A spoken language interface for SSA/SDA based on modern speech processing technology

Dick Stottler, Sean Bartlett
Stottler Henke Associates, Inc.

ABSTRACT

Good spoken language interfaces are a natural, intuitive, and often efficient interface mechanism for a variety of different applications. Unfortunately, bad spoken language systems are frustrating and inefficient. Meanwhile, several large online companies have made huge investments in their spoken language interface technologies. People in their normal lives have become used to interacting with their personal assistants via voice and have come to expect very high levels of performance from spoken language interface systems. To the degree that the large investments by online companies can be easily and inexpensively leveraged for Space Situational Awareness (SSA)/Space Domain Awareness (SDA) applications, they represent a significant opportunity for greatly improving SSA/SDA applications and their corresponding user acceptance. We have been working with Amazon and others to create a spoken language interface for SSA/SDA, based on modern spoken language interface technology. We are working to understand the opportunities and limitations and have put together a demonstration that runs on any Alexa device with a screen, as well as on Internet-connected PCs, that runs through different SSA/SDA scenarios and involves queries, commands, and notifications. The main point of the demonstration is to demonstrate the spoken language interface to give conference attendees an understanding as to what is possible so they can start determining how they would best use spoken language interfaces for their SSA/SDA applications. We can also share our lessons learned and the highly efficient nature of the development of the spoken language interface.

There are different trade-offs between the different SLI development tools and these will be discussed. Some require Internet connections to operate and some do not, for example. The computational requirements for both development and real-time operation are typically modest. Updating the application requires about 30 seconds and the real-time, online component can be fielded with the computational power of a standard desktop without specialized processors. The robustness is inversely proportional to the breadth, diversity, and similarity of different intents of the discourse. We believe that SSA/SDA applications are very applicable because their discourse breadth is fairly narrow and predictable, consisting of the different kinds of plausible events that occur in space; applicable tactics, techniques, and procedures; the different spacecraft of interest, their subsystems and components; and indications, warnings, and cautions.

Our presentation will consist primarily of demonstrating the spoken language interface and letting anyone who would like to, try it for themselves. So far, the resulting spoken language interface appears fairly robust. A Machine Learning (ML) paradigm is used to match utterances to the user's intent. The user may say either the exact utterances associated with the intent or something similar. In the first case, intent recognition is near 100%. In the second it is still very high, over approximately 90%. And when an intent is not recognized, the utterance which was incorrectly understood can be quickly added to the correct intent list so that it correctly classifies the utterance in the future. Thus, the application can easily and naturally improve over time.

We believe this is a real educational opportunity for conference attendees as to what is possible and practical, inexpensively, with spoken language interfaces for SSA/SDA applications so that they can start thinking about the SSA/SDA user interface/user experience design and what role spoken language should play in that.

1. INTROUCTION/MOTIVATION/BENEFITS

Good spoken language interfaces (SLIs) are a natural, intuitive, and often efficient user interface mechanism for a variety of different applications. Perhaps most importantly, they provide one unifying User Interface (UI) for many diverse applications, tools, and features. This greatly facilitates training, reduces training time, and improves retention, especially over long absences. In addition, people are already used to interacting with automated personal assistants and devices via voice and have come to expect very high levels of performance from SLIs. If done well, SLIs greatly improve SSA/SDA applications and their corresponding user acceptance. An added benefit is that SLIs

leave the user's eyes and hands free. There is no need to look at a keyboard and watch what you are typing. Finally, several large companies have invested several billion dollars in developing SLI technology, including accompanying SLI development tools. For very little effort, this great investment can be leveraged.

2. BACKGROUND

Stottler Henke is involved in three efforts related to SLIs for SDA: SHERLOC, funded by AFRL, Virtual Space Assistant (VIRSA), part of the Hallmark program, funded by DARPA, and the Astronaut Agent, funded by NASA. In order to help space domain awareness (SDA) decision-makers identify and prioritize threats, the SHERLOC system is designed to perform level two data fusion of multi-source evidence to interpret the current situation based on knowledge of past cases, and produce real-time threat assessments. Different information sources, from Indications and Warnings (I&W) alerts to observable tracking data to communications traffic and collected intelligence, can most effectively contribute to situational understanding when considered together for the aggregate picture they present. SHERLOC provides a probabilistic framework for interpreting otherwise hidden interrelationships between multi-source input data, to inform the analysis of new situations. This approach has been applied in an experimental setting to use cases that include identifying co-orbital threats, detecting breakups, and predicting space weather effects on spacecraft (Jensen, Stottler & Belardi, 2021). In each of these applications, level two fusion builds on analytics from other providers, such as the capability to identify divergences from pattern of life, to help with sensemaking at the level where an aggregate SDA picture is being assembled. [Jensen, R., Stottler, R., & Belardi, C. (2021). Data Fusion of Historical Space Weather Outliers and Satellite Anomalies. To appear in Proceedings of the Advanced Maui Optical and Space Surveillance Technologies Conference 2021.]

Space has become an increasingly congested and contested environment as nations and commercial entities have become increasingly reliant on space-based capabilities. Enhanced space-domain awareness is necessary to manage these challenges, but the tools and processes available today will be insufficient for the future. For example, there is an abundance of information available to space operators, but it can be underutilized given the operational tempo and the complexity of decisions being made at vast distances and orbital speeds. As part of the DARPA Hallmark program, our objective was to develop a virtual space assistant (VIRSA) to enhance decision making in space operations by (1) filtering and searching heterogeneous data and (2) providing assistance to operators for tasks that are time-sensitive, data-intensive, recurring, or otherwise challenging; and by (3) reducing operator workload with an intuitive user interface. Our requirements necessitated that the virtual assistant be tailored to real-world operational tasks and that it be complementary to existing capabilities as well as intuitive, unobtrusive, quick, reliable, and convenient. [Ludwig, J., Presnell, B., & Stottler, R. (2021). Developing a Virtual Assistant for Space Operations. To appear in Proceedings of the Advanced Maui Optical and Space Surveillance Technologies Conference 2021.]

As NASA seeks to explore further from the Earth, reliance on real-time ground support will have to be greatly reduced and even eliminated in some cases. To this end, NASA is funding the development of the Astronaut Agent, to fulfill many of the responsibilities of ground personnel for manned space missions. The Astronaut Agent will play the role of the IVA, during Extra Vehicular Activities (EVAs), including monitoring spacesuit sensor values and life support systems, helping the EVA astronauts step through procedures, and automatically diagnosing space suit life support faults and providing relevant advice. Because of astronauts' unique activities, where both their eyes and hands are occupied (and their hands are in bulky gloves!), a spoken language interface is the ideal (and primary) interface with the astronauts. To monitor the life support system and sensor values, three independent diagnostic technologies are used: Machine Learning, Model-Based Reasoning, and Sensor Data Analysis/Trending. Results are intelligently combined, analyzed, and processed by the Intelligent Information Integration Module to provide appropriate text to the astronauts through the SLI. [Stottler, R., Ramachandran, S., Belardi, C., and Mandayam, R., (2020). On-board, Autonomous, Hybrid Spacecraft Subsystem Fault and Anomaly Detection, Diagnosis, and Recovery. AMOS , September 2020.].

3. MODERN SPOKEN LANGUAGE INTERFACE TOOLS

Modern SLIs are generally developed through one or more integrated tools that address each stage of the processing stage. The first stage uses an audio model to generate phonemes from the audio stream. Often, a generic model will work very well in regular office-like settings. If there is a large amount of noise, or accents are very pronounced,

hours of audio data may need to be collected to train a more specific audio model. In the next stage a language model converts the phonemes into the text words most likely spoken. It generally includes things like acronyms and special vocabulary and may produce multiple candidate strings, possibly with a certainty measure. In the final stage an ML converts the text string into an Intent and the software code associated with the Intent is executed, based on stored example strings of words associated with each Intent. For most applications a minor amount of effort may be used to define specialized vocabulary and acronyms and a rather larger effort is used to define the intents and enter example strings of words that a user would use to signal each intent. Adding the intents and associated strings is a very incremental process which incrementally increases the breadth and accuracy of the SLI.

When reviewing candidate SLI development tools, there are several limitations that should be considered.

- Internet Connectivity Often Required
- Narrow Scope / Minimal Number of Intents: The number of intents that can be successfully differentiated is in the low hundreds, and even this will be decreased if there are many of the same words that appear in the strings of words associated with many different intents
- Static vs. Dynamic Intents: Partially to mitigate the preceding limitation, some SLI tools allow intents to effectively be turned on and off, to keep small the number of intents active and capable of being selected and executed. This is useful if the context or recent events or user actions make the scope of reasonable intents much smaller than the overall total number of intents.
- Folding in outside context/information: This may be to limit the number of intents, as described above, or to adjust the likelihoods of competing intents based on recent events or actions.
- Word Recognition in presence of noise, accents, special vocabulary: Different tools will achieve different transcription accuracies than each other and in different noise environments and with different types of speakers.

These result in the following criteria when matching an SLI development tool to a specific application:

- Ease of development / Development Time
- Ability to utilize/interface to external code/software
- Turning Intents Off/On
- Receiving multiple intents with likelihoods
- Receiving Transcriptions along with Intents
- Internet-disconnected version available

To support multiple projects, we have examined a large number of SLI development tools, not all covered here. These included Alexa/Lex, Google Voice, Rhasspy, Almond, Sepia Framework, and MyCroft. Each is listed below with some of the pros and cons of each.

- Alexa/Lex
  - Good voice recognition
  - High-quality text-to-speech
  - Easy to use for prototyping
  - No offline version
  - Code only runs on Amazon servers, limited to a few programming languages
  - Very limited number of intents (250)
  - No intent ranking
  - Oriented toward casual applications
- Google Voice
  - Good voice recognition
  - High-quality text-to-speech
  - No offline version
  - Code only runs on Google servers, limited to a few programming languages
  - No intent ranking
  - Multiple types with different capabilities, difficult to assess which type is most suitable
- Rhasspy
  - Open source
  - Can run entirely offline
  - Can run on Raspberry PI
  - Simple to set up and use

  - Modular design allows swapping out various parts (Wake Word recognition, Speech to Text, etc.) with open-source components
  - Modules connected via MQTT, allowing easy integration with programs written in any language
  - Only runs on Debian and Docker has issues with audio input/output on some operating systems
  - Designed for home automation
- Almond
  - Open source
  - Compatible with interacting with new services
  - Allows you to generate natural language understanding training data in new domains from scratch
  - Can set up notifications
  - Difficult to set up, issues with sound on non-Linux systems
  - Operates on completely different concepts than other voice tools and has its own domain language, making transfer of code/ideas difficult
- Sepia Framework
  - Open source
  - Can run entirely offline
  - Can run on Raspberry PI
  - Designed for home automation
- MyCroft
  - Open source
  - Not free

4. SDA SLI PROTOTYPE DESCRIPTION

We engaged in this exercise, specifically to determine how quickly an SLI could be developed for an SDA application and how effective the resulting SLI would be. Specifically, we did NOT integrate the SLI with any underlying computational tools or databases. We are only demonstrating and evaluating the SLI component, specifically if the SLI correctly determines the proper intent and correctly receives and parses the parameters of those intents (e.g., RSO number or name). But nothing at all is done with the selected Intent or received parameters and they are not attached to any underlying tools, applications, capabilities or data, which, in any case, are mostly classified. The SDA SLI demonstrates spoken language understanding as it relates to:

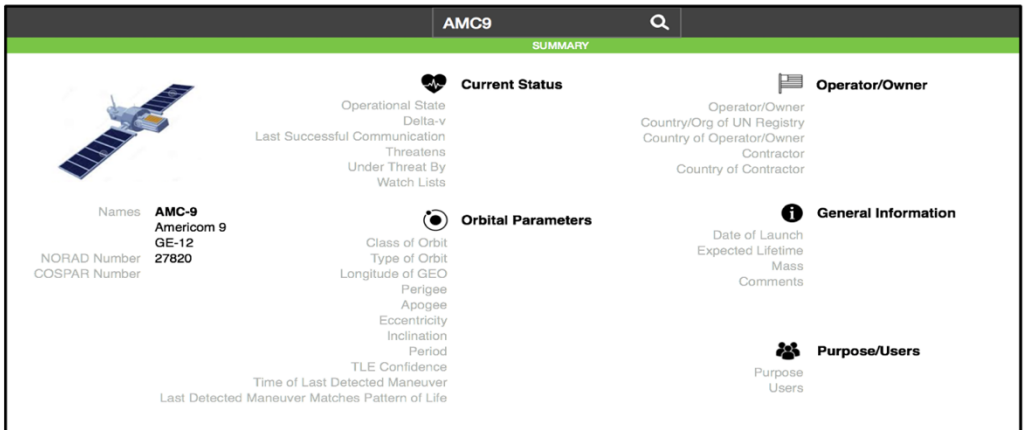
- Queries
  - Information Lookup
  - Utilizing underlying tools and computations
  - Keyword search across tools (warnings, alerts, chats, tasks, etc.)
- Commands
- Alerts / Follow-on Dialogs

Each of these is discussed below, in turn.

4.1 Queries for Information

Queries for information have to handle synonyms, of course. This primarily arises in two ways: synonyms for the word “satellite” (RSO, Object, Sat, Bird, Asset, Resident Space Object) and synonyms for specific satellites, e.g., 44485, AEROCUBE 10A, AC 10A, and JIMSAT, which all refer to the same object. Example queries (and one response) are sown below:

- Show me the missile defense satellites in GEO
- What other assets could support tasks assigned to AMC-9?
- Display the current status for RSO 12345
- Is USA 270 GEO?
- Who owns AM-33?
- What is the use of TIANTONG-1 1?
- Show me the check list for …
- Show me the Summary for AMC-9



4.2 Queries Utilizing Underlying Tools/ Computations

Perhaps one of the greatest benefits of an SLI is that it can provide a common user interface and interaction mechanism for diverse toolsets and capabilities. Instead of having to learn the UIs for several different tools and to learn to navigate to the different features, capabilities, and computations, the user can simply verbally ask for what they would like in order to invoke the underlying computations and capabilities. There will be different concepts embedded and implicit in the different tools that the SLI must be aware of and parse appropriately for the different tools. These concepts include:

- Past and Future Time
- Distances
- Ground locations
- Locations in space
- RSOs
- Fuzzy Terms (e.g., “near,” “in the vicinity of,” or “close”)
- Pairwise Relationships (e.g., RSO-to-RSO, Ground-to-RSO, and RSO-to-Any RSO)

Examples of queries utilizing underlying tools/ computations are:

- What ground assets can view TDRS-12 now <or in 12 hours>?
- What RSOs can POGO see right now?
- What RSOs were in the vicinity of TDRS-12 since its last communication?
- What is the space weather situation <near TDRS-12>?
- What is the probability that space weather caused the anomaly?
- Is this anomaly similar to a past case?
- When will RSO 12345 be over North Korea?
- How likely was the maneuver on purpose?
- When is the next conjunction for RSO 12345?

4.3 Keyword search across tools

A common frustration that operators expressed was their inability to remember in which of the many tools that they used was information previously displayed or discussed. A very useful capability was a simple keyword search across tools (Warnings, Alerts, Chats, Tasks, etc.). Examples include:

- Search for TIANTONG-1 1 in warnings and chat
- Find the words “Debris,” “AMC-9,” “vicinity” in all tools
- Search for “Unexpected Maneuvers” in alerts and chat
- Show me any alerts for Starlink-57

4.4 Commands

Another set of capabilities for which an SLI can provide a UI is invoking commands. Because the operators have to handle multiple issues, simultaneously, each with different time scales, and in order to prevent anything from "falling through the cracks," the most common commands were to create reminders or alerts at specific times or events or at a specific amount of time before an event was expected to occur. Examples include:

- Notify me when RSO 12345 maneuvers
- Remind me in 45 minutes to check on TIANTONG-1 1
- Remind me 5 minutes before then to look at the close conjunction checklist
- Add 42065 to the HVA Watchlist

4.5 Alerts / Follow-on Dialogs

Although alerts, as outputs to the user, were outside the scope of our SDA SLI prototype, a very common input is a dialog that follows up on an alert. Alerts would be generated by third-party tools and sent through or repeated by the SLI. Typically, a user might issue a command (or make a query) regarding the most recently spoken alert. Examples of alerts and the follow-on request are shown below:

- Russian-owned 12345 maneuvered and has a close approach to USA-270 in 3 hours and 5 minutes
  - Remind me 15 minutes before then to look at the close approach checklist
- Solar flare will produce space weather effects in 2 hours and 35 minutes
  - Notify me 20 minutes before then
  - Or Find relevant COAs
- Launch Detected in North Korea 30 minutes ago and has close approach to USA 270 in 67 minutes
  - Notify me 20 minutes before then

5 DEMONSTRATION DESCRIPTION

The SDA SLI prototype is invoked from any Alexa device, by saying, "Alexa, Open Space Domain Awareness." Thereafter, "Alexa" does not have to be repeated, unless too much time has passed. Whether or not Alexa is still listening (so that the "Alexa" wake word does not need to be said) is indicated by a blue line at the bottom. If it is not present, then the SLI utterance must be preceded by "Alexa." An irritating "feature" is that if even more time elapses, then Alexa automatically leaves the Space Domain Awareness skill and it needs to be re-opened. We implemented 60 different Intents corresponding to 60 different capabilities, so we were representing a fairly robust toolset. However, all of the SLI's responses are canned, it is NOT connected to any underlying capabilities, but the responses indicate that the correct Intent was selected. We also only included a few dozen real RSOs and their synonyms instead of the few thousand that would normally be required. However, for example, when using the 5-digit catalog number, it was correctly parsed to the correct number almost always perfectly.

6 RESULTS

Our main objective was to determine how quickly an SLI could be developed and how effective that SLI would be. To fully develop the SLI for the sixty Intents took about thirty person-hours. We were partly able to accomplish so much in so little time because we were already familiar with the Alexa/Lex development tools. However, the learning curve for those tools is not high. The resulting SLI accuracy with which it recognizes the correct Intent is very, very high when the exact phraseology is used (i.e., if the same string of words is used as one of the example utterances associated with the Intent). In fact, it is nearly 100%. The achieved user accuracy, which usually involves not saying the exact example utterances associated with an Intent is still very good, over 90%, if most variations are reasonably captured in the alternative utterances list. And since additional utterances can be easily added (processing time for adding new Intents or adding new utterances for an existing Intent is about 30 seconds), the user accuracy can be almost arbitrarily improved to any level desired based on user feedback, limited only to many of the same words appearing in the lists associated with different Intents.

7 LESSONS LEARNED

Though our work developing SLIs for several applications, we have learned several lessons:

- Depending on the needs of the application, the noise environment, and the speakers, sophisticated SLIs can be developed quickly with low effort. This is especially true if one of the rapid development tools can be used, which will tend to be true if the number of total Intents is limited and there is little overlap in the words between different intents.

- Several different SLI-Tools exist with their own strengths and weaknesses. There is a wide array of available SLI-development tools, each optimized for different types of applications and offering very different levels of sophistication. Not surprisingly, the greater the degree of sophistication, the longer the development time.

- No one tool is best for all applications—different tools will be better suited for different applications. With one exception (see the next bullet), it is likely that a good choice of an SLI development tool can be found for your application.

- Internet-connectivity (or not) is a major differentiator, for now. If your application cannot be connected to the Internet, that will greatly limit your choices. Some choices still exist so it may still be possible to find an appropriate SLI development tool for your Internet-disconnected application, but the word recognition accuracy tends to be less for these tools.

- Alexa/Lex is good for very rapid prototyping and may be acceptable for fielding for some applications. Development time with Alexa/Lex is so minimal that it is often a good idea to develop an initial prototype with it and get some user feedback, so as to determine the actual requirements of a finalized product. Depending on the domain/application, this may require changing SLI development tools, but the chance to get user feedback from an early, low-effort prototype was probably worth it.

- Procedure-aiding and other applications with a lot of overlapping words between intents require dynamic intents. Because Alexa/Lex does have dynamic Intents, it is not good as the number of Intents stretches into the hundreds (and it has a hard limit of 250), especially if there are a lot of overlapping words between different Intents. This is often true of procedures which can easily have hundreds or thousands of steps and tend to use a lot of the same verbs and nouns. In cases like this, dynamic Intents are critical to keep the number of choices the SLI system has to choose between down to a manageable number. E.g., halfway through a long procedure, there is no reason to have Intents active associated with steps very early or very late in the procedure.

- SLIs, if done well, achieve high user acceptance and greatly reduce training time. Refresher training becomes less important.

- If done poorly, they cause frustration. We have all experienced the frustration when our spoken language interface devices require several attempts or can never figure out our intent. An important part of user training is a clear and easy description of what the SLI application can and cannot do.

8 FUTURE WORK

We are currently pursuing several avenues to further develop this work. Of course, the most obvious is to connect a similar Internet-disconnected SLI to the required underlying SDA tools and data and field it. Additionally, further work needs to be done on the underlying tools and computations. In our case, this would correspond to further work on SHERLOC and VIRSA. As these systems are further fielded, we would incorporate the feedback and other lessons from their operational use into both the SDA tools and the SLI. And, for NASA, we continue to further develop both the Astronaut Agent SLI and its underlying reasoning capabilities.

9 CONCLUSIONS

As described throughout this paper, there are several conclusions:

- Good spoken language interfaces are a natural, intuitive, and efficient user interface mechanism for SDA applications
- SLIs can provide one UI for a diverse set of SLI features and tools
- SLIs allow eyes-free operation, allowing user to keep their concentration on the SDA displays
- SLIs greatly enhance SSA/SDA applications and their corresponding user acceptance

- SLIs greatly reduce training efforts
- SLIs can be developed with minimal development effort

10 REFERENCES

[1] Jensen, R., Stottler, R., & Belardi, C. (2021). *Data Fusion of Historical Space Weather Outliers and Satellite Anomalies*. To appear in Proceedings of the Advanced Maui Optical and Space Surveillance Technologies Conference 2021.

[2] Ludwig, J., Presnell, B., & Stottler, R. (2021). *Developing a Virtual Assistant for Space Operations*. To appear in Proceedings of the Advanced Maui Optical and Space Surveillance Technologies Conference 2021.

[3] Stottler, R., Ramachandran, S., Belardi, C., and Mandayam, R., (2020) *On-board, Autonomous, Hybrid Spacecraft Subsystem Fault and Anomaly Detection, Diagnosis, and Recovery*. AMOS, September 2020.