Conceptualizing Requirements using User Stories and Use Cases: A Controlled Experiment

Fabiano Dalpiaz[1] ✉[0000−0003−4480−3887] and Arnon Sturm[2][0000−0002−4021−7752]

[1] Utrecht University, The Netherlands
f.dalpiaz@uu.nl
[2] Ben-Gurion University of the Negev, Israel
sturm@bgu.ac.il

Abstract. [Context and motivation] Notations for expressing requirements are often proposed without explicit consideration of their suitability for specific tasks. Consequently, practitioners may choose a sub-optimal notation, thereby affecting task performance. **[Question / problem]** We investigate the adequacy of two well-known notations: use cases and user stories, as a starting point for the manual derivation of a static conceptual model. In particular, we examine the completeness and correctness of the derived conceptual model. **[Principal ideas / results]** We conducted a two-factor, two-treatment controlled experiment with 118 subjects. The results indicate that for deriving conceptual models, user stories fit better than use cases. It seems that the repetitions in user stories and their conciseness contribute to these results. **[Contribution]** The paper calls for evaluating requirements notations in the context of various requirements engineering tasks and for providing evidence regarding the aspects that need to be taken into account when selecting a requirement notation.

Keywords: Requirements Engineering · Conceptual Modeling · Use Cases · User Stories · Controlled experiment.

1 Introduction

Many notations exist for expressing requirements for software systems, ranging from natural language sentences [8], semi-formal models [32,22], to formal languages [3,13]. Among this landscape, requirements are most often expressed following some templates or controlled languages, like EARS [31], UML use cases, and user stories [6]. The adequacy of the notation depends on the type of system under design, the application domain, and the granularity of the requirements. Nevertheless, the research community overlooked the contextual adequacy of these notations, and thus evidence for practitioners on the selection of an effective notation that fits their needs is missing.

Furthermore, the selection of a suitable requirements notation also depends on the expectations regarding the requirements. They can be used for communication among stakeholders (i.e., when they express high-level specifications), for analytical tasks such as finding inconsistency and detecting feasibility, or for serving the entire software development process. As a first step within the development process, an analyst refines an initial set of high-level requirements into lower-level specifications and may use conceptual models as an artifact that represents the major entities and relationships that are referred to in the high-level requirements [18,29,42]. In this work, we limit our attention to static/structural conceptual models that emphasize the domain entities and their relationships. Such conceptual models can be employed in requirements engineering in order to (i) provide a holistic overview for team members to understand the product domain [1,27]; (ii) identify quasi-synonyms that may lead to misunderstandings [9]; (iii) support model-driven engineering [24]; and (iv) analyze certain quality aspects such as security and privacy [28].

In this research, we study the process of deriving a conceptual model (like an entity-relationship diagram or an UML class diagram) that represents the main concepts in a collection of high-level requirements. This type of models has been shown to be a useful learning tool for new employees [27], for representing the domain in which the system is to operate [1], and for supporting the transition to later phases in object-oriented software development [15,44].

We investigate the relative suitability of two mainstream notations for expressing requirements regarding analysts' effectiveness 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 derivation of static conceptual models?*

The two natural language 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 some criticism on their suitability to express requirements [14], they are widely adopted in the software industry. The latter are chosen because of their popularity in projects that follow agile development methods like Scrum and Kanban [19,26].

We answer our MRQ via a controlled experiment in which senior undergrad students, taking a course on object-oriented analysis and design are briefed to individually derive conceptual models (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 are able to measure the precision and recall. Furthermore, we evaluate the preference of the students in extracting models from either notation.

The results show 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 accuracy of the derived models tends to be higher with user stories. Although preliminary, we believe that these results may inspire other research on the effectiveness of alternative requirements notations for different requirements-related tasks.

The rest of the paper is organized as follows. In Section 2, we set the background for this study and review related studies. In Section 3, we present the design of the experiment we performed. In Section 4, we elaborate on the experiment results whereas in Section 5 we interpret and discuss those results. In Section 6, we indicate the threats to validity. We conclude and set plans for future research in Section 7.

2 Background and Related Work

Use cases are a popular notation, part of the UML [34], for expressing requirements that describe the interaction between a user and a system. Although typically used for expressing functional requirements, adaptations and extensions exist to make them suitable for representing quality aspects such as security and privacy [35,28]. A use case defines a list of steps between an actor and the system; they are specified following a textual template and using a use case diagram. In the context of this work, we focus on a simple template notation adapted from Larman's book [23], illustrated in Listing 1, which is based on the widely used notations by Cockburn [4] and Kruchten [21].

Listing 1. A use case for the Planning Poker game 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

User stories are another widespread notation [19,26] that originates from the agile software development paradigm [6] and that consists of simple descriptions of a feature written from the perspective of the person who wants them. Multiple templates exist for representing user stories [39], among which the Connextra format is one of the predominant ones [26]: *As a <role>, I want <action>, so that <benefit>*. The "so that" part, despite its importance in providing the rationale for a user story [25], is often omitted in practice. In our study, we formulate user stories using the Connextra template and we group related user stories into epics. See Listing 2 for some examples.

Listing 2. Some user stories 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.

Conceptual models can help refine an initial set of high-level requirements into lower-level specifications, by representing the major entities and relationships that are referred to in the high-level requirements [18,29,42]. Several researchers have investigated the derivation of such models from use cases. Insfrán *et al.* [18] propose a process that assists in the refinement of high-level requirements, expressed as a mission statement, 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.* [43] observe that informal use case specifications may contain vague and ambiguous terms that make it hard to derive precise UML models, including class and sequence diagrams. As a solution, they propose a restricted version of the use cases template for analysts to adopt. The approach was found easy to apply by practitioners and led to significant improvements in terms of class correctness and class diagram completeness.

Fewer methods exist that derive conceptual models from user stories. Lucassen *et al.* [27] propose an automated approach for extracting conceptual models from a collection of user stories by relying on and adapting natural language processing heuristics from the literature. The resulting models show good precision and recall, also thanks to the structure that is set 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.* [38] 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. Although this work is relevant, our goal is to study the independent use of UC and US. Trkman *et al.* [37] point out how user stories, being defined independently, do not clearly represent execution and integration dependencies. Their solution includes the use of a different type of conceptual model, i.e., 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.

To the best of our knowledge, no experimental studies that compare the effectiveness of requirements notations (for certain tasks) exist. Therefore, practitioners have no concrete evidence regarding which notation suits best their needs. The closest works to ours regard the comparison of (graphical) notations used in information systems design. Ottensooser *et al.* [33] compare the Business Process Modeling Notation (BPMN) against textual user cases in interpreting business process descriptions. Their experiment shows that BPMN adds value only with trained readers. Cardoso *et al.* [2] conduct an experiment that shows how the adequacy of languages depends on how structured a business process is. Hoisl *et al.* [17] compare three notations (textual, semi-structured, diagrammatic) for expressing scenario-based model tests; their experimental results show a preference toward natural language based notations.

3 Experiment Design

We investigate how requirements can be *conceptualized* taking as input two different widely used requirements notations: use cases and user stories. By conceptualizing requirements, we refer to the manual derivation of a static conceptual model starting from a set of requirements specified in each of the notations.

3.1 Hypotheses

To compare the differences in the effectiveness of UC and US as a starting point for the manual derivation of a conceptual model, we measure *correctness* 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.

We observe that use case descriptions are organized in a transactional, process-oriented fashion, thus making it easier to comprehend the flow and the intended system. User stories, on the other hand, are short and refined statements that have a standard format, thus making it easier to understand each requirement separately. Nevertheless, even if organized into epics, it is difficult to understand the system and the way it operates as a whole. This difference leads us to have the following hypothesis:

H0: user stories and use cases are equally good for the derivation of a static conceptual model

To measure the quality of a conceptual model, we use the recall and precision of the resulting model with respect to the gold standard one. Our hypotheses are formalized as follows:

$$H_0^{CM\text{-}Precision} : US^{CM\text{-}Precision} = UC^{CM\text{-}Precision}$$
$$H_0^{CM\text{-}Recall} : US^{CM\text{-}Recall} = UC^{CM\text{-}Recall}$$

3.2 Design

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

Independent Variables. The first variable is the notation 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. It has two possible values: Data Hub (DH) and Planning Poker (PP). These case studies are obtained from a publicly available dataset of user story requirements [7]. 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 types of dependent variables that are specified by comparing the elements in the *subject solution* (the conceptual model derived by a subject) against the *gold standard solution*:

- *Recall*: 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).

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

- *Precision*: 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).

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

While measuring recall and precision, 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

Furthermore, since relationships can be identified only when the connected entities are identified, we introduce an *adjusted* version of precision and recall for the relationships, which calculates precision and recall with respect to those relationships in the gold standard among the entities that the subject has identified. So, 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 relationship R3 is considered while computing precision and recall 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. They learned the use of 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 those assignments, they achieved good results, indicating that they understood the notations well.

Recruiting the students was done on a volunteering basis. Nevertheless, they were encouraged to participate in the experiment by providing them with additional bonus points to the course grade based on their performance. Before recruiting the students, the research design was submitted to and approved by the department ethics committee.

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 [10]), in which we alternate the treatment and the case study.

The form consists of 4 parts: (1) a pre-task questionnaire that checks the background and knowledge of the subjects; (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 that measure 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 was taught in the course).

We prepared the requirements set used in the experiment through several stages. First, we collected the original requirements from [7]. Second, we filtered the requirements to fit the experiment settings and wrote corresponding use case specifications. Third, we double checked the specifications to align contents and granularity of the use cases and the user stories and verified that both can be used to derive the same conceptual model, which we then set as the gold standard solution. Finally, we translated the requirements to Hebrew, so the subjects can perform the task in their native language. The case studies had different complexity. Table 1 presents various metrics and indicates that the DH case introduces higher complexity than the PP case.

Table 1. Case Studies Metrics

	Data Hub	Planning Poker
Number of user stories	24	21
Number of use cases	4	3
Number of lines in use cases	38	36
Number of entities	10	6
Number of relationships	13	8

To create the gold standard, whose conceptual models are listed in Appendix A, the authors of this paper have first created independently a conceptual model that depicts the main entities and relationships from both the use cases and the user stories separately. We further verified the conceptual model by adopting the heuristics used by the Visual Narrator [27], which suggest to look for (compound) nouns to identify concepts and to detect relationships by searching for action verbs. Then, we compared our models and produced the reconciled versions in the Appendix A.

Execution. The experiment took place in a dedicated time slot and lasted approximately 1 hour, although we did not set a time limit for the subjects to complete the tasks. The assignment of the groups (i.e., the forms) to the 118 subjects was done randomly. The distribution of groups was as followed:

- 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.

Analysis. The paper forms delivered by the students were checked by the second author against the gold standard. We performed a lenient analysis of the solutions, in which we did not consider over-specification: if the student had identified an additional entity mentioned in the requirements, which we deemed as an attribute rather than an entity, we would not penalize the student. We also accepted solutions that deviated slightly from the gold standard, so we could cope with alternative solutions. The results were encoded into IBM's SPSS, which was used to calculate precision, recall, and the other statistics listed in Section 4. Both authors analyzed the data to cross-check the analyses and to identify the most relevant findings.

4 Experiment Results

We run a series of analyses over the results (all materials are available online [10]). We first compare the background of the two groups. Table 2 presents the comparison criteria among the groups, the mean ($\overline{x}$) and standard deviation (σ) for each, and the statistical analysis results in terms of statistical significance ($p < 0.05$), and effect size. For non-parametric tests, like Wilcoxon Signed-Rank test or Mann-Whitney U test, we follow Frits *et al.*'s recommendation [12]: test statistics are approximated by normal distributions to report effect sizes. For parametric tests, like the T-Test, we employ Hedges' g value [16]. To facilitate the interpretation of the results, we transform effect sizes to

Hedges' g. To do so, we employ an online calculator[1]. All criteria were indicated by the subjects except for the grade, which is the final grade of the course. The familiarity and the homework participation criteria were retrieved using a 5-point Likert-type scale (1 indicates low familiarity and participation and 5 indicates high familiarity and participation), lecture participation criteria take either true or false, while the grade is on a scale from 0 to 100.

Table 2. Pre-Questionnaire Results: mean, standard deviation, significance, and effect size.

	Form A		Form B		p	Effect
	$\overline{x}$	σ	$\overline{x}$	σ		size
Class Diagram (CD) Familiarity	**4.16**	**0.71**	4.07	0.60	0.356	0.147
UC Familiarity	**3.73**	**0.62**	3.57	0.74	0.293	0.174
US Familiarity	**3.92**	**0.76**	3.70	0.69	0.111	0.298
UC Homework Delivered	**4.45**	**0.80**	4.34	0.66	0.153	0.237
US Homework Delivered	**4.48**	**0.66**	4.26	0.73	0.066	0.308
CD Homework Delivered	**4.54**	**0.57**	4.30	0.74	0.071	0.301
Participation in the UC Lecture	**0.91**	**0.29**	0.89	0.32	0.629	0.065
Participation in the US Lecture	**0.95**	**0.23**	0.90	0.30	0.349	0.186
Grade	**85.54**	**5.87**	81.49	14.58	0.048	0.360

Although the division into groups was done randomly, it appears that the background of the group of subjects assigned to Form A was superior than the group of subjects assigned to Form B. Applying a Mann-Whitney test [30] to the "subjective" criteria, we found no statistically significant differences, yet when applying a T-test [36] to the last three rows, we found that the difference for the *grade* was statistically significant and had a *small-to-medium* effect size ($g = 0.360$). These differences should be taken into account when analyzing the results.

In analyzing the results of the completeness and correctness of the conceptual models, we performed an Anova test [11] and found out that the interaction between the case study and the notation, concerning the adjusted total precision and recall, is statistically significant. This probably occurred due to the complexity differences between the two case studies as appears in Table 1. We thus analyze each case study separately.

Table 3 and Table 4 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 gray rows indicate statistically significant differences (applying T-test).

In all metrics, the conceptual models derived from the set of user stories outperform the conceptual models derived from the set of use cases. For the DH case study, the difference was statistically significant in the case of the relation, for all the adjusted metrics as well as for the total precision. Furthermore, the effect sizes for DH indicate an *intermediate effect* for many metrics, all those with $g > 0.5$ according to Cohen [5]. Analyzing the preferences of the students regarding the use of the two notations (Table 5)—using the Mann-Whitney test since our visual analysis of the distributions

[1] `https://www.psychometrica.de/effect_size.html`

Table 3. Data Hub Results.

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

Table 4. Planning Poker Results.

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

revealed non-normality for some statements—we found no statistically significant differences between the two groups. In the tasks related to deriving a conceptual model, identifying classes, identifying relationships, providing a system overview, and clearly presenting a single requirement, there was a consensus regarding the benefits of using user stories to describe the requirements. However, there was no consensus regarding their benefit over use cases with respect to comprehending the system structure. Furthermore, in both groups most subjects generally prefer to use user stories to use cases.

Gathering the preferences of both groups together, Table 6 indicates a clear preference towards user stories. These preferences are of statistical significance (applying Wilcoxon test [40]) 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 *intermediate effect*, equal or above to 0.5.

Based on the results, we can conclude that for the Data Hub application we can reject both H_0 hypotheses on the equality of both notation in the effectiveness of deriving a conceptual model for the metrics defined above (the grey rows in Table 3). In that case, introducing user stories resulted in better conceptual models. For the other metrics, we accept H_0 hypotheses, and can infer that no difference exists in using both notations in

Table 5. Preferences by group; effect size is omitted due to the high p values.

Statement	Form A	Form B	p
Use cases fit for developing a conceptual model	3.24	3.45	0.267
User stories fit for developing a conceptual model	**3.87**	**3.69**	0.324
Use cases help in identifying classes	3.48	3.46	0.966
User stories help in identifying classes	**3.74**	**3.86**	0.809
Use cases help in identifying relationships	3.63	3.46	0.531
User stories help in identifying relationships	**3.72**	**3.66**	0.534
Use cases help comprehend the system structure	3.50	**3.63**	0.402
User stories help comprehend the system structure	**3.57**	3.57	0.855
Use cases provide a system overview	3.28	3.36	0.563
User stories provide a system overview	**3.54**	**3.43**	0.648
A use case clearly presents a single requirement	3.41	3.32	0.730
A user story clearly presents a single requirement	**3.74**	**3.86**	0.583
Which method do you prefer?	**US=31** UC=21	**US=37** UC=18	0.413

Table 6. Preferences by notation.

Statement	User Stories		Use Cases		p	Effect
	$\overline{x}$	σ	$\overline{x}$	σ		size
Fit for developing a conceptual model	**3.78**	**0.91**	3.35	1.00	0.002	0.613
Help in identifying classes	**3.79**	**0.92**	3.46	0.87	0.010	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	0.613
Which method do you prefer?	68		39			

deriving a conceptual model. Drilling down into the actual conceptual models and their alignment with the gold standard solution, we had additional observations.

For the Data Hub case study:

1. *Site Admin* was less recognized in the use cases (US-96%, UC-80%) – it appears only once in the use cases and 4 times in the user stories.
2. *Usage Metric* was less recognized in the user stories (US-31%, UC-69%) – it appears only once in the user stories and 4 times in the use cases.
3. *Billing System* was less recognized in the use cases (US-50%, UC-27%) – though in both techniques it appears only once.
4. *Account* was less recognized in the use cases (US-50%, UC-19%) – it appears twice in the use cases and 4 times in the user stories.
5. The *Publisher – Usage Metrics* relationship was less recognized in the user stories (US-5%, UC-44%) – it appears twice in the use cases and only implicitly in the user stories.
6. The *Publisher – User* relationship was less recognized in the use cases (US-59%, UC-18%) – this relationship is implicitly mentioned 3 time in the user stories and only once in the use cases.

7. *The Site Admin – User* relationship was less recognized in the use cases (US-59%, UC-18%) - it appears once in each of the descriptions.

For the Planning Poker case study, we had the following observations:

1. *Game Round* was less recognized in the user stories (US-37%, UC-61%) – it appears 9 time in the use cases and only twice in the user stories.
2. The *Moderator – Estimator* relationship was less recognized in the use cases (US-37%, UC-14%) – an implicit relationship between the moderator and estimator appears 7 times in the user stories and only 2 times in the use cases.
3. The *Moderator – Estimation Item* relationship was less recognized in the use cases (US-50%, UC-14%) – an implicit relationship between the moderator and estimation item appears 6 times in the user stories and only 4 times in the use cases.

5 Discussion

As highlighted in Table 1, the *complexity of the case studies* affects 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.

The results indicate the existence of *differences between the groups*. The best performing group had the Data Hub case specified with user stories and the Planning Poker specified with use cases. That group achieved better results in the case of the Data Hub (having user stories). The other group (inferior in the subjects' background and grading) also achieved better results when having user stories. The later results, related to Planning Poker, were not statistically significant; yet, this can be attributed to the fact that the group that had the user stories was inferior to the group that had the use cases. Another explanation can be related to the complexity of the case study: the Planning Poker case study was simpler and, therefore, differences were of low magnitude.

When referring to the qualitative inspection of the results, it seems that when related to actors, there are *multiple repetitions* of these concepts 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 in 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 user stories supremacy may be the fact 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. It would be interesting, therefore, to explore which of the two notations is the most adequate to re-construct a process model that describes how the actions are sequentially linked.

One major difficulty for the students was to understand the *difference between entities and relationships*. Several errors were made because the students did not mark the identified concepts correctly. This may be due to shallow reading, time pressure, or an actual difficulty in distinguishing them. However, it seems that the template structure of user stories made it easier for the students to distinguish entities from relationships.

The subjects also perceive the user stories notation as better fit for the tasks we ask 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 students also acknowledge the benefits for other tasks, yet the difference was not significant.

6 Threats to Validity

Our results need to be considered in view of threats to validity. We follow Wohlin *et al.*'s classification [41]: construct, internal, conclusion, and external validity.

Construct validity threats concern the relationships between theory and observation and are mainly due to the method used to assess the outcomes of the tasks. We examined the use of two RE notations for the purpose of conceptual model derivation and we used two sets of requirements. The selection of the domains may affect the results; our choice is justified by our attempt to provide domains that would be easy to understand by the subjects. Moreover, it might be that the specification using the two notations were not aligned in the sense that they emphasize different aspects (process in the software vs. individual features). However, this is exactly one of the triggers of our research. To mitigate the risk of favoring one notation over the other, before the experiment, both authors created a conceptual model from either notation, independently, to minimize bias that could stem from the way the specifications were written.

Internal validity threats, which concern external factors that might affect the dependent variables, may be due to individual factors, such as familiarity with the domain, the degree of commitment by the subjects, and the training level the subjects underwent. These effects are mitigated by the experiment design that we chose. We believe that due to the domains characteristics, the students were not familiar with them, 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 as they took advantage of entire time allocated for the experiment. The revealed differences among the groups may also affect the results, though the trend exist in both groups. 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 better conceptual model. Yet, we attribute this difference to the lower complexity of the second domain. This leads to another threat that the order of the domains within the experiment may also affect the results. Another threat could emerge from the multiple tasks and fatigue.

Conclusion validity concerns the relationship between the treatment (the notation) and the outcome. 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 the experiment, for grading the subjects' answers; thus, only limited human judgment was required.

External validity concerns the generalizability of the results. The main threat in this area stems from the choice of subjects and from using simple experimental tasks. The subjects were undergraduate students with little 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 [20], 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.

7 Summary

We provided initial evidence on the benefit of using user stories over use cases for deriving static conceptual models. We performed a controlled experiment with 118 undergraduate students that we conducted as part of a system design course. The results indicate that, probably because of the conciseness and focus of user stories and the repetitions of entities in the user stories, the derived conceptual models are more complete and correct.

This work is a first attempt in the direction of evaluating requirements notations side-by-side for a specific task. We started from the task of deriving a static conceptual model; this is a self-contained activity that can be performed in a relatively short time, especially for not-so-large specifications.

Our paper calls for further experimentation for this particular task, as well as for other tasks that are based on requirements notations. The research community needs to build a corpus of evidence to assist practitioners in the choice of a notation (and technique) for the RE tasks at hand. We plan to continue this research with larger case studies using qualitative methods to investigate the trade-offs among RE notations.

References

1. Arora, C., Sabetzadeh, M., Nejati, S., Briand, L.: An Active Learning Approach for Improving the Accuracy of Automated Domain Model Extraction. ACM Transactions on Software Engineering and Methodology **28**(1) (2019)

2. Cardoso, E., Labunets, K., Dalpiaz, F., Mylopoulos, J., Giorgini, P.: Modeling Structured and Unstructured Processes: An Empirical Evaluation. In: Proc. of the Intl. Conference on Conceptual Modeling. pp. 347–361 (2016)
3. Ciancarini, P., Cimato, S., Mascolo, C.: Engineering Formal Requirements: An Analysis and Testing Method for Z Documents. Annals of Software Engineering **3**(1), 189–219 (1997)
4. Cockburn, A.: Writing Effective Use Cases. Addison-Wesley Professional (2000)
5. Cohen, J.: Statistical Power Analysis. Current Directions in Psychological Science **1**(3), 98–101 (1992)
6. Cohn, M.: User Stories Applied: for Agile Software Development. Addison Wesley (2004)
7. Dalpiaz, F.: Requirements Data Sets (User Stories) (2018), `http://dx.doi.org/10.17632/7zbk8zsd8y.1`, Mendeley Data, v1
8. Dalpiaz, F., Ferrari, A., Franch, X., Palomares, C.: Natural Language Processing for Requirements Engineering: The Best Is Yet to Come. IEEE Software **35**(5), 115–119 (2018)
9. Dalpiaz, F., van der Schalk, I., Brinkkemper, S., Aydemir, F.B., Lucassen, G.: Detecting Terminological Ambiguity in User Stories: Tool and Experimentation. Information and Software Technology (2019)
10. Dalpiaz, F., Sturm, A.: Experiment User Stories vs. Use Cases (2020). https://doi.org/10.23644/uu.c.4815591.v1, Figshare
11. Fisher, R.A.: On the 'Probable Error' of a Coefficient of Correlation Deduced from a Small Sample. Metron **1**, 1–32 (1921)
12. Fritz, C.O., Morris, P.E., Richler, J.J.: Effect Size Estimates: Current Use, Calculations, and Interpretation. Journal of Experimental Psychology: General **141**(1), 2 (2012)
13. Fuxman, A., Liu, L., Mylopoulos, J., Pistore, M., Roveri, M., Traverso, P.: Specifying and Analyzing Early Requirements in Tropos. Requirements Engineering **9**(2), 132–150 (2004)
14. Glinz, M.: Problems and Deficiencies of UML as a Requirements Specification Language. In: Proc. of the Intl. Workshop on Software Specifications & Design. pp. 11–22 (2000)
15. Harmain, H., Gaizauskas, R.: CM-Builder: A Natural Language-based CASE Tool for Object-oriented Analysis. Automated Software Engineering **10**(2), 157–181 (2003)
16. Hedges, L.V.: Estimation of Effect Size from a Series of Independent Experiments. Psychological Bulletin **92**(2), 490 (1982)
17. Hoisl, B., Sobernig, S., Strembeck, M.: Comparing Three Notations for Defining Scenario-based Model Tests: A Controlled Experiment. In: Proc. of the Intl. Conference on the Quality of Information and Communications Technology (2014)
18. Insfrán, E., Pastor, O., Wieringa, R.: Requirements Engineering-based Conceptual Modelling. Requirements Engineering **7**(2), 61–72 (2002)
19. Kassab, M.: An Empirical Study on the Requirements Engineering Practices for Agile Software Development. In: Proc. of EUROMICRO Intl. Conference on Software Engineering and Advanced Applications. pp. 254–261 (2014)
20. Kitchenham, B.A., Pfleeger, S.L., Pickard, L.M., Jones, P.W., Hoaglin, D.C., El Emam, K., Rosenberg, J.: Preliminary Guidelines for Empirical Research in Software Engineering. IEEE Transactions on Software Engineering **28**(8), 721–734 (2002)
21. Kruchten, P.: The Rational Unified Process: An Introduction. Addison-Wesley (2004)
22. van Lamsweerde, A.: Requirements Engineering: From System Goals to UML Models to Software Specifications. Wiley (2009)
23. Larman, C.: Applying UML and Patterns: An Introduction to Object-Oriented Analysis and Design and Iterative Development. Prentice Hall (2004)
24. Loniewski, G., Insfran, E., Abrahão, S.: A Systematic Review of the Use of Requirements Engineering Techniques in Model-driven Development. In: Proc. of the Intl. Conference on Model Driven Engineering Languages and Systems. pp. 213–227 (2010)

25. Lucassen, G., Dalpiaz, F., van der Werf, J., Brinkkemper, S.: Improving Agile Requirements: The Quality User Story Framework and Tool. Requirements Engineering **21**(3), 383–403 (2016)
26. Lucassen, G., Dalpiaz, F., van der Werf, J., Brinkkemper, S.: The Use and Effectiveness of User Stories in Practice. In: Proc. of the Intl. Working Conference on Requirements Engineering: Foundation for Software Quality. pp. 205–222 (2016)
27. Lucassen, G., Robeer, M., Dalpiaz, F., van der Werf, J.M.E., Brinkkemper, S.: Extracting Conceptual Models from User Stories with Visual Narrator. Requirements Engineering **22**(3), 339–358 (2017)
28. Mai, P.X., Goknil, A., Shar, L.K., Pastore, F., Briand, L.C., Shaame, S.: Modeling Security and Privacy Requirements: a Use Case-Driven Approach. Information and Software Technology **100**, 165–182 (2018)
29. Maiden, N.A.M., Jones, S.V., Manning, S., Greenwood, J., Renou, L.: Model-driven Requirements Engineering: Synchronising Models in an Air Traffic Management Case Study. In: Proc. of the Intl. Conference on Advanced Information Systems Engineering. pp. 368–383 (2004)
30. Mann, H.B., Whitney, D.R.: On a Test of Whether one of Two Random Variables is Stochastically Larger than the Other. The Annals of Mathematical Statistics (1947)
31. Mavin, A., Wilkinson, P., Harwood, A., Novak, M.: EARS (Easy Approach to Requirements Syntax). In: Proc. of of the IEEE Intl. Requirements Engineering Conference. pp. 317–322 (2009)
32. Mylopoulos, J., Chung, L., Yu, E.: From Object-oriented to Goal-oriented Requirements Analysis. Communications of the ACM **42**(1), 31–37 (1999)
33. Ottensooser, A., Fekete, A., Reijers, H.A., Mendling, J., Menictas, C.: Making Sense of Business Process Descriptions: An Experimental Comparison of Graphical and Textual Notations. Journal of Systems and Software (2012)
34. Rumbaugh, J., Jacobson, I., Booch, G.: The Unified Modeling Language Reference Manual. Pearson Higher Education (2004)
35. Sindre, G., Opdahl, A.L.: Eliciting Security Requirements with Misuse Cases. Requirements Engineering **10**(1), 34–44 (2005)
36. Student: The Probable Error of a Mean. Biometrika **6**(1), 1–25 (1908)
37. Trkman, M., Mendling, J., Krisper, M.: Using Business Process Models to Better Understand the Dependencies among User Stories. Information and Software Technology **71**, 58–76 (2016)
38. Wautelet, Y., Heng, S., Hintea, D., Kolp, M., Poelmans, S.: Bridging User Story Sets with the Use Case Model. In: Proc. of the Intl. Conference on Conceptual Modeling Workshops. pp. 127–138 (2016)
39. Wautelet, Y., Heng, S., Kolp, M., Mirbel, I.: Unifying and Extending User Story Models. In: Proc. of the Intl. Conference on Advanced Information Systems Engineering. pp. 211–225 (2014)
40. Wilcoxon, F.: Individual Comparisons by Ranking Methods. Biometrics Bulletin **1**(6), 80–83 (1945)
41. Wohlin, C., Runeson, P., Höst, M., Ohlsson, M.C., Regnell, B., Wesslén, A.: Experimentation in Software Engineering. Springer (2012)
42. Yue, T., Briand, L.C., Labiche, Y.: A Systematic Review of Transformation Approaches between User Requirements and Analysis Models. Requirements Engineering **16**(2), 75–99 (2011)
43. Yue, T., Briand, L.C., Labiche, Y.: Facilitating the Transition from Use Case Models to Analysis Models: Approach and Experiments. ACM Transactions on Software Engineering and Methodology **22**(1) (2013)

44. Yue, T., Briand, L.C., Labiche, Y.: aToucan: An Automated Framework to Derive UML Analysis Models from Use Case Models. ACM Transactions on Software Engineering and Methodology **24**(3) (2015)

Appendix A

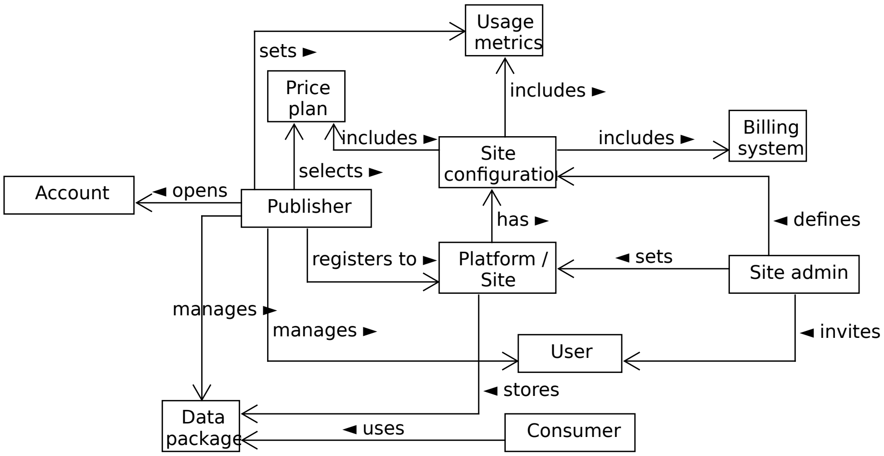

Fig. 1. The Data Hub gold standard solution

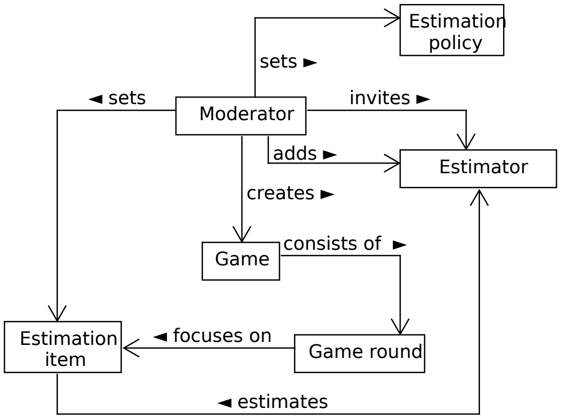

Fig. 2. The Planning Poker gold standard solution