Design of a conversational assistant for standardized issue description in manufacturing

Industrial Design Engineering
Master's Thesis by Boris Hadzisejdic

TU Delft

Design of a conversational assistant for standardized issue description in manufacturing

November, 2021

MSc Integrated Product Design
Faculty of Industrial Design Engineering
Delft University of Technology

Author
Boris Hadžisejdić

Supervisory team
Chair: Dr. Z. Rusák
Assistant professor Computer Aided Design Engineering
Industrial Design Engineering
Delft University of Technology

Mentor: Ir. S. Kernan Freire
PhD candidate Sustainable Design Engineering
Industrial Design Engineering
Delft University of Technology

Company mentor: B. Kiss
Director OpEx, EMEA & APAC
Diversey BV

I. Acknowledgements

First of all, I want to express my gratitude for the supervision and guidance by Dr. Zoltan Rusák and Ir. Samuel Kernan Freire. Zoltan, I am grateful for the opportunities and the time you have given me. Thank you for the thoughtful insights and moral support. Samuel, I am thankful for your availability and the insightful feedback you were able to consistently provide to me. I hope I could have contributed to your PhD at least a fraction of what you have contributed to my project. Thank you for the free rides to Enschede and I wish you all the best with your PhD and after.
Thank you both for your enthusiasm, energy and banter. It was more than appreciated.

I want to thank Barnabas Kiss for the discussions and ideas on the prototype that moved my project forward. I want to specially thank you for convincing maintenance staff to participate in the tests when they initially were reluctant to participate.

Thank you to the entire COALA team with assisting me in my journey. For taking the time to talk and brainstorm with me, for helping with testing and providing me with valuable insights in the development of the project. Special shoutouts to Jasper, Santiago, Soude, Sarath, Pim, and Jeroen!

Thank you to Stefan and Mina from BIBA. Your guys' tendency to share insightful knowledge is something to admire.

Thank you to Felix and Ronne for the assistance in my project. Felix, I still owe you a beer. Ronne too, but I know you do not drink beer. Maybe some wine? Thanks go out to all my friends for having to put up with me in the past years.

I want to thank my parents, Bosa and Dalibor, for the unconditional love and for their part in making me the man I am now. Thank you for moving me out and back in again and moving me out again and back in again. The seemingly limitless supply of great food whether living in Rotterdam or Delft is very much appreciated. Soon, I think it will be time to move out again. Eliza, although not officially family, this section is dedicated to family. This is my way of saying you are family to me. Thank you very much for your support. I want to thank my brother, Nino. Although I wished you were here during the last months of this journey, your moral support was felt all the way from America. Looking forward to seeing you again in December.

A very special thank you is reserved for my girlfriend, Evy. Your love and support was crucial in achieving all this. You keep pushing me forward and continuously elevate me. Somehow, you have this skill of always putting a smile on my face, no matter the circumstances. I am very lucky to have you by my side.

Boris Hadzisejdic
Rotterdam, November 2021

II. Abstract

This thesis describes the design and development of a conversational assistant (CA) for standardized issue descriptions in manufacturing. The CA was specifically targeted at the acquisition of data on standardized issue descriptions and at the same time reduce operator friction with the system. CAs can be used to handle and process repetitive data, such as gathering data at scale. For Diversey specifically, the conversational assistant can gather data on issues and in the future on root causes, solutions and best practices. At Diversey, agile manufacturing is applied to the production process. Due to the complexity of this approach, the production process has many stoppages. Context analysis at Diversey showed that operators primarily resort to their own intuition and experience to resolve stoppages. Evaluations with operators showed that the system for data acquisition on stoppages currently in place, is an inconvenience and a time sink. The data that was captured with the system was often ambiguous, incomplete and non-descriptive. However, such data can be of significant value to the company as it can be used for process improvement of the production line.

A Wizard-of-Oz-like experiment was conducted with the operators where the researcher roleplayed a CA for acquiring issue description, which resulted in 71 issue capturing dialogues. Even though these dialogues were filled with implicit and explicit knowledge, without proper understanding of the context, the dialogues were too chaotic and unstructured for a machine learning (ML) algorithm to understand and process.

For a complete issue description, that a ML algorithm can process, several key entities were identified: *machine location, machine component, machine component state, product component* and *product component state.* Furthermore, several challenges were found related to the speech patterns of the operators. Two of these were tackled in the prototype: Usage of synonyms and pronouns.

With the open-source framework Rasa, a prototype was developed for a CA that would capture product related issues through a form structure into a database. The form requires the following entities: *machine location, product component* and *product component state.* The form is used to guide the user to capture good, structured data, while allowing some flexibility in the input. With the synonym and pronoun handling features, some of this flexibility is realised.

A conversational flow test was conducted to test and improve the CA prototype. Although failing the accuracy requirements set for intents by 7.4% and for entities by 2%, it provided an indication that with some information on the functionality of the CA, the participants were able to correctly capture the set issue. With some training of the operators, it would be feasible to implement such a CA in a manufacturing environment.

Through contextual filtering, the model can filter towards the input of the user, allowing for only context-specific issue to be captured. With a qualitative test it was concluded that this feature decreased the duration of issue capturing. The contextual filtering and issue description control provide feedback to the user whether correct information is captured.

Finally, a form for capturing unknown issue is proposed to expand upon the training data of the CA and to be able to properly acquire data. With this form, operators themselves can capture unknown issues and expand the knowledge base of the CA. Although some supervision will probably be needed to improve the quality of the data captured from this form.

III. Glossary

This glossary contains definitions of the terms used in this report to elaborate on the meaning and context of these otherwise generic terms:

- Cognitive advisor = a conversational AI which can assist a user in their actions and provide recommendations to the user
- Conversational AI = refers to technologies, such as chatbots or conversational assistants which users can talk to
- Conversational assistant (CA) = a conversational AI for assisting a user in their actions
- Conversation-driven development (CDD) = the process of listening to your users and using those insights to improve your AI assistant.
- Cyber-physical-system (CPS) =
- Dialog manager / Dialog management = is a component of a CA responsible for the responses and the flow of the conversation
- Hybrid team = a team of multiple agents, which can be either humans or machines, that work together interdependently
- Industry 4.0 = A revolution in manufacturing where interconnectivity, machine learning and real-time data are key factors.
- Machine Learning (ML) = a sub-field of artificial intelligence, made up of a set of algorithms, features, and data sets that continuously improve themselves with experience. As the input grows, the AI platform machine gets better at recognizing patterns and uses it to make predictions.
- Natural Language Processing (NLP) model = a model that can process human language and interact with the user
- Natural Language Understanding (NLU) model = a model that can understand human language and extract the context and intent
- Pipeline = document in Rasa where components of the NLU and dialog management can be configured
- Rasa = platform for developing conversational assistants and chatbots
- Rasa X = an API tool for Conversation-driven development
- SKU = Stock keeping unit, unique number to identify a certain product
- Training data = the NLU, rules and stories used to train Rasa chatbots and conversational assistants

Contents

1. Introduction

This is a design project for developing a conversational assistant (CA) for standardized issue description in manufacturing, which belongs to the overarching COALA project. The solution is designed specifically for the Diversey context of an operating line where 5 and 10 litre canisters are labelled, filled and packaged and the findings will be used as a basis for other manufacturing contexts. In chapter 1.1 the background of the project is provided and in 1.2 the problem is defined.

1.1 Background

In manufacturing, efficiency is a key motivator for development of new technologies as it provides a competitive advantage in the market. In Industry 3.0 (I3.0) automation of production processes was central to this competitive advantage. However, in the advancements of production, Industry 4.0 (I4.0) is introduced in which continuous development is taking place. Although I4.0 builds on the developments of I3.0 technology-wise, the transition is a gradual development that has taken place over the last few decades (Torn & Vaneker 2019). Where automation was key, autonomous decision making is becoming central. Predictive measures such as demand forecasting are replaced for on-demand manufacturing. I4.0 combines physical manufacturing and operations with smart digital technologies, machine learning and big data to create a comprehensive, integrated and connected habitat for companies that focus on manufacturing and supply chain management (Epicor, n.d.).

This focus on interconnectivity, real-time data and machine learning is only pushed further with the developments of artificial intelligence (AI). The European Union is pushing for Europe to become a world leader in AI by increasing public and private investments. With the research and innovation funding programme H2020, the EU is enabling AI development and implementation on both a technological and societal level. The European Commission (2020, October 2) claims that the application of AI has the potential to significantly enhance society and help humans achieve climate and sustainability goals by bringing high-impact innovations.

One such H2020 EU project is COALA, a project with 17 partners including TU Delft and three business partners: Whirlpool, Diversey and Piacenza. TU Delft and the COALA partners work together to develop a human-centered voice-enabled Cognitive Advisor that provides support to operators in manufacturing for situations characterized by cognitive load, time pressure and little or zero tolerance for quality incidents. The COALA Cognitive Advisor will be a form of conversational AI, which can communicate with the operators and assist them on the production line with cognitive tasks. Conversational AI use natural language understanding (NLU) and processing (NLP) to understand and respond to the input provided by the user.

Diversey is a company in the transition from I3.0 to I4.0. It applies agile manufacturing in most of its facilities in Europe. Due to agile manufacturing, Diversey was able to provide on-demand supply of a variety of items to its customers, as well as make them in relatively small batches (5-10 tons). However, this method requires change-overs of the production line to switch between different products on the line. Furthermore, due to the sheer complexity of the line, many stoppages take place. The change-overs in combination with the downtime due to stoppages takes currently up 40-45% of the total production time on average (COALA, personal communication, 2021).

It is mostly up to the operators to resolve such stoppages as quickly as possible. Current approaches for solutions are reliant on the operator's explicit and implicit knowledge which have to be applied directly. As one can imagine, such practices are difficult to capture. Operators have to report on the issues in between maintaining the production line. It is often done after the fact but not directly, and often times it is either filled in rather hastily, ignored for longer periods of time or not filled in at all. For the operators, data collection on issues and solutions is low on the list of importance. Why would they have to capture something that the operators

already know and apply? No wonder that the current system in place for data collection on issues at Diversey provides data that is often than not ambiguous, incomplete and non-descriptive.
However such information is required by Diversey for process improvement. Furthermore, it is desired to be of good quality. Next to that, the COALA Cognitive Advisor will need training data on conversations to improve the Advisor, and it needs a lot of data. To harmonize these datasets and improve the data, standardized terminology will be needed.

The solution for this thesis is designed specifically for the Diversey context of an operating line where 5 and 10 litre canisters are labelled, filled and packaged. Through a human-centred design approach it is looked at what the operators need, while simultaneously improving upon the data quality for Diversey and delivering conversational data for the Cognitive Advisor.

1.2 Problem definition

The duration of the downtime due to a stoppage mentioned in the previous section, is partly dependent on operator's explicit and implicit knowledge and response time of resolving the issue. With the introduction of COALA's Cognitive Advisor this responsibility can be split between operator and COALA. Through recommendations, COALA could provide operators with best practices for resolving issues.

However, in order to be able to communicate such recommendations, COALA needs to learn. It needs to learn and collect data on issues that can arise, what the root of the cause is, how to solve the issues and it needs to learn the operator's explicit and implicit knowledge.

Furthermore, the system in place at Diversey for data collection on issue description leaves room for improvement. The system allows operators to collect machine specific issues and provide descriptions. However, this system is often times ignored by the operators, and when data is captured, it is often lacking information or confusing as terminology is not aligned amongst different Diversey teams. This information however is crucial for reducing downtime and increasing production efficiency.

For this thesis, a human-centred conversational assistant was proposed which can collect issue descriptions on the Diversey production line from operators through dialogue. The data that is created through these dialogues should contain standardized terminology for the description of the issue. The contributions to the COALA project provided with this thesis were firstly a collection of conversational data that can be used for further development of the assistant, a minimum viable product of a conversational assistant and features meant to improve the assistant.

2. Assignment & Project Scope

The goal of this project is to develop a human-centred conversational assistant for acquiring standardized issue descriptions from operators on the high-priority Diversey production line. To achieve this goal, the following elements are considered: Human-Centred Design, Natural Language Understanding (NLU), Data Acquisition, Domain specific terminology and User acceptance.

The following research questions should be answered in order to achieve the project goal:

- Why use a conversational assistant for data acquisition in manufacturing?
- What is the added benefit of a conversational assistant for data acquisition in manufacturing?
- How can a conversational assistant be designed for data acquisition of standardized issue descriptions?

The expected outcome of the project is a service prototype which is able to collect issue descriptions from operators during issue handling; it will be optimized for efficient data collection. When the information is missing, the prototype should be able to have a dialog with the operators to further clarify and complete the information.

2.1 Approach

To answer the research questions and have a service prototype, the Lean Startup Method (LSM) in combination with the Riskiest Assumption Test (RAT) was utilized. With this combinational approach uncertain assumptions can be quickly tested in sprints and allow for a flexible approach towards learning what kind of conversational assistant is needed for implementing into the context of Diversey and how it can be implemented.

The full process is shown in figure 2.1. The process is divided into four phases (table 2.1):

Phase	Riskiest assumption(s)
Design thinking	-
Sprint 1: Conversational data	By having human-like conversations, it is possible for a CA to extract high quality issue descriptions while ensuring positive user experience.
Sprint 2: MVP prototype	Having a frame-based strict pattern for capturing data will provide structured data while allowing for flexible user input.
Sprint 3: Prototype features	It’s possible to create a CA that uses and presents standardized terminology while reducing friction between user and CA.

Table 2.1: Phases in the process with riskiest assumptions per sprint

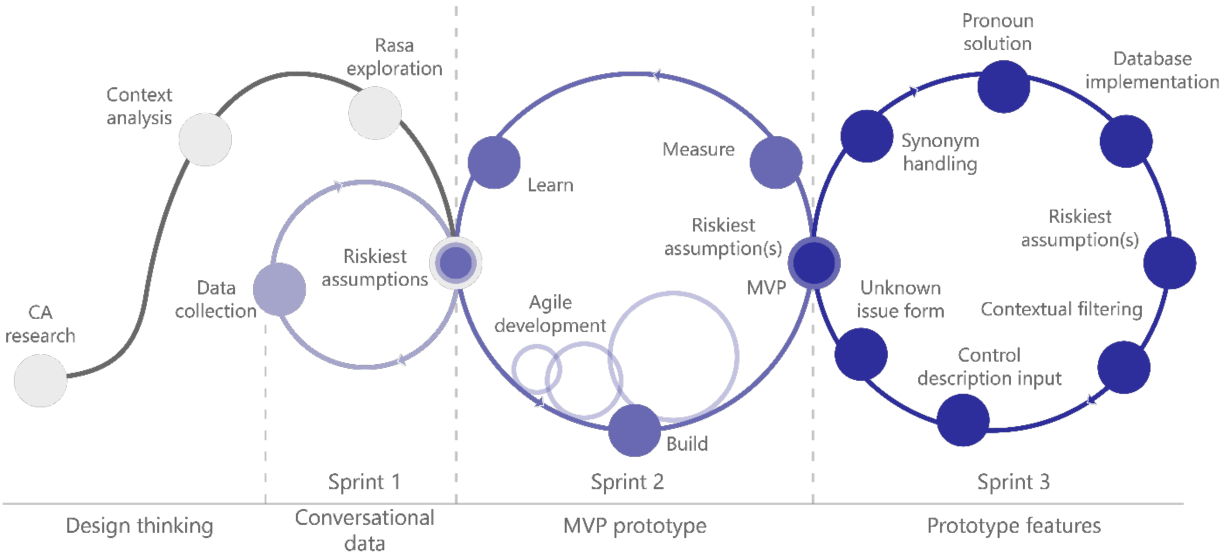


Figure 2.1: LSM with RAT-sprints for developing a conversational assistant for manufacturing
CA* = Conversational assistant
MVP = Minimal viable product**

The Design Thinking phase encompassed the necessary theory and context to go into the sprints. Research into conversational assistants and the specific context domain for the use case of the prototype was analysed. The Design Thinking phase mainly focused on answering the first two research questions. With the sprints, the riskiest assumptions were tested which were used to answer the third research questions.

In sprint 1, it was tested whether human-like conversations are needed between CA and operator to extract high quality issue descriptions. Human-like conversations can be achieved with an agent-based dialogue management, section 3.2 describes the characteristics of conversational AI such as frame-based/agent-based dialogue management.
Noteworthy here is that the riskiest assumption failed and was adjusted for sprint 2 to a system that would provide more structure in the output but maintain some flexibility in the input. The system should allow for the operators to present the issues that would come from natural human-like conversations in the input

Sprint 2 tested whether a frame-based CA prototype would allow for capturing issue descriptions in such a way where the output is structured, while the input is allowed to be flexible. Agile development was applied to create quick iterations on the prototype.

In sprint 3 prototype features were explored which could improve the data acquisition. During the first sprint, possible challenges for the CA were discovered. In sprint 3, attempts were made to resolve two of these challenges: synonym usage and pronoun usage. Other features explored were: Writing issue descriptions to a database, contextual filtering, control description input and an extra form for capturing unknown issues.

3. Conversational assistant

This chapter introduces the concept of a conversational assistant, discusses the types of conversational AI and the terminology used for this report, discusses why we should use conversational assistants in manufacturing and how it should work.

3.1 What is a conversational assistant?

A conversational assistant is a type of conversational AI that is designed to complement, not replace, the abilities and functions of the person it is assisting. The assistant does not substitute the person's job, rather it optimizes the individual's performance in the task (Parayno, 2020). Common applications of conversational AI are customer support chatbots (frequently-asked-questions [FAQ] bots) or virtual assistants such as Apple's Siri, Amazon Alexa, Samsung's Bixby or Google Home. With various machine learning components, such as Automatic Speech Recognition (ASR), Natural Language Understanding (NLU), Dialog manager and Natural Language Generation (NLG), the conversational assistants are able to understand and respond to the user.

The level of complexity of conversational AI can differ drastically. Customer support chatbots are there to answer simple frequently asked questions to the user while a virtual assistant such as Amazon Alexa needs to be able, for instance, to play a requested song on demand and be able to set a timer for 6 minutes. While the first can have a pre-determined strict path it guides the user (up to a point where it possibly cannot assist the user anymore and, if implemented, forwards the user to contact a human), the second needs to be flexible and be able to do a wide range of capabilities.

3.2 Types of conversational AI

This section is dedicated to clarifying the types of conversational AI. Table 3.1 provides an overview of the characterizations of conversational AI.

Characteristic	Types	Description
Dialogue management	Finite-state	The user is taken through a series of pre-determined steps and the user's input is limited to single predefined words or phrases
	Frame-based	The user is asked questions that enable the system to fill slots in a template in order to perform a task
	Agent-based	Natural, dynamic dialogue that draws on preceding context as well as both sides' actions and beliefs
Goal-orientation	Goal-oriented	Goal-oriented (or task-oriented, closed-domain) conversational AI assist users in completing a specific task
	Non-goal oriented	Non-goal-oriented (open-domain) systems interact with users to provide reasonable responses and entertainment
Dialogue initiative	User	The dialog is led by the user
	Assistant	The dialog is led by the assistant
	Mixed	Both the user and the assistant can lead the conversation
Input modality	Voice-based	The user uses spoken language to interact with the assistant
	Text-based	The user uses written language to interact with the assistant
	Button input	The user uses buttons to interact with the assistant
	Mixed	The user uses a combination of spoken language, written language and buttons to interact with the assistant
Output modality	Voice-based	The assistant interacts with the user by voice
	Text-based	The assistant interacts with the user with text
	Visual	The assistant uses non-verbal communication like images, graphics, facial expressions or body movements
	Mixed	A combination of the above

Table 3.1: Characterization of conversational AI (adapted from Kernan Freire (2020), and Laranjo (2018))

Amazon's Alexa is a form of conversational AI that is a voice-based, frame-based non-goal-oriented assistant with user dialogue initiative. A FAQ chatbot as described in section 3.1 is a finite-state goal-oriented assistant.

Finite-state assistants are simple chatbots that allow the user to have only single predefined words or phrases as input. An example of such an assistant is a simple travel inquiry system. An example below (Böhm, 2002):

> System: What is your destination?
> User: London.
> System: What is your origin?
> User: Amsterdam.
> System: What day do you want to travel?
> User: Thursday.
> System: What is the departure time?
> User: ...

Figure 3.1: Start of a conversation of a travel inquiry system with finite-state system

Limiting the user to predefined inputs is both an advantage and disadvantage of the system. The advantages of structured dialogues are that it is suitable for well-structured tasks, the chance of the conversation failing is lowered and the output is expected to be more structured than for the other types of conversational assistants. The disadvantage is that it is not flexible in its conversation as it is unable to cope with more complex dialogues where the user does not follow the strict path.

With a frame-based systems the user is not limited to predefined inputs, however the output is. Certain information should be extracted from the user's input to fill slots in a template in order to perform a task. Below in figure 3.2, is the start of a conversation of a travel inquiry system with frame-based dialogue management:

> System: What is your destination?
> User: London if possible.
> System: What is your origin?
> User: Somewhere in the Netherlands.
> System: The following airports are available for travel to London:
> - Amsterdam
> - Rotterdam
> - Eindhoven
>
> What airport do you prefer?
> User: Eindhoven.
> System: What day do you want to travel?
> User: ...

Figure 3.2: Start of a conversation of a travel inquiry system with frame-based system

In the example the system is aware of the user input and is even able to store some information; the system stores London as a destination and recognizes that the user wants to departure somewhere from the Netherlands and therefore provides airports in the Netherlands from which one can travel to London. As shown, the input is flexible however the user is still guided towards finishing the assistant's task.

An agent-based system has natural and dynamic dialogues that draw on preceding context as well as both sides' actions and beliefs. In simpler words, it converses as a human would. An example of a travel inquiry system with an agent-based dialogue system is shown in figure 3.3.

System: What is your destination?
User: I don't know.
System: I can help you decide! But I will need some more information. Want to proceed?
User: Sure.
System: When would you like to travel?
User: This spring.
System: Great. I have heard Spain is lovely in this time of year.
User: I would prefer to travel somewhere in Asia.
System: ...

Figure 3.3: Start of conversation of a travel inquiry system with agent-based system

As you can imagine, the dialogue can go in many directions from this point. The system can keep providing destinations, or it can ask more questions to get more specific information. Maybe the user changes their preference mid conversation and opts to travel to America. An agent-based system must be flexible to handle such conversations. The input can have a wide variety of user utterances while the output can have many different outcomes.

For sprint 1 the riskiest assumption was made that the conversational assistant had to communicate human-like and the initial characterization was assumed to be an agent-based goal-oriented assistant with mixed dialogue initiative. The assumption failed as the desired result of the conversation needed to be structured output for issue descriptions and was therefore adjusted for sprint 2. However, through context analysis it became clear that a finite-state dialogue management was also not desired. The system currently in place for issue descriptions (as discussed in section 4.4.6) was too restrictive on the operators. So I have opted for a frame-based dialogue management which allows for a mix of strictness in the information that needs to be gathered (with forms, sections 5.6.1 and 7.3), but also allows for flexibility in the language and words that is used in the user's input (by using NLU section 5.3, 8.2 and 8.3).

3.2.1 Cognitive Advisor vs. Conversational assistant

COALA's Cognitive Advisor is intended as a frame-based goal-oriented system with a voice-based input and output modality where both the user and the Advisor can lead the conversation. Furthermore, the goal is to build the Advisor up from a frame-based dialog management system to an agent-based dialog management system.

The conversational assistant for this report however, was a frame-based goal-oriented system with a text-based input and output modality where the user leads the conversation. Buttons were also implemented to be able to quicken the interaction when the user's information can be structured.

3.3 Why use conversational assistants?

Why would it make sense to implement a conversational assistant into the Diversey use case? The benefit of conversational assistants is with handling and processing repetitive data, such as addressing customer inquiries or gathering data at scale. Current methods of data gathering at Diversey still leave a lot of room for improvement (section 4.4.6) Issues are often either not captured at all or incorrectly captured, a conversational assistant can improve such practices. Furthermore, an assistant can provide recommendations to an operator for best practices on the production line. Chacón et al. advocate this:

> Currently, Industry 4.0 signifies a great opportunity for operators to become a part of the new manufacturing systems. On the one hand, operators generate information and data to programme machines and robots and optimise process flows; on the other hand, they receive useful support for their work as well as effective cooperation with intelligent systems. This bidirectional dialogue allows new types of powerful interactions between operators and machines. Hence, a new kind of workforce should be trained in order to obtain a significant impact on the development of the industry. (Chacón et al., 2020)

The information and data the operators generate, the conversational assistant can use as training data and when it has learnt all this information, it can provide support to them through recommendations on best practices.

3.4 How will it work?

The use case of the conversational assistant is to capture issue descriptions during issue handling of a stoppage on the production line. The operators have to describe the stoppage to the assistant and the assistant has to understand and process what the operator conveys to the assistant. Current methods of issue reporting still leave room for improvement. The system in place at Diversey uses a desktop computer in the middle of the operating line. Chapter 4 dives deeper into the context of the use case for this report.

3.4.1 How will the Cognitive Advisor work?

For the Cognitive Advisor, it is more than just capturing the issue descriptions. It will also capture the root cause analysis (RCA), the solution and the implicit knowledge such as know-hows (section 4.3.1). Furthermore, it will provide recommendations on best practices (section 4.3). Communication with the assistant will be through a headset with a microphone that the operator will be wearing. That way, the necessary information can be collected hands-free by voice and the operator can continue working normally on their tasks.

A schematic of the components needed to communicate by voice with the operator is shown in figure 3.4. A voice interface will capture the voice input from the operator. This voice input is converted to audio input and the speech-to-text (STT) component converts it to text. After which the NLU component dissects the text and compares it to its training data, resulting into intents & entities. The dialog manager uses machine learning to create responses to these intents and entities, which by this point were still user input. The responses are converted from text to audio in the text-to-speech (TTS) component whereafter the voice interface provides a response to the user by voice.

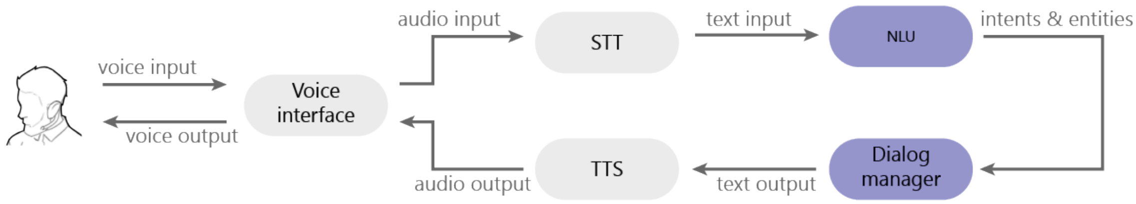


Figure 3.4: A schematic of a cognitive Advisor with voice interface

3.4.2. How will the conversational assistant work?

The conversational assistant however, is text-based. The schematic is shown in figure 3.5. The choice was made to be able to make quicker iterations and improvements on the NLU and dialog managers by removing the need of a voice interface, STT and TTS component combination. Furthermore the variability of the STT system incorrectly registering words is taken out of the system. Next to that, Rasa provides their own open-source API interface named Rasa X, which was utilized for the prototyping throughout the project. The disadvantage of this approach is that text input from a user will differ from voice input and needs to be taken into account during development of the CA.

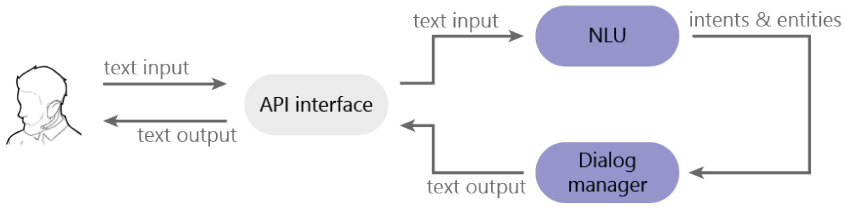


Figure 3.5: A schematic of a conversational assistant using text as input and output

3.5 Operator acceptance

The system in place at Diversey for capturing data on issues is called the ODCE-system. It does not work correctly as intended for the operators as described in section 4.4.6, and operators prefer to ignore the system. The operators are discontent with the system as it brings them annoyance and extra labour. The system is not used correctly because of it. This discontentment should be considered when developing the conversational assistant. “Numerous CAs were discontinued because of inadequate responses to user requests or making errors because of limited functionalities and knowledge of a CA, which can lead to frustration” (Brendel et al., 2020). Therefore, it was important to reduce possible frictions between operator and CA. Accuracy requirements were set in sprint 1 to prevent this.

3.6 Standardized terminology

Operators and AI should collaborate in a coordinated manner and form a heterogenous team (van den Bosch et al., 2019). This heterogenous team is also referred to as a “hybrid team” where multiple agents, either humans or machines, work together interdependently. One important factor in making a hybrid team work is that the agents of a hybrid team have a shared vocabulary which they can develop and refine.

Here I suggest a method of co-learning for both operators and CA. The operators learn to operate with the CA while the CA is able to gather information on issue descriptions. The conversational assistant must share the operator's vocabulary to understand them, but the operators should also move towards terminology that is standardized so that the co-learning process runs more smoothly.

Some of the speech related challenges presented in section 4.4.6 are explored in sprint 3 to move towards standardized terminology.

4. COALA & Diversey

The next chapters are used to give the context of the conversational assistant. Section 4.1 is specifically used for the context of the COALA project, section 4.2 is dedicated to Diversey and its role in COALA, section 4.3 is for the high-level Cognitive Advisor that encompasses the conversational assistant, section 4.4 discusses the stakeholders of this project and 4.5 goes into the value creation for the stakeholders. Section 4.6 goes in depth into the context of the factory.

4.1 COALA project

As mentioned in the introduction, COALA is a H2020 EU project with 17 partners. TU Delft and the COALA partners work together to develop a human-centered voice-enabled Cognitive Advisor that provides support to operators in manufacturing for situations characterized by cognitive load, time pressure and little or zero tolerance for quality incidents. Furthermore, the assistant should aid in reducing the time needed for on-the-job training of workers.

The introduction of COALA into these manufacturing facilities was divided into four key objectives:

- Reduce the number of quality incidents in manufacturing
- Reduce the time needed for on-the-job training of workers in manufacturing
- Overcome barriers and reduce scepticism regarding the use of a voice-enabled Cognitive Advisor in manufacturing environments
- Improve the competencies of blue-collar workers in managing AI opportunities, challenges, and risks in the shop floor

These high level objectives are not direct objectives to this project, but elements are applicable here. Through the use of the CA, for instance, workers competencies with AI will improve simply through having experience with the use of such systems. With the training data the COALA Cognitive Advisor will be able to provide the necessary support.

4.2 Diversey

Diversey BV is a global producer of professional hygiene products. It's a company that makes cleaning, sanitation and maintenance products. In 2016, the global market for food safety and professional hygiene was valued 36 billion euros of which 8 billion is approximated to be the total EU market. The global market is predicted to increase at a compound annual growth rate of 4% from 2019 to 2024. Diversey's market share is estimated to be 12% of the total market (COALA, personal communication, 2021).

Diversey applies agile manufacturing in most of its facilities in Europe. Due to agile manufacturing, Diversey is able to provide on-demand supply of a variety of items to its customers, as well as make them in relatively small batches (5-10 tons). However, this method requires change-overs of the production line to switch between different products on the line. Furthermore, due to the sheer complexity of the line, many stoppages take place. The change-overs in combination with the downtime due to stoppages takes currently up 40-45% of the total production time on average (COALA, personal communication, 2021).

With COALA, Diversey aims to tackle the change-over time and downtime, but also reduce workers' cognitive effort in handling unanticipated complex production line tasks.

4.3 Cognitive Advisor

The COALA Cognitive Advisor will be integrated into the IT infrastructure of several companies in order to be applied in the production facilities for problem identification, root cause analysis (RCA) and solution delivery, figure 4.1 provides an overview. It will utilize posture tracking for contextual awareness and voice communication for interacting with operators in the facilities.

The posture tracking system makes use of a stereo camera and will be utilized for ethical location tracking of operators (Surendranadha Panicker, 2021). Plans are also made to include sensor data and machine data of production lines as input for the Cognitive Advisor to make it even more context aware. The COALA team will focus on four core components of the Cognitive Advisor shown in table 4.1. This thesis falls into the in yellow highlighted component; the knowledge acquisition. This component is for gathering explicit and tacit knowledge on the operating line through dialogs and feed the knowledge into the other components.

Cognitive advisor components	Purpose
Knowledge acquisition	The knowledge acquisition component is targeted at gathering explicit and tacit knowledge on the operating line through dialogues with the operators.
Knowledge representation	The knowledge representation component carries out two tasks: Validate the captured knowledge by the knowledge acquisition component and provide support in decision-making processes in manufacturing (i.e. through means of a graphical presentation of the knowledge).
User profile	The user profile component will support the recommender engine with the required user data to adapt the recommendations to the specific user.
Recommender engine	The recommender engine component of the cognitive advisor will provide recommendations on best practices to the operators.

Table 4.1: The four components of the cognitive advisor and its purpose

4.3.1 Knowledge acquisition

Operating a manufacturing line at target speeds while also resolving stoppages in a timely manner is a cognitively demanding activity. The goal of the knowledge acquisition component is to elicit this knowledge through dialogues with the operator while minimising the impact on their work and cognitive load. Downtime and production target figures reveal significant differences between different operator shifts. Existing documentation of the machines is limited, outdated and slow to access as it is paper based. Therefore, operators rely heavily on their own explicit and tacit knowledge.

The knowledge acquisition schematic is shown in figure 4.1. Non-verbal data such as the posture tracking and sensor data can be used as input for the dialogs and will influence the conversation between the Cognitive Advisor and operators. The verbal data is split into *Issue descriptions, Root cause analysis* and *Solutions*. The issue descriptions are for collecting the observable issues on the production line, root cause analysis is for capturing the root of the cause and with solutions capture the fix to the issue is collected. These elements should feed into the tacit knowledge of know-hows for the specific production line.

As mentioned in the previous section, the knowledge acquisition component is targeted at collecting both explicit and tacit knowledge. However, for this thesis, the focus is only on explicit knowledge; more specifically explicit knowledge of issue descriptions. The issue descriptions provide a good starting point for creating a conversational assistant from scratch. Adding tacit knowledge such as intuition and know-hows increase complexity and is currently not desired. Furthermore, providing explicit knowledge on issue reporting is a direct benefit for Diversey as it emphasizes improving data quality.

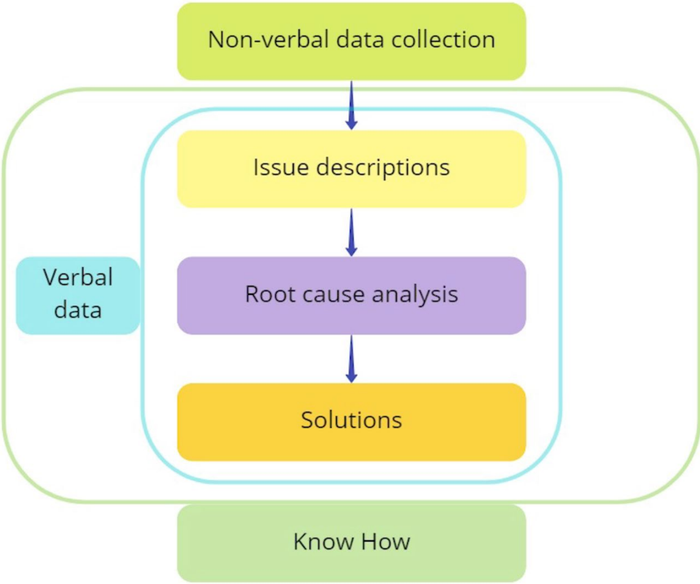


Figure 4.1: An overview of what knowledge will be collected with the Cognitive Advisor

Figure 4.2 provides an overview of all the data that can be captured at the 5/10L production line of Diversey.

Data from	Types	Description
Conversation	Issue descriptions	Description of the issue
	Machine location	The high level machine location of the issue
	Product component	The product component related to the issue
	Product component state	The state of the product component related to the issue
	Machine component	The component of the machine related to the issue
	Machine component state	The state of the component of the machine related to the issue
	Solutions	Solution for the issue
ZED 2 camera	Operator position	The operator position is tracked on the production line
	Stoppage location	Location of the stoppage
Production line sensors	Canister sensor	Checks whether canisters are on the assembly line
	Label sensor	Controls whether the label is correctly placed onto the canister
	Cap sensor	Controls whether the cap is correctly placed onto the canister
	Box label sensor	Controls whether the label is correctly placed onto the box
	Position sensor	Controls whether the canister is correctly placed onto the pallet
Production line machines	Speed production line	The user uses spoken language to interact with the assistant
	Speed filler	The user uses written language to interact with the assistant
	Pressure filler	The user uses buttons to interact with the assistant
	Weight canisters	The user uses a combination of spoken language, written language and buttons to interact with the assistant

Other	SKU	Stock keeping unit, unique number to identify a certain product
	Batch	Unique number for the batch of the product
	Date	Date of the issue
	Time	Time of the issue

Figure 4.2: Table of data types with where they can be captured from and its description

4.4 Stakeholders CA

The relevant stakeholders for this project have been mapped out in a stakeholder map in figure 4.3. The arrows between parties indicate a form of influence on the other party, (e.g. business management expecting a certain batch size per hour for the operators to produce).

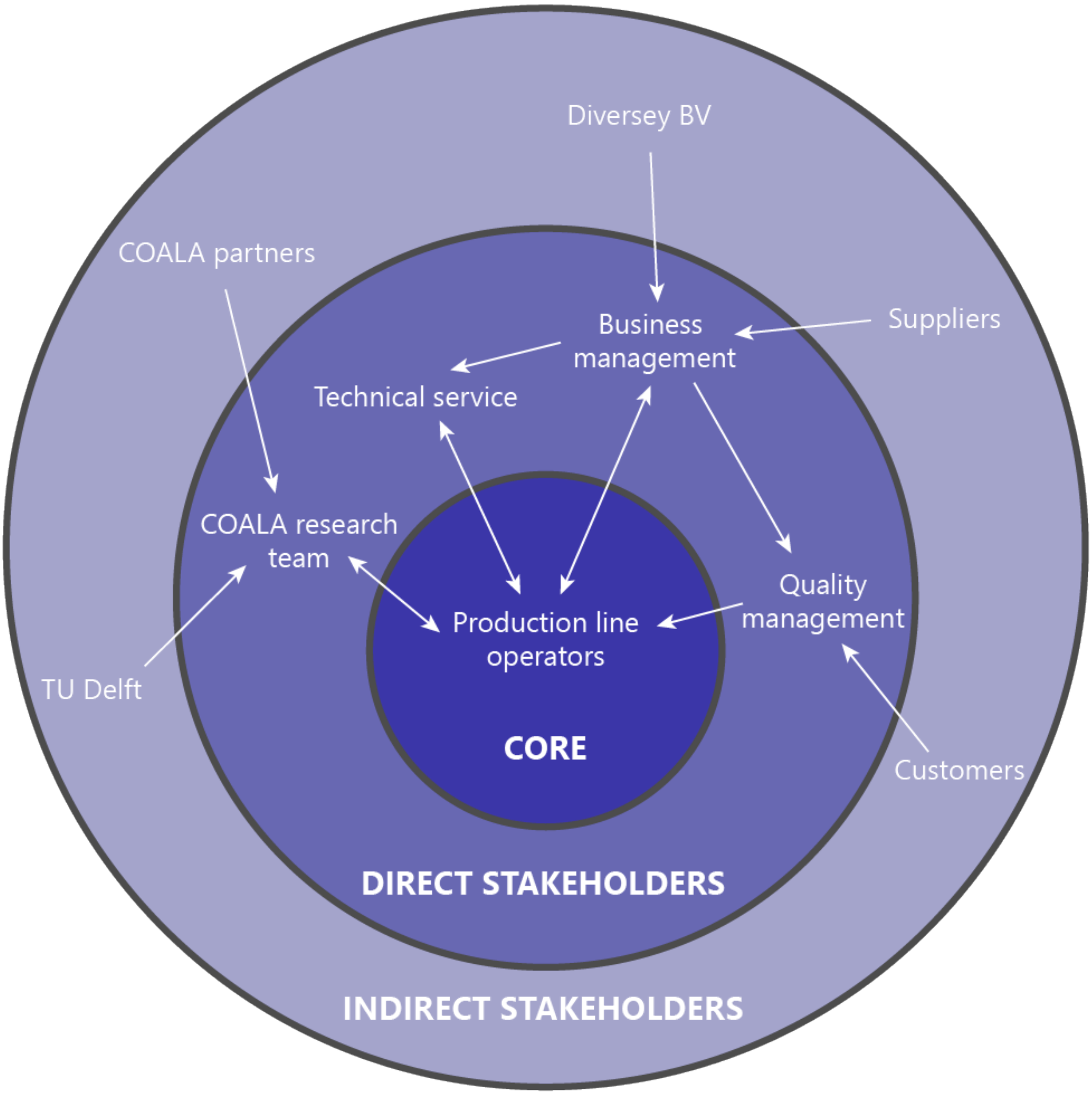


Figure 4.3: Stakeholders divided into core, direct and indirect stakeholders. Arrows signify influence on another party.

At the core are the production line operators who are the end-users of the CA. The direct stakeholders are the parties that are taken into account for the development of the CA. Both the core stakeholders and the direct stakeholders are the beneficiaries of a working conversational AI system on the production line. For instance, an improvement in data quality would directly benefit all parties in the core and of the direct stakeholders. Knowledge can be transferred from expert operator to a novice operator through knowledge acquired with the CA, while technical services and quality management can use the data to pinpoint what the issue is with a certain machine or the quality of a certain product respectively.

The indirect stakeholders have an influence on the direct stakeholders. To give an example, the customers expect a certain quality of the product coming from Diversey and quality management has to assure this level of quality is reached. These influences from the indirect stakeholders impact the behaviours of the direct stakeholders. However, for this work the indirect stakeholders are left as out-of-scope for the development of the CA.

Depending on the stakeholder's group's availability, they either had an advisory role, were interviewees or user participants for data collection or testing of the prototype. The COALA research team and Business management of Diversey had an advisory role, input from quality management was gathered through interviews and user tests, operators were interviewees and participated with the data collection and technical service took part in user tests.

4.5 Value creation for stakeholders

The reason to implement a conversational assistant is to create value for the stakeholders involved. For this section, the Diversey parties, aside from the production line operators, are generalized into one stakeholder; technical service, quality staff and business management are generalized into: Diversey. Similarly, COALA research team, TU Delft and COALA partners are generalized into: COALA. The benefits and disadvantages of the conversational assistant to the different stakeholders is provided in table 4.2.

By using a voice user interface, the operators are able to capture issues hands-free and are able to continue working on the production line. The introduction of the conversational assistant will however increase the cognitive load for the operators as it is expected more issues will be captured. This increases the chance that the operators have to handle the issue and report on it simultaneously.

Section 4.4.6 describes the issues with the issue reporting. There is potential here to be gained with introducing a conversational assistant into the operations for Diversey. For one, more data can be reported on issues for all levels of severity. Secondly, the quality of the data can be improved. By creating a structure of issue description with the conversational assistant where the output is structured, the operators are guided to correctly capture the issue descriptions.

Noteworthy here, is that with the introduction of the knowledge acquisition component, the cognitive load on the operators is expected to increase. The operators have to learn how to communicate with the assistant next to the increase of the amount of issue reporting. However, the introduction of the recommender system is expected to reduce the cognitive load as some cognition will be done by the assistant.

One use for the data provided to Diversey is for process improvement. Patterns in issues could provide the necessary information to, for instance, find what machine should be repaired or replaced.

Stakeholder(s)	Benefits	Possible disadvantages
Operators	Hands-free issue reporting through voice-enabled conversations	Increase of cognitive load as downtime is reduced and more issues are captured
	Operators can work continuously on the operating line	Increase of cognitive load due to simultaneous issue handling and reporting
Diversey	Reduction of downtime	Operator burnout
	More data reporting on issue	
	Better quality of data reported on issues	
COALA	Findings and improvements can be used for implementation into the Cognitive Advisor	-
	Issue descriptions can be used as training data to relearn the model and improve	Added complexity and training data can reduce accuracy (can also improve)

Table 4.2: The benefits and possible disadvantages of introduction of a conversational assistant for different stakeholders.

4.4 Diversey Enschede Factory

The use case of this thesis is the Diversey factory in Enschede in the Netherlands. Diversey allowed, although Covid restrictions made it more difficult, for the COALA team to visit the factory. During the visits, we were able to check the shop floor, interview business management, quality management and interview and observe operators on the production line.

At the factory in Enschede, the full process from raw material to finished goods is done. This includes processes such as mixing the detergents and other liquids to fully packaging the product on operating lines. Several operating lines can run simultaneously for different kinds of packaging. This project focuses on one specific operating line, the 5/10 litre operating line.

A context analysis was conducted to understand the environment the operators are working in. Methods applied during the context analysis were observations of the operating line and the operators working on it and unstructured interviews with operators and two employees of quality control department.

4.4.1 5/10 litre operating line

The 5/10 litre operating line (5/10L line) (figure 4.4) is considered a priority line that must always be running.

Figure 4.4: The 5/10L operating line with on the left the filler, in the middle the box unfolder and box turner, in the foreground to the right of these the box printer and in between the two machines in the back the box packer.

On this operating line agile manufacturing is applied, a method of producing several different products on the same operating line. Depending on the client's needs, the operating line allows for different labels per product, different canisters, different caps, and several different liquids can be pumped into the filler to fill the canisters. As the name of the operating line indicates, there are also two sizes of packaging that are produced on the line, 5 and 10 litre canisters. Switching between products can create substantial downtime where no products are created, but a change-over is taking place. These reconfigurations can take approximately 20-25 minutes for simple changeovers and up to 2 hours for complex changeovers (COALA, personal communication, 2021). This can include high pressure cleaning of the filler, cleaning the piping system that delivers the product to the filler, swapping the labels needed or even changing the entire line to fit another canister size, (e.g. swapping from 5 litre canisters to 10 litre canisters or vice versa).

During the interviews, an operator mentioned that during one single shift up to 80 stoppages took place. Depending on the severity of the issue, the duration of solving stoppages has a wide range of taking just a few seconds up to taking several hours to solve. This downtime in combination with the change-overs currently takes up 40-45% of the total production time on average (COALA, personal communication, 2021).

4.4.1.1 Layout 5/10L line

To be able to create a finished product of a correctly labelled, filled, and closed canister with a cap which is boxed, machines are needed. For the entire process from empty canister to a plastic wrapped pallet full with boxed canisters 15 large machines are needed, this is excluding the conveyor belts, buffer areas and sensors.

Figure 4.5 shows a simplified top view of the line and figure 4.6 gives a schematic overview of the canister journey and the cardboard package journey and where both journeys connect at the box packer and continue as one.

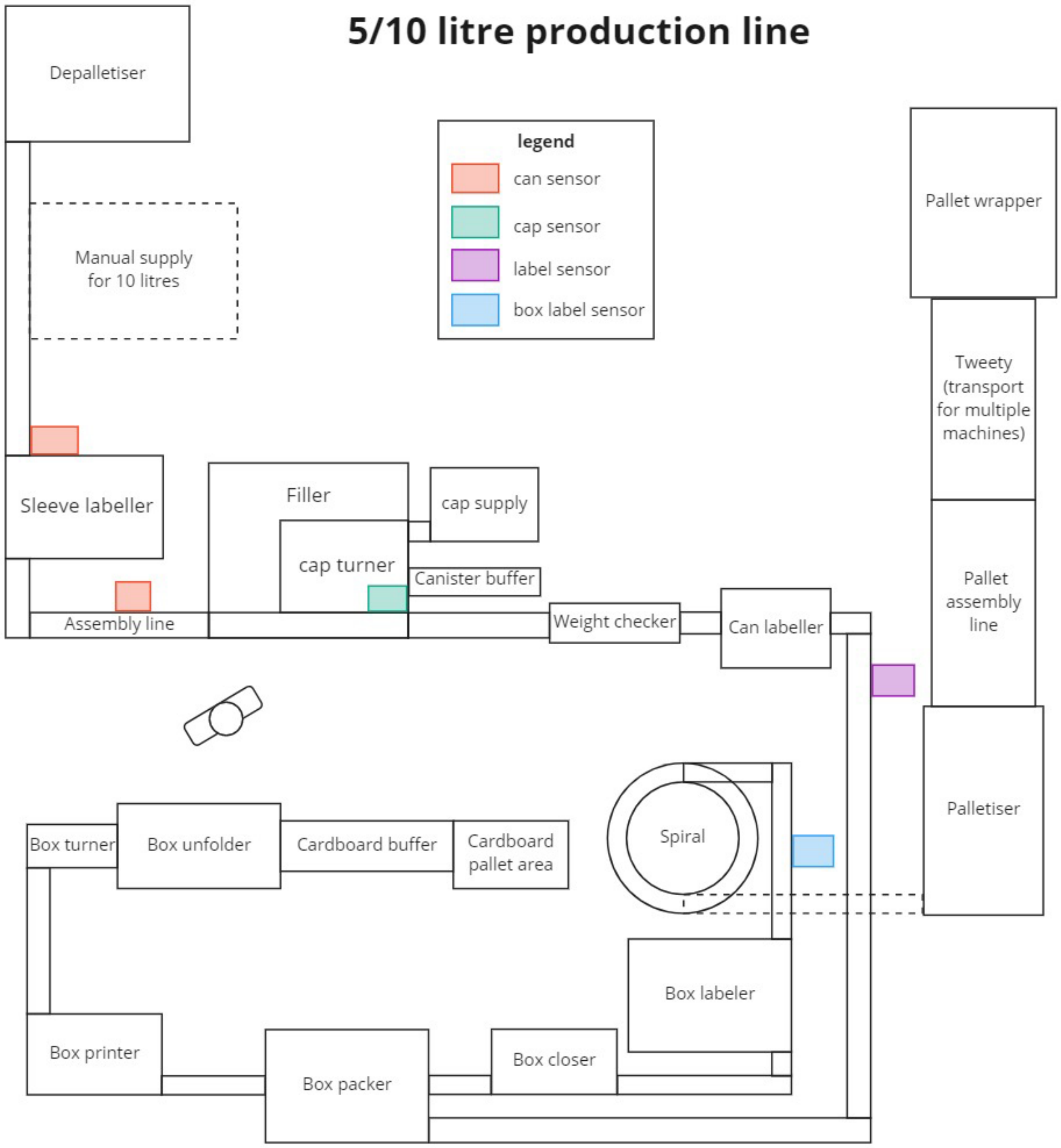


Figure 4.5: Simplified top view of the 5/10 litre operating line

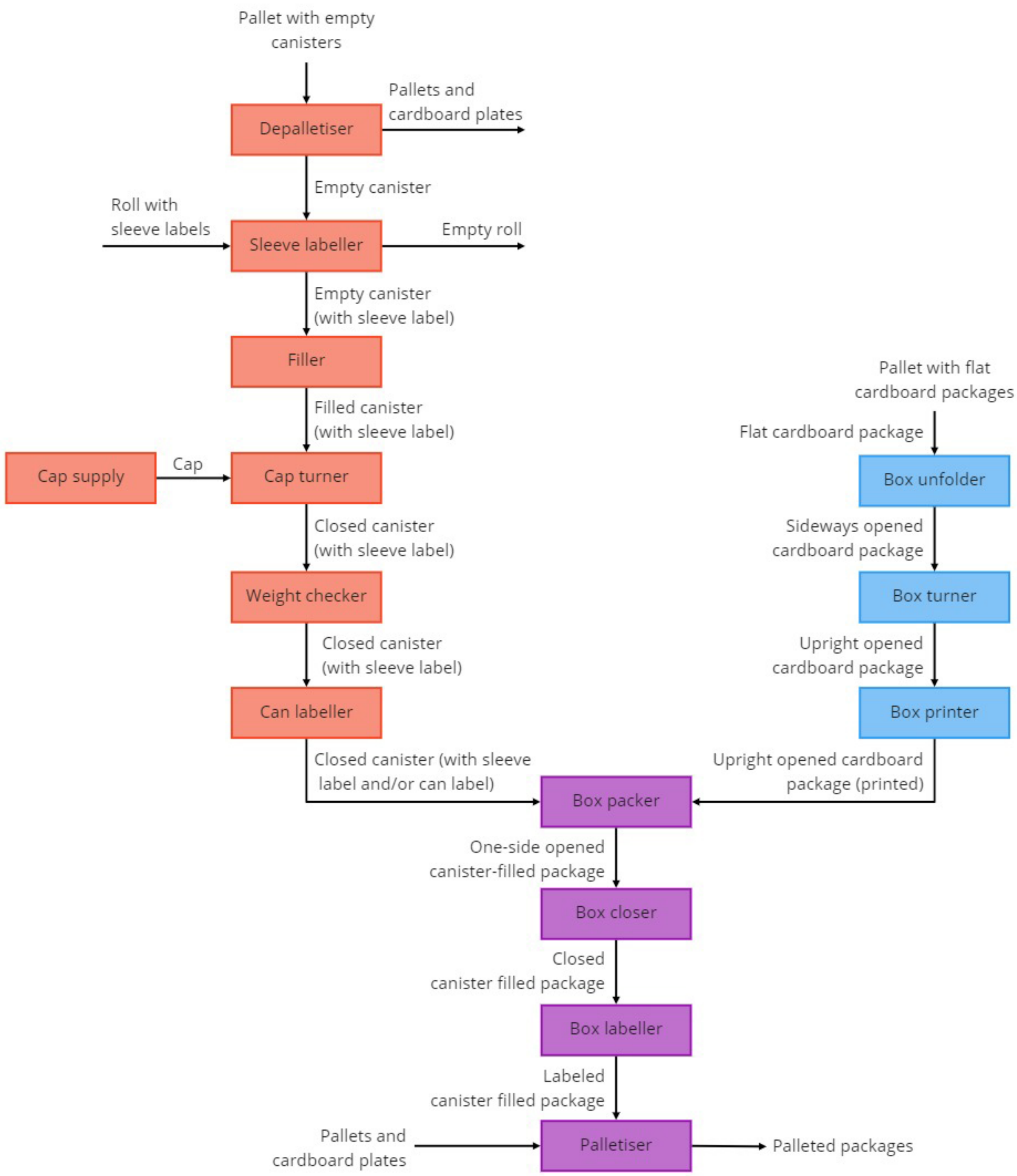


Figure 4.6: Product journey on 5/10 litre production line. In red the canister journey, in blue the cardboard package journey and in purple the two combined

The journey of the canisters starts with the supply of the canisters at the depalletiser, where rows of 6 canisters are taken off the pallet and placed onto the assembly line. The canisters are transported towards the sleeve labeller. Just in front of the sleeve labeller, a canister sensor checks whether canisters are coming from the depalletiser and whether these are upright. If not, the line is stopped. At the sleeve labeller a sleeve label can be placed on the can, depending on the product (liquid) and client's desire whether the canister is required to have a sleeve label. If not, the sleeve labeller acts as part of the assembly line. Afterwards the canister is transported towards the filler where the canister can be filled with the product. Again, a canister sensor is placed just in front of the filler to check whether canisters are on the assembly line and if these are upright. At the filler several products can be filled in the canister such as detergent and alcohol. Within the filler, the cap turner places a cap on the filled canister. The caps are supplied from the cap supply. A sensor in the cap turner checks whether caps are placed incorrectly. Whenever this happens, the canister is pushed onto the buffer lane, otherwise the canisters continue towards the weight checker where the canisters are checked whether their weight falls in the predefined range set by the operator. If not, the line is stopped and the operator is tasked to manually check the weight of the can. If the weight of the can is incorrect after manual check, the operator has to examine whether the filler is filling the canisters incorrectly and adjust accordingly. The next station after the weight checker is the canister labeller. Here a label can be placed on the canister. Similar to the sleeve label, not every canister has a label placed. The label sensor controls whether the label of the canister labeller is placed correctly. If everything is alright, the canister is transported further towards the box packer where the canister is placed inside a box.

The journey of the cardboard package starts at the area for pallets with flat cardboard packages. The flat carton is manually placed by the operator onto the buffer for the box unfolder. The flat cardboard packages are opened on one side in the box unfolder. The box turner is there to set the boxes upright. The upright boxes with the opening aiming upwards can be printed if needed at the box printer. Afterwards they are transported to the box packer where the cardboard packages are filled with the finished canisters. One box is used for two 5 litre canisters and for one 10 litre canister.

After the box packer, the packages move onto the box closer where the one open side is closed. At the box labeller two labels are placed onto the side of the box. The spiral is used to transport the boxes upwards towards the palletiser where they are to be ordered and placed onto pallets. The Tweety machine is a mobile assembly line utilized for several operating lines. For the 5/10 litre operating line, Tweety transports the full pallets towards the pallet wrapper where the pallets are wrapped. At this point the full pallets are ready to be shipped to the client.

4.4.2 Operators

The operators (figure 4.7) working on the 5/10L line have 8-hour shifts where 2 or 3 operators work at the same time. The line works 24/7 and the day is divided into 3 shifts. Most often, two operators work on the 5/10L line, sometimes only one and sometimes three. In the case of two operators, usually an experienced operator is paired with a novice. There are many different experience levels amongst operators, some experienced operators with up to 20 years of experience and novices who are just starting. These differences in shifts and different experience levels amongst operators influence the shift's production capacity and could also influence how the CA will have to operate.

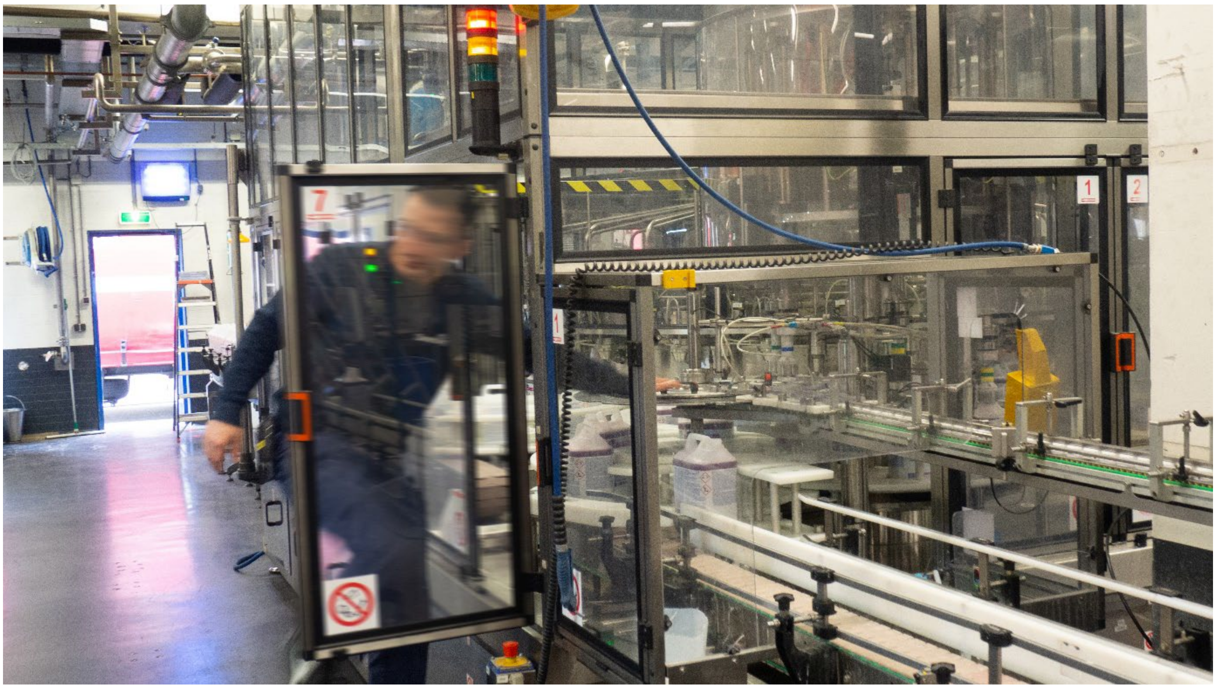


Figure 4.7: An operator in the process of fixing a stoppage at the filler/cap turner.

As shown in the previous section (section 4.4.1), there are several machines running and as will be shown in the next sections (sections 4.4.3 and 4.4.4), many factors can influence whether the production runs smoothly or whether issues arise and the line stops. It is up to the operator to keep the operating line running smoothly and reduce downtime. There are several regular tasks the operators have. The most prevalent ones from the visit are listed below:

- Solve stoppages at the operating line
- Adjust speed of machines and assembly line
- Supplying the sleeve-, canister- and box labeller with enough labels
- Supply the cap supply with enough caps
- Prepare change-overs
- Manual placement of 10 litre canisters on assembly line
- Final check of pallet and adding sticker before sending it off to the pallet wrapper
- Capturing issues with ODCE-system
- End of shift summary on computer
- Report complex issues with ALIS-system

Some of the qualifications observed which would be assigned to an “expert” operator are:

- The quick responsiveness to issues and fast recognition of the issue
- Assurance that paperwork is done correctly and checked doubly.
- Preparation of change-over while production line is running
- Knowing how to conduct the change-over in a timely manner without mistakes

4.4.3 Change-overs

As change-overs are set as out of scope for this project, they will not be discussed in detail, however for the context, it is good to be aware of the practices during change-overs and the operator's task within change-overs.

For change-overs, operators have to prepare the operating line for the next product. Depending on the next product, tasks like starting a cleaning process for the filler and setting the correct range for the weight checker are to be done. Some of the tasks during change-over are listed below:

- Prepare the operating line for next product with paperwork (check batch and stock keeping unit (SKU))
- Set correct range of weight of next product for the weight checker
- Set speed and pressure of filling at the filler
- Start cleaning process for the filler
- Start cleaning process of pipe system
- Control Paperwork for new batch (SKU, amount etc)
- Prepare new labels
- Prepare new canisters if needed
- Prepare new caps if needed
- 4-eye check of paperwork (is everything correct?)

4.4.4 Issues and issue handling

Operators regularly face many issues in one shift which need to be resolved timely to reduce downtime of the production line. Responding to the stoppages and figuring out where it originated on a high-paced production line while doing this as quick as possible can be taxing on the operators. Optimizing the operators' responses to stoppages is in the benefit of the company. To take over some of the cognitive load while simultaneously increasing the response time to stoppages would be beneficial for both company and operator.

One of the operators has mentioned during an interview that an operator experienced up to 80 stoppages in one single shift. Stoppages can take a few seconds, a few minutes and sometimes even several hours. The operators use sensory cues to recognize a stoppage. The line is constantly producing sound and some of the experienced operators can recognize by sound whenever something is jamming and find the issue of the stoppage. Some can recognize sound is not being produced in a particular area of the operating line and pinpoint where the stoppage is. Also, visual cues are used. An operator can see where the line is not running and use these indications to find the cause of the issue. Also, lights are used to indicate whether a machine is running (green), a machine has stopped (orange) and whether the machine has a major issue (red).

There are many variables that influence issues. All the machines can have several causes for an issue to happen, the components of the final product can also have several flaws for an issue to arise and an operator could have set a certain setting (e.g. filling speed of the filler) wrong for a specific product. Capturing all issues and causes can be a difficult task. However, some common issues have been captured during observations and from interviews, table 4.3 gives an overview of the captured common issues. In the table, the consequence, severity and solution of the issue and the machine related to the issue are given. Table 4.4 gives an overview of the severity levels of the issues observed. The system for issue capturing (ODCE-system) used at the factory can recognize a stoppage (more on this in section 4.4.6). Whenever the stoppage takes more than 3 minutes, the system pops up on the desktop computer at the operating line and asks the operators to fill in the issue. Therefore, the severity of "high" has been set to 3 minutes of stoppage or higher.

An example of a medium to high range issue which have been observed is of issues related to the weight checker. In different shifts there were several instances observed for the issue of the weight checker where the severity was ranging from medium to high. In the first shift, the experienced operator was next to the weight checker when the line stopped. The operator could directly respond and solve the issue. However, in a different shift a similar situation happened where a canister had the "wrong" weight, but the operators were puzzled as the weight of the canister was correct and fit in the range set by the operators. They had to reset the weight checker and adjust some settings to make it run correctly, this process took over 3 minutes to solve.

Common issues	Consequence(s)	Severity*	Observed solution(s)	Related machine(s)
Canisters wrongly placed on production line	Canister is not registered by the can sensor in front of the sleeve labeller and the line stops.	Low to medium	Set canister upright before passing the canister sensor.	Depalletiser
Sleeve incorrectly placed onto the canister	The sleeves on the canisters either are not placed on the canisters at all, are wrinkled or torn apart. The machine can also stop.	Medium	Stop the sleeve labeller, enter machine, tear off faulty sleeves and correctly connect sleeve roll with the sleeve grippers.	Sleeve labeller
Foaming of product in the canisters	The weight of the canisters can be too low, the product can flow out of the canister into the filler. The assembly line can become slippery.	Medium to high	Adjust filling speed and pressure.	Filler
Cap incorrectly placed onto the canister	The canister is pushed onto the canister buffer lane.	Low	Manually remove the cap from the canister in the buffer lane and manually place cap correctly onto the canister.	Cap turner
Weight of the canister not in weight range	The weight checker stops the production line.	Medium to high	Set the canister aside and check manually the weight of the can. Adjust filler accordingly.	Weight checker
Canister label incorrectly placed onto the canister	The line stops as the canister label sensor recognizes an incorrectly placed label.	Medium	Remove canisters with wrongly placed label, re-adjust the canister labeller so that labels are placed correctly.	Canister labeller
Box label incorrectly placed onto the cardboard package	The line stops as the box label sensor recognizes an incorrectly placed label.	Medium	Remove boxes with wrongly placed labels, re-adjust the box labeller.	Box labeller
Cardboard pallet plates bended (concave)	The cardboard pallet plates are not grabbed by the palletiser pallet grippers and not placed onto the pallet.	Medium	Flip the pallet plates pile so that the plates are convex and the gripper can grip onto the plates.	Palletiser
Pallet with boxes dropped	The boxes fell onto sensors at the pallet transport and the direction the sensor were targeting was incorrect.	High	Stop the palletiser, manually remove pallet and the boxes with canisters and manually adjut the direction the sensors are pointing.	Palletiser

Table 4.3: Common issues captured during observations and interviews, with consequence, severity and solution of the issue and the related machine(s).

*** = see table 4.4 for explanation of the severity.**

Severity	Explanation
Low	The line does not have to stop
Low to medium	Depending on the operator's response, the line either does not have to stop or there is a stoppage less than 3 minutes
Medium	The line has a stoppage less than 3 minutes
Medium to high	Depending on the operator's response, the line has a stoppage less than 3 minutes or more than 3 minutes
High	The line has a stoppage more than 3 minutes

Table 4.4: Overview of severity levels

4.4.5 Communications and issue reporting

Several tools are used for communication and issue reporting by the operators: ALIS, ODCE, Word and Whatsapp. For each tool there are clear benefits but also disadvantages. Table 4.5 gives an overview.

Tool	Explanation	Benefits	Disadvantages	Noteworthy observations
ALIS	ALIS is a factory wide system used for reporting major issues and a way of communication between teams at Diversey, (e.g., between business management and operators, technical service and operators or quality staff and business management).	ALIS allows for factory wide communication between teams. A message between operator and technical service can also be accessed by quality staff for instance.	Communications are not always processed.	Some operators claim that ALIS works perfectly fine, while quality management indicated that sometimes their communications in ALIS are not processed and have to communicate the matter in person.
ODCE	ODCE is used to report issues on the production line. This is used for all operating packaging lines at the Diversey factory. Notably, it is used the least on the 5/10 liter line and in addition it is also used incorrectly producing incomplete data.	Capturing data on issues gives other teams and shifts a better insight in what goes well and what wrong. Furthermore it allows for process improvement.	Data quality is lacking. A lot of missing information. Often ignored by operators even when the warning for a 3+ min stoppage pops up on the screen.	Two stoppages of 3+ minutes happened in same shift, because first ODCE pop-up was ignored, the second stoppage was captured for the first issue. Result: ODCE data filled in for wrong issue (2nd for 1st). Second issue not reported (skipped by operator)
Windows Word	Communicating the shift and issues between different shifts is done quick and dirty through the Windows Word application on the desktop computer of the 5/10L line. Short summaries of the shift are written at the end of the shift with some small mentions of issues.	Allows for quick and dirty summary of the shift.	The summaries are not processed company wide, only between shifts. A report such as: "shift went well" do not provide a lot of information for the new shift.	Some interviewed operators mentioned they preferred using Word over ODCE as an issue reporting tool.

Whatsapp	Whatsapp is used to communicate quickly with the mixing staff, mostly for change-overs to check whether the next mixture of the liquid is ready to be used.	Allows for quick communication.	Communication not processed company wide, only between two persons (or a group if a group chat is used)	Not all operators use Whatsapp, only a handful.
Paperwork	Paper sheets are used for product control on the operating line. The batch and stock keeping unit (SKU) numbers are checked during change-over.	The paperwork allows for quick 4-eye checks with 2 operators. 4-eye checks are used to reduce operator mistakes at the line.	Process is relatively slow and increases downtime during change-over. Furthermore, paperwork needs to be physically transported.	Currently, operators prefer it over digital systems.

Table 4.5: Overview of issue reporting tools with explanation of the tool, benefits, disadvantages and note-worthy observations during context analysis

ALIS is mostly used for communicating larger issues between teams. Operators for instance can report on a breakdown of a machine or machine component, while quality management could communicate on contaminations or quality of a product. ODCE is used or should be used for reporting of issues on the operating line. Here the choice is with the operators whether they report an issue or not. Only after a stoppage of 3 minutes or higher a message pops up on the desktop screen at the operating line and the operators have to report the issue with ODCE. However, during observations this pop-up was ignored often.

4.4.6 Issues with issue reporting

The operators prioritize issue handling, while often issue description is done poorly or ignored entirely. Current methods used at the Diversey shop floor for issue description leave much room for improvement. Evaluations with operators showed that the system is an inconvenience and a time sink. Simpler and quicker tools such as Word are used to communicate in between shifts. The data that was captured with the ODCE system was often ambiguous, incomplete and non-descriptive. The operator’s main task is to maintain the line and keep it running, so the operators prefer not to waste time and energy on issue reporting. However, such data can be of significant value to the company as it can be used for process improvement of the production line, maintenance can make use of the data to identify and locate the root of the cause of a major stoppage and the knowledge that is acquired can be carried over to other operators.

5. Rasa

During the entire process, what is possible with Rasa was analysed and is provided in this chapter. Section 5.1 provides the Rasa terminology, section 5.2 goes into how Rasa works, chapters 5.3 and 5.4 go in-depth into the NLU and dialog manager respectively, of a natural language processing system. Sections 5.5, 5.6 and 5.7 discuss elements that can be implemented into the model. These are the intents & entities, CA actions and stories & rules respectively. Chapter 5.8 discusses conversation driven development.

5.1 Rasa terminology

Table 5.1 provides the terminology used within the Rasa framework which will also be used in throughout the report.

Terminology	Definition
Domain	The domain defines the universe in which your assistant operates. It specifies the intents, entities, slots, responses, forms, and actions your bot should know about
Intents	Something the user tries to convey or accomplish with their input (e.g. greeting, mentioning issue)
Entities	Keywords that can be extracted from user input (notation for entities: [lid]{"entity" = "p_comp"}, lid = product component entity)
Slots	A slot is used to store information the user provided as well as other information gathered (e.g. results of a database query)
Forms	A type of custom action that asks the user for multiple pieces of information. Forms are used to set slots.
Utterance	Responses are messages that the assistant can send to the user. Utterances are also used in this document to refer to responses.
Actions	After each user input, the model will predict an action that the assistant should perform next. Respones are actions. Custom actions are actions that can run custom code (e.g. validate the user input)
Rules	Rules describe short pieces of conversations that should always follow the same path.
Stories	Stories can be used to train models that are able to generalize to unseen conversation paths (unhappy paths).
Happy path	Happy path is when the user behaves as expected.
Unhappy path	Unhappy paths are paths when the user diverges from the happy path. Stories can be used to train the model to handle unhappy paths.

Table 5.1: Terminology used within Rasa framework

5.2 How does Rasa work?

Rasa is an open-source machine learning framework to create conversational assistants that automate text and voice-based conversations with users. In these conversations, an assistant has, simply put, two tasks: Understanding the user and providing correct responses. Rasa takes care of these tasks with the natural language understanding (NLU) and the dialog manager.

To create such an assistant, Rasa has provided 3 tools for development (table 5.2):

Tools for development	Description
Rasa Open Source	Rasa Open Source is the core tool for developing a CA
Rasa SDK	Rasa SDK is the "software development kit" provided by Rasa for writing custom actions for the CA which are not included in Rasa Open Source
Rasa X	Rasa X is an API tool for conversation-driven development (CDD)

Table 5.2: Tools for development of a CA provided by Rasa

Rasa Open Source is the core tool for developing a CA. Most of the development of a Rasa CA can be done in the YAML language, the files of a default Rasa Open Source model are shown in table 5.3:

File	Function
config.yml	The pipeline and policies are defined in the config file
domain.yml	The domain defines the universe in which your assistant operates. It specifies the intents, entities, slots, responses, forms, and actions your bot should know about
nlu.yml	In the nlu-file the training data is provided for intents, entities, synonyms and lookups tables
rule.yml	This file is used for defining rules; rules describe short pieces of conversations that should always follow the same path

stories.yml	This file is used for defining rules; stories can be used to train models that are able to generalize to unseen conversation paths (unhappy paths)
actions.py	This file is used for writing custom python code for custom actions the bot can do
credentials.yml	With this document one can connect the model to websites or applications
endpoints.yml	This file is used to define the different endpoints the bot can use (namely used for actions server)

Table 5.3: Files in default Rasa model

For custom actions however, the Rasa SDK has to be installed and python is used to code. The "actions.py" file is the base file used for adding custom actions to the CA but can be extended with other code.

Rasa X is an API tool in which conversations are stored and can be analysed and used for conversation-driven development (CDD). CDD is the process of capturing conversations with real users and implementing their feedback in the development of the CA. Figure 5.1 shows a conversation in Rasa X.

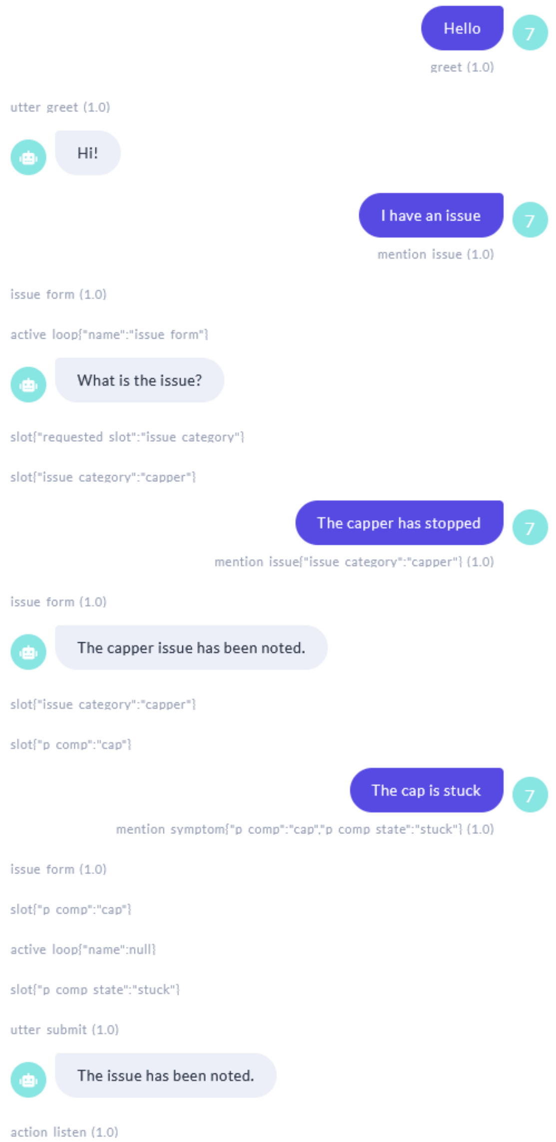


Figure 5.1: An example conversation within Rasa X interactive learning.

5.3 NLU

Training data is provided to the NLU, such that it can recognize and assign (with a certain confidence) user input to intents and entities. An intent is something the user tries to convey in their message. Entities are keywords that can be extracted from user's input. More on this in sections 5.3.3 and 5.5. The NLU of a Rasa model can be configured in a pipeline. The pipeline allows the developer to configure how the model processes input text. Most NLU pipelines consist of a *tokenizer, a featurizer* and *intent*- and *entity classifier(s)* (figure 5.2). Within the pipeline, the developer can tweak the components, for instance, an intent & entity classifier can be tuned so that it only classifies intents. Another classifier can then be used for entity recognition.

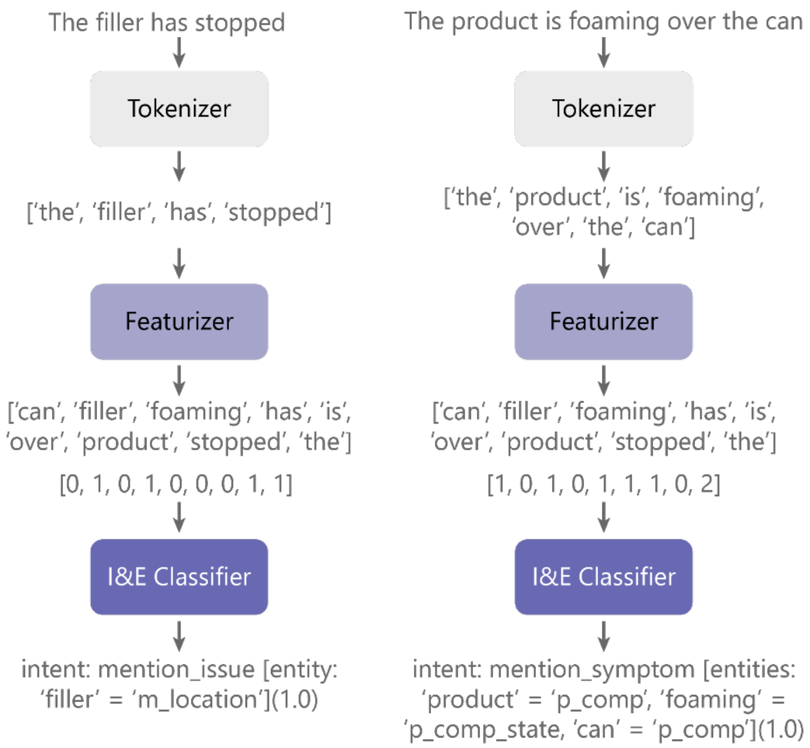


Figure 5.2: A tokenizer/featurizer/I&E Classifier system for two different user inputs.

5.3.1 Tokenizer

A tokenizer breaks up a sequence of strings into pieces called tokens (Techopedia, 2021). A tokenizer commonly used by Rasa is the Whitespace Tokenizer. This tokenizer breaks up a user sentence into words whenever a whitespace is recognized. Tokens can be words, phrases or even entire sentences.

5.3.2 Featurizer

A featurizer transforms the tokens as well as some of their properties into features or word embeddings that can be used by machine learning algorithms (Ricadat, 2018). Through featurization words are transformed into vectors where words with similar meaning create similar vector representations.

> Word embeddings are in fact a class of techniques where individual words are represented as real-valued vectors in a predefined vector space. Each word is mapped to one

vector and the vector values are learned in a way that resembles a neural network (Brownlee, 2019).

5.3.3 Intent & entity classifiers

Intent classifiers assign one of the intents defined in the domain file to incoming user messages. Similarly it assigns entities defined in the training data whenever one is mentioned in a user message.

5.4 Dialog manager

In Rasa, the components of the dialog manager are defined in the config file under policies. These policies allow the model to decide what the next action should be in a conversation. For the default Rasa model three policies are used (table 5.4):

Policy	Function
MemoizationPolicy	Checks whether the conversation matches any stories in the training data and predicts the next action if any stories apply
TEDPolicy	Is a complex machine learning policy that tries to predict the next best action
RulePolicy	Takes care of conversations that follow set rules and makes predictions based on these rules if any rule applies

Table 5.4: Default Rasa policies used for prediction next action.

Rules and stories can be used to specify what utterance of the CA should follow after a certain intent or entity has been mentioned by the user. For instance, whenever the user greets the CA, a rule can be set for the CA to respond with a greet. Rules are used for small chunks of conversation that should always follow the same path and are handled by RulePolicy. Stories are used to train a machine learning model to recognize patterns in conversations which have been predefined in the training data in the stories.yml file; stories are handled by the MemoizationPolicy. Stories can also be used to train the machine learning model to handle unhappy paths in the conversation of the user with the CA. Whenever the user interacts with the CA as intended, this is called a happy path. An unhappy path is where the user diverges from the happy path.

The Transformer Embedding Dialogue Policy (TED Policy) is a different beast. When creating predictions, the TED policy utilizes a transformer architecture to determine which dialogue turns to pay attention to and which to disregard. It takes three pieces of information as input at each dialogue turn: the user's message, the previously predicted system action, and any values saved to the assistant's memory as slots. Before being put into the transformer, each of them is featurized and concatenated (White, 2021, August 19).

5.5 Intents & entities

Whenever the conversational assistant recognizes a user's intent, the CA can respond with an action, such as an utterance. For instance, whenever the user greets the CA, most likely the user is expecting a "greet" back, therefore the CA can be programmed such that it utters a greeting back at the user. Here, one such exchange of words is called a single turn interaction. To have such interactions, the CA has to have been trained on training data. Intents are specified in the NLU file (figure 5.3). Where per each intent, several examples of user input should be provided for the Rasa model to be trained on. Below, an example of a greet intent with two example data is provided.

```
- intent: greet
  examples: |
    - hey
    - hello
```
Figure 5.3: An intent "greet" with 2 training examples

Within the examples, entities can be specified. Entities are keywords which can be extracted from user input. Below an example (figure 5.4):

```
- intent: mention_issue
  examples: |
    - There is a stoppage
    - The [filler](m_location) has stopped
    - The location is the [filler](m_location)
    - the [filler](m_location) has a stoppage
```
Figure 5.4: An intent "mention_issue" with training examples and entities for m_location

5.5.1 Entity roles and groups

Entities could be further specified with roles and groups to further specify the training data and with it attempt to improve the NLU model, below are two examples (table 5.5.) with roles and groups for the following sentence: "The cap is not on the canister". In the first a distinction between two product component entities is made through roles, where one is an object and the other the subject, while in the second example the role and entity for both are the same but group is used to make the distinction in the entities:

Training data example	Entities with roles and groups
The [cap]{"entity": "p_comp", "role": "object"} is not on the [canister]{"entity": "p_comp", "role": "subject"}	Cap: Entity = product component, role = object Canister: Entity = product component, role = subject
The [cap]{"entity": "p_comp", "role": "issue", "group": "1"} is not on the [canister]{"entity": "p_comp", "role": "issue", "group": "2"}	Cap: Entity = product component, role = issue, group = 1 Canister: Entity = product component, role = issue, group = 2

Table 5.5: Training data examples with roles and groups.

However, during development Rasa X version 0.39.3 was used and this version had no support yet for roles and groups. Therefore, I opted out of using roles and groups as I believe that the benefit of a clean user interface for testing through Rasa X overpowered the added benefit of roles and groups.

5.6 Actions

A response (or an utterance) is the simplest of actions a CA can respond with. Other actions are forms or custom actions. Forms are used to set predefined slots and follow a strict path for the user. Both forms and slots are specified in the domain file. Forms can be useful to collect specific data. A custom action is a specific action the CA can do that has been coded in Python in the actions.py file. For instance, whenever the user wants to get the current date and time, the CA can have a custom action where it provides this. Other examples of custom actions are to query a database or make an API call. Rasa SDK is a software development kit in python for running custom actions on a Rasa chatbot.

Responses, forms, slots and actions are specified in the domain, alongside the intents and entities.

5.6.1 Forms

With a form pieces of information can be captured in slots and a form can guide the user into filling these slots with questions. It allows for relatively "strict" information to be gathered. The flexibility comes from how good the model can fill the slots from user's input, this is dependent on the NLU training data. In figure 5.5, a form for capturing issues is given. The form is required to set the slot for 'm_location' when a machine location entity is provided. The same approach is sued for product component state and the product component. It will only go into the next required slot when the first one has been set.

```
forms:
  issue_form:
    required_slots:
      m_location:
      - entity: m_location
        type: from_entity
      p_comp_state:
      - entity: p_comp_state
        type: from_entity
      p_comp_z:
      - entity: p_comp_z
        type: from_entity
```

Figure 5.5: A form for capturing issues at Diversey

5.7 Stories and rules

Stories and rules are a part of the training data for the NLU system. Stories and rules are closely related, but there is a slight difference between the two.

5.7.1 Rules

Rules are strict conversation patterns that model should follow. Rules are mostly used for short pieces of conversation that should always take place. For instance, whenever the user greets the CA, with a rule can be set that the CA responds with a greet back (figure 5.6).

```
- rule: Greet anytime the user greets
  steps:
  - intent: greet
  - action: utter_greet
```

Figure 5.6: A rule for greeting the user whenever the user utters a greet

5.7.2 Stories

With stories, one can train the model to recognize patterns in conversations and have responses according to this training data. A story is a representation of a dialog between a user and the CA. Stories can be used to train a ML-model to be able to generalize to unseen dialog paths. An example of the story code is provided in figure 5.7.

```
- story: greet and explain purpose_COALA
  steps:
    - intent: greet
    - action: utter_greet
    - intent: purpose_COALA
    - action: utter_purpose_COALA
```

Figure 5.7: A story for greeting the user and explaining the purpose of COALA

Stories can be used to handle expected unhappy paths. For instance, with the issue form setup in sprint 2 (section 7.4) it could possibly happen that the operator would like to stop the issue form. This is such an unhappy path that can be implemented into the model to reduce friction and prevent annoyance for the operator. In figure 5.8 is the code of such a story to stop the issue form. When the issue form is active and the user provides an intent “stop”, the CA asks whether the user wants to stop. When the user affirms, the issue form is deactivated.

```
- story: issue form stop stop
  steps:
    - intent: mention_issue
    - action: issue_form
    - active_loop: issue_form
    - intent: stop
    - action: utter_ask_stop
    - intent: affirm
    - action: action_deactivate_loop
    - active_loop: null
```

Figure 5.8: A story for stopping the issue form

5.8 Conversation driven development

Rasa (2021, July 12) suggests the best method for building a great NLU model is to use conversation driven-development (CDD). This means letting real user-conversations guide the development of the model. So using real data from real users for training data. Real user messages can be chaotic, full of errors and typos, and far from 'perfect' representations of the predefined intents. However, the model has to be able to predict these messages and have a correct response to them.

Rasa X is an API interface which can also be used for CDD. With Rasa X a developer can review conversations, annotate data, share & test the assistant.

6. SPRINT 1: Conversational data

This sprint focuses on the data collection where the researcher roleplayed the CA and its implications. It tests the riskiest assumption of a CA which provides human-like conversations.

6.1 Riskiest assumption

The riskiest assumption tested in this sprint is highlighted in table 6.1.

Riskiest assumption
By having human-like conversations, it is possible for a CA to extract high quality issue descriptions while ensuring positive user experience.

Table 6.1: Riskiest assumption sprint 1.

Humanlike conversation is something that is often strived for with conversational assistants. Especially with the push towards contextual assistants and adaptative CAs. Here the assumption is made to test whether it is desired for the Diversey use case. To test this assumption, a data collection on issue descriptions was organised where the assistant was roleplayed by me. By roleplaying, the humanlike conversations can be imitated.

6.2 Data collection for issue description

A data collection was conducted at the Enschede Diversey factory. The data collection has an important role in the development of the COALA conversational AI. Not only does this data have a role in training the model for knowledge acquisition, it will also be used to guide the development of the assistant and the design of the dialogs. Furthermore it provides a direction in what should be collected with the CA, what kind of data is needed for process improvement.

6.2.1 Goals data collection

As mentioned in section 4.4, the first visit in April was for context analysis of the Enschede factory, the second visit at the Diversey factory had several goals in mind: (1) a preliminary operator speech pattern analysis, (2) identifying possibilities and restrictions for a conversational assistant, (3) gathering more contextual knowledge and (4) testing of audio quality in a noisy (shop floor) environment. Table 6.2 provides an overview of prioritisation in weight per goal which was applied during the data collection.

Goal	Weight (total 1.0)
1. Preliminary operator speech pattern analysis	0.5
2. Identifying possibilities and restrictions for a conversational assistant	0.3
3. Gathering contextual knowledge	0.1
4. Testing of audio quality in a noisy environment	0.1

Table 6.2: Overview of prioritisation in weight per goal

6.2.1.1 Preliminary operator speech pattern analysis

The goal here is to explore how the operators communicate, specifically how they convey an issue that has recently happened. Questions such as “How do they convey the issue?” “How do they speak naturally?” are relevant.

6.2.1.2 Identify possibilities and limitations for conversational AI and operator interaction

The conversational assistant has technical limitations but also opportunities. The goal here has been set to find an effective and efficient way to communicate with the operators to gather good quality data. Do the operators need to change the way they communicate or should the CA be trained in such a way that it can process the operator’s communication as is? Is there a middle ground for both parties? These are the questions relevant for identifying possibilities and restrictions for conversational AI and operator interaction.

6.2.1.3 Gather contextual knowledge

In the first visit for context analysis, general contextual knowledge was gathered of the factory and how it functions. The goal here is to gather contextual knowledge and an understanding of the operators during issue handling.

6.2.1.4 Testing of audio quality in a noisy environment

The operating line consists of several machines and multiple assembly line pieces that connect these machines. All these instruments utilize energy of which some is converted into sounds or noises. The conversational assistant will need to process audio into speech (speech-to-text), the quality of the audio should suffice for the conversational assistant to be able to process it. The goal here is to test how good the audio would be to recognize the operators voice in a noisy environment. Note, the test was not specifically for testing the microphone, but whether this could result in an issue or not.

6.2.2 Data collection method

The method used to collect the dialogues was a researcher roleplaying a COALA Cognitive Advisor. The researcher would play the role of the Cognitive Advisor while the operators would continue their operating work but also wear a mono-headset which recorded audio. During issue handling, the researcher asked the operators questions related to the issue and recorded these dialogs. Data entities were used to check whether a full descriptive issue was given, these entities were: *Subject, object, type of issue, symptom, location, current task, SKU, RCA* and *solution*. Also, the timestamp, SKU and batch number were noted. Furthermore, a brief description of the issue or context of the issue was written down. Section 6.3 clarifies these entities and mentions the iteration steps done during annotation. During the roleplay, pen and paper were used to mark whether the operator mentioned these entities or whether these were missed during the dialogues. The researcher would try to ask for entities which were not mentioned, while simultaneously trying not to disrupt their work. The recorded audio was to be transcribed and this transcribed data was to be structured in an Excel sheet for annotation.

Although this project is focused on the issue description, during the data collection changeovers and RCA were captured, this is in regard to the contextual knowledge gathering goal in section 6.2.1.3. Capturing these elements of the process allowed for me to better understand the process of the operating line.

6.2.2.1 Annotations of issue descriptions

Due to time constraints a selection of dialogues for two machines were selected for annotation out of the 101 recordings: The filler and the cap supply machine. These two machines were chosen specifically because one machine (the filler) had the most issues, while the other machine (cap supply machine) had many similar issues (e.g., issues regarding the caps getting stuck while being transported to the filler to be placed on the liquid-filled canisters). The idea was that these similar issues which were described several times could give us insights into how the operators describe issues differently. The dialogues were manually transcribed and annotated by two researchers and later compared to maximise accuracy and consistency. In appendix C1 an overview of a view annotation examples is given.

6.2.3 Data results

The data collection resulted into a set of 101 audio recordings of which 71 were verbal issue descriptions, 7 were issue description recordings during changeover and 22 were changeover descriptions. Figure 6.1 shows a histogram of the recorded case per machine.

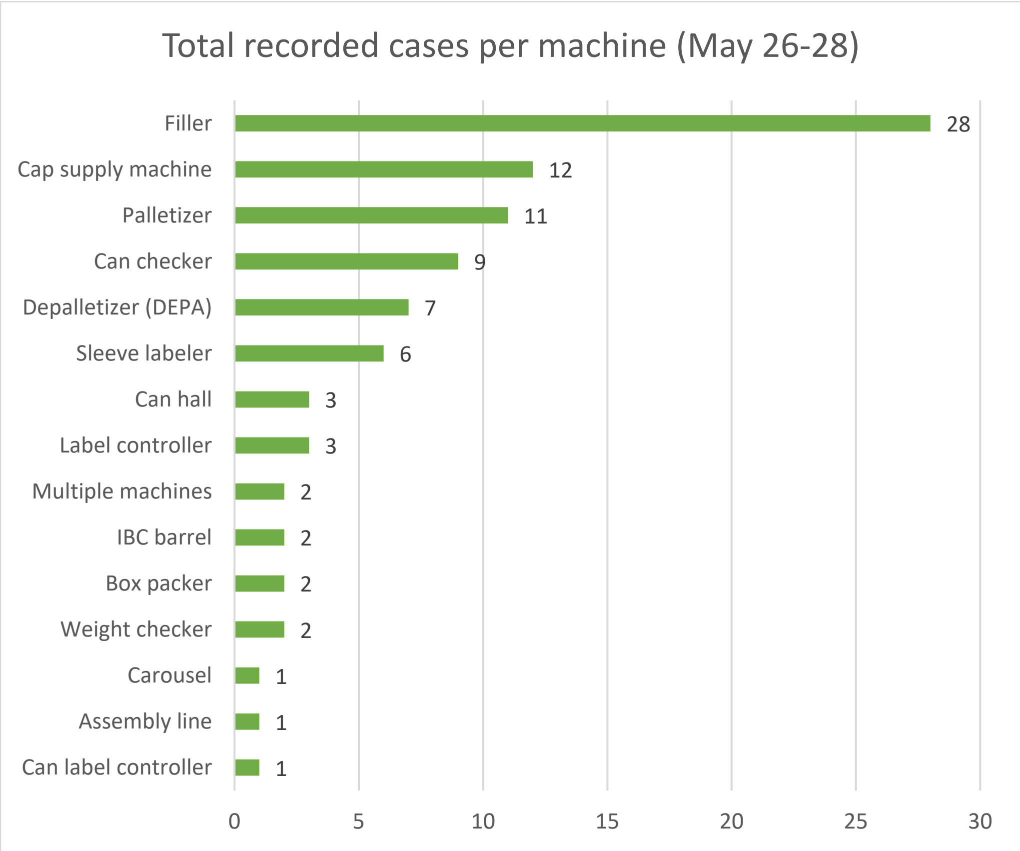


Figure 6.1: Histogram of recorded cases per machine

6.3 Restructuring data entities

Data entities were used to capture information from operators on the issues. During collection the entities used to validate whether enough information was given were: *Subject, object, type of issue, symptom, location, current task, solution,* and RCA (table 6.4).

During the annotation process however, these were updated iteratively, and some modifications were made to the coding scheme of the entities aiming for a more robust entity system which makes a clearer distinction between the entities.

Two researchers were annotating independently and confusion arose during the process, especially for the entities: *subject/object and type of issue/symptom*. One researcher would assign a specific word as subject while the other as object. Furthermore, it became clear that the subject and object were almost always assigned as part of the symptom, therefore these were changed to symptom component and symptom. The way the operators communicated the descriptions, it was difficult to distinguish *symptom* from *type of issue*; therefore type of issue was removed. The SKU was never collected through dialog, but from the production batch paperwork; therefore it was also removed from the dialog collections. The operators would often provide happily and enthusiastically a solution to the issue; consequently solution and solution component were added for annotations. Similarly, but only in some cases of the dialogs, the operators provided the intended behaviour of a certain machine; hence the addition of *intended behaviour* to the annotation entities. Table 6.5 provides the entities used for annotation with its description.

Another iteration was done on the data/annotation entities to make it fit the scope of the project and transform the data entities into entities which Rasa can recognise. Firstly, the solution, solution component, RCA and Intended behaviour were removed due to the scope of the project. Secondly, the remainder was changed to the entities in table 6.6.

A similar issue arose as with the first iteration with the subject and object. In some cases parts of the sentence could be assigned to symptom AND cause and confusion amongst the researchers arose. Instead of using symptoms and causes, the focus was set on the machines and products and the states of them to remove this issue.

Rasa can recognize intents and also entities. However, when a symptom component and a cause component are provided in the same utterance, it is expected that Rasa will have difficulties distinguishing which component is which, unless there is a lot of specific training data that provides examples where for instance, in the first part of the sentence always a symptom component is provided and in the second part the cause component. An example from the data collection session provided in figure 6.2, follows this logic. In green the symptom, in yellow the symptom component, in blue the cause and in red the cause component (table 6.3). However, this sentence can easily be rewritten to the example in figure 6.3. It is unsure whether the model will correctly recognize the entities as it is dependent on the training data of the model.

The combination of the form as mentioned in section 5.1.4.1, with the entities set in table 6.6 is expected to solve this difficulty. This is incorporated into the riskiest assumption explored in sprint 2.

O: The cap turner has now malfunctioned, I think cap turner got overloaded.

Figure 6.2: Example dialog taken from data collection of symptom plus component and cause plus component are present in the same sentence

O: The cap turner got overloaded and therefore the cap turner has now malfunctioned.

Figure 6.3: Rewritten example dialog from figure 5.3.2 where the sentence starts with the cause and cause component

Annotation entities	Colour
Symptom component(s)	Yellow
Symptom(s)	Green
Cause(s)	Blue
Cause component(s)	Red

Table 6.3: Colour coding of annotation entities

Entities for data collection	Definition
Subject	The thing that is performing the action
Object	The thing that is the receiver of the action
Type of issue	The type of the issue

Symptom	The consequence of the issue
Location	The place of the issue
Current task	The current task of the operator
SKU	Stock keeping unit, unique number to identify a certain product
RCA	The root cause analysis of the issue
Solution	The solution of the issue

Table 6.4: Entities used for data collection and their definition

Annotation entities	Description
Symptom component(s)	The main component related to the symptom
Symptom(s)	Observable indication(s) of the issue
Cause(s)	The direct reason behind the issue
Solution(s)	The description on how to solve/deal with issue
Location	The high-level location (in the factory) of the issue
Solution component(s)	Component that is part of the solution
Cause component(s)	Component that is part of the cause
Related part product(s)	The part of the product that is related to the issue
RCA	OPTIONAL – Root cause analysis
Intended behaviour	OPTIONAL – Intended behaviour of machine

Table 6.5: Annotation entities and the definitions used for annotation

Entities for Rasa	Description
Machine location	The high level machine location of the issue
Machine component	The component of the machine related to the issue
Machine component state	The state of the component of the machine related to the issue
Product component	The product component related to the issue
Product component state	The state of the product component related to the issue

Table 6.6: Entities used for conversational assistant prototype in Rasa

6.4 Challenges CA

The issue description dialogs collected in May 2021 provided an indication of how operators naturally describe issue descriptions. These dialogs were used to inform the design of the conversational assistant prototype and as training material for the NLU models that COALA will rely on for interpreting what the operators say.

Table 6.7 was created with goals 6.2.1.1 and 6.2.1.2 in mind. It gives an overview of the challenges with an example, explanation of the challenge for the CA and a possible solution. Eight challenges were found. Seven out of the 8 can be both text-based and voice-based challenges, while the last one is recognized as an only voice-based issue. The examples in the table are dialogs that are either taken directly from the data collection or simplified versions of the dialogs.

Challenges	Example	Challenge for CA	Possible solution CA

1. Missing information due to context	Operator: "Look there, it is stuck." While operator is explaining what is going on with a certain machine, he is pointing towards this machine.	During the interviews, both interviewer and operator were in the same environment and experience a similar "context". The CA can currently not experience a similar context as the operator. For the CA, there is information missing in the form of visual indications provided by the operator.	The assistant lets the operator know it cannot follow, they should use words to describe the situation.
2. Pronoun usage as reference to an object	Operator: "He is stuck."	When the operators use "he/she/it" to reference an object such as a machine or a product, the CA cannot interpret what object the operator is referencing.	The assistant lets the operator know it does not know what they mean by "he" and should use standardized terminology.
3. Multiple entity handling	Operator: "The <u>cap supply machine</u> and <u>the filler</u> have stopped."	It will be difficult for the CA to recognize that both machines have stopped as the CA does not have the cognitive capacity to assign a machine state to two machines at the same time (Stopped for both cap supply machine and filler).	Handle issues separately in isolation or use one state per one component (The cap supply machine has stopped, and the filler has stopped). Entities with roles could provide a solution.
4. Using synonyms or incorrect terminology for objects	Operator: "The <u>filling machine</u> has stopped."	If the CA is unaware of this terminology (filling machine is a synonym for filler), it cannot recognize the entity and therefore has issues with processing the description of the issue.	Develop the CA such that it is aware of synonyms used by the operators for objects such as machines, machine components or products or product components.
5. Reference to unknown entity or component	Operator: "The <u>wrapper</u> has stopped working and therefor the entire 5/10L line has a stoppage."	In this example, the wrapper is a machine that wraps the filled pallets with a plastic wrap. Let's say this is an unknown entity as it is not directly on the 5/10L operating line. Without knowing the entity, it cannot fully understand what the cause is of the stoppage.	Let the CA be aware or be able to learn of a bigger context than just the 5/10L operating line. Or use out-of-scope intents/entities.
6. Response looping	Operator: "Yeah, it had no cap." Interviewer: "Could you elaborate?" Operator: "A cap was put on wrongly."	Response looping refers to when the CA would want the operator to give more information about an issue, but the operator responds with a similar response. In this example the interviewer is looking for what "it" is and what the cause was for "it" not having a cap.	Let the CA be specific in what it wants to know from the operator, instead of "Could you elaborate?" ask "Could you elaborate what the cause was for this?" or "What do you mean by 'it'"?
7. Multiple issue handling	Operator: "The product is foaming at the filler and the capper has stopped because a cap is stuck in the cap supply."	In its current form it is not able to handle multiple issues at the same time, while stoppages can arise simultaneously.	The CA needs to know what problem belongs to what machine. In the example, it needs to know that 'the product foaming' only happens at the filler and a cap being stuck only happens at the cap supply.
8. Starting new sentence mid-sentence (voice related issue)	Operator: "He has to use the first uhhh... On the pallet he places a pallet plate..."	The CA can get confused when the operator starts a new sentence in the middle of another sentence. There is a possibility that the CA cannot process the information as it would be unclear for the CA (not a text-based issue as it is expected	Let the assistant ask the operator to be clearer and simpler in their explanation.

		that the operator thinks before sending a text-based reply).	

Table 6.7: Challenges recognized during the data collection session, included with an example, explanation why it would be a challenge for the CA and a possible solution for the challenge

All these challenges are taken into account during development of the prototype. Furthermore, in sprint 3 two challenges from this data collection are tackled: *Synonyms* and *pronoun usage*. The choice was made to focus on these two challenges on the following bases:

- It is expected that resolving these two would improve data structure.
- Challenge 1 is being tackled by the implementation of ethical posture tracking.
- Challenge 3 would quickly and greatly increase the complexity of training data.
- Challenge 5 could be implemented through an out-of-scope mechanism where the CA tells the user what they are referring to is out of scope and the implementation of this feature is expected to have less benefits for the contribution to the COALA project.
- Challenge 6 is something that will most likely take place during development anyway. Expected is that when training data increases, response looping decreases.
- Challenge 7 complicates the training data and complicates the capture of structured data on issue descriptions. For now, it is better to split the issues into separate issue descriptions.
- Challenge 8 could be resolved by telling the user to repeat their utterance more clearly again.

The audio captured with the mono-headset provided audio with sound cracks and the operators were often times difficult to understand due to clipping. Due to the varying amount of noise at the production line the operators sometimes could speak and communicate in a normal tone of voice while other times they had to shout to make themselves understandable. When shouting, the audio picked up with the microphone of the headset would be distorted. These audio issues could prove to be difficult for a STT system to understand the operators.

6.5 Conclusions sprint 1

With sprint 1, the riskiest assumption of a conversational assistant that can have humanlike conversations has been tested with the data collection and has failed the test due to the undesired chaotic results of humanlike conversations. The resulted dialogs provided an indication of how operators naturally describe issue descriptions. Several challenges were found that would possibly be a predicament for a machine learning algorithm such as an NLU model of a conversational assistant. What became clear is that the output from the issue descriptions should remain structured while the input should be flexible. Therefore for the second sprint the riskiest assumption was to test a frame-based CA.

The restructured data entities were expected to reduce the difficulty in distinguishing the data but had to be validated. This was done in sprint 2.

6.6 List of requirements sprint 1

The following programme of requirements is based on the design thinking phase and on the findings from the first sprint. This list is used to guide the development of the service prototype of the conversational assistant and of future iterations.

Requirements:

1. The prototype communicates through a text-based user interface with the user.

(Chapter 3.4.2) This requirement was set to be able to make quick iterations on the prototype and remove a dependency of STT and TTS systems.

2. The prototype is able to temporarily store user's text-based input.

Being able to temporarily store input, allows for some intellegince

3. The product should collect an issue description within 1 to 5 user utterances.

The lower the amount of user utterances, the lower the possibility of user friction arising.

4. The product is able to correctly recognize intents in 95% of the cases.

(Chapter 3.5) The higher the accuracy, the lower the chances of friction between operator and the prototype.

5. The product is able to correctly recognize entities in 75% of the cases.

(Chapter 3.5) The higher this accuracy, the lower the chances of friction between operator and the prototype.

6. The prototype is able to handle chitchat of the user outside and during collection of the issue description.
7. The product allows the user to stop the gathering of issue description.

Wishes

The product captures issue descriptions with as little as possible user utterances.

The product does not induce user annoyance.

7. SPRINT 2: MVP prototype

This sprint was dedicated to building a prototype CA with a strict output in the form of structured data. In this sprint it was explored what structured data was and what Rasa capabilities could be used to test the riskiest assumption.

7.1 Riskiest assumption sprint 2

In sprint 1 the riskiest assumption of a humanlike conversation with the conversational assistant was explored. The result of sprint 1 indicated that this was not desired. Humanlike conversations can be unstructured and chaotic, while the desired result of a conversation about issue descriptions is a clearly structured dataset, readable by both humans and algorithms. Therefore, the following riskiest assumption for sprint 2 was explored:

Riskiest assumption
Having a frame-based strict pattern for capturing data will provide structured data while allowing for flexible user input.

Table 7.1: Riskiest assumption for sprint 2

The assumption was made here that a strict capturing pattern was needed to provide more structure to the data. However, to maintain the compatibility and user experience as good and high as possible, this dialogue management should still allow for flexibility in the input.

7.2 Setting up the model

To be able to test the riskiest assumption, a model had to be developed. In this chapter the necessary elements are described for setting up the model; pipeline, elements in the domain, training data in the NLU and rules & stories.

7.2.1 Pipeline

In figure 7.2 an adjusted version of the default pipeline is provided. Additions that are not previously mentioned are: SpaCy NLP, EntitiSynonymMapper, ResponseSelector and Fallback-Classifier.

The SpaCy NLP is a pre-trained embedding. Pre-trained embeddings are useful when training data is still limited. However, when training data increases, it is advised to switch to a supervised embedding when domain specific terminology is relevant.

EntitySynonymMapper is used for defining synonyms for entities. Figure 7.1 shows code for synonym mapping. Whenever a user utters “filling machine” the value of the entity is set to “filler”.

```
- synonym: filler
  examples: |
    - fillermachine
    - filling machine
    - fillingmachine
```

Figure 7.1: Training examples for filler synonym

ResponseSelector is used to utter the responses to the user.

With the FallbackClassifier the developer can set a threshold, where, when the highest confidence of a certain action is lower than the threshold, the system falls back to a previous state.

```
language: nl

pipeline:
```

```
  - name: "SpacyNLP"
  # language model to load
  # Dutch large model: nl_core_news_lg
  # Dutch small model: nl_core_news_sm
    model: "nl_core_news_sm"
    case_sensitive: false
  - name: WhitespaceTokenizer
  - name: RegexFeaturizer
  # - name: RegexEntityExtractor
  - name: LexicalSyntacticFeaturizer
  - name: CountVectorsFeaturizer
  - name: CountVectorsFeaturizer
    analyzer: char_wb
    min_ngram: 1
    max_ngram: 4
  - name: DIETClassifier
    entity_recognition: true
    model_confidence: softmax
    constrain_similarities: true
    epochs: 150
  - name: EntitySynonymMapper
  - name: ResponseSelector
    epochs: 150
  - name: FallbackClassifier
    threshold: 0.3
    ambiguity_threshold: 0.1
policies:
  - name: MemoizationPolicy
  - name: TEDPolicy
    model_confidence: softmax
    constrain_similarities: true
    max_history: 10
    epochs: 200
  - name: RulePolicy
```
Figure 7.2: Pipeline used for MVP prototype

7.3 Issue form

With Rasa, information can be captured in slots and a form can be used to guide the user into filling these slots. A form called “issue form” (figure 7.3) was used to capture the issue description and the slots to fill are the following:

- machine location from entity (m_location)

- product component from entity (p_comp)

- product component state from entity (p_comp_state)

```
forms:
  issue_form:
    required_slots:
      m_location:
      - entity: m_location
        type: from_entity
      p_comp_z:
      - entity: p_comp_z
        type: from_entity
      p_comp_state:
      - entity: p_comp_state
        type: from_entity
```

Figure 7.3: Form specification in domain file.

Due to lack of information and training data for machine component (m_comp) and machine component state (m_comp_state), these were for the time being left out of the model. Furthermore, adding more slots to be filled would increase the complexity of the form and possibly the amount of user utterances needed to capture the full issue description. In figure 7.4 a schematic of a happy path of the issue form is provided.

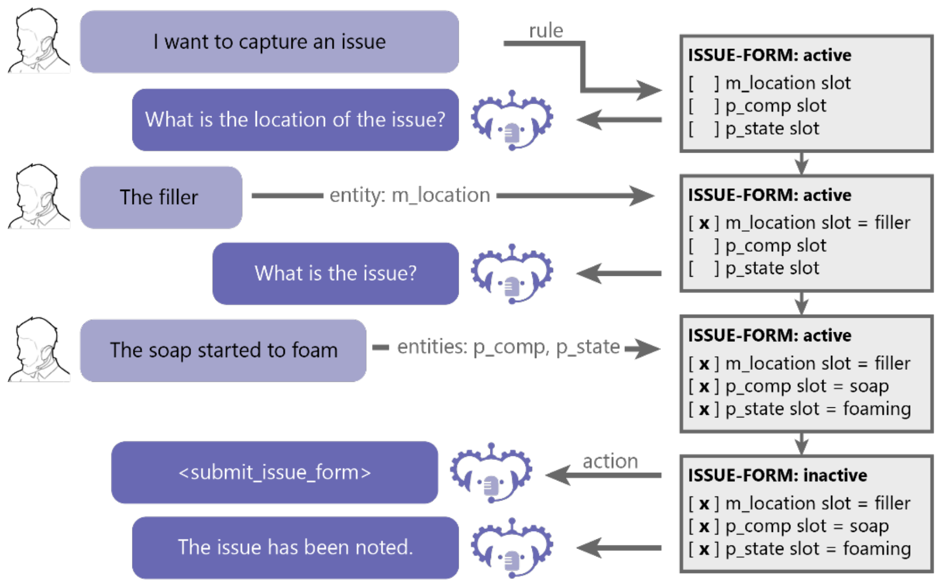


Figure 7.4: Schematic of a happy path conversation with slot filling for issue form.

NOTE: In sprint 3, the form has slightly been adjusted. The order of slots has been changed to provide a simple solution to pronoun usage. More on this in section 8.3.

7.3.1 Buttons in the prototype

The ODCE system uses buttons, figure 7.5, which the operators can press to capture the issue. The CA has been developed to implement such buttons and use them as an intermediate step for the operators with a familiar approach of capturing issues. In figure 7.6 buttons are used to specify the machine location.

Figure 7.5: ODCE options

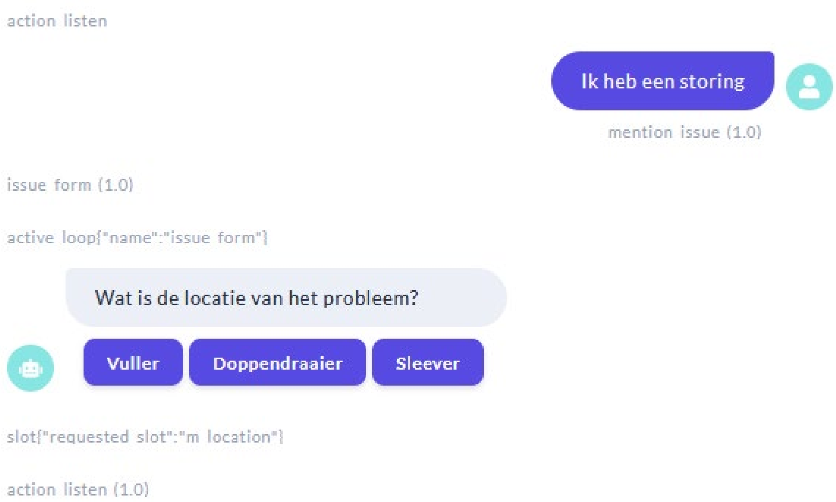


Figure 7.6: Buttons implemented in the issue form for m_location.

7.4 Custom action exploration

This chapter describes explorations done for two custom actions: action_show_time and validate_issue_form.

The first was for understanding implementation of custom actions into the Rasa model. The second was to figure out if it is desired to validate individual input per slot over filling the form and how this could be implemented. Next to that, it was also explored what would be the consequences of these implementations through manually testing the conversation.

7.4.1 Action show time

Firstly, an action was written that would utter the time whenever it was asked for. But to trigger this action a rule needs to be set. Figure 7.7 shows the simple rule. When the user provides the intent “give_time” the action_show_time is always triggered. Figure 7.8 shows the dialog in Rasa X. The model provides the exact date and time.

```
- rule: give time
  steps:
  - intent: give_time
  - action: action_show_time
```

Figure 7.7: Rule for providing the time to the user

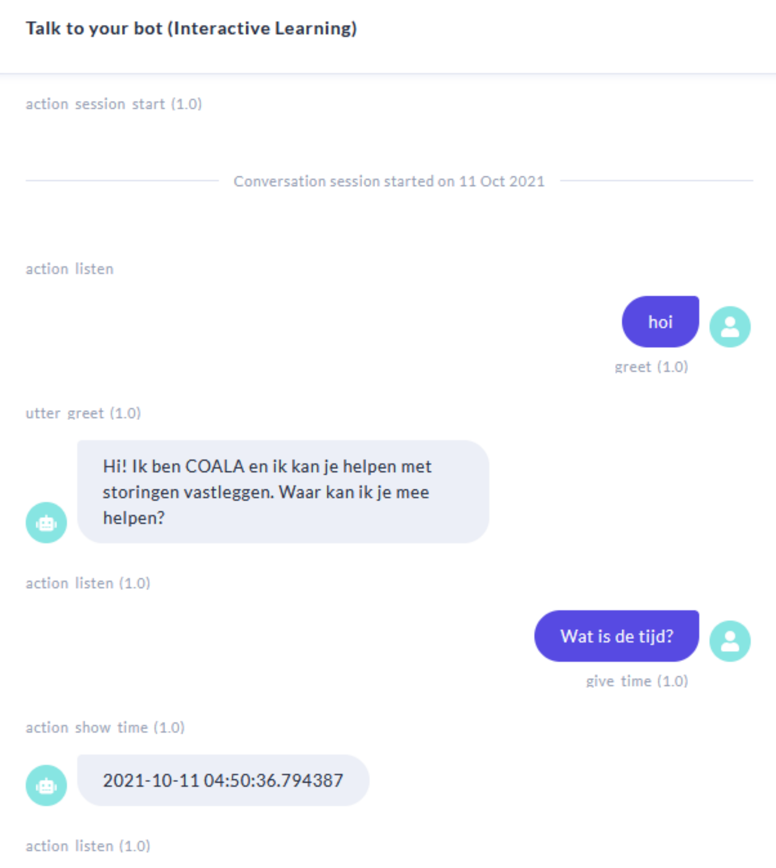


Figure 7.8: Conversation with rule ‘give time’

7.4.2 Validate issue form

Figure 7.9 provides an example where a slot is validated and the message is dispatched by the CA to the user that the machine has been captured. Such validations can be used to provide feedback back to the user. But was left out of implementation in sprint 2 and 3. Reasoning behind this was not to overload the user with messages. However, from the prototype test in sprint 3 (appendix D), it became clear that such feedback is actually good for the user to know whether the system understands their input.

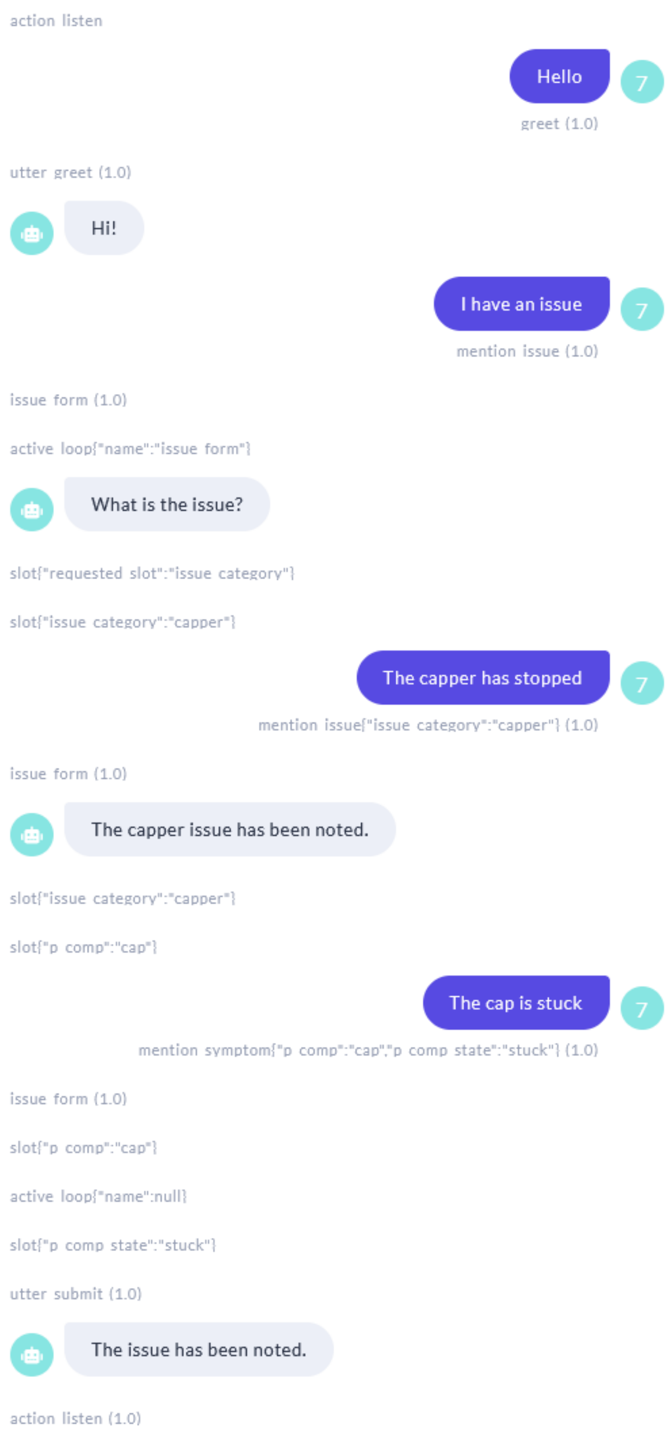


Figure 7.9: Validation of the machine location

7.5 Conversational flow test

With the MVP prototype, a conversational flow test was conducted to test the prototype and improve it. For the full setup, goals, results and findings see appendix B.

The results were 8 conversations with 7 participants. In the conversational flow test, all participants but one failed the first time to perform the task with the prototype without any researchers input. The second round of issue capturing everybody managed to capture the issue. The intent and entity recognition accuracies were 87.6% and 73% respectively.

7.6 Conclusion sprint 2

After testing the prototype, it failed the accuracy requirements. It reached an 87.6% accuracy on intents and 73% accuracy on entities, while 95% and 75% were the requirements for intents and entities respectively. On the one hand, it is expected that through conversation driven development this accuracy will improve, while on the other hand increasing the complexity of the model could possibly result in a reduction of the accuracy, up until there is enough training data after which it is expected that the accuracy should increase again.

From the conversational test it followed that after some clarification on the workings of the CA, the participants managed to succeed with capturing the issue. It is therefore concluded that it would be feasible to implement a CA into manufacturing environments with some training on the functionalities of the CA.

7.7 List of requirements sprint 2

From sprint 1, a list of requirements was set for the development of the CA. In this section, this list is updated.

Requirements from sprint 1:

1. The prototype communicates through a text-based user interface with the user.

2. The prototype is able to temporarily store user's text-based input.

3. The product should collect known issue description within 1 to 5 user utterances.

4. The product is able to correctly recognize intents in 95% of the cases.

5. The product is able to correctly recognize entities in 75% of the cases.

6. The prototype is able to handle chitchat of the user outside and during collection of the issue description.

7. The product allows the user to stop the gathering of issue description.

New requirements:

8. The prototype is able to upload the captured issue description to a database.

Although the form allows for the slots to be set and temporarily store the information it cannot do this permanently. Such information needs to be exported to a database.

9. The prototype is able to extract intents and entities from a database and use as input.

By being able to query from a database, the system can utilize external information not specifically specified in the training data.

10. The prototype is able to filter the issues to machine specific issues from user input.

With this implementation some intelligence is added for a context aware system.

Wishes

1. The product captures issue descriptions with as little as possible user utterances.

8. SPRINT 3: Prototype features

In this sprint several domain specific challenges were tackled and 6 features were explored, tested and implemented in the prototype.

8.1 Riskiest assumption

Because this sprint had a development driven structure the following riskiest assumption was set for sprint 3:

Riskiest assumptions
It’s possible to create a CA that uses and presents standardized terminology while reducing friction between user and CA.

Table 8.1: Riskiest assumption for sprint 3.

With this riskiest assumption, it was explored what features should be implemented to find a balance between using and presenting standardized terminology and at the same time reducing friction between user and CA.

8.2 Synonym handling

Rasa has an architecture that can handle synonyms, EntitySynonymMapper. When included in the pipeline, the model can assign a certain value to a specific user input when correctly trained. For instance, when the user wants to specify the filler (in Dutch: vuller) as input but instead uses a synonym “fillingmachine”, the model can assign the value “vuller” to the input (figure 8.1).

To make this work however, the synonyms need to be included both in the synonym list and in the examples of the training data, as shown in figure 8.2.

Figure 8.1: Conversation example where a synonym is used and the correct value is assigned to it. “Filling-machine” is changed to “vuller”.

```
- intent: mention_issue
  examples: |
      - ik heb een probleem met de [vuller](m_location)
      - probleem met de [fillingmachine](m_location)

- synonym: vuller
  examples: |
    - vul machine
    - vulmachine
    - vullermachine
    - filler
    - filling machine
    - fillingmachine
```

Figure 8.2: Code in the domain file with intent mention_issue with training examples for both the “vuller” and “filling machine”

8.2.1 Initial synonym handling ideation

Initial ideation of synonym handling was to correct the operators on their terminology as shown in figure 8.3. However, it was expected such a correction would add an element of friction and frustration. Next to that, it slows down the process of capturing issues. Therefor it was not implemented in the system and opted for a more flexible approach presented in 8.2.

Operator: The filling machine has a stoppage. CA: I believe you mean filler by 'filling machine', correct? If so, can you use standardized terminology next time?

Figure 8.3: Correction of the operator during the process of issue capture

8.3 Pronoun usage

During the data collection in sprint 1, operators have used pronouns to mention a certain machine or product instead of using the correct term. Words such as "he", "she", "it" or "there" were used to denote machines and product components.

The order of the required slots as presented in figure 7.3 in chapter 7.3 was restructured to the order in figure 8.3. Instead of capturing the issue in the order of machine location -> product component -> product component state, the issue was captured with the following order: machine location -> product component state-> product component.
Product component states cannot be referred to with pronouns. So when that slot is filled after the question "What is the issue?" and the user does not use terms the CA needs to set the product component slot it will ask to specify the product.

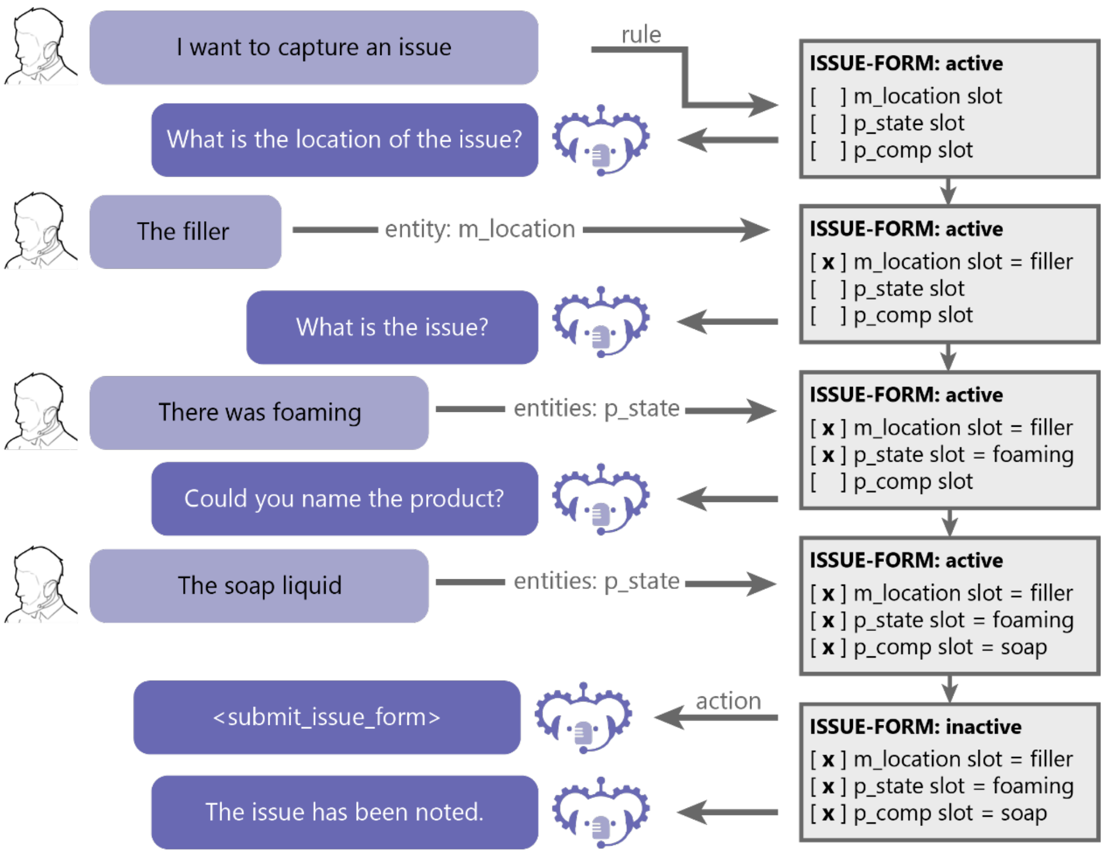


Figure 8.3: Restructured issue form. Handling pronouns is built in the form itself.

8.3.1 Initial pronoun usage ideation

The initial ideation for handling pronouns was similar to the initial ideation for synonym handling in 8.2.1. Whenever the user would use a pronoun, the system would correct them on it (figure 8.4).

Operator: It has stopped. CA: What is the location? Operator: Cap turner. CA: What is the issue? Operator: It is stuck. CA: What do you mean by “it”? Operator: The cap.

Figure 8.4: Initial ideation of pronoun correction

```
- intent: mention_symptom
  examples: |
    - [Het](pronoun) vloeit over
    - Het [product](p_comp) vloeit over
```

Figure 8.5: Training examples which could complicate the training data for the model and possibly decrease the accuracy

To implement such a mechanism, either pronouns have to be specified as entities in the training data (figure 8.5) or a custom action needs to be created to process pronouns. This would significantly increase the complexity of the model which will slow down the training process of the model and the response time of the CA. Also, a high probability is that the accuracy would reduce due to the increase complexity, which is not desired. Therefor it was opted for the simpler solution presented in 8.3.

8.4 Writing issue description to a database

With the issue form, the CA can temporarily store the slots while the conversation is running. If this process is halted, the information is lost. Therefor, an action has been developed which stores the slots set during the conversations to a database. The Google Drive API is used for exporting the issue descriptions to a sheet. The logic for the action is shown in figure 8.6.

The slots are collected from the issue form and the date and time is included. These values are put into a list and this list is exported to a google drive google sheet (table 8.2). After uploading, the CA utters the message "utter_submit_issue_form" and clears all slots, so that a new issue can be captured.

1	m_location	p_comp	p_comp_state	date	time
2	vuller	product	schuimen	14/09/2021	09:08:16
3	vuller	product	schuimt	14/09/2021	09:23:57
4	doppendraaier	dop	vast	14/09/2021	09:33:31
5	vuller	product	schuimen	14/09/2021	12:39:16
6	doppendraaier	dop	geblokkeerd	14/09/2021	13:19:11
7	doppendraaier	dop	vast	14/09/2021	13:48:09
8	doppendraaier	dop	vast	14/09/2021	13:48:10
9	doppendraaier	dop	vast	14/09/2021	13:48:19
10	doppendraaier	dop	vast	14/09/2021	13:48:31
11	doppendraaier	dop	vast	14/09/2021	13:48:54
12	doppendraaier	doppen	dop	14/09/2021	13:53:52
13	vuller	product	schuimen	14/09/2021	14:14:20
14	vuller	vloeistof	loopt eruit	16/09/2021	11:07:38
15	vuller	product	overstroomt	16/09/2021	11:41:38
16	vuller	Vloeistof	schuimt	16/09/2021	11:45:03
17	vuller	Vloeistof	loopt eruit	16/09/2021	11:55:44

Table 8.2: Google sheet with captured entities, date and time of issue.

```python
# Upload to Google Drive
class ActionSubmitIssueForm(Action):
    def name(self) -> Text:
        return "action_submit_issue_form"

    def run(
        self,
        dispatcher: CollectingDispatcher,
        tracker: Tracker,
        domain: DomainDict,
    ) -> List[EventType]:
        """Once we have all the information, attempt to add it to the
        Google Drive database"""

        import datetime
        # collect slots from form
        m_loc = tracker.get_slot("m_location")
        p_comp = tracker.get_slot("p_comp_z")
        p_state = tracker.get_slot("p_comp_state")
        # get date and time
        date = datetime.datetime.now().strftime("%d/%m/%Y")
        time = datetime.datetime.now().strftime("%H:%M:%S")

        # Create a list of all the slots
        issue_info = [m_loc, p_comp, p_state, date, time]
        # try to upload to google drive
        try:
            gdrive = GDriveService()
            gdrive.append_row(
                gdrive.ISSUE_SPREADSHEET_NAME,
                gdrive.ISSUE_WORKSHEET_NAME,
                issue_info
            )
            dispatcher.utter_message(template="utter_submit_issue")
            return [AllSlotsReset()]
        except Exception as e:
            logger.error(
                f"Failed to write data to gdocs. Error: {e.message}",
                exc_info=True,
            )
            dispatcher.utter_message(template="utter_issue_submit_failed")
            return []
```

Figure 8.6: Code for sending the slots to the google drive sheet.

8.5 Contextual filtering

With contextual filtering it was explored whether it would be feasible for the CA to filter towards user input. The idea was set to, depending on what machine location was mentioned by the user, the top 3 issues for that specific machine would be presented to the user in the form of buttons.

#	issue_description	m_loc	p_comp	p_state
1	vloeistof schuimen	vuller	product	schuimen
2	vloeistof schuimen	vuller	vloeistof	schuimen
3	vloeistof schuimen	vuller	vloeistof	schuimt
4	vloeistof schuimen	vuller	product	schuimt
5	overvloeien vloeistof	vuller	product	loopt eruit
6	overvloeien vloeistof	vuller	vloeistof	loopt eruit
7	overvloeien vloeistof	vuller	vloeistof	overgestroomt
8	plaatsing cans verkeerd	vuller	cans	verkeerd geplaatst
9	geen flacon bij flacon sensor	vuller	can	missend
10	geen flacon bij flacon sensor	vuller	can	ontbreekt
11	vuller asynchroon met transportband	vuller		
12	vuller asynchroon met transportband	vuller		
13	dop blokkeert aanvoer	doppendraaier	dop	vast
14	dop verkeerd geplaatst op flacon	doppendraaier	dop	verkeerd geplaatst
15	sleeve gekreukeld geplaatst	sleeve etiketteerder	sleeve	gekreukeld
16	sleeve gescheurd	sleeve etiketteerder	sleeve	gescheurd
17	sleeve rol is op	sleeve etiketteerder	sleeve rol	is op

Table 8.3: A comma-separated values file (csv-file) was used for development of contextual filtering. Included in the dataset, are the index, the issue description, the machine location, the product component and product component state.

In table 8.3 is the dataset used for testing contextual filtering and in figure 8.7 the code used in for contextual filtering. The Pandas library is used for reading and interpreting the csv-file in the action.

In the code, the class was specified with the name: "action_ask_p_comp_state" and is specified in the domain so that when the form is looking for the product component state, this action is executed. This is due to a built-in mechanism in Rasa where when the model is looking for a specific slot, it will firstly do a certain action. The simplest action is a response/utterance, however custom actions are also a possibility. If the action is named as follows: "action_ask_{slot_name}" and the slot name is set, in this case, as "p_comp_state" it will do this action whenever the model is looking for this slot.

The logic is as follows:

If the machine location input from the user is a value under the "m_loc" column in the csv-file, it will give a list of the issue descriptions (duplicates are included) for that specific machine. After that it will count all values and list them from the highest duplicate value to lowest. The next step is removing the duplicate values but keeping the order from highest to lowest. To get the top 3 issues with highest duplicates this final list is sorted to the indexes and out come the top 3 issues. In figure 8.8 you can see the lists/dictionaries that form after each step in the logic. The top 3 list can now be printed to the buttons.

```
import pandas as pd
from typing import Dict, Text, List, Optional, Any
from rasa_sdk import Action, Tracker
from rasa_sdk.executor import CollectingDispatcher


class AskForProductComponentStateAction(Action):

    def name(self) -> Text:
        return "action_ask_p_comp_state"

    def run(
        self, dispatcher: CollectingDispatcher, tracker: Tracker, domain: Dict
    ) -> List[EventType]:
        m_location = tracker.get_slot('m_location')
        found = df[df['m_loc'].str.contains(m_location)]
        if len(found) > 0:
             """If user input is in database value, attempt to provide buttons
             for issue description related to p_comp_state and p_comp"""

            df_issue = df.loc[df['m_loc'] == m_location, 'issue_description']

            # Counts how many duplicates per instance
            counts = dict()
            for i in df_issue:
                counts[i] = counts.get(i, 0) + 1
            # Sorts the values with highest duplicates from highest to lowest
            sort = sorted(counts, key=counts.get, reverse=True)
            # Create a list of of top 3 values by index
            top3 = []
            for idx, x in enumerate(sort):
                if(idx > 2):

                    break
                top3.append(x)
             # Utters a message and provides the buttons for top 3 issues
            dispatcher.utter_message(text=f"Wat is het voornaamste probleem
            dat je kan observeren? Hierbij de top 3 problemen. Als het een
            ander probleem is kan je het ook typen.", buttons=[{"title": x,
            "payload": x} for x in top3])
        else:
             dispatcher.utter_message(text=f"Dit is onbekend voor mij, wil je
             het toch vastleggen?")
            # Here comes the code to go into unknown product issue form.
        return []
```

Figure 8.7: Code snippet for the contextual filtering action.

```
df_issue:

0                 vloeistof schuimen
1                 vloeistof schuimen
2                 vloeistof schuimen
3                 vloeistof schuimen
4               overvloeien vloeistof
5               overvloeien vloeistof
6               overvloeien vloeistof
7             plaatsing cans verkeerd
8        geen flacon bij flacon sensor
9        geen flacon bij flacon sensor
10   vuller asynchroon met transportband
11   vuller asynchroon met transportband
Name: issue_description, dtype: object

counts:  {'vloeistof schuimen': 4, 'overvloeien vloeistof': 3, 'plaatsing cans verkeerd': 1, 'geen flacon bij flacon sensor': 2, 'vuller asynchroon met transportband': 2}

sorted:  ['vloeistof schuimen', 'overvloeien vloeistof', 'geen flacon bij flacon sensor', 'vuller asynchroon met transportband', 'plaatsing cans verkeerd']

top3:    ['vloeistof schuimen', 'overvloeien vloeistof', 'geen flacon bij flacon sensor']
```

Figure 8.8: Terminal print for machine location = filler of the df_issue list, counts dictionary, sorted list and top3 list in the code snippet from figure 8.7

In figure 8.9 we can see the code in action in a Rasa X conversation. The user put in "filler" (in Dutch: vuller) for machine location and the buttons contain the top 3 issues for the filler: *'vloeistof schuimen', 'overvloeien vloeistof', 'geen flacon bij flacon sensor'.*

Figure 8.9: Rasa X conversation snippet where the contextual filtering provides the top 3 issues related to the filler (in Dutch: vuller)

In figure 8.10 another example where the sleeve labeler (in Dutch: sleeve etiketteerder) is mentioned by the user and the contextual filtering provides its top 3 issue descriptions.

Figure 8.10: Contextual filtering for sleeve labeler

A benefit of the contextual filtering is that no extra stories or rules are needed to implement the feature. It has been integrated into the form handling function. Furthermore, if the database of writing issue description and the database used for contextual filtering are connected, the entire system is scalable for all issues at the Diversey production line.

In the prototype features test it was found that it directly provides feedback to the user whether it understood them.

8.5.1 Limitations with current contextual filtering

As the input from the buttons is taken from the issue descriptions in the csv-file, the words in the issue descriptions have to be also present in the NLU training data to recognize the input. An easy fix would be to gather the slots for product component and product component state from text instead of from entities. However, anything can then be set for the slot while with entities it can be controlled what is recognized as correct input and what is not. However, this solution is currently not desired as the system needs to check what is being put in by the user.

Another solution would be to use the product component and its states listed in the csv-file. However, due to several entities could give the same description of the issue (e.g. product foaming / liquid foaming), the system would provide such similar responses back to the user if they are in the top 3 issues, which is undesired. The system should provide unique issue descriptions per button.

Next to that, due to taking the top 3 indices for the top 3 issues from the sorted list, the *'geen flacon by flacon sensor'* issue description is in the top 3 list while *'vuller asynchroon met transportband'* has the same number of duplicates as '*geen flacon by flacon sensor*' (both have 2). This could create a bias towards certain issues based on the index number and should be taken into account in future work.

Furthermore, the action of writing to a dataset mentioned in 8.4 and reading from a dataset discussed in this section are not yet operating with the same dataset. These two actions have to be connected in future implementations by using the same dataset.

8.6 Issue description check

During the development in sprint 3, an extra control step was added to the prototype to add a check whether the issue is correctly described which also provides feedback back to the user.

Good practice at Diversey always includes a controlling check of an act of one party by another party. One example is the "double check" (in Dutch: vier-ogen check), where one operator controls whether everything has been correctly set for the change-over by the operator responsible for the change-over.
For the CA, the idea is similar where the operators are the controlling party that checks whether the CA captured the issue correctly.

In figure 8.11 a conversation is shown where the user captures a filler issue. At the end of the issue description, the CA utters a check. The check reads as follows: *"Is the location "filler", the product component "liquid" and the status of the product component "foaming?""* After which the user can either respond with an affirm or deny. If everything went correctly the user can affirm and the issue will be captured, if the user denies, the CA will exit the form and start over with the issue capturing.

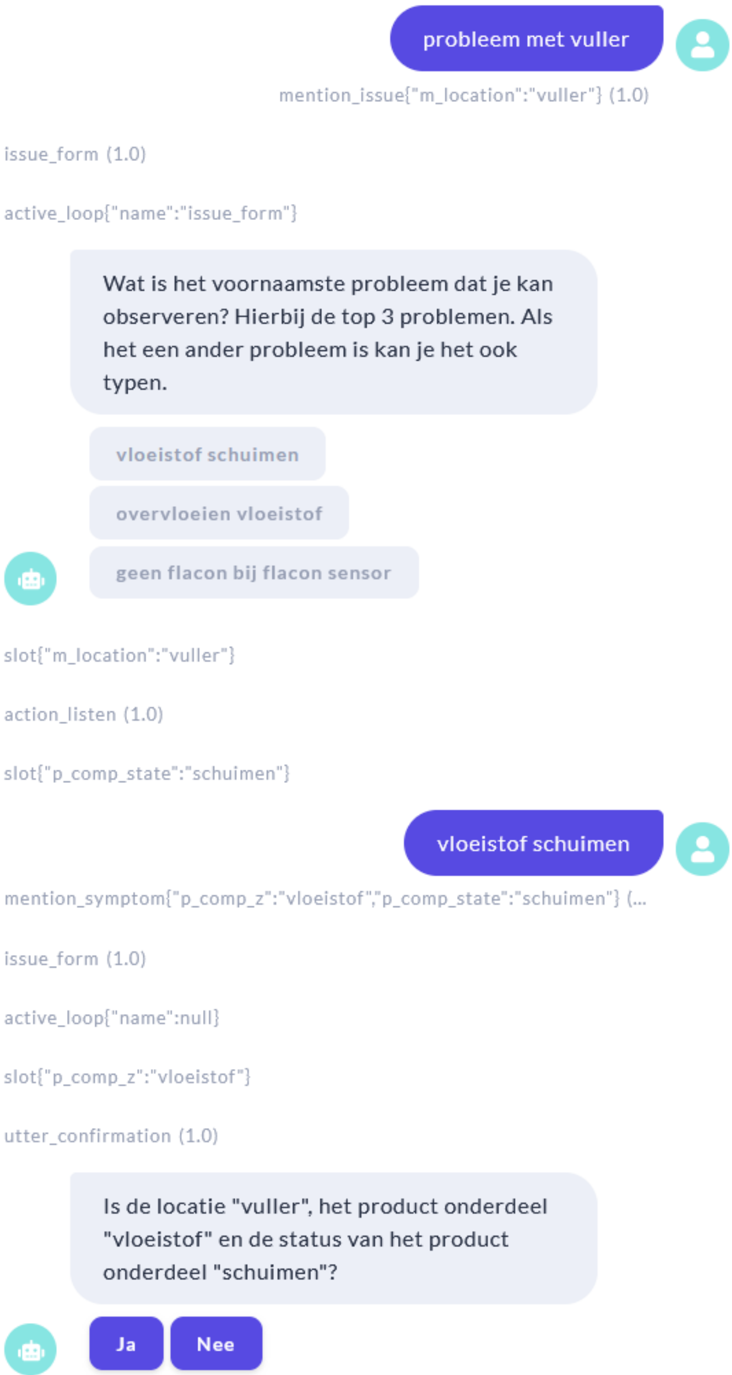


Figure 8.11: Check in a conversation where the user captures an issue at the filler. The check reads as follows: *"Is the location "filler", the product component "liquid" and the status of the product component "foaming?""*

8.7 Unknown issue form

Although previous steps were focused on implementations of a system with the assumption that it had already the right knowledge of issues/stoppages at the production line. That way it could be tested whether the system would work for capturing issues and filtering towards the context of the conversation. However to get to this stage, the system needs the data on all issues at the production line. The unknown issue form was implemented to allow operators to capture these issue descriptions and expand the knowledge base of the CA.

The unknown issue form requires the same slots to be set as the issue form, namely: the machine location, product component, product component state. However, instead of extracting it from entities, it will use text to fill the slots. Whatever text is given, is captured for required slot. This way, new unknown issue to the CA can be captured and added to the database. In figure 8.12 the code in the domain file is given for the unknown issue form and its responses.

```
Forms:
    unknown_product_issue_form:
    required_slots:
      m_location:
      - type: from_text
      p_comp_state:
      - type: from_text
      p_comp_z:
      - type: from_text
Responses:
  utter_start_unknown_product_issue_form:
      - text: Oke, we starten nu met het vastleggen van een onbekend product
      probleem, het is belangrijk dat de input die je geeft correct geschreve
      is.
      - text: We starten nu met het vastleggen van een onbekend product
      probleem, het is belangrijk dat de input die je geeft correct geschreve
      is.
  utter_ask_unknown_product_issue_form_m_location:
      - text: Wat is de machine locatie van het probleem?
  utter_ask_unknown_product_issue_form_p_comp_z:
      - text: Oke, probleem is bij {m_location}, en wat is het product
      component gerelateerd aan het probleem?
      - text: Oke, probleem is bij {m_location}, en wat is het product
      onderdeel gerelateerd aan het probleem?
  utter_ask_unknown_product_issue_form_p_comp_state:
      - text: '{p_comp_z}, duidelijk. Wat is de status van het product
      component?'
      - text: '{p_comp_z}, duidelijk. Wat is de status van het product
      onderdeel?'
```

Figure 8.12: The unknown issue form specified in the domain file where the slots are captured from text and its responses for the required slots.

Figure 8.14 shows a conversation in Dutch for capturing an issue at the palletiser machine where the canisters are incorrectly placed. Translated to English the conversation is as follows (figure 8.13):

Operator: I want to capture an unknown issue.
CA: Ok, let's start capturing the unknown issue. It is important that you write your input correctly. What is the machine location of the issue?
Operator: Palletiser
CA: Issue with the palletiser, what is the product component related to the issue?
Operator: Canister
CA: Canister. What is the status of the product component?

Operator: Wrongly positioned
CA: Is the location "palletiser", the product component "canister" and the status of the product component "wrongly positioned?" (Buttons = [Affirm, deny])
Operator: Yes
CA: Great, the issue has been stored to the database.

Figure 8.13: Translated conversation from figure 8.14

```
Bot loaded. Type a message and press enter (use '/stop' to exit):
Your input ->  Ik wil een onbekend probleem vastleggen
Oke, we starten nu met het vastleggen van een onbekend product probleem,
 het is belangrijk dat de input die je geeft correct geschreven is.
Wat is de machine locatie van het probleem?
Your input ->  palletiser

- action: unknown_product_issue_form
- slot_was_set:
  - m_location: palletiser

Oke, probleem is bij palletiser, en wat is het product onderdeel gerelat
eerd aan het probleem?
Your input ->  canister

- action: unknown_product_issue_form
- slot_was_set:
  - p_comp_z: canister

canister, duidelijk. Wat is de status van het product onderdeel?
Your input ->  verkeerd gepositioneerd

- action: unknown_product_issue_form
- slot_was_set:
  - p_comp_state: verkeerd gepositioneerd

? Is de locatie "palletiser", het product onderdeel "canister" en de st
atus van het product onderdeel "verkeerd gepositioneerd"?  1: Ja (/affi
rm)
Top, de storing is vastgelegd en data is gestuurd naar de database.
```

Figure 8.14: A snippet of the terminal in which the conversation is tested for capturing unknown issues

8.8 Prototype feature test

A test was conducted with two design students with knowledge on the production line to validate the implemented features. Two versions were tested of the prototype. One had buttons present and the other had none. The one with buttons had the contextual filtering included, while the other had not. For the full setup and results see appendices D1 through D4.

Two important findings followed from this test.
Firstly, in the version with no buttons, feedback during the process of data capture was missing. The users were not aware what was captured while this was desired. Secondly, with the buttons, the participants mentioned having a bias towards the buttons. They would try to describe the issue with a button that would correspond the most with what they observed.

9. Conclusions & Recommendations

This chapter discusses the final outcome of this graduation project as well as recommendations for future development.

9.1 Conclusions

The goal of this project is to develop a human-centred conversational assistant for collecting standardized issue descriptions from operators on the high-priority Diversey production line.

To reach this goal the following research questions:

- Why use a conversational assistant for data acquisition in manufacturing?
- What is the added benefit of a conversational assistant for data acquisition in manufacturing?
- How can a conversational assistant be designed for data acquisition of standardized issue descriptions?

The first two research questions were explored and answered in the Design Thinking phase (chapter 3.3 and 3.4). Conversational assistants can be used to handle and process repetitive data, such as addressing customer inquiries or gathering data at scale. For Diversey specifically the conversational assistant can gather data on issues, root causes, solutions and best practices. Current methods of data acquisition at Diversey have only focused on reporting stoppages, with a conversational assistant a rigorous structure can be developed to expand the data acquired and create value for the company.
An extra benefit of a voice-enabled assistant is that the user can do the cognitive tasks hands-free with the assistant. Hence, the operators can both maintain the line and do the cognitive tasks simultaneously.
When the assistant has the necessary knowledge, it can provide support to the operators through recommendations on best practices.

The last research question was used as a guideline for the development of the conversational assistant. The result is the prototype of the CA which can acquire data on product related issue descriptions, which is one of the steps towards an operable Cognitive Advisor on the shopfloor. The prototype includes a system to process issue descriptions that are known to the system. It includes features such as synonym understanding, contextual filtering and a check for the issue descriptions that improve upon useability of the assistant and move towards standardized and structured data. Next to that, a feature was implemented to capture unknown issues to the model. With this, operators are able to expand the knowledge base of the CA through conversations.

The prototype and its features are reliant on the training data to improve accuracy. The development of the prototype should remain an iterative process to keep improving the NLU model while expanding the dataset on issue descriptions.

9.1.1 Context analysis

The context analysis conducted at the production line at Diversey showed that the operators primarily resort to their own intuition and experience to resolve issues. It showed that the system in place of capturing such implicit and explicit knowledge is not utilized optimally. Evaluations with operators showed that the system is an inconvenience and a time sink. Simpler and quicker tools were used to communicate information between shifts. The data that was captured with the system was often ambiguous, incomplete and non-descriptive. However, such data can be of significant value to the company as it can be used for process improvement of the production line, maintenance can make use of the data to identify and locate the root of the cause of a major stoppage and the knowledge that is acquired can be carried over to other operators.

9.1.2 Conversational data

In sprint 1, through a COALA-roleplay experiment, it was explored what kind of dialogue management was needed for the conversational assistant with the assumption that humanlike conversations were needed. The resulted dialogs provided an indication of how operators naturally describe issue descriptions. Several speech-related challenges were found that would possibly be a predicament for a machine learning algorithm such as an NLU model of a conversational assistant. The dialogs were full of information and knowledge on the issues that arose during the experiment and the fixes to resolve them. However, these conversations were chaotic and unstructured. So unstructured, it was difficult for two researchers to find common ground without the context of the experiment, let alone for a machine learning algorithm that needs to make sense out of the data.

A balance should be struck where the operators have some flexibility how they capture the issue with the CA, while their input can still be understood by the NLU model and processed into data that can be used for future training data of the CA and be of value to Diversey.

9.1.3 MVP prototype

In sprint 2 a minimum viable product prototype was developed for the task, that simultaneously allowed flexible input for the user, while, through a strict pattern of gathering issue descriptions, could create structured data. During testing of the prototype, all participants except one, failed to fully capture the provided issue with the CA. The MVP prototype failed the intent accuracy requirement by a difference of 7.4% and its entity accuracy requirement by 2%. With some assistance and knowledge on how the CA functions however, all participants were able to complete the issue capture.

Although the participants failed to initially capture the issue and the CA failing the accuracy requirements, it provided an indication that with some information or training on the functionality of the CA, it would be feasible to implement such a CA in a manufacturing environment.

To make a proper statement on the feasibility of a CA in such a context however, more user tests need to be conducted in the future with the operators.

9.1.4 Prototype features

Within sprint 3, several features were added to the prototype focused on standardizing the data, providing feedback to the user and reducing friction during the process of data acquisition.

9.1.4.1 Synonym handling and pronoun usage

Firstly, two challenges from sprint 1 were tackled: Synonym handling and pronoun usage.

Synonyms and pronouns were often used by the operators to denote a certain machine, machine component or product component.

With Rasa, it is possible to implement a mechanism that would assign a value to a certain word. This allows to change the value of a synonym to a standardized term (i.e., ‘filling machine’ is changed to ‘filler’). This way, the operators have some flexibility in terminology of the input, while still providing the correct terms in the data. Furthermore, during capturing of the issue,

the operators are presented with this adjusted value. This gives the operator the feedback what the correct terminology is. Although, in its current state, it is up to the operator whether they implement such terminology in their vocabulary or remain using the synonyms because the CA can process it.
For both synonyms and pronouns, it was explored whether the CA would respond with a correction on user errors. But the choice was made not to implement such a feature, as it would increase the friction in the interaction with the CA. To ensure a seamless experience and make users engage more with the CA, such frictions should be either fixed or removed (Ponnusamy et al, 2020). For pronoun usage a simpler solution was opted for where the order of the form was adjusted, such that the usage of pronouns would be resolved by the form.

9.1.4.2 Contextual filtering

Contextual filtering compares the user's input, to a dataset with known issues. If the input of the user has the same value as a value in the dataset, it will filter towards this input. Creating a conversation for issue description with context specific information. The contextual filtering feature present the top 3 issues in buttons (if available), related to the machine location provided by the user. Although it was concluded from the prototype feature test that the feature decreased the duration of issue capturing, the participants of the test had a bias towards describing issues with the buttons. They would use a button that was closest to the description of the issue as possible, instead of trying to create the best description for the issue.
What is expected, is that if small issues arise often, this feature can be used to quickly capture such small issues and prevent operators having to type out such problems repeatedly.

9.1.4.3 Issue description check

The issue description check can be used by the operators to control their input and whether the CA correctly captured it. The benefit of both the check and the contextual filtering is that it provides feedback back to the user, making the user aware whether the system understood their input.

9.1.4.4 Unknown issue form

The unknown issue form can be used to capture unknown issues. The knowledge base of the CA will need to be expanded to be able to properly capture issues. With this, the operators can capture unknown issues and write to a database.

9.2 Recommendations

Recommendations for future work are listed below.

9.2.1 Setup at Diversey

In the conclusion, it has been mentioned already that more data is needed. What would greatly increase the data, is to make the system for capturing issues operable at Diversey. Two elements are needed to make it operable for capturing issue descriptions: a server with Rasa that can continuously run and hardware which can be used to type on and that can open the server, such as a mobile phone or a tablet. In the prototype feature test (appendix D1) it was already proven that issues could be captured remotely through a combination of a local server running on a laptop and an ngrok-server which makes a local server publicly available through an URL. This way, the participants were able to use the prototype through a link.

9.2.2 Onboarding of COALA into the Diversey process

Although some steps have already been made towards introducing COALA at Diversey, to make it work on the shop floor the benefits of such a system need to be proven to the operators. And when it is introduced, I would suggest an onboarding process, where the operators can familiarize with COALA. In the conversational flow test, all participants but one failed the first time to perform the task with the prototype without any researchers input. The second round of performing the task during the test, all succeeded. Having an understanding of what the assistant needs from the participant, improves the performance of the task.

9.2.3 Additional features

Below, are additional features that could be implemented to the prototype.

9.2.3.1 Connect issue form with unknown issue form

With the current setup of the issue form, whenever a slot is not set, it would utter the previous response to the user. What could be implemented is a mechanism that after a certain number of failed attempts of setting a slot, the CA will attempt to move the conversation to the unknown issue form and start an unknown issue capture. Before starting the unknown issue form however, the user has to affirm that this is the action they want the CA to take.

9.2.3.2 Tuning the algorithm to also work for machine related issues

Sometimes issues can also just be machine related and no product related. It is important to capture such issues as well. The question now is how can these be captured?

The capturing of issues could be, for instance, separated by machine and product. A solution would be to have two forms for issue capturing, one for machine related issues and one for product related. Whenever the model wants to try to capture an issue, the CA could prompt an utterance "Is this a machine related issue or a product related issue?" That way the operator could select what form to use. However, this requires quick observable skill by the operator to know what kind of issue it is.

9.2.3.3 Add optional information for issue description

The current method of issue description is only focused on the machine location, product component and the state of the product component. Nonetheless, more information is needed to create the data complete. Information about the machine component related to the issue and its state, for instance, are not captured. However, for issues, not all information is relevant all the time. A product issue can have no related machine component, while a machine issue can create a stoppage where the product is not at all relevant. By adding optional information to the issue form, the user is asked whether additional information is relevant and should be captured.

9.2.3.4 Adjust issue description input

An implementation should be made such that the user can correct the input in the form when it has been wrongly set. Currently at the check, when the user denies the question whether all input has been correctly captured, it will clear the slots and restart the form. Instead of fully clearing the slots, the operator could specify the slot they want to replace and replace it.

9.2.3.5 Connect the two databases into one

Currently, the database from which the contextual filtering feature queries from, is not connect to the database the system writes to. By connecting the two, information can be queried from and issue can be capture to the same database.
Another option to consider, is to have two databases: One with all the issues that can possibly happen at the production line and use the contextual filtering action to read from this database. And a knowledge base that has all the possible unique issues that can arise.

9.2.3.6 Action for training of the model

With the current prototype, whenever changes have been applied to the NLU or NLP model, it needs to be manually trained to include the changes. What is proposed here, is that an action is implemented where the operators can actuate the model through the conversation to train itself on the new data that was stored in the database. This newly trained model can then process the newly acquired known issues and unknown issues. Iteratively, issues can then be captured, the model can train and learn and be used to capture issues again.

Although, such a process is not instant and could possibly hinder the operators from capturing new issue descriptions during the training process of the model, the operators should be careful when they would start this action. A possibility is to automatically train the model during downtimes which are not caused by stoppages, such as the end of the shift of the operators, change-overs or during the cleaning process of the filler.

9.3 Proposal of Machine to machine learning with human post-processing

Another approach for increasing the amount of data aside from the one mentioned in section 9.2.1, is to use machine to machine (M2M) learning as proposed by Shah et al. (2018). While keeping the naturalness of individual utterances, M2M delivers increased diversity and coverage of dialogue flows compared to a Wizard-of-Oz approach according to Shah. The approach is separated into two phases: A simulated user bot and a domain-specific bot converse to generate large quantities of dialogue templates. In the second phase, people use these dialogue templates to create more natural contextual rewrites. Thus creating a large dataset of conversations which can be used for training of CAs.

10.0 References

Böhm, H. (2002). *Dialogue management using Finite State Models* [Slides]. Saarland University. http://www.coli.uni-saarland.de/~korbay/Courses/DM-SS02/DM-slides/hagen-fsm-slides.pdf

Brendel, A. B., Greve, M., Diederich, S., Buehrke, J., & Kolbe, L. M. (2020, August 12). *'You are an Idiot!' - How Conversational Agent Communication Patterns Influence Frustration and Harassment*. ResearchGate. Retrieved 11 November 2021, from https://www.researchgate.net/publication/341381241_%27You_are_an_Idiot%27_-_How_Conversational_Agent_Communication_Patterns_Influence_Frustration_and_Harassment

Brownlee, J. (2019, August 7). *What Are Word Embeddings for Text?* Machine Learning Mastery. Retrieved 6 October 2021, from https://machinelearningmastery.com/what-are-word-embeddings/

Chacón, A., Angulo, C., & Ponsa, P. (2020). Developing Cognitive Advisor Agents for Operators in Industry 4.0. *New Trends in the Use of Artificial Intelligence for the Industry 4.0*. Published. https://doi.org/10.5772/intechopen.90211

del Fierro, K. (2021, September 21). *Chatbots, Virtual Assistants & Conversational AI*. Aisera - AI Service Management (AISM). Retrieved 27 September 2021, from https://aisera.com/chatbots-virtual-assistants-conversational-ai

Epicor. (n.d.). *What is Industry 4.0—the Industrial Internet of Things (IIoT)?* Epicor Software Corporation. Retrieved 10 October 2021, from https://www.epicor.com/en/resource-center/articles/what-is-industry-4-0/

European Commision. (2020, October 2). *Artificial intelligence (AI)*. European Commission - European Commission. Retrieved 10 October 2021, from https://ec.europa.eu/info/research-and-innovation/research-area/industrial-research-and-innovation/key-enabling-technologies/artificial-intelligence-ai_en

Generating NLU Data. (2021, July 12). Rasa Open Source Documentation. Retrieved 29 September 2021, from https://rasa.com/docs/rasa/generating-nlu-data/

Hadorn, B., Courant, M., & Hirsbrunner, B. (2016). Towards Human-Centered Cyber-Physical Systems: A Modeling Approach. *ResearchGate*. Published. https://doi.org/10.13140/RG.2.2.30788.01928

Hall, B. R. (2019, April 2). *What Is the Riskiest Assumption Test and Why Are Startups Embracing It?* Clutch.Co. Retrieved 4 October 2021, from https://clutch.co/app-developers/resources/what-is-riskiest-assumption-test

IBM. (2021, August 17). *Conversational AI*. Retrieved 22 September 2021, from https://www.ibm.com/cloud/learn/conversational-ai#toc-conversati-0zW26VEB

Interactions. (2021, April 9). *Conversational AI | What is Conversational AI?* Retrieved 11 October 2021, from https://www.interactions.com/conversational-ai/

Kernan Freire, S. (2020). *A conversational ai agent for industrial maintenance reporting* (master thesis). Delft University of Technology. https://repository.tudelft.nl/islandora/object/uuid:9012abed-2736-4e7c-85e2-7fff8a309165?collection=education

Laranjo, L., Dunn, A. G., Tong, H. L., Kocaballi, A. B., Chen, J., Bashir, R., Surian, D., Gallego, B., Magrabi, F., Lau, A. Y. S., & Coiera, E. (2018). Conversational agents in healthcare: a systematic review. *Journal of the American Medical Informatics Association*, *25*(9), 1248–1258. https://doi.org/10.1093/jamia/ocy072

Parayno, R. (2020, May 1). *Conversational assistants in the time of Coronavirus*. Medium. Retrieved 22 September 2021, from https://uxdesign.cc/conversational-assistants-in-the-time-of-coronavirus-3d5cd1d5bba

Ponnusamy, P., Roshan Ghias, A., Guo, C., & Sarikaya, R. (2020). Feedback-Based Self-Learning in Large-Scale Conversational AI Agents. *Proceedings of the AAAI Conference on Artificial Intelligence*, *34*(08), 13180–13187. https://doi.org/10.1609/aaai.v34i08.7022

Ricadat, P. (2018, November 15). *Our experience building chatbots with Rasa — Tuning the NLU pipeline*. Medium. Retrieved 6 October 2021, from https://chatbotslife.com/our-experience-building-chatbots-with-rasa-tuning-the-nlu-pipeline-74a80cd565b8

Riskiest Assumption Test: Why You Need It And When RAT Precedes An MVP. (2020, February 20). Cadabra Studio. Retrieved 4 October 2021, from https://cadabra.studio/blog/riskiest-assumption-test

Shah, P., Hakkani-Tür, D., Tür, G., Rastogi, A., Bapna, A., Nayak, N., & Heck, L. (2018). Building a Conversational Agent Overnight with Dialogue Self-Play. *ArXiv*. Published. https://arxiv.org/abs/1801.04871

Smallcombe, M. (2020, June 17). *Structured vs Unstructured Data: 5 Key Differences*. Xplenty. Retrieved 3 November 2021, from https://www.xplenty.com/blog/structured-vs-unstructured-data-key-differences/

Surendranadha Panicker, S. (2021). *Ethical task tracking of operators in agile manufacturing* (master thesis). Delft University of Technology. http://resolver.tudelft.nl/uuid:3408e8c3-809b-436d-94eb-efb4f0532b17

Techopedia. (2021, June 3). *Tokenization*. Techopedia.Com. Retrieved 6 October 2021, from https://www.techopedia.com/definition/13698/tokenization

Torn, I., & Vaneker, T. (2019). Mass Personalization with Industry 4.0 by SMEs: a concept for collaborative networks. *Procedia Manufacturing*, *28*, 135–141. https://doi.org/10.1016/j.promfg.2018.12.022

van den Bosch, K., Schoonderwoerd, T., Blankendaal, R., & Neerincx, M. (2019). Six Challenges for Human-AI Co-learning. *Adaptive Instructional Systems*, 572–589. https://doi.org/10.1007/978-3-030-22341-0_45

White, K. (2021, August 19). *Unpacking the TED Policy in Rasa Open Source*. Rasa. Retrieved 10 October 2021, from https://rasa.com/blog/unpacking-the-ted-policy-in-rasa-open-source/

Wochinger, T. (2021, October 5). *Rasa NLU in Depth: Intent Classification*. Rasa. Retrieved 7 October 2021, from https://rasa.com/blog/rasa-nlu-in-depth-part-1-intent-classification/

Zhou, J., Zhou, Y., Wang, B., & Zang, J. (2019). Human–Cyber–Physical Systems (HCPSs) in the Context of New-Generation Intelligent Manufacturing. *Engineering*, *5*(4), 624–636. https://doi.org/10.1016/j.eng.2019.07.015

Appendix A1 Annotation examples for data collection

The method used to collect the dialogues was a researcher roleplaying a COALA Cognitive Advisor. I would play the role of the Cognitive Advisor while the operators would continue their operating work but also wear a mono headset which recorded audio. During issue handling, I asked the operators questions related to the issue and recorded these dialogs. Entities were used to check whether a full descriptive issue was given, these entities were: Subject, object, type of issue, symptom, location, current task, SKU, RCA and solution. Also, the timestamp, 2/3-word description of the issue, SKU and batch number were noted. During the roleplay, pen and paper were used to mark whether the operator mentioned entities or whether entities were missed during the dialogues. I would try to ask for entities which were not mentioned, while trying not to disrupt their work. The recorded audio was to be transcribed and this transcribed data was to be structured in an Excel sheet for annotation.

Due to time constraints a selection of dialogues for two machines were selected for annotation out of the 101 recordings: The filler and the cap supply machine. These two machines were chosen specifically because one machine (the filler) had the most issues, while the other machine (cap supply machine) had many similar issues (issues regarding the caps getting stuck while being transported to the filler to be placed on the liquid-filled canisters). The idea was that these similar issues which were described several times could give us insights into how the operators describe issues differently. The dialogues were manually transcribed and annotated by two researchers to maximise accuracy and consistency.

This resulted into 29 annotated dialogues of issue descriptions of which 16 were for the filler and 13 about the cap supply machine.

Below, two examples of annotated dialogs with dialog annotation tables are provided. "O:" indicates the operator is speaking and "R:" indicates the researcher is speaking. These dialogs follow natural speaking patterns and as will be highlighted, show some complications.

The dialog in figure C1.2 is an instance where without context, it is difficult to follow. For instance, to what does the operator refer to with "*he*" in the sentence "*he is going to take it so to speak*." From context we know the operator refers to the palletizer arm which would grab the cardboard pallet plates and place them correctly on the pallet. This missing information from the dialogues could provide difficulties for COALA to handle and should be considered in future developments.

Example dialog 1
R: What's going on here? O: Yes, you have pallet plates here. He always has to use the first uhhh... On the pallet he places a pallet plate and at the last layer he places a pallet plate. It is probably a bit too full now, so that it no longer takes pallet plates. So I'm going to get rid of a few first. If it immediately starts to have problems with loading, uhh... it will slide. Let's take a look at what happens from the other side. R: *inaudible* O: Here you can see, here he is going to take it so to speak. He hasn't caught yet, see? R: Yes. O: It is also possible that the plates are a little too convex. Look, he only takes half. R: *inaudible*

O: Yes, so the plates are too convex anyway. So I have to turn them all around. R: *inaudible* O: Yes, if it is too bent it goes down I think, then it won't catch it well. Yes, it's just a sensor so to speak. And it just grabs the plate at a certain height and then it can't see the material anymore and if it is convex, then it might just grab below it. R: *inaudible* O: Yes, it is placed on the bottom one (layer) and the last one (layer). He does 2 pallet plates per pallet. Just for stability.

Figure C1.1: Example of a dialog of an issue at a palletiser

Annotation entities	Definition
Symptom-component(s)	He [pallet arm]
Symptom	1. It no longer takes pallet plates. 2. He hasn't caught yet, see? 3. Look, he only takes half.
Cause	1. It is probably a bit too full now. 2. The plates are a little too convex. 3. So the plates are too convex anyway.
Solution	1. I'm going to get rid of a few first. 2.So I have to turn them all around.
Location	-
Solution_component	-
Cause_component	-
Related_part_product	Pallet plates
RCA	Yes, if it [...] grab below it.
Intended behaviour	1. He is going to take it so to speak. 2. It is placed... Just for stability.

Table C1.1: Entities for annotating the issue description in example dialog 1

Compared to example one, information in example two is more concise and easier to follow. Yet, in this dialog several symptoms and causes are given and COALA should be able to distinguish between them.

Example dialog 2
R: What is happening? O: The cap turner has now malfunctioned. I think he got overloaded. Yes overloaded... Overloaded. Cap turner 1 overloaded. R: What is the solution to that? O: Solution uhh... Reset to zero point and reset. At some point it ends up in a magnetic disk, if it falls outside of it, then it will say "hey, something is not right here" and it will say it is overloaded. Then it's just a matter of giving a <u>little tap and then it's back in</u>.

Figure C1.2: Example of a dialog about the cap turner

Annotation entities	Definition
Symptom-component(s)	The cap turner 1
Symptom	1. The cap turner has now malfunctioned. 2. Then it will say “hey, something is not right here” and it will say it is overloaded
Cause	1. Cap turner 1 is overloaded. 2. At some point it ends up in a magnetic disk, if it falls outside of it.
Solution	1. Reset to zero point and reset 2. Then it’s just a matter of giving a little tap and then it’s back in.
Location	-
Solution_component	[cap turner]
Cause_component	-
Related_part_product	Cap
RCA	-
Intended behaviour	-

Table C1.2: Entities for annotating the issue description in example dialog 2

As mentioned in the examples, this annotation process provided insights into how operators naturally describe issues and the potential challenges and opportunities this poses for NLP.

Appendix A2 Consent forms data collection

Below the consent forms in English and in Dutch.

Consent form

Boris Hadžisejdić
Tudelft
b.hadzisejdic@student.tudelft.nl

Barnabass Kiss
Diversey BV.
Barnabas.kiss@diversey.com

The Technical University of Delft supports the practice of protecting research participants' rights. Accordingly, this project was reviewed and approved by the Human Research and Ethics Committee. The information in this consent form is provided so that you can decide whether you wish to participate in our study. It is important that you understand that your participation is considered voluntary. This means that even if you agree to participate you are free to withdraw from the data collection at any time, without consequences and without a need to give a valid reason. If you wish to withdraw your consent after the experiment is completed, we ask that you try to do so within 24 hours, by contacting the researchers.

The aim of this study is to collect data on the issues operators face on the production line, how they are diagnosed and solved. Additionally, we will be testing the performance of automatic audio transcriptions, a core component of the digital assistant we are developing for Diversey; collect dialogues for training our digital assistant to talk to operators; and informally access the user experience of wearing a wireless headset during work and reporting on issues by voice. **The recorded audio will be transcribed within 5 months and then deleted. The resulting transcriptions will be anonymized and no personal information will be stored about you. The collected data will NOT be used for evaluating operators in terms of their performance nor for process improvement in the company.** The collected data will only be used by members of this research project from TU Delft and will not be shared with others without your explicit permission. No videos, images, or other data that would enable anyone to identify you will be stored. The researchers will adhere to Diversey's COVID measures. Furthermore, the wireless headset will be cleaned before use. This consent form will be carefully and securely stored for at most five years (until March 31, 2026). During the study, you are asked to work as usual and respond to our questions about production line issues and your experience with the study.

If you have any questions not addressed by this consent form, please do not hesitate to ask.

Declaration of consent (please tick the appropriate boxes)

	YES	NO
1. I agree to participate in this study	◯	◯
2. I have read the study information above and understand that my participation is voluntary and that I am free to withdraw at any time, without giving any reason.	◯	◯
3. I understand that an audio-recording will be made and stored for the duration of the associated masters graduation project (5 months).	◯	◯
4. I agree for my non-identifiable data to be made available in an anonymized dataset.	◯	◯

____________________ ____________________ ________

Name participant — Signature participant — Date

____________________ ____________________ ________

Name researcher — Signature researcher — Date

Consent form

Boris Hadžisejdić
Tudelft
b.hadzisejdic@student.tudelft.nl

Barnabass Kiss
Diversey BV.
Barnabas.kiss@diversey.com

TU Delft ondersteunt de praktijk om de rechten van onderzoeks- deelnemers te beschermen. Overeenkomstig is dit project beoordeeld en goedgekeurd door de Human Research and Ethics Committee. De informatie in dit toestemmingsformulier wordt verstrekt zodat u kunt beslissen of u wilt deelnemen aan ons onderzoek. Het is belangrijk dat u begrijpt dat uw deelname als vrijwillig wordt beschouwd. Dit betekent dat, zelfs als u ermee instemt om deel te nemen, u zich op elk moment kunt terugtrekken uit de gegevensverzameling, zonder consequenties en zonder dat u een geldige reden hoeft op te geven. Als u uw toestemming na afloop van het experiment wilt intrekken, vragen we u dit binnen 24 uur te proberen door contact op te nemen met de onderzoekers.

Het doel van deze studie is om gegevens te verzamelen over de problemen waarmee operators op de productielijn worden geconfronteerd, hoe ze worden gediagnosticeerd en opgelost. Daarnaast zullen we de prestaties testen van automatische audiotranscripties, een kerncomponent van de digitale assistent die we ontwikkelen voor Diversey; dialogen verzamelen om onze digitale assistent te trainen om met operators te praten; en informeel de gebruikerservaring verzamelen van het dragen van een draadloze headset tijdens werk en het melden van problemen via spraak. **De opgenomen audio wordt binnen 5 maanden getranscribeerd en vervolgens verwijderd. De resulterende transcripties worden geanonimiseerd en er wordt geen persoonlijke informatie over u opgeslagen. De verzamelde gegevens zullen NIET worden gebruikt voor het evalueren van operators in termen van hun prestaties, noch voor procesverbetering in het bedrijf.** De verzamelde gegevens worden alleen gebruikt door leden van dit onderzoeksproject van de TU Delft en worden niet gedeeld met anderen zonder jouw uitdrukkelijke toestemming. Er worden geen video's, afbeeldingen of andere gegevens opgeslagen waarmee iemand u kan identificeren. De onderzoekers zullen zich houden aan de COVID-maatregelen van Diversey. Verder wordt de draadloze headset voor gebruik gereinigd.
Dit toestemmingsformulier wordt maximaal vijf jaar (tot 31 maart 2026) zorgvuldig en veilig bewaard. Tijdens de studie wordt u gevraagd om te werken zoals u gewend bent en te reageren op onze vragen over productielijnkwesties en uw ervaring met de studie.
Als u vragen heeft die niet in dit toestemmingsformulier staan, aarzel dan niet om ze te stellen.

Toestemmingsverklaring (kruis de betreffende vakjes aan)

	YES	NO
1. Ik ga ermee akkoord om deel te nemen aan deze studie.	○	○
2. Ik heb bovenstaande informatie gelezen en begrijp dat mijn deelname vrijwillig is en dat ik me op elk moment kan terugtrekken, zonder opgave van reden.	○	○
3. Ik begrijp dat er een audio-opname wordt gemaakt en bewaard voor de duur van het bijbehorende master afstudeerproject (5 maanden).	○	○
4. Ik ga ermee akkoord dat mijn niet-identificeerbare gegevens beschikbaar worden gesteld in een geanonimiseerde dataset.	○	○

____________________ ____________________ __________

Naam participant — Handtekening participant — Datum

____________________ ____________________ __________

Naam onderzoeker — Handtekening onderzoeker — Datum

Appendix B1 Conversational Flow Test

For the Diversey use case a conversation flow test was conducted on the 16th of September at the Enschede factory.

COALA is intended to be an intelligent conversational AI that can collect data on issues, root causes and solutions. Furthermore, it can give recommendations for possible solutions or, when needed, contact technical services to help on the operating line. To improve COALA, a chatbot was created with a tool called Rasa. The current version of the chatbot prototype responds to input through a set of follow-up questions to capture data on issue descriptions and store it in a data sheet. In the ideal scenario, the Rasa bot stores the following entities: machine location, product component, product component state. These entities are used to describe the issue on the production line. With Rasa we can train the natural language understanding (NLU) core of COALA (basically COALA's brain) and improve how good the conversational AI is in collecting correct data and therefore also improve the recommendations it can provide. However, to train the core, we need to test it with domain specific users. This is the reason of the visit. We need user input to improve upon the NLU core and user insights into what we (maybe) have not considered to implement.

B1.1 Goal

The purpose of the conversation flow test is threefold:

- Identify whether unknown flaws are present in the dialogs for collecting issue descriptions
- Test whether identified challenges gathered in the data collection described in appendix C1 will arise
- Gather user insights and feedback on the dialogs and the dialog handling

B1.2 Approach

The approach was to use Rasa X to collect conversations with participants who have knowledge of the domain: operators, operator team leads, technical service and quality control personnel. Rasa X allows for the conversations to be stored either on a server or locally, and for the conversations to be post-processed. The Rasa prototype will use a form, a method for the chatbot to remember what has been said/written (fill slots of the form), to capture data on issue descriptions and store it in a data sheet. The slots will be filled with the following entities: machine location, product component, product component state.

To gather new findings from the conversations, attempts were iteratively made in between testing to improve the prototype.

The results of the test were evaluated in Rasa X in between testing with participants (post-processing) to identify errors in the dialogs and to improve the handling of the dialogs by, for instance, adding features to the prototype to prevent issues happening in the conversations. Rasa X allows for analysing individual conversations between chatbot and user. During post-processing, the accuracy for intent and context recognition of the prototype was manually processed.

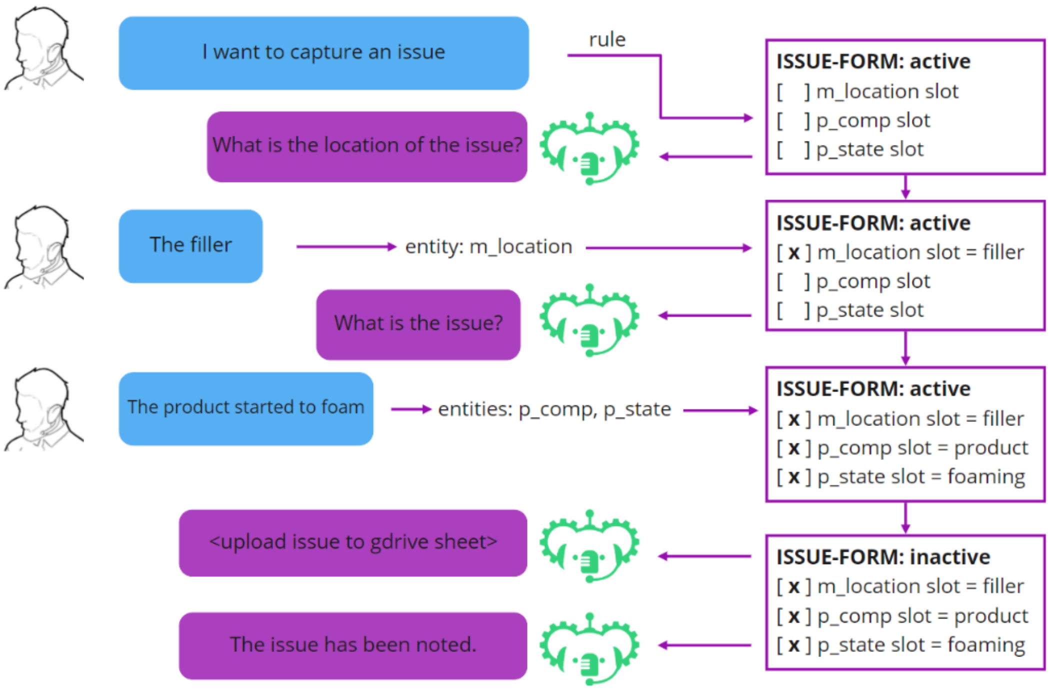


Figure B1.1: A happy path visualised for capturing an issue description with the issue form.

Figure B1.1 shows a happy path for the conversation flow test where the issue description is captured with an issue form. Note, the utterances from the chatbot are reliant on what slots have been filled. For instance, when slot m_location has been filled, the requested slot to be set is p_comp and the related chatbot utterance to this entity is: "What is the issue?". Rasa allows for several slots to be set simultaneously.

B1.3 Setup

Two researchers were involved in the test. One researcher did the introduction and assisted the participants when they got stuck with the testing, while the other observed the participants and made notes. The participants were tasked to talk out loud and communicate what they were doing and their thought process during the actions. Noteworthy actions and thought processes were captured by the observing researcher, next to the feedback provided by the participants in the discussions at the end of the testing. Furthermore, the sentiment towards the prototype was also observed.

A laptop was used to access the prototype and allow to participants to communicate with it by typing.

A scenario was created which the participants had to capture with the Rasa chatbot. The participants had to act as production line operators on the 5/10L line and had to describe the issue in figure B1.2. Images were used to illustrate the issue and mimic a real life situation the operators could experience at the operating line while having to describe it to a conversational assistant. Unfortunately, operators were not available for testing, see section B1.3.1 on the limitations of the setup and why the participants had to 'act' as operators. For the first attempt, the participants were not explained how to capture the issue, this is to test the goals set for the test. By not explaining the approach of capturing an issue, the participants are also not influenced in how they would capture it. This way unidentified flaws can be found. Also, it can be tested whether previous challenges have been resolved (section 6.4 and appendix C1). After a first attempt, the researchers explained the intended way of capturing this specific issue to the participants. Afterwards they were allowed to test it again, but most importantly the feedback from the participants was captured. Due to the short time the participants had, the second test with knowledge on the intended way was only done by one out of all the participants. The feedback was either captured with unstructured interviews or with a form as guidance, see Appendix B2 for the full test setup in Dutch. Reasoning behind this approach is to allow for open discussion on the prototype and leave room for the participants to give feedback. If this process got off to a slow start, the form was used to guide the conversation between participant and researcher on the prototype.

After the tests, the following was captured (see table B1.2):

- the amount of user utterances to capture the issue and fill the issue form
- whether the participants managed to finish the issue form without input from the researcher and whether they were able to do that with input from the researcher
- the number of correct intents that were recognized out of the total amount of intents in the conversation
- the number of correct entities recognized out of the total amount of entities in the conversation
- interesting findings taken directly from the conversation
- adjustments that were applied after the participant finished their conversation with the prototype.

These values provide information on the useability of the prototype and were used to validate it.

Figure B1.2: An issue to be described by the participants; Foaming of the product (liquid) at the filler.

B1.3.1 Limitations

Logically, operators were desired as user testers for the prototype as the prototype is mainly meant to be used by operators. However they were unavailable due circumstances with operations at Diversey. Still, participants with some knowledge on the production line and the specific terminology were needed. Therefore, staff from technical services and quality management were asked to participate. As earlier mentioned, they were tasked to act as operators. However, at the time of testing the availability of the Diversey staff members was limited, and participants were recruited one-by-one when available. Six participants from technical services and one from quality management participated in the test.

B1.4 Results

The results are 8 conversations with 7 participants, see Appendix B3 for the full conversations and Appendix B4 for the observations during conversation and participants' feedback. Table B1.1 gives an overview of the results of the conversations.

Table B1.1: Results of the conversations, including: the amount of user utterances*, finished issue form**, correct intent recognition, correct entity recognition, interesting findings and adjustments applied to chatbot.
**** = amount of user utterances to finish issue form***
***** = first answer indicates the participant was able to finish the task without help from researcher, second answer indicates the participant was able to finish the task with the help from the researcher.***
****** = The numbers between brackets (i.e. (0.77)) indicates the certainty of the chatbot of what the intent is.***
******* = adjustment in between brackets ([…]) are future adjustments which should be considered for implementation. Adjustments are applied after the conversation.***

Chat #	Amount of user utterance*	Finished issue form**	Correct intent recognition	Correct entity recognition	Interesting finding(s)***	Adjustments****
1	4	No/Yes	4/5, 80%	3/4, 75%	“loopt” was recognized as p_comp_state while “loopt leeg” should be recognized.	-
2A	5	No/No	4/5, 80%	1/3, 33%	“Vulmachine” not recognized as synonym of “vuller”. “Niet in verpakking afgevuld” recognized as deny intent (1.00).	For test: Added “Vulmachine” as entity in mention_issue intent (the reason why synonym was not found)
2B	4	Yes	4/4, 100%	3/3, 100%	Compared to chat 1, “loopt eruit” was correctly recognized as p_comp_state	-
3	7	No/Yes	7/7, 100%	5/6, 83%	p_comp_state slot has been overwritten twice (3 instances with p_comp_state)	-
4	8	No/Yes	6/7, 86%	3/6, 50%	“Geen” recognized as p_comp_state (0.77). “Vult boven gewicht” recognized as deny intent (0.89). Sleever not recognized as entity at all, should be recognized as m_location. Participant had issues with finding p_comp_state.	[Sleever not recognized as an entity indicates that the lookup table is not working as intended.] Added several intent examples of mention_symptom intent with new p_comp_state entities taken from test.
5	5	No/Yes	4/5, 80%	5/6, 83%	“Afvulmachine” not recognized as m_location entity. Participant had issues with finding p_comp_state.	Added “afvulmachine” and “afvul machine” as entity in mention_issue intent. Swapped order of slots p_comp and p_comp_state. p_comp_state now connect to the question “Wat is er gebeurd?” and p_comp to “Benoem het product alsjeblieft.”
6	5	Yes	5/5, 100%	4/4, 100%	“Het schuimt” is used to describe the issue, p_comp missing. Filled in afterwards after correct question. “Het” correctly recognized as pronoun entity, no execution of pronoun rule however.	-
7	8	No/Yes	6/8, 75%	3/5, 60%	“Le” incorrectly recognized as affirm intent (0.88). “Waarc” correctly	[Consider adding “zeep” as an entity for p_comp]

					recognized as affirm intent, although the prototype was not looking for an affirm intent. Afterwards, it deactivated the issue_form loop. "Overvullen" recognized as m_location entity (0.61). "Zeep" not recognized as p_comp entity.	[A story of an unhappy path is present in the model where the user can stop the issue_form with a stop intent and a affirm intent after an utter_stop response. As the chatbot recognized "Waarc" as an affirm intent, it used this story to exit the issue_form active loop. Consider whether changes need to be applied.]
Total			87.6%	73%		

B1.5 Findings

Aside from the findings highlighted in table 1 under 'Interesting finding(s)', some noteworthy findings are highlighted below.

- For participants 6 and 7 the issue form was adjusted after the conversation with participant 5. The slots for p_comp and p_comp_state were swapped in order and the final chatbot utterance had to be adjusted to target the p_comp entity. So, for the utterance "What happened?" (in Dutch: "Wat is er gebeurd?") the requested slot was swapped from p_comp to p_comp_state. The utterance for the final slot was changed from "What is the status of the product?" (in Dutch: "Wat is de status van het product?") to "Could you name the product?" (in Dutch: "Zou u het product kunnen benoemen?"). The results from participants 6 and 7 are indicating that the changes applied after participant 5 were fruitful. Unfortunately, due to no other available participants, the test was cut short, and the changes could not be tested further.
- Out of the 7 participants, only one managed to fill the issue form without input from the researcher: participant 6. This might be due to the iterative changes applied during the test.
- Two main challenges from data collection arose during the test: synonym handling and pronoun usage.
- Overall, the participants would like to be "guided" more in the conversations. One participants suggested more specific questions related to the input from the user.

B1.6 Proposed changes

Below, some proposed changes are listed to be implemented to the chatbot:

- Training examples of synonyms as entities should be added, only using a synonym list does not work. The same issue applies for lookup tables.
- Use entity AND intent recognition for product component state and use only entity recognition for product component. Only using entity recognition for product component state is too strict and will create frustrations with the user because the chatbot will start looping with the same questions.
- Consider only adding machine component to issue description, machine component state might be unnecessary

- Consider adding story for issue form without stop intent but with affirm intent and let the bot continue with the issue form instead of what happened during conversation 7.
- Swap the order of the slots: p_comp_state before p_comp
- Wat is er gebeurd? (What happened?) → p_comp_state
- Kan je het product benoemen? (Can you name the product?) → p_comp
- For filling the issue form, a final confirmation will be needed to let the operators double check if the bot has captured the issues correctly.

Appendix B2: Test description (Dutch)

Chatbot test 16-9-2021

Deze test is bedoeld om het prototype te testen. Er wordt specifiek niet getest op uw kunnen, maar of de prototype werkt. Het prototype is onvolledig en heeft uw input nodig om verbeterd te worden.

Eerst laten we je het zelf testen, daarna leggen we uit hoe het prototype moet werken en laten we je het nog een keer testen. En tot slot willen we graag uw input over het prototype en samen nadenken over hoe we het kunnen verbeteren.

De test zal als volgt gaan:

- We laten u als eerst een bepaalde scenario uittesten met het prototype. Ik vraag u om hardop na te denken.
- Daarna zal ik u uitleggen hoe het prototype hoort te werken. Indien u het wilt, kunt u het opnieuw uitproberen.
- Daarna zullen we met behulp van de hele korte invul enquête onderaan de pagina feedback vragen over het prototype. Als u nog goeie ideeën en inzichten heeft, hoor ik die graag.

Scenario:

U bent een operator en met behulp van de chatbot wilt u het probleem in de afbeeldingen vastleggen.

U kunt beginnen door de chatbot te groeten.


U kunt aan het eind met de lijst hieronder het prototype beoordelen:

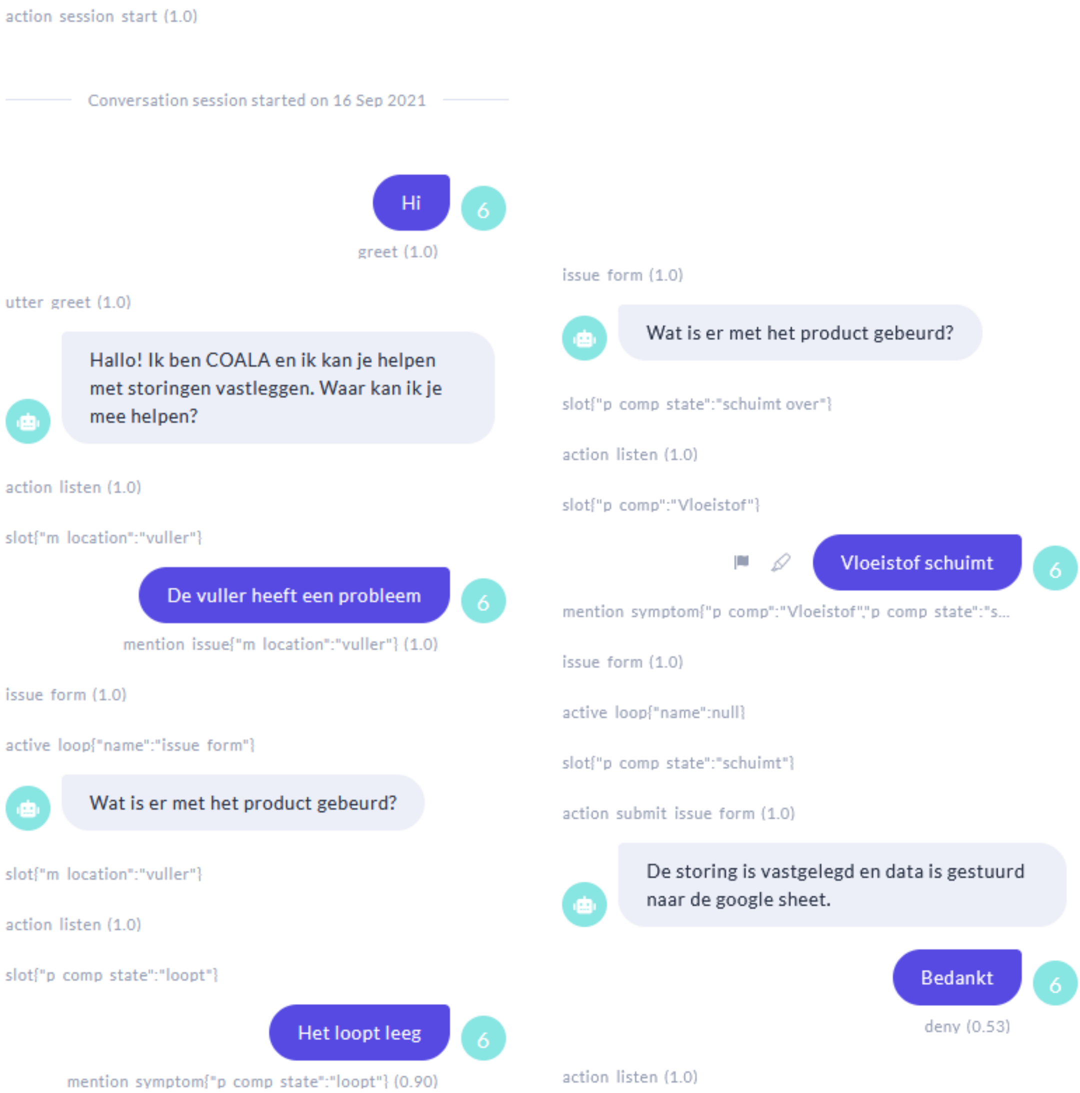
belemmerend	o o o o o o o	ondersteunend
complex	o o o o o o o	eenvoudig
inefficiënt	o o o o o o o	efficiënt
verwarrend	o o o o o o o	overzichtelijk
vervelend	o o o o o o o	spannend
oninteressant	o o o o o o o	interessant
conventioneel	o o o o o o o	origineel
gebruikelijk	o o o o o o o	nieuw

PS: Het prototype representeert niet het eindproduct.

Appendix B3: Full conversations with Rasa X for conversational flow test

Below are the full conversations of the conversational flow test.

Participant #1



Participant #2

action session start (1.0)

Conversation session started on 16 Sep 2021

utter greet (1.0)

Hi! Ik ben COALA en ik kan je helpen met storingen vastleggen. Waar kan ik je mee helpen?

Hoi

greet (1.0)

slot{"p comp":"product"}

Er ligt product in de vulmachine

mention issue{"p comp":"product"} (1.0)

issue form (1.0)

active loop{"name":"issue form"}

Wat is de locatie van het probleem?

Vuller | Doppendraaier | Sleever

slot{"p comp":"product"}

action listen (1.0)

Vulmachine

mention issue (1.0)

action execution rejected (1.0)

action default fallback (0.30)

rewind

action listen (1.0)

slot{"m location":"vuller"}

vuller

mention issue{"m location":"vuller"} (1.0)

issue form (1.0)

Wat is de status van het product?

slot{"m location":"vuller"}

action listen (1.0)

Niet in verpakking afgevuld

deny (1.0)

action default fallback (0.30)

action execution rejected (1.0)

rewind

action listen (1.0)

/restart

restart (1.0)

action restart (1.0)

Restart

Conversation session started on 16 Sep 2021

utter greet (1.0)

Hi! Ik ben COALA en ik kan je helpen met storingen vastleggen. Waar kan ik je mee helpen?

Hoi

greet (1.0)

action listen (1.0)

Er is een storing

mention issue (1.0)

issue form (1.0)

active loop{"name":"issue form"}

Wat is de locatie van het probleem?

Vuller | Doppendraaier | Sleever

slot{"requested slot":"m location"}

slot{"m location":"vuller"}

/mention_issue{"m_location":"vuller"}

mention issue{"m location":"vuller"} (1.0)

issue form (1.0)

Wat is er met het product gebeurd?

Participant #3

action session start (1.0)

Conversation session started on 16 Sep 2021

Hoi

8

greet (1.0)

utter greet (1.0)

Hi! Ik ben COALA en ik kan je helpen met storingen vastleggen. Waar kan ik je mee helpen?

action listen (1.0)

slot{"p comp state":"misslag"}

Ik heb een misslag gehad

8

mention issue{"p comp state":"misslag"} (0.97)

issue form (1.0)

active loop{"name":"issue form"}

Wat is de locatie van het probleem?

Vuller | Doppendraaier | Sleever

slot{"p comp state":"misslag"}

action listen (1.0)

/mention_issue{"m_location":"vuller"}

8

mention issue{"m location":"vuller"} (1.0)

slot{"m location":"vuller"}

issue form (1.0)

Wat is er met het product gebeurd?

slot{"m location":"vuller"}

action listen (1.0)

issue form (1.0)

Wat is er met het product gebeurd?

slot{"m location":"vuller"}

vuller

8

mention issue{"m location":"vuller"} (1.0)

action listen (1.0)

issue form (1.0)

Wat is er met het product gebeurd?

slot{"p comp state":"Overstroomd"}

Overstroomd

8

mention symptom{"p comp state":"Overstroomd"} (1.0)

action listen (1.0)

loopt eruit

8

mention symptom{"p comp state":"loopt eruit"} (1.0)

slot{"p comp state":"loopt eruit"}

Participant #4

action session start (1.0)

Conversation session started on 16 Sep 2021

Hi

greet (1.0)

utter greet (1.0)

Hallo! Ik ben COALA en ik kan je helpen met storingen vastleggen. Waar kan ik je mee helpen?

action listen (1.0)

issue form (1.0)

active loop{"name":"issue form"}

Wat is de locatie van het probleem?

Vuller | Doppendraaier | Sleever

slot{"requested slot":"m location"}

probleem met de afvullen van 5literlijn

mention issue (1.0)

action listen (1.0)

Sleever

mention issue (0.80)

action default fallback (0.30)

action execution rejected (1.0)

rewind

action listen (1.0)

sleever

mention issue (0.80)

action default fallback (0.30)

action execution rejected (1.0)

rewind

action listen (1.0)

slot{"m location":"vuller"}

/mention_issue{"m_location":"vuller"}

mention issue{"m location":"vuller"} (1.0)

issue form (1.0)

slot{"m location":"vuller"}

Wat is er met het product gebeurd?

slot{"requested slot":"p comp"}

action listen (1.0)

slot{"p comp state":"Geen"}

Geen sleeve om de can

mention symptom{"p comp state":"Geen"} (0.77)

issue form (1.0)

Wat is er met het product gebeurd?

slot{"p comp state":"Geen"}

action listen (1.0)

Vult boven gewicht

deny (0.89)

action default fallback (0.30)

action execution rejected (1.0)

rewind

action listen (1.0)

slot{"p comp":"Product"}

Product schuimt

mention symptom{"p comp":"Product","p comp state":"sc...

issue form (1.0)

active loop{"name":null}

slot{"p comp state":"schuimt"}

action submit issue form (1.0)

De storing is vastgelegd en data is gestuurd naar de google sheet.

...

Participant #5

action session start (1.0)

Conversation session started on 16 Sep 2021

Hallo

1

greet (1.0)

utter greet (1.0)

Hi! Ik ben COALA en ik kan je helpen met storingen vastleggen. Waar kan ik je mee helpen?

action listen (1.0)

slot{"p comp":"product"}

Er ligt product in de afvulmachine

1

mention issue{"p comp":"product"} (0.95)

issue form (1.0)

active loop{"name":"issue form"}

Wat is de locatie van het probleem?

Vuller | Doppendraaier | Sleever

slot{"p comp":"product"}

action listen (1.0)

slot{"m location":"vuller"}

/mention_issue{"m_location":"vuller"}

1

mention issue{"m location":"vuller"} (1.0)

issue form (1.0)

Wat is de status van het product?

slot{"m location":"vuller"}

action listen (1.0)

slot{"m location":"vuller"}

Tijdens het afvullijn is er product in de vuller gekomen

1

mention issue{"p comp":"product","m location":"vuller"} (...

issue form (1.0)

Wat is de status van het product?

slot{"m location":"vuller"}

action listen (1.0)

slot{"p comp state":"Schuimt"}

Schuimt

1

mention symptom{"p comp state":"Schuimt"} (1.0)

issue form (1.0)

active loop{"name":null}

slot{"p comp state":"Schuimt"}

action submit issue form (1.0)

Participant #6

action session start (1.0)

Conversation session started on 16 Sep 2021

Hi

D

greet (1.0)

utter greet (1.0)

Hallo! Ik ben COALA en ik kan je helpen met storingen vastleggen. Waar kan ik je mee helpen?

Hallo

D

greet (1.0)

utter greet (1.0)

Hi! Ik ben COALA en ik kan je helpen met storingen vastleggen. Waar kan ik je mee helpen?

slot{"m location":"vuller"}

Probleem bij vuller

D

mention issue{"m location":"vuller"} (1.0)

issue form (1.0)

active loop{"name":"issue form"}

Wat is er gebeurd?

slot{"m location":"vuller"}

action listen (1.0)

slot{"p comp state":"schuimt"}

Het schuimt

D

mention symptom{"pronoun":"Het","p comp state":"schui...

issue form (1.0)

Benoem het product alstublieft.

slot{"p comp state":"schuimt"}

action listen (1.0)

slot{"p comp":"vloeistof"}

De vloeistof

D

mention symptom{"p comp":"vloeistof"} (1.0)

issue form (1.0)

active loop{"name":null}

slot{"p comp":"vloeistof"}

action submit issue form (1.0)

De storing is vastgelegd en data is gestuurd naar de google sheet.

...

Participant #7

action session start (1.0)

Conversation session started on 16 Sep 2021

utter greet (1.0)

Hi! Ik ben COALA en ik kan je helpen met storingen vastleggen. Waar kan ik je mee helpen?

Hallo

greet (1.0)

action listen (1.0)

issue form (0.77)

active loop{"name":"issue form"}

Wat is de locatie van het probleem?

Vuller | Doppendraaier | Sleever

slot{"requested slot":"m location"}

Le

affirm (0.88)

action listen (1.0)

slot{"m location":"vuller"}

Ik heb lekkage in de vuller

mention issue{"m location":"vuller"} (1.0)

issue form (1.0)

Wat is er gebeurd?

slot{"m location":"vuller"}

action listen (1.0)

Waarc

affirm (1.0)

action deactivate loop (0.32)

action execution rejected (1.0)

active loop{"name":null}

slot{"requested slot":null}

action listen (1.0)

slot{"m location":"overvullen"}

Waarschijnlijk overvullen

mention issue{"m location":"overvullen"} (0.61)

issue form (1.0)

active loop{"name":"issue form"}

Wat is er gebeurd?

slot{"m location":"overvullen"}

action listen (1.0)

slot{"m location":"afvullen"}

Het is overgestroomt met afvullen

mention symptom{"p comp state":"overgestroomt","m l...

issue form (1.0)

Kan je het product benoemen?

slot{"m location":"afvullen"}

action listen (1.0)

Zeep

affirm (0.80)

action execution rejected (1.0)

action deactivate loop (0.36)

active loop{"name":null}

slot{"requested slot":null}

action listen (1.0)

slot{"p comp":"Vloeistof"}

Vloeistof

mention symptom{"p comp":"Vloeistof"} (1.0)

issue form (0.51)

active loop{"name":"issue form"}

slot{"m location":"afvullen"}

action submit issue form (1.0)

De storing is vastgelegd en data is gestuurd naar de google sheet.

action listen (1.0)

Appendix B4: Observations prototype test

Here, the observations found during the conversation flow test (Appendix B1) are given.

B4.1 Testing prototype Raza

Overall most people were quite understanding about that it was a prototype and that it still needed to learn.

Participant #1

- **Greeting the bot felt weird, but after a while start with: hoi**
- **The participant thinks he needs to address the cause right away**
- **"Er ligt product in de vulmachine"**
- **Does not recognize the vulmachine but does vuller**
- **The subject of the sentence is often missing like 'er' 'het' (pronoun usage).**
- **The participant thinks in causes and is too detailed with his descriptions.**
- **"niet afgevuld in de can" → He proposes to set buttons like with the location or give example on how to react**
- **Technical people already think in solutions before just describing the problem. So they would like to solve it instead of 'wasting' time filling in what happened.**
- **They used ~maxibu? But are now using managerplus program where they use dropdown menu's to describe what happened and therefore he would like to have that as well with the Raza bot**
- **Participant assumes that not everybody's personal description of the problem is in there so he suggests a dropdown where they can choose the most applicable**
- **There are a lot of internal meanings to a word like 'product'**
- **A registration system is obligatory from the government to already log certain activities**
- **The participant is curious how it will look like and finds it intriguing but perhaps unrealistic, he says if there are too many mistakes but the service then people won't use it like ODCE**

Participant #2

- **Started with greeting again: hoi**
- **"Een misslag gehad" → Thinks immediately what went wrong**
- **The greeting of COALA helps to think about what happened instead of going to the solution/cause**
- **The bot malfunctions again at the phrasing of the action**
- **The participant would like there to be solutions as well**
- **He finds the solution to be recognizable, he sees it a lot with FAQ sites and think immediately to KPN**
- **He says that he finds it easy to use but thinks the terminology is very important, the solution could help up to a point**
- **Operators usually give too little info about what is happening and then maintenance needs to go there to get more information about the problem**
- **Mechanics think that the operators need more knowledge about the problems so they can solve it as well → He provides an example: crinkle in the label or the distance between bottles**
- **They have a step-by-step approach to go through the problem**

- **There is a lot of changing people at the moment where knowledge is lost**
- **There are a lot of variables that have influence on the machines, so they don't have a set value but a range. This means changes to values to fit the context, however, the participant describes it as a "gevoelskwestie". The operators need to know if they should try the lower or higher part of the range.**

Participant #3 (Marcel)

- The place to type is not that clear for him and he already says that typing is not his thing. So it is going slowly
- For him the buttons are not clearly clickable and finds typing very annoying so this could have helped
- "Hij vult te veel" and "vult boven gewicht" are very context induced, so complex to understand
- The bot does not quite work for him, the question he got from Rasa was also not that clear
- Participant says there are 1000's of problems which the bot needs to work with, so the questions need to be more aimed, otherwise the bot and the user will 'lose' each other
- Being guided and making the scope smaller would help, so asking questions related to the information that he already gave
- The solution is recognizable → He thinks about the Ikea bot and says that you can't get a matrass when you look for a chair.
- Operators have a problem on a specific line, then a component and then a part of the component → Peel the layers until you are at the problem
- Idea: maybe voice recognition because there are a lot of accents and dialects and by recognizing the operator the bot already knows the line
- He finds it a positive development and is looking forward towards the end result
- He does not find the solution very original but it is new and original in this context.
- "Hopelijk mag het wortels schieten" → "May it grow"
- Filling in the current tool is already a hard task and is very conventional
- There are quite some older people that find it hard to accept change
- The foundation needs to be solid like every good building
- He would like to be involved with the development of this project!

Participant #4 (Anton)

- Is from quality control
- Has difficulty starting and greeting the bot. Eventually says hello
- Notices right away that you can click the buttons, clicks 'vuller'
- 'product status' is hard to guess what the bot expects and needs as input, it is too broad and this causes the participant to overexplain himself. It asks more than it wants to receive 'p_comp'
- If the questions can be more detailed, then he would understand it better
- The bot itself went quite smoothly
- He suggests that the voice would help a lot instead of typing

- Operators type too fast so what they type is quite cryptic so it is not clear for the people in control what they mean when they read it later on the day
- He says that there are still a lot of challenges along the way before it can really be used

Participant #5

- **Warehouse employee**
- **OCDE is not being filled in properly**
- **Is doubting how to greet COALA**
- **The first question is going well even though he uses a lot of speech related words like 'probably' and 'I'.**
- **Second question and second try goes well without interfering**
- **We changed the questions in between and this helped. Only Rasa did not recognize the product name 'zeep'. He wanted to write very specialized wordings like 'hypo' and 'acid' or even specific liquids**
- **Easing into it is important and knowing what you 'can' say → What is being recognized by Raza**
- **Operators see the tool ODC now as extra work and don't do it when there is a malfunction of the software or a easy fix. The software also only allows all or nothing approach; you can't skip one stoppage report**
- **Participant think there needs to be a re-education in order to work**

Because this was not his context, we asked what he searches for in a tool if it was introduced in the warehouse:

- **Ease of use so less steps and integration with the systems we already have**
    - **He is aware of the fact that the more steps you need to take as human, the more mistakes can slip in the process**

Appendix C: Prototype code

For the code, you can send an email to borishadzisejdic@gmail.com.

Appendix D1: Prototype feature test

A qualitative test was conducted to test the prototype features implemented in sprint 3 (chapter 8) of this project. The test was conducted with two design students who have some knowledge on the production line at Diversey.

D1.1 Goal

The test was conducted to validate the features qualitatively and gather insights on what benefits are of the features.

D1.2 Setup

Two versions of the CA model were used to test: Version A and B.
Version A had synonym handling, pronoun handling and the check, but no buttons and no contextual filtering.
Version B had synonym handling, pronoun handling, check and contextual filtering and the buttons were included.

The participants were provided with the test description (appendix D2), and tasked to capture the two issues shown in the 4 images in the description. An ngrok server was created to share the Rasa model through Rasa X to the participants. Unknowingly, they had to capture both issues twice. Once with version A and once with version B. Participant 1 started with A and then tested with B, participant 2 did it vice versa.

D1.3 Result

Appendix D3 has the full conversations of the tests and appendix D4 has the observations and feedback of the test.

D1.4 Findings

- The two most predominant observations were that version A did not provide enough feedback during capturing while version B did through the buttons. However, both participants observed that in version B they had a slight bias towards the proposed issues presented in the buttons of contextual filtering. They would try to describe the issue with one of the buttons that would match the most with what they observed.
- Another observation both had was that they could imagine that such a task would become repetitive very quickly.
- Capturing the issue with the buttons was really fast and was therefore appreciated. However the previously mentioned bias towards the buttons did still bother both participants.

Appendix D2: Prototype feature test description (Dutch)

Prototype test

Deze test is bedoeld om het prototype te testen. Er wordt specifiek niet getest op uw kunnen, maar of de prototype werkt. Het prototype is onvolledig en heeft uw input nodig om verbeterd te worden.

Eerst laten we je het zelf testen, daarna leggen we uit hoe het prototype moet werken en laten we je het nog een keer testen. En tot slot willen we graag uw input over het prototype en samen nadenken over hoe we het kunnen verbeteren.

De test zal als volgt gaan:

- We laten u als eerst een bepaalde scenario uittesten met het prototype. Ik vraag u om hardop na te denken.
- Daarna zal ik u uitleggen hoe het prototype hoort te werken. Indien u het wilt, kunt u het opnieuw uitproberen.
- Daarna zullen we met behulp van de hele korte invul enquête onderaan de pagina feedback vragen over het prototype. Als u nog goeie ideeën en inzichten heeft, hoor ik die graag.

Scenario:

U bent een operator en met behulp van de chatbot wilt u de 2 problemen in de afbeeldingen vastleggen.

U kunt beginnen door de chatbot te groeten.

Probleem 1:

Probleem 2:

U kunt aan het eind met de lijst hieronder het prototype beoordelen:

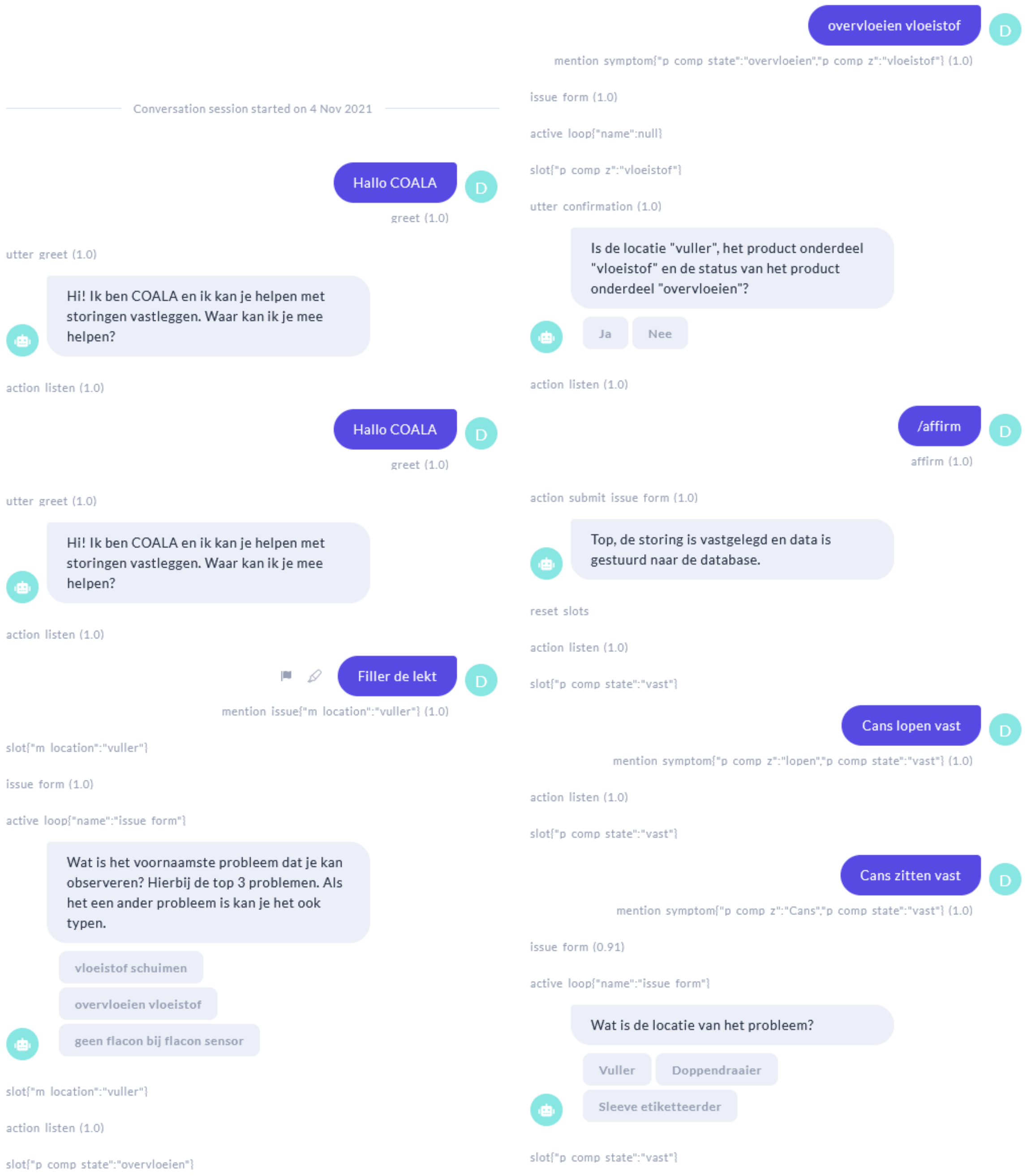
belemmerend	o o o o o o o	ondersteunend
complex	o o o o o o o	eenvoudig
inefficiënt	o o o o o o o	efficiënt
verwarrend	o o o o o o o	overzichtelijk
vervelend	o o o o o o o	spannend
oninteressant	o o o o o o o	interessant
conventioneel	o o o o o o o	origineel
gebruikelijk	o o o o o o o	nieuw

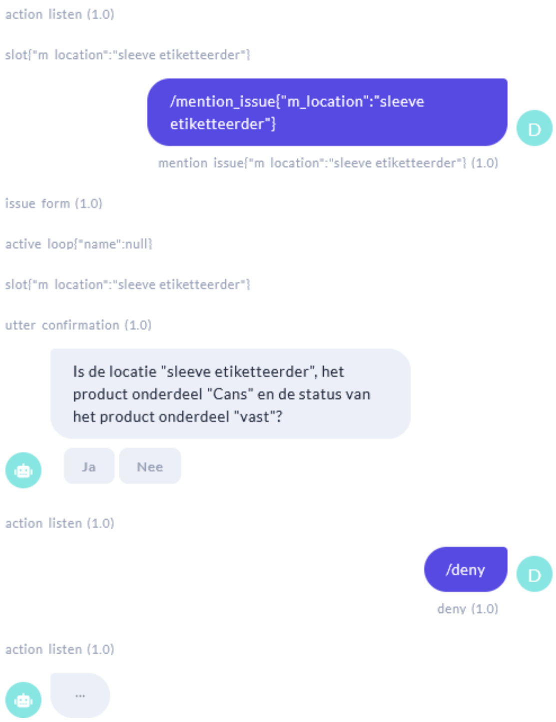
PS: Het prototype representeert niet het eindproduct.

Appendix D3: Full conversations prototype feature test

Here the full conversations captured in Rasa X for the prototype feature test are provided.

Participant #1, test A:




action listen (1.0)
slot{"m location":"sleeve etiketteerder"}
/mention_issue{"m_location":"sleeve etiketteerder"}
D
mention issue{"m location":"sleeve etiketteerder"} (1.0)
issue form (1.0)
active loop{"name":null}
slot{"m location":"sleeve etiketteerder"}
utter confirmation (1.0)
Is de locatie "sleeve etiketteerder", het product onderdeel "Cans" en de status van het product onderdeel "vast"?
Ja
Nee
action listen (1.0)
/deny
D
deny (1.0)
action listen (1.0)
...

Participant #1, test B:

Conversation session started on 4 Nov 2021

Hallo COALA! 1

greet (1.0)

utter greet (1.0)

Hallo! Ik ben COALA en ik kan je helpen met storingen vastleggen. Waar kan ik je mee helpen?

action listen (1.0)

Er is een probleem bij de filler 1

mention issue{"m location":"vuller"} (1.0)

slot{"m location":"vuller"}

issue form (1.0)

active loop{"name":"issue form"}

Wat is het voornaamste probleem dat je kan observeren?

slot{"m location":"vuller"}

action listen (1.0)

Er ligt zeep op de grond 1

mention symptom{"p comp z":"ligt"} (1.0)

slot{"p comp z":"ligt"}

issue form (1.0)

Wat is het voornaamste probleem dat je kan observeren?

slot{"p comp z":"ligt"}

action listen (1.0)

slot{"m location":"machine"}

Er ligt vloeistof in de machine 1

mention symptom{"p comp z":"vloeistof","m location":"machine"} (1.0)

issue form (1.0)

slot{"m location":"machine"}

Wat is het voornaamste probleem dat je kan observeren?

slot{"p comp z":"vloeistof"}

action listen (1.0)

slot{"p comp z":"Vloeistof"}

Vloeistof 1

mention symptom{"p comp z":"Vloeistof"} (1.0)

issue form (1.0)

Wat is het voornaamste probleem dat je kan observeren?

slot{"p comp z":"Vloeistof"}

action listen (1.0)

slot{"p comp state":"overvloeit"}

Het overvloeit 1

mention symptom{"p comp state":"overvloeit"} (1.0)

issue form (1.0)

active loop{"name":null}

slot{"p comp state":"overvloeit"}

utter confirmation (1.0)

Is de locatie "machine", het product onderdeel "Vloeistof" en de status van het product onderdeel "overvloeit"?

action listen (1.0)

Ja 1

affirm (1.0)

action submit issue form (1.0)

Dankjewel, de storing is vastgelegd en data is gestuurd naar de database.

reset slots

action listen (1.0)

slot{"m location":"sleeve etiketteerder"}

Er gaat iets mis bij de sleever 1

mention issue{"m location":"sleeve etiketteerder"} (1.0)

issue form (1.0)

active loop{"name":"issue form"}

Wat is het voornaamste probleem dat je kan observeren?

slot{"m location":"sleeve etiketteerder"}

action listen (1.0)

slot{"p comp state":"vast"}

De can loopt vast

1

mention symptom{"p comp z":"flacon","p comp state":"vast"} (1.0)

issue form (1.0)

active loop{"name":null}

slot{"p comp z":"flacon"}

utter confirmation (1.0)

Is de locatie "sleeve etiketteerder", het product onderdeel "flacon" en de status van het product onderdeel "['loopt', 'vast']"?

action listen (1.0)

Ja

1

affirm (1.0)

action submit issue form (1.0)

Perfect, de storing is vastgelegd en data is gestuurd naar de database.

reset slots

action listen (1.0)

Participant #2, test B:

Conversation session started on 4 Nov 2021

Hoi

greet (1.0)

utter greet (1.0)

Hallo! Ik ben COALA en ik kan je helpen met storingen vastleggen. Waar kan ik je mee helpen?

action listen (1.0)

De productie lijn staat stil

mention issue{"p comp z":"productie"} (1.0)

slot{"p comp z":"productie"}

issue form (1.0)

active loop{"name":"issue form"}

slot{"p comp z":"productie"}

Wat is de locatie van het probleem?

slot{"p comp z":"productie"}

action listen (1.0)

slot{"m location":"vuller"}

De vuller

mention issue{"m location":"vuller"} (1.0)

issue form (1.0)

Wat is het voornaamste probleem dat je kan observeren?

slot{"m location":"vuller"}

action listen (1.0)

slot{"m location":"vuller"}

De vloeistof ligt onder aan de vuller

mention issue{"p comp z":"vloeistof","p comp state":"onder","m location":"vull...

issue form (1.0)

active loop{"name":null}

slot{"m location":"vuller"}

utter confirmation (1.0)

Is de locatie "vuller", het product onderdeel "vloeistof" en de status van het product onderdeel "onder"?

action listen (1.0)

slot{"p comp state":"loopt uit"}

Nee de vloeistof loopt uit de canister

mention symptom{"p comp z":"flacon","p comp state":"loopt uit"} (1.0)

action listen (1.0)

Nee

deny (1.0)

issue form (0.76)

active loop{"name":"issue form"}

slot{"m location":"vuller"}

utter confirmation (1.0)

Is de locatie "vuller", het product onderdeel "flacon" en de status van het product onderdeel "loopt uit"?

action listen (1.0)

Ja dat klopt

affirm (1.0)

action submit issue form (1.0)

Dankjewel, de storing is vastgelegd en data is gestuurd naar de database.

reset slots

action listen (1.0)

Hee Coala, ik heb weer een probleem

mention issue (1.0)

issue form (1.0)

active loop{"name":"issue form"}

Wat is de locatie van het probleem?

slot{"requested slot":"m location"}

action listen (1.0)

De labeler

mention issue (1.0)

action execution rejected (1.0)

action default fallback (0.30)

rewind

action listen (1.0)

Labeler

mention issue (1.0)

action execution rejected (1.0)

action default fallback (0.30)

rewind

action listen (1.0)

slot{"m location":"sleeve etiketteerder"}

Sleever

mention issue{"m location":"sleeve etiketteerder"} (0.77)

issue form (1.0)

Wat is het voornaamste probleem dat je kan observeren?

slot{"m location":"sleeve etiketteerder"}

action listen (1.0)

slot{"m location":"machine"}

Het etiket blijft hangen in de machine

mention issue{"p comp z":"etiket","m location":"machine"} (1.0)

issue form (1.0)

Wat is het voornaamste probleem dat je kan observeren?

slot{"m location":"machine"}

action listen (1.0)

slot{"p comp state":"vast"}

Het label zit vast

mention symptom{"pronoun":"Het","p comp z":"label","p comp state":"vast"} (1...

issue form (1.0)

active loop{"name":null}

slot{"p comp z":"label"}

utter confirmation (1.0)

Is de locatie "machine", het product onderdeel "label" en de status van het product onderdeel "vast"?

action listen (1.0)

Ja bednakt COALA

affirm (0.74)

action submit issue form (1.0)

Dankjewel, de storing is vastgelegd en data is gestuurd naar de database.

reset slots

action listen (1.0)

Participant #2, test A:

Conversation session started on 4 Nov 2021

Hoi Coala

greet (1.0)

utter greet (1.0)

Hey! Ik ben COALA en ik kan je helpen met storingen vastleggen. Waar kan ik je mee helpen?

action listen (1.0)

slot{"m location":"vuller"}

Er is een probleem bij de vuller

mention issue{"m location":"vuller"} (1.0)

issue form (1.0)

active loop{"name":"issue form"}

Wat is het voornaamste probleem dat je kan observeren? Hierbij de top 3 problemen. Als het een ander probleem is kan je het ook typen.

vloeistof schuimen

overvloeien vloeistof

geen flacon bij flacon sensor

slot{"m location":"vuller"}

action listen (1.0)

slot{"p comp state":"overvloeien"}

overvloeien vloeistof

mention symptom{"p comp state":"overvloeien","p comp z":"vloeistof"} (1.0)

issue form (1.0)

active loop{"name":null}

slot{"p comp z":"vloeistof"}

utter confirmation (1.0)

Is de locatie "vuller", het product onderdeel "vloeistof" en de status van het product onderdeel "overvloeien"?

Ja

Nee

action listen (1.0)

/affirm

affirm (1.0)

action submit issue form (1.0)

Top, de storing is vastgelegd en data is gestuurd naar de database.

reset slots

action listen (1.0)

Hee COALA

greet (1.0)

utter greet (1.0)

Hallo! Ik ben COALA en ik kan je helpen met storingen vastleggen. Waar kan ik je mee helpen?

action listen (1.0)

slot{"m location":"sleeve etiketteerder"}

Heb een probleem bij de sleever

mention issue{"m location":"sleeve etiketteerder"} (1.0)

issue form (1.0)

active loop{"name":"issue form"}

Wat is het voornaamste probleem dat je kan observeren? Hierbij de top 3 problemen. Als het een ander probleem is kan je het ook typen.

sleeve gekreukeld geplaatst

sleeve gescheurd

sleeve rol is op

slot{"m location":"sleeve etiketteerder"}

action listen (1.0)

slot{"p comp state":"gekreukeld"}

sleeve gekreukeld geplaatst

mention symptom{"p comp z":"sleeve","p comp state":"gekreukeld"} (1.0)

issue form (1.0)

active loop{"name":null}

slot{"p comp z":"sleeve"}

utter confirmation (1.0)

Is de locatie "sleeve etiketteerder", het product onderdeel "sleeve" en de status van het product onderdeel "gekreukeld"?

Ja   Nee

action listen (1.0)

/affirm

A

affirm (1.0)

action submit issue form (1.0)

Dankjewel, de storing is vastgelegd en data is gestuurd naar de database.

reset slots

Appendix D4: Observations and feedback prototype features test

Here the observations and feedback during the tests are listed.

Participant 1, test B (with buttons):

The CA started slow when user uttered hello. Had to refresh.

“Cans lopen vast” the CA gets stuck and does not know what to do.

Aside from above mentioned 1st issue capture went rather smoothly.

The user started with issue first and then went into location.

Feedback:

The buttons for m_location gave a hint of what the location was, otherwise for the participant it was difficult to see in the images what the machine location was. The context was missing with the 2nd issue.

The user would like to adjust the issue description in the middle of documentation.

After denying the control step, the participant wanted to correct the description and expected the CA to utter: “Ok, what is wrong then?”

Participant sees the benefit of a voice assisted CA and also sees the difficulties with a text-based approach.

According to the participant the questions are clear.

When typing it gives a good overview of what is said because you can scroll back and read what has been said. This is not that easily possible with a voice assisted CA.

Participant 1, test A (without buttons):

During issue capturing, the CA was looping the question for p_comp_state, while p_comp was already captured. The participant got confused by this. 3 times it asked for the observable issue. In issue capturing, it replaced filler by machine for m_location.

Feedback:

Participant shared that he could imagine that if he was an experienced operator that he does not have to always see the buttons because he already knows what the issue is. It depends a little what people prefer. The participant provides an example where if you are in the car, some people appreciate if the navigation talks to you and communicates everything while others prefer to look at the screen themselves.

The issue form is very formal, in stead of: “What is the predominant issue that you can observe?” replace by “What is the issue/What do you see going wrong?”

And: “Perfect, issue is captured” instead of “Thank you, the issue has been captured and sent to the database.”

It would be nice to scale the CA to the experience of the operator.

Participant 2, test A (without buttons):

Surprisingly could process very accurately the first user input: “De productie lijn staat stil.” (1.00 confidence).

Another surprise: after check the user denies, and corrects the system and it replaced the correct slot with the correct entity and worked.

First issue capture went surprisingly smoothly. One little hiccup which was not anticipated, was resolved by the CA itself.

CA does not know the synonym “labeler”. User utters it twice before mentioning the sleever.

“Het etiket blijf hangen in de machine” is recognized as an mention issue intent instead of mention symptom intent. Machine captured as machine location entity. It erased the previous sleever input.

Afterwards the participant utters: “label zit vast” and works, even though the “label” was never specified as entity in the training data.

Feedback:

Participant: The system is very supportive in the fact that it was guiding me. Although I did not get any feedback when it needed assistance.

To the participant it was straightforward. The participant would have liked it even more if it would fill out some things automatically for them.

It is difficult to get a grasp on what the easiest way is to capture an issue in the right way.

The feedback in between with second test is appreciated. Would prefer a better visual layout. Visual queues that the bot has recognized something or is working to process the information. Participant gives an idea: Some kind of lamp going green when CA has properly recognized the user.

The “typing” bubble by the CA is exciting because you are not sure what is going to come.

Would most probably find the task boring to do. Especially if you imagine in the timeframe of the operators.

The issue is related to the sleever or labeler, than the participant is expecting a question about the specific machine. Only clear with the control step what is being captured. Want more feedback.

Shared an insight gathered from an operator: The operator did not feel that capturing issues with ODCE was of value to her. What was of value, was data on the production line. It gave her insights into how she was performing and if she needed to adjust her approach if she was producing worse than she wants.

Partcipant 2, test B (with buttons):

It is foaming, but I definitely can see it flowing over, so my choice goes for flowing over.

Capturing issue went smoothly.

Feedback:

With the 2nd issue participant had a bias for the proposed buttons.

I looked at the options of the buttons and picked one that seemed to fit the problem the best.

Provide a 4th option: Else. Then the user can press that button and type. Although in the text it says you can type another answer, it still is not that clear to the participant.

Capturing the issue was fast. Participant is curious how fast it will be if the issue is not one of the options and how good it will work.

What will it do then? Can it provide the next top 3? Or do you just have to type the issue then?

Now it was clear directly that the CA understood the participant as you got visual feedback through the buttons and specific to the machine.

Participant talks about if the top 3 issue are approximately 80% of the issue, then he thinks it is a great approach. Then only 20% of the issues have to be typed while most can be captured rather quickly with the buttons.

Maybe an icon that indicates the significance or severity of the issue?

Participant would like to not have to say hi every time.

Is it possible to let the previous issue be repeated by the CA?

The participant feels that if you have to do this over and over, some pain points are there in the process that can create frustrations.

Some form of external motivation to fill it in, provides an example: "This is the 100th time this issue has been captured by you! Good job!"

The participant can imagine that the task could be repetitive.

Appendix E: Project brief

DESIGN FOR our future

TUDelft

IDE Master Graduation

Project team, Procedural checks and personal Project brief

This document contains the agreements made between student and supervisory team about the student's IDE Master Graduation Project. This document can also include the involvement of an external organisation, however, it does not cover any legal employment relationship that the student and the client (might) agree upon. Next to that, this document facilitates the required procedural checks. In this document:

- The student defines the team, what he/she is going to do/deliver and how that will come about.
- SSC E&SA (Shared Service Center, Education & Student Affairs) reports on the student's registration and study progress.
- IDE's Board of Examiners confirms if the student is allowed to start the Graduation Project.

USE ADOBE ACROBAT READER TO OPEN, EDIT AND SAVE THIS DOCUMENT

Download again and reopen in case you tried other software, such as Preview (Mac) or a webbrowser.

STUDENT DATA & MASTER PROGRAMME

Save this form according the format "IDE Master Graduation Project Brief_familyname_firstname_studentnumber_dd-mm-yyyy". Complete all blue parts of the form and include the approved Project Brief in your Graduation Report as Appendix 1 !

family name	Hadzisejdic 4947
initials	B. given name Boris
student number	4396618
street & no.	Overstortpad 111
zipcode & city	3063 SK Rotterdam
country	Netherlands
phone	0614376413
email	b.hadzisejdic@student.tudelft.nl

Your master programme (only select the options that apply to you):

IDE master(s):	★ IPD / Dfl / SPD
2nd non-IDE master:	
individual programme:	- - (give date of approval)
honours programme:	Honours Programme Master
specialisation / annotation:	Medisign
	Tech. in Sustainable Design
	Entrepeneurship

SUPERVISORY TEAM **

Fill in the required data for the supervisory team members. Please check the instructions on the right !

** chair	Rusák, Z.	dept. / section:	IoT
** mentor	Kernan Freire, S.	dept. / section:	IoT
2nd mentor	Barnabas Kiss		
	organisation: Diversey		
	city: Enschede	country:	Netherlands

comments (optional): My chair and mentor, although from the same department, provide different perspectives to the project. My chair has AI focus and knowledge, while mentor has IDE background and specific knowledge in conversational AI.

Chair should request the IDE Board of Examiners for approval of a non-IDE mentor, including a motivation letter and c.v..

Second mentor only applies in case the assignment is hosted by an external organisation.

Ensure a heterogeneous team. In case you wish to include two team members from the same section, please explain why.

APPROVAL PROJECT BRIEF
To be filled in by the chair of the supervisory team.

chair Rusák, Z. date 28 - 04 - 2021 signature Zoltan Rusak Digitally signed by Zoltan Rusak Date: 2021.04.28 10:59:14 +02'00'

CHECK STUDY PROGRESS
To be filled in by the SSC E&SA (Shared Service Center, Education & Student Affairs), after approval of the project brief by the Chair.
The study progress will be checked for a 2nd time just before the green light meeting.

Master electives no. of EC accumulated in total: 24 EC

Of which, taking the conditional requirements into account, can be part of the exam programme 24 EC

List of electives obtained before the third semester without approval of the BoE

YES all 1st year master courses passed

★ NO missing 1st year master courses are:

ID4070 (IDE academy, 4 EC)

name J. J. de Bruin date 29 - 04 - 2021 signature J. J. de Bruin, SPA Digitally signed by J. J. de Bruin, SPA Date: 2021.04.29 09:35:58 +02'00'

FORMAL APPROVAL GRADUATION PROJECT
To be filled in by the Board of Examiners of IDE TU Delft. Please check the supervisory team and study the parts of the brief marked **. Next, please assess, (dis)approve and sign this Project Brief, by using the criteria below.

- Does the project fit within the (MSc)-programme of the student (taking into account, if described, the activities done next to the obligatory MSc specific courses)?
- Is the level of the project challenging enough for a MSc IDE graduating student?
- Is the project expected to be doable within 100 working days/20 weeks ?
- Does the composition of the supervisory team comply with the regulations and fit the assignment ?

Content: ★ APPROVED NOT APPROVED

Procedure: APPROVED ★ NOT APPROVED

- the missing course IDE Academy (4 EC) should be finished before the green light meeting
- no (unknown) abbreviation in title, please write RCA in full

comments

name Monique von Morgen date 10 - 05 - 2021 signature

Initials & Name B. Hadzisejdic 4947 Student number 4396618

Title of Project Design of a conversational assistant for standardized issue description

Design of a conversational assistant for standardized issue description

project title

Please state the title of your graduation project (above) and the start date and end date (below). Keep the title compact and simple. Do not use abbreviations. The remainder of this document allows you to define and clarify your graduation project.

start date 15 - 04 - 2021 | 17 - 09 - 2021 end date

INTRODUCTION **

Please describe, the context of your project, and address the main stakeholders (interests) within this context in a concise yet complete manner. Who are involved, what do they value and how do they currently operate within the given context? What are the main opportunities and limitations you are currently aware of (cultural- and social norms, resources (time, money,...), technology, ...).

COALA is a H2020 EU project with 17 partners including TU Delft and three business partners: Whirlpool, Diversey and Piacenza.
The goal of this project is to develop a cognitive digital assistant for optimizing manufacturing by:
- reducing the number of quality incidents
- by reducing the time needed for on-the-job training of workers
- Reducing scepticism regarding the use of a voice-enabled cognitive digital assistant in manufacturing environments

The main opportunities for the cognitive digital assistant are in optimizing the root cause analysis in manufacturing and sharing the operators expert knowledge more efficiently with novice workers.
However, many limitations are still present in its current early state. Time and money are needed to develop the technology, data is needed to train the algorithms for the AI and the technology should be accepted by the operators to work correctly. In order for the operators to accept the tech, a goal has been set that no personal data is stored by the assistant, which further complicates the situation.
Furthermore, information given by the assistant to the operators comes with a certain risk. The information given should be as correct as possible with a low risk factor.

I will focus on optimizing the conversation between the cognitive assistant and the operators with this context in mind. More specifically I will focus on the collection of information during issue handling where multiple conversation can take place. This information can be used by the assistant in a later stage to give recommendations to the operators. One example of such a conversation is illustrated in figures 1 and 2.

space available for images / figures on next page

introduction (continued): space for images

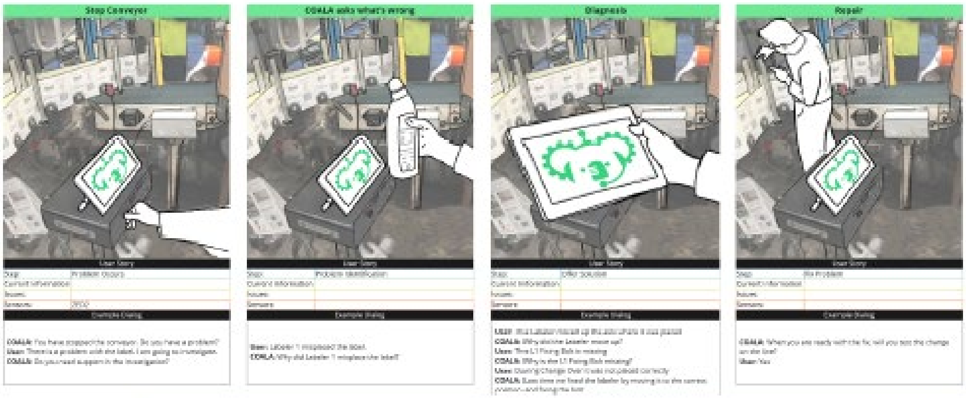


image / figure 1: Issue handling storyboard (part 1)

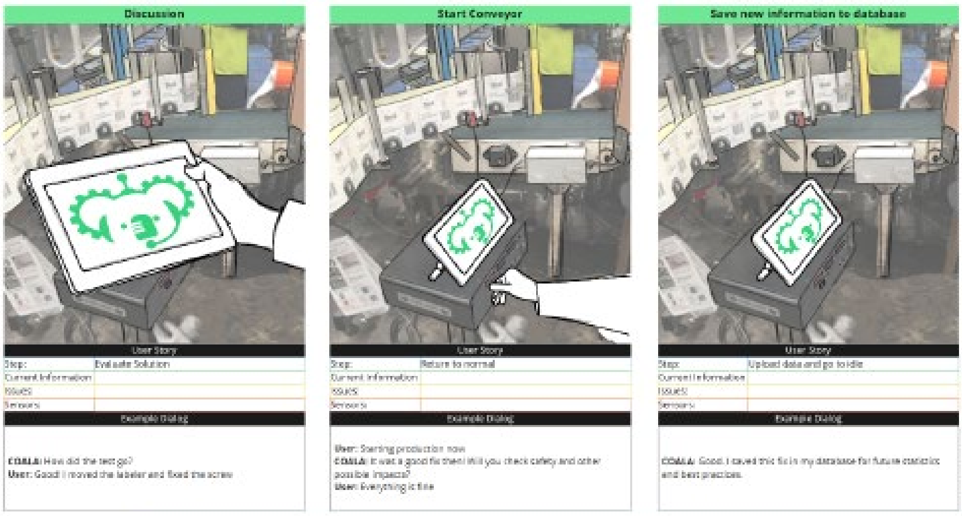


image / figure 2: Issue handling storyboard (part 2)

Initials & Name B. Hadzisejdic 4947 Student number 4396618

Title of Project Design of a conversational assistant for standardized issue description

PROBLEM DEFINITION **
Limit and define the scope and solution space of your project to one that is manageable within one Master Graduation Project of 30 EC (= 20 full time weeks or 100 working days) and clearly indicate what issue(s) should be addressed in this project.

The COALA digital assistant will be integrated into the IT infrastructure of several companies in order to be applied in the production facilities for problem identification, root cause analysis and solution delivery. It will utilize posture tracking for contextual awareness and voice communication for interacting with operators in the facilities.
I will be designing and optimizing the dialog between the COALA digital assistant and the operators for collecting data during issue handling.
This data will later be used by the assistant to give recommendations to the operators during issue handling. However this falls outside the graduation project scope.

Relevant questions:

- How can dialogs between the digital assistant and the expert operators be designed for capturing the operators implicit knowledge during issue handling?
- What is the most efficient way to conduct a dialog for capturing the root cause during issue handling?
- How can the digital assistant use contextual information to shorten dialogs between it and the operator?
- How can dialogs between the digital assistant and the expert operators be designed to promote co-operation?

ASSIGNMENT **
State in 2 or 3 sentences what you are going to research, design, create and / or generate, that will solve (part of) the issue(s) pointed out in "problem definition". Then illustrate this assignment by indicating what kind of solution you expect and / or aim to deliver, for instance: a product, a product-service combination, a strategy illustrated through product or product-service combination ideas, ... In case of a Specialisation and/or Annotation, make sure the assignment reflects this/these.

I will design a service prototype which is able to collect information from the operators during issue handling, which is optimized for efficient data collection.

The expected outcome is a service prototype which is able to collect information from the operators during issue handling, which is optimized for efficient data collection. When the information is missing, the prototype should be able to have a dialog with the operators to further clarify and complete the information.

Several prototype iterations are expected where the complexity per iteration increases. First prototype will mainly focus on the collection of data. The second prototype will be for data analysis and the third prototype is aimed at the dialog between operator and assistant when information is missing.

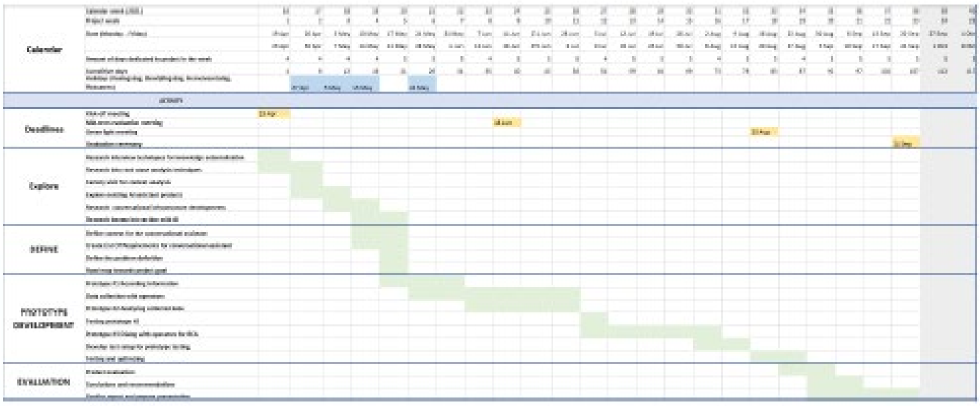
Before this is possible however, observations and interviews of and with the operators are to be done for creating an understanding of the context. Furthermore, conversations with experts on human interaction with AI and human psychology are to be had. Research into current products and developments for AI is to be done.
All these elements should result in a greater understanding of the context, the possibilities of AI but also the restrictions.
This knowledge should be utilized for optimizing the prototypes.

Initials & Name B. Hadzisejdic 4947 Student number 4396618
Title of Project Design of a conversational assistant for standardized issue description

PLANNING AND APPROACH **

Include a Gantt Chart (replace the example below - more examples can be found in Manual 2) that shows the different phases of your project, deliverables you have in mind, meetings, and how you plan to spend your time. Please note that all activities should fit within the given net time of 30 EC = 20 full time weeks or 100 working days, and your planning should include a kick-off meeting, mid-term meeting, green light meeting and graduation ceremony. Illustrate your Gantt Chart by, for instance, explaining your approach, and please indicate periods of part-time activities and/or periods of not spending time on your graduation project, if any, for instance because of holidays or parallel activities.

start date 15 - 4 - 2021 17 - 9 - 2021 end date



The Gantt chart above shows the planning of the project. The presented planning consists of a total of 22 weeks. The idea is to work 5 days a week on the project for 14 of the 22 weeks. The other 8 weeks have a 4 day work week. The first 4 weeks of the project have 4 working days, in order to finish the course Read a Book. The other 4 day work weeks are spread (every other 2/3 weeks) throughout the project in order to create time for reflection on the project so far. However, due to my back issues (hernia) and the fact that a lot of sitting will be done during the project, 2 extra weeks are planned to be added. In this case, the 14 full work weeks will be replaced by 4 day work weeks. This way, the back issues are more manageable and I will have more time to physically work on this matter.

The first 5 weeks of the project are mainly focused on exploring and defining the context and exploring the possibilities with AI. During week 5 a start will be made on prototype #1, in order to collect data as soon as possible, as this is needed for prototype #2. Weeks 8 through 12 are planned for prototype #2 and weeks 12 through 19 for prototype #3. Weeks 16 through 20 are for developing a test setup and testing, optimizing and evaluating the final prototype. After which conclusions and recommendations are giving for the project wrap up in the presentation.

MOTIVATION AND PERSONAL AMBITIONS

Explain why you set up this project, what competences you want to prove and learn. For example: acquired competences from your MSc programme, the elective semester, extra-curricular activities (etc.) and point out the competences you have yet developed. Optionally, describe which personal learning ambitions you explicitly want to address in this project, on top of the learning objectives of the Graduation Project, such as: in depth knowledge a on specific subject, broadening your competences or experimenting with a specific tool and/or methodology, ... Stick to no more than five ambitions.

I believe the importance of design and its impact on the world is increasing. Especially with subjects such as this one, where AI has the center stage. With proper knowledge of the context, designers should be able to design something that is human-centered. The project of COALA is in line with this mindset. The digital assistant is not a machine that replaces the operators, but instead engages with the operators for co-operation in order to improve problem solving resulting in greater manufacturing speeds.

During my elective semester I followed the course 'Machine Learning for Intelligent Products' and it further enriched my interest for Artificial Intelligence and its many possibilities. I want to further develop my knowledge in this expertise area.

Personal ambitions:
- Develop a greater understanding of NLP
- Able to create a service prototype by coding
- Human-centered product

FINAL COMMENTS

In case your project brief needs final comments, please add any information you think is relevant.

Three courses need to be finished and the ECTs should be registered before the green light meeting: IDE Academy (ID4070), Read a Book (ID5040) and Initiate to Graduate (ID5080). The necessary workshops for IDE Academy have been done, I am currently in contact with the staff of IDE Academy to finalize the administration for the ECTs. Extra 4 days are planned in the first 4 weeks of the graduation project for finishing Read a Book. The ECTs for Initiate to Graduate will be given when this project brief has been approved.