IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received 14 June 2023, accepted 5 July 2023, date of publication 10 July 2023, date of current version 21 July 2023.

Digital Object Identifier 10.1109/ACCESS.2023.3293849

TOPICAL REVIEW

Model-Driven Approaches for Conversational Agents Development: A Systematic Mapping Study

ÁNGEL ANTONIO MARTÍNEZ-GÁRATE[1], JOSÉ ALFONSO AGUILAR-CALDERÓN[2], CAROLINA TRIPP-BARBA[2], AND ANÍBAL ZALDÍVAR-COLADO[2]

[1]Facultad de Informática Culiacán, Universidad Autonóma de Sinaloa, Culiacán 80013, Mexico
[2]Facultad de Informática Mazatlán, Universidad Autonóma de Sinaloa, Mazatlán 82017, Mexico

Corresponding author: José Alfonso Aguilar-Calderón (ja.aguilar@uas.edu.mx)

This work was supported in part by the Programa de Fomento y Apoyo a Proyectos de Investigación (PROFAPI) funded by Universidad Autónoma de Sinaloa (UAS), Mexico, under Project PRO-A8-042, Project PRO-A8-033, and Project PRO-A4-021; and in part by CONFÍE. The work of Ángel Antonio Martínez-Gárate was supported by Consejo Nacional de Humanidades, Ciencia y Tecnología (CONAHCYT), Mexico.

ABSTRACT Conversational agents are a piece of software widely used nowadays in several domains, such as e-commerce, customer support, education, among others. To build one, proficiency in distinct areas of knowledge is necessary. Moreover, its development demands the availability of frameworks, tools, and proposals to facilitate its application in several domain areas. Recently, the adoption of models in a software development process has increased through the application of Model-Driven Development in several areas to improve and increase productivity in software development. This work aims to provide state of the art through a Systematic Mapping Study regarding to Model-Driven Development approaches to automate or semi-automate the development of Chatbots. For this study, 429 scientific articles were analyzed, of which, after the inclusion and exclusion criteria, only 20 were considered primary studies.

INDEX TERMS Chatbot, model-driven engineering, model-driven development, systematic mapping study, review, conversational agent, bot, development frameworks, chatbot development methods, MDD, MDD chatbot development.

I. INTRODUCTION

Aconversational agent, also called chatbot or bot, is a computer program that simulates a conversation; it can communicate with the user through different means, such as voice or text. In [1], it was described as a system that can interact with human users with natural language. However, as [2] points out, it is software that imitates human conversations using artificial intelligence. Their use has recently increased considerably in several areas, such as education, customer service and recreational purposes.

The idea of communicating with a computer has been considered for years; since 1950, Alan Turing has already raised several similar concepts by addressing the question *can machine think*. In addition to proposing the Turing test to assess whether a machine could display intelligent behavior similar to or indistinguishable from a human [3]. Finally, in the year 1966, what is considered the first conversational agent was presented, which was called ELIZA [4], which allowed communication through natural language by analyzing the inputs, breaking down the phrases based on specific rules, thus obtaining the intentions through keywords and generating responses from this.

Chatbot development requires proficiency in several specialized areas of knowledge [5], including Software Engineering (SE), Usability, User eXperience (UX), Machine Learning (ML), and Natural Language Processing (NLP) in order to establish the best interaction with the user. Thus making a complex development process.

Over the years, several technologies and techniques have emerged to design, develop and implement conversational agents more efficiently. The main idea is to bring the

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

gap between an expert bot developer and a user without experience in software development. In this regard is the case of AIML (Artificial Intelligent Markup Language), a markup language designed exclusively to create conversational agents [6]. This technology was used to create the chatbot interface called ALICE (Artificial Linguistic Internet Computer Entity) [7]. Since then, its use has spread to create various agents. Regrettably, there is a lack of frameworks and tools to facilitate the construction of conversational agents by providing the components for user interaction and natural language understanding platforms [8].

Modeling has recently been emphasized as a valuable tool for understanding common errors in software development. This is because it allows the program's functionality to be developed to be specified and analyzed in more detail using graphical representation techniques or concepts that are understandable to anyone without technical programming knowledge. According to this, Model-Driven Development (MDD) has emerged as a paradigm for semi-automated software development. The goal is obtaining the software to build source code from the specification of its functionality in graphical models [9]. MDD provides advantages in productivity over other software development methods because the model simplifies the engineering process. It represents a software product's intended behaviors or actions before coding begins. The development team constructs models collaboratively, providing a set of clear definitions of what the software is and how it works. As a result, tests, rebuilds, and re-deployments can be faster when developing multiple applications than with traditional software development [9].

Despite the required complexity of the development and implementation of chatbots in real-world scenarios, there needs to be a reference guide in the context of MDD proposal to assist in implementing these systems systematically. Consequently, it is essential to establish a baseline to serve as a starting point when developing conversational agents employing MDD software development paradigm. This is necessary to develop reliable proposals that can be applied to the real world. Furthermore, this will satisfy user interaction expectations since requirements and chatbot system elements are highly interconnected.

This study aims to provide state of art concerning Model-Driven Development approaches for conversational agents development through a Systematic Mapping Study (SMS) [10] based on the protocol used in [11]. This study is oriented toward knowing the MDD proposals to automate the development process of conversational agents. Likewise, the evolution of the work on this topic will be shown, as well as providing a classification of the available information. This research was performed in order to provide a better understanding of current research in this area and identify opportunities for future work. Furthermore, these research outcomes allow researchers to analyze the current research directions in this area to identify the gaps that must be attended in order to apply this technology outside the research literature, i.e., in industrial cases. Our final goal in this SMS was to produce a classification for this area, giving ideas for researchers or professionals looking for information regarding MDD in conversational agent development. Moreover, those classifications can support researchers conducting primary or secondary studies.

The contributions of this SMS are as follows:

- This SMS includes literature from predefined databases, and based on predefined inclusion/exclusion criteria, 20 articles (primary studies) were included and selected from the 429 initial results.
- The MDD proposals found in the primary studies to automate the development of conversational agents.
- An exhaustive discussion of existing proposals and suggestions for new research directions.

The target audience of this research work refers to but is not limited to, software engineers, information technology managers, independent research developers related to the area of conversational agents, as well as research students.

This article is organized as follows: Section II presents the basic concepts in the context of this work. Then, Section III introduces the research protocol and the conduction of this SMS. The results obtained from the mapping and the analysis from the primary studies are presented in Section IV. Section V introduces a discussion derived from the research. Then, some threats to validity are detailed in Section VI. Finally, conclusions and ongoing work are presented in Section VII.

II. FUNDAMENTALS

In this section, the concepts that are useful for understanding the development of conversational agents will be described, such as Model-Driven Development (MDD) basis such as Domain-Specific Language, Model, Metamodel and Model-Driven Architecture, as well as Natural Language Processing, Natural Language Comprehension, Natural Language Generation, Intent, and Entity.

A. MODEL-DRIVEN DEVELOPMENT

Model Driven Development (MDD) is a software development paradigm in which models are used as the central artifact of the development process. It consists mainly of representing the system by employing models (modeling) and taking advantage of its characteristics; contrary to traditional programming paradigms, these techniques, systems, and programs will be created mainly by modeling, not coding. To achieve this, they use tools that allow the models to be transformed into code that the computer can interpret automatically.

Engineering problems often benefit from creating complex modeling systems. Certainly, since the software is one of the most complex engineering systems, these can be used similarly. However, it is rare that they are used to the leading role and are constantly relegated to the background in the development process. Model-Driven Development methods came to benefit from this area of opportunity, the main

characteristic of these methods being automation through models [12].

According to [13], the main objective of Model-Driven Development is to minimize the cost and effort of software production, seeking to improve the quality of applications independent of the platforms on which it is executed (referring to this at the level of abstraction at that is reached in MDD in which the logic of the problem is separated from the technological characteristics). Furthermore, in [14], it is stated that the motivation behind using MDD to improve productivity is to increase the profits obtained by software development companies after each project, and this is achieved through two main benefits obtained by using this paradigm: i) it increases the short-term production performance of programmers by raising the value of each software artifact created, and ii), long-term productivity is also improved, since the useful life of these primary artifacts is increased by reducing the speed at which they become obsolete. Because of this, models are being used to represent the problem in the real world, and they can be used independently of their technological architecture, thus expanding the possibilities of each piece of software since it is detached from the solution of your development and/or deployment platform making it possible to migrate or export the tool to another platform more quickly if desired.

B. DOMAIN-SPECIFIC LANGUAGE

A Domain-Specific Language (DSL) is a programming language designed to be used in a particular field. Unlike general-purpose languages (such as C# or Python), domain-specific languages are designed to solve a specific problem or are used for certain situations. For example, in [15], the authors defined languages explicitly designed for a specific domain, context, or company to facilitate the task of people who need to describe things in that domain. According to [16], a DSL provides predefined expressions of a high level of abstraction to represent concepts that belong to the application domain, which enables domain experts to understand, validate, modify, and even develop using the language without requiring programming knowledge. The DSL concept has become a popular research area within the field of Software Engineering (SE) [17], [18], [19]. Moreover, it is one fundamental component in software development methodologies such as Model-Driven Development [9], [20], [21]. A DSL is defined through three main components: abstract syntax, concrete syntax, and semantics.

C. MODEL

A model is a representation in a particular medium of a concept or idea of an object [22]. In SE, a model represents a system software to be built. This includes fundamental behavior, functionalities, and quality attributes. It is an abstraction of the relevant properties of the system software to construct. In [15], the author clarifies that a model implements at least two roles by applying abstraction: feature reduction, which implies that only the properties of interest of the object are reflected, and feature mapping, which indicates that it is abstracted and generalized from an original individual.

D. METAMODEL

A metamodel is the representation of a model (a model of another model), defined by the Object Management Group (OMG) [23] as a particular type of model that specifies the abstract syntax of a modeling language. At this level of abstraction, the concepts, rules, and all the components that will be used to define other models are specified. In SE, they can be seen as all the elements the developer can use in a model. When a metamodel is created, the principles and structures used in the models created later are defined. In the DSL universe, the abstract syntax defines the language concepts and their relationships and includes well-formed rules constraining the models that can be created. Metamodeling techniques are normally used to define abstract syntax. Metamodel is a component in the definition of a DSL, similar to defining the syntax, language rules, and semantics of a textual DSL.

E. MODEL-DRIVEN ARCHITECTURE

Model-Driven Architecture (MDA) is a specific procedure to implement MDD proposed by the Object Management Group (OMG). It is a framework that provides guidelines software developers should follow throughout the development process in a model-driven context. Following [24], MDA is a fundamental change from an object-oriented design to a model-driven one. Points out that the central idea of this methodology is to separate the platform-independent model, which is not related to the implementation technology and only contains the specification of the business logic. Subsequently, the conversion rules are defined according to the different implementation technologies in order to be able to convert the platform-independent model to a specific model platform to convert it into code finally. This is one of the bases ideas on which MDA separates development into three models: the Computation Independent Model (CIM), the Platform Independent Model (PIM), and the Platform Specific Model (PSM). These models comprise the entire development process before the code generation, which is done from the PSM [9]. MDA is used to implement an MDD software development process since it establishes that, from the CIM, the requirements will be specified in models, which, through model-to-model transformations (M2M), the PIM models will be obtained, specifying the functionality of the software but without considering aspects of the implementation technology (language, architecture, etc.). These PIM models, using the model-to-text (M2T) transformations, will be converted into models with technological aspects or simply in the product's source code to be developed.

F. NATURAL LANGUAGE PROCESSING

Natural language processing (NLP) is a computer science field investigating the communication between machines and people through natural languages such as Spanish, English, or Chinese. In [25], it is defined as a technology that enables computers to understand human languages. To achieve that objective, it is necessary to carry out a process in which people's natural languages are transformed into a coding system that a machine can interpret. Later, this information is processed by the computer, and finally, an action is specified, which can be to issue a response in the same natural language. Although this has many applications, such as speech recognition, text-based data mining, and text or speech generation, it has become a significant topic of interest [26].

G. NATURAL LANGUAGE COMPREHENSION

Natural language is the form of communication humans commonly use; processing this language to become valuable information for software is essential in developing chatbots. [27] describes natural language understanding (NLU) as a field of natural language processing that converts natural language into a semantic representation that a computer can interpret.

H. NATURAL LANGUAGE GENERATION

In order to carry out a conversation, the agent must not only be able to understand the user but must also be prepared to express himself with a method that is natively understandable to the human being. The authors in [28] define Natural Language Generation (NLG) as the natural language processing task of generating more natural language from a machine representation system with a base of knowledge using a logical form. An NLG system is like a translator that converts data into a natural language representation. The goal of an NLG system is to capture the meaning of the text and then create an information transmission procedure that a person can understand and use to interact with a system.

I. INTENT

In the context of a chatbot, the intention or the information that the user wishes to transmit in an expression is called *intent*, and a *intent* is a set of expressions with the same objective. In [29], the authors point out that every chatbot requires an intent recognition component to understand the user's goal or objective. The bot must classify the end user's expression into one of the predefined intents. Recognizing the *intent* within the *input* is essential because, in this way, the user's motivation is recognized when entering a phrase or command. An example is if the user enters the text ''I need the address of the hospitals in Sinaloa'', the *intent* should be detected as ''address of hospitals''. The bot would understand that the user is making a query in which he wants to obtain the address of the hospitals, and it can take the correct action in that situation.

J. ENTITY

In conversational agents, an entity is a primordial element in exchanging phrases between the chatbot and the user. Therefore, the bot must be able to recognize the entities of interest with this. Then, the necessary complementary information is obtained to carry out the requested actions.

The fundamental concepts introduced in this section are used throughout this SMS to provide a better understanding and ensure completeness in this research. Particularly concerning the research questions understanding and the discussion section comprehension.

III. METHODOLOGY

In this work, a Systematic Mapping Study (SMS) of the literature was conducted to identify and classify the technologies that use Model-Driven Development to automate or semi-automate the development process of conversational agents. At the same time, detecting the advantages and disadvantages offered over the development methods to recognize the areas of opportunity in this field. SMS is an analysis in which valuable information on a topic is synthesized, providing a structure through the classification of research reports and results, offering a summary of the available publications [10]. SMSs are primarily implemented as a beginning step to analyze primary studies or systematic reviews. However, this type of analysis is a fundamental early step in determining which explicit topics of a field it may be interesting to undertake a more detailed literature review. Following [11], an SMS is designed to structure a research area, collect and synthesize evidence, and present research findings. Furthermore, an SMS application allows for detecting gaps in the current literature.

The procedure that was implemented to perform this SMS is organized into five stages: (1) construction of the research questions, (2) search for primary studies, (3) selection of documents applying inclusion and exclusion criteria, (4) definition of the scheme classification and (5) classification analysis and systematic mapping.

As far as we know, the literature needs studies such as SMSs that point out the research trends of MDD in conversational agents. Therefore, the results of this review provide research topics regarding MDD approaches for conversational agents software development to boost the level of investigation in this field and bridge research gaps.

A. RESEARCH QUESTIONS

Conversational agents have become more popular during the past few years. Nonetheless, it is necessary to have expertise in techniques, programming languages, artificial intelligence, and platforms to become an expert in this field. Motivated by this observation, in this SMS, we aimed to obtain a more profound understanding of MDD applied to conversational agent development. To achieve this, a set of research questions were formulated. It will extract the data of interest from the publications and classify and

synthesize the information available on the technologies that use Model-Driven Development for the development process of conversational agents in the literature. These research questions are described below.

- Q1. Which research topics in Model-Driven Development are currently being addressed in the domain of conversational agents?
  This question is oriented to find out what is currently being applied in the conversational agents domain, mainly around Model-Driven Development.
- Q2. What technologies implement Model-Driven Development techniques in the development of conversational agents?
  This question focuses mainly on knowing the technique of MDD used in developing conversational agents as a DLS, metamodels, frameworks, APIs, etc.
- Q3. What challenges exist when using technologies that apply any Model-Driven Development techniques in the industry in the context of conversational agents development?
  This is important because identifying these challenges opens the door to improvements in this topic and allows to work on new proposals.

B. SEARCH FOR PRIMARY STUDIES

In order to answer the research questions, a selection of keywords was performed. First, synonyms and alternative names were identified to build the appropriate search strings for the information sources. The keywords obtained were *model driven engineering, model-driven development, MDE, MDD, chatbot, conversational agents, conversational system, automation, chatbot design, chatbot deployment, modeling, DSL, code generator, code generation, framework, tool, environment*. After an initial search, the keywords were refined until synonyms were obtained. Finally, the initial key terms were merged using the AND logical operator, whereas we used the OR operator for their synonyms. The search strings created are detailed in Table 1.

Afterward, the search process was performed to find articles to answer the three research questions defined in Section III-A. The bibliographic databases IEEEXplorer, ACM Digital Library, Science Direct, Springer Link, Web of Science, and Google Scholar were selected. According to [30], the selected databases are well-known and commonly adopted in secondary studies in the SE area. In addition, articles identified as ''grey literature'', such as theses and studies obtained from *OpenGray*, were also considered in this research. This process was chosen as it is fundamental to ensuring the accuracy and totality of the evidence.

C. INCLUSION/EXCLUSION CRITERIA

Once the search for primary studies was complete, 429 results were obtained. The next step consisted of filtering the total number of articles found. For this, the inclusion and exclusion criteria were defined. The criteria applied for inclusion were: i) the SMS includes studies available in English, ii) it will cover articles from scientific peer-reviewed journals, conferences, and postgraduate theses whose subject includes some technology that uses MDD techniques or discusses its use in the semi-automation of the development of conversational agents regardless of the domain, iii) research in which a method is reasonably present, iv) if a similar article were published in different sources by the same authors, only the recent/extended publication would be included, v) if an article published in a journal followed the same conference study, only the journal publication would be included, and vi) the keywords from the search string must be present in the title or abstract of the article; this allows to optimize the search. Concerning the exclusion criteria: i) publications in a language other than English will not be considered, ii) if there are two articles with similar content, one of which is a journal article and another conference article, the conference version will be eliminated, iii) if a primary study is found in a keynote, tutorial, poster section, or panel discussion session, it will not be considered for this systematic mapping study, iv) articles that do not address technology that uses MDD techniques or discusses its use in the semi-automation of the development of Chatbots were not considered, and v) not considered articles in the case that abstract-only papers, this is if the whole paper is not available or only found as an arXiv preprint (not published yet).

At the end of the selection process, 20 primary studies were obtained from the 429 initial results, see Table 2.

TABLE 1. Strings used in the search of primary studies.

Source	Search string
IEEEXplorer	(“Model driven engineering” OR MDE OR “model driven development” OR MDD) AND (Chatbot OR “conversational agents” OR “conversational system”) AND (automation OR modeling OR “chatbot design” OR “code generator” OR DSL OR “chatbot deployment” OR framework OR tool OR environment)
ACM Digital Library	(“Model driven engineering” OR MDE OR “model driven development” OR MDD OR “Model based”) AND (Chatbot OR “conversational agents” OR “conversational system”) AND (Automation OR modeling OR “chatbot design” OR “code generator” OR DSL OR “chatbot deployment” OR framework OR tool OR environment)
Science Direct	(“Model driven engineering” OR MDE OR MDD) AND (Chatbot OR “conversational agents” OR “conversational system”) AND (Automation OR modeling OR framework)
Springer Link	(“Model driven engineering” OR MDE OR “model driven development” OR MDD) AND (Chatbot OR “conversational agents” OR “conversational system”) AND (Automation OR modeling OR “chatbot design” OR “code generator” OR DSL OR “chatbot deployment” OR framework OR tool or environment)
Google scholar	(“Model driven engineering” OR “Model driven” OR MDE OR “model driven development” OR MDD OR “Model Based”) AND (Chatbot OR “conversational agents” OR “conversational system”) AND (Automation OR modeling OR “chatbot design”

D. CLASSIFICATION SCHEME

The studies were classified by year of publication in a linear diagram to analyze research papers' trends. Likewise, in order

TABLE 2. Primary studies obtained according to the inclusion/exclusion criteria for each source.

Database	Total results	Final results
IEEEXplorer	35	4
ACM Digital Library	20	0
Science Direct	19	0
Springer Link	98	6
Google scholar	257	10
Total	**429**	**20**

TABLE 3. Type of scientific research article.

Type of scientific research article	Description
Evaluation research	An investigation of a technical problem or solution in practice. Shows its benefits and disadvantages, the novelty does not necessarily reside in the technique or solution but in the criteria with which the phenomenon is being evaluated.
Proposed solution	A proposed solution to a problem must be novel or include an improvement or extension to an existing one. It usually shows the proposal's benefits through some resources such as examples or argumentation.
Validation research.	This research shows the characteristics of a proposal or solution; however, it has not yet been implemented in practice.
Philosophical article	Articles that propose a new way of address issues such as a new conceptual framework.
Opinion article	These articles contain the author's opinion about a solution and problem, its benefits and disadvantages, and his proposal for it.
Personal experience article	Articles that contain the author's opinion based on their experience in previous work are commonly used to share the experience of some industry professional or researcher who has prior knowledge of a problem or tools in practice.

to synthesize the type of information obtained from the primary studies, two classification schemes were constructed.

The initial classification scheme implements the scheme proposed in [31], see Table 3. First, the type and design of the scientific article is evaluated. This critical information allows us to obtain how investigations on the topic of interest are being addressed. As the author indicates, it is essential to mention that it is not always possible to frame a publication in a single class since sometimes the investigations have more than one objective that leads them to enter more than one category.

The second classification scheme is detailed in Table 4, it evaluates the type of contribution at a practical level. the goal is to determine how contributions and tools are emerging to users of the community in the development of conversational agents. It is essential because development automation techniques are commonly used in the business environment.

E. QUALITY ASSESSMENT

A quality assessment step was implemented, considering the work presented in [32], to improve the quality of this SMS. This evaluation was carried out to filter the studies collected in the search phase. The objective was to provide a method for selecting articles that would bring value to this research.

TABLE 4. Classification by type of contribution.

Type of contribution	Description
Library	Set of functions and/or procedures commonly used to be used within a specific domain.
SDK (Software Developmento Kit)	Set of tools for software development oriented to a specific platform.
API (Application Programming Interface)	Set of functions and procedures created to interact with an external platform.
Framework	A development environment with valuable tools for creating software that saves the developer time and effort.
Architecture	Guide or way of how software should be developed based on specific guidelines in order to have a general structure for future projects.

This evaluation classified articles with the minimum quality according to two criteria. Criterion number one was to consider the minimum quality standard of the primary study if the study was a research proposal, the research goal was clear, and if there was an appropriate explanation of the context in which the investigation was performed together with the reported results. The second criterion, objectivity, is if the research design was applied to the research goal. Then, the article provided detailed outcomes with reliable results and reached an acceptable conclusion. Ultimately, the article made significant contributions regarding relevant criteria that could be implemented for practical applications in the industry.

To apply the quality assessment, the authors designed a checklist in a spreadsheet formed by yes/no questions. Each article with at least one negative answer on the checklist was removed as a minimum quality threshold was essential for this analysis. Following the process was applied, a total of 20 articles were retained as primary studies for the extraction of information to answer the research questions from Section III-A.

The extraction of information from the 20 selected primary studies was carried out and classified using the schemes presented above. All the information obtained from this process was used to answer the research questions. This research generated new knowledge about trends in publications of technologies that use MDD to automate the process of developing conversational agents, making it possible to locate the opportunities in research in this area more efficiently and offering a broader perspective of the background on this topic.

IV. MAPPING RESULTS

This section introduces the results of the SMS through extracting information from the primary studies. Several publications from the literature focused on singular aspects of MDD in conversational agent development. The publications provided remarkable knowledge about the RQs. It is essential to accentuate that the 20 primary studies offered answers to more than one RQ. Moreover, information extra from

TABLE 5. Primary studies selected.

Title	Year
Creating and Migrating Chatbots with CONGA [33]	2021
Choosing a chatbot development tool [5]	2021
Xatkit: A Multimodal Low-Code Chatbot Development Framework [34]	2020
MDD based case tool for Automatic Generation of ChatBot [35]	2019
A Framework to Create Conversational Agents for the Development of Video Games by End-Users [36]	2020
Model-based Chatbot Generation Approach to Converse with Open Data Sources [37]	2021
Diplomat: A conversational agent framework for goal-oriented group discussion [38]	2021
Towards a model-driven approach for multiexperience AI-based user interfaces [39]	2021
Towards Automating the Synthesis of Chatbots for Conversational Model Query [40]	2020
Extensible Chatbot Architecture Using Metamodels of Natural Language Understanding [41]	2021
Towards Conversational Syntax for Domain-Specific Languages using Chatbots [42]	2019
Collaborative Creation and Training of Social Bots in Learning Communities [43]	2019
Architecture of a Framework for Generic Assisting Conversational Agents [44]	2006
A Modular Data-Driven Architecture for Empathetic Conversational Agents [45]	2021
Dynamic Natural Language User Interfaces Using Microservice [46]	2020
Model-Driven Chats: Enabling Chatbot Development for Non-technical Domain Experts Through Chat Flow Visualization and Auto-generation [47]	2021
Huey: Intelligent Agents for Natural Human to Machine Communication [48]	2021
Model-Driven Chatbot Development [49]	2020
Multi-platform chatbot modeling and deployment with the jarvis framework [50]	2019
Flexible modelling using conversational agents [51]	2019

the analysis of the primary studies obtained in the SMS performed is introduced: the primary study contribution type, the publication venues of each primary study proposal and suggestions obtained from the SMS.

A. RESEARCH QUESTIONS

The research questions proposed in this SMS were answered as follows.

Question 1. Which research topics in Model-Driven Development are currently being addressed in the domain of conversational agents?

In order to answer this question, the classification introduced in Section III-D of this study was considered. According to the primary studies analyzed in this SMS, current research focuses on the automation of chatbot development. To achieve this goal, several researchers have concentrated on creating their metamodels and DSLs to create development environments that allow building conversational agents more easily and quickly. According to the classification scheme detailed in Table 3, most of the articles are proposals for solutions, and in general, they have focused on creating frameworks in order for the user only focuses on using them to create his chatbots. Another topic of great interest is the automation of the platform migration process, which has been proven possible through independent platform models, such as Xatkit and Conga. Finally, the communication of external components and tools to automate development, or add new features for the conversational agent, is another topic of interest. The popularization of APIs under REST (REpresentational State Transfer) architecture stands out at this point. Table 6 shows the classification of each primary study according to the type of scientific research article.

TABLE 6. Results of type of scientific research article classification.

Type of scientific research article	Primary studies
Evaluation research	[5], [49]
Proposed solution	[33], [34], [35], [50], [36], [37], [38], [39], [40], [41], [42], [43] [45], [46]. [47] [48]
Validation research	[51], [39], [44]

TABLE 7. Type of technology applied on proposals extracted from each primary study.

Technology	Primary study	Total
DSL	[33], [34], [35], [39], [42], [48], [49]	7
EMF (ECORE MM)	[35], [37], [40], [41], [47], [49]	6
M2T	[35], [49]	2
REST	[35]	1
State Machine	[35]	1
UML	[37]	1
Pyhton	[38]	1
IFML	[39]	1
Deep Learning	[44]	1
MDA	[44]	1
Reinforcement Learning	[45]	1
Microservices	[46]	1

Question 2. What technologies implement Model-Driven Development techniques in the development of conversational agents?

From the 20 primary studies obtained, the most applied techniques for MDD in conversational agent development were extracted. The most implemented technology is DSL, 6 of the 20 primary studies implemented it on-domain language to improve the development of conversational agents in an MDD context. The second technique is Eclipse Modeling Framework (EMF) (including metamodel definition) with five appearances. This technology is used through the Eclipse IDE for the definition of metamodels through ECORE, a common standard for data models that many technologies and frameworks are based on. This includes server solutions, persistence frameworks, UI frameworks, and support for transformations among models. The other technologies that were found are model-to-text transformations (M2T), REST Architecture, UML Models, Phyton programming language, Interaction Flow Modeling Language (IFML), Deep Learning, Reinforcement Learning, Model-Driven Architecture (MDA), and Microservices, all of them with an appearance in each article. These technologies are detailed in Table 7.

The technological proposals that were found in the primary studies are shown next:

Creating and migrating chatbots with conga [33]. The authors in present an IDE called CONGA for creating and migrating chatbots. It provides a DSL to model it independently of a development tool. The tool has a recommendation module that enables the developer to be easily guided according to the requirements of the chatbot build. This approach come up with a code generator, which works through a module that allows extracting the properties from previously created conversational agent models. CONGA is one of the complete frameworks found in primary studies. Nonetheless, the reverse engineering process to obtain the models from an already developed conversational agent is only available for Dialogflow projects. Furthermore, it only allows migration to Rasa (open source machine-learning framework for automated voice and text conversations). Furthermore, manual intervention is necessary for some projects, i.e., when the bot needs the use of *emojis*, a feature that CONGA does not currently support. Finally, in using Google libraries, the authors pointed out that there are situations in which it is not possible to perform fully automatic migration due to the native technologies of each target platform.

Xatkit: A Multimodal Low-Code Chatbot Development Framework [34]. Xatkit, previously known as *jarvis Multiplatform Chatbot Modeling and Deployment with the Jarvis Framework* [50] is another of the most mentioned proposals in the selected articles. This is an open-source framework for conversational agent development applying MDD. This approach provides a base structure defined for each development project. Regarding MDD techniques, Xatkit implements its DSLs, which are independent of each other. This advantage permits easy reuse of all models from any stage of development in different projects. In addition, one of the most relevant components of this framework is *Platform Package*, which enables the user to define new platforms for the bot so that the scope of the conversational agent can be extended. Additionally, it includes a Xatkit Development Toolkit with which the developers can extend the project to others more efficiently.

MDD based case tool for Automatic Generation of ChatBot [35]. This is a tool developed to generate conversational agents through MDD. An ECORE meta-metamodel conforms to the proposal, an IDE for using the metamodel, and M2T *model to text* transformation rules. This approach allows generating of bots for Facebook. In addition, the modules bring support to integrate the chatbot to REST (Representational State Transfer) services. A conversational agent with support for REST services provides the ability to communicate with many tools, increasing the interoperability and compatibility of this proposal. Nevertheless, it does not use any natural language provider service, so the generated assistants' NLP is limited.

A Framework to Create Conversational Agents for the Development of Video Games by End-Users [36]. The research presents a framework for using and creating conversational agents focused on video games. The authors use MDD to create models through the design of a DSL, using their own natural language expressions. The framework presented is an Eclipse-based tool with a chatbot interface that works with Martin Fowler's DSL State Machine. Nevertheless, the method and tool for automating the creation of chatbots have been tested with a single language only, so the framework is one for specific cases (video games), not to create all kinds of chatbots for different platforms.

Model-based Chatbot Generation Approach to Converse with Open Data Sources [37]. The work presented proposes a tool to develop chatbots capable of interacting with OpenAPI through Xatkit and employing an Eclipse *plugin*. This approach can convert an OpenAPI specification to a UML model containing the definition of the conversational agent. One of the most relevant aspects of this proposal is that this UML model can improve the automation of the development of chatbots that allow queries to different OpenApis. Nonetheless, importing the API must be only of the type of Socrata, CKAN, OData, or OpenAPI. In any case, manual configuration is required when some data is missing in the API specification.

Diplomat: A conversational agent framework for goal-oriented group discussion [38]. The authors introduces a framework for developing goal-oriented discussion group moderators chatbots is presented. The tool consists of a set of Python classes that are used to define the agent's characteristics; these characteristics are classified into six types the transcript file, which contains the group chat's history, including the authors of the messages, and the timestamp of messages. The chatbot's interventions consist of sending messages, passive interactions (that are activated over time), reactive interactions (activated depending on specific user actions), the type of agent characteristics (conversation rules and relates the transcript file to the available interventions), and finally, the configuration features that modify the behavior of the agent application. It is a tool that facilitates the development of conversational agents in discussion groups, a specific field of application. However, technical programming knowledge is required because the framework implementation is formed by a set of Python (programming language) classes, and a DSL was not created. In addition, the entire transcript file needs to be reloaded each time there is a change. The proposal does not introduce an example of a chatbot developed with this tool, and only the "Wizard of Oz" (research experiment in which subjects interact with a computer system that is considered autonomous by the subjects but is actually operated or partially operated by a hidden human) approach was used to validate the development of the tool.

Towards a model-driven approach for multiexperience AI-based user interfaces [39]. This research, proposes a generalization of Xatkit's DSL for developing all types of conversational agents, which facilitates the definition of more complex flows and behaviors in the interaction with the user. The DSL abstract syntax is conformed by three components: the *intent package*, the *behavioral package*, and the

run-time package. The first one describes the concepts used to model the definition of *intents*, i.e., the types of events that can occur in the system and how they are provoked are described. In this component, the *intents* and *entities* for events triggered by a user are defined. The second is used to represent a state machine to model the bot's behavior in response to different events. The third one is the *run-time package*, is goal is to specify the classes that will be used at the run-time of the deployed bot. Once the conversational agent is defined, describing the relationships between the agent and the other software components is necessary. This is done through an IFML extension. This is helpful to specify the positioning and visibility of the chatbot in the graphical interface, as well as the information shared between the agent and the rest of the interface. This proposal shows an approach in which the development of conversational agents integrated with different components simultaneously can be facilitated. It is the first step to attacking the semi-automatized chatbot development problem. Nevertheless, several characteristics of the bots are necessary to add to the metamodel, such as access control through roles in which a user can or cannot consult certain information according to their user permissions.

Towards Automating the Synthesis of Chatbots for Conversational Model Query [40]. The study presents a scheme of development of chatbots to make queries using natural language. The scheme proposes linking an *intent* to each query type. To do that, the authors introduce an EMF *plugin* with which the Xatkit framework can query EMF models depending on the *intents* detected by the chatbot. In order to do this, the conversational agent developer needs to provide a domain metamodel defining the structure of the models that will be consulted. Then, it is necessary to complement the models with synonyms for each characteristic and class for the training of the NLP model. This information generates the agent capable of carrying out these queries. This scheme can be used to create chatbots that make queries from a series of configurations and models, also showing an extension to the Xatkit framework.

Extensible Chatbot Architecture Using Metamodels of Natural Language Understanding [41]. This primary study proposes an architecture to create chatbots based on a metamodel where the concepts of NLU services are defined. It allows the creation of metamodels for new services (currently developed Dialogflow and Rasa) and the rules to relate them to other metamodels. This architecture permits the developer to easily exchange the natural language provider of the chatbot as long as the corresponding metamodels exist. In addition, its metamodel brings support for the definition of roles and groups of entities.

Towards Conversational Syntax for Domain-Specific Languages using Chatbots [42]. The authors introduce a framework to add natural language support via chatbots to the syntax of domain-specific languages. This is done by employing an automated process for modeling the chatbot. First, it generates a general configuration and declares how to reference objects and features of the instantiable classes. The tolerable level of inconsistency allowed during the modeling process is supported. Then, the conversational agent developer adjusts this configuration manually, adding synonyms for the classes or features to declare that classes are not instantiable. After that, the framework synthesizes a description of the chatbot and finally implements the chatbot on a platform.

Collaborative Creation and Training of Social Bots in Learning Communities [43]. The proposal is a framework to create chatbots. The proposal considers the definition of properties, such as the environment with which the agent must interact, the actions of the bot, and the users with whom it can interact. Deep learning techniques allow these conversational agents to implement text classification and generation functions. The composition allows the creation of generative bots but simultaneously allows the use of *If-Then* statements if searching for a "retrieval chatbot" approach. Once these characteristics have been initialized, the bot can be induced to the training phase, where it is provided with training data to provide a knowledge base. It is one of the few studios that allows generating chatbots with a "generative chatbot" approach.

Architecture of a Framework for Generic Assisting Conversational Agents [44]. The primary study presents a framework for conversational assistant agents and explains the usefulness of using conversational agents to assist inexperienced users in handling software components. The authors state that it is necessary to have the option to develop these assistants with a time of effort and cost equal to or less than the development of the components themselves. Considering this, the framework is proposed, and this is based on the conventional sequence of help systems: 1) formalization of the user's request, 2) resolution of the request, and 3) presentation of assistance. The idea is that developers with little knowledge of NLP can develop these wizards and take advantage of MDA (Model-Driven Architecture) techniques to generate new components through models. The framework was tested with different components of different domains. However, they are all single components, so there is no validation in a more complex case. Furthermore, do not use any natural language processing provider services that use deep learning.

A Modular Data-Driven Architecture for Empathetic Conversational Agents [45]. In this article, an architecture is presented to develop a conversational agent that can be modified for different scenarios. An empathetic controller is integrated into this conversational agent, and it works with the dialogue history to perceive the emotions and feelings of the user as well as to create content according to their needs. The architecture is based on data-driven generative conversational agents and is divided into five types of modules. The first type processes the input by converting the sound into text and then uses an NLU component to extract the meaning of the text. The second controls the structure of the dialogue and generates a response that is converted from text to spoken dialogue. The third type of

module is in charge of extracting the input's high-level characteristics and trying to find the user's emotions and feelings. Finally, the fifth type aims to shape the output according to the global context of the conversation. In it are the empathy control modules, responsible for modeling the empathy system. Following this architecture, it is possible to create an agent's open-domain conversational algorithms using reinforcement learning algorithms to generate more empathetic conversational agents.

Dynamic Natural Language User Interfaces Using Microservice [46]. This work presents an automated approach to building conversational agents to replace or complement graphical interfaces. This is done by analyzing a set of microservices defined by the OpenAPI specification and a set of requirements written in natural language; with these means, each operation's operations, parameters, properties, and outputs can be inferred. All the parameters are collected to obtain a representative model. On the other hand, the *intents* of each requirement applies key phrase detection; each one is linked to the model. Additionally, multiple phrases are then generated to invoke each *intent* from the requirements, and the *intent* is detected. These phrases train a neural network that links the user's text to the *intents*. This approach automates the creation of conversational agents for existing or new systems as long as an OpenAPI specification of the microservices to be used is available, with which a large part of the development and effort time can be saved.

Model-Driven Chats: Enabling Chatbot Development for Non-technical Domain Experts Through Chat Flow Visualization and Auto-generation [47]. This approach proposes to use MDD concepts to map chatbot conversation flows. Presents a standardized graphical dialog flow symbology using MDD to create a more straightforward representation of dialog flow using XML-based formats, which is expected to improve traceability, testing, maintenance, and documentation. The corresponding metamodels were created to represent Watson Assistant (WA) dialog flows to achieve this. Furthermore, they defined the transformation rules, from the dialog representation of WA to the proposal and vice versa.

Huey: Intelligent Agents for Natural Human to Machine Communication [48]. In this work was developed an architecture which includes two new programming languages for creating voice-bots (software that combines the natural processing language and artificial intelligence to start conversations) and chatbots from grammar. Works independent of any operating system, particularly the cloud, environment, or proprietary hardware. These languages were created using the concepts of DSLs.

Question 3. What challenges exist when using technologies that apply any Model-Driven Development techniques in the industry in the context of conversational agents development?

According to the primary studies (see Table 5), research has focused on constructing a development framework to have a complete development environment in which chatbots can be created. The challenges lie that saving time and effort in bot construction is a complicated task that must be addressed. In this field, one of the problems that research has faced is the heterogeneous technologies necessary for designing and developing a conversational agent and many target platforms. It is challenging to create an integrated development environment considering all the existing possibilities, which could cause problems if it is necessary to migrate a bot created through the different proposals. Hence, the main challenge these technologies face is the ability to provide compatibility between platforms and different development tool providers. This challenge has been addressed mainly with the creation of metamodels that consider the possibility of creating new target platforms and development technologies, mainly natural language processing tools. In this regard, understanding how a metamodel works to specify abstract and concrete syntax and semantics in a metamodel represents complexity as a research challenge related to understanding the language engineering process. Suppose the industry overcomes the definition of metamodels to create tools for conversational agent development. In that case, the next issue is the challenge of techniques for analyzing models, checking their performance, as well as the fundamentals of what a good model is. Moreover, research needs to identify techniques and tools for traceability, i.e., to reflect system changes into model changes.

B. KEY LIMITATIONS FROM PRIMARY STUDIES

According to the analysis of the primary studies, certain key limitations are detailed next. Some proposals are limited to a particular technology to obtain all the benefits offered, i.e., CONGA, one of the complete frameworks found in primary studies, is only available for Dialogflow projects, and the migration technology is limited and cannot perform automatic migration. Xatkit platform, the second one most mentioned in literature for the capability of the component for the definition of new platforms for conversational agents, reverse engineering is not supported as well one of the most platforms for bots nowadays which is WhatsApp. Moreover, since Xatkit research is constantly active, there is a proposal for the use of IFML, a standard modeling language from the OMG (Object Management Group), to represent the interaction flow in an interface with users or other software components. In this regard, a limitation is the lack of access control through roles to regulate the user permissions to consult certain information during user interaction with the conversational agent.

As reported in primary studies, a common fact in MDD implementation in Chatbot proposals is the lack of support for natural language provider service, limiting the ability of the conversational agent. Further, some proposals support the automatic generation of conversational agents using MDD. However, their application examples are only with a single language and only applicable to a specific domain, not to different application areas or platforms. Additionally, the

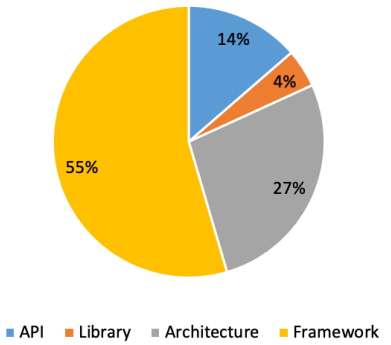

FIGURE 1. Classification of proposals.

approaches are tested using the Wizard of Oz to validate the development of the tool.

A few platforms are implementing API REST, but they are limited only to the type of Socrata and CKAN for open data, OData, which is an open protocol to allow the creation and consumption of APIs REST, and OpenAPI, a specification language for HTTP APIs that provides a standardized means to define the API to others consumers.

C. PRIMARY STUDY CONTRIBUTION TYPE

Several technologies that implement MDD techniques were found in the selected primary studies. Considering the classification presented in Section III, in Table 4, most primary studies, 55%, focused on designing a complete framework that offers the developer all the tools he needs to implement a conversational agent. The 27% proposes an architecture for MDD conversational agent development, and the 14% designed APIs for this goal. Figure 1 shows the primary studies' classified percentage. Moreover, to be more precise in the explanation, in Table 8, each primary study is organized according to the classification: Library (1), API (2), Framework (13), and Architecture (6). It is essential to highlight that although the Architecture and Framework categories are independent in the classification system used, most of these environments offer a pattern of concrete design to develop conversational agents. However, as seen in the classification, primary studies were also found that developed their architecture.

TABLE 8. Primary studies classification by type of contribution.

Contribution Type	Primary Study
Library	• *Diplomat: A conversational agent framework for goal-oriented group discussion* [38]
API	• *Towards Conversational Syntax for Domain-Specific Languages using Chatbots* [42] • *Dynamic Natural Language User Interfaces Using Microservice* [46]
Framework	• *Towards a model-driven approach for multiexperience AI-based user interfaces* [39] • *Towards Conversational Syntax for Domain-Specific Languages using Chatbots* [42] • *Model-Driven Chats: Enabling Chatbot Development for Non-technical Domain Experts Through Chat Flow Visualization and Auto-generation* [47] • *Flexible modelling using conversational agent* [51], • *Architecture of a framework for generic assisting conversational agents* [44] • *Creating and migrating chatbots with conga* [33] • *A Framework to Create Conversational Agents for the Development of Video Games by End-Users* [36] • *Model-based Chatbot Generation Approach to Converse with Open Data Sources* [37] • *Xatkit: A Multimodal Low-Code Chatbot Development Framework* [34] • *MDD based case tool for Automatic Generation of ChatBot* [35] • *Multi-platform chatbot modeling and deployment with the jarvis framework* [36] • *Collaborative creation and training of social bots in learning communities* [43] • *Model-Driven Chatbot Development* [49]
Architecture	• *Model-based Chatbot Generation Approach to Converse with Open Data Sources* [37] • *Towards Automating the Synthesis of Chatbots for Conversational Model Query* [40] • *Extensible Chatbot Architecture Using Metamodels of Natural Language Understanding* [41] • *Dynamic Natural Language User Interfaces Using Microservice* [46] • *Huey: Intelligent Agents for Natural Human to Machine Communication* [48] • *A Modular Data-Driven Architecture for Empathetic Conversational Agents* [45]

D. PUBLICATION VENUES

The publications concerning MDD in conversational agent development are dissipated in different sources such as journals, conferences, and workshops. According to this, articles on this research topic are not concentrated in specific sources, so it implies that very few specific venues publish this type of research. The International Conference on Conceptual Modeling (ER), The International Conference on Software Engineering (ICSE), and The International Conference on Web Engineering (ICWE) have published articles since 2019 regarding this topic; the three of them are part of the CORE Conference Ranking, which provides assessments of significant conferences in the computing disciplines. The rankings are managed by the CORE Executive Committee, with periodic rounds for evaluation for a request for a petition to be part of it. Hence, on the report on this SMS, these are one of the most popular venues for MDD and Chatbot research. Furthermore, the results of this research highlight that it is not a proper conference specialized in MDD for the generation of conversational agents from any field, so publication venues, even in the form of workshops, do not exist yet. Even though there are robust software engineering conferences, conference articles usually have space limitations, which restrict the area to disseminate the research results to the community.

TABLE 9. Source/venue extracted from each primary study.

Source/Type	Total of Primary Studies	Main Venue
Journal	7	• IEEE Access • IEEE Software
Conference	12	• The International Conference on Conceptual Modeling (ER) • The International Conference on Software Engineering (ICSE) • The International Conference on Web Engineering (ICWE)
Thesis	1	Harvard University

The few journal articles found in this SMS, only seven, means that twelve of the twenty primary studies have been published as part of conference and workshop proceedings with limited space and time. This fact may lack important and elaborate insights of interest to researchers and practitioners. According to the results of this article, six of the twenty primary studies obtained from research journals are from IEEE Access and IEEE Software, three on each one. Table 9 details the publication type/source with regard to MDD in conversational agents development obtained in this SMS, as well as the main venue.

E. SUGGESTIONS FOR FUTURE RESEARCH

According to this SMS, research efforts on MDD approaches for developing conversational agents have increased and maintained in recent years. Research in this area has focused on reducing the development time by proposing tools that improve automation using MDD through the construction of metamodels and DSLs to speed up and facilitate the creation of chatbots. In addition, to improving the migration process between technologies and platforms, architectures have also been proposed that allow using the information already available for constructing agents with Open APIs.

While it is true that research in this area has improved the integration of conversational agents in the industry, there are still challenges and areas of opportunity since each tool has its advantages but also has limitations. Some suggestions are detailed next:

- Improve the definition of metamodels, considering metaclasses for specifying user roles to improve the bot's capabilities.
- Another improvement area underlines the lack of use of natural language providers or DSLs to facilitate the development of conversational agents for non-expert users.
- Research articles often cite scalability as an area of MDD metamodeling that needs to be enhanced prior to extensive industrial adoption by any means possible. In this regard, research and tools have focused on the DSL (metamodel) creation instead of the maintenance of the DSL and the models created from it; that is, maintenance has not been considered and represents a challenge.
- Few articles present tools to create generative conversational agents. However, these have a broad knowledge base and are not focused only on a specific domain.
- Another challenge is obtaining the data to train their natural language processing models. In this regard, several businesses and companies often must wait to obtain years of data from interactions with users to have reliable data through interaction with its users.
- Invitation to the software engineering research community to establish MDD approaches for developing conversational agents-specific publication venues to consolidate this research area, which has been increasing since 2020. This suggestion could be a starting point to inform both the state-of-the-art and the state-of-the-practice quickly.
- Encourage the software engineering research community to publish more journal articles with rich insights on MDD-Chatbot development research to disseminate better the research work, which could bring the gap between research and practice in the industry of the proposals created in the academic research area.
- Methods, methodologies, and tools are fundamental elements and mandatory in guiding conversational agents' development, design, and simulation environments. Therefore, methodologies and tools to guide the development of chatbot definition and construction are exciting research areas that should be deeply studied in the literature. This is because it has been reported that the unmethodical application of complicated techniques, methods, and frameworks will most likely decrease effectiveness, increase development time, and tend to make systems error-prone, reducing interoperability and compatibility.
- There still needs to be a complete methodological MDD approach for the automatic generation of the bot for different platforms, from requirements to the code. Proposals analyzed in this SMS indicate that current MDD processes live aside from the requirements engineering phase.
- There still needs to be more design patterns or quality metrics for chatbots in current MDD proposals, nor do they offer an adequate method for software testing.
- Another interesting suggestion for the readers is using autoregressive language models to integrate general knowledge in the definition of DSLs in a model-driven context. This can help get information for the user. Hence, improving the function to answer the user's questions out the conversational agent knowledge base. This technology is gaining significant importance in natural language processing because of its classification

capabilities and performance in various tasks. Their integration in a DSL for a code generation tool has yet to be fully investigated.
- Since conversational agents must be continually maintained and modified to satisfy requirements not anticipated during analysis and design, novel methods need to be proposed to support specific software adaptability, scalability, and maintainability.
- The evaluation of conversation models needs to be standardized. A bot needs help understanding the complexity of human language and conversation, mainly in a multilingual conversation (i.e., another human language). This issue is necessary to solve because standardization of these models will likely be the breakthrough factor they need to get off the ground and increase technological tool development.
- Other suggestion for MDD chatbot development approaches is lacking support for multilingual language models. Although they have shown good performance in multilingual and cross-lingual natural language understanding tasks, none of the proposals from the primary studies in this SMS have multi-language support in their definition or DSL.

V. DISCUSSION

Through this systematic mapping, information has been obtained about the publications that show the use of Model Driven Engineering techniques to automate the development process of conversational agents. In this section, the different perceptions about trends in research and development are discussed, and the main advantages of the current proposals and the challenges in future works are faced.

Various technologies, tools, and platforms to develop chatbots have existed on the market for some time. In [5], the main development tools are analyzed where a comparison of the main characteristics of each technology is presented, and the essential factors to be taken into account to select one are indicated. These factors are divided into two categories: technical and administrative. However, it is essential to mention that this diversity of tools can create difficulties when choosing the right one, in addition to the problems that can arise if it wants to include a new post-development feature that was not initially planned. Hence, it is the possibility to migrate to another platform. The study concludes that the existing tools cover various possibilities and scenarios in implementing conversational agents. Although even using these tools, there are still challenges in the entire process of developing conversational agents, i.e., there is still a space for improvement in the development methodologies of the platforms since they do not offer design patterns or quality metrics for chatbots, nor do they offer an adequate method for software testing. On the other hand, most of the tools are focused on closed-domain bots because they use pre-defined training phrases to specify the *intents*. An important point to note is using different formats to define the chatbots by current approaches. Because of this, the W3C is currently working on standardization for conversational agent construction.

Nearly all of the primary studies proposed to solve user queries by detecting predefined *intents* in user messages. Also, proposals beyond these intents were identified, finding valuable information for the user in the companies' FAQ pages website, user manuals, OpenAPIs, or documents already available. These documents can be used to answer the questions that were not found in the defined *intents* because it is not always possible to contemplate all the cases and questions of the user; this is an advantageous alternative with which the chatbot can expand its knowledge base. The following alternative to get the information for the user must come from a more comprehensive data library, for which a language model that possesses general knowledge could be used, such as the Transformers GPT-3, GPT-J, GPT-NEO, Megatron-Turing, etc. These can be used as a last alternative to answer the user's questions, but none of the proposals found in this SMS consider these tools.

In recent years, there has been a growing interest in creating tools that improve conversational agent development. Some authors have used existing technologies to create frameworks and architectures capable of developing chatbots using a comprehensive tool. Part of the research done so far has sought component reuse and scalability. In addition, it has been sought that tools can be integrated through the use of DSLs. Other authors have defined their architectures using their methodology to provide conversational agents with NLP. This means there is no standardization in constructing this type of software. Some proposals even allow the creation of generative chatbots because they have the potential to have a broad knowledge base and converse about any topic.

The new proposals that have emerged to develop conversational agents are not exempt from presenting their challenges since they all have limitations. Firstly, if the proposed tool uses an external natural language processing service, it is necessary to integrate them by defining metamodels that allow the possibility of defining new technologies. This is important because we are talking about a changing environment, and if we rely on a single tool, it could change or even disappear. In addition, the different tools offer different features and utilities, so it would be dangerous to rely on only one. Likewise, compatibility must be a key factor and allow for migration. Another opportunity that was detected, and which is little explored in the technological proposals found in the primary studies analyzed in the SMS, is the possibility of providing chatbots with the ability to offer recommendations or suggestions to guide the user in the thread of the conversation to increase the ease of use of the agents. On the other hand, little attention is devoted to the proposals found to the integration with messaging platforms that are widely used in the chatbots market, such as WhatsApp, Telegram, etc., that both large companies and small businesses use to communicate with their customers. Because of this, it may be of interest to explore automation in integration and communication with these platforms using

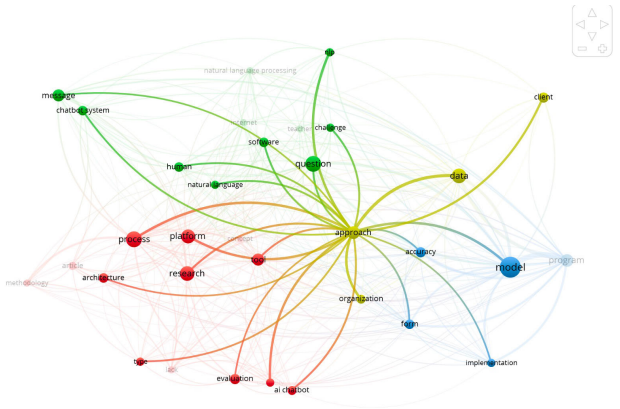

FIGURE 2. Bibliometric data visualization of the occurrences of the search terms.

existing tools. Therefore, another of the areas of opportunity detected focuses on the search for tools that make it possible to obtain data to train and define chatbots more easily. There are indeed proposals that present architectures that take advantage of REST APIs to define conversational agents with a certain degree of automation. However, these possibilities are not widely exploited in the scientific literature analyzed.

A bibliometric analysis was carried out, considering the terms detailed in Table 1. These terms were used in the search strings applied in each data source in this SMS. The analysis was performed using Vosviewer (https://www.vosviewer.com), which is a software designed to analyze scientific productivity during a period. Moreover, to identify trends in the literature regarding MDD and conversational agent development, an initial analysis of co-occurrences of keywords was performed, considering publications with a minimum of five incidences. This resulted in four clusters (see Fig. 2) involving thirty-four keywords. The clusters were (1) approach, (2) model, (3) question, and (4) platform. Our analysis showed a weak relationship but is still present with terms such as implementation and program (in blue), natural language and software (in green), platform, architecture, and process (in red), and organization (in yellow).

Finally, most of the research is focused on creating new proposals, but evaluation and validation research using previous proposals has begun to emerge. One recommendation is to promote evaluation and validation research and focus on improving or extending existing proposals. This would provide robust tools and more possibilities for developing standardized conversational agents.

VI. THREATS TO VALIDITY

This section discusses the potential threats to the completeness of the literature search.

A. IDENTIFICATION OF PRIMARY STUDIES

The first threat to the validity of this systematic mapping study is incompleteness. The risk of this threat highly depends on the selected search string, the limitation of the employed search engine, and the quality of the literature selection. To reduce the risk of an incomplete search string was constructed iteratively in order to get all the possible synonyms. The second threat concerns avoiding the limitations of the search engines used. Multiple search engines were considered, even Google Scholar, which can find too much gray literature and specific, well-known studies. In this regard, using it as something other than a standalone tool just for articles searched for systematic reviews is mandatory. Instead, the recommendation is to apply such an enhancement to other traditional search methods as a complement.

B. DATA EXTRACTION

The final threat to validity concerns the literature selection for data extraction; prior to an article exclusion, authors must validate that the paper is not relevant for the SMS to steer clear of the exclusion of relevant papers during literature classification. Even though the inclusion and exclusion criteria results have been double-checked and accepted by other authors, the articles were assessed and classified according to the author's expertise and criteria. It is a fact that other researchers may have evaluated the publications differently.

VII. CONCLUSION

This research presents the results obtained through a systematic mapping, thanks to which information was obtained about the state of the art in the field of automation of the development process of conversational agents through Model Driven Engineering techniques. A total of 429 publications from different scientific sources such as IEEEXplorer, ACM Digital Library, Science Direct, Springer Link, Web of Science, and Google Scholar. Finally, 20 primary studies were selected and analyzed by passing through the specific inclusion and exclusion criteria previously defined, which allowed answering the research questions, all to show the current state of research on this topic.

The systematic mapping study shows that research on this topic has increased in recent years, proving that it is currently a topic of interest for the scientific community. In this regard, most of the research has focused on developing new technological proposals to obtain more complete tools that facilitate and accelerate the development of conversational agents, thus reducing the cost and time for developers and companies. Although some articles are based on other proposals to extend operation, more studies are required to validate and evaluate the different approaches. Another of the areas of opportunity detected focuses on the search for tools that allow data to be obtained to train and define *chatbots* with greater ease. However, some proposals present architectures that use existing APIs to define conversational agents with a certain degree of automation; these possibilities are not widely exploited in the literature.

The application domain, training, and language need to be the subject of further research. The difference between the agents that emerged in the 1960s and those that have recently

gained momentum lies in the choice of specific languages for a predefined domain that allows the inference of responses to the user when the agent does not have the exact data. It also highlights that the use of autoregressive language models has not been fully investigated. This technology is gaining significant importance in natural language processing because its classification capabilities and performance in various tasks have not been fully investigated.

The SMS provides several recommendations for future studies, as mentioned in Section IV-E, which will give researchers some guidelines for future research work. As future work, we want to evaluate how autoregressive language models can be adopted in Model-Driven Development tools (starting with a DSL) and investigate new ideas that need to be developed to satisfy the need for a robust MDD tool for conversational agent generation. The priority is a MDD approach for generating standardized conversational agents for different platforms considering autoregressive language models for user response prediction.

Finally, concerning the current state of automation of generating conversational agents from models without the need to write source code, the lack of robust frameworks that allow for a model-driven process that guides the developer from the requirements phase to the implementation of the generated source code stands out. Thus, interoperability is still an important issue to solve.

ACKNOWLEDGMENT

The authors would like to thank the assistance provided by members of the Research Group Tecnologí-a Educativa I+D+i.

REFERENCES

[1] K. Rarhi, A. Bhattacharya, A. Mishra, and K. Mandal, "Automated medical chatbot," *SSRN Electron. J.*, pp. 1–15, Dec. 2017. [Online]. Available: https://papers.ssrn.com/sol3/papers.cfm?abstract_id=3090881, doi: 10.2139/ssrn.3090881.

[2] B. R. Ranoliya, N. Raghuwanshi, and S. Singh, "Chatbot for university related FAQs," in *Proc. Int. Conf. Adv. Comput., Commun. Inform. (ICACCI)*, Udupi, India, Sep. 2017, pp. 1525–1530, doi: 10.1109/ICACCI.2017.8126057.

[3] A. M. Turing, "Computing machinery and intelligence," *Mind*, vol. LIX, no. 236, pp. 433–460, Oct. 1950, doi: 10.1093/mind/LIX.236.433.

[4] J. Weizenbaum, "ELIZA—A computer program for the study of natural language communication between man and machine," *Commun. ACM*, vol. 9, no. 1, pp. 36–45, Jan. 1966, doi: 10.1145/365153.365168.

[5] S. Perez-Soler, S. Juarez-Puerta, E. Guerra, and J. de Lara, "Choosing a chatbot development tool," *IEEE Softw.*, vol. 38, no. 4, pp. 94–103, Jul./Aug. 2021, doi: 10.1109/MS.2020.3030198.

[6] M. S. Satu, M. H. Parvez, and Shamim-Al-Mamun, "Review of integrated applications with AIML based chatbot," in *Proc. Int. Conf. Comput. Inf. Eng. (ICCIE)*, Rajshahi, Bangladesh, Nov. 2015, pp. 87–90, doi: 10.1109/CCIE.2015.7399324.

[7] B. AbuShawar and E. Atwell, "ALICE chatbot: Trials and outputs," *Computación Sistemas*, vol. 19, no. 4, pp. 625–632, Dec. 2015, doi: 10.13053/CyS-19-4-2326.

[8] A. Abdellatif, D. Costa, K. Badran, R. Abdalkareem, and E. Shihab, "Challenges in chatbot development: A study of stack overflow posts," in *Proc. IEEE/ACM 17th Int. Conf. Mining Softw. Repositories (MSR)*, Seoul, Republic of Korea, May 2020, pp. 174–185.

[9] J. A. Aguilar, I. Garrigós, J.-N. Mazón, and J. Trujillo, "An MDA approach for goal-oriented requirement analysis in Web engineering," *J. Universal Comput. Sci.*, vol. 16, no. 17, pp. 2475–2494, Sep. 2010, doi: 10.3217/jucs-016-17-2475.

[10] K. Petersen, R. Feldt, S. Mujtaba, and M. Mattsson, "Systematic mapping studies in software engineering," in *Proc. Electron. Workshops Comput.*, Jun. 2008, pp. 68–77.

[11] J.-A. Aguilar-Calderón, C. Tripp-Barba, A. Zaldívar-Colado, and P.-A. Aguilar-Calderón, "Requirements engineering for Internet of Things (IoT) software systems development: A systematic mapping study," *Appl. Sci.*, vol. 12, no. 15, p. 7582, Jul. 2022, doi: 10.3390/app12157582.

[12] B. Selic, "The pragmatics of model-driven development," *IEEE Softw.*, vol. 20, no. 5, pp. 19–25, Sep. 2003, doi: 10.1109/MS.2003.1231146.

[13] Y. P. Rengifo, J. M. Suarez, and E. C. Correa, "Desarrollo dirigido por modelos (MDD) en el contexto educativo," *Scientia Technica*, vol. 20, no. 2, pp. 172–181, Jun. 2015, doi: 10.22517/23447214.9321.

[14] C. Atkinson and T. Kuhne, "Model-driven development: A metamodeling foundation," *IEEE Softw.*, vol. 20, no. 5, pp. 36–41, Sep. 2003, doi: 10.1109/MS.2003.1231149.

[15] M. Brambilla, J. Cabot, and M. Wimmer, *Model-Driven Software Engineering in Practice*. San Rafael, CA, USA: Morgan & Claypool, Sep. 2012.

[16] J. D. Sosa, O. Diaz, and S. Trujillo, "Defining DSL expressions collaboratively in multidisciplinary embedded engineering," in *Proc. 37th EUROMICRO Conf. Softw. Eng. Adv. Appl.*, Oulu, Finland, Aug./Sep. 2011, pp. 217–220, doi: 10.1109/SEAA.2011.41.

[17] B. Hoffmann, N. Urquhart, K. Chalmers, and M. Guckert, "An empirical evaluation of a novel domain-specific language—Modelling vehicle routing problems with Athos," *Empirical Softw. Eng.*, vol. 27, no. 7, pp. 1–52, Sep. 2022, doi: 10.1007/s10664-022-10210-w.

[18] B. Jahic, N. Guelfi, and B. Ries, "SEMKIS-DSL: A domain-specific language to support requirements engineering of datasets and neural network recognition," *Information*, vol. 14, no. 4, pp. 1–25, Apr. 2023, doi: 10.3390/info14040213.

[19] A. J. Salman, M. Al-Jawad, and W. Al Tameemi, "Domain-specific languages for IoT: Challenges and opportunities," *IOP Conf. Series, Mater. Sci. Eng.*, vol. 1067, Dec. 2021, Art. no. 012133.

[20] D. R. Cacciagrano and R. Culmone, "IRON: Reliable domain specific language for programming IoT devices," *Internet Things*, vol. 9, Mar. 2020, Art. no. 100020, doi: 10.1016/j.iot.2018.09.006.

[21] A. Sabraoui, A. Abouzahra, K. Afdel, and M. Machkour, "MDD approach for mobile applications based on DSL," in *Proc. Int. Conf. Comput. Sci. Renew. Energies (ICCSRE)*, Agadir, Morocco, Jul. 2019, pp. 1–6.

[22] P. A. Fernández-Sáez, "Un análisis crítico de la aproximación model-driven architecture," M.S. dissertation, Departamento de Ingeniería del Software e Inteligencia Artificial, Universidad Complutense de Madrid, Madrid, Spain, 2009.

[23] (2005). *Object Management Group Terms and Acronyms*. Accessed: Oct. 10, 2021. [Online]. Available: https://www.omg.org/gettingstarted/terms_and_acronyms.htm

[24] Y. Liu and Y. Wang, "A study of metamodeling based on MDA," in *Proc. 3rd Int. Conf. Comput. Res. Develop.*, vol. 2, Mar. 2011, pp. 171–173, doi: 10.1109/ICCRD.2011.5764107.

[25] D. Wang, J. Su, and H. Yu, "Feature extraction and analysis of natural language processing for deep learning English language," *IEEE Access*, vol. 8, pp. 46335–46345, 2020, doi: 10.1109/ACCESS.2020.2974101.

[26] S. Jaf and C. Calder, "Deep learning for natural language parsing," *IEEE Access*, vol. 7, pp. 131363–131373, 2019, doi: 10.1109/ACCESS.2019.2939687.

[27] A. Nigam, P. Sahare, and K. Pandya, "Intent detection and slots prompt in a closed-domain chatbot," in *Proc. IEEE 13th Int. Conf. Semantic Comput. (ICSC)*, Newport Beach, CA, USA, Jan. 2019, pp. 340–343, doi: 10.1109/ICOSC.2019.8665635.

[28] M. Virkar, V. Honmane, and S. U. Rao, "Humanizing the chatbot with semantics based natural language generation," in *Proc. Int. Conf. Intell. Comput. Control Syst. (ICCS)*, Madurai, India, May 2019, pp. 891–894, doi: 10.1109/ICCS45141.2019.9065723.

[29] I. Qasse, S. Mishra, and M. Hamdaqa, "IContractBot: A chatbot for smart contracts' specification and code generation," in *Proc. IEEE/ACM 3rd Int. Workshop Bots Softw. Eng. (BotSE)*, Madrid, Spain, Jun. 2021, pp. 35–38, doi: 10.1109/BotSE52550.2021.00015.

[30] B. Kitchenham and S. Charters, "Guidelines for performing systematic literature reviews in software engineering," School Comput. Sci. Math., Keele Univ., Dept. Comput. Sci., Univ. Durham, Durham Univ. Joint Report, EBSE Tech. Rep. 2007-001, Jul. 7, 2007. [Online]. Available: https://www.bibsonomy.org/bibtex/aed0229656ada843d3e3f24e5e5c9eb9

[31] R. Wieringa, N. Maiden, N. Mead, and C. Rolland, "Requirements engineering paper classification and evaluation criteria: A proposal and a discussion," *Requirements Eng.*, vol. 11, no. 1, pp. 102–107, Mar. 2006, doi: 10.1007/s00766-005-0021-6.
[32] T. Dybå and T. Dingsøyr, "Strength of evidence in systematic reviews in software engineering," in *Proc. 2nd ACM-IEEE Int. Symp. Empirical Softw. Eng. Meas.*, Kaiserslautern, Germany, Oct. 2008, pp. 178–187, doi: 10.1145/1414004.1414034.
[33] S. Pérez-Soler, E. Guerra, and J. de Lara, "Creating and migrating chatbots with Conga," in *Proc. IEEE/ACM 43rd Int. Conf. Softw. Eng., Companion (ICSE-Companion)*. Madrid, Spain, May 2021, pp. 37–40, doi: 10.1109/ICSE-Companion52605.2021.00030.
[34] G. Daniel, J. Cabot, L. Deruelle, and M. Derras, "Xatkit: A multimodal low-code chatbot development framework," *IEEE Access*, vol. 8, pp. 15332–15346, 2020, doi: 10.1109/ACCESS.2020.2966919.
[35] W. V. Parada, F. A. Giraldo, and M. I. Londoño R, "MDD based case tool for automatic generation of ChatBot," in *Proc. 45th Latin Amer. Comput. Conf. (CLEI)*, Panama, Panama, Sep. 2019, pp. 1–7, doi: 10.1109/CLEI47609.2019.235059.
[36] R. Baena-Perez, I. Ruiz-Rube, J. M. Dodero, and M. A. Bolivar, "A framework to create conversational agents for the development of video games by end-users," in *Optimization and Learning*, vol. 1173, B. Dorronsoro, P. Ruiz, J. C. de la Torre, D. Urda, and E.-G. Talbi, Eds. Cham, Switzerland: Springer, Feb. 2020, pp. 216–226.
[37] H. Ed-douibi, J. L. C. Izquierdo, G. Daniel, and J. Cabot, "A model-based chatbot generation approach to converse with open data sources," in *Web Engineering*, vol. 12706, M. Brambilla, R. Chbeir, F. Frasincar, and I. Manolescu, Eds. Cham, Switzerland: Springer, May 2021, pp. 440–455.
[38] K. Hogan, A. Baer, and J. Purtilo, "Diplomat: A conversational agent framework for goal-oriented group discussion," in *Contemporary Issues in Group Decision and Negotiation*, vol. 420, D. C. Morais, L. Fang, and M. Horita, Eds. Cham, Switzerland: Springer, Jun. 2021, pp. 143–154.
[39] E. Planas, G. Daniel, M. Brambilla, and J. Cabot, "Towards a model-driven approach for multiexperience AI-based user interfaces," *Softw. Syst. Model.*, vol. 20, no. 4, pp. 997–1009, Aug. 2021, doi: 10.1007/s10270-021-00904-y.
[40] S. Pérez-Soler, G. Daniel, J. Cabot, E. Guerra, and J. de Lara, "Towards automating the synthesis of chatbots for conversational model query," in *Enterprise, Business-Process and Information Systems Modeling*, vol. 387, S. Nurcan, I. Reinhartz-Berger, P. Soffer, and J. Zdravkovic, Eds. Cham, Switzerland: Springer, May 2020, pp. 257–265.
[41] R. Matic, M. Kabiljo, M. Zivkovic, and M. Cabarkapa, "Extensible chatbot architecture using metamodels of natural language understanding," *Electronics*, vol. 10, no. 18, p. 2300, Sep. 2021, doi: 10.3390/electronics10182300.
[42] S. Pérez-Soler, M. González-Jiménez, E. Guerra, and J. Lara, "Towards conversational syntax for domain-specific languages using chatbots," *J. Object Technol.*, vol. 18, no. 2, pp. 1–21, Jul. 2019.
[43] A. T. Neumann, P. de Lange, and R. Klamma, "Collaborative creation and training of social bots in learning communities," in *Proc. IEEE 5th Int. Conf. Collaboration Internet Comput. (CIC)*, Los Angeles, CA, USA, Dec. 2019, pp. 11–19, doi: 10.1109/CIC48465.2019.00011.
[44] J.-P. Sansonnet, D. Leray, and J.-C. Martin, "Architecture of a framework for generic assisting conversational agents," in *Intelligent Virtual Agents*, vol. 4133, J. Gratch, M. Young, R. Aylett, D. Ballin, and P. Olivier, Eds. Berlin, Germany: Springer, 2006, pp. 145–156.
[45] V. Scotti, R. Tedesco, and L. Sbattella, "A modular data-driven architecture for empathetic conversational agents," in *Proc. IEEE Int. Conf. Big Data Smart Comput. (BigComp)*, Jeju Island, South Korea, Jan. 2021, pp. 365–368, doi: 10.1109/BigComp51126.2021.00080.
[46] R. Mahmood, A. Joshi, A. Lele, and J. Pennington, "Dynamic natural language user interfaces using microservices," in *Proc. CEUR Workshop*, Cagliari, Italy, 2020, pp. 1–11.
[47] A. Khalil, F. H. Leiva, A. Shonibare, E. M. Arsenault, L. Turner, S. Khalifa, L. Tam-Seto, B. Linden, V. Wood, H. Stuart, J. Nolan, and C. McDowell, "Model-driven chats: Enabling chatbot development for non-technical domain experts through chat flow visualization and auto-generation," in *Advances in Information and Communication*, vol. 1363, K. Arai, Ed. Cham, Switzerland: Springer, Apr. 2021, pp. 1036–1050.
[48] L. Harry, "Huey: Intelligent agents for natural human to machine communication," M.S. dissertation, Division Continuing Educ., Harvard Univ., USA, 2021.
[49] S. Pérez-Soler, E. Guerra, and J. de Lara, "Model-driven chatbot development," in *Conceptual Modeling*, vol. 12400, G. Dobbie, U. Frank, G. Kappel, S. W. Liddle, and H. C. Mayr, Eds. Cham, Switzerland: Springer, Oct. 2020, pp. 207–222.
[50] G. Daniel, J. Cabot, L. Deruelle, and M. Derras, "Multi-platform chatbot modeling and deployment with the Jarvis framework," in *Advanced Information Systems Engineering*, vol. 11483, P. Giorgini and B. Weber, Eds. Cham, Switzerland: Springer, May 2019, pp. 177–193.
[51] S. Pérez-Soler, E. Guerra, and J. de Lara, "Flexible modelling using conversational agents," in *Proc. ACM/IEEE 22nd Int. Conf. Model Driven Eng. Lang. Syst. Companion (MODELS-C)*. Munich, Germany, Nov. 2019, pp. 478–482, doi: 10.1109/MODELS-C.2019.00076.

ÁNGEL ANTONIO MARTÍNEZ-GÁRATE received the bachelor's degree in computer system engineering from Universidad Autnoma de Sinaloa (UAS), in 2019, where he is currently pursuing the master's degree in information science. In recent years, he has focused on the research and development of conversational agents and the automation of software development.

JOSÉ ALFONSO AGUILAR-CALDERÓN received the M.Sc. degree from Universidad Autnoma de Sinaloa, Mexico, in 2008, and the Ph.D. degree from the University of Alicante, Spain, in 2011. Currently, he is a full-time Professor (Research) with Facultad de Informtica Mazatln and a Master Professor (Posgrado en Ciencias de la Informacin) with Universidad Autnoma de Sinaloa. He is a member of the AMEXCOMP Technology Society and the National System of Researcher's with Consejo Nacional de Humanidades, Ciencia y Tecnología (CONAHCYT), Mico. His research interests include web engineering (WE), requirements engineering (RE), model-driven development (MDD), usability and accessibility in web, and software defined networks (SDN).

CAROLINA TRIPP-BARBA received the degree from the Faculty of Informatics, Universidad Autnoma de Sinaloa, Mexico, in 2006, and the M.Sc. and Ph.D. degrees from Universitat Politcnica de Catalunya (UPC), Barcelona, Spain, in 2009 and 2013, respectively. Currently, she is a full-time Research Professor with Facultad de Informtica Mazatln, Universidad Autnoma de Sinaloa. Her research interests include vehicular ad hoc networks (VANETs), the design of routing protocols, smart cities, and software defined networks (SDN). She is a member of the AMEXCOMP Technology Society and a member of the National System of Researcher's with Consejo Nacional de Humanidades, Ciencia y Tecnología (CONAHCYT), Mexico.

ANÍBAL ZALDÍVAR-COLADO received the degree from the Faculty of Informatics, University of Sinaloa, Mexico, in 2000, and the M.Sc. and Ph.D. degrees from the University of Durango (UAD), Mexico, in 2000 and 2011, respectively. Currently, he is a full-time Professor (Research) with Facultad de Informtica Mazatl, Universidad Autnoma de Sinaloa, Mexico. His research interests include applied mathematical modeling, systems simulation, and educational technology. He is a member of the National System of Researcher's with Consejo Nacional de Humanidades, Ciencia y Tecnología (CONAHCYT), Mexico.