Der Git-Spickzettel – für den schnellen Überblick!

windows .developer

windows .developer

3.2019

Deutschland 9,80 €
Österreich 10,80 €
Schweiz 19,50 sFr

www.windowsdeveloper.de

Die neue Git-vasion

Go Git!
Git erobert die Entwicklerwelt

Kraftvolles Git
Git in der Windows PowerShell und PowerShell Core

Einfach den richtigen Flow finden
Branching-Modelle und Workflows für SCM-Systeme

4 195809 809803 03

Ganz ir-relationale Datenbanken
Cosmos DB eigenen Projekten einsetzen
▶ 62

Ein Pizza-Bot(e) für alle Fälle
Sprachgestützte Unterhaltungen mit Bots
▶ 50

Microsoft Power BI Embedded
Berichte in die eigene Anwendung einbinden
▶ 44

Sprachgestützte Unterhaltungen mit Bots

Ein Pizza-Bot(e) für alle Fälle

Vom Sprachbrowser zum Bot: Heute können Computer nicht mehr nur Telefongespräche auswerten, sondern so einiges mehr in Sachen Kommunikation. Was können Bots heute leisten und wie entwickelt man einen eigenen Bot mit dem Microsoft Bot Framework?

von Friederike Geissler

„Willkommen in der Telefonwarteschleife der Pizza GmbH. Interessieren Sie sich für eine Pizzabestellung, sagen Sie ‚Pizzabestellung'. Falls Sie Ihre Bestellung ändern möchten, sagen Sie ‚Bestellung'." Eine solche Dialogführung durch einen sogenannten Sprachbrowser ist der evolutionäre Vorgänger eines Bots. Bots (von engl. robot) sind Computerprogramme, die automatisiert sich wiederholende Aufgaben abarbeiten, ohne auf eine Interaktion mit einem menschlichen Benutzer angewiesen zu sein. Wo Sprachbrowser ausschließlich auf Telefongesprächauswertungen angewiesen sind, können Bots allerdings mehr – viel mehr.

Bots verarbeiten und erkennen Sprache, Bilder, Dateien und Daten. Sie telefonieren, schreiben E-Mails sowie SMS und tauschen Nachrichten in sozialen Netzwerken aus. Außerdem planen sie Termine, nehmen Zahlungen und Feedback an. Sie erinnern an bevorstehende Ereignisse oder informieren über Neuigkeiten aus abonnierten Interessensgebieten. Sie reagieren stets flexibel auf die Stimmung des Benutzers und schätzen seine Intention ein. Der Bot ist die Weiterentwicklung der klassischen Warteschleife. So kann ein Bot via Skype, Facebook Messenger, Telefon oder andere Kanäle Bestellungen oder Ähnliches aufnehmen. Ein typischer Anwendungsfall: Ein Kunde möchte eine Pizza bestellen und schreibt dem Pizza-Bot die Nachricht: „Habe Hunger. Jetzt!". Der Bot reagiert daraufhin mit einer Reihe von Fragen (z. B. „Welchen Durchmesser soll die Pizza haben?") oder Vorschlägen (etwa: „Wie wäre es mit unserer Tagespizza?") und leitet zum Schluss einen Bezahlvorgang ein. Ein Pizza-Bot kann potenziell den Kalender und den Aufenthaltsort des Benutzers prüfen,

um die Bestellung zu verifizieren und den Bestellvorgang zu vereinfachen. Hierfür ist maßgeblich, dass der Benutzer vorab der Nutzung seiner persönlichen Informationen zugestimmt hat, sodass der Bot sie suchen und verarbeiten darf.

Genauigkeit der Spracherkennung und Authentifizierung

Ein messbares Kriterium von erfolgreicher Spracherkennung bei Bots ist die Präzision, also das Verhältnis von korrekt erkannten Äußerungen zu falsch erkannten. Aktuell liegt die Präzision weltweit führender Anbieter bei 91-95 Prozent. Beim chinesischen Suchanbieter Baidu sind 99 Prozent in Umgebungen mit geringen Hintergrundstörgeräuschen angestrebt. Andrew Ng, Chefentwickler von Baidu, sagt dazu: „Sobald die Präzision von Spracherkennung […] 99 % erreicht, werden wir Spracherkennung nicht mehr kaum, sondern täglich verwenden“ [1]. Diese Präzision hat Nuance, einer der ersten Anbieter von Diktiersoftware, laut eigener Aussage mit seiner Dragon-Spracherkennung bereits erreicht [2].

Spracherkennungsdienste filtern die Stimme des Benutzers aus anderen Stimmen und Störgeräuschen, die das Mikrofon mit aufnimmt. Sobald die Stimmdaten extrahiert sind, werden sie analysiert. Die Einmaligkeit von menschlichen Stimmmustern wird softwaretechnisch teils in Authentifizierungsprozessen genutzt. Wie verschiedene Studien und Praxisbeispiele belegen, ist Stimmerkennung allein jedoch nicht als Authentifizierungsmethode geeignet, da sie gegenüber Replay-Attacken und manipulierten Audioaufnahmen anfällig ist [3], [4]. Einen interessanten Lösungsansatz bieten Huan Feng, Paul Scerri und Katia Sycara mit ihrer Software zur Auswertung biometrischer Daten an. Diese bezieht nämlich tragbare Geräte zur Authentifizierung ein. Wenn der Benutzer eine Smartwatch trägt, ist es so möglich, mit Pulsmuster und Körpervibrationsaufzeichnung sicherzugehen, dass das, was gesagt wird, auch vom Benutzer stammt [5] und nicht z. B. von einem Lautsprecher abgespielt wird.

Vorteile für die Benutzer

An dem Pizzabeispiel ist gut ersichtlich, welchen Mehrwert Bots für ihre Benutzer liefern: Im digitalen Zeitalter, in dem mehr und mehr Menschen ununterbrochen online sind, bietet ein Bot eine 24/7-Erreichbarkeit. Anders als manche Kundenservicehotlines ist er kostenlos erreichbar und es gibt keine langen Wartezeiten, um etwa den nächsten freien Kundenberater zu sprechen. Die Nutzung auf mobilen Endgeräten ist zudem optimal, um Informationen standortabhängig zu filtern. Stellen wir uns vor, dass der Kunde dem Bot auf die Frage „Wo bist du?“ eine GPS-Freigabe für die nächste Stunde schickt. Egal, wo der Kunde mit seinem GPS-fähigen Endgerät dann in der nächsten Stunde hingeht, der Pizzabote kann ihn geocachen, um die Pizza auszuliefern.

Ein weiterer Vorteil besteht darin, dass multilinguale Benutzer potenziell die Option haben, sich mit dem Bot in der eigenen Muttersprache zu unterhalten. So wäre es für den Kunden möglich, eine Pizza auf Spanisch zu ordern, selbst wenn die Mitarbeiter des Pizzaunternehmens nur Englisch verstünden. Denn die meisten Bot Frameworks unterstützen die Einbindung von Übersetzungsdiensten wie Google Translate. Die Übersetzungsqualität ist allerdings abhängig von den verwendeten Sprachen. Viel ausgeprägter als das europäische ist das angloamerikanische Bot-Angebot, was darauf zurückzuführen ist, dass die Global Player auf diesem Gebiet überwiegend aus den USA stammen. Auch Benutzer mit motorischen oder visuellen Handicaps profitieren von einem Bot mit Sprachunterstützung und Hands-off-Prinzip. Bots können so programmiert sein, dass die Benutzung einer Tastatur überflüssig ist. Sprachassistenten wie Google Assistant, Siri, Cortana und Alexa sind nichts anderes als Bots, die ihre Navigation einzig über die Stimme auslegen.

Vorteile für Unternehmen

Einen gut programmierten und kontinuierlich gewarteten Bot einzusetzen, kann die Kundenservicehotline gänzlich ersetzen und bietet daher personelle und finanzielle Einsparpotenziale für Unternehmen. Gut programmiert heißt in diesem Fall, dass die Bots auf verschiedene Kanäle und die Einbindung externer Dienste angepasst sind. Der Bot muss eine Anzahl von Musterkonversationen, die ein Kunde mit ihm führen würde, kennen. Zudem ist es wichtig, ihm eine Trainingsphase zuzugestehen, in der er mit echten Kundenanfragen konfrontiert wird. Wie die Fiducia GAD IT AG, ein deutsches Dienstleistungsunternehmen aus dem Bereich Banking, mit ihrem Chatbot-Prototyp nachwies, ist besonders in oft auftretenden Kundenanliegen der Einsatz eines Chatbots effizienter, weil der Korrekturbedarf geringer ausfällt [6]. Auch die Studie von Steven Okamoto, Kassem Fawaz und Kang Shin zeigt ein hohes Einsparpotenzial bei der Einführung in den Geschäftsablauf von bis zu 46 Prozent Leistungssteigerung [7]. Kritiker halten allerdings dagegen, dass die beste Kundenberatung durch Personen stattfindet, die nicht in Mustern denken [8].

Die wachsende Zahl von Bot-Start-ups zeigt jedoch einen großen Bedarf an dieser Technologie in verschiedensten Branchen und Bereichen auf – mit unterschiedlichen Spezialisierungsgraden. Andrew Mortensen, Mitgründer des Amy-Bots von x.ai, meint hierzu, Ziel sei es nicht, halbversiert auf allen Themengebieten, sondern gut auf einem speziellen zu sein [9]. Es gibt Start-ups, die sich auf Investmentbankingberatung (Stichwort Fintech Start-ups) konzentrieren, und solche, die ausschließlich einzelne Aufgaben wie etwa die Organisation von Terminen lösen.

Ein Beispiel aus der aktuellen Forschung

2018 wurde von der innobis AG, einer IT- und SAP-Beratung für Banken, ein Prototyp im Rahmen einer wissenschaftlichen Abschlussarbeit realisiert und ausgewertet, um die Möglichkeiten dieser Technologie zu

erforschen [10]. Der entstandene Bot löst für das Unternehmen die Aufgabe, Daten eines Benutzers für ein Fördermittelformular des Europäischen Sozialfonds abzufragen. Der Prototyp ist in C# programmiert und basiert auf dem Microsoft Bot Framework. Dieses ermöglicht – wie viele andere Bot Frameworks – ein kostenloses Hosting (in diesem Fall in Azure). Folgende Anforderungen galten für den Prototyp: Spracherkennung nach dem Hands-off-Prinzip, Multilingualität, plattformübergreifende Programmierung, die Einbindung eines AI-Service sowie die Möglichkeit, Dateien, z. B. Quittungen, hochzuladen.

Nach einer ersten Prüfung des Vorhabens zeigte sich einerseits, dass die Anforderungen realisierbar sind. Im Usability-Test wurde jedoch andererseits auch deutlich, dass Benutzer zögern, einem Bot persönliche Daten mitzuteilen. Ein weiteres Problem war die geplante Fusion des Bots in eine sogenannte Cortana-Fähigkeit. Cortana-Fähigkeiten sind vergleichbar mit Erweiterungen von Cortana durch Drittanbieter. So gibt es in den USA etwa die Cortana-Fähigkeit „Domino's Pizza", die es einem Cortana-Benutzer ermöglicht, über den Dialog mit Cortana Pizza bei der Firma Domino's zu bestellen. Das Problem mit der Integration des Bots als Cortana-Fähigkeit ist, dass sie für Geräte mit deutschen Spracheinstellungen noch nicht freigegeben sind.

Für den Prototyp wurde der Microsoft AI Service LUIS (Language Understanding Intelligence Service) eingesetzt. LUIS ist über ein API vollautomatisch nutzbar. Wie viele andere auf Entitäten und Intentionen basierende Web Services auf dem Gebiet AI scheitert er allerdings bei unbekannten Äußerungen in der Analyse. Ein großes Potenzial bietet jedoch die Ansammlung von Kundenanliegen, die ein Bot auswertet. Wichtig ist dabei die fortwährende Kontrolle und Wartung durch den Bot-Entwickler, weil ansonsten gegebenenfalls falsche Zusammenhänge erlernt werden. Dies zeigte sich in Googles Prototyp „Tay", dessen Benutzern es gelang, dem Bot rassistische und beleidigende Äußerungen anzutrainieren [11].

Die Erfahrungen aus der Praxis zeigen, dass die Einsatzmöglichkeiten von Bots für Unternehmen durchaus lukrativ sein können, selbst wenn die erhobenen Daten nicht direkt von den Benutzern kommen.

Das Leistungsspektrum eines Bots

Dem Einsatzgebiet von Bots sind quasi keine Grenzen gesetzt. Einige Bot Frameworks wie SILVIA unterstützen Avatare, um dem Benutzer ein möglichst realistisches Kommunikationsgefühl zu vermitteln und nonverbale Gesten wie ein Lächeln, Stirnrunzeln oder Schulterzucken zum Ausdruck zu bringen. Das ist besonders im Zusammenhang mit emotionaler Intelligenz sinnvoll. Bei der Auswertung von emotionaler Intelligenz wird erhoben, wie aufnahmefähig ein Benutzer in einer bestimmten Gefühlssituation ist. Der Bot reagiert situationsabhängig und unterschiedlich auf Stress, Wut oder Humor.

Ein oft angeführtes Beispiel für das Leistungsspektrum von Bots ist die Hotel-Flug-Buchung. In diesem Fall werden Reservierungs- und Buchungs-E-Mails ausgewertet und multimediale Inhalte als Information zu Hotels in der Nähe des Benutzers eingeblendet [12]. Proaktive Nachrichten sind hier eine Möglichkeit, den Bot selbst die Konversation auf ein bestimmtes Ereignis hin eröffnen zu lassen, z. B. „Lieber Benutzer, an deinem aktuellen Standort hat gerade ein Wellnesshotel eröffnet. Soll ich dir ein Zimmer reservieren?".

Ein Blick in die Programmierung

Da Bots untereinander kompatibel programmierbar sind und je nach Anbieter Schnittstellen zum Sprachassistenten des Anbieters haben, ergeben sich Potenziale in der Steuerung von externer Hardware. Dies zeigt beispielsweise das Projekt „Internet of Stranger Things" von Pete Brown, bei dem eine Lichterkette zu Demonstrationszwecken über einen Bot gesteuert wird [13]. Die Möglichkeiten der Bot-Einbindung in stimmgesteuerte Hardware schließen auch intelligente Beleuchtungs- und Hausautomatisierungssysteme wie Philips Hue bis hin zu personalisierbaren Robotern wie Jibo [14] ein; es geht also nicht nur um die Anbindung von Stand-alone-Lautsprechern wie Alexa Echo.

Vorbereitungen für die Bot-Entwicklung

Um einen Bot zu entwickeln, ist der leichteste Weg die Registrierung bei einem Bot-Framework-Anbieter. Dieser stellt einen (oder mehrere) Server, der die Benutzeranfragen und Bot-Antworten über Internetverbindung verarbeitet. Dafür ist zunächst ein Benutzeraccount bei einem der Anbieter zu eröffnen. Soll zum Beispiel für das Microsoft Bot Framework entwickelt werden, muss ein Microsoft-Account erstellt werden, mit dem dann ein Microsoft-Azure-Account zu erstellen ist. Oft wird bei der Registrierung eine gültige Zahlungsinformation verlangt, z. B. eine Kreditkartennummer. Anschließend ist zumeist weitere Software zu installieren, um den Bot lokal debuggen zu können. Dazu gehört eine Entwicklungsumgebung, die vom Framework verlangt wird (z. B. Visual Studio oder ein Extratool wie der Bot-Framework-Emulator – beide sind für die Nutzung des Microsoft Bot Frameworks nötig). Kenntnisse von Tunneling-Software wie ngrok können sich ebenfalls als nützlich erweisen, falls der Bot hinter einem Unternehmens-Proxy laufen soll.

Sind diese Schritte abgeschlossen, sind die Grundzutaten für einen (Pizza-)Bot vorhanden. Es ist jedoch möglich, weitere Frameworks einzubinden, etwa ein AI-Framework, damit der Bot anhand der Konversation intelligent reagiert. In Bezug auf den Programmieraufwand ist der effizienteste Weg, Frameworks des gleichen Anbieters zu nutzen. Daneben besteht noch die Möglichkeit, Drittanbieterdienste über eine programmtechnische Schnittstelle einzubinden. Für jedes Framework, das zusätzlich angeboten werden soll, ist eine Registrierung erforderlich und gegebenenfalls eine weitere für die programmatische Schnittstelle.

Oft ist es möglich, den Bot auf verschiedenen sogenannten Kanälen zu vertreiben. Ein Sprachassistent wäre

ein solcher Kanal, aber auch Kommunikationsnetzwerke wie Skype, Twitter, Slack, Microsoft Office, Telegram oder Facebook Messenger. Ist eine Veröffentlichung des Bots auf einem oder mehreren Kanälen (welche verfügbar sind, ist anbieterabhängig) geplant, dann ist für das Entwickeln und Testen hier ein Account nötig.

Drei Anwendungsoptionen des Microsoft Bot Frameworks

Im Folgenden werden drei Anwendungsoptionen des Microsoft Bot Frameworks vorgestellt, die beim bereits beschriebenen Prototyp für die innobis Verwendung fanden. In allen Fällen geht es darum, dem Bot das Dialogmuster, nach dem er arbeiten soll, zur Verfügung zu stellen:

- FormFlow, ein JSON-Format für einen geregelten und vordefinierten Ablauf (das sich vor allem eignet, um formularähnliche Daten abzufragen),
- DirectLine, eine programmatische Schnittstelle zur Einbindung von Drittanbieterdiensten (z. B. eines Sprachassistenten), und
- LUIS-Service, ein AI-Framework zur Einbindung des Language Understanding Intelligence Service von Microsoft.

Um einem Bot den Dialog per FormFlow vorzugeben, muss eine JSON-Datei mit vordefinierten Fragen und Antworten erstellt werden, die später in den Bot geladen wird. Bei FormFlow gibt es verschiedene Implementierungsmöglichkeiten. Es ist möglich, eine Datenhaltungsklasse mit Attributen für jedes vorgeschriebene Eingabefeld zu implementieren oder auch, diese aus einer JSON-Datei generieren zu lassen, die

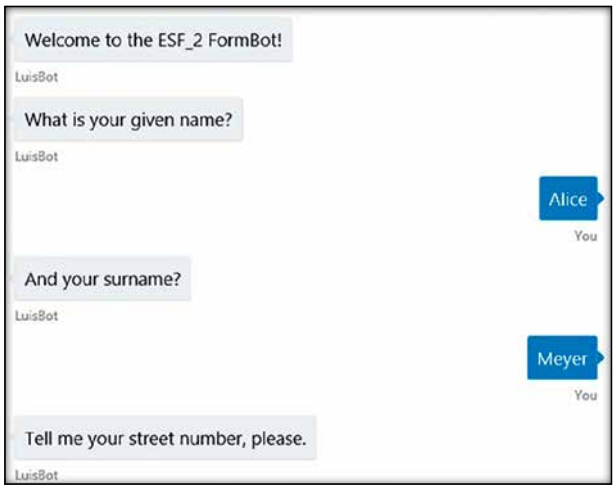

Abb. 1: In C# aufgerufenes JSON-Skript (Screenshot)

ein bestimmtes Muster erfüllt (siehe Listing 1 mit einem Auszug aus der FormFlow-Datei, die im innobis-Prototyp Verwendung fand). In den Properties sind die abzufragenden Werte definiert, wobei die Frage mithilfe eines &-Platzhalters für den Property-Namen ergänzt ist. Erforderliche Eingaben sind zu Beginn zu kennzeichnen. Eingebettete reguläre Ausdrücke bieten außerdem die Möglichkeit, die Benutzereingabe zu prüfen. Darüber hinaus ist es möglich, Validierungsfunktionen in JSON zu benennen. Der Aufruf des JSON-Skripts kann über Dateilesezugriffe in C# wie in **Abbildung 1** gezeigt erfolgen.

Es ist hierbei möglich, Validierungs-Regexe zu definieren, numerische Intervalle für die Validierung mitzureichen oder die Namen von Validierungsmethoden mitzugeben. Zudem ist es notwendig, je nach verwendeter Sprache des Benutzers unterschiedliche JSON-Dialoge anzulegen (oder diese mit einem Übersetzungsframework übersetzen zu lassen).

Listing 1: Codeauszug aus der JSON-Datei, die den FormFlow-Dialog generiert

```
{
  "References": [ "LuisBot.dll" ],
  "Imports": [ "LuisBot.Resource" ],
  "type": "object",
  "Templates": {
    "NotUnderstood": {
      "Patterns": [ "I didn't understand: \"{0}\".",
            "Try again, I don't know about \"{0}\"." ]
    }
  },
  "properties": {

    "Vorname": {
      "Before": [
        {
          "Message": [
            "Welcome to the ESF_2 FormBot!"
          ]
        }
      ],
      "Prompt": {
        "Patterns": [ "What is your given name?" ]
      },

      "type": [
        "string",
        "null"
      ]

    },
    "Nachname": {

      "Prompt": {
        "Patterns": [ "And your surname?" ]
      },

      "type": [
        "string",
        "null"
      ]

    },
    "Hausnummer": {

      "Prompt": {
        "Patterns": [ "What is your street number?" ]
      },

      "type": [
        "string",
        "null"
      ]

    },
...
```

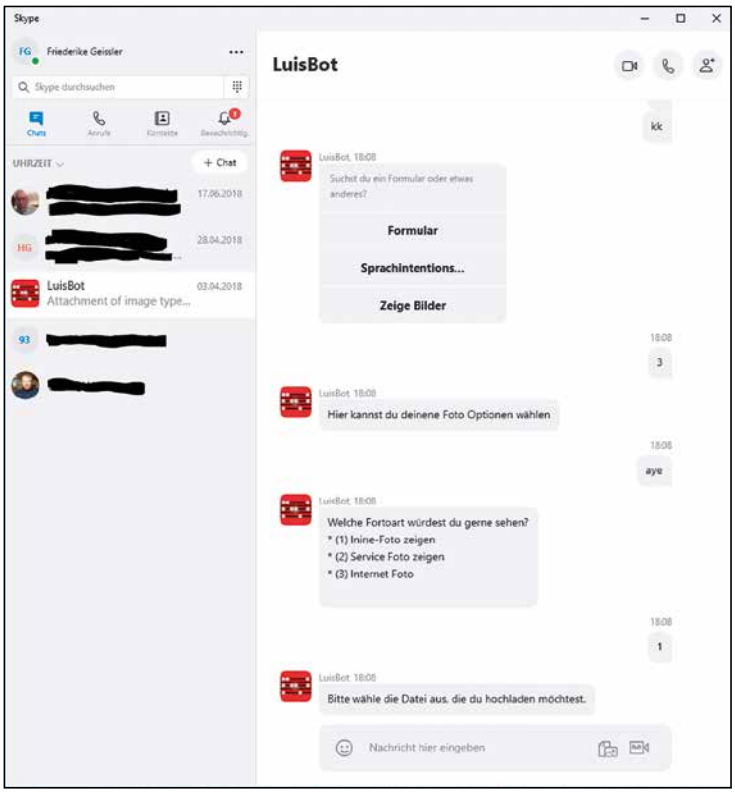

Abb. 2: Skype-Dialog zwischen Bot und Benutzer (Screenshot)

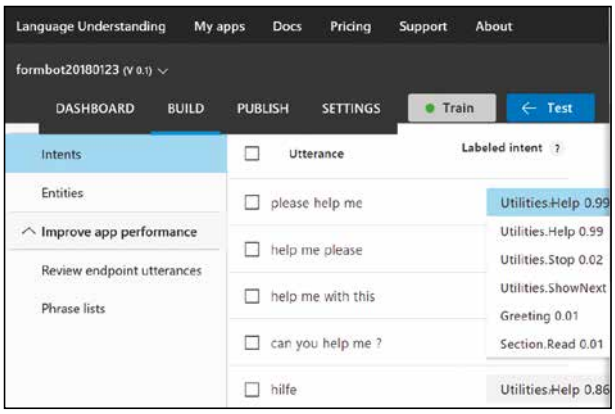

Abb. 3: Korrekturmöglichkeiten im LUIS.ai-Webportal (Screenshot)

Abhängig davon, welchen Kanal der Benutzer wählt, um mit dem Bot in Kontakt zu treten, bieten sich zusätzliche spezielle Eigenschaften des Kanals. Mit einer Text-zu-Sprache-Generierung, die ein integraler Bestandteil von sprachfähigen Bots ist, gibt es die Option, einen Telefondialog in der favorisierten Sprache des Benutzers zu generieren. **Abbildung 2** zeigt ein Beispiel eines Skype-Dialogs zwischen Bot und Benutzer.

Eine DirectLine-Verbindung wird – oberflächlich betrachtet – als WebSocket-Verbindung oder als HTTP-GET-Anfrage des Clients mittels eines Handshake-Verfahrens aufgebaut. Um sie zu nutzen, ist ein DirectLine-Schlüssel für den Bot zu generieren. Für vertiefende Informationen eignet sich die Microsoft-Dokumentation [15] oder Robert Osbornes Blog, in dem er beschreibt, wie einer Alexa-Fähigkeit eine DirectLine-Schnittstelle hinzufügt wird [16].

DirectLine kann im Microsoft Bot Framework genutzt werden, um einen LUIS Service anzubinden. Das ist ein AI-Framework. Der Wunsch, AI-Frameworks einbinden zu wollen, wird anhand von Benutzeranfragen wie „Ich möchte die Unternehmensangaben im ESF_2-Formular ausfüllen“ deutlich. Es gibt hierbei zahlreiche Möglichkeiten, wie ein Benutzer diese Anfrage (synonym) formulieren könnte, aber jedes Mal ist es die Aufgabe des Bots, anhand der Worte zu erkennen, dass er mit einem „Willkommen“ antworten soll.

Der LUIS Service ermöglicht es dem Bot, auf möglichst natürliche Art und Weise mit Menschen zu kommunizieren. Er wird von Microsoft als Cloud-Lösung auf einer AI-Plattform angeboten. Der Bot analysiert die gesprochene Sprache und leitet die Anfrage dann an LUIS weiter. LUIS analysiert die Frage und ermittelt die Intention. Um LUIS Services einzubinden, muss die Intention des Benutzers auf der AI-Plattform bekannt sein. Darüber hinaus helfen zusätzliche Parameter, die die Intention erläutern. Die Intention eines Benutzers wäre zum Beispiel, eine Pizza zu bestellen. Ein dazugehöriger Parameter wäre dann etwa die Größe der Pizza. Die Schwierigkeit liegt in diesem Fall darin, dass Menschen ihren Wunsch nach einer Pizza in einer bestimmten Größe unterschiedlich ausdrücken. Die Aufgabe von LUIS ist es daher, die Intention und möglichst alle notwendigen Parameter zu erfassen und zurückzuliefern. Microsoft arbeitet daran, auch Stimmungen im gesprochenen Text zu erkennen und diese dem Bot ebenfalls zu vermitteln.

LUIS berechnet, wie wahrscheinlich es ist, dass eine bestimmte Äußerung einer bestimmten Intention entspricht. So wird bei der Benutzeranfrage „Bitte hilf mir!“ bestimmt, dass es sich mit einer Wahrscheinlichkeit von 99 Prozent um das Bedürfnis handelt, den Hilfetext zu sehen (*Utilities.Help*). Jedes Mal, wenn eine Anfrage bei LUIS eintrifft, werden alle vom Entwickler definierten Intentionen danach bewertet, wie wahrscheinlich sie sind. Umgekehrt wird ebenfalls berechnet, dass zur Äußerung „Bitte hilf mir“ die Intention „Grüße mich“ (Greeting) nur zu 0,01 Prozent passend ist. Wäre allerdings „Grüße mich“ fälschlicherweise mit 99 Prozent bewertet worden, wäre eine Korrektur erforderlich. Diese erfolgt entweder manuell über die LUIS-Website oder über einen programmatischen LUIS-API-Aufruf.

In der Entwicklungsumgebung legt der Entwickler dann nur noch fest, welche Aktion auf welche Intention folgt. Wurde die Intention „Grüße mich“ (**Abb. 3,** unter *Greeting*) erkannt, wird ein Gruß als Post über die DirectLine-Verbindung geschickt. Der Bot-Framework-Server benachrichtigt daraufhin den oder die Teilnehmer der Konversation über die neue Nachricht. Der Kontext enthält dabei die Metadaten zur Konversation (z. B. wer die Dialogpartner sind).

Abbildung 4 zeigt die Situation im Bot, wenn einer Äußerung keine Intention zugeordnet werden konnte. Die Nachricht wird an den LUIS-Dialog verwiesen und dieser kommuniziert im Hintergrund mit dem LUIS API. Dieses

meldet zurück, dass „None“ der wahrscheinlichste Beweggrund der Anfrage war. Daraufhin reagiert der Bot mit durch den Entwickler festgelegten Maßnahmen. In diesem Beispiel wählt der Bot aus einer Liste mit möglichen Variationen aus und fordert zur erneuten Eingabe auf. Die Liste selektiert zufällig den Wert „Diese Antwort verstehe ich nicht“ (Listing 2). Die möglichen Antworten sind programmatisch oder über den JSON-Aufbau zu definieren.

Mit der Einbindung von LUIS wird ein erweitertes Navigationskonzept geschaffen, das die FormFlow-Navigationskommandos ergänzt. Es eignet sich besonders für Bereiche, in denen nicht klar ist, welches Anliegen der Benutzer hat, also etwa in einer Hilferubrik oder auf einer Kontaktseite. Sinnvoll ist die Nutzung auch bei FAQs, wobei die meisten Anbieter für diesen Bereich auch sogenannte QA-Frameworks (Question-and-Answer-Frameworks) anbieten.

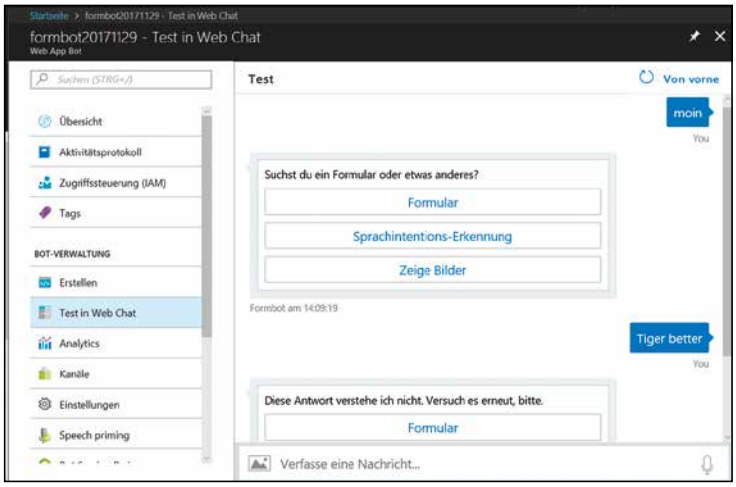

Abb. 4: LUIS-Dialog, wenn einer Äußerung keine Intention zugeordnet werden konnte (Screenshot)

Sicherheitsbedenken, technische Schwierigkeiten und gesellschaftliche Akzeptanz

Eine der größten Herausforderungen im Bereich Bot-Programmierung ist das Thema Sicherheit. Da sämtliche Kommunikation über das Internet läuft – sofern keine Anbieter wie Snips.ai im Einsatz sind [17], die mit lokal analysierender AI arbeiten – sind die ausgetauschten Informationen exponiert. Das wird durch eine Verschlüsselung oder andere, meist kostenpflichtige, Sicherheitsmechanismen der Anbieter kompensiert. Verantwortlich für die Funktionalität, Verfügbarkeit und Sicherheit sind laut den AGB verschiedenster Anbieter die Entwickler bzw. Produkteigner, da die Anbieter lediglich eine Betaversion zur Verfügung stellen. Das bedeutet, dass für die unternehmerische Nutzung in der

EU ein Service Level Agreement notwendig ist, was auch aus datenschutzrechtlicher Sicht empfehlenswert ist.

Wenn der Bot wie im innobis-Projekt in einem Sprachassistenten integriert ist, ergeben sich zudem technische Abhängigkeiten vom Anbieter. Es ist für den Bot-Entwickler nicht möglich, spezielle Verhaltensweisen des benutzten Sprachassistenten zu beeinflussen (so schloss sich etwa die Anwendung, wenn der Benutzer außerhalb des Cortana Canvas klickte). Probleme gab es auch mit zu schnellen Ein- und Ausblendungen der Fragen und Timeouts. Darüber hinaus war es nicht möglich, der Cortana-Stimmerkennung Sonderzeichen wie @ oder π zu diktieren. Die Spracherkennung hat generell Schwierigkeiten, kurze und sprachlich gleich oder ähnlich klingende Antworten (Homophone) des FormFlow zu unterscheiden. Es war zum Beispiel nicht möglich, den Invokationsnamen „LuisBot“ zu diktieren. Dies wurde unterschiedlich als „luis pot“ oder „lewis bot“ fehlinterpretiert. Dieses Problem dürfte auch andere (deutsche) Produkt-, Firmen- und Eigennamen betreffen. Als Problemlösung konnten kanalspezifische Angaben des Benutzers herangezogen werden. Es erwies sich als möglich, nach einer expliziten Zustimmung des Benutzers die Kontakt-, Orts- und Benutzerdaten auszuwerten. Seltene Vornamen oder Familiennamen können auf diese Weise übernommen werden, ohne dass der Benutzer sie diktieren muss.

Letztlich ist auch die Nutzerakzeptanz für Bots in Deutschland eine Herausforderung für Unternehmen. Selbst in der Generation, die als „Digital Natives“ gilt, gibt es Benutzer, die bislang wenig mit Sprachunterstützung gearbeitet haben, bzw. Probleme haben, erlernte Konzepte wie etwa die Diktierfunktion in WhatsApp auf andere Produkte zu übertragen. Laut dem Onlineportal für Statistik Statista GmbH haben 2016 nur etwa 25 Millionen von rund 80 Millionen Einwohnern in Deutschland angegeben, bereits einen Sprachassistenten genutzt zu haben [18]. Es bleibt also eine Herausforderung, mit dem Wachstum in dieser Branche Schritt zu halten, den das Marktforschungsinstitut Gartner Inc. für dialogunterstützende Assistenten [19] genauso vorhersagt wie den Zwei-Billionen-Umsatz für intelligente Lautsprecher für 2020 [20].

Anbieter und Produkte

Einen guten Überblick über die Produkte und Anbieter auf dem Markt bietet die Website von Predictive Analysis [21]. Dort werden neben den multinationalen Konzernen und ihren Produkten auch kleinere Unternehmen und ihre Lösungen vorgestellt und ausgewertet. Generell lässt sich hierbei sagen, dass Google mit seinem Google Assistant über die meisten Nutzer weltweit verfügt. Auch in der Präzision der Spracherkennung ist Google weiter vorangeschritten als beispielsweise die Microsoft- oder Amazon-Produkte. Apple dagegen liegt in Bezug auf die Präzision zwar hinter Google, aber noch vor Microsoft und Amazon. Wie die Slideshow von Konstantin Savenkov, Intento Inc., „Aspect Natural Language Understanding“ zeigt [22], ist auch IBM mit seiner Watson.ai-Plattform eine nicht zu unterschätzende Größe, was Performanz ab einer gewissen Datensatzmenge angeht. Die Untersuchung, die Savenkov dazu durch-

Listing 2: Codeauszug aus den Nachbearbeitungsroutinen zur erkannten LUIS-Intention

```
[Serializable]
  public class BasicLuisDialog : LuisDialog<object>
  {

    /// <summary>
    /// Der Konstruktor verwendet Infos aus der web.config, um sich gegenüber
dem
    /// LUIS-Sprachmodell-API zu authentifizieren.
    /// </summary>
    public BasicLuisDialog() : base(new LuisService(new
                LuisModelAttribute(ConfigurationManager.AppSettings["LuisAppId"],
                                   ConfigurationManager.AppSettings["LuisAPIKey"])))
    {

    }

    /// <summary>
    /// Die “keine-Intention”-Intention ist die Intention, die die AI erkennt, wenn
    /// sie sich nicht sicher ist, welche Intention gemeint ist.
    /// </summary>
    /// <returns></returns>
    [LuisIntent("")]
    [LuisIntent("None")]
    public async Task NoneIntent(IDialogContext context, LuisResult result)
    {
      string luisno = Resource1.LuisNone;//sorry. Didn’t understand…
      await context.PostAsync(Resource1.LuisNone+$"{result.Query}");
      context.Wait(MessageReceived);
    }

    /// <summary>
    /// Diese Intention ist beispielhaft für eine selbstimplementierte Intention.
    /// Damit sie auf diese Kundenanfrage antwortet, muss die Äußerung des
    /// Kunden von der AI als ähnlich zu einer bekannten Intention im Web Service
    /// als am höchsten wahrscheinlich erkannt werden.
    /// </summary>
    /// <returns></returns>
    [LuisIntent("Greeting")]
    public async Task GreetingIntent(IDialogContext context, LuisResult result)
    {
      await context.PostAsync(Resource.Resource1.regionalGreeting);
      context.Wait(MessageReceived);
    }
```

geführt hat, zeigt die Stärken und Schwächen der fünf größten Bot-Anbieter auf.

Es gibt neben den kommerziellen Anbietern im Übrigen auch zwei Open-Source-Projekte, Lucida [23] und Athena [24], die sich mit künstlicher Intelligenz und sprach- bzw. bildverarbeitenden dialoggestützten Systemen beschäftigen. Lucida hat allerdings recht einschränkende Systemvoraussetzungen, weil es nur auf zwei bestimmten Ubuntu-Versionen installierbar ist. Dieses Problem kann in VirtualBox umgangen werden, sofern ein anderes Betriebssystem auf 64-Bit-Basis verfügbar ist. Lucida erfordert einen Arbeitsspeicher von mehr als 6 GB RAM. Athena zielt darauf, die Ergebnisse von Siri, Alexa und Co. zu erreichen und zu erweitern. Es ist modular aufgebaut und bietet momentan Module für Musikwiedergabe, Social Media (Twitter) und Hausautomation. Athena ist in Python programmiert.

Fazit

Sprachgestützte Unterhaltungen mit Bots weisen einen vielversprechenden Weg, Strukturen im Unternehmen effizienter und effektiver zu machen. Zudem bieten sie die Möglichkeit, bisher nicht erreichte Benutzergruppen zu erschließen. Die datenschutzrechtlichen und sicherheitskritischen Bedingungen können durch Verschlüsselungstechniken, neue Authentifizierungstechniken und Service Level Agreements gelöst werden. Nicht nur Fremdanbieterplattformen, sondern auch eigene Systeme sind dank vorhandener Open-Source-Projekte wie Athena nutzbar. Dies lohnt sich, weil Bots und der Handel mit Bot-fähigen Geräten eine zunehmende Verbreitung erfahren, sodass es sogar denkbar ist, dass Bot2Bot-Konversationen regulär stattfinden (auch Multi-Agent-Systeme genannt). Ein klassisches Beispiel hierfür ist die Organisation von Terminen und z. B. die Bitte an einen Bot: „Bitte Cortana, unterhalte dich mal mit Siri von meinem Mann und mit Google Assistant von meiner Schwester, und prüfe, wann wir Zeit haben, Pizza essen zu gehen."

Friederike Geissler, Junior Consultant Development & Integration Services bei der innobis AG, hat im Unternehmen ihre Abschlussarbeit zum Thema „Sprachassistenten in mobilen Bankanwendungen" geschrieben. Hierbei sind zwei Prototypen, eine App und ein Bot entstanden, die einen Benutzer durch einen Sprachassistenten beim Ausfüllen eines Onlineformulars unterstützen.

Links & Literatur

[1] Ng, Andrew: https://twitter.com/AndrewYNg/status/809579698883727360

[2] Nuance: „Control your computer voice with speed and accuracy": https://www.nuance.com/dragon.html#standardpage-mainpar_backgroundimage_copy

[3] Carlini, N.; Wagner, D: „Audio adversarial examples: Targeted Attacks on Speech-to-Text", University of California, 2018: https://arxiv.org/pdf/1801.01944.pdf

[4] Song, L.; Mittal, P.: „Inaudible Voice Commands", Princeton University, 2017: https://arxiv.org/pdf/1708.07238.pdf

[5] Feng, H.; Fawaz, K.; Shin, K.: „Continuous Authentication for Voice Assistants. Computer Science – Cryptography and Security": https://arxiv.org/abs/1701.04507

[6] „Bots für die Bank: Die Zukunft der Kommunikation": https://customers.microsoft.com/en-us/story/fiducia-gadit-cognitive-services-cortana-azure-powerbi-banking-germany

[7] Okamoto, S.; Scerri, P.; Sycara, K.: „Toward an understanding of the impact of software personal assistants on human organizations": http://dx.doi.org/10.1145/1160633.1160745

[8] Haslam, R.: „The bots are coming – 5 essential things fintech firms need to know": https://banknxt.com/56496/bots-5-essential-fintech/

[9] Lunden, I.: „Rise of the bots. Techcrunch": https://techcrunch.com/2016/04/07/rise-of-the-bots-x-ai-raises-23m-more-for-amy-a-bot-that-arranges-appointments/?guccounter=1

[10] LuisBot, Innobis AG: https://github.com/fr6087/Hoermirzu

[11] Vincent, J.: „Twitter taught Microsoft AI's Chatbot to be a racist asshole in less than a day": https://www.theverge.com/2016/3/24/11297050/tay-microsoft-chatbot-racist

[12] Microsoft Botbuilder Sample Projects: https://github.com/Microsoft/BotBuilder-Samples

[13] Brown, P.: „Internet of Stranger Things": https://blogs.windows.com/buildingapps/2016/10/31/the-internet-of-stranger-things-wall-part-1-introduction-and-remote-wiring/

[14] Jibo: https://www.jibo.com/

[15] Key concepts of DirectLine API 3.0. Microsoft Botframework documentation: https://docs.microsoft.com/en-us/azure/bot-service/rest-api/bot-framework-rest-direct-line-3-0-concepts?view=azure-bot-service-3.0

[16] Osborne, R.: „Connecting Alexa to a Botframework Chatbot": https://www.robinosborne.co.uk/2016/12/19/connecting-alexa-to-a-botframework-chatbot/

[17] Snips Platform Documentation: http://snips.ai.

[18] Digital Assistants: Statista dossier: https://de.statista.com/statistik/daten/studie/620225/umfrage/nutzung-von-sprachassistenten-in-deutschland/

[19] Love, J: „Gartner's top 10 predictions for 2016 and the 'post-app' era": https://www.itworldcanada.com/article/gartner-top-ten-predictions-for-2016-and-post-app-era/377594

[20] „Gartner Says Worldwide Spending on VPA-Enabled Wireless Speakers Will Top $2 Billion by 2020": https://www.gartner.com/newsroom/id/3464317

[21] Top 22 intelligent personal assistants or automated personal assistants. PredictiveAnalysis: https://www.predictiveanalyticstoday.com/top-intelligent-personal-assistants-automated-personal-assistants/#content-anchor

[22] Savenkov, K: „NLU/ Intent Detection", Intento, 2017: https://www.slideshare.net/KonstantinSavenkov/nlu-intent-detection-benchmark-by-intento-august-2017

[23] Lucida: https://github.com/claritylab/lucida

[24] Athena: https://github.com/rcbyron/hey-athena-client