Gottfried Wilhelm
Leibniz Universität Hannover
Fakultät für Elektrotechnik und Informatik
Institut für Praktische Informatik
Fachgebiet Software Engineering

Investigating Communication Problems in Software Projects by means of an Interview Study

Untersuchung von Problemen durch Kommunikation in Softwareprojekten basierend auf einer Interviewstudie

Bachelorarbeit

im Studiengang Informatik

von

Nidal Houchaime

Prüfer: Prof. Dr. Kurt Schneider
Zweitprüferin: Dr. Jil Klünder
Betreuerin: Dr. Jil Klünder

Hannover, 23.02.2024

Erklärung der Selbstständigkeit

Hiermit versichere ich, dass ich die vorliegende Bachelorarbeit selbständig und ohne fremde Hilfe verfasst und keine anderen als die in der Arbeit angegebenen Quellen und Hilfsmittel verwendet habe. Die Arbeit hat in gleicher oder ähnlicher Form noch keinem anderen Prüfungsamt vorgelegen.

Hannover, den 23.02.2024

Nidal Houchaime

I hereby certify that the following Bachelor thesis has been produced independently and without external help apart from the sources and aids used in this thesis. This thesis has not been submitted for examination by any other exam committee in a same or similar manner.

Hannover, the 23.02.2024

Nidal Houchaime

Zusammenfassung

Untersuchung von Problemen durch Kommunikation in Softwareprojekten basierend auf einer Interviewstudie

In der modernen Softwareentwicklung spielt Kommunikation eine wichtige Rolle und ist ein entscheidender Faktor für den Erfolg eines Softwareprojekts. Aufgrund des dynamischen Charakters und der Abhängigkeit der modernen Softwareentwicklung von der Kommunikation zwischen Kunden und Entwicklern, sowie innerhalb des Entwicklungsteams selbst, sollte die Bedeutung der Erkennung und Lösung von Kommunikationsstörungen nicht unterschätzt werden. Ziel dieser Arbeit ist es, Konflikte in der Kommunikation, die während des Softwareentwicklungsprozesses auftreten können, im Kontext agiler Entwicklungsmethoden und der Entwicklung durch verteilten Teams näher zu untersuchen.

Abstract

Investigating Communication Problems in Software Projects by means of an Interview Study

Communication plays an important role in modern software development, and is a critical factor in determining the success of a software project. Due to the dynamic nature and reliance of modern software development on communication between clients and developers, as well as within the development team itself, the importance of detecting and resolving communication breakdowns should not be underestimated. This thesis aims to take a closer look at communication conflicts that can occur during the software development process by examining it within the context of agile development methods and distributed development teams.

Contents

1 Introduction **1**
1.1 Motivation . . . 1
1.2 Problem Statement . . . 1
1.3 Solution Approach . . . 2
1.4 Thesis Structure . . . 2

2 Background and Related Works **3**
2.1 Communication . . . 3
2.1.1 Formal Communication . . . 3
2.1.2 Informal Communication . . . 3
2.1.3 Mid-iteration Communication . . . 4
2.1.4 Communication in Development Teams . . . 4
2.1.5 Communication in Software Development Practices . . . 5
2.1.6 Media Richness Theory . . . 5
2.2 Conflict . . . 6
2.2.1 Task Conflicts . . . 6
2.2.2 Relationship Conflicts . . . 6
2.2.3 Customer Dependent Defects . . . 6
2.2.4 Customer Independent Defects . . . 6
2.2.5 Conflicts within Development Teams . . . 6
2.2.6 Conflicts in Distributed Development Teams . . . 7

3 Data and Literature Analysis **9**
3.1 Data Analysis . . . 9
3.1.1 Data Origin . . . 9
3.1.2 Manual Inspection . . . 10
3.1.3 Grounded Theory . . . 12
3.1.4 Research Questions . . . 15
3.2 Literature Analysis . . . 18
3.2.1 Database Search . . . 18
3.2.2 Literature Selection Criteria . . . 19
3.2.3 Refined Search . . . 19
3.2.4 Summary of Database Search . . . 21

4 Research Design **23**
4.1 Communication Conflicts in Communication Channels 23
4.1.1 Scrum 23
4.1.2 Extreme Programming 26
4.1.3 Client Communication 27
4.1.4 Defects 28
4.2 Communication Problems in Distributed Development Teams 29
4.2.1 Temporal Distance 29
4.2.2 Geographical Distance 30
4.2.3 Trust 31
4.2.4 Socio-cultural Distance 32

5 Discussion **35**

6 Conclusion and Future Works **39**
6.1 Conclusion 39
6.2 Future Works 40

A Appendix **41**
A.1 Keyword Mind Map 41

Chapter 1

Introduction

1.1 Motivation

Due to modern day organizations shifting from traditional software development practices towards agile software development practices[29], the need to examine the social aspects of software projects has gained significant traction within the field of software engineering. One such social aspect that plays an important role in software projects is communication, and is a good indicator for the success (or failure) of a software development project([31] as cited in [36]).

In an interview study conducted by David Sasse[34] that investigated social aspects in software projects, a key component that was discovered as a result of the interviews, was the fact that communication problems would affect multiple social aspects such as team work, error management, conflict management, and the relationship to the client. While the aforementioned master thesis investigated social aspects, this thesis aims to delve deeper into what communication problems arise during the software development process.

1.2 Problem Statement

One of the main reasons to shifting from traditional software development methods to agile software development methods is the emphasis on individuals, interactions, and customer collaboration[29, 16]. All three of these characteristics of agile software development require communication in one form or another. Moreover, according to Misra et al.[29], communication, more specifically face-to-face communication, plays an important role in agile development methods. The importance and need for communication in software development is therefore unavoidable and should be explored. Furthermore, in order to understand how communication occurs, it is as important to explore communication conflicts and problems. After all, as mentioned above, a big factor in the success of a software development project is communication. Based on existing literature spanning multiple scientific fields including computer science and psychology, the goal of this thesis is to identify communication issues that repeatedly occur during a software development project and how they can be best detected and possibly avoided and resolved.

1.3 Solution Approach

This thesis investigates communication problems and breakdowns in relation to software development. Starting with the interview transcripts provided by David Sasse[34], factors discussed by the interviewees relating to communication were highlighted and analyzed. Thereafter, upon analyzing the relevant passages, keywords were identified and codified accordingly. Based on this codification, research on existing scientific literature regarding communication in the context of software development teams, communication tools and channels, as well as communication methodologies and communication conflicts took place. Other related topics based on the aforementioned codification that stood out and were deemed to be important such as personalities and team constellations were also examined.

From the data gathered by analyzing the interviews and the existing scientific literature, suggestions on detecting and mitigating communication conflicts in time will be made. Furthermore, possible areas of conflict that could occur will be discussed and suggestions will be offered. Lastly, possible improvements on already existing communication tools and methodologies will be suggested.

1.4 Thesis Structure

This thesis spans 6 chapters. Chapter 3 goes into detail about how data was acquired and the decisions made in choosing the appropriate literature. Chapter 2 focuses on the background information and explanations needed to understand communication flow and exchange based on related scientific literature. Chapter 4 is the core of this thesis and discusses communication and communication problems related to software development. Chapter 5 discusses the observations made as well as possible improvements and suggestions that could be applied. Chapter 6 is the conclusion of this thesis and contains suggestions for possible future work.

Chapter 2

Background and Related Works

In this chapter, background information that is pertinent to later chapters of this thesis will be provided. The topics that will be covered are communication and conflicts. The communication section discusses communication in the context of software development, software development teams, and media richness theory. The conflicts section will delve into the topics of the relationship between personality traits and conflicts, as well as conflicts within software development teams.

2.1 Communication

In the context of software development, communication is the process of people from diverse backgrounds working on a project together agreeing on building something together while sharing information and merging activities[26]. Goles et al.[17] argue that since communication is a two-way process and the flow of information is multi-directional, it should not only extend to day-to-day operations and routines, but also go beyond that to allow for open communication of desires and needs.

2.1.1 Formal Communication

Formal communication is the explicit communication that occurs throughout the software development process[31](as cited in [20]).

2.1.2 Informal Communication

Informal communication is the explicit communication that occurs amongst team members(i.e. conversations)[31](as cited in [20]).

2.1.3 Mid-iteration Communication

Mid-iteration has been defined by Korkala et al.[25] as "communication that is used for the customer-developer communication taking place inside any particular iteration".

2.1.4 Communication in Development Teams

When it comes to communication in traditional and agile development, their methodologies differ. Traditional development focuses on skill sets - each team member specializes in certain tasks (programming, unit testing, documentation etc.) [2], and is therefore less people-oriented than its agile counterpart. For the most part, traditional development relies heavily on documentation as the main means of communication in software projects [2].

Agile development focuses on team member interactions, social, and interpersonal skills of software developers[2]. Furthermore, according to Balijepally et al.[2], the development team is "collectively responsible" when it comes to all aspects of the development process.

Communication in agile teams can occur as either a form of collaboration to accomplish a certain task, or a discussion to solve problems [27, 39]. In addition to collaborations and discussions, Marjaie et al.[27] define communication in agile teams as a series of shared experiences, achievements, and the current status of tasks or projects.

Moreover, when agile teams need to communicate "concepts, ideas, or desires", Melnik et al.[28] posit that verbal channels of communication tend to be more efficient because of the ability to provide "rapid mutual feedback". Furthermore, similar to Marjaie et al.[27], Melnik et al.[28] suggest that "conversation and social interaction help to create common knowledge out of an experience". Through the use of "high-touch" communication and "high velocity knowledge sharing", agile teams are capable of delivering the product to the customer sooner than traditional development teams [28].

The success of a development team is dependent on collaborative teamwork[2]. In order to foster a collaborative environment, Balijepally et al. state that trust and goodwill are a "key requirement" between team members[2]. A software developer's ability to work efficiently as a member of the development team, to be able to communicate and collaborate with team members and share responsibilities – rewards as well as failures – are "critical determinants" for a successful project in an agile environment[2]. Moreover, similar to Balijepally et al.[2], Marjaie et al. stress the importance of communication amongst team members as being a "facilitative platform" for team collaboration[27].

2.1.5 Communication in Software Development Practices

According to Melnik et al.[28], agile methods focus on improving communications for both team members(internally) and clients alike, through a continuous feedback loop. Furthermore, Melnik et al.[28] reiterate that acquiring knowledge in agile development is a "highly social (rather than technical/mechanical/formal) process".

2.1.6 Media Richness Theory

Rich Communication

Communication that can explain ambiguous problems in a certain time frame is considered rich, whereas communication that takes longer to process and understand is considered to be low in richness[13].

Communication Media Ranking

Daft and Lengel[13] proposed a communication media ranking system based on communication richness: (1) face-to-face, (2) telephone, (3) personal documents, (4) impersonal documents, (5) numeric documents. Face-to-face communication ranked highest due to its ability to provide the immediate feedback necessary, as well as being able to provide important social cues through body language, tone of voice, and natural language [13]. Similar to Daft and Lengel[13], Melnik et al.[28] refer to a number of scientific fields that "independently analyzed" the problem with different communication mediums that are being used amongst team members, wherein it has been "unanimously contend[ed]" that face-to-face communication was the most effective communication style.

Alistair Cockburn's Modes of Communication curve which was reproduced by Melnik et al.[28], gives important insight into the effectiveness of communication based on the communication medium that was used. It is important to note, that Melnik et al.[28] caution that the "informal curves" used are based on "project experiences" and have not been "scientifically substantiated".

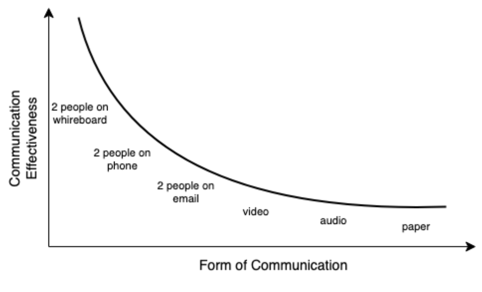

Figure 2.1: Snippet of the Keyword Mind Map

2.2 Conflict

A conflict is a situation of competition wherein parties are aware of their incompatibilities and acknowledge that their wishes do not align with that of the others[6].

2.2.1 Task Conflicts

Task conflicts are disagreements (i.e.: differing views, ideas and opinions) relating to the contents of a task[23]. Task conflicts can have a positive effect on team performance[2].

2.2.2 Relationship Conflicts

Relationship conflicts are interpersonal incompatibilities amongst team members[23]. These incompatibilities can range from tensions or annoyances to outright animosity towards each other [23]. Relationship conflicts tend to be detrimental to a team's performance and effectiveness[2].

2.2.3 Customer Dependent Defects

Customer dependent defects can be traced back to insufficient customer feedback and communication, which can be avoided by continuous customer communication, feedback, and choosing an efficient mid-iteration communication method[25].

2.2.4 Customer Independent Defects

Customer independent defects are defects that can be tracked back to the development team[25].

2.2.5 Conflicts within Development Teams

Depending on project complexity and team size, a number of communication problems can occur[27]. Part of the difficulty lies in the amount of communication channels available and individual team members' ability and choice of which ones to use[27]. The size of a team is important when factoring in the complexity of communication. While smaller teams have an easier time communicating, by increasing the size of a team, communication becomes more challenging[27].

Furthermore, certain aspects of agile development methods might lead to task conflicts, and if not resolved adequately, could bleed over into relationship conflicts with negative consequences for the development team [2]. Transitioning from a traditional development environment to an agile one which tends to be more collaborative and less individualistic can also lead to conflicts [2].

Moreover, Trimmer et al.[38](as cited in [30]) argue that differing backgrounds and skills, as well as team member's roles and expectations of each other not always being aligned with one another, can cause potential conflict amongst team members[2].

Having too many extraverts in a development team can lead to conflicts, since extraverts enjoy teamwork mainly for the socialization aspect of it and tend to dominate the conversation compared to their introverted peers [2]. Balijepally et al.[2] caution that the effect of individual personalities of team members on the development team is "highly dependent" on their tasks.

2.2.6 Conflicts in Distributed Development Teams

In addition to conflicts that arise in co-located development teams, distributed development teams face the added challenges of temporal distance, geographical distance, and socio-cultural distance.

Temporal Distance

Ågerfalk et al.[40] define temporal distance as a "directional measure of the dislocation in time experienced by two actors wishing to interact".

Geographical Distance

Ågerfalk et al.[40] define geographical distance as a "directional measure of the effort required for one actor to visit another at the latter's home site".

Socio-cultural Distance

Ågerfalk et al.[40] define socio-cultural distance as a "directional measure of an actor's understanding of another actor's values and normative practices".

Chapter 3

Data and Literature Analysis

This chapter delves into the methodologies used, data acquired, and the subsequent literature chosen for this thesis. As the interview transcripts provided by David Sasse[34] were the starting point for this thesis, they were analyzed using grounded theory. A description and application of this methodology will be discussed in the following sections.

Furthermore, based on the results of analyzing the interview transcripts, a database search takes place to find available relevant literature surrounding the topics of communication that were coded and categorized through the aforementioned process. Beginning with the database engine as a general research starting point, a more detailed search query is conducted to refine this thesis' core discussion points.

3.1 Data Analysis

3.1.1 Data Origin

The data for this thesis originate from interview transcripts provided by David Sasse[34]. These interview transcripts were used in his Master's thesis *Interview Study on Social Aspects in Modern Software Projects* that inspected social aspects and interpersonal problems within software projects. The Master's thesis aimed to develop concepts to aid in creating a more effective software development environment and a higher rate of employee satisfaction. David Sasse[34] conducted an interview study made up of 20 participants working in the field of software development. Amongst the data collected, a big part of the dependencies of social aspects and interpersonal relationships relied on communication. Based on a diagram developed by David Sasse[34], communication plays a part in teamwork, cultural diversity, mistake culture, knowledge transfer, relationship to the client, and trust. Since this thesis focuses on communication conflicts within software development, these interview transcripts provide important insight into communication and a wealth of information as a starting point.

3.1.2 Manual Inspection

Preceding the use of Grounded Theory for a more qualitative analysis of the data, a preliminary analysis of all possible keywords relating to communication and its surrounding contexts, that were encountered during the reading of the interview transcripts, and deemed important, took place.

Some interesting observations made from this manual inspection is the fact that 75% of participants had a hybrid work setup, 70% of the participants suggested mediation as a method for conflict management, 60% of participants discussed open communication and transparency being important when working on software projects, as well as mentioning the importance of trust in software development relating to team members and project management, 55% related to the importance of having a team that works well with each other and the difficulties of maintaining team member relationships due to working from home, and finally, 50% of the participants discussed the importance of informal communication i.e. "water cooler talk" and whether team building events are worth the time and effort.

Keywords	Mentions
Agile/Traditional methods	5
Client relationship project success	3
Communication channels	5
Communication guidelines	1
Communication styles	2
Conflict detection	1
Conflict Management	6
Conflict Solution	1
Conflict vs. Crisis	1
Cultural diversity	4
Documentation	6
Ego	2
Empathy	2
Envy	1
Fear	6
Hierarchies	2
HO team relationship	11
Home Office	3
Hybrid	15
Informal vs. Formal	10
Interpretation miscommunication	3
Introversion vs. Extraversion	6
Mediation	14
Mental Health	4
Mistake culture	2
Office	1
One-on-one conversations	8
Open communication / Transparency	12
Pair Programming	1
Personality clashes	7
Project management communication breakdown	9
Redundant information	5
Reflection	7
Selfishness	1
SWQ	1
Sympathy	1
Team building events	10
Team communication	4
Team Constellation	11
Teamwork	7
Trust	12
Work-life balance	2
Wrong information	3

Table 3.1: Keywords and the number of mentions

Mind Map

The diagram below is a snippet of the mind map that was created using the keywords that were attained during the manual inspection. It was designed for a better visualization on the relationship between the prototype categories and communication. For the full mind map please refer to Appendix A.1.

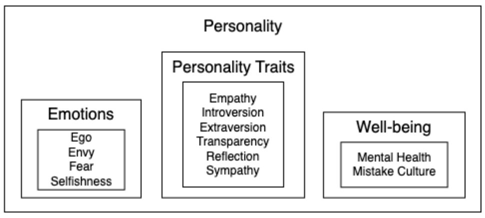

Figure 3.1: Snippet of the Keyword Mind Map

3.1.3 Grounded Theory

Grounded theory is a method used to analyze qualitative data and designing theories based on that data[32]. According to Saracho et al.([32]as cited in [35]), inductive grounded theory is the concept of developing "abstract conceptual categories" and identifying "patterned relationships within it". This is achieved through multiple levels of analysis that are each based on the previous one by using comparative, interactive, and iterative methods[32].

Grounded theory "specifies strategies and methods of *analysis*" that can be applied to multiple data collection methods, including but not limited to, interviews and documents[32] (which are the core part of this thesis). Furthermore, during the analysis phase, codes and concepts are established that encourage further data collection[32]. In summary, grounded theory aids in identifying the context surrounding the data and "focuses the researcher's eye on how meaning is being constructed"[32]. Part of grounded theory is the Coding phase, which will be discussed in the next section.

Coding

Coding is a method used to evaluate and organize data in an attempt to comprehend texts[10]. It aids the researcher in identifying categories and patterns, as well as forming new connections[10]. These connections allow the researcher to identify similar relationships within the data[10]. Clifford et al.[10] argue that by identifying these categories and patterns, it allows the researcher to get familiarized with the data and therefore start asking new questions.

Since this thesis focuses on communication problems and its surrounding context within software development, the initial step before the coding phase was to highlight the relevant passages of the interview transcript.

Initial Coding

Initial coding is the first step of the coding phase[32]. In this step of the coding phase, researchers tend to move quickly but thoroughly through the data, and more importantly, remain open to further examination of the interpreted data[32]. Moreover, in the initial coding step, data is analyzed "word by word and line by line" wherein researchers interpret the data and construct initial codes as a "single word, a couple of words, a sentence, or a couple a sentences"[32].

The following is an excerpt of the initial coding of the transcript interview of Participant-ID: 01 conducted by David Sasse[34]:

Transcript Data	Initial Coding
01: Also vorgesehen war natürlich alles in Präsenz, aber seit Corona wurde es komplett umgestellt[...]dann kriegt man die Leute nicht alle an einen Standort. Deswegen haben wir in der Regel hybride Veranstaltungen. Also ganze selten haben wir komplett Präsenz.	work environment
01:[...]die Zusammenarbeit, Kommunikation waren für mich, sind für mich ganz wichtig und die standen schon ganz oben	important factors for working in a team
01: Also das war halt, das war halt das Problem. Im Team selber lief es eigentlich sehr gut. Da hat auch jeder, also da hat jeder sehr gut zusammengearbeitet. Das Vertrauen war halt da. Nur zwischen dem, zwischen dem Team und Projektleitung zu Auftraggebern gab es einen Bruch, der dann die Ziele gefährdet hat oder gefährdet.	communication breakdown due to external factors i.e. project management and client can lead to project slowdown/failure

Table 3.2: Initial Coding Interview Transcript 01.

Some concepts that were obtained from the initial coding of this interview were: development team, project management, communication conflicts, interpersonal conflicts.

Focused Coding

Upon having concluded the initial coding step, the next step is focused coding. Focused coding aids in identifying codes and indexing the data according to their meanings, actions and processes[32]. This method of coding enables researchers to remain open to the possibility of identifying more than one recurring initial code that encourages more data gathering and coding[32]. Furthermore, researchers should also remain open to modifying the codes, should the need arise[32]. Saracho et al.[32] discuss the fact that focused codes are more "directed, selective, abstract, and conceptual than the initial codes". With the conclusion of focused coding, frequent themes should arise[32]. Saracho et al.[32] comment on the fact that the data collected during the coding phase might seem "nonsensical" but also point to the fact that "all data is meaningful".

The following is an excerpt of the focused coding from interview transcript 01:

Transcript Data	Focused Coding
01: Also vorgesehen war natürlich alles in Präsenz, aber seit Corona wurde es komplett umgestellt[...]dann kriegt man die Leute nicht alle an einen Standort. Deswegen haben wir in der Regel hybride Veranstaltungen. Also ganze selten haben wir komplett Präsenz.	work environment
01:[...]die Zusammenarbeit, Kommunikation waren für mich, sind für mich ganz wichtig und die standen schon ganz oben	teamwork
01: Also das war halt, das war halt das Problem. Im Team selber lief es eigentlich sehr gut. Da hat auch jeder, also da hat jeder sehr gut zusammengearbeitet. Das Vertrauen war halt da. Nur zwischen dem, zwischen dem Team und Projektleitung zu Auftraggebern gab es einen Bruch, der dann die Ziele gefährdet hat oder gefährdet.	communication conflicts, project management conflicts

Table 3.3: Focused Coding Interview Transcript 01.

Some recurrent themes that were discovered during the focused coding step include: work environment, teamwork, communication conflicts, project management.

Coding Results

Based on the findings from the coding phase of the grounded theory method, two mind maps were conceptualized. The mind maps can be divided into two phases: in the first phase, the codes were grouped together based on relevance and context. They were then "merged" into more abstract concepts to allow for a wider array of exploration. Moreover, after further analyzing the codes that emerged from the first phase, an interesting observation was made: since multiple categories were merged into communication conflicts, the remaining categories seemed to all be connected to and lead back to communication conflicts. The three remaining categories are: teams, communication channels, and conflict management.

Category Descriptions

Teams: Teams make up the most important part of any software development project; they can make or break a project. This thesis defines teams as not only the development team itself, but also other aspects that make up the team. These include but are not limited to project management, team collaboration, team constellation, distributed teams, and personality characteristics.

Communication channels: Communication channels include communication tools (email, telephone etc.) used as a standalone to facilitate collaboration and within communication methodologies (i.e. agile methods) that are used by development teams. Communication channels are just as important as teams; if any communication conflicts arise during their application it could lead to communication breakdowns and in the worst case scenario to project failure.

Conflict management: Conflict management is geared towards resolving communication conflicts and guidelines on improving communication. If conflicts remain unresolved, it could slow down the project and possibly increase the time needed to finish the project, which in turn would mean increased financial costs for the client. Just like *teams* and *communication channels*, the lack of conflict management could lead to project failure.

It is important to note that these descriptions are only imagined scenarios to offer a justification in choosing these categories and are not scientifically verified. This will be explored in later chapters of this thesis.

3.1.4 Research Questions

Initial Research Questions

Just like in Section 3.1.2, some initial questions started to form while writing some notes and memos. Saracho et al.[32] describe the concept of note taking and writing memos as an important step in remembering ideas and analysis, as well as helping researchers find clarity in topics and reflecting upon them. By writing things down, it allows for "codes, categories, thoughts, reflections, and ideas" to become more manageable and allows for additional theorizing[32].

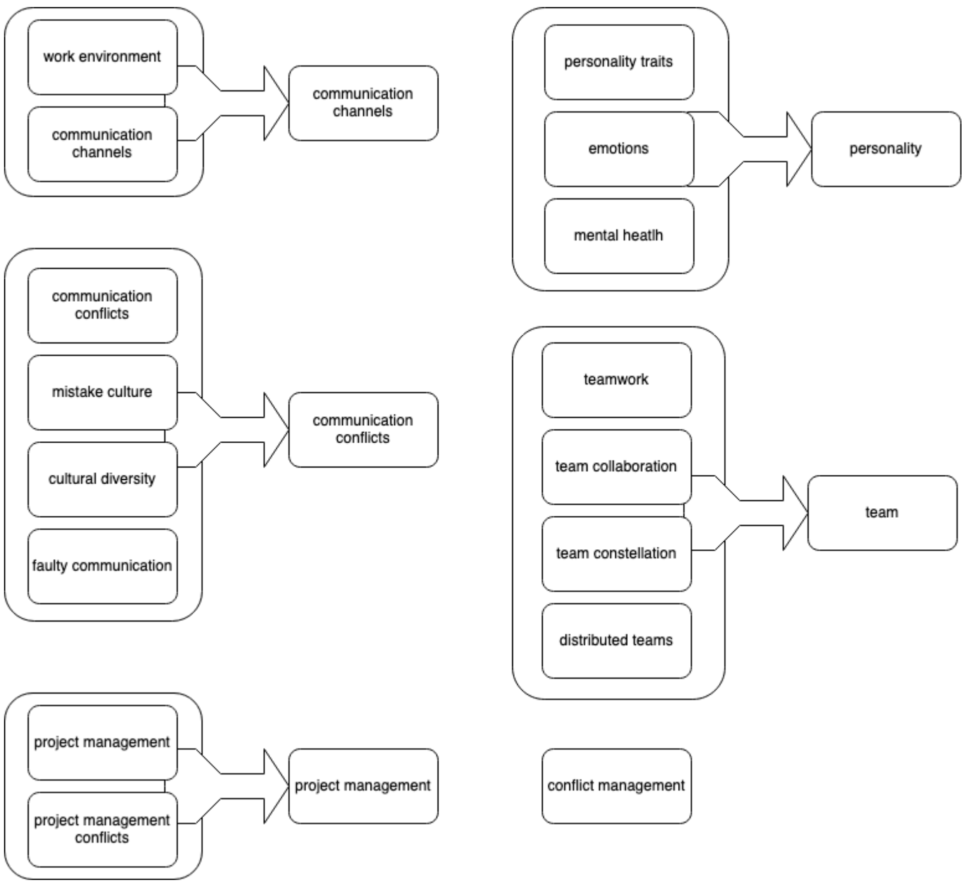

Figure 3.2: "Merging" Codes

Initial Research Questions
When/why does the conflict phase occur?
How does envy possibly lead to communication difficulty?
How does sympathy/not standing each other affect communication?
How important is team constellation in terms of communication?
Why has HO made communication between people different/worse?
Flat hierarchies vs. traditional hierarchies, how does that affect information flow?
How effective are communication seminars? Are online/in-person team building exercises/events to better communication useful?
Are quarterly gatherings worth it?
What is good communication?
How does HO work-life balance affect private relationships?
Communication with clients important to project success?
How important is privacy?
Agile vs. Traditional methods, what is better for communication?
Why do projects fail?
Conflict vs. crisis, what is the tipping point?
Conflicts are subjective. Is there a way to model them?
Are professional conflicts easier to solve than personal ones?
Why are people afraid of admitting mistakes or admitting it is something they can not do?

Table 3.4: Initial Research Questions

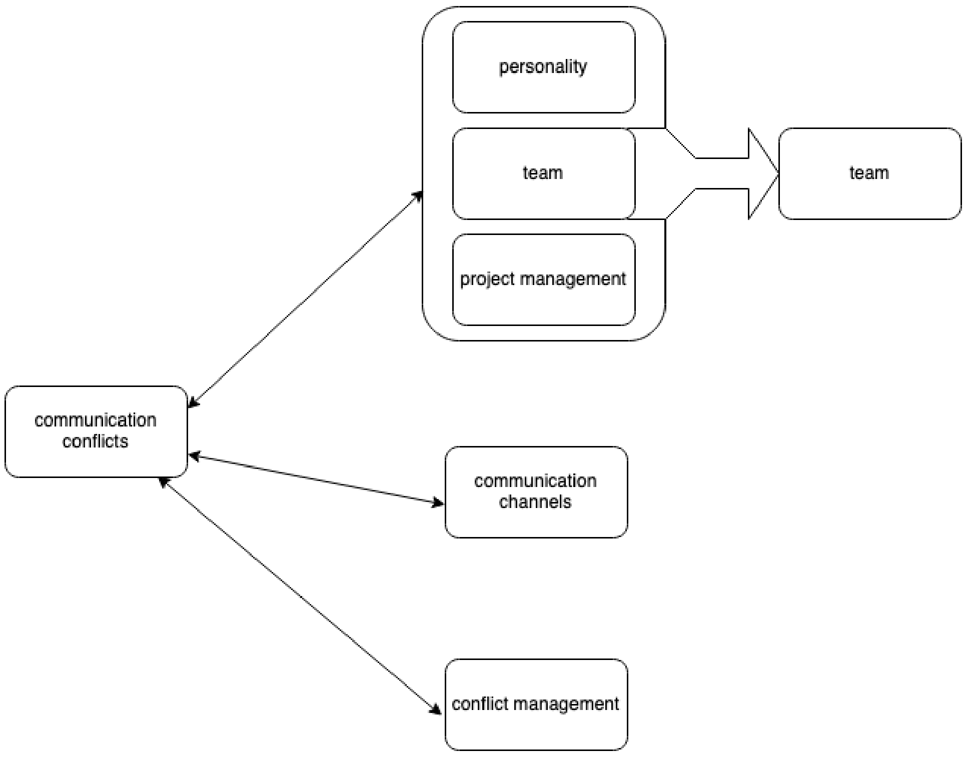

Figure 3.3: Final Categories

Final Research Questions

With the help of grounded theory in applying qualitative data analysis, and by taking the results from the coding phase and memo taking into account, two questions relating to possible communication problems in software development were defined:

RQ1: What communication conflicts can occur within communication channels?

RQ2: What communication conflicts can occur within distributed development teams?

Rationale

RQ1: Communication Channels

Communication channels make up a significant part of software development projects. They are used to aid teams to self-organize, structure, and collaborate on the software project they are currently working on. In Chapter 4 various communication channels used by development teams will be discussed in an attempt to identify communication conflicts that occur through their usage.

RQ2: Distributed Teams

Through initial analysis of the interview transcripts, and the additional analysis using grounded theory, it was observed that: 1) the majority of participants had a hybrid or even full home-office work environment, and 2) teams play a big role in communication conflicts. Taking both factors into account, the need to explore communication conflicts within distributed development teams arose.

3.2 Literature Analysis

While Section 3.1 dealt with data analysis and establishing research questions, this section discusses the methods wherein the literature for this thesis was acquired.

3.2.1 Database Search

Preliminary Search Queries

A preliminary search was undertaken using *Google Scholar*[1]. The first search query was *communication* – this was in part out of curiosity to see what results would come up, as well as to see if anything relating to communication and software development will be discovered in the search results.

Needless to say, the search query returned 10 million results, and most of the literature was from the social sciences and psychology fields. While small parts of this thesis will explore psychological aspects relating to software development, it is not at the core of it. The search results from this initial query confirmed the need for stricter search criteria by keeping the categories defined in Section 3.1 as well as the research questions in mind.

Preceding the use of grounded theory to develop categories and research questions, the following is a small list of search queries that took place. The results returned were less than optimal, and the refined search criteria will be discussed in the sections below.

[1]https://scholar.google.de/

Search Queries
Communication AND team AND “software projects”
Communication AND project AND management AND software
Communication AND breakdown AND project AND management AND software AND development
Conflict AND phase AND software AND project
Mediation AND team AND conflict AND software AND project
Introversion AND extraversion AND software AND development AND team
Home AND office AND software AND development

Table 3.5: Initial Search Queries

3.2.2 Literature Selection Criteria

This section defines a set of inclusion and exclusion criteria for the selection of the literature for this thesis. The use of inclusion and exclusion criteria helps in setting limits and range on the selection of the literature. This selection criteria aided in narrowing down the literature acquired for this thesis.

Inclusion Criteria:

- Peer reviewed literature
- The literature must be relevant to software engineering and related to:
  - Communication (ex.: communication theory, communication tools,..)
  - Communication conflicts (ex.: team relationships, personalities,...)
  - Conflict resolution (ex.: mediation, guidelines,...)
- Secondary literature must be in relation or tangential to software engineering.

Exclusion Criteria:

- Published literature in a language other than German or English.
- Literature that was peer reviewed but still deemed unreliable due to suspicions of plagiarism

3.2.3 Refined Search

Upon defining the criteria for selecting the literature for this thesis, it was imperative, based on preliminary search queries, that the next step was to hone in on related keywords based on communication channels, teams, and conflicts. Below is a sample of related keywords:

Problems:	conflicts, defects, issues, threats, detriments, breakdowns, performance, challenges
Teams:	distributed teams, development teams, project management, intragroup, globally distributed, collaborative, virtual teams, outsourced, multicultural, multinational
Communication channels:	agile practices, agile methods, scrum, documentation, XP, extreme programming, daily-meetings, stand-ups

The next step of the literature search refinement was to make use of *Google Scholar*'s advanced search feature to narrow down the number of results available. Some examples of search queries will be described below.

Agile Practices

allintitle: communication **AND** ("agile software development" **OR** "extreme programming" **OR** "XP")

This search query is a part of a series of search queries undertaken to find relevant literature related to communication in agile development practices. The first term of the query *communication* ensures that literature related to communication will be included in the title. The second part of the search query *"agile software development"* ***OR*** *"extreme programming"* ***OR*** *"XP"* ensures that one of these search terms will be included in the title. As there are different ways of describing the term *extreme programming* it was important to include them in the search. The search query returned 51 results (excluding 2 Spanish results), out of which 12 were subject to manual inspection, and 3 were included as part of the literature for this thesis.

Total	Examined	Included
51	12	3

Table 3.6: Search results for Extreme Programming

Project Management

allintitle: "project management" **AND** ("software development" **OR** "software projects" **OR** "teams")

This search query is related to communication in teams. Since project management plays an important part in the context of software development, the search query used was kept intentionally broad and inspected manually for relevancy in the context of communication and conflicts for this thesis. Relevant secondary and supplemental literature was extracted from primary literature acquired through the search query. The first part of the search query *project management* ensures that project management is included in the title. The second part of the query *("software development"* ***OR*** *"software projects"* ***OR*** *"teams")* ensures that one of the search terms will be included in the title. The search query returned 629 results, 20 of which were manually inspected for relevancy, and 3 were included in the literature for this thesis.

Total	Examined	Included
629	20	3

Table 3.7: Search results for Project Management

3.2.4 Summary of Database Search

Upon conclusion of the database search, a total of 18 articles relating to communication and communication conflicts were included for this thesis. It is important to note that, similar to the grounded theory method introduced in Section 3.1.3 that was used for the data analysis portion of this chapter, the literature was consistently compared with other literature for relevancy and disregarded or included if necessary.

Chapter 4

Research Design

4.1 Communication Conflicts in Communication Channels

This section aims to examine the communication flow and communication conflicts that can occur in agile development practices. The agile development practices chosen for this thesis are Scrum and Extreme Programming.

4.1.1 Scrum

Case Study

In a case study conducted by Stray et al.[37], they analyzed meeting transcripts of two agile software development teams, over a period of eight meetings, in order to get a better understanding of their decision making, communication, and collaboration. It is important to note that for this thesis, only relevant information pertaining to communication was explored. Possible communication conflicts that can occur will be discussed in Chapter 5.

Daily Meeting Breakdown

Upon analyzing the transcripts, it was observed that 35% of the daily scrum was spent on asking to elaborate on problems and trying to find possible solutions for them[37]. Several team members partook in these discussions and many decisions were made during these meetings[37].

Furthermore, 24% of the meeting was devoted to answering the three main scrum questions wherein each team member expanded upon what they have done since the last meeting, what they will be doing, and if they encountered any obstacles[37].

Additionally, 11% of the meeting consisted of project management[37]. This mainly consisted of discussing the progress of specific tasks and the time that had been spent working on them[37].

Clarifications made up 8% of the meeting; these consisted of statements requesting more information as well as asking team members to elaborate on their previous statements[37]. Stray et al.[37] observed that, on average, each meeting had six clarifications and had anywhere between three to nine clarifications per meeting.

Moreover, 6% of the meeting was spent on what Stray et al.[37] describe as "meeting management"; these include decisions made by the Scrum Master such as asking team members to speak or to curtail their discussions.

Finally, digression constituted 5% of the daily scrum meetings[37]. Stray et al.[37] distinguish between three types of digressions: (1) discussions of side topics, (2) trouble with meeting equipment, and (3) statements that lightened the atmosphere during the meeting. Statements that lightened the atmosphere has the highest occurrence at 40%, followed by side topics at 32%, and equipment trouble made up 28% of digressions[37].

While Stray et al.[37] assumed that the daily scrum meetings would mainly be about answering "the three Scum questions", they observed that team members mostly focused on "problem-focused communication" and consisted of 46% more words than all other categories discussed above. Moreover, clarifications are the main reason that "problem-focused communication" was initiated[37]. Stray et al.[37] posit that clarifications seem to play an important role in establishing solutions to problems. Furthermore, the reason clarifications make up only 8% of the meeting is because the follow up questions were kept relatively short[37]. "Meeting management" is important in furthering communication structure at the daily Scrum meetings[37].

Daily Meeting Duration

The daily scrum for both development teams lasted on average 16.8 minutes, but Stray et al.[37] observed that in three out of the eight meetings, despite it being officially over, the meeting still continued. This was due to the Scrum Master reviewing information related to the project with the team or team members talking amongst themselves[37]. Stray et al.[37] noticed that as a result of team members often arriving late for the daily scrum meetings, they rarely started on time.

Scrum Master

Stray et al.[37] observed the Scrum Master taking over a somewhat similar role to that of a Project Manager by frequently making decisions such as what team members should speak or cutting them off when they were getting involved in longer discussions. Stray et al.[37] argue that by allowing the Scum Master to make such requests turned the Scrum Master into a de facto leader rather than a mediator of these meetings. By taking away this decision from the team, it contradicts the collective responsibility[2] and "self-organizing nature"[37] of an agile team.

Team Members

Team members seemed to be more interested in discussing obstacles that were encountered while working on a task rather than future events, since this would allow them to explain why they were unable to continue working on the plan from the day before[37]. Stray et al.[37] postulate that these daily meeting are too limited by time, uncertainty and incomplete information to allow for an in-depth discussion on finding solutions to a problem and making a decision accordingly.

Furthermore,Stray et al.[37] suggest that companies should focus on training their employees in areas such as collaborative teamwork and domain-specific expertise. Moreover, agile teams should consist of developers that are experts in their fields; if that is not the case, then it is up to the company to find a way for team members to become experts[37].

Even though the limited time in these daily meetings might prove challenging when it comes to decision making, the fact that a big part of the meeting is dedicated to "problem-focused communication" is integral to a team's success[37].

Documentation

One challenge a development team might have while attending daily meetings is the fact that decisions made during the meeting are not documented[37]. Stray et al.[37] did not observe any decisions being documented during the meetings by any of the development teams. Wrong decisions can lead to "major consequences" down the line and should therefore be "recorded and re-evaluated"[37].

Additionally, due to the lack of documentation, team members who missed the meeting ended up missing important information that might have been discussed throughout the meeting[37]. Team members across all development teams that were observed, did not feel the need to discuss documenting the meeting[37].

Furthermore, team members were also not updating each other on the decisions and results of the meeting[37]. Stray et al.[37] maintain that this "lack of documentation" stems from the fact that agile development advocates minimizing "unnecessary work" and "wasteful documentation".

Communication Pattern

Through analyzing the various transcripts, Stray et al.[37] recognized a pattern in the various interactions that occurred during meetings. The diagram below is the result of this observation.

The discussion usually began with the team member discussing the question of what they have done so far before moving on to obstacles that were faced trying to achieve that task[37]. Thereafter, team members asked for clarifications when an obstacle was described which tends to lead to a discussion based on the problem that was identified[37]. Upon the conclusion of the previous discussion, the Scrum Master urges the current team member to discuss their plans for that day, which often led to team members once again asking for further clarification and discussing possible solutions[37]. Finally, the Scum Master sums up the current team member's statement and asks the next team member to speak[37].

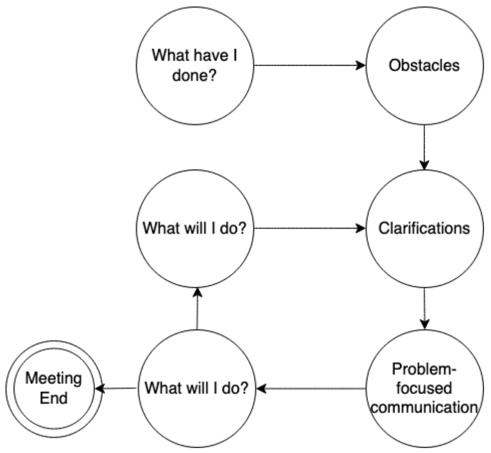

Figure 4.1: Sequence of Interactions [cf.] Stray et al.[37]

Communication Transitions

Moreover, Stray et al.[37] analyzed the transitions to "problem-focused communication" wherein another pattern seemed to emerge. The top three transitions are clarifications, obstacles, and tasks coordination[37]. Asking for clarifications most commonly led to a "problem-focused discussion"[37]. Following clarifications is obstacles, wherein team members immediately transitioned to discussing the problem and possible solutions[37]. Finally, by trying to coordinate tasks, team members realized that this in turn leads to new problems transitioning the discussion to"problem-focused communication"[37].

4.1.2 Extreme Programming

Case Study

In a case study conducted by Korkala et al.[25], they provided empirical results based on four different case studies that used Extreme Programming

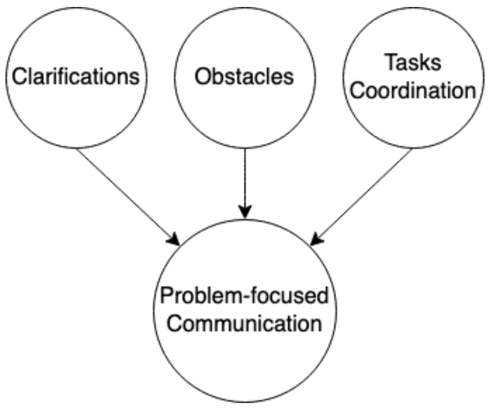

Figure 4.2: Transitions reproduced from Stray et al.[37]

as an agile development practice. Three of the case studies had limited access to an onsite customer while one case study had an onsite customer[25].

Customer Presence

Case 1 (C1) had a customer onsite 80% of the time who was available face-to-face for "mid-iteration" communication, Case 2 (C2) had the customer onsite during the beginning and end of each iteration who was also available for face-to-face and email during mid-iteration communication, Case 3 (C3) had the customer on site during the beginning and end of each iteration and was available through email during mid-iteration communication, and lastly for Case 4 (C4) the customer was only onsite for the first two weeks and was available through email and telephone during mid-iteration communication[25].

4.1.3 Client Communication

In the case of C2, the customer requested "simple reports" as they did not have the comprehension required for "technically oriented reports"[25]. Furthermore, customers from C3 did not feel the need to change the mid-iteration communication channel despite the increasing amounts of defects[25]. Korkala et al.[25] posit that one possible explanation for this request could be the inexperience the customer might have with agile development and argue that due to this inexperience, the customer might be unaware of the importance of communication and feedback in agile development.

Korkala et al.[25] observed that the simplicity of the reports that were sent to the customers did not enable the feedback necessary to further the development project. This observation was supported by analysing the e-mail correspondence between the development teams and the customers[25].

The development team from C2 and C3 never received feedback on the reports that were sent, and the development team from C4 received feedback once from the customer[25]. Furthermore, Korkala et al.[25] caution that due to the simplicity of the reports, the customer is unable to verify whether certain features have been implemented adequately since some of the content in these reports were ambiguous.

4.1.4 Defects

Since C4 only had the customer onsite for the first two weeks of the development project, the customer dependent defects remained high during the second, third, and final release[25]. Korkala et al.[25] maintain that due to the lean communication channels used after the second week of the project, it confirms the Media Richness Theory that lean channels are not well suited for handling ambiguous problems.

Additionally, C2 and C3 had the advantage of regular face-to-face communication and feedback due to the availability of on-site customers[25]. However, the development team in C3 experienced some communication issues due to the customer being unable to attend the meetings, resulting in them having to be postponed[25].

Moreover, Korkala et al.[25] noticed an "alarming trend" upon comparing C3 and C4. As a result of C3 and C4 having lower face-to-face communication opportunities and relying more on lean communication channels – which as stated above, are not well suited in cases of ambiguity – higher defect rates occurred[25].

Overall, the defect rates were increasing as the projects progressed, resulting in the development teams having to spend time fixing them rather than implementing new features or improving them[25]. Korkala et al.[25] argue that based on the research results, the increased dependence on lean communication channels rather than high-bandwidth communication channels, allows for higher defect rates to occur.

Additionally, on average, the 3 case studies (C2-C4) that had limited access to an onsite customer, 25.3% of all tasks were dedicated to fixing defects. Korkala et al.[25] also observed that 62.6% of software defects in C4 could have been avoided by either having access to an onsite customer or more efficient "mid-iteration" communication tools[25].

Moreover, Korkala et al.[25] observed that the time it took to fix customer dependent defects was ~ 1.6 times higher than customer independent defects. Customer independent defects in the developer category made up 37% of the time needed to fix defects while customer dependent defects in the requirements category made up 50% of the costs. Korkala et al.[25] argue that due to these results, special attention must be given to mid-iteration communication channels when onsite customers are limited or not available.

4.2 Communication Problems in Distributed Development Teams

Distributed development teams pose new challenges on software development projects due to the added complexity of having to accommodate for temporal distance, geographical distance, and socio-cultural distance[2, 40]. These increased complexities can affect formal and informal communication, delay coordination between team members, and decrease trust and the feeling of togetherness[40]. Due to these complexities, this section aims to investigate communication conflicts from the perspective of these three distances.

4.2.1 Temporal Distance

Temporal distance in distributed development teams reduces the opportunities for synchronous communication between team members, leading to delayed feedback[40] that could hinder the progress of the software project. Although the opportunity for synchronous communication is reduced, it allows for keeping a better record of communications[40] (i.e. documenting the communication that occurred through various written communication channels such as emails). Due to distributed development teams often spanning multiple time zones, team members might not always be available for feedback when it is needed[40].

Moreover, through the use of asynchronous communication tools (ex.: emails, instant messaging, documentation...) temporal distance increases the response time[40] compared to synchronous communication tools. Ågerfalk et al.[40](as cited by [5]) discuss the fact that a team member that just started the work day might become overwhelmed by the number of questions and requests for feedback received overnight through these communication tools. Furthermore, the use of e-mail as a communication tool increases the chances of misunderstandings[40](as cited in [15]) especially if the team member perceives it as aggressive or antagonizing[40] (as cited in [24]). Similarly to Ågerfalk et al.[40], while analyzing the interview transcripts for this thesis, the topic of misunderstandings through the use of emails came up. Due to the perceived context of importance, a communication issue occurred. Where one team member might deem the email as something to take note of, another team member might perceive it as highly important and needing to be taken care of immediately.

Moreover, due to the delayed response time through the use of asynchronous communication tools, it could take longer to resolve problems in a timely manner[40](as cited in [5]), which in turn can worsen the problem if not handled properly[40](as cited in [24]) and lead to delays in the completion of the software project on time.

4.2.2 Geographical Distance

Geographical distance increases the cost of holding face-to-face meetings[40], which as explored in Section 4.1.2 and reiterated by Misra et al.[29] as well as Ågerfalk et al.[40](as cited in [9]), is the main form of communication that should take place during software development. While geographical distance uses multiple communication channels to keep track of the progress of the software project, it is still difficult to adequately communicate visions and strategies that are important to the software project[40].

Informal Communication

One of the biggest issues affecting distributed development teams due to geographical distance, is the absence of informal communication[40]. Informal communication is important for developing and nurturing relationships between members of the development team and has the added advantage of improving information flow regarding changes and progress made in the software project[40](as cited in [20]). Ågerfalk et al.[40] argue that informal communication is an important aspect of distributed development teams, and caution that documentation is often not enough to resolve misunderstandings that stem from it[40](as cited in[15, 12]. Contradicting Ågerfalk et al.[40], Casey et al.[8] (as cited in [18]) suggest the need for formal methods of communication as a result of the lack of informal communication. Whether a distributed development team chooses to implement formal methods of communication or not, they should remain aware of the fact that there is a lack of informal communication and use the communication tools at their disposal to make up for it.

While geographical distance diminishes the amount of informal communication that takes place between distributed team members, in development teams that work at the same location, an estimated 75 minutes is spent on informal communication daily[40](as cited in [20]). These informal communications are activities team members partake in that are outside of the scope of the software project such as coffee breaks, "office talks", and lunch breaks. Nevertheless, even for team members working at the same location, just a 30 meters distance can have a tremendous effect on the amount of communication that occurs between team members[40](as cited in [1]). This can be due to the time it takes to cross the distance to get to a team member if they were working in another part of the office or building.

Additionally, geographical distance increases the amount of effort needed to initiate contact[40](as cited in [19]. Due to this lack of communication, team members might take it upon themselves to make modifications to the software project instead of trying to contact the team member responsible for this part of the project[40](as cited in [5]). Ågerfalk et al.[40] posit that this might be due to the lack of informal contact with the remote team member and the lack of knowledge of their skills and roles in the software project. In order to mitigate this communication issue, project managers need to communicate with all team members from all locations to ensure that they know who is responsible and can be contacted for certain project tasks[8].

4.2.3 Trust

Similar to the lack of informal communication, it is difficult to foster trust in distributed development teams. Wherein co-located development teams have the chance to build trust based on their shared beliefs and experiences, distributed development teams are not afforded the same opportunities[14]. According to Daim et al.[14](as cited in [22]), social interactions and enthusiasm increase trust in distributed teams. In addition to social interactions and enthusiasm, individual team members can foster trust within their team by learning how to cope with uncertainties and taking initiative[14](as cited in [22]). Consistent communication and responding in a timely manner also foster trust within distributed teams[14](as cited in [22]).

Daim et al.[14] maintain that project managers play an important role in building trust amongst team members and should emphasize their roles and importance of being part of the team. Additionally, project managers should encourage team members to partake in activities outside of work to build trust within the team[14].

Technological Dependence

The high level of dependence in distributed teams on technology is another side effect of geographical distance[40]. This reliance on technology can have a negative effect on team collaboration, productivity, and innovation[14]. For example, due to the reliance on third-party tools used to communicate with team members[40], the release of new versions of a third-party tool in one country and not the other[40](as cited in [3] could have a negative impact on communication and feedback. This might also have an adverse effect on the software project by increasing the time needed to release it. Furthermore, similar to third-party tools that are dependent on software releases, video conference experiences are dependent on hardware capabilities such as PC network abilities, camera resolution, and headset quality[14]. This is an interesting observation, as organizations might not always provide their employees the necessary equipment needed for an adequate video conference experience.

Moreover, another critical factor that relies on technology and can lead to communication problems is the availability of reliable infrastructure[8]. Reliable infrastructure includes a constant supply of electricity, a reliable internet connection and adequate bandwidth[8]. In their research, Casey at al.[8] observed severe communication issues that occurred due to the inadequacy of a telecommunication system that affected the development teams training and knowledge transfer. Moreover, Casey et al.[8] argue that having sufficient communication tools available is essential in maintaining a development teams motivation, and that these tools are important in assuring successful collaboration within a distributed development team. Similar to Casey et al.[8], in their interview study, Daim et al.[14] report that during video conferences, packet loss, flickering video, and delays in transmission were deemed as unacceptable by participants. Daim et al.[14] maintain that consistent, low-latency performance is needed for seamless real-time communication within distributed development teams.

4.2.4 Socio-cultural Distance

Language Barriers

Socio-cultural distance in distributed development teams runs the risk of misunderstandings, especially if there is a high social-cultural distance between team members[40]. A huge communication barrier that distributed development teams often face, is the fact that some or even all team members are non-native English speakers[40]. The use of synchronous communication tools could overwhelm these team members as they struggle to keep up with the conversation[40](as cited in [24, 33]). This might be due to the fact that the conversation could be moving too fast or some unfamiliar words such as the use of slang are being used. Ågerfalk et al.[40] discuss the fact that non-native speaking team members tend to prefer asynchronous communication tools, as it allows them to revise their statements and make sure that they are getting their points across clearly. Another advantage of using asynchronous communication for non-native speakers is the ability to use translation software to facilitate their understanding and use of certain words or phrases. Asynchronous communication tools also allow for grammar and spelling mistakes correction which are often integrated into the communication tool being used.

Furthermore, language competency does not only affect non-native English speakers, even amongst native English speaking team members misunderstandings can arise[40]. This is due to the fact that team members have a problem understanding each other because of the different dialects and accents that exist, even within the English language[40](as cited in [7]. For example, an American team member of a distributed development team might have trouble understanding a Scottish team member from "across the pond". Due to these language barriers, Ågerfalk et al.[40](as cited in [21]) argue that this language barrier is the reason why it remains as one of the biggest communication problems in software development teams. Ågerfalk et al.[40](as cited in [33]) caution that native speaking team members should be careful not to alienate their non-native speaking team members as they are at a disadvantage in being able to express themselves properly, and should therefore develop a level of mutual understanding in order to be able to communicate successfully[40](as cited in [11]). This point is reiterated by Bjørn et al.[4] in which they argue that team members of a distributed development team have limited options to develop a shared meaning within the team due to the fact that they rarely have face-to-face communication that allow for finding common ground and thereby increasing the risk of a communication breakdown.

Casey et al.[8] argue that due to this inability for distributed development teams to communicate face-to-face, project managers need to find a way to develop team relationships. Project managers need to take socio-cultural communication problems in a distributed development team into account by monitoring and addressing them when they occur[8]. Daim et al.[14] suggest that project managers need to clearly communicate what is expected of all team members in order to achieve project success regardless of their locations, since there is always the risk of misunderstandings occurring especially in distributed development teams.

Conflict Management Styles

A less apparent socio-cultural difference within distributed software development teams is the approach to conflict management. While some team members may have a more open communication style by stating their opinions when they're having an issue with one of their team members, remote team members, due to their differing cultural backgrounds, will shy away from it[8]. Casey et al.[8] describe this as a cultural aversion to conflict, referring to the fact that remote team members might be reluctant to express their feelings and openly disagreeing with their team members, even when unreasonable requests were being made and inappropriate behaviour was taking place. The remote team members preferred to ignore it rather than confront their colleagues about it[8].

Furthermore, Casey et al.[8] observed the fact that there seemed to be no formal procedure in place for deescalation, questioning the effectiveness of the informal procedure. It seemed to be up to the discretion of the project manager on how to approach the conflict, referring to it as an "informal[...]escalation procedure" that is "specific to the project"[8]. With regards to the co-located members of the distributed development team, there seemed to be a similar informal approach to conflict management[8]. Casey et al.[8] argue that this informal approach to conflict management for co-located team members may be effective, as the team members have the chance to communicate face-to-face to resolve their issues, but due to the lack of informal communication within distributed development teams, the procedure might not be as effective to resolve issues that arise within remote team members.

Moreover, Daim et al.[14] argue that by allowing team members the space to have open communication, without the fear of repercussion from their peers when they have differing viewpoints, it could encourage team members to reflect on the limitations of their own views. This might inspire team members to seek out more information and knowledge in better understanding this differing viewpoint to incorporate it in future decisions made by the team[14].

Chapter 5

Discussion

RQ1: What communication conflicts can occur within communication channels?

Scrum Regarding Scrum, various incidents where possible communication conflicts could occur stood out. By inadvertently taking over the role of project manager, the scrum master runs the risk of being resented by their team members. Determining who gets to speak, or cutting off team members and not allowing them to continue the discussion could lead to team members feeling alienated. This in turn could lead to communication conflicts as team members might be too intimidated to speak up if they're having a problem finishing a task.

In the long run, having people often arriving late to meetings might lead to conflicts down the line. Individual team members might start harbouring resentment towards them for not feeling like their time is being respected. Due to the possibility of having a development team with a culturally diverse background, attitudes towards time and time management may differ.

The team members that are arriving late to meetings might also give off the impression that they are not fully invested in the project like their peers. Unless it was previously decided that the meeting will begin once everyone arrives, steps should be taken to ensure that everyone is satisfied with the decision; if that is not the case, then those persons who might be delayed in attending the meeting should consider not going and reviewing meeting summaries or notes from peers. This might prove problematic if documentation is inadequate.

The daily meetings themselves could pose a threat to communication if there wasn't enough time to discuss problems team members were having and taking adequate steps to resolve them. Due to the lack of documentation and communication within the team to keep each other informed, team members might not be know about changes or decisions that were made during the meetings, which could lead to communication problems if for example team members were assigned tasks without knowing it.

Extreme Programming Due to the refusal of clients to transition to other communication channels throughout the project despite the increasing amounts of defects, communication conflicts occurred due to the lack of feedback needed from the clients. Additionally, while clients might not feel the need to receive technical reports, it poses a challenge for developers if the client does not respond or give any feedback on the simplified reports that they were sent. This could lead to frustration and communication conflicts if either party feels ignored.

Furthermore, communication conflicts occured due to some clients only being available through email or phone communication, rather than face-to-face communication (the preferred method for XP). This resulted in an increased rate of customer dependent defects since the use of communication tools such as email and phone does not allow for immediate feedback like face-to-face communication.

RQ2: What communication conflicts can occur within distributed development teams?

Due to the temporal distance not allowing for synchronous communication, the delayed feedback and an increase in response time could lead to communication conflicts, especially during times where immediate feedback is necessary (ex.: task problem). Additionally, the use of asynchronous communication tools such as e-mail increases the chances of miscommunication and misunderstandings, especially if team members perceive it as antagonising or aggressive.

Moreover, the lack of informal communication and building trust is one of the biggest challenges faced within distributed development teams. While informal communication is important for developing relationships, it has the added advantage of improving information flow between team members.

Furthermore, geographical distance also increases the effort needed to initiate contact, which can lead to problems if team members take it upon themselves to for example, make changes in the software project without contacting the person responsible for that task.

Due to the increased reliance on technology in distributed teams, inadequate software and hardware infrastructure could lead to breakdown in communication, especially if there's a reliance on third-party software tools or remote team members are located in a region that does not provide adequate network support.

Language barriers within distributed teams pose a huge risk for miscommunication and misunderstanding to occur, due in part to the increased reliance of non-native speakers on asynchronous communication as the use of synchronous communication tools might be overwhelming.

Finally, an underlying factor that poses a challenge for distributed teams is the difference in conflict management styles. Where some team members are more open in their communication and attitude, team members from a different cultural background might shy away from confrontation or even outright ignore conflict.

Chapter 6

Conclusion and Future Works

This chapter summarizes the results of the thesis and possible improvements that could take place in the future.

6.1 Conclusion

This thesis aimed to investigate communication problems based on interview transcripts provided by David Sasse[34]. Communication conflicts in the context of software development can be hard to detect, as many factors can play into it. Findings from preliminary analysis of the interview transcripts indicated that work environments, personalities, communication tools, and project management amongst other factors, can all play a role in communication breakdown. In order to narrow down the scope of communication conflicts, a qualitative data analysis was applied to the interview transcripts.

The first step was to analyze the interview transcripts and determine relevant passages related to communication conflicts. After determining the relevant passages, a qualitative data analysis using the grounded theory method was applied. Through the application of initial coding, focused coding, and memos, three categories that relate to communication conflicts were determined. The three categories were: communication channels, teams, and conflict management.

Having narrowed down the scope of communication conflicts and forming research questions based on these categories, the next step was to find relevant literature. This was conducted through the use of a database search, taking into account the inclusion and exclusion criteria that was determined based on the findings of the data analysis phase.

The final step, the core of this thesis, was to investigate communication conflicts that occur within communication channels and distributed development teams. The communication channels chosen for this thesis to investigate communication conflicts were the agile development practices

Scrum and Extreme Programming. With regards to distributed development teams, communication conflicts were investigated within the context of temporal, geographical, and socio-cultural distances.

6.2 Future Works

Improving the Investigation of Communication Channels

This section could have been better structured and defined. A variability in literature choices like the section on communication conflicts in distributed teams might have been better suited as opposed to the decision to focus on case studies. Given the ambiguity of communication conflicts explored in this section, it was difficult to define.

Exploring Personality Traits

Due to the scope of this thesis, it was not possible to delve deep into the connection between personalities and communication conflicts in the context of software development teams. This is a topic of personal interest, as it involves psychological and social aspects alongside software engineering. It would be interesting to develop an application that could determine who would work best together (i.e. team constellation) through analyzing survey answers for example.

Appendix A

Appendix

A.1 Keyword Mind Map

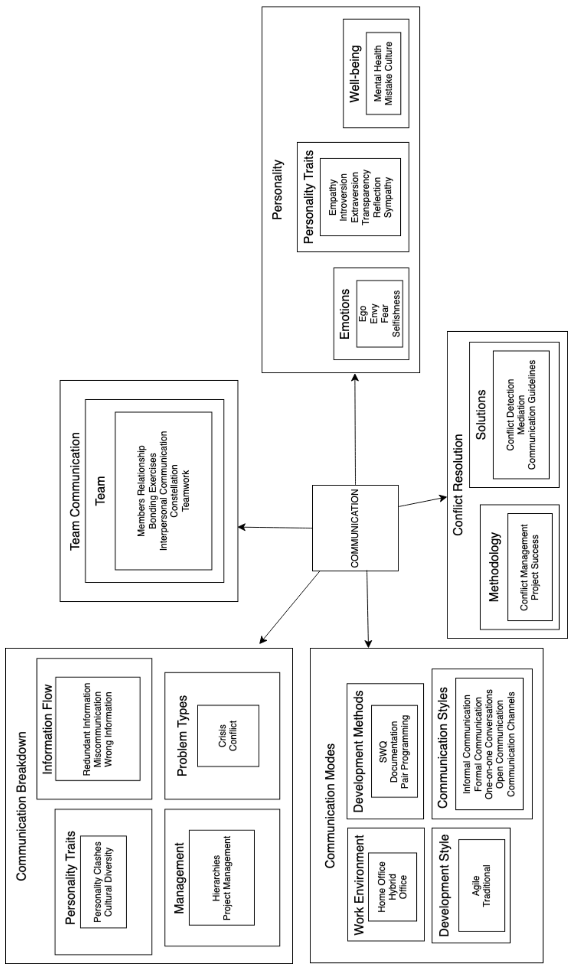

Figure A.1: Snippet of the Keyword Mind Map

Bibliography

[1] Thomas J. Allen. *Managing the Flow of Technology.* MIT Press, 1977.

[2] VenuGopal Balijepally, RadhaKanta Mahapatra, and Sridhar P. Nerur. Assessing Personality Profiles of Software Developers in Agile Development Teams. *Communications of the Association for Information Systems*, 18, 2006.

[3] R.D. Battin, R. Crocker, J. Kreidler, and K. Subramanian. Leveraging resources in global software development. *IEEE Software*, 18(2):70–77, 2001.

[4] Pernille Bjørn and Ojelanki Ngwenyama. Virtual team collaboration: building shared meaning, resolving breakdowns and creating translucence. *Information Systems Journal*, 19(3):227–253, May 2009.

[5] D. Boland and B. Fitzgerald. Transitioning from a co-located to a globally- distributed software development team: A case study at analog devices inc, 3rd. international workshop on global software development, edinburgh. 2004.

[6] Kenneth E. Boulding. *Conflict and defense: A general theory.* Harper & Brothers, United States of America, 1962.

[7] Valentine Casey and Ita Richardson. Practical experience of virtual team software development. 11 2004.

[8] Valentine Casey and Ita Richardson. Project Management within Virtual Software Teams. In *2006 IEEE International Conference on Global Software Engineering (ICGSE'06)*, pages 33–42, Florianopolis, Brazil, October 2006. IEEE.

[9] Herbert H. Clark. *Using Language.* Cambridge University Press, Cambridge, 1996.

[10] N. J. Clifford, Shaun French, and Gill Valentine, editors. *Key methods in geography.* Sage Publications, Thousand Oaks, CA, 2nd ed edition, 2010.

[11] Catherine Durnell Cramton. The mutual knowledge problem and its consequences for dispersed collaboration. *Organization Science*, 12:346–371, 2001.

[12] Bill Curtis, Herb Krasner, and Neil Iscoe. A field study of the software design process for large systems. *Commun. ACM*, 31(11):1268–1287, nov 1988.

[13] Richard L Daft and Robert H Lengel. ORGANIZATIONAL INFORMATION REQUIREMENTS, MEDIA RICHNESS AND STRUCTUR.

[14] Tugrul U. Daim, Anita Ha, Shawn Reutiman, Brennan Hughes, Ujjal Pathak, Wayne Bynum, and Ashok Bhatla. Exploring the communication breakdown in global virtual teams. *International Journal of Project Management*, 30(2):199–212, February 2012.

[15] D.E. Damian and D. Zowghi. The impact of stakeholders' geographical distribution on managing requirements in a multi-site organization. In *Proceedings IEEE Joint International Conference on Requirements Engineering*, pages 319–328, 2002.

[16] Martin Fowler and Jim Highsmith. Facilitating change is more effective than attempting to prevent it. Learn to trust in your ability to respond to unpredictable events; it's more important than trusting in your ability to plan for disaster.

[17] Tim Goles and Wynne W Chin. Information systems outsourcing relationship factors. 36(4), 2005.

[18] J.D. Herbsleb and R.E. Grinter. Architectures, coordination, and distance: Conway's law and beyond. *IEEE Software*, 16(5):63–70, 1999.

[19] J.D. Herbsleb and R.E. Grinter. Splitting the organization and integrating the code: Conway's law revisited. In *Proceedings of the 1999 International Conference on Software Engineering (IEEE Cat. No.99CB37002)*, pages 85–95, 1999.

[20] J.D. Herbsleb and A. Mockus. An empirical study of speed and communication in globally distributed software development. *IEEE Transactions on Software Engineering*, 29(6):481–494, June 2003.

[21] Vegar Imsland, Sundeep Sahay, and Yvonne Watianen. Key issues in managing a global software outsourcing relationship between a norwegian and russian firm: Some practical implications. *26th Information Systems Research Seminar in Scandinavia, Finland*, 2003.

[22] Sirkka Jarvenpaa and Dorothy Leidner. Communication and trust in global virtual teams. *J. Computer-Mediated Communication*, 3, 06 1998.

[23] Karen A. Jehn. A Multimethod Examination of the Benefits and Detriments of Intragroup Conflict. *Administrative Science Quarterly*, 40(2):256, June 1995.

[24] Lori Kiel. Experiences in distributed development: A case study. 2003.

[25] M. Korkala, P. Abrahamsson, and P. Kyllonen. A Case Study on the Impact of Customer Communication on Defects in Agile Software Development. In *AGILE 2006 (AGILE'06)*, pages 76–88, Minneapolis, MN, USA, 2006. IEEE.

[26] Robert E. Kraut and Lynn A. Streeter. Coordination in Software Development. *COMMUNICATIONS OF THE ACM*, 38(3):13, March 1995.

[27] Seyed-Ali Marjaie and Urvashi Rathod. Communication in Agile Software Projects: Qualitative Analysis using Grounded Theory in System Dynamics.

[28] G. Melnik and F. Maurer. Direct Verbal Communication as a Catalyst of Agile Knowledge Sharing. In *Agile Development Conference*, pages 21–31, Salt Lake City, UT, USA, 2004. IEEE.

[29] Subhas Misra, Vinod Kumar, Uma Kumar, Kamel Fantazy, and Mahmud Akhter. Agile software development practices: evolution, principles, and criticisms. *International Journal of Quality & Reliability Management*, 29(9):972–980, October 2012.

[30] Lisa Hope Pelled. Demographic diversity, conflict, and work group outcomes: An intervening process theory. *Organization Science*, 7(6):615–631, 1996.

[31] M. Pikkarainen, J. Haikara, O. Salo, P. Abrahamsson, and J. Still. The impact of agile practices on communication in software development. *Empirical Software Engineering*, 13(3):303–337, June 2008.

[32] Olivia N. Saracho, editor. *Handbook of research methods in early childhood education*. Contemporary perspectives in early childhood education. Information Age Publishing, Inc, Charlotte, NC, 2015.

[33] S. Sarker and S. Sahay. Information systems development by us-norwegian virtual teams: implications of time and space. In *Proceedings of the 35th Annual Hawaii International Conference on System Sciences*, pages 10 pp.–, 2002.

[34] David Sasse. Interviewstudie zu sozialen aspekten in modernen softwareprojekten, leibniz universität hannover, 2023.

[35] Jonathan A. Smith, Rom Harré, and Luk van Langenhove, editors. *Rethinking methods in psychology.* Sage Publications, London ; Thousand Oaks, Calif, 1995.

[36] Dirk Stelzer and Werner Mellis. Success factors of organizational change in software process improvement. *Software Process: Improvement and Practice*, 4(4):227–250, December 1998.

[37] Viktoria Gulliksen Stray, Nils Brede Moe, and Aybuke Aurum. Investigating Daily Team Meetings in Agile Software Projects. In *2012 38th Euromicro Conference on Software Engineering and Advanced Applications*, pages 274–281, Cesme, Izmir, Turkey, September 2012. IEEE.

[38] Kenneth J Trimmer, Madeline A Domino, and J Ellis Blanton. The Impact of Personality Diversity on Conflict in ISD Teams. *Journal of Computer Information Systems*, 2002.

[39] L. Wu and H. Sahraoui. Accommodating software development collaboration. In *12th Asia-Pacific Software Engineering Conference (APSEC'05)*, page 8 pp., Taipei, Taiwan, 2005. IEEE.

[40] Pär J Ågerfalk, Brian Fitzgerald, Helena Holmström, Brian Lings, Björn Lundell, and Eoin Ó Conchúir. A FRAMEWORK FOR CONSIDERING OPPORTUNITIES AND THREATS IN DISTRIBUTED SOFTWARE DEVELOPMENT.