VILNIUS UNIVERSITY
FACULTY OF MATHEMATICS AND INFORMATICS
SOFTWARE ENGINEERING STUDY PROGRAMME

Adapting Niklas Luhmann's Card Indexing System for Software Documentation

Niklaso Liūmano kortelių indeksavimo sistemos pritaikymas programinės įrangos dokumentacijai

Bachelor's thesis

Author:	4th year, 3rd group student Greta Žemgulytė	(signature)
Supervisor:	Gediminas Rimša	(signature)
Reviewer:	Vaidas Jusevičius	(signature)

Vilnius – 2023

CONTENTS

Abstract

The objective of this thesis is to explore the application of Niklas Luhmann's card indexing system and the closely-related Zettelkasten note-taking method for software documentation and to assess its effectiveness compared to existing methods. The work is divided into four main sections, which in turn focus on the exploration, adaptation, utilization, and evaluation of the Zettelkasten method.

The thesis explores how the Zettelkasten method was originally used by its creator, and explains the principles and processes of how to create the collection of notes called a Zettelkasten. It is examined and recommendations are made on how the Zettelkasten method can be modified and used to organize and manage software-related knowledge. The thesis demonstrates the practical application of the Zettelkasten approach by organizing the existing documentation of a software system. The method's effectiveness is evaluated based on the author's personal experience and the experience of other software developers. The Zettelkasten method is compared with other documentation approaches theoretically and through the use of a survey.

The results indicate that the adapted Zettelkasten method is comparable to using a wiki in terms of effectiveness for software documentation. The method is deemed beneficial for teams that follow Agile principles or use a code-first approach for software development. However, using the method requires more initial time investment and training. Based on the research findings, it can be concluded that the Zettelkasten method holds promise as an alternative approach to software documentation.

Overall, this thesis contributes to the field of software documentation by exploring the adaptation and application of the Zettelkasten method. It highlights the advantages and challenges of using this method, provides practical guidelines for its implementation, and offers insights into its effectiveness compared to existing methods. The findings of this research can serve as a valuable resource for software development teams seeking to improve their documentation practices and enhance knowledge management within their organizations.

Keywords: Zettelkasten method, software documentation, Niklas Luhmann

Santrauka

Šio darbo tikslas – ištirti Niklaso Liūmano kortelių indeksavimo sistemos ir su ja glaudžiai susijusio Zettelkasten metodo taikymą programinės įrangos dokumentacijai bei įvertinti metodo veiksmingumą lyginant su esamais metodais. Darbas suskirstytas į keturis pagrindinius skyrius, kurie atitinkamai skirti Zettelkasten metodo tyrinėjimui, pritaikymui, naudojimui ir vertinimui.

Darbe nagrinėjama, kaip Zettelkasten metodą naudojo jo kūrėjas, ir paaiškinami užrašų rinkinio, vadinamo Zettelkasten, kūrimo principai ir procesai. Analizuojama bei pateikiamos rekomendacijos, kaip modifikuoti ir naudoti Zettelkasten metodą su programine įranga susijusioms žinioms organizuoti ir valdyti. Darbe demonstruojamas praktinis Zettelkasten metodo taikymas organizuojant esamą programinės įrangos sistemos dokumentaciją. Zettelkasten metodo efektyvumas vertinamas remiantis asmenine autoriaus ir kitų programinės įrangos kūrėjų patirtimi. Metodas lyginamas su kitais dokumentavimo metodais teoriškai ir apklausos pagalba.

Rezultatai rodo, kad nėra reikšmingo skirtumo tarp Zettelkasten ar wiki metodų efektyvumo programinės įrangos dokumentavimui. Zettelkasten metodas laikomas naudingu komandoms, kurios laikosi Agile principų arba naudoja "kodo pirmumo" metodiką programinės įrangos kūrimui. Vis dėlto Zettelkasten naudojimas reikalauja pradinių laiko investicijų ir apmokymų. Remiantis tyrimo rezultatais, galima daryti išvadą, kad Zettelkasten metodas yra perspektyvi alternatyva programinės įrangos dokumentavimui.

Bendrai šis darbas prisideda prie programinės įrangos dokumentacijos srities, nes jame nagrinėjamas Zettelkasten metodo pritaikymas ir naudojimas. Jame išryškinami šio metodo taikymo privalumai ir iššūkiai, pateikiamos praktinės jo taikymo gairės ir įžvalgos apie jo veiksmingumą, palyginus su esamais metodais. Šio tyrimo rezultatai gali būti vertingas šaltinis programinės įrangos kūrimo komandoms, siekiančioms tobulinti savo dokumentacijos rengimo praktikas ir gerinti organizacijos žinių valdymą.

Raktiniai žodžiai: Zettelkasten metodas, programinės įrangos dokumentacija, Niklasas Liūmanas

Introduction

Niklas Luhmann was a German philosopher and sociologist who published more than 70 books and nearly 400 scholarly articles [Rot13]. To help with his writing, N. Luhmann used a note-taking method of his own creation. Niklas Luhmann's method, known as the Zettelkasten method, is a system of organizing notes and ideas using index cards. It has been widely used in various fields such as research, writing, and knowledge management. Adapting this system for software documentation could be a useful approach for organizing and documenting software-related knowledge.

The Zettelkasten method is based on the idea of building a personal knowledge management system that is flexible, scalable, and adaptable. In this system, each index card represents a discrete idea, topic, or concept, and is linked to other cards through a system of cross-referencing. This creates a collection of cards called a Zettelkasten – a network of ideas and concepts that can be easily navigated and organized [Sch18]. The links between index cards are created by using N. Luhmann's card indexing system that allows for inserting new indexes in between existing ones without breaking the existing order or changing any of the old indexes [Ahr17]. For the purpose of this paper, it should be noted that N. Luhmann's card indexing system and the Zettelkasten method are closely intertwined, so adapting the indexing system means adapting the note-taking method as well.

The Zettelkasten method can be particularly relevant for software documentation since it has several similarities with real-life software systems:

- Modularity: Software systems are often modular, consisting of different components or modules that work together to achieve a particular goal [Pre10]. Similarly, a Zettelkasten consists of multiple Zettels that can be organized to represent different parts of the software system.
- Links between components: In a software system, components often interact with each other, and changes to one component can have ripple effects on other components [Pre10]. Similarly, in a Zettelkasten, notes can be linked together to represent relationships between different components or concepts.
- Flexible structure: In software development, the complexity of a system can often make it difficult to create a clear and concise representation of its structure. While some systems may have a clear hierarchical structure, many others do not. For example, a system that is divided into multiple layers or components may have multiple possible ways to visualize the relationships between them [CBB10]. In such cases, the Zettelkasten method can be used to create a non-hierarchical structure of interconnected notes that reflects the complexity of the system.
- Iterative development: Software systems are often developed iteratively, with new features or functions added over time [Mar14]. Similarly, a Zettelkasten can be updated and refined over time to reflect new insights or changes to the software system.

Adapting Niklas Luhmann's card indexing system for software documentation could be a useful approach for organizing and documenting software-related knowledge, particularly for tasks such as domain modeling. Domain modeling is the process of creating a conceptual model of a particular problem domain, which can help developers understand the domain better and design better software solutions [MT15]. By using the Zettelkasten method, developers could create a network of linked ideas and concepts related to the domain, which would be easy to navigate. This approach could help developers build a more robust understanding of the problem domain and design better software solutions.

In addition to domain modeling, the Zettelkasten method could also be used for other software documentation tasks such as documenting design decisions, capturing requirements, and maintaining a knowledge base. By using a system of cross-referencing, developers could quickly locate related information, understand the context of a particular idea or concept, and ensure that their documentation is accurate and up-to-date.

Overall, the Zettelkasten method could be used to represent a software system in a modular, hierarchical, and iterative way, reflecting the structure and development process of the system. By breaking down complex concepts into smaller, more digestible notes and linking related notes together, a Zettelkasten could help developers and stakeholders better understand and manage a software system.

Object of research

The object of research for this thesis is the application of the Zettelkasten method for software documentation. The author will explore how the Zettelkasten method can be adapted to improve software documentation practices, specifically in tasks such as domain modeling, documenting design decisions, capturing requirements, and maintaining a knowledge base. The object of research would encompass the methodology, tools, and techniques used to implement the Zettelkasten method for software documentation and how it can be applied to improve software development practices.

Goal and objectives

The goal of this paper is to adapt and use Niklas Luhmann's card indexing system for software documentation.

The objectives leading to this goal are:

1. Completely adapt the Zettelkasten method for software documentation.
2. Use the Zettelkasten method to organize the documentation of an existing system.
3. Conduct a survey to compare the Zettelkasten method with existing methods of software documentation.

The chosen objectives are aligned with the overall goal of adapting and using Niklas Luhmann's card indexing system for software documentation. The first objective is to completely adapt the Zettelkasten method for software documentation, which means taking the original system and making necessary changes to make it work effectively for software documentation while fully embracing and applying the principles and practices of the original method. This is an essential step to ensure that the method can be successfully used in a software development environment.

The second objective is to use the adapted Zettelkasten method to organize the documentation of an existing system. This objective is important as it allows the practical application of the adapted Zettelkasten method to a real-world scenario. It provides an opportunity to test and refine the method and identify any potential issues or areas for improvement.

The final objective is to conduct a survey to compare the Zettelkasten method with existing methods of software documentation. This objective is crucial in evaluating the effectiveness of using the Zettelkasten method for software documentation. Comparing it to other methods allows for an objective assessment of its strengths and weaknesses, and helps identify any areas where it may outperform other methods. The survey provides a means to gather feedback from developers who have used different documentation methods and compare their experiences with using the Zettelkasten method.

Overall, the chosen objectives are important in achieving the goal of adapting and using Niklas Luhmann's card indexing system for software documentation and evaluating its effectiveness in comparison to existing methods.

1. Exploring the Zettelkasten method

To adapt Niklas Luhmann's card indexing system for software documentation it is necessary to fully understand the system itself. Niklas Luhmann used this system throughout his career as a sociologist. Originally, Luhmann used index cards, which he called ”slips” or ”Zettels”, as the primary medium for recording his notes. Each Zettel contained a single idea or concept, and he wrote on one side of the card only. Every Zettel was assigned a unique number, which he used to keep track of its place in the system. N. Luhmann kept the Zettels in physical slip-boxes, which allowed him to easily sort and categorize them as he added new ideas. He would periodically review and rearrange the Zettels to ensure that they were organized in a way that made sense to him. N. Luhmann also used cross-referencing to link related ideas across different Zettels. When he encountered a new idea that related to an existing Zettel, he would note the number of the old Zettel on the new one, and vice versa. When he needed to retrieve information from this system, N. Luhmann would use the indexes to quickly locate the relevant Zettels. He could also use the cross-referencing system to find related ideas and concepts. Overall, the Zettelkasten method works by building an interconnected system of literature references and thoughts. The Zettelkasten works as a ”second brain”, where ideas are stored and linked, which can lead to the development of new ideas [Ahr17].

1.1. Note-taking process

Filling up a Zettelkasten involves a specific note-taking process. The classic Zettelkasten method consists of three primary kinds of notes [Kad21]:

- Fleeting note – quick note that you take on the go.
- Literature note – a Zettel containing a brief summary of an entire book, article, or other written material.
- Permanent note – a Zettel containing a single succinct idea. It also includes references to other Zettels.

The process to fill up the Zettelkasten involves persistently capturing, refining, assigning a unique identifier, creating links, and storing Zettels in a slip-box [Ahr17]:

1. Capturing notes: The process starts with capturing any ideas or information on individual fleeting notes. This could be anything from a literature quote to an observation.
2. Refining notes: Once the notes are captured, they are reviewed and refined into either literature notes or permanent notes. This could involve clarifying thoughts, summarizing ideas, or organizing them by topic. Both types of Zettels would originally be stored in separate boxes.
3. Assigning a unique identifier: Each Zettel in the slip-box should have a unique identifier, such as a number or a keyword. This identifier helps locate specific Zettels later on and makes it easier to link related ones together.
4. Creating links: Connections between Zettels are found and links are created. This could

involve linking them based on common themes or ideas or linking together Zettels that provide evidence for a particular argument or idea.

5. Storing Zettels in a slip-box: Zettels are stored in the slip-box in a way that makes sense to their writer. This could involve organizing Zettels by topic or by date, or grouping related ones together.
6. Review and revise: Zettels have to be regularly reviewed and revised to keep them up-to-date and relevant. This could involve adding new Zettels, revising existing ones, or reorganizing the Zettelkasten to reflect changes in thinking.

1.2. Indexing system

Niklas Luhmann had a unique way of indexing Zettels that differed from the traditional topical organization. Rather than categorizing them by subject matter, he assigned them fixed numbers that served only as a means of identification.

This indexing system, also called the Folgezettel (German for “follow-up slip”) method, was particularly effective in organizing Zettels in a paper-based Zettelkasten, making them easier to locate. When N. Luhmann created a new Zettel that was relevant to an existing one, such as a comment or correction, he added it directly after the referenced Zettel. If a Zettel was related to multiple Zettels, it would be placed after any one of them, and then linked to the remaining Zettels by referencing their indexes. This way N. Luhmann could navigate through the Zettels either by examining adjacent ones or by following links to other Zettels, which enabled him to find related ideas quickly [Kad21].

In N. Luhmann’s indexing system, each card is assigned a number that corresponds to its unchanging position in the slip-box. For instance, the first note in the first section of the collection is designated as card ”1/1”. Subsequently, card ”1/2” follows, and so on. When a new card is created to explore a related aspect that extends beyond the content covered by card ”1/1” and card ”1/2”, it is assigned the number ”1/1a”, inserting it between card ”1/1” and ”1/2”. If further breakdown or exploration is required, additional cards like ”1/1b” or ”1/1a1” can be inserted accordingly. An example of indexed cards can be seen in Figure 1. This sequential numbering system facilitated the systematic organization and retrieval of information within the system [Sch18].

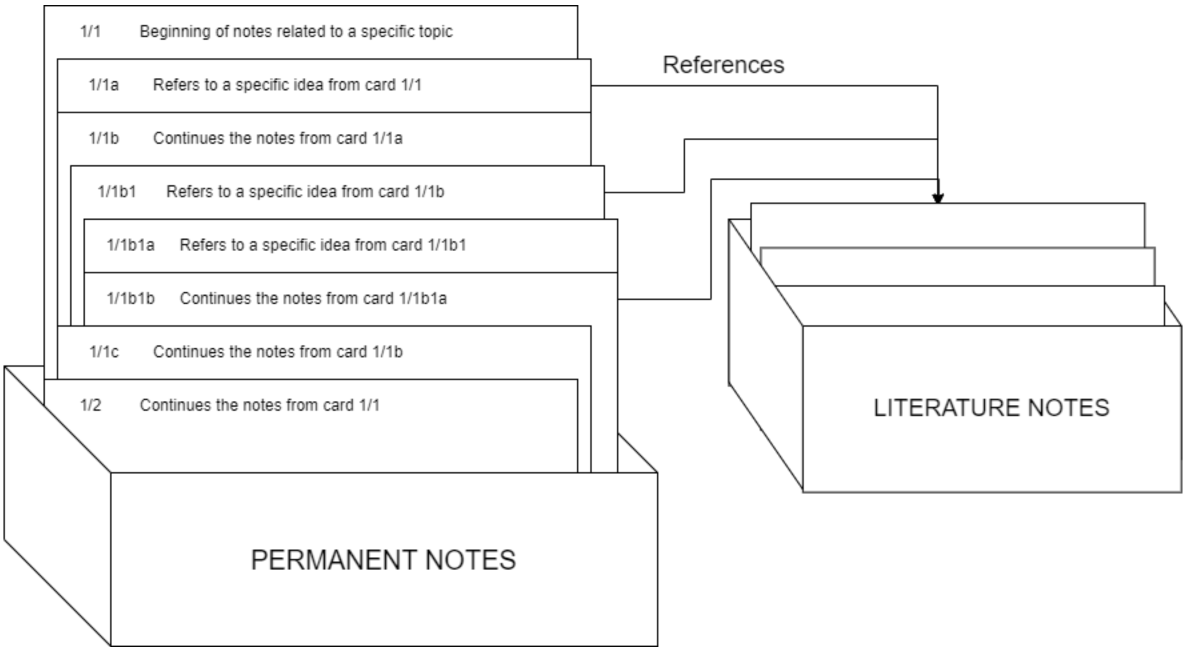

1/1 Beginning of notes related to a specific topic
1/1a Refers to a specific idea from card 1/1
1/1b Continues the notes from card 1/1a
1/1b1 Refers to a specific idea from card 1/1b
1/1b1a Refers to a specific idea from card 1/1b1
1/1b1b Continues the notes from card 1/1b1a
1/1c Continues the notes from card 1/1b
1/2 Continues the notes from card 1/1
PERMANENT NOTES
References
LITERATURE NOTES

Figure 1. Zettelkasten with card indexes

Overall, the Zettelkasten method is a flexible and adaptable system for note-taking and organization that emphasizes the importance of linking related ideas and concepts across different Zettels. It allows for a personalized and evolving approach to knowledge management and has been widely used by academics, writers, and researchers as a tool for creative thinking and intellectual exploration.

2. Adapting the Zettelkasten method

The Zettelkasten method is originally not meant for software documentation. There are some hurdles that must be overcome to adapt and use it for this purpose:

1. Difficulty in version control: With the Zettelkasten method, individual Zettels do not usually have different versions. In software documentation, version control is critical to ensure that readers are working with the correct documentation for their version of the software [SB95].
2. Inability to handle multimedia content: The Zettelkasten method is primarily focused on text-based Zettels, and it may not be suitable for managing multimedia content like images, videos, or audio recordings. Software documentation often includes screenshots, diagrams, and other visual aids that help understand the software's features [CBB10].
3. Lack of structure: The Zettelkasten method does not provide a predefined structure for organizing information, which can make it difficult to navigate and find information quickly. In contrast, software documentation requires a clear and consistent structure to help understand the software's features and functionality [CBB10].
4. Limited collaboration: The Zettelkasten method is primarily designed for individual use, and it may not support collaborative editing or commenting. In software documentation, it is often essential to have a team of writers and developers working together to create and maintain documentation [Mar14].

These hurdles arise due to the inherent differences between the Zettelkasten approach and the unique requirements of software documentation. As such, in order to make the Zettelkasten method work effectively for organizing software documentation, it is crucial to address these concerns or find ways in which they can be mitigated.

2.1. Moving to a digital format

While originally the Zettelkasten was created using paper Zettels and physical boxes, current software enables us to move away from such primitive tools. Using Niklas Luhmann's card indexing system in a digital format can help to address some of the challenges of applying this note-taking method to software documentation.

There are many digital tools that enable the creation and use of digital Zettels – basically any Wikipedia-like tool could work, that is, any tool that allows creating pages and placing links to other pages in them, without imposing a strict structure. Some such apps are Foam, Zettlr, Zettel Notes, Evernote, Roam Research, Obsidian, or TiddlyWiki. These tools enable the creation and linking of markdown files. They also provide features such as tagging and search functions, which would be helpful for organizing and retrieving Zettels.

One of the strengths of the Zettelkasten method is its emphasis on linking related Zettels. In an electronic format, it becomes possible to create backlinks between Zettels to connect related ideas and concepts. This is important for software documentation, as it could help a first-time reader find needed information. By jumping from one link to another the reader can use the knowledge they just acquired to guide them to what they are looking for and at the same time familiarize themselves with how the system is structured [Ahr17].

By using a digital platform for Zettelkasten note-taking, it becomes easier to manage versions of Zettels and track changes over time. This could involve using version control software or simply tracking changes within the note-taking software itself. Having versions of software linked to their corresponding version of the Zettelkasten would solve the problem of version control.

Digital note-taking platforms can support multimedia content, such as images, videos, and audio files. This can be particularly useful in software documentation, where multimedia content can help to illustrate complex concepts and procedures [May14].

Digital platforms can also facilitate collaboration by allowing multiple individuals to access and edit Zettels in real time. This can be especially useful in software documentation, where multiple individuals may need to contribute to the same document [Mar14]. Markdown applications can also offer a high degree of customization, allowing users to create custom templates and categorize Zettels using tags or keywords. These features could allow multiple users to establish common standards for the Zettelkasten, which could further improve collaboration.

2.2. Ever-changing structure

One of the defining features of the Zettelkasten note-taking method is its lack of pre-defined structure. Unlike many other note-taking systems, the Zettelkasten method does not provide a specific framework or hierarchy for organizing information. Instead, the system is designed to be highly flexible and adaptable to an individual's unique thought processes and learning style [Ahr17].

While there are ways that Zettels can be grouped and linked together, there is no strict system on how they are connected. The links between Zettels are typically bidirectional, meaning that the links work both ways. This creates a network of interconnected Zettels, allowing for a non-linear way of organizing and exploring information [Ahr17]. While this lack of structure can be liberating, it can also be a challenge when it comes to navigating it. Without a clear organizational framework, it can be difficult to find specific Zettels or to understand the relationships between different pieces of information. This can lead to a sense of disorganization, particularly as the number of Zettels in the system grows [Mae06].

To solve the problem of disorganization it would be possible to modify the Zettelkasten method to make its structure more strict. There is an option to make the links between Zettels one-directional and not allow forming loops – so following the links from one Zettel to another can never lead back to the starting Zettel. In this way, it would be possible to transform the system into a tree-like structure with a strict hierarchy, similar to, for example, a file tree. Such an approach would be great, for example, for the documentation of code – which forms strict hierarchies – having classes, sub-classes, dependencies, etc. And yet, there are already other ways to represent strict hierarchies in documentation – even the aforementioned file tree would already represent such relationships [RMA15].

When adapting the Zettelkasten method for software documentation the author thinks that it is important to lean into its loose structure and lack of strict hierarchies, which can become a strength. Many real-world systems are non-linear and do not follow a strict top-down hierarchy. This emergent structure can also be observed in complex software systems that are composed of many independent parts or modules, where the behavior of the system as a whole is determined by the interactions between these parts. As the system evolves and new components are added, the structure can become more complex and difficult to predict [Mai09; Mit09]. When it comes to adapting to systems with an emergent structure, the Zettelkasten method could overtake other documentation methods due to the adaptability of its structure. So, when using this method for documentation it is important not to hinder the natural development of connections between different documented concepts. Imposing directionality or restrictions to the links between Zettels would be a waste of the Zettelkasten method's original flexibility. The unexpected connections that form between different elements could lead to inspiration for how to improve or further develop software.

2.3. The problem of collaboration

The Zettelkasten method is highly personalized because it is designed to reflect an individual's unique thought processes and knowledge organization. This system is built around the idea of creating a second brain, which means that the notes and connections made in the Zettelkasten are specific to the individual's thinking and learning [Ahr17].

While the Zettelkasten method is primarily designed for individual note-taking and knowledge management, there are ways the system could be adapted for collaborative use:

1. Standardize identifiers: To make it easier for multiple individuals to navigate the Zettelkasten, it could be helpful to standardize the unique identifiers used in the system. This could involve setting naming conventions or agreeing on a set of common tags that will be used [TH20].
2. Use shared software tools: Collaborative Zettelkasten note-taking could be facilitated by using software tools that allow for easy access and editing by multiple individuals [Mor12].
3. Agree on a common framework: While the Zettelkasten method is designed to be highly flexible, collaborative work often requires a shared understanding and agreement on how information is organized and linked [Mor12]. As such, it could be helpful to agree on a common organizational framework for the collaborative project. This could involve agreeing on a set of topics or themes that will be used to organize Zettels, or developing a shared hierarchy of information [Eva03].
4. Review and discuss notes regularly: Collaborative Zettelkasten note-taking should involve regular review and discussion to ensure that everyone is on the same page and to identify connections and themes that may have been missed [Mar14].
5. Establish clear roles and responsibilities: To ensure that the collaborative Zettelkasten note-taking process is productive and efficient, it could be helpful to establish clear roles and responsibilities for each member of the team. This could involve designating specific

individuals to oversee note-taking, organization, and review [Mar14].

To combine the personalization aspect of the Zettelkasten method with collaboration in software documentation, the author thinks a mixed approach could be taken. The Zettelkasten could have both an "official" version - one that is common to the whole development team and is shared with relevant outside parties - and an individual version that each developer would keep for themselves.

The Zettelkasten could be kept in a collaborative environment, for example, a GitHub repository, where the main branch of the repository would be the "official" version of the documentation. The main branch would have Zettels that are relevant to the whole team. They would have to be written in a less personal way - having in mind, that they should be understandable and relevant to multiple readers [CBB10].

Not all notes that would be helpful to a developer personally are meaningful for general use [TH20]. With this in mind, an application's developers could create their own long-lived personal branches of the repository based on the main branch of the Zettelkasten. Each developer would be free to add their own Zettels and make changes to their own branch, which would include personal observations or ideas related to the project. These Zettels would not be merged with the main branch of the documentation repository but would be available for individual use. The personal branch could also be used exclusively locally if some notes should not be seen by other members of the team. Developers could also periodically merge changes from the main branch of the Zettelkasten to their individual branch to ensure that it is up-to-date.

In software development, long-lived branches are generally avoided since they can lead to conflicts and make the process of merging code difficult [HF11]. However, with the Zettelkasten method, it would be possible to create long-lived personal branches for software documentation without the same risks. Existing Zettels should rarely be modified - because of the small scope of a single Zettel, new additions to the program should usually be documented solely through the addition of new Zettels, while linking them to existing ones. So, unless there are major changes to a program's functionality or how its components are linked, merging new Zettels into or from the main branch should not cause conflicts.

In addition, in the event that certain personal notes become relevant and useful to a wider audience, there is a possibility to incorporate them into the main branch of the documentation repository. In this case, the Zettels can be evaluated and considered for integration into the main branch of the documentation repository. This integration would involve reviewing a Zettel's content, ensuring its applicability to the project as a whole, and making any necessary adjustments or adaptations. By incorporating valuable personal notes into the main branch, the documentation can become more enriched.

By using this approach, each team member could have a space for personal notes and ideas that may not be relevant to the general documentation, while still collaborating with the rest of the team. This approach could allow for flexibility and customization while ensuring that everyone has access to general information and can work together effectively.

2.4. Best uses for the method

The Zettelkasten method can be applied to various types of software documentation, but its utility may vary depending on the context and purpose of the documentation. In the case of internal documentation, the Zettelkasten method can be particularly valuable.

Internal documentation is primarily intended for developers, engineers, or other team members involved in the software development process. It serves as a knowledge repository that helps team members understand the system architecture, codebase, design decisions, and other technical aspects [Pre10]. In this context, the Zettelkasten method can enhance the effectiveness of internal documentation.

The Zettelkasten method aligns well with Agile software development practices, as it facilitates the incremental and iterative capture of knowledge [Ahr17; Mar14]. Developers can continuously add new Zettels as they encounter new challenges, solve problems, or gain insights. The ability to capture knowledge in small, discrete units makes it easier to document and share information in an agile and flexible manner, adapting to the evolving needs of the development team.

Additionally, the Zettelkasten method encourages developers to add personal notes and insights to the documentation. These notes can include explanations, code snippets, observations, best practices, or lessons learned. By capturing personal experiences and knowledge, the Zettelkasten method allows developers to share their expertise and provide valuable insights to their colleagues. These notes can contribute to a collective understanding of the software system and facilitate knowledge transfer within the development team [TH20].

While the Zettelkasten method can still be applied to end-user documentation, its benefits may be less pronounced in this context. End-user documentation focuses on providing clear, concise, and user-friendly instructions or explanations to non-technical users [CBB10]. Personal notes and individual insights may not be directly relevant or useful for the general audience, as they are typically specific to the developer's perspective or coding practices.

Overall, the Zettelkasten method provides several benefits that could make it particularly useful for internal software documentation within development teams. With that in mind, the author had several ideas for more precise areas of documentation where the method could be used:

1. Domain modeling: The Zettelkasten method could be used for domain modeling in software documentation by creating structured and interconnected Zettels that reflect the domain's structure. Developers could capture their ideas, observations, and insights as they occur, and then link the Zettels together to create a more comprehensive understanding of the domain.
2. Documenting design decisions: By creating Zettels for each design decision, developers could easily link them to other related Zettels to provide context and detail. This approach could make it easier for new team members to understand the rationale behind design decisions and can help ensure that the design remains consistent over time.
3. Capturing requirements: By creating Zettels for each requirement, developers could eas-

ily link them to existing Zettels, e.g., ones containing user stories or acceptance criteria. This approach could ensure that all requirements are captured with their relevant context, which can help prevent misunderstandings.

4. Maintaining a knowledge base: As developers work on a project, they encounter a variety of information such as common issues and their resolutions, troubleshooting guides, lessons learned from past projects, and insights into industry best practices. By using the Zettelkasten method, they could create a system for capturing and organizing this information.

One of the main benefits of using the Zettelkasten method for software documentation is that can allow for a mixed approach, which can unify all these areas of documentation. The key to this is the bidirectional nature of the links between Zettels. Unlike other documentation systems that may specify a specific type of relationship between elements, such as hierarchical or sequential relationships [CBB10], the Zettelkasten method simply links related concepts – without specifying the exact way they are related. That means connections between Zettels could represent different types of relationships. As a result, the Zettelkasten method could enable developers to capture and organize a wide range of information, such as domain models, design decisions, requirements, and technical details, into a unified knowledge base.

2.5. Setting common standards

The importance of setting common standards for software documentation cannot be overstated. Standards provide quality assurance support, reduce costs, support interoperability, and promote international consensus on best practices. However, there are also considerations such as possible reduced flexibility, confusion from competing standards, and the need for ongoing maintenance and updates. Despite these downsides, the benefits of standards far outweigh the negatives [Sma13]. When using the Zettelkasten method for software documentation, setting some common standards can enhance the usability and organization of the documentation.

2.5.1. Setting naming conventions

While Niklas Luhmann's card indexing system is effective for organizing physical index cards, it may not be as useful for digital note-taking and documentation. The main benefit of N. Luhmann's indexing system is that it allows for new Zettels to be linked to existing ones while at the same time keeping their previous order unchanged [Ahr17]. It also provides a unique identifier with meaningful connections, encouraging careful consideration of each Zettel in its context and making related Zettels easier to locate. While useful for arranging physical Zettels in a sequence, it also imposes a hierarchical structure on a system that is meant to be non-hierarchical [Kad21].

The main disadvantage of N. Luhmann's indexing system is that it is unnecessary in a digital Zettelkasten. The digital counterpart to the previous physical indexes of the Zettels, which served as their primary identifier, would be the names of the corresponding digital files. When using digital Zettels, it is much easier to insert new ones and link them without changing their order simply by using backlinks (which are not affected by the addition of new Zettels). Also, the benefits of N. Luhmann's indexing system can be easily replicated in a digital Zettelkasten through the use of descriptive phrases as filenames. Digital Zettels can be viewed together by keyword, and it is even shown which Zettels link to the current one being viewed. Unlike a paper-based system, a digital Zettelkasten allows for a search of every word in a Zettel, not just the main topics or links. Therefore, the hierarchical arrangement enforced by N. Luhmann's indexing system is unnecessary in a digital Zettelkasten, and descriptive filenames can provide the same benefits without limiting the flexibility of the system [Kad21].

One option for naming Zettels could be to use a unique ID that is separate from the content of the Zettel, e.g., corresponding to the date the Zettel was created. This would provide the advantage of avoiding broken links due to filename changes. However, such numbering would not provide the readers with much context [Kad21]. The use of numbers for card identifiers can be limiting and may not provide enough flexibility for categorizing Zettels in a way that is meaningful for software documentation. As such, it is more reasonable to assign other, more useful, types of identifiers.

Titling Zettels with full names can be more beneficial in a digital Zettelkasten system. Names can provide more flexibility for organizing and categorizing Zettels, allowing for more specific and descriptive categorization. Names can also be easily searched and filtered, making it easy to find Zettels related to a specific topic or keyword [Kad21]. This can help to speed up the process of organizing and searching for Zettels, which can be particularly useful in software documentation since the Zettelkasten will be viewed by multiple people.

One difficulty when naming Zettels might be file naming conventions. Restricted symbols, such as the forward-slash ("/"), can pose a problem due to conflicts with file systems. To address this, alternative naming approaches can be used, such as replacing restricted symbols with suitable alternatives like hyphens ("-") or underscores ("_") to ensure meaningful names without causing file system errors.

An additional downside of using phrases to name Zettels is that at a later point, a Zettel may be deemed more appropriate for a different topic, or it may require a different name for some other reason. Such a change would result in broken links to that Zettel throughout the entire Zettelkasten. However, some software may automatically update links. Alternatively, it is possible to use the find/replace function to search the Zettelkasten and change all links to the Zettel accordingly [Kad21].

Another issue that may arise when naming Zettels is that in some cases it is possible for two or more Zettels to have the same name. This can lead to confusion and difficulty in organizing and linking Zettels, or simply not be allowed by the used software. This can happen if Zettels are named after their content or if they cover similar topics. For example, two Zettels with the title "Using the Application" could cover different aspects of using an application, or two Zettels with the title "User Issue" could have different subtopics.

A solution to this problem is to use a naming convention where the name of the Zettel includes the topic, subtopic, or concept it relates to, in addition to its contents. For example, instead of naming a Zettel ”UserIssue”, it could be named ”UserIssue-RenamingFiles-ChangesNotSaved”. This naming convention allows for more descriptive and specific names, making it easier to distinguish between Zettels that cover similar topics. Additionally, using a consistent format for naming Zettels, such as starting with the main topic and then adding subtopics or details, can also help avoid duplicates.

If it is still impossible to avoid duplicate names, there exists the option of adding an index (e.g., Zettels titled ”Using the Application 1” and ”Using the Application 2”). This, however, leads back to the problem of lack of context and imposes hierarchy where it may not be desired. In this case, the author recommends simply merging the two Zettels together, as, if they cannot be separated into subtopics, they should not be. Such merging, however, should only be done with careful consideration, as it is best to keep the Zettels as concise as possible.

Having this in mind, here are some Zettel naming conventions that could be used, depending on the needs of the development team (examples are based on possible Zettels a file-sharing web service might have):

1. Verb-based: Uses verbs to describe the action the Zettel is explaining. Examples: ”Uploading a File”, ”Sharing a Folder”, ”Downloading a Document”, etc.
2. Feature-based: Uses the name of the feature or functionality being explained in the Zettel. Examples: ”File Versioning”, ”Collaborative Editing”, ”Permission Settings”, etc.
3. Problem-based: Uses the problem or error being addressed in the Zettel. Examples: ”Fixing a Broken Link”, ”Resolving Upload Errors”, ”Troubleshooting Access Issues”, etc.
4. Task-based: Uses the name of the task or process being explained in the Zettel. Examples: ”Creating a New User Account”, ”Resetting Passwords”, ”Setting Up 2-Factor Authentication”, etc.

If the documentation involves multiple software products, e.g., a file-sharing service and a user management service, it is important to ensure that the Zettel names are unique and do not overlap even if the services have Zettels on similar topics. One way to achieve this is to include the name of the service as part of the Zettel’s name.

For example, if the Zettel is about resetting passwords in the file-sharing service, the name could be ”FileSharingService-ResettingPasswords”. Similarly, if the Zettel is about resetting passwords in the user role/permission management service, the name could be ”UserManagementService-ResettingPasswords”. This naming convention would ensure that the Zettels are unique and easy to search for based on the specific service they are related to.

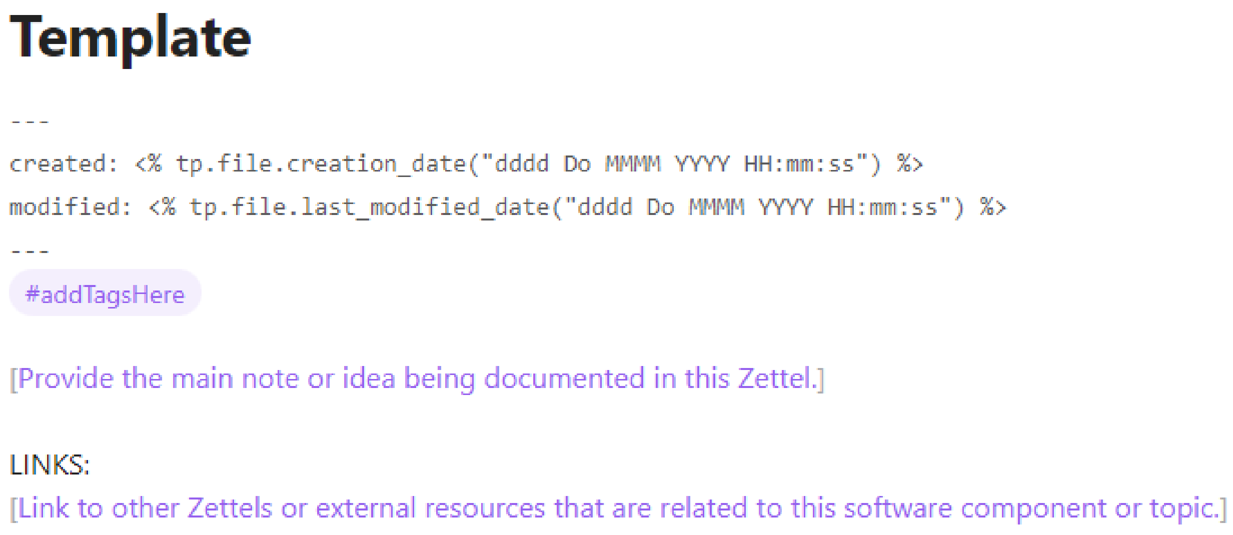
2.5.2. Standardizing Zettels

To make documentation effective, it is important to have a clear and standardized structure that is communicated to the reader. This structure not only benefits the reader by making it easy to navigate the document and locate information, but it also helps the writer by providing a framework for organizing the content and identifying areas that need further attention. A standardized structure also ensures that the document is comprehensive and includes all the necessary information, as each section represents an important piece of the overall picture [CBB10].

While the overall structure of the Zettelkasten is meant to be flexible, the Zettels themselves can be standardized. There is no one-size-fits-all answer to what a Zettel template should contain, as it could vary depending on the specific needs and preferences of the user or organization or even on the software that is being documented. What a Zettel template looks like and how its elements are organized and positioned will also heavily depend on the tool being used for the creation of the Zettelkasten. Some tools can automatically format the Zettel's title according to the file name or add relevant metadata. Nevertheless, there are some common elements that could be included in a Zettel to help ensure that it is clear, organized, and easily searchable:

- Title: A descriptive and concise title that summarizes the content of the Zettel, which can be used to link or refer to it from other Zettels or documents.
- Content: The main body of the Zettel, which can include text, links, code snippets, diagrams, or any other relevant information related to the topic.
- Links: Links to other Zettels that are related to the current one, creating a network of interconnected information.
- References: Citations or links to external resources or documentation that are relevant to the Zettel.
- Tags: Keywords or labels that can help with categorizing and searching for Zettels.
- Metadata: Optional information such as the creation date, author, source, or status of the Zettel.

An example Zettel template can be seen in Figure 2.



Template

```
---
created: <% tp.file.creation_date("dddd Do MMMM YYYY HH:mm:ss") %>
modified: <% tp.file.last_modified_date("dddd Do MMMM YYYY HH:mm:ss") %>
---
```

#addTagsHere

[Provide the main note or idea being documented in this Zettel.]

LINKS:
[Link to other Zettels or external resources that are related to this software component or topic.]

Figure 2. Zettel template example created with Obsidian

In addition to the content of the Zettels, their length should also be standardized – or at least controlled. Keeping the content of Zettels short has several benefits:

1. It allows for easier navigation and quicker scanning. When a Zettel is short and focused, developers can quickly understand its main point and decide whether or not it is relevant to their current task.
2. Short Zettels encourage brevity and clarity of thought. Developers are forced to distill their ideas down to their essence, which can help improve communication and reduce misunderstandings.
3. Short Zettels are easier to link together. When Zettels are focused on a single topic, it is easier to create meaningful links between them. This can help developers see connections between different parts of the system and gain a more comprehensive understanding of the domain.

Overall, by using a standardized template, it may become easier to quickly create and organize new Zettels. Also, keeping the content of Zettels short could help encourage clarity, brevity, and meaningful linking – this could help readers gain a more comprehensive understanding of the information. As such, these practices could lead to more effective documentation overall.

2.5.3. Standardizing tags

In the context of software documentation, the Zettelkasten method can be enhanced by incorporating tags, despite their absence in the original framework. Tags play a pivotal role in facilitating navigation and classification. While the arrangement of notes itself aids in navigation, tags offer a means of classification through metadata. They provide descriptive attributes that define a note's characteristics, such as topic, function, feature, or user role, enabling the organization of notes into distinct categories. By assigning multiple tags to each note, users can retrieve information in various ways and quickly locate related notes that share common tags. This significantly enhances the usability and efficiency of the documentation, making it easier to navigate and comprehend [Sma13]. Therefore, in the Zettelkasten method for software documentation, the consistent use of tags is vital for effective information management.

When creating a Zettel, it is named in a way that reflects both its topic and content. The main goal of this is to make sure that the Zettel's name is unique and easily recognizable so that it can be easily found and identified later on. When dealing with large and complex systems, there can be many different topics and subtopics to keep track of. Adding all related topics to the Zettel names is usually not necessary and can make the names long and difficult to read. Instead, a better approach would be to avoid overloading the names with too much information – only using the most important keywords for naming, while using tags to expand on the other relevant topics and subtopics.

For example, instead of naming a Zettel ”User requested a change to authorization requirements for security purposes”, it could be named simply ”Change to authorization requirements”, and tags would be added to expand on the various topics that the Zettel is relevant to, such as ”user requests”, ”security” and so on. This would make the name concise and easy to read, while still providing enough information to identify the Zettel's contents.

Tags in the Zettelkasten method can serve more purposes than just describing the topics of the Zettels. They can also be used to create different views of the Zettels based on various criteria, such as their relevance to a particular project, their status, their type, and so on. This is similar to how views are used in domain modeling.

In domain modeling, views are different ways of looking at the same system. The elements of the system remain the same, but the relationships between them are represented differently in each view [CBB10]. Similarly, in Zettelkasten, while the Zettels are unchanged, they can be filtered and organized according to tags to represent a specific view of the system.

Just as domain modeling views allow us to focus on specific aspects of the system and analyze them in detail [CBB10], Zettelkasten tags allow us to quickly retrieve and analyze specific information within our documentation. By filtering Zettels according to tags, we can create different ”views” of our system documentation, focusing on specific features or aspects of the software. Tags can allow us to better understand the system we are working with, by providing us with different perspectives on the same information.

Since tags are so important when using the Zettelkasten method digitally, it is desired to set some standards for them, especially while working in a collaborative environment. Unlike Zettel names, which have to be unique and descriptive, tags can be shorter and more general in nature. This is because a tag is used to group together multiple Zettels that share a common theme or topic.

When creating tags for Zettels, it is generally better to use short and concise terms rather than long and complex ones. Short tags are easier to remember and apply consistently, reducing the risk of confusion and making it easier to quickly search for Zettels. Additionally, using shorter tags allows for more flexibility in how Zettels are categorized, as it is easier to add new tags or modify existing ones without creating redundancy or inconsistency [TH20].

Furthermore, it is better to use several short tags to describe a Zettel rather than a single long one. This is because a single tag that is too specific may only encompass a small number of Zettels, making it less useful as a search or filtering term. In contrast, several short tags can be used to describe a Zettel from different angles or perspectives, making it more versatile and applicable to a wider range of contexts [TH20].

The specific tags used in the Zettelkasten method for software documentation will highly depend on the system being documented and the needs of the team or individual using it. Overall, using short and concise tags in the Zettelkasten method for software documentation will help to keep Zettels organized and easily accessible, without adding unnecessary complexity or redundancy to the system.

2.6. Requirements for tools

Choosing the right tools to use when implementing the Zettelkasten method for software documentation would be crucial to its success. Even so, there are several things that would be generally desired and are looked at when deciding what tool to use for documenting software with the Zettelkasten method [Ahr17; Kad21; TH20]:

- Collaboration: Does the tool offer collaboration features for working in a team environment?
- Linking: How easy is it to create links between Zettels?
- Searching: How effective is the search functionality?
- Ease of use: How easy is the tool to use and navigate?
- Customization: Can the tool be customized to fit specific needs?
- Integration: Can the tool be easily used together with other tools?
- Accessibility: Can the documentation be easily accessed by interested parties?
- Tagging: Can Zettels be easily tagged and filtered for easier retrieval?
- Graph-view: Does the tool offer a graph view feature for visualizing the relationships between Zettels?

Options for collaboration are a must-have when working in a team. While any text-based tool could be used in tandem with a version control tool such as Git to enable sharing the Zettel files, any additional features that would facilitate collaboration are desired. Meanwhile, linking and searching are necessary features, and they will be used often when creating or reading the Zettelkasten. While it is not mandatory to have an easy-to-use application with options for customization, these factors may determine how readily and quickly it is adapted by development teams. The author thinks it is worth it, however, to elaborate more on the importance of the other points.

2.6.1. Integration

When using the Zettelkasten method for software documentation, it is important to have a tool that can be integrated with other tools used by the development team. This is because it can be inconvenient and time-consuming to manually create links between different work items and documentation. Automating this process would minimize the possibility of human-caused errors and increase efficiency. For example, if a team uses another tool for issue tracking, such as Jira or Azure DevOps, it would be ideal if the Zettelkasten tool could automatically create backlinks between work items and documentation. This would ensure that all relevant information is properly connected, without the need for manual linking.

It is technically possible to use some existing wiki platforms for implementing the Zettelkasten method, which would solve integration concerns. It should be noted, however, that these platforms are not inherently designed with the specific needs and principles of the Zettelkasten approach in mind. Most wiki platforms are designed for creating and organizing larger documents or pages rather than discrete, atomic units of information like Zettels and may lack specific features that are tailored to the unique needs of the Zettelkasten method. Such platforms often employ a hierarchical structure, where pages are organized in a tree-like directory system [Klo06]. This hierarchical organization may not align well with the associative nature of the Zettelkasten method, where connections between notes can span across different topics and categories. While it is possible to adapt existing wiki platforms for Zettelkasten-like workflows, it may still require customization, configuration, and the use of additional tools or plugins.

2.6.2. Accessibility

Additionally, the documentation should be easily accessible and shareable among interested parties. An online tool, such as Notion or Roam Research, could be a good option as it can be easily linked to and accessed from a browser. This would enable readers to quickly access the documentation they need, even if they do not have some specific software installed. However, it is important to note that some offline tools like Obsidian or Foam can also be made more accessible through plugins, allowing for similar functionality.

2.6.3. Tagging

When using the Zettelkasten method for software documentation, it is important to be able to add tags to Zettels and filter by them for several reasons. First, tags provide a way to organize and categorize Zettel, making it easier to find and retrieve specific information when needed. For example, a Zettel related to a specific feature or function of the software can be tagged with the relevant keyword, making it easy to find all related Zettels by filtering with that tag.

Second, filtering by tags can help to identify gaps or areas for improvement in the documentation. By looking at the distribution of tags, it may become apparent that certain topics or areas are not well-covered, or that there is redundant information in other areas. This information can then be used to guide the creation or revision of Zettel to improve the overall quality of the documentation.

2.6.4. Graph view

Graph-view functionality would be helpful when choosing a tool because it would allow users to visualize the relationships between different Zettels and concepts. With graph view, Zettels can be organized spatially and linked together to form a map or a diagram that represents the structure of the knowledge base. This can be particularly useful for domain modeling, where different concepts and entities need to be organized and linked together in a meaningful way [CBB10].

With graph view functionality, users could easily navigate and explore the knowledge base, zoom in on specific areas of interest, and get a high-level overview of the entire system. It could also help to identify gaps and inconsistencies in the documentation and highlight areas that require further research or clarification. Overall, graph view functionality is important for visualizing complex information, which would be especially useful in software documentation where there are often many interrelated concepts and components to consider [CBB10].

3. Using the Zettelkasten method

3.1. Choosing tools

With the increasing popularity of Zettelkasten, there are now many tools available that can be used to create and manage Zettels. Each of these tools has its own strengths and weaknesses, making the selection of the best tool largely dependent on a development team's needs and preferences. When choosing the right tool to implement the Zettelkasten method for software documentation, the author considered several applications that could meet their needs. While there are many more applications available that could help manage Zettels and markdown files, the author focused on comparing the following seven:

- Foam: `https://foambubble.github.io/foam/`
- Zettlr: `https://docs.zettlr.com/en/`
- Zettel Notes: `https://docs.zettelnotes.org/`
- Evernote: `https://help.evernote.com/hc/en-us`
- Roam Research: `https://roamresearch.com/#/help`
- Obsidian: `https://help.obsidian.md/`
- TiddlyWiki: `https://tiddlywiki.com/`

After considering the desired features and criteria previously outlined in the requirements, the author chose to use Obsidian for software documentation using the Zettelkasten method because of these reasons:

- Ease of Use: Obsidian is relatively easy to use and navigate, with a clean and simple user interface. It has a markdown-based note-taking system that allows for easy editing and formatting of Zettels. The application also has a built-in help feature that can assist users in learning how to use the software more effectively.
- Customization: Obsidian is highly customizable, allowing users to personalize the interface to fit their specific needs. Users can customize the themes, plugins, and hotkeys to streamline their workflow and improve their productivity.
- Collaboration: Obsidian can be used in a team environment with its integration with Git, which allows for version control and collaboration with others. The Obsidian Git plugin has helpful features such as automatic file backup, or the option to stage and commit individual files.
- Linking: Obsidian has a powerful linking system that makes it easy to create and navigate links between Zettels. Users can create links using double brackets around a Zettel's title, and Obsidian will automatically create a backlink to that Zettel in the referenced one.
- Searching: Obsidian has an effective search function. Users can search for Zettels using keywords or tags, making it easy to quickly locate relevant information.
- Tagging: Obsidian allows for the easy creation of tags, which makes it easy to organize and categorize Zettels by topic, project, or any other criteria. Users can filter Zettels by tag, making it easier to find related Zettels and retrieve specific information.

- Graph-View: Obsidian has a powerful graph view feature that allows users to visualize the relationships of different Zettels. The graph view shows the links between Zettels in a graphical format, allowing users to see the connections between different ideas and concepts. The tool also allows to filter which Zettels are shown in the graph by including or excluding certain tags.
- Community Resources: Obsidian has a large and active community of users who create plugins, themes, and other resources that can enhance the functionality of the application. This community is also a great resource for learning about the best practices for using Obsidian.

While other options, like Zettlr or Zettel Notes, have some of the desired features, they do not have the same level of customization or community resources as Obsidian. TiddlyWiki, Evernote, and Roam Research are good options, but they do not have the graph view feature (though it could be achieved through the use of other tools), which is an important consideration for organizing and visualizing complex information. While Foam does have a graph view feature, in the documentation it says the tool is quite unstable. By comparison, Obsidian is more stable and has a more robust set of features.

Regarding the accessibility of the documentation, while Obsidian itself does not have an online version, there are plugins that make it possible to publish the Zettelkasten to be viewed in a browser. One such plugin is Obsidian Publish, which is developed and sold by Obsidian. There is also a free and ope- source publishing tool called Digital Garden. Both tools allow configuring specific Zettels to be published online. By using these plugins it becomes possible to make the documentation accessible to its readers through a browser link, without the need to have any additional software installed.

Integration with other software tools that are used for software development is hard to achieve with any of the compared documentation tools. It is not difficult to manually create links from the Zettelkasten to, for example, tasks in the backlog (and the other way around). However, automating the creation of backlinks would require a significant investment of time and resources. It would entail the creation of plugins or extensions for both the documentation tool and the tool used for tracking development tasks, which would be quite laborious and time-consuming. Although integrating the Zettelkasten documentation method with other software tools used for software development could increase its effectiveness, it is not a necessary step for demonstrating its value. This is why the author chose to forego this integration, choosing to, instead, create two-way links between the tools manually.

3.2. Organizing existing documentation

The author used the adapted Zettelkasten method to reorganize the documentation of an existing system. The author chose two interconnected web services. One of these services was a file-sharing web service, while the other was a metadata recording and retrieval service. These are the steps that the author took to organize the documentation into Zettels:

1. Analyzing the existing documentation
2. Breaking the documentation down into Zettels
3. Assigning unique identifiers
4. Linking related Zettels together
5. Creating and assigning tags

3.2.1. Analyzing the existing documentation

The author took a look at the existing documentation to get a sense of what information is already available and how it is organized. The majority of the documentation was in the form of a wiki, consisting of several multi-page documents. Additional documentation was stored separately in SharePoint, and some was kept in a different, rarely-used older wiki. The documentation included various forms of media, such as tables, diagrams, and even video recordings.

The documentation was found to be extensive, containing various types of information such as design documents, technical debt, known issues, frequently encountered problems, and usage instructions. Some documents also contained links to the documentation of other services provided by the same company. These other services were being used by or were using the file-sharing and metadata services.

While the Zettelkasten could include the documentation of multiple services, the author decided that it is not necessary to include more than two, since they should be enough to be able to evaluate the Zettelkasten method. Since it would be beneficial to at least know what the other connected services are, the author decided to only include small parts of their documentation.

The analysis of the existing documentation revealed some potential challenges that would need to be addressed in order to effectively implement the Zettelkasten method. For example, the author found that some information was duplicated across multiple documents, making it difficult to locate specific information. The documentation was also intended for different audiences, with some information intended for use within the development team and some meant to be shared with other teams attempting to use the services.

3.2.2. Breaking the documentation down into Zettels

The author broke the existing documentation down into Zettels. This involved breaking longer documents into shorter sections and identifying key concepts or procedures that can be documented separately.

During this process, it became clear that breaking down some types of documentation was easier than others. For instance, various tables and diagrams could each have their own Zettel, without needing to be broken down further. It was the same case with various definitions of used words, concepts, API endpoints, etc. This made it easy to transfer and adapt these types of documentation to the Zettelkasten.

On the other hand, if something was written as a continuous text, additional work was needed to separate it into multiple Zettels. Splitting up some paragraphs, especially ones that covered multiple topics, proved to be quite challenging in some cases. It required the author to read through and understand the text and in some cases completely rewrite the information to be able to break it into discrete pieces.

In such cases, it was difficult to determine the appropriate granularity of each Zettel. If the Zettels are overly specific, it may result in an excessive number of Zettels that become difficult to manage and navigate. On the other hand, if the Zettels are too broad, they may lack the necessary level of detail and fail to capture the intricacies of the information being documented.

Another challenge was ensuring that no essential information is lost while breaking down the documentation. Some covered topics were scattered across multiple documents, making it difficult to capture and organize them accurately. To ensure no information was missed, the author created Zettels for every piece of documentation, even if some information was repeated. Only after everything was separated into pieces, did the author consolidate Zettels that were too similar.

3.2.3. Assigning unique identifiers

Once the documentation was broken down, each Zettel was assigned a unique name. Some parts of the documentation already had their corresponding identifiers, e.g., table and diagram names. Most of these names were already unique, and any overlapping titles could be made distinct simply by making them more specific.

Some difficulties arose, however, because some elements of the file-sharing web service, and even the service itself, were named differently in several parts of the documentation. The reason for this was probably because the documents were created at different times, and the service and element names had changed over time. This meant that the author had to consolidate the names or choose to only keep one of them. The author chose to refer to elements by the names they were given in the most recent documentation – this involved replacing the older names and also listing them in the documentation, so renamed elements could still be found by their previous names.

The author also encountered some anticipated hurdles when thinking of names. One of them was that, since two web services were being documented, some of their documentation covered similar topics. For example, each service had a Zettel that was about authorization. Naming them both ”Authorization” would create overlap – as such, the author used the naming standards set previously and made the names distinct by extending them with the name of the service that the documentation belonged to.

3.2.4. Linking related Zettels together

While creating Zettels, the author looked for connections and relationships between different pieces of information. Using the Zettelkasten method’s linking system, the author added backlinks to connect related Zettels together.

During this process, the author encountered several challenges. Initially, the author struggled with the inclination to impose hierarchical structures on the Zettels, as they were accustomed to organizing information in a top-down manner. The Zettelkasten method encourages a non-hierarchical approach – once the author embraced this mindset shift and let go of the hierarchical thinking, many new connections began to emerge.

One particular issue arose with Zettels that were initially organized as lists in the wiki, such as the comprehensive list of API endpoints. It made sense to link these individual API endpoints to the main Zettel that defined their corresponding services. This, however, created an overwhelming number of connections to the main service definition (Zettel titled ”FileShare Service”), which navigation between Zettels more difficult.

To address this problem, the author devised a solution that involved the creation of intermediary Zettels. By introducing intermediary Zettels, such as a dedicated Zettel containing the entire list of API endpoints, the author was able to establish meaningful links between the intermediary Zettel and the specific services they pertained to.

In this approach, the intermediary Zettel (titled ”FileShare API endpoints”) served as a centralized hub or reference point for the related API endpoints. Each API endpoint could then be linked to the intermediary Zettel, establishing a clear connection to the corresponding service. This method ensured a more cohesive and logical structure within the Zettelkasten, allowing for efficient navigation and retrieval of information. A graph of the resulting Zettelkasten can be seen in Figure 3.

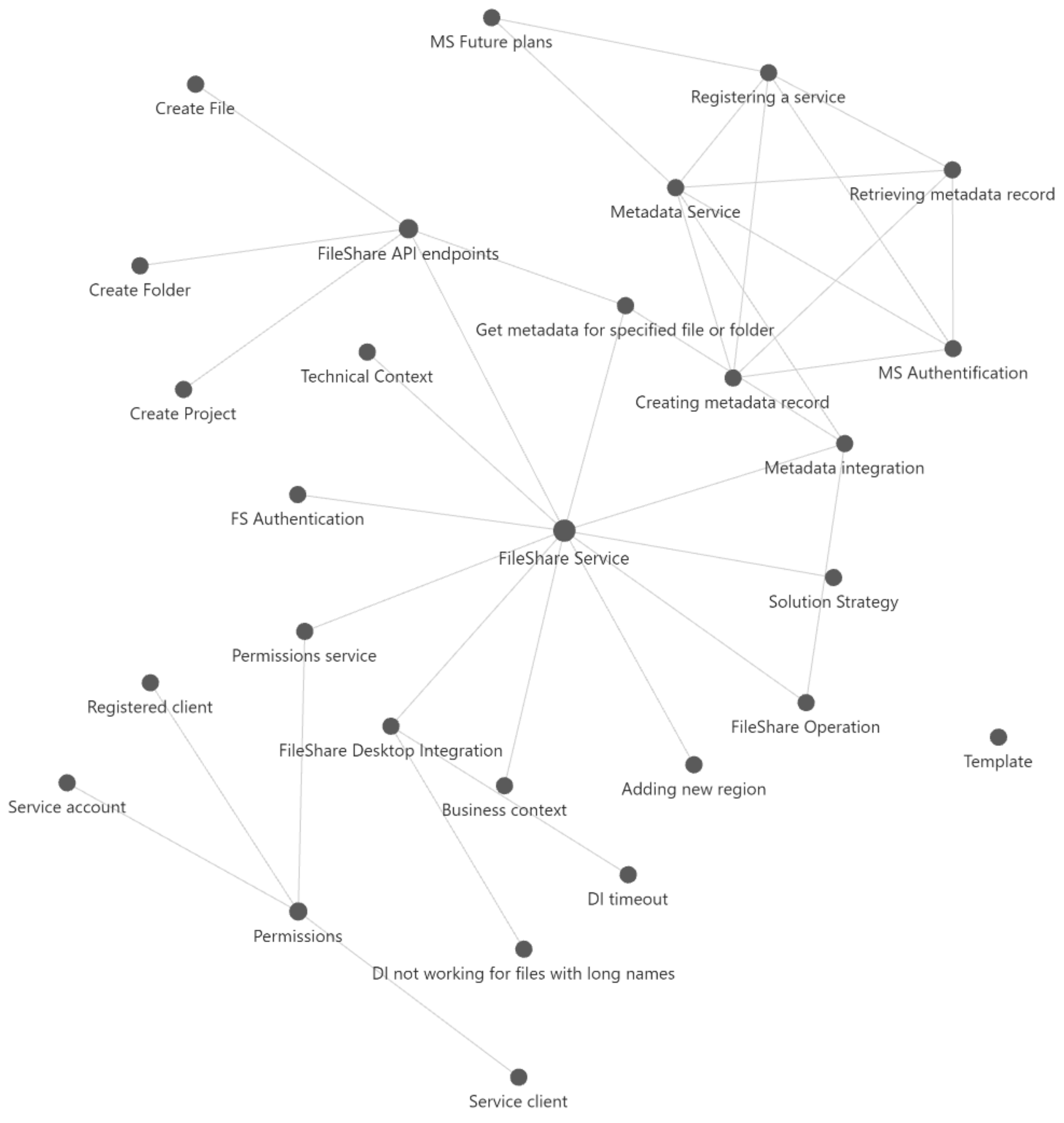


Figure 3. Graph view of the Zettelkasten

3.2.5. Creating and assigning tags

When creating and assigning tags, the author followed a systematic approach. They first identified the key themes, topics, and attributes that were relevant to the existing documentation.

While specific tags related to individual elements of the service can be valuable in certain contexts, for the existing documentation they may have introduced redundancy or confusion. The existing identifiers and links were already effective in conveying the topic and subject matter of the Zettels – they provided sufficient context and served as implicit tags themselves. As such, the need for specific tags related to elements of the documented service was minimized.

Because of this, the author only used generic tags, which were useful for the documentation because they provided a high-level categorization framework that cut across various elements of the software system. The author established a set of predefined tags (and their scopes) that aligned with the documentation's scope and objectives:

- Glossary: Zettels defining and explaining key terms, concepts, or jargon used in the Zettelkasten.
- Design: Zettels related to design documents, architectural decisions, system design, and high-level design considerations.
- Technical Debt: Zettels that discuss technical debt, code refactoring needs, areas for improvement, and potential issues arising from existing code structures.
- Known Issues: Zettels that document known issues, bugs, software defects, or limitations of the system.
- Usage Instructions: Zettels containing step-by-step instructions, user guides, tutorials, and best practices for utilizing the software effectively.
- Development Instructions: Zettels that provide guidance, instructions, best practices, or guidelines specifically related to the development process of the software.
- Security: Zettels related to security considerations, vulnerabilities, security guidelines, and recommended practices for ensuring data protection.
- Integration: Zettels that cover integration points, API documentation, third-party system integrations, and interoperability aspects.

These tags were chosen to reflect the most common and meaningful aspects of the software being documented. By using a predefined set of tags, the author aimed to maintain a standardized and organized tagging system across the documentation. A graph of the Zettelkasten with tags included can be seen in Figure 3.

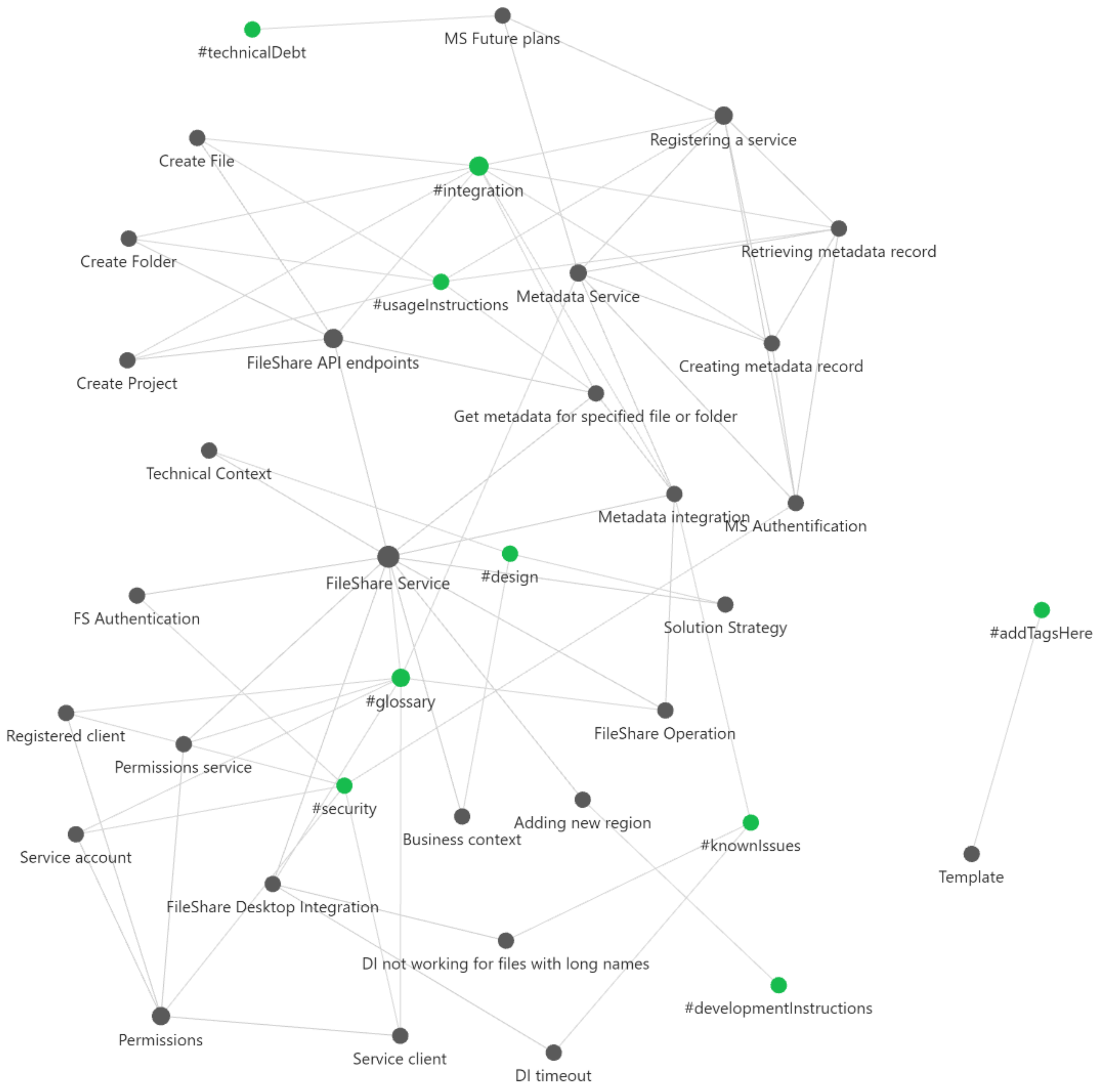


Figure 4. Graph view of the Zettelkasten with tags included

3.3. Resulting Zettelkasten

By taking the previously mentioned steps, the author created two Zettelkasten. The first Zettelkasten was created using the actual documentation of the service. This is the Zettelkasten which would be later shown to the services' developers to evaluate the effectiveness of the method for organizing documentation. The second Zettelkasten was created as a reflection of the first one but with certain elements excluded or replaced by generic keywords. Despite the changes, the author used the same naming convention and tagging system as in the first Zettelkasten. This second Zettelkasten was created to demonstrate the method in this thesis without disclosing confidential information and is the version that is shown in this thesis[1].

3.4. Recommendations for using the method

After analyzing the Zettelkasten method and exploring its adaptation for software documentation, the author proceeded to apply the method to organize existing documentation. This hands-on experience provided valuable insights and practical knowledge, leading to the formulation of the following recommendations for using the method to document software:

1. Documentation tools: When choosing tools, consider factors such as collaboration features, ease of use, search functionality, customization options, integration capabilities with other tools, accessibility, tagging capabilities, and graph view features. Evaluate different tools based on these criteria to find the best fit for your team's needs.
2. Collaboration: Develop a common organizational framework, such as a set of topics or themes. Regularly review the documentation as a team.
3. Personalization: Allow developers to keep their personal observations, ideas, and notes in an individual version of the documentation. Evaluate the content of personal notes periodically to identify any that may be relevant and useful to a wider audience. Consider incorporating these valuable personal notes into the main documentation.
4. Documentation standards: While the overall structure of the Zettelkasten should remain flexible, standardize its elements:
   - Names: Opt for full names and ensure that they remain unique. Establish a consistent format for naming Zettels, such as starting with the main topic and then adding subtopics or details. If duplicate names are unavoidable, consider merging the related Zettels together.
   - Tags: Opt for concise tags that are easy to remember and apply consistently. Use tags that would encompass multiple Zettels. Also, when tagging Zettels, use multiple short tags instead of fewer long ones.
   - Zettels: Adopt a standardized template for Zettels that includes essential elements such as a descriptive title, main content, links to related Zettels, references, tags, and optional metadata. Additionally, control the length of Zettels, keeping them short and focused.
5. Linking: Let go of the inclination to impose hierarchical structures on the Zettels – just connect related concepts. In situations where linking multiple Zettels to a single main Zettel creates an overwhelming number of connections, consider introducing intermediary Zettels.

[1]The Zettel files that were created can be found here: `https://github.com/GGretute/Zettelkasten`

4. Evaluating the Zettelkasten method

Evaluating the effectiveness of using the Zettelkasten method for software documentation in comparison to other methods could be done in several ways. The author considered several different approaches:

- Measuring the ease of use: One way to evaluate the effectiveness of the Zettelkasten method would be to measure the ease of use of the system compared to other methods. This could be done through user surveys, usability testing, or other methods that allow developers to provide feedback on the system's usability.
- Measuring the organization of information: Another way to evaluate the effectiveness of the Zettelkasten method would be to compare the organization of information in the Zettelkasten to that of other methods. This could be done by evaluating the consistency, completeness, and accuracy of the documentation produced by the Zettelkasten compared to other methods.
- Measuring the time and effort required: The Zettelkasten method may require more upfront effort to set up and maintain than some other methods. Therefore, another way to evaluate the effectiveness of the Zettelkasten method would be to compare the time and effort required to use the Zettelkasten to that of other methods.
- Comparing outcomes: Finally, the most useful way to evaluate the effectiveness of the Zettelkasten method would be to compare the outcomes of using the Zettelkasten to that of other methods. This could include metrics such as the quality of the software produced, the time required to complete projects, and the overall satisfaction of team members.

Overall, evaluating the effectiveness of the Zettelkasten method for software documentation would require a comprehensive approach that considers a range of factors, including ease of use, organization of information, time and effort required, and outcomes. Having this in mind, it would take a long time to truly measure how effective the Zettelkasten method is for software documentation – this would require for it to be used in a real-life software development scenario. This is why the author chose to only evaluate the Zettelkasten method in these two ways:

- based on the author's personal knowledge – they would theoretically evaluate the method based on the experience gained while using it and other documentation methods;
- based on the experience of other software developers – the method would be evaluated through the help of a survey.

4.1. Theoretical evaluation

4.1.1. Comparison with wiki

Given that the original documentation was in the form of a wiki, the author saw an opportunity to compare the Zettelkasten method with the wiki method. The comparison was particularly relevant to the author, as they were already accustomed to using a wiki for their documentation needs. While there may be other documentation methods available, the wiki method is widely used and well-established, making it a good benchmark for evaluating the effectiveness of the Zettelkasten method for software documentation. The author hoped this targeted comparison would enable a more in-depth understanding of the unique aspects and potential benefits of the Zettelkasten method.

Using the Zettelkasten method for software documentation is quite similar to using a wiki. Both methods utilize backlinking as a way of connecting related information, which can make it easier to navigate and organize a large amount of information. Also, both the Zettelkasten method and the wiki method are designed to be flexible and adaptable, allowing users to create and organize information in a way that works best for them [Ahr17; Klo06].

There are some differences between using a wiki and using the Zettelkasten method for software documentation:

- The Zettelkasten method uses small, bite-sized notes called Zettels, while wikis typically contain longer, more detailed documents [Klo06].
- The structure of a Zettelkasten is flexible and can be adjusted as needed, whereas wikis typically follow a hierarchical tree-like structure of pages and folders [Klo06].
- The Zettelkasten method is designed to encourage the creation of connections between notes, while wikis typically prioritize the organization of information in a linear manner [Klo06].
- In a Zettelkasten, tags are used to categorize notes, while wikis often rely on a table of contents and category pages [Klo06].

In comparison to the Zettelkasten method, using a wiki often leads to longer documents with more detailed content. While this can be useful in some cases, it can also make it harder to navigate and find relevant information quickly. Additionally, longer documents may be more difficult to link together effectively, as it can be harder to identify the main point of each section and find meaningful connections between them.

Based on the author's personal experience in using both the wiki and Zettelkasten documentation, there were noticeable differences in the time required for creating and reading the respective formats. When creating Zettels, the author found that it took more time compared to the wiki format. This was primarily due to the additional work involved in creating identifiers and establishing links between Zettels. Each Zettel required thoughtful consideration of its unique identifier and determining how it would be interconnected with other relevant Zettels. This process heavily contributed to the time investment in creating the Zettelkasten documentation.

On the other hand, the author observed that reading and understanding the Zettels was generally easier compared to wiki documents. The concise and focused nature of the individual Zettels made it easier to grasp and absorb information. This allowed for more efficient reading and comprehension, without the need to navigate through lengthy pages or sections.

However, it is important to note that the author's perception of ease in reading and understanding the Zettels might have been influenced by their personal organization of the Zettelkasten according to their own thought process. The author's familiarity with the structure and connections within the system could have positively impacted their reading experience. It is possible that another reader, with a different organizational approach or perspective, might have a varied experience while navigating and comprehending the Zettelkasten documentation.

In conclusion, both the Zettelkasten and wiki share similarities in utilizing backlinking and offering flexibility but differ in terms of note size, structure, emphasis on connections, and categorization methods. While creating the Zettelkasten requires a greater time investment, the method's focus on concise and interconnected notes may facilitate easier reading and comprehension compared to longer wiki documents. Overall, both methods have their strengths and weaknesses, and the choice between them should depend on the specific requirements and preferences of the software development team.

4.1.2. Zettelkasten and continuous documentation

The author believes that while creating many pages of documentation at once may take longer when using a Zettelkasten, the method might be better suited for building up documentation in small pieces bit by bit. Based on their own experience working in a software team that follows Agile principles, this incremental approach to adding small bits of documentation over time could encourage more developers to be proactive in creating documentation.

Software developers often encounter various useful information while working on their projects. This could involve finding solutions to repetitive user problems or gaining insights into how older or legacy code functions. These small pieces of information can be valuable to other developers, but they often do not make it into the final documentation [TH20]. The author believes that one reason for this is the daunting task of adding such information to an already existing, well-developed document, as it may not fit seamlessly with the existing flow. Alternatively, creating new documents for each piece of information poses the challenge of determining where they fit within the existing file hierarchy.

This is where the Zettelkasten method can shine in terms of incorporating diverse bits of information. By creating a new Zettel, connecting it to related concepts, and adding relevant tags, developers can add valuable information without worrying about disrupting the existing flow or hierarchy. The flexibility of the Zettelkasten method allows for the seamless integration of new information, enabling developers to capture and share valuable insights more effectively.

In the author's opinion, this approach would be particularly useful for software teams that follow a code-first approach. Based on their experience with this development approach, they have observed that the documentation often lags behind the existing codebase. The updates to documentation tend to be more large-scale and infrequent, resulting in a struggle to align the documented information with the current functionality [Mar08]. Moreover, large-scale updates may inadvertently overlook important details or specific insights.

By adopting the Zettelkasten method and encouraging the continuous addition of small bits of documentation, these challenges can be mitigated. Rather than relying solely on periodic updates, the incremental nature of the Zettelkasten method allows developers to capture and document relevant information as it arises. This way, the documentation can more effectively keep pace with the evolving codebase, ensuring that valuable insights and solutions are captured in a timely manner.

Ultimately, while the initial effort of creating identifiers and linking notes may require more time, the modified Zettelkasten method shows great potential as an alternative approach to software documentation. The author believes that the method can be most advantageous for software teams following Agile principles or employing a code-first approach. By addressing the documentation lag, reducing the risk of oversight, and enabling a more seamless integration of knowledge, this approach can foster effective knowledge management and enhances collaboration within the software development process.

4.2. Survey

4.2.1. Evaluation criteria

To measure the effectiveness of using the Zettelkasten method for software documentation, a survey was conducted. The method was evaluated in two main aspects: how easily can the documentation be created and read. There were multiple questions in each category, with possible responses being a rating on a scale of 1 to 10, where higher numbers indicated a better score. This rating system allowed for a quantitative assessment of participants' feedback.

By analyzing the survey responses and calculating the average ratings for each question, an overall assessment of the effectiveness of the Zettelkasten method for software documentation was derived. The average ratings were used to determine the level of effectiveness:

- not effective – average score below 4;
- partially effective – average score between 4 and 6;
- mostly effective – average score between 6 and 8;
- completely effective – average score above 8.

In order to provide more context to the survey's answers participants were asked about prior experiences that would be relevant to the topic. The aim was to establish a baseline understanding of their background knowledge and potential biases. For the same reason, a quantitative assessment was done on the wiki method of documentation as well. The results of these questions would serve as a reference point for comparison and enable the identification of any potential causality or biases that could influence participants' perceptions towards the Zettelkasten method. The answers would also allow for a more in-depth comparison of the wiki and Zettelkasten documentations.

Lastly, participants were asked open-ended questions about their thoughts on the Zettelkasten method. The goal of this was to gather additional feedback that would give insight into how the method's adaptation could be later improved. Answers to these questions would not have a direct influence on the final evaluation of the method's effectiveness.

4.2.2. Methodology

The participants of the survey consisted of the author's colleagues – 10 software developers from the same company. While not all developers worked directly on the two services being documented, most of them were familiar with these services as they worked on other related software products.

During the survey, its participants were presented with existing documentation that had been organized using a wiki and using the Zettelkasten method. Then the participants were asked to provide feedback and insights by answering a series of survey questions.

The survey was conducted through Google Forms, which provided a convenient and accessible platform for collecting responses from the participants. Using a survey format allowed for structured feedback and facilitated a comparative analysis between the Zettelkasten method and the more traditional approaches for software documentation.

4.2.3. Survey questions

Refer to Appendix 1 for all survey questions[2].

The survey consists of four sections, each serving a specific purpose:

1. General Information: This section collects basic demographic information about the participants, such as their years of experience in software development and their familiarity with the Zettelkasten method and using a wiki for software documentation. This information helps provide context to the responses.
2. Reading Documentation: Participants are asked to view the documentation of two services, one organized in a wiki and the other using the Zettelkasten method. They are then asked to rate their understanding of the system by viewing the documentation at a quick first glance, the intuitiveness of navigation, the speed of finding information, and how well the connections between components are represented. These questions aim to evaluate the accessibility, coherence, and comprehensibility of the documentation methods.
3. Creating Documentation: Participants are provided with an explanation of the principles and process of creating documentation using the Zettelkasten method. They are then asked to rate the ease of learning to create documentation using a wiki and the Zettelkasten method, the speed of adding new documentation, the flexibility in accommodating changes, and the facilitation of collaboration among team members. These questions assess the usability and flexibility of the documentation creation process.
4. Overall Feedback: Participants are invited to provide their thoughts on the strengths and weaknesses of the Zettelkasten method compared to other documentation methods, suggest improvements, and share their willingness to use the Zettelkasten method for their own documentation needs. This section aims to gather comprehensive feedback and additional insights from the participants.

4.2.4. Survey results

Please refer to Appendix 2 for all survey answers.

[2]Google Forms link: `https://forms.gle/BjF22NetHxE6o2z28`

According to the survey results, the Zettelkasten method received an overall average rating of 7.2875, slightly lower than the wiki's average of 7.3875. Both methods fell within the range of 6 to 8, suggesting that according to the set criteria, participants considered the Zettelkasten method to be mostly effective for software documentation.

The average rating for readability-related questions regarding the Zettelkasten method was 7.275, slightly higher than the wiki's average rating of 7.025. This suggests that participants found the Zettelkasten method to be slightly more readable compared to the wiki for software documentation.

In terms of ease of creation, the Zettelkasten method received an average rating of 7.3, slightly lower than the wiki's average rating of 7.75. This indicates that participants perceived the wiki as slightly easier to use for creating documentation compared to the Zettelkasten method.

Overall, the survey results indicate that the Zettelkasten method received favorable ratings in terms of readability, while the wiki was rated higher in terms of ease of creation. A chart comparing the average ratings for each question about the wiki and Zettelkasten documentation can be seen in Figure 5.

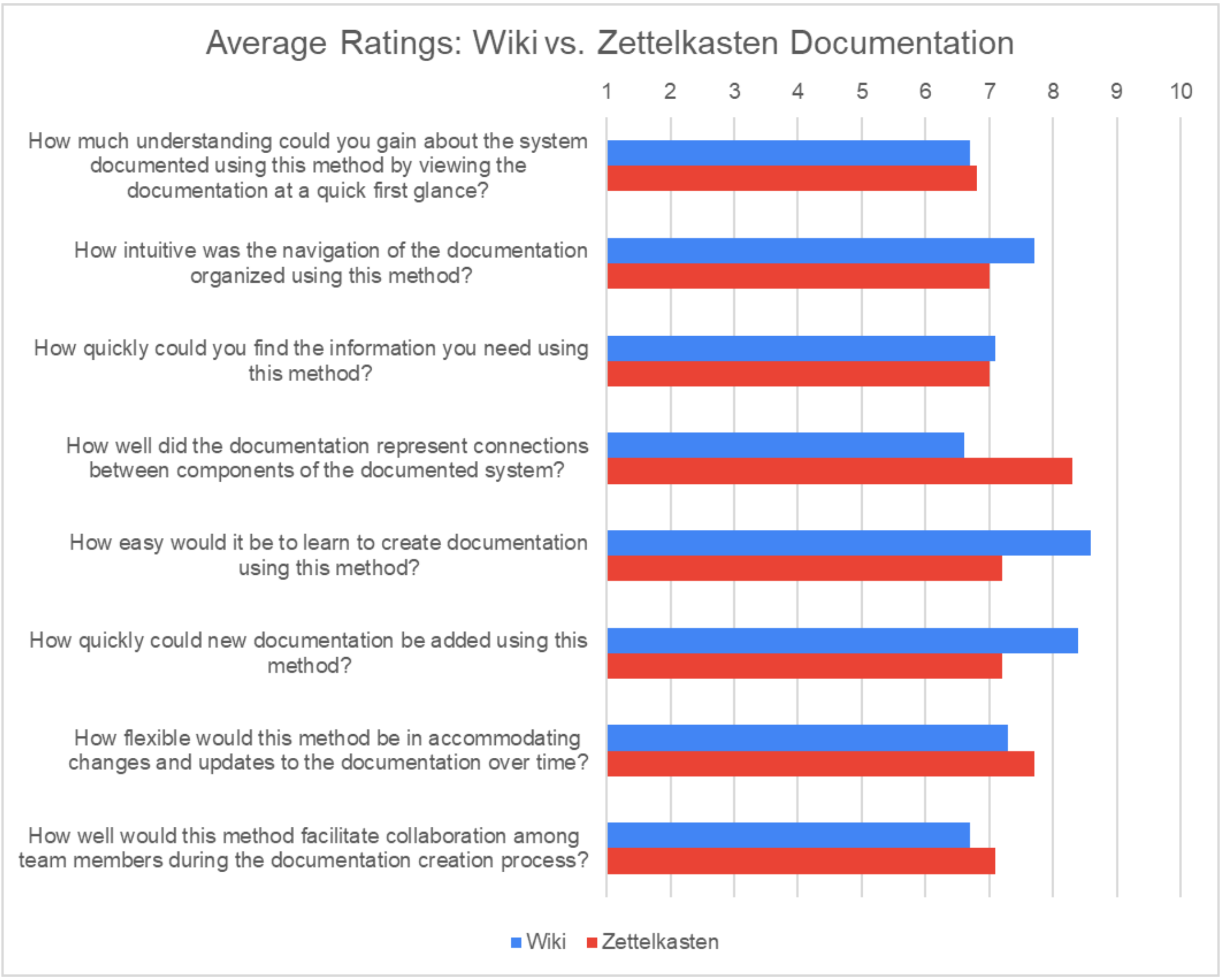


Figure 5. Average Ratings: Wiki vs. Zettelkasten Documentation

According to the survey results, participants who were more familiar with the Zettelkasten note-taking method tended to rate its usage for software documentation more positively. Similarly, participants who were more familiar with using a wiki for software documentation showed a positive correlation with their ratings of wiki-related questions.

According to the survey respondents, the Zettelkasten method exhibited strengths compared to other documentation methods. These strengths included its interconnected nature, which facilitated understanding the context of individual components. Additionally, the concise notes were praised for their ability to support quick comprehension. The method's flexibility in finding information through various approaches was also acknowledged, accommodating different user needs. Respondents considered the Zettelkasten method well-suited for ad-hoc note-taking and capturing random information, making it adaptable to diverse scenarios.

On the other hand, the survey respondents also identified several weaknesses of the Zettelkasten method when compared to other documentation methods. A prominent concern raised was the additional time and training required to effectively utilize this method. Due to its lesser prevalence in work environments compared to more traditional methods, navigating the Zettelkasten, especially for first-time users, was deemed challenging. Another commonly mentioned issue was the time-consuming process of adding tags and establishing relationships between notes. Lastly, the absence of strict rules within the system was viewed as a potential source of errors.

4.2.5. Survey bias and limitations

The survey results regarding the effectiveness of the Zettelkasten method for software documentation might be inconclusive due to certain limitations. Although the participants were able to view and explore the organized documentation, their experience was limited to a short duration during the survey. They did not have the opportunity to actively use and engage with the Zettelkasten method for an extended period of time to create their own documentation.

By only experiencing the organized documentation without actively participating in the creation process, the participants may not have fully grasped the nuances and benefits of the Zettelkasten method. Creating and maintaining a Zettelkasten requires ongoing engagement, iterative linking, and continuous refinement over time.

Furthermore, the participants' understanding and perception of the Zettelkasten method might have been influenced by their prior familiarity with the wiki-based documentation. Since the survey focused on comparing the Zettelkasten method with using a wiki, the participants might have relied on their existing knowledge and biases from their experience with the wiki format. This could introduce some subjective biases or preconceived notions that may have influenced their feedback.

Additionally, the survey questions may not cover all aspects of software documentation, and there is a possibility of bias toward the Zettelkasten method. Software documentation encompasses a wide range of factors – the survey may not capture the full spectrum of requirements and preferences that developers have for documentation methods. The questions of the survey might have also been influenced by the author's views. Similarly, the participants' answers might have been affected by the framing of the questions and the specific language used. The way the questions are presented may inadvertently highlight the strengths of the Zettelkasten method or downplay the strengths of other documentation methods. It is crucial to acknowledge this potential bias when interpreting the survey results.

To obtain more conclusive insights on the effectiveness of the Zettelkasten method, it would be beneficial to conduct a longitudinal study where participants have the opportunity to actively use the method for an extended period. This would provide a more comprehensive understanding of the method's impact on documentation creation, organization, and knowledge management. Additionally, involving participants with varying levels of familiarity with both the Zettelkasten method and the wiki format could help capture a more diverse range of perspectives and experiences.

In summary, while the survey results offer initial insights, the limited exposure of the participants to the Zettelkasten method and potential biases from their and the author's prior experience may make the results inconclusive in fully evaluating the effectiveness of the Zettelkasten method for software documentation. Further research and hands-on implementation are needed to gain a more comprehensive understanding of the method's benefits and limitations.

Results

In this thesis, the following results were achieved:

1. Recommendations for using the Zettelkasten method for software documentation were proposed.
2. A Zettelkasten containing the existing documentation of two services was created.
3. A survey questionnaire was made to evaluate the effectiveness of the Zettelkasten method compared to other documentation approaches.
4. Survey data was collected, consisting of responses from ten participants regarding their opinions on the Zettelkasten method.

Conclusions

Based on the conducted research and analysis, the following conclusions can be drawn from this thesis:

1. The adaptation of Niklas Luhmann's Zettelkasten method for software documentation was successfully achieved.
2. There is no significant difference in the overall effectiveness between the Zettelkasten and wiki methods for software documentation. Survey results indicated that both methods are effective, with the Zettelkasten method having a slight advantage in terms of readability and the wiki being easier to use for creating documentation.
3. The Zettelkasten method offers advantages for software teams that adhere to Agile principles or adopt a code-first approach. This is supported by survey results which indicated that the Zettelkasten method showcased strengths in terms of concise note-taking, adaptability to ad-hoc scenarios, and accommodating varied information.
4. The implementation of the Zettelkasten method may initially demand a greater time investment. The survey results confirmed this and highlighted several other weaknesses associated with this approach, such as the need for additional training for new users, and the increased potential for errors due to the absence of rigid guidelines.

Further research and experimentation can explore enhancements and optimizations to address the identified weaknesses and maximize the benefits of the Zettelkasten method in software documentation practices.

Glossary

- Zettelkasten method – A note-taking method developed by Niklas Luhmann, which involves the creation of a Zettelkasten.
- Zettelkasten – A collection of interlinked notes that is created using the Zettelkasten note-taking method.
- Zettel – Also known as a slip, it refers to a note within a Zettelkasten.

References

[Ahr17] S. Ahrens. *How to Take Smart Notes: One Simple Technique to Boost Writing, Learning and Thinking – for Students, Academics and Nonfiction Book Writers*. Createspace Independent Publishing Platform, 2017. 170 p.

[CBB10] P. Clements, F. Bachmann, L. Bass. *Documenting Software Architectures: Views and Beyond (2nd ed.)* Addison-Wesley Professional, 2010. 592 p.

[Eva03] E. Evans. *Domain-Driven Design: Tackling Complexity in the Heart of Software (1st ed.)* Addison-Wesley Professional, 2003. 560 p.

[HF11] J. Humble, D. Farley. *Continuous Delivery: Reliable Software Releases through Build, Test, and Deployment Automation*. Pearson Education, Inc, 2011. 127 p.

[Kad21] D. Kadavy. *Digital Zettelkasten: Principles, Methods, and Examples*. Kadavy, Inc., 2021. 88 p.

[Klo06] J. E. Klobas. *Wikis: Tools for Information Work And Collaboration (1st ed.)* Chandos Publishing, 2006. 252 p.

[Mae06] J. Maeda. *The Laws of Simplicity*. The MIT Press, 2006. 100 p.

[Mai09] M. W. Maier. *The art systems of architecting (3rd ed.)* CRC Press, 2009. 440 p.

[Mar08] R. C. Martin. *Clean Code: A Handbook of Agile Software Craftsmanship (1st ed.)* Pearson, 2008. 464 p.

[Mar14] R. C. Martin. *Agile Software Development, Principles, Patterns, and Practices (1st ed.)* Pearson Education Limited, 2014. 524 p.

[May14] R. E. Mayer. *The Cambridge Handbook of Multimedia Learning (2nd ed.)* New York, USA: Cambridge University Press, 2014. 99 p.

[Mit09] M. Mitchell. *Complexity: A Guided Tour*. New York, USA: Oxford University Press, 2009. 349 p.

[Mor12] J. Morgan. *The Collaborative Organization: A Strategic Guide to Solving Your Internal Business Challenges Using Emerging Social and Collaborative Tools*. McGraw-Hill, 2012. 304 p.

[MT15] S. Milett, N. Tune. *Patterns, Principles, and Practices of Domain-Driven Design (1st ed.)* Wrox, 2015. 800 p.

[Pre10] R. S. Pressman. *Software Engineering: A Practitioner's Approach (7th ed.)* Raghothaman Srinivasan, 2010. 895 p.

[RMA15] L. Rosenfeld, P. Morville, J. Arango. *Information Architecture: For the Web and Beyond (4th ed.)* O'Reilly Media, 2015. 483 p.

[Rot13] S. Roth. Les Deux Angleterres Et Le Continent: Anglophone Sociology as the Guardian of Old European Semantics. *Journal of Sociocybernetics*. 2013, volume IX, pp. 19–34.

[SB95] E. B. for Software Standardisation, C. (BSSC). *Guide to software configuration management*. Noordwijk, The Netherlands: ESA Publications Division, 1995. 43 p.

[Sch18] J. F. Schmidt. Niklas Luhmann's Card Index: The Fabrication of Serendipity. *Sociologica*. 2018, volume XII, pp. 53–60.

[Sma13] R. F. Smallwood. *Managing Electronic Records: Methods, Best Practices, and Technologies (1st ed.)* Wiley, 2013. 464 p.

[TH20] D. Thomas, A. Hunt. *The Pragmatic Programmer: Your Journey to Mastery (20th Anniversary ed.)* Pearson Education, Inc, 2020. 127 p.

Appendices

Appendix no. 1

Comparing Software Documentation Methods - Survey

Comparing Software Documentation Methods

This survey is designed to gather feedback from developers who have experience with different methods of software documentation. The survey aims to compare the Zettelkasten method with existing methods of software documentation, and evaluate the effectiveness of using the Zettelkasten method for software documentation. The survey provides a means to gather feedback from developers who have used different documentation methods, and compare their experiences with using the Zettelkasten method.

General Information

1: How long have you been working in software development?

▢ 0 to 2 years
▢ 2 to 5 years
▢ 5 to 10 years
▢ over 10 years

2: How familiar are you with the Zettelkasten note-taking method?

Not at all familiar ▢ 1 ▢ 2 ▢ 3 ▢ 4 ▢ 5 ▢ 6 ▢ 7 ▢ 8 ▢ 9 ▢ 10 Extremely familiar

3: How familiar are you with using a wiki for software documentation?

Not familiar at all ▢ 1 ▢ 2 ▢ 3 ▢ 4 ▢ 5 ▢ 6 ▢ 7 ▢ 8 ▢ 9 ▢ 10 Extremely familiar

4: How familiar are you with the two documented services?

Not at all familiar ▢ 1 ▢ 2 ▢ 3 ▢ 4 ▢ 5 ▢ 6 ▢ 7 ▢ 8 ▢ 9 ▢ 10 Extremely familiar

Reading documentation

Please view the documentation of two services organized in a wiki and the same documentation organized using the Zettelkasten method. Answer the following questions:

5: How much understanding could you gain about the system documented using a wiki by viewing the documentation at a quick first glance?

Very little understanding ▢ 1 ▢ 2 ▢ 3 ▢ 4 ▢ 5 ▢ 6 ▢ 7 ▢ 8 ▢ 9 ▢ 10 A comprehensive understanding

6: How much understanding could you gain about the system documented using the Zettelkasten method by viewing the documentation at a quick first glance?

Very little understanding ▢ 1 ▢ 2 ▢ 3 ▢ 4 ▢ 5 ▢ 6 ▢ 7 ▢ 8 ▢ 9 ▢ 10 A comprehensive understanding

7: How intuitive was the navigation of the documentation organized using a wiki?

Not intuitive at all ▢ 1 ▢ 2 ▢ 3 ▢ 4 ▢ 5 ▢ 6 ▢ 7 ▢ 8 ▢ 9 ▢ 10 Extremely intuitive

8: How intuitive was the navigation of the documentation organized using the Zettelkasten method?

Not intuitive at all ▢ 1 ▢ 2 ▢ 3 ▢ 4 ▢ 5 ▢ 6 ▢ 7 ▢ 8 ▢ 9 ▢ 10 Extremely intuitive

9: How quickly could you find the information you need using the wiki?

Not quickly at all ▢ 1 ▢ 2 ▢ 3 ▢ 4 ▢ 5 ▢ 6 ▢ 7 ▢ 8 ▢ 9 ▢ 10 Extremely quickly

10: How quickly could you find the information you need using the Zettelkasten method?

Not quickly at all ▢ 1 ▢ 2 ▢ 3 ▢ 4 ▢ 5 ▢ 6 ▢ 7 ▢ 8 ▢ 9 ▢ 10 Extremely quickly

11: How well did the wiki documentation represent connections between components of the documented system?

Not well at all ▢ 1 ▢ 2 ▢ 3 ▢ 4 ▢ 5 ▢ 6 ▢ 7 ▢ 8 ▢ 9 ▢ 10 Extremely well

12: How well did the Zettelkasten documentation represent connections between components of the documented system?

Not well at all ▢ 1 ▢ 2 ▢ 3 ▢ 4 ▢ 5 ▢ 6 ▢ 7 ▢ 8 ▢ 9 ▢ 10 Extremely well

Creating documentation

After being explained the principles and process of creating documentation using the Zettelkasten method, please answer the following questions:

13: How easy would it be to learn to create documentation using a wiki?

Not easy at all ▢ 1 ▢ 2 ▢ 3 ▢ 4 ▢ 5 ▢ 6 ▢ 7 ▢ 8 ▢ 9 ▢ 10 Extremely easy

14: How easy would it be to learn to create documentation using the Zettelkasten method?

Not easy at all ▢ 1 ▢ 2 ▢ 3 ▢ 4 ▢ 5 ▢ 6 ▢ 7 ▢ 8 ▢ 9 ▢ 10 Extremely easy

15: How quickly could new documentation be added using a wiki?

Not quickly at all ▢ 1 ▢ 2 ▢ 3 ▢ 4 ▢ 5 ▢ 6 ▢ 7 ▢ 8 ▢ 9 ▢ 10 Extremely quickly

16: How quickly could new documentation be added using the Zettelkasten method?

Not quickly at all ▢ 1 ▢ 2 ▢ 3 ▢ 4 ▢ 5 ▢ 6 ▢ 7 ▢ 8 ▢ 9 ▢ 10 Extremely quickly

17: How flexible would the wiki be in accommodating changes and updates to the documentation over time?

Not flexible at all ▢ 1 ▢ 2 ▢ 3 ▢ 4 ▢ 5 ▢ 6 ▢ 7 ▢ 8 ▢ 9 ▢ 10 Extremely flexible

18: How flexible would the Zettelkasten method be in accommodating changes and updates to the documentation over time?

Not flexible at all ▢ 1 ▢ 2 ▢ 3 ▢ 4 ▢ 5 ▢ 6 ▢ 7 ▢ 8 ▢ 9 ▢ 10 Extremely flexible

19: How well would the wiki facilitate collaboration among team members during the documentation creation process?

Not well at all ▢ 1 ▢ 2 ▢ 3 ▢ 4 ▢ 5 ▢ 6 ▢ 7 ▢ 8 ▢ 9 ▢ 10 Extremely well

20: How well would the Zettelkasten method facilitate collaboration among team members during the documentation creation process?

Not well at all ▢ 1 ▢ 2 ▢ 3 ▢ 4 ▢ 5 ▢ 6 ▢ 7 ▢ 8 ▢ 9 ▢ 10 Extremely well

Overall Feedback

21: What are the strengths of the Zettelkasten method compared to other documentation methods?

____

22: What are the weaknesses of the Zettelkasten method compared to other documentation methods?

____

23: Any other comments or suggestions on how the Zettelkasten method could be better adapted for software documentation?

____

24: Would you consider using the Zettelkasten method for your own documentation needs? Please explain why or why not.

____

Conclusion

Thank you for taking the time to complete this survey. Your feedback is valuable in evaluating the effectiveness of the Zettelkasten method, and will help identify any areas where the Zettelkasten method adaptation for software documentation may be improved.

Appendix no. 2

Comparing Software Documentation Methods - Survey Results

Id	How long have you been working in software development?	How familiar are you with the Zettelkasten note-taking method?	How familiar are you with using a wiki for software documentation?	How familiar are you with the two documented services?	How much understanding could you gain about the system documented using a wiki by viewing the documentation at a quick first glance?
1	5 to 10 years	7	9	7	7
2	0 to 2 years	1	5	5	4
3	2 to 5 years	1	10	1	4
4	over 10 years	4	4	4	7
5	5 to 10 years	2	9	9	7
6	2 to 5 years	1	8	7	8
7	5 to 10 years	4	9	3	9
8	over 10 years	3	8	5	8
9	0 to 2 years	8	3	3	4
10	5 to 10 years	1	10	2	9

Id	How much understanding could you gain about the system documented using the Zettelkasten method by viewing the documentation at a quick first glance?	How intuitive was the navigation of the documentation organized using a wiki?	How intuitive was the navigation of the documentation organized using the Zettelkasten method?	How quickly could you find the information you need using the wiki?	How quickly could you find the information you need using the Zettelkasten method?
1	8	7	8	8	8
2	6	8	10	6	8
3	6	8	6	6	7
4	8	7	7	8	8
5	7	9	6	5	7
6	6	9	4	9	4
7	7	8	6	9	7
8	8	8	10	7	9
9	5	3	6	4	6
10	7	10	7	9	6

Id	How well did the wiki documentation represent connections between components of the documented system?	How well did the Zettelkasten documentation represent connections between components of the documented system?	How easy would it be to learn to create documentation using a wiki?	How easy would it be to learn to create documentation using the Zettelkasten method?	How quickly could new documentation be added using a wiki?
1	7	9	8	6	9
2	5	9	9	8	8
3	6	9	8	9	8
4	6	8	7	7	7
5	5	7	10	7	9
6	7	6	9	6	8
7	7	8	10	8	10
8	8	10	8	6	8
9	7	9	7	8	7
10	8	8	10	7	10

Id	How quickly could new documentation be added using the Zettelkasten method?	How flexible would the wiki be in accommodating changes and updates to the documentation over time?	How flexible would the Zettelkasten method be in accommodating changes and updates to the documentation over time?	How well would the wiki facilitate collaboration among team members during the documentation creation process?	How well would the Zettelkasten method facilitate collaboration among team members during the documentation creation process?
1	6	6	5	5	7
2	8	7	6	7	8
3	9	7	10	7	8
4	7	6	8	6	7
5	7	8	8	6	7
6	5	9	6	6	6
7	7	8	8	8	8
8	10	7	9	6	9
9	6	6	9	7	2
10	7	9	8	9	9

Id	What are the strengths of the Zettelkasten method compared to other documentation methods?
1	Structure, map and ease of use.
2	It clearly shows the dependencies of files
3	The graph structure.
4	Information classification
5	I seems that it connects different aspects of documentation together, which could be better for people not familiar with the documentation.
6	The short notes make it easy to read
7	Many ways to find information, if you don't know what you're looking for
8	Better for ad-hoc taking of notes.
9	better overview
10	Some more random information might fit well in this format, as in developer notes.

Id	What are the weaknesses of the Zettelkasten method compared to other documentation methods?
1	Requires more time to add "tags"/"relations" to the documentation, not a lot of developers are used to this method, so it usage might require additional training.
2	if the file needs to be moved to another project all the prior links would have to e deleted
3	The lack of hard rules makes it easier to duplicate entries, or link with the wrong ones, or even not link anything at all. This is a bigger problem for tags, which could allow for multiple spellings. It's also not guaranteed that all related items will be tagged, thus you could miss them thinking it should be under a specific tag.
4	Organizing notes correctly can be time-consuming and challenging.
5	It's quite overfilled with information that may not be relevant in some cases, though the tags help with that a little.
6	Quite difficult to navigate
7	Quite complicated navigation for a first-time user
8	It's not been used in any of my work environments. Probably because the more traditional ways (like wiki) are easier to share with others and publish as complete documents. Documenting various aspects of software often involved more significant amount of text and graphics than one would comfortably fit on a card (...one page).
9	might take time to find the exact item
10	It might take a very long time to set up.

Id	Any other comments or suggestions on how the Zettelkasten method could be better adapted for software documentation?
1	Needs better integration with widely used software as otherwise map view is not visible. AI integrations for auto generating text.
2	Find a way to make the graphical representation look less messy
3	
4	Multiple types of relations between information.
5	Maybe several different trees could be made for different aspects of documentation, so it's easier to search within a particular scope
6	Improve the tools for navigation, other than that, seems viable for small projects
7	Make links to other nodes more visible, and not jump out of tree view after note is clicked.
8	Not sure, sorry.
9	no
10	Make adding documentation quicker - maybe automatically create links to other notes?

Id	Would you consider using the Zettelkasten method for your own documentation needs? Please explain why or why not.
1	Depending on the project, it is good for internal documentation, but questionable for public API documentation. For public documentation it would increase the scope while the end result would be public webpage with the same content: tags, structure, references.
2	Yes, it is more intuitive, makes finding things easier and feels more organised
3	
4	No. It is pretty hard and time-consuming to use Zettelkasten method on existing documentation if it was not prepared as short notes.
5	Yes, I would consider it for personal projects, but it would have to be more complete to use it for work
6	No, it just doesn't seem as convenient as using a wiki
7	I would, but it would require for it to be explained to co-workers as well
8	Yes, if I had a convenient web-driven system for it. I feel it might be quicker, more agile than writing wiki pages, which would inspire myself and more of my colleagues to contribute to the knowledge base.
9	no, it takes too much time and effort
10	No, it doesn't fit my current needs, but might be interesting to try if I have the opportunity.