Sprachassistierter Entwicklungsprozess für automatisierungstechnische Systeme

Ein Ansatz zur Strukturierung komplexer Entwicklungsprozesse

D. White, M.Sc., Prof. Dr.-Ing. Dr. h. c. **M. Weyrich**,
Universität Stuttgart, Stuttgart

Kurzfassung

Der Systementwicklungsprozess nimmt immer mehr an Komplexität zu, da die Systeme selbst immer komplexer werden. Gleichzeitig Vermischen sich die verschiedenen Disziplinen wie Maschinenbau, Elektrotechnik und Softwaretechnik zunehmend, so dass Unternehmen einer Disziplin sprunghafte Komplexitätszuwächse bei ihren Systemen und in ihrer Entwicklung haben. Deshalb wird in dieser Veröffentlichung ein Konzept eines Sprachassistenten erarbeitet, der durch eine Entwicklungsphase führt. Daraus geht hervor, dass die Software zur Unterstützung der Entwicklung ein Informationsmodell benötigt, um die Daten des entwickelten Systems zu speichern und diese mit dem vorhandenen Wissen zu verbinden. Dieses Wissen kann entweder intern oder im Web vorhanden sein. Der Entwicklungsprozess soll daher Kooperation unterstützen, so dass die Assistenzsoftware und Ingenieure miteinander interagieren.

1. Einleitung

Die Konzepte des modellbasierten Systems Engineering verbreiten sich zunehmend, um die Komplexität in der Entwicklung heutiger Automatisierungssysteme zu bewältigen. Dabei werden graphische Modelle eingesetzt, um für den Menschen die einzelnen Bestandteile eines Automatisierungssystems verständlich darzustellen [1]. Dies ist unter anderem erforderlich, da verschiedene Disziplinen immer verzahnter miteinander interagieren und multidisziplinäres Wissen ausreichend berücksichtigt werden soll [2]. Automatisierungstechnische Systeme weisen dabei einen hohen Anteil mechatronischer Komponenten auf. Insbesondere der hohe Software-Anteil, welcher die Systeme dazu befähigt ihre Aufgaben zu erfüllen, erfordert im Entwicklungsprozess die Darstellung der statischen Abhängigkeiten, der einzelnen Komponenten eines Systems untereinander [3]. Diese Eigenschaft erfordert zudem, dass im Entwicklungsprozess die dynamischen Abhängigkeiten, wie z.B. das Anpassen eines Parameters für einen Aktor nach einer vom System wahrgenommenen Veränderung in der Umgebung des Systems, beachtet werden müssen.

Dadurch erhöht sich zum einen die Anzahl an vorhandenen Modellen im Entwicklungsprozess, wodurch das System komplizierter wird und zum anderen erhöhen sich in besonderem Maße die Abhängigkeiten unter den Modellen, wodurch das System an Komplexität gewinnt [4]. Die Abhängigkeiten sind dabei von besonderer Bedeutung, wenn das System sich selbst anpassen kann, da hier Änderungen einer Komponente an einer anderen Komponente vorgenommen werden können [5], [6].

Die bisherige modellbasierte Systementwicklung zeigt in der Praxis, dass trotz der gegebenen Vorteile Komponenten dennoch klassisch entwickelt und ungenügend dokumentiert werden. Insbesondere bei Komponenten, die zwischen Zulieferern und Erstausrüstern (OEM) ausgetauscht werden, fehlen Modelle aus verschiedenen Gründen, wie z.B. der Geheimhaltung [7], [8].Neben der steigenden Komplexität und fehlenden Modellen ist die agile Entwicklung ein wichtiges Thema moderner Unternehmen [9]. Um der agilen Entwicklung gerecht zu werden und dennoch eine strukturierte Entwicklung gewährleisten zu können, muss ein entwicklungsbegleitendes System vorhanden sein, welches die Architektur und den Funktionsumfang des zu entwickelnden Systems überwacht sowie die Freiheitsgrade des Entwicklers sicherstellt. Nach Einschätzung der Autoren eignet sich dafür ein Sprachassistent, der in einer Dialogform die Entwicklung unterstützt und dabei ein Informationsmodell nutzt um Wissen über das System zu erfassen.

Der konzipierte Sprachassistent fragt auf Basis vorgefertigter Fragen, wie die Modelle des Systems aussehen sollen und verarbeitet die Antworten. Der Entwickler kann hierbei der Fragereihenfolge des Sprachassistents folgen oder kann an unterschiedliche Stellen in der Entwicklung springen. Das hier vorgestellte Konzept sieht vor, dass der Sprachassistent fehlende Informationen bei dem Entwickler zum Beginn eines Entwicklungstages wieder anspricht. Dadurch kann ein starrer Prozess flexibilisiert werden, ohne dass die Dokumentation oder die Vorgehensstruktur darunter leiden. Eine Kernfunktionalität des Sprachassistenten ist dabei, dass in Zusammenwirken zwischen Entwickler und Sprachassistenten Modelle spezifiziert werden.

Im Folgenden wird anhand der Anforderungsphase gezeigt, dass eine solche konversationsgestützte Entwicklung dem Entwickler das Vorgehen freistellt, jedoch wichtige Schritte aufeinander folgen lässt und fehlende Informationen erkennt und anspricht. Mit diesem Konzept soll der Entwicklungsprozess dahingehend verbessert werden, dass der Sprachassistent die Stellen im Entwicklungsprozess aufzeigt, an denen Informationen und Modelle ergänzt werden sollten.

2. Einführung in die Sprachassistenz

Sprachassistenten respektive Chatbots gewinnen zunehmend an Bedeutung. Sie werden in verschiedensten Bereichen eingesetzt, zum Beispiel in Smart Home Geräten, beim Kundensupport auf Internetseiten oder in medizinischen Produkten für ältere Menschen [10]. Dabei können sie dem Menschen komplizierte Interaktionen erleichtern, da sie auf große Datenbanken zugreifen können. Grundlegende Probleme, die ein Sprachassistenten zu lösen versucht, sind zum einen die Produktivität von Prozessen und zum anderen der Vorgang der Informationsbeschaffung [11]. Durch die Hilfe, die durch den Sprachassistenten zur Verfügung gestellt wird, ist es dem Nutzer möglich, sich mehr auf Informationen zu fokussieren und weniger auf die Informationsbeschaffung und deren Bereitstellung.

Aber auch bei der Systementwicklung, speziell bei der modellbasierten Systementwicklung, entstehen Vorteile: So kann der Entwickler seine Vorstellungen mittels natürlicher Sprache an den Sprachassistenten weitergeben. Der Sprachassistent nimmt diese auf und ordnet sie im Idealfall direkt in ein grafisches Modell ein. Dies kann für den Entwickler deutlich einfacher und schneller gehen als das Modell selbst zu zeichnen, da er dafür weder der Syntax noch die Darstellungsform kennen muss. Wie [12] zeigt, wird zudem der Vorteil geboten, dass die Eingabe ortsunabhängig erfolgen können. Als weiterer Vorteil wird in der Studie von [13] angegeben, dass eingaben per Sprache als natürlicher wahrgenommen werden, als eingaben per Touchscreen. Dabei ist zu beachten, dass der Touchscreen nach wie vor als natürlichste Interaktionsmöglichkeit zwischen Maschinen und Menschen gilt.

Die Herausforderung für den Sprachassistenten besteht darin, dass er das Gedankenkonstrukt des Benutzers einer losen Struktur folgend fassen und einordnen können muss. Für diese fassen und einordnen der Gedankenkonstrukte wird in der vorliegenden Veröffentlichung eine Struktur vorgestellt.

3. Eigenschaften von Sprachassistenten

Ein Sprachassistent bietet ein breites Anwendungsgebiet und somit sind seine Eigenschaften ebenfalls weit gefächert. Umgangssprachlich wird ein Chatbot, der neben einer Chatfunktion eine Audioeingabe unterstützt, sowohl als Sprachassistent als auch als klassischer Chatbot bezeichnet. Bezeichnungen dieser Systeme gehen in unterschiedlichen Literaturen auseinander. Die der Eigenschaftstabelle zugrundeliegende Quelle unterscheidet bezüglich der Eingabeform. Die Tabelle 1 zeigt hierbei die grundlegenden Eigenschaften und Einsatzmöglichkeiten von Sprachassistenten.

Tabelle 1: Überblick der Eigenschaften von Sprachassasistenten [14]

Vergleichsdimension	Textform	Audio
Dialogform	Unterhaltungsform (Chat), die eine vergleichsweise komplexe Frage- und Antwortstruktur besitzt.	Eingabe von Sprachbefehlen und Fragen, Verkettungen werden in geringem Maße unterstützt.
Ausgabegerät	Schnittstellen, insbesondere das Smartphone oder andere gängige Endgeräte.	Schnittstelle in Form eines smarten Lautsprecher (z.B. Amazon Echo oder Google Assistant).
Ort der Nutzung	„Integriert in bestehende Unternehmenspräsenzen auf Plattformen, wie z. B. Facebook Messenger oder WhatsApp“	In ruhigem Umfeld wo eine Nutzung der Sprachfunktion möglich ist
Aufbau einer Persönlichkeit	Erstellung eines Avatars	„plattformspezifische Persönlichkeit (z. B. Alexa von Amazon)“

4. Einsatzzwecke für Sprachassistenten in der Entwicklung

Sprachassistenten eignet sich, um durch eine strukturierte Entwicklung zu führen. Allerdings werden in der Entwicklung die atomaren Entwicklungsprodukte nicht immer nach einer festen Struktur hintereinander erstellt. Es wird somit ein nicht-lineares Dialogmuster für einen Sprachassistenten in der Entwicklung benötigt. [15]. Entsprechend der „Men are better at – Machines are better at“ – Listen [16], [17], die das Prinzip der Aufgabenteilung zwischen Menschen und Maschine beschreiben, sollte der Sprachassistent versuchen die Struktur des Entwicklungsverlaufs beizubehalten ohne jedoch den Entwickler an die Struktur fest zu binden.
Die grundlegende Architektur eines intelligenten Sprachassistenten ist in Bild 1 dargestellt.

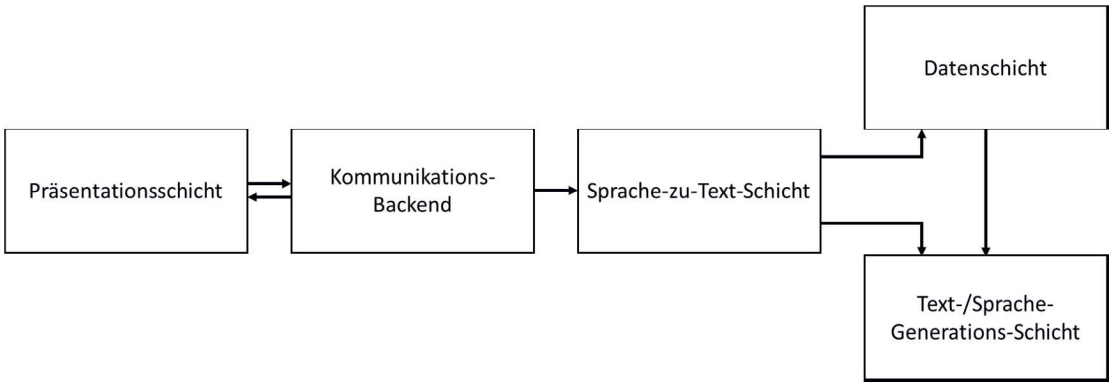

Bild 1: Grundlegende Architektur eines Sprachassistenten [18]

Die Sprache-zu-Text-Schicht besteht aus drei Komponenten: dem Natural Language Processing, Natural Language Understanding und Decision Engine. Die Kombination aus allen drei Komponenten, erlaubt es Aussagen des Benutzers zu verarbeiten. Die Datenschicht beinhaltet die Wissensdatenbank, die Datenanalyse und den Datenspeicher. An dieser Stelle wird die Fragestruktur hinterlegt. Während Natural Language Understanding die Eingaben des Benutzers in eine für die Maschine leserliche Sprache wandelt, ist Natural Language Generation der Prozess die Sprache des Computers in eine menschliche Sprache umzuwandeln. Dies befindet sich in der Text-/Sprache-Generations-Schicht [19].

Für ein solches System bietet sich in der Entwicklung der Anwendungsfall der Benutzungsschnittstelle. Durch die Eigenschaften eines Sprachassistenten ist die Integration jedoch nicht trivial, da bestimmten verbalen Aussagen spezifische Aktionen zugeordnet werden müssen. Diese Zuordnung bedeutet, dass aus einem chaotischen Input ein geordneter Output generiert werden muss.

Als Anwendungsfälle ergeben sich dann die in Bild 2 dargestellten.

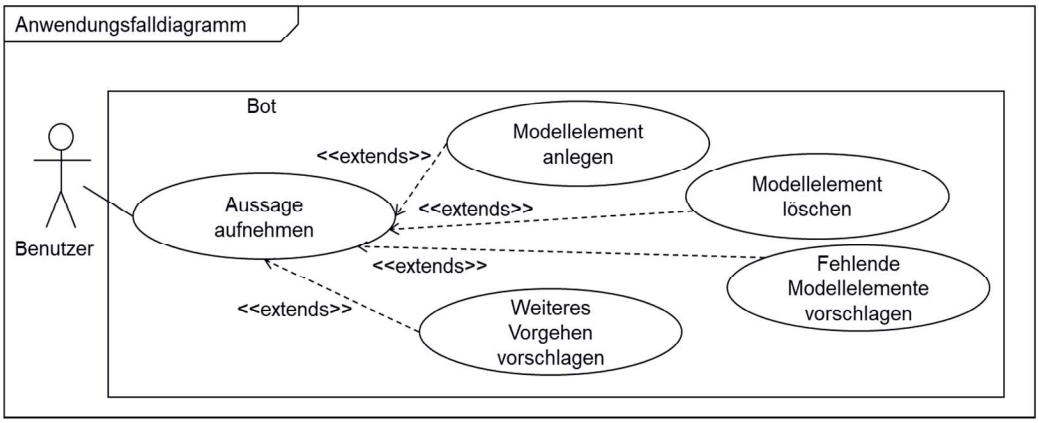

Bild 2: Anwendungsfalldiagramm eines Sprachassistenten für die Entwicklung

Um einen grafischen Entwicklungsprozess zu unterstützen, soll ein Sprachassistent in der Entwicklung in der Lage sein, Diagramme zu erstellen. Da jedes Teil eines Diagramms zu einem Modell gehört und ein Modellelement die kleinste Einheit ist, wird dieser Anwendungsfall als „Modellelement anlegen“ bezeichnet. Die erweiternden Anwendungsfälle „fehlende Modellelemente vorschlagen“ und „weiteres Vorgehen vorschlagen“ schließen mit ein, dass der Sprachassistent eine Kenntnis über das verwendete Vorgehensmodell und die Syntax von Modellen verfügt.

Aufgrund der Zuvor erwähnten nicht-linearen Dialogstruktur und der gerichteten Zuordnung ergibt sich für den Konversationspfad des Sprachassistenten die Struktur in Bild 3.

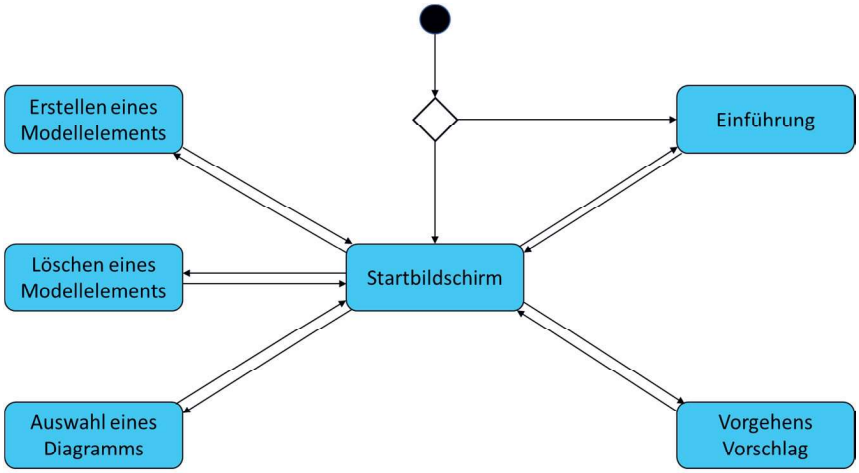

Bild 3: Konversationslogik für einen Sprachassistenten in der Entwicklung

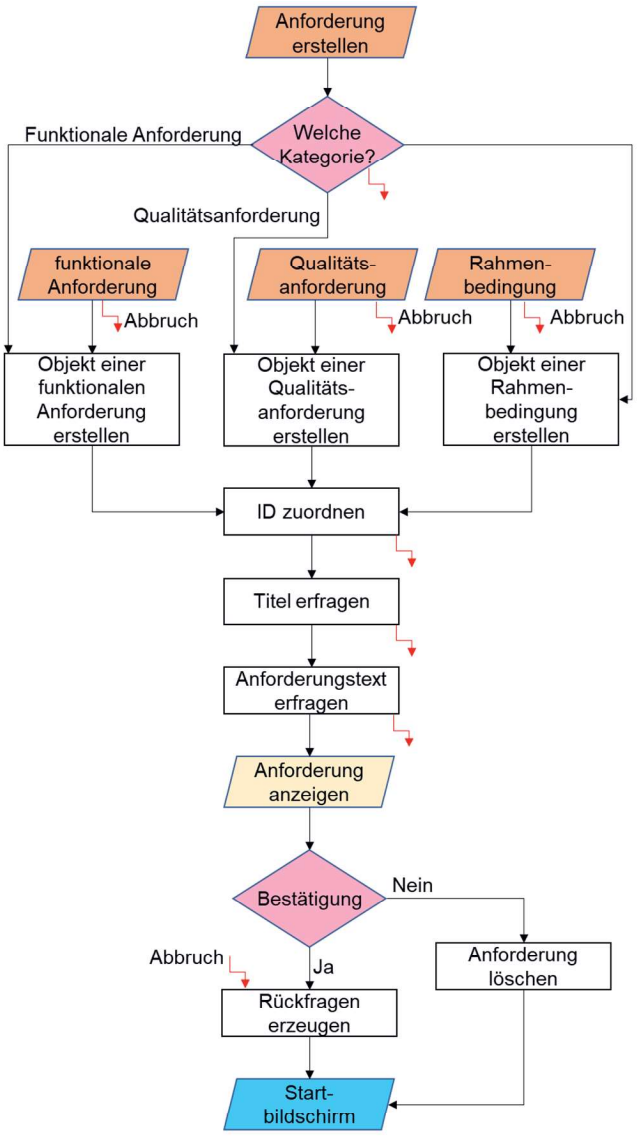

Bild 4: Konversationslogik zum Anlegen einer Anforderung

Diese Sternstruktur ist nötig, da über den Startbildschirm jede Konversationsart gestartete werden kann. Innerhalb einer Konversation wie „Erstellen eines Modellelements“ wird hingegen eine bestimmte Struktur verfolgt. Werden jedoch Aussagen getroffen, die nicht in die Struktur passen, wäre es sinnvoll diese Aussagen zurückzustellen, sodass der Sprachassistent nach Abschluss der Konversation diese über den Startbildschirm wieder aufnehmen kann.

In Bild 4 ist als Beispiel der Ausschnitt aus der Konversationslogik des aufgebauten Sprachassistenten-Systems für das Anlegen einer Anforderung dargestellt. Das Beispiel zeigt, dass zum Erstellen einer Anforderung eine gewisse Struktur notwendig ist. Um der nicht-linearen Dialogstruktur jedoch gerecht zu werden, kann der Vorgang jederzeit abgebrochen werden. Die bisherigen Informationen werden dabei gespeichert und

sofern die erstellte Anforderung unvollständig ist, werden Rückfragen generiert. Diese Rückfragen sind in der Konversationslogik so eingeordnet, dass sie als Vorgehensvorschlag (vgl. Bild 3) gestellt werden und wenn das entsprechende Modellelement bearbeitet werden soll. So wird auf eine Unvollständigkeit des Modells hingewiesen und die Modellqualität auf einem durch die Einstellungen des Sprachassistenten vorgegebenen Mindestmaß gehalten.

5. Beschreibung des Prototypens

Für die Verarbeitung der Eingaben benötigt der Sprachassistent verschiedene Komponenten. In der Bild 5 sind die nötigen Komponenten dargestellt.

Die Sprache-zu-Text Komponente dient dem Anwender dazu, den Sprachassistenten über eine zur Verfügung gestellte Spracheingabe oder Texteingabe zu nutzen. Diese Komponente gibt den Sprachbefehl in Textform an die Dialog-Komponente weiter. Innerhalb dieses Blockes findet keine Kategorisierung oder Veränderung von Daten und Eingaben statt, lediglich die Umwandlung in eine vom Sprachassistenten-System verständliche Datenform.

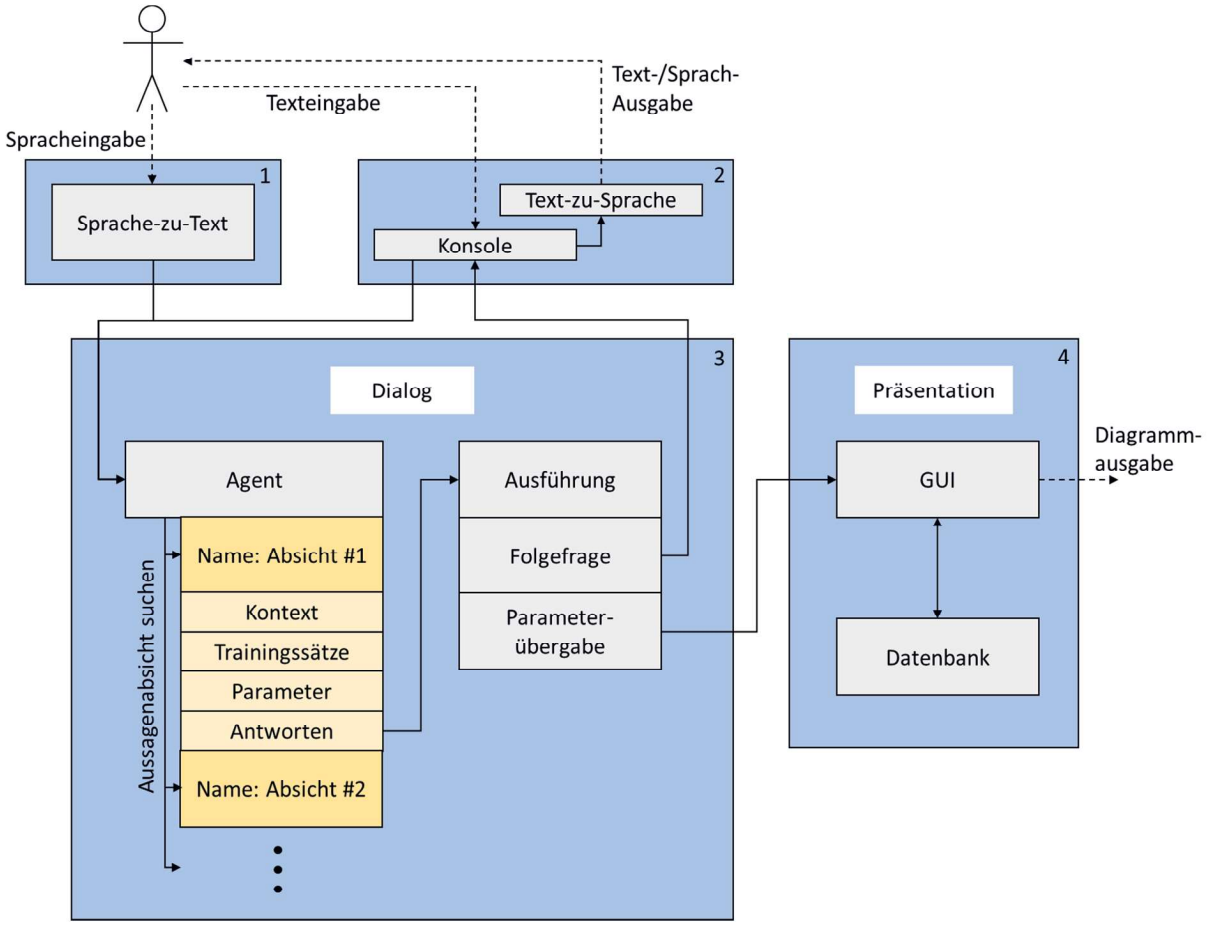

Bild 5: Komponentenübersicht des Sprachassistenten-Systems

Die Konsole im Block 2 in Bild 5 erfüllt zwei Grundlegende Aufgaben: Die eine bei der Eingabe und die andere bei der Ausgabe von Daten. Sollte der Anwender Texteingaben der Spracheingabe vorziehen, wird diese Texteingabe durch die Konsole ohne Sprachverarbeitung an einen Agenten weitergegeben. Die Software-Komponente Agent gleicht die Text- oder Spracheingabe mit den im Sprachassistenten-System hinterlegten Absichten ab. Eine Absicht wiederum ist ein im Sprachassistenten-System abgespeicherter Text, der eine Reaktion des Sprachassistenten beschreibt. Der Sprachassistent soll also die Anwender-Absicht über die Spracheingabe erkennen und entsprechend eine im Sprachassistenten-System hinterlegte Absicht auswählen. Ein Entwickler kann dieser hinterlegten Absicht Trainingssätze zuweisen, bei der diese anspricht. Über einen einer Absicht zugeordneten Kontext kann angegeben werden, dass trotz entsprechender Sätze eines Anwenders, die zu den Trainingssätzen passen, die Absicht nicht anspricht. Diese Trainingssätze können zu jeder Absicht ergänzt werden. In ihnen wird festgelegt, wie die entsprechende Absicht ausgelöst werden soll. Diese Trainingssätze enthalten Schlagworte, die eindeutig einer Absicht zugewiesen werden können. Zudem können sie durch „Elemente normaler Konversation“ ergänzt werden.
Beispiel:

- „erstelle eine Klasse“
- „beginne bitte mit der Klassenerstellung“
- „ich möchte eine Klasse erstellen“

Mit Hilfe von manuellen Spracheingaben kann der Sprachassistent bzw. genauer die Komponente Agent auch so trainiert werden, dass auch ähnliche Aussagen und Formulierungen dazu führen bestimmte Absichten auszulösen. Spezielle Absichten benötigen zur Umsetzung ihrer Aufgabe eine spezielle Auswahl an Eingabeparametern. Sollten in einem einfachen Beispiel die Hintergrundfarbe einer graphischen Ausgabe geändert werden, so ist es relevant, die gewünschte Farbe explizit zu benennen. Da es nicht effektiv ist, für alle möglichen Farben Trainingsätze einzurichten, lassen sich sogenannte Entitäten erzeugen, die Eingaben Themengruppen wie zum Beispiel Farben, zuordnen. Innerhalb dieser Entitäten sind demnach die Farboptionen gelistet. Wird eine Absicht erkannt, werden die hinterlegten Parameter und die Antwort an die Komponente Ausführung übergeben. Sollte die Ausführung Folgefragen an den Anwender haben, laufen auch diese Folgefragen über die Konsole und werden nach Bedarf mit sogenanntem Text-zu-Sprache auch in Audioform ausgegeben.
Innerhalb des Dialog-Blockes werden die grundlegenden Funktionen des Sprachassistenten realisiert. Zunächst wird eine Texteingabe mit Hilfe der Konversationslogik verarbeitet und fort-

führend werden Parameter für eine Ausgabe oder Folgefrage erzeugt. Eine Antwort des Systems ist eine Aktion, die eine Absicht ausführt. Sei es eine Folgefrage an den Anwender oder eine Aufforderung an die Darstellung, jede ist explizit ihrer Absicht zugeordnet.
Die Hauptaufgabe des in der Komponente Dialog integrierten Agenten ist es, eine Eingabe durch den Anwender der entsprechenden Absicht zuzuordnen. Hierfür greift der Agent auf die, in den jeweiligen Absichten formulierten, Trainingssätzen und den Inhalt des Kontextes zu.
Der Agent wählt exakt eine Absicht aus, die passend zur Eingabe (Kontext, Trainingssätze) ist. Absichten, die aufgrund ihres Kontextes nicht ausgelöst werden können, sind dem Agenten nicht zugänglich, er kann sie somit nicht auf Trainingssätze durchsuchen und folglich nicht aufrufen.
Bei der Ausführung sind grundlegend zwei Optionen voneinander zu unterscheiden. Zum einen kann eine Absicht eine Folgefrage an den Anwender zur Folge haben. Diese Folgefrage wird wie beschrieben über die Konsole an den Anwender ausgegeben. Zum anderen kann die Ausführung Parameter erstellen, die die Basis für Zeichenoperationen der Darstellung bilden, wie zum Beispiel ein Name, Attribut oder eine Operation einer bestimmten Klasse. Diese werden dementsprechend an die Komponente Darstellung weitergegeben.
Die Komponente Darstellung besitzt eine kleine Datenbank. Parameter, die über die Ausführung erhalten werden, beinhalten Aufforderungen wie „erstelle eine Anforderung“ oder „zeichne eine Komposition“. Diese Aufforderungen müssen innerhalb der Darstellung verarbeitet werden, was bedeutet, zum Beispiel der Parameter „Komposition“ muss in eine grafische Darstellung, einer Komposition umgewandelt werden. Diese Verbindung zwischen den empfangenen Parametern und der grafischen Symbole sind in einer Datenbank hinterlegt.
Für den Prototypen wurde die Browserapplikation Dialogflow in Verbindung mit der Entwicklungsplattform Actions on Google verwendet. In der Vorbereitung wurden unterschiedliche Plattformen angetestet. Dabei ergab sich deutlich, dass diese Plattformen noch in ihren Frühstadien sind, wodurch noch nicht alle nötigen Aspekte eines Sprachassistenten für die Entwicklung umgesetzt werden können.
Über Dialogflow, Actions on Google und dem Smart Display von Google verbindet der Prototyp einen Sprachassistenten mit einer graphischen Oberfläche. Die Struktur des Prototypens ist in Bild 6 dargestellt.

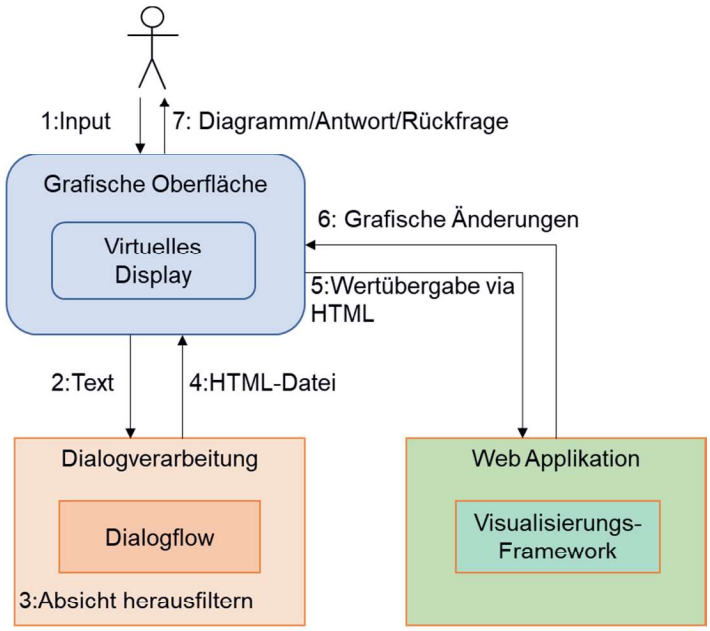

Bild 6: Systemarchitektur des Sprachassistenten-Systems

Der Grundlegende Ablauf im Prototypens des Sprachassistenten ist folgender:

1. Der Benutzer übergibt seine Anweisungen als Text bzw. per Sprache in die Chatkonsole, welche in dem virtuellen Display integriert ist. Als virtuelles Display wird das Smart Display von Google verwendet.
2. Nun springt das Sprachassistenten-System in die Dialogverarbeitung, dessen Konversationsverlauf von der Browserapplikation Dialogflow gesteuert wird. In der Dialogverarbeitung werden dann die Absichten zugeordnet und die Parameter erzeugt.
3. Danach sendet die Dialogverarbeitung eine Anfrage an eine Webseite. Diese Nachricht ist in einer Datei verpackt und enthält die Informationen über die erkannte Absicht, die Parameter und die Antwort der Dialogkomponente. Die Webseite verarbeitet die Informationen und erzeigt eine HTML-Datei.
4. Die HTML-Datei wird zurück an die grafische Oberfläche gesendet.
5. Mit Hilfe der Daten aus der HTML-Datei wird die Web Applikation geladen. Die Web Applikation, welche in HTML aufgebaut ist, beinhaltet die interaktive Programmierschnittstelle, welche wiederum eine JavaScript API ist. Die in der HTML-Datei mitgelieferten Werte sind in lösen in der Web-Applikation Funktion aus.
6. Über die Funktionen der Web App werden die darin angeforderten Änderungen am Diagramm vollzogen.
7. Der Benutzer kann die Antwort in der Konsole lesen und die Änderungen am Diagramm in der grafischen Benutzeroberfläche sehen.

6. Schlussfolgerung

Das Sprachassistenten-System zeigt, dass die Konversationslogik für einen Entwicklungsprozess wie er bei automatisierungstechnischen Systemen vorkommt, unterstützend eingesetzt werden kann. Das Sprachassistenten-System

- unterstützt durch sein Informationsmodell eine Mindestqualität der Modelle, wie im angeführten Beispiel am Anforderungsmodell gezeigt
- und lässt dem Entwickler dennoch die Freiheit keiner starren Struktur zu folgen.

Durch das Ablegen der Rückfragen bei offenen Punkten und die Verknüpfung an die entsprechenden Stellen im Modell werden leere Diagramme vermieden. Das Informationsmodell unterstützt zudem, dass die leeren Stellen eines Modells an der richtigen Stelle dieses Modells nachgetragen werden. Dies geschieht durch die Ablage verbundener Daten in einer Datenbank. Das gezeigte Konzept der Interaktion zwischen dem Entwickler und dem Sprachassistenten-System erfüllt die Kooperation durch eine flexible Struktur. Diese Struktur lässt dem Entwickler die Freiheit unterschiedliche Modellelemente hintereinander zu erzeugen. Die Elemente der Konversation werden von dem Prototyp verlässlich und zu jedem Zeitpunkt erreicht. Somit ist diese Grundstruktur der Interaktionsmöglichkeit für eine sprachassistierte modellbasierte Entwicklung zu empfehlen. Die dadurch gegebene Freiheit zum unsystematischen Vorgehen eines Entwicklers und die dennoch stets abrufbare Struktur des Systemmodells verknüpft die verschiedenen Elemente automatisierungstechnischer Systeme und bietet damit dennoch einen systematischen Überblick. Zukünftig wird das Sprachassistenten-System darum erweitert alle Phasen eines Systementwicklungsprozesses zu begleiten und dabei unterschiedliche Vorgehensmodelle einzuhalten.

7. Literatur

[1] T. Frank, K. Eckert, T. Hadlich, A. Fay, C. Diedrich, and B. Vogel-Heuser, Erweiterung des V-Modells® für den Entwurf von verteilten Automatisierungssystemen. *at - Automatisierungstechnik* 61, 2, 79-91, De Gruyter <https://doi.org/10.1524/auto.2013.0009> Abgerufen am 04.05.2020

[2] J.-P. Schmidt, T. Müller, and M. Weyrich, Methodology for the model driven development of service oriented plant controls," *Procedia CIRP*, vol. 67, pp. 173–178, 2018.

[3] S. Biffl, A. Lüder, and D. Gerhard, Multi-Disciplinary Engineering for Cyber-Physical Production Systems: Data Models and Software Solutions for Handling Complex Engineering" Projects. Springer, 2017

[4] J. Jasperneite and O. Niggemann, Systemkomplexität in der Automation beherrschen," *atp Ed.*, vol. 54, no. 09, pp. 36–45, 2012.

[5] L. Berardinelli, S. Biffl, E. Maetzler, T. Mayerhofer, and M. Wimmer, Model-Based Co-Evolution of Production Systems and their Libraries with AutomationML," *2015 IEEE 20th Conf. Emerg. Technol. Fact. Autom.*, pp. 1–8.

[6] T. Müller, N. Jazdi, J.-P. Schmidt, and M. Weyrich, Cyber-Physical Production Systems: enhancement with a self-organized reconfiguration management" Procedia CIRP, 2020 (accepted).

[7] T. Huth and T. Vietor, Systems Engineering in der Produktentwicklung : Verständnis, Theorie und Praxis aus ingenieurswissenschaftlicher Sicht," pp. 125–130, 2020.

[8] J. Gausemeier, R. Dumitrescu, D. Steffen, A. Czaja, O. Wiederkehr, and C. Tschirner, Systems Engineering in der industriellen Praxis," *Heinz Nixd. Inst.*, 2013

[9] M. Müller, K. Hörmann, L. Dittmann, and J. Zimmer, „Automotive SPICE®in der Praxis: Interpretationshilfe für Anwender und Assessoren", dpunkt. verlag, 2016

[10] N. Sahlab, C. Sailer, N. Jazdi, and M. Weyrich, „Designing an elderly-appropriate voice control for a pill dispenser", 2020.

[11] G. Spur, and E. Uhlman**,** „Industrieroboter", *In: Grote KH., Feldhusen J. (eds)* Verfügbar unter: https://doi.org/10.1007/978-3-642-17306-6_221 Springer, Berlin, Heidelberg 2011

[12] R. Maskeliunas, K. Ratkevicius, and V. Rudzionis, Voice-based human-machine interaction modeling for automated information services," *Elektron. ir Elektrotechnika*, vol. 4, no. 4, pp. 109–112, 2011.

[13] B. Martin, https://www.spitch.ch/de/blog/spitch-studie-sprechen-mit-computern-ist-am-nat-rlichsten/, abgerufen am 03.05.2020

[14] K. Stanoevska-Slabeva, *Wirtsch Inform Manag",* Verfügbar unter: https://doi.org/10.1007/s35764-018-0117-7*,* Springer Fachmedien Wiesbaden, 2018

[15] A. Sieber, Dialogroboter: Wie Bots und künstliche Intelligenz Medien und Massenkommunikation verändern", Springer, Wiesbaden 2019

[16] T. B. Sheridan, Supervisory control," *Handb. Hum. factors*, pp. 1243–1268, 1987.

[17] S. W. A. Dekker and D. D. Woods, MABA-MABA or Abracadabra? Progress on Human-Automation Co-ordination," *Cogn. Technol. Work*, vol. 4, no. 4, pp. 240–244, 2002

[18] B. Borah, D. Pathak, P. Sarmah, B. Som, S. Nandi, Survey of Textbased Chatbot in Perspective of Recent Technologies", Computational Intelligence, Communications, and Business Analyics. CICBA 2018. Communications in Computer and Information Science. Vol 1031. Singapore: Springer, 2019

[19] E. Reiter, R. Dale, *Building Natural Language Generation Systems*", Cambridge University Press, 2000