HOCHSCHULE
HANNOVER
UNIVERSITY OF
APPLIED SCIENCES
AND ARTS
–
Fakultät IV
Wirtschaft und
Informatik

Bluetooth Exposure Logging zur Besucherstromanalyse im Außenbereich

Kilian Dangendorf

Masterarbeit im Studiengang „Angewandte Informatik“

15. November 2022

Autor: Kilian Dangendorf, B. Sc.
1579 564
kilian.dangendorf@gmail.com

Erstprüferin: Prof. Dr. Frauke Sprengel
Abteilung Informatik, Fakultät IV
Hochschule Hannover
frauke.sprengel@hs-hannover.de

Zweitprüfer: Simon Niechzial, M. Sc.
ATMINA Solutions GmbH
Theaterstraße 8, 30159 Hannover
simon.niechzial@atmina.de

Selbständigkeitserklärung

Hiermit erkläre ich, dass ich die eingereichte Masterarbeit selbständig und ohne fremde Hilfe verfasst, andere als die von mir angegebenen Quellen und Hilfsmittel nicht benutzt und die den benutzten Werken wörtlich oder inhaltlich entnommenen Stellen als solche kenntlich gemacht habe.

Hannover, den 15. November 2022 Kilian Dangendorf

Zusammenfassung

Bluetooth ist ein weit verbreitetes drahtloses Übertragungsprotokoll, das in vielen mobilen Geräten wie bspw. Tablets, Kopfhörer oder Smartwatches verwendet wird. Bluetooth-fähige Geräte senden mehrmals pro Minute öffentliche Advertisements, die u.a. die einzigartige MAC-Adresse des Gerätes beinhalten. Das Mitschneiden dieser Advertisements mittels Bluetooth-Logger ermöglicht es, Bewegungen der Geräte zu analysieren und lassen somit Rückschlüsse auf die Bewegungen der Besitzenden zu.

Zum Schutz der Privatsphäre werden seit 2014 zufällig erzeugte MAC-Adressen in Advertisements verwendet. Eine sog. randomisierte MAC-Adresse bleibt durchschnittlich 15 Minuten lang gültig und wird dann durch eine neue zufällige Adresse ersetzt. Der Aufenthalt eines Geräts zu einem späteren Zeitpunkt kann nicht bestimmt werden. Dennoch kann der Wechsel eines Geräts von einem Bluetooth-Logger zu einem anderen innerhalb dieser 15 Minuten erkannt und somit eine Bewegung des Gerätes abgeleitet werden.

Durch Apps der Kontaktpersonennachverfolgung wie die Corona-Warn-App (CWA) senden auch vermeintlich inaktive Smartphones Bluetooth-Advertisements. Mit etwa einem Viertel der Aufzeichnungen unterstützt die CWA die Auswertungen dieser experimentellen Arbeit.

Um die praktische Anwendbarkeit zu demonstrieren, wurde der Erlebniszoo Hannover als Testgelände genutzt. Die Auswertung der über sieben Wochen gesammelten Daten ermöglichte die Analyse von Stoßzeiten, stark besuchten Orten und Besucherströmen.

Schlüsselwörter — Bluetooth Exposure Logging, Besucherstromanalyse, ESP32, IoT, Edge Computing, randomisierte MAC-Adressen.

Lizenz

Inhaltsverzeichnis

Abbildungsverzeichnis

Tabellenverzeichnis

Listings

1. Einleitung und Ausgangssituation

Bluetooth ist eines der am weitesten verbreiteten Funkübertragungsprotokolle. In den letzten zehn Jahren hat sich der jährliche Verkauf von Bluetooth-Geräten von 2,4 auf 4,7 Milliarden verdoppelt[1]. Es gibt eine Vielzahl an mobilen Bluetooth-Geräten (z.B. Smartphones, Smartwatches, Tablets, Laptops, Lautsprecher, Kopfhörer, Autos, Mietroller/-räder, u.v.a.). Um die Kopplungsbereitschaft zu kommunizieren, senden Bluetooth-Geräte mehrmals pro Minute sog. *Advertisements*. Die enthaltene *MAC-Adresse* kann als Identifikator genutzt werden, da sie für jedes Gerät einzigartig ist.

In "The Use of Bluetooth for Analysing Spatiotemporal Dynamics of Human Movement at Mass Events: A Case Study of the Ghent Festivities" aus dem Jahr 2012 nutzen Versichele u. a. Bluetooth zur Erkennung von Besucherströmen ([Ver+12b; Ver+12a]). Dabei stellen sie auf einem Testgelände mehrere Bluetooth-Messgeräte auf. Diese erfassen MAC-Adresse von Bluetooth Geräten und legen sie mit dem Zeitpunkt der Messung ab. Die Auswertungen der Messdaten umfassen Aufenthaltsdauern, Stoßzeiten, hoch frequentierte Orte und Besucherströme.

Zum Schutz der Privatsphäre werden seit dem Jahr 2014 zufällig erzeugte, sog. *randomisierte MAC-Adressen*, eingesetzt. Eine randomisierte MAC-Adresse besteht nur ca. 15 Minuten und wird danach von einer neuen zufälligen Adresse abgelöst, die keinen Rückschluss auf die vorherige zulässt. Als Identifikator für den Zeitraum von 15 Minuten dient eine randomisierte MAC-Adresse dennoch, da die Kollisionswahrscheinlichkeit zweier MAC-Adresse bei 70,4 Billionen[2] möglichen Randomisierungen nahezu null beträgt.

Der Ansatz von Versichele u. a. setzt MAC-Adressen, die sich nicht ändern, voraus. Die von ihnen erarbeiteten Ergebnisse basieren auf dem Wiedererkennen eines Gerätes über mehrere Messstationen und über einen längeren Zeitraum (sogar über mehrere Tage). Mit randomisierten MAC-Adressen ist ein Wiedererkennen nur innerhalb von 15 Minuten möglich. Der Aufenthalt eines Gerätes ist z.B. nach einer Stunde nicht zu bestimmen. Auch ist ein Wiedererkennen eines Gerätes an einem anderen Tag nicht möglich. Der Übergang eines Gerätes von einem Bluetooth-Messgerät (im Folgenden *Bluetooth-Logger* genannt) zu einem anderen innerhalb der Gültigkeitsdauer einer MAC-Adresse kann aber ermittelt werden. Eine große Grundmenge an Daten ist nötig, um eine aussagekräftige Menge an Übergängen von Geräten mit randomisierter MAC-Adresse abzuleiten. Die steigende Anzahl an Bluetooth-fähigen

[1] Vgl. Marktupdate der Bluetooth SIG, Kennzahl „Total Bluetooth Device Shipments", 2012: 2,4 Mrd., 2021: 4,7 Mrd.
`https://www.bluetooth.com/de/bluetooth-resources/2018-bluetooth-market-update/`
`https://www.bluetooth.com/de/2022-market-update/`.

[2] Eine MAC-Adresse ist 48 Bit lang (siehe Abschnitt 3.2), davon werden bei einer randomisierten MAC-Adresse 46 Bit zufällig gesetzt. Daraus ergeben sich $2^{46} \approx 70,4$ Billionen mögliche MAC-Adressen.

Geräten könnte diese ermöglichen. Diese aktuellen Bedingungen führten zu der ersten Forschungsfrage:

1. Wie können Besucherströme anhand von Bluetooth-Geräten, die eine randomisierte MAC-Adresse nutzen, ermittelt werden?

Um die praktische Nutzbarkeit zu demonstrieren, wurde der *Erlebniszoo Hannover* als Testgelände ausgewählt. Dieses umfasst eine Fläche von 22 Hektar. Ein vollständiger Rundgang ist etwa 5 km lang, ein verkürzter etwa 2 km. Jährlich werden ca. 1 Mio. besuchende Personen verzeichnet[3]. Für Besuchende wird auf dem Testgelände ein öffentliches WLAN angeboten. Das wurde für die Internetanbindung der Bluetooth-Logger benutzt.

Unter der Überschrift des *nachhaltigem Tourismus'* ist seitens der Betreibenden ein Besuchermanagement erwünscht. Der erste Schritt zu diesem Ziel ist die Besucherstromanalyse. Perspektivisch kann auf dieser Grundlage eine Besucherstromentzerrung mithilfe eines Empfehlungsdienstes (Recommender-System) erfolgen. Das Besuchererlebnis könnte optimiert werden, wenn die periodischen und räumlichen Auslastungsspitzen sich auflösen. Konkret könnten Personen über eine App „angestupst“ werden, weniger stark besuchte Bereiche zu besuchen (Nudging). Ein Beispiel hierfür ist der *Strandticker*[4] der Lübecker Bucht. Je Strandabschnitt werden Besucherzahlen gemessen und die jeweilige Auslastung als Strandampel dargestellt. Besuchende können diese in Echtzeit abrufen, um hoch frequentierte Bereiche zu meiden.

Bisherige Erfassungen der Besucherströme im Testgelände erfolgten durch Umfragen. Diese wird hier mit lokaler Sensorik automatisiert. Mit dem Ziel das Besuchererlebnis zu optimieren, sollen folgende Fragen der Betreibenden mit den Ergebnissen dieser Arbeit beantwortet werden:

- Um welche Tageszeit und an welchem Wochentag sind viele Besuchende vor Ort?
- Welche Orte sind stark bzw. schwach besucht?
- Welche Route nehmen die Besuchenden im Gelände? Folgen sie dem beschilderten Rundgang?

Vor diesem Hintergrund stellte sich die zweite Forschungsfrage:

2. Wie können im Erlebniszoo Hannover Stoßzeiten, hoch frequentierte Orte und Besucherströme bestimmt werden?

Im Jahr 2020 wurde die Corona-Warn-App (kurz *CWA*) eingeführt. Diese hat ca. 25 Mio. aktive Nutzende[5], das entspricht mehr als jeder vierten Person in Deutschland. Zur Kontaktpersonennachverfolgung sendet die App mehrmals pro Sekunde Advertisements. Daraus ergab sich die dritte Forschungsfrage:

3. Können die Daten der CWA (ergänzend oder ausschließlich) für eine Besucherstromanalyse genutzt werden?

[3] Coronabedingt fielen die Besucherzahlen im Jahr 2021 auf ca. 700.000 Besuchende. In den Vorjahren wurden jeweils mehr als 1 Mio. Besuchende gezählt. Vgl. Statista: „Anzahl der Besucher des Zoos Hannover vom Jahr 2008 bis 2019 unter: `https://de.statista.com/statistik/daten/studie/248598/umfrage/besucherzahlen-des-zoo-hannover/`

[4] Siehe Strandticker der Lübecker Bucht unter: `https://www.luebecker-bucht.guide/beachticker`

[5] „[...] Insgesamt 25,1 Mio. (monatlich) aktive Nutzende bzw. 28,1 Mio. (monatlich) aktive Endgeräte (Stand: 26. Februar 2022) aus den App-Stores.“ [Ope22, vgl.].

Zur Beantwortung der formulierten Forschungsfragen wird wie folgt vorgegangen. Auf dem Testgelände werden an geeigneten Stellen Bluetooth-Logger aufgestellt. Abbildung 1.1 zeigt diese exemplarisch als *Logger A* und *B*. Innerhalb ihres Aufzeichnungsbereichs erfassen die Bluetooth-Logger unabhängig alle Bluetooth-Advertisements, die von Bluetooth-fähigen Geräten ausgesendet werden. Als Identifikator eines Gerätes – hier X, Y und Z – dient die MAC-Adresse. Die Meldung an einen zentralen Server ist in Abbildung 1.1 mit den Sprechblasen dargestellt. Serverseitig werden empfangene Messdaten anonymisiert, transformiert und in einer Datenbank abgelegt.

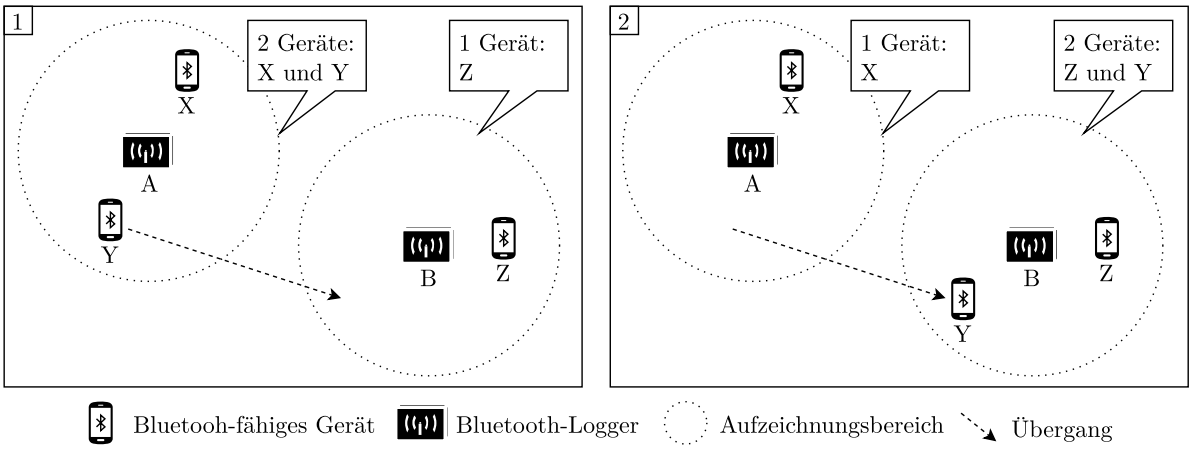


Abbildung 1.1.: Exemplarischer Versuchsaufbau zu zwei aufeinander folgenden Zeitpunkten. Bluetooth-Logger erfassen Advertisements Bluetooth-fähige Geräte innerhalb ihres Aufzeichnungsbereichs. Wechselt das Gerät Y von Bluetooth-Logger A zu B, lässt sich dies als Übergang von Position A nach B erkennen.

Die anschließende Datenauswertung ergibt:

- Zu Zeitpunkt *1* befinden sich bei Bluetooth-Logger A zwei Geräte, bei B eins, zu dem späteren Zeitpunkt *2* bei A nur eins und bei B zwei. Mithilfe dieser Daten können zu einem bestimmten Zeitpunkt stark bzw. schwach besuchte Bereiche analysiert werden.
- Wechseln Geräte von einem Bluetooth-Logger zu einem anderen innerhalb der Gültigkeitsdauer einer randomisierten MAC-Adresse, kann dies als Übergang erkannt werden. In Abbildung 1.1 wechselt Gerät Y von Bluetooth-Logger A zu B. So können neben absoluten Gerätezahlen auch Ströme erkannt werden.

Mit diesem Versuchsaufbau wird die Anwesenheit von *physischen Geräten* in eine *virtuelle Repräsentation* überführt, welche Aussagen über Besucherströme zulässt.

Aus dem Versuchsaufbau ergibt sich die folgende Gliederung. Die Technischen Grundladen werden in Kapitel 3 gelegt. Dabei wird der Datenschutz im Sinne des Bundesdatenschutzgesetzes beachtet. Dann wird die eingesetzte Systemarchitektur in Kapitel 4 auf Grundlage der Fünfschichtenarchitektur erarbeitet, entlang derer sich die weitere Gliederung orientiert. Kapitel 5 erklärt MQTT, das eingesetzte Kommunikationsprotokoll zwischen Bluetooth-Logger

und Server. Danach wird den Messdaten chronologisch durch den Versuchsaufbau gefolgt. Die genutzte Hardware sowie die Funktionsweise der Bluetooth-Logger wird in Kapitel 6 erklärt. Während der Datenspeicherung, behandelt in Kapitel 7, findet die Anonymisierung der Daten statt. Nach ihrer Reduktion werden die Messdaten in einer Datenbank abgelegt. Kapitel 8 bildet mit der Datenauswertung den Schwerpunkt dieser Arbeit. Es werden die abgelegten Daten zur Beantwortung der Forschungsfragen abgefragt und visuell aufbereitet. Mit einer kritischen Bewertung in Kapitel 9 schließt diese Arbeit.

2. Technologien und Ansätze für die Besucherstromanalyse

Die Besucherstromanalyse beginnt mit der Besucherstrommessung. In diesem Kapitel soll ein Überblick über die möglichen Messtechnologien gegeben werden. Dabei wird die eingesetzte Hardware und das Potential der einzelnen Technologien sowohl im Innen- als auch im Außenbereich betrachtet.

Das *Global Positioning System*, kurz GPS ist die führende Technologie zur Positionsbestimmung im Außenbereich. Dabei werden passiv Radiosignale von Satelliten empfangen und die jeweilige Entfernung bestimmt. Durch Multilateration wird der Standort auf 10 m genau errechnet. Im Innenbereich ist eine genaue Ortung mit GPS aufgrund von Verdeckung nicht möglich. Hier wird auf andere Technologien zurückgegriffen, um die exakte Position im Raum zu bestimmen.

Ähnlich der Positionsbestimmung bei GPS werden im Innenbereich von Spachos und Plataniotis in [SP20] Bluetooth-*Beacons* an festen Positionen im Raum platziert. Beacons senden – wie die deutsche Übersetzung des Leuchtturms – in regelmäßigen Abständen dasselbe Signal, mit dem Ziel wiedererkannt zu werden. Von den Beacons gesendete Advertisements werden von einer Smartphone-App empfangen und auf Grundlage der empfangenen Signalstärke (Received Signal Strength Indicator, kurz RSSI) mehrerer Beacons wird mithilfe von Multilateration und dem Path-Loss-Modell die genaue Position im Raum bestimmt. Ziel ist es, in der App standortspezifische Informationen bereitzustellen. Dies setzt eine hohe Dichte an Beacons voraus, damit an einem Punkt im Raum Advertisements mehrerer Beacons empfangen werden können.

Den gleiche Ansatz verfolgen Sophia u. a. in [Sop+21], allerdings mit dem Mikrocontroller ESP32. Dieser wird jeweils als Sender (Beacon) programmiert und an mehreren Stellen im Raum platziert. Gleichzeitig wird der ESP32 mit anderer Firmware als mobiler Empfänger eingesetzt. Die Berechnung der Position im Raum erfolgt analog [SP20].

Eine Alternative zu Bluetooth für die Positionsbestimmung stellen sog. *WiFi Probe Requests*, kurz *WiFi Probes*, dar. WLAN-Geräte senden diese – vergleichbar mit Bluetooth-Advertisements – in regelmäßigen Abständen. Dieses Protokoll soll das Entdecken und Beitreten verfügbarer WLAN-Netzwerke beschleunigen. WiFi Probes enthalten die MAC-Adresse des sendenden Gerätes. Dessen Randomisierung hat sich ebenfalls seit dem Jahr 2014 etabliert. Anders als bei Bluetooth, wo ca. 15 Minuten Advertisements von einer MAC-Adresse gesendet werden, wird für jede WiFi Probe eine neue zufällige MAC-Adresse generiert. Eine Übergangserkennung lassen WiFi Probes nicht mehr zu.

Hong, De Silva und Chan nutzen WiFi Probes randomisierter MAC-Adressen zur Besucherstromanalyse in [HDC18]. In einem mehrstöckigen Museum werden zehn Raspberry Pis verteilt,

die als Messgeräte fungieren. Für die Besucherstromanalyse verlagern Hong, De Silva und Chan die Berechnung der Position auf einen Server, der die Messungen verarbeitet[1]. Die genaue Positionsbestimmung steht hier nicht im Vordergrund, stattdessen sollen Übergänge von Besuchenden zwischen den Messgeräten modelliert werden. Grundlage waren Übergangsaufzeichnungen über sechs Monate aus der Vergangenheit, als noch keine randomisierten MAC-Adressen eingesetzt wurden. Mithilfe der bestehenden Übergangsmatrix und dem *verdeckten Markowmodell* (Hidden Markov Model) wird versucht die Bewegungsbahnen in aktuellen Messdaten erkenntlich zu machen.

Es stellt sich die Frage, wie die bisher genannten Ansätze der Besucherstrommessung auf den Außenbereich übertragbar sind. Im Innen- und Außenbereich herrschen unterschiedliche technische Voraussetzungen. So ist die Infrastruktur bezogen auf Stromversorgung und Netzwerkanbindung im Außenbereich i. d. R. schlechter ausgebaut. Das erzwingt größere Abstände der Messgeräte und sorgt u. U. für ungenauere Messungen. Allerdings ist die exakte Positionsbestimmung im Außenbereich nicht so wichtig, da interessante Orte (Points of Interest, kurz POI) i. d. R. weiter auseinander liegen. Umweltfaktoren können bei dem Design der Sensorik – durch bspw. Regen- oder UV-Schutz – berücksichtigt werden. Diese Annahmen bestätigen sich in dem folgenden Ansatz von Bonné u. a. Sowohl im Innen- als auch Außenbereich wurde *WiFiPi* eingesetzt [Bon+13]. Mithilfe eines Raspberry Pis werden WiFi-Probes gesammelt, um auf Besucherzahlen zu schließen. Für eine genauere Positionsbestimmung wird der RSSI aufgezeichnet, aber nicht ausgewertet. Das Ergebnis sind hoch frequentierte Bereiche in Echtzeit. Übergänge werden hier nicht berechnet.

Den Ansatz der WiFi Probes verfolgt auch Basalamah mit sog. *WiFi Sniffers* in [Bas16]. Allerdings finden die Untersuchungen im Jahr 2016 noch vor der vollflächigen Verbreitung randomisierter MAC-Adressen statt. Entlang eines Pilgerpfads ermittelt Basalamah Stoßzeiten und Gerätezahlen über acht WiFi Sniffers. Die Besucherstromanalyse endet mit dem Aufstellen der Übergangsmatrix. Als Hardware wird hier der Einplatinencomputer BeagleBone eingesetzt, Messdaten werden über das Mobilfunknetz gesendet.

Bluetooth findet von Versichele u. a. im Außenbereich Einsatz in [Ver+12b; Ver+12a]. Wie eingangs aufgezählt gehören Aufenthaltsdauern, Stoßzeiten, hoch frequentierte Orte und Übergänge zu den Ergebnissen. Eingesetzte Hardware ist ein Einplatinencomputer mit Bluetooth-Dongle. Die Übertragung der Messdaten erfolgt je nach Standort über Kabel oder Mobilfunknetz.

Eine Kombination aus Bluetooth und WiFi Probes für den Außenbereich wird in der Software *ESP32-Paxcounter* entwickelt [Ws22]. Wie im Namen enthalten findet als Hardware der Mikrocontroller ESP32 Einsatz. Das Ziel ist das Zählen unterschiedlicher Geräte innerhalb des Empfangsbereich. Dabei verlassen personenbezogene Daten nicht das Messgerät, nur die Anzahl an empfangenen Geräten wird weitergegeben. So lassen sich Auswertungen zu hoch frequentierten Orten erstellen, jedoch keine Übergänge ableiten.

Die oben genannten Analysen bieten eine Grundlage für weitere Berechnungen im Kontext des Besuchermanagement. Auf Basis der Übergangswahrscheinlichkeiten können Vorhersagen getroffen oder Simulationen ausgeführt werden. Hu u. a. nutzen eine Übergangsmatrix als Grund-

[1] Die zuvor vorgestellten Ansätze berechneten die Position auf Seiten des bewegenden Objektes.

lage für Vorhersagen in [Hu+12]. Mithilfe eines neuronalen Netzes können Besucherzahlen zu bestimmten Tageszeiten vorhergesagt werden.

Die aufgeführten Technologien für die Positionsbestimmung auf Geräten der Besuchenden (GPS und Bluetooth-Beacons) kommen für eine Besucherstrommessung aufgrund der notwendigen Kooperationsbereitschaft der Besuchenden nicht infrage. Passive Messungen auf Basis von Bluetooth und WiFi Probes sind vielversprechend, ein explizites Einverständnis der Besuchenden ist hierfür nicht nötig. WiFi Probes lassen, seit der MAC-Adresse-Randomisierung, keinen Rückschluss auf Übergänge mehr zu. So fällt die Wahl auf Bluetooth. Hier lassen sich Übergänge zumindest während der Gültigkeitsdauer einer randomisierten MAC-Adresse von ca. 15 Minuten erkennen. Durch die steigenden Auslieferungszahlen der Bluetooth SIG sind in den nächsten Jahren weiterhin Bluetooth-Geräte zu erwarten. Auch begründet das, den Einfluss der CWA auf eine Besucherstrommessung zu analysieren.

Bezüglich der Hardware finden sich in den vorgestellten Ansätzen, neben professioneller Sensorik, Behelfslösungen auf Basis des Raspberry Pis, BeagleBones oder ESP32, die aussagekräftig sind. Dies bekräftigt deren zukünftigen Einsatz für die Besucherstrommessung auf Basis von Bluetooth.

3. Präliminarien

Die Grundlage der Datenerfassung dieser Arbeit bilden *Bluetooth-Advertisements* (auch *Advertising Events*, dt. *Aufmerksamkeitshinweise*). Jedes Bluetooth-Gerät sendet auf einem öffentlichen unverschlüsselten Kanal periodisch Advertisements[1]. Diese dienen dazu, anderen Geräten ihre Anwesenheit mitzuteilen, um eine potentielle Verbindung vorzubereiten. Das Aufzeichnen dieser Anwesenheitsinformation wird in Abschnitt 3.1 als *Bluetooth Exposure Logging* definiert.

Als Identifikator eines Bluetooth-Gerätes wird die MAC-Adresse genutzt, deren Aufbau in Abschnitt 3.2 erklärt wird. *Randomisierte MAC-Adressen* wurden eingeführt, um das Verfolgen bzw. Wiedererkennen eines Gerätes mittels Aufzeichnen der Advertisements zu verhindern und so die Privatsphäre der Nutzenden zu schützen (siehe Abschnitt 3.3). Welche Informationen dennoch aus Advertisements extrahiert werden können, beschreibt Abschnitt 3.4.

Anforderungen an den Datenschutz und daraus folgende Maßnahmen der Anonymisierung werden in Abschnitt 3.5 formuliert. Zuletzt werden in Abschnitt 3.6 Annahmen getroffen, die für den weiteren Verlauf dieser Arbeit gelten.

3.1. Bluetooth Exposure Logging

Im Kontext von Bluetooth-Scannern findet man Begriffe wie *Bluesniffing*[2], welches die Aufgabe hat, den Inhalt von Bluetooth Verbindungen unbefugt mitzulesen. Dieses „Ausspähen von Daten“ ist nach §202a StGB untersagt[3]. Um sich davon zu distanzieren, wird hier die Bezeichnung *Bluetooth Logging* gewählt. Es werden nur Metadaten von Bluetooth-Advertisements erfasst, nicht aber der Inhalt (*payload*) von bestehenden Verbindungen.

Zur Eindämmung des Coronavirus' entwickeln Apple Inc. und Google LLC das *Exposure Notification Framework* (kurz ENF) [AG20]. Dieses setzt die CWA zur Kontaktpersonennachverfolgung ein. Dabei senden Smartphones mehrmals pro Sekunde Zufallsschlüssel über Bluetooth und speichern empfangene Schlüssel (sog. *Exposure Logging*). Erfolgt zu einem empfangenen Schlüssel eine COVID-19-positiv Meldung, berechnet die CWA Entfernung sowie Dauer der aufgezeichneten Begegnung. Überschreiten diese einen Schwellwert, werden Nutzende vor erhöhtem Risiko gewarnt. Sowohl der gesendete Schlüssel als auch die Bluetooth MAC-Adresse der Sendenden sind zum Schutz der Privatsphäre randomisiert (siehe Abschnitt 3.3). Das Zeitintervall zur erneuten randomisierten MAC-Adresse sowie einhergehendem CWA-Zufallsschlüssel liegt betriebssystem seitig zwischen zehn und 20 Minuten [vgl. AG20, S. 5].

[1] Bluetooth Advertisements wurden mit Bluetooth 4.0 im Jahr 2010 eingeführt. Ältere Spezifikationen finden hier keine Beachtung.

[2] Auch *Bluesnarfing*, *Bluejacking*, *Bluebugging*, u.v.a.

[3] Vgl. Strafgesetzbuches §202a unter: `https://www.gesetze-im-internet.de/stgb/__202a.html`

Neben den Bluetooth-Advertisements der CWA werden im Rahmen dieser Arbeit sämtliche Bluetooth „Begegnungen“ (*Exposures*) aufgezeichnet (*geloggt*). Es ergibt sich die Verfahrensbezeichnung *Bluetooth Exposure Logging*. Ein Messgerät wird *Bluetooth-Logger* genannt.

3.2. Öffentliche MAC-Adresse

Jedes Netzwerkgerät besitzt eine eindeutige MAC-Adresse. MAC steht für *Media Access Control* (dt. Medienzugriffssteuerung) und dient der Identifizierung von Geräten eines Netzwerks. Ursprünglich setzt sie sich die 48 Bit lange Adresse aus zwei Teilen zusammen; 24 Bit Herstellerkennung und 24 Bit Gerätekennung (vgl. Abbildung 3.1). Die Herstellerkennung (*Organizationally Unique Identifier*, kurz OUI) wird von der *IEEE Registrierungsstelle*[4] vergeben und als Nachschlagewerk zur Verfügung gestellt. So lässt sich von einer MAC-Adresse auf den Hersteller eines Gerätes schließen (sog. *OUI Lookup*).

Die Gerätekennung (*Network Interface Controller*, kurz NIC) teilt der Hersteller seinen Geräte zu. Eine aus OUI und NIC zusammengesetzte MAC-Adresse ist für jedes Netzwerkgerät eindeutig und unveränderbar in der Hardware hinterlegt.

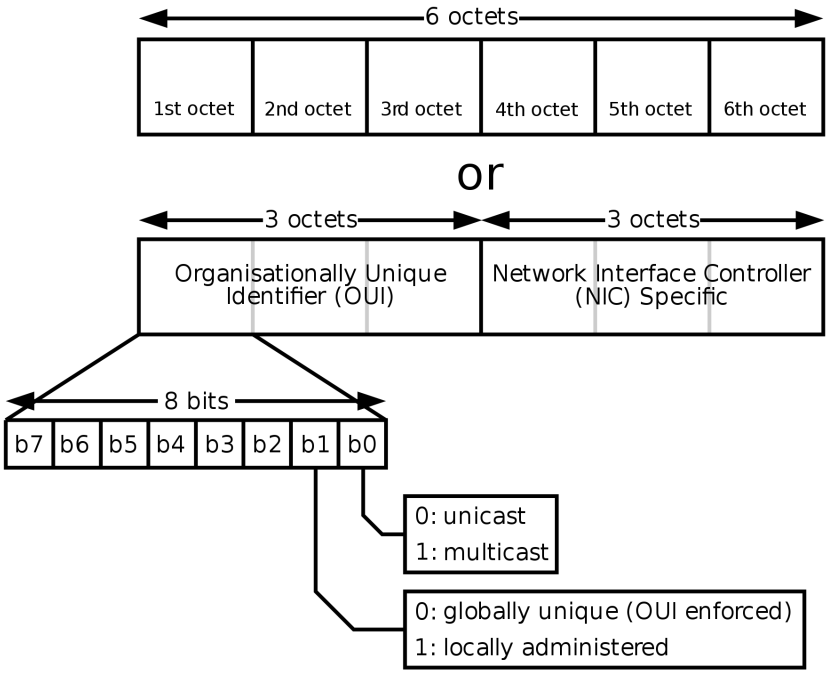


Abbildung 3.1.: Aufbau einer MAC-Adresse. Die ersten drei Bytes (*octets*) bilden die Herstellerkennung (*OUI*). Das Bit *b1* des ersten Bytes gibt an, ob es sich um eine öffentliche oder randomisierte MAC-Adresse handelt. Quelle: `https://en.wikipedia.org/wiki/MAC_address`.

[4]IEEE Registration Authority der IEEE Standards Association.
Mehr Info unter: `https://standards.ieee.org/faqs/regauth/`

Um die Bluetooth-Kommunikation zu ermöglichen, wird die MAC-Adresse publiziert. Bei mobilen Geräten (wie bspw. Smartphones, -watches oder Bluetooth-Kopfhörern) lassen sich mithilfe der MAC-Adresse Anwesenheit und Bewegungsprofil der Nutzenden ableiten. Es handelt sich um personenbezogene Daten, da sie eine eins zu eins Zuordnung einer MAC-Adresse zu einer Person ermöglichen.

Mit der Einführung von Bluetooth 4.2 im Jahr 2014 wurde neben dem Energiesparmodus (Bluetooth Low Energy, kurz *BLE*) die MAC-Adressen-Randomisierung zum Schutz der Privatsphäre eingeführt. In den darauffolgenden Jahren unterstützen immer mehr Geräte diesen Standard, so setzt bspw. Apple bereits seit September 2014 randomisierte MAC-Adressen ein [HDC18, S. 3]. Die vorherig beschriebene – sich nicht ändernde – MAC-Adresse wird in diesem Zuge als *öffentliche MAC-Adresse* abgegrenzt.

Ob es sich um eine öffentliche oder zufällige MAC-Adresse handelt, ist im zweiten Bit des ersten Bytes der Herstellerkennung kodiert. Wie in Abbildung 3.1 zu erkennen ist, handelt es sich um eine lokal generierte (*locally administred*) zufällige MAC-Adresse, wenn das Bit $b1$ gesetzt ist.

Öffentliche MAC-Adressen erlauben das Erkennen von wiederkehrenden Geräten sowie die Berechnung von Aufenthaltsdauern bspw. über einen Tag oder an bestimmten Stationen. Während in 2012 ausschließlich öffentliche MAC-Adressen genutzt wurden, ist der Anteil heute auf ca. ein Prozent gefallen[5]. Es ist davon auszugehen, dass dieser Anteil in den nächsten Jahren weiter schwindet. In dieser Arbeit finden öffentliche MAC-Adresse deshalb keine gesonderte Beachtung. Im Folgenden werden alle MAC-Adressen als randomisiert behandelt.

3.3. Randomisierte MAC-Adresse

Während die öffentliche MAC-Adresse noch in der Hardware hinterlegt ist, wird sie aber nicht publiziert. Nach außen tritt eine zufällig generierte (*randomisierte*) MAC-Adresse. Hierbei unterscheiden sich zwei Typen: statisch-zufällige und dynamisch-zufällige MAC-Adressen. Eine hierarchische Übersicht der Typen ist in Abbildung 3.2 dargestellt.

Statisch-zufällige MAC-Adressen erhalten ihre Gültigkeit über einen längeren unbestimmten Zeitraum (Tage oder Wochen). So wird sie bspw. bei jedem Neustart des Gerätes neu generiert. Eine statisch-zufällige MAC-Adresse kann unbegrenzt gesetzt werden. Der Unterschied zur öffentlichen liegt in diesem Fall in der Unabhängigkeit der Herstellerkennung. Anders als bei der Unterscheidung zwischen öffentlicher und randomisierter MAC-Adresse, gibt es kein Bit, das anzeigt, ob es sich um eine statisch- oder dynamisch-randomisierte MAC-Adresse handelt.

Dynamisch-zufällige MAC-Adressen werden alle zehn bis 20 Minuten erneuert. Die ablösende randomisierte MAC-Adresse lässt keinen Rückschluss auf die vorherige zu und vice versa. So wird das Verfolgen einer MAC-Adresse über dieses Zeitfenster von durchschnittliche 15 Minuten hinaus verhindert.

[5] Anteil Messungen mit öffentlicher MAC-Adresse an allen Messungen basierend auf den Auswertungen dieser Arbeit in Abschnitt 8.2.

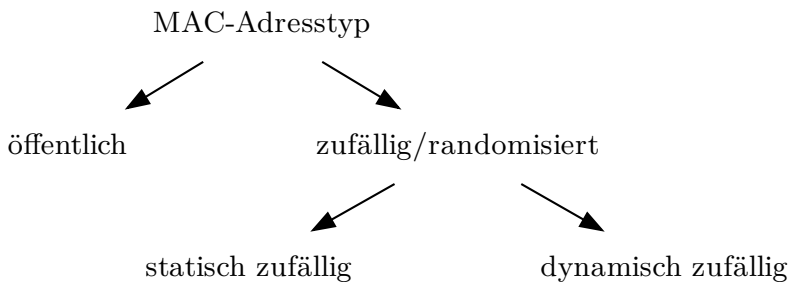


Abbildung 3.2.: Typen von MAC-Adressen. Die *öffentliche* MAC-Adresse ist unveränderbar in der Hardware hinterlegt. Zum Schutz der Privatsphäre werden *zufällige* MAC-Adressen generiert und veröffentlicht. Diese unterscheiden sich in *statisch-zufällige* und *dynamisch-zufällig* MAC-Adressen, wobei die statisch-zufälligen eine längere Gültigkeit (Tage/Wochen) besitzen als die dynamisch-zufälligen (Minuten).

Eine randomisierte MAC-Adresse dient im Rahmen dieser Arbeit zur Erkennung von Übergängen des einen Messbereichs zu einem anderen innerhalb des Zeitfensters der Gültigkeit. Aufenthaltsdauern einer MAC-Adresse innerhalb eines Messbereichs sind nicht aussagekräftig, da nicht zwischen dem Wechsel der MAC-Adresse und dem Verlassen des Messbereichs unterschieden werden kann. Anders als bei der öffentlichen MAC-Adresse lassen randomisierte keine Rückschlüsse auf den Hersteller zu. Zur weiteren Klassifizierung können die Bluetooth Metadaten *Manufacturer-Specific-Data*, *Service-Data* sowie der *Gerätename* herangezogen werden.

3.4. Klassifizierung von Bluetooth-Geräten

Vergleichbar mit der Registrierung der Herstellerkennung einer MAC-Adresse bei IEEE (vgl. Abschnitt 3.2) können bei der Bluetooth SIG Kennungen für *Manufacturer-Specific-Data* und *Service-Data* registriert werden.

In dem Feld der Manufacturer-Specific-Data können Hersteller eigene Protokolle entwickeln. Die ersten zwei Byte enthalten die Herstellerkennung (*Company Identifier*). Ist dieses Feld gesetzt, lässt sich – selbst bei einer randomisierten MAC-Adresse – der Hersteller des Bluetooth-Gerätes bestimmen. Bluetooth SIG stellt hierzu eine Liste bereit[6].

Die Service-Data spezifiziert, um welche Art Dienst bzw. Protokoll es sich bei der Bluetooth-Verbindung handelt. Die ersten zwei Bytes enthalten die UUID, anhand derer sich die Art bei Bluetooth SIG nachschlagen lässt[7]. Apple Inc. und Google LLC definieren für den *Expos-*

[6]Zuordnung der 16 Bit Herstellerkennung: `https://www.bluetooth.com/specifications/assigned-numbers/company-identifiers/`

[7]Zugewiesene Nummern: `https://www.bluetooth.com/specifications/assigned-numbers/`

ure Notification Service des ENF[8] die UUID `0xfd6f`[9] [vgl. AG20, S. 4, Exposure Notification Service]. Mithilfe der Service-Data kann ermittelt werden, welche Messungen der CWA entspringen.

Zur Klassifizierung des Gerätetyps kann das Feld Gerätename (*friendly name*) genutzt werden. Dieses enthält häufig Schlüsselwörter wie bspw. *Watch*, *Band* oder *Earbud*. Mithilfe von regulären Ausdrücken lässt sich der Gerätename klassifizieren, hier bspw. in *Smartwatch*, *Fitnesstracker* und *Kopfhörer*. Eine vollständige Liste der vorgenommenen Klassifizierung ist in Abschnitt A.3 zu finden.

3.5. Datenschutz und Anonymisierung der Bluetooth-Daten auf Grundlage des BDSG

Bei Standortdaten handelt es sich um personenbezogene Daten (vgl. Datenschutzgrundverordnung [DSGVO16, Artikel 4, Absatz 1] bzw. Bundesdatenschutzgesetz [BDSG18, §46]). Hier steht an erster Stelle der Grundsatz der *Zweckbindung*. Für die Besucherstromanalyse des Testgeländes ist nicht das Verhalten einer *einzelnen* natürlichen Person von Interesse, sondern das Verhalten einer *Menge* an Personen. Die Datenschutz-Grundverordnung berücksichtigt den Ausnahmefall des *statistischen Zwecks*: „[...Es] wird vorausgesetzt, dass die Ergebnisse der Verarbeitung zu statistischen Zwecken keine personenbezogenen Daten, sondern aggregierte Daten sind [...]“ ([vgl. DSGVO16, Grund 162]). Zu diesem Zweck ist die Datenerfassung ohne ausdrückliche Einwilligung möglich (vgl. [DSGVO16, Artikel 9, Absatz 2j], bzw. [BDSG18, §27, Absatz 1]).

Die *Informationspflicht* gegenüber jeder betroffenen Person entfällt für statistischen Zweck, wenn sie „einen unverhältnismäßigen Aufwand erfordern würde [...] In diesen Fällen ergreift der Verantwortliche geeignete Maßnahmen zum Schutz [...], einschließlich der Bereitstellung dieser Informationen für die Öffentlichkeit“ [DSGVO16, Artikel 14, Absatz 5b]. Für das Testgelände wird öffentlich auf der Internetseite über die Erhebung der Bluetooth-Daten zu statistischen Zwecken informiert. Eine geeignete Schutzmaßnahme bei der Übertragung von Daten bildet die TLS-Verschlüsselung [DSGVO16, Artikel 32, Absatz 1a].

Über die Möglichkeit des *Widerspruchs* der Datenerfassung ([DSGVO16, Artikel 12, Absatz 6]) wird ebenfalls auf der Internetseite informiert; Bluetooth abschalten.

Weiterhin gilt der Grundsatz der *Datenminimierung*. Hier wird eine Pseudonymisierung der personenbezogenen Daten [DSGVO16, Artikel 89, Absatz 1], falls möglich die Anonymisierung gefordert [BDSG18, §27, Absatz 3]. Dies soll so früh wie möglich nach der Datenerfassung erfolgen.

Im Falle einer öffentlichen MAC-Adresse kann der Hersteller anhand der Herstellerkennung abgeleitet werden. Außerdem dient die MAC-Adresse als Identifikator eines Gerätes. Diesen

[8] ENF: Exposure Notification Framework, siehe Abschnitt 3.1.

[9] Die UUID `0xfd6f` wurde von Apple bei der Bluetooth SIG registriert. Vgl. Eintrag auf S. 19 in: `https://btprodspecificationrefs.blob.core.windows.net/assigned-values/16-bit%20UUID%20Numbers%20Document.pdf`

Zweck erfüllt auch ein Hash der Adresse (im Folgenden *MAC-Hash* genannt). Von diesem ist kein Rückschluss auf die MAC-Adresse mehr möglich. Dennoch lässt sich mithilfe des Pseudonyms MAC-Hash eine Messung eines Gerätes identifizieren. Im Fall einer randomisierten MAC-Adresse gilt sie nach Ablauf der Gültigkeit als anonymisiert. Selbst unter Hinzunahme der neuen MAC-Adresse („zusätzlicher Informationen“ nach [BDSG18, §46.5]) kann nicht die alte und daraus nicht der MAC-Hash erzeugt werden.

$$mac \rightarrow macHash$$

Der Zuordnung eines Herstellers ($manufacturer$) dient das Feld Manufacturer-Specific-Data. Ob es sich um ein Advertisement der CWA ($isCwa$) handelt wird über die Service-Data ermittelt. Der Gerätename ($friendlyName$) wird für die Klassifizierung ($type$) des Bluetooth-Gerätes genutzt.

$$\begin{aligned} mac \mid manufacturerSpecificData &\rightarrow manufacturer \\ serviceData &\rightarrow isCwa \\ friendlyName &\rightarrow type \end{aligned}$$

Ist die jeweilige Zuordnung geschehen, können nach dem Prinzip der Datensparsamkeit diese drei volatilen personenbezogenen Daten $manufacturerSpecificData$, $serviceData$ und $friendlyName$ gelöscht werden.

$$\begin{aligned} data_{personal} &= \{\ mac\ manufacturerSpecificData\ serviceData\ friendlyName\ \}; \\ &\downarrow \\ data_{anonym} &= \{\ macHash\ manufacturer\ isCwa\ type\ \}; \end{aligned}$$

So wird $data_{personal}$ nach $data_{anonym}$ überführt und nur letzteres dauerhaft gespeichert.

3.6. Annahmen

Im Folgenden werden zwei Annahmen formuliert, die die technische Grundlage dieser Arbeit bestimmen. Das Testgelände umfasst eine Fläche von 22 Hektar. Eine exakte Positionsbestimmung ist nicht erforderlich, da Attraktionen einen großen Abstand haben. Im Außenbereich ist der Aufzeichnungsbereich eines Bluetooth-Loggers von ca. 24 Metern Radius ausreichend genau[10].

Außerdem wird angenommen, dass eine repräsentative Menge besuchender Personen erstens Bluetooth-Geräte bei sich führen und zweitens der Anteil an Geräten pro Besuchenden konstant ist. Dieser kann im Einzelfall unterschiedlich ausfallen, so bspw. im Vergleich einer Schulklasse mit den Teilnehmenden einer Kaffeefahrt. Für den gesamten Testzeitraum sollte ein repräsentativer Anteil ermittelt werden können. Unter diesen Voraussetzungen lassen sich auf Basis gemessener Geräte Besucherzahlen schätzen.

[10] In einem Museum, in dem Exponate nah beieinander stehen, ist eine genauere Lokalisierung bspw. mithilfe des Path-Loss Modells und Multilateration gefragt.

4. Systemarchitektur

In diesem Kapitel wird die Dreischichtenarchitektur des *Internets der Dinge* (*Internet of Things*, kurz *IoT*) vorgestellt. Diese wird um zwei Schichten erweitert und findet in Abschnitt 4.1 als Fünfschichtenarchitektur Einsatz. Nach der Beschreibung der einzelnen Schichten aus logischer Sicht, wird in die physische Sicht gewechselt (Abschnitt 4.2), indem das Deployment mit *Docker Compose* kurz beschrieben wird. Die auf Architekturebene berücksichtigten Schutzmaßnahmen der Systemsicherheit werden in Abschnitt 4.3 beschrieben.

Im IoT hat sich die Dreischichtenarchitektur etabliert [Wu+10]. Diese besteht – wie in Abbildung 4.1 dargestellt – aus der Wahrnehmungs- (*Perception*), der Netzwerk- (*Network*) sowie der Anwendungsschicht (*Application*). Dieser Dreischichtenarchitektur lassen sich die folgenden

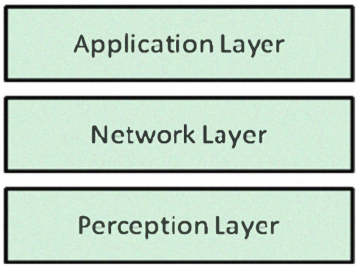


Abbildung 4.1.: Dreischichtenarchitektur des IoT. Die Wahrnehmungsschicht (*Perception*) beinhaltet Informationen sammelnde Sensorik, die Netzwerkschicht (*Network*) transportiert gesammelte Daten sicher zur Anwendungsschicht, die diese Daten verarbeitet (*Application*) [Wu+10].

drei Bausteinen des Versuchsaufbaus zuordnen:

1. Hardware, die im Testgeländer verteilt wird und Bluetooth-Messungen aufnimmt, sog. Bluetooth-Logger (Wahrnehmungsschicht),
2. ein sicheres Kommunikationsprotokoll zwischen Messgeräten und Server (Netzwerkschicht) sowie
3. ein Server, der Messdaten der Bluetooth-Logger sammelt, verarbeitet und die Ergebnisse bereitstellt (Anwendungsschicht).

4.1. IoT-Architektur

Die dritte Schicht (Anwendungsschicht) ist noch zu abstrakt formuliert. Deshalb definieren Wu u. a. eine neue Fünfschichtenarchitektur [Wu+10], die später als *die IoT-Architektur* angesehen wird [vgl. Kha+12; STJ15; GZY16]. Hier wird je eine weitere Schicht oberhalb und unterhalb der Anwendungsschicht hinzugefügt. Die Verarbeitungsschicht (*Processing*) liegt zwischen Netzwerk- und Anwendungsschicht. Ihre Aufgabe ist es, die eintreffenden Datenmengen der Wahrnehmungsschicht zu filtern, zu aggregieren, ggf. weiter zu verarbeiten und Ergebnisse in einer Datenbank abzulegen. Auf diese Weise ist die Anwendungsschicht loser gekoppelt und übernimmt ausschließlich die Bereitstellung der Daten für Benutzende.

Die Betriebsschicht (*Business*) ist das Management für das gesamte IoT-System. Sie tritt als Benutzende der Anwendungsschicht auf und ist als Präsentationsschicht für administrative Zwecke des IoT-Systems zu verstehen.

Abbildung 4.2 zeigt die zugeschnittene Systemarchitektur. Hier werden einzelne Bestandteile bzw. Dienste des Systems den fünf Schichten zugeordnet. Die klassische Präsentationsschicht ist in dem Modell von Wu u. a. nicht vorgesehen. Sie tritt als Konsument des Dienstes *API* der Anwendungsschicht auf.

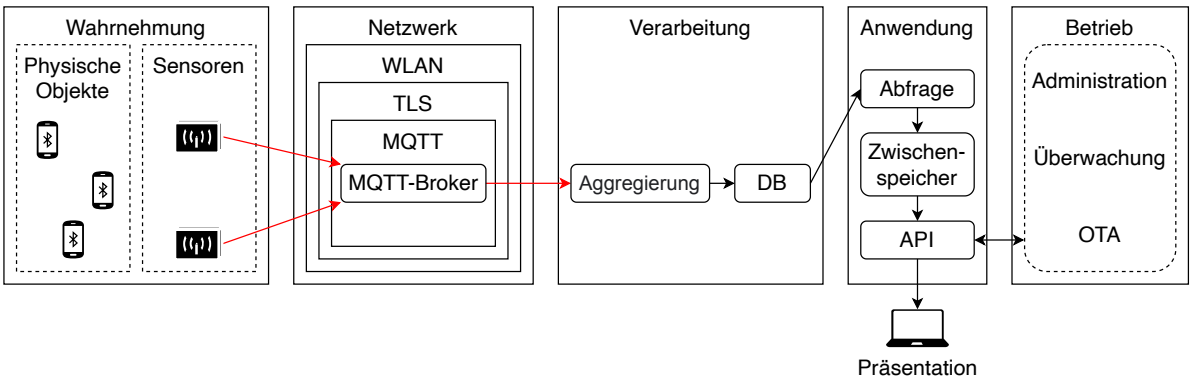


Abbildung 4.2.: Verwendete Systemarchitektur basierend auf der Fünfschichtenarchitektur des IoT. Die Dienste sind in abgerundeten Rechtecken dargestellt, Pfeile bilden den Datenfluss zwischen diesen. Rot markiert sind sicherheitskritische Datenflüsse.

Diese Unterteilung in Schichten bietet eine starke Kohäsion logischer Komponenten. Durch den strikten Datenfluss von links nach rechts wird eine lose Kopplung hergestellt. Im Folgenden wird jede Schicht einleitend beschrieben, die folgenden Kapitel behandeln jede Schicht ausführlich.

Wahrnehmungsschicht Die Wahrnehmungsschicht beginnt bei physischen Objekten, in diesem Fall Bluetooth-fähigen Geräten von Besuchenden im Testgelände [vgl. Kha+12, III. Generic Architecture]. Mithilfe von Sensoren (Bluetooth-Loggern) wird die Präsenz der Geräte aufgezeichnet über die Netzwerkschicht an die Verarbeitungsschicht übergeben. Da sich

die Bluetooth-Logger *am Rand* des Systemnetzwerks befinden, werden sie als *Edge-Geräte* bezeichnet. Kapitel 6 beschreibt den Einsatz des Mikrocontrollers ESP32 als Bluetooth-Logger.

Netzwerkschicht Die Bluetooth-Logger im Testgelände sind über WLAN mit dem Internet verbunden. Als Kommunikationsprotokoll wird TLS-verschlüsseltes MQTT eingesetzt. Die Funktion des *Edge-Gateways* übernimmt der MQTT-Broker. Hier werden eintreffende Messdaten angenommen und weiterverteilt. Da MQTT die Grundlage für die Funktion der Bluetooth-Logger der Wahrnehmungsschicht bildet, ist es in der Gliederung dieser Arbeit vorgezogen. Die Funktionsweise sowie die sicherheitskritische Zugriffssteuerung wird in Kapitel 5 erläutert.

Verarbeitungsschicht Ziel der Verarbeitungsschicht ist das Aufbereiten und Verfügbarmachen der Messdaten für die Anwendungsschicht. Im Dienst *Aggregierung* werden einzelne Messungen zunächst anonymisiert und anschließend in Zeitfenstern zusammengefasst. Dies dient der Datenreduzierung und eine Datenstruktur wird für die spätere Auswertung erstellt. Resultierende Daten werden in einer Datenbank des Dienstes *DB* persistiert. Vgl. Kapitel 7: Datenspeicherung.

Anwendungsschicht Da die Daten bereits in einer geeigneten Datenstruktur in der Datenbank vorliegen, kann der Dienst *Abfrage* der Anwendungsschicht diese anfragen und kombinieren. Resultierende Auswertungen werden in Kapitel 8 beschrieben. Die Anwendungsschnittstelle des Dienstes *API*[1] ist zentraler Ausgang für aufbereitete Daten. Diese nutzen sowohl die Betriebsschicht als auch jedwede Präsentationsschicht[2]. Durch die Entkopplung der Präsentationsschicht bleibt das System unabhängig, aufbereitete Daten können über die Anwendungsschnittstelle flexibel in jedwede Applikation eingebunden werden (bspw. App, Website, etc.). Um aufwendige Anfragen oder Berechnungen einzusparen, kann der Dienst *Zwischenspeicher* in Anspruch genommen werden.

Betriebsschicht Die Aufgaben der Betriebsschicht lassen sich in drei Bereiche unterteilen:

- Administrative Einstellungen meinen die Datenpflege der Standorte der Bluetooth-Logger (Erhebung in Abschnitt 6.4). Dabei durchbricht die Betriebsschicht als einzige die strikte Schichtenarchitektur. Hier erfolgt ein Datenfluss von rechts nach links.
- Unter „Gesundheitsüberwachung“ der Bluetooth-Logger fällt einerseits eine Auswertung über die Verfügbarkeit der Bluetooth-Logger (vgl. Abschnitt 8.1). Andererseits können aus den Statusnachrichten der Bluetooth-Logger Verbindungsstatus und Firmware-Version entnommen werden.
- Das Durchführen und Überwachen kabelloser Firmware-Updates (*OTA*) der Bluetooth-Logger (beschrieben in Unterabschnitt 6.2.2).

Im weiteren Verlauf dieser Arbeit wird die Betriebsschicht nicht erneut beschrieben. Ein kurzer Einblick in die Betriebsschicht ist in Abschnitt A.2 ausgelagert.

[1] API kurz für *Application Programming Interface*, zu deutsch Anwendungsschnittstelle.

[2] Da die Präsentationsschicht nur als Konsument des Dienstes *API* auftritt, ist sie nicht Teil der IoT-Architektur und findet keine gesonderte Betrachtung.

4.2. Physikalische Topologie

Die Schichtenarchitektur trennt das System in logische Komponenten. In der Implementierung werden diese Komponenten auf physische Maschinen verteilt und stellen so die physikalische Topologie dar. So werden die Bluetooth-Logger als Wahrnehmungsschicht im Testgelände installiert. Die Anwendungs- und Betriebsschicht wird in ein Bestandssystem integriert. Dieses besteht aus einem *C#-Backend*, welches mittels *Entity Framework* SQL-Abfragen an die Datenbank stellt (*Abfrage*), die Ergebnisse ggf. in einer lokalen *Postgres-Datenbank* zwischenspeichert (*Zwischenspeicher*) und mit *GraphQL* die *API* darstellt. Die Präsentationsschicht ist zusammen mit der Betriebsschicht in *React* implementiert. Da es sich um proprietäre Software handelt, wird die Implementierung dieser beiden Schichten hier nicht detaillierter betrachtet. Dienste der Netzwerk- und Verarbeitungsschicht werden auf einem Server betrieben und im Folgenden genauer erläutert.

Das Konzept *Infrastructure as Code* (kurz IaC) sieht die automatische Konfiguration des Technologie-Stacks und Netzwerks einer Anwendung durch Dateien vor. So wird anstelle der manuellen Installation von Abhängigkeiten, einem Deployment-Tool eine Liste von Abhängigkeiten übergeben, die automatisch installiert werden. Die Konfiguration in Dateien bietet Vorteile der Versionskontrolle, spart manuelle Serverkonfiguration und erleichtert die Migration von Diensten von einer Maschine auf die andere.

Für die Dienste *MQTT-Broker*, *Aggregierung* und *DB* wird das Deployment-Tool *Docker Compose* eingesetzt. Docker Compose verwaltet Einstellungen und Netzwerke von Docker-Containern. Ein Auszug der Konfigurationsdatei `docker-compose.yaml` ist in Listing 4.1 dargestellt. Das GitHub-Repository, das die vollständige Datei enthält, ist in Abschnitt A.1 verlinkt. Unter `services` finden sich die drei Dienste `mqtt_broker`, `db` und `data_aggregation`, dabei bleibt die logische Trennung erhalten.

Listing 4.1: Auszug der `docker-compose.yaml`.

```yaml
version: 3.7
services:
  mqtt_broker:
    image: eclipse-mosquitto:2.0.14
    ports:
      # expose port to the internet, check firewall rules
      - 8883:8883
    volumes:
      - ./mqtt_broker:/mosquitto/config:ro

  db:
    image: mariadb:10.7.3
    volumes:
      - ./volumes/db/mariadb_data:/var/lib/mysql:rw

  data_aggregation:
    build: ./data_aggregation/
```

Unter der Konfiguration des MQTT-Brokers finden wir zunächst das Docker-Basisimage. Dieses beinhaltet eine Open Source Implementierung des MQTT-Protokolls. Unter Ports wird der Port

des Hosts dem des Containers zugeordnet. Dies bewirkt, dass Anfragen auf Port 8883 des Hosts an den Container weitergeleitet werden. Unter `volumes` wird das Verzeichnis `./mqtt_broker` in den Container eingehängt. In diesem Fall enthält dieses Verzeichnis Konfigurationsdateien für den MQTT-Broker.

Als zweiter Dienst ist die Datenbank angegeben, die das Basisimage der *MariaDB* nutzt. In dem unter `volumes` angegebenen Verzeichnis wird die Datenbank angelegt. Backups der Datenbank lassen sich so bequem vom Host aus realisieren, indem das relative Verzeichnis `./volumes/db/mariadb_data/` rekursiv kopiert wird.

Der dritte Dienst ist ein eigenes Docker-Image, dessen Bauanleitung im Unterverzeichnis `./data_aggregation/` zu finden ist. In diesem Verzeichnis liegen neben dem *Dockerfile*, das ebenfalls nach dem Konzept IaC aufgebaut ist, das Python-Programm, welches die Datenaggregierung und -speicherung implementiert. Alle Dienste werden seitens Docker Compose mit einem virtuellen Netzwerk verbunden. So erreicht der Dienst *Aggregierung* die Datenbank über den Hostnamen `db`.

Nach der Definition der `docker-compose.yaml` ist keine weitere Administration erforderlich. Der Kommandozeilenbefehl `docker-compose up` liest die Konfigurationsdatei ein und startet die Container. Sobald alle Container initialisiert wurden, ist der MQTT-Broker für den Nachrichtenempfang über Port 8883 bereit und leitet eingehende Nachrichten ggf. an den Dienst *Aggregierung* weiter. Dieser verarbeitet die Messdaten und speichert sie in der Datenbank ab. Obwohl die Dienste auf einem Host laufen, bleibt die logische Trennung erhalten. Flexibel können Dienste hinzugefügt oder entfernt werden. Eine Migration wird ermöglicht durch ein Backup des Wurzelverzeichnisses der Datei `docker-compose.yaml`.

4.3. Secure by Design

Das Entwurfskonzept *Secure by Design* (dt. Sicherheit ab Entwurf) sieht die Berücksichtigung der Schutzziele der IT-Sicherheit bereits in der Entwurfsphase vor. Diese fünf Schutzziele lassen sich durch das Akronym *CIA*^{3} zusammenfassen und beziehen sich hier auf die Messdaten der Bluetooth-Logger:

- **C**onfidentiality (Vertraulichkeit),
- **I**ntegrity (Integrität),
- **A**vailability (Verfügbarkeit),
- **A**uthenticity (Authentizität) und
- **A**ccountability (Nachweisbarkeit).

Oft wird im Kontext der IT-Sicherheit die Privatsphäre (Privacy) als Schutzziel genannt. Diese wird durch die Vertraulichkeit und Authentizität hergestellt.

Sicherheitskritische Datenflüsse sind in Abbildung 4.2 rot markiert. Über diese fließen Daten in das System. Die Vertraulichkeit dieser Daten wird mithilfe von TLS-Verschlüsselung geschützt. Ebenso schützt TLS die Authentizität. Mittels TLS-Zertifikaten können Kommunikationspartner den MQTT-Broker authentifizieren (siehe Abschnitt 5.2). So wird einem *Man-in-the-Middle-Angriff* vorgebeugt und die Integrität der Daten geschützt. Die seitens der DSGVO geforderte Schutzmaßnahme der Privatsphäre wird somit umgesetzt.

Im MQTT-Broker sind Rollen verteilt. Der MQTT-Broker ordnet dabei jedem Bluetooth-Logger einen Identifikator zu und stellt so die Nachweisbarkeit der Herkunft der Daten her. So dürfen Bluetooth-Logger ihre Messdaten senden aber nicht die der anderen empfangen (siehe Abschnitt 5.1). Dies schützt die Vertraulichkeit der Daten anderer Bluetooth-Logger. Sollte ein Bluetooth-Logger böswillig entwendet und modifiziert werden, könnte ein potentieller Angreifer verfälschte Messdaten in das System schleusen. Anhand des Identifikators können verfälschte Messdaten gefiltert und die Integrität wiederhergestellt werden.

Daten verlassen das System über den Dienst *API*. Diese sind bereits anonymisiert und aggregiert und fallen nicht mehr unter den Schutz der DSGVO. Die Implementierung sieht hier dennoch eine Verschlüsselung sowie Authentisierung des Nutzenden vor. Nur autorisierte Nutzende können Daten dieser Schnittstelle empfangen. Änderungen an der Datenbank können zum Schutz der Integrität von außen nicht vorgenommen werden.

Bezüglich der Verfügbarkeit ist der Server ein *Single Point of Failure* (dt. einzelner Ausfallpunkt). Mithilfe der in Abschnitt 4.2 beschriebenen Migration, wird der Austausch des Servers erleichtert. Ein Backup kann mithilfe von IaC auf einem anderen physischen Server schnell ausgerollt und gestartet werden, ohne diesen zuvor zu konfigurieren.

5. Kommunikationsprotokoll MQTT

MQTT (kurz für Message Queuing Telemetry Transport) ist der De-facto-Standard für M2M-Kommunikation[1] im IoT. Nach Google Trends ist es seit Jahren das populärste Protokoll seiner Klasse[2]. Im Vergleich zu HTTP ist MQTT viermal so schnell und weniger rechenintensiv [vgl. GŞB21, S. 547]. MQTT setzt auf TCP auf und unterstützt somit verschlüsselte Kommunikation über TLS.

Vergleichbar mit einem Postamt übernimmt der *MQTT-Broker* die zentrale Position. Hier treffen alle Nachrichten unter *Topics* ein und werden an *Clients* verteilt, die sie *abonnieren*. Es gibt einen Broker und beliebig viele Clients.

Eine MQTT-Nachricht besteht aus einem hierarchischen MQTT-Topic sowie den Nutzdaten. Das Topic klassifiziert dabei die Nachricht hierarchisch. So handelt es sich bspw. bei dem Topic `sensor/BLE/Scanner/5`, um eine Nachricht des Bluetooth-Loggers mit der laufenden Nummer fünf. Das Trennzeichen für die Klassifizierung ist dabei der Schrägstrich. Mit sog. *Wildcards* können Abonnements für generische Topics abgeschlossen werden. So passt bspw. das Topic `sensor/#` auf alle Nachrichten, deren Topic mit `sensor/` beginnen.

Ein MQTT-Client kann sich als Produzent (*publisher*) und/oder Konsument (*subscriber*) bei dem Broker registrieren. Als Produzent treten hier die Bluetooth-Logger auf, die Messdaten senden (Aktion *publish*). Konsumenten sind hier Dienste, die Messdaten verarbeiten. Dafür abonnieren (Aktion *subscribe*) sie Nachrichten eines oder mehreren Topics. Beim Eintreffen einer Nachricht beim MQTT-Broker, sendet sie dieser an alle abonnierenden Clients weiter. Jeder Client wird vom Broker über einen Identifikator (*client id*) verwaltet. Sendet der Client beim ersten Verbindungsaufbau keinen Identifikator, wird vom Broker einer erstellt.

Eine besondere Nachricht ist die sog. *Will-Message*. Ein Client kann bei dem Broker seinen „letzten Willen“ hinterlegen, eine Nachricht, die gesendet wird, sollte der Client nicht mehr erreichbar sein. Diese Nachricht findet Einsatz bei der Überwachung der Bluetooth-Logger.

Mithilfe von MQTT müssen Clients nur die IP des Brokers, nicht aber die der Konsumenten kennen. Wird die IP des Brokers über eine URL aufgelöst, ist der Broker in der Lage die IP zu wechseln, ohne dass Änderungen an Clients nötig sind. Der Gewinn ist hier mehr Flexibilität durch lose Kopplung.

[1] M2M ist kurz für *Machine-to-Machine* und bezeichnet den Informationsaustausch zwischen technischen Geräten ohne menschliche Beteiligung (vgl. Definition der Bundesnetzagentur unter `https://www.bundesnetzagentur.de/DE/Fachthemen/Telekommunikation/Nummerierung/M2M/M2M_node.html`).

[2] Vgl. Google Trends Vergleich von MQTT (83%), *AMQP* (9%) und *XMPP* (7%) unter: `https://trends.google.de/trends/explore?q=mqtt,amqp,xmpp`

Definition eingesetzter Topics Um Messdaten an den Server zu senden wird das *Sensor-Topic* definiert:

`sensor/BLE/Scanner/<id>`

Die hierarchische Klassifizierung ermöglicht die Erweiterung des Systems um Messgeräte anderer Technologien. Für die Überwachung der Bluetooth-Logger seitens der Betriebsschicht wird das *Admin-Topic* eingeführt:

`admin/BLE/Scanner/<id>`

Hier senden Bluetooth-Logger einerseits eine Nachricht nach erfolgreichem Verbindungsaufbau, andererseits wird die *Will-Message* über dieses Topic veröffentlicht. Durch diese beiden Nachrichten kann der Verbindungsstatus eines Bluetooth-Loggers überwacht werden.

Neben den zwei genannten Topics wird ein weiteres in anderer Richtung (Server zu Bluetooth-Logger) implementiert; das *OTA-Topic*:

`ota/BLE/Scanner/<id>`

Dieses Topic abonnieren Bluetooth-Logger und können so über Firmware-Updates informiert werden.

5.1. MQTT-Zugriffssteuerung

Der hier eingesetzte MQTT-Broker ist öffentlich erreichbar. Mithilfe der *Zugriffssteuerungsliste* (*Access Control List*, kurz ACL) wird dafür gesorgt, dass autorisierte MQTT-Clients Daten schreiben bzw. lesen. Zur Authentisierung setzt MQTT Nutzername-Passwort-Paare analog der UNIX Benutzerverwaltung ein. Listing 5.1 zeigt die verwendete MQTT-ACL. Eine Nutzerrolle (Schlüsselwort `user`) bezeichnet hier eine Gruppe von Clients, die sich mit Nutzername und Passwort authentisiert haben. Der Identifikator eines Clients kann mithilfe der Variablen `%c` generisch behandelt werden.

Nach der ersten Regel der ACL darf bspw. Bluetooth-Logger mit der Nummer fünf mit der authentifizierten Nutzergruppe `sensor_ble` auf *seinem* Sensor-Topic `sensor/BLE/Scanner/5` schreiben, da er sich mit dem Identifikator „5“ beim Broker registriert hat (`%c`=5). Der Regelmodifikator `write` schließt dabei ein Lesen dieses Topics aus. So wird sichergestellt, dass ein Bluetooth-Logger nur auf dem ihm zugeordneten Topic Daten senden kann. Die zweite Regel erfolgt analog der ersten für das Admin-Topic.

Listing 5.1: Auszug der Zugriffssteuerungsliste (ACL) des MQTT-Brokers. Bluetooth-Logger der Gruppe `sensor_ble` dürfen nur auf ihren Sensor-Topics schreiben, aber nicht lesen, Konsumenten der Gruppe `reader_ble` dürfen Sensor-Topics nur lesen, aber nicht schreiben.

```
1  # All Bluetooth-Logger are allowed to write to their topics only, no more.
2  # %c is the client id
3  user sensor_ble
4  topic write sensor/BLE/Scanner/%c
5  topic write admin/BLE/Scanner/%c
6
7  user reader_ble
8  topic read sensor/BLE/Scanner/#
```

```
9     topic read admin/BLE/Scanner/#
```

Der zweite Regelblock in Listing 5.1 betrifft die Konsumenten der Messdaten. Die Nutzerrolle `reader_ble` darf sowohl die Sensor- als auch die Admin-Topics lesen, aber nicht schreiben. Das *Prinzip der geringsten Privilegien* (*Least Privilege*) findet hier Anwendung, einerseits um die Datenintegrität zu schützen (*Secure by Design*), zum anderen um die Schichtenarchitektur einzuhalten.

5.2. TLS Zertifikat für MQTTS

Analog zu HTTP wird MQTT über TLS als *MQTTS* bezeichnet. In erster Linie schützt die TLS-Verschlüsselung die Vertraulichkeit der Messdaten auf dem Weg zum Server. Des Weiteren dient der Einsatz von TLS zur Authentifizierung des MQTT-Brokers.

Das *CA/Browser Forum*[3] empfiehlt eine maximale Gültigkeitsdauer von 13 Monate für TLS-Zertifikate. Einmal pro Jahr wird das Zertifikat des MQTT-Brokers ausgetauscht. Für die Authentifizierung wird das Zertifikat auch an die Clients verteilt. Um nicht bei jedem Zertifikatstausch alle Clients zu aktualisieren, werden mehrere Zertifikate genutzt:

- Zunächst wird ein *Wurzelzertifikat* erstellt. Dieses hat eine Gültigkeit von 20 Jahren und wird an alle Clients verteilt. Der private Schlüssel des Wurzelzertifikats wird an einem sicheren Ort abgelegt.
- Für den MQTT-Broker wird ein weiteres Zertifikat erstellt, das *Serverzertifikat*. Dieses hat eine kurze Gültigkeit von 13 Monaten und wird mit dem privaten Schlüssel des Wurzelzertifikats signiert.

Durch diese Indirektion ist es möglich, das Serverzertifikat – oder den gesamten Server – auszutauschen. Clients akzeptieren das neue Zertifikat, wenn es erstens gültig und zweitens mit dem privaten Schlüssel des Wurzelzertifikats signiert wurde.

[3] Das CA/Browser Forum formuliert als Zusammenschluss von Zertifizierungsstellen und Browser-Entwicklern folgende Mindestanforderungen an Zertifikate: *Baseline Requirements for the Issuance and Management of Publicly-Trusted Certificates*, Version 1.8.4, Abschnitt 6.3.2. Zu finden unter `https://cabforum.org/wp-content/uploads/CA-Browser-Forum-BR-1.8.4.pdf`

6. Datenakquise

Im Folgenden wird die Funktionsweise des *Bluetooth Exposure Logging* erläutert. Abschnitt 6.1 beschreibt die Bauteile der Bluetooth-Logger, die Sensoren der Wahrnehmungsschicht darstellen. In Abschnitt 6.2 wird eingesetzte Firmware sowie die Möglichkeit des kabellosen Updates beschrieben. Resultierende Rohdaten aus Abschnitt 6.3 werden über MQTT an die Netzwerkschicht weitergegeben. Wo die Bluetooth-Logger im Testgelände aufgestellt werden, ist in Abschnitt 6.4 festgehalten.

6.1. Hardware: ESP32 als Bluetooth-Logger

Die CWA sendet BLE-Advertisements (Bluetooth 4.2). Mindestens diesen Bluetooth-Standard muss die Hardware beherrschen, um CWA-Daten erfassen zu können. Zum senden der Messergebnisse wird WLAN benötigt. Da die Technologien Bluetooth und WLAN dasselbe Band von 2,4 GHz nutzen, muss die Koordination der Koexistenz erprobt sein. Für die geforderte Verschlüsselung muss der Speicher ausreichend groß sein, damit eine TLS-Bibliothek mit Wurzelzertifikat eingesetzt werden kann. In [Sop+21] sowie [Ws22] wird für ähnliche Anforderungen ein ESP32 eingesetzt.

Die ESP32-Serie wurde im Jahr 2016 von der Firma *Espressif* eingeführt. Mikrocontroller dieser Serie beherrschen WLAN und BLE in Koexistenz und verfügen über ausreichend Arbeits- und Flashspeicher[1].

Für diese Arbeit wurde eine Entwicklungsplatine des ESP32-WROOM-32U[2] eingesetzt. Dieser unterstützt eine externe Antenne, die den Empfangsbereich im Vergleich zur standardmäßig verbauten PCB-Antenne[3] erhöht. Mit Stromversorgung, Antenne und Gehäuse ausgestattet liegen die Kosten pro Bluetooth-Logger zur Zeit bei 20 € (vgl. Abbildung 6.1).

Jedem Bluetooth-Logger wird eine Nummer zugewiesen und im Flashspeicher dauerhaft abgelegt. Beim Aufstellen wird der genaue Standort des Bluetooth-Loggers notiert. So kann, bei der späteren Datensammlung, jeder Messung ein Standort zugeordnet werden. Abbildung 6.1 zeigt den verkabelten Bluetooth-Logger der laufenden Nummer 11.

[1] Kurzer Überblick: Doppelkernprozessor mit bis zu 240 MHz, 520 KB RAM, 4 MB Flash.

[2] Datenblatt verfügbar unter:
`https://www.espressif.com/sites/default/files/documentation/esp32-wroom-32d_esp32-wroom-32u_datasheet_en.pdf`

[3] PCB: Kurz für *printed circuit board* (Leiterplatte). Eine PCB-Antenne ist werksseitig in der Leiterplatte eingebettet.

Position	Kosten
ESP32-WROOM-32U Entwicklungsplatine	8 €
Spannungsstabiles 2A Micro-USB Netzteil	5 €
2,4 GHz IPEX-Antenne mit 6 dbi Gewinn	6 €
Spritzwassergeschützten Abzweigdose als Gehäuse	1 €
$\sum$	20 €

Abbildung 6.1.: Kosten eines ESP32-Bluetooth-Logger bestehend aus Entwicklungsplatine, Stromversorgung, externer Antenne und Gehäuse.

6.2. Firmware

Die Firmware ist in C++ geschrieben. Der Quellcode ist der Übersichtlichkeit halber als Nur-Header-Bibliothek (Header-only) angelegt und teilt sich nach Semantik in die folgenden sieben Dateien[4] auf:

- `main.cpp`: Einstiegspunkt und zentrale Datei. Hier werden alle Dateien inkludiert und deren Zuständigkeit logisch verteilt. Einige Funktionen aus anderen Dateien werden hier vorwärts deklariert, damit Abhängigkeiten zur `main.cpp`, nicht aber zu anderen Quelldateien, existieren.

  Während des Boot-Vorgangs wird ein sog. *Watchdog-Timer* mit 100 Sekunden initialisiert. In der Hauptschleife wird dieser Timer wiederkehrend zurückgesetzt („*feed the watchdog*“). Läuft der Timer ab, wird der ESP neugestartet. Damit wird sichergestellt, dass unerwartete Programmzustände nach Ablauf des Timers zurückgesetzt werden.

- `globals_kd.h`: Enthält Konfigurationsparameter, die von jeder anderen Datei genutzt werden. Hier wird der Gerätename gesetzt, WLAN und MQTT Zugangsdaten hinterlegt, Bluetooth Einstellungen vorgenommen, MQTT Topics definiert und das TLS-Wurzelzertifikat bereitgestellt.

- `get_time.h`: Nachdem WLAN verfügbar ist, wird die Systemzeit mit einem Zeitserver synchronisiert. Zusätzlich werden Funktionen für das Abrufen der aktuellen Systemzeit in verschiedenen Formaten angeboten.

- `mqtts.h`: Der Verbindungsaufbau zum MQTT-Broker und das Behandeln von Verbindungsfehlern ist hier implementiert. Zum Senden von Nachrichten auf dem Admin- und Sensor-Topics wird eine Funktion zur Verfügung gestellt. Eintreffende Nachrichten auf dem OTA-Topic werden angenommen und validiert. Nach dem Verbindungsaufbau wird die Statusnachricht versendet und die *Will-Message* registriert, die der MQTT-Broker, im Falle eines Verbindungsabbruchs, veröffentlicht.

[4] Das GitHub-Repository, das den vollständigen Quellcode enthält, ist in Abschnitt A.1 verlinkt.

- `ble.h`: Das Suchen (engl. scan) nach Geräten wird hier umgesetzt. Für zehn Sekunden wird nach neuen Geräten gesucht. In einer Callback-Funktion wird für jedes neu gefundene Gerät ein JSON-String erzeugt. Dieser wird über die von `main.cpp` vorwärts deklarierte Funktion `transmitSensorsData` versendet. Innerhalb der zehn Sekunden Scan-Zeit werden wiederkehrende Advertisements derselben MAC-Adresse nicht erneut gesendet. Advertisements der CWA werden zwar mehrmals pro Sekunde empfangen, aber nur einmal pro zehn Sekunden versendet. Hier findet eine erste dezentrale Aggregierung der Messdaten statt, die die Bezeichnung *Edge-Computing* rechtfertigt.
- `ota.h`: Es werden zwei Zustände gehalten, ob ein Update verfügbar ist oder ob ein Update bereits in Arbeit ist. Eintreffende Nachrichten über ein Firmwareupdate werden hier verarbeitet. Nach einem Vergleich der Versionsnummern wird die Nachricht entweder ignoriert oder das Update durchgeführt. Hierzu wird die neue Firmware über HTTPS heruntergeladen und installiert.
- `led_blink.h`: Die eingebaute LED soll Fehlercodes codieren. In einem `enum` sind sprechende Werte für WLAN- bzw. BLE-Fehler hinterlegt. So blinkt die LED dreimal, wenn keine WLAN-Verbindung hergestellt werden konnte, viermal, wenn ein MQTT-Fehler vorliegt.

Der Quellcode ist nach dem Prinzip der losen Kopplung gegliedert, sodass sowohl die Messtechnik (Bluetooth) oder das Kommunikationsprotokoll (MQTT) ausgetauscht werden könnten und Änderungen an der `main.cpp`, nicht aber an den anderen Dateien notwendig werden. So implementiert `main.cpp` bspw. die Funktion

```
bool transmitSensorsData(const char *msg) ,
```

die wiederum die Funktion

```
bool sendMessage(const char *msg, bool admin = false)
```

aus `mqtts.h` aufruft. Eine Nachricht über ein gefundenes Bluetooth Gerät in `ble.h` erfolgt über die Funktion `transmitSensorsData` in `main.cpp` und wäre von einer Änderung des Kommunikationsprotokoll nicht betroffen.

6.2.1. Firmware initial aufspielen

Im ersten Schritt wird die vergebene laufende Nummer des Bluetooth-Loggers in den dauerhaften Speicher geschrieben (auch *gebrannt*). Ist das erfolgt, kann die Firmware eingespielt werden. Die eingebrannte Nummer wird zu Beginn des Programmablauf gelesen und den Topics angehängt. Sollte im Speicher keine Nummer hinterlegt sein, werden die letzten 5 Stellen der MAC-Adresse des ESPs als eindeutige Kennung verwendet. So können alle Bluetooth-Logger mit derselben kompilierten Firmware beschrieben werden und sind dennoch zu unterscheiden. Das hat sowohl beim Kompilieren der Binärdateien Vorteile durch die Zeitersparnis, als auch beim Verteilen neuer Firmware mittels OTA.

6.2.2. OTA-Firmware-Update

OTA steht für „Over the Air“ und meint, dass Updates kabellos eingespielt werden können. Da die TLS-Verbindung viel Hauptspeicher benötigt, wurden zwei Zustände im Programm eingeführt; *normal* und *OTA*. In der Regel wird die normale Schleife durchlaufen, die fortlaufend nach Bluetooth-Geräten scannt. Wird eine OTA-Nachricht empfangen, wird die angegebene Versionsnummer mit der aktuellen Firmwareversion verglichen. Listing 6.1 zeigt eine OTA-Nachricht für die Version 1.1.20. Ist die Version in der Nachricht höher, wird in den Zustand *OTA* gewechselt. Dieser wird zunächst initialisiert. Hier wird der Speicher der Bluetooth-Messung freigegeben ohne den der TLS-Handshake nicht erfolgen könnte. Außerdem wird der Watchdog-Timer auf fünf Minuten angehoben, sodass dieser nicht vorzeitig das Update unterbricht. Nun wird das OTA-Update gestartet und die neue Firmware wird über HTTPS von der in der OTA-Nachricht angegeben URL heruntergeladen.

Listing 6.1: Inhalt einer OTA-Nachricht des OTA-Topics `ota/BLE/Scanner/<id>`.

```
{
  "version": "1.1.20",
  "url": "https://domain.to.bin.files:2008/firmware-1-1-20.bin"
}
```

Die kompilierte Firmware darf nicht größer sein, als die Hälfte des verfügbaren Speichers. Für OTA-Updates wird der Speicher so partitioniert, dass in der einen Hälfte die aktuell laufende Firmware liegt und die andere frei ist, um neue Firmware einzuspielen. Nach einem Update folgt der Neustart von der anderen Partition mit der neuen Firmware, der Speicher der alten wird freigegeben.

Kommt es beim Update zu einem Fehler, wird darüber per MQTT auf dem Admin-Topic berichtet. Den Zustand *OTA* verlässt der ESP durch einen Neustart. War das Update erfolgreich, startet der ESP mit der neuen Software. Gab es einen Fehler, wird die vorherige gestartet. Bei jedem Start wird eine Statusnachricht auf dem Admin-Topic gesendet (siehe Listing 6.2). Diese enthält die aktuelle Firmware-Version. So kann der Erfolg des Updates überprüft werden.

Listing 6.2: Inhalt einer Statusnachricht des Admin-Topics `ota/BLE/Scanner/<id>`.

```
{
  "status": "online",
  "firmware": "1.1.20"
}
```

6.2.3. HTTPS-Server für Firmware

Die Verwendung von TLS beim Update-Prozess erfüllt zwei Zwecke. Zum einen wird die Vertraulichkeit der Firmware durch die Verschlüsselung geschützt, zum anderen kann der ESP den Server authentifizieren. Wurde das Serverzertifikat nicht von dem privaten Schlüssel des Wurzelzertifikats – das im ESP hinterlegt ist – signiert, wird der Update-Prozess abgebrochen. So wird sichergestellt, dass nur Firmware von *autorisierten* Servern geladen wird.

Zur Erstellung des weiteren Serverzertifikats wird der private Schlüssel des Wurzelzertifikats aus Abschnitt 5.2 erneut genutzt. So kann der ESP dasselbe Wurzelzertifikat für beide Verbindungen nutzen, wobei die Serverzertifikate abweichen.

Das Zusammenspiel aus eingebrannter Nummer im Speicher des ESPs, den OTA-Nachrichten und dem HTTPS-Server ermöglicht flexible Update-Strategien. Es wird *eine* Firmware für potentiell *alle* ESPs bereitgestellt. Mithilfe der OTA-Nachrichten lassen sich ESPs einzeln updaten. Der Erfolg des Updates kann über die Statusnachricht bestimmt werden. Ein stufenweises Ausrollen neuer Firmware ist ebenso möglich wie ein Stapelupdate.

6.3. Rohdaten

Eine Beispielmessung findet sich in Listing 6.3. Aus dem Identifikator im Sensor-Topic lässt sich ableiten, dass die Messung von Bluetooth-Logger *2* stammt. Der Zeitstempel (*timestamp*) gibt an, wann das Advertisement dieser MAC-Adresse empfangen wurde. Die drei optionalen Felder Manufacturer-Specific-Data, Service-Data und der Gerätename dienen der Klassifizierung (vgl. Abschnitt 3.4).

Listing 6.3: Inhalt einer Messungsnachricht des Sensor-Topics `sensor/BLE/Scanner/2`.

```
{
  "mac": "88:36:cf:xx:xx:xx",
  "timestamp": "1662898091"
  "manufacturer-specific-data": "7d02010300ffcc[...]",
  "service-data": "",
  "name": "HUAWEI Band [...]",
}
```

6.4. Platzierung im Testgelände

Das Aufstellen der Bluetooth-Logger stellte sich als schwieriger heraus als erwartet. Zunächst war angedacht, die Bluetooth-Logger in gleichmäßigem Abstand entlang des Rundwegs zu platzieren. Dies war nicht möglich. Die Positionen sind an folgende fünf Bedingungen geknüpft:

- Bluetooth-Logger sollten strategisch gut platziert sein, um bspw. den *Rundgang* abzudecken,
- sie sollten *versteckt* sein, damit sie nicht entwendet oder beschädigt werden,
- der Aufstellort sollte die Bluetooth-Logger vor *Wettereinfluss* schützen,
- eine 230V *Steckdose* ist von Nöten und
- sie benötigen *WLAN-Empfang*, um Messdaten an den MQTT-Broker zu schicken.

Abbildung 6.2.: Standorte der Bluetooth-Logger im Testgelände entlang des vereinfachten Rundgangs (blau). Der Ein- und Ausgang an den Positionen *6* und *8* stellt den Start und das Ende des Rundgangs dar.

Die Karte in Abbildung 6.2 zeigt die Positionen der Bluetooth-Logger im Testzeitraum. Mit den Punkten *6* und *8* wird der Ein- und Ausgang des Testgeländes voll abgedeckt. Folgt man dem blau markiertem Rundgang, erreicht man die Bluetooth-Logger *9* und *2*. Hier ist erstmals wieder WLAN-Empfang gegeben, aber nur in einem kleinen Radius um die Position *2*. Der erwünschte größere Abstand zwischen *9* und *2* ließ sich deshalb nicht herstellen. Bei den Positionen *7* und *5* ist eine gute WLAN-Abdeckung gegeben. Im mittleren sowie im westlichen Bereich wurden die Bedingungen zum Aufstellen eines Bluetooth-Loggers nicht erfüllt. Zwei weitere wurden aufgestellt, meldeten jedoch keine Messungen. Einer war zu stark verdeckt und konnte sich nicht mit dem WLAN verbinden, der andere wurde abgesteckt vorgefunden.

7. Datenspeicherung

Im Schnitt treffen drei Messungen pro Sekunde beim MQTT-Broker ein[1]. In der Verarbeitungsschicht werden diese gesammelt und verarbeitet. Dies geschieht im Dienst *Aggregierung*, der als MQTT-Client auftritt. Mithilfe des generischen Topics

`sensor/BLE/Scanner/#`

werden alle Sensor-Topics abonniert. Messungen können den Bluetooth-Loggern (im Folgenden auch *Station* genannt) zugeordnet werden, da der Identifikator Teil des konkreten Topics ist. So leitet sich aus dem Topic

`sensor/BLE/Scanner/5`

die Nummer *5* als Identifikator ab. Im ersten Schritt werden in Abschnitt 7.1 Messdaten klassifiziert und anschließend anonymisiert.

Anschließend war zunächst eine Filterung angedacht. Die Motivation ist eine Bereinigung der Daten von stationären und mobilen Geräten der Mitarbeitenden, die das Ergebnis verfälschen. Bezogen auf öffentliche MAC-Adressen könnten bspw. außerhalb der Öffnungszeiten gesammelte MAC-Adressen auf die schwarze Liste gesetzt werden, da diese vermutlich von stationären Geräten stammen. MAC-Adressen von Mitarbeitenden könnte man bspw. in der Öffentlichkeit verschlossenen Bereichen sammeln und ebenfalls auf die schwarze Liste setzen. Im Fall von randomisierten MAC-Adressen würde diese Liste nach 15 Minuten unnütz, da darüber hinaus Geräte nicht wiedererkannt werden können. Aus diesem Grund wurde von einer Filterung abgesehen.

Nach der Anonymisierung wird in Abschnitt 7.2 beschrieben, wie Messungen in sog. *Zeitfenstern* aggregiert werden, mit dem Ziel die Datenmenge zu reduzieren ohne Datenqualität zu verlieren. Das dort verwendete Datenmodel wird anschließend in ein relationales Modell überführt, sodass gesammelte Messungen in der Datenbank des Dienstes *DB* abgelegt werden können.

Felder der Statusnachrichten sowie der Will-Messages werden als Metadaten der Bluetooth-Logger in der Datenbank abgelegt. Es findet keine weitere Verarbeitung dieser Daten statt. In der Betriebsschicht wird daraus der aktuelle Verbindungsstatus der Bluetooth-Logger abgeleitet (siehe Abschnitt A.2).

[1] Es wurden 13.105.271 Messungen in 4.233.600 Sekunden (sieben Wochen) aufgezeichnet. Daraus ergeben sich $\frac{13.105.271}{4.233.600} \approx 3,1$ Messungen pro Sekunde.

7.1. Daten anonymisieren

Das Konzept und die Motivation der Datenanonymisierung ist in Abschnitt 3.5 beschrieben. Dabei werden Messungen zunächst klassifiziert und im Anschluss anonymisiert. Tabelle 7.1 zeigt vier echte Messungen verschiedener Art. Eine, die von einer öffentlichen MAC-Adresse stammt und drei, die von randomisierten MAC-Adresses gesendet wurden. Von diesen dreien stammt die Erste von der CWA, bei der Zweiten ist das Feld Manufacturer-Specific-Data gesetzt und die Dritte enthält einen Gerätenamen.

	Öffentliche MAC-Adresse	Randomisierte MAC-Adresse		
		CWA	Hersteller	Gerätename
MAC-Adresse	**88:36:cf**:xx:xx:xx	19:c6:48:1c:9c:ca	52:21:03:3e:b8:03	82:77:16:f7:c7:15
Manufacturer-Specific-Data	**7d02**010300ffcc[...]	keine	**4c00**10060d1a22[...]	keine
Service-Data	keine	0000**fd6f**-0000-[...]	keine	keine
Gerätename	HUAWEI **Band** [...]	kein	kein	Galaxy **Watch** [...]

Tabelle 7.1.: Vier Beispielmessungen *vor* der Anonymisierung. Jeweils hervorgehoben sind die, für die Klassifizierung nützlichen, Informationen.

Messungen der Tabelle 7.1 werden nun in klassifizierte und anonymisierte Messungen der Tabelle 7.2 überführt. Anhand der MAC-Adresse kann bestimmt werden, ob es sich um eine öffentliche oder randomisierte handelt (vgl. Abschnitt 3.2). Diese Eigenschaft wird in dem Wahrheitswert *isPublic* abgelegt.

Der Hersteller aus der ersten Messung ergibt sich aus der Herstellerkennung (OUI). Daneben ließe sich der Hersteller ebenso aus der Manufacturer-Specific-Data oder dem Gerätenamen ableiten. Der Gerätename verrät außerdem, dass es sich um ein Gerät der Klasse Fitnesstracker handelt[2] Die zweite Messung wird als CWA klassifiziert. In der Service-Data findet sich die UUID `0xfd6f`, die das ENF identifiziert, welches die CWA nutzt. Findet sich diese UUID wird der Wahrheitswert *isCwa* gesetzt. Bei der dritten Messung kann anhand der Manufacturer-Specific-Data auf den Hersteller geschlossen werden. Die letzte Messung lässt sich über den Gerätenamen klassifizieren.

	Öffentliche MAC-Adresse	Randomisierte MAC-Adresse		
		CWA	Hersteller	Gerätename
MAC-Hash	6fa46e2d353a2f[...]	66fab87ecc249b[...]	60f027ec747ba8[...]	5cebcabcbd2958[...]
Hersteller	**Huawei Device Co., Ltd.**		**Apple, Inc.**	
isPublic	**wahr**	falsch	falsch	falsch
isCwa	falsch	**wahr**	falsch	falsch
Gerätetyp	**Fitnesstracker**			**Smartwatch**

Tabelle 7.2.: Vier Beispielmessungen *nach* der Anonymisierung. Jeweils hervorgehoben wurden die, durch die Klassifizierung gewonnenen, Informationen.

[2]Die Liste der Klassifizierung anhand des Bluetooth-Gerätenamens findet sich im Anhang in Tabelle A.1.

Nachdem die Klassifizierung abgeschlossen ist, wird jeweils der MAC-Hash erzeugt und die ursprünglichen Daten aus Tabelle 7.1 gelöscht. Zusammen mit dem Zeitstempel (*timestamp*) der Messung gehen die Daten aus Tabelle 7.2 in die weitere Datenverarbeitung ein.

7.2. Aggregierung in Zeitfenster

Nach der Anonymisierung lassen sich je Bluetooth-Logger (hier *Station* genannt) arbiträr viele Messdaten des folgenden Format sammeln:

$$data_{Station} = \{\ macHash\ manufacturer\ isCwa\ type\ isPublic\ timestamp\ \};$$

Dabei werden Advertisements eines Gerätes mehrmals erfasst. Hält sich ein Gerät bspw. 15 Minuten im Empfangsbereich eines Bluetooth-Loggers auf, würde dessen Advertisement 90 mal aufgezeichnet[3]. Eine Liste aller Messungen enthielte redundante Informationen, da sich nur der Zeitpunkt dieser Advertisements ändert.

7.2.1. Reduktion durch Transformation

Um Messungen eines Gerätes zusammenzufassen werden *Zeitfenster* (*TimeWindow*) durch Startzeitpunkt (*timeFrom*) und Endzeitpunkt (*timeTo*) definiert. Das resultierende Objektmodell ist in Abbildung 7.1 dargestellt. Innerhalb eines Zeitfensters werden je Gerät (identifiziert durch *macHash*) alle Messungen (*BleMeas*) zu einer aggregiert. Dabei wird der Aufzeichnungszeitpunkt (Attribut *timestamp*) ersetzt durch die Zeitpunkte des ersten (*timeIn*) und letzten (*timeOut*) Advertisements innerhalb dieses Zeitfensters.

Für die anschließenden Auswertungen sind die Daten vor und nach der Transformation äquivalent. So findet dieses Objektmodell in der Implementierung Einsatz beim Eintreffen der Messungen. Diese Datenstruktur ist robust in Bezug auf sog. *Nachläufer*. Treffen Messungen zu spät[4] im System ein – bspw. aufgrund schlechter Internetanbindung oder kurzfristiger hoher Netzlast –, können sie nachträglich dem entsprechenden Zeitfenster zugeordnet werden. Das zugehörige *BleMeas*-Objekt kann anhand des MAC-Hashes identifiziert und gegebenenfalls in den Attributen *timeIn* bzw. *timeOut* angepasst werden.

Im weiteren Verlauf dieser Arbeit werden Zeitfenster der festen Größe von *zwei Minuten* eingesetzt (die Zeitdifferenz von Startzeitpunkt und Endzeitpunkt). Die Größe dieses Intervalls wurde nach den folgenden Kriterien gewählt:

Akkuratesse: Sowohl die Latenz als auch die Ungenauigkeit der Daten steigen je größer das Zeitfenster wird. Um die zeitliche Präzision sowie die Feingranularität der Echtzeitdaten zu erhalten, ist ein kleines Intervall gewünscht.

[3] Ein Bluetooth-Logger scannt für zehn Sekunden, dabei würde das Advertisement nur einmal gemeldet. Daraus ergeben sich sechs Advertisement pro Minute und 90 pro 15 Minuten.

[4] *Zu spät* meint hier nach Ablauf des zugehörigen Zeitfensters. Endete bspw. ein Zeitfenster um 12:00:00 Uhr, es treffen aber Messdaten von 11:59:55 Uhr erst um 12:00:05 Uhr ein, so ist dies außerhalb des aktuellen Zeitfensters.

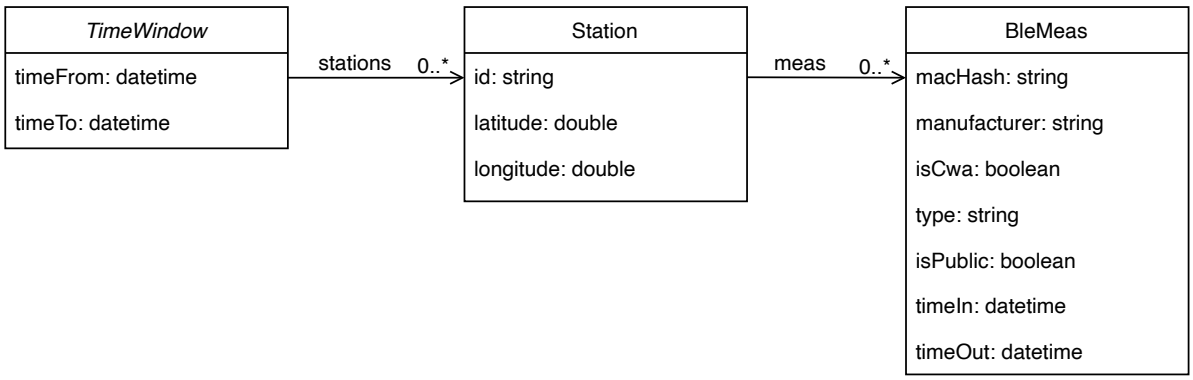


Abbildung 7.1.: Objektmodell der Messdaten (*BleMeas*) je Bluetooth-Logger (*Station*) je Zeitfenster (*TimeWindow*). Ein Zeitfenster hält eine Liste beliebiger Länge (*0..**) an Bluetooth-Loggern, die innerhalb dieses Zeitfensters Messdaten erfasst haben. Diese Messdaten werden als Liste des jeweiligen Bluetooth-Loggers gehalten. Erfolgte keine Messung, bleiben Listen leer.

Datenreduktion: Je größer das Zeitfenster gewählt wird, desto mehr Daten können eingespart werden. Um eine hohe Datenreduktion zu erzielen, wird ein großes Intervall angestrebt.

Wiederkehrende Advertisements eines Gerätes sollen innerhalb eines Zeitfensters abgelegt werden, sodass das jeweils erste und letzte einen ungleichen Zeitstempel aufweisen. Ist diese Eigenschaft erfüllt, unterstreicht dies die Anwesenheit dieses Gerätes und Irrläufer[5] können ausgeschlossen werden. Während das Sendeintervall der CWA bei Bruchteilen von Sekunden liegt, kann es nach der Bluetooth Core Specification bis zu wenigen Minute lang sein[6].

Zwei Minuten sind ausreichend lang, um wiederkehrende Advertisements in einem Zeitfenster zu erfassen und dennoch ausreichend kurz, um Echtzeitdaten geringer Latenz zu erhalten. Mit dieser Diskretisierung würde das Gerät des Eingangsbeispiels in 15 Minuten statt 90 Einträgen acht erzeugen ($\lceil \frac{15}{2} \rceil = 8$).

Zeitfenster lassen sich bei Abfragen nachträglich beliebig aggregieren. Analog des Transformationsschritts werden Messungen mehrerer Zeitfenster nach dem MAC-Hash gruppiert. Dabei wird der Eintrittszeitpunkt auf den jeweiligen Minimalwert, der Austrittszeitpunkt auf den jeweiligen Maximalwert gesetzt. Im aggregierten Zeitfenster bleiben kurzweilige Geräte-Einträge erhalten, Messungen über mehrere Zeitfenster werden zusammengefasst.

Während des Testzeitraums von sieben Wochen (siehe Kapitel 8) wurden insgesamt 13 Mio. Messungen aufgenommen. Diese wurden 34 Tsd. Zeitfenstern zugeteilt und dabei auf insgesamt

[5] Irrläufer meint MAC-Adressen, die nur ein einziges mal aufgezeichnet werden. Diese können auf Messfehler zurückzuführen sein, oder von passierenden Geräten außerhalb des Testgeländes herrühren.

[6] Siehe Bluetooth Core Specification 5.3, Part C, 9.3.11, S. 1326: „[. . . A]dvertising might be alternately enabled for only a few seconds and disabled for several minutes.“ (Verfügbar unter: `https://www.bluetooth.com/specifications/specs/core-specification-5-3/`)

2,3 Mio. Einträgen aggregiert. Die resultierende Datenreduktion beträgt 82 %[7].

7.2.2. Relationale Auflösung

Die eben vorgestellte Datenstruktur (vgl. Abbildung 7.1) eignet sich gut für die erste Erfassung von Daten, da sie sich stark an dem Eintreffen neuer Messungen orientiert.

Für Abfragen macht es die Indirektion von *TimeWindow* zu *BleMeas* über *Station* kompliziert. So lassen sich für ein gegebenes Zeitintervall alle Messdaten aller Bluetooth-Logger gut abfragen. Aufwendig sind Abfragen über alle Messungen *einer* Station unterschiedlicher Zeitfenster, oder über alle Stationen, die *ein bestimmtes* Gerät aufgezeichneten. Ein effizienteres Datenmodell ist für das Persistieren gefragt. Hierfür werden die 1:n-Beziehungen der Entitäten aufgelöst. *TimeWindow* verliert die Liste *stations*, *Station* die Liste *meas*. Dieser Schritt macht die beiden Entitäten zu reinen Datenträgern ohne Kopplung. Abbildung 7.2 zeigt das transformierte Objektmodell. Neue Aggregationsbeziehungen gehen von der zentralen Entität *BleMeas* aus. Sie erhält die jeweils einwertigen Attribute *timeWindow* sowie *station*.

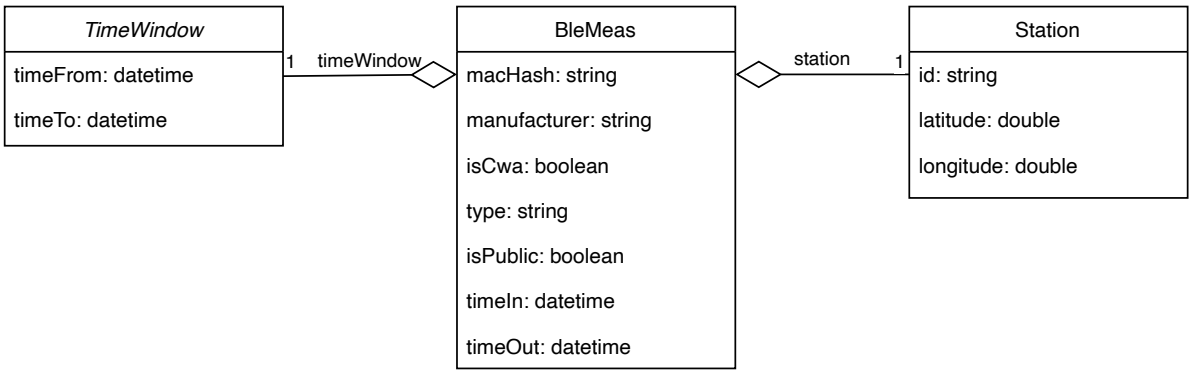


Abbildung 7.2.: Transformiertes Objektmodell der Messdaten nach Auflösung der 1:n-Beziehungen. Eine Messung (*BleMeas*) stammt von *einem* Bluetooth-Logger (*Station*) innerhalb *eines* Zeitfensters (*TimeWindow*). Zeitfenster und Bluetooth-Logger existieren unabhängig von einer Messung.

Objekte des abgeänderten Modells in Abbildung 7.2 passen problemlos in das folgende relationale Datenbankmodell. Letzteres ist in Abbildung 7.3 dargestellt. Im Zuge der Normalisierung wird für die Tabelle Zeitfenster (*time_windows*) der Primärschlüssel des Feldes Startzeit (*time_from*) und für die Tabelle Bluetooth-Logger (*stations*) das Feld *id* definiert. Für die Tabelle der Messungen *ble_meas* reicht ein Feld für den Primärschlüssel nicht aus. Ein MAC-Hash kann von mehreren Bluetooth-Logger in mehreren Zeitfenstern erfasst werden. Um der zweiten Normalform[8] zu genügen wird hier der Verbundschlüssel aus MAC-Hash, Zeitfenster und Bluetooth-Logger gewählt.

[7] Anteil aggregierter Einträge an Referenzmessungen: $\frac{2.317.840\ \text{[Einträge]}+34.554\ \text{[Zeitfenster]}}{13.104.033\ \text{[Referenzmessungen]}} \approx 17,95\ \%$

[8] Die zweite Normalform wird angestrebt, um mögliche Redundanzen in der Datenbank zu verhindern und so Inkonsistenzen vorzubeugen.

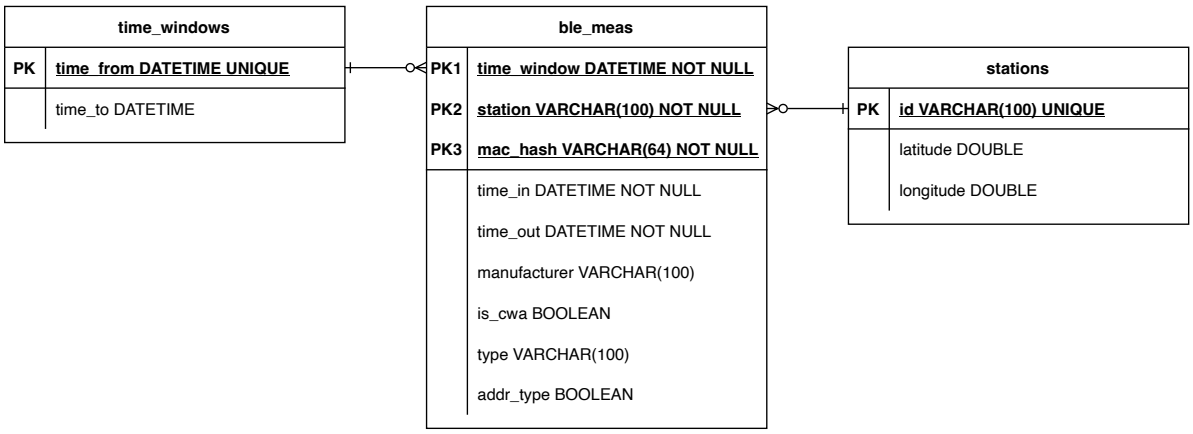


Abbildung 7.3.: Relationales Datenbankmodell. Eine Messung in Tabelle *ble_meas* wird über den Verbundschlüssel aus den Feldern Zeitfenster (*time_window*), Bluetooth-Logger (*station*) und MAC-Hash (*mac_hash*) identifiziert, wobei die Felder Zeitfenster und Bluetooth-Logger jeweils Primärschlüssel ihrer Tabellen sind.

In der Implementierung finden diese Modelle im Dienst *Aggregierung* sequentiell Einsatz. Eintreffende Messdaten werden im Objektmodell gepuffert gesammelt (Abbildung 7.1). Zum Persistieren werden sie in das transformierte Modell überführt (Abbildung 7.2) und in einer SQL-Datenbank des relationalen Modells abgelegt (Abbildung 7.3).

Am Ende der Datenspeicherung wurden die Messdaten anonymisiert, die Datenmenge wurde auf 28 % reduziert und die Transformation ermöglicht zeitlich feingranulare Abfragen für die Auswertung.

8. Datenauswertung

Nachdem die Messdaten aufbereitet in der Datenbank abgelegt wurden, beschäftigt sich dieses Kapitel damit, die Daten zielorientiert auszuwerten, um die Forschungsfragen zu beantworten. In der Fünfschichtenarchitektur ist dies in der Anwendungsschicht im Dienst *Abfrage* zu verorten.

Zu Beginn wird die Verfügbarkeit der Bluetooth-Logger untersucht, da über den gesamten Testzeitraum einige ausfielen (Abschnitt 8.1). Die Anteile öffentlicher und randomisierter MAC-Adressen an allen Messungen werden in Abschnitt 8.2 berechnet. Abschnitt 8.3 versucht Bluetooth-Geräte nach Hersteller bzw. nach Gerätetyp zu klassifizieren. Anschließend werden in Abschnitt 8.4 Stoßzeiten über die Wochentage analysiert. Im darauffolgenden Schritt werden die Standorte der Bluetooth-Logger einbezogen. Diesen wird in Abschnitt 8.5 die Anzahl an gemessenen Geräten je Tageszeit zugeordnet. So können hoch frequentierte Bereiche erkannt und erste Vermutungen zu Besucherströmen aufgestellt werden. Abschnitt 8.6 behandelt das Erkennen von Übergängen zwischen den Bluetooth-Loggern. Diese werden mit Aufenthaltswahrscheinlichkeiten in einer Übergangsmatrix kombiniert. Dass das System Messdaten korrekt widerspiegelt wird mithilfe des Feldvergleiches in Abschnitt 8.7 überprüft. Abschließend wird der Wert und Einfluss der CWA-Messungen untersucht (Abschnitt 8.8).

Die Aufzeichnung der Messdaten erfolgte über sieben Wochen in der Sommersaison 2022 (vom 12. August bis zum 29. September). Alle Bluetooth-Logger meldeten in diesem Zeitraum insgesamt 13 Mio. Messungen, dabei ist eine Messung im Schnitt ca. 200 Byte groß. Daraus ergeben sich ca. 51 MB Daten pro Tag[1]. Während der Aggregierung wurden die 13 Mio. Referenzmessungen auf 2,3 Mio. Messungen reduziert (vgl. Unterabschnitt 7.2.1).

8.1. Verfügbarkeit der Bluetooth-Logger

Die Gruppierung aller Messungen nach eindeutiger MAC-Adresse und Bluetooth-Logger pro Tag ist in Abbildung 8.1 dargestellt. Es zeigen sich höhere Zahlen am Wochenende sowie abnehmende im Spätsommer. Die meisten Messungen gab es am Sonntag, dem 21. August.

Von den insgesamt neun aufgestellten Bluetooth-Logger, meldeten nur sechs an mehreren Tagen Daten. Bluetooth-Logger *8* war am 28.08., *9* am 02.09. und *1* am 22.09. zuletzt online. Von Bluetooth-Logger *7* trafen ausschließlich am 23.09. Messungen ein. Bluetooth-Logger *9* wurde mit einem defekten Netzteil vorgefunden. Auf zu geringen WLAN-Empfang zurückführen lässt sich, dass Bluetooth-Logger *1*, *7* und *8* ausgefallen sind. Diese Auswertung ist für die *Überwachung* seitens der Betriebsschicht interessant.

[1] Datenmenge pro Tag: 13.105.271 Messungen $\cdot \frac{200}{2^{20}} \frac{\text{MB}}{\text{Messung}} \cdot \frac{1}{49 \text{ Tage}} \approx 51,0 \frac{\text{MB}}{\text{Tag}}$.

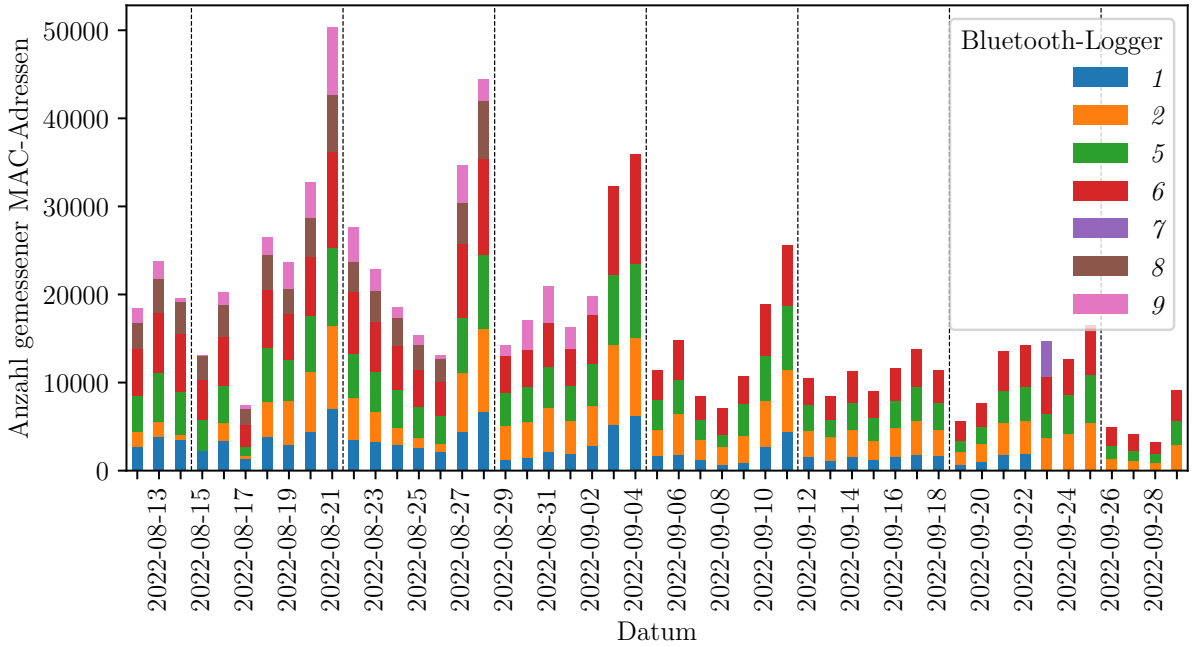


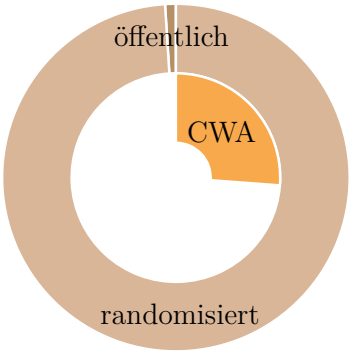
Abbildung 8.1.: Anzahl unterschiedlicher MAC-Adressen pro Tag pro Bluetooth-Logger (*station*) über den Testzeitraum. Senkrechte Linien unterteilen in Wochen.

Die Erkenntnisse über die Verfügbarkeit der Bluetooth-Logger beschränken den folgenden Auswertungszeitraum auf den *Zeitraum vom 14. bis 28. August*, da nur in diesem alle verfügbaren Bluetooth-Logger zuverlässig Messungen meldeten. Auswertungen, die unabhängig des jeweiligen Bluetooth-Logger-Standorts sind, werden auf dem gesamten Testzeitraum erhoben.

8.2. Anteile öffentlicher und randomisierter MAC-Adressen

Im Testzeitraum wurden insgesamt 2,3 Mio. Messungen verzeichnet. Deren Verteilung ist in Abbildung 8.2 dargestellt. Es zeigt sich der geringe Anteil an öffentlichen MAC-Adressen von einem Prozent. Messungen der CWA sind eine Teilmenge der randomisierten und betragen ein Viertel. Dieser Anteil ist ausreichend groß, um mit diesen Daten zu arbeiten. Gleichzeitig ist er klein genug, dass Ergebnisse ohne die CWA aussagekräftig bleiben.

Hinweis zum Anteil der CWA: Zum Ende dieser Arbeit fiel eine fehlerhafte Annahme in der Bluetooth-Bibliothek der Messgeräte auf. Diese aggregiert Geräte anhand der MAC-Adresse. Wird ein zweites Advertisement derselben MAC-Adresse empfangen, wird dieses ignoriert, da angenommen wird, dass dieses Gerät bereits bekannt ist. Über eine MAC-Adresse können aber mehrere Dienste (spezifiziert durch *Service-Data*) gesendet werden. Jedoch wird



Typ	Anzahl Messungen	Anteil
öffentlich	29.469	1,3 %
randomisiert	2.288.371	98,7 %
$\sum$	2.317.840	
CWA	594.524	25,6 %

Abbildung 8.2.: Anteile öffentlicher und randomisierter MAC-Adressen. Von 2,3 Mio. Messungen stammen 1,3 % von öffentlichen MAC-Adressen, 98,7 % von randomisierten. 25,6 % der Messungen entspringen der CWA.

nur der erste Dienst aufgezeichnet. Da die CWA mehrmals pro Sekunde Advertisements sendet, werden die anderer Dienste desselben Gerätes i.d.R. überdeckt. Dies hat zur Folge, dass der Anteil in Abbildung 8.2 zu hoch angesetzt sein kann. Die absolute Zahl von 594 Tsd. bildet die untere Grenze der tatsächlichen CWA-Messungen, da mindestens diese Anzahl an Messungen der CWA zugeordnet wurden. Es lassen sich dennoch Erkenntnisse exklusiv aus CWA-Messungen erzeugen. Zur Korrektur der Verhältnisse müsste die Aggregierung der Bluetooth-Logger auf Basis der MAC-Adresse *und* der Service-Data erfolgen. Das Open-Source-Projekt Corona-Warn-App meldete im Februar 2022 insgesamt 25,1 Mio. Nutzende der CWA [Ope22]. Dies entspricht 29,9 % der deutschen Bevölkerung[2]. Unter der Annahme, dass dieser Anteil für Besuchende des Testgeländes repräsentativ ist, bekräftigt die geringe Differenz von 29,9 % und 25,6 % die Akkuratesse der Messungen sowie den geringen Fehler der zuvor beschriebenen fehlerhaften Annahme.

Die Verteilung von randomisierten gegenüber den öffentlichen MAC-Adressen bleibt hiervon unberührt. Da es im weiteren Verlauf um die *Anzahl an Geräten* geht und diese über die MAC-Adresse identifiziert werden, ist von dieser fehlerhaften Annahme lediglich die Klassifizierung betroffen.

8.3. Klassifizierung von Bluetooth-Geräten

Die während der Anonymisierung in Abschnitt 7.1 vorgenommene Klassifizierung wird nun ausgewertet. Mithilfe des OUI-Lookups sowie der Manufacturer-Specific-Data wurde versucht, den Hersteller der Bluetooth-Geräte zu bestimmen (vgl. Abschnitte 3.2 und 3.4). Dabei ließ sich das OUI-Lookup auf Messungen von öffentlichen MAC-Adressen anwenden, die Klassifizierung anhand der Manufacturer-Specific-Data wurde bei Messungen von randomisierten MAC-Adressen genutzt. Die Ergebnisse der Herstellerzuordnung sind in Tabelle 8.1 dargestellt.

[2]Das statistische Bundesamt meldet die Bevölkerung Deutschlands zum 30.06.2022 mit 84.079.800 Personen. Vgl. Pressemitteilung unter:
`https://www.destatis.de/DE/Presse/Pressemitteilungen/2022/09/PD22_410_12411.html`

Hersteller	Anzahl Messungen	Anteil	Messungen von öffentlichen MAC-Adressen
Apple	1.341.948	57,9 %	19
Unbekannt CWA	594.524	25,6 %	0
Unbekannt	304.500	13,1 %	10.501
Microsoft	37.040	1,6 %	0
Samsung Electronics Co.	10.397	0,4 %	0
Garmin International	3.781	0,2 %	0
Huawei Device Co.	3.717	0,2 %	3.717
Bose Corporation	2.600	0,1 %	2.600
183 andere	19.333	0,8 %	12.632
$\sum$	2.317.840		29.469

Tabelle 8.1.: Top sechs Gerätehersteller im Testgelände bestimmt durch OUI-Lookup und Manufacturer-Specific-Data. Apple, Microsoft, Samsung und Huawei sind bekannte Hersteller von Smartphones, Tablets und Smartwatches und setzen randomisierte MAC-Adressen ein. Garmin legt seinen Schwerpunkt auf Smartwatches, Bose auf Audio-Produkte, beide senden ausschließlich von öffentlichen MAC-Adressen.

Der Hersteller Apple hat die meisten zugewiesenen Messungen von 57 %. Den zweithöchsten Anteil übernehmen Messungen der CWA, die nach Spezifikation keine Herstellerinformationen veröffentlichen [vgl. AG20, S. 5]. Von den übrigen 16,5 % der Messungen konnten 13,1 % keinem Hersteller zugeordnet werden. Die restlichen 3,4 % verteilen sich auf alle anderen Hersteller. Diese Verteilung scheint nicht repräsentativ zu sein, wenn man sie mit den Marktanteilen der Hersteller am Absatz von Smartphones[3] vergleicht. Hier hat Samsung den größten Anteil mit 21,2 %, gefolgt von Apple mit 17,2 % und Xiaomi mit 13,4 %. Während Xiaomi in der o.g. Auswertung gar nicht enthalten ist, ist Samsung mit weniger als einem Prozent vertreten. Es gibt zwei Erklärungsversuche für den hohen Wert bei Apple in Tabelle 8.1:

- Neben Smartphones senden viele weitere Geräte des Herstellers Apple Bluetooth-Advertisements, z.B. Tablets, Laptops, Kopfhörer, Schlüsselfinder, etc. Die Häufigkeit des Sendens hat dabei keinen Einfluss, da das Zeitfenster von zwei Minuten die regulären Sendeintervalle abdeckt (vgl. Unterabschnitt 7.2.1).
- Das Protokoll, das in der Manufacturer-Specific-Data genutzt wird, hat der Hersteller Apple bei der Bluetooth SIG registriert. So kann diesem Protokoll der Hersteller zugeordnet werden. Es ist aber möglich, dass Geräte anderer Hersteller dieses Protokoll implementieren, um eine Interoperabilität zu ermöglichen. Diese Geräte wurden in dieser Auswertung dann fälschlicherweise dem Hersteller Apple zugeschrieben.

[3] Vgl. Statista: „Marktanteile der Hersteller am Absatz von Smartphones weltweit im 3. Quartal 2022 unter: https://de.statista.com/statistik/daten/studie/451386/umfrage/weltweite-marktanteile-der-smartphone-hersteller-nach-quartal/

Das OUI-Lookup der öffentlichen MAC-Adresse zeigt eine zuverlässige Zuordnung des Herstellers in Tabelle 8.1 in der letzten Spalte. So senden alle gemessenen Geräte von Huawei und Bose mit öffentlichen MAC-Adressen. Dies legt aber die Vermutung nahe, dass Geräte mit randomisierter MAC-Adresse in dieser Klassifizierung gar nicht auftauchen, das sie keinen Schluss auf den Hersteller zulassen. So war diese Klassifizierung nach Hersteller vor der Randomisierung von MAC-Adresse ein nutzbares Werkzeug. Auf Grundlage der Manufacturer-Specific-Data ist keine Zuordnung des Herstellers möglich.

Der Gerätename ist in ca. 2,5 % der Messungen gesetzt[4]. Die Verteilung auf die Gerätetypen der 2.462 unterschiedlichen Gerätenamen ist in Tabelle A.1 im Anhang zu finden. Den größten Anteil bilden Smartwatches mit 27,7 %. Dabei ist die Grundmenge die Anzahl aller Geräte, bei denen ein Gerätename gesetzt war. Es lässt sich die Aussage treffen, dass etwa ein Viertel der Gerätenamen zu Schlüsselwörtern der Smartwatch passen. Auf Grundlage dieser Zuordnung kann aber nicht auf alle Geräte geschlossen werden, da erstens der Anteil von 2,5 % zu gering ist und zweitens die Gerätenamen nicht auf alle Gerätetypen gleichmäßig verteilt sind. So konnte der Gerätetyp Smartphone nur anhand des Wortes *iphone* erkannt werden, das den Hersteller auf Apple beschränkt. Rückschlüsse auf Smartphones anderer Hersteller ließen die 2.462 Gerätenamen nicht zu.

Die ursprüngliche Idee war es, aus der Klassifizierung Besucherprofile abzuleiten (bspw. Schulklassen oder Familien). Diese Auswertung zeigt, dass eine repräsentative Klassifizierung mit den hier vorgestellten Ansätzen nicht möglich ist.

8.4. Stoßzeiten

Die Frage der Betreibenden „Um welche Tageszeit und an welchem Wochentag sind viele Besuchende vor Ort?“ wird mit Abbildung 8.3 beantwortet. Über alle Messungen des siebenwöchigen Testzeitraums, unabhängig vom aufnehmenden Bluetooth-Logger, werden eindeutige MAC-Adressen nach Wochentag und Tageszeit gruppiert und aufsummiert. Diese sind als Bluetooth-Geräte zu verstehen. Dabei wird sich an den Öffnungszeiten des Testgeländes von 9:00 bis 18:30 Uhr orientiert. Die SQL-Abfrage ist in Unterabschnitt A.4.1 zu finden.

Die Verteilung unter der Woche von Montag bis Donnerstag zeigt sich vergleichbar. Zur Mittagszeit wird jeweils der Maximalwert erreicht, davor und danach gleichen die Aufenthaltszahlen einer Normalverteilung.

Zum Freitag verschiebt sich das Tagesmaximum in den Nachmittag, während die Tagessumme im Vergleich zu den Vortagen ansteigt. Die folgende Tabelle 8.2 zeigt die Tagessummen der Wochentage.

Am Wochenende sind deutlich mehr Besuchende zu verzeichnen. Die Hauptbesuchszeit ist am im Nachmittag. Während unter der Woche der Durchschnitt bei ca. 87 Tsd. MAC-Adressen liegt, verdoppelt sich dieser am Wochenende. Diese Zahlen und Verhältnisse decken sich mit

[4]Von 2.317.840 Messungen enthielten 57.648 einen Gerätenamen, darunter finden sich 2.462 unterschiedliche. Dabei ist es möglich, dass unterschiedliche Geräte denselben Gerätenamen verwenden.

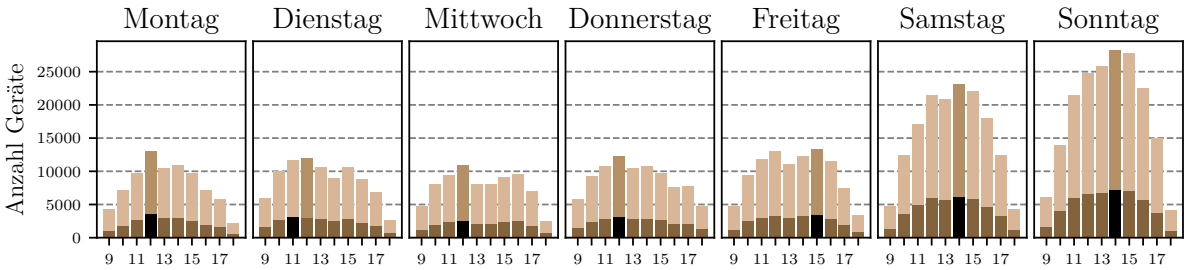


Abbildung 8.3.: Stoßzeiten: Eindeutige MAC-Adressen je Wochentag und Stunde aufsummiert, das jeweilige Tagesmaximum ist hervorgehoben. Der Anteil der CWA ist dunkel eingefärbt. Dieser hat denselben Wochenverlauf.

Wochentag	Anzahl eindeutiger MAC-Adressen	
Montag	80.388	
Dienstag	88.115	
Mittwoch	77.596	$\varnothing \approx 86.673$
Donnerstag	89.174	
Freitag	98.093	
Samstag	156.407	$\varnothing = 173.011$
Sonntag	189.615	
$\varnothing$	111.341	

Tabelle 8.2.: Anzahl eindeutiger MAC-Adressen pro Wochentag.

Stoßzeiten von Google[5]. Die an einigen Wochentage auftretende Abweichung des Tagesmaximum um eine Stunde, lässt sich auf unterschiedliche Intervallgrenzen zurückzuführen. Während in dieser Arbeit die Summe für Stunde 13 von 13:00 Uhr bis 13:59 Uhr gebildet wird, wählt Google den Zeitraum von 12:30 Uhr bis 13:29 Uhr.

8.5. Anzahl Geräte je Bluetooth-Logger

Während die Auswertung der Stoßzeiten nur über die Tabelle Messungen (*ble_meas* in Abbildung 7.3) gemacht wurde, werden nun Bluetooth-Logger und Zeitfenster einbezogen. Gesucht ist die Anzahl an gemessenen Geräten (*devicesCount*) aller Bluetooth-Logger (*station*) in Abhängigkeit eines Zeitintervalls ([*timeFrom*,*timeTo*]):

$$devicesCount_{timeFrom,timeTo} = \{\ station\ devicesCount\ \};$$

[5]Siehe Artikel „Stoßzeiten, Wartezeiten und Besuchsdauer“ unter:
`https://support.google.com/business/answer/6263531?hl=de`

Die folgende SQL-Abfrage in Listing 8.1 liefert das gewünschte Ergebnis. Hier findet neben der Tabelle der Messungen (*ble_meas*) die Tabelle der Zeitfenster (*time_windows*) Einsatz. Der Aufbau der Abfrage über die Zeitfenster wird vielfach eingesetzt und deshalb kurz erklärt.

Listing 8.1: SQL-Abfrage für die Anzahl unterschiedlicher MAC-Adresse je Bluetooth-Logger innerhalb eines beliebigen Zeitintervalls. Das Zeitintervall wird mit den SQL-Variablen bzw. Abfrageparametern `@time_from` und `@time_to` angegeben.

```
1 SELECT station, COUNT(DISTINCT mac_hash) AS devices_count
2 FROM ble_meas
3 WHERE time_window IN (SELECT time_from
4                       FROM time_windows
5                       WHERE time_from >= @time_from
6                         AND time_to <= @time_to)
7 GROUP BY station;
```

Beginnend mit der Unterabfrage von Zeile 3 bis 6 werden alle Zeitfenster (`time_windows`) gewählt, die in das Intervall von `@time_from` bis `@time_to` fallen. Nun werden alle Messeinträge (`ble_meas`) danach gefiltert, ob das Zeitfenster des Eintreffens der Messung (`time_window`) in dieser Unterabfrage enthalten ist. Die gefilterte Teilmenge der Messungen wird nach Bluetooth-Logger (`station`) gruppiert und die Anzahl an unterschiedlichen MAC-Hashes aggregiert. Neben dieser Anzahl (`devices_count`) wird der Identifikator des Bluetooth-Loggers (`station`) angegeben. Weitere Felder der Tabelle `station` können über einen Verbund (*join*) beigefügt werden[6]. Da das Zeitfenster variabel gehalten ist, können sowohl Echtzeitdaten[7] als auch aggregierte Daten der Vergangenheit abgefragt werden.

Für die folgenden Auswertungen wurde *Sonntag, der 21.08.2022*, gewählt. Wie aus Abbildung 8.1 hervorgeht, wurden an diesem Tag am meisten Geräte von insgesamt sechs Bluetooth-Loggern gemessen. Dieses Datum lag in den Sommerferien vieler Bundesländer und das Wetter war sonnig und trocken[8].

Aus den Daten der o.g. Abfrage in Listing 8.1 wird Abbildung 8.4 erzeugt, indem der Startzeitpunkt (`@time_from`) auf den 21.08.2022, 0:00 Uhr und der Endzeitpunkt (`@time_to`) 24 Stunden später gesetzt wird. Hier dargestellt ist die Anzahl unterschiedlicher Geräte je Bluetooth-Logger an der jeweiligen Position im Testgelände. Die erste Zahl ist jeweils der Identifikator des Bluetooth-Loggers, die zweite die Geräteanzahl. Der Radius der Kreise steigt logarithmisch zu den verzeichneten Gerätezahlen. So werden für die Visualisierung geringe Zahlen betont und hohe gedämpft (die Kreisgröße bietet keinen Rückschluss auf die Reichweite der Bluetooth-Logger). Der Rundgang über das Testgelände führt vereinfacht vom Ein- und Ausgang (oben links, Bluetooth-Logger *6* und *8*) gegen den Uhrzeigersinn über die *2* und *9*, danach zur *5*, zur *1* und zurück zum Ausgang.

[6] Eine vollständige Abfrage mit Verbund findet sich im Anhang in Unterabschnitt A.4.2.

[7] Echtzeitdaten meint hier die Wahl des kürzesten Zeitintervalls, dessen Daten den Zustand der letzten zwei Minuten wiedergeben (vgl. Unterabschnitt 7.2.1).

[8] Siehe bspw. Wetterkontor vom 21.08.2022 mit 25,9°C bei 9,5 Sonnenstunden und keinem Niederschlag unter: `https://www.wetterkontor.de/de/wetter/deutschland/rueckblick.asp?id=81&datum0=21.08.2022&datum1=21.08.2022&jr=2022&mo=10&datum=21.08.2022&t=4&part=0`

Abbildung 8.4.: Visualisierung der Anzahl Geräte je Bluetooth-Logger aggregiert über den ganzen Tag des 21.08.2022. Die erste Zahl ist jeweils der Identifikator des Bluetooth-Loggers, die zweite die Anzahl gemessener Geräte. Die Kreisradien skalieren mit der Anzahl gemessener Geräte logarithmisch.

Im nächsten Schritt wird die Verteilung der Gerätezahlen zu bestimmten Tageszeiten betrachtet. Es werden vier Zeitintervalle von jeweils 30 Minuten Länge zu signifikanten Zeiten ausgewählt; beginnend um 9:00, 12:00, 15:00 und 18:00 Uhr. Das Testgelände öffnet um 9:00 und schließt um 18:30 Uhr. Hieraus ergeben sich das erste und letzte Zeitintervall. Die Mittagszeit ist für die Restaurants an den Standorten *9* und *2* sowie *5* interessant. Um 15 Uhr startet täglich eine Veranstaltung an Standort *5*. Die Gerätezahlen zu diese vier Zeiten werden in Abbildung 8.5 verglichen.

Nach der Öffnung um 9:00 Uhr sind am Eingang (*6* und *8*) die größten Zahlen erkennbar. Analog dem Rundgang erreicht der Großteil der Besuchenden die Bluetooth-Logger *2* und *9*, ein kleinere Teil sogar *5*. Entgegen des Rundgangs sind nur wenige Geräte bei Bluetooth-Logger *1* zu messen.

Zur Mittagszeit sind an Standorten der Restaurants *9* und *2* sowie *5* Häufungen zu sehen. Die letzte Station des Rundgangs (*1*) ist von mehreren Besuchenden erreicht. Neue Besuchende kommen konstant nach und verteilen sich auf dem Gelände.

Um 15:00 Uhr sticht Bluetooth-Logger *5* deutlich heraus, da hier eine Veranstaltung angekündigt ist. Über den Rest des Geländes sind Besuchende gleichmäßig verteilt. Im Vergleich zur

Mittagszeit nehmen die Zahlen am Eingang ab, was für weniger nachkommende Besuchenden spricht.

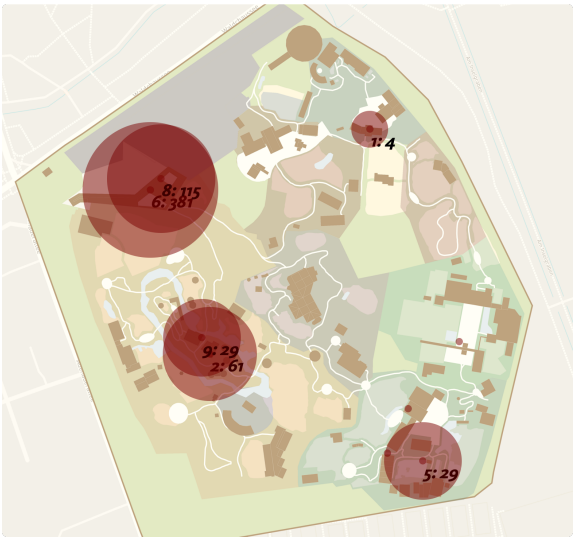
Vor der Schließung um 18:30 Uhr meldet Bluetooth-Logger *1* als letzte Station des Rundgangs die höchsten Zahlen. An den Standorten *9* und *2* sowie *5* sind rückläufige Zahlen zu verorten. Am Ausgang kommen Besuchende an den Bluetooth-Logger *6* und *8* erneut vorbei.

Mit dieser Auswertung lässt sich das physische Besucheraufkommen virtuell repräsentieren. Hoch frequentierte Orte und Spitzenzeiten können erkannt werden. Dies beantwortet die zweite Frage der Betreibenden „Welche Orte sind stark bzw. schwach besucht?“ in Echtzeit. Die dritte Frage zielt auf Besucherströme ab. Diese lassen sich entlang des Rundgangs bereits vermuten. Wie in Abbildung 8.5 in Sequenz gesetzt, ergibt sich über den Tag hinweg der Kreislauf entgegen des Uhrzeigersinns.

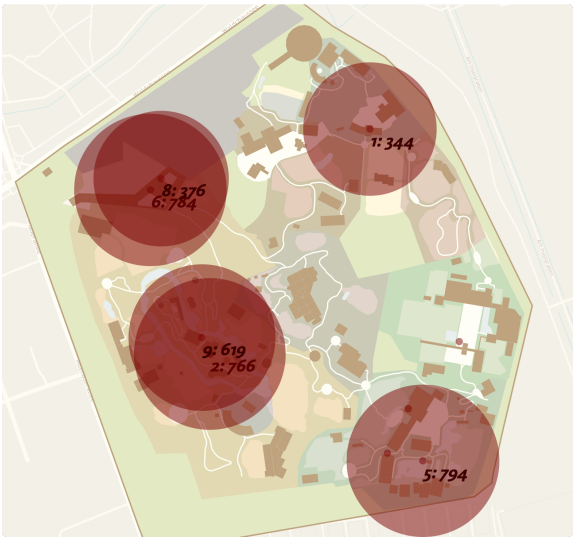
Setzt man die Anzahl besuchenden Personen ins Verhältnis zu der Gesamtanzahl unterschiedlicher MAC-Adressen eines Tages, ergibt sich der Extrapolationsfaktor b mit

$$b = \frac{\text{Anzahl besuchende Personen pro Tag}}{\text{Anzahl gemessene Geräte pro Tag}}.$$

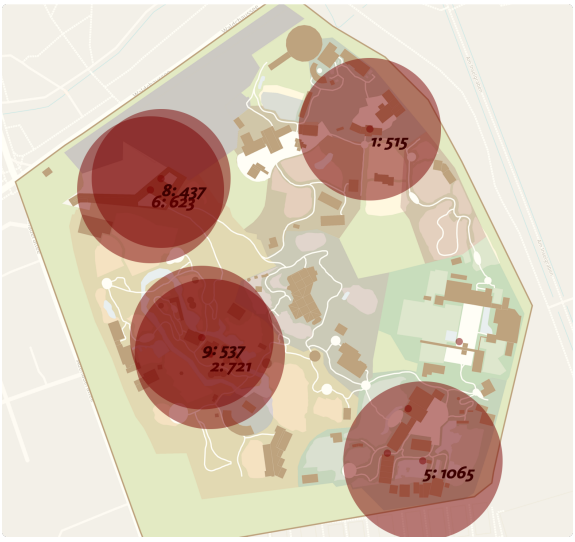
Mit diesem Faktor ließe sich die Verteilung der Besuchenden auf dem Testgelände schätzen. In Bezug auf Abbildung 8.5 wäre die Schätzung zwischen 15:00 und 15:30 Uhr bei Bluetooth-Logger *1* eine Besucherzahl von $515 \cdot b$.



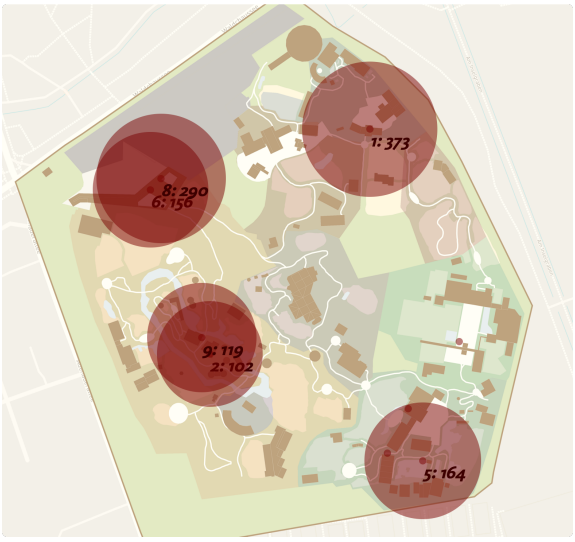
9:00 bis 9:30 Uhr nach Öffnung: Häufung am Eingang (*6* und *8*), erste Station des Rundgangs (*9* und *2*) werden erreicht.



12:00 bis 12:30 Uhr: Restaurants bei Bluetooth-Logger *9*, *2* und *5*. Letzte Station des Rundgangs (*1*) besucht.



15:00 bis 15:30 Uhr: Veranstaltung bei Bluetooth-Logger *5*, sonst ausgeglichene Verteilung.



18:00 bis 18:30 Uhr vor Schließung: Letzte Station des Rundgangs (*1*) sowie Ausgang (*6* und *8*) gehäuft.

Abbildung 8.5.: Visualisierung der Anzahl Geräte je Bluetooth-Logger über vier Tageszeiten des 21.08.2022 zu je einer halben Stunde aggregiert. Der Rundgang startet beim Ein-/Ausgang (*6* und *8*) und führt entgegen des Uhrzeigersinns an den Bluetooth-Logger *9* und *2*, *5* und *1* vorbei.

8.6. Übergangserkennung

Um einen Übergang messen zu können, muss ein Gerät von mehr als einem Bluetooth-Logger aufeinanderfolgend erfasst werden. Meldet bspw. Bluetooth-Logger *6* ein Gerät und fünf Minuten später Bluetooth-Logger *2* dasselbe Gerät, wird dies als Übergang von *6* zu *2* interpretiert. Dabei stellt die begrenzte Gültigkeitsdauer der MAC-Adresse des Gerätes eine Einschränkung dar, da Geräte mithilfe der MAC-Adresse identifiziert werden. Wechselt ein Gerät die MAC-Adresse während es von keinem Bluetooth-Logger erfasst wird, kann kein Übergang erkannt werden.

Im vorherigen Abschnitt wurde die Anzahl gemessener Geräte je Bluetooth-Logger untersucht. Die in Abbildung 8.4 dargestellten Messungen vom 21.08.2022 ergeben in Summe 50.313 Geräte. Ohne die Gruppierung nach Bluetooth-Logger wurden 39.484 unterschiedliche Geräte gemessen. Die Differenz bilden 10.829 Geräte, die von mehr als einem Bluetooth-Logger erfasst wurden. Diese Messungen sind die Grundlage für die Übergangserkennung, die sich somit auf *Sonntag, den 21.08.2022* beschränkt.

Wir stellen uns einen gerichteten Graphen vor, dessen Knoten die Bluetooth-Logger und dessen Kanten die Übergänge von Bluetooth-Geräten zwischen diesen darstellen. Nun wird über die 10.829 Messungen mehrfach erfasster Geräte iteriert. Wurde ein Gerät bspw. von Bluetooth-Logger *6* und später von *2* erfasst, aber von keinem anderen dazwischen, wird das Kantengewicht von *6* nach *2* um eins inkrementiert. Es ergibt sich die folgende Matrix T. Dabei meint der Wert t_{42}, dass 331 Geräte *von* Bluetooth-Logger *6 nach 2* gezählt wurden.

$$T = \begin{array}{c} \\ 1 \\ 2 \\ 5 \\ 6 \\ 8 \\ 9 \end{array} \begin{array}{c} \begin{array}{cccccc} 1 & 2 & 5 & 6 & 8 & 9 \end{array} \\ \begin{pmatrix} & 26 & 37 & 157 & 278 & 34 \\ 28 & & 160 & 61 & 105 & \mathbf{1481} \\ 259 & 36 & & 19 & 37 & 10 \\ 49 & 331 & 21 & & \mathbf{3247} & 355 \\ 26 & 167 & 10 & \mathbf{708} & & 203 \\ 21 & \mathbf{2709} & 34 & 76 & 144 & \end{pmatrix} \end{array} \tag{8.1}$$

Die Hauptdiagonale bleibt zunächst leer, da mit bisherigen Daten keine Aussagen über den Aufenthalt von Geräten gemacht werden kann. Bleibt eine wiederkehrende Messung eines Gerätes aus, kann das sowohl bedeuten, dass das Gerät den Bluetooth-Logger verlassen hat als auch, dass sich die MAC-Adresse geändert hat.

Die in Gleichung (8.1) markierten herausstechend großen Zahlen sind in den Übergängen beider Richtungen zwischen *2* und *9* sowie *6* und *8*. Je näher sich zwei Bluetooth-Logger sind, desto wahrscheinlicher ist es, einen Übergang zwischen ihnen zu erkennen. Es ist unwahrscheinlich, dass sich die MAC-Adresse nicht ändert zwischen zwei weit entfernten Bluetooth-Loggern. Um die Kantengewichte vergleichbar zu halten wird ein gleichmäßiger Abstand zwischen den Bluetooth-Logger angestrebt.

8.6.1. Aggregierung nahegelegener Bluetooth-Logger

Kandidaten für die Aggregierung sind die Bluetooth-Logger-Paare *6* und *8* sowie *2* und *9*. In Gleichung (8.2) sind links die Abstände in Metern zwischen allen Bluetooth-Loggern zu finden. Diese wurden als Luftlinie zwischen den jeweiligen Koordinatenpaaren berechnet. So beträgt der Abstand 201 m zwischen *2* und *8*, und zwischen *2* und *9* nur 20 m. Die Abstände nach der Aggregierung sind nachfolgend in D_{agg} dargestellt.

$$D = \begin{array}{c} \\ 1 \\ 2 \\ 5 \\ 6 \\ 8 \\ 9 \end{array} \begin{array}{c} \begin{array}{cccccc} 1 & 2 & 5 & 6 & 8 & 9 \end{array} \\ \begin{pmatrix} 0 & 307 & 370 & 258 & 243 & 298 \\ & 0 & 269 & 193 & 201 & 20 \\ & & 0 & 430 & 430 & 285 \\ & & & 0 & 17 & 173 \\ & & & & 0 & 181 \\ & & & & & 0 \end{pmatrix} \end{array} \rightarrow \quad D_{agg} = \begin{array}{c} \\ 1 \\ 2,9 \\ 5 \\ 6,8 \end{array} \begin{array}{c} \begin{array}{cccc} 1 & 2,9 & 5 & 6,8 \end{array} \\ \begin{pmatrix} 0 & \mathbf{302} & 370 & \mathbf{250} \\ & 0 & \mathbf{277} & \mathbf{187} \\ & & 0 & 430 \\ & & & 0 \end{pmatrix} \end{array} \tag{8.2}$$

Die Position von *2,9* wird als Mittelwert der Koordinaten der beiden Bluetooth-Logger *2* und *9* berechnet (*6,8* analog). Neu berechnete Abstände sind rechts hervorgehoben. Während die Standardabweichung aller Werte ungleich 0 aus D 118 m beträgt, fällt diese in D_{agg} auf 79 m[9]. Die Abstände zwischen den Bluetooth-Loggern sind somit gleichmäßiger und sollten für repräsentativere Daten sorgen.

Nun wird die Matrix T analog aggregiert. Die markierten Werte aus Gleichung (8.1) fallen weg, da sie nun keinen Übergang mehr bilden. Von den eingangs genutzten 10.829 Messungen, bleiben 2.684 aus denen die folgende Matrix T_{agg} erzeugt wird.

$$T_{agg} = \begin{array}{c} \\ 1 \\ 2,9 \\ 5 \\ 6,8 \end{array} \begin{array}{c} \begin{array}{cccc} 1 & 2,9 & 5 & 6,8 \end{array} \\ \begin{pmatrix} & 26+34 & 37 & 157+278 \\ 28+21 & & 160+34 & 61+105+76+144 \\ 259 & 36+10 & & 19+37 \\ 49+26 & 331+355+167+203 & 21+10 & \end{pmatrix} \end{array}$$

$$= \begin{array}{c} \\ 1 \\ 2,9 \\ 5 \\ 6,8 \end{array} \begin{array}{c} \begin{array}{cccc} 1 & 2,9 & 5 & 6,8 \end{array} \\ \begin{pmatrix} & 60 & 37 & 435 \\ 49 & & 194 & 386 \\ 259 & 46 & & 56 \\ 75 & 1.056 & 31 & \end{pmatrix} \end{array}$$

Dabei werden alle Übergänge von und zu einem aggregierten Bluetooth-Logger addiert. So bildet sich bspw. $t_{24} = 386$ als Summe aller Übergänge von *2* zu *6* und *8* sowie *9* zu *6* und *8*. Die Matrix T_{agg} ist die Grundlage für die Visualisierung in Abbildung 8.6.

[9] Für D: $\bar{x} = 245$ und $s = \sqrt{\frac{\sum (x-\bar{x})^2}{n}} \approx 118,2$ und
für D_{agg}: $\bar{x} \approx 302,6$ und $s \approx 79,1$

Abbildung 8.6.: Übergangsgraph aller Messungen des 21.08.2022. Knoten bilden Bluetooth-Logger *1* und *5* sowie die Aggregate *6,8* und *2,9*. Die Pfeilstärke und Spitzengröße skaliert linear mit dem Kantengewicht. Von *6,8* zu *2,9* wurden mit 1.056 die meisten, von *6,8* zu *5* mit 31 die wenigsten Übergänge erkannt. Weiße Linien zeigen den kurvigen Wegverlauf, die blaue Linie zeigt die jeweils kürzeste Strecke zwischen den Bluetooth-Loggern entlang des Wegs.

Der Übergangsgraph zeigt die bekannten Bluetooth-Logger *1* und *5* sowie die aggregierten *2,9* und *6,8* an den gemittelten Positionen als Knoten. Zellen der Matrix T_{agg} bilden die Kantengewichte. So ist $\|T_{agg}\|_{max} = t_{42} = 1.056$ der Übergang mit den meisten Messungen von *6,8* zu *2,9* und $\|T_{agg}\|_{min} = t_{43} = 31$ der mit den wenigsten von *6,8* zu *5*. Diese Extremwerte lassen sich auf zwei Gründe zurückführen.

- Zum einen spiegelt sich hier die *Entfernung* der Bluetooth-Logger wider. Je näher Bluetooth-Logger sind, desto wahrscheinlicher ist es, dass Übergänge innerhalb der Gültigkeitsdauer einer MAC-Adresse gemessen werden können. Nach der Aggregierung ist die Kante mit dem Maximalwert von 1.056 Übergängen auch die geographisch kürzeste Strecke mit 187 m, die mit dem Minimalwert von 31 Übergängen die längste mit 430m (siehe D_{agg}). Dieses Muster ist nicht eindeutig. Die Kante von *5* zu *1* hat das viertgrößten Kantengewicht mit 259 Übergängen, aber den zweitlängsten Weg mit 370m. Dieses Beispiel wird durch den Wegverlauf im Testgelände untermauert. Die Strecke, die zurückgelegt werden muss, um von *5* zu *1* zu gelangen, ist deutlich länger als die angenommene Luftlinie (vgl. blaue Linie des Wegverlaufs in Abbildung 8.6).

- Zum anderen schlägt sich der ausgeschilderte *Rundgang* im Testgelände nieder. Vom Eingang führt ein breiter Weg von *6,8* zu *2,9*. Ab hier teilt sich der Weg in kleinere gewundene Wege. Abbildung 8.6 zeigt in blau die GPS-Bewegungsbahn des abgelaufenen Rundgangs. Während im ersten von *6,8* zu *2,9* ein gerader Weg gelaufen werden konnte, zeigen sich im weiteren Verlauf mehr Kurven und Windungen.

  Bzgl. des Minimalwerts führt der Weg von *6,8* zu *5* entlang des Rundgangs an *2,9* vorbei. So würde ein Bluetooth-Gerät, selbst wenn es innerhalb der Gültigkeitsdauer einer MAC-Adresse, von *6,8* zu *5* gelangte, als zwei Übergänge erkannt; von *6,8* zu *2,9* und von *2,9* zu *5*.

Der in Abbildung 8.5 bereits vermutete Rundgang entgegen des Uhrzeigersinns bestätigt sich hier in den errechneten Übergängen. Die Ausnahme bildet Knoten *2,9*, der ein höheres Kantengewicht entgegen des Rundgangs hat. Alle anderen Knoten folgen dem größten Kantengewicht entlang des Rundgangs.

Mithilfe der Bluetooth-Logger lässt sich der Besucherstrom im Testgelände vereinfacht darstellen. Von den 50.313 Messungen des 21. Augusts waren nur 10.829 für die Übergangserkennung geeignet. Eine größere Anzahl an Bluetooth-Loggern würde dieses Verhältnis vergrößern, weil häufiger an unterschiedlichen Stellen dieselbe MAC-Adresse gemessen würde und somit mehr Übergänge erkannt werden könnten.

Da die Übergangserkennung auf einer Teilmenge der insgesamt gemessenen MAC-Adresse beruht, ist der Extrapolationsfaktor aus Abschnitt 8.5 hier nicht gültig. Anteile an gehenden Geräten lassen sich aus T ableiten, für eine Schätzung der besuchenden Personen fehlt der Anteil bleibender Geräte.

8.6.2. Anteil bleibender Geräte

Eine randomisierte MAC-Adresse nicht mehr zu empfangen, ist ein mehrdeutiges Ereignis. Es kann bedeuten, dass das sendende Gerät den Aufzeichnungsbereich verlassen *oder* die MAC-Adresse durch eine neue ersetzt wurde. Gesucht ist eine Funktion, die eindeutig bestimmt, wie viele Geräte anteilig den Aufzeichnungsbereich eines Bluetooth-Loggers verlassen haben bzw. dort verweilen.

Wie in Abschnitt 7.2 beschrieben, werden anwesende Bluetooth-Geräte als aggregierte Messungen innerhalb eines zweiminütigen Zeitfensters in der Datenbank abgelegt. Der jeweilige Eintritts- und Austrittszeitpunkt des Gerätes je Bluetooth-Logger je Zeitfenster wird dabei mitgeschrieben. Ob ein Gerät über ein gesamtes Zeitfenster anwesend war, lässt sich anhand der Differenz von Eintritts- und Austrittszeitpunkt bestimmen. Berücksichtigend der Funktionsweise der Bluetooth-Logger und Messungenauigkeiten wird ein Schwellwert für diese Differenz festgelegt, ab dem das Zeitfenster als *ausgeschöpft* gilt. Der Anteil ausgeschöpfter an allen gemessenen Zeitfenstern wird interpretiert als Anteil bleibender an allen gemessenen Geräte. Dieses Verhältnis wird auf Basis der zweiminütigen Zeitfenster erstellt und ist somit zeitvariant.

Zunächst wird der Schwellwert bestimmt, ab dem ein Zeitfenster als ausgeschöpft gilt. Abbildung 8.7 zeigt die absolute Häufigkeit von Differenz aus Eintritts- und Austrittszeitpunkt für

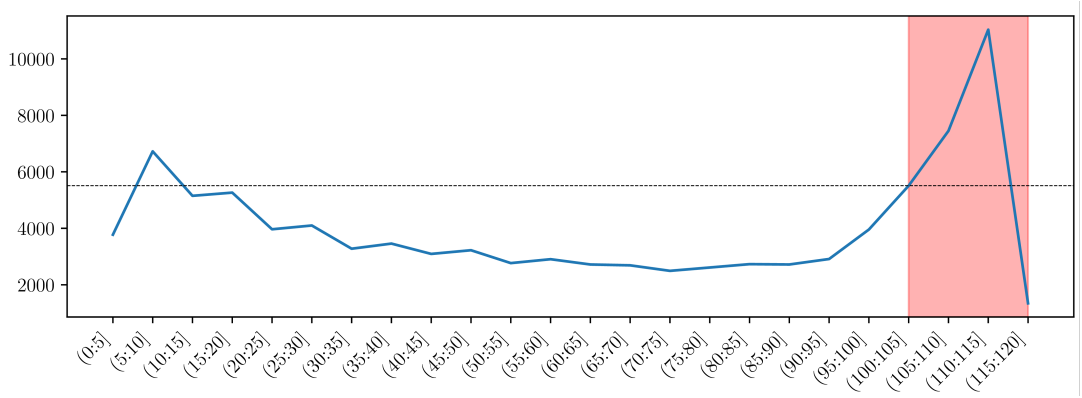


Abbildung 8.7.: Ausgeschöpfte Zeitfenster: Anzahl aller Messungen in Abhängigkeit der Differenz aus Eintritts- und Austrittszeitpunkt je Zeitfenster in Sekunden. Im markierten Intervall $(105 : 120]$ gilt ein Zeitfenster als *ausgeschöpft*. (Der Ausschlag bei null Sekunden mit 27.912 Messungen wurde hier ausgelassen.)

alle Zeitfenster des Testzeitraums aggregiert auf 5 Sekunden lange Intervalle. Es zeigt sich eine starke Häufung ab ca. 105 Sekunden Aufenthalt. Dies legt die Vermutung nahe, dass Geräte, die diesen Schwellwert überschreiten, im nächsten Zeitfenster ebenfalls erfasst werden. Dass die Werte bereits ab 105 Sekunden steigen, liegt an dem Scan-Verhalten der Bluetooth-Logger. Wie in Abschnitt 6.2 beschrieben, wird für zehn Sekunden nach *neuen* Geräten gescannt. Bereits erfasste Geräte werden innerhalb dieser zehn Sekunden nicht erneut gemeldet. Ein Erreichen von 120 Sekunden Differenz aus Eintritts- und Austrittszeitpunkt ist somit ausgeschlossen und erklärt die geringen Werte im letzten Zeitintervall. Unter Berücksichtigung, dass die exakte Scan-Zeit der Bluetooth-Logger mit der CPU-Auslastung variiert, wird der Schwellwert auf *105 Sekunden* festgelegt. Gemessene Geräte oberhalb dieses Schwellwerts gelten als *bleibend*.

Der Anteil bleibender Geräte ist je Bluetooth-Logger unterschiedlich. Mithilfe der SQL-Abfrage in Unterabschnitt A.4.3 werden die Anteile an ausgeschöpften Zeitfenstern bestimmt. Dabei ist der Grundwert die Summe aller Messungen je Zeitfenster je Bluetooth-Logger. Erfolgte in einem Zeitfenster keine Messung (bspw. nachts), wird dieses auch nicht berücksichtigt. Die folgende Tabelle 8.3 zeigt die ausgeschöpften Zeitfenster je Bluetooth-Logger und die daraus resultierende Wahrscheinlichkeit des Bleibens.

Für bspw. Bluetooth-Logger *6* wurden am 21. August von 28.609 Zeitfenstern 5.811 ausgeschöpft. Es bleiben alle zwei Minuten $\frac{5.811}{28.609} = 20,31\ \%$ der Geräte bei *6*. Die Gegenwahrscheinlichkeit bedeutet, dass 79,69 % der Geräte alle zwei Minuten Bluetooth-Logger *6* verlassen.

Die Wahrscheinlichkeit des Bleibens in Tabelle 8.3 wirkt auf den ersten Blick zu niedrig angesetzt. Einerseits könnten mit einer Anpassung des Schwellwerts, der entscheidet, ab wann ein Zeitfenster ausgeschöpft ist, die Wahrscheinlichkeiten korrigiert werden. Andererseits werden bei einer Geschwindigkeit von fünf Kilometern pro Stunde ca. 83 m pro Minute zurückgelegt. So ist es realistisch, dass 86 % der Besuchenden den Aufzeichnungsbereich von Bluetooth-Logger *1* innerhalb von zwei Minuten passieren.

Bluetooth-Logger	Ausgeschöpfte Zeitfenster	Summe Zeitfenster	Wahrscheinlichkeit des Bleibens je Zeitfenster	Gegenwahrscheinlichkeit
1	2.689	19.358	0,1389	0,8611
2	5.033	25.141	0,2002	0,7998
5	3.281	24.991	0,1313	0,8687
6	5.811	28.609	0,2031	0,7969
8	1.207	11.358	0,1063	0,8937
9	1.809	14.338	0,1262	0,8738

Tabelle 8.3.: Anteile ausgeschöpfter Zeitfenster je Bluetooth-Logger und die daraus errechnete Wahrscheinlichkeit des Bleibens bzw. Verlassens des Aufzeichnungsbereiches eines Bluetooth-Loggers.

Außerdem ist zu berücksichtigen, dass keine Filterung der Messungen vorgenommen wird (vgl. Kapitel 7). Stationäre Geräte oder Mitarbeitende beeinflussen die Wahrscheinlichkeit des Bleibens und könnten neben Besuchenden für die erhöhten Werte bei Bluetooth-Logger *2* und *6* verantwortlich sein.

8.6.3. Übergangswahrscheinlichkeiten

In der zu Beginn aufgestellten Matrix T in Gleichung (8.1) blieb die Hauptdiagonale offen. Mit der *Wahrscheinlichkeit des Bleibens* kann sie nun zu einer Übergangsmatrix komplettiert werden. Die Summe aus allen Übergängen von Bluetooth-Logger *6* beträgt 4.003 (Summe der Spalte 4 in T) und wird nun mit 79,69 % bewertet. So errechnet sich $p_{42} = \frac{311}{4.003} \cdot 79,69\ \% \approx 6,59\ \%$ der nachfolgenden Matrix P in Gleichung (8.3). Mit diesem Wert fließt die Abhängigkeit zum Zeitfenster von zwei Minuten ein. Analog für alle Bluetooth-Logger ergibt sich die folgende zeilenstochastische zeitvariante Übergangsmatrix:

$$P = \begin{array}{c} \\ \textit{1} \\ \textit{2} \\ \textit{5} \\ \textit{6} \\ \textit{8} \\ \textit{9} \end{array} \begin{array}{c} \begin{array}{cccccc} \textit{1} & \textit{2} & \textit{5} & \textit{6} & \textit{8} & \textit{9} \end{array} \\ \begin{pmatrix} 0,1392 & 0,0421 & 0,0599 & 0,2540 & 0,4498 & 0,0550 \\ 0,0122 & 0,2001 & 0,0697 & 0,0266 & 0,0458 & 0,6456 \\ 0,6226 & 0,0865 & 0,1322 & 0,0457 & 0,0889 & 0,0240 \\ 0,0098 & 0,0659 & 0,0042 & 0,2031 & 0,6464 & 0,0707 \\ 0,0209 & 0,1340 & 0,0080 & 0,5682 & 0,1059 & 0,1629 \\ 0,0061 & 0,7933 & 0,0100 & 0,0223 & 0,0422 & 0,1262 \end{pmatrix} \end{array} \tag{8.3}$$

Eine Zeile dieser Matrix gibt für alle zwei Minuten die Wahrscheinlichkeit eines Geräts des Bleibens bzw. des Übergangs von einem Bluetooth-Logger zum anderen an.

Es ist zu beachten, dass Werte der Matrix P auf der Grundlage des 21. Augusts berechnet wurden. So repräsentiert P die Besucherströme dieses Tages. Um die Übergänge zwischen den Bluetooth-Loggern repräsentativer wiederzugeben, könnte der Berechnungszeitraums auf mehrere Tage ausgeweitet werden. Allerdings spielen weitere Faktoren dabei eine Rolle. so, wie sich die Stoßzeiten zwischen Sonntag und Montag (vgl. Abbildung 8.3) unterscheiden, werden sich auch die Übergänge an diesen Tagen signifikant unterscheiden. Eine aggregierte

Übergangsmatrix je Wochentag bzw. je Tageszeit könnte hier weiterhelfen. Einen weiteren Einfluss stellt das Wetter dar. So suchen Besuchende bei Regen einen Unterstand oder bei Sonne Schatten und beeinflussen dadurch die Übergänge. Das Wetter zu klassifizieren und anhand dessen aggregierte Übergänge zu bestimmen, wäre eine weitere Lösung für diesen Ansatz.

Mit der Matrix P als Grundlage könnten die Ansätze von [Hu+12] und [HDC18] zur Vorhersage von Besucherströmen umgesetzt werden. Um eine Idee davon zu bekommen, ist ein erster Modellversuch in Abschnitt A.5 beschrieben. Dieser ist aber nicht Bestandteil der hier beschriebenen Besucherstromanalyse.

Mithilfe des Extrapolationsfaktors b (vgl. Abschnitt 8.5) können nun je Übergang Besucherzahlen geschätzt werden. Dabei ist die Schätzung besuchender Personen je Bluetooth-Logger die Grundlage. Diese wird mit nach den Wahrscheinlichkeiten aus P auf die Kanten des Übergangsgraphen verteilt. So kann der Übergangsgraph in Abbildung 8.6 mit geschätzten Besucherzahlen versehen werden.

Hiermit ist die dritte Frage der Betreibenden beantwortet: „Welche Route nehmen die Besuchenden im Gelände? Folgen sie dem beschilderten Rundgang?“ Wie in Unterabschnitt 8.6.1 diskutiert, spiegelt der Übergangsgraph den Rundgang vereinfacht wider. Mithilfe der Wahrscheinlichkeit des Bleibens kann dieser Graph mit Besucherzahlen geschätzt werden und gibt Auskunft darüber, wie viele Besuchende sich wohin bewegen.

8.7. Feldvergleich und Abdeckungskarte

Um die Reichweite der Bluetooth-Logger zu ermitteln und die Akkuratesse der Übergangserkennung zu überprüfen wurde ein Feldvergleich durchgeführt. Hierfür wurde ein Bluetooth-Sender mit öffentlicher MAC-Adresse eingesetzt. Messungen dieses Gerätes lassen sich mithilfe des MAC-Hashes identifizieren. Mittels GPS-Ortung wird der aktuelle Standort des Senders bestimmt und über den Weg durch das Testgeländes protokolliert. So lassen sich Tupel aus im System eingegangener Messung und Koordinaten des Senders zu einem bestimmten Zeitpunkt kombinieren. Hieraus lässt sich die Luftlinie zwischen der Position des Senders und der des Bluetooth-Loggers berechnen. Der Maximalwert der Abstände je Bluetooth-Logger wird als Reichweite interpretiert. Die resultierende Abdeckungskarte ist in Abbildung 8.8 dargestellt. Da dieser Feldvergleich außerhalb des in Kapitel 8 definierten Testzeitraums stattfand, weichen einige Bluetooth-Logger-Standorte zu den vorherigen Darstellungen ab. Diese sind im Folgenden mit einer Prim gekennzeichnet.

In der Tabelle findet sich der Ausreißer *2* mit einer maximalen Reichweite von nur elf Metern. Dies ist auf die Abschirmung des Bluetooth-Logger zurückzuführen. Im Fall von *2* liegt dieser in einer Nische neben viel Metall, dessen schirmender Effekt den Empfangsradius reduziert.

Ein zweiter Effekt ist die Verdeckung der Bluetooth-Logger. Dieser lässt sich deutlich bei *3′* erkennen. Messungen werden nördlich nicht aber südlich von Bluetooth-Logger *3′* verzeichnen. Während dieser „freie Sicht“ nach Norden hat, besteht nach Süden eine Verdeckung durch Gebäude und Bäume. Die Abdeckungskarte nähert den Empfangsbereich mit einem Kreis, mit dem Wissen über die Verdeckung wäre ein Halbkreis korrekt. Bei *6* und *10′* hingegen sind

Bluetooth-Logger	Maximale Reichweite
1	20,6 m
2	11,7 m
3′	37,4 m
5	22,6 m
6	18,0 m
7′	24,4 m
9′	29,5 m
10′	28,3 m
∅	24,1 m

Abbildung 8.8.: Abdeckungskarte aus Feldvergleich und Reichweitenbestimmung. Die blaue Linie zeigt den abgelaufenen Weg. Graue Punkte stellen Standorte entlang dieses Weges dar an denen eine Messung im System einging. Der Standort mit dem jeweils größten Luftlinienabstand zum Bluetooth-Logger ist beschriftet. Diese maximalen Abstände sind in der Tabelle zusammengefasst. Die blauen Kreise stellen den Empfangsbereich der Bluetooth-Logger dar. Dabei ist der Radius die maximale Reichweite.

Messungen nur in der einen Hälfte des Kreises zu sehen, da Bereiche der anderen Hälfte nicht zugänglich sind.

Die Abweichungen der Reichweiten aller Bluetooth-Logger lassen sich auf eine Mischung aus Abschirmung und Verdeckung zurückführen. Die durchschnittliche maximale Reichweite beträgt 24 Meter. Neben der Abdeckungskarte bildet die Reichweite eine wichtige Kennzahl bei der Bestimmung des Standorts zusätzlicher Bluetooth-Logger.

Übergänge des Bluetooth-Senders werden mithilfe des MAC-Hashes der öffentlicher MAC-Adresse identifiziert. Diese sind mit dem jeweiligen Eintritts- und Austrittszeitpunkt versehen in Tabelle 8.4 aufgeführt. Der Feldvergleich startete bei Bluetooth-Logger *7′*. Der Aufzeichnungsbereich wird um 12:25 Uhr verlassen, der von *10′* zwei Minuten später erreicht. Nach einem Aufenthalt dort von 22 Minuten wird Bluetooth-Logger *10′* verlassen und zu *5* gewechselt. Diese Daten würden in der Übergangsmatrix die Übergänge von *7′* zu *10′*, von *10′* zu *5*, etc. jeweils um eins inkrementieren.

Die Messungen des Systems in Tabelle 8.4 geben akkurat die abgelaufene Strecke in Abbil-

von		nach	
Bluetooth-Logger	Austrittszeitpunkt	Bluetooth-Logger	Eintrittszeitpunkt
7′	12:24:00	*10′*	12:26:00
10′	12:48:00	*5*	12:50:00
5	12:50:00	*9′*	12:52:00
9′	12:52:00	*3′*	12:54:00
3′	12:56:00	*1*	13:00:00
1	13:02:00	*6*	13:14:00
6	13:14:00	*2*	13:16:00
2	13:18:00	*7′*	13:22:00

Tabelle 8.4.: Errechnete Übergänge des Feldvergleichs. Start- und Endpunkt bildet dabei der Bluetooth-Logger *7′*.

dung 8.8 wieder. Dies stellt sowohl die Funktion des Systems, als auch die Übergangserkennung unter Beweis.

8.8. Wert der CWA-Messungen

Im Folgenden wird überprüft, ob sich die Daten der CWA für die Besucherstromanalyse eignen. Unter Berücksichtigung der in Absatz 8.2 diskutierten fehlerhaften Annahme, stammen *mindestens* 3,4 Mio. Messungen von der CWA. Mit ausschließlich diesen Messungen, wird diese Teilmenge auf ihre Aussagekraft geprüft.

Stoßzeiten Die Abbildung 8.3 zeigt den Anteil der CWA-Messungen im Vergleich zu allen Messungen. Am Dienstag findet sich eine Abweichung des Tagesmaximums. Während dies auf Grundlage aller Messungen in der Stunde von 12:00 bis 12:59 Uhr liegt, geben die CWA-Messungen das Tagesmaximum bei 11:00 bis 11:59 Uhr an. Die Stundensummen von elf und zwölf weichen dabei um zwei Prozent bei allen und um drei Prozent bei den CWA-Messungen ab. An den anderen Wochentage hebt sich das Tagesmaximum deutlicher ab. Die Zuordnung des Tagesmaximums am Dienstag sollte aufgrund der geringen Differenzen auf beide Stunden (11:00 bis 11:59 Uhr *und* 12:00 bis 12:59 Uhr) erfolgen, da diese ausgeglichen besucht erscheinen. Abgesehen von dieser Abweichung zeigt sich eine identische Verteilung. Bezogen auf die Stoßzeiten sind die CWA-Messungen ebenso aussagekräftig wie die Summe aller Messungen.

Anzahl Geräte je Bluetooth-Logger Die Anzahl an Geräten je Bluetooth-Logger aus Abschnitt 8.5 wird in Abbildung 8.9 erneut verwendet. Zu den roten Kreisen, die die Anzahl aller Geräte repräsentieren, kommen gelbe Kreise hinzu, die für die Anzahl an CWA-Geräten stehen. Durch die logarithmische Skalierung der Kreisradien in Abhängigkeit von der Geräteanzahl unterscheidet sich das entstehende Bild kaum von Abbildung 8.4. Aus der nebenstehenden Tabelle ergibt sich der jeweilige Anteil der CWA-Geräte an allen gemessenen Geräten

Bluetooth-Logger	Alle Geräte	CWA-Geräte	CWA-Anteil
1	7.039	1.764	25 %
2	9.398	2.308	25 %
5	8.902	2.235	25 %
6	10.809	2.770	26 %
8	6.533	1.874	29 %
9	7.631	2.073	27 %
$\sum$	50.312	13.024	26 %

Abbildung 8.9.: Anzahl aller und CWA-Geräte je Bluetooth-Logger im Vergleich aggregiert über den ganzen Tag des 21.08.2022. Die Beschriftung in der Grafik links setzt sich zusammen aus der Nummer des Bluetooth-Loggers, der Anzahl aller gemessener Geräte sowie der CWA-Geräte in Klammern. Rote Kreise repräsentieren die Anzahl aller, gelbe die der CWA-Geräte. Die Kreisradien skalieren logarithmisch, so ergeben sich ähnliche Kreisgrößen trotz großer Wertdifferenzen.

je Bluetooth-Logger. Dieser beträgt am 21.08.2022 im Mittel 26 % und passt zu dem in Abschnitt 8.2 diskutierten Anteil über den gesamten Testzeitraum. Durch die Stabilität dieses Anteils können Hochrechnungen auf Besucherzahlen, auf Basis der CWA-Geräte, analog zu der auf Basis aller Geräte erfolgen.

Übergangserkennung Während der Anteil CWA-Geräte je Bluetooth-Logger stabil bei durchschnittlich 26 % liegt, liegen die Anteile der CWA-Übergänge je Kante zwischen 2 % und 31 %. Dies lässt an der Aussagekraft der CWA-Übergänge zweifeln. Folgende Gründe für die geringen Anteile werden identifiziert:

- Die in Absatz 8.2 erklärte *fehlerhafte Annahme* kann Ursache dafür sein. Sendet ein Gerät neben Advertisements der CWA zusätzlich Advertisements eines anderen Services, wird nur einer dieser beiden Services erkannt (der, der zuerst empfangen wurde). Somit können Advertisements, die von der CWA stammen fälschlicherweise von dem anderen Service verdeckt werden. Bei der Filterung nach dem Wahrheitswert *isCwa* (vgl. Abschnitt 7.1) werden möglicherweise CWA-Daten ausgefiltert. Dieser Fehler liegt in der Firmware der Bluetooth-Logger und später in der Datentransformation. Die bereits vorgestellte Maßnahme, Messungen anhand des Tupels aus MAC-Adresse und Service-Data zu identifizieren, anstelle nur der MAC-Adresse, würde diesen Fehler ausschließen.
- Die *geringe Sendeleistung* der Advertisements der CWA kann ein weiterer Grund sein. Seitens des ENFs kann die Sendeleistung zum Schutz vor Nachverfolgung variiert werden

[vgl. AG20, S. 4]. Sendet ein Gerät schwächer, ist es unwahrscheinlich auf dem gesamten Aufzeichnungsbereich des Bluetooth-Loggers gemessen zu werden. Eine Maßnahme zur Verbesserung wäre, die Dichte der Bluetooth-Logger zu erhöhen. Kürzere Abstände zwischen den Bluetooth-Logger sorgen für mehr Messungen und häufigeres Wiederfinden von CWA-Geräten.

- Den letzten Grund stellt die *Gesamtanzahl der Daten* dar. Die Übergänge wurden auf Basis eines Tages berechnet. Von 39.484 gemessenen Geräten konnten nur 2.684 für die aggregierte Übergangserkennung genutzt werden. Der CWA-Anteil dieser beträgt mit 747 Messungen ca. 28 % und wird nun auf die zwölf Kanten des Übergangsgraphen verteilt. Um repräsentative Anteile zu bestimmen sind das zu wenig Messungen. Den Zeitraum der Übergangserkennung auf CWA-Basis zu vergrößern, könnte zu besseren Aussagen führen.

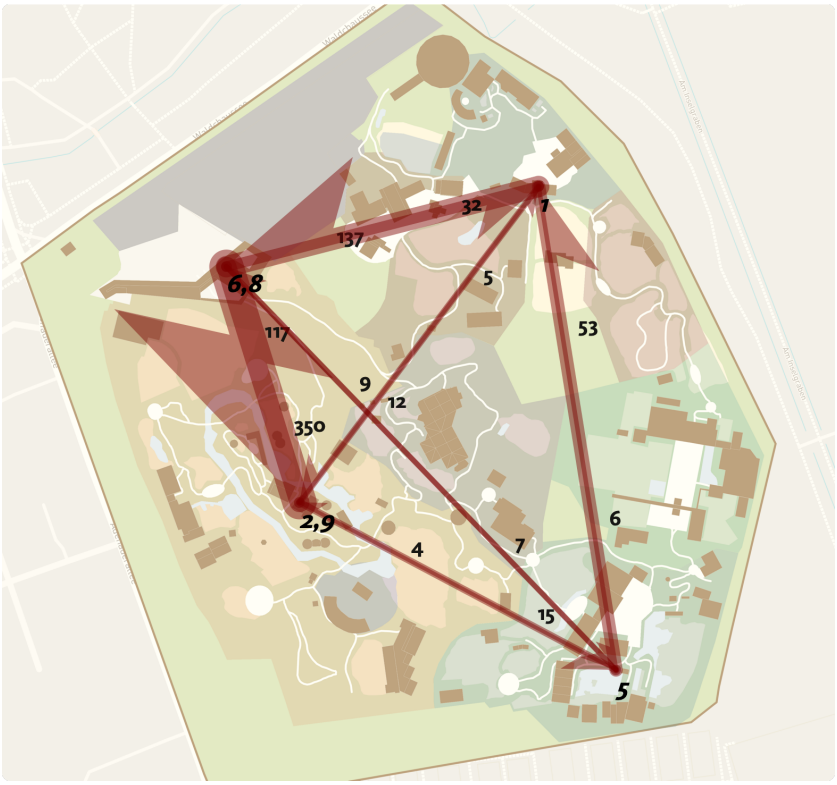


Abbildung 8.10.: Übergangsgraph der CWA-Messungen des 21.08.2022. Von *6,8* zu *2,9* wurden mit 350 die meisten Übergänge erkannt. Die Kante von *5* zu *2,9* zeigt die geringsten Übergänge mit einem Gewicht von vier.

Der Übergangsgraph aller Übergänge aus Abbildung 8.6 ist in Bezug auf die CWA-Messungen in Abbildung 8.10 erneut dargestellt. Trotz der diskutierten abweichenden Anteile ähneln sich die Übergangsgraphen relativ. Hohe Kantengewichte zwischen *6,8* und *2,9* fallen hier etwas stärker aus. Wie in Unterabschnitt 8.6.1 diskutiert, folgen – abgesehen von dieser Kante – die jeweils größten Kantengewichte dem Rundgang entgegen des Uhrzeigersinns.

Kurz gesagt lassen sich exklusiv auf den Daten der CWA Stoßzeiten und hoch frequentierte

Bereiche ebenso gut identifizieren wie auf der gesamten Datenmenge. Eine Hochrechnung auf Besucherzahlen kann analog erfolgen. Bei der Übergangserkennung von CWA-Daten weichen die Ergebnisse durch Ungleichmäßigkeit ab. Die Extrapolation der CWA-Übergänge auf Besucherzahlen entspricht nicht der benötigten Aussagekraft. Insgesamt bilden die Daten der CWA eine Anteil von etwa 25 %. Die CWA-Daten stellen in jeder den o.g. Auswertungen eine gute Ergänzung dar.

9. Fazit

Unter Berücksichtigung der Schutzziele der IT-Sicherheit wurde eine Systemarchitektur entwickelt. Das Kommunikationsprotokoll MQTT stellt sicher, dass vertrauliche Messdaten nur von autorisierten Rollen innerhalb des Systems gelesen und verarbeitet werden dürfen. Messdaten der Bluetooth-Logger werden verschlüsselt an den authentisierten Server geschickt. So wird die – seitens der DSGVO geforderte – Vertraulichkeit geschützt sowie einem Man-in-the-Middle-Angriff vorgebeugt. Die Firmware gestaltet sich modular und erweiterbar und lässt sich kabellos aktualisieren. Auf Basis eines Mikrocontrollers wurde eine Hardware zur Erfassung von Bluetooth-Advertisements entwickelt.

Der Aufbau von neun Bluetooth-Loggern auf dem Testgelände stellte sich als schwieriger heraus als erwartet. Eine gleichmäßige Verteilung über das Gelände ließen die Bedingungen an den Standort sowie die Infrastruktur vor Ort nicht zu. Dadurch wurden einige Bereiche gut abgedeckt, andere nur sporadisch bis gar nicht. In den nicht abgedeckten Bereichen gingen Daten für die Auswertung verloren.

Die Verfügbarkeit der Bluetooth-Logger war schlechter als erwartet. Von den neun aufgestellten Bluetooth-Logger sendeten nur vier zuverlässig Daten über den gesamten Testzeitraum von sieben Wochen. In den meisten Fällen sind die Ausfälle auf eine schlechten WLAN-Abdeckung zurückzuführen. Sowohl die Schwierigkeiten beim Platzieren, als auch die geringe Verfügbarkeit legen den Schluss nahe, eine andere Technologie zur Übertragung der Messdaten zu wählen. Als alternative Übertragungstechnologien kämen *LoRaWAN*[1] oder Mobilfunk in Betracht. Die gesendete Datenmenge aller Bluetooth-Logger lag pro Tag bei ca. 51 MB.

Im System eintreffende Messungen werden an erster Stelle klassifiziert. Sowohl der Versuch, Messdaten eindeutig einem Hersteller zuzuordnen, als auch sie repräsentativ nach Gerätetypen zu klassifizieren, schlug fehl. Das damit angedachte Erkennen von Besucherprofilen war somit nicht möglich. Der darauffolgenden Besucherstromanalyse legt dies keinen Stein in den Weg. Durch die Aggregierung der Messdaten in zweiminütige Zeitfenster wurde die Datenmenge auf 28 % reduziert. Diese Datenstruktur wird in der Datenbank abgelegt. Hier wäre eine Filterung der Daten wünschenswert. Mitarbeitende sowie stationäre Geräte verfälschen hier die Messergebnisse. Dieser Fehler fällt kleiner aus, je mehr Besuchende sich auf dem Testgelände aufhalten.

Die anschließende Datenauswertung zeigt effizient die Stoßzeiten und die gemessenen Gerätezahlen je Bluetooth-Logger aus der Datenbank. Der Vergleich berechneter Stoßzeiten mit denen von Google verifiziert die Ergebnisse und somit die virtuelle Repräsentation der Messdaten. Mithilfe der Anzahl gemessener Geräte je Bluetooth-Logger ergibt sich ein Bild der

[1] Long Range Wide Area Network, kurz LoRaWAN, ist ein Funkprotokoll, das häufig für entfernte Sensoren eingesetzt wird. Mehr dazu unter: `https://www.lora-wan.de`

Besucherverteilung auf dem Testgelände. Diese Auswertung kann sowohl in Echtzeit als auch auf Stunden oder Tage in der Vergangenheit aggregiert effizient erfolgen.

Die Berechnung der Übergänge hingegen ist zeitintensiv, was an den Eigenschaften der relationalen Datenbank liegt. Hier könnte eine *Zeitreihendatenbank* effizientere Ergebnisse liefern. Resultierende Übergangsgraphen bestätigen die Erwartungen bzgl. der Route von Besuchenden entlang des Rundgangs. Im Verhältnis zur Gesamtzahl gemessener Geräte werden insgesamt wenig Übergänge erkannt. Dies liegt an der geringen Abdeckung des Testgeländes und den Bedingungen an den Standorten. Um die Übergangsmatrix zu vervollständigen, wurde eine Funktion zur Ermittlung der Wahrscheinlichkeit des Bleibens definiert. Diese ist anfällig auf ungefilterte stationäre Geräte und Mitarbeitende. Der hier definierte Schwellwert, ab dem ein Zeitfenster als ausgeschöpft gilt, ist als Variable zu verstehen. Die Übergangsmatrix wurde bisher für einen Tag errechnet. Mit einer Ausweitung des Zeitraums sowie einer größeren Anzahl an Bluetooth-Loggern wäre die zugrundeliegende Datenmenge größer und es könnten genauere Werte bestimmt werden.

Der Feldvergleich stellt die korrekte Überführung von gemessenen physischen Geräten in ihre virtuelle Repräsentation unter Beweis. Vom System errechnete Übergänge stimmen mit den tatsächlichen überein. Unter Berücksichtigung der diskutierten Verdeckung und Schirmung stellen die resultierende Abdeckungskarte und die Reichweiten der Bluetooth-Logger wichtige Werkzeuge für die Bestimmung strategisch guter Standorte weiterer Bluetooth-Logger dar. Die Abdeckungskarte zeigt, wie klein die Bereiche sind, die von den Bluetooth-Loggern abgedeckt werden. Vor diesem Hintergrund ist die Aussagekraft der Auswertungen sehr hoch zu bewerten.

9.1. Beantwortung der Forschungsfragen

Die Besucherstromanalyse von Versichele u. a. im Jahr 2012 beruhte auf öffentlichen MAC-Adressen weniger Bluetooth-fähiger Geräte. Heute sind weit mehr Geräte im Umlauf, allerdings nutzen diese randomisierte MAC-Adressen. Hieraus ergab sich die erste Forschungsfrage:

1. Wie können Besucherströme anhand von Bluetooth-Geräten, die eine randomisierte MAC-Adresse nutzen, ermittelt werden?

Aus dem abstrakten Versuchsaufbau wurde eine IoT-Systemarchitektur aufgebaut (vgl. Kapitel 4). Für die Aufzeichnung von Bluetooth-Messungen wurde in Abschnitt 6.1 eine Firmware für den Mikrocontroller ESP32 entwickelt. Unter Berücksichtigung des BDSG wurde ein sicherer Transfer (vgl. Kapitel 5) und die Anonymisierung der Messdaten umgesetzt (Abschnitt 7.1). Aufbauend darauf zeigen die Ergebnisse der Übergangserkennung in Abschnitt 8.6, dass Besucherströme von Geräten, die randomisierte MAC-Adressen nutzen, ermittelt werden können. Auf Grundlage dieser lassen sich Übergänge von Besuchenden schätzen.

Für das Testgelände lagen bisher Erfassungen von Besucherströmen auf Basis von Umfragen vor. Nach der Automatisierung dieser durch Bluetooth-Logger stellte sich die zweite Forschungsfrage:

2. Wie können im Erlebniszoo Hannover Stoßzeiten, hoch frequentierte Orte und Besucherströme bestimmt werden?

Diese wurde mithilfe der drei Fragen der Betreibenden beantwortet:

- Um welche Tageszeit und an welchem Wochentag sind viele Besuchende vor Ort?
  Zunächst wurde die Hardware im Testgelände aufgestellt (vgl. Abschnitt 6.4). Aggregiert nach Wochentag und Tageszeit wurden die Messungen aller Bluetooth-Logger als Stoßzeiten herausgestellt (vgl. Abschnitt 8.4), diese beantworteten die Frage.
- Welche Orte sind stark bzw. schwach besucht?
  In Abschnitt 8.5 wurde die Anzahl gemessener Geräte je Bluetooth-Logger ausgewertet. Diese Auswertung ist sowohl in Echtzeit als auch als Aggregat beliebiger Zeitintervalle der Vergangenheit möglich. Mithilfe eines Extrapolationsfaktors lässt sich von der Anzahl der gemessenen Geräte auf Besucherzahlen schließen.
- Welche Route nehmen die Besuchenden im Gelände? Folgen sie dem beschilderten Rundgang?
  Die Übergangserkennung wurde in Abschnitt 8.6 erarbeitet. Das Ergebnis ist eine Übergangsmatrix, die sowohl die Wahrscheinlichkeit des Bleibens enthält (vgl. Unterabschnitt 8.6.2), als auch eine Wahrscheinlichkeitsverteilung des Gehens zu anderen Orten beinhaltet (vgl. Unterabschnitt 8.6.1). Aus dem erarbeiteten Übergangsgraphen lässt sich erkennen, wie viele Geräte in einem gesetzten Zeitintervall sich von einem Ort zum anderen bewegen. Dabei folgt der Großteil der gemessenen Geräte dem vereinfachten Rundgang. Da die Werte der Übergangsmatrix Wahrscheinlichkeiten sind, lassen sich diese auf Besucherzahlen projizieren.

Zu den Ergebnissen der Besucherstromanalyse im Testgelände gehören trotz randomisierter MAC-Adressen Stoßzeiten, hoch frequentierte Orte in Echtzeit und Besucherströme.

Etwa jede vierte Person in Deutschland nutzt die CWA. Da diese mehrmals pro Sekunde spezielle Bluetooth-Advertisements sendet, stellte sich die dritte Forschungsfrage:

3. Können die Daten der CWA (ergänzend oder ausschließlich) für eine Besucherstromanalyse genutzt werden?

Der Anteil der CWA-Messungen liegt bei ca. 25 % aller Messungen (vgl. Abschnitt 8.2). Dieser Wert deckt sich mit den fast 30 % aktiven Nutzenden der CWA in Deutschland. Der Wert der CWA-Daten wurde in Abschnitt 8.8 diskutiert. Ausschließlich mit CWA-Daten lassen sich Stoßzeiten und Gerätezahlen je Bluetooth-Logger zuverlässig bestimmen und bieten eine Grundlage, um auf Besucherzahlen zu schließen. Für die Übergangserkennung zeigen sich exklusiv die Daten der CWA als zu ungenau. Ergänzend sind die CWA-Daten für die Auswertung eine Bereicherung.

Und sollte die CWA eines Tages eingestellt werden, stehen 75 % der Bluetooth-Messungen, für die hier beschriebene Besucherstromanalyse zur Verfügung.

9.2. Übertragbarkeit und Ausblick

Das WLAN des Testgeländes hat sich als ungeeignete Übertragungstechnologie herausgestellt. Mit einer der genannten Alternativen könnte erstens eine bessere Verfügbarkeit der Bluetooth-

Logger hergestellt werden und zweitens eine flexiblere Platzierung der Bluetooth-Logger auf dem Gelände erfolgen.

Gleichmäßige Abstände zwischen den Bluetooth-Loggern sorgen für eine zuverlässige Erkennung von Übergängen. Die durchschnittlichen Reichweite von 24 m der Bluetooth-Logger gibt für eine vollflächige Abdeckung einen Abstand von etwa 50 m vor. Je nach Gelände kann dieser vergrößert werden. Die Ergebnisse dieser Arbeit zeigen, dass bei einem Abstand von ca. 300 m auf dem Testgelände eine robuste Übergangserkennung stattfinden kann. Auf einem Gelände, in dem die Durchschnittsgeschwindigkeit der Besuchenden hoch ist (bspw. Innenstadt), kann der Abstand vergrößert werden.

Bluetooth-Logger können in einem anderen Außengelände eingesetzt werden. Unter der Annahme, dass die Genauigkeit der Positionsbestimmung auf den Aufzeichnungsbereich eines Bluetooth-Loggers beschränkt ist, können hier beschriebene Verfahren analog genutzt werden. In einer Messehalle, die vergleichbare Bedingungen hat, ist der Einsatz ebenfalls denkbar. Für den Innenbereich ist die Positionsbestimmung i. d. R. zu ungenau. Außerdem herrschen andere Einflüsse durch bspw. Wände, Stockwerke oder Treppenhäuser. Hier sollte sich an den Ansätzen aus Kapitel 2 für den Innenbereich orientiert werden.

Die erarbeitete Besucherstromanalyse bildet die Grundlage für weitere Berechnungen im Kontext des Besuchermanagement. Den weiterführenden Ansätzen aus Kapitel 2 folgend, könnte die Übergangsmatrix zur Modellierung der Besucherströme eingesetzt werden, um Vorhersagen über Besucherzahlen zu treffen [vgl. Hu+12; HDC18].

Je mehr Daten kombiniert werden, desto genauer werden die Vorhersagen sein. So könnten neben den Messungen der Bluetooth-Logger, Ferienzeiten und Feiertage, Wetterdaten oder Events in der Nähe einbezogen werden. Die, auf dieser Basis getroffene Vorhersagen ermöglichen es, einzelnen Besuchenden einen alternativen Rundgang vorzuschlagen, um Häufungspunkte zu vermeiden.

A. Anhang

A.1. GitHub-Repositories

Die im Rahmen dieser Arbeit entwickelte Software steht unter folgenden Links auf GitHub unter der MIT Lizenz zur Verfügung:

- Die Firmware der Bluetooth-Logger, wie sie in Abschnitt 6.1 beschrieben ist: `https://github.com/kiliandangendorf/esp32-bluetooth-logger`
- Die vollständige Konfigurationsdateien des Servers (Abschnitt 4.2 und Unterabschnitt 6.2.3) und der Quellcode der Datenspeicherung (Kapitel 7) mit Betriebsanleitung: `https://github.com/kiliandangendorf/crowd-flow-analysis-with-esp32-bluetooth-logger`

A.2. Einblick in die Betriebsschicht

Ein Ausschnitt des Frontends der Betriebsschicht ist in Abbildung A.1 dargestellt. Hier zeigt sich eine Liste aller dem System bekannten Bluetooth-Logger. Nach Auf- bzw. Umstellung der Bluetooth-Logger wird hier der Standort als Längen- und Breitengrad festgelegt bzw. geändert. Änderungen werden in der Tabelle *stations* des Datenbankmodells in Abbildung 7.3 vorgenommen. Zum Schutz der Datenintegrität ist neben den transformierten Messdaten, die vom Dienst *Aggregierung* in der Datenbank abgelegt werden, dies die einzige Stelle an der Daten verändert werden (vgl. Abschnitt 4.3).

Um die in Abschnitt 8.1 diskutierte Verfügbarkeit zu Überwachen, werden aus den Statusnachrichten sowie der Will-Messages der Bluetooth-Logger der Verbindungsstatus *online* bzw. *offline* abgeleitet und angezeigt (vgl. Abschnitt 6.2).

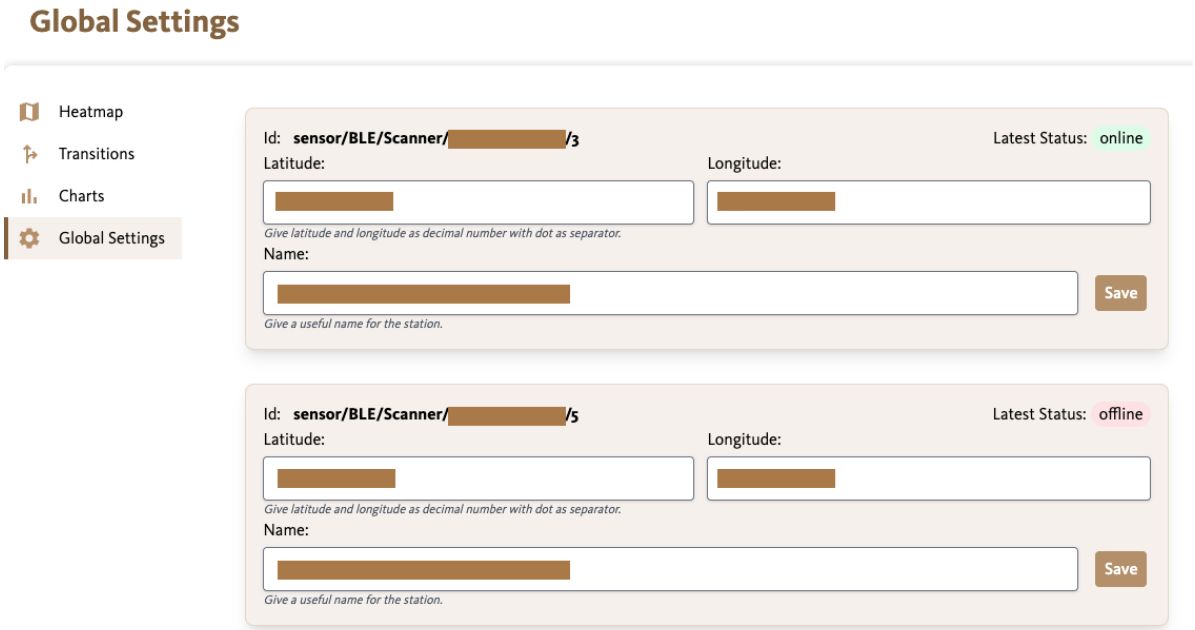


Abbildung A.1.: Einblick in die Betriebsschicht. Je Bluetooth-Logger wird der Verbindungsstatus (*online*, *offline*) angezeigt und es können Einstellungen am Standort in Form Längen- und Breitengrad vorgenommen werden.

A.3. Klassifizierung anhand des Bluetooth-Gerätenamens

Von 2.317.840 Messungen im Testgelände[1] enthielten 57.648 einen Gerätenamen, das sind etwa 2,5 %. Darunter befanden sich 2.462 unterschiedliche Gerätenamen. Durch einen Stringvergleich konnten 54,8 % der Gerätenamen folgenden Gerätetypen zugeordnet werden.

Zugeordneter Gerätetyp	Schlüsselwörter	Beispiele für Gerätename	Anzahl	Anteil
Smartwatch	watch, gear, coros, ...	Galaxy **Watch** [...], HUAWEI **WATCH** [...], Xiaomi**Watch**S1Active [...], **Gear** S3 [...] LE, **COROS** PACE 2 [...]	683	27,7 %
Fitnesstracker	band, fit, heart, ...	Mi Smart **Band** [...], HUAWEI **Band** [...], Galaxy **Fit**2 [...], RB09_**Heart**	450	18,3 %
Glukosesensor	dexcom	**Dexcom**[...]	90	3,7 %
Kopfhörer und Soundbox	bose, buds, sound, beoplay, ...	LE-**Bose** Ear**buds**, Raycon [...] Ear**buds** BLE, **Sound**Core mini, LE-**Beoplay** EX	67	2,7 %
Schlüsselfinder	tag	**Tag**-It, Smart **Tag**	28	1,1 %
Smartphone	phone	i**Phone** von [...], i**Phone** (96)	14	0,6 %
E-Roller	scooter, lime, bird	NB**Scooter**[...], **lime**-[...], **BIRD**	9	0,4 %
Kamera	leica, gopro	**Leica** Q2-[...], **GoPro** [...]	8	0,3 %
Sonstige		nut, S1e9[...]3dC	1.113	45,2 %
$\sum$			2.462	

Tabelle A.1.: Klassifizierung anhand des Bluetooth-Gerätenamens. Seriennummern und Namen von Personen sind ausgelassen, Schlüsselwörter hervorgehoben.

[1] Messungen in einem anderen Gelände können zu anderen Ergebnissen führen.

A.4. SQL-Abfragen

A.4.1. Bestimmung der Stoßzeiten

Die Anzahl unterschiedlicher MAC-Hashes des gesamten Testzeitraums wird nach Wochentag und Stunde der Aufzeichnung aufsummiert. Dabei wird die Tageszeit auf den Bereich der Öffnungszeiten des Testgeländes beschränkt. Das Ergebnis ist in Abbildung 8.3 visualisiert.

```
SET @time_from = '2022-08-12 00:00:00.0';
SET @time_to = '2022-09-29 23:59:59.9';

SELECT DAYNAME(time_window) AS weekday
  , HOUR(time_window) AS daytime
  , COUNT(DISTINCT mac_hash) AS devices_count
FROM ble_meas
WHERE timewindow IN (SELECT time_from
                     FROM ble_timewindows
                     WHERE time_from >= @time_from
                       AND time_to <= @time_to)
  AND HOUR(timewindow) >= 9
  AND HOUR(timewindow) < 19
GROUP BY weekday, daytime
ORDER BY CASE weekday
            WHEN 'Monday' THEN 1
            WHEN 'Tuesday' THEN 2
            WHEN 'Wednesday' THEN 3
            WHEN 'Thursday' THEN 4
            WHEN 'Friday' THEN 5
            WHEN 'Saturday' THEN 6
            WHEN 'Sunday' Then 7
    END
        ASC, daytime;
```

A.4.2. Anzahl Geräte je Bluetooth-Logger

Für jeden Bluetooth-Logger wird für den gesamten Tag des 21.08.2022 die Anzahl unterschiedlicher MAC-Hashes aufsummiert. Durch den Verbund enthält das Ergebnis den Standort des Bluetooth-Loggers für die Visualisierung. Die Abbildungen 8.4 und 8.5 wurden mithilfe dieser Abfrage erstellt.

```
SET @time_from = '2022-08-21 00:00:00.0';
SET @time_to = '2022-08-21 23:59:59.9';

SELECT *
FROM (SELECT station, COUNT(DISTINCT mac_hash) AS devices_count
      FROM ble_meas
      WHERE time_window IN (SELECT time_from
                            FROM ble_timewindows
                            WHERE time_from >= @time_from
                              AND time_to <= @time_to)
      GROUP BY station) AS stations_devices_count
         JOIN stations
              ON stations_devices_count.station = stations.id;
```

A.4.3. Ausgeschöpfte Zeitfenster je Bluetooth-Logger

Ein Zeitfenster gilt als ausgeschöpft, wenn die Differenz zwischen Eintritts- und Austrittszeitpunkt eines Gerätes größer als 105 Sekunden ist. Diese Abfrage berechnet den Anteil ausgeschöpfter Zeitfenster des 21.08.2022 je Bluetooth-Logger. Das Ergebnis ist in Tabelle 8.3 dargestellt.

```
SET @time_from = '2022-08-21 00:00:00.0';
SET @time_to = '2022-08-21 23:59:59.9';

SELECT DISTINCT station
             , COUNT(mac_hash) AS all_tw
             , SUM(timewindow_used) AS used_tw
             , SUM(timewindow_used) / COUNT(mac_hash) AS ratio
FROM (SELECT station
           , mac_hash
           , IF(
               TIME_TO_SEC(TIMEDIFF(MAX(time_out), MIN(time_in))) > 105
               , 1, 0) AS timewindow_used
      FROM ble_meas
      WHERE time_window IN (SELECT time_from
                            FROM ble_timewindows
                            WHERE time_from >= @time_from
                              AND time_to <= @time_to)
      GROUP BY station, mac_hash, time_window) AS used_tw_per_station
GROUP BY station;
```

A.5. Modellversuch

Im Folgenden ist der Modellversuch auf Grundlage von Matrix P aus Unterabschnitt 8.6.3 skizziert. Dieser orientiert sich an dem Ansatz von Hong, De Silva und Chan [vgl. HDC18, S. 6.2].

Besuchende können nur am Ein-/Ausgang das Gelände betreten bzw. verlassen. Es werden zwei künstliche Zustände *in* und *out* eingeführt. Von *in* gelangen Besuchende ausschließlich zum Eingang bei Bluetooth-Logger *6*. Verlassen können sie das Testgelände nur von Bluetooth-Logger *8* zu Zustand *out*[2].

Ein Gewichtungsfaktor $(1-q)$ gibt an, welcher Anteil der Besuchenden das Gelände über *in* betreten, $(1-p)$ gibt an, welcher Anteil der Besuchenden das Gelände über *8* verlassen. Aus Matrix P ergibt sich die modifizierte Matrix P_M:

$$P_M = \begin{array}{c} \\ 1 \\ 2 \\ 5 \\ 6 \\ 8 \\ 9 \\ in \\ out \end{array} \begin{array}{c} \begin{array}{cccccccc} 1 & 2 & 5 & 6 & 8 & 9 & in & out \end{array} \\ \begin{pmatrix} 0,1392 & 0,0421 & 0,0599 & 0,2540 & 0,4498 & 0,0550 & 0 & 0 \\ 0,0122 & 0,2001 & 0,0697 & 0,0266 & 0,0458 & 0,6456 & 0 & 0 \\ 0,6226 & 0,0865 & 0,1322 & 0,0457 & 0,0889 & 0,0240 & 0 & 0 \\ 0,0098 & 0,0659 & 0,0042 & 0,2031 & 0,6464 & 0,0707 & 0 & 0 \\ 0,0209 \cdot p & 0,1340 \cdot p & 0,0080 \cdot p & 0,5682 \cdot p & 0,1059 \cdot p & 0,1629 \cdot p & 0 & 1-p \\ 0,0061 & 0,7933 & 0,0100 & 0,0223 & 0,0422 & 0,1262 & 0 & 0 \\ 0 & 0 & 0 & 1-q & 0 & 0 & q & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix} \end{array},$$

$$p, q \in \mathbb{R}, 0 \leq p, q \leq 1$$

Die Anzahl an Besuchenden je Bluetooth-Logger zu einem bestimmten Zeitpunkt t wird in einem Zustandsvektor s_t angegeben. Zum Startzeitpunkt $t = 0$ sind noch keine Besuchenden im Testgelände. Der Zustandsvektor kann formuliert werden als

$$s_0^T = \begin{array}{c} \begin{array}{cccccccc} 1 & 2 & 5 & 6 & 8 & 9 & in & out \end{array} \\ \begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 0 & x & 0 \end{pmatrix} \end{array}$$

wobei x für die Gesamtanzahl an Besuchenden des Tages steht. Die Übergangsmatrix gilt je Zeitfenster von zwei Minuten. Mit geeigneten Werten für p und q lassen sich Zustandsvektoren für spätere Zeitpunkte berechnen. Der erwartete Zustand nach einem Zeitfenster ist somit

$$s_1 = s_0 * P_M \ ,$$

der nach n Zeitfenstern

$$s_n = s_0 * P_M^n \ .$$

[2]Zwar sind der Ein- und Ausgang an der selben Position im Testgelände, doch deckt Bluetooth-Logger *6* die Seite des Eingangs und *8* die des Ausgangs besser ab.

Literatur

[AG20] Apple Inc. und Google LLC. *Exposure Notification - Bluetooth Specification*. Apr. 2020. URL: https://covid19-static.cdn-apple.com/applications/covid19/current/static/contact-tracing/pdf/ExposureNotification-BluetoothSpecificationv1.2.pdf (besucht am 06. 07. 2022).

[Bas16] Anas Basalamah. "Crowd Mobility Analysis using WiFi Sniffers". In: *International Journal of Advanced Computer Science and Applications* 7.12 (2016). DOI: 10.14569/IJACSA.2016.071249. URL: http://dx.doi.org/10.14569/IJACSA.2016.071249.

[BDSG18] *Bundesdatenschutzgesetz (BDSG)*. 2018. URL: https://www.gesetze-im-internet.de/bdsg_2018/ (besucht am 06. 11. 2022).

[BLS19] Johannes Becker, David Li und David Starobinski. "Tracking Anonymized Bluetooth Devices". In: *Proceedings on Privacy Enhancing Technologies* 2019 (Juli 2019), S. 50–65. DOI: 10.2478/popets-2019-0036.

[Bon+13] Bram Bonné u. a. "WiFiPi: Involuntary tracking of visitors at mass events". In: *2013 IEEE 14th International Symposium on „A World of Wireless, Mobile and Multimedia Networks" (WoWMoM)*. 2013, S. 1–6. DOI: 10.1109/WoWMoM.2013.6583443.

[DSGVO16] *Verordnung (EU) 2016/679 des Europäischen Parlaments und des Rates vom 27. April 2016 zum Schutz natürlicher Personen bei der Verarbeitung personenbezogener Daten, zum freien Datenverkehr und zur Aufhebung der Richtlinie 95/46/EG (Datenschutz-Grundverordnung)*. 2016. URL: https://www.bmj.de/DE/Themen/FokusThemen/DSGVO/_documents/Amtsblatt_EU_DSGVO.pdf (besucht am 06. 11. 2022).

[GŞB21] Cavide Balkı Gemirter, Çağatay Şenturca und Şebnem Baydere. "A Comparative Evaluation of AMQP, MQTT and HTTP Protocols Using Real-Time Public Smart City Data". In: *2021 6th International Conference on Computer Science and Engineering (UBMK)*. 2021, S. 542–547. DOI: 10.1109/UBMK52708.2021.9559032.

[GZY16] Yongan Guo, Hongbo Zhu und Longxiang Yang. "Service-oriented network virtualization architecture for Internet of Things". In: *China Communications* 13.9 (2016), S. 163–172. DOI: 10.1109/CC.2016.7582308.

[HDC18] Hande Hong, Girisha Durrel De Silva und Mun Choon Chan. "CrowdProbe: Non-Invasive Crowd Monitoring with Wi-Fi Probe". In: *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.* 2.3 (Sep. 2018). DOI: 10.1145/3264925. URL: https://doi.org/10.1145/3264925.

[Hu+12] Yuting Hu u. a. "Prediction of tourists flow distribution based on transition probability matrix". In: *2012 8th International Conference on Information Science and Digital Content Technology (ICIDT2012)*. Bd. 3. 2012, S. 636–640.

[Kha+12] Rafiullah Khan u. a. "Future Internet: The Internet of Things Architecture, Possible Applications and Key Challenges". In: *2012 10th International Conference on Frontiers of Information Technology*. 2012, S. 257–260. DOI: 10.1109/FIT.2012.53.

[Ope22] Open-Source-Projekt Corona-Warn-App. *Wie viele aktive Nutzende hat die Corona-Warn-App?* 2022. URL: https://www.coronawarn.app/de/science/2022-03-03-science-blog-5/ (besucht am 12. 11. 2022).

[Sop+21] S. Sophia u. a. "Bluetooth Low Energy based Indoor Positioning System using ESP32". In: *2021 Third International Conference on Inventive Research in Computing Applications (ICIRCA)*. 2021, S. 1698–1702. DOI: 10.1109/ICIRCA51532.2021.9544975.

[SP20] Petros Spachos und Konstantinos N. Plataniotis. "BLE Beacons for Indoor Positioning at an Interactive IoT-Based Smart Museum". In: *IEEE Systems Journal* 14.3 (2020), S. 3483–3493. DOI: 10.1109/JSYST.2020.2969088.

[STJ15] Dhananjay Singh, Gaurav Tripathi und Antonio Jara. "Secure layers based architecture for Internet of Things". In: *2015 IEEE 2nd World Forum on Internet of Things (WF-IoT)*. 2015, S. 321–326. DOI: 10.1109/WF-IoT.2015.7389074.

[Ver+12a] M. Versichele u. a. "Intelligent Event Management with Bluetooth Sensor Networks". In: *2012 Eighth International Conference on Intelligent Environments*. 2012, S. 311–314. DOI: 10.1109/IE.2012.25.

[Ver+12b] Mathias Versichele u. a. "The Use of Bluetooth for Analysing Spatiotemporal Dynamics of Human Movement at Mass Events: A Case Study of the Ghent Festivities". In: *Applied Geography* 32 (März 2012), S. 208–220. DOI: 10.1016/j.apgeog.2011.05.011.

[Ws22] Klaus Wilting (cyberman54) und Brandmueller (spmrider). *ESP32-Paxcounter*. Version 3.4.0. Nov. 2022. URL: https://github.com/cyberman54/ESP32-Paxcounter.

[Wu+10] Miao Wu u. a. "Research on the architecture of Internet of Things". In: *2010 3rd International Conference on Advanced Computer Theory and Engineering(ICACTE)*. Bd. 5. 2010, S. V5-484-V5-487. DOI: 10.1109/ICACTE.2010.5579493.