An Open Source Standards-Compliant Voice Browser with Support for Multiple Language Understanding Implementations

Dirk Schnelle-Walka
Harman International, Germany
dirk.schnelle-walka@harman.com

Stefan Radomski
Technische Universität Darmstadt, Germany
radomski@tk.informatik.tu-darmstadt.de

Vikram Ramanarayanan, Patrick Lange & David Suendermann-Oeft
Educational Testing Service R&D, San Francisco, USA
<vramanarayanan,plange,suendermann-oeft>@ets.org

Abstract

There are several voice browser implementations for dialog systems, but none of them are both open-source and standards-compliant, while retaining compatibility with multiple implementations of system components such as natural language understanding (NLU) and dialog management modules. We present an standards-compliant open source solution that closes this gap while incorporating support for modern dialog concepts like flexible switching of user goals, custom grammar design and adaptivity to users. We show that our implementation can flexibly interface with two different NLU implementations to extract semantic information from user input and expose it to a VoiceXML application which integrates into a cloud-based dialog system that handles real user traffic.

1 Introduction

While there are various voice browser implementations available, they are either standards-compliant or available as open-source, but not both.Part of the causes of this deficit is that industrial implementations tend to be proprietary for commercial reasons, while academic implementations generally tend to focus on research examples that involve relatively small volumes of data. Bridging this gap is crucial to the continued development of the field and the integration of industrial and academic voice technology expertise. Standard compliance is crucial for interoperability among different systems developed by different parties, while being open-source is important for continued community development and progress, as well as widespread use of the technology.

The standards-compliant JVoiceXML software implementation (Schnelle-Walka et al., 2013; Prylipko et al., 2011) has attempted to bridge this gap. JVoiceXML is a VoiceXML interpreter written entirely in the Java programming language, supporting the VoiceXML 2.1 standard. The strength of JVoiceXML is its open architecture. Besides the support of Java APIs such as JSAPI and JTAPI, custom speech engines can easily be integrated. Examples are the text based platform and the MRCPv2 platform which are available with the distribution. It can be used within a telephony environment (Prylipko et al., 2011) but also without any telephony card as a standalone server.

This paper demonstrates an extension of the basic voice browser functionalities to incorporate support for modern dialog concepts like flexible switching between user goals, custom grammar design and adaptivity to users. In addition, we show that it can interface with different NLU implementations to extract semantic information from user input and make it available in VoiceXML applications, namely: (i) the Language Understanding Intelligent Service or LUIS (Williams et al., 2015) and (ii) the HALEF dialog system (Ramanarayanan et al., 2017a).

2 Reference Implementation I: LUIS

For the extension of VoiceXML to support state-of-the art natural language understanding capabilities, we make use of JVoiceXML's capability to support custom grammar types (in our case, `application/nlu`). The new type is made available to the interpreter via a dedicated factory that is loaded when the interpreter starts. This new type provides a component to parse any utterance with the help of any grammar document or an URI thereof into a semantic interpretation. This makes it possible to combine the new capability with any speech recognizer or textual input.

For automatic speech recognition (ASR) we employ the text implementation platform to capture strings as decoded input. Generally, this can be substituted by any unconstrained ASR. For the NLU engine we selected LUIS as a reference. Conceptually, this engine can be replaced by any other NLU engine to produce comparable output in terms of application, intent and associated entities. LUIS is based on *active learning* to enable developers utilize machine-learning based models without the need for large corpora. Its corpus grows based on real usage data (Williams et al., 2015).

From the grammar document, we use only its URI. Once the ASR returns a recognition hypothesis from the user's spoken input, the hypothesis will be passed to the grammar parser to determine its semantic interpretation. The grammar parser issues multiple requests to the LUIS server to check if any of the active grammars is able to derive meaning from the utterance, i.e. the intent is not *None* and at least one entity was recognized. Those with the highest confidence scores will be taken as the result of the interpretation and in turn create an ECMAScript object thereof. For example, the utterance *"I would like a large pizza with pepperoni"* (also see Section 3.1.6.1 of the VoiceXML standard) would be parsed as:

```
{
  nlu-application: "pizza",
  nlu-intent: "order-pizza",
  order-pizza: {
    number: "1",
    size: "large",
    topping:"pepperoni",
  }
}
```

This allows us to take advantage of VoiceXML as a scripting language with unconstrained user input for mixed initiative dialogs without the need for additional changes in the VoiceXML document and grammar design. The grammar with the new type can be used at any place where grammars are involved.

3 Reference Implementation II: The HALEF Dialog System

The modular and standards-compliant HALEF[1] multimodal dialog framework (Ramanarayanan et al., 2017a) is another example use-case that leverages the JVoiceXML voice browser platform. The HALEF dialog system has collected over 35.000 calls from people all over the world who interacted with multiple conversational applications (Ramanarayanan et al., 2017b). Design considerations in building the open-source HALEF system require standard compliance (in particular, with the VoiceXML 2.1 standard), the ability to process SIP traffic and support for multiple grammar standards, all of which are provided by the open-source JVoiceXML platform. In this case, for each dialog turn, the ASR returns the decoded recogniton hypothesis as a simple ECMAScript variable. We then perform NLU on this input by querying a webservice that invokes previously trained statistical models.

4 Conclusions

We have presented an open-source standards-compliant voice browser implementation and shown how it can flexibly interface with two different NLU implementations: LUIS and HALEF. Both approaches enable the reuse of established knowledge in creating standards compliant applications with VoiceXML for more modern dialog concepts as they were available when the standard was created. No additional changes in the VoiceXML document are required in the case of LUIS while HALEF only relies on an additional web service call.

References

Dmytro Prylipko, Dirk Schnelle-Walka, Spencer Lord, and Andreas Wendemuth. 2011. Zanzibar OpenIVR: an Open-Source Framework for Development of Spoken Dialog Systems. In *Proceedings of Text Speech and Dialog 2011*, August.

Vikram Ramanarayanan, David Suendermann-Oeft, Patrick Lange, Robert Mundkowsky, Alexei V Ivanov, Zhou Yu, Yao Qian, and Keelan Evanini. 2017a. Assembling the Jigsaw: How Multiple Open Standards Are Synergistically Combined in the HALEF Multimodal Dialog System. In *Multimodal Interaction with W3C Standards*, pages 295–310. Springer.

Vikram Ramanarayanan, David Suendermann-Oeft, Hillary Molloy, Eugene Tsuprun, Patrick Lange, and Keelan Evanini. 2017b. Crowdsourcing Multimodal Dialog Interactions: Lessons Learned from the HALEF Case. In *American Association of Artificial Intelligence (AAAI 2017) Workshop on Crowdsourcing, Deep Learning and Artificial Intelligence Agents*.

Dirk Schnelle-Walka, Stefan Radomski, and Max Mühlhäuser. 2013. JVoiceXML as a Modality Component in the W3C Multimodal Architecture. *Journal on Multimodal User Interfaces*, pages 183–194.

Jason D Williams, Eslam Kamal, Mokhtar Ashour, Hani Amr, Jessica Miller, and Geoff Zweig. 2015. Fast and easy language understanding for dialog systems with Microsoft Language Understanding Intelligent Service (LUIS). In *16th Annual Meeting of the Special Interest Group on Discourse and Dialogue. 2015.*, pages 159–161.

[1] http://halef.org