2020

Automation of Customer Initiated Back Office Processes: A Design Science Research Approach to link Robotic Process Automation and Chatbots

Christian Daase
Otto-von-Guericke University, christian.daase@ovgu.de

Daniel Staegemann
Otto-von-Guericke University Magdeburg, daniel.staegemann@ovgu.de

Matthias Volk
Otto von Guericke Universitat Magdeburg, matthias.volk@ovgu.de

Abdulrahman Nahhas
Otto von Guericke University Magdeburg, abdulrahman.nahhas@ovgu.de

Klaus Turowski
ITI, University of Magdeburg, Magdeburg, Germany., klaus.turowski@ovgu.de

Automation of Customer Initiated Back Office Processes: A Design Science Research Approach to link Robotic Process Automation and Chatbots

Completed research paper

Christian Daase
Magdeburg Research and Competence Cluster Very Large Business Applications
Otto-von-Guericke University Magdeburg
Magdeburg, Germany
Email: christian.daase@ovgu.de

Daniel Staegemann
Magdeburg Research and Competence Cluster Very Large Business Applications
Otto-von-Guericke University Magdeburg
Magdeburg, Germany
Email: daniel.staegemann@ovgu.de

Matthias Volk
Magdeburg Research and Competence Cluster Very Large Business Applications
Otto-von-Guericke University Magdeburg
Magdeburg, Germany
Email: matthias.volk@ovgu.de

Abdulrahman Nahhas
Magdeburg Research and Competence Cluster Very Large Business Applications
Otto-von-Guericke University Magdeburg
Magdeburg, Germany
Email: abdulrahman.nahhas@ovgu.de

Klaus Turowski
Magdeburg Research and Competence Cluster Very Large Business Applications
Otto-von-Guericke University Magdeburg
Magdeburg, Germany
Email: klaus.turowski@ovgu.de

Abstract

While the emerging technology of robotic process automation is primarily suitable for back office processes, companies use traditional chatbots to support customer interaction in the front office. However, customer requests that require more than written information usually demand an employee to execute an internal process. This paper summarizes the results of a technical design process for a combination of both technologies. After an introduction on both topics, the findings of a literature review regarding existing approaches are outlined. The development of the IT artefact is then carried out according to the design science research methodology. In particular, the research focuses on the constitution of a design theory in consideration of criteria that are found to be important for a purposeful appearance to the external user. After a proof of concept by testing the developed artefact and a summary of the results, an outlook on possible future developments is provided.

Keywords Automated Customer Request Processing, Customer Experience, Robotic Process Automation, Chatbots, Design Science Research

1 Introduction

In information technology, operational systems such as enterprise resource planning and customer relationship management require certain tasks like cross-system data extraction to complete an entire process (Aguirre and Rodriguez 2017; Lacity and Willcocks 2015). Such repetitive activities are rule-based, which is why delegating highly qualified and expensive staff for these tasks is a waste of potential. Often companies are interested in qualified employees spending more time on work that cannot be automated. In other words, highly structured routine tasks should be handled by robots so that the personnel have more time for value-adding tasks (Aguirre and Rodriguez 2017; Anagnoste 2017; Siderska 2020). Software robots can replace human workers in various areas while creating new roles, which did not exist previously (Kirkpatrick and Wheelock 2017). They have to be maintained, upgraded and a human must intervene if an unexpected event cannot be processed. Automation therefore not only influences the digital infrastructure, but also shifts the focus to new areas of work. The (further) development of two technologies, presented hereafter in a generalized overview, has played an outstanding role in this context in recent years.

The first one, robotic process automation (RPA) is a technology, which was developed to automate rule-based software tasks by using software robots (Lacity and Willcocks 2015), which work like a human on the graphical user interface of the front end (Asatiani and Penttinen 2016; van der Aalst et al. 2018) and use tools such as a virtual keyboard or mouse (Scheer 2019). The benefits of software robots compared to a human worker are obvious: apart from maintenance outages, the robots can be used around the clock all over the year while working accurately and efficiently (Kirkpatrick and Wheelock 2017; Siderska 2020). The only limitations of their performance may be their configuration, the processing capacity of the hardware and the response times of the applications involved (Aguirre and Rodriguez 2017). Companies do not have to hold back on such implementations because most simple automations do not require specific programming skills (Lacity and Willcocks 2015) apart from a general knowledge of the involved computer system. Thus, even the everyday worker can use RPA to have the execution of his computer based routine tasks imitated by a robot (Kirkpatrick and Wheelock 2017). Up to now, RPA has proven to be very useful for back office processes (Anagnoste 2017). This is partly because internal workflows are known to the company and can therefore be rule-based standardized. However, design issues with a software robot may arise if the process involves uncontrollable external actors like customers (Rutschi and Dibbern 2019), who especially in customer-oriented businesses shape the front office workflows.

In order to close the gap between front office and back office, an attractive interface for the external users is beneficial, if not essential. It also requires a functionality to convert the unstructured input, such as requests or textual data, into standardized formats to trigger the right internal processes. One possible solution are traditional chatbots. While RPA serves to convert structured input into structured output, chatbots normally convert unstructured input into unstructured output (Rutschi and Dibbern 2019). Chatbots can be seen as a further development of conventional user interfaces (Berger et al. 2019), but in contrast to applications with menus, a minimal version of a chatbot only consists of three elements: a text input field, a chat log and a button to send a new message. Although the development of chatbots took its beginning in the mid-1960s with Joseph Weizenbaum's chatbot called ELIZA (Weizenbaum 1966), the technical requirements are still very similar today. Triggering processes through chatbots might be problematic for the fact that they are usually built to return text messages, which are identified as appropriate answers following a predefined path. However, it is possible to store parameters while following the context of a dialogue and to query them using the API.

RPA and chatbots provide comparable advantages in their respective areas of application, ranging from cost reduction and error minimization to permanent reachability (Rutschi and Dibbern 2019). In addition, the combination of both technologies offers further improvements in staff scheduling. The chatbot takes on the role of a service employee by being able to receive a customer request and the data required to execute a process in natural language (Dale 2016). If this data is of such complexity that it would normally have to be processed by a human operator, the connected software robot (i.e. the RPA component) in the back end can provide these processes around the clock and also spare a service employee during the day to perform other tasks, comparable to other AI-based software solutions (Maedche et al. 2019). Application areas could be, for example, account management at a bank (Iyer et al. 2019) or the administration of customer orders (Heo and Lee 2018). The combination of RPA and chatbots presented in this paper aims to offer benefits especially to smaller companies with very limited human and financial resources, but several computer-based processes with clearly defined routines. Therefore, the concept could potentially serve as a cost efficient automation alternative for organizations that are neither capable of running their own IT department exclusively dedicated to providing complex business automation solutions, nor able to integrate highly tailored and expensive third-party technologies. This goal can be achieved by taking advantage of two essential properties of the basic components involved. Firstly, RPA can be used to model various processes with relatively low effort (Lacity and Willcocks 2015), giving organizations a flexibility that might constitute a competitive advantage (Staegemann et al. 2020). This modelling can for example be realized by recording the manual execution by an employee or by reconstructing the process in a flowchart within an RPA editor. Secondly, all resources used in this study for the development of a prototype are available free of charge, so that the presented key concept could be adopted quite easily. The only exception to the statement of simple integration is the server technology that connects the individual components, and which would have to be extended by a qualified programmer.

2 Research Methodology and Outline

Prior to developing a software artefact, a theoretical blueprint must be developed that meets scientific standards. This forms the basis for an initial evaluation of the technical feasibility and the interplay of the components. Logical justifications for choice and assembly of the elements must be possible based on this preliminary work. In order to achieve this objective, the scheme for a design theory presented by Jones and Gregor (2007) is adopted. Besides the technical issues, which are crucial for the implementing company, it is equally important to consider which aspects make this form of communication attractive for the external user. Since every business process has its own specifications regarding workflow and requirements (Scheer 1999) and is strongly linked to the business model of the respective company, it is not possible to assume a universal list of criteria for all kinds of processes. For this reason, existing case studies, solutions and insights from the artefact will be used to establish abstract criteria that can improve the user experience with chatbots in combination with further processing of the input (via RPA). Essentially, the main goal of the presented paper is to answer the following research questions:

RQ1: *Which components need to be included in a design theory of the combination of RPA and chatbots in order to use the common benefits also for user initiated back office processes?*

RQ2: *Which criteria could be relevant to seize the advantages of both technologies to provide a higher level of user-experience?*

For the purpose of answering the research questions and address the aforementioned gap, the design science research methodology (DSRM) is followed, leading to the development of a purposeful IT artefact (Hevner et al. 2004; Peffers et al. 2007). The successful commissioning of this prototype then serves as proof that business processes initiated by external users via a chat interface can be automated within the internal digital infrastructure without requiring a service employee or incurring significant costs. The DSRM considers the emergent nature of this kind of artefacts (Sonnenberg and Vom Brocke 2012) and serves in an adapted form as a basis for this research. The first step is to define the objectives of the research, in this case the research questions. The corresponding first evaluation phase consists of the conduct of a literature search applying the recommendations set up by Cooper (1988) on existing approaches about the topics of chatbots and process automation in combination to examine the need for research and its relevance. However, due to space limitations, the literature review is not intended to provide a comprehensive overview of a wide range of automation approaches, but rather to present a small selection of comparable ideas to highlight the improvements that the concept presented here could bring. This first evaluation phase is followed by a design for the construction (Sonnenberg and Vom Brocke 2012), while the choice of components is permanently justified and evaluated. Therefore, the DSR activity of designing the artefact and the corresponding evaluation phase are combined as an adaptation to the common DSRM. This integration should facilitate the traceability of the different design decisions by avoiding repeated references to the text passages where a component is initially introduced. The findings are summarized as a design theory with six core components according to Jones and Gregor (2007). Finally, an IT artefact (Hevner et al. 2004) is developed as a proof of concept. During the subsequent third evaluation phase, this artefact is being tested in an artificial environment to see if it works as planned with explicit input values (Jones and Gregor 2007). This demonstration that the specific selection and assembly of components results in a functioning artefact is also referred to as design knowledge, which is manifested in design theory (Vom Brocke and Maedche 2019). This procedure corresponds to the design-evaluate-construct-evaluate pattern (Sonnenberg and Vom Brocke 2012), with the difference that the design decisions are justified and evaluated directly during the conception phase. A fourth evaluation phase in a natural setting is planned as soon as the artefact has been adapted to a specific real-world task.

This paper follows a straightforward structure. The foundation and methodology are explained in these first and second sections of the paper. The third section contains a structured literature review that, in addition to the above-mentioned purposes, serves as a basis for several design decisions, which are later summarized as a design theory in an attempt to answer RQ1. In addition, conclusions on possible criteria regarding the user experience are revealed. The individual components of the concept and their composition are discussed in the fourth section. Furthermore, important insights for RQ2 are gained step by step, which are incorporated into the practical development. The fifth section summarizes the results of the study, including the proof of concept. Also, this last section closes with a conclusion and an outlook on possible future developments.

3 Literature Review

For the literature review, databases are used, which primarily provide articles on computer science or economics. Current topics in information technology are characterized by the fact that their development is progressing rapidly, and numerous new publications are constantly appearing (Vom Brocke et al. 2009). This makes it difficult to provide an overview on the latest state of the art, but it also offers a wide range of scientific contributions of varying scope. By following the approach according to Cooper (1988), the conducted review attempts to conform six basic characteristics. The focus (1) of the research is on studies on applications, with the goal (2) of a functional integration. The research is conceptually organized (3), as existing concepts for the abstract idea of this paper are to be used in the development and the articles are regarded from a neutral perspective (4). Regarding the audience (5) of individual articles, a high relevance for practitioners is a criterion for the search. The coverage (6) is representative, as neither all existing articles are to be reviewed, nor is there a central selection of generally accepted literature on the subject. Furthermore, the main goals of the examination are the discovery of important aspects for the design theory and the identification of alternative strategies for automated processing of requests from external users. The following four databases are used for the initial search: ScienceDirect, AIS Electronic Library, ACM Digital Library and IEEE Xplore Digital Library. In addition, Google Scholar is used, which, however, does not allow any distinction between the abstract and the full text during the search. This addition is necessary as the four databases mentioned before do not provide material according to the following strict criteria at the time of the work on this paper. The reason why the inclusion of these databases is nevertheless mentioned here is that the revealed lack of available material underlines the novelty of the exact approach of the publication at hand. First, it is determined that the terms *robotic process automation* and *chatbot* must be present together in the title, keywords or abstract fields, since both terms alone are not sufficiently relevant for the investigation. ScienceDirect only allows a single input for these fields without linking, so *robotic process automation chatbot* is used. For Google Scholar, the search phrase *robotic process automation +chatbot* is used, including the missing space after the plus sign. The remaining databases allow AND links of several terms and the specific search in individual sections of the articles. Also, the time frame of all search queries is not limited. The only exception is Google Scholar with a search starting in 2016, in order to allow a limitation of the results despite the missing restriction to the abstract of a text. This strict specification of the search is intended to give other researchers the opportunity to evaluate whether the search conducted here sufficiently covers their own research (Vom Brocke et al. 2009).

Except for a book chapter on the exemplary use of artificial intelligence in the medical field found in ScienceDirect, Google Scholar is the only source that provides results complying with these constraints. Only articles published in English are considered as relevant. In addition, the publications are not allowed to address the topics RPA and chatbots exclusively separately or as examples of artificial intelligence. Redundancies that are displayed due to publications on more than one portal are also not considered. The results have to originate from journals or conference proceedings. According to these criteria, summarized in Table 1, the three most relevant found articles are briefly discussed below.

Inclusion criteria	Exclusion criteria
English language	Only separated treatment of RPA/chatbots
RPA and chatbots in the same context	Focus on RPA as general AI technology
Conference or journal paper	Duplicates

Table 1. Inclusion/exclusion criteria for the literature review

The contribution by Gajra et al. (2020) proposes a combination of a chatbot based on the platform Dialogflow and software robots developed with UiPath Studio. Although it is stated that RPA can be used for many processes in data management, the paper focuses on only a few data fetching tasks. While the components used are partly identical to those presented in the present paper, the authors incorporated

UiPath's so-called orchestrator to enable on-demand use of several software robots. Since the orchestrator is chargeable, the free alternative used in this study consists of a permanently running robot that is used to control the process distribution. Although the published version of the research paper contains only an unreadable blurred graphic of the workflow, it seems that either the software robot sends the process result via e-mail to the user or the chatbot returns a message with data that was retrieved from a database using the API. In the latter case, no RPA robot is involved. A major drawback of this approach compared to the one pursued in this study is that there appears to be no way to pause a conversation until a process result is available, which could then be sent via the chat interface. In contrast, in the present work, an approach is proposed where all communication takes place via the server, including the chat interface and the storage of the chat protocol. Dialogflow, which is also used for this project, is solely responsible for the interpretation of user messages, while the implemented logic of the server decides on the further procedure, for example initiating a process or simply sending a response text. Conversely, the textual process results sent by the software robot are also managed by the server logic. This architecture is beneficial to the user experience, since it makes it possible to get an immediate response as to whether a process has been initiated or the user should wait until the robot sends a message depending on the process result. In addition, the implementation of chat functionality except for message interpretation offers optimization possibilities such as synchronization of multiple browser tabs and enabling multi-level processes, meaning that the user could initiate a second process depending on the result of a first process. The architecture of Gajra et al. does not allow the software robot to intervene in the course of the conversation.

The case study by Heo and Lee (2018) examines so-called cardbots as automation technology at the Korean company Naver. According to the authors, a cardbot is a kind of chatbot in which the user receives a predefined selection of requests during the conversation (cards). Naver initially developed a platform for communication between vendors and customers, which proved to be as time-consuming as the personal sales process in a store. Naver only wanted to use a chatbot to provide access to an FAQ, but then implemented a chatbot to automate the whole sales process. However, according to the study, only 7 percent of user input led to proper answers from the chatbot and only 0.1 percent of conversations led to a sale. These numbers are explained by the fact that customers did not know what questions they could ask. This problem reveals an important criterion for answering RQ2, which is the need to guide users to appropriate inputs. Based on these results, Naver implemented the aforementioned cardbot and was therefore able to increase the successful responses to 29 percent and the sales conversations to 12.4 percent. Although the idea behind this case study is quite similar to the one in this paper, using a cardbot instead of a natural language chatbot not only raises space limitations regarding the chat interface, but also problems in extending the capabilities of the system. With a cardbot, every possible next conversation step would have to be visualized for the user. In addition, adding new functionalities would require a re-planning of the structure of these predefined conversation paths. In contrast, the present paper proposes a system that incorporates a traditional chatbot to allow developers to easily add new functionality at any stage of a conversation while still ensuring a meaningful conversational guidance.

The problem of a missing possibility to access a software robot from outside is also recognized by Rizk et al. (2020) in their study. To reduce this barrier, the researchers propose a framework for adapting automated processes by allowing external users to interact with the robot via natural language. While this idea is similar to the one developed in the present paper, the actual approach differs because each digital agent has its own natural language understanding capabilities. Thus, the offered functionalities of the system could easily be extended or reduced, since only the respective software robots would have to be integrated or decoupled. The main disadvantages of this approach are presented by the authors in their conclusion. First, each user message must be forwarded to each agent to determine which one contains the desired process or reacts appropriately, which leads to an increase in coding effort and processing time. Second, managing many different agents increases the complexity of the central digital assistant, also referred to as an orchestrator by the authors. In addition, adding purely informative conversation capabilities without an offered process execution would also require a new agent to be connected to the orchestrator. Since the use cases established by Rizk et al. rely more on programmed functions and external microservices, the exact integration and potential benefits of RPA key concepts (i.e. the use of different applications) remain unclear. In contrast, the present paper follows the approach of developing conversation capabilities only once and connecting all software robots with the same interpreter. This reduces the complexity of building a complete chat robot, while new conversation paths can be implemented with minimal effort.

Additionally to the results of the structured review, another publication, which covers a single automation using a chatbot without any connection to a software robot was found. Researchers at the TU Graz (Berger et al. 2019) combined a chatbot with the university's search function via a NodeJS back end. NodeJS is a run-time environment that was first introduced in 2009 and which allows server-side applications to be developed based on JavaScript. The main advantage, according to the authors, is that this is an enhanced user interaction, as a user can now interact directly with the information system without having to perform multiple steps through a normal user interface. The developers used Dialogflow to analyse the user input, which is a platform owned by Google that provides tools for processing natural language input and machine learning. One of the reasons given for this decision is that Dialogflow has advanced cognitive capabilities, allows the transfer of data into the open-standard file format JSON and is available at no costs. The NodeJS Server is a central processing unit that connects the three components: the graphical interface of the chatbot, Dialogflow as the interpreter and the search system. Because the study found that only about 50 percent of the users were fairly satisfied, the authors conclude that a significant improvement of the system is necessary before the old system can be replaced.

The components of this last study are of similar importance for the development of the IT artefact of this paper, even if the developers did not use RPA. The reasons given by the authors, such as cost reduction, increased efficiency and simplification, were also taken into account in the development of the following design.

4 Design

The development of the IT artefact does not include a complete re-engineering of a chatbot, as several frameworks already exist. Before the decision for an interpreter is made, the criteria with which the suitability for this project is assessed must be defined. These criteria include, among others, text comprehension and data collection, which are explained in further detail in the second subsection. To showcase the generality of the proposed artefact, the prototype is tested with two independent interpreters. As stated later, Dialogflow and the platform chatbot.com are chosen.

4.1 Basic Components

The main component, which forms the connection to all other components, is a local server, as in the TU Graz study (Berger et al. 2019). The purpose of it is to receive user requests in the form of text from the chat interface and to forward them to the respective interpreter. From the user's message, any data and intentions are then extracted, which again are returned and stored on the server. *Intention* in this case means a data package in JSON, which contains all required parameters for a certain process. As described in the case study (Berger et al. 2019), NodeJS is also used here, as this offers two important benefits. First, it provides fundamental server functionality and easy-to-use modules for API calls, for example in conjunction with Dialogflow. Secondly, a dynamic client-side scripting language is also required for the development of the chat interface. In order to focus on the RQs instead of explaining the peculiarities of another scripting language (like PHP), JavaScript can be used for both components. With these basic elements mentioned so far, the functionality of a normal chatbot can already be replicated.

This structure is extended with an RPA component, or in other words with a software robot together with the process-dependent external systems. As soon as a user follows a context defined in the interpreter up to the transmission of a complete intention, it is stored on the server marked with a keyword. The software robot periodically sends requests to the server to receive such intentions. It then processes a user's request using the external systems, for example databases or email programs and sends a response text based on the process result. In this way, risk-free processes without the transmission of sensitive data as well as risk-carrying tasks that require sensitive data or authentication are made possible. The content of this study only includes functional tests with a simple security architecture because the focus lies on the concept. Since the interpreter cannot obtain information from the enterprise systems, the software robot, for which this is the case, must be able to communicate with the interpreter via the server. All these components worked on the same computer during the study so that all steps could be visibly traced. The development of the software robot takes place in UiPath Studio as it is an intuitive tool with a simple drag-and-drop technique for the creation of workflows as sequences or flowcharts. Figure 1 visualizes the program flow according to this design concept.

As indicated by the caption, the continuous arrows mark the obligatory procedure, regardless of whether the user submits an identifiable intention. The dotted arrows, on the other hand, mark the paths that are pursued exclusively after the detection of at least one new intention. The software robot queries new process requests, since no event-based system is used and the robot is supposed to regularly send requests to the server from its side. For this reason, the arrow pointing from the software robot in the direction of the server is not dotted. Since the queries and the execution of the processes by the software robot run parallel to the server, there is no fixed order which is why the arrows are not numbered.

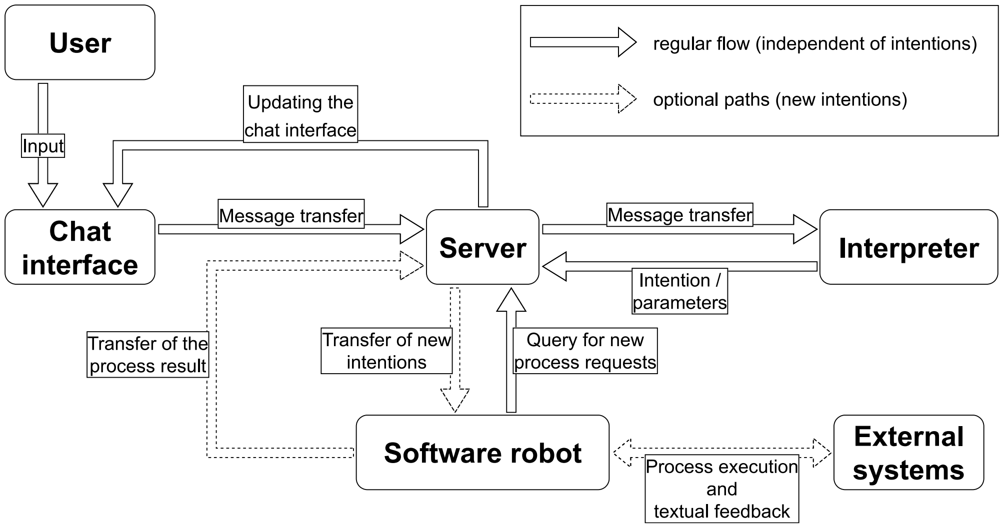

Figure 1. Scheme of the design concept

4.2 Components in Detail

The first component that appears is the chat interface. It has both, an input and an output function and consists of three elements: a text field for entering messages, a chat log for an overview of the previous conversation, and a control element that triggers the transmission of a message. While the purpose of the first two elements for the user is their visual presence itself, for the button the functionality after activation is decisive. As mentioned before, for the prototypical implementation, JavaScript is used as the scripting language for the dynamic interaction. The analysis of existing chat systems of social networks for the communication between real persons also showed that the synchronization of several chat windows is common, which means if a user has opened several tabs, that in each instance all messages of the conversation are displayed. For the purpose of improving the user experience, this idea was adopted into the human-machine communication. This ensures that if there is an undefined number of open windows or when an open window is closed, the entire chat history is still available in the remaining windows. Synchronization is thus included as a criterion for an improved user experience with regard to RQ2.

The server transmits inputs and outputs, stores or forwards intentions and manages the user sessions. For each connected subsystem, it must contain the necessary functionality. It is crucial for the development that there should be no way to terminate the server from outside by an unprocessable request. Another necessity for the server is to develop a data structure for the individual sessions. Each session is understood as an object, with a session ID, a single permanent connection to the client and a collection of all chat messages. In addition, the session ID must be encrypted, so that a user cannot manipulate the ID from his side in order to access the session of another user. In long-term real-world scenarios, sessions have to be provided with a time stamp that renews with each interaction to automatically close inactive sessions. To be able to integrate the software robot, the time for the processing must be considered. Since neither the required time of the RPA component nor the speed of the user when writing new messages can be determined, the session has to be paused until the end of the process to prevent a new intention from being transmitted in the meantime or the context of the conversation being led away from the original intention. The condition for this is that the software robot itself would not be disabled at any time, otherwise all affected sessions would be blocked permanently. For test purposes, it is also helpful to be able to immediately be sure of the transmission of a message without observing the consoles. Thus, the system is designed so that a message is first sent to the server, saved in the chat log and then sent back to the client's chat via an event stream. So, the displaying of an own message guarantees that its arrival on the server.

When analysing user input, the understanding of natural language is one of many quality characteristics to consider when choosing an interpreter. A further requirement is the ability to acquire data to enable process automation at all. Moreover, it is helpful if the comprehension of language can be extended at any time without affecting the existing structures of the further processing. While the last two criteria are mainly interesting for developers, the first criterion has a direct influence on the user experience and is therefore also meant to be a criterion for RQ2. Modern platforms, like Dialogflow, eliminate the need for developers to program algorithms in order to analyse the transmitted strings. However, the principle is comparable to that of ELIZA, where an input is searched for key phrases (Weizenbaum 1966). The exact keywords are usually unknown to the user. This problem can be solved with synonyms and abstract patterns, just as Weizenbaum summarized the terms *how*, *what* and *when* as a common class of words. Dialogflow allows collections of words as so-called entities, where equivalent words refer internally to a uniform term. Thus, alternative formulations are possible for the user and it is guaranteed for the further process that a certain expression is existent if an intention is pursued. However, synonyms cannot handle another problem, which is typos. Since the possible combinations of characters for a word with typos are virtually endless, advanced interpreters like Dialogflow rate an input with a score against all possible conversation paths respectively their key phrases. This score is based on the similarity of the input to each key phrase and ranges either from 0 to 1 or from 0 to 100, where 1 respectively 100 (percent) means that both terms are identical. The calculation is performed internally for each interpreter according to its own rules and with a possible connection to more advanced cognitive systems. Furthermore, the developer can define a threshold above which an input is regarded as understood for a particular path. Then the path with the highest score is followed. It can also happen that no path at all reaches a score higher than the threshold. In this case, the meaningful continuation of the conversation is another criterion for an optimized user experience (RQ2). Weizenbaum recognized the problem and used a stored statement of the user which was transformed into an interested question from ELIZA. Modern interpreters use various approaches to solve this problem. Usually there is a set of predefined answers for all unresolved messages, so-called fallback interactions, which consist of different formulations of the sentence “Can you please repeat the question in other words?". A significant problem, however, is that an interpreter cannot recognize whether the general concern can be answered at all, while a person can understand the meaning of a question but cannot answer it with his knowledge. This would lead to an infinite loop of fallbacks for persistent users. Alternatively, a system such as Weizenbaum's could be used to direct the conversation to the previous context after a few failed attempts. However, this is not necessary with the functional tests in the third evaluation phase, since in the artificial test scenario only the correct functionality for a reasonable usage is verified, taking into account that there is only a bot on the other side of the interaction.

The second of the criteria for the interpreter, that of data acquisition, consists of two parts: the transmission of data from the user to the interpreter and the transmission from the interpreter to the server. Since the data is transferred in JSON format and for reasons of abstraction several interpreters are to be installed, a convention for naming keywords is necessary so that the server could correctly process intentions from all sources. For example, all interpreters should use the parameter labelled *intent* as the trigger for a process. Despite the efforts for abstraction of the server, a specialized program section must be developed at this point for each interpreter, since this standardized section can be nested differently in the response of the API call for different interpreters. In order to exploit the advantages of chatbots over applications, the necessary amount of data per process has to be very limited. The step-by-step transfer of data is time-consuming and the developer would have to consider backward steps, otherwise errors could arise that would not occur with a form. A small amount of data is therefore another criterion for the development of suitable processes (RQ2). Following the analysis of the data package by the interpreter, the server assembles a uniform data package for the software robot, which now combines the intention and the associated data completely independently from the interpreter.

The third criterion for the selection of an interpreter is the expandability of speech comprehension. While with ELIZA the entire script has to be edited and reloaded (Weizenbaum 1966), today's interpreters allow individual phrases to be added via a graphical user interface. The training of an interpreter can be distinguished into two variants: sentences that the developer implements with his own knowledge and sentences that the user enters without knowing their effect. On the platforms of this project, the latter are stored in an archive and can later be added manually by the developer. In addition, sentences with frequent typos (such as *their/there/they're*) can consciously be included in the training to expand the range of accepted sentences. However, since this can bypass the threshold, this option is not used.

The software robot contains a separate workflow for each possible process. The robot regularly queries all current intentions from the server and processes them before submitting a new request to the server. From the JSON object from the server, the parameters are extracted which must be identically named in the interpreter and in the workflow. To reduce the workload of the systems due to too many unnecessary API calls, a time span is defined until a new request for intentions can be made by the robot. In addition, redundancy in data transfer is reduced by setting up a memory for the software robot, which could store current process parameters and process results that have already been processed. Since the quick reusability of results is a factor for the speed of the system, it is also a criterion for answering RQ2. Nevertheless, the memory requirement must be considered and how many results should be stored before a new process execution would be more reasonable. While the external systems are always process-dependent, only a web browser and a table calculation are used in this project, on whose graphical interfaces RPA can exploit its strengths.

4.3 Research Questions and Proof of Concept

The six core components of a design theory according to Jones and Gregor (2007) are summarized in Figure 2 from the findings of the study. From the analysis of various case studies and preliminary work by other researchers, it could be determined both the expected benefits of a combination of chatbots and RPA, as well as the resources required for this in an abstract way. The working principle that a possible prototype could follow in the execution of processes and the specific technical background were derived from logical conclusions and practical development. Finally, as an indispensable component of a design theory, the testable propositions regarding a purposeful utilization are listed, as well as sources of knowledge justifying the decisions in the research process and the results of this study.

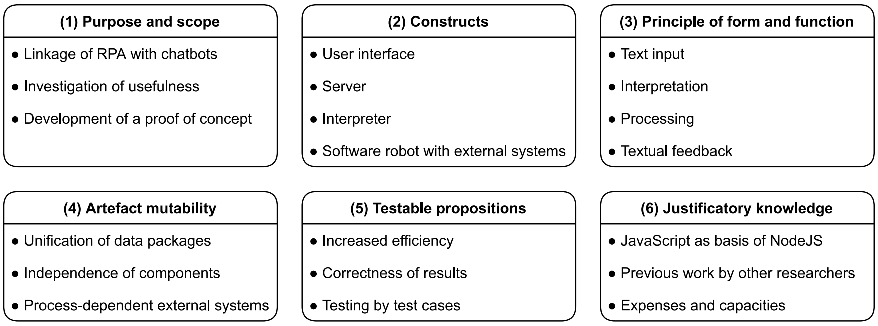

Figure 2. Summarized core components of the design theory

The criteria for an optimized user experience have already been presented in the appropriate sections. For example, the clarity of actions for the user was derived from case studies. From observations of human-to-human communication, it was determined that the synchronization of several windows, tolerance of alternative formulations and continuous conversation can be of decisive importance for the user. The evaluation of the structure of other chat interfaces has shown that these are only optimally suited for short messages and small amounts of data. The reusability of results also saves time and is therefore a further criterion. Without claiming completeness, these findings are summarized in Table 2.

Involved component	Criteria
Interpreter	• User instructions • Error tolerance • Continuous conversation
Server	• Synchronization of chat windows
Software robot	• Minimized data volumes • Storage of process results

Table 2. Derived criteria for an optimized user experience

The proof of concept is provided with the help of a sample process, in which the focus is not on the meaningfulness of the process itself, but on functionality. A user should be able to ask for an open vacancy, whereupon the chatbot asks him about his interests. After the user has stated an area of expertise, the software robot in the back end searches the public job advertisements on the company's website for the appropriate ones. The same process was also successfully completed with alternative formulations and minor intentional spelling mistakes. Figure 3 shows the execution of this example process in two parts from the user's perspective. After a general interest in a job has been expressed, a specialized subject area is asked for, which is shown in the left part. Note that the expression *AI* is internally mapped to the term *artificial intelligence*. The user is informed about the start of the process to ensure some patience, whereafter the software robot searches the company's site for jobs that contain the key phrase. Since two positions were available at the time of the test, the first one could be rejected and the second one was retrieved immediately from the cache without additional effort, which is shown in the right part.

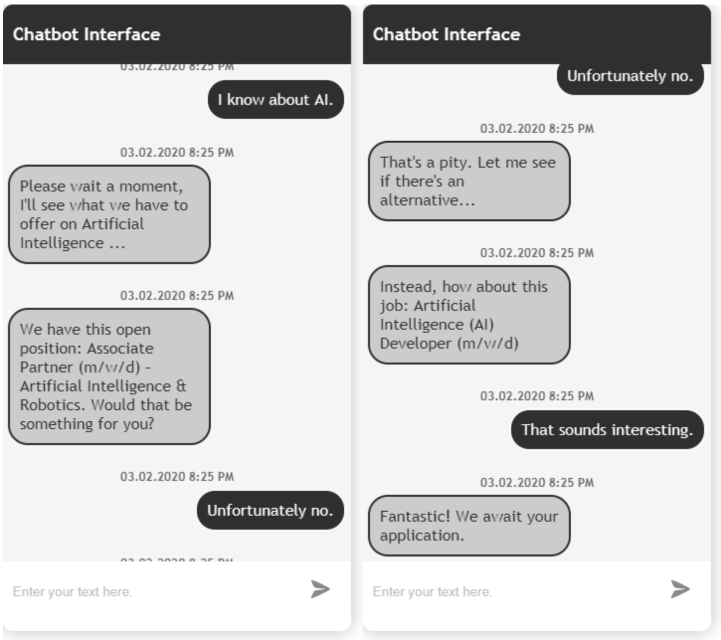

Figure 3. Example process result

The research questions from section two are briefly summarized in Figure 2 and Table 2. The proof of concept demonstrates that the fundamental structure based on the findings of the research questions can already functionally imitate a simple interaction with a service employee. Even if the amount of time required for a single process is quite high while the scope of the same is small, a basis for further advanced developments and research could have been provided. More complex processes, including authentication mechanisms and the exchange of sensitive data with appropriate security measures, are planned as the next step to extend the capabilities of this prototype.

5 Conclusion and Future Research

After a short introduction to the eponymous topics robotic process automation and chatbots, this paper examined the benefits and a possible design of a combination of both technologies. For the final evaluation of a practical implementation an IT artefact was developed and tested as proof of concept. In order to meet the emergent nature of the artefact, the design science research methodology was applied in an adapted form for this study. The technical foundations, including the criteria for the design and justification of the research interest, were determined by means of a literature review respectively the found and analysed references. Finally, the results of the study regarding the two RQs stated in the second section and the developed application were presented. In Figure 4, using the DSR grid by vom Brocke and Maedche (2019), the conducted research is depicted in a comprehensive overview.

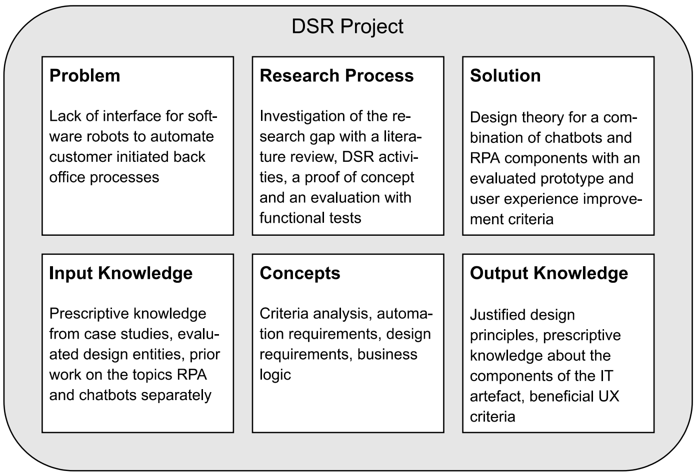

Figure 4. Conducted research in the DSR grid

Three evaluation phases were carried out. In the first phase, the basic idea was examined by means of the literature review regarding its actuality, relevance and existing concepts. The analysis of case studies and publications revealed a research gap and suitable applications for linking RPA with chatbots. The second evaluation phase covered the entire design process up to the design theory. All decisions were justified under consideration of the research questions. Further criteria, such as cost and complexity, also influenced the conceptual design. The elaboration of important criteria for the user experience prevailed in the development. The schematic representation of the structure supported the justifications and the thesis of correctness and completeness of the components. In the third evaluation phase, the developed prototype was tested in an artificial scenario with regard to its usability, efficiency, robustness and suitability. The handling of unusual situations was also observed with various configurations and inputs. Since there was never a program crash, the functionality for these explicit scenarios could be proven. In terms of the desired advantages over human labour, the study shows that for a single user, both the speed of process execution and the quality of the result meet the expectations. However, the test scenario of the proof of concept shows that if the number of requests is high, for example due to many concurrent users, longer waiting times can be expected. The improvement of the performance through more adapted workflows of the software robot is a priority for further research on this topic.

A fourth evaluation phase is planned after the commissioning in a real environment. Further adaptations and extensions are necessary before the prototype can be used in a company. Once this is the case, a case study will be conducted on outsourcing communication to the chatbot and testing the user experience and performance in the process. Surveys and expert interviews with the departments that benefit from this system can be part of the evaluation. Furthermore, the concept offers an immense room for future enhancements. For example, speech comprehension could be improved by analysing large amounts of data and using advanced cognitive solutions utilizing the capabilities of big data and AI. In addition, the integration of multimedia content such as images, videos and speech synthesis would be possible.

6 References

Aguirre, S., and Rodriguez, A. 2017. “Automation of a Business Process Using Robotic Process Automation (RPA): A Case Study,” in *Applied Computer Sciences in Engineering*: *4th Workshop on Engineering Applications, Cartagena, Colombia, 27.-29.09.2017, Proceedings*, J. C. Figueroa-García, E. R. López-Santana, J. L. Villa-Ramírez and R. Ferro-Escobar (eds.), Cham: Springer International Publishing, pp. 65-71.

Anagnoste, S. 2017. “Robotic Automation Process - The next major revolution in terms of back office operations improvement,” *Proceedings of the International Conference on Business Excellence* (11:1), pp. 676-686 (doi: 10.1515/picbe-2017-0072).

Asatiani, A., and Penttinen, E. 2016. “Turning Robotic Process Automation into Commercial Success – Case OpusCapita,” *Journal of Information Technology Teaching Cases* (6:2), pp. 67-74.

Berger, R., Ebner, M., and Ebner, M. 2019. “Conception of a Conversational Interface to Provide a Guided Search of Study Related Data,” *International Journal of Emerging Technologies in Learning (iJET)* (14:7), pp. 37-47.

Cooper, H. M. 1988. “Organizing knowledge syntheses: A taxonomy of literature reviews,” *Knowledge in Society* (1:1), pp. 104-126 (doi: 10.1007/BF03177550).

Dale, R. 2016. “The return of the chatbots,” *Natural Language Engineering* (22:05), pp. 811-817 (doi: 10.1017/S1351324916000243).

Gajra, V., Lakdawala, K., Bhanushali, R., and Patil, S. 2020. “Automating Student Management System Using ChatBot and RPA Technology,” *SSRN Electronic Journal* (doi: 10.2139/ssrn.3565321).

Heo, M., and Lee, K. J. 2018. “Chatbot as a New Business Communication Tool: The Case of Naver TalkTalk,” *Business Communication Research and Practice* (1:1), pp. 41-45 (doi: 10.22682/bcrp.2018.1.1.41).

Hevner, A. R., March, S. T., Park, J., and Ram, S. 2004. “Design Science in Information Systems Research,” *MIS Quarterly* (28), pp. 75-105.

Iyer, N., Gori, H., Kumawat, D., and Rote, U. 2019. “Bank Reconciliation Bot,” in *2nd International Conference on Advances in Science & Technology (ICAST) 08.-09.04.2019,* Mumbai, India.

Jones, D., and Gregor, S. 2007. "The Anatomy of a Design Theory," *Journal of the Association for Information Systems* (8:5), pp. 312-335 (doi: 10.17705/1jais.00129).

Kirkpatrick, K., and Wheelock, C. 2017. "Research Report - Robotic Process Automation," Tractica.

Lacity, M. C., and Willcocks, L. 2015. *What Knowledge Workers Stand to Gain from Automation*. https://hbr.org/2015/06/what-knowledge-workers-stand-to-gain-from-automation. Accessed 5 November 2020.

Maedche, A., Legner, C., Benlian, A., Berger, B., Gimpel, H., Hess, T., Hinz, O., Morana, S., and Söllner, M. 2019. "AI-Based Digital Assistants," *Business & Information Systems Engineering* (61:4), pp. 535-544 (doi: 10.1007/s12599-019-00600-8).

Peffers, K., Tuunanen, T., Rothenberger, M. A., and Chatterjee, S. 2007. "A Design Science Research Methodology for Information Systems Research," *Journal of Management Information Systems* (24:3), pp. 45-77 (doi: 10.2753/MIS0742-1222240302).

Rizk, Y., Isahagian, V., Boag, S., Khazaeni, Y., Unuvar, M., Muthusamy, V., and Khalaf, R. 2020. "A Conversational Digital Assistant for Intelligent Process Automation," in *Business Process Management: Blockchain and Robotic Process Automation Forum*, A. Asatiani, J. M. García, N. Helander, A. Jiménez-Ramírez, A. Koschmider, J. Mendling, G. Meroni and H. A. Reijers (eds.), Cham: Springer International Publishing, pp. 85-100.

Rutschi, C., and Dibbern, J. 2019. "Mastering Software Robot Development Projects: Understanding the Association between System Attributes & Design Practices," in *Proceedings of the 52nd Hawaii International Conference on System Sciences*, pp. 5971-5980.

Scheer, A.-W. 1999. *ARIS - Business Process Modeling*, Berlin, Heidelberg: Springer.

Scheer, A.-W. 2019. "The Development Lines of Process Automation," in *The Art of Structuring*, pp. 213-220.

Siderska, J. 2020. "Robotic Process Automation — a driver of digital transformation?" *Engineering Management in Production and Services* (12:2), pp. 21-31 (doi: 10.2478/emj-2020-0009).

Sonnenberg, C., and Vom Brocke, J. 2012. "Evaluations in the Science of the Artificial: Reconsidering the Build-Evaluate Pattern in Design Science Research," in *Design science research in information systems*: *Advances in theory and practice; 7th international conference, DESRIST 2012, Las Vegas, NV, USA, 14.-15.05.2012*, K. Peffers, M. Rothenberger and B. Kuechler (eds.), Berlin: Springer, pp. 381-397.

Staegemann, D., Volk, M., Daase, C., and Turowski, K. 2020. "Discussing Relations Between Dynamic Business Environments and Big Data Analytics," *Complex Systems Informatics and Modeling Quarterly* (23), pp. 58-82 (doi: 10.7250/csimq.2020-23.05).

van der Aalst, W. M. P., Bichler, M., and Heinzl, A. 2018. "Robotic Process Automation," *Business & Information Systems Engineering* (60:4), pp. 269-272 (doi: 10.1007/s12599-018-0542-4).

Vom Brocke, J., and Maedche, A. 2019. "The DSR grid: Six core dimensions for effectively planning and communicating design science research projects," *Electronic Markets* (6:1), p. 39 (doi: 10.1007/s12525-019-00358-7).

Vom Brocke, J., Simons, A., Niehaves, B., Reimer, K., Plattfaut, R., and Cleven, A. 2009. "Reconstructing the Giant: On the Importance of Rigour in Documenting the Literature Search Process," in *ECIS 2009 Proceedings*.

Weizenbaum, J. 1966. "ELIZA: A computer program for the study of natural language communication between man and machine," *Communications of the ACM* (9:1), pp. 36-45.