VOSA: A Reusable and Reconfigurable Voice Operated Support Assistant Chatbot Platform

Joseph Willrich Lutalo* and Tonny J. Oyana*

*Department of Networks, Makerere University, Kampala, Uganda

Abstract

Current research shows that offering customer support in any form is a guaranteed means to boost and sustain business growth. Modern support services are steadily embracing automation to improve effectiveness, support scalability, and reduce costs, with the most promising approaches leveraging artificial assistants in the form of chatbots and interactive support services. In this paper, we employ the Design Science Research method to explore and then practically implement an original, reusable, re-configurable chatbot platform for designing and delivering autonomous product and customer support services leveraging voice interactions. Further, additional focus was placed on leveraging a scan-to-know information access model, and we especially considered users operating on mobile computers such as smartphones, with active connectivity. The implemented chatbot platform was explored and evaluated from the context of two practical cases; banking and hotel customer support. Important results obtained included; an original architecture of an autonomous, natural-interface, reusable and reconfigurable personal assistant chatbot platform; a working proof-of-concept reference implementation of the VOSAC, currently published and accessible via the Google Play Store; a simple and efficacious method of encoding and expressing knowledge in the form of QAKBs, useful in extending/reconfiguring QAKB-compliant personal assistants such as the VOSAC, and finally, the realization of a Scan-to-Know information retrieval method for making access to autonomous support services more scalable and cheaper to implement.

Keywords: conversational agents, human-computer interaction, question-answering system, chatbots, knowledgebases, digital voice assistants.

David C. Wyld et al. (Eds): SE, SAIM, SIPM, CoNeCo, ICITE, ACSIT, CMIT, FCST, SNLP – 2024
pp. 19-38, 2024. CS & IT - CSCP 2024 DOI: 10.5121/csit.2024.140803

1 Introduction

1.1 Background

Not all users of a product, or consumers of a service, immediately understand fully or exploit fully, the product or service at their disposal, even with extensive experience or familiarity with the concerned product or service [1].

This observation is based on the fact that not all the necessary information about a product or service is immediately accessible to the user of a product, or consumer of a service when it is needed. This is referred to as an information deficiency problem (IDP) in this research. An IDP could arise during emergencies, critical or urgent business operations and services.

Such information deficiencies could traditionally be remedied by having the user of a product, or consumer of a service consult a domain expert, a product-manual, a customer-support agent or any available documentation on the product or service [2]. Conventionally, especially for products, such consultation is called After-Sales-Service (ASS), and for services, typically called Customer Support Service (CSS).

In the absence of an ASS or CSS human agent, the most natural alternative is the consultation of printed or written product or service documentation. Such a need might arise in a situation where a user engages a product away from its manufacturer or product support team. This documentation then, traditionally, would be in the form of written manuals accessible as physical booklets or digital files on electronic media - online or offline.

However, several challenges exist with regards to printed product or service documentation, among which the burden of locating, searching and reading relevant information from them put up a barrier for incapable or busy users. Without special approaches to accessing and presenting such information, we argue that IDPs are inevitable, and in the domains of business especially, such an important matter that needs addressing [1]. This situation mostly makes sense, when it is assumed that the consumers of support services are typically human, and thus warrant some careful considerations when such services are to be designed for the masses.

In this research, we are especially concerned with whether there is a reusable technology platform that can be leveraged to implement authoritative autonomous customer and product support services leveraging voice-interactions in the form of a chatbot. A reusable platform, because we don't wish to address the autonomous support service delivery problem for a specific or limited domain, but rather, to realize a solution that can be leveraged for any kind of service or product, with minimal or no fundamental change to the underlying technology being leveraged. And then finally, we wish to see

that the manner in which these autonomous support services are delivered to their human end-users, is humanized [3];ergonomic, economic and especially natural. This is the reason we focus on leveraging simple natural interfaces such as voice-based interactions via a smartphone-based chat application, such that the realized platform can serve the majority of end-users with or without technical dexterity or experience manipulating or interacting with sophisticated support systems. Essentially, we wish to realize a voice-enabled reconfigurable support service chatbot platform.

1.2 A Hypothetical Support Assistant Scenario

To put the IDP concept in perspective, consider the following hypothetical case:

Consider a human-being, operating a remote research office on a faraway planet or perhaps, on a moon; an astronaut. Consider that access to any other human while on this job is not immediately possible, and that any assistance or delegation required by the astronaut is via the aid of non-human automatons - most likely, intelligent machines and software.

Next, conceive of a situation in which the astronaut wishes to share vital information, with a colleague or affiliate team somewhere else, significantly removed from their station - on a mother ship in orbit perhaps, or back on their home planet. Further, consider that the only available means of such long-distance communication is via the operation of a specially designed Interplanetary Information Transmission Antenna (IITA), that is part of their mission-support-kit.

However, if it were the case, that upon attempting to engage and use the IITA, it instead malfunctions. The astronaut might have no better option than to troubleshoot; establish how to remedy or by-pass problem. For such troubleshooting, we could conceive of a specially crafted mission-support assistant technology that the researcher could engage. They would use it to obtain knowledge or tips on how-to resolve the problem.

1.3 The Proposed Idea and Solution

This support assistant could be in the form of an Information-Reference and Lookup Booklet (IRLB), or as is proposed in this paper, as an automaton either embodied as a standalone physical robot with voice-interaction capabilities or as a Voice Operated Support Assistant (VOSA) running on the astronaut's mobile computer.

In the case where the support assistant is a VOSA, the astronaut might go about their IITA troubleshooting operation or IDP solution thus;

1. Using voice, they inform the VOSA about the observed malfunction in the IITA.
2. The VOSA analyzes the input utterance, extracts relevant cues about an information lookup request, and using this, together with a knowledge model it has access to, finally identifies which particular information is needed by the user to solve the problem.
3. The VOSA then compiles and reports a solution back to them if found
4. If there is still need, a further query might be posed to the VOSA so as to further clarify or resolve the problem.

2 Related Works

2.1 Introduction

This chapter treats of the matter of existing literature, research and formal ideas relating to the VOSA concept and technology as expressed in this paper.

2.2 Related Assistant Technologies in Practice

Superior after-sales service (ASS) can boost both first-time and repeat sales, thereby increasing marketshare [4]. This definitely means, whether it is big corporations, lean startups or cottage industries, the relevance of offering customer support in some form can't be ignored if business growth and profitability are desired. However, it is not only for business purposes that support services might be needed. Governments and even Non-Governmental Organizations which conduct or extend services to the public also have a serious need for support services either as part of service continuity, sensitization efforts or as information gathering mechanisms that are part of bigger undertakings [5] [6].

Providing ASS as part of product support, or customer service support (CSS) in the case of services, is an activity that could leverage automation for the sake of business sustainability, but also as a means of ensuring scalability and timeliness of service delivery. It has been argued that investing in chatbots as opposed to investment in human-backed CSS delivers a more effective solution [7]. Chatbots, which are the way in which most virtual assistants are implemented nowadays, come in various forms; some support text-only interactions, others will support both text and voice inputs or commands, and others only handle voice interaction. However, further advances

in this area have seen the arrival of chatbots with immersive and natural interfaces beyond just voice and text. One very promising and disruptive chatbot technology in this space is Replika [8], launched in 2017, and said to have as many as 10 million active users as of 2022 [9]. This new generation of chatbots offer augemented reality and virtual reality interfaces that allow the artificial assistant or companion to merge more naturally with the human user's environment, and this further enhances the conversations and emotional, psychological impact the artificial agent has on the person. In the case of support services, definitely, these advances make automation-driven customer support ever closer to or as realistic as traditional human-driven support - with even face-to-face interactivity between human and artificial intelligence becoming more tolerable, emotive and to some observers, perhaps even spiritual [9]. Other promising chatbot technologies important to be aware of include Kiku (also known as "Mitsuku"), Zeve and others [10]. An important observation to make about most of these emerging chatbot technologies though, is that they have focused more on filling the "artificial companion" role, with most aiming at realizing an artificial friend for their human users, and less on filling critical business roles such as customer or product support, which is the focus of our research interests.

That virtual assistants have a serious role to play in modern commerce and social life in general can't be taken for granted anymore. Though several applications exist that have attempted to introduce use of virtual assistants in business contexts, we shall explore a few that are representative of the general trend. Of major interest to us shall be those assistant technologies that have endeavored to leverage natural interfaces employing voice.

It must be understood that voice interactivity is largely underpinned by text-driven interactions such as traditional chatbots leverage - typically, the voice interactivity manifests in the form of text-to-speech (TTS) and speech-to-text (STT) mechanisms [11]. For example, embedding virtual assistants in e-commerce websites, or any website for that matter, in the form of embodied conversation entities (ECA)[12] is one way the concept of Voice Operated Support Assistants (VOSA) could be realized; merely adding a layer of STT and TTS would then make such ECA voice-operable. To illustrate these points, consider popular virtual assistants such as Cortana, Alexa, Siri and Google Assistant that all support voice interaction as part of their core user experience [13].

However, not all virtual assistants need leverage voice. The case of Adobot, a CSS chatbot for Adobe's Photoshop is based on text-based questions from the user, and text output in the form of answers based on a Question-Answer Knowledge Base (QAKB) for the software package [14]. However, where possible, it can be argued that the addition of voice-interaction

increases user engagement with the virtual assistant [15] [16], makes user-support more humane, and also exposes virtual support services to users that can't either type or those that are blind but can talk and hear [16]. Definitely, this later case is also a boost to designing with accessibility in mind - an aspect of modern product and service design that can't be taken for granted anymore [17].

Further, it is of importance to consider whether these virtual assistants need be implemented directly within, alongside or independently of the product or service they offer support for. For example, in the case of software products or digital services, some virtual assistants are being integrated directly into the core of associated operating systems; the case of Cortana for Windows, Siri for Apple and Google Assistant for the case of Android are common examples [18]. It might also be the case that a product's virtual support assistant is implemented independently of the product, and can be used away from it. This is the case with one documented Question-Answer chatbot designed to support users of Adobe's Photoshop application [14]. In the more typical cases though, virtual assistants have been designed to integrate directly with the operating system where supported products and services run, and in these cases, the virtual assistant becomes a feature of the operating system and not a particular product or service.

Where such virtual assistants are part of an operating system, it might be the case that they offer support across several products and services at once; for example, for Android users, the Google Assistant can be leveraged to find and dial contacts, perform internet searches via Google's search engine, but could also be used to launch non-Google-related apps the user has installed on their phone. In a more general sense, it is possible to design virtual assistants that support products across multiple vendors - implying that building a single, generic virtual assistant solution - especially in the manner of a platform that others can leverage to build support services on-top of, could have impact across several businesses, domains, product and service lines, especially where configurability or adaptability is designed into the assistant technology. A documented case is that of Window's Cortana, which can also support Google's Android products, or the case of Amazon's Alexa, that supports both Amazon's technologies such as Resound and Fire, but also Android and iOS PDAs [13]. The idea of designing artificial support services or virtual assistants as a service or platform that can be leveraged by several entities or for arbitrary ends has already been explored by some recent research work in the chatbot-platforms domain[19].

Next, we must consider the issue of how these virtual assistants obtain or generate their information in response to user queries or interactions. First, it must be noted that these virtual assistants are as intelligent as the knowledge

Chatbot/ Assistant Technology	Accessibility	Supported Interactions	Conversation Mode	Knowledge Modelling	Target Scenarios	Interaction Link
Replika	Web, Mobile, VR	Textual, Vocal, Visual	Multi-turn	Large Language Model; GPT-2[26]	Psychological Support, Artificial Companion, IDP Solver	https://replika.ai/
Kiku ("Mitsuku")	Web	Textual, Vocal, Visual	Single-Turn	Knowledge Base[28]	Artificial Companion, IDP Solver	https://chat.kuki.ai/chat
Cortana	Desktop	Textual, Vocal, Visual	Single-Turn	Neural Network[29]	Customer Support, Artificial Companion, IDP Solver	https://apps.microsoft.com/detail/9nffx4szz23l
SIRI	Mobile	Textual, Vocal	Multi-turn	Knowledge Graph[30]	Artificial Companion, Product Support, IDP Solver	https://www.apple.com/siri/
Google Assistant	Mobile	Textual, Vocal, Visual	Multi-turn	Deep Neural Networks (DNN)[27]	Artificial Companion, Product Support, IDP Solver	https://assistant.google.com/
VOSAC	Mobile	Textual, Vocal	Single-Turn	Question-Answer Knowledge Base (QA corpus)[25]	Artificial Companion, General Support, finite-IDP Solver	https://bit.ly/4vosac

Table 1: Assistant Chatbot Technologies Summarized

they have access to [20]. For example, this means that unless a knowledge base about a product or service exists that the virtual assistant has access to, it might not make sense or might be ineffective to use the assistant to offer support related to that domain. And even where such a knowledge base for a particular service or product exists, errors, inconsistencies or omissions in the knowledge itself might impact the relevance and reliability of the associated assistant or its virtual support services.

Concerning methods for compiling or building knowledge that powers a virtual assistant, the literature talks of two basic categories of knowledge bases; manually created knowledge bases and automatically generated knowledge bases [20]. The first category might include rule-based knowledge systems in which the behavior and interactivity of the virtual assistant is based on logic and knowledge hardcoded into the system, or might be of the kind where the assistant references explicitly expressed question and answers as are found in a Frequently-Asked-Questions manual or on web-powered information catalogs as are found on Question-Answer websites [21] [22]. In the case of auto-generated knowledge-bases, trends include using of web-scraping technologies to generate large corpora of information based on disparate information sources from across the internet, in online-accessible books as well as human interactions on social media as an example[23] [24]. In this later case, also modern methods include building Knowledge Bases from Question-Answer corpora - which have been referred to as KBQA[25], use of Neural Networks trained on large texts and online content such as with Large Language Models such as GPT[26] or other machine learning methods such as with Google's use of Deep Neural Networks for the Google Assistant[27].

In general, we summarize the key aspects of the virtual assistant technology field covered in this chapter, with **Table 1**, which attempts to contrast several popular assistant technologies in the form of chatbots against

each other, with major focus on; how these technologies are accessed; which kinds of interaction interfaces they support; how their knowledge is modelled; whether or not they support single-turn or multi-turn conversations [31], as well as what kinds of major use-case categories they fit in.

3 The Proposed VOSA System

3.1 Overview of the System

Based on our understanding of the requirements for the VOSA, as well as what we knew of existing, related virtual support technologies, the design and implementation of a real VOSA takes the form described in the rest of this chapter.

First of all, the VOSA system basically consists of two major components - a subsystem for creating and managing knowledge models also known as "knowledge bases", and then a client subsystem for querying those models so as to offer real-time answers to users, in response to queries they pose about a topic in the models. The client is called a VOSA Client or simply "VOSAC".

3.2 VOSA Ecosystem Architecture

At its simplest, the VOSA is a system for information storage, querying and retrieval, with the key distinction that both querying and presentation of information is conducted with voice as the primary interface, and that there is no strict or formal query language employed by the user except natural language such as English. Essentially, the VOSA offers a Natural User Interface (NUI) [32] with which one can access information in one or more knowledge bases.

The information storage in a VOSA ecosystem could dwell directly on the clients - such as when a VOSAC holds cached or pre-loaded knowledge bases (KBs), though, because the system is expected to be readily extensible and also user-configurable in terms of which KBs it queries or references in real-life usage - thus adaptable, the use of online or over-the-network KBs is a key aspect of the VOSA architecture, so that some form of centralized KB access and management - creation, dispatch and update of standardized knowledge bases, is possible without having to directly manipulate the ones held on VOSACs. Essentially, this calls for some sort of subscribe-pull or subscribe-push client-KB-update mechanism in the VOSA ecosystem, and as shall be seen when we explore and discuss the reference

implementation of a VOSAC in section 3.4, the clients essentially subscribe to a Question-Answer-Knowledgebase (QAKB) via a QAKB Quick-Response Code (QAKB-QRCODE), and then can later pull any published updates associated with the QAKB from a networked server holding the QAKB, also known as a QAKB Hosting Server (QAKBHS).

In a typical VOSA ecosystem, it is expected that one or a few QAKBHS serve an arbitrary number of clients such as VOSACs, QAKB-Importers, QAKB Processors and such related technologies. The QAKBHS can be seen as a web service in the sense that it helps to publish, advertise and also serve QAKBs in a managed client-server kind of architecture.

Figure 1 expresses this architecture diagrammatically, and the two major subsystems in the ecosystem are hereafter discussed.

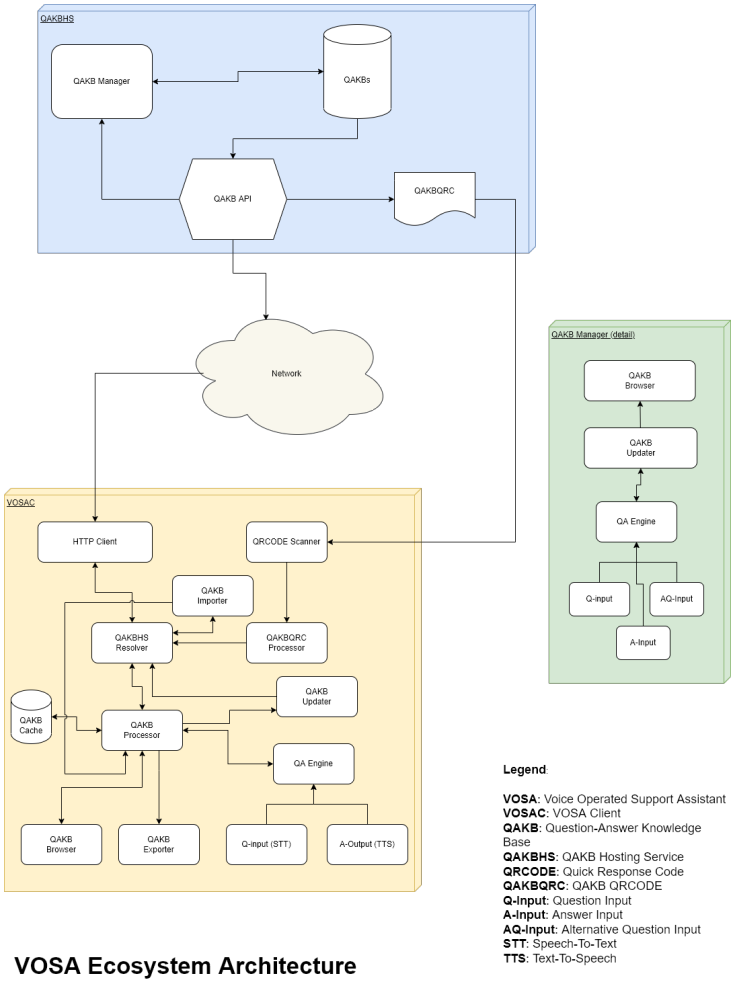

Figure 1: VOSA Ecosystem Architecture

3.3 QAKBHS: The VOSA Management Server

To facilitate centralized administration and management of the key artifacts in a VOSA ecosystem - which are the knowledgebases and the QAKB-QRCODEs used to access them, the QAKBHS exists. In essence, this server offers QAKB authors and administrators an interface that allows a knowledge model for a specific topic to be designed, and which can then be published in a streamlined way. Because the QAKBHS handles the QAKB editing and management facilities in a VOSA ecosystem, this subsystem then, is designed to be accessible to and only used by the content-managers and not the customers or users for whom the support services are being designed. The QAKBHS server then, has a public interface accessible via a standard web browser (which in Figure 1 is labelled "QAKBHS Public Interface"), that offers facilities for browsing one or more QAKBs, adding new ones, deleting or updating existing QAKBs.

The actual QAKB is designed as a structure that holds the name and type of the knowledge model, and then a set of QAKB entries where each entry consists of a primary question, a single answer, and zero, one or more alternative questions - also known as "secondary questions", that relate to the primary question or are alternative ways of phrasing the primary question. An example of a QAKB structure conforming to this specification is shown in Appendix A.1. The preview and editing of these QAKB entries is what the "QA Engine" in Figure 1 is for; Q-Input maps to the primary question, AQ-Input to the alternative questions and then A-Input to the answer associated with a QAKB entry.

Besides offering the QAKB management interface to content administrators, the QAKB Manager also interacts with a database to read and store QAKB data in response to requests either via the management interface or via an HTTP API that VOSA clients use to fetch published knowledge models. We shall refer to this API as the QAKB API.

As shall be seen in section 3.4 on the VOSAC, the only way that a client first discovers a QAKB is via the scanning of a published QAKB-QRCODE. The only thing special about these QRCODES is that they encode the basic information required by the clients to fetch and present a published QAKB: the name of a QAKB; the type of the QAKB (standard options include "Product", "Service" and "Topic"); the fully qualified URL pointing to a network resource via which the full contents of the QAKB can be fetched via HTTP(S) from a QAKBHS.

It is the purpose of the QAKBHS to encode this information into a QR-CODE so that all a client needs do is scan the QRCODE and then learn about or access the published QAKB. In the reference implementation of

the VOSA, this QAKB header information is encoded as a JSON object before being encoded visually as a QRCODE. An example of the standard way in which such a QAKB header is encoded is shown in Listing 1, and the associated QAKB-QRCODE is shown in Figure 2.

Of course, once generated - typically from the QAKBHS, a QAKB-QRCODE can then be printed in any manner - say on a wall, t-shirt, business card, product package, restaurant menu, on a website, on social-media pages, etc, independently of the QAKBHS, so that anyone with a QRCODE-scanner can read its contained QAKB header information. However, under ordinary or standard circumstances, a QAKB-QRCODE-aware client such as a VOSAC is used to read and act upon the code and its contents, as by standard VOSA protocols. In the next section then, we introduce and explore the VOSAC, and how such interactions as the ones described above actually happen.

Listing 1: Menuza Bytes CSS QAKB-QRCODE contents - an example of QAKB Header Info

```
{
  ``name'': ``MENUZA INTELLIGENCE'',
  ``kind'': ``SERVICE'',
  ``url'': ``https://qakbhs.chwezi.tech/api/qakb/2/''
}
```

Figure 2: VOSAC-compatible Standard QAKB-QRCODE Example for accessing Customer Support Services at a Hotel

3.4 VOSAC: The VOSA Client

To interact with a VOSA ecosystem from an end-user context, special software that is capable of voice-mediated information querying, retrieval and presentation is required. In the reference implementation of a VOSA system, this role is played by a special mobile application that is called the "VOSAC".

The internal architecture of a standard VOSAC is presented in Figure 1,

and as can be seen, consists of a non-trivial composition of several specialized components as we shall now discuss.

3.5 Key VOSAC Components And Their Relation To The Typical VOSAC-User Interaction Process

Consider a scenario of a user interacting with a VOSAC at a restaurant, to quickly learn about the active menu options on a particular day. For this scenario, the essential steps of the process are as such:

1. User has an information deficiency problem (IDP): for example, they don't know what the active menu is at the restaurant on a particular day.

2. Restaurant offers automated, self-help customer support services (CSS) via the standard VOSAC for any interested customers or persons.

3. The associated business has published their CSS knowledge base via a QAKBHS accessible via the public internet and thus have a QAKB-QRCODE for their customers to interact with.

4. The CSS QAKB-QRCODE has been printed and is displayed in customer-accessible areas such as on the restaurant website or on physical materials at the business premises or in their neighborhood - e.g on notice-boards, on on-wall TV displays, in brochures, or on business cards.

5. The customer has been directed to where they can obtain a VOSAC - for example, a download link to the VOSAC is on the restaurant's website or that it is assumed the customers already have or know where and how to obtain/install a VOSAC - say, via public app repositories and stores, or scan-to-download QRCODE messages.

6. The customer then, starts their VOSAC, and using which, they scan the displayed business CSS QAKB-QRCODE so as to access customer support information in relation their IDP.

7. To read a displayed QAKB-QRCODE, the VOSAC uses a standard QRCODE scanner, that knows to input a QRCODE image, and output its contents as text. Once the text in the QAKB-QRCODE (QAKBQRC) is obtained, this is then forwarded to the QAKBQRC Processor for interpretation.

8. The QAKBQRC Processor then extracts the essential information, such as the type and location of the actual QAKB resource over the network, and forwards this to the QAKBHS Resolver.

9. The QAKBHS Resolver uses an HTTP Client to query and fetch the QAKB via the network, from the associated QAKBHS, and then forwards the returned QAKB data payload to the QAKB Importer.

10. The QAKB Importer then takes that QAKB data and in coordination with the QAKB Processor, writes it to a local, on-client database called the QAKB Cache.

11. Now that the VOSAC knows about the CSS knowledge base, it can present the user with an interface where the user can activate, deactivate, delete or update the cached QAKB at will. This functionality is what the QAKB Browser is for. Additionally, for alternative, offline-compatible sharing and loading of previously cached QAKBs, the VOSAC would offer a QAKB Exporter facility that takes all actively cached QAKBs on the VOSAC and locally saves them in a standard format in a file on the client device. The QAKB Importer then, can later be used to read such locally stored QAKB collections and thus initiate or restore a VOSAC's access to one or more knowledge bases in a single import operation as opposed to scanning multiple QAKB-QRCODEs.

12. To start Question-Answer (QA) interaction with the VOSAC, the user needs to first activate one or more QAKBs via the QAKB Browser, then launch the VOSAC's QA interface that is powered by the QA Engine. In the reference implementation of the VOSAC, the QA Engine presents a NUI in in the form of a voice and visual feedback interface (the A-Output component) that is used to echo to the user both the question and answer to the question, and then a voice-only input facility (Q-Input), that uses the client's microphone to capture the user's questions or utterances, and then encodes them into text that the QA Engine understands, by leveraging Speech-to-Text (STT) facilities of the client device. The A-Output similarly employs the device's Text-to-Speech (TTS)facility so as to take textual outputs from the QA Engine and speak them out to the user via natural language.

13. The user then – possibly a customer in this case, posses their question to the VOSAC: for example, they speak out loud the question: "What is on the menu today?"

14. The Q-Input captures their utterance, forwards it to the QA-Engine as text, the QA Engine interacts with the QAKB Processor to then locate or generate the most suitable answer based on the active QAKBs in the QAKB Cache, and once this answer is computed as text, the A-Output speaks it out to the user as natural language: for example, the VOSAC answers back "On the Menu today are snacks, sandwiches, a luncheon, tea and coffee."

15. If the answer from the VOSAC addresses the user's IDP, the VOSA-session might stop there (say the user closes the VOSAC application), otherwise, the user might either pose the same question *differently* or pose a different question altogether, to attempt to get a different or better answer, in which case the same process repeats as from step 1, but with some intermediate steps such as #2-12 being omitted from the process on subsequent interactions - such as when the required QAKB has already been fetched and is up-to-date; in which case steps #2-11 can be omitted. However, when the active QAKB list needs to be tweaked - such as when the IDP to be solved relates to a different or several QAKBs already cached, then step #12 might also need to be repeated, before the user can proceed to step #13.

4 Discussion and Future Work

Based on lab and field explorations of VOSA technology, several challenges were encountered - some related to the actual design of the VOSA, others to the process employed in interacting with or applying VOSA in real-life scenarios. We shall briefly highlight these, together with potential solutions, as well as opportunities for advancing the VOSA and/or related technologies.

First of all, the VOSAC can import QAKBs from any network-accessible source. However, there is no client-side ability to export or import QAKBs from local storage. First, on-device servers and manually constructed QAKB-QRCODEs may be able to overcome this limitation. Second, in circumstances when a QAKBHS is not practical, employing JSON-hosting services that allow a well-formatted QAKB definition to be accessed by URL as raw text or JSON would solve this problem.

When numerous user-activated QAKBs match a single question presented to the VOSAC, the VOSAC may return any of the potential solutions. One practical remedy for this, is to activate only one, or a limited number of QAKBs on the VOSAC during a QA session, thus giving the user control over which solution space the VOSAC utilises during the session. If a user-

posed question does not match any QA-entry in a VOSAC's active QAKBs, the VOSAC will respond with an echo of the question or a default response for certain hard-coded QA-scenarios (e.g., "How are you?", "What is your name?", "What time is it?"). A more effective remedy might be to fallback to an autonomous general intelligence service, such as the Google Search API - which can generally offer useful responses to arbitrary user-posed queries when no relevant user-configured knowledge model is available on the VOSAC, or perhaps falling-back to some well-tuned alternative knowledge model such as the popular GPT or another custom Large Language Model (LLM).

Currently, there is no real language-agnostic way to define QAKBs except English, when targeting the reference implementation of a VOSAC. This issue is intimately related to the way in which the VOSAC's QA-voice interface works; the inputs are presumed to be English by default, and so the Speech-To-Text conversion employed applies an English grammar model regardless of the language in which the user presents their queries. Furthermore, even if the content administrators publish a QAKB in a non-English language, such as French or Swahili, at the point of solving the user-query-QA-entry match relative to the active QAKBs, the reference implementation of a VOSAC expects that the QAKB is composed of English questions and answers, and so English-tuned Natural Language Processing (NLP) can be applied. Thus, legitimate user-query-QA-entry matches might be missed even where they should exist based on the active QAKBs. To make the VOSAC and its NLP mechanism language-agnostic at runtime, one potential approach is to develop some sort of language-configuration or language-setting mechanism at the VOSAC level, or to construct QAKBs with a language setting in their specification useful in detecting and resolving these anomalies.

The current VOSAC implementation can't accept typed questions even though it has a facility for echoing a user's parsed query and the resultant resolved answer as visually-perceptible text. This was an ergonomic design choice; to encourage VOSA usage to predominantly or exclusively be a hands-free user experience especially during the QA-sessions, and to lower the engagement barrier for illiterate and semi-literate users that can often readily converse or listen to English (and or other natural languages), but have difficulty typing or reading text. The VOSA technology is meant to offer support services in the widest usage contexts, which might also include children and people with some incapacitation.

5 Conclusion

Sufficient customer and product support can boost the expansion and profitability of a business. The objective of this project, which employed the Design Science Research methodology, was to theoretically investigate and subsequently implement a technique for providing users with instantaneous, autonomous support services through voice interactions with a virtual assistant on internet-connected mobile computers such as smartphones. A reference implementation of the standalone, natural-interface, reusable, and reconfigurable VOSA chatbot is now available through the Google Play Store. Other significant contributions made by this research include a straightforward and effective method of encoding and expressing knowledge in the form of QAKBs, which can then be used to tune, re-purpose, or reconfigure QAKB-compliant chatbots such as the VOSAC. Finally, a scan-to-know method - for accessing and retrieving information, was realized. According to our evaluations, using client-side knowledge model caches, QRCODEs, and APIs improved the accessibility, effectiveness, and scalability of product and customer support services noticeably. This technology requires further study and development in aspects such as the reduction of false positives during user-query resolution, as well as enabling more intelligent, naturalistic multi-turn conversations, such as those now possible in some contemporary virtual assistants and chatbots.

Appendices

A.1 Standard QAKB Data Format via QAKBHS-QAKB API

Originally retrieved via the online reference implementation of a QAKBHS [1] , this QAKB Data Payload describes a knowledge model, expressed as a QAKB, that is used in customer support and after-sales services for the business calculator 256 BC popular in Uganda.

Listing 2: Example QAKB Specification: 256 BC QAKB

```
{
  "name": "256 BC QAKB",
  "kind": "PRODUCT",
  "qa": [
    {
      "id": 1,
      "q": "How do I start the app?",
```

[1] https://qa.chwezi.tech/api/qakb/1/

```
      "a": "To start the 256 BC calculator app, navigate to your phone's list of apps,
           and click on the app with the name \"256 BC\"",
      "aq": []
    },
    {
      "id": 2,
      "q": "How do clear the screen?",
      "a": "To clear the screen after calculating or entering some input on the
           calculator, click on the button labelled \"C-L-R\"",
      "aq": []
    },
    {
      "id": 3,
      "q": "How do I convert between US Dollars and Uganda Shillings?",
      "a": "To convert between US Dollars and Uganda Shillings, use one of the two
           buttons in the bottom-right corner of the calculator interface. To convert
           from Shillings to Dollars, click \"UGX~$\", and from Dollars to Shillings,
           click \"$~UGX\".",
      "aq": [
        "How to perform currency conversion?",
        "Forex exchange method on 256 BC?",
        "Foreign Currency calculations"
      ]
    }
  ]
}
```


References

[1] Colin Armistead. After sales support strategy. 1990.

[2] Mathangi Sri. Nlp in customer service. In *Practical Natural Language Processing with Python*, pages 13–63. Springer, 2021.

[3] Scott Schanke, Gordon Burtch, and Gautam Ray. Estimating the impact of "humanizing" customer service chatbots. *Information Systems Research*, 32(3):736–751, 2021.

[4] Morris A Cohen and Hau L Lee. Out of touch with customer needs? spare parts and after sales service. *MIT Sloan Management Review*, 31(2):55, 1990.

[5] Anna Ya Ni and Alfred Tat-Kei Ho. Challenges in e-government development: Lessons from two information kiosk projects. *Gov Inform Q*, 22(1):58–74, 2005.

[6] Jane E Fountain. Paradoxes of public sector customer service. *Governance*, 14(1):55–73, 2001.

[7] Dinh Thang Le, Thanh Thoa Pham Thi, Cuong Pham-Nguyen, et al. Towards a context-aware knowledge model for smart service systems. In *International Conference on Computational Collective Intelligence*, pages 767–778. Springer, 2020.

[8] Martino Mensio, Giuseppe Rizzo, and Maurizio Morisio. The rise of emotion-aware conversational agents: threats in digital emotions. In *Companion Proceedings of the The Web Conference 2018*, pages 1541–1544, 2018.

[9] Tracy J Trothen. Replika: Spiritual enhancement technology? *Religions*, 13(4):275, 2022.

[10] Giuliana Guazzaroni. Virtual and augmented reality in art during the pandemic. In *Extended Reality Usage During COVID 19 Pandemic*, pages 89–93. Springer, 2022.

[11] Naim Zierau, Christian Hildebrand, Anouk Bergner, Francesc Busquet, Anuschka Schmitt, and Jan Marco Leimeister. Voice bots on the frontline: Voice-based interfaces enhance flow-like consumer experiences & boost service outcomes. *Journal of the Academy of Marketing Science*, pages 1–20, 2022.

[12] Karolina Kuligowska and Mirosława Lasek. Virtual assistants support customer relations and business processes. In *The 10th International Conference on Information Management, Gdańsk*, 2011.

[13] Amrita S Tulshan and Sudhir Namdeorao Dhage. Survey on virtual assistant: Google assistant, siri, cortana, alexa. In *International symposium on signal processing and intelligent recognition systems*, pages 190–201. Springer, 2018.

[14] Luong Tran Hien, Ly Tran Thi Ly, Cuong Pham-Nguyen, Thang Le Dinh, Hong Tiet Gia, et al. Towards chatbot-based interactive what-and how-question answering systems: the adobot approach. In *2020 RIVF International Conference on Computing and Communication Technologies (RIVF)*, pages 1–3. IEEE, 2020.

[15] Alisha Pradhan, Kanika Mehta, and Leah Findlater. " accessibility came by accident" use of voice-controlled intelligent personal assistants by people with disabilities. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems*, pages 1–13, 2018.

[16] Katherine O'Brien, Anna Liggett, Vanessa Ramirez-Zohfeld, Priya Sunkara, and Lee A Lindquist. Voice-controlled intelligent personal assistants to support aging in place. *Journal of the American Geriatrics Society*, 68(1):176–179, 2020.

[17] Alisha Pradhan, Amanda Lazar, and Leah Findlater. Use of intelligent voice assistants by older adults with low technology use. *ACM Transactions on Computer-Human Interaction (TOCHI)*, 27(4):1–27, 2020.

[18] Martin Boyanov, Ivan Koychev, Preslav Nakov, Alessandro Moschitti, and Giovanni Da San Martino. Building chatbots from forum data: Model selection using question answering metrics. *arXiv preprint arXiv:1710.00689*, 2017.

[19] Godson Michael D'silva, Sanket Thakare, Sharddha More, and Jeril Kuriakose. Real world smart chatbot for customer care using a software as a service (saas) architecture. In *2017 International Conference on I-SMAC (IoT in Social, Mobile, Analytics and Cloud)(I-SMAC)*, pages 658–664. IEEE, 2017.

[20] Jack Cahn. Chatbot: Architecture, design, & development. *University of Pennsylvania School of Engineering and Applied Science Department of Computer and Information Science*, 2017.

[21] Keeheon Lee, Jeongwon Jo, Jinyoung Kim, and Younah Kang. Can chatbots help reduce the workload of administrative officers?-implementing and deploying faq chatbot service in a university. In *International conference on human-computer interaction*, pages 348–354. Springer, 2019.

[22] Anita Ilić, Ana Ličina, and Dušan Savić. Chatbot development using java tools and libraries. In *2020 24th international conference on information technology (IT)*, pages 1–4. IEEE, 2020.

[23] Aliane Loureiro Krassmann, João Marcos Flach, Anita Raquel Cestari da Silva Grando, Liane Margarida Rockenbach Tarouco, and Magda Bercht. A process for extracting knowledge base for chatbots from text corpora. In *2019 IEEE Global Engineering Education Conference (EDUCON)*, pages 322–329. IEEE, 2019.

[24] Addi Ait-Mlouk and Lili Jiang. Kbot: a knowledge graph based chatbot for natural language understanding over linked data. *IEEE Access*, 8:149220–149230, 2020.

[25] Wanyun Cui, Yanghua Xiao, Haixun Wang, Yangqiu Song, Seung-won Hwang, and Wei Wang. Kbqa: learning question answering over qa corpora and knowledge bases. *arXiv preprint arXiv:1903.02419*, 2019.

[26] Lucrezia Grassi, Carmine Tommaso Recchiuto, and Antonio Sgorbissa. Knowledge-grounded dialogue flow management for social robots

and conversational agents. *International Journal of Social Robotics*, 14(5):1273–1293, 2022.

[27] Veton Kepuska and Gamal Bohouta. Next-generation of virtual personal assistants (microsoft cortana, apple siri, amazon alexa and google home). In *2018 IEEE 8th annual computing and communication workshop and conference (CCWC)*, pages 99–103. IEEE, 2018.

[28] Patrick Ignatius Patrick. Assessing loki by learning from mitsuku and dataset themes. B.S. thesis, University of Twente, 2020.

[29] Kam Fai Wong. Invited talk i nlp for microblog summarization.

[30] Ken Baclawski, Michael Bennett, Gary Berg-Cross, Todd Schneider, Ravi Sharma, Janet Singer, and Ram D Sriram. Ontology summit 2020 communiqué: Knowledge graphs. *Applied ontology*, 16(2):229–247, 2021.

[31] Ghadi Alyahya and Abeer Aldayel. Hatred stems from ignorance! distillation of the persuasion modes in countering conversational hate speech. *arXiv preprint arXiv:2403.15449*, 2024.

[32] Steve Ballmer. Ces 2010: A transforming trend-the natural user interface. *The Huffington Post*, 2010.