HSR
HOCHSCHULE FÜR TECHNIK
RAPPERSWIL

FHO Fachhochschule Ostschweiz

ins institute for networked solutions

IoT - Provisioning und Management

Bachelorarbeit FS 2017

Abteilung Informatik
Hochschule für Technik Rapperswil

Autoren: Andreas Stalder, David Meister
Betreuer: Prof. Beat Stettler, Urs Baumann
Projektpartner: INS Institute for Networked Solutions
Datum: 23. März 2017

Inhaltsverzeichnis

1. Einführung in Internet of Things

1.1 Übersicht

Das Internet der Dinge unterscheidet sich in einigen Aspekten vom klassischen Internet. End-Benutzer haben über sogenannte Terminals wie Laptops oder Smartphones über die globale Internet Infrastruktur kommuniziert [11]. Diese Terminals wurden meist von Benutzern eingeschaltet, benutzt und wieder ausgeschaltet. Damit Geräte mit dem Internet auf sinnvolle Art und Weise kommunizieren konnten, war eine manuelle Tätigkeit von Benutzern notwendig [17]. Beispiele dafür sind das Abrufen von E-Mails, Surfen im Web, Streaming von Videos oder Spielen von Online Games [11].

Mit „Internet of Things“ (IoT) wird eine andere Philosophie verfolgt. Es gibt keine einheitliche Definition und Abgrenzung von IoT. Grundsätzlich versucht man Objekte und Gegenstände, welche im klassischen Sinne des Internets nicht berücksichtigt wurden, ans Netz anzuschliessen. Mit minimalen menschlichen Eingriffen sollen diese Geräte Daten sammeln, austauschen und aufgrund von Software und Algorithmen Entscheidungen treffen [18]. Man spricht im Zusammenhang von „Things “ auch von „Smart Devices“ oder „Smart Objects“.

1.1.1 Smart Objects

Smart Objects oder auch „Things“ ergänzen das herkömmliche Internet um eine Vielzahl neuartiger Teilnehmer. Man ist versucht, die mit dem Internet erschaffene virtuelle Welt mit Objekten der tatsächlichen „echten“ Welt zu verbinden. Der Begriff „Smart“ ist seit der Erscheinung des iPhones weltweit bekannt. Er beschreibt die Fähigkeit eines Objekts mit dem Internet zu kommunizieren.

Während Smartphones oder Smart-TVs noch als herkömmliche Internet Terminals angesehen werden können, so erweitern die Smart Objects das bisherige Internet um eine neue Art von Teilnehmer. Smart Objects lassen sich wie folgt beschreiben:

- haben eine physikalische Repräsentation mit Eigenschaften wie Form und Grösse
- haben Mindestmass an Kommunikationsfunktionalitäten wie Request/Reply
- besitzen eine UID (unique identifier)
- haben mindestens einen Namen und eine Adresse
- besitzen ein Mindestmass an Rechenfähigkeiten
- besitzen Sensoren, um physikalische Erscheinungen wie Druck, Licht, Temperatur, etc. zu messen

Der letzte Punkt in der oberen Definition beschreibt den tatsächlichen Unterschied zu herkömmlichen Devices im Internet. Konzeptionell liegt bei IoT der Fokus mehr auf Daten und Informationen von physikalischen Objekten als bei Punkt-zu-Punkt Kommunikation von Terminals [11].

1.2 Einsatzgebiete

Das ”Internet of Things” hat viele Einsatzmöglichkeiten und wir stehen erst am Anfang. Es werden immer neue Einsatzbereich entdeckt und vorhandene optimiert und erweitert. Um die Einsatzmöglichkeiten aufzuzeigen, wird hier das Beispiel ”Gesundheitsvorsorge” genauer gezeigt.

1.2.1 Allgemein

Die Gesundheitsvorsorge ist ein riesiger Bereich, welche alle Menschen betrifft und enorme Kosten verursacht. Heutzutage wird noch sehr viel manuell erledigt. In naher Zukunft wird sich das ändern und das Internet der Dinge wird immer wichtiger, um die bestmögliche Behandlung jedes Patienten sicherzustellen. IoT bringt noch viele weitere Verbesserungen und in Zukunft werden immer neue Einsatzgebiete gefunden. Hier wird nur ein kleiner Teil beschrieben.

1.2.2 Im Alltag

Selbstüberwachung In den letzten paar Jahre ist die Selbstüberwachung zu einem riesigen Thema geworden. Firmen wie Apple oder Fitbit haben diesen Bereich stark gefördert. So gibt es heute schon für mehrere Körperdatensensoren, die alles Überwachen. Ein Beispiel dafür sind die Smartwatches. Diese können bereits heute den Puls rund um die Uhr überwachen oder sie zählen die Schritte mit. So kann jeder sein eigenes Fitnessprofil von sich erstellen. Ein weiteres Beispiel findet man bei Patienten, welche Medikamente verabreicht bekommen. Da Medikamente häufig vergessen werden, gibt es kluge Medikamentenspender, welche den Patienten benachrichtigen, wenn dieser die Medikamente nicht genommen hat.

Fernüberwachung Neben der Selbstüberwachung gibt es auch noch die Fernüberwachung. Dabei wird der Patient vom Pflegepersonal überwacht und bei kritischen Situationen kann sofort gehandelt werden. Für Rollstuhlfahrer oder Senioren gibt es zum Beispiel ein Falldetektor. Falls der Patient umfällt, reagiert der Sensor und meldet diesen Unfall direkt an das Pflegepersonal. So kann schnell reagiert und dem Patienten geholfen werden. Auch das Überwachen der Patientenwerte kann so über das Internet sichergestellt werden. Durch eine kontinuierlich Abfrage der Werte, ist die Reaktionszeit enorm kleiner, als beim manuellen messen. Hierfür benötigt man Pflegepersonal vor Ort.

1.2.3 Im Spital

Patientenzimmer Durch die komplette Überwachung des Patienten in seinem Zimmer, verkürzt sich die Reaktionszeit enorm. Der Patient wird zwar schon überwacht, aber durch die Anbindung an das Internet wären weitere Verbesserungen möglich. So kann man bei gewissen Problemen nicht nur das Pflegepersonal alarmieren, sondern zum Beispiel auch direkt benötigtes Personal aus anderen Spitälern anordnen. Auch könnte man die Messdaten aus der Fernüberwachung mit der aus dem Patientenzimmer verknüpfen und so ein besseres Patientenbild erstellen. Der Patient wäre so lückenlos überwacht und bei Problemen wären die Daten sehr hilfreich.

Infrastruktur In Spitälern werden viele verschiedenen Geräte eingesetzt, welche momentan noch nicht miteinander kommunizieren. Durch die Verknüpfung der Geräte könnte man die Daten verschiedener Geräte miteinander kombinieren und vergleichen, um ein besseres Bild erstellen zu können. Oder auch die Wartung würde viel einfacher werden. Aus der Ferne werden alle Geräte inventarisiert, gewartet und

überwacht. So wird bei einem Fehler sofort der Techniker gerufen, damit ein defektes Gerät sofort wieder Einsatzbereit gemacht wird.

1.3 Sensortypen

Beim Thema ”Internet of Things” spielen die Sensoren eine zentrale Rolle. Es gibt viele verschiedene Typen von Sensoren, welche unterschiedlichste Daten liefern und man muss daher wissen, wie man mit dem jeweiligen Sensortyp umgeht.

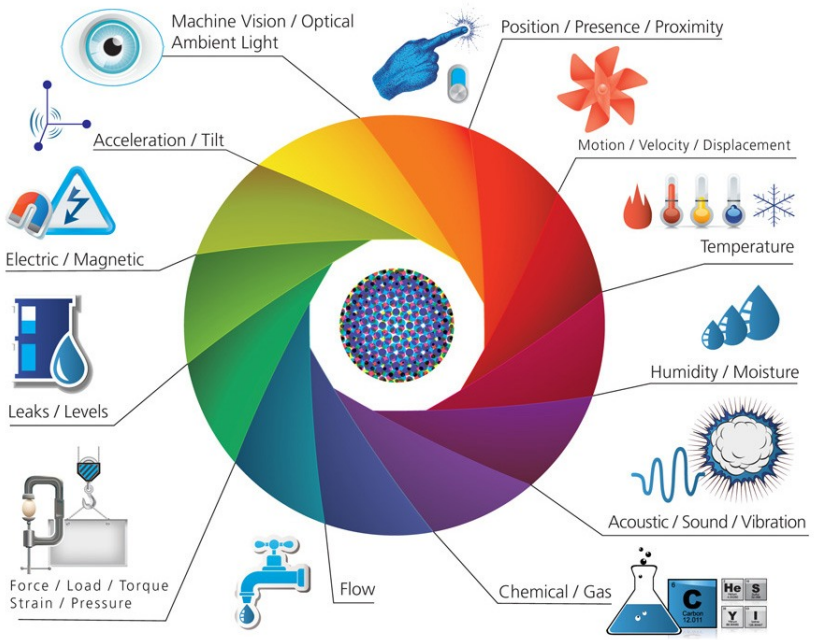

Abbildung 1.1: Sensortypen[16]

1.3.1 Temperature

Temperatursensoren werden in vielen Gebieten eingesetzt. Häufig wird dieser Sensortyp zur Überwachung von Gebäuden eingesetzt, oder hilft bei maschinell hergestellten Esswaren, die richtige Temperatur zu halten. Auch kann ein Bauer diese Sensoren verwenden, um die Bodentemperatur zu Überwachen. So kann man effizienter und gewinnbringender Arbeiten, ohne selber Messungen durchzuführen.

Bespielgeräte

- Smarte Heizungsteuerungen in einem Haushalt
- Wetterstationen

1.3.2 Acceleration/Tilt

Beschleunigungs- und Lagesensoren hat wohl jeder in seiner Hosentasche. Nahezu alle neuen Smartphones haben solche eingebaut. Auch in der Autoindustrie findet man solche sehr häufig. Durch solche Sensoren kann man viele Verschiedene Daten erhalten. So kann man zum Beispiel Bewegungsprofile einer Person erstellen und den Fitnesslevel bestimmen.

Bespielgeräte

- Schrittzähler (z.B in Smart Watches)

1.3.3 Acoustic/Sound/Vibration

Nicht nur in der Musikbranche sind Akustik- und Soundsensoren sehr wichtig. So wird auch der Lärm in einem Gebiet oder in einer Stadt gemessen, um Verbesserungen der Lebensqualität zu erreichen. Sehr wichtig sind auch die Vibrationssensoren, welche wichtige Daten zu Unterwassererdbeben senden. So können Tsunamis immer früher erkannt werden und retten Leben.

Bespielgeräte

- Erdbebenwarnsysteme
- Sprachsteuerungen

1.3.4 Chemical/Gas

Chemikaliensensoren werden häufig in den Städten mit viel Verkehrsaufkommen eingesetzt. Dadurch wird die Luftqualität bestimmt. Auch in Laboren ist dies ein wichtiger Sensor, um die Qualität oder Reinheit von Gasen zu messen.

Bespielgeräte

- Smart City Luftüberwachung
- Überwachung von gasbetriebenen Geräten

1.3.5 Electric/Magnetic

Magnetische Sensoren könnten in verschiedenen Geräten verbaut werden. Zum Beispiel smarte Türschlösser könnten via solchen Sensoren geschlossen oder geöffnet werden. Auch Elektrizitätswerke können diverser solche Sensoren verwenden, um die Systeme zu überwachen.

Bespielgeräte

- Smarte Schlösser
- Stromüberwachung

1.3.6 Flow

Für die Überwachung von Flüssen oder Wasserleitungen werden Flusssensoren verwendet. So können zum Beispiel Wasserversorger den Verbrauch jedes Haushalts über das Internet messen lassen und müssen nicht vor Ort die Zähler ablesen. Oder auch für die Überwachung von Flüssen kann dieser Sensor verwendet werden. So wird man bei zu schnellen und zu vielem Wasser vor Überschwemmungen gewarnt.

Bespielgeräte

- Smarte Wasserversorgungen

- Überschwemmungsschutz

1.3.7 Force/Load/Torque/Strain/Pressure

Im Fitnessbereich gibt es schon seit Jahren mehrere Körperwaagen, welche solche Sensoren verwenden. Man wird gewogen und gleichzeitig sendet das Gerät allerlei Daten an den Cloud-Dienst. Auch Parksysteme oder automatische Wiegesysteme in Lagern sind Beispiele, welche bereits im Einsatz sind. Diese Sensoren sind vielfältig einsetzbar.

Bespielgeräte

- Wiegesysteme/Körperwaage
- Parksysteme

1.3.8 Humidity/Moisture

Ein wichtiger Bestandteil der Luftqualität ist auch die Luftfeuchtigkeit. Diese wird mit diesem Typ gemessen. So können Smart Buildings die Luftfeuchtigkeit laufend messen und immer wieder optimieren. Auch in der Landwirtschaft kann so eine Automation eingeführt werden, damit die Erde immer optimal bewässert ist.

Bespielgeräte

- Bewässerungsanlagen
- Pflanzensensoren

1.3.9 Leaks/Levels

Lecks- und Levelsensoren sind zum Beispiel in der Landwirtschaft notwendig. Die Landwirtschaft benötigt viel Wasser und man möchte unnötige Lecks vermeiden. Ein weiterer wichtiger Einsatzbereich ist die Überwachung von Flüssigkeitsständen, zum Beispiel bei Staudämmen oder in Lagersystemen.

Bespielgeräte

- Leitungsüberwachungen
- Lagerverwaltungen

1.3.10 Machine Vision / Optical Ambient Light

Machine Vision ist ein immer wichtig werdender Sensor. Durch diese kann man den Dingen das Sehen ”lehren”. So sind automatische Eintrittkontrollen möglich. Auch in den SmartCars kommen diese Sensoren zum Einsatz. Da werden durch die Sensoren die Fussgänger, die Fahrbahn oder auch andere Autos erkannt. Bei Optical Ambient Light geht es um Sensoren, welche die Umgebungsbeleuchtung messen und diese Optimal anpassen. So gibt es schon mehrere smarte Leuchtmittel, welche so über das Internet eingestellt werden können.

Bespielgeräte

- SmartCars
- Eintrittskontrollen

1.3.11 Motion/Velocity/Displacement

Bewegungsensoren sind praktische Helfer bei Sicherheitssystemen. Eine zentrale Überwachung von mehreren Gebäuden ist so problemlos möglich. Bei der Pflege von Rollstuhlfahrer wäre zum Beispiel auch eine Falldetektion möglich. Das zentrale Pflegezentrum hätte so eine schnelle Meldung über Unfälle und könnte mehrere Personen überwachen und betreuen.

Bespielgeräte

- Sicherheitsanlagen
- Verkehrsüberwachung

1.3.12 Position/Presence/Proximity

Immer wichtiger wird auch dieser Typ von Sensoren. Durch die Bestimmung von Distanzen oder der Position in einem Raum, ergeben sich viele Einsatzmöglichkeiten. Ein bekanntes Beispiel ist der Abstandssensor bei Autos, um Parkschäden oder Auffahrunfälle zu vermeiden. Oder auch die Gebäudeüberwachung profitiert durch solche Sensoren, da man mehrere Gebäude zentral Überwachen kann.

Bespielgeräte

- Smarte Parkhäuse
- SmartCars

1.4 Architektur

Man könnte ein IoT System in vier wichtige Gruppen unterteilen: [9]

- Dinge (things)
- das lokale Netzwerk
- das Internet
- Back-End Services (z.B. Cloud Services)

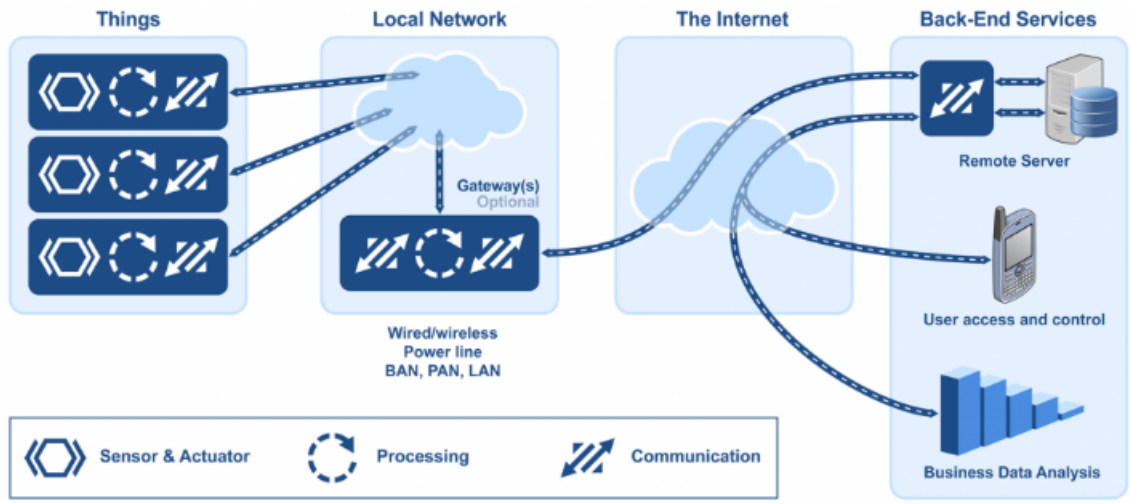

Abbildung 1.2: IoT Systemübersicht[7]

Grundsätzlich scheint die Architektur vertraut. Smart Objects kommunizieren über ein lokales Netzwerk mit Diensten im Internet.

Bisher konnten Geräte wie Laptops, PCs und Smartphones beinahe einheitlich mit dem Internet verbunden werden; entweder verkabelt über Ethernet oder drahtlos über ein lokales WLAN oder mobile Netze wie UMTS und LTE. Die Endgeräte verfügten jeweils über viel Rechenleistung, Speicher und ein leistungsfähiges Betriebssystem mit einem vollständig implementierten TCP/IP Stack.

In einem IoT System muss man von einer grossen Anzahl an Geräten mit Sensoren ausgehen. Diese Geräte verfügen meist über eine extrem niedrige Bandbreite, wenig Speicher und Rechenleistung [24].

Die Kommunikation erfolgt oft nicht vertikal der Architektur, sondern auch horizontal auf derselben Ebene. Sensordevices können beispielsweise miteinander kommunizieren oder Cloud Services Daten der Sensoren untereinander austauschen.

1.4.1 Wireless Sensor Netzwerke

Bei einer Vielzahl von verbundenen Sensoren, welche über einen grossen Bereich verstreut sind, bietet sich ein Wireless Sensor Network (WSN) an. In einem WSN werden die Sensoren nicht direkt mit dem Internet verbunden. Die Daten werden drahtlos von Teilnehmer zu Teilnehmer versendet. Muss ein Datenpaket in ein entferntes Netzwerk wie das Internet, so wird ein Gateway oder Edge Node benötigt.

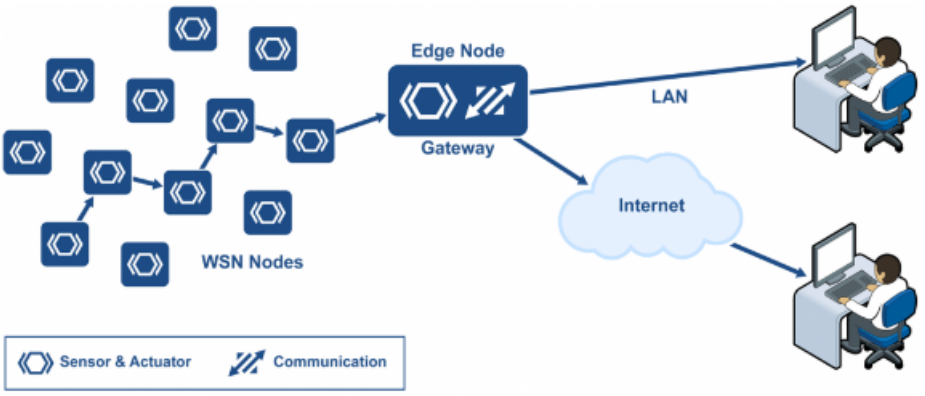

Abbildung 1.3: IoT WSN[8]

WSN Nodes sind typischerweise günstig im Einkauf. Sie können mit sehr wenig Leistung betrieben werden, dies ermöglicht den Batteriebetrieb. Durch diese Eigenschaften können WSN Nodes einfach, schnell und in sehr grosser Anzahl bereitgestellt werden.

1.5 Kommunikationsmodelle

Internet of Things verbindet Objekte aus der realen Welt miteinander. Um Objekte aus der Realität in die virtuelle Welt zu transformieren, werden Sensoren verwendet. Es gilt nun, diese Sensoren mit dem Internet zu verbinden.

Um unterschiedliche Bedürfnisse abzudecken, sind verschiedene Arten der Kommunikation entstanden. Die mit Sensoren ausgestatteten Geräte können sich in ihrer Weise, mit dem Internet zu kommunizieren stark unterscheiden.

1.5.1 Device-to-Device

Beim Device-to-Device Kommunikationsmodell kommunizieren mehrere Teilnehmer direkt miteinander (Peer-to-Peer). In diesem Szenario kommunizieren unterschiedliche Glühbirnen drahtlos mit einem Lichtschalter. Denkbar wären sämtliche Anwendungsgebiete aus dem „Smart Home“Bereich. Kommunikation mit dem Internet ist nicht zwingend notwendig. Eine grosse Herausforderung besteht darin, dass mehrere Teilnehmer unterschiedlicher Hersteller miteinander interagieren können. Dazu müssen die Teilnehmer denselben Protokoll-Stack implementieren.

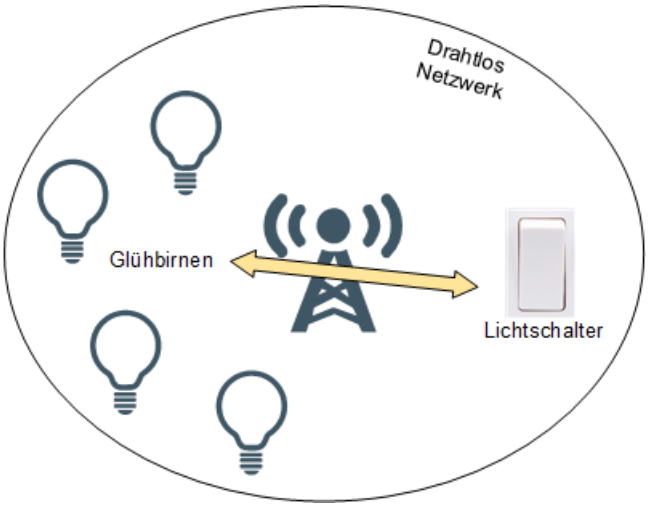

Abbildung 1.4: Device-to-Device Kommunikation

1.5.2 Device-to-Cloud

Die Gerätehersteller bieten für ihre End-User Cloud-Dienste im Internet an. Die Sensorgeräte kommunizieren direkt End-to-End über TCP/IP mit dem jeweiligen Cloud-Dienst. Die Benutzer können über eine Mobile App oder eine Webseite auf die jeweiligen Sensordaten zugreifen. Häufig wird aufgrund proprietärer Kommunikationsprotokolle ein Vendor-lock-in betrieben. Dies erschwert die Interoperabilität von Sensoren unterschiedlicher Hersteller [18].

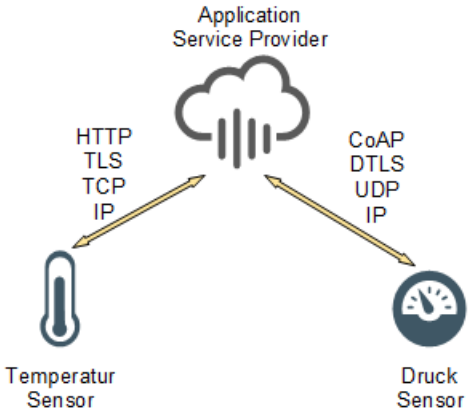

Abbildung 1.5: Device-to-Cloud Kommunikation

1.5.3 Device-to-Gateway

Anstatt einer Ende-zu-Ende Kommunikation zwischen Sensoren und Servern wird in diesem Modell ein Gateway zwischen diesen Komponenten eingesetzt. Sensoren kommunizieren somit nicht direkt mit einem Server. Auf diese Art und Weise können eine grosse Anzahl Sensoren mit Internetdiensten verbunden werden ohne dass die Sensoren selbst über einen direkten Internetzugriff verfügen. Der Gateway muss somit über eine Schnittstelle verfügen, damit Dienste im Internet indirekt mit den Sensoren kommunizieren können. Aus Sicht des Diensts ist es irrelevant, wie der Gateway mit den Sensoren kommuniziert.

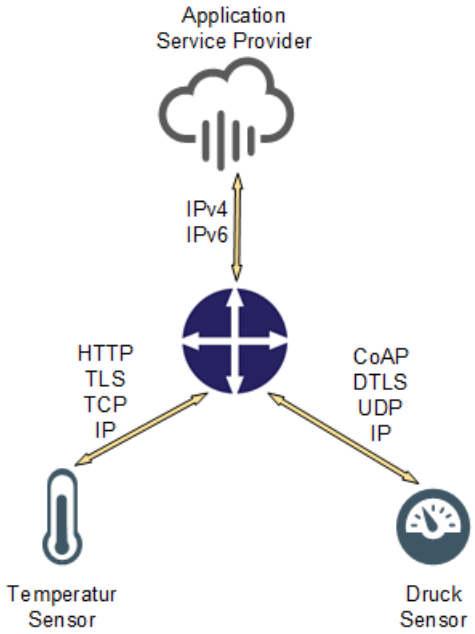

Abbildung 1.6: Device-to-Gateway Kommunikation

1.5.4 Back-End Data-Sharing

Sobald sich die Sensordaten auf einem Server befinden, können diese auf bekannte Weise anderen zur Verfügung gestellt werden. Beispielsweise könnte man den Zustand eines Sensors als JSON Objekt über eine REST-Schnittstelle abfragen. Ein weiterer Serviceprovider muss somit nicht mehr direkt mit den Sensoren kommunizieren. Da Sensordevices oft über limitierte Möglichkeiten verfügen, grössere Datenmengen bereitzustellen, verringert man mit dieser Art der Kommunikation die Anzahl Abfragen auf den Sensordevices.

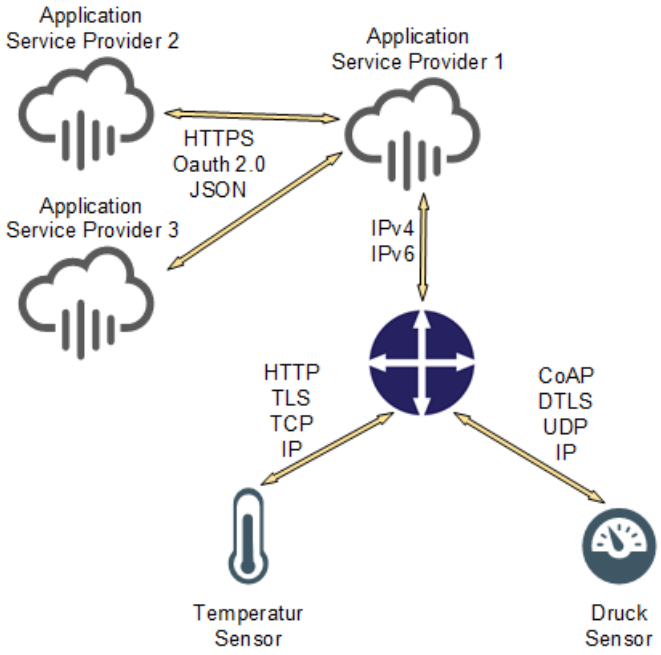

Abbildung 1.7: Backend-Data-Sharing Kommunikation

1.6 IoT Kommunikationsprotokolle

Seit der Entstehung des Internets werden für unterschiedliche Aufgaben Kommunikationsprotokolle entwickelt. Eine grosse Herausforderung war stets die Interoperabilität zwischen Geräten unterschiedlicher Hersteller. Standardisierungsgremien wie die International Standards Organization (ISO) und die Internet Engineering Task Force (IETF) haben in den vergangenen Jahrzehnten Richtlinien und Standardisierung von Kommunikationsprotokollen veröffentlicht.

Mit zunehmender Popularität des Internets der Dinge sind eine unüberschaubare Menge an proprietären und offenen Kommunikationsprotokollen entstanden. In der Geschichte des Internets hat sich gezeigt, dass sich langfristig nur offene Protokolle durchsetzen werden [12], proprietäre Protokolle hingegen werden aufgrund der fehlenden Interoperabilität niemals eine breite Verwendung finden.

1.6.1 Anforderungen

Bereits heute zeichnen sich die populärsten IoT Kommunikationsprotokolle ab. Um zu verstehen weshalb-, und vor allem in welchen Szenarien welches Protokoll eingesetzt wird respektive werden sollte, muss man sich mit den unterschiedlichen Anforderungen vertraut machen.

Die typischen bekannten Fragen nach der Verbreitung/Unterstützung und Skalierbarkeit stellen sich auch hier. Ebenfalls muss auf die mutmassliche Datenmenge geachtet werden. So dürfte eine vergleichsweise hohe Datenmenge für Geräte, welche über einen direkten, verkabelten Internetzugang verfügen, kein Problem darstellen, während Geräte in einem Mesh-betriebenen WSN wohl über deutlich weniger Bandbreite verfügen dürften.

Weitere Anforderungen wären Realtime Kommunikation, Stromverbrauch, Sicherheit und Network Address Translation (NAT) [13].

1.6.2 Request/Response

Request/Response ist das wohl bekannteste Pattern. Ein Client fordert mittels eines Requests eine Response von einem Service an. Der Service hört auf einkommende Requests, verarbeitet diese und antwortet (Response) den aufrufenden Clients. Request/Response Kommunikation skaliert schlecht, deshalb sollte beim Versenden einer Meldung an viele Teilnehmer auf ein anderes Pattern zurückgegriffen werden.

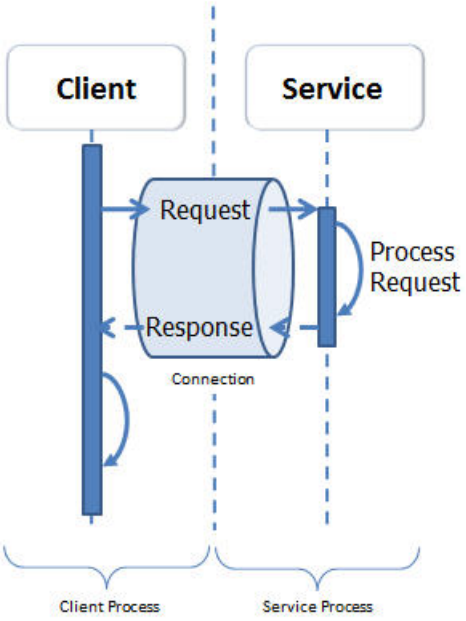

Abbildung 1.8: Request/Response Kommunikation [21]

HTTP In den 1990er Jahren wurde HTTP verwendet um statische HTML-Dokumente übers Internet abzurufen. Bis heute hat sich die grundsätzliche Funktion von HTTP nicht verändert. HTTP bietet über seine Methoden ein umfangreiches Interface für die Request/Response Kommunikation zwischen Clients und Servern über das Internet.

Aufgrund seiner hohen Verbreitung, Standardisierung und Unterstützung ist HTTP auch im IoT Umfeld beliebt. Für fast jede Programmiersprache und Laufzeitumgebung existieren Libraries, was das Entwickeln sehr angenehm macht. HTTP eignet sich jedoch nicht für alle Anwendungsfälle im IoT Bereich. Für jeden Request wird der gesamte HTTP (und darunterliegende) Header benötigt. Zusätzlich ist das Protokoll textbasiert, was mit dem zusätzlichen, grossen Overhead eine erhebliche Datenmenge bedeuten könnte. Für Endgeräte an Mobilen Netzwerken könnte dies ungeeignet sein [13].

In der Version 1, respektive 1.1 gibt es mit HTTP keine Möglichkeit, echte Push-Meldungen zu versenden. Bei Push-Meldungen sendet der Server eine Response (besser: Nachricht) an den Client ohne vorgängigen Request. In der Version 2 von HTTP sind echte Push-Meldungen vorgesehen, jedoch gibt es wenige Implementation und Erfahrungswerte damit.

CoAP Das Constrained Application Protocol (CoAP) implementiert wie HTTP das Request/Response Pattern [13]. Mit CoAP existiert ein massgeschneidertes IoT-Protokoll, welches nach dem REST Paradigma konzipiert wurde. HTTP ist schwergewichtig, hat einen grossen Overhead und generiert damit hohe Datenmengen. Ausserdem ist vor jeder Session den für TCP benötigten 3-Way Handshake nötig.

CoAP wurde entwickelt, um diesen Schwächen von HTTP entgegenzuwirken. Bei sogenannten Low-Power and Lossy Networks (LLN's) sind die Nodes im Vergleich zu herkömmlichen Computersystemen sehr eingeschränkt, was die Verwendung von HTTP schwierig gestaltet. CoAP bietet grundsätzlich folgende Features:[5]

- Request/Response Kommunikation zwischen Endpoints
- Discovery von Services und Ressourcen
- URI's und Media Types
- Kompatibilität mit HTTP

- Multicast Support
- sehr kleiner Overhead (Header von 4 Byte)
- implementiert das Observer Design Pattern
- UDP als Transportprotokoll
- Asynchroner Nachrichtenaustausch

CoAP kann Peer-to-Peer zwischen Devices eingesetzt werden, aber auch zwischen Device und Service oder zwischen Device und einem Proxy.

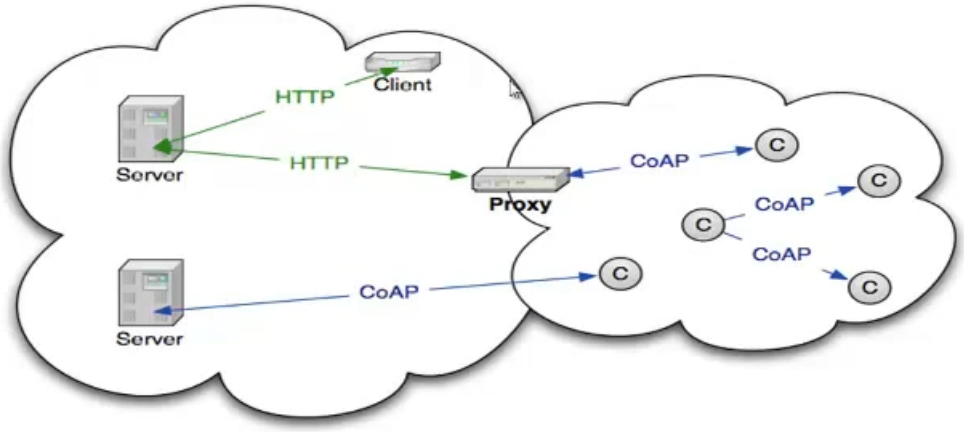

Abbildung 1.9: CoAP Architektur [22]

Durch die massgeschneiderten Features für IoT wird CoAP hauptsächlich in WSN's eingesetzt [13].

1.6.3 Publish/Subscribe

Beim Publish/Subscriber Pattern gibt es einen Sender (Publisher) und einen Empfänger (Subscriber). Der Empfänger hört auf gewisse Themen (Topics). Dies kann nur ein Thema sein oder auch viele Verschiedene. Der Sender kategorisiert seine Nachrichten in Themen und sendet diese zu den jeweiligen Empfänger. Der Sender und der Empfänger wissen aber nichts voneinander. Sie senden oder hören nur im Netzwerk, ob eine für sie interessante Nachricht angekommen ist. Durch die einfache Verknüpfung von Publisher und Subscriber, eignet sich dieses Verfahren sehr gut im IoT-Bereich. Die Sensoren sind die Publisher, sie liefern zum Beispiel Temperaturdaten in die richtige Kategorie. Alle Server/Clouddienste, welche sich für Temperaturdaten interessieren, können auf dieses Kategorie hören. Durch die einfache Handhabung, skaliert dieses Pattern sehr gut.

In der folgenden Grafik sieht man das Publish and Subscribe Pattern. Der Publisher hat eine "Address Changed" Message in den Channel geschickt. Nun erhalten alle Subscriber, welche dem Channel folgen, diese Nachricht und verarbeiten sie.

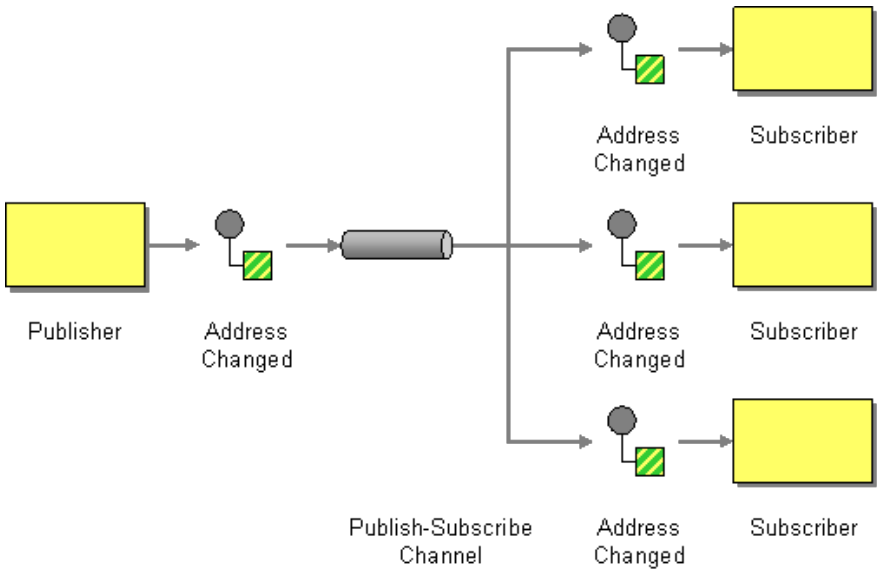

Abbildung 1.10: Publish and Subscribe Pattern[4]

MQTT MQTT (Message Queue Telemetry Transport) ist ein von IBM entwickeltes Protokoll. Es ist ein speziell für IoT entwickeltes Protokoll um Machine-to-Machine Kommunikation herzustellen. Bei dem Protokoll wurde speziell auch ein schlankes Design geachtet. So ist die kleinst Mögliche Nachricht 2 Byte gross.

Der Hauptverwendungszweck von MQTT ist vorallem der Austausch von Daten zwischen Geräten und Server (D2S).[19] Da das Protokoll für die D2D Konnektivität entwickelt wurde, wird es auch in diesem Bereich eingesetzt. Das heisst die Vorteile liegen in Netzen mit vielen kleinen Geräten, welche wenig Kommunizieren und sich als Kollektion sehen.[20]

Bei MQTT wird ein Broker verwendet. Der Sensor ”published” seine Daten mit einem Topic und den Daten an den Broker. Dieser sendet die Nachricht an alle, welche sich auf das gewünschte Topic Subscribed haben. Durch das schlanke Design, gibt es natürlich auch Nachteile. Aber bei kleinen und einfachen Systemen ist MQTT ein beliebtes Protokoll.

AMQP AMQP (Advanced Message Queuing Protocol) ist ein bekanntes und viel eingesetztes Protokoll. Das binäre Netzwerkprotokoll wird von vielen grossen Firmen[20] entwickelt, wie zum Beispiel Microsoft oder auch Cisco. Momentan ist die Version 1.0 seit 2010 als aktuellen Standard im Einsatz.

AMQP wird durch sein Queuing Design im Server zu Server Bereich eingesetzt (S2S).[19] Daher ist es in Bereichen einzusetzen, in der die Geschwindigkeit und der Prozessor nicht relevant sind. Zusätzlich wird es in Bereichen eingesetzt, in dem eine Nachricht nur von A nach B gesendet werden soll und man keine Nachricht verlieren möchte.

Die Sensoren senden die Nachrichten an einen Message Broker, welcher die Nachricht in die richtige Queue schiebt. Nun können sich die Dienste an der Queue anmelden und die Nachrichten konsumieren. Dabei gibt es die Möglichkeit Topics zu setzen, um Kategorien einzuführen. Es können jeder Nachricht auch noch Attribute hinzugefügt werden, wie zum Beispiel Name oder Durability. Durch den grossen Funktionsumfang des Protokolls ist es natürlich auch schwieriger einzurichten und die minimale Paketgrösse wächst damit auch. Die kleinstmögliche Paketgrösse ist 60 Byte.

XMPP XMPP (Extensible Messaging and Presence Protocol) wurde speziell für das Internet der Dinge erweitert, um den Anforderungen gerecht zu werden. XMPP gibt es schon seit mehreren Jahren und wurde in vielen Chatprogrammen eingesetzt. Auch heute findet man das Protokoll beim Facebook-Messenger wieder. Mit der IoT-Erweiterung/Anpassung will man nun nicht mehr Menschen miteinander verknüpfen, sondern Dinge.

XMPP ist das beste Protokoll um Geräte mit Menschen zu verbinden. Dies ist eine Spezialform des Geräte zu Server Pattern (D2S). [19] XMPP sollte dann verwendet werden, wenn die Geschwindigkeit und der Prozessor nicht wichtig sind, wenn das Gerät immer verbunden sein soll und wenn nur wenige Konnektivitätspunke in einem grossen Bereich vorhanden sind.[20]

Bei XMPP wird kein Broker, sondern ein zentraler Server verwendet. Dieser soll allerlei verschiedene Geräte miteinander verbinden. Dieses Protokoll wird häufig für das Remote Management von Konsumergeräten verwendet.

2. IoT Device Management

In Zukunft ist eine stark ansteigende Anzahl an IoT Devices zu erwarten. Laut der International Data Corporation (IDC) dürften im Jahre 2020 in etwa 30 Milliarden Devices weltweit verbunden sein [1]. Unternehmen könnten potenziell mehrere Tausend Sensoren für ihre Zwecke einsetzen. Bereits bei herkömmlichen Computersystemen und Servern stellt das Management eine grosse Herausforderung dar. IoT Devices dürften potenziell in einer sehr viel grösseren Anzahl verbreitet sein als herkömmliche Geräte. Herausforderungen wie die Heterogenität, Verteilung und Security verschärfen sich mit der stetig wachsenden Anzahl an Geräten.

IoT Devices durchleben in ihrem Lebenszyklus verschiedene Stadien. Ein Device Management Tool soll die Administration in jeder dieser Phasen Unterstützen.

2.1 Device Lifecycle

Der Lebenszyklus eines IoT Devices könnte beispielsweise aus folgenden fünf Phasen bestehen.

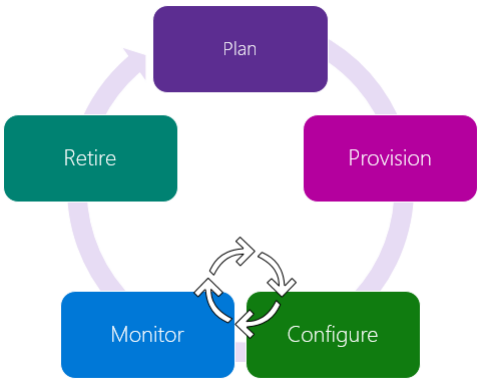

Abbildung 2.1: IoT Management Azure [10]

Plan In der Planungsphase möchte man aufgrund von vorliegenden Daten eine Veränderung am System vornehmen. Um fundierte Entscheide in der Planungsphase zu ermöglichen werden Daten und Messwerte von Devices benötigt. Je einfacher diese Daten zugänglich-, respektive abfragbar sind, desto qualitativ hochwertiger und exakter kann geplant werden.

Provision Neue Geräte müssen vor der produktiven Nutzung bereitgestellt werden. Dieser Prozess kann mehrere Personen und Aufgaben involvieren. Grundsätzlich werden neue Geräte in ein bestehendes System eingebunden oder ein komplett neues System aufgebaut.

Configure Damit ein Device in den vorgesehenen Zustand versetzt werden kann benötigt es eine Konfiguration. Bei einer grossen Anzahl Devices empfiehlt es sich diesen Prozess bestmöglichst zu automatisieren. Dazu müssen alle Beteiligten

Monitor In der Monitoringphase sollen die Zustände der Devices überwacht werden. Ziel ist es, die Funktionalität und Korrektheit des angestrebten Verhalten sicherzustellen. Dies wird mittels periodischer Abfragen oder Observations sichergestellt.

Retire Am Ende des Lebenszyklus sollen Geräte geordnet aus dem System entfernt werden. Dabei gilt es den Prozess bestmöglich zu automatisieren und allfällige Vorschriften betreffend Datensicherheit und Datenschutz zu beachten. Andere Systeme wie das Inventar könnten ebenfalls in diesem Prozess beteiligt sein.

2.2 Device Management Aufgaben

2.2.1 Provisionierung und Authentisierung

Bevor neue Devices produktiv genutzt werden können, müssen sie entweder manuell, oder automatisch provisioniert werden. Beim Device Provisioning muss ein Gerät in einen Zustand gebracht werden, in dem es für eine oder mehrere vorgesehene Personen erreichbar respektive verfügbar ist.

Notwendige Schritte So unterschiedlich die Geräte im Netzwerk und Internet auch sein können, so haben sie abstrakt gesehen die selben Schritte zu durchlaufen um für die produktive Nutzung Verfügbar zu werden:

- ein Device muss physisch am vorgesehenen Ort platziert werden,
- die Stromversorgung (Stromnetz, Batterie, Akku) für das Gerät muss sichergestellt werden
- das Device muss gestartet werden
- eine Initialkonfiguration muss geladen werden
- die Netzwerk- respektive Internetverbindung muss hergestellt werden

Ab dem Zeitpunkt der Erreichbarkeit des Netzwerks unterscheiden sich manuelles und automatisches Provisioning. Beim manuellen Provisioning greift der User oder Administrator entweder direkt über ein grafisches User Interface auf das Gerät zu und nimmt weitere Einstellungen vor oder er verbindet sich über das Netzwerk auf eine Benutzeroberfläche des Geräts. Beim automatischen Provisioning meldet sich das Device über das Netzwerk an einem Controller respektive einer Management Instanz.

Authentisierung Bei einem Eintritt in ein bestehendes System, in diesem Falle die Bereitstellung eines neuen Devices, muss sich das Gerät am System authentisieren. Aus Sicherheitsgründen möchte man sicherstellen, dass die Zugriffe in das System kontrolliert werden. Es gibt verschiedene Möglichkeiten der Authentisierung:

- Authentisierung durch Wissen (z.B. Passwort, PIN, Challenge)
- Authentisierung durch Besitz (z.B. RFID-Chip, Zertifikat, One Time Pin)
- Authentisierung durch biometrische Merkmale (z.B. Fingerprint, Handvene)

Nicht alle Verfahren eignen sich für die Authentisierung von Devices. Ein Gerät selbst verfügt über keine biometrischen Merkmale, höchstens der Benutzer, welcher versucht das Gerät bereitzustellen. Ein Shared Secret in Form eines Passworts oder eine Chain-of-Trust mit Zertifikaten eignen sich für die Device Authentisierung gut.

IoT Provisioning IoT Devices sollten vor der produktiven Nutzung in einem definierten Prozess bereitgestellt werden. IoT Devices sind häufig nicht in einer standardisierten Büroumgebung wie zum Beispiel

Notebooks und Drucker. Mögliche Standorte sind Lagerhallen, Produktionsstätten oder auch andere Orte im Freien. Oft ist es nicht möglich, solche Geräte an das Stromnetz anzuschliessen, deshalb werden diese über Batterien oder einen Akku gespiesen. Die Netzwerk- und Internetverbindung wird über ein Wireless-Mesh-Netzwerk realisiert. In einem ersten Schritt muss ein Device mit einem Netzwerk verbunden werden. Es ist weitgehend bekannt, dass eine grosse Vielfalt an Devices existieren. Manche von ihnen enthalten ein einfaches User Interface, andere nicht. So unterscheidet sich das Ausrollen von IoT Geräten von herkömmlichen PC's und Laptops indem man nicht direkt über das geräteeigene UI Einstellungen vornehmen kann. Ein Management von IoT Devices muss also unterschiedliche Anforderungen abdecken. Es könnten sowohl Geräte im herkömmlichen Sinne über eine direkte Internetverbindungen provisioniert werden, als auch neuartige Geräte, welche an ein Low-Power & Lossy Network (LLN) angeschlossen sind.

Ein neues IoT Device im System kontaktiert entweder seinen lokalen Gateway oder direkt einen Server über TCP/IP. Das System kann nun das neue Device aufnehmen und weitere Aufgaben wie beispielsweise das Laden der Konfiguration oder das Speichern von Geräteinformationen veranlassen. Netzwerke mit IoT Devices sind sehr dynamisch. Es werden häufig neue Teilnehmer aufgenommen oder alte Teilnehmer entfernt. Tools, welche diese Prozesse automatisieren sind stark gefragt. Devices sollen sich selbständig an Netzwerken anmelden können und sich bei zuständigen Servern für weitere Anweisungen registrieren. Solche Discovery Mechanismen sind für zukünftige Anwendungen essentiell [2].

Mit dem Prozess der automatischen Registrierung an Netzwerken und Systemen sind Fragen der Sicherheit verbunden. Man möchte selbstverständlich die Zugriffe in Netzwerke und Systeme korrekt authentisieren und autorisieren. Sicheres Bootstrapping von IoT Devices umfasst neben Authentisierung und Autorisierung auch die Übertragung von Security Parametern für die Ausführung von vertrauenswürdigen Operationen [3].

2.2.2 Konfiguration

Konfigurationsmanagement Konfigurationsmanagement beschäftigt sich mit der Erstellung, Verteilung, Versionierung und Sicherung der Konfigurationen von unterschiedlichen Devices. Durch eine Konfiguration wird das Verhalten eines Geräts bestimmt. Die grösste Schwierigkeit bei den Konfigurationen liegt in der Heterogenität. Unterschiedliche Hersteller verwenden für ihre Geräte unterschiedliche Konfigurationen. Selbst Devices derselben Hersteller benötigen oft eine andere Konfiguration. Im Kern lassen sich diese Tatsachen nicht ändern, man kann lediglich versuchen eine möglichst generische Basiskonfiguration für eine Gruppe von Devices bereitzustellen, um die manuellen Tätigkeiten zu minimieren.

Automatische Konfiguration Eine manuelle Konfiguration von sämtlichen IoT Devices scheint nicht zeitgemäss. So müsste vor jeder Auslieferung das Gerät mit allen Besonderheiten vorinstalliert werden. Stattdessen sollte eine möglichst generische Konfiguration automatisch geladen werden (Zero Touch Provisioning). Bei Bedarf könnte die Konfiguration remote angepasst werden [25]. Auf diese Weise würden beispielsweise in einem WSN von IoT Devices sämtliche Sensorgeräte initial mit derselben Konfiguration gestartet werden. Möchte man punktuell eine Veränderung an einem Gerät vornehmen, so wäre dies Remote möglich, da mit der generischen Initialkonfiguration Connectivity und Managability sichergestellt würden.

Softwareverteilung Nebst dem erstmaligen Laden und Anpassen der Konfigurationen von Devices müssen auch deren Soft-, respektive Firmware verwaltet werden. Bei notwendigen Patches und Upgrades sollten diese von Remoteseite aus provisioniert werden können. So können bei bekannt werdenden Sicherheitslücken eine grosse Anzahl Geräte mit Updates versorgt werden. Die Schwierigkeit liegt darin, die möglicherweise grosse Datenmenge der Soft- oder Firmwares auf Geräte in LLN's zu übertragen.

Backup und Restore Backup und Restore der Devicekonfigurationen sind wichtige Aufgaben im Device Management. Die Sicherungsvorgänge müssen entweder automatisch über ein Scheduling, oder manuell von einer Person vorgenommen werden können. Eine manuelle Sicherung ist vor allem bei einer einmaligen Anpassung an der Konfiguration notwendig.

Die Aufgaben des Konfigurationsmanagements sind immens wichtig. Bei fehlerhaften Konfigurationen oder Ausfällen respektive Defekten von Devices drohen kostspielige Downtimes. Es wäre gar möglich, dass eine ganze Produktionskette für längere Zeit ausfällt. Weitgehend automatisierte Prozesse bei Konfigurationen, sei es das Laden oder das Sichern, stellen einen enormen Mehrwert für den Betrieb von IoT Devices dar.

2.2.3 Monitoring und Diagnose

Was ist Monitoring? Bei all den Millionen Devices ist ein Monitoring unerlässlich. Ohne ein ausgeklügeltes Monitoring kann die Überwachung von so vielen Devices sehr schnell chaotisch enden. Daher muss das Monitoring gut durchdacht sein, um sein Ziel nicht zu verfehlen. Doch was ist unser Ziel mit dem Monitoring?

Das Hauptziel des Monitorings ist das proaktive Überwachen der Geräte. Dadurch verringert sich nicht nur die Zeit, die benötigt wird um den Fehler zu erkennen, sondern auch die benötigte Reparaturzeit. So kann die Uptime jedes Sensors möglichst gross gehalten werden und die Produktivität steigt. Ein weiteres Ziel ist das Erkennen von Muster. So können all die gesammelten Daten zusammengefügt werden und die Muster analysiert werden. Dies führt zu einer besseren Früherkennung und auch zu einem Know-How-Gewinn. So können zukünftige Probleme besser erkannt und schneller behoben oder sogar vermieden werden.[15]

Nun gibt es aber neue Hindernisse bei der Überwachung von IoT-Sensoren. Nicht nur, dass es eine grössere Anzahl zu überwachende Geräte gibt, sondern auch immer neue Protokolle, Geräte die sich verschieben (SmartCars) oder auch Probleme durch den noch eher jungen Entwicklungsstand gewisser Geräte. Um ein vernünftiges Monitoring im Bereich IoT bereitzustellen, muss man daher viel bedenken, was beim normalen Server oder Netzwerkmonitoring nicht wichtig war.

Monitoring im IoT-Bereich Im Bereich IoT gibt es spezielle Anforderungen an das Monitoring. Durch die vielen Geräte und die dynamischen Netzwerke muss das Monitoring sehr Flexibel sein. Täglich werden neue Geräte eingeführt und alte entsorgt. Nicht nur der Austausch von Geräten ist mühselig, sondern auch der Standortwechsel. Die Geräte bewegen sich zum Teil und sind so in verschiedenen Netzwerken. Wenn hier kein sinnvolles System eingesetzt wird, häufen sich die fehlerhaften Meldungen und das Monitor wird ineffizient.

Ein weiterer wichtiger Punkt bei IoT-Geräten, ist die Team Kollaboration. Ein einzelner Mitarbeiter hat nicht die Kapazität, alleine Millionen von Sensoren zu Überwachen und zu Warten. Zusätzlich kommen noch weitere Komponenten wie Gateways, Netzwerke oder Clouddienste dazu. Um allen Bereichen ein praktikables Monitoring bereit zu stellen, benötigt man viel Know-how und ein guter Umgang mit den erhaltenen Daten.

Bei all den verschiedenen Geräten, stellt sich die Frage. Wie strikt muss die Überwachung eigentlich sein? Hier kann man das Ganze in zwei Bereiche unterteilen. Real-Time Monitoring und Non-Real-Time Monitoring. Die meisten Sensoren benötigen ganz klar kein Real-Time Monitoring. Die Sensoren wachen alle paar Minuten auf, messen die gewünschten Werte, senden diese an einen Gateway und legen sich wieder schlafen. Hier wäre ein Real-Time Monitoring nicht das richtige, da es zu vielen Fehlalarmen kommt.

Bei anderen Sensoren kann dies aber sehr wohl ein wichtig sein. Bedenkt man ein Katastrophen-Überwachungssystem, dass nicht ausfallen darf, muss man ein Real-Time Monitoring einrichten. Daher

ist es sehr wichtig, dies richtig abzuschätzen, um den grössten Nutzen herauszuholen.

Schlussendlich unterscheidet sich auch die Art der Überwachung. IoT-Monitoring ist nicht mit einem Servermonitoring gleichzustellen. Bei Serverfarmen ist die Auslastung, der verfügbare Speicher oder auch die Verfügbarkeit wichtig. Bei Sensoren sieht dies anders aus. Da die Geräte sowieso keine grosse Last bewältigen müssen, oder der Speicher nicht relevant ist, muss das Monitoring auf andere Faktoren angepasst werden.

Wichtig ist daher die Verfügbarkeit der Sensoren. Spricht der Sensor immer wieder mit mir oder ist er nicht mehr Erreichbar. Auch die gewonnen Daten können wichtig sein. Falls zum Beispiel bei Temperaturmessungen arktische Kälte anstelle von sommerlichen Temperaturen gemessen werden, ist das sicher auch ein Indiz für Fehlverhalten.[6]

2.2.4 Maintenance und Update

Was versteht man unter Maintenance und Update Unter Maintenance versteht man die Wartung und das dazugehörige updaten der Geräte. Bei einer kleinen Anzahl von Geräten geht das vielleicht noch gut mit einer Excel-Liste, aber bei Millionen von Geräten wird es schnell unübersichtlich. Daher wird ein Maintanence Management benötigt.

Ein Bestandteil des Maintenance Management ist das Asset-Management. Hier werden alle Gerätedaten, Hanbücher, Garantien und andere relevanten Daten zum Gerät gespeichert.[23] So hat man eine zentrale Verwaltung aller wichtigen Daten und kann effizienter arbeiten.

Alle heutigen Geräte werden mit einem Softwarestand ausgeliefert, bei denen noch Fehler vorhanden sein können. Oder auch die Technik wird ständig weiterentwickelt. Daher veröffentlicht der Gerätehersteller ständig neue Updates und Patches, um neue Funktionen und Sicherheitsupdates einzuspielen. Ohne ein Update-Management wäre diese Aufgabe für so viele Geräte unmöglich. Daher ist eine zentrale Verwaltung unbedingt nötig.

Maintenance und Update im IoT-Bereich Da es bei IoT-Geräten häufig um Sensoren handelt, welche auch an aussenbereichen platziert werden können, ist ein Wartungsmanagement sehr wichtig. Die Sensoren sind durch die Umwelt härteren Bedingungen ausgesetzt als ein normaler Computer oder Netzwerkgeräte.

Ohne ein ausgereiftes Management, kann diese Menge aber nicht gewartet werden. So soll das Management eine Empfehlung geben, wann die Batterie auszutauschen ist, oder wann der Sensor nicht mehr die gewünschte Genauigkeit liefert. So weiss der Techniker genau, wann welche Sensoren/Geräte ausgetauscht werden müssen.

Bei allfälligen Problemen mit einem IoT-Gerät gibt es ein grosses Problem. Das Gerät kann nicht kurz vom Strom genommen und neu gestartet werden. Der Sensor befindet sich vielleicht 50 Kilometer entfernt an einem schwer erreichbaren Ort. Dadurch muss die Managementumgebung die Wartungspläne auch an dieses Problem anpassen.

IoT Geräte sind Jahre, wen nicht sogar Jahrzehnte lang im Einsatz. Doch Hersteller können Konkurs gehen oder pflegen ein Produkt nicht weiter. Nun kann es zum Problem kommen, wenn ein Sensor keine Updates mehr erhält und schwerwiegende Sicherheitslücken vorhanden sind. Vor solchen Szenarien sollte man gewarnt werden, nicht das ein Sensor noch zum Einstiegspunkt für Hacker wird.

Durch die vielen Software Updates bei der hohen Anzahl an Geräten, ist man ohne automatische Software Verteilung und Update Installation verloren. Von Hand kann diese hohe Anzahl an Geräten gar nicht mehr bewältigt werden. So gibt man pro Sensor den gewünschten Softwarestand frei und die Managementumgebung spielt diese automatisch auf die jeweiligen Sensoren. Bei Problemen wird der Administrator

benachrichtigt und kann eingreifen.

Updates sind im IoT-Bereich wichtig, da der Markt von Geräten regelrecht geflutet wird. Diese sind bei der Lieferung meist noch gar nicht ausgereift und können zum Sicherheitsproblem für Firmen werden. Wenn man nun nicht eine effiziente Updateumgebung besitzt, kann dies schnell zu einem grossen Problem werden.

Bei der vielen Anzahl von Updates, darf man das Netzwerk nicht vernachlässigen. Bedenkt man, das die Netzwerke speziell für Low-Power ausgelegt sind, muss ein ausgeklügelter Updatemechanismus entwickelt werden. Was passiert wen das Update nicht Vollständig zum Gerät gelangt? Oder muss jedes Update an die einzelnen Sensoren geschickt werden, oder kann man ein Update von einem Knotenpunkt aus verteilen. Bei so vielen Geräten kann ein solches Netz schnell mal in die Knie gehen, wenn man allen Geräten gleichzeitig ein Update schickt.

2.2.5 Security Management

Was bedeutet Security Management? Wenn man die Definition nach FCAPS nimmt, geht es bei dem Security Management zum Beispiel um Access Logs, Security Alarm/Event Reporting, Data Privacy, und Selective Resource Access. Wen man diese Punkte abgedeckt hat, hat man schon viel umgesetzt und ein relativ gutes Security Management. Hier sind nun die einzelnen Bereiche genauer erklärt.

Access Logs sind eine hilfreiche Massnahme um bei bei Angriffen zu sehen, wer auf das System zugegriffen hat. Mehr Sicherheit schafft man mit Access Logs zwar nicht, aber dafür ist man in der Forensikanalyse dankbar um diese Informationen. Daher sollte man diese Information unbedingt in das Management einbauen.

Bei einem Angriff hilft der Security Alarm und das Event Reporting. Um nicht alles von Hand überprüfen zu müssen, gibt es Warnsysteme, die bei einem Angriff sofort eine Meldung ausgeben. Hat man, wie im IoT Bereich, mehrere Tausend oder gar Millionen von Geräten, so ist man auf solche Systeme angewiesen.

Nicht nur all die Systeme müssen den Datenschutz beachten, sondern auch die Managementumgebung. Es nützt alles nichts, wenn man den Verkehr zwischen den Geräten absichert, aber danach alle Daten unverschlüsselt im Management vorliegen. Daher muss das Security Management im Bereich Datenschutz nicht nur die Geräte, sondern auch sich selbst absichern.

Die Access Rights gehören zu einem wichtigen Bestandteil des Security Managements. Bei jedem Gerät muss man genau definieren können, wer auf was Zugriff hat. Dies muss bei vielen Geräten möglichst vereinfacht werden. Man stelle sich vor, man müsste bei all seinen Sensoren jeden Einzeln konfigurieren. Dies wäre eine unmögliche Aufgabe.

Security Management im Bereich IoT Sicherheit ist im IoT-Bereich in den letzten Jahren immer wichtiger geworden. Viele Hersteller wollen das Gerät so schnell wie möglich auf den Markt bringen und achten nicht auf die Sicherheit. Laut einer Studie von HP gab es bei den meisten Produkten schwerwiegende Sicherheitsbedenken.[14] Doch das Security Management kann hier nur Ansatzweise helfen.

Viele Geräte unterstützen gar keine grosse Sicherheitsfeatures, welche man in einem Management verwalten könnte. Doch man sollte möglichst viele Faktoren in das Security Management einbauen. Wie stellt man nun sicher, dass nur die Berechtigten Personen die Daten bekommen. Wie kann man bei all den Geräten sicher sein, dass keine Daten sonst wo abfliessen und in Fremde Hände geraten. Bei all den verschiedenen Hersteller wird es sicher keinen Standard geben, der für alle Geräte angewendet werden kann.

Wie schon mehrmals erwähnt, sind IoT-Geräte nicht gerade Hochleistungscomputer. Daher wird es schwerer grosse Schlüssel auszutauschen, da diese doch eine gewisse Rechenpower benötigen oder eine lange

Zeit für die Berechnung benötigen. So ist ein grosser Schlüsselaustausch nicht das Optimale Verfahren um die Geräte abzusichern. Doch man möchte ja trotzdem eine sichere Übertragungsmethode haben.

Die Vielfalt an Geräten bringt auch eine grosse Angriffsfläche mit sich. Man ist an mehreren Punkten angreifbar und kann nicht alles abdecken. Was macht man nun, wen ein Gerät oder sogar ein gesamtes Netzwerk von Sensoren übernommen wird. Da man meist nicht Vorort sein kann, muss daher für dieses Szenario eine Lösung vorhanden sein.

Eine Möglichkeit von Securitymanagement sind die Black- und Whitelists. Doch bei einer solch grosser Anzahl Geräten kann dies schnell unübersichtlich werden. Doch das Problem gehört nicht nur in das Management, sondern kann auch auf das Netzwerk erweitert werden. All die Geräte wollen ja auch durch die Firewall und diese Regeln müssen dynamisch mit dem Management zusammen angepasst werden. Manuell ist dies gar nicht mehr möglich.

Seit längerer Zeit ist es klar das IoT-Botnetze gerne für DDoS-Angriffe genutzt werden. So können mehrere tausend kleinere Geräte eine grosse Last erzeugen und andere Webdienste attackieren. Vor diesen Botnetzen muss man sich absichern, damit nicht man nicht all seine Sensoren und Geräte gekapert werden.

3. Requirements

3.1 Allgemeine Beschreibung

3.1.1 Produktperspektive

Mit Internet of Things sind eine Vielzahl neuartige Devices entstanden. Während in herkömmlichen Netzwerken hauptsächlich Personal Computer, Notebooks, Server usw. verwaltet werden mussten, so bringen IoT Devices den IT-Abteilungen neue Herausforderungen. Zum einen dürfte die Anzahl Geräte gegenüber herkömmlichen Computer deutlich ansteigen, zum anderen sind IoT Devices in Sachen Funktionalität und Rechenleistung, sowie auch der Netzwerkbandbreite deutlich beschränkt.

Mit <insert Application Name here> soll eine Management Applikation bereitgestellt werden, um eine Vielzahl unterschiedlicher IoT Devices administrieren zu können.

3.1.2 Produkfunktionen

<insert Application Name here> soll den Benutzern erlauben, IoT Geräte zu verwalten. Die Aufgaben reichen vom Erfassen und Discovery von Devices über die Konfigurationsverwaltung und Softwareverteilung bis zu Backup und Restore. Ausserdem sollen Management-relevante Kommandos auf Devices ausgeführt- und Security Aspekte beachtet werden. Die Details zu den Produktfunktionen sind den Use Cases zu entnehmen.

3.1.3 Benutzer Charakteristik

Zielpersonen der Applikation sind Betreiber von IoT Devices. Dies können im Enterprise Umfeld IT-Mitarbeiter in operationeller Funktion-, oder auch Softwareentwickler für IoT Applikationen sein. Heimanwender können bei entsprechenden Kenntnissen ebenfalls zur Zielgruppe gehören. Es werden solide Grundkenntnisse in TCP/IP Netzwerken sowie Verständnis der verwendeten IoT Architekturen und Devices vorausgesetzt.

3.1.4 Einschränkungen

Eventuelle Einschränkungen werden in der Designphase noch genauer spezifiziert. Momentan gibt es noch keine spezifische Einschränkungen

3.2 Use Cases

3.2.1 Use Cases Diagramm

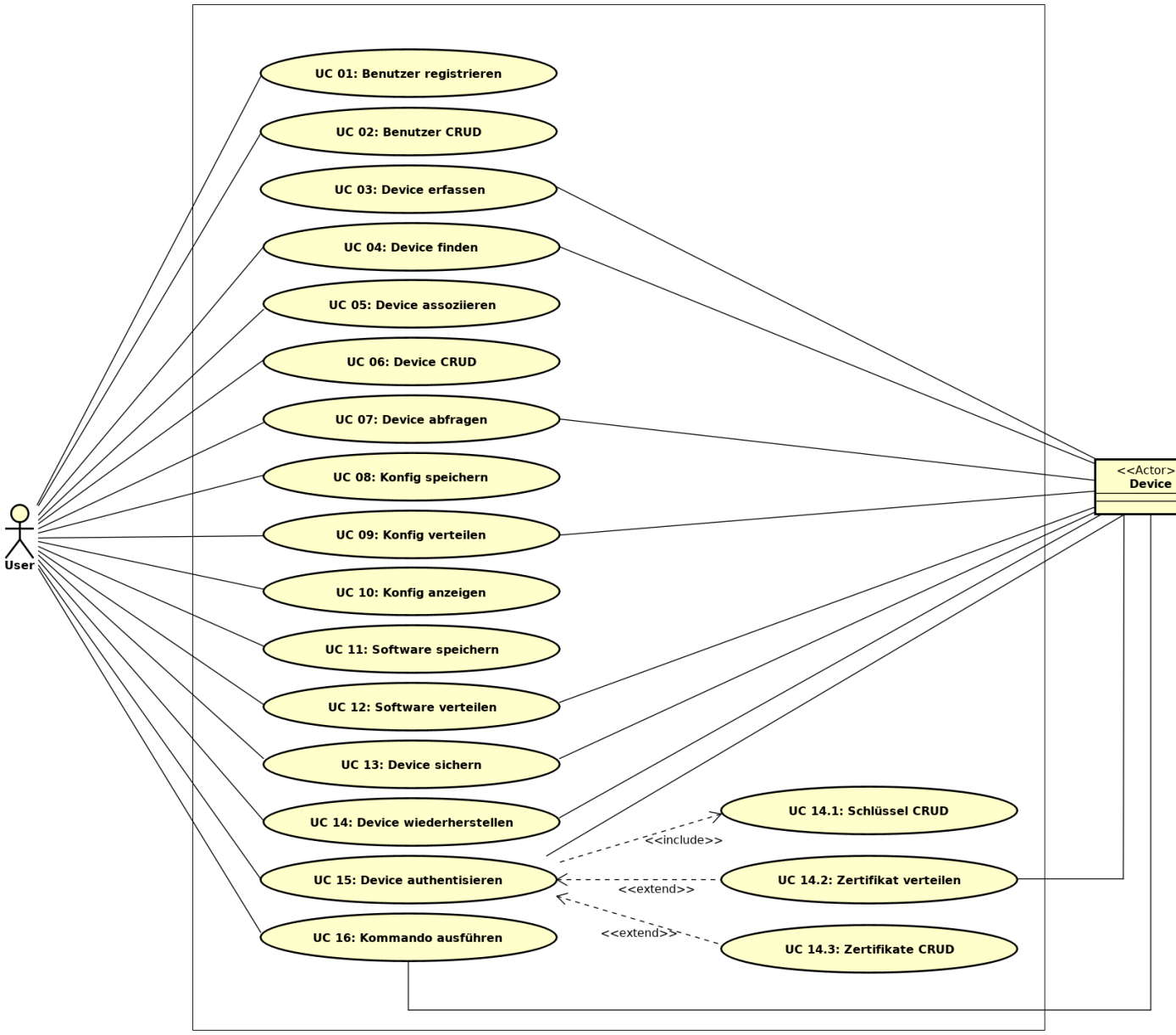

Abbildung 3.1: Use Case Diagramm

3.2.2 Aktoren

Der Benutzer der Applikation ist in diesem System der einzige primäre Aktor. Dieser bewirtschaftet die Applikation und verwaltet alle Devices und Benutzer. Als Sekundärer Aktor werden die einzelnen Devices gezählt.

3.2.3 Beschreibungen (Casual)

Benutzer registrieren

ID	01
Name	Benutzer registrieren
Beschreibung	Der Benutzer legt sich einen neuen Account für die Management Software an.
Preconditions	• Applikation gestartet
Postconditions	• Neuer Benutzer ist gespeichert
Main Success Scenario	1. Benutzer wählt ”Benutzer registrieren” 2. Benutzer gibt Loginangaben ein 3. Benutzer speichert die Eingaben
Extensions	-

Benutzer CRUD

ID	02
Name	Benutzer CRUD
Beschreibung	Benutzerverwaltung der Applikation
Preconditions	• Applikation gestartet • Benutzer ist registriert • Benutzer ist eingeloggt
Postconditions	• Änderungen gespeichert
Main Success Scenario	**Create:** 1. UC 01: Benutzer registrieren **Read:** 1. Benutzer lässt Userdaten anzeigen **Update:** 1. Benutzer lässt Userdaten anzeigen 2. Benutzer verändert Attribute 3. Benutzer speichert Änderungen **Delete:** 1. Benutzer wird gelöscht
Extensions	-

Device erfassen

ID	03
Name	Device erfassen
Beschreibung	Der Benutzer möchte ein Device manuell hinzufügen. Der Endpunkt ist dem Benutzer bekannt.
Preconditions	• Applikation gestartet • Benutzer eingeloggt • Device Endpunkt ist dem Benutzer bekannt
Postconditions	• Falls ein Gerät gefunden wird, wird es angezeigt
Main Success Scenario	1. Benutzer öffnet Device Erfassung 2. Benutzer gibt Device Endpunkt ein 3. Benutzer startet Suchvorgang 4. Anfrage wird an Device gesendet 5. Antwort vom Device wird angezeigt
Extensions	5.a Timeout Fehlermeldung wird dem Benutzer angezeigt

Device finden

ID	04
Name	Device finden
Beschreibung	Der Benutzer möchte ein- oder meherere Devices finden. Endpunkt des Devices ist dem Benutzer unbekannt. Gefundene Devices sollen dem Benutzer aufgelistet werden.
Preconditions	• Applikation gestartet • Benutzer eingeloggt
Postconditions	• Gefundene Devices werden dem Benutzer angezeigt
Main Success Scenario	1. Benutzer öffnet Device Discovery 2. System listet eingegangene Anfragen von Devices auf
Extensions	2.a System zeigt Fehlermeldung an

Device assoziieren

ID	05
Name	Device assoziieren
Beschreibung	Der Benutzer möchte ein Device verwalten. Dazu muss er das gefundene Device in das System adoptieren.
Preconditions	• Applikation gestartet • Benutzer eingeloggt • Mindestens 1 Device gefunden
Postconditions	• Assoziation zu gefundenem Device erstellt
Main Success Scenario	1. Benutzer selektiert ein gefundenes Device aus.(UC 03 / UC 04) 2. Benutzer wählt ”Device adoptieren” aus 3. Assoziation wird im System eingetragen
Extensions	-

Device CRUD

ID	06
Name	Device CRUD
Beschreibung	Deviceverwaltung der Applikation
Preconditions	• Applikation gestartet • Benutzer eingeloggt
Postconditions	• Änderungen gespeichert
Main Success Scenario	**Create:** 1. UC 05: Device assoziieren **Read:** 1. Attribute eines Devices anzeigen **Update:** 1. Benutzer lässt Device Attribute anzeigen 2. Benutzer verändert Attribute 3. Benutzer speichert Änderungen **Delete:** 1. Device Assoziation wird gelöscht
Extensions	-

Device abfragen

ID	07
Name	Device abfragen
Beschreibung	Gewünschte Parameter und Attribute werden vom Device abgefragt
Preconditions	• Applikation gestartet • Benutzer eingeloggt
Postconditions	• Antwort vom Device oder Fehlermeldung wird angezeigt
Main Success Scenario	1. Benutzer wählt Device aus 2. Benutzer startet Abfrage 3. System sendet Anfrage an Device 4. System speichert Antwort (UC 06: Device CRUD)
Extensions	1.a Benutzer wählt mehrere Devices aus 3.a System sendet Anfragen an mehrere Devices 3.b Timeout-Fehlermeldung

Konfiguration speichern

ID	08
Name	Konfiguration speichern
Beschreibung	Konfigurationen von beliebigen Devices können im System gespeichert werden
Preconditions	• Applikation gestartet • Benutzer eingeloggt • Konfigurationsfile für Benutzer zugänglich
Postconditions	• Konfigurationsfile im System gespeichert
Main Success Scenario	1. Benutzer wählt ”Upload Konfiguration” 2. Benutzer wählt via File-Explorer eine Konfigurationsdatei aus 3. Konfigurationsdatei hochgeladen 4. Feedback an den Benutzer, dass die Datei erfolgreich hochgeladen ist.
Extensions	3.a I/O Fehler wird angezeigt 3.b Timeout-Fehler wird angezeigt

Konfiguration verteilen

ID	09
Name	Konfiguration verteilen
Beschreibung	Konfigurationen können an Devices versendet werden.
Preconditions	• Applikation gestartet • Benutzer eingeloggt • Konfiguration im System vorhanden
Postconditions	Konfiguration an Device geschickt
Main Success Scenario	1. Benutzer wählt Device aus 2. Benutzer wählt Konfiguration aus 3. Benutzer versendet Konfiguration 4. System zeigt Antwort des Devices an
Extensions	1.a Benutzer wählt mehrere Device aus 4.a System zeigt Antworten mehrerer Devices an 4.b Fehlermeldung wird angezeigt 4.c Timeout-Fehlermeldung

Konfiguration anzeigen

ID	10
Name	Konfiguration anzeigen
Beschreibung	Konfigurationsdetails im System können angesehen werden
Preconditions	• Applikation gestartet • Benutzer eingeloggt • Konfiguration im System vorhanden
Postconditions	• Konfigurationsdetails angezeigt
Main Success Scenario	1. Benutzer wählt Konfiguration aus 2. System zeigt Konfiguration an
Extensions	-

Software speichern

ID	11
Name	Software speichern
Beschreibung	Spezifische Software oder Firmware für Devices können im System gespeichert werden.
Preconditions	• Applikation gestartet • Benutzer eingeloggt • Softwarefile für Benutzer zugänglich
Postconditions	• Softwarefile im System gespeichert
1. Benutzer wählt ”Upload Software” 2. Benutzer wählt via File-Explorer eine Softwaredatei aus 3. Softwaredatei hochgeladen 4. Feedback an den Benutzer, dass die Datei erfolgreich hochgeladen ist.	
Extensions	3.a I/O Fehler wird angezeigt 3.b Timeout-Fehler wird angezeigt

Software verteilen

ID	12
Name	Software verteilen
Beschreibung	Jedes Device hat einen gewissen Softwarestand. Dieser kann durch neue Software ersetzt oder gepatched werden.
Preconditions	• Applikation gestartet • Benutzer ist eingeloggt • Devices assoziiert (UC 05)
Postconditions	• Softwarestand wurde ausgeliefert
Main Success Scenario	1. Benutzer selektiert das betreffende Device aus. 2. ”Software ausliefern” wird ausgewählt 3. Softwarestände für das Device werden angezeigt 4. Softwarestand wird selektiert 5. ”Sind Sie sich sicher?”-Abfrage wird angezeigt 6. Updatevorgang wird gestartet 7. Device Feedback anzeigen
Extensions	1.a Benutzer selektiert mehrere Geräte. 3.a Fehlermeldung, da kein Softwarestand vorhanden ist 5.a Abfrage wird verneint -> Vorgang wird abgebrochen 7.a Timeout wird erreicht. 7.b Fehlermeldungen zu der Wiederherstellung wird angezeigt.

Device sichern

ID	13
Name	Device sichern
Beschreibung	Von den Devices können Sicherungen gemacht werden, welche den Software- sowie den Konfigurationsstand beinhalten. Diese Sicherungen werden gespeichert.
Preconditions	• Applikation gestartet • Benutzer ist eingeloggt • Devices assoziiert (UC 5)
Postconditions	• Sicherung des Gerätes sind gespeichert
Main Success Scenario	1. Benutzer selektiert das betreffende Device aus. 2. ”Device sichern” wird ausgewählt 3. Die Sicherung wird benannt und ein Datum wird vergeben 4. ”Sind Sie sich sicher?”-Abfrage wird angezeigt 5. Sicherungsvorgang wird gestartet 6. Sicherung wird heruntergeladen 7. Device Feedback anzeigen 8. Sicherung wird gespeichert
Extensions	1.a Benutzer selektiert mehrere Geräte. 3.a Fehlermeldung wird angezeigt, da der gleiche Name schon vorhanden ist 4.a Abfrage wird verneint -> Vorgang wird abgebrochen 6.a I/O Fehlermeldung 7.a Timeout wird erreicht. 7.b Fehlermeldungen zu der Wiederherstellung wird angezeigt. 8.a I/O Fehlermeldung

Device wiederherstellen

ID	14
Name	Device wiederherstellen
Beschreibung	Das Device muss wiederhergestellt werden. Dadurch wird ein Software- sowie Konfigurationsstand auf das Device geschrieben.
Preconditions	• Applikation gestartet • Benutzer ist eingeloggt • Devices assoziiert (UC 5)
Postconditions	• Das Gerät funktioniert korrekt • Das Gerät hat den richtigen Softwarestand • Das Gerät hat die richtige Konfiguration
Main Success Scenario	1. Benutzer selektiert das betreffende Device aus. 2. ”Device wiederherstellen” wird ausgewählt 3. Backups für das Device werden angezeigt 4. Backup wird selektiert 5. ”Sind Sie sich sicher?”-Abfrage wird angezeigt 6. Wiederherstellungsvorgang wird gestartet 7. Device Feedback anzeigen
Extensions	1.a Benutzer selektiert mehrere Geräte. 3.a Fehlermeldung, da keine Backups vorhanden sind 5.a Abfrage wird verneint -> Vorgang wird abgebrochen 7.a Timeout wird erreicht. 7.b Fehlermeldungen zu der Wiederherstellung wird angezeigt.

Device authentisieren

ID	15
Name	Device authentisieren
Beschreibung	Alle Devices müssen beim Erstellen authentisiert werden. Dazu werden die Logindaten eingegeben und es wird eine Verbindung zum Gerät erstellt.
Preconditions	• Applikation gestartet • Benutzer ist eingeloggt • Devices sind erfasst
Postconditions	• Device ist erreichbar • Device ist authentisiert • Authentisierungsdaten sind gespeichert
Main Success Scenario	1. Device wird ausgewählt 2. Logindaten werden eingegeben 3. Der Verbindungsaufbau wird gemacht 4. Feedback des Devices 5. Speicherung der Authentisierungsdaten
Extensions	1.a Device wird durch UC 03: Device erstellen erfasst 1.b Device wird durch UC 04: Device finden erfasst 4.a Timeout 4.b Fehlermeldung des Devices anzeigen

Schlüssel CRUD

ID	15.1
Name	Schlüssel verwalten
Beschreibung	Alle Schlüssel (Z.B Username und Passwort) können verwaltet werden.
Preconditions	• Applikation gestartet • Benutzer ist eingeloggt • Devices assoziiert (UC 5)
Postconditions	• Schlüssel sind richtig abgespeichert
Main Success Scenario	**Create:** 1. UC 15.2: Device authentisieren **Read:** 1. Device wird ausgewählt 2. Schlüssel anzeigen wird gewählt 3. Schlüsseldaten werden angezeigt **Update:** 1. Device wird ausgewählt 2. Schlüssel anzeigen wird gewählt 3. Schlüsseldaten werden angepasst 4. Schlüsseldaten werden abgespeichert **Delete:** 1. Schlüsseldaten werden gelöscht
Extensions	Read/Update 2.a Admin Verifizierung

Zertifikate verteilen

ID	15.2
Name	Zertifikate verteilen
Beschreibung	Zertifikate werden auf die jeweiligen Devices verteilt.
Preconditions	• Applikation gestartet • Benutzer ist eingeloggt • Devices assoziiert (UC 5) • Zertifikate sind im richtigen Format vorhanden
Postconditions	• Zertifikate befinden sich auf den Devices • Zuteilung von Device-Zertifikat ist im Management hinterlegt
Main Success Scenario	1. Benutzer selektiert das betreffende Device aus. 2. Der Benutzer wählt ”Zertifikat verteilen” 3. Das gewünschte Zertifikat wird ausgewählt und bestätigt 4. Das Zertifikat wird an das Devices gesendet 5. Feedback anzeigen
Extensions	1.a Benutzer selektiert mehrere Geräte. 5.a Fehlermeldungen werden angezeigt. 5.b Timeout wird erreicht.

Zertifikate CRUD

ID	15.3
Name	Zertifikate CRUD
Beschreibung	Alle Zertifikate werden zentral verwaltet. Diese Zertifikate stammen von den Sensoren, damit man das Vertrauen überprüfen kann.
Preconditions	• Applikation gestartet • Benutzer ist eingeloggt • Devices assoziiert (UC 5)
Postconditions	• Zertifikate sind gespeichert • Veraltete Zertifikate sind gelöscht
Main Success Scenario	**Create:** 1. UC 15.2: Zertifikate verteilen **Read:** 1. Zertifikate werden von Device angefordert 2. Zertifikat wird auf Gültigkeit geprüft 3. Zertifikat wird in der Datenbank gespeichert **Update:** 1. Neues Zertifikat wird vom Device angefordert 2. Zertifikat wird auf Gültigkeit geprüft 3. Zertifikat wird in der Datenbank gespeichert **Delete:** 1. Zertifikat wird gelöscht
Extensions	-

Kommandos ausführen

ID	16
Name	Kommandos ausführen
Beschreibung	Dem Gerät werden Kommandos, wie zum Beispiel ”Reboot” oder ”Shutdown”, gesendet.
Preconditions	• Applikation gestartet • Benutzer ist eingeloggt • Devices assoziiert (UC 5)
Postconditions	• Kommandos sind ausgeführt • Device Feedback
Main Success Scenario	1. Benutzer selektiert das betreffende Device aus. 2. Das auszuführende Kommando wird gewählt/eingegeben 3. Das Kommando wird an alle ausgewählten Devices gesendet 4. Devicefeedback wird angezeigt.
Extensions	1.a Benutzer selektiert mehrere Geräte. 4.a Fehlermeldungen zu dem Kommando werden angezeigt. 4.b Timeout wird erreicht.

3.3 Nichtfunktionale Anforderungen

In diesem Kapitel behandeln wir die nichtfunktionalen Anforderungen an das Projekt. Wir behandeln Aspekte und Anforderungen aus den Bereichen Qualität, Schnittstellen und Randbedingungen.

3.3.1 Qualität

Bei der Softwarequalität stützen wir uns auf die ISO/IEC 9126 Norm. Es werden die Merkmale Funktionalität, Zuverlässigkeit, Benutzbarkeit, Effizienz, Wartbarkeit und Übertragbarkeit aufgeführt.

Funktionalität IoT Devices können von vielen unterschiedlichen Herstellern kommen. Devices können somit sehr unterschiedliche Attribute enthalten. Ebenfalls können sich die Art der Kommunikation und unterstützte Netzwerkprotokolle unterscheiden. Um die Funktionalität best möglich sicherzustellen, wird die Herstellerunterstützung vorerst stark eingeschränkt.

Zuverlässigkeit Eine Managementplattform von IoT Devices ist nicht realtime-kritisch. Für die Provisionierung, Fehlersuche oder Updateprozesse kann sie aber durchaus notwendig sein. Da die Applikation über das Internet erreichbar ist, muss die Applikationssicherheit gewährleistet sein. Eine Kompromittierung der Management Plattform könnte die gesamte IoT Landschaft eines Unternehmens beeinträchtigen.

Durch die vielseitigen Aufgaben muss auf die Parallelität geachtet werden. Es werden durchaus zeitintensive Tasks ausgeführt welche potenziell die Applikation über längere Zeit blockieren könnten.

Benutzbarkeit Um eine einfache Bedienung zu gewährleisten soll eine einfache und zweckmässige Benutzeroberfläche zur Verfügung gestellt werden.

Effizienz Die Effizienz hängt stark von den IoT Devices und deren Internetbandbreite und Rechenleistung ab.

Wartbarkeit Sämtliche Teile der Software sollen möglichst modular und lose gekoppelt aufgebaut werden. Eine Management Applikation für IoT könnte potenziell sehr umfangreich sein und eine Weiterentwicklung muss in Betracht gezogen werden.

Übertragbarkeit Die Applikation soll für Clients über gängige Web Browser zugänglich sein. Deshalb sollte eine Übertragbarkeit auf verschiedene Clientsysteme keine Probleme darstellen.

3.3.2 Schnittstellen

Benutzerschnittstellen Die Steuerung des Programms ist über eine grafische Weboberfläche vorgesehen. Tastatur und Maus sind notwendig.

Netzwerkschnittstellen Um die Applikation zu bedienen benötigt der Client einen funktionierenden Internetanschluss. Die Applikation selbst muss ebenfalls über einen Internetzugang verfügen und die nötigen Kommunikationsports müssen geöffnet sein.

3.3.3 Sicherheit

Als eingeloggter Benutzer könnte man viel Schaden am IoT System verursachen. Man könnte beispielsweise Geräte herunterfahren oder mit bösartiger Malware austatten. Es gilt unbedingt die Grundregeln der Informationssicherheit einzuhalten. Sämtliche Zugriffe sollen autorisiert- und vertrauliche Informationen verschlüsselt übertragen werden.

4. Literaturverzeichnis

[1] International Data Corporation. Connecting the iot: The road to success. http://www.idc.com/infographics/IoT, 2015.

[2] Dominique Guinard, Vlad Trifa, Stamatis Karnouskos, Patrik Spiess, and Domnic Savio. Interacting with the soa-based internet of things: Discovery, query, selection, and on-demand provisioning of web services. http://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=5416674, 2010.

[3] Tobias Heer, Oscar Garcia-Morchon, René Hummen, Sye Loong Keoh, Sandeep S. Kumar, and KlausWehrle. Security challenges in the ip-based internet of things. https://link.springer.com/article/10.1007/s11277-011-0385-5, 2011.

[4] Gregor Hohpe and Bobby Woolf. Enterprise integration patterns. http://www.enterpriseintegrationpatterns.com/img/PublishSubscribeSolution.gif, 2011.

[5] IETF. The constrained application protocol (coap). https://tools.ietf.org/html/rfc7252, 2014.

[6] Sheetal Kumbhar. Getting ready for iot monitoring. http://www.iot-now.com/2016/11/17/55128-getting-ready-for-iot-monitoring, 2016.

[7] micrium.com. Devices & networks. https://www.micrium.com/wp-content/uploads/2014/03/internet-of-things.png, 2017.

[8] micrium.com. Devices & networks. https://www.micrium.com/wp-content/uploads/2014/03/wireless-sensor-network.png, 2017.

[9] micrium.com. Iot networks. https://www.micrium.com/iot/devices/, 2017.

[10] Microsoft. Overview of device management with iot hub. https://docs.microsoft.com/en-us/azure/iot-hub/iot-hub-device-management-overview, 2017.

[11] Daniele Miorandi, Sabrina Sicari, Francesco De Pellegrini, and Imrich Chlamtac. Internet of things: Vision, applications and research challenges. http://www.sciencedirect.com/science/article/pii/S1570870512000674, 2012.

[12] Dominik Obermaier. Internet der dinge: Leichtgewichtiges messaging. https://www.informatik-aktuell.de/betrieb/netzwerke/internet-der-dinge-protokolle-verfahren-und-integration-von-mqtt.html, 2014.

[13] Dominik Obermaier. Iot-protokolldschungel – ein wegweiser. https://www.informatik-aktuell.de/betrieb/netzwerke/iot-protokolldschungel-ein-wegweiser.html, 2015.

[14] Angela Orebaugh. Internet der dinge: Was zu tun ist, um iot-security realität werden zu lassen. http://www.searchsecurity.de/meinung/Internet-der-Dinge-Was-zu-tun-ist-um-IoT-Security-Realitaet-werden-zu-lassen, 2015s.

[15] John Pardey. Netzwerkmanagement. http://www.it-administrator.de/themen/netzwerkmanagement/grundlagen/1
2012.

[16] Postscapes. Internet of things infographic. http://www.postscapes.com/what-exactly-is-the-internet-of-things-infographic, 2014.

[17] Alexandru Radovici. Introduction to the internet of things summer school. https://www.youtube.com/watch?v=G4-CtKkrOmc, 2015.

[18] Karen Rose, Scott Eldridge, and Lyman Chapin. The internet of things: An overview.

https://www.internetsociety.org/sites/default/files/ISOC-IoT-Overview-20151221-en.pdf, 2015.

[19] Stan Schneider. Understanding the protocols behind the internet of things. http://electronicdesign.com/iot/understanding-protocols-behind-internet-things, 2013.

[20] Stan Schneider. Understanding the iot protocols. https://de.slideshare.net/RealTimeInnovations/io-34485340, 2014.

[21] servicedesignpatterns.com. Request/response. http://www.servicedesignpatterns.com/Images/RequestResponse.jp 2011.

[22] Zach Shelby. Constrained application protocol (coap) tutorial. https://www.youtube.com/watch?v=4bSr5x5gKvA, 2014.

[23] Unknown. Computerized maintenance management system. https://de.wikipedia.org/wiki/Computerized_Maintenance_Management_System, 2015.

[24] JP Vasseur. The internet of things: Architecture and protocols. https://www.youtube.com/watch?v=co2MLqkJVXs, 2014.

[25] John Weber. Fundamentals of iot device management. http://iotdesign.embedded-computing.com/articles/fundamentals-of-iot-device-management/, 2016.

Abbildungsverzeichnis