation Validity	**0.34**	0.14	0.29	0.10	0.028$^{\bullet}$	0.413
Total Completeness	**0.54**	0.11	0.50	0.11	0.061	0.364
Total Validity	**0.48**	0.11	0.43	0.10	0.017$^{\bullet}$	0.476
Adjusted Relation Completeness	**0.66**	0.19	0.55	0.20	0.007$^{\bullet\bullet}$	0.563*
Adjusted Relation Validity	**0.52**	0.19	0.43	0.16	0.007$^{\bullet\bullet}$	0.514*
Adjusted Total Completeness	**0.68**	0.09	0.63	0.10	0.004$^{\bullet\bullet}$	0.525*
Adjusted Total Validity	**0.58**	0.11	0.53	0.08	0.002$^{\bullet\bullet}$	0.523*

(b) Planning Poker

	User Stories		Use Cases		Sig.	Effect
	$\bar{x}$	σ	$\bar{x}$	σ	p	size r
Entity Completeness	**0.80**	0.17	0.78	0.17	0.520	0.118
Entity Validity	**0.75**	0.18	0.72	0.17	0.380	0.171
Relation Completeness	**0.45**	0.22	0.42	0.28	0.623	0.120
Relation Validity	**0.37**	0.21	0.35	0.23	0.618	0.091
Total Completeness	**0.62**	0.17	0.60	0.19	0.532	0.111
Total Validity	**0.54**	0.17	0.52	0.17	0.496	0.118
Adjusted Relation Completeness	**0.63**	0.25	0.58	0.24	0.322	0.204
Adjusted Relation Validity	**0.48**	0.23	0.44	0.22	0.409	0.178
Adjusted Total Completeness	**0.63**	0.20	0.60	0.20	0.440	0.150
Adjusted Total Validity	**0.53**	0.17	0.51	0.16	0.489	0.121

Table 2
Preferences by notation. Legend: $^{\bullet}$ for $p < .05$; $^{\bullet\bullet}$ for $p < .01$; * for $g \geq 0.5$.

Statement	User Stories		Use Cases		Sig.	Effect
	$\bar{x}$	σ	$\bar{x}$	σ	p	size r
Fit for developing a conceptual model	**3.78**	0.91	3.35	1.00	0.002$^{\bullet\bullet}$	0.613*
Help in identifying classes	**3.79**	0.92	3.46	0.87	0.010$^{\bullet}$	0.500*
Help in identifying relationships	**3.67**	0.98	3.53	0.96	0.336	0.183
Help comprehend the system structure	**3.59**	0.89	3.57	0.99	0.988	0.003
Provide a system overview	**3.49**	1.09	3.33	1.12	0.228	0.229
Clearly presents a single requirement	**3.81**	0.99	3.37	0.97	0.002$^{\bullet\bullet}$	0.613*
Which method do you prefer?	68		39			

Gathering the preferences of both groups together, Table 2 indicates that the subjects favor user stories. The differences are of statistical significance (applying Wilcoxon test [58]) in the case of developing a conceptual model, identifying classes, and clearly presenting a single requirement. The validity of these findings is confirmed by their *medium* effect size, equal to or above 0.5. For the other tasks, there was a consensus regarding the benefits of using user stories to describe the requirement as well. Furthermore, in both groups, most subjects prefer to use user stories to use cases.

3.3. Discussion

The conclusion we can draw from the first experiment (see [59] for more details) is that the notation seems to affect the conceptual model derivation. This conclusion, however, needs to be interpreted according to other factors, as we describe in the following.

The *complexity of the case studies* seems to affect the results. The Planning Poker case study was less complex than the Data Hub case study (14 versus 23 concepts). Even though Planning Poker was presented as the second case study in the experiment forms—one would expect the participants to be less effective because of being tired—, the conceptual models fit better the gold standard solution. For Data Hub, the complexity emerges due to various factors: the number of entities, the number of relationships, the introduction of an external system (the billing system) with which the system under design interacts, the multiple interactions among the roles/actors, and the existence of several related roles/actors with similar names. The results may also be affected by the *course context*: the focus was the design of a system; thus, interactions among actors were of less importance, as well as those with external systems.

When referring to the qualitative inspection of the results, it seems that the concepts that identify actors occur with *multiple repetitions* in the user stories and, thus, the subjects were able to better identify the actors as well as the relationships between them. In use cases, actors are usually mentioned only at the beginning of each use case (in the “actor” section), and the described actions are implicitly referring to the interaction between the actor and the system. In user stories, instead, actors are expressed in every user story in the “As a” part. Similar to actors, it seems that in the user stories, there are entities that recur multiple times, as they are used in many operations. This also led to better identification of such entities when deriving the conceptual models.

Another explanation for the better performance with user stories may be that these are focused on the specification of individual features that an actor needs, whereas use cases blur the identification of entities within a transaction flow.

The subjects also perceived user stories as a better suitable notation for the tasks we asked them to perform. This is remarkable since the course in which this experiment was embedded focuses on the use of the UML for system design. The subjects also acknowledge the benefits for other tasks, yet the difference was not significant.

3.4. Threats to validity

Our results need to be considered in view of several threats to validity [60]. *Construct validity* The selection of the domains/cases may affect the results; we aimed to provide domains that would be easy to understand by the subjects. Moreover, the specification using the two notations were not fully aligned as they emphasize different aspects (a process in the software for PP vs. individual features for DH). However, this is exactly one of the triggers of our research. To mitigate the risk of favoring one notation over the other, two authors independently created a conceptual model from either notation before the experiment, and then they reconciled their outputs.

Internal validity We mitigate external factors that might affect the dependent variables (such as familiarity with the domain, the degree of commitment by the subjects, and the training level of the subjects). The subjects were not familiar with either domain, and thus, they probably were not affected. The random assignment that was adopted should eliminate various kinds of external factors. Although the experiment was done on a voluntary basis, the subjects were told that they would earn bonus points based on their performance, and thus we increased the motivation and commitment of the subjects. In addition, it might be that acquiring reasoning abilities in extracting entities and their relationships affect the results. These indicate that the second task resulted in a better conceptual model. Yet, we attribute this difference to the lower complexity of the second domain, although this has to be tested with other studies. Moreover, although we tried to ensure similar complexity of the cases by having specifications of the same size, the conceptual model for DH was larger than that for DH (see our previous paper for details [55]). Another threat concerns the order of the domains within the experiment, which may also affect the results. A final threat is fatigue, yet the experiment was relatively short. In fact, all subject delivered their outcomes way before the deadline, so the time limit was not an issue as well.

Conclusion validity We followed the various assumptions of the statistical tests (such as the normal distribution of the data and data independence) when analyzing the results. In addition, we used a predefined solution, which was established before the experiment, for grading the subjects' answers; thus, only limited human judgment was required.

External validity The main threat stems from the choice of subjects and from the use of simple experimental tasks. The subjects were undergraduate students with limited experience in software engineering, in general, and in modeling in particular. Kitchenham et al. argue that using students as subjects instead of software engineers is not a major issue as long as the research questions are not specifically focused on experts [61], as is the case in our study. In addition, it might be that the template selected for the two notations also affected the results. Yet, these are the most common ones in their categories. Generalizing the results should be taken with care as the case studies are small and might be different in the way user stories and use cases are written in industry settings.

4. Quasi-experiment: from elicitation to conceptual models

The results from the first, controlled experiment led us to conduct a second experiment in which each subject conducted an end-to-end process: requirements elicitation via an interview, specification via UC or US, and derivation of a conceptual model. This contrasts with the first experiment in which the subjects started from a specification provided by the researchers. Our major goal was to establish whether the difference that we observed was inherent to the notation itself and whether such a difference would be visible also in a setting in which the subject plays the role of an analyst starting from the elicitation phase. We were also interested in identifying other factors, which were controlled in the first experiment, that may affect the quality of the derived conceptual model.

In the following, we describe the second experiment in an analogous manner as we did for the first experiment.

4.1. Hypotheses

In this experiment, we still compare the differences in the quality of a manually derived conceptual model from UC and US. However, this time we simulate a process in which the creators of the conceptual models are those who write the requirements specification. Furthermore, we allow for more flexibility in the time allocated for the derivation process. Here again, quality is measured by means of the *validity* and *completeness* of the derived conceptual models with respect to expert solutions.

Our conjectures are similar to the previous experiment, and we believe that there is a trade-off in using the two notations. Thus, we test H_0 again.

The planning of the second experiment, which spreads over multiple weeks, is visualized in Fig. 1. The subjects were trained on the preparation and conduction of a requirements interview, on the specification of requirements as user stories and lastly, on the various requirements modeling techniques, and in particular on class diagrams. The idea of deriving conceptual models from requirements was mentioned both in the lecture on user stories (as proposed in previous work [17]) and in the requirements modeling lecture, which provides heuristics to construct class diagrams from requirements documents [62]. Since the subjects were graduate students, most of whom have background experience in basic software engineering and UML, we did not give a lecture on use cases but referred the students to the relevant chapter from Larman's textbook [28]. The instructor of the course was the first author of this paper. Through this experimental process, the subjects were simulating the entire process that affects the conceptual model derivation, unlike the first experiment, in which they started from a specification that was prepared by us.

The dependencies between activities concern their flow for the involved students (e.g., derivation cannot be done until a specification is created) and for the lectures (or other education modalities) that should be delivered prior to the various activities: interviews, specification, and derivation of models.

4.2. Experiment design

In the following, we describe the variables and their measurements, the subjects, and the tasks. *Independent Variables* The first variable (IV1) is the notation: User Stories (US) and Use Cases (UC). The second independent variable (IV2) is the case study, which has three possible values: Urban Traffic Simulator (SIM), International Football Association portal (IFA), and Hospital Management System (HOS). Each of these cases is a fictitious setting that was defined by each of the three authors of this paper. SIM refers to a municipality that struggles with traffic and pollution issues and would like to acquire and configure an urban traffic simulator to analyze possible solutions that would improve the situation. IFA refers to the creation of a portal that would support teams, referees, and league managers in the setup and management of football leagues. HOS is the case of a hospital that requires a unified information system that would combine and improve the many specific systems that are currently in operation.

Dependent Variables We could not define a single unified *gold standard* per case, as we did in the first experiment because each subject created her own requirements specification using one notation. Thus, we created an *expert model* from each of the specifications. We compare the *student model* against the expert model using two dependent variables:

- *Validity (DV1)*: the ratio of classes in a student model that also appear in the expert model, over the number of classes that are correctly or incorrectly represented in the student model.

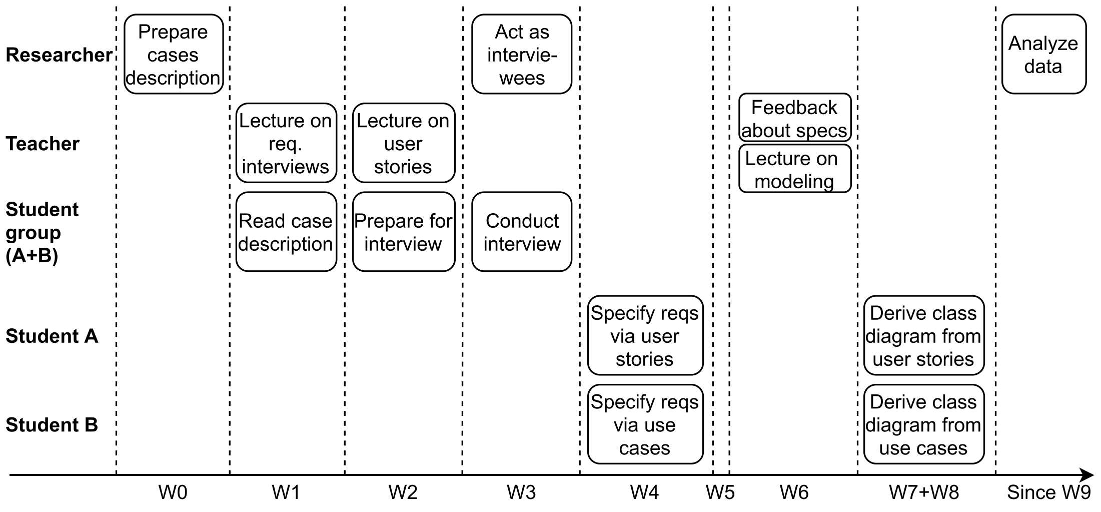


Fig. 1. Timeline of the second experiment.

- *Completeness (DV2)*: the ratio of classes in a student model that are correctly or incorrectly represented, over the number of classes in the expert model.

Note that, since we did not prescribe a specific derivation process in this experiment, we only focus on classes, and we do not analyze the relationships. Given the number of student and expert models, we use the measurements of Table 3 and calculate DV1 and DV2 as follows:

- *Validity* $= \frac{|AL|}{|AL|+|SO|+|WR|}$
- *Completeness* $= \frac{|AL|+|WR|}{|AL|+|WR|+|OM|}$

Note that we do not consider missing concepts in our metrics because those concepts are not necessarily errors but could simply represent secondary domain elements that the student decided to include in the model.

Moderating Variables We consider two variables that moderate the effects of the independent variables on the dependent variables. First, the derivation process adopted by the student, as explained in her report (MV1): no process adopted, partial process was adopted, or systematic process was adopted. When the student reports on the procedure to determine entities (through roles and objects identification, for example), their attributes, and the relationships (through verbs, for example), we consider this as a systematic process. Such a systematic process includes sufficient information for another subject to reproduce the extraction process. When the student reports on the justification of the model with no details regarding the procedure, we consider it as no process. Cases in between these two extremes are considered as partial derivation process. The judgment was made by the researcher, who acted as a domain expert for that case. Second, to test whether the experience and background of the student have an impact, we consider the exam grade (MV2). The grade was converted to categorical values (High if at least 80%, Low if below 65%, Medium otherwise) by the first author, based on his experience with the grading system in the department and for the specific course. Note that the exam grade is independent of the assignments concerning the interviews and the derivation of the conceptual models.

Subjects We involved master's students taking the course on Requirements Engineering at Utrecht University. The course teaches how to elicit, specify, analyze, model, and manage requirements for software systems. 32 students participated in the assignment. Out of these, 24 granted us consent to use their data (requirements, models, and reports) for this research. Most of these students had prior experience with modeling languages.

Task and Execution The experiment was spread over eight weeks (W1–W8 in Fig. 1), with the periods before and after that focused on the case preparation and on the analysis of the results. In W1, a lecture was given regarding requirements interviews, using adapted materials from the SaPeer method for teaching requirements interviews [63,64]. Moreover, groups of two students were assigned to a 1-page case description. In W2, the students attended a lecture on user stories and started the scheduling of and preparation for the interview; they were suggested to collect additional materials and to prepare for how to conduct the interview. In W3, all the interviews took place, with the authors of this paper playing the role of the domain experts, one per case. The interviews regarding SIM and HOS took place face to face, while those regarding IFA took place via Skype since the corresponding domain expert does not reside in the same country as the students. In W4, the students delivered two separate reports that included a reflection on the interview and their specifications, either via user stories or via use cases. They were briefed to work independently, without collaboration. The students in a group agreed on which notation each of them would use. In W6, the students attended a lecture on requirements modeling. In that week, the students received feedback regarding their requirements specification from the course instructors (the first two authors). From that moment on, till the end of W8, they worked independently on the creation of a second individual assignment, which included the derivation of a structural/static conceptual model, a class diagram.

In the next weeks, we analyzed the collected data. We created the domain expert versions of the conceptual models, starting from the

Table 3
Measurements used to calculate DV1 and DV2.

Situation	Description
Aligned (AL)	A concept is represented as a class in both models, either with the same name or using synonyms or clearly linkable names.
Wrongly represented (WR)	A class in the domain expert model is incorrectly represented in the student model, either (i) via an attribute, method, or relationship rather than class, or (ii) using a generic term (e.g., "user" instead of "urban planner").
System-oriented (SO)	A class in CM-Stud that denotes a technical implementation aspect, e.g., access control. Classes that represent a legacy system or the system under design (portal, simulator) are legitimate.
Omitted (OM)	A class in CM-Expert that does not appear in any way in CM-Stud.
Missing (MI)	A class in CM-Stud that does not appear in any way in CM-Expert.

specifications and without accessing the students' class diagrams. This activity required some iterations to ensure uniformity across the three researchers. Then, after an informal analysis of the student models, we derived a tagging guide that we used to label the student models and the expert models, using the measurements listed in Table 3, and to populate a spreadsheet. That spreadsheet was then used to execute most of the statistical analyses, except for the tests for normality that were executed using SPSS, and the effect size was determined with an online calculator available at https://www.psychometrica.de/effect_size.html. The student specifications and models, the expert models, the spreadsheet, the tagging guidelines, and the system descriptions are all available in our online appendix [59].

4.3. Results

In Table 4, we present descriptive statistics regarding the subjects' models. Each row represents the outcomes of a single subject, indicating the notation used (IV1), the case applied (IV2), the applied derivation process (MV1), the exam grade (MV2), the size ratio (i.e., the number of classes within the subject model divided by the number of classes within the expert model), the model validity (DV1), and the model completeness (DV2). An interesting initial observation is that all the models created by the students were smaller in size to those created by the domain experts, as the size ratio is always below 1.

Fig. 2 presents the size ratio aggregated by the main independent and moderating variables. Although this metric only indicates relative size, without focusing on the alignment between student and expert models, we can observe some differences. In particular, we see how the simulation case (IV2) seems the most challenging to fully represent: many students had a considerably lower number of classes than in the corresponding expert models. In addition, the variance is larger than those on the other cases, indicating inconsistencies between the subjects. Finally, we see that the presence of a well-described derivation process (MV1) seems to lead to a model whose size is closer to the corresponding expert model. The differences in the notation (IV1) and the grade (MV2) are marginal.

We further analyze the effect of each of the independent and mediated variables on the validity (DV1) and completeness (DV2) of the subjects' models. The results are summarized through diverging stacked bar charts in Figs. 3–6. Furthermore, to examine the differences caused by the different variables, we performed a T-test with Welch's correction (the data is normally distributed but equal variance assumption does not hold) to check statistical significance, and we calculated the effect size using Hedges' g. Table 5 presents the results of this analysis. We discuss the results per each variable with the help of the p and g values.

IV1: Notation Fig. 3 indicates that deriving a conceptual model from use cases outperforms such derivation from user stories with respect to both validity (3.3%) and completeness (9.5%). These differences are limited and not statistically significant, as the UC-US row of Table 5 shows. Nevertheless, the difference in completeness shows a *medium effect*, for Hedges' g is between 0.5 and 0.8.

IV2: Case

Fig. 4 indicates that the case to which the subjects were assigned affects the validity and completeness of the derived conceptual model. In particular, the simulation case results in the lowest validity and completeness, with results that are circa 17% worse than the hospital and the football management cases for validity, and circa 15% worse for completeness. The statistical analysis confirms the results. Although the differences do not achieve statistical significance, probably due to the low number of samples per case, the effect size is large for the SIM-HOS and the SIM-IFA cases, both for validity and completeness.

MV1: Process

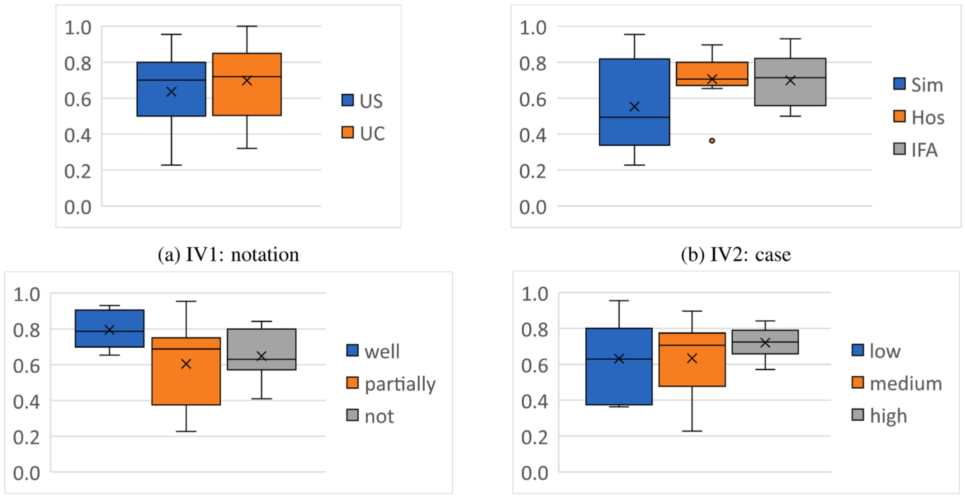
Fig. 5 indicates that the derivation process affects the completeness of the derived conceptual model. In particular, when the students reported on a systematic process, completeness is increased by approximately 13–30%. On the other hand, validity is not affected by this factor in our experiment. The statistical analysis in Table 5 reveals that statistical significance and large effect sizes exist for completeness when comparing partial processes and systematic ones (◐ - ●). A large effect size also exists for the completeness of no-process versus systematic process (○ - ●), but with no statistical significance. It is certainly remarkable that, despite the low sample size, we achieved significance and large effect size for the ◐ - ● case.

MV2: Grade

Fig. 6 indicates that the grades of the subjects have a limited effect on the validity and completeness of the derived conceptual model. Subjects with high grades at the final exam were able to derive more valid conceptual models by 4%-12%; for completeness, instead, the students with medium grade outperformed the rest, although to a limited extent. The statistical analysis does not show significance, although the difference in

Table 4
Results for the second experiment: every row indicates one specific student. The entire results can be found in our online appendix [59].

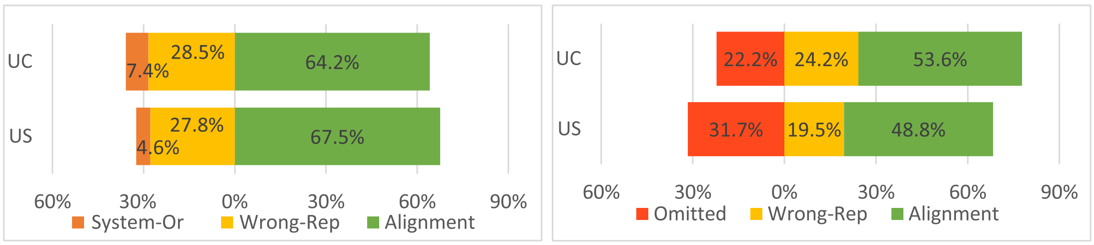
Group	IV1: Not.	IV2: Case	MV1: Proc	MV2: Exam	AL	WR	SO	OM	MI	Size ratio	DV1: Val.	DV2: Comp.
g01	US	HOS	●	M	13	9	4	3	0	0.65	0.50	0.88
g02	US	HOS	◐	L	8	8	0	6	0	0.36	0.50	0.73
g04	US	SIM	◐	L	11	4	0	6	10	0.95	0.73	0.71
g05	US	IFA	◐	H	9	1	4	10	1	0.75	0.64	0.50
g06	US	HOS	◐	H	12	3	0	7	0	0.70	0.80	0.68
g08	US	IFA	◐	M	11	2	1	8	0	0.55	0.79	0.62
g09	US	SIM	◐	M	3	9	0	8	1	0.23	0.25	0.60
g10	US	HOS	●	M	13	1	0	5	3	0.80	0.93	0.74
g11	US	IFA	◐	L	8	4	0	10	0	0.50	0.67	0.55
g12	US	HOS	○	M	8	3	0	6	4	0.71	0.73	0.65
g14	US	IFA	○	L	17	2	0	3	2	0.80	0.89	0.86
g02	UC	HOS	◐	M	9	6	3	1	0	0.75	0.50	0.94
g03	UC	IFA	○	H	13	2	0	3	0	0.80	0.87	0.83
g04	UC	SIM	○		6	1	2	11	3	0.58	0.67	0.39
g05	UC	IFA	○	H	8	8	1	1	0	0.57	0.47	0.94
g06	UC	HOS	◐	H	9	1	1	5	2	0.69	0.82	0.67
g07	UC	SIM	◐	L	5	6	1	12	2	0.38	0.42	0.48
g08	UC	IFA	○	L	11	7	2	5	0	0.63	0.55	0.78
g09	UC	SIM	●	H	14	4	1	4	2	0.77	0.74	0.82
g10	UC	HOS	◐	L	12	0	0	3	0	0.80	1.00	0.80
g11	UC	IFA	●	M	14	4	0	3	0	0.71	0.78	0.86
g13	UC	HOS	●	M	18	9	6	2	1	0.90	0.55	0.93
g14	UC	IFA	●		16	11	0	1	2	0.93	0.59	0.96
g15	UC	SIM	○	M	6	9	0	6	2	0.41	0.40	0.71



(a) IV1: notation (b) IV2: case

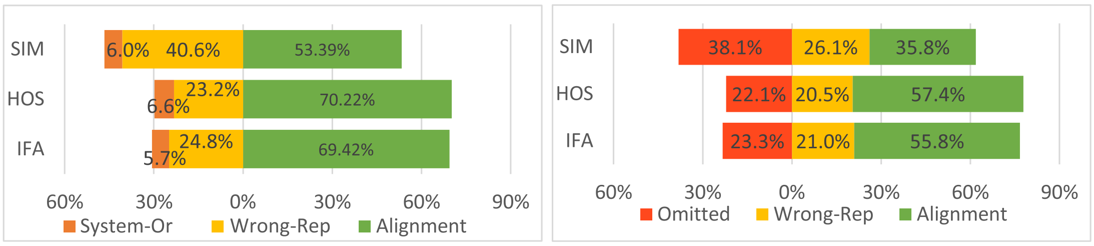
(c) MV1: process (d) MV2: grade

Fig. 2. Ratio between classes in the student model and in the expert model.



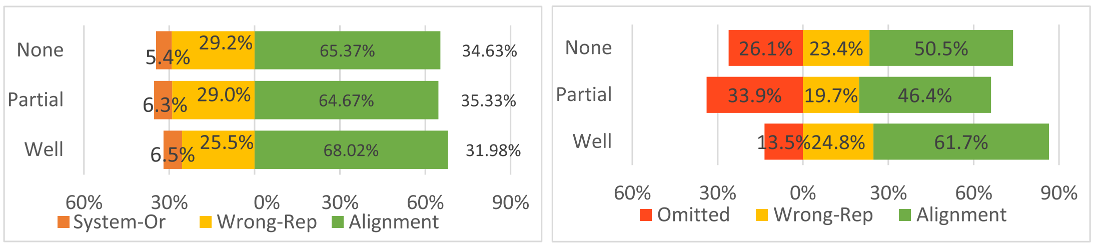
(a) DV1: Validity (b) DV1: Completeness

Fig. 3. Diverging stacked bar charts for validity and completeness by notation, respectively.



(a) DV1: Validity (b) DV1: Completeness

Fig. 4. Diverging stacked bar charts for validity and completeness by case, respectively.

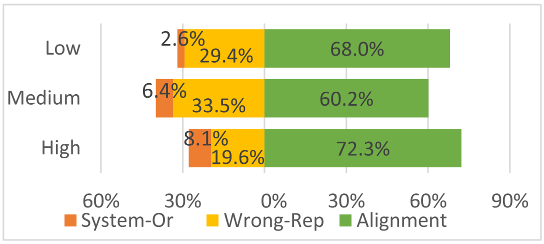


(a) DV1: Validity (b) DV1: Completeness

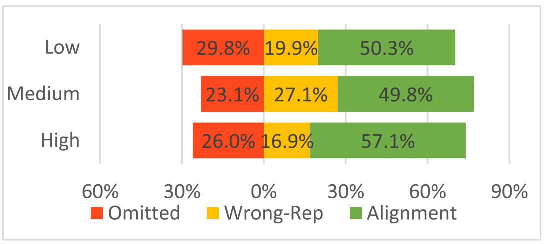
Fig. 5. Diverging stacked bar charts for validity and completeness by process, respectively.

validity for medium-high and in completeness for medium-low show a medium effect.

Following the results, we cannot reject the initial hypothesis H_0, thus we can conclude that the notation used does not affect the completeness and validity of the derived conceptual model. However, the analysis reveals that other factors seem to have a higher impact.



(a) DV1: Validity



(b) DV1: Completeness

Fig. 6. Diverging stacked bar charts for validity and completeness by exam grade, respectively.

Table 5
T-Tests to analyze the significance of validity and completeness when splitting the data by each independent and moderating variable. The columns indicate the groups, the size of each group n_1 and n_2, mean $\bar{x}$ and standard deviation σ for each group, significance in terms of p-value and Hedges' g effect. Notation: $^{••}$ if $p < .01$, $^{•}$ if $p < .05$, ** if $g \geq 0.8$ (large effect), and * if $g \geq 0.5$ (medium effect).

			Validity						Completeness					
Groups	n_1	n_2	Group 1		Group 2		Sig.	Eff. size	Group 1		Group 2		Sig.	Eff. size
			$\bar{x}$	σ	$\bar{x}$	σ	p	g	$\bar{x}$	σ	$\bar{x}$	σ	p	g
IV1: Notation														
UC-US	13	11	64.16	17.91	67.54	18.82	0.673	0.184	77.80	17.08	68.32	11.36	0.135	0.644*
IV2: Case														
SIM-HOS	6	9	53.39	18.75	70.22	18.58	0.144	0.903**	61.90	14.77	77.87	10.69	0.068	1.286**
SIM-IFA	6	9	53.39	18.75	69.42	13.74	0.135	1.011**	61.90	14.77	76.74	16.11	0.116	0.951**
HOS-IFA	9	9	70.22	18.58	69.42	13.74	0.923	– 0.049	77.87	10.69	76.74	16.11	0.871	– 0.083
MV1: Derivation process														
○ - ◐	7	11	65.37	17.63	64.67	20.41	0.943	– 0.036	73.87	16.85	66.09	12.85	0.352	– 0.537*
○ - ●	7	6	65.37	17.63	68.02	14.87	0.792	0.161	73.87	16.85	86.46	7.43	0.136	0.939**
◐ - ●	11	6	64.67	20.41	68.02	14.87	0.723	0.550*	66.09	12.85	86.46	7.43	0.002$^{••}$	1.797**
MV2: Exam grade														
L - M	7	9	68.02	19.47	60.16	20.43	0.479	– 0.391	70.16	12.96	76.92	12.68	0.348	0.528*
L - H	7	6	68.02	19.47	72.25	13.29	0.680	0.248	70.16	12.96	74.02	14.24	0.651	0.285
M - H	9	6	60.16	20.43	72.25	13.29	0.219	0.671*	76.92	12.68	74.02	14.24	0.717	– 0.218

4.4. Discussion

The results of the second experiment indicate that the notation used for specifying the requirements (IV1), the major objective of our study when we initiated it affects the derived conceptual model only to a limited extent. Thus, when introducing equivalent requirements in either notation, we expect that an analyst would be able to derive conceptual models of comparable quality.

On the other hand, the complexity of the case (IV2) seems to have a stronger effect on the derivation of a conceptual model, as visible by the results in Fig. 4. In our study, in particular, the urban traffic simulator SIM case led to significantly lower validity (DV1) and completeness (DV2) than the other two cases that are more classic types of information systems.

Regarding the moderating variables, the proficiency of the subjects (MV2, operationalized by their grade in the final exam) did not show any visible effect. The derivation process (MV1), instead, has a high impact on completeness. When the subjects described (and, presumably, followed) a systematic process, they were able to better identify the concepts that are present in the corresponding expert model. This is not surprising, as the systematic process allows for a detailed exploration of the domain or system to be that prevents overlooking certain concepts, and that guides the analyst in focusing on the important entities and relationships.

We could not obtain statistically significant results for most of all comparisons regarding the second experiment. Nevertheless, the effect size was more often medium or large, thereby indicating that a difference existed. We attribute this to the small number of subjects: when splitting the 24 subjects, we end up with small groups that can hardly lead to p-values below the common significance thresholds of 0.05 or 0.01. Additional studies are clearly necessary: the identified differences, although visible, need to be confirmed with more samples.

4.5. Threats to validity

The results of the second experiment also need to be considered in view of several threats to validity [60].

Construct validity We examined the use of two RE notations and the derivation process for the purpose of developing a conceptual model based on the requirements determined by the subjects. The fact that the requirements are different among the subjects even within the same domain may affect the results; our choice is justified by our attempt to simulate a real environment in which conceptual modelers participate in the development process as it starts. We compare the derived models to the ones created by experts similarly as in the controlled experiment.

Internal validity

As the experiment was conducted in a rolling manner, several difficulties exist. We are not aware of the actual time and efforts allocated for the derivation task, and we can not guarantee the same conditions to perform the task as these were accumulated during the experiment. As the entire experiment took place through multiple weeks, many noisy elements may interfere. Nevertheless, as the experiment was part of the subject grade in a course, we believe that the subjects paid full attention to its execution. We explicitly asked the subjects to derive the conceptual models based on the set of requirements they had, though we can not be sure that the knowledge and background the subjects gained throughout the experiment were not used to derive the conceptual models. We also asked the subjects to elaborate on the derivation process, but we can not confirm that this reflects the actual process. Furthermore, we could not measure nor control the relative ability of the students in deriving conceptual models; nevertheless, the analysis when splitting per grade (MV2) does not show significant differences depending on the performance in the course. We have no indication of the tools used to develop the models, yet, this is of minimal importance since we were interested mainly in the classes and the relationships and ignored syntax errors. Note that each model was checked against the requirements specified by the subjects themselves. Thus, the quality of these specifications may also affect the results. Finally, while we kept the case descriptions of similar size (1 page each), the SIM case proved to be more complex. While this decreases the generality of our findings, the results indicate the important role of case complexity in the derivation process.

Conclusion validity We followed the various assumptions of the statistical tests (such as normal distribution of the data and data independence) when analyzing the results. In addition, we used a predefined solution, which was established before grading the subjects' answers. The predefined solutions follow a systematic grammatical analysis, and comparing them was done following a protocol we agreed upon (as discussed above) to ensure consistency among the experts. The limited sample size ($n = 24$) makes some of the findings preliminary, as more subjects are necessary to confirm the results. To mitigate the threat, we studied significance in conjunction with effect size, and the findings in the forthcoming Section 5 are stated as hypotheses for future work.

External validity Since the subjects were students with limited experience in modeling, this might affect the results, yet it works for both notations and procedures. In addition, it might be that the template selected for the two notations also affected the results. Yet, these are the most common ones in their categories. Generalizing the results should be taken with care, and additional studies are needed.

5. Implications

The two experiments complement each other: while the controlled experiment allowed us to identify that the notation may impact the derived conceptual model, the second experiment further examined additional factors that affect the derivation process, besides the notation.

The results of the two experiments show several similarities. First, the complexity of the case/domain that is under test affects the quality of the derived conceptual model in both experiments. We designed the experiments with specifications/descriptions of similar complexity in terms of their size. Yet, we noticed that when the domains or systems are difficult to understand either due to size, familiarity, or inherent complexity, or their models are difficult to develop, the quality of the derived models has shown to be lower. In the first experiment, the Planning Poker case was simpler and clear. Thus, the subjects achieved better results, both when using user stories or use cases. In the second experiment, deriving the conceptual model related to the urban simulation domain, which was difficult to grasp with respect to the two other domains, gained limited validity and completeness.

Implication 1: Case Complexity The complexity of the case/domain seems to significantly affect the quality of the derived conceptual model. This leads to a hypothesis for future research: *H1: The domain/system complexity affects the ability of a human to derive a valid and complete conceptual model.*

As for the notations, in the first experiment, specifying requirements using US resulted in better conceptual models and this finding was statistically significant. However, the effect size is small to medium, indicating that the difference in using the two notations should be considered with caution. In the second experiment, specifying requirements using UC resulted in better conceptual models, although with low magnitude. Based on our studies, it seems that the tested notations have a limited impact on the quality of the derived conceptual models.

Implication 2: Notation The choice of a notation between user stories and use cases does not seem to affect the task of deriving static conceptual models. However, we still need to study whether the notation affects other tasks, such as the identification of ambiguity or the derivation of dynamic conceptual models.

Some moderating variables seem to play a role. The length of the specification was different in the two experiments. In the first experiment, the specifications were concise, fitting a single page. In the second experiment, the specifications, created by the subjects themselves, spanned across multiple pages. Furthermore, the first experiment introduces time constraints. Thus, in such a setting, the required cognitive effort might have affected the results. The user stories in the first experiment that adequately follows the template were easy to analyze in comparison to the use cases. In the second experiment, this effect was eliminated as there was sufficient time for the participants to perform the task, some of the user stories were longer, and the number of requirements was high.

Implication 3: Other Factors The research reveals that multiple factors may affect the quality of the derived conceptual model. This leads to further hypotheses:
H2: Time constraints affect the quality of the derived conceptual model.
H3: The specification size affects the quality of the derived conceptual model.
H4: The specification style affects the quality of the derived conceptual model.

The settings and the execution of the two experiments lead us to conclude that the most important factor seems to be the derivation process. In the first experiment, as we adopted a systematic process, the independent derivation processes led (the researchers) to very similar conceptual models (even though they originated from different notations). The derivation process that the students learned is actually a grammatical analysis of the requirements [65]. In the second experiment, the subjects who used a well-defined derivation process also used a grammatical analysis approach and achieved conceptual models of better quality in terms of completeness. Yet, we could not identify differences in terms of validity.

Implication 4: Systematic Derivation Following a systematic derivation process seems to increase model completeness. Together with implication 1, this calls for tools that can assist the derivation process [17,36], leading to the following hypothesis:
H5: Automated tooling that implements systematic derivation guidelines helps analysts to derive more complete conceptual models.

6. Summary

We conducted two experiments to investigate the factors that potentially affect the derivation of static conceptual models from specifications written as user stories and use cases. In the first, controlled experiment, 118 subjects received the same requirements expressed either in use cases or user stories. In the second experiment, the 24 subjects were required to perform the entire process of requirement elicitation, specification, and derivation of conceptual models from their own requirements.

The analysis of the results shows that requirement notations seem to have a limited impact on the quality of the derived conceptual model, both with respect to validity and completeness. The most influential factor is the adopted derivation process. In both experiments, the use of a systematic grammatical analysis results in conceptual models of relatively high completeness and validity. Furthermore, our results also show that more complex domains or systems lead to derived models of lower quality.

The limited effect of the requirements notation is in line with the findings of de Oliveira Neto et al. [66]. In that work, the effect of the notation was examined for the task of test case generation. There, it was found that goal models, textual requirements, and user stories achieve test cases with similar quality (though it was pointed out that each notation better supports certain aspects).

The results obtained in this research call for further experimentation with tasks that are based on requirements notations and techniques. The research community needs to build a corpus of evidence to assist practitioners in the choice of notations and techniques for the RE tasks at hand. We plan to continue this research with larger case studies and with the use of qualitative methods that may help reveal the adequacy of RE notations for certain tasks.

CRediT authorship contribution statement

Fabiano Dalpiaz: Conceptualization, Methodology, Formal analysis, Resources, Writing - review & editing, Visualization. **Patrizia Gieske:** Formal analysis, Writing - review & editing, Visualization. **Arnon Sturm:** Conceptualization, Methodology, Formal analysis, Writing - review & editing, Visualization.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

The two experiments obtained research ethics approval from the relevant committees at Ben-Gurion University of the Negev and at Utrecht University, respectively. At Ben-Gurion University the research was approved by the Software and Information Systems Department Ethics Committee and the project approval identifier is SISE-2019-05. At Utrecht University, the research was approved by the Science-Geosciences Ethics Review Board and the research project identifier is Bèta S-20339. We would like to thank all the students who voluntarily decided to share their produced empirical data for research purposes. Also, we are grateful to Sergio Espaa and Sjaak Brinkkemper for their terminological assistance.

References

[1] F. Dalpiaz, A. Ferrari, X. Franch, C. Palomares, Natural language processing for requirements engineering: the best is yet to come, IEEE Softw. 35 (5) (2018) 115–119.
[2] J. Mylopoulos, L. Chung, E. Yu, From object-oriented to goal-oriented requirements analysis, Commun. ACM 42 (1) (1999) 31–37.
[3] A. van Lamsweerde, Requirements Engineering: From System Goals to UML Models to Software Specifications, Wiley, 2009.
[4] P. Ciancarini, S. Cimato, C. Mascolo, Engineering formal requirements: an analysis and testing method for z documents, Ann. Softw. Eng. 3 (1) (1997) 189–219.
[5] A. Fuxman, L. Liu, J. Mylopoulos, M. Pistore, M. Roveri, P. Traverso, Specifying and analyzing early requirements in Tropos, Requir. Eng. 9 (2) (2004) 132–150.
[6] S. Wagner, D.M. Fernández, M. Felderer, A. Vetrò, M. Kalinowski, R. Wieringa, D. Pfahl, T. Conte, M.-T. Christiansson, D. Greer, C. Lassenius, T. Männistö, M. Nayebi, M. Oivo, B. Penzenstadler, R. Prikladnicki, G. Ruhe, A. Schekelmann, S. Sen, R. Spínola, A. Tuzcu, J.L.D.L. Vara, D. Winkler, Status quo in requirements engineering: a theory and a global family of surveys, ACM Trans. Softw. Eng. Method. 28 (2) (2019).
[7] A. Mavin, P. Wilkinson, A. Harwood, M. Novak, EARS (Easy approach to requirements syntax). Proceedings of the IEEE International Requirements Engineering Conference, 2009, pp. 317–322.
[8] M. Cohn, User Stories Applied: for Agile Software Development, Addison Wesley, 2004.
[9] G. Lucassen, F. Dalpiaz, J. van der Werf, S. Brinkkemper, The use and effectiveness of user stories in practice. Proceedings of the International Working Conference on Requirements Engineering: Foundation for Software Quality, 2016, pp. 205–222.
[10] V. Gervasi, D. Zowghi, Reasoning about inconsistencies in natural language requirements, ACM Trans. Softw. Eng. Method. 14 (3) (2005) 277–330.
[11] T. Yue, L.C. Briand, Y. Labiche, A systematic review of transformation approaches between user requirements and analysis models, Requir. Eng. 16 (2) (2011) 75–99.
[12] E. Insfrán, O. Pastor, R. Wieringa, Requirements engineering-based conceptual modelling, Requir. Eng. 7 (2) (2002) 61–72.
[13] N.A.M. Maiden, S.V. Jones, S. Manning, J. Greenwood, L. Renou, Model-driven requirements engineering: synchronising models in an air traffic management case study. Proceedings of the International Conference on Advanced Information Systems Engineering, 2004, pp. 368–383.
[14] O.I. Lindland, G. Sindre, A. Solvberg, Understanding quality in conceptual modeling, IEEE Softw. 11 (2) (1994) 42–49.
[15] J. Mylopoulos, Conceptual Modelling and Telos, in: P. Loucopoulos, R. Zicari (Eds.), Conceptual Modeling, Databases, and Case: An Integrated View of Information Systems Development, John Wiley & Sons, New York, 1992, pp. 49–68.
[16] C. Arora, M. Sabetzadeh, S. Nejati, L. Briand, An active learning approach for improving the accuracy of automated domain model extraction, ACM Trans. Softw. Eng. Method. 28 (1) (2019).
[17] G. Lucassen, M. Robeer, F. Dalpiaz, J.M.E. van der Werf, S. Brinkkemper, Extracting conceptual models from user stories with visual narrator, Requir. Eng. 22 (3) (2017) 339–358.
[18] F. Dalpiaz, I. van der Schalk, S. Brinkkemper, F.B. Aydemir, G. Lucassen, Detecting terminological ambiguity in user stories: tool and experimentation, Inf. Softw. Technol. (2019).
[19] G. Loniewski, E. Insfran, S. Abrahão, A systematic review of the use of requirements engineering techniques in model-driven development. Proceedings of the International Conference on Model Driven Engineering Languages and Systems, 2010, pp. 213–227.
[20] P.X. Mai, A. Goknil, L.K. Shar, F. Pastore, L.C. Briand, S. Shaame, Modeling security and privacy requirements: a use case-driven approach, Inf. Softw. Technol. 100 (2018) 165–182.
[21] H. Harmain, R. Gaizauskas, CM-Builder: a natural language-based CASE tool for object-oriented analysis, Autom. Softw. Eng. 10 (2) (2003) 157–181.
[22] T. Yue, L.C. Briand, Y. Labiche, aToucan: an automated framework to derive UML analysis models from use case models, ACM Trans. Softw. Eng. Method. 24 (3) (2015).
[23] M. Glinz, Problems and deficiencies of UML as a requirements specification language. Proceedings of the International Workshop on Software Specifications & Design, 2000, pp. 11–22.
[24] M. Kassab, An empirical study on the requirements engineering practices for agile software development. Proc. of EUROMICRO Intl. Conference on Software Engineering and Advanced Applications, 2014, pp. 254–261.
[25] F. Dalpiaz, A. Sturm, Conceptualizing requirements using user stories and use cases: a controlled experiment. Proceedings of the International Working Conference on Requirements Engineering: Foundation for Software Quality, 2020, pp. 221–238.
[26] J. Rumbaugh, I. Jacobson, G. Booch, The Unified Modeling Language Reference Manual, Pearson Higher Education, 2004.
[27] G. Sindre, A.L. Opdahl, Eliciting security requirements with misuse cases, Requir. Eng. 10 (1) (2005) 34–44.
[28] C. Larman, Applying UML and Patterns: an Introduction to Object-Oriented Analysis and Design and Iterative Development, Prentice Hall, 2004.
[29] A. Cockburn, Writing Effective Use Cases, Addison-Wesley Professional, 2000.
[30] P. Kruchten, The Rational Unified Process: An Introduction, Addison-Wesley, 2004.
[31] Y. Wautelet, S. Heng, M. Kolp, I. Mirbel, Unifying and extending user story models. Proceedings of the International Conference on Advanced Information Systems Engineering, 2014, pp. 211–225.
[32] G. Lucassen, F. Dalpiaz, J. van der Werf, S. Brinkkemper, Improving agile requirements: the quality user story framework and tool, Requir. Eng. 21 (3) (2016) 383–403.
[33] J. Abrial, Data Semantics, North-Holland, 1974.
[34] P.P.-S. Chen, The entity-relationship model–toward a unified view of data, ACM Trans. Database Syst. (TODS) 1 (1) (1976) 9–36.
[35] Y. Wand, R. Weber, Research commentary: information systems and conceptual modeling–a research agenda, Inf. Syst. Res. 13 (4) (2002) 363–376.
[36] T. Yue, L.C. Briand, Y. Labiche, Facilitating the transition from use case models to analysis models: approach and experiments, ACM Trans. Softw. Eng. Method. 22 (1) (2013).

[37] B. Anda, D.I. Sjøberg, Investigating the role of use cases in the construction of class diagrams, Empir. Softw. Eng. 10 (3) (2005) 285–309.
[38] M.H. Fortuna, C.M. Werner, M.R. Borges, Info cases: integrating use cases and domain models. Proceedings of the IEEE International Requirements Engineering Conference, 2008, pp. 81–84.
[39] Y. Wautelet, S. Heng, D. Hintea, M. Kolp, S. Poelmans, Bridging user story sets with the use case model. Proceedings of the International Conference on Conceptual Modeling Workshops, 2016, pp. 127–138.
[40] Y. Wautelet, M. Velghe, S. Heng, S. Poelmans, M. Kolp, On modelers ability to build a visual diagram from a user story set: a goal-oriented approach. Proceedings of the International Working Conference on Requirements Engineering: Foundation for Software Quality, 2018, pp. 209–226.
[41] M. Trkman, J. Mendling, M. Krisper, Using business process models to better understand the dependencies among user stories, Inf. Softw. Technol. 71 (2016) 58–76.
[42] D. Berry, R. Gacitua, P. Sawyer, S. Tjong, The case for dumb requirements engineering tools. Proceedings of International Conference on Requirements Engineering: Foundation for Software Quality, 2012, pp. 211–217.
[43] M. Saeki, H. Horai, H. Enomoto, Software development process from natural language specification. Proceedings of the International Conference on Software Engineering, ACM, 1989, pp. 64–73.
[44] L. Mich, NL-OOPS: From natural language to object oriented requirements using the natural language processing system LOLITA, Nat. Lang. Eng. 2 (1996) 161–187.
[45] H. Harmain, R. Gaizauskas, CM-Builder: A natural language-based CASE tool for object-oriented analysis, Autom. Softw. Eng. 10 (2) (2003) 157–181.
[46] V. Ambriola, V. Gervasi, On the systematic analysis of natural language requirements with CIRCE, Autom. Softw. Eng. 13 (1) (2006) 107–167.
[47] C. Arora, M. Sabetzadeh, L. Briand, F. Zimmer, Extracting domain models from natural-language requirements: approach and industrial evaluation. Proceedings of the International Conference on Model Driven Engineering Languages and Systems, ACM, 2016, pp. 250–260.
[48] V.B.R.V. Sagar, S. Abirami, Conceptual modeling of natural language functional requirements, J. Syst. Softw. 88 (2014) 25–41.
[49] S. España, M. Ruiz, A. González, Systematic derivation of conceptual models from requirements models: a controlled experiment. Proceedings of the International Conference on Research Challenges in Information Science, IEEE, 2012, pp. 1–12.
[50] O. Pastor, J. Gómez, E. Insfrán, V. Pelechano, The OO-method approach for information systems modeling: from object-oriented conceptual modeling to automated programming, Inf. Syst. 26 (7) (2001) 507–534.
[51] A. Ottensooser, A. Fekete, H.A. Reijers, J. Mendling, C. Menictas, Making sense of business process descriptions: an experimental comparison of graphical and textual notations, J. Syst. Softw. (2012).
[52] E. Cardoso, K. Labunets, F. Dalpiaz, J. Mylopoulos, P. Giorgini, Modeling structured and unstructured processes: an empirical evaluation. Proceedings of the International Conference on Conceptual Modeling, 2016, pp. 347–361.
[53] B. Hoisl, S. Sobernig, M. Strembeck, Comparing three notations for defining scenario-based model tests: a controlled experiment. Proceedings of the International Conference on the Quality of Information and Communications Technology, 2014.
[54] F. Dalpiaz, Requirements Data Sets (User Stories), 2018.Mendeley Data, v1 https://doi.org/10.17632/7zbk8zsd8y.1
[55] F. Dalpiaz, A. Sturm, Experiment User Stories vs. Use Cases, 2020, https://doi.org/10.23644/uu.c.4815591.v1.Figshare
[56] R.A. Fisher, On the 'probable error' of a coefficient of correlation deduced from a small sample, Metron 1 (1921) 1–32.
[57] J. Cohen, Statistical power analysis, Curr. Dir. Psychol. Sci. 1 (3) (1992) 98–101.
[58] F. Wilcoxon, Individual comparisons by ranking methods, Biom. Bull. 1 (6) (1945) 80–83.
[59] F. Dalpiaz, A. Sturm, P. Gieske, Extraction of Conceptual Models: User Stories vs. Use Cases, 2020.
[60] C. Wohlin, P. Runeson, M. Höst, M.C. Ohlsson, B. Regnell, A. Wesslén, Experimentation in Software Engineering, Springer, 2012.
[61] B.A. Kitchenham, S.L. Pfleeger, L.M. Pickard, P.W. Jones, D.C. Hoaglin, K. El Emam, J. Rosenberg, Preliminary guidelines for empirical research in software engineering, IEEE Trans. Softw. Eng. 28 (8) (2002) 721–734.
[62] T. Cziharz, P. Hruschka, S. Queins, T. Weyer, Handbook of requirements modeling IREB standard, Int. Requir. Eng. Board (2015).
[63] A. Ferrari, P. Spoletini, M. Bano, D. Zowghi, SaPeer approach for training students in requirements elicitation interviews—educational material, 2019.
[64] M. Bano, D. Zowghi, A. Ferrari, P. Spoletini, B. Donati, Learning from mistakes: an empirical study of elicitation interviews performed by novices. Proceedings of the IEEE International Requirements Engineering Conference, 2018, pp. 182–193.
[65] L.A. Maciaszek, Requirements Analysis and System Design: Developing Information Systems with UML, Addison-Wesley Longman Ltd., GBR, 2001.
[66] F.G. de Oliveira Neto, J. Horkoff, R. Svensson, D. Mattos, A. Knauss, Evaluating the effects of different requirements representations on writing test cases. Proceedings of the International Working Conference on Requirements Engineering: Foundation for Software Quality, 2020, pp. 257–274.