

Smart-Home-Lösungen mittels Sprachsteuerung und Raspberry Pi

STUDIENARBEIT

für die Prüfung zum
Bachelor of Science
des Studienganges Angewandte Informatik
an der
Dualen Hochschule Baden-Württemberg Karlsruhe

Abgabedatum 14.05.2017

Bearbeitungszeitraum	Theoriesemester 5 und 6
Autoren	Maximilian Hirte, Robin Warth
Matrikelnummern	8994521, 6028632
Kurs	TINF15B4
Ausbildungsfirma	Siemens AG Östl. Rheinbrückenstr. 50 76187 Karlsruhe
Betreuer der DHBW	Prof. Dr. Jürgen Röthig

Erklärung

Wir versichern hiermit, dass wir unsere Studienarbeit mit dem Thema: „Smart-Home-Lösungen mittels Sprachsteuerung und Raspberry Pi“ selbstständig verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel benutzt habe. Wir versichern zudem, dass die eingereichte elektronische Fassung mit der gedruckten Fassung übereinstimmt.

Ort Datum

Maximilian Hirte

Ort Datum

Robin Warth

Inhaltsverzeichnis

Abbildungsverzeichnis	V
Quellcodeverzeichnis	VI
Tabellenverzeichnis	VII
Abkürzungsverzeichnis	VIII
1 Einleitung - erster Entwurf	1
2 Aufgabenstellung	2
3 Grundlagen	3
3.1 Smart Home	3
3.2 Sprachverarbeitung	3
3.3 Analyse und Vergleich von digitalen Assistenten	5
3.3.1 Amazon Alexa	7
3.3.2 Apple Siri	9
3.3.3 Google Assistant	10
3.3.4 Microsoft Cortana	12
3.3.5 Samsung Bixby	13
3.4 Raspberry Pi	14
3.5 Funkstandards	14
3.6 OAuth2	15
3.6.1 Rollen	15
3.6.2 Protokollablauf	16
3.6.3 Berechtigungsvergabe	17
3.6.4 Authorization Code Grant	18
3.6.5 Bearer Token	20
3.7 Development-Tools	20
3.7.1 Frameworks	20
3.7.2 Amazon Developer Services	20
3.8 Gebrauchstauglichkeit	21
4 Konzept	23
4.1 Anforderungsanalyse	23

4.1.1	Grund der Umsetzung	23
4.1.2	Funktionale Anforderungen	23
4.1.3	Nicht-Funktionale Anforderungen	24
4.2	Herangehensweise	25
4.3	Architektur	25
4.4	Design	26
4.4.1	Skill-Typen	26
5	Implementierung	27
5.1	Einrichtung	27
5.1.1	Raspberry Pi	27
5.1.2	Python	28
5.1.3	Node.js	28
5.1.4	Cloud 9	28
5.1.5	Git	28
5.1.6	Amazon Developer Konto	28
5.1.7	Alexa	29
5.1.8	Amazon Web Services (AWS) Lambda	34
5.1.9	ngrok	35
5.2	Entwicklung	35
5.2.1	AWS Lambda	35
6	Ergebnis, Fazit und Ausblick	42
	Literatur	IX
	Anhang	XIV

Abbildungsverzeichnis

Abb. 3.1: Abstrakter Protokollablauf [RF6749]	16
Abb. 3.2: Authorization Code Grant [RF6749]	18
Abb. 4.1: Raspberry Pi-Alexa-Kommunikation	26
Abb. 5.1: Setup Skill-Information	29
Abb. 5.2: Setup Interaction Modell	30
Abb. 5.3: Setup Configuration Endpoints	31
Abb. 5.4: Setup Configuration Account Linking	32
Abb. 5.5: Setup Test	34

Quellcodeverzeichnis

5.1	Node.js Installieren	28
5.2	GitHub Repository	28
5.3	Request: Report-State	36
5.4	Request: Report-State	37
5.5	Response: Report-State	37
5.6	Request: Discovery	38
5.7	Request: Power-Controller	39
5.8	Response: Power-Controller	40

Tabellenverzeichnis

3.1	Formaler Vergleich der aktuell meist verbreitetsten Sprachassistenten	6
-----	---	---

Abkürzungsverzeichnis

API	Application Programming Interface
ARN	Amazon Resource Name
AVS	Alexa Voice Service
AWS	Amazon Web Services
GPIO	General Purpose Input/Output
HTTP	Hypertext Transfer Protocol
HTTPS	Hypertext Transfer Protocol Secure
IDE	Integrated Development Environment
JSON	JavaScript Object Notation
LED	lichtemittierende Diode
LWA	Login With Amazon
SSH	Secure Shell
WLAN	Wireless Local Area Network

1. Einleitung - erster Entwurf

Wir leben mittlerweile in einer Welt, in der jegliche Teilprozesse des täglichen Lebens bereits automatisiert sind bzw. automatisiert werden sollen, so auch das eigene Zuhause. Seit Anfang des Jahrtausends stellen sich Firmen und Institute die Aufgabe, das intelligente Zuhause zu entwickeln [VOE16] und schon mehrere Jahre ist der Begriff Smart Home in aller Munde. Es können jegliche Systeme innerhalb des Eigenheims zentral gesteuert und miteinander verknüpft werden. Über eine App kann dann der Benutzer seine persönlichen Einstellungen vornehmen und das Lebensgefühl somit auf eine neue Ebene setzen.

In letzter Zeit geht der Trend dahin, diese Hausautomatisierung per Sprachbefehle zu koordinieren. Zu diesem Zweck gibt es verschiedene unabhängige digitale Sprachassistenten, die dem Bewohner das Leben vereinfachen. Besonders Amazon-Alexa und der Google-Assistent ringen im Kampf um die Marktvormachtstellung und überzeugen beide im Bereich der Sprachverständnis und -verarbeitung, aber vor allem auch mit den Fähigkeiten, die sie bieten.

Jedoch bringt eine komplette Wohneinrichtung mit Systemen, die schon ab Werk die Verbindung zu den digitalen Assistenten gewährleisten, immense Kosten mit sich. Diese ließen sich reduzieren, wenn sich z. B. nicht nur die von Amazon und Google vorgesehenen und zu hohen Preisen angepriesenen Geräte koppeln lassen würden, sondern jegliche Systeme, Geräte und Anlagen, die für den Smart Home-Gebrauch geeignet sind.

Die Motivation dieser Studienarbeit ist somit, eine solche Kostenersparnis zu erzielen. Amazon bietet mit seinen Developer Services die Möglichkeit, Alexa mit beliebigen internetfähigen Geräten zu verbinden oder den Alexa Voice Service (AVS) direkt auf nicht-Amazon Geräten einzurichten. In dieser Arbeit haben wir es uns als Ziel gesetzt, diese durch Alexa gegebenen Möglichkeiten zu nutzen und somit einen Raspberry Pi mit Alexa zu verbinden. Mithilfe gängiger Funkstandards soll der Raspberry Pi weitere Geräte ansteuern, die ansonsten über keinerlei Funktionen verfügen mit digitalen Sprachassistenten zu interagieren. Neben der Verringerung der Kosten bietet eine solche Lösung eine weitere Stufe der Individualisierbarkeit und vor allem Geräte- bzw. Herstellerunabhängigkeit für die persönliche Smart Home-Zusammenstellung in den eigenen vier Wänden.

2. Aufgabenstellung

Die Aufgabe dieser Studienarbeit besteht darin, eine eigene Smart Home Lösung mittels des von Amazon bereitgestellten Sprachassistentens und eines Raspberry Pis zu entwickeln. Das Hauptaugenmerk liegt dabei das Repertoire des schon bestehenden Alexa-Systems um weitere Haushaltsgeräte zu erweitern. Diese besitzen entweder keine Möglichkeit einer Anbindung an Alexa (passender Alexa Skill) oder haben teilweise gar nicht die benötigte technische Infrastruktur, wie z. B. WLAN oder Rechenkapazität. Diese Anbindung erfolgt dann über den Raspberry Pi, der für diesen Zweck konfiguriert und angeschlossen wird. Die Mindestanforderung an das Projekt ist die Inbetriebnahme einer Funksteckdose, die dann über den Raspberry Pi mit einem Alexa Skill per Sprache ein- und ausgeschaltet werden kann. Im Anschluss soll dann auch eine Kamera an das System angeschlossen werden und per Sprachbefehle gesteuert werden.

Nach Möglichkeit soll dann ein Kosten- und Aufwandsvergleich zwischen der eigens konfigurierten und implementierten Variante sowie der teuren, dazu zu erwerbenden Kamera aus dem Amazonhaus aufgestellt werden.

Im Zuge der Ausarbeitung sollen dann neben Alexa auch andere derzeit aktuelle Sprachassistenten unter die Lupe genommen und miteinander verglichen werden. Dabei soll auch abgewägt werden, welche von den Konkurrenzprodukten von Amazon potentiell für die Verwendung in dieser Studienarbeit in Betracht gezogen werden könnten.

3. Grundlagen

In diesem Kapitel werden der Arbeit zu Grunde liegende Begriffe erklärt, Zusammenhänge zwischen diesen aufgezeigt sowie verschiedene Systeme vorgestellt und verglichen.

3.1. Smart Home

Unter dem Begriff Smart Home wird eine Menge von technischen Systemen, Geräten und Anlagen im Haushalt verstanden, die miteinander interagieren und sowohl zentral als auch dezentral bequem gesteuert werden können. Die damit verbundenen Ziele sind:

- Automatisierung von Alltagsvorgängen
- Verbesserung von Wohn- und Lebensqualität
- Erhöhung der häuslichen Sicherheit
- effiziente Energienutzung [DEV17]

Die gesamte Haushaltstechnik wird somit vernetzt und kann individuell angepasst und automatisiert werden. So können z. B. Geräteeinstellungen vorgenommen werden und über eine Schnittstelle, wie Computer oder Smartphone, verwaltet werden. Es ist somit z. B. möglich seine Heizung per Zeitsteuerung zu kontrollieren und auf eigene Bedürfnisse ideal zu zuschneiden. Per App auf dem Smartphone kann die Heizung, das Licht, der Fernseher oder jedes andere angeschlossene Gerät an- und abgeschaltet werden. Hierfür muss der Benutzer nicht einmal zu Hause sein. Prinzipiell lassen sich alle Geräte ansteuern, die über WLAN, Bluetooth, ZigBee oder andere Funkstandards verfügen. [SCH17]

Die nächste Stufe der Entwicklung ist dann die endgerätlose Steuerung. D. h., die Geräte auch zu steuern, ohne jedes mal den PC oder die App zu benutzen. Der Fokus wird immer mehr auf Gesten- und vor allem auf Sprachsteuerung gelegt, weswegen letzteres auch das Thema dieser Arbeit ist.

3.2. Sprachverarbeitung

Seit der zügigen Entwicklung der Computertechnik in der zweiten Hälfte des 19. Jahrhunderts gibt es das Problem und den Wunsch, natürliche Sprache mit Hilfe einer Maschine

zu erkennen und zu verarbeiten. In der heutigen Zeit ist Interaktion mit Maschinen so gut wie unumgänglich. Der herkömmliche Weg über Tastatur, Maus und Touchscreen ist für den Menschen immer noch unzureichend, denn das Hauptkommunikationsmittel ist die Sprache. [RAO17, S. 1]

Die Sprachverarbeitung unterteilt sich unter anderem in Sprachsynthese, also eine symbolischen Notation in ein Sprachsignal umzusetzen, und in Spracherkennung, die in dieser Arbeit hauptsächlich beleuchtet wird. Der Schwerpunkt hierbei liegt darin ein Sprachsignal oder auch Laut in eine textliche Form umzusetzen, sodass der Computer diese erkennt. Dies ist jedoch nach wie vor ein sehr großes Problem und wird auch nicht in den nächsten beiden Jahrzehnten so gelöst werden können, dass der Computer die menschliche Sprachwahrnehmungsfähigkeit komplett erreicht. Der Grund dafür ist die Vielfältigkeit und Komplexität der natürlichen Sprache. Auf der Ebene des Wortschatzes ist der Computer schon sehr gut aufgestellt und kann die meisten Wörter erkennen und unterscheiden, doch auf Ebene der Syntaktik und Semantik hat die Maschine bisher keine Chance. Der PC ist somit noch nicht in der Lage die Fähigkeit des Menschen, fortwährend zu untersuchen und zu entscheiden, ob das Gehörte Sinn ergibt, zu simulieren. [PFI17, S. 21]

Aus diesem Grund werden für Spracherkennungsassistenten spezielle Anwendungsszenarien konzipiert. Ein solcher digitale Assistent kann somit zwar keine vollständigen Konversationen führen, aber er verfügt zum Beispiel über die Fähigkeit einen Wetterbericht aus dem Internet zu suchen, wenn er nach dem morgigen Wetter gefragt wird. Durch solche Spezialisierung auf Anwendungsfälle werden bisher gute Resultate erreicht, da es weniger Erkennungsfehler gibt. Weiterhin ist diese Lösung kompakter, da sie sowohl weniger Speicher als auch Rechenleistung benötigt. [PFI17, S. 28]

Im Jahr 2017 hat die Mozilla Foundation das Projekt Common Voice durchgeführt, welches aus 400.000 validierten, englischen Sprachaufnahmen von 200.000 Personen besteht. Diese Aufnahmen ließen sie durch verschiedene Spracherkennungssoftware laufen, um somit die Fehlerquote bei der automatisierten Übersetzung der Sprachaufnahmen in Text zu minimieren. Ziel war es, eine Quote unter der Zehn-Prozent-Marke zu erreichen. Das erscheint für uns Menschen relativ viel, aber laut Forschungsergebnissen, auf die sich Mozilla beruft, liegt die Fehlerquote des menschlichen Sprachverständnis bei rund 6%. [MOZ17] Aus diesen Werten lässt sich also schließen, dass vorhandene Software im Bereich der Spracherkennung schon den menschlichen Fähigkeiten annähert.

Die maschinelle Erkennung und Verarbeitung erfolgt über Sprachsignale. Die genauen Algorithmen sowie der gesamte interne technische Ablauf der digitalen Assistenten sind nicht Teil dieser Arbeit und werden nicht weiter erläutert.

3.3. Analyse und Vergleich von digitalen Assistenten

Ein digitaler Sprachassistent (oft auch Intelligenter Persönlicher Assistent oder auch nur digitaler Assistent) ist ein System zur Sprachverarbeitung. Das System kann gesprochenen Anweisungen Folge leisten und zudem in menschlicher, meist weibliche, Stimme antworten. Die häufigsten Anwendungsfälle stellen dabei einfache Sprachsuchen im Internet oder Abarbeiten von einfachen Aufgaben, wie Schreiben und Versenden von Nachrichten oder Erstellen von Kalendereinträgen bzw. Erinnerungen dar.

Im Folgenden werden die zur Zeit fünf bedeutendsten Assistenten genauer untersucht und anhand folgender Kriterien miteinander verglichen:

- **Sprachsynthese** - Hier wird die Stimme des Assistenten bewertet. Es wird verglichen, welche Stimmen am menschlichsten und sympathischsten klingen. Die Offenheit für Humor zählt hier auch mit dazu.
- **Spracherkennung** - In diesem Punkt wird die Fähigkeit die Spracheingabe des Benutzers richtig zu verstehen verglichen. Muss der Befehl oftmals wiederholt werden schneidet der Assistent hierbei schlechter ab. Die Stimmen- bzw. Benutzererkennung spielt hierbei auch eine Rolle.
- **Verbreitung** - Vergleichsschwerpunkt ist hierbei, auf welcher Anzahl an Plattformen die jeweilige Software zur Verfügung steht.
- **Allgemeinwissen** - Wichtig für diesen Punkt ist die Fähigkeit, auf Wissenfragen bzw. Suchanfragen die richtige Antwort zu liefern, aber auch diese Antwort passend zu präsentieren (visuell, textlich oder mündlich). Die Bereitstellung von Lokalnachrichten, wie das Kinoprogramm, sowie Navigationsinformationen zählen hier auch mit rein.
- **Übersetzungsfähigkeit** - Hier wird verglichen, inwiefern die Assistenten in der Lage sind in andere Sprachen zu übersetzen und dies auch auszugeben. Ein weiterer Punkt ist einzelne Fremdwörter in Sätzen zu verstehen und zu verarbeiten. Ein Beispiel dafür ist englischsprachige Lieder abzuspielen. Dieser Punkt ist allgemein sehr interessant, ist aber speziell für diese Studienarbeit eher unwichtig.
- **Funktionsumfang** - Hier spielt neben den mitgelieferten Funktionen auch die Einbettung in das System eine große Rolle. D. h., inwiefern auf Drittanbieter-Software, wie Musik-Streaming- oder Messenger-Dienste, zugegriffen werden kann und wie gut der jeweilige Assistent an die vorliegende Hardware (z. B. smarterer Lautsprecher oder auch Mobiltelefon) angepasst ist und mit den dort vorhandenen Anwendungen kooperieren kann.

- Erweiterbarkeit - In diesem Punkt wird bewertet, ob und wie gut Benutzer den Funktionsumfang durch eigene, personalisierte Befehle (Skills, Actions etc.) erweitern können. Außerdem zählt die Anzahl der bereits bestehenden Erweiterungen mit hinein.
- Smart Home Unterstützung - Dies ist ein sehr wichtiger Punkt für diese Studienarbeit, denn hier wird verglichen, welche Assistenten Smart Home überhaupt unterstützen und wie gut dies geschieht. Der Punkt Erweiterbarkeit und die damit verbundene Personalisierung des eigenen Systems spielt hierbei auch hinein.
- Gender-Correctness - Dieser Punkt ist für den formalen Vergleich der Sprachassistenten eher unwichtig, aber durch die politische Korrektheit in der heutigen Zeit wurde dieser Punkt mit in dazugenommen. Hier wird angegeben, welche Software auch anbietet eine Stimme des männlichen Geschlechts auszuwählen.

In der folgenden Tabelle werden anhand der beschriebenen Merkmale die Assistenten Amazon Alexa [ALEXAa], Apple Siri [SIRI], Google Assistant [GOASS], Microsoft Cortana [MICOR] und Samsung Bixby [BIXBY] miteinander verglichen. Die Einstufung in den jeweiligen Kategorien wurde selbst vorgenommen und soll die Leistungsunterschiede schematisch darstellen. Die Begründung für die jeweilige Einstufung folgt in den nächsten Unterkapiteln.






	 amazon alexa	 Hey Siri	 Google ASSISTANT	 Hi. I'm Cortana.	 Bixby For Everyone
Sprachsynthese	✓✓✓	✓✓✓	✓✓	✓✓	✓✓
Spracherkennung	✓✓	✓✓	✓✓✓	✓✓	✓✓
Verbreitung	✓✓✓	✓✓	✓✓✓	✓✓	✓
Allgemeinwissen	✓✓	✓	✓✓✓	✓	✓✓
Übersetzungsfähigkeit	✓	✓	✓✓	✓✓✓	✗
Funktionsumfang	✓✓✓	✓✓✓	✓✓✓	✓✓	✓✓✓
Erweiterbarkeit	✓✓✓	✓	✓✓✓	✓✓	✗
Smart Home Unterstützung	✓✓✓	✓✓	✓✓✓	✗	✗
Gender-Correctness	✗	✓✓	✗	✗	✓✓

Tabelle 3.1.: Formaler Vergleich der aktuell meist verbreitetsten Sprachassistenten

Anzumerken ist, dass sich der Vergleich größtenteils auf die deutsche Version der Assistenten bezieht (ausgenommen: Bixby), da es möglicherweise Unterschiede zu anderen Sprachen gibt. Zudem ist klar, dass sich die Softwareprodukte während der Ausarbeitung weiterentwickeln. Dieser Vergleich basiert somit auf den Daten, die bis Februar 2018 zur Verfügung standen.

3.3.1. Amazon Alexa

Im Jahr 2015 kam mit den smarten Echo-Lautsprechern auch Alexa auf den Markt und reiht sich somit in der Historie hinter Siri und Google Now ein. Die größte Verbreitung genießt es auf den Echo-Geräten, aber auch auf weiteren Amazon-Geräten, wie Fire-TV oder Fire-Tablets. Der digitale Assistent lässt sich aber auch als App auf dem Smartphone installieren. [BAG17, S. 65]

Die Tester von Heise online, welche die vier bekanntesten Sprachassistenten unter die Lupe genommen haben, beschreiben Alexa mit einer „fast beängstigend guten Spracherkennung und mit einer angenehmen, freundlichen Stimme“. [BLE17, S. 81] Allgemein punktet der Assistent aus dem Hause Amazon in der Art der Kommunikation. Als tägliche Begleiterin im eigenen Heim ist sie sehr sympathisch und kann sogar auf Humor reagieren. Für ein gewisses Level an Spaß besitzt Alexa Fähigkeiten der Nachahmung von Filmcharakteren, wie Yoda aus Star Wars, oder auch Spaßfakten, wie die Zahl 42 als Antwort auf alles. Auf Basis der Kommunikation hat Alexa somit einen gewissen Vorsprung gegenüber dem Hauptkonkurrenten Google. [BLE17, S. 81]

Auf viele Wissensfragen hat Alexa eine gute, richtige Antwort parat. Jedoch sind Navigationsinformationen, wie der schnellste Weg nach Hamburg, noch sehr ausbaufähig. Sie ist aber auf ihren Echo-Lautsprechern bisher auch größtenteils auf den stationären Betrieb ausgelegt. Somit ist der Assistent im Bereich des Lokalservices wie z. B. dem Kinoprogramm der Stadt besser ausgerüstet, auch wenn manchmal die zugehörige App von Nöten ist. [BLE17, S. 81] Seit neustem gibt es dazu auch den Echo Show, ein Echo Gerät mit integriertem Touchscreen, auf welchem die Informationen noch visualisiert werden können. [BAG17, S. 65]

Fehlendes Wissen und Funktionen lassen sich durch Erweiterungen nachrüsten, welche Amazon Skills nennt. Mittlerweile gibt es ca. 3000 deutschsprachige und in den USA bereits über 16000 Skills (Stand: Oktober 2017 [BLE17]). Diese Skills sollen das Leben durch z. B. bestimmte Radiosender, Online-Kochbücher oder Fahrpläne per Sprachzugriff vereinfachen und verbessern. Solche Skills kann auch jeder Benutzer selber implementieren, indem er sich kostenlos bei den Amazon Developer Services registriert und somit zum Entwickler wird. Die Hauptstärke von Alexa liegt also genau in diesen erweiterbaren Skills, wodurch der Assistent immer weiter dazulernt. [BLE17, S. 81]

Das Nachsehen gegenüber der Konkurrenz hat Alexa vor allem beim Verständnis und der Interpretation von Sprachen, welche von der eingestellten abweichen. Versucht der Anwender z. B. bei einer auf Deutsch eingestellten Station einen englischen Titel per Spracheingabe auf dem Musik-Streaming-Dienst zu starten, kann es passieren, dass Alexa nicht das richtige Lied oder den passenden Interpreten abspielt. [BLE17, S. 81]

Weiterhin ist, obwohl der Assistent auf den Online-Shopping-Riesen Amazon optimiert sein soll, der Einkauf mittels Sprache nicht wirklich zufriedenstellend. Oftmals lassen sich Artikel nur in die Merkliste legen und nicht direkt bestellen. Vor allem bei Bekleidung und Schuhen tritt dieses Problem auf. Wenn der Benutzer jedoch einen Artikel bestellen will, den er in der Vergangenheit schon einmal erworben hat, hat Alexa keine Probleme. [BLE17, S. 81]

Ein letztes Manko von Alexa ist das Kontextverständnis. So tut sie sich sehr aufeinanderfolgende Fragen miteinander verknüpfen, was eine der Stärken von Google und dessen Deep Learning ist. Dennoch hat Amazon im Bereich der digitalen Assistenten noch die Nase vorn, da die Echo-Lautsprecher einige Monate vor dem Google Home auf den Markt kamen. Amazon konnte somit bereits länger Erfahrung sammeln und von der Anzahl an Skills profitieren, welche von Drittanbietern entwickelt wurden und Alexa kontinuierlich verbessern. [CLA18]

Amazon hat das Marktpotential im Bereich der Sprachassistenten erkannt und möchte somit sein Produkt so weit wie möglich verbreiten und auf so vielen Geräten wie möglich anbieten. Demnächst soll Alexa demzufolge auch direkt in Autos verbaut werden und somit die Hands-Free-Steuerung während der Fahrt erleichtern. Seat und BMW wollen den Assistenten in mehreren Modellen einbauen. Außerdem sprachen die Firmenbosse davon eine Alexa-Cortana-Verbindung zu erstellen und das hieße, dass Alexa dann auch auf Windows-Geräten erreichbar wäre. [BAG17, S. 66]

Amazon hat somit ein Produkt geschaffen, dass dem Benutzer das Leben vereinfachen kann und auch Zukunftspotential besitzt. Auch im Bereich von Smart Home bietet Alexa eine gute Unterstützung. Es gibt schon verschiedene Geräte die automatisch mit dem Assistenten gekoppelt werden können. Durch die benutzerdefinierten Skills können auch eigene Projekte realisiert werden. Beispielsweise eine Verbindung mit einem Raspberry Pi, welcher als Mittelsmann dient und weitere nicht-Alexa-Fähige Geräte ansteuern kann. Aus diesen Gründen ist Alexa auch ein optimaler Kandidat zur Verwendung in dieser Studienarbeit und wurde bereits zu Beginn favorisiert.

3.3.2. Apple Siri

Siri wurde im Jahre 2011 veröffentlicht und ist somit auch die älteste Assistentin. Sie ist nur auf den Apple-eigenen Plattformen iOS, macOS, watchOS sowie tvOS verfügbar. Seit Februar 2018 gibt es auch analog zu den Amazon und Google Lautsprechern einen Apple HomePod, auf dem mit Siri kommuniziert werden kann. [BAG17, S. 68]

Die Vorreiterin punktet vor allem durch die sehr gute Einbettung in das System an sich und der damit verbundenen Interaktion mit den von Apple mitgelieferten Apps auf dem Smartphone. Ohne Probleme lassen sich durch einfache Fragen bzw. Aufforderungen der Taschenrechner öffnen und Therme berechnen, eine Wettervorhersage einholen, neue Termine in den Kalender eintragen oder die Verabredungen für den heutigen Tag ausgeben lassen. Weiterhin können Emails oder Notizen diktiert, abgespeichert oder versendet werden. Mittlerweile gibt es auch eine ausgereifte Unterstützung für häufig genutzte nicht-Apple Apps, z. B. Messenger-Dienste wie WhatsApp. Wenn der Benutzer aber Dienste von externen Anwendungen anfordert, versucht Siri so gut wie immer auf Apple-eigene Apps zuzugreifen. Die angeforderten Ergebnisse sind daher oftmals unzureichend. Der Assistent ist ursprünglich nur für die Ersetzung der Touchbedienung auf den eigenen Geräten entwickelt wurden und dies ist auch sehr gut gelungen. Unter Apple Fans erfreut sich Siri daher sehr großer Beliebtheit. [BLE17, S. 85]

Im Bereich der Sprachausgabe überzeugt Siri durch eine angenehme und natürliche Stimme, welche über die Jahre hinweg immer weiter verbessert werden konnte. Die Tester von Heise online beschrieben Siri als „vertrauenerweckend sogar fast freundlich“, [BLE17, S. 85] was auch mit der Personalisierung zu tun hat. Denn der Benutzer kann sich von dem Assistenten mit seinem Namen ansprechen lassen. Zudem stellt sich Siri auf die Stimme des Anwenders ein, wodurch das Benutzererlebnis sowie die Sicherheit zusätzlich verbessert wird. Für Humor ist sie aber eher nicht offen, weswegen sie in normaler Kommunikation minder unterhaltsam abschneidet als ihrer Konkurrenten. Außerdem kann in den Einstellungen, wenn bevorzugt, auch genderkorrekt eine Männerstimme favorisiert werden, was bei weitem nicht alle digitale Assistenten unterstützen. [BLE17, S. 85]

Ähnlich wie mit den Alexa-Skills und die Actions von Google bietet Apple mit dem SiriKit die Chance, selbst zum Entwickler zu werden. Die Möglichkeiten sind aber im Gegensatz zur Konkurrenz eher beschränkt, da im Prinzip nur Erweiterungen für bestehende Anwendungen implementiert werden können, um diese mit Sprache zu steuern. Die Art von Applikationen, die erweitert werden können, sind mit einer genauen Instruktion in der SiriKit-Dokumentation zu finden. [SKDOC] Apple ist somit für freie Erweiterungen nicht so offen wie die Konkurrenz.

Minuspunkte sammelt Siri im Bereich von Übersetzungen sowie im Fremdsprachenverständnis. In diesen Punkten hinkt sie im Vergleich mit den anderen Assistenten weit hinterher bzw. hat Siri noch Schwierigkeiten einzelne Wörter in andere Sprachen übersetzen. Zudem weist die Apple-Lösung gegenüber Google und Amazon klare Wissenslücken auf. Oftmals bekommt der Anwender gar keine oder falsche Antworten (nicht zur Fragestellung passend) und Wissensfragen werden willkürlich sprachlich oder visuell beantwortet, was für den Benutzer unverständlich und auf Dauer nervig ist. Auch ortsbezogene Suchanfragen sind noch nicht wirklich ausgereift. Navigation und Restaurantvorschläge in der Nähe funktionieren einwandfrei. Auf Anfragen, wie örtliches Kinoprogramm, können jedoch nur selten zugegriffen werden. [BLE17, S. 85]

Zusammenfassend lässt sich sagen, dass Siri als älteste Assistentin zwar ausgereift ist, aber dennoch einige Schwächen aufweist und sowohl Amazon als auch Google sie in manchen Bereichen überholt haben. Wie von Apple bekannt, funktioniert alles solange gut, sofern Apple-Produkte und Software benutzt wird. So auch im Bereich von Smart Home. Auch hier ist der Anwender auf das firmeneigene HomeKit angewiesen, was auch in diesem Bereich eher Minuspunkte sammelt und sich somit nicht für den Gebrauch in dieser Studienarbeit anbietet. [BLE17, S. 85]

3.3.3. Google Assistant

Schon im Jahr 2012 kam Google Now auf den Markt, die Vorversion des heutigen Google Assistant. Somit konnte sich auch der Sprachassistent von Google schon einige Jahre auf dem Bereich der Spracherkennung und -verarbeitung entwickeln. Der Assistent ist auf jeglichen Android-Geräten seit Android 6.0 erhältlich und mittlerweile gibt es sogar eine iOS-Version für die Apple Smartphones. Nachdem der Internetriese immer mehr Marktanteile an Amazon abgeben musste und die Google-Suchanfragen seit der Veröffentlichung des Echo-Lautsprechers zurückgingen, erkannten sie das Potenzial von smarten Lautsprechern. Ende 2016 (2017 mit deutscher Sprachenunterstützung) brachte Google daher mit ihrem Google Home einen eigenen smarten Lautsprecher auf den Markt. [BAG17, S. 66 f]

Der Assistent punktet vor allem durch herausragende Smartphone-Integration, sodass die applikationsübergreifende Interaktion sehr gut funktioniert. Per Sprache kann die Kamera geöffnet werden, eine Whatsapp-Nachricht diktiert und versendet sowie sogar der Flugmodus aktiviert werden, was ein selbstständiges Schließen bzw. Beenden des Assistenten zur Folge hat. [BLE17, S. 84]

Weitere Vorteile von Google sind auf dessen riesigen Datenfundus zurückzuführen und somit ist der Google Assistant auch im Allgemeinwissen marktführend. Er kann z. B. bei

ortsbezogenen Informationen, wie Kinoprogramm oder auch Navigation, die Konkurrenz ausstechen. Die Entscheidung, welche der Assistent über die Form der Ausgabe fällt, also die mündliche oder visuelle Ausgabe des Ergebnisses, wirkt intuitiv und gut durchdacht. Weiterhin zählt der Assistent genauso wie Cortana zu den Sprachgenies und kann sogar ganze Sätze übersetzen. [BLE17, S. 84]

Ein Alleinstellungsmerkmal ist die Stimmenerkennung, denn der Google Assistant ist bisher als einziger in der Lage, Stimmen zu unterscheiden und kann somit sogar mehrere Benutzer differenziert bedienen. [CLA18] Hinzu kommt, dass der Assistent kontextbezogene Nachfragen ermöglicht. Google analysiert jede einzelne Frage und merkt sich die Informationen darin, um später einen Zusammenhang zu folgenden Fragen zu haben. Das beherrscht der Assistent eindeutig besser als die Konkurrenz und somit lassen sich annähernd vollständige Gespräche führen. [BLE17, S. 84]

Aufgrund der langen Laufzeit und dem riesigen Entwicklungsaufwand seit der Erstveröffentlichung ist auch die Sprachsynthese so gut, dass man alles versteht. Jedoch hat der Google Assitant auch in der aktuellen Version noch eine recht roboterhafte Stimme. Die Heise online Tester beschrieben die Stimme als „etwas hochnäsiger und humorarm“. [BLE17, S. 84] Hierbei hinterlässt Alexa einen wesentlich besseren Eindruck. Bei der Kommunikation spielt das Gefühl eben eine große Rolle. Der Benutzer urteilt hier sehr subjektiv und empathisch und redet natürlich lieber mit jemandem, der eine freundliche Stimme hat. [BLE17, S. 84]

Analog zu den Alexa Skills gibt es für den Google Assistant die Actions on Google, bei denen benutzerdefinierte Anwendungen implementiert und hinzugefügt werden. Bislang gibt es aber bei weitem weniger vorhandene und vor allem auch kaum deutschsprachige Actions, da Amazon mit ihrem Produkt einfach viel früher dran war und hier klare Marktvorteile genoss. Doch auch mit der geringeren Anzahl an Actions deckt Google mit Hilfe ihrer Suchmaschine ein weites Spektrum an Anwendungsfällen ab und ist somit auch in diesem Bereich auf dem Vormarsch. [CLA18]

Im Bereich von Smart Home nehmen sich beide Assistenten, also von Google und Amazon, nicht viel. Sie unterstützen ähnlich viele Geräte und lassen sich auch noch erweitern. Vorteile hat nur Google bei der Sprachbedienung, da diese oftmals intuitiver erscheint. [CLA18]

Der Google Assistant und Alexa dominieren aktuell größtenteils den Markt. Noch hat Alexa aufgrund der früheren Veröffentlichung gewisse Vorteile an Verbreitungsgrad im Bereich Smart Home, die es für Google gilt aufzuholen. Durch den riesigen Deep Learning Mechanismus von Google dürfte dies nicht mehr all zu lange dauern. Für diese Studienarbeit würde ebenfalls der Google Assistant in Frage kommen. Aufgrund der Verbreitung

und vermehrt subjektiver Entscheidungen wurde die Entscheidung zugunsten Alexa schon im Vorfeld der Studienarbeit getroffen.

3.3.4. Microsoft Cortana

Im Jahr 2014 kam mit Cortana Microsofts Antwort im Bereich der Sprachassistenten auf den Markt. Cortana ist hauptsächlich auf Geräten mit Windows 10 Betriebssystem zu finden. Der Assistent wird hier direkt mitgeliefert.

Cortanas Alleinstellungsmerkmal gegenüber ihren Konkurrenten ist ihre Fähigkeit zu übersetzen, denn sie kann neben Wörtern auch Sätze per Sprachbefehl durch den Microsoft Translator in verschiedenste Sprachen übersetzen und in den meisten Fällen das Ergebnis auch vorlesen. In diesen Tests schneidet der Assistent sogar besser ab, als der Googles Assistant. Ähnlich wie Siri ist Cortana auf das Betriebssystem sehr gut zugeschnitten und somit lassen sich Programme ohne Probleme per Sprachsteuerung öffnen. [BLE17, S. 82]

Im Vergleich zu den Konkurrenzprodukten muss Microsoft aber in einigen Punkten zurückstecken. Zwar wirken Cortanas Antworten nicht künstlich, sondern sie zeigt sogar etwas Empathie und Mitgefühl, aber dennoch klingt ihre Stimme hölzern. Für den Benutzer ist dies sehr gewöhnungsbedürftig. [BLE17, S. 82] Weiterhin ist die Anzahl der unterstützten Sprachbefehle gegenüber der Assistenten der Konkurrenz sehr eingeschränkt. Um diese Lücke zu stopfen gibt es eine von Microsoft bereitgestellte Programmierschnittstelle, aber im Gegensatz zu Amazon und Google ließen sich hier noch nicht viele externe Programmierer finden. Somit gibt es immer noch weniger als 100 Skills, die hinzugekommen sind und das auch nur auf Englisch und somit bisher nicht für den deutschsprachigen Raum bestimmt. [BAG17, S. 67]

Auch im Bereich von Sprachsuchen hat Cortana das Nachsehen gegenüber Google, Amazon, Apple und Samsung. Viele Antworten kann sie nicht selbst geben und öffnet stattdessen den Browser und überlässt dem Benutzer die Arbeit bei der Suche via Suchmaschine. Dabei benutzt sie immer Edge als Webbrowser und Bing als Suchmaschine, was sich auch nicht umstellen lässt. Minuspunkte gibt es nicht nur für diesen Zwang, sondern auch dafür, dass sie die Antworten, welche sie nicht selber beantworten kann, oftmals im Browser auch gar nicht vorliest. Edge muss der Benutzer auch selbst wieder schließen, was nicht sonderlich benutzerfreundlich ist. [BLE17, S. 82]

Die Sprachsuchergebnisse sind zudem auch selten zufriedenstellend. Hierzu ein Beispiel: Auf die Frage „Seit wann gibt es dich?“ öffnet sie die Homepage www.seid-seit.de, was natürlich erstens nichts mit der Fragestellung zu tun hat und zweitens der Benutzer zusätzlich noch das Fenster selbst wieder schließen muss.

Dies alles wird auch der Grund sein, warum Cortana gegenüber den anderen Assistenten wenig Beachtung geschenkt wird. Jedoch soll noch im Laufe des Jahres 2018 eine Kooperation zwischen den beiden Freundinnen Cortana und Alexa ermöglicht werden. Somit könnte Alexa mit einem Sprachbefehl aufgefordert werden Cortana zu öffnen und andersherum. Die Entwickler erhoffen sich dadurch die Schwächen gegenseitig auszumerzen. Somit könnte der Benutzer dann über Cortana bei Amazon einkaufen. Zudem ist einer der wichtigsten Punkte, dass Cortana somit erstmals auf einem intelligenten Echo-Lautsprecher verfügbar sein wird und dadurch auch zum ersten Mal mit Smart Home in Verbindung gebracht wird. Bis dato kommt die Verwendung von Cortana aber nicht für diese Studienarbeit in Frage.

3.3.5. Samsung Bixby

Der jüngste unter den digitalen Assistenten ist die im Jahr 2017 von Samsung veröffentlichte Sprachassistentin Bixby. Diese ist bisher nur auf den Samsung Smartphones Galaxy S9, Galaxy S8 und durch eine Zusatzinstallation auf den Samsung Galaxy S7 Geräten verfügbar und genießt damit eine sehr beschränkte Verbreitung. Die Installation besteht aus Bixby Voice, den eigentlichen Sprachassistenten, Bixby Home, einer digitalen Pinnwand und Organisationseinheit und Bixby Vision. Letzteres ist in der Lage ein Objekt mit der Kamera oder in der Galerie App zu scannen und zu diesem Objekt im Internet Informationen zu sammeln. Hierbei können auch Shops gesucht werden, welche das gescannte Objekt verkaufen. [WIG17]

Zunächst wurde der Assistent nur auf Koreanisch und Chinesisch veröffentlicht, aber mittlerweile wurde auch die englische Sprache hinzugefügt. Deutsch fehlt noch immer, wodurch sich der geringe Verbreitungsgrad hierzulande von selbst erklärt.

Ähnlich wie bei Siri kann der Benutzer bei Bixby zwischen verschiedenen Stimmen wählen. Tester beschrieben die Stimme Stephanie mit einem angenehmen Klang, John mit einer sympathischen Stimme und Julia mit einer tieferen Aussprache. Alle drei zeichnen sich aber jeweils durch eine klare und deutliche Kommunikation aus. [REI17]

Beim Sprachverständnis hat der Assistent jedoch noch seine Probleme. In den Einstellungen kann zwischen drei Empfindlichkeitsstufen ausgewählt werden, aber die höchste kommt bei weitem nicht an Google heran. Somit muss das Schlüsselwort "Hi Bixby" öfters lauter wiederholt werden. Weiterhin gibt es einige Schwierigkeiten mit Dialekt, aber dafür wird die Stimme mit der Zeit gelernt und auch ein manuelles Sprachtraining ist möglich. [REI17]

Im Bereich von übergreifender Interaktion kann Bixby ordentlich punkten. Allgemeine Dinge wie Wecker, Wetter sowie Youtube funktionieren einwandfrei und teilweise

sogar besser als bei allen anderen Konkurrenzprodukten. So schießt der Befehl "take a photo" direkt ein Foto und öffnet nicht nur wie Google die Kamera. In den Feedback-Einstellungen kann der Benutzer zu dem entscheiden, ob die Antworten auf Fragen akustisch, visuell oder ausschließlich textlich sowie kurz oder lang beantwortet werden sollen. [REI17]

Allgemein lässt sich sagen, dass Bixby bisher wenig Zeit hatte sich zu beweisen. Dafür müsste es aber auch für mehr Sprachräume und Geräte zugänglich gemacht werden. Zudem hat es einige Nachteile, da auf denselben Smartphones auf denen Bixby erhältlich ist auch Google mit deren Produkt vertreten ist. Im direkten Vergleich schneidet Bixby durch das schlechtere Sprachverständnis und fehlendes Verknüpfen mit vorhergehenden Fragen hier schlechter ab. Dennoch hat Bixby großes Potential und kann durch mehr Erfahrung auf dem Markt, den Vorsprung der Konkurrenz aufholen. [HAR17] Da bisher auch keine Smart Home Unterstützung verfügbar ist, scheidet der Assistent ebenfalls für diese Studienarbeit aus.

3.4. Raspberry Pi

[folgt]

3.5. Funkstandards

ISM-Band von 433,05 bis 434,79 MHz

Industrial, Scientific, Medical Band

Wir benutzen 433 MHz (70-Zentimeter-Band)

Betriebsfunk und Amateur-Funk, Handfunkgeräte

Als ISM-Band für z.B. Funkswitcher, Funkthermometer, Funksteckdosen

Max. Leistung von 10mW

Reichweite 0,3 in Stadt, 2,5 auf freier Fläche (quasioptisch)

69 Kanäle, gebührenfrei, anmeldefrei

«Non specific Short-Range-Devices» (SRD) – Geräte geringer Sendeleistung und kurzer Sendereichweite – dürfen ohne weitere Frequenzuteilung betrieben werden, sofern sie die technischen Voraussetzungen erfüllen

Geräte für Anwendungen kommen sich selten in die Quere, trotz große Verbreitung (Garagentoröffner, Autoschlüssel..), da Aktivierungsradius gering ($\approx 100\text{m}$) und die Nutzungsdauer kurz (0,1 bis 3s). Trotzdem kann vorkommen dass 433,92MHz blockiert ist, wegen z.B. Funkkopfhörer, der im Umkreis von 50m auf derselben Frequenz arbeitet

nachteile: reichweite, kosten und stromverbrauch

nutzung nicht beschränkt -> störungen durch nachbarn wahrscheinlich
sichere übertragung nur bei leistungsfähiger kanalcodierung, module mit mehreren kanäle
haben deutliche vorteile, indem sie auf andere frequenz ausweichen können
(<http://rn-wissen.de/wiki/index.php?title=Funkmodule>)

3.6. OAuth2

Das Autorisierungsframework und offene Protokoll OAuth2 (Open Authorization 2) ist ein Industriestandard [RF6749] für sichere API-Autorisierung in den Bereichen Mobile-, Web-, Heim- und Desktop-Anwendungen. OAuth2 ist der Nachfolger des 2006 erstellten Protokoll OAuth. Das Framework ermöglicht es einer Drittanbieter-Anwendung einen eingeschränkten Zugriff auf einen HTTP(S)-Service zu erhalten. Dies geschieht entweder direkt im Namen eines Ressourceneigentümers, durch Orchestrierung einer Genehmigungsinteraktion zwischen dem Ressourceneigentümer und dem HTTP-Service, oder indem der Drittanbieter-Anwendung der Zugriff auf den Service nach eigenem Ermessen gewährt wird. [OAUTHa]

3.6.1. Rollen

Gemäß der Spezifikation von OAuth2 [RF6749] werden die Beteiligten in vier verschiedene Rollen unterteilt [OAUTHb]:

- Client (Drittanbieter-Anwendung)
 - Der Client ist die Anwendung, die versucht, Zugriff auf das Benutzerkonto des Benutzers zu erhalten. Es muss die Erlaubnis des Benutzers einholen, bevor es dies tun kann.
- Ressourcenserver (API)
 - Der Ressourcenserver ist der API-Server, der für den Zugriff auf die Informationen des Benutzers verwendet wird.
- Autorisierungsserver
 - Dies ist der Server, der die Schnittstelle darstellt, auf der der Benutzer die Anfrage genehmigt oder ablehnt. Die Interaktion zwischen Autorisierungsserver und Ressourcenserver wird über das Protokoll nicht weiter spezifiziert. In kleineren Implementierungen kann dies derselbe Server wie der API-Server sein,

aber bei größeren Implementierungen wird dies oft als separate Komponente aufgebaut.

- Ressourcen-Eigentümer (Benutzer)
 - Der Ressourcen-Eigentümer ist eine Einheit, die Zugang zu einem Teil einer geschützten Ressource gewährt kann. Meistens ist dies eine Person als Endbenutzer, welche Zugriff auf Teile ihres Accounts gewährt.

3.6.2. Protokollablauf

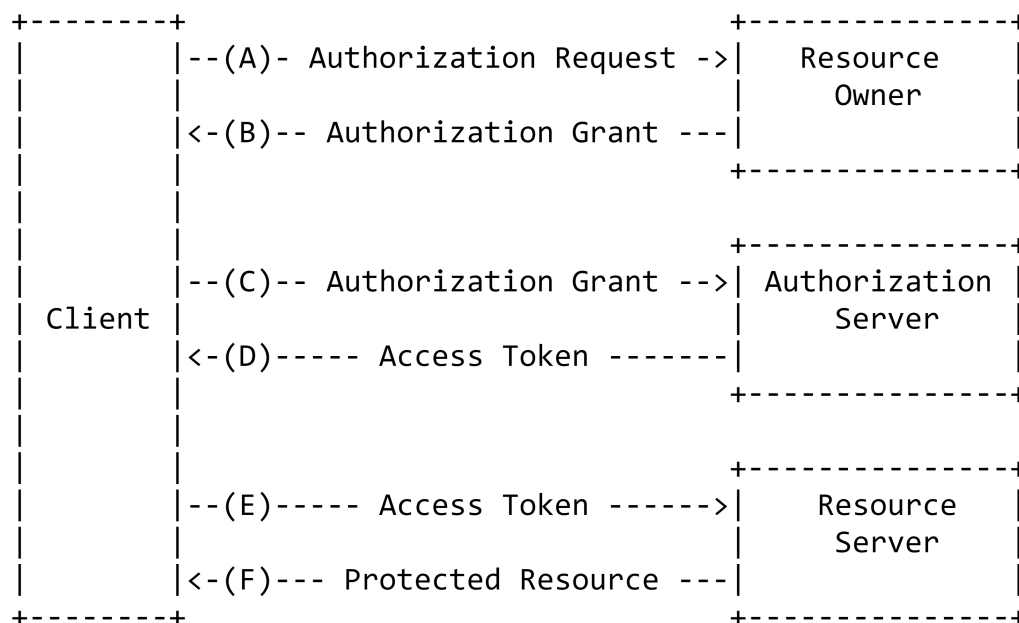


Abb. 3.1.: Abstrakter Protokollablauf [RF6749]

Der in Abbildung 3.1 dargestellte OAuth2 Protokollablauf beschreibt die Interaktion zwischen den vier Rollen und umfasst die folgende Schritte [RF6749]:

- (A) Autorisierungsanfrage (Authorization Request)
 - Der Client fordert die Berechtigung des Ressourceneigentümers an. Die Berechtigungsanfrage kann direkt an den Ressourceneigentümer gestellt werden, wie abgebildet, oder vorzugsweise indirekt über den Autorisierungsserver als Vermittler.
- (B) Berechtigungsvergabe (Authorization Grant)
 - Der Client erhält eine Berechtigungsvergabe, welche ein Berechtigungsnachweis für die Autorisierung durch den Ressourceneigentümers darstellt. Hierfür gibt es vier, durch OAuth2 spezifizierte, verschiedene Verfahren zur Berechtigungsvergabe. Hierzu mehr in Abschnitt 3.6.3.

- (C) Berechtigungsvergabe (Authorization Grant)
 - Der Client fordert ein Access-Token an, indem er sich beim Autorisierungsserver authentifiziert und Berechtigungsvergabe vorhält.
- (D) Zugriffstoken (Access Token)
 - Der Autorisierungsserver authentifiziert den Client und validiert die Berechtigungsvergabe und stellt, falls gültig, ein Zugriffstoken aus.
- (E) Zugriffstoken (Access Token)
 - Der Client fordert die geschützte Ressource von dem Ressourcenserver an und authentifiziert sich dabei, durch Vorlage des Zugriffstoken.
- (F) Geschützte Ressource (Protected Resource)
 - Der Ressourcenserver validiert das Zugriffstoken und gibt, falls gültig, die geschützten Ressourcen zurück. Diese können z. B. die Identifikation des Benutzers im System, der Benutzername, die E-Mail Adresse oder weitere Daten sein.

3.6.3. Berechtigungsvergabe

Die Art der Berechtigungsvergabe hängt von der Methode ab, die der Client benutzt um eine Autorisierung anzufordern und von den Typen, die vom Autorisierungsserver unterstützt werden. Es werden hierfür fünf verschiedene Verfahren in OAuth2 beschrieben:

- Autorisierungscode Erteilung (Authorization Code Grant)
- Implizite Erteilung (Implicit Grant)
- Kennwortberechtigungen für Ressourceneigentümer gewähren (Resource Owner Password Credentials Grant)
- Kundenanmeldeinformationen gewähren (Client Credentials Grant)
- Erteilung durch Erweiterung (Extension Grants)

Die bevorzugte Methode für den Kunden, eine Berechtigungsvergabe vom Ressourceneigentümer zu erhalten (dargestellt durch die Schritte (A) und (B) in Abbildung 3.1), ist die Verwendung des Autorisierungsserver als Intermediär. Dieses Verfahren wird in der OAuth2 Spezifikation Authorization Code Grant genannt.

Das Alexa Skills Kit unterstützt sowohl Authorization Code Grant (für Custom-Skills und Smart Home-Skills), als auch Implicit Grant (nur für Custom-Skills). Da in dieser

Studienarbeit auf die Smart Home-Skill API zurückgegriffen wird, ist die Verwendung des Authorization Code Grant Verfahren unerlässlich.

Im Folgenden wird nun nur auf das Authorization Code Grant Verfahren näher eingegangen, da die anderen Verfahren für die Studienarbeit nicht von Bedeutung sind.

3.6.4. Authorization Code Grant

Das Authorization Code Grant Verfahren wird verwendet um die beiden Zugriffsarten, Zugriffstoken und Aktualisierungstoken, zu erhalten. Das Verfahren ist für vertrauliche Clients optimiert. Da es sich hierbei um einen redirektionsbasierten Ablauf handelt, muss der Client in der Lage sein, folgende Aufgaben zu erfüllen:

- Interaktion mit dem User-Agenten des Ressourcen-Eigentümers (typischerweise ein Web-Browser)
- Eingehende Anfragen vom Autorisierungsserver empfangen (über Weiterleitung im Browser)

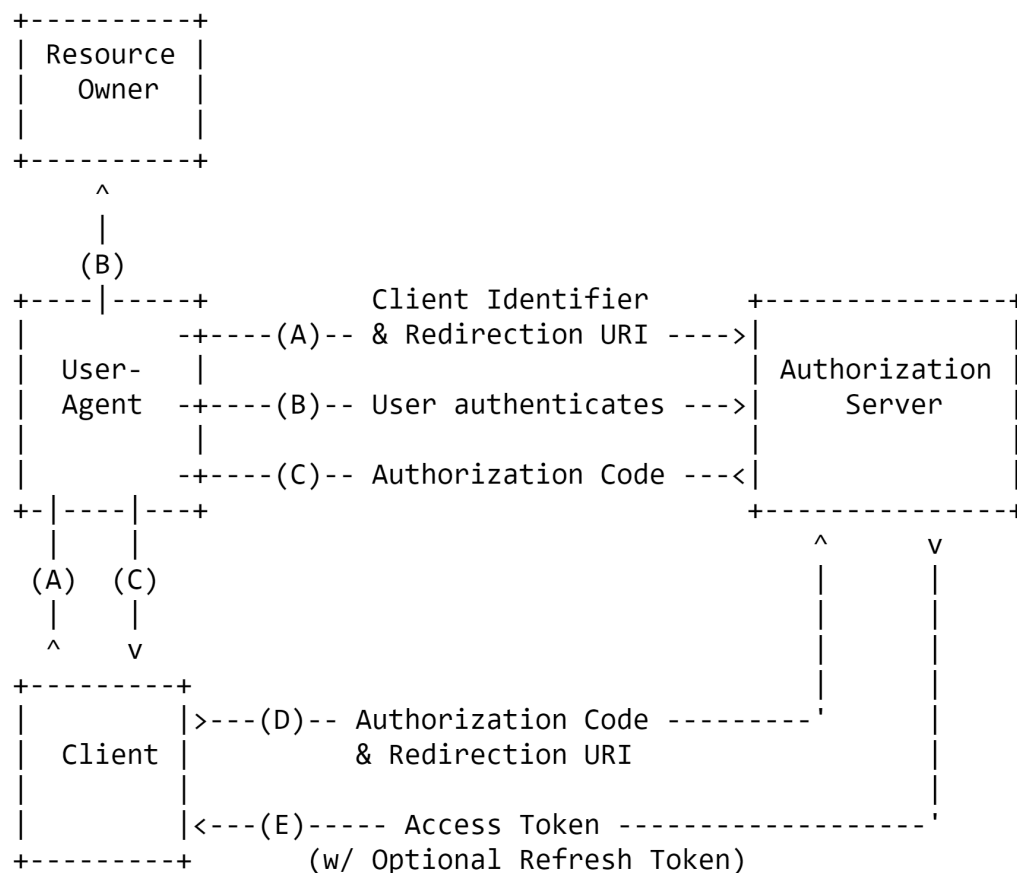


Abb. 3.2.: Authorization Code Grant [RF6749]

Das in Abbildung 3.2 dargestellte Verfahren beinhaltet die Folgenden Schritte (hier zu Beachten: Schritt A, B und C sind in zwei geteilt, da sie durch den User-Agenten geschleust werden) [RF6749]:

- (A) Client Identifikation & URI Umleitung (Client Identifier Redirection URI)
 - Der Client initiiert den Ablauf, indem er den User-Agent des Ressourceneigentümer mit dem Autorisierungsserver verbindet. Der Client verschickt seine Kundenkennung (Client Identifier), den angeforderten Bereich (Requested Scope), den lokalen Zustand (Local State) und eine Umleitungs-URI (Redirection URI), an die der Autorisierungsserver den User-Agent zurück schickt, sobald der Zugriff gewährt (oder verweigert) ist.
- (B) Benutzer authentifizieren (User authenticates)
 - Der Autorisierungsserver authentifiziert den Ressourceneigentümer (über den User-Agent) und stellt fest, ob der Ressourceneigentümer die Zugriffsanfrage des Clients gewährt oder verweigert.
- (C) Autorisierungscode Rückgabe (Authorization Code)
 - Unter der Annahme, dass der Ressourcen-Eigentümer den Zugriff gewährt, leitet der Autorisierungsserver den User-Agenten über die zuvor, in der Anfrage oder bei der Client-Registrierung, bereitgestellte Umleitungs-URI an den Client zurück. Die Umleitungs-URI enthält einen Autorisierungscode und alle lokalen Zustände, die der Kunde zuvor angegeben hat.
- (D) Übergabe von Autorisierungscode & URI Umleitung (Authorization Code Redirection URI)
 - Der Client fordert vom Token-Endpunkt des Autorisierungsservers ein Zugriffstoken an, indem er den im vorherigen Schritt erhaltenen Autorisierungscode der Anfrage beifügt. Bei der Anfrage authentifiziert sich der Client beim Autorisierungsserver somit durch den Autorisierungscode. Zudem fügt der Client die Umleitungs-URI, welche er genutzt hat um den Autorisierungscode zu erhalten, seiner Anfrage an, um sich zusätzlich zu verifizieren.
- (E) Rückgabe von Zugriffstoken mit optionalem Aktualisierungstoken (Access Token with Optional Refresh Token)
 - Der Autorisierungsserver authentifiziert den Client, validiert den Autorisierungscode und stellt sicher, dass die empfangene Umleitungs-URI mit der URI übereinstimmt, welche für die Umleitung des Clients in Schritt (C) verwendet wurde. Falls gültig, antwortet der Autorisierungsserver mit einem Zugriffstoken (Access-Token) und optional mit einem Refresh-Token.

3.6.5. Bearer Token

Das Bearer Token (Inhaberkennzeichen) [RF6750] ist eine genauer spezifizierte Variante des bereits erwähnten Zugriffstoken (Access-Token). Ein Zugriffstoken wird durch den OAuth2 Standard [RF6749] wie folgt beschrieben: „a string representing an access authorization issued to the client“, also eine Zeichenkette, die eine Zugriffsberechtigung repräsentiert, die dem Client erteilt wurde. Mithilfe von Zugriffstoken wird dem Client somit ermöglicht, sich bei einem Autorisierungsserver, ohne Anmeldedaten wie z. B. Benutzername und Passwort, zu authentifizieren.

3.7. Development-Tools

[folgt]

3.7.1. Frameworks

[folgt]

3.7.2. Amazon Developer Services

Mit den Amazon Developer Services wird jedem Benutzer, der eine Mailadresse besitzt, die mit einem Amazon-Konto verknüpft ist, die Möglichkeit geboten, eigene Applikationen für jegliche Geräte auf Android oder iOS basierend oder für die Amazon eigenen Fire Tablets und Fernseher kostenlos zu entwickeln, zu builden, zu testen und sogar über den Amazon Appstore zu vertreiben. Nach der Registrierung eines Amazon Developer Accounts stehen dem Entwickler etliche Services und APIs zur Verfügung, die direkt eingebunden und benutzt werden können. Das umschließt beispielsweise vorgefertigte Login-Formulare durch Amazon-Konten, Werbung für die eigene Applikation auf den Amazon-Fire-Geräten zu schalten oder Multi-Screen-Möglichkeiten mit zwei-Wege-Kommunikation zwischen Fire TV und der Smartphone Anwendung. Weiterhin werden verschiedenste Plugins und Extensions wie Unity, Apache Cordova oder Xamarin unterstützt. [ROU17a]

Das für dieses Projekt wichtigste Werkzeug ist das Alexa Skills Kit. Dieses ermöglicht es eigene Skills, also eine Art Fähigkeit bzw. App, selbst zu erstellen und zu programmieren. Mit selbst erstellten oder von Drittanbieter stammenden Skills kann Alexa auf bestimmte eigens konzipierte Anwendungsfälle, durch Erkennung bestimmter Wörter, erkennen und auf diese reagieren. Diese selbsterstellten Skills können dann auch über den Alexa Skills Store vertrieben werden. [ROU17b]

Über die eigenen Skills können dann über die Verbindung mittels des Raspberry Pi die

angeschlossenen Geräte wie Funksteckdosen oder die Kamera, welche sonst nicht von Alexa aus erreichbar wären, angesprochen werden.

3.8. Gebrauchstauglichkeit

Gemäß „DIN EN ISO 9241“ [ISO06] sind die Kriterien für die Gebrauchstauglichkeit (Usability) über die sieben „Grundsätze der Dialoggestaltung“ beschrieben:

- Aufgabenangemessenheit
 - „Ein interaktives System ist aufgabenangemessen, wenn es den Benutzer unterstützt, seine Arbeitsaufgabe zu erledigen.“
- Selbstbeschreibungsfähigkeit
 - „Ein Dialog ist in dem Maße selbstbeschreibungsfähig, in dem für den Benutzer zu jeder Zeit offensichtlich ist, in welchem Dialog, an welcher Stelle im Dialog er sich befindet, welche Handlungen unternommen werden können und wie diese ausgeführt werden können.“
- Erwartungskonformität
 - „Ein Dialog ist erwartungskonform, wenn er den aus dem Nutzungskontext heraus vorhersehbaren Benutzerbelangen sowie allgemein anerkannten Konventionen entspricht.“
- Lernförderlichkeit
 - „Ein Dialog ist lernförderlich, wenn er den Benutzer beim Erlernen der Nutzung des interaktiven Systems unterstützt und anleitet.“
- Steuerbarkeit
 - „Ein Dialog ist steuerbar, wenn der Benutzer in der Lage ist, den Dialogablauf zu starten sowie seine Richtung und Geschwindigkeit zu beeinflussen, bis das Ziel erreicht ist.“
- Fehlertoleranz
 - „Ein Dialog ist fehlertolerant, wenn das beabsichtigte Arbeitsergebnis trotz erkennbar fehlerhafter Eingaben entweder mit keinem oder mit minimalem Korrekturaufwand seitens des Benutzers erreicht werden kann.“
- Individualisierbarkeit

- „Ein Dialog ist individualisierbar, wenn Benutzer die Mensch- System-Interaktion und die Darstellung von Informationen ändern können, um diese an ihre individuellen Fähigkeiten und Bedürfnisse anzupassen.“

4. Konzept

In diesem Abschnitt geht es um den grundsätzlichen Entwurf und das Konzept des praktischen Teils der Studienarbeit. Hier wird dargestellt, welches konkrete Ziel mit der Studienarbeit verfolgt und mit welchen Mitteln es erreicht werden soll.

4.1. Anforderungsanalyse

In der Anforderungsanalyse wird beschrieben, welche funktionalen und welche nicht-funktionalen Anforderungen an das Projekt gestellt sind.

4.1.1. Grund der Umsetzung

[folgt]

4.1.2. Funktionale Anforderungen

Nach Beendigung des Projekts soll die Möglichkeit bestehen, Geräte per Sprachbefehle ein und aus zu schalten. Einige der Geräte sollen über Funksteckdosen, als Steuerelement, an das Smart Home angebunden werden. Hierfür stehen beliebig viele Geräte zur Verfügung, wie beispielsweise Lampen, LED-Streifen, Ventilatoren und weitere Stromverbraucher.

Zudem soll eine Kamera, angeschlossen an einen Raspberry Pi, über Sprachbefehle bedient werden können. Es sollen verschiedene Einsatzmöglichkeiten für die Kamera geschaffen werden, wie Zeitraffer-Aufnahmen, Überwachungskamera oder stationärer Fotoapparat mit Fernsteuerung über Sprachbefehl und Webinterface.

Die Funksteckdosen, die Kamera und eventuell weitere Geräte sollen neben Sprachbefehlen auch über eine webbasierte Hauszentralsteuerung erreichbar sein. Diese Hauszentralsteuerung muss von beliebigen und webbrowsersfähigen Geräten, mit Anbindung ans Heimnetz, ansteuerbar sein. Die webbasierte Hauszentralsteuerung soll zudem über ein passendes Untermenü die zur Verfügung stehenden Funktionen aufzeigen und eine Übersicht über mögliche Sprachbefehle bieten. Über die webbasierte Hauszentralsteuerung sollen die selben Funktionen vorliegen, welche auch über Sprachbefehle zur Verfügung stehen.

Zur einfacheren Ansteuerung, erleichterter Bedienbarkeit und Komfort sollen personalisierte Befehle und Befehlsprotokolle erstellt werden können. Idealerweise im Webinterface der Hauszentralsteuerung. Zu diesen Befehlen und Protokollen zählen Befehle zur parallelen Ansteuerung von verschiedenen Geräten, Zeitschaltuhren sowie Protokolle für Energiesparmaßnahmen und Sicherheit.

(- Ansteuern von Geräten mittels Infrarotsender)

4.1.3. Nicht-Funktionale Anforderungen

Im Folgenden werden die qualitativen Anforderungen bzw. die Qualitätsattribute des Projektes genauer definiert.

Systemumgebung

Der Zugang zum System erfolgt über ein internes lokales Heimnetzwerk, über Webbrowser und Sprachbefehle. Funktionsrealisierung erfolgt mittels Node.js Server auf Raspberry Pi, Alexa Voice Service unterstützendem Gerät, wie zum Beispiel Amazon Echo Dot, und mit an Raspberry Pi verbundenen Geräten.

Das Aufrufen der webbasierten Heimzentralsteuerung ist betriebssystemunabhängig und kann von beliebigen Geräten mit Javascript fähigem Webbrowser erfolgen, wie beispielsweise Smartphone, Tablet oder Heimcomputer.

Performance

Das System soll direkt auf Sprachkommandos und User-Inputs mittels Webinterface reagieren. Eine maximale Verzögerung von zwei Sekunden, bis der Benutzer ein auditives oder visuelles Feedback bekommt, wird angestrebt.

Bedienbarkeit

Das System soll einfach und intuitiv bedienbar sein und daher möglichst genau den Anforderungen (Siehe 3.8 Gebrauchstauglichkeit, ISO 9241-110:2006) gerecht werden.

Das Modul soll den Benutzer unterstützen seine Arbeit zu machen und darf dabei keine unnötigen Informationen anzeigen oder dem Benutzer überflüssige Arbeitsschritte auferlegen. Es soll dem Benutzer in jedem Moment klar sein, was er tun kann und was er zu tun hat, um sein Ziel zu erreichen. Informationen wie Rückmeldungen und

Beschriftungen von Bedienelementen sowie Funktionen sollen auf den Benutzer natürlich wirken und ihm allgemein bekannt vorkommen. Hinweise sollen dem Benutzer helfen die Bedienoberfläche der webbasierten Heimzentralsteuerung und der Sprachsteuerung kennenzulernen und ihre Funktionen zu verstehen. Bei falscher Bedienung sollen Fehler vom System erkannt werden, dem Benutzer klar gemacht werden, was schiefgelaufen ist und ihm somit helfen es beim nächsten Versuch besser zu machen. Der Benutzer muss immer die Kontrolle über das System haben um sein Smart Home zu bedienen und ungewünschtes Verhalten abwenden können. Zudem soll der Benutzer sein Smart Home individualisieren können, daher muss die Möglichkeit zur individuellen Benennung von Befehlen und somit auch Geräten geschaffen werden.

Zuverlässigkeit

Das System, inklusive Sprach- und Webservice, muss allzeit verfügbar sein und nach Unterbrechung, wie zum Beispiel durch einen Stromausfall schnell und möglichst autonom anlaufen. Für Aktionen wie Sprachbefehle oder weitere, durch den Benutzer durchgeführte, Aktionen besteht eine angestrebte Fehlertoleranz von unter zehn Prozent, damit das System allzeit zuverlässig seine Arbeit verrichtet.

4.2. Herangehensweise

- Planung: Anforderungsanalyse aufstellen und Design / Technischer Entwurf festlegen
- mit Raspberry Pi Geräte an steuern (zunächst Funksteckdosen)
- Raspberry Pi mit Alexa verbinden
- eigene Skills implementieren und Testen–¿ Proof of Concept abgeschlossen
- Einrichtung durch zweite Installation gegenprüfen
- Repertoire auf Kamera erweitern

4.3. Architektur

Das System ist unterteilt in einer Art von Client-Server-Architektur. Der Client wird hierbei repräsentiert von dem Alexa Voice Service und der Heimzentralsteuerung im Webbrowser, da diese in erster Linie dazu dienen, Kommandos vom Benutzer entgegen zu nehmen und zudem Informationen über das System bereit zu stellen.

Die Server-Seite wird vertreten vom Raspberry Pi und dem darauf laufenden Node.js Laufzeitumgebung mit Express.js Webserver. Über den Raspberry Pi werden zudem die

Smart Home Geräte angesteuert und deren Logik hinterlegt.

Die Verbindung von Server und Client erfolgt über HTTP- und TCP-Protokoll.

4.4. Design

Auf Abbildung 4.1 kann man die Trennung und den Zusammenhang zwischen Raspberry Pi (Server) und Amazon Echo Dot (Client) [ECHO] erkennen.



Abb. 4.1.: Raspberry Pi-Alexa-Kommunikation

Auf dem Raspberry Pi [RASP_a] befindet sich die Node.js [NODE_a] Laufzeitumgebung mit Express.js Webserver. Der Express.js Webserver kommuniziert über HTTP mit dem Alexa Voice Service über ein neues Alexa Skill [ALEX_{Ab}], welches speziell für die Anbindung von Raspberry Pi geschaffen wird.

4.4.1. Skill-Typen

Custom Skills

Smart Home Skills

5. Implementierung

In diesem Kapitel werden die einzelnen Entwicklungsschritte der in der Architektur und Design vorgestellten Elemente genauer, aufgeschlüsselt, dargestellt.

5.1. Einrichtung

Im Folgendem wird die Einrichtung der gewählten Systeme, Plattformen und Server erläutert.

5.1.1. Raspberry Pi

Zunächst muss der Raspberry Pi eingerichtet werden. [W3SCHa] Die Entscheidung für die Geräteauswahl fiel auf den Raspberry Pi 3 Model B. Das Projekt ist aber auch auf alle anderen Raspberry Pi Modellen portierbar.

Für eine einfache Bedienung des Raspberry Pi kommt in diesem Projekt das „Raspbian Stretch with Desktop“ [RASPb] zum Einsatz. Solange die Ansteuerung der GPIO Ports gewährleistet ist, kann auch ein beliebiges anderes Betriebssystem auf dem Raspberry Pi eingerichtet werden.

Um ein Betriebssystem für den Raspberry Pi zu installieren, muss das Betriebssystem auf der MicroSD-Karte des Raspberry Pi installiert („geflasht“) werden. Am einfachsten geht dies mit dem kostenlosen Programm Etcher. [ETC17] Etcher ist für Windows, Linux und MacOS erhältlich. In Etcher muss lediglich das Medium (die MicroSD-Karte) sowie ein Image zur Installation ausgewählt und die Installation gestartet werden. Sobald Etcher das Betriebssystem auf dem Raspberry Pi installiert hat, kann die MicroSD-Card in den Raspberry Pi gesteckt werden und von dort direkt gebootet werden.

Der Raspberry Pi kann entweder über direkt angeschlossene Peripheriegeräte bedient werden oder per Remote. Für die Bedienung über eine SSH empfiehlt sich Putty. [PuTTY] In unserem Projekt wurde außerdem RealVNC [RVNC] verwendet, welches auf dem Raspbian Stretch bereits vorinstalliert ist. RealVNC hat gegenüber SSH den Vorteil, dass

der gesamte Desktop über Remote bedient werden kann.

Hier folgt demnächst ein Bild vom Raspberry Pi, welcher für dieses Projekt benutzt wurde.

5.1.2. Python

[folgt]

5.1.3. Node.js

Für das Projekt wird Node.js in Version 9.x verwendet. Um das aktuellste Node.js v9.x zu installieren führt man folgende Befehle aus: [NODEb]

```
curl -sL https://deb.nodesource.com/setup\_9.x | sudo -E bash -  
sudo apt-get install -y nodejs
```

Quellcode 5.1: Node.js Installieren

5.1.4. Cloud 9

Für die Entwicklung auf dem Raspberry Pi wurde die Web-IDE Cloud 9 [C9.io] verwendet, welche auf dem Raspberry Pi als Server installiert ist. Zur Einrichtung von Cloud 9 wurde die online verfügbare Anleitung von Joshua Lunsford verwendet. [LUN17] Die Web-IDE ermöglicht besonders einfaches Entwickeln und Testen über jeden Webbrowser auf Computern im selben Netzwerk wie der Raspberry Pi.

5.1.5. Git

Zur Sicherung und Versionierung wurde ein GitHub Repository eingerichtet.

```
git clone https://github.com/RobinWarth/Smart-Home-Solutions.git  
git remote add origin https://github.com/RobinWarth/Smart-Home-  
  ↪ Solutions.git
```

Quellcode 5.2: GitHub Repository

5.1.6. Amazon Developer Konto

Für das Erstellen von Alexa Skills muss ein Amazon Developer Konto erstellt werden. [AMA18a]

5.1.7. Alexa

Zunächst einmal müssen im ersten Schritt unter Skill Information grundlegende Informationen über den Skill festgelegt werden, welche die weiteren Schritte und deren Komplexität beeinflussen.

The screenshot shows the Amazon Developer Console interface for setting up a new skill. The top navigation bar includes links for Dashboard, Apps & Services, Alexa (selected), Reporting, Support, Docs, and Settings. The user is logged in as Robin Warth.

The main content area displays the 'Raspberry Dude' skill setup page. It includes a 'Back to All Skills' link and a 'Getting Started' link. The skill's ID is 'amzn1.ask.skill.920e5e6a-89da-4f0e-9728-69870ca80c88'.

On the left, there is a sidebar with a list of sections: Skill Information (checked), Interaction Model (checked), Configuration (checked), Test (checked), Publishing Information (unchecked), and Privacy & Compliance (checked). Below this is a 'Skills Beta Testing' section with a 'Status Not yet eligible' message.

The main form area contains the following fields:

- Skill Type:** A dropdown menu set to 'Smart Home'.
- Language:** A dropdown menu set to 'German'.
- Application Id:** A text field containing the skill's ID.
- Name:** A text field containing 'Raspberry Dude'.
- Payload Version:** Radio buttons for 'v3 (preferred)' (selected) and 'v2 (legacy - deprecated; please select v3)'.

At the bottom of the form are two buttons: 'Save' and 'Submit for certification'. A 'Next' button is located at the bottom right of the page.

Abb. 5.1.: Setup Skill-Information

Wie schon in [Konzept -> Skilltypen] beschrieben, fiel die Entscheidung auf einen Skill, welcher die Smart Home Skill API unterstützt. Außerdem müssen in diesem Schritt noch die unterstützten Sprachen des Skills sowie ein Skill Name festgelegt werden.

Die Payload Version bestimmt in welcher Form und Stil Nachrichten im JSON-Format zwischen Skill, Alexa Smart Home API und Server ausgetauscht werden sollen. Dies wird im weiteren Verlauf noch einmal erneut aufgegriffen, wenn der Aufbau und Stil der JSON-Nachrichten weiter erläutert wird. [Verweis auf das Kapitel] Auch wenn hier noch zusätzlich die Auswahl einer Payload Version aufgeführt ist, so lässt sich doch nur die Payload Version „v3“ auswählen. Payload Version „v2“ ist zum Zeitpunkt dieser Studienarbeit schon so veraltet, dass man sie nicht einmal auswählen kann. Sie ist nur für ältere schon erstellte Skills noch aufgeführt

The screenshot shows the 'Interaction Model' step in the Alexa Skill Setup process. On the left, a sidebar lists the setup steps: Skill Information, Interaction Model (highlighted in orange), Configuration, Test, Publishing Information, and Privacy & Compliance. The 'Interaction Model' step is marked with a green checkmark. Below the sidebar, there is a 'Skills Beta Testing' section with a 'NEW' tag and a status of 'Not yet eligible'. The main content area has a language selector set to 'German' and a link to 'Add a New Language'. A large blue box contains an information icon and the text: 'Your skill uses Smart Home Skill API. The interaction model is already built into the Smart Home Skill API and you can find the supported utterances [here](#). You just have to handle the directives sent by Alexa in your Smart Home adapter (Lambda function)'. Below this box are 'Save' and 'Submit for certification' buttons. At the bottom right is a yellow 'Next' button.

Abb. 5.2.: Setup Interaction Modell

Da in diesem Skill die Smart Home Skill API zum Einsatz kommt, muss in dem Schritt Interaction Model nichts weiter getan werden. Das Interaktionsmodell ist bereits in die API eingebaut.

Im Schritt Configuration wird zunächst der Service Endpunkt festgelegt, welcher der von Alexa gesendeten Direktiven geeignet verarbeiten werden kann. Da hier die Smart Home Skill API zum Einsatz kommt, wird man von Amazon dazu verpflichtet einen Amazon Web Services (AWS) Lambda Server zu nutzen. Zu dem Aufbau und der Einrichtung des AWS Lambda Server wird in Abschnitt [bla bla keine Ahnung, AWS Lambda] noch gezielt eingegangen.

German ☒ Add a New Language

Skill Information ☒
Interaction Model ☒
Configuration ☒
Test ☒
Publishing Information ☐
Privacy & Compliance ☐

Skills Beta Testing NEW
 Status Not yet eligible

Global Fields
 These fields apply to all languages supported by the skill.

Endpoint
 Service Endpoint Type: **AWS Lambda ARN (Amazon Resource Name)**
Recommended
 Lambda ARN url for Smart Home Adapter.
[More info about AWS Lambda](#)
[How to integrate AWS Lambda with Alexa](#)

Default
 arn:aws:lambda:eu-west-1:245784272632:function:myFunctionRaspberry

Provide geographical region endpoints?
 (Optional) ☒ Yes ☐ No

☐ North America ☒ Europe ☐ India ☐ Far East

Europe
 arn:aws:lambda:eu-west-1:245784272632:function:myFunctionRaspberry

Account Linking

Abb. 5.3.: Setup Configuration Endpoints

Wurde aber statt der Smart Home Skill API das Custom Interaction Model als Skill Typ gewählt, ist es möglich anstatt dem AWS Lambda Server auch jeden beliebigen anderen zertifizierten HTTPS Webserver anzugeben.

Bei Amazon Web Services wie dem Lambda Server (Siehe 5.1.8 Amazon Web Services (AWS) Lambda) wird jedem Server eine eindeutige Adresse zur Ressource und der darin enthaltenen Funktionen zugeteilt. Diese Adresse nennt sich Amazon Resource Name (ARN). Die ARN des Webservices dieser Studienarbeit lautet:

„arn:aws:lambda:eu-west-1:245784272632:function:myFunctionRaspberry“.

In der ARN sind Informationen über den Server Typ (hier: „Lambda“), über den Server Standort (hier: „eu-west“, Irland) und die Funktion (hier: „myFunctionRaspberry“) enthalten, welche den Endpunkt repräsentiert und die Direktiven von Alexa entgegennimmt.

Das Account Linking ist für Skills der Smart Home Skill API zwingend notwendig, aber für Skills mit Custom Interaction Model ist dies optional. Das Account Linking erfolgt mit Hilfe des OAuth2 Authentifizierungsstandard. [PAR18]

Account Linking

Authorization URL
The url where customers will be redirected in the companion app to enter login credentials.

Client Id
Unique public string used to identify the client requesting for authentication.

Domain List (Optional)
The list of domains that the authorization URL will fetch content from. You can provide up to 30 domains.

Scope
List of permissions to request from the skill user. You can provide up to 15 scopes.

Redirect URLs (Optional)
The list of valid HTTPS redirection endpoints that could be requested during authorization to redirect the user back to after the authorization process.
[Learn more](#).

Authorization Grant Type (Optional)
Specifies the OAuth authorization grant that Alexa uses to obtain an access token from your provider.
[Learn more](#).

Access Token URI
This URI will be used for both access token and token refresh requests.

Client Secret

Client Authentication Scheme (Optional)

Permissions

Request users to access resources and capabilities
Please request permissions to resources and capabilities that are absolutely core to the customer experience delivered by the skill.
[Learn More](#)

Alexa Skill Messaging
Use clientId and clientSecret for skill messaging, enabling out-of-session (non-customer invoked) events to be sent to skill code.

Privacy Policy URL
Link to the Privacy Policy for this skill. This is mandatory for account linking.

Buttons: Save, Submit for certification

Client ID: amzn1.application-oa2-client.8ce2860ce41b4f5d8602614ef7a0cbf8

Client Secret: *****

Domains:

- 1 pitangui.amazon.com ✖
- 2 layla.amazon.com ✖
- 3 alexa.amazon.co.jp ✖

Scopes:

- 1 profile:user_id ✖

Redirect URLs:

- https://pitangui.amazon.com/api/skill/link/MNR6JOVLWAVVK
- https://layla.amazon.com/api/skill/link/MNR6JOVLWAVVK
- https://alexa.amazon.co.jp/api/skill/link/MNR6JOVLWAVVK

Authorization Grant Type: ☐ Implicit Grant ☒ Auth Code Grant

Access Token URI: https://api.amazon.com/auth/o2/to

Client Authentication Scheme: HTTP Basic (Recommended) ▾

Permissions: ☒ Send Alexa Events ⓘ

Client ID: amzn1.application-oa2-client.bf5d5e600cb941528035489d8d0eabf2

Client Secret: ***** [Show](#)

Privacy Policy URL: https://github.com/RobinWarth/Smart-Home-Solutions

Next

Das Account Linking ist notwendig, um jeden Benutzer des Skills eindeutig identifizieren zu können. Im Allgemeinen wird hier eine Verknüpfung zwischen Alexa-Amazon-Account und dem Account beim Hersteller der Smart Home Geräte, wie z. B. Lampen, geschaffen.

In diesem Project erfolgt das Account Linking mit einem Amazon Account. Die Einstellungen hier sind alle auf Login With Amazon (LWA) getrimmt. [AMA18b] Der Aufwand, um einen guten, sicheren und verlässlichen OAuth2 Server selbst zu erstellen, wäre für diese Studienarbeit zu groß gewesen. Zudem muss der OAuth2 Server auch von Amazon akzeptiert sein und daher wären zusätzliche gültige Zertifikate notwendig. Die Möglichkeiten, welche durch Account Linking geschaffen werden, würden ohnehin nicht ausgereizt werden in dieser Studienarbeit und dem Einsatz Gebiet von dem hier erstellten Alexa Skill. [AMA18c]

...

Um den Skill mit der Alexa App auf dem Smartphone und einem Amazon Echo Gerät zu testen, muss nur der Schalter in Schritt Test umgelegt werden. Der Skill wird nun für die mit dem Konto verbundenen Geräte automatisch zum Testen eingerichtet. Alternativ kann der Skill auch mittels, dem zur Zeit der Studienarbeit in der Beta-Phase befindlichen, Test-Simulator ausgeführt werden. Der Test-Simulator bietet dieselbe Möglichkeit zum Testen wie ein Echo-Gerät und sogar zusätzliche Funktionen. Im Test-Simulator kann direkt mit Alexa über Sprache oder geschriebenem Text interagiert werden. Wurde zu Beginn das Custom Interaction Model statt der Smart Home Skill API gewählt, lassen sich im Test Simulator auch die JSON Direktiven anzeigen, welche von Alexa oder dem Skill-Endpoint gesendet werden.

German
Add a New Language

Skill Information
Interaction Model
Configuration
Test
Publishing Information
Privacy & Compliance

Try the testing simulator (beta), an intuitive interface for testing multi-turn dialogs and device renderings.
[Go to Test Simulator ^{BETA}](#)

On March 1, 2018 - Test Simulator (beta) will replace the test enablement function and all simulator content on this page.

Start testing this skill
☒ **Yes** Show this skill in the Alexa App.

You will be able to see your skill in the Skills tab in Alexa App and you can enable the skill and start testing.

After completing your testing please submit the skill for certification. If it passes Amazon's testing and certification process, it will become available to Alexa end users

The skill is available in "Skills > Your Skills" page of the Alexa App when you select 'Yes' above. You can then enable the skill and test its functionality on your device by asking Alexa, **Entdecken Sie meine Geräte**.

Submit for certification
Next

Abb. 5.5.: Setup Test

5.1.8. Amazon Web Services (AWS) Lambda

Um einen Alexa Skill mit der Smart Home Skill API zu erstellen, wird ein AWS Lambda Server vorausgesetzt. Hierfür muss ein Konto für AWS erstellt und ein AWS Lambda Server gemietet werden.

Bevor der Lambda Server eingerichtet wird, muss ein Alexa Skill bereits erstellt sein. Bei der Einrichtung des AWS Lambda Server für den Alexa Skill wird die Skill-ID benötigt. Ebenfalls ist ein AWS Account zwingend notwendig.

Nach der Anmeldung bei AWS wird zu Lambda navigiert. Für den Lambda Server ist es wichtig, die richtige Region auszuwählen. Mögliche Regionen für Alexa Skills sind Asia Pacific (Tokyo), EU (Ireland), US East (N. Virginia) und US West (Oregon). [AMA18d]

Als nächstes muss eine Lambda Function werden, um in dieser die von Alexa gesendeten Direktiven zu entgegen zu nehmen und zu verarbeiten.

Bei der Erstellung einer Lambda Function kann man aus verschiedenen Blaupausen oder eine leere Funktion, ohne Codebeispiel, wählen. Für den Einsatz der Lambda Function in dieser Studienarbeit hat zur Zeit der Bearbeitung leider keine passende Blueprint

existiert, diese waren veraltet und haben eine alte „Payload-Version“ der Alexa Direktiven unterstützt. Dank Codebeispielen in der Online Dokumentation von Amazon Alexa kann die Lambda Funktion dennoch zügig einsatzbereit gemacht werden. Der Name der hier erstellten Lambda Funktion lautet „myFunctionRaspberry“.

Danach muss eine Rolle gewählt werden, da zu diesem Zeitpunkt noch keine eigene definierte Rolle existiert. Für dieses Project wurde eine Rolle Namens „myRoleRaspberry“ mithilfe des Templates „Basic Edge Lambda permissions“ erstellt.

Durch das Betätigen des „Create Function“ Buttons wird die Funktion von AWS eingerichtet. Um auf Ereignisse zu reagieren, werden Trigger zur Funktion hinzugefügt. Standardmäßig wurde ein Trigger „Amazon CloudWatch Logs“ bereits der Funktion hinzugefügt. Dieser ist für das Logging zuständig. Jede Log Ausgabe wie z. B. „console.log('test');“ landet somit automatisch in den „Amazon CloudWatch Logs“ mit Datum und eindeutigen Identifikatoren. Um die Lambda Funktion für die Alexa Smart Home Skill API bereit zu machen, wird der Alexa Smart Home Trigger ausgewählt. Dieser Trigger muss nun nur noch mithilfe der Skill-ID bereit gemacht werden.

5.1.9. ngrok

[folgt...]

5.2. Entwicklung

Hier wird [folgt]

5.2.1. AWS Lambda

[folgt]

Export Handler

Mit `exports.handler` wird die Funktion definiert, welche ausgeführt wird, sobald eine Direktive an die Lambda Funktion gesendet wird. In dem Parameter „request“ stecken die Informationen, welche mit der Direktive gesendet werden. Der Parameter „context“ wird von AWS Lambda dazu verwendet Laufzeitinformationen zur ausgeführten Lambda-Funktion bereitzustellen.

Hier im export-handler werden die verschiedenen Typen an Direktiven zunächst einmal

anhand von dem „header“ unterschieden und entsprechend weitere Aktionen durchgeführt.

```
exports.handler = (request, context) => {
  if (request.directive.header.namespace === 'Alexa' && request.
    ↪ directive.header.name === 'ReportState') {
    console.log("DEBUG:", "ReportState_Request", JSON.stringify(
      ↪ request));
    handleReportState(request, context, "");
  }
  else if (request.directive.header.namespace === 'Alexa.Discovery'
    ↪ && request.directive.header.name === 'Discover') {
    console.log("DEBUG:", "Discover_Request", JSON.stringify(
      ↪ request));
    handleDiscovery(request, context, "");
  }
  else if (request.directive.header.namespace === 'Alexa.
    ↪ PowerController') {
    if (request.directive.header.name === 'TurnOn' || request.
      ↪ directive.header.name === 'TurnOff') {
      console.log("DEBUG:", "TurnOn_or_TurnOff_Request", JSON.
        ↪ stringify(request));
      handlePowerControl(request, context);
    }
  }
}
```

Quellcode 5.3: Request: Report-State

Direktiven

Die Direktiven welche von Alexa gesendet werden sind Nachrichten im JSON-Format. Die Direktiven beschreiben Aktionen, welche ausgeführt werden sollen, sowie Informationen zu dem Client welcher die Aktion ausführen möchte.

Beispiele für Direktiven sind:

Report State Über die ReportState-Direktive wird der Status der Smart Home Geräte abgefragt. Dies geschieht insbesondere dann, wenn der Benutzer über sein Smartphone in der Alexa App seine Geräte ansieht. In diesem Fall werden in kleinen zeitlichen

Abständen kontinuierlich ReportState-Direktiven gesendet.

```
{
  "directive": {
    "header": {
      "namespace": "Alexa",
      "name": "ReportState",
      "payloadVersion": "3",
      "messageId": "e9029fff-5d95-44bb-837b-8032d8c5f07d",
      "correlationToken": "AAAAAAAAAACwH+Bi...
        ↪ TAMUZSXeE2sPN6MqRNCVt0lQKGORY2vxQI="
    },
    "endpoint": {
      "scope": {
        "type": "BearerToken",
        "token": "Atza|IwEBIJqaj14hApPxtKKwPXA...1APp5F-
          ↪ a2cDP_EW7Fgj-A"
      },
      "endpointId": "wirelessSwitch1",
      "cookie": {
        "key1": "arbitrary key/value pairs for skill to
          ↪ reference this endpoint.",
        "key2": "There can be multiple entries",
        "key3": "but they should only be used for reference
          ↪ purposes.",
        "key4": "This is not a suitable place to maintain
          ↪ current endpoint state."
      }
    },
    "payload": {}
  }
}
```

Quellcode 5.4: Request: Report-State

```
{
  "context": {
    "properties": [
      {
        "namespace": "Alexa.PowerController",
```

```

        "name": "powerState",
        "value": "OFF",
        "timeOfSample": "2018-03-07T06:58:25.978Z",
        "uncertaintyInMilliseconds": 500
    }
]
},
"event": {
    "header": {
        "messageId": "e9029fff-5d95-44bb-837b-8032d8c5f07d-R",
        "correlationToken": "AAAAAAAAAQCwH+Bi...
        ↪ TAMUZSXeE2sPN6MqRNCVt0lQKGORY2vxQI=",
        "namespace": "Alexa",
        "name": "StateReport",
        "payloadVersion": "3"
    },
    "endpoint": {
        "scope": {
            "type": "BearerToken",
            "token": "Atza|IwEBIJqaj14hApPxtKKwPXA...1APp5F-
            ↪ a2cDP_EW7Fgj-A"
        },
        "endpointId": "wirelessSwitch1",
        "cookie": {}
    },
    "payload": {}
},
"payload": {}
}

```

Quellcode 5.5: Response: Report-State

Discovery Über die Discovery-Direktive fragt Alexa nach verfügbaren Smart Home Geräten. Diese Direktive wird gesendet, wenn der Benutzer aktiv in der Alexa App nach Geräten suchen lässt.

```

{
    "directive": {
        "header": {

```

```

        "namespace": "Alexa.Discovery",
        "name": "Discover",
        "payloadVersion": "3",
        "messageId": "af4efd28-3d33-4f5e-9d18-ae755a40c8c0"
    },
    "payload": {
        "scope": {
            "type": "BearerToken",
            "token": "Atza|IwEBIJqaj14hApPxtKKwPXA...1APp5F-
                    ↪ a2cDP_EW7Fgj-A"
        }
    }
}

```

Quellcode 5.6: Request: Discovery

Power Controller Über den PowerController wird versucht, aktiv einen Gerätestatus zu verändern, wenn z. B. wenn der Benutzer sein Licht einschaltet oder die Heizung aufdrehen möchte.

```

{
    "directive": {
        "header": {
            "namespace": "Alexa.PowerController",
            "name": "TurnOn",
            "payloadVersion": "3",
            "messageId": "64da8006-d724-4861-a929-d560cbf2b92b",
            "correlationToken": "AAAAAAAAAQCwH+
                                ↪ BiiholKtQip3as2gbcBAIAAAA...BdP/jF6aJ9bc="
        },
        "endpoint": {
            "scope": {
                "type": "BearerToken",
                "token": "Atza|IwEBIJqaj14hApPxtKKwPXA...1APp5F-
                        ↪ a2cDP_EW7Fgj-A"
            },
            "endpointId": "wirelessSwitch1",
            "cookie": {

```

```
        "key1": "arbitrary key/value pairs for skill to  
            ↪ reference this endpoint.",  
        "key2": "There can be multiple entries",  
        "key3": "but they should only be used for reference  
            ↪ purposes.",  
        "key4": "This is not a suitable place to maintain  
            ↪ current endpoint state."  
    },  
    "payload": {}  
}
```

Quellcode 5.7: Request: Power-Controller

```
{  
  "context": {  
    "properties": [  
      {  
        "namespace": "Alexa.PowerController",  
        "name": "powerState",  
        "value": "ON",  
        "timeOfSample": "2018-03-07T12:34:45.360Z",  
        "uncertaintyInMilliseconds": 500  
      }  
    ]  
  },  
  "event": {  
    "header": {  
      "messageId": "64da8006-d724-4861-a929-d560cbf2b92b-R",  
      "correlationToken": "AAAAAAAAAQCwH+  
        ↪ BiiholKtQip3as2gbcBAIAAAA...BdP/jF6aJ9bc=",  
      "namespace": "Alexa",  
      "name": "Response",  
      "payloadVersion": "3"  
    },  
    "endpoint": {  
      "scope": {  
        "type": "BearerToken",  

```



```
        "token": "Atza|IwEBIJqaj14hApPxtKKwPXA...1APp5F-  
        ↪ a2cDP_EW7Fgj-A"  
    },  
    "endpointId": "wirelessSwitch1",  
    "cookie": {}  
},  
    "payload": {}  
},  
    "payload": {}  
}
```

Quellcode 5.8: Response: Power-Controller

Hier ist deine Referenz auf den Anhang: A.1 Response: Discovery

6. Ergebnis, Fazit und Ausblick

[folgt]

Literatur

- [ALEXAa] AMAZON.COM. *Amazon Alexa Logo*. abgerufen am 16.03.2018 09:24 Uhr. 2018. URL: https://m.media-amazon.com/images/G/01/mobile-apps/dex/avs/docs/ux/branding/mark2._TTH_.png [siehe S. 6].
- [ALEXAb] AMAZON.COM. *Amazon Alexa Logo*. abgerufen am 01.12.2017 15:23 Uhr. 2017. URL: https://www.wink.com/img/vendor/logo_amazon.svg [siehe S. 26].
- [AMA18a] AMAZON.COM. *Amazon Developer*. abgerufen am 27.02.2018 12:23 Uhr. 2018. URL: <https://developer.amazon.com/de/> [siehe S. 28].
- [AMA18b] AMAZON.COM. *Login with Amazon for Websites*. abgerufen am 27.02.2018 12:33 Uhr. 2018. URL: <https://developer.amazon.com/de/docs/login-with-amazon/web-docs.html> [siehe S. 33].
- [AMA18c] AMAZON.COM. *Authenticate an Alexa User to a User in Your System with Account Linking*. abgerufen am 27.02.2018 12:33 Uhr. 2018. URL: <https://developer.amazon.com/de/docs/smarthome/authenticate-an-alexa-user-account-linking.html> [siehe S. 33].
- [AMA18d] AMAZON.COM. *Host a Custom Skill as an AWS Lambda Function*. abgerufen am 27.02.2018 12:40 Uhr. 2018. URL: <https://developer.amazon.com/de/docs/custom-skills/host-a-custom-skill-as-an-aws-lambda-function.html> [siehe S. 34].
- [BAG17] J. BAGER. „Assistent allgegenwärtig - Digitale Assistenten: vom Spielzeug für Nerds zur Bedienoberfläche für alles“. In: *c't 22/2017* [2017]. Online unter <https://www.heise.de/ct/ausgabe/2017-22-Digitale-Assistenten-vom-Spielzeug-fuer-Nerds-zur-Bedienoberflaeche-fuer-alles-3854034.html> - abgerufen am 06.02.201 12:14 Uhr, S. 64–69 [siehe S. 7–10, 12].

- [BIXBY] Samsung GROUP. *Samsung Bixby Logo*. abgerufen am 16.03.2018 09:41 Uhr. 2018. URL: <https://www.bixbysamsung.com/img/bixby-for-everyone.png> [siehe S. 6].
- [BLE17] Reche M. BLEICH H. „Freundlich, hölzern, clever und arrogant - Vier Sprachassistenten in ihrem natürlichen Umfeld“. In: *c't 22/2017* [2017]. Online unter <https://www.heise.de/ct/ausgabe/2017-22-Vier-Sprachassistenten-in-ihrem-natuerlichen-Umfeld-3853597.html> - abgerufen am 13.11.2017 10:58 Uhr, S. 80–85 [siehe S. 7–12].
- [C9.io] C9.IO. *C9.io Github Repository*. abgerufen am 27.02.2018 11:57 Uhr. 2018. URL: <https://github.com/c9> [siehe S. 28].
- [CLA18] G. CLAUSER. *Amazon Echo vs. Google Home: Which Voice Controlled Speaker Is Best for You?* abgerufen am 19.02.2018 11:12 Uhr. 2018. URL: <https://thewirecutter.com/reviews/amazon-echo-vs-google-home/> [siehe S. 8, 11].
- [DEV17] DEVOLO AG. *Definition Smart Home*. abgerufen am 17.11.2017 14:22 Uhr. 2017. URL: <https://www.devollo.de/smart-home/> [siehe S. 3].
- [ECHO] Amazon ECHO. *Amazon Echo Dot 2*. abgerufen am 01.12.2017 15:24 Uhr. 2017. URL: https://images-eu.ssl-images-amazon.com/images/I/41iz5Tw82IL._SY300_QL70_.jpg [siehe S. 26].
- [ETC17] RESIN.IO. *Etcher Download*. abgerufen am 27.10.2018 11:35 Uhr. 2018. URL: <https://etcher.io/> [siehe S. 27].
- [GOASS] Google LLC. *Google Assistant Logo*. abgerufen am 16.03.2018 09:32 Uhr. 2018. URL: <https://i1.wp.com/www.techdotmatrix.com/wp-content/uploads/2017/11/Google-Assistant-troubleshoot.png?fit=800%5C%2C424&ssl=1> [siehe S. 6].
- [HAR17] D. HARDAWAR. *One week with Samsung Bixby - It has potential, but can't yet compete with Apple, Google or Amazon*. abgerufen am 09.02.2018 13:17 Uhr. 2017. URL: <https://www.engadget.com/2017/08/23/samsung-bixby/> [siehe S. 14].
- [ISO06] *ISO 9241: Ergonomie der Mensch-System-Interaktion - Teil 110: Grundsätze der Dialoggestaltung*. Standard. 2006 [siehe S. 21].

- [LUN17] J. LUNSFORD. *Setup JavaScript IDE on Pi 3*. abgerufen am 27.02.2018 12:03 Uhr. 2017. URL: <https://dev.to/jtlunsford/setup-javascript-ide-on-pi-3> [siehe S. 28].
- [MICOR] Microsoft CORPORATION. *Microsoft Cortana Logo*. abgerufen am 16.03.2018 09:38 Uhr. 2018. URL: <https://www.nimblechapps.com/wp-content/uploads/2015/01/I-am-Cortana1.jpg> [siehe S. 6].
- [MOZ17] H. BRAUN. *Mozilla Common Voice: Sprachsteuerung für alle und ohne Rückgriff auf die Cloud*. abgerufen am 06.12.2017 14:01 Uhr. 2017. URL: <https://www.heise.de/newsticker/meldung/Mozilla-Common-Voice-Sprachsteuerung-fuer-alle-und-ohne-Rueckgriff-auf-die-Cloud-3904454.html> [siehe S. 4].
- [NODEa] NODE.JS Foundation. *Node.js Logo*. abgerufen am 01.12.2017 15:25 Uhr. 2017. URL: <https://nodejs.org/static/images/logos/nodejs-new-pantone-black.png> [siehe S. 26].
- [NODEb] NODE.JS Foundation. *Installing Node.js via package manager*. abgerufen am 27.02.2018 11:50 Uhr. 2018. URL: <https://nodejs.org/en/download/package-manager/> [siehe S. 28].
- [OAUTHa] IETF OAuth Working GROUP. *OAuth 2.0*. abgerufen am 27.02.2018 12:27 Uhr. 2018. URL: <https://oauth.net/2/> [siehe S. 15].
- [OAUTHb] Aaron PARECKI. *OAuth 2.0 Simplified*. abgerufen am 16.03.2018 15:21 Uhr. 2018. URL: <https://aaronparecki.com/oauth-2-simplified/> [siehe S. 15].
- [PFI17] Kaufmann T. PFISTER B. *Sprachverarbeitung. Grundlagen und Methoden der Sprachsynthese und Spracherkennung*. 2. Aufl. Zürich, Schweiz: Springer Vieweg, 2017. ISBN: 978-3-662-52838-9 [siehe S. 4].
- [PuTTY] S. TATHAM. *Download PuTTY: latest release (0.70)*. abgerufen am 27.02.2018 11:40 Uhr. 2017. URL: <https://www.chiark.greenend.org.uk/~sgtatham/putty/latest.html> [siehe S. 27].
- [RAO17] Manjunath K.E. RAO S. K. *Speech Recognition Using Articulatory and Excitation Source Feature. Speech Technology*. Indien: Springer, 2017. ISBN: 978-1-4842-2922-4 [siehe S. 4].

- [RASP_a] Raspberry Pi FOUNDATION. *Raspberry Pi Logo*. abgerufen am 01.12.2017 15:26 Uhr. 2017. URL: <https://www.raspberrypi.org/app/uploads/2011/10/Raspi-PGB001.png> [siehe S. 26].
- [RASP_b] Raspberry Pi FOUNDATION. *Download Raspbian for Raspberry Pi*. abgerufen am 26.02.2018 11:29 Uhr. 2018. URL: <https://www.raspberrypi.org/downloads/raspbian/> [siehe S. 27].
- [REI17] A. REINHARDT. *Samsung Bixby Voice im Alltags-Check*. abgerufen am 09.02.2018 13:18 Uhr. 2017. URL: <https://www.teltarif.de/samsung-bixby-sprachassistent-erfahrungsbericht/news/70083.html> [siehe S. 13, 14].
- [RF6749] Internet Engineering Task Force (IETF). *RFC 6749*. abgerufen am 15.03.2018 12:00 Uhr. 2012. URL: <https://tools.ietf.org/html/rfc6749> [siehe S. 15, 16, 18–20].
- [RF6750] Internet Engineering Task Force (IETF). *RFC 6750*. abgerufen am 19.03.2018 13:32 Uhr. 2012. URL: <https://tools.ietf.org/html/rfc6750> [siehe S. 20].
- [ROU17_a] M. ROUSE. *Amazon Developer Services*. abgerufen am 09.01.2018 09:57 Uhr. 2017. URL: <http://searchaws.techtarget.com/definition/Amazon-Developer-Services> [siehe S. 20].
- [ROU17_b] M. ROUSE. *Alexa Skills Kit*. abgerufen am 09.01.2018 10:31 Uhr. 2017. URL: <http://searchaws.techtarget.com/definition/Alexa-Skills-Kit> [siehe S. 20].
- [RVNC] RealVNC LIMITED. *Remotezugriffssoftware für Desktop-Computer und Mobilgeräte — RealVNC*. abgerufen am 27.02.2018 11:46 Uhr. 2018. URL: <https://www.realvnc.com/de/> [siehe S. 27].
- [SCH17] K. SCHILLER. *Was ist ein Smart Home? Geräte und Systeme*. abgerufen am 10.11.2017 12:20 Uhr. 2017. URL: <https://www.homeandsmart.de/was-ist-ein-smart-home> [siehe S. 3].
- [SIRI] IOSWELT. *Apple Siri Logo*. abgerufen am 16.03.2018 09:29 Uhr. 2018. URL: <http://ioswelt.de/wp-content/uploads/2017/10/59e781abe09a00.21036803.jpeg> [siehe S. 6].

- [SKDOC] Applce INC. *SiriKit — Apple Developer Documentation*. abgerufen am 19.03.2018 13:31 Uhr. 2018. URL: <https://developer.apple.com/documentation/sirikit> [siehe S. 9].
- [VOE16] F. VÖLKELE. *Die Historie des Smart Home von 1963 - Heute*. abgerufen am 02.02.2018 10:53 Uhr. 2016. URL: https://www.smartest-home.com/blog/smart_home_historie_1939_bis_2017_frank_voelkel/ [siehe S. 1].
- [W3SCHa] O.V. *Node.js and Raspberry Pi*. abgerufen am 14.11.2017 10:00 Uhr. o.J. a. URL: https://www.w3schools.com/nodejs/nodejs_raspberrypi.asp [siehe S. 27].
- [WIG17] K. WIGGERS. *How to use the brand-new Samsung Bixby voice assistant, and everything it can do - Bixby 2.0 brings improved speech recognition and third-party app support*. abgerufen am 15.02.2017 15:23 Uhr. 2017. URL: <https://www.digitaltrends.com/mobile/samsung-bixby-how-to-use/> [siehe S. 13].

Anhangsverzeichnis

A Quellcode	XV
A.1 Response: Discovery	XV

A. Quellcode

A.1. Response: Discovery

```
{
  "header": {
    "namespace": "Alexa.Discovery",
    "name": "Discover.Response",
    "payloadVersion": "3",
    "messageId": "af4efd28-3d33-4f5e-9d18-ae755a40c8c0"
  },
  "payload": {
    "endpoints": [
      {
        "endpointId": "wirelessSwitch1",
        "manufacturerName": "Raspberry Dude",
        "friendlyName": "Licht",
        "description": "Smart Device Switch",
        "displayCategories": [
          "LIGHT"
        ],
        "cookie": {
          "key1": "arbitrary key/value pairs for skill to
            ↪ reference this endpoint.",
          "key2": "There can be multiple entries",
          "key3": "but they should only be used for reference
            ↪ purposes.",
          "key4": "This is not a suitable place to maintain
            ↪ current endpoint state."
        },
        "capabilities": [
          {
            "type": "AlexaInterface",
```

```
        "interface": "Alexa",
        "version": "3"
    },
    {
        "interface": "Alexa.PowerController",
        "version": "3",
        "type": "AlexaInterface",
        "properties": {
            "supported": [
                {
                    "name": "powerState"
                }
            ],
            "retrievable": true
        }
    }
]
},
{
    "endpointId": "wirelessSwitch2",
    "manufacturerName": "Raspberry Dude",
    "friendlyName": "PC-Switch",
    "description": "Smart Device Switch",
    "displayCategories": [
        "SWITCH"
    ],
    "cookie": {},
    "capabilities": [
        {
            "type": "AlexaInterface",
            "interface": "Alexa",
            "version": "3"
        },
        {
            "interface": "Alexa.PowerController",
            "version": "3",
            "type": "AlexaInterface",
            "properties": {
```

```
        "supported": [  
            {  
                "name": "powerState"  
            }  
        ],  
        "retrievable": true  
    }  
}  
],  
{  
    "endpointId": "wirelessSwitch3",  
    "manufacturerName": "Raspberry Dude",  
    "friendlyName": "TV-Switch",  
    "description": "Smart Device Switch",  
    "displayCategories": [  
        "SWITCH"  
    ],  
    "cookie": {},  
    "capabilities": [  
        {  
            "type": "AlexaInterface",  
            "interface": "Alexa",  
            "version": "3"  
        },  
        {  
            "interface": "Alexa.PowerController",  
            "version": "3",  
            "type": "AlexaInterface",  
            "properties": {  
                "supported": [  
                    {  
                        "name": "powerState"  
                    }  
                ],  
                "retrievable": true  
            }  
        }  
    ]  
}
```

```
}  
  }  
  ]  
    }  
      ]
```