Facilitating Dynamic Assessment of Psychological Interventions

Roger Wisnes

Master's thesis in Software Engineering at

Department of Computer science, Electrical engineering and Mathematical sciences, Western Norway University of Applied Sciences

Department of Informatics, University of Bergen

July 2021

Western Norway University of Applied Sciences

Abstract

The mental healthcare system requires an improvement in patient accessibility and a decrease in resource dependency. Internet-Delivered Psychological Treatment (IDPT) is a tool with a documented positive effect to bring about this improvement. However, there is still more research needed in this field to reduce patient dropout rates.

One hypothesis for increasing user adherence in IDPT systems is to create a personalized user experience for the individual patient through adaptive strategies. This study examines how to facilitate the implementation of well-documented (rule-based) algorithms for such adaptiveness.

Through the work, the research methodology Design Science was used to design and develop a digital artifact for an open-source, early-stage prototype of an IDPT framework. The intention of the artifact was to facilitate the implementation of rule-based adaptive algorithms, by simplifying the process of creating and adapting Cognitive Behavioral Therapy (CBT)-based treatment programs.

The result of the study is an artifact that provides a graphical representation of the structure and content of CBT-based treatment programs. The representation is a hierarchical layout with three levels of interactive network graphs where nodes represent the elements of the CBT program (interventions, modules, and tasks), and the edges represent the relationships between these elements. Along with the graphs, detailed views of the information within the elements are also provided.

The study concludes that the graphical representation of the content of a CBT-based treatment program within IDPT can simplify the previously mentioned process by clarifying how the elements of the CBT program are related to each other, and thus provide the therapist with an easier way to correct any errors and omissions within the elements, and adjust criteria that adaptive algorithms can use.

Acknowledgements

First, I would like to thank my supervisors Yngve Lamo and Svein-Ivar Lillehaug, for the advice I received through working on the artifact of this Master's thesis. Furthermore, I would like to thank Suresh Kumar Mukhiya for the information he provided regarding the framework that was to be expanded by the artifact.

I would also like to thank my academic colleagues, Ingrid Johansen and Oskar Leirvåg, for the countless academic discussions. Finally, thanks to my family for all their support, particularly Maren, for the patience and flexibility.

Contents

List of Figures

List of Tables

Acronyms

BtB Beating the Blues.

CBT Cognitive Behavioral Therapy.

COPE Coping with Breast Cancer.

HVL Western Norway University of Applied Sciences.

iCBT Internet-based Cognitive Behavioral Therapy.

IDPT Internet-Delivered Psychological Treatment.

LMS Learning Management System.

SoC Separation of Concerns.

UI User Interface.

UiB University of Bergen.

Chapter 1

Introduction

Through this chapter, the motivation for the thesis will be explained in detail. After that, a description of the problem and research questions will follow, before a short introduction to the methodology and ethical aspects. Finally, the chapter will provide an outline for the thesis.

1.1 Motivation

The combination of mental health and social media is recently the cause of numerous alarming headlines such as *"Anxiety, loneliness and Fear of Missing Out: The impact of social media on young people's mental health"* [12] and *"Six ways social media negatively affects your mental health"* [4]. Over the past decade, there has been a considerable increase in psychological issues, particularly anxiety and depression, among adolescent Norwegian girls [15, 16]. In general, mental disorders are among the leading causes of suicide, and in 2019 the World Health Organization (WHO) reported a yearly suicide rate of near 800.000 people [40], meaning approximately one case every 40 seconds. Additionally, there are far more attempted suicides [40].

According to the Norwegian Institute of Public Health, there are, in Norway alone, about 500 to 600 instances of suicide committed every year, leaving approximately ten times as many survivors [17]. The psychological impacts on family members and others with a close relationship with the lost ones are wide-ranging and include complications such as anxiety, stress, physical health difficulties, and even depression [52], which again is one of the major reasons for attempted suicides in the first place.

The significant rate of global suicides suggests that a better tool is needed for preventive work towards mental health. Today, the Norwegian Directorate of Health asserts that an adult patient in Norway has to wait an average of over 40 days from the point of referral in the specialist health service until the mental health care has started. However, the Norwegian Ministry of Health and Care Services aims for this waiting time to be under 40 days by the end of 2021 [25].

As reported by several mental health media, such as the Centre for Mental

Health and Royal College of Psychiatrists, the waiting time between the point of referral and the point of treatment seems to aggravate the situation for a significant portion of the patients [31, 42]. In a case where the patent suffers from depression, such an aggravation could be severe for the patient's mental health. This suggests that it can be crucial to reduce the waiting time. However, the lack of resources in the mental health service complicates this reduction [57].

To mo make up for the lacking resources, a pursuit of functioning digitization of the mental health services has been going on for several years. Internet-Delivered Psychological Treatment (IDPT) is such a digital tool intended to make mental healthcare more accessible and thereby reducing the need for both economic and non-economic resources. Although the research has come a long way in the past decades [1], there are still more problems to be solved, one of which is dropout rates [50].

IDPT systems are typically based on psychological interventions, which are actions *"aimed at promoting a better adaptation of the individual to a given situation and thereby optimizing his or her personal resources in relation to autonomy, self-knowledge, and self-help"* [47]. One common psychological intervention used in IDPT to treat depression, among other mental disorders, is Internet-based Cognitive Behavioral Therapy (iCBT). A recent study conducted in the United States claims that iCBT programs have had a dropout rate ranging from 0% to as high as 75% [50], implying the need for a system that increases user adherence.

As reported by Mukhiya et al. [36], it seems that adaptive elements in IDPT may be able to enhance the results from internet interventions and thereby reduce patient dropout rates. The study also reports deficient research on strategies for design and adaptive elements in IDPT systems and therefore encourages further research on adaptive IDPT systems [36].

1.2 Problem Description

Suresh Kumar Mukhiya, a Ph.D. scholarship holder at the Western Norway University of Applied Sciences (HVL), is currently working on a framework for creating adaptive IDPT systems. With a number of contributors, including *INTROMAT (INTROducing Mental health through Adaptive Technology)*, the framework is an early-stage prototype and open-source project with the vision of becoming an IDPT system enhancing user interaction and increasing user adherence. The framework is intended to be used for creating any healthcare interventions, and the overall ambition is to modularize the project as an aggregation of plugins.

Some widely used terms within IDPT are CBT and iCBT, which are treatment programs for diagnoses such as depression and ADHD, among others. These terms will be described in more detail in Chapter 2, but for the sake of the problem description, a brief explanation is that a program like this is one intervention, made up of a set of modules, where each of which contains a set of tasks.

Throughout the project of this Master's thesis, the objective is to expand

Mukhiya's framework by creating a digital artifact facilitating dynamic assessment of psychological interventions. That is, the goal of this thesis is to make it easier to ensure that the modules and tasks of an intervention are presented in an order that is tailored to the individual patient. For the production of the artifact, the main focus will be on frontend development, yet some additions in the backend will occur as well.

At the present time, the open-source project has not yet been given an official name and will therefore be referred to as the IDPT framework or system throughout this thesis.

1.3 Research Questions

The main research question of this thesis is how to implement, design, and evaluate an artifact facilitating dynamic assessment of psychological interventions in IDPT. In a more specific formulation, the research questions of this thesis are as follows:

RQ1 Given that there must be some form of artificial interconnection between the elements of a psychological intervention to enable reliable algorithms for dynamic assessment of these elements, how can it be ensured and clarified that all elements within an intervention are universally interconnected?

RQ2 In cases where the use of adaptive algorithms for the assessment of psychological interventions is facilitated, how can it be ensured that the elements within the interventions are presented and recommended in an appropriate order?

RQ3 Building on the results from the previous two research questions, and given that a therapist will create a new iCBT-based treatment program in the IDPT framework, how can the process of creating and adapting this treatment program be simplified for the therapist?

1.4 Research Methodology

Due to the nature of this Master's thesis, with the production of a digital artifact, it was chosen to use Design Science as the research methodology. This is a methodology that focuses on iterative processes for developmental progress in order to achieve results within an organizational problem through digital artifacts.

Hevner et al. [26] have defined, in their paper, a set of guidelines for the implementation of this methodology, and based on these, it was considered to fit well with the process involved in this thesis. Design Science with guidelines will be described in more detail in Chapter 3.

1.5 Ethical Aspects

Since the IDPT system may store information about individual human beings, one major ethical question to keep in mind is patient confidentiality and disclosure. A strict pursue of the General Data Protection Regulation (GDPR) is therefore of great importance for any further work. However, during the work of this thesis, no real data was handled, and the artifact itself does not consider patient-related information. Hence, it was not applicable to concentrate on GDPR within the given time frame.

During the process, no information concerning real patients was obtained. An ethics approval was therefore not necessary.

1.6 Thesis Outline

The following is a brief overview of the structure of this thesis. The overview addresses the most significant subjects of the chapters:

- **Chapter one** introduces the motivation for this thesis, as well as the research questions and methods, and a description of the objectives.
- **Chapter two** will focus on the background for the study, including a more detailed description of CBT and iCBT. Furthermore, adaptive algorithms will be presented, and finally, some related work.
- **Chapter three** goes into more detail about design science as the chosen research methodology and discusses a set of guidelines for how to conduct research using this methodology, as well as how these guidelines are met. In addition, the artifact's requirements are described, along with the design choices and design process.
- **Chapter four** introduces the relevant technologies, architecture, and design patterns, before describing on a more code-related level how the artifact is implemented.
- **Chapter five** will examine how well the resulting artifact meets the requirements through experimental testing.
- **Chapter six** will provide an answer to each of the research questions, as well as a summary of the thesis' contribution to the domain knowledge. Finally, the chapter will provide a brief reflection regarding methodology and technologies.
- **Chapter seven** provides a summary of the process and results achieved through this thesis and finally provides suggestions for further research and work related to these results.

Chapter 2

Background

This chapter will have a look at the background of the study. The following two sections describe CBT and iCBT, specifically with respect to the treatment of depression. Furthermore, adaptive algorithms will be presented, and finally, some related work, including a systematic review that addresses adaptive elements within IDPT.

2.1 Cognitive Behavioral Therapy

According to the American Psychological Association (APA), Cognitive Behavioral Therapy (CBT) is a psychological treatment method with a documented positive effect for numerous psychological issues, including depression, anxiety disorders, eating disorders, and phobias, among several others. The method has been demonstrated to be at least as effective as other psychological and psychiatric approaches, and there is a considerable amount of scientific evidence that the developed methods produce change [3, 27, 10, 51].

The core principles of CBT involve the ideas that psychological issues are partly based on damaging or discouraging thinking patterns and partly on learned patterns of discouraging behavior, as well as the idea that patients can learn to improve their coping mechanisms, thus easing their situation by somewhat acting as their own therapist [3, 14]. For the patient to achieve the skill-set required to master a working self-help routine, the treatment usually concentrates on techniques for problem-solving, relaxation, and behavioral activation, as well as identifying and handling "cognitive distortion" [45], which is a dysfunctional automatic thought described in more detail in Table 2.1.

The structure and duration of the CBT treatment may vary based on the severity of the patient's condition. Nevertheless, it is a common practice for the therapist to establish a collaborative relationship with the patient in order to develop a treatment strategy depending on the patient's problem. In their guide to brief CBT, Cully and Teten [45] propose the eight-session treatment structure described in Table 2.2, where each session constitutes a subset of a total of 14 modules including both information, activities and exercises, as well as homework for the patient.

Cognitive distortions
All-or-nothing thinking: Assuming the extreme. E.g., ”My wife didn’t tell me good night, yesterday. She doesn’t love me anymore.”
Catastrophizing: Having a pessimistic view on the future. E.g., ”If I don’t get that job, I will never succeed in life.”
Disqualifying/discounting the positive: Thinking that the positive do not count. E.g., ”My friend complimented me on my new shirt, but that is just because he feels obligated to.”
Emotional reasoning: Ignoring the facts, in favor of ones emotions. E.g., ”I worked way harder for the exam than her. If she got an A, so should I.”
Labeling: Labeling someone/something without seeking information. E.g., ”My husband would never do something like that.”
Magnification/minimization: Overestimating the negative/underestimating the positive. E.g., ”My girlfriend paid my bill. It’s probably because she thinks I’m terrible with money.”
Mental filter/tunnel vision: Only concentrating on the negative. E.g., “My wife said that I talk too much. I’m obviously not interesting at all.”
Mind reading: Assuming to know others thoughts. E.g., ”I’m wearing the same outfit as yesterday, so they think I’m dirty.”
Overgeneralization: Exaggerating negative conclusion. E.g., ”My daughter failed a class, this semester. Maybe I’m a bad parent.”
Personalization: Feeling responsible for the negative in others. E.g., ”My teammate said we must report our own work. Was he was referring to me?”
”Should”/”must” statements: Having expectations on people’s behavior. E.g., ”I must always have an immaculate home, or else I am a slob.”

Table 2.1: Cognitive distortions [45, 18].

Session	Content
Session 1	Orient patient to CBT. Assess patient concerns. Set initial treatment plan.
Session 2	Continue assessing patient concerns, and setting initial plan or begin intervention techniques.
Session 3	Begin/continue intervention techniques.
Session 4	Cont. intervention techniques. Re-assess goals/treatment plan.
Session 5	Continue/refine intervention techniques.
Session 6	Continue intervention techniques.
Session 7	Continue intervention techniques. Discuss ending treatment and prepare for maintaining changes.
Session 8	End treatment and help patient to maintain changes.

Table 2.2: Proposed brief CBT treatment structure [45].

Normally, both the patient and the therapist are required to physically attend all sessions through the entirety of the CBT treatment in a face-to-face manner. This requirement poses a challenge in terms of access to mental healthcare, particularly so in remote areas, as the demand for qualified personnel exceeds the actual number of educated therapists [28]. Challenges like this suggest that the implementation of CBT over the internet may be the solution, delivering both cost-effectiveness for all parties and remote access to the required therapy [22].

2.2 Internet-Based Cognitive Behavioral Therapy

The prevalence of Internet-based Cognitive Behavioral Therapy (iCBT) programs has increased over the past decades [58], and similar to the face-to-face practice of traditional CBT, an iCBT treatment may be guided by a therapist. Through a guided treatment, the patient and therapist will communicate asynchronously through an online message system, either embedded in the iCBT program itself or a third-party system such as email, thus providing significant patient improvements [23]. Furthermore, iCBT treatments may be unguided or self-guided, where the entire treatment is automated with little to no communication between the patient and the therapist.

There is a wide range of iCBT programs in existence. One example is the COPE project, which will be presented in Section 2.4. Another example of an iCBT program is Beating the Blues (BtB) [6], providing what they call CCBT, or *"Cognitive Behavioral Therapy delivered by an interactive computer program"*, responding to the patient's individual circumstances by introducing the stages of the therapy analogous to how a therapist would in a traditional

CBT treatment [5]. Their main treatments are for depression and anxiety and span over eight sessions consisting of three to four modules of 10 to 15 minutes each [6].

According to a 2003 study [41], BtB patients showed a significantly better result by the end of treatment compared to those of traditional CBT. The study also concluded that the results are independent of the severity of depression [41]. However, a more recent study [19] contradicts these results, stating that no significant difference was found in symptom improvement. Instead, this study indicated low user adherence among the BtB patients, as only 18% of the patients completed all treatment modules [58, 19].

2.3 Adaptive Algorithms

One hypothesis for what causes this dropout rate is the lack of or insufficient personalization of the treatment program for the individual patient, and there seems to be limited research on how user adherence is affected by the implementation of adaptiveness in IDPT-systems [35]. In order to facilitate further research in this field, there is a need for IDPT-systems with well-documented implementations of adaptive algorithms so that the research has the opportunity to take adaptive strategies into account as a possible factor for user adherence.

Regarding adaptive algorithms, there are specifically four different strategies discussed in connection with the IDPT framework extended through this thesis [36]:

- **Rule-Based Adaptation:** This is a strategy based on rules defined in advance by domain experts (i.e., psychiatrists), and the quality of the rules, as well as their consequences are measured through a wide range of metrics.
- **Adaptation Through Feedback Loops:** A general feedback loop is performed through four steps. First, data about the current state and context of the system is collected, then this data is analyzed in order to be able to perform a selection of actions before these actions are finally executed.
- **Goal-Driven Adaptation:** This is a strategy where the goals and needs of the users are the main factors for adaption.
- **Adaptation Through Predictive Algorithms:** In essence, this is a strategy based on Artificial Intelligence.

In this thesis, the main focus will be on facilitating a rule-based strategy, and in this context, two ideas have been discussed for how this can be accomplished: One idea is to provide a passive foundation, based on the assignment of constraints such as inclusion criteria and exclusion criteria, which adaptive algorithms can use as rules to produce general recommendations for building psychological interventions. The next idea is that user profiling can build on the previous idea by using the constraints in order to be able to produce more personalized recommendations for the individual patient regarding the order in which the elements of the treatment program should be presented.

2.4 Related Work

In 2020, Mukhiya et al. [35] conducted a systematic review targeting *"Adaptive Elements in Internet-Delivered Psychological Treatment Systems"*. This systematic review serves as an overview of relevant sources addressing the topic at hand. Furthermore, some related work was found through the use of Bergen Open Research Archive (BORA) and Google Scholar. Although a high number of related studies were discovered, few of them focused on any relevant implementation.

Through this section, the aforementioned systematic review will be presented, along with two Master's theses subjecting adaptiveness in the mental health project "COPE".

2.4.1 Adaptive Elements in IDPT Systems: Systematic Review

Recent developments in IDPT have heightened the need for adaptive elements to decrease patient dropout rates in computerized mental healthcare. In addition to the objective of creating a conceptual framework, the systematic review conducted by Mukhiya et al. in 2020 [35] aimed to inspect and identify the adaptive elements, information architecture, dimensions of adaptiveness, and implementation strategies for interventions in IDPT, along with examining the efficacy influence of system adaptation. The review concludes that further study is required on all of the aforementioned points, as there is a research deficiency focusing on adaptiveness in IDPT systems.

In their systematic review, Mukhiya et al. [35] discussed multiple potential elements of adaptiveness in IDPT systems, such as information architecture, design, user interface, content presentation, -complexity, and -recommendation, and intervention content, among several others. These adaptive elements are described as *"the main components that are personalized for the user"* [35].

As this thesis will focus on intervention content as the main adaptive element, the systematic review brings relevance through analysis of several sources with the same focus. However, although some of these sources go into detail about an adaptive algorithm [29, 37], the lack of focus on implementation applies to these sources as well.

2.4.2 Coping with Breast Cancer (COPE)

According to Heitmann [24], COPE is a collaboration project between SLATE at UiB and HVL intended on breast cancer survivors experiencing psychological problems related to their cancer treatment or situation. This subsection looks into two Master's theses partaking in said project.

Development of a mobile artifact to support adaptive iCBT using multi modality support and usage data

Heitmann [24] was part of the COPE project team, whose aim was to develop a CBT application with support for patient progress tracking and data collection to enable a personalized user experience for the patient. The main focus for

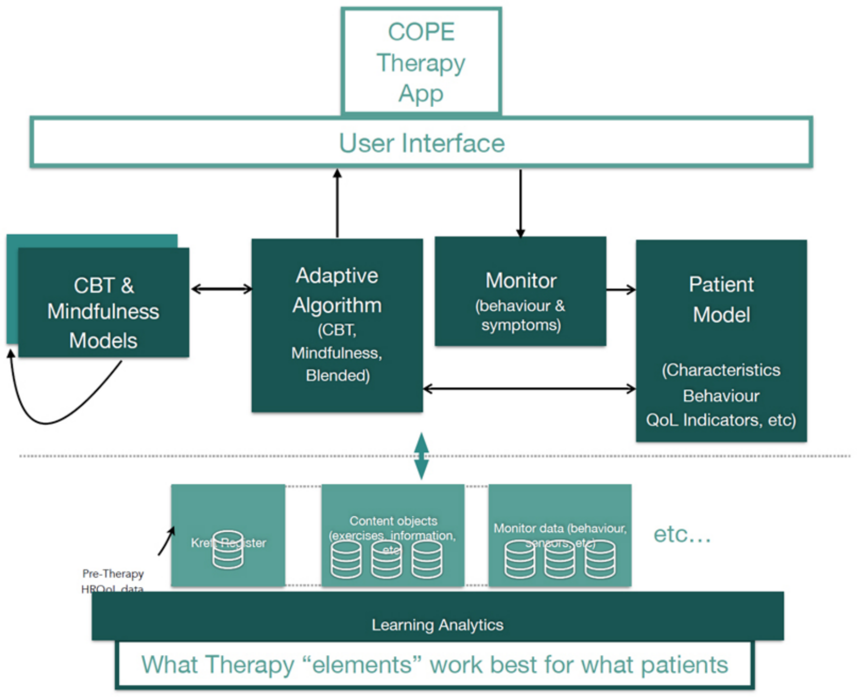

COPE Therapy App
User Interface
CBT & Mindfulness Models
Adaptive Algorithm (CBT, Mindfulness, Blended)
Monitor (behaviour & symptoms)
Patient Model (Characteristics Behaviour QoL Indicators, etc)
Kreft Register
Content objects (exercises, information, etc
Monitor data (behaviour, sensors, etc)
etc…
Pre-Therapy HRQoL data
Learning Analytics
What Therapy “elements” work best for what patients

Figure 2.1: The architecture of the COPE platform [24].

Heitmann's study was facilitating an adaptive iCBT system on a mobile platform through the use of multi-modality. As dynamic assessment relies highly on adaptiveness, said focus makes for a high relevance for this thesis' main objective. Although Heitmann does not go into detail about the adaptiveness itself (as this is the subject of another COPE-related thesis [48]), he does look into the collection of user data as a tool for facilitating patient personalization.

The COPE architecture, as shown in Figure 2.1, is based on Intelligent Tutoring System (ITS) [33], including user profiling, which is done through numerous approaches. The system collects the user's content bookmarking events, log entries for media playback, interactions with learning material, and interactions with activities related to CBT submissions such as questionnaire data. The latter one was deemed particularly important, as questionnaire data was assumed to be useful in future predictions of the user's results. Additionally, Heitmann suggests that monitoring the user's physical activity, along with logging position data and timestamps of the application usage, may prove useful regarding adaptiveness.

Building a flexible CBT model based on structured data for the COPE app

Considering patient personalization for the COPE project, Salimath [48] examined how to structure and organize meta-knowledge of treatment modules and sub-modules and how the treatment within these modules and sub-modules can be fitted with diagnoses, symptoms, or patient goals. The study describes a domain model serving as a basis for structuring CBT modules with their associated content within COPE. The domain model consists of four treatment modules, each divided into five sub-modules with hierarchical taxonomy according to Figure 2.2. The treatment is specifically generated for individual patients to fulfill a certain goal, using an adaptive course sequencing technique [7], where information from a measured patient model is interpreted as signs or symptoms of a disorder.

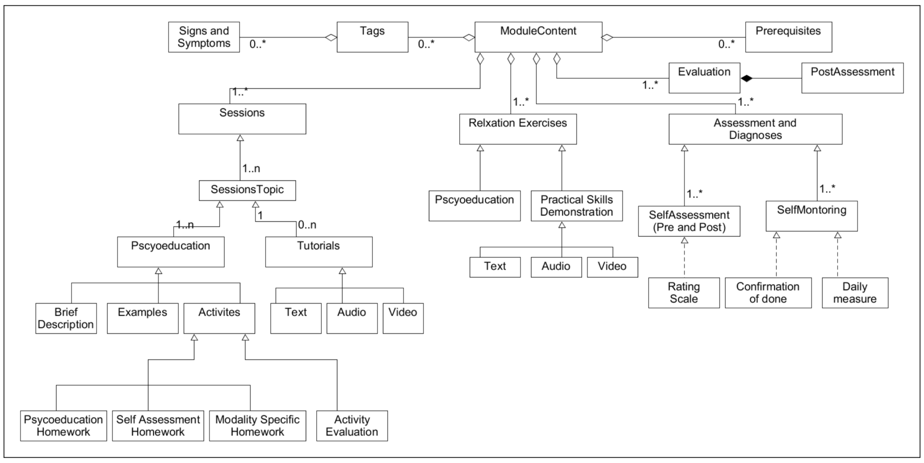


Figure 2.2: COPE module content [48].

Chapter 3

Design and Method

This chapter will describe the chosen research methodology, design science, and a set of guidelines for how to conduct research using this methodology, along with a brief argumentation for how these guidelines have been covered.

Furthermore, this chapter will provide a more detailed description of the requirements for the artifact, the design choices, and the iterations performed throughout the design process.

3.1 Research Method

As described by Hevner et al. [26], there are two research paradigms that make up a majority of the research in the field of Information Systems. These two paradigms are *behavioral science* and *design science*, of which the former aims to develop and verify theories concerning human or organizational behavior, while the latter *"seeks to extend the boundaries of human and organizational capabilities by creating new and innovative artifacts"* [26]. As the research at hand involves the creation of a digital artifact, this suggests that design science may be the most appropriate research methodology.

3.1.1 Design Science

With roots in engineering and the sciences of the artificial, design science is essentially a problem-solving paradigm that aims to solve identified organizational problems through creating and evaluating digital artifacts, by defining ideas, practices, products, and technical capabilities through innovations where analysis, design, implementation, management, and use of information systems can be achieved effectively. The key to design science is a distinct contribution to the knowledge base, through either finding a more efficient way of addressing a previously solved problem or finding an innovative or unique way of addressing important unsolved problems [26].

In their paper, Hevner et al. [26] describe a set of seven guidelines for conducting and evaluating design science research. These guidelines are referenced

Guideline	Description
Guideline 1	**Design as an Artifact:** Design-science research must produce a viable artifact in the form of a construct, a model, a method, or an instantiation.
Guideline 2	**Problem Relevance:** The objective of design-science research is to develop technology-based solutions to important and relevant business problems.
Guideline 3	**Design Evaluation:** The utility, quality, and efficacy of a design artifact must be rigorously demonstrated via well-executed evaluation methods.
Guideline 4	**Research Contributions:** Effective design-science research must provide clear and verifiable contributions in the areas of the design artifact, design foundations, and/or design methodologies.
Guideline 5	**Research Rigor:** Design-science research relies upon the application of rigorous methods in both the construction and evaluation of the design artifact.
Guideline 6	**Design as a Search Process:** The search for an effective artifact requires utilizing available means to reach desired ends while satisfying laws in the problem environment.
Guideline 7	**Communication of Research:** Design-science research must be presented effectively both to technology-oriented as well as management-oriented audiences.

Table 3.1: Design-Science Research Guidelines [26].

and summarized in Table 3.1. In the context of this thesis, the guidelines represent the following:

Guideline 1: The resulting product of this thesis is an expanding artifact for the IDPT system that Mukhiya is currently working on. The artifact comprises an interactive graphical visualization of psychological interventions (or "cases", as they are called in the framework), as well as their underlying modules with associated tasks, and thereby covers Guideline 1.

Guideline 2: Problem relevance was covered in Chapter 1. There is a need for more and better tools to reduce patient dropout rates in the online mental healthcare system, and covering Guideline 2, Chapter 1 argues that facilitating dynamic assessment of psychological interventions may be a viable solution to that problem.

Guideline 3: The artifact can present a clear graphical Overview of available psychological interventions with related substructures and facilitate the dynamic assessment of such interventions. The evaluation of this artifact, conforming to Guideline 3, is covered in Chapter 5.

Guideline 4: This guideline is covered in its own section, as well as the answers to the research questions in Chapter 6.

Guideline 5: Research rigor is covered through the description of the development and design process in Chapters 3 and 4, and through the evaluation in Chapter 5.

Guideline 6: The search process towards finding a solution to the problems defined in Chapter 1 is described in more detail later in this chapter. The various iterations of the development described in Section 3.2 thus cover Guideline 6.

Guideline 7: Finally, the communication of the research in its entirety, covering Guideline 7, is carried out through this thesis.

3.2 Development and Design Process

Throughout the development process, at the transition between the iterations, progress was evaluated through meetings. At these meetings, implemented features and design choices from the previous iteration were discussed, along with suggestions for adjustments and improvements that could be considered through the upcoming iteration.

3.2.1 Iterations

In short, the complete process of the development can be described through the iterations listed below. Although some design elements have been implemented or adjusted over multiple iterations, the following points are the main pillars of the artifact.

1. Identifying problems and determining requirements
2. Choosing a framework for visualization
3. Designing a graphical visualization
4. Implementing and designing the Overview page

Identifying problems and determining requirements

The choice of requirements for the artifact constitutes a natural first iteration in the development process, and as described in Chapter 1, this thesis seeks to help reducing patient dropout rates for IDPT systems through facilitating dynamic assessment of the treatment elements. The leading idea for how to achieve this goal was to find a way to present a complete graphical Overview of the available psychological interventions with their associated modules and tasks, and then use this graphical presentation to visualize what elements could be recommended to complete, and preferably in what order.

Understanding the components of the IDPT system is an important part of deciding what elements to include in the Overview. In the GitBook repository [34] corresponding to the GitHub project for the IDPT framework, Mukhiya has started producing a description of the architecture of the IDPT system, where the architecture is associated with a Learning Management System (LMS), and the main components are explained through comparisons to the LMS scenario.

Cases: In the GitBook description mentioned above, the cases are compared to the LMS element "Course" and are, in the context of this thesis, mostly referred to as "interventions". These cases are structured treatment programs designed for treating a specific disorder or issue, such as depression, ADHD, anxiety, etc. An IDPT system typically consists of at least one such case, and as described in Figure 3.1, a case includes one or more modules.

Modules: Associated with the context of LMS, the modules are compared to a "Chapter". Each module addresses a specific part of the case, such as a module for breathing exercises for ADHD or a module for sleeping or concentration issues for depression. A distinct module can be added to one or more cases and consists of at least one task, as illustrated in Figure 3.1.

Tasks: The last comparison Mukhiya presents, in the aforementioned GitBook description, as to how the IDPT system relates to the LMS scenario, is the correlation between IDPT tasks and LMS "Lessons". A task can be of one in two categories, *informative* or *interactive*, and contributes to the IDPT system's collection of passive or active data, respectively, where the main difference is as follows:

- Informative tasks are providing learning material intended to psychoeducate patients and their families about the topic at hand, e.g., the mental health issue itself, symptoms and causes, and self-help programs. These learning materials can be presented as text, audio, video, or a mix of two or more.
- Interactive tasks require more considerable participation from the patient, either in the form of physical activities (e.g., breathing exercise, walking, workout, stretching, etc.), or through computerized tasks (e.g., question answering, multiple-choice questionnaires, feedback, etc.)

Assignments: In the IDPT system, an assignment is a form of a computerized interactive task and can be of type "Survey", "Quiz", or "Psychometric assessment".

As the framework in question is still an early-stage prototype at the time of writing this thesis, there are some elements in the framework that have not been included in the Overview. One example of such is assignments, which is intended to be a sub-element as part of tasks. Here, the functionality allowing it to be opened as a "View" is completely missing, and the functionality allowing it to be added to a task is non-complete. The time frame for this Master's thesis taken into account, the assignments-component was chosen not to be included in the Overview. This could be suggested as a part of further work.

Another challenge that influenced the choice of requirements for the artifact was the missing user profiling. As user profiling was not implemented before or during the work on the artifact, this inconvenienced the work on adaptiveness. Due to the time frame of the project, the implementation of adaptiveness was somewhat downgraded in favor of facilitating adaptiveness.

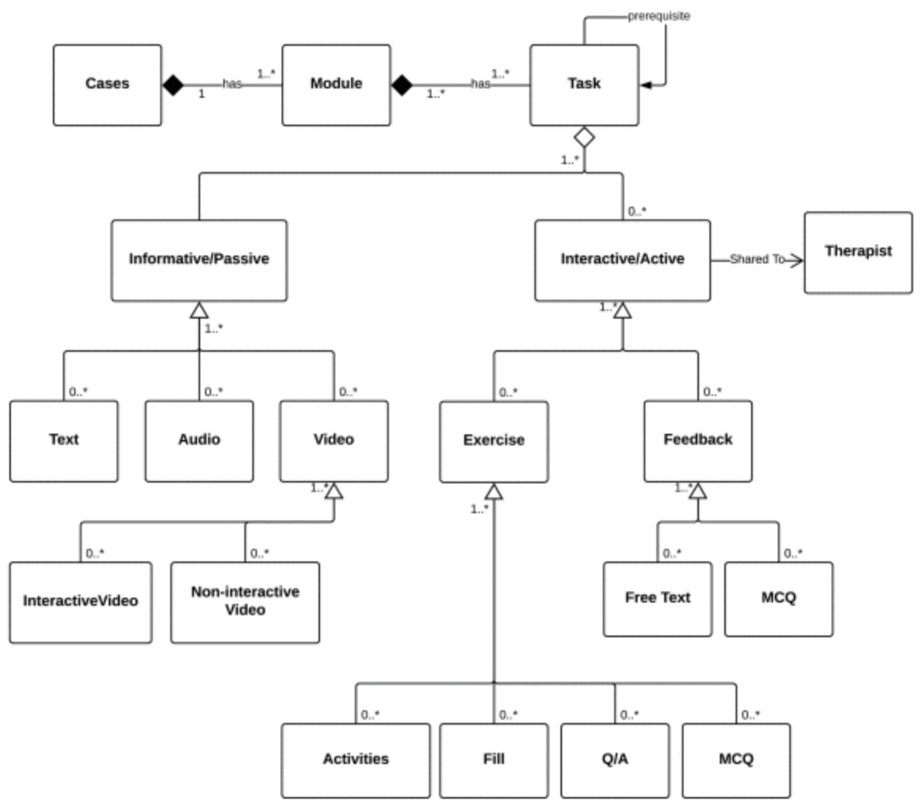


Figure 3.1: The conceptual model of IDPT [36]. *Note: The figure is taken from [36], but has been adjusted by removing some white space between the entities to enhance the readability of small text.*

In conclusion, the requirements for the artifact were chosen to be presenting the different elements of the treatments in the form of a distinct Overview that can visualize the correlations between the elements and enable an interaction-based functionality to view or edit details of the elements.

Choosing a framework for visualization

As the requirements of the artifact involve visualization of the entire treatment structure, there is a need for a framework that enables the visualization of graphs. The frontend of the system to be expanded with the artifact is developed with ReactJS, which is an open-source project under active development. Since it is an open-source project, this may indicate that the framework needed for the visualization is, to some extent lacking solid documentation. Although lack of documentation does not necessarily mean that the framework is of poor quality, it can still have a bearing on the process of finding a suitable framework. Thus, the determination process could be described through its own iteration.

Deciding what framework to use required the following three fundamental steps:

1. Specify the requirements and preferences for the resulting graph.
2. Search for alternative frameworks.
3. Compare and conclude.

Regarding the first step, it is important to note that the modules and tasks within an intervention do not necessarily have to be performed in a linear or fixed chronological order. Moreover, there may also be items that can be ignored in some cases. To satisfy this detail, it is necessary that the graph can present instances of nonlinear interconnections, suggesting a form of a network graph.

Furthermore, there were several points that were taken into account, one of which was the number of nodes. As an intervention may contain a large number of modules, and a module may correspondingly contain a large number of tasks, a graph can end up consisting of a considerable number of nodes. Such cases could require a functionality enabling a manual enhancement of the visualization through user interaction. Thus, the graph should be able to zoom in and out, as well as being moved around within its frame.

Related to the number of nodes, is the *placement* of nodes. Hard-coding coordinates for the nodes has its advantages and disadvantages, where arguments in favor include faster rendering and the same outcome for each render, while arguments against include the multitude of different possible combinations of nodes as well as the possibly extensive algorithm required to perform the assignment of coordinates. In conclusion, the alternatives are weak hard-coding or built-in algorithm from the framework in question. Either way, there is a risk for overlapping nodes or, particularly edges, which require a failsafe functionality that enables manual movement of nodes through user interaction.

The last requirement for the graph was the ability to link custom key/value pairs within a node, as this functionality could simplify the processing of user interactions and thereby possibly reduce the use of data and time needed to render the graph. In other words, the node object should be able to contain a number of properties that do not affect how the graph framework interprets the node (e.g., if needed, a case node should be able to hold a list of the IDs belonging to the modules included in said case).

When the requirements for the graph were concluded, the next step was finding alternatives. To assist in completing this task, the platform *Openbase* is a powerful tool. With its 1.5 million open-source packages monitored [38], Openbase is aiming to become “the Yelp of open-source packages” [49, 9]. Through this platform, a number of different frameworks were found, and some of these were rejected due to solely supporting charts, implementation of a graph editor, or graphs with an acyclic tree structure.

After these rejections were made, a handful of packages were left, from Openbase’s list of recommended graph libraries:

dagre-d3-react: Based on the screenshot of the sample graph provided in the readme [39, 11], as shown in Figure 3.2, this package gives a nice first impression. However, this package does not have as much feedback compared to some of the other candidates, and the last commit is about a year old, indicating that the package may be deprecated or at least lacking continuous maintenance [11]. For this reason, dagre-d3-react was rejected.

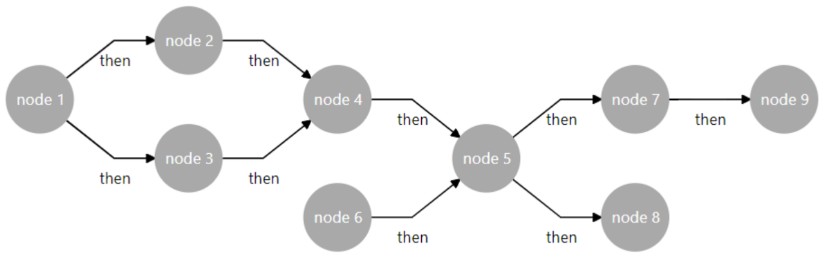


Figure 3.2: A demo graph for dagre-d3-react [11].

react-d3-graph: Unlike the previous package, this one still seems to be regularly maintained, judging from the dates of the open items in their issues log put in relation to recent commits [8]. After a quick review of intro documentation, this package seemed to comply with all requirements and thus was a possible further candidate.

react-vis-force: After the rejection of dagre-d3-react, and the reasoning behind that choice, it was a quick decision to reject this package as well, since the latest commit was from 2017 [55].

react-graph-vis: The last candidate was also among the most recommended packages on Openbase's list, and based on the issues log and commit log, this package seems to undergo satisfyingly sustainable maintenance [30]. As this package seemed to comply with all requirements, it was chosen as the second and last main candidate.

For the third and final step in choosing a visualization framework, the two candidates react-d3-graph and react-graph-vis were further compared against each other. As both were able to satisfy all requirements, this last comparison was somewhat characterized by personal preferences in terms of appearance and implementability.

Apart from the two frameworks being built on different fundamental libraries, namely *D3* and *vis.js*, one of the first differences that were registered were the default settings for the graphs. As shown in Figures 3.3 and 3.4, the node labels are centered inside the node, for react-graph-vis, while outside the nodes, for react-d3-graph. In addition, the edges are black and directed, for react-graph-vis, while they are grey and undirected for react-d3-graph. These differences are, of course, quick to fix. However, they still represent a possibly greater need for adjustments in the D3-based package compared to that of Vis.

In terms of configurations, both packages have extensive documentation describing what they are capable of, and both packages have access to configuration options at the foundation level of their respective base libraries. Based

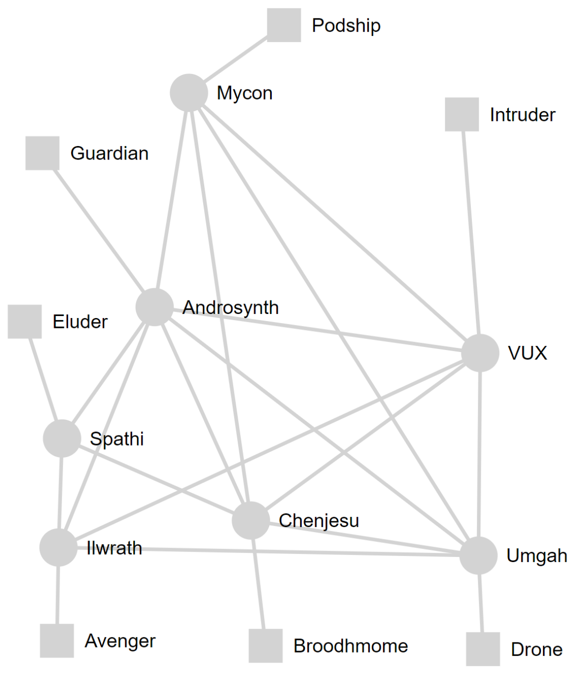


Figure 3.3: A demo graph for react-d3-graph, created with the “d3 live playground” [43].

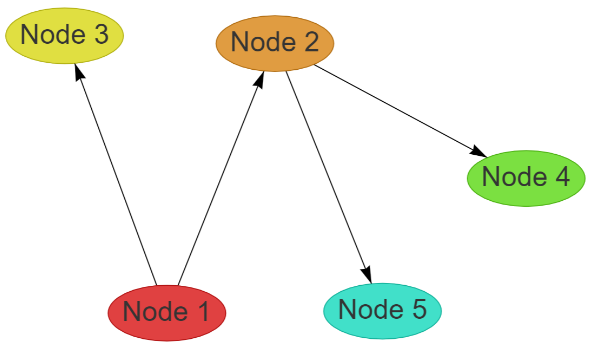


Figure 3.4: A demo graph for react-graph-vis [44].

on this documentation, the D3-package seems to have more options than the Vis-package, which speaks positively for D3.

Nevertheless, there was one detail that ultimately led to the final decision:

Implementing react-graph-vis to get to know it and look at possibilities was a simple ”plug-and-play” operation, as it was enough to install react-graph-vis as “dev dependency”. react-d3-graph, on the other hand, turned out to be quite cumbersome to implement as adding the required package dependency triggered multiple “Dependency Not Found Errors”, and fixing these resulted in other “Type Errors”. Thus the final choice fell on react-graph-vis.

Designing a graphical visualization

Once the choice of the framework was made, the work of designing the first version of the Overview could start. The main focus throughout this third iteration was the implementation and preparation of a functioning artifact. The aim of this artifact was to display a complete intervention through an interactive graph that would comply with the algebraic definition of a directed multi-graph according to University of Bergen lecturer and Professor Uwe E. Wolter's “Script for the course INF 223, Spring 2020” [59]:

Definition 3.2.1 (Directed multi-graph)**.** A **graph** $G = (G_V, G_E, sc^G, tg^G)$ is given by:

- a collection G_V of **vertices**,
- a collection G_E of **edges**,
- a collection $sc^G : G_E \to G_V$ assigning to each edge its **source**, and
- a collection $tg^G : G_E \to G_V$ assigning to each edge its **target**.

$\square$

Building on the implementation from the previous iteration resulted in a simple “dummy-graph”, as shown in Figure 3.5. Apart from the edge between “Node 3” and “Node 5” seemingly pointing in both directions, it is possible to observe, from the figure, that this graph seems to follow the rules from Definition 3.2.1. However, the argument for why this is true will be described in more detail in Chapter 4. Here it is also explained how the double-pointed edge is actually two separate edges and thus does not break with the definition.

To advance on the implementation so that the graph could have the opportunity to depict an intervention, the next step in this iteration had to be introducing a test-case. An intervention used in this IDPT framework to illustrate the system features was a test-case based on an iCBT program for ADHD, consisting of seven modules ranging from “Week 1:(...)” to “Week 7:(...)”, and 17 tasks as parts of the different modules.

As previously described, meetings were held in connection with the transition between the iterations, and at one of these meetings, an initial draft was presented for how the graph could behave. The draft was described in accordance with Figure 3.6, with the following functionality:

- **Figure 3.6 - Part A:** This is the initial view, where no nodes are selected, and all interventions are represented with their own node. A user-click on a node will select said node, as shown in Part B.

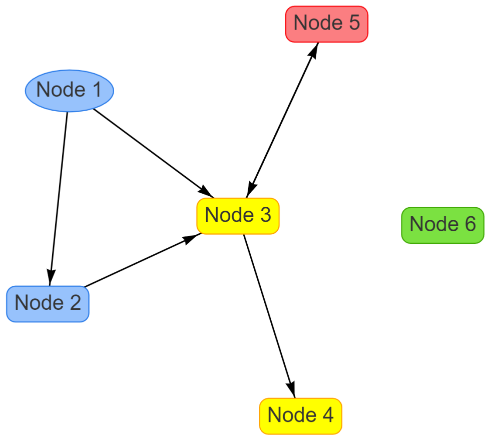


Figure 3.5: A dummy graph used to get familiar with react-graph-vis.

- **Figure 3.6 - Part B:** A selected intervention node displaying the associated modules with their connections as a sub-graph within. A user-click on a module node will select said node, as shown in Part C.
- **Figure 3.6 - Part C:** A selected intervention node with a selected module node, displaying the tasks corresponding to the said module as a new sub-graph. A user-click on a task node will redirect to the "View page" corresponding to that task.
- A user-click on a selected node will deselect said node and revert to the previous state, and a user-click on the white background of the graph will revert the graph to its initial view.

At the aforementioned meeting, the supervisors provided multiple main remarks as feedback concerning the appearance of the graph. They expressed concern on whether the multi-nested sub-graph structure would be overly complex and whether it would be a better option to break down the structure of the graph into several separate components representing interventions, modules, and tasks, respectively. Thus, the previously described functionality could activate and deactivate separate sub-graphs without the executing graph changing and becoming too extensive. In addition, it was recorded that the edges of the module-related sub-graph disappeared when a module node was selected.

The final feedback remark addressed the redirection from a task node to the View page corresponding to that node and whether this could work against the purpose of the graph, in that multiple informational elements from the graph would disappear upon redirection. Related to this remark, it was noted that this graph did not have the ability to display detailed information about modules or interventions, as only tasks could redirect to the View page. A proposed alternative solution was that the related View page could be implemented as part of the Overview page, and thus both modules and interventions could have the same functionality.

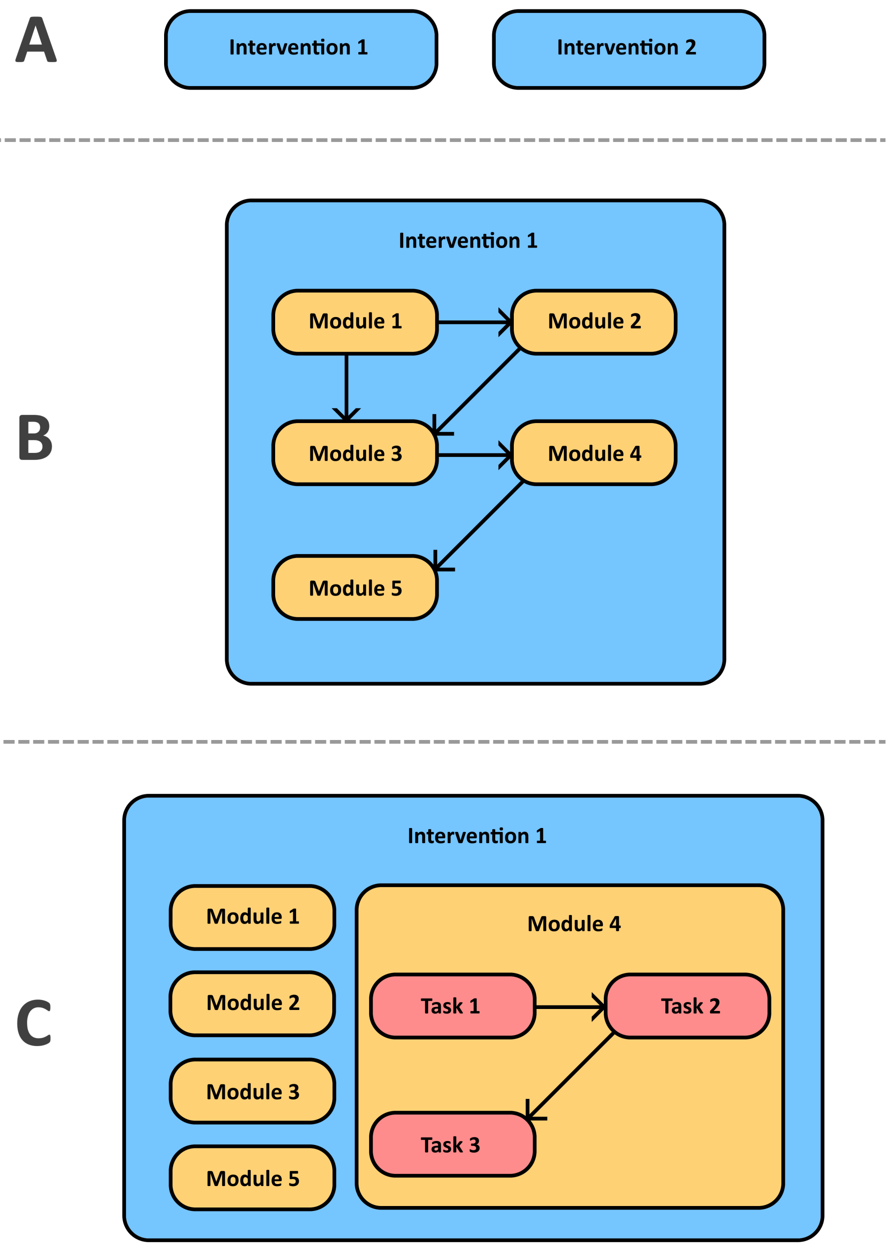


Figure 3.6: An initial draft for a potential graph appearance. **A** depicts the different interventions, **B** depicts a selected intervention, and **C** depicts a selected intervention with a selected module.

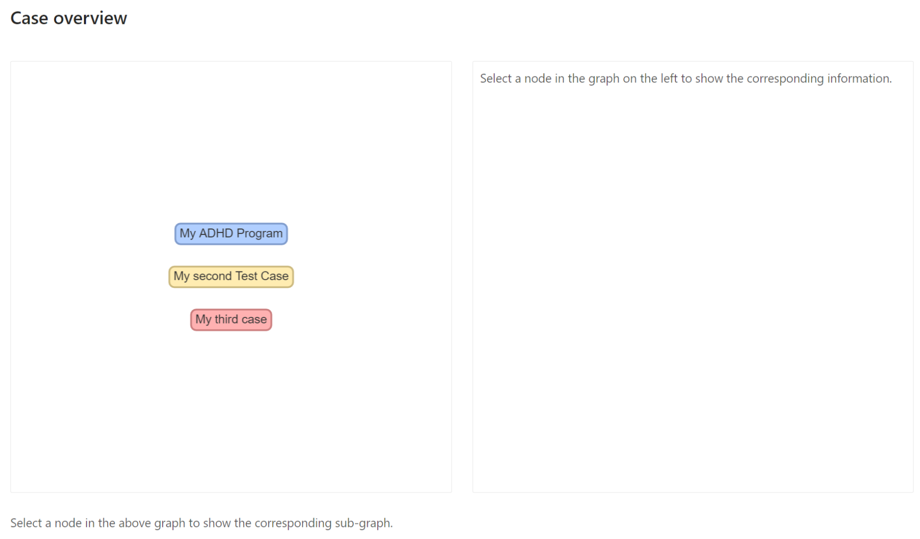
All remarks from this feedback were included in the upcoming iteration and led to a comprehensive change in the graph structure, as well as the overall design of the Overview page.

Implementing and designing the Overview page

For the final iteration, the full Overview page was designed and implemented in accordance with the functionality description on page 21, related to Figure 3.6, as well as the feedback from the meetings.

The complete and final result of the artifact is one page consisting of three main components split into each their own row – one for each entity, cases, modules, and tasks – and the first of these is the "Case Overview".

As shown in Figure 3.7, the Case Overview consists of two columns, or cards, where the graph is located in the card on the left, while the information corresponding to the selected node from the graph should be displayed inside the card on the right. In Figure 3.7, however, there is no selected node. Instead, the rightmost card displays an instructive text, telling the user to "select a node in the graph on the left to show the corresponding information". Similarly, this instructive text is displayed in the rows for modules and tasks when there is no selected node on the corresponding graph. In addition, in the next row, the text "select a node in the above graph to show the corresponding sub-graph" is displayed when there is no current selected node. This text is displayed below the module row as well when there is no selected module node.

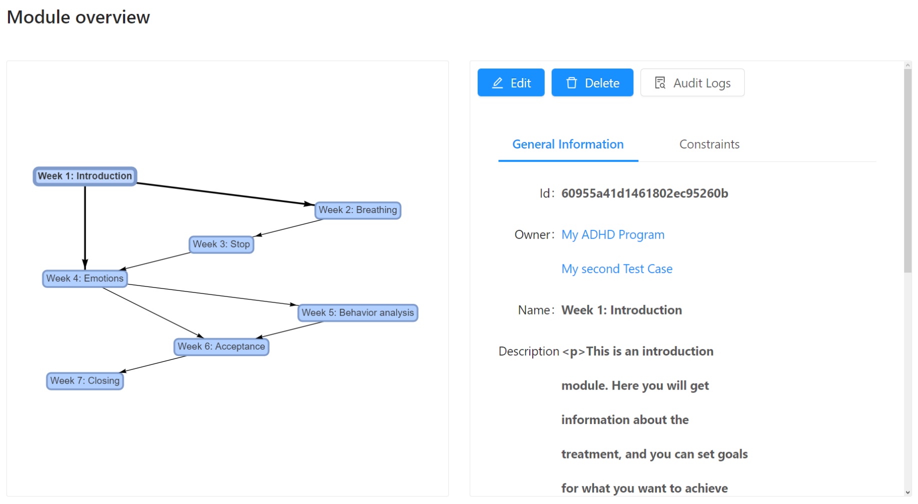

Case overview

Select a node in the graph on the left to show the corresponding information.

Select a node in the above graph to show the corresponding sub-graph.

Figure 3.7: The first row of the Overview, showing the Case Overview with no selected node.

Regarding the colors of the nodes in Figure 3.7, three different colors have been implemented at present, illustrating the usability status of the element represented by the node. The color red indicates that the represented element is currently a "draft" and thus not ready for use, while yellow and blue indicate that the element is "inactive" or "active", respectively. Note that "(in)active", in this case, does not refer to whether the user has clicked on the node but rather the content of the node itself.



Id: 60955a41d1461802ec95260b

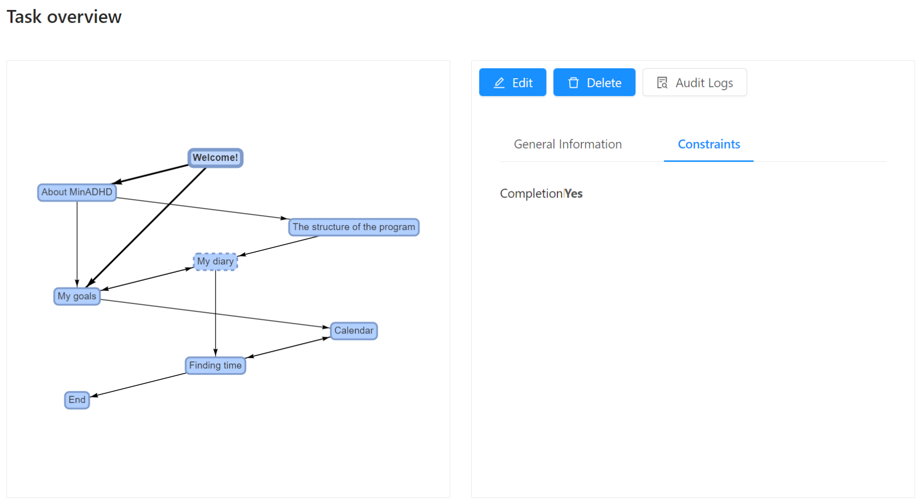
Owner: My ADHD Program

My second Test Case

Name: Week 1: Introduction

Description <p>This is an introduction module. Here you will get information about the treatment, and you can set goals for what you want to achieve

Figure 3.8: The second row of the Overview, showing the Module Overview with “Week 1:(...)” as the selected node.



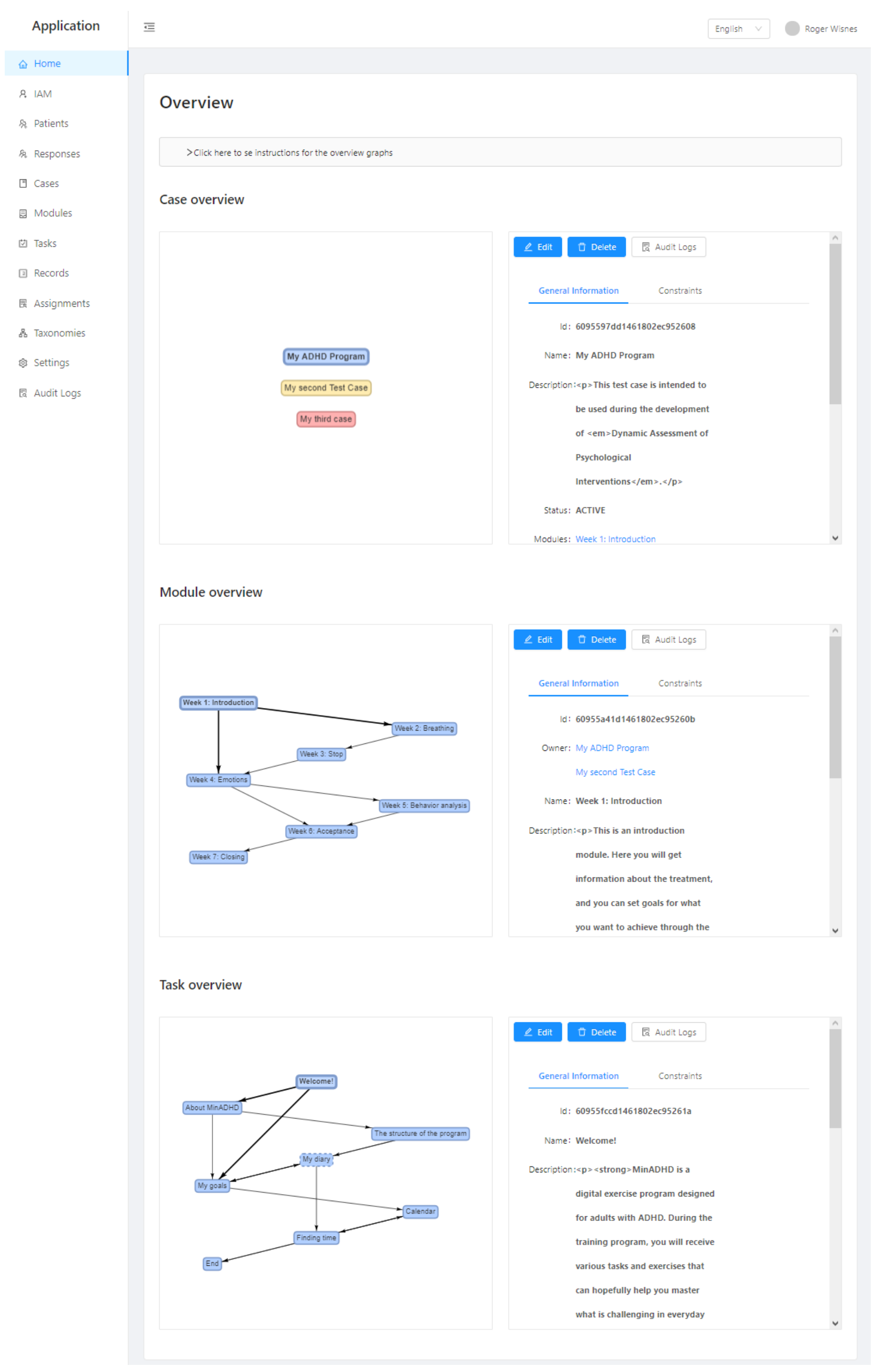
Completion Yes

Figure 3.9: The third and final row of the Overview, showing the Task Overview with “Welcome!” as the selected node and “Constraints” as the active tab on the View page.

When the user has selected the (in this case) blue node, the next row of the Overview is displayed – the “Module Overview”. In execution and design, this row is exactly the same as the first, with the exception of the graph. As presented in Figure 3.8, this graph visualizes the modules that are parts of the selected node from the case-graph. In this figure, a node “Week 1:(...)” is selected, thus showing the View page corresponding to the selected module, in addition to highlighting the edges where the source node is the selected one, thereby indicating which modules can be performed next.

The last component row in the Overview is "Task Overview", and can be seen in Figure 3.9. This row shows the tasks that belong to the selected module node in the graph above, and "Constraints" is selected, in the figure, as the active tab on the View page. The constraints tab is intended to be a tab for holding all the different information elements that apply to the entity in question and that can be used as inclusion and exclusion criteria.

Finally, the overall design of the Overview page can be seen in Figure 3.10. Here, it is demonstrated that the Overview has not received its own item in the navigation menu but has instead been placed on the Home page for easy access. In addition, there is one last detail just below the main heading – a collapse box displaying the text "Click here to see instructions for the Overview graphs" – intended to keep the page clean but still be able to provide instructions and explanations regarding the use and appearance of the graphs.



Overview

>Click here to se instructions for the overview graphs

Case overview

Id: 6095597dd1461802ec952608

Name: My ADHD Program

Description: <p>This test case is intended to be used during the development of <em>Dynamic Assessment of Psychological Interventions</em>.</p>

Status: ACTIVE

Modules: Week 1: Introduction

Module overview

Id: 60955a41d1461802ec95260b

Owner: My ADHD Program
My second Test Case

Name: Week 1: Introduction

Description: <p>This is an introduction module. Here you will get information about the treatment, and you can set goals for what you want to achieve through the

Task overview

Id: 60955fccd1461802ec95261a

Name: Welcome!

Description: <p><strong>MinADHD is a digital exercise program designed for adults with ADHD. During the training program, you will receive various tasks and exercises that can hopefully help you master what is challenging in everyday

Figure 3.10: The overall design of the full Overview.

Chapter 4

Implementation

Now that the design process is covered, this chapter can go into more detail about the implementation. Through the following section, the technologies that have been used to create the artifact will be presented, and then the succeeding will address the architectural layout, along with the Separation of Concerns (SoC) as a design pattern.

Finally, the last section will focus on the artifact, how the graphs comply with Definition 3.2.1 from the previous chapter, as well as the final implementation of the graphs, state-management, frontend communication with backend, and the composition of the Overview page.

4.1 Tools and Frameworks

A number of different technologies have been involved in the production of the artifact, some more extensive than others, and in this section, the most important technologies and their significance for the end result will be presented. Although there have been some changes in the backend, the majority of the work done in the development process has been with the frontend. The following technology descriptions are somewhat affected by this.

4.1.1 ReactJS

The IDPT system that was to be expanded by the artifact was mainly developed with JavaScript and ReactJS [13], a JavaScript library for building User Interfaces (UIs), which has become exceedingly popular over the years and ended up in second place among the "Most Loved Web Frameworks" of 2020, according to Stack Overflow's survey from February 2020 [53]. JavaScript with ReactJS was, therefore, a natural choice of main technology for the artifact, and thus, in terms of significance, JavaScript and ReactJS have been the most important technologies for the project as a whole.

4.1.2 Vis

As described in Subsection 3.2.1, a separate selection process was carried out to find a framework suitable for the production and display of the graphs, and the final choice was react-graph-vis [30] – a React component, building on the library vis.js [56]. The three graphs in the artifact form a considerable part of the end result, which reflects the significance of this framework in connection with the Master's project.

4.1.3 GraphQL

In terms of frontend to backend communication, in order to query information from the database, GraphQL [21] was already utilized in the original system. This query language is designed to deliver exactly what information is queried and nothing more, and as a consequence, it reduces the risk of data leakage and excessive data transfers. In addition, GraphQL can be used as an architectural layout, which will be further described in Section 4.2. Throughout the development of the artifact, this was used to retrieve the information needed to produce the graphs.

4.1.4 Redux

Redux [46] is a state container that can be used when an app contains large amounts of application state that is needed in several places in the code. This library played a major role in the frontend, in connection with the temporary storage of data received through the use of GraphQL. Similar to GraphQL, JavaScript, and ReactJS, this library was implemented before the development of the artifact was started.

4.1.5 Ant Design

For parts of the overall UI designed in the system, Ant Design [2] is used, and this applies to the artifact as well. In addition to being the source framework for the previously described rows, columns, and cards, it is also used to provide the collapse box at the top of the Overview page. Hence, Ant Design had a good impact on the UI design of the artifact.

4.1.6 MongoDB

As GraphQL is used to interact with the database, the document-based, distributed database MongoDB [32] was somewhat included in the work. Here, some additions to the repository were needed to make query filtering work as desired. MongoDB thus had some significance for the artifact.

4.1.7 Other

Besides the JetBrains IDE WebStorm, the version control tool GitHub, and the above technologies, StyledComponents [54] was the only additional technology used in the development process. This was used to make headlines on the Overview page and thus had no considerable significance for the artifact as a whole.

4.2 System Architecture and Design Patterns

Although the production of the artifact did not involve any new implementation of architecture or other considerable design patterns, these already implemented aspects had some relevance to the development process. In the following subsections, the two most influential elements will be examined.

4.2.1 GraphQL as an Architecture

GraphQL is agnostic to both transport-layer and database, resulting in the ability to cooperate with any network protocol as well as both SQL and NoSQL databases – such as MongoDB, in this case – contributing to the practicability of its usage at an architectural level. Fundamentally, there are three different architectures built with a GraphQL server: A GraphQL server with a connected database, a GraphQL layer integrating existing systems, and finally, a hybrid of the previous two.

The architectural layout used in the IDPT system involved in this thesis is built on the first of the three architectures, which is illustrated in Figure 4.1. Out of the three, this layout is most common for newer projects that do not have existing systems and consists mainly of a server implementing the GraphQL specifications and an integrated database.

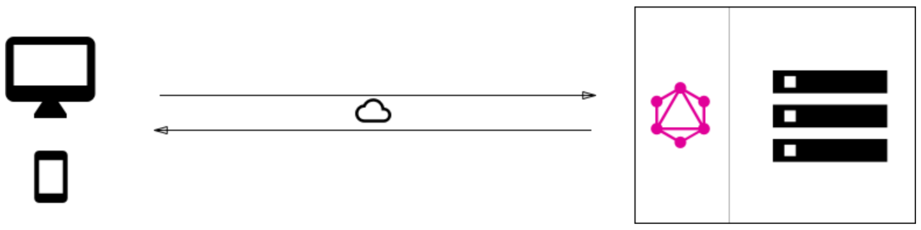
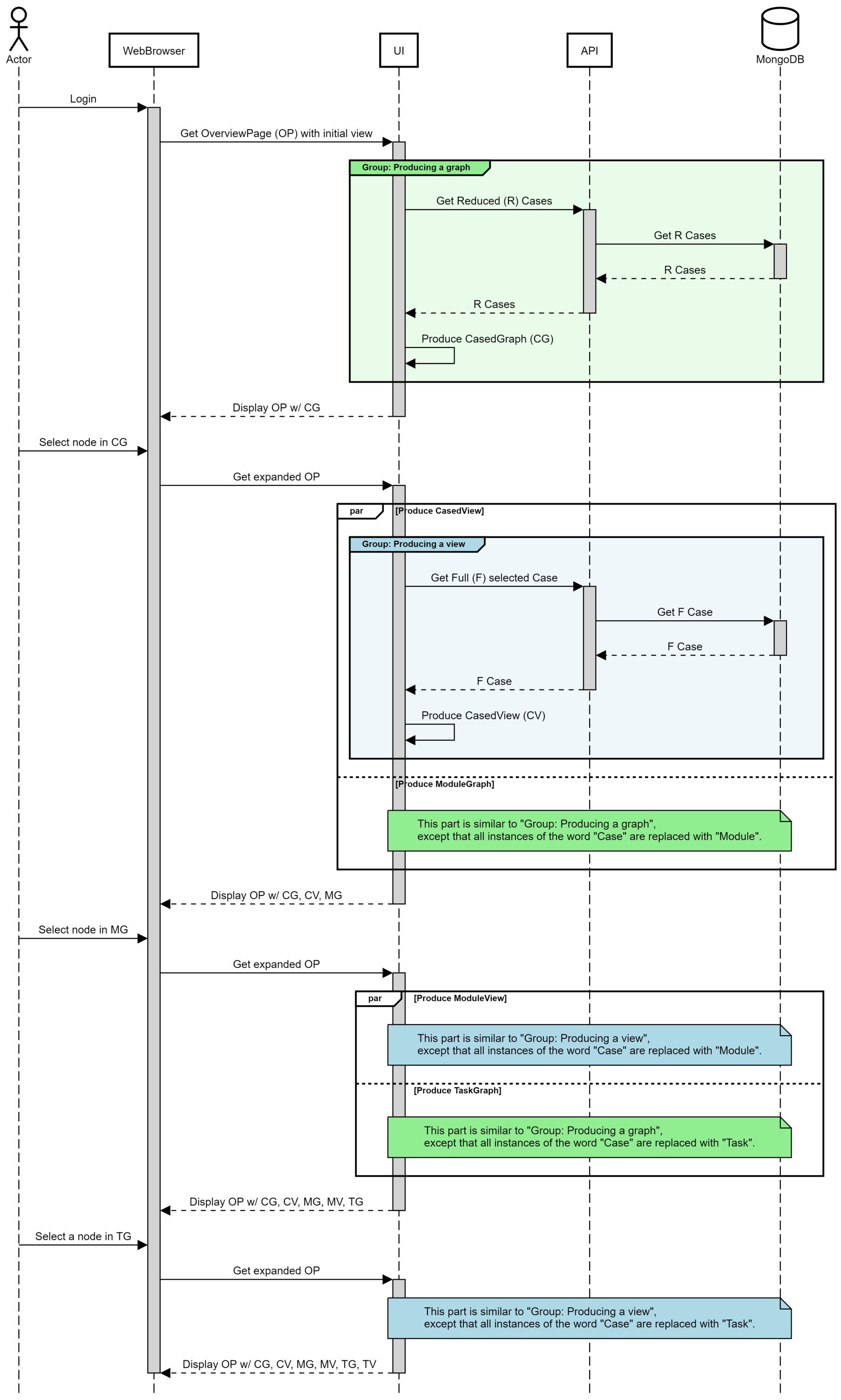


Figure 4.1: The GraphQL architecture, consisting of one GraphQL server with one integrated database [20].

Put in context with the sequence diagram in Figure 4.2, the participant's *WebBrowser* and *UI* can be interpreted as the client-side, while the participant *API* and the database *MongoDB* can be interpreted as the server-side, as presented in Figure 4.1. The activity pattern illustrated through the sequence diagram in Figure 4.2 shows how the GraphQL architecture works:

- A client-side action triggers a query transmission.
- The GraphQL server receives the query payload and requests the related data from the database. This is done through *resolvers*.
- The data received from the database is then used to construct a *response object*, which is returned to the client-side.

Regarding the sequence diagram in Figure 4.2, it should be noted that the three user interactions "Select node in (...)" can be performed multiple times, as long as there is a selected node in the above graph.



Actor
WebBrowser
UI
API
MongoDB
Login
Get OverviewPage (OP) with initial view
Group: Producing a graph
Get Reduced (R) Cases
Get R Cases
R Cases
R Cases
Produce CasedGraph (CG)
Display OP w/ CG
Select node in CG
Get expanded OP
par
[Produce CasedView]
Group: Producing a view
Get Full (F) selected Case
Get F Case
F Case
F Case
Produce CasedView (CV)
[Produce ModuleGraph]
This part is similar to "Group: Producing a graph", except that all instances of the word "Case" are replaced with "Module".
Display OP w/ CG, CV, MG
Select node in MG
Get expanded OP
par
[Produce ModuleView]
This part is similar to "Group: Producing a view", except that all instances of the word "Case" are replaced with "Module".
[Produce TaskGraph]
This part is similar to "Group: Producing a graph", except that all instances of the word "Case" are replaced with "Task".
Display OP w/ CG, CV, MG, MV, TG
Select a node in TG
Get expanded OP
This part is similar to "Group: Producing a view", except that all instances of the word "Case" are replaced with "Task".
Display OP w/ CG, CV, MG, MV, TG, TV

Figure 4.2: Sequence diagram for the graphs.

4.2.2 Separation of Concerns

The principle of Separation of Concerns (SoC) is widely used in software development and architecture and is related to the Single Responsibility Principle (SRP), and the Don't Repeat Yourself (DRY) principle. Both SRP and DRY are, to some extent, used in the system; however not strict enough to gain much attention in this thesis. SoC, on the other hand, is used to a greater extent both at the architectural and detail level. The idea is to prevent co-location of design or code associated with different features of the software, meaning that – with the principle of SoC – when the software needs modification, multiple factors will influence the time and effort needed to apply those modifications. These factors include:

- **The amount of code in need of modification:** If SoC is thoroughly practiced, all code relevant to a specific behavior in the application will be separated from the rest of the code. This suggests that it should only be necessary to change code that is directly associated with the modification in question. In the context of the artifact from this Master's thesis, this implicates that changing the design of the task graph will only require changes to the one file that is responsible for producing this graph.
- **The difficulty of the modification's implementation:** Since code belonging to specific functionality or behavior is separated from the rest of the application, leading to fewer changes being necessary to perform a modification, this will likely reduce the need to entirely understand or modify other code, and thereby make it easier to find out where the changes are to be made. A strict practice of SoC, therefore, means that if someone should want to change from *Vis*- to *D3*-implementation for this artifact's module graph, then they only need to understand and modify the code that applies to that graph – in this case, one single file.
- **The probability that other existing code or features will break:** In cases where multiple concerns are not separated, the developer may unintentionally change code associated with one feature when another feature was the target. Since code that needs to be modified is more likely to break compared to code that is not touched, SoC is a powerful tool to reduce the likelihood of breakage in unrelated features. Hence, a modification of the code related to the task graph is less likely to break any code related to the module graph.

4.3 The Artifact

There are a number of ways to implement new features in an existing system, and when a software developer is faced with a major challenge, it is quite common to try different approaches before succeeding. This section will not take into account developmental digressions as a consequence of this but rather describe the implementations in functional chronological order, starting with the graph.

4.3.1 Implementing the Graph

The implementation of react-graph-vis works as presented by Listing 4.1: The graph component takes in three parameters “graph”, “options”, and “events”, which represent the graph’s data (nodes and edges), configurations, and interaction functions, respectively. Finally, the graph component has a callback function “getNetwork” that allows the class to store the graph instance through the use of state – more on state-management in Subsection 4.3.2.

```
<Graph
  graph={this.getData()}
  options={this.options}
  events={this.events}
  getNetwork={network => {
    this.setState({network})
  }}
/>
```

Listing 4.1: The Graph component.

Implementing by definition

As described in Chapter 3, it was decided to start by implementing a dummy graph since this implementation had already been partly done in connection with the choice of visualization framework. The first intention of the implementation was to achieve a result complying with Definition 3.2.1. Section 3.2 described a version of this dummy graph, where two of the nodes appeared to have one single bi-directional edge between them (see Figure 3.5). However, while depicting two distinct edges in its place, Figure 4.3 is the exact same graph, with lines four through eight in Listing 4.2 as the only code difference. These five lines of code provide a “smoothing” of the edges in a clockwise curve, thus preventing a complete overlap between non-equivalent edges.

In Figure 4.3, all nodes and edges are marked with their own red digits or blue letters, respectively. Put in relation to Listing 4.3, these digits and letters represent the IDs of the nodes and edges. For the purpose of linking the graph to Definition 3.2.1, the following proposed notation will be used:

- Vertices (or nodes) are given by the collection of node IDs: $G_V = \{1, 2, 3, 4, 5, 6\}$,
- Edges are given by the collection edge IDs: $G_E = \{a, b, c, d, e, f\}$,
- Sources are given by the collection $sc^G = \{(i, x), ..., (j, y)\}, (i, j) \in G_E, (x, y) \in G_V$, and
- Targets are given by the collection $tg^G = \{(i, n), ..., (j, m)\}, (i, j) \in G_E, (n, m) \in G_V$.

Since all edges must have one source and one target node, sc^G and tg^G must form a paired collection of tuples pointing to the same edge ID. By building on the notation, this means that it is possible to merge sc^G and tg^G, so that the new notation becomes: $st^G = \{(i, x, n), ..., (j, y, m)\}, (i, j) \in G_E, (x, y, n, m) \in G_V$. These tuples are recognizable from the edges in Listing 4.3, and since the

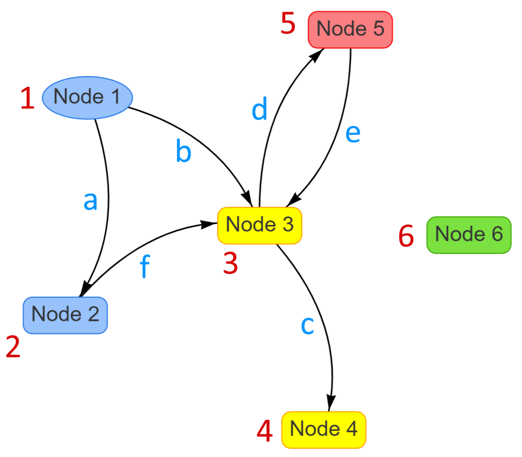


Figure 4.3: The dummy graph from Figure 3.5, but with rounded edges.

```
options = {
  (...)
  edges: {
    smooth: {
      enabled: true,
      type: "curvedCW",
      roundness: .3
    },
  },
};
```

Listing 4.2: Smoothing of the graph's edges.

```
getData = () => {
  return {
    nodes: [
      {id: 1, group: "g1", shape: "ellipse", label: "Node 1"},
      {id: 2, group: "g1", label: "Node 2", x: 0, y: 200},
      {id: 3, group: "g2", label: "Node 3", x: 200, y: 0},
      {id: 4, group: "g2", label: "Node 4", x: 200, y: 200},
      {id: 5, group: "g3", label: "Node 5", x: 400, y: 0},
      {id: 6, group: "g4", label: "Node 6", x: 600, y: 0},
    ],
    edges: [
      {id: a, from: 1, to: 2},
      {id: b, from: 1, to: 3},
      {id: c, from: 3, to: 4},
      {id: d, from: 3, to: 5},
      {id: e, from: 5, to: 3},
      {id: f, from: 2, to: 3},
    ]
  };
};
```

Listing 4.3: The getData-function corresponding to the dummy graph in Figures 3.5 and 4.3.

elements of G_V are directly linked to the node IDs in the same listing, this means that the graph complies with Definition 3.2.1 by the implementation.

The final implementation

Unlike the hard-coded getData function for the dummy graph (see Listing 4.3), the end result is dynamically based on entity data received from the GraphQL response. To illustrate this difference, Listing 4.4 represents the getData function for the final module graph. Here the nodes are assigned the properties *ID* and *label* based on the module's corresponding properties *ID* and *name*. In other words, the node ID represents the automatically generated module ID from the database, which ensures that each node has a unique ID.

Furthermore, the Vis framework allows multiple different ways for a node to be assigned a color. In the code for the dummy graph, color assignment is done by Vis's integrated automation based on the property *group*. Although groups could be assigned specific colors using adjustments to the options parameter, it was chosen to follow the SoC principle so that the color of the node is given directly by the globally shared helper-constant `graphStatusColor` on the form `{'X': 'Y',}`, where `X` is a name-string corresponding to the three entity-related status-alternatives *ACTIVE*, *INACTIVE*, and *DRAFT*, as well as a failsafe alternative *DEFAULT*, while `Y` represents a hexadecimal color code associated with the status.

Finally, each node is assigned an x- and a y-value that represents the node's coordinates in the graph system. These values are produced with a simple counter-based algorithm that places each node in one of three columns and with a given vertical distance between each node. The arguments for the implementation of this algorithm will be described in more detail in Chapter 5.

Still examining Listing 4.4, the edges are created using a nested loop, where each node is assigned an ID based on a counter variable. In the event of an expansion of the artifact, which affects or uses edges from several of the graphs at the same time, or particularly if the graphs' data is to be stored in a database, it should be considered whether this ID assignment should be auto-generated in a more secure way. With the current implementation of the artifact, however, this method was deemed secure enough for the artifact's use and functionality, as the edges are currently used for the visualization purpose alone. Furthermore, each edge is assigned a property *from*, which represents the IDs of all modules registered as the prerequisite of the current module in the nested loop, and finally, the property *to* representing the ID of the current module.

The getData functions for the case and task graphs are built as almost exact copies of the module function. For the case graph, the construction of nodes is exactly the same, while edges are not produced at all due to the fact that the interventions have no interconnections that can be shown in the current implementation of the system. For the task graph's function, there are a few more differences:

- The nodes are given one additional property *shapeProperties*, defined as `completionRequired==="No" ? {borderDashes:[5,5]} : null`, where *completionRequired* is given by a Boolean field-variable belonging to the current task.

- The IDs of the edges are concatenated from `'taskEdge'`, and the sources and targets are defined as *from* the current task *to* the next.

```
getData = () => {
  const {moduleRows} = this.props;

  let data = {nodes: [], edges: []};
  let counters = {edgeId: 0, xValue: -200, yValue: 50};

  moduleRows.forEach(row => {
    const rowId = fields.id.forView(row.id);

    data.nodes.push({
      id: rowId,
      label: fields.name.forView(row.name),
      color: graphStatusColor[fields.status.forView(row.status)
                             || 'DEFAULT'],
      x: counters.xValue = (counters.xValue += 200) % 600,
      y: counters.yValue -= 50,
    });

    const prereq = fields.prerequisite.forView(row.prerequisite);
    prereq.forEach(pre =>
      data.edges.push({
        id: 'moduleEdge' + counters.edgeId++,
        from: fields.id.forView(pre.id),
        to: rowId
      })
    );
  });

  return data;
};
```

Listing 4.4: The getData-function corresponding to the final (Module) graph implementation.

The next parameter *options* provides the main configurations of the graphs, and with the exception of lines six and eight in Listing 4.5, which are not relevant to the case graph, all the graphs have the implementation shown in this listing. Apart from the last two properties, the rest is self-explanatory. The "hover" property causes elements of the graph to be highlighted when the mouse hovers over them, while the "physics" property set to false prevents Vis from using its built-in node placement algorithm.

Finally, regarding the last parameter *events*, all three graphs have exactly the same implementation, consisting of three different functions, as shown in Listing 4.6. The first event *selectNode* is triggered by the user interacting with the graph, and receives an event object that can be used to find the node ID, which in turn is used to trigger a dispatch to retrieve all information about the selected node from the database. The next event *deselectNode* is triggered in the same way as the previous one – if another node is selected, or if the background of the graph is clicked – and sends a dispatch to reset the node selection in global state. The last event "stabilized" is triggered automatically

```
options = {
  nodes: {
    shape: "box",
    borderWidth: 2,
  },
  edges: {width: 1},
  interaction: {hover: true},
  physics: {enabled: false},
};
```

Listing 4.5: The graph-options.

every time the graph is rendered, and ensures that the graph's zoom is adapted to the graph window.

```
events = {
  selectNode: (event) => {
    const nodeId = event.nodes[0];
    this.props.dispatch(actions.doFind(nodeId));
  },

  deselectNode: ()=>{this.props.dispatch(actions.doDeselect());},

  stabilized: () => {
    const {network} = this.state;

    if (!!network) {
      network.fit();
    }
  }
};
```

Listing 4.6: The graph-events.

4.3.2 State-Management with Redux

As mentioned in the previous subsection, the graphs are stored locally through the state in their respective classes. The data used to produce the graphs, on the other hand, are stored in the global state using Redux. The work of creating the *Redux Store* with full setup and then implementing it with React was already completed in the IDPT system. It was therefore only necessary to add *Actions*, *Reducers*, and *Selectors*, in the context of the artifact. To continue compliance with the SoC principle, one set with each of the three additions was created for each entity type (cases, modules, and tasks).

At a fundamental level, all three sets are equal, consisting of three different Actions: *doFind*, which requests all information for one particular node, *doDeselect*, which resets said node selection; and finally *doFetch*, which requests all units of the respective node entities, with a shortened amount of information. The greatest difference between them is that modules and tasks have the ability to filter the fetching based on a number of IDs.

4.3.3 GraphQL

What is relevant on the client-side of the artifact, in connection with GraphQL, is the actual production of queries. In addition to utilizing the previously implemented `find(id)` query, requesting all of the information related to the entity with the provided ID, a query function `graph(filter)` has been created for each of modules and tasks (see Listing 4.7), and a `graph()` for cases (see Listing 4.8). These functions request only the *rows* needed to create the graphs.

Note that the query for modules is the same as the one for tasks, except for line 11 being removed, line 12 being changed from “next” to “prerequisite”, and all instances of the word “task” are changed to “module”.

```
 1  static async graph(filter) {
 2    const response = await graphqlClient.query({
 3      query: gql`
 4        query TASK_GRAPH( $filter: TaskFilterInput ) {
 5          taskGraph( filter: $filter ) {
 6            count
 7            rows {
 8              id
 9              name
10              status
11              completionRequired
12              next {
13                id
14              }
15            }
16          }
17        }
18      `,
19      variables: { filter, },
20    });
21    return response.data.taskGraph;
22  }
```

Listing 4.7: The fetch-query for tasks.

```
 1  static async graph() {
 2    const response = await graphqlClient.query({
 3      query: gql`
 4        query CASED_GRAPH {
 5          casedGraph {
 6            count
 7            rows {
 8              id
 9              name
10              status
11            }
12          }
13        }
14      `,
15    });
16    return response.data.casedGraph;
17  }
```

Listing 4.8: The fetch-query for cases.

On the server-side of the artifact, a filter option was added to the tasks and modules to filter on IDs (plural). This had to be done in the GraphQL definitions and in the MongoDB repositories. In addition, separate GraphQL resolvers were created for each of the three entities, which were closely based on the resolvers for lists, but with minor modifications such as the removal of unnecessary parameters.

4.3.4 Overview Page

At the top level, the structure of the Overview page is composed of five elements. The page title and the collapse component that provide instructions to the user are the first two. The final three are components forming one new sub-page for each of the entities – `CasedGraphPage`, `ModuleGraphPage`, and `TaskGraphPage`. Furthermore, each of these components consists of its own subtitle before, in the case of module or task, a filter component, and finally the row consisting of two columns, each with its own card that holds a graph or `ViewContent`, respectively.

The first sub-page (`CasedGraphPage`) is mostly passive, apart from a simple check to find out if a node is selected and thus if it should be displayed in the View card (see Listing 4.9). In addition, it has a `componentDidMount` dispatching a `doFetch` action and a `componentWillUnmount` dispatching a `doDeselect` action. An identical *unmount* function and a similar check is performed on the other two sub-pages as well to deselect node at page redirect and to see if a node in the related graph is selected.

```
{!casedRecord ? <p>{i18n('overview.instructions.left')}</p> :
  <CasedViewContent
    loading={loading}
    record={casedRecord}
  />
}
```

Listing 4.9: Check if ViewContent should be displayed.

In addition to the above points, it is also examined on ModuleGraphPage whether a case node has been selected in order to either display a prompt to select a node or display the module content related to the selected case node. A similar logic is used on the last sub-page as well, where it is checked for selection of both case and module, as shown in Listing 4.10.

```
render() {
  const {casedRecord, moduleRecord} = this.props;

  if (!casedRecord) { return null; }
  if (!moduleRecord) {
    return <p>{i18n('overview.instructions.above')}</p>;
  }
  return this.renderTask();
}
```

Listing 4.10: The render-method for TaskGraphPage.

Regarding filters for modules and tasks, these are produced using the selected node in the previous graph, in accordance with Listing 4.11.

```
componentDidMount() {
  const {casedRecord, dispatch} = this.props;

  let modules = null;
  if (!!casedRecord) {
    modules = casedRecord.modules.map(m => m.id);
  }
  const ids = {ids: modules};

  dispatch(actions.doFetch(ids));
}
```

Listing 4.11: The filter for ModuleGraphPage.

Finally, it can be mentioned that the ViewContent components are an extracted part of a previous implementation, which then receives a prop from the calling parent, and thus consists of a component for the toolbar and a component for the node information, as seen in the last figures in Chapter 3.

Chapter 5

Evaluation

Through this chapter, it will be examined how well the resulting artifact meets the requirements described in Chapter 3. By the use of artificial data, the features of the artifact will go through experimental testing.

The overall objective for the artifact was to help reducing patient dropout rates for IDPT systems through facilitating dynamic assessment of psychological interventions. It is not possible, however, to test and evaluate the achievement of this objective during the work with this Master's thesis, as this would require comprehensive user testing with actual patients, which implies a far more considerable context than what is allowed by the given time frame. The evaluation process will instead focus on the more subordinate requirements that, through the hypothesis, were assumed to be able to meet the overall objective.

5.1 Design and Ease of Use

In Chapter 3, the requirements for the artifact were described as the ability to present the different cases, modules, and tasks within the treatment in the form of a distinct overview that can visualize the correlations between those treatment-elements, and then enable an interaction-based functionality allowing the user to view or edit the details of them. Moreover, some requirements were described for how this presentation should behave:

- The presentation should be able to handle instances of nonlinear interconnections, suggesting a network graph as a natural choice.
- The graph should be able to zoom in and out, as well as being moved around within its frame through user interactions.
- Regarding the node's coordinates, the requirements were described as weak hard-coding or built-in algorithm from the framework in question.
- The graph should enable manual movement of nodes through user interaction to ensure the possibility of enhanced readability in cases of overlapping nodes or edges.

- The node objects should be able to contain custom properties that do not affect how the graph framework interprets the node.

5.1.1 Top Level Graph Appearance

Through the previous two chapters, several figures have been provided showing that the implementation allows for nonlinear graphs, and since the possibility of nonlinear interconnections is included in network graphs by definition, it was deemed not to be necessary to test this requirement to any greater extent than what is already done through the implementation. As an additional argument to support this claim, a separate examination was conducted under *Implementing by definition*, in Section 4.3.

Regarding movement and options for enlarging and reducing graph size through zooming, this was a default setting offered by the Vis framework, which had to be turned off manually through configurations of the options parameter if this was not a desirable feature.

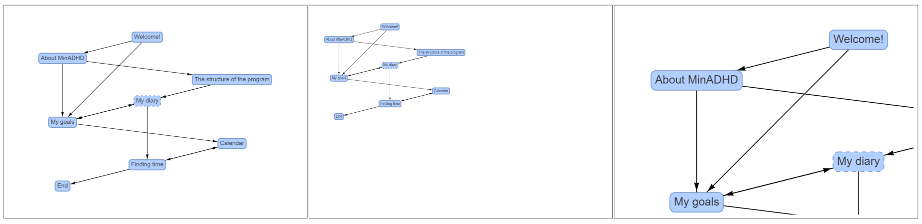


Figure 5.1: Illustrating the zoom and movement features.

As shown in Figure 5.1, these features were functioning at a satisfactory level by manual testing and experimentation. However, through this experimentation, a small bug was discovered, as illustrated by Figure 5.2, occurring in specific cases where a browser window was zoomed in to 200% or more. This bug was discovered using the following system properties and browsers:

- Operating system: Windows 10 Pro (build 19043.1083)
- Processor: Intel(R) Core(TM) i7-10510U
- GPU: NVIDIA GeForce MX250, and Intel(R) UHD Graphics
- Memory: 16,0 GB
- Browsers: Opera Version 77.0.4054.203 System:Windows 10 64-bit, and Microsoft Edge Version 91.0.864.67 (Official build) (64-bit)

Although this bug poses a disadvantage to the artifact, it was not considered a compromising factor for the overall result, as it did not occur when zooming the browser window below 200%.

The zoom and movement features were tested through both mouse interaction and touch screen, and the movement feature works as a drag-and-drop solution, while the zoom feature uses the scrolling wheel on the mouse or pinch-zooming on a touch screen.

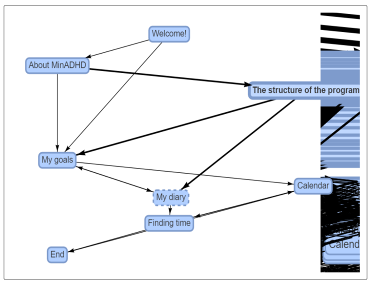


Figure 5.2: Illustrating a bug in the zoom feature.

5.1.2 Assignment of Coordinates

The Vis framework offers a built-in algorithm for node placements based on an implementation of artificial physics that can be adjusted using the options parameter. This algorithm was at first greatly appreciated. However, it was soon discovered that the algorithm was an iterative process based on a random initial placement of nodes, which led to each node being given a different position at each render.

When working with the same set of entities over long periods, and the visual presentation of the information looks different each time it is used, the concept works against its purpose. Hence, it became necessary to force a fixed initial position for each node and then set an upper limit for iterative steps so that the final result of the graph was the same at each render. This could be done by either determining a Seed and defining it in options or by giving each node its own set of coordinates. The choice fell on the latter, and a simple algorithm was implemented to assign an x- and a y-value to each node.

The leftmost graph in Figure 5.3 shows the result using the built-in algorithm combined with the simple support algorithm that was implemented. Although the result was the same for each render, it was not clean enough to satisfy the requirements. The support algorithm was therefore improved to the point that it could function as an independent solution, and the new algorithm was a simple counter-based process that placed each node in one of three columns, and with a given vertical distance between each node, as depicted with the rightmost graph in Figure 5.3.

The red edges in Figure 5.3 represent the traditional order for performing the tasks, emphasizing the design improvement from the original solution. Note that the red color is applied to the figure after the screenshot has been taken to emphasize the evaluation argument, and is not shown in the original graph.

The algorithm used in the final version of the artifact has one downside. This is shown in Figure 5.4: Since no nodes are placed on the same horizontal line, there is no overlap between nodes. However, in some cases, there may be overlap between edges that have the same direction, such as the edges between the three lower nodes in the figure. One possible solution to reduce this downside could be to implement a mathematically based coordinate assignment that ensures that no two nodes have a center located on the same horizontal or vertical line and that no pair of nodes has a directional shift with a common factor.

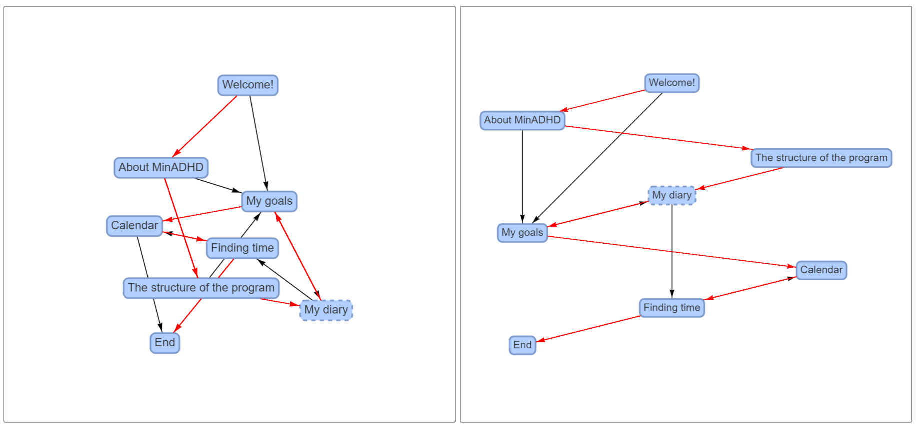


Figure 5.3: Two versions of an example task graph to illustrate the physics options. The graph on the left has the physics option set to true, while the one on the right is set to false. The red edges represent the traditional order for completing the tasks.

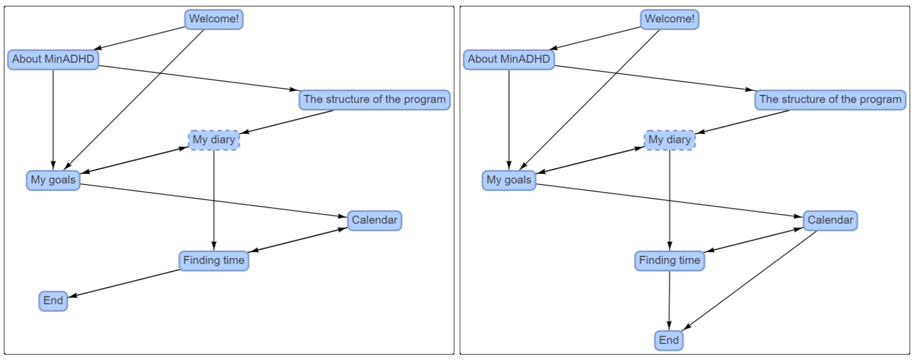


Figure 5.4: Overlapping edges.

To completely eliminate the possibility of overlap, such a mathematical solution would have to take into account several diagonals, in addition to the vertical and horizontal lines, and would thus be too extensive to make. Hence, since the Vis framework allows drag-and-drop of nodes by default, it was decided that the current solution was good enough for its purpose.

5.1.3 Data Storage

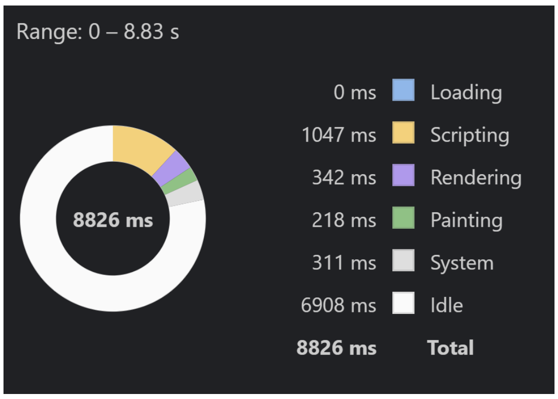
When deciding how data should be queried from the database and how it should be stored and used on the client-side, there are multiple possible solutions:

Request all data at once: Even in cases where there is not much information that can be retrieved from the database, this can involve a great deal of unnecessary storage on the client-side, particularly if there is not much user activity (i.e., the user only wants to see one module- or task graph). If there is a lot of information, however, unnecessary storage will be inescapable, and frequent data transfers can result in latency, high memory usage, and possibly reduced performance.

Request as limited information as possible: This will lead to far more queries against the database, and thus spend some time with each query. Nevertheless, the time that is actually spent is short enough that the user will likely not pay much attention to it in practice. Hence, the main disadvantage of this solution is that frequent queries will cause correspondingly more data transfers.

A hybrid solution: In cases where all of the information related to the selected node in the graph above is selected, this could lead to even greater use of time and data for frequent activity and thus appears to be a counterproductive solution to this artifact.

As there are no statistical measurements for how this IDPT framework will be used, it was not possible to know with certainty which solution would be the undeniably best alternative. It was therefore decided to request the absolute minimum of data needed to present new information at all times, as this is generally the recommended solution.



0 ms	🟦	Loading
1047 ms	🟨	Scripting
342 ms	🟪	Rendering
218 ms	🟩	Painting
311 ms	⬜	System
6908 ms	⬜	Idle
8826 ms		**Total**

Figure 5.5: Performance summary, created with Opera.

In connection with the final evaluation, a performance test was executed through the Opera browser specified earlier in this section. The user activity performed was as follows: Reload Overview page → select the case "My ADHD Program" → select the module "Week 1: Introduction" → select the task "Welcome!". After each node selection, the constraints tab was selected in the corresponding View column. The summary of the performance test is depicted in Figure 5.5.

5.1.4 Entity Manipulation

The overall design of the artifact follows that of the rest of the system. Hence, since the View component that makes up the right column of the Overview was already implemented, this was not changed to any great extent. The intention of adding this component to the Overview was to provide easier access to examine relevant information about the elements in the treatment programs so that necessary changes to the elements can be made without problems.

Despite the fact that it was not possible to carry out surveys with the domain experts, at the end of the work on the artifact, the intention to include the View component was considered to be fulfilled.

Chapter 6

Discussion

Through this chapter, an answer will be formulated to each of the research questions presented in Section 1.3, and provide a summary of the thesis' contribution to the domain knowledge. Finally, a brief overview of impressions that have arisen regarding methodology and technologies will be reflected.

6.1 Answers to Research Questions

RQ1: Given that there must be some form of artificial interconnection between the elements of a psychological intervention to enable reliable algorithms for dynamic assessment of these elements, how can it be ensured and clarified that all elements within an intervention are universally interconnected?

Within one intervention there may be a number of modules, and furthermore, each of these modules may contain multiple tasks. The overall idea is that all modules within a specific intervention should have some form of interconnection, and likewise, all tasks that are within the same module should have a corresponding form of connection. As described in Chapter 3, these relationships can be intuitively described as the edges of graphs where the elements that are connected (modules or tasks) form the nodes of the graph.

A natural step forward from this idea is to actually implement the intervention, with associated modules and their tasks, as a graph structure. As long as this graph structure can keep track of the link between the hierarchical levels, and thus which modules belong to which intervention, and further which tasks belong to which module, then the tasks at the lower level will be indirectly connected to the same level within other modules, through hierarchical links. This could open up for a number of alternative ways of detecting missing connections.

One possible solution is that a graph structure could enable algorithmic automation using, for example, Breadth-First Search (BFS) or Depth-First Search (DFS), to detect single nodes or cluster nodes that are separated from the rest of the graph, and thus be able to notify the user/therapist of potential disadvantages of such disconnections. Due to a large number of different diagnoses, and consequently an even greater number of possible interventions, this solution was withheld, as in some cases it is conceivable that “holes” in a graph are desirable, and thus that an aggressive warning algorithm could work against its purpose.

An alternative solution – the one that was chosen – is that once the graph structure is implemented, it can be taken one step further and also visualized. This solution does not exclude the possibility of adding automation as an additional aid but rather provides an additional possibility for the therapist to be able to detect missing connections manually.

As for the interventions, at the top level of the hierarchy, no link between them has been implemented. This is because, in the current IDPT system, there is no natural connection between different interventions. If this should nevertheless be a relevant addition, such an extension of the case graph can easily be carried out at a later occasion.

RQ2: In cases where the use of adaptive algorithms for the assessment of psychological interventions is facilitated, how can it be ensured that the elements within the interventions are presented and recommended in an appropriate order?

Essentially, the idea for answering this research question was that the order of the elements must be visualized in some form. In order for such a visualization of recommendations to work, some kind of strategy for adaptability is needed. A widely used strategy for this is ”rule-based adaptation”, and as described in Section 2.3, there are two ideas that are discussed for how to implement this in the context of the IDPT framework extended through this thesis:

1. The creation of a passive foundation for algorithms based on the assignment of constraints. This is possible to accomplish through the use of the taxonomies that are the product of a Master’s theses done in parallel with this one.
2. A user-based, dynamic algorithm built on user profiling. This option can build on the previous one by utilizing the constraints as inclusion criteria and exclusion criteria.

When the adaptive algorithm is implemented, it is, of course, possible to present the elements in an iCBT-based treatment program using a list representation where the elements are displayed in a priority-chronological order. However, without any extra features, this alone will not be able to take into account cases where two different steps further are equally recommended.

An alternative idea – the one chosen as the answer in this thesis – is that a graphical representation of the treatment program has the opportunity to visualize which elements are possible to proceed with after the previous element has been completed, even in cases of equally strong recommendations. When a working algorithm for dynamic assessment of psychological interventions is implemented, this graphical representation can be manipulated so that it highlights the edges of the graph that are most recommended. This solution will be able to visualize the degree of recommendation through details such as the thickness of the edges, the degree of line splitting (dotted, stippled, or solid lines), color codes, and so on, and will thus be able to offer a far greater degree of adjustments for visualization compared to a list representation.

RQ3: Building on the results from the previous two research questions, and given that a therapist will create a new iCBT-based treatment program in the IDPT framework, how can the process of creating and adapting this treatment program be simplified for the therapist?

The results from the two previous research questions involve a three-part graphical representation of interventions, the modules that are part of individual interventions, and tasks that belong to specific modules, respectively. This representation provides a clear overview of the structure of the treatment program regarding the content of the elements and how the elements are related to each other.

In the production of a new iCBT-based treatment program, this visual representation will be able to simplify the creation and adaption process by facilitating the detection of any missing relationships and other potential errors regarding informational details in the elements of the treatment program.

Furthermore, if an adaptive algorithm is implemented that uses inclusion and exclusion criteria that are linked to the elements of the treatment program, it may be essential to ensure that these criteria are correctly registered. To address this point, in connection with the artifact of this Master's thesis, a simple tab system has been included on the View page introducing the tab "Constraints" as an additional step towards making the implementation of dynamic assessment of psychological interventions easier (see Figure 3.9).

No specific constraints have been implemented to the View page at the moment, other than "CompletionRequired" for tasks and "Prerequisite" for modules, but the idea is that all new forms of constraints linked to a specific element will be collected under this tab, and thus deliver a clear overview of the criteria that apply to the individual element.

6.2 Contributions

Through Section 1.2 and Section 1.3, a problem description and a set of research questions were defined, describing relevant problems within Internet-Delivered Psychological Treatment. The artifact developed through this thesis is a result of the Design Science process that followed from these descriptions.

The artifact is a tool for graphically visualizing interventions in the IDPT framework to make it easier for the therapist to create and adjust informational details on and relationships between elements within iCBT-based treatment programs. In connection with the implementation of (particularly rule-based) adaptive strategies for dynamic assessment of such psychological interventions, it is a relevant challenge to ensure that the criteria to be used in algorithms for rule analysis are correct.

6.3 Reflection

A number of different elements have been included in the work on this Master's thesis. The research methodology Design Science was one of them, and through the helpful guidelines presented by Hevner et al. [26], this methodology has provided a defined structure for the overall process and the development of the artifact.

When it comes to utilized technologies, the overall impression has been positive:

- **ReactJS:** It is understandable why ReactJS is so popular. It offers easy customization of components and entails a great deal of usability regarding both changes to existing content and additions of new. The overall impression is that ReactJS delivers at a high level.
- **Vis:** Through the work on the artifact, there were divided opinions regarding Vis. The first impression was fantastic, but as the implementation became more extensive, it turned out that some of the built-in solutions were not as optimal as desired (e.g., algorithms for node placements and difficult readjustments of options parameters). The end result, on the other hand, gave a much better impression.
- **GraphQL:** In addition to being the query language used for the IDPT framework, GraphQL in itself serves as an architectural layout for the entire system. It took some time to get acquainted with this technology in the context of the system, as the implementation was at a different level than what is reviewed by GraphQL's own documentation. As soon as this challenge was overcome, this was considered to be a good choice of technology.
- **Redux:** In connection with the development of the artifact, Redux was already implemented. Any challenges with the fundamental implementation of this library are therefore uncertain when working with this thesis. The overall impression of Redux is thus exclusively positive, as it allows for an orderly implementation of state management.
- **Ant Design:** This technology is used for large parts of the design for both the IDPT framework and for the artifact itself. It offers countless solutions for both large and small design elements and seems to be a plug-and-play solution. In connection with the development of the artifact, there is nothing to complain about at Ant Design.
- **MongoDB and other technologies:** The database has hardly been touched during the development process. Hence there is not much to say about MongoDB. The same applies to the other technologies that have been utilized.

Chapter 7

Conclusions and Further Work

This thesis presents the design and implementation of an artifact for an open-source framework for Internet-Delivered Psychological Treatment. The artifact provides a graphical representation of iCBT-based treatment programs and serves as a digital tool to assist therapists in the process of creating and customizing these treatment programs within the framework by providing a comprehensive overview of their content and structure.

This final chapter will provide a summary of the process and results achieved through this thesis and, finally, propose alternatives for further research and work related to these results.

7.1 Conclusion

Using the research methodology Design Science, a digital artifact was designed, implemented, and finally evaluated with the overall research objective of facilitating dynamic assessment of psychological interventions. It was concluded that this facilitation could be achieved by ensuring that relevant criteria that can be used by rule-based adaptive strategies can be correctly recorded in elements within an iCBT-based treatment program and thus provide a foundation for adaptive algorithms to be implemented.

Through the design and implementation process, a set of guidelines provided by Hevner et al. [26] was followed and, using existing theories, a solution was developed based on a defined problem description and research questions. The resulting artifact uses a network graph that follows the definition of a Directed multi-graph to provide an overview of the elements within a psychological intervention. The network graph is built with a three-part hierarchical structure where each level consists of a component that represents interventions, modules, and tasks, respectively. Furthermore, each component consists of two parts – an interactive graph that visualizes the relationships between the elements belonging to the selected node in the graph above and a display component that provides detailed information about the selected node at the same level in the hierarchy.

7.2 Further Work

As the IDPT framework in question still is an early-stage prototype, at the time of writing this thesis, there are some elements in the framework that have not been included in the artifact. One example of such is "Assignments", which is intended to be a sub-element as part of tasks. Once this is implemented in full, it would be beneficial to include assignments in the Overview page provided by the artifact.

As soon as the user profiling and taxonomies mentioned under the answer to RQ2, in Section 6.1, have been implemented, and the process of implementing adaptive strategies can begin, multiple other proposals await:

- Consider whether the algorithm for assigning coordinates to the nodes of the graphs should be extended to take into account recommended orders for the execution of the elements within the treatment program.
- Consider whether the entity status should trigger the "hidden"-property of the nodes of the graphs so that colors can instead represent recommended paths.

Bibliography

[1] Gerhard Andersson et al. “Internet-delivered psychological treatments: from innovation to implementation.” eng. In: *World psychiatry: official journal of the World Psychiatric Association (WPA)* 18.1 (Feb. 2019), pp. 20–28. ISSN: 1723-8617. DOI: 10.1002/wps.20610.

[2] Antd. *Ant Design - The world’s second most popular React UI framework*. en. URL: https://ant.design/ (visited on July 8, 2021).

[3] American Psychological Association. *What Is Cognitive Behavioral Therapy?* en. July 2017. URL: https://www.apa.org/ptsd-guideline/patients-and-families/cognitive-behavioral (visited on Feb. 28, 2021).

[4] Sabrina Barr. *Six ways social media negatively affects your mental health without you even knowing*. en. Section: Lifestyle. Oct. 2020. URL: https://www.independent.co.uk/life-style/health-and-families/social-media-mental-health-negative-effects-depression-anxiety-addiction-memory-a8307196.html (visited on Feb. 22, 2021).

[5] BeatingTheBlues. *CCBT*. en-GB. URL: https://www.beatingtheblues.co.uk/ccbt/ (visited on Mar. 11, 2021).

[6] BeatingTheBlues. *How does CBT work Online — Mental Health Support — BTB*. en-GB. URL: https://www.beatingtheblues.co.uk/how-it-works/ (visited on Mar. 11, 2021).

[7] Peter Brusilovsky and Julita Vassileva. “Course sequencing techniques for large-scale web-based education.” In: *International Journal of Continuing Engineering Education and Life Long Learning* 13.1-2 (Jan. 2003). Publisher: Inderscience Publishers, pp. 75–94. ISSN: 1560-4624. DOI: 10.1504/IJCEELL.2003.002154. URL: https://www.inderscienceonline.com/doi/abs/10.1504/IJCEELL.2003.002154 (visited on Mar. 28, 2021).

[8] Daniel Caldas. *danielcaldas/react-d3-graph*. original-date: 2017-03-29T21:07:12Z. July 2021. URL: https://github.com/danielcaldas/react-d3-graph (visited on July 2, 2021).

[9] Giuseppe Campanelli. *Openbase — The Yelp of Open-Source Packages*. en. Jan. 2021. URL: https://javascript.plainenglish.io/openbase-the-yelp-of-open-source-packages-e638f95239cf (visited on July 1, 2021).

[10] Gregory N. Clarke et al. “Cognitive-Behavioral Treatment of Adolescent Depression: Efficacy of Acute Group Treatment and Booster Sessions.” en. In: *Journal of the American Academy of Child & Adolescent Psychiatry* 38.3 (Mar. 1999), pp. 272–279. ISSN: 0890-8567. DOI: 10.1097/00004583-

199903000-00014. URL: https://www.sciencedirect.com/science/article/pii/S0890856709629221 (visited on Apr. 4, 2021).

[11] Justin Coberly. *justin-coberly/dagre-d3-react*. original-date: 2019-09-10T01:28:36Z. May 2021. URL: https://github.com/justin-coberly/dagre-d3-react (visited on July 2, 2021).

[12] Rhys Edmonds. *Anxiety, loneliness and Fear of Missing Out: The impact of social media on young people's mental health — Centre for Mental Health*. URL: https://www.centreformentalhealth.org.uk/blogs/anxiety-loneliness-and-fear-missing-out-impact-social-media-young-peoples-mental-health (visited on Feb. 22, 2021).

[13] Facebook. *React – A JavaScript library for building user interfaces*. en. URL: https://reactjs.org/ (visited on July 7, 2021).

[14] Kristina Fenn and Majella Byrne. "The key principles of cognitive behavioural therapy." en. In: *InnovAiT* 6.9 (Sept. 2013). Publisher: SAGE Publications, pp. 579–585. ISSN: 1755-7380. DOI: 10.1177/1755738012471029. URL: https://doi.org/10.1177/1755738012471029 (visited on Feb. 28, 2021).

[15] Folkehelseinstituttet. *Psykisk helse i Norge*. no. Feb. 2018. URL: https://www.fhi.no/publ/2018/psykisk-helse-i-norge/ (visited on Feb. 23, 2021).

[16] Folkehelseinstituttet. *Quality of life and mental health among children and adolescents*. en. Aug. 2019. URL: https://www.fhi.no/en/op/hin/groups/mental-health-children-adolescents/ (visited on Feb. 23, 2021).

[17] Folkehelseinstituttet. *Selvmord og selvmordsforsøk - Folkehelserapporten*. no. May 2020. URL: https://www.fhi.no/nettpub/hin/psykisk-helse/selvmord-i-norge/ (visited on Jan. 30, 2021).

[18] Manaswi Gautam et al. "Cognitive Behavioral Therapy for Depression." In: *Indian Journal of Psychiatry* 62.Suppl 2 (Jan. 2020), S223–S229. ISSN: 0019-5545. DOI: 10.4103/psychiatry.IndianJPsychiatry_772_19. URL: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC7001356/ (visited on Mar. 8, 2021).

[19] Simon Gilbody et al. "Computerised cognitive behaviour therapy (cCBT) as treatment for depression in primary care (REEACT trial): large scale pragmatic randomised controlled trial." eng. In: *BMJ (Clinical research ed.)* 351 (Nov. 2015), h5627. ISSN: 1756-1833. DOI: 10.1136/bmj.h5627.

[20] GraphQL. *Architecture & Big Picture*. en. URL: https://www.howtographql.com/basics/3-big-picture/ (visited on July 10, 2021).

[21] GraphQL. *GraphQL — A query language for your API*. URL: https://graphql.org/ (visited on July 7, 2021).

[22] Erik Hedman et al. "Cost-effectiveness of Internet-based cognitive behavior therapy vs. cognitive behavioral group therapy for social anxiety disorder: Results from a randomized controlled trial." en. In: *Behaviour Research and Therapy* 49.11 (Nov. 2011), pp. 729–736. ISSN: 0005-7967. DOI: 10.1016/j.brat.2011.07.009. URL: https://www.sciencedirect.com/science/article/pii/S0005796711001586 (visited on Mar. 8, 2021).

[23] Erik Hedman et al. "Effectiveness of Internet-based cognitive behaviour therapy for depression in routine psychiatric care." en. In: *Journal of Affective Disorders* 155 (Feb. 2014), pp. 49–58. ISSN: 0165-0327. DOI: 10.

1016/j.jad.2013.10.023. URL: https://www.sciencedirect.com/science/article/pii/S0165032713007659 (visited on Apr. 5, 2021).

[24] Martin Heitmann. “Development of a mobile artifact to support adaptive iCBT using multi modality support and usage data.” en, nob. Accepted: 2021-01-20 Publisher: The University of Bergen. MA thesis. Bergen, Norway: Western Norway University of Applied Sciences, and University of Bergen, Jan. 2021. URL: https://bora.uib.no/bora-xmlui/handle/11250/2723790 (visited on Mar. 13, 2021).

[25] Helsedirektoratet. *Psykisk helse for voksne - ventetid.* no. Mar. 2020. URL: https://www.helsedirektoratet.no/statistikk/kvalitetsindikatorer/psykisk-helse-for-voksne/gjennomsnittlig-ventetid-for-voksne-i-psykisk-helsevern (visited on Jan. 30, 2021).

[26] Alan R. Hevner et al. “Design Science in Information Systems Research.” In: *MIS Quarterly* 28.1 (2004). Publisher: Management Information Systems Research Center, University of Minnesota, pp. 75–105. ISSN: 0276-7783. DOI: 10.2307/25148625. URL: https://www.jstor.org/stable/25148625 (visited on June 26, 2021).

[27] Stefan G. Hofmann et al. “The Efficacy of Cognitive Behavioral Therapy: A Review of Meta-analyses.” en. In: *Cognitive Therapy and Research* 36.5 (Oct. 2012), pp. 427–440. ISSN: 1573-2819. DOI: 10.1007/s10608-012-9476-1. URL: https://doi.org/10.1007/s10608-012-9476-1 (visited on Apr. 4, 2021).

[28] John Paul Jameson and Michael B. Blank. “The role of clinical psychology in rural mental health services: Defining problems and developing solutions.” In: *Clinical Psychology: Science and Practice* 14.3 (2007). Place: United Kingdom Publisher: Blackwell Publishing, pp. 283–298. ISSN: 1468-2850(Electronic),0969-5893(Print). DOI: 10.1111/j.1468-2850.2007.00089.x.

[29] Constantino M. Lagoa et al. “Designing adaptive intensive interventions using methods from engineering.” In: *Journal of Consulting and Clinical Psychology* 82.5 (2014). Place: US Publisher: American Psychological Association, pp. 868–878. ISSN: 1939-2117(Electronic),0022-006X(Print). DOI: 10.1037/a0037736.

[30] Vincent Lecrubier. *crubier/react-graph-vis.* original-date: 2015-04-09T08:56:41Z. June 2021. URL: https://github.com/crubier/react-graph-vis (visited on July 2, 2021).

[31] Centre for Mental Health. *Mental health services unable to cope with demand for psychological therapies, says coalition — Centre for Mental Health.* URL: https://www.centreformentalhealth.org.uk/news/mental-health-services-unable-cope-demand-psychological-therapies-says-coalition (visited on Feb. 25, 2021).

[32] MongoDB. *The most popular database for modern apps.* en-us. URL: https://www.mongodb.com (visited on July 8, 2021).

[33] María Lucila Morales-Rodríguez et al. “Architecture for an Intelligent Tutoring System that Considers Learning Styles.” en. In: *Research in Computing Science* 47.1 (Dec. 2012), pp. 37–47. ISSN: 1870-4069. DOI: 10.13053/rcs-47-1-4. URL: http://rcs.cic.ipn.mx/2012_47/Architecture%20for%20an%20Intelligent%20Tutoring%20System%20that%20Considers%20Learning%20Styles.pdf (visited on Mar. 23, 2021).

[34] Suresh Kumar Mukhiya. *Architecture.* June 2021. URL: https://idpt.gitbook.io/web/architecture (visited on June 29, 2021).

[35] Suresh Kumar Mukhiya et al. "Adaptive Elements in Internet-Delivered Psychological Treatment Systems: Systematic Review." en. In: *Journal of Medical Internet Research* 22.11 (2020). Company: Journal of Medical Internet Research Distributor: Journal of Medical Internet Research Institution: Journal of Medical Internet Research Label: Journal of Medical Internet Research Publisher: JMIR Publications Inc., Toronto, Canada, e21066. DOI: 10.2196/21066. URL: https://www.jmir.org/2020/11/e21066/ (visited on Feb. 25, 2021).

[36] Suresh Kumar Mukhiya et al. "Adaptive Systems for Internet-Delivered Psychological Treatments." In: *IEEE Access* 8 (2020). Conference Name: IEEE Access, pp. 112220–112236. ISSN: 2169-3536. DOI: 10.1109/ACCESS.2020.3002793.

[37] Inbal Nahum-Shani et al. "Q-learning: A data analysis method for constructing adaptive interventions." In: *Psychological Methods* 17.4 (2012). Place: US Publisher: American Psychological Association, pp. 478–494. ISSN: 1939-1463(Electronic),1082-989X(Print). DOI: 10.1037/a0029373.

[38] Openbase. *About Openbase.* en. URL: https://openbase.com/about (visited on July 1, 2021).

[39] Openbase. *dagre-d3-react: Docs, Tutorials, Reviews — Openbase.* URL: https://openbase.com/js/dagre-d3-react (visited on July 2, 2021).

[40] World Health Organization. *Suicide.* en. Feb. 2019. URL: https://www.who.int/news-room/fact-sheets/detail/suicide (visited on Feb. 23, 2021).

[41] J. Proudfoot et al. "Computerized, interactive, multimedia cognitive-behavioural program for anxiety and depression in general practice." eng. In: *Psychological Medicine* 33.2 (Feb. 2003), pp. 217–227. ISSN: 0033-2917. DOI: 10.1017/s0033291702007225.

[42] Royal College of Psychiatrists. *Two-fifths of patients waiting for mental health treatment forced to resort to emergency or crisis services.* en. Oct. 2020. URL: https://www.rcpsych.ac.uk/news-and-features/latest-news/detail/2020/10/06/two-fifths-of-patients-waiting-for-mental-health-treatment-forced-to-resort-to-emergency-or-crisis-services (visited on Feb. 25, 2021).

[43] *react-d3-graph.* URL: https://danielcaldas.github.io/react-d3-graph/sandbox/index.html (visited on June 30, 2021).

[44] *React-graph-vis.* URL: http://crubier.github.io/react-graph-vis/ (visited on June 30, 2021).

[45] MIRECC Education Grant Recipients, Jeffrey A Cully, and Andra L Teten. "A Therapist's Guide to Brief Cognitive Behavioral Therapy." en. In: (2008), p. 111.

[46] Redux. *Redux - A predictable state container for JavaScript apps. — Redux.* en. URL: https://redux.js.org/ (visited on July 7, 2021).

[47] Miguel Ricou et al. "Psychological intervention at a primary health care center: predictors of success." In: *BMC Family Practice* 20.1 (Aug. 2019). tex.ids= ricouPsychologicalInterventionPrimary2019, p. 116. ISSN: 1471-2296. DOI: 10.1186/s12875-019-1005-9. URL: https://doi.org/10.1186/s12875-019-1005-9 (visited on Jan. 30, 2021).

[48] Spurti Prashant Salimath. “Building a flexible CBT model based on structured data for the COPE app.” eng. Accepted: 2021-01-20T00:52:07Z Publisher: The University of Bergen. MA thesis. Bergen, Norway: Western Norway University of Applied Sciences, and University of Bergen, Jan. 2021. URL: https://bora.uib.no/bora-xmlui/handle/11250/2723792 (visited on Mar. 21, 2021).

[49] Paul Sawers. *Openbase wants to be the Yelp for open source software packages.* en-US. Jan. 2021. URL: https://venturebeat.com/2021/01/13/openbase-helps-developers-find-and-compare-open-source-packages/ (visited on July 1, 2021).

[50] Iony D. Schmidt, Nicholas R. Forand, and Daniel R. Strunk. “Predictors of Dropout in Internet-Based Cognitive Behavioral Therapy for Depression.” en. In: *Cognitive Therapy and Research* 43.3 (June 2019), pp. 620–630. ISSN: 1573-2819. DOI: 10.1007/s10608-018-9979-5. URL: https://doi.org/10.1007/s10608-018-9979-5 (visited on Feb. 24, 2021).

[51] Laura E. Sockol. “A systematic review of the efficacy of cognitive behavioral therapy for treating and preventing perinatal depression.” en. In: *Journal of Affective Disorders* 177 (May 2015), pp. 7–21. ISSN: 0165-0327. DOI: 10.1016/j.jad.2015.01.052. URL: https://www.sciencedirect.com/science/article/pii/S0165032715000658 (visited on Apr. 4, 2021).

[52] Ailbhe Spillane et al. “What are the physical and psychological health effects of suicide bereavement on family members? An observational and interview mixed-methods study in Ireland.” In: *BMJ Open* 8.1 (Jan. 2018). ISSN: 2044-6055. DOI: 10.1136/bmjopen-2017-019472. URL: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC5781012/ (visited on Feb. 21, 2021).

[53] StackOverflow. *Stack Overflow Developer Survey 2020.* 2020. URL: https://insights.stackoverflow.com/survey/2020/?utm_source=social-share&utm_medium=social&utm_campaign=dev-survey-2020 (visited on July 7, 2021).

[54] styled-components. *styled-components.* en. URL: https://www.styled-components.com (visited on July 8, 2021).

[55] Uber. *uber/react-vis-force.* original-date: 2016-10-03T22:53:52Z. June 2021. URL: https://github.com/uber/react-vis-force (visited on July 2, 2021).

[56] Vis. *vis.js.* URL: https://visjs.org/ (visited on July 7, 2021).

[57] Milton L. Wainberg et al. “Challenges and Opportunities in Global Mental Health: a Research-to-Practice Perspective.” eng. In: *Current Psychiatry Reports* 19.5 (May 2017), p. 28. ISSN: 1535-1645. DOI: 10.1007/s11920-017-0780-z.

[58] Christian A. Webb, Isabelle M. Rosso, and Scott L. Rauch. “Internet-based Cognitive Behavioral Therapy for Depression: Current Progress & Future Directions.” In: *Harvard review of psychiatry* 25.3 (2017), pp. 114–122. ISSN: 1067-3229. DOI: 10.1097/HRP.0000000000000139. URL: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC5421393/ (visited on Mar. 9, 2021).

[59] Uwe Wolter. “Category Theory and Diagrammatic Modelling.” en. In: *Script for the course INF 223, Spring 2020* (Mar. 2020), p. 195.

Appendix A

Source code

The source code for the open-source IDPT framework extended with the artifact covered by this thesis is available at the following URL:

`https://github.com/sureshHARDIYA/idpt`.

Here, all of the work related to the development of the artifact is performed under the branch "dapi".