Paper number ITS-XXXX

A Voice Interaction for the LinkBeyond Mobile MaaS Application

André Costa[1*], Manuel Relvas[2], Duarte Ricciardi[3], Alberto Abad[4]

[1]A-to-Be Mobility Technology, SA, Portugal (andre.costa@a-to-be.com)

[2]A-to-Be Mobility Technology, SA, Portugal

[3]Via Verde Serviços, SA, Portugal

[4]INESC-ID / Instituto Superior Técnico

Abstract

New paradigms like MaaS (Mobility as a Service) are emerging based on a holistic view of people's mobility needs. The A-to-Be's LinkBeyond Mobile is a multi-service system, made of a mobile application capable of integrating different mobility players supported by a multi-operator Back-End infrastructure. The discovery and/or consumption of some mobility and utility services may be useful to be done while driving (e.g. search for a fuel station or car washing) and, therefore, a non-tactile interaction with the system is required to preserve safety. This paper describes a voice interaction prototype implementation using a Mobile application for discovering and consuming utility services for the Portuguese context, specifically car washing and vacuuming. It starts by detailing the addressed use-cases followed by a comparative analysis of market voice platforms supporting European Portuguese and, finally, describes the evaluation done to the prototype built based on the chosen voice processing platform.

Keywords:

Mobility application, Voice interaction, Natural language understanding

Introduction

Mobility paradigms are changing, not only in the way services are made available, but especially in the way how they are presented and consumed by citizens. The concept of Mobility as a Service (MaaS) is a fundamental instrument, demanding new tools to support the new business and operating models. The A-to-Be's LinkBeyond Mobile [1] solution aims to position itself in the market as a way of interacting with mobility services through a mobile application, connecting operators and service providers to deliver a seamless mobility experience. The interaction available with these mobility services is done through the discovery and consumption of services, among other functionalities. In terms of architecture, the A-to-Be's LinkBeyond Mobile solution is made of an application running on a mobile device (APP) that may communicate with a wireless device known as Local Access Mediator (LAM) module, which will interact with the operator's equipment. All this interaction is supported by a multi-operator Back-End infrastructure.

The Back-End infrastructure is comprised of several cloud-based applications using a microservice architecture and that makes the discovery and consumption of services available to the users through a

Mobile SDK, that can be easily integrated in any mobile application. It supports multitenancy, meaning that a single instance of the Back-End can support several operators running simultaneously. To setup the system, the involved supplying entities must register themselves in the system by using the configuration portal: 1) Merchant - entity responsible for user and payments management and 2) Publisher - entity responsible for uploading operational data by using configuration files corresponding to one or more operators (identity, services and offer, provision places, tariffs and prices). The system is fully compliant with the GTFS specification and extends this specification to support more mobility services besides public transportation. Not only the Back-End supports operator's static offers defined through configuration files, it can also integrate with the operator's Backend in order to get the offer information in real-time (dynamic offers). Furthermore, for services that needs to be consumed while the user may not have network coverage (for example, an underground off-street park), the system supports the offline offers. In that case, the necessary service offer information is downloaded and stored in the mobile SDK when the customer's login and it is used together with the LAM to securely give access to the service. All the services usage is coordinated with the system responsible for the user's account management, that can be provided by A-to-Be's MoveBeyond or any 3rd party system to which LinkBeyond Mobile can integrate.

Via Verde, a leading Portuguese company providing electronic payment for mobility-related services has a 25 years' innovative track record related with highway tolling, On Board Unit based, that has expanded to other services where is a need for access and payment. Knowing that business future will not rely on the OBU alone anymore, Via Verde started developing a new business era, basing its services on the usage of Mobile Applications running on Smartphones. Since some of the services to be addressed are mobility related while being inside a vehicle or even driving, the next logical step is to have a non-tactile interaction with the system in order not to compromise safety. Therefore, by using A-to-Be's LinkBeyond Mobile solution, Via Verde can have a multi-service platform solution and a voice interaction mechanism, using European Portuguese, in order to better understand the voice processing technology maturity in different usage environments and assess the integration complexity in a mobile application.

This paper presents a prototype implementation of a European Portuguese voice interaction mechanisms using an Android Mobile application. It starts by detailing the use-cases defined for discovering and consuming car washing and vacuuming cleaning services followed by a comparative analysis of 2 selected market voice processing platforms that support European Portuguese, while used in different usage environments (quiet, car background noise, music). Finally, the development of the prototype is described, focusing on the integration complexity of the voice processing platform and presenting an evaluation based on dialog understanding and associated latency.

Use-cases definition

The A-to-Be's LinkBeyond Mobile is a multi-service system that provides off-the-shelf capability of discovering and activating different mobility and utility services, such as car washing and vacuuming through a conventional Smartphone GUI. Therefore, for assessing voice interaction, four use-cases were defined and, to avoid further complexity, the LinkBeyond Mobile application existing flow was kept. The four use-cases are detailed from Table 1 to Table 4. Regarding the dialog flow specified, the following rules apply:

1. the user can ask for the last sentence vocalized by the App to be repeated by saying a sentence meaning ‘repeat’;
2. the mobile app repeats the last sentence when 1) the final user asks to do so, 2) it does not understand the sentence spoken by the user or 3) it is not capable of executing the user’s request considering the context, despite having understood the sentence;
3. the final user can abort the dialog by saying ‘quit’, ‘forget’ or other similar sentences.

Table 1 – Service information use-case

	Sentence	Action
User	“What is the closest cleaning service?”	System gets the list of available cleaning services sorted by proximity.
App	“There is a cleaning service available in the airport station, at 7,8 km. Do you want to learn more about this location or go to the next one?”	
User	“Next, please.”	System gets the second closest cleaning service.
App	“There is a cleaning service available in the Olivais station, at 11,7 km. Do you want to learn more about this location or go to the next one?”	
User	“I want to know more!”	System gets the details of the chosen cleaning station.
App	“In the Olivais station, there is a roll over washing machine, with four variants: a) Simple, b) Super, c) Special and d) Extra.”	

Table 2 - Jet wash use-case

	Sentence	Action
User	“What are the cleaning services in this station?”	System gets the cleaning services in the current location.
App	“In the airport station, there are 2 services available: a) jet wash and b) vacuum cleaning;	Final user approaches the jet wash machine.
User	“Please, activate the jet wash service!”	System gets the details of the service.
App	“This service has a cost of 50 cents and a duration of 30 seconds, do you confirm?”	
User	“Yes!”	System activates the jet wash service.

	Sentence	Action
App	"Jet wash service activated with success. To add 30 seconds for 50 cents please say 'extend' "	
User	"Extend!"	System extends the jet wash service.
App	"Jet service extended for 30 seconds more. For adding 30 seconds more for 50 cents please say 'extend'"	Jet wash service automatically ends after the established duration.
App	"Jet wash service used for total time of 1 minute and 1 euro was charged on your Via Verde account. Thank you!"	

Table 3 - Roll over washing use-case

	Sentence	Action
User	"Please wash my car in this roll over washing machine!"	System gets the details of the roll over washing service in the current location.
App	"In the Olivais station there is a roll over washing machine with 4 variants: a) Simple, b) Super, c) Special and d) Extra;	Final user approaches the roll over washing machine and decides which variant he or she wants.
User	"Activate Simple!"	System gets the details of the Simple variant.
App	"This variant has a cost of 2 euros and 95 cents and a duration of 6 minutes, do you confirm?"	
User	"Yes!"	System locks the roll over washing machine for usage.
App	"Simple variant activated with a cost of 2 euros and 95 cents, charged on your Via Verde account. Please, put your vehicle in the correct position and say, 'I'm ready' to start washing."	Final user puts the vehicle in the correct position.
User	"I'm ready!"	System activates the roll over wash machine.
App	"Washing process started! The service will stop automatically."	Final user waits until the washing process is concluded.

Table 4 - Vacuum cleaning use-case

	Sentence	Action
User	"I would like to vacuum my car!"	System gets the details of the vacuuming service in this location or nearby.
App	"In the A5 station there is a vacuum cleaning service with a cost of 1 euro and a duration of 7 minutes. Do you confirm?	
User	"Yes!"	
App	"OK. Please, go to the vacuuming zone and when ready please say 'Activate vacuuming'"	Final user goes to the vacuum cleaner number 3.

	Sentence	Action
User	"Activate vacuuming!"	System gets the user's closest vacuum cleaner using LAM advertisement. However, the user position is not the expected one (ie not near the cleaner nozzle), but nearer to the LAM of cleaner 2.
App	"You are standing beside the vacuum cleaner number 2. Do you confirm?"	
User	"No, I'm in the vacuum cleaner number 3!"	
App	"You are standing beside the vacuum cleaner number 3. Do you confirm?"	
User	"Yes!"	System activates vacuum cleaner number 3.
App	"Vacuuming process started! Thank you!"	

Voice processing platforms: comparative analysis

The purpose of this analysis is to provide a comparison between voice processing engines that support Portuguese from Portugal. The engines must extract user intentions from spoken sentences and must generate sentences correctly, including colloquial inflections and regional/foreign accents. Both engines work by trying to match a spoken sentence to a set of pre-defined sample sentences, from which an intention is extracted (eventually together with some related parameters). The voice platform building blocks used for this analysis are depicted in Figure 1 and described next [2].

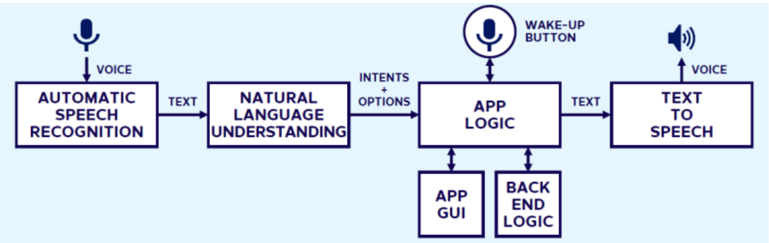

Figure 1 - High-level block diagram

AUTOMATIC SPEECH RECOGNITION (ASR) converts the captured speech into text. Cloud-based solutions provide large vocabulary continuous speech recognition (LVCSR) capability thanks to the exploitation of very large acoustic and statistical language models [3]. On the other hand, embedded solutions are configured with smaller recognition models limiting the transcription ability of this type of services [4]. In some cases, the set of words and sentences that can be effectively recognised by embedded solutions needs to be configured through the definition of a closed grammar[1].

[1] Notice that when using closed grammar-based technologies, sentences that are recognised by the ASR are only those that are defined for the NLU phase (that are implicitly converted to a closed-grammar) , so in practice both steps (ASR and NLU) are performed together.

NATURAL LANGUAGE UNDERSTANDING (NLU) extracts intents from sentences that are spoken by the user and recognized in the ASR phase. Intents can have associated parameters that characterize or quantify the intents. The intents and their eventual parameters allow the app logic to determine the evolution of the state of interaction with the user and the corresponding voice dialogue.

TEXT TO SPEECH (TTS) generates the vocal messages issued by the app. Must be able to interpret punctuation marks and / or SSML (Speech Synthesis Markup Language) elements.

More than comparing two providers of natural language processing platforms, we are comparing two technologies: 1) speech recognition using a statistical language model that requires a cloud-based infrastructure and 2) speech recognition using closed speech grammar suitable for embedded installations. While the former allows far more natural and free dialogues, the latter depends on stricter rules previously established for each application environment. The evaluation methodology had the following steps:

1. Acquire speech samples from different people and in different acoustic environments

   The speech samples were recorded in 3 different environments: 1) quiet room, 2) moving car with radio off and 3) moving car with radio on. Testers with standard, regional and foreign accents were selected in order to record the speech samples consisting in user readings of a selected set of 15 sentences extracted from the use cases (samples were recorded on a Samsung Galaxy S6 - medium range smartphone).
2. Send the speech samples into the engines in order to extract metrics

   Standard voice processing analysis metrics are used:
   a. Word Error Rate (WER)[2] – ASR analysis (number of inserted + deleted + substituted words) / number of words;
   b. Classification Accuracy (NLU analysis) number of intents correctly detected / number of sentences;
3. Present the speech messages generated by the engines to appreciation of several testers

   Analysed by using 6 representative sentences of the use cases, regarding 3 different vocalizers. The testers listened to the voice samples from each vocalizer evaluating:
   a. Intelligibility (understanding what is being said);
   b. Naturalness (how natural/robotic the voice sounds);
   c. Overall (which is the preferred vocalizer).

ASR Word Error Rate

Figure 2 shows the average WER obtained for the statistical language model platform ASR. As expected, the WER increases when there is more background noise, being below 20% when in a quiet room environment and increasing to almost 50% when in a moving car with radio on.

[2] WER is not applicable to closed grammar technologies since ASR and NLU steps are performed together

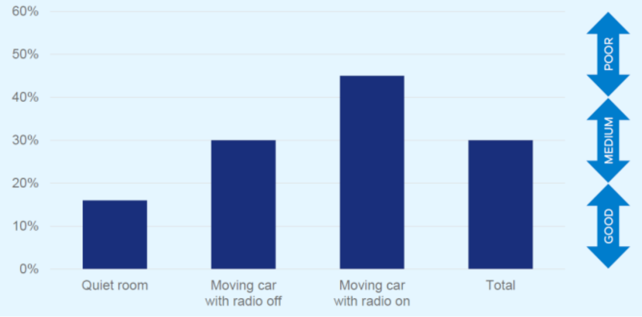

Figure 2 - statistical language model platform Average WER

NLU Classification Accuracy

Figure 3 depicts the classification accuracy for both NLUs: 1) statistical language model and 2) closed speech grammar. The results show that, when used in a quiet room, the closed speech grammar NLU performs slightly better, obtaining more than 95% of success in getting the correct intent from the user's spoken sentence. However, when the background noises increase, the statistical language model NLU shows a higher classification accuracy obtaining almost 80% of success when used in a moving car with radio on, comparing to 50% of the closed speech grammar NLU.

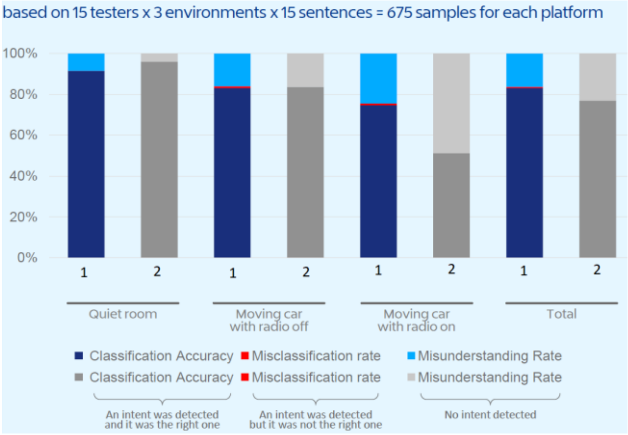

Figure 3 - NLUs classification accuracy

TTS Intelligibility

Regarding the TTS intelligibility results, three different vocalizers were used: 1) smartphone native, 2) cloud-based and 3) for embedded applications. Figure 4 shows that the testers selected vocalizer 2 as preferred, flowed by vocalizers 3 and 1, respectively.

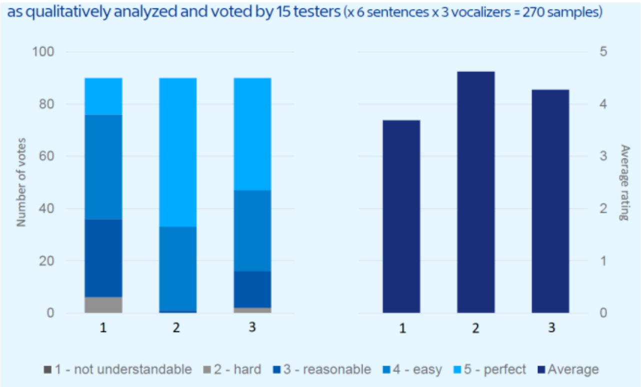

Figure 4 - TTS Intelligibility evaluation

TTS Naturalness

Figure 5 depicts the tester's evaluation for the vocalizer naturalness. The results show that the more natural vocalizer selected by the users was number 2.

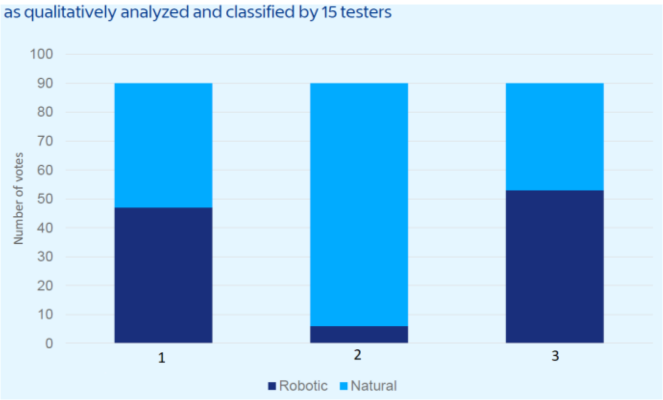

Figure 5 - TTS Naturalness evaluation

TTS overall evaluation

Figure 6 shows the vocalizer's qualitative assessment made by the testers and the results clearly show that the preferred one was number 2 by far (78%), while number 3 and number 1 obtained 19% and 3% of preference, respectively.

Looking at the results presented above, the statistical language model + vocalizer 2 is the recommended platform for a prototype implementation because:

1. it shows a fair speech recognition performance, even in acoustically aggressive environments;
2. its foundation technology (statistical language model) provides an unbeatable ability to understand open dialogues (i.e. without an explicit and exhaustive pre-definition of dedicated grammar rules) and resilience to acoustically aggressive environments;
3. it was very clearly elected by the testers as the most intelligible, natural and overall performant platform.

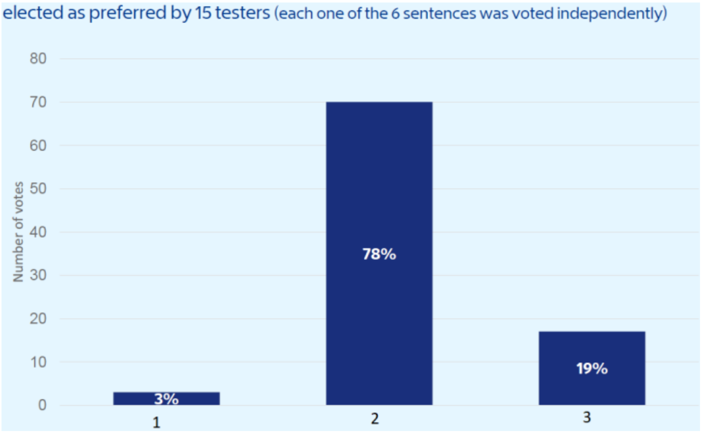

Figure 6 - TTS Overall evaluation

Prototype: development, test and evaluation

The purpose of the prototype development is a practical evaluation of the use of voice interaction in the LinkBeyond multi-service Mobile Application. Figure 7 shows the architecture of the prototype developed and Table 5 describes the app usage flow when using voice interaction.

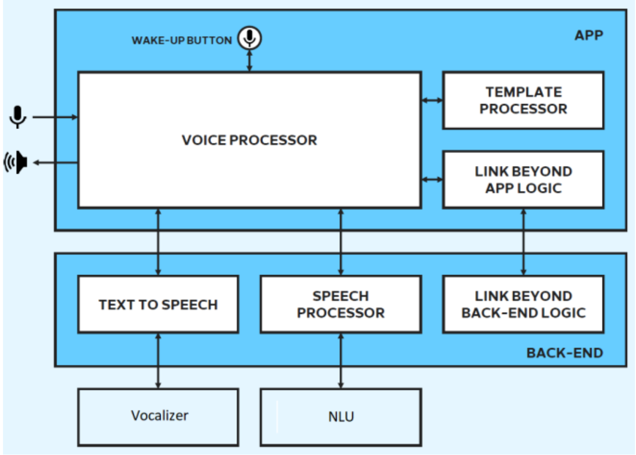

Figure 7 - Prototype architecture

Voice Processor (Mobile App)

Wake-up[3] button management, audio capture and packaging, Speech Processor invocation for intent detection, exception instruction processing (e.g. repeat and terminate) and generation of voice messages using the Template Processor and invoking the Text to Speech.

Template Processor (Mobile App)

Dynamic message building based on templates and context variables, including automatic insertion of SSML prosodic markers.

Speech Processor (LinkBeyond Back-End)

Intent detection by invoking the cloud NLU, including managing credentials and implementing the selection of services and/or variants from a given list.

Text to Speech (LinkBeyond Back-End)

Generation of voice messages through the cloud-based vocalizer invocation and credential management.

Table 5 - Prototype usage flow

Phase	Name	Description
1	Command	The user pushes the wake-up button and verbalizes the instructions regarding the operation he intends to perform.
2	Packaging	The app recognizes the end of the sentence (by timeout, silence or pressing wake-up button) and creates the corresponding audio file.
3	ASR+NLU	The app invokes the BackEnd service that performs the extraction of intents. This service uses the NLU and returns the intent extracted from the user sentence.

[3] Since the cloud-based platform solution was chosen, the implementation of a custom wake-up word is not possible.

Phase	Name	Description
4	Business Logic	The app parses the user intent, updates the internal state machine (typically involving more BackEnd interactions), refreshes the graphical user interface and selects the vocal response template.
5	NLG	The app instantiates the selected template, generating the response string according to the current service context (Natural Language Generation).
6	TTS	The app sends the response string to the BackEnd which uses the vocalizer to return the corresponding audio file.
7	Response	The app plays the audio file of the response.

The prototype assessment was carried out at two levels: 1) compliance with previously defined use cases and 2) measurement of dialog latency, identifying the contribution of each phase of the vocal interaction process. Given the interest in studying dialog latency in a complex process of interaction, a test use case was defined as the result of the combination of Information and Roll over Washing use cases. Figure 8 shows the dialog duration for this combined use-case using 3 connection types.

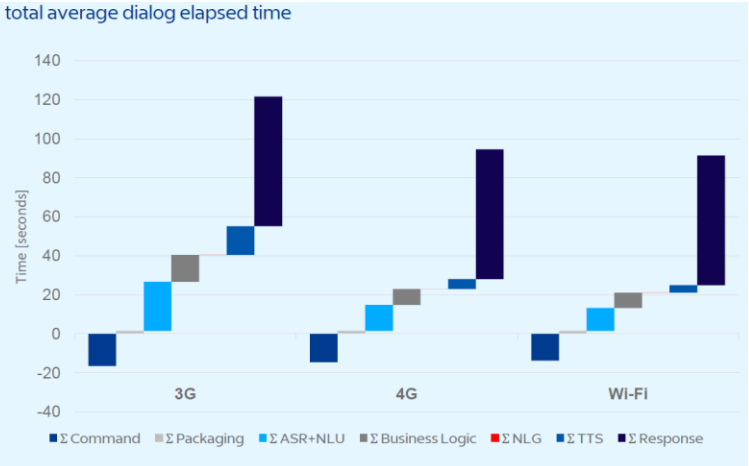

Figure 8 - Dialog duration per connection type

Regarding comprehension and naturalness parameters, all previously defined use cases were successfully implemented. In the latency evaluation performed, most (70%) of the time was spent on voice interaction process which refers to the messages vocalized by the user (13%) and by the app (57%). However, this time is not perceived by the users as undesirable latency, as the perceived latency refer to the gaps between the user-issued commands and the corresponding answers:

- intent detection (ASR+NLU) - 14% of the total interaction process;
- business logic (App+BackEnd) -8% of the total interaction process;
- voice message generation (TTS) - 7% of the total interaction process.

There will always be a significant impact of exogenous factors (e.g. quality of the mobile network) in the overall performance. In order to reduce latency in conversation streams, the initial architecture was modified to invoke the NLU directly from the Mobile App using streaming mode and use Android's native TTS function. The results achieved were:

- 64% reduction of the perceived latency (NLU + business logic + TTS);
- 84% reduction of the voice processing delay (NLU + TTS).

Conclusions

The results achieved were positive and the ability of current technology to recognize voice instructions, to extract the intents expressed in these instructions and to deliver spoken messages with a high level of comprehension and naturalness was confirmed. The main objective of work was thus achieved, which was to understand how suitable a voice interface could be for the intended application - a multi-service app for the discovery and consumption of mobility and mobility-related services.

The results obtained clearly point to a preference for the technology presented by the statistical language model because it provides a good recognition rate of the intentions expressed by the users' spoken messages, particularly superior in environments with higher levels of ambient noise and a low complex integration and configuration processes, providing good graphical interfaces to define intents and their eventual parameters that can be done by people without programming skills. As far as synthesized messages are concerned, the cloud-based vocalizer got a clear preference from testers but contributes very significantly to a high latency in the dialogues. The native smartphone vocalizer proved to be good enough for the intended application, not compromising the ultimate purposes.

Although not considered in the scope of the project, the existence of a wake-up word seems relevant for the purpose of providing a 100% vocal interface, particularly in driving situations. To pursue this goal, a specialized product can be used for this purpose without the assumption of being a full conversational tool (e.g. CMUSphinx, Snowboy, Picovoice).

Service discovery and activation will generally require online connectivity, so a cloud-based voice processing system does not imply additional restrictions. However, situations may occur where services are advertised by LAMs and there is no network available (e.g. underground car park). To cover such situations, simple offline voice dialog management can be considered, possibly sharing the technological solution found for wake-up word detection.

Acknowledgments

This work was sponsored by Via Verde Serviços, under the project related with Voice Proof of Concept.

References

[1] **A-to-Be LinkBeyond**, https://www.a-to-be.com/mobility-payments/a-to-be-linkbeyond-line/

[2] *A survey on dialogue systems: Recent advances and new frontiers.* **Chen, Hongshen, et al.** Acm Sigkdd Explorations Newsletter 19.2 (2017): 25-35

[3] *Deep neural networks for acoustic modelling in speech recognition.* **Hinton, Geoffrey, et al.** IEEE Signal processing magazine 29 (2012).

[4] *A guide to theory, algorithm, and system development.* **Huang, Xuedong, et al.** Spoken language processing: Prentice hall PTR, 2001.