Evaluating the Effectiveness of Suggestions for Conceptual Modeling

Bachelor's Thesis

Milo Plomp
5515599

A thesis presented for the
BSc. Information Science
August 2018

Utrecht University
Department of Computing Science

Supervised by Fatma Başak Aydemir
Second Supervisor: Fabiano Dalpiaz

Preface

I wrote this thesis as the final step towards completing my bachelor's degree in Information Science at Utrecht University. Although I have always felt quite sure which topics in Information Science I was the most passionate about, picking the right thesis topic proved to be quite challenging. After some considerations, I settled on an internal project after my supervisor proposed some projects. I decided to choose the project that I found interesting, even though that topic seemed quite challenging. And having little to no knowledge about programming, natural language processing, setting up an empirical evaluation experiment, and running statistical tests did turn out to make this thesis the most challenging project I have ever done. But it has also been a great learning experience, allowing me to say that I am proud of what I have learned and the research that I have produced.

First and foremost, I would like to thank my first supervisor, Fatma Başak Aydemir, for her support and feedback. I have thoroughly enjoyed contributing to your research and your involvement in my research has been greatly motivating. Thank you for the opportunity of taking on this project, motivating me to get the most out of my research, allowing me to ask (so many) questions at any time, and taking so much time out of your days to work on our tools.

Secondly, I would like to thank my experiment participants. Finding (initially) 20 students that were willing to take 1,5 hours out of their day to voluntarily help me seemed like a daunting task, but I was pleasantly surprised at how willing you were to help. Without you, writing this thesis would have been impossible.

Lastly, I would like to thank Fabiano Dalpiaz, my second supervisor. It is rare that a second supervisor is involved during a research project before it is finished, but it has been a great motivator to have two supervisors interested in my research. Thank you for allowing me to take on this project, and for your feedback during the meetings.

Contents

Chapter 1

Introduction

This research aims to provide empirical evidence for the effectiveness of natural language processing (NLP) powered suggestions on collaborative conceptual modeling. We focus on the effectiveness of a web-based approach for analyzing a group of models in a domain, and consulting a domain ontology, in order to detect missing concepts to suggest to modelers.

1.1 Research aim

The research on supporting collaborative modeling using suggestions has been promising, but lacks empirical evaluation on the effectiveness of the used methods and modeler-service interaction. Our aim is to provide this empirical evidence by doing the following:

- Evaluating the influence of suggestions on model creation, by conducting an empirical evaluation with possible end users
- Doing a qualitative analysis on modeler-service interaction to guide further developments of the service
- Implementing the suggestion service in modeling tools for two new languages, and testing differences in effectiveness between these languages
- Evaluating the influence of the suggestion service on various aspects of the model creation process, and determining differences in effectiveness

Research into the use of NLP to assist the model creation process is fairly limited. Most proposed solutions focus on the analysis of requirement documents, or aim to create conceptual models without (or with minimal) human interference [30, 26, 40]. Our research focuses on evaluating an approach that differs from the existing literature in the sense that it uses NLP to assist the collaborative modeling process, thereby not restricting human interference. This way, more emphasis is placed on support of and collaboration between human modelers, rather than automated model generation.

1.2 Research structure

When creating large-scale multi-concern models, effective collaboration is difficult to achieve, but often necessary to create a unified view of the models [31]. Aydemir & Dalpiaz developed a suggestion service that aids the collaboration of modelers working on separate models in the same domain [1]. This service aims to align conceptual models of large-scale systems by applying NLP techniques to generate suggestions for models to be created. Models for the same domain that capture different concerns are grouped in projects, and rely on each other for suggestion generation. In that regard, the application of this service is mainly in the field of online collaborative modeling, where modelers use online tools to (asynchronously) work on models, often times also in different geographical locations. Examples of fields where this is an often present situation include software development and enterprise modeling. Modelers are being allowed to borrow from knowledge in other models in the same project, possibly aligning these models and increasing collaboration. Since this approach uses only natural language text for analysis, there is no need to understand meta-models of the modeling languages that are used, allowing for easier implementation of the service and making the approach meta-model agnostic.

This thesis reports the design and results of an empirical study that is conducted to analyze the effectiveness of conceptual modeling techniques, specifically the influence of the use of the suggestion service. The set-up of this experiment is based on elements of previous research, which are listed in the following paragraphs. As a first step, iStar and BPMN are chosen as the modeling langues to be used for evaluation, based on their often combined presence in modeling projects [5, 6, 7]. Consequently, two open source modeling tools, both supporting one of these languages, are integrated with the suggestion service.

The empirical evaluation relies on the involvement of participants that represent real-life users. A sample group of possible end users is chosen, based on [10, 19, 21]. The experiments (including the modeling task and interview) are conducted in a laboratory setting [27, 16], where participants are asked to perform a modeling task based on previous research. The resulting models are compared to models created previously, which serve as control models [2]. Participants are not presented with a base model, but start the modeling task from scratch [13, 14].

The next chapter will introduce our four research questions, which define quality factors for models to be created and the nature of modeler-service interaction. The set-up and answering of these research questions relies heavily on literature, such as [13, 14] for research question 1, [11, 18, 39] for research question 3, [1, 11, 18, 39] for research question 4, and [1] for all.

1.3 Research questions

Our empirical evaluation approach can be used to determine effectiveness of conceptual modeling techniques, but is specified to evaluating the effectiveness of the suggestion service. Based on our research aim, and the structure that is used for this thesis, we define the following research questions:

RQ1 How much does this suggestion service improve model completeness?

RQ2 How much does this suggestion service help in using the syntactically correct terminology?

RQ3 How much does this suggestion service improve model alignment?

RQ4 How does the modeler-service interaction for this suggestion service affect its effectiveness?

All questions can only be answered by conducting an empirical evaluation, as this evaluation produces models that can be tested for completeness, syntax, and alignment, and allows us to evaluate modeler-service interaction.

1.4 Thesis structure

The next chapter elaborates on the design of our experiment for empirical evaluation. A technical set-up description will provide information on the chosen modeling languages, and the open source tools that are extended to implement the suggestion service. After that, the experiment set-up is explained, providing several possible scenarios to be used. The chapter will conclude with an approach to answering every research question, including necessity for answering that research question, the approach to data collection, and the approach to data analysis.

Chapter 3 will give answers to the research questions, by providing the analysis of results for every question. For every question, the approach to data analysis and statistical analysis is described. All quantitative and qualitative results are listed, followed by (statistical) analysis results and conclusions for every research question.

The discussion in chapter 4 will compare our results and research approach to other research, and will provide suggestions for future research. In this discussion, we focus on four main themes relating to our research. This thesis will conclude with a conclusion in chapter 5, where the main results are briefly highlighted.

Chapter 2

Experiment Design

2.1 Technical set-up

2.1.1 Suggestion service for requirements modeling

The base for the tools that are created for this research is a suggestion service created by Aydemir and Dalpiaz [1]. This suggestion service uses natural language processing (NLP) techniques to analyze conceptual models in machine-readable formats (in this research, JSON[1] is chosen as the file extension to be used). Labels from the models are extracted and stored in a database. These models are subsequently used, together with the domain ontology, to generate suggestions. Feedback on these suggestions is stored to filter future suggestions.

2.1.2 Modeling languages

This suggestion service has previously been implemented in a web-based modeling interface for the PACAS project[2] and evaluated with domain experts via a focus group. Further evaluation with participants that represent real-life users was necessary to assess modeler-service interaction and the effectiveness of model alignment.

The aim of the suggestion service is effective model alignment, even when the meta-models of models do not align. To evaluate this form of model alignment, it was necessary to create two modeling tools that produced models in different modeling languages and with different meta-models. For this research, bpmn-js[3] (for BPMN 2.0) and piStar[4] (for iStar 2.0) were chosen as the base-tools which could be extended to include the suggestion service.

BPMN (Business Process Model and Notation) is a graphical expression of business processes, and therefore often focuses on the *how* and *what* questions [5]. The i* (iStar) language is a goal-oriented language, allowing for a graphical representation of goals and how to achieve those goals. It is therefore applicable to both requirements engineering and business modeling, putting

[1] https://www.json.org/
[2] https://www.pacasproject.eu
[3] https://bpmn.io/toolkit/bpmn-js/
[4] http://www.cin.ufpe.br/ jhcp/pistar/

it in the same domain as BPMN [6]. The combination of these two modeling languages enables the representation of systems that support businesses as-is, but also the representation of what systems should do [7]. Since there is a high probability of encountering the use of both of these languages in real-world cases, these languages were chosen as the modeling languages for this experiment.

2.1.3 Modeling tools extension

Extending bpmn-js and piStar

We have chosen bpmn-js and piStar as the tools to be extended for this research[5]. Both bpmn-js[6] and piStar[7] are open source web-based tools, written in JavaScript, for viewing and creating BPMN and iStar models. The base bpmn-js tool used for this research is the *modeler* example[8]. Both tools include functionalities for creating models with all language-specific elements, loading models, and saving models (in XML for bpmn-js, JSON for piStar, and SVG for both).

Several adaptions to these tools were necessary to enable communication with the suggestion service. The first step is to allow for the suggestions to have a place on the canvas. Following the same design as the bpmn-js tool, all elements for communicating with the suggestion service and displaying the results are included on the screen. These elements include: a text field for the experiment number, a button for requesting suggestions, a button for displaying previous suggestions, and buttons to download the model in a (simplified) JSON format.

When a user requests suggestions, a commit file containing every model element is sent to the suggestion service. The tools provide an XML model containing the complete model (including element coordinates and elements without labels). This XML file is converted to JSON and filtered to only include the element id's (assigned by the tools) and element names (only elements that have labels are included). A POST request is made, including the experiment number in the URL to distinguish between experiment results.

After the JSON model is committed, a POST request for the suggestion file is made. The service sends back a JSON file containing every suggestion (5 at a time) with its name and suggestion id. This id is not displayed to the user, but is required for communicating feedback to the service. Suggestions are displayed in a table on screen by parsing the suggestion file and inserting rows for every element. For every suggestion, two buttons are displayed to allow for "useful" or "not useful" feedback.

When the user clicks on one of the feedback buttons, the corresponding suggestion is removed from the table and pushed to the past suggestion table. Instantly, feedback is sent to the suggestion service in a JSON file containing the suggestion id, suggestion name, and the feedback.

One problem with running these tools locally and communicating with a web server, is that the browser usually restricts access to resources on that server due to the server and application having different origins. This problem is solved with Cross-Origin Resource Sharing (CORS) by using a browser extension during the experiments.

[5]Tools are available via: https://github.com/RELabUU/icva-experiments/blob/master/ThesisTools.zip
[6]https://bpmn.io/toolkit/bpmn-js/
[7]http://www.cin.ufpe.br/ jhcp/pistar/
[8]https://github.com/bpmn-io/bpmn-js-examples/tree/master/modeler

Tools overview

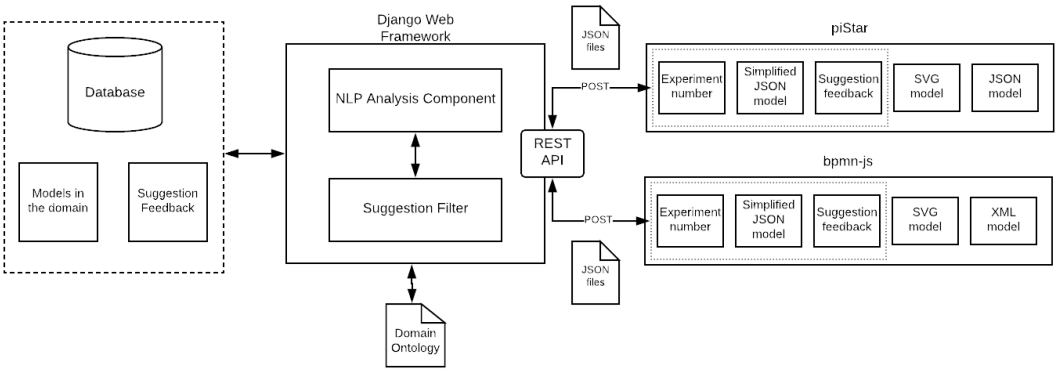

Figure 2.1: Architectural view of the suggestion service and the modeling tools. Adapted by author from figure 3 in [1].

Figure 2.1 shows an adapted architectural view of the suggestion service described in the paper of Aydemir and Dalpiaz [1], with the extension of the modeling tools created for this research (piStar and bpmn-js). It is important to note that for communication with the suggestion service, only the experiment number, simplified JSON model and suggestion feedback is used. All three of these data elements were not available in the base versions of piStar and bpmn-js.

The experiment number is manually updated by the observer and separates model information for different cases. The next section will provide more information on the experiment set-up. The simplified JSON model is a selective version of the textual models created by the tools, incorporating only element id's and label content (since the suggestion service analyzes the textual models using NLP). Lastly, the suggestion feedback is sent as a JSON file to the service as well. This file gets constructed with the first suggestion feedback and is updated accordingly. Communication with the service uses a POST request over the REST API.

2.2 Tool evaluation

2.2.1 Participants

In order to effectively test the modeling tools, it is crucial to involve participants that represent real-life users [10]. Involving possible end-users (or accurate representations thereof) allows for gathering results that would be similar to real-world outcomes, thus creating a more effective and representative experiment. Although the implemented suggestion-service is language-independent [1], a conscious choice for representative participants was made: the involvement of participants with knowledge of modeling (languages) defers the focus of the experiments away from explanation of the tools and experiments, resulting in a more efficient experiment. These experiments will therefore involve a group of 20 Information Science students from Utrecht University who have completed courses in modeling BPMN and iStar. Determining an adequate

sample size for qualitative research is often disputed as it relies on factors such as cultural preferences and resource restraint [19]. Based on a literature study on sample size in information science research by Marshall et al., we determine a sample size of 20 to be sufficient and feasible [21].

The decision for familiarity in modeling also extends to the domain to be modeled. Participants are asked to model the thesis registration process for the Master of Business Informatics (MBI) at Utrecht University; a case that is similar to processes they are familiar with, but not entirely familiar.

2.2.2 Experiment set-up

Experiments are conducted at Utrecht University, in a dedicated laboratory setting. This allows for the creation of conditions that are as similar as possible for each experiment, since there are no location differences and exterior interference. A laboratory setting is well-suited for this experiment, since the experiment involves individuals working on a relatively well-defined case [27]. It should be noted that this setting takes the usage of the tools outside of their intended context [16], but this downside is disregarded due to these experiments serving as a primary academic evaluation of the effectiveness of the suggestion service.

The MBI thesis registration process has been modeled to create base models for evaluating the effectiveness of the service. These BPMN models, iStar models, and UML class diagrams were produced in 1 hour. The iStar and BPMN base models will only be used as control models to which the models created with the suggestion service can be compared. Figure 2.1 shows "models in the same domain" that are loaded into the database. Only the class diagram and a model of the domain ontology will be loaded into the database, and will therefore be the only base models for generating suggestions. This will ensure that every participant receives suggestions based on the same models (apart from their created model), removing any bias that arises from differences in completeness of the base iStar and BPMN models. An overview of the experiment set-up can be seen in table 2.1.

Table 2.1: Experiment set-up overview

Human Modelers	**Models for suggestions**	**Control Model**	**Evaluation**
10 iStar modelers	Domain ontology and class diagram	Base iStar model	Models created by human modelers are compared to corresponding base models
10 BPMN modelers	Domain ontology and class diagram	Base BPMN model	

We have chosen not to include a control group for this research, since the base models serve as adequate control measures. Including a second control group in this research would be superfluous. Comparing the output of these experiments to the base models provides enough information about the influence of the suggestion service on model effectiveness [2].

Experiment structure

There are three possible structures for this experiment.

- *Scenario 1:* In the first scenario, there are two modelers working on one project at the same time. This would simulate a real-world scenario where modelers are dependent on

each other's models to receive suggestions. However, suggestions can only be generated when models are committed and stored in the database. When suggestions are requested by one modeler before a commit from the other modeler, suggestions will be generated based only on the domain ontology. This could be prevented by loading the base models into the database, but this would introduce a bias due to differences in completeness of the base models. Another downside of this scenario is that it requires more resources, since a separate observation of two modelers at the same time is required. Figure 2.2 shows an overview of this project scenario. The exclamation mark indicates where the problem of requesting suggestions before a commit by another modeler might arise (not shown in this figure). In that case, suggestions can only be drawn from the domain ontology or base models.

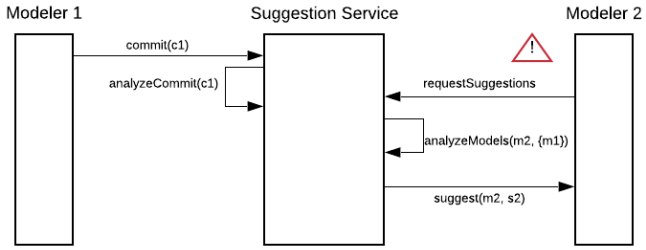

Figure 2.2: Overview of scenario 1. Adapted by author from figure 1 in [1].

- *Scenario 2:* To resolve resource conflict, we could opt to let two modelers work on the same project, but not at the same time. However, this would mean that in every project, only one modeler receives suggestions based on the model from the other modeler. A solution would be to use control groups, where for every language five modelers will receive suggestions based on the other five modelers' models. Figure 2.3 splits the use of the suggestion service in two, because of modelers not working at the same time. It is important to note that modeler 1 receives suggestions based on base model 1 (bm1), but modeler 2 receives suggestions based on model 1 (m1), which is the model created by modeler 1, as model 1 is committed at that time.

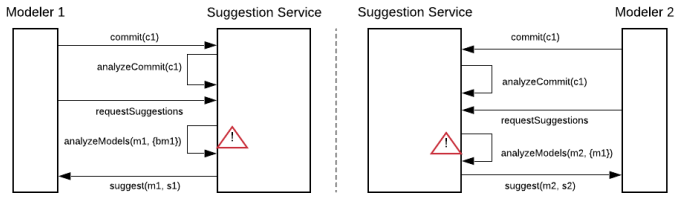

Figure 2.3: Overview of scenario 2. Adapted by author from figure 1 in [1].

- *Scenario 3:* The third scenario would have one modeler working on every project. Although this would be a less convincing simulation of a real-world scenario, it does have the benefit of

allowing the base models to be used as control models. Models created using the suggestion service can be compared to their corresponding base models. This comparison is possible in scenario 1 as well, but since the effectiveness of suggestions in that scenario is dependent on the models in the database (created by other modelers), a bias may arise because of modeler's modeling skills and differences in base model completeness. We will use the same base models for generating suggestions in this scenario, regardless of the modeling language used. By using only the base class diagram (B_{class}) and domain ontology models for suggestion generation, we are able to test differences in effectiveness of suggestions when comparing modeling languages. Figure 2.4 shows an overview of this scenario with two modelers working on different projects. The figure shows one modeler working on a project, receiving suggestions that are based on two base models.

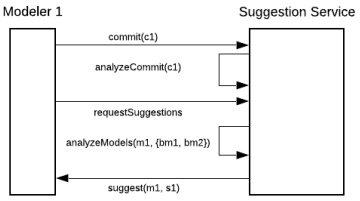

Figure 2.4: Overview of scenario 3. Adapted by author from figure 1 in [1].

The evaluation above is summarized in table 2.2, with a + or – indicating a positive or negative answer to the statements in the first column. Based on this evaluation, we opt for using scenario 3 as the project structure for these experiments. Every participant is assigned a project id, producing a total of 20 projects. Every project includes either the iStar of BPMN model created by the participant, the B_{class} model, and the ontology model. Communication with the service uses a POST request, where the URL is updated in accordance with the project id. This enables the service to separate all projects and their models and results.

Table 2.2: Overview of pros and cons for project structure scenarios

	Scenario 1	**Scenario 2**	**Scenario3**
Accurate simulation of real-world scenario	+	+	–
No need for a control group	+	–	+
Suggestions are directly available	–	+	+
Resources are sufficient	–	+	+
Possibility of both inter- and intralanguage comparisons	–	–	+
Suggestion sources are not biased by differences in modeler skill levels and base model completeness	+	–	+

Initial models

When comparing the differences of the impact of the suggestion service on modeling in iStar and BPMN, we are performing an inter-language (between two modeling languages) comparison. In

order to establish equal affecting variables (variables that affect the outcome of observations) for the experiments for both tools, we should consider informational equivalence of the initial model that is presented (the model that participants can expand on) [14]. For example, without informational equivalence, the initial iStar model might contain more information than the BPMN model, leading to skewed experiment results between both tools. There are two possible solutions to achieve informational equivalence: assessing the informational equivalence of both initial models, or providing no initial model at all. The first option would require a comparison of the base models against the domain ontology [13, 14]. However, the base models are created by expert modelers. Since our experiment involves novice modelers, we argue that providing a base model for the project would undermine the focus of our experiment. What's more, the introduction of base models would limit the amount of modeling that is required of the participants and thereby limit the chances of asking for suggestions, therefore limiting the amount of both quantitative and qualitative data that is to be collected. We therefore choose not to provide an initial model to participants, letting them model from scratch.

2.2.3 Answering the research questions

RQ1: how much does this suggestion service improve model completeness?

An ontology represents domain knowledge in the form of concepts (and relationships between those concepts) [18]. Model completeness indicates the share of concepts in the domain ontology that are covered by the created model. For this analysis, we consider the labels in the created models as noun chunks. Testing the created models for completeness against the domain ontology gives an indication of the model's ability to represent domain knowledge and can therefore be used as a means for model validation [34, 18].

Collecting the data
To answer this research question, we use the base models B_{BPMN} and B_{iStar} as models for the control group and the models created by the participants as the treatment group. It should be noted that we are comparing results of multiple participants in our experiment to data that is created by a single modeler, but we will explain the statistical analysis method later on.

An influence on model completeness is the language that is used for modeling. One language might include more elements that are able to accurately model the domain than another. By evaluating the suggestion service using two languages, we are able to draw conclusions on the inter-language differences of suggestion effectiveness.

A second influence on model completeness is the difference in base model completeness. The B_{iStar} model differs in completeness from the B_{BPMN} model, as it is smaller and therefore a weaker base for suggestions. We choose to eliminate this factor by using only the B_{class} model for suggestion generation. An overview of the uses of the base models can be seen in table 2.3.

In order to answer this research question, we need the following data:

- Lists of concepts included in the B_{iStar} and B_{BPMN} models
- Lists of concepts included in the created iStar (C_{iStar}) and created BPMN (C_{BPMN}) models
- List of concepts included in the domain ontology

Table 2.3: Overview of the uses of base models

	Ontology	$\mathbf{B_{class}}$	$\mathbf{B_{iStar}}$	$\mathbf{B_{BPMN}}$
Comparison material for determining model completeness	X			
Comparison material for determining use of syntactically correct terminology	X			
Comparison material for model alignment		X		
Serving as a base for suggestion generation	X	X		
Serving as a control group for RQ1 and RQ2			X	X

Analyzing the data
In order to test for model completeness, we will use the domain ontology as the baseline for a complete model. A list of concepts will be drawn from the ontology, which is deemed as the "complete" model, and used as comparison material against which lists of concepts from created models can be compared. Before running statistical tests, we will determine the model completeness of the B_{iStar} and B_{BPMN} models (in percentages), and all C_{iStar} and C_{BPMN} models individually. Completeness scores are determined by matching concepts in the created models to the domain ontology using exact matches, as well as substring matches.

The first step of analysis is to determine if, in general, the created models are significantly more complete than the base models. Depending on the normality of distribution of the completeness scores, either a one sample t-test or a Wilcoxon Signed Rank test is used to test for significance. After this first step, the same analysis is done for the C_{iStar} and C_{BPMN} models separately. These tests allow us to check if the completeness scores for the group with suggestions differ significantly from the mean completeness scores of the base models. We hypothesize the mean completeness scores of the base models to illustrate the average completeness scores for models created by experts without suggestions. This hypothesis is necessary, since we are limited by the low number of base models.

If we can establish that for both languages a significantly more complete model is produced when receiving suggestions, we can test differences between the modeling languages. By running a one-way ANOVA with modeling language as the independent variable, we can establish if model completeness significantly differs between modeling languages.

RQ2: how much does this suggestion service help in using the syntactically correct terminology?

The suggestion service aims to let modelers use a common vocabulary by searching for noun chunks that are missing in the created model, but present in the project (the domain ontology and all the related models) [1]. We argue that an improvement in syntactic relatedness between models allows modeler to use a common vocabulary, leading to a decrease in ambiguity in models. In the research by Aydemir & Dalpiaz, noun chunks (seen as strings) were considered missing if: there was no exact match within the project, the string was not a substring or superstring of a noun chunk in the project, or if there was a noun chunk in the project that had no similar or related noun chunk in the created model. The last two heuristics (similarity and relatedness) focus on semantics, whereas this research question looks at syntactic correctness. These heuristics

will therefore not be used to determine syntactic correctness. The substring match heuristic will only be used for suggestion generation, so the only heuristic also used for determining syntactic correctness is the exact match.

Collecting the data
To answer this research question, we compare the noun chunks in the created models to the noun chunks in the domain ontology. Noun chunks are determined syntactically correct if they are an exact match with a noun chunk in the domain ontology.

In order to answer this research question, we need the following data:

- Lists of concepts included in the B_{iStar} and B_{BPMN} models.
- Lists of concepts included in the created iStar (C_{iStar}) and created BPMN (C_{BPMN}) models.
- A list of concepts included in the domain ontology.

Analyzing the data
It is important to note that, for every created model, there is a possibility that only a part of the concepts that are modeled are actually present in the ontology. The share of these concepts is indicated using the vertical lines in figure 2.5, where "DO" indicates domain ontology and "CM" indicates created models. The horizontal lines indicate the concepts that are present in the created model, but not in the ontology. Since we cannot make assumptions about the syntactic correctness of concepts that are not in the domain ontology, we will only run tests for syntactic correctness on concepts that are both in the created model and the domain ontology.

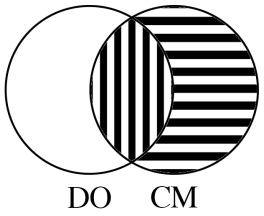

Figure 2.5: Venn diagram illustrating the data collection for RQ2

To find that share of common concepts, we cannot only use the exact match heuristic. We define syntactic correctness as an exact match, so using only that heuristic to define the common concepts would automatically result in a 100% syntactic correctness score. The first step in this analysis is to find the common concepts by applying a substring match. After this match is applied to find the common concepts, the exact match is used to define the share of those concepts that use syntactically correct terminology. The analysis of this research question therefore follows the following steps:

- The domain ontology concept list is split into nouns, creating an array containing every noun
- We iterate over that array and keep only the elements (nouns) that are also present in the model that is used for comparison

We now have an array containing all common concepts/nouns between the domain ontology and the comparison model.

- The domain ontology concept list is split, but this time into noun chunks

  Where the first concept list split resulted in an array containing, for example, the nouns "first" and "supervisor", this split results in an array containing "first supervisor".

- We iterate over the array containing the common concepts and keep only the elements that are also present in the array containing the noun chunks from the domain ontology

  Every noun chunk from the ontology containing more than 1 noun is automatically deleted, since the array containing common concepts includes only single nouns. This is not a problem, however, since we do not define syntactic correctness as a substring match. Therefore, a concept like "supervisor" should be regarded as syntactically incorrect when compared to "first supervisor".

- We filter the common concepts array to only include unique elements
- The resulting number of elements in the array is compared to the number of unique elements in the common concepts array, resulting in a syntax score

By calculating the percentage of concepts in the list of common concepts that are syntactically correct, we can draw conclusions on the influence of the suggestion service on syntactic correctness. Similar to RQ1, we will first determine if the C_{iStar} and C_{BPMN} models are significantly more syntactically correct than the B_{iStar} and B_{BPMN} models. Again, we will run a one-sample t-test or a Wilcoxon Signed Rank test (depending on the normality of distribution) for both languages, first determining if the created models are significantly more syntactically correct in general. The same tests are then run to determine significance for both languages separately.

A one-way ANOVA with syntactic correctness as the dependent variable and modeling language as the independent variable, will also be run to determine if the syntax scores for one language differ significantly from those of the other language.

RQ3: how much does this suggestion service improve model alignment?

Aligning models takes away inconsistencies and ambiguity that may otherwise arise in large projects. Both RQ1 and RQ2 also focused on alignment, but on the alignment of the created models with the domain ontology. For this research question, we aim to determine the impact of the suggestion service on aligning the created models.

Collecting the data

To test the alignment of models, it is important to first define what model alignment is. Various papers mention model alignment alongside "relatedness" and "similarity" [1, 11, 18, 39]. In that sense, we can see a relation with the relatedness and similarity heuristics mentioned previously. We argue that exact matches and substring matches also increase model alignment. However, since suggestions were only generated using the substring and exact match heuristics for this research, we will also only focus on these heuristics to determine model alignment.

Data collection for this research question is similar to the data collection for RQ1 and RQ2; the difference is mostly in analysis. In order to answer this research question, we need the following data:

- Lists of concepts included in the the B_{iStar} and B_{BPMN} models
- Lists of concepts included in the C_{iStar} and C_{BPMN} models
- List of concepts included in the B_{class} diagram

Analyzing the data
To test model alignment, there are two possible focus points: inter-language alignment and intra-language alignment. The first focus point looks at aligning models of different languages (BPMN, iStar and class diagram in our case), whereas the second looks at aligning models of a single language.

In order to test the impact of the suggestion service on intra-language alignment, we would need to establish the model alignment for the base models. However, since our experiment includes only one B_{iStar} and one B_{BPMN} model, we do not have any comparison material to test model alignment (we do not have another base model for those languages). Although we could still make assumptions about the alignment of the created models, without a baseline we cannot make judgments about the impact of the suggestion service on model alignment. We will therefore only focus on inter-language alignment.

Again, the first step towards analyzing inter-language alignment is to establish a baseline by determining the alignment of the B_{iStar} and B_{BPMN} models with the B_{class} diagram. The following two examples explain different ways of establishing the baseline:

- Suppose we have a BPMN model with 62 concepts and a class diagram with 43 concepts. When comparing the two models using the right heuristics, we find 37 concept matches. We would have 25 unmatched concepts for BPMN and 6 unmatched concepts for the class diagram, resulting in a total of 31 unmatched concepts. The resulting percentage used to define model alignment between the two models is approximately 54% (37 matches out of a total of 68 matched + unmatched).
- Suppose again we have the same models and matches as stated above. We could also determine model alignment based on the total amount of concepts in either model. That would result in a model alignment of approximately 60% for BPMN and approximately 86% for the class diagram.

Using the second approach, alignment for one model is not affected by unmatched concepts in other models. However, we argue that since alignment is always dependent on multiple models, it is unavoidable that alignment is also affected by multiple models. We therefore opt to use the first method for establishing alignment scores. It is important to note that we define alignment only based on the alignment with the class diagram. This allows us to use the same comparison base for both languages, removing a possible bias that may otherwise exist due to differing model completeness between BPMN and iStar. The analysis follows the following steps:

- The concept list for the class diagram is split into nouns, creating an array containing every noun
- The same is done for the concept list of the comparison model
- We iterate over the array containing the elements (nouns) of the class diagram, keeping only the elements that are also present in the comparison model

  We now have an array containing all nouns that are present in both models.

- The amount of unmatched concepts from both models are calculated and combined
- The alignment score is calculated as the share of matched concepts (by taking the length of the previously mentioned array), compared to the total amount of matched and unmatched concepts

Once we have established the alignment of both the B_{iStar} and B_{BPMN} models with the B_{class} diagram, we can run tests on the alignment of the created models with the B_{class} diagram.

We will first determine if, in general, the created models are significantly more aligned than the base models. Depending on the normality of distribution, we will use a one sample t-test or a Wilcoxon Signed Rank test to test for significance. The same analysis will be done for both languages separately.

Again, a one-way ANOVA with model alignment as the dependent variable, and modeling language as the independent variable, is run to determine if the alignment scores for one language differ significantly from the alignment scores for the other language.

RQ4: how does the modeler-service interaction for this suggestion service affect its effectiveness?

Like many other systems, the effectiveness of this suggestion service is dependent on the way it is used. Modeler-service interaction for this research can take several forms, which requires several forms of data collection. To answer this research question, we will collect qualitative data from observations and quantitative data from observations, models, and service interaction. This research question also includes RQ1, RQ2 and RQ3, as we define service effectiveness by its ability to improve model completeness, alignment, and the use of syntactically correct terminology.

Collecting the data

Qualitative data

We use a combination of participant observation and interviews to collect qualitative data about the use of the suggestion service. Participant observation does not require the observer to engage in the activity being observed, but the participants should be aware of being observed. Data collection should be done as unobtrusively as possible to induce "normal" behaviour of participants. [33]. Several difficulties arise with participant observation in this experiment:

- Participants might not be aware as to why they do or do not perform certain actions. Asking questions during the observation might help to identify problems that the participant is not aware of [36].
- Not all processes are visible to the observer as they are happening inside the head of participants. Examining thought processes of participants is possible through questions (as the previous point suggests), but also through methods such as the think aloud protocol [33].
- In order to test the effectiveness of suggestions on creating models, it is crucial that participants actually request suggestions. However, steering participant actions does conflict

with unobtrusive observation. We aim to minimally steer the observations, but only when necessary.

Combining the observation data described above with an interview is a form of research triangulation [35, 38]. Interviews can help identify hidden problems [36], but can also aid the comparison of participant views after experimentation to participant actions during experimentation [35]. This combination of qualitative and quantitative data collection can expose relationships that are otherwise hidden [12].

To aid and structure our observations, an observation protocol is created. The observation protocol, based on [3], can be found in appendix A. This protocol is used to collect data (such as domain familiarity and comments), but also to structure the observation with a script. With the script, we aim to give just the right amount of information to the participants so that they understand the goal of the experiment, but are not biased by our comments [33].

Quantitative data

Not only comments and interview answers (qualitative data) are collected during experiments, as quantitative data also provides interesting insights into modeler-service interaction. Quantitative data often also has the capacity to convey implicit thought processes of the participants. For example, a participant might express their enthusiasm about the usefulness of the suggestion service, but might request suggestions relatively infrequently. Quantitative data helps us to support and explain results from observer data and the created models (and therefore aids the explanation of the results from RQ1, RQ2 and RQ3). We will collect the following data:

- How many times the service is called
- The amount of suggestions received
- The proportion of suggestions that is rated useful
- The proportion of suggestions that is rated not useful
- The proportion of suggestions that is rated useful, but not modeled

There is no need to collect the data manually, as the suggestion service will keep a log file of this data.

Analyzing the data

Data analysis for this research question differs from the other research questions, since it also includes qualitative data. The qualitative data (observation and interview comments) are used to provide explanations for results and to illustrate the use of the service. Comments that are collected during experiments can be compared to interview comments and actions, determining if these answers and actions match implicit impressions that arise during experiments [35]. For example, because we ask our participants to think aloud, one participant might express having a difficult time trying to determine the appropriate concepts for the model. We might expect the participant to then request suggestions, but if not, it might give insights into the perceived usefulness of the service for the participants.

Although the qualitative data will not provide answers for measuring effectiveness of the service, it does help in understanding the use of the service and therefore supports the quantitative data.

The quantitative data is more easily used to determine service effectiveness. In the previous three research questions, we have measured model alignment, model completeness, and the use of syntactically correct terminology for every created model. For every model, we are able to compare these measures to the quantitative data to determine the effectiveness of the suggestion service. For example, more frequent suggestion requests might produce more aligned models.

We are interested in seeing if model alignment, model completeness, and the use of syntactically correct terminology is dependent on the quantitative measures. By testing all scores of RQ1, RQ2, and RQ3, to the quantitative measures for correlation, we are able to determine to what degree each measure influences the other measures.

Chapter 3

Results

The experiments were conducted over the course of eight days, resulting in a total of 16 created models (8 for both languages). Out of the 20 initially scheduled experiments, two were cancelled due to scheduling and timing conflicts. The first two conducted experiments were used to test the experiment protocol. Since changes were made to the protocol consequently, these two experiments were excluded from analysis.

This chapter answers the research questions in separate sections, based on both quantitative and qualitative data collected during and after the experiments.

3.1 Research Question 1

This section aims to answer the following research question: *"How much does this suggestion service improve model completeness?"*. The analysis uses lists of concepts from the domain ontology, the base models, and the created models.

3.1.1 Extracting the data

The ontology model and base models were available as CSV files, from which the text elements were extracted manually. The created models were available as JSON and XML files, for which the appropriate nodes were filtered out.

The resulting text file with the concepts in the domain ontology included 130 concepts (including synonyms) as noun chunks. Splitting this text file for every noun resulted in a text file with 241 nouns. We did not filter out duplicate nouns. Splitting the concept list into nouns and testing for completeness by comparing nouns, rather than noun chunks, is a substring match. This way, we still find a match for a model that includes a concept like "thesis", rather than "thesis project" (which might be listed in the domain).

To calculate the percentage of the domain that is covered by the created models, a script is written in JavaScript that takes the ontology text file (with 241 concepts/nouns) and the model text files as input strings. We view the ontology noun list as a complete list, so we iterate over the nouns in this string to find how many are covered by the model string. There are 241 nouns

that can be covered by the comparison string, so if a model covers 120 of those concepts, 49.79% of the domain is covered (rounded off to two decimals).

3.1.2 Completeness scores

Completeness scores were calculated for the B_{iStar} model, the B_{BPMN} model, and all created models. The results are listed in table 3.1.

Table 3.1: Completeness scores for all models

Model	Modeling language	Amount of nouns covered	% of domain covered
B_{BPMN}	BPMN	90	37.34
B_{iStar}	iStar	80	33.20
4	BPMN	102	42.32
5	iStar	50	20.75
6	iStar	123	51.04
7	iStar	121	50.21
8	BPMN	106	43.98
9	iStar	113	46.89
10	iStar	94	39.00
11	iStar	133	55.19
12	iStar	112	46.47
13	BPMN	79	32.78
14	iStar	115	47.72
15	BPMN	125	51.87
16	BPMN	128	53.11
17	BPMN	110	45.64
18	BPMN	101	41.91
19	BPMN	119	49.38

By running a Shapiro-Wilk test for normality on the completeness scores of the created models, we can conclude that these scores are normally distributed (p=.024). This allows us to run a one sample t-test to determine if the mean completeness score for the created models is significantly higher than the mean completeness score of the base models. When running a one sample t-test of the created models completeness scores against the mean base model completeness score (35.27%), we find that the completeness scores of the created models (M=44.89, SD=8.58, N=16) are significantly higher; t(15)=4.49, p=.000. This indicates that, without considering modeling language, the suggestion service allowed participants to create a model with a significantly higher completeness score than the average base model. However, as we are comparing base models of a single expert (one for each model) to created models of multiple non-experts, we cannot conclusively state that modelers using the suggestion service create more complete models. After all, there might be differences in completeness scores influenced by other factors, such as differences in motivation and expertise. We therefore limit our conclusion to stating that non-expert modelers using the suggestion service create models that are at least as complete as models created by expert modelers that do not use the suggestion service.

Independently, the completeness scores for the modeling languages are not normally distributed. A Wilcoxon Signed Rank test allows us to compare mean completeness scores for both languages against the respective completeness score of the base model. For iStar, we find that the completeness scores of the created models are significantly higher than the completeness score for

the B_{iStar} model; Z=34, p=.025. For BPMN, we also find that the completeness scores of the created models are significantly higher than the completeness score for the B_{BPMN} model; Z=35, p=.017.

Running a one-way ANOVA with "modeling language" as the independent variable shows a significant difference in model completeness for the different modeling languages: $F(1,14)$=6.157, p=.026. Since we only have two group types, a post-hoc test is not required. We find that the completeness scores for the C_{BPMN} models (M=49.48) are significantly higher than the completeness scores for the C_{iStar} models (M=40.30).

3.2 Research Question 2

This section aims to answer the following research question: *"How much does this suggestion service help in using the syntactically correct terminology?"*. Similar to research question 1, we use lists of concepts from the domain ontology, the base models, and the created models for this analysis.

3.2.1 Syntactically correct common concepts

The lists of concepts were available from the analysis of RQ1. As described in the previous chapter, we define syntactic correctness as the share of common concepts between the ontology and created model that use correct terminology (as defined by the ontology). By comparing nouns and noun chunks, our resulting syntax scores indicate the noun "chunks" that are present in both the ontology and created models, and use the same terminology. The syntax scores are listed in table 3.2.

Table 3.2: Share of syntactically correct common concepts

Model	**Modeling language**	**% of syntactically correct common concepts**
B_{BPMN}	BPMN	40.74
B_{iStar}	iStar	52.94
4	BPMN	50.00
5	iStar	60.00
6	iStar	31.25
7	iStar	32.35
8	BPMN	48.15
9	iStar	43.33
10	iStar	33.33
11	iStar	35.14
12	iStar	31.03
13	BPMN	30.43
14	iStar	27.27
15	BPMN	38.89
16	BPMN	31.58
17	BPMN	35.71
18	BPMN	37.93
19	BPMN	34.29

The syntax scores are tested for normality and follow a normal distribution (p=.020). We therefore use a one sample t-test to determine significance. The syntax scores of the created models are compared to the mean syntax score of the base models (M=46.84). We find that the syntax scores of the created models (M=37.54, SD=8.74, N=16) are significantly lower than the mean syntax score of the base models; t(15)=-4.256, p=.001.

Next we look at differences in modeling languages. We find that for iStar, the syntax scores of the created models (M=36.71, SD=10.49, N=8) are significantly lower than the syntax score of the B_{iStar} model (M=52.94); t(7)=-4.375, p=.003. For BPMN, the syntax scores of the created models (M=36.82, N=8) are not significantly lower than the syntax score of the B_{BPMN} model; Z=12, p=.401. When running a one-way ANOVA, we also do not find a significant difference in syntax scores between modeling languages: $F(1,14)$=.641, p=.437.

We conclude that non-expert modelers using the suggestion service do not create models that are at least as significantly correct as models created by experts without the suggestion service. For iStar and the general scores, the syntax scores of the created models were significantly lower than those of the base models. For BPMN alone, the syntax scores of the created models were not significantly lower than the syntax score of the B_{BPMN} model.

3.3 Research Question 3

This section aims to answer the following research question: *"How much does this suggestion service improve model alignment?"*. We use the concept lists of the class diagram, the base models, and all created models.

3.3.1 Measuring model alignment

Model alignment is defined as the share of matched concepts (between the class diagram and the comparison model) when looking at the total amount of concepts. Since we use a substring match for this analysis, comparison is again done using concept lists including only nouns, rather than noun chunks. The complete approach to measuring model alignment is described in the previous chapter. The resulting alignment scores are listed in table 3.3.

The overall alignments scores are tested for normality of distribution, but do not follow a normal distribution. We will therefore use a Wilcoxon Signed Rank test to determine if the mean alignment scores of the created models differ significantly from the mean alignment scores of the created models. We find that the alignment scores of the created models (M=19.27, N=16) are not significantly higher than the alignment score of the base models (M=16.81); Z=83, p=.438.

When looking at both languages separately, we also do not find a significant difference. The alignment scores for iStar (M=19.31, N=8) are not significantly higher than the alignment score of the B_{iStar} model; Z=15.00, p=.674. The alignment scores for BPMN (M=19.02, N=8) are also not significantly higher than the alignment score of the B_{BPMN} model; Z=29.00, p=.123.

Running a one-way ANOVA shows a significant difference in the alignment scores for the BPMN and iStar groups: $F(1,14)$=21.008, p=.000. We see that the C_{iStar} models have significantly lower alignment scores (M=14.06) than the C_{BPMN} models (M=20.60). Even though the alignment scores of the created models do not significantly differ from the base models, the suggestions seem to have the most impact on alignment for the BPMN models.

Table 3.3: Alignment scores

Model	Modeling language	Alignment to class diagram in %
B_{BPMN}	BPMN	14.89
B_{iStar}	iStar	18.73
4	BPMN	14.67
5	iStar	8.57
6	iStar	21.75
7	iStar	21.00
8	BPMN	18.60
9	iStar	19.51
10	iStar	11.07
11	iStar	22.61
12	iStar	14.42
13	BPMN	10.29
14	iStar	19.11
15	BPMN	19.44
16	BPMN	19.47
17	BPMN	19.61
18	BPMN	21.41
19	BPMN	15.73

We conclude that there is not a significant difference in alignment scores of models created by non-experts using the suggestion service compared to experts not using the suggestion service. Therefore, we cannot conclusively say whether or not the non-expert models are at least as aligned correct as the expert models.

3.4 Research Question 4

This section aims to answer the following research question: *"How does the modeler-service interaction for this suggestion service affect its effectiveness?"*. As is described in the previous chapter, we use a combination of qualitative and quantitative data to find an answer to this question. The qualitative data is comprised of observation notes and interview responses and will be analyzed in the following section. We use these results together with quantitative data collected during experiments, and the results of RQ1, RQ2 and RQ3, to provide explanations for the results for these research questions.

3.4.1 Qualitative Data

Coding references

Looking at the codebook in appendix D, we are able to view the nodes used for coding the qualitative data, along with the reference counts for those nodes. It should be noted that this codebook does not include the nodes containing information about interview questions, since the reference count for those nodes is irrelevant; we are only interested in the qualitative information in those references. A summary of the primary points of interest from the codebook are listed in table 3.4.

Table 3.4: Summary of codebook in appendix D

Node	Reference count
Observer Request	32
Own Motivation	93
Attitude: Negative	79
Attitude: Confused or Unsure	60
Attitude: Positive	58
Attitude: Neutral	36
Domain: Suggestion Check	2
Suggestions: Looking at Domain	53
Domain: Inclusion in Model	59
Suggestions: Inclusion in Model	49

The first point of interest in this codebook is the distinction in motivation for requesting suggestions. We can see that 32 instances are coded where the observer asks participants to request suggestions, compared to 93 instances where participants request suggestions out of own motivation. Although this would indicate personal motivation is the main driver for requesting suggestions, we should be careful with that statement. The observer asked participants to request suggestions at most once every 10 minutes, allowing for a total of 96 observer requests over 16 experiments. This means that one third of the possible observer requests were necessary. Requests out of own motivation could be made as often as participants liked.

The second point of interest is the division of attitude ratings concerning the suggestion service. Participant attitudes are coded primarily as negative (79), followed by confused or unsure (60), positive (58) or neutral (36).

We can also see that the primary action of participants after requesting suggestions, without looking at feedback, is to check the domain (53 instances), whereas the suggestions are not often used to check the content of the domain (2 instances). Including elements into the model is mostly done following the domain (59) compared to the suggestions (49), but this might also be explained due to the extensiveness of the domain and the participants referring to the domain more often than the suggestions.

Word count

In the previous subsection, the reference count was used to determine instances for various nodes. Although this enables us to order the presence of instances, it does not tell us about the content of those instances. In this subsection, instances in various nodes will be analyzed to determine the driving factors for those nodes.

For this analysis, we run a word count query on various nodes, resulting in a sorted list of the most frequent words coded in that node. The list is filtered, leaving out words that are not relevant for analysis, and related words are combined. The query also includes stemmed words for finding matches. When combining words that are related, we make sure to only include unique instances for the word count. This way, an instance stating *"I am checking missing elements"* only adds to the word count once when combining the words "check" and "missing". The tables in the following section only list word combinations that are deemed useful for analysis and reporting and is therefore not a complete list of word counts in the nodes. It should also be noted that these word counts only include comments and notes made during the experiments.

Requesting out of own motivation
Table 3.5 lists the combined word counts for various words coded in the "Own Motivation" node. Looking at the presence of various words in this node allows us to find the main motivation of participants to request suggestions. We see that the most frequent combination is "Useful + Included", indicating that participants often requested suggestions to find useful concepts to include in their model. The suggestions were also often used as a check system and to get inspiration. Motivations out of curiosity and domain alignment were less common.

Table 3.5: Overview of word counts for the "Own Motivation" node

Words	**Explanation**	**Word count**
Useful + Included	Indicates checking suggestions for useful concepts to include	21
Check + Missing	Indicates checking for concepts that participants have missed in the domain	18
Inspiration + Ideas + Stuck + Know	Indicates checking suggestions for inspiration when participants are stuck	11
Curious + Wondering + Know	Indicates checking suggestions out of curiosity	8
Domain + Know	Indicates checking suggestions to align the model with the domain	4

Attitudes towards suggestions
In this subsection, the attitudes of participants towards the suggestion service are analyzed. Participant attitude is measured as mainly negative, but for being successfully able to implement this service, a more positive attitude is desirable. By analysing the attitudes, therefore, we are able to guide future development of the suggestion service.

Table 3.6: Overview of word counts for the "Negative" attitude node

Words	**Explanation**	**Word count**
Later + Now + Yet + Moment + Read	Provided suggestions were not useful at the moment of requesting	18
Already Included + Already Modeled	Suggestions were already included in the model	16
Thesi + oeni + Weird + Stupid [...]	Indicates "weird" suggestions that were not useful for participants	11
Relevant + Superfluous	Suggestions were found to be not relevant	6
Generic	Suggestions were found to be too generic to include	5
Previous + Similar	Provided suggestions were too similar to what had already been suggested	4

Table 3.6 states that the primary motivator for negative attitudes is the provision of suggestion at a too early stage of modeling. The second motivator, on the other hand, states that late provision of suggestions also induces negative attitudes. The top two motivators for negative attitudes are therefore both based around timing difficulties of the suggestions. This is a useful point of improvement for future development.

Participants were also influenced negatively by "weird" suggestions (such as the incorrect stemming of the word "thesis"). The final three motivators for negative attitudes are related to

the service providing suggestions that were deemed too similar to previous suggestions, generic, and irrelevant. For future development, we should look at fine-tuning the suggestion generation algorithms.

Table 3.7: Overview of word counts for the "Positive" attitude node

Words	**Explanation**	**Word count**
Various	Instances where participants expressed positivity about suggestions not yet included in the model	34
Domain + Website	Participants expressed positivity after finding suggestions in the domain	11
"Already Included"	Positivity expressed for suggestions that were already included	9
Various	Instances where terminology was changed based on suggestions	4

Table 3.7 lists motivators for positive attitudes. We find that participants mostly react positively to suggestions that are not yet included in the model (34 instances), although positivity was also expressed about suggestions that were included already (9 instances). We also find that participants show positivity after finding the provided suggestions in the domain, indicating that they may rely on the domain to "validate" the suggestions. There were four instances where participants expressed positivity for being able to change the terminology of elements already in the model based on the suggestions.

Motivators for "Neutral" and "Confused or Unsure" attitudes will not be analyzed, since these are often induced by a combination of motivators for positive and negative attitudes.

Interviews

After completing the modeling task, participants were asked some questions regarding the experiment in a short interview. This interview served to compare participant views to actions. The interview responses are analyzed in this subsection.

Table 3.8: Interview responses to motivations for (not) requesting suggestions

Question	**Primary instances**	**Reference count**
Motivations for requesting suggestions	**Using the suggestions as a general check**	**12**
	Checking for elements that had not been included yet	**12**
	Getting inspiration when stuck	8
	Checking the terminology of elements	2
Motivations for not requesting suggestions	**Needed the domain for information and structure**	**9**
	Participants wanted to execute own ideas first	4
	The domain was trusted more	3
	The suggestions did not provide enough information	3

The first two questions in the interviews following the modeling tasks regarded the motivations of participants for requesting or not requesting suggestions. The latter question focused on conscious or unconscious decisions to get information from sources other than the suggestions, such as the domain or own experience. The main motivations are extracted using a manual check of the interview responses and are listed in table 3.8.

We see that participants mostly request suggestions to use them as a general check, of which a check for excluded elements is the most prevalent one. Other checks include checking terminology of elements and, however rarely, checking the content of the domain. When comparing these answers to the observation comments listed in table 3.5, we see that in both cases the main motivation is listed as using suggestions for checking what the missing elements in the model are. However, it is noticeable that, during the interviews, participants were less likely to mention requesting suggestions out of curiosity or to help them when they were stuck compared to the observation comments.

The main motivation for not requesting suggestions is the lack of structure that the suggestions provided. Suggestions were found to be useful, but only if the participants already knew which part of the domain to model and when. As one participant mentioned: *"I often did not know where and how to include suggestions in the model"*. This lack of structure might explain why participants primarily looked at the domain after receiving suggestions, as we can see in appendix D.

Table 3.9: Interview responses on model creation

Question	**Primary instances**	**Reference count**
Attitudes towards the usefulness of suggestions	**The suggestions were provided at the wrong time**	**5**
	The suggestions pointed towards needed elements	4
	The suggestions did not provide enough information	3
Views on creating a better model using suggestions	Positive	1
	Somewhat positive	**7**
	Neutral	3
	Somewhat negative	0
	Negative	5
Views on using the service to improve collaboration among modelers	**Positive**	**10**
	Neutral	3
	Negative	1

Table 3.9 lists the interview responses concerning model creation. We find that the primary attitude towards the usefulness of suggestions is negative, caused by wrong timing of the suggestions. This aligns with the observation comments in table 3.6, where the top two motivators for negative attitude were also related to timing issues of the suggestions. The positive attitude relating to the service pointing towards needed elements is similar to the main motivation for requesting suggestions expressed during the observations, as is listed in table 3.5.

Furthermore, participants responded somewhat positive when asked if they created a better model with the suggestions than without, and were generally positive on the improvement of collaboration with other modelers.

Group differences

In the previous subsections, we compared instances for various nodes in order to analyse global results. In this section, we will look at differences in results among various groups. Groups are divided by modeling language, modeling confidence, and domain familiarity.

Table 3.10: Group differences for attitudes towards suggestions

	iStar	**BPMN**	**C2**	**C3**	**C4**	**C5**	**F1**	**F2**	**F3**	**F4**	**F5**
Negative	47	32	**10**	35	26	8	8	30	26	2	13
Positive	31	27	**4**	24	24	6	6	25	17	1	9

Table 3.10 lists the total coding references for the "Positive Attitude" and "Negative Attitude" nodes for all groups. The groups in the table indicate modeling language, modeling confidence (C2 to C5), and domain familiarity (F1 to F5). No participant mentioned having very low confidence in model creation (C1), so that group is not included.

We find that, for all groups, the main attitude towards the suggestions is negative. The division of attitudes within group types (such as modeling language) also does not differ significantly. The only outlying result can be found in C2, where approximately 71% of the instances indicates a negative attitude. However, this result is more significantly influenced by the small sample size in this group. We therefore conclude that attitudes towards suggestions are not influenced by group attributes.

Table 3.11: Group differences for actions concerning suggestions

	iStar	**BPMN**	**C2**	**C3**	**C4**	**C5**	**F1**	**F2**	**F3**	**F4**	**F5**
Suggestions Included	35	24	3	27	23	6	7	18	19	3	12
Domain Included	26	23	2	26	21	0	3	26	6	3	8
Suggestions Included	57%	51%	60%	51%	52%	**100%**	70%	**40%**	**76%**	50%	60%
Not Useful	57	36	8	39	36	10	4	41	31	2	15
Useful	29	28	4	22	24	7	5	28	15	1	8
Not Useful	**66%**	**56%**	66%	63%	60%	58%	**44%**	**59%**	67%	66%	65%

Table 3.11 lists group differences for the inclusion of elements in the model and the provided comments on usefulness of the suggestions.

The first row lists the instances where suggestions were included in the model right away, whereas the second row lists instances where elements were included after consolidating the domain. The third row lists the share of instances where suggestions were included. When looking at differences between modeling languages, we do not find a significant difference. We find that, for the group with the highest level of modeling confidence, 100% of the instances on element inclusion is based on suggestions. This result is significantly higher than the results of the lower confidence groups. A possible explanation might be provided when looking at table 3.8, where the main motivation for not requesting suggestions is stated as needing the domain for information and structure. A higher confidence level for modeling decreases the need to rely on the domain for

structure, allowing for suggestions to be included quicker. However, this result might also be heavily influenced by the smaller group size. When looking at domain familiarity, we find a significant increase in instances where suggestions are included from F2 to F3. An increase in domain familiarity again decreases the need to rely on the domain, allowing for faster inclusion of suggestions. However, this does not explain the decrease in suggestion inclusion rates after F3. We therefore cannot state that there is a significant difference in actions concerning element inclusion between group types.

The bottom rows in the table list instances where comments were made on the usefulness of the suggestions. When looking at the modeling languages, we find that both groups commented primarily negatively on the usefulness, with a higher level for the iStar group. This difference might be influenced by a variety of factors, such as the ability of iStar to adequately capture the domain, and the usefulness of the suggesting of nouns for the iStar structure. We find another noticeable difference when looking at the perceived usefulness of suggestions by comparing groups on domain familiarity. The perceived usefulness decreases for the groups with higher domain familiarity. We see the largest decrease in perceived usefulness between F1 and F2, even though the F1 group noted the larger share of included suggestions. This result might indicate a contradiction in sentiments and actions for participants. However, when looking at instance counts, we do not find the group differences to be significant enough to determine an influence by group types.

3.4.2 Quantitative data

Table 3.12 lists, for every experiment, the scores of RQ1, RQ2, RQ3 (completeness, syntax, and alignment, respectively), the amount of suggestions received, the percentage of suggestions rated useful or not useful, and the percentage of suggestions that were rated not useful but still modeled. The last score is calculated by comparing the suggestions rated not useful to the concept lists of the created models. We use an exact match, by comparing the suggestions as noun chunks.

Table 3.12: Quantitative data for RQ4

Experiment	RQ1	RQ2	RQ3	Amount	% useful	% not useful	% still modeled
4	42.32	50.00	14.67	80	18.75	80.00	9.38
5	20.75	60.00	8.57	38	31.58	68.42	7.69
6	51.04	31.25	21.75	16	31.25	50.00	25.00
7	50.21	32.35	21.00	45	28.89	71.11	12.50
8	43.98	48.15	18.60	74	22.97	77.03	19.30
9	46.89	43.33	19.51	80	7.50	86.25	14.49
10	39.00	33.33	11.07	33	21.21	66.67	9.09
11	55.19	35.14	22.61	34	14.71	41.18	0.00
12	46.47	31.03	14.42	39	23.08	23.08	22.22
13	32.78	30.43	10.29	23	8.70	21.74	20.00
14	47.72	27.27	19.11	58	1.72	62.07	8.33
15	51.87	38.89	19.44	24	20.83	62.50	20.00
16	53.11	31.58	19.47	36	13.89	13.89	0.00
17	45.64	35.71	19.61	20	25.00	70.00	7.14
18	41.91	37.93	21.41	39	5.13	10.26	25.00
19	49.38	34.29	15.73	29	0.00	0.00	0.00

The first step of analysis is to check for a correlation between the scores in table 3.12. A Pearson correlation test is run, resulting in the correlation scores listed in table 3.13. In this table, an asterix indicates a significant correlation at the .05 level, whereas a double asterix indicates a significant correlation at the .01 level. We are primarily interested in the correlation between the measures relating to suggestions, compared to the scores for RQ1, RQ2, and RQ3. The significance of the correlation scores between these pairs of measurements are highlighted in gray.

Table 3.13: Correlation of results for RQ4

		RQ1	**RQ2**	**RQ3**	**Amount**	**% useful**	**% not useful**
RQ1	Correlation	1	-.577*	.820**	-.037	-.187	-.144
	Sig.		.019	.000	.892	.489	.595
RQ2	Correlation	-.577*	1	-.339	.455	.321	.470
	Sig.	.019		.199	.077	.225	.066
RQ3	Correlation	.820**	-.339	1	.034	-.098	.006
	Sig.	.000	.199		.900	.718	.984
Amount	Correlation	-.037	.455	.034	1	-.186	.511**
	Sig.	.892	.077	.900		.490	.043
% useful	Correlation	-.187	.321	-.098	-.186	1	.469
	Sig.	.489	.225	.718	.490		.067
% not useful	Correlation	-.144	.470	.006	.511*	.469	1
	Sig.	.595	.066	.984	.043	.067	

We see that there is no significant correlation between the amount of suggestions received, the feedback, and the scores for completeness, syntax, and alignment. For the measures relating to suggestions, we only find a significant correlation between the amount of suggestions received and the percentage of suggestions rated not useful (p=.043). This result might indicate that participants become increasingly less positive towards suggestions as they request more.

Separating the correlation results for BPMN and iStar, we find a significant correlation for BPMN between the measures of amount of suggestions and syntax scores: r=.712, p=.048. We find that a higher amount of suggestions received indicates a more syntactically correct BPMN model. This relates to our finding for research question 2, where we stated that the suggestions did not significantly decrease syntax scores for only the BPMN models. We conclude that, for BPMN, more suggestions are required to create a syntactically correct model, but that modelers using the service do not produce a significantly more correct model than modelers that do not use the service.

Chapter 4

Discussion

The results in the previous chapter provide insights into the effectiveness of our tools and service. In this chapter, we will look at the implications of our results, compare these results to other research with a related work section, and provide suggestions for future research.

4.1 Related work

4.1.1 Natural language processing and conceptual modeling

There is room for further research into the application of natural language processing in conceptual modeling. Robeer et al. propose a method for automatically extracting conceptual models from use cases using NLP [30]. They underline the conflict between the comprehensibility of natural language (NL) and its issue of ambiguity. Although natural language is oftentimes used for communicating requirements, it does not benefit from the same restrictions and formality as conceptual models. In this research, the extraction of conceptual models relied on the semi-structured nature of use cases. Although the research yielded some promising results on the possibility of using NLP for conceptual model extraction, there were still some issues related to the difficulty of analyzing verbs, conjunctions, and other textual elements in NLP, and the format of the use cases. This challenge in NLP is also present in our research, as participants often expressed confusion about "weird" suggestions, that perhaps should not have been suggested. Although this solution creates conceptual models with limited human interference, the output is also limited as it creates a single type of model (UML). However, we can view part of this research as being complementary to ours. Our solution does not completely automate domain analysis, leaving room for modeler interpretation, and it only guides the model creation process. Combining a more automated domain analysis (taking requirement documents as input) with a project domain analysis (taking models as input) might effectively minimize human interference in the model creation process.

Similar to this research, Overmyer et al. propose a solution that gives linguistic assistance for model development [26]. The authors state that few tools exist to assist the transition of textual requirement documents to conceptual models, and that the heuristics used for this transition often only rely on modeler experience. The proposed tool, called LIDA, aids the analysis of

documents like use cases and interviews. Users are able to use the tool to analyze these texts, assigns types (for example, class or attribute), and create UML models that can be further developed in other tools. Although this tool aids model development using linguistic analysis, there is no implementation of automatic natural language processing. Analysis of documents still requires human interference and is therefore still reliant on modeler experience and views. Furthermore, this solution is also not modeling language-independent, since it relies on a subset of UML for model generation. Again, this system focuses on a different phase of the modeling process, but might therefore complement our research.

These are just two solutions in a larger body of research looking into the use of NLP for model creation. Many other solutions are presented that aim to minimize modeling efforts by simplifying the analysis of natural language requirements documents [40, 25, 9, 22, 15]. These solutions all have one thing in common: they focus on the pre-modeling stage of requirements engineering. The proposed solutions take requirements documents as input and either guide analysis of these documents, assign types to elements (or words) in the documents, or produce complete models based on the documents. The first problem with this approach is that the solutions are not language-independent, and therefore require knowledge about modeling language meta-models, making the implementation less flexible. Another visible problem is the distinction in human interference. Solutions that focus mostly on the document analysis part require too much human interference, whereas the solutions that also look at model creation tend to want to minimize human interference too much. Our research differs in that sense, since it focuses on the modeling part of requirements engineering, does not require human interference for analysis, does not analyze requirements documents, and only guides the modeling process.

4.1.2 Empirical evaluations in conceptual modeling

Our research is definitely not the first empirical evaluation of conceptual modeling techniques. A vast amount of research has been devoted to evaluating various aspects of conceptual modeling techniques using empirical evaluations, and with that several frameworks have been developed for conducting these empirical evaluations. Gemino & Wand have proposed such a framework for empirical evaluation of conceptual modeling techniques [14]. Their proposed framework differs from our approach, however, as it focuses on the comparison of modeling grammars/languages. Our approach is focused on the evaluation of the effectiveness of the suggestion service, and does not look at the modeling grammar's ability to model a domain. Burton et al. propose a framework for evaluating modeling scripts (which aligns with our research approach), but also place their focus on the influence of modeling grammar abilities to create those scripts [4]. In that sense, our research differs from the most common empirical evaluation techniques, as we do not define the modeling language as the primary affecting variable. We look for differences in modeling scripts based on the suggestion service as an affecting variable, rather than the modeling language's ability to create those scripts.

Similar to our research, Rittgen has done an empirical evaluation on the collaborative creation of conceptual models [29, 28]. In his research, participants also start with little to no domain knowledge, as this will simulate a situation where information and knowledge is created by modelers. What is different, is that the translation of natural language to model artifacts is not done with natural language processing techniques, but by negotiation of participants working together. In our experiments, there was no room for negotiation, as participants were working alone. An interesting result is the "negotiation pattern" that Rittgen found, indicating that collaborative modeling is often a well-structured process (rather than a creative process, which

is often thought). It would be interesting to study this negotiation pattern for our suggestion service, as our service does not allow direct face-to-face negotiation with other modelers. The benefit of having tool-supported collaborative modeling is also supported by an empirical evaluation by Dean et al., who state that an increase in modeler groups (with modelers being allowed to work on models synchronously) speeds up the modeling process and decreases model ambiguity [8].

4.1.3 Conceptual model quality factors

Wand and Weber have described evaluation approaches for conceptual modeling techniques, and proposed several directions for research into various aspects of conceptual modeling [37]. They cite various papers researching quality factors of modeling scripts (models created using a modeling grammar), such as completeness, syntactic quality, and ontology soundness. However, they also state that most research focuses solely on these quality factors, rather than the process of modeling and the modeling language capacities to achieve these qualities. The authors also propose the usefulness of an inter-grammar approach to evaluate modeling language abilities to create modeling scripts for a certain domain. Our research ties in with these points of interest, as it defines certain quality factors for conceptual models (completeness, syntactic correctness, and model alignment), but it also focuses on the modeling process (by doing a qualitative analysis of modeler-service interaction) and modeling language abilities (by evaluating the influence of modeling language on our defined quality factors).

We see that our definition of quality factors aligns with other research [20, 23, 17]. However, it should also be noted that our definition of model quality factors is not an exhaustive list. It is also not meant to be, but this does mean that we cannot make assumptions about the service's abilities to allow for "better" model creation. For example, Moody & Shanks also list flexibility and simplicity as quality factors [23], Lindberg et al. mention traceability [20], and Kesh mentions soundness [17]. Certain quality factors should be omitted in our definition, however, as they do not fit our research approach. For example, since we take a modeling language-independent approach, we cannot take soundness of models into consideration, as this would require evaluation of the correctness of model elements. For now, we can only draw conclusions on the influence of the suggestion service on the subset of quality factors that we have chosen.

In our research, we have focused on two quality categories, namely that of syntactic quality and, partly, semantic quality [24]. The last category focuses on the relationship between the model and its domain, splitting two quality goals: validity (whether or not there are statements/elements in the model that are incorrect) and completeness. Where we do look at completeness, we do not check for validity. We also have not looked at qualities relating to model interpretation (pragmatic quality). These quality factors could have been evaluated in our research approach, and were not omitted due to research restrictions. This indicates that there are certain quality categories that we could have looked at (in more detail).

4.1.4 Conceptual model alignment

Kof et al. have described conceptual model alignment as a means for requirements validation [18]. However, their research differs from ours in the sense that they define alignment as a model's match with the ontology model. This match is purposefully not evaluated in our research. Their

approach does match our approach for determining model completeness, which is also similar to our approach for measuring model alignment (with the only difference being the comparison model). This approach entails defining the concepts that are present in one model, but not present in the other. Different lexical forms for concepts are taken into account, meaning that, similar to our research, a substring match is used to determine alignment. A useful research approach that we have not taken is determining which unmatched concepts actually should be matched. We assume that every unmatched concept should be matched, but this might not always be the case.

Other research looks at model merging to check for consistency between models. Sabetzadeh et al. mention the difficulty in reaching model consistency for distributed development [32]. They further mention that the matching of two models does not automatically imply overall consistent matching. Model merging looks into the matching of related models by various team members concerning the same domain. Although this often does not take different modeling languages (or concerns) into account, it could be useful to research the possibility of model merging for different modeling languages as a means to check consistency and alignment.

4.2 Suggestions for future research

A first suggestion for future research is conducting an empirical evaluation on a larger scale. Due to resource and time constraint, we were limited to a sample size of (initially) 20 participants. A smaller sample size has some limitations. For example, we were not able to test the influence of more affecting variables, as this would require dividing the experiment group further. This restriction also limited us to using only the base models as control models. In future research, it would be useful if the participant group could be divided to include a subgroup creating models without suggestions, and a subgroup creating models with suggestions.

Resource constraint also limited our experiment set-up, as we were only able to let one participant working on the modeling task at a time. As has been stated before, this is not the most accurate representation of a real-life situation. In future research, it would be useful to focus on the collaboration aspect of modeling by letting two modelers working at the same time, thereby making their suggestions depend on each other's models.

Another interesting research approach would be to test the application of the suggestion outside of an academic context. A first step would be to involve domain experts evaluating the effectiveness of the service (similar to [30]), after which implementation in business and modeling processes can be studied. This would define the required steps for both the service and the places of implementation to enable successful use of the service in real-life situations.

Lastly, testing different heuristics for the suggestion service might indicate affecting factors within the service that influence its effectiveness. Perhaps a different approach to natural language processing techniques will influence effectiveness.

4.2.1 Threats to validity

External validity
External threats to validity relate to the generalizability of our results. There has been one case were the modeling task was finished before the one hour mark. In another case, a participant's Attention Deficit Hyperactivity Disorder (ADHD) medication stopped working, possibly effecting

the produced model. Both of these cases might affect the generalizability of our results. Several other factors might also affect generalizability. We had a relatively low number of participants, and none of these participants were expert modelers (as opposed to the modelers who created the base models, and the modelers for whom this service is intended). The experiments were also conducted in a lab setting, rather than a real-life situation, and we chose a scenario that is not the most accurate representation of a real-life situation. This makes it harder to make assumptions on the implication of our results in the real world.

Internal validity
Internal threats to validity relate to how our experiments were conducted. There might be a bias in participant selection due to convenience sampling, as the researcher primarily reached out to own contacts. Also, although an observation protocol was followed, there might have been a small variation in information provided to participants before experiments. The qualitative data was manually coded, thereby possibly giving room to subjective interpretation.

Construct validity
There is a mono-operation bias in this research, as we only use one domain to evaluate the effectiveness of suggestions.

Chapter 5

Conclusion

With this research, we have designed an experiment for empirical evaluation of conceptual modeling techniques. Specifically, we have tested the effectiveness of a service that analyzes textual models in a domain and provides suggestions to modelers for models to be created based on these models. In order to test this service, we have extended two open source modeling tools (for iStar and BPMN) by implementing the suggestion service. The empirical evaluation included 18 representative end users, who created a total of 16 models over the course of eight days. Our results aimed at answering four research questions.

The first question asked: "How much does this suggestion service improve model completeness?". A script was written and used to determine the share of ontology nouns that were present in the comparison model. We found that, in general, the models created in our experiments were significantly more complete than the base models. We found that this significance also holds for both languages separately. Comparing the languages, we found that the created BPMN models are significantly more complete than the created iStar models. We concluded this research question by stating that non-expert modelers using the suggestion service create models that are at least as complete as models created by expert modelers without the suggestion service.

Our second question asked: "How much does this suggestion service help in using the syntactically correct terminology?". A script was used to determine the share of noun chunks in the ontology that were also present in the comparison model. We found that, in general, the syntax scores of the created models were significantly lower than the syntax scores of the base models. We found that this also holds for the iStar models alone, but the BPMN models did not have significantly lower syntax scores. We concluded by stating that non-expert modelers using the suggestion service do not create models that are at least as syntactically correct as expert modelers that do not use the service.

The third question asked: "How much does this suggestion service improve model alignment?". A script was used to determine the share of matched nouns between a base model and a comparison model. The alignment scores for the created models were not significantly higher than the base models. There was also no significant difference for both languages separately. However, we did find that the created iStar models had significantly lower alignment scores than the created BPMN models.

For the fourth question, "How does the modeler-service interaction for this suggestion service affect its effectiveness?", we combined qualitative and quantitative data analysis. Starting with

the qualitative data, we found that participants were primarily negative towards the suggestion service. The service was trusted less than the domain website, which aligned with participant actions during the modeling task. The main reason for negative attitudes was listed as wrong timing of the suggestions, but positivity was expressed about the service pointing towards excluded elements. Interview responses mostly aligned with experiment data. No significant group differences were found concerning the qualitative data. For the quantitative data, three measures for modeler-service interaction were determined. No significant correlation between these measures was found. We did find a significant correlation between the amount of suggestions and syntax scores for BPMN models, indicating that BPMN models might require more suggestions in order to create a syntactically correct model, but that the suggestion service does not allow for significantly more syntactically complete models.

Appendix A

Observation Protocol

Observation information

Date:		**Observer:**	
Location:		**Experiment no.:**	
Time started:		**Language:**	
Time ended:			

Participant information

Confidence in model creation:	*unconfident / somewhat unconfident / neutral / somewhat confident / confident*
Model domain familiarity:	*unfamiliar / somewhat unfamiliar / neutral / somewhat familiar / familiar*

Script

Purpose

This experiment aims to test the influence of suggestions on creating a conceptual model. Its application is in collaborative modeling, letting modelers use knowledge from each other's models. The suggestions are therefore generated based on other models in the same domain. Right now, there are already other models in the project on which the suggestions will be based. You will be creating a model in *[language]* that captures the process of signing up for a Business Informatics masters thesis at Utrecht University. The results do not depend on your familiarity with this domain.

Tool

For this task, you will use an open source modeling tool that we have extended to be able to provide suggestions for what you are modeling. These suggestions can be included in your model as you see fit. When I start the time, you can also take some time to get familiar with the tool elements on the *[location]* side of the screen. You will see a button saying Give suggestions at *[location]*. Suggestions can be requested whenever you deem it useful or necessary. I ask you to

rate these suggestions either useful or not useful, after which the suggestions will be visible in the "Past feedback" section. It is necessary to rate all suggestions, but keep in mind that they are used as a filter for future suggestions. You can access the suggestions you have rated previously by clicking View past suggestions. Use only the green buttons, the other buttons are only used by the observer. You will be modeling for 1 hour and I will stop you when we reach that time limit. If you have any questions regarding the purpose or the structure of the project, feel free to ask me now.

Experiment
During this experiment, I will be a silent observer whenever possible, but I might interrupt you for questions. You can also ask me questions whenever you need to. I will interrupt you after 20, 40 and 60 minutes to download the model you have created thus far. During this experiment, it is necessary that you vocalize and explain your actions, rationale and thoughts about the modeling process. If you forget to do so, I will remind you to think aloud. You are allowed to stop the experiment whenever you want to.

Structure

- Welcome the participant and explain purpose of the experiment first.
- Let the participant fill in the consent form.
- Fill in *observation information* and *participant information.*
- Let the participant read through the domain briefly if desired.
- Briefly look at the modeling language with participant if desired.
- Walk through the script.
- Show the tool.
- Fill in experiment number.
- Set the timer when modeling task starts.
- Interrupt when the participant forgets to think aloud or explain actions.
- Every 10 minutes, tell the participant to request suggestions if he/she does not do so.
- *After 20 and 40 minutes:* stop the participant after he/she finishes their action and download the model created thus far.
- *After 60 minutes:* stop the participant after he/she finishes their action and download the model created thus far. The experiment is finished.

Observation comments

Observer comment	Time (minutes):	Participant comment:	Corresponding action:

Appendix B

Interview Protocol

The following questions are asked after the observations.

- Can you explain some of your motivations to request suggestions?
- Can you explain some of your motivations to not request suggestions?
- Did you think the suggestions were useful?
- Do you think that the suggestions helped you create a better model than you would have without suggestions?
- Do you think that the suggestion service improves collaboration with other modelers?
- Did the suggestions make you more confident in your ability to create a model?
- Are there possible changes to the tool that would increase its usefulness?
- Do you have any final comments on this experiment?

Appendix C

Experiment Consent Form

Principal investigator: _____

Study working title: _____

INFORMED CONSENT

Introduction
You are asked to participate in an observation study for testing the influence of a suggestion service on conceptual modeling. A combination of data sources are used to provide answers to research questions in this study: observation data, interview data and model data. You will receive a modeling task with a duration of 1 hour after which a short interview will be held. After this experiment, we will not contact you for further questions.

Your data will be anonymised for processing and publication. We will not share personal data that might identify you to others. We will keep experiment data for a maximum of 5 years.

You are able to drop out of this experiment at any time.

Please tick the following boxes for agreement:

☐ **I agree that comments made during this experiment will be anonymised and used for processing and reporting.**

☐ **I understand that the research data, without any personal information that could identify me (not linked to me) may be shared with others.**

PARTICIPANT SIGNATURE

Name _____

Signature _____

Date _____

OBSERVER SIGNATURE

Name _____

Signature _____

Date _____

Appendix D

Experiment Codebook

Please refer to the following page for the codebook of the experiment data. This codebook defines the nodes used in the qualitative data analysis, along with the reference instances for every node.

Codebook

Please note: node information for interview questions is left out of this codebook.

Nodes

Name	Description	Files	References
Domain	**Top-level node:** collecting data relating to the domain (website).	16	123
Actions	Defines actions relating to the domain.	2	9
Comments	Defines comments made about the domain.	5	8
Confusion	Defines comments made about the domain by confused participants.	8	11
Inclusion in Model	Defines instances where elements from the domain are included in the model.	16	59
Motivation	Indicates the motivation of participants to check the domain.	12	34
Suggestion Check	Defines instances where participants use suggestions to check the content of the domain.	2	2
No Suggestions	Defines instances where participants use neither the domain nor the suggestion service.	8	11
Noticeable quotes		5	10
Past suggestions	Indicates instances and reasons for referring to past suggestions.	7	12
Suggestions	**Top-level node:** collecting data relating to the suggestion service.	16	644
Actions		16	286
Feedback		16	150
Not Useful	Defines instances and reasons of participants for rating suggestions as “not useful”.	15	93
Useful	Defines instances and reasons of participants for rating suggestions as “useful”.	14	57
Inclusion in Model	Defines instances where suggestions are included in the model.	12	49
Looking at Domain	Defines instances where participants refer to the domain to check suggestions.	14	53
No Action	Defines instances where a suggestion request is not followed by any action relating to those suggestions.	12	34
Attitude	Indicates the (motivations for) certain attitudes of participants towards the service.	16	233
Confused or Unsure		16	60
Negative		16	79

Name	Description	Files	References
Neutral		15	36
Positive		15	58
Requests		16	125
Motivation		16	125
Observer Request	Counts the instances where the observer tells participants to request suggestions.	13	32
Own Motivation	Indicates motivations of participants to request suggestions.	15	93

Bibliography

[1] Aydemir, F. B., and Dalpiaz, F. Towards aligning multi-concern models via nlp. In *2017 IEEE 25th International Requirements Engineering Conference Workshops (REW)* (2017), IEEE, pp. 46–50.

[2] Becker, T. E. Potential problems in the statistical control of variables in organizational research: A qualitative analysis with recommendations. *Organizational Research Methods 8*, 3 (2005), 274–289.

[3] Benbunan-Fich, R. Using protocol analysis to evaluate the usability of a commercial web site. *Information & management 39*, 2 (2001), 151–163.

[4] Burton-Jones, A., Wand, Y., and Weber, R. Guidelines for empirical evaluations of conceptual modeling grammars. *Journal of the Association for Information Systems 10*, 6 (2009), 1.

[5] Chinosi, M., and Trombetta, A. Bpmn: An introduction to the standard. *Computer Standards & Interfaces 34*, 1 (2012), 124–134.

[6] Dalpiaz, F., Franch, X., and Horkoff, J. istar 2.0 language guide. *arXiv preprint arXiv:1605.07767* (2016).

[7] De la Vara González, J. L., and Diaz, J. S. Business process-driven requirements engineering: a goal-based approach. In *Proceedings of the 8th Workshop on Business Process Modeling* (2007), Citeseer.

[8] Dean, D., Orwig, R., Lee, J., and Vogel, D. Modeling with a group modeling tool: group support, model quality, and validation. In *HICSS (4)* (1994), pp. 214–223.

[9] Du, S., and Metzler, D. P. An automated multi-component approach to extracting entity relationships from database requirement specification documents. In *International conference on application of natural language to information systems* (2006), Springer, pp. 1–11.

[10] Dumas, J. S., and Redish, J. *A practical guide to usability testing*. Intellect books, 1999.

[11] Edmonds, B., and Hales, D. Replication, replication and replication: Some hard lessons from model alignment. *Journal of Artificial Societies and Social Simulation 6*, 4 (2003).

[12] Eisenhardt, K. M. Building theories from case study research. *Academy of management review 14*, 4 (1989), 532–550.

[13] Gemino, A., and Wand, Y. Evaluating modeling techniques based on models of learning. *Communications of the ACM 46*, 10 (2003), 79–84.

[14] Gemino, A., and Wand, Y. A framework for empirical evaluation of conceptual modeling techniques. *Requirements Engineering 9*, 4 (2004), 248–260.

[15] Harmain, H. M., and Gaizauskas, R. Cm-builder: an automated nl-based case tool. In *Automated Software Engineering, 2000. Proceedings ASE 2000. The Fifteenth IEEE International Conference on* (2000), IEEE, pp. 45–53.

[16] Kaulio, M. A., and Karlsson, I. M. Triangulation strategies in user requirements investigations: a case study on the development of an it-mediated service. *Behaviour & information technology 17*, 2 (1998), 103–112.

[17] Kesh, S. Evaluating the quality of entity relationship models. *Information and Software Technology 37*, 12 (1995), 681–689.

[18] Kof, L., Gacitua, R., Rouncefield, M., and Sawyer, P. Ontology and model alignment as a means for requirements validation. In *Semantic Computing (ICSC), 2010 IEEE Fourth International Conference on* (2010), IEEE, pp. 46–51.

[19] Lenth, R. V. Some practical guidelines for effective sample size determination. *The American Statistician 55*, 3 (2001), 187–193.

[20] Lindland, O. I., Sindre, G., and Solvberg, A. Understanding quality in conceptual modeling. *IEEE software 11*, 2 (1994), 42–49.

[21] Marshall, B., Cardon, P., Poddar, A., and Fontenot, R. Does sample size matter in qualitative research?: A review of qualitative interviews in is research. *Journal of Computer Information Systems 54*, 1 (2013), 11–22.

[22] Mich, L. Nl-oops: from natural language to object oriented requirements using the natural language processing system lolita. *Natural language engineering 2*, 2 (1996), 161–187.

[23] Moody, D. L., Shanks, G. G., and Darke, P. Improving the quality of entity relationship modelsexperience in research and practice. In *International Conference on Conceptual Modeling* (1998), Springer, pp. 255–276.

[24] Moody, D. L., Sindre, G., Brasethvik, T., and Sølvberg, A. Evaluating the quality of information models: empirical testing of a conceptual model quality framework. In *Proceedings of the 25th international conference on software engineering* (2003), IEEE Computer Society, pp. 295–305.

[25] Omar, N., Hanna, J., and McKevitt, P. Heuristic-based entity-relationship modelling through natural language processing. In *Artificial intelligence and cognitive science conference (aics)* (2004), Artificial Intelligence Association of Ireland (AIAI), pp. 302–313.

[26] Overmyer, S. P., Lavoie, B., and Rambow, O. Conceptual modeling through linguistic analysis using lida. In *Proceedings of the 23rd international conference on Software engineering* (2001), IEEE Computer Society, pp. 401–410.

[27] Pinsonneault, A., and Kraemer, K. Survey research methodology in management information systems: an assessment. *Journal of management information systems 10*, 2 (1993), 75–105.

[28] Rittgen, P. Coma: A tool for collaborative modeling. In *CAiSE Forum* (2008), vol. 344, pp. 61–64.

[29] Rittgen, P. Collaborative modeling-a design science approach. In *System Sciences, 2009. HICSS'09. 42nd Hawaii International Conference on* (2009), IEEE, pp. 1–10.

[30] Robeer, M., Lucassen, G., van der Werf, J. M. E., Dalpiaz, F., and Brinkkemper, S. Automated extraction of conceptual models from user stories via nlp. In *Requirements engineering conference (RE), 2016 IEEE 24th international* (2016), IEEE, pp. 196–205.

[31] Sabetzadeh, M., and Easterbrook, S. View merging in the presence of incompleteness and inconsistency. *Requirements Engineering 11*, 3 (2006), 174–193.

[32] Sabetzadeh, M., Nejati, S., Liaskos, S., Easterbrook, S., and Chechik, M. Consistency checking of conceptual models via model merging. In *Requirements Engineering Conference, 2007. RE'07. 15th IEEE International* (2007), IEEE, pp. 221–230.

[33] Seaman, C. B. Qualitative methods in empirical studies of software engineering. *IEEE Transactions on software engineering 25*, 4 (1999), 557–572.

[34] Shanks, G., Tansley, E., and Weber, R. Using ontology to validate conceptual models. *Communications of the ACM 46*, 10 (2003), 85–89.

[35] Sullivan, P. Multiple methods and the usability of interface prototypes: the complementarity of laboratory observation and focus groups. In *Proceedings of the 9th annual international conference on Systems documentation* (1991), ACM, pp. 106–112.

[36] Trösterer, S., Beck, E., Dalpiaz, F., Paja, E., Giorgini, P., and Tscheligi, M. Formative user-centered evaluation of security modeling: Results from a case study. *International Journal of Secure Software Engineering (IJSSE) 3*, 1 (2012), 1–19.

[37] Wand, Y., and Weber, R. Research commentary: information systems and conceptual modelinga research agenda. *Information systems research 13*, 4 (2002), 363–376.

[38] Wilson, C. E. Triangulation: the explicit use of multiple methods, measures, and approaches for determining core issues in product development. *interactions 13*, 6 (2006), 46–ff.

[39] Young, S. J., Russell, N., and Thornton, J. *Token passing: a simple conceptual model for connected speech recognition systems.* Cambridge University Engineering Department Cambridge, UK, 1989.

[40] Yue, T., Briand, L. C., and Labiche, Y. atoucan: an automated framework to derive uml analysis models from use case models. *ACM Transactions on Software Engineering and Methodology (TOSEM) 24*, 3 (2015), 13.