

Webbasierte Bewertung der Notation von Modellierungssprachen

DIPLOMARBEIT

zur Erlangung des akademischen Grades

Diplom-Ingenieurin

im Rahmen des Studiums

Wirtschaftsinformatik

eingereicht von

Yuliia Kolbasiuk

Matrikelnummer 11934917

an der Fakultät für Informatik

der Technischen Universität Wien

Betreuung: Assoc. Prof. Dr. Dominik Bork

Mitwirkung: Univ.Ass. BSc MSc Syed Juned Ali

Wien, 1. September 2024

Yuliia Kolbasiuk

Dominik Bork



Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar
The approved original version of this thesis is available in print at TU Wien Bibliothek.

Web-based Evaluation of Modeling Language Notations

DIPLOMA THESIS

submitted in partial fulfillment of the requirements for the degree of

Diplom-Ingenieurin

in

Business Informatics

by

Yuliia Kolbasiuk

Registration Number 11934917

to the Faculty of Informatics

at the TU Wien

Advisor: Assoc. Prof. Dr. Dominik Bork

Assistance: Univ.Ass. BSc MSc Syed Juned Ali

Vienna, 1st September, 2024

Yuliia Kolbasiuk

Dominik Bork



Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar
The approved original version of this thesis is available in print at TU Wien Bibliothek.

Erklärung zur Verfassung der Arbeit

Yuliia Kolbasiuk

Hiermit erkläre ich, dass ich diese Arbeit selbständig verfasst habe, dass ich die verwendeten Quellen und Hilfsmittel vollständig angegeben habe und dass ich die Stellen der Arbeit – einschließlich Tabellen, Karten und Abbildungen –, die anderen Werken oder dem Internet im Wortlaut oder dem Sinn nach entnommen sind, auf jeden Fall unter Angabe der Quelle als Entlehnung kenntlich gemacht habe.

Wien, 1. September 2024

Yuliia Kolbasiuk



Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar
The approved original version of this thesis is available in print at TU Wien Bibliothek.

Acknowledgements

I am incredibly grateful to Assoc. Prof. Dr. Dominik Bork for supervising my thesis, for providing all needed input, feedback and comments, and for being patient with me, even though it took me longer to write it than I had hoped or intended.

I also extend special thanks to Syed Juned Ali for supporting me with the technical part, helping me with the implementation, and reviewing the thesis as well.

I want to extend my gratitude to my incredible parents, Oleh and Svitlana, and my amazing boyfriend, Adnan. They have always believed in me and supported me, no matter what. I cannot express enough how grateful I am to them.



Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar
The approved original version of this thesis is available in print at TU Wien Bibliothek.

Kurzfassung

Diese Arbeit befasst sich mit der Herausforderung, die semantische Transparenz von grafischen Notationen in konzeptionellen Modellierungssprachen zu evaluieren und zu verbessern. Traditionelle Ansätze zur Spezifikation von Modellierungssprachen müssen wesentliche Notationsaspekte berücksichtigen, um domänenspezifische Anforderungen effektiv widerzuspiegeln. Wir schlagen eine Plattform zur Evaluierung grafischer Notationen für die empirische Evaluierung grafischer Notationen vor, um den Aufbau, die Ausführung und die Ergebnisanalyse von Evaluierungsexperimenten zu automatisieren und dadurch die Reproduzierbarkeit und Effizienz zu verbessern. Durch den Einsatz der Design Science Forschungsmethodik und agiler Entwicklungsmethoden wollen wir die erstellten Artefakte evaluieren. Unser Ziel ist es, zu untersuchen, wie bestimmte Plattformfunktionen die Bewertung der semantischen Transparenz verbessern. Die Bewertungstechnik umfasst Aufgaben zur Einleitung, Begriffsassoziation, Notationsassoziation, Fallstudie und Feedback. Unser System nutzt fortschrittliche Bild- und Texterkennungstechniken, um nicht nur Vorschläge für neue Notationen zu unterbreiten, sondern auch die Effizienz der Bewertung erheblich zu verbessern. Wir sind bestrebt, kritische Anforderungen wie effiziente Anpassung, Verbesserung der Notation und Einbeziehung der Teilnehmer zu erfüllen und gleichzeitig die Beschränkungen herkömmlicher Papier-und-Stift-Konfigurationen zu überwinden.



Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar
The approved original version of this thesis is available in print at TU Wien Bibliothek.

Abstract

This thesis addresses the challenge of evaluating and improving the semantic transparency of graphical notations of conceptual modelling languages. Traditional approaches to modelling language specifications must consider essential notational aspects to effectively reflect domain-specific requirements. We propose a graphical notation evaluation platform for empirical graphical notations evaluation to automate the setup, execution, and result analysis of evaluation experiments, thereby enhancing reproducibility and efficiency. By employing the Design Science Research methodology and agile development methods, we aim to evaluate the built artefacts. Our goal is to investigate how certain platform features enhance semantic transparency evaluation. The evaluation technique comprises initiation, term association, notation association, case study, and feedback tasks. Our system, leveraging advanced image recognition and text recognition techniques, not only advances proposals for new notations but also significantly improves evaluation efficiency. We strive to meet critical requirements such as efficient customization, notation improvement, and participant involvement while effectively addressing the limitations of traditional paper-and-pen setups.



Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar
The approved original version of this thesis is available in print at TU Wien Bibliothek.

Contents

Kurzfassung	ix
Abstract	xi
Contents	xiii
1 Introduction	1
1.1 Problem Definition	1
1.2 Expected goals	2
1.3 Structure of the Work	4
2 Foundations	5
2.1 Conceptual Modelling	5
2.2 Visual Aspects in Conceptual Modelling	6
2.3 Semantic Transparency	8
2.4 Notation Evaluation and Improvement Technique	9
3 State-of-Art	13
4 Research Methodology	17
4.1 Process artefact	17
4.2 Product artefact	19
5 Graphic Notation Evaluation Platform	21
5.1 Software Architecture Overview	21
5.2 GNEP Data Model	23
5.3 Features and Functionalities	23
5.4 Implementation of the Main functionalities	26
5.5 User Interaction	41
6 Evaluation	45
6.1 Illustrative Scenario as Design Science Research Method	45
6.2 Results Evaluation	46
6.3 Research Questions (RQ)	64
	xiii

7 Conclusions	71
List of Figures	73
List of Tables	75
List of Algorithms	75
Bibliography	77

Introduction

This chapter explains the fundamental concepts of modelling methods, which include abstract syntax, concrete syntax, and semantics. It also discusses the critical role of method engineers in ensuring that domain-specific knowledge is accurately reflected in the abstract and concrete syntax of modelling languages.

The chapter also discusses the challenges in evaluating and improving the intuitiveness of graphical notations within Domain-Specific Modeling Languages (DSMLs). It concludes by pointing out gaps in current practices, such as insufficient involvement of target audiences and lack of design justification, which hinder the effectiveness of notation evaluation and improvement. To address these issues, we propose a web-based environment for automated, empirical evaluation of modelling language notations to improve semantic transparency and understanding of graphical notations.

1.1 Problem Definition

Modelling methods contain two main components: a modelling technique, which includes a modelling language and a modelling procedure, and mechanisms and algorithms that operate on the models described by the modelling language [1]. When taking a closer look at the elements of the conceptual modelling language, one can see that it consists of abstract syntax, concrete syntax, and semantics. The abstract syntax defines metamodel concepts [22]. A concrete syntax defines a textual or visual representation (notation) of a model [13], and semantics refers to the meaning of the abstract syntax elements. Most modelling language specifications concentrate on the syntactic aspects. However, notational aspects are essential when extending a modelling language to reflect domain-specific requirements.

A method user applies the modelling method by creating models using the modelling language, following the modelling procedure and applying the available mechanisms.

The method engineer is responsible for a consistent and adequately defined modelling method. They ensure that domain-specific knowledge is translated by the language's abstract syntax and graphical notations. In addition to technical skills, the method engineer often possesses professional skills in application domains, which can be verticals such as financial services, telecommunications, public administration, and manufacturing, or horizontals such as business process modelling, application development, workflow management, and knowledge management [1].

An intuitive modelling language notation establishes effective communication between its user and method engineer. It can be the first precondition for ensuring that the modelling language effectively communicates its message to business-oriented experts. To do this, the method engineer has to ensure that domain-specific knowledge is translated not only by the language's abstract syntax but also by graphical notations.

When defining a visual notation, one usually relies on intuition and existing standard practices rather than a specific scientific approach. It is not easy to verify if the intuitiveness of the visual notation is achieved. Modelling language method engineers lack support when evaluating and improving Domain-Specific Model Language (DSML) notations. The stated gap can be better understood when looking into a conceptual model evaluation process, which can only be done against business and people's needs and expectations, compared to a software product that can be evaluated against the specification. The difficulty only increases once the intuitive understanding of the visual notation has been ensured.

Several approaches [2, 5] support the development of new DSMLs and their evaluation. Each of them considers notation aspects to a different extent. It is essential to mention that most of them provide generic guidelines and recommendations for developing DSMLs, but concrete support was not provided [2]. In the approach described in [5], a few guidelines for textual and graphical concrete notation are given, which could be reasonably helpful and used by a method engineer while designing or by a user when evaluating graphical notations. Nevertheless, one might use an approach based on metrics to evaluate intuitively understandable notation empirically.

The current design practice is characterised by the lack of involvement of the target audience, minor form variations of geometrical notation shapes used for different purposes, a lack of design justification, and an unselfconscious design approach [11]. The existing approaches [2, 6] do not provide automation support for the empirical evaluation. As a result, there is a high manual effort in setting up the evaluation technique experiments and the execution and analysis of the obtained results. Further manual effort is required to derive assessment results and learn about improvement possibilities.

1.2 Expected goals

The current work seeks to provide a web-based environment for empirical modelling language evaluation. The designed environment will enable an automated setup for

evaluating and improving the semantic transparency of modelling language notations [2]. It aims to facilitate the technique's reproducibility by the modelling community and address a lack of efficiency in an experiment setup, execution, and analysis of results. Also, it is essential to investigate if the models with higher semantic transparency correlate with a better understanding of the modelling content by a model user.

The evaluation procedure consists of five sequential tasks: initiation, term association, notation association, case study, and feedback. During the initiation, the participant's demographic information is collected. The term association task requests a user to make drafts of representations of the terms of the given modelling domain. During the notation association task, a participant has to provide an intuitive association to the provided notations. Next, one has to create or interpret models and participate in the feedback survey. The procedure could be executed iteratively, and the data would be stored in a database for further analysis.

The system will use image recognition techniques to advance proposals for new notations and text recognition to improve the evaluation of the initial notations.

The goal is to create an environment where method engineers can be involved in setting up a technique for the notation evaluation, executing a procedure of applying it, and receiving the suggested results.

We aim to satisfy the initial requirements of the notation evaluation and improvement techniques provided in [2], which are:

- efficient customization
- notation improvement
- efficient use
- involves participant suggestion
- semantic transparency
- technique independence
- modular structure

We will address limitations and topics related to the paper-and-pen setup. The evaluation sheets and reports will be automatically generated based on the setup parameters.

It is crucial to deliver a sufficient means for a user of the technique to draft graphical representations according to the modelling domain. For that, a WYSIWYG web editor will be provided. The evaluation experiment's conduction is designed so that a participant is not influenced by any external factors while evaluating to ensure the quality of the suggestions.

1.3 Structure of the Work

We divided the structure of the work into the following chapters:

- **Foundations** chapter describes the necessary background knowledge to work on the web implementation of the notation evaluation technique, like conceptual modelling, DSML, and semantic transparency.
- **State-of-Art** chapter presents works related to our research to show existing approaches in notation evaluation regarding the problem definition.
- **The Research Methodology** chapter describes used research methods and approaches to building and evaluating created artefacts.
- **The Graphic Notation Evaluation Platform** chapter presents the implementation of the solution to the stated problem.
- **Evaluation** chapter presents the evaluation process with the results for tasks of the evaluation technique mentioned in the research questions.
- **Conclusions** chapter summarizes the results of the current work and presents future improvement possibilities.

Foundations

This chapter covers the foundational concepts of conceptual modelling and the visual aspects of modelling languages. It discusses domain-specific and general-purpose modelling methods, the components of modelling languages, and the importance of semantic transparency in visual notations.

It also introduces an iterative evaluation technique for improving visual notations, particularly in Domain-Specific Modeling Languages (DSMLs). This technique involves evaluating initial notations, revising based on feedback, and comparing to measure improvements in semantic transparency. The goal is to optimize notations for better understanding and usability, ensuring effective communication of intended meaning to users.

2.1 Conceptual Modelling

Conceptual modelling methods help simplify complex concepts using abstraction to achieve a specific objective. There are two main types of conceptual modelling methods: domain-specific and general-purpose. The purpose of domain-specific modelling methods is to focus on and address the essential elements of a particular field or domain. In contrast, general-purpose modelling methods prioritise comparability, interoperability, re-usability, and standardization across domains [22]. One distinction lies in the application domain: In computer science, modelling methods are designed with the goal of model-driven systems development, which often requires proper visualisation and focuses on model transformation and code generation capabilities. Meanwhile, modellers use information science or knowledge management modelling methods to develop model-driven systems and create abstract representations for "purposes of understanding and communication" [22].

When it comes to comprehensive modelling, there are three key components: a modelling language, a modelling procedure, and mechanisms and algorithms. The modelling language is the method's foundation and can be broken down into syntax, notation, and semantics[22].

It is essential to note that General-Purpose Modelling Languages (GPMLs), such as the Unified Modelling Language (UML), differ from DSMLs based on their scope and level of abstraction. GPMLs are designed to be widely applicable across different domains and industries, aiming to provide a standardised set of modelling concepts and notations that can be used in various contexts.

Since GPMLs cater to diverse stakeholders and purposes, finding test subjects with a comprehensive understanding and expertise in all areas covered by these languages can be challenging. Additionally, GPMLs often utilise abstract and generic concepts, making it more difficult to identify and assess semantically transparent notations within these languages.

DSMLs are specialised languages designed to solve specific problems within a certain domain. Developing DSMLs requires close collaboration between language developers and end-users, experts in the domain. While developers provide technical knowledge, end-users contribute to language concepts and notation that best suits the domain. For instance, the article [33] presents a DSML called RT-Sequencer that evolved from the DSML Sequencer to support the Data Acquisition domain and the Real-Time Control system. Involving end-users enriches the process and ensures that the resulting language meets their needs [23].

2.2 Visual Aspects in Conceptual Modelling

Visual notations are meant to support human communication and problem-solving. It is important to optimise them for the human mind to achieve maximum effectiveness. This concept is now known as cognitive effectiveness, measured by speed, ease, and accuracy, with which the human mind could process the representation. Cognitive effectiveness determines the ability of the visual notation to convey information to the stakeholders and support design and problem-solving by software engineers. Cognitive effectiveness is a widely accepted assumption in the IT field but is not an intrinsic property of visual representations. It needs to be intentionally designed into them. Just presenting information in a graphical format does not guarantee its effectiveness. Significant differences exist between effective and ineffective diagrams; ineffective ones can be less effective than plain text [4].

According to the decoding theory [4], human graphical information processing consists of perceptual and cognitive phases.

Perceptual processes are fast and automatic (seeing), whereas cognitive processing is relatively slow and requires attention control, referred to as understanding, as shown in Figure 2.1. To maximise cognitive effectiveness, one has to optimise the notation for

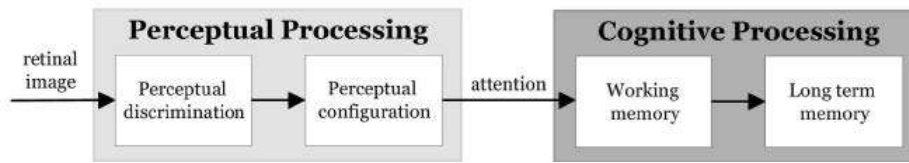


Figure 2.1: Perceptual and cognitive processing [4]

processing by the human mind. When shifting some processing power from the cognitive system to the perceptual, we free up the resources for the other tasks [4]. That is why designing cognitively adequate visual notations is of great importance.

Moody [4] defined a set of principles to support designing the cognitively effective visual notation. By doing so, visual notation design was transformed into a design discipline from the unconscious process. Figure 2.2 shows a visualisation of principles for designing cognitively effective visual notations [4].

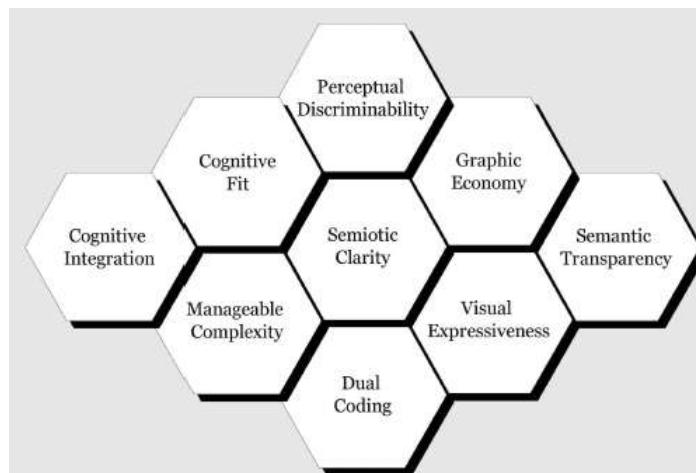


Figure 2.2: Principles for designing cognitively effective visual notations [4]

One principle worth mentioning is the principle of Semantic Transparency, which we elaborate on in the next Section 2.3. It implies how easily a symbol's meaning can be understood based on its appearance. This concept helps formalise natural or intuitive visuals and can be evaluated through experiments.

In addition to Semantic Transparency, several other principles are crucial for designing cognitively effective visual notations. Listed below, these principles are ordered by their importance:

- Principle of Semiotic Clarity
- Principle of Perceptual Discrimination

- Principle of Complexity Management
- Principle of Cognitive Integration
- Principle of Visual Expressiveness
- Principle of Dual Coding
- Principle of Graphic Economy
- Principle of Cognitive Fit

2.3 Semantic Transparency

The Physics of Notations (PoN) has a principle recognised as one of the most effective tools for notation designers to enhance the understanding of novices regarding notation semantics [4]. It is semantic transparency, and it helps formalise the intuitiveness of graphical notation and shows to what extent a notation implies its meaning to the user. It has been observed that symbols with built-in mnemonics are semantically transparent and can significantly reduce cognitive load, allowing their meaning to be perceived directly. This kind of representation improves recognition and understanding for novice users [15].

Semantic transparency is not a binary concept but a continuum as depicted in Figure 2.3 [15]. At one end of it, a symbol's appearance accurately conveys its meaning to a novice reader (for example, a stick figure representing a person). On the other hand, a symbol's meaning requires conscious effort to remember (for example, a rectangle to represent a UML class). At the opposing end of the continuum, a symbol's appearance may mislead a novice reader into thinking it means something else (for example, a red hexagon to indicate "start") [15].

According to Moody et al. [4], signs can be categorised as *semantically immediate* or *transparent*, *semantically translucent*, *semantically opaque*, and *semantically perverse* on the semantic transparency continuum. Semantically immediate or transparent signs can be understood easily, while semantically translucent hints at their meaning. Semantically opaque signs are purely conventional and require prior knowledge or understanding of the context to interpret them correctly. On the other hand, semantically perverse signs convey a meaning opposite to what they intended to convey.

When evaluating semantic transparency, one should distinguish between two different scopes of conceptual modelling languages: general-purpose and domain-specific. DSML is a language tailored to a specific application domain that offers appropriate notations and abstractions, and GPML is abstract and used for various modelling purposes by diverse stakeholders. The latter makes it more challenging to evaluate the intuitiveness of GPML compared to DSML. Evaluating intuitiveness is more achievable when designing a new DSML as potential users are involved in the designing process; however, it is crucial to reflect domain requirements in the syntax and intuitive notation.

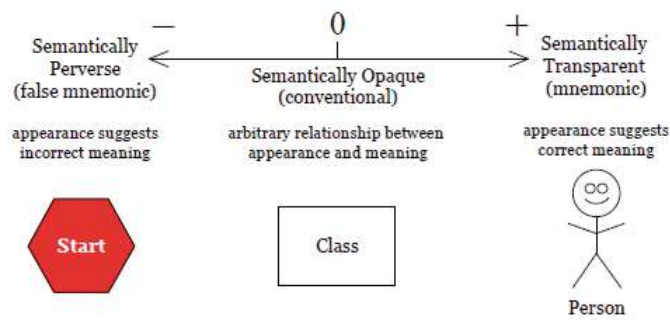


Figure 2.3: Semantic Transparency is a Continuum [15]

However, as observed in [16], the relationship between cognitive effectiveness and semantic transparency was only confirmed for some of its variables, not all. It may be related to the interrelations of all principles, defined by Moody [4] and other factors. Improved semantic transparency in the concrete syntax may not always result in better model understanding, despite the context provided by the model and language key when available. It's important to remember that semantic transparency is only one of the nine principles of the PoN [17].

Determining the level of semantic transparency is often a subjective process. Experts, such as researchers or designers of notation, attempt to estimate how likely someone new to the subject will understand the meaning of certain symbols. However, experts may not be the best judges, as it can be challenging to think like someone new to the subject [15].

2.4 Notation Evaluation and Improvement Technique

An evaluation technique for notation intuitiveness, proposed in [6], and its extended version for a notation evaluation and improvement technique [2] provide means for method engineers to improve visual notations. The mentioned techniques use an iterative evaluation approach to improve the semantic transparency of graphical notations. They have been applied to evaluate and improve the notation of the Process-Goal Alignment (PGA) modelling language [12]. As a result, several notation improvements were suggested, and business practitioners evaluated their semantic transparency.

We use the extended notation evaluation and improvement technique as a baseline for the current research [2]. The technique can be applied iteratively, allowing method engineers to improve graphical notations, if necessary, continuously. Each iteration consists of two phases, each containing a set of tasks. The first phase aims to evaluate the initial notation, and the second one aims to improve the evaluated notations, considering the results of the first phase. The first phase utilises user participation, and the second phase uses empirical evaluation to help method engineers evaluate and improve semantically transparent modelling language notations.

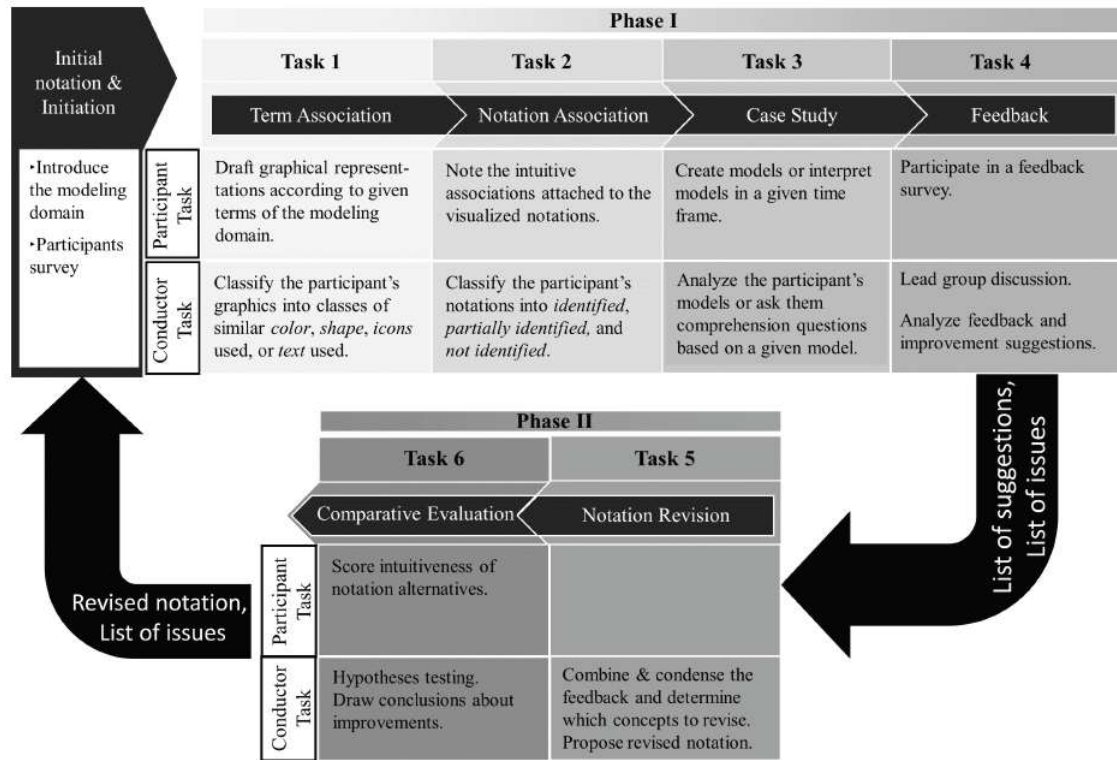


Figure 2.4: Procedure of applying the notation evaluation and improvement technique [2]

2.4.1 Phase I: Evaluation of the Initial Notation

Four core tasks have been suggested to assess the initial notation as shown in Figure 2.4: *term association*, *notation association*, *case study*, and *feedback*. Participants must complete these tasks to avoid any possible interference [2].

Initiation Task Background information is collected from the participants, which includes classical demographic aspects and questions about their experience with modelling languages, modelling tools, and the relevant domain.

Term Association Task During the Term Association Task, participants are given modelling language concepts and asked to draw up to three graphical representations that they find the most intuitive for each term's meaning. They are provided with paper and coloured pencils to create their sketches. The conductor then categorises the returned notation drafts based on visual variables such as position, shape, size, colour, brightness, orientation, and texture. Frequency analysis is done to calculate the instances of specific visual elements, which can reveal possible inadequacies and suggest improvements in the initial notation. The most frequently created notations can provide insights into potential

dominant notations. Automation of the conductor's task is achievable by digitizing participants' sketches and applying graphical classification algorithms.

Notation Association Task The Notation Association Task involves showing participants various modelling language concepts and asking them to provide up to three associations that come to mind for each notation. Participants are not given any information about the name or meaning of the concept to ensure unbiased evaluation. The conductor then examines the responses and categorises them based on whether they correctly identify the meaning of the notation, partially identify it, or do not identify it at all. This analysis helps determine the percentage of participants with matching associations and the relative ranking of each association, indicating the distinguishability between different notations. Natural language processing techniques can automate the analysis by quantifying identified concepts and efficiently identifying recurring terms.

Case Study Task For the Case Study Task, participants will be asked to use modelling language concepts to create a model based on a given case. This can be done using a modelling tool or with a sample model and comprehension questions. The goal is to evaluate the semantic transparency of the language concepts used in context. Once submitted, the models will be analysed for semantic and syntactic correctness, identifying any semantic errors, syntactic errors, or incomplete models. In addition, the responses to the comprehension questions will be categorised as fully correct, partially correct, or wrong. This analysis allows to calculate the percentage of correct answers for each modelling language concept. For larger groups of participants, comprehension questions can be presented as multiple-choice questions to facilitate fully automated analysis.

Feedback Task Participants must give feedback on the evaluation tasks and the case study solution in the Feedback Task. The survey focuses on the modelling language's quality, the modelling tool's usefulness, and suggestions for improvement. The feedback given on specific modelling language concepts is analysed to identify areas that require notation revisions and to guide the direction of such revisions. This feedback provides valuable input for improving the overall modelling process.

2.4.2 Phase II: Notation Revision and Comparative Evaluation

Notation Revision Task The conductor should review the initial notation of the concept that received a low score in the notation association task and case study task, along with the negative feedback. To improve the revision, the conductor can utilise the visual breakdown of participants' graphic representations in the term association task. Additionally, concrete suggestions for improvement are provided in the feedback task.

Comparative Evaluation Task The initial and the revised notation are compared to determine the semantic transparency of DSML. The dependent variable is the Semantic Transparency Score (STS), measured on a scale of 0 to 100, with 50 indicating no

2. FOUNDATIONS

preference. Participants are asked to evaluate both notations and provide their preferences. The hypothesis states that the revised notation will outperform the initial one in terms of semantic transparency. The study employs a within-subjects design with a graphical rating scale for STS measurement. The required sample size is calculated based on effect size and statistical power. Various statistical analyses, such as parametric and non-parametric tests, are used to evaluate the data and determine improvements in semantic transparency. The study also examines the potential confounding effects of demographic variables, such as gender, education level, modelling expertise, and time. The research aims to identify elements requiring notation revisions and assess the impact of proposed improvements on semantic transparency.

State-of-Art

The chapter describes current research and methodologies related to designing and evaluating modelling language notations, focusing on semantic transparency and its impact on visual communication. It discusses experiments assessing semantic transparency, highlights the Physics of Notations (PoN) principles, and mentions concerns about traditional metrics. Additionally, it examines a study on syntax changes and concludes by emphasizing the need for user-centred evaluation methods to ensure adequate visual notations.

Several works focus on designing and evaluating modelling language notations, regardless of the modelling language. A set of experiments applied to the modelling language was designed to empirically evaluate semantic transparency in [10]. Initially, participants are asked to sketch an intuitive notation for the provided element and finally select the most intuitive one. The semantic transparency experiment was introduced as an extension of [11] and applied to the UML. Participants sketched samples that were analysed to identify stereotypes and prototypes.

According to [18], there are no clear instructions on how to put the principles of PoN into practice. Additionally, despite their thoroughness, some have criticised the informal nature of these guidelines. One concern is whether they can be effectively replicated and verified systematically. The semantic transparency principle suggests that the user is directly involved in determining if a particular symbol conveys its intended meaning. The authors have concluded that implementing principles only covers two out of nine. These two principles, *perceptual discriminability* and *visual expressiveness* are the best options for implementation because they offer clear, measurable criteria and require the least subjective interpretation. On the contrary, a significant problem of PoN operationalization is that more specific satisfaction criteria must be met. For instance, the suggestiveness of the semantic meaning by the graphical notation can only be evaluated empirically by involving users [18].

Many studies have employed comprehension tests to evaluate semantic transparency (ST). These tests are also known as blind interpretation studies or recognition tests [18]. In a test, participants must match a sign with its corresponding meaning from a set of provided answers. Matching signs with concepts are more likely to involve guessing, unlike open-ended questions requiring testing. A name recognition test is more appropriate than a comprehension test, as participants need to recognise the concept rather than necessarily comprehend it [16].

The hit rate metric is commonly used to assess comprehension by analyzing questionnaire data from such recognition tests. This metric calculates the percentage of correct answers per sign, and signs reaching the comprehensibility threshold of 67%, as defined in ISO 9186:2001, are self-explanatory. In these tests, novices matched signs to their corresponding concepts [16].

The semantic transparency coefficient (STC) metric was also applied using the same test. Perceived ST was measured using three metrics. In three evaluations, participants were asked to rate their level of agreement or disagreement concerning the semantic transparency of a notation's concrete syntax. Similarly, in the other three evaluations, participants rated their level of agreement or disagreement regarding the semantic transparency of each sign in a notation [16].

The conventional hit rate metric may not be entirely reliable for evaluating ST, considering potential issues with the internal validity of research instruments. Recognition tests may yield higher rates of correct answers due to guessing or familiarity, making the metric less valid as it lacks defined thresholds for different ST states. The STC metric requires further testing and validation, particularly for semantically transparent signs. Expert analysis was employed in almost a third of evaluations, but the verifiability of such assessments is compromised without a protocol or numerical results.

The research methodology, introduced in [15], comprised five experiments to enhance user comprehensibility in requirements engineering notations. In the symbolization experiment, naive participants generated symbols for certain concepts, a task typically reserved for experts. Subsequently, a nonreactive stereotyping analysis identified the most common symbols produced, forming the stereotype symbol set. The prototyping experiment involved naive participants selecting the "best" representations and defining the prototype symbol set. The semantic transparency experiment evaluated the ability of naive participants to infer meanings from novice-designed symbols compared to expert-designed ones. The recognition experiment assessed participants' ability to learn and remember symbols from the four symbol sets. The authors introduced the semantic transparency coefficient as a new metric, offering advantages over traditional measures.

Notably, using explicit design principles significantly improved semantic transparency and cognitive effectiveness. Internal validity was upheld through controlled variables, external validity was achieved using naive participants, and statistical validity was achieved through verifying method assumptions [15].

This quasi-experiment study analysed the effects of changing the concrete syntax on

semantic transparency, understandability, and the ability to review models. In contrast to previous studies, the evaluation was performed at the model level instead of through isolated symbol recognition tasks. Novice participants, totalling 57, understood and reviewed tasks on models. The measurements taken included accuracy, speed, and ease of completion. Although the alternative i^* concrete syntax didn't improve accuracy or speed, it significantly reduced visual effort. Using these symbols, combined with a language key, seemed to alleviate the symbol recognition deficit frequently reported in prior research to the point where it had no noticeable effect on participants' performance in understanding and reviewing tasks [17].

In conclusion, the research on designing and evaluating modelling language notations highlights the significance of semantic transparency in ensuring effective visual communication. The research employed a set of experiments to assess the semantic transparency of different notations empirically. While the principles of PoN offer valuable guidelines for designing adequate visual notations, challenges arise in their practical implementation and operationalization. The principle of semantic transparency suggests the direct involvement of users in determining the meaning conveyed by graphical symbols, making user engagement crucial for evaluation. Various comprehension tests have been used to evaluate semantic transparency and the STC metric.

However, the reliability of conventional hit rate metrics in ST evaluation is questioned, calling for further validation of the STC metric. Expert analysis was employed in evaluations, but its verifiability is limited without specific protocols or numerical results.



Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar
The approved original version of this thesis is available in print at TU Wien Bibliothek.

Research Methodology

Design Science Research (DSR) focuses on solving essential and unresolved problems through unique, innovative, more effective, and efficient methods. When conducting DSR, the goal is to create a valuable and innovative solution to a specific problem. The solution must also be novel, meaning it solves a previously unsolved problem or improves upon an existing solution. Creating the solution involves searching for the most effective approach to the problem. It is crucial to thoroughly evaluate the solution to ensure it is effective [9].

DSR addresses research by building and evaluating the artefacts to meet research objectives. Building refers to creating something for a particular purpose, while evaluation involves determining how successful the creation achieves its intended goals [8].

We distinguish between design artefacts that are products and processes. Process artefacts are methods and procedures that help individuals achieve a particular task. A product artefact is a practical design, model, or approach implementation.

4.1 Process artefact

A process artefact is a technique to evaluate and improve graphical notations, introduced by Bork in [6] and improved Bork and Roelens in [2]. These are the initial requirements that the proposed technique should fulfil [2] :

- **Efficient customization** - The technique enables efficient customization for the modelling language.
- **Notation improvement** - The method should be applied to the initial notation to assess effectiveness. Improvement suggestions should be derived afterwards, and their impact should be empirically tested.

- **Efficient use** - Applying the technique should take a reasonable amount of time.
- **Involve participant suggestion** - Participants contribute to the notation improvement by proposing the most semantically transparent notation for a modelling language concept.
- **Semantic transparency** - The main focus of the technique is the semantic transparency of the graphics notation.
- **Technical independence** - The technique should apply without requiring technical infrastructure.
- **Modular structure** - The tasks should be structured modularly to allow for easy adjustments based on the current situation.

The evaluation of the technique was conducted using the PGA modelling method, which included a paper-and-pen setup. However, the authors stated that there is a possibility to automatise the current technique by performing specific tasks in a web-based platform, as mentioned in the paper [2]. They suggested that this platform automatically generates evaluation sheets once concepts and sample notations are uploaded. It will also provide a user-friendly web editor for drawing and saving notations. The system will analyse newly created notations automatically using OpenCV or similar technologies [32]. Text analysis will be implemented to evaluate notation associations, perform statistical analysis on responses, and generate evaluation reports.

Ultimately, this web platform will improve the setup, execution, and analysis of the technique, resulting in more efficient processes and better outcomes. A user study was set up to evaluate the technique. Participants were asked to assess the initial PGA notations using the procedure shown in Figure 2.4 and suggest improvements [2].

We use an illustrative scenario to evaluate a web-based version of a technique and compare it to the paper-and-pen method. An illustrative scenario in the context of DSR evaluation can be used to evaluate the effectiveness of the web-based platform [19]. We used the data gathered while evaluating the PGA modelling method to conduct an assessment.

As described in the paper [2], the output of the tasks for term association and notation association can be classified manually or automatically. While manual classification of participants' responses was carried out in the paper [2], our current research employs automated classification techniques to cluster participants' sketches and classify their responses. We will compare the results of the classifications of both approaches.

In conclusion, the web-based platform will support the execution of Phase I of the notation evaluation and improvement technique, depicted in Figure 2.4; it covers tasks where automation is feasible and will provide the most value:

1. Initiation Task: Participant survey

2. Term Association Task
3. Notation Association Task
4. Case Study Task
5. Feedback Task

Notably, there is a bidirectional connection between the evaluation of instantiations and the evaluation of models and methods that are embodied in the instantiations. By operationalising the models and techniques, we demonstrate their feasibility and effectiveness by building instantiation. It means one can also confirm the underlying artefact by evaluating the latter. On the other hand, assessing embedded models and methods contributed to the quality of the instantiation [27]. An evaluation of the notation evaluation and improvement technique by the illustrative scenarios using the PGA modelling method contributes to the quality of the web-based platform.

4.2 Product artefact

In the current research, a product artefact is a web-based notation evaluation environment, an instantiation. According to [20], an instantiation proves the idea's practicality and allows researchers to test their theories in real-life situations; this helps gain a better understanding of the real world.

The approach for software development is the Feature Driven Development method from the agile development methods [14]. It is based on user requirements later translated into product features and an overall model. The Feature Driven Development method is a rapid and iterative approach to assessing required features and adjusting artefact requirements in stages. This methodology facilitates the swift production and continuous revision of the prototype. Our primary goal is to create a product artefact that a method engineer can use to set up notation evaluation experiments automatically. However, human interaction is necessary with this artefact, making it a socio-technical tool.

The approach that was selected for constructing and assessing a product artefact is as follows:

1. Build a product prototype. The goal is to design the web-based evaluation platform prototype.
 - a) Define requirements and specifications.
 - b) Design an overall model.
 - c) Prepare a list of features.
 - d) Plan by features.
 - e) Conduct modelling iterations.

4. RESEARCH METHODOLOGY

2. Evaluation of the product artefact. We conduct two types of testing to ensure the smooth functioning of our system. Firstly, we perform functional testing to identify possible failures. Secondly, we conduct unit testing to verify the accuracy of the function's logic and ensure that everything functions correctly.

The following research questions are derived based on the described methodological approach:

- What are the appropriate means to support the semantic transparency evaluation method proposed in [2] by a web-based environment?
- To what extent can natural language processing techniques support the evaluation of semantic transparency in notation association task responses?
- To what extent can image processing techniques support the evaluation of semantic transparency in term association task responses?

Graphic Notation Evaluation Platform

This chapter provides an overview of the Graphic Notation Evaluation Platform (GNEP), a web-based tool designed to help method engineers evaluate and improve graphic notations. GNEP aims to replace traditional paper-based evaluation methods with an automated, efficient, and scalable solution. The architecture is based on a multi-tier design using the Django web framework, ensuring clear separation between data, business logic, and presentation layers, enhancing flexibility, security, and maintainability. The platform features a Model-View-Controller (MVC) pattern and a unified class diagram for configuring, executing, and analyzing notation evaluation tasks. It allows for configuring experiments, managing participant input, and generating detailed results. The platform also provides user interfaces for method engineers and participants, streamlining the evaluation process and contributing to improving graphic notations in modelling languages.

5.1 Software Architecture Overview

The Graphic Notation Evaluation Platform (GNEP) aims to provide means to the methods engineer to evaluate graphic notations and estimate their intuitiveness for further improvement. GNEP intends to replace the existing paper-based notation evaluation and improvement technique [2]. Also, with its development, we aim to contribute to the method engineers' community with a tangible artefact to set up experiments to evaluate the designed notations, conduct those experiments, and analyse data. We use the technique proposed by Bork and Roelens in [2] as a basis for requirements setting for the platform development. The data and software architecture models are created based on the specification of notation evaluation and improvement technique. The notation evaluation and improvement technique is fundamental in determining design decisions.

In this web application, we have adopted a multi-tier architecture for its efficiency, security, and scalability. The Django application employs the Model-View-Controller (MVC) architecture to divide code into input, business, and user interface logic modules. The architecture also ensures a clear separation between the layers, allowing changes in one layer not to affect others.

The Model handles data-related tasks, including data transformation and processing, to ensure consistency across the application. The Data Access Layer is defined in the `models.py` file. It forms the foundation for business logic, where data transformation occurs, enabling the handling of diverse user inputs for accurate responses. Business Logic Layer receives HTTP requests from the Presentation Layer, communicates with the database, and sends a response to the Presentation Layer. All business logic is developed in `views.py`.

The View is closely linked to the Model and displays data to users while collecting data through user input forms. Its dynamic nature allows the inclusion of interactive elements like animations and visualisations. The Controller, the architecture's orchestrator, manages the interaction between Views and Models, influencing the Model's behaviour and controlling user interface responsiveness.

The Presentation Layer is responsible for visualizing data and defining how data should be presented on web pages and other documents, using the Django system template with Bootstrap, HTML, and JavaScript [21].

Figure 5.1 illustrates the high-level software architecture of GNEP, which we developed using the Django web framework. The browser sends an HTTP request to the web server. When a web server receives a request, it is passed on to the WSGI server. This software component implements the WSGI specification and mediates between a web server and a Python web application or framework. Then, it calls the appropriate Python code to generate a response. The URL setting in the `urls.py` file selects the view based on the URL in the request. The view interacts with the database through `models.py`, renders the HTML or other formats, and returns an HTTP response. The WSGI server then returns the response to the web server, which sends it back to the client with the requested page [21].

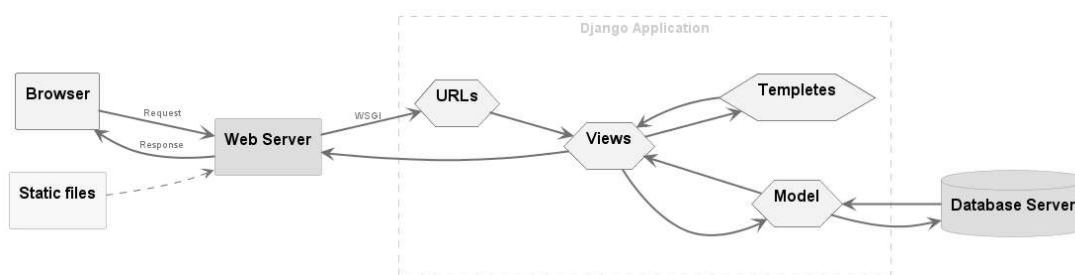


Figure 5.1: Specific Django Project Architecture

5.2 GNEP Data Model

The Graphic Notation Evaluation Platform Architecture uses the data model derived from the notation evaluation technique description and its evaluation illustrated in the paper[2]. Figure 5.2 depicts the data model as a Unified Modelling Language (UML) class diagram. The class diagram visually represents the system’s structure. It includes all the essential classes required to configure, set up, conduct the execution of the notation evaluation technique tasks [2], and obtain their results.

Firstly, we describe classes needed for experiment creation and its setup. *Experiment* class defines the unique configuration of the notation evaluation technique application and the sequence in which tasks will be executed. *Experiment* class is composed of *ExperimentTask* classes. The instances of *ExperimentTask* are predefined and automatically generated by the system: Initiation, Term Association, Notation Association, Case Study, and Feedback. The method engineer, represented by the *MethodEngineer* class, can configure experiments based on their specific requirements and reuse task configurations across multiple experiments. It is made possible through the *ExperimentTaskBase* abstract class, which defines the main properties of the task and is inherited by the *ExperimentTask* class.

To enhance the customization of *ExperimentTask*, we have developed an abstract class called *ExperimentParameterBase*. Classes like *ExperimentParameterText*, *ExperimentParameterFile*, *ExperimentparameterMultipleText* and *ExperimentParameterQuestion* inherit its properties. By doing this, we have increased the flexibility of participant view customization. Additionally, we have incorporated the *ParticipantInputBase* class, representing the participant’s input in the experiment. This class further inherits *ParticipantTextInput*, *ParticipantGraphicalInput*, and *ParticipantQuestionInput* based on the experiment type to define the input to the respective parameter class. Finally, we have included *Time* class that measures the time spent on each task, providing valuable insights into the experiment.

5.3 Features and Functionalities

We developed the Graphical Notation Evaluation Platform using the Django framework, which uses Python to structure components like models, views, and templates. We incorporated certain Python functions in our views to enhance the application’s flexibility and extensibility while using specific JavaScript parts to improve the application’s logic further. As we used function-based views, we created the Application Layer diagram that clearly shows the functional components of the application, which is done in Figure 5.3.

Method engineers act as users of the General Notation Evaluation Platform. To initiate the access, they provide the required input to **User Interface for registration** interface, allowing them to access the **Experiment Interface**. This interface offers **Add Experiment** interface as an input option.

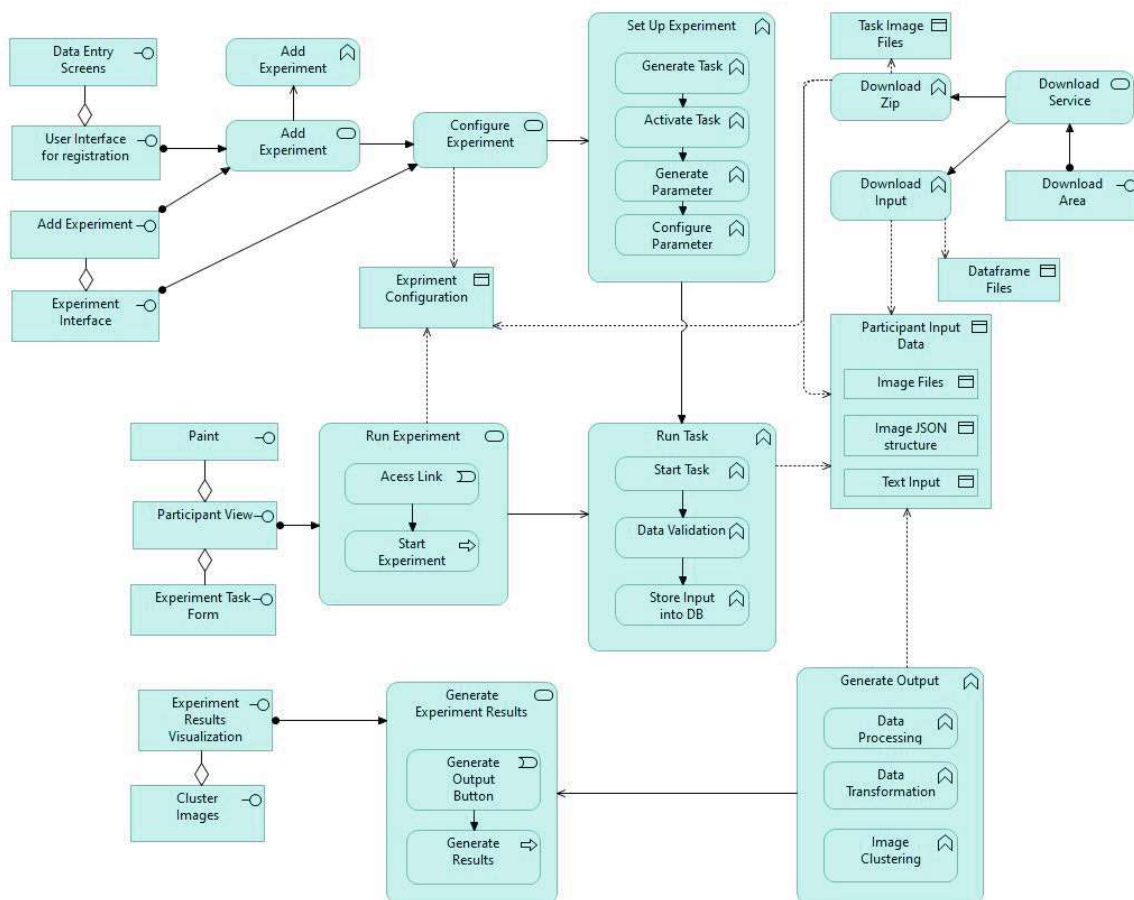


Figure 5.3: Application Layer of GNEP

The application offers the functionality to **Configure Experiment**. To trigger that functionality, the user has to add a new experiment to the system or reuse the existing one. The function **Add Experiment** supports creating the new experiment, which is realised through **Add Experiment** service.

The **Configure Experiment** Service is available via **Experiment Interface**; it facilitates the **Set Up Experiment** function. It includes the pre-configured sets of steps that must be done to finalise the experiment's setup. Those steps are available through **Experiment Interface**, and their execution is configured sequentially. In the **Generate Task** function, we create tasks that contain the logic of the notation evaluation technique [2]. The next step is to execute the **Activate Task** function, which allows the user to reuse configured tasks and activate only necessary tasks for the current run. The parameters for the experiment task are generated through the **Generate Parameter** function. The **Configure Parameter** function allows customization of the tasks to a certain extent, which is meant to support the initial requirement of the notation evaluation technique.

After finalizing the experiment setup, the **Participant View** can be accessed. It includes an interface for the **Experiment Task Form** and **Paint**, a custom tool for notation creation based on task requirements in [2]. The main functionality, which is accessible via the **Participant View**, is **Run Experiment** service. Accessing the experiment link provided by the method engineer triggers it. The process **Start Experiment** consists of the execution of each experiment task; it uses the existing **Experiment configuration**. The **Run Task** function is executed for every task set up in the **Configured Experiment** service. Firstly, we execute the **Start Task** function. After that, the input is validated with the run of **Data Validation** function. The data is stored in the database using the **Store Input into DB** function. **Participant Input Data** is stored after the execution of **Run Task** function. Two output types are: **Text Input** and **Image Files**. The **Generate Output** function uses Image Files. Images created with the **Paint** Interface are stored as an **Image JSON structure** to provide more insights into graphical input.

After gathering all necessary input from participants, the experiment can be finalised. The method engineer can generate results from the **Generate Experiment Results** service.

The **Generate Experiment Results** service provides a method engineer with an overview and insights into the data collected from the experiment participants; the output is available for each task as soon as a user accesses it via **Generate Output Button**. The process of generating results **Generate Results** involves common steps such as data processing and transformation, which are triggered by the respective functions **Data Processing** and **Data Transformation**. However, there may be variations depending on the task at hand.

Download Area interface allows the user to access **Download Service**. Based on the user's request, the service executes two functions: **Download Zip** and **Download Input**. After running the **Download Zip** function, images related to the selected task are downloaded. Upon running the **Download Input** function, a data frame containing information about the experiment tasks is generated.

Additional functionality is available for users via **Cluster Images** interface. It conducts a more in-depth analysis by clustering received images as a result of the experiment task.

5.4 Implementation of the Main functionalities

After defining the overview of software architecture, this Section offers a detailed description of each application service and its functions, displayed in Figure 5.3.

5.4.1 Add Experiment Service

The method engineer can create an experiment by calling the function `add_experiment()` to generate an instance of the *Experiment* class. It can be done by pressing the "Submit"

button on the screen from Figure 5.4. When it is being generated, the class's attribute status is set to 0, meaning the method engineer configures it.

Possible statuses of *Experiment* class, which are used to control output:

- "Configuring" status (value "0") indicates that the method engineer is configuring the experiment and should not yet be started;
- "Pending" status (value "1") indicates that the experiment is ready to be executed and can be accessed by participants;
- "In Progress" status (value "2") indicates that the experiment is in progress of collecting input;
- "Completed" status (value "3") indicates that the experiment is completed, and results for each task are generated for the use of the method engineer.



Figure 5.4: Add Experiment Service

5.4.2 Configure Experiment Service

The **Configure Experiment** Service provides functionality to customise the graphical notation evaluation technique run. To set up the experiment, a collection of functions is executed sequentially, such as `generate_task()`, `activate_task()`, `generate_parameter()`, and `configure_parameter()`.

The `generate_task()` function creates instances of the *ExperimentTask* class with specific attributes such as name, active, configured, generated, description, created_at, and priority, and associates them with a particular experiment, the value of the attributed are mentioned in the Table 5.1. These tasks are initially marked as inactive, not generated and not configured and represent activities related to the experiment as visible in Figure 5.5.

Each task is assigned a different value of *type* attribute; the *priority* corresponds to the sequence in which tasks should be executed.

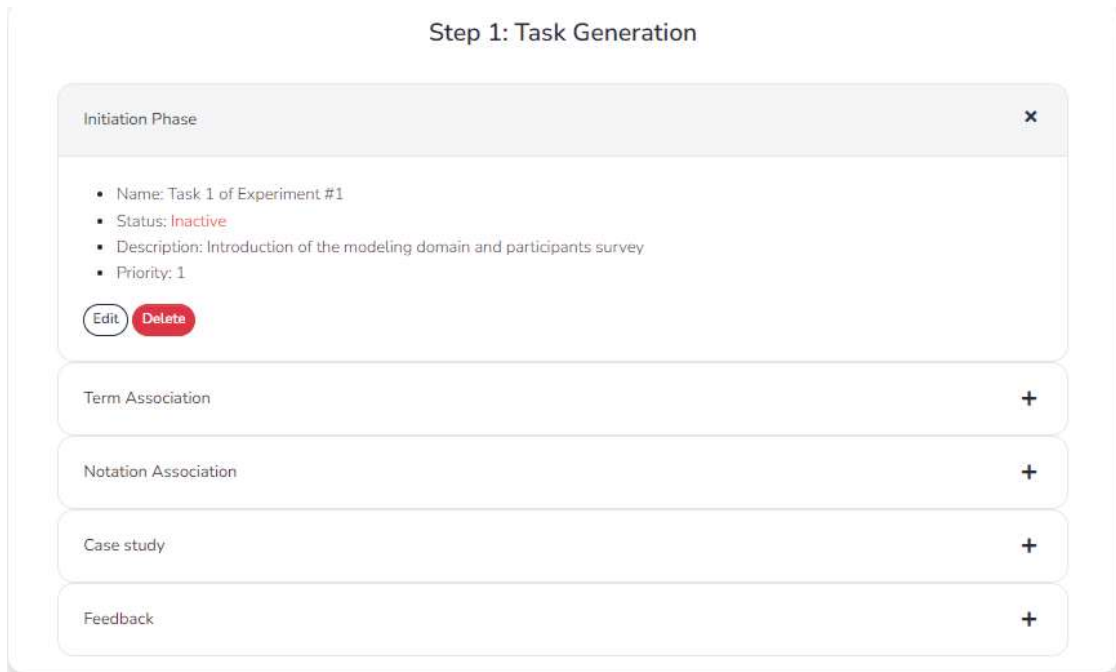


Figure 5.5: Task Generation Interface

Task Name	Active Flag	Configured Flag	Task Type	Generated Flag	Priority	Description
Task 1	False	False	Initiation Task	False	1	Introduction of the modelling domain and participants survey
Task 2	False	False	Term Association	False	2	Graph representation of given terms of the modelling domain
Task 3	False	False	Notation Association Task	False	3	Intuitive association of the visualised notations
Task 4	False	False	Case Study Task	False	4	Model interpretation or creation
Task 5	False	False	Feedback Task	False	5	Feedback survey of participants

Table 5.1: Task attributes after execution of `generate_task()` function

After generating predefined tasks, the method engineer can activate tasks needed for the experiment as the next step of the experiment set up by calling the function `activate_task()` as it is shown in Figure 5.6. The function retrieves necessary data from the database, prepares formsets for parameter customization, and validates the user's form data. If all forms are valid, the task *active* parameter is set, and the user is redirected to the next step of the experiment setup. Overall, `activate_task()` facilitates user-friendly task activation and efficient parameter management within an experiment. A detailed overview of the task attributes and their values can be found in Table 5.2.

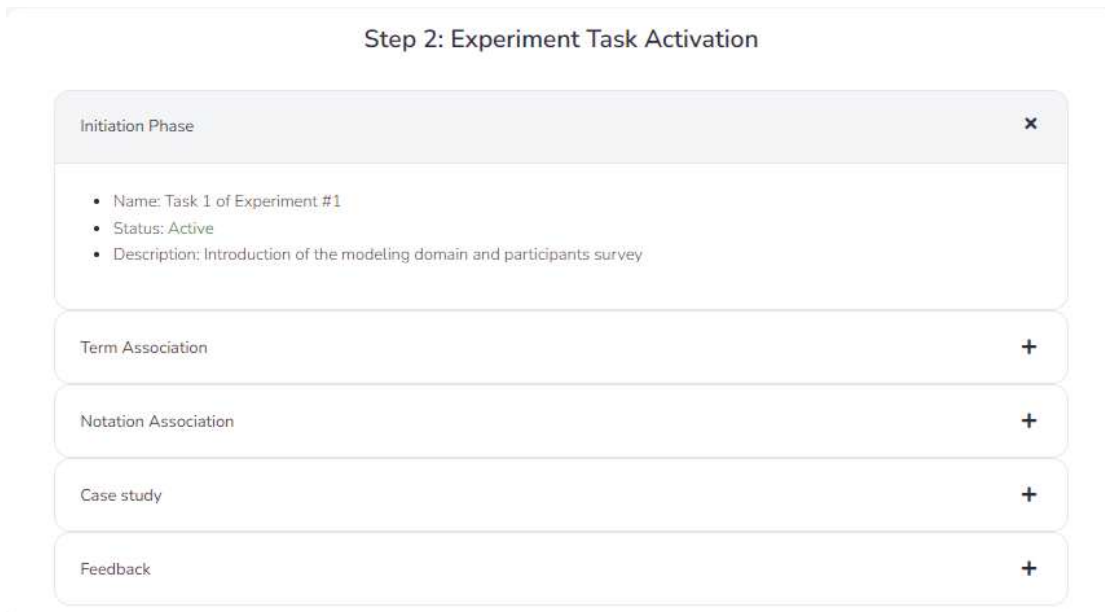


Figure 5.6: Task Activation Interface

Task Name	Active Flag	Configured Flag	Task Type	Generated Flag	Priority	Description
Task 1	True	False	Initiation Task	False	1	Introduction of the modelling domain and participants survey
Task 2	True	False	Term Association	False	2	Graph representation of given terms of the modelling domain
Task 3	True	False	Notation Association Task	False	3	Intuitive association of the visualised notations
Task 4	True	False	Case Study Task	False	4	Model interpretation or creation
Task 5	True	False	Feedback Task	False	5	Feedback survey of participants

Table 5.2: Task attributes after execution of `activate_task()` function

The `generate_parameter()` function dynamically creates experiment parameters based on the task type. Parameters are tailored to each task's needs, including descriptions, multiple-choice questions, file uploads, and more, as depicted in Figure 5.7. The function marks the task as *generated* and sends a success message to the user. Its goal is to simplify experiment parameter setup, improve flexibility and customization, and guide the user to the next step of the process. In Table 5.3, we have listed the parameters along with their respective class and task names to which they belong. We have divided the parameters into *Visible for Users* and *Not Visible for Users*. The former group contains parameters intended to be used by users, while the latter group includes parameters that serve more technical purposes. After executing this function, the *generated* attribute is updated

5. GRAPHIC NOTATION EVALUATION PLATFORM

to "true," indicating that the tasks and their parameters are ready for configuration, as could be seen in Table 5.4.

Parameter Name	Class Name	Task Name	Visible for User
Definition Of Domain	ExprimentParameterText	Task 1	True
Background Participant Information Question	ExprimentParameterQuestion	Task 1	True
Term	ExprimentParameterMultipleText	Task 2	True
Number Of Sketches Per Term	ExprimentParameterText	Task 2	False
Introduction of Task	ExprimentParameterText	Task 3	True
Notation	ExprimentParameterFile	Task 3	True
Number of Terms per Notation	ExprimentParameterText	Task 3	False
Introduction of Task	ExprimentParameterText	Task 3	True
Case Study Introduction	ExprimentParameterText	Task 4	True
Case Study Introduction From File	ExprimentParameterFile	Task 4	True
Case Study Question	ExprimentParameterQuestion	Task 4	True
Feedback Question	ExprimentParameterQuestion	Task 5	True
Show Task Summary to Participant	ExprimentParameterText	Task5	False

Table 5.3: Task Parameters after execution of function `generate_parameter()`

Step 3: Set Up Parameters - Task 1 of Experiment #1

Definition Of Domain

Input

Background Participant Information Question: Add Another Question

Select Question Type:

Single-answer ▼

Type Question Here

Answer option 1

Answer option 2

Answer option 3

Answer option 4

Answer option 5

Delete:

Save

Figure 5.7: Parameter Configuration Interface

The `configure_parameter()` function is written with the same logic as the function

`activate_task()`. Its primary purpose is configuring parameters for an experiment by fetching different parameter types related to the task. Before a method engineer submits a form, it is required to specify the values of the parameters. The function validates the form data, saves the modifications if valid, and marks the task as "configured". The function then redirects the user to the next step of the experiment setup. The `configure_parameter()` function facilitates parameter management and contributes to the application's functionality.

Task Name	Active Flag	Configured Flag	Task Type	Generated Flag	Priority	Description
Task 1	True	True	Initiation Task	True	1	Introduction of the modelling domain and participants survey
Task 2	True	True	Term Association	True	2	Graph representation of given terms of the modelling domain
Task 3	True	True	Notation Association Task	True	3	Intuitive association of the visualised notations
Task 4	True	True	Case Study Task	True	4	Model interpretation or creation
Task 5	True	True	Feedback Task	True	5	Feedback survey of participants

Table 5.4: Task attributes after execution of `generate_parameter()` and `configure_parameter()` functions

The last step of the experiment setup triggers the function that sets the experiment status to "Pending". Through the *Experiment Interface*, this set of tasks can be selected for further usage by participants.

5.4.3 Run Experiment Service

After a method engineer has finished configuring an experiment and stored its configuration in the database, a link to the *Participant View* interface is generated for participants to access. Once a participant accesses it, the experiment's status changes to "In Progress". This change in status serves as an indicator to the method engineer that the experiment is ongoing.

The `start_experiment()` function is triggered every time a participant accesses an experiment link. It checks if a user is logged in. If not, it creates a new participant user and assigns them specific role attributes. It then retrieves the experiment associated with the provided `experiment_id`, logs in to the user, and redirects them to the experiment view. This function ensures a clean session for participants and authenticates them before accessing experiment content.

The function `start_task()` is responsible for managing the initial interaction of a participant with a specific task in an experiment. When a participant requests to start a task, the function creates relevant input structures, such as text, question, and graphical inputs. These structures are customised for the participant, task, and relevant parameters.

The function also records the task start time for the participant. It ensures duplicate input structures are not created for the same participant and task to avoid redundancy. Furthermore, it identifies the type of task and redirects the participant to the appropriate view or page associated with the specific task type within the experiment.

To ensure better customization, a designated function is created for each task type. The `task_introduction()` function manages the introductory task of an experiment. It fetches essential data such as the experiment, task, and participant involved, retrieves relevant question parameters, and checks for user interaction. If any question is unanswered, it issues an error message, and if all questions are answered, it marks the task as finished, records the end time and provides a success message. Figure 5.8 displays an instance of the user questionnaire.

The function `task_term()` retrieves terms, which are multi-text parameters, and tracks the completion status. It provides access to the interface from Figure 5.9, which allows users to create images per respective term and store them in the database in JSON and PNG formats through a user-friendly interface. When a POST request is made, it validates the input data and marks the task as complete. For GET requests, it manages pagination for multi-text parameters.

What is your gender?

Female

Male

Not defined

Where were you born?

Your Answer

What is your education?

High-school

Bachelor's

Master's

Doctor's

What is your experience in modeling

1 2 3 4 5

Submit

Figure 5.8: Participant Questionnaire of Introduction Task

The function called `task_notation()` manages the participant's interaction with a notation association task in an experiment. It retrieves and processes several parameters, such as images of the notation. Once the participant submits their respective input, the

function validates and stores them.

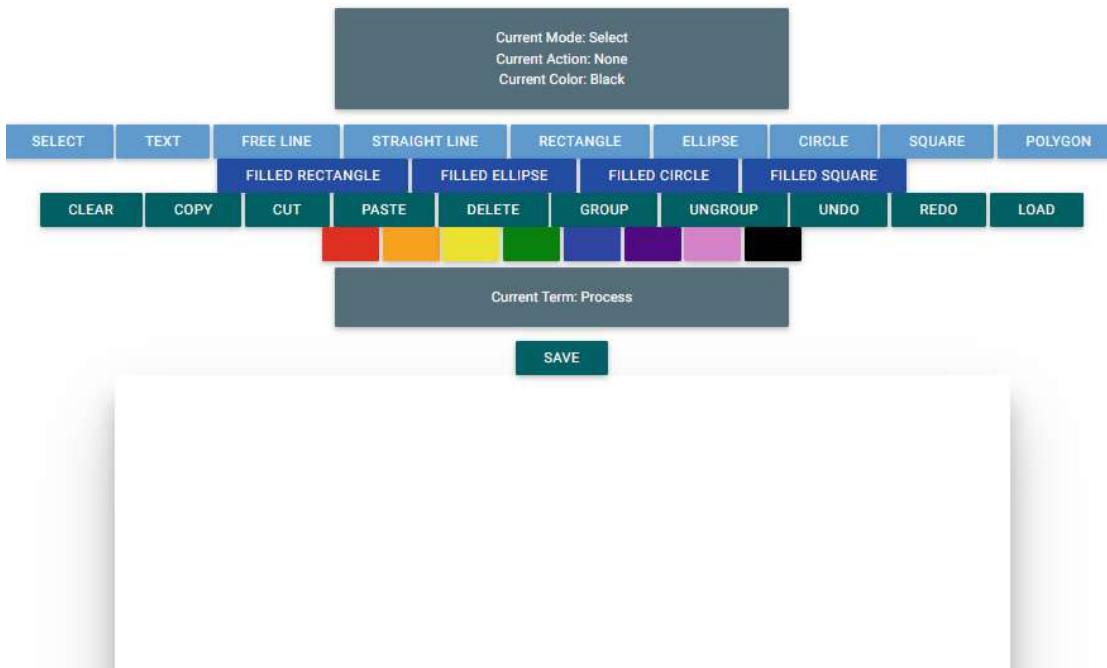


Figure 5.9: Paint Interface

Term Association Summary		
Task Term	Participant Notation	Graphical Notation
activity		
Notation Association Summary		
Task Graphical Notation	Participant Association	Graphical Notation Legend
	Process	Process
	Activity	Activity

Figure 5.10: Participant Summary in Feedback Task

The function named `task_case_study()` manages the participant's engagement with a case study task. It retrieves relevant parameters such as text and questions. The participant can submit their answers to the questions; the function validates and saves them.

The `task_feedback()` function handles the participant's feedback task in an experiment. After completing the Term and Notation Association Task, users will be presented with a summary of their input in a tabular form on the output page as shown in Figure 5.10. The function retrieves text parameters and question types similar to those in the case study task and displays the output accordingly. Participants can provide feedback for each notation they evaluate in the experiment.

After the finalization of the experiment, the participant input, which is stored in the database based on the format defined in the class diagram in Figure 5.2, is used by the **Generate Experiment Results Service**.

5.4.4 Generate Experiment Results Service

Executing the function `complete_experiment()` changes the experiment's status to "Completed" and triggers the **Generate Experiment Results Service**. The `generate_output()` function serves this service, and its logic is split according to the task type.

Initiation Phase Task

For the Initiation Phase Task, the function generates detailed visualisations of participant responses. It can process questions like single-choice, open-ended, multiple-choice, and rating scales. Based on the question type, we generate various visualisations from the participants' answers, such as histograms and word clouds. Figure 5.11 shows an example of the question and its visualisation. The function uses the Plotly [28] library to create interactive visualisations and the Pandas library to manipulate data. Furthermore, it can handle form submissions to update the visualisations dynamically based on user-selected columns. The function output aims to provide a clear overview of participant responses for a given task in the experiment, allowing method engineers to conclude.

Term Association Task

The function of the Term Association Task is primarily responsible for processing and preparing the graphical inputs the participants provide. These inputs are then analysed to extract the essential elements, including shapes and colours, which are then grouped. This process aims to identify and associate related elements within the graphical inputs accurately.

To achieve this, it fetches data from the database, including information about graphical input details and associated parameters. The image data is stored in the JSON format, depicted in Figure 1, and is then manipulated using the Python library Pandas [29], which

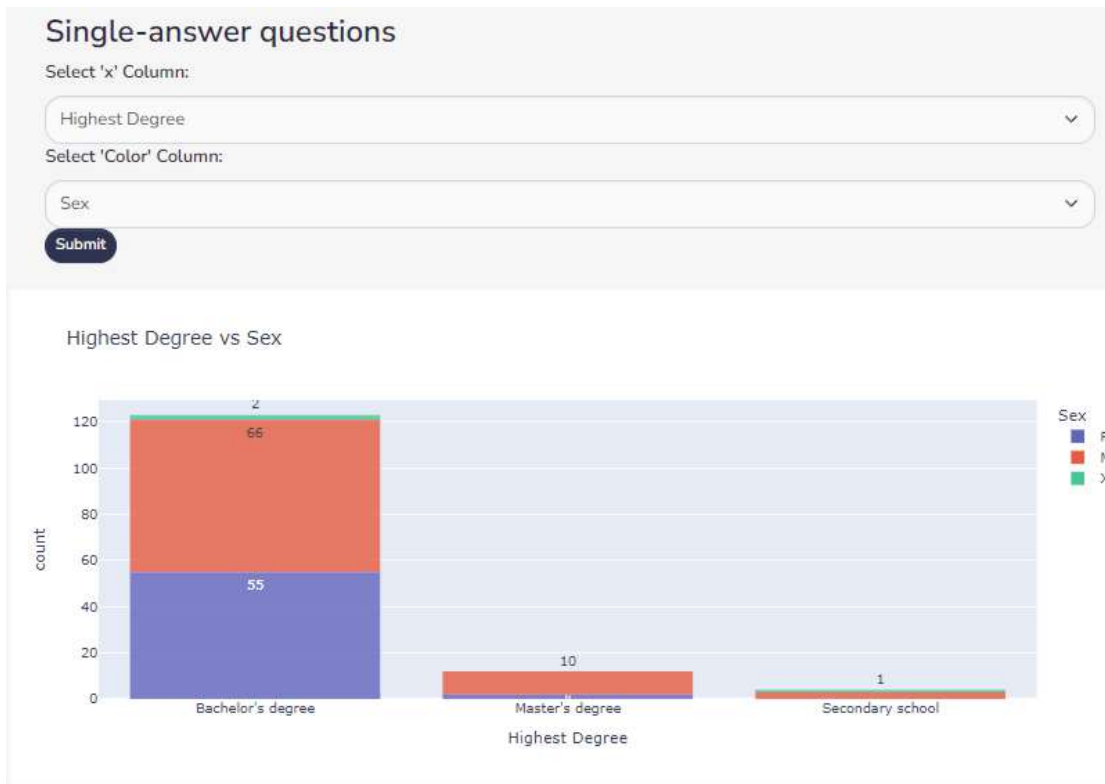


Figure 5.11: Initiation Phase Task Result Interface

flattens and merges it to create a structured data frame. Once the data is processed, it is analysed to extract information about the shapes (rectangles, circles, etc.) and colours used in graphical notations. The occurrences of shapes and colours are aggregated, and the results are visualised as shown in Figure 5.12.

The PNG image format is used as input for the contour detection model, which is edge-based [25]. It is designed to identify the basic shapes and colours of the sketched notations. The function `detect_contours()` is applied to contour recognition on each image. It uses the OpenCV library to detect the images' contours and recognise the basic shapes and colours based on contour information, such as area, length, and centre coordinates [24]. After identifying the contours in the images, the function `generate_output()` processes the results of the `detect_contours()` function. The processed data is aggregated and converted to JSON format for easier handling and stored in a context dictionary. If the request is an AJAX request, the function returns a JSON response with the processed data.

The user can trigger the function `image_cluster_configuration()`, which performs image clustering based on a set of input parameters such as ImageNet model [26], number of clusters and usage of Principal Component Analysis (PCA), and then it renders the results of image clustering into a template. The `image_clustering` class,

```
{
  "objects": [
    {
      "type": "rect",
      "originX": "left",
      "originY": "top",
      "left": 246.5,
      "top": 315,
      "width": 512,
      "height": 215,
      "fill": "",
      "stroke": "Blue",
      "strokeWidth": 3,
      "strokeDashArray": null,
      "strokeLineCap": "butt",
      "strokeLineJoin": "miter",
      "strokeMiterLimit": 10,
      "scaleX": 1,
      "scaleY": 1,
      "angle": 0,
      "flipX": false,
      "flipY": false,
      "opacity": 1,
      "shadow": null,
      "visible": true,
      "clipTo": null,
      "backgroundColor": "",
      "fillRule": "nonzero",
      "globalCompositeOperation": "source-over",
      "transformMatrix": null,
      "rx": 0,
      "ry": 0
    },
  ],
  "background": ""
}
```

Listing 1: Example of image data in JSON format

The screenshot displays two web interface sections. The top section, titled "Shape based on Contour Recognition", features a "Process" button and a "Show Output" button. Below these are export options (Copy, Excel, PDF, CSV) and a search input field. A table lists terms: "No.sketches", "Heptagon", "Hexagon", "Square", "Rectangle", and "Octagon/Circle". The "Process" row shows values: 1, undefined, undefined, undefined, undefined, and 3. Below the table is a pagination control showing "Showing 1 to 1 of 1 entries" and "Previous 1 Next".

The bottom section, titled "Shape & Color Info based on JSON Output", also has export options and a search input. Its table lists terms: "No.sketches", "Rectangle", "Circle", "Square", "Line", "Ellipse", "Indigo", "Black", "Blue", "Red", "Yellow", "Orange", "Green", and "Violet". The "Process" row shows values: 2, 3, and 3.

Figure 5.12: Term Association Task Result Interface

called in the function, has methods for loading images, generating new image vectors (features) based on ImageNet models or PCA, and performing KMeans clustering [30]. The results are then returned and displayed in the Django template for further analysis.

List of the ImageNet models that are available for usage:

- VGG16
- VGG19
- ResNet50
- InceptionV3
- Xception
- InceptionV3
- InceptionResNetV2
- DenseNet201
- MobileNetV2

We use pre-trained ImageNet models to perform image classification. ImageNet is a large-scale ontology of images organised hierarchically based on their semantic meaning in WordNet. It was created to help manage the overwhelming amount of image data available and to facilitate the development of advanced models and algorithms in computer vision. The paper [26] illustrates ImageNet’s usefulness through object recognition, image classification, and automatic object clustering applications, which aligns well with our purpose.

Notation Association Task

The function of the Notation Association Task aims to group and analyse terms that participants provide to identify graphical notations. It retrieves the text participant inputs from the database. The program first uses spaCy’s English-language pre-trained machine learning model for natural language processing to calculate similarity scores between the parameter legends, notation definitions, and the participants’ inputs [31]. This model can perform various NLP tasks, including text classification, making it suitable for our needs.

The resulting similarities scores are ordered into clusters using KMeans, and custom labels are assigned to each cluster based on their mean values. To describe the relationships between what is depicted by the graphical notation and what was understood by the participant, we specified the following cluster labels:

- fully identified
- identified
- partially identified
- not identified

In conclusion, the function assigns each participant input to a specific cluster based on its similarity score with the notation definition.

During the second part of the function, we compare Positive and Negative Term associations to every graphical notation specified by the method engineer with the participant input. We search for these associations within the participant term associations (participant input). This process involves converting all words to their base form and comparing the values of the participant association to the method engineer input.

After reviewing each participant’s input for positive and negative associations and identifying the cluster in the first step, we can calculate evaluation metrics for each term association.

In our case, it is a precision score (PS) and recall score (RS):

$$PS = TP / (TP + FP)$$

$$RS = TP / (TP + FN)$$

Based on the definitions of the Precision score and Recall score, we have established clear descriptions for True Positive (TP), True Negative (TN), False Positive (FP), and False Negative (FN) in our research.

- TP - is a term classified as "fully identified" and does not match "negative terms" and matches with the "positive term" specified by the method engineer;
- FP is a term that has a match with a negative association and has no matches with the positive association and is classified as "fully identified".
- FN is a term that is classified as "not identified", "partially identified", and "identified", matches with positive association and does not match with "negative term";
- TN is a term that is classified into "not identified", "partially identified", and "identified", matches with the negative association and does not match with positive ones;

F1 score is the harmonic mean of precision and recall scores, providing a balanced measure of the model's performance. It is calculated using the formula:

$$F1score = \frac{2 \times PS \times RS}{PS + RS}$$

Finally, the results are presented in a structured format on the Django template as depicted in Figure 5.13.

Case study Task and Feedback Task

The provided code includes a function in a Django web application that manages the processing and visualisation of participant inputs for a specific type of experiment task. This function is designed for tasks categorised as Case study and Feedback Tasks. It starts by retrieving participant question inputs and associated parameters from the database. If there are no inputs, a flag is set to indicate this condition.

After organising the data according to the types of questions, the function generates separate data frames for each question type: single-choice, open-ended, multiple-choice, and rating scales. The function then generates JSON representations of the data for rendering in the web interface. It involves aggregating and formatting the data, particularly for multiple-choice questions that use commas or semicolons to separate answers. Finally, the function uses a Django template to render the data, enabling users to view detailed results in a table-based form based on the type of questions (Figure 5.14).






Term	Notation	Positive Terms	Negative Terms	Precision Score	Recall Score	F1 Score
activity			financial structure; internal goal; importance; customer goal; value stream; financial goal; value proposition; competence; process;performance	1.0	0.17	0.29
color		importance, performance	activity; financial structure; internal goal;customer goal; value stream; financial goal; value proposition; competence; process	0.0	0.0	0.0
competence			activity; financial structure; internal goal; importance; customer goal; value stream; financial goal; value proposition;process;performance	1.0	0.19	0.32
customer goal			activity; financial structure; internal goal; importance; value stream; financial goal; value proposition; competence; process;performance	0.75	0.03	0.06
financial goal			activity; financial structure; internal goal; importance; customer goal; value stream; value proposition; competence; process;performance	0.77	0.21	0.33

Figure 5.13: Notation Association Task Result Interface

Single-answer questions

Search:

Question	Answer	Count
Is the model clear?	No	1
Is the model clear?	Not clear	1

Showing 1 to 2 of 2 entries Previous Next

Figure 5.14: Feedback Task Result Interface

5.4.5 Download Service

A user can trigger the execution of the Download Service by accessing the interface *Download Area*. It is serviced by two functions: `download_zip()` and `download_input()`.

Function `download_zip()` provides the user with the zip archive that contains graphic files associated with the selected task, including the graphical input from the experiment participants and the task's parameter notations. The button "Download files" triggers the downloading of the zip file.

Function `download_input()` creates the data frame that contains aggregated data of the participant input and parameter properties per each type of task. By pressing

"Download Participant Input," a user initiates downloading the respective task's CSV file. The interface for the Download Service is shown in Figure 5.15.



Figure 5.15: Download Service

5.5 User Interaction

The platform is primarily designed for two categories of users - method engineers and experiment participants. To better understand the interaction of these users with the platform, we have used sequence diagrams. These diagrams show how different parts of the system interact with each other over time, carry out the required actions, and complete processes. A sequence diagram descends from top to bottom, showing a sequence of interactions and how the services execute them.

We categorise our users based on their permission and the data we collect from them. To access the main functionalities of the platform, users must complete the registration process and provide the required information to acquire method engineer permissions. However, we ensure that experiment participants remain anonymous, so we limit the amount of data collected from them. These users do not provide any additional personal data and only have access to the execution of the experiment.

5.5.1 Method Engineer Interaction with GNEP

The method engineer user interactions with the application services are depicted in Figure 5.16. The method engineer initiates a new session by logging into the application, providing the registered email and password, and finishing it by logging out from the system. This type of user can configure an experiment and view the experiment results they create.

The main actions that the method engineer can do are the following:

1. Login into the application.
2. Add a new experiment.
3. Set up an experiment.
 - a) Generate experiment task.
 - b) Activate experiment task.
 - c) Generate parameter for task execution.
 - d) Configure parameter.
4. Complete experiment after collection of participant data.
5. Generate output based on submitted data.
6. Download generated results.

To generate an output of the tasks, an experiment has to be executed by its participants before the event, and the experiment status has to be changed to "In Progress".

5.5.2 Experiment Participant Interaction with GNEP

Participant users are granted access to the application only after receiving an access link. However, they can only access certain functionalities, such as Participant View Interface. The application ensures that users remain anonymous and does not collect personal information. However, the application tracks user activities using a technical ID.

List of the activities that participant users can perform is depicted in Figure 5.17:

1. Start a new session by logging into the application.
2. Start experiment.
3. Start experiment task.
 - a) Start Introduction Task.
 - b) Start Term Association Task.
 - c) Start Notation Association Task.
 - d) Start Case Study Task.
 - e) Start Feedback Task.
4. Finish the experiment by closing the session.

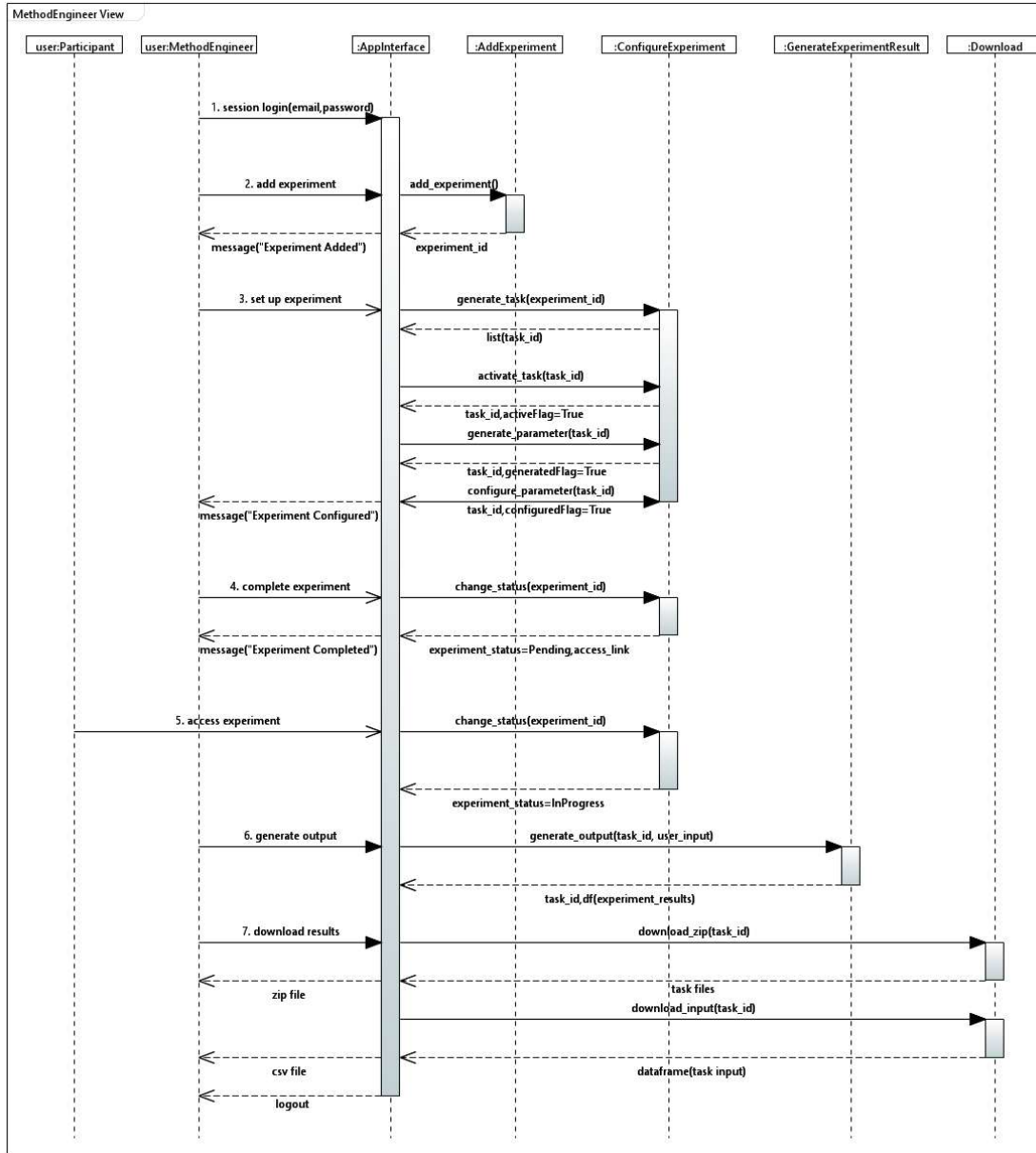


Figure 5.16: Method Engineer View

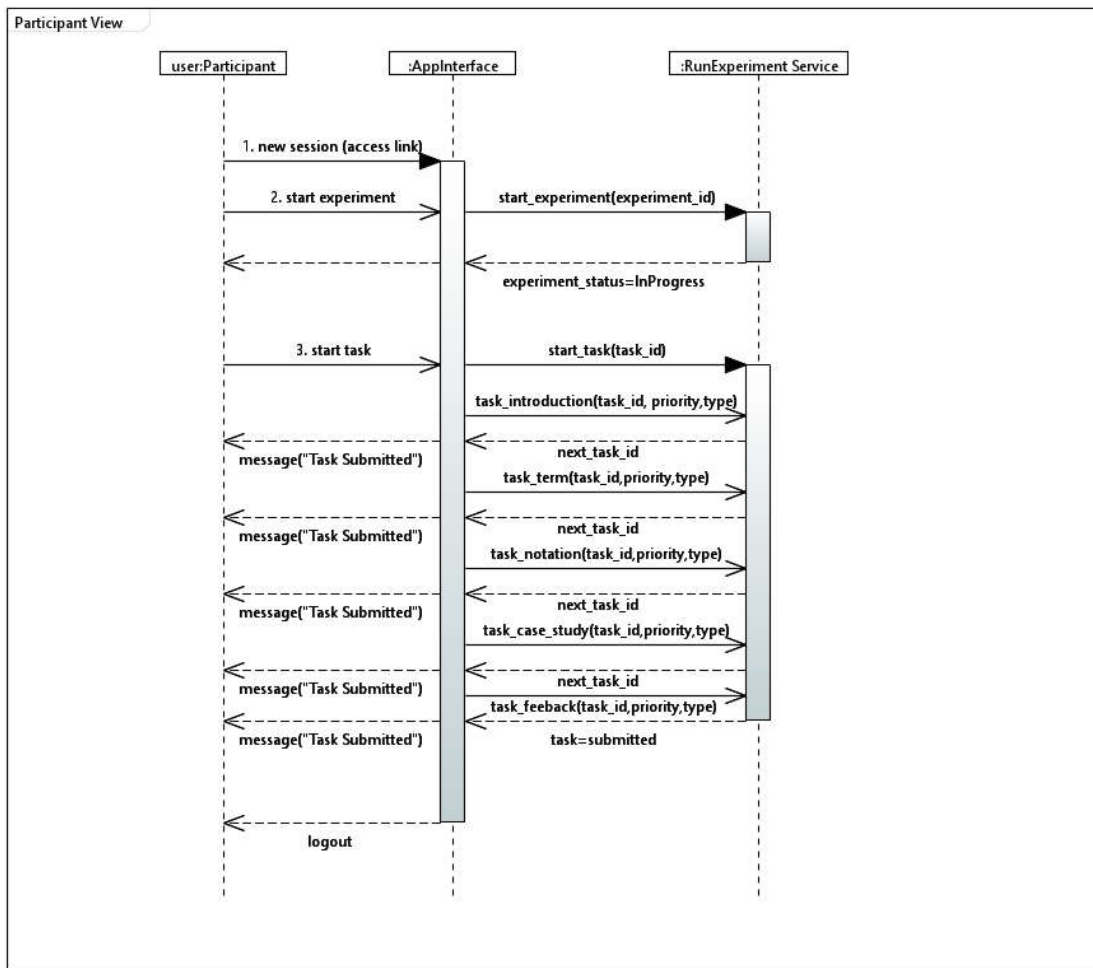


Figure 5.17: Experiment Participant View

CHAPTER 6

Evaluation

We conducted a two-step evaluation of the created web-based platform for graphical notation assessment. Initially, we tested the application using the unit and functional testing to evaluate the product artefact. Subsequently, we evaluated the process artefact, which involved adapting the technique proposed in [2] through an illustrative scenario using the Process-Goal Alignment (PGA) modelling language.

6.1 Illustrative Scenario as Design Science Research Method

6.1.1 General View on Illustrative Scenario

Illustrative scenarios are a powerful tool in the Design Science Research (DSR) framework, particularly during the evaluation phase [19]. They help determine a created artefact's effectiveness by showcasing its practicality and relevance in addressing specific challenges or requirements.

In DSR, where the focus is on developing artefacts, illustrative scenarios are crucial for researchers to demonstrate their creations' real-world implications and effectiveness. Unlike prototypes, which primarily aim to showcase the functionality of artefacts, illustrative scenarios go a step further by placing these artefacts in practical contexts. This approach offers a holistic view of the artefact's operation in scenarios that mimic or represent actual usage.

One unique feature of illustrative scenarios is their adaptability to different contexts. Researchers can tailor these scenarios to represent ideal conditions, allowing for a focused illustration of the artefact's capabilities. This approach highlights specific features and functionalities that might be critical under certain conditions. It is important to note

that, unlike case studies that may involve less-than-ideal observed facts, illustrative scenarios provide a more controlled environment for showcasing the artefact's utility.

By deploying illustrative scenarios in DSR evaluation, stakeholders, practitioners, and researchers can gain a nuanced understanding of how an artefact performs when applied to practical problems. It bridges the gap between theoretical development and real-world application, offering a clear vision of the artefact's potential impact and effectiveness in addressing specific challenges.

6.1.2 Application of Illustrative Scenario

The article [2] describes applying the extended notation evaluation and improvement technique, previously referred to in the Section 4.1 as a pen-and-paper setup (paper-based technique), to evaluate the PGA modelling language notations. The evaluation included an initial notation evaluation and an evaluation of the improved notations.

The described user study was conducted to assess the effectiveness and usability of the initial PGA Notation. It involved 139 Master's level students enrolled in an IT Management class, randomly assigned to two groups. The first group contains 70 participants, and the second one contains 69.

PGA is a language for modelling business architecture. It uses the ADOxx tool and aligns with the Open Models Laboratory. The primary goal of PGA is to help businesses achieve strategic fitness in their architecture. It involves a three-step process: developing a prioritised business architecture hierarchy, executing performance measurement, and performing strategic fit improvement analysis.

The data collected during the first phase of the extended notation evaluation and improvement technique will be used to evaluate the web-based version of the extended notation evaluation and improvement technique. This will allow for a comparison between the two versions. We will use real-world data and the illustrative scenario to evaluate Design Science Research.

6.2 Results Evaluation

The primary focus of this section is to describe the results of applying the first phase of the extended notation evaluation technique to the PGA modelling language notations. It includes five tasks: Initiation Task, Term Association Task, Notation Association Task, Case Study Task and Feedback Task, mentioned in Chapter 4.1.

Based on the results of the Initiation Task, we have confirmed the findings from the paper [2] that a significant number of students had prior modelling experience in fields such as Enterprise Architecture, Business Process Management, and Requirements Engineering. Figure 6.3 a shows the distribution of users with experience in Enterprise Architecture based on their biological sex. The users' experience in Enterprise Architecture includes

knowledge of the Archimate modelling language. Of all the participants, 34% reported having experience in Enterprise Architecture, and 65.7% did not have any experience.

Based on the data presented in Figure 6.3 b, it can be concluded that 90% of the participants have experience in Business Process Management, including BPMN, EPC, UML activity diagrams, and Petri-nets. 14% of the students were familiar with Requirements Engineering, specifically Entity-Relationship (ER) models, as shown in Figure 6.3 c.

Participants were required to familiarise themselves with the PGA modelling language. The 59% of participants, 139 in total, were male and had an average age of 22 years old, as depicted in Figure 6.1.

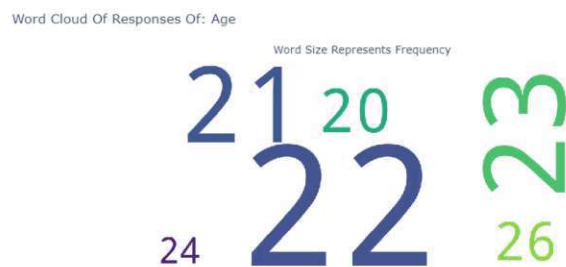
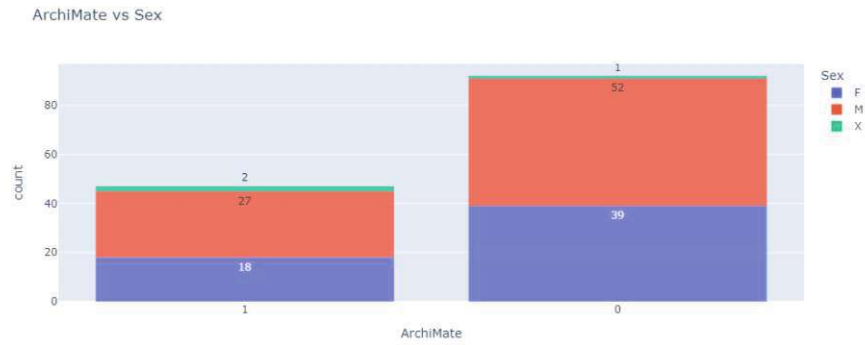


Figure 6.1: Age of Participants

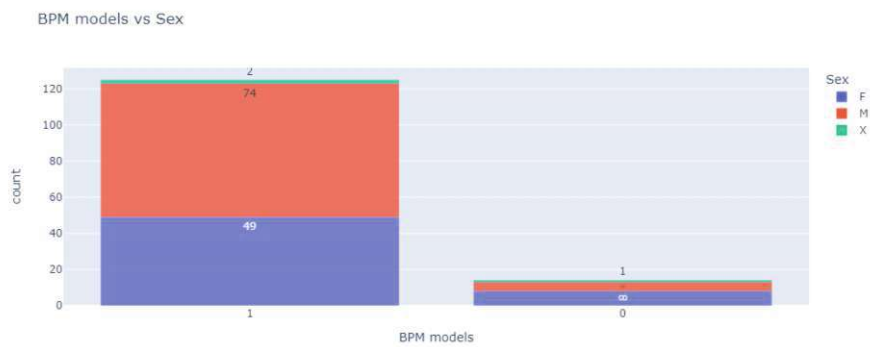
The PGA meta-model consists of several elements denoted by graphical notations, shown in Figure 6.2, for easy understanding by business-oriented users, such as *Activity*, *Process*, *Competence*, *Value Proposition*, *Financial Structure*, *Internal Goal*, *Customer Goal*, and *Financial Goal*. The business architecture heat map uses *Value Stream* relationships to display the hierarchical value structure. A colour-coding system based on strategic *Importance* is applied to each element to differentiate between them. Then, a *Performance* measurement mechanism is used to identify an appropriate performance indicator, set a target and analyse the actual outcomes for each element [2].

Activity 	Process 	Competence 	Value Proposition 	Financial Structure 	Internal Goal
Customer Goal 	Financial Goal 	valueStream 	Performance 	Importance 	

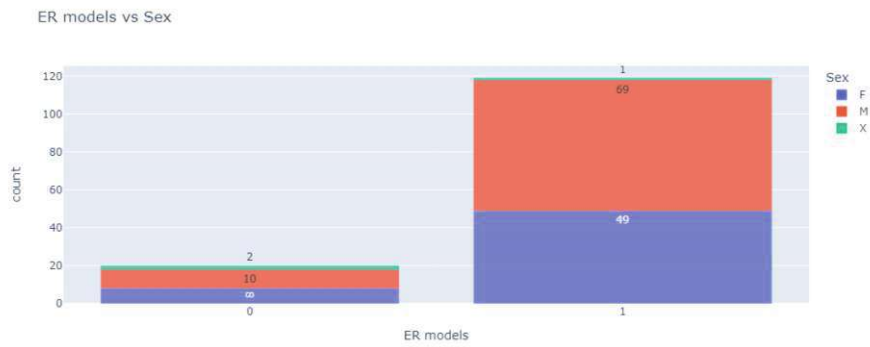
Figure 6.2: Initial PGA notations [2]



(a) Experience in Enterprise Architecture



(b) Experience in Business Process Management



(c) Experience in Requirements Engineering

Figure 6.3: Results of the Initiation Task - Experience in Modelling, Age

6.2.1 Results of the Term Association Task

Participants sketch their version of the graphical notation to the given meta-model concept in the Term Association Task. We utilised the existing drawings from the

previous application of the Notation evaluation and improvement technique, described in the article [2], where the participants used paper and coloured pencils. To showcase the GNEP's functionality fully, we recreated the sketches using the built-in drawing tool to simulate the intended platform's usage.

The built-in tool "Paint", depicted in Figure 5.9 provides a predefined variety of colours and several possibilities to draw notations using:

- free line
- straight line
- predefined main shapes (filled-in and not-filled-in)
- text (keyboard input).

The sketches were saved in PNG and JSON format; an example is demonstrated in the Listing 1. Additionally, we utilised contour recognition to analyse hand-drawn sketches of the initial notations. Doing so could provide a comprehensive comparison between these two approaches: the contour recognition technique involves identifying the edges of the shapes, while the JSON input provides detailed information on the drawing elements. By analysing the results of both approaches, we receive a fuller overview of the image analysis.

To understand the underlying patterns and most frequently used elements, we broke down the graphical notations into individual elements such as shape, colour, and text variables.

The Figure, 6.4, provides a comprehensive overview of the total number of shapes identified from the drawings using different approaches. The depictions of PGA concepts, such as *Activity* and *Process*, primarily utilise rectangular shapes. In contrast, *Internal Goal*, *Customer Goal* and *Financial Goal* are mainly represented by circles.

This distinction may result from the participant's familiarity with Business Process Management and Requirements Engineering. Concepts such as *Process*, *Value Proposition*, *Financial Structure*, and *Competence* are represented mainly by lines. It can be explained by the assumption that lines represent connection links between other shapes or are parts of the figures, like arrows.

It is worth noting that some concepts can be depicted with multiple shapes and elements within a single drawing. Therefore, it is crucial to identify the dominant element of each sketch to recognise potential patterns accurately. By doing so, we can better understand and analyse the figures in the drawings more effectively. We chose the ones with the most significant area when identifying shapes using both techniques.

In the shape recognition technique, the area is calculated within the function, and the JSON format output provides detailed information on the shapes and elements used in

6. EVALUATION

Term	No.sketches	Rectangle	Circle	Square	Line	Ellipse	Purple(indigo)	Black	Blue	Red	Yellow	Orange	Green	Pink
Activity	107	83	13		14	2	143	67	55	4	2	3	0	7
Competence	70	28	21		50	4	243	77	8	3	0	0	2	8
Customer Goal	70	22	61		43	5	256	59	16	2	1	0	8	0
Financial Goal	70	30	46		32	3	223	88	21	0	9	0	3	0
Financial Structure	70	34	31		54	0	256	100	35	22	0	1	13	1
Importance	70	41	10		17	0	110	70	75	19	0	0	0	5
Internal Goal	70	27	80		49	5	230	51	40	2	5	0	14	1
Performance	70	24	22		45	0	193	55	44	0	4	2	9	3
Process	70	110	25		103	1	318	138	1	1	0	0	0	1
Value Proposition	70	33	15		63	2	291	58	1	0	0	0	4	8
Value Stream	70	38	17		40	1	318	108	14	2	5	2	4	6

Figure 6.4: Terms Associations Task: JSON output

an image. We use this information to calculate the area of each element, identify the dominant element, and compare it with the results of shape recognition.

Term	Dominant Shape	Total	Count	Count %
Importance	Rectangle	70	68	97.14
Performance	Rectangle	70	68	97.14
Activity	Rectangle	107	102	95.33
Internal Goal	Rectangle	70	65	92.86
Competence	Rectangle	70	64	91.43
Value Proposition	Rectangle	70	64	91.43
Value Stream	Rectangle	70	61	87.14
Customer Goal	Rectangle	70	59	84.29
	Square	70	7	10
Financial Goal	Rectangle	70	59	84.29
Financial Structure	Rectangle	70	58	82.86
Process	Rectangle	70	50	71.43
	Obround	70	16	22.86
Value Stream	Square	70	7	10

Table 6.1: Results of Contour Recognition in Term Association Task

The comparison between the two approaches reveals insights into using graphical notation associations across various terms. We displayed the percentage of the identified shape that exceeds 10%. Table 6.1 and 6.2 provide a detailed breakdown of the shapes detected in graphical notation sketches associated with various terms. It presents the results in

five key columns: the specific term being analysed, the dominant shape identified in the sketches, the total number of instances analysed, the count of sketches for which the dominant shape was identified, and the percentage that these occurrences represent out of the total. The total number of sketches analysed for each term remains consistent, providing a stable basis for comparison. The "Dominant Shape" column identifies each term's most frequently occurring shape, revealing patterns in how different concepts are visually represented.

The result of the contour recognition algorithm shows that rectangles are prominently featured for such terms as *Important*, *Performance*, *Activity*, *Internal Goal*, *Competence* and *Value Proposition* accounting for above 90% of the shapes detected. The algorithm identified square and obround as additional shapes for terms such as *Square* and *Process*; however, these shapes were predominant in only 10% of total sketches. We notice a clear trend towards identifying rectangles as the dominant shape across most terms. It suggests a standardised approach to representing concepts, where rectangles are commonly used to denote various elements. However, it also means that the algorithm had trouble detecting complex depictions of notation, which resulted in its simplification to a rectangle shape. Notably, *Customer Goal* is a mix of rectangles and squares, possibly indicating a nuanced interpretation of this term.

The analysis in Table 6.2 highlights the diverse graphical notations associated with various terms. Rectangles are the most common shape, representing 64.49% of the term *Activity* and 54.29% of the term *Process*. Other terms display a variety of dominant shapes, including circles, paths, and lines. For instance, *Internal Goal* is predominantly represented by circles, accounting for 44.29%, which might imply a focus on connections or looking at the big picture in this situation. Similarly, *Financial Structure* exhibits a mix of rectangles, circles, paths, and lines, indicating the complex nature of this term and that it is difficult to represent using only basic shapes. Rectangles appear 37.14% of the time, followed by circles at 24.29%, paths at 18.57%, and lines at 14.29%.

Furthermore, the occurrence of multiple dominant shapes for some terms in the second table highlights the flexibility and creativity in graphical notation depiction. For example, *Customer Goal* is depicted using circles 42.86% of the time and rectangles 15.71%, reflecting the possibility of varied interpretations or perspectives regarding this concept. Similarly, *Value Proposition* is associated with rectangles (28.57%), paths (25.71%), circles (20%), and lines (15.71%), suggesting a versatile understanding of this concept.

These results indicate that while rectangles dominate specific terms, the presence of multiple shapes for other terms reflects the nuanced and varied ways concepts are visually interpreted and represented in graphical notations.

Table 6.3 highlights the most commonly used colour. Interestingly, the colour indigo appears to be the most widely used colour across multiple terms, including *Process*, *Value Proposition*, *Competence*, *Activity*, *Value Stream*, *Customer Goal*, *Financial Goal*, *Financial Structure*, *Performance* and *Internal Goal*. This consistent usage of indigo suggests a standard or preferred colour by participants during the drawing process of

Term	Dominant Shape	Total	Count	Count %
Value Stream	Path	70	47	67.14
Activity	Rectangle	107	69	64.49
Process	Rectangle	70	38	54.29
Importance	Path	70	35	50
Internal Goal	Circle	70	31	44.29
Competence	Path	70	30	42.86
Customer Goal	Circle	70	30	42.86
	Rectangle	70	11	15.71
Financial Structure	Rectangle	70	26	37.14
	Circle	70	17	24.29
	Path	70	13	18.57
	Line	70	10	14.29
Performance	Path	70	24	34.29
	Rectangle	70	19	27.14
	Circle	70	14	20
	Line	70	9	12.86
Financial Goal	Circle	70	22	31.43
	Rectangle	70	21	30
	Path	70	15	21.43
Value Proposition	Rectangle	70	20	28.57
	Path	70	18	25.71
	Circle	70	14	20
	Line	70	11	15.71

Table 6.2: Results of Shapes Detection in Term Association Task

graphical notation sketches. Additionally, while some variability in colour usage, like black and blue, was observed, it was to a lesser extent than indigo.

Overall, the table underscores the fact that colour was not seen as an essential visual cue among participants in conveying the meaning of graphical notation across different terms.

Nevertheless, participants used text and graphics to represent the meaning of an element. They used text to complement the proposed graphics visually and verbally. In most cases, the text was either the first letter of the PGA concept or the content of a PGA concept. Among the terms examined, *Financial structure* stands out, with the most common symbol being '\$', which represents financial aspects. Symbols such as 'c', '!', and 'i' are used to signify *Customer Goal*, *Importance* and *Internal Goal*, respectively. Furthermore, multiple representations for specific terms, such as *Value Proposition*, are denoted by 'v' and 'vp', and the term *Activity* appears with variations in both text ('activity') and letter ('a').

Term	Most Common Color	Total	Count	Count %
Process	Indigo	70	54	77.14
	Black	70	14	20
Value Proposition	Indigo	70	52	74.29
	Black	70	16	22.86
	Blue	70	13	18.57
Competence	Indigo	70	50	71.43
	Black	70	15	21.43
Activity	Indigo	107	75	70.09
	Black	107	22	20.56
Value Stream	Indigo	70	49	70
	Black	70	13	18.57
Customer Goal	Indigo	70	46	65.71
	Black	70	16	22.86
Financial Goal	Indigo	70	45	64.29
	Black	70	15	21.43
	Blue	70	8	11.43
Financial Structure	Indigo	70	44	62.86
	Black	70	17	24.29
Performance	Indigo	70	44	62.86
	Black	70	12	17.14
	Blue	70	8	11.43
Internal Goal	Indigo	70	43	61.43
	Black	70	18	25.71
Importance	Indigo	70	37	52.86
	Black	70	13	18.57
	Blue	70	13	18.57

Table 6.3: Results of Color Detection in Term Association Task

6.2.2 Comparison of Techniques for the Term Association Task

The authors of the paper analysed the graphical representations of participants in the article [2]. They examined different visual variables such as colour, shape, icons, and text. Interestingly, despite being instructed to use various colours, participants mainly used blue, which was attributed to the flexibility of the evaluation technique that allowed multiple people to assess the representations using pen and paper. The commonly recurred shapes were rectangles, triangles, ellipses, and arrows, which reflected the participants' experience in Requirements Engineering and Business Process Management. In addition, icons such as thinking balloons and bull's-eyes were often added to basic shapes to enhance semantic encoding. This use of icons aligns with the principle of Dual Coding [4], complemented by textual elements representing either the first letter or the complete name of the PGA concept or its content. Furthermore, participants used spatial enclosure and graphical positioning to represent specific PGA concepts, such as activities within

processes and value streams in the business architecture, often depicted using hierarchical arrangements of arrows.

To compare the results of two technique applications in the Term Association Task, we took the paper-based technique results from the article [2] where visual variables have a cumulative frequency of at least 50 %. We applied the same principle to the values of the web-based technique, with the only difference of putting values below 50 % for variables text and icon for better comparison. The comparison in Table 6.4 between the values obtained from the paper-based and web-based techniques for evaluating graphical notations reveals some interesting insights.

The colour variable in the PGA concepts shows similar resulting values in both technique applications. However, there is a slight difference: a blue colour is used in the paper-based technique application, while indigo (purple) is used in the web-based technique application. This difference can be attributed to the colour distortion that occurred during the digitization and reconstruction of the image.

During our analysis of the shape variable, we observed a noticeable variation in the results obtained from both techniques. We found out that the paper-based technique was more accurate in identifying the shapes used in initial notation sketches. It is important to note that a predefined shape such as a rectangle was dominant in the results of such PGA concepts: *Activity*, *Process*, *Competence*, *Value Proposition*, *Financial Structure* and *Performance*.

In the PGA concept *Activity*, the shape rectangle appears to be dominant in both technique applications with a difference of 10%. The second most frequently occurring element is a path (18%), which does not depict the shape but allows us to infer that some participants drew freehand elements.

When analysing the *Process* results, they show that the shape "rectangle" is more reoccurring in the web-based than in the paper-based technique. This disparity might be attributed to the fact that participants primarily used a combination of shapes like rectangles and arrows to sketch the notation. The algorithms did not recognise these complex structures using the web-based technique, resulting in the rectangle being the dominant shape. In the context of PGA concepts, the *Competence*, *Value Proposition* and *Importance* path is identified as one of the predominant elements alongside the rectangle using a web-based technique algorithm. This differs from the paper-based results, which showed triangles and ellipses. Participants used free lines to represent shapes resembling ellipses and triangles.

The results indicate that in terms of *Financial Structure*, *Internal Goal*, *Customer Goal*, *Financial Goal*, the primary shape observed in the paper-based technique is an ellipse. In contrast, the web-based technique results show a circle. The difference may be attributed to the interpretation of the original sketches during reconstruction. However, the shape's circular nature and occurrence are consistent across both techniques.

When discussing *Value Stream*, we noticed that the dominant shape in the results of the

paper-based technique is an arrow, whereas, in the results of the web-based technique, the path is dominant. The web-based technique algorithm cannot recognise arrows as they are not part of the shapes it can identify, so it is assumed that they were drawn using the freehand line tool. *Performance* results combine the explanation from above; the rectangle is consistent in both techniques' results, and the web-based technique algorithm could not capture the ellipse shape since it was drawn chiefly by hand.

Furthermore, the absence of specific icons in the web-based technique, such as in the *Competence* and *Customer Goal* concepts, suggests that the digital platform may have limitations in representing certain graphical elements effectively since the Paint tool contains only keyboard input which is limited and web-based technique was not designed to capture complex aspects. The results of the paper-based technique include a variety of icons identified by experiment conductors, compared to the results of the web-based technique, where we observe only text and limited icons like a dollar sign and exclamation point.

The differences in the data emphasise the importance of the method engineer's role in carefully analysing and interpreting the results. The web-based platform's results are designed to assist the method engineer in their work and streamline manual processes.

6.2.3 Results of the Notation Association Task

In the Notation Association Task, the participants write up to three associations for each graphical notation that the method engineer provides. Figure 6.5 provides an overview of the terms used for the notation association task. Notably, the value-stream relation, represented by a non-directed line, was not explicitly tested. The meaning of this relation only becomes apparent when included in a hierarchical business architecture heat map [2].

We used Similarity score, calculated between the notation legend and the associations provided by experiment participants. Based on the similarity score values, associations were classified into four clusters: fully identified, identified, partially identified and not identified, mentioned in Section 5.3.

The application was designed to enable the method engineer to define the desired association with the notation, Positive Term, and the undesired association, Negative Term. In our case, no Positive Terms exist except for "colour", which has associated terms. It happened because no method engineer was involved in applying the technique.

We have adapted the definitions of TP, FP, FN, and TN for this particular application of the technique since no Positive Terms were specified. The default assumption is that all terms are considered positive unless stated otherwise.

Each graphical notation has associated Negative Terms. Negative Terms include all meta-model elements not related to the notation. This is done to avoid confusion with meta-model concepts of the modelling method.

Adjusted definitions:

6. EVALUATION

PGA concept	Variable	Paper-based Technique Value	Web-based Technique Value
Activity	Shape Color Icons Text	Rounded rectangle (75%) Blue (71%) Person (18%) Activity (46%), A (23%)	Rectangle (65%) Indigo (70%) - Activity (4%), A (4%)
Process	Shape Color Text	Rectangle (28%), Arrow (28%) Blue (75%) Act. nr. (71%)	Rectangle (54%) Indigo (77%) Act. nr. (7%)
Competence	Shape Color Icons Text	Rectangle (31%), Triangle (23%) Blue (76%) Thinking balloon (21%), Person (14%), Light bulb (14%) C (57%)	Path (43%), Rectangle (20%) Indigo (71%) - C (6%)
Value Proposition	Shape Color Icons Text	Ellipse (27%), Rectangle (24%) Blue (78%) Dollar/Euro (29%), + sign (10%) V (27%), VP (27%)	Rectangle (29%), Path (20%) Indigo (74%) - V (4%), VP (4%)
Financial Structure	Shape Color Icons Text	Ellipse (35%), Rectangle (33%) Blue (63%) Dollar/Euro (80%) Cost & revenues (40%), C & R (20%)	Rectangle (37%), Circle (24%) Indigo (63%) Dollar (26%) Cost & revenue (7%), C & R (3%)
Internal Goal	Shape Color Icons Text	Ellipse (54%) Blue (67%) Bull's-eye/arrow (64%) I (29%), x (21%)	Circle (44%), Rectangle (26%) Indigo (62%) - (9%)
Customer Goal	Shape Color Icons Text	Ellipse (34%), Cloud (16%) Blue (65%) Bull's-eye/arrow (48%), Person (33%) C (44%), Customer (goal) (22%)	Circle (43%), Path (19%) Indigo (66%) - C (13%), Customer (goal) (3%)
Financial Goal	Shape Color Icons	Ellipse (30%) Blue (65%) Dollar/Euro (67%)	Circle (31%), Rectangle (30%) Indigo (64%) Dollar (2%)
Value Stream	Shape Color Icons Text	Arrow (69%) Blue (73%) Dollar/Euro (47%), Stream (18%) V (67%)	Path (67%) Indigo (70%) Dollar (3%) V (2%)
Performance	Shape Color Icons	Rectangle (32%), Ellipse (21%) Blue (72%) Graph (18%), V checkbox (18%), Muscle (13%)	Path (34%), Rectangle (27%) Indigo (63%) -
Importance	Shape Color Icons	Rectangle (26%), Triangle (26%) Blue (66%) Exclamation mark (75%)	Path (50%) Indigo (53%) Exclamation mark (15%)

Table 6.4: Comparison of Term Association Task Results

- TP - A term classified as "fully identified" does not match "negative terms".
- FP - A term that has a match with a negative association and is classified as "fully identified."
- FN - A term that is classified as "not identified," "partially identified," and "identified" and does not match "negative terms".
- TN - A term that is classified into "not identified," "partially identified," and "identified" and matches with the negative association.

We calculate the Precision score for each participant input based on the occurrence of Positive and Negative Term associations in the participant input. This score is based on the definition mentioned in Section 5.3.

Term	Notation	Positive Terms	Negative Terms	Precision Score	Recall Score	F1 Score
financial structure			activity; internal goal; importance; customer goal; value stream; financial goal; value proposition; competence; process; performance	1.0	0.42	0.59
process			activity; financial structure; internal goal; importance; customer goal; value stream; financial goal; value proposition; competence; process; performance	1.0	0.32	0.48
financial goal			activity; financial structure; internal goal; importance; customer goal; value stream; value proposition; competence; process; performance	0.77	0.21	0.33
competence			activity; financial structure; internal goal; importance; customer goal; value stream; financial goal; value proposition; process; performance	1.0	0.19	0.32
activity			financial structure; internal goal; importance; customer goal; value stream; financial goal; value proposition; competence; process; performance	1.0	0.17	0.29
internal goal			activity; financial structure; importance; customer goal; value stream; financial goal; value proposition; competence; process; performance	1.0	0.11	0.2
customer goal			activity; financial structure; internal goal; importance; value stream; financial goal; value proposition; competence; process; performance	0.75	0.03	0.06
value proposition			activity; financial structure; internal goal; importance; customer goal; value stream; financial goal; competence; process; performance	0.67	0.03	0.06
color		importance, performance	activity; financial structure; internal goal; customer goal; value stream; financial goal; value proposition; competence; process	0.0	0.0	0.0
value stream			activity; financial structure; internal goal; importance; customer goal; financial goal; value proposition; competence; process; performance	0.0	0.0	0.0

Figure 6.5: Terms from Notation Association Task

Precision score is a performance metric that evaluates the accuracy of a system in identifying the "correct" associations. It considers both True Positives (correct identifications) and False Positives (incorrect identifications). A higher Precision score indicates a more accurate and reliable identification in the classification process. To conclude, the Precision score measures the proportion of correctly identified terms in the "fully identified" category out of total terms in the "fully identified" category.

In addition to the Precision score, our evaluation incorporates two vital metrics — Recall score and F1 score.

Recall score, often called sensitivity or true positive rate, assesses the application's ability to capture all Positive Terms initially defined by the method engineer. It is calculated as the ratio of True Positives to the sum of True Positives and False Negatives as defined in Section 5.3. A high Recall score indicates that the system effectively identifies most of the positive associations, minimizing the chances of false negatives. It measures the completeness of the positive term identifications.

We employ the F1 score, which balances precision and recall. F1 score is the harmonic mean of precision and recall, calculated using the formula mentioned in Section 5.3. This metric is particularly valuable as it considers false positives and negatives. A higher F1 score signifies a well-rounded performance in precision and recall, providing a comprehensive evaluation of the classification process.

Together, Precision, Recall, and F1 score offer a holistic assessment of our classification method's accuracy, completeness, and balance in identifying positive and negative terms in the participant term associations.

Table 6.6 provides an overview of performance metrics for various terms used in graphical notation associations. The terms are evaluated based on their Precision score, Recall score, and F1 score, which comprehensively assess the classification system's effectiveness.

PGA Concept	Precision score	Recall score	F1 score
Financial Structure	1	0.42	0.59
Process	1	0.32	0.48
Financial Goal	0.77	0.21	0.33
Competence	1	0.19	0.32
Activity	1	0.17	0.29
Internal Goal	1	0.11	0.2
Customer Goal	0.75	0.03	0.06
Value Proposition	0.67	0.03	0.06

Table 6.5: Measures of Graphical Notation Associations

Some notation like *Financial Structure*, *Process*, *Competence*, *Activity*, and *Internal Goal* have high Precision scores of 1, indicating that they are correctly identified graphical notations with consistent accuracy. It's worth noting that there is a significant difference in their Recall scores. For instance, *Internal Goal* has a lower score of 0.11, suggesting that while experiment participants may have correctly associated the notation with terms, the system did not accurately identify them and labelled them as "not identified", "identified", or "partially identified". Additionally, slightly lower F1 scores indicate a clear trade-off with recall, which leaves room for improvement in capturing all instances of these terms.

On the other hand, some terms, like *Financial Goal*, *Customer Goal* and *Value Proposition* show lower Precision scores (0.77, 0.75 and 0.67 respectively). It signifies a higher rate of false positives in their identification, meaning that a higher number of graphical notations were wrongly associated by the participant and contained Negative Terms.

For instance, *Financial Goal* graphical notation was wrongly associated with *Financial Structure* and *Performance*, which are meant to identify other graphical notations. Table 6.5 shows a more balanced situation with a Precision score of 0.77 and a Recall score of 0.21, resulting in an F1 score of 0.33. This means there is a moderate ability to

identify positive instances with a reasonable level of accuracy, as reflected in the trade-off between precision and recall.

On the other hand, *Customer Goal* and *Value Proposition* have extremely low Recall scores (0.03). It indicates that the system did not accurately identify the majority of the positive associations. This could be due to the lack of specification of positive terms and the assumption that any association that is not negative is positive. Consequently, their F1 scores are only 0.06, highlighting challenges in correctly identifying and comprehensively capturing these terms.

Value Stream and *Color* graphical notations were not identified correctly by participants; it results in zero values in all measures, indicating a complete inability to capture positive instances for these terms.

As we mentioned above, lower Recall scores point out that some graphical notations were associated with the wrong meta-model concepts, as highlighted in red in Figure 6.6. Of 69 participants, 36 (equivalent to 52.17%) confused the Competence notation with the Performance notation. Similarly, 36 out of 102 participants (35.29%) associated the Internal Goal icon with Process. These results suggest a weak perceptual differentiation between these notations in the initial PGA notation.

Term	User Associated Terms	Occurance	Similarity score	Cluster Categories	Negative Term Occurance	Positive Term Occurance
competence	performance	36	0.6969799357	identified	36	0
internal goal	process	36	0.5020319715	partially identified	36	0
financial goal	financial structure	16	0.6218119975	identified	16	0
process	activity	13	0.5981112446	identified	13	0
financial structure	financial goal	10	0.6218119975	identified	10	0
internal goal	competence	10	0.6127737045	identified	10	0
value proposition	importance	10	0.7094018889	identified	10	0
customer goal	value proposition	9	0.5303479871	identified	9	0
activity	process	8	0.5981112446	identified	8	0
value stream	activity	8	0.5590629934	identified	8	0

Figure 6.6: Negative Terms Associations

The results underscore the importance of considering a combination of Precision, Recall, and F1 score for a nuanced evaluation.

6.2.4 Comparison of Techniques for the Notation Association Task

Table 6.6 displays correct notation associations defined and described in paper [2], indicating accurate identification of respective terms. The percentage of correct associations for PGA graphical notations ranges from 0% to 36.23%. *Process* and *Activity* were ranked the highest, 36.23% and 24.29%, respectively. *Financial Goal* ranked second with 20.29%, and *Financial Structure* ranked third with 12.75%. The notations of other PGA elements such as *Internal Goal*, *Competence*, and *Value Proposition* could be more precise, as the percentages are below 5%. Notably, the non-directed line representing the *Value Stream* relation was not explicitly tested. The meaning of this relation only becomes apparent when it is included in a hierarchical business architecture heat map [2].

PGA Concept	Correct Association	Correct Association, %
Process	25	36.23
Activity	17	24.29
Financial Goal	14	20.29
Financial Structure	13	12.75
Internal Goal	5	4.9
Competence	2	2.9
Value Proposition	3	2.83
Customer Goal	0	0
Performance	0	0
Importance	0	0

Table 6.6: Notation Associations with similarity score equals one

For better comparison of the techniques in the Notation Association Task, we applied the same metrics used in the paper-based technique [2] to the results in the web-based technique. These metrics were the percentage of participants giving correct associations and the relative rank of this association [2].

It is essential to note that the correct association in the paper-based technique is the participants' input that matches the graphical notation definition (term). The article by [16] defines it as the hit rate metric, which represents the percentage of correct answers per graphical notation. Since it is an identical match, the similarity score between participant input and the PGA concept equals one.

In the web-based version of the technique, we expanded the definition of the correct association to include all associations that belong to the "fully identified" cluster based on the high similarity score values (from 0.76 to 1). We also excluded associations from this selection that matched the other PGA notations ("negative terms"). The correct association rate increased for all PGA concepts except for *Performance* and *Importance*, where none of the associations was flagged as "fully identified". Concepts such as *Financial Structure* and *Competence* show differences of more than 10%, 51%, and 17%, respectively.

In Table 6.7, we presented the latest metrics for the correct association in the "fully identified cluster" of participant inputs. The "association list" column contains values from Table 6.6 highlighted with bold font. Additionally, the associations are listed in descending order of occurrence, and occurrence values are enclosed in brackets. Only one participant has given this input if a number is not provided. The colour demonstrates the association's relationship to the PGA concept of graphical notation. Our primary focus is the semantic connection between the concepts, not the relation between graphical notation and association. We use green to highlight associations that accurately convey the meaning and could be added to the correct association list and red for ones that only partially cover the concept meaning or can be misleading.

PGA Concept	Association List	Correct Association	Correct Association %
Financial Structure	accounting (21), financial structure (13), financial (10), finance (5), financial statement (4), financial balance (3), asset management , current balance , financial activity , financial limitation , financial objectives , financial performance , financial results , financials , profitability	65	63.73
Process	process (25), continuous process , process flow	27	39.13
Financial Goal	financial goal (14), goal (4), budget goal , saving goal	20	28.99
Activity	activity (17), activities	18	25.71
Competence	achievement (11), competence (2), objective	14	20.29
Internal Goal	internal goal (5), operational goal (2), complementary goal , goal , strategic goal , system goal	11	10.78
Customer Goal	goal (3)	3	4.29
Value Proposition	value proposition (3), value	4	3.77
Performance	-	0	0
Importance	-	0	0

Table 6.7: Notation Associations that belong to "fully identified" cluster

Let us take a look at the *Financial Structure* association list. Most participant inputs relate to the financial structure concept or fall under the finance umbrella, such as

"accounting" or "financial statement". Some terms have broad meanings, like "finance" or "financial"; however, they cannot be used interchangeably. We include the term "financials" because it provides insights into a company's financial structure, such as revenues, expenses, assets, liabilities, and equity. The association list of *Competence* includes "achievement" and "objective", which do not convey the meaning of the term. From the association list of the concepts *Financial Goal*, *Internal Goal*, *Customer Goal* and *Value Proposition*, we excluded associations that refer to the different concepts or are just partially match the concept.

On the other hand, concepts like *Process* and *Activity* were enriched with additional associations which do not contradict the meaning of the term, like "continuous process", "process flow", and "activities".

Table 6.9 shows the correct association rate based on the modified association list; we can observe that the Correct Association rate for most graphical notations assessed via the web-based platform has remained mostly the same. The *Financial Structure*, *Process*, and *Activity* notations improved a rate of correct associations up to 3%.

PGA Concept	Correct Association	Correct Association %
Process	27	39.13
Activity	18	25.71
Financial Goal	14	20.29
Financial Structure	14	13.72
Internal Goal	5	4.9
Competence	2	2.9
Value Proposition	3	2.83
Customer Goal	0	0
Performance	0	0
Importance	0	0

Table 6.8: Modified Correct Notation Association list

According to ISO 9186:2001 [16], the PGA graphical notations hit metrics values reached 67% should be self-explanatory. Since the values are lower than the threshold mentioned, we will use relative rank to compare the semantic transparency of notations.

In Table 6.9, we can see the ranks of the paper-based technique's initial phase, provided in paper [2] and the web-based technique. The paper-based technique used the rank based on the percentage of correct association (hit rate), and the web-based technique ranked notations based on the formula of R_{combined} . The R_{combined} is based on the percentage of correct associations per graphical notation and the accuracy and relevance of each participant's input, which our system assesses.

$$R_{\text{combined}} = w_1 \cdot R_{\text{percentage}} + w_2 \cdot R_{\text{F1score}}$$

R_percentage - a rank of Correct Association (hit rate); R_F1score - a rank based on F1 score values; w_1, w_2 - weights with values 0.5.

Initial PGA Concept	Paper-based Technique Relative Rank	Web-based Technique Relative Rank	R_combined
Process	1	1	1
Activity	1	2	3
Financial Goal	2	3	2
Financial Structure	3	4	2
Internal Goal	5	5	5
Competence	8	6	4
Value Proposition	9	7	6
Customer Goal	-	-	-
Performance	-	-	-
Importance	-	-	-
Value Stream	-	-	-

Table 6.9: Comparison of Correct Notation Associations in Notation Association Task

When comparing the web-based and paper-based techniques, we observe that the rank order from the first to the last position remains the same. It indicates no significant change in the values of the correct association rate. The *Process* notation maintains its top rank, followed by the *Activity* sequential notation, with the *Value Proposition* notation at the last position. Notably, the PGA notations *Customer Goal*, *Performance*, *Importance*, and *Value Stream* did not have any correct associations identified, so we have omitted their ranking.

It is essential to note that we enhanced the relative ranking by incorporating the F1 score ranking, which demonstrates the performance of the web-based technique algorithm. The *Process* notation ranks first in both approaches. The *Activity* concept moves to the third place in the overall ranking compared to the paper-based technique. However, this can be explained by the performance of the *Financial Structure* notation, which rises to the second position from the third in the web-based technique. Along with the *Financial Goal*, they have higher F1 score values, indicating a greater ability to identify positive instances compared to other notations accurately.

These fluctuations impacted the rank of *Competence* notation, which maintains its low rank in the paper-based technique but climbs up to fourth in R_combined due to the high Precision score and moderately low Recall score in the web-based technique.

Since the system could not identify more correct associations for the notions of *Internal Goal* and *Value Proposition* in the web-based technique, their performance rank did not change significantly compared to the paper-based technique. Their lower Recall score prevented them from being placed higher in the ranking.

Customer Goal, *Performance* and *Importance* have no rank in the paper-based and web-based techniques.

6.3 Research Questions (RQ)

6.3.1 Response to RQ 1

Research Question 1: What are the appropriate means to support the semantic transparency evaluation method by a web-based environment?

The semantic transparency evaluation in the notation evaluation and improvement technique is based on two key metrics: term and notation association. In the term association metric, participants are asked to visually represent the concepts associated with a notation. These drawings are then classified by similar colour, shape, icons and text. This metric helps to measure participants' ability to interpret and convey the intended meanings of the notation's elements.

Similarly, participants provide up to three associations for each notation in the notation association metric. These associations are categorised into groups of fully identified, identified, partially identified, and not identified.

Both metrics comprehensively support the visual notation's semantic transparency evaluation, indicating the symbols' effectiveness in conveying their intended meanings. These evaluations are helpful for method engineers to refine visual notations, ensuring clarity and user-friendly interpretations for effective communication.

In paper [2], the authors evaluated the notation evaluation and improvement technique against the list of requirements; we use the exact requirements for assessing the web-based notation evaluation technique. The paper-based technique fulfilled four of the seven completely and three partially 6.10. The web-based technique fulfils six of them and one partially.

Requirement	Paper-based Technique	Web-based Technique
Efficient customization	Fulfilled	Fulfilled
Notation improvement	Fulfilled	Partially Fulfilled
Efficient use	Partially Fulfilled	Fulfilled
Involve participant suggestion	Fulfilled	Fulfilled
Semantic transparency	Fulfilled	Fulfilled
Technical independence	Partially Fulfilled	Fulfilled
Modular structure	Partially Fulfilled	Fulfilled

Table 6.10: Fulfillment of requirements in Paper-based and Web-based Techniques

Efficient customization requirements are fully satisfied in both techniques allowing customization to specific DSMLs. The web-based notation evaluation technique can be applied as a highly customised experiment, allowing the engineer to select a list of tasks, configure them, and specify parameters per each task.

The paper-based technique meets the requirement for **notation improvement** by offering updated notations for PGA and BCM modelling languages. The web-based

technique primarily focuses on the first phase of notation improvement, while the second phase is not covered. However, suggestions for improvement are provided in the feedback task, and the results of the term and notation association tasks offer opportunities for further notation enhancement. As a result, this requirement is partially fulfilled.

Efficient use of paper-based techniques is possible when conducting small-group experiments. However, when scaling out, additional effort is required. This limitation can be addressed through a web-based notation evaluation technique that provides an automated process for experiment setup, data collection, and result analysis.

Involve participant suggestions. Both approaches allow participants to provide feedback on initial notations. The web-based technique incorporated these improvements into the system by allowing experiment participants to provide feedback on each concept that was part of the evaluation, meeting the requirement.

Semantic transparency. The main focus of these techniques is to evaluate the semantic transparency of the graphical notation at the model and concept levels.

Technical independence. The platform that implements the web-based technique allows users to interact with it without installing additional software. This requirement is fully fulfilled, in contrast to the paper-based technique, which partially satisfies this requirement due to the potential need for technology support when scaling out.

The web-based technique fulfils the requirement for a **modular structure**. The experiments are implemented to allow users to reuse configured tasks in a defined sequence or on their own. Since the web-based technique focuses on the first phase of the paper-based technique, it satisfies the requirement statement.

We used the SWOT analysis to comprehensively assess the graphical notation evaluation platform where the web-based evaluation technique is implemented and highlight its strengths, weaknesses, opportunities, and threats. Conducting a SWOT analysis is essential for method engineers and stakeholders to understand the technique's current position in the research field and identify areas for improvement and emerging opportunities.

- **Strengths**

- *Semantic Transparency Evaluation:* The technique effectively evaluates semantic transparency at both the model and concept levels, providing valuable insights into the clarity and effectiveness of graphical notations.
- *Streamlined Experiment Setup:* An automated setup process for the technique ensures ease of usage, allowing efficient assessments to enhance semantic transparency in modelling language notations.
- *Automated Process:* The process setup streamlines the experiment's initiation, data collection, and result analysis, enhancing efficiency and reducing manual effort.

- *Automated Metric Generation*: Automating the generation of result metrics makes the evaluation process more efficient and less prone to human error, aiding in systematically assessing semantic transparency across various aspects of the notations. Metrics can be used in further evaluation by the method engineer.
 - *Synergistic Analysis*: Integrating machine learning-driven analysis techniques with user-driven assessments enhances the depth and accuracy of the system evaluation. This synergy combines the strengths of automated algorithms with method engineer expertise, leading to more comprehensive insights into semantic transparency.
 - *Flexibility and Possible Customization*: The web-based technique allows for highly customised experiments. Engineers can tailor tasks and parameters to specific Domain-Specific Modelling Languages (DSMLs), enhancing flexibility and relevance.
 - *User Involvement*: Users and community members can participate in the evaluation of graphical notation by providing feedback on initial notations and facilitating continuous improvement and refinement based on user suggestions and preferences.
 - *Technical Independence*: The platform’s accessibility without additional software installation ensures technical independence and ease of use for participants, promoting broader participation and engagement.
- **Weaknesses**
 - *Limited Notation Improvement Coverage*: While the technique evaluates semantic transparency, it may offer limited coverage in the second phase of notation improvement, potentially missing opportunities for comprehensive enhancement and graphical notation revision.
 - *Challenges in Classification of Complex Signs*: Results of shape and icon recognition algorithms are not comparable with manual classification results since detecting hand-drawn icons and complex and nested shapes is limited.
 - *Metrics Reliability*: The reliability of conventional hit rate metrics in ST evaluation is questioned, calling for further validation and exploration of the STC metric.
 - **Opportunities**
 - *Integration of Machine Learning*: Integrating machine learning-driven analysis techniques with user-driven assessments can enhance the depth and accuracy of evaluation results, providing more comprehensive insights and facilitating continuous improvement.
 - *Enhanced visualisation Tools*: Implementing enhanced visualisation tools to analyse and interpret evaluation results can improve method engineer

comprehension and guide further refinement and investigation, enhancing the technique's effectiveness.

- *Advanced Shape and Color Recognition Technique*: Introduction of the contour recognition technique to support nested shape recognition and automated colour identification will further enhance the framework.
- *Collaboration within the Community*: The technique's web-based nature enables broader accessibility and reach, potentially attracting a wider audience of method engineers and stakeholders interested in improving semantic transparency in graphical notations and reinforcement of method and model engineers' collaboration.

- **Threats**

- *Technological Advances*: Rapid technological advancements may render current features or methodologies obsolete, requiring continuous innovation and adaptation to remain relevant and competitive in notation evaluation methodology.
- *Data Privacy and Security Concerns*: Data privacy and security concerns may arise, especially with collecting and storing sensitive user data, requiring robust measures to ensure compliance and trustworthiness.

Overall, the web-based notation evaluation technique demonstrates strengths in efficient customization, semantic transparency evaluation, and automated processes. However, it proposes limited notation improvement coverage. Opportunities for integrating machine learning, enhanced visualisation tools, and expanded community reach present further development and improvement avenues.

6.3.2 Research Question 2

Research Question 2: To what extent can image processing techniques support the evaluation of semantic transparency in term association task responses?

Participants were asked to deliver a drawing of the graphical notation associated with a specific meta-model concept in the Term Association Task. We used the term association metric to support the semantic transparency evaluation of the provided graphical notations. Variables such as colour, shape, and text are extracted from the provided graphical notation sketches and used to classify the drawings and find recurring patterns. This metric helps to measure participants' ability to interpret and convey the intended meanings of the notation's elements.

Approaches, used in the Term Association Task for semantic transparency evaluation:

- Machine Learning-Driven Analysis:
 - Using shape and colour classification from JSON output.

- Usage of the contour recognition of shapes.
- User-Driven Input:
 - Identifying patterns based on image clustering configured by a user.

The Graphic Notation Evaluation Platform integrates sophisticated canvas manipulation functionalities initiated through the JavaScript functions. Users can select various drawing modes such as Freeline, Straightline, Text, Rectangle, Square, Ellipse, and Circle, enabling them to create and manipulate shapes easily. Additionally, users can customise the stroke colour, fill colour, and stroke width of the drawn objects, providing flexibility in graphical representation. The system also supports essential canvas operations such as clearing the canvas, deleting objects, copying and pasting objects, cutting objects, grouping and ungrouping objects, undoing and redoing actions, and saving and loading the canvas state. These versatile functionalities enable users to create and modify graphical notations according to their requirements and preferences. The functionality allows for the decomposition of graphical notations into variables stored in JSON format, which is analysed in detail, enhancing the evaluation process of semantic transparency in term association tasks. The analysis includes identifying the most recurrent or dominant element on the sketch provided by participants, helping the method engineer conclude if the graphical notations are semantically transparent to the experiment participants.

Our system incorporates contour recognition technology, allowing users to identify image shapes. This feature can be customised and configured according to users' requirements and preferences. The system's contour recognition algorithm recognises basic shapes such as circles, stars, heptagons, hexagons, lines, squares, and rectangles, which is enough to provide insights into the structural composition and spatial organisation of graphical elements within the analysed images.

User-driven analysis capabilities, in combination with machine learning-driven analysis, provide a comprehensive and synergistic approach to data interpretation and decision-making, facilitating more profound insights into the underlying patterns and relationships within the evaluated data.

Integrating image clustering functionality into the graphical notation evaluation process offers significant advantages in understanding and analysing visual representations. Leveraging pre-trained ImageNet models for image classification enhances the accuracy and efficiency of the clustering process, as these models have been trained on vast amounts of annotated images to recognise various objects and patterns. Utilising ImageNet aligns with our objective of improving the evaluation of graphical notations by providing a robust framework for object recognition, image classification, and clustering, ultimately facilitating a deeper understanding of visual representations.

This combination of machine learning and user-driven analysis empowers method engineers to extract meaningful insights and make informed decisions about the notation improvement and how participants perceive the provided graphical notations; this allows them to make assumptions about the semantic transparency of the graphical notations.

6.3.3 Research Question 3

Research Question 3: To what extent can natural language processing techniques support the evaluation of semantic transparency in notation association task responses?

In the Notation Association Task, participants deliver text associations to the graphical notation of the meta-model concept, up to three terms per concept. After collecting the responses, we used the notation association metric to evaluate the semantic transparency of the graphical notations. These associations are classified into fully identified, identified, partially identified, and not identified groups.

Approaches used in the Notation Association Task for semantic transparency evaluation:

- Machine Learning-Driven Analysis:
 - Utilizing Natural Language Processing (NLP) techniques.
 - K-means clustering of textual input based on similarity scores.
- User-Driven Input:
 - Using hit rate metric and ranking to evaluate semantic transparency of the graphical notation.
 - The F1 score assess system clustering based on user requirements.

We used NLP techniques to compare meta-model concepts provided by method engineers with participant associations, calculating similarity scores between term and participant input.

To achieve this, we rely on `en_core_web_lg`, a pre-trained machine-learning model specifically designed for English language processing by the SpaCy library. This model, known as "large", has a strong capacity for various NLP tasks, such as named entity recognition, text classification, and part-of-speech tagging. Trained on a large dataset of web-based text containing over 2 million words, it serves as a reliable tool in our effort to understand and evaluate linguistic nuances effectively.

As a result, we expanded the definition of the "identified" notation. We also employ machine learning algorithms, such as K-means clustering, to categorise user associations into four clusters: "fully identified", "identified", "partially identified", and "not identified". This process uncovers patterns and trends within the responses, enabling the assessment of semantic transparency.

To evaluate the semantic transparency of the graphical notation, we employ the hit rate and notation ranking as an evaluation metric. Firstly, we accurately quantify the proportion of users who correctly identify the notations. Secondly, we rank graphical notations based on the correct association rate. This metric offers insights into the overall performance of participants in the notation association task and their comprehension of the

notation's semantic elements, meaning their estimation of how semantically transparent the graphical notation is.

Additionally, it was essential to demonstrate the effectiveness of the system's classification based on the similarity score. We engaged the method engineer in defining the criteria for "desired" and "not desired" experiment participant associations. It was achieved by establishing evaluation metrics such as the F1 score per each graphical notation; the F1 score integrates precision and recall scores to identify positive and negative associations between user responses and the notation legends. With this value's help, we fine-tuned the ranking of the notations to include the method engineer's input.

By combining NLP techniques with user-driven input and evaluation metrics, we aim to leverage computational methods and human judgment to comprehensively assess the semantic transparency of notation association task responses.

Conclusions

The goal of this master's thesis was to contribute to the evaluation of the graphical notations in domain-specific modelling languages. By reviewing existing graphical notation evaluation approaches and their assessments, we identified potential gaps in the notation evaluation, such as high manual effort in the empirical evaluation and no involvement of the target audience and method engineer. We aimed to address these gaps by expanding the Notation Evaluation and Improvement Technique designed by Bork and Roelens [2], primarily by automating the first phase of the technique.

By following the Design Science Research methodology, we defined the artefact, methods, criteria and research questions for its evaluation. In the article [2], the authors explored ideas for technique improvement and automatisation. We used them as initial requirements for the technique improvement and designed a graphical notation evaluation platform. We split our artefacts into process and product artefacts to thoroughly evaluate them, where evaluation of each confirms the feasibility and effectiveness of the other. With the help of the Feature Driven Development method, we constructed the prototype of the web-based platform that provides the method engineer with the functionality to create customised experiments for graphical notation evaluation, run them and analyse the provided assessment results.

Additionally, we used an illustrative scenario to evaluate a web-based technique and compare it to the paper-and-pen technique described in [2]. The results of the applications of the two techniques were compared for Term Association and Notations Association Tasks.

After the web-based technique and the prototype evaluation, we can make the following conclusions:

- The Graphic Notation Evaluation Platform provides model engineers with the automated technique set-up. It supports experiment conducting and its result

7. CONCLUSIONS

generation, saving a lot of manual effort for method engineers and participants compared to the paper-and-pen technique.

- The experiments are highly customised, and experiment tasks can be reused and configured separately; this creates high flexibility for the method engineer directly involved in the evaluation process.
- The generated results of the technique are meant to be auxiliary and aid users of the web-based platform in analysing and evaluating semantic transparency of the graphical notations.
- The goal of the web-based platform is to contribute to collaboration between the method engineer and model engineer and foster collaboration within the community via feedback features.
- The web-based technique doesn't include a notation improvement phase; it could be a direction for future work.
- The current algorithm version can be enhanced to improve image and text analysis and better support semantic transparency evaluation.

List of Figures

2.1	Perceptual and cognitive processing [4]	7
2.2	Principles for designing cognitively effective visual notations [4]	7
2.3	Semantic Transparency is a Continuum [15]	9
2.4	Procedure of applying the notation evaluation and improvement technique [2]	10
5.1	Specific Django Project Architecture	22
5.2	Class Diagram Of GNEP	24
5.3	Application Layer of GNEP	25
5.4	Add Experiment Service	27
5.5	Task Generation Interface	28
5.6	Task Activation Interface	29
5.7	Parameter Configuration Interface	30
5.8	Participant Questionnaire of Introduction Task	32
5.9	Paint Interface	33
5.10	Participant Summary in Feedback Task	33
5.11	Initiation Phase Task Result Interface	35
5.12	Term Association Task Result Interface	37
5.13	Notation Association Task Result Interface	40
5.14	Feedback Task Result Interface	40
5.15	Download Service	41
5.16	Method Engineer View	43
5.17	Experiment Participant View	44
6.1	Age of Participants	47
6.2	Initial PGA notations [2]	47
6.3	Results of the Initiation Task - Experience in Modelling, Age	48
6.4	Terms Associations Task: JSON output	50
6.5	Terms from Notation Association Task	57
6.6	Negative Terms Associations	59



Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar
The approved original version of this thesis is available in print at TU Wien Bibliothek.

List of Tables

5.1	Task attributes after execution of <code>generate_task()</code> function	28
5.2	Task attributes after execution of <code>activate_task()</code> function	29
5.3	Task Parameters after execution of function <code>generate_parameter()</code> .	30
5.4	Task attributes after execution of <code>generate_parameter()</code> and <code>configure_parameter()</code> functions	31
6.1	Results of Contour Recognition in Term Association Task	50
6.2	Results of Shapes Detection in Term Association Task	52
6.3	Results of Color Detection in Term Association Task	53
6.4	Comparison of Term Association Task Results	56
6.5	Measures of Graphical Notation Associations	58
6.6	Notation Associations with similarity score equals one	60
6.7	Notation Associations that belong to "fully identified" cluster	61
6.8	Modified Correct Notation Association list	62
6.9	Comparison of Correct Notation Associations in Notation Association Task	63
6.10	Fulfillment of requirements in Paper-based and Web-based Techniques . .	64



Die approbierte gedruckte Originalversion dieser Diplomarbeit ist an der TU Wien Bibliothek verfügbar
The approved original version of this thesis is available in print at TU Wien Bibliothek.

Bibliography

- [1] Karagiannis, D., Kühn, H.: “Metamodelling Platforms”. In: *Bauknecht, K.; Min Tjoa, A.; Quirchmayer, G. (Eds.): Proceedings of the Third International Conference EC-Web 2002 – Dexa 2002*, Aix-en-Provence, France, September 2-6, 2002, LNCS 2455, Springer-Verlag, Berlin, Heidelberg, p. 182.
- [2] Bork, D., Roelens, B.: “A technique for evaluating and improving the semantic transparency of modeling language notations”. In *Software and Systems Modeling*, 20, 939–963 (2021).
- [3] Kouhen, A.E., Gherbi, A., Dumoulin, C., Khendek, F.: “On the Semantic Transparency of Visual Notations: Experiments with UML”. In *Model-Driven Engineering for Smart Cities*, pp. 122, Springer, 2015.
- [4] Moody, D.: “The ‘physics’ of notations: toward a scientific basis for constructing visual notations in software engineering”. In *IEEE Transactions on Software Engineering*, 35(6), 75-779 (2009).
- [5] Karsa, G., Krahn, H., Pinkernell, C., Rumpe, B., Schindler, M., Volkel, S.: “Design guidelines for domain specific languages”. In *Proceedings of the 9th OOPSLA. Helsinki School of Economics*, TR no B-108. Orlando, Florida, USA, October 2009
- [6] Bork, D., Schruffer, C., Karagiannis, D.: “Intuitive Understanding of Domain-specific Modeling Languages: Proposition and Application of an Evaluation Technique”. In *International Conference on Conceptual Modeling*, pp.311-319. Springer (2019)
- [7] Bork, D., Karagiannis, D., Pittl, B.: “A survey of modeling language specification techniques” In *Inf. Syst.*, 87 (2022).
- [8] Ge, M., Helfert, M.: “A Design Science Oriented Framework for Experimental Research in Information Quality”. In *15th International Conference on Informatics and Semiotics in Organizations (ICISO)*, May 2014, Shanghai, China. pp.145-154.
- [9] Hevner, A.R., March, S.T., Park, J., Ram, S.: “Design science in information systems research”. In *MIS Q.* 28(1), 75-105 (2004).

- [10] Genon, N., Caire, P., Toussaint, H., Heymans, P., Moody, D.: “Towards a more semantically transparent i* visual syntax”. In *Int. Conference on Requirements Engineering: Foundation for Software Quality*, pp. 140-146. Springer (2012).
- [11] El Kouhen, A., Gherbi, A., Dumoulin, C., Khendek, F.: “On the semantic transparency of visual notations: experiments with UML”. In *International SDL Forum*, pp.122-137. Springer (2015).
- [12] Roelens, B., Steenacker, W., Poels, G.: “Strategic alignment: realizing strategic fit within the business architecture: the design of a process-goal alignment modeling and analysis technique”. In *Softw. Syst. Model.* 18(1), 631–662 (2019).
- [13] Bork, D., Karagiannis, D., Pittl, B.: “Systematic analysis and evaluation of visual conceptual modeling language notations”. In *12th International Conference on Research Challenges in Information Science, RCIS 2018*, Nantes, France, May 29-31, 2018.
- [14] Sharma, S., Sarkar, D., Gupta, D.: “Agile Processes and Methodologies: A Conceptual Study”. In *International Journal on Computer Science and Engineering, IJCSE 2012*, Vol.4.
- [15] Caire P., Genon N., Heymans P. and Moody D. L.: “Visual notation design 2.0: Towards user comprehensible requirements engineering notations”. In *2013 21st IEEE International Requirements Engineering Conference (RE)*, Rio de Janeiro, Brazil, 2013, pp. 115-124, doi: 10.1109/RE.2013.6636711.
- [16] Kuhar, S., Polancic, G.: “Conceptualization, measurement, and application of semantic transparency in visual notations: A systematic literature review”. In *Software and Systems Modeling (SoSyM)*, Volume 20, Issue 6D, 2021, pp 2155–2197.
- [17] Santos, M., Gralha, C., Goulao, M., Araujo, J., Moreira, A.: “On the Impact of Semantic Transparency on Understanding and Reviewing Social Goal Models”. In *2018 IEEE 26th International Requirements Engineering Conference (RE)*, pp. 228–239. IEEE (2018).
- [18] van der Linden D, Zamansky A, Hadar I.: “How Cognitively Effective is a Visual Notation? On the Inherent Difficulty of Operationalizing the Physics of Notations”. In *Schmidt R, Guédria W, Bider I, Guerreiro S, editors, Enterprise, Business-Process and Information Systems Modeling*. Springer, Cham. 2016. p. 448-462. (Lecture Notes in Business Information Processing).
- [19] Peffers, K., Rothenberger, M., Tuunanen, T., Vaezi, R. (2012). “Design Science Research Evaluation”. In *Peffer, K., Rothenberger, M., Kuechler, B. (eds) Design Science Research in Information Systems. Advances in Theory and Practice*. DESRIST 2012. Lecture Notes in Computer Science, vol 7286. Springer, Berlin, Heidelberg.

- [20] Cleven, A., Gubler, P., Hüner, K.: “Design alternatives for the evaluation of design science research artifacts”. In *Proceedings of the 4th International Conference on Design Science Research in Information Systems and Technology*, DESRIST 2009, Philadelphia, Pennsylvania, USA, May 7-8, 2009.
- [21] Bioco, J., Rocha, A.: “Web Application for Management of Scientific Conferences”. In *WorldCIST'19*, 2019, AISC 930, pp. 765–774, 2019.
- [22] Bork, D., Buchmann, R.A., Karagiannis, D., Lee, M., Miron, E.T.: “An Open Platform for Modeling Method Conceptualization: The OMiLAB Digital Ecosystem”. In *Communications of the Association for Information Systems*, 44, pp. 673-697 (2019).
- [23] Frank, U: “Domain-Specific Modeling Languages - Requirements Analysis and Design Guidelines”. In *Reinhartz-Verger, I.: Sturm, A., Clark, T., Bettin, J., Cohen, S.: Domain Engineering: Product Lines, Conceptual Models, and Languages*. Springer 2013, pp. 133-157.
- [24] “How to Detect Shapes in Images in Python using OpenCV?”. URL: <https://www.geeksforgeeks.org/how-to-detect-shapes-in-images-in-python-using-opencv/>
- [25] Yuen, P. C., Feng, G. C., Zhou, J. P.: “A contour detection method: Initialization and contour model”. In *Pattern Recognition Letters*, 20(2), 141-148.
- [26] Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., Fei-Fei, L.: “Imagenet: A large-scale hierarchical image database”. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pp. 248–255, 2009. IEEE.
- [27] Mijač, M.: “Evaluation of Design Science instantiation artifacts in Software engineering research”. In *Central European Conference on Information and Intelligent Systems; Varazdin : Faculty of Organization and Informatics Varazdin*. (2019): 313-321.
- [28] Plotly Technologies Inc.: “Collaborative data science Publisher: Plotly Technologies Inc.”. Montréal, QC Date of publication: 2015 URL: [urlhttps://plot.ly](https://plot.ly).
- [29] “The pandas development team.” *pandas-dev/pandas: Pandas*. Feb 2024. Zenodo. Version: v2.2.1. DOI: <https://doi.org/10.5281/zenodo.3509134>.
- [30] Lloyd, Stuart P.: “Least squares quantization in PCM.” *Information Theory, IEEE Transactions on* 28.2 (1982): 129-137.
- [31] Honnibal, M., Montani, I.: “spaCy 2: Natural language understanding with Bloom embeddings, convolutional neural networks and incremental parsing”. 2017.
- [32] Itseez. “Open Source Computer Vision Library”. 2015. Available at: <https://github.com/itseez/opencv>.

- [33] Kos, T.; Mernik, M.; Kosar, T.: “Evolution of Domain-Specific Modeling Language: An Example of an Industrial Case Study on an RT-Sequencer”. In *Applied Sciences*, 2022, 12, 12286.