FH Vorarlberg
University of Applied Sciences

Robotergestützte Prozessautomatisierung
in SAP

Masterarbeit
zur Erlangung des akademischen Grades

Master of Science

Fachhochschule Vorarlberg

Informatik Master

Betreut von
Prof. (FH) Dipl.-Ing. Dr. techn. Karl-Heinz Weidmann

Vorgelegt von
Dominik Meyer, BSc

Dornbirn, 08.07.2022

Kurzreferat

Robotergestützte Prozessautomatisierung im SAP

Die tägliche Arbeit von Büroangestellten besteht oft aus Aufgaben, die sich in verschiedenen Zeitabständen wiederholen. Die Art unterscheidet sich abhängig vom Berufsfeld. Jedoch kostet deren Ausführung Zeit und den stört Arbeitsfluss. Könnten diese Aufgaben automatisiert werden, hätte dies den Vorteil, dass zeitliche Ressourcen freigesetzt würden und es dadurch zu einer Steigerung der Produktivität käme, da diese zusätzlichen Ressourcen anderweitig genutzt werden könnten. Daher ist es notwendig geeignete Aufgaben ausfindig zu machen und anhand der entsprechenden Technologien zu automatisieren.

Das Ziel der vorliegenden Arbeit ist es, die repetitiven Aufgaben in der definierten Zielgruppe zu eruieren und diese in weiterer Folge zu automatisieren. Dazu wird folgende Forschungsfrage gestellt: „Implementierung eines Chatbots anhand von der SAP zur Verfügung gestellter Technologien, um repetitive Aufgaben zu automatisieren."

Die Umsetzung der Arbeit erfolgte im Rahmen der Tätigkeit bei einem SAP Beratungshaus. Dadurch beschränkt sich die Zielgruppe auf hausinterne SAP Berater*innen und SAP Technologien. Ziel dieser Arbeit war es, einen Prototypen zu implementieren der es ermöglicht, dass die Berater*innen sich täglich wiederholende Aufgaben anhand eines Chatbots automatisiert ausführen können. Als Ergebnis dieses Greenfield Projektes sollte ein Proof-of-concept zur Verfügung gestellt werden, der die ausgewählten Anwendungsfälle abdeckt und eine Grundlage für die Erweiterung bildet. Die Hauptaufgabe liegt zum einen in der Erfassung der Anwendungsfälle und zum anderen im Aufbau von entsprechendem Know-How für die Umsetzung mit den ausgewählten Technologien.

Als Technologie, zur Interaktion mit dem Benutzer, wird für den Chatbot die SAP Conversational AI verwendet. Zur Ausführung der Aufgaben erfolgt die Umsetzung anhand der SAP Intelligent Robot Process Automation im Zusammenspiel mit dem SAP GUI Scripting und anwendungsfallspezifischen ABAP Programmen.

Die Evaluation der User Expirience erfolgt in Form des SUMI Fragebogens. Dieser untersucht die Dimensionen der Effizienz, Beinflussbarkeit, Hilfsbereitschaft, Kontrollierbarkeit und Erlernbarkeit. Für die Durchführung der Untersuchung wurden den Anwender*innen Szenarien vorgelegt, welche alltägliche Situationen widerspiegeln. Basierend auf diesen Szenarien sollten die Aufgaben mit Hilfe des Chatbots gelöst werden. Das Ergebnis der Untersuchung zeigt, dass der Prototyp vor allem in den Bereichen der Effizienz und der Kontrollierbarkeit einiges an Verbesserungspotenzial offenbart hat. Jedoch wird der Nutzen der Anwendung von acht der zehn Testpersonen als wichtig bis sehr wichtig eingestuft.

Die Umsetzung legt die Möglichkeiten und das Potenzial der Technologien offen und stellt mit dem Prototypen eine solide Grundlage für zukünftige Entwicklungen zur Verfügung. Weiters werden durch die Evaluation die Schwächen und Stärken offengelegt und ein Konzept verwendet, welches auch für zukünftige Erweiterungen verwendet werden kann.

Abstract

Robot Process Automation in SAP

The daily work of office workers often consists of tasks that are repeated at various intervals. The type differs depending on the occupational field. On one hand the execution consumes a lot of time and on the other hand interrupts the workflow. If these tasks could be automated, this would have the advantage that time resources would be freed up and this would lead to an increase in productivity, as these additional resources could be used otherwise. Therefore, it is necessary to find suitable tasks to automate using the appropriate technologies.

The aim of this study is to identify the repetitive tasks in the defined target group and automate them. This leads to the following research question: "Implementation of a chatbot using technologies provided by SAP to automate repetitive tasks".

The implementation of the work was carried out within the context of the work at an SAP consulting company. Thus, the target group is limited to in-house SAP consultants and SAP technologies. The aim of this work was to implement a prototype that helps the consultants to automate daily repetitive tasks with the help of a chatbot. As a result of this greenfield project, a proof-of-concept should be made that covers the selected use cases and builds a fundament for future development. The main tasks are to capture the use cases and also to build up the corresponding know-how for the implementation with the selected technologies.

SAP Conversational AI is used as the technology for interaction with the user. For the execution of the tasks, the implementation is carried out using SAP Intelligent Robot Process Automation together with SAP GUI scripting and application-specific ABAP programs.

The evaluation of user experience is measured using the SUMI questionnaire. This examines the dimensions of efficiency, influence, helpfulness, controllability and learnability. For the survey, the users were put in scenarios that reflect everyday situations. Based on these scenarios, the tasks had to be solved with the help of the chatbot. The result of the study shows that the prototype has some potential for improvement, especially in the fields of efficiency and controllability. The application's usefulness is rated as important up to very important by eight out of the ten test persons.

The implementation reveals the possibilities and potential of the technologies and provides a solid basis for future developments with the prototype. Furthermore, the evaluation demonstrates the weaknesses and strengths and uses a concept that can also be used for future extensions.

Inhaltsverzeichnis

Abbildungsverzeichnis

Tabellenverzeichnis

Abkürzungsverzeichnis

CMO ..... Chief Marketing Officer

CSO ..... Chief Security Officer

CIO ..... Chief Information Officer

CAI ..... Conversational Artificial Intelligence

RPA ..... Robot Process Automation

NLP ..... Natural Language Processing

NLG ..... Natural Language Generation

GUI ..... Graphical User Interface

SAP ..... Systeme Anwendungen Produkte

KI ..... Künstliche Intelligenz

BTP ..... Business Technology Platform

1 Grundlagen der Chatbottechnologie

Bei einer Studie der Firma Oracle gaben rund 80% der Befragten an, bereits eine Art der Chatbot Technologie zu verwenden oder dies in Zukunft planen. Bei den befragten Personen handelte es sich um CMO, CSO und leitende Vertriebsmitarbeiter*innen. Durch den Einsatz dieser Technologie können Aufgaben unterschiedlicher Komplexitäten automatisiert werden. Beginnend von einer einfachen Wettervorhersage bis hin zu personalisierten Einkaufsempfehlungen. (Intelligence 2016)

Basierend auf diesen Zahlen wird im nachfolgenden Kapitel auf die Anfänge der Chatbots eingegangen und weiterführend werden die Bedeutung und deren Anwendungsgebiete in der heutigen Zeit aufgearbeitet. Aufbauend darauf erfolgt die Erläuterung zur Umsetzung im Rahmen der Arbeit bezugnehmend auf Technologie, Zielgruppe und Anwendungsfälle.

1.1 Zeitliche Entwicklung

Die Technologie wurde erstmals im Jahre 1966 angewendet. Durch den Einsatz von Maschinellem Lernen und Künstlicher Intelligenz ist der Unterschied zwischen Bots von damals und heute tiefgründig. Im folgenden Teil wird auf den Ursprung der Technologie, sowie deren Einsatz mit heutigem Stand der Technik eingegangen.

1.1.1 Ursprung

Die Entwicklung des ersten Chatbots stammt aus dem Jahr 1966. In diesem Jahr entwickelte Joseph Weizenbaum „ELIZA“. Das Programm sollte erstmals eine Kommunikation zwischen Mensch und Maschine in natürlicher Sprache ermöglichen. Das Programm, damals war das Wort Chatbot oder gar Ähnliches nicht bekannt, sollte die Rolle eines Therapeuten übernehmen. Die Antworten basierten dabei auf der Erkennung von sogenannten Schlüsselwörtern. Diese Funktionalität wird in Abschnitt 2.2.1 nochmals genauer aufgegriffen. (Kohne u.a. 2020, S. 7–9)

Ein weiterer Pionier in diesem Feld war Alan Turing. Von diesem stammte auch der gleichnamige Turing Test. Im Rahmen des Turing Tests geht es darum festzustellen, wie menschenähnlich sich die Maschine in der Kommunikation verhält in Bezug auf Empathie, Länge der Antworten, Satzstellungen und Ähnlichem. Dabei unterhält sich die Person mit zwei unbekannten Gesprächspartnern. Bei einem der beiden handelt es sich tatsächlich um einen Menschen, beim anderen um ein Computersystem. In einer Reihe von Unterhaltungen versuchen beide Teilnehmer die Testperson zu überzeugen, dass es sich bei ihnen nicht um eine Maschine handelt. Der Test wird als bestanden betrachtet, wenn es der Maschine in mehr als 30% der Fälle gelingt, als Mensch erkannt zu werden. (Gentsch 2019, S. 29)

Der Turing Test wurde erstmals 2014 von dem Chatbot Namens „Eugene Goostman“ bestanden. Dem Bot, welcher einen 13-jährigen Jungen imitiert, gelang es im Rahmen eines Wettbewerbs 33% der Gesprächspartner davon zu überzeugen, dass es sich bei der Unbekannten gegenüber um einen realen Menschen handelte. (Gentsch 2019, S. 99)

1.1.2 Aktueller Stand

Die Bedeutung und Verwendung von Chatbots werden durch die Grafik in Abbildung verdeutlicht.

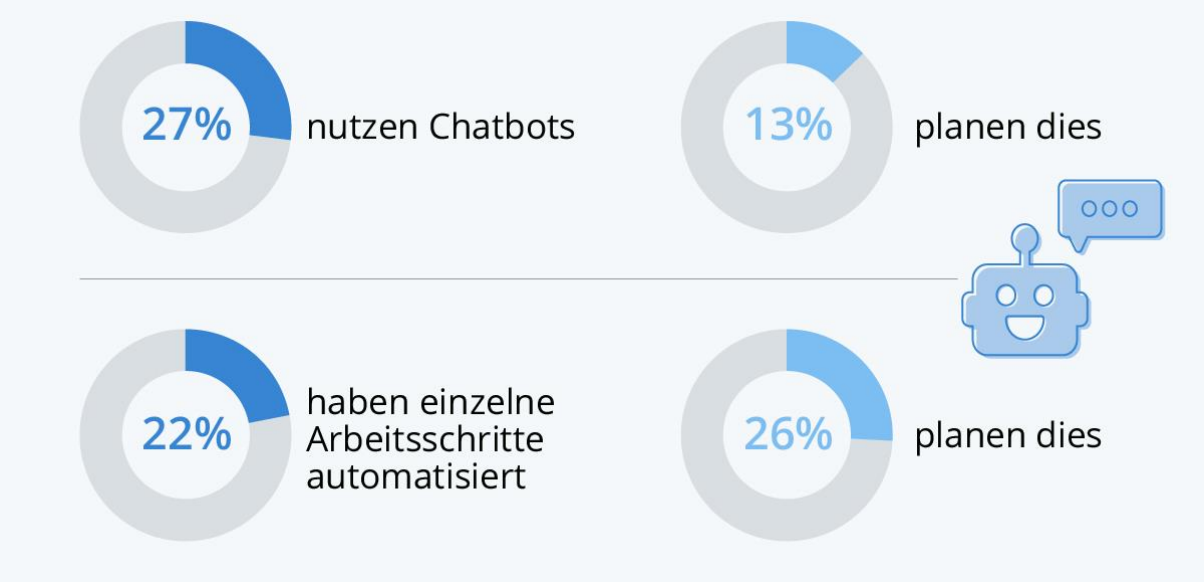


Abbildung 1: Verwendung von Chatbots in Deutschen Unternhemen (Suhr 2020)

Laut einer durchgeführten Studie von bitkom greift bereits jedes vierte Unternehmen in Deutschland auf die Bottechnologie zurück. 27% davon geben an, den Bot für einfache Kundenfragen zu nutzen. Darunter fallen etwaige Fragen zu Öffnungszeiten, Fristen, Rücksendungen oder ähnlichen, trivialen Anfragen. Darüber hinaus ist die Verwendung bei 13% der Befragten in nächster Zeit geplant.

Der Einsatz der Technologie beschränkt sich dabei aber nicht nur auf einfache Anfragen. Von den 1104 befragten Personen gaben 22% an, die Technologie auch auf die Automatisierung von Büro- und Verwaltungsprozessen anzuwenden. Weitere 26% haben die Umsetzung in Zukunft geplant.

Bei den befragten Teilnehmern*innen handelt es sich um Geschäftsführer, Vorstandmitglieder oder CIO von Unternehmen mit einer Mindestgröße von 20 Mitarbeitern. (Suhr 2020)

1.2 Anwendungsgebiete

Wie vorher durch die beiden Studien von Oracle und bitkom belegt, sind Chatbots bereits ein wichtiger Bestandteil des operativen Arbeitens oder werden es ist in naher Zukunft sein. Doch was für einen Nutzen haben Chatbots und in welchem Anwendungsfeld eignet sich der Einsatz dieser Technologie?

Da Chatbots überall zum Einsatz kommen können, wo die Kommunikation mit dem Menschen eine Rolle spielt, ergibt sich ein sehr breites Anwendungsgebiet. Eine Möglichkeit der Differenzierung ergibt sich zwischen externer und interner Kommunikation.

1.2.1 Externe Kommunikation

Bei der externen Kommunikation steht die Kommunikation mit den Endkunden im Mittelpunkt. In dieser Form ist Zahl der Nutzer*innen meist größer als in der internen Kommunikation. Durch die breite Gruppe der anwendenden Personen entstehen verschiedenste Anwendungsfälle abhängig von der Branche. Nachfolgend werden einige Fälle aufgelistet, um die Breite von bereits existierenden Anwendungsmöglichkeiten der Technologie aufzuzeigen:

- Handel: Kaufempfehlungen
- Logistik: Status zu einem Paket abfragen
- Gesundheit: Beratung (analog zur Apotheke)
- Automotiv: Fragen zum Fahrzeug
- Reisen: Flugvergleiche und -buchungen

Die genannten Beispiele sind weitestgehend im Bereich der Kundenberatung anzusiedeln. Jedoch ist auch ein Einsatz in Neukundengewinnung oder Kundenbindung möglich, was nachstehendes Beispiel verdeutlicht.

Bei dem Wettanbieter Tipico besteht die Möglichkeit eine Wette über den Bot abzugeben. Es muss dabei weder durch die unzähligen Wettmöglichkeiten gescrollt werden, noch muss Wissen zur Tippabgabe oder Ähnlichem vorhanden sein. Durch den Bot wird dabei die Technologieschwelle, die bei der Nutzung einer App gegeben ist, für Kunden verringert.

Beim Einsatz in der Kundenbindung bringt ein Chatbot jenen Vorteil mit, dass er 24 Stunden jeweils 7 Tage die Woche verfügbar ist. Die Kunden müssen für einfache Fragen nicht in einer langen Warteschlange verharren, sondern bekommen direkt und einfach Auskunft über den digitalen Partner. Die Komplexität der Aufgaben hält sich jedoch in Grenzen. Diese Grenzen und deren Problematik werden in Abschnitt 3.2 genauer behandelt.

1.2.2 Interne Kommunikation

In der internen Kommunikation ist die Zahl der Teilnehmer*innen meist wesentlich geringer und die Aufgaben sind oft klarer definiert. Bei dieser Form werden meist repetitive Aufgaben durch den Bot abgedeckt. Dadurch müssen Anwender*innen die sich wiederholenden Aufgaben nicht manuell durchführen, sondern können dies in Form des Chatbot veranlassen und die dadurch verfügbaren Ressourcen anderweitig nutzen. Unter diese, sich wiederholende und gleichbleibende Aufgaben, können folgenden Fälle fallen:

- Zurücksetzen eines Passwortes
- Erstellen einer Servicemeldung
- Buchung eines Raumes

Je nach Branche und Unternehmen fallen unterschiedliche Aufgaben an. (Kohne u.a. 2020, S. 29–37)

1.2.3 Praktische Umsetzung

Anhand von Abschnitt 1.2.1 und Abschnitt 1.2.2 wird versucht eine Unterscheidung zwischen den Anwendungsfeldern vorzunehmen. Basierend auf dieser lässt sich auch die Umsetzung der Arbeit einordnen.

Die Arbeit wurde im Rahmen der Tätigkeit bei der Firma proTask implementiert. Resultierend daraus wurden auch die Technologie, die Zielgruppe und die Anwendungsfälle definiert. Auf diese wird in nachfolgenden Abschnitten detailliert eingegangen.

1.2.3.1 Technologie

Die Firma proTask positioniert sich als SAP-Systemhaus. Daraus ergeben sich auch die in der Umsetzung verwendeten Technologien. Hauptaugenmerk lag dabei auf der Verwendung von SAP zur Verfügung gestellten Möglichkeiten. Bei diesen handelte es sich zum einen um SAP Conversational AI Version 2021 (CAI), und zum anderen um die SAP Intelligent Robot Process Automation (RPA) mit SDK Version 1.21.57 und Agent Version 2.0.20.

Die CAI fungiert als Schnittstelle zum*zur Benutzer*in und implementiert den Chatbot. Die RPA führt entsprechende Automatisierungen für den*die Benutzer*in aus und wird über die CAI angesteuert. Die detaillierte Architektur der Anwendung kann aus Abschnitt 2.3 entnommen. Weiteres werden CAI und RPA ausführlich in Abschnitt 4 und 5 behandelt.

1.2.3.2 Zielgruppe

Die im vorherigen Absatz beschriebenen Technologien fanden, bis zur Durchführung der Arbeit noch keine Anwendung innerhalb des Unternehmens und wurden somit erstmalig verwendet. Die Hauptzielgruppe beschränkte sich aufgrund der Tatsache, dass bisher weder Erfahrungen auf dem Gebiet der Chatbots noch auf dem Gebiet der von SAP dafür zur Verfügung gestellten Technologien gemacht wurden, auf alle hausinternen Mitarbeiter der Beratungsabteilung. Bei den Personen der Zielgruppe handelt es sich um hausinterne SAP-Berater*innen, unabhängig ihrer Erfahrung und ihres Alters. Dadurch fallen alle Angestellten der Beratungsabteilung des Unternehmens in die Zielgruppe, welche sich auf eine Anzahl von 25 Personen beläuft.

1.2.3.3 Beschreibung der Anwendungsfälle

Bei der Umsetzung der Anwendungsfälle sollte auf sich wiederholende Prozesse abgezielt werden, welche derzeit noch von den Anwendern*innen manuell ausgeführt werden. Anhand der Implementierung des Projektes werden diese Routinearbeiten automatisiert und dadurch entsprechende zeitliche Ressourcen freigesetzt. Diese zusätzlich verfügbaren Ressourcen können in weiterer Folge anderweitig genutzt werden und auf lange Sicht die Produktivität steigern.

Zur Erhebung der Anwendungsfälle erfolgte ein Brainstorming in Form einer E-Mail-Nachricht an die gesamte Zielgruppe. Anhand dieser Nachricht wurden alle Personen aufgefordert repetitive Aufgaben, welche täglich bei ihrer Tätigkeit anfallen, aufzulisten. Aus diesen Auflistungen wurden die am häufigsten genannten Anwendungsfälle ausgewählt, die in weiterer Folge umgesetzt wurden. Die Anzahl der Rückmeldungen belief sich auf 20 Personen. Basierend auf diesen Rückmeldungen wurden die nachfolgenden Anwendungsfälle (Abschnitt 1.2.3.3.1 – 1.2.3.3.6), welche nach ihrer genannten Häufigkeit absteigend aufgelistet werden, zur Umsetzung auserkoren.

1.2.3.3.1 Anmelden an Kundensystem

Durch den breiten Kundenstamm und den damit verbundenen diversen Systemlandschaften tritt im täglichen Arbeiten regelmäßig die Anforderung auf, dass entweder eine bestehende Sitzung benötigt wird oder eine neue Sitzung auf einem System erstellt werden soll. Im Zuge dessen muss zuerst durch alle offenen Fenster navigiert werden, um dann das gesuchte System zu finden. Wird keine Sitzung gefunden, erfolgt die Anmeldung via dem SAP Logon. Aufgrund der vielen Systeme verwenden alle Anwender*innen, zusätzlich zu der globalen, eine lokale Passwortdatenbank zur Verwaltung ihrer Passwörter. Bei einigen Kunden wird auf dem System ein einheitlicher Nutzer*in für die gesamte Abteilung verwendet, bei anderen Kunden wiederum stehen allen Beratern*innen eigene User zur Verfügung. Abhängig von den Spezifikationen bedarf es auch einer Änderung der Kennwörter nach bestimmten Zeitabschnitten, was es zusätzlich erschweren würde, sich die jeweiligen User-Passwort Kombinationen auswendig zu merken. Diese Faktoren führen dazu, dass, wie eingangs erwähnt, zusätzlich zur globalen auch eine lokale Passwortdatenbank benötigt wird.

Um eine Anmeldung über den Passwortmanager zu starten, erfolgt zuerst die Suche nach dem Systemeintrag. Bei erstmaligem Öffnen der Anwendung ist zudem die Eingabe eines Kennwortes erforderlich. In weiterer Folge müssen die Anmeldedaten händisch von den Benutzern*innen, aus dem globalen oder lokalen Passwortmanager ausgelesen werden. Nach Eingabe der Daten im Logon wird der Anmeldevorgang abgeschlossen und eine Sitzung eröffnet.

Durch die Umsetzung dieses Projektes soll dieser Prozess automatisiert werden. Es soll nach einer bestehenden Verbindung, die auf die eingegebenen Parameter zutrifft, gesucht und das Fenster in den Vordergrund gebracht werden. Wird keine Sitzung gefunden, erfolgt die Erstellung einer neuen Verbindung und die Anmeldung wird gestartet. Während dieses Vorganges sollen die Daten aus der jeweiligen Passwortdatenbank automatisiert ausgelesen werden. Damit gewährleistet werden kann, dass nur Benutzer*innen Zugriff auf die Systeme bekommen, welche auch über diesen verfügen dürfen, kann dies über das Berechtigungsmanagement gesteuert werden.

1.2.3.3.2 Abmelden an Kundensystem

Ähnlich wie im vorhergehenden Abschnitt gestaltet sich der Anwendungsfall des Abmeldens an einem System. In vielen Produktivumgebungen kann nur ein*e Benutzer*in zur selben Zeit mit dem gleichen User angemeldet sein. Somit kann es immer wieder vorkommen, dass man eine Benachrichtigung eines Kollegen erhält, sich vom System abzumelden. Durch die Vielzahl an offenen Sessions müsste somit durch alle Fenster navigiert werden.

Anhand des Chatbots soll dieser Vorgang vom*von der User*in losgelöst werden. Dabei sollen alle Sitzungen durchsucht und in weiterer Folge beendet werden.

1.2.3.3.3 Freigabe eines Transportauftrages

Eine SAP-Transportlandschaft sollte auf Empfehlung des Herstellers (SAP) aus einer Drei-System-Landschaft bestehen. Wie in Abbildung 2 zu erkennen, setzt sich diese aus folgenden Komponenten zusammen:

- Entwicklung (DEV) – In diesem System erfolgen alle Entwicklungen und das Customizing.
- Qualitätssicherung (QAS) – Sollte ein Abbild des Produktiv System darstellen. Alle Entwicklungen werden in diesem System getestet, um die Auswirkungen auf die Produktivumgebung sicherstellen zu können.
- Produktiv (PROD) – In diesem System können keine Änderungen durchgeführt werden. Alle Änderungen müssen aus dem QAS in das PROD transportiert werden.

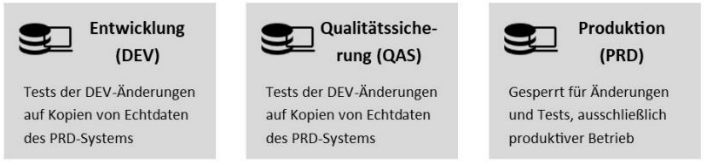


Abbildung 2: Empfohlene Drei-System-Landschaft (Braasch 2016, S. 575)

Als Schnittstelle zwischen diesen Systemen fungiert das Transport Management System (TMS). Anhand des TMS können Entwicklungen oder Customizing Änderungen über die Systeme hinweg transportiert werden. Bei dem Transport eines Auftrages muss die Transportschiene (DEV -> QAS -> PROD) ausnahmslos eingehalten werden, da kein System übersprungen werden kann.

Jegliche Änderungen im Entwicklungssystem erfordern einen existierenden Transportauftrag, welcher in weiterer Folge freigegeben und importiert wird. Die verschiedenen Auftragsarten sind bei der Umsetzung im Rahmen der Arbeit nicht von Bedeutung, jedoch die Zusammensetzung einer Auftragsnummer. Eine Auftragsnummer besteht immer aus einer 3-stelligen System ID, gefolgt vom Buchstaben K und sequenziellen Nummern im Raum von 900000 bis 999999. (Braasch 2016, S. 574–577)

Der Anwendungsfall zur Freigabe eines Transportauftrages tritt täglich, bei Entwickler*innen in Form von Programmierungen oder bei Beratern*innen in Form von Customizing Aufträgen, auf und stellt eine sich gleichbleibende, wiederholende Tätigkeit dar. Daher soll der Chatbot den Benutzern*innen Unterstützung bieten, indem Transportaufträge, basierend auf den Eingabeparametern, freigegeben werden, um diese in ein anderes System transportieren zu können.

1.2.3.3.4 Transportaufträge auflisten

Um in weiterer Folge einen Transportauftrag freigeben zu können, muss der*die Anwender*in die Transportnummer des Auftrages angeben. Damit im System nicht durch diverse Transaktionen, wie beispielsweise SE03 oder SE10, navigiert werden muss, soll der Bot abhängig von den Parametern die Transporte auflisten. Müssten die Benutzer*innen eigenständig im System nach den Aufträgen suchen, wäre die Implementierung des Anwendungsfalles zur Freigabe eines Transportauftrages aus Abschnitt 1.2.3.3.3 ein Mehraufwand, da in diesen Transkationen der Auftrag direkt freigegeben werden kann.

Der Chatbot soll basierend auf den erfassten Parametern, welche bei der Eingabe durch den*die Benutzer*in definiert werden, eine Liste der Aufträge ausgeben. Diese möglichen Parameter orientieren sich an jenen der Standardtransaktion. Im Falle der Transaktion wäre dies die Filterung nach freigegebenen oder änderbaren Aufträgen. Der*die Benutzer*in kann die Auftragsnummer dann innerhalb der Konversation mit dem Bot auslesen und verwenden, ohne dabei im System navigieren zu müssen.

1.2.3.3.5 Importieren eines Transportauftrages

Um einen Transportauftrag in einem Zielsystem wirksam zu machen, muss dieser importiert werden. In den meisten Fällen stellt sich die Situation so dar, dass der Auftrag aus dem Entwicklungssystem (DEV) freigegeben und im Anschluss direkt auf dem Testsystem (QAS) importiert wird, um diverse Tests durchzuführen.

Durch die Implementierung des Chatbots soll der Import eines Transportauftrages automatisiert werden. Dem Import eines Auftrages geht stets eine Freigabe voraus. Bei der Umsetzung ist daher wichtig, dass der Bot sich eine etwaige, vorhergehend freigegebene Transportnummer merkt und in weiterer Folge der Transport, basierend auf der Nummer, importiert werden kann. Es soll aber auch die Möglichkeit des Imports einer beliebigen Auftragsnummer gewährleistet werden.

1.2.3.3.6 Änderungen an Benutzer*in durchführen

Eine täglich anfallende Aufgabe erfordert das Ändern oder Zurücksetzen eines Kennwortes auf einem Kundensystem. Die Kontaktaufnahme des Kunden erfolgt meist in telefonischer Form. Zur Änderung eines Kennworts muss sich der*die Berater*in auf dem Kundensystem anmelden und ein neues Kennwort vergeben. Das neu vergebene Kennwort wird den Kunden telefonisch diktiert, was oft eine Fehlerquelle hinsichtlich Groß- und Kleinschreibung, Sonderzeichen oder Ähnlichem darstellt.

Eine weitere, oft anfallende Tätigkeit, umfasst das Entsperren und Sperren von Benutzern*innen. Der Ablauf verhält sich hierbei identisch wie im vorhergehenden Fall.

Durch die Unterstützung des Chatbots sollten telefonische Anliegen dieser Art schneller abgearbeitet werden. Der*die Berater*in erfasst alle erforderlichen Parameter der Kunden und gibt diese dem Bot weiter. Zusätzlich soll die Option geboten werden, eine E-Mail-Benachrichtigung zu versenden, um eine potenzielle Fehlerquelle innerhalb dieses Prozesses zu eliminieren.

Die detaillierten technischen Beschreibungen der Anwendungsfälle, mit möglichen Ausnahmefällen, können aus dem Anhang entnommen werden.

1.3 Zusammenfassung

In diesem Kapitel wurden einführend die Anfänge der Chatbots beleuchtet. Die Entwicklung von ELIZA als einfaches Dialogsystem, wird in der Geschichte als einer der ersten und wichtigsten Meilensteine betrachtet. Durch die durchgeführte Studie von bitkom wird zusätzlich die Bedeutung in der heutigen Zeit unterstrichen, in der belegt werden konnte, dass die Bot Technologie bereits in jedem vierten Unternehmen in Deutschland Anwendung findet. Das Anwendungsgebiet beschränkt sich dabei nicht nur auf den externen Einsatz mit Kundenkontakt, sondern findet auch intern eine breite Palette an Einsatzgebieten. Basierend auf den internen Anwendungsmöglichkeiten erfolgte die Hinführung zur Umsetzung im Rahmen dieser Arbeit. Es wurden die zu verwendenden SAP-Technologien erwähnt und begründet. Die detaillierte Beschreibung dieser ist aus Abschnitt 4 und 5 zu entnehmen. Basierend auf der Technologie und den Anwendungsmöglichkeiten konnte die Zielgruppe von hausinternen SAP-Berater*innen definiert werden. Abschließend erfolgte die Beschreibung und Begründung zur Auswahl der zu implementierenden Anwendungsfälle.

2 Aufbau und Architektur

Basierend auf den Informationen aus Abschnitt 1 wird in folgendem Kapitel der Terminus Chatbot genau definiert. Weiters wird auch auf die Architektur und die damit verbundene Gestaltung des Systems im Rahmen der Umsetzung eingegangen.

2.1 Defintion

Ein Chatbot fungiert als Anwendung die es den Benutzern*innen ermöglicht, mit einem maschinellen System in natürlicher Sprache zu kommunizieren. Als Eingabeformen werden entweder Text- oder Audioeingaben verwendet. Der Fokus der Aufgaben beschränkt sich meist auf vordefinierte Aufgaben mit gleichbleibendem Workflow. („Was ist ein Chatbot?“ 2021)

Die Begriffe Chatbot und Konversationsagent werden oft als Synonyme verwendet. Sie sind, abhängig von der Definition, jedoch nicht immer als identisch zu betrachten. Ein Chatbot verwendet oft ein vordefiniertes Set an Regeln. Bei Fragen oder Anfragen der Benutzer*innen versucht dieser durch die Anwendung der Regelsets auf die Eingaben, Antworten zu geben. (Pamela 2018)

Menschen kommunizieren interaktiv und innerhalb einer Konversation können die Anforderungen dadurch variieren. Daher können für identische Anforderungen unterschiedliche Prozesse entstehen. Somit folgen Menschen, anders als ein Chatbot, nicht immer dem gleichen Prozess. Ein Konversationsagent (engl.: CAI) ist in der Lage mit diesen Varianzen umgehen zu können. (Amelia 2020)

Folgende Szenarien sollen den Unterschied nochmals verdeutlichen. Ein Kunde möchte das billigste Hotel in der Stadt buchen und bittet den Konversationsagenten darum. Dieser fragt den Kunden nach gewünschten Extras wie einen Parkplatz, WLAN oder Ähnliches. Während des Prozesses bemerkt der Kunde, dass er einen Parkplatz braucht und der Agent dies berücksichtigen soll und sich das Datum um einen Tag nach hinten verschiebt. Da ein Chatbot auf einem regelbasierten Set arbeitet, können diese Anforderungen nicht verarbeitet werden oder der Prozess müsste nochmals neu gestartet werden mit der Eingabe der Daten. Der Agent merkt sich jedoch die vorherigen Parameter und kann auf die Änderung im Prozess reagieren und führt diesen mit den geänderten Parametern aus.

Wie bereits erwähnt, werden die Begriffe oft als Synonym gebraucht, da in der Praxis die Grenzen oft verschmelzen oder nicht klar zu definieren sind. Dies wird auch von Lotze folgendermaßen zitiert:

> *„Chatbot ist der klassische Begriff, der die eigentliche Funktion des „Plauderns“ in den Vordergrund rückt. Da der Begriff Chatbot, wohl der bekannteste Terminus für KIs mit Sprachfunktion ist, wird er heute außerhalb des wissenschaftlichen Diskurses übergeneralisiert gebraucht als Bezeichnung für alle Systeme, mit denen man „plaudern“ kann.“ (Lotze 2016, S. 28)*

Im Kontext dieser Arbeit wird das Wort Chatbot als Synonym für den Konversationsagenten verwendet. Somit werden die Begrifflichkeiten Chatbot und Konversationsagent als Synonyme betrachtet.

2.2 Architektur eines Chatbotsytems

Im Folgenden Abschnitt wird einführend auf die Architektur in den Anfängen der Chatbot Technologie eingegangen, um in weiterer Folge den heutigen Stand der Technik zu skizzieren. Weiterführend werden die Konzepte und deren Anwendung in der praktischen Umsetzung dargelegt.

2.2.1 Anfänge

Erste Entwicklungen von Chatbots, wie ELIZA, konnten nur triviale Konversationen ausführen. Es wurde mit Regelsets, Schlüsselwörtern, Mustererkennung (Pattern Matching) und internen Datenbanken gearbeitet. Der in Abschnitt 1 erwähnte Chatbot „ELIZA“ wurde nach diesem Prinzip implementiert. Die Architektur stellte sich damals in dieser Form dar:

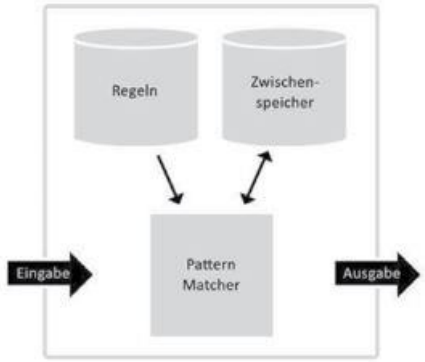


Abbildung 3: Chatbot Architektur in den Anfängen (Lotze 2016, S. 33)

Bei der Eingabe eines Users wurde nach Schlüsselwörtern gesucht. Auf die gefundenen Muster konnten dann die entsprechenden Regeln angewendet werden.

Bei der Mustererkennung wird versucht die Texteingabe mit Schlüsselwörtern zu vergleichen. Daher ist eine sehr hohe Fehleranfälligkeit gegeben, da jeder Mensch sich meist unterschiedlich artikuliert und dadurch eine große Breite an Wörtern abgedeckt werden muss. In jenen Fällen, in denen kein Schlüsselwort erkannt werden konnte, wird auf eine Standardantwort zurückgegriffen. Dadurch soll das Gespräch aufrechterhalten werden. Jedoch sind diese Antworten meist aus dem Kontext gerissen und geben dem Gespräch einen unnatürlichen Verlauf.

Im Regelspeicher (Regel in Abbildung 3) werden die Regelsets definiert, auf die bei bestimmten Schlüsselwort Kombinationen zurückgegriffen wird.

Der Zwischenspeicher dient dazu, Informationen abzulegen, auf die in weiterer Folge wieder zurückgegriffen werden kann. Anhand dieses Vorganges erfolgt die Rückgabe einer generierten Antwort. (Lotze 2016, S. 31–33)

2.2.2 Aufbau

Der Aufbau einer gegenwärtigen Chatbotarchitektur kann folgendermaßen dargestellt werden:

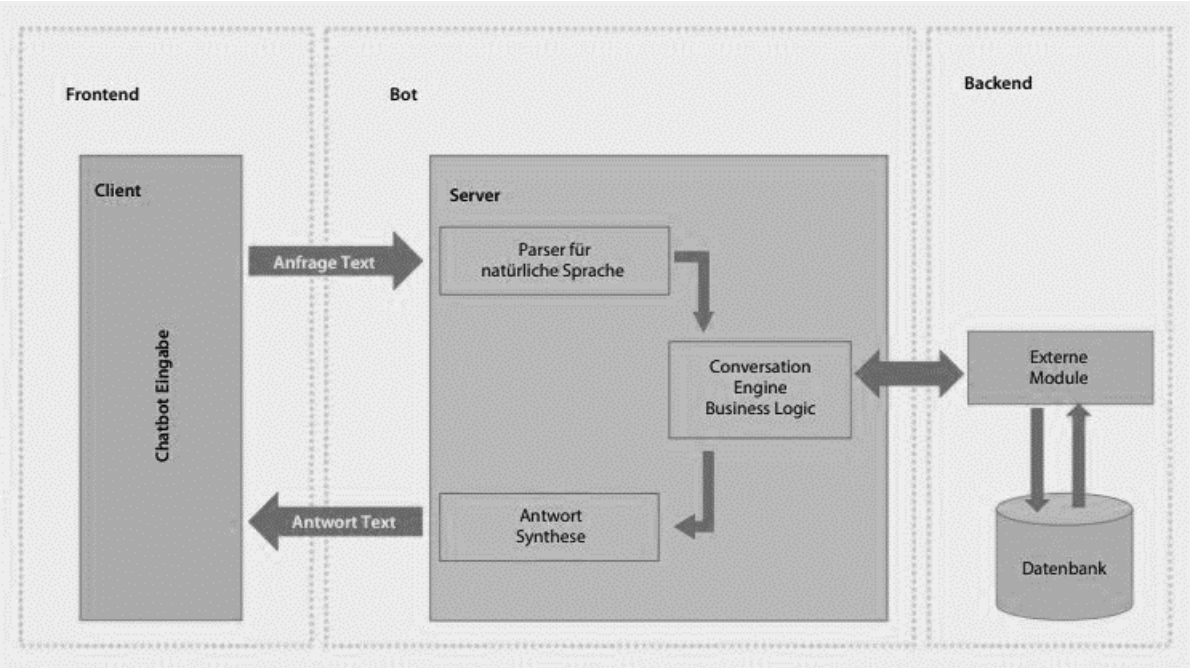


Abbildung 4: Architektur Chatbot (Stäcker; Stanoevska-Slabeva 2018, S. 40)

Der Prozess startet mit einer Eingabe des*der Benutzers*in. Diese kann entweder in Form von Sprache oder Text erfolgen.

Bei einer Spracheingabe erfolgt dann die Umwandlung von der natürlichen Sprache in eine Textform (Speech-to-Text). Bei einer textuellen Eingabe ist dies nicht von Nöten. Als Ergebnis liegt dem Bot nun die Absicht des*der Users*in in Form von Text vor. Dieser Text wird anhand eines Natural Language Processing (NLP) Verfahrens analysiert.

Nach der Analyse des vorliegenden Textes ist dem Bot nun die Absicht (Intent – siehe Abschnitt 2.2.3.1.1.1) des*der Benutzers*in bekannt. Basierend auf den besten Übereinstimmungen wird die Aufgabe, welche aus der Absicht hervorgeht, ausgeführt. Konnte bei der Analyse der Eingabe keine klare Absicht erkannt werden, erfolgt die Erfassung zusätzlicher Informationen.

Sind alle benötigten Informationen verfügbar, wird die gewünschte Aufgabe gestartet. Bei dieser kann es sich um einen API-Aufruf, einen Datenbankabruf oder den Aufruf einer externen Quelle handeln.

Nach der Ausführung erfolgt anhand von Natural Langauge Processing (NLG) die Generierung einer menschenähnlichen Antwort. Diese wird dann in gleicher Form zur Eingabe (Text oder Sprache) zurückgegeben. (Adamopoulou; Moussiades 2020)

2.2.3 Technologie

Der Aufbau eines Chatbots aus den Anfängen der Technologie findet sich in den heutigen Systemen zu Teilen wieder. Der größte Unterschied zu den ersten Architekturen, wie ELIZA, stellt das NLP, mit dem damit verbundenen Einsatz von maschinellem Lernen und Künstlicher Intelligenz, dar.

2.2.3.1 Natural Language Processing (NLP)

Ein wichtiger Bestandteil eines Chatbots ist das NLP. Anstatt des Einsatzes von einfachen Regelwerken mit Schlüsselwörtern (siehe Abschnitt 2.2.1) kommt hierbei zusätzlich maschinelles Lernen zum Einsatz. Dieses Konzept baut auf der Erkennung von bestimmten Schlüsselwörtern auf, anhand dieser dann einem Regelpfad gefolgt wird. Dies wird benötigt, um einerseits Anfragen der Benutzer*innen zu verstehen und andererseits dadurch entsprechende Antworten und Aufgaben schlussfolgern zu können. Wie in Abbildung 5 ersichtlich, werden diese beiden Teile innerhalb des NLP durch Natural Language Understanding (NLU) und Natural Language Generation (NLG) abgedeckt.

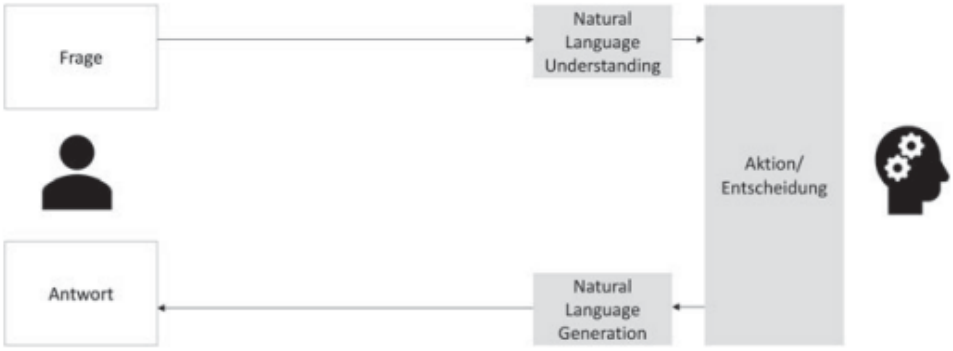


Abbildung 5: Natural Language Processing (Kohne u.a. 2020, S. 44)

Beim NLG wird versucht die menschliche Sprache des*der Benutzers*in zu verstehen, um daraus entsprechende Informationen zur Verfügung zu stellen. Anhand dieser Informationen wird eine Aktion oder Entscheidung getroffen. Basierend auf dieser erfolgt die Generierung einer Antwort mit Hilfe von NLG.

2.2.3.1.1 Natural Language Understanding (NLU)

Bei Konversationen können sich die Satzstellungen je nach Benutzergruppe unterscheiden. Die Herausforderung beim NLU liegt darin, dass aus den Eingaben des Menschen die richtigen Inhalte extrahiert werden. Diese werden anhand von Absichten (Intents) und Entitäten (Entities) strukturiert. (Kohne u.a. 2020, S. 42–45)

2.2.3.1.1.1 Absicht

Hierbei wird die Absicht des*der Benutzers*in beschrieben, welche durch die Eingabe von bestimmten Ausdrücken vermittelt werden soll. Bei der Umsetzung handelt es sich bei einer Absicht (engl.: Intent) um eine Sammlung von Aussagen, die unterschiedlich formuliert sind, aber die gleiche Aussage bzw. Absicht verfolgen. (SAP 2022, S. 27–29)

Neben den Anwendungsfällen aus Abschnitt 1.2.3 ist es ebenso notwendig, dass der Chatbot abseits dieser auch Funktionalitäten anbietet, wie beispielsweise die Auskunft für den*die Benutzer*in, welche Aufgaben abgedeckt werden. Folgendes Beispiel aus der Umsetzung der Arbeit soll dies verdeutlichen: Der*die Benutzer*in hat die Absicht Informationen darüber erhalten, bei welchen Aufgaben der Bot Unterstützung anbieten kann. Es ergab sich folgende Sammlung an Aussagen (Anm.: Hierbei handelt es sich um einen Ausschnitt)

- Ich brauche Hilfe/Unterstützung.
- Wobei kannst du mir helfen?
- Was kannst du?
- Wobei kannst du mich unterstützen?
- Hilf mir!
- Hilfe

Zusätzlich zu diesen Aussagen kann noch ein Schwellwert definiert werden, wie hoch die Übereinstimmung sein muss, damit die Absicht erkannt wird. Dieser Wert steuert wie genau eine Übereinstimmung von Wortlauten und Satzstellungen in der Eingabe im Vergleich zu der Sammlung in der Absicht sein muss, bevor die Absicht erkannt wird. Der Wert kann in einer Skala von 0 (niedrig) bis 100 (hoch) definiert werden.

2.2.3.1.1.2 Entität

Anhand einer Entität wird der Kontext innerhalb einer Absicht dargestellt. Die Absicht bildet dabei die Grundlage und die Entität definiert, welche Informationen daraus extrahiert werden. (Kohne u.a. 2020, S. 45)

In folgendem Szenario, welches ebenfalls aus der Umsetzung zum Anwendungsfall über das Ändern eines*r Anwenders*in (Abschnitt 1.2.3.3.6) entstammt, hat der*die Benutzer*in die Absicht ein Passwort zu ändern und eine Mail mit dem neuen Passwort zu senden. Die Aussage des Intent könnte folgendermaßen lauten: „Setze das Passwort für DOMINIK auf dem System ABC zurück und sende eine Mail mit dem neuen Passwort“. Die Absicht ist es, ein Passwort zurückzusetzen. Um dies in einen Kontext zu setzen, können folgende Entitäten erkannt werden:

- Setze Passwort zurück = Aktion über Zurücksetzen eines Passworts
- DOMINIK = Username
- sende Mail = Aktion eine Mail zu senden
- ABC = ID des Systems

2.2.3.2 Natural Language Generation (NLG)

NLG beschreibt die Generierung einer Antwort der Maschine in einer natürlichen Sprache. Die große Herausforderung darin liegt, dass versucht werden muss eine möglichst natürliche Antwort zu geben, um die Qualität der Konversation nicht zu senken. Für die Generierung müssen Daten konsolidiert und anhand von Mustern extrahiert werden (Automatisierte Insights). Zusätzlich bedarf es der Zusammenfassung von Texten, um aus den Kernaussagen eine Antwort zu generieren (Textzusammenfassung). Eine weitere Technik sind die Maschinenübersetzungen. Die Übersetzung erfolgt basierend auf einem Set an Regeln. Die Herausforderung bei dieser Form liegt in der Komplexität der Sätze. Oft ist eine wörtliche Übersetzung nicht möglich und Sätze müssen in der übersetzten Sprache restrukturiert werden, um diesen eine semantische Bedeutung geben zu können. Die genannten Methodiken stehen bei ihrer Verwendung in engem Zusammenhang mit dem maschinellen Lernen. (Kabel 2020, S. 57–58)

2.3 Gesamtarchitektur der Anwendung

Nach der Hinführung zur Problemstellung erfolgt nun die Beschreibung der Gesamtarchitektur der Anwendung. Dadurch soll ein Verständnis über den Gesamtaufbau geschaffen werden, bevor in nachfolgenden Kapiteln auf die einzelnen Komponenten eingegangen wird.

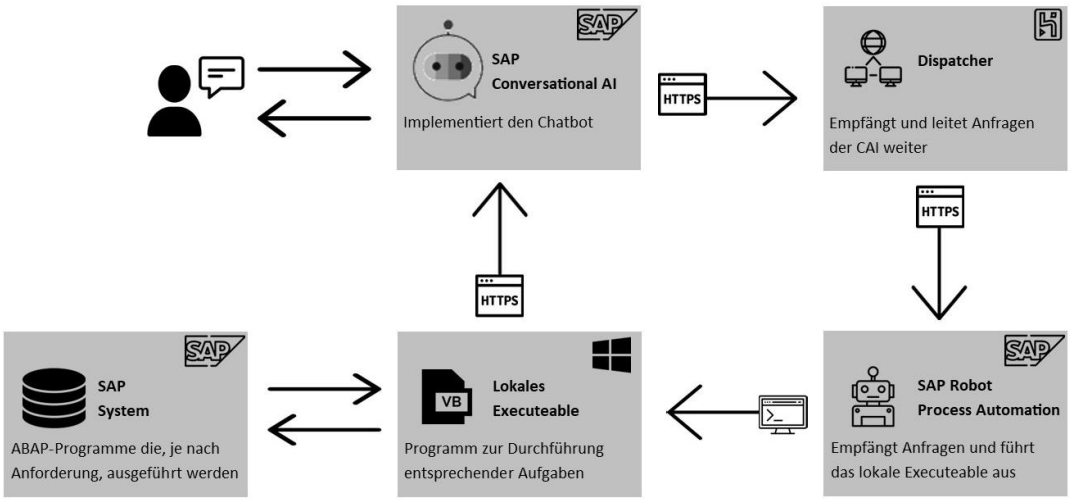


Abbildung 6: Architektur des Projektes (eigene Darstellung)

Wie aus Abbildung 6 erkennbar ist, wird die Aufgabe durch den*die Benutzer*in initiiert. Dieser kommuniziert mit dem Chatbot, der über die SAP Conversational AI (CAI) implementiert und zur Verfügung gestellt wird. Die CAI erfasst je nach Anforderung die erforderlichen und optionalen Parameter von den Benutzern*innen. Nach Erfassung dieser erfolgt eine HTTPS-Post-Anfrage an den Dispatcher. Der Dispatcher, welcher als Proxy fungiert, liest die benötigten Felder aus und leitet die Anfrage, basierend auf dem Wert im Feld *Name*, an die

entsprechende URL weiter. Die SAP Robot Process Automation (RPA), welche innerhalb der Business Technologie Plattform läuft, empfängt die Anfrage vom Dispatcher und startet die lokale ausführbare Datei auf dem Terminal des*der Benutzers*in mit den aufbereiteten Inputparametern. Anschließend führt das Programm die gewünschten Operationen lokal aus. Abhängig von den Parametern werden in weiterer Folge die ABAP-Programme im gewünschten System angesteuert und ausgeführt. Weiterführend wird das Ergebnis der ABAP-Programme ausgelesen und dem*der Benutzer*in über die CAI zurückgegeben.

Die ursprüngliche Idee lag darin, dass die Automatisierung vollständig im RPA implementiert werden sollte, da die Umsetzung mit den Technologien erfolgen sollte, welche bisher noch keine Anwendung fanden, um entsprechendes Wissen aufzubauen. Jedoch ergaben sich im Laufe der Umsetzung unterschiedliche Problematiken, welche nachfolgend aufgezeigt werden.

Eine Problematik war, dass sich die Anmeldedaten für die Systeme je nach Benutzer*in sehr stark unterscheiden. Auf einigen Systemen werden globale User verwendet, die auch in einer abteilungsweiten Passwortdatenbank abgelegt werden. Für andere Systeme werden personenspezifische User verwendet, die wiederum in einer lokalen Passwortdatenbank gesichert werden. Die Pflege und vollständige Umsetzung in Form von RPA wäre hierbei sehr umständlich und teilweise auch nicht machbar gewesen. Dadurch erfolgte die Auslagerung der Funktionalität in ein lokales Programm, in dem auch die Passwortdatenbank automatisiert ausgelesen wird.

Zudem existierte zum Zeitpunkt der Implementierung ein Programmfehler (auf Anbieterseite) bei der Verwendung der BAPI Aktivität, die benötigt wurde, um SAP-Systeme ansprechen zu können. Basierend darauf musste eine alternative Lösung erörtert werden, die in der Implementierung von ABAP-Programmen sowie der Verlagerung der Steuerungslogik in eine lokale Datei resultierte.

Ein weiterer Stichpunkt zur Entscheidung der Auslagerung in eine lokale Datei kann aus Kapitel 5.3 entnommen werden. In diesem Abschnitt werden auch die Verwendung und der Nutzen des Dispatchers detailliert erläutert. Weiteres wird die Architektur und praktische Umsetzung detailliert erläutert.

Die Erklärung zum Aufbau und zur Umsetzung der lokalen Datei sowie zu den ABAP Programmierungen findet sich in Kapitel 6 wieder. Die andere SAP-Komponente, CAI, wird in den Kapitel 4 detailliert beschrieben.

2.4 Zusammenfassung

In diesem Kapitel wurde einführend der Unterschied zwischen einem Chatbot und einem Konversationsagenten erklärt. Daraus ging hervor, dass die Grenzen in der Praxis oft verlaufend sind und man den Begriff Chatbot heute generalisiert für jegliche Konversationsagenten verwendet. In den Anfängen der Technologie wurde ausschließlich mit Regelwerken gearbeitet. In den heutigen Implementierungen spielt aber zusätzlich das maschinelle Lernen im Zusammenspiel mit NLP eine wichtige Rolle. Das NLP besteht aus dem NLU, dem Verstehen und Interpretieren der natürlichen Sprache und dem NLG, welches die Generierung einer natürlichen Antwort umfasst. Zusätzlich wurde die gesamte Architektur der Anwendung bestehend

aus der SAP Conversational AI, dem Dispatcher (Proxy), dem SAP RPA sowie der lokalen Datei, skizziert. Aufbauend darauf stützt sich auch die Begründung zur Auslagerung der Anwendungslogik in eine lokalen Skript Datei.

3 Konzepte

Bei der Kommunikation zwischen Mensch und Maschine gibt es unzählige Herausforderungen. Beginnend bei der Übersetzung der natürlichen Sprache in eine maschinell verständliche Sprache, über die Interpretation des Inputs im Zusammenhang mit dem Erkennen der Absicht des*der Benutzers*in bis hin zur Generierung einer Antwort in einer natürlichen Sprache. Jedoch umfasst eine Kommunikation weit mehr als nur diesen Prozess. Bei einer Konversation zwischen zwei menschlichen Gesprächspartnern spielen auch Faktoren wie Mimik, Tonlage oder Gestik eine wichtige Rolle. Denn basierend auf diesen Faktoren formuliert der Gesprächspartner seine Antwort.

Weiters fällt die Wortwahl auch anders aus, wenn eine Person mit einer Maschine kommuniziert anstatt mit einem menschlichen Gegenüber. Zusätzlich sind viele Personen auch voreingenommen bei der Mensch-Maschine-Kommunikation und verlieren schnell an Lust und Geduld. All diese Faktoren wurden bei der Entwicklung versucht zu berücksichtigen. Nachfolgend wird einführend auf Eckpunkte bei der Entwicklung eingegangen um zu zeigen, wo die Herausforderungen lagen. Weiterführend wird das Thema in Bezug auf die Erwartungen des Menschen an eine Maschine und deren Grenzen aufgegriffen.

3.1 Planung eines Chatbot

Wie bereits einführend erwähnt, spielen einige Faktoren bei der Kommunikation eine wichtige Rolle. In diesem Abschnitt wird versucht, auf die für die Umsetzung essenziellen Konzepte einzugehen und die praktische Umsetzung derer zu erläutern.

3.1.1 Interaktion

Ein erster wichtiger Bestandteil stellt die Interaktion mit den Benutzern*innen dar. Unabhängig von der Eingabeart spielt die Emotion eine wichtige Rolle. Das Erkennen von Emotionen und die Reaktion darauf beschränkt sich nicht nur auf die Entwicklung von Chatbots. Diese Thematik ist für jegliche Anwendungen relevant, in der ein Mensch mit einer Maschine kommuniziert und wird hierzu im Feld von Affective Computing untersucht. Beim Affective Computing wird versucht anhand von erfassten Daten (Stimme, Körpersprache und Gesichtszügen) menschliche Emotionen zu messen und dadurch die Interaktion zwischen Mensch und Maschine zu verbessern. (Rukavina u.a. 2016)

Aus einer Untersuchung hinsichtlich des Affective Computing ging hervor, dass eine zu hohe Reaktionsfreudigkeit auf Emotionen bei den Nutzern*innen ein Unwohlsein und in weiterer Folge auch Misstrauen hervorruft. Im Kontext von Chatbots kann eine zu hohe Emotionalisierung das Gefühl von Überwachung und Kontrolle vermitteln, was in weiterer Folge die Akzeptanz bei den Anwendern*innen stark beeinträchtigt. (Benke; Gnewuch; Maedche 2022)

Die Interaktion der Benutzer*innen mit einem Chatbot kann auf zwei unterschiedliche Arten erfolgen. Zum einen textuell und zum anderen sprachlich. Je nach Art der Eingabe ergeben sich daher unterschiedliche Faktoren, die zur Bewertung von Emotionen herangezogen werden.

Die Eingabe in Form von Text wird grundsätzlich als die einfachste Form der Erfassung angesehen, da hierbei nur der Textkanal analysiert wird. Faktoren wie Körpersprache, Stimmfarbe oder Gestik können nicht erfasst werden. Jedoch führt diese limitierte Verfügbarkeit an Daten dazu, dass eine der größten Herausforderungen darin liegt, eine aussagekräftige Antwort zu generieren. Um den Antworten eine möglichst hohe Aussagekraft geben zu können, fließen im besten Fall auch Faktoren wie der aktuelle Kontext des Gesprächs, sowie Emotionen der Gesprächspartner mit ein. Bei Generierung von Antworten kommt NLG aus Abschnitt 2.2.3.2 zum Einsatz.

Bei einer sprachlichen Eingabe sollte die Antwort des Bots in gleicher Form erfolgen. Die Stimmfarbe macht einen nicht unerheblichen Teil der Persönlichkeit des Bots aus. Eine klar erkennbare Roboterstimme hat einen anderen Einfluss auf die Benutzer*innen als die Stimme eines Menschen. Bei der Eingabe können zudem Faktoren wie die Stimmlage, Atemfrequenz oder auch Lautstärke erfasst werden. Dadurch fließen, zusätzlich zu den Daten von der textuellen Analyse, weitere Informationen in die Generierung einer Antwort mit ein. (Mensio; Rizzo; Morisio 2018)

3.1.2 Zweck

Eine im Jahr 2020 durchgeführten Studie ergab, dass 64% der 250 Befragten die Steigerung der Effizienz als größten Mehrwert in der Anwendung von Chatbots sehen. (Hundertmark 2020)

Um die Effizienz der Anwender*innen durch die Verwendung von Chatbots steigern zu können, sollte der Zweck vor der Einführung genau definiert werden. Es darf nicht um jeden Preis darauf abgezielt werden, möglichst viele Aufgaben zu automatisieren. Werden den Anwendern*innen zwar die Aufgaben durch Automatisierungen abgenommen, ist eine Erledigung dieser aber umständlicher als bisher, stößt der Chatbot auf sehr wenig Akzeptanz und findet in weiterer Folge wenig Verwendung. Der Mehrwert der Verwendung muss klar gegeben sein. Dies steht in engem Zusammenhang mit den Erwartungen und Grenzen, welche in Abschnitt 3.2 beleuchtet werden.

Bei der Definition des Zwecks bzw. Nutzens kann zwischen aufgabenorientiert und nicht-aufgabenorientiert unterschieden werden. Bei einem aufgabenorientierten Bot wird das Problem klar definiert, wie beispielsweise eine Information zu einem Fahrplan zu beschaffen. Die nicht-aufgabenorientierten Chatbots sind hauptsächlich auf die Konversation mit einem*einer Benutzer*in ausgelegt. In diese Kategorie würde der eingangs erwähnte Bot ELIZA fallen. (Stucki; D'Onofrio; Portmann 2020, S. 6–7)

3.1.3 Persönlichkeit

Der Mensch neigt dazu, in verschiedenen Dingen menschliche Eigenschaften zu suchen. Hierbei spricht man von Anthropomorphismus. (Bravo 2019)

Daher ist es bei der Entwicklung wichtig, den Chatbot mit einer klaren Identität zu versehen. Die Identität hängt von unterschiedlichen Faktoren, wie dem Design der Oberfläche, der Stimmfarbe oder auch der Umgang mit Fehlern, ab. Die im Rahmen der Arbeit wichtigsten Faktoren werden nachfolgend erläutert.

Bereits durch das Design des User Interface (UI) bekommt der Bot eine erste Gestalt. Anhand des Designs kann beeinflusst werden, wie der Chatbot wahrgenommen wird. Die Möglichkeit zur Umsetzung von Designideen hängt jedoch stark von der verwendeten Plattform ab. Abhängig davon ist es meist nur möglich, die Persönlichkeit in Hinblick auf die Designkomponente mit einem Icon oder einem Bild zu steuern. Bei sprachgesteuerten Assistenten, wie Alexa, fällt das Design hinsichtlich der Aufmachung weg. (Kaiser; Buttkereit; Hagenauer 2019, S. 18–20)

Durch die bildliche Darstellung in Form eines Icons wird den Anwendern*innen bereits ein erster Eindruck zur Persönlichkeit vermittelt. Dieser wird bei Verwendung einer Stimme nochmals untermauert. Bei einer stimmlichen Repräsentation kann auch ein Avatar verwendet werden, der das Persönlichkeitsbild nochmals klarer hervorhebt. (Kabel 2020, S. 32–33)

Der Chatbot trägt in den meisten Fällen das Bild und die Werte des Unternehmens nach außen hin zu den Anwendern*innen. Die definierten Werte müssen daher entsprechend übersetzt werden und beeinflussen somit das Bild des Chatbot. Dies geschieht vor allem über die Ausdrucksweise und die Tonlage. Ein Chatbot, der in rechtlichen Fragen Beratung anbietet, kann durch Verwendung von Fachterminis einen sehr kompetenten Eindruck vermitteln und somit das Vertrauen in die Kanzlei stärken.

Bei der Tonalität spielen jedoch nicht nur die definierten Werte eine Rolle, sondern auch die definierte Zielgruppe. Ein Bot, der für eine Zielgruppe von Jugendlichen ausgelegt ist, verwendet mit Sicherheit einen unterschiedlichen Satzaufbau und eine andere Wortwahl als jener eines Versicherungsanbieters. (Kaiser; Buttkereit; Hagenauer 2019, S. 22–24)

3.1.4 Praktische Umsetzung der Konzepte

Die angeführten Konzepte hinsichtlich Interaktion, Zweck und Persönlichkeit, welche bei der Planung eines Chatbots eine wichtige Rolle spielen, werden nun nachfolgend für die Implementierung der Arbeit beleuchtet und aufgezeigt.

Bei der Art der Interaktion handelt es sich um textuell basierte Interaktion. Die Eingabe und die Antwort des Bots erfolgen in Form von Text. Das textbasierte Setup wurde daher ausgewählt, da die tägliche Anwendung im Großraumbüro, bei Besprechungen oder bei einem Kunden vor Ort stattfindet. Daher ist eine Kommunikation auf sprachlicher Basis in den Anwendungsfällen meist unpassend.

Die definierte Zielgruppe beschränkte sich auf hausinterne SAP-Berater*innen. Dadurch lag der Fokus nur zu einem kleinen Teil auf der Verarbeitung von Emotionen, da die effiziente Erfüllung der Aufgaben im Vordergrund stand. Darüber hinaus wurde kein sehr großes Spektrum an Funktionalitäten implementiert. Dies führte dazu, dass den Nutzern*innen die Funktionen bereits nach sehr wenigen Anwendungen bekannt waren und dadurch eine gleichbleibende Wortwahl gewählt wurde. Das Hauptaugenmerk lag somit auf der Erfassung von Parametern und Absichten und in weiterer Folge der effizienten Ausführung der Aufgaben. Es gilt jedoch zu erwähnen, dass in jeder Konversation Emotionen eine Rolle spielen und somit nicht gänzlich vernachlässigt werden sollten. Dadurch war es notwendig, dass der Bot eine einfache Form des Smalltalks unterstützen sollte. Dabei ging es um die Erkennung von negativen Emotionen (Wut, Trauer, Unzufriedenheit) anhand der Wortwahl. Bei der Reaktion auf jene

negative Emotionen wurde zwischen dem Kontext einer Aufgabenfunktionalität und dem Kontext unabhängig von den Funktionalitäten des Chatbots unterschieden.

Im Kontext der Aufgabenfunktionalität wurde darauf abgezielt, den Gemütszustand nach einer Ausführung eines Anwendungsfalles zu erfassen. Konnten etwaige negative Emotionen erkannt werden, versuchte der Bot den*die Nutzer*in nochmals auf die Hilfefunktionalität hinzuweisen.

Im Kontext, welcher sich nicht auf die Funktionalitäten des Chatbots bezieht, war das Ziel, auf einfach Fragen oder Aussagen zu dem Gemütszustand antworten zu können. Nach der Antwort versuchte der Chatbot die Konversation wieder in Richtung der Erledigung von Aufgaben zu lenken.

Die Implementierung erfolgte als aufgabenorientierter Chatbot. Die Positionierung ergab sich daraus, dass zum einen die Anwendungsfälle klar definiert und abgegrenzt werden konnten und zum anderen, dass die Steigerung der Effektivität im Vordergrund stand, welches als eines der Hauptziele definiert wurde. Die Abgrenzung der Aufgaben konnte dadurch erreicht werden, dass bei jedem Anwendungsfall ein Start- und Endpunkt sowie ein Set an Eingabeparametern definiert werden konnte.

Für jede Aufgabe wurde ein Startpunkt definiert, welcher sich dabei unabhängig vom Kontext darstellte. Dies ermöglicht es dem Bot, zu jedem Zeitpunkt der Konversation, losgelöst von vorherigen Aufgaben und Inputs, eine neue Absicht (Anforderung) zu erfassen.

Die Ausführung von aktuellen Aufgaben beeinflusste jedoch den Startzeitpunkt einer neuen Aufgabe und die damit verbundene Erfassung der Parameter. Befand sich der Bot noch innerhalb der Ausführung (lokale Datei) einer Aufgabe, wurde der Start einer anderen so lange blockiert, bis diese abgeschlossen war. Durch die Definition von Merkmalen, wann ein Anwendungsfall als abgeschlossen betrachtet wurde, konnte dieses Verhalten gesteuert werden. Diese Steuerung spielte eine wichtige Rolle für das Konzept aus Abschnitt 3.2.

Für jeden Anwendungsfall wurde zudem ein endliches Set an möglichen Inputparametern definiert. Dadurch konnten die Antwortmöglichkeiten auf dieses Set, je nach Anwendungsfall, beschränkt werden. Dies ermöglicht es, bereits vorab ein bestimmtes Muster der Konversation zeichnen zu können. Beispielsweise konnte bei der Freigabe eines Transportes davon ausgegangen werden, dass mit hoher Wahrscheinlichkeit anschließend ein Import gestartet wird.

Ein weiteres Hauptaugenmerk spielte die Akzeptanz, da der Chatbot Aufgaben übernimmt, die von den Mitarbeitern*innen meist schon über mehrere Jahre hinweg manuell ausgeführt wurden. Somit war es von Beginn an von großer Bedeutung, den Bot entsprechend als aufgabenorientiert zu positionieren und dem*der Benutzer*in aufzuzeigen, welche Möglichkeiten und welchen Mehrwert durch die Verwendung gegeben sind. Dieser sollen den Anwendern*innen auch dahin gehend beschrieben werden, dass die bisherigen, mühsamen Aufgaben durch den Bot erledigt werden und die zusätzlichen Ressourcen anderweitig verwendet werden können.

Wie in Abschnitt 3.1.3 erwähnt, spielt die Zielgruppe hinsichtlich der Persönlichkeit eine wichtige Rolle. Da die Zielgruppe die hausinternen Berater*innen war, wurde auch die Namensgebung gemäß der CI des Unternehmens gestaltet. Der Bot bekam den Namen „proBot“. Dieser

setzt sich zusammen aus dem ersten Teil der Firmennamens proTask und aus einem Wort, das das Produkt spezifizieren soll – „Bot“.

Ebenfalls wurde das Icon als Instrument zur Schaffung einer Persönlichkeit erwähnt. Da die Bereitstellung als alleinstehende Anwendung über die CAI erfolgte, war hier nur bedingt die Möglichkeit vorhanden, eigene Designideen umzusetzen. Das in Abbildung 7 dargestellte Logo soll zur Persönlichkeit des proBot beitragen. Im Schriftzug findet sich das Firmenlogo wieder. Dies wird von einem Robotericon umrandet, welches einen Chatbot suggerieren soll. Die Farben sind ebenfalls gänzlich in der CI des Unternehmens gehalten.

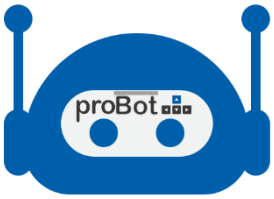


Abbildung 7: Icon „der proBot“ (eigene Darstellung)

Die Tatsache, dass die Benutzer*innen die Aufgaben zuvor manuell ausgeführt haben, hatte auch einen Einfluss auf die Ausdrucksweise des proBot. Hierbei lag der Fokus darauf, in kurzen Sätzen und möglichst unter Verwendung von Fachterminis zu interagieren.

3.2 Erwartungsmanagement und Grenzen

Eine weitere Rolle bei der Umsetzung spielten die Erwartungen und Grenzen. Wie bereits in Abschnitt 1.2.3 erwähnt, sollte darauf abgezielt werden, geeignete Prozesse zu automatisieren und somit die Akzeptanz der Anwender*innen zu maximieren. Die Akzeptanz wird vor allem durch die Erwartungen beeinflusst.

3.2.1 Erwartungen

Die Kommunikationsart zwischen Mensch und Maschine unterscheidet sich sehr stark von jener zwischen zwei menschlichen Gesprächspartnern. In einer Studie von Hill, Ford und Farreras wurde untersucht, wie sich die Erwartungen einer menschlichen Konversation in der Kommunikation mit einem Chatbot wiederfinden. Es wurde gezielt untersucht, ob Menschen anders kommunizieren, wenn ihnen bewusst gemacht wird, dass es sich bei ihrem Gesprächspartner um einen Chatbot handelt. Im Rahmen dieser Untersuchung wurden 100 zufällige Konversationen in Form von Textnachrichten (SMS) mit 100 zufälligen Konversationen des Cleverbot (Anm.: ein Chatbot) verglichen.

Als eines der Ergebnisse der Untersuchungen konnte festgestellt werden, dass die Anzahl der Wörter pro Nachricht in der Konversation mit dem Chatbot geringer war als in der Konversation mit dem menschlichen Gesprächspartner. Jedoch war die Anzahl der Nachrichten doppelt so hoch, im Vergleich zur Kommunikation mit dem menschlichen Gegenüber.

Daraus resultierte, dass die Testpersonen in der Kommunikation mit dem Chatbot vorsichtiger agierten, da sich ihre Erwartungen stark unterschieden. Die Testpersonen haben, anders als mit ihren menschlichen Gesprächspartnern, keine offensichtlichen Gemeinsamkeiten, über die ein Small Talk geführt werden könnte. Dies führt dazu, dass der Chatbot sehr eingeschränkt in der Thematik agieren kann eine Konversation aufrecht zu erhalten und die gesamte Breite einer Konversation simulieren zu können. (Hill; Ford; Farreras 2015)

Basierend auf dieser Studie ist das Erwartungsmanagement ein essenzieller Faktor hinsichtlich der Benutzerbarkeit und Akzeptanz. Können die Benutzer*innen weder eine Konversation mit dem Chatbot führen (Unsicherheit), noch sind ihnen die Funktionen oder der Nutzen klar, wird auf die Verwendung verzichtet.

In einer weiteren Studie von Zamora wurden die Wahrnehmungen und Erwartungen von Benutzern*innen gegenüber Chatbots in Form einer App untersucht. Bei der Befragung wurden 54 Teilnehmer befragt. Diese setzten sich jeweils aus 28 indischen und 26 amerikanischen Teilnehmern zusammen, um eine kulturübergreifende Gruppe abzudecken. Die Kernpunkte dieser Erhebung unterteilten sich in das Verstehen der Erwartungen der Benutzer*innen, die Eingabepräferenzen und das Identifizieren der Domänen, in denen eine Nutzung einen Mehrwert generiert. Da sich dieser Abschnitt mit dem Erwartungsmanagement befasst, liegt das Hauptaugenmerk auch in den Ergebnissen der Untersuchung hinsichtlich dieser. Dazu mussten die Befragten ihre Zufriedenheit für die Verwendung zu folgenden Punkten evaluieren:

- Wie einfach konnte eine Konversation gestartet werden konnte?
- Wie gut hat der Bot meine Eingaben verstanden?
- Wie war die Qualität der Ergebnisse?
- Wie relevant war das Ergebnis für meine Absicht?

Basierend auf diesen Punkten gaben 27 Teilnehmer*innen an, dass sie keine zukünftige Verwendung planen, demgegenüber standen 22 Teilnehmer*innen die eine zukünftige Verwendung in Betracht ziehen. Hauptgründe dafür lagen vor allem darin, dass die Antworten zu lange dauerten oder dem*der Benutzer*in Benachrichtigungen angezeigt wurden, die außerhalb des Kontextes lagen. Weiters gaben einige auch an, dass sie es bevorzugen, so weiter machen zu wollen wie bisher, da der Chatbot das Lösen der Aufgaben eher umständlicher macht, als vereinfacht.

Außerdem mussten die Befragten Personen in drei Wörtern einen „guten“ Chatbot beschreiben. Als Ergebnis wurden folgenden Begriffe genannt: performant (58-mal), klug (40-mal), nahtlos (20-mal) und persönlich (16-mal). (Zamora 2017)

3.2.2 Grenzen

Um etwaigen Enttäuschungen oder nicht erfüllten Erwartungen vorzubeugen, bedarf es bereits zu Beginn einer Positionierung des Chatbots. Den Benutzern*innen muss erläutert werden, was der Zweck des Chatbots ist. Beispielsweise kann ein Chatbot, der über Aktienkurse informiert, keinen Wetterbericht zurückgeben. Eine Methodik, um die Erwartungen zu steuern, ist das Konzept von „Fail gently“. Bei Unverständnis über die Anforderung versucht der Bot

den*die Benutzer*in zum eigentlichen Problem zurückzuführen und die Anforderungen nochmals zu erfassen. (Kabel 2020, S. 145–146)

Diese Thematik betrifft die Grenzen des Chatbot. Je nach Konzeption des Chatbots stößt dieser, abhängig von der Datengrundlage, inhaltlich schnell an Grenzen, da in der Konzeptionsphase versucht wird, Gespräche bereits vorab abzubilden und basierend darauf die Daten zu sammeln. Die Abbildung erfolgt hierbei dadurch, dass sich im Vorhinein Gedanken über die möglichen Userinputs gemacht werden. Daher muss den Benutzern*innen klar suggeriert werden, ab wann eine Grenze der Ausführbarkeit auftritt, bevor diese die die Geduld verlieren. Je komplexer die Aufgaben werden und umso weiter sie über das Standardrepertoire hinausgehen, desto schwieriger wird die Erfassung der Anforderungen für den Chatbot. (Kaiser; Buttkereit; Hagenauer 2019, S. 5)

3.2.3 Design Thinking

Als ein Instrument zur Konzeption von Chatbots kann die Methodik des Design Thinking angewendet werden. Bei diesem Ansatz erfolgt eine sehr starke Nutzerzentrierung, bei der die Lösungen aus Sicht des Kunden erarbeitet werden. Dieser Prozess besteht aus sechs Phasen:

1. Problem und Kontext -> In diesem Schritt versucht man die Probleme, welche der*die Benutzer*in hat zu verstehen.
2. Beobachten -> wie der*die Benutzer*in das Problem bisher löst
3. Perspektiven -> Das Problem wird nun aus Sicht des*der Benutzers*in betrachtet, um in späterer Folge einen Mehrwert gewährleisten zu können.
4. Ideen zur Lösung -> Entwickeln von Ideen und Lösungen um das bisherige Problem des*der Benutzers*in zu lösen
5. Prototyp entwickeln -> Eine erste Entwicklung eines Proof-of-concept um darstellen zu können, wie die Lösung aussehen kann
6. Feedback durch Tests -> Durchführung von Tests zur Optimierung (Kaiser; Buttkereit; Hagenauer 2019, S. 6)

3.2.4 Praktische Umsetzung Erwartungen und Grenzen

Ein wichtiger Faktor bei der Umsetzung lag darin, den Nutzern*innen bereits vorab zu signalisieren, dass die Kommunikation mit einem Chatbot erfolgt und nicht mit einem menschlichen Assistenten. Dies wurde zum einen durch den Namen proBot und das Icon aus Abschnitt 3.1.4 signalisiert und zum anderen durch die initiale Vorstellung, in der sich der Chatbot auch deutlich als Bot erkennbar macht. Zusätzlich wird dem*der Benutzer*in bei der Begrüßung bereits angeboten, mehr über die Funktionalitäten zu erfahren. Eine erstmalige Konversation könnte dann folgendermaßen aussehen:

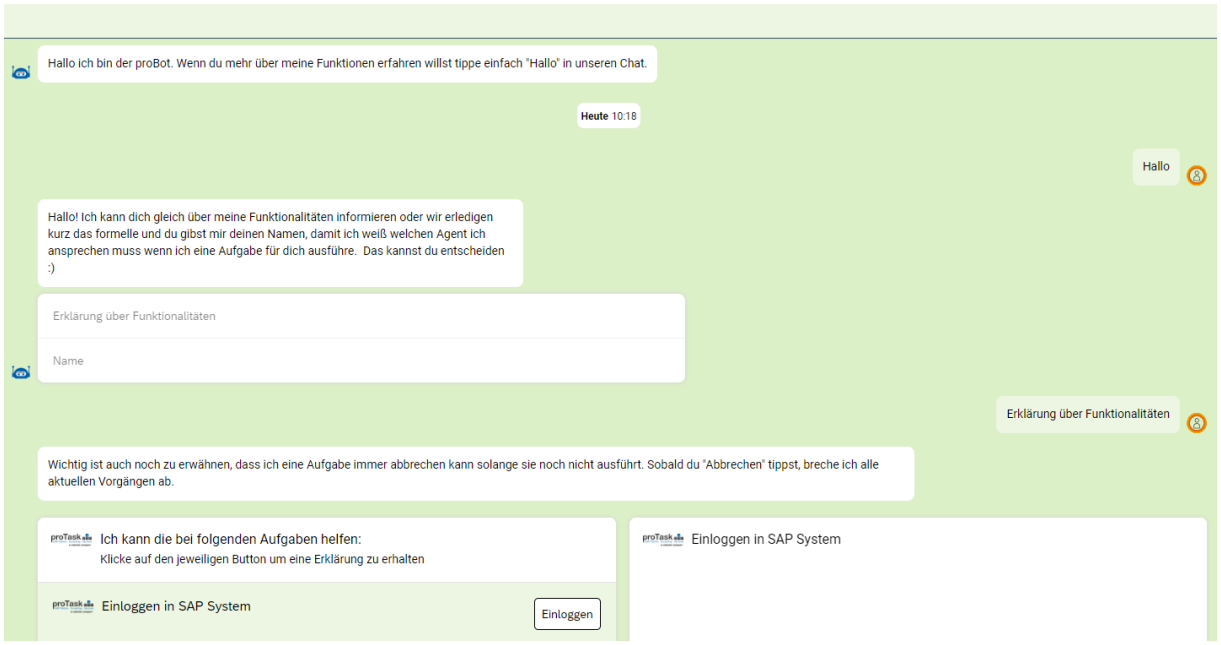


Abbildung 8: Beispielhafte Konversation mit dem proBot (eigene Darstellung)

Wie aus Abbildung 8 ersichtlich, stellt sich der Chatbot mit seinem Namen vor und bietet dem*der Benutzer*in sofort an, seine Funktionalitäten zu erklären. Dadurch soll dem*der Benutzer*in direkt suggeriert werden, bei welchen Aufgaben der Bot eine Arbeitserleichterung darstellen kann. In weiterer Folge werden über einen Klick auf den Button „Erklärung über Funktionalitäten“ alle Aufgaben aufgelistet, die der proBot erledigen kann. Um eine detaillierte Erklärung zu einer Funktionalität zu erhalten, kann ebenfalls wieder auf die Schaltfläche, in Abbildung 8 „Einloggen“, geklickt werden.

Somit können zwei Kernpunkte hinsichtlich des Erwartungsmanagements ausgemacht werden. Als erster Punkt kann die Erklärung der Funktionalitäten angeführt werden. Als zweiter Kernpunkt sind die Buttons von sehr großer Bedeutung, da diese vor allem Eingabefehler des*der Benutzers*in vorbeugen, welche zu Unverständlichkeit und in weiterer Folge zu Unzufriedenheit führen können. Zusätzlich kann durch dieses Element das Gespräch in bestimmte Richtungen gelenkt werden, da basierend auf den Rückmeldungen des Bots, die Antworten des*der Anwenders*innen ein stückweit vorhergesagt werden können. Folgendes Beispiel soll dies verdeutlichen:

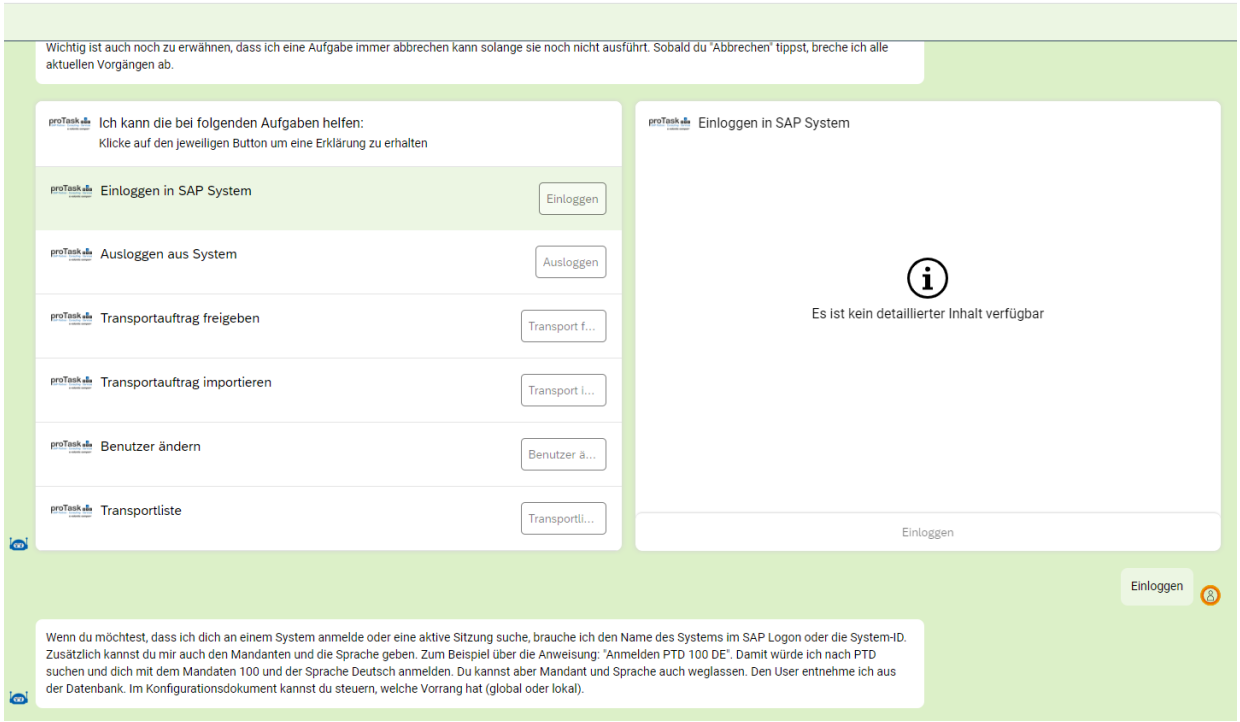


Abbildung 9: Erklärung einer Funktionalität des proBot (eigene Darstellung)

Der*die Benutzer*in hat durch den Klick auf die Schaltfläche „Einloggen“ den Anwendungsfall des Einloggen in ein SAP-System zur detaillierten Erklärung ausgewählt. Der Chatbot beschreibt nun die möglichen Parameter, die für diese Aufgabe benötigt werden. Zusätzlich gibt er dem*der Nutzer*in einen möglichen Eingabevorschlag, um die Aufgabe zu starten. Dadurch kann das Gespräch gesteuert werden. Um jedoch nicht das Gefühl des Kontrollverlustes zu vermitteln, werden immer wieder Sätze verwendet, die hervorheben, dass der*die Anwender*in als letzte Instanz die Entscheidung trifft, welche Parameter verwendet werden und dadurch die Entscheidungsgewalt über die Ausführung der Aufgaben hat. Dieses Stilmittel wird auch bei der Begrüßung verwendet, bei der selbständig entschieden werden kann, ob zuerst die Formalität mit dem Namen erledigt wird oder zuerst die Information zu den Funktionalitäten erfolgt.

Da der proBot als aufgabenorientierter Bot positioniert wird, hat die Komponente des Smalltalks in der Phase des Proof-of-concept keine essenzielle Bedeutung. Es wird auf Fragen hinsichtlich des Gemütszustandes reagiert, jedoch wird dann versucht, das Gespräch wieder in Richtung der Aufgaben zu lenken, bei welchen eine Unterstützung angeboten werden kann. Für eine produktive Verwendung ist es nötig, diese Komponente noch intensiver zu berücksichtigen, um das Vertrauen der Anwender*innen zu stärken. Eine bewusste Steuerung auf eine emotionale Frage könnte wie ein Abbildung 10 erfolgen.

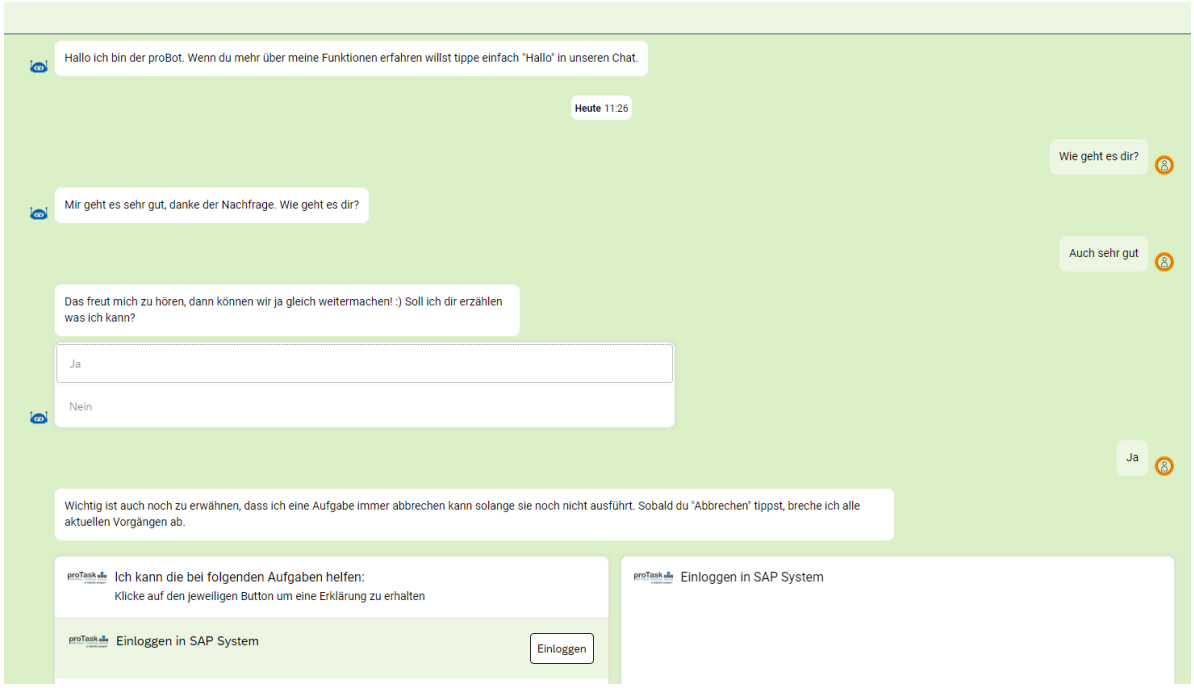


Abbildung 10: Steuerung der Konversation (eigene Darstellung)

Wie in der Studie von Zamora erwähnt, werden Anwender*innen den Chatbot zukünftig nicht verwenden, wenn dieser das Erledigen der Aufgaben umständlicher gestaltet. Aufgrund dieser Aussagen wurden Aufgaben ausgewählt, die immer dem gleichen Ablauf folgen. Mehr dazu kann aus Abschnitt 3.2.1 entnommen werden.

Es kann auch immer wieder vorkommen, dass der Chatbot Anforderungen der Nutzer*innen nicht verstehen kann. Dies kann verschiedene Gründe haben. Es kann daran liegen, dass die Datenbasis mit den Absichten und Entitäten nicht umfangreich genug gestaltet wurde oder die Benutzer*innen eine Anforderung stellen, die außerhalb der Möglichkeiten des Bots liegen. Hierbei ist es wichtig, basierend auf dem Fail Gently Prinzip, darauf hinzuweisen, dass bei dieser Aufgabe keine Unterstützung möglich ist und zu fragen, ob nochmals die Funktionalitäten aufgelistet werden sollten. Somit kann eine klare Positionierung erfolgen, was im Bereich der Möglichkeiten liegt und was außerhalb der Kompetenzen fällt.

Die Tatsache, dass performant und klug als Eigenschaften für einen Chatbot am öftesten genannt wurden, zeigt, dass der Chatbot die Aufgaben mit ähnlicher Geschwindigkeit und gleicher Qualität lösen sollte, wie diese zuvor durch den*die Nutzer*in ausgeführt wurden. Die Performanz ist stark an die Implementierung der Plattform geknüpft. In dieser Thematik ist der technologische Spielraum bei der Umsetzung sehr stark beschränkt und kann dadurch nicht durch den*die Entwickler*in des Chatbots auf der Plattform beeinflusst werden.

Die Anforderung an den Bot smart zu sein, sollte nicht gleichbedeutend damit gesehen werden, dass die vollständige Arbeit des Denkens vom*von der Nutzer*in auf den Bot abgeschoben wird. Vielmehr wird darunter verstanden, dass der Bot den Anwender*in auf mögliche Erleichterungen hinweist und bei der Umsetzung unterstützt. Dies traf bei der Implementierung für den Anwendungsfall des Imports eines Transportes zu. Meist geht diesem Import eine Freigabe voraus. Der Chatbot merkt sich dann die Nummer und gibt dem*der Benutzer*in

nach Abschluss der Freigabe die Möglichkeit den Auftrag direkt zu importieren. Dafür wird lediglich die System-ID benötigt. Dadurch wird den Anwendern*innen der Aufwand, sich die Transportnummer zu merken, abgenommen und diese können dem Bot anhand von „Importiere den Auftrag nun auf HFX“ die Anweisung geben, den Auftrag auf dem System zu importieren.

Weiteres wird bei der Freigabe eines Transportauftrages lediglich die Transportnummer des Auftrages aus dem System benötigt. Basierend auf dieser liest der Bot dann die System-ID jenes Systems aus, auf der die Freigabe erfolgen soll. Die genaue Implementierung der Anwendungsfälle kann aus Kapitel 4 entnommen werden.

Als weiteres Instrument, um eine Akzeptanz der Nutzergruppe gewährleisten zu können, wurde nach dem beschriebenen Design Thinking Prozess gearbeitet. Nachfolgend wird in kurzen Sätzen, auf die einzelnen Phasen aus Abschnitt 3.2.3 eingegangen, um den Prozess von der Idee bis zur Umsetzung zu skizzieren:

1. Die Probleme der Nutzer*innen lagen darin, dass einfache Routinearbeiten im täglichen Arbeiten einen hohen Zeitaufwand beanspruchten und zudem den Arbeitsfluss unterbrachen.
2. Diese Arbeiten wurden bisher manuell vom Benutzer*in gelöst.
3. Aus Sicht der Benutzer*innen folgen die Aufgaben immer dem gleichen Workflow.
4. Als Lösung sollen die Aufgaben automatisiert werden. Somit muss der*die Benutzer*in nur den Start der Aufgabe veranlassen und kann sich bis zur Beendigung anderen Tätigkeiten und Aufgaben widmen.
5. Ein Prototyp sollte im Rahmen dieser Arbeit implementiert werden.
6. Die Testmethodik sowie die Ergebnisse dieser Implementierung können aus Abschnitt 7 entnommen werden.

3.3 Zusammenfassung

In diesem Kapitel wurden die elementaren Konzepte zur Umsetzung des Chatbots beleuchtet. Hauptaugenmerk lag dabei auf der Planung und dem Erwartungsmanagement. Bei der Planung war es entscheidend die Interaktion der Zielgruppe entsprechend zu gestalten, sowie den Zweck und die Persönlichkeit, ebenfalls basierend auf der Zielgruppe, zu gestalten. Darüber hinaus spielte vor allem das Erwartungsmanagement eine wichtige Rolle, da Chatbots, welche die Erwartungen nicht erfüllen, keine Akzeptanz und Verwendung bei den Anwender*innen finden und somit keinen zusätzlichen Nutzen darstellen.

4 SAP Conversational AI

In diesem Kapitel erfolgt die Erläuterung der Architektur der SAP Conversational AI (SAP CAI). Die Verwendung der Plattform geht aus der Forschungsfrage hervor. Um einen Überblick dafür zu erhalten, muss vorab die Architektur verstanden werden, bevor in weiterer Folge die Implementierung der einzelnen Anwendungsfälle beschrieben wird. In Kapitel 2.2.3 wurden bereits eine Absicht (Intent) und Entität (Entitiy) beschrieben, was hier nochmals aufgegriffen und plattformspezifisch beschrieben wird. Um nachfolgende Architektur und Erklärung zu verstehen, ist es notwendig die beiden Begrifflichkeiten zu kennen. Bezugnehmend auf die Gesamtarchitektur in Abschnitt 2.3 dient die CAI als Schnittstelle zum*zu der Benutzer*in und erfasst die Anforderungen. Nach Erfassung dieser werden entsprechende Aktionen über die Robot Process Automtation (RPA - Kapitel 5) ausgeführt.

4.1 Architetkur SAP CAI

Bei der SAP CAI handelt es sich um eine von SAP zur Verfügung gestellte Plattform zur Entwicklung und Bereitstellung von Konversationsagenten. Bei der Erstellung sind keine Programmierkenntnisse erforderlich und wird somit folgerichtig auch als Low-Code-Tool platziert. Folgende Abbildung 11 zeigt die allgemeine Architektur der Plattform. Nachfolgend werden ebenfalls die einzelnen Bestandteile und ihre Funktion erklärt.

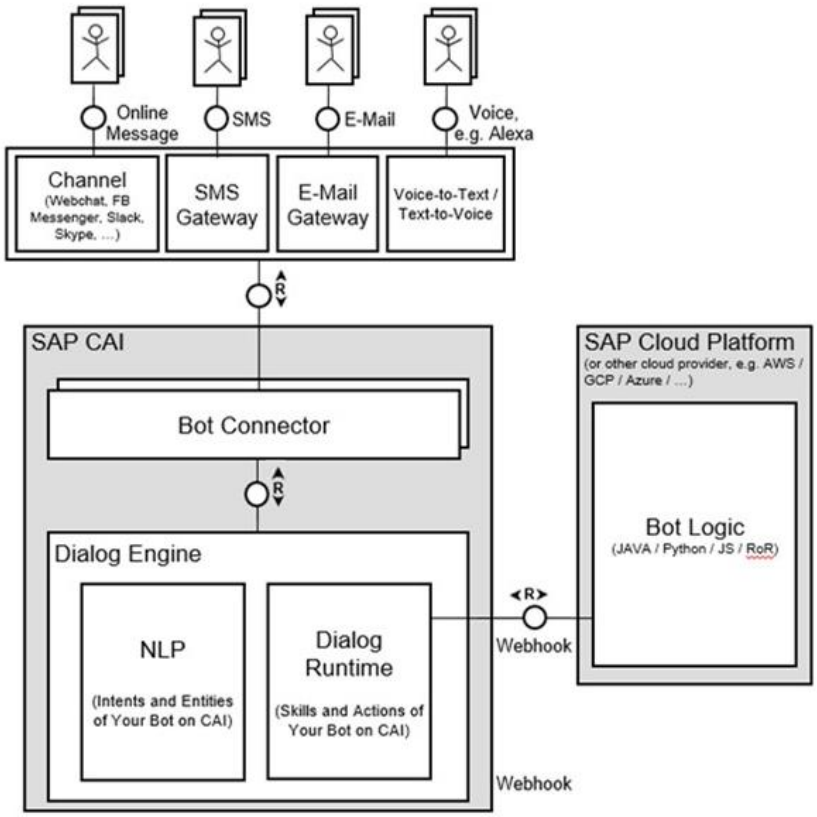


Abbildung 11: Architektur SAP CAI (Methukupally 2019)

4.1.1 Eingabe

Die Eingabe der Benutzer*innen können über verschiedene Schnittstellen erfolgen:

- Chatkanal -> Die Eingabe erfolgt online anhand eines Nachrichtenkanals
- SMS Gateway -> Die Eingabe erfolgt per SMS-Nachricht
- E-Mail -> Die Eingabe erfolgt per E-Mail
- Spracheingabe -> Die Eingabe erfolgt in gesprochener Form und wird in Text umgewandelt. (Methukupally 2019)

4.1.2 Bot Konnektor

Über den Bot Konnektor werden die verschiedenen Kanäle angebunden. Somit stellt dieser die Schnittstelle zum Nachrichtenaustausch mit der SAP CAI dar. Die ausgehenden Nachrichten werden dann entsprechend dem benötigten Format für den verbundenen Kanal formatiert, eingehende Nachrichten wiederum von einem kanalspezifischen Format in ein für die SAP CAI verständliches Format übersetzt. Alle Nachrichten, die ausgetauscht werden, laufen über den Konnektor. Hierbei wird eine POST Anfrage gestellt, die den Bottoken zur Authentifizierung beinhaltet. Es besteht auch die Möglichkeit, Nachrichten von „außerhalb“ zu senden. Für diese Methodik wird die API /messages verwendet. In der gesendeten POST Anfrage wird die Nachricht im Body definiert. (SAP 2022, S. 228–232)

4.1.3 NLP

Die Funktionalität von NLP wird in Abschnitt 2.2.3.1 erläutert. Bei dieser Komponente werden die Absichten und die Entitäten aus der Eingabe extrahiert. Basierend darauf können dann die entsprechenden Skills angesteuert werden. Die Übermittlung erfolgt via des Bot Konnektors, der die Nachricht in ein für CAI verständliches Format geparst hat.

4.1.4 Dialog Runtime (GUI)

Über die Dialog Runtime (GUI) wird der Konversationsfluss abgebildet. Dies wird auch als Low-Code-Tool platziert und bedarf demnach keiner Programmierung. In der GUI werden die Skills definiert. (Methukupally 2019)

4.1.5 Bot Logic

Die Botlogik dient als Middleware und ist zuständig für das Ausführen der Aufgaben und das Beschaffen der Daten, die an die CAI zurückgegeben werden. Dies wird über API-Aufrufe an das SAP RPA realisiert.

4.1.6 Ablauf

Nach Erklärung der einzelnen Komponenten in den vorherigen Abschnitten, werden nun ein beispielhafter Ablauf beschrieben und die Aufgaben der einzelnen Komponenten dargestellt.

1. Als ersten Schritt gibt der*die Anwender*in einen Ausdruck über einen der Kanäle ein. Die Form unterscheidet sich je nach Kanal.
2. Über eine HTTP Post Anfrage wird der Ausdruck an den Bot Konnektor übertragen. Im Bot Konnektor erfolgt das Parsen der Anfrage in ein, für die CAI, verständliches Format.
3. In weiterer Folge wird der geparste Ausdruck in der Dialog Engine analysiert. In erster Instanz versucht das NLP die Absichten und Entitäten zu erkennen und zu extrahieren. Danach ruft die Dialog Runtime die entsprechenden Aktionen auf, die aus den vorher extrahierten Elementen abgeleitet werden können. (Methukupally 2019)

4.2 Spezfikationen

In diesem Abschnitt werden, ergänzend zu den allgemeinen Erklärungen der Begriffe in Kapitel 2.2.3.1.1, die plattformspezifischen Besonderheiten der SAP CAI angeführt.

Eine Absicht (Intent) beschreibt, eine Aktion, welche basierend auf den Eingaben des*der Benutzers*in ausgeführt werden soll. In einem Intent werden alle Konstruktionen von Ausdrücken gesammelt, die verwendet werden könnten, um die gewünschte Absicht auszudrücken. Die Qualität zum Erkennen der Absicht hängt sehr stark von der Anzahl und der Komplexität der gesammelten Ausdrücke ab. Zusätzlich kann über die Strenge (strictness) gesteuert werden, wie genau die Eingaben mit den Ausdrücken übereinstimmen müssen, damit ein Intent erkannt wird. (SAP 2022a, S. 27–29)

Bei einer Entität handelt es sich um Schlüsselwörter, die aus einer Aussage extrahiert werden. Innerhalb von Entitäten kann zwischen drei Typen unterschieden werden:

- Freie Entität – hat keine vordefinierte Liste an Worten und lernt anhand des maschinellen Lernens. Dies wird dadurch realisiert, dass in den Ausdrücken innerhalb der Intents die jeweiligen Schlüsselwörter getagged werden, die auf die Entität zutreffen.
- Beschränkte Entität – besteht aus einer Liste von vordefinierten Schlüsselwörtern, die anhand der Strenge, welche regelt, ab welcher Genauigkeit eine Entität als erkannt angesehen wird, definiert wird.
- Reguläre Entität – über einen regulären Ausdruck kann definiert werden, welche Voraussetzungen ein Schlüsselwort erfüllen muss, dass es als Entität erkannt wird. (SAP 2022, S. 31–35)

Als Skill wird ein Gesprächsblock verstanden, welcher einen klaren Zweck in der Kommunikation mit den Anwendern*innen erfüllen soll. Ein Skill kann beispielsweise das Erklären von Funktionalitäten abbilden. (SAP 2022, S. 88)

Ein Auslöser (Trigger) ist eine Bedingung, die festlegt, wann ein Skill ausgeführt wird. Sind einem Skill keine Trigger zugewiesen, kann dieser nicht durch die Eingabe eines*einer Anwenders*in gestartet werden.

Eine Anforderung beschreibt die Voraussetzung (Requirement), die aus der Aussage der Eingabe extrahiert werden muss. Es wird zwischen verpflichtenden und optionalen Anforderungen unterschieden. Die extrahierten Werte werden unter dem definierten Variablennamen in den Speicher der Konversation geschrieben.

Damit eine Aktion (Action) ausgeführt werden kann, müssen zuvor alle verpflichtenden Anforderungen des Skill erfasst worden sein, welche innerhalb des Skills definiert wurden. Nachfolgend werden jene Aktionen aufgelistet, die für die Umsetzung der Arbeit relevant waren. Es gilt jedoch zu erwähnen, dass es eine zusätzliche Vielzahl an möglichen Optionen gibt:

- Eine Nachricht an die Benutzer*innen senden. Bei dieser Art stehen verschiedene Formen, wie eine einfache Textnachricht, eine Nachricht mit zusätzlichen Buttons (die einen Skill starten) oder auch eine Liste zur Verfügung.
- Eine Verbindung zu einem externen Service herstellen. Diese Art wurde benötigt für die API-Aufrufe und stellte somit die Schnittstelle zwischen CAI und RPA dar.
- Einen Fallbackchannel hinterlegen, bei dem auf einen menschlichen Gesprächspartner weitergeleitet werden kann.
- Einen anderen Skill aufrufen.
- Den Konversationsspeicher bearbeiten.

Um Daten während einer Konversation zu speichern und darauf zugreifen zu können, wird das Memory Objekt verwendet. Das Objekt ist vom Beginn, indem es als leerer Speicher initialisiert wird, bis zum Ende einer Konversation verfügbar. (SAP 2022a, S. 101–117)

Ein weiteres Konzept, das in der CAI Anwendung findet, ist Handlebars. Bei Handlebars handelt es sich um eine „templating language“. Hierbei wird der Ausdruck zwischen {{ und }} mit dem Wert der Variable ersetzt. Mit dieser Technik kann im API-Aufruf auf die Variablen im Speicher zugegriffen werden. (Katz 2021)

4.3 Implementierung der Anwendungsfälle

Basierend auf vorheriger Erklärung der Architektur und den Begrifflichkeiten, wird in diesem Abschnitt die Umsetzung auf Seiten der SAP Conversational AI behandelt. Es wird jeweils die Absicht (Intent) mit einigen Beispielen aus der Sammlung der Aussagen, sowie die jeweiligen Entitäten, welche eine Rolle für den Anwendungsfall spielen, aufgelistet.

4.3.1 Umsetzung

Bei allen Anwendungsfällen wird die System-ID benötigt, welche durch die Entität „systemid“ abgebildet wird. Diese Entität wurde als reguläre Entität umgesetzt, da es für die Form eine

klar definierte Namenskonvention gibt. Die System-ID startet immer mit einem Großbuchstaben, gefolgt von:

- einer Ziffer und einem Großbuchstaben
- von zwei Ziffern
- zwei Großbuchstaben

Basierend darauf konnte auch der reguläre Ausdruck definiert werden. Für jeden Anwendungsfall aus Abschnitt 1.2.3.3 wurde ein eigener Skill erstellt. Jeder Skill enthält einen Auslöser (Trigger). Der Auslöser ist bei jedem Anwendungsfall das Erkennen des Intent (Absicht).

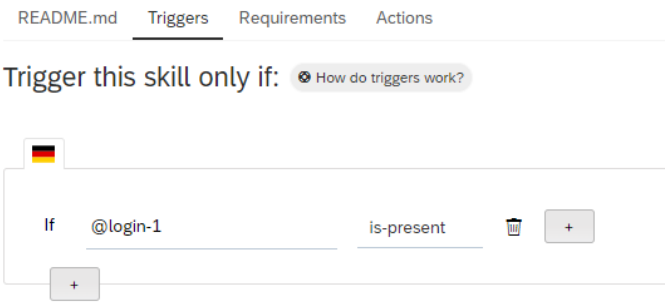


Abbildung 12: SAP CAI GUI - Trigger für den Skill Anmelden (eigene Darstellung)

In Abbildung 12 wird als Auslöser das Erkennen der Absicht *login-1* definiert. Eine Absicht (Intent) beinhaltet eine Sammlung von Ausdrücken. Für den Anwendungsfall des Anmeldens könnte diese wie folgt aussehen:

- Erstelle eine neue Session auf dem System PTD
- Melde mich auf PTD an
- Einloggen PTD
- Logge mich auf dem System PTD ein.
- Starte die Anmeldung auf PTD

Je umfangreicher die Ausdrücke gesammelt werden, desto besser können die Eingaben verarbeitet werden. Wird eine Absicht erkannt, startet der Skill. Beim Starten werden zuerst die Anforderungen (Requirements) geprüft. In den Anforderungen wird versucht, die Entitäten (Schlüsselwörter) aus der Aussage zu extrahieren und in den Speicher abzulegen.

Als notwendige Entität für alle Anwendungsfälle wird die System-ID benötigt. Wie in Abbildung 13 ersichtlich, wird nach der Entität „#systemid“ gesucht und unter dem Variablennamen „sysID“ im Speicher abgelegt. Fehlt der Parameter, wird eine Antwort definiert, in der nach der System-ID gefragt wird. Als sekundäre (optionale) Anforderungen kann der*die Benutzer*in noch den Mandanten und die Sprache definieren. Diese sind jedoch nicht zwingend erforderlich für eine Ausführung.

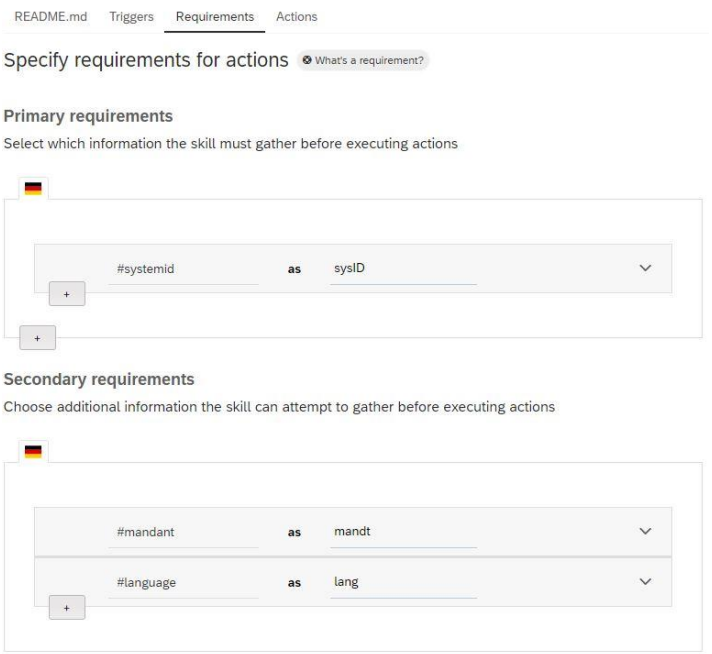


Abbildung 13: SAP CAI GUI - Anforderungen für den Skill Anmelden (eigene Darstellung)

Sind alle primären Anforderungen erfüllt, werden die Aktionen (Actions) gestartet. Für jede Aktion sind, abhängig von Anwendungsfall, verschiedene Konditionen definiert worden, die erfüllt werden mussten, bevor diese ausgeführt werden konnte. (siehe Abb. 14)

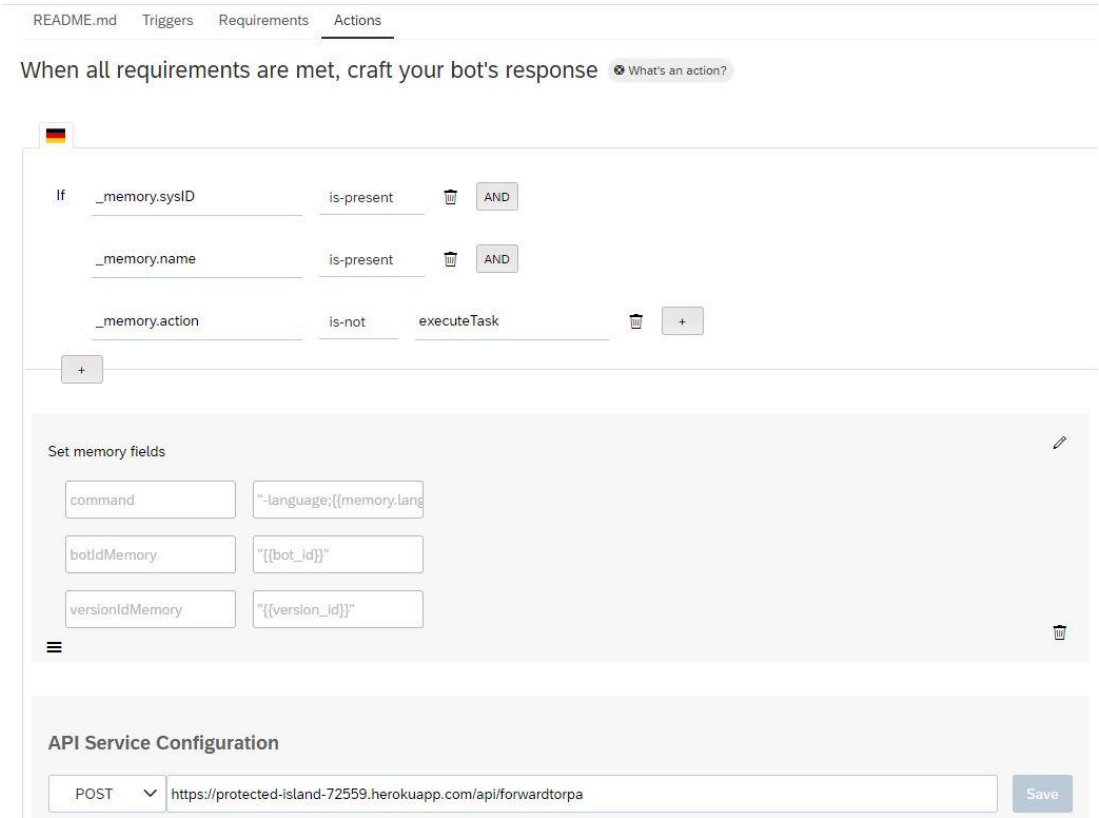


Abbildung 14: SAP CAI GUI - Bedingungen und Aktion für den Skill Anmelden (eigene Darstellung)

Nach Erfassung aller Anforderungen und Erfüllung der Konditionen wird ein API-Aufruf abgesetzt. Zuvor werden im Feld *command* alle Variablen zusammengeführt, die für den Prozess benötigt werden. Die Variablen unterscheiden sich je nach Anwendungsfall. Die Parameter werden jeweils als Schlüsselwörter extrahiert und mit dem Variablennamen und einem Booleanflag gespeichert. In der API-Anfrage werden für das Feld *Input* jene Werte übergeben, die in weiterer Folge von der Automation in der Cloud Factory des RPA verarbeitet werden. Die Schlüssel-Werte-Paare müssen die gleiche Bezeichnung vorweisen wie die Namen der Inputvariablen der Automation. Der HTTP-Body der API-Anfrage aus Abbildung 15 ist für jeden der Anwendungsfälle identisch. Über den Inhalt des *command* Feldes kann die Aufgabe dynamisch festgelegt werden. Nach Absenden der Anfrage wird diese von der Automation in der Cloud Factory von SAP RPA weiterverarbeitet. Der Ablauf dazu wird in Kapitel 5 behandelt.

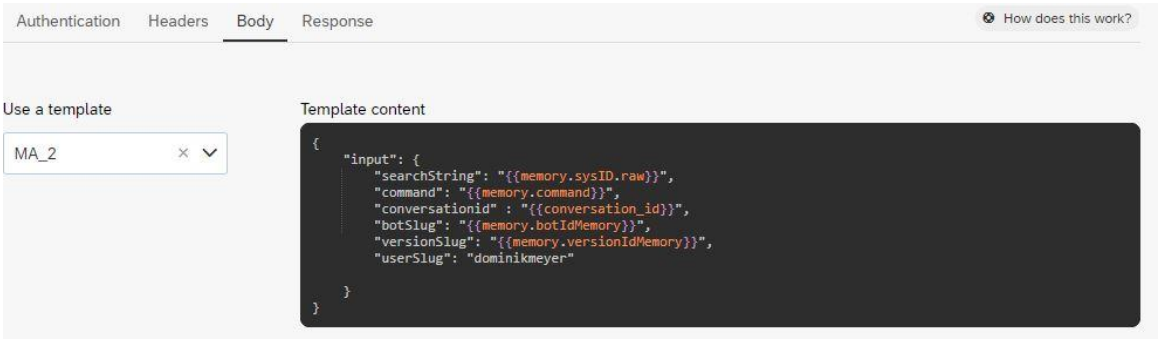


Abbildung 15: SAP CAI GUI - Body des API-Aufrufes für den Skill Anmelden

4.3.2 Besonderheiten

Für Anwendungsfälle wie das Importieren eines Transportauftrages oder das Ändern eines*einer Benutzers*in gibt es verschiedene Parameter. Diese Parameter sind als beschränkte Entitäten definiert worden, bestehend aus einer Liste an vordefinierten Wörtern. Bei Erkennen der Absicht und der Schlüsselwörter in der Aussage wird der entsprechende Skill ausgelöst. Je nach Kondition wird die Variable der erkannten Entität im Speicher auf *Wahr* gesetzt. Damit der*die Anwender*in eine Rückmeldung bekommt, welche Parameter nun gesetzt sind, werden diese in der Variable *parameterString* zusammengefasst.

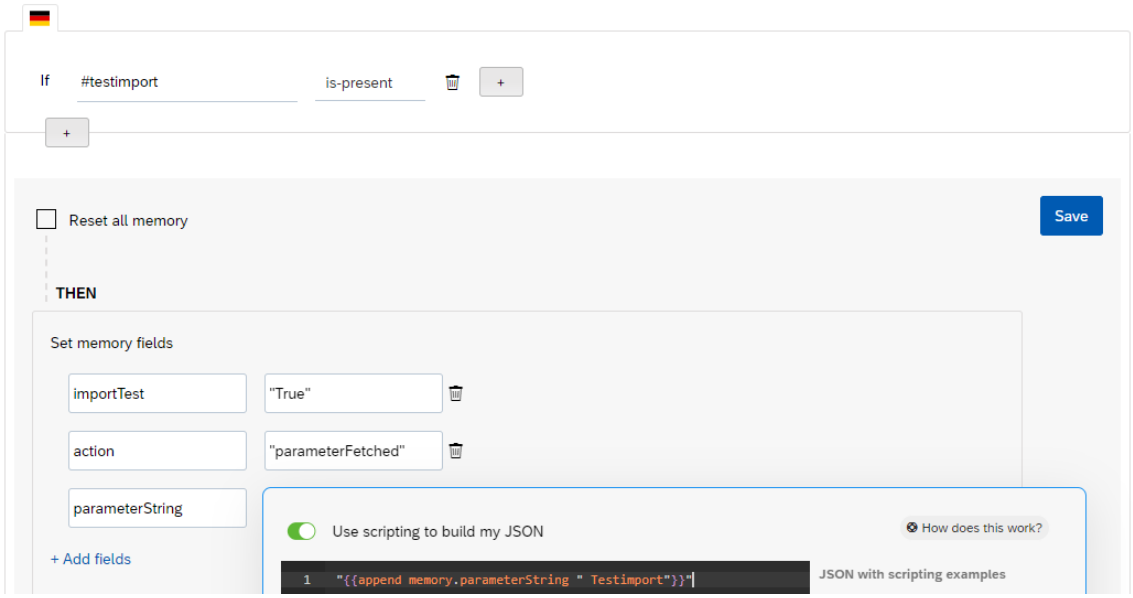


Abbildung 16: SAP CAI GUI - Parameter erfassen (eigene Darstellung)

Zum einen kann der*die Benutzer*in die Parameter beliebig hinzufügen. Das Hinzufügen macht es jedoch auch notwendig die Funktionalität des Entfernens zu unterstützen, um die Parameter bei Bedarf wieder entfernen zu können. Bei der Absicht einen Parameter zu entfernen, muss zum einen die Entität des Parameters erkannt werden und zum anderen das Schlüsselwort, welches das Entfernen kennzeichnet. Über die Handlebars Funktionen wird dann der Parameter aus der Wortfolge entfernt.

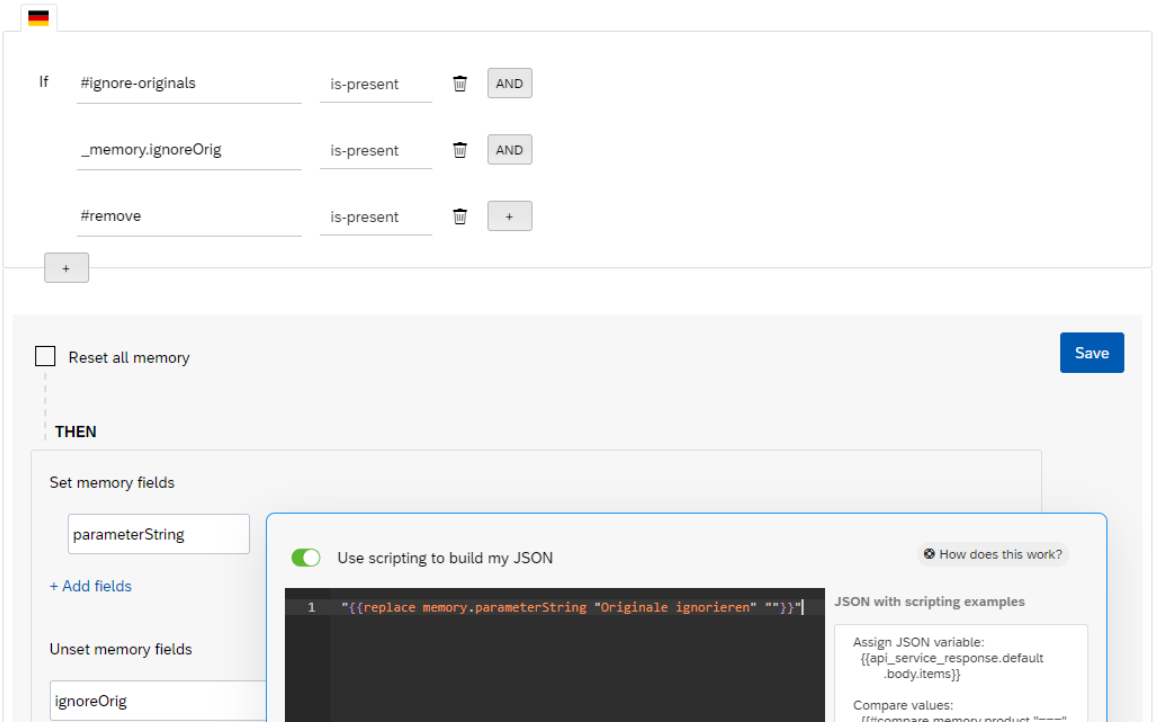


Abbildung 17: SAP CAI GUI - Parameter entfernen (eigene Darstellung)

Für ein besseres Verständnis wird die Funktionalität nochmals anhand eines Beispiels erläutert. Als konkreter Anwendungsfall wird das Hinzufügen des Parameters zum Testimport eines Auftrages herangezogen. Eine Konversation könnte dabei wie folgt aussehen:

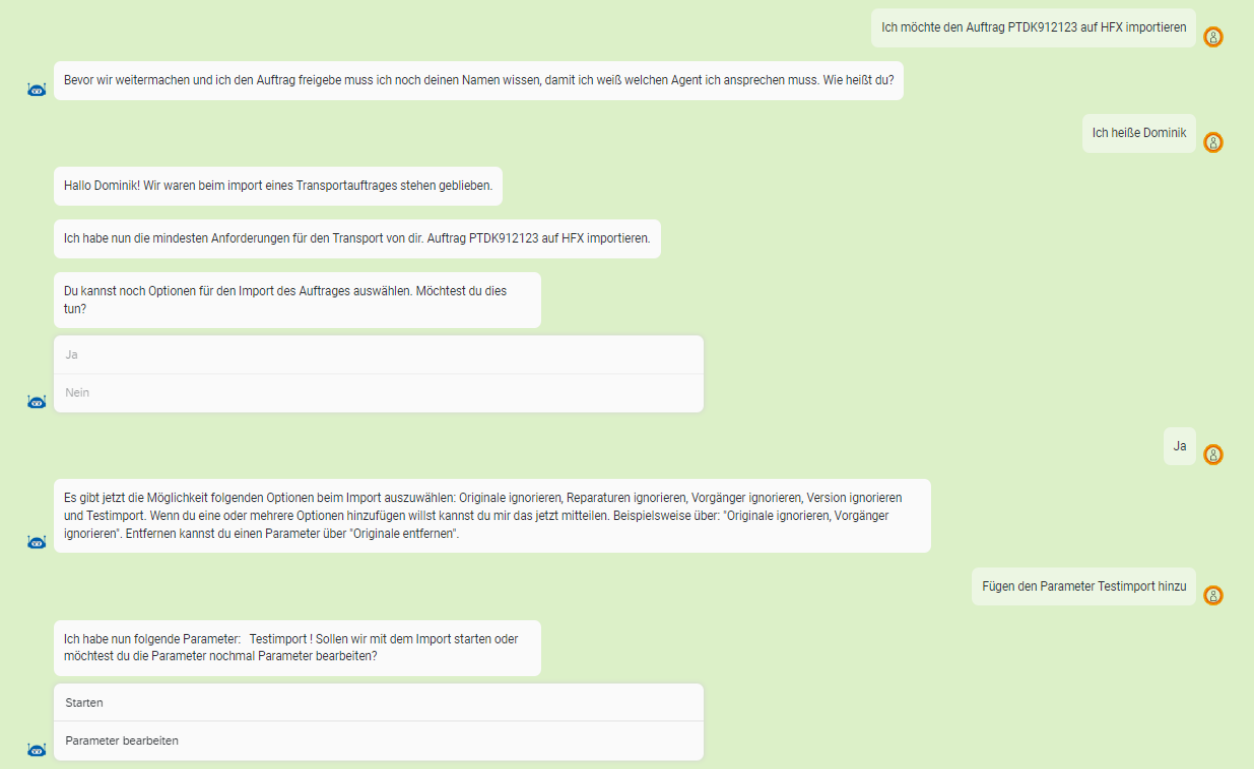


Abbildung 18: Beispielhafte Konversation zu Parametern (eigene Darstellung)

Der*die Anwender*in erklärt seine Absicht, einen Transportauftrag importieren zu wollen. Der proBot prüft danach, ob noch Parameter hinzugefügt werden sollten. Anhand des Buttons kann entweder sofort gestartet oder es können noch zusätzliche Optionen hinzugefügt werden. Bei Auswählen des Ja-Buttons erläutert der Bot die Möglichkeiten für die Parameter. Es können beliebige, durch „,“ getrennte Optionen, aneinandergereiht werden. Nach jedem Vorgang, beidem die Parameter bearbeitet wurden, stellt der Bot die Rückfrage, ob der Vorgang gestartet werden soll oder die Optionen nochmals bearbeitet werden möchten. Bei der Eingabe des Wortes *Testimport* wird die Entität *#Testimport* erkannt und somit der Skill *get-parameter-import* ausgelöst. Abhängig von den Einstellungen hinsichtlich der Strenge und der Sammlung von den Entitätsworten, kann auch die Eingabe von *Test*, im Kontext des Importes eines Transportauftrages als Entität *#Testimport* erkannt werden.

4.4 Zusammenfassung

In diesem Kapitel wurde einführend die Plattform und deren Architektur vorgestellt. Bei der Architektur wurde eingangs auf die einzelnen Komponenten eingegangen, um abschließend mit einem Beispiel ein ganzheitliches Bild zeichnen zu können. Anschließend erfolgte nochmals die Erläuterung der grundlegenden Designkonzepte eines Chabots aus Abschnitt 2. Basierend auf diesen grundlegenden Designkonzepten erfolgte die Veranschaulichung der plattformspezifischen Feinheiten. Aufbauend auf den theoretischen Grundlagen wurde die Implementierung im Rahmen der Arbeit aufgezeichnet.

5 SAP Robot Process Automation

Nachdem die Komponente zur Interaktion mit dem*der Benutzer*in (SAP CAI) vorgestellt wurde, wird in diesem Kapitel jene behandelt, die die Aufgaben ausführt. Hierzu wurde ebenfalls eine von SAP zur Verfügung gestellte Technologie verwendet – SAP Intelligent Robotic Process Automation (SAP iRPA). Analog zu vorherigem Kapitel wird zuerst auf die theoretischen Grundlagen, bestehend aus Architektur und plattformspezifischen Besonderheiten eingegangen, um dann in weiterer Folge die konkrete Umsetzung darzustellen.

5.1 Architektur

Die Grundidee des RPA liegt darin, sich wiederholende Prozesse zu automatisieren. Konkret sollen anhand von SAP RPA die Automatisierungen nicht nur auf SAP-Systeme beschränkt werden, sondern auch auf Automatisierungen über die Grenzen dieser hinweg. Diese Ende-zu-Ende Lösung ermöglicht es dadurch, Prozesse unabhängig von der Umgebung zu modellieren und zu automatisieren. Die Architektur von SAP RPA setzt sich aus den drei Hauptkomponenten des Desktop Agent, der Cloud Factory und dem Cloud Studio zusammen. Diese werden anschließend in den einzelnen Abschnitten detailliert aufgearbeitet werden, um weiterführend einen Gesamtüberblick geben zu können.

5.1.1 Desktop Agent

Der Desktop Agent ist ein JavaScript basiertes Framework, welches auf dem Gerät der Anwender*innen installiert wird. Über den Desktop Agent werden die Programme zur Ausführung gebracht. Aufgrund der Tatsache, dass die Automatisierung nicht nur auf SAP-Programme beschränkt ist, verfügt der Desktop Agent auch unterschiedliche Treiber und Konnektoren (APIs), um auf entsprechende Ressourcen zugreifen zu können. (Koch; Stass 2022, S. 61)

Damit ein Automatisierung augeführt werden kann, werden die Komponenten aus Abbildung 19 benötigt. Um die Komponenten der Abbildung zuordnen zu können, werden diese nachfolgend beschrieben und sind anhand der Nummerierung auf der Abbildung wiederzufinden:

1. Application Launcher -> zuständig für das Herunterladen der Dateien, welche zur Ausführung benötigt werden
2. Work Manager -> bringt die Skripte zur Ausführung
3. Konnektoren -> sorgen dafür, dass Informationen zwischen den Anwendungen ausgetauscht werden können. Es gibt eine Vielzahl an Konnektoren, die nachfolgend aufgelistet werden:
    a. Web Konnektor - alle Elemente des DOM können über diesen Konnektor angesprochen werden.
    b. UI Automation Konnektor – Protokoll von Microsoft über welches Anwendungen angesprochen werden können die dies unterstützen.

c. SAP GUI Konnektor – Ermöglicht es, die SAP GUI Anwendung für Windows anzusteuern.

d. SAP UI5/ S/4 Konnektor – Erweiterung zum Webkonnektor

e. Win32 – API für den Informationsaustausch mit Windows Anwendungen

f. Java – Steuerung von Javaprogrammen

g. HLLAPI – API zur Ansteuerung von Mainframe Computern (SAP 2022, S. 68–203)

4. Javascript Umfeld -> bringt die Skripte zur Ausführung werden können (SAP 2022, S. 7–8)

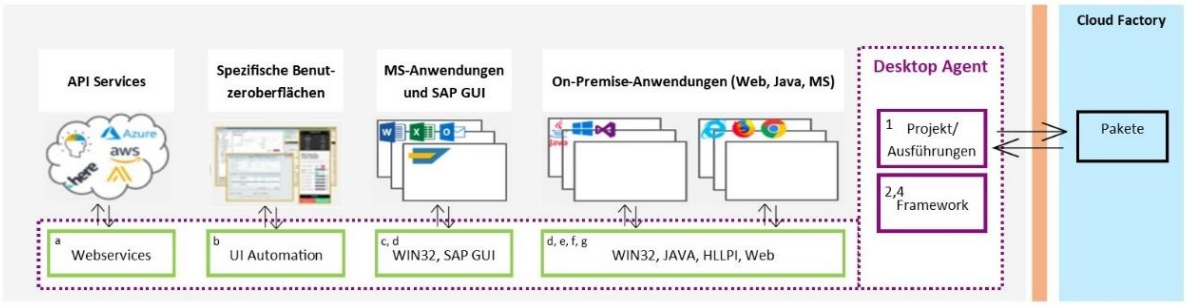


Abbildung 19: Aufbau Desktop Agent (eigene Darstellung, angelehnt an (SAP 2022, S. 6))

5.1.2 Cloud Factory

Die Cloud Factory fungiert als Orchestrierungs- und Überwachungsschicht für alle eingesetzten Agenten, die registriert werden. Die Cloud Factory erfüllt folgenden Aufgaben im RPA:

- Steuerung und Verteilung der Jobs auf die Agenten, anhand dieser die Ausführung erfolgt
- Bereitstellung von Paketen, welche die Konfigurationen und ausführbaren Elemente enthalten
- Verwaltung der Agenten
- Planung (beaufsichtigt und unbeaufsichtigt - siehe Abschnitt 5.2) und Überwachung der Ausführungen

Über die Definition von Umgebungen kann auch innerhalb der Cloud Factory eine 3-System-Landschaft realisiert werden. Je nach Umgebungen dürfen bestimmte Gruppen die Agenten ausführen.

Wie in Abbildung 19 ersichtlich, umfasst das Umfeld der Cloud Factory eine Komponente *Pakete*. Ein Paket enthält das kompilierte Projekt mit allen Szenarien (Automatisierungen), sowie Konfigurationen hinsichtlich Ausführungsmodus, -umgebung und -zeitbereiche. Zusätzlich besteht ein Paket noch aus dem Framework und den Ressourcen (SDK). Dieses Paket wird über die Cloud Factory zur Verfügung gestellt und, was durch den bidirektionalen Pfeil suggeriert

wird, vom Desktop Agent heruntergeladen. Weiters werden alle Jobs über die Cloud Factory abgewickelt und auch historisch erfasst.

5.1.3 Cloud Studio

Zur Entwicklung von Automatisierungen kann das Cloud Studio verwendet werden. Über dieses Low-Code bzw. No-Code-Tool können Automatisierungen erstellt werden. Als alternative Umgebung kann auch auf die Verwendung des Desktop Studio zurückgegriffen werden. Der Support dieser Umgebung wurde jedoch eingestellt und somit sind Neuerungen nur in der Cloud Umgebung verfügbar. Im Rahmen der Arbeit erfolgte die Umsetzung vollständig in der Cloud Umgebung.

5.1.4 RPA Gesamtarchitektur

Folgender Abschnitt soll die vorher erläuterten Komponenten nochmals zu einem Gesamtbild zusammenfügen und visualisieren. Dabei werden auch die einzelnen Entwicklungsphasen erläutert und veranschaulicht, welche Rolle die Komponenten in den einzelnen Phasen spielen. In der nachfolgenden Abbildung 20 wird die gesamte Architektur dargestellt:

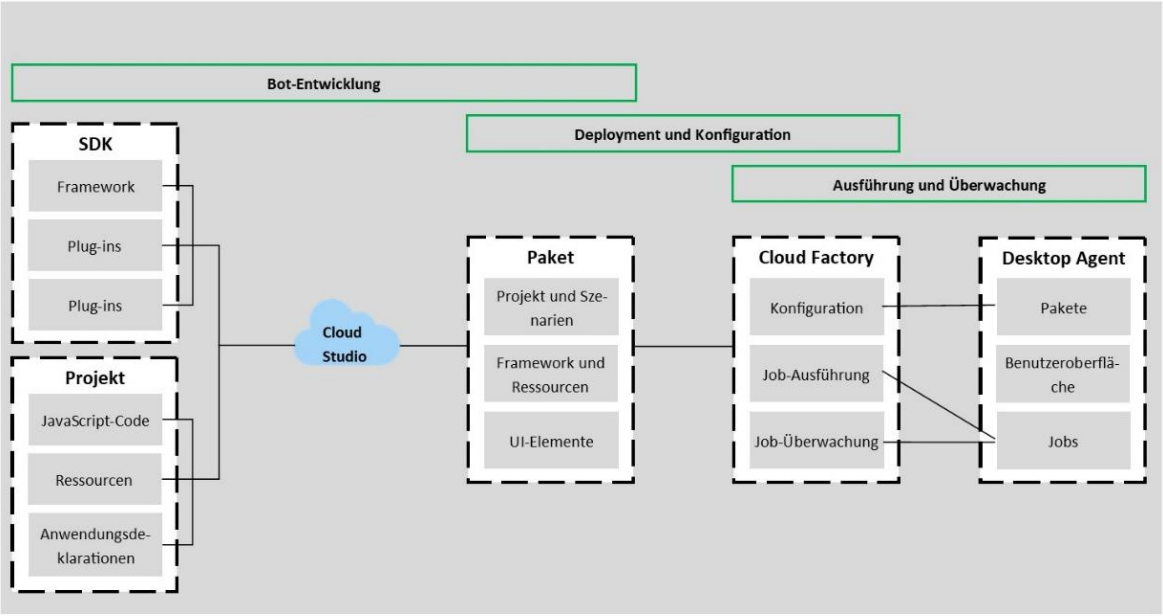


Abbildung 20: Gesamtarchitektur SAP RPA (angelehnt an (Koch; Stass 2022, S. 65))

Während der Bot-Entwicklungsphase erfolgt die Implementierung im Desktop Studio bzw. Cloud Studio. Je nach Anwendungsfall können verschiedene Plugins (Konnektoren) hinzugefügt werden. Diese Phase der Entwicklung besteht aus vier Arbeitsschritten:

1. Anwendungserfassung -> Erfassung der Anwendungen
2. Workflow Design -> Definition der auszuführenden Schritte durch den Bot, basierend auf den deklarierten Anwendungen aus Schritt 1

3. Testen und Debuggen -> Testen der RPA Automatisierungen auf dem Desktop Agent direkt aus der Entwicklungsumgebung heraus
4. Paket-Export -> Exportieren des Paketes in die Cloud Factory

Wie aus Schritt 4 hervorgeht, entsteht als Ergebnis der Entwicklung ein Paket, welches die Szenarien sowie das Framework (Plug-ins aus SDK) enthält und in der Cloud Factory als Paket zur Verfügung gestellt wird. Hierfür wird das fertige Paket importiert und bereitgestellt. Anschließend können in der Cloud Factory Konfigurationen hinsichtlich Startzeitpunkt, Modus oder Ähnlichem vorgenommen werden.

Als letzter Schritt erfolgt die Phase der Ausführung und Überwachung. Nach dem Deployment kann das Paket beaufsichtigt oder unbeaufsichtigt ausgeführt werden. Bei einer beaufsichtigten Ausführung obliegt das Starten dem*der Benutzer*in. Bei einer unbeaufsichtigten Ausführung wird das Paket, basierend auf der Konfiguration der Cloud Factory, gestartet. Die Ausführung erfolgt, wie in Abschnitt 5.1.1 beschrieben, durch den Desktop Agent. (Koch; Stass 2022, S. 61–67)

5.2 Spezifikationen

In diesem Abschnitt werden zusätzliche Spezifikationen des SAP Intelligent Robotic Process Automation aufgelistet, welche bei der Umsetzung in nachfolgenden Kapitel 5.3 Anwendung gefunden haben.

Wie bereits erwähnt wurde, wird zwischen einem beaufsichtigten und unbeaufsichtigten Szenario unterschieden. Bei zweiterer Konfiguration besteht die Möglichkeit einen API-Auslöser zu definieren. Anhand diesem kann das Projekt über die API von einem externen Kanal gestartet und dadurch zur Ausführung gebracht werden. (Koch; Stass 2022, S. 246–247).

Die API-Auslöser werden innerhalb der Cloud Factory für eine Umgebung definiert. Um den Auslöser in weiterer Folge ansprechen zu können, wird zum einen ein Token und zum anderen ein generierter API-Schlüssel benötigt. Die Generierung des API-Schlüssel erfolgt über die Cloud Factory. Die Authentifizierung arbeitet mit der OAuth2 Technologie und den Service Key Credentials aus der BTP für die Cloud Factory. Über die Business Technology Platform können von der SAP zur Verfügung gestellte Services genutzt werden. Unter dieses Service fällt das RPA.

Anhand der Agent Attribute kann die Verteilung der Jobs konfiguriert werden, um sicherstellen zu können, dass ein spezifischer Agent die Automatisierung ausführt. Dafür müssen die Attribute zum einen dem API-Auslöser und zum anderen einem oder mehreren Agenten zugewiesen werden. Die Attribute werden als Buchstabenfolge definiert. Bei einer eingehenden Anfrage wird mit der Zeichenfolge, welche dem Auslöser hinzugefügt wurde, nach entsprechenden Übereinstimmungen in den Agent Attributen, über alle Agenten hinweg, gesucht. Beide Vorgänge werden innerhalb der Cloud Factory abgewickelt.

5.3 Praktische Umsetzung

In Abschnitt 2.1 wurde erwähnt, dass alle Anwendungsfälle in der Cloud Factory implementiert werden sollten. Während der Implementierungsphase trat jedoch eine Komplikation auf, die dazu führte, dass die Grundidee adaptiert werden musste.

Bereits in Kapitel 2.3 wurde die Passwortdatenbank Problematik erwähnt. Hinzu kam, dass es zum Zeitpunkt der Implementierung eine Problematik (auf Seiten der SAP) gab, welche die Verwendung der BAPI SDK mit der Version 2.0.20 nicht ermöglichte. Da für einige Anwendungsfälle Operationen auf diversen SAP-Systemen ausgeführt werden sollten, führte dies dazu, dass eine alternative Lösung gesucht werden musste. (Meyer 2022)

Die adaptierte Lösung gestaltete sich so, dass die Automatisierungen weiterhin von SAP RPA ausgeführt werden, die Programmlogik jedoch vollständig in Form des SAP GUI Scripting (Abschnitt XY) umgesetzt wurde.

Die Implementierung in der SAP RPA fiel daher entsprechend kurz aus. Die finale Umsetzung stellt sich wie aus Abbildung 21 ersichtlich dar.

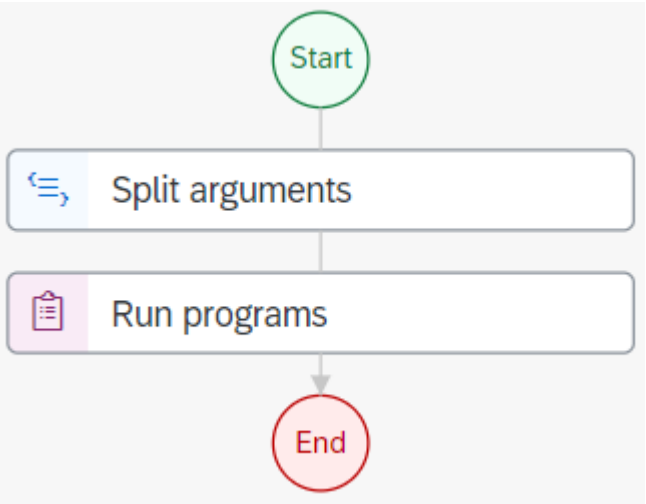


Abbildung 21: Automatisierung in der Cloud Factory (eigene Darstellung)

Im ersten Schritt werden die Argumente, welche als Eingabeparameter an die Automatisierung übergeben werden, verarbeitet. Die Eingabeparameter setzen sich wie folgt zusammen:

- *searchString* – Ein Stringliteral, dass in den Systemverbindungen, welche in einer XML Datei hinterlegt sind, gesucht wird.
- *command* – Kommando basierend auf den Usereingaben in der CAI
- *conversationid* – Eindeutige ID der Konversation aus der CAI
- *botSlug* – Eindeutige Nummer, die den Bot in der CAI identifiziert
- *versionSlug* – Eindeutige Versionsnummer des Bot in der CAI
- *userSlug* – Username, dem der Bot zugeordnet werden kann in der CAI

Innerhalb des Schrittes *Split arguments* wird zum einen das übergebene Kommando aus der CAI über das Trennzeichen „;“ in eine Array geteilt und basierend auf diesem wird ein entsprechendes Befehlszeilenkommando gebaut. Folgendes Beispiel soll dies verdeutlichen:

1. Die CAI erfasst das Kommando *action;login;language;DE;mandt;;* und übergibt dies im Feld *command* in der HTTP Post Anfrage an den API-Auslöser. Das Kommando bedeutet, dass der*die Benutzer*in sich gerne anmelden möchte. Weiteres soll als Anmeldesprache Deutsch und der Standardmandant verwendet werden.
2. Nach Empfangen der Anfrage über die API werden anhand der zugewiesenen Attribute für den Auslöser nach Übereinstimmungen mit diesem in den Agent Attributen gesucht. In weitere Folge erfolgt die Ausführung der Automatisierung über den entsprechenden Agenten.
3. In der Automatisierung wird im 1. Schritt das eingehende Kommando über das Trennzeichen „;“ in ein Array geteilt. Die Eingabeparameter sind dabei identisch zu jenen der Automatisierung. Zusätzlich wird im Outputparameter *o_output* das Kommo.(Abbildung XY)

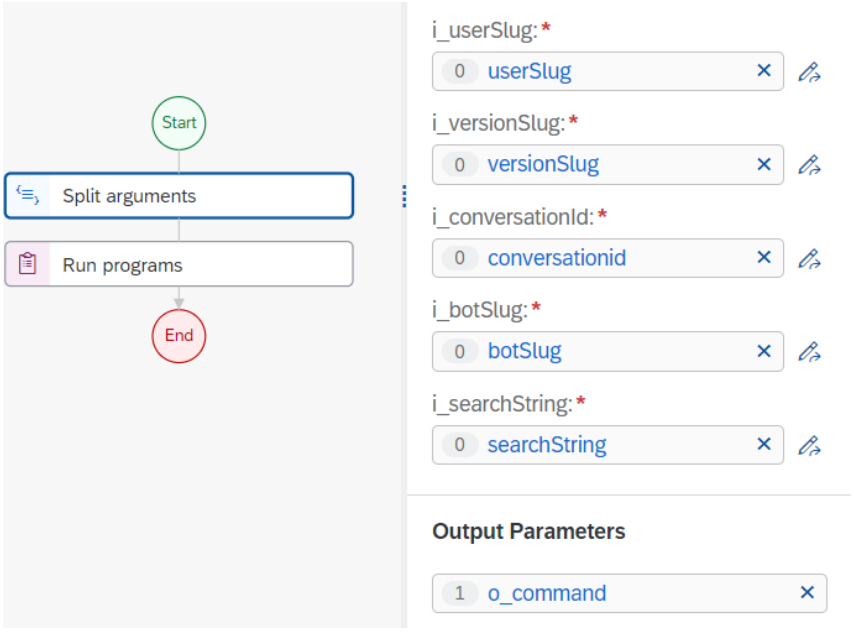


Abbildung 22: Automatisierung im Cloud Studio - Schritt 1 (eigene Darstellung)

Nach der Teilung werden die Eingabeparameter *botSlug*, *userSlug*, *versionSlug* und *searchString* sowie die Werte des Array zu einem Befehlszeilenargument zusammengeführt. Dies würde mit dem Beispiel aus Schritt 1 wie folgt aussehen:

/c Start SAPGUI.exe action login -botSlug "a2494627-231c-45f1-8a47-f345cf169728" -userSlug "dominikmeyer" -versionSlug "916b83fb-8093-46c3-a955-7777f6a03663" -conversationid "77f100ff-a16e-46de-95d0-ee79454e0da8" -searchString "PTD"

Folgende Parameter wurden bisher noch nicht aufgegriffen und werden nachfolgend beschrieben:

- */c* – Starten eines Kommandofensters
- *SAPGUI.exe* – Name der ausführbaren Datei des SAP GUI Scripting, welche die Programmlogik implementiert
- *action* – Aktion, die ausgeführt wird. Die weiteren Parameter, welche für die Aktionen verwendet werden können, sind aus Abschnitt XY zu entnehmen.

Das zusammengesetzte Kommando wird am Ende von Schritt 1 exportiert und in Schritt 2 (Abbildung 23) zur Ausführung gebracht:

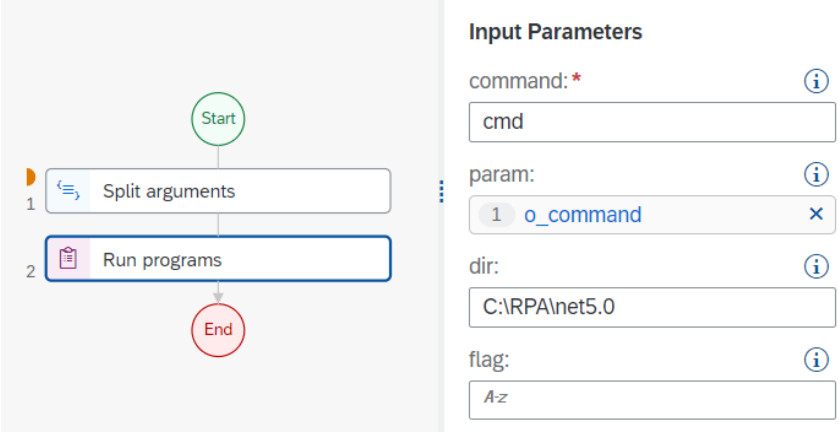


Abbildung 23: Automatisierung im Cloud Studio - Schritt 2 (eigene Darstellung)

Somit erfüllte die SAP RPA die Funktionalität zum Empfangen von API-Anfragen, welche die Automatisierung auslösen. Nach Starten der Automatisierung werden die Body Parameter der HTTP Anfrage an die gleichnamigen Parameter der Automatisierung übergeben. Die Automatisierung führt die jeweiligen Schritte der Aufgabe durch (Abbildung 21). Nach Abschluss der Ausführung wird ein Befehlszeilenkommando zurückgegeben, welches über ein Kommandofenster im Desktop Agent zur Ausführung gebracht wird.

Eine weitere Problematik war, dass für jeden API Trigger mehrere Agent Attribute definiert werden konnten. Beim Auslösen der Automatisierung über die API würde dann ein Agent ausgewählt werden, welcher gerade verfügbar ist und auf die Attribute (im Auslöser) zutrifft. Somit war die Möglichkeit nicht gegeben, den Agent dynamisch über die API-Anfragen ansteuern zu können. Dieses Verhalten ist jedoch speziell für Aufgaben wie das An- und Abmelden von großer Bedeutung, da dies für den*die Nutzer*in auf dem entsprechenden Notebook ausgelöst werden sollte. Um dies zu umgehen, wurde ein Dispatcher (Proxy) eingerichtet. Dieser Proxy empfängt die Anfragen von der SAP CAI und liest das Feld *name* im Kopf aus. Basierend auf dem Wert dieses Feldes erfolgt die Weiterleitung an die API URL in der Cloud Factory, welche das übereinstimmende Agent Attribut aufweist. Somit wurde jedem Trigger nur jeweils ein Agent Attribut zugeordnet, damit bei jeder Ausführung nur ein spezifischer Agent ausgewählt wird.

5.4 Zusammenfassung

In diesem Kapitel wurden einführend die Bestandteile der SAP RPA Technologie vorgestellt. Dazu gehört der Desktop Agent, welcher die Szenarien zur Ausführung bringt. Hierbei wird zwischen beaufsichtigter Ausführung, welche vom*von Benutzer*in initiiert werden muss und unbeaufsichtigter Ausführung, welche eingeplant wird, unterschieden. Innerhalb der Cloud Factory werden diese Szenarien zur Verfügung gestellt und konfiguriert. Der Grundstein für diese Automatisierungen wird im Cloud Studio gelegt, in dem die Programmierung erfolgt.

Die Grundidee, dass die Umsetzung vollständig im Cloud Studio implementiert werden sollte, musste aufgrund von Komplikationen verworfen werden. In der finalen Version ist die Automatisierung nur noch dafür zuständig, dass alle Eingabeparameter zu einem Befehlszeilenargument zusammengefügt und in weiterer Folge über die Kommandozeile ausgeführt werden. Somit wurde die Programmlogik weitestgehend ausgelagert und wird im nächsten Kapitel erläutert.

6 SAP GUI Scripting und ABAP

Da die Umsetzung im Rahmen der Arbeit nicht vollständig anhand des SAP RPA implementiert werden konnte, kamen die Technologien des SAP GUI Scriptung und ABAP zum Einsatz. Nachfolgend werden die jeweiligen Konzepte erläutert, um abschließend auf die praktische Umsetzung eingehen zu können.

6.1 SAP GUI Scripting Architektur

Mit dem Release SAP R/3 wurden die „active elements“ eingeführt. Diese sollten es ermöglichen, GUI Elemente der SAP-Oberfläche codebasiert ansprechen zu können. Die Technologie konnte aber von vielen neueren Anwendungen nicht verwendet werden und wurde somit zum Vorgänger von SAP GUI Scripting.

Im SAP GUI Scripting wird die gesamte SAP GUI Oberfläche als Hierarchie von Objektmodellen implementiert, ähnlich zum HTML-DOM. Die Objekte implementiert ein Interface. In diesem Interface sind alle Aktionen definiert, welche ausgeführt werden können. Die Hierarchie zur Laufzeit setzt sich folgendermaßen zusammen:

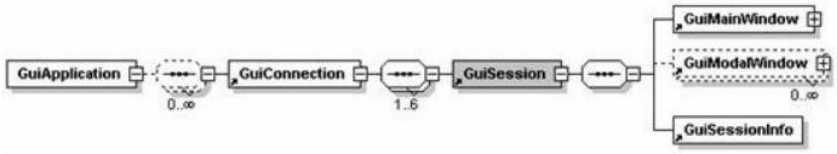


Abbildung 24: SAP GUI Scripting Hierarchie (SAP 2019, S. 6)

Das Objekt GuiApplication (siehe Abbildung 24) fungiert als Wurzelobjekt. Dieses Wurzelobjekt kann keine bis unbeschränkt viele Objekte der GuiConnection beinhalten. Die GuiConnection wiederum beinhaltet die Verbindung zu einem SAP-System. Für eine GuiConnection können maximal sechs Sessions erstellt werden. Die Sessions werden in dem Objekt GuiSession abgebildet. Innerhalb jeder Session gibt es ein GuiMainWindow und eine GuiSessionInfo. Zusätzlich können keine, bis unendlich viele Modal Fenster existieren (GuiModalWindow). (SAP 2019, S. 5–6)

Auf die gesamten Objekte wird über den Running Object Table (ROT) zugegriffen. ROT ist eine Tabelle, die alle registrierten COM-Objekte beinhaltet. Bei COM-Objekten handelt es sich um Laufzeitobjekte von Microsoft (DDL), welche sprachenunabhängig über die Schnittstelle angesprochen werden können. (stevewhims 2021)

Da es in einigen Programmiersprachen jedoch nicht möglich ist, auf die ROT zugreifen zu können, implementiert das SAP GUI Scripting eine Wrapperklasse. Diese Klasse stellt die Möglichkeit zur Verfügung, auf eine laufende SAP GUI Instanz zugreifen zu können.

Um in weiterer Folge im Abschnitt 6.3 die Umsetzung nachvollziehen zu können, werden in Abbildung 25 die Komponenten des GuiMainWindow Objektes dargestellt. Über diese Klassen kann das jeweilige Objekt angesprochen und eine Aktion ausgeführt werden:

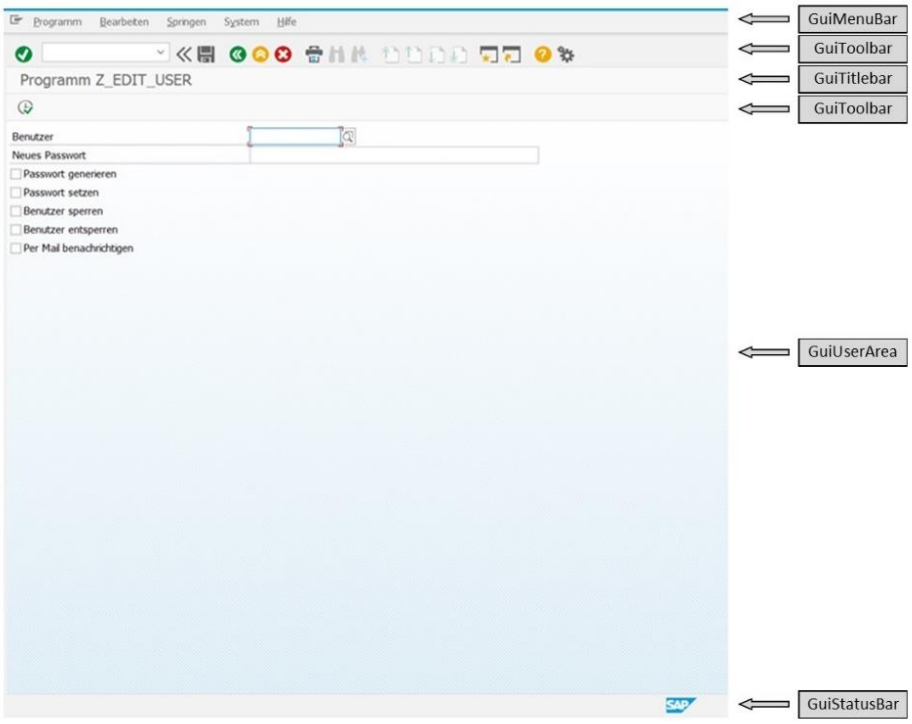


Abbildung 25: Elemente GUI Scripting SAP GUI (SAP 2019, S. 7)

6.2 ABAP

Die Ausführung der jeweiligen Operationen im System wurde anhand von ABAP Programmierungen realisiert. In diesem Abschnitt wird kurz die Programmiersprache erläutert, bevor dann abschließend die gesamte Umsetzung, im Zusammenspiel mit dem SAP GUI Scripting, abgebildet wird.

In den Anfängen wurde die Sprache als Makro Assembler verwendet. Damals bedeutete die Abkürzung Allgemeiner-Bericht-Aufbereitungs-Prozessor. Mit dem Release von R/3 wurde ABAP erstmals eine Interpretersprache. Lediglich der Systemkern, welcher die Schnittstelle zwischen Betriebssystem und SAP-System darstellt, wurde noch in der Sprache C geschrieben. Alle anderen Anwendungen wurden in ABAP implementiert, was auch eine neue Namensgebung mit sich brachte. ABAP bedeutete nun fortan „Advanced Business Application Programming“. Danach folgten noch weitere Meilensteine wie die Einführung der Objektorientierung (ABAP Objects) oder auch die Einführung von Unicode, welche die Semantik der Sprache nochmals änderte. Die letzte große Neuerung war die Integration von ABAP in den SAP Netweaver, welcher dadurch eine Namensänderung zu SAP NetWeaver Application Servers ABAP, kurz AS ABAP, erhielt. (Schwaiger 2019, S. 42–46)

Für den Abschnitt der praktischen Umsetzung ist es hilfreich, dass die Architektur eines AS ABAP Systems aufgezeichnet wird. Diese wird in Abbildung 26 skizziert:

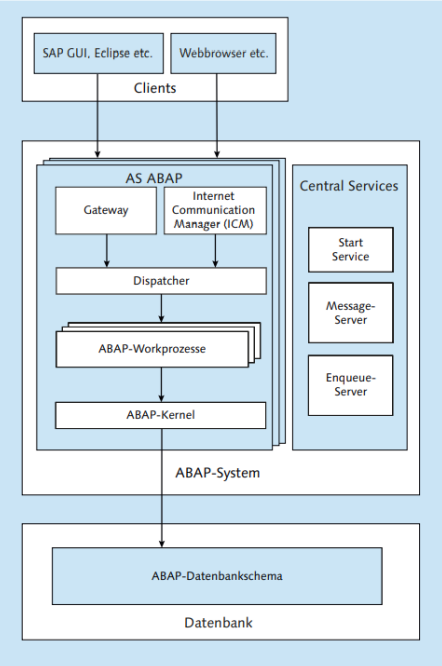


Abbildung 26: Aufbau SAP Netweaver AS ABAP (Gahm 2016, S. 118)

Der Zugriff auf das System erfolgt über entsprechende Clients wie die SAP GUI, Eclipse oder den Webbrowser. Die Anfragen der Clients werden vom Gateway bzw. ICM empfangen und an den Dispatcher weitergeleitet. Dieser erstellt daraus folgend ABAP-Workprozesse. Die ABAP-Workprozesse arbeiten die Anfragen in Form von ABAP-Programmen ab und werden im Kernel zur Ausführung gebracht. Abhängig von der Anweisungsart werden unterschiedliche Komponenten im Kernel verwendet. Die Verwaltung aller Datenbankinhalte sowie des Quellecodes und der Entwicklungsobjekte erfolgt in der Datenbank. (Gahm 2016, S. 117–119)

Das System arbeitet transaktionsgesteuert. Über Transaktionen können Daten angelegt, geändert, gelöscht oder angezeigt werden. Hinter Transaktionen verbergen sich ABAP Programme, die über den Transaktionscode zur Ausführung gebracht werden. (SAP o. J.)

6.3 Praktische Umsetzung SAP GUI Scripting und ABAP

In Kapitel 5.3 wurde beschrieben, wie SAP RPA über die Befehlszeile eine lokale Datei ausführt. In diesem Abschnitt wird jenes Programm, sowie das Konzept für den Zugriff auf die SAP-Systeme und die Ausführung der Programme, geschildert.

Die Umsetzung erfolgte in Form eines vb.net Projektes (SAP GUI Scripting), welches als Resultat eine ausführbare Datei bereitstellte. Im Anhang wird der gesamte Ablauf gezeigt, welcher beim Starten des Programmes durchlaufen wird. Nachfolgend wird das Flussdiagramm aufgrund der Übersichtlichkeit in einzelne Teile unterteilt, um auch detailliert auf die Anwendungsfälle eingehen zu können.

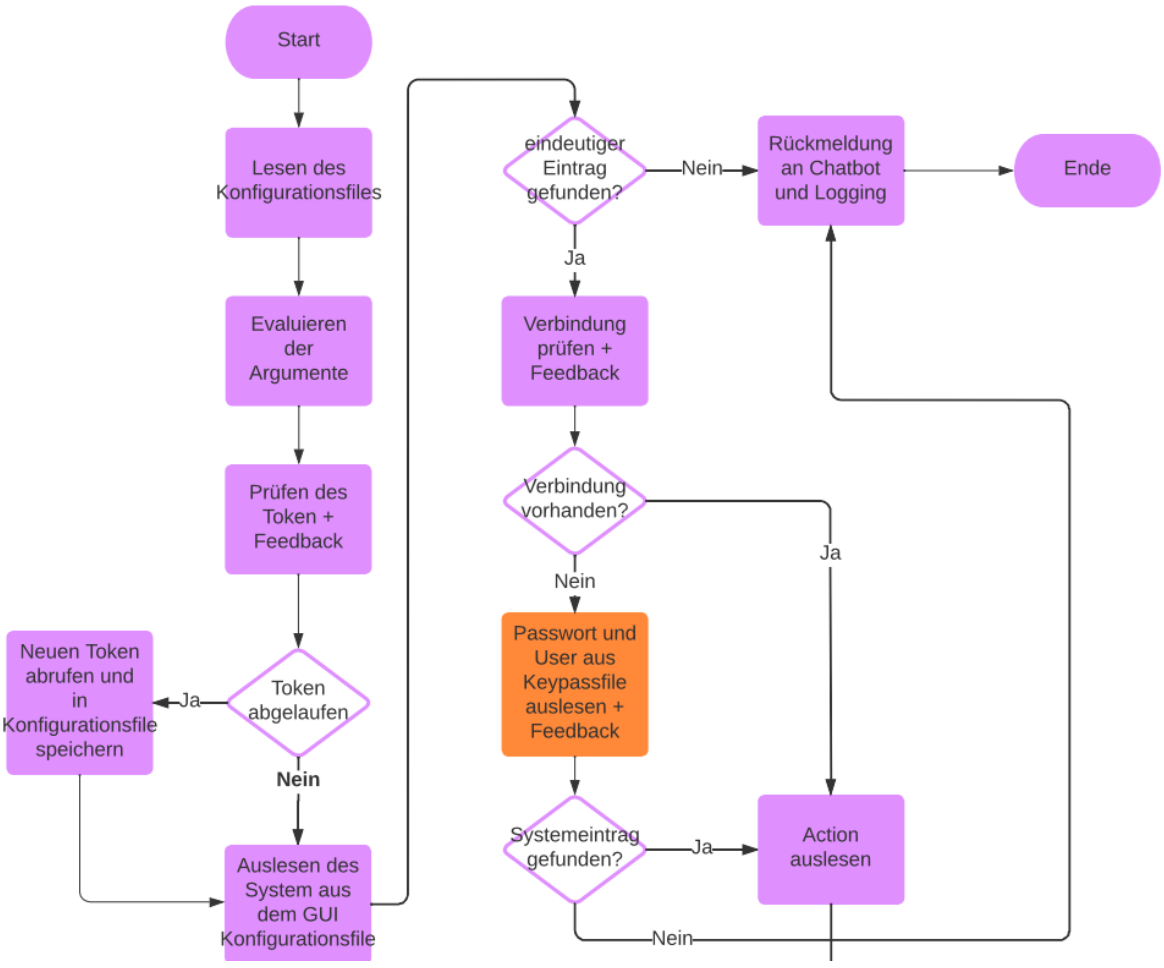


Abbildung 27: Flussdiagramm Programmablauf Einstieg (eigene Darstellung)

- Als Startpunkt wird das Programm über die Kommandozeile gestartet. Abhängig vom Anwendungsfall unterscheiden sich auch die Argumente.

- Im nächsten Schritt wird das Konfigurationsfile gelesen. In diesem werden folgende Einstellungen festgelegt:
    - ClientID und ClientSecret - für das Abfragen eines neuen Tokens für die SAP CAI
    - Authenfikations-URL - über die der Token angefordert werden kann
    - Bottoken - um Nachrichten an die Bot-API senden zu können
    - ConnectionFile – Pfad zu der XML Datei in der alle Systemeinträge der SAPGUI abgelegt sind
    - Passwortdatenbanken – jeweils der Pfad und das Passwort zur lokalen und globalen Datenbank
    - Priorität – beschreibt, welche Passwortdatenbank eine höhere Priorität hat, wenn mehrere User*innen gefunden werden
    - Token – dieser Wert darf nicht verändert werden. Innerhalb dieses Tags wird der Token der SAP CAI inklusive Ablaufdatum abgelegt
- Beim Schritt zur Evaluierung der Argumente werden diese beim Trennzeichen „blank“ getrennt und in eine Hashmap (analog zu einer Hashmap in Java), mit den jeweiligen Schlüssel-Wert Paaren, übertragen. Folgendes Kommando (aus Abschnitt XY) würde in der Hashmap wie folgt aussehen:

  action login -botSlug "a2494627-231c-45f1-8a47-f345cf169728" -userSlug "dominikmeyer" -versionSlug "916b83fb-8093-46c3-a955-7777f6a03663" -conversationid "77f100ff-a16e-46de-95d0-ee79454e0da8" -searchString "PTD"

Schlüssel	Wert
action	login
botSlug	a2494627-231c-45f1-8a47-f345cf169728
userSlug	dominikmeyer
versionSlug	916b83fb-8093-46c3-a955-7777f6a03663
conversationid	77f100ff-a16e-46de-95d0-ee79454e0da8
searchString	PTD

- Bei der Prüfung des Token wird des Ablaufdatum geprüft. Ist dies kleiner als das aktuelle Datum, wird ein neues Token abgefragt und in das Konfigurationsfile geschrieben. Ansonsten erfolgt die Verwendung des bestehenden Tokens.
- Über das Schlüssel-Wert-Paar *searchSystem* aus dem Argument in der Befehlszeile wird nach einem passenden Systemeintrag im ConnectionFile gesucht. Es wird dem*der Benutzer*in mitgeteilt, dass nach einem System gesucht wird.
- Wird ein eindeutiger Systemeintrag gefunden, erfolgt die Prüfung anhand der System-ID. Werden mehrere Systeme mit zutreffenden Kriterien gefunden, erfolgt eine Rückmeldung an den Chatbot, dass die Eingabe genauer spezifiziert werden muss.

- Nachfolgend wird nach einer Verbindung, basierend auf der System-ID, gesucht.
- Ist keine Verbindung vorhanden, werden anhand eines Javaprogrammes (orange) die Werte aus der Passwortdatenbank ausgelesen. Der*die Nutzer*in wird informiert, dass nach einer User-Passwort-Kombination in der Datenbank gesucht wird. Die JAR-Datei wird über die Kommandozeile gestartet und erwartet als Parameter den Systemnamen. Das Programm liest die benötigten Werte aus dem Konfigurationsfile aus und gibt die User-Passwort Kombination für den entsprechenden Systemnamen zurück. Die Kommunikation zwischen den beiden Programmen erfolgt über die Shell.
- Wenn in der lokalen und globalen Datenbank keine übereinstimmenden Werte gefunden werden, wird eine Nachricht an den Chatbot gesendet, dass die Kombination nicht gefunden werden kann. Ein Eintrag wird dann als übereinstimmend betrachtet, wenn der Systemname aus den Parametern mit jenem in der Passwortdatenbank übereinstimmt.
- Anschließend wird die Aktion ausgelesen, welche ausgeführt werden sollte. Dies führt dann zu anwendungsfallspezifischen Schritten, welche nachfolgend in eigenen Abschnitten erläutert werden.

6.3.1 Anmelden an Kundensystem

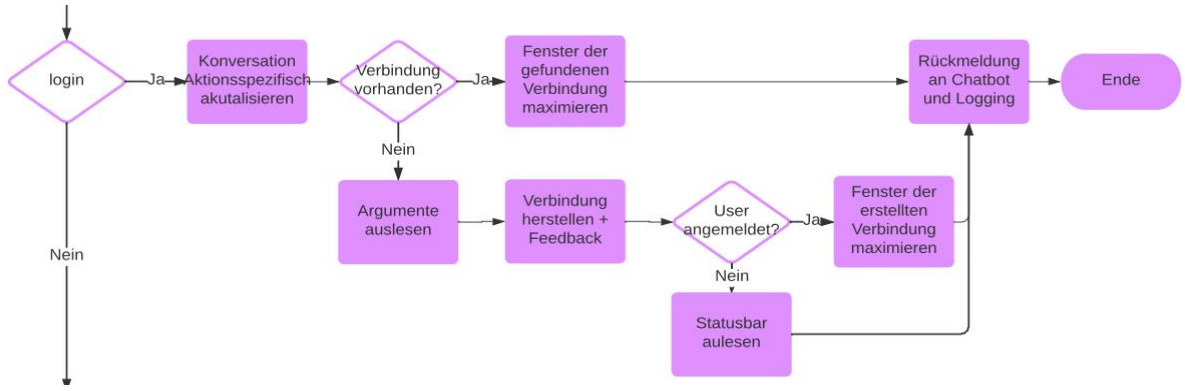


Abbildung 28: Flussdiagramm Programmablauf Anmelden (eigene Darstellung)

- Bei Eingabe der gewünschten Aktion als *login* erfolgt im ersten Schritt das Updaten der Konversation bzw. des Chatbot. In diesem Anwendungsfall würde der*die Benutzer*in darüber informiert werden, dass eine User-Passwort Kombination gefunden wurde und die Anmeldung gestartet wird. Die Nachrichten werden über die Bot API (siehe Abschnitt 4.1) übermittelt. Zusätzlich wird die Memory der Konversation dahingehend aktualisiert, dass das Feld *action* auf den Wert *executeTask* gesetzt wird, welcher kennzeichnet, dass eine Aufgabe ausgeführt wird und der Bot aufgrund dessen zu dem Zeitpunkt bis zur Beendigung dieser Aufgabe keine weiteren Aufgaben ausführen kann.
- Ist bereits eine Verbindung vorhanden, wird das Fenster maximiert und der*die Benutzer*in erhält eine Rückmeldung, dass eine existierende Session gefunden wurde.

- Ist keine Verbindung vorhanden, werden die Argumente ausgelesen. Der*die Nutzer*in wird darüber informiert, dass die Anmeldung gestartet wird. Beim Anmelden stehen die Anmeldesprache und der*die Benutzer*in als Argumente zur Verfügung. Sind diese nicht gefüllt, werden die Standardwerte verwendet.
- Nach Eröffnen der Verbindung wird über die Session Info geprüft, ob die Anmeldung erfolgreich war. Ist dies nicht der Fall, wird die Statusbar ausgelesen und die Nachricht an den*die Benutzer*in zurückgegeben. Konnte die Anmeldung jedoch erfolgreich ausgeführt werden, wird dem*der Benutzer*in die Nachricht gesendet, dass die Anmeldung erfolgreich war. Weiters wird das Feld *action* wieder geleert, damit der Bot neue Aufgaben entgegennehmen kann.

6.3.2 Abmelden an Kundensystem

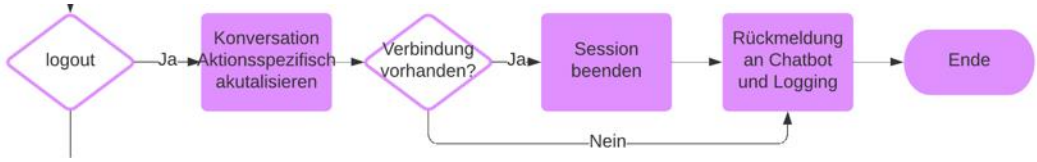


Abbildung 29: Flussdiagramm Programmablauf Abmelden (eigene Darstellung)

- Analog zu vorherigem Abschnitt erfolgt das Update an den*die Benutzer*in. Hierbei wir der*die Anwender*in darüber informiert, dass nach einer Session, basierend auf dem eingegebenen *searchString* gesucht wird.
- Bei Finden einer Session, wird diese über die Transaktion */nex* beendet und eine Rückmeldung an den Chatbot gesendet, dass die Session beendet wurde. Sollte keine Session gefunden werden, erfolgt ebenfalls eine Rückmeldung mit entsprechender Information. Das Memory-Handling erfolgt analog zu dem Anwendungsfall des Anmeldens. Beim Starten der Aufgabe wird das Feld *action* auf *executeTask* gesetzt und bei Beendigung wieder geleert.

6.3.3 Transportaufträge und Benutzer*in ändern

Die Aktionen des An- und Abmeldens erfordern keine bestehende Verbindung zu dem gesuchten System im Parameter *searchString*. Somit werden die Aktionen aus Abbildung 30 erst aufgerufen, wenn es sich bei der Aktion weder um das An- noch Abmelden handelt. Die Anwendungsfälle, welche auf nachfolgender Seite dargestellt werden, verwenden für ihre Operationen eigens programmierte Transaktionen (im SAP) und werden daher als Gesamtes betrachtet:

- Transportauftrag freigeben (releaseTA)
- Transportauftrag importieren (importTA)
- Benutzer*in ändern (changeUser)
- Transportliste anzeigen (listTransports)

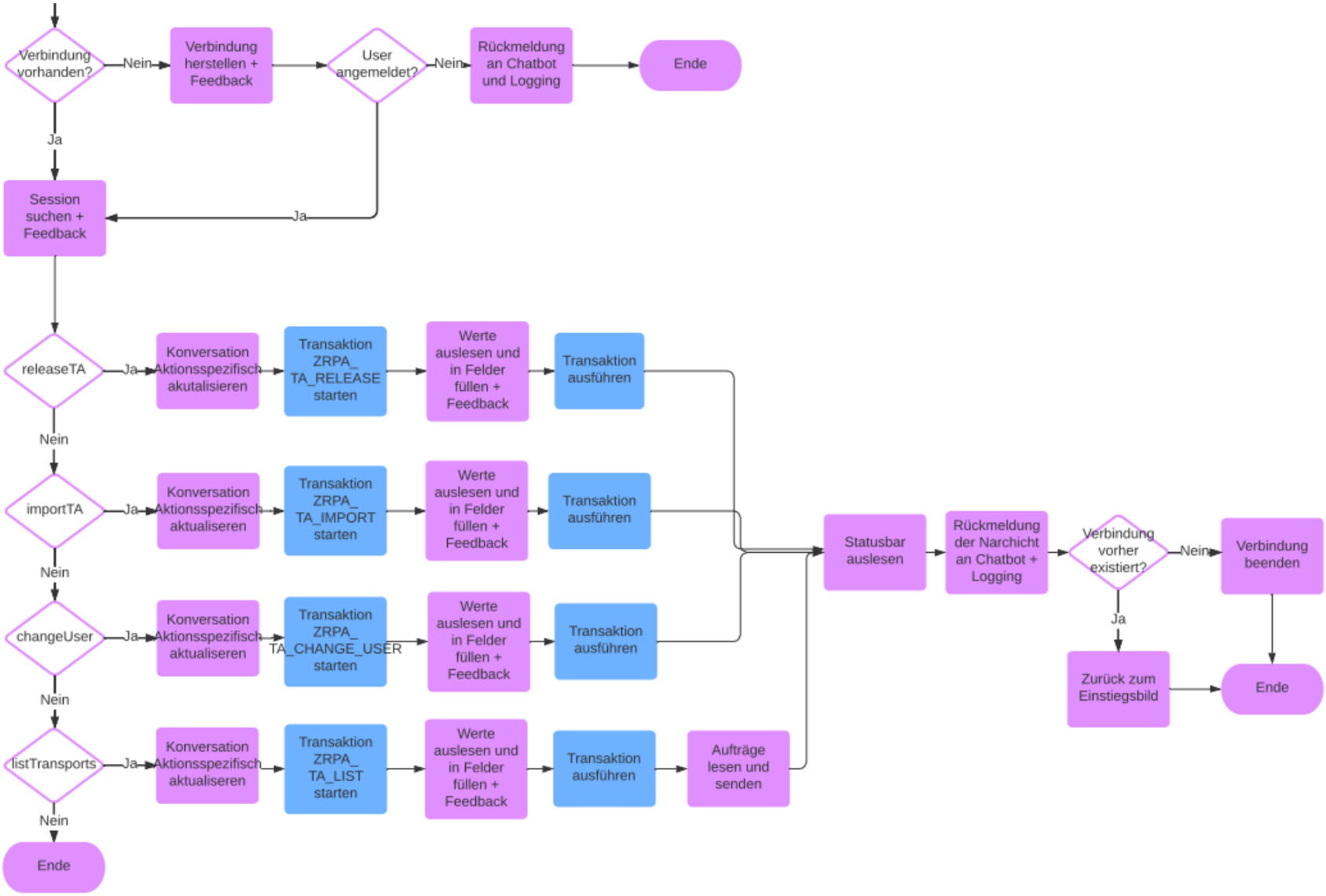


Abbildung 30: Flussdiagramm Programmablauf Transportoperationen und User ändern (eigene Darstellung)

- Im ersten Schritt wird geprüft, ob eine Verbindung, basierend auf dem Parameter *searchString,* besteht.
- Ist keine Verbindung vorhanden, erfolgt die Anmeldung basierend auf den ausgelesenen Daten von Abbildung 30. Zudem erfolgt die Rückmeldung an den*die Benutzer*in, dass keine Verbindung vorhanden ist und diese hergestellt wird.
- Wenn beim Versuch des Anmeldens ein Fehler aufgetreten ist, wird der*die Anwender*in benachrichtigt, dass keine Verbindung hergestellt werden konnte.
- War die Anmeldung erfolgreich, wird die Session gespeichert, um auf dieser alle weiteren Aktionen ausführen zu können. Der*die Benutzer*in erhält die Information, dass nun mit dem Ausführen der Aufgabe gestartet wird.
- Wie bereits bei vorherigen Anwendungsfällen wird die Konversation dahingehend aktualisiert, dass der*die Benutzer*in eine Rückmeldung erhält. Diese fällt je nach Aufgabe unterschiedlich aus. Zusätzlich wird in der Memory der Konversation die *action* auf *executeTask* gesetzt.
- Für jeden Anwendungsfall wird anschließend eine Transaktion innerhalb der Session gestartet. Die blaue Hinterlegung des Schrittes deutet bereits darauf hin, dass die Operation im SAP-System ausgeführt wird. Die eigens entwickelten Transaktionen rufen im ersten Schritt den Selektionsbildschirm des ABAP Programmes auf. Jedes Programm verwendet standardisierte SAP-Funktionsbausteine, um eine Systemkompatibilität garantieren zu können.
- Auf dem Selektionsbildschirm werden alle Parameter aktionsabhängig befüllt. Diese gestalten sich wie folgt:
    - ZRPA_TA_RELEASE – die Transaktion gibt einen Transportauftrag im System frei:

Parameter	Typ	Name in Hashmap
Auftragsnummer	Textfeld	request

Tabelle 1: Parameter Programm Transportfreigabe (eigene Darstellung)

    - ZRPA_TA_IMPORT – das Verhalten der Programms soll sich mit jenem Standardverhalten der Transaktion STMS orientieren. Daher können folgende Parameter verwendet werden:

Parameter	Typ	Name in Hashmap
Auftragsnummer	Textfeld	request
Originale ignorieren	Checkbox	ignoreOrig
Reperaturen ignorieren	Checkbox	ignoreOrig
Vorgänger ignorieren	Checkbox	ignorePre
Version ignorieren	Checkbox	ignoreVers
Testimport	Checkbox	importTest

Tabelle 2: Parameter Programm Transportimport

    - ZRPA_TA_CHANGE_USER – führt Änderungen am*an Benutzer*in durch. Anders als bei den anderen Transaktionen schließen sich hier Parameter aus.

Wenn beispielsweise in der Kommandozeile ein neues Passwort mitgegeben wird, wird automatisch die Checkbox zum Zurücksetzen des Passwortes abgewählt und jene zum Setzen eines Wunschkennwortes ausgewählt. Gleiches gilt für das Entsperren eines*einer Benutzers*in. Wird das Entsperren gewünscht, wird automatisch des Sperren abgewählt.

Parameter	Typ	Name in Hashmap
Username	Textfeld	userName
Password	Textfeld	userPassword
Passwort zurücksetzen	Checkbox	resetPWD
Passwort setzen	Checkbox	setPWD
User entsperren	Checkbox	unlockUsr
User sperren	Checkbox	lockUsr
E-Mail senden	Checkbox	sendMail

Tabelle 3: Parameter Programm Benutzer*in ändern (eigene Darstellung)

- ZRPA_TA_LIST – gibt eine Liste an Transporten aus. Bei der Implementierung war zu beachten, dass Transportaufträge hierarchisch organisiert werden. Somit musste bei der Rückgabe von Kinderknoten der Elternknoten ebenfalls zurückgeben werden. Für die Transaktion werden die Parameter zur Verfügung gestellt, welche auch in der Standardtransaktion (SE10) ausgewählt werden können:

Parameter	Typ	Name in Hashmap
Username	Textfeld	userName
Freigeg. Transporte	Checkbox	released
Änderbare Transporte	Checkbox	changeable

Tabelle 4: Parameter Programm Transportliste (eigene Darstellung)

Damit der Aufruf in der Befehlszeile dynamisch gehalten werden kann, wird bei jeder Aktion nur auf die Parameter zurückgegriffen, welche erwartet werden. Bei den optionalen Parametern wird zuerst auf deren Existenz geprüft. Ist diese gegeben, wird der entsprechende Wert in das Feld befüllt. Der Name in der Hashmap (Spalte: Name in Hashmap) deckt sich mit jenem in der Befehlszeile. Die Tabellen sollen symbolisieren, welche Möglichkeiten für die jeweilige *action* zur Verfügung stehen.

- Nach Befüllen der Parameter wird das Programm ausgeführt.
- Abschließend wird die Statusbar gelesen, um dem*der Benutzer*in das Ergebnis der Aufgabe mitteilen zu können. Die Statusbar wird vom Programm über das Schlüsselwort MESSAGE befüllt. Weiteres wird das Ergebnis der Operation in eine lokale Logdatei geschrieben.
- Einzig bei dem Anwendungsfall zur Anzeige der Transportliste muss zuerst über die Ausgabe iteriert werden, um aus den Einträgen eine Liste an Transportaufträgen zu erstellen, welche an den Bot übermittelt wird.

- Nach Rückgabe an den Chatbot wird ebenfalls ein lokaler Log (XML Datei) geschrieben, in dem die Ergebnisse der Operationen festgehalten werden. Jeder Eintrag setzt sich wie folgt zusammen:
    - Type – Erfolg der Operation (S=Sucess; E=Error; W=Warning)
    - Message – Entspricht dem Text aus der Statusbar
    - Time – Zeit der Ausführung
- Als letzter Schritt wird nochmals geprüft, ob die Verbindung zu Beginn bereits existierte. Existierte bereits eine Verbindung, wird die Session auf den Einstiegbildschirm gesetzt (Transkation: /n). Ansonsten wird die Verbindung beendet.

Aufgrund der Tatsache, dass das Programm von der CAI nur mit Aktionen aufgerufen wird, welche auch behandelt werden, bedarf es keiner zusätzlichen Rückmeldung.

6.4 Zusammenfassung

In diesem Kapitel wurde der letzte Bestandteil der Umsetzung behandelt. Zuvor wurde bereits auf die Komponenten zur Erfassung der Anforderungen und der Kommunikation mit dem*der Nutzer*in (SAP CAI), sowie auf die Komponenten zur Ausführung der Automatisierungen (SAP RPA) eingegangen. Die noch offene Komponente des SAP GUI Scripting und ABAP implementiert die gesamte Geschäftslogik. Einführend konnte die Architektur des SAP GUI Scripting mit den Klassenobjekten und deren Hierarchie erläutert werden. Anschließend erfolgte ein geschichtlicher Exkurs, welcher die Meilensteine der Sprache ABAP von einer Marko Assembler Sprache hin zu einer objektorientierten Sprache beleuchtete. Basierend darauf wurde noch kurz auf den Aufbau eines ABAP Systems eingegangen.

Analog zu vorhergehenden Kapiteln erfolgte abschließend die Erläuterung der technischen Implementierung im Rahmen der Arbeit. Die Visualisierung stellte sich in Form eines Flussdiagramms dar, bei dem der gesamte Programmdurchlauf, vom Auslesen der Konfigurationsdatei sowie der User-Passwort Kombination bis hin zur Operation im SAP-System, dargestellt wurde. Weiters wurde in diesem Kapitel auch das Zusammenspiel von drei verschiedenen Programmiersprachen (Java – Passwort auslesen, vb.net – SAP GUI Scripting, ABAP – Transaktionen im SAP-System) behandelt.

7 Evaluation

Als letzter Teil der Umsetzung erfolgte die Evaluation des implementierten Systems. In diesem Kapitel wird auf die verwendete Methodik eingegangen und im Zuge dessen die Wahl begründet. Abschließend erfolgen die Präsentation und Interpretation der Ergebnisse.

7.1 Methodik

Das Ziel der Umsetzung war es in erster Linie, die Arbeit für die Benutzer*innen zu erleichtern und sich wiederholende Arbeitsschritte zu automatisieren. Um das Erreichen der Zielsetzung evaluieren zu können, sollte die User Experience gemessen werden. Unter User Experience versteht man das Erlebnis, welches die Anwender*innen bei der Verwendung des Produktes empfinden bzw. während der Verwendung erleben. Daraus kann auch schlussgefolgert werden, dass ein Produkt, welches mit positiven Erlebnissen verbunden wird, mit hoher Wahrscheinlichkeit regelmäßigere Verwendung findet, als dies bei einem negativ behafteten Produkt der Fall wäre. Sind die Anwender*innen der Anwendung gegenüber positiv gestimmt, werden auch etwaige Fehler und Mängel einfacher toleriert als bei einer entgegengesetzten Haltung.

Im Alltag erleben wir täglich eine Vielzahl an Ereignissen. Diese Ereignisse bleiben durch die Emotionen in stärkerer oder schwächerer Erinnerung. Die Emotionen beeinflussen auch die Erwartungen an eine Anwendung. Ein*e Anwender*in, welcher*e bisher noch keine Erfahrungen mit Chatbots gemacht hat, wird der Verwendung anders gegenüberstehen, als jener*e, der*die schon positive oder negative Erfahrungen gemacht hat. (Moser 2012, S. 3–5)

Bei der Evaluation der UX stehen eine Vielzahl von Möglichkeiten zur Verfügung. Einige dieser Möglichkeiten wären Feldtestungen oder Labortestungen. Eine weitere und zugleich auch eine sehr verbreitete Methodik stellt die Verwendung eines Fragebogens dar. Die Vorteile dieser Art liegen darin, dass es eine breite Palette an standardisierten Ausarbeitungen gibt und in einer kurzen Zeitspanne eine Vielzahl an Nutzern*innen befragt werden kann.

Ob der Fragebogen die richtige Wahl der Methodik darstellt, hängt laut Schrepp, Moser und Thomaschewski davon ab, inwiefern eine der nachfolgenden Fragestellungen beantwortet werden möchte: (Schrepp; Hinderks; Thomaschewski. 2016)

- Ist die User Experience des Produkts ausreichend?
- Ist die neue Version des Produkts in Bezug auf UX besser als die alte?
- Ist das Produkt in Bezug auf UX besser als die Konkurrenzprodukte?
- Wo liegen die generellen Stärken und Schwächen des Produkts?

Wie bereits eingangs erwähnt, war die Zielsetzung, die Automatisierung von sich wiederholenden Arbeitsschritten in Form eines Chatbots zu realisieren. Somit konnte das Ziel der Evaluierung auf den ersten Punkt der vorherigen Aufzählung, der ausreichenden UX, zurückgeführt werden.

Nach der Wahl der entsprechenden Methodik war es nötig, den passenden Fragebogen zu finden. Die Fragebögen teilen sich in verschiedene Dimensionen auf. Basierend auf den

relevanten Dimensionen sollte die Wahl des Fragebogens getroffen werden. Folgende Tabelle lag der Entscheidung zugrunde:

Dimension	UEQ	AttrakDiff2	VisAWI	meCUE	SUS	SUMI
Gesamtbeurteilung	--	--	--	--	Vollständig	Vollständig
Attraktivität	Vollständig	Vollständig	--	--	--	--
Effizienz	Vollständig	Teilweise	--	--	Teilweise	Vollständig
Durchschaubarkeit	Vollständig	Teilweise	Teilweise	Vollständig	Teilweise	Vollständig
Steuerbarkeit	Vollständig	Teilweise	--	--	--	Vollständig
Stimulation	Vollständig	Vollständig	--	--	--	--
Originalität	Vollständig	--	Vollständig	--	--	--
Identität	--	Vollständig	--	Vollständig	--	--
Schönheit	--	--	Vollständig	Vollständig	--	--
Wertigkeit	--	--	Vollständig	--	--	--
Nützlichkeit	--	--	--	Vollständig	--	Vollständig
Verbundenheit	--	--	--	Vollständig	--	--
Immersion	--	--	--	Vollständig	--	--
Emotionen	--	--	--	Vollständig	--	Vollständig

Abbildung 31: Fragebögen und Dimensionen (Schrepp; Hinderks; Thomaschewski. 2016, S. 3)

Die Entscheidung über die Wahl des Fragebogens fiel zwischen SUMI, UEQ und meCUE. Es wurden die erwähnten Fragebogen in Betracht gezogen, da diese eine standardisierte Möglichkeit der UX Testungen bieten. Da der UEQ Fragebogen die Dimensionen hinsichtlich der Emotionen nicht bediente, was in der Botentwicklung ein wichtiges Themenfeld darstellt, kam dieser schlussendlich nicht in Frage. Weiteres bediente der meCUE Fragebogen zwar die Dimensionen der Emotionen, jedoch nicht jene hinsichtlich der Effizienz und Gesamtbeurteilung. Dadurch fiel die Entscheidung auf den SUMI Fragebogen, da dieser die wichtigsten Dimensionen für die Umsetzung der Arbeit abdeckte, welche nachfolgend beschrieben werden:

- Effizienz – Dadurch, dass der Chatbot die Aufgaben automatisieren sollte, welche zuvor manuell ausgeführt wurden, ist die Effizienz von sehr großer Bedeutung. Die Akzeptanz würde stark darunter leiden, wenn die Aufgaben mit der gleichen Qualität wie zuvor ausgeführt werden.
- Durchschaubarkeit – Die Funktionalitäten und somit die Nutzung des Chatbots sollten für jeden in der Zielgruppe schnell und einfach sichtbar gemacht werden.
- Steuerbarkeit – Dem*der Benutzer*in soll während der Verwendung das Gefühl vermittelt werden, dass die Kontrolle zu jeder Zeit auf seiner*ihrer Seite liegt.
- Nützlichkeit – Dem*der Nutzer*in sollen anhand des Chatbots die repetitiven Aufgaben abgenommen werden, um dadurch zusätzliche zeitliche Ressourcen zu schaffen.
- Emotionen – Die Emotionen sind vor allem in der Botentwicklung eine sehr wichtige Komponente, da sich der*die Benutzer*in mit dem Chatbot unterhält und diesem Anweisungen gibt. (Schrepp; Hinderks; Thomaschewski. 2016, S. 2–3)

Die SUMI (Software Usability Measurement Inventory) Methode wurde ursprünglich dafür entwickelt, die Nutzerzufriedenheit einer Anwendung zu messen. Mit der Entwicklung über die Jahre wurde sie aber immer mehr dahingehend verwendet, um die User Experience zu evaluieren. Bei der Entwicklung des Fragebogens sollten genau jene Dimensionen abgedeckt werden, welche vorhergehend beschrieben wurden (Effizienz, Durschaubarkeit, Steuerbarkeit, Nützlichkeit, Emotionen). Als Ergebnis entstand ein aus 50 Fragen bestehender Fragebogen. Für die Bewertung der Fragen wird eine 3-stufige Skala verwendet mit den Werten: Zustimmen, Unentschlossen, nicht Zustimmen. („SUMI Background Reading“ o. J.)

Detailliert setzt sich der Fragebogen aus fünfzig Fragen zusammen, welche in fünf unterschiedliche Gruppen unterteilt werden. Die Gruppen werden wie folgt von SUMI beschrieben:

- Effizienz – beschreibt das Gefühl der Testperson in Bezug darauf, Aufgaben schnell, einfach und effektiv erledigen zu können.
- Beeinflussung – beschreibt das Gefühl, welches die Testperson während der Nutzung verspürt (Stress, Frustration, Freude, …)
- Hilfsbereitschaft – bezieht sich auf die Kommunikation mit dem*der Anwender*in und die Unterstützung bei der Lösung von Problemen
- Kontrolle – bezieht sich darauf, dass sich die Software bei der Eingabe von Befehlen konsistent verhält und die erwarteten Aktionen ausführt.
- Erlernbarkeit – bezieht sich auf das Gefühl des*der Anwenders*in die Software zu erlernen.

Für den Fragebogen war es nötig, eine Grundlage zur Beantwortung der Fragen zu schaffen. Somit musste ein Konzept erarbeitet werden, welches festlegt, wie die Testungen erfolgen sollten. Hierbei wurde auf das Konzept des speed dating zurückgegriffen. Bei der ursprünglichen Art des speed dating geht es darum, dass Personen eine Reihe an verschiedenen Treffen mit unterschiedlichen Gesprächspartnern durchführen, um basierend auf diesen Erfahrungen eine Entscheidung zu treffen. Nach gleichem Prinzip funktioniert auch die Methodik des speed dating in UX Testungen. Das speed dating als Designmethodik zielt darauf ab, den*die Benutzer*in einer gleichen Situation auszusetzen, deren Ziel es ist, mögliche Funktionalitäten, sowie eine Idee über eine mögliche Zukunft zu vermitteln. Die zwei Hauptbestandteile des speed dating in UX Testungen bilden das Storyboard und das user enactment (UE). (Zimmerman; Forlizzi 2017, S. 31)

Bei der Verwendung der Storyboard Technik werden Szenarien erstellt, in denen sich die Zielgruppe einfach zurechtfindet. Ziel ist es, dass eine Aktivität vorgelegt wird, deren Ziel eindeutig definierbar ist. Beispielsweise wäre das Ziel bei dem Szenario, in der sich die Testperson auf einem Kundensystem anmelden soll, klar definierbar. Dies wäre in diesem Fall, dass die Anmeldung erfolgreich abgeschlossen werden konnte. Das Storyboard eignet sich zur Untersuchung eines breiten Kontextes, da der Testperson bei der Ausführung die größtmögliche Freiheit gelassen wird.

In der Technik des user enactments wird eine Skript verwendet, nach dem die Testpersonen geführt werden. In dem Skript werden die Rolle, die Aktionen und der Dialog definiert. Somit wird der Raum der Freiheit stärker eingegrenzt als bei der Storyboard Technik. Diese Art

eignet sich meist am besten für eine tiefere Untersuchung, die sich auf Testpersonen spezialisiert, die eine bestimmte Rolle spielen sollten. (Zimmerman; Forlizzi 2017, S. 46)

Bei der Verwendung von Chatbots hat jeder*jede Anwender*in eine andere Erwartungshaltung, welche auf den bisher gemachten Erfahrungen beruht. Dadurch unterscheidet sich auch die Kommunikation der Testpersonen mit dem Chatbot hinsichtlich Ausdrucksweise, Wortwahl oder Satzkonstruktionen. Mit der Verwendung der Storyboard Methodik sollte der größtmögliche Freiraum gelassen werden und nur die Definition des Ziels erfolgen, damit der Weg zur Erreichung von jedem*r Anwender*in eigens geschaffen werden konnte. Für die Identifikation der Zielgruppe mit den Szenarien wurden alltägliche Situationen abgebildet, die von den Anwendungsfällen aus Abschnitt 1.2.3.3.1 abgedeckt werden. Somit war es nötig, in jeder Story einen Start- und Endpunkt abzustecken. Der Endpunkt war gleichbedeutend mit dem erreichten Ziel. Folglich ergaben sich folgende Storyboards:

- Die Testperson wurde in das Szenario versetzt, dass entweder schnellstmöglich die Anmeldung für ein System ausgeführt werden sollte oder eine bestehende Session maximiert werden soll, um Daten abfragen zu können. Dieses Szenario deckt den Anwendungsfall des Anmeldens an einem Kundensystem ab.
- In einem anderen Szenario bekommt die Testperson einen Anruf eines*r Mitarbeiters*in (imaginär) und wird aufgefordert sich vom System abzumelden, damit er*sie sich auf dem System anmelden kann. Anhand dieses Szenarios soll der Anwendungsfall für des Abmelden an einem Kundensystem modelliert werden.
- In einem weiteren Szenario erhält die Testperson eine E-Mail von einem Kunden (imaginär), dass die Entwicklung vom DEV-System auf das QAS-System transportiert werden kann. Die Testperson muss sich daher eine Liste der Transports anzeigen lassen und den entsprechenden Auftrag, basierend auf der Beschreibung, auswählen und freigeben. Anschließend soll dieser auch sofort importiert werden, da diese Schritte in der täglichen Arbeit meist aufeinander folgen. In diesem Szenario werden die drei Anwendungsfälle zur Anzeige der Transportliste sowie Freigabe und Import eines Transportauftrages abgedeckt.
- Das letzte Szenario wird ebenfalls durch einen Anruf (imaginär) gestartet, in dem der Kunde die Testperson auffordert, das Kennwort für einen bestimmten*e Benutzer*in zu ändern und diesen auch zu entsperren. Dies deckt sich mit dem Anwendungsfall zur Änderung eines*r Benutzers*in.

Die Szenarien wurden in der beschriebenen Reihenfolge durchgeführt und sind unabhängig voneinander zu betrachten. Die grafische Aufbereitung sowie die detaillierte Wortwahl kann aus dem Anhang entnommen werden.

7.2 Ergebnisse

Der Fragebogen zur Evaluation setzt sich aus den genannten Punkten in Abschnitt 7.1 zusammen. Basierend darauf wurde die Software anhand dieser Dimensionen gegen die standardisierte SUMI Datenbank evaluiert. Dadurch wird die Anwendung mit den durchschnittlichen „state of the art“ Anwendungen verglichen. Basierend auf den eingegebenen Werten

erfolgt die Generierung einer Skala, welche einen Mittelwert von 50 und eine Standardabweichung von 10 darstellt. („SUMI Background Reading“ o. J.)

Nachfolgend sollen die Ergebnisdaten erläutert werden, bevor anschließend in Abschnitt 7.3 die Interpretation erfolgt.

Abbildung 32 zeigt die Ergebnisse der durchgeführten Tests, kalkuliert auf Basis der Werte in der SUMI Datenbank. Abhängig von der Stichprobe variiert auch die Standardabweichung. Der Umfang der Testpersonen gestaltete sich im Rahmen von zehn Personen, welcher als empfohlenes Mindestmaß angegeben wird.

	Mean	St Dev	Median	IQR	Minimum	Maximum
Global	59.00	10.93	60.0	19.0	43	73
Efficiency	54.80	9.99	55.5	15.0	35	67
Affect	54.80	8.34	54.0	10.0	41	69
Helpfulness	59.00	10.40	60.5	17.0	38	70
Controlability	57.60	7.53	59.5	11.0	43	66
Learnability	60.90	5.95	61.5	7.0	50	69

Abbildung 32: Ergebnisse SUMI (eigene Darstellung)

Der Mittelwert über alle Dimensionen hinweg beläuft sich auf 59,0 und liegt somit über jenem in der SUMI Datenbank (50). Das beste Ergebnis (60,9) stellt sich in der Dimension der Erlernbarkeit, gefolgt von jenem der Hilfsbereitschaft (59,0), dar. Als schlechteste Dimensionen wurden die Effizienz und Beeinflussbarkeit bewertet.

Die Dimension der Hilfsbereitschaft, welche vorhergehend mit dem zweithöchsten Mittelwert aufgelistet wurde, wies in der Kategorie der Standardabweichung mit 10,4 den höchsten Wert auf. Die Erlernbarkeit, welche den besten Mittelwert erreichen konnte, hatte auch die niedrigste Standardabweichung.

Deckend zur Kategorie der Standardabweichung wies die Dimension der Hilfsbereitschaft auch den größten Interquartilsabstand auf. Der geringste Wert wurde von der Erlernbarkeit erreicht, welcher 10 Einheiten unter dem Wert aller Dimensionen lag.

Auf die Frage zur Bedeutung der Anwendung antworteten jeweils vier Personen mit einer sehr hohen bzw. hohen Bedeutung. Für zwei Personen stellte die Anwendung keine wichtige Entwicklung dar. Das beste Ergebnis, basierend auf diesen Gruppen, konnte bei jenen Testpersonen erreicht werden, welche die Anwendung als wichtig einstuften, gefolgt von der Gruppe, die diese als nicht sehr wichtig betrachteten. In der Gruppe, welche die Bedeutung als sehr wichtig ansah, wurde das schlechteste Ergebnis erzielt. Jedoch erzielten die Dimensionen der Effizienz, der Erlernbarkeit und der Kontrollierbarkeit das beste Ergebnis.

	n	G	E	A	H	C	L
Extremely important	*4*	56.5	59.5	55.0	54.8	61.2	63.5
Important	*4*	61.2	55.2	55.8	61.8	56.0	59.0
Not very important	*2*	59.5	44.5	52.5	62.0	53.5	59.5
Not important at all	*0*						

Abbildung 33: Ergebnisse nach Bedeutung für die Testgruppe

Als positive Eigenschaft der Anwendung wurden mehrfach die Zeitersparnis sowie die intuitive Bedienung genannt. Bei den Kritikpunkten fielen die Antworten vielfältiger und ausführlicher aus. Nachfolgend werden diese Aussagen zusammengefasst:

- Erklärung von Parametern – bei einigen Arbeitsschritten war nicht klar, welche Parameter eingegeben werden können.
- Aufrechterhaltung des Konversationsflusses – bei einer falschen Eingabe bzw. einer unverständlichen Eingabe mussten die Aufgaben teilweise neu gestartet werden.
- Unterstützung von Dialekt und Abkürzungen – Eingaben in Dialektform und Abkürzungen wie PW (Passwort) konnten nicht verstanden und interpretiert werden.
- Vorschläge bei fehlerhaften Eingaben – bei fehlerhaften Eingaben erfolgte keine Unterstützung hinsichtlich weiterem Vorgehen.

7.3 Interpretation

Der Mittelwert der Befragung lag über 50 und indiziert somit ein grundsätzlich gutes Ergebnis der Implementierung. Um die Interpretation übersichtlich zu gestalten, wird jede Dimension einzeln behandelt um abschließend ein Fazit zu erörtern.

Die Dimension der Lernfähigkeit wies das beste Ergebnis auf. Daraus lässt sich schließen, dass die Anwendung von allen intuitiv genutzt werden konnte, ohne dass es nötig war, zuvor eine Dokumentation zu lesen oder eine Einschulung zu erhalten. Zusätzlich war die Standardabweichung die geringste, woraus sich schließen lässt, dass alle Teilnehmer*innen in dieser Dimension eine einheitliche Meinung vertraten. Ableitend auf dieser Tatsache stellt der Bot mit seinen Hilfestellungen und der Art der Antworten eine guten Grundlage dar, dass in Zukunft auch Personen damit arbeiten können, die diese Technologie noch nicht genutzt haben.

Das zweitbeste Ergebnis konnte in der Dimension der Hilfsbereitschaft erreicht werden. Auf den ersten Blick scheint somit, dass das Verständnis der Nachrichten und die Kommunikation einfach verständlich sind. Die hohe Standardabweichung deutet jedoch daraufhin, dass dies nur für einzelne Testpersonen der Fall war. Dies lässt sich auch dadurch belegen, dass während der Durchführung die Wortwahl der Testpersonen sehr unterschiedlich ausfiel. Während einige jeden Anwendungsfall mit dem Anzeigen der Hilfe starteten und somit eine Idee der Wortwahl bzw. Befehle erhielten, verwendeten andere keinen üblichen Satzbau, wie es in der Konversation mit einem menschlichen Konversationspartner der Fall wäre. In dieser Dimension liegt das Verbesserungspotential vor allem darin, dass eine breitere Möglichkeit an

Wortkonstellationen unterstützt werden soll. Weiters kann die Möglichkeit der Unterstützung von Dialekten in Betracht gezogen werden, jedoch stellt sich hier die Frage, ob die Anwendergruppe breit genug wäre. Ein positiver Aspekt dieser Umsetzung wäre mit Sicherheit, dass dadurch positive Emotionen und eine Verbindung hervorgerufen werden könnten.

Bei der Dimension der Kontrollierbarkeit konnte ebenfalls ein überdurchschnittliches Ergebnis erreicht werden. Weiters waren hier die Standardabweichung und der Interquartilsabstand sehr gering, was darauf schließen lässt, dass das System den Erwartungen entsprechend reagiert hat. Für die Anwender*innen war es somit einfach, zwischen den Teilaufgaben zu wechseln und auch neue Funktionen zu erlernen. Dies kann als sehr wichtig angesehen werden, da es auch ein Ziel der Umsetzung war, dass jeder Anwendungsfall eine in sich geschlossene Aufgabe mit klarem Beginn und Abschluss bildet. Darüber hinaus kann aus dem Ergebnis resultiert werden, dass der*die Anwender*in zu jeder Zeit der Bedienung das Gefühl der Kontrolle über die Anwendung verspürte, was ein wichtiger Faktor für die Akzeptanz darstellt.

Die beiden Dimensionen der Kontrollierbarkeit und der Effizienz schnitten mit einem Wert von 54,8 am schlechtesten ab. Die Personen hatten somit nicht das Gefühl, dass die Aufgaben effizienter als zuvor gelöst werden können. Da diese Dimensionen eine sehr starke Abhängigkeit haben, führte das Gefühl der fehlenden Effizienz zu einer negativen Assoziation und spiegelte sich in der Beeinflussbarkeit wider. Die fehlende Effizienz lässt sich auch in den Beobachtungen und Aussagen der Testpersonen wiederfinden. Bei einigen Testpersonen war bei einer fehlerhaften Eingabe nicht klar, welche Schritte nötig waren, um die Aufgabe weiterhin ausführen zu können und worin der Fehler lag. Von Seiten des Chatbots kam somit der Konversationsfluss abhanden. Dies führte dazu, dass der*die Benutzer*in die Aufgabe nochmals starten mussten. Zusätzlich waren die möglichen Parameter nicht bei jeder Aufgabe klar kommuniziert und zum anderen konnte die Aufmachung der Buttons nicht erkennbar für die Nutzer*innen gestaltet werden. In diesen Dimensionen liegt das größte Verbesserungspotenzial des Chatbots. Die fehlende Effizienz wirkt sich sehr stark auf die Akzeptanz aus, da die Anwender*innen das Gefühl haben, die Aufgaben manuell besser und schneller erledigen zu können. In diesem Bereich spielen vor allem der Konversationsfluss und die Parameter eine wichtige Rolle. Bei einer fehlerhaften Eingabe sollte der Chatbot in Zukunft entsprechend reagieren und die Nutzer*innen auf Fehler hinweisen und eine Lösung vorschlagen. Weiteres müssen die verfügbaren Parameter deutlicher hervorgehoben werden. Eine Möglichkeit wäre, dies auch hinsichtlich der Konsistenz, über Buttons zu realisieren. Ebenfalls bedarf es einer Verbesserung der grafischen Aufmachung und Berücksichtigung der Wirkung von Farben auf die Personen. Die Farben könnten gegebenenfalls auf die Stimmung des*der Anwenders*in angepasst werden, um somit die Emotionen zu steuern und etwaigen negativen Gefühlen entgegenwirken zu können.

Für die Unterteilung der Gruppen, welche die Anwendung nach Bedeutung des Nutzens evaluierten, wurde das schlechteste Ergebnis in jener Gruppe erreicht, welche die Wichtigkeit als sehr hoch einstufte.

In der Gruppe, welche die Bedeutung als weniger wichtig einstufte, war vor allem das Ergebnis der Effizienz unterdurchschnittlich und wich stark von den anderen Gruppen ab, was vermutlich auch dazu führte, dass die Anwendung als weniger wichtig angesehen wurde, da die Aufgaben nicht mit jener Effizienz erledigt wurden, die vorher bei der manuellen Ausführung gegeben war. Jedoch bewertete jene Gruppe die Hilfsbereitschaft am besten. Daraus resultiert,

dass die Anwendung bei der Erledigung der Aufgaben eine gute Kommunikation bietet. Jene Kategorie der Hilfsbereitschaft wurde von der Gruppe mit hoher Wichtigkeit mit dem schlechtesten Wert bewertet. Die hohe Standardabweichung erklärt sich vor allem durch die zuvor erwähnten Faktoren wie fehlende Erklärung von Parametern, Abbrechen des Konversationsflusses sowie mangelhafte grafische Aufmachung.

7.4 Ausblick

Im Mittelpunkt der weiteren Vorgehensweise steht die Optimierung der bestehenden Anwendungsfälle. In den nächsten Schritten sollte das Hauptaugenmerk auf die Verbesserung der Dimensionen der Effizienz und der Kontrollierbarkeit gelegt werden. Es bedarf einer verbesserten Steuerung des Konversationsflusses. Der Chatbot soll den*die Anwender*in auf Unverständnis bei der Eingabe hinweisen und den Benutzern*innen Vorschläge hinsichtlich weiterem Vorgehen anbieten.

In den weiteren Prozessdurchläufen sollte auch die Datengrundlage von Ausdrücken und Entitäten erweitertet werden. Anhand der Durchführung der Testungen konnten neue Eindrücke hinsichtlich der Kommunikation der Zielgruppe mit dem Chatbot gewonnen werden. Diese Ausdrucksweisen und Argumentationen können nun eingearbeitet werden und somit die Qualität der Konversation stark verbessern, da es für viele Anwender*innen sehr mühsam war, wenn der Bot deren Eingabe nicht verstehen konnte, obwohl diese ihrer Ansicht nach klar formuliert war.

Durch die Tatsache, dass sich die Zielgruppe auf hausinterne Mitarbeiter*innen beschränkt, ist es möglich, den Chatbot sehr passgenau auf diese Gruppe anzupassen. Dies kann durch eine Vielzahl an Testungen und in deren Folge auch Verbesserungen erreicht werden.

Im Hinblick auf die Gesamtarchitektur können Überlegungen dahingehend angestellt werden, dass die Funktionalität des SAP GUI Scripting in die RPA Plattform ausgelagert wird. Dies würde aber auch eine weitestgehende Überarbeitung der Gesamtarchitektur mit sich bringen.

Zusätzlich besteht auch in Zukunft die Möglichkeit, weitere Anwendungsfälle zu integrieren. Dafür müssten auf Seiten der SAP CAI die Ausdrücke und Entitäten gepflegt werden. Weiters bedarf es der Einarbeitung in das bestehende Programm des SAP GUI Scripting und der Implementierung eines Programmes, welches in weiterer Folge die Operationen ausführt.

Basierend auf der Umsetzung und der Evaluation kann das Ergebnis der Arbeit als zufriedenstellend betrachtet werden. Diese Aussage wird darauf basierend getroffen, dass die Umsetzung als Greenfield Projekt gestartet wurde und somit weder eine Wissensbasis über die Konzepte noch über die technologischen Plattformen vorhanden war. Durch die Umsetzung konnten in erster Linie die Konzepte, welche eine tragende Rolle hinsichtlich der User Experience spielen, und in weiterer Folge die von der SAP zur Verfügung gestellten Technologieplattformen aufgearbeitet und verwendet werden. Als Ergebnis dieser Umsetzung konnte ein Prototyp erarbeitet werden, welcher die Grundlage für zukünftige Entwicklungen darstellen kann. Durch das große Potential der Technologie wird die Thematik auch weiterhin verfolgt und laufend erweitert werden.

Literaturverzeichnis

Adamopoulou, Eleni; Moussiades, Lefteris (2020): „An Overview of Chatbot Technology.“ In: Artificial Intelligence Applications and Innovations. Cham: Springer International Publishing (= IFIP Advances in Information and Communication Technology), S. 373–383. Online im Internet: DOI: 10.1007/978-3-030-49186-4_31

Amelia (2020): What’s the Difference Between a Chatbot and Conversational AI? Online im Internet: URL: https://www.youtube.com/watch?v=xHAnLceQnT0 (Zugriff am: 04.05.2022).

Benke, Ivo; Gnewuch, Ulrich; Maedche, Alexander (2022): „Understanding the impact of control levels over emotion-aware chatbots.“ In: Computers in Human Behavior, 129 (2022), S. 107122. Online im Internet: DOI: https://doi.org/10.1016/j.chb.2021.107122

Braasch, Bert (Hrsg.) (2016): SAP HANA – Administration: Architektur, Werkzeuge und nötiges Hintergrundwissen: alle Phasen des Lebenszyklus von der Installation bis zum Update: praktische Anleitungen zu Sizing, Monitoring, Backup, Recovery u.v.m. 1. Auflage. Bonn: Rheinwerk Publishing.

Bravo, David (2019): Mynd. Anthropomorphismus: Ein Design-Prinzip, das Sie kennen sollten. Online im Internet: URL: https://blog.mynd.com/de/anthropomorphismus/ (Zugriff am: 26.06.2022).

„Chatbot-Typen und Einsatz-Szenarien | SpringerLink“ (o. J.): Chatbot-Typen und Einsatz-Szenarien | SpringerLink. Online im Internet: URL: https://link.springer.com/chapter/10.1007/978-3-658-25494-0_2 (Zugriff am: 17.05.2022).

Gahm, Hermann (Hrsg.) (2016): ABAP-Entwicklung für SAP HANA: mit SAP NetWeaver AS ABAP 7.4/7.5 Anwendungen für SAP HANA entwickeln; Entwicklungsumgebung, Datenbankprogrammierung und Einbindung nativer HANA-Objekte; erweiterte Funktionen wie Geodaten, Fuzzy-Suche und Predictive Analysis. 2., aktualisierte und erweiterte Auflage. Bonn: Rheinwerk Publ.

Gentsch, Peter (2019): Künstliche Intelligenz für Sales, Marketing und Service: Mit AI und Bots zu einem Algorithmic Business – Konzepte und Best Practices. Wiesbaden: Springer Fachmedien Wiesbaden. Online im Internet: DOI: 10.1007/978-3-658-25376-9 (Zugriff am: 03.05.2022).

Hill, Jennifer; Ford, W. Randolph; Farreras, Ingrid G. (2015): „Real conversations with artificial intelligence: A comparison between human–human online conversations and human–chatbot conversations.“ In: Computers in Human Behavior, 49 (2015), S. 245–250. Online im Internet: DOI: https://doi.org/10.1016/j.chb.2015.02.026

Hundertmark, Sohpie (2020): Chatbot Umfrage DACH 2020. Online im Internet: URL: https://www.hundertmark.ch/chatbot-umfrage-dach-2020/ (Zugriff am: 13.05.2022).

Intelligence, Insider (2016): Business Insider. 80% of businesses want chatbots by 2020. Online im Internet: URL: https://www.businessinsider.com/80-of-businesses-want-chatbots-by-2020-2016-12 (Zugriff am: 03.05.2022).

Kabel, Peter (2020): Dialog zwischen Mensch und Maschine: Conversational User Interfaces, intelligente Assistenten und Voice-Systeme. Wiesbaden: Springer Fachmedien Wiesbaden. Online im Internet: DOI: 10.1007/978-3-658-29585-1 (Zugriff am: 11.05.2022).

Kaiser, Markus; Buttkereit, Aline-Florence; Hagenauer, Johanna (2019): Journalistische Praxis: Chatbots: Automatisierte Kommunikation im Journalismus und in der Public Relation. Wiesbaden: Springer Fachmedien Wiesbaden (= essentials). Online im Internet: DOI: 10.1007/978-3-658-25494-0 (Zugriff am: 16.05.2022).

Katz, Yehuda (2021): Introduction | Handlebars. Online im Internet: URL: https://handlebarsjs.com/guide/ (Zugriff am: 18.05.2022).

Koch, Oliver; Stass, Guido W. (2022): Robotic Process Automation mit SAP. 1. 1. Aufl. Bonn: Rheinwerk Verlag.

Kohne, Andreas u.a. (2020): Chatbots: Aufbau und Anwendungsmöglichkeiten von autonomen Sprachassistenten. Wiesbaden: Springer Fachmedien Wiesbaden. Online im Internet: DOI: 10.1007/978-3-658-28849-5 (Zugriff am: 03.05.2022).

Lotze, Netaya (2016): Chatbots: Eine linguistische Analyse. Peter Lang International Academic Publishers. Online im Internet: DOI: 10.3726/b10402 (Zugriff am: 04.05.2022).

Mensio, Martino; Rizzo, Giuseppe; Morisio, Maurizio (2018): „The Rise of Emotion-aware Conversational Agents: Threats in Digital Emotions." In: Companion of the The Web Conference 2018 on The Web Conference 2018 - WWW '18. Lyon, France: ACM Press, S. 1541–1544. Online im Internet: DOI: 10.1145/3184558.3191607 (Zugriff am: 13.05.2022).

Methukupally, Sai (2019): SAP Conversational Ai chatbot architecture and implementation styles for varying data privacy constraints (On premise/Private cloud cases). SAP Conversational Ai chatbot architecture and implementation styles for varying data privacy constraints (On premise/Private cloud cases) | SAP Blogs. Online im Internet: URL: https://blogs.sap.com/2019/11/19/sap-conversational-ai-chatbot-architecture-and-implementation-styles-for-varying-data-privacy-constraints-on-premise-private-cloud-cases/ (Zugriff am: 17.05.2022).

Meyer, Dominik (2022): iRPA Desktop Agent not executing Cloud Factory Project | SAP Community. Online im Internet: URL: https://answers.sap.com/questions/13588000/irpa-desktop-agent-not-executing-cloud-factory-pro.html (Zugriff am: 24.05.2022).

Moser, Christian (2012): User Experience Design. Berlin, Heidelberg: Springer Berlin Heidelberg (= X.media.press). Online im Internet: DOI: 10.1007/978-3-642-13363-3 (Zugriff am: 26.05.2022).

Pamela (2018): Ai-enhanced customer experience. Chatbots and Conversational Agents, what is the difference? Online im Internet: URL: https://medium.com/ai-enhanced-customer-experience/chatbots-and-conversational-agents-what-is-the-difference-8f3b754b5042 (Zugriff am: 03.05.2022).

Rukavina, Stefanie u.a. (2016): „Affective Computing and the Impact of Gender and Age." In: PLoS ONE, 11 (2016), 3, S. e0150584. Online im Internet: DOI: 10.1371/journal.pone.0150584

SAP (2019): „SAP GUI Scripting API.“In: (2019), S. 302.

SAP (2022a): Concepts of SAP Conversational AI. SAP SE. Online im Internet: URL: https://help.sap.com/doc/9b639cad3d734675971ab22ed10bbf28/latest/en-US/UserGuide-ToConceptsOfSAPConversationalAI.pdf

SAP (2022b): „Desktop Studio Developer Guide.“In: (2022), S. 522.

SAP (o. J.): SAP-Transaktionen. Online im Internet: URL: https://help.sap.com/saphelp_nw73/helpdata/DE/41/7af4c2a79e11d1950f0000e82de14a/content.htm?no_cache=true (Zugriff am: 25.05.2022).

Schrepp, Martin; Hinderks, Andreas; Thomaschewski., Jörg (2016): „User Experience mit Fragebögen evaluieren - Tipps und Tricks für Datenerhebung, Auswertung und Präsentation der Ergebnisse.“In: (2016). Online im Internet: DOI: 10.18420/MUC2016-UP-0016 (Zugriff am: 26.05.2022).

Schwaiger, Roland (2019): Schrödinger programmiert ABAP. 3., aktualisierte und erweiterte Auflage. Bonn: SAP Press.

Stäcker, Oliver; Stanoevska-Slabeva, Katarina (2018): „Quo vadis Chatbots?“ In: Wirtschaftsinformatik & Management, 10 (2018), 6, S. 38–46. Online im Internet: DOI: 10.1007/s35764-018-0122-x

stevewhims (2021): IRunningObjectTable (objidl.h) - Win32 apps. Online im Internet: URL: https://docs.microsoft.com/en-us/windows/win32/api/objidl/nn-objidl-irunningobjecttable (Zugriff am: 24.05.2022).

stevewhims (o. J.): The Component Object Model - Win32 apps. Online im Internet: URL: https://docs.microsoft.com/en-us/windows/win32/com/the-component-object-model (Zugriff am: 26.06.2022).

Stucki, Toni; D’Onofrio, Sara; Portmann, Edy (2020): Chatbots gestalten mit Praxisbeispielen der Schweizerischen Post: HMD Best Paper Award 2018. Wiesbaden: Springer Fachmedien Wiesbaden (= essentials). Online im Internet: DOI: 10.1007/978-3-658-28586-9 (Zugriff am: 13.05.2022).

Suhr, Frauke (2020): Digitalisierung: Jedes vierte Unternehmen nutzt Chatbots. Jedes vierte Unternehmen nutzt Chatbots. Online im Internet: URL: https://de.statista.com/infografik/23494/umfrage-zu-chatbots-bei-unternehmen-in-deutschland/ (Zugriff am: 03.05.2022).

„SUMI Background Reading“ (o. J.): SUMI Background Reading. Online im Internet: URL: https://sumi.uxp.ie/about/sumipapp.html (Zugriff am: 27.05.2022).

„Was ist ein Chatbot?“ (2021): Was ist ein Chatbot? Online im Internet: URL: https://www.ibm.com/de-de/campaign/was-ist-ein-chatbot (Zugriff am: 03.05.2022).

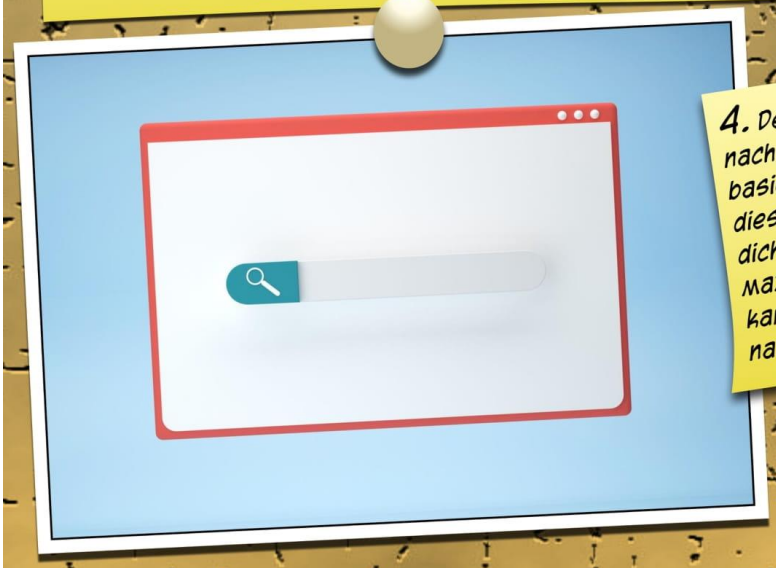
Zamora, Jennifer (2017): „I’m Sorry, Dave, I’m Afraid I Can’t Do That: Chatbot Perception and Expectations.“ In: Proceedings of the 5th International Conference on Human Agent Interaction. New York, NY, USA: Association for Computing Machinery (= HAI ’17), S. 253–260. Online im Internet: DOI: 10.1145/3125739.3125766 (Zugriff am: 16.05.2022).

Zimmerman, John; Forlizzi, Jodi (2017): „Speed Dating: Providing a Menu of Possible Futures.“ In: She Ji: The Journal of Design, Economics, and Innovation, 3 (2017), 1, S. 30–50. Online im Internet: DOI: https://doi.org/10.1016/j.sheji.2017.08.003

Anhang


Anmelden(1)
1. Du telefonierst mit einem Kunden und dieser hätte gerne ein neues Programm auf seinem Entwicklungssystem.
2. Du Möchtest dich so schnell wie Möglich anmelden oder eine aktive Session nutzen. Du hast momentan jedoch sehr viele Fenster offen und müsstest jetzt alles durchsuchen. Du wendest dich an den proBot.
3. Du öffnest den proBot und kommunizierst, dass dieser dich auf dem System (PTD) anmelden soll.

Anmelden(2)



4. Der proBot startet und sucht nach einer bestehenden Session, basierend auf der System ID. Ist diese nicht vorhanden meldet er dich auf dem System an und maximiert das Fenster. Jetzt kannst du durch das System navigieren.

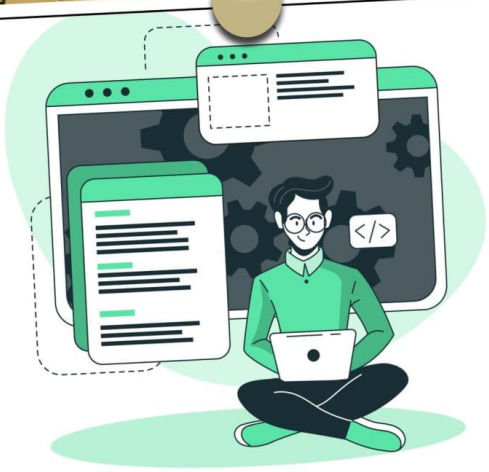
Abmelden(1)

1. Du bist gerade dabei ein Programm zu schreiben. Als eine Kolleginn dich darauf aufmerksam macht, dich von System abzumelden, damit sie sich dort einloggen kann.

2. Du weißt nicht mehr auf welchem Bildschirm sich die Session befindet und müsstest alles durchsuchen. Das würde dich aus dem Rhythmus bringen.

3. Du öffnest den proBot und kommunizierst, dass dieser dich vom System (PTD) abmelden sollte.

Abmelden(2)



4. Der proBot sucht nach einer entsprechenden Session und beendet diese dann für dich damit dieser du dich weiter auf dein Programm konzentrieren kannst.

Transport(1)

1. Du bekommst eine Mail in welcher der Kunde bestätigt, dass die Tests für dein Programm erfolgreich waren und nun transportiert werden kann.

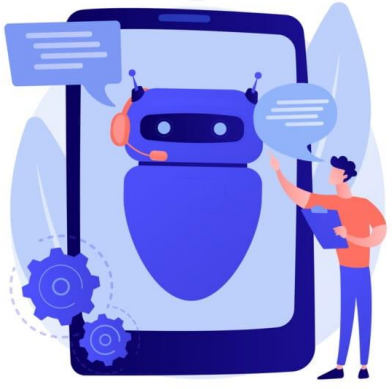
2. Du weißt die Nummer des Auftrages nicht mehr auswendig und befindest dich zudem auf einem anderen System.

3. Du öffnest den proBot und fragst diesen nach einer Liste der Transporte für deinen User (PT_DM) auf dem System (PTD).

Transport(2)

4. Du suchst nach den entsprechenden Auftrag mit dem Text: "Entwicklung" und kopierst/merkst dir die Transportnummer.

5. Du teilst dem proBot mit, dass du gerne den Transport mit der entsprechenden Nummer auf dem System PTD freigeben würdest.

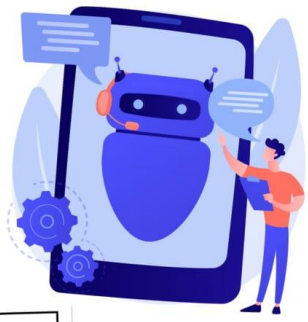


6. Der proBot erfasst alle erforderlichen Parameter von dir und führt den Transport aus. Er fragt dich im Anschluss ob du den Auftrag gerne auf einem anderen System importieren möchtest.

Transport(3)

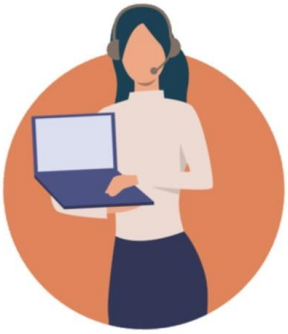
7. Du nutzt diese Möglichkeit und möchtest, dass der Auftrag auf HFX importiert wird.

7. Der proBot hat sich die Transportnummer gemerkt. Das System, auf welchem der Import erfolgt, hat er von dir mitgeteilt bekommen. Er startet mit dem Import und informiert dich nach Abschluss.



8. Danach informierst du den Kunden per Mail, dass der Auftrag nun erfolgreich Transportiert wurde und das Programm genutzt werden kann.

Transport(1)

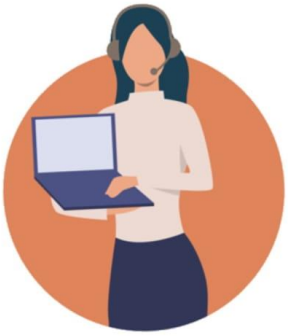


1. Während du gerade mit dem Kunden telefonierst fällt dir ein, dass du noch gerne den Auftrag IEGK912123 auf dem System PTD importieren möchtest.

2. Du gibst dem Bot die Anweisung den Import zu starten und den Auftrag auf dem System zu importieren.

3. Nachdem der proBot alle Parameter erfasst hat, wird der Import gestartet.

Transport(2)



4. Währenddessen kannst du dich weiter um den Kunden kümmern und der Auftrag wird vom proBot importiert.

Benutzer(1)

1. Du bekommst einen Anruf von einem Kunden, dass er sein Passwort vergessen hat und du das Passwort für den Benutzer PT_TEST auf dem System PTD zurücksetzen solltest.

2. Du befindest dich jedoch gerade in einem anderen Meeting und hast keine Zeit dafür. Jedoch muss diese Aufgabe sofort erledigt werden, damit der Kunde wieder arbeiten kann.

3. Du wendest dich an den proBot und sagst diesem, dass dieser das Passwort für den Benutzer zurücksetzen und ihn per Mail benachrichtigen soll.

Benutzer(2)

4. Der proBot startet mit der Aufgabe und schickt automatisch eine Benachrichtigung an die hinterlegte E-Mail Adresse.

Einloggen auf System

<table>
<tr><td>Name</td><td>Einloggen auf ein System</td></tr>
<tr><td>Beschreibung</td><td>Benutzer*in möchte sich auf seinem Notebook in ein System einloggen</td></tr>
<tr><td>Akteure</td><td>• Benutzer*in
• Bot</td></tr>
<tr><td>Voraussetzungen</td><td>• Der Desktop Agent befindet sich im „unattended“ Modus und ist aktiviert.
• Die Keepass Passwort Datenbank ist dahingehend gepflegt, dass die Systemeinträge mit jenen aus dem SAP Logon übereinstimmen.
• Scripting Parameter im SAP- System sind gepflegt</td></tr>
<tr><td>Auslöser</td><td>User*in kommuniziert mit dem Bot und teilt dem Bot mit, dass dieser sich, basierend auf den erfassten Parametern in ein System einloggen sollte.</td></tr>
<tr><td>Essenzielle Schritte</td><td>
<table>
<tr><th>Step</th><th>Intention der Systemumgebung</th><th>Reaktion des Systems</th></tr>
<tr><td>1</td><td>Benutzer*in ruft den Bot zur Eingabe auf</td><td>System zeigt das Eingabefenster des Bots an und begrüßt Benutzer*in</td></tr>
<tr><td>2</td><td>Benutzer*in teilt dem Bot auf welches System sich dieser einloggen sollte. Der Systemname ist dabei obligatorisch und ist jener nachdem gesucht wird. Es können optional noch Anmeldesprache und Client angegeben werden.</td><td>Bot löst den entsprechenden API Trigger in der Cloud Factory aus. Cloud Factory startet das Szenario im Desktop Agent und gibt eine Rückmeldung</td></tr>
<tr><td>3</td><td>Benutzer*in liest entsprechende Rückmeldung und weist den Bot auf weitere Schritte hin.</td><td>Verarbeitung der weiteren Eingaben oder Beenden.</td></tr>
</table>
</td></tr>
<tr><td>Ausnahmefälle</td><td>
<table>
<tr><th>Step</th><th>Auslöser</th><th>Reaktion</th></tr>
<tr><td>2</td><td>Benutzer*in teilt dem Bot nicht alle erforderliche Parameter mit.</td><td>Der Bot stellt Rückfrage und weist auf fehlenden Parameter hin.</td></tr>
</table>
</td></tr>
</table>

	2	Der Bot kann die Anmeldung am System nicht durchführen.	Logging der möglichen Ursachen: • GUI Scripting auf System deaktiviert • Benutzer*in bereits angemeldet auf gleicher Workstation -> GUI maximieren • Benutzer*in auf anderer Workstation angemeldet -> Rückmeldung, dass händische Aktion gefordert wird • Keine Verbindung möglich • Benutzer*in und Passwort können nicht ausgelesen werden • Fehlende Parameter
Nachbedingung	Benutzer*in ist auf dem System angemeldet oder erhält Rückmeldung über etwaige Problematiken.		
Kommentare/Anmerkungen			

Ausloggen aus System

Name	Ausloggen aus einem System		
Beschreibung	Benutzer*in möchte sich auf seinem Notebook von einem System abmelden		
Akteure	• Benutzer*in • Bot		
Voraussetzungen	• Der Desktop Agent befindet sich im „unattended“ Modus und ist aktiviert. • Scripting Parameter im SAP- System sind gepflegt		
Auslöser	Benutzer*in kommuniziert mit dem Bot und teilt dem Bot mit, dass dieser sich, basierend auf den erfassten Parametern von einem System abmelden sollte.		
Essenzielle Schritte	Step	Intention der Systemumgebung	Reaktion des Systems

<table>
<tr><td></td><td><table>
<tr><td>1</td><td>Benutzer*in ruft den Bot zur Eingabe auf</td><td>System zeigt das Eingabefenster des Bots an und begrüßt Benutzer*in</td></tr>
<tr><td>2</td><td>Benutzer*in teilt dem Bot von welchem System sich dieser abmelden sollte. Der Systemname ist dabei obligatorisch und ist jener nachdem gesucht wird.</td><td>Bot löst den entsprechenden API Trigger in der Cloud Factory aus. Cloud Factory startet das Szenario im Desktop Agent und gibt eine Rückmeldung. Es werden alle Sessions beendet, die auf die Systembeschreibung zutreffen.</td></tr>
<tr><td>3</td><td>Benutzer*in liest entsprechende Rückmeldung und weist den Bot auf weitere Schritte hin.</td><td>Verarbeitung der weiteren Eingaben oder Beenden.</td></tr>
</table></td></tr>
<tr><td>Ausnahmefälle</td><td><table>
<tr><td>Step</td><td>Auslöser</td><td>Reaktion</td></tr>
<tr><td>2</td><td>Benutzer*in teilt dem Bot nicht alle erforderliche Parameter mit.</td><td>Der Bot stellt Rückfrage und weist auf fehlenden Parameter hin.</td></tr>
<tr><td>2</td><td>Der Bot kann die Abmeldung am System nicht durchführen.</td><td>Logging der möglichen Ursachen:
GUI Scripting auf System deaktiviert
Keine Verbindung vorhanden, welche auf den Systemname zutrifft.</td></tr>
</table></td></tr>
<tr><td>Nachbedingung</td><td>Benutzer*in ist auf dem System abgemeldet oder erhält Rückmeldung über etwaige Problematiken.</td></tr>
<tr><td>Kommentare/Anmerkungen</td><td></td></tr>
</table>

Transportauftrag freigeben und importieren

Name	Transportauftrag freigeben und importieren
Beschreibung	Benutzer*in möchte einen Transportauftrag auf einem System freigeben und auf einem anderen importieren.
Akteure	• Benutzer*in

<table>
<tr><td></td><td>• Bot</td></tr>
<tr><td>Voraussetzungen</td><td>• Der Desktop Agent befindet sich im „unattended“ Modus und ist aktiviert.
• Scripting Parameter im SAP- System sind gepflegt.
• Systeme befinden sich innerhalb einer Transportschiene.
• BAdI zur freigeben und importieren ist auf beiden Systemen implementiert.</td></tr>
<tr><td>Auslöser</td><td>User*in kommuniziert mit dem Bot und teilt dem Bot mit, dass dieser sich einen Transportauftrag auf einem System freigeben und auf einem anderen wieder importieren soll.</td></tr>
<tr><td>Essenzielle Schritte</td><td>
<table>
<tr><th>Step</th><th>Intention der Systemumgebung</th><th>Reaktion des Systems</th></tr>
<tr><td>1</td><td>Benutzer*in ruft den Bot zur Eingabe auf</td><td>System zeigt das Eingabefenster des Bots an und begrüßt den/die Benutzer*in</td></tr>
<tr><td>2</td><td>Benutzer*in teilt dem Bot, dass ein Transportauftrag freigegeben, importiert oder freigegeben und importiert werden sollte. Dabei sind die Auftragsnummer und die jeweiligen Systeme obligatorisch.</td><td>Bot löst den entsprechenden API Trigger in der Cloud Factory aus. Cloud Factory startet das Szenario im Desktop Agent und ruft den entsprechenden BAdI mit dem Transportauftrag auf.</td></tr>
<tr><td>3</td><td>Benutzer*in liest entsprechende Rückmeldung und weist den Bot auf weitere Schritte hin.</td><td>Verarbeitung der weiteren Eingaben oder Beenden.</td></tr>
</table>
</td></tr>
<tr><td>Ausnahmefälle</td><td>
<table>
<tr><th>Step</th><th>Auslöser</th><th>Reaktion</th></tr>
<tr><td>2</td><td>Benutzer*in teilt dem Bot nicht alle erforderliche Parameter mit.</td><td>Der Bot stellt Rückfrage und weist auf fehlenden Parameter hin.</td></tr>
<tr><td>2</td><td>Der Bot kann den Auftrag nicht importieren/freigeben.</td><td>Der BAdI gibt entsprechende Rückmeldungen basierend auf der Aktion. Diese wird dem*der User*in dargestellt. Basierend darauf kann der*die User*in weitere Schritte einleiten.</td></tr>
</table>
</td></tr>
<tr><td>Nachbedingung</td><td>Der Auftrag wird in das entsprechende System importiert bzw. vom entsprechenden System exportiert.</td></tr>
</table>

Kommentare/Anmerkungen	Der UseCase ist grundsätzlich so zu betrachten, dass der Benutzer bei Freigeben eines Auftrags diesen im Anschluss importiert. Es soll aber auch möglich sein nur eine der Aktionen auszuführen.

Passwort eines*r Benutzers*in ändern

Name	Passwort eines Benutzers ändern
Beschreibung	Benutzer*in möchte das Passwort von einem*r User*in auf einem bestimmten System ändern.
Akteure	• Benutzer*in • Bot
Voraussetzungen	• Der Desktop Agent befindet sich im „unattended“ Modus und ist aktiviert. • Scripting Parameter im SAP- System sind gepflegt. • BAdI zum ändern des Passworts ist implementiert bzw. importiert (Eigenentwicklung)
Auslöser	User*in kommuniziert mit dem Bot und teilt dem Bot mit, dass dieser das Passwort auf einem System zu einem bestimmten Usern ändern soll.
Essenzielle Schritte	

Step	Intention der Systemumgebung	Reaktion des Systems
1	Benutzer*in ruft den Bot zur Eingabe auf	System zeigt das Eingabefenster des Bots an und begrüßt den/die Benutzer*in
2	Benutzer*in teilt dem Bot, dass ein Passwort für ein*e User*in auf einem System geändert werden sollte. Dabei ist das System, der*die User*in obligatorische Parameter. Optional kann das neue Passwort als Parameter angegeben werden. Weiteres ist es auch möglich anzugeben, ob der*die User*in benachrichtigt werden soll,	Bot löst den entsprechenden API Trigger in der Cloud Factory aus. Cloud Factory startet das Szenario im Desktop Agent und ruft den entsprechenden BAdI auf.

		dass sein*ihr Passwort geändert wurde und wie das neue Initialkennwort lautet.	
	3	Benutzer*in liest entsprechende Rückmeldung und weist den Bot auf weitere Schritte hin.	Verarbeitung der weiteren Eingaben oder Beenden.
Ausnahmefälle	Step	Auslöser	Reaktion
	2	Benutzer*in teilt dem Bot nicht alle erforderliche Parameter mit.	Der Bot stellt Rückfrage und weist auf fehlenden Parameter hin.
	2	Der Bot kann das Passwort nicht ändern.	Der BAdI gibt entsprechende Rückmeldungen basierend auf der Aktion. Diese wird dem*der User*in dargestellt. Basierend darauf kann der*die User*in weitere Schritte einleiten.
Nachbedingung	Das Passwort zu dem*der User*in wurde geändert und dieser wird benachrichtigt, wenn dies vom Aufrufer gewünscht wurde.		
Kommentare/Anmerkungen	Benachrichtigung könnte jederzeit erfolgen und somit nicht als Parameter mitgegeben werden?		

Sperren/Entsperren eines*r Benutzers*in

Name	Sperren/Entsperren eines*r Benutzer*in
Beschreibung	Benutzer*in möchte das ein*e User*in auf einem bestimmten System gesperrt/entsperrt wird.
Akteure	• Benutzer*in • Bot
Voraussetzungen	• Der Desktop Agent befindet sich im „unattended“ Modus und ist aktiviert. • Scripting Parameter im SAP- System sind gepflegt.

<table>
<tr><td></td><td>

- BAdI zum Sperren/entsperren des*der Users*in ist implementiert bzw. importiert (Eigenentwicklung)

</td></tr>
<tr><td>Auslöser</td><td>User*in kommuniziert mit dem Bot und teilt dem Bot mit, dass dieser einen*e User*in auf einem bestimmten System sperren/entsperren soll.</td></tr>
<tr><td>Essenzielle Schritte</td><td>
<table>
<tr><th>Step</th><th>Intention der Systemumgebung</th><th>Reaktion des Systems</th></tr>
<tr><td>1</td><td>Benutzer*in ruft den Bot zur Eingabe auf</td><td>System zeigt das Eingabefenster des Bots an und begrüßt den/die Benutzer*in</td></tr>
<tr><td>2</td><td>Benutzer*in teilt dem Bot, dass ein*e User*in auf einem System gesperrt/entsperrt werden sollte. Dabei sind das System und der*die User*in obligatorische Parameter. Optional ist es möglich, dass der*die User*in benachrichtigt wird, dass sein*ihre User*in gesperrt/entsperrt ist.</td><td>Bot löst den entsprechenden API Trigger in der Cloud Factory aus. Cloud Factory startet das Szenario im Desktop Agent und ruft den entsprechenden BAdI auf.</td></tr>
<tr><td>3</td><td>Benutzer*in liest entsprechende Rückmeldung und weist den Bot auf weitere Schritte hin.</td><td>Verarbeitung der weiteren Eingaben oder Beenden.</td></tr>
</table>
</td></tr>
<tr><td>Ausnahmefälle</td><td>
<table>
<tr><th>Step</th><th>Auslöser</th><th>Reaktion</th></tr>
<tr><td>2</td><td>Benutzer*in teilt dem Bot nicht alle erforderliche Parameter mit.</td><td>Der Bot stellt Rückfrage und weist auf fehlenden Parameter hin.</td></tr>
<tr><td>2</td><td>Der Bot kann den*die User*in nicht entsperren/sperren.</td><td>Der BAdI gibt entsprechende Rückmeldungen basierend auf der Aktion. Diese wird dem*der User*in dargestellt. Basierend darauf kann der*die User*in weitere Schritte einleiten.</td></tr>
</table>
</td></tr>
<tr><td>Nachbedingung</td><td>Benutzer*in kommuniziert mit dem Bot und teilt dem Bot mit, dass dieser einen User auf einem bestimmten System sperren/entsperren soll.</td></tr>
</table>

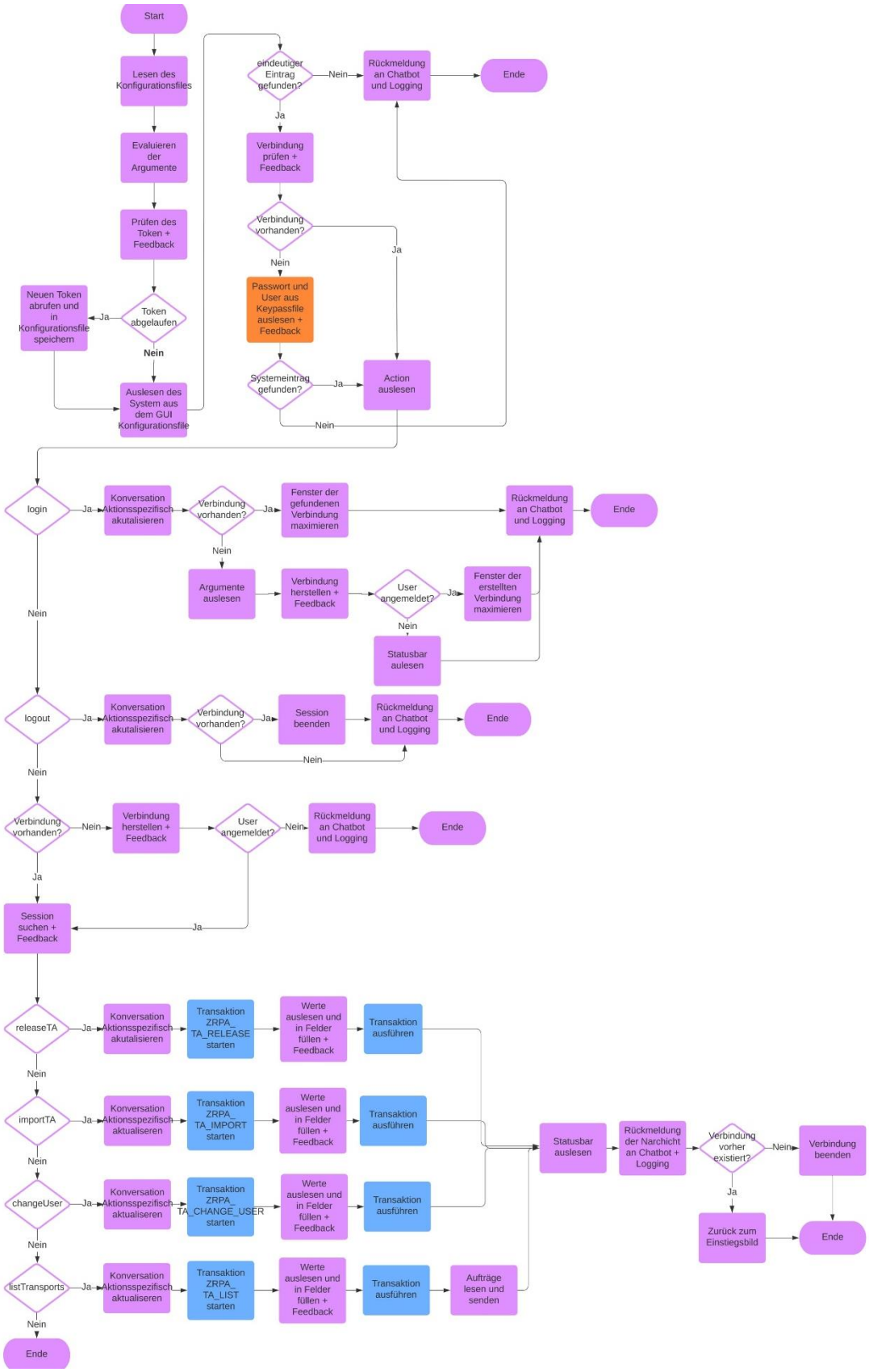
Kommentare/Anmerkungen	Benachrichtigung könnte nur bei Entsperren erfolgen?

Transportliste anzeigen

Name	Transportliste anzeigen
Beschreibung	Benutzer*in möchte sich die Transportliste anzeigen lassen.
Akteure	• Benutzer*in • Bot
Voraussetzungen	• Der Desktop Agent befindet sich im „unattended“ Modus und ist aktiviert. • Scripting Parameter im SAP- System sind gepflegt. • BAdI zur Anzeige der Transportaufträge des*der Users*in ist implementiert bzw. importiert (Eigenentwicklung)
Auslöser	User*in kommuniziert mit dem Bot und teilt dem Bot mit, dass eine Liste der Transportaufträge angezeigt werden soll.
Essenzielle Schritte	

Step	Intention der Systemumgebung	Reaktion des Systems
1	Benutzer*in ruft den Bot zur Eingabe auf	System zeigt das Eingabefenster des Bots an und begrüßt den/die Benutzer*in
2	Benutzer*in teilt dem Bot, dass dieser die Transportaufträge von einem bestimmten System anzeigen soll.	Bot erfasst die System ID und fragt, ob noch weitere Parameter hinzugefügt werden sollten. Es besteht die Möglichkeit auf Benutzer, freigegebene uder/oder änderbare Transporte zu filtern.
3	Benutzer*in teilt dem Bot weitere Parameter zum Hinzufügen oder entfernen mit.	Bot Verarbeitet die Daten und teilt nach jeder Änderung die aktuellen Parameter mit und stellt Buttons mit „Aufgabe starten“ und „Parameter bearbeiten zur Verfügung“

<table>
<tr><td></td><td><table>
<tr><td>4</td><td>Benutzer*in startet die Aufgabe</td><td>Chatbot setzt einen API Call an die RPA ab. RPA startet die lokale Datei und die Transportaufträge werden im Chatfenster als Liste dargestellt.</td></tr>
</table></td></tr>
<tr><td>Ausnahmefälle</td><td><table>
<tr><td>Step</td><td>Auslöser</td><td>Reaktion</td></tr>
<tr><td>2</td><td>Benutzer*in teilt dem Bot nicht alle erforderliche Parameter mit.</td><td>Der Bot stellt Rückfrage und weist auf fehlenden Parameter hin.</td></tr>
<tr><td>2</td><td>Der Bot kann Aufträge nicht auslesen.</td><td>Der BAdI gibt entsprechende Rückmeldungen basierend auf der Aktion. Diese wird dem*der User*in dargestellt. Basierend darauf kann der*die User*in weitere Schritte einleiten.</td></tr>
<tr><td>4</td><td>Benutzer*in möchte Parameter nochmals ändern</td><td>Bot stellt aktuelle Parameter dar und fragt nach weiteren Änderungen.</td></tr>
</table></td></tr>
<tr><td>Nachbedingung</td><td>Es werden alle Transportaufträge im Chatfenster aufgelistet</td></tr>
<tr><td>Kommentare/Anmerkungen</td><td></td></tr>
</table>


Start
Lesen des Konfigurationsfiles
Evaluieren der Argumente
Prüfen des Token + Feedback
Token abgelaufen
Ja
Neuen Token abrufen und in Konfigurationsfile speichern
Nein
Auslesen des System aus dem GUI Konfigurationsfile
eindeutiger Eintrag gefunden?
Nein
Rückmeldung an Chatbot und Logging
Ende
Ja
Verbindung prüfen + Feedback
Verbindung vorhanden?
Ja
Nein
Passwort und User aus Keypassfile auslesen + Feedback
Systemeintrag gefunden?
Ja
Action auslesen
Nein
login
Ja
Konversation Aktionsspezifisch akutalisieren
Verbindung vorhanden?
Ja
Fenster der gefundenen Verbindung maximieren
Rückmeldung an Chatbot und Logging
Ende
Nein
Argumente auslesen
Verbindung herstellen + Feedback
User angemeldet?
Ja
Fenster der erstellten Verbindung maximieren
Nein
Statusbar aulesen
Nein
logout
Ja
Konversation Aktionsspezifisch akutalisieren
Verbindung vorhanden?
Ja
Session beenden
Rückmeldung an Chatbot und Logging
Ende
Nein
Nein
Verbindung vorhanden?
Nein
Verbindung herstellen + Feedback
User angemeldet?
Nein
Rückmeldung an Chatbot und Logging
Ende
Ja
Session suchen + Feedback
Ja
releaseTA
Ja
Konversation Aktionsspezifisch akutalisieren
Transaktion ZRPA_ TA_RELEASE starten
Werte auslesen und in Felder füllen + Feedback
Transaktion ausführen
Nein
importTA
Ja
Konversation Aktionsspezifisch aktualisieren
Transaktion ZRPA_ TA_IMPORT starten
Werte auslesen und in Felder füllen + Feedback
Transaktion ausführen
Nein
changeUser
Ja
Konversation Aktionsspezifisch aktualisieren
Transaktion ZRPA_ TA_CHANGE_USER starten
Werte auslesen und in Felder füllen + Feedback
Transaktion ausführen
Nein
listTransports
Ja
Konversation Aktionsspezifisch aktualisieren
Transaktion ZRPA_ TA_LIST starten
Werte auslesen und in Felder füllen + Feedback
Transaktion ausführen
Aufträge lesen und senden
Nein
Ende
Statusbar auslesen
Rückmeldung der Narchicht an Chatbot + Logging
Verbindung vorher existiert?
Nein
Verbindung beenden
Ja
Zurück zum Einstiegsbild
Ende

Eidesstattliche Erklärung

Ich erkläre hiermit an Eides statt, dass ich vorliegende Masterarbeit selbstständig und ohne Benutzung anderer als der angegebenen Hilfsmittel angefertigt habe. Die aus fremden Quellen direkt oder indirekt übernommenen Stellen sind als solche kenntlich gemacht.

Die Arbeit wurde bisher weder in gleicher noch in ähnlicher Form einer anderen Prüfungsbehörde vorgelegt und auch noch nicht veröffentlicht.

Dornbirn, am 08.07.2022 Dominik Meyer