

Bachelor Thesis

---

30th January 2018

# Designing and Conducting Controlled Experiments on an Experimental Requirements Modeling Tool for Evaluating ImitGraphs

**David Lay**

of Mainz, Germany (14-707-939)

supervised by

Prof. Dr. Martin Glinz

Parisa Ghazi



**University of  
Zurich<sup>UZH</sup>**

**Department of Informatics**





Bachelor Thesis

---

# Designing and Conducting Controlled Experiments on an Experimental Requirements Modeling Tool for Evaluating ImitGraphs

David Lay



University of  
Zurich<sup>UZH</sup>

Department of Informatics



**Bachelor Thesis**

**Author:** David Lay, david.lay@uzh.ch

**Project period:** 01. October 2017 - 30. January 2018

Department of Informatics, University of Zurich

---

# Acknowledgements

I would like to express my very great appreciation to my advisor Parisa Ghazi for her extraordinary support in this thesis process. Also, her patient guidance, encouragement, useful suggestions and critiques kept the project on schedule for the last four months.

Moreover, I want to thank Professor Doctor Martin Glinz for accepting me as one of his students and thus being able to write this thesis at the Requirements Engineering Research Group of the University of Zurich.

I am particularly grateful for the assistance given by Nico Strebel who adapted the modeling tool, with which we conducted the experiments, based on the given suggestions. His willingness to implement the features has been very much appreciated.

Further, I want to thank all participants who took part in the experiments and sacrificed parts of their free time. This work would not have been possible without them.



---

# Abstract

The complexity and variety of graphical models (e.g. class diagrams or activity diagrams) lead to a wide range of user interfaces to create and manipulate such graphical models. Moreover, the development of working prototypes for software usability tests is costly. To simplify such usability tests ImitGraphs – a specialized graphical model to simulate the behavior of its intended graphical model – were proposed in earlier works. To evaluate the behavior of ImitGraphs, we designed and conducted two series of experiments that should test whether participants interact with an ImitGraph similar as with its intended graphical model and if such usability tests can be conducted with participants that have no prior knowledge of designing graphical models. Analyzing the obtained data showed, that participants complete given tasks with similar types and number of steps, regardless of the model used or their experience. Based on the observed behavior we conclude that ImitGraphs could indeed replace the intended graphical models in usability tests and thus allows to conduct more frequent tests in earlier phases of the software development cycle. Also, a wider range of participants with no prior knowledge about designing graphical models could attend usability tests that are based on ImitGraphs.





---

# Zusammenfassung

Die Komplexität und Vielfalt verschiedener grafischer Modelle (z.B. Klassendiagramme oder Aktivitätsdiagramme) führen zu einer Grosszahl an Benutzerschnittstellen um solche Modelle zu bearbeiten. Auch ist die Entwicklung funktionierender Prototypen zur Verwendung in Software Usability-Tests kostspielig. Um diese zu vereinfachen, wurden in früheren Arbeiten ImitGraphen – ein spezialisiertes grafisches Modell, welches andere grafische Modelle imitiert – vorgestellt. Um das Verhalten mit ImitGraphen evaluieren zu können, wurden zwei Experimente entworfen und mit mehreren Teilnehmern durchgeführt. Die Experimente sollten beurteilen, inwiefern sich das Verhalten mit ImitGraphen zum Verhalten mit dem dazugehörigen grafischen Modell gleicht und ob Usability-Tests, basierend auf ImitGraphen, auf einer breiteren Bevölkerungsschicht durchgeführt werden können. Resultate zeigen, dass sich unabhängig von der Erfahrung der Teilnehmer oder des verwendeten Modells, die Anzahl und Art der durchgeführten Schritte, um Aufgaben abzuschliessen, nicht unterscheiden. Anhand der gefundenen Ergebnisse können ImitGraphen zukünftig in Usability-Tests eingesetzt werden und somit grafische Modelle in diesen Tests ersetzen. Somit können Usability-Tests in früheren Phasen der Entwicklung von Benutzerschnittstellen durchgeführt werden. Ebenfalls untermauern die Experimente die Annahme, dass Teilnehmer ohne Kenntnisse über grafische Modelle zur Durchführung von Usability-Tests, welche auf ImitGraphen beruhen, geeignet sind.



---

# Contents

<b>1</b>	<b>Introduction</b>	<b>1</b>
1.1	Context . . . . .	1
1.2	Thesis Goal . . . . .	1
1.3	ImitGraphs . . . . .	2
1.3.1	Definition of ImitGraphs . . . . .	2
1.3.2	ImitGraph Commands . . . . .	4
1.4	Experimental Tool . . . . .	5
1.4.1	ImitGraph Implementation . . . . .	5
1.5	Thesis Outline . . . . .	6
<b>2</b>	<b>Related Work</b>	<b>7</b>
<b>3</b>	<b>Methodology</b>	<b>9</b>
3.1	Proposed Plan . . . . .	9
3.2	Research Objectives . . . . .	9
3.3	Variables . . . . .	11
3.4	Participants . . . . .	12
3.5	Apparatus . . . . .	12
3.6	Procedure . . . . .	13
3.6.1	General Procedure . . . . .	13
3.6.2	Procedure of Experiment 2 . . . . .	14
3.6.3	Procedure of Experiment 1 . . . . .	16
3.7	Experiment Design . . . . .	18
<b>4</b>	<b>Results and Discussion</b>	<b>21</b>
4.1	Experiment 1 . . . . .	21
4.1.1	Task Completion Time . . . . .	21
4.1.2	Steps Taken and Interaction Behavior . . . . .	25
4.1.3	Errors . . . . .	29
4.1.4	Overall Satisfaction . . . . .	30
4.2	Experiment 2 . . . . .	31
4.2.1	Task Completion Time . . . . .	32
4.2.2	Steps Taken and Interaction Behavior . . . . .	34
4.2.3	Errors . . . . .	37
4.2.4	Learnability . . . . .	38
4.2.5	Overall Satisfaction . . . . .	40

---

<b>5</b>	<b>Validity</b>	<b>41</b>
5.1	Construct Validity . . . . .	41
5.2	Conclusion Validity . . . . .	41
5.3	Internal Validity . . . . .	41
5.4	External Validity . . . . .	42
<b>6</b>	<b>Conclusion and Future Work</b>	<b>43</b>
6.1	Conclusion . . . . .	43
6.2	Future Work . . . . .	45
	<b>Bibliography</b>	<b>47</b>
<b>A</b>	<b>Task Descriptions and Questionnaires</b>	<b>49</b>
A.1	Trial Description . . . . .	49
A.2	ImitGraph Task Descriptions . . . . .	55
A.3	Graphical Model Task Descriptions . . . . .	62
A.4	Questionnaires . . . . .	68
<b>B</b>	<b>Content on the Delivered CD-ROM</b>	<b>71</b>

---

# Keywords

**ImitGraph/IG** Parametrized graphical model that imitates the behavior of a corresponding graphical model.

**Corresponding/Intended Graphical Model** The original graphical model which is imitated by the ImitGraph.

**AD/CD/ERD** Abbreviations for Activity Diagram, Class Diagram and Entity Relationship Diagram

**ImitGraph Specification** The rules for an specific ImitGraph combined in the tables for joints, connections and nodes.

**Task Pair** A pair of tasks, compared in the evaluation, consisting of an ImitGraph task and its corresponding graphical model.

**ImitGraph Commands** Series of partially textual and partially visual commands that instruct participants how to complete a task.

**Graphical Model Commands** Series of textual commands that instruct participants how to complete a task.

**Experimental Tool** The tool used to conduct the experiments. Capable of handling ImitGraphs and the corresponding graphical models. Based on draw.io.

**Open Source** Software with an available source code to study, change, and distribute the software to anyone and under any purpose.

**JSON** JavaScript Object Notation, open standard file format to transmit data objects.

**draw.io Library** Available elements to complete each task in draw.io, also referred as palette.

**Experienced Participants** Participants with prior knowledge of designing graphical models.

**Novice Participants** Participants with no prior knowledge of designing graphical models.

## List of Figures

1.1	Node and connection of a specified ImitGraph (Class Diagram) . . . . .	2
1.2	Library of Class Diagram and its ImitGraph in the experimental tool . . . . .	5
3.1	Set-up view as seen by participants . . . . .	13
3.2	Explanation of the "Branch" command in the handout and tool . . . . .	15
3.3	Activity Diagram ImitGraph task description handout . . . . .	16
3.4	Initial situation and one possible outcome of Activity Diagram tasks . . . . .	17
3.5	Activity Diagram task description handout . . . . .	18
4.1	Box-plot and means of task completion times of Experiment 1 . . . . .	22
4.2	Box-plot and means of number of steps taken in Experiment 1 . . . . .	25
4.3	Example of dropping a node onto a connection . . . . .	28
4.4	Example of adding elements before connecting . . . . .	29
4.5	Percentage of participants who completed tasks without errors . . . . .	30
4.6	Preferences for models as stated by participants . . . . .	31
4.7	Box-plot and means of task completion times of Experiment 2 . . . . .	32
4.8	Box-plot and means of number of steps taken in Experiment 2 . . . . .	35
4.9	Change in efficiency from task to task in percentage of participants . . . . .	38
4.10	Change in efficiency and errors made from task to task . . . . .	39
A.1	Initial situation and one possible outcome of ImitGraph task (Activity Diagram) . .	58
A.2	Initial situation and one possible outcome of ImitGraph task (Class Diagram) . . .	60
A.3	Initial situation and one possible outcome of ImitGraph task (Entity Relationship Diagram) . . . . .	62
A.4	Initial situation and one possible outcome of Activity Diagram task . . . . .	64
A.5	Initial situation and one possible outcome of Class Diagram task . . . . .	66
A.6	Initial situation and one possible outcome of Entity Relationship Diagram task . .	68
A.7	ImitGraph questionnaire . . . . .	69
A.8	Graphical model questionnaire . . . . .	70

## List of Tables

1.1	Joint types . . . . .	3
1.2	Connection types . . . . .	3
1.3	Node types . . . . .	4
3.1	Proposed Plan . . . . .	10
3.2	Counterbalanced task order using a Latin Square . . . . .	19
4.1	Descriptive statistics of task completion times (sec) of Experiment 1 . . . . .	22
4.2	TOST results of task completion times of Experiment 1 . . . . .	23
4.3	Descriptive statistics of number of steps taken in Experiment 1 . . . . .	26
4.4	Descriptive statistics of task completion times (sec) of Experiment 2 . . . . .	32
4.5	TOST results of task completion times of Experiment 2 . . . . .	33
4.6	Descriptive statistics of number of steps taken in Experiment 2 . . . . .	35
4.7	Descriptive statistics of errors made during tasks of Experiment 2 . . . . .	37

# Introduction

## 1.1 Context

The use of graphical models in various stages of software development becomes increasingly more important and therefore software engineers spend a significant percentage of their time interacting with modeling tools [7]. To achieve a high productivity during the design phase of a model, effective and efficient user interfaces for manipulating graphical models is crucial. Designing new or improving existing user interface techniques requires multiple cycles of testing and adapting the system under development. Thus, the development of user interfaces became an iterative process [15]. One way to test the quality of a developed technique is to carry out ongoing usability tests with a range of participants. The recruited participants then perform several tasks with conventional and new interaction techniques, so that the improvement of the new interaction technique can be measured. In an ideal world all models that the user interface can handle should be covered by usability tests [13]. But in consequence of the variability and complexity of graphical models, a full comprehensive usability test, which could cover all graphical models, becomes expensive. Thus, interface designers either test only a subset of their intended graphical models or test at a late stage of tool development [7]. To avoid this pitfall a simple model is needed that has enough complexity to simulate different graphical models by adaption of its properties.

Such a special type of graph that can handle the various complexities of graphical models was defined by Ghazi and Glinz in [7] and is called ImitGraph. It is designed to imitate the behavior of different graphical models used in software engineering and can be adapted to the needs of the intended graphical model. Such an interaction behavior is then similar if equivalent given instructions for different models (ImitGraph and graphical model) lead to equivalent actions taken [7]. Given a similar interaction behavior, ImitGraphs could be used in usability tests instead of graphical models and thus lead to a (i) faster and less expensive development of prototypes with the possibility of evaluating ideas early, an (ii) effective evaluation on various graphical models, and a (iii) chance to recruit participants with no prior knowledge of the intended graphical models for usability tests [7].

## 1.2 Thesis Goal

At the current stage, this thesis has the objective to evaluate ImitGraphs and find out the extent to which ImitGraphs can simulate the interaction behavior and layout properties of graphical

models used for software-modeling purposes. The main goal of this thesis is the designing and conducting of two controlled experiments as well as the later analysis and interpretation of the collected data. Experiment 1 compares the interactions that are needed to create or edit ImitGraphs and the interactions that are needed to perform a similar task on the original graphical model. The second experiment compares differences in the interaction between novice and experienced participants when working with ImitGraphs.

To achieve these goals, we specified three different ImitGraphs that are based on some of the most frequently used graphical models in software engineering. Namely Activity Diagrams, Class Diagrams and Entity Relationship Diagrams. We then designed three tasks for the specified ImitGraphs and similar tasks for the corresponding graphical models. To test for interaction similarities between ImitGraphs and the graphical model, a tool similar to an existing modeling tool that can handle ImitGraphs was implemented.

## 1.3 ImitGraphs

As described in the Section 1.1, an ImitGraph is a special type of graph that simulates the behavior of different graphical models used in software-modeling domains and thus can be used for usability tests instead. To meet the requirements of imitating complex graphical models, ImitGraphs should meet the following four requirements as stated in [7]:

1. Imitate the behavior of graphical models when being manipulated
2. Simple enough to allow quick implementation of new interaction techniques
3. Potential to represent a large group of graphical models
4. Easy to learn for participants in usability tests

Having these four properties ImitGraphs can be specified to simulate the behavior of corresponding graphical models chosen in advance. Having this specification usability testers should interact in similar ways with ImitGraphs as with the intended graphical model. Therefore, the simpler ImitGraphs could be used in usability tests for modeling tools instead of the original graphical models. This would allow a quick implementation of different graphical models using ImitGraphs so that usability tests could be conducted on a wider range of graphical models if needed.

### 1.3.1 Definition of ImitGraphs

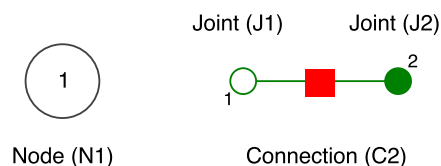


Figure 1.1: Node and connection of a specified ImitGraph (Class Diagram)



ImitGraphs are based on simple graphs but have several additional properties regarding their nodes and connections. With those additional properties ImitGraphs can be specified to meet the requirements of the corresponding graphical models. ImitGraphs are composed of three elements, namely nodes, connections and joints. In Figure 1.1 an example of a node and connection can be seen, specified by an ImitGraph that is imitating a Class Diagram.

**Node** Nodes are circular elements of different sizes, colors and can hold a label. Additionally, nodes define how they can be connected. Therefore, it is defined which joint of a connection can be attached at which connection point of a node.

**Connection** Connections are lines with a rectangle in the middle and connect two joints. The line can have different lengths and orientations. Rectangles can have different sizes, colors and can hold a label. Additionally, connections define which joints are connected to it and how it is orientated (horizontally, vertical or "any").

**Joint** Joints are circles at the end of a connection. They have a fixed size but can have different colors and can hold a label.

Table 1.1: Joint types








Joint	Symbol	Label
J1		yes
J2		yes
J3		no
J4		no

Table 1.2: Connection types

Type	Symbol	Label	First Joint	Second Joint	Orientation
C1		no	J1	J1	any
C2		no	J2	J1	any
C3		no	J3	J4	any

To specify ImitGraphs different types of nodes, connections and joints have to be specified. For each element of an ImitGraph its properties have to be defined in advance. To achieve an ImitGraph that can imitate the behavior of the intended graphical model as adequately as possible, the element properties need to be defined correspondingly. If an element in the original graphical model holds a label, the corresponding ImitGraph element should hold a label as well. The results of such an instantiation are shown in three tables, one table for nodes, connections and joints each. Table 1.1, Table 1.2 and Table 1.3 show a sample specification for an ImitGraph that imitates Class Diagrams. It can be seen that this specific ImitGraph has four joints, three connections, and two nodes. Such tables are explained to usability testers so that they can execute commands on the ImitGraph under test.

Table 1.3: Node types

Type	Symbol	Label	Connection Type	Joint Type	Min	Max	Connection Point
N1	○	yes	C1	J1	1	n	any
			C2	J2	0	n	any
			C2	J1	0	n	any
			C3	J3	0	n	bottom
			C3	J4	0	n	top
N2	●	no	C3	J3	1	n	any
			C3	J4	1	1	top

### 1.3.2 ImitGraph Commands

ImitGraph commands are partially visual and partially textual. Such a command is executed by a usability tester for a specific ImitGraph. It instructs the participant how to manipulate an existing or new ImitGraph and often has more than one element that needs to be manipulated. A single command does not suggest a specific layout of the elements in the command or how the command should be executed in the tool or a specific order of the single actions taken to complete a command. Therefore, participants of usability tests based on ImitGraph have similar freedoms as if they would work with the graphical model, the ImitGraph is imitating. They also should be able to produce similar errors with ImitGraphs as with the original model. Since ImitGraph commands are designed to be intuitive, participants with no prior knowledge can execute the commands too, assumed they got an explanation about ImitGraphs in advance. Overall nine commands exist and can be given to usability testers. All the commands together with examples can be found in Appendix A.1 and were part of the trial participants completed before executing the tasks on ImitGraphs. A short description of each command can be found below:

**Create** Instructs the participants to create a new node with the specified properties.

**Connect** Instructs the participants to add the specified connection between already existing nodes.

**Branch** Instructs the participants to add a defined sequence of nodes and connections to the graph. Starts in a given node and ends in a new or existing node.

**Find Node** Instructs the participants to find the node specified in the command.

**Find Connection** Instructs the participants to find the specified connection.

**Remember as** Instructs the participants to assign a specified name to a previously found connection or node.

**Insert** Instructs the participants to add the specified node or branch into a previously found connection defined by the command.

**Remove** Instructs the participants to remove the specified element. If the specified element is a node, all its connections have to be removed too.

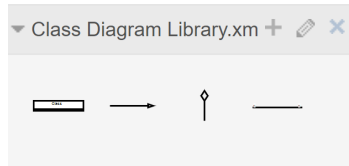
**Replace** Instructs the participants to replace the specified element with another specified element.

## 1.4 Experimental Tool

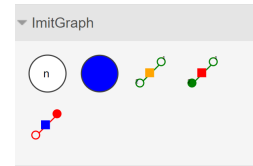
Since the goal of this thesis is to study similar interaction behaviors between ImitGraphs and the corresponding graphical models, the tool to conduct the experiment must be able to handle ImitGraphs and the corresponding graphical models in analogous ways. Parts of the tool were designed before starting this thesis and adjustments were made in the design phase of this thesis experiments. To handle ImitGraphs, the tool is based on the open source tool draw.io. This allowed us to build one tool, where it was possible to decide whether to use ImitGraphs or graphical models. Google Chrome was used to run the tool, but any other web browser would do the trick.

Upload and download of previously designed ImitGraphs or graphical models could be done with the tool. Thus, each participant received the exact layout for each task. Available elements for graphical models were gathered in a library with the possibility of saving the library and upload it for the experiment later. Such a library with the available elements for manipulating a class diagram can be seen in Figure 1.2b.

Interactions such as adding or deleting an element were possible in same ways for ImitGraphs and graphical models, so that participants could choose their preferred method of manipulating the given models. For both models the tool created a log file which recorded the actions taken and stored them as a JSON (JavaScript Object Notation) file ready for download. These log files were later used for the analysis of both conducted experiments.



(a) Class Diagram library



(b) ImitGraph library based on specification

Figure 1.2: Library of Class Diagram and its ImitGraph in the experimental tool

### 1.4.1 ImitGraph Implementation

Once a specific ImitGraph is defined (as described in Section 1.3.1), the ImitGraphs description needs to be transferred to the tool. The specification for one ImitGraph could be uploaded as JSON file and contained all the important information about the specified ImitGraph. Therefore, nodes, connections, and joints were defined with their properties and made available in the tool after uploading the file. The tool allowed to upload several different ImitGraph specifications simultaneously in multiple tabs. Thus, all tasks could be prepared in advance.

The only difference between working with ImitGraphs and the graphical models, are the error messages raised for ImitGraphs when using actions not allowed by the ImitGraphs definition. As an example, consider Table 1.3 for a defined ImitGraph. An error message would be raised if a usability tester is trying to attach a second joint J4 of connection C3 to the node N2. But the node is only allowed to hold a single joint J4 of connection C3. Therefore, as soon as the usability tester attaches the joint J4 to the node an error message is raised, stating that the connection is not allowed. Similar messages are raised if the connection point (to which the connection is attached)

is not allowed or the orientation of the connection is set wrong. After receiving an error message, the graph is reset to the situation before receiving the error message and the usability tester could check the ImitGraphs definition or command for the right solution, expect for these error messages, the used tool does no distinction between ImitGraphs or the corresponding graphical models.

## 1.5 Thesis Outline

The remainder of this document presents detailed descriptions of how the experiments were designed, conducted and analyzed. It is organized as follows. In Chapter 2 the related work is presented and followed by the methodology of how the experiments were designed in Chapter 3. The analysis and discussion of the obtained data from both experiments can be found in 4. In Chapter 5 the experiments validity is discussed. To complete our work the conclusion is presented in Chapter 6 together with the future work.

## Chapter 2

---

# Related Work

In this chapter we describe work that is related to the content of this thesis. The chapter is divided into two paragraphs. The first paragraph states work that is related to designing user interfaces and why usability tests should be conducted at an early stage of development. Controlled experiments, as used in this thesis, are then outlined in the second paragraph including examples.

**User Interface Design and Testing** An ergonomic and intuitive user interface design is one of the most important and most difficult aspects of designing a new system [12]. Therefore, several principles and guidelines [6, 10, 18] exist that explain how to implement such user interfaces. Research on improving user interfaces of modeling tools focus on different aspects. The approaches can deal with visualizing models as in [5, 11] or optimization of the layout as described in [19].

The designed improvements of user interfaces need to be tested following the guidelines of how to implement the interface [15]. Such usability tests are necessary to evaluate the effectiveness of the implemented interface [13] and should be carried out at an early stage of the design cycle as changes to the interface become costly and difficult if implemented too late [8]. Several approaches exist that deal with early usability feedbacks as in [1, 9, 17]. Mostly the approaches (as those mentioned above) that could improve modeling tools are tested on a specific graphical model. The approach used in this thesis leverages ImitGraphs that are supposed to allow usability testers to test their prototypes as early as possible and for different graphical models. This is a result of the easiness of how ImitGraphs are defined and that ImitGraphs can simply be adapted to imitate different graphical models. As ImitGraphs should simulate the behavior of the other graphical models the interaction should be similar between ImitGraphs and the corresponding graphical models. Another gain of choosing ImitGraphs is that the usability tests of the prototypes could be performed on a wider range of participants, that have no prior knowledge about working with graphical models.

**Conducting Controlled Experiments** The process of how usability tests should be carried out is described in various guidelines in the literature [2, 4, 14, 21, 22]. One approach that has been adapted from Psychology and is widely spread to evaluate interfaces and styles of interaction [2], is the conducting of controlled experiments, as done in this thesis. The decision to conduct controlled experiments is based on the work by Easterbrook et al. in [4]. They compare different research methods for empirical software engineering and discuss their trade-offs, where controlled experiments seemed most suitable for our demand. Once decided to conduct controlled experiments, the work by Tullis and Albert [21] on usability metrics was used as guidance. They describe in a step-by-step guide how to collect, analyze and present usability metrics. How to

design and conduct a controlled experiment overall is described by Scott in [14] and Wohlin and Höst in [22]. Whereas the paper by Wohlin gives an overview about controlled experiments the book by Scott is again a step-by-step guide. Scott describes how controlled experiments should be designed from a HCI (Human-Computer Interaction) perspective and how the results should be analyzed with a statistic approach. Furthermore, the book provides information on how to write research papers.

To get more practical knowledge and examples of controlled experiments the works of different authors [3, 16, 20] were studied. Thelin et al. conducted a controlled experiment to evaluate a reading technique called usage-based reading [20]. In their experiment they compared two different approaches, likewise to this thesis. Similar to Tehlin et al., Calero et al. present in [3] an example of how to conduct a controlled experiment. They replicated the work of a previous paper, what shows the need for a detailed description of how controlled experiments are conducted. Other controlled experiments were examined to get a broader overview of what is possible and how controlled experiments should be conducted. The structure of this thesis is partially based on these similar controlled experiments.

## Chapter 3

---

# Methodology

In this chapter the proposed plan of the Bachelor Thesis and the methods for the two conducted experiments are outlined in detail. Because the experiments were carried out by the same participants and similar tasks had to be completed, both methodologies are described together in this chapter.

### 3.1 Proposed Plan

The work contained studying the relevant literature, designing test cases for the experiments, and conducting the experiments with different participants. Furthermore, the functional requirements for the tool were specified and implemented later on. The proposed time plan in Table 3.1 has been followed with slight deviations. Some steps were processed in parallel, such as designing the test cases and implementing the requirements of the tool. Conducting the experiments and writing the thesis took 16 weeks of work overall. Even though the main goal was to compare the interactions that are needed to create or edit ImitGraphs with the interactions needed to perform the similar tasks on the original graphical models, we decided to run another experiment. That experiment was conducted to show if there are any differences in the interaction with ImitGraphs among experienced and novice participants.

### 3.2 Research Objectives

The first step of the design phase was to define a goal that should be achieved with the thesis. Based on this goal, further detailed steps were taken. We defined our first goal as follows:

**Goal 1:** Comparing the type and number of steps needed to create and edit graphical models to the type and number of steps needed to create and edit its corresponding ImitGraph, to understand different interaction behaviors between ImitGraphs and graphical models.

To supplement the goal, a research question (RQ) and several sub-questions were defined. Based on these questions, a hypothesis was specified.

**RQ 1:** Is the process of creating and editing ImitGraphs comparable to the process of creating and editing the actual graphical model?

Table 3.1: Proposed Plan

Proposed Week	Goals	Actually Completed
W1-W2	Studying the relevant literature. Familiarizing with the existing ImitGraph source code and the development environment for making changes.	04. October 2017 - 15. October 2017
W2-W3	Designing the test cases, the experiments and deciding what should be measured during the experiments. Specifying the functional requirements of the tool based on the design of the experiments.	16. October 2017 - 5. November 2017
W4-W6	Implementing the requirements that have been specified in the previous step.	30. October 2017 - 27. December 2017
W7-W8	Performing test runs with different scenarios, gather hypothetical data and making sure that the gathered data is sufficient for analysis.	13. November 2017 - 26. November 2017
W8-W9	Adapt the design of the experiments and the tool based on the findings of the previous step. Looking for participants in parallel.	13. November 2017 - 26. November 2017
W10	Contacting participants and designing the setup of the experiments.	27. November 2017 - 3. December 2017
W11-W14	Conducting experiments and analyzing the results. Starting with thesis writing.	4. December 2017 - 17. December 2017
W14-W16	Creating the necessary documentation and writing the thesis.	18. December 2017 - 30. January 2018

- RQ 1.1: Are the task completion times for ImitGraph tasks and other graphical model tasks comparable, given a similar task description?
- RQ 1.2: Are similar types of errors made with ImitGraphs as with the corresponding graphical model?
- RQ 1.3: Are similar types and numbers of steps taken between an ImitGraph and its corresponding graphical model?

**Hypothesis H<sub>1</sub>:** The type and numbers of steps taken for creating and editing do not differentiate between ImitGraphs and its corresponding graphical model.

Based on Goal 1, the research questions, and the hypothesis we designed our first controlled experiment. Experiment 1 should compare the interactions between ImitGraphs and the corresponding graphical models by a single group of experienced participants. Therefore, it is desired to show that the number and types of steps taken between graphical models and ImitGraphs do not differentiate by means to the metrics stated in Section 3.3.

Furthermore, we defined a second goal together with its corresponding research questions and hypothesis as follows:

**Goal 2:** Comparing the performance of novice and experienced participants to know if we can use novice participants in usability tests that are based on ImitGraphs.



**RQ 2:** Can usability tests with ImitGraphs be performed by a wider range of participants that have no prior knowledge of designing graphical models?

- RQ 2.1: Is the process of creating and editing ImitGraphs comparable between novice and experienced participants?
- RQ 2.2: Is the task completion time of novices similar to the task completion time of experienced participants?
- RQ 2.3: Are more errors made by novice participants than by experienced?
- RQ 2.4: Are novice participants taking similar actions compared to experienced participants when working with ImitGraphs?
- RQ 2.5: Can a learnability be observed in terms of getting more efficient over time?

**Hypothesis H<sub>2</sub>:** Novice participants perform similarly to experienced participants when creating and editing ImitGraphs.

Following Goal 2, Experiment 2 was designed to test differences in the interaction behavior with ImitGraphs between experienced participants (participants that are familiar with creating graphical models) and novice participants (participants with no prior knowledge of creating graphical models).

### 3.3 Variables

**Controlled Experiments** Controlled experiments combine scientific methods and the use of experimentation [22]. The main goal of controlled experiments is to show that a change to the value of one variable (independent variable) has a real significant impact on the value of another variable (dependent variable) [2]. Thus, before defining how the controlled experiments should be conducted, we have to define the independent and dependent variables.

**Independent Variables** The independent variables are the variables for which the effect should be evaluated. In our first experiment, the independent variables correspond to the different graphical models we used in the experiment. Overall six different graphical models were used: Activity Diagram, Entity Relationship Diagram (ERD), Class Diagram and the three corresponding ImitGraph diagrams (one for each original graphical model).

In the second experiment the independent variables correspond to the different groups of participants: Novice participants and experienced participants.

**Dependent Variables** The dependent variables used in both experiments are connected to the sub-questions defined in Section 3.2. For the first and second experiment we measured the time each participant needed to complete the task. It is important to point out, that the task time includes the process to analyze the task description and the time to complete the task itself. In addition, the actions taken (adding, deleting, connecting, editing, moving and undoing) by the participants during the tasks were logged by the tool the participants were using and were later evaluated. Together with the actions taken, we analyzed the different behaviors of manipulating the graphical models and ImitGraphs based on the screen recordings that were made during the tasks.

The various types of errors during the task and at the end of a task were noted and analyzed after the experiments. An error during the task means, that the participant took actions that were

not allowed based on the description of the model in use. If the participant realized making such an error and undid the error, it did not count as an error in the final submission of the task. We evaluated these errors to get a better insight into the creation process of graphical models and ImitGraphs.

The learnability was measured by analyzing an efficiency change of participants over time. Then we compared the efficiency participants had in their last task to the efficiency the average participant had in this specific ImitGraph task. The efficiency is based on the number of steps an omniscient participant would need to complete the task divided by the actual steps taken by a participant.

### 3.4 Participants

To conduct both experiments, two different groups of participants were recruited. One group with and one without knowledge of designing graphical models for software engineering. We decided to recruit 18 participants overall due to the available time and effort associated with recruiting and conducting the experiments.

For the first experiment, 12 (of the overall 18) experienced participants were chosen. All of them were computer science college students between the 7<sup>th</sup> and 11<sup>th</sup> semester at the University of Zurich. These 12 participants were all familiar with the three graphical models (Activity Diagram, ERD and Class Diagram) used in the tasks due to their education. In their education, those participants worked with all three stated graphical models in several lectures and are familiar with the type of commands given in the tasks. Thus, participants are chosen from an appropriate user population that can be compared to modeling experts. Since they knew how to design such graphical models they are further mentioned as experienced participants. Six of the described 12 experienced participants were also included in the evaluation of the second experiment, due to the order in which they completed the tasks. These six participants did not have to complete more tasks.

For the second experiment, we recruited six novice participants, to the six mentioned experienced ones above. Novices were in the same age (22-27 years old) as the experienced participants and were also students out of different courses, with respect to two exceptions that were not students. To prove that the participants really did not know how to design graphical models, we asked the participants about their knowledge in advance. Participants therefore were chosen from a user population with no prior knowledge of designing graphical models.

All participants were able to speak German and English so that the experiments could be held in German but the task descriptions handed out to the participants were written in English. The participants volunteered for the experiment but received a little gift after participation.

### 3.5 Apparatus

The experiment was conducted in two different rooms at the University of Zurich. Both rooms were set up similar and distraction has been blocked. Participants were placed in front of a notebook next to the observer. Therefore the observer was able to see the screen during the tasks and could make additional notes. The notebook used was a 14 inch Asus VivoBook S400CA running Windows 10. Participants could use a USB mouse connected to the notebook to ease the tasks they were executing.

To record the screen activity during the tasks an additional free screen recording tool was used. Namely "Free Online Screen Recorder" by Apowersoft. This tool allowed us to review all the conducted experiments at a later stage to analyze the participants' behaviors with the graphical models. The two questionnaires participants had to fill in were designed with "Google Forms" and opened before participants started the tasks.

The tool used to conduct the experiment is described in Section 1.4. The tool ran on Google Chrome Incognito mode as a local application. The different tasks were set-up before participants entered the room to conduct the experiment. Each task was opened in a separate tab of the browser so that participants could switch to the next task after finishing the previous one. An example screen-shot of the set-up a participant had before taking the first task is provided in Figure 3.1. One can see the trial for ImitGraphs in the opened tab, the three different tasks for ImitGraphs in the adjacent tabs and the ImitGraphs questionnaire in the last tab.

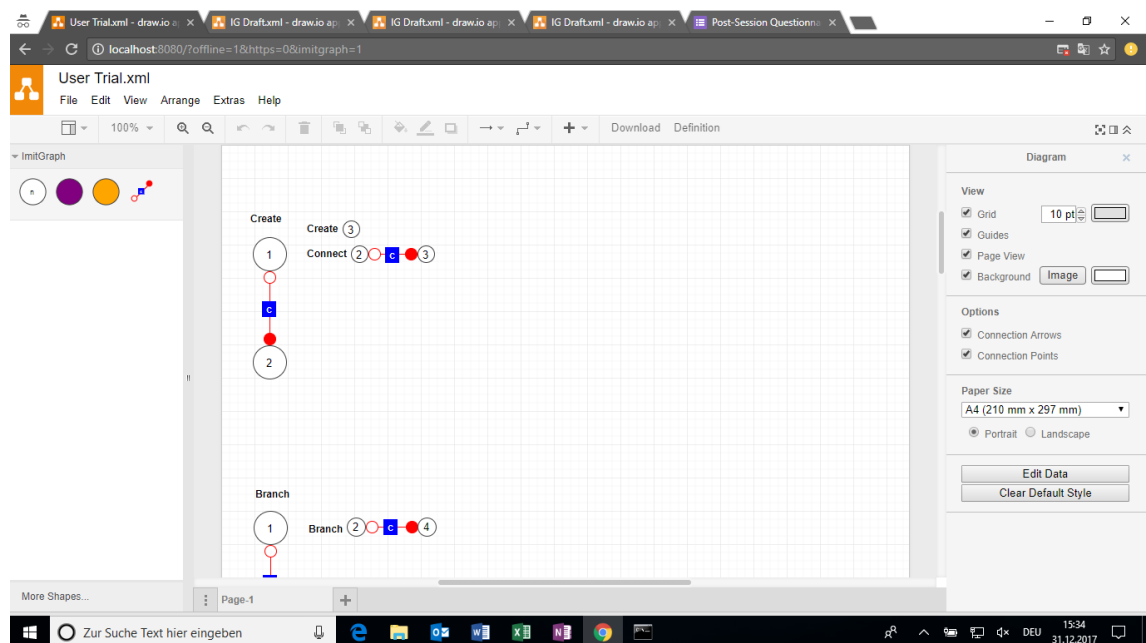


Figure 3.1: Set-up view as seen by participants

## 3.6 Procedure

### 3.6.1 General Procedure

After the participants were seated in front of the notebook, the observer told them about the different tasks they had to complete, what they had to expect and that the screen is recorded. It was also stated that the participants do not have to rush through the tasks, but should feel confident about their final submission of each task. Before each task, the participants received a printed-out task description for that specific task. With the task description participants had to change a given graphical model or ImitGraph with the provided tool on the notebook. The task

completion time was measured from handing out the task description until the participant stated, that he finished the task. After the task was finished the observer downloaded the log file with the actions taken as well as the graphical model the participants submitted. Then the next task description was handed out so that the following task started.

During the tasks the participants were allowed to ask questions regarding the graphical models or ImitGraphs. General answers were given but not the direct solution (right connection or node insertion) for the specific problem itself. If asked about the experiment and its reason, the participants were kindly asked to wait until finishing the experiment. At the end of an experiment, experiment details were shared with the participants. Breaks between the different tasks could have been taken, but never were.

In this thesis we conducted two controlled experiments. Since the experiments had overlapping tasks and all tasks of Experiment 2 were also tasks of Experiment 1, we first describe Experiment 2 in the following section.

### 3.6.2 Procedure of Experiment 2

The second experiment should test whether different interaction behaviors with ImitGraphs exist between experienced and novice participants. Therefore, six experienced and six novice participants received the same three ImitGraph tasks. Each task started from a finished, correct ImitGraph that had to be modified by the participants according to the given commands. The three ImitGraphs were designed to imitate Activity Diagrams, Entity Relationship Diagrams (ERDs) and Class Diagrams. To fulfill the requirements of those three graphical models we defined the rules for each ImitGraph based on the elements used in the original models. The rules consisted of the definition for nodes and connections described in Section 1.3.1. According to these rules the palette of possible nodes and connections was built. Three tables (definition of an ImitGraph) combined all the rules for such a specific ImitGraph.

After executing the three tasks participants were asked to fill in a questionnaire about the usability of ImitGraphs. Participants were asked to rate the ImitGraphs rather than the tool they worked with. The questionnaire had eight mandatory rating questions (scale from 1-7) and two voluntary open-end questions about ImitGraphs. It is based on ASQ techniques for post-task readings and contains questions from the Computer System Usability Questionnaire (CSUQ) as stated by Albert and Tullis in [21, p. 139]. Furthermore, it also contains questions regarding the participants and their experience with drawing graphical models and the tools used for it. The full questionnaire can be found in Appendix A.4.

#### ImitGraph Trial

Before taking the actual tasks, the observer explained ImitGraphs to the participants in detail. Each participant received the same explanation that took around ten minutes, regardless the experience of the participant. For a better explanation the observer handed out the ImitGraph description on paper and clarified ImitGraphs based on this handout. Together with the handout we provided the participants a test set-up for ImitGraphs in the tool used for the experiments. Figure 3.1 also shows the test set-up for participants before they started the tasks on ImitGraphs. In this trial participants could experiment with the tool and ImitGraphs while the explanation took place. If participants had any questions during the trial, they were answered accordingly. Participants were allowed to keep the trial handout during the tasks, to look up commands and examples.

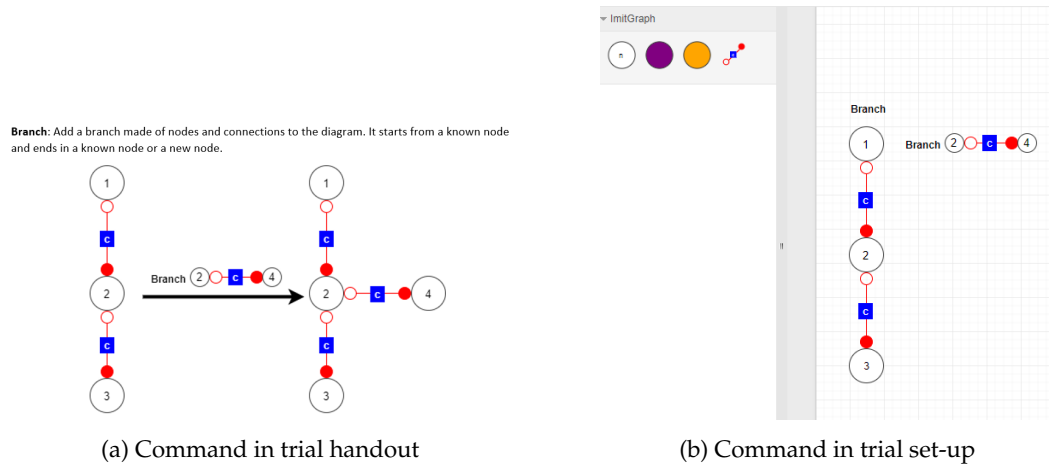


Figure 3.2: Explanation of the "Branch" command in the handout and tool

The provided description included the definitions for elements of an ImitGraph and how an ImitGraph can be specified. Thus, it also contained the specification for the ImitGraph used in the trial. Together with the description and the test set-up, all possible ImitGraph commands (see Section 1.3.2) were explained to the participants. Each command given in the handout could be tested with the tool in parallel and provided the participants a better understanding of the commands. The handout showed the initial state before executing a command, the command itself and the desired outcome after executing the command. Figure 3.2 shows an example of the command in the handout and how it looked in the trial set-up on screen. The whole handout can be found in Appendix A.1. After explaining ImitGraphs, the participants were asked if they wanted to move on with the tasks or rather experiment with ImitGraphs any longer in the provided test set-up.

## Tasks

The participants had to complete three tasks about ImitGraphs. All tasks were designed to simulate the tasks of Experiment 1 for the three graphical models the ImitGraphs should imitate. Each task started with handing out the task description to the participant. This task description included the ImitGraphs definition and the commands the participant had to execute in the given order. Figure 3.3 shows the task description handout for the Activity Diagram ImitGraph task with the ImitGraphs definition on the first page (Figure 3.3a) and the commands on the second page (Figure 3.3b). The initial ImitGraph that should be manipulated according to the given commands was set-up in the tool and ready to be edited by the participants. This initial ImitGraph was always correct and completed

After handing out the task description, the participants were free to decide when to start editing the given ImitGraph. The goal for each participant was to correctly execute the commands in the given order so that a correct ImitGraph (based on the ImitGraphs definition under test) should be the result. Figure 3.4a shows the initial situation for the Activity Diagram ImitGraph task and one possible outcome - if executing the tasks of Figure 3.3b in the given order - can be seen in Figure 3.4b.

Participants could use any technique to insert, edit or delete nodes and connections they wanted to use and the tool allowed. Errors when connecting nodes wrongly were indicated by

## ImitGraph Task

### ImitGraph Task Description

Please edit the given ImitGraph according to the definition and commands below. If you have any questions during the task, feel free to ask.

### ImitGraph Definition

Joint Types		
Type	Symbol	Label
J1		no
J2		no

Connection Types					
Type	Symbol	Label	First Joint	Second Joint	Orientation
C1		optional	J1	J2	any
C2		no	J1	J1	horizontal

Node Types						
Type	Symbol	Label	Connection Type	Joint Type	Min	Max
N1		no	C1	J1	1	1
N2		no	C1	J2	1	1
N3		yes	C1	J1	1	1
N4		no	C1	J2	0	1
N5		no	C1	J1	0	1
N6		no	C2	J1	1	2
			C1	J2	1	1
			C1	J1	2	2
			C1	J2	1	1
			C1	J1	2	3
			C1	J2	1	1

(a) ImitGraph specialization

### Task Commands

- Find Node
- Find Node
- Delete current location
- Find Node
- Delete current location
- Connect
- Connect
- Find connection
- Remember as L1
- Find connection
- Remember as L2
- Insert L1

(b) Task commands

Figure 3.3: Activity Diagram ImitGraph task description handout

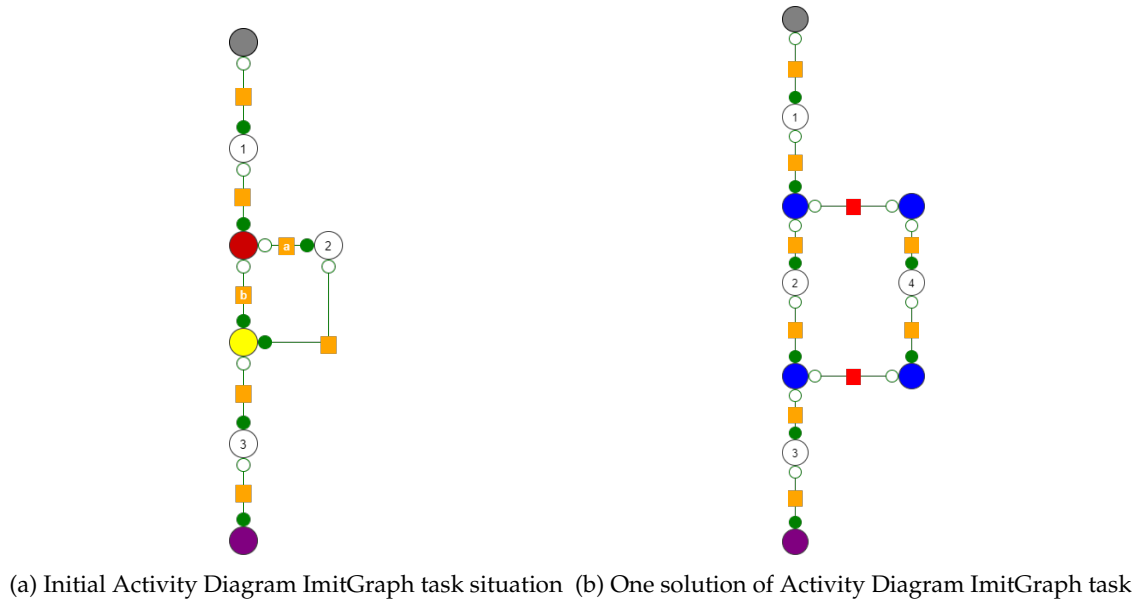
the tool itself but not by the observer. Any insertions or deletions that would lead to a wrong outcome were counted as errors during the creation of the ImitGraph, but should not be mistaken as errors that still consists when participants stated out, that they were finished. The errors still existing in the submission were counted after the experiment and only told the participants if asked.

All the tasks, ImitGraph specifications used, initial situations and desired outcomes can be found in Appendix A.2.

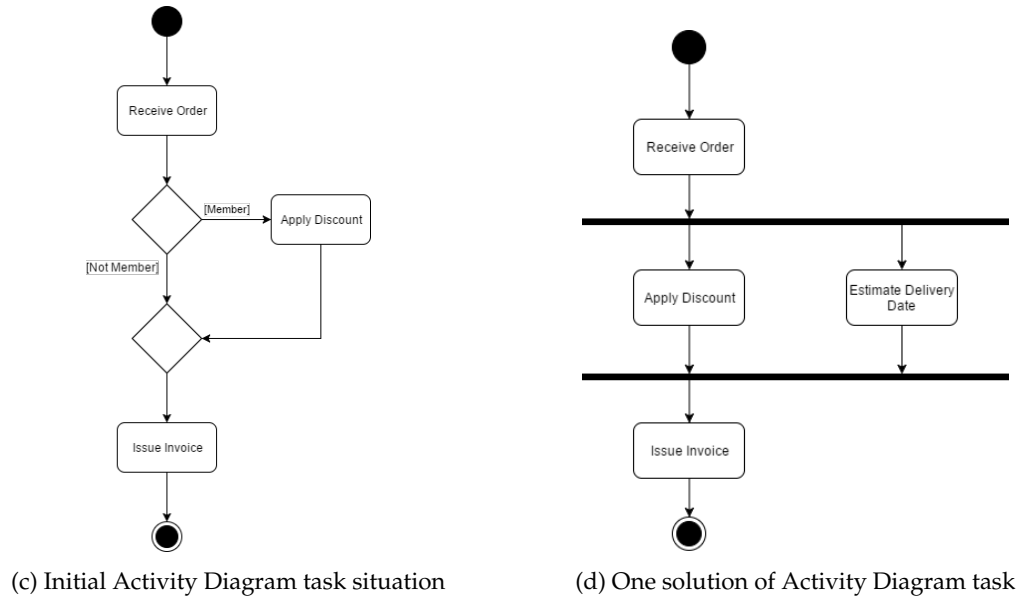
### 3.6.3 Procedure of Experiment 1

The first experiment should test different interaction behaviors when working with ImitGraphs or their corresponding graphical models. 12 experienced participants were chosen and conducted the experiment. Overall, they received six tasks of which three are the tasks of Experiment 2 described in Section 3.6.2. The remaining three tasks covered the three corresponding graphical models to the different ImitGraphs (Activity Diagram, Class Diagram and Entity Relationship Diagram). Therefore, the Activity Diagram ImitGraph task was designed on the Activity Diagram task. Task layouts and the commands given were similar between the graphical models and the corresponding ImitGraphs. This similarity allowed us to compare the steps taken to manipulate an ImitGraph to the steps taken to manipulate the corresponding graphical model, so that always one ImitGraph could be compared to its original graphical model.

As in Experiment 2, a task description with the elements of the graphical model and the commands were handed-out before each task. The handout for the Activity Diagram task can be seen in Figure 3.5. Participants had to manipulate the given graphical model based on the handed out commands and the rules known for the graphical model. The commands were only textual and based on commands experienced participant knew from their education. The initial state of the Activity Diagram task and one possible solution can be seen in Figure 3.4c and 3.4d. After fin-



(a) Initial Activity Diagram ImitGraph task situation (b) One solution of Activity Diagram ImitGraph task



(c) Initial Activity Diagram task situation (d) One solution of Activity Diagram task

Figure 3.4: Initial situation and one possible outcome of Activity Diagram tasks

ishing all three graphical model tasks participants were asked to fill in a questionnaire regarding these graphical models. The questions were the same as the questions in the questionnaire for ImitGraphs. In Appendix A.4 the full questionnaire about graphical models can be found.

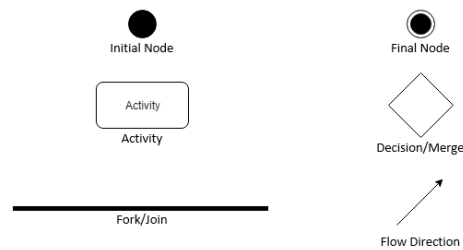
Tasks were grouped to their affiliation, meaning that either all the ImitGraph tasks or all tasks about the graphical models were conducted first. Participants did not receive an introduction into the graphical models' part since experienced participants knew about the rules of these graphical models.

## Activity Diagram Task

### Activity Diagram Task Description

Please edit the given Activity Diagram according to the commands below. If you have any questions during the task, feel free to ask.

### Activity Diagram Elements



### Task Commands

- Delete decision after activity "Receive Order"
- Delete decision after activity "Apply Discount"
- Connect activity "Receive Order" to activity "Apply Discount"
- Connect activity "Apply Discount" to activity "Issue Invoice"
- Add parallel activity to activity "Apply Discount" called "Estimate Delivery Date"

Figure 3.5: Activity Diagram task description handout

A possible scenario for one participant of Experiment 2 therefore looked like stated below (Scenario for PE1 and PE2 as seen in Table 3.2):

1. Graphical model tasks
  - 1.1. Task: Activity Diagram (Handout: Figure 3.5, Initial Situation: 3.4c)
  - 1.2. Task: Entity Relationship Diagram
  - 1.3. Task: Class Diagram
  - 1.4. Questionnaire: Graphical models
2. ImitGraph tasks
  - 2.1. Explanation and Trial: ImitGraphs
  - 2.2. Task: Entity Relationship Diagram ImitGraph
  - 2.3. Task: Class Diagram ImitGraph
  - 2.4. Task: Activity Diagram ImitGraph (Handout: Figure 3.3, Initial Situation: 3.4a)
  - 2.5. Questionnaire: ImitGraphs

All task descriptions as well as initial states and possible outcomes of the graphical models can be found in Appendix A.3.

## 3.7 Experiment Design

Experiment 1 was chosen to be a within-subject design, where each experienced participant conducted every single task for both conditions. The conditions for the first experiment were the



two types of models, namely ImitGraphs and the corresponding graphical models. Thus, individual differences are less likely to influence the results [2]. Experiment 2 was designed to be a between-subject design, where results from different participants were compared for the same tasks.

Both experiments were counterbalanced using a Latin square as described in [2]. Always two participants received the tasks in the same order. For the first experiment, the corresponding graphical models and ImitGraph tasks had a different order, so that the similarity could not be detected by participants that easily. The Latin square used for the experiments is seen in Table 3.2. In the table the tasks are sorted where T1-T3 stand for the graphical model tasks and tasks T4-T6 stand for the ImitGraph tasks. Abbreviations below the task numbers stand for the models used (AD = Activity Diagram, ERD = Entity Relationship Diagram, CD = Class Diagram, IG = ImitGraph). Numbers indicate the specific order in which the participants executed the tasks. Participants (experienced) PE7-PE12 were included in the evaluation for both experiments, since they did the ImitGraph tasks first, like novice participants (PN) did.

Table 3.2: Counterbalanced task order using a Latin Square

Participant	T1	T2	T3	T4	T5	T6
	AD	ERD	CD	IG AG	IG ERD	IG CD
PE1	1	2	3	6	4	5
PE2	1	2	3	6	4	5
PE3	3	1	2	5	6	4
PE4	3	1	2	5	6	4
PE5	2	3	1	4	5	6
PE6	2	3	1	4	5	6
PE7	6	4	5	1	2	3
PE8	6	4	5	1	2	3
PE9	5	6	4	3	1	2
PE10	5	6	4	3	1	2
PE11	4	5	6	2	3	1
PE12	4	5	6	2	3	1
PN1	-	-	-	1	2	3
PN2	-	-	-	1	2	3
PN3	-	-	-	3	1	2
PN4	-	-	-	3	1	2
PN5	-	-	-	2	3	1
PN6	-	-	-	2	3	1



# Results and Discussion

In the following sections, the results of both experiments are discussed. All data were collected by the logging environment of the tool itself and the data from the questionnaires. Additionally, we used material from the screen recording tool for the behavior analyses. To convert the data from the logging tool (JSON) into an excel-readable file a Python program was written. For the data analysis, Excel was used together with the add-on "XLSTAT". "XLSTAT" allowed us to use a wider variety of statistical functions needed to analyze the obtained data. Such functions include the tests for normality and non-parametric tests to compare samples. Since we followed two different experiments, this chapter is partitioned in Experiment 1 and 2. Further, the sections are subdivided into the dependent variables and research questions mentioned in Section 3.2.

## 4.1 Experiment 1

The goal of Experiment 1 was to test if participants have a similar interaction behavior with ImitGraphs as with the corresponding graphical models. Therefore, we observed the task completion times, actions taken, and errors made during the tasks by each participant. We compared a specific ImitGraph task to the task of the corresponding graphical model. So that the Activity Diagram task is compared to the Activity Diagram ImitGraph task and so on.

### 4.1.1 Task Completion Time

Before analyzing the time data, we had to check the data for normality so that we could use parametric tests. Because we only had 12 participants we wanted to use parametric tests since they are more efficient and require fewer data points than non-parametric tests [3]. Therefore, the Shapiro-Wilk test for normality was applied to the data. Results of the test showed that the times for four of the six tasks are indeed distributed normally. But it showed, that the times for the Class Diagram ImitGraph task and the ERD ImitGraph task were not. Regarding this the box-plot in Figure 4.1a was created. In the box-plot we can see the reasons for the non-normal distribution. Two outliers can be identified, one for both of the tasks that had a non normal distribution. Both outliers were checked for wrong measuring or other unforeseen influences, but none were found. The specific participants just needed more time for those tasks. Out of this reason, we decided to keep the outliers and use non-parametric tests as well as parametric tests for the comparison of the Class Diagram and ERD tasks and their corresponding ImitGraph tasks. Both tests were chosen due to the loss of information when using a non parametric test [14, pp. 223-227].

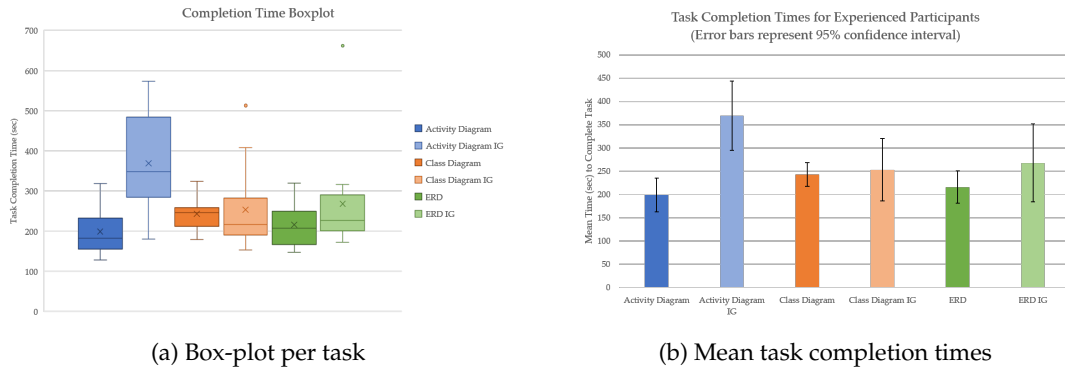


Figure 4.1: Box-plot and means of task completion times of Experiment 1

Descriptive statistics on the task completion times can be seen in Table 4.1, which shows the important values for each task. The standard deviations vary from 39 to 131 seconds which is rather high and can be explained by a higher variability in the task completion times and by the observed outliers. It also shows that the standard deviation is higher for all the ImitGraph tasks, meaning that the task completion times of participants varied more in the ImitGraph tasks. It can be seen that the means of the Activity Diagram tasks are further away from each other (170 seconds) as with the other tasks (Class Diagram 10 seconds, ERD 52 seconds), even though we have outliers for the latter tasks. Also, is the minimum of the Activity Diagram ImitGraph task close to the mean of the Activity Diagram task, which means that there probably is a difference in the task completion time between the Activity Diagram task and its ImitGraph task. What can be seen, is that the mean times for the ImitGraph tasks are always higher compared to the tasks of the corresponding graphical model. In Figure 4.1b task mean times are visualized with a 95% confidence interval.

Table 4.1: Descriptive statistics of task completion times (sec) of Experiment 1

Task	Min	Max	Mean	Standard Deviation
Activity Diagram	128	319	199	58
Activity Diagram IG	180	573	369	117
Class Diagram	179	324	243	39
Class Diagram IG	153	513	253	105
ERD	147	320	216	55
ERD IG	172	662	268	131

Comparing the task times directly to each other for each individual participant, we could see that only one participant was faster in the Activity Diagram ImitGraph task than in the Activity Diagram task. But in the Class Diagram tasks six and in the ERD tasks three participants were faster in completing the ImitGraph task. That means we cannot directly say if the task completion time of the latter two diagrams differ from its ImitGraph task completion time and therefore need more tests.

**Significant Differences** For further analyses with parametric and non-parametric tests a preset level of significance was set. We decided to use a 95% level of confidence which means  $\alpha = 0.05$ . To test if there is a significant difference in task mean times of two corresponding tasks we used a paired t-test for the Activity Diagram tasks, since the data is normally distributed and we are comparing means within the same participants. Using this test, we received a high variance and a  $p$ -value of 0.00017, which is considerably smaller than 0.05. We therefore have a significant difference in the task completion times between the Activity Diagram task and the corresponding ImitGraph task.

The other tasks were tested for significant differences with the non parametric Wilcoxon Signed-Rank test [14, p. 215] and the parametric t-test. Comparing the Class Diagram tasks, we received a  $p$ -value of 0.677 for the Wilcoxon Signed-Rank test and a  $p$ -value of 0.73 with the t-test. Both  $p$ -values are higher than the preset  $\alpha$  level and therefore the Class Diagram tasks have no significant difference in the mean task times. Same applies comparing the ERD tasks completion time with  $p$ -values of 0.23 for both tests.

Therefore, no significant differences in the task completion times could be found between the graphical models and ImitGraphs tasks, except in the Activity Diagram tasks. Meaning that the completion time of an ImitGraph task does not differ significantly from the task completion time of the corresponding graphical model.

**Equivalence Testing** To test the task times on equivalence a TOST (two-one-sided-test) was conducted. The TOST requires normal distributed data sets, so that outliers were eliminated for this test. We set the bounds of equality to  $\pm 60$  seconds, since we thought a minute difference in task completion time is similar enough when working with a new diagram for the first time. The  $\alpha$  level was set again to 0.05. So that to fulfill the equivalence tests, the 90% confidence interval has to be included in the TOST interval.

Running the tests yield the results in Table 4.2a for the Class Diagram tasks and Table 4.2b for the ERD tasks. It is seen that both tasks are equivalent regarding a  $\pm 60$ sec boundary.

Table 4.2: TOST results of task completion times of Experiment 1

(a) TOST on Class Diagram tasks		(b) TOST on ERD tasks	
Test	Value	Test	Value
Lower bound (TOST)	-60.000	Lower bound (TOST)	-60.000
Lower bound (90 %)	-26.616	Lower bound (90 %)	-52.597
Upper bound (90 %)	53.509	Upper bound (90 %)	19.900
Upper bound (TOST)	60.000	Upper bound (TOST)	60.000
Test interpretation	Equivalent	Test interpretation	Equivalent

**Thinking Before Executing** In the post-session questionnaires, we asked the participants to rate how much time they needed to think about the commands before executing them. Each participant had to answer the question once for ImitGraphs and once for the graphical models. The rating was based on a seven-point interval scale, where seven stands for a reasonable amount of time and one for no thinking before executing at all. Results showed that participants had the perception of needing more time to think about the commands in the graphical model tasks

(Mean = 3.5, Median = 4) compared to the commands of the ImitGraph tasks (Mean = 4.7, Median = 5). Running a Wilcoxon signed-rank test on the results showed that there is indeed a significant difference in perceived time of thinking before executing the commands in the different tasks, ( $p$ -value = 0.016) for a 95% confidence level. We could not observe this difference in the actual task times, since participants mostly started with drawing the models immediately.

## Task Completion Time Discussion

From the various results it can be seen, that there are differences but also similarities in the task completion times between ImitGraphs and the corresponding graphical model. Considering the Activity Diagram tasks, a significant difference in the task completion times between the two diagrams could be observed. Meaning that the ImitGraph task took the participants more time for completion. By checking the screen capturing and analyzing the tasks, we found two possible reasons for the difference in task completion times of the Activity Diagram tasks stated below:

1. The Activity Diagram task required less adding and connecting of elements than the corresponding ImitGraph task. Instead of inserting two "Fork/ Join" elements into the Activity Diagram, participants had to add four nodes and two connections into the ImitGraph Activity Diagram. Resulting in more steps to be taken which requires more time to complete a task. Since each of these connections had to lie horizontal and connected nodes at specific positions, lots of participants were struggling with the adding and thus the ImitGraph task could have taken more time.
2. The "Inserting" command in the ImitGraph task might be too difficult for participants. The command can be found in Figure 3.3b and is the last command of the Activity Diagram ImitGraph task. This command requires to insert lots of nodes in a parallel branch to the already existing diagram and might not be as intuitive as other commands. Participants had difficulties with this command, even though a similar example was presented in the trial. Troubles with this command lead to a higher task completion time as it can be seen in the results.

Considering both possible reasons we tend to say that a different specification of the Activity Diagram ImitGraph, or a slight variation of the palette could lead to closer task completion times. Also, if the inserting command of the ImitGraph task was split up into shorter commands, participants could be faster in the completion, since the tasks would be more intuitive.

The Class Diagram tasks and ERD tasks were shown to be similar within a  $\pm 60$ sec boundary. The mean times are close to each other and even closer if outliers are eliminated. Outliers may be explained by the fact that the specific tasks were the first tasks on ImitGraphs for the two participants who produced the outliers. That six participants completed the Class Diagram ImitGraph task faster than the Class Diagram task might due to the ease of the specified ImitGraphs and intuitive commands, that needed less time in thinking before executing. This assumption is based on the different commands given for graphical models and ImitGraphs. Commands are partially visual for ImitGraphs and participants therefore already know which elements have to be added or deleted. On the other side, participants have to think about the elements that should be added to the graphical models and how they should be connected. This thinking takes place before starting and during executing the command, what could require more time to complete a task.

Based on the observed results, we conclude that if the ImitGraph tasks afford a similar number of steps to be taken as the corresponding graphical model task does, then the task completion

times are equivalent (to a certain boundary) between the two tasks. Thus, we can answer research question RQ 1.1 and say that the task completion times between ImitGraphs and their corresponding graphical models are indeed similar given a similar task description.

In the post-session questionnaires, we asked the participants to rate how quickly they could complete the tasks with the different models based on their perception. Participants stated for both model types that they were rather quickly with completing the tasks and no significant differences between the perceived task completion time on the different models could be found. This also reinforces the assumption that the task completion times are indeed similar between an ImitGraph task and the corresponding task on the graphical model.

### 4.1.2 Steps Taken and Interaction Behavior

During the experiment, the tool logged the steps taken such as adding, deleting or connecting different elements for each task. Together with the log, we noted additional steps taken with the recordings of the screen capturing tool. As with the task completion times we were using parametric and non-parametric tests to show differences between the corresponding tasks. Also, we were taking a closer look at specific participants and how they interacted with the graphical models compared to the corresponding ImitGraph.

#### Overall Number of Steps

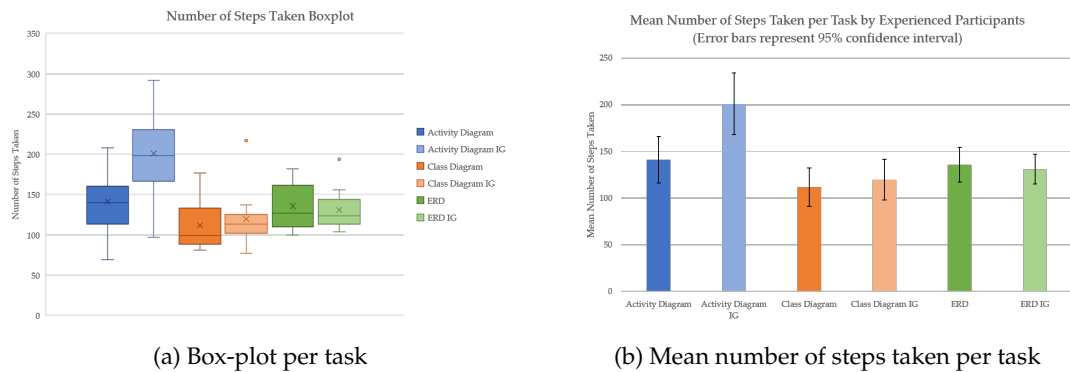


Figure 4.2: Box-plot and means of number of steps taken in Experiment 1

The overall number of steps a participant took to edit the given task are composed of adding, connecting, selecting, editing, moving, deleting and undoing elements. Participants could choose their preferred way of how to complete one of the mentioned actions. Table 4.3 shows the descriptive statistics about the overall number of steps taken for each task of Experiment 1. It can be seen that the means of steps taken to edit the given Activity Diagram and the corresponding ImitGraph differ in 60 steps. This difference is higher compared to the other tasks with an eight-step difference in means for the Class Diagram tasks and a five-step difference for the ERD tasks. We can also see, that on average fewer steps were taken to edit the ERD compared to the ERDs ImitGraph. Also, the median is lower for the ERD ImitGraph task as for the ERD task, as we can see in Figure 4.2a. Further, two outliers can be identified in the box-plots. These outliers were produced

by the same participants as the outliers for the task completion times, who needed considerably more time and steps to complete the two tasks.

Table 4.3: Descriptive statistics of number of steps taken in Experiment 1

Task	Min	Max	Mean	Standard Deviation
Activity Diagram	69	208	141	39
Activity Diagram IG	97	292	201	52
Class Diagram	81	177	112	32
Class Diagram IG	77	217	120	34
ERD	100	182	136	29
ERD IG	104	194	131	25

**Significant Differences** Before using tests on the gathered data, we checked it for normality with the Shapiro-Wilk test. The results showed that only the Activity Diagram tasks were distributed normally. We therefore tested the Activity Diagram tasks with a paired t-test but the other tasks with the non-parametric Wilcoxon Signed-Rank test. Results showed that there is indeed a significant difference in the overall steps taken between the Activity Diagram task and its corresponding ImitGraph task for an  $\alpha$ -level of 0.05. The received  $p$ -value was 0.004. Therefore, the ImitGraph task for the Activity Diagram took significant more steps on average than the Activity Diagram task for graphical models.

The other tasks showed no significant difference in the overall steps for an  $\alpha$  level of 0.05. The  $p$ -values were 0.56 for the Class Diagram tasks and 0.75 for the ERD tasks. We thus tend to say that the Class Diagram task and the ERD task do not differ in the number of steps taken to their corresponding ImitGraph tasks. Therefore, it can be said that participants do not take significantly more steps to complete either ImitGraph tasks or the corresponding graphical model task.

The perceived efficiency, stated by participants in both post-session questionnaires, also confirms the assumption that similar number of steps were taken to complete the corresponding tasks. Participants rated their efficiency with the different graphical models on a scale from one to seven. Whereas the mean perceived effectiveness was higher for the graphical models (Mean = 5.3, Median = 6) as for ImitGraphs (Mean = 5.9, Median = 6), we could not find a significant difference between the different models. 33% of the participants rated their perceived efficiency on ImitGraph as high as with the graphical models. To confirm the assumption that similar types of steps were taken, we will take a closer look at the single types of steps taken in the next section.

## Types of Steps

We could see in the overall steps taken that there were significant differences between the number of steps taken for the Activity Diagram task and its ImitGraph task. But both other pairs of tasks did not differ in the overall number of steps taken by participants. To get a better understanding where differences and similarities have its source, we take a deeper look at each pair of tasks.

**Activity Diagram Tasks** A significant difference in the overall number of steps taken for the Activity Diagram tasks could be observed. This again lies in the specification of the Activity Diagram ImitGraph as stated in Section 4.1.1. Since participants had to add two nodes and two connections



more to the given ImitGraph there must be a difference. By checking the mean number of added nodes, we can actually see this difference. On average participants added nine elements (nodes or connections) to the Activity Diagram whereas they added 13 elements to the Activity Diagram ImitGraph.

This alone would not make such a big difference. But if more connections and nodes are added also more selecting, editing, moving and connecting must be done to connect the connections to nodes. Thus, the additional elements lead to an accumulation of extra steps that needed to be taken by each participant.

**Class Diagram Tasks** For the Class Diagram tasks, we could see a picture, task time was not showing on first sight. Participants added nine elements to the Class Diagram and eleven to the corresponding ImitGraph on average. The reason is a similar one as to the Activity Diagram task. Participants had to add one node and one connection more to the ImitGraph compared to the graphical model. If it was not for those two elements participants would have added the same number of elements to both diagrams on average. One part of the remaining seven-step difference in the overall number of steps taken can again be explained by the additional connections that had to be made. If one connection more has to be inserted, at least two more steps needs to be taken to connect the connection to two corresponding nodes assigned by the command.

The reason why the task completion time did not show this difference might be due to the fact, that the commands used in the Class Diagrams ImitGraph task were rather intuitive and easy to execute. Resulting that participants did not have to think about the execution of the command as they had to do with the commands for the graphical model. Thus, the task completion times for the Class Diagram tasks were more resembling even though more elements had to be added.

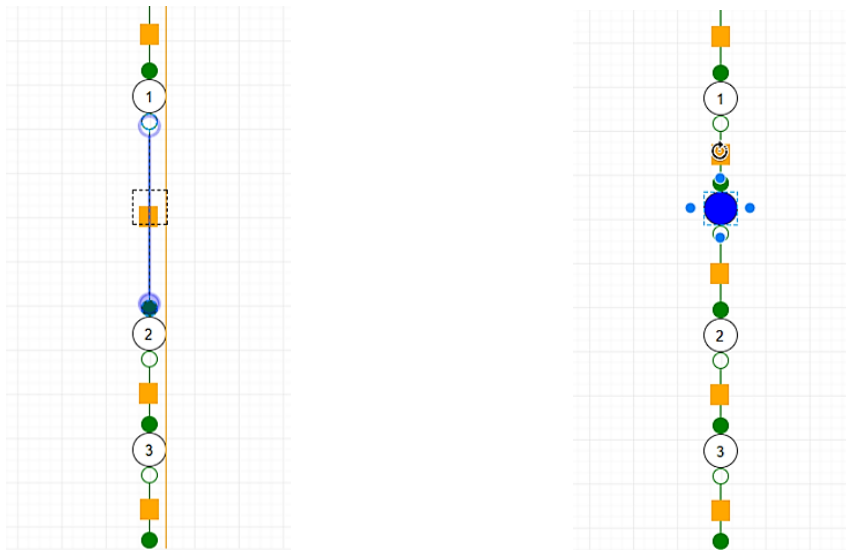
**Entity Relationship Diagram Tasks** The Entity Relationship Diagram tasks were more resembling than the other two pairs of diagrams. For the ERD tasks participants had to add the same number of elements to the ImitGraph as to the graphical model. We could see this in the overall steps taken (five-step difference) and in the adding of elements. Participants added 12 nodes on average for both diagrams. The five-step difference in the overall steps between the ERD tasks can be explained by the number of selections and editing made by the participants. It could be seen that participants selected and edited more nodes in the ERD task than in the corresponding ImitGraph task. But it could not be seen any trend in the difference of selecting or editing nodes and we therefore tend to say that this difference is only due to chance.

## Interaction Behavior

For both ImitGraphs and graphical models, different ways for adding and connecting nodes were possible and participants were free to choose whether method they preferred. Per example participants could connect nodes by attaching the end of a connection to a connection point of the desired element, could drop an element on one end of a connection or could drop elements onto a connection so that the node is connected to two similar connections afterward as it can be seen in Figure 4.3. Many more possibilities how to connect elements were possible.

Participants could also add nodes in different ways. It was possible to add elements by clicking on the element in the palette or drag and drop elements to the diagram. Additionally participants could copy paste elements already used in the diagram.

As we could see from the screen recordings and data from the log tool, there were no big differences in the behavior of a participant when interacting with an ImitGraph or the graphical model. If a participant copy/pasted lots of elements in the graphical model, he also copied a



(a) Before dropping node on connection (highlighted)

(b) After dropping node on connection

Figure 4.3: Example of dropping a node onto a connection

similar number of elements in the ImitGraph task. Same goes for dropping nodes onto a connection. Participants that knew about this feature in the tool, used it in same ways for ImitGraphs as for graphical models. So that the behavior of manipulating an ImitGraph or its corresponding graphical model only depended on the participants' preferences.

Yet there is a difference we want to point out. After reading an ImitGraph command participants often added all the required elements first and then connected them according to the given command. An example of such an insertion technique used by a participant is given in Figure 4.4 for the "Inserting" command already mentioned and visible in Figure 3.3b. We could not see this type of interaction for the graphical models. In the corresponding graphical models mostly one element was added and connected right afterward. How participants connected the elements was again similar, just the order of adding and connecting was different.

From the obtained data we could see that for all ImitGraphs more connections were made than for the corresponding graphical models. And the number of connections made differed significantly between the two corresponding models for the Activity and Class Diagrams. As we explained above, this is also due to the additional connections that had to be connected, but the difference could not be explained alone by this. Only the screen recordings could tell where the difference came from. In the graphical models it was possible to connect elements by dragging out connections from a connection point and connect it directly to another previously inserted element. Some participants used this method to connect elements of the graphical models, but the method could not be used for ImitGraphs. This method requires less manual connections to be made and lead to the observed difference. But we could also observe that participants who used this feature also tried using it with ImitGraphs. By trying out the feature in ImitGraph tasks participants then realized, that ImitGraphs did not support that specific feature. Participants thereupon used other methods to connect nodes in ImitGraphs. Since we could see that participants were trying the feature for ImitGraphs we think that it would have been used by participants for ImitGraphs too, if it was available. The fact that this feature could not be used with ImitGraphs might also explain the lower perceived effectiveness for ImitGraphs compared

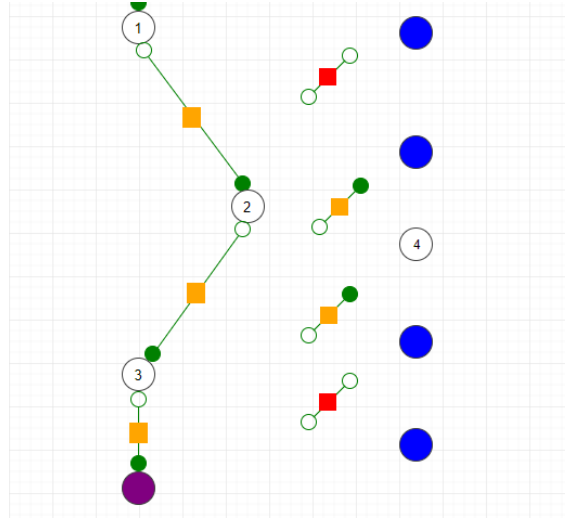


Figure 4.4: Example of adding elements before connecting

to the graphical models as stated by participants in the post-session questionnaires. We already discussed this perceived effectiveness in Section 4.1.2.

In summary, the behavior of a participants when adding, connecting or manipulating elements with ImitGraphs did not differ from the behavior with the corresponding graphical model, if the features were available for both models. Since ImitGraphs lead to similar behaviors like their corresponding graphical models, the steps taken and how they are taken by participants are analogous. This answers research question RQ 1.3 so that indeed similar numbers and types of steps are taken by participants regardless of the used model. We can conclude, that usability tests could be done by using ImitGraphs instead of the original graphical model, considering the numbers and types of steps taken.

### 4.1.3 Errors

Before stating minor errors, participants had at the end of a task, we want to call out that all, but one participant completed the tasks successfully. Meaning that the diagrams they submitted had no major errors except the minor ones stated in this section. In Figure 4.5 the percentages of experienced participants who completed the tasks without any errors can be seen. For the Activity Diagram tasks and the Class Diagram ImitGraph task only one participant each had errors overall.

More participants made errors in the Class Diagram task (five participants) and ERD Diagram task (eight participants). These errors were mostly cardinality errors, meaning that participants mixed up cardinalities and set them at mirrored positions of a connection. For the ImitGraph tasks, this mixing up was not a problem and setting cardinalities was mostly done properly. This is due to the given commands for ImitGraphs, where cardinalities were shown at the desired location in the command itself. Another reason that more cardinality errors were made in the graphical models' part, may be due to the different definitions of the graphical models. Comparing the used Class Diagram and ERD it can be seen, that cardinalities are set mirrored. Participants could have mixed up these definitions and thus made errors. To reduce the error rate of the participants in the ERD and Class Diagram task a refresher about both diagrams might be necessary before

conducting the experiment.

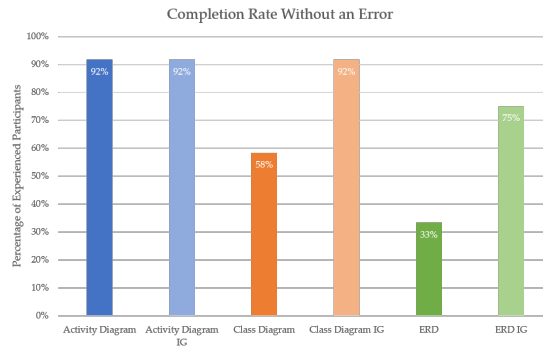


Figure 4.5: Percentage of participants who completed tasks without errors

Another error that could be seen were loose connections. Meaning that elements of the diagrams were not properly attached by connections. This error could be observed for all diagrams and cannot be connected to either ImitGraphs or the corresponding graphical models. It occurs due to the used tool, where connections seem to be attached correctly but in reality they are not. If this was the case and participants did not move the connected node, they could not recognize the error. Whereas some participants realized the error during the tasks others did not and thus had errors in the final submission. Overall six of the twelve experienced participants made this error. Three of them more than once.

The in the beginning of this section mentioned participant who did not complete the Activity Diagram ImitGraph successfully had errors no other participant had. The participant connected nodes in a wrong order after the "Inserting" command of the task. Thus, the task was counted as not successful. But since he inserted as many nodes as other participants and the task completion time was similar to others we did not exclude the participant from the analyses of the previous sections.

Overall, we can say that similar errors are made with ImitGraphs as with the corresponding graphical models based on our observations. Clearly it needs to be said that working with ImitGraphs raised error messages as stated in Section 1.4 and working with graphical models did not. But since participants knew the rules of the graphical models they should not need such error messages. The raised error messages should only be used as indication that nodes and connections might be connected falsely. Nevertheless, research question RQ 1.2 can be answered that similar errors are made with ImitGraphs as with the corresponding graphical models.

#### 4.1.4 Overall Satisfaction

In the post-session questionnaires, we asked the participants to rate the simplicity and how comfortable they are with drawing ImitGraphs and graphical models. Additionally, participants had to state their overall satisfaction with the two models. They could rate the questions on a scale from one to seven, where seven meant "strongly agree" and one "strongly disagree". No significant differences in the answers between the two models could be found. Participants stated that both ImitGraphs (Mean = 5.9, Median = 6) and graphical models (Mean = 6.1, Median = 6) were rather simple to draw. Same goes for how comfortable participants felt while drawing the models

(Mean ImitGraphs = 5.6, Mean graphical models = 5.9) and how satisfied they were (Mean IG = 6, Mean GM = 5.9).

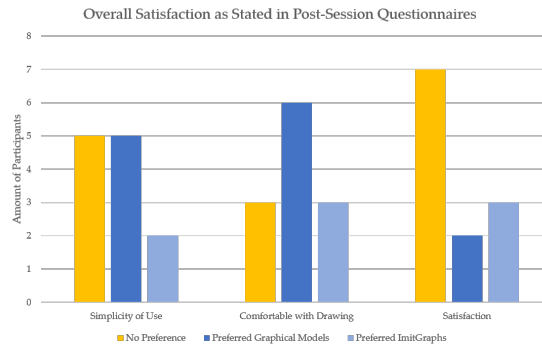


Figure 4.6: Preferences for models as stated by participants

In Figure 4.6 the differences in the preferences can be seen. Five participants had no preferences in the simplicity of using ImitGraphs or graphical models. Whereas five participants stated that using graphical models was simpler than using ImitGraphs. The difference might come from the missing feature for drawing ImitGraph or just by the fact that the participants were familiar with the use of graphical models. The two participants who stated that the use of ImitGraphs was simpler also stated that it was more comfortable drawing ImitGraphs and their perceived overall satisfaction was also higher for drawing ImitGraphs than for drawing graphical models.

We observed that more participants were comfortable with drawing graphical models than ImitGraphs. This again might come from the habit of drawing these graphical models, whereas ImitGraphs were new to all participants. In the overall satisfaction seven out of the twelve participants showed no preference for one of the two models. We therefore think that the overall satisfaction when working with ImitGraphs is similar to the overall satisfaction when drawing graphical models. This supports our assumption that usability tests could indeed be conducted on ImitGraphs instead of the graphical model.

## 4.2 Experiment 2

In Experiment 2 experienced participants and novice participants took the same tasks on ImitGraphs. We studied if there are any different interaction behaviors between those two groups. Therefore, we observed task completion times, numbers and kind of steps taken, and errors made during the tasks for each participant. Comparisons are made between the different groups of participants for one task. To keep the diagrams and tables simpler, we used abbreviations for the different tasks (AD = Activity Diagram, CD = Class Diagram, ERD = Entity Relationship Diagram and IG = ImitGraph).

### 4.2.1 Task Completion Time

First, we checked the task completion times for a normal distribution so that we could use parametric tests. The Shapiro-Wilk test showed that the task completion times are indeed distributed

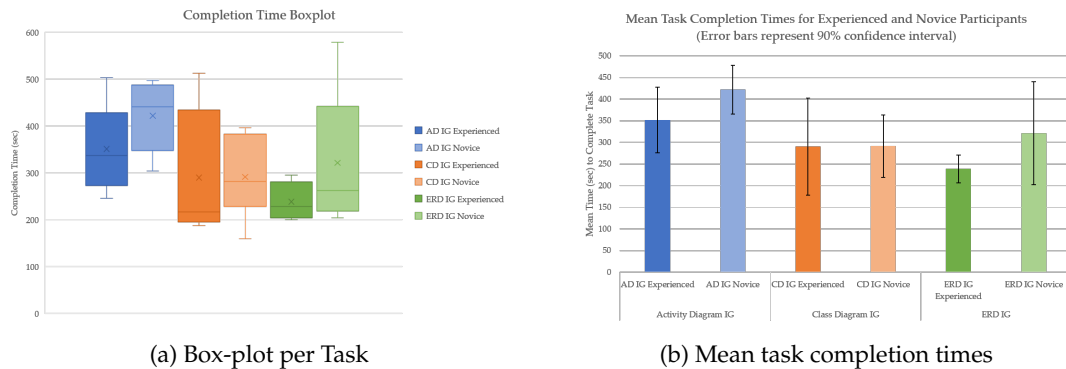


Figure 4.7: Box-plot and means of task completion times of Experiment 2

normally. But since we only had six data points we wanted to check the box-plot in Figure 4.7a first to assure normality of the data. In the box-plot the data does not look that normal anymore. Medians are not in the middle of the box and the whiskers are asymmetrical and are a sign for a non-normal distribution. We do not have any outliers, but the variance of the data is high. We thus decided to use non-parametric and parametric tests to analyze the data further on significant differences between experienced and novice participants.

Using descriptive statistics presented in Table 4.4. We can see the high standard deviations from 38 seconds up to 144 seconds. Reasons lie in the data points that are far away from the calculated means. These data points cannot be treated as outliers since they were correctly measured and are not marked as outliers in the box-plots. Means and medians of the task completion times are always higher for novice participants compared to experienced participants in the same task. But we can also see that at least some novice participants could keep pace with the experienced participants. The mean task completion times are also presented in Figure 4.7b with a 90% confidence interval.

Table 4.4: Descriptive statistics of task completion times (sec) of Experiment 2

	Activity Diagram IG		Class Diagram IG		ERD IG	
	Experienced	Novice	Experienced	Novice	Experienced	Novice
Min	246	304	188	159	200	204
Max	503	497	513	396	295	579
Mean	351	422	290	291	239	322
Standard Deviation	92	68	137	88	39	144

**Significant Differences** As mentioned above we used parametric and non-parametric tests to study the data on significant differences between the groups of participants. We preset the  $\alpha$ -level to 0.05 and thus test on a 95% confidence level. Since the analyzed data is from two independent samples, we used t-tests for two samples and Mann-Whitney tests to compare the task completion times. For the Activity Diagram ImitGraph task we received a  $p$ -value of 0.18 for the t-test and a  $p$ -value of 0.24 for the Mann-Whitney test. The  $p$ -values for the Class Diagram ImitGraph task are 0.98 and 0.82. Whereas for the ERD ImitGraph task the  $p$ -values were 0.20 and 0.39. Running the

tests showed that there are no significant differences between experienced and novice participants in task completion times, when executing tasks on ImitGraphs.

**Equivalence Testing** Not showing a significant difference does not mean that the two groups are equivalent. To study the equivalence a TOST was conducted. The TOST requires normally distributed data which can be vaguely assumed. We set the bounds of equality to  $\pm 60$  seconds, since we thought a minute difference in task completion time is similar enough, especially when working with a new tool for the first time. Like the novice users did. The  $\alpha$  level was set again to 0.05. To fulfill the tests of equivalence the 90% confidence interval has to be included in the TOST interval.

Running the tests yield the results summarized in Table 4.5. It is seen that the task completion times between experienced and novice participants cannot be declared as equivalent since the 90% confidence interval is not included in the TOST interval for any of the tasks. Therefore, the task times are not equivalent in a boundary of  $\pm 60$  seconds and thus the two groups do not perform equivalent within that boundary.

Table 4.5: TOST results of task completion times of Experiment 2

(a) TOST on Activity Diagram ImitGraph task		(b) TOST on Class Diagram ImitGraph task	
Test	Value	Test	Value
Lower bound (TOST)	-60.000	Lower bound (TOST)	-60.000
Lower bound (90 %)	-158.566	Lower bound (90 %)	-121.295
Upper bound (90 %)	17.233	Upper bound (90 %)	118.962
Upper bound (TOST)	60.000	Upper bound (TOST)	60.000
Test interpretation	Not equivalent	Test interpretation	Not equivalent

(c) TOST on ERD ImitGraph task	
Test	Value
Lower bound (TOST)	-60.000
Lower bound (90 %)	-193.471
Upper bound (90 %)	27.804
Upper bound (TOST)	60.000
Test interpretation	Not equivalent

**Thinking Before Executing** In the post-session questionnaire participants answered to the question "I needed a reasonable amount of time for thinking about the commands before carrying them out" on a scale from one to seven (1 = almost no time to think, 7 = more time to think). The results showed that experienced participants gave four out of seven points on average (Median = 5), whereas novice participants gave 4.5 points on average (Median = 5.5). But two novice and two experienced participants only gave two points as an answer, meaning they did not really have to think about a command. The slight difference between experienced and novice participants can be explained on the fact that novice participants had to think about how to carry out the command with the tool, but not about the command itself. On the other hand, experienced

participants had knowledge of the tool and only had to think about the command, which might have required less time. In the time data itself we could not observe much thinking before a task as participants just started right away with the first command, even before reading all the commands or the ImitGraphs definition. Therefore, we conclude that thinking before carrying out the commands took place, but not in such a way that the participants had to invest more than a few seconds for one command. Nor did the actual time spend for thinking differed between experienced and novice participants.

## Task Completion Time Discussion

From the results we could see that there is no significant difference between the task completion times of experienced and novice participants when working with ImitGraphs. But we could not prove equivalence in a  $\pm 60$  seconds interval. We think that there are similarities and also differences in the task completion times due to the fact that the experienced participants had knowledge of the used tool in some way and are used to work with graphical models. All experienced participants stated in the questionnaire to have used draw.io at least once before conducting the experiment. Six experienced participants even occasionally used draw.io in the last year whereas novice participants never used draw.io before. Therefore, experienced participants had the tendency to be slightly faster than novice participants, even though some novices were as fast as experienced participants.

Yet the difference in task completion time is not significant. This may come from the fact that for all the participants ImitGraphs were completely new and that the influence of the tool could be reduced. The tool even may have less influence due to the fact that all participants received a trial before conducting the experiment with ImitGraphs. To test if there is an actual difference or equivalence in the task completion time experiments with more participants have to be conducted.

Due to the found results we can say that there is no difference in the task completion time between experienced and novice participants, but similarity cannot be proven either to the specific interval. But since at least some novice participants were quicker in completing tasks as experienced participants we tend to say that it is possible to conduct usability tests on ImitGraphs also with novice participants and thus have an answer to the research question RQ 2.2.

That participants actually felt comfortable about their task completion time could also be seen in the post-session questionnaire. Participants stated that they could quickly complete the tasks based on their own perception. We could not find significant differences in the answers between novice and experienced participants. Experienced participants gave 5.5 points out of seven and novice 5.8 out of seven. Both groups had a median of six points. Since no group stated that it took them too long to complete a task and felt confident about their completion time, we think that also novice participants could be used for usability tests that are based on ImitGraphs.

### 4.2.2 Steps Taken and Interaction Behavior

The logging tool logged the most important steps participants took during the tasks. Together with the recordings of the screen capturing tool we analyzed if novice participants took the same number and types of steps as experienced participants and if the interaction with ImitGraphs is similar between the two groups.



## Overall Number of Steps

Taking a look at the descriptive statistics in Table 4.6 and the diagrams of Figure 4.8 we can see similarities in the overall number of steps taken by experienced and novice participants. The means of the novice participants were lower in the Activity Diagram ImitGraph and Class Diagram ImitGraph tasks as for experienced participants. Reviewing the median, we can see that it is the same in the Class Diagram task, meaning half of the participants took fewer steps and half of it more. In the ERD task the experienced participants took less steps on average than the novice participants. What also can be seen is that some novice participants took fewer steps than experienced participants leading to the lower whiskers in the box-plot of Figure 4.8a. The standard deviation is again high since some participants took noticeably more steps than others. Only with the descriptive statistics we cannot tell if there are significant differences in the number of steps taken during the tasks between the two groups and need more tests.

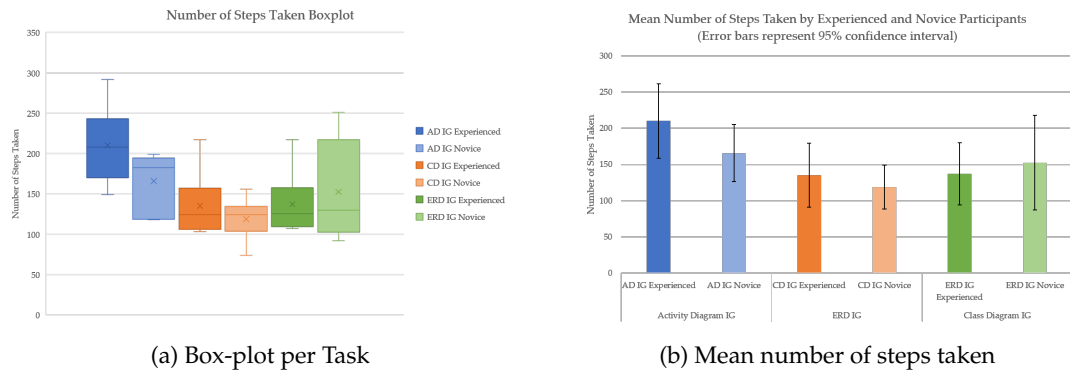


Figure 4.8: Box-plot and means of number of steps taken in Experiment 2

Table 4.6: Descriptive statistics of number of steps taken in Experiment 2

	Activity Diagram IG		Class Diagram IG		ERD IG	
	Experienced	Novice	Experienced	Novice	Experienced	Novice
Min	149	118	103	74	107	92
Max	292	199	217	156	217	251
Mean	210	166	135	119	137	153
Median	208	183	124	124	126	130
Standard Deviation	49	37	42	29	41	62

**Significant Difference** To check whether there is a significant difference in the overall steps taken between novice and experienced participants, we used the non-parametric Mann-Whitney test, since the Shapiro-Wilk test for normality showed a non-normal distribution for two of the data sets used. Again, we set the  $\alpha$  level to 0.05. Conducting the test, we received the expected results. The  $p$ -values showed that there were no significant differences in the number of steps taken between experienced and novice participants. The received  $p$ -values were 0.10 for the Ac-

tivity Diagram ImitGraph task, 0.70 for the Class Diagram ImitGraph task and 0.94 for the ERD ImitGraph task. Those are all greater than the predefined  $\alpha$  level. We therefore conclude that there is no significant difference in the overall steps taken by experienced and novice participants, but we cannot prove equivalence since the data is not distributed normally.

**Perception of Effectiveness** Participants rated in the post-session questionnaire how effectively they could complete the tasks on a scale from one to seven. In the answers we could see a slight but not significant difference between experienced and novice participants. Whereas the experienced participants gave 5.3 points on average (Median = 5.5) novice participants gave 6.2 points on average (Median = 6). The answers are based on the perception of the participants, but we could see that experienced participants thought of completing the tasks less effective than novice participants thought of it.

The difference, even though it is not significant, might come from the fact that experienced participants rated their effectiveness based on their experience with graphical models. And since ImitGraphs were new to all participants the perceived effectiveness was lower than experienced participants were used to.

## Interaction Behavior

Differences between the interaction of experienced and novice participants with ImitGraphs could not be found at first sight. But we were curious why novice participants took fewer steps in two of the three tasks even though there was no significant difference. Adding nodes or connections were correlating with the overall steps taken and only slight but no significant differences were found. We could neither find a significant difference in deleting nodes or connections nor undoing steps taken previously. But reviewing the recordings showed that there were indeed two things we should consider, which could describe the slight difference in the number of steps taken.

1. Experienced participants care more about the layout. We found out that some experienced participants tend to have a nicer layout at the end of a task. Meaning that they decided, even though technically finished a task, to select and move around nodes to achieve such a layout. Novice participants tended to leave the diagram as it was after completing the last command of a task. We base this statement on the selections and movements made by participants. Experienced participants selected and moved more nodes or connections as novice participants. This could be based on the fact that experienced participants have knowledge of designing graphical models which should have a better and cleaner organized layout and therefore adapted this thinking onto ImitGraphs. To achieve a final layout that resembles more between novice and experienced participants the definitions of an ImitGraphs connections had to be stricter. So that a connections orientation might be specified as "vertical" instead of "any".
2. Novice participants tried out different things with the tool. During the tasks and especially during the trial novice participants asked more questions about what is possible and what not. While answers were given in the trial, the observer did not answer these questions during the tasks. Experienced participants however did not ask those kinds of questions that often. This might be because they already knew the tool and maybe felt comfortable with the way how they were adding and manipulating elements of a diagram. As novices tried out the different possibilities of the tool, they discovered the more efficient methods to add nodes or connections. Such methods are dropping nodes onto connections as in Figure 4.3 or adding a node by dropping it on the end of a connection. Using such methods requires

less overall steps to be taken. Five out of six novice participants used at least one of these methods whereas only one experienced participant used them.

Even though there might be slight differences in the types and steps taken, participants interacted similar with ImitGraphs regardless of their experience. Except for above-mentioned methods nodes were added and connected in same manners. Participants were troubling with the same commands and neither group seemed to have more difficulties as the other group. Based on these results we answer research question RQ 2.4 and conclude that novice participants interact with ImitGraphs as experienced participants and could therefore be used in usability tests with ImitGraphs.

### 4.2.3 Errors

**During Task** The experimental tool checked for ImitGraphs whether a connection made by participants is allowed or not and presented respective errors to the participants. We therefore can compare the errors made during a task between experienced and novice participants. If a participant did such an error he had to change the connection point at the attached node (top, bottom and left, right) or modify the layout of the connection (horizontal, vertical) according to the ImitGraphs definition. Taking a look at the descriptive statistics of Table 4.7 we can see that more of these errors were made on average by experienced participants. Also, the conducted t-test and Mann-Whitney test showed that there is a significant difference, in such errors made, between experienced and novice participants for the Class Diagram ImitGraph task. The other tasks revealed no significant differences.

The differences between the two groups might be due to the fact, that those novice participants wanted to do everything perfectly. We could observe that after running into an error message, novice participants first checked the ImitGraphs definition and tried to fix the error based on the description. Whereas experienced participants tried other possibilities first (different connection point, different connection layout). If using such a trial and error approach more errors are made. It could be seen that more experienced users did this and the differences in errors made could be based on these two different approaches.

The types of errors made during a task did not differ between experienced and novice users, meaning that all participants ran into similar errors regardless of their experience. This also confirms similarities between the two groups.

Table 4.7: Descriptive statistics of errors made during tasks of Experiment 2

	Activity Diagram IG		Class Diagram IG		ERD IG	
	Experienced	Novice	Experienced	Novice	Experienced	Novice
Min	1	0	2	0	0	0
Max	7	5	8	2	4	7
Mean	4.17	2.17	4.17	0.83	1.67	1.83
Median	5	2	4	1	1	1
Standard Deviation	2.23	1.72	2.40	0.75	1.51	2.71

**At Submission** If errors were not corrected during the task by undoing or deleting previous steps, they were counted as errors at the submission. As in Experiment 1, hardly no errors were

submitted but two exceptions. Otherwise wrong cardinalities and loose connections could be found in the submitted tasks. What could be seen is that only two of the six experienced participants made cardinality errors, whereas every novice participant made at least one. This is a quite remarkable difference especially because we explained cardinalities for ImitGraphs to every participant in the same way. We think that this comes from the practical knowledge of the experienced participants. They may be more trained to look at this kind of details than novices. To avoid such errors, cardinalities should be explained more detailed and their importance should be stated. We think, then also novice participants would have made less cardinality errors.

Loosely attached connections could be found for both groups of participants in similar ways. All participants that were aware of the error corrected it during the task. But as stated in Section 4.1.3 due to the tool it was possible to just not recognize this error.

One experienced and one novice participant made more than just cardinality or loose connection errors. They both had more errors in their submission of the Activity Diagram ImitGraph. Both did not connect nodes that should have been connected by the "Inserting" command, already mentioned and visible in Figure 3.3b. This is another indication that the "Inserting" command was too difficult for the participants and should be split up in future experiments.

#### 4.2.4 Learnability

To check whether we can observe a learnability for participants that work with ImitGraphs, we decided to evaluate a change in the efficiency over the time. Before calculating the efficiency, we had to determine the number of steps an omniscient participant would take for each task. This number was then divided by the actual steps a participant took for the specific task. So, if a participant needed 55 steps to complete the Activity Diagram ImitGraph task but it would only need 22 steps to complete the task, the participant had an efficiency of 0.4. After calculating the efficiencies for each participant in each task, the efficiency change from the first to the second task, from the first to the third task and from the second to the third task was analyzed. We then noted how many participants increased their efficiency from one task to the next. Figure 4.9 provides the percentages of novice and experienced participants who increased their efficiency over time.

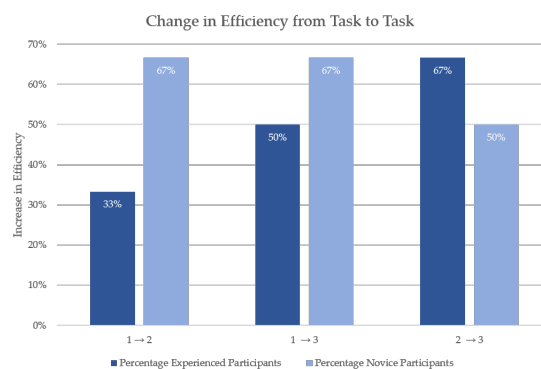


Figure 4.9: Change in efficiency from task to task in percentage of participants

We can see that 67% of the novice participants and 33% of experienced participants improved their efficiency from their first task to the second task. From the second to the third task 67% of experienced participants and 50% of novices improved their efficiency. Overall more novice

participants could improve their efficiency compared to experienced participants.

Additionally, we calculated the efficiency compared to the average efficiency of the specific task. The results can be seen in Figure 4.10a and are based on the three different groups, which completed the tasks in different orders. We can see that for the third task the groups completed, they reached an efficiency over 100% meaning that they were more efficient than the average participant. As an example: The Class Diagram task was the third task for Group 1 but the second for Group 2 and the first task for Group 3. The efficiency of Group 1 in the Class Diagram task was higher than the efficiency of the average participant in that task. This is true for all tasks. Based on this, even though we have only a few data points, we think that participants were improving their efficiency with ImitGraphs over time.

But the efficiency improvement may be based on two factors. The first factor is that participants learned how to use the tool in more efficient ways. This would also explain why there are more novice participants that could improve their efficiency. Because the experienced users already knew about the tool and were closer to their point, where only little improvement with the tool could be done. The second factor is that participants learned how to execute ImitGraph commands and thus made fewer errors. That fewer errors were made over time can also be seen in Figure 4.10b where participants made fewer errors on the third task than the average participant in this specific task. Making fewer errors requires less steps and results in a higher efficiency. But to observe a real learning curve more experiments with more users and several trials have to be done in the future.

To answer the research question RQ 2.5, we tend to say that learning can be observed for participants that work with ImitGraphs. But the learning might be due to different factors.

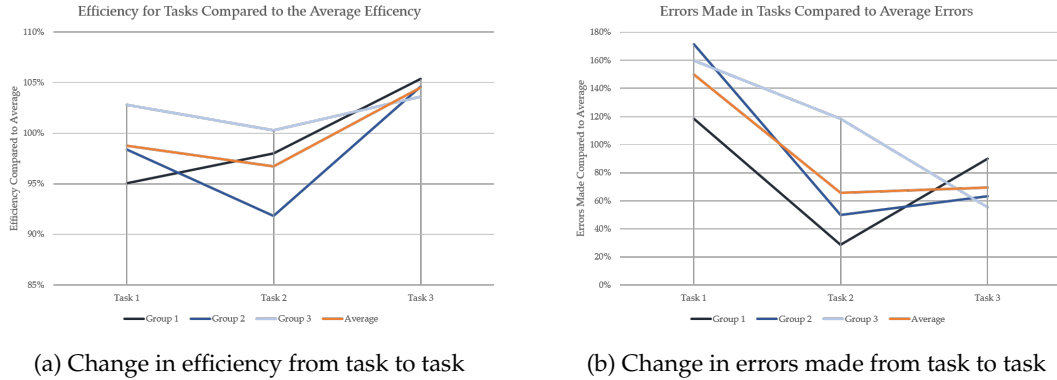


Figure 4.10: Change in efficiency and errors made from task to task

**Behavior Adaption** Based on the recordings of the screen capturing tool, we could see a behavior adaption of participants when working with ImitGraphs. That change could also influence the participants' efficiency and was also explained in Section 4.2.3. If a participant ran into an error message raised by the ImitGraph tool, the participants mostly checked the ImitGraphs definition to correct the error stated by the message. This behavior changed for some participants and especially for experienced ones after getting more error messages and can best be exemplified with the following example:

*Assume a connection that can only be attached at the right or left connection point of a node.*

*The participant now tries to attach the connection on top of the node and receives an error message. If this is the first error message received by the participant, he checks the ImitGraphs definition and realizes that he has to attach the connection on the left or right connection point of the node. After a while the participant again runs into an error message. But this time the participant tries out every connection point of the node and does not check the ImitGraphs definition. Several error messages pop up until the participant finds the correct connection point.*

The adapted behavior can be seen as a trial and error behavior and could be observed for both groups of participants. Since this behavior requires more steps the efficiency decreases for the specific participant. We think the behavior adaption occurred based on an ImitGraphs definition layout. Several participants stated in the post-session questionnaire, that they found the definition with the three tables quite confusing and wished for another more convenient definition layout. As an example they stated, to change the entries of the "Joint Type" column in Figure 3.3a to pictures instead of textual entries. We too think that an adaption of the tables could lead to less trial and error behaviors, a better overview of an ImitGraphs definition and thus resulting in a higher efficiencies and lower task completion times.

Nevertheless, participants also stated in the post-session questionnaire that ImitGraphs were easy to learn. The mean score of the Question "Drawing ImitGraphs was easy to learn" was 6.5 points out of seven, regardless of the experience. This indicates that, although the definitions could be more convenient, they were still easy enough to understand and practicable.

## 4.2.5 Overall Satisfaction

In the post-session questionnaire participants were asked how simple and comfortable it is to work with ImitGraphs. Additionally, we asked them how satisfied they overall are with drawing the graphs. Participants could answer on a scale from one to seven based on their perception, where seven is a total agreement and one a total disagreement.

The results showed, that novice participants (Mean = 6.7) perceive the use of ImitGraphs simpler than experienced participants (Mean = 5.7). We think that this difference is because experienced participants had practical knowledge of drawing graphical models. And since ImitGraphs were new to them they rated the simplicity lower due to the fact that they had to work with ImitGraph definitions not known and error messages that are not part of drawing graphical models. But that ImitGraphs are not difficult to draw showed the mean values for simplicity. Both mean values were over 5.5 and therefore participants agreed that ImitGraphs are indeed simple to draw.

Both groups of participants felt comfortable (Mean = 5.6) and were overall highly satisfied (Mean = 6.3) with drawing ImitGraphs. Thus, we can say that the perceived effort of drawing ImitGraphs does not differ significantly between experienced and novice participants. Usability tests with ImitGraphs could therefore be conducted with novice participants since they do not perceive a higher effort than experienced participants.

## Chapter 5

---

# Validity

In this chapter the validity of the results is discussed according to known threads stated by Wohlin and Hoest in [22].

### 5.1 Construct Validity

If the used test actually measures the constructs it claims to measure, is described by the construct validity. We kept focusing on different measurements that are capable of identifying different interaction behaviors between ImitGraphs and the original graphical models. Thus, we think the task completion time, the overall and different types of steps, errors made, and the learnability are essential measurements and represent the construct. The tasks were designed to act as work-flows similar to work-flows modeling experts have.

### 5.2 Conclusion Validity

To which degree it is possible to draw conclusions from the data observed is part of the conclusion validity and we have to consider following issues:

**Measurement** Every action and its time were logged by the tool. Therefore, the precision of the data is accurate. Together with the logs, the activities were captured by a screen capturing tool for further analysis.

**Statistical power** The statistical power we had in the experiments is not high and might be the greatest threat to validity in our experiments. Due to costs we only had 12 participants for each experiment. But to keep the threads to statistical tests low we used both, parametric and non-parametric tests to analyze the data. We think that using more participants would lead to the same results, based on the observations gained throughout the experiments.

### 5.3 Internal Validity

The internal validity ensures that the measured outcome results from the change of the independent variable and is not based on other side effects not known to the researcher. Regarding this we have to consider following issues [3]:

**Differences between participants** The first experiment was designed as a with-in subject experiment and therefore reduces the variability among the participants. Experienced participants had approximately the same experience with working on graphical models and were all on similar levels of their education. The novice participants of the second experiment all had no prior knowledge of creating graphical models and that was the desired initial situation.

**Task differences** The tasks for an ImitGraph and the corresponding graphical model were designed as similar as possible, so that a different interaction was only due to the type of model used. For the second experiment, tasks were the same for each participant and thus different outcomes of the dependent variable only due to the different participants.

**Learning effect** Tasks were ordered differently for participants to eliminate learning effects if not searched for. The participants executed the tasks in a predefined order.

**Fatigue effects** The different order of the tasks and variety of daytimes the experiment took place, helped to avoid fatigue effects. The experiments never took more than an hour for Experiment 1 or 40 minutes for Experiment 2. Experienced participants whose data was used in both experiments made the ImitGraph tasks first. Therefore, no fatigue effects between experienced and novice participants should exist.

**Participant Motivation** The participants were ensured that the experiments are taken anonymously. Since participants were contacted by phone and voluntarily took the experiment we got the feeling that all participants were highly motivated during the experiment.

## 5.4 External Validity

The external validity is concerned with the ability to generalize the results of an experiment [22]. Regarding this we have to consider following issues [3]:

**Materials and Tasks** The tasks were designed to represent real cases as close as possible, but to further analyze the behavior with ImitGraphs experiments with more participants are necessary.

**Participants** We decided to run the experiment with computer science students for Experiment 1 due to the difficulty of getting professionals. But since the students are close to their degree or already have one they are close to start working in the industry and therefore suitable for this experiment. In order to say that the interaction with ImitGraphs is comparable to professionals, more experiments with different participants should be conducted. For Experiment 2 a wider variation (age and occupation) of the novice participants might be necessary for further experiments.



# Conclusion and Future Work

To evaluate the effectiveness of their work, user interface designers need to conduct tests of their newly designed product. But since these tests can get expensive designers either test only subsets of their intended graphical models or at a late stage of tool development [7]. To achieve a short development time of a working prototype that can test multiple graphical models and also has the possibility to recruit participants with no prior knowledge of graphical models ImitGraphs were proposed in [7]. Such ImitGraphs then could be used for usability tests in early stages of development. To analyze ImitGraphs we tested in two experiments if the process of creating ImitGraphs is comparable to the process of creating the corresponding graphical model and if usability tests with ImitGraphs can be performed on a wider range of participants that have no prior knowledge of graphical models. Overall 18 participants took part in the experiments, 12 with experience in designing graphical models and six novices. The results we found confirmed our hypotheses that ImitGraphs can be used in usability tests instead of the original graphical models and that novice participants can be used for usability tests with ImitGraphs. All results are summarized below and the future work presented thereafter.

## 6.1 Conclusion

### Experiment 1

In the first experiment similarities in the interaction behavior between ImitGraphs and the corresponding graphical models were tested. No significant differences in task completion times, overall steps taken, or errors made were detected. The task completion times we could observe were equivalent to a  $\pm 60$  seconds time difference between the graphical models and ImitGraphs. We noticed that the task completion times for an ImitGraph task and its corresponding graphical model task are as close to each other as the models are to each other. The better the ImitGraph is imitating the model, the closer are the task completion times. Surely ImitGraphs were new to all participants and therefore the mean task completion times were higher for ImitGraphs, but not significantly. Before executing the different commands of a task, participants had to think less about commands of ImitGraphs than about the commands of graphical models. This is due to the composition of an ImitGraphs command which is partially textual and partially visual and therefore is more intuitive for a participant. The freedom of interacting with the different models is not lost by the different commands.

That the freedom how to interact with ImitGraphs and the graphical model was similar could be observed in the overall steps taken by participants. No significant differences could be found

in the overall steps taken by participants, nor could we observe any critical interaction behaviors that would differentiate ImitGraphs from the corresponding graphical models. If participants knew about certain features of the used tool they used the feature for both types of models. Also, the adding and connecting of elements was made in similar ways for both types of models.

Only a few errors were made and were mostly cardinality or loose connection errors. Even though no significant differences in errors made between the models were found, we think that ImitGraphs are less error-prone than the graphical models, at least regarding the final submission. We think this is again due to the specific commands given for ImitGraphs, where parts of the solution and how nodes should be connected are already visible. Nevertheless, an ImitGraph command should be tested on its difficulty before giving it to participants in experiments. Since some commands (as the "Inserting" command) are not that intuitive and take participants more time to execute, commands should be split up if the graphical model command would allow that.

In the post-session questionnaires participants stated that they were as efficient with ImitGraphs as with the corresponding graphical models. And even though some participants stated, that it was a bit complicated to understand the definitions of an ImitGraph, the learning was quite easy. Overall participants showed similar satisfaction rates for both model types.

Based on the found results we tend to say that participants actually behave similarly when working with ImitGraphs and the corresponding graphical models and therefore we can approve our first hypothesis  $H_1$  stated in 3.2. Since the participants behave the same with both models, gained feedback from usability tests on ImitGraphs can be transferred to the original models. Thus, a wider range of usability tests can be performed on ImitGraphs without implementing usability tests for each individual graphical model. To conduct usability tests based on ImitGraphs, they have to be explained to participants and need to be implemented in the prototype under test. But a ten-minute explanation for ImitGraphs appears to be enough and once a definition for an ImitGraph is found it can be used for further optimization and more tests [7].

## Experiment 2

In the second experiment we tested if usability tests on ImitGraphs could be done on a wider range of participants that have no prior knowledge of graphical models. Therefore, we analyzed if the behavior of novice participants is similar to the behavior of experienced participants when editing ImitGraphs. We could observe that there is no significant difference in task completion times between the two groups of participants. But the completion times were not equal either to a boundary of  $\pm 60$  seconds. Due to no significant differences in task times, we tend to say that it does not matter if a novice or experienced participants are used to conduct usability tests on ImitGraphs. From the results we cannot say that experienced participants completed the tasks faster since also some novice participants completed the tasks quickly and successfully.

Similar results could be seen in the overall steps taken by the different groups of participants. Novice participants did not need more steps to complete an ImitGraph task than experienced participants and no significant differences could be found. Two different interaction behaviors however could be observed. The first difference is that experienced participants cared more about a clear layout and thus took some additional steps. Second, novice participants tended to experiment more with the tool given to them, meaning that they used more of the available features of the tool. These two differences do not contradict with our assumption that usability tests on ImitGraphs can be done with novice participants since overall participants used similar methods to add, connect or delete nodes regardless of their experience. If new interaction methods would be tested on ImitGraphs they could be tested with novices as with experienced participants.

Errors were made by both groups of participants. In one task we could observe significantly

more errors made by experienced participants than by novices. This result was rather unexpected and might be explained on the fact that novices took slightly more time in the ImitGraph trial as experienced participants took, even though both groups were told to take as much time as they want. Also, two experienced participants made considerably more errors than an average participant but were not outliers. Since we could not observe the effect on different tasks, we think that in general similar errors in comparable amounts are made regardless the experience of the participants.

We also observed the learnability of the participants. Even though the number of participants and tasks executed were low we noticed a slight learnability of the participants when working with ImitGraphs. Most participants could improve their efficiency over time and all participants were more efficient on their last task than the average participant in this task. We also could see that fewer errors with ImitGraphs were made as longer the participant worked with ImitGraphs.

Based on the found results we tend to say that usability tests with ImitGraphs can be conducted on a wider range of participants that have no prior knowledge of designing graphical models. As a result, we can approve  $H_2$  stated in 3.2. Thus, more participants can be used for prototype usability tests since there are no differences in the interaction with ImitGraphs between experienced and novice participants. As we have seen in Experiment 1 experienced participants interact with graphical models in similar ways as with ImitGraphs. Therefore, we could say that novice participants would also interact in at least similar ways with graphical models. Meaning that if a feature is tested on ImitGraphs instead of the original graphical model and the interaction between these two is similar, the feature could also be tested with novice participants since they interact with ImitGraphs similar to participants with more experience.

## 6.2 Future Work

Even though we showed that usability tests can be conducted on a wider range of participants it needs to be defined to what range of participants that actually applies. In our experiment, only younger participants (age 20-27) took part in the experiments. But we think that also older participants could take part in usability tests with ImitGraphs as long as they have a computer affinity. This needs to be tested on a broader part of the population. Also, a test with real modeling experts might show that they interact with certain kind of features differently than novice participants. But to conduct such tests, experts must be identified and convinced to take part in the experiment, which might be cost intensive.

Also, the costs of implementing such ImitGraphs into a working prototype, which is not part of the final product have to be compared to the benefits that more participants could be recruited for usability tests and that usability tests could be made on a wider range of graphical models.



---

# Bibliography

- [1] Abrahao, S. and Insfran, E. (2006). Early usability evaluation in model driven architecture environments. In *Sixth International Conference on Quality Software*, pages 287–294. IEEE.
- [2] Blandford, A., Cox, A. L., and Cairns, P. (2008). Controlled experiments. *Research methods for human-computer interaction*, pages 1–16.
- [3] Calero, C., Piattini, M., and Genero, M. (2001). Empirical validation of referential integrity metrics. *Information and Software Technology*, 43(15):949–957.
- [4] Easterbrook, S., Singer, J., Storey, M.-A., and Damian, D. (2008). Selecting empirical methods for software engineering research. *Guide to advanced empirical software engineering*, pages 285–311.
- [5] Frisch, M., Dachsel, R., and Brückmann, T. (2008). Towards seamless semantic zooming techniques for uml diagrams. In *Proceedings of the 4th ACM symposium on Software visualization*, pages 207–208. ACM.
- [6] Galitz, W. O. (2007). *The Essential Guide to User Interface Design: An Introduction to GUI Design Principles and Techniques*. Wiley Desktop Editions. John Wiley & Sons.
- [7] Ghazi, P. and Glinz, M. (2017). Imitgraphs: Towards faster usability tests of graphical model manipulation techniques. *Proceedings - 2017 IEEE/ACM 9th International Workshop on Modelling in Software Engineering, MiSE 2017, (May)*:61–67.
- [8] Holzinger, A. (2005). Usability engineering methods for software developers. *Communications of the ACM*, 48(1):71–74.
- [9] Hornbæk, K., Høegh, R. T., Pedersen, M. B., and Stage, J. (2007). Use case evaluation (uce): A method for early usability evaluation in software development. In *IFIP Conference on Human-Computer Interaction*, pages 578–591. Springer.
- [10] Johnson, J. (2013). *Designing with the Mind in Mind: Simple Guide to Understanding User Interface Design Guidelines*. Elsevier.
- [11] Kagdi, H. and Maletic, J. I. (2007). Onion graphs for focus+ context views of uml class diagrams. In *4th IEEE International Workshop on Visualizing Software for Understanding and Analysis 4*, pages 80–87. IEEE.
- [12] Lee, A. and Lochovsky, F. (1985). *User interface design*, pages 3–20. Springer.

- [13] Lewis, J. R. (2006). Usability testing. *Handbook of human factors and ergonomics*, 12:e30.
- [14] MacKenzie, I. S. (2012). *Human-Computer Interaction: An Empirical Research Perspective*. ELSEVIER.
- [15] Mayhew, D. J. (1999). The usability engineering lifecycle. In *CHI'99 Extended Abstracts on Human Factors in Computing Systems*, pages 147–148. ACM.
- [16] Nekrasovski, D., Bodnar, A., McGrenere, J., Guimbretière, F., and Munzner, T. (2006). An evaluation of pan & zoom and rubber sheet navigation with and without an overview. In *Proceedings of the SIGCHI conference on Human Factors in computing systems*, pages 11–20. ACM.
- [17] Nielsen, J. (1994). *Usability engineering*. Elsevier.
- [18] Shneiderman, B., Plaisant, C., Cohen, M. S., Jacobs, S., Elmqvist, N., and Diakopoulos, N. (2016). *Designing the User Interface: Strategies for Effective Human-Computer Interaction, 6th Edition*. Pearson.
- [19] Spönemann, M. (2015). *Graph layout support for model-driven engineering*. Number 2015/2 in Kiel Computer Science Series. Department of Computer Science. Dissertation, Faculty of Engineering, Christian-Albrechts-Universität zu Kiel.
- [20] Thelin, T., Runeson, P., and Regnell, B. (2001). Usage-based reading—an experiment to guide reviewers with use cases. *Information and Software Technology*, 43(15):925–938.
- [21] Tullis, T. and Albert, B. (2008). *Measuring the user experience*. Morgan Kaufmann, 1st edition.
- [22] Wohlin, C. and Höst, M. (2001). Special section: Controlled experiments in software engineering. *Information and Software Technology*, 43(15):921–924.

## Appendix A

---

# Task Descriptions and Questionnaires

### A.1 Trial Description

The following pages show the original handout for the trial. The handout was used together with a trial set-up provided in the experimental tool to explain ImitGraphs to the participants.

# ImitGraph Explanation

---

## Definition of ImitGraphs

ImitGraphs imitate the behaviour of other graphical models. Every ImitGraph is composed of nodes, connections and joints.

**Node:** A node is a circular element that can be assigned different sizes, colours and it can hold a label.



Node A

**Connection:** A connection is a line with a rectangle in the middle. It connects two joints. The rectangle can be assigned different colours and a label.

**Joint:** A joint is a circle at the end of a connection. Joints attach connections to nodes. They can be assigned different colours and labels.



Connection (with label) with two Joints (without labels)



Connection (without label) with two Joints (with labels)



**Current Location:** Can be a node or a connection. It is the last node created, the node referenced by the last “Find Node” command, or the connection referenced by the last “Find Connection” command.







## Specialized ImitGraphs

ImitGraphs can be specialized to imitate the behaviour of a graphical model as accurate as possible. Therefore, the nodes, connections and joints are defined for every task.

Example of an ImitGraph definition:

Joint Types		
Type	Symbol	Label
J1		no
J2		no

Connection Types					
Type	Symbol	Label	First Joint	Second Joint	Orientation
C1		no	J1	J2	any

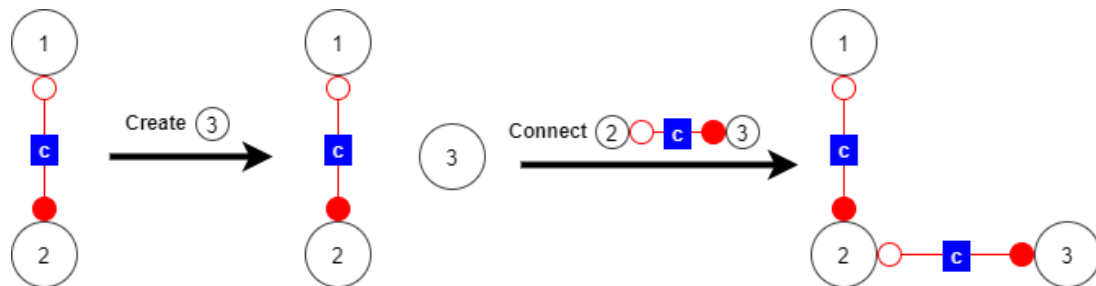
Node Types							
Type	Symbol	Label	Connection Type	Joint Type	Min	Max	Connection Point
N1		yes	C1	J1	1	1	any
			C1	J2	1	1	any
N2		no	C1	J1	1	1	any
			C1	J2	1	1	any
N3		no	C1	J1	1	1	bottom
			C1	J2	1	1	top

## Possible Commands with ImitGraphs

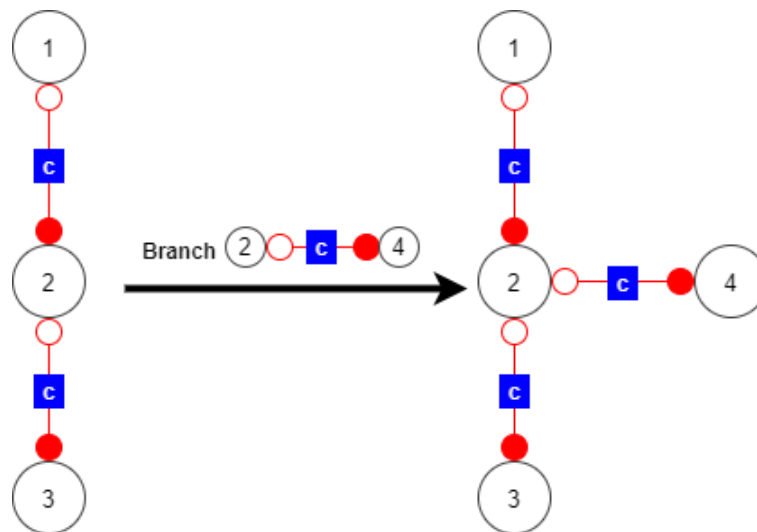
The commands of ImitGraphs are partially textual and partially visual. Examples with the definition stated above can be found below.

**Create:** Creating a new node with the specific properties.

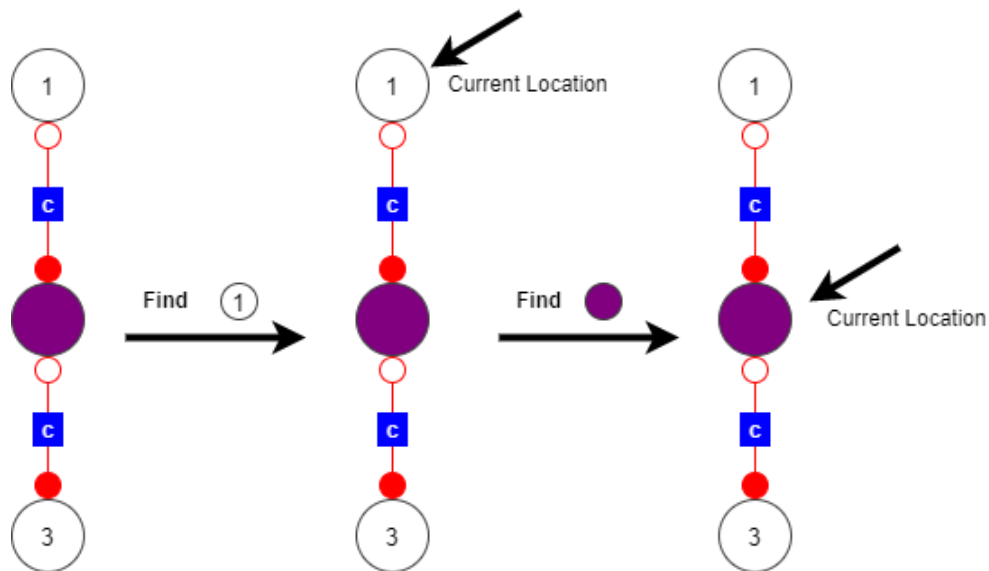
**Connect:** Create a connection specified between two nodes.



**Branch:** Add a branch made of nodes and connections to the diagram. It starts from a known node and ends in a known node or a new node.



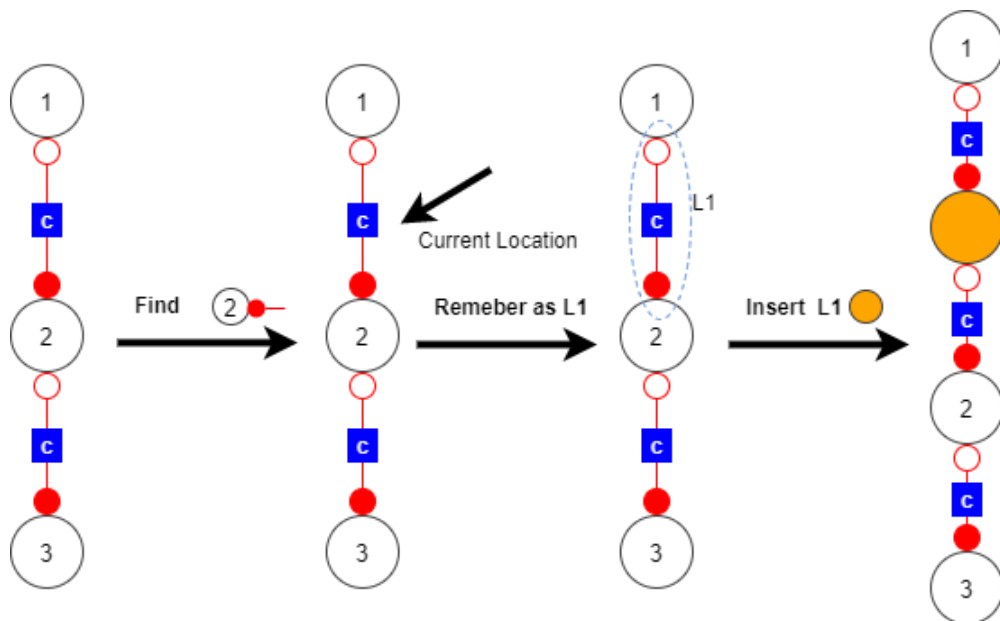
**Find Node:** Find the node specified after this command. The found node should be considered as current location.

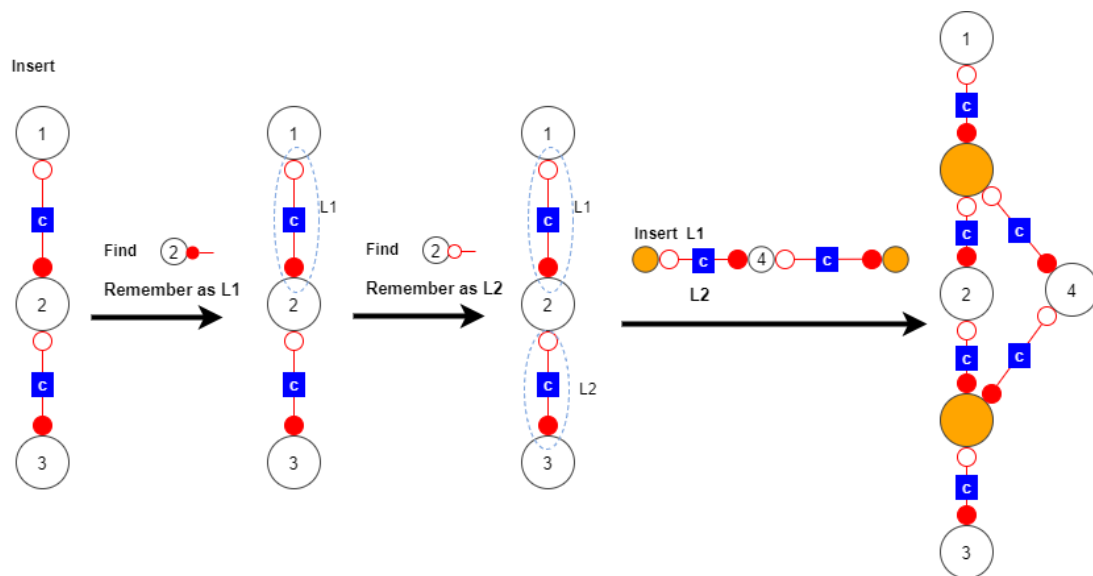


**Find Connection:** Find the connection specified after this command. The found connection should be considered as current location.

**Remember as:** Assign a specified name to the current location.

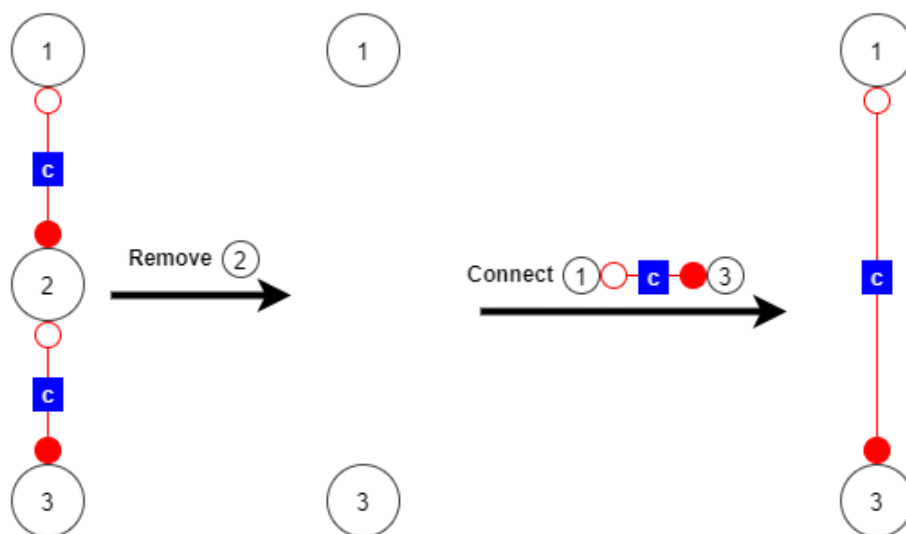
**Insert:** When the current location is a connection then a node or a sequence of nodes and connections can be inserted. The beginning of the new branch is the current location. The end can be a new node or an existing node.





**Remove:** Delete the current location. If the current location points to other nodes, delete the connections too.

**Replace:** Replace the current location with the specified replacement.



## A.2 ImitGraph Task Descriptions

Following pages show the original ImitGraph task descriptions. Included are pictures of the initial situation and a possible correct outcome of each task. The tasks are in the following order:



1. Activity Diagram ImitGraph Task
2. Class Diagram ImitGraph Task
3. Entity Relationship Diagram ImitGraph Task



# ImitGraph Task







## ImitGraph Task Description

Please edit the given ImitGraph according to the definition and commands below. If you have any questions during the task, feel free to ask.










## ImitGraph Definition

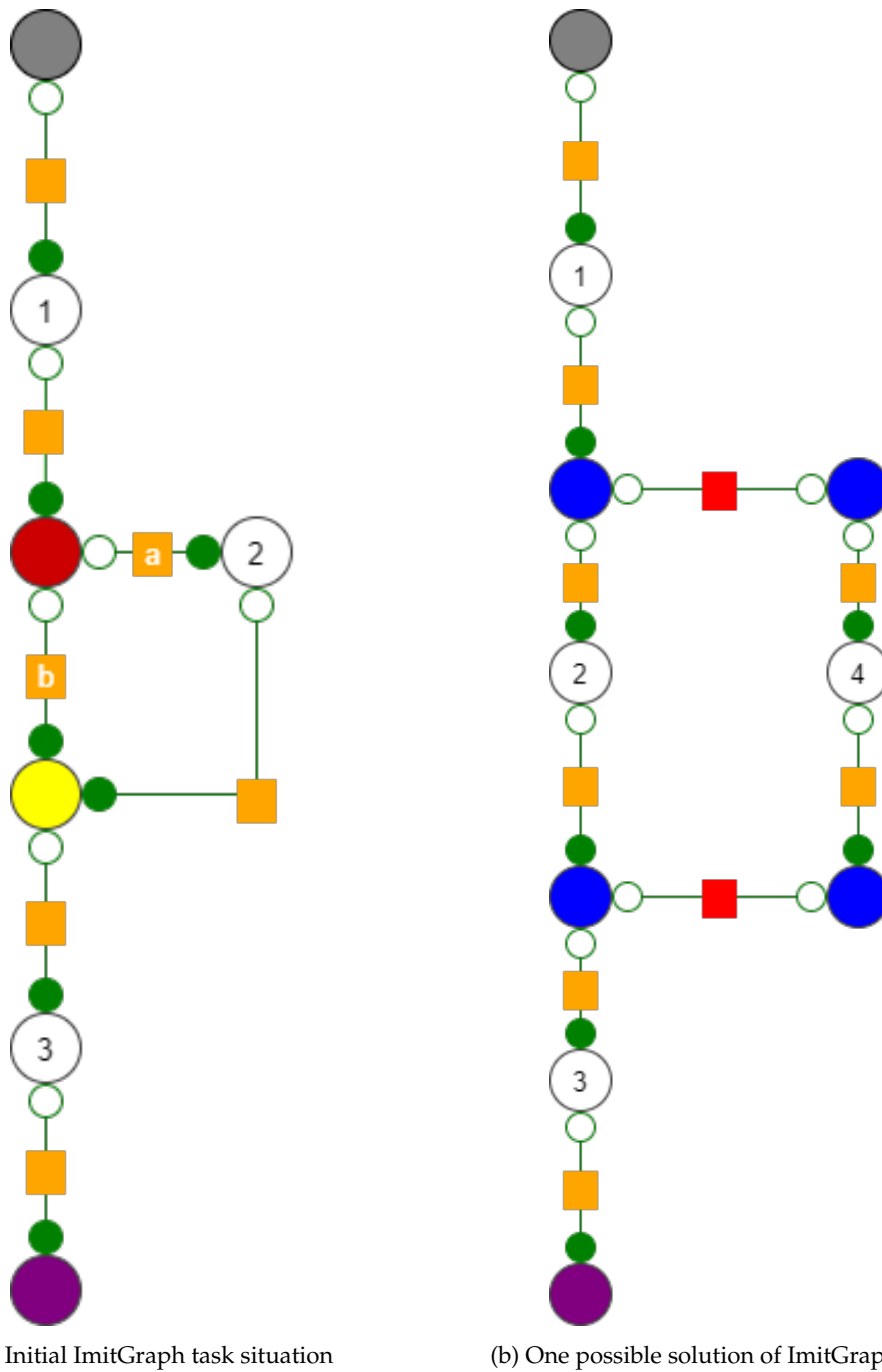
Joint Types		
Type	Symbol	Label
J1		no
J2		no

Connection Types					
Type	Symbol	Label	First Joint	Second Joint	Orientation
C1		optional	J1	J2	any
C2		no	J1	J1	horizontal

Node Types							
Type	Symbol	Label	Connection Type	Joint Type	Min	Max	Connection Point
N1		no	C1	J1	1	1	any
N2		no	C1	J2	1	1	any
N3		yes	C1	J2	1	1	any
			C1	J1	1	1	any
N4		no	C1	J2	0	1	top
			C1	J1	0	1	bottom
			C2	J1	1	2	left, right
N5		no	C1	J2	1	1	top, bottom, left, right
			C1	J1	2	2	top, bottom, left, right
N6		no	C1	J2	2	3	top, bottom, left, right
			C1	J1	1	1	top, bottom, left, right

## Task Commands

- Find Node 
- Find Node 
- Delete current location
- Find Node 
- Find Node 
- Delete current location
- Connect 
- Connect 
- Find connection 
- Remember as L1
- Find connection 
- Remember as L2
- Insert L1  L2



(a) Initial ImitGraph task situation

(b) One possible solution of ImitGraph task

Figure A.1: Initial situation and one possible outcome of ImitGraph task (Activity Diagram)










# ImitGraph Task



## ImitGraph Task Description

Please edit the given ImitGraph according to the definition and commands below. If you have any questions during the task, feel free to ask.


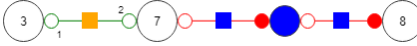


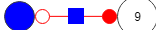
## ImitGraph Definition

Joint Types		
Type	Symbol	Label
J1		yes
J2		yes
J3		no
J4		no

Connection Types					
Type	Symbol	Label	First Joint	Second Joint	Orientation
C1		optional	J1	J1	any
C2		optional	J2	J1	any
C3		no	J3	J4	any

Node Types							
Type	Symbol	Label	Connection Type	Joint Type	Min	Max	Connection Point
N1		yes	C1	J2	1	n	any
			C2	J2	0	n	any
			C2	J1	0	n	any
			C3	J3	0	n	bottom
			C3	J4	0	n	top
N2		no	C3	J3	1	n	any
			C3	J4	1	1	top

## Task Commands

- Branch 
- Branch 
- Find Node 
- Find Node 
- Branch 

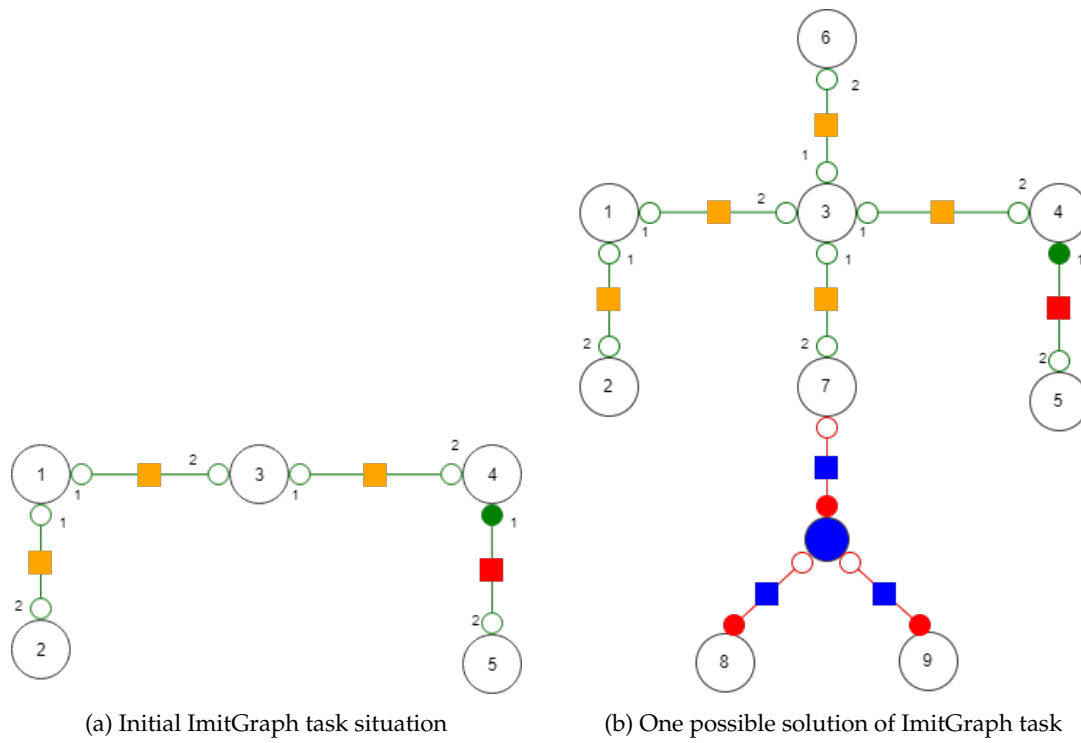




Figure A.2: Initial situation and one possible outcome of ImitGraph task (Class Diagram)



# ImitGraph Task





## ImitGraph Task Description

Please edit the given ImitGraph according to the definition and commands below. If you have any questions during the task, feel free to ask.





## ImitGraph Definition

Joint Types		
Type	Symbol	Label
J1		no
J2		yes

Connection Types					
Type	Symbol	Label	First Joint	Second Joint	Orientation
C1		no	J1	J1	any
C2		no	J1	J2	any

Node Types							
Type	Symbol	Label	Connection Type	Joint Type	Min	Max	Connection Point
N1		yes	C1 C2	J1 J2	0 1	n n	any top, bottom, left, right
N2		yes	C1	J1	1	1	any
N3		yes	C1	J1	1	1	any
N4		yes	C2	J1	2	2	top, bottom, left, right

## Task Commands

- Branch 
- Branch 
- Branch 
- Branch 

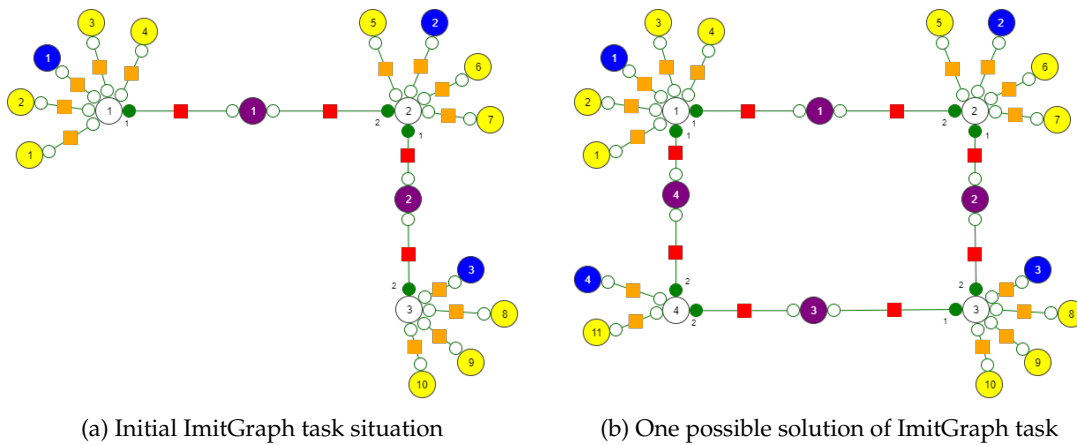


Figure A.3: Initial situation and one possible outcome of ImitGraph task (Entity Relationship Diagram)

### A.3 Graphical Model Task Descriptions

Following pages show the original task descriptions for the graphical models. Included are pictures of the initial situation and a possible correct outcome of each task. The tasks are in the following order:

1. Activity Diagram Task
2. Class Diagram Task
3. Entity Relationship Diagram Task

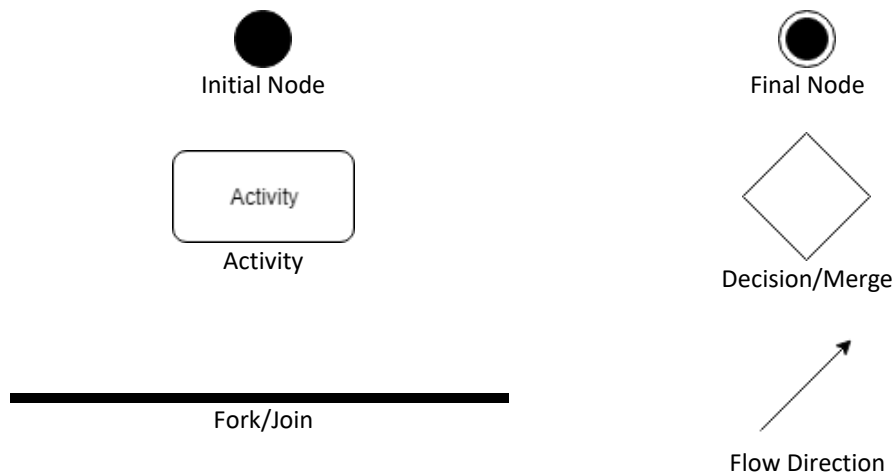
# Activity Diagram Task

---

## Activity Diagram Task Description

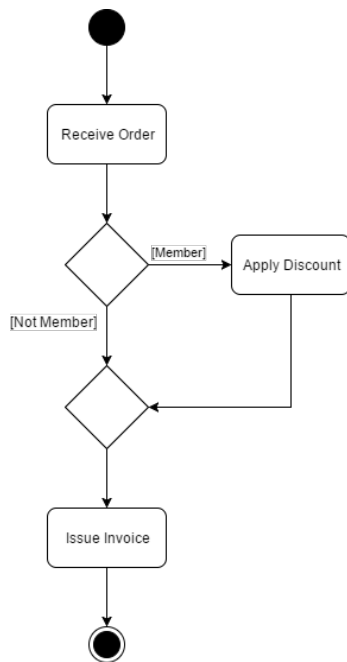
Please edit the given Activity Diagram according to the commands below. If you have any questions during the task, feel free to ask.

## Activity Diagram Elements

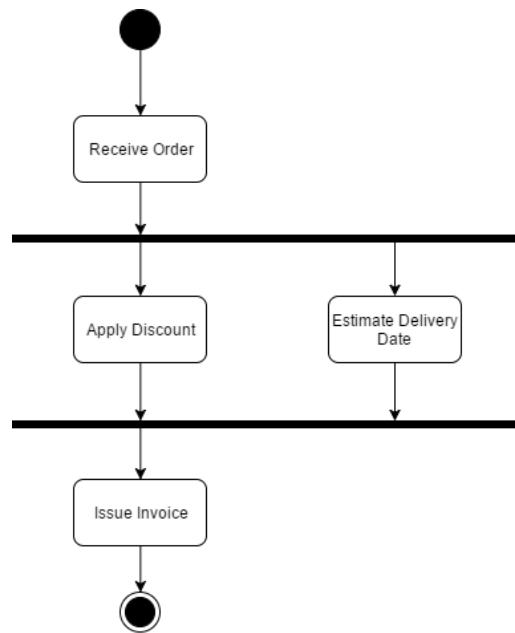


## Task Commands

- Delete decision after activity "Receive Order"
- Delete decision after activity "Apply Discount"
- Connect activity "Receive Order" to activity "Apply Discount"
- Connect activity "Apply Discount" to activity "Issue Invoice"
- Add parallel activity to activity "Apply Discount" called "Estimate Delivery Date"



(a) Initial Activity Diagram task situation



(b) One possible solution of Activity Diagram task

Figure A.4: Initial situation and one possible outcome of Activity Diagram task

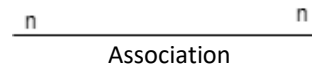
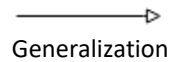
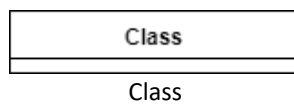
# Class Diagram Task

---

## Class Diagram Task Description

Please edit the given Class Diagram according to the commands below. If you have any questions during the task, feel free to ask.

## Class Diagram Elements



## Task Commands

- Each flight has two airports and each airport has between 0 and more flights
- Each flight needs crew members between two and ten and each crew member can be on several flights
- There are two types of crew, pilots and flight attendants

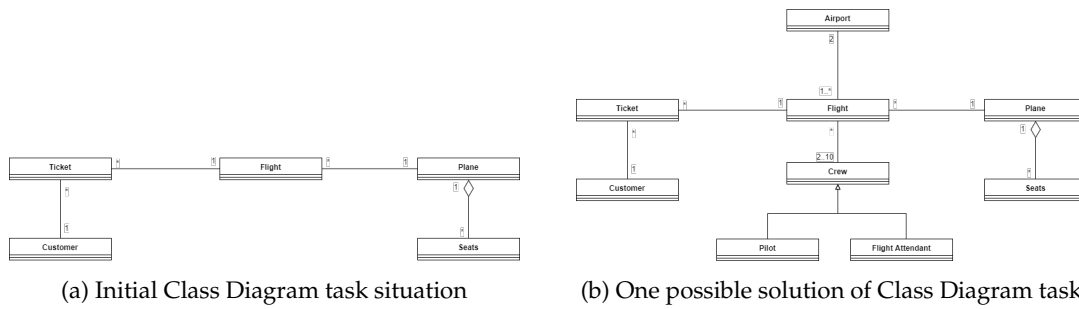


Figure A.5: Initial situation and one possible outcome of Class Diagram task



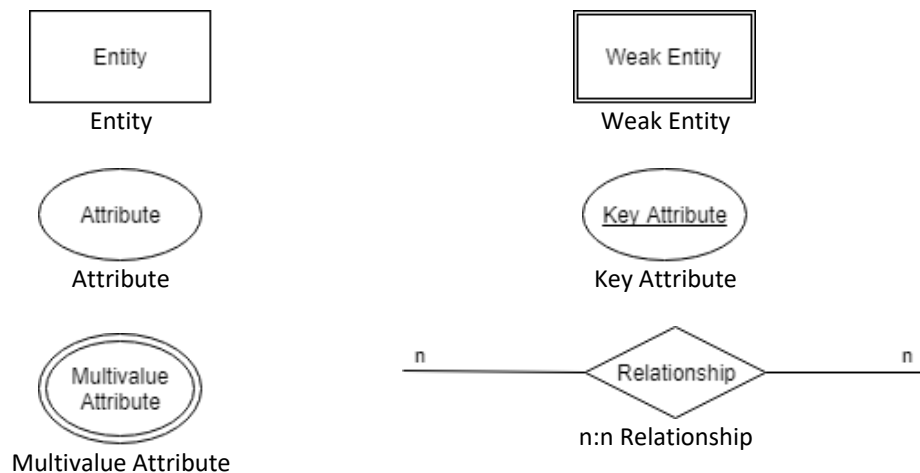
# Entity Relationship Diagram Task

---

## Entity Relationship Diagram Task Description

Please edit the given Entity Relationship Diagram according to the commands below. If you have any questions during the task, feel free to ask.

## Entity Relationship Diagram Elements



## Commands

- Each student enrolls in a university
- A university has multiple students
- A university has two attributes, a key attribute called "Name" and a normal attribute called "Address"
- Each university has multiple lecturers and each lecturer is in multiple universities

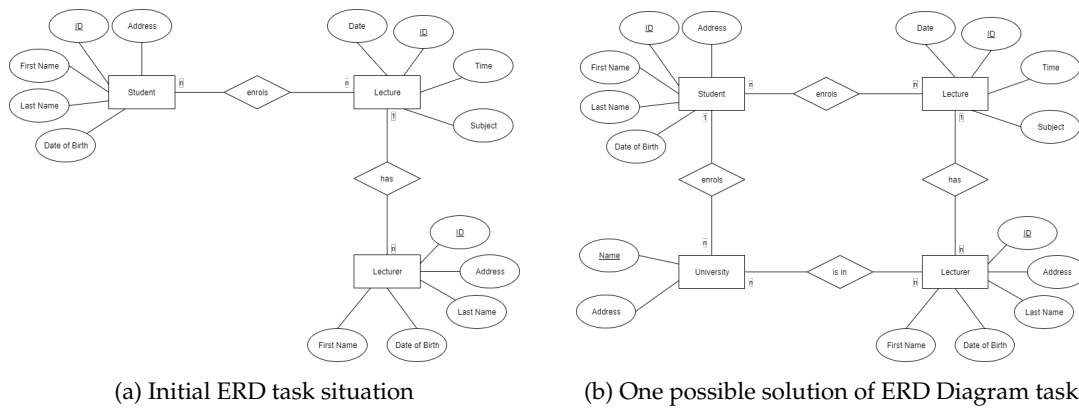


Figure A.6: Initial situation and one possible outcome of Entity Relationship Diagram task

## A.4 Questionnaires

Following, the two questionnaires used after the experiments are shown. The first page shows the ImitGraph questionnaire with questions about the participant as well. On the second page the questions for the graphical models are shown.

Assume that the ImitGraphs you created are meaningful (like Activity Diagrams). Please rate the usability of ImitGraphs based on this assumption: \*

1 = Strongly Disagree, 7 = Strongly Agree

	1	2	3	4	5	6	7	NA
Overall, I am satisfied with how easy it is to draw ImitGraphs	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
It was simple to draw ImitGraphs	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
I am able to complete the tasks quickly	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
I can effectively complete the tasks	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
I am comfortable with drawing ImitGraphs	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
Drawing ImitGraphs was easy to learn	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
Overall I am satisfied with the drawing of ImitGraphs	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
I needed a reasonable amount of time for thinking about the commands before carrying them out	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>

List the most negative aspects of working with ImitGraphs

Meine Antwort

List the most positive aspects of working with ImitGraphs

Meine Antwort

(a) Mandatory ImitGraph questions

(b) Voluntary questions

How often are you using graphical models? \*

- ☐ Never  
☐ Rarely  
☐ Occasionally  
☐ Regularly  
☐ Don't Know

Which tools you used to edit graphical models? \*

Meine Antwort

Have you ever used [draw.io](https://draw.io) before participating in this experiment? \*

- ☐ Yes  
☐ No

How often did you use [draw.io](https://draw.io) in the last year?

- ☐ Never  
☐ Rarely  
☐ Occasionally  
☐ Regularly  
☐ Don't Know

(c) Mandatory experience questions

Figure A.7: ImitGraph questionnaire

Please rate the usability of the graphical models: \*

1 = Strongly Disagree, 7 = Strongly Agree

	1	2	3	4	5	6	7	Not Applicable
Overall, I am satisfied with how easy it is to draw the graphical models	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
It was simple to draw the graphical models	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
I am able to complete the tasks quickly	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
I can effectively complete the tasks	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
I am comfortable with drawing the graphical models	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
Drawing the graphical models was easy to learn	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
Overall I am satisfied with drawing the graphical models	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
I needed a reasonable amount of time for thinking about the requirements before applying them	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>

List the most negative aspects of working with graphical models

Meine Antwort

List the most positive aspects of working with graphical models

Meine Antwort

(a) Mandatory graphical models questions

(b) Voluntary questions

Figure A.8: Graphical model questionnaire

## Appendix B

---

# Content on the Delivered CD-ROM

**Abstract.txt** The abstract of the thesis.

**Zusfsg.txt** The German abstract of the thesis.

**Bachelorarbeit.pdf** The thesis (this document).