phwt
Private Hochschule
für Wirtschaft und Technik

Ausarbeitung Rechnernetze

Amazon Web Services + Internet of Things

Private Hochschule für Wirtschaft und Technik Vechta/Diepholz
vorgelegt von: Claas Meints, Magnus Müller

Inhaltsverzeichnis

1 Einleitung

Der Begriff des Cloud Computings genießt aktuell einen hohen Stellenwert in der Informationstechnologie und verändert die IT Landschaft maßgeblich. Seit mehr als 10 Jahren schreitet das Anbieten von IT-Dienstleistungen in der Wolke kontinuierlich voran und die absolute Mehrheit an IT-Unternehmen weltweit nutzt Cloud Computing Angebote oder denkt über diese nach[Jan20]. Insbesondere für kleine und mittelständische Unternehmen bietet das Auslagern von Rechenleistung und weiteren Diensten interessante Möglichkeiten. Denn die Anschaffung und Wartung eines eigenen Rechenzentrums stellt kleinere Unternehmen vor eine Herausforderung[Til10]. Doch nicht nur für Unternehmen bietet die Cloud neue Möglichkeiten, sondern auch Privatleute profitieren von den Vorteilen der Cloud. Die Meisten Privatpersonen verbinden Cloud bisher mit der Sicherung von ihren Daten auf einem Server.

Neben der Cloud existiert aktuell ein weiterer Trend der ebenfalls eine hohe Aufmerksamkeit genießt, nämlich das Internet of Things (zu deutsch: Internet der Dinge). Dieser Begriff beschreibt die fortschreitene Vernetzung von Produkten, Services, Maschinen und Sensoren mittels IP-basierter Protokolle. Ähnlich wie die Cloud spielt das IoT eine wichtige Rolle für die Industrie (smart factories, predictive maintenance) aber auch private Haushalte (smart Homes)[Bra19][Bra22].

Beide Themen sind für sich schon Trends die aktuell viel Aufmerksamkeit genießen. Umso interessanter ist die Kombination der beiden Themen. In einem IoT Netzwerk können riesige Mengen an Daten anfallen, welche nicht immer Zentral auf einem IoT Gerät verarbeitet werden können. Daher bietet sich das Verarbeiten der Daten in einer Cloud an. In der Cloud werden nun auf Grundlage der gesammelten Daten Berechnungen durchgeführt und beispielsweise mithilfe Künstlicher Intelligenz Aktionen bestimmt, die anschließend ausgeführt werden.

Thema dieser Ausarbeitung ist eben genau dieses Zusammenspiel der beiden Themenfelder. Hierzu wird zuerst auf die Grundlagen des Cloud Computings eingegangen und

anschließend das Themenfeld IoT beleuchtet. Nach den Grundlagen wird kurz auf Amazon Web Services eingegangen und im Anschluss ein Beispielprojekt vorgestellt, in dem IoT Geräte mit AWS verknüpft werden.

2 Was ist Cloud Computing

2.1 Geschichte und Urpsrung

Bei der Cloud handelt es sich nicht um eine neue Technologie sondern eher um eine Wiederverwendung mehrerer bekannter Technologien. Um also das Konzept hinter Cloud Computing zu verstehen wird kurz auf die einzelnen Vorgänger eingegangen.

Dir Grundidee reicht bereits in die 1960er Jahre zurück. Bereits damals bestand der Grundgedanke, größere Probleme auf mehrere Rechner zu verteilen und somit effizienter zu berechnen. Das hierbei entstehende System nennt sich verteiltes System und die einzelnen Rechner werden Knoten genannt. Dies wurde damals in Form von "Cluster Computing"durchgeführt. Hierbei werden identische Rechner durch ein hochgeschwindigkeits Netzwerk miteinander Verbunden, was zu einer völlig neuen Rechenleistung und Verfügbarkeit führte. Diese Lösung stellt sich als deutlich günstiger als ein Großrechner heraus und wird noch heute in Supercomputern angewandt [Til10].

Wenn das Netzwerk, anders als beim Grid-Computing, aus Knoten mit verschiedenen Aspekten aufgabeut ist, wird vom Grid-Computing gesprochen. Die verschiedene Knoten können sich hierbei in Hardware, Software oder ihrer Anbindung an das Netzwerk unterscheiden. Zudem könne sich die verschiedenen Knoten vom Netzwerk trennen. Dieses Konzept wird häufig dazu genutzt, um unbenutzte CPUs verschiedener Rechner auszunutzen und somit Große Probleme in kleinen Teilproblemen auf verschiedenen Rechnern zu bearbeiten [Til10].

Ein weiteres Konzept stellt das Utility-Computing dar. Der Grundgedanke hierbei ist es, dass der Nutzer nur seine genutzte Rechenleistung bezahlt. Oft wird hier der Vergleich mit der Steckdose gezogen, da hier der Kunde ebenfalls nur für seinen bezogenen Strom bezahlt[Rep13]. Etwaige Überlegungen sind jedoch an der nicht verfügbaren Hardware und dem mangelhaften Netzausbau gescheitert. Schließlich wurde das Konzept in den

1990er Jahren in das Konzept der Cloud integriert und ist fester Bestandteil des Konzepts [Til10].

2.2 Definition

Eine einheitliche Definition für Cloud Computing existiert in der Literatur nicht. Vorallem weil der Begriff relativ unscharf ist und sich die Technologie noch im Wandel befindet. Nach besteht Cloud Computing aus einer Ansammlung von Diensten, Anwendungen oder IT-Ressourcen, die dem Nutzer flexibel und skalierbar über das Internet angeboten werden. AWS als größter Cloud Anbieter definiert Cloud Computing wie folgt: "Cloud Computingïst die bedarfsabhängige Bereitstellung von IT-Ressourcen über das Internet zu nutzungsabhängigen Preisen. Statt physische Rechenzentren und Server zu erwerben, zu besitzen und zu unterhalten, können Sie über einen Cloud-Anbieter wie Amazon Web Services (AWS) nach Bedarf auf Technologieservices wie beispielsweise Rechenleistung, Speicher und Datenbanken zugreifen."[Toc22a]

2.3 Merkmale

Da es jedoch keine einheitliche Definition gibt lässt sich Cloud Computing lediglich auf Grundlage der Merkmale Charakterisieren. Die Merkmale des National Institute of Technologys and Standards (NIST) gelten in der Praxis als auch Forschung als anerkannt und sind in Abbildung 2.1 zu sehen.

Ressource Pooling

Der Begriff Ressource Pooling beschreibt das Ansammeln von verschiedensten IT-Ressourcen zu einem Pool auf den die Benutzer zugreifen. Der Benutzer greift aus Seiner Sicht auf Ressourcen wie z.B. eine virtuell Maschine, Speicherkapazität oder eine Dienstinstanz zu. Mithilfe von Virtualisierung lassen sich physische Ressourcen für den Benutzer auf virtuelle Ressourcen abbilden und gleichzeitig besteht eine Trennung. Der Benutzer sieht also nicht was unter seiner virtuellen Maschine physisch liegt. Durch die Virtualisierung lassen sich Rechenzentren erheblich effizienter betreiben. Die Auslastung lässt sich hierbei statt der herkömmlichen 8 - 15 Prozent auf 70 - 80 Prozent steigern. Dies ist Interessant, da Server während sie untätig sind fast dieselbe Leistung benötigen, als wären sie ausgelastet[Inc11]. Der Benutzer hat somit keinen Einfluss auf die physischen Ressourcen mit

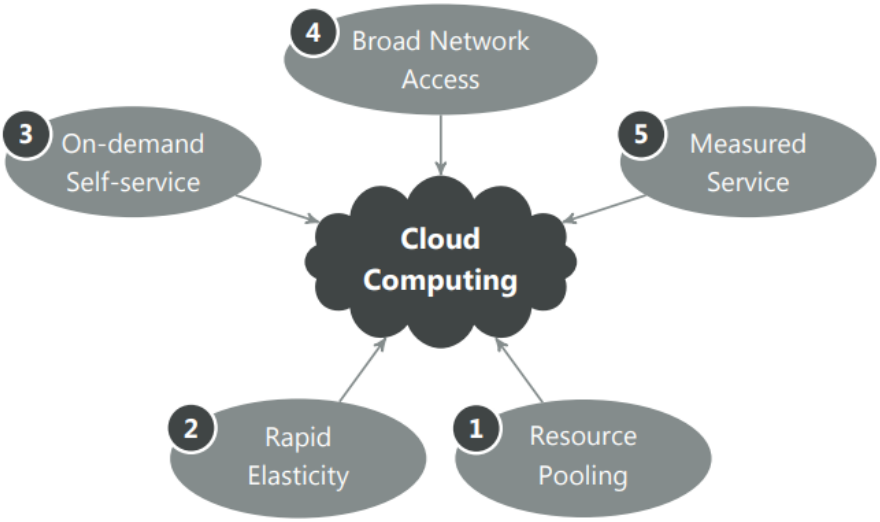

Abbildung 2.1
Charakteristika des Cloud Computings [Til10]

denen seine Dienste ausgeführt werden. Viele Cloud Anbieter bieten jedoch die Option an den Standort der physischen Ressourcen einzugrenzen z.B. aufgrund von Datenschutzrichtlinien. [Til10]

Rapid Elasticity

Der Cloud ist es möglich auf Änderungen der Rechenleistung schnell und dynamisch zu reagieren. Die meisten Anbieter können dies auch ohne Vorlaufzeit, sodass der Benutzer das Gefühl suggeriert bekommt, aus unendlichen Ressourcen zu schöpfen. Die Anpassung erfolgt hierbei händisch durch den Benutzer oder automatisch durch den Anbieter. Wenn also z.B. ein Geschäft zu Weihnachten eine höhere Anzahl an Anfragen auf ihrer Website erwartet, passt sich die Cloud automatisch an die steigenden Anfragen an. [Til10]

On-demand Self-service Eine weitere Charakteristik der Cloud ist die Möglichkeit sich nach Bedarf selbst bedienen zu können. Der Cloudanbieter muss somit dem Benutzer auf Anfrage die angeforderte Rechenleistung bereitstellen, ohne dabei viel Konfiguration vornehmen zu müssen um die Elastizität zu gewährleisten. Der wirtschaftliche Betrieb ist hierbei nur durch automatisierte Verwaltung und Optimierung gewährleistet und bildet ein wichtiges Merkmal eines Cloud Anbieters. [Til10]

Broad Network Access

In den meisten Fällen erfolgt der Zugriff auf die Ressourcen der Cloud über ein Netzwerk. Um einen Zugriff von diversen Geräten zu ermöglichen, werden standardisierte REST Schnittstellen oder Web-APIs und Protokolle bzw. Datenformate wie z.B. HTTP, XML JSON usw. verwendet. Voraussetzung hierfür ist jedoch eine ausreichende Bandbreite der Verbindung. [Til10]

Measured Service

Nach dem Vorbild des Utility Computings wird bei Cloud Anbietern nach den tatsächlich genutzten Ressourcen bezahlt. Um dies zu gewährleisten muss der Anbieter die vom Kunden genutzten Ressourcen mithilfe eines Messverfahrens und einer Messgröße messen. Beispielsweise Datenspeicher werden anhand des genutzten Speicherplatz abgerechnet und Rechenleistung nach den durchgeführten CPU Zyklen. Zusammen mit der hohen Elastizität (Rapid Elasticity) lässt sich der Kunde also präzise seine genutzten Abrechnen was als eines der Hauptmerkmale der Cloud gilt. [Vaq+09]

2.4 Bereitstellungsmodelle

Clouds lassen sich aufgrund ihrer Öfnnung nach außen in vier verschiedene Bereitstellungsmodelle einteilen. Die verschiedenen Modelle sind in **??** veranschaulicht.

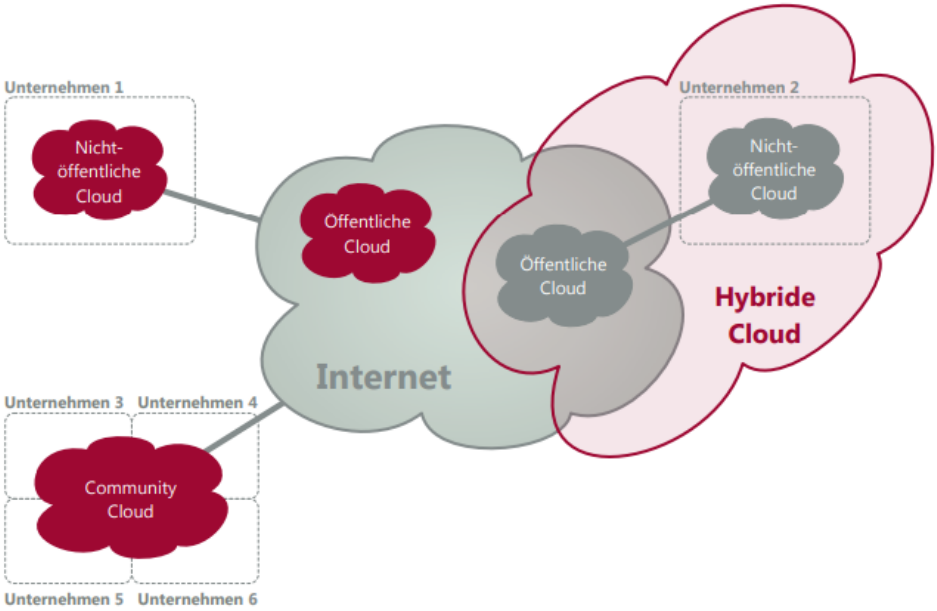

Abbildung 2.2
Die verschiedenen Bereitstellungsmodelle [Til10]

Public Cloud

Die Public Cloud ist eine öffentliche Cloud, die ihre Dienste der Öffentlichkeit anbietet und gegen Bezahlung genutzt werden kann. Sämtliche Infrastruktur und die angebotenen Dienste sind hierbei im Besitz des Anbieters. Ein Beispiel für diese Cloud ist AWS mit seinen Diensten[Wir22].

Private Cloud

Die Private Cloud beschränkt sich meist auf eine Organisation und ist nicht öffentlich zugänglich. Dies lässt eine erweiterte Einstellung der Datensicherheit hinsichtlich der Organisationsspezifischen Anforderungen zu[Wir22]. Der Standort der Server kann sich hierbei auf dem Grundstück der Firma aber auch woanders befinden. Die Verwaltung kann ebenfalls durch die Organisation oder einen Anbieter erfolgen [Til10].

Community Cloud

Die Community Cloud ist eine private Cloud, die jedoch von mehreren Organisationen mit den gleichen Anforderungen geteilt wird. Hier kann die Verwaltung und der Standort ebenfalls jeweils intern oder extern sein [Til10].

Hybrid Cloud

Die Hybride Cloud ist eine Mischung verschiedener Bereitstellungsmodelle. Ein gängiges Beispiel hierfür ist eine Private Cloud, welche als Datenspeicher für Datenbanken genutzt wird und durch eine öffentliche Cloud ergänzt wird. Die öffentliche Cloud wird dazu genutzt Daten zu verarbeiten und kann unerwartete Lastspitzen abfangen [Til10].

2.5 Service-Ebenen

It Ressourcen werden beim Cloud Computing als sog. Dienste angeboten [Zar12]. Um die verschiedenen Dienste der Cloud zu differenzieren haben sich drei Ebenen im Cloud Computing durchgesetzt, Infrastructure as a Service (IaaS), Platform as a Service (PaaS)und Software as a Service (SaaS). Anhand **??** lässt sich erkennen, dass die verschiedenen Ebenen ein unterschiedlichen Aufwand von Seiten des Benutzer erfordern. Bei IaaS liegt relativ viel Verantwortung beim Kunden, wohingegen bei SaaS keinerlei Verantwortung beim Kunden liegt. Die unzähligen Services lassen sich in diese Ebenen einordnen oder gehören weiteren eigenen Ebenen an wie z.B. Database as a Service oder Function as a Service.

Durch diese Bandbreite an Diensten ist ebenso die Vielfalt an Ebenen gewachsen und diese werden unter dem Begriff XaaS gesammelt. Im folgenden wird jedoch nur auf die vom NIST definierten Ebenen eingegangen.

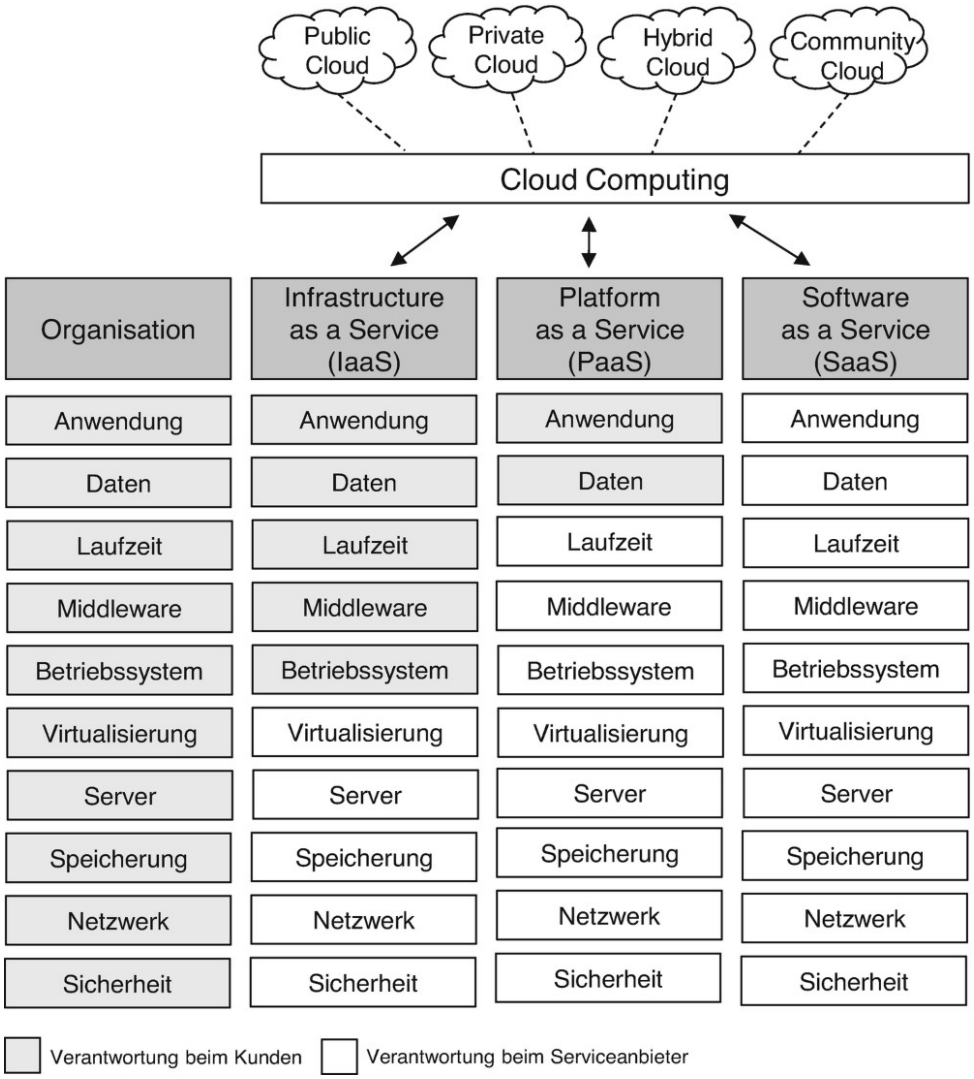

Abbildung 2.3
[Wir22]

Infrastructure as a Service

Wie es der Name ahnen lässt, wird dem Kunden hier Infrastruktur oder virtuelle Hardware in der Cloud bereitgestellt. Beispiele hierfür sind das anmieten von virtuellen Maschinen oder Speicherplatz für einen Backup.[Til10] Dabei handelt es sich um die unterste Ebene, da der Kunde die meiste Verantwortung im Vergleich zu den anderen ebenen besitzt. Bei einer virtuellen Maschinen muss der Kunde sich beispielsweise um alles oberhalb des

Platform as a Service

Hier erhält der Benutzer die Möglichkeit selbst entwickelte oder auch gekaufte Software in einer Laufzeitumgebung auszuführen [P M11]. Alles unterhalb der Laufzeitumgebung wird vom Provider bereitgestellt und festgelegt. In diesem Rahmen kann der Benutzer sich frei bewegen [Til10].

Software as a Service

In dieser Ebene stellt der Anbieter dem Benutzer eine Software bereit, die dieser Nutzen kann. Die gesamte Infrastruktur, Lizenzierung und Aktualisierung liegt in der Verantwortung des Anbieters, wodurch der Benutzer von all dem nichts mitbekommt. Der Benutzer greift somit lediglich über seine Webbrowser auf die Anwendungen zu, welche in der Cloud ausgeführt werden. Dadurch lassen sich Anwendungen aktualisieren ohne, dass der Anwender etwas davon mitbekommt, geschweige denn ein Update durchführen muss. [P M11]

3 Internet of Things (IoT)

3.1 Was ist IoT

Unter IoT wird die Vernetzung vieler physicher Geräte mittels dem Internet verstanden. Der Begriff ist bereits seit 1999 in Gebrauch[Pra18] und wird für eine vielzahl unterschiedlicher technischer Lösungen und Protokolle verwendet. Die verschiedenen technischen Lösungen eint dabei die Verwendung des Internet Protokolls (IP) innerhalb des Protokoll-Stacks[RS19b].
Der erste Schritt in Richtung IoT ist dabei für gewöhnlich die Ausstattung von Geräten mit einer vielzahl von Sensoren. So können Geräte den Zustand ihrer Umgebung erfassen und diese über die Netzwerkverbindung an ein Backend übertragen. Dadurch wird es möglich physische Zustände lokal, aber potenziell auch von überall auf der Welt aus zu überwachen. Das Backend kann also zum einen einer Privatperson ermöglichen die eigenen Geräte zu überwachen, zum anderen können die Geräte auch als Datenquellen zum Beispiel für Unternehmen dienen, welche dann die Informationen vieler Geräte in einem Cloud-Backend bündeln[Nav16].

Für viele technische Anwendungen ist neben Sensordaten auch die Möglichkeit der Steuerung von Aktoren notwendig. Hierbei ist sowohl die Vernetzung mit anderen IoT-Geräten als auch die Vernetzung mit dem Backend wichtig. Über die Kopplung mit einer IoT-Sensoreinheit kann so zum Beispiel die Regelung des Aktors vorgenommen werden. Wird der Aktor an ein Backend angeschlossen, ist darüber eine automatische Fernsteuerung möglich[RS22].
Neben Sensoren und Aktoren sind Einheiten für die Verarbeitung von Daten notwendig. Dabei kann die Datenverarbeitung sowohl im Backend geschehen, als auch durch ein IoT-Gerät. Die Verarbeitungslogik kann sowohl in ein bestehendes IoT-Gerät integriert werden, als auch als separate Einheit ausgeführt sein. Durch die so geschaffene Möglichkeit der lokalen Verarbeitung von Daten wird das Gesamtsystem resilienter gegen den Ausfall von Netzwerkverbindungen. Auch die Latenz wird durch die direktere Verbindung im lokalen Netzwerk verringert[RS22].

3.2 Anwendungsgebiete

Durch die bereits beschriebene Vielfältigkeit der technischen Lösungen, die unter dem Begriff IoT zusammengefasst werden ergibt sich auch eine Vielzahl an Anwendungsgebieten und Einsatzmöglichkeiten.
Einer der präsentesten Bereiche in dem IoT-Geräte eingesetzt werden ist die Hausautomatisierung. Hier eingesetzte IoT-Geräte können zum einen dem Komfort der Benutzer*innen erhöhen, in dem Arbeiten im Haushalt automatisiert oder erleichtert werden. Darüber hinaus existieren bereits viele Lösungen, welche primär der Sicherheit gegen Einbruch etc. dienen. Eine weitere wichtige Aufgabe von IoT-Geräten im Haus ist die energetische Optimierung. So werden „intelligente“ Heizsysteme mit Temperaturfühlern und Aktoren an den Rolläden gekoppelt, um die gewünschte Temperatur im Innenraum möglichst kostengünstig und Emissionsarm zu erreichen. Auch die Anpassung des elektrischen Energieverbrauchs an die Verfügbarkeit und den Preis von elektrischer Energie kann mittels IoT erfolgen[ZG20].
Auch für die Sammlung medizinisch relevanter Daten werden vermehrt IoT-Geräte eingesetzt. Ein Beispiel kann die Messung von Blutruck oder die Durchfühung eines Elektrokardiogramms durch ein Armband sein. Da diese permanent getragen werden, können Gesundheitliche Probleme auch ohne Verdachtsfall erkannt werden. Eine rechtzeitige Behandlung wird so wahrscheinlicher[ZG20].

In der Landwirtschaft können IoT-Geräte dazu verwendet werden zum Beispiel das Pflanzenwachstum zu überwachen. Durch die Erfassung der Daten kann optimal bewässert und gedüngt werden. Auch vollautomatische Systeme können mittels Steuerungstechnik realisiert werden. IoT bietet dabei die Möglichkeit Sensordaten zu erfassen und im Cloud-Backend mit Wetterdaten zu verknüpfen um langfristig die Bewässerung planen und Optimieren zu können[ZG20].
Der Vermutlich wichtigste Einsatzbereich von IoT ist die Industrie. Der Einsatz von IoT-Geräten in der Industrie wird in Deutschland mit dem Begriff „Industrie 4.0“ beschrieben. Mögliche aufgaben von IoT-Geräten sind die Bereitstellung von Daten zur Bestimmung wartungsintervallen oder aber auch die Automatisierung ganzer Produktionsstraßen. Viele Unternehmen streben dabei eine sogenannte „Smart Factory“ an, also eine Produktionsstraße, welche unterschiedliche Produkte fertigen kann und die Produktionsdaten von einem oder mehreren in dem Produkt integrierten IoT-Geräten erhält. Alle an dem Fertigungsprozess beteiligten Geräte sind dabei untereinander und mit dem Produkt vernetzt. Im idealfall ist für ein Angepasstes Produkt nur eine Änderung der im Produkt gespeicherten Produktionsdaten notwendig, ohne dass die am Fertigungsprozess beteiligten Geräte neu Programmiert oder anderweitig angepasst werden müssen[ZG20][MSG20].

3.3 Technologische Umsetzung von IoT

Da eine Ethernet-Verbindung für IoT-Geräte oft nicht verfügbar oder für den Anwendungsfall ungeeignet ist, kommunizieren ein Großteil der IoT-Geräte Drahtlos. So können auf den Layern Eins und Zwei zum Beispiel die energiesparenden Funkstandards IEEE 802.15.4 oder LPWAN bzw. LoRaWAN eingesetzt werden. Spielt der Energieverbrauch eines IoT-Geräts keine übergeordnete Rolle, da das Gerät zum Beispiel permanent mit Strom versorgt wird, werden auch WLAN oder der Mobilfunk für die Kommunikation eingesetzt (vgl. Abbildung 3.1). Auf Layer Drei werden für die Protokoll-Stacks hauptsächlich IPv4 und IPv6 bzw. 6LoWPAN verwendet[RS19a].
Auf den darüberligenden Layern gibt es eine Vielzahl unterschiedlicher Prokotkolle und Standards mit denenIoT-Geräte kommunizieren. Sie erfüllen jeweils unterschiedliche Anforderungen, wie Latenz oder Sicherheit, die an die Kommunikation gestellt werden. Darüber hinaus verwenden einige Unternehmen eigene proprietäre Standards für die Anbindung ihrer Geräte. Ein Beispiel für ein besonders in der Hausautomatisierung verbreitetes ofennes Protokoll ist MQTT. Neben der Verwendung spezieller IoT-Protokolle gibt es

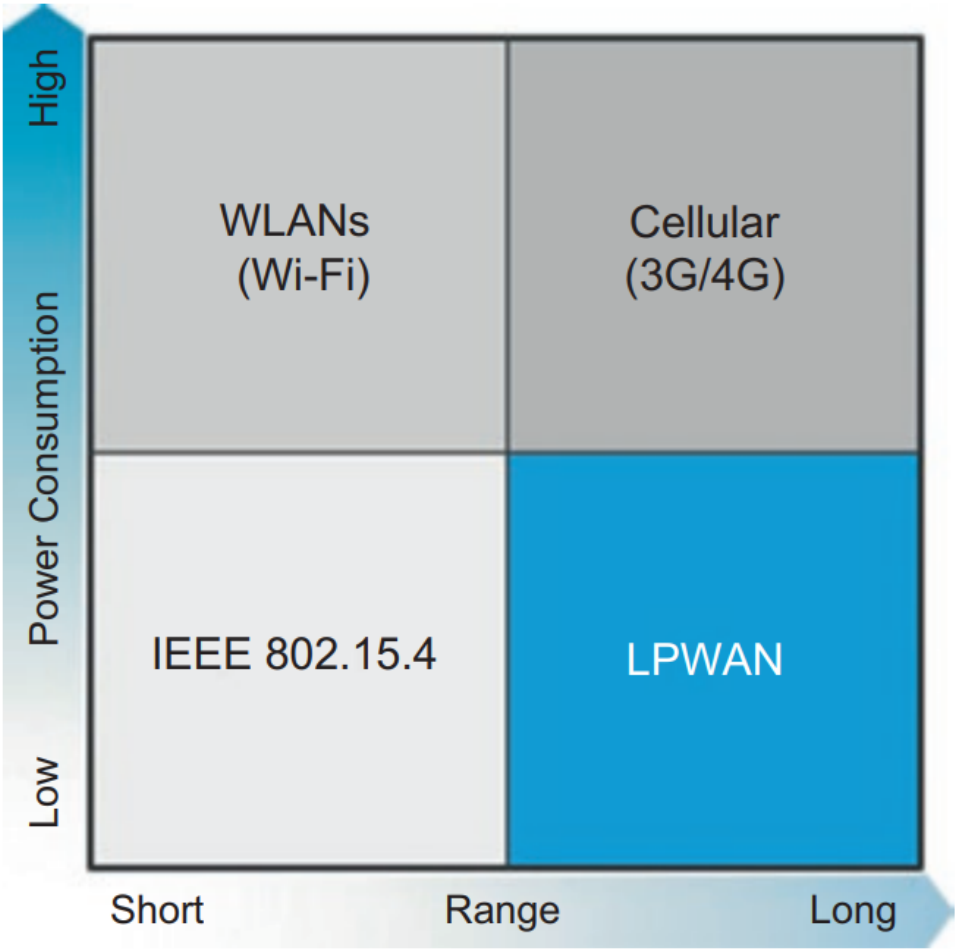

Abbildung 3.1
Funkstandards[RS19a]

auch die Möglichkeit Websocket- oder REST-APIs für die Kommunikation zu verwenden[RS19a].

4 Amazon Web Services (AWS)

Das Unternehmen Amazon bietet selbst mehr als Erfolgreich Cloud Computing Dienste unter dem Namen Amazone Web Services an. Anfänglich wurden nicht genutzte Rechenkapazitäten verkauft, jedoch hat sich das Anbieten von IT-Ressourcen in der Cloud als extrem profitabel erwiesen und mittlerweile erzielt das Cloud Geschäft des Unternehmens mehr Gewinn als der bekannte Online Handel. Bei AWS handelt es sich aktuell um den Cloud Anbieter mit dem weltweit größten Marktanteil und derzeit werden den Benutzern mehr als 200 verschiedene Dienste angeboten. Somit stehen den Benutzern alle erdenklichen Services zur Verfügung[Bra20].

Im Bereich IaaS bietet der Anbieter beispielsweise Services wie EC2 und S3 an. Mit EC2 wird dem Benutzer Rechenleistung in Form von Instanzen auf dem Server bereitgestellt und dieser hat dabei die vollständige Kontrolle über die virtuellen Ressourcen. Der Benutzer bezahlt hier lediglich die in Anspruch genommene Rechenleistung. Amazons simple Storage Service S3 bietet einen Objektspeicher mit Web-Schnittstellen. Es ist möglich große Datenmengen schnell in den Datenspeicher hinzuzufügen oder sie abzurufen. Neben S3 bietet Amazon noch weitere Speicherdienste wie z.B. Backup, welche weniger schnelle Zugriffe ermöglichen aber dadurch auch erheblich günstiger pro genutzte Speicherkapazität sind. [Ama18]

Auch oberhalb der genannten Ebenen bietet AWS Services wie Lambda an. Dieser ist oberhalb von SaaS in der Ebene Function as a Service einzuordnen, denn der Benutzer besitzt weniger Verantwortung als bei SaaS. Mithilfe von Lambda ist es möglich Code einfach Serverseitig durch definierte Trigger ausführen zu lassen. Der Kunde zahlt hier je nachdem wie oft der Code ausgeführt wird. [Ama18]

In Bezug auf das Thema IoT stellt Amazon eine Flotte an Services bereit. FreeRTOS und GreenGrass stellen hier Betriebssysteme dar, welche auf den IoT Geräten laufen und bereits vordefinierte Schnittstellen zur Kommunikation mit der Cloud besitzen. FreeRTOS dient hier als Echtzeitbetriebssystem für die einzelnen IoT Geräte und GreenGrass eher

als ein Verwaltungsbetriebssystem z.B. zum Sammeln und Synchronisieren der erfassten Daten auf einem übergeordneten Gerät. [Ama18]

Um eine Steuerung und Überwachung in der Cloud zu ermöglichen dienen Dienste wie IoT Core, Fleetwise und Device Manegement. Durch diese lassen sich Flotten an IoT Geräten verwalten, Überwachen und auch Steuern. Zudem ermöglichen diese ein Sammeln der Daten und ermöglichen ein Updaten der Gerätesoftware von zentraler Stelle.[Ama18]

Zur weiteren Verarbeitung der Daten gibt es weitere Services wie IoT Sietwise, Analytics und Events. Beispielsweise lassen sich Daten für das weitere Verarbeiten mithilfe von Machine learning tools vorbereiten durch Filtern, Transformieren und Anreichern. Aus den gesammelten Daten lassen sich Aktionen erstellen, die dann wieder von den IoT Geräten ausgeführt werden. Ein Anwendungsfall hierfür ist eine Maschine in einer Produktionshalle. Durch das Auswerten der gesammelten Daten der gesamten Sensorik, lassen sich Rückschlüsse auf den Zustand der Maschine schließen. Somit ist es möglich vollständig automatisiert in die Produktion einzugreifen, sobald eine Unregelmäßigkeit im Betrieb festgestellt wird.[Ama18]

5 ESP32 + AWS Core IoT

5.1 AWS Core IoT

Wie bereits erwähnt kann für die Anbindung eines IoT-Geräts an die AWS-Cloud der AWS Core IoT Service genutzt werden. Dieser stellt die Basis aller IoT bezogenen AWS Dienste. Damit ein Gerät angebunden werden kann, muss dieses zunächst als IoT Core Gerät in der AWS Console angelegt werden. Bei diesem Prozess werden Zertifikate erstellt, mit denen sich ein Gerät bei AWS authentifizieren kann. Diese können heruntergeladen und in die Gerätesoftware eingebettet werden[Toc22b].
Um Gerätesoftware für die Anbindung an AWS Core IoT zu entwickeln, bietet AWS unterschiedliche Frameworks und Software Development Kits an. Diese beinhalten vorgegeben Schritte um die Zertifikate einzubinden. Auch die Kommunikation über MQTT mit der AWS-Cloud kann über vorgefertigte Bibliotheken geschehen[Toc22c].
Bei der Erstellung des Gerätes in der AWS Console wird bereits für jedes Gerät eine

API am AWS Core IoT Endpunkt geschaffen. Darüber hinaus bietet der Endpunkt unter „/mqtt“ bereits einen MQTT-Server auf dem MQTT-Topic erstellt werden können. Mit dem in der Console verfügbaren MQTT-Test-Client kann die Verbindung zu den einzelnen Geräten getestet werden.

5.2 Demo-Projekt Quizz-Buzzer

Zur Demonstration der Funktionen von AWS Core IoT wird ein Quizz-Buzzer auf Basis eines ESP32 als IoT-Gerät entwickelt. Konkret wird das Entwicklungsboard „ESP32-S devkitC V4“ mit dem ESP-WROOM-32U verwendet. Das Modul verfügt über eine WLAN Schittstelle und zwei Prozessorkerne, sowie eine Vielzahl digitaler Ein- und Ausgänge. Wie in Abbildung Abbildung 5.1 dargestellt werden zwei LEDs über zwei digitale Ausgänge beschaltet und vier Taster an digitale Eingänge geschaltet.

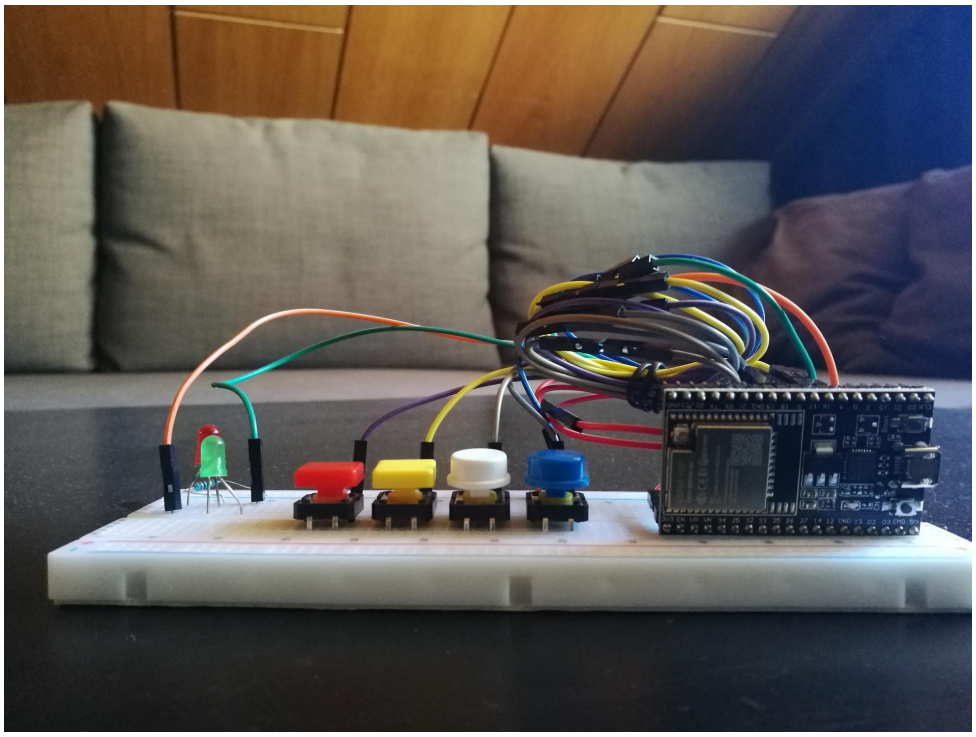

Abbildung 5.1
Buzzer auf Basis eines ESP32 (Eigendarstellung)

Der Quizz-Buzzer ist in der Lage einem dem jeweiligen Taster zugeordneten Buchstaben von A bis D per MQTT an ein MQTT-Topic am Endpunkt in der AWS-Cloud zu schicken.

Für die Übertragung wird das JSON-Format genutzt, das Format ist in Abbildung Abbildung 5.2 dargestellt. Wird auf dem gleichen MQTT-Topic von einem anderen Teilnehmer mit „isCorrect“ geantwortet wird entweder die rote oder die grüne LED für einen kurzen Zeitraum eingeschaltet.
Der in gekürzer Fassung in Abschnitt A.1 dargestellte Programmcode für den Buzzer wird mit der Espressif 32: development platform for PlatformIO und dem AWS IoT Device SDK for Embedded C entwickelt. Die Basis für die Kommunikation mit AWS Core IoT ist dabei Beispielprogramm ESPIDF-AWS-IoT entnommen.
Um die Latenz des Buzzers Möglichst gering zu halten werden jeweils Teile des Programmcodes auf einem der zwei Kerne ausgeführt. Das Handling von WiFi-Events, wie zum Beispiel einem Verbindungsabbruch wird von dem Setup-Code (vgl. Abschnitt A.1) aus aufgerufen und läuft somit auf Kern 0. Gleiches gilt für die Verarbeitung der Eingangssignale der Taster, die Inerrupts werden dem Kern 0 zugeordnet(vgl. Abschnitt A.1). Die eigentliche Kommunikation mittels MQTT, sowie das Parsen und erstellen von Strings im JSON-Format wird in einer auf Kern 1 ausgeführten Task bearbeitet(vgl. Abschnitt A.1). Um die Information über das Drücken eines Tasters an die Kommunikations-Task zu

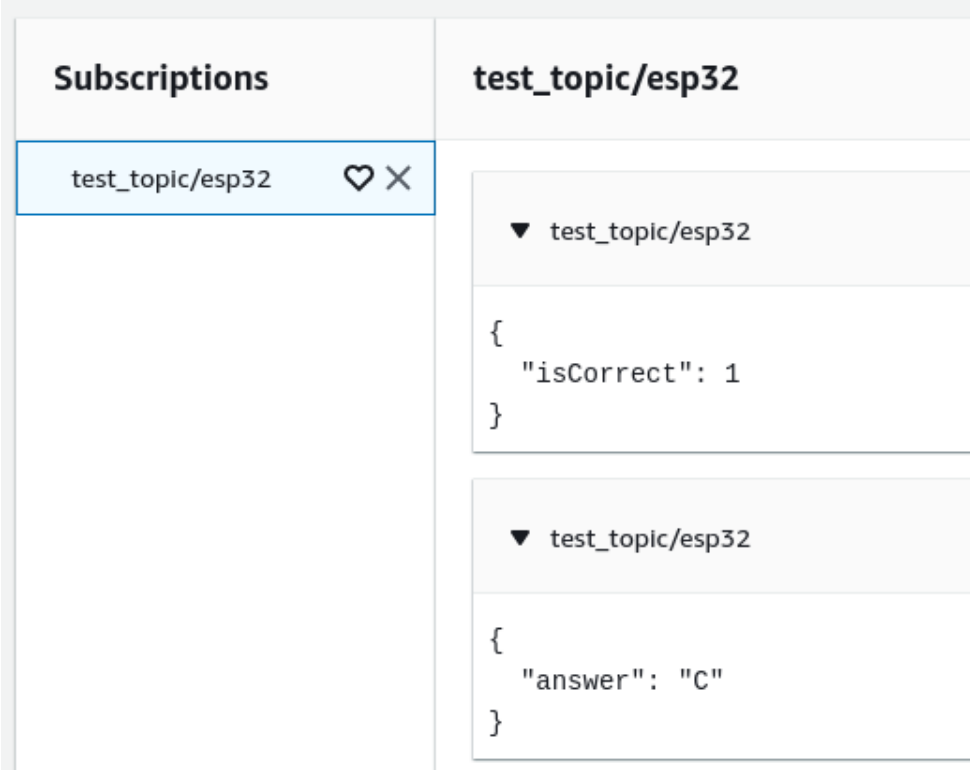

Abbildung 5.2
MQTT-Test-Client (Eigendarstellung)

übermitteln wird eine FreeRTOS-Queue (bereits in der Platform enthalten) verwendet.

Wird ein Taster gedrückt, wird die Programmausführung auf Kern 0 unterbrochen und die Interrupt Service Routine ausgeführt. Diese sendet die Information darüber welcher Taster gedrückt wurde an die Queue. Die Kommunikations-Task prüft die Queue zyklisch auf neue Elemente und versendet die entsprechenden Informationen. Dazu wird ein zuvor erstelltes JSON Objekt angepasst und als String mit der „publish()“Funktion versendet. Durch Auslagerung der Interrupts wird sichergestellt, dass die Kommunikation nicht unterbrochen wird.

6 Zusammenfassung

Abschließend lässt sich sagen, dass sich die beiden Technologien IoT und Cloud Computing optimal ergänzen. Cloud-Dienstleister wie AWS machen es mit ihren Cloud-Diensten verhältnismäßig einfach sichere und skalierbare Lösungen zu entwickeln. Mit AWS Core IoT und den anderen IoT-Diensten bietet AWS die Möglichkeit, auf IoT zugeschittene Cloud-Produkte zu nutzen. Das Demoprojekt zeigt die wichtigsten Schritte die zum Einrichten eines simplen IoT-Geräts notwendig sind und wie AWS Core IoT im Zusammenspiel mit dem AWS IoT Device SDK for Embedded C funktioniert.

Literatur

[Ama18] Inc. und/oder seine Partner Amazon Web Services. *Übersicht über Amazon Web Services*. online. 2018. URL: https://de.statista.com/infografik/20802/weltweiter-marktanteil-von-cloud-infrastruktur-dienstleistern/.

[Bra19] Mathias Brandt. *Wie groß sind die "Next Big Things"tatsächlich?* online. 2019. URL: https://de.statista.com/infografik/16558/geschaetzte-einnahmen-aus-dem-verkauf-von-tech-produkten-in-den-usa/.

[Bra20] Mathias Brandt. *Amazon ist die Nummer 1 in der Cloud*. online. 2020. URL: https://de.statista.com/infografik/20802/weltweiter-marktanteil-von-cloud-infrastruktur-dienstleistern/.

[Bra22] Mathias Brandt. *Österreichs Wirtschaft setzt auf IoT*. online. 2022. URL: https://de.statista.com/infografik/26712/umfrage-zur-nutzung-des-internets-der-dinge-durch-unternehmen-in-europa/.

[Inc11] VMware Inc. *How VMware Virtualization Right-sizes IT Infrastructure to Reduce Power Consumption*. online. 2011. URL: https://www.vmware.com/content/dam/digitalmarketing/vmware/en/pdf/vmware-reduce-power-consumption-wp.pdf.

[Jan20] Matthias Janson. *Cloud Computing auf dem Vormarsch*. online. 2020. URL: https://de.statista.com/infografik/22671/nutzung-von-cloud-computing-in-unternehmen-in-deutschland/.

[MSG20] Hardik Majiwala, Suresh Sharma und Pankaj Gandhi. „Lean and Industry 4.0 Strive to Create Smart Factory Through Integration of Systems: An Exploratory Review". In: *4th International Conference on Internet of Things and Connected Technologies (ICIoTCT), 2019*. Hrsg. von Neeta Nain und Santosh Kumar Vipparthi. Cham: Springer International Publishing, 2020, S. 184–195. ISBN: 978-3-030-39875-0.

[Nav16] Soumyalatha Naveen. „Study of IoT: Understanding IoT Architecture, Applications, Issues and Challenges". In: Mai 2016.

[P M11] T. Grance P. Mell. „The NIST definition of cloud computing“. In: *National Institute of Standards and Technology* 1 (2011).

[Pra18] Sagar Bhat Pradyumna Gokhale Omkar Bhat. „Introduction to IOT“. In: *International Advanced Research Journal in Science, Engineering and Technology* 5.1 (2018).

[Rep13] Jonas Repschläger. „Entscheidungsfindung im Cloud Computing – Konzeption und Analyse eines Modells zur Anbieterauswahl“. Diss. Technischen Universität Berlin, 2013. URL: https://d-nb.info/1065664664/34.

[RS19a] Ammar Rayes und Samer Salam. „IoT Protocol Stack: A Layered View“. In: *Internet of Things From Hype to Reality: The Road to Digitization*. Cham: Springer International Publishing, 2019, S. 103–154. ISBN: 978-3-319-99516-8. DOI: 10.1007/978-3-319-99516-8_5. URL: https://doi.org/10.1007/978-3-319-99516-8_5.

[RS19b] Ammar Rayes und Samer Salam. „The Internet in IoT“. In: *Internet of Things From Hype to Reality: The Road to Digitization*. Cham: Springer International Publishing, 2019, S. 37–65. ISBN: 978-3-319-99516-8. DOI: 10.1007/978-3-319-99516-8_2. URL: https://doi.org/10.1007/978-3-319-99516-8_2.

[RS22] Ammar Rayes und Samer Salam. „Internet of Things (IoT) Overview“. In: *Internet of Things from Hype to Reality: The Road to Digitization*. Cham: Springer International Publishing, 2022, S. 1–34. ISBN: 978-3-030-90158-5. DOI: 10.1007/978-3-030-90158-5_1. URL: https://doi.org/10.1007/978-3-030-90158-5_1.

[Til10] Gottfried Vossen Till Haselmann. *Database-as-a-Service für kleine und mittlere Unternehmen*. online. 2010. URL: https://www.uni-muenster.de/imperia/md/content/angewandteinformatik/aktivitaeten/publikationen/daas-fuer-kmu.pdf.

[Toc22a] Amazon Web Services Inc. oder Tochterfirmen. *Was ist Cloud Computing?* online. 2022. URL: https://aws.amazon.com/de/what-is-cloud-computing/.

[Toc22b] Amazon Web Services Inc. oder Tochtergesellschaften. *AWS IoT API Reference*. Online. 2022. URL: https://docs.aws.amazon.com/iot/latest/apireference/Welcome.html.

[Toc22c] Amazon Web Services Inc. oder Tochtergesellschaften. *AWS IoT Device SDK for Embedded C*. Online. 2022. URL: https://github.com/aws/aws-iot-device-sdk-embedded-C.

[Vaq+09] L. Vaquero et al. „A break in the clouds: Towards a Cloud Definition“. In: Hrsg. von xxx. ACM SIGCOMM Computer Communication Review 39, 2009. Kap. 1, S. 50.

[Wir22] Bernd W. Wirtz. „Internet of Things, Cloud Computing und Big Data“. In: *Multi-Channel-Marketing: Grundlagen – Instrumente – Prozesse*. Wiesbaden: Springer Fachmedien Wiesbaden, 2022, S. 211–235. ISBN: 978-3-658-03345-3. DOI: 10.1007/978-3-658-03345-3_12. URL: https://doi.org/10.1007/978-3-658-03345-3_12.

[Zar12] Rüdiger Zarnekow. *Grundlagen der Cloud Computings - Anforderungen an einen Cloud Dienst*. online. 2012. URL: http://nbn-resolving.de/urn:nbn:de:kobv:83-opus4-40460.

[ZG20] Zaheeruddin und Hina Gupta. „Foundation of IoT: An Overview“. In: *Internet of Things (IoT): Concepts and Applications*. Hrsg. von Mansaf Alam, Kashish Ara Shakil und Samiya Khan. Cham: Springer International Publishing, 2020, S. 3–24. ISBN: 978-3-030-37468-6. DOI: 10.1007/978-3-030-37468-6_1. URL: https://doi.org/10.1007/978-3-030-37468-6_1.

A Anhang

A.1 Quizz Buzzer Codebeispiel (gekürzt)

```
...

static QueueHandle_t gpio_evt_queue = NULL;

// Interrupt Service Routine
static void gpio_isr_handler(void* arg)
{
    gpio_num_t gpio_num = (gpio_num_t)arg;
    xQueueSendFromISR(gpio_evt_queue, &gpio_num, NULL);
}

...

// Einbinden der Zertifikate fuer AWS Core IoT
extern const uint8_t aws_root_ca_pem_start[]
    asm("_binary_aws_root_ca_pem_start");
extern const uint8_t aws_root_ca_pem_end[]
    asm("_binary_aws_root_ca_pem_end");
extern const uint8_t certificate_pem_crt_start[]
    asm("_binary_certificate_pem_crt_start");
extern const uint8_t certificate_pem_crt_end[]
    asm("_binary_certificate_pem_crt_end");
extern const uint8_t private_pem_key_start[]
    asm("_binary_private_pem_key_start");
extern const uint8_t private_pem_key_end[]
    asm("_binary_private_pem_key_end");

...

static void event_handler(
    void* handler_args, esp_event_base_t base, int32_t id,
    void* event_data)
```

```
33 {
34     ...
35 }
36
37 // Verarbeiten eingehender Nachrichten
38 void iot_subscribe_callback_handler(
39     AWS_IoT_Client* pClient, char* topicName, uint16_t topicNameLen,
40     IoT_Publish_Message_Params* params, void* pData)
41 {
42     cJSON *root, *isCorrect = NULL;
43     root = cJSON_ParseWithLength((char*)params->payload,
44         (size_t)params->payloadLen);
45     isCorrect = cJSON_GetObjectItem(root, "isCorrect");
46     if (isCorrect) {
47         gpio_set_level(GPIO_OUTPUT_IO_0,
48             cJSON_GetNumberValue(isCorrect));
49         gpio_set_level(GPIO_OUTPUT_IO_1,
50             !cJSON_GetNumberValue(isCorrect));
51         vTaskDelay(pdMS_TO_TICKS(800));
52         gpio_set_level(GPIO_OUTPUT_IO_0, 0);
53         gpio_set_level(GPIO_OUTPUT_IO_1, 0);
54     }
55 }
56
57 void disconnectCallbackHandler(AWS_IoT_Client* pClient, void* data)
58 {
59     ...
60 }
61
62 // Hauptprogramm
63 void aws_iot_task(void* param)
64 {
65     ...
66
67     do {
68         rc = aws_iot_mqtt_connect(&client, &connectParams);
69         if (SUCCESS != rc) {
70             ESP_LOGE(TAG, "Error(%d) connecting to %s:%d",
71                 rc, mqttInitParams.pHostURL, mqttInitParams.port);
72             vTaskDelay(1000 / portTICK_RATE_MS);
73         }
74     } while (SUCCESS != rc);
75
```

```
76      ...
77
78      const char* TOPIC = "test_topic/esp32";
79      const int TOPIC_LEN = strlen(TOPIC);
80
81      ESP_LOGI(TAG, "Subscribing...");
82      rc = aws_iot_mqtt_subscribe(
83          &client, TOPIC, TOPIC_LEN, QOS0,
84          iot_subscribe_callback_handler, NULL);
85      if (SUCCESS != rc) {
86          ESP_LOGE(TAG, "Error subscribing : %d ", rc);
87          abort();
88      }
89
90      cJSON *root, *answer;
91      root = cJSON_CreateObject();
92      answer = cJSON_CreateString("A");
93      cJSON_AddItemToObject(root, "answer", answer);
94
95      sprintf(cPayload, "%s", cJSON_Print(root));
96
97      paramsQOS0.qos = QOS0;
98      paramsQOS0.payload = (void*)cPayload;
99      paramsQOS0.isRetained = 0;
100
101     // Senden von Nachrichten an den MQTT Endpunkt von AWS Core IoT
102     while ((NETWORK_ATTEMPTING_RECONNECT == rc
103         || NETWORK_RECONNECTED == rc || SUCCESS == rc)) {
104
105         ...
106
107         gpio_num_t io_num;
108         if (xQueueReceive(gpio_evt_queue,
109             &io_num, pdMS_TO_TICKS(100))) {
110             const char answerLetter = 'A' + io_num - GPIO_INPUT_IO_0;
111             cJSON_SetValuestring(answer, &answerLetter);
112             sprintf(cPayload, "%s", cJSON_Print(root));
113             paramsQOS0.payloadLen = strlen(cPayload);
114             rc = aws_iot_mqtt_publish(
115                 &client, TOPIC, TOPIC_LEN, &paramsQOS0);
116         }
117     }
118
```

```
119     ESP_LOGE(TAG, "An error occurred in the main loop.");
120     abort();
121 }
122 
123 static void initialise_wifi(void)
124 {
125     ...
126 }
127 
128 // Setup-Code
129 void app_main()
130 {
131     esp_log_level_set("*", ESP_LOG_NONE);
132 
133     gpio_config_t io_conf = {};
134     io_conf.intr_type = GPIO_INTR_DISABLE; // disable interrupt
135     io_conf.mode = GPIO_MODE_OUTPUT; // set as output mode
136     io_conf.pin_bit_mask = GPIO_OUTPUT_PIN_SEL; // bit mask of the pins
137     io_conf.pull_down_en = 0; // disable pull-down mode
138     io_conf.pull_up_en = 0; // disable pull-up mode
139     gpio_config(&io_conf);
140 
141     io_conf.intr_type = GPIO_INTR_POSEDGE; // interrupt of rising edge
142     io_conf.pin_bit_mask = GPIO_INPUT_PIN_SEL; // bit mask of the pins
143     io_conf.mode = GPIO_MODE_INPUT; // set as input mode
144     io_conf.pull_down_en = 1; // enable pull-up mode
145     gpio_config(&io_conf);
146 
147     gpio_evt_queue = xQueueCreate(10, sizeof(gpio_num_t));
148 
149     gpio_install_isr_service(ESP_INTR_FLAG_DEFAULT);
150     gpio_isr_handler_add(GPIO_INPUT_IO_0, gpio_isr_handler,
151         (void*)GPIO_INPUT_IO_0);
152     gpio_isr_handler_add(GPIO_INPUT_IO_1, gpio_isr_handler,
153         (void*)GPIO_INPUT_IO_1);
154     gpio_isr_handler_add(GPIO_INPUT_IO_2, gpio_isr_handler,
155         (void*)GPIO_INPUT_IO_2);
156     gpio_isr_handler_add(GPIO_INPUT_IO_3, gpio_isr_handler,
157         (void*)GPIO_INPUT_IO_3);
158 
159     ...
160 
161     initialise_wifi();
```

```
162     xTaskCreatePinnedToCore(
163         &aws_iot_task, "aws_iot_task", 9216, NULL, 5, NULL, 1);
164 }
```