IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received November 22, 2021, accepted December 24, 2021, date of publication January 14, 2022,
date of current version January 24, 2022.

Digital Object Identifier 10.1109/ACCESS.2022.3143323

A Conversation-Driven Approach for Chatbot Management

GIOVANNI ALMEIDA SANTOS[1], (Member, IEEE), GUILHERME GUY DE ANDRADE[1], GEOVANA RAMOS SOUSA SILVA[1], FRANCISCO CARLOS MOLINA DUARTE, JR.[2], JOÃO PAULO JAVIDI DA COSTA[3], (Senior Member, IEEE), AND RAFAEL TIMÓTEO DE SOUSA, JR.[1], (Senior Member, IEEE)

[1]Department of Electrical Engineering, University of Brasília, Brasília 70910-900, Brazil
[2]National School of Public Administration, Brasília 70610-900, Brazil
[3]Department 2 Lippstadt, Hamm-Lippstadt University of Applied Sciences, 59063 Hamm, Germany

Corresponding author: Guilherme Guy de Andrade (guilhermeguy@aluno.unb.br)

This work was supported in part by the Brazilian National Council for Scientific and Technological Development (CNPq) under Grant 312180/2019-5 PQ-2 and through the National Institute of Science and Technology (INCT) on Cybersecurity under Grant 465741/2014-2, in part by the Brazilian Ministry of the Economy under Grant 005/2016 and Grant 083/2016, in part by the Administrative Council for Economic Defense (CADE) under Grant 08700.000047/2019-14, in part by the General Attorney of the Union-AGU under Grant 697.935/2019, in part by the National Auditing Department of the Brazilian Health System (DENASUS) under Grant 23106.118410/2020-85, and in part by the General Attorney's Office for the National Treasure (PGFN) under Grant 23106.148934/2019-67.

ABSTRACT Managing and evolving a chatbot's content is a laborious process and there is still a lack of standardization. In this context of standardization, the absence of a management process can lead to bad user experiences with a chatbot. This work proposes the Chatbot Management Process, a methodology for content management on chatbot systems. The proposed methodology is based on the experiences acquired with the development of Evatalk, the chatbot for the Brazilian Virtual School of Government. The focus of this methodology is to evolve the chatbot content through the analysis of user interactions, allowing a cyclic and human-supervised process. We divided the proposed methodology into three distinct phases, namely, manage, build, and analyze. Moreover, the proposed methodology presents a clear definition of the roles of the chatbot team. We validate the proposed methodology along with the creation of the Evatalk chatbot, whose amount of interactions was of 22,771 for the 1,698,957 enrolled attendees in the Brazillian Virtual School of Government in 2020. The application of the methodology on Evatalk's chatbot brought positive results: we reduced the chatbot's human hand-off rate from 44.43% to 30.16%, the chatbot's knowledge base examples increased by 160% whilst maintaining a high percentage of confidence in its responses and keeping the user satisfaction collected in conversations stable.

INDEX TERMS Chatbot, virtual assistant, content management, conversation-driven development, human-supervised learning.

I. INTRODUCTION

Chatbots are gaining more space in customer service since they reduce costs and speed up the whole customer support process. Furthermore, organizations can take advantage of the data collected through chatbot conversations to understand their customers, know their interests and their opinions about the offered services [1].

The first chatbot made was ELIZA between 1964 and 1966 [2] and the following years had improvements in the chatbot development techniques. Although some state-of-the-art chatbot algorithms emerged in those years, the hardware necessary for running them in a feasible time was not accessible or existent. Therefore, most of the chatbot developments were limited to academic and research purposes.

Since the 1990s, chatbots have been gaining space in the market and after the 2000s, especially after 2016, there was an even faster growth of interest on the subject [3]. This growth, consequently, brought new challenges such as how to design conversations and manage chatbot content. Concerning chatbot design, scalability and usability can be major issues since they have a direct impact on the user experience of a chatbot.

The associate editor coordinating the review of this manuscript and approving it for publication was Zijian Zhang.

This work is licensed under a Creative Commons Attribution 4.0 License. For more information, see https://creativecommons.org/licenses/by/4.0/

Scalability refers to the way a chatbot design handles the increase of users, interactions, and content contained in the chatbot, and usability refers to the actual usability of a chatbot design and if users are able to perform the desired tasks. Chatbot designers need knowledge about users' behavior before scalability and usability start to improve.

One way to provide great customer service using a chatbot is to acquire knowledge from conversations and evolve through content management. Chatbot content management is a challenge that includes a plethora of tasks that range from technological tasks (implementation, configuration), to data analysis, and content management. Chatbot content management requires a team with specific skills to deal with the chatbot big data. The incoming data needs to be transformed into new knowledge. This impacts customer satisfaction and the capacity of the chatbot to solve problems without human hand-off.

This paper proposes an approach to content management for textual chatbots supervised by humans. The method was validated by creating and maintaining the Evatalk chatbot, which had 22,771 interactions from May to December 2020.

This paper is organized as follows: chatbot design and methods are contextualized in Section II; a chatbot content management process called CMP is proposed in Section III; the results achieved through CMP's application in the Evatalk project are discussed in Section IV; and this work is concluded in Section VI.

II. BACKGROUND

In this section, we show that human supervised learning is essential in state-of-the-art chatbots. Besides the overview on different types of chatbots, current content management techniques and evaluation of chatbot performance are shown.

A. LIMITATIONS OF AUTOMATIC LEARNING

Many works sought to automate the construction of knowledge bases. In [4], neural networks were used to build a machine-machine conversational knowledge sharing where knowledge bases are built on top of other chatbot knowledge bases. While this is very useful for general-purpose agents, it is not applicable for domain-specific cases, such as user support for a company with its own services and use cases.

In [5], a reinforcement learning model was developed for expanding knowledge bases with open-world data from conversations, and, in [6], a self-feeding chatbot that uses sentiment analysis decides what is new knowledge and should be added to the knowledge base. Both works showed positive results in their testing, and it is clear that the conversation-based approach can increase the chatbot's predictive capacity. However, the automatic learning techniques have a caveat: they may start to learn the wrong answers and out-of-domain knowledge.

According to [7], the main limitations of automatic learning are: training a model on its own output reinforces right actions but also the wrong actions; users do not limit themselves to chatbot domain and automatic learning will not prevent learning outside the domain; a bad model will also be bad in knowing when the output was incorrect in case a chatbot uses user feedback for self-feeding.

In contrast, a collaborative knowledge-base construction by both machines and humans improves the informativeness of responses while balancing with better fluency and human-likeness when compared with a machine-only response [8]. These aspects strengthen the user experience and, consequently, users' trust in the brand that the chatbot is representing [9].

B. CHATBOT DESIGN

Artificial Intelligence (AI) is the capability of a machine to receive data, process it, and perform tasks based on the information extracted from the data received [10]. AI machines react by themselves according to the incoming data, and this reaction is adequate and similar to how a human would react to the same situation.

Even the simplest chatbot is an AI machine because when it receives a message, it answers by itself according to the message received. Chatbot design techniques are broad. However, we can separate chatbots into two main types [11]: the deterministic and rule-based model, and the probabilistic and machine learning-based model.

Some authors classify the rule-based approach as a non-AI method [12], [13]. However, the techniques applied in the rule-based approach also give AI capability to the chatbot. For example, the Artificial Intelligence Markup Language (AIML) is one of the tools used in the rule-based approach [13] and as the name implies, it uses AI.

Rule-based chatbots work with predefined rules [14]. The developer needs to list all possibilities in advance and the chatbot does not work if the input message differs from the predefined patterns, because the generalization is very limited. The conversation context is only understood if the developer programs it directly and objectively, and it works like a decision tree.

On the other hand, machine learning-based chatbots are trained on datasets called knowledge bases [14]. The most important thing is that the user does not need to send messages exactly as they are in the knowledge base, because, after training the chatbot, it can generalize new messages. The conversation context is comprehended the same way if the knowledge base contains conversation flows. The course of action is determined based on the probabilities of an incoming message being an intention registered on the knowledge base. The action chosen is the one with the highest probability or none if the probabilities are not high enough.

In cases where a chatbot needs to be built fast, there is limited scope, and personalized customer interaction is not needed, rule-based chatbots are sufficient and more accurate [15]. If the scope is too big and the chatbot is a vital part of a service, the rule-based approach is not the best option since the decision tree becomes huge and the program will have several nested conditions, increasing maintenance and evolution complexity.

For robust and scalable conversational agents, the best option is the machine learning-based approach. However, the knowledge base needs to keep up with users, since ''the experience of a particular interactive system may change over time as users grow accustomed to the system'' [16, p. 2]. Therefore, after the chatbot's release, the post-deployment content management must start to keep the knowledge base coherent with user behavior.

C. CONTENT MANAGEMENT

The chatbot content management process is a continuous cycle, allowing a fast iteration over the content to be able to react, and sometimes be proactive, towards changing user behavior. It consists of evaluating conversations and adding new data to the knowledge base.

The continuous learning of chatbots is essential, allowing that the chatbots can learn from and about users and can adapt to the enterprise requirements [17]. According to Subsection II-A, automatic learning is not precise and human-supervised learning is ideal, however, it must be a well-organized process, because the chatbot's quality may start to decline over time if the human-supervised learning process is not well-defined and organized, as bad decisions are made regarding its content.

Although the initial content fed before release is very important, the key to successful customer service with machine learning-based chatbots is to have a conversation-driven development for the chatbot content. That is, once the chatbot is released, the chatbot team must keep up with the conversations to better understand users, and also collect messages that should have been understood but were not.

One approach of human-supervised chatbot content management is annotation. This process consists of marking messages as valid or invalid and sending them back to model training [18]. In [19], annotation was done through crowdsourcing and participants marked chatbot responses as inappropriate, interpretable, and appropriate. The chatbot version with annotated data was preferred by most of the users and was better rated in terms of response appropriateness and the overall ability to engage users.

Incase of a high amount of messages a prohibitive time may be spent by the human support to do those annotations. Therefore, supporting tools are vital to reduce this time [20]. For example, WebChat is a chatbot content management tool that allows annotation on subjectivity, polarity, offensiveness, and swears. Its interface was designed specifically to facilitate accessing chat information, as well as streamline the annotation and evaluation of the collected data [18].

In view of the necessity of reviewing conversations in a dedicated platform, another work developed a web platform that connects to existing chatbots via API and provides an interface for domain experts to evaluate conversations and suggest improvements to the chatbots [21]. The most interesting feature of this work is the existence of deployment stages. Since domain experts are not necessarily programming developers, their suggestions go to a review dashboard for developers to approve the improvements. These stages guarantee that both business and technical aspects are correctly implemented.

Regarding available tools on the internet, three tools stand out: Rasa X,[1] Botfront[2] and Dialogflow.[3] The first two are made for chatbots developed with the Rasa framework, while Dialogflow is all-in-one with the chatbot itself and the management tools. Rasa X was developed by the Rasa team in order to support the conversation-driven development of chatbots and has been appearing in the literature [22], [23]. On the other hand, Botfront requires less technical knowledge from its end-users by not requiring markup languages to update content. Lastly, Dialogflow has everything in one place in the cloud, but it does not give the possibility of having your own server, while the Rasa stack and Botfront allow it by being open-source software. They are all powerful tools to build chatbots and choosing one of them depends mostly on the customer necessities.

The term Conversation-Driven Development (CDD) in the context of chatbots was proposed initially by the Rasa Team [24] considering the feedback of their engaged community and their experience in the field. Rasa X is actually a tool developed to support CDD, which makes conversations the main source of information to trigger changes in the chatbot content. However, CDD is centered on machine learning and the trained model performance. Therefore, in this work, we propose a conversation-driven approach that embraces both technological aspects and business goals for chatbot management.

In [25], we discussed an architecture for a chatbot capable of answering common questions for the Brazilian Virtual School of Government (EV.G). That proposal had drawbacks related to the inclusion of team members without a computer-related background. In Section IV, we present changes made to the architecture to address this and other issues related to content management.

D. EVALUATION METRICS

Once a chatbot is ready to go to production, the development cycle ends and evaluation starts. The conversations must be monitored to evaluate the chatbot's performance to ascertain if it is fulfilling its purpose. The first step is to efficiently store these conversations for a later review. Storing chatbot conversations requires a specific data model, in order to identify users, channels, and message time [26].

Having the conversations properly stored, analysts can work on extracting useful information for the company and from general performance. Chatbot data has valuable information for both its own improvement and for the brand it represents. Chatbots can be evaluated through five perspectives [27]: user experience, information retrieval, linguistics, business, and technology.

[1] https://rasa.com/
[2] https://botfront.io/
[3] https://dialogflow.cloud.google.com

User experience evaluation involves any metric coming from the user point of view. This can be a subjective or objective analysis based on surveys and direct contact with users. Before starting this evaluation it is important to have in mind what users expect from chatbots and that includes [17]: guidance, useful information, human-likeness, deal with lack of knowledge, demonstrate understating, and keep the conversation on track.

Information retrieval evaluation verifies the adequacy of chatbot content, that is, to inspect if it is capable to respond what the brand needs it to answer and if users' requests are being treated accordingly. This can be assessed by quantitative metrics or by having a domain expert (business analyst) inspecting the knowledge base to ensure the content is appropriated to the company goals [28].

Linguist evaluation refers directly to the chatbot persona because it determines how the chatbot communicates. Chatbot's main objective is to respond correctly to users, however, their success rate is highly coupled with the personal connection they establish with their users. One condition for this connection is that the chatbot shares the same vocabulary as its users [29]. Correspondingly, regardless of the language used, the chatbot must follow grammatical rules and have correct orthography.

The technology evaluation may indicate a technical assessment, but according to [27] it is about the humanity of the chatbot, that is, its capacity of understanding natural language and acting accordingly. Lastly, the business perspective investigates the costs of a chatbot, because chatbots must automate the business processes to save resources, which means that the value that the chatbot delivers or saves must be equal, or higher, than its costs. This value can be, but is not limited to, a monetary value.

E. CHATBOT CHALLENGES

A chatbot deals with different types of users. First-time users have no idea of the chatbot's capabilities they are interacting with. This can result in an explorer user that will not stay inside the defined content of a chatbot, testing sentences to see if the chatbot can understand it. Some users are not willing to interact with a robot or are expecting a human being. Some users have resistance or fear of technology and will not explore it, or be resistant to new ways to execute some tasks.

These unique behaviors require the chatbot team to practice introspection to predict the possibilities of user interaction, and this raises the importance of having a big, diverse, and multidisciplinary team [30].

The target audience's confidence in the chatbot's ability to answer properly has an influence on user interaction, as they will avoid interacting with a chatbot perceived as incompetent. A chatbot has similarities to commercial products as its branding carries a quality component that is perceived by its consumers. A chatbot brand can include many components, such as the combination of the name, personality traits, graphical interface, graphical assets, and inherited traits from its organization. Also, the chatbot can be used as a marketing tool to advance a brand's marketing efforts [31].

The chatbot may act as an organization's representative, so the impact of bad content development impacts not only the chatbot but also the brand reputation [9]. It is necessary to pay attention to the quality of users' interactions to act on points where many users find difficulties or are frustrated, to simplify the conversation flow.

III. THE PROPOSED METHOD

This work proposes a methodology for chatbots' content management. The Chatbot Management Process (CMP) is shown on Fig. 1 and contains six steps, divided into three phases. CMP is a cyclic process, adaptable to the organizational needs, and it is based on real users' conversations, which is the driving force of CMP.

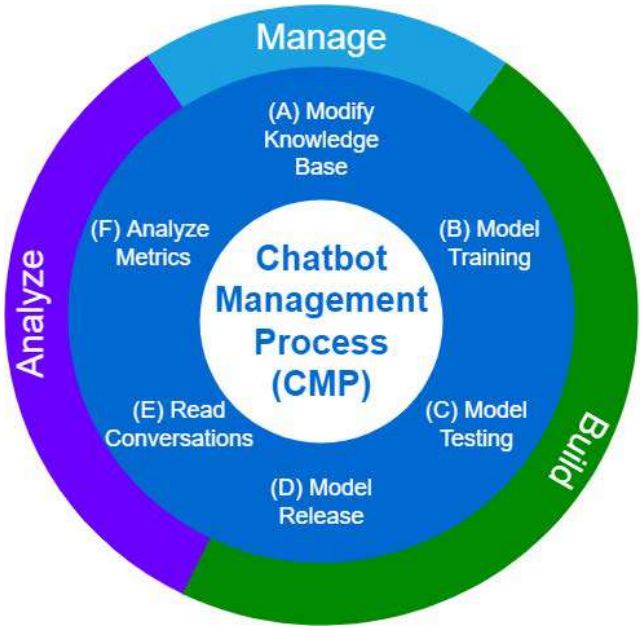


FIGURE 1. Chatbot management process proposed by this work.

The CMP is a human-supervised process due to the limitations of automatic learning, discussed in Section II. The phases in Fig. 1 will be explained in detail in the next sections, and they contemplate steps to make changes, test them and evaluate if they were effective. The CMP cycle does not stop as long as the chatbot is attending to users and saving their conversations' history.

One may ask why chatbot developers should not maintain the knowledge base by themselves without user conversations since they know the organization's goals and services, and the answer is: because they know it too well. Users certainly do not speak as employees. They usually use wordy sentences to describe a service or anything that the company, probably, has an internal name for it. That is why it is not a good idea to have a company employee providing content for the chatbot knowledge base.

However, when the chatbot was just developed and the company does not have a source of user interactions to feed the first version, it is expected that developers will take care of the content to have a minimum viable chatbot to respond to basic questions. Then, when the chatbot is released and starts interacting, the CMP starts and the content will be adjusted to fit users' way of communicating and most common questions that were not foreseen.

A. MANAGE PHASE

The *(A) Modify knowledge base* step includes the changes needed for the chatbot to perform as intended by developers. These changes can be the addition, removal, or modification of training data to improve the chatbot to understand the user's language or the adjustment of the conversation flow so users receive the appropriate responses. Additionally, action-based responses that execute computer code in the background are also corrected and improved.

Prior to release and when the cycle is starting for the first time, the content will be generated by the developers, and, if available, from another source of content from within the organization. Initial content is adapted to fit the data format of the technology used. If the cycle has already started and this step is not being executed for the first time, real users' conversations will be the content managed and fed to the chatbot.

If a chatbot has the responsibility to answer a lot of subjects or subjects are too closely related, then the complexity to achieve and maintain an acceptable level of quality is high. Limiting the content scope can help to create the first version and, through small increments, improve the chatbot and add new subjects to its content. Another benefit of a smaller initial scope is the acquisition of data about users's communication behavior on a smaller, and more manageable, scale. Besides that, each subject must be well-defined to reduce subject intersection and the possibility of the chatbot confusing them.

B. BUILD PHASE

The Build phase is composed of technological processes and quality assurance for content. The first step in this phase is the *(B) Model Training* in Fig. 1. Training a supervised machine learning model is the process of using the available training data and labeled examples to determine values for the hyperparameters and for all the weights and biases [32], [33]. The available data presents both input and labeled output data. The trained model is crucial to estimate outputs in applications where only the input data is available [34], [35].

If metrics do not present good results, the values of training hyperparameters must be changed and performances must be compared between changes to find the best-fit model. A poor choice of hyperparameters or an ineffective model testing mechanism will also result in a poor model [36].

The majority of chatbot creation tools rely on the notion of *intents* [37] as the training data, which are a set of typical human user conversational expressions [38]. Therefore, the training data is composed mainly of intents that are mapped to pre-defined responses. By training these intents, the chatbot is able to recognize similar unseen expressions in a conversation and give an appropriated response.

In step *(C) Model Testing* in Fig. 1, technical aspects and metrics must be taken into account. From a technical point of view, developers must assure that the training metrics are above proper limits. Examples of metrics include precision, recall and F-score [39]. The precision is the proportion of intents that the system returns which are accurately correct, recall is the proportion of intents that should have been returned and were correctly returned, and F-score is the combination precision and recall with a weighted harmonic mean.

Although these are the basic and more commonly used metrics for evaluating the resulting training model, in the context of chatbot development and natural language processing, there are specific metrics such as lexical diversity, average cosine-similarity, sentence average BLEU-2 score, and response perplexity [40].

From a user point of view, conversational flows must be tested to assure the responses are correctly mapped to questions. This testing is very important to detect human mistakes when mapping responses to each question, because this kind of error does not necessarily impact machine learning metrics and may not be detected by the previous testing. It is also important to test the chatbot with a focus on its domain by validating if the conversation flows are aligned with business goals, or if the communication style used on the chatbot is befitting to the business.

If problems are detected during testing, the workflow returns to step *(A)* for content correction and adjustment, and then, tests are run again. Once the model is evaluated and tests are passing, it is released to end-users. If it is the first cycle of the CMP, step *(D) Release Model* marks the release of the chatbot itself. Otherwise, the older model is replaced by the new one.

C. ANALYZE PHASE

The previous phases are related to getting the chatbot ready for production release and are common even in environments where the CMP is not implemented. The Analyze Phase is where the post-deployment management is concentrated, and it is the greater contribution of the CMP.

The Analyze Phase starts with step *(E) Read Conversations* because in the CMP all changes in the knowledge base are motivated by conversations. A user message is a powerful data source, providing useful insight into the user's behavior. The act of reading conversations is essential to understand user behavior. A chatbot's knowledge base created with no comprehension of the target audience is inadequate.

Step *(E) Read Conversations* is the driver for modifications that will take place in the *Manage Phase* on the next cycle. Here, conversational problems are identified for correction in the next phase. While reading conversations it is important to set aside these situations:

1) The user did not receive a response for a valid question.
2) The user received a response for a question but was not the response he wanted.
3) The user followed the expected conversation flow, but he was overall unsatisfied.

In the first case, the user asked something valid, but the chatbot gave the fallback response. Valid questions are sentences with full meaning that are part of the chatbot domain. For example, if a user messages ''I want to buy a bicycle'' for a chatbot that sells cars, that is not a valid question. However, if he asks the same sentence for a chatbot that sells bicycles, that is a valid question. Sometimes users ask questions that are in the domain but in an incomprehensible way, which is also an invalid question.

Invalid questions receive a fallback message, which is a default message that is sent when the chatbot does not know the response to a question. If the user sent an out-of-scope message, the fallback is the correct response, because it is not expected to answer that. Chatbots are built to respond to a limited range of questions that must be related to the business it is part of. Too large knowledge bases are prone to wrong responses because the chatbot has more options to analyze when making a prediction.

A fallback response may also represent an undetected problem with the model, because it may also happen when the chatbot cannot properly classify the user's message even when that message is present in the knowledge base. This is another motive for analyzing conversations to identify problems.

The second situation is when the chatbot responds to what it was trained to respond to, but the user was not satisfied with this answer. This happens when the knowledge base groups too many sentences together that may not represent the same intention or the answer is not adequate for the question. In both cases, it means that the user's feeling was not well represented on the knowledge base. It is a business problem rather than a technological problem.

The third situation is when the whole conversation happened as programmed and expected by the chatbot team, however, the user was not satisfied. In this situation, it is necessary to review the conversational flow in order to make it more user-friendly and make sure that it is the shortest path possible to the solution desired. Also, responses have to be reviewed in order to assess if they are conveying the message effectively.

Reading conversation by conversation is important to measure the chatbot performance according to each user, but the analysis of the conversations as a whole is also a powerful asset when checking the chatbot efficiency. This happens on Step *(F) Analyze Metrics* and it depends on conversation storage and data processing for defining metrics.

If conversations are well stored and processed, a wide range of metrics can be defined not only to evaluate the chatbot's health but also to drive business changes and validate the services offered to customers. However, business metrics are highly coupled with each organization, therefore we will focus on metrics related to the chatbot's efficiency, they are:

- Percentage of user messages that received the fallback answer.
- Percentage of conversations that the user did not solve his problem.
- User satisfaction regarding the customer service offered by the chatbot.
- Number of users that returned to talk to the chatbot.

Identifying the correct success criteria for each metric demands a careful analysis of content and goals. For example, if a chatbot has the feature of human hand-off when it does not solve the problem by itself, one may think that the success criteria is the creation of a support request. However, if the organizational goal is to reduce the number of support requests, the criteria is actually the proportion of interactions that did not request human hand-off.

User satisfaction can be explicitly collected through some kind of questionnaire in conversations or indirectly through analysis of conversations. For a direct approach, it is recommended that the chatbot presents the questionnaire when the conversation appears to have reached an end state. Unfortunately, this can apply a positive bias on data collected, since frustrated users may exit the conversation early, and satisfied users reach the end of conversation flows until the questionnaire.

After analyzing metrics, the CMP cycle restarts at the *Manage Phase*, propagating the discoveries and improvements generated in the previous cycle. However, the cycles are effective only if business rules and services do not change abruptly. In this case, it is necessary to start a knowledge base from scratch.

D. TEAM ROLES

The CMP requires more than just an educational background, and it is more about skill set. Chatbot development opened the door for several new roles, each one with a specific responsibility related to the chatbot and with CMP is not different. Actually, with CMP member roles are more necessary than ever because each step is centered in some conversational aspect.

The first and more obvious profile needed is Software Developers. However, each part of the chatbot needs a different kind of developer. The web interface needs frontend developers, while programmatic responses or actions need backend developers. It also depends on the level of programming for each conversational flow and chatbot framework used because each framework requires specific knowledge. These two kinds of developers may also work together in supporting platforms for CMP, such as a platform for managing the knowledge base and versioning.

The second profile required is Machine Learning Engineers, which are responsible for the selection and configuration of the Natural Language Processing algorithms. Their work is essential because it determines the chatbot's precision

when answering users. High-quality algorithms generate good language models and, as a consequence, the chatbot will respond with more confidence. If using a framework, engineers work on adjusting the settings of the framework. If they are building their own chatbot from scratch, their work is essential to create the Natural Language Understanding algorithms.

The third profile is Conversation Designers. They are responsible for the user experience and making sure the conversations are fluid and answers are clear. In the CMP, they are the ones that read the conversation history to identify bad conversation flows and understand what users expect from the chatbot. They translate business to chat conversations. This is the only role that does not necessarily require a computer-related technological background and can be assumed by linguists, pedagogues, designers or a developer. The ideal work set is having a team of Conversation Designers with different backgrounds.

The last main profiles are Business Analysts and Data Engineers. Conversations present valuable data for organizations, but data needs to be transformed into metrics. The data engineers have to process pure conversations to structural data while the business analysts have to identify metrics and build informational charts. Their work is important to check the chatbot health in the CMP and also for business evaluations.

IV. VALIDATION OF THE PROPOSED CHATBOT MANAGEMENT PROCESS ON THE EvaTalk CHATBOT

The CMP methodology was validated through the evolution of the EvaTalk project [25]. EvaTalk is a chatbot system created to handle common and recurrent user difficulties on the *Escola Virtual de Governo* (EV.G.), a Brazilian Virtual School of Government. Although its first version was functional and reduced human workload, the old architecture presented in [25] was failing in two aspects: usability and automation. Training models and updating the software required a significant effort. In the following subsection, the proposed upgraded architecture and its implementation are presented.

A. ARCHITECTURE

The new architecture reused most of the technological modules already developed but one module was completely refactored. In Fig. 2, we depict the proposed upgraded architecture, where the Module (C) related to development has been completely refactored using three components: *EvaTalk Admin*, *Data Repository*, and *Model Trainer*.

EvaTalk Admin is a platform built for encapsulating technical duties and enabling people with different skills other than Software Development to work with chatbot data. This way a team member can interact with the training data and create changes to the chatbot content while being trained only on key aspects of chatbot development, instead of having to learn how to operate more complex computer systems. The EvalTalk Admin is actually a set of services that provide

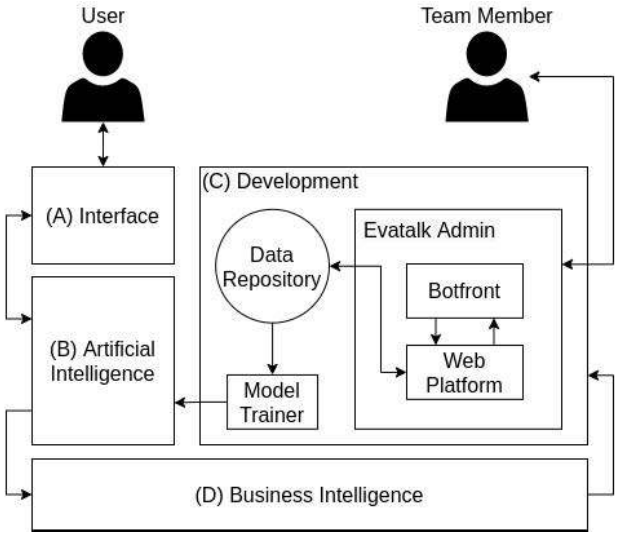


FIGURE 2. Proposed upgraded architecture, where the Module (C) has been completely refactored using three components: EvaTalk admin, data repository and model trainer.

user-friendly interfaces to make changes to the chatbot content. It is composed of the Admin Web Platform and Botfront. These tools simplify the tasks for the team members, reducing the amount of technical knowledge required.

The Admin Web Platform provides charts to check the chatbot performance, the chatbot conversation history, and an interface to execute Version Control actions. The charts support the *(F) Analyze Metrics* step by proving insights about the chatbot performance. Although the data used was existent in EvaTalk's previous version, it was necessary to use data analysis techniques every time any information was needed. In this new architecture, information is easily provided in the Admin Web Platform.

The conversation history supports the *(E) Read Conversations* step with a dedicated interface to navigate throughout user conversations with organized metadata to check how the chatbot classified each message. The version control page executes commit and resets the training data by only clicking buttons. Version Control is well disseminated across the Software Development world. However, as stated in Subsection III-D, Conversation Designers may not have a computer-related technological background, and it is not ideal that they deal with Version Control tools like Git, since it introduces difficulties and risks to the execution of their functions. This organization and interface require minimal technological knowledge, since it works like any other web page.

Botfront is an open-source application that provides a user-friendly interface to deal with the content of Rasa chatbots. It supports all three first steps: *(A) Modify knowledge base*, *(B) Model Training*, *(C) Model Testing*. Rasa already facilitates development by having training data with a markup language. Although these are very easy and straightforward languages for developers, it is still a technical skill that other

team members may have difficulties with. Although Rasa X turns the development more user-friendly, Botfront goes a step further by removing the necessity to deal with the markup languages in the training data, thus being easier for non-developers.

The Admin Web Platform communicates with Botfront in order to export the training data and send it to the *Data Repository*, which holds all history of changes made on the knowledge base in a Version Control System (VCS) called Git, allowing the tracking and management of changes in content. This way, if something is deployed and the chatbot starts behaving unexpectedly, then it can be reverted to a previous working state.

The *Model Trainer* watches for incoming changes in the *Data Repository* and trains a new model that will be available for the chatbot to consume, executing step *(D) Release Model*. Before the adoption of *Model Trainer*, models were trained in the same server as the chatbot application and the training automation mechanism was limited. The creation of these two components was essential to allow a greater separation between the chatbot's code and data, permitting more flexibility on both of them.

B. WORKFLOW

The first step is the interaction with *EvaTalk Admin* where team members can preview the changes, analyze conversations and have a safe environment. There, their changes are isolated from the chatbot users, and if an error occurs, it can be easily reverted without affecting end-users. After finishing the changes, they are sent to the *Data Repository*.

Before the content is published, it needs to be reviewed by a developer or an automated testing system, to ensure the technical quality of the content. That would be part of the CMP's *(C) Model Testing*. It is recommended that the developer making the review belongs to either the Software Developer profile or the Machine Learning Engineer profile from Subsection III-D. Only after this step, the content is promoted from development status to production status.

Periodically, the *Model Trainer* application checks the *Data Repository* for changes and then builds a new version of the model, serving it to the chatbot afterward. On EvaTalk the *Model Trainer* is deployed on each environment, with a specific content version from the *Data Repository* selected, and is used to build a model with the changes that matter for the current environment, it means that, for instance, it creates a model with the stable version on a production environment and an unstable version on a development environment.

C. ENVIRONMENTS

Achieving the automation is closely related to the adoption of a container orchestration approach to EvaTalk's components, bringing improvements to EvaTalk's maintainability and scalability. Currently, EvaTalk is deployed to three official environments: production, content management, and development. Other environments are used for specific tests. Each environment is a slightly different version of EvaTalk, created for a specific purpose.

The production environment is where the final user interacts with a stable version of the chatbot, the content management environment is a separated environment that allows team members to modify and test the chatbot's content, and the development environment is used for the testing of changes that are more related to the technology aspect of EvaTalk. The content management and the development environments are both used for development tasks, but with different focuses.

Each environment is described in a *YAML* file, composed by a stack of containers and its configurations, and each one of these stack files is saved to a repository, allowing the use of version control tools, and the standardization of configurations. One of these configurations is the content version from the *Data Repository* that is present in the environment.

With the usage of a containerization approach to our architecture, we decreased the differences between the production and development environments, it also allowed us to simplify the process to create and manage new environments for EvaTalk, while also facilitating the testing of modifications and system stability and deployment process.

With regard to the CMP's execution, the container adoption has a direct impact on the execution of the Build Phase since it provided a technologically stable and reproducible environment, which is taken to great effect since we can execute CMP's step *(B) Model Training* on any machine in a controlled manner by using the previously mentioned *Model Trainer* component. This way the content can be published in an environment identical to production for human testing before its release.

Also, before using containers, we needed to configure various settings on the host machine to execute certain tasks periodically or to start the software programs used in those machines. Now, these processes are integrated in the containers, avoiding fiddling with host machine settings. The Build Phase could be executed without containers, but the operational aspect is greatly simplified with their application.

D. RESULTS

EvaTalk's chatbot release was split into two phases to reduce the risk of interfering with the user support process on EV.G. In the first phase the chatbot was added on EV.G's website while the legacy support process was still working, without further modifications to the website. This way, not all users needed to interact with the chatbot to reach EV.G's support team, and if the chatbot failed there was another path to solve the user's issues. In the second release phase, the chatbot replaced the form used to open a support request. From this point forward all support requests needed to go through the chatbot.

As shown in the graph on Fig. 3, after the second release phase the chatbot provided by the EvaTalk System had a significant amount of sessions every month. This increase of sessions is due to the removal of the user support form, which

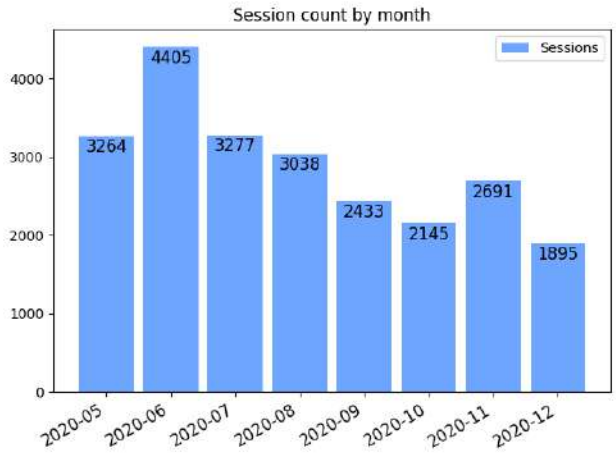


FIGURE 3. EvaTalk's sessions amount by month.

FIGURE 4. EvaTalk's average confidence by month.

was formerly the only way of sending a support request, and then it was sharing space with the chatbot. Also, the COVID-19 pandemic resulted in more people working from home and seeking online courses.

The chatbot framework used on EvaTalk classifies the received messages on intents and assigns a confidence degree to each message. This confidence is then used to decide what is the chatbot's next action. This confidence score is a value ranging from 0 to 1.0. Usually, a higher value means that the chatbot is correctly predicting users' intentions, a low value might be an indicator of problems in the components used to create models. These problems may come from the knowledge base data, or the machine learning algorithm, or the training process parameters. Fig. 4 shows the average confidence for users' messages since Eva's second phase release, aggregated by month. Initially, the confidence average was low because of the lack of data in the knowledge base and unoptimized model training parameters. Nevertheless, CMP's application increased the average confidence as our knowledge base grew in size and team members developed a better understanding of users.

However, the confidence score provided by the chatbot framework alone does not ensure that the chatbot is working properly, because it can give wrong responses with high confidence. We rely on two metrics to analyze the health of EvaTalk. The first one is based on business-oriented success criteria, the second one is the user satisfaction collected from users.

Each session is a user interaction and can result in one of three states: No interaction, No support, Support. The "No interaction" state happens when a user does not interact with the chatbot, only opening the chatbot interface, in other words, it is a session with no events sent from the user. The "No support" state is when the user opens the chatbot interface, sends at least one message, but does not create a support request for EV.G's support team. Lastly, the "Support" classification happens when a user ends up creating a support request within the chatbot interaction. The graph on Fig. 5 shows a comparison between the "Support" and "No Support" classifications from conversations of EvaTalk. Sessions without interactions were omitted since they would not provide relevant data.

The business-oriented success criteria for a conversation with EvaTalk's chatbot is the "No support" state since the chatbot goal is the reduction of support requests. From the graph on Fig. 5, it is possible to see that the majority of interactions are classified as "No support". This means that the chatbot application was able to reduce the demand for EV.G's support team, solving users' demands before they would open a support request. A way to analyze this data is to think that before the chatbot usage, every one of these interactions would end up as a support request. But beware, there is a limitation in this analysis, since we cannot measure how the chatbot's presence itself attracted more users to interact with it. Since the chatbot's release, the content evolution made through the application of CMP improved the success rate of the interactions with the chatbot.

At the end of conversations, the chatbot may ask the user about its satisfaction with the interaction with the chatbot. This happens when we are able to detect that the conversation has finished, however, our process to detect this state is not perfect which limits the number of users that are asked about their satisfaction and introduces a positive bias on our date, since, as previously stated on Subsection III-C, frustrated users may exit the conversation early. The impact of this bias on our data is reduced by the CMP's practice of reading messages. Also, buttons to finish conversations are present to induce users to indicate the end of a conversation and start the satisfaction form. The collection of the user satisfaction is split into two steps, the first step is to ask the user if they are interested in answering the satisfaction survey, if the chatbot receives a positive answer then it asks the user to evaluate its satisfaction with this interaction.

The satisfaction is collected in a range from 1 to 5, with 1 meaning the user was very unsatisfied and 5 meaning

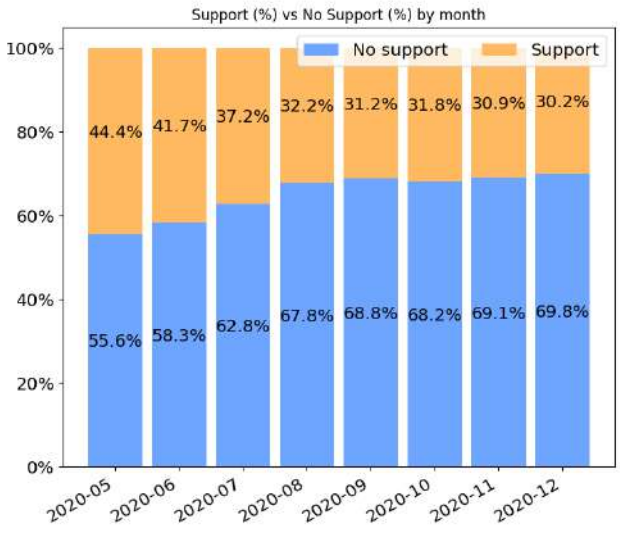


FIGURE 5. Percentage of sessions with user interaction classified as support *versus* no support on EvaTalk's sessions, by month.

that the user was very satisfied, which consists of a Likert-type question. According to [41], the suggested data analysis procedures for Likert-type data are frequencies, mode or median, kendall tau and chi-square. We compute frequencies for variability, mode for central tendency, and chi-square for statistical difference among months.

Regarding variability, Fig. 6 shows the distribution of responses in the satisfaction form from June 2020, when this metric started to be collected, until December 2020. This distribution also helps us identify the mode, which was 5 for all months, indicating a high satisfaction for most users. Executing a chi-square test on results produced a p value that was less than 0.05 ($p = 0{,}0024$), which indicates that results are significantly different from one month to another.

Fig. 7 shows an important metric for EvaTalk's training data: the number of examples present in the chatbot's knowledge base. This metric was collected by counting the number of training examples present in the knowledge base at the time of the last modification of the month. The time frame for this graph is different from others presented since it shows data from before the second release phase of EvaTalk. The second release phase happened in May 2020 and there was a surge of new interactions that were added as training examples through CMP's application since this was a period where the chatbot usage was intensified.

There is a relation between the amount of training examples and the chatbot's confidence. When combining data from Fig. 7 and Fig. 4, it is possible to see an increase in average confidence in June 2020, which is a consequence of the increase of training examples in May 2020. But, in July 2020, there was a decrease in chatbot's confidence, even though the number of intent examples increased on the previous month.

There are more factors that influence the quality of the chatbot's predictions. On EvaTalk's development, we found that the team's experience in classifying the training examples

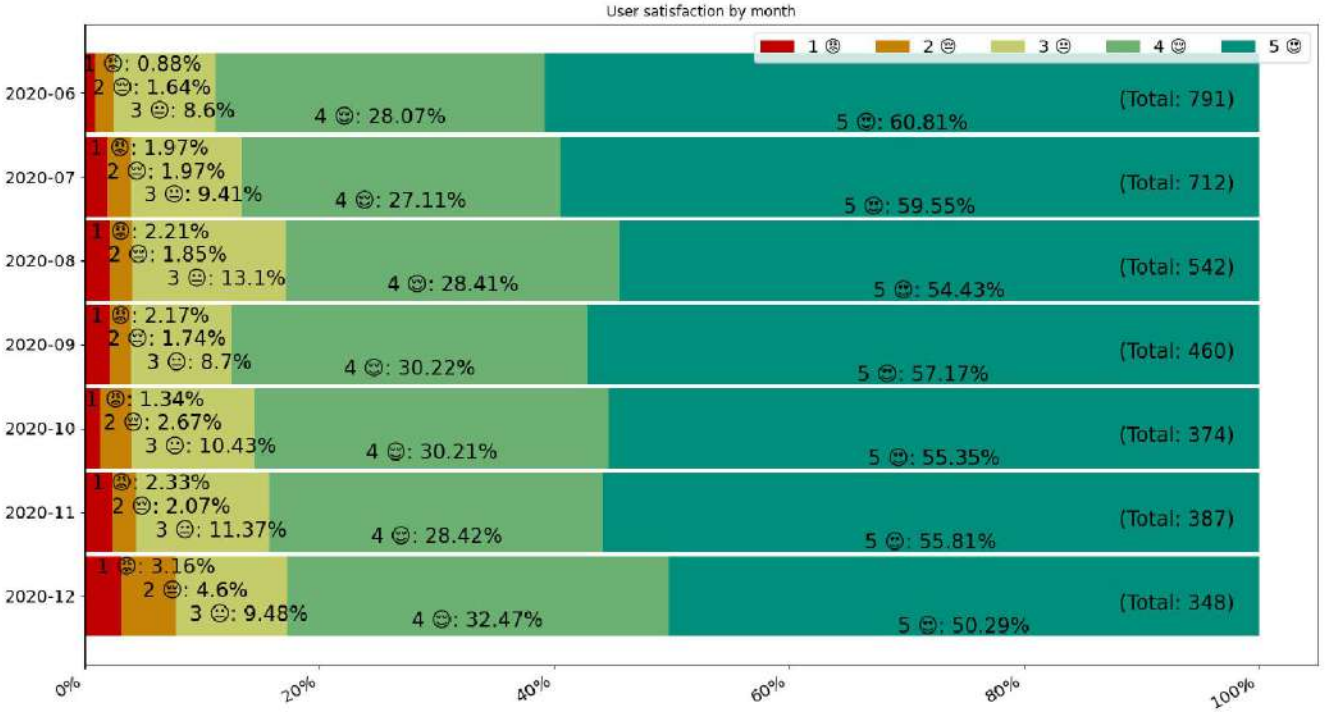


FIGURE 6. User satisfaction by month.

FIGURE 7. EvaTalk's training examples at the end of each month.

and the configuration of the chatbot's training process had a huge impact on the chatbot's confidence level. Throughout July and October 2020 the team changed the way messages were evaluated, tested different training parameters, cleaned up, and reclassified some training examples. These adjustments went through all CMP's phases. This resulted in a significant increase in the chatbot's confidence from August 2020 onwards. Besides the confidence increase, it is possible to see the maintenance of acceptable levels on the user satisfaction metrics and a reduction of human hand-off in the same period.

V. VALIDITY AND LIMITATIONS

Although Section IV presents a real implementation of the CMP, it does not cover all the possibilities to apply the method. Since it is purely a management process, it can be used in conjunction with different technologies to attend to different chatbot requirements. The first challenge for a company that is implementing the CMP is to elicit the changes needed to their existing architecture, in case there is one, or to design a new one.

Not all chatbot engines available in the market can support the CMP, and it may be necessary to design a personalized technological architecture, especially since each company has its own success metrics. Even though we present conventional metrics for chatbots, companies may have other ones for specific services offered by the chatbot, and it may require custom techniques to gather these metrics and also to visualize them in a meaningful way.

Note that scenarios with a high volume of user messages are out of the scope of this work. In EvaTalk's case, the month with the highest demand had around 1,100 messages per week. With a small team, it was possible to keep up with all the conversations, but that may not be feasible with thousands of messages per day, for example.

In the CMP, it is not strictly necessary to read every single message to be aware of how the chatbot is working and to execute the steps. For that reason, the team does not need to get through all conversations in scenarios with a vast amount of messages. Still, there is a challenge in filtering which are the important messages to be read and which are not.

VI. CONCLUSION

The CMP is a methodology to support machine learning chatbots on post-deployment management. It comprehends phases to change the knowledge base, build models, test modifications and analyze metrics to check the chatbot health. With CMP it is easier to delegate tasks to people with different skill sets, and the responsibility of each team member is well-defined.

The methodology was validated through the EvaTalk System, which is a complete platform that offers both the chatbot interface and management tools for post-deployment management. EvaTalk proved that with the right tools and people the CMP is scalable to attend a high-demand chatbot. Also, the analysis phase proved to be very important for the process, as long as the organization's goals are well aligned with metrics.

With CMP's application we reduced the chatbot's human hand-off rate from 44.43% to 30.16%, the chatbot's knowledge base examples increased by 160% (from 1059 to 2762 examples) whilst maintaining or increasing the chatbot's average percentage of confidence in its responses and keeping the level of user satisfaction collected in conversations stable, which means that the additions to the knowledge base were useful to users.

REFERENCES

[1] M. Akhtar, J. Neidhardt, and H. Werthner, "The potential of chatbots: Analysis of chatbot conversations," in *Proc. IEEE 21st Conf. Bus. Informat. (CBI)*, Jul. 2019, pp. 397–404.

[2] M. T. Zemčík, "A brief history of chatbots," *DEStech Trans. Comput. Sci. Eng.*, vol. 2019, pp. 14–18, Oct. 2019.

[3] E. Adamopoulou and L. Moussiades, "An overview of chatbot technology," in *Artificial Intelligence Applications and Innovations* (IFIP Advances in Information and Communication Technology), I. Maglogiannis, L. Iliadis, and E. Pimenidis, Eds. Cham, Switzerland: Springer, 2020, pp. 373–383.

[4] S. Arsovski, H. Osipyan, M. I. Oladele, and A. D. Cheok, "Automatic knowledge extraction of any chatbot from conversation," *Expert Syst. Appl.*, vol. 137, pp. 343–348, Dec. 2019. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0957417419304932

[5] S. Mazumder, N. Ma, and B. Liu, "Towards a continuous knowledge learning engine for chatbots," 2018, *arXiv:1802.06024*.

[6] J. J. Bird, A. Ekárt, and D. R. Faria, "Learning from interaction: An intelligent networked-based human-bot and bot-bot chatbot system," in *Proc. UK Workshop Comput. Intell.* Cham, Switzerland: Springer, 2018, pp. 179–190.

[7] B. Hancock, A. Bordes, P.-E. Mazaré, and J. Weston, "Learning from dialogue after deployment: Feed yourself, chatbot!" 2019, *arXiv:1901.05415*.

[8] J. Jiang and N. Ahuja, "Response quality in human-chatbot collaborative systems," in *Proc. 43rd Int. ACM SIGIR Conf. Res. Develop. Inf. Retr.* New York, NY, USA: Association for Computing Machinery, Jul. 2020, pp. 1545–1548, doi: 10.1145/3397271.3401234.

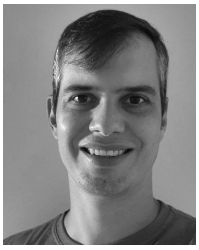
[9] A. Følstad, C. B. Nordheim, and C. A. Bjørkli, "What makes users trust a chatbot for customer service? An exploratory interview study," in *Proc. Int. Conf. Internet Sci.* Cham, Switzerland: Springer, 2018, pp. 194–208.

[10] M. Haenlein and A. Kaplan, "A brief history of artificial intelligence: On the past, present, and future of artificial intelligence," *California Manage. Rev.*, vol. 61, no. 4, pp. 5–14, Aug. 2019.
[11] L. F. D'Haro and R. E. Banchs, "Learning to predict the adequacy of answers in chat-oriented humanagent dialogs," in *Proc. IEEE Region 10 Conf. (TENCON)*, Nov. 2017, pp. 455–460.
[12] S. Nithuna and C. A. Laseena, "Review on implementation techniques of chatbot," in *Proc. Int. Conf. Commun. Signal Process. (ICCSP)*, Jul. 2020, pp. 157–161.
[13] S. Hussain, O. A. Sianaki, and N. Ababneh, "A survey on conversational agents/chatbots classification and design techniques," in *Proc. Workshops Int. Conf. Adv. Inf. Netw. Appl.* Cham, Switzerland: Springer, 2019, pp. 946–956.
[14] R. Agarwal and M. Wadhwa, "Review of state-of-the-art design techniques for chatbots," *Social Netw. Comput. Sci.*, vol. 1, no. 5, pp. 1–12, Sep. 2020.
[15] M. M. H. Dihyat and J. Hough, "Can rule-based chatbots outperform neural models without pre-training in small data situations: A preliminary comparison of AIML and Seq2Seq," in *Proc. 25th Workshop Semantics Pragmatics Dialogue*, 2021, pp. 1–3.
[16] A. Følstad and P. B. Brandtzaeg, "Users' experiences with chatbots: Findings from a questionnaire study," *Qual. User Exp.*, vol. 5, no. 1, pp. 1–14, Dec. 2020.
[17] A. P. Chaves and M. A. Gerosa, "How should my chatbot interact? A survey on social characteristics in human–chatbot interaction design," *Int. J. Hum.-Comput. Interact.*, vol. 37, no. 8, pp. 729–758, May 2021, doi: 10.1080/10447318.2020.1841438.
[18] L. Lin, L. F. D'Haro, and R. Banchs, "A web-based platform for collection of human-chatbot interactions," in *Proc. 4th Int. Conf. Hum. Agent Interact.*, Oct. 2016, pp. 363–366.
[19] Z. Yu, Z. Xu, A. W. Black, and A. Rudnicky, "Chatbot evaluation and database expansion via crowdsourcing," in *Proc. Chatbot Workshop LREC*, vol. 63, 2016, p. 102.
[20] X. Liu, W. Xue, Q. Su, W. Nie, and W. Peng, "metaCAT: A metadata-based task-oriented chatbot annotation tool," in *Proc. 1st Conf. Asia–Pacific Chapter Assoc. Comput. Linguistics 10th Int. Joint Conf. Natural Lang. Process., Syst. Demonstrations*, 2020, pp. 20–25.
[21] J. Feine, S. Morana, and A. Maedche, "A chatbot response generation system," in *Proc. Conf. Mensch Comput. (MuC)*. New York, NY, USA: Association for Computing Machinery, Sep. 2020, pp. 333–341. [Online]. Available: https://doi-org.ez54.periodicos.capes.gov.br/10.1145/3404983.3405508
[22] K. Deepika, V. Tilekya, J. Mamatha, and T. Subetha, "Jollity chatbot—A contextual AI assistant," in *Proc. 3rd Int. Conf. Smart Syst. Inventive Technol. (ICSSIT)*, Aug. 2020, pp. 1196–1200.
[23] A. Huddar, C. Bysani, C. Suchak, U. D. Kolekar, and K. Upadhyaya, "Dexter the college FAQ chatbot," in *Proc. Int. Conf. Converg. Digit. World Quo Vadis (ICCDW)*, Feb. 2020, pp. 1–5.
[24] A. Nichol. (May 2020). *Conversation-Driven Development*. [Online]. Available: https://tinyurl.com/ennbbf75
[25] G. G. Andrade, G. Silva, F. D. Júnior, G. Santos, F. L. de Mendonça, and R. S. Júnior, "EvaTalk: A chatbot system for the Brazilian government virtual school," in *Proc. 22nd Int. Conf. Enterprise Inf. Syst.*, 2020, pp. 556–562.
[26] M. Luca, A. Montresor, C. Caprini, and D. Miorandi, "An architecture-independent data model for managing information generated by human-chatbot interactions," in *Proc. 8th Int. Conf. Model-Driven Eng. Softw. Develop.* San Francisco, CA, USA: SciTePress, 2020, pp. 344–351.
[27] D. Peras, "Chatbot evaluation metrics," in *Economic and Social Development: Book of Proceedings*. Varazdin, Croatia: Varazdin Development and Entrepreneurship Agency (VADEA), 2018, pp. 89–97.
[28] W. Maroengsit, T. Piyakulpinyo, K. Phonyiam, S. Pongnumkul, P. Chaovalit, and T. Theeramunkong, "A survey on evaluation methods for chatbots," in *Proc. 7th Int. Conf. Inf. Educ. Technol.*, 2019, pp. 111–119.
[29] K. Kvale, O. A. Sell, S. Hodnebrog, and A. Følstad, "Improving conversations: Lessons learnt from manual analysis of chatbot dialogues," in *Proc. Int. Workshop Chatbot Res. Design*. Cham, Switzerland: Springer, 2019, pp. 187–200.
[30] F. A. M. Valério, T. G. Guimarães, R. O. Prates, and H. Candello, "Here's what I can do: Chatbots' strategies to convey their features to users," in *Proc. XVI Brazilian Symp. Hum. Factors Comput. Syst. (IHC)*. New York, NY, USA: Association for Computing Machinery, Oct. 2017, pp. 1–10, doi: 10.1145/3160504.3160544.
[31] M. Chung, E. Ko, H. Joung, and S. J. Kim, "Chatbot e-service and customer satisfaction regarding luxury brands," *J. Bus. Res.*, vol. 117, pp. 587–595, Sep. 2020.
[32] M. Kuhn and K. Johnson, *Applied Predictive Modeling*, vol. 26. New York, NY, USA: Springer, 2013.
[33] Google Developers. (Feb. 2020). *Descending Into ML: Training and Loss | Machine Learning Crash Course*. [Online]. Available: https://developers.google.com/machine-learning/crash-course/descending-into-ml/training-and-loss
[34] M. Mohammed, M. B. Khan, and E. B. M. Bashier, *Machine Learning: Algorithms and Applications*. Boca Raton, FL, USA: CRC Press, 2016.
[35] Microsoft Docs. (May 2021). *What is a Machine Learning Model?* [Online]. Available: https://docs.microsoft.com/en-us/windows/ai/windows-ml/what-is-a-machine-learning-model
[36] H. B. Braiek and F. Khomh, "On testing machine learning programs," *J. Syst. Softw.*, vol. 164, Jun. 2020, Art. no. 110542. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0164121220300248
[37] S. Pérez-Soler, E. Guerra, and J. de Lara, "Model-driven chatbot development," in *Conceptual Modeling*, G. Dobbie, U. Frank, G. Kappel, S. W. Liddle, and H. C. Mayr, Eds. Cham, Switzerland: Springer, 2020, pp. 207–222.
[38] E. Michiels, "Modelling chatbots with a cognitive system allows for a differentiating user experience," in *Proc. PoEM Doctoral Consortium*, 2017, pp. 70–78.
[39] L. Derczynski, "Complementarity, F-score, and NLP evaluation," in *Proc. 10th Int. Conf. Lang. Resour. Eval. (LREC)*, 2016, pp. 261–266.
[40] J. Sedoc, D. Ippolito, A. Kirubarajan, J. Thirani, L. Ungar, and C. Callison-Burch, "ChatEval: A tool for chatbot evaluation," in *Proc. Conf. North Amer. Chapter Assoc. Comput. Linguistics (Demonstrations)*, 2019, pp. 60–65.
[41] H. N. Boone and D. A. Boone, "Analyzing Likert data," *J. Extension*, vol. 50, no. 2, pp. 1–5, 2012.

GIOVANNI ALMEIDA SANTOS (Member, IEEE) received the bachelor's and master's degrees in computer science from the Federal University of Campina Grande (UFCG), Paraíba, Brazil, in 1998 and 2001, respectively. He is currently pursuing the Ph.D. degree in electrical engineering with the University of Brasilia (UnB). From 2001 to 2003, he worked as a Software Developer at the Brazilian Senate and the Federal Highway Police Department. From 2003 to 2010, he was a Lecturer in computer science at the Catholic University of Brasilia (UCB). Since 2010, he has been a Lecturer in software engineering with the UnB. His research interests include distance learning, machine learning, and autonomous driving.

GUILHERME GUY DE ANDRADE is currently pursuing the bachelor's degree in software engineering with the University of Braslía (UnB). From 2019 to 2021, he worked with the development of the Virtual Assistant with the Brazilian Virtual School of Government (EV.G), where he is also a Research Assistant at the Decision Technologies Laboratory (LATITUDE). His research interests include virtual assistants and software engineering.

GEOVANA RAMOS SOUSA SILVA received the bachelor's degree in software engineering from the University of Braslía (UnB), in 2021, where she is currently pursuing the master's degree in informatics. From 2019 to 2021, she worked with the development of the virtual assistant for the Brazilian Virtual School of Government (EV.G), where she is also a Research Assistant at the Decision Technologies Laboratory (LATITUDE). Her research interests include virtual assistants, human–computer interaction, software engineering, and programming education.



FRANCISCO CARLOS MOLINA DUARTE, JR., received the bachelor's degree in computer science and the bachelor's degree in software engineering from the Catholic University of Brasília (UCB), in 2005 and 2006, respectively. From 2014 to 2016, he worked as a Special Advisor with the Presidency of the Republic. From 2016 to 2018, he was the General Co-ordinator of information technology at the National School of Public Administration (ENAP). Since 2018, he has been the Technical Manager of the Brazilian Virtual School of Government (EV.G) project, composed by ENAP public servants and researchers from the University of Brasília (UnB).

JOÃO PAULO JAVIDI DA COSTA (Senior Member, IEEE) received the Diploma degree in electronic engineering from the Military Institute of Engineering (IME), Rio de Janeiro, Brazil, in 2003, the M.Sc. degree in telecommunications from the University of Brasilia (UnB), Brazil, in 2006, and the Ph.D. degree in electrical and information engineering from the Ilmenau University of Technology (TU Ilmenau), Germany, in 2010. From 2010 to 2019, he was an Associate Professor of signal processing at the UnB and he co-ordinated the Laboratory of Array Signal Processing (LASP) and several research projects. For instance, from 2014 to 2019, he co-ordinated the main project related to distance learning courses at the National School of Public Administration (ENAP) and a Special Visiting Researcher (PVE) project related to satellite communication and navigation with the German Aerospace Center (DLR) supported by the Brazilian Government. From March 2019 to July 2020, he was a Senior Development Engineer at the EFS in the areas of electric and autonomous vehicles. Moreover, from October 2019 to July 2020, he was a Lecturer at the Ingolstadt University of Applied Sciences also on the area of autonomous vehicles. Since August 2020, he has been a Permanent Professor of applied electrical engineering at the Hamm-Lippstadt University of Applied Sciences, Germany. He has published more than 195 scientific publications and patents. His research interests include automotive field, beyond 5G, GNSS, distance learning, and adaptive and array signal processing. He has obtained seven best paper awards in international conferences. He is an Editor of the Special Issue on Electric Vehicles Section: Review Papers of the journal *Energies*.

RAFAEL TIMÓTEO DE SOUSA, JR. (Senior Member, IEEE) received the bachelor's degree in electrical engineering from the Federal University of Paraíba (UFPB), Campina Grande, Brazil, in 1984, the master's degree in computing and information systems from the Ecole Supérieure d'Electricité–Supélec, Rennes, France, in 1985, and the Ph.D. degree in telecommunications and signal processing from the University of Rennes 1, Rennes, in 1988. He worked in the private sector, from 1988 to 1996. Since 1996, he has been a Network Engineering Associate Professor with the Department of Electrical Engineering, University of Brasília, Brazil, where he is currently the Co-ordinator of the Professional Post-Graduate Program on Electrical Engineering (PPEE) and supervises the Decision Technologies Laboratory (LATITUDE). He was a Visiting Researcher with the Group for Security of Information Systems and Networks (SSIR), Ecole Supérieure d'Electricité–Supélec, from 2006 to 2007. His professional experience includes research projects with Dell Computers, HP, IBM, Cisco, and Siemens. He has co-ordinated research, development, and technology transfer projects with the Brazilian Ministries of Planning, Economy, and Justice, as well as with the Institutional Security Office of the Presidency of Brazil, the Administrative Council for Economic Defense, the General Attorney of the Union, and the Brazilian Union Public Defender. He received research grants from the Brazilian research and innovation agencies, such as CNPq, CAPES, FINEP, RNP, and FAPDF. He has developed research in cyber, information, and network security, distributed data services, and machine learning for intrusion and fraud detection, as well as signal processing, energy harvesting, and security at the physical layer. He is also the Chair of the IEEE VTS Centro-Norte Brasil Chapter (IEEE VTS Chapter of the Year 2019).