



A Study Into Ideal Software Development for the Disabled

Santeri Kivimäki

BACHELOR'S THESIS
28.5.2023

Software Engineering

ABSTRACT

Tampereen ammattikorkeakoulu
Tampere University of Applied Sciences
Bachelor of Engineering
Degree Programme of Software Engineering

SANTERI, KIVIMÄKI:
A Study into Ideal Software Development for the Disabled

Bachelor's thesis 81 pages, appendices 1 page
May 2023

The purpose of this study was to survey the problems that the target group – the disabled, might have or encounter in their daily lives. The goal was to delve into first-hand experiences and examine the underlying conditions that have an impact on the accessibility of the software and parallelly providing some solutions to the software usage issues to display software usage ability. Since the first-hand experiences turned out to be too difficult to gather (due to GDPR reasons) an alternative way for data collection was used (Subjective interpretation of recorded user experience).

This study was created within the definition of disability of CDC in mind, while still focusing on the W3C Accessibility guidelines to bring accessibility results to the increasingly more prevalent disability problem across the globe.

In this thesis, an understanding of the ideal software development process was established by combining already existing software development framework paradigms.

Also, various manifestations of disability and impairment were studied, analyzed and conjointly, together with their existing, globally accepted classifications reflected upon software usage regarding various software use stages – resulting in a “User challenges – heatmap”.

With the help of a heatmap it was possible to cherry pick the most severe user challenges, for which in this thesis has been provided various suggestions in how to enhance the so called “ideal” software development framework in a way that would also take into consideration the needs of the disabled. And possibly even to capitalize on these findings when implementing software engineering education.

The thesis is built to provide a chronological step-by-step constructed view of the current situation, the problem, the processes that take place to solve this and the solutions suggested in solving the problem, and the results.

Key words: please use lower case initial letters

CONTENTS

1	INTRODUCTION	6
2	RESEACH IN THIS THESIS	9
2.1	Research Approach	9
2.2	Motivation For Research	9
2.3	Research Question and Hypotheses.....	10
2.4	Research Objectives/Guidelines for working.....	10
2.5	Research Methodology	11
2.5.1	1. Conceptual-analysis research method approach.....	11
2.5.2	2. Decision-making methodological research method approach.....	12
2.5.3	3. Nomothetic research approach.....	12
2.5.4	4. Activity-analytic research approach	13
2.5.5	5. Constructive research approach.....	15
2.5.6	Presented research methodologies used	16
3	THE CONCEPT OF AN IDEAL SOFTWARE PROCESS	18
3.1.1	A Standard Software Process	18
3.1.2	Software development frameworks referenced	20
3.1.3	Kanban, Scrum, and DevOps as a Springboard for an Ideal Software Development Framework.....	22
3.1.4	Ideal Software Development Framework.....	28
4	DESCRIBING SOFTWARE USE LIMITATIONS FROM THE DISABILITY PERSPECTIVE	34
4.1	Disability classifications.....	34
4.2	Types of Disabilities deterring the use of technology and software	36
4.2.1	Visual Disability	36
4.2.2	Auditory Disability	38
4.2.3	Mobility, Dexterity, and Ambulatory-based Disability	40
4.2.4	Cognitive Disability	40
4.2.5	Communication Disability	41
4.2.6	Self-Care Disability	42
5	SOFTWARE STAGES OCCURRING IN EVERYDAY USE.....	44
5.1	Current stages in accessing software	45
5.1.1	Installation	45
5.1.2	Launching software	45
5.1.3	Login to Program	46
5.2	Current stages in using program	46

5.2.1	Navigation	46
5.2.2	Data Retrieval.....	47
5.2.3	Data Manipulation.....	47
5.2.4	Data Saving	48
5.3	Current stages in communications	48
5.3.1	Visuals.....	48
5.3.2	Communication.....	49
6	PROBLEMS AND POSSIBLE SOLUTIONS IN EVERYDAY SOFTWARE USE	51
6.1	Heatmap of disabilities	51
6.2	Problems in accessing software.....	52
6.2.1	Problems with Installation.....	52
6.2.2	Problems with Launching Software	54
6.2.3	Problems with Login/Accessing Service	55
6.3	Problems in using program	56
6.3.1	Problems in Navigation.....	56
6.3.2	Problems in Data Retrieval.....	58
6.3.3	Problems in Data manipulation.....	59
6.3.4	Problems in Data Saving	60
6.4	Problems in user communication	61
6.4.1	Problems with Visuals	61
6.4.2	Problems with Communication	62
6.5	Heatmap of Disabilities Reviewed.....	64
7	TAMK PERSPECTIVE TO THIS ISSUE	68
8	DISCUSSION OF RESULTS	71
	REFERENCES	73
	APPENDICES.....	81

GLOSSARY

WHO	World Health Organisation
ICF	International Classification of Functioning
CDC	Center for Disease Control and Prevention
W3C	The World Wide Web Consortium
GDPR	Guidelines of European Data Protection Regulation
SDLC	Software Development Life Cycle
WIP	Work In Progress
Agile	Project management approach
WG-SS	Washington Group Short Set on Functioning
UNESCO	United Nations Educational, Scientific and Cultural Organization
DO-IT	Disabilities, Opportunities, Internetworking, and Technology
RSI	Repetitive stress injuries
ALS	Arterial lateral sclerosis
TBI	Traumatic brain injury
ADHD	Attention deficit hyperactivity disorder
CDAC	Communication Disabilities Access Canada
WRAF	Work and Family Researchers Network
IEEE	Institute of Electrical and Electronics Engineers
CMMI	Capability Maturity Model Integration
GUI	Graphic User Interface
TUNI	Tampere Universities
RAM	Random Access Memory
ESRB	Entertainment Software Rating Board
RNIB	Royal National Institute of Blind People
VoIP	Voice Over Internet Protocol
Sprint	One cycle of work
Accessibility	Ease of use

Key words: Accessibility, Inaccessible, Disability

1 INTRODUCTION

In today's world of technology, the competition among companies is endless, and there is a continuous stream of new products becoming available. This also applies to software – hence, in this thesis the expression “product” will sometimes be used as a synonym for software.

In a bid to not let the quality of products nor the number of customers drop, a lot of effort is put into optimizing the product's gains and cost-effectiveness by the companies and their separate departments. In addition to continuously upgrading the product over several generations of technology, the groundwork for the products to be sold also requires continuous presence: the most successful of these companies have been required a long-term presence on the market, some of the products that the companies have invented, even becoming household names, for example Microsoft, known certainly by everyone. Over time the continuous iterations of products which carry the company's name, pursue promoting the theme of the company; the idea of the company will be presented in the very essence of the product, further cementing the company's name as a successful venture.

To a company that wishes to stay competitive, this kind of an approach is the one of the better ways of marketing if not the best one. Cheap marketing will be done just by the mere presence of the product. This is most likely an ideal situation for a company where just the quality of the products speaks for themselves. However, this one-sided focusing into certain product often narrows down the development capabilities of the said company. Investing into such depths under one category means that the company often cannot, or rather does not want to spread themselves to other categories of technology, as there is a risk of spreading the finances too thin either by experimenting with new ideas for technology or just risking losing the leading position to another competitor by losing funds in the category of technology there were known in the first place.

However, how many of us has come to think of it, that when pursuing lucrative business, it seldomly leaves room for special target groups, as for example, the disabled, as discussed in the main focus of this thesis.

This same trend is what seems to plague a lot of technological development nowadays. A lot of the development seems to stagnate in the safe areas where most of the profit can be made without risk.

One such area is assistive technology. The core aim for assistive technology is to help those who are impaired or disabled to overcome their difficulties and to be able to function on the same level as those not needing assistive technology. According to the WHO (2018), approximately a billion people in the world still need assistive technology, yet despite the global needs regarding assistive products, access to assistive products remains limited (World Health Assembly, 2018). A billion more products are waiting to be delivered to those who are lacking their assistance. That is a prime example of untapped potential to companies who move after financial gains.

Yet, even after the World Health Assembly in 2018 urged all member states to invest more in the research and development of assistive products, the change still cannot be seen. It is likely that the companies don't see the value in investing in a such endeavor – or perhaps it is because people with special needs are not aligned with the previously described *modus operandi*. It might also be possible that it is just that the companies or the authorities driving the development just don't have the practical experience to deal with the issue.

While working, most sources accessed for this thesis seemed to follow the same kind of already established viewpoints; the existing institutions seem to only refer to each other's resources or provide statistics instead of addressing issue. Most of the information encountered was in the form statistics and numbers and the actual firsthand experiences were very few and far between and there was little talk about actual scenarios that had happened. In other words, it is quite challenging to find actual first-hand experiences.

Even in one of the leading websites about bringing accessibility to the disabled (Stories of Web Users, W3C, 2017) there was only over a handful of stories documented. According to this source, these stories were even entirely fabricated to provide useful scenarios to portray disabilities in different situations. The same

difficulty in obtaining subjective information regarding use of software by the disabled as was ascertained in this thesis appears to be the case with even with the leading authorities in this matter.

The sporadic number of actual real-life situations and statements created a lot of thinking due to the problems that might stay hidden. From a logical standpoint in software development and accessibility, first-hand experiences, and feedback matter.

This thesis, as such, nonetheless aims to delve into the augmentation of some honestly and sincerely of the key concepts in software development and scrutinizing on how well the standard software development processes fare when critiqued by the software needs of those less fortunate, the impaired and the disabled. Who by chance, also would indirectly benefit from this thesis.

2 RESEACH IN THIS THESIS

2.1 Research Approach

This study combines different frameworks of software development and creates a glimpse into the world of those who are disabled or impaired with the intent to find out about how well modern software development fills the needs of the above-mentioned group.

2.2 Motivation For Research

My grandfather had developed the world's first environment control system for the disabled during the end of 1970s. That system was called EAR-100. Even people who were paraplegic could for example, open doors and take an elevator ride by whistling in different pitches to the system that came with EAR 100.

Today – 2023, we live in a time of unprecedented bountifulness, not only in a sense of materialistic abundance, but also in terms of diverse technological breakthroughs, including plurality of software's. Despite this, the disabled people and their operating in modern society is still impeded by insufficient attention to their special needs from software development perspective. In modern society there is a continuous debate on who is a valued member of society.

These objectives regarding efficiency do not consider that humanity is made from people with different capabilities. The main workforce of the population completes their tasks with great results and the average people follow closely behind with satisfactory results, and the rest are those who get the shorter end of the stick i.e., those who are born or made disabled in life, those people suffer the most from the high expectations of society.

With this thesis I intend to contribute at least in some degree to this issue in Software Engineering.

2.3 Research Question and Hypotheses

The research question of this thesis is:

Is the normally ideal line of software development refined well enough to consider the needs of those who are impaired or disabled?

From this basis, the research question can be divided into two hypotheses:

(H1) - Ideal software development does not consider the needs of the disabled.

(H2) – Ideal software framework can be adjusted to better fit the needs of the disabled.

The analyzing of the ideal software framework might spur the creation of new guidelines, perspectives, suggestions etc.

The hypotheses have been grasped by creating a list of research objectives which have been in a chronological manner. The thesis has been built to answer these objectives – hence the reserved objectives serve as the structure for this thesis.

2.4 Research Objectives/Guidelines for working

1. An ideal software development process needs to be identified and presented clearly.
2. Describe what kind of disabilities and impairments there are and what kind of limitations they bring when about to use software.
3. Establish a systematic disabilities classification with necessary categories to be used in this thesis.
4. Define the amount of difficulty the disabilities bring when using software.
5. Establish a software use lifecycle.
6. Combine the categorization of the disabilities and the lifecycle to describe/visualize the problems the disabled have in different phases.

7. Reflect the thesis results in towards TAMK's Software Engineering study program to possibly bring forth additional value adding insights to this line of study.
8. Again, in a quick summary, the pivotal objective is to find out an answer to how a software development process could be further advanced for it to also take into consideration the limitations and the needs of the disabled.

2.5 Research Methodology

In this chapter, the various research methodologies of the thesis will be processed, analyzed, and explained.

The following information will discuss the five various research approaches used in the field of technology and Business Economics (Kasanen et al., 1991). Based on the digital sources located during the thesis development, the research method is described, and a general interpretation for each of approaches is also provided.

Additionally, a practical, applicable example relevant to this thesis is suggested.

This section's content is written parallel to the entire thesis, which explains why the structure of "Research Methodologies" overlaps with the structure of the thesis itself, creating a dialogue between them.

2.5.1 1. Conceptual-analysis research method approach

Conceptual analysis as a research method takes e.g., in written form expressed terms and sentences, interpreting, comparing, and creating a presentative structure, involving all the essential contents and meanings of the original source data (Lammintakanen J. et al., 2008, Käsiteanalyttisen tutkimuksen soveltaminen terveyshallintotieteessä - esimerkkinä Rodgersin käsiteanalyysimenetelmä).

In simpler terms, the conceptual-analytical research approach seeks to describe existing theory and its structural components. It can utilize any sensory activity combined with the concept structures produced by the researcher's mind.

Example: When contemplating the ideal software development process, significant amounts of theory and concepts arise. These may be mostly unknown to someone unfamiliar with the subject, but they can be presented and opened to the reader through examples and illustrations.

2.5.2 2. Decision-making methodological research method approach

Decision-making methodology aims to determine what should be considered as the ideal course of action – in each context. It assists researchers in the decision-making process, aiming to answer to the question, how to proceed or act in regard to – for instance, a problem at hand (Soininvaara S., 2018, School of Business and Management).

In simple terms, the decision-making methodological research approach utilizes theory, drawing from it a descriptive (normative) (Cambridge Dictionary, 2023, Normative Definition) generalization that can be used to provide a recommended course of action in a specific situation/problem.

Practical example: For instance, when there is a need to make a recommendation on how to further improve an ideal software development process, taking certain aspects into account, providing an answer to such a question can be interpreted as decision-making methodological work.

2.5.3 3. Nomothetic research approach

Nomothetic methodology aims to provide generalized assertions that refer to greater macro level, yet unorganized data. This can be obtained via systematic, repeated observation in a bid for applicable outcomes (Greelane, 2019, Mitä

nomoteettiset ja idografiset tarkoittavat ja miksi niillä on merkitystä tutkimuksessa).

In simpler terms, when using the nomothetic research approach, the goal is to describe phenomena or situations based on empirical evidence as clearly as possible.

Practical Example: The aim is to provide a clear description of practical-level problems that arise in the daily use of computer programs for individuals with disabilities, while considering the life cycle of program usage.

2.5.4 4. Activity-analytic research approach

Activity-analytic methodology is referred to when the focus concerns complex problems that are in a disarranged state and that the actual object of study appears as complex, wide issue to grasp at once. In the methodology, the issue often requires activity-analytic and explanatory actions (Olkkonen, 1994, pages. 72-74).

In less abstract terms, the activity-analytic research approach is suitable when the phenomenon under study is closer to implicit rather than explicit. To fully grasp the phenomenon, it requires activity-analytic practices, where multiple diverging and converging phases repeatedly intertwine back and forth, forming spirals of knowledge processing that take start during the action and stop during the introspection of the results. With each spiral, implicit knowledge evolves towards explicit knowledge (from unclear to clear), creating a better version with each cycle.

Example: By posing the question of how to comprehensively define accessibility challenges, one can find a lot of unanswered questions. Finding answers is a long process. For instance, it could involve investigating the different perspectives of program life cycles and the various categories of disabilities. As these aspects are gradually brought together and continuously reflected upon, what observations are made regarding the challenges faced in each phase of program usage for each disability?

2.5.5 5. Constructive research approach

In constructive methodology the researcher capitalizes on already previously existing theory as a basis, upon which the researcher's construction is conjointly "welded", thus creating a higher level of outcome (Virtanen A., *Konstruktiivinen tutkimusote – Miten koulutus ja elinkeinoelämän odotukset kohtaavat ammattikorkeakoulun opinnäytetöissä* and Lukka K., *Konstruktiivinen tutkimusote*, 2001).

In simpler terms, the constructive research approach can be illustrated with an analogy: when building a house, one needs to know which elements and materials can be utilized in the construction and in what order. In the constructive research approach, the researcher investigates what the materials are (information, artifacts) and in what order (construction) they should be utilized.

Example: If one wants to create a clear, analytical structure for a thesis, it can be helpful to articulate the research objectives/guidelines in a chronological and logical progression that leads the thesis, culminating it in the final of the thesis being completed.

2.5.6 Presented research methodologies used

This chapter has been filled after the work is complete to complete the dialogue of the work and the research methodologies used with practical examples. Each research approach will be presented with one practical example from the thesis.

Practical example of Conceptual-analysis research approach: Explaining and visualizing the formation of the ideal software process (Ref: 3.1.3 Kanban, Scrum, and DevOps as a spring for an ideal software development framework). The formation of the new information and models can be counted as Conceptual analysis.

Practical example of Decision-making methodological research approach: For instance, when there is a need to make a recommendation on how to further improve an ideal software development process, taking certain aspects into account, providing an answer to such a question can be interpreted as decision-making methodological work.

Practical Example of Nomothetic research approach: The aim is to provide a clear description of practical-level problems that arise in the daily use of computer programs for individuals with disabilities, while considering the usage of software programs.

Practical Example of Activity-analytic research approach: By posing the question of how to comprehensively define accessibility challenges, one can find a lot of unanswered questions. Finding answers is a long process. For instance, it could involve investigating the different perspectives of program life cycles and the various categories of disabilities. As these aspects are gradually brought together and continuously reflected upon, what observations are made regarding the challenges faced in each phase of program usage for each disability?

Practical Example of Constructive research approach: If one wants to create a clear, analytical structure for a thesis, it can be helpful to articulate the research objectives and guidelines in a chronological and logical progression that leads the thesis, culminating it in the final of the thesis being completed.

The following visualized box diagram describes the used research approaches under their respective boundaries of research approaches (Ref: 2.5.1-2.5.6, respective research approaches)

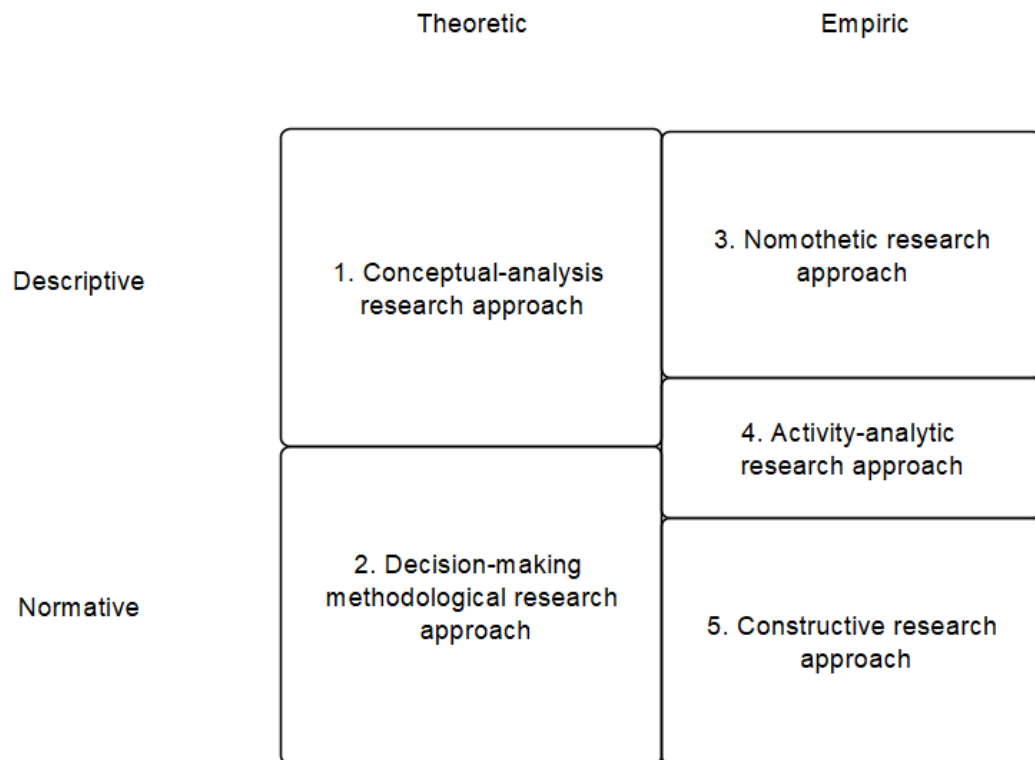


FIGURE 1 - Presented Research Methodologies – Following the ideas of Kasanen et al. 1993

Following the presented research methodologies used, the thesis will move to present the next part, software development framework as a subject.

3 THE CONCEPT OF AN IDEAL SOFTWARE PROCESS

One of the main goals of this thesis is to find out what an ideal software development process is. To reach that goal, we must first address the idea of a standard software development process.

3.1.1 A Standard Software Process

Software is defined as a collection of computer programs, procedures, rules, and data (Britannica, 2023, Software Definition). These building blocks all contain input information in the way that the machine or program in question requires for it to be understood. This information required by the system is known as code. It comes in different varieties which all have different rules; syntaxes they are called, that determine the working order in how it must be inputted.

Should all the modules, parts and operations be created and coded according to all criteria that the syntax and the machine/program requires, the result should be functional. Overall, trying to fill these conditions that are required for functionality are easy in a sense as everything must go according to the rules; there are no exceptions. However, even the smallest overlooked problem could cause the program or machine to cascade into a series of errors, rendering the whole product unusable at the very least to some degree until a fix is found.

Building a program from scratch can be attributed to the above-mentioned idea in a general sense, yet most often the building blocks are the same or recurring in different versions. The same building blocks are seen in many processes which have over time developed to become so essential to the functionality and thus, the usefulness of all data systems that without them they would be crippled (GeeksforGeeks, 2023). They are listed below with examples:

1. Data storage and retrieval: saving or accessing any information in long term would be impossible.
2. Data processing and manipulation: information could not be modified, and it would come in read-only packages.

3. Navigation: without guiding the user between multiple functions the programs would all have only one function.
4. Communication: moving information between the user, the system and its many modules would be impossible without this.
5. Security and access control: without access control data security cannot be achieved.
6. Reporting and visualization: if data systems did not provide logs when something happened tracing how information moved would be hard.
7. Automation and scripting: all higher levels of programming could not be achieved without scripting and its automation.

The way in which these processes are achieved and built, that differs based on the framework, but most of these functionalities if not all of them are the bare minimum for a software program. Everything else is built on top of these functions, so this can be said to present in every software process.

These bare functionalities that are present in almost all features are the basis for all software development processes. However, these bare functionalities are merely a part of the software development process.

One of the well accepted methods in software development which involves the bare necessities is the Software Development Life Cycle (Clark H., 2022, The Software Development Life Cycle (SDLC): 7 Phases and 5 Models). It involves seven stages of software development which seem to occur on most if not all software development processes, making such as a method more or less the standard as producing software in the long run would be hard without these steps.

The seven phases are as follows:

1. Analysis and planning; specifying needs and budgets
2. Requirements; documenting what is needed
3. Design and Prototyping; creating testable software and further upgrading that
4. Software development; creating the actual software with the bare necessities
5. Testing; quality testing of the said software
6. Deployment; pushing out the software product and asking for feedback
7. Maintenance; further refining the product and ironing out any bugs.

Depending on the inspected model the number of phases varies, but it seems like that the 7 phases in question are recurring even if the order is different. The central idea behind the software development stays the same between all the models despite the diverging production methods. The longer the development has advanced into the process (in short, the closer to being ready the product is), the ficker the boundaries between the phases become; the product ends up an intertwined product of multiple phases.

The software product result can vary immensely due to the different requirements from the software. The different results can be attributed to the Software Development Life Cycle which touches upon so many wide subjects in the development process, giving it the potential to become the standard even if it is not the standard yet.

Thus, the introduced Software Development Life Cycle's 7 standard phases will be used as a standard for thesis.

3.1.2 Software development frameworks referenced

To proceed with the standard software development process as a base, the main idea of this chapter is to define what kind of a software process would be ideal. To present this idea, three commonly used software frameworks were reviewed. The choice of frameworks is based on the material given during the study program of Software Engineering. The choices from the study program material line up with literature.

In Hati's (2023) article, the focus of software development methodology is on the Agile framework, listing many of its positive traits. Hati states that "unlike other development tools, a framework for software development will deliver you a pre-defined code to ensure a seamless process." Should the development goal align with the software development framework, it is a beneficial choice to pick one rather than to be without one (Hati S., 2023, Top Software Development Frameworks for 2023)

A part of the just mentioned Agile frameworks' success was also found in other articles. Forbes, one of the biggest newsletters focusing on innovation, lists a part of the Agile framework, the Scrum methodology multiple times in its article, proving its apparent popularity (Expert Panel®, 2023, 13 Tech Leaders Share Their Favorite Software Development Life Cycle Methodologies)

However, it is not just the Agile framework getting all the attention. While there are mentions of other development frameworks such as the Waterfall model or the Spiral model which are not part of the Agile model, two other outliers such as these frameworks stand out.

A paper of Lombardi (2022) mentions DevOps. It's a combined abbreviation of the words: "development" and "operations". It's a development framework which focuses on unifying the work of the usually separate teams in a way, which could be described as two teams working together as one, one on the development side and on the maintenance side (Lombardi P., 2022)

To address the last framework out of three, we should go back to the already mentioned Forbes (Expert Panel®, 2023) article. Kanban, one of the development frameworks mentioned also shines in its own way in software development despite at first being created for car manufacturing (Kanbanize, 2023). Kanban's mandated trust and self-initiative across all units in development reduces wasted time, creating a more efficient workflow. In the article, Kanban as a method was viewed as an "Adult Scrum" – methodology.

The other sources also give similar results but still are not limited to the above-mentioned examples as there are other sources (Kirvan P., 2022, Jory MacKay, 2019; Iordanidis J., & Bland M., 2021, Synopsys, 2017; The Upwork Team, 2020; Nagar T., 2022). However, the 3 of the most recurring software development frameworks were Kanban, Scrum, and DevOps. These three software development frameworks will be used for reference in the next part of the thesis for determining the ideal software development process.

3.1.3 Kanban, Scrum, and DevOps as a Springboard for an Ideal Software Development Framework

The three different frameworks have been quickly explained below with the baseline categories of Software Development Life Cycle (SDLC). Despite the changing various stages in different types of SDLC sequences, one suitable one has been picked to help form the seven different categories for the evaluating the differences between the frameworks (The Upwork Team, 2020).

	Kanban	Scrum	DevOps
1 Basic mode of function	Pull-method management. Production based on customer demand with no additional cost.	Project management framework with its tools. Usually, 10 people are in a team and the amount total teams vary.	A set of practices, tools, and philosophies. Heavily emphasizes team empowerment and cross-communication
2 Requirements for use	Visualized workflow, freedom to act towards the objective on all levels.	Understanding Agile and its events: Sprint, Sprint retrospective, Sprint review, and Sprint planning	2 Differently specialized teams, Fluid communication channels, DevOps environment tools.
3 Design/Use principle	Understanding workflow and Implemented task lists such as, To-do, In progress, and Done.	Small increments over sprints with constant re-adjusting to direct the work effort to tackle current goal.	Continuous integration feedback loop between two specialized teams; one develops, one tests. Information is transmitted between teams as needed.
4 Software development method	Using the task lists to redirect working efforts towards needed objectives.	Small increments over sprints with constant re-adjusting to direct the work effort to tackle current goal.	Developer team creates the data, Operations team tests it and sends feedback to Developer team until the product is ready.
5 Testing implementation	If the testing was not a success, the achieved task will be put back to under "In progress" category.	At the end of a Sprint a retrospective takes place. The work efforts will be reviewed there with the intent to fix them.	Operations team gathers information on deployment and sends feedback to Developer team for further refining.
6 Deployment	Continuous small iterations	If there were no problems found during Sprint retrospective the update to software will be deployed	Operations team gathers information on Deployment and sends feedback to Developer team for further refining.
7 Maintenance and updating	Based on customer demand	Over continuous iterations	Over continuous iterations.

The three frameworks represented in the table have some similarities here and there across the board if looked merely from above. However, there are small nuances to their modus operandi; their mode of operation as software development frameworks can be either wondrous for the chosen purpose or terrible to

the point of regretting, so clarifying the pros and cons of each are required to get an accurate understanding.

To keep up with the idea of the thesis being chronological as stated in chapter 2.3 (Ref: Research Questions and Hypotheses) the pros and cons will be presented in an orderly, chronological manner.

Pros vs. cons of basic mode of function of the frameworks

	Kanban	Scrum	DevOps
1 Basic mode of function	Pull-method management. Production based on customer demand with no additional cost.	Project management framework with its tools. Usually, 10 people are in a team and the amount total teams vary.	A set of practices, tools, and philosophies. Heavily emphasizes team empowerment and cross-communication

Kanban prefers to use the pull-method for managing the issues. In software terms, this is actualized through creating bottlenecks and limiting Work in Progress (WIP) – tasks to maximize output towards the smaller number of tasks. This can be either really cost efficient or create inefficient solutions if too many orders occur at once. Spilling a lot of effort into many tasks at once creates slow progress. It is good for efficient production and bad at sudden changes (Kanbanize, 2023).

Scrum utilizes the same idea as Kanban but with a laxer idea, often preferring to change directions in software development workflow every two to four weeks. Scrum deals with problems quickly due to the small intervals between sprints, while being bad at showing progress as the results trickle in small amounts after each sprint (Scrum, 2023).

DevOps does not function with sprints (at least by default), but still operates like Scrum, in a repeating manner, using two or more teams instead of one for cross-team co-operative development. DevOps can be bad at changing directions suddenly in software development, as the development progress relies on multiple teams, making organizing change troublesome. However, it still shines utilizing large masses of workforce for quicker results than the other frameworks. (Atlasian, 2023).

	Kanban	Scrum	DevOps
2 Requirements for use	Visualized workflow, freedom to act towards the objective on all levels.	Understanding Agile and its events: Sprint, Sprint retrospective, Sprint review, and Sprint planning	2 Differently specialized teams, Fluid communication channels, DevOps environment tools.

The requirement for use is second. Kanban aims to visualize workflow, so understanding the workflow is required. It can either be easy to or hard to adapt for use depending on the amount of ongoing software processes. (Kanbanize, 2023).

Scrum and DevOps both aim to also improve workflow in their own way. Just like with Kanban, one must study and understand it. To use Scrum and DevOps, communication, understanding roles and the tools provided are necessary (Scrum, 2023 & Atlassian, 2023)

	Kanban	Scrum	DevOps
3 Design/Use principle	Understanding of workflow moves into Implemented task lists such as, To-do, In progress, and Complete.	Small increments over sprints with constant re-adjusting to direct the work effort to tackle current goal.	Continuous integration feedback loop between two specialized teams; one develops, one tests. Information is transmitted between teams as needed.

The Design/Use principle is the third. Kanban uses the understanding of workflow to place problems into footnotes, signifying the progress of the task issued to the footnote which is moved as creation of the solution progresses. The main use which Kanban is good at highlighting bottlenecks which it aims to solve by redirecting workflow with the footnotes. The problem with this approach is that too many problems at once will overwhelm the footnote issue system, forcing a lot of demand on the workforce (Kanbanize, 2023)

Scrum and DevOps both use continuous iterations (Merriam-Webster, 2023) to reach the goal in smaller steps, continuously providing small visible progress at the end of each cycle. Scrum uses Sprints for defining the iteration length, DevOps have varying lengths between the iterations, and can adopt Agile practices (like Scrum Sprints) should it be necessary. The continuous iterations produce accurate results with little waste of work but fail to produce any visible results fast (Atlassian, 2023 & Scrum, 2023).

	Kanban	Scrum	DevOps
4 Software development method	Using the task lists to redirect working efforts towards needed objectives.	Small increments over sprints with constant re-adjusting to direct the work effort to tackle current goal.	Developer team creates the data, Operations team tests it and sends feedback to Developer team until the product is ready.

Up fourth is the software development type/method.

In software development Kanban uses the footnotes priorly mentioned footnotes. The footnote issues often go back and forth until a wanted form of software is processed and only then can it go on and continue for further use. This is how the refinement usually happens in software should Kanban be used as a base (Kanbanize, 2023).

Scrum and DevOps opt again for the same kind of approach. Both have their adjusted time intervals to produce versions of small software iterations which are discussed with are either approved or disapproved within the team for further use. The good with approach within software development is that problems can dealt with quickly and backtracking in progress does not deal as much as damage as usual. The bad side to continuous iteration is that not a lot of time is allocated to one task, so it's possible to miss details that require more time than allocated. (Atlassian, 2023 & Scrum, 2023)

	Kanban	Scrum	DevOps
5 Testing implementation	If the testing was not a success, the achieved task will be put back to under "In progress" category.	At the end of a Sprint a retrospective takes place. The work efforts will be reviewed there with the intent to fix them.	Operations team gathers information on deployment and sends feedback to Developer team for further refining.

5th, Testing implementation.

Kanban implements testing by forcing the issues go through testing; in this case the footnotes for software which reach the success category (“Complete” or equivalent) in progress are tested if they work. If the footnotes do not work, they are returned to under to Work-In-Progress category for further development. This testing can quickly stall and create a bottleneck if the development is too long, but it is still good for solving smaller tasks quickly (Kanbanize, 2023).

In addition to standard testing that happens inside the working environment for quality control, Scrum and DevOps both also rely heavily on feedback either from customer or other developers to address issues at the end of an iteration cycle. This can be good for quick feedback, but continuous gathering on feedback can be stressful on communication teams and the privacy of others (Atlassian, 2023 & Scrum, 2023).

	Kanban	Scrum	DevOps
6 Deployment	After success deployment takes place	If there were no problems found during Sprint retrospective the update to software will be deployed	Operations team gathers information on Deployment and sends successful data to be deployed

The second last category, Deployment.

All three categories deploy the code forward if the software is fleshed and functional enough to be considered done. The successful criterion for being deployed successfully is that the software works and the information on the state of the software after deployment can be collected for further analysis and maintenance.

	Kanban	Scrum	DevOps
--	--------	-------	--------

7 Maintenance and updating	Based on customer demand	Over continuous iterations	Over continuous iterations.
----------------------------	--------------------------	----------------------------	-----------------------------

Last but not least, Maintenance and Updating.

Kanban, Scrum, and DevOps deliver the results in different ways, but the Maintenance and post-launch work of updating the software has no clear rule besides following the same pattern for delivering software as explained prior to this chapter. It is up to debate for the financial return vs wanted product longevity of the product on how long the maintenance and updating will take place.

The seven aspects provided in the table for the three frameworks have been covered. Next up, the ideal software process will be assembled with the help of these frameworks.

3.1.4 Ideal Software Development Framework

The “ideal” in the ideal software development framework would imply that the standard quality would be close to a greater aim than the known standard, close to, if not perfect (Merriam-Webster, 2023, Definition of Ideal).

Considering the popularity of the useful confirmed frameworks prior to this chapter of thesis (Ref: 3.13 Kanban, Scrum and DevOps as a springboard for an ideal software development process) the efficiency of the frameworks is evident. It would be hard for a poorly optimized development process to gain that much recognition.

Those frameworks (Kanban, Scrum and DevOps) are the sum of their parts. In practice, this would mean that the parts can also be changed to completely change the outcome of the development process. Implicitly, this would imply that the parts can also be detached, moved, and rearranged to form a completely new process. Thus, the parts of those already recognized frameworks will be used to construct an ideal software development process.

Considering that the thesis is searching for an answer for the hypothesis number one (Ref: 2.3 Research Questions and Hypotheses, (H1)) – to find and answer on whether or not the ideal software development process will take into account the needs of the disabled, the building of the ideal software development should try to aim an ideal software with the needs of the disabled in mind to get as close and accurate to the goal as possible.

To quickly get an answer, fitting parts from the already processed pros and cons of the seven highlighted phases from frameworks in the Software Life Development Cycle perspective will be chosen to build an ideal software development process (Ref: 3.13 Kanban, Scrum and DevOps as a springboard for an ideal software development process).

The choices and explanations about the choices are based on the subjective understanding collected during the Software Engineering training at Tampere University of Applied Sciences to apply on this thesis and should one want to extend or alter the criteria, then it's a topic for another undertaking.

1 Basic use of function.

To consider the needs of the disabled, the basic mode of function should minimize the amount of required communication as there might be difficulties in that area. The option out of the three would be the pull-method based Kanban as it would only be required when needed. The other frameworks do not provide a good solution as possibly required feedback from customers after each iteration cycle in the other development models might just be too much communication required should the target group already have problems within that area.

2. Requirements for use.

For a solution to minimize the number of requirements, the threshold for use should be low as the solution is geared towards the disabled who already might have problems in using software. Kanban is not a good fit for a minimal threshold as the very principle for Kanban to be used to is to visualize the workflow. That would in essence mean doing a complete study in the disabilities of the customer to understand the required workflow. That would not be unfit for the minimal use requirements, but also a massive breach of privacy. For GDPR reasons, doing the development of software in slow, less-invasive small bites would be a better option. So, either one of the iteration-based frameworks, Scrum or DevOps are a better fit.

3. Design/Use Principle

Kanban would use broad categories of the footnotes to approximate progress for the software, Scrum and DevOps use inter-team and customer-based feedback after the iteration cycle to address the problems found. The more responsive way of communicating in-between each iteration cycle of Scrum and DevOps is better suited to advance the goal of software development for the disabled. Kanban's generalized footnote task categories leave less to communication and put more emphasis on the development side, leaving the only available option being either Scrum or DevOps for an ideal software that meets the needs of the disabled users.

4 Software development method

Kanban would fit in brute-forcing the development issues in known software problems, due to its usage of limiting the Work-In-Progress footnote issues. Scrum and DevOps are better at considering multiple angles of a problem and making slow and sure progress instead of brute forcing the solution. The software for the disabled is a fairly unknown territory as even the WHO would only consider trying to make the known technology more affordable instead of creating new ones according to one of their statements released (WHO, 2023). Considering this, from the financial perspective Kanban would be better to expedite financial gains, but in the long run Scrum or DevOps would be more fit to push breakthroughs slowly but surely as their strengths lie taking the software development slowly and from multiple angles while chipping away at the problems.

5. Testing implementation

Testing in this matter requires prudence and customer feedback without bothering the target group, namely the disabled, too much while still getting results. A hands-on practical approach is required to get accurate feedback data from use, yet too much testing cannot be done to not breach privacy. Accessing the assistive technology in use which require software to work can be painful for the user

should it be accessed to recover the testing data, for example detaching a hearing aid device from the ear. None of the reviewed frameworks provide a direct solution to this kind of a problem as the testing was intended to be easily accessed without considering other factors. DevOps could employ other specialized teams to do the field testing carefully for the target group and risk trouble to get the testing results back to the software development team, but even then, the scenario could hardly be called ideal due to the amount of difficulty encountered.

6. Deployment

As a basic functionality DevOps is intended to work in two or more development teams, so it could on some level work if the legislation of GDPR is not breached. Only DevOps could have the possibility on some level to return the successful test data and deploy the successfully tested software. The other frameworks cannot achieve this as they failed in the earlier step of 5. Testing.

7. Maintenance and updating

For maintenance and updating the software, Kanban would be the best for privacy reasons as the updates would only be coming in only when the customer willed for the software updates to occur. However, from the accessibility perspective, Scrum and DevOps would be the better option as the software development department would stay in touch with the customer at the end of each iteration, ensuring that no problems would occur. Thus, the better choice depends on the scenario, leaving the clear choice unpicked.

Within the allocated capabilities of the assessed frameworks the ideal assembled software framework would be as follows:

- 1 Basic mode of function – Kanban pull method.
- 2 Requirement for use – Scrum/DevOps based iteration cycles.
- 3 Design/Use principle – Scrum/DevOps communication principle.
- 4 Software development methods – Scrum/DevOps long term development
5. Testing implementation – DevOps multi-team approach for testing
- 6 Deployment – DevOps, implying testing is accomplished.
- 7 Maintenance and updating – Repeated problems due to no good protocol.

The ideal software development process would try to use ready-made parts of the frameworks to their fullest potential while aiming for maximized power regarding producing disability software.

The produced result was put together with disability software creation in mind for maximum efficiency to reach the ideal software development process. Despite choosing the software development most suited for reaching the goal of developing software for the disabled at maximum efficiency, the methodologies included in the three frameworks started to stagger when presented with the challenges that comes with practical testing as there are no suitable protocols in place to deal with the situation.

Even in the ideal scenario where DevOps could pull off the field testing and deploy the field data, the same scenario would occur many times when maintenance and updating of the software takes place, further reducing the chances for long term success. This ideal scenario for the ideal software development does not even consider the different types of disabilities, so it is extremely unlikely for it to find success in developing software. Should the ideal software development process want to consider the needs of disabled, it is unlikely that it would pull through as even before introducing multiple types of disabilities to consider, the increased difficulty is almost enough to render the framework useless in regard to developing software for the disabled.

Ultimately, to answer the hypothesis number one: (Ref: 2.3 Research Questions and Hypotheses) (H1) - Ideal software development does not consider the needs of the disabled is true.

Due to difficulties encountered during this review, it can be noticed that the ideal software development framework provided in this can overcome superficial problems in relation to disability challenges to some degree but falls apart when introduced to greater difficulty due to the fundamental differences in requirements.

During the rest of the thesis, the study will focus on opening the disability types for the reader, explaining the problems related to the disability types with some practical examples, and suggesting some solutions to overcome those problems which were presented to see, if the hypothesis two (Ref: 2.3 Research Questions and Hypotheses) (H2) – Ideal software framework can be adjusted to better fit the needs of the disabled.

4 DESCRIBING SOFTWARE USE LIMITATIONS FROM THE DISABILITY PERSPECTIVE

Here the disability perspective is described by introducing the disability categories. Later on, the impact of the disabilities on the use of software will also be described

4.1 Disability classifications

To identify the limits that different types of disabilities cause, we must first go through the disability classification types to understand the types of disabilities themselves.

For the purposes of this thesis, focusing on the classifications which cover most of the disability categories was the best choice as the greatest coverage would also give the greatest percentage of accuracy for the results.

According to the CDC definition, disability is a broad concept that covers but is not limited to impairment, activity limitation, and participation restriction.

Impairment is a problem in body function or structure; an activity limitation is a difficulty encountered by an individual in the execution of a task or action; while a participation restriction is a problem experienced by an individual involved (CDC, 2020, Disability and Health)

To analyze the software aspect of such a complex and ubiquitous phenomenon which can be found on all levels of life, the next step is to cut down the problem to small enough sizes to analyze the problem in detail but to still not overdo it.

A lot of other parties involved in this problem have spent many decades trying to efficiently tackle problem, and one of the created ways to identify disabilities was good enough to be accepted as a standard. This thesis also aims to make use of that same standard for quick identification and categorization.

The Washington Group created the Washing Group Short Set on Functioning (WG-SS) of the year 2022, which is a questionnaire for quick identification on

basic level. It is deemed to be the best instrument by the practitioners of UNESCO (UNESCO, 2023, Disability).

For countries or individuals wanting to collect more in-depth information, the Washington Group also developed a 36-point set of questions (2022, WG-SS Enhanced), which is the more advanced part of the survey.

The six-point questionnaire (WG-SS) covers visual, auditory, motor, cognitive, self-care and communication capabilities, helping the mapping, diagnosing and development of technology for the disabled. These six core areas cover most of the disabilities. The individual might be affected by more than one of the areas on the list. In this scenario this questionnaire provides an adequate basis for further analyzing, so it will be used as a baseline for this thesis (The Washington Group, Short Set on Functioning (WG-SS), 2022)

According to the WHO, 15% of the population experience some form of disability. In other words this means that a population group of roughly 1 billion people suffer from this problem (WHO, 2023, Disability).

Those whose lives are marred by disabilities in their daily lives do not get to use technology as much as those who are not disabled. During the 71st assembly of WHO regarding disability (2018), it was said that only 10% get to use assistive products on par with their needs, yet there are billion people worldwide who won't get to use assistive technology according to their needs. This does only take account on the availability of the products, not the software aspect. The increasing population count also means that the already known one billion disabled people will be growing more than twofold to 2 billion or more by 2050 (WHO, 2018, Improving access to assistive technology). The increase of the disabled subpopulation will cause the issue of disability and that way, the alternative ways of using software becoming a more prominent problem than it currently is if not addressed.

In a bid to address this growing problem, the next part in the thesis will analyze on how the WG-SS categorized disabilities will impact software usage.

4.2 Types of Disabilities deterring the use of technology and software

In this section the different types of disabilities that affect people's lives are explained in depth. Here the six types of disabilities which affect a person's daily capabilities to use any software are clarified.

4.2.1 Visual Disability

Visual impairments differ from each other in severity. The ones that affect the capability to use technology and software can vary from divergent forms of color-blindness, bad eyesight from aging, visual organ degeneration, or from the structural differences that cause the eye to refract light in a wrong way, to great or complete loss of vision due to cataract (Disabled World, 2022).

On the global level, there are 2.2 billion people who are affected by near or distance vision impairment. Of those 2.2 billion people, 1 billion of them are moderate to severe cases. A great majority of the 1 billion cases are from aging-related unaddressed presbyopia, which is followed by unaddressed refractive errors and cataract. In terms of regional differences, the vision impairment rates are much greater in lower-income areas. In low- and middle-income regions the difference is estimated to be four times greater with distance impairment. The prevalence of near vision, however, is even more prevalent. The rate of unaddressed near vision impairment is estimated to be greater than 80% in the populations across western, eastern, and central sub-Saharan Africa, while the comparative rates in high-income regions of North America, Australia, Western Europe, and Asia-Pacific are reported to be lower than 10% (WHO, 2022)

In addition to the refractive related problems in the eye, many categories of color-blindness exist. According to the National Eye Institute (2019), color blindness can take hold in a person if: the optic nerve or the retina is damaged, often in the case of glaucoma developing slowly over the years or the macula degenerating. Also that which affects the brain and the nervous systems can damage the ocular organs, like Alzheimer's or multiple sclerosis (body tissue calcification) and even

medicine if prescribed and taken incorrectly. This is also linked to the recommendation of doctors that watching a screen is not recommended as even the eyes require a break; longer repeating intense sessions can leave lasting damage, especially in low-light conditions. (National Eye Institute, 2019)

The current state of technology has created several ways with visual disabilities to use technology and software. Most often the computer is typically navigated using a keyboard or some other kind of equivalent, like braille displays, speech input or screen readers (Washington DO-IT, 2022).

Windows Narrator is one example of the screen readers programs that can assist an individual's uptake of information with a weakened visual sense through also utilizing other senses in the progress, namely in this case the auditory sense. Windows Narrator utilizes text and information recognition technology to translate all relevant event on the screen from text to speech. This speech is provided in tandem with a blue, high contrast box to draw in and limit the focus of the user while reading the text and information in a language of their choice, reinforcing the chances that the information reaches the user (FIGURE 2. Windows Narrator start screen).

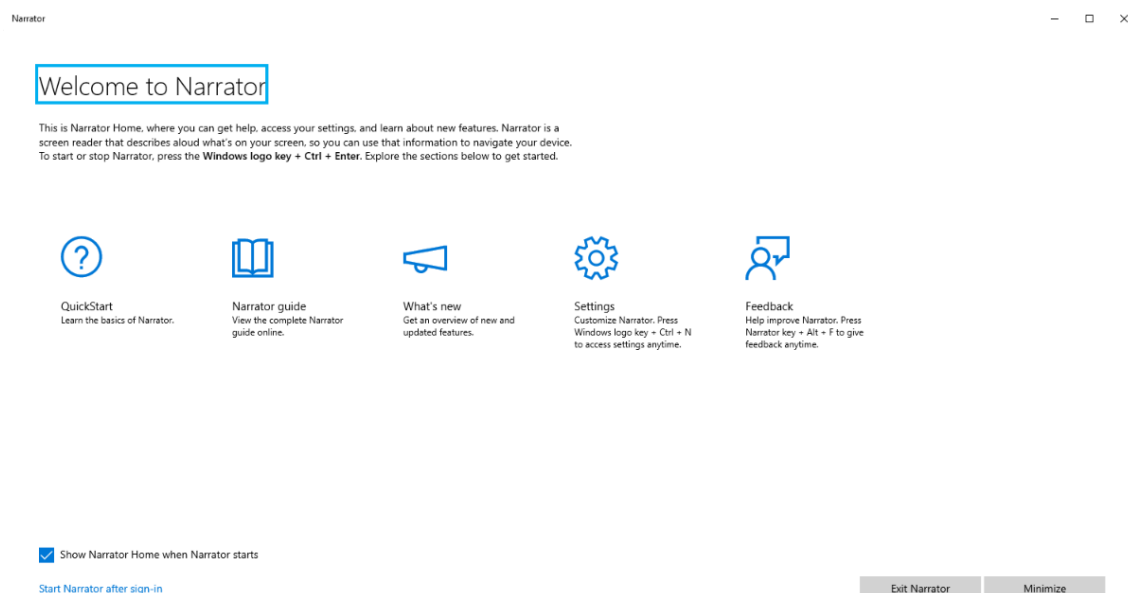


FIGURE 2. Windows Narrator start screen.

The small text size will be a challenge to many, especially older people. A basic form of a Magnifier is provided as an accessibility feature in Windows to help with that issue. 3rd party forms of the software also exist. The function of the Windows Magnifier is to enlarge the screen contents up to a factor of 16 should the user choose to do so to help see the text better. (Fable Insights, 2021 & Fable Team, n.d.)

For Windows users, there are built-in pre-sets for different types of color-blindness. These pre-sets override everything on the screen to match the color-blind-

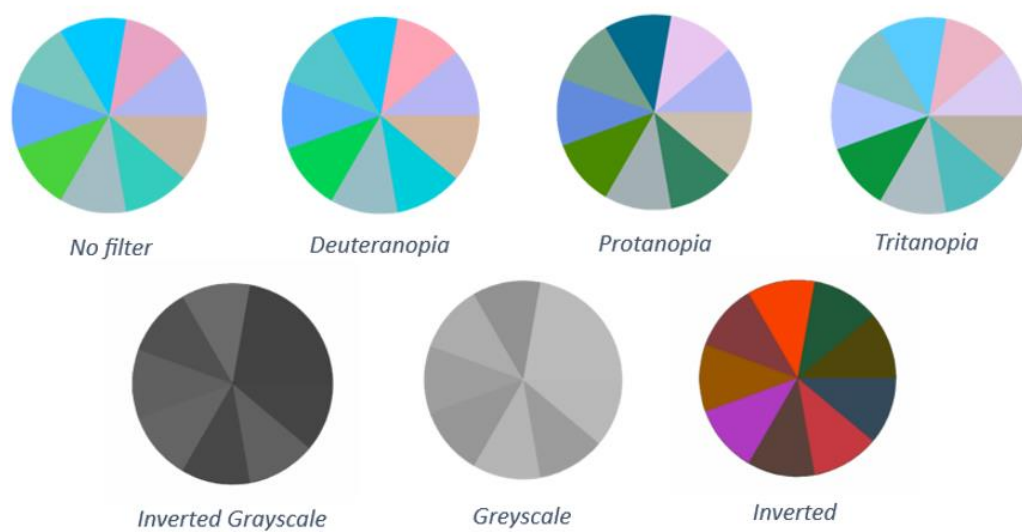


FIGURE 3. Color filters in Windows' color settings

ness, assisting the differentiation of colors for those who require it. With the red-green (green weak, deuteranopia), red-green (red weak, protanopia), and blue-yellow (tritanopia), inverted (reversed color on screen), grayscale (variety of gray shades for uniform color) and grayscale inverted (inverted gray) (FIGURE 3.)

4.2.2 Auditory Disability

Hearing loss is the reduced capability in one or both ears of being able hear what people say by -20 decibels. For someone to be recognized as someone with a disabled hearing capability, their hearing range is reduced by -35 decibels. Approximately 432 million, 5% of the population world-wide is affected by hearing loss. The prevalence of hearing loss and hearing disability increases with age. It is also more frequent in the developing countries. These estimates are predicted

to increase as the population also increases, further increasing the inability to communicate with others through software. (WHO, 2023).

Known causes for hearing loss and auditory disabilities are caused by but not limited to: genetic factors, damage to the body of the child during pregnancy, birth or infancy, birth asphyxia, nutritional deficiencies (especially during childhood), chronic ear infections, collection of fluid in the ear, meningitis (or other severe infection), chronic diseases, trauma to the head, loud noises, ototoxic medicines (medicine that is toxic to the ear-area as a side effect, for example, antibacterial antibiotics) and viral infections or other ear conditions (WHO, 2023)

Auditory disabilities delay the development of an individual and the capabilities in language, communication and education of the individual are also greatly reduced, especially if this occurs from adolescence or younger age, not to mention the social isolation that would follow the incapability of not being able to communicate smoothly.

To offset this detriment, additional ways of communicating are required. Most often to receive information and understand others, people with hearing loss or auditory disability use captioning and transcription to read the information instead hearing it. This comes at an additional financial and ergonomic cost as to communicate with others, typically it comes in the form of hearing aid machines, cochlear and middle ear implants are used to substitute if not replace some malfunctional parts of the inner ear. Substitute methods for communication to speaking also exist, such as text-to-speech synthetization, all enabled by software. (WHO, 2023 & W3C, 2017)

4.2.3 Mobility, Dexterity, and Ambulatory-based Disability

The name of disability that prevents movement is varied by the name it's classified as, but the definition is the same. One or more body parts exhibits impaired mobility or is completely cut off from movement. According to the CDC Disability and Health Data System data from 2020, this condition is found in 11.1% of the population in America.

Whether the disability in that 11% data from the CDC (2020) is more from lower end of severity like from low dexterity, repetitive stress injuries (RSI), Arthritis and Rheumatism to the much more severe if not critical disabilities like the complete loss fingers or limbs (Whether physically or from paralysis in a different state (Quadriplegia/Tetraparesis or Quadriplegia)), central nervous system damage to the spinal cord, Stroke, arterial lateral sclerosis (ALS), Parkinson's' and cerebral palsy, the approach to help this target group is the same, to create alternative ways to live their lives to their fullest. (W3C, 2017).

This group most likely has trouble utilizing typical, physical navigation tools like a mouse and a keyboard. To help them access and use technology and software, other assistive technologies such as voice commands, eye-or eye-lid tracking technology (Mattila R., 2020) and keyboard substitutes, like switches that utilize simpler mechanics to achieve the same results. There are also sips and puffs that utilize inhaling or blowing air into a whistle or other tube-like instrument, which works either based on the number of pressurized breathes or inhalations to activate functions. Historically, a whistle variant also exists in which whistling to reach different frequencies to achieves the same effect (W3C, 2017 & Olli Kivimäki, 2023).

4.2.4 Cognitive Disability

Cognitive impairments and disabilities reduce one's capacity to think, process and access information, perceive, recognize, and understand the information and the world around the individual (Disabled World, 2022). According to a National Library of Medicine (2020) global study on cognitive impairment prevalence, the

occurring percentage in Europe was between 5.65% and 7.65%. This broad category is comprised of dementia, Down syndrome, traumatic brain injury (TBI), autism, dyslexia, attention deficit hyperactivity disorder (ADHD), low vision, or any form inherited genetic disability.

These factors indirectly and directly cause problems for the individual's capability to learn things (Disabled World, 2022) and also reduce the chances of getting any use out of software. Even without any of these disabilities people's cognitive capabilities still decline as they age. According to the CDC (2019), 1 out of 9 nine recognize by themselves that their mental faculties are not working as well as before (CDC, 2019). To combat this mental decay, at some point in their lives those who need it often turn to assistive technology tools that can help them understand the information better, such as writing and reading assistance from programs in addition to word prediction/transcription tools (W3C, 2017)

4.2.5 Communication Disability

Communication is the one of the sources of happiness. Those who cannot achieve communication with others will inevitably feel left out. To combat this, there are dedicated assistive technologies created for this purpose, but even this will not fully cover the gap to normalcy.

There are communication disorders in five categories that make up the complex communication: concepts, verbal, nonverbal, graphic language, and speech (Tee-Melegrito A., 2023).

A reduction in the capability of any of these communication areas is considered as a communication disability. The problems in any areas of communication can stem from central nervous system problems, such as cerebral palsy, which reduces speech clarity. Aphasia, damage to the reading, learning and communication centered areas of the brain may cause inability to the same communication areas in varying degrees (CDAC, 2023).

The support required to overcome this includes speech, gestures, body language, writing, drawing and interpretation. On the technical side, speech generating devices, captioning in real-time and speech synthetization devices are used (CDAC, 2023).

4.2.6 Self-Care Disability

Self-care disability, by definition, is the situation where a person is stuck with a physical, mental, or emotional condition lasting six months or more, who has difficulty in accomplishing activities, such as dressing up, keeping oneself clean or moving with autonomy (Work and Family Researchers Network, 2018).

The prevalence of self-care disability in the US during the year of 2007 was 3.0 percent when the population age of 5 and above was considered. The overall trend for seemed to decline towards the adult age and then seemed to start again as the age increased in the people tested (Cornell University, 2007 Disability Status Report of United States).

The tools for combating this disability are mostly created for promoting self-care with the intention on getting the individuals back on their feet and working independently rather than having the patients turn even more dependent. These tools include medicines, diagnostics, counselling based on the diagnoses, and remote digital technologies that could be accessed from outside the formal health services (WHO, 2022).

In addition to problems that stem from the six presented disability categories, all the other disabilities that do not directly fall under the WG-SS categories are not listed here as a miscellaneous group due to not having that many uniform solutions that can be provided, examined nor would they provide any observable advances to the development of this thesis.

Still, the listed target groups are still people who can be considered impaired or disabled. These people would still use assistive technology or other common tools for accessing information like mobile phones should it be within their capabilities to do so. If they also have problems accessing them, they could indirectly

benefit from this as most of the technologies reviewed are related to the most prevalent disability categories, making the possibility of one or more of their symptoms being able to help through the common, already useful recognized assistive technologies.

Since the limitations of the disabilities within the arranged scope of the thesis have been cleared, the next parts on the thesis will address the software usage stages which occur in everyday life and how the disabilities affect that process.

5 SOFTWARE STAGES OCCURRING IN EVERYDAY USE

The idea in this section of the thesis was to gain an accurate first-hand idea on what different stages or parts in common software there are and then to see practical examples on how the programs are utilized, giving a precise visual on how the programs utilize the parts of their sum.

The processes that are chosen to be analyzed here all fit with definition of software requirements according to IEEE (Institute of Electrical and Electronics Engineers) & CMMI (Capability Maturity Model Integration): A condition or capability needed by a user to solve a problem or achieve an objective (Estimancy, n.d.)

The software processes created adhere to the requirements, as such, it's a valid claim to which can be used as a baseline for this part of the thesis.

The software stages used in this part were observed, thought, and created from observations in everyday software processes. These simple building blocks that programs require to function on a grassroots level were segmented into 3 categories. These create the barebones functionalities of a software and together create the basis for the software itself. However, not all of the possible, but only the relevant functionalities for the thesis are listed here.

5.1 Current stages in accessing software

The next 3 subsections contain information on how to access software.

5.1.1 Installation

Installation refers to placing in position or connecting for service or use something that is relevant in the context. It also applies to software use in this case (Dictionary, 2023, Install definition).

In this scenario, the installation of software can either be digital or take place from a physical disk should the system requirements be met. The ways to install are: By installing it directly from a CD or DVD, installing a software from a download, installing from a distribution platform such as Steam or using a USB flash drive for installation (Computer Hope, 2022).

5.1.2 Launching software

On Windows, the usual way of launching software is either by double clicking the shortcut that appears by default on dashboard or pressing the Windows button and typing out the name of the program and then pressing enter (Hoffman C., 2021).

This can also be achieved in an alternate way. If the program has been pinned to the taskbar, it can be quickly started by pressing the Windows button + the number it resides on the taskbar. The first program on the row would be Windows button + 1, the second would be Windows button + 2, etc. (Hoffman C., 2021).

5.1.3 Login to Program

Logging to a program or a platform, such as a window requires a password which is a premeditated string of letters and numbers in a certain order which when entered, would provide access to the said platform or program.

On many platforms, such as Windows, there are other ways to also sign in. Using Windows Hello, it is possible to use face, iris, fingerprint, or a PIN code which are tied to your Microsoft account. In the case of an emergency, it is also possible to use a USB stick as a security key to login if you forget your password (Microsoft, 2023, Windows sign-in options and account protection).

5.2 Current stages in using program

The next 4 subsections contain information on possible scenarios on how to use the software.

5.2.1 Navigation

Once the software has been accessed, navigating software can be done through a multitude of ways. The current standard medium for navigating software (and electronics for that matter) is using a GUI (Graphic User Interface). The design varies from program and platform but all of them are designed with intuitiveness in mind. The different, smaller elements that make the GUI are built so that there needs to be minimal prior understanding of system to be able to understand it (Metwalli S., 2023).

The Graphical User Interfaces are by default mostly navigated by mouse and keyboard, but other tools exist for alternative means of use. In some platforms or programs but not limited to, such as Linux (Platform), or programs (Git Bash) it is sometimes even required to use commands as a mean of navigating to some parts of the said software (Metwalli S., 2023).

5.2.2 Data Retrieval

Modern data transfer is so fast as even 5G networks are making their entrance, yet people often forget that retrieval must take place first before any data transfer. The act of data retrieval is the process of identifying the correct data that is related to the request put forth by the user and then extracting it (Rouse M., 2014).

The said data retrieval can for example be answering a query for, a video service search such as YouTube. This initiates a callback from the database to find the appropriate videos which fit the search criteria (Rouse M., 2014).

5.2.3 Data Manipulation

After retrieving the said data which the user wants from the software database, the user most likely would like to alter the data somehow or at least check its status. One of the simplest examples of manipulating data to reach one's goal of getting information is to simply organize the data by sorting it – whether by built in functionality, for example, in Windows file directory.

Actual data manipulation, i.e., changing data to other form than it was prior to fetching the data in the saved form that the database found from it as, can vary in size and scale massively. It can either be as simple as changing a name file or be a big change on the line of moving or replacing hundreds of lines of data in the database, to for example enabling forecasting of future events based on the current data in a company environment (Yoshi S., 2021)

5.2.4 Data Saving

Should the data manipulation be in a wanted form, next in basic use is saving or writing that data. To save the data, for example, inside the Windows environment would suffice to press the hotkey combination of CTRL+S. That data saving example could be counted among the simplest ways of saving information.

Often the data saving is performed on the local hard disk or flash drive, whatever is available for the system. However, it is also possible for the system to attempt to push the information into some other database remotely if the user's skill is up to par. Google's Firebase, for example, provides a way of emulating this before pushing the data to be saved into the internet (Google Firebase, 2023)

5.3 Current stages in communications

The next 2 subsections contain information on possible scenarios on how to use the software to communicate with others.

5.3.1 Visuals

" From computer graphics to data visualization, computers play an increasingly important role in two and three dimensions and rapidly digesting data from photo and video, enabling applications in computer vision and scientific observation. As such, nearly every corner of Computer Science – including machine learning, human-computer interaction, computing, and database architectures, and data science – interacts with the visual dimension." (Hanocka R., 2023)

Just as specified on the page of Department of Computer Science of Chicago – nearly every aspect of computer mechanics interacts with the visual dimension. This even more so for practical intuitive visual elements, such as tutorials which guide the user to reach their objective in using the software program.

An example of this is Google's Gmail. The functionalities behind the buttons are abbreviated in text; quickly explaining the action that would happen should the

user click a button. It also conveys the way to send messages by highlighting the compose-button, grabbing the user's attention quickly. The amount of unread email messages is shown by a number indicator next to the inbox.

The search option is there for queries for straightforward action of the user knows what to look for. Also, the incoming and sent emails are classified under their respective outgoing and incoming tags which cut down on search time.

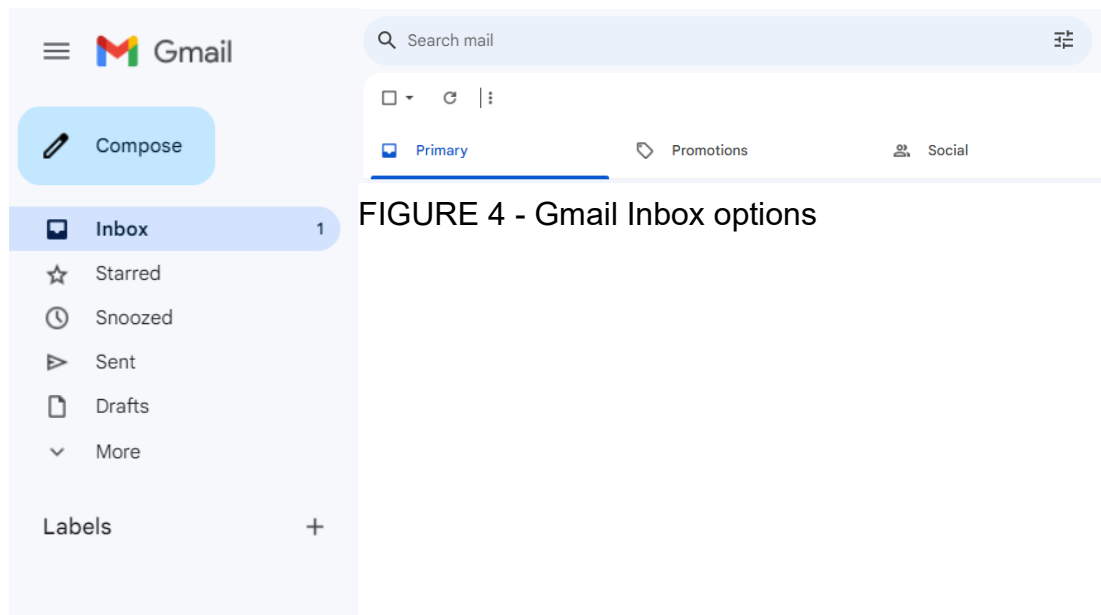


FIGURE 4 - Gmail Inbox options

5.3.2 Communication

Communications is the exchange of information between two or more things, such as people, devices, software, etc. In digital communication, the data is moved between two computing devices through medium, which can be either a fiber optic cable or wireless signal. The congregation of digital users using same digital services is a computer network. To connect people for using communication software, the correct settings, such as the DNS and IP address configurations must be set in place (Computer Hope, 2023).

One example of usual communication between people using software can either be using some kind of social media or communication service where text, voice, video messages and other files could be sent to each other or using a voice chat like Skype or Discord some other equivalent.

The next part of thesis will address the possible problems that can happen when the presented disability categories are also taken into consideration when using software in similar scenarios that have been presented as examples in this chapter.

6 PROBLEMS AND POSSIBLE SOLUTIONS IN EVERYDAY SOFTWARE USE

6.1 Heatmap of disabilities

To get confirmation on how serious the impact of different disabilities has on the use of software, the disability categories (Ref: WG-SS explained in 4.1 Disability classifications) were collided with the software stages (Ref: 5.1-5.3.2 Software stages occurring in everyday life) to get an accurate idea.

Heatmap	WG-SS					
Software parts	Visual	Auditory	Motor	Cognitive	Communication	Self care
Installation	5	2	3	3	3	2
Launching software	3	2	4	3	2	2
Login/Accessing service	5	2	4	4	2	3
Navigation	5	2	5	4	2	3
Data retrieval	5	2	4	3	3	3
Data manipulation	5	2	5	4	3	3
Data saving	4	2	4	3	2	2
Visuals	5	3	4	4	3	3
Communication	4	4	4	4	5	3
	4.56	2.33	4.11	3.56	2.78	2.67
Legend						
	1, Normal					
	2, Some difficulty					
	3, Varies between some difficulty and severe difficulty					
	4, Constantly hard to do or might even cause pain					
	5, Very hard to do or bordering on impossible					

FIGURE 5. Heatmap of disabilities when using software.

The colors provide a quick way of interpreting the severity scaling of the impact. The heaviest problems can be found from the visual disability, followed by motor and cognitive categories. Even in the more favorable use scenarios the usage difficulty rounds up to being between severe and very hard, making it a tremendous challenge. Thus, this part of the thesis will mostly focus on trying to alleviate the more serious issues related to software usage.

The scenarios of use provided in the earlier stages (Ref: 5.1-5.3.2 Software stages occurring in everyday life) of will be processed here with the disability problems factored in, trying to find or suggest solutions to the use case scenarios.

Afterwards, the heatmap will be reviewed to see on how much the suggested changes can have an impact on the severity ratings.

6.2 Problems in accessing software

The next 3 chapters will assess the problems in relation to accessing the software from the perspective of the disabled and suggesting some changes which could make the problem easier to approach or fix.

6.2.1 Problems with Installation

As earlier already stated in the thesis part 5.1.1 Installation, installation refers to scenario in which software is successfully installed onto a computer, in a condition that the software can be used.

One example of installing software is for this example, Notepad++. It's an application that is installable from an .exe-file which contains all relevant configurable files for a quick installation.

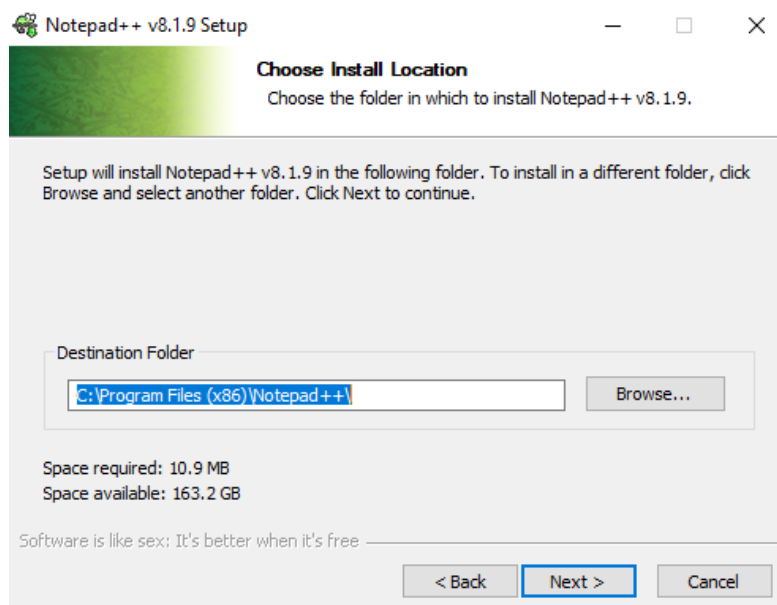


FIGURE 6. Notepad++ v8.19 Setup

The number of problems the installation can cause vary greatly, but all in all most disability categories reduce the capability to use of computers and their related software, so generally speaking using computer can be hard.

The installation progress for all .exe-files is the same, Notepad++ is no different. It can either be installed with custom settings into a self-chosen location on the computer disk or allowed to install by itself into the default location by double clicking or pressing enter on it after locating the file (Fisher T., 2022, What is an EXE file and how do you open it). The installation process can be finished with a few mouse clicks or pressing enter a few times.

The process itself can seem easy but it can be a nigh impossible challenge at worst if mobility or visual disabilities are present. Pressing on the keyboard or using the mouse can be painful or hard (Ref: 4.2.3 Mobility/Dexterity/Ambulatory Disability). Locating the text on the screen or reading in general can cause the finding the executable file or installing the file correctly if locating the buttons can prove to be hard (Ref: 4.2.1 Visual Disabilities).

This can be circumvented to some degree with either using the standard-issue assistive technologies (Ref: 4.2, mentioned under respective disability categories) but there are ways that software can mitigate the problem.

The installation can be done quickly by using hotkeys to minimize the mouse and keyboard use by using software like AutoHotkey (Steve Gray et.al., 2014 AutoHotkey V2.0 documentation) where the installation process can be shortened down from multiple steps to a few or one button press.

Another possible way to go about the installation problem is to forego the installation problem altogether and use software like Ninite (Ninite, 2023, Installation). Ninite is an auto-installer which can group up programs chosen for installation without extra configuration required for any of the individual programs that are accepted into the script (Ninite, 2023, Scripting Ninite). It can also be automated to install the chosen programs as the executables appear on the computer. If this is not enough, Ninite also has support for multiple remote support/access software frameworks for remote help for hardline help anytime (Ninite, 2023, Ninite Help).

While a 3rd party auto-installer is not the most data-secure solution, it still provides a way of alleviating the issue. All in all, these solutions still require configuration on some level before they can be used to install the software, so the help provided to the disability aspect is moderate at best as it cannot be accomplished that easily.

6.2.2 Problems with Launching Software

As stated in 5.1.2 Launching software, the usual way of launching software is the same as installing it – a double click will be enough due to Windows built-in configurations. Writing out the system's name in the search field at the bottom left will also work. The search function gets faster the more you use it due to Windows Indexing, reinforcing the using habits of the said feature, potentially helping atrophied hand and fingers muscles due to use. (Microsoft, 2023, Windows Search Indexing in Windows 10).

The task of launching itself is a simple endeavor but still a daily necessity, providing possible problems multiple times a day if there should be trouble in launching software. The typical way of launching software might not be the best option if there are problems with using the search function of Windows. Having problems locating the program due to cognitive or visual disability, or the difficulty encountered when trying to use a keyboard with motor disability can be too much for some people.

In a situation where it is impossible to fight the issue, it might be better to leave to the computer itself. In Windows, it is fully possible to add applications to start automatically after booting up the computer from the settings (Microsoft, 2023, Add an App to run automatically at startup in Windows 10).

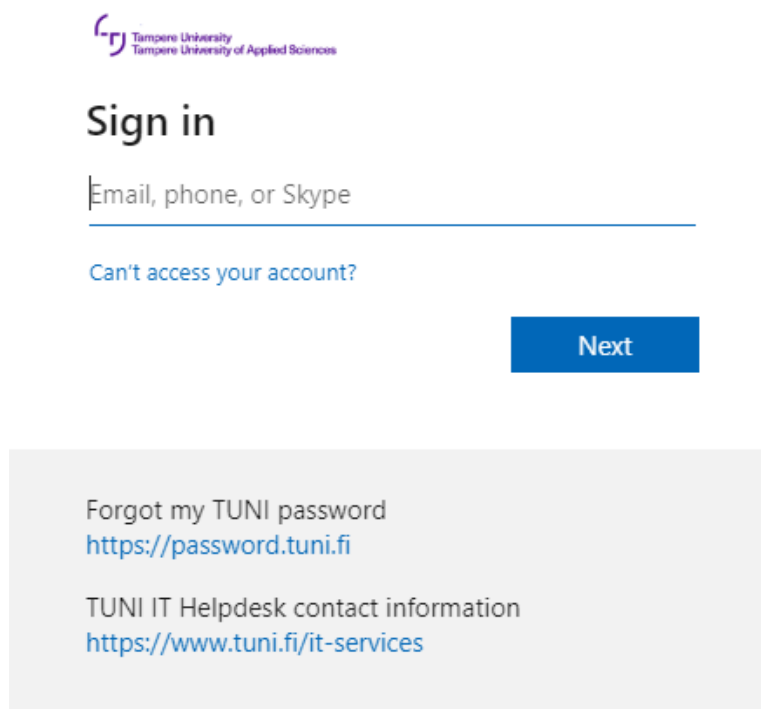
Another solution to launching the wanted software could be to use voice commands. From the heatmap's perspective, it seems like a viable solution to introduce voice commands as a solution to using software as the severity of communication disability was rated the second lowest in severity. This is of course implying that the capability for speech is not also affected.

The suggestions here provide two easy to use systems which do not have a high requirement for use, providing a cost-effective way of increasing the usability of software for the disabled regarding launching software, alleviating the software launching problem somewhat in the visual, motor, and cognitive categories.

6.2.3 Problems with Login/Accessing Service

Logging in to a program often requires using a medium for accessing the server. Nowadays a lot of the services where logging in is required are on the internet, requiring internet access and an internet browser.

Usually, the problems consist of forgetting some part of one's credentials, misspelling the credentials or overall, just reaching the criteria required to pass the security check that the site has put up for protecting your account.



Tampere University
Tampere University of Applied Sciences

Sign in

Email, phone, or Skype

[Can't access your account?](#)

Next

Forgot my TUNI password
<https://password.tuni.fi>

TUNI IT Helpdesk contact information
<https://www.tuni.fi/it-services>

FIGURE 7. Example of a login screen, the TUNI environment login screen

The problems posed by disability pose increased difficulty in landing the required string of letters and numbers for the email or user account and the password to login.

It is entirely possible to solve the problem with logging in by using 3rd party software of Google or other such services where the server stores and remembers the credentials, letting the user log into the environment without asking the password (Google Chrome Help, 2023, Manage Passwords)

The act of storing one's passwords creates into one place security risk, but it still provides a way of circumventing the accessing issue. However, this risk can still be mitigated by using 2-factor authentication (Microsoft, 2023, What is two-factor authentication).

The suggested solution provides a fairly data-secure and easy way to increase chances of the correct user successfully logging in, decreasing the severity ratings slightly.

6.3 Problems in using program

The next 4 chapters will assess the problems in relation to using the software from the perspective of the disabled and suggesting some changes which could make the problem easier to approach or fix.

6.3.1 Problems in Navigation

The usual problems in navigation are switching pages or finding the wanted software features hidden behind multiple screens. For a new user, it often involves a lot of back-and-forth movement between different screens.

To understand the flow of the software's functionalities and what the intent behind the software is, intuitiveness and understanding the software's visuals are necessary. The impact on the problems related to not being able to navigate between software often stems from inability from either seeing or perceiving the software

menus, affecting visual and cognitive areas, or inability to act on one's understanding of software sufficiently, making the fluid navigation of the software hard due to motor problems.

In a scenario where visual or cognitive faculties require assistance, voice control systems such as Siri or Google Assistant from one's phone (TechTarget, 2023) to complement bad vision using voice commands to achieve common tasks. For more refined use of navigation, it is recommended to combine the virtual assistant's help with the use of more traditional means of assistive technology (Ref: 4.2.1 Visual disabilities).

In a scenario where reduced motor capability is present, it is possible to use one's eyes to move the mouse cursor with a software named Tobii EyeX (Tobii, n.d. & Hachman M., 2016). The software requires a hardware sensor installed to be able to mimic mouse movement with the software it comes with, fully creating a scenario where the hands are not needed for mouse movement. If this is paired with one of the built-in accessibility features of Windows, On-Screen Keyboard, it is possible also to use the keyboard better than before (Microsoft, 2023).

Should there be only problems moving or controlling the extremities, i.e., the fingers accurately, it might be possible in the future to use Virtual Reality tools to move and control the software in the future. There are already tests on how efficient and accurate the virtual reality sensor technology is, proving the interest in the potential of the technology (Jan, P. Vox et. al., 2021).

However, it is still in future as current computer management peripherals are entirely two dimensional and used on a flat screen. Introducing the third dimension, the depth metrics into the calculations for the software would cause reworks for it to be utilized. It is an interesting thought, but the technology is not capable enough yet to provide the imagined result (Yang Li et.al., 2019).

The current suggestions provide a solution and lessen the severity risk of using software mainly in the visual disability category as the more potent solutions of virtual reality are still far away in the future.

6.3.2 Problems in Data Retrieval

Data retrieval means identifying the correct or the most appropriate answer to a query that the user has put forth for the software to search and solve (Ref: 5.2.2 Data retrieval). A practical example of such software which would respond to such data retrieval queries could be a video service from the internet, like YouTube which many can access easily. It could also be a part of a greater whole like the File Explorer system which is a part of Windows. It searches anything on the system disk, trying to answer the question from local data on disk.

What all these programs have in common is that they search information and that the query tags/keywords must be accurate for it land an accurate answer. The keywords work on a Boolean basis, in a binary manner (Computer Hope, 2023). Binary means that the answer can be 0 or 1, on or off. In this case, the queries answer in a manner that follows the same idea as binary numbers. The system can either find an answer akin to one (1), a positive result for the search, or return a zero (0), a negative answer if there were no answer to search. More advanced systems can provide more accurate answers.

The problem with searches that often occurs, as mentioned above, is that the wording is incorrect or enough incoherent in order that the system cannot connect the search word to an answer.

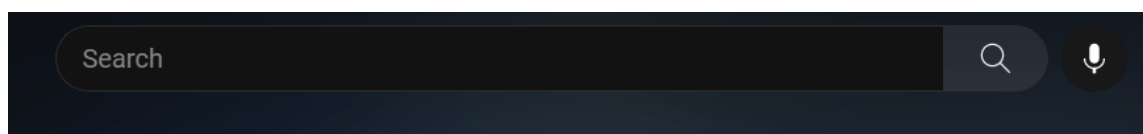


FIGURE 8. YouTube search bar from dark themed browser

To solve the data search problems, suggestive algorithms, also known on some level as search engines, could be introduced to expedite the search process. Algorithms are networks of data which choose the best options out of the possible answer in a repetitive manner. (Merriam-Webster, 2023). It is also possible to forego the whole step of searching altogether use to use voice commands if there is support for it. In FIGURE 8., The YouTube search bar, there is a microphone

icon. The icon is a button for activating voice search in which YouTube conveniently supports. For general use outside of internet, Windows voice recognition can be used.

The suggestions provided can be quickly adapted as there is built-in support for the features, providing a quick relief to daily use and the heatmap severity ratings, implying if there are no problems in taking them to use.

6.3.3 Problems in Data manipulation

Data manipulation can be any form of change applied to data which has been retrieved successfully. It can be sorting, modifying, or completely deleting the data which has been retrieved into an active form. The un-confirmed changes to data will stay in the flash memory (RAM) until the change in the data is saved back into the storage disk for later use (TechTarget, 2019).

The common problems in data saving are either on the hardware side or on the software side. For the purposes of this thesis, only the software side will be used. The software is “dormant” in the form of an unopened file before it is activated in through data retrieval. Then it can be manipulated i.e., capable of being changed to the wanted form, and then it can be saved through data saving for further use.

In data manipulation, an easy practical example of a scenario can be writing something down into an active file, for example in Microsoft Word just like this thesis. A more elaborate example in the software development environment would be coding with its syntax rules (Merriam-Webster, 2023). The problems in data manipulation are still the same despite the complexity. The data needs to manipulate successfully to a degree that its usable later.

A partially helpful, patchwork solution to this problem is to get used to hotkeys or other assistance programs which can cut down on the required amount of work. Another solution to this issue is to use AI-related text checkers when possible. Grammarly (Grammarly, 2023) is one such tool which has the role of checking text for grammar mistakes. For coding, it is recommended to stick to compilers where the syntax will be checked automatically to get quick feedback with the

possibility of the software providing the implementations for the syntax problems. Visual Studio Code provides syntax checking and fixing features, built-in straight out of the box (Visual Studio, 2023). It is also possible to change the Visual Studio Code syntax outlook for greater visual accessibility (Visual Studio, 2023).

The suggested solutions provided here are mostly for daily use solutions, such as writing emails and grammar checking with some software development-related tips from the Software Development program experience. The solutions provide some degree of relief to visual problems as long in the scenarios where to severity is not too high.

6.3.4 Problems in Data Saving

After data manipulation comes saving the data into the disk or some other remote station located on the internet, like the OneDrive or other cloud storage equivalent (Muchmore M & Duffy J., 2023).

The problems encountered in saving data are often related to file being incompatible to the required file form or the file still being open in some form. The file cannot be saved if it is still being used somewhere else. In a Boolean manner, the file can either be open or saved at a time, both states cannot be active at once. Only in the future when quantum mechanics-based quantum computers take the place of the current computers with quantum super position at its core, can both states of file activity occur at once (Caltech Faculty, n.d).

Currently, the next best solution to something so advanced to quantum mechanics is online saving. For example, Microsoft-based tools provide a way of inputting the file online, putting the program up to task on creating the updates to the file directly to the file version. The system is designed to pull data from the cloud, so it also works even if the user should change the computer. Only logging to one's user in is necessary to access the same files from some other computer (Cavanell Z., 2021).

The solution provided is merely using a 3rd party solution, but it's effective. The solution provides an easily accessible solution to data saving despite file type. Overall, the efficiency of the solution can be thought to be subpar as the file saving problems can still occur despite using a new environment.

6.4 Problems in user communication

The next two subsections contain information on possible problems and some suggested solutions on how to use the software to communicate with others.

6.4.1 Problems with Visuals

The visuals are there to communicate with the user the relevant information in a way that can create an understanding between the software, its medium, the computer and the user.

To successfully communicate with user, the visuals need to relay the function behind the visuals to user in an appropriate form. One example of this kind of software are games. The visuals in games often cater to their target group with the intention to target a specific group. To ensure that the incorrect kind of games are not exposed to wrong age groups for example, there are special committees that give out gradings based on the content of games or other entertainment, like the ESRB (ESRB, 2023)

The possible problem in software visuals is the opposite of the successful scenario. Problems in visuals can occur if the visuals are too strong or too faint, for example, making the text too hard to read or the colors too strong or one-sided (Ref: 4.2.1 Visual Disability). This especially so if there is disability involved. Too strong or faint visuals can lead in those with low vision or cognitive disability experience disorientation or pain in trying to process all the flashy colors.

In addition to already mentioned ways of combating visual and cognitive disability (Ref: 4.2.1 Visual Disability, Ref: 4.2.4 Cognitive Disability) there are other ways to combat the problems mentioned.

One such way is to combine hardware and software capabilities in smart glasses as an additional filter before looking at the computer screen. One product of that category is RNIB smart glasses. It helps people with low to very low vision to identify and detect objectives up to three meters away. The smart glasses were created in Oxford University and have later been in development in RNIB for four years, even winning the Google Impact Challenge competition, drawing funding to their product's development with their victory from one of the software development's giants (Royal National Institute of Blind People, 2016).

The solutions here also consider the help of the hardware which warrants extra costs. The suggested solution is still not a commercialized product but promises some degree of help within three meters, so the process of using a stationary computer and its software should be helped. Thus, the most critical WG-SS category, the visual disability category, has been alleviated a good amount.

6.4.2 Problems with Communication

If the visuals are the how the communication between the software and the user happen, then the communication in the software is the communication between the different users.

In digital communication, software transforms the voice user has to digital signals with VoIP which can be sent and received by the parties present in the call. (Federal Communications Commission, 2003). The requirement for VoIP protocol to be used is an internet broadband connection.

From a disability perspective, the problems that can occur when communicating with others remotely in this kind of situation are many. Cognitive disability can make speaking or understanding speech a strained effort. Visual disability can reduce or altogether remove the chance to enjoy video calls with other users. Cognitive problems might make gesturing hard, making it harder to convey feelings through facial expressions to others in a video call. Communication disability can deter speaking to others, auditory disability can in return reduce the capability to hear others (Ref: 4.2.5 Communication Disability).

Overall, the disability severity was the highest in the communication category in the heatmap (Ref: 6.1 Heatmap of disabilities), so the diverse cast of problems is also supported by this information.

As for the solutions, Visual disability problems can be dealt to some degree by using a speech reader to form the visual resources used into an auditory form for circumventing the issue.

Auditory problems could be solved by introducing again a Braille touchpad or sign language (Ref: 4.2.2 Auditory disability).

Cognitive and communication disability (Ref: 4.2.4 Cognitive Disability & 4.2.5 Communication Disability) problems can be deflected away to some degree by using 3rd party help for formulating sentences and options for conversations. One such example of software is the OpenAI's ChatGPT (OpenAI. 2023), a smart algorithm that can provide accurate answers within a changing premise, just like a conversation often takes turns. It could be used to provide meaningful insights into the possible avenues on where the conversations might go. This would help the user to alleviate the problems through thinking ahead by preparing for speaking with others.

The current version of ChatGPT is ChatGPT-4, the fourth updated version of the said software. It's capable of having conversations with people. One other possible use for the ChatGPT would be to have the algorithm used as a speech synthesizer in the future. It could be possible to use an artificially thought answer to buy time in a conversation to think about meaningful comebacks.

It could also be possible in the future to have the ChatGPT-4 tap into conversations for speaking assistance. Some people have already managed to utilize its skills to some degree create spontaneous dialogue in a game for the Non-Playable-Characters to talk outside of the normal scripted conversations, providing credibility to the proposition (Cambridge Dictionary, 2023 & Inworld-AI practical example video in Youtube from the user Bloc and Inworld, 2023).

As for the efficiency of the suggestions, it is a bit hard to measure. The overall efficiency leans towards the positive so the severity can still be reduced, but all of them still require some degree of capability in using software, so the suggestions are not suited for scenarios where the user cannot use software that well on his/her own.

6.5 Heatmap of Disabilities Reviewed

The results show that the suggested solutions for most of the cases provided only a moderate to slight response, and only once was it effective enough to be called good.

The suggested solutions provided were chronologically reviewed during this part to see how the solutions would impact the end result and each solution one way or the other affected the number values in the heatmap.

The words used for describing the changes do not have a predefined numeric value which could be used for grading effectiveness, so subjective interpretations of effectiveness based on experience were used.

By definition, moderate means to keep the within reasonable limits, to not to be excessive, a medium quality or quantity in the given context (Dictionary, 2023). The number value for moderate results given will be 0.5.

As for the cases of lower effectiveness, there were a few cases where slight reduction of severity was the case. Slight means of small degree, small amount, of little importance. In short, not enough to make a dent into the end results but it's still a better result than no improvement of results (Dictionary, 2023). The number value for slightly positive results given will be 0.25.

Good is above moderate; the meaning is of satisfactory quality, quantity, or degree (Dictionary, 2023). Thus, good results will be more effectively also shown in the heatmap changes. The number value given will be 1.0.

To recollect, the severity rating changes were as follows.

moderate reduction to installation

moderate reduction to launching software

slight reduction to login/accessing software

moderate reduction to navigation

slight reduction to data retrieval

moderate reduction to data manipulation

slight reduction to data saving

good reduction of visual disability to visuals

moderate reduction to communication

as for the visual changes:

Heatmap	WG-SS					
Software parts	Visual	Auditory	Motor	Cognitive	Communication	Self care
Installation	4.50	2.00	2.50	2.50	3.00	2.00
Launching software	2.50	2.00	3.50	3.00	2.00	2.00
Login/Accessing service	4.75	2.00	3.75	3.75	2.00	3.00
Navigation	4.50	2.00	4.50	3.50	2.00	3.00
Data retrieval	4.75	2.00	3.75	3.00	3.00	3.00
Data manipulation	4.75	2.00	4.75	3.50	3.00	3.00
Data saving	3.75	2.00	3.75	3.00	2.00	2.00
Visuals	4.00	3.00	4.00	4.00	3.00	3.00
Communication	3.50	3.50	3.50	3.50	4.50	3.00
	4.11	2.28	3.78	3.31	2.72	2.67
Legend						
	1, Normal					
	2, Some difficulty					
	3, Varies between some difficulty and severe difficulty					
	4, Constantly hard to do or might even cause pain					
	5, Very hard to do or bordering on impossible					

FIGURE 7. Post-review disability heatmap

Earlier on the thesis it was concluded that even the ideal software development framework would not be able to accurately provide software with the needs of the disabled. The thesis was built to continue afterwards with the introduction of the software use stages and the disability classifications with the intent to further re-evaluate whether or not that statement is true.

So, in a deeper analysis, and from an alternative angle. To address one of the unaddressed parts of this thesis in regard to this problem, Hypothesis (H2): “Ideal software framework can be adjusted to better fit the needs of the disabled” – will be brought up again.

The software frameworks were analyzed from a top-down view, akin to from a bird’s eye. Here, in the latter parts of the thesis, the smaller, the more specific parts of software and the disability classifications were collided to see if on the smaller level, micro-level, there could be some changes that could be possibly applied to the software framework.

The end results show that overall, the number value changes when comparing FIGURE 7. and FIGURE 8. altogether were greatest in the visual and motor disability categories, with the average severity value going down by 0.45 and 0.33 respectively. The least amount change happened in the self-care disability department where the amount of change was a round zero.

To these two conjugated areas, the software use stages, and the introduced WG-SS disability categories were able to see small changes with help of a lot of suggestions. Many of the suggestions were generalizations catering to the actual disability category, not the accurate needs of a possible user, so in real-life applications the effectiveness of the provided solutions may vary radically, depending on the severity of the case.

Also, to add to the unstable results of the suggestions, some if not all of the disability solutions rely also on the physical aspect to be solved. Using a keyboard and a mouse, for example, is one of the more known if not the most known way of managing a computer and its software. In some cases, other types of assistive technologies are necessary, bringing in another dimension to this issue, physical dimension.

Software development aims to crack down and solve software-related problems, but when a part of or the whole root of the problem lies in the physical dimension, there are suddenly problems that software is not suited to handle, making the

problem paradoxical in nature, and making only software development ill-suited to handle the issue.

Trying to solve the problem would cause a lot of issues for a traditional software development company. Financially, bringing in specialists to develop something that is new cannot be marketed or sold easily to a vast target group, so it is not a lucrative business venture. Also, from a technical standpoint, creating a lot of prototypes for a type of a disability only to tackle one problem is ineffective.

The effectiveness of the solutions would be greater when the severity of the cases would be lower, as the simpler cases would take less funds and time to construct, enabling a greater audience. More specialized solutions would be needed the more severe the case is. The specialization would narrow down the amount of people who would need the product; at some point the product would not be created to help the general classification of disability, but instead an individual's needs. For companies which intend to keep their businesses afloat, this would be akin to bringing bane to investors and income alike, shooting into one's own feet. Perhaps this is why the greater companies have not yet taken a step towards solving this issue as there is not too much to gain.

These problems are merely what have been encountered with a review study of disability in regard to software development. This paper cannot be said to be an in-depth study nor to provide an accurate answer as there is a lot of unknown ground to be covered, but it still gives an approximate nudge, a guide on which direction the development most likely would go should the task be undertaken with the resources the thesis presents.

Thus, the answer to (H2) "Ideal software framework can be adjusted to better fit the needs of the disabled" is yes, it could be possible if a) The framework is adjusted the approach to the smaller parts of the software as seen from the changes in the heatmap reviews, and b) the severity is not too high.

7 TAMK PERSPECTIVE TO THIS ISSUE

In software development, despite the model or framework used for development, the path used to produce software is heavily dependent on the feedback. The good and the bad points are conveyed. The given feedback causes a retrospective; an effort of rumination to redirect progress made to a better suit wanted goals (Atlassian, 2023).

The frameworks presented and reviewed in this thesis work on feedback (Ref: 3.1.2 Software development frameworks referenced). The process which prepares people for software development, namely the action of teaching students to work as software developers, should be no different. It can become better and more efficient of a process by executing a retrospective session.

For the retrospective, the materials used over the course of 4 years of studying were analyzed and assessed to answer on how well it gives capability to understand, process and create software which can consider the needs of the disabled.

In an act of semi-scientific research, a move in the form of material search was made to reach the goal. The research process involved going through all of the general course content descriptions. Comparisons for keywords were made by scanning the material for the keywords: “Accessibility”, “Disability”, and “Efficiency” – which likely respond to the core subjects presented in the thesis. If not present or detectable, then the considered study program was not estimated to be offering accessibility or other keyword related solutions, at least not focusing on the issue.

The end results showed zero matches with the keywords – hence, obviously accessibility does not cross the threshold criteria to include such substance to software engineering training. Of course, accessibility being a nonfunctional requirement is not as high on the list, but it still should have some impact or presence in the material.

Should there have been any matches, deeper analysis of the provided material would have been necessary. However, the result was concluded to be negative, so no further action was necessary.

The negative result creates a precedent, where the study program does not seemingly support the area of alternative software development.

Direct citations from TAMK's homepages:" At TAMK, the guiding principle is that everyone is treated equally in similar situations and without discrimination", or: "... management is committed to promoting accessibility and equality practices" (Tampere Universities, 2023 Accessibility, and equality at TAMK). These are examples of valuable insights but however these citations could also be implemented at the course material level.

This contradiction creates evidence that in a modern Finnish highly valued applied university of sciences accessibility and the other keyword related issues might not be seen as an important issue which has not yet appeared into the radar of people who can change these things. Or perhaps the practical side of things has yet to be developed to the degree where the inclusion and equality factors would be up to par of the statements.

Should there be intent to wish for the amelioration of this issue, this thesis and its results could be used to implement the following improvement ideas (importance formed based on suggested solutions):

1. Optional courses focusing on alternative means of software development, such as the software which the disabled might use.
2. Algorithm-development (a more practical way of AI and Machine Learning alternative for creating programs).
3. Giving more options to the Embedded systems – type courses to handle more variable types of hardware and software settings than just circuit boards.

4. Professional Software Development or similar courses could also include alternative standard software development types instead of the standard processes when preparing students for software development.
5. Project management and Management skill-based courses could include alternative angles to introduce the less seen groups such as the disabled in some form
6. Software Usability and User Experience – related courses could have some kind of a way to introduce accessibility guidelines to make software more easily usable by the disabled.

These guidelines are merely suggestions that could provide way small nudge towards to changing the learning material to accommodate the alternative requirements from smaller, marginalized groups, such as the disabled without disturbing the normal, already standardized workflow in teaching too much. The solutions offered here provide directions to take the development towards to obtaining the goal of inclusiveness for all without compromising too much from the established standard.

The change should appear (if possible) in small increments to let both the teachers and the students acclimate to the change and to see the feedback. The amount of interest there is in solving this problem should be reviewed and change the teaching material based on demand, akin to the Kanban pull-method of minimizing losses and maximizing output without no additional cost (Ref: 3.1.3 Kanban, Scrum and DevOps as a springboard for an ideal software process).

8 DISCUSSION OF RESULTS

This thesis has aimed to explore the disability aspect and offer suggestions to the problems with a generalized look. The solutions suggested here should be used as a springboard for further research in another paper.

The results found within this paper's limitations have shown that methodologies can always be refined further. The offered philosophies, solutions, toolkits, and procedures that come with software development methodologies are well suited for their purposes. Those well-established procedures, however, fall into disarray when some part of the procedure is not followed. Of course, it is expected that not following the rules of some process can cause problems, just like in software development where the syntax in coding should be followed to avoid problems.

The disability problems handled in this thesis are a subject that has avoided the eyes of the many, even to the degree that not even the more superficial themes of making software easier to use, such as accessibility were not found on the current Software Development course roster (Ref: 7 TAMK perspective to this issue).

Seeing as the disability aspect has yet to become a mainstay aspect in course material in a high-class education environment, the results of the papers also make sense.

The results of the paper were produced through answering the two hypotheses and the following eight working guidelines that use software development frameworks and smaller individual processes in software development. The analyzed results following the initial research question ("Is the normally ideal line of software development refined well enough to consider the needs of those who are impaired or disabled?") show that currently, the standard, nor the ideal software development process constructed from the best parts of the standard processes cannot fully cater to the needs of the disabled (Ref: 3.1.4 Ideal Software Development Process, (H1) - Ideal software development does not consider the needs of the disabled.)

Through exploring the most common software development processes, the use scenarios and the related problems were evaluated to see if the smaller individual processes could be changed or rather, adjusted to help the disabled in their daily challenges in using software.

Through a generated heatmap formed based on colliding studied categories of disability and software use stages, the results showed that by modifying individual software processes, the software development framework could be adjusted to follow the needs of the disabled to some degree if certain criteria are met. The results were better when the severity was lower, so for effectiveness purposes that was set as one of the criteria. The second criterion was that the individual processes should be fitted into a framework for creating more malleability into the failing parts of the software development methodology (Ref: (H2) – Ideal software framework can be adjusted to better fit the needs of the disabled.).

The higher severity cases were harder to provide sensible solutions, as the current technology cannot fully cater to the scenarios where the severity ratings is higher.

The problems assessed in this paper require more refinement and problem solving through multiple iterations. Akin to using the software development methods that Scrum and DevOps use, a certain number of iterations would be spent, and the problem would be more geared toward being solved.

This way of thinking leaves more into the future. Just like the software development, slow and steady changes win the race instead of breaking everything with rushed changes. In the future through the small leaps taken over time, it is fair to expect that a solution for the problems of disability will be found, in an effort to bring equal levels to software use even if there is disability involved.

REFERENCES

- Ann, 2018 “Self-Care Disability,” *Work and Family Researchers Network* (blog), Read May 8, 2023, <https://wfrn.org/glossary/self-care-disability/> - Originally from Erickson, W., & Lee, C. (2008). 2007 disability status report: United States. Ithaca, NY: Cornell University Rehabilitation Research and Training Center on Disability Demo-graphics and Statistics. <http://digitalcommons.ilr.cornell.edu/cgi/viewcontent.cgi?article=1256&context=edicollect>.
- Atlassian, 2023, “Agile Retrospectives, Use the past to define the future”, Read May 16, 2023, <https://www.atlassian.com/agile/scrum/retrospectives>
- Atlassian, 2023, “What Is DevOps?,” Atlassian, Read May 17, 2023, <https://www.atlassian.com/devops>.
- Britannic, 2023, “Software | Definition, Types, & Facts | Britannica,” Read May 10, 2023, <https://www.britannica.com/technology/software>.
- Caltech Faculty, n.d, “What Is Superposition and Why Is It Important?,” Caltech Science Exchange, Read May 22, 2023, <http://scienceexchange.caltech.edu/topics/quantum-science-explained/quantum-superposition>.
- Cambridge, “Normative,” May 17, 2023, <https://dictionary.cambridge.org/dictionary/english/normative>.
- Cavanell Z., 2021, “Windows 365, Your Cloud PC | What It Is, How It Works, and How to Set It Up,” Read May 14, 2023, <https://techcommunity.microsoft.com/t5/microsoft-mechanics-blog/windows-365-your-cloud-pc-what-it-is-how-it-works-and-how-to-set/ba-p/2548150>.
- CDAC, n.d, “Communication Disabilities – Communication Disabilities Access Canada,” Read May 12, 2023, <https://www.cdacanada.com/resources/communication-disabilities/communication-disabilities/>.
- CDAC, n.d, “People Who Have Communication Disabilities – Communication Disabilities Access Canada,” Read May 2, 2023, <https://www.cdacanada.com/resources/communication-disabilities/statistics/>.
- CDC, 2018, “Disability and Health Data System (DHDS),” Centers for Disease Control and Prevention, Read March 28, 2023, <https://dhds.cdc.gov/LP?CategoryId=DISEST&IndicatorId=STATTYPE&ShowFootnotes=true&View=Map&yearId=YR6&stratCatId1=CAT1&stratId1=BO1&stratCatId2=&stratId2=&responseId=Q6DIS1&dataValueTypeId=AGEADJPREV&MapClassifierId=quantile&MapClassifierCount=5>
- CDC, 2019, “Subjective Cognitive Decline — A Public Health Issue,” Read March 12, 2023, <https://www.cdc.gov/aging/data/subjective-cognitive-decline-brief.html>.
- CDC, 2023, “Disability and Health Overview | CDC,” Centers for Disease Control and Prevention, Read March 20, 2023, <https://www.cdc.gov/ncbddd/disabilityandhealth/disability.html>.

Clark H., 2022, "The Software Development Life Cycle (SDLC): 7 Phases and 5 Models," Read May 10, 2023, <https://theproductmanager.com/topics/software-development-life-cycle/>.

CMMI, 2023, "CMMI Institute - What is CMMI?," Read May 15, 2023, <https://cmmiinstitute.com/cmmit/intro>.

Computer Hope, 2022, "How to Install Software," Read May 12, 2023, <https://www.computerhope.com/issues/ch000561.htm>.

Computer Hope, 2023, "What Are Communications?," Read May 12, 2023, <https://www.computerhope.com/jargon/c/communic.htm>.

Computer Hope, 2023, "What Is Binary?," Read May 16, 2023, <https://www.computerhope.com/jargon/b/binary.htm>.

Dictionary, 2023, "Definition of Good | Dictionary.Com," www.dictionary.com, Read May 20, 2023, <https://www.dictionary.com/browse/good>.

Dictionary, 2023, "Definition of Install | Dictionary.Com," www.dictionary.com, Read May 12, 2023, <https://www.dictionary.com/browse/install>.

Dictionary, 2023, "Definition of Moderate | Dictionary.Com," www.dictionary.com, Read May 20, 2023, <https://www.dictionary.com/browse/moderate>.

Dictionary, 2023, "Definition of Slight | Dictionary.Com," www.dictionary.com, Read May 20, 2023, <https://www.dictionary.com/browse/slight>.

Disabled World, 2022, "Cognitive Disability: Information on Intellectual Disabilities," Read March 28, 2023, <https://www.disabled-world.com/disability/types/cognitive/>.

Disabled World, 2022, "Vision Disability: Types and Information," March 28, 2023, <https://www.disabled-world.com/disability/types/vision/>.

Ericson W. & Camille L., 2008, "2007 Disability Status Reports: United States," <https://ecommons.cornell.edu/handle/1813/90004>.

ESRB, 2023, "ESRB Ratings Guides, Categories, Content Descriptors," ESRB Ratings, Read May 13, 2023, <https://www.esrb.org/ratings-guide/>.

Estimancy Blog, n.d., "Software Requirements: 8 Best Practices to Write Them," November 5, 2018, <https://www.estimancy.com/en/2018/11/05/software-requirements-8-best-practices-to-write-them/>.

Expert Panel®, 2021, "Council Post: 13 Tech Leaders Share Their Favorite Software Development Life Cycle Methodologies," Read May 17, 2023, <https://www.forbes.com/sites/forbestech-council/2021/08/10/13-tech-leaders-share-their-favorite-software-development-life-cycle-methodologies/>.

Fable Team, n.d., "Assistive Technology Glossary," Read May 10, 2023, <https://makeit-fable.com/glossary/>.

Fable, 2021, "Screen Magnifier | What Is Screen Magnifier?," Read May 6, 2023, <https://makeit-fable.com/glossary-term/screen-magnifier/>.

Federal Communications Commission, 2003, "Voice Over Internet Protocol (VoIP)," Read May 4, 2023, <https://www.fcc.gov/general/voice-over-internet-protocol-voip>.

Fisher T., 2022, "What's an EXE File and How Do You Open One?," Read May 21, 2023, <https://www.lifewire.com/exe-file-2622732>.

From under the username of Bloc, 2023, *Future of Elder Scrolls & RPG - Skyrim NPCs & Inworld AI (like GPT-4 for Gaming)*, <https://www.youtube.com/watch?v=d6sVWEu9HWU>.

GeeksforGeeks, "Software Engineering | Software Characteristics," Read May 22, 2023, <https://www.geeksforgeeks.org/software-engineering-software-characteristics/>.

Google Chrome Help, "Manage Passwords - Computer - Google Chrome Help," Read May 21, 2023, <https://support.google.com/chrome/answer/95606?hl=en&co=GENIE.Platform%3DDesktop>.

Google, "Read and Write Data on the Web | Firebase Realtime Database," Read May 21, 2023, <https://firebase.google.com/docs/database/web/Read-and-write>.

Grammarly, 2023, "Responsible AI", Read May 12, 2023, <https://www.grammarly.com/responsible-ai>.

Greelane, 2019, "Mitä nomoteettiset ja idiografiset tarkoittavat ja miksi niillä on merkitystä tutkimuksessa," Read 12 May, 2023, <https://www.greelane.com/fi/science-tech-matematiikka/yhteiskuntatieteet/nomothetic-3026355>.

Hachman M., 2016 "Tobii EyeX Review: The 'Eye Mouse' Is Magical, but Just Not for Everyone," Read May 15, 2023, <https://www.pcworld.com/article/418755/tobii-eyex-review-the-eye-mouse-is-magical-but-just-not-for-everyone.html>.

Hanocka R., n.d, "Visual Computing," Department of Computer Science, Read May 24, 2023, <https://cs.uchicago.edu/research/visual-computing/>.

Hati S., "Top Software Development Frameworks For 2023," Read May 16, 2023, <https://www.knowledgehut.com/blog/web-development/software-development-frameworks>.

Hoffman C., 2012, "5 Ways To Quickly Launch Programs On Windows," Read May 10, 2023, <https://www.howtogeek.com/124109/5-ways-to-quickly-launch-programs-on-windows/>.

IEEE, 2023 "The World's Largest Technical Professional Organization Dedicated to Advancing Technology for the Benefit of Humanity," Read May 21, 2023, <https://www.ieee.org/>.

Inworld AI, 2023, "Inworld – The Developer Platform for AI Characters," Read May 25, 2023, https://inworld.ai/?utm_campaign=bloctheworker-skyrim&utm_medium=referral&utm_source=youtube.

Iordanis J. & Bland M., 2021, "8 Software Development Methodologies Explained," Read May 2, 2023, <https://easyagile.com/blog/software-development-methodologies/>.

Jan P. Vox et al., 2021, "An Evaluation of Motion Trackers with Virtual Reality Sensor Technology in Comparison to a Marker-Based Motion Capture System Based on Joint Angles for Ergonomic Risk Assessment," Read May 4, 2023, <https://doi.org/10.3390/s21093145>.

Kanbanize, 2023, "What Is Kanban? Explained in 10 Minutes | Kanbanize," Kanban Software for Agile Project Management, Read May 14, 2023, <https://kanbanize.com/kanban-resources/getting-started/what-is-kanban>.

Kasanen et al., 1991

Kasanen et al., 1993

Lamintakanen J. et. al, 2008, "Käsiteanalyttisen tutkimuksen soveltaminen terveyshallintotieteessä - esimerkkinä Rodgersin käsiteanalyysimenetelmä," Read May 5, 2023, <https://journal.fi/hallinnontutkimus/article/download/100654/58176/178429>

Lawrenz L. & Tee-Melegrito A., 2023, "Communication Disorders: Definition, Types, Causes, and Symptoms," Read May 8, 2023, <https://www.medicalnewstoday.com/articles/communication-disorders#definition>.

Lombardi P., 2022, "11 Software Development Methodologies (Plus How To Pick One) | Indeed.Com," Read May 27, 2023, <https://www.indeed.com/career-advice/career-development/software-development-methodologies>.

Lukka K., 2014, "Konstruktivinen tutkimusote," Read May 12, 2023, <https://metodix.fi/2014/05/19/lukka-konstruktivinen-tutkimusote/>.

Mattila R., 2020, "Puhumaton Pää", Watched March 10, 2023, <https://youtu.be/EbaSM0QYelg>

Merriam-Webster, 2023, "Definition of ALGORITHM," May 15, 2023, <https://www.merriam-webster.com/dictionary/algorithm>.

Merriam-Webster, 2023, "Definition of ITERATION," May 8, 2023, <https://www.merriam-webster.com/dictionary/iteration>.

Merriam-Webster, 2023, "Definition of SYNTAX," May 1, 2023, <https://www.merriam-webster.com/dictionary/syntax>.

Metwalli S., 2023, "What Is a Graphical User Interface (GUI)? | Built In," Read May 15, 2023, <https://builtin.com/software-engineering-perspectives/graphical-user-interface>.

Microsoft Security, 2023, "What Is Two-Factor Authentication (2FA)?" Read May 15, 2023, <https://www.microsoft.com/en-ww/security/business/security-101/what-is-two-factor-authentication-2fa>.

Microsoft Support, "Use the On-Screen Keyboard (OSK) to Type - Microsoft Support," Read May 8, 2023, <https://support.microsoft.com/en-us/windows/use-the-on-screen-keyboard-osk-to-type-ecbb5e08-5b4e-d8c8-f794-81dbf896267a>.

Microsoft Support, 2023, "Search Indexing in Windows 10: FAQ" Read May 1, 2023, <https://support.microsoft.com/en-us/windows/search-indexing-in-windows-10-faq-da061c83-af6b-095c-0f7a-4dfecda4d15a>.

Microsoft Support, 2023, "Windows Sign-in Options and Account Protection" Read May 11, 2023, <https://support.microsoft.com/en-us/windows/windows-sign-in-options-and-account-protection-7b34d4cf-794f-f6bd-ddcc-e73cdf1a6fbf>.

Microsoft Support, n.d, "Add an App to Run Automatically at Startup in Windows 10 - Microsoft Support," Read May 11, 2023, <https://support.microsoft.com/en-us/windows/add-an-app-to-run-automatically-at-startup-in-windows-10-150da165-dcd9-7230-517b-cf3c295d89dd>.

Muchmore M & Duffy J., 2023 "The Best Cloud Storage and File-Sharing Services for 2023," Read May 20, 2023, <https://www.pcmag.com/picks/the-best-cloud-storage-and-file-sharing-services>.

Nagar T., 2022, "Best Software Development Methodologies: Pros & Cons In 2022," March 19, 2022, <https://devtechnosys.com/insights/software-development-methodologies/>.

National Eye Institute, 2019, "Causes of Color Blindness," Read March 25, 2023, <https://www.nei.nih.gov/learn-about-eye-health/eye-conditions-and-diseases/color-blindness/causes-color-blindness>.

National Library of Medicine, 2020, "Global Cognitive Impairment Prevalence and Incidence in Community Dwelling Older Adults—A Systematic Review - PMC," Read May 8, 2023, <https://www.ncbi.nlm.nih.gov/pmc/articles/PMC7709591/>.

Ninite Documentation, "Ninite Help," Read May 28, 2023, <https://ninite.com/help/>.

Ninite Documentation, "Scripting Ninite | Ninite Help," Read May 28, 2023, <https://ninite.com/help/enterprise/automation.html>.

Ninite, "Ninite - Install or Update Multiple Apps at Once," Read May 28, 2023, <https://ninite.com/>.

Olkkonen, 1994, pages. 72-74

OpenAI, "Introducing ChatGPT," Read May 24, 2023, <https://openai.com/blog/chatgpt>.

Paul Kirvan, "The Top Secure Software Development Frameworks | TechTarget," Security, Read May 18, 2023, <https://www.techtarget.com/searchsecurity/tip/The-top-secure-software-development-frameworks>

Planio, "Software Development Process: How to Pick The Process That's Right For You," Planio, Read May 16, 2023, <https://plan.io/blog/software-development-process/>.

Rouse M., 2014, "Data Retrieval," Read May 10, 2023, <https://www.techopedia.com/definition/30140/data-retrieval>.

Royal National Institute of Blind People, 2016 "Smart Glasses," Read May 15, 2023, <https://www.rnib.org.ukhttps://www.rnib.org.uk/professionals/health-social-care-education-professionals/knowledge-and-research-hub/reports-and-insight/smart-glasses/>.

Scrum, "How Long A Sprint Should Be? | Scrum.Org," Read May 16, 2023, <https://www.scrum.org/resources/blog/how-long-sprint-should-be>.

Scrum, 2023, "What Is Scrum?," <https://www.scrum.org/learning-series/what-is-scrum>

Scrum, n.d, "What Is Scrum?," Read May 17, 2023, <https://www.scrum.org/learning-series/what-is-scrum>.

Soininvaara S., 2018, "School of Business and Management," Read March 12, 2023, https://lut-pub.lut.fi/bitstream/handle/10024/158577/progradu_soininvaara_siiri.pdf

Steve Gray et.al., 2014, "Quick Reference | AutoHotkey V2 Documentation," Read May 28, 2023, <https://www.autohotkey.com/docs/v2/>.

Synopsys Editorial Team, 2023, "Top 4 Software Development Methodologies," Application Security Blog, March 28, 2023, <https://www.synopsys.com/blogs/software-security/top-4-software-development-methodologies/>.

TAMK, 2023, "Accessibility, and equality at TAMK", Read 18 May, 2023, <https://www.tuni.fi/en/about-us/tamk/accessibility-at-tamk/accessibility-and-equality>

TechTarget, 2019, "Flash Memory vs. RAM: What's the Difference?" Read May 18, 2023, <https://www.techtarget.com/searchstorage/feature/Flash-memory-vs-RAM-Whats-the-difference>.

TechTarget, n.d, "What Is Voice Control (Voice Assistance) ? | Definition from TechTarget," Read May 20, 2023, <https://www.techtarget.com/whatis/definition/voice-assistant>.

The American Foundation for the Blind, “An Update on the Finger Reader, an On-the-Go Reading Device in Development at MIT,” Read March 25, 2023, <https://www.afb.org/aw/16/7/15438>.

Tobii EyeX, n.d., “Tobii EyeX – Tobii Help Center,” Read May 28, 2023, <https://help.tobii.com/hc/en-us/sections/360001805718-Tobii-EyeX>.

UNESCO, 2021, “Disability,” <http://uis.unesco.org/en/glossary-term/disability#slideoutmenu>.

Upwork Editorial Team, 2020, “6 Most Common Software Development Methodologies | Upwork,” Read May 27, 2023, <https://www.upwork.com/resources/most-common-software-development-methodologies>.

Upwork Editorial Team, 2020, “What Is the Software Development Life Cycle (SDLC) and Which Method Is the Best? | Upwork,” Read May 20, 2023, <https://www.upwork.com/resources/what-is-software-development-life-cycle>.

Virtanen A., 2022 “Konstruktivinen tutkimusote Miten koulutus ja elinkeinoelämän odotukset kohtaavat ammattikorkeakoulun opinnäytetöissä,” *Ammattikasvatuksen aikakauskirja* 8, no. 1 (2006): 46–52, <https://journal.fi/akakk/article/view/114874>

Visual Studio Code, “Syntax Highlight Guide,” Read May 28, 2023, <https://code.visualstudio.com/api/language-extensions/syntax-highlight-guide>.

Visual Studio Code, 2023, “Visual Studio Code Tips and Tricks,” Read May 22, 2023, <https://code.visualstudio.com/docs/getstarted/tips-and-tricks>.

W3C Web Accessibility Initiative (WAI), 2023, “Diverse Abilities and Barriers,” Web Accessibility Initiative (WAI), <https://www.w3.org/WAI/people-use-web/abilities-barriers/>.

W3C Web Accessibility Initiative (WAI), 2023, “Stories of Web Users,” Web Accessibility Initiative (WAI), <https://www.w3.org/WAI/people-use-web/user-stories/>.

W3C Web Accessibility Initiative (WAI), 2023, “Tools and Techniques,” Web Accessibility Initiative (WAI), <https://www.w3.org/WAI/people-use-web/tools-techniques/>.

Washington DO-IT, 2022, “How Can people with mobility impairments operate computers?”, Read March 24, 2023, <https://www.washington.edu/doit/how-can-people-mobility-impairments-operate-computers>

Washington Group, “How Can People with Mobility Impairments Operate Computers? | DO-IT,” Read March 25, 2023, <https://www.washington.edu/doit/how-can-people-mobility-impairments-operate-computers>.

Washington Group, “WG Short Set on Functioning (WG-SS), 2022,” The Washington Group on Disability Statistics, Read March 28, 2023, <https://www.washingtongroup-disability.com/question-sets/wg-short-set-on-functioning-wg-ss/>.

WHO, “Assistive Technology,” Read March 25, 2023, <https://www.who.int/health-topics/assistive-technology>.

WHO, 2018, “Improving Access to Assistive Technology,” Read March 22, 2023.

WHO, 2022, “Vision Impairment and Blindness,” Read May 1, 2023, <https://www.who.int/news-room/fact-sheets/detail/blindness-and-visual-impairment>.

WHO, 2023, “Deafness and Hearing Loss,” Read March 27, 2023, <https://www.who.int/news-room/fact-sheets/detail/deafness-and-hearing-loss>.

WHO, 2023, “Disability,” Read March 24, 2023, https://www.who.int/health-topics/disability#tab=tab_1, <https://www.who.int/news-room/fact-sheets/detail/disability-and-health>

WHO, n.d., “Self-Care Interventions for Health,” Read March 28, 2023, <https://www.who.int/health-topics/self-care>.

Yang Li et al., 2019, “Gesture Interaction in Virtual Reality,” *Virtual Reality & Intelligent Hardware* 1, no. 1: 84–112, <https://doi.org/10.3724/SP.J.2096-5796.2018.0006>.

Yoshi S., 2021, “What Is Data Manipulation? How It Helps Users Organize Data,” G2, Read May 15, 2023, <https://www.g2.com/articles/data-manipulation>.

APPENDICES

Heatmap Excel file as a copy:



Heatmap Matrice
for thesis done in m

Link to Excel file:

https://mega.nz/file/IOozRDZa#Glf_XLxMWDYMzw8DHYunZcvbtOvtU_t-buheKCp3-cl