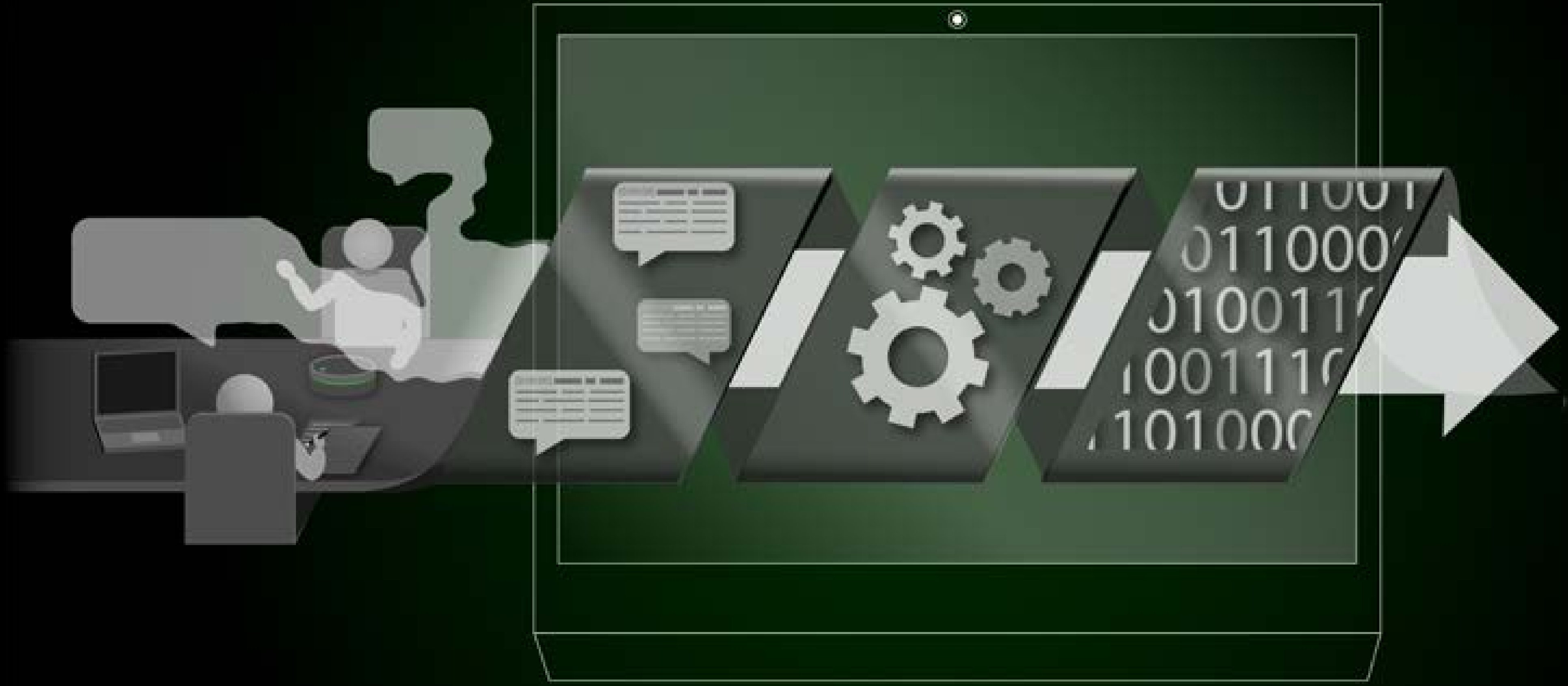
Designing a System to Aid in Conversational Requirement Elicitation Practices

Integrating REConSum and Trace2Conv into a system that can be used and experienced in practice.

Master Thesis Industrial Design Engineering
Jeroen van kekem
24-05-2024
DPM 2101



INFORMATION

Designing a System to Aid in Conversational Requirement Elicitation Practices

Master Thesis - Undustrial Design Engineering
University of Twente - Fizor.
May 2024
Jeroen van Kekem - s1543628

Education
Faculty: Favulty Engineering Technology
Department: Design, Production and Management
Master Programme: Industrial Design Engineering
Track: Management of Product Development

Educational institution
University of Twente
Drienerlolaan 5
7500 AE Enschede

Company
Fizor.
Herculesplein 2
3584 AA Utrecht
Netherlands

Report
Number of Pages: ...
Number of Appendices: ...

PREFACE

This thesis journey has been a challenging yet rewarding experience, particularly navigating the substantial gap between industrial design engineering and IT. While there is overlap in design and production methodology between the two areas of discipline, there were also encountered multiple differences where there are similarities and differences in the language used, different principles, and different visions regarding how products are designed, and requirements are formulated. I learned a lot during my thesis that will prove useful in my future career.

This thesis journey has been marked by the exploration of an interdisciplinary realm where I grappled with the intricacies of harmonizing the methodologies of software design with the foundational principles of industrial design engineering. In this journey, looking to the design problem as a system allowed me to understand the initial systems on a functional level and allowed me to start designing at a point where the underlying mechanics were not yet fully understood and the skills necessary to realize any functionality with them were learned.

I want to extend my sincere gratitude to Tjerk Spijkman (company supervisor) and Roy Damgrave (UT supervisor) for their continuous support throughout my thesis. My meetings with them allowed me to get a better understanding of things, gave me new useful insights, and allowed me to continue when I was stuck for a moment. Both have made a lot of time when I needed something, which I appreciated a lot.
I want to thank Tjerk especially for sharing his expertise regarding the research topic and related business practices what allowed me a lot in the design process. Furthermore, his contribution to testing and evaluating design ideas helped a lot in the development of the realized system. I want to thank Roy especially for his expertise in the Industrial Design process and helping me through obstacles in the design process. I really enjoyed the interesting discussions we had each meeting about different findings or design ideas and helped me to get new and interesting insights.

My journey at Fizor was nothing but great. The time at the company made me feel as I was part of the team, I got the opportunity to participate in the many learning opportunities like workshops and got the space to really get to know the people working there and the company culture. I really appreciate the continued support and resources that were made available to me. It is a great company with a company culture that resonates a lot with me, and I look very much forward to start working there now my thesis has been concluded. As I conclude this chapter of my academic journey, I eagerly anticipate the prospect of transitioning into a professional role at Fizor. I am enthusiastic about contributing to its ongoing success.

SUMMARY

Conversational RE is an important part of the requirement engineering tasks performed within the project life cycle of an agile software development team. Efforts to automate of Requirement Engineering processes using Natural Language Processing can be widely found in literaturre and Fizor, the client also has produced two functionalities/systems/tools that make use of NLP with the aim to improve processes within the Requirement Engineering workflow, specifically automatically processing conversations to enable different functionalities for users. These are called REConSum and Trace2Conv.

The initial problem statement has been translated to a desired system consisting of multiple system layers that build on elicitation interviews used to elicit requirements and ends in a use case to use this (processed) data. A knowledge base has been created with a base in literature and the input of 8 Buiness Analysts that provided valuable insights in annekdotes, general workflow, and challenges. The initial system has been analysed and with this initial system a system prototype has been realized that integrates the subsystems of REConSum to aids Business Analysts in Conversational Requirement Elicitation activities.

This thesis has produced a prototype that has the potential to be useful for Business Analysts in the activity of processing captured elicitation data into structured requirements. While it may not always be of use, in longer real-life conversations the ability to filter conversations to search for specific conversation data has the potential to be valuable for a Business Analyst (Spijkman et al., 2023).

The main functionality on system level that has been tried to realize is to capture all conversation data generated during the elicitation interview (so no conversation data is lost) and allowing a Business Analyst to locate requirement relevant information in this conversation dataset using filters. The prototype that has been realized is a system that consists of multiple subsystems labeled ‘MS Teams, NLP, GUI, and Marker’, each with their own behaviour and components, that together achieve the main functionality of the system.

The components of REConSum have been used as start of the system design and have been integrated into one system prototype that can be used through a GUI. This removed the need for installing python and the desired packages, writing code that runs the algorithms of REConSum, and manually downloading and installing of StanfordCoreNLP (a system necessary for functioning, requiring the downloading and installing of java) and running it manually. This solution was necessary as the knowledge and time needed to run REConSum would ask to much of any user. A solution has been provided for users to interact with the outputs that are being generated by the system through the design of the system GUI, storing the system outputs in excel files, and realizing the systematically accessing and storing of information.To create a seamless transition between the MS Teams Conversation Artefact output and the expected input an import function has been integrated that performs all necessary steps to format, process and view the conversation in the background. This in total takes 5-10 minutes in total from start to end, and takes 15 minutes of processing time after which the conversation can be directly used by the Business Analyst.

Within the scope of this thesis only the systems functionality could be realized and validated. The system will most definetley need to see improvements to fit more specific user requirements for example. With how the system is designed future research questions can be tailored to test the effectiveness, efficiency, and user friendlyness in regards to capturing, processing, and using requirement relevant information from conversation.

Within the scope of this thesis conversation artefact with enough accuracy could be realized. Furthermore, in face to face conversations, speakerturns are formulated as required because of how MS Teams functions. Solutions with high chance to solving these problems have been provided in a implementation plan that provides guidelines for further research and development and how to implement the prototype within the workflow of Fizor as a first step. This implementation provides initial requirements of implementing and validating the method to accurately capture requirement artefacts. It also provides for each part of the system suggestions for further research and development.

This prototype is not a finished product but should be considered as a tool designed for Fizor to get useful feedback from users to be used in further research and development while minimally disrupting the current workflow of Business Analysts.

GLOSSARY

Algorithm:
An algorithm is a defined set of instructions or rules that, when followed, allows a computer to solve a specific problem or perform a particular task efficiently.
Business Analyst:
A Business Analyst is a professional who analyzes organizational processes, identifies needs and requirements, and recommends solutions to help businesses improve efficiency, productivity, and profitability.
Conversational Requirement Elicitation:
Conversational Requirement Elicitation is the process of gathering and understanding the needs, preferences, and expectations of stakeholders through dialogue and interaction, to inform the design and development of conversational systems or interfaces.
Conversation Artefact:
A Conversation Artefact refers to any tangible or recorded element produced during a conversation, such as transcripts, recordings, or annotations, which can be analyzed or referenced for insights into the communication dynamics, content, or outcomes of the interaction.
Natural Language:
Natural language refers to the way humans communicate with each other using spoken or written words, expressions, and symbols that have evolved naturally within a community or culture, as opposed to formalized or artificial languages.
REConSum:
A NLP prototype tool that can assist practitioners and researchers in processing elicitation conversations by summarizing the transcriptions and extracting requirements-relevant information
Requirement Artefact:
A documented or tangible representation of a requirement, which could include specifications, user stories, use cases, diagrams, or any other artifact used to capture, communicate, or manage requirements during the software development lifecycle.
Speakerturn:
A Speakerturn is a term used in conversational analysis to denote a segment of speech or dialogue produced by a single speaker within a conversation
Trace2Conv:
An initial effort to establish backward traceability from requirements to relevant transcript segments in requirement conversations.

ABBRIVIATIONS

BA (Business Analyst)
CA (Conversation Artefact)
GUI (Graphical User Interface)
MLP (Manual Language Processing)
MS Teams (Microsoft Teams)
NLP (Natural Language Processing)

0. TABLE OF CONTENTS

1. INTRODUCTION

This thesis delves into a system prototype developed in response to the goals and needs articulated by Fizor, a company central to this study. This chapter will begin by introducing the concept of Conversational Requirement Elicitation (Conversational RE), which forms the core focus of this thesis.

To understand the system design and its potential value to Fizor and its key user(s), Conversational RE's position within the requirement engineering process will be introduced as well as the value of Conversational RE itself as activity. The key user group 'Business Analysts', that perform this activity, will be discussed. As well as the best practices related to Conversational RE, challenges related to conversational RE, and requirements related to Conversational RE.

Additionally, this thesis outlines the primary stakeholder groups involved in Conversational RE and identifies the key user of the system design. Moreover, an introduction to Fizor will be provided, elaborating on their relationship to and interest in Conversational RE, as well as their objectives and aspirations, for which a design process has been implemented to craft a suitable solution. This thesis concludes the graduation program of the study in Industrial Design Engineering at the University of Twente where one or multiple potential solutions are designed in order to solve a specific industry problem.

1.1) Conversational RE

Conversational RE is an important part of the requirement engineering tasks performed within the project life cycle of an agile software development team. Agile software development, characterized by its iterative approach and rapid development cycles known as sprints, emphasizes efficiency and adaptability. Through phases encompassing planning, design, development, testing, deployment, and review, agile teams rapidly iterate on system components within short time frames typically spanning between a couple of week to a few months (Al-Saqqa et al., 2020). In the project life cycle Requirement Elicitation (RE) concerns the activities of seeking, uncovering, acquiring, and elaborating requirements (Spijkman et al., 2023). RE stands as a cornerstone, pivotal for effectively capturing and communicating client needs early in the development process. This early comprehension, coupled with the elicitation of evolving requirements, serves as a critical strategy for mitigating the substantial costs associated with error correction later in system development (A. Davis et al., 2006). Recognized as an integral part of every development cycle, RE involves an interactive process between analysts and clients aimed at identifying and capturing the essential requirements of a system or product to be developed (Bano et al., 2019; Davey & Cope, 2009; C. J. Davis et al., 2006; Ferrari et al., 2016; Sutcliffe & Sawyer, 2013). This process typically unfolds in the early stages of developing information (Davey & Cope, 2008; C. J. Davis et al., 2006) and is universally regarded as one of the most pivotal steps in system development (Davey & Cope, 2008). In Conversational RE conversations are used as the method to elicit requirements from clients and stakeholders. This is one method among others available like observations, unstructured interviews, structured interviews, protocol analysis, and others(Davey & Cope, 2008). Among these methods conversations (or semi structured interviews) stands out as a widely used and highly effective approach, particularly through semi-structured interviews, renowned for their knowledge transfer efficacy (Bano et al., 2019; Davey & Cope, 2008).

1.2) Requirement Engineering Workflow

While RE is an integral part of any development cycle only the agile software development cycle will be considered as the scope of this thesis. To better illustrates its role within the agile software development, cycle the requirement engineering tasks, of which RE is one, will first be covered. Within the agile software development team the Business Analyst is responsible for requirement engineering tasks that take place during the project lifecycle. These activities can be grouped into multiple requirement engineering phases. These phases can be defined as 'needs assessment, Requirement management planning, Requirement elicitation, Requirement analysis, Requirement monitoring and controlling, Solution evaluation, and project closure'. Each phase will be covered briefly while elaborating more on (Conversational) Requirement Elicitation and requirement analysis as they are the focus of this thesis.

Needs assessment is conducted prior to the project life cycle, aiming to analyze requirements and identify business problems or opportunities. The findings form the foundation for subsequent requirements processes.

Planning requirements management occurs within the Planning Process Group, providing guidance for developing and managing requirements throughout the project lifecycle. It involves crafting, assessing, and revising plans to align with various lifecycle phases.

Elicitation is a discovery process that extracts insights from stakeholders and various sources to understand business problems or opportunities. It operates iteratively throughout the project lifecycle, contributing to the definition of product backlog and its refinement.
Analysis involves scrutinizing a
nd integrating information to derive comprehensive requirements. It follows a progressive and iterative approach, persisting until the necessary level of requirements is attained. Both elicitation and analysis are ongoing in an adaptive life cycle.

Requirements monitoring and controlling oversee the status of requirements and manage the requirements baseline throughout the project's life cycle. It ensures proper approval and management of changes to requirements, distinguishing it from project management processes.

Solution evaluation validates a solution's effectiveness in meeting stakeholder needs and delivering value. It may identify new requirements for solution refinement. Different organizations may use various terms and methods for these processes, but the goal remains consistent across projects.

Conversational RE, best practices

Bano et al. (2019) presents a novel pedagogical approach for training student analysts in the art of elicitation interviews. 7 high level category mistakes are presented, "...namely question formulation, question omission, interview order, communication skills, analyst behaviour, customer interaction, teamwork and planning" (Bano et al., 2019) . Throughout the paper, best practices of performing elicitation interviews are described.

For a well-planned interview, take time as analyst to prepare in advance by writing down clear and unambiguous questions. The analyst needs to familiarize themself with the problem domain for an effective elicitation interview It is necessary for the analysts to form a questioning strategy and include prompts based on the context of their interview. This can help in eliciting requirements, as well as overcoming the challenge of client-analyst interaction. A good questioning strategy consists of a start, in which the analyst builds rapport with the customer. A mid-section in which he understands the existing business process and the problems faced by the client in current process in order to reason on the need for a new system. Towards the end of the interview, the analyst needs to summarize the findings to the customer to confirm the understanding. Summarizing the findings of the interview is a best practice for overcoming the misinterpretations during the interview and overcoming any cognitive limitations during customer and analyst communication. Effort must be done by the analyst to remove the semantic gap and push the customer to the boundaries of their tacit knowledge. The use of common vocabulary during interview is also very important, and the analysts should plan and prepare so that they will not use the words that might confuse the customer. It is typically the responsibility of the analyst to create a friendly environment that can stimulate the communication with the customer.

Conversational RE and Requirement Analysis |

Stakeholders

There are three stakeholder groups to consider in regards to Conversational RE being Business Analysts, Client(s)/Stakeholder(s), and the Development team. The Requirement Artefact is the main item of interest for both the development team and the client as it dictates what will be delivered in the end of the project lifecycle. Both the client and development team prioritize accuracy, completeness, and clarity in documented requirements to ensure the final product aligns with client needs and minimize miscommunications that could lead to unforeseen costs (C. J. Davis et al., 2006).

Elicitation interview

BAs will need to use the system during Elicitation Interviews to enable its intended functionality. Understanding this process is crucial. In the PACT Analysis (results visualised in figure 19), various location settings where the elicitation conversations occur are identified. These locations can range from within the company premises to external locations, such as client sites.

At each elicitation there is at least 1 business analyst and 1 client present. However the amount of analysts varies between 1 and 2, and the amount of client(s) or stakeholder(s) can vary between 1

1. INTRODUCTION

and up to 10. The amount of analysts determines the amount of tasks that need to be performed simultaneously, if 2 analysts are present one will be working on maintaining the conversation while the other is tasked with documenting the requirement relevant information. If an analyst is on their own all these tasks need to be performed by the single analyst.

During elicitation interviews, BAs rely on prompted questions to extract requirement-relevant information from clients or stakeholders. This is essential as clients/stakeholders possess unique domain knowledge and may struggle to express requirements effectively (C. J. Davis et al., 2006). Additionally, ambiguities in client communications can lead to misunderstandings regarding specific requirements (Ferrari et al., 2016), necessitating continuous clarification by the BA.

During elicitation interviews, BAs typically have multiple systems at hand to aid conversations and capture data. Laptops serve various purposes, including visualization support (e.g., presenting PowerPoint slides or relevant charts) and note-taking using text processors or recording capabilities. Other tools aiding data capture include portable conference speakers, improving recording quality, and traditional tools like notebooks and writing utensils. Online conversations often require laptops with audio and video recording hardware. Currently, several systems on the market integrate seamlessly with tools already used during elicitation conversations, enhancing their functionalities. These systems include drawing tablets and smart pens that merge writing/documentation with digital notetaking, enabling various computer-based interactions like copy-paste and moving notes. Additionally, conference speakers facilitate integration with conference software, enhancing conversation management with features like microphone controls (figure 2).

Elicitation settings typically fall into two categories: face-to-face conversations and online conversations. During elicitation interviews, BAs are tasked with multiple simultaneous activities that require consistent or improved performance. Conducting elicitation interviews requires BAs to juggle multiple mental tasks simultaneously. They must maintain conversational flow, actively listen to participants, and provide appropriate responses (Bano et al., 2019). Concurrently, they're constructing a mental framework of domain goals, rules, and application views to interpret client information (Ferrari et al., 2016). conversations can range from a few hours to multiple days, thereby making it not only likely for the analyst to miss out on certain information, but also cognitively demanding as they would need to focus both on the note-taking and on keeping a natural flow(Spijkman et al., 2023).

Capturing requirement relevant information for analysis

Upon concluding a conversation, BAs must distill the relevant information into structured requirements, which are then compiled into a Requirement Artefact. This artefact serves as the foundation for project requirements and evolves over time (Eger et al., 2012). While there are various requirement formats, there's a focus on User Stories, a widely used format in the software industry and agile practices (Lucassen et al., 2016). The User Story format typically follows: "As a <role>, I want <goal>, [so that <benefit>]" (Lucassen et al., 2016).

Notetaking

Presently, the predominant method involves memorization and note-taking, with tools like Word and Excel being popular for documenting requirements. Note taking serves several functions for the note taker. While one primary function is to reproduce information, it is not the sole purpose. If reproduction were the only function, more efficient methods, such as recording conversations, would suffice(Carrier & Titus, 1979). Note taking also functions as an external memory device, storing data for later retrieval and study, as described in literature (Berrezueta, 2023). Another crucial function is that note takers encode data by reorganizing it, making it their own (Berrezueta, 2023). This function is considered even more important than mere reproduction (Berrezueta, 2023).

Taking notes from both oral and written verbal presentations is widely accepted as a useful strategy for enhancing information retention (Carrier & Titus, 1979). When individuals actively engage in material they are trying to learn, such as by taking notes, their memory improves, particularly at deeper levels of comprehension, such as the situation model level. Research supports that note taking can enhance performance and that reviewing notes of any type increases recall compared to not reviewing notes (Carrier & Titus, 1979). Therefore, note-taking emerges as a potent cognitive tool, facilitating both encoding and retrieval processes to promote effective learning and memory retention.

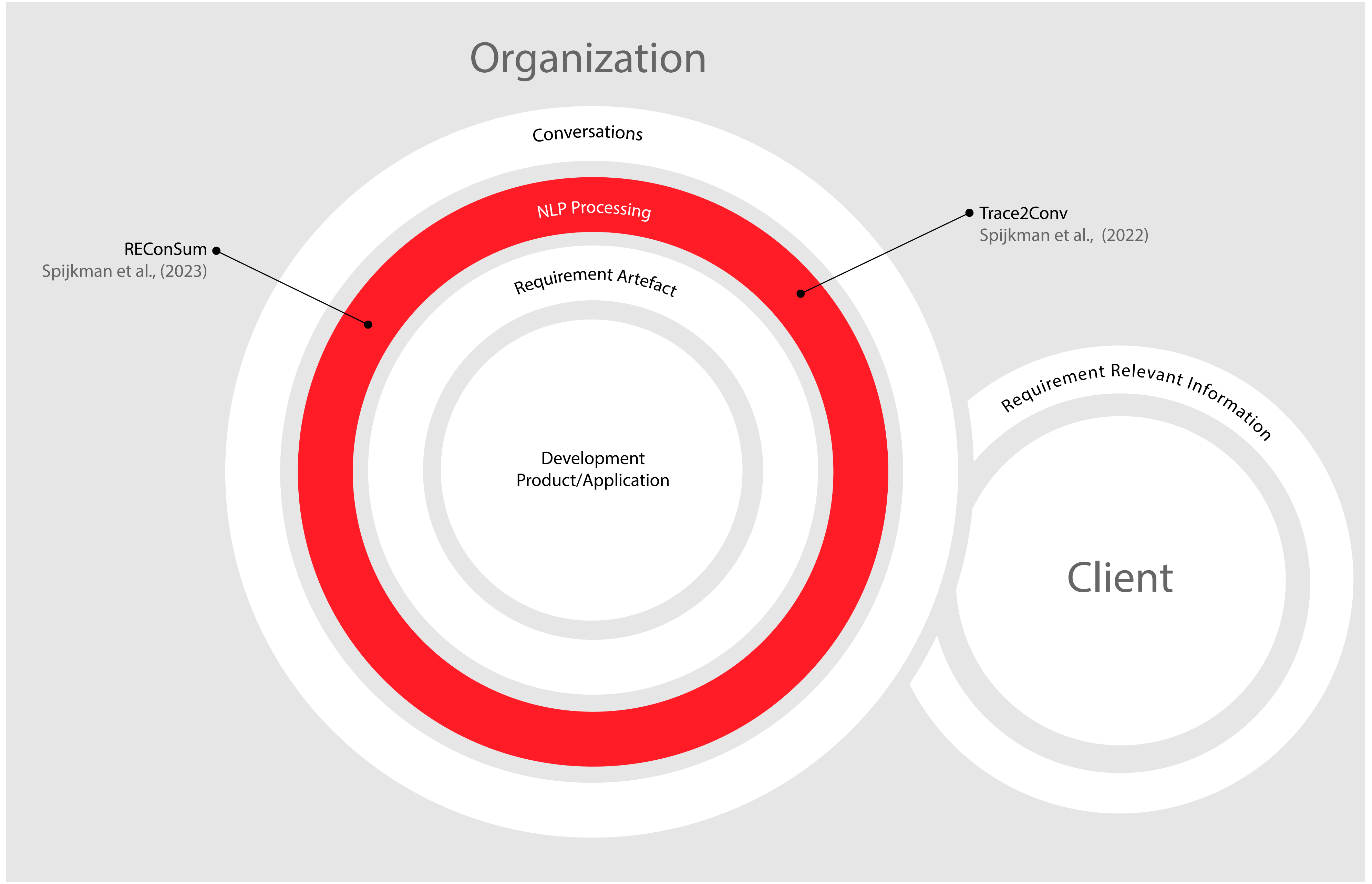


Figure 1: The initial systems positioned in the flow of information from the client to the development team.

1. INTRODUCTION

FIGURE 2: MARKET ANALYSIS OF SYSTEMS THAT INTEGRATE FUNCTIONALITY IN SYSTEMS USED DURING CONVERSATION.

PAPER TABLETS, SMART PENS, DRAWING TABLETS
Linking the physical act of writing to the digital domain.

CONFERENCE SPEAKERS
Integrating microphone, speaker and conversation management functionalities in portable format.

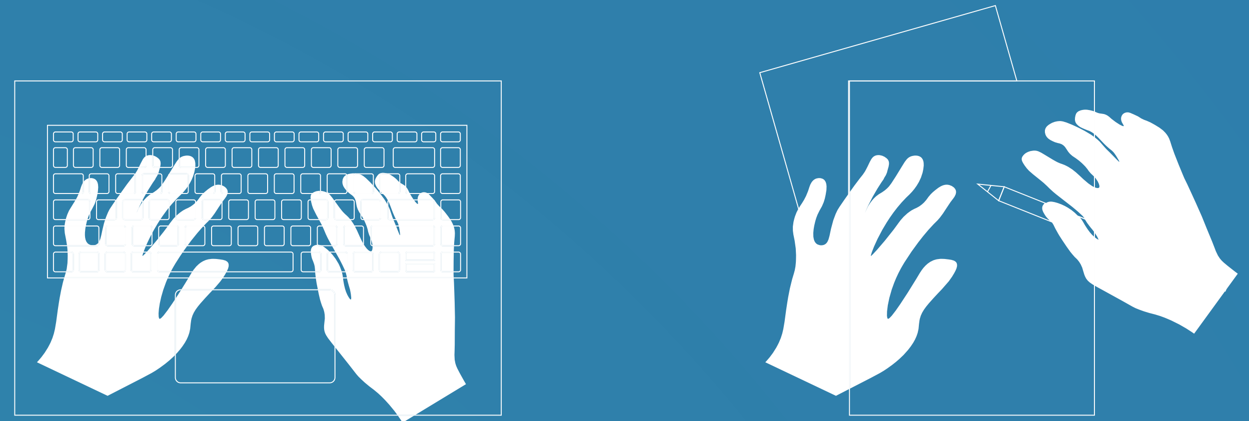


Figure 3: Use/positions of hands during a Elicitation conversation to take notes.

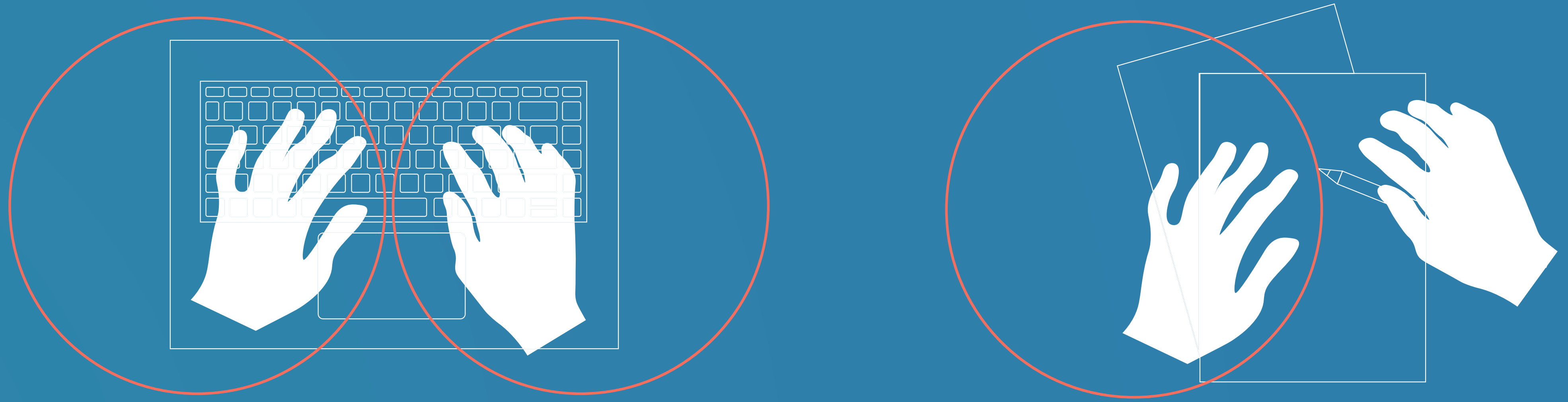


Figure 4: The aim is that a product is within reach during notetaking (~30 cm). One free hand is available for the use of a product (when writing) or a product can be placed closeby.

1. INTRODUCTION

Memorization

Amidst these mental activities, BAs must also capture requirement-relevant information for later processing. Notetaking serves this purpose, but the constraints of multitasking limit the depth and context captured in notes. Human memory supplements notes but isn't sufficient for comprehensive data retention (PACT analysis). Humans possess two types of memory: working memory (or short-term memory) and long-term memory. During a conversation, relevant information resides in working memory, essential for immediate processing. However, once the conversation concludes, this information transitions to long-term memory over time, as working memory has limited capacity. Retrieving this information later for processing into requirements poses challenges, as human brains aren't inherently optimized for recalling facts (Johnson, 2021). The coordination required for memory recall increases the likelihood of errors or incomplete recall, potentially leading to loss or misremembering of relevant information (Johnson, 2021). Factors such as the duration of the conversation, the number of participants, and the frequency of conversations can further impact the ability to recall specific details. For instance, longer conversations or those with multiple participants increase mental load and complexity, reducing the ability to memorize all relevant information. Additionally, if multiple conversations occur in quick succession, there may be overlap in memories when recalling specific information. Time pressure to process conversations into requirements can also impair processing ability. Given the imperative for complete and accurate requirements, these challenges underscore the value of tools to aid in conversation processing for BAs, the use case that has been argued in this thesis.

Recording

Sometimes a recording device is used to record the meeting to rewatch later. The big difference between online and face to face conversations is that in online conversations all participants use a laptop with a microphone and camera to communicate with each other in an online meeting room. For online conversations a record functionality is available allowing to record the conversation to rewatch later. This is a low effort interaction (one click) that is already widely used in online meetings. The recordings of requirements conversations contain valuable information that can easily be lost in the overall picture of the elicitation (Spijkman et al., 2023) but manually listening back to a conversation of multiple hours or days takes a lot of time and requires also listening to a large conversation dataset of irrelevant information. Such investment of time is not available to everyone and inefficient.

Challenges and Opportunities

Efforts to automate of Requirement Engineering processes using Natural Language Processing can be widely found in literature of which Meth et al. (2013) provides an extensive overview of the state of the art in the automation of Requirement Engineering processes. 4 types of tool categories are defined called Abstraction Identification, Requirements Model Generation, Requirement Quality Analysis, and Requirement Identification (Meth et al., 2013). Fizor also has produced two functionalities/systems/tools that make use of NLP with the aim to improve processes within the Requirement Engineering workflow, specifically automatically processing conversations to enable different functionalities for users (figure 1).

1.3) Fizor

History of Fizor.

Fizor part of the Forza IT group. Forza IT group, established in 2007, finds its origin in the company Forza Consulting as it has been split off from the company. Forza Consulting is in origin a software house that handles everything from management to execution(Maurice Hoog, 2021). Forza Consulting stands as a prominent provider of Oracle NetSuite solutions, specializing in delivering comprehensive ERP and CRM services to businesses spanning diverse industries. With a commitment to excellence and supported by a team of seasoned experts, Forza Consulting empowers organizations to streamline operations, enhance efficiency, and drive growth through innovative technology solutions.
Over time, specializations have emerged within the Forza IT group, each deserving its own showcase to better reach their specific target audience. For example, Scanman has been established to provide automation solutions worldwide, and Fizor, established in 2019, is a new branch for business application consultancy within the low-code-now-code IT landscape (Maurice Hoog, 2021).

Fizor, the independent Low Code specialist of the Netherlands

Fizor stands as a pioneering consultancy firm specializing in the delivery of cutting-edge low-code solutions tailored to clients' unique requirements. Collaborating with state-of-the-art platforms, Fizor crafts bespoke software

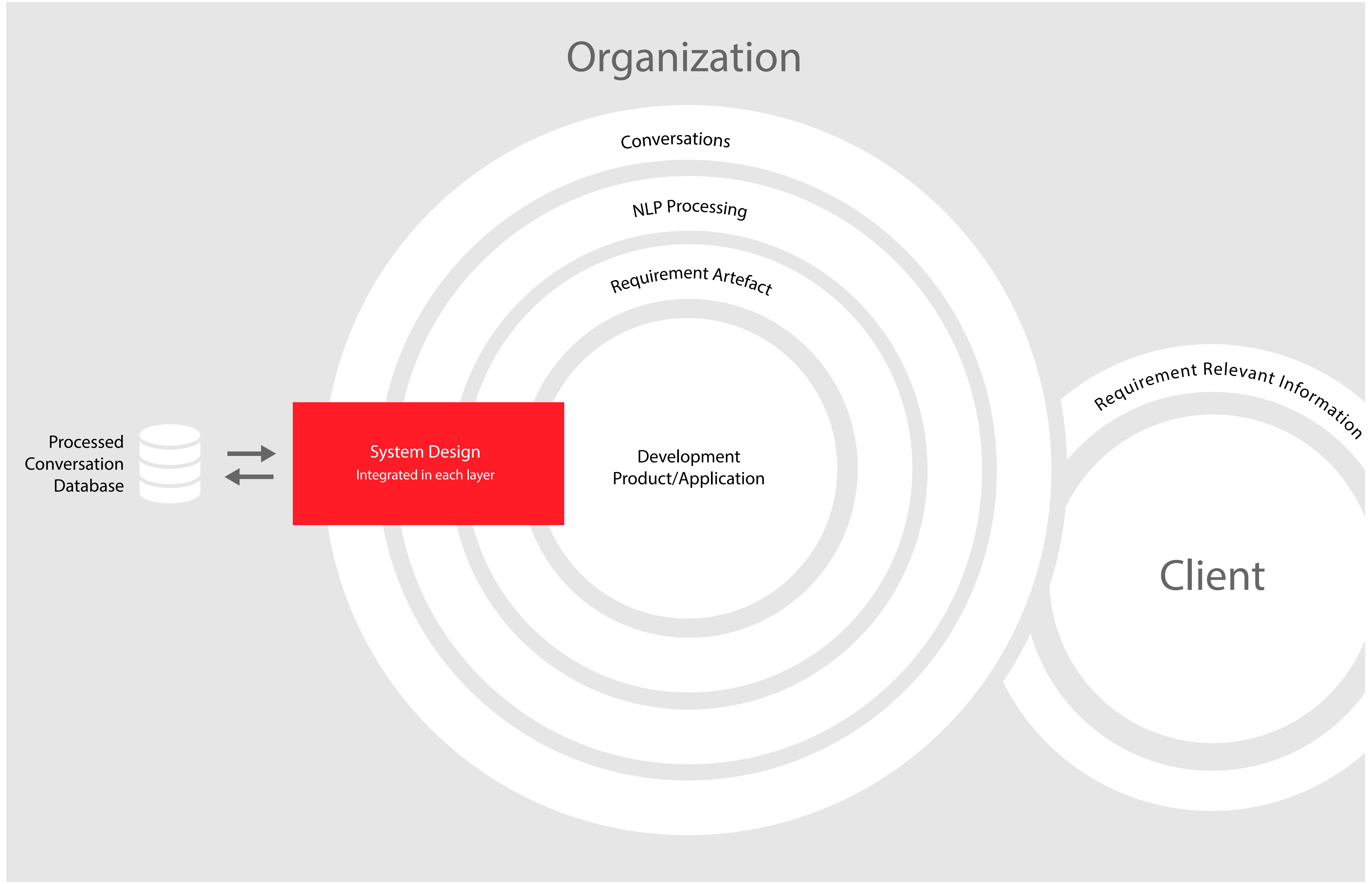


Figure 5: Conceptualization of the desired future system.

1. INTRODUCTION

applications that precisely address client needs. Operating within the low-code subdomain, a field coined by Forrester in 2014, Fizor benefits from its affiliation with the Forza IT group, established in 2007, to amass extensive experience and expertise in the low-code market. Since its formal establishment in 2019, Fizor has strategically leveraged this accumulated knowledge to establish a prominent foothold in the low-code market. Fizor has branched in logistics, wholesale, and the manufacturing industry. With in-depth knowledge of low code platforms like Thinkwise, Novulo, Betty Blocks, WEM, and USoft fizor guides clients in choosing the most suitable path.

Low Code Development Platforms (LCDPs) represent a transformative toolset allowing both programmers and non-programmers to swiftly develop and deploy business applications with minimal reliance on coding, environment setup, and training efforts (Waszkowski, 2019). By employing a graphical user interface, LCDPs streamline the development process, offering an accessible pathway to application creation. With a burgeoning adoption rate among companies, the utilization of low-code solutions heralds a significant advancement in the realm of essential business application development (Waszkowski, 2019).

Fizor, based in Utrecht, Netherlands, is a small company with approximately 30 employees. Their specialty lies in agile development, delivering custom solutions tailored to clients' needs. Positioned as the leading independent low-code specialist in the Netherlands, Fizor aids in modernizing application landscapes and bringing client projects to fruition. Their approach involves deep dives into processes that require enhancement, aligning with clients' IT strategies. Evaluation of low-code solutions is carried out with a commitment to honesty and prioritization of client interests. Fizor excels in crafting and implementing ingenious solutions, covering a broad spectrum of applications customized to meet client requirements. Their dedication to transparency ensures adherence to both time and budget constraints. What sets Fizor apart is their versatility in selecting the most suitable solutions for each client, collaborating with esteemed partners like Forrester and Gartner to harness proven technologies for digital innovation. With Fizor's low-code systems, clients benefit from fully optimized solutions tailored to their objectives and processes, boasting development speeds up to 10 times faster than traditional high-code methods.

Fizor's interests in Conversational RE

Fizor has strong ties to the University of Utrecht (UU) through the research of T. Spijkman's Ph.D. research on Conversational RE. Through this collaboration two systems have been realized called REConSum (Spijkman et al., 2023) and Trace2Conv (Spijkman et al., 2022).

REConSum (Requirements Elicitation Conversations Summarizer), described as "a NLP prototype tool that can assist practitioners and researchers in processing elicitation conversations by summarizing the transcriptions and extracting requirements-relevant information" (Spijkman et al., 2023), streamlines information processing by detecting relevant questions in interview transcripts.

Trace2Conv represents an initial effort to establish backward traceability from requirements to relevant transcript segments in requirement conversations (Spijkman et al., 2022), This is achieved through matching requirements to speakerturns based on tokenization and lemmatization techniques.

Fizor has indicated the future goal of implementing the functionalities of REConSum and Trace2Conv in practice as part of a larger system. Figure 5 presents a clear overview of the aim of the client regarding the future. The future system automatically captures conversations and processes them using the functionalities of, at least initially, the functionalities of REConSum and Trace2Conv, and stores these processed conversations in a shared conversation database, where the data of the conversations can be used to enable different system functionalities and to aid in different activities through the agile software development process.
To achieve this goal Fizor wants to know how these functionalities can be used to aid in industry practices. The ask is to explore Conversational RE practices in industry and through prototyping gather user data that can be used in further research and development. The next chapter will cover the design method and approach that has been used to present a design solution to the proposed problem or question.

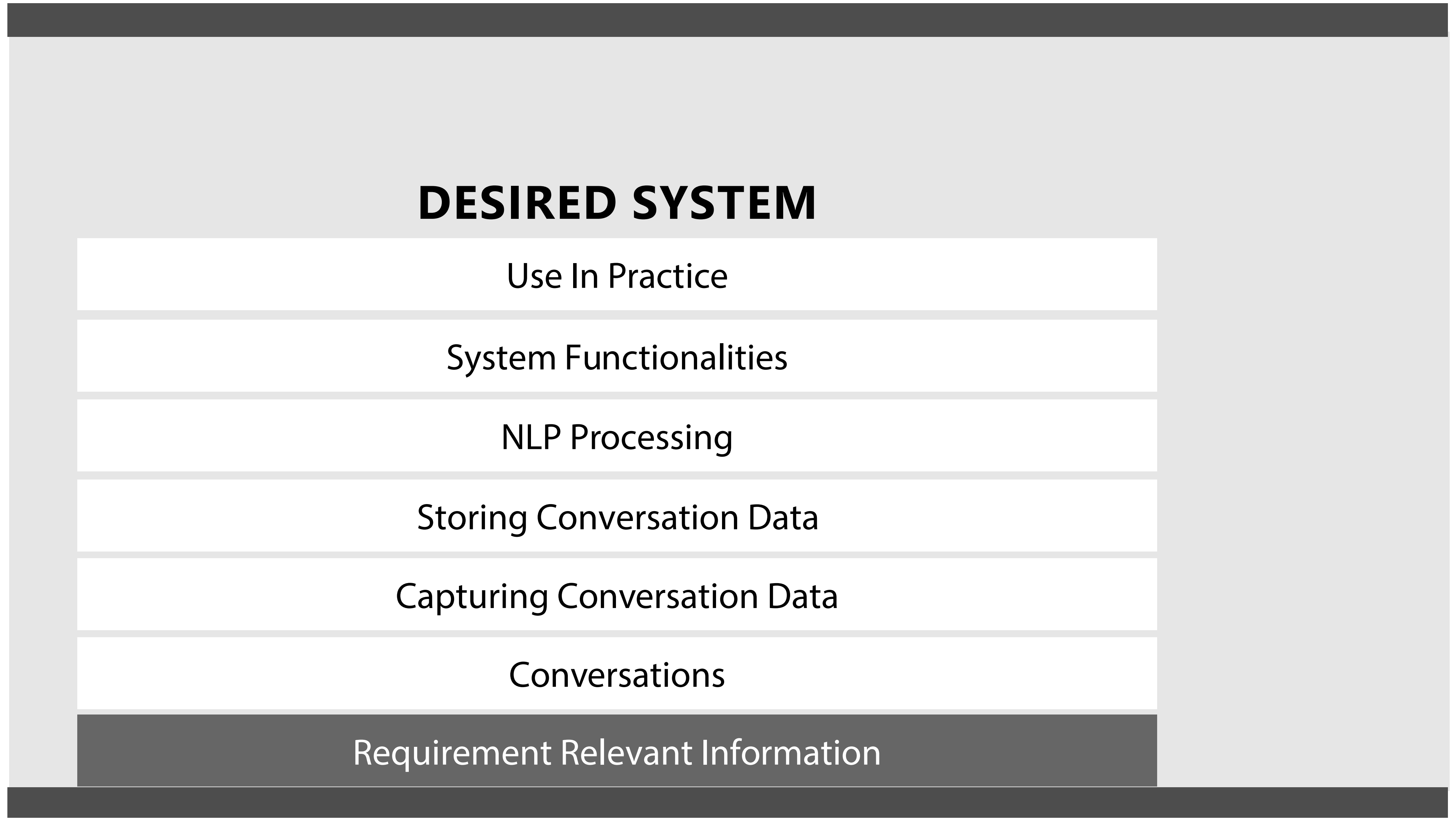

DESIRED SYSTEM
Use In Practice
System Functionalities
NLP Processing
Storing Conversation Data
Capturing Conversation Data
Conversations
Requirement Relevant Information

Figure 6: The foundations of the desired system.

2. DESIGN APPROACH & METHOD

Gathering use data would require the realization of some sort of prototype for a user to interact with, so it can provide feedback on the interaction. Because prototyping plays a central role the design phases mentioned in Leary & West (2023) have been considered in which system prototypes would be designed on paper, wireframe, and functional prototype level and evaluated with users to determine the best ideas (figure 8). Three metrics are important in the evaluation of the usability of a product or system. These metrics are effectiveness, how well the task has performed with and without using the system (defining measurable performance factors), on efficiency, how much time takes it to perform with and without using the system (measuring time), and how well the system performs in regards to user friendliness, allowing users to provide feedback on how they experienced using the system and if they liked it or not.

Preliminary research presented some knowledge gaps from the field of IT that needed to be filled in to make the design of a solution possible. The main knowledge gaps were domain specific terms that made some papers hard to understand and the other were the systems in which REConSum and Trace2Conv were developed and the code of which the systems consisted. To bridge the gap between Industrial Design Engineering knowledge and IT, the system to be designed and the initial systems of REConSum have been approached and conceptualized as systems, allowing system engineering principles to be applied and allowing to create an understanding of the initial situation and to ideate with them without understanding the underlying code. Multiple client conversations and whiteboard sessions where used to get clarification on domain language and parts of REConSum and Trace2Conv in the early phases of the design process.
The design approach for this thesis was structured around the waterfall model, with distinct phases including problem analysis, development/design, validation, and evaluation. While this framework guided the process, design activities did not necessarily follow a strictly sequential fashion. There were two approaches considered in this thesis, both revolving around getting user data from use in practice, to provide a design solution to the proposed problem. Both approaches originated from the problem analysis where a knowledge base needed to be formed. To structure the analysis the "cubic design approach" by Mohammad Rajabali Nejad (2020) was used. This involved a systematic methodology for analyzing and designing systems, grounded in fundamental system engineering principles and entailed analyzing the system, relevant stakeholders, and the environment to elucidate their intricate interactions (figure 9).
In the system analysis REConSum and Trace2Conv need to be first analysed on system level (what is their functionality?). After that analysis on subsystem level (what subsystems make up the system and what is needed to achieve the system functionality, so what outside systems and interactions are needed to achieve this). To design with the algorithms already developed to create a functional prototype also the supersystem (the overarching system that runs the systems of REConSum and Trace2Conv) needed to be analysed and the component level (the code that makes up the algorithms).
For the human or stakeholder analysis and the interaction with the systems a understanding of Conversational RE, the stakeholders to consider, best practices in Conversational RE, the context in which Conversational RE is performed and the technologies and products that are used within its context. Also, challenges related to Conversational RE needed to be identified. Information from Conversational RE in industry practice was needed in order to search for abovementioned topics and to search for systematic problems in relation to Conversational RE that could be solved with the introduction of a system design that processes elicitation conversations. A main user for the system has been identified, the Business Analyst as he has the closest ties to Conversational RE as he is the one planning, performing, and processing the elicitation interview. Other stakeholders with interests in processed conversation data can also have interest in the system that will be designed but are out of the scope of this thesis. Figure 7 presents the knowledge base that could be formed within the timeframe of this thesis.
The environment analysis entailed all the environmental elements like the contexts, other products and systems used in relation to the system design or Conversational RE, or other environmental factors that needed to be considered in the system design. Most of this information has been gathered through the interviews with Business Analysts.

In the creation of the knowledge base as part of the first approach users within the scope of this thesis were interviewed in the hope to uncover a common industry challenge that has relation to conversational requirement elicitation or could benefit from processed elicitation conversations. However, while each Business Analysts was within scope, the only thing that could be concluded in this research was that there is a lot of variation between projects, between, clients, and between organizations. Challenges would present themselves in occasion instead of structural challenges that could try to be solved through the introduction of multiple design solutions that could then be validated through user testing.
The seconds approach revolved around introducing a prototype into practice that would include the functionalities of REConSum and Trace2Conv. Fizor has the benefit that it falls within the scope of this thesis and uses Conversational RE themselves to elicit requirement relevant information from clients. A opportunity was identified to implement a prototype in the workflow of Business Analysts to identify user requirements through use in practice.
In the chapter problem analysis through the analysis mentioned above requirements have been formulated that the system design must meet. The goal of this thesis is to create a system prototype for Fizor that can be introduced in the Conversational RE workflow without impeding Business Analysts in their activities, tailored to a specific use case, potentially providing value to the organization itself, and providing Fizor with the means to use the generated outputs and user feedback in further research and development. The chapter realized system will cover the system that has been realized in the end, validated in the chapter validation, and is accompanied with a implementation plan (chapter implementation) of how to implement the system within Fizor (including pointers for further research and development). A manual of how the prototype has been constructed has been included in the appendix for Fizor.

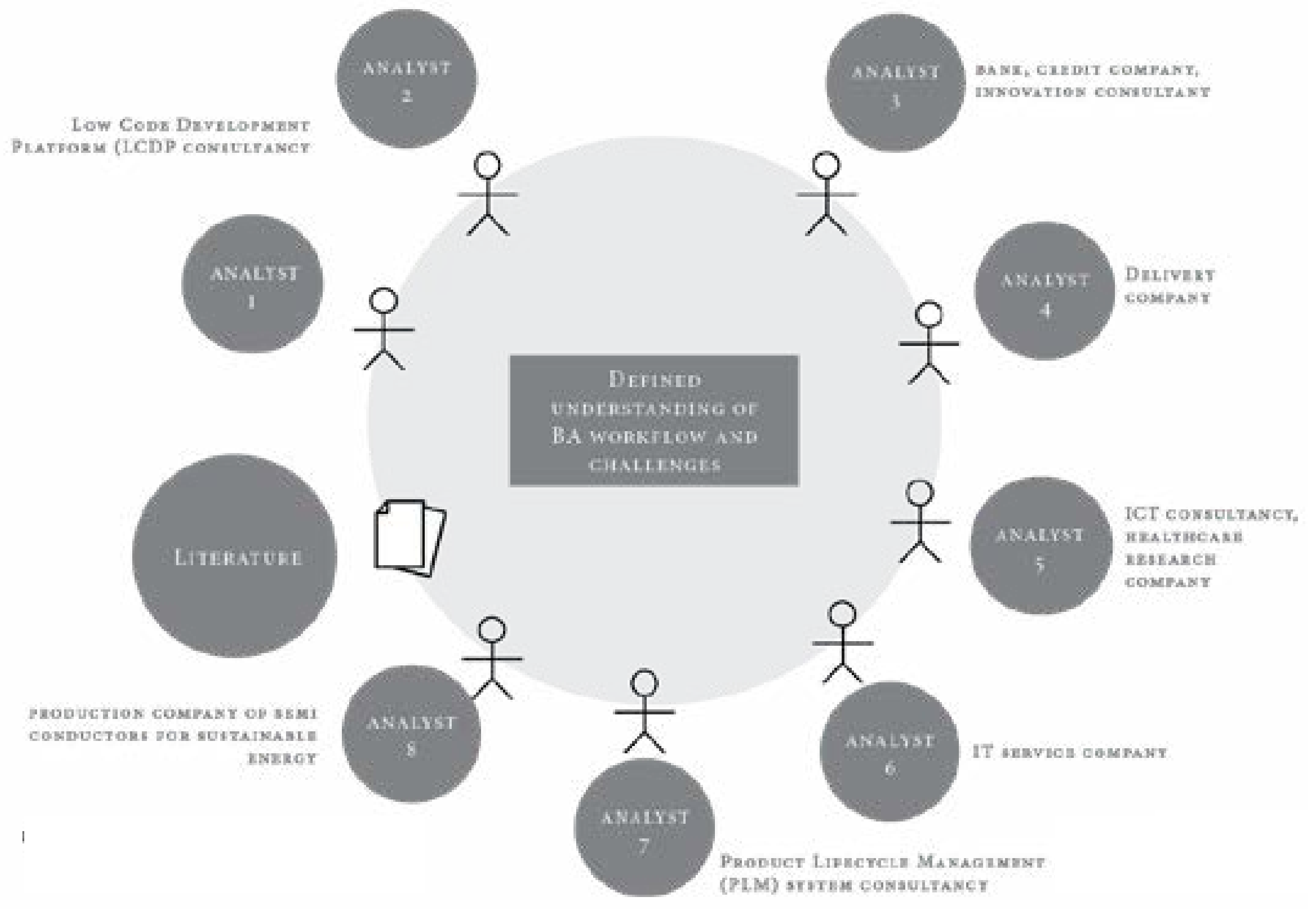

Analyst 2
Low Code Development Platform (LCDP consultancy
Analyst 3
Bank, credit company, innovation consultant
Analyst 1
Analyst 4
Delivery company
Defined understanding of BA workflow and challenges
Literature
Analyst 5
ICT consultancy, healthcare research company
Production company of semi conductors for sustainable energy
Analyst 8
Analyst 6
IT service company
Analyst 7
Product Lifecycle Management (PLM) system consultancy

Figure 7: Sources of information to form an understanding of BA workflow and challenges.

2. DESIGN APPROACH & METHOD

USER EXPERIENCE DESIGN AND EVALUATION METHODS (LEARY & WEST, 2023)

METHOD	DESIGN PHASE				DATA SOURCE
	FRONT-END ANALYSIS	PROTOTYPING			
		PAPER	WIREFRAME	FUNCTIONAL	
Ethnography	●				Single user or users
Focus groups	●	●			Group of users
Card sorting	●	●			Single, multiple or group of of users
Cognitive walktrough		●	●	●	Expert
Heuristic evaluation		●	●	●	Expert
A/B testing		●	●	●	Multiple users
Think-aloud				●	Multiple users
EEG/ Eyetracking				●	Multiple users
Analystics				●	Multiple users

Figure 8: Evaluation methods of user experience in different phases of the design process.

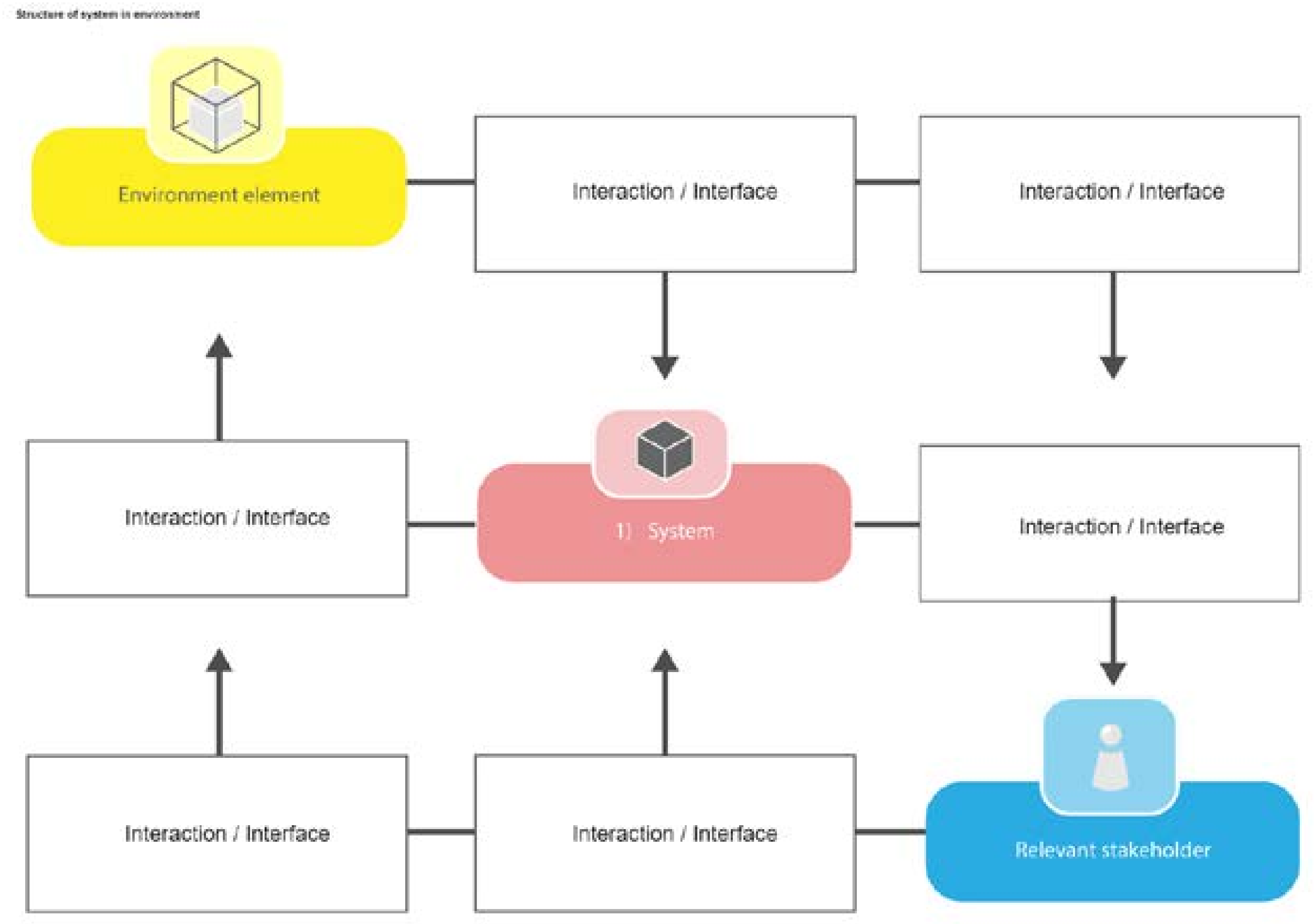


Figure 9: Analysis format of the system and its related elements and interactions between these elements.

3. PROBLEM ANALYSIS

In the introduction of goals and needs of Fizor a desired future system has been formulated. The desired future system consists of foundational layers (figure 6) that cannot function without the layer beneath it. Before the analysis is discussed let's first define what is meant by each layer to give directions to the requirements of such a system. The scope of Conversational RE results in the first two layers of Requirement Relevant Information and conversations as they form the basis for this type of elicitation. With capturing conversation data is meant the recording of conversation data and processing this into natural language in a specific format (Conversation Artefact. This Conversation Artefact needs to be stored in conversation database so it can be used by the next layer of NLP functionalities. Through different functionalities the stored conversation is annotated in multiple ways to enable different system functionalities where a Graphical User Interface needs to be present to allow users to interact with the system. Finally use in practice entails a specific user and use environment where the system can be, and is used for one or multiple usecases. To understand what is currently present a system analysis has been performed to identify what has been currently realized.

3.1) System analysis | REConSum and Trace2Conv

An in-depth analysis of the initial functionalities of REConSum and Trace2Conv has been conducted to discern the various system components, functional requirements, and technical specifications necessary for their utilization. For a comprehensive understanding of each system component within these functionalities or sub-systems. Both systems will be explained in depth in this chapter (and all subsystem elements are also covered in the appendix chapter 2) but to give a preliminary overview of how the systems work: both systems use Natural Language Processing (NLP) to process conversations captured in natural language (conversation artefacts) into data that can be used to achieve different functionalities that can be useful in practices related to Conversational RE. Both systems convert conversation artefacts into a matrix where speakerturns (rows of conversation data) are annotated with different values so a system can understand the contents of the text stored within each speakerturns. REConSum identifies relevant questions in a conversation dataset in an effort to make it easier to detect requirement relevant information. Trace2Conv links a requirement artefact to corresponding conversations allowing users to trace backwards to speakerturns in a conversation that are linked to the formulated requirements.
First and foremost, both subsystems require a Conversation Artifact, abbreviated as CA, which is essentially a file storing the conversation in natural language. However, to enable the processing of conversation data, this conversation artifact needs to adhere to a specific formatting known as "speakerturns."

Speakerturns

A conversation can be documented using a set of speakerturns, wherein each speakerturn groups the spoken text transcribed in natural language with the person speaking and the corresponding time in the conversation. The expected format of the speakerturns by the system is: "[time] speaker: text" as can been seen in figure 10, an example of a set of speakerturns. This set of speakerturns need to be stored in a text file (.txt).

Natural language Processing

Both functionalities leverage Natural Language Processing (NLP) capabilities. NLP can be divided into two main parts: Natural Language Understanding (NLU) and Natural Language Generation (NLG), which respectively focus on comprehending and generating text (Khurana et al., 2023). REConSum and Trace2Conv fall under the NLU category, where input text is analyzed and annotated to make it understandable to a system. Figure 11 illustrates various types of annotations that can be associated with input text. Automating the processing of input text to reduce or categorize information can significantly benefit the conversational requirement elicitation process, given the large volume of conversation data that needs manual processing. Multiple solutions have been proposed in the literature to automate parts of the elicitation process using NLP. For instance, (Meth et al. (2013) categorize four tool categories: Abstraction Identification, Requirements Model Generation, Requirement Quality Analysis, and Requirement Identification, providing valuable insights into where and how to apply NLP to automate parts of the Conversational RE process. Stanford CoreNLP (figure 11) is an example of a high-performance tool that identifies syntactic and semantic information, as well as discourse context (Hirschberg & Manning, 2015). REConSum utilizes this tool to determine if a body of text contains a question or not (Spijkman et al., 2023). The tool offers a standard NLP preprocessing pipeline, including Part-Of-Speech (POS) tagging (e.g., noun, verb, preposition), identification of named entities (e.g., people, places, organizations), sentence parsing into grammatical structures, and identification of co-references between noun phrase mentions (Hirschberg & Manning, 2015).

Annotations

To read and process conversation data, the system creates a matrix that stores each speakerturn on a row, known as a DataFrame (refer to Figure 12 for a visualization of such a DataFrame). At this stage, the system can add multiple annotations to these rows of data. Currently, three types of annotations are added to the set of speakerturns: question, relevant question, and token.

REConSum identifies whether a speakerturn contains a question and further determines if the question is relevant using Part-Of-Speech (POS) tags and DialogTag (Spijkman et al., 2023). REConSum determines if these questions are relevant by assessing whether they contain domain-specific terms; with the assumption that the presence of those terms is an indicator of relevance (Spijkman et al., 2023).

To make this functionality work the manually installation of StanfordCoreNLP is required and the installation and inclusion of the TF-Wiki Corpus (a large dataset of natural language, see chapter 2 appendix) in the REConSum datafolder.

Meanwhile, Trace2Conv tokenizes each speakerturn (figure 12 & 13), treating each word as a token, categorizing them, and subsequently filtering and lemmatizing them to retain only the most relevant tokens.

Limitations of NLP

NLP serves as a means for systems to comprehend human language. Despite significant advancements, achieving a complete and accurate understanding of human language remains a challenge, and may always remain so. In an ideal scenario, there would be a 100% accurate conversion between NLP processing input and processed output, resulting in perfect recall and precision. Perfect recall refers to the proportion of relevant items (all occurrences of information that exactly matches search criteria of a user) actually retrieved in response to a search query, while precision denotes the proportion of retrieved items relevant to the query, often used to measure correctness (Meth et al., 2013).

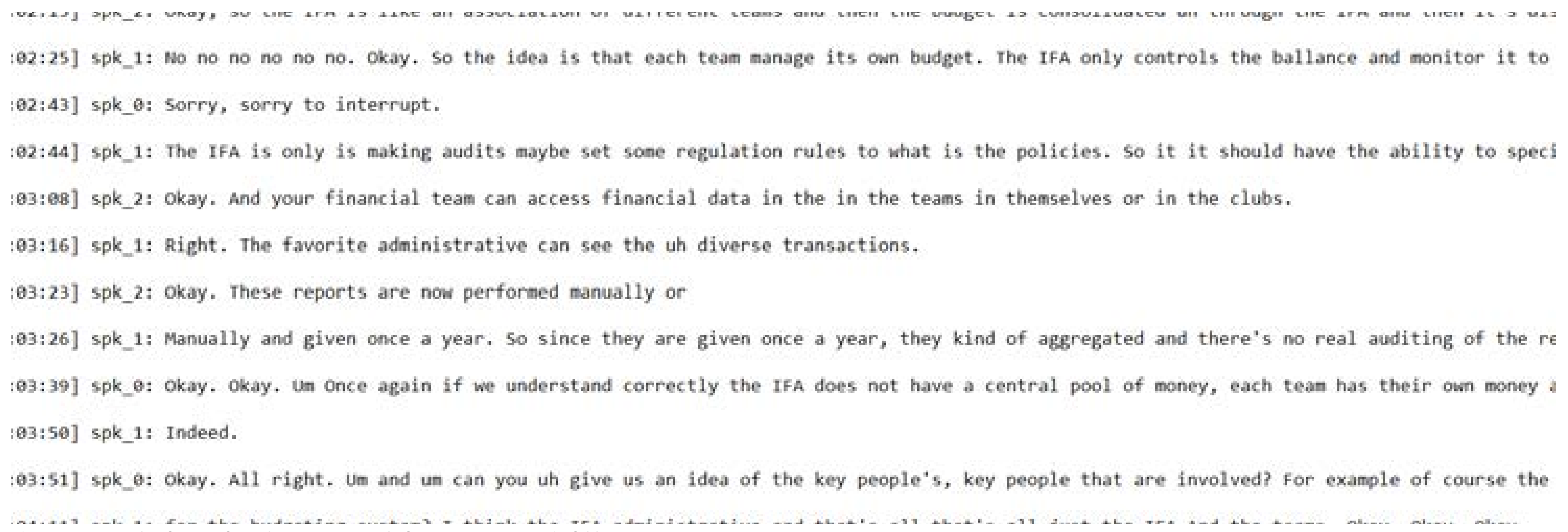
NLP encounters difficulties with certain aspects of human language that are inherently challenging for systems to grasp. Examples include contextual ambiguity, synonym handling, homonym confusion, understanding sarcasm and irony, ambiguity resolution, informal language and cultural specificity, domain specificity, misspelled or misused words, and predicting intention (Khurana et al., 2023). Consequently, there is a perpetual trade-off between recall and precision. In automated elicitation processes, prioritizing recall over precision is crucial, as errors of commission are easier to rectify than errors of omission (Meth et al., 2013).This implies that while systems can streamline processing for users, humans will always need to perform some level of processing, which can be categorized into two phases: the automation phase and the manual phase (Meth et al., 2013).

System functionalities

Understanding the purposes for which REConSum and Trace2Conv are utilized is essential since their functionalities must be integrated into the future system. REConSum serves to streamline the handling of large conversation datasets by filtering them based on the questions posed during the conversations. In a semi-structured interview scenario, displaying only the questions enables rapid inspection of the corresponding answers without the need to review the entire conversation. However, currently, the system only outputs the DataFrame to the user with annotations, necessitating the design of additional functionality. Trace2Conv facilitates swift navigation to context regarding requirements within a requirement artifact. This is achieved by establishing links between the requirement artifact and related conversations through token matching. Presently, features such as requirement overview, token overview, and speakerturn overview enable seamless navigation between the requirement artifact and the conversation(s) (refer to Appendix Chapter 2 for details).

Conclusion: Current situation

From this analysis a current system situation has been constructed as can be seen in figure 14. This overview helps to visualize what is already present on each layer needed to achieve use in practice. As can been seen in the overview the first solution that must be provided to the user is a method to capture Conversation Artefacts. Later in the chapter will be covered what is needed to make the systems function and the implications on the system requirements.



```
:02:25] spk_1: No no no no no no. Okay. So the idea is that each team manage its own budget. The IFA only controls the ballance and monitor it to
:02:43] spk_0: Sorry, sorry to interrupt.
:02:44] spk_1: The IFA is only is making audits maybe set some regulation rules to what is the policies. So it it should have the ability to speci
:03:08] spk_2: Okay. And your financial team can access financial data in the in the teams in themselves or in the clubs.
:03:16] spk_1: Right. The favorite administrative can see the uh diverse transactions.
:03:23] spk_2: Okay. These reports are now performed manually or
:03:26] spk_1: Manually and given once a year. So since they are given once a year, they kind of aggregated and there's no real auditing of the re
:03:39] spk_0: Okay. Okay. Um Once again if we understand correctly the IFA does not have a central pool of money, each team has their own money a
:03:50] spk_1: Indeed.
:03:51] spk_0: Okay. All right. Um and um can you uh give us an idea of the key people's, key people that are involved? For example of course the
```

Figure 10: Example of a set of speakerturns (Example from REConSum main folder).

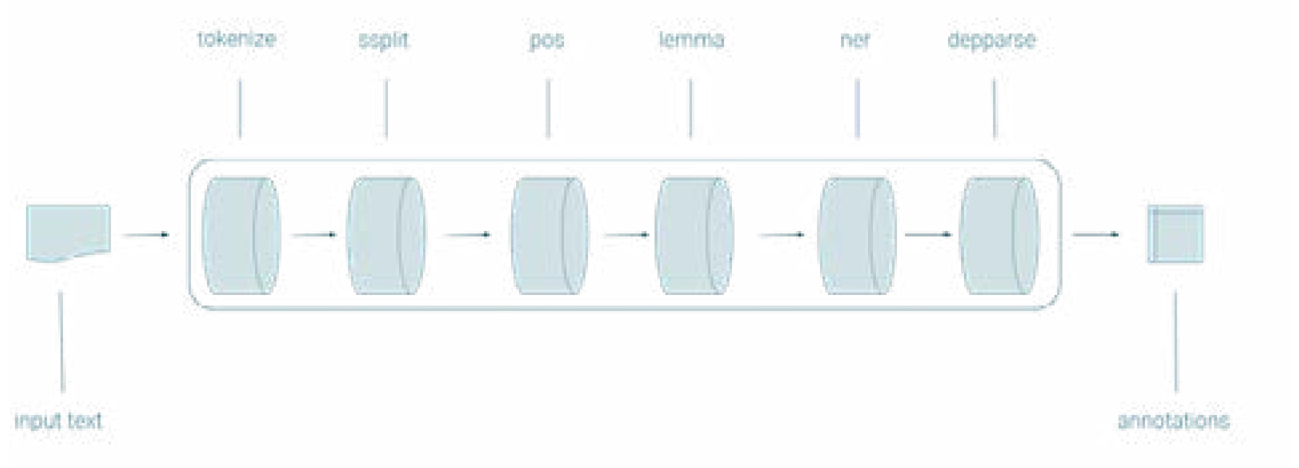
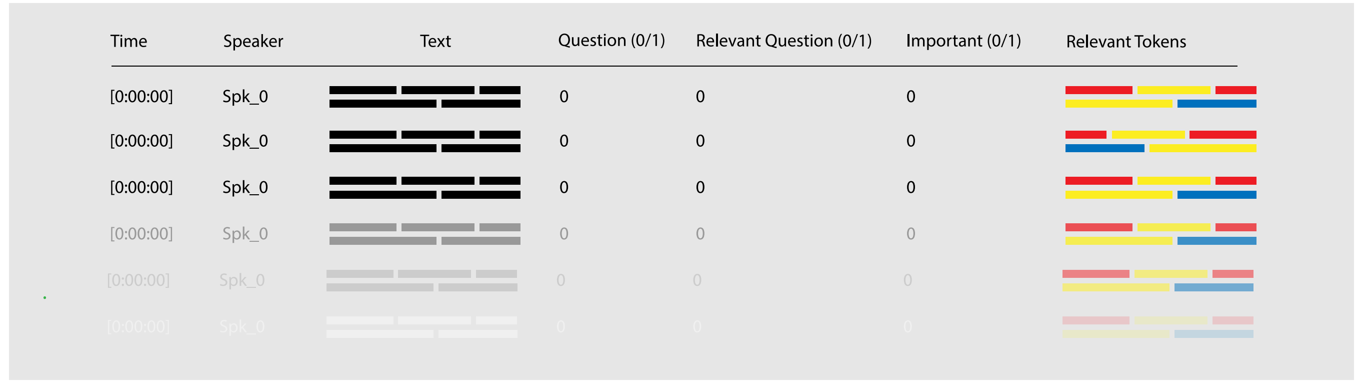
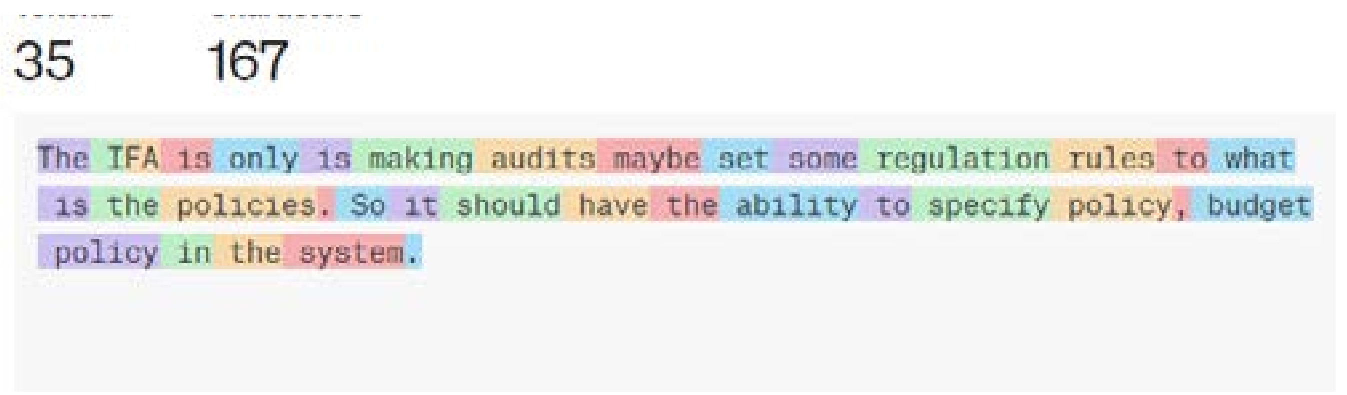


Figure 11: Stanford CoreNLP pipeline.



Time	Speaker	Text	Question (0/1)	Relevant Question (0/1)	Important (0/1)	Relevant Tokens
[0:00:00]	Spk_0		0	0	0	
[0:00:00]	Spk_0		0	0	0	
[0:00:00]	Spk_0		0	0	0	
[0:00:00]	Spk_0		0	0	0	
[0:00:00]	Spk_0		0	0	0	
[0:00:00]	Spk_0		0	0	0	

Figure 12: Visualisation of DataFrame with annotated speakerturns (important has been added in the system design)



35 167

The IFA is only is making audits maybe set some regulation rules to what is the policies. So it should have the ability to specify policy, budget policy in the system.

Figure 13: Tokenization visualized. Trace2Conv utilizes NLP to tokenize each word of a sentence for each speakerturn.

3. PROBLEM ANALYSIS

3.2) How to generate Conversation Artefacts

In Spijkman et al. (2023) is argued that recordings of conversations can hold useful information that can be missed by an Business Anaysts during the conversation due to mental demands of the elicitation process (as mentioned in the introduction). While manual transcribing of the conversation is a possibility this takes a lot of time and effort. Therefore, the proposition is to use modern online meeting tools like Microsoft Teams and Zoom to generate conversation artefacts due to its widespread use in online meetings, it's capabilities to record a conversation without much effort, and automatic transcribing functionalities. However, for this to be a suitable solution in practice the conversation needs to be converted accurate enough, so no requirement relevant information is converted incorrectly as of the high interests of the client and the development teams for a complete and correct set of requirements.

In the analysis of different meeting software and software to convert speech to text Microsoft Teams was deemed to be the best choice in this matter for a multitude of reasons that will be covered in this paragraph. First of all, it needs to be a system that is already used by the user and the organization in which the user operates. If the system prototype only works with meeting software that is used by no one else it will not be used because of the effort it would take to switch to a totally new system. As the system prototype will be first implemented within Fizor this means using the software that is primarily used there: MS Teams. It is also one of the mostly used meeting software in the world as it held the second-largest market share in 2023 (Sujay Vailshery, 2024b), 91 of the Fortune 100 companies utilize Microsoft Teams (Redmond, 2019), and has 300 million active daily users as of 2023 (Sujay Vailshery, 2024). Furthermore, when compared to other speech-to-text systems, such as Amazon Transcribe, Google Cloud, and IBM Watson, with both clean speech and noisy speech MS Teams performed the best (Xu et al., 2021). While this source is now a couple of years old it underscores that MS Teams is a good choice to perform the necessary tasks in this category.

Also, MS Teams integrates all the system functionalities that are required to capture requirement artefacts. These being recording the conversation, converting conversation speech to text, and formatting the text into speakerturns. One software solution performing all these functionalities is preferred over multiple systems that each need to be acquired, learned, and managed.

Furthermore, with an eye on the future, MS Teams has already products that are integrated with MS Teams, for example speakers, microphones, and even whole meeting room system. Fizor also has one of those MS Teams integrated meeting rooms and therefore it is a logical choice. Also MS Teams provides API's to use generated conversation artefacts in other systems (will be mentioned in this thesis but the implementation falls out of scope) and provides developers with the ability to use the MS Teams functionalities.

By implementing MS Teams to generate Conversation Artefacts of conversations that took place during this thesis and analyzing the generated results the performance of the CA generating system as is will be analysed. A method will be formulated and validated for users to capture conversations in online and face to face context.

3.3) System conceptualization

Users will need to interact with the system using a Graphical User Interface (GUI). To visualize this GUI and to ideate with useful interactions a usability flow has been created that incorporates all the system elements currently integrated in both REConSum and Trace2Conv (figure 16). This usability flow has been used as communication tool with Fizor and Business Analysts. Concluded from this activity is that the GUI needs to included at least a set of core interactions and functionalities.

Let's start by examining the data that needs to be presented to the user. All system outputs must be accessible to the user in the GUI. This entails presenting speakerturns of a conversation where the text, speaker, and index/time are clearly visible for accurate comprehension. For Trace2Conv specific, the system must present requirement artifacts and a token overview of all (filtered) tokens to the user.

Users should be able to filter speakerturns based on different system annotations, enabling them to toggle filters for questions, relevant questions (REConSum), and token views on and off (Trace2Conv). As mentioned in the paragraph on the Limitations of NLP, systems may process a conversation not 100% correctly. Users must have the ability to inspect and modify system annotations in case of processing errors. Therefore, users should be able to deselect the question and relevant annotations from speakerturns, as well as turn off irrelevant tokens and add relevant tokens (In figure 17 checkboxes are suggested to turn specific annotation on and off).

Database manipulation and navigation are crucial aspects. Users need a method to supply the system with desired inputs (requirement artifacts (Trace2Conv specific) and conversation artifacts). System data should be stored systematically to facilitate locating desired system outputs. Naming files and including search functionalities, such as a search bar, can enhance usability. The conceptualized system database (Figure 15) stores multiple projects, each containing multiple conversations (and one conversation artifact). To streamline searching, various data can be linked to a conversation, including title and description, conversation participants, date of conversation, and the set of speakerturns. Users should have the capability to add, edit, and remove projects and conversations to manage the database as needed.

Users must be able to navigate easily within conversations, between projects, and corresponding requirement artifacts. It's crucial for users to be continuously informed about the system's state and to include clear navigation options for seamless movement within the system (Johnson, 2021). Additionally, users should have the capability to navigate to speakerturns containing a specific token by clicking on that token within the requirements where they occur (Trace2Conv specific). To facilitate seamless interaction between the requirement artifact and speakerturns, a function has been conceptualized to display both artifacts simultaneously (Trace2Conv specific). The system's processing functionality will operate in the background after users have provided the necessary inputs and is not visualized in the usability flow.

3.4) What can be realized within the scope of this thesis

To design with the algorithms already developed for REConSum and Trace2Conv to create a functional prototype at the supersystem (the overarching system that runs the systems of REConSum and Trace2Conv) needed to be analysed and on component level (the code that makes up the algorithms). This analysis concluded in that REConSum was developed in Python and Trace2Conv was developed in BettyBlocks (a Low Code Development Platform). With limited knowledge of python and no knowledge BettyBlocks both systems were analysed on what was possible to realize with these systems (within the timeframe of this thesis and with the limited programming skills.

After analyzing the platforms and their capabilities, the decision was made to utilize the Python platform. With prior experience with the system and a wealth of documentation and a vast database of reusable codes (packages) that python offers it was deemed easier to realize a functional prototype compared to using a LowCode platform, especially given the limited experience with the latter. This doesn't mean that Low Code can't be used to develop the future system, it only means that with the available resources of time and skill this proved to be the best solution.

Using python, and therefore designing with the algorithms of REConSum comes with the added benefit of having access to the 'preprocessing algorithm'. This algorithm converts a conversation artefact into data that can be used by a system, a functionality that is necessary to make any NLP functionality work.

Developing an application in Python necessitates that users have Python installed, along with the required packages, on the hardware running the system. However, the installation process for Python, with its specific package versions, presents a significant use effort and time commitment. Knowledge of python or other programming languages should not be required knowledge of users and therefore a solution must be usable without any prior knowledge of or installing of python.

Figure 18 presents an overview of all the things that need to be realized to make REConSum usable.

3.5) Defining main user, implementation environment, and design problem.

To understand which activities related to Conversational RE can be supported, a stakeholder analysis identified four key stakeholder groups closely involved in the requirement elicitation process. The primary stakeholder is the Business Analyst (BA), responsible for conducting all Conversational RE processes. Additionally, clients/ stakeholders and development teams have

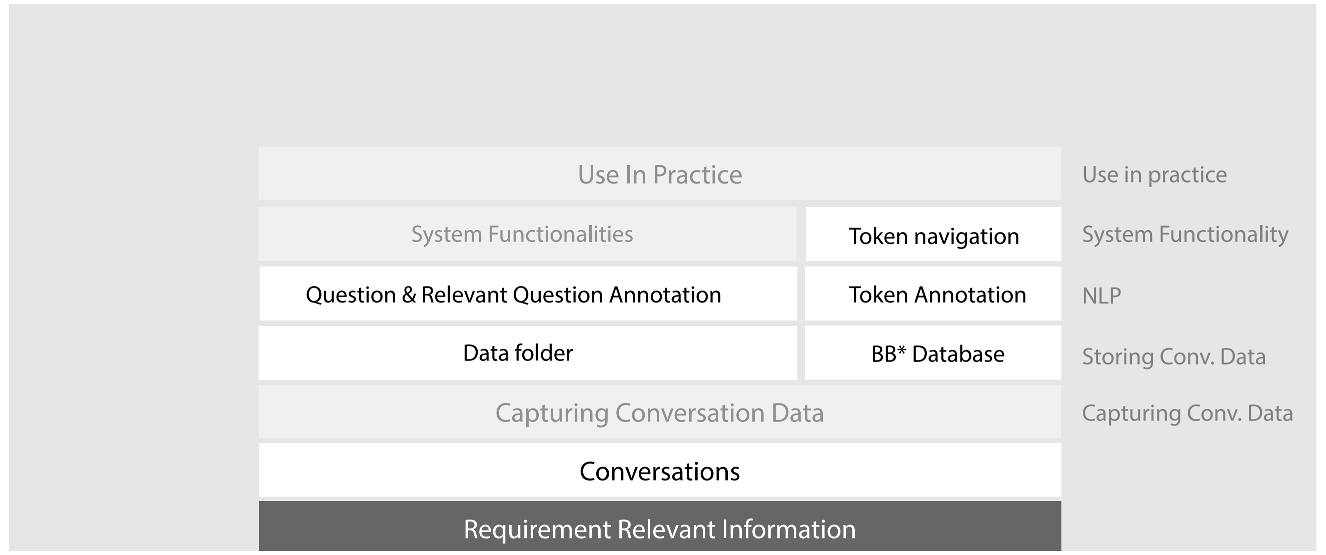


Figure 14: The starting point of this project. What has been yet realized with REConSum and Trace2Conv. *BB=BettyBlocks (LowCode Platform).

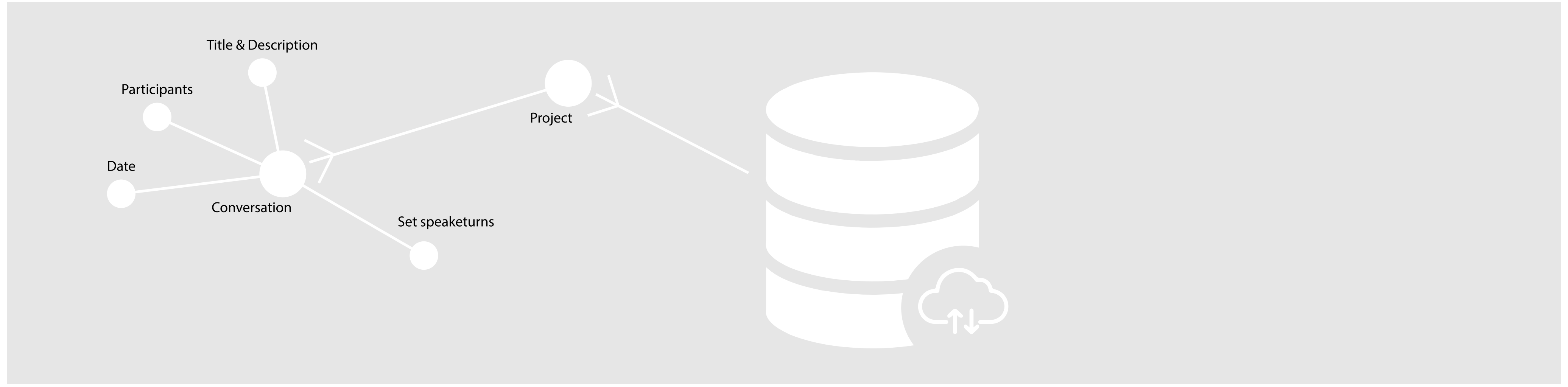


Figure 15: Conversation database concept.

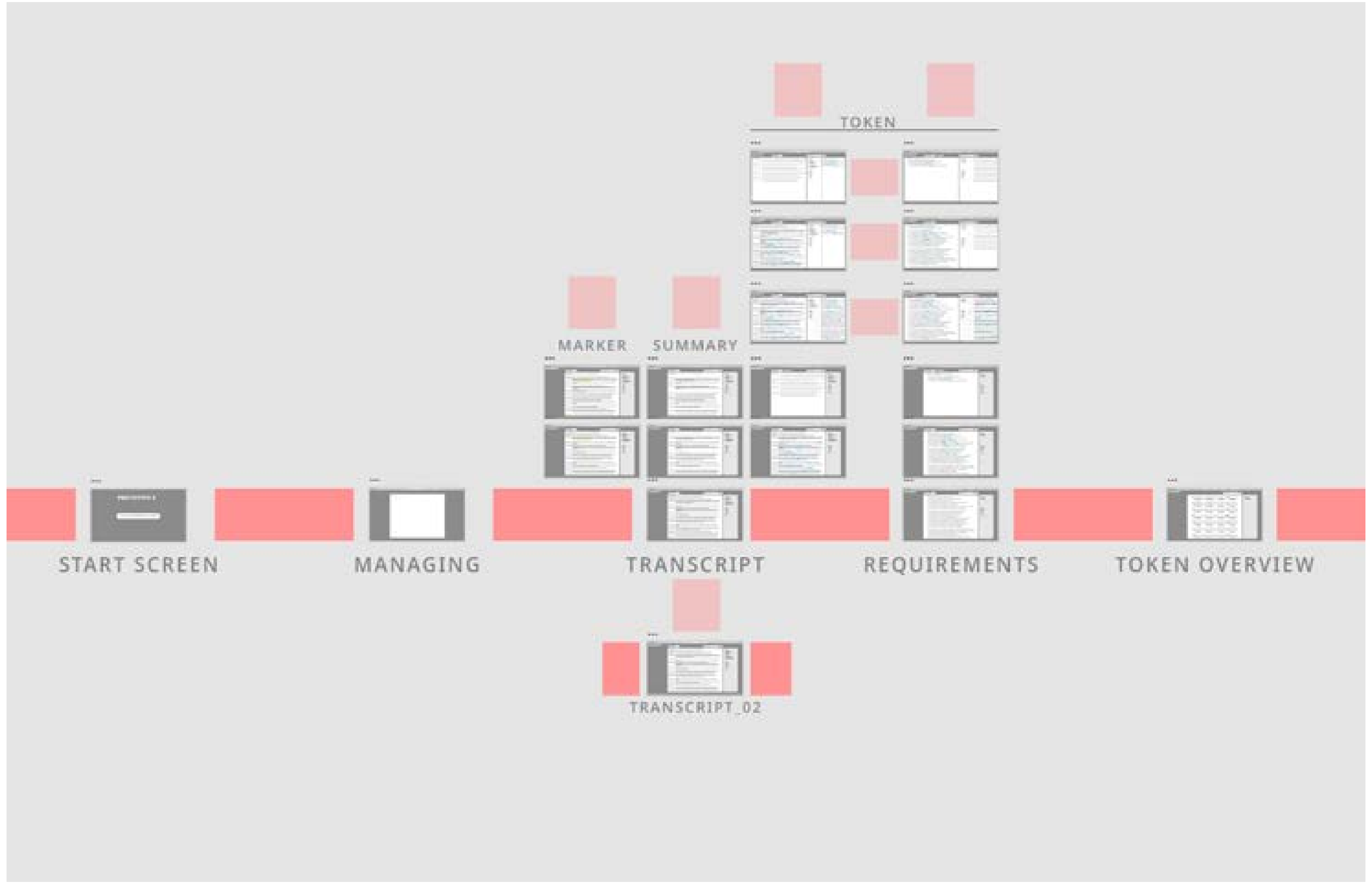


Figure 16: Wireframe of full usability flow. Separating managing function, transcript functions, requirement artefact functions, and the token settings

3. PROBLEM ANALYSIS

significant interaction with Conversational RE activities. The organization in which the BA and development team operate constitutes the final stakeholder group. It's assumed that stakeholders closer to the Conversational RE process have a greater interest in functionalities utilizing processed conversation data. Therefore, focusing on the activities of Business Analysts, who perform all Conversational RE activities, is a logical starting point.

A literature review provided insights into various Conversational RE activities across the agile software development process (Project Management Institute, 2016),best practices for conducting conversational requirement elicitation interviews (Bano et al., 2019), challenges associated with ambiguity in requirements and the cognitive processing demands on BAs during elicitation interviews (Ferrari et al., 2016), limitations in clients' and stakeholders' ability to effectively communicate needs (C. J. Davis et al., 2006), and the need for BAs to extract tacit knowledge from clients based on unique domain knowledge unknown to the BA. Meth et al. (2013) provided an extensive overview of automation in the RE process, serving as input for potential challenges in RE activities and a source of inspiration for different use cases in system design.
To gather anecdotal information, interviews were conducted with seven Business Analysts from different organizations within the scope of this research (interview plan in Appendix Chapter 4). These interviews aimed to gather insights into the various conversational requirement elicitation activities performed throughout the agile software development process. Discussions covered activities such as needs assessment, requirement management planning, requirement elicitation, requirement analysis, requirement monitoring and controlling, solution evaluation, and project closure, constituting the phases of the Conversational RE process (Project Management Institute, 2016). The goal was to verify BA workflows, identify challenges within these activities, and brainstorm different use cases for REConSum and Trace2Conv functionalities. These interviews yielded valuable information. Fizor's objective is to address industry challenges using the functionalities of REConSum and Trace2Conv. Ideally, a robust case could have been identified in this problem analysis where these functionalities are applicable, validated using measurable variables. Applying a case study approach to design potential solutions for a specific industry problem, with high user involvement in development and validation, could have been effective. Metrics such as effectiveness (achieving desired goals), efficiency (reduced time for specific activities), and user-friendliness (clarity and enjoyment of use) could have been applied to evaluate different design concepts. Figure 8 illustrates the type of design process focused on user experience that could have been applied when identifying use cases (Leary & West, 2023)

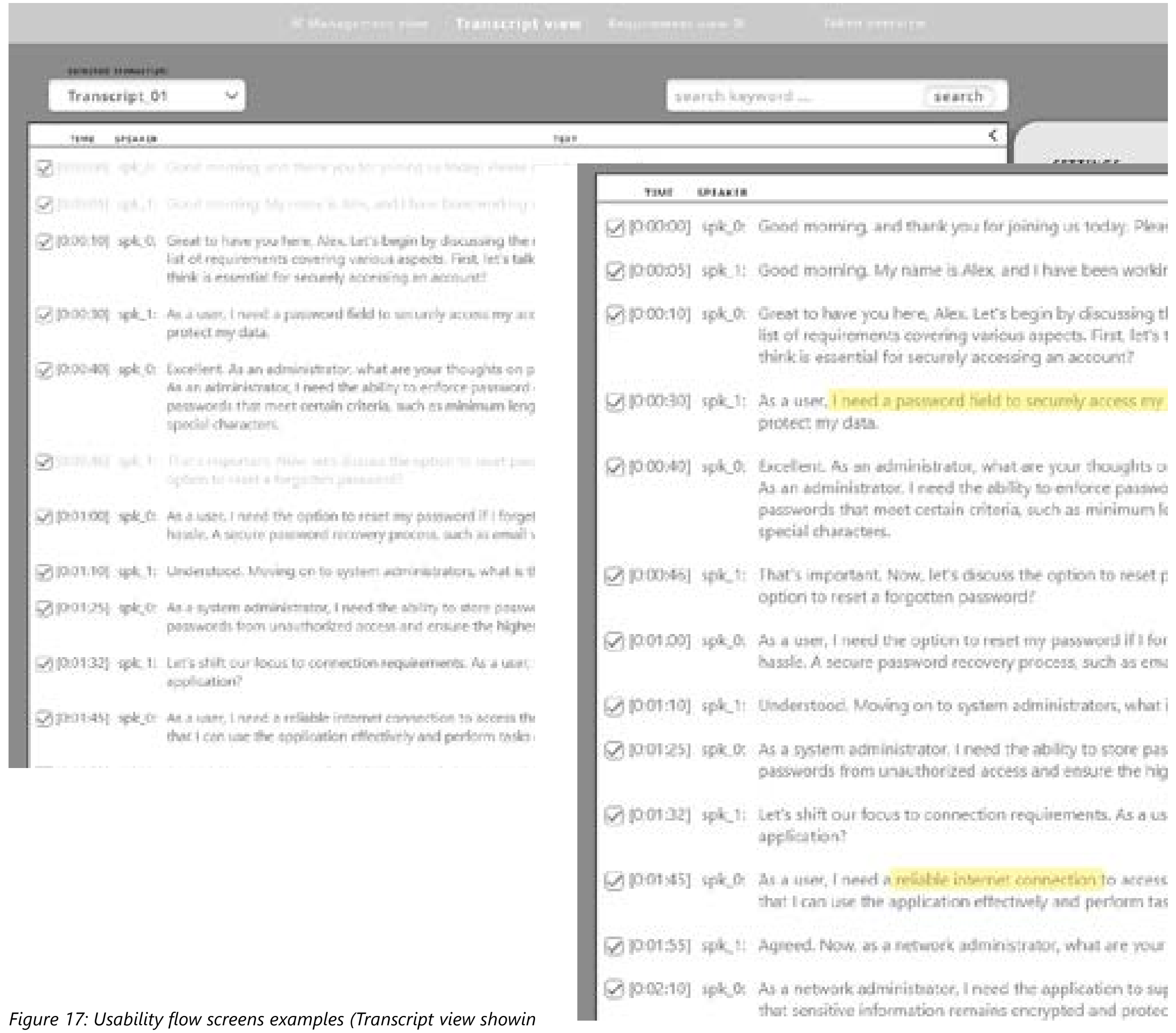
However, conversations with Business Analysts revealed significant variability in the (conversational) requirement elicitation process between organizations (e.g., work methods, available resources, experience levels) and projects (e.g., client differences, types of problems). This necessitated a different problem interpretation. The problem shifted from "what problems can be solved in practice?" to "how to make the system functionalities of REConSum and Trace2Conv usable for users in practice?" Solving this problem would enable the use and testing of the systems in real-world scenarios.

To create a use experience, defined as " the experience of someone using a product, system, or service, for example whether they find it enjoyable and easy to use: " (Cambridge University Press & Assessment, 2024), a usable system must be designed for test users. A viable solution must first meet the requirements of availability, low use effort, and scalability.

To illustrate, consider a carpenter and his toolbox. For a tool to be useful, it must be available when needed. Additionally, a carpenter may find alternative uses for a hammer beyond its primary function, by making the solution available during the user's workflow, different use cases may emerge besides the intended use case. Moreover, a carpenter prefers tools that make their life easier, a solution will only be adopted by users if they improve their life in some way. Finally, to adequately test a new tool's effectiveness, a sufficient sample size is necessary. The paragraph about use requirements will go into detail what this means for the system design.

3.6) Use case definition

An initial use case needs to be defined to which the functional prototype is tailored to and that, as mentioned in a previous paragraph, integrates the algorithms of REConSum. When considering that conversations can take up to multiple days and Business Analysts need to juggle multiple mental tasks simultaneously. They must maintain conversational flow, actively listen to participants, and provide appropriate responses (Bano et al., 2019). Concurrently, they're constructing a mental framework of domain goals, rules, and application views to interpret client information (Ferrari et al., 2016). Amidst these mental activities, BAs must also capture requirement-relevant information for later processing. Notetaking serves this purpose, but the constraints of multitasking limit the depth and context captured in notes. Human memory supplements notes but isn't sufficient for comprehensive data retention. Factors such as the duration of the conversation (hours to multiple days), the number of participants (up to 10), and the frequency of conversations can further impact the ability to recall specific details. For instance, longer conversations or those with multiple participants increase mental load and complexity, reducing the ability to memorize all relevant information. Additionally, if multiple conversations occur in quick succession, there may be overlap in memories when recalling specific information. Time pressure to process conversations into requirements can also impair processing ability. Given the imperative for complete and accurate requirements, these challenges underscore the value of tools to aid in conversation processing for BAs. Prompted questions serve as a primary method to elicit requirement-relevant information as it is likely that answers to these questions contain this information. Detecting if a speaker turn contains a question can enable the reduction of the conversational dataset by presenting only the questions of the interview to the user. This approach minimizes the amount of data needing inspection. While some BAs already record conversations for later review, this method is time-consuming, requiring listening to the entire conversation. A more efficient solution is needed. To be valuable, such a system must reduce the amount of conversation data requiring inspection and expedite the search for relevant information.



☑ [0:00:00] spk_0: Good morning, and thank you for joining us today. Plea
☑ [0:00:05] spk_1: Good morning. My name is Alex, and I have been workir
☑ [0:00:10] spk_0: Great to have you here, Alex. Let's begin by discussing tl list of requirements covering various aspects. First, let's t think is essential for securely accessing an account?
☑ [0:00:30] spk_1: As a user, I need a password field to securely access my protect my data.
☑ [0:00:40] spk_0: Excellent. As an administrator, what are your thoughts o As an administrator, I need the ability to enforce passwo passwords that meet certain criteria, such as minimum le special characters.
☑ [0:00:46] spk_1: That's important. Now, let's discuss the option to reset p option to reset a forgotten password?
☑ [0:01:00] spk_0: As a user, I need the option to reset my password if I for hassle. A secure password recovery process, such as ema
☑ [0:01:10] spk_1: Understood. Moving on to system administrators, what i
☑ [0:01:25] spk_0: As a system administrator, I need the ability to store pas passwords from unauthorized access and ensure the hig
☑ [0:01:32] spk_1: Let's shift our focus to connection requirements. As a us application?
☑ [0:01:45] spk_0: As a user, I need a reliable internet connection to access that I can use the application effectively and perform tas
☑ [0:01:55] spk_1: Agreed. Now, as a network administrator, what are your
☑ [0:02:10] spk_0: As a network administrator, I need the application to sup that sensitive information remains encrypted and protec

Figure 17: Usability flow screens examples (Transcript view showin

3. PROBLEM ANALYSIS

RECONSUM | WHAT NEEDS TO BE REALIZED TO USE IT?

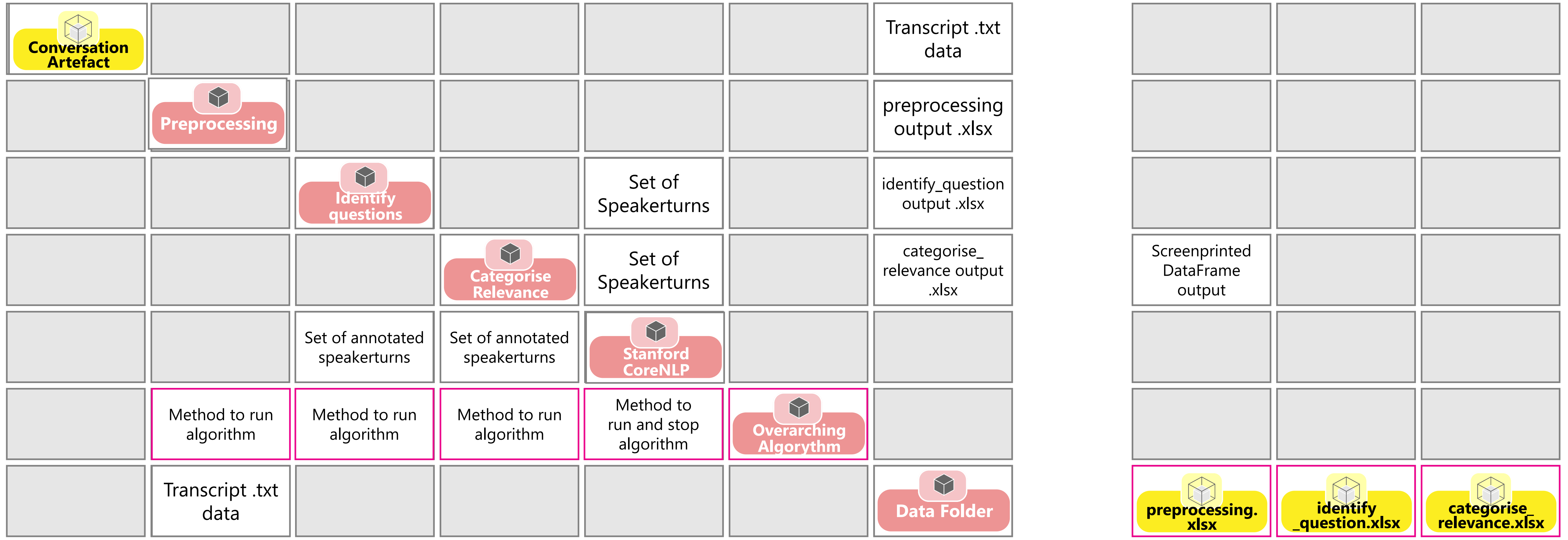
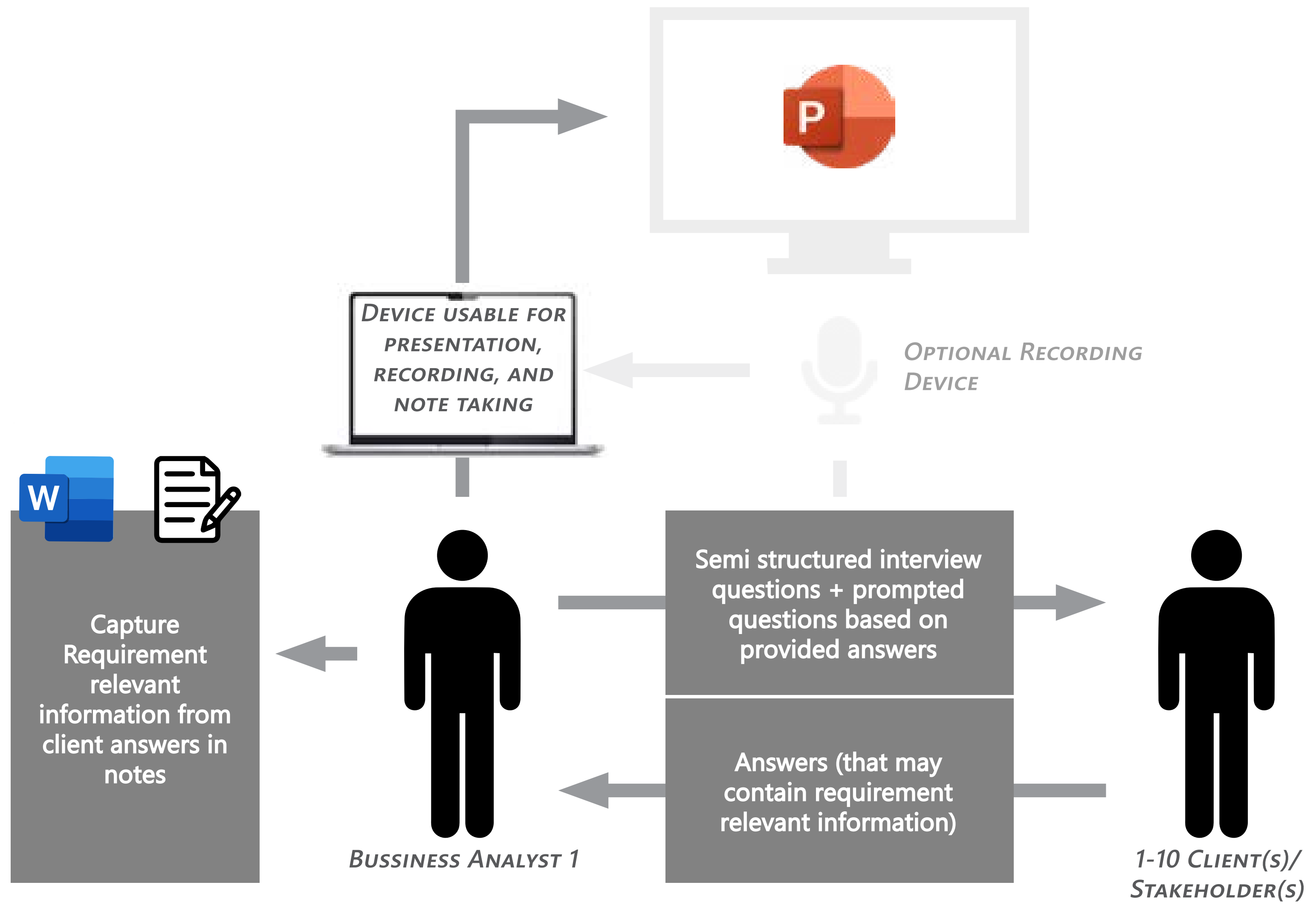


Figure 18: What needs to be realised to use REConSum functionality. The red outlined blocks need to be designed.

3. PROBLEM ANALYSIS

Face to face elicitation setting: one Business Analyst


Device usable for presentation, recording, and note taking
Optional Recording Device
Capture Requirement relevant information from client answers in notes
Semi structured interview questions + prompted questions based on provided answers
Answers (that may contain requirement relevant information)
Bussiness Analyst 1
1-10 Client(s)/ Stakeholder(s)

Face to face elicitation setting: two Business Analysts

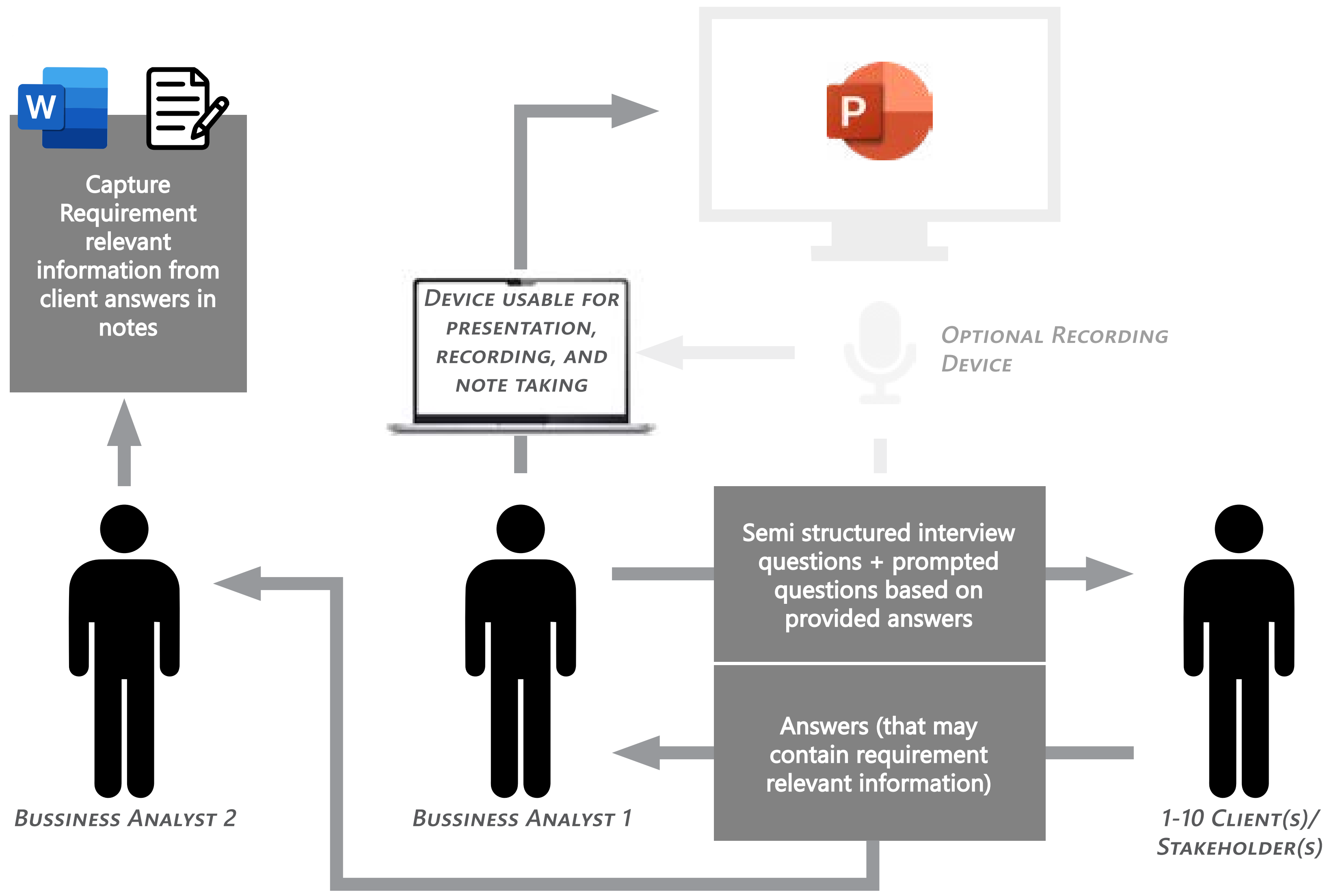

Capture Requirement relevant information from client answers in notes
Device usable for presentation, recording, and note taking
Optional Recording Device
Semi structured interview questions + prompted questions based on provided answers
Answers (that may contain requirement relevant information)
Bussiness Analyst 2
Bussiness Analyst 1
1-10 Client(s)/ Stakeholder(s)

Figure 19: PACT analysis results, activities during elicitation conversations.

3. PROBLEM ANALYSIS

Online elicitation setting

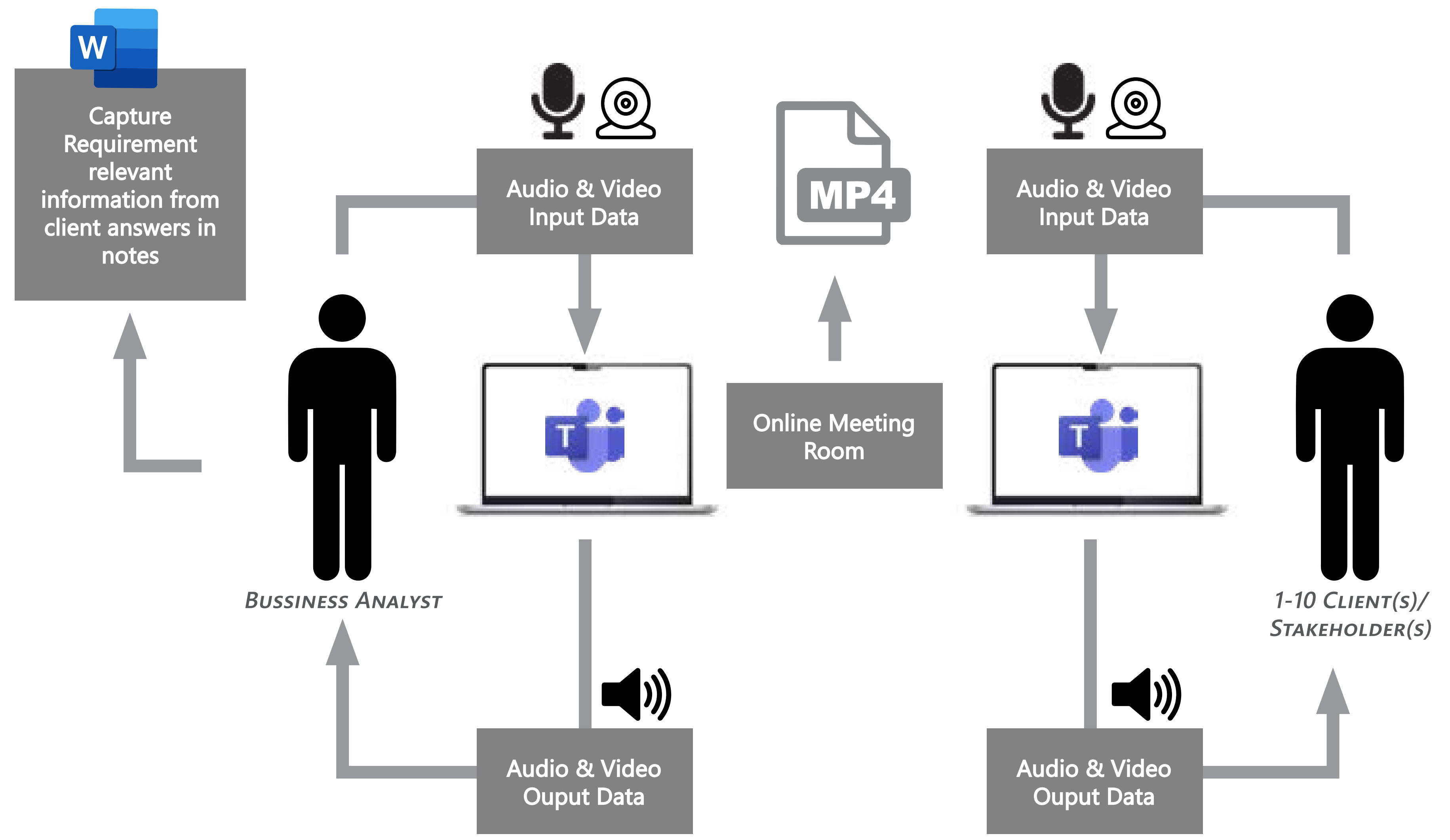
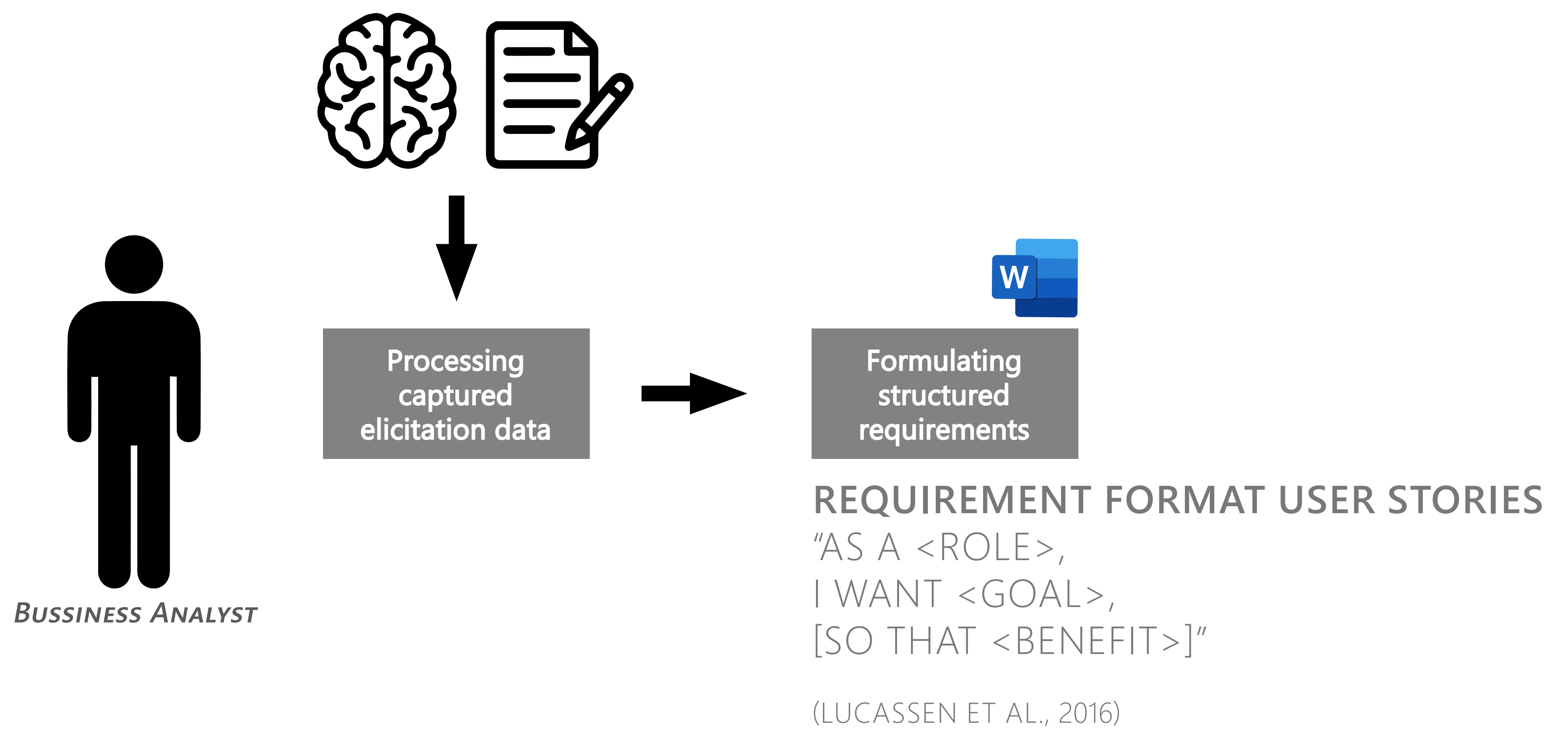


Figure 19: PACT analysis results, activities during elicitation conversations.

Requirement processing



Bussiness Analyst
Processing captured elicitation data
Formulating structured requirements
REQUIREMENT FORMAT USER STORIES
"AS A <ROLE>,
I WANT <GOAL>,
[SO THAT <BENEFIT>]"
(LUCASSEN ET AL., 2016)

Figure 19: PACT analysis results, Processing requirement relevant information.

3. PROBLEM ANALYSIS

within the reduced dataset. A baseline criterion for usefulness is that inspecting the reduced conversational dataset should take less time than listening to the entire conversation. Additionally, the system should enable BAs to locate specific conversation data within one hour, using features like search functions and filters to facilitate navigation.

With the functionalities of REConSum questions and relevant questions are annotated for each speakerturn. As previously noted, NLP systems have limitations in fully comprehending human language. Who better to understand human language than humans themselves? There exists an opportunity to leverage and enhance the capabilities of BAs to recognize and capture relevant data during elicitation conversations. BAs already need to process conversation data in real-time and determine its relevance, essentially annotating for themselves which parts of statements are pertinent. Given the existence of systems that can convert conversations into natural language in real-time, could BAs annotate conversations as they are captured? Notetaking, a common practice among analysts during elicitation conversations, serves as a form of annotated data capture, focusing on the most critical information due to limited capacity for data capture while simultaneously performing other mental activities. Since this data capture activity already exists, what if it were enhanced through design and integrated with the existing system prototype? This integration could facilitate the capture of more perceived relevant information by BAs.

3.7) Defining use requirements

To introduce a system to be used during Conversational RE and Requirement analysis it is important to understand the people present (P), the activities they perform (A), the context in which it is performed (C), and the technologies or products used in these activities (T). A PACT-analysis (Benyon et al., 2005),based on the input from BA interviews and the general workflow based on literature (Project Management Institute, 2016), was used to formulate requirements for systems that are introduced in both activities. Figure 19 presents an overview of the PACT analysis results. BAs will need to use the system during Elicitation Interviews to enable its intended functionality. Understanding this process is crucial. various location settings where the elicitation conversations occur are identified.

At each elicitation there is at least 1 business analyst and 1 client present. However the amount of analysts varies between 1 and 2, and the amount of client(s) or stakeholder(s) can vary between 1 and up to 10. The amount of analysts determines the amount of tasks that need to be performed simultaneously, if 2 analysts are present one will be working on maintaining the conversation while the other is tasked with documenting the requirement relevant information. If an analyst is on their own all these tasks need to be performed by the single analyst.

These locations can range from within the company premises to external locations, such as client sites. BAs should rely only on the systems they bring to the conversation. Thus, the system design must fit within the everyday carry capacity of BAs, fitting into standard laptop bags or suitcases.

During elicitation interviews, BAs typically have multiple systems at hand to aid conversations and capture data. Laptops serve various purposes, including visualization support (e.g., presenting PowerPoint slides or relevant charts) and note-taking using text processors or recording capabilities. Other tools aiding data capture include portable conference speakers, improving recording quality, and traditional tools like notebooks and writing utensils. Online conversations often require laptops with audio and video recording hardware.

Currently, several systems on the market integrate seamlessly with tools already used during elicitation conversations, enhancing their functionalities. These systems include drawing tablets and smart pens that merge writing/documentation with digital notetaking, enabling various computer-based interactions like copy-paste and moving notes. Additionally, conference speakers facilitate integration with conference software, enhancing conversation management with features like microphone controls.

From this analysis the following use requirements are extracted. As stated in throughout the introduction and the problem analysis the Conversational RE process is a mentally demanding activity for Business Analysts with high interests in precision and completeness.

•A system used during the conversation should mostly rely on touch or require minimal interaction to minimize distractions.
•Given the importance of eliciting requirements, system usage should not distract from the conversation.

•Interactions during conversations should be intuitive and require little to no mental processing.

•Additionally, interactions should be limited, considering that one hand may be occupied with writing (with a pen and notebook) or both hands with typing (when using a laptop for notes).

•The possible variation in amount of clients/stakeholders will result in an variable amount of background noise and people talking simultaneously during the conversation, especially in a face to face conversation. The speech to text conversion should perform as desired despite this.

•As conversations can take place at external locations it should have a form-factor that fits within a laptop bag or case.

•The inspection of the system outputs to search for specific conversation data should take less time then listening back to the whole conversation.

To keep the use effort low more requirements are considered.

•The system design must integrate with products and systems already used by the user.

•As elicitation interviews can take place in uncontrolled environments a Business Analyst should only have to rely on systems or products, they bring themselves to the interview.

•The effectiveness and efficiency of tasks performed should never decrease.

•No prior knowledge of for example python needs to be required to use the system.

•Therefore, a solution must not require installation or setup by the user (if so no more then 10 minutes and only require users to press 'next') and should not significantly disrupt workflow (Use of the system should not add more than approximately half an hour of time).

•The system should not require more than 10 minutes of installing.

•Conversation Artefacts generated with Microsoft Teams can be directly imported into the system prototype without the user needed to make any changes to the contents of the generated artefact.

3.8) Defining functional requirements

Database requirements

•All inputs and outputs need to be stored in a database, stored per conversation.

•The database should include everything needed to run the system (code/systems/assets)

•The TF-Wiki dataset should be accessible by the system in the database to make the REConSum functionalities work.

•To enable the annotation and processing of the speakerturns, the StanfordCoreNLP system needs to be stored within the system database. StanfordCoreNLP needs to be accessible by the system in the database to start when it is needed and stop automatically when the necessary processing is finished.

•System outputs stored in the system database need to be accessible by the user through the system GUI.

•As stated in the system analysis NLP has for multiple reasons sometimes troubles with understanding human language (Natural Language) correctly. To validate the functionality of the NLP outputs, outputs will need to be systemically stored making it easy for researchers to inspect produced outputs and allows the GUI to locate conversations at a consistent location.

Overall system requirements

•The system prototype should easily fit into a laptop bag or case and must be usable and accessible at any time.

•The system should run on any (windows) system. To validate the functionality of the system and support research in its use, it's important to use a datatype that can be easily inspected by researchers.

GUI requirements

•A GUI needs to be realized where a BA is allowed to import a generated conversation artefact.

•A user must be able to view the set of speakerturns for each imported conversation.

•For each speakerturn in a set of speakerturns the time index, the person speaking, and the spoken text need to be clearly communicated to the user.

•The BA needs to be able to filter a conversation to view only the questions.

•The BA needs to be able to filter a conversation to view only the relevant questions.

•For each question the user should be able to view the answer as answers are likely to contain requirement relevant information.

Navigation requirements
•The BA needs to be able to navigate between conversations and to be able to easily locate a desired conversations from the database.

•At all times the current state of the system (current conversation, current filter, etc) needs to be communicated to the user.

Implementation requirements
To implement the prototype into practice a couple of implementation requirements need to be met.
•Outputs are systematically stored to be analysed at a later point.

•The system prototype can be easily duplicated to supply 10-20 people with the prototype.

•Conversation Artefacts are captured without any speech to text conversion errors in requirement relevant information.

Note that these requirements contain performance factors as for example 'easy' and 'clearly'. In this phase of development it is hard to specifically define measurable performance factors. Based on user feedback, gathered from using the system prototype, requirements can be further specified, added, or changed.

3.9) Design Brief
This thesis will provide a prototype that has the potential to be useful for Business Analysts in the activity of processing captured elicitation data into structured requirements. While it may not always be of use, in longer real-life conversations the ability to filter conversations to search for specific conversation data has the potential to be valuable for a Business Analyst (Spijkman et al., 2023). The main functionality on system level will be to capture all conversation data generated during the elicitation interview (so no conversation data is lost) and allowing a Business Analyst to locate requirement relevant information in this conversation dataset with filters. The prototype will allow users to import conversations and to view the conversation using a designed system GUI. Processing and storing user inputs. The conversation can be filtered using the annotations and for each question it will be made possible to view the answer to the question.

Also, as humans are better in understanding human language then systems, and the goal is to capture more requirement relevant information while reducing to listening back to a full conversation, a proof of concept will be realized to enable Business Analysts to annotate conversations themselves in real time during the conversation. This interaction will be integrated with notetaking, an activity already performed where conversation data is annotated. The goal is not impeding the notetaking process with the inclusion of such a prototype.

To make this possible Business Analysts need to be able to capture elicitation conversations with high accuracy in natural language. Through test conversations the performance of the MS Teams speech to text functionality will be tested and a method will be formulated how a conversation can be captured and formatted to be used by the system prototype in a online, and face to face setting.

The prototype and this method will be validated where the performance is evaluated to the specified criteria mentioned above.
An implementation plan has been included how this prototype can be implemented for further research and development also in regard to the results of the validation phase.

The next chapter will present the prototype that resulted from the design process. A manual has been included in the appendix chapter 0 as deliverable for the client explaining in steps how the whole system is built and behaves.

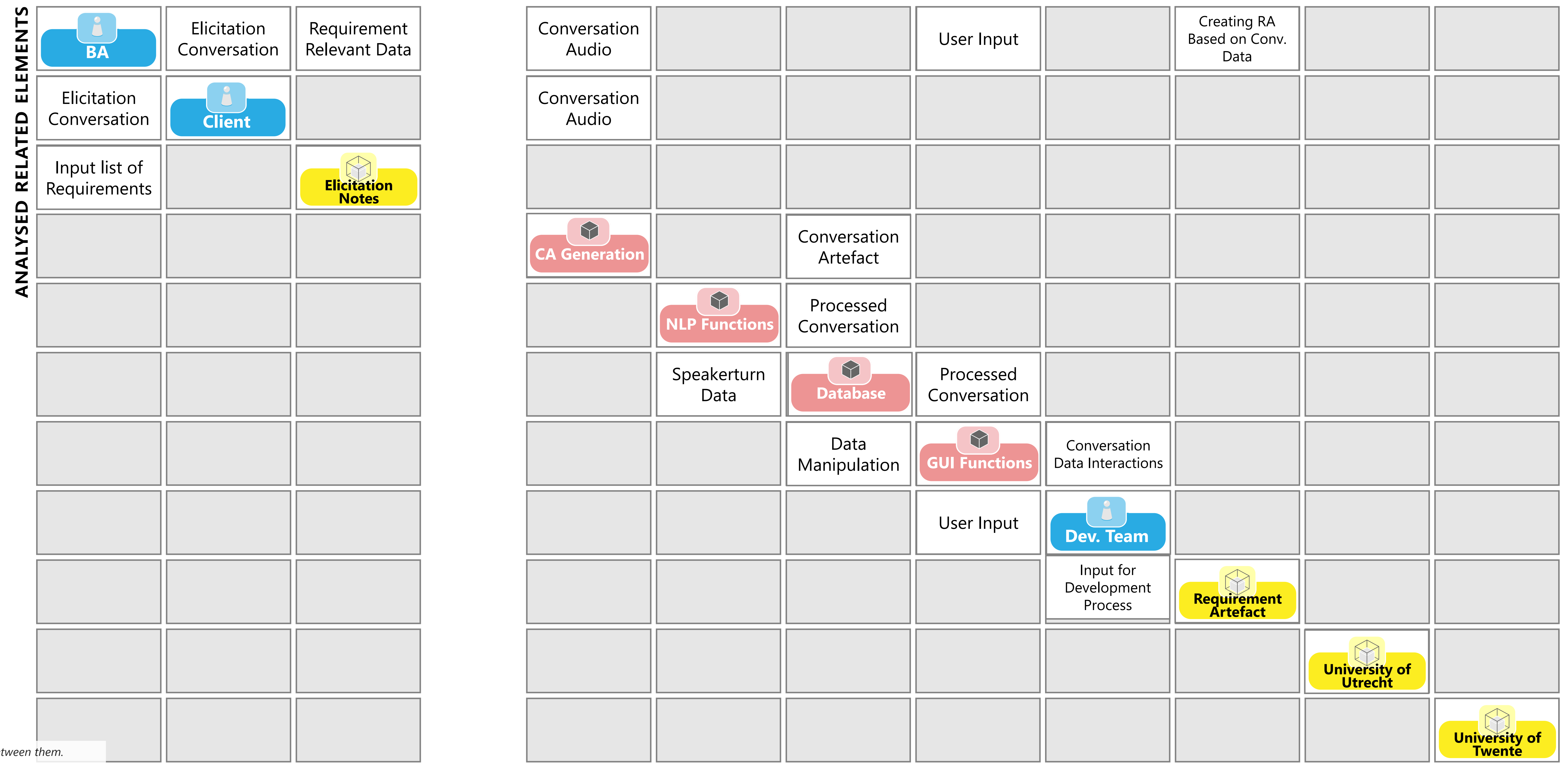


Figure 20: Analysis results: The humans, environment and systems that should be considered in the system design and the interactions between them.

4. REALIZED SYSTEM

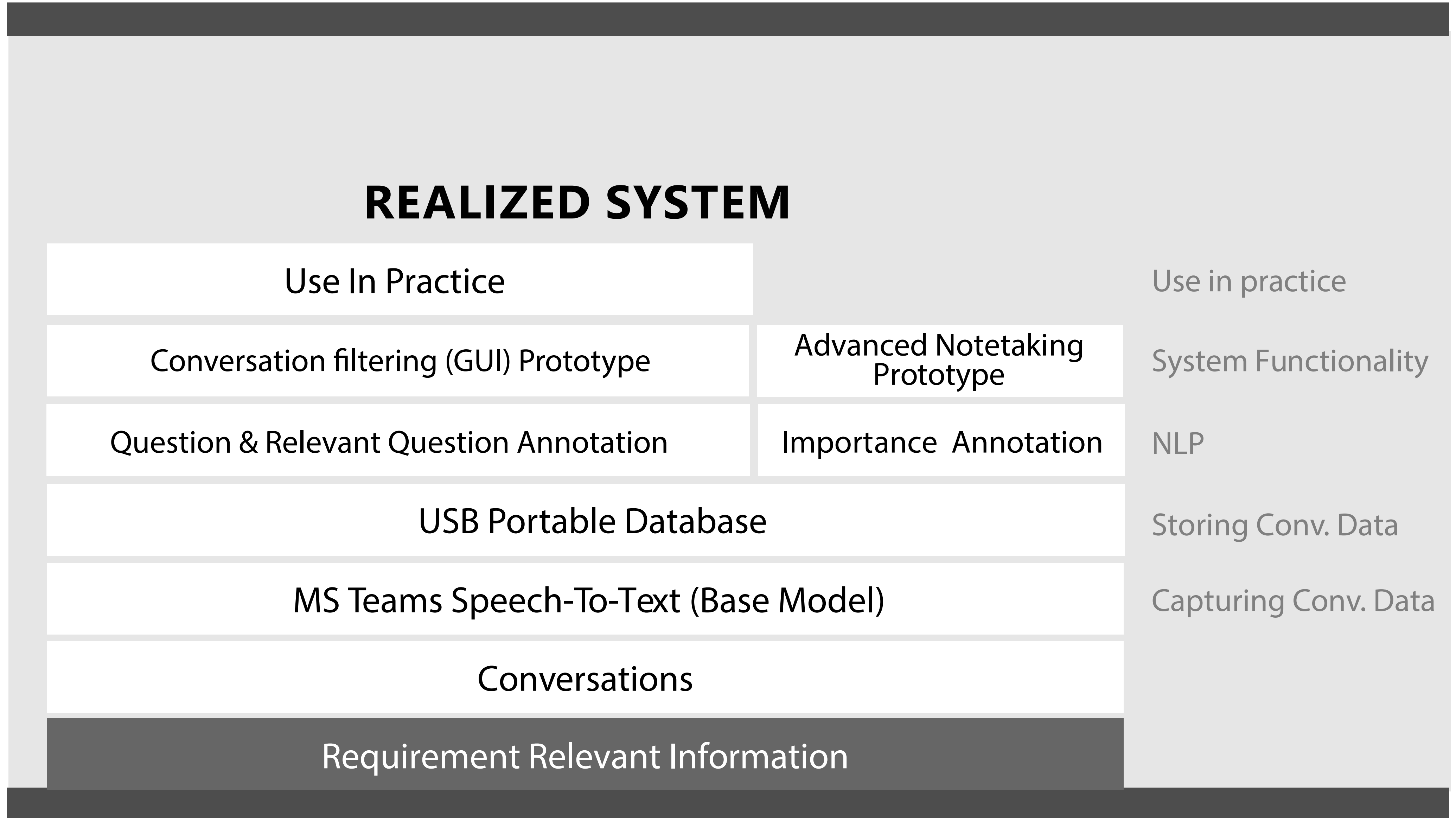
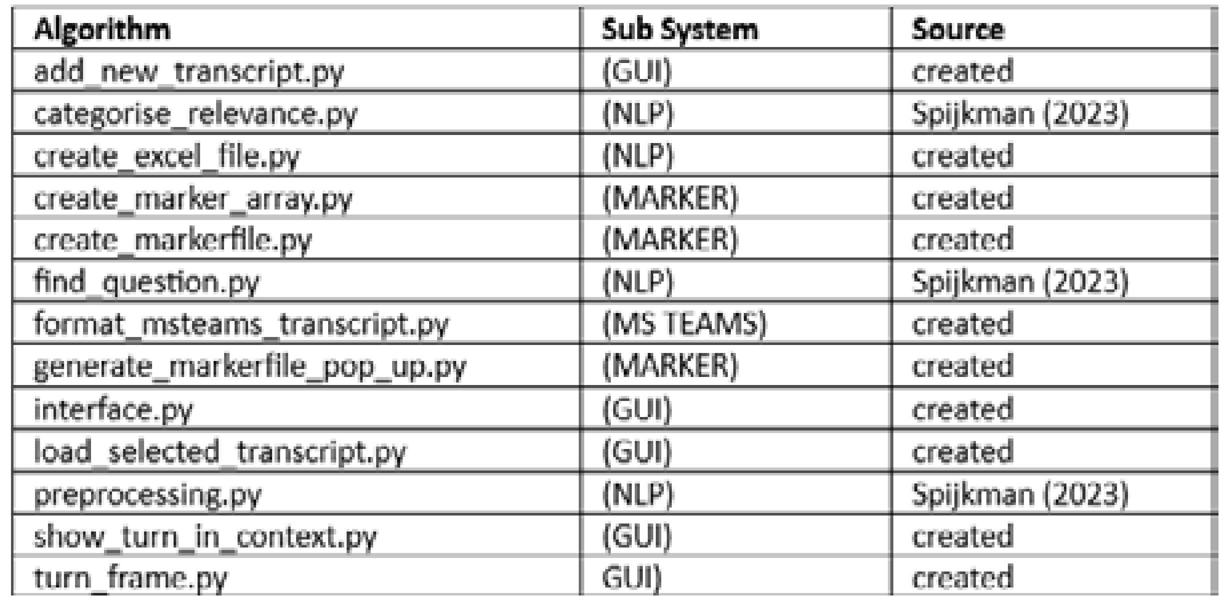

REALIZED SYSTEM
Use In Practice
Use in practice
Conversation filtering (GUI) Prototype
Advanced Notetaking Prototype
System Functionality
Question & Relevant Question Annotation
Importance Annotation
NLP
USB Portable Database
Storing Conv. Data
MS Teams Speech-To-Text (Base Model)
Capturing Conv. Data
Conversations
Requirement Relevant Information

Figure 21: Realized System, the realized solution on each 'foundational layer'.



Algorithm	Sub System	Source
add_new_transcript.py	(GUI)	created
categorise_relevance.py	(NLP)	Spijkman (2023)
create_excel_file.py	(NLP)	created
create_marker_array.py	(MARKER)	created
create_markerfile.py	(MARKER)	created
find_question.py	(NLP)	Spijkman (2023)
format_msteams_transcript.py	(MS TEAMS)	created
generate_markerfile_pop_up.py	(MARKER)	created
interface.py	(GUI)	created
load_selected_transcript.py	(GUI)	created
preprocessing.py	(NLP)	Spijkman (2023)
show_turn_in_context.py	(GUI)	created
turn_frame.py	GUI)	created

Table 1: Overview of all the algorithms that are present within the system.

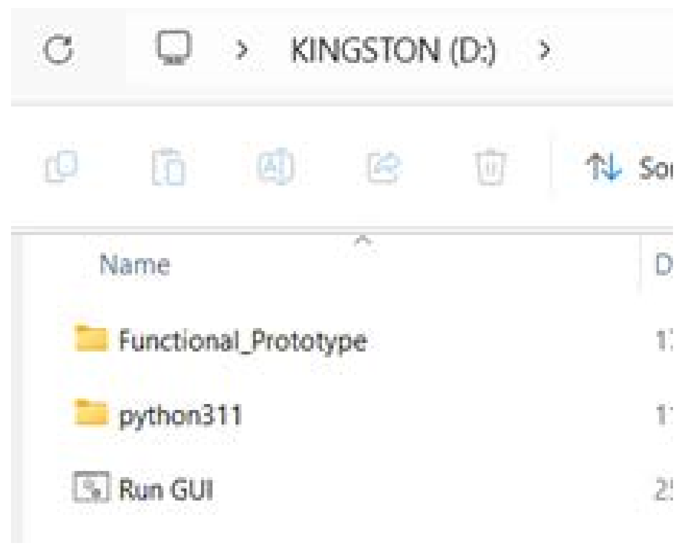
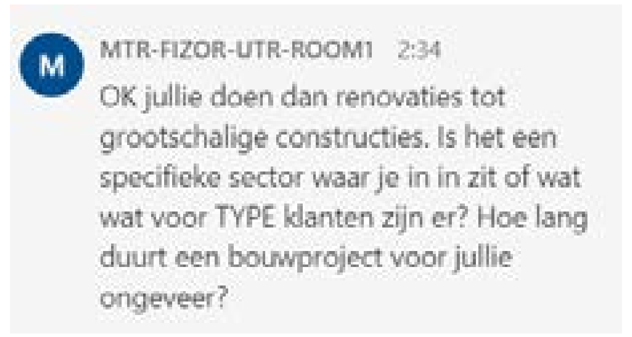


Figure 24: Main folder of the system database containing the functional prototype, python to run the prototype and a bootloader called 'RUN GUI.bat' used to start the system GUI.



MTR-FIZOR-UTR-ROOM1 2:34

OK jullie doen dan renovaties tot grootschalige constructies. Is het een specifieke sector waar je in in zit of wat wat voor TYPE klanten zijn er? Hoe lang duurt een bouwproject voor jullie ongeveer?

Figure 22: Example speakerturn from test conversation generated with MS Teams.

Meeting with Jeroen van Kekem-20240112_

Meeting with Jeroen van Kekem-20240112_

Meeting with Jeroen van Kekem-20240112_

Figure 23: Conversations stored in the MS Stream environment.

Figure 25: Home screen of GUI design.

4. REALIZED SYSTEM

This chapter provides an in-depth exploration of the system prototype developed to address the identified problem. Serving as an initial stride in the development journey, this prototype aims to generate conversation artefacts from natural language, process them, and store them in a central database for utilization across the agile software development process. The primary objective of this prototype is to furnish a functional model capable of practical application, facilitating user feedback collection to enhance various facets of the system. Figure 21 presents the realized system in the context of the foundational layers that make up the desired system.

The prototype that has been realized is a system that consists of multiple subsystems or functionalities, each with their own behaviour and components, that together achieve the main functionality of the system. To illustrate this table 1 presents an overview of all the algorithms that have been created or integrated to realize the system. The main functionality on system level is to capture all conversation data generated during the elicitation interview (so no conversation data is lost) and allowing a Business Analyst to locate requirement relevant information in this conversation dataset through the use of filters. The system can be split up into 4 subsystem functionalities labeled in this thesis as GUI (allowing the user to interact with the system), NLP (automating and tying all algorithms together to process and annotate conversations to store them into the conversation database), Data storage (a usb drive storing all the necessary components to make the system run), and finally a method for users to annotate conversations themselves during the elicitation interview themselves. Appendix chapter 0 will cover each part of the system in detail explaining the behaviour of the system and how it all ties together so it can be replicated by Fizor.
This system is built and tailored to use with Conversation Artefacts that are generated with MS Teams. However as mentioned earlier, for the system to be usable in practice this Conversation Artefact needs to be of high accuracy (no requirement relevant information translated incorrectly). Two main challenges were uncovered during test conversations to test the performance of using MS Teams to generate CAs being the correct formatting of speakerturns (matching the captured audio to the correct person and time), and accurate conversion of speech to text by MS Teams. In the use flow that will be presented online conversation will be used as the elicitation setting as this requires as-is no steps of the user.
First the realized use flow will be covered from conversation to use of the processed conversation data by the Business Analyst. After that the separate systems will be covered how they make this functionality possible. Note that the marker functionality is not integrated in this flow as use in practice has not been realized, it will be covered later in the chapter.

4.1) Use Flow

For the new project that has started a Requirement Elicitation conversation takes place between the Business Analyst and a new client. Because it is for both parties more convenient for the conversation to take place online a MS Teams meeting has been scheduled.

At the start of the meeting the Business Analyst activates the record and transcribe functionality and selects the conversation language (only English and Dutch function correctly because of how the system is programmed)

In the background MS Teams generates a CA converting speech to text and linking it to the person speaking and the time of speaking (set of speakerturns, figure 22).

At the end of the conversation the generated CA is stored in MS Stream (cloud storage of MS Teams), as can be seen in figure 23.

The Business Analyst searches the desired conversation and downloads it on a (windows) laptop or pc.

The Business Analyst takes the system prototype ,a usb drive, out of his bag and selects the GUI bootloader located in the main folder (figure 24) resulting in the GUI opening (figure 25).

In the GUI the user can import a conversation by selecting the desired conversation (figure 26) and provide the conversation with a name (so it can be recognized later in the conversation database), and (as MS Teams may included for example email addresses in the speaker names) the ability is given to change the names of the speakers (The prototype only works for now with 2 speakers but can be easily increased).

In about 15 minutes (not requiring the user to be there in person) the system has formatted, processed, and annotated the conversation and stored the outputs in the conversation database (note that the processing functionality requires the installation of java, necessary to run part of the processing functionality and takes approximately 5 minutes only needing to press 'next'. The rest of the system can directly be used without any installing).

As users will need to process multiple conversations, conversations are systematically stored and can be accessed through the dropdown menu in the GUI. From here the user can select a desired conversation to load into the GUI.

To search for specific conversation data the amount of conversation data can be reduced through the use of the different filters. The user has the ability to view the whole conversation (through full view, figure 28), all questions of the conversation (through question view, figure 29), all relevant questions (through summary view, figure 30). Using the 'show in context' button (figure 31) the user can navigate to answers to specific questions (as it can show up to 10 speakerturns before and after the selected question) without the need to read through the whole conversation.

4.2) NLP & Database storage

The two main functionalities besides the GUI functionalities that make up the system are importing conversations and loading processed conversations as described in the use flow.

For the importing of conversations

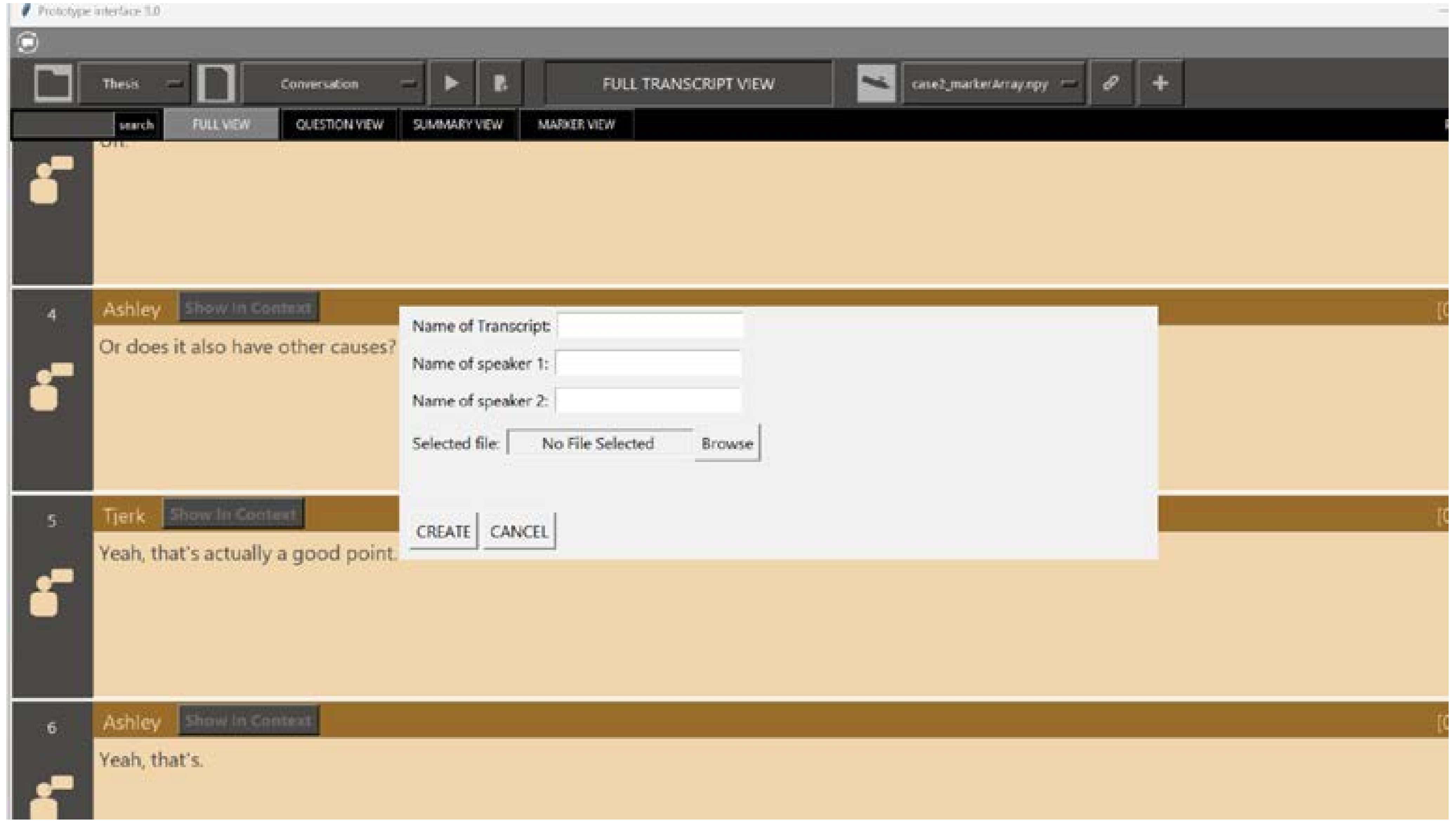
To structure the storage of outputs, and providing them with consistent file names and locations so they may be located by the system the first step when a conversation is imported a database entry is created (figure 27, step 1). In the selected project folder, a conversation folder is created and named with the name supplied by the user. In this conversation folder outputs are systematically stored in separate folders using consistent names. MS Teams-generated conversation artefacts contain visual elements and text formatting that do not conform to the system's expected formatting structure. To format the artefact correctly without any user involvement an algorithm automates this process (figure 27, step 2). The algorithm operates in two stages: First, it removes all visual elements from the conversation artefact, converts it to a text file format, and stores it in the database as raw_transcript. This initial step ensures that the raw conversation data is stripped of any unnecessary formatting (chapter 10 of the appendix). Next, the algorithm formats the speakerturns according to the system's requirements, preparing them for further processing. This formatted version of the transcript is then stored in the system as formatted_transcript (chapter 10 of the appendix). Note that the algorithm is programmed to how MS Teams currently formats their artefacts as of this moment. If MS Teams would change this, if other software is used, or if languages other than English and Dutch is used the algorithm needs to be adapted.

As it is important that the conversation is processed correctly it was important to validate that the system processed the conversation correctly throughout the processing steps. It be able to inspect the processing output at each stage the output is stored in the conversation database as an excel file. Storing the outputs as an excel file has the added benefit of being easy to inspect as data is stored in a matrix (figure 33) but it also enables for quantitative analysis of processed conversation by researchers if a large volume of conversations has been processed by users. In figure 27, step 3 the first excel file is created of the preprocessed DataFrame.
StanfordCoreNLP, the system that enables the processing and annotating of speakerturns for REConSum has been integrated into the system removing the need for users to manually install and run it manually. The system is correctly started and stopped (figure 27, step 4 and 7) at the right time automating this process. The integration of this system also enables the opportunity to easily add more NLP functionalities such as tokenization, to enable Trace2Conv functionalities.
The algorithms responsible for identifying questions and relevant questions have been left unchanged besides the fact that they store their outputs in the database for validation. They are run at the correct time (figure 27, step 5 and 6) and the Wiki-TF dataset is included in the system database. With the user in mind python is installed on the usb drive with the correct packages needed to run the system. The prototype takes no more then 3gb of space and the processed conversations can be inspected without any installment needed.

Loading conversations

To present the speakerturns to the user correctly an algorithm has been designed called turn_frame.py that creates for each speakerturn a 'turn frame' that presents the index, speaker, and text of a speakerturn to the user. the algorithm turn_frame.py has been designed to present time, speaker, and text data to the user. If another conversation is loaded, the existing data is first cleared before loading the currently selected

4. REALIZED SYSTEM

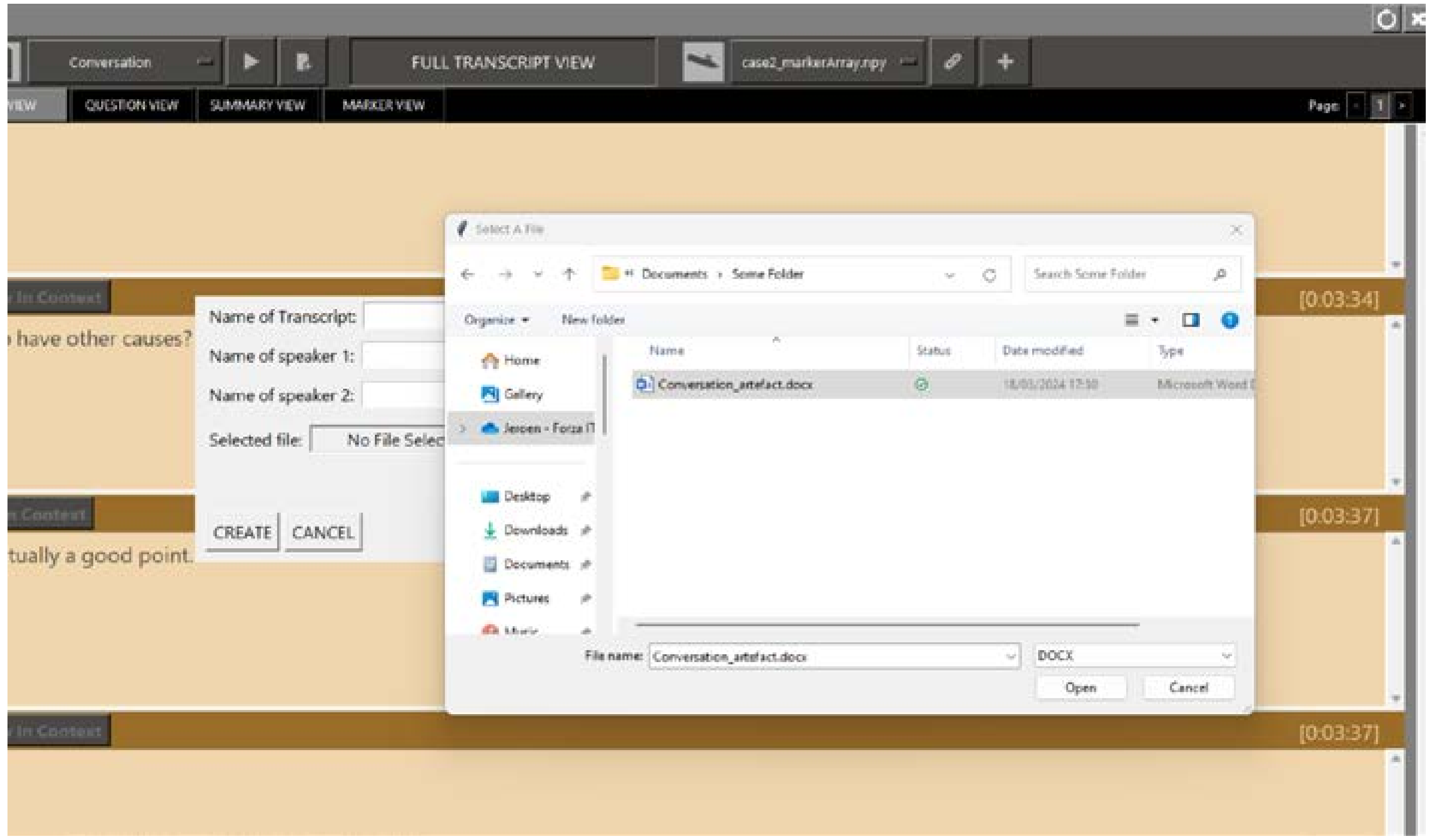


4 Ashley
Or does it also have other causes?

5 Tjerk
Yeah, that's actually a good point.

6 Ashley
Yeah, that's.

Name of Transcript:
Name of speaker 1:
Name of speaker 2:
Selected file: No File Selected



[0:03:34] have other causes?

[0:03:37] tually a good point.

[0:03:37]

Name of Transcript:
Name of speaker 1:
Name of speaker 2:
Selected file: No File Selec

Conversation_artefact.docx 18/03/2024 17:30 Microsoft Word D

File name: Conversation_artefact.docx DOCX

Figure 26: Within a couple of clicks the user can locate and import conversations from the Stream database.

ALL NLP FUNCTIONALITIES LINKED WITH EACH OTHER AND THE DATABASE

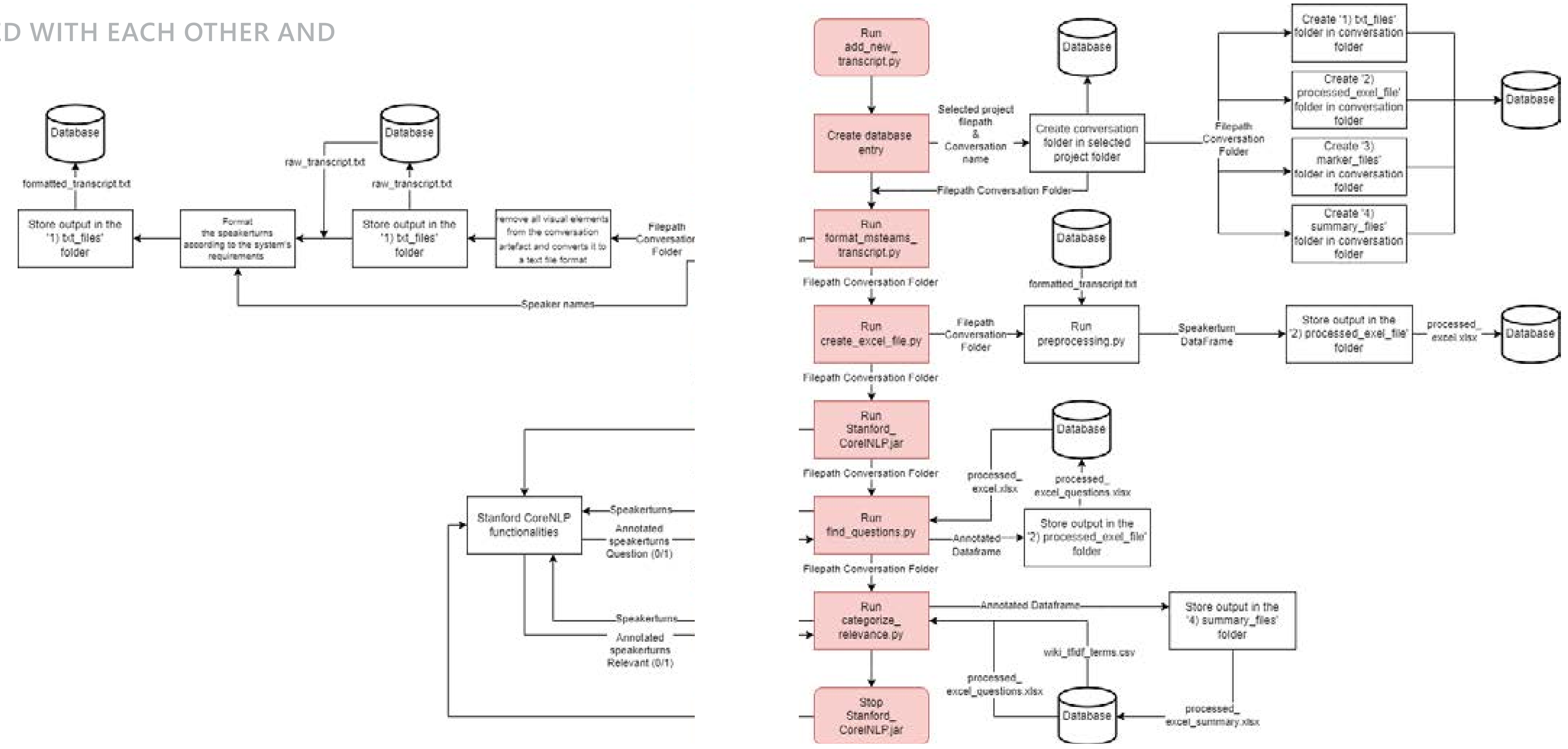
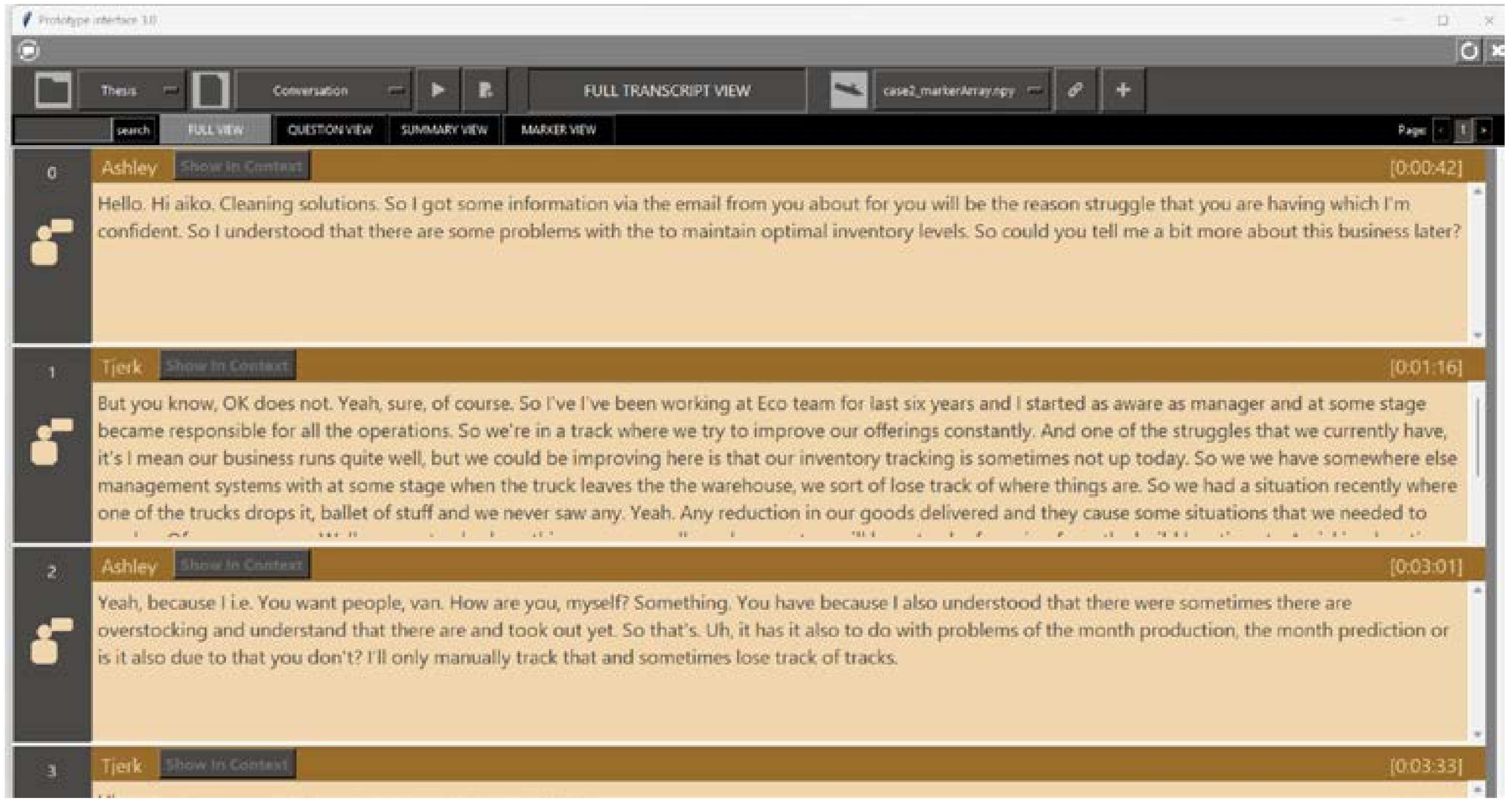

Database
formatted_transcript.txt
Store output in the '1) txt_files' folder
Format the speakerturns according to the system's requirements
Database
raw_transcript.txt
raw_transcript.txt
Store output in the '1) txt_files' folder
remove all visual elements from the conversation artefact and converts it to a text file format
Filepath Conversation Folder
Speaker names
Stanford CoreNLP functionalities
Speakerturns
Annotated speakerturns Question (0/1)
Speakerturns
Annotated speakerturns Relevant (0/1)
Run add_new_transcript.py
Create database entry
Selected project filepath & Conversation name
Create conversation folder in selected project folder
Database
Filepath Conversation Folder
Filepath Conversation Folder
Create '1) txt_files' folder in conversation folder
Create '2) processed_exel_file' folder in conversation folder
Create '3) marker_files' folder in conversation folder
Create '4) summary_files' folder in conversation folder
Database
Run format_msteams_transcript.py
Database
formatted_transcript.txt
Filepath Conversation Folder
Run create_excel_file.py
Filepath Conversation Folder
Run preprocessing.py
Speakerturn DataFrame
Store output in the '2) processed_exel_file' folder
processed_excel.xlsx
Database
Filepath Conversation Folder
Run Stanford_CoreNLP.jar
Database
Filepath Conversation Folder
processed_excel.xlsx
processed_excel_questions.xlsx
Run find_questions.py
Annotated Dataframe
Store output in the '2) processed_exel_file' folder
Filepath Conversation Folder
Run categorize_relevance.py
Annotated Dataframe
Store output in the '4) summary_files' folder
wiki_tfidf_terms.csv
processed_excel_questions.xlsx
Database
processed_excel_summary.xlsx
Stop Stanford_CoreNLP.jar

Figure 27 Designed system functionalities



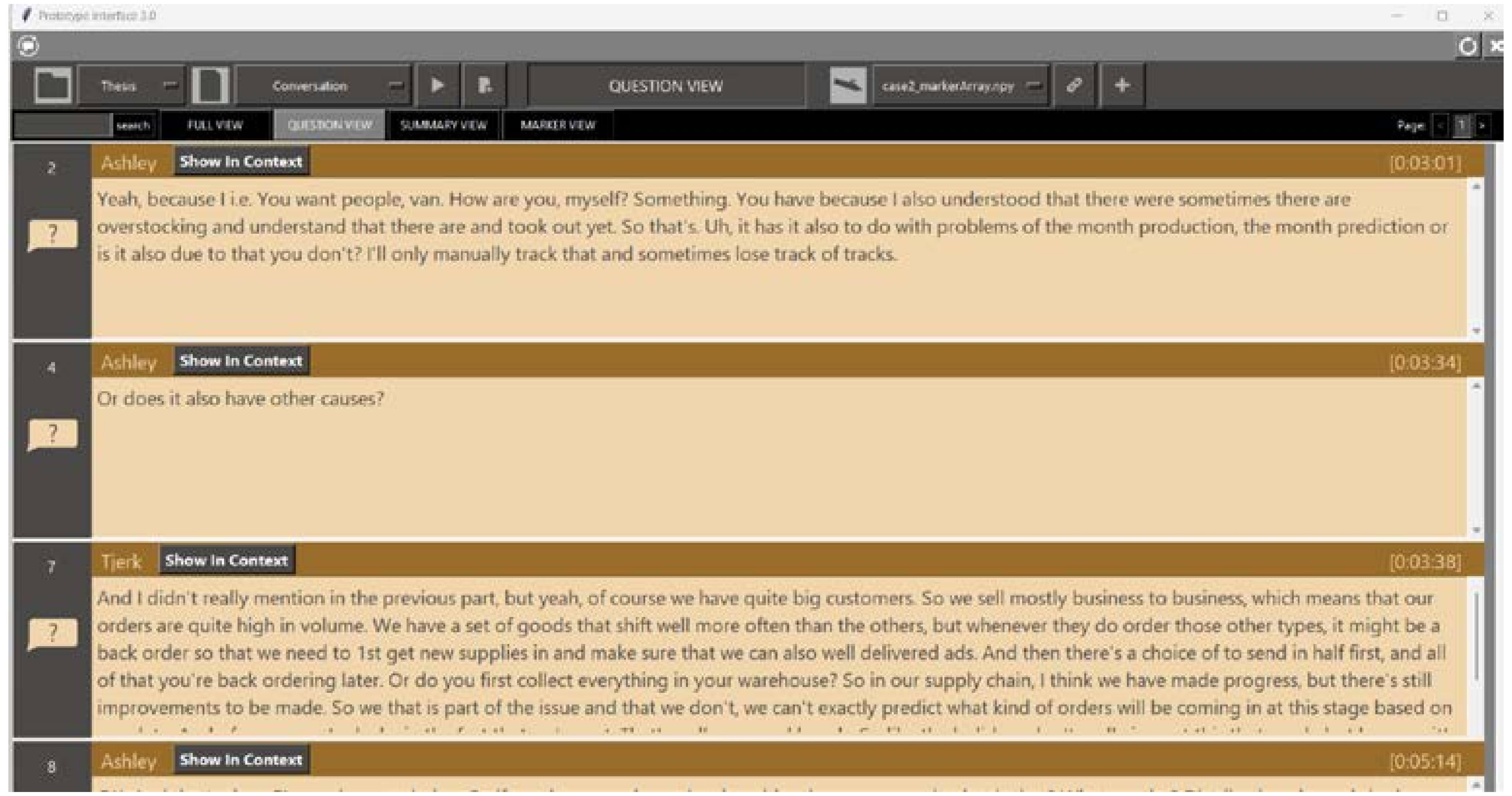
0 Ashley [0:00:42]
Hello. Hi aiko. Cleaning solutions. So I got some information via the email from you about for you will be the reason struggle that you are having which I'm confident. So I understood that there are some problems with the to maintain optimal inventory levels. So could you tell me a bit more about this business later?

1 Tjerk [0:01:16]
But you know, OK does not. Yeah, sure, of course. So I've I've been working at Eco team for last six years and I started as aware as manager and at some stage became responsible for all the operations. So we're in a track where we try to improve our offerings constantly. And one of the struggles that we currently have, it's I mean our business runs quite well, but we could be improving here is that our inventory tracking is sometimes not up today. So we we have somewhere else management systems with at some stage when the truck leaves the the warehouse, we sort of lose track of where things are. So we had a situation recently where one of the trucks drops it, ballet of stuff and we never saw any. Yeah. Any reduction in our goods delivered and they cause some situations that we needed to

2 Ashley [0:03:01]
Yeah, because I i.e. You want people, van. How are you, myself? Something. You have because I also understood that there were sometimes there are overstocking and understand that there are and took out yet. So that's. Uh, it has it also to do with problems of the month production, the month prediction or is it also due to that you don't? I'll only manually track that and sometimes lose track of tracks.

3 Tjerk [0:03:33]

Figure 28: All speakerturns of the conversation.



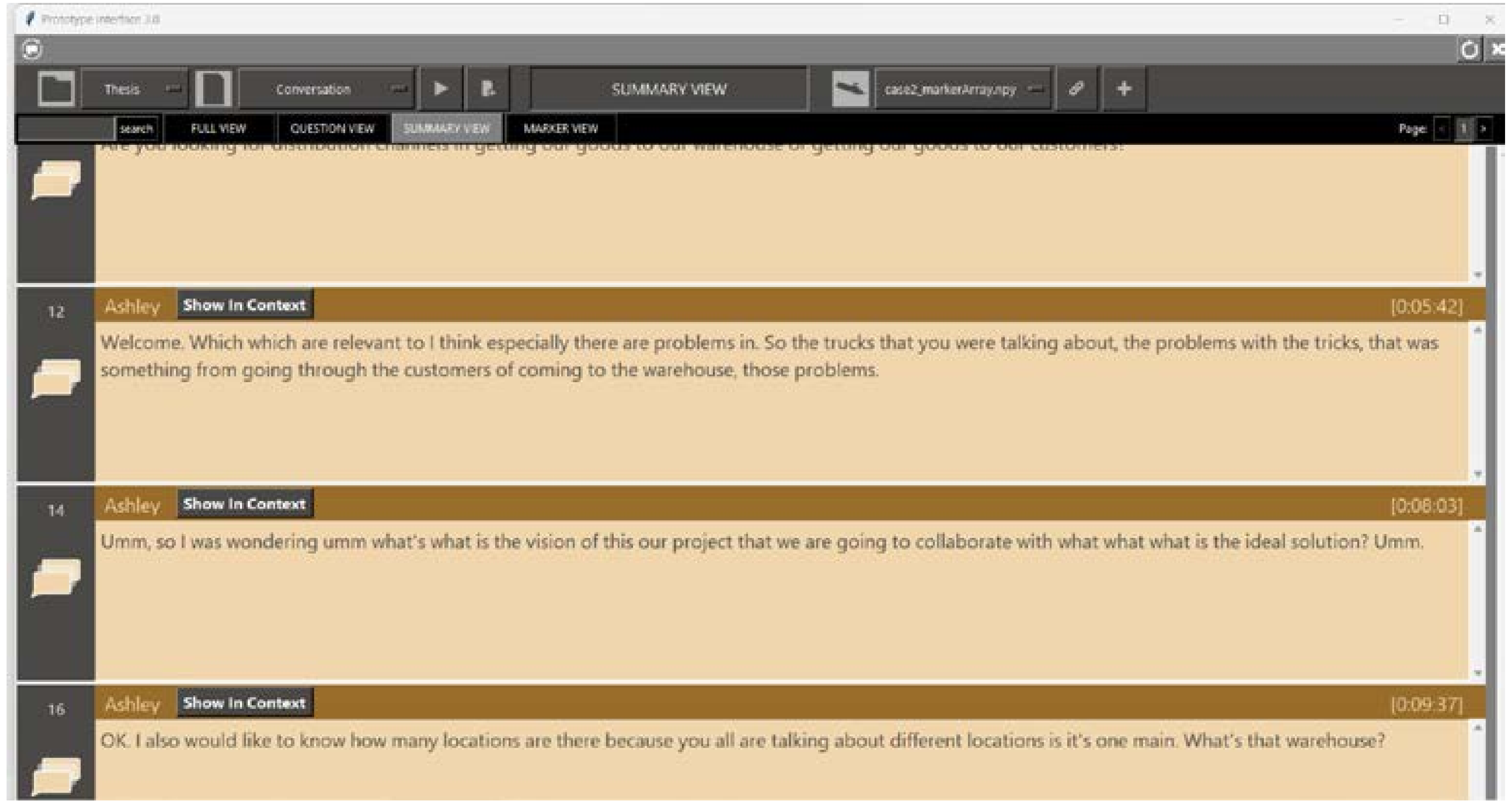
2 Ashley [0:03:01]
Yeah, because I i.e. You want people, van. How are you, myself? Something. You have because I also understood that there were sometimes there are overstocking and understand that there are and took out yet. So that's. Uh, it has it also to do with problems of the month production, the month prediction or is it also due to that you don't? I'll only manually track that and sometimes lose track of tracks.

4 Ashley [0:03:34]
Or does it also have other causes?

7 Tjerk [0:03:38]
And I didn't really mention in the previous part, but yeah, of course we have quite big customers. So we sell mostly business to business, which means that our orders are quite high in volume. We have a set of goods that shift well more often than the others, but whenever they do order those other types, it might be a back order so that we need to 1st get new supplies in and make sure that we can also well delivered ads. And then there's a choice of to send in half first, and all of that you're back ordering later. Or do you first collect everything in your warehouse? So in our supply chain, I think we have made progress, but there's still improvements to be made. So we that is part of the issue and that we don't, we can't exactly predict what kind of orders will be coming in at this stage based on

8 Ashley [0:05:14]

Figure 29: Question view. The conversation filtered to show only the questions of the conversation.



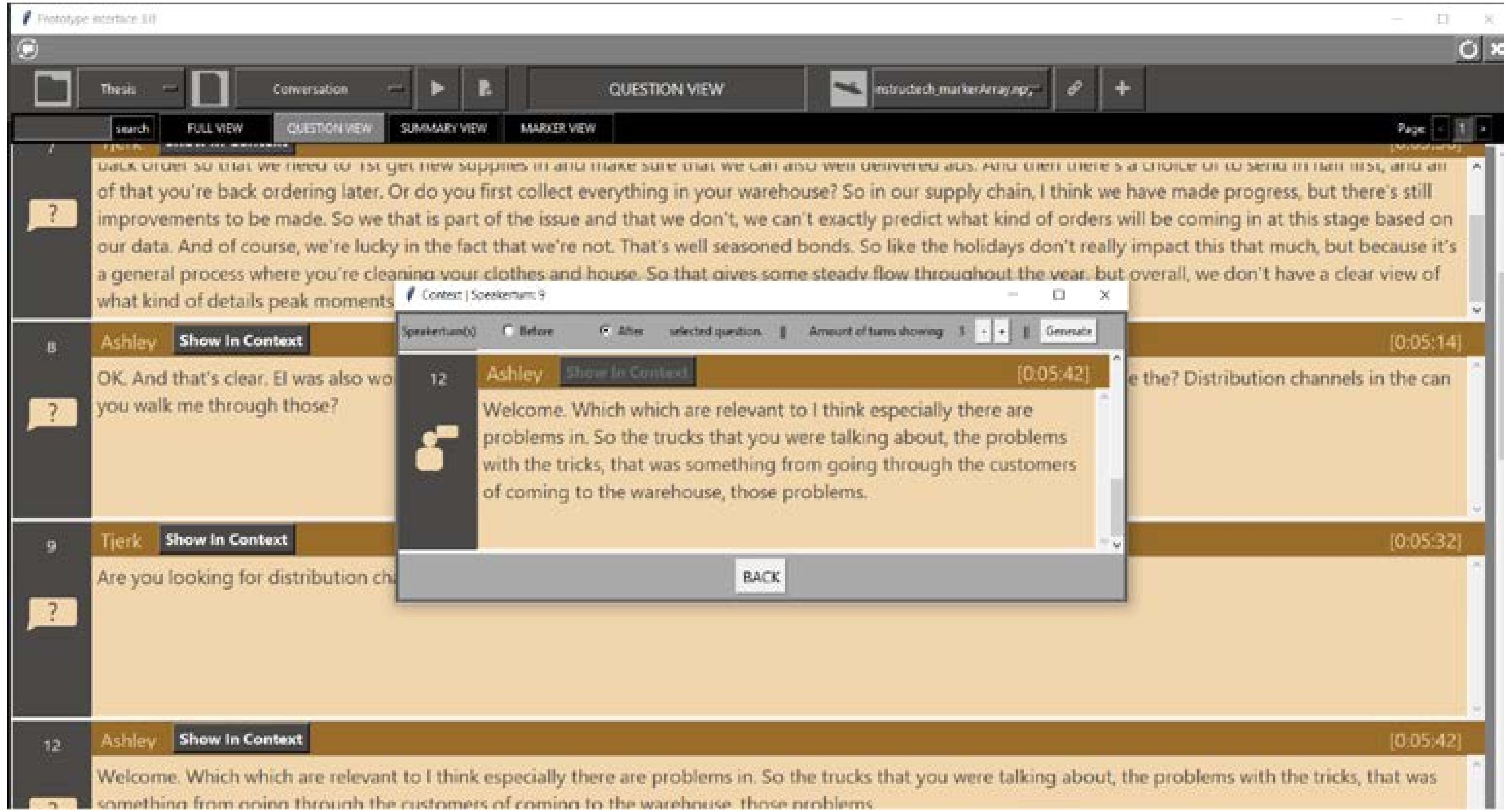
Are you looking for distribution channels in getting our goods to our warehouse or getting our goods to our customers?

12 Ashley Show In Context [0:05:42]
Welcome. Which which are relevant to I think especially there are problems in. So the trucks that you were talking about, the problems with the tricks, that was something from going through the customers of coming to the warehouse, those problems.

14 Ashley Show In Context [0:08:03]
Umm, so I was wondering umm what's what is the vision of this our project that we are going to collaborate with what what what is the ideal solution? Umm.

16 Ashley Show In Context [0:09:37]
OK. I also would like to know how many locations are there because you all are talking about different locations is it's one main. What's that warehouse?

Figure 30: Summary view. The conversation is filtered to show only questions relevant to the conversation.



back order so that we need to 1st get new supplies in and make sure that we can also well delivered aus. And then there's a choice of to send in half first, and all of that you're back ordering later. Or do you first collect everything in your warehouse? So in our supply chain, I think we have made progress, but there's still improvements to be made. So we that is part of the issue and that we don't, we can't exactly predict what kind of orders will be coming in at this stage based on our data. And of course, we're lucky in the fact that we're not. That's well seasoned bonds. So like the holidays don't really impact this that much, but because it's a general process where you're cleaning your clothes and house. So that gives some steady flow throughout the year, but overall, we don't have a clear view of what kind of details peak moments

8 Ashley Show In Context [0:05:14]
OK. And that's clear. El was also wo … e the? Distribution channels in the can you walk me through those?

9 Tjerk Show In Context [0:05:32]
Are you looking for distribution ch

12 Ashley Show In Context [0:05:42]
Welcome. Which which are relevant to I think especially there are problems in. So the trucks that you were talking about, the problems with the tricks, that was something from going through the customers of coming to the warehouse, those problems.

Figure 31: Speakerturn presented in the context of the conversation.

conversation. Please note that these features are included with the intent to provide a positive user experience of navigating the conversation to filter the speakerturn dataset to search for specific information.

To address the likelihood of requirement-relevant information being contained in answers to prompted interview questions, each speakerturn includes a "Show in Context" button (disabled in the Full View). Clicking this button opens a pop-up where the user can view up to 10 speakerturns before or after the selected speakerturn. By default, the first speakerturn after the selected one is presented to the user, as there's a high probability that it contains the answer to the question. This feature is enabled in all views except the Full View. To differentiate the annotation based on which the speakerturns are filtered, distinct icons are used for the Full View, Questions View, Summary View, and Marker View. This functionality enhances the user's ability to navigate speakerturns effectively, ensuring they can access relevant information with ease.

4.3) Conversation Artefact Generation

Microsoft Teams offers a unique capability to convert speech to text in real-time through Azure cognitive speech services (Speech Studio, n.d.), a feature not found in other conferencing software. This functionality, coupled with automatic linking of converted text to the respective speaker with timestamps (referred to as Speakerturns), enhances the efficiency of converting conversations into natural language (figure 22 presents an example of a generated speakerturn).

Microsoft Teams offers a unique capability to convert speech to text in real-time through Azure cognitive speech services (Speech Studio, n.d.), a feature not found in other conferencing software. This functionality, coupled with automatic linking of converted text to the respective speaker with timestamps (referred to as Speakerturns), enhances the efficiency of converting conversations into natural language.

The utilization of MS Teams has been analysed in both online and face-to-face settings. However, it's essential to acknowledge certain limitations associated with MS Teams that were uncovered during testing the performance of the system. One such limitation pertains to the generation and sorting of speakerturns by the system. MS Teams generates speakerturns by associating transcribed text with the source of the audio, typically the MS Teams client from which the audio originates. Consequently, it's imperative for each user to be present in the MS Teams meeting using their respective MS Teams client to ensure accurate differentiation of speakerturns.

During the performance testing of the speech-to-text functionality (refer to Chapter 5 for validation details), a notable challenge arose in face-to-face conversations conducted using a single laptop with only one MS Teams client. In this scenario, only one speakerturn containing all conversation text was generated, highlighting the necessity for individual MS Teams clients for accurate speakerturn differentiation. While online conversations inherently overcome this limitation, face-to-face settings necessitated the development of an online conversation configuration. This configuration involved the use of two laptops, each equipped with its own MS Teams client, within a single meeting room. Additionally, to ensure proper linkage of speakerturns to respective individuals, a manual microphone mute button was integrated with the MS Teams client (refer to Figure 38).
To ensure the usability of the system, several enhancements were incorporated into the conversation artefact generation process to align with the requirements of documenting speakerturns and capturing accurate requirement-relevant data. The efficacy of MS Teams' speech-to-text functionality was evaluated through various test conversations. However, the system in its current state cannot be deemed practical due to the inadequate accuracy of the conversation artefacts. In both online and face-to-face settings, factors such as unfamiliar words, pronunciation variations, and foreign language terms contribute to this accuracy challenge. Specifically, in face-to-face scenarios, hardware quality, microphone capabilities, room acoustics, background noise, and simultaneous conversations further exacerbate the issue.

To elevate the accuracy to usable levels within the broader system design, as-is a crucial procedural step is necessary. This step involves comparing the conversation artefacts with the audio recordings of the conversations and making necessary corrections. However, this correction process is time-intensive, as elaborated upon in the validation chapter, rendering it impractical for direct implementation.

In instances where only one laptop and one MS Teams account are used during meeting recordings, MS Teams consolidates all conversation text into a single speakerturn (refer to the validation chapter). To address this limitation, a "push-to-talk" system was implemented. This solution entails the use of two laptops, each equipped with a microphone and a push button (as depicted in Figure 38). Both laptops share a single MS Teams account and are situated in the same meeting room, thereby facilitating accurate separation of speakerturns. Notably, the online setting already operates as intended, as all participants are inherently present in the virtual meeting room with their respective accounts.
In the chapter implementation a solution is presented that makes use of the MS Teams meeting room system (currently being used in Fizor), a custom speech model of MS Teams that can be trained, and a MS Teams integrated smart speaker that have the potential to solve the problems related to accuracy and speakerturn formulation but could not be tested within the scope of this thesis.

4.4) Marker functionality

This section discusses the prototype that has been realized in an effort to enhance the notetaking process. This prototype, that uses two Arduinos to function, interacts with the system that has been realized in this thesis. As a first step to enable enhanced notetaking an interaction has been designed that enables Business Analysts to annotate important 1/0 to the current speakerturn by means of an impulse. Providing an impulse 1/0 can be easily detected and needs almost no effort to perform.

The idea on which is built is the notion that BAs will capture important parts of the conversation by documenting keywords and sentences. What if these notes could be linked to the conversation? The word keyword would suggest a factor of importance for a specific part of conversation data. If notes would be made in sequential fashion and the speakerturns could be annotated each time a note would be written down the index of the notes could be matched with the index of the marked/annotated speakerturns. Index 1 of the notes would match for example index 1 of the marked speakerturns. This would provide an overview of all the parts of the conversation that were deemed as important by the analyst, providing a filtered dataset of conversation data. This assumption proved to be flawed during user testing as the Business Analyst doesn't only make notes to capture data but also uses them during the conversation, so the interaction needs to be designed further. There could be iterated on this by for example providing separate areas for data capture where sequence can be applied or iterating with the strategy of providing inputs or allow multiple kinds of inputs, however this needs to be researched further.

Design of interaction

Humans create an order to everything they see where there are a couple of principles about how we perceive hierarchy that hold true for every human (Johnson, 2021). Big > small, high contrast > low contrast, color > grays, red> other colors (Johnson, 2021). There is already an action that humans widely use in their daily live to bring hierarchy to natural language and that is the act of Marking information. By marking, or coloring text people perceive it as more important (color>grays). You can also see the action of marking everywhere from paper documents to digital systems such as excel and word. There have been ideated with two types of marking, marking of relevant conversation moments after the conversation, by marking speakerturns in a GUI (figure 17, on the right) or marking conversation speech as being important in the moment. The latter has been considered more impactful as the marking afterwards requires the user to read the whole conversation again to process if a speakerturn is important.

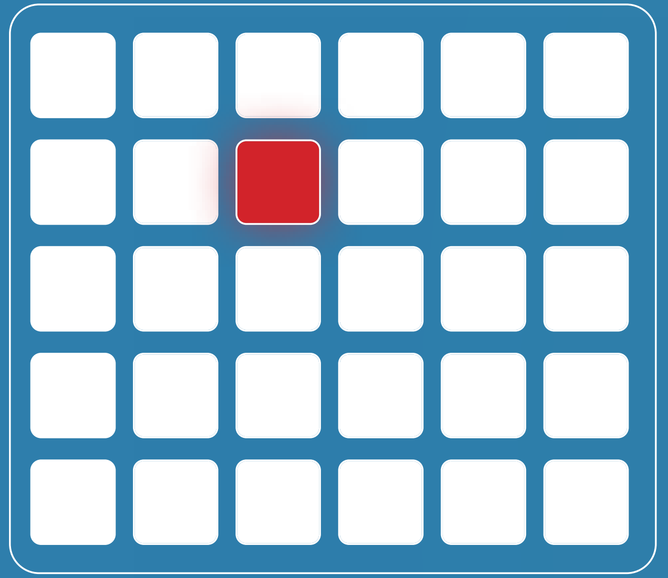
Different impulse types

Three types of impulse have been considered. All types of impulses integrate with the note taking process but each in a different way. First, a button that is integrated in the systems that are already used such as a pen and a laptop. Another being the detection of when a note is being taken, and finally a board with only button inputs relying solely on annotations. Figure 32 presents an overview of each of these types of impulses.

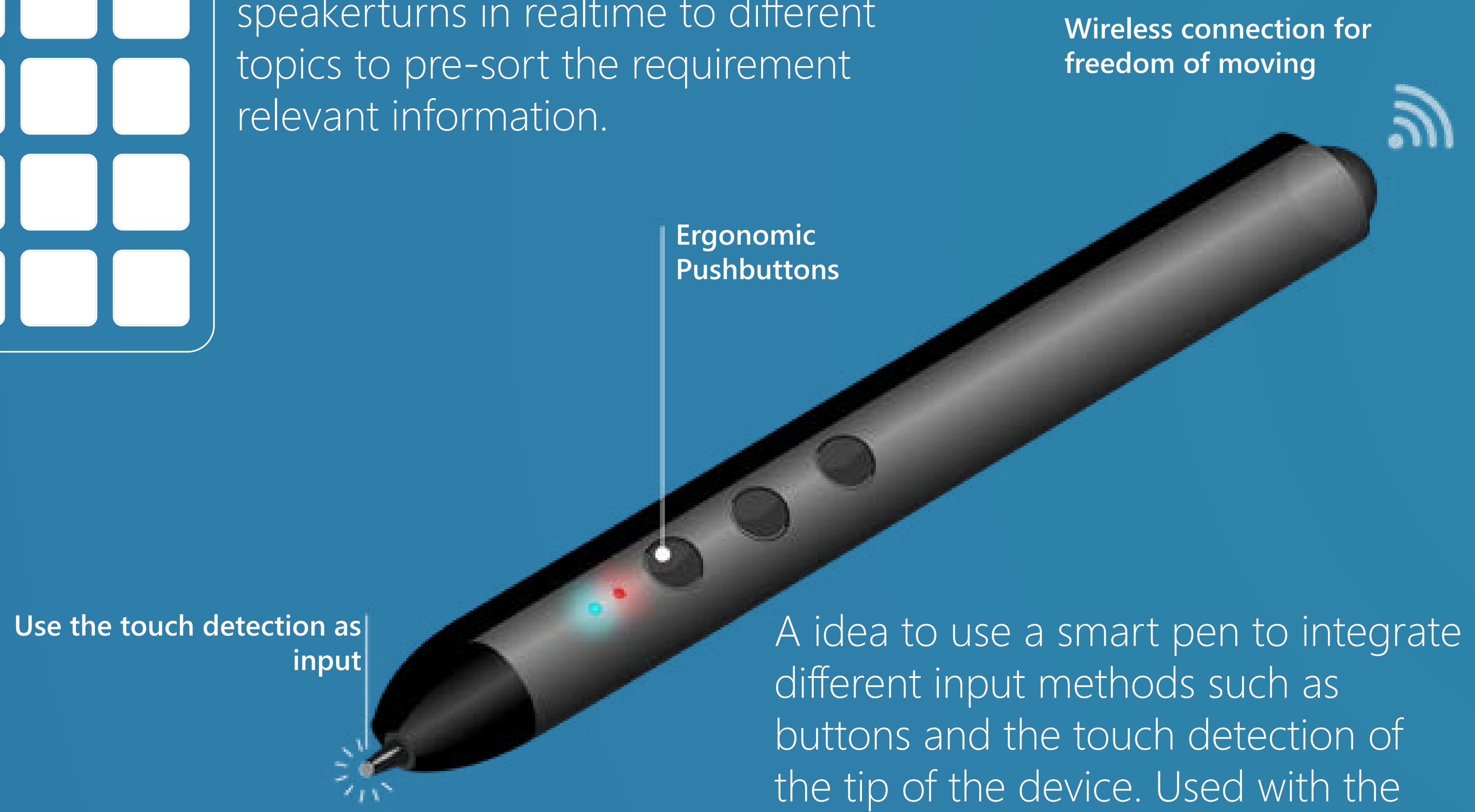
As proof of concepts a button prototype has been created that allows Business Analysts to annotate a speakerturn with important 1/0. In later development this button can be integrated in systems such as smart pens, or in a keyboard for example (figure 32) to make use of the advantages making notes has on digital devices. It can for example be integrated with systems such as remarkable (figure 36), a product that acts as a 'paper tablet'(ReMarkable, n.d.) or drawing tablets that allow direct digitalization of notes or integration with word processing software such as Microsoft word. Besides the ability to provide an input this input needs to be linked to the

4. REALIZED SYSTEM

Figure 32: Interaction ideas to integrate with 'enhanced notetaking' functionalities.

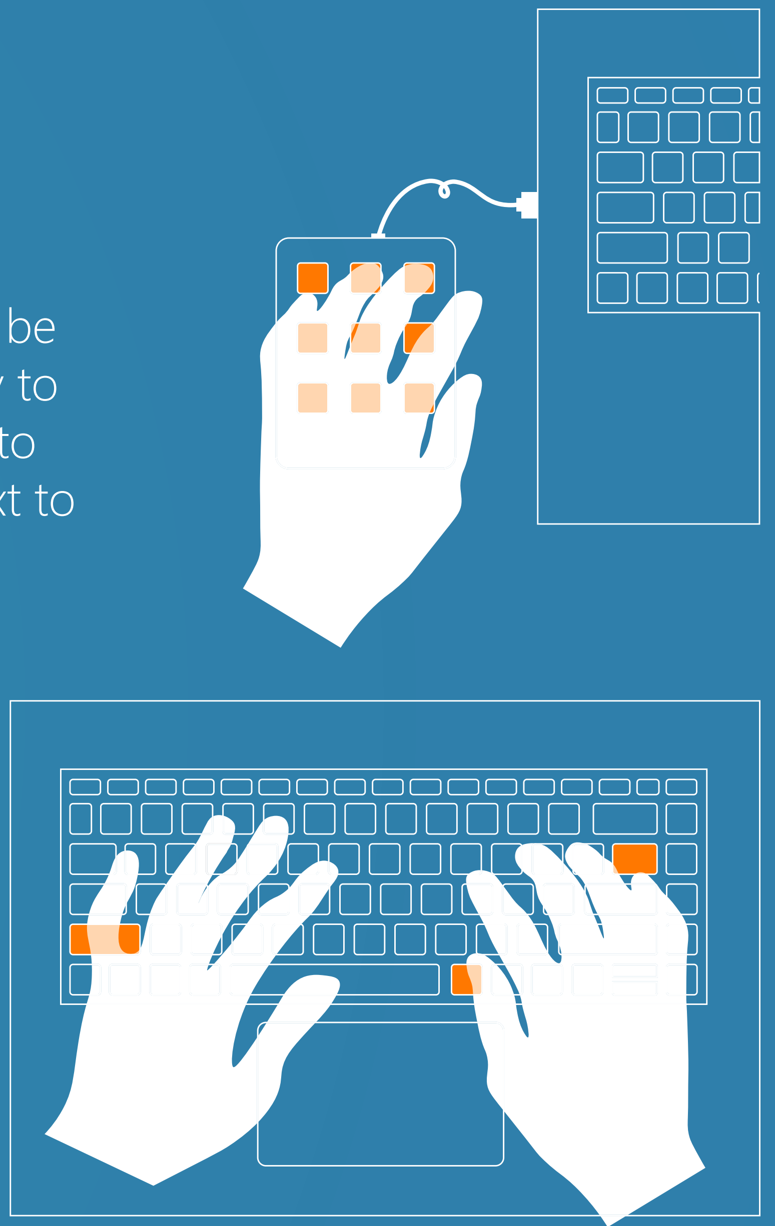


An idea to bind different annotators (such as different topics/keywords) to different buttons to allow to bind speakerturns in realtime to different topics to pre-sort the requirement relevant information.


Wireless connection for freedom of moving
Ergonomic Pushbuttons
Use the touch detection as input

A idea to use a smart pen to integrate different input methods such as buttons and the touch detection of the tip of the device. Used with the interaction of writing.

When using the keyboard it may be more beneficial bind functionality to specific keys on the keyboard or to bind buttons that are located next to the keyboard.



4. REALIZED SYSTEM

speakerturns that are being generated. While it may be possible in later development using the Speech Software Development Kit (Speech SDK) of the Microsoft Azure speech services (Speech SDK, n.d.) this could not be realized in the prove of concept. Microsoft Teams itself allows from itself no direct interaction with the speakerturns while they are being generated so the annotation functionality has been realized by synchronizing the prototype with the conversation. By using this method, the device can store all the inputs of the Business Analyst with the time it was inputted. Afterwards the system can compare the timestamps of the inputs and annotate the corresponding speakerturn.

Synchronizing and storing user inputs

As Microsoft Teams allows for almost no interaction with its data during the conversation (no access to speaker, time, or text data while it is being generated, with no ability to remotely start and stop the conversation with external signals) some trickery needed to be applied to allow the synchronisation of the conversation with the prototype. Two arduinos have been used to make this possible in the end where the Arduino Uno keeps track of the time and the Arduino Pro Micro allows the external start and end of the meeting recording functionality. By setting the current time on 0:00:00 when the start signal is given the time of the device and the recording can be synched. These signals of the Arduino Pro Micro can be realized as it can provide keyboard inputs (by connecting it to a laptop using usb), making it able to input keybindings that could be used to manage MS Teams functionalities. While this solution allowed the synchronization between MS Teams and the protoype it proved very difficult to setup and is not suited to be used by users in practice at it can easily go wrong needing the meeting to be restarted to reset the prototype, or no inputs are able to synch. It can however currently be used in controlled test settings where a researcher sets up the device.

Linking markers to imported conversations

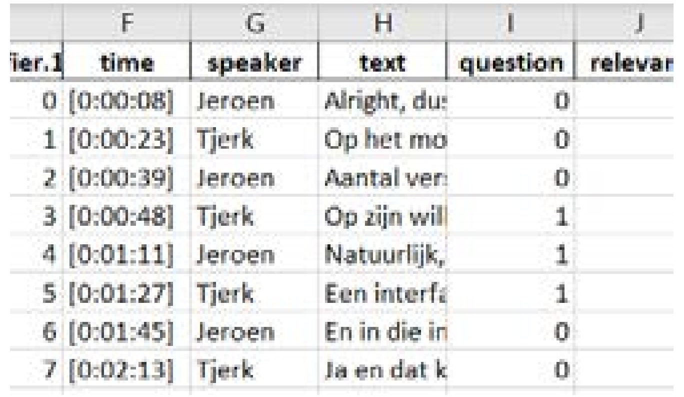
After the user has imported the corresponding conversation into the system the created marker array can be linked to the conversation through the system GUI. Using the marker file overview in the correct markerfile can be selected and linked. The create_markerfile.py algorithm is run, annotating the speakerturns that have been marked (the time marker compared to between which speakerturn times it is positioned) and storing the annotated speakerturns in the conversation database. To give an example, if there is a speakerturn with the time [0:00:50] and another with [0:02:30], the speakerturn with [0:00:50] will be annotated as marked with a time marker of [0:01:30] as it occurs before the start of the next speakerturn. After annotating the output is again stored under a different name in the 3) markerfiles folder (appendix chapter 16).

As can be seen in figure 37 the marked speakerturns can be inspected in the GUI using the marker view filter.

4.5) Potential of advanced notetaking interaction

When Business Analysts participate in an elicitation conversation, they are already annotating the conversation data through note taking and mentally. With MS Teams converting speech to text in real time already to capture a Conversation Artefact containing the whole conversation dataset it can be of value if Business Analysts can annotate, based on their own experience, what parts of the conversation are relevant. In the GUI the user could then view only the conversation data that was deemed relevant.

This concept is still in the proof-of-concept stage and requires additional refinement and validation of its value. Enabling Business Analysts to annotate real-time conversations with basic inputs has the potential to significantly enhance note-taking capabilities or serve as a research tool for extracting requirement-relevant insights without interrupting the conversation.



	F	G	H	I	J
ier.1	time	speaker	text	question	relevar
0	[0:00:08]	Jeroen	Alright, du:	0	
1	[0:00:23]	Tjerk	Op het mo	0	
2	[0:00:39]	Jeroen	Aantal ver:	0	
3	[0:00:48]	Tjerk	Op zijn wil	1	
4	[0:01:11]	Jeroen	Natuurlijk,	1	
5	[0:01:27]	Tjerk	Een interfa	1	
6	[0:01:45]	Jeroen	En in die in	0	
7	[0:02:13]	Tjerk	Ja en dat k	0	

Figure 33: Annotated part of a mock elicitation conversation between me and the company supervisor using both the marker prototype and the system design.

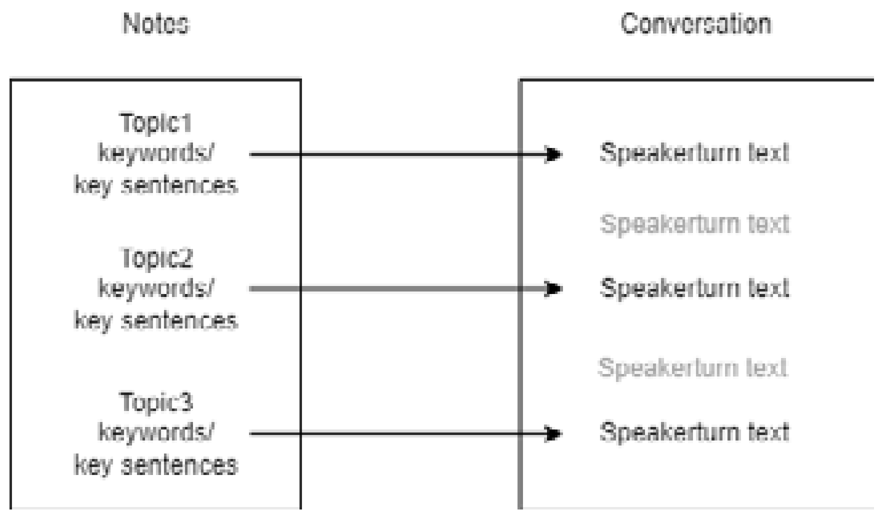


Figure 34: Idea to link notes to marked speakerturns. If notes would be made in sequence and a signal is provided each time a note is made note 1 would match with the first marked speakerturn providing more information on that specific note.

KEYBOARD PROTOTYPE

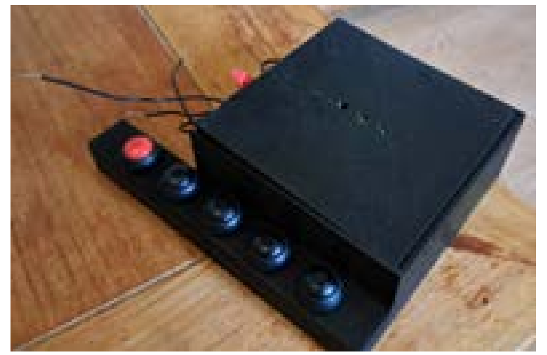
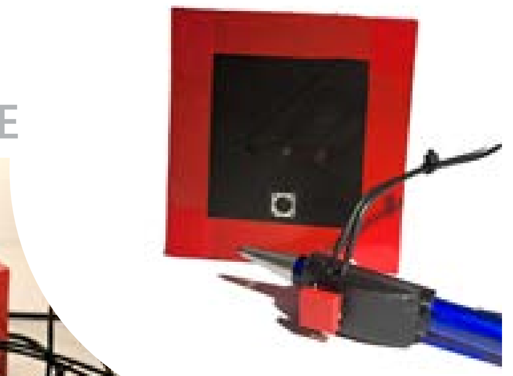


Figure 36: remarkable (ReMarkable, n.d.)

PEN/BUTTON PROTOTYPE



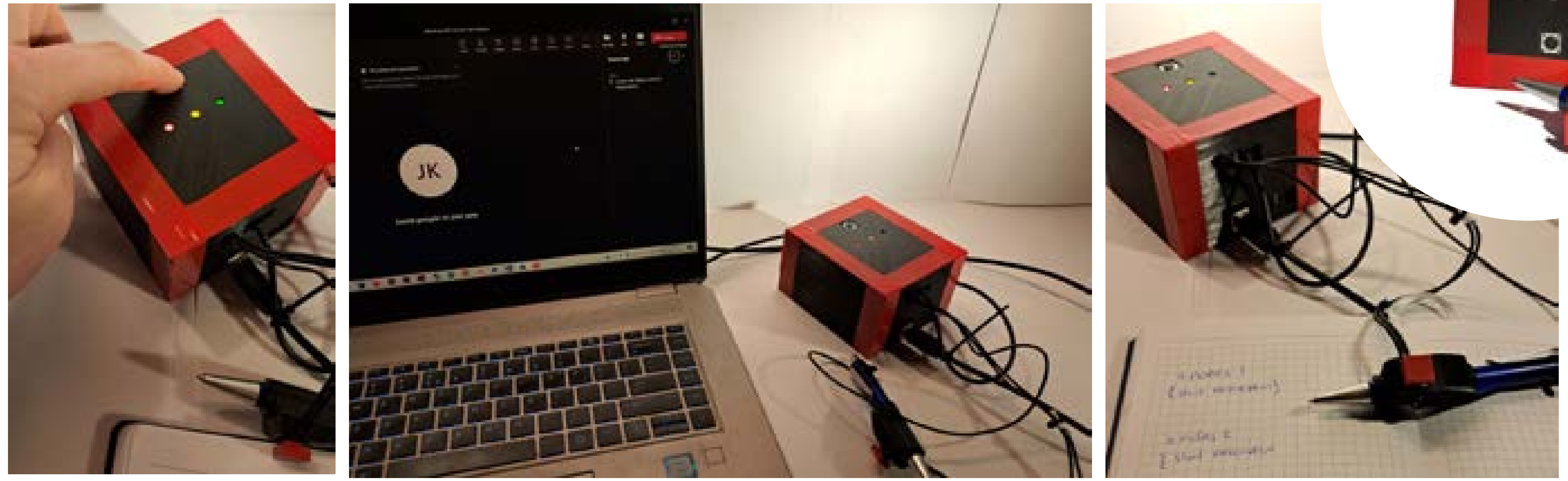
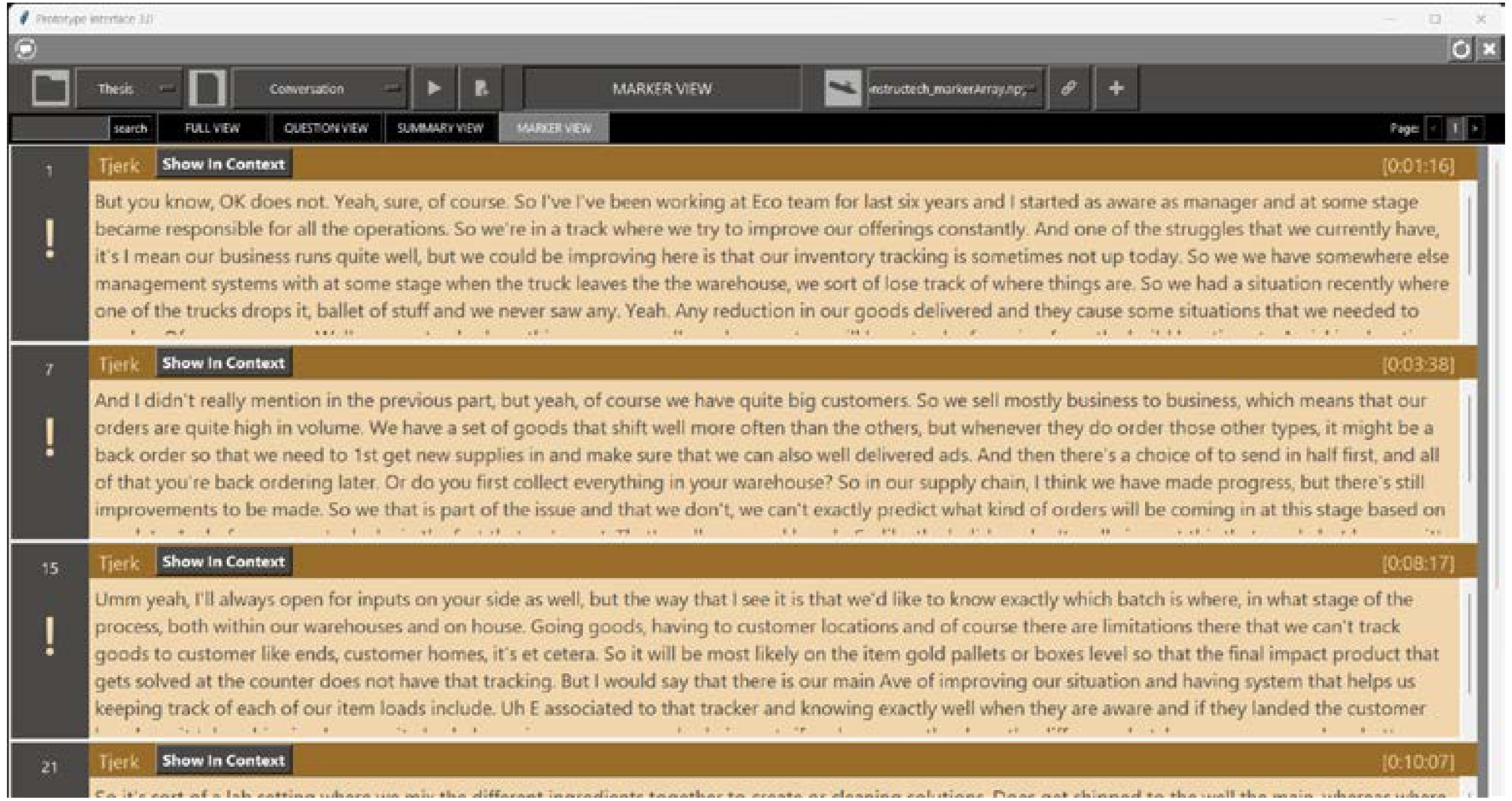


Figure 35: Functional marker prototype ideas integrating inputs into a pen and button board. The functionality of the pen protype has been realized.



1 Tjerk Show In Context [0:01:16]

!

But you know, OK does not. Yeah, sure, of course. So I've I've been working at Eco team for last six years and I started as aware as manager and at some stage became responsible for all the operations. So we're in a track where we try to improve our offerings constantly. And one of the struggles that we currently have, it's I mean our business runs quite well, but we could be improving here is that our inventory tracking is sometimes not up today. So we we have somewhere else management systems with at some stage when the truck leaves the the warehouse, we sort of lose track of where things are. So we had a situation recently where one of the trucks drops it, ballet of stuff and we never saw any. Yeah. Any reduction in our goods delivered and they cause some situations that we needed to

7 Tjerk Show In Context [0:03:38]

!

And I didn't really mention in the previous part, but yeah, of course we have quite big customers. So we sell mostly business to business, which means that our orders are quite high in volume. We have a set of goods that shift well more often than the others, but whenever they do order those other types, it might be a back order so that we need to 1st get new supplies in and make sure that we can also well delivered ads. And then there's a choice of to send in half first, and all of that you're back ordering later. Or do you first collect everything in your warehouse? So in our supply chain, I think we have made progress, but there's still improvements to be made. So we that is part of the issue and that we don't, we can't exactly predict what kind of orders will be coming in at this stage based on

15 Tjerk Show In Context [0:08:17]

!

Umm yeah, I'll always open for inputs on your side as well, but the way that I see it is that we'd like to know exactly which batch is where, in what stage of the process, both within our warehouses and on house. Going goods, having to customer locations and of course there are limitations there that we can't track goods to customer like ends, customer homes, it's et cetera. So it will be most likely on the item gold pallets or boxes level so that the final impact product that gets solved at the counter does not have that tracking. But I would say that there is our main Ave of improving our situation and having system that helps us keeping track of each of our item loads include. Uh E associated to that tracker and knowing exactly well when they are aware and if they landed the customer

21 Tjerk Show In Context [0:10:07]

Figure 37: Marker view output. All speakerturns that were marked as important by the Business Analyst during an elicitation conversation.

5. VALIDATING THE REALIZED SYSTEM

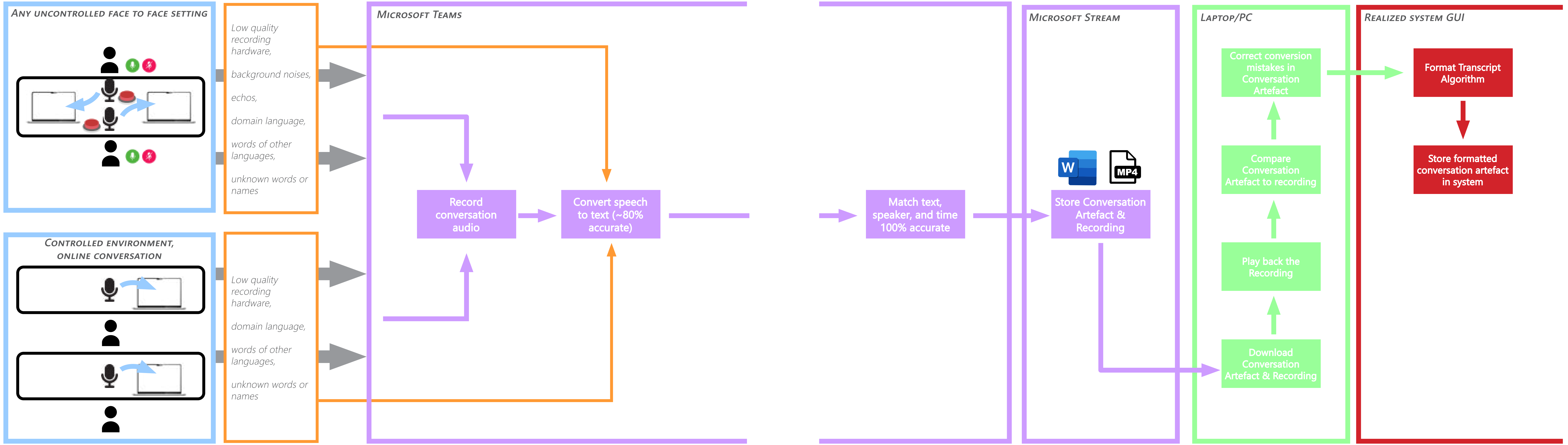

Any uncontrolled face to face setting
Controlled environment, online conversation
Low quality recording hardware,
background noises,
echos,
domain language,
words of other languages,
unknown words or names
Low quality recording hardware,
domain language,
words of other languages,
unknown words or names
Microsoft Teams
Record conversation audio
Convert speech to text (~80% accurate)
Match text, speaker, and time 100% accurate
Microsoft Stream
MP4
Store Conversation Artefact & Recording
Laptop/PC
Correct conversion mistakes in Conversation Artefact
Compare Conversation Artefact to recording
Play back the Recording
Download Conversation Artefact & Recording
Realized system GUI
Format Transcript Algorithm
Store formatted conversation artefact in system

Figure 38: Validation results of the conversation artefact generation process and the necessary step to make the output usable for the system.

5. VALIDATING THE REALIZED SYSTEM

In the validation of the realized system, the focus is on assessing whether the system can be practically used, whether its design facilitates the capture and search for specific conversation data, and whether the concept of advanced notetaking proves useful. While the system has been validated for functionality, further validation and development through practical use are necessary. Key questions addressed in this chapter include:

1. Can the system be used as-is in practice?
2. Does the system design facilitate the capture and search for specific conversation data?
3. Is the concept of advanced notetaking useful?

Although the system has been validated for functionality, practical validation through real-world usage is crucial. Suggestions for validating different aspects of the system are provided in this chapter.

The subsequent chapter will detail an implementation plan for integrating the system into the workflow of Fizor. It will explore how this implementation can add value to the company and facilitate the capture of research data.

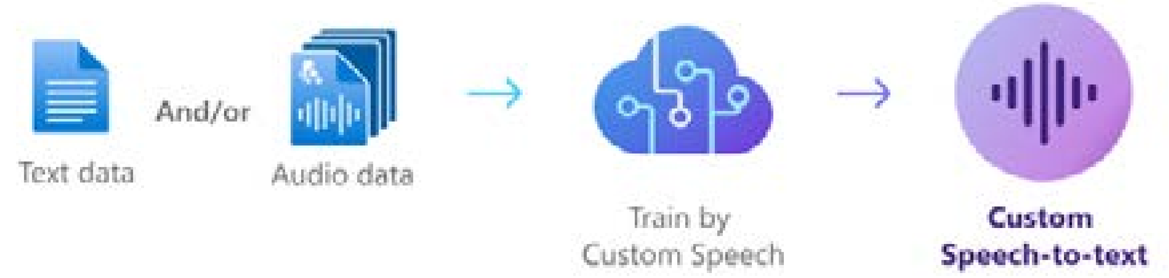


Figure 39: MS Teams custom speech model.

5.1 Can the system be used as-is?

To assess the system's usability in its current state, various aspects must addressed. First, must be determined its effectiveness in capturing conversations across different settings. Understanding any limitations in specific environments is crucial for identifying where the system can be most effectively utilized. Beyond mere capture, the system's ability to automatically process these conversations is paramount. Without this capability, its practical utility may be limited. Users should also be able to inspect the processed results to ensure accuracy and relevance. Moreover, accessibility and ease of use are key factors. A system that is cumbersome or difficult to operate will likely face adoption challenges. Thus, evaluating its accessibility and user-friendliness is essential. Finall must be considered whether the system generates useful data. The ability to derive actionable insights from the processed conversations adds significant value. Overall, a comprehensive assessment of these factors will provide insights into the system's current usability and areas for improvement.

Can the system be used to capture conversations in any conversation setting?

The interviews conducted during the analysis phase were utilized as a case study to analyze the performance of MS Teams' speech-to-text functionality across three environments. These environments were as follows:
1. Online Setting: Participants were situated externally, using laptop microphones for audio recording.
2. Face-to-Face Setting (Uncontrolled Environment): Conversations were recorded using a single laptop and its microphone in public spaces or study areas. Only one MS Teams client was utilized for recording.
3. Dedicated Meeting Rooms: Equipped with high-quality recording hardware and sound isolation.

In each setting, conversations were recorded and transcribed using MS Teams' speech-to-text functionality. The performance of each transcript was evaluated by comparing it to the conversation audio to identify transcription errors. Appendix chapters 8 and 9 present the raw captured conversation and the corrected conversations, highlighting the differences. This captured conversation constitutes the output generated from the final test conversation between the client supervisor (playing the client for a fictive case) and a business analyst expert.

A crucial requirement for the conversation artifact to be usable is high accuracy, ensuring that no crucial conversation elements are transcribed incorrectly. The face-to-face setting exhibited the poorest performance due to environmental noise, such as background conversations and echoes, significantly affecting transcription accuracy. Additionally, occasional spikes in the laptop microphone resulted in audio that was sometimes illegible, leading to inaccurate text conversion. Furthermore, speakerturns were not generated correctly in this setting, as outlined earlier, system 1. This issue has been addressed through the method illustrated in Figure 38 .

In contrast, the online setting demonstrated better performance due to its isolated environment and individual presence in MS Teams meetings, mitigating environmental noise issues and ensuring accurate speakerturn generation. Dedicated meeting rooms performed the best overall in terms of transcription accuracy, primarily due to superior recording hardware and controlled environments. However, no setting proved accurate enough as-is. MS Teams employs a standard base model for speech-to-text conversion, but inaccuracies occur due to factors such as unrecognized domain-specific terms, background noise, language variations, and pronunciation issues. Microsoft acknowledges the impact of domain-specific language and speaking styles on transcription accuracy accuracy (Microsoft Custom Speech, n.d.).

It's worth noting that the manual correction of conversation artifacts renders the system unsuitable for practical use by users. However, if the aim is for researchers to test and improve system functionalities utilizing conversation artifacts, the correction step can be performed by researchers to enable the inputs to be used by the rest of the system.

In what setting can the system be used as is to capture conversation data?

As mentioned, utilizing a dedicated meeting room equipped with high-quality audio recording equipment, particularly an 'MS Teams meeting room setup' (hardware with its own operating system tailored to the functionalities of MS Teams for optimal hybrid collaboration), is likely to yield the highest quality conversation artifacts. Employing this environment initially offers several advantages, which will be discussed here.
In the dedicated meeting room, both in-person and online conversations are supported, facilitating the accurate formulation of speakerturns. However, the absence of speakerturns in online conversations can be remedied by integrating a smart speaker, which is also cited as one of the potential benefits of utilizing an MS Teams meeting room.

Ability to train the base model of MS Teams

To achieve conversation artifacts with 100% accuracy, a custom speech model and word list can be employed (as per MS Teams source). Currently, MS Teams utilizes a standard base model for converting speech to text, leading to inaccuracies due to factors like unrecognized domain-specific terms, background noise, language variations, and pronunciation issues. Microsoft acknowledges the influence of domain-specific language and speaking styles on transcription accuracy (Microsoft Custom Speech, n.d.). Although not integrated into the realized system, a solution for this challenge is already available in the market. Microsoft has introduced a custom speech-to-text model (figure 39) and phrase lists, offering the potential for near-perfect transcriptions (Microsoft Custom Speech, n.d.).

Phrase lists offer a lightweight approach to enhance speech recognition of unique words and phrases, thereby improving transcription accuracy. These lists are presented in advance to optimize recognition and can be implemented just-in-time before speech recognition begins. Additionally, phrase lists are lightweight and can be integrated into various programming languages, including Python. By combining a custom speech-to-text model with phrase lists, transcription accuracy can be significantly enhanced, ensuring that transcriptions meet the required standards for readability and content accuracy.

The custom speech model enables the recognition of complex and diverse speech patterns, language variations, and domain-specific terminology, thereby enhancing transcription accuracy. This model can be trained to recognize specific domains such as medical terminology, financial jargon, or IT terminology. Moreover, the training of the custom speech model is conducted using training data (conversation audio). Figure 41 illustrates the difference in transcription accuracy between the base model and the trained custom speech model. As part of the speech model training, ambient noise filtering can be implemented based on training data comprising audio data of the ambient noise, thereby improving the accuracy of the speech conversation. However, this can only be achieved in a consistent environment.

5. VALIDATING THE REALIZED SYSTEM

It's important to note that the accuracy of conversation artifacts after implementing the aforementioned solutions needs validation, as this was not possible in this thesis. Both the custom model, the phrase list, and the high-quality audio recording equipment require resource investments that need to be considered for evaluating the solution.

Ability to recognize multiple speakers in a face-to-face setting

The ability to accurately recognize multiple speakers in a face-to-face setting using just one laptop was not achieved during this thesis. The push-to-talk method (illustrated in Figure 38) was explored but found to be impractical for elicitation interviews as it requires participants to manage their microphone's status, leading to potential distractions and lapses in concentration.

To enhance the system's capability to capture and generate conversation artifacts, an improvement could involve integrating a smart speaker (designed to seamlessly integrate with Microsoft Teams functionalities, and depicted in Figure 40) into the meeting room environment. This smart speaker facilitates face-to-face conversations involving up to 10 participants by setting up voice profiles for each individual. By recognizing participants' voices, the smart speaker (figure 40) can accurately attribute speakerturns to the corresponding individuals, thus addressing the challenge of incorrect speakerturn formulation in face-to-face conversations. Integrating such a product would enable the processing of client/stakeholder meetings into conversation artifacts and facilitate the documentation of brainstorming sessions involving multiple participants in the same room. It's worth noting that the smart speaker needs to be connected to a Microsoft Teams meeting room setup and is therefore not a portable solution.

By incorporating both of these additional solutions, the envisioned system (as illustrated in Figure 66) can be realized, offering improved capabilities for capturing and generating conversation artifacts in various settings.

Is the system capable of automatically processing a conversation?

Yes, the system in its current state can automatically process conversation artifacts (if generated using MS Teams) by detecting which speakerturns contain questions and determining the relevance of these questions, as outlined in the NLP functionalities chapter. All outputs generated at each processing stage are stored within the system for analysis.

The functionality of these processes has been validated by processing multiple conversations numerous times, analyzing the outputs at each stage to ensure correct functionality and timing. Appendices 8-16 present the outputs generated at each stage, including the results from processing the test conversation used in the final user test. It should be noted that while not all questions have been validated to be identified correctly, manual inspection of conversations can aid in quantitatively proving the effectiveness of this functionality. By manually processing system outputs on a large scale, conclusive results can be obtained, demonstrating the system's ability to accurately identify all questions present in the speakerturns. If it is determined that the processing is not 100% accurate, the system should prioritize recall over precision, ensuring that all questions are at least identified. Additionally, an expert evaluation should be conducted to assess whether any relevant questions are filtered out (again, prioritizing recall over precision).

Running the processing functionality requires only an average of 10 mouse clicks and the provision of a conversation title, two speaker names, and a file path, taking up approximately 2 minutes. The processing itself takes an average of 10-15 minutes and can be run in the background. Additionally, users are required to install Java, which takes a one-time investment of approximately 5 minutes with no need for complicated installation steps. Given the minimal time required to process a conversation into measurable outputs, the system presents an opportunity to generate a significant amount of research data while minimizing user effort.

Can the processed results be inspected?

Database

The processed results are valuable for both researchers and users of the system, with researchers interested in analyzing the results and users wanting to utilize the processed conversation data for various activities. By storing outputs in consistent locations and formats (using consistent names and Excel format), researchers can easily locate specific processing outputs for research purposes.
It's essential for any processed conversation data to be accessible, meaning users have physical access and permissions to access the data. To address this requirement, a portable database stored on a USB flash drive has been created, allowing users to access it anywhere. However, in organizations with multiple users, a shared centralized conversation database is necessary. This centralized database should employ a consistent strategy for storing generated conversations.

Integration with the MS Teams Graph API enables automatic access to conversations generated by MS Teams. Users can effortlessly import transcripts generated in Stream directly into the database, enabling prompt processing post-conversation. Microsoft Graph APIs for Microsoft Teams meeting transcripts facilitate this seamless integration, requiring only MS Teams login credentials to authorize automatic importation (Microsoft Graph API, n.d.).

GUI

Users can inspect the processed results through the graphical user interface (GUI), as illustrated by the GUI visuals in the chapter realized system. The visualization of speakerturns was validated by comparing the processed outputs stored in the system to the data visualized in the GUI. All GUI functionalities have been tested using small test conversations to compare expected system behavior with actual behavior. The GUI underwent iterations until it performed as expected.

However, while the system behaves as expected, it's essential to ensure that information is presented to users as desired. Users may have additional or different requirements, perceive system functionality differently, or desire more customization options. Gathering user feedback from system usage is crucial for addressing these requirements. Figure 8 provides an overview of different techniques for evaluating user interactions (Leary & West, 2023). Performance metrics such as effectiveness in task performance, efficiency, and user-friendliness should be assessed using a sufficiently large sample size (10-20) to draw conclusive statements.

5.2 Does the design aid in the capture of and searching for specific conversation data? & Does the design aid in the amount of requirement relevant information that can be captured with notetaking using the marker prototype?

As referenced earlier in this thesis, a final test was set up to provide answers to these questions. The test involved an interview and a group discussion, with a total of 9 experts, including the client supervisor and their PhD supervisor.

A simulated elicitation interview was conducted, where a fictive client (represented by the client supervisor) was interviewed by a Business Analyst (played by one of the experts). The interview took place face-to-face, utilizing a setup (as shown in Figure 44) to ensure the correct separation of speakert urns. These buttons were managed to keep the focus of the participants on the conversation. The interview focused on eliciting requirements for a fictive case formulated for this purpose (included in the test plan in Chapter 7 of the appendix).

The Business Analyst was provided with the marker prototype (with the pen input replaced by a button) and received a short introduction on its functionality beforehand. They were asked to use the marker prototype while taking notes when requirement-relevant information was provided by the client. The other experts were instructed to take notes of the conversation for comparison. However, due to a desynchronization between the marker time and the conversation time, this comparison could not be made as planned. This issue arose from the setup of the prototype in advance, which resulted in an unforeseen conflict between MS Teams and the prototype. Chapters 8-16 of the appendix display all the outputs generated during the session.

Following the interview, a demonstration of the system design and processes was conducted. One of the experts was tasked with performing various actions within the prototype, highlighting areas for improvement in button wording and icon clarity. Subsequently, a group discussion ensued, covering topics to assess the performance of both the system design and the marker prototype. While a group discussion with experts cannot lead to definitive conclusions (Eger et al., 2012), it can provide valuable insights into the validity of the designs.

The main discussion points included questions such as 'Would you use the systems as-is, why

5. VALIDATING THE REALIZED SYSTEM

(not)?', 'Does the marker prototype aid in the notetaking process, how (not)?, and can it fully replace notetaking?', and 'Does the system design provide value to the Business Analyst by allowing the filtering of conversations?' Alongside brainstorming for further improvements, these questions guided the discussion.

Due to time constraints, the interview was limited to 15 minutes, resulting in a small conversation dataset. With such a short duration, manual processing of the data is less burdensome, as the conversation is easily remembered, leaving little room for irrelevant data to be generated. Consequently, the utility of the systems for such small datasets may be questioned, especially considering that processing time is comparable to the conversation duration. While the necessary test conditions could not be fully realized during the thesis, allowing the experts to experience the systems' functionality in practice facilitated an insightful discussion.

Overall, the consensus among the experts was that both the system design and the marker prototype show significant potential and warrant further development. Despite the limitations of the small dataset and time constraints, the discussion highlighted the interest and enthusiasm for advancing these systems.

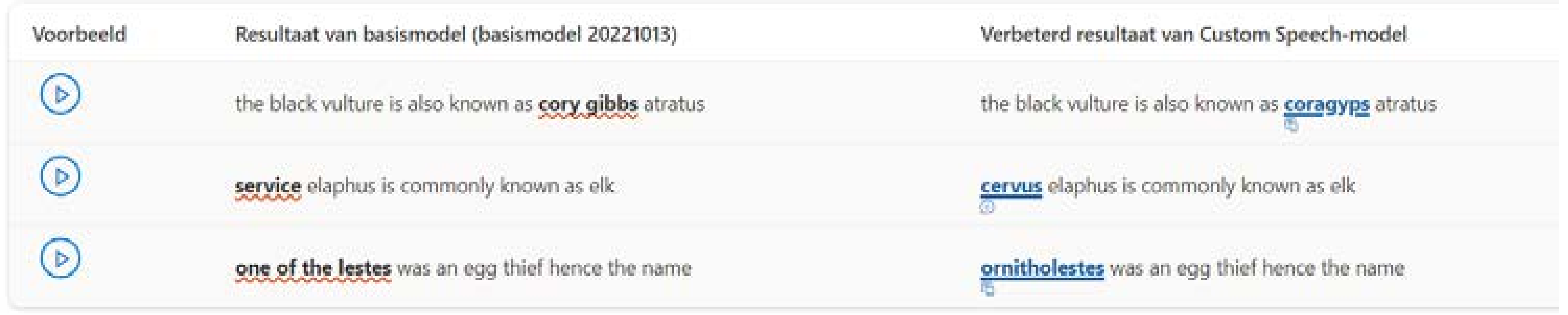
During the discussion, the marker prototype's function of marking speakerturns to aid in capturing requirement-relevant information was scrutinized. It was acknowledged that while dedicated transcribers could serve this purpose, not all companies have the resources for such an option. The cost-effectiveness and performance of each approach remain inconclusive, with mixed opinions among the experts regarding the use of the marker prototype as-is for data capture. However, further development was deemed necessary, especially considering that the push-button interaction was novel and required some time for analysts to adjust during the conversation. Nevertheless, aside from its data capture capability, the marker prototype was deemed valuable for providing measurable data from elicitation conversations without disrupting the natural flow of the discussion. Gaining insights into when and how analysts take notes could significantly contribute to further research in conversational requirement elicitation.

The potential to interact with speakerturns in real-time was recognized as offering valuable opportunities, such as quickly collecting, manipulating, and presenting speakerturns during the conversation, thereby enhancing communication between analysts and clients. However, realizing this functionality requires further research.

The ability to identify and filter questions, particularly relevant ones, was highly valued and deemed useful for practical use as-is. Suggestions for multiple changes, primarily focused on improving the GUI to better align with personal preferences, were also mentioned.

In conclusion, both the system design and the marker prototype were seen to hold high potential for providing value to business analysts. However, practical testing and development are necessary to fully realize this potential. A development roadmap, informed by the results of the validation phase, has been created for the client, outlining recommendations for further system development. The next chapter will detail how the system can progress from the current state to a realizable system, addressing accuracy and speakerturns issues. Additionally, the next chapter will outline plans for further research and development, including a roadmap for implementing conversation data processing into Fizor's workflow.

Figure 40: MS Teams integrated smart speakers of different brands.

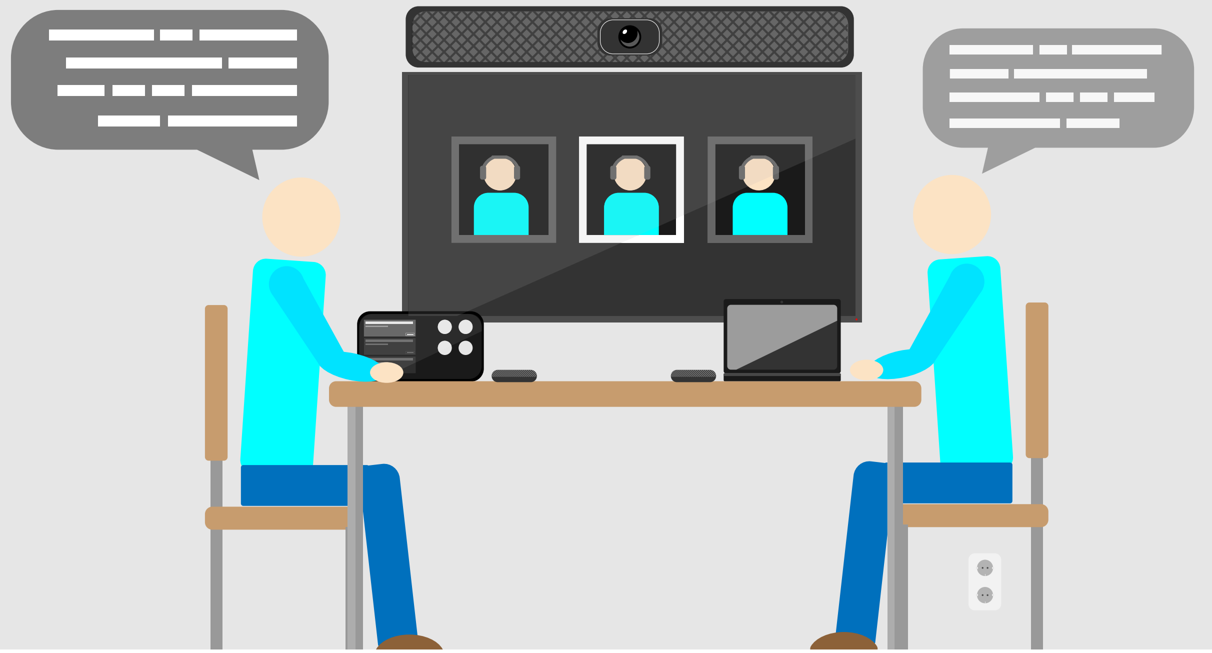


Voorbeeld	Resultaat van basismodel (basismodel 20221013)	Verbeterd resultaat van Custom Speech-model
▷	the black vulture is also known as **cory gibbs** atratus	the black vulture is also known as **coragyps** atratus
▷	**service** elaphus is commonly known as elk	**cervus** elaphus is commonly known as elk
▷	**one of the lestes** was an egg thief hence the name	**ornitholestes** was an egg thief hence the name

Figure 41: Example of domain specific words being mistranslated. Using the custom speech model of MS Teams makes it so it is translated correctly (from speech.microsoft.com).

Figure 42: MS Teams meeting room setup.

A MS Meeting room consists of a console that is used to manage the meeting room and runs on it's own operating system. To this console different kinds of hardware are usaually connected such as cameras speakers and monitors to enable hyrbid collaboration.



5. VALIDATING THE REALIZED SYSTEM

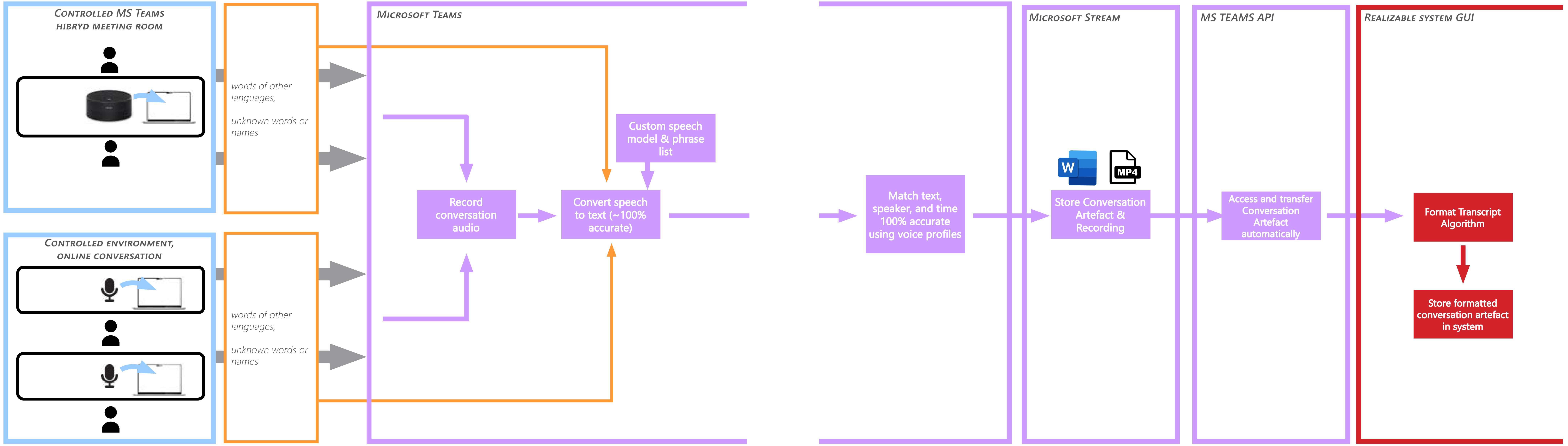

Controlled MS Teams
hibryd meeting room
words of other languages,
unknown words or names
Controlled environment,
online conversation
words of other languages,
unknown words or names
Microsoft Teams
Record conversation audio
Convert speech to text (~100% accurate)
Custom speech model & phrase list
Match text, speaker, and time 100% accurate using voice profiles
Microsoft Stream
W
MP4
Store Conversation Artefact & Recording
MS TEAMS API
Access and transfer Conversation Artefact automatically
Realizable system GUI
Format Transcript Algorithm
Store formatted conversation artefact in system

Figure 43: Realizable system making use of MS Teams hybrid meeting rooms and compatible smart speakers.

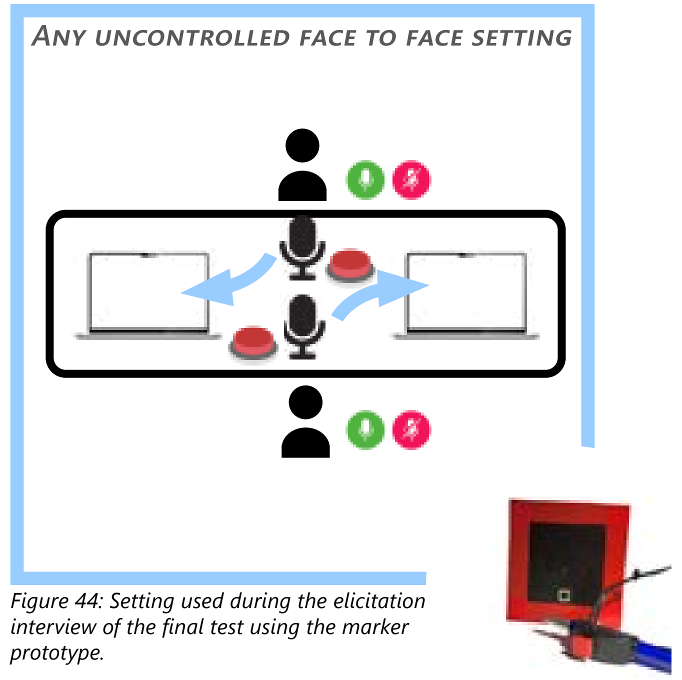


Figure 44: Setting used during the elicitation interview of the final test using the marker prototype.

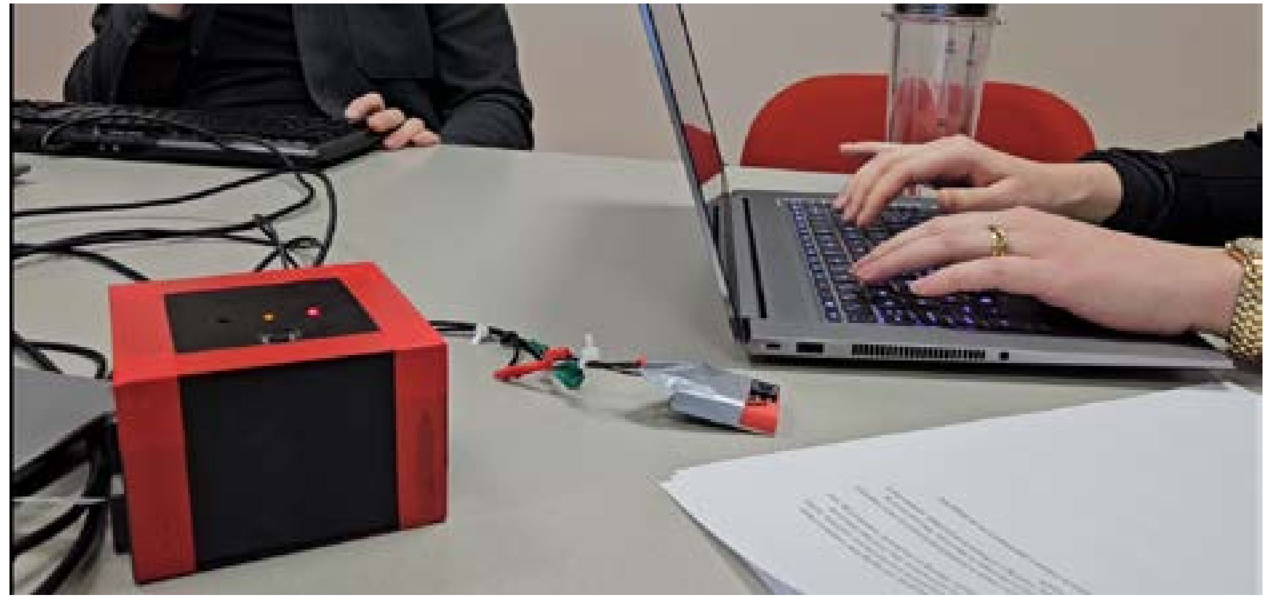
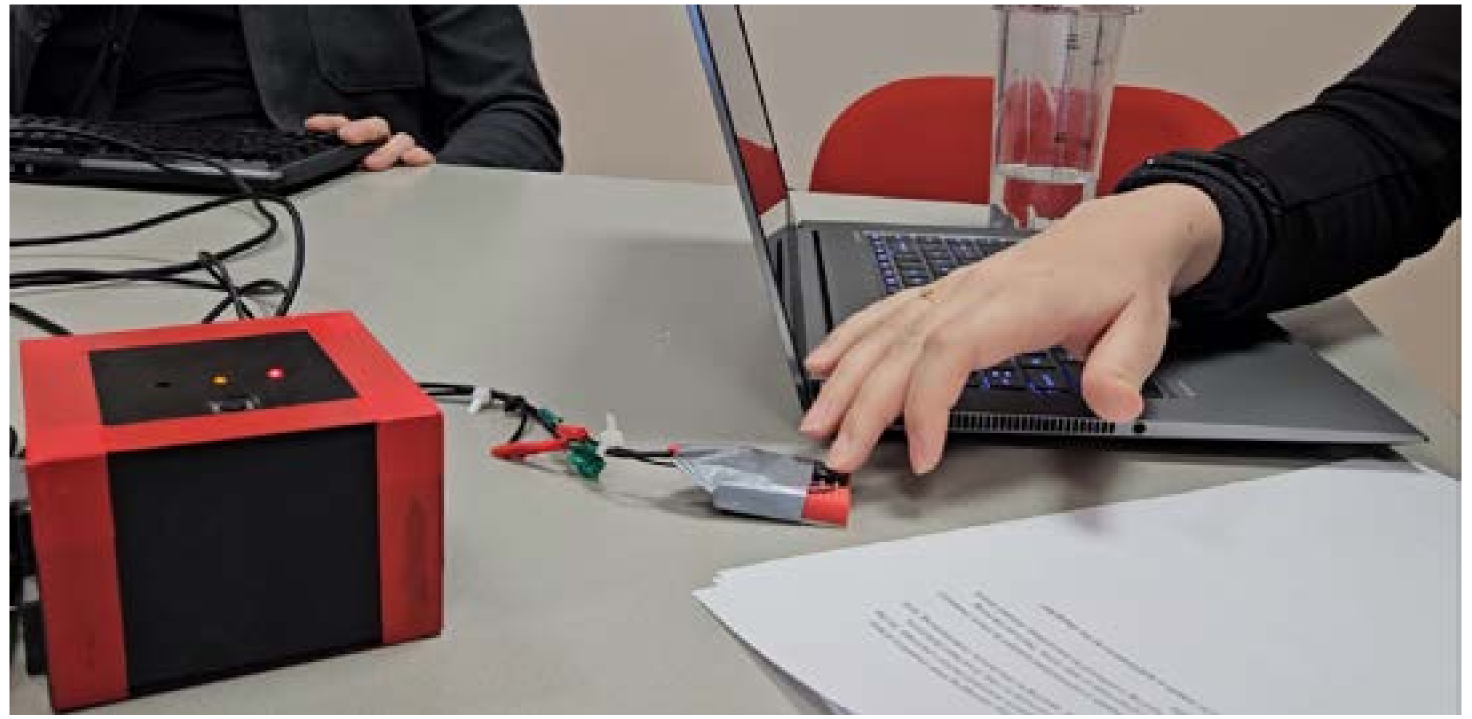


Figure 45 Use of marker prototype in the created test setting.



6. IMPLEMENTATION ROADMAP

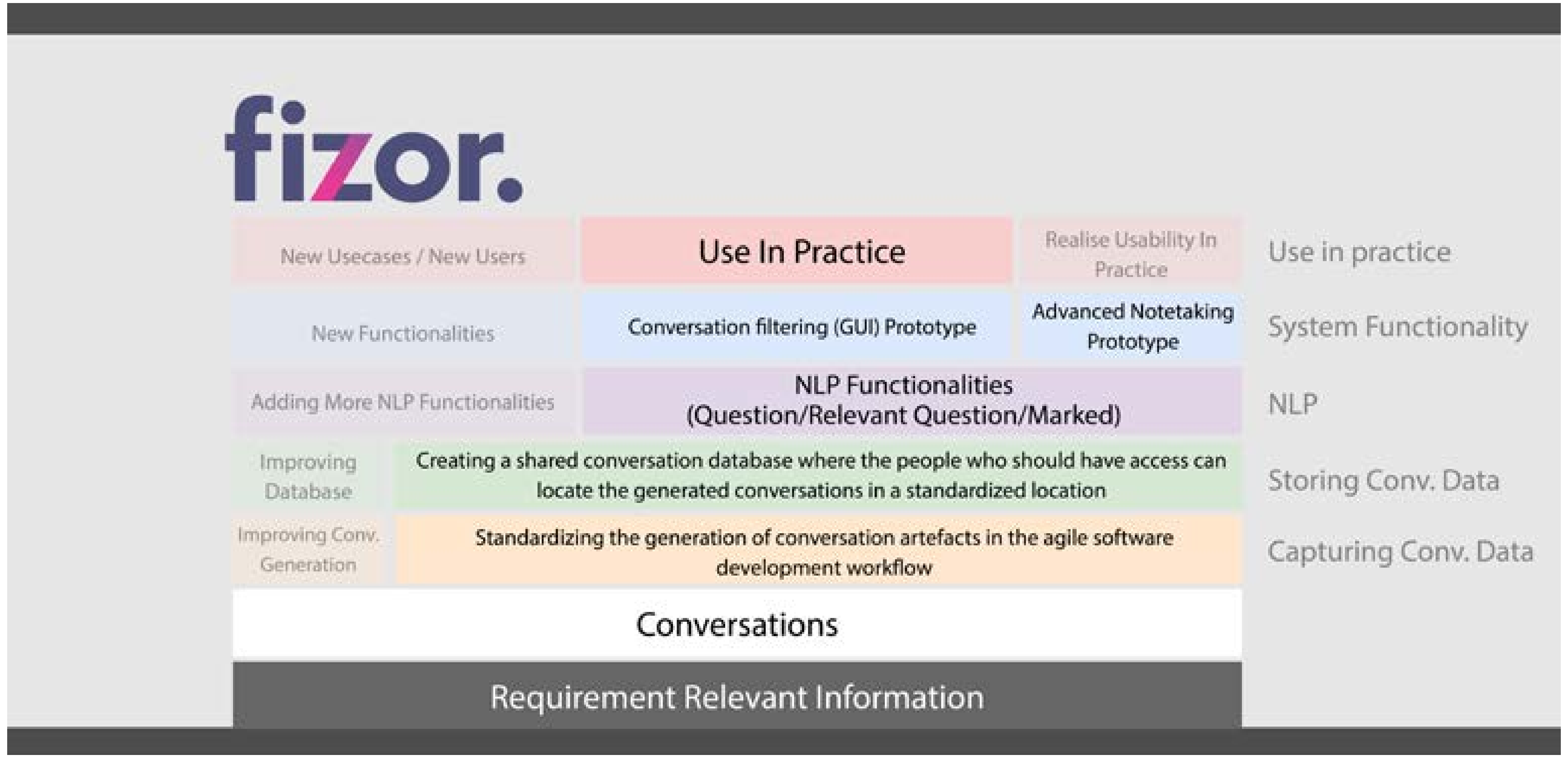

fizor.
New Usecases / New Users
Use In Practice
Realise Usability In Practice
Use in practice
New Functionalities
Conversation filtering (GUI) Prototype
Advanced Notetaking Prototype
System Functionality
Adding More NLP Functionalities
NLP Functionalities (Question/Relevant Question/Marked)
NLP
Improving Database
Creating a shared conversation database where the people who should have access can locate the generated conversations in a standardized location
Storing Conv. Data
Improving Conv. Generation
Standardizing the generation of conversation artefacts in the agile software development workflow
Capturing Conv. Data
Conversations
Requirement Relevant Information

Figure 46: Implementation and further research and development of the designed system.

6. IMPLEMENTATION ROADMAP

Implementing an automated conversation capture and processing system holds significant promise for organizations, especially those heavily reliant on conversations to gather requirement-relevant information. While the system prototype still requires further validation and development, it is ready for internal implementation at Fizor. By integrating the system into our workflows and continuously evaluating its performance, functionalities can be refined over time. This iterative process, informed by a growing knowledge base, will enable us to enhance different aspects of the system while preserving its core functionality.

6.1 Implementation step 1: Standardizing Layers One and Two (Capturing & Storing)

The initial phase of implementation focuses on establishing a robust foundation for capturing and storing conversation data effectively. This entails building upon the requirement-relevant information elicited from conversations. It is essential to prioritize the implementation of the first two layers, which involve accurately capturing conversations and storing them in a centralized conversation database. Without access to accurately processed conversation data, the other functionalities of the system cannot operate optimally.

Fortunately, Fizor already possesses a dedicated meeting room equipped with high-quality recording equipment and a tailored MS Teams meeting room setup. Assuming that the training of MS Teams' speech model ensures the necessary accuracy in conversation data capture, generating conversation artifacts merely requires users to activate the record button during online conversations. Additionally, the inclusion of a smart speaker, coupled with the setup of voice profiles for each participant, facilitates the accurate capture of face-to-face conversations held in the meeting room.

However, capturing conversations alone does not suffice to provide value; the captured data must be accessible to relevant stakeholders. Thus, it is imperative to establish a standard for storing conversations at consistent, shared locations accessible to authorized individuals. Furthermore, linking various types of data to the conversation artifact, such as recognizable names, descriptions, participants, and dates, could enhance accessibility and usability (database concept of figure 15). It provides an overview of this standardized approach. While MS Teams already shares a copy of the meeting recording with all participants, creating a standardized shared database ensures accessibility for individuals interested in the conversation but unable to participate. This includes those unable to attend due to illness or stakeholders like developers keen on staying informed about developments. Moreover, standardizing the storage of conversation data lays the necessary groundwork for subsequent implementation steps.

6.2 Implementation step 2: Utilizing Processed Conversation Data

Access to automatically generated high-accuracy conversation artifacts holds intrinsic value by eliminating the need for manual transcription, thus providing stakeholders with comprehensive access to the entire conversation without additional time investment. This accessibility facilitates retrospective review of discussions, enabling stakeholders to revisit and reflect on conversations without delay. Moreover, diverse conversation types, such as brainstorms, can be effectively captured, preserving ideas and considerations articulated during meetings for future reference and communication.

Additionally, the groundwork has been laid to integrate the system design into the manual processing of conversations into requirements. Initially, the system design will be distributed to 10-20 Business Analysts, a process requiring minimal resources, including USB drives and a few minutes for duplication. Over the course of several months, analysts will utilize the system in various scenarios, documenting their experiences through questionnaires. These questionnaires will capture feedback on usability, performance, and any encountered limitations or missing functionalities. Subsequently, the collected feedback will undergo evaluation to inform future development endeavors.

6.3 Implementation step 3: Use Evaluation

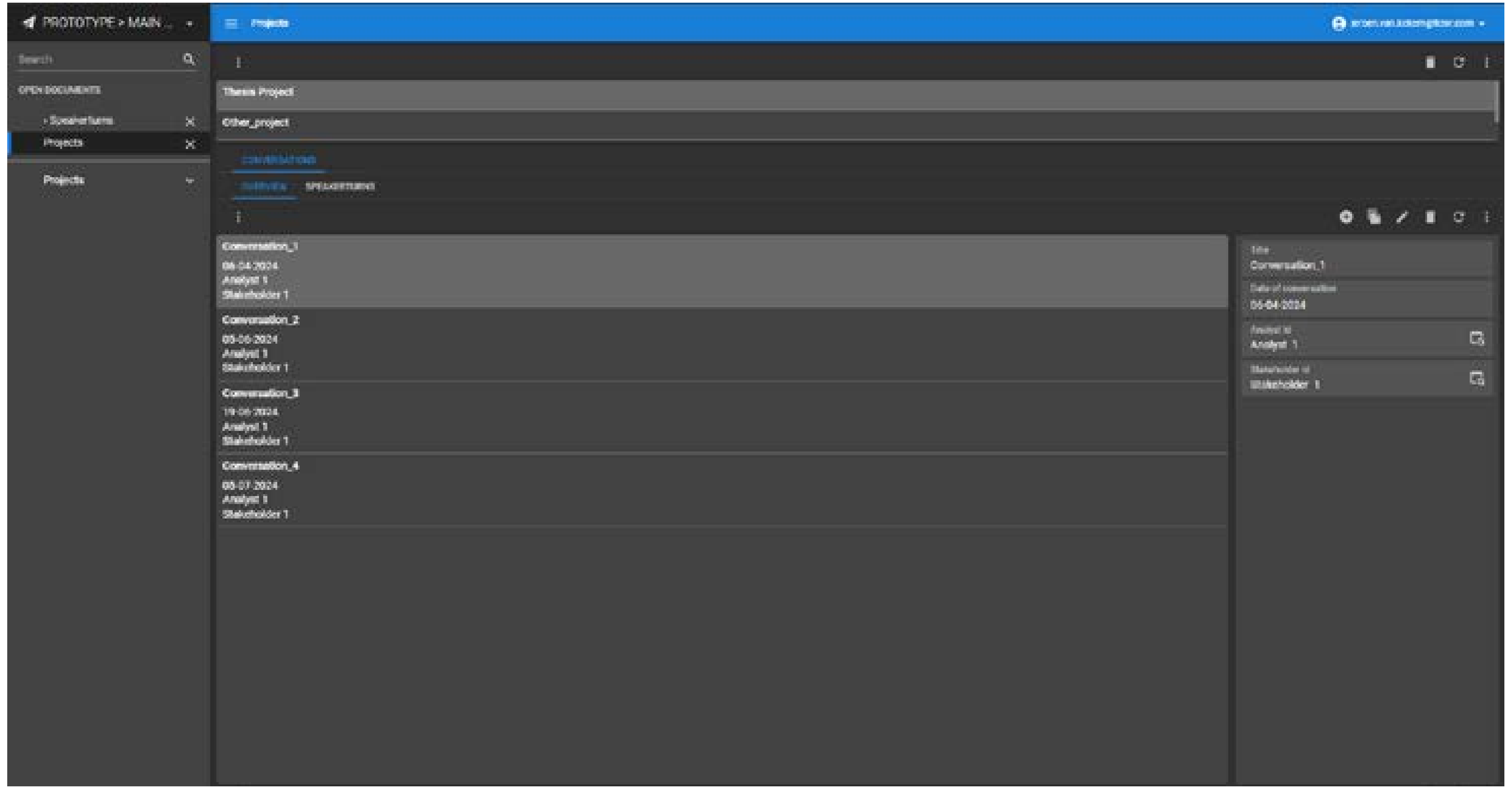
Following the conclusion of the test period, users will undergo interviews aimed at evaluating the collected usage data. Through these interviews, users will be prompted to provide detailed feedback, elucidating any encountered issues, positive experiences, or suggestions for improvement. This feedback will serve to establish a comprehensive knowledge base of user requirements and enhancement opportunities. Subsequently, research and development initiatives will be initiated to enhance various layers of the system based on insights gleaned from the user evaluation process.

6.4 Implementation step 4: Research and development

The improvement journey begins by leveraging the insights gathered from user feedback and evaluation to refine the system's performance and functionality. The modular nature of the system allows for targeted enhancements to individual layers without necessitating a complete overhaul, ensuring seamless integration and continuity of operations. Incremental improvements pave the way for sustained progress over time, with foundational advancements often unlocking new possibilities in higher layers.

***Conversation Artefact Generation* –** The quest for accurately capturing conversations across diverse settings remains an ongoing pursuit, necessitating continued research and development efforts. Evaluating the performance of speech conversion using the trained speech model of MS Teams represents a crucial step in validating its efficacy. Through systematic evaluation of generated conversation artefacts and iterative model refinement, strides can be made towards achieving higher levels of accuracy and reliability. Additionally, identifying and rectifying incorrectly converted words offers a pathway to enhancing conversion accuracy. By curating a comprehensive phrase list that encompasses names of frequently encountered companies and individuals, domain-specific terminology related to Low Code practices, processes, and other IT jargon, conversion accuracy can be bolstered. This list serves as a foundational resource that evolves over time, reflecting the dynamic nature of language and communication

6. IMPLEMENTATION ROADMAP



Thesis Project
Other_project

Conversation_1
06-04-2024
Analyst 1
Stakeholder 1

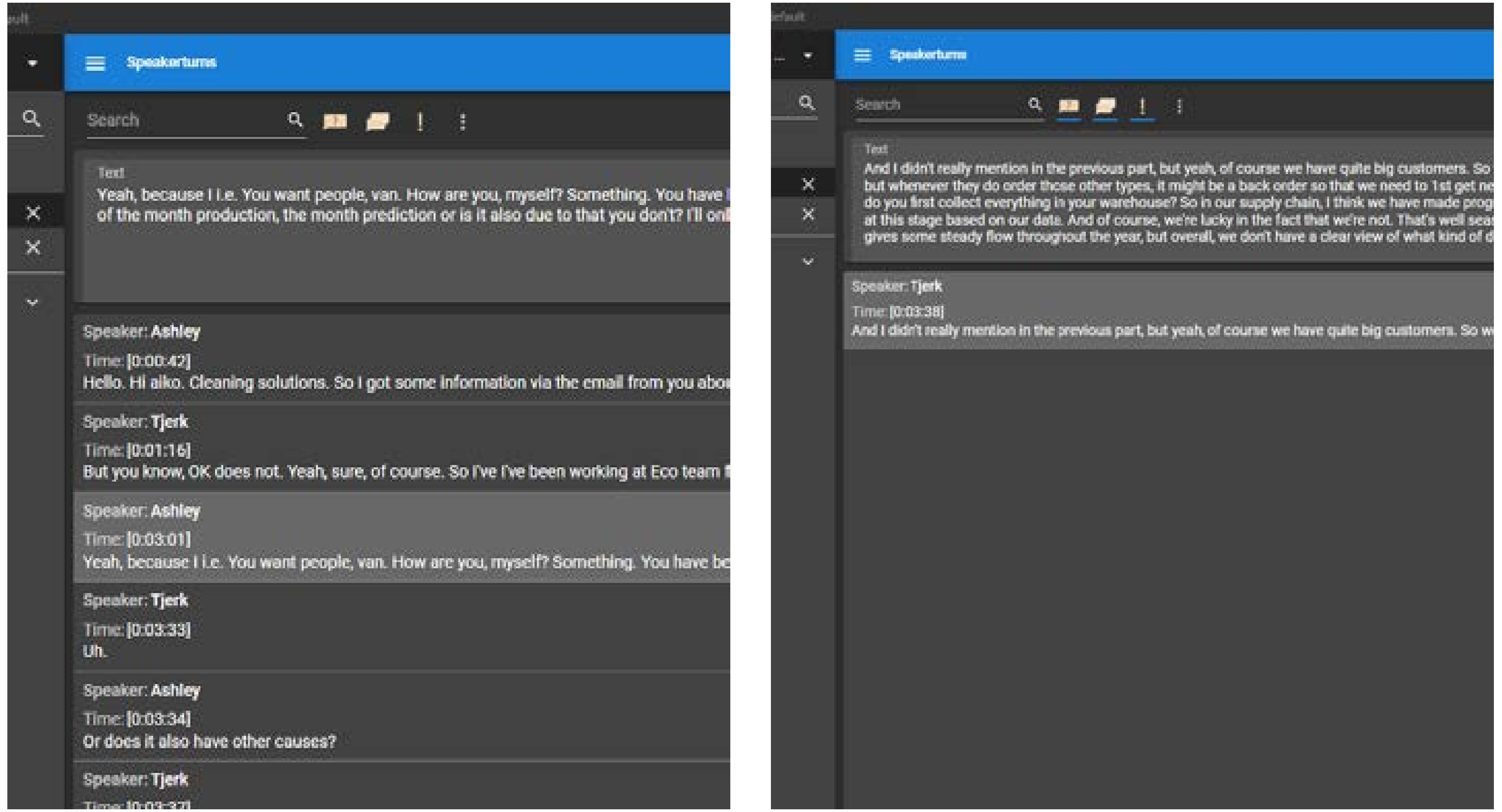
Conversation_2
05-06-2024
Analyst 1
Stakeholder 1

Conversation_3
19-06-2024
Analyst 1
Stakeholder 1

Conversation_4
05-07-2024
Analyst 1
Stakeholder 1

Title
Conversation_1
Date of conversation
06-04-2024
Analyst id
Analyst 1
Stakeholder id
Stakeholder 1

Figure 47: GUI functionalities realized in Low Code Development Platform Thinkwise.



Text
Yeah, because I i.e. You want people, van. How are you, myself? Something. You have
of the month production, the month prediction or is it also due to that you don't? I'll on

Speaker: Ashley
Time: [0:00:42]
Hello. Hi aiko. Cleaning solutions. So I got some information via the email from you abo

Speaker: Tjerk
Time: [0:01:16]
But you know, OK does not. Yeah, sure, of course. So I've I've been working at Eco team

Speaker: Ashley
Time: [0:03:01]
Yeah, because I i.e. You want people, van. How are you, myself? Something. You have be

Speaker: Tjerk
Time: [0:03:33]
Uh.

Speaker: Ashley
Time: [0:03:34]
Or does it also have other causes?

Speaker: Tjerk
Time: [0:03:37]

Text
And I didn't really mention in the previous part, but yeah, of course we have quite big customers. So
but whenever they do order those other types, it might be a back order so that we need to 1st get ne
do you first collect everything in your warehouse? So in our supply chain, I think we have made prog
at this stage based on our data. And of course, we're lucky in the fact that we're not. That's well sea
gives some steady flow throughout the year, but overall, we don't have a clear view of what kind of d

Speaker: Tjerk
Time: [0:03:38]
And I didn't really mention in the previous part, but yeah, of course we have quite big customers. So w

Figure 48: GUI functionalities realized in Low Code Development Platform Thinkwise.

within Fizor's ecosystem.

Anticipating advancements in speech-to-text functionalities, the prospect of future innovations holds promise for broader applicability. The emergence of new products facilitating seamless integration with platforms like MS Teams presents opportunities to expand the scope of conversation artefact generation. Technological breakthroughs enabling the accurate separation of speaker turns in various conversation settings could revolutionize artefact generation, paving the way for enhanced versatility and utility.

Moreover, exploring methods for recognizing speakers in portable solutions and enabling real-time interaction with speaker turn data represents fertile ground for further inquiry. By delving into these areas, Fizor can stay at the forefront of technological innovation, driving advancements that empower more effective communication and collaboration.

NLP Functionalities – Expanding the repertoire of annotation functionalities within the system represents a pivotal avenue for enriching its capabilities. Leveraging the existing integration of the StanfordCoreNLP system provides a solid foundation for incorporating additional annotations, such as tokenization, into the system's framework. By harnessing these annotations, synergies with advanced tools like Trace2Conv can be explored, opening avenues for enhanced functionality and utility.

The integration of more annotations not only broadens the system's feature set but also unlocks new possibilities for data analysis and manipulation. For instance, enabling finer-grained search capabilities and more granular conversation filtering can empower users to extract deeper insights from conversation data. By leveraging NLP outputs to analyze system performance, valuable insights can be gleaned, driving iterative improvements and fueling further research in this domain.

Moreover, the exploration of NLP outputs holds potential to catalyze research endeavors aimed at advancing the understanding of conversational dynamics and linguistic patterns. Through systematic analysis of NLP-generated outputs, trends, correlations, and anomalies can be identified, paving the way for innovative research initiatives and breakthroughs in natural language processing.

By prioritizing the enhancement of NLP functionalities, Fizor can augment the sophistication and effectiveness of its conversation processing capabilities, ultimately empowering users with more robust tools for information extraction, analysis, and decision-making.

Database – Expanding the capabilities of the database management system represents a critical step towards enhancing the system's usability and scalability. Presently, the system's database structure lacks flexibility, as it only supports a single project folder without provisions for additional folder creation or organization within the GUI. To address this limitation, the system should be augmented with comprehensive file management functionalities, empowering users to create, edit, delete, and move folders to suit their organizational needs.

By enabling users to establish a hierarchical file structure within the system, they can effectively organize and manage conversation data according to project, department, or any other relevant categorization scheme. This enhanced flexibility not only streamlines data management processes but also fosters greater user autonomy and efficiency.

Furthermore, to facilitate seamless integration with the existing infrastructure of organizations, the database should incorporate capabilities for linking to cloud storage platforms. By leveraging APIs and integration frameworks, such as those provided by leading cloud service providers, the system can enable direct storage of conversation data in shared cloud repositories. This integration not only enhances data accessibility and collaboration but also ensures data security and compliance with organizational policies.

Also, to enable functionalities of, for example Trace2Conv, the system must enable more NL artefacts to be stored and linked to specific projects and conversations such as requirement artefacts.

Interaction Systems – Recognizing the prototype nature of the system's graphical user interface (GUI), there's a clear imperative for further development to elevate its effectiveness, efficiency, and user-friendliness. Leveraging Fizor's proficiency in Low Code Development (LCD), platforms like Thinkwise offer distinct advantages. By harnessing existing platform knowledge, development efforts become more streamlined, minimizing the need for extensive coding and accelerating time-to-market.

Figure 72 serves as a testament that the interface can be swiftly replicated in Thinkwise within a day, realizing the ability to navigate and filter conversations. This demonstrates the platform's agility and adaptability to Fizor's requirements. However, to ensure seamless integration between the GUI and the underlying processing system, it's essential to establish an API interface that facilitates bidirectional communication.

By establishing an API interface, both the GUI and the processing system can evolve independently while preserving interoperability and system functionality. For instance, the GUI seamlessly integrates with Excel files, facilitating streamlined data communication from the system design output. This interoperability ensures data consistency and integrity across different components of the system architecture.

Furthermore, implementing numerous UI enhancements can significantly enhance the overall user experience. Elements such as tooltips, icons, and informative messages play a pivotal role in enhancing clarity and providing guidance to users, particularly in error scenarios. These enhancements contribute to a more intuitive and user-friendly interface, empowering users to navigate the system with confidence and efficiency.
Moreover, enforcing checks to ensure correct data input is crucial for maintaining data accuracy and integrity. For example, mandating the use of only MS Teams conversation artifacts helps prevent formatting discrepancies that may disrupt Natural Language Processing (NLP) functionality. By enforcing data input standards and error handling mechanisms, the system enhances reliability and consistency in processing conversation data.

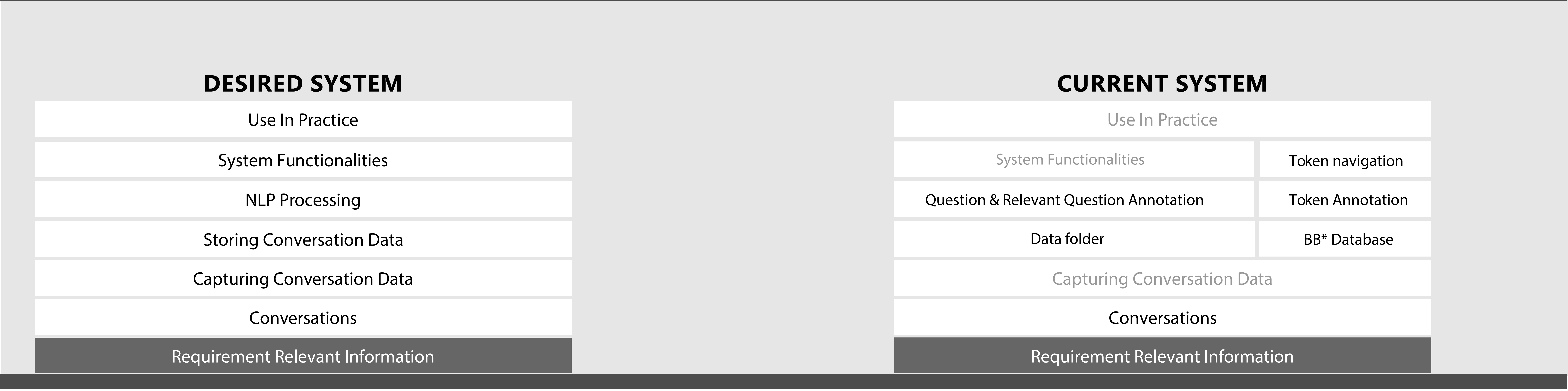


Figure 49: Thesis results presented in sequence.

REALIZED SYSTEM

Use In Practice		Use in practice
Conversation filtering (GUI) Prototype	Advanced Notetaking Prototype	System Functionality
Question & Relevant Question Annotation	Importance Annotation	NLP
USB Portable Database		Storing Conv. Data
MS Teams Speech-To-Text (Base Model)		Capturing Conv. Data
Conversations		
Requirement Relevant Information		

IMPLEMENTATION & FURTHER RESEARCH AND DEVELOPMENT

New Usecases / New Users	Use In Practice	Realise Usability In Practice
New Functionalities	Conversation filtering (GUI) Prototype	Advanced Notetaking Prototype
Adding More NLP Functionalities	NLP Functionalities (Question/Relevant Question/Marked)	
Improving Database	Creating a shared conversation database where the people who should have access can locate the generated conversations in a standardized location	
Improving Conv. Generation	Standardizing the generation of conversation artefacts in the agile software development workflow	
Conversations		
Requirement Relevant Information		

7. CONCLUSION & EVALUATION

This thesis has produced a prototype that has the potential to be useful for Business Analysts in the activity of processing captured elicitation data into structured requirements. While it may not always be of use, in longer real-life conversations the ability to filter conversations to search for specific conversation data has the potential to be valuable for a Business Analyst (Spijkman et al., 2023).

The initial problem statement has been translated to a desired system consisting of multiple system layers that build on elicitation interviews used to elicit requirements and ends in a use case to use this (processed) data (figure 49). The initial system has been analysed and with this initial system a system prototype has been realized.

The main functionality on system level that has been tried to realize is to capture all conversation data generated during the elicitation interview (so no conversation data is lost) and allowing a Business Analyst to locate requirement relevant information in this conversation dataset using filters.

The prototype that has been realized is a system that consists of multiple subsystems labeled 'MS Teams, NLP, GUI, and Marker', each with their own behaviour and components, that together achieve the main functionality of the system.

The components of REConSum have been used as start of the system design and have been integrated into one system prototype that can be used through a GUI. This removed the need for installing python and the desired packages, writing code that runs the algorithms of REConSum, and manually downloading and installing of StanfordCoreNLP (a system necessary for functioning, requiring the downloading and installing of java) and running it manually. This solution was necessary as the knowledge and time needed to run REConSum would ask to much of any user.

A solution has been provided for users to interact with the outputs that are being generated by the system through the design of the system GUI, storing the system outputs in excel files, and realizing the systematically accessing and storing of information.

To create a seamless transition between the MS Teams Conversation Artefact output and the expected input an import function has been integrated that performs all necessary steps to format, process and view the conversation in the background. This in total takes 5-10 minutes in total from start to end, and takes 15 minutes of processing time after which the conversation can be directly used by the Business Analyst.

Within the scope of this thesis only the systems functionality could be realized and validated. The system will most definetley need to see improvements to fit more specific user requirements for example. With how the system is designed future research questions can be tailored to test the effectiveness, efficiency, and user friendlyness in regards to capturing, processing, and using requirement relevant information from conversation.

Within the scope of this thesis conversation artefact with enough accuracy could be realized. Furthermore, in face to face conversations, speakerturns are formulated as required because of how MS Teams functions. Solutions with high chance to solving these problems have been provided in a implementation plan that provides guidelines for further research and development and how to implement the prototype within the workflow of Fizor as a first step. This implementation provides initial requirements of implementing and validating the method to accurately capture requirement artefacts. It also provides for each part of the system suggestions for further research and development.

This prototype is not a finished product but should be considered as a tool designed for Fizor to get useful feedback from users to be used in further research and development while minimally disrupting the current workflow of Business Analysts.

8. REFERENCES

Al-Saqqa, S., Sawalha, S., & Abdelnabi, H. (2020). Agile software development: Methodologies and trends. International Journal of Interactive Mobile Technologies, 14(11). https://doi.org/10.3991/ijim.v14i11.13269

Bano, M., Zowghi, D., Ferrari, A., Spoletini, P., & Donati, B. (2019). Teaching requirements elicitation interviews: an empirical study of learning from mistakes. Requirements Engineering, 24(3). https://doi.org/10.1007/s00766-019-00313-0

Benyon, D., Turner, P., & Turner, S. (2005). Designing interactive systems: People, Activities, Contexts, Technologies. Pearson Education.

Berrezueta, S. (2023). Proceedings of the 18th Latin American Conference on Learning Technologies (LACLO 2023). Springer.

Cambridge University Press & Assessment. (2024). Cambridge Dictionary. https://dictionary.cambridge.org/dictionary/english/user-experience

Carrier, C. A., & Titus, A. (1979). The effects of notetaking: A review of studies. In Contemporary Educational Psychology (Vol. 4, Issue 4). https://doi.org/10.1016/0361-476X(79)90050-X

Davey, B., & Cope, C. (2008). Requirements Elicitation – What's Missing? Issues in Informing Science and Information Technology, 5. https://doi.org/10.28945/1027

Davey, B., & Cope, C. (2009). Consultants' experience of requirements elicitation conversations - An empirical model. 17th European Conference on Information Systems, ECIS 2009.

Davis, A., Dieste, O., Hickey, A., Juristo, N., & Moreno, A. M. (2006). Effectiveness of requirements elicitation techniques: Empirical results derived from a systematic review. Proceedings of the IEEE International Conference on Requirements Engineering. https://doi.org/10.1109/RE.2006.17

Davis, C. J., Fuller, R. M., Tremblay, M. C., & Berndt, D. J. (2006). Communication challenges in requirements elicitation and the use of the Repertory Grid technique. Journal of Computer Information Systems, 46(5 SPEC. ISS.). https://doi.org/10.1080/08874417.2006.11645926
E
ger, A., Bonnema, M., & Lutters, E. (2012). Product Design (1st ed.). Eleven.

Ferrari, A., Spoletini, P., & Gnesi, S. (2016). Ambiguity and tacit knowledge in requirements elicitation interviews. Requirements Engineering, 21(3). https://doi.org/10.1007/s00766-016-0249-3

Hirschberg, J., & Manning, C. D. (2015). Advances in natural language processing. In Science (Vol. 349, Issue 6245). https://doi.org/10.1126/science.aaa8685

Johnson, J. (2021). Designing with the Mind in Mind. In C. Hockaday (Ed.), Designing with the Mind in Mind (3rd ed.). Morgan Kaufmann Publishers.

Khurana, D., Koli, A., Khatter, K., & Singh, S. (2023). Natural language processing: state of the art, current trends and challenges. Multimedia Tools and Applications, 82(3). https://doi.org/10.1007/s11042-022-13428-4

Leary, H., & West, R. (2023). Foundations of Learning and Instructional Design Technology. In Foundations of Learning and Instructional Design Technology. https://doi.org/10.59668/473

Lucassen, G., Dalpiaz, F., van der Werf, J. M. E. M., & Brinkkemper, S. (2016). The use and effectiveness of user stories in practice. Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), 9619. https://doi.org/10.1007/978-3-319-30282-9_14

Meth, H., Brhel, M., & Maedche, A. (2013). The state of the art in automated requirements elicitation. In Information and Software Technology (Vol. 55, Issue 10). https://doi.org/10.1016/j.infsof.2013.03.008

Microsoft Custom Speech. (n.d.). Retrieved April 26, 2024, from https://speech.microsoft.com/portal/customspeech/overview

Microsoft Graph API. (n.d.). Retrieved April 26, 2024, from https://devblogs.microsoft.com/microsoft365dev/microsoft-graph-apis-for-microsoft-teams-meeting-transcripts-now-generally-available/

Mohammad Rajabali Nejad. (2020). Safety by Design - Engineering Products and Systems. SafetyCube.com.

Project Management Institute. (2016). Requirements management : a practice guide.
Redmond, W. (2019, April 24). Microsoft: Earnings Release FY19 Q3. https://www.microsoft.com/en-us/Investor/earnings/FY-2019-Q3/press-release-webcast
reMarkable. (n.d.). Retrieved April 26, 2024, from https://remarkable.com/?utm_source=google&utm_medium=paid&utm_campaign=brand&gad_source=1&gclid=CjwKCAjwoa2xBhACEiwA1sb1BGswUuieTBCgpc2DgbA4VeFKb-3WGZMk22XygB9pp3UF-lokj0UScxoCu5cQAvD_BwE

Speech SDK. (n.d.). Retrieved April 26, 2024, from https://learn.microsoft.com/en-us/azure/ai-services/speech-service/speech-sdk

Speech Studio. (n.d.). Azure Cognitive Services Speech. Retrieved January 10, 2024, from Azure Cognitive Services Speech

Spijkman, T., Dalpiaz, F., & Brinkkemper, S. (2022). Back to the Roots: Linking User Stories to Requirements Elicitation Conversations. Proceedings of the IEEE International Conference on Requirements Engineering, 2022-August. https://doi.org/10.1109/RE54965.2022.00042

Spijkman, T., de Bondt, X., Dalpiaz, F., & Brinkkemper, S. (2023). Summarization of Elicitation Conversations to Locate Requirements-Relevant Information (pp. 122–139). https://doi.org/10.1007/978-3-031-29786-1_9

Sujay Vailshery, L. (2024a, February 5). Microsoft Teams: number of daily active users 2019-2023. https://www.statista.com/statistics/1033742/worldwide-microsoft-teams-daily-and-monthly-users/

Sujay Vailshery, L. (2024b, October 10). Statista: Global market share of videoconferencing software 2023. https://www.statista.com/statistics/1331323/videoconferencing-market-share/

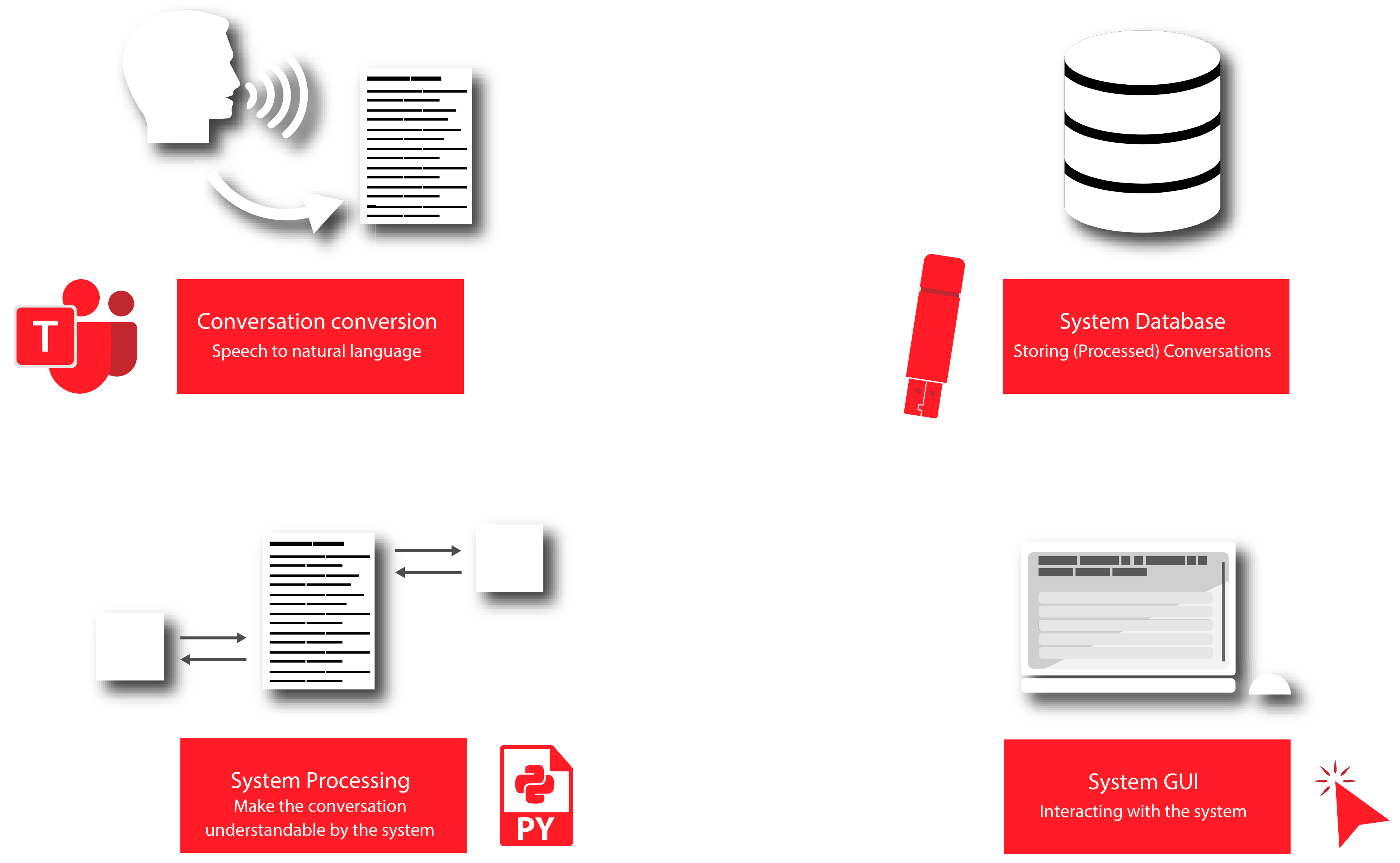
Sutcliffe, A., & Sawyer, P. (2013). Requirements elicitation: Towards the unknown unknowns. 2013 21st IEEE International Requirements Engineering Conference, RE 2013 - Proceedings. https://doi.org/10.1109/RE.2013.6636709

tkinter — Python interface to Tcl/Tk. (n.d.). Retrieved April 26, 2024, from https://docs.python.org/3/library/tkinter.html

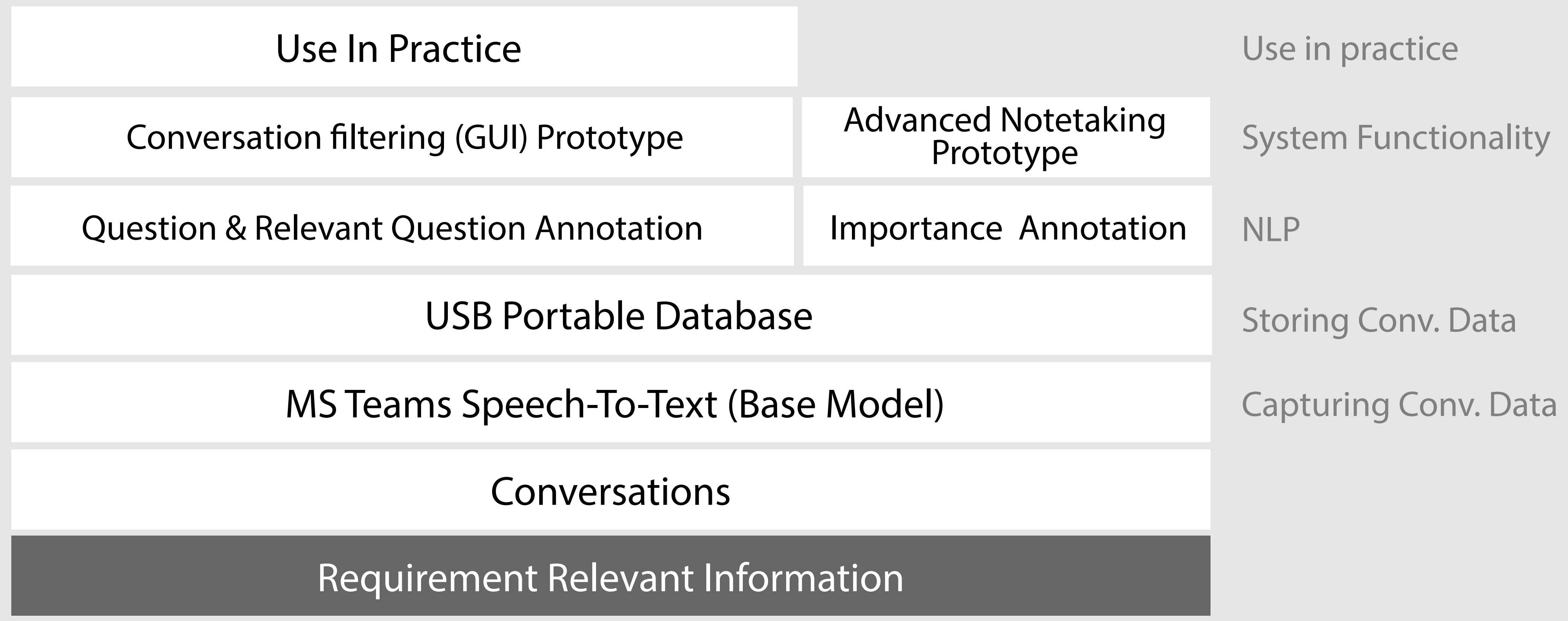
Waszkowski, R. (2019). Low-code platform for automating business processes in manufacturing. IFAC-PapersOnLine, 52(10). https://doi.org/10.1016/j.ifacol.2019.10.060

Xu, B., Tao, C., Feng, Z., Raqui, Y., & Ranwez, S. (2021). A Benchmarking on Cloud based Speech-To-Text Services for French Speech and Background Noise Effect. http://arxiv.org/abs/2105.03409

9. APPENDIX 0: REALIZED SYSTEM DEVELOPMENT MANUAL



Conversation conversion
Speech to natural language
System Database
Storing (Processed) Conversations
System Processing
Make the conversation understandable by the system
PY
System GUI
Interacting with the system



Use In Practice
Conversation filtering (GUI) Prototype
Advanced Notetaking Prototype
Question & Relevant Question Annotation
Importance Annotation
USB Portable Database
MS Teams Speech-To-Text (Base Model)
Conversations
Requirement Relevant Information
Use in practice
System Functionality
NLP
Storing Conv. Data
Capturing Conv. Data

Figure 22 & 23: Realized System, presenting the realized solution on each 'foundational layer'.

APPENDIX 0: REALIZED SYSTEM DEVELOPMENT MANUAL

This chapter provides an in-depth exploration of the system prototype developed to address the identified problem. Serving as an initial stride in the development journey, this prototype aims to generate conversation artefacts from natural language, process them, and store them in a central database for utilization across the agile software development process. The primary objective of this prototype is to furnish a functional model capable of practical application, facilitating user feedback collection to enhance various facets of the system.

The chapter is organized around the realization of four subsystems, as delineated in the problem analysis. Subchapter System 1 delves into the rationale and methodology behind harnessing Microsoft Teams to generate conversation artefacts. System 2 provides an intricate exposition of the graphical user interface (GUI) developed to enable the desired user interactions. System 3 elucidates the natural language processing (NLP) functionalities incorporated into the system design and delineates their operational processes. System 4 delineates the mechanisms governing data storage and retrieval within the system design. Lastly, System 5 unveils a prototype design harmonizing with such a system, seamlessly integrating with the notetaking process.

System 1: Generating conversation artefacts

Microsoft Teams offers a unique capability to convert speech to text in real-time through Azure cognitive speech services (Speech Studio, n.d.), a feature not found in other conferencing software. This functionality, coupled with automatic linking of converted text to the respective speaker with timestamps (referred to as Speakerturns), enhances the efficiency of converting conversations into natural language (figure 25 presents an example of a generated speakerturn).

The utilization of MS Teams is illustrated in Figure 24, depicting its application in both online and face-to-face settings. However, it's essential to acknowledge certain limitations associated with MS Teams that were uncovered during testing the performance of the system. One such limitation pertains to the generation and sorting of speakerturns by the system. MS Teams generates speakerturns by associating transcribed text with the source of the audio, typically the MS Teams client from which the audio originates. Consequently, it's imperative for each user to be present in the MS Teams meeting using their respective MS Teams client to ensure accurate differentiation of speakerturns.

During the performance testing of the speech-to-text functionality (refer to Chapter 6 for validation details), a notable challenge arose in face-to-face conversations conducted using a single laptop with only one MS Teams client. In this scenario, only one speakerturn containing all conversation text was generated, highlighting the necessity for individual MS Teams clients for accurate speakerturn differentiation. While online conversations inherently overcome this limitation, face-to-face settings necessitated the development of an online conversation configuration. This configuration involved the use of two laptops, each equipped with its own MS Teams client, within a single meeting room. Additionally, to ensure proper linkage of speakerturns to respective individuals, a manual microphone mute button was integrated with the MS Teams client (refer to Figure 27).

Upon conclusion of a conversation or termination of the recording function, an audio/video file (.mp4) and its corresponding transcript (.docx) are automatically stored in the cloud environment known as MS Stream. This cloud environment facilitates easy access to these files, enabling users to download them effortlessly (refer to Figure 26). User interaction beyond activating the record and transcribe functions at the conversation's outset is minimal. Users simply select the desired conversation for processing and storage within the system. Subsequently, they can download the conversation to any location on their desktop, ensuring accessibility. With just a few clicks within the GUI, users can swiftly locate and process the chosen conversation, requiring only a few minutes.

This seamless transition is made possible through a designed algorithm. This algorithm effectively eliminates extraneous visual elements and superfluous text introduced by MS Teams. Additionally, it formats the set of speakerturns to align with the preprocessing (.py) algorithm of REConSum. Leveraging this algorithm as the foundational component of the system design facilitates the conversion of conversation artefacts into a DataFrame (data that the processing algorithms can interact with).

To ensure the usability of the system, several enhancements were incorporated into the conversation artefact generation process to align with the requirements of documenting speakerturns and capturing accurate requirement-relevant data. The efficacy of MS Teams' speech-to-text functionality was evaluated through various test conversations. However, the system in its current state cannot be deemed practical due to the inadequate accuracy of the conversation artefacts. In both online and face-to-face settings, factors such as unfamiliar words, pronunciation variations, and foreign language terms contribute to this accuracy challenge. Specifically, in face-to-face scenarios, hardware quality, microphone capabilities, room acoustics, background noise, and simultaneous conversations further exacerbate the issue.

To elevate the accuracy to usable levels within the broader system design, a crucial procedural step is necessary. This step involves comparing the conversation artefacts with the audio recordings of the conversations and making necessary corrections. However, this correction process is time-intensive, as elaborated upon in the validation chapter, rendering it impractical for direct implementation.

In instances where only one laptop and one MS Teams account are used during meeting recordings, MS Teams consolidates all conversation text into a single speakerturn (refer to the validation chapter). To address this limitation, a "push-to-talk" system was implemented. This solution entails the use of two laptops, each equipped with a microphone and a push button (as depicted in Figure 27). Both laptops share a single MS Teams account and are situated in the same meeting room, thereby facilitating accurate separation of speakerturns. Notably, the online setting already operates as intended, as all participants are inherently present in the virtual meeting room with their respective accounts.

The conversation artefact is retrieved from the Stream environment (refer to Figure 26) and can be stored anywhere on the Business Analyst's desktop or laptop. Subsequently, the user can locate the conversation artefact via the system's GUI (figure 28). The user has the option to assign a name to the conversation for easy identification within the system and provide names for each speaker. While the current prototype accommodates two speakers, this can be easily scaled to include more speakers in the future. This transition from MS Stream to system input is designed to require minimal effort, involving just a few clicks.

Upon pressing the "create" button, a formatting algorithm is triggered. This algorithm, which has been designed, executes two key functions. Firstly, it converts the file to a text format (.txt) while eliminating all unnecessary visual elements. The resulting output, referred to as "raw_transcript," is stored in the system for validation purposes (figure 30). Subsequently, the algorithm removes extraneous text introduced by Microsoft Teams, retaining only the speakerturns. This refined output is stored as "formatted transcript" within the system. Both outputs are provided in Appendix Chapter 10 for reference, these have been generated as part of a test case. As depicted in Figure 29, the conversation artefact supplied by MS Teams is a Word file (.docx) containing images and additional components, which are stripped away during the reformatting process.

APPENDIX 0: REALIZED SYSTEM DEVELOPMENT MANUAL

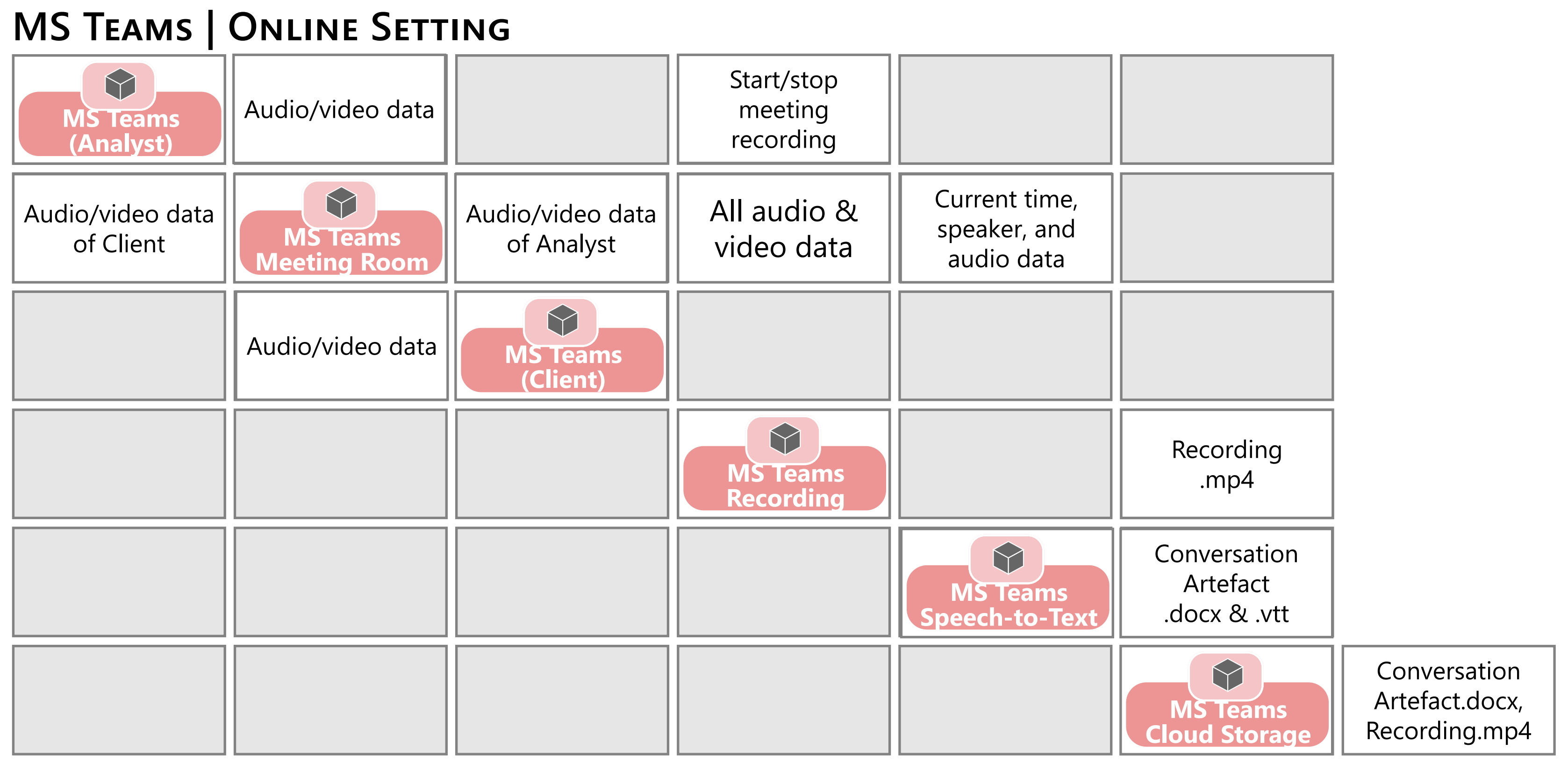
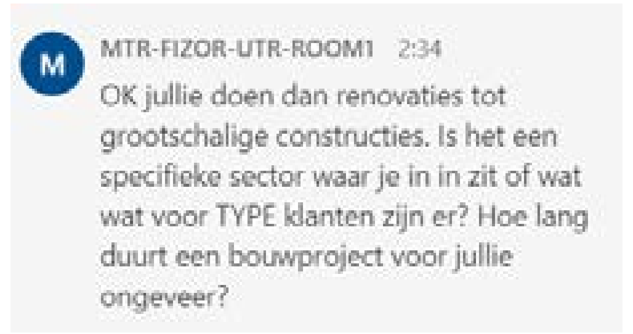

MS Teams | Online Setting
MS Teams (Analyst)
Audio/video data
Start/stop meeting recording
Audio/video data of Client
MS Teams Meeting Room
Audio/video data of Analyst
All audio & video data
Current time, speaker, and audio data
Audio/video data
MS Teams (Client)
MS Teams Recording
Recording .mp4
MS Teams Speech-to-Text
Conversation Artefact .docx & .vtt
MS Teams Cloud Storage
Conversation Artefact.docx, Recording.mp4

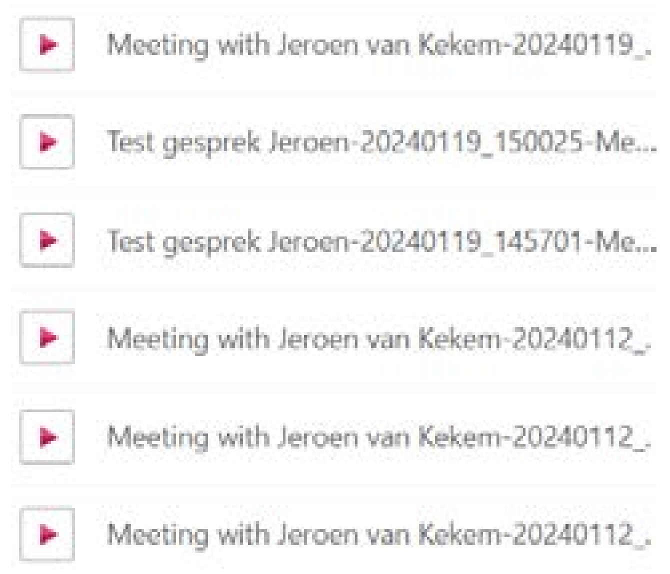
Figure 24: MS Teams functionality in online setting (left) and face to face setting (right).

ease.


MTR-FIZOR-UTR-ROOM1 2:34

OK jullie doen dan renovaties tot grootschalige constructies. Is het een specifieke sector waar je in in zit of wat wat voor TYPE klanten zijn er? Hoe lang duurt een bouwproject voor jullie ongeveer?

Figure 25: Example speakerturn from test conversation generated with MS Teams.


Meeting with Jeroen van Kekem-20240119_

Test gesprek Jeroen-20240119_150025-Me...

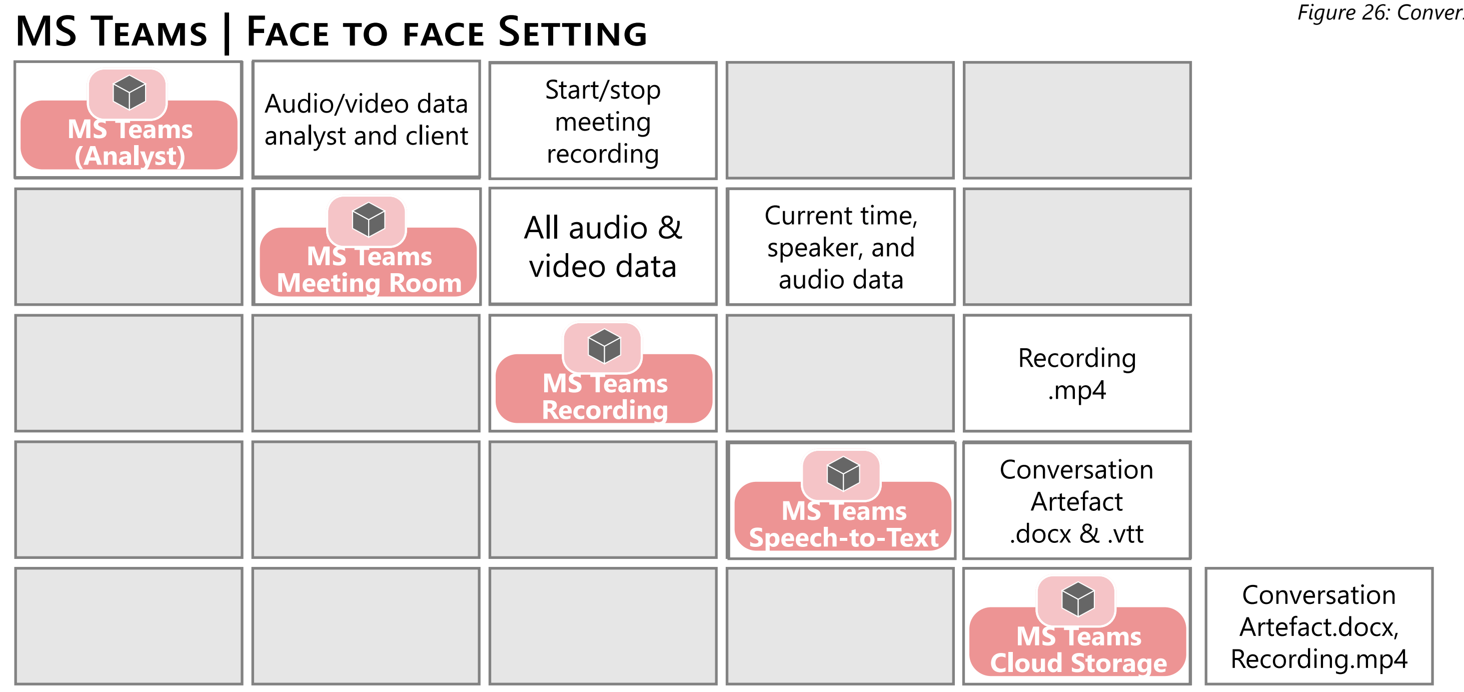
Test gesprek Jeroen-20240119_145701-Me...

Meeting with Jeroen van Kekem-20240112_

Meeting with Jeroen van Kekem-20240112_

Meeting with Jeroen van Kekem-20240112_

Figure 26: Conversations stored in the MS Stream environment.


MS Teams | Face to face Setting
MS Teams (Analyst)
Audio/video data analyst and client
Start/stop meeting recording
MS Teams Meeting Room
All audio & video data
Current time, speaker, and audio data
MS Teams Recording
Recording .mp4
MS Teams Speech-to-Text
Conversation Artefact .docx & .vtt
MS Teams Cloud Storage
Conversation Artefact.docx, Recording.mp4

APPENDIX 0: REALIZED SYSTEM DEVELOPMENT MANUAL

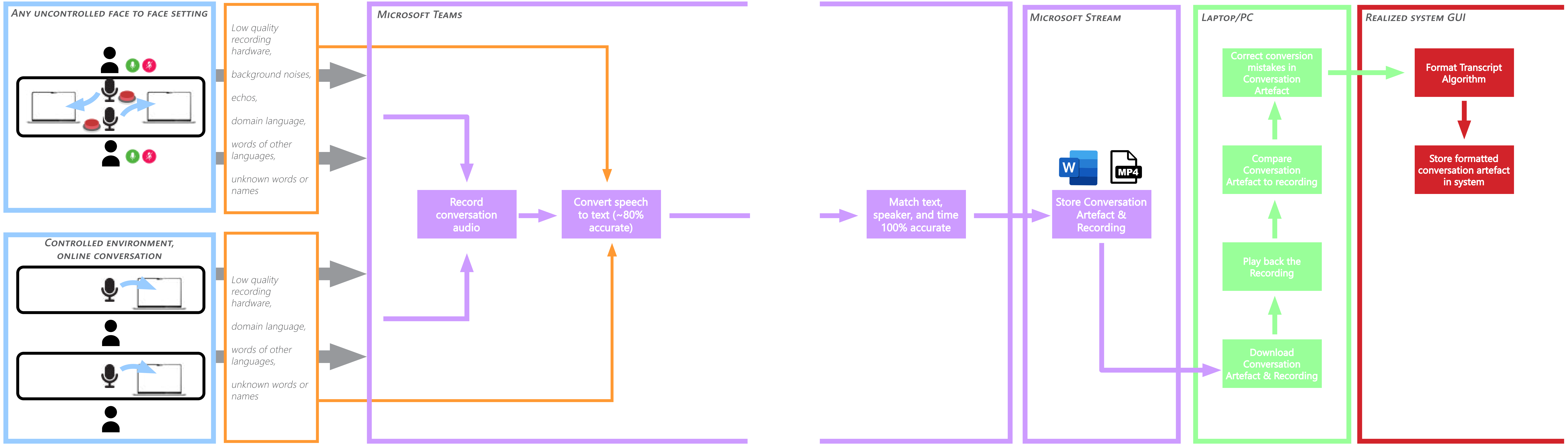
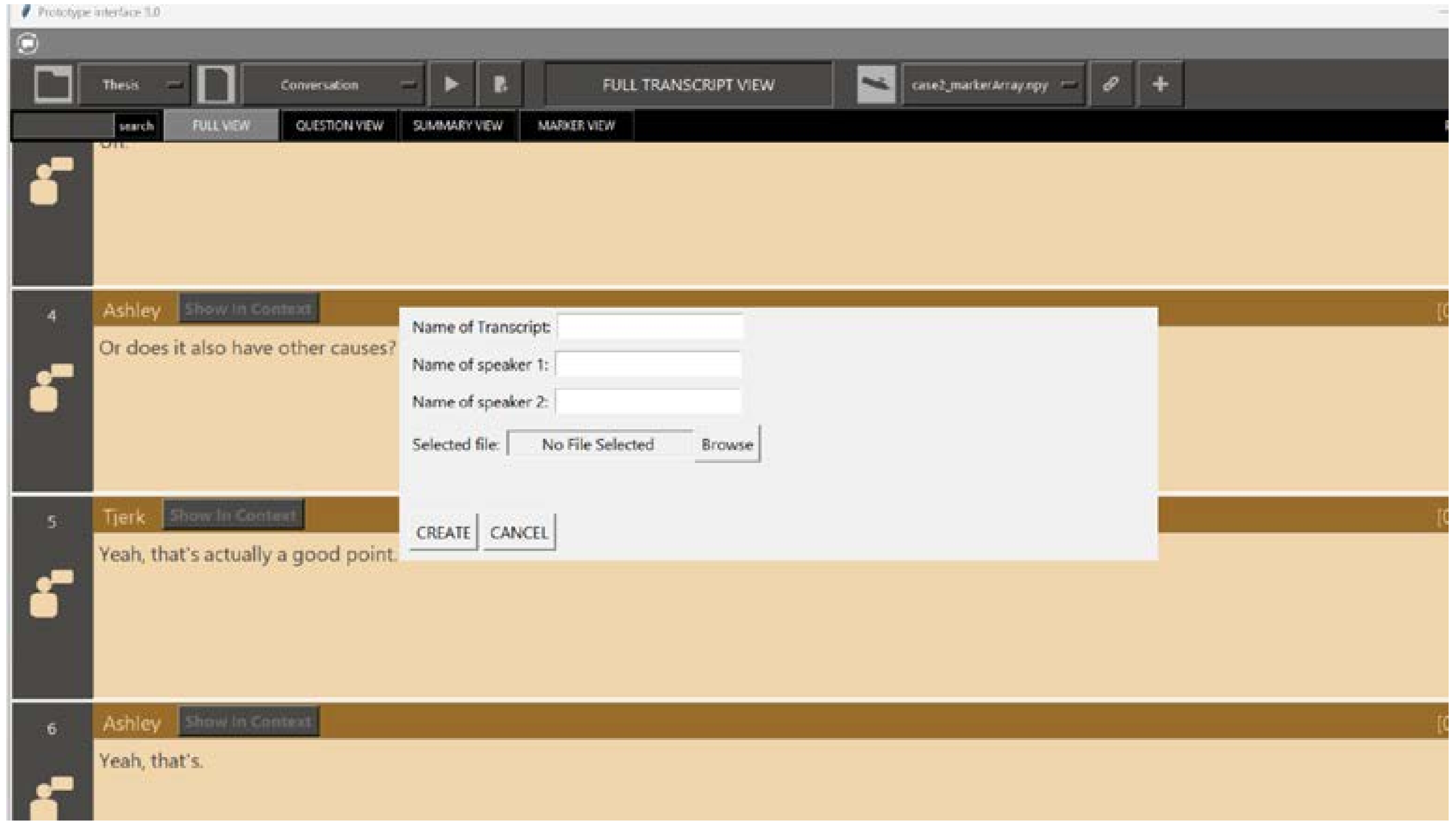

Any uncontrolled face to face setting
Controlled environment, online conversation
Low quality recording hardware,
background noises,
echos,
domain language,
words of other languages,
unknown words or names
Low quality recording hardware,
domain language,
words of other languages,
unknown words or names
Microsoft Teams
Record conversation audio
Convert speech to text (~80% accurate)
Match text, speaker, and time 100% accurate
Microsoft Stream
Store Conversation Artefact & Recording
Laptop/PC
Download Conversation Artefact & Recording
Play back the Recording
Compare Conversation Artefact to recording
Correct conversion mistakes in Conversation Artefact
Realized system GUI
Format Transcript Algorithm
Store formatted conversation artefact in system

Figure 27: Realized system to capture and process conversation artefacts.

APPENDIX 0: REALIZED SYSTEM DEVELOPMENT MANUAL



4 Ashley
Or does it also have other causes?

5 Tjerk
Yeah, that's actually a good point.

6 Ashley
Yeah, that's.

Name of Transcript:
Name of speaker 1:
Name of speaker 2:
Selected file: No File Selected

Figure 28: Within a couple of clicks the user can locate and import conversations from the Stream database.

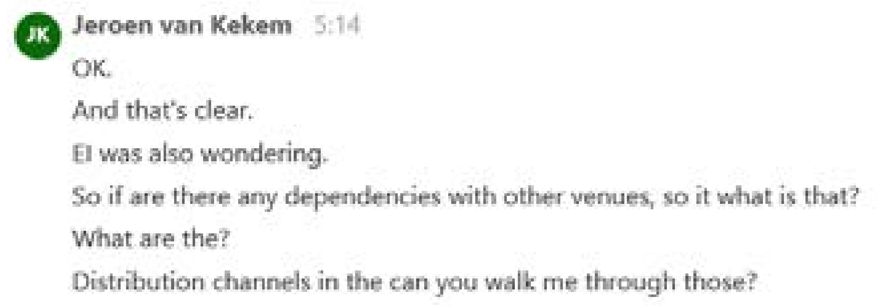
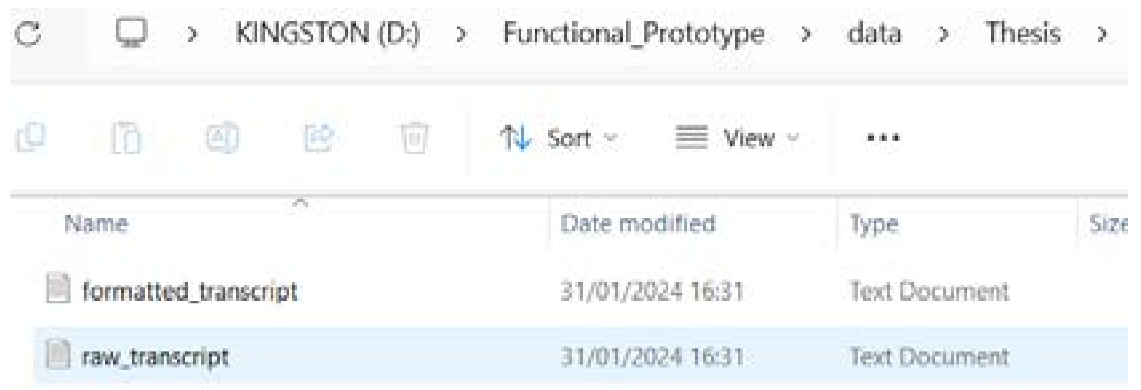

Jeroen van Kekem 5:14
OK.
And that's clear.
El was also wondering.
So if are there any dependencies with other venues, so it what is that?
What are the?
Distribution channels in the can you walk me through those?

Figure 29: Speakerturn format of stored Conversation Artefact in the MS Stream environment.

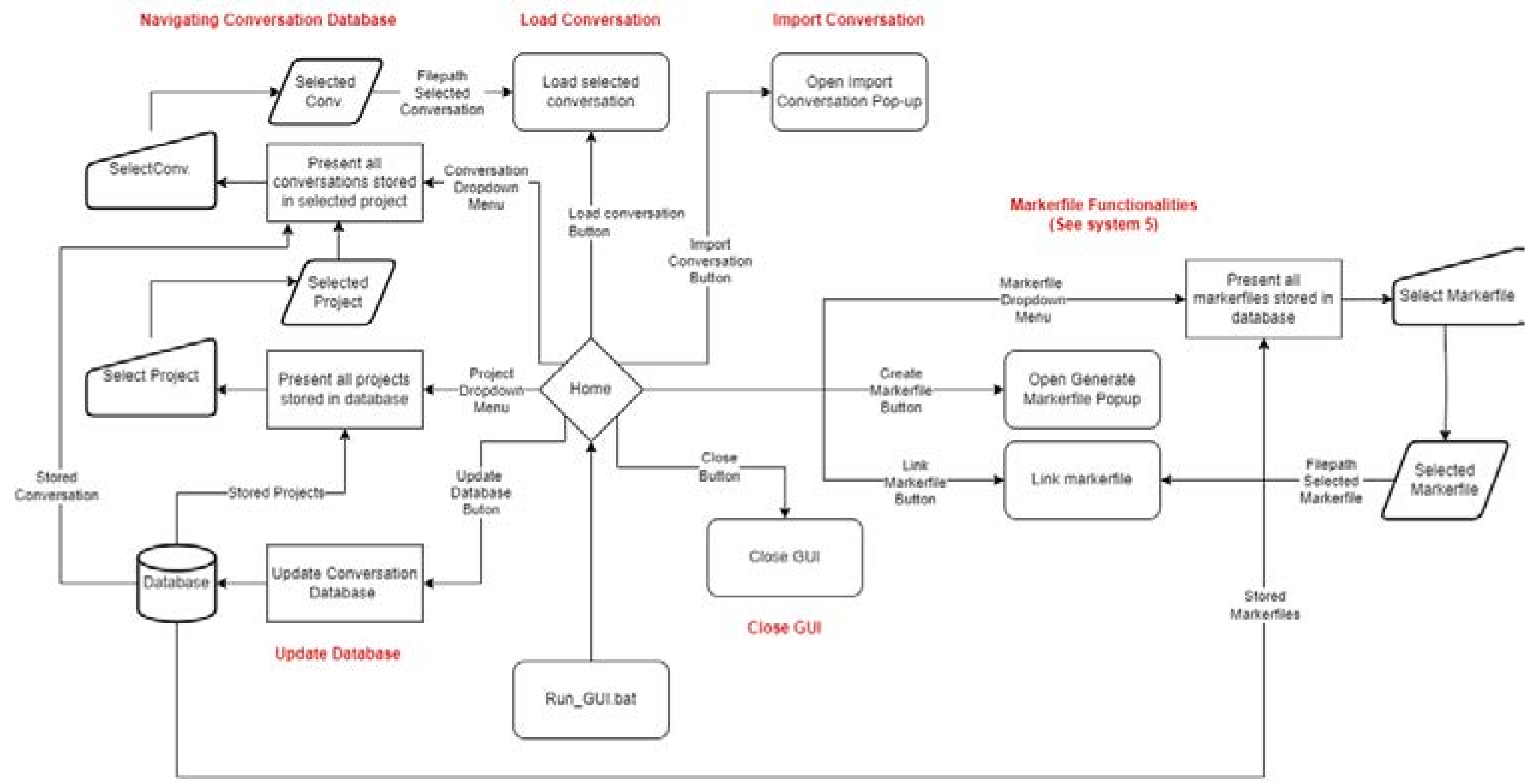


KINGSTON (D:) > Functional_Prototype > data > Thesis >

Name	Date modified	Type	Size
formatted_transcript	31/01/2024 16:31	Text Document	
raw_transcript	31/01/2024 16:31	Text Document	

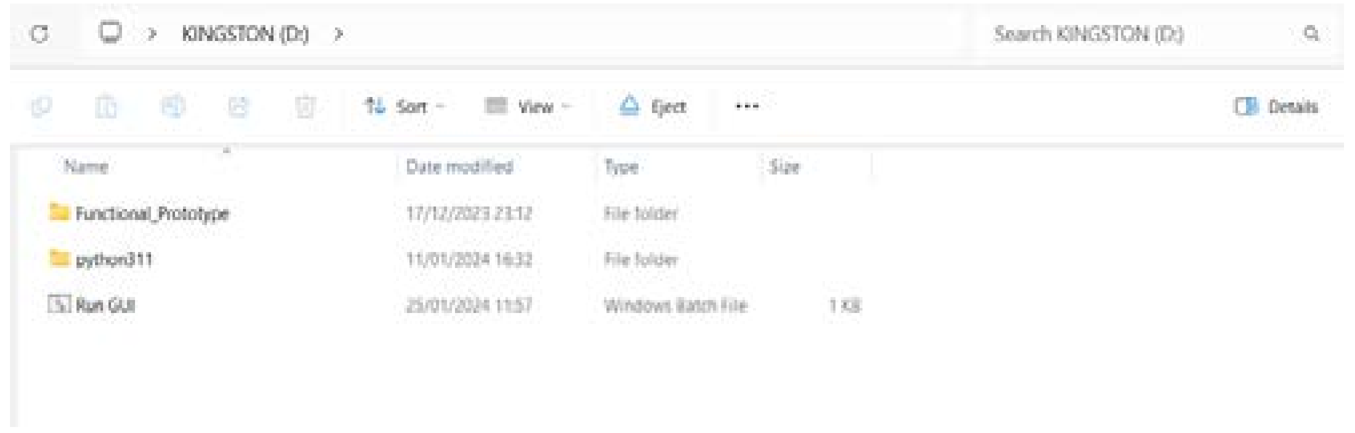
Figure 30: Outputs of both formatting steps stored in the database.

APPENDIX 0: REALIZED SYSTEM DEVELOPMENT MANUAL



Navigating Conversation Database
Load Conversation
Import Conversation
Selected Conv.
Filepath Selected Conversation
Load selected conversation
Open Import Conversation Pop-up
SelectConv.
Present all conversations stored in selected project
Conversation Dropdown Menu
Load conversation Button
Import Conversation Button
Markerfile Functionalities (See system 5)
Selected Project
Markerfile Dropdown Menu
Present all markerfiles stored in database
Select Markerfile
Select Project
Present all projects stored in database
Project Dropdown Menu
Home
Create Markerfile Button
Open Generate Markerfile Popup
Stored Conversation
Stored Projects
Update Database Buton
Close Button
Link Markerfile Button
Link markerfile
Filepath Selected Markerfile
Selected Markerfile
Database
Update Conversation Database
Close GUI
Stored Markerfiles
Close GUI
Update Database
Run_GUI.bat

Figure 31: GUI flowchart each end of this flowchart will be continued in the other sections of this chapter.


KINGSTON (D:) >

Search KINGSTON (D:)

Sort · View · Eject · ··· · Details

Name	Date modified	Type	Size
Functional_Prototype	17/12/2023 23:12	File folder	
python311	11/01/2024 16:32	File folder	
Run GUI	25/01/2024 11:57	Windows Batch File	1 KB

Figure 32: Main folder of the system database containing the functional prototype, python to run the prototype and a bootloader called 'RUN GUI.bat' used to start the system GUI.

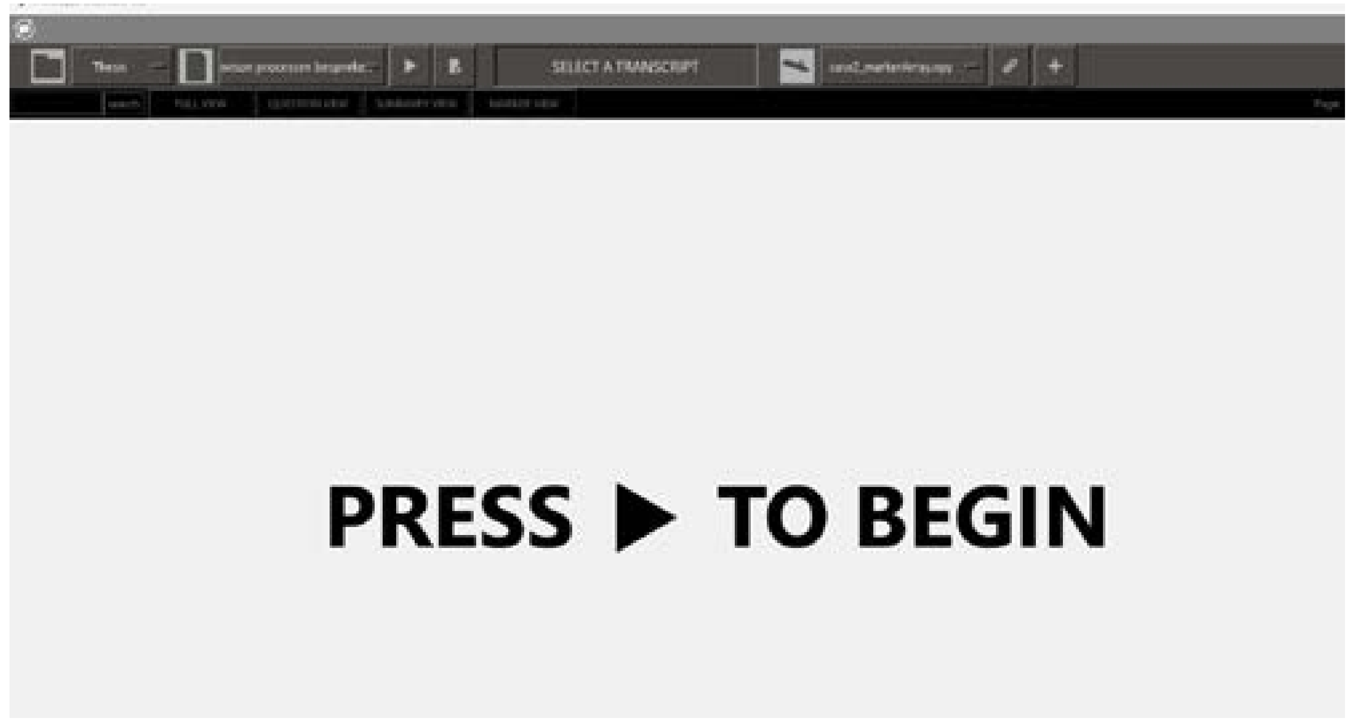


Figure 33: Home screen of GUI design.

APPENDIX 0: REALIZED SYSTEM DEVELOPMENT MANUAL

5.2 System 2: GUI

The system's graphical user interface (GUI) serves as the primary means for users to interact with the system. Developed using Python, the GUI leverages tkinter, which is the standard interface toolkit for Python (Tkinter — Python Interface to Tcl/Tk, n.d.). Comprising multiple screens, the GUI facilitates various types of interactions. These screens include:

1. Home Screen: The central hub where users can access different functionalities and navigate to other sections of the system.
2. Import Conversation Pop-up: Allows users to import conversation artefacts into the system seamlessly. This pop-up streamlines the process of bringing conversation data into the system for analysis.
3. Speakerturn Canvas: Provides a visual representation of speakerturns within the conversation artefact. This canvas enables users to view and interact with speakerturns efficiently.
4. Speakerturn in Context Pop-up: Offers additional contextual information for individual speakerturns, enhancing the user's understanding of the conversation flow and content.
5. Create Markerfile Pop-up: Enables users to create marker files, which serve as annotations or markers within the conversation artefact. This functionality aids in organizing and annotating the conversation data for further analysis.

Development and pre-use requirements

The GUI has been developed using Python 3.11, utilizing the tkinter module, which encompasses all standard GUI components available in Python. The system is self-contained on a single USB flash drive, comprising a folder that stores all system data, the Python interpreter necessary to run the algorithms, and a bootloader responsible for initiating the GUI upon user selection.
To ensure a seamless and user-friendly experience, all essential packages have been pre-installed on the USB drive. Presently, the system is compatible only with Windows operating systems. Appendix chapter 6 provides a comprehensive overview of all required packages, including their versions, needed to execute the system. To replicate the system, simply copy this list into a .txt file and install the packages using the command 'pip install -r requirements.txt' in python.
Moreover, approximately 2.5 GB of free space is required to run the system effectively. Additionally, it is imperative to have Java pre-installed on the system to enable certain functionalities within the system.

Starting the system

To initiate the system, the user simply runs the 'Run GUI' bootloader, which directs them to the home screen of the system GUI (refer to figure 33). Here, the user can engage in various interactions pertaining to the conversation database and marker functionality (as described in system 5). Interactions related to the conversation database include:

1. Viewing the contents of the conversation database.
2. Navigating to specific conversations within the database.
3. Loading conversation artefacts.
4. Importing conversation artefacts.

Navigating the conversation database

The navigation of the conversation database involves the use of two dropdown menus displaying all available projects and conversations within the database (refer to figure 34). When a new conversation is imported into the system, it may not immediately appear in the dropdown menu. To update the database and display the newly imported conversation, the user can either click the update button (see figure 34, 8) or restart the application. Figure 31 illustrates the flow of information between the GUI, the user, and the database.

Import new conversation

To import a new conversation into the system, the user can select the "Import Conversation Artefact" button in the GUI (refer to figure 34, 4). This action triggers the execution of the algorithm, which has been specifically designed for this purpose. Upon clicking the button, a pop-up window appears, prompting the user to provide necessary data linked to the conversation. The required information includes:

1. Conversation Name: This serves as an identifier for the conversation within the database.
2. Speaker Names: Currently, the system supports two speakers, but this capacity can be easily expanded. The names of the speakers can be derived from the profile names of the meeting room participants in MS Teams.
3. Filepath of the Conversation Artefact: The user can locate the conversation artefact by using the browse function included in the pop-up.
After ensuring that all required information is provided accurately, the user can press the "Create" button to initiate the processing steps described in the next section, "System 3: NLP Processing." It's important to note that the prototype will import the conversation artefact into the project folder that is currently selected. Please be aware that this process may take approximately 10-15 minutes to complete fully. Once processed, the resulting output can be located in the database by clicking the "Update" button (refer to figure 34, 8).

Load conversation

To display processed conversations, the user can load any conversation in the Speakerturn Canvas by clicking the "Load Conversation" button (see figure 34, 3). Activating this button triggers the execution of the load_transcript.py algorithm. Upon clicking the button, the algorithm enables the Viewbar, which provides access to several filtering options for the set of speakerturns (this flow is visualised in figure 38). These options include:

1. Full View Button: Displays all speakerturns of the conversation.
2. Question View Button: Shows only the speakerturns that contain a question.
3. Summary View Button: Presents the user with all the speakerturns containing a relevant question.
4. Marker View Button: This functionality will be discussed in System 5.

Additionally, the algorithm turn_frame.py has been designed to present time, speaker, and text data to the user. For each speakerturn, a turn_frame is created with the corresponding data. If another conversation is loaded, the existing data is first cleared before loading the currently selected conversation. Please note that these features are included with the intent to provide a positive user experience of navigating the conversation to filter the speakerturn dataset to search for specific information. When the load conversation algorithm is run, the filter is set to full view by default.

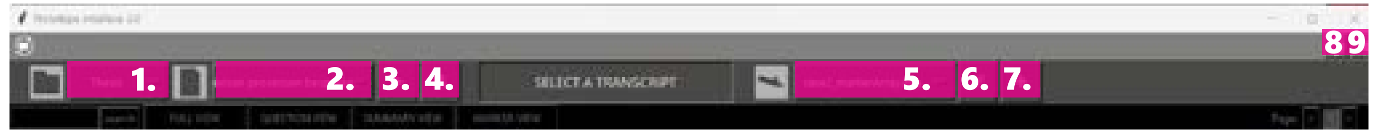


Figure 34: Navigating the system database in the GUI. (1)Project and (2)conversation navigation, (3)load conversation, (4)import conversation artefact, (5)select markerfile, (6)link markerfile, (7)create markerfile, (8)update database, (9)close application.

Speakerturns navigation

Page navigation

The system presents a maximum of 100 speakerturns at a time (due to technical limitations in python). To navigate through the pages of speakerturns, the user can utilize the page navigation buttons.

Show speakerturns in context

To address the likelihood of requirement-relevant information being contained in answers to prompted interview questions, each speakerturn includes a "Show in Context" button (disabled in the Full View). Clicking this button opens a pop-up where the user can view up to 10 speakerturns before or after the selected speakerturn. By default, the first speakerturn after the selected one is presented to the user, as there's a high probability that it contains the answer to the question. This feature is enabled in all views except the Full View. To differentiate the annotation based on which the speakerturns are filtered, distinct icons are used for the Full View, Questions View, Summary View, and Marker View. This functionality enhances the user's ability to navigate speakerturns effectively, ensuring they can access relevant information with

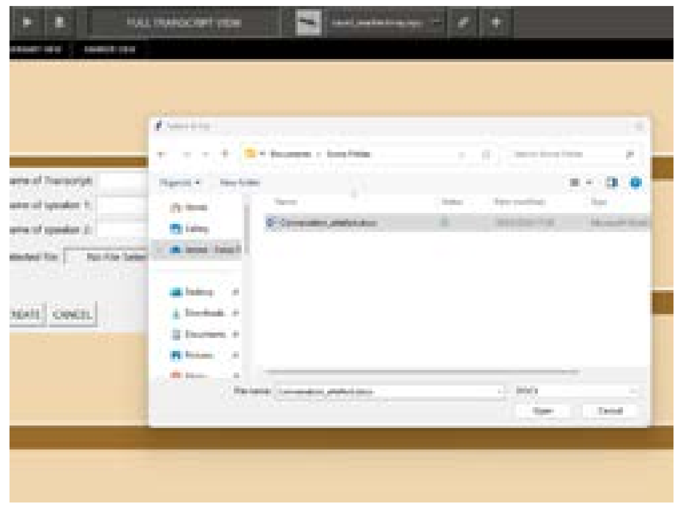


Figure 35: Browse for the desired conversation artefact to import.

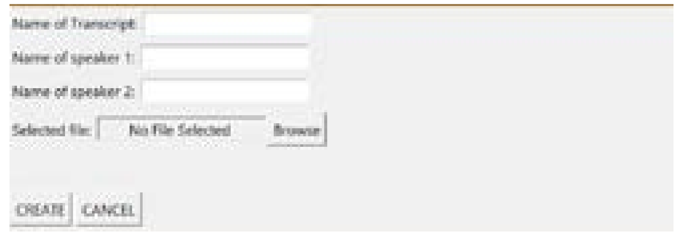

Name of Transcript:
Name of speaker 1:
Name of speaker 2:
Selected file: No File Selected Browse
CREATE CANCEL

Figure 36: Pop-up to import a conversation artefact.

APPENDIX 0: REALIZED SYSTEM DEVELOPMENT MANUAL

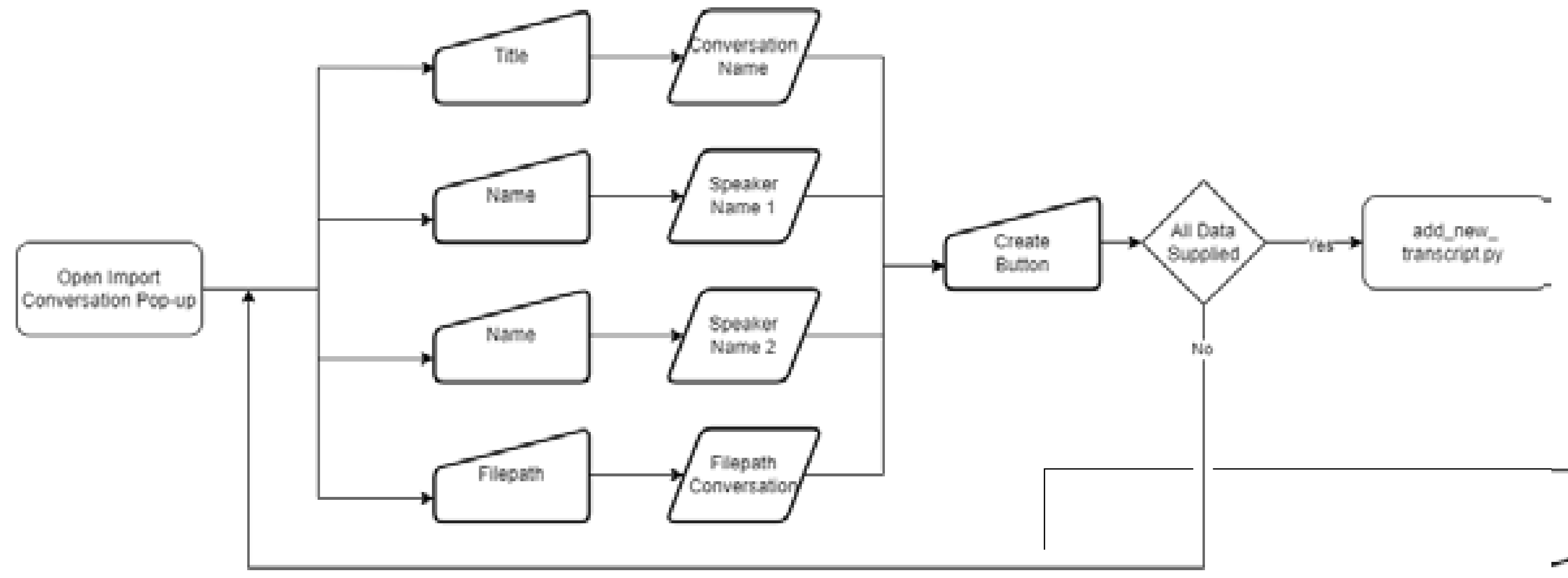


Figure 37: Flowchart of the import conversation pop-up.

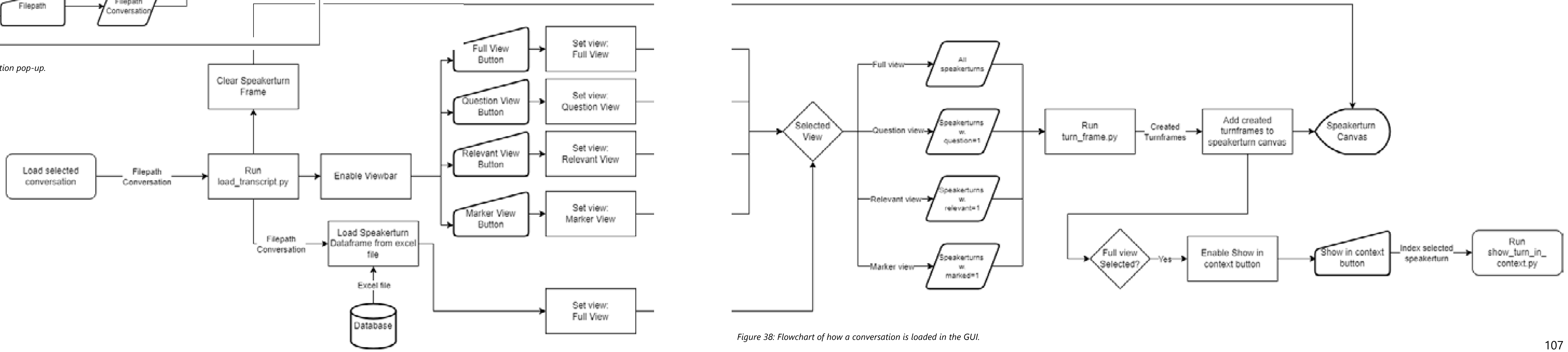
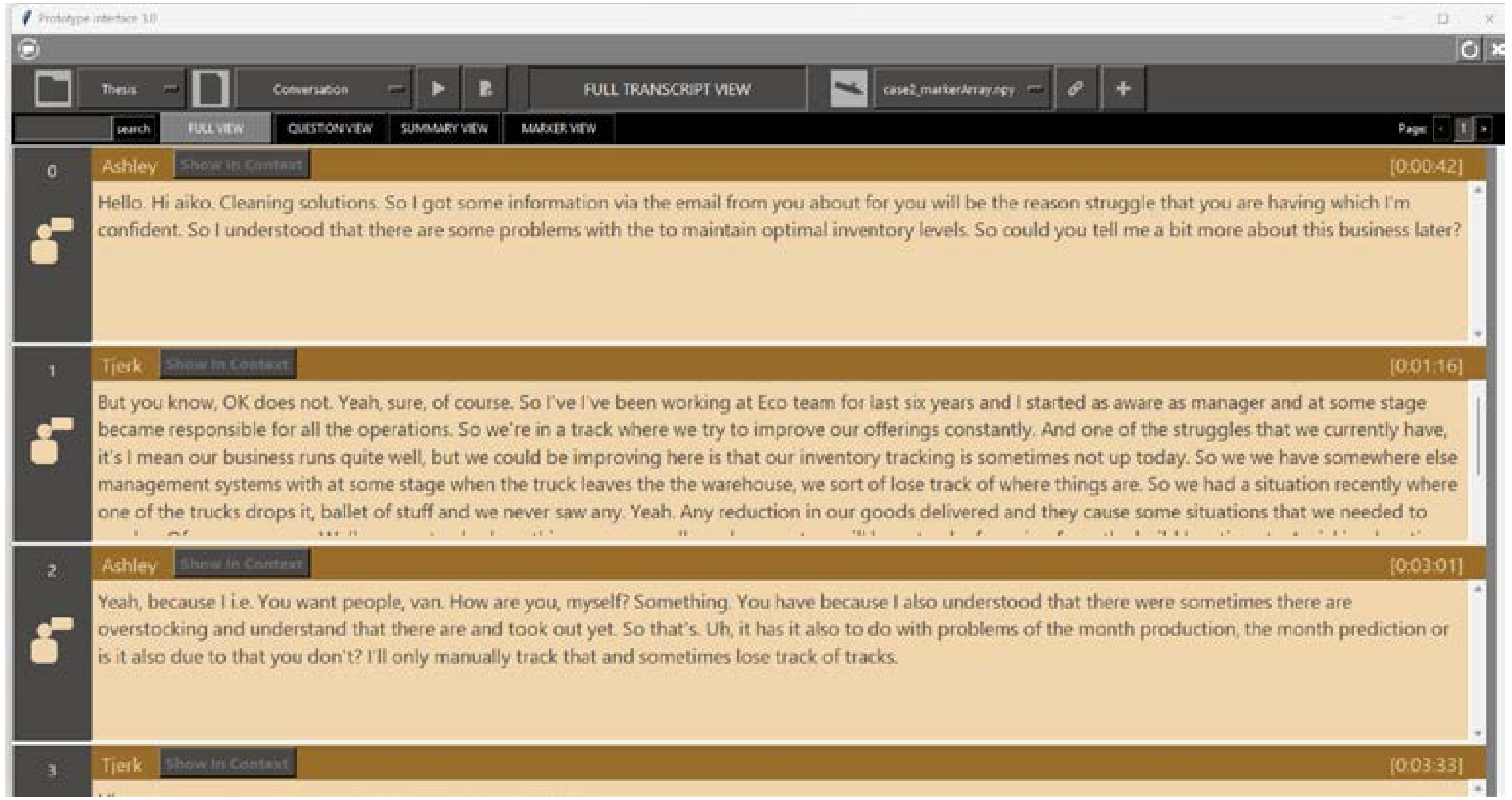


Figure 38: Flowchart of how a conversation is loaded in the GUI.

APPENDIX 0: REALIZED SYSTEM DEVELOPMENT MANUAL



0 Ashley [0:00:42]

Hello. Hi aiko. Cleaning solutions. So I got some information via the email from you about for you will be the reason struggle that you are having which I'm confident. So I understood that there are some problems with the to maintain optimal inventory levels. So could you tell me a bit more about this business later?

1 Tjerk [0:01:16]

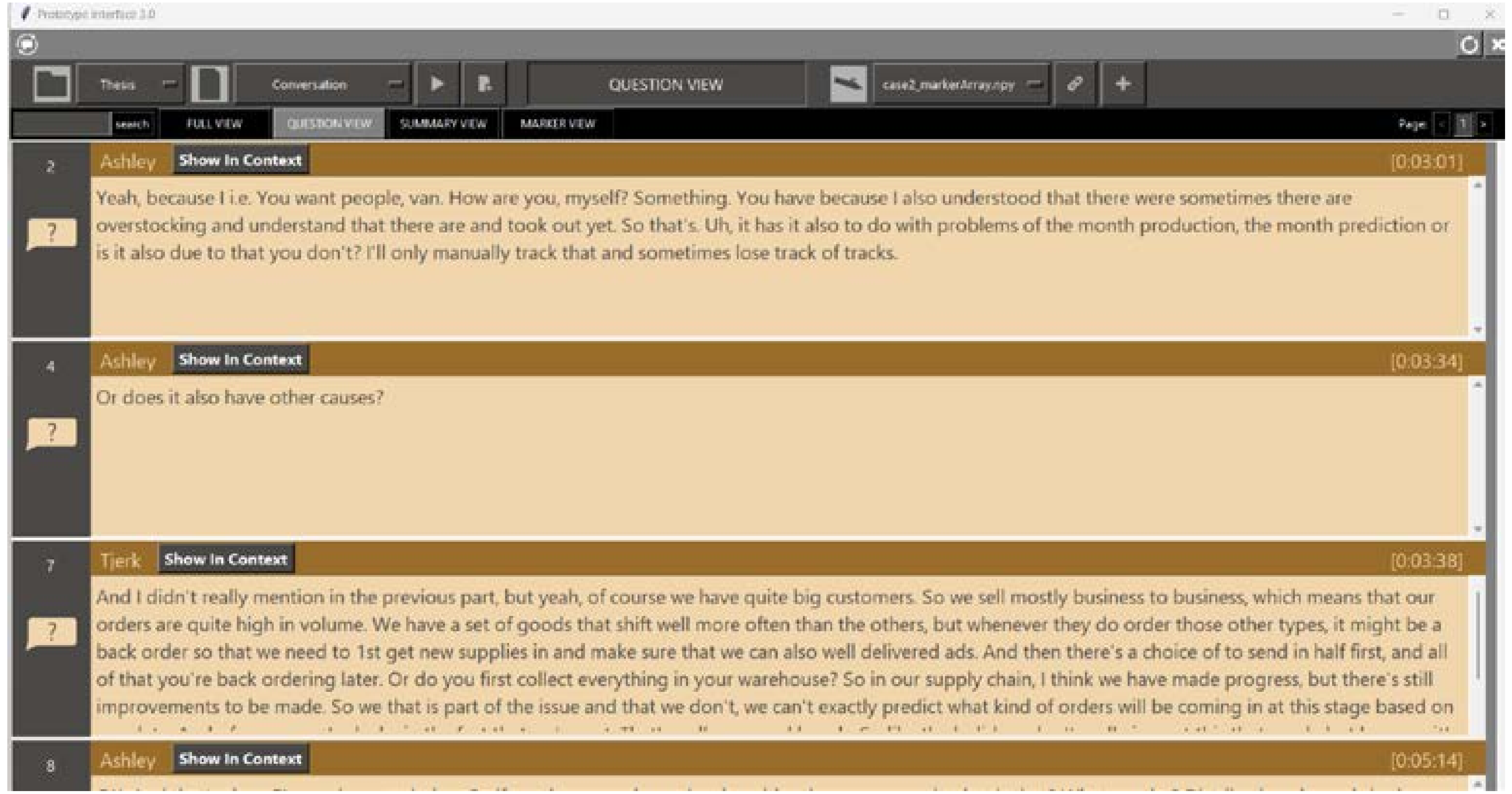
But you know, OK does not. Yeah, sure, of course. So I've I've been working at Eco team for last six years and I started as aware as manager and at some stage became responsible for all the operations. So we're in a track where we try to improve our offerings constantly. And one of the struggles that we currently have, it's I mean our business runs quite well, but we could be improving here is that our inventory tracking is sometimes not up today. So we we have somewhere else management systems with at some stage when the truck leaves the the warehouse, we sort of lose track of where things are. So we had a situation recently where one of the trucks drops it, ballet of stuff and we never saw any. Yeah. Any reduction in our goods delivered and they cause some situations that we needed to

2 Ashley [0:03:01]

Yeah, because I i.e. You want people, van. How are you, myself? Something. You have because I also understood that there were sometimes there are overstocking and understand that there are and took out yet. So that's. Uh, it has it also to do with problems of the month production, the month prediction or is it also due to that you don't? I'll only manually track that and sometimes lose track of tracks.

3 Tjerk [0:03:33]

Figure 39: All speakerturns of the conversation.



2 Ashley [0:03:01]

Yeah, because I i.e. You want people, van. How are you, myself? Something. You have because I also understood that there were sometimes there are overstocking and understand that there are and took out yet. So that's. Uh, it has it also to do with problems of the month production, the month prediction or is it also due to that you don't? I'll only manually track that and sometimes lose track of tracks.

4 Ashley [0:03:34]

Or does it also have other causes?

7 Tjerk [0:03:38]

And I didn't really mention in the previous part, but yeah, of course we have quite big customers. So we sell mostly business to business, which means that our orders are quite high in volume. We have a set of goods that shift well more often than the others, but whenever they do order those other types, it might be a back order so that we need to 1st get new supplies in and make sure that we can also well delivered ads. And then there's a choice of to send in half first, and all of that you're back ordering later. Or do you first collect everything in your warehouse? So in our supply chain, I think we have made progress, but there's still improvements to be made. So we that is part of the issue and that we don't, we can't exactly predict what kind of orders will be coming in at this stage based on

8 Ashley [0:05:14]

Figure 40: Question view. The conversation filtered to show only the questions of the conversation.

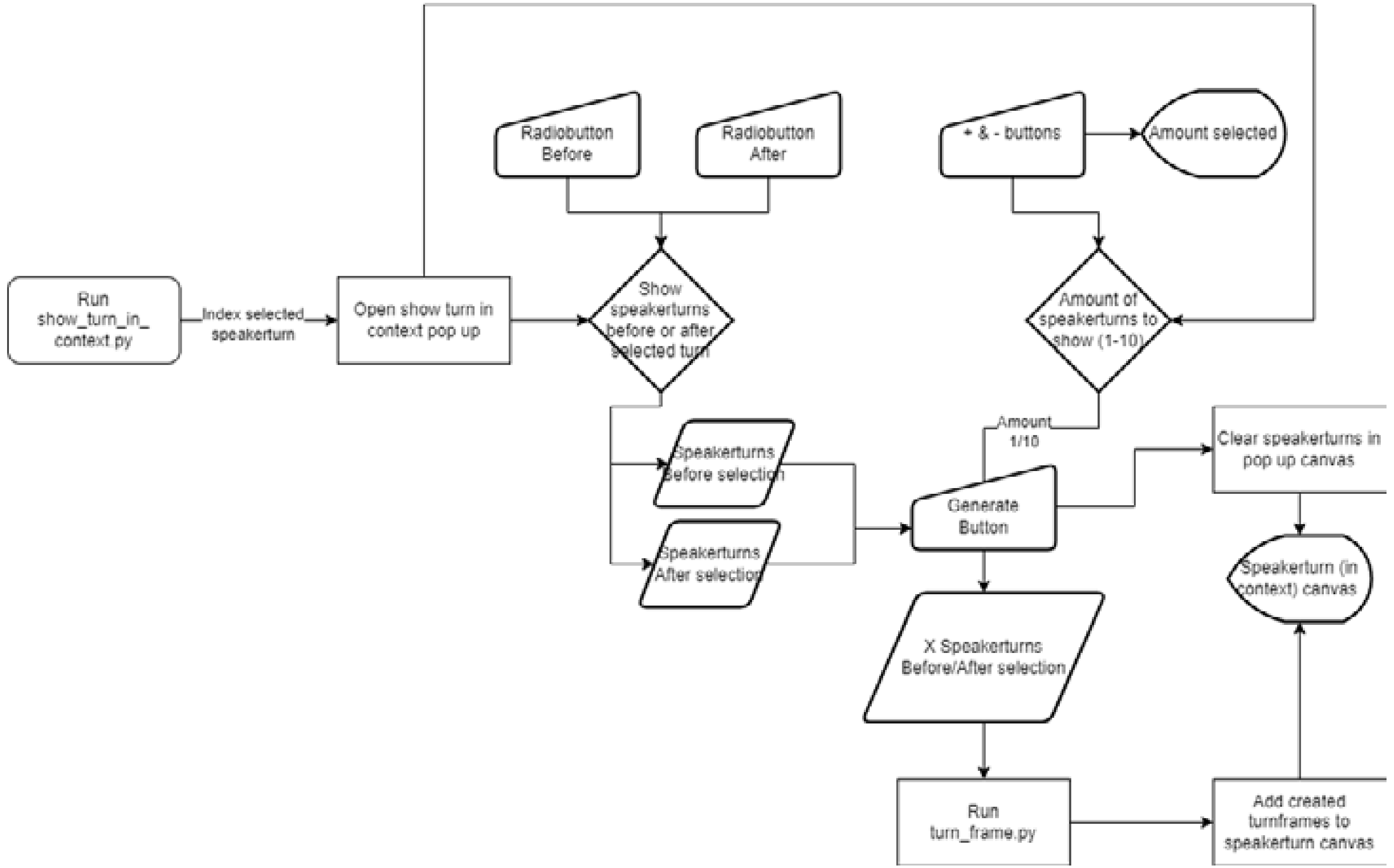
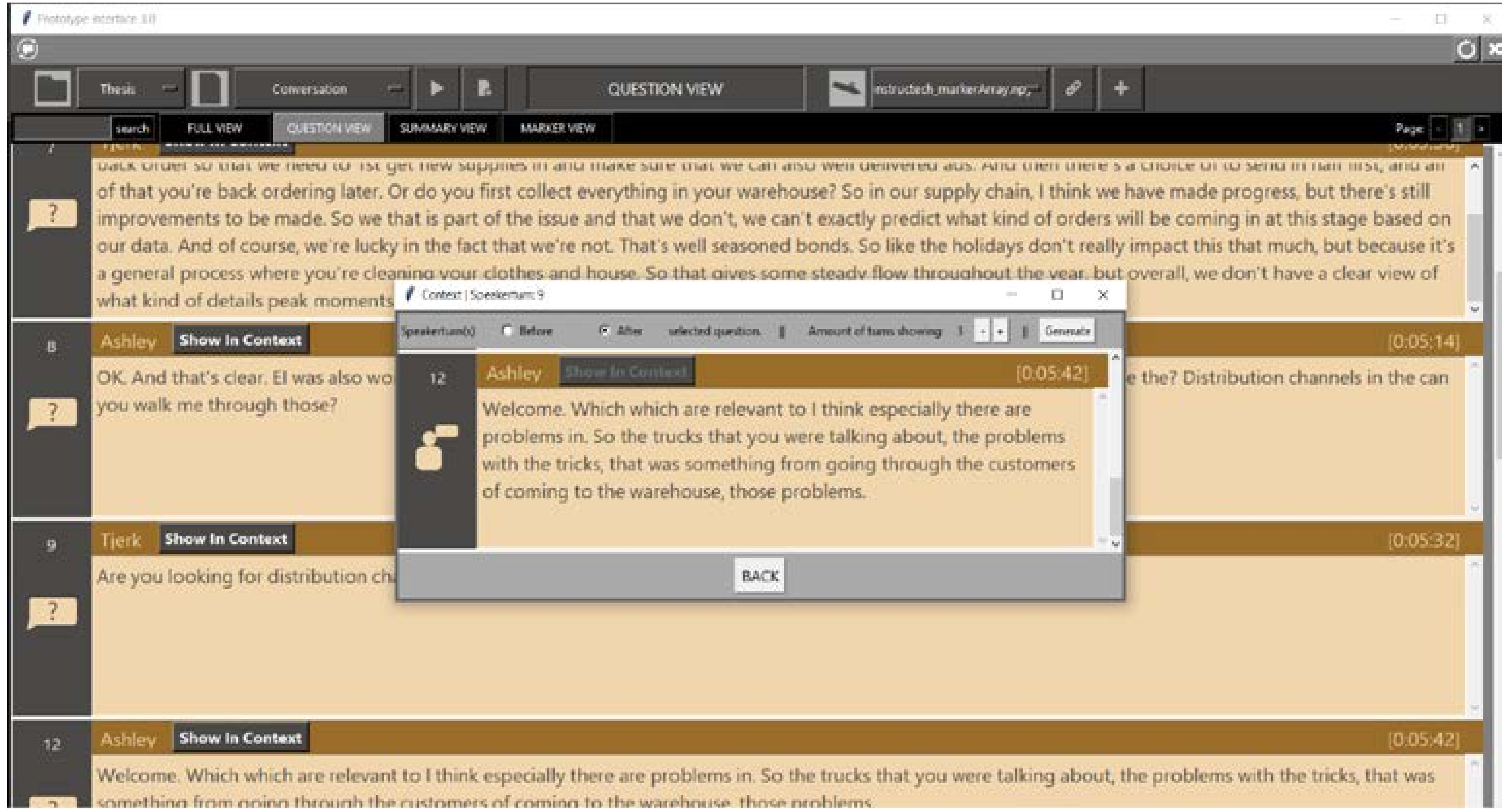

Run show_turn_in_context.py
Index selected speakerturn
Open show turn in context pop up
Radiobutton Before
Radiobutton After
Show speakerturns before or after selected turn
Speakerturns Before selection
Speakerturns After selection
+ & - buttons
Amount selected
Amount of speakerturns to show (1-10)
Amount 1/10
Generate Button
Clear speakerturns in pop up canvas
Speakerturn (in context) canvas
X Speakerturns Before/After selection
Run turn_frame.py
Add created turnframes to speakerturn canvas

Figure 41: Flowchart of speakerturn in context pop-up.



of that you're back ordering later. Or do you first collect everything in your warehouse? So in our supply chain, I think we have made progress, but there's still improvements to be made. So we that is part of the issue and that we don't, we can't exactly predict what kind of orders will be coming in at this stage based on our data. And of course, we're lucky in the fact that we're not. That's well seasoned bonds. So like the holidays don't really impact this that much, but because it's a general process where you're cleaning your clothes and house. So that gives some steady flow throughout the year, but overall, we don't have a clear view of what kind of details peak moments

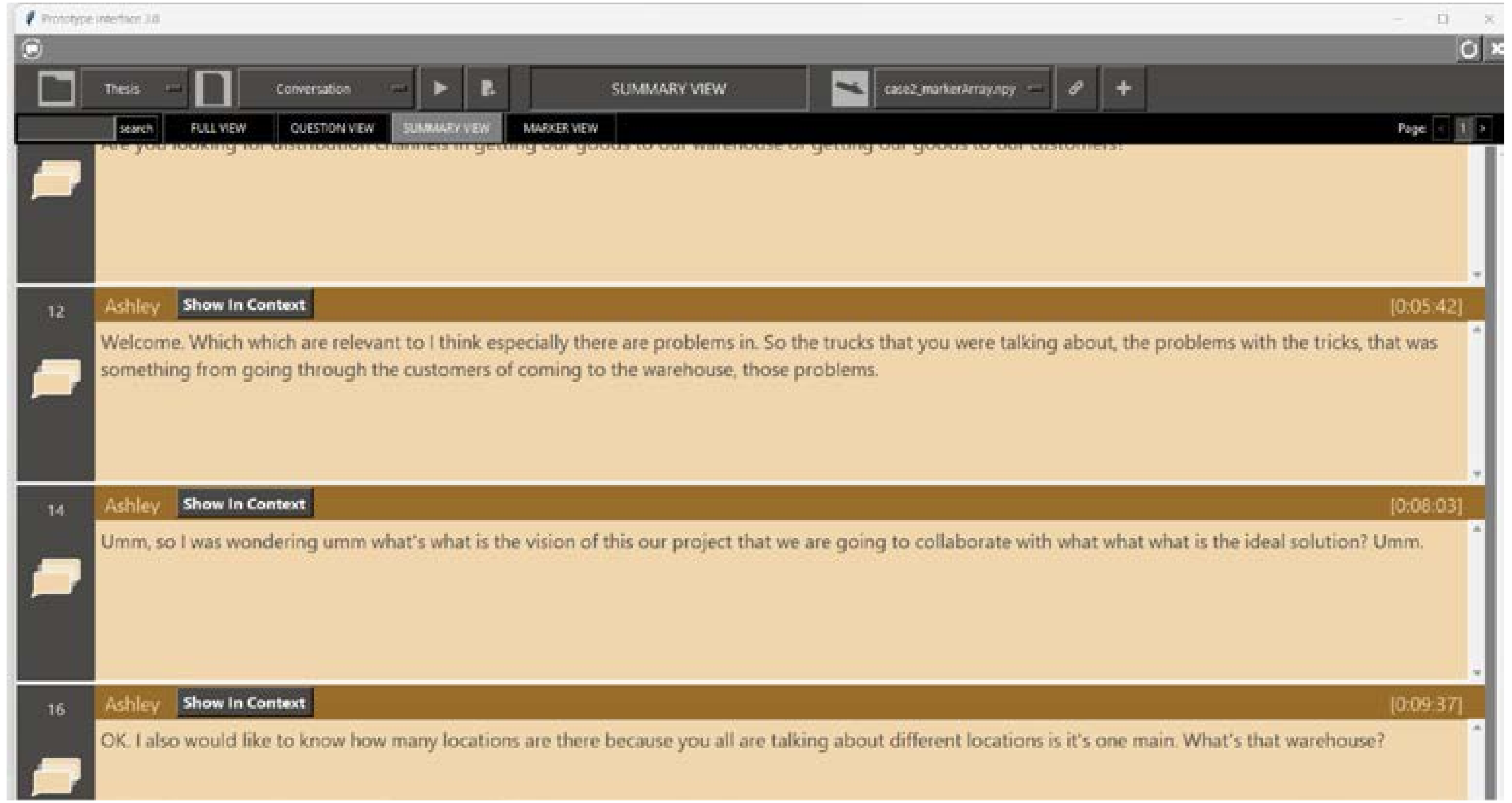
8 Ashley Show In Context [0:05:14]
OK. And that's clear. El was also wo ... e the? Distribution channels in the can you walk me through those?

12 Ashley Show In Context [0:05:42]
Welcome. Which which are relevant to I think especially there are problems in. So the trucks that you were talking about, the problems with the tricks, that was something from going through the customers of coming to the warehouse, those problems.

9 Tjerk Show In Context [0:05:32]
Are you looking for distribution ch

12 Ashley Show In Context [0:05:42]
Welcome. Which which are relevant to I think especially there are problems in. So the trucks that you were talking about, the problems with the tricks, that was something from going through the customers of coming to the warehouse, those problems

Figure 42: Speakerturn presented in the context of the conversation.



Are you looking for distribution channels in getting our goods to our warehouse or getting our goods to our customers?

12 Ashley Show In Context [0:05:42]
Welcome. Which which are relevant to I think especially there are problems in. So the trucks that you were talking about, the problems with the tricks, that was something from going through the customers of coming to the warehouse, those problems.

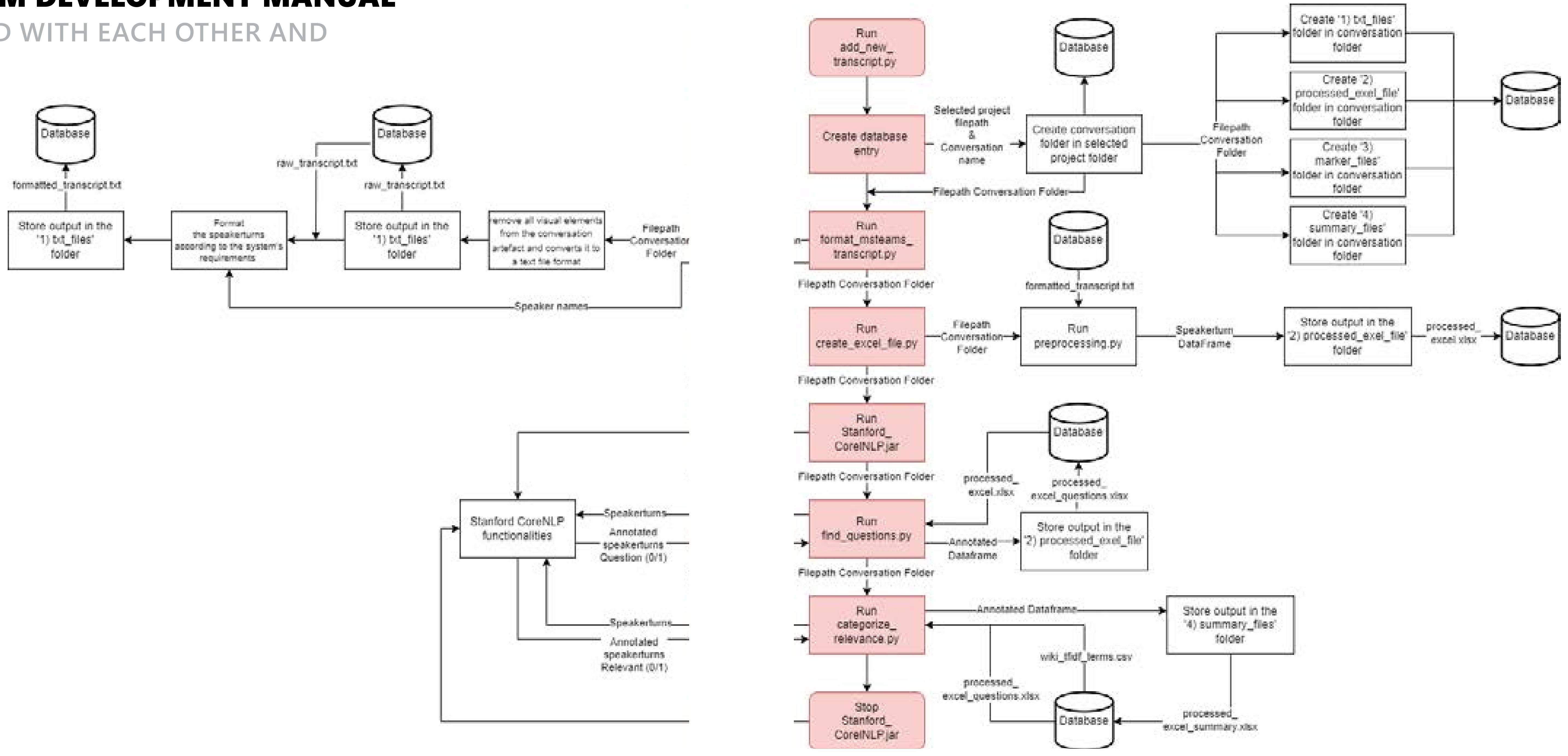
14 Ashley Show In Context [0:08:03]
Umm, so I was wondering umm what's what is the vision of this our project that we are going to collaborate with what what what is the ideal solution? Umm.

16 Ashley Show In Context [0:09:37]
OK. I also would like to know how many locations are there because you all are talking about different locations is it's one main. What's that warehouse?

Figure 43: Summary view. The conversation is filtered to show only questions relevant to the conversation.

APPENDIX 0: REALIZED SYSTEM DEVELOPMENT MANUAL

ALL NLP FUNCTIONALITIES LINKED WITH EACH OTHER AND THE DATABASE



Run add_new_transcript.py
Database
Create '1) txt_files' folder in conversation folder
Create '2) processed_exel_file' folder in conversation folder
Create '3) marker_files' folder in conversation folder
Create '4) summary_files' folder in conversation folder
Database
Selected project filepath & Conversation name
Create database entry
Create conversation folder in selected project folder
Filepath Conversation Folder
Filepath Conversation Folder
Run format_msteams_transcript.py
Database
formatted_transcript.txt
Filepath Conversation Folder
Run create_excel_file.py
Filepath Conversation Folder
Run preprocessing.py
Speakerturn DataFrame
Store output in the '2) processed_exel_file' folder
processed_excel.xlsx
Database
Filepath Conversation Folder
Run Stanford_CoreNLP.jar
Database
Filepath Conversation Folder
processed_excel.xlsx
processed_excel_questions.xlsx
Run find_questions.py
Annotated Dataframe
Store output in the '2) processed_exel_file' folder
Filepath Conversation Folder
Run categorize_relevance.py
Annotated Dataframe
Store output in the '4) summary_files' folder
wiki_tfidf_terms.csv
processed_excel_questions.xlsx
Database
processed_excel_summary.xlsx
Stop Stanford_CoreNLP.jar
Database
formatted_transcript.txt
Store output in the '1) txt_files' folder
Format the speakerturns according to the system's requirements
raw_transcript.txt
Database
raw_transcript.txt
Store output in the '1) txt_files' folder
remove all visual elements from the conversation artefact and converts it to a text file format
Filepath Conversation Folder
Speaker names
Stanford CoreNLP functionalities
Speakerturns
Annotated speakerturns Question (0/1)
Speakerturns
Annotated speakerturns Relevant (0/1)

Figure 44: Designed system functionalities

APPENDIX 0: REALIZED SYSTEM DEVELOPMENT MANUAL

5.3 System 3: NLP Functionalities

This subchapter delineates the various processes integrated into the system prototype. These processes are activated when a user imports a conversation and provides essential information such as the file location of the conversation artefact, participant names, and a recognizable conversation name. Upon confirmation of the data, the system executes multiple process steps (Figure 44) , including:

1. Create Database Entry: This step involves creating an entry for the conversation in the database, enabling efficient storage and retrieval of conversation data.
2. Format MS Teams Transcript: The system formats the conversation transcript obtained from MS Teams, removing unnecessary visual elements and standardizing the format for further processing.
3. Create Excel File: The system generates an Excel file containing the formatted conversation data, facilitating easy analysis and manipulation of the information.
4. Run Stanford CoreNLP: This process utilizes the Stanford CoreNLP library for natural language processing tasks such as tokenization, sentence splitting, part-of-speech tagging, and named entity recognition.
5. Find Questions: The system identifies and extracts questions from the conversation transcript, recognizing them as potential sources of requirement-relevant information.
6. Categorize Relevance: This step involves categorizing speakerturns based on their relevance to the elicitation goals, ensuring that requirement-relevant information is appropriately identified and categorized for further analysis.

Figure 45 depicts the initial configuration of the REConSum components that required connection through algorithm programming. The realized NLP processing comprises multiple process steps, with each step producing output stored in the database. The outputs are stored in the system as Excel files, organized systematically for each conversation. This organization allows for the analysis of processing results after each process step, as further elaborated in System 4: Data Storage.

FORMAT_MSTEAMS_TRANSCRIPT

The algorithm format_msteams_transcript was developed to address the formatting issues present in the conversation artefacts generated by MS Teams, as discussed in System 1: Generating Conversation Artefacts. As illustrated before, MS Teams-generated artefacts contain visual elements and text formatting that do not conform to the system's expected structure (Chapter 5, System 1). This algorithm operates in two stages.

First, it removes all visual elements from the conversation artefact, converts it to a text file format, and stores in the database as raw_transcript. This initial step ensures that the raw conversation data is stripped of any unnecessary formatting (chapter 10 of the appendix).

Next, the algorithm formats the speakerturns according to the system's requirements, preparing them for further processing. This formatted version of the transcript is then stored in the system as formatted_transcript (chapter 10 of the appendix).

CREATE_EXCEL_FILE

The create_excel_file algorithm plays a crucial role in the NLP processing pipeline of the system. This algorithm operates on the formatted_transcript.txt output generated by the format_msteams_transcript algorithm. Here's how it works:

1) Input Data Access: The algorithm accesses the formatted_transcript.txt file, which contains the correctly formatted speakerturns obtained from the MS Teams conversation artefact.
2) Preprocessing: The preprocessing.py algorithm of REConSum is executed using the formatted transcript as input. This preprocessing step involves converting the formatted transcript into a DataFrame, which essentially organizes the speakerturn data into a structured matrix format. The algorithm is modified to facilitate the input-output interaction and incorporate additional functionalities such as annotation of important or marked speakerturns (figure 48).
3) DataFrame Creation: Once the preprocessing is complete, the resulting DataFrame is stored in the database as processed_excel.xlsx. This DataFrame contains all the speakerturn data organized in rows, along with any annotations or markings added during the preprocessing stage.

In chapter 11 of the appendix an output has been included, generated as part of a test case.

RUN_StanfordCoreNLP

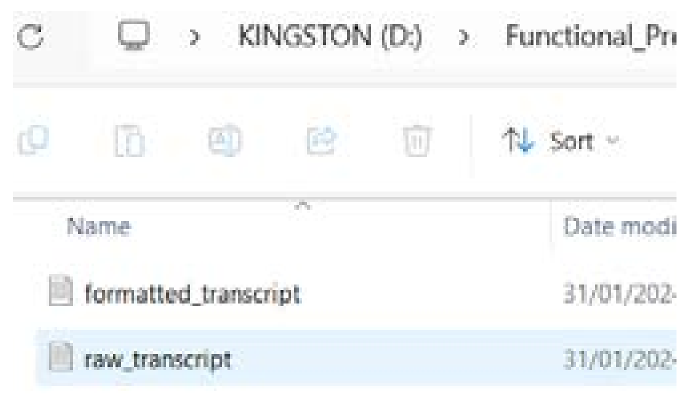
To enable the detection of questions and annotate speakerturns accurately, the StanfordCoreNLP tool needs to be run locally to process the text of speakerturns. This functionality is crucial for the proper functioning of the find_questions.py algorithm and for annotating speakerturns based on relevance.
To simplify the process for users and eliminate the need for manual installation and execution of StanfordCoreNLP, the system includes this tool in its database. When the processing reaches the stage where StanfordCoreNLP is required, the system automatically calls and runs it. Users only need to have Java installed on their devices, and the execution of StanfordCoreNLP is handled seamlessly by the system with just a few clicks. This eliminates the need for users to undertake any complex installation steps. Once the processing is complete and the categorise_relevance.py algorithm finishes its task, the StanfordCoreNLP system is automatically stopped, ensuring efficient resource utilization and system management.

FIND_QUESTIONS

The find_question.py algorithm is executed concurrently with the StanfordCoreNLP tool, leveraging its capabilities to process each speakerturn from the processed_excel.xlsx file. For each speakerturn, the algorithm utilizes the StanfordCoreNLP system to determine whether it contains a question. It annotates this information in the speakerturn matrix by assigning a value of 1 if the speakerturn contains a question and 0 otherwise.

CATEGORIZE_RELEVANCE

Exactly. The categorize_relevance.py algorithm follows the completion of find_questions.py. It accesses the annotated speakerturn data stored in processed_excel_questions.xlsx and employs the wiki_tfidf_terms.csv dataset. For each speakerturn containing a question, the algorithm assesses its relevance by comparing the text with the NL dataset. Speakerturns are then labeled as relevant (assigned 1) or irrelevant (assigned 0). The resulting annotations are integrated into a new DataFrame named processed_excel_summary.xlsx. This process helps summarize the conversation by pinpointing key information.
In chapter 13 of the appendix an output has been included, generated as part of a test case.
Once the annotation process is complete, the algorithm saves the newly annotated DataFrame under a different name, processed_excel_questions.xlsx. This annotated DataFrame provides valuable insights into the presence of questions within the speakerturns, facilitating further analysis and processing of the conversation data.
In chapter 12 of the appendix an output has been included, generated as part of a test case.


KINGSTON (D:) > Functional_Pr

Name	Date modi
formatted_transcript	31/01/202
raw_transcript	31/01/202

Figure 46: Outputs generated by the format_msteams_transcript.py algorithm.

REConSum | WHAT NEEDS TO BE REALIZED TO USE IT?

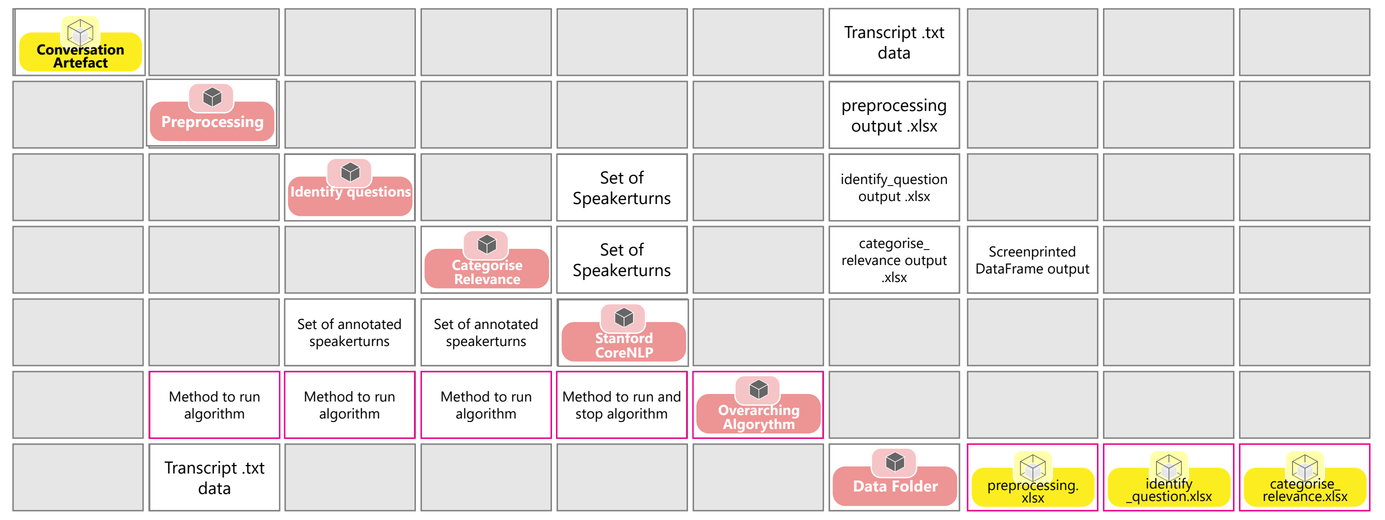
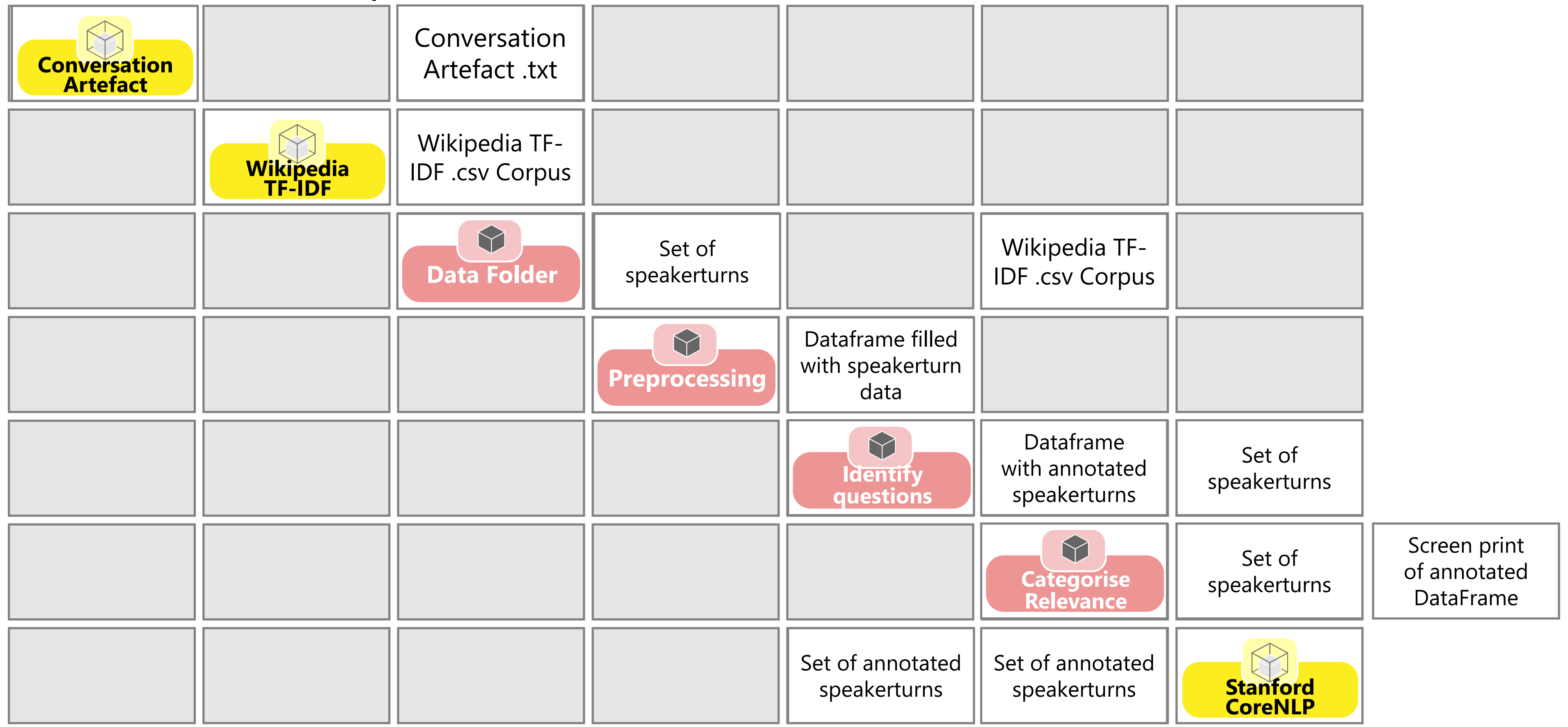


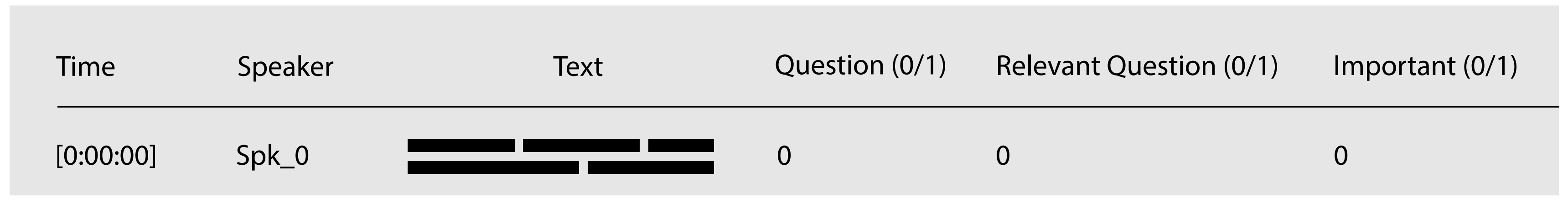
Figure 45: What needs to be realised to make REConSum function.

APPENDIX 0: REALIZED SYSTEM DEVELOPMENT MANUAL

FIGURE 43: RECONSUM | CURRENT FLOW OF INFORMATION.


Conversation Artefact
Conversation Artefact .txt
Wikipedia TF-IDF
Wikipedia TF-IDF .csv Corpus
Data Folder
Set of speakerturns
Wikipedia TF-IDF .csv Corpus
Preprocessing
Dataframe filled with speakerturn data
Identify questions
Dataframe with annotated speakerturns
Set of speakerturns
Categorise Relevance
Set of speakerturns
Screen print of annotated DataFrame
Set of annotated speakerturns
Set of annotated speakerturns
Stanford CoreNLP

To start using ReConSum, users should first ensure that the Conversation Artefact is correctly formatted and placed in the system's data folder. Additionally, they need to manually download the Wikipedia TF-IDF file and move it to the same data folder. Setting up StanfordCoreNLP is also required, which involves downloading it and ensuring it runs smoothly. Once these initial steps are completed, users will need to manually execute the systems using Python. It's important for users to define the paths to their input data to ensure the systems operate effectively. If all done correctly this is the resulted flow of information.



Time	Speaker	Text	Question (0/1)	Relevant Question (0/1)	Important (0/1)
[0:00:00]	Spk_0	[redacted]	0	0	0

Figure 48: DataFrame visualised. Each row contains all the data of a single speakerturn. Using NLP speakerturns can be annotated with different annotations such as the ones realised in the system design.

APPENDIX 0: REALIZED SYSTEM DEVELOPMENT MANUAL

System 4: Data storage

The system employs a structured data storage approach, utilizing a USB drive with approximately 3GB of storage space. This choice offers portability, scalability (allowing easy duplication onto multiple drives), and accessibility. The USB drive hosts various types of data, including a complete installation of Python 3.11 and the required packages to execute the code.

Within the USB drive, the "Functional_Prototype" folder houses the fully developed system. In the following sections, we'll delve into the specifics of the data structure employed and how outputs are organized within the system.

Interfacing with algorithms | using relative paths

To ensure that algorithms, assets, and input/output data are accessible throughout the system, paths are included in the code to specify their storage locations. Relative paths are used to define the location of the system on the drive, allowing the system to run seamlessly on different systems where the drive name may vary (e.g., 'C:' and 'D:'). Within the system, all components are defined using absolute paths to maintain consistency in their location.

Functional_Prototype | code folder

The code folder serves as the repository for all the algorithms essential for the system's functionality. In total, there are 13 algorithms housed within this folder. Table 1 provides an overview of all the algorithms present, along with their functionality and whether they were included or developed for this system.

In addition to the algorithms, the code folder also houses the Stanford CoreNLP system (version 4.5.4) and the Wikipedia_tfidf_terms.csv dataset. Both of these components are crucial for the system to operate as intended, as outlined in System 3.

Functional_Prototype | assets folder

The interface design uses multiple icons that need to be accessed from somewhere. The assets folder provides a storage solution for this purpose.

Functional_Prototype | markerfiles_ unsorted folder

(see system 5)

Functional_Prototype | data folder

The data folder serves as the repository for all the inputs and outputs generated by the system. To enhance the organization of data within the data folder and facilitate the locating of specific conversations, a decision was made to organize conversations at the project level. In the prototype, only one project has been added, where all conversations are stored.

Given that the system produces multiple outputs associated with each conversation, all conversation outputs are grouped per conversation, with each conversation having its own designated folder. For each imported conversation, the following folders are created automatically:

1. txt_files: Contains the text files generated during processing.
2. processed_excel_files: Contains the Excel files generated during processing.
3. marker_files: Stores marker files associated with the conversation.
4.summary_files: Holds summary files generated by the system.

Excel files as data format

Storing the outputs in Excel format was chosen for its accessibility and familiarity, eliminating the need for users to install complex software while enabling easy interaction with the data. Additionally, Excel allows for the analysis of conversations on a large quantitative scale, which is beneficial for research purposes.

Figure 52 provides an overview of how the output of a conversation is structured in Excel format, as demonstrated with a test conversation. This format facilitates the validation of outputs generated throughout the system processing.

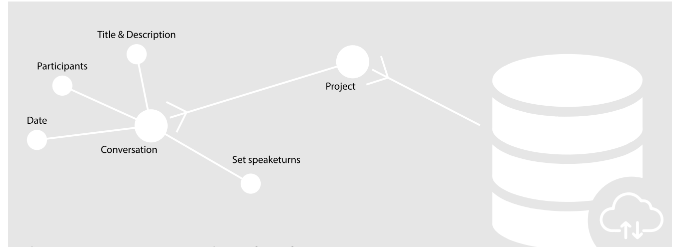


Figure 49: Conversation database concept.

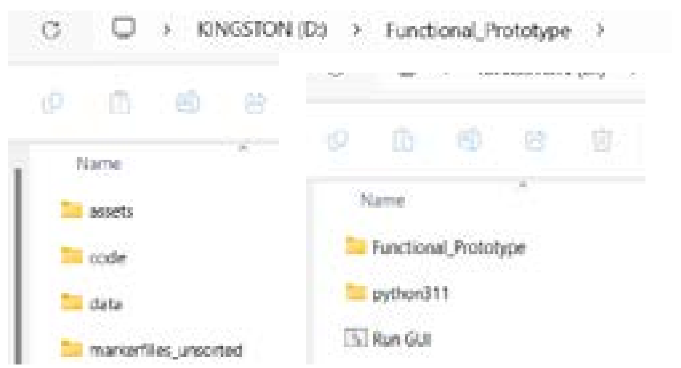
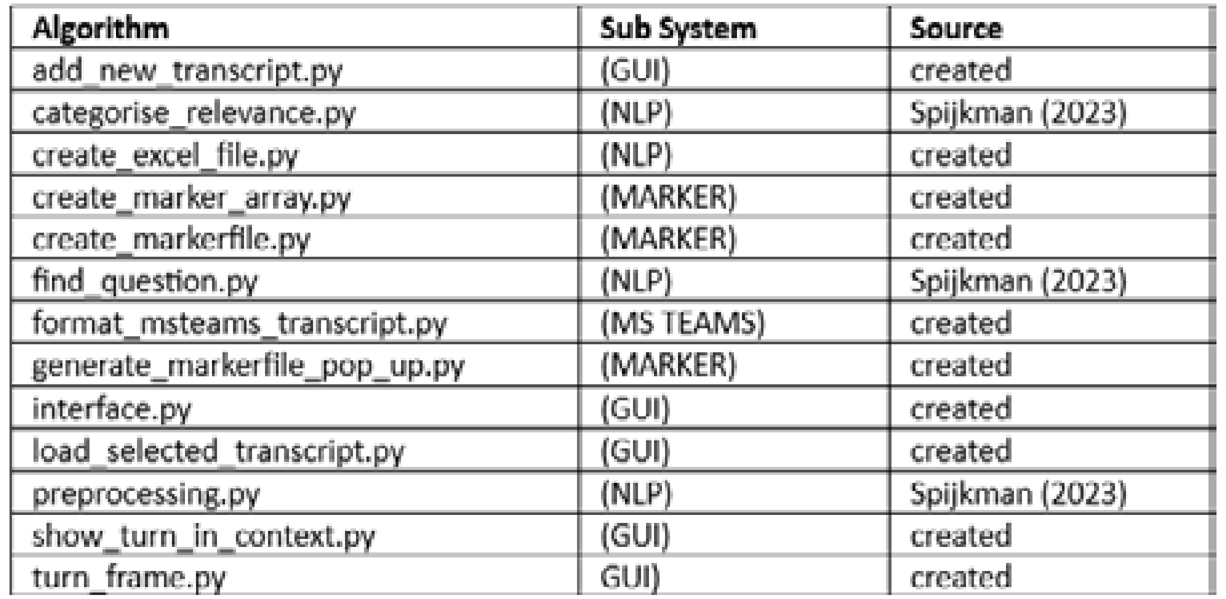
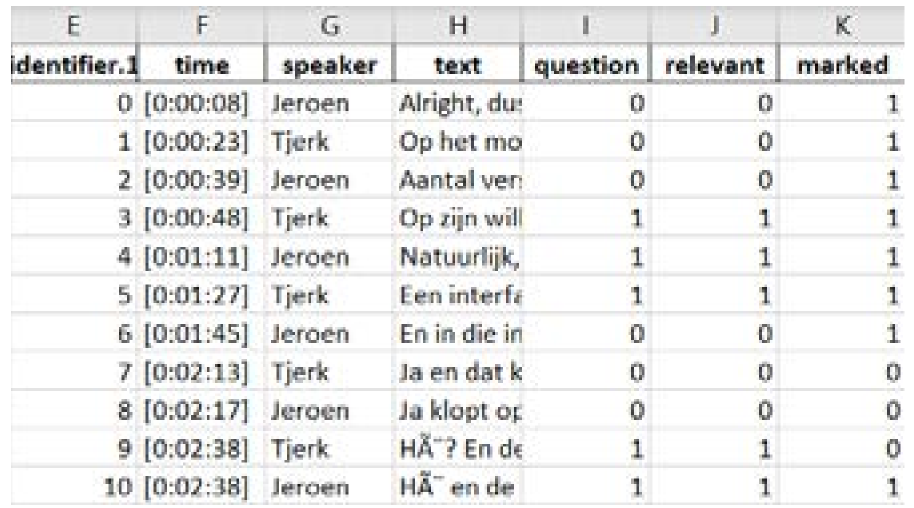


Figure 50 & 51: Data structure of the Functional_Prototype, and main folder.



Algorithm	Sub System	Source
add_new_transcript.py	(GUI)	created
categorise_relevance.py	(NLP)	Spijkman (2023)
create_excel_file.py	(NLP)	created
create_marker_array.py	(MARKER)	created
create_markerfile.py	(MARKER)	created
find_question.py	(NLP)	Spijkman (2023)
format_msteams_transcript.py	(MS TEAMS)	created
generate_markerfile_pop_up.py	(MARKER)	created
interface.py	(GUI)	created
load_selected_transcript.py	(GUI)	created
preprocessing.py	(NLP)	Spijkman (2023)
show_turn_in_context.py	(GUI)	created
turn_frame.py	GUI)	created

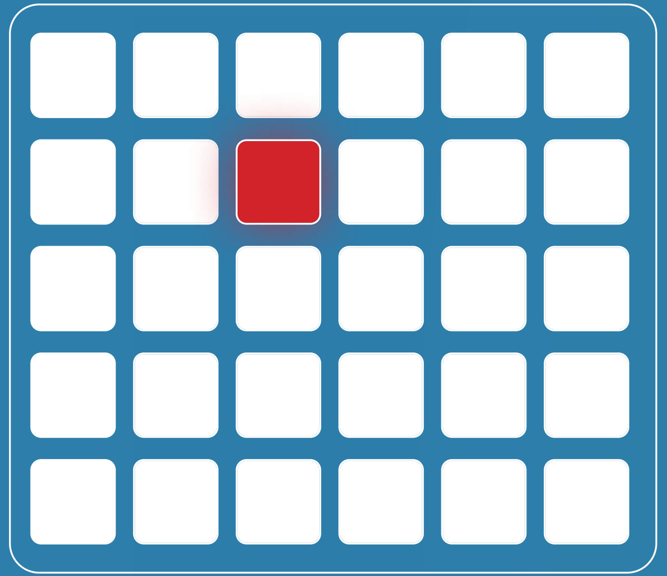
Table 1: Overview of all the algorithms that are present within the system.



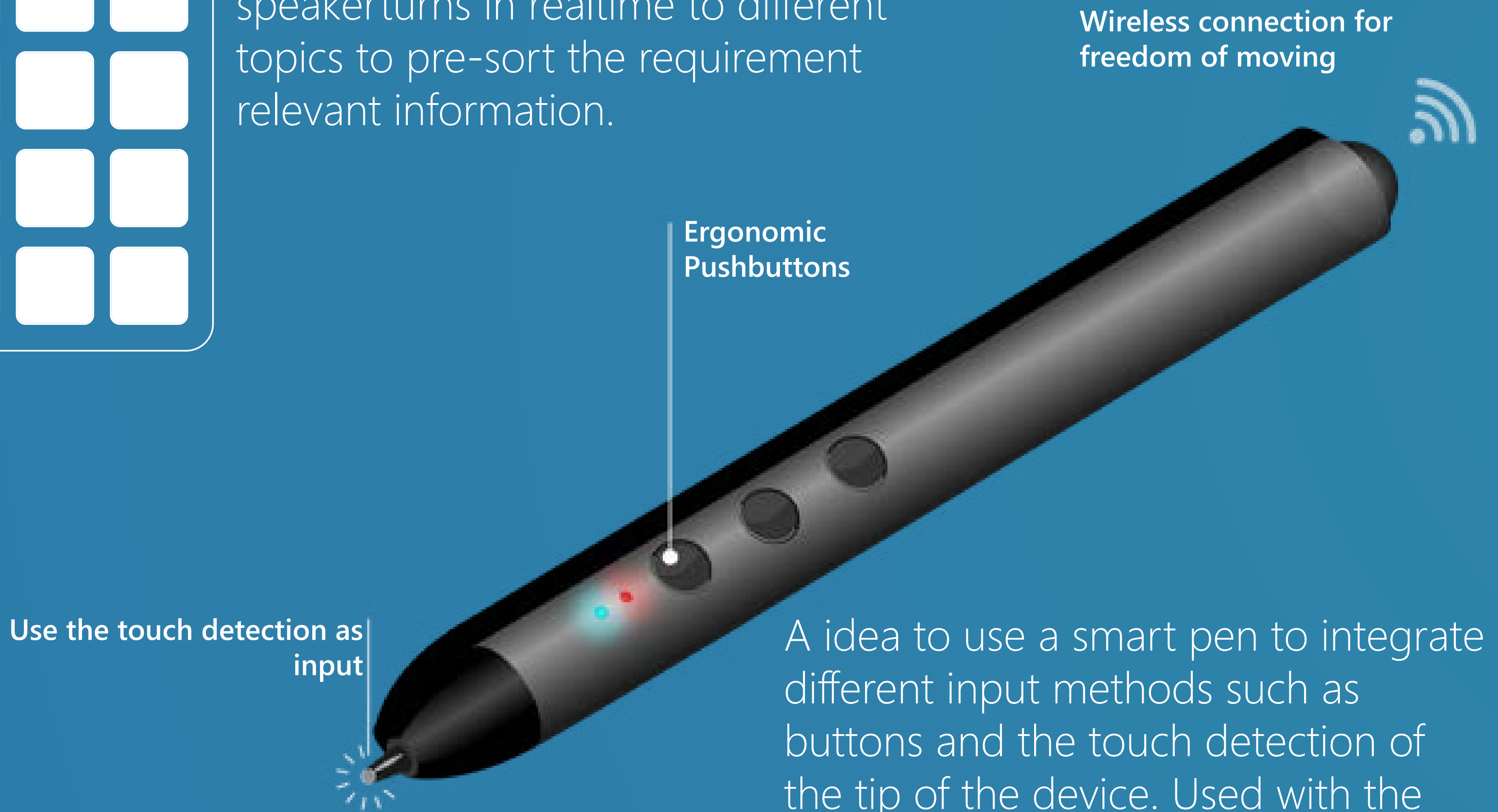
E	F	G	H	I	J	K
identifier.1	time	speaker	text	question	relevant	marked
0	[0:00:08]	Jeroen	Alright, du:	0	0	1
1	[0:00:23]	Tjerk	Op het mo	0	0	1
2	[0:00:39]	Jeroen	Aantal ver:	0	0	1
3	[0:00:48]	Tjerk	Op zijn will	1	1	1
4	[0:01:11]	Jeroen	Natuurlijk,	1	1	1
5	[0:01:27]	Tjerk	Een interfa	1	1	1
6	[0:01:45]	Jeroen	En in die in	0	0	1
7	[0:02:13]	Tjerk	Ja en dat k	0	0	0
8	[0:02:17]	Jeroen	Ja klopt op	0	0	0
9	[0:02:38]	Tjerk	HÃ¨? En de	1	1	0
10	[0:02:38]	Jeroen	HÃ¨ en de	1	1	1

Figure 52: Annotated part of a mock elicitation conversation between me and the company supervisor using both the marker prototype and the system design.

Figure 53: Interaction ideas to integrate with ‘enhanced notetaking’ functionalities.

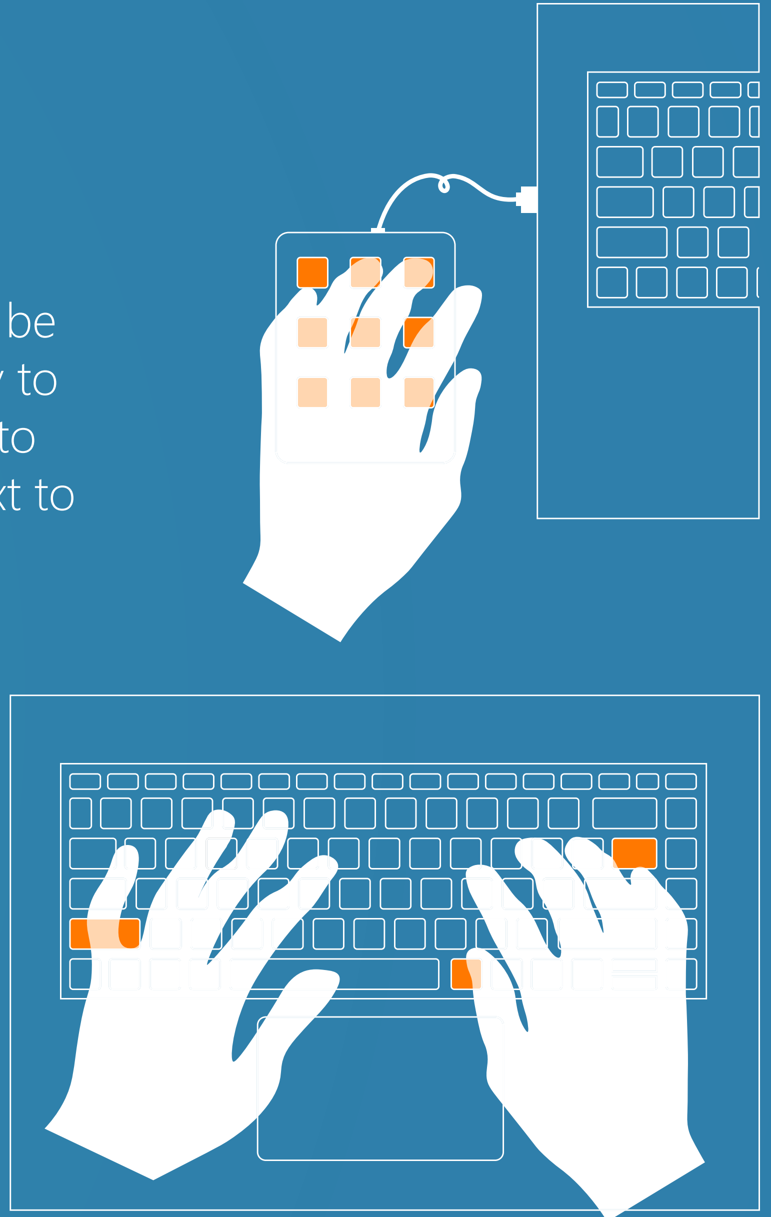


An idea to bind different annotators (such as different topics/keywords) to different buttons to allow to bind speakerturns in realtime to different topics to pre-sort the requirement relevant information.


Wireless connection for freedom of moving
Ergonomic Pushbuttons
Use the touch detection as input

A idea to use a smart pen to integrate different input methods such as buttons and the touch detection of the tip of the device. Used with the interaction of writing.

When using the keyboard it may be more beneficial bind functionality to specific keys on the keyboard or to bind buttons that are located next to the keyboard.



APPENDIX 0: REALIZED SYSTEM DEVELOPMENT MANUAL

System 5: Improving Manual Language Processing

This section discusses the prototype that has been realized in an effort to enhance the notetaking process. This prototype, that uses two Arduinos to function, interacts with the system that has been realized in this thesis. As a first step to enable enhanced notetaking a interaction has been designed that enables Business Analysts to annotate important 1/0 to the current speakerturn by means of an impulse. Providing a impulse 1/0 can be easily detected and needs almost no effort to perform.

The idea on which is built is the notion that BAs will capture important parts of the conversation by documenting keywords and sentences. What if these notes could be linked to the conversation? The word keyword would suggest a factor of importance for a specific part of conversation data. If notes would be made in sequential fashion and the speakerturns could be annotated each time a note would be written down the index of the notes could be matched with the index of the marked/annotated speakerturns. Index 1 of the notes would match for example index 1 of the marked speakerturns. This would provide an overview of all the parts of the conversation that were deemed as important by the analyst, providing a filtered dataset of conversation data. This assumption proved to be flawed during user testing as the Business Analyst doesn't only make notes to capture data but also uses them during the conversation, so the interaction needs to be designed further. There could be iterated on this by for example providing separate areas for data capture where sequence can be applied or iterating with the strategy of providing inputs or allow multiple kinds of inputs, however this needs to be researched further.

Design of interaction

Humans create an order to everything they see where there are a couple of principles about how we perceive hierarchy that hold true for every human (Johnson, 2021). Big > small, high contrast > low contrast, color > grays, red> other colors (Johnson, 2021). There is already an action that humans widely use in their daily live to bring hierarchy to natural language and that is the act of Marking information. By marking, or coloring text people perceive it as more important (color>grays). You can also see the action of marking everywhere from paper documents to digital systems such as excel and word. There have been ideated with two types of marking, marking of relevant conversation moments after the conversation, by marking speakerturns in a GUI (figure 11) or marking conversation speech as being important in the moment. The latter has been considered more impactful as the marking afterwards requires the user to read the whole conversation again to process if a speakerturn is important.

Different impulse types

Three types of impulse have been considered. All types of impulses integrate with the note taking process but each in a different way. First, a button that is integrated in the systems that are already used such as a pen and a laptop. Another being the detection of when a note is being taken, and finally a board with only button inputs relying solely on annotations. Figure 53 presents an overview of each of these types of impulses.

As proof of concepts a button prototype has been created that allows Business Analysts to annotate a speakerturn with important 1/0. In later development this button can be integrated in systems such as smart pens, or in a keyboard for example (figure 55) to make use of the advantages making notes has on digital devices. It can for example be integrated with systems such as remarkable (figure 56), a product that acts as a 'paper tablet'(ReMarkable, n.d.) or drawing tablets that allow direct digitalization of notes or integration with word processing software such as Microsoft word. Besides the ability to provide a input this input needs to be linked to the speakerturns that are being generated. While it may be possible in later development using the Speech Software Development Kit (Speech SDK) of the Microsoft Azure speech services (Speech SDK, n.d.) this could not be realized in the prove of concept. Microsoft Teams itself allows from itself no direct interaction with the speakerturns while they are being generated so the annotation functionality has been realized by synchronizing the prototype with the conversation. By using this method, the device can store all the inputs of the Business Analyst with the time it was inputted. Afterwards the system can compare the timestamps of the inputs and annotate the corresponding speakerturn.

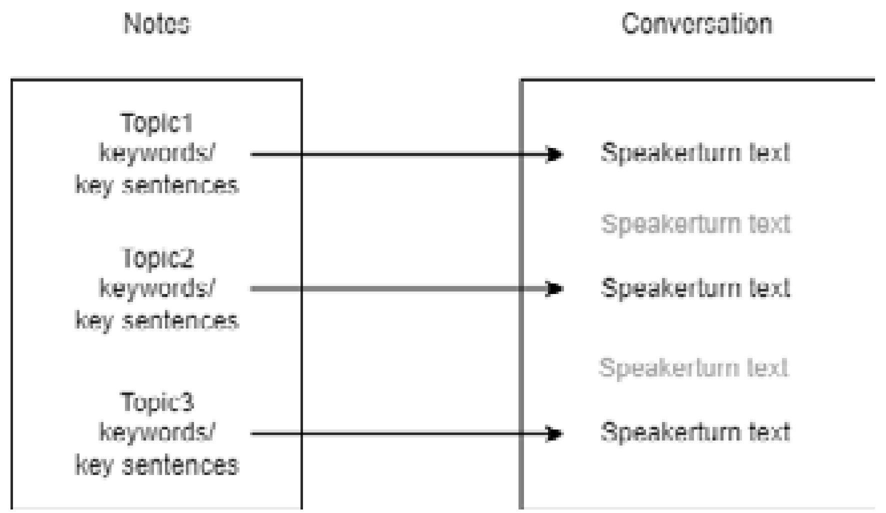


Figure 54: Idea to link notes to marked speakerturns. If notes would be made in sequence and a signal is provided each time a note is made note 1 would match with the first marked speakerturn providing more information on that specific note.

KEYBOARD PROTOTYPE

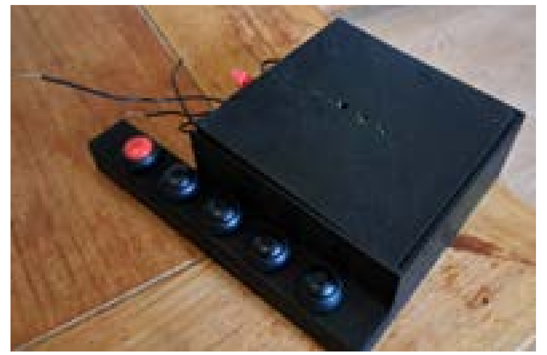


Figure 56: remarkable (ReMarkable, n.d.)

PEN/BUTTON PROTOTYPE

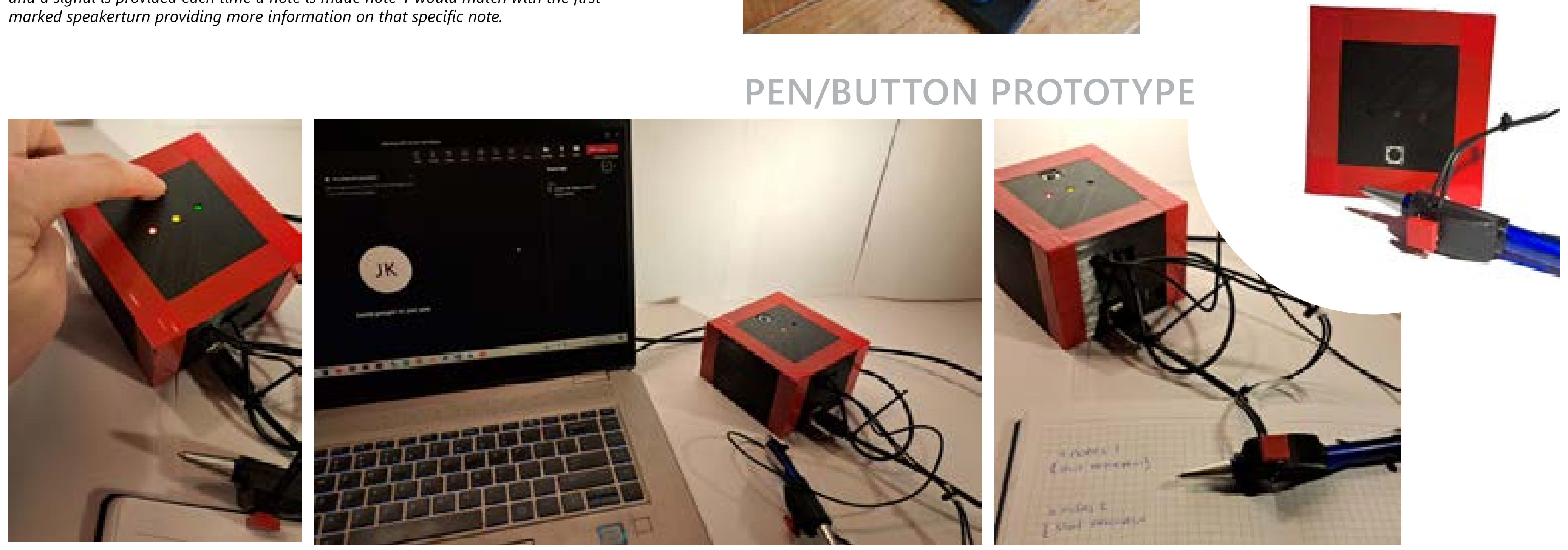


Figure 55: Functional marker prototype ideas integrating inputs into a pen and button board. The functionality of the pen protype has been realized.

APPENDIX 0: REALIZED SYSTEM DEVELOPMENT MANUAL

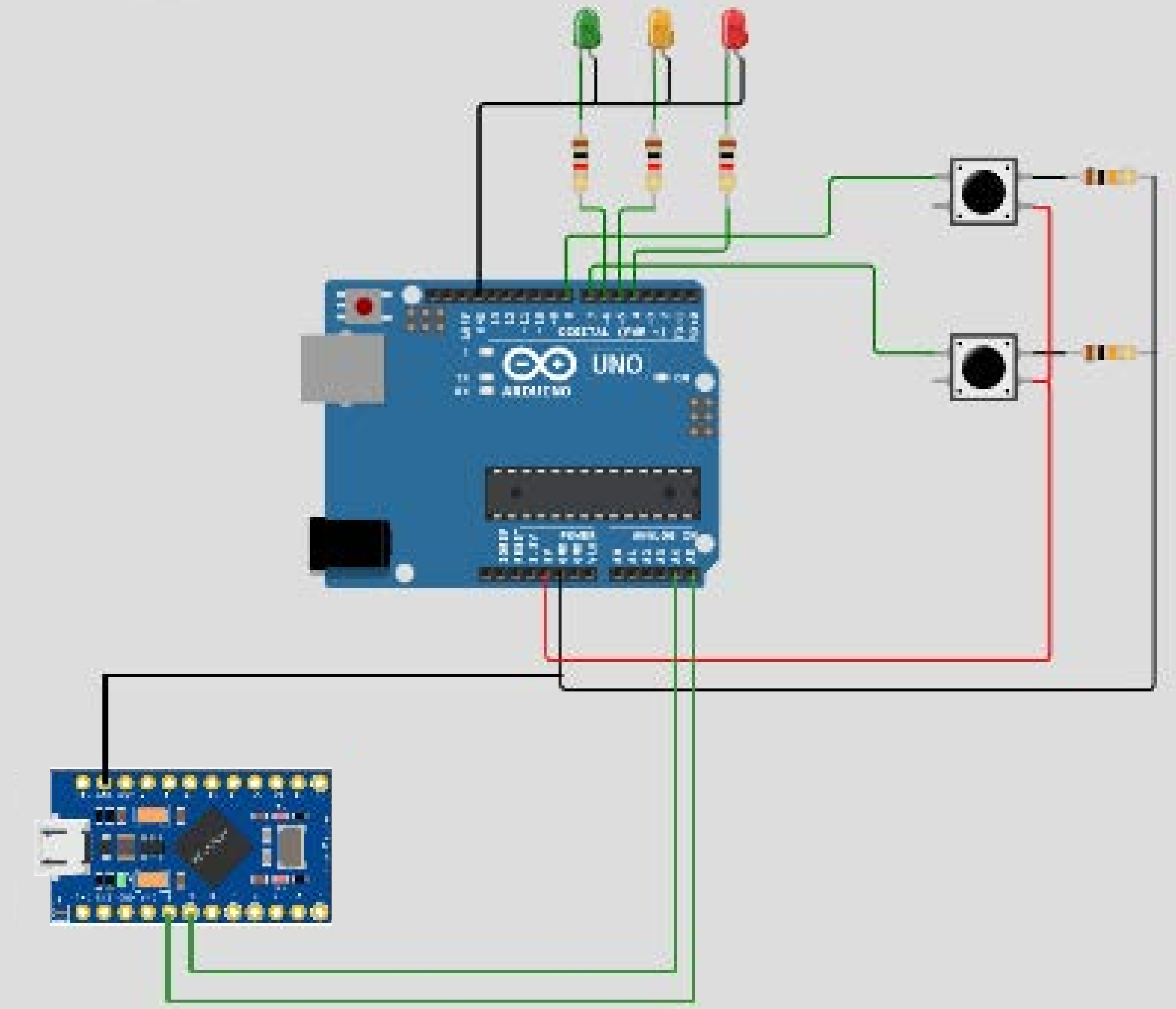
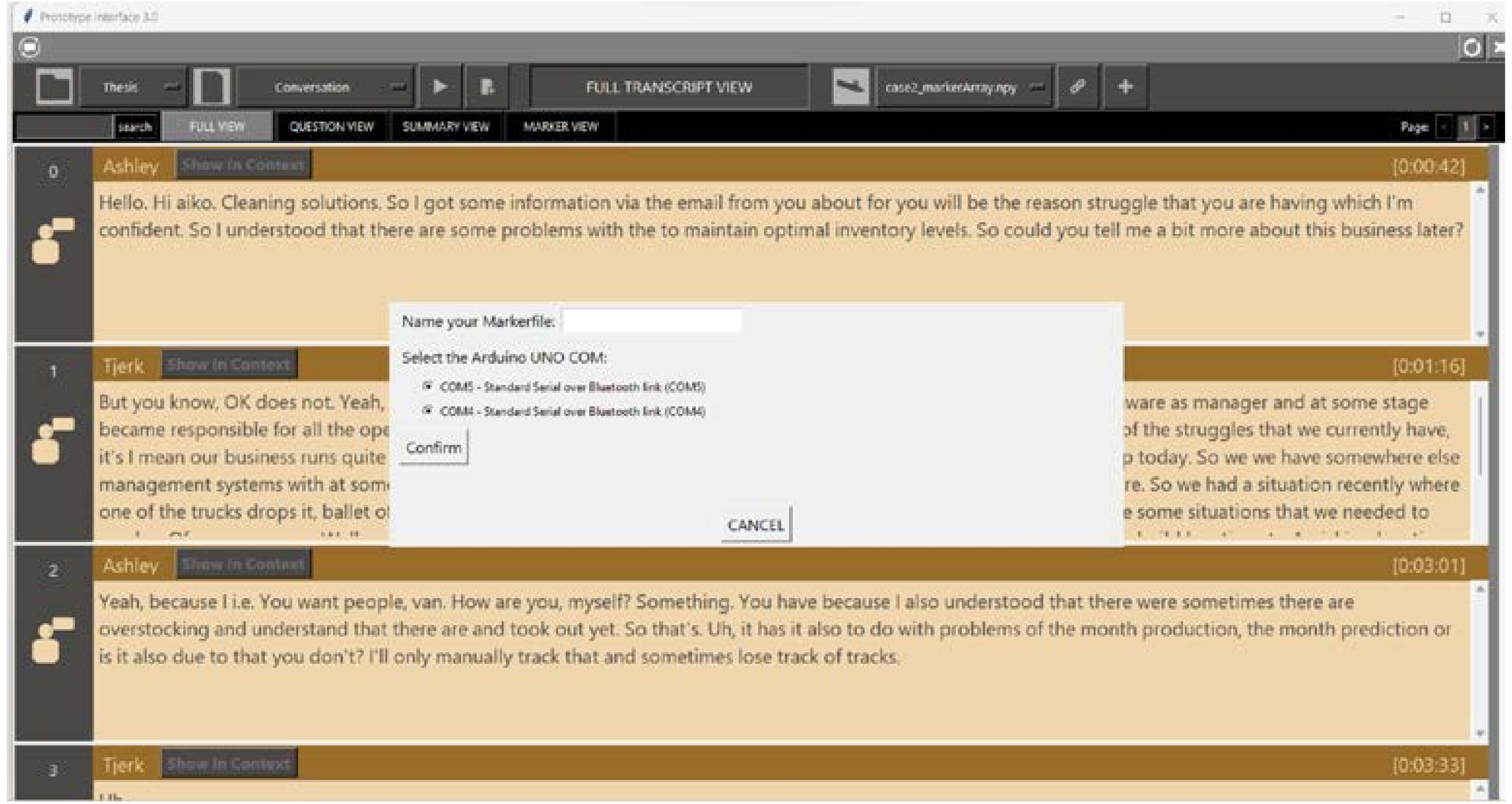


Figure 57: Graphical representation of the hardware used in the marker prototype and how it is build.



0 Ashley [0:00:42]

Hello. Hi aiko. Cleaning solutions. So I got some information via the email from you about for you will be the reason struggle that you are having which I'm confident. So I understood that there are some problems with the to maintain optimal inventory levels. So could you tell me a bit more about this business later?

1 Tjerk [0:01:16]

But you know, OK does not. Yeah, ware as manager and at some stage became responsible for all the ope of the struggles that we currently have, it's I mean our business runs quite p today. So we we have somewhere else management systems with at som re. So we had a situation recently where one of the trucks drops it, ballet o e some situations that we needed to

Name your Markerfile:

Select the Arduino UNO COM:

- COM5 - Standard Serial over Bluetooth link (COM5)
- COM4 - Standard Serial over Bluetooth link (COM4)

2 Ashley [0:03:01]

Yeah, because I i.e. You want people, van. How are you, myself? Something. You have because I also understood that there were sometimes there are overstocking and understand that there are and took out yet. So that's. Uh, it has it also to do with problems of the month production, the month prediction or is it also due to that you don't? I'll only manually track that and sometimes lose track of tracks.

3 Tjerk [0:03:33]

Figure 58: Method to link the marker prototype with the system GUI to allow the communication of time markers.

APPENDIX 0: REALIZED SYSTEM DEVELOPMENT MANUAL

Synchronizing and storing user inputs

As Microsoft Teams allows for almost no interaction during the conversation (no access to speaker, time, or text data while it is being generated, with no ability to remotely start and stop the conversation with external signals) some trickery needed to be applied to allow the synchronisation of the conversation with the prototype. Two arduinos (figure 57) have been used to make this possible in the end where the Arduino Uno keeps track of the time and the Arduino Pro Micro allows the external start and end of the meeting recording functionality. By setting the current time on 0:00:00 when the start signal is given the time of the device and the recording can be synched. These signals of the Arduino Pro Micro can be realized as it can provide keyboard inputs (by connecting it to a laptop using usb), making it able to input shortcuts that could be used to manage MS Teams functionalities.

There was however no shortcut available to start and stop a recording (what would be preferable as this would make the solution a lot more usable in practice). A convoluted solution has been used by needing the user to click on the '...' button and press escape. With the start signal of the user the Arduino Pro Micro inputs a sequence of multiple keyboard presses containing <tab> and <enter> presses making it able to start and stop the recording. This proved to be a method that is very difficult to setup and is not suited to be used by users in practice at it can easily go wrong needing the meeting to be restarted, or no inputs are able to synch. It can however currently be used in controlled test settings where a researcher sets up the device.

The storing of the inputs is also realized by a usb connection, this time between the Arduino Uno and the laptop. Two algorithms have been designed to make the functionality possible, a Arduino algorithm that is run when it is powered and a python algorithm that is initiated using the system GUI design. To use the prototype the user first connects both arduinos by usb to a laptop that will be in the MS Teams meeting. In the system GUI the user needs to select the Create Markerfile button (see system 2: GUI, figure 34, 7). This will open a pop-up (generate_markerfile_pop_up.py) where the user can name the Markerfile that will be generated and selects the correct COM (Arduino Uno), pressing confirm will connect the Arduino prototype to the system design and create_marker_array.py will start running in the background.

Now to the Arduino prototype. A casing has been 3d modelled and FDM printed to secure all necessary hardware. The prototype consists of a Arduino UNO and PRO MICRO, 5 1k resistors, a marker button, a start/stop button, and a red, a yellow, and a green LED, wired as shown in figure 57. When the Arduinos are connected (and powered) to the laptop the red LED turns on indicating that the system is powered (state 1). The internal clock of the Arduino UNO starts when it is powered, pressing the start/stop button resets the time to 0 and the Arduino PRO MICRO is signaled to start the meeting recording. This moves the system to state 2 and turns on the orange LED. Now each time that the marker button is pressed the system will access the current time and convert this in a string formatted as expected, '[0:04:10]' for example, a button press on minute 4 and 10 seconds. The system stores a list of all the markers that it has generated, each time sending a list of all markers that have been created this conversation over serial communication to the system design (and blinking the green LED). If the start/stop button is pressed again (the green LED blinks twice and then moves the system back to state 1) the marker 'stop' is send via serial communication (and removed from the marker list afterwards) to the system design indicating that it can store the markers into the system as a .npy file (in the markerfile_unsorted folder) , also the meeting recording is ended. Figure 59 shows the marker array that is stored in the system.

After the user has imported the corresponding conversation into the system the created marker array can be linked to the conversation. Using the marker file overview in the GUI (system 2: GUI, figure 34, 5) the correct markerfile can be selected and linked using the link markerfile button (system 2: GUI, figure 34, 6). When this button is pressed the create_markerfile.py algorithm is run, storing the markerfile in the folder 3) markerfiles of the corresponding conversation, accessing the output processed_excel_summary.xlsx, annotating the speakerturns that have been marked (the time marker compared to between which speakerturn times it is positioned). To give an example, if there is a speakerturn with the time [0:00:50] and another with [0:02:30], the speakerturn with [0:00:50] will be annotated as marked with a time marker of [0:01:30] as it occurs before the start of the next speakerturn. After annotating the output is again stored under a different name in the 3) markerfiles folder (appendix chapter 16).

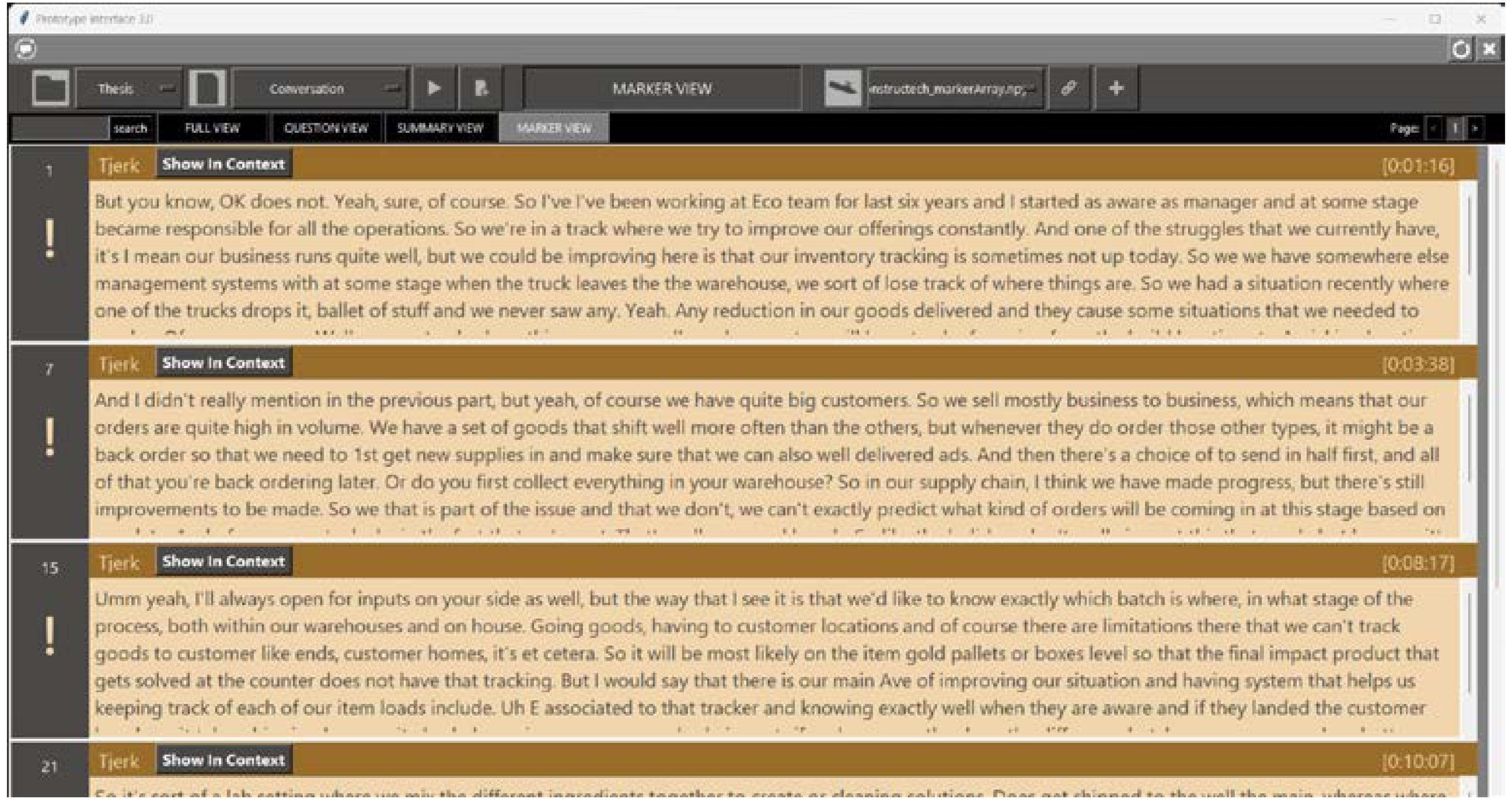
Viewing the marker outputs in the GUI

When the markerfile is correctly linked and processed the outputs of the marker functionality can be viewed in the marker view in the system GUI. Just as the question and summary view the speakerturns can be viewed in context (see system 2:GUI).

```
“NUMPY v {'descr': '<U10', 'fortran_
order': False, 'shape': (10,), }
  [ 0 : 0 1 : 4 0 ]
  [ 0 : 0 2 : 3 2 ]
  [ 0 : 0 4 : 5 3 ]
  [ 0 : 0 8 : 2 5 ]
  [ 0 : 0 9 : 1 6 ]
  [ 0 : 1 0 : 1 1 ]
  [ 0 : 1 4 : 2 5 ]
  [ 0 : 1 5 : 0 1 ]
  [ 0 : 1 8 : 5 1 ]
  [ 0 : 2 0 : 4 8 ]
```

Figure 59: Marker array output (.npy).

APPENDIX 0: REALIZED SYSTEM DEVELOPMENT MANUAL



1 Tjerk Show In Context [0:01:16]

!

But you know, OK does not. Yeah, sure, of course. So I've I've been working at Eco team for last six years and I started as aware as manager and at some stage became responsible for all the operations. So we're in a track where we try to improve our offerings constantly. And one of the struggles that we currently have, it's I mean our business runs quite well, but we could be improving here is that our inventory tracking is sometimes not up today. So we we have somewhere else management systems with at some stage when the truck leaves the the warehouse, we sort of lose track of where things are. So we had a situation recently where one of the trucks drops it, ballet of stuff and we never saw any. Yeah. Any reduction in our goods delivered and they cause some situations that we needed to

7 Tjerk Show In Context [0:03:38]

!

And I didn't really mention in the previous part, but yeah, of course we have quite big customers. So we sell mostly business to business, which means that our orders are quite high in volume. We have a set of goods that shift well more often than the others, but whenever they do order those other types, it might be a back order so that we need to 1st get new supplies in and make sure that we can also well delivered ads. And then there's a choice of to send in half first, and all of that you're back ordering later. Or do you first collect everything in your warehouse? So in our supply chain, I think we have made progress, but there's still improvements to be made. So we that is part of the issue and that we don't, we can't exactly predict what kind of orders will be coming in at this stage based on

15 Tjerk Show In Context [0:08:17]

!

Umm yeah, I'll always open for inputs on your side as well, but the way that I see it is that we'd like to know exactly which batch is where, in what stage of the process, both within our warehouses and on house. Going goods, having to customer locations and of course there are limitations there that we can't track goods to customer like ends, customer homes, it's et cetera. So it will be most likely on the item gold pallets or boxes level so that the final impact product that gets solved at the counter does not have that tracking. But I would say that there is our main Ave of improving our situation and having system that helps us keeping track of each of our item loads include. Uh E associated to that tracker and knowing exactly well when they are aware and if they landed the customer

21 Tjerk Show In Context [0:10:07]

Figure 60: Marker view output. All speakerturns that were marked as important by the Business Analyst during an elicitation conversation.