NetButler: Voice-Based Edge/Cloud Virtual Assistant for Home Network Management

Diogo Martins[1,2], Bruno Parreira[2], Pedro M. Santos[1,3][0000−0002−7162−0560], and Sérgio Figueiredo[4]

[1] Universidade do Porto, Faculdade de Engenharia
[2] Altran S.A.
[3] CISTER Research Center, Instituto Politécnico do Porto
diogo.lm111@gmail.com, bruno.parreira@altran.com, pss@isep.ipp.pt, sergio.figueiredo@altran.com

Abstract. Virtual assistants (VA) are becoming a standard tool in many aspects of our daily lives that require technical support. Voice-based VAs in particular, such as Amazon Alexa and Google Assistant, have become common in smart phones and domestic IoT devices (e.g., Google Home, Amazon Echo), replying to user inquiries (e.g., weather forecast) or performing simple services (e.g., play music). Through dedicated interfaces, VAs can be used or extended to support new services, and one particular area typically requiring assistance is the management of home networks. Activating specific features or troubleshooting connectivity problems may be difficult or impossible for users that are not tech-savvy. In this paper we introduce NetButler, a voice-based virtual assistant tailored to support the management of home networks, that leverages a third-party cloud-based voice service (Alexa) and dedicated routines at the home gateway. Offered functionalities are the setup of a guest network and diagnosis of connectivity problems, by quantifying the signal strength of the devices in the local network and performing a throughput test to an external server. We evaluate the user experience with the NetButler system with 8 test users. We report an average of up to 15s to set up a guest network and between 30 to 60s to diagnose various problems, and we find overall user satisfaction to be 3.75 in a 1-to-5 scale by means of a after-interaction questionnaire.

Keywords: Virtual Assistant · Voice Recognition/Synthesis · Home Network Management.

1 Introduction

A relevant share of the clients of residential broadband services have limited knowledge about the operation of modern communication networks and their own domestic network. Addressing issues or enabling additional functionalities - which are apparently hidden or out-of-reach - in their domestic network often involves contacting a support call center. This entails a number of inconveniences for both service provider and client: the service provider needs to assign human resources to customer service; it is hard to debug the problem with the imperfect information provided by the client; and the client may grow impatient or frustrated with the unfruitful interaction with another person. The problem is further aggravated by the relentless growth of the ecosystem of Internet-of-Things (IoT) devices, whose world market is estimated to reach US$ 135.3 billion by 2025 [7]. As home networks become crowded with personal devices and household appliances requiring Internet connection, clients that are not tech-savvy are increasingly more exposed to non-trivial connectivity problems.

Virtual assistants (VA) may offer an alternative to the traditional customer support processes, specifically first-line customer support. VAs are becoming increasingly relevant as they offer an intuitive and streamlined interaction experience with humans, whether through static interfaces (questionnaires), text messaging (*chatbots*), or voice interaction. On the latter category, voice-activated VAs are already offered as an embedded service or app in the smart phones and domestic IoT products of most major manufacturers of consumer electronics: some prominent examples include Amazon Alexa, Google Assistant, Apple Siri, and Microsoft Cortana. To offer some insight on their relevance, it is estimated that by 2027 the market size of VAs will be worth US$ 45.1 billion [6]. VAs allow a number of functionalities such as playing music, control smart devices, inform the user about various subjects (e.g., weather information), and even the possibility of hotel room reservation [13]. However, the potential of these systems for transforming household connectivity management into a more straightforward process for regular users has still not been properly exploited

In this paper we present **NetButler**, a voice-activated VA that helps the user enable functionalities and solve connectivity problems in the domestic network. The user communicates with the VA via smart phone, presenting the need/problem in question; the VA then applies the necessary actions at the home gateway or local-network router (typically, in households, these two network devices are in the same physical device provided by the ISP, so we use both terms interchangeably). Internally, the VA is composed by a third-party voice recognition/synthesis service, a cloud-hosted VA manager, and an agent at the home gateway to carry out the necessary routines. In its current version, the VA is able, based on the user's voice inputs, to set up a Wi-Fi *Guest* network and perform network diagnostics such as detecting wireless devices with poor signal strength in the WLAN network and testing bandwidth to external servers.

Our contributions are the following:

- Identification of scenarios of human-VA interaction towards diagnosing problems/activating features in home networks and their associated lexical field;
- Design of the VA architecture and implementation of its components in the home gateway and in third-party cloud services, whose entry point is the user's smart phone;
- Evaluation of the system with real users, that report low interaction times and overall positive experience.

The remainder of the article is as follows. In Section 2 we review the relevant works and technology related with voice-based virtual assistants. The system architecture and implemented use-cases are presented in Section 3. In Section 4, we describe an evaluation of the user experience with human testers. Final remarks are drawn in Section 5.

2 State-of-the-Art

Home Automation Systems (HAS): several projects were carried out with the aim of designing and implementing an architecture control equipment in a domestic environment. In [3], a message-based middleware platform was implemented to connect different types of IoT devices to monitor the energy used by household appliances. The interface is made through the Telegram application (application for text messages and multimedia content), in which the user interacts with an chatbot assistant. Energy consumption in KWh and cost for various timescales (per day, month, etc.) are shown through a ThingSpeak dashboard. The authors of [1] propose an architecture that supports remote control equipment through a web interface, in addition to virtual assistant control. The architecture includes sensors placed in several rooms measure temperature and humidity and actuators to control light intensity in the rooms, and integrates several types of open-source services: Firebase for the database, Google's WebSpeech for speech recognition, and DialogFlow for integrating Natural Language Processing (NLP) with the IoT system. The authors report that about 70% of the response time is spent by the NLP module, while the remaining is spent on processing in microcontrollers and communication between modules. In [12], a Home Automation System (HAS) controlled by an Android application communicates with the equipment in the house and with a web server is reported. The system receives voice commands through the mobile device and a VRS (Voice Recognition System) tool interprets and forwards the commands, in text form, to the microcontroller and the web server.

Systems Implementing & Using Voice-based Assistants: in [14], a virtual assistant prototype that reacts to the emotion displayed by the user is proposed. The key points on the user face (eyes, nose and mouth) are monitored to extract the associated emotion. While emotion identification is satisfactory, the intensity of the emotion is quite harder to identify. In [8], an webpage-based assistant interacts with university students and clarifies doubts regarding features/services. Through data collected at a satisfaction questionnaire at the end of the interaction, high rates of approval by the students were registered. The authors of [15] report a virtual assistant developed to assist users in online purchasing. The assistant is divided into agents responsible for 4 different stages in the interaction with the user. Finally, a virtual assistant to support human-human interaction in virtual environments is reported in [9]. The assistant offers suggestions of conversation topics in order to promote interactions between people from different cultures. Successful tests have been performed with interactions between Americans and Japanese.

Voice-Based Virtual Assistants: commercially available voice-based virtual assistants include Amazon Alexa, Google Assistant, Apple Siri, and Microsoft Cortana. The performance indicators of a virtual assistant (e.g., naturalness the accuracy of the response) often depend on the functionality being evaluated [13]. Open-source virtual assistants are also available and offer flexibility in order to be adapted to the particular needs of the target application. The Mycroft virtual assistant[4] is a community-supported virtual assistant; users can purchase additional features from an online marketplace. The Snips virtual assistant [4] is similar to Mycroft, i.e., it is community-supported, available for various platforms and offers a vast library of features online. A major difference is that this assistant can perform offline processing (without access to a cloud); this reinforces security (as no personal data is sent to the cloud) but requires a domestic hosting device powerful enough to host the speech recognition/synthesis technologies. This tool has been discontinued due to acquisition by another firm[5].

Systems for voice-based network interactions: Limited research has been realized on enabling voice-enabled network management systems. Authors in [2] proposed a wireless home automation system which automatically translates spoken words into text commands using MATLAB, which are then transmitted to a microcontroller controlling the household appliances via their corresponding relays. Rajalakshmi et al. [11] proposed a system to connect and control IoT devices using voice. The solution integrated: i) Alexa Voice Service for developing a customized skill for connecting IoT devices and publishing them; ii) Amazon Web Services (AWS) IoT service as centralized management platform for the multiple IoT devices; and iii) AWS Lambda service trigger to process voice commands from the user. The authors also suggest possibilities regarding the extension to computer networking and data monitoring.

3 NetButler: a Virtual Assistant for Domestic Networks

The system architecture and components are the following (depicted in Figure 1):

– **Voice Recognition/Synthesis (VRS) Service**: natural language processor responsible for mapping user voice utterances (i.e., spoken words or sentences) into *intents* that the system can act on, and convert system replies into speech;
– **User Equipment**: microphone/speaker-equipped device through which the user interacts with the VRS (e.g. smart phone);

[4] https://mycroft.ai/
[5] https://snips.ai/

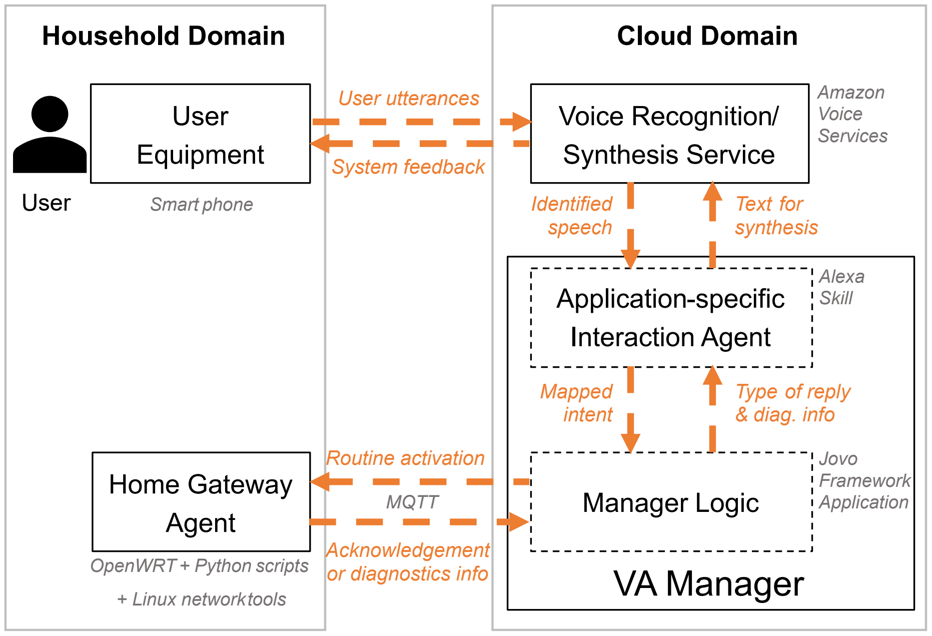


Fig. 1: Architecture of the voice-based home gateway assistant. Conceptual elements in black; technological elements (in the current version) in gray; data flows in orange.

– **VA Manager**: component that handles the interaction with the user (through a sub-component, the **application-specific interaction agent**) and commands the execution of the necessary actions at the home gateway based on the user inputs (through a second sub-component, the **manager logic**);
– **Home Gateway Agent**: component running on the home gateway/router that carries out the necessary actions as commanded by the VA manager.

We would like to highlight that the VRS service (and the VA Manager) could, in principle and in alternative to the present architecture, be hosted locally, i.e., at the user's premises. However, at the time of the execution of this project, the use of a third-party cloud-based VRS system was found to be the only viable option; we provide additional details at the end of Section 3.2.

The system's generic operation is as follows.

1. The user always initiates the interaction with the system through the user equipment (i.e., a smart phone in the current implementation), by instantiating the interface of the voice recognition/synthesis service (in smart phones, typically a service pre-installed in the device's OS, but possibly also an after-market app);
2. The cloud-based VRS service transforms the user's utterances into text, that in turn is processed by the application-specific interaction agent to extract the underlying intent;

3. The VA manager will map the received intent into one of several use-case workflows (described in the following subsection). In all such workflows, the VA manager signals the home gateway to initiate routines to implement/debug the identified need/problem. In some workflows, the VA may have to request additional information from the user before initiating those routines.
4. After the setup/debugging routines have been carried out at the home gateway, the home gateway replies to the VA manager with relevant confirmation or diagnostic information. The VA manager may instruct the application-specific interaction agent to forward a reply to the user (through the VRS service), with or without query-specific information.
5. Depending on the user reply, the interaction may be terminated or continue, in case the prior setup/resolution attempts were not successful and the use-case workflow encompasses additional steps for feature setup/debugging.

The following subsection describes the two use-cases developed for this system, and we conclude the section with a description of the current NetButler implementation.

3.1 NetButler Use-cases And Respective System Operation

The voice-based assistant provides support to two main classes of needs/problems, and offers execution of stand-alone functions. We now describe them in two dimensions: from the user's perspective, and in terms of the system's internal workflow.

Use-case 1 (UC1) – Creation of Guest Network: This use case aims at allowing the user to create a *Guest* network, without password and isolated through a firewall from the household network, at the time of the request or at a scheduled time, i.e. activating the network at a specific time and/or during a certain period. Some utterances associated to this use case are *"Share my Internet"*, *"Turn on the Guest Wi-Fi"* and *"Activate the Wi-Fi for my guests"*.

Internally, after the user requests the provisioning of network connectivity for guests, the VRS forwards the text/intent to the VA manager, that in turn commands the home gateway agent to activate the *Guest* network. The VA manager then sends a reply to the VRS system confirming that the network will be available within a few seconds, or at a time in the future, if it is being scheduled. A similar procedure is available to disable the network. Figure 2 describes the procedure to activate a *Guest* network.

Use-case 2 (UC2) – Diagnosis of Connectivity Issues: In this use case, the user informs the system he/she is experiencing limited connectivity to the Internet or a poor quality connection, thus hampering the use of online services. It is presumed that the user does not know the cause of such issues; the system will help the user identify the reason and offer suggestions to troubleshoot the

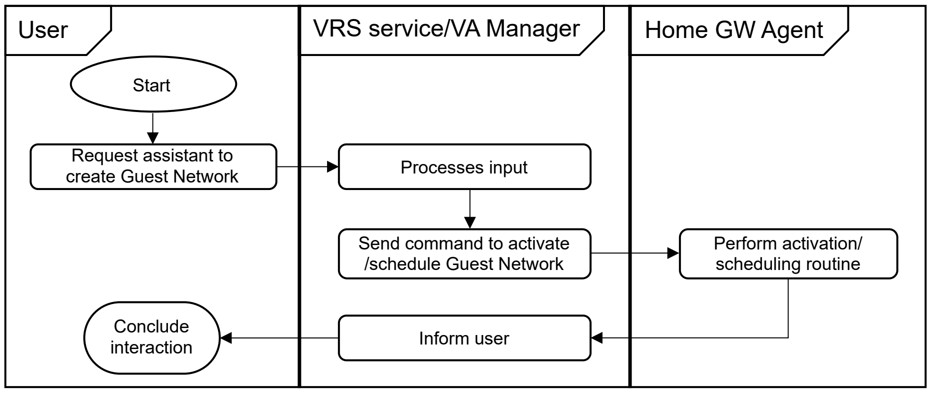

User
VRS service/VA Manager
Home GW Agent
Start
Request assistant to create Guest Network
Processes input
Send command to activate /schedule Guest Network
Perform activation/ scheduling routine
Inform user
Conclude interaction

Fig. 2: System workflow to create a guest network.

problem. The utterances associated to this use-case are, for example, *"I am having Internet problems"*, *"I am having problems watching a video"* and *"I can't access the internet"*.

Internally, NetButler addresses this as follows. The user initiates the interaction with NetButler, through the third-party VRS service. The VA manager acts on the intent detected from the user's utterance, initiating a diagnostic routine at the home gateway to ascertain the cause of the problem. Once the routine is complete, results are sent back to the VA manager that in turn builds a reply to the user and instructs the VRS to synthesize it. This procedure is shown in Figure 3.

The routine checks for two possible causes, by measuring and reporting related metrics back to the user. In both cases, a threshold is defined for the system to decide whether the user should be informed of that potential cause or not. Even if an issue is detected and reported, NetButler always inquires the user if he/she wishes to proceed with the diagnostic routine. The assistant always terminates the interaction by inquiring the user whether the home gateway should be rebooted; this operation is typically performed by first line customer support as it solves many common problems. The sub-routines and tested causes are presented next following order (and summarized in Figure 4 for convenience):

Poor signal quality: the system provides the user with the signal strength of wireless devices in the local network. We enforce a threshold of -80dBm to decide whether a device as *good* or *poor* signal strength; NetButler informs that at least one device with weak signal strength was found. System replies are as follows:

At least one device with below-threshold signal strength: *I detected at least one device with low WI-FI signal. If your device is far away from the router you might get Internet issues. Check your devices signal strength and get closer to the router if you find your signal low. If your device has good signal strength I can run a speed test*

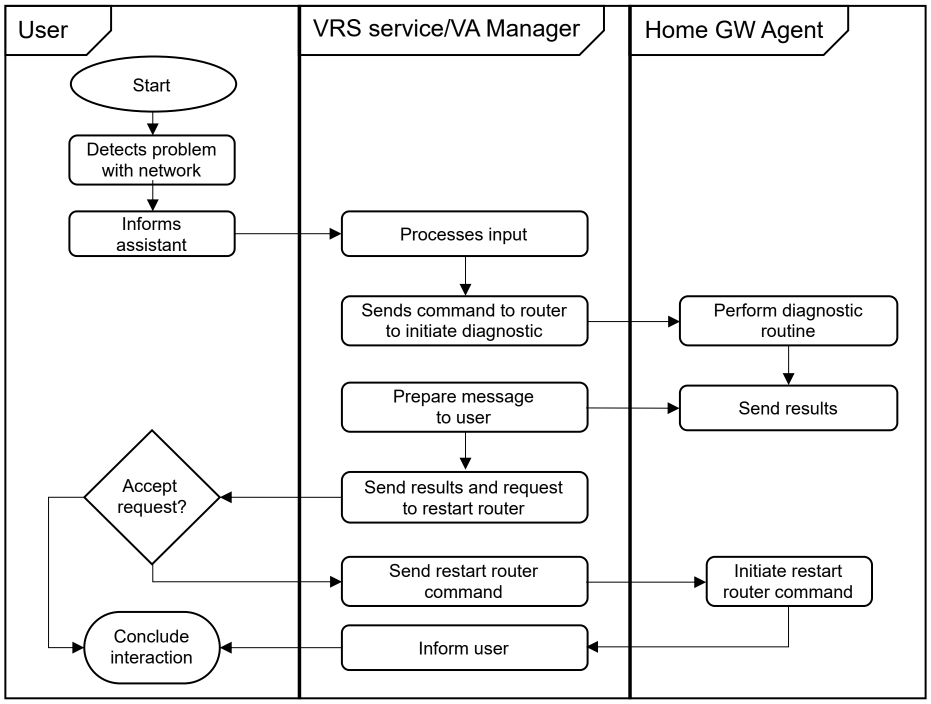

User
VRS service/VA Manager
Home GW Agent
Start
Detects problem with network
Informs assistant
Processes input
Sends command to router to initiate diagnostic
Perform diagnostic routine
Prepare message to user
Send results
Accept request?
Send results and request to restart router
Send restart router command
Initiate restart router command
Conclude interaction
Inform user

Fig. 3: System workflow to diagnose connectivity issues.

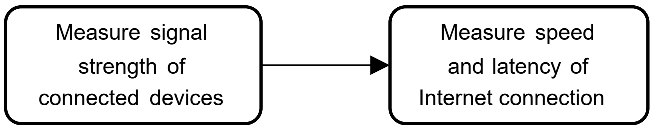

Measure signal strength of connected devices
Measure speed and latency of Internet connection

Fig. 4: Sub-routines of the diagnostic routine.

to finish the diagnosis. Can I run it?

All devices with above-threshold signal strength: *I don't detect any problems with the signal power on your devices. Can I run a speed test to finish the diagnosis?*

External bandwidth limitation: the system performs a connection speed test to a cloud server from the home gateway. We apply a threshold of 25Mbit/s (based on Netflix's recommended connection speed[6]) to classify the Internet connectivity as *acceptable* or *poor*. The possible replies are:

Poor service (low throughput) observed: *I figured that your Internet speed is very low at the moment, there is nothing you can do since the problem is external.*

[6] https://help.netflix.com/en/node/306

Table 1: Selected queries and predefined replies to stand-alone functions.

Function	Query	Reply
Speedtest	*Run a speed test* *Check my internet speed* *How fast is my internet?* *Measure my internet speed* *What is my latency?*	*The results of the speed test are the following: your download speed is [x] megabits per second, your upload speed is [y] megabits per second, and your latency is [z] milliseconds.*
Reboot	*Reset the router* *Restart the router* *Reboot the router* *Turn the router off ↪ and on again*	*Your router will reboot in a few seconds.*
#devices	*How many devices ↪ are connected?* *How many devices are ↪ using Internet?* *Count devices connected ↪ to the network*	*You have [x] devices connected to your network.*

Can I reboot the router?

Acceptable service/throughput observed: *The results of the speed test show me that you should be able to use your Internet with no problem. If you are having any issues check the other connected devices for any possible activity regarding the usage of your Internet. You might want to interrupt any download or video streams on other devices. Do you wish to reboot the router?*

Stand-alone Functions: NetButler offers the user the option to execute some stand-alone functions (i.e., outside the context of a diagnostic routine), namely: (i) run an Internet speed and latency test (**Speedtest**); (b) reboot the device (**Reboot**); and (iii) count the number of devices in the local network (**#devices**). Table 1 presents some possible queries and the predefined replies for the stand-alone functions.

3.2 Technological Components & Integration

The different system components were implemented with the following technologies:

– Voice Recognition/Synthesizer Cloud Service: **Amazon Alexa**
– VA Manager: **Alexa Skills Kit & Jovo Framework**
– Home Gateway Agent: **Linux Networking Tools**

Additionally, to support the communication between the VA manager and the home gateway agent, MQTT is used. We now motivate these options and explain the interaction between components that enables the system's operation.

Amazon Alexa: Amazon Alexa offers cloud-based speech interpretation and synthesis services that can be integrated with external applications. External applications use Alexa through *Alexa Skills*, that can be viewed (in some sense) as an application-specific instance of Alexa. The programming of a Skill involves providing a list of utterances and corresponding intentions related to the target application, alongside some configurations to establish a connection to the external application. After a Skill is set up, the following occurs: (a) the generic Alexa interaction agent is capable of identifying a user's request for the target application (e.g., NetButler), hence forwarding all subsequent user-Alexa interaction (for that session) to the related Skill; (b) the Skill interprets user inputs to verify if they match utterances defined in the Skill configuration, and forwards the corresponding intention to the external application components, namely Jovo; (c) as the external application components provides feedback, the Skill triggers the speech synthesis of the reply provided by the external application components through Alexa Voice Services. The smart phone component to connect to the cloud-based speech processor is also provided by Amazon.

Alexa Skills Kit & Jovo Framework: The Skills described earlier are offered by Alexa Skills Kit[7] (ASK). Alexa's Skills can be programmed with a dedicated SDK or through a third-party platform such as Jovo Framework. Jovo[8] allows to develop and run applications that use the voice interaction services from third-party providers (e.g., Amazon Alexa, Google Assistant and Samsung Bixby). The use of the Jovo Framework was partly responsible for the option of using Amazon Alexa; from the three options mentioned earlier, Amazon Alexa was chosen as it features the largest presence in households [5].

The Jovo Framework offers abstraction to the programming of Alexa Skills. Applications are programmed in Node Javascript, and the association with the Alexa online console is made through a *webhook* created by the framework. The Jovo application can be executed in any machine where the Jovo framework is installed (in our implementation we used a standard personal computer, but it could be equally deployed in a cloud server). The application's source code files can be binned into two categories: the user input mapping files and the VA manager logic. In the first category, user inputs are organized in a data structure within a JSON file. Within the file are all voice commands that may be recognized during the interaction of the user with the system. These phrases are organized in categories called **intents**, i.e., the idea or will that the user wants to transmit when interacting with the assistant. For example, the phrases *"Run a speed test"* and *"How fast is my Internet?"* belong the same intent, which would be *Speed test*. It also allows for slots to be filled in by the user, e.g., the time for scheduling the creation of the *Guest* network.

The VA manager logic handles the requests and subsequent interaction with the user. Once the VA manager receives from the VRS service the user utterance in text form, the VA manager searches the user input mapping files for

[7] https://developer.amazon.com/en-GB/alexa/alexa-skills-kit
[8] https://www.jovo.tech

a related utterance and identifies the corresponding intent. Depending on the identified intent, the system proceeds to follow one of the workflows described in Figures 2 and 3 or to execute one of the stand-alone functions. To that end, the VA manager prepares MQTT messages to be transmitted to the home gateway agent indicating the operations to be realized, to which the home gateway agent will reply with confirmation messages or the requested data. In that line, the VA manager also carries out a initial setup to subscribe the topics for which the router publishes and present some error messages in case the connection to the broker fails. The support for MQTT communication in the Jovo application is provided by the dedicated library MQTT.js[9]. The set of pre-defined replies to be offered to the user by NetButler is also stored in the VA manager logic.

Linux Networking Tools: the home gateway/router can be equipped with OpenWRT[10], a well-established open-source Linux-based operating system tailored for routers and network devices. Recall that we assume the home gateway and local-network router to be located in the same physical device, hence we use both terms are used interchangeably. In the scope of our system, the agent hosted at home gateway is composed of two elements: the agent itself, programmed in Python scripts and resorting to Linux commands and packages when necessary, and a MQTT broker that supports the communication with the VA manager. The home gateway agent starts by subscribing all topics on which the VA manager posts messages and vice versa. Topics are aligned with the particular features/problems being tackled by the system, e.g., *Guest Network*.

Upon reception of a new message, the agent has routines to perform all the functionalities described in Section 3.1: activate/deactivate of the *Guest* network; measure signal strength of devices; measure Internet speed and latency; reboot the device; and count number of devices. The creation of the *Guest* WLAN network is performed using Unified Configuration Interface (UCI), an interface provided by OpenWRT for editing the router settings. The created *Guest* network is detached from the remainder of the local network, forwarding the clients' traffic directly to the WAN (Wide Area Network) port. The speed and latency tests, as well as device counting and signal strength measurements, are performed by the Python module *speedtest-cli*[11]. There is also a support routine to encode the data to send to the assistant: due to the fact that the Alexa platform does not allow text inputs to contain numbers as numerals, a third-party library is used to transform numerals into their names[12]. For reference, development and implementation of the home gateway agent were carried out in a Linksys WRT-1200 router.

As a remark about the adopted architecture, in early design stages we set as desirable features: (i) the use of an open-source virtual assistant; (ii) implement

[9] https://www.npmjs.com/package/mqtt
[10] https://openwrt.org/
[11] https://github.com/sivel/speedtest-cli
[12] https://github.com/collin5/python-n2w

an architecture that would be local to the user's network, i.e., not requiring a cloud component; (iii) avoid the need for additional hardware/devices. However, at the time of this work, there were no readily-available open-source products that would allow this architecture to be implemented. The Snips virtual assistant [4], suited for deployment in Android phones, would allow for the system to be independent of any connection to the Internet, but the service has been discontinued in January 2020. We tried installing the alternative open-source voice-based VA, Mycroft, in the home gateway with the aid of external memory, but this approach proved unsuccessful due to the gateway's limited capabilities. The only remaining solution to offer a fully-local experience would be to deploy an additional physical device in the user's household to run Mycroft, a compromise that we considered to be extremely burdensome and outweighing the disadvantages incurred by waiving the first two desired features.

4 Performance Evaluation

The objective of this section is to evaluate the user experience when using the proposed virtual assistant. For that, three dedicated tests were designed and carried out by eight test users. At the end of the tests, the users are asked to fill a questionnaire to assess the quality of the interaction and outcomes.

4.1 Tests Description & Evaluation Metrics

The three tests map directly into the two use-cases of user-system interaction presented in Section 3.1, aiming at evaluating the use-case workflows under controlled conditions. The second use-case, *Diagnosis of Connectivity Issues*, was assigned two independent tests as the system is able to identify two potential causes for connectivity problems.

The sequence of steps is common to all tests: the user interacts with the system, that in turn replies with information about the problem and/or suggestions to address it. Some tests require may subsequent user-system interaction, or specific preparation to set up the conditions necessary for the test. In all cases, user-system interaction is performed using a smart phone.

Test 1 (T1) – Activation and deactivation of Guest network: There are no prerequisites for this test. The user interacts with a virtual assistant and the system enables the *Guest* network; in turn, the user checks the successful creation of the *Guest* network in the smart phone. In a second stage, the user interacts with the assistant to disable the network; a new check is made on the user's smart phone to confirm the *Guest* network has been disabled. We break down this test in two parts: **T1/On** (turning on the *Guest* network) and **T1/Off** (turning off).

Test 2 (T2) – Weak device signal: The prerequisite for this test is that the user must be at distance from the home gateway such that the signal strength of the user's smart phone reaching the home gateway is low (less than -80 dBm). The user initiates the interaction with NetButler through the smart phone, indicating issues in accessing the Internet. The system performs the first of the two diagnostic sub-routines described in Figure 4 – measurement of the device's signal strength – and identifies the cause of the problem at the end of the first sub-routine – poor signal strength. The system suggests the user to bring the device closer to the gateway to experience higher signal strength and, consequently, improve service quality.

Test 3 (T3) – Low Internet speed: In this test, the user simulates to be experiencing low Internet speed, that the system addresses by measuring the throughput and latency of a connection to an external server. As a prerequisite, the user's device must be within range of the home gateway to obtain good signal strength. The user starts the test by complaining about the quality of Internet service to the virtual assistant, that then initiates the two-stage diagnostic routine. After the device's signal strength is deemed appropriate in the first sub-routine, the system proceeds to measure throughput and latency and provides feedback on whether observed throughput is acceptable or poor. As this is the final stage of the diagnostic routine, the system always terminates the interaction after this feedback. Hence, the interaction time does not change considerably whether an actual bandwidth limitation is in place or not (and hence we did not enforce it as a prerequisite).

We evaluated the quality of the system-user interaction through an quantitative metric – the duration of the interaction – and a qualitative analysis – identification of difficulties throughout the interaction, such as misunderstandings by the user and/or system of the counterpart's intentions or feedback, respectively. Additionally, we carried out a subjective evaluation through a questionnaire filled by the users regarding their experience with the virtual assistant. The 6 questions are presented in Table 2 (note that questions were posed in Portuguese; a translation is presented). Testers were asked to reply on a Likert scale [10] of 1 to 5 (1 being the least positive feedback and 5 the most positive feedback).

4.2 Evaluation

The pool of available test users was limited to eight people and composed mostly of young people (below 30 years old), due to the restrictions imposed by the COVID-19 situation; we intend to expand our set of system testers in the future to obtain more representative results. The eight users were binned into the following categories:

- *Adult:* 2 males with 45 and 53 years and a female with 50 years, background on engineering and accountancy;
- *Young adult with IT background:* 2 males of 19 and 23 years;

Table 2: Questionnaire posed to testers (translated from Portuguese).

Questions	Scale and Options
1. How do you classify the clarity and objectivity of the answers presented by the system?	1-Unclear; 3-Sufficient; 5-Clear
2. How do you classify the degree of technical terminology in the speech?	1-High; 3-Intermediate; 5-Low
3. How do you classify the suitability of the proposed solutions to your problem/need?	1-Inadequate; 3-Helpful; 5-Ideal
4. How do you classify the speed with which the system identified the problem/need?	1-Slow; 3-Appropriate; 5-Fast
5. Overall, how do you classify your experience with the virtual assistant?	1-Bad; 3-Average; 5-Good
6. Do you consider that virtual assistants can help in managing home networks?	1-No; 3-Maybe; 5-Yes

Table 3: Duration of interaction (in seconds).

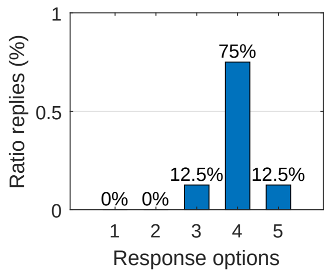
	T1/On	T1/Off	T2	T3
Avg. μ	14.3	13.1	31.4	68.6
Std. dev. σ	1.030	0.661	1.323	1.580

– *Young adult with non-IT background:* 2 females with 22 and 23 years and a male with 25 years.

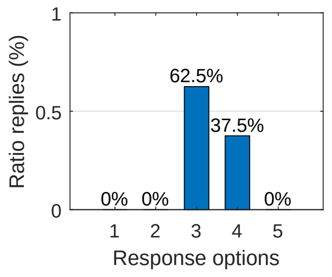
All users were Portuguese, albeit queries to the system must be made in English. The users were instructed about each use-case, but were not given any indications about the format and/or terminology that their query should follow. Some examples of posed questions per test are: **T1:** "I want to share my Internet."; **T2:** "I don't have Internet."; **T3:** "My Internet connection is slow". Each one of the 8 users performed all three tests, totalling 24 tests.

Interaction Duration & Issues: Table 3 reports the average duration (and respective standard deviation) for all tests. Tests T1/On & T1/Off, that address a well-defined functionality, took 14.3s and 13.1s in average respectively. In turn, tests T2 and T3 relate to troubleshooting use-cases and hence last more due to the diagnostic routine – 31.4s & 68.6s respectively. In fact, T3 takes more time simply because both steps of the diagnostic routine are performed. Test duration among the eight users was very similar, as standard deviation is fairly small; this shows that the system offers a consistent user experience.

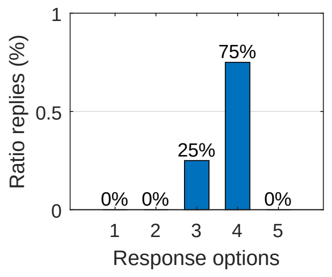
The values presented for the interaction duration consider only the tests in which the system correctly identified the target issue from the initial user input. Throughout the 24 tests, there were a total of 4 occurrences in which the initial user input did not succeed in starting the interaction. In two of the cases, the users did not mastered the English language, and the query had to be rephrased; in the other two cases, the user utterance was not recognized by the system.



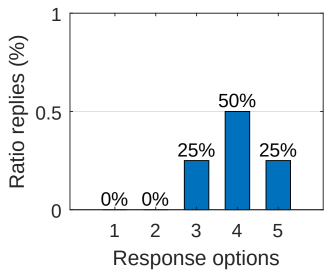
(a) Question 1.



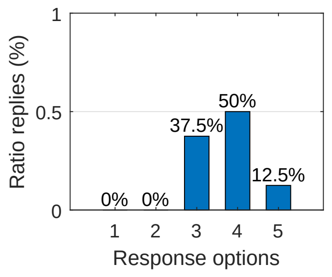
(b) Question 2.



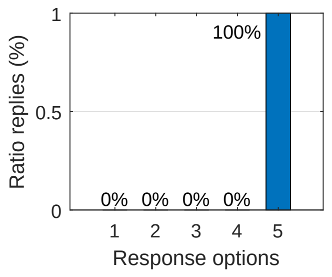
(c) Question 3.



(d) Question 4.



(e) Question 5.



(f) Question 6.

Fig. 5: Replies to user experience questionnaire.

User Experience and Satisfaction: Figure 5 presents the replies to the questionnaire performed at the end of the tests. The clarity of the responses provided by the system was judged to be of level 4 (out of 5, with 3 being *Sufficient* and 5 being *Clear*) by 75% of the users. The level of technical terminology was deemed *Intermediate* by 62.5% of the users, with the remainder of the users considering it to be tendentially *Low*. This shows that the users experienced a natural interaction with the VA and understood the replies it provided; this can be considered an advantage given that the system aims to support non-tech-savvy users. The pertinence of the proposed solutions and speed of the resolution were also perceived as positive by the users: users reporting a level of 4 out 5 amounted to 75% regarding perceived pertinence and 50% regarding speed. Finally, we observed an overall good appreciation of the system, as shown by the answers to Question

5 (*"Overall, how do you classify your experience with the virtual assistant?"*), with 62,5% of the users reporting a satisfaction level of 4 or higher.

5 Conclusion

We present **NetButler**, a voice-based virtual assistant that helps clients of residential broadband services to address needs and problems in their domestic networks. The system leverages Amazon's natural language processing (NLP) service Alexa and operates routines in the home gateway through Linux/OpenWRT networking tools; a cloud-hosted VA manager orchestrates user interaction and home gateway operations. We evaluate the user experience and perception of the system under three use-cases: creation of a guest network, poor signal quality, and poor Internet service due to external causes. In evaluation with 8 non-technical users, the system was able to address all user requests with interaction time greatly inferior to those that a call to a support center would entail. Half the users reported an experience satisfaction level of 4 in a 1-to-5 Likert scale, with 5 being the most positive.

All test users highlighted, in the final question of questionnaire (*"Do you consider that virtual assistants can help in managing home networks?"*), the tremendous potential of the virtual assistants. This motivates us to pursue follow-up work to the current version of NetButler. First, we plan to carry out larger-scale user experience tests, as the pool of users in the current work was fairly limited and biased towards younger users. A particular issue to be addressed is the identification of the user's own device: while the home gateway is able to report MAC addresses, NetButler is currently not able to learn the MAC address of the user's device (as communication between the smart phone and VA Manager is done through the third-party Alexa tool) nor offers the option for the user to input it. This feature would be helpful to provide more meaningful indications to the user regarding his/her own device. We will continue to explore the possibility of a local solution, i.e., to have the voice-assistant deployed at the user's home (e.g., in the home gateway or smart phone), so that no connection to the Internet is required. Additional features, such as dynamic bandwidth adjustment, will be studied; and support for other languages will be incorporated, given that we observed this aspect to be critical to the naturalness of the interaction.

Acknowledgements

This work was partially supported by National Funds through FCT/MCTES (Portuguese Foundation for Science and Technology), within the CISTER Research Unit (UIDB/04234/2020).

References

1. Alexakis, G., Panagiotakis, S., Fragkakis, A., Markakis, E., Kostas, V.: Control of smart home operations using natural language processing, voice recogni-

tion and IoT technologies in a multi-tier architecture. Designs **3**, 32 (Jul 2019). https://doi.org/10.3390/designs3030032

2. Amrutha, S.R., Aravind, S., Mathew, A., Sugathan, S.: Voice controlled smart home (2015)
3. Chilcañán, D., Navas, P., Escobar, M.: Virtual Assistant for IoT process management, using a middleware. In: Proceedings of the 2018 2nd international conference on algorithms, computing and systems. pp. 209–213 (2018)
4. Coucke, A., Saade, A., Ball, A., Bluche, T., Caulier, A., Leroy, D., Doumouro, C., Gisselbrecht, T., Caltagirone, F., Lavril, T., Primet, M., Dureau, J.: Snips Voice Platform: an embedded Spoken Language Understanding system for private-by-design voice interfaces. CoRR (2018), http://arxiv.org/abs/1805.10190
5. eMarketer: Alexa, Say What?! Voice-Enabled Speaker Usage to Grow Nearly 130% This Year (May 2017; accessed July 2020), https://www.emarketer.com/Article/Alexa-Say-What-Voice-Enabled-Speaker-Usage-Grow-Nearly-130-This-Year/1015812
6. Grand View Research: Intelligent Virtual Assistant Market Size, Share & Trends Analysis Report by Product (Chatbot, Smart Speakers), by Technology, by Application (BFSI, Healthcare, Education), by Region, and Segment Forecasts, 2020 - 2027 (April 2020; accessed July 2020), https://www.grandviewresearch.com/industry-analysis/intelligent-virtual-assistant-industry
7. Grand View Research: Smart Home Market with COVID-19 Impact Analysis by Product (Lighting Control, Security Access Control, HVAC Control, Entertainment, Home Healthcare), Software & Services (Proactive, Behavioural), and Region - Global Forecast to 2025 (June 2020; accessed July 2020), https://www.marketsandmarkets.com/Market-Reports/smart-homes-and-assisted-living-advanced-technologie-and-global-market-121.html
8. Harvey, P.H., Currie, E., Daryanani, P., Augusto, J.C.: Enhancing student support with a virtual assistant. In: Vincenti, G., Bucciero, A., Vaz de Carvalho, C. (eds.) E-learning, e-education, and online training. pp. 101–109. Springer International Publishing, Cham (2016)
9. Isbister, K., Nakanishi, H., Ishida, T., Nass, C.: Helper agent: Designing an assistant for human-human interaction in a virtual meeting space. In: Proceedings of the SIGCHI conference on human factors in computing systems. pp. 57–64. CHI '00, Association for Computing Machinery, New York, NY, USA (2000). https://doi.org/10.1145/332040.332407, https://doi.org/10.1145/332040.332407
10. Likert, R.: "a technique for the measurement of attitudes". Archives of Psychology **140**, 1–55 (1932)
11. Rajalakshmi, A., Shahnasser, H.: Internet of things using node-red and alexa. In: 2017 17th international symposium on communications and information technologies (ISCIT). pp. 1–4 (Sep 2017). https://doi.org/10.1109/ISCIT.2017.8261194
12. Tharaniya soundhari, M., Brilly Sangeetha, S.: Intelligent interface based speech recognition for home automation using android application. In: 2015 international conference on innovations in information, embedded and communication systems (ICIIECS). pp. 1–11 (Mar 2015). https://doi.org/10.1109/ICIIECS.2015.7192988
13. Vtyurina, A., Fourney, A.: Exploring the role of conversational cues in guided task support with virtual assistants. In: Proceedings of the 2018 CHI conference on human factors in computing systems. CHI '18, Association for Computing Machinery, New York, NY, USA (2018). https://doi.org/10.1145/3173574.3173782, https://doi.org/10.1145/3173574.3173782

14. Wang, Z., Cheng, N., Fan, Y., Liu, J., Zhu, C.: Construction of virtual assistant based on basic emotions theory. In: Tao, J., Tan, T., Picard, R.W. (eds.) Affective computing and intelligent interaction. pp. 574–581. Springer Berlin Heidelberg, Berlin, Heidelberg (2005)
15. Xu, B., Pan, Z., Yang, H.: Agent-based model for intelligent shopping assistant and its application. In: The first conference on affective computing and intelligent interaction, beijing. pp. 306–311 (2003)