WearPrivate

Datenschutzfreundliche Vermessung und Auswertung persönlicher Daten mit Wearables zur Erhöhung der Arbeitssicherheit

Ergebnisbericht D3.3

Konzeption des Hauptdemonstrators

Version	1.0
Datum	22.11.2024
Verfasser	Marcus-Sebastian Schröder (NMS) Philipp Neuschwander (IESE)

Das diesem Bericht zugrundeliegende Vorhaben wurde mit Mitteln des Bundesministeriums für Bildung und Forschung unter den Förderkennzeichen 16KIS1511K, 16KIS1512, 16KIS1514 und 16KIS1665 gefördert.

Die Verantwortung für den Inhalt dieser Veröffentlichung liegt bei den Autoren.

GEFÖRDERT VOM

Bundesministerium für Bildung und Forschung

Ansprechperson

Marcus-Sebastian Schröder
neusta mobile solutions GmbH
Konsul-Smidt-Str. 24
28217 Bremen

E-Mail: m.schroeder@neusta.de

Inhaltsverzeichnis

Liste der Abkürzungen

DB	Datenbank
DP	Differential Privacy
iOS	iPhone Operating System
IT	Informationstechnik
PAP	Policy Administration Point
PDP	Policy Decision Point
PEP	Policy Enforcement Point
PIP	Policy Information Point
PMP	Policy Management Point
PRP	Policy Registration Point
REST	Representational State Transfer
SDK	Software Development Kit

1 Einleitung

Der im Rahmen des Projekts WearPrivate erstellte Hauptdemonstrator dient zur technischen Umsetzung repräsentativer Anwendungsszenarien. Dieses Dokument beschreibt das Demonstratorkonzept und die implementierten Funktionalitäten und Schnittstellen.

2 Datennutzungskontrolle im Demonstrator

Im Rahmen von WearPrivate wird die Lösung MYDATA zur technischen Datennutzungskontrolle im Gesamtsystem verwendet. Entsprechend soll MYDATA auch im Hauptdemonstrator integriert werden. Die Ergebnisberichte D3.1 [1] und D3.2 [2] beschreiben die grundlegenden Prinzipien der Datennutzungskontrolle, die Lösung MYDATA und die Integration von MYDATA in das Gesamtsystem bereits konzeptionell. Nachfolgend wird die konkrete Integration von MYDATA im Hauptdemonstrator näher erläutert sowie die Vereinfachungen, die wir in Abweichung zu [1] vorgenommen haben.

Abbildung 1 zeigt die konzeptionelle Integration und Positionierung der MYDATA Komponenten im Gesamtsystem aus [1].

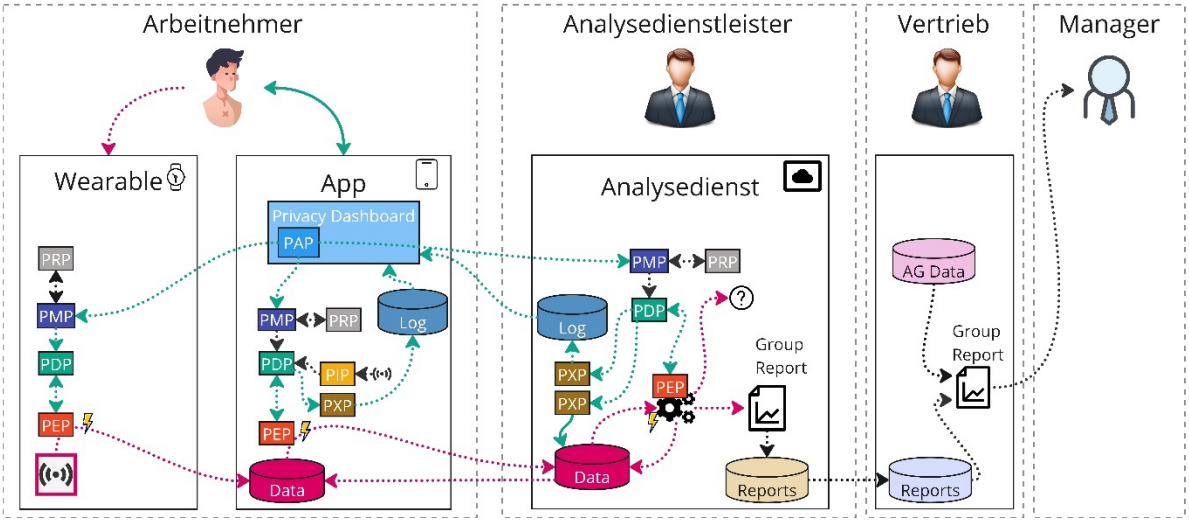

Abbildung 1 Konzeptionelle Integration und Positionierung der MYDATA-Komponenten im Gesamtsystem (aus [1])

Dem gegenüber stellt Abbildung 2 die konkrete Integration und Positionierung der MYDATA Komponenten im Demonstrator dar. Gegenüber der idealtypischen Konzeption wurden einige technische Vereinfachungen vorgenommen, um den Realisierungsaufwand dort in Grenzen zu halten, wo dies keinen nennenswerten Effekt auf die Demonstrierbarkeit der Konzepte hat oder wo unter den gegebenen Umständen eine vollständige Realisierung nicht möglich war.

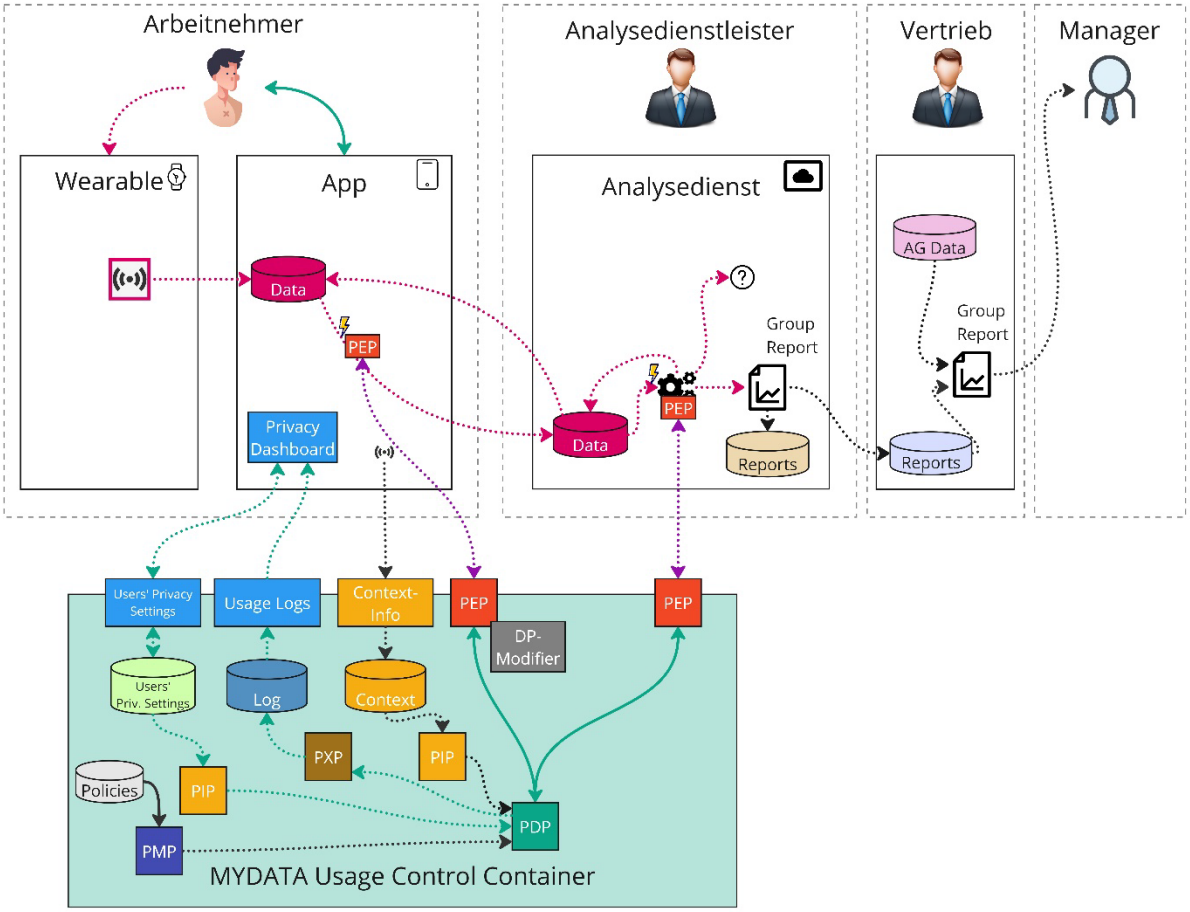

Abbildung 2 Integration und Positionierung der MYDATA-Komponenten im Demonstrator

Im Folgenden werden die wesentlichen Abweichungen vom ursprünglichen Konzept näher beschrieben.

2.1 Keine Datennutzungskontrolle im Wearable

Da wir nach dem Ausscheiden des ursprünglichen Projektpartners Ambiotex, der eigene Smart-Shirts als Wearable-Hardware einsetzen wollte, auf die Firmware der Drittanbieter-Wearables keinen Zugriff haben, können wir dort keine Kontrollmechanismen integrieren. Aus diesem Grund werden wir im Demonstrator lediglich in der Smartphone-App des Arbeitnehmers und im System des Analysedienstes in der Cloud Datennutzungskontrolle integrieren.

2.2 Verwendung eines externen MYDATA-Containers statt lokaler Komponenten

MYDATA bietet mit seinem Open-Source-SDK eine Implementierung für die Programmiersprache Java. Sollten im Rahmen von WearPrivate andere Programmiersprachen für die Komponenten der Systemkette verwendet werden, sind verschiedene Optionen denkbar, wie MYDATA dennoch integriert und verwendet werden kann [2]:

1. Entwicklung von Policy Enforcement Points (PEPs) in der jeweiligen Programmiersprache und Verwendung des bestehenden PDP über programmiersprachenunabhängige Schnittstellen

2. Externalisierung des Java-PEP und dessen Betrieb als Dienst oder »Sidecar« neben der betreffenden Anwendungskomponente; Integration eines leichtgewichtigen PEP-Proxies in der jeweiligen Programmiersprache, der den Java-PEP über eine programmiersprachen-unabhängige Schnittstelle einbindet und Teile der Durchsetzung an diesen delegiert.

Die App auf dem Smartphone des Arbeitnehmers wird in der Programmiersprache Swift für die iOS-Plattform implementiert. Hier ist eine direkte Integration von Java-basierten MYDATA-Komponenten nicht ohne Weiteres möglich. Das System des Analysedienstleisters in der Cloud verwendet ebenfalls bereits andere Technologien und Programmiersprachen, so dass eine direkte Integration der Java-basierten MYDATA-Komponenten ebenfalls nicht unmittelbar möglich ist. Entsprechend bleiben uns die beiden vorgenannten Optionen zur Integration von MYDATA. Wir haben uns im Projektkonsortium für Option 2 entschieden, um Entwicklungsaufwände der Anwendungspartner einzusparen, so dass diese sich auf die App und den Analysedienst konzentrieren können. Die Umsetzung und Bereitstellung des MYDATA Usage-Control-Containers übernimmt das Fraunhofer IESE in enger Abstimmung mit den Konsortialpartnern.

Der MYDATA Usage-Control-Container stellt die von App und Analysedienst benötigten PEP-Funktionalitäten bereit und bietet diese über eine Web-Anwendungsschnittstelle an. Sowohl App als auch Analysedienst müssen in ihrem Code lediglich leichtgewichtige PEP-Proxies integrieren, welche die Durchsetzung der Datennutzungskontrolle an die vom MYDATA-Container bereitgestellten PEPs delegieren. Dadurch drehen die Daten eine Schleife über den externen MYDATA-Container; die Datennutzungs-Kontrolllogik bleibt jedoch die gleiche wie im idealtypischen Konzept vorgesehen.

In einer produktreifen Umsetzung des Systems wären die MYDATA Komponenten jeweils lokal in der Smartphone-App beziehungsweise im Cloud-System des Analysedienstes verortet.

2.3 Statische Policies und User Privacy Settings statt Policy Administration Point

Das Projektkonsortium hat sich dafür entschieden, dem Arbeitnehmer eine festgelegte Menge von Einstellungsmöglichkeiten anzubieten, mit denen er seine Präferenzen festlegen kann. Aus diesem Grund stellen wir dem Nutzer keinen Policy Administration Point (PAP) zur Spezifikation von individuellen Richtlinien im klassischen Sinn bereit, sondern verwenden eine kleine Menge allgemeingültige Datennutzungsregeln mit individueller Konfiguration (vgl. [2]). Entsprechend sind die Richtlinien im Demonstrator statisch hinterlegt, berücksichtigen jedoch die vom jeweiligen Nutzer gewählten Einstellungen. Die nutzerindividuellen Einstellungen werden in der Richtline durch Anfragen an einen Policy Information Point (PIP) abgerufen und anschließend bei der Richtlinienauswertung berücksichtigt.

Die Entscheidung für statische Policies anstelle von frei gestaltbaren Policies ist im Kontext der Demonstratoranwendung naheliegend, weil Anwender in der Regel IT-Sicherheits- und Datenschutz-laien sind und oft genug auch wenig IT-Affinität haben. Daher kamen wir zu der Einschätzung, dass die Mehrheit der potenziellen Nutzer weder die Muße noch die Kompetenz hat, sich komplexe Datennutzungsrichtlinien selbst auszudenken und in einem Policy-Editor selbst zu erstellen.

Beispiele für parametrisierbare statische Richtlinien, die im Demonstrator verwendet werden, sind in Abschnitt 2.6 zu finden.

2.4 Kein Policy Execution Point zum Löschen der Daten im Analysedienst

Das richtlinienbasierte Löschen von Daten im Analysedienst war nicht Teil der im Projektkonsortium vereinbarten Optionen, die dem Arbeitnehmer im Demonstrator bereitgestellt werden sollen.

Grundsätzlich wäre es durchaus möglich, einen entsprechenden Policy Execution Point (PXP) oder einen PXP-Proxy mit Anbindung an einen PXP im MYDATA Usage-Control-Container bereitzustellen. Das eigentliche Problem ist nicht die technische Umsetzung des PXP, sondern der Umgang mit Daten, die zuvor vielleicht bereits als Trainingsdaten Eingang in den datengetriebenen, KI-basierten Analysedienst gefunden haben. Hier ist es konzeptionell sehr schwierig, die Trennlinie zu definieren, bis zu der die Betroffenen ihr Recht auf Löschung einfordern können, wenn sie zuvor ihr Einverständnis erklärt haben, ihre Daten für Trainingszwecke freizugeben. Der spannende Aspekt ist daher eher die juristische als die IT-technische Herausforderung, die mit dem Löschen einhergeht.

2.5 Verfügbarkeit von Kontextinformationen

Sollen Kontextinformationen (z. B. die aktuelle Geolokation des Smartphones) bei der Auswertung von Datennutzungsrichtlinien berücksichtigt werden, so müssen diese Kontextinformationen regelmäßig dem externen MYDATA Usage-Control-Container mittels Push-Mechanismus bereitgestellt werden, damit diese per PIP dem PDP verfügbar gemacht werden können.

In der Praxis ist die Weiterleitung der Daten an einen externen MYDATA-Container nachteilig in Bezug auf den Energieverbrauch der Anwendung und damit die Entladung der Smartphone-Batterie. In einer kommerziell genutzten Lösung sollte dies daher vermieden werden. Für Demonstrationszwecke spielt die verkürzte Batterielaufzeit jedoch eine untergeordnete Rolle.

Die Weiterleitung an einen externen MYDATA-Container ist auch aus Datenschutz-Gründen nicht erstrebenswert. Da Kontextdaten für eine Leistungs- und Verhaltenskontrolle missbraucht werden könnten, sollten sie das Smartphone des Nutzers am besten nicht verlassen, sondern lokal ausgewertet werden, wie es unser Datenschutzkonzept auch vorsieht. Im Demonstrator sehen wir von dieser Datenschutzschwäche jedoch ab und nutzen die Container-Lösung zur Vereinfachung der Implementierung des Demonstrators.

2.6 Beispiele für Richtlinien im Demonstrator

Im Folgenden stellen wir beispielhafte Datennutzungsrichtlinien vor, die für den Demonstrator konzipiert wurden. Aufgrund der Flexibilität des MYDATA-Frameworks ist es relativ einfach, vielseitige Datennutzungsrichtlinien mit der von MYDATA bereitgestellten Policy-Sprache zu realisieren, die von den MYDATA-Komponenten (PEP, PDP, PXP ...) dann durchgesetzt werden. Es bleibt der Kreativität der Dienstanbieter überlassen, für ihren jeweiligen Anwendungsfall sinnvolle Richtlinien zu ersinnen.

Richtlinie 1 Daten vor der Übertragung an den Analysedienst mittels Differential Privacy schützen

```
<policy id='urn:policy:wearprivate:app-policy'
        xmlns='http://www.mydata-control.de/4.0/mydataLanguage'
        xmlns:parameter="http://www.mydata-control.de/4.0/parameter"
        xmlns:constant="http://www.mydata-control.de/4.0/constant"
        xmlns:event="http://www.mydata-control.de/4.0/event"
        xmlns:pip="http://www.mydata-control.de/4.0/pip"
        xmlns:variable="http://www.mydata-control.de/4.0/variable"
        xmlns:variableDeclaration="http://www.mydata-control.de/4.0/variableDeclaration"
>
  <variableDeclaration:string name="dataTransferToAnalyticsSvcAnonymMode">
    <pip:string method="urn:info:wearprivate:readUserSetting" default="HIGH">
      <parameter:string name="userId">
        <event:string eventParameter="userId" default=""/>
      </parameter:string>
      <parameter:string name="setting"
                        value="$.appSettings.dataTransferToAnalyticsSvcAnonymMode"/>
    </pip:string>
  </variableDeclaration:string>
  <mechanism event='urn:action:wearprivate:app-sends-data-to-analytics-service'>
    <if>
      <not>
        <equals>
          <variable:string reference="dataTransferToAnalyticsSvcAnonymMode"/>
          <constant:string value="OFF"/>
        </equals>
      </not>
      <then>
        <modify eventParameter="data" method="anonymizeDataPackage" jsonPathQuery="$">
          <parameter:string name="anonymizationMode">
            <variable:string reference="dataTransferToAnalyticsSvcAnonymMode"/>
          </parameter:string>
        </modify>
      </then>
    </if>
  </mechanism>
</policy>
```

Richtlinie 2 Datennutzung für Gruppenbericht einschränken und protokollieren

```
<policy id='urn:policy:wearprivate:cloud-policy'
        xmlns='http://www.mydata-control.de/4.0/mydataLanguage'
        xmlns:parameter="http://www.mydata-control.de/4.0/parameter"
        xmlns:event="http://www.mydata-control.de/4.0/event"
        xmlns:pip="http://www.mydata-control.de/4.0/pip"
>
  <mechanism event='urn:action:wearprivate:analytics-svc-uses-data-to-generate-grp-report'>
    <if>
      <pip:boolean method="urn:info:wearprivate:readUserSetting" default="false">
        <parameter:string name="userId">
          <event:string eventParameter="userId" default=""/>
        </parameter:string>
        <parameter:string name="setting"
value="$.analyticsServiceSettings.contributeDataToGroupReport"/>
      </pip:boolean>
      <then>
        <allow/>
        <execute action='urn:action:wearprivate:logEvent'>
          <parameter:string name='userId'>
            <event:string eventParameter="userId" default=""/>
          </parameter:string>
          <parameter:string name='eventType' value="analytics-svc-uses-data-to-generate-grp-
report"/>
          <parameter:string name='entryTitle' value="Teilnahme Gruppenbericht"/>
          <parameter:string name='entryText' value="Der Analysedienst hat Daten für den
Gruppenbericht verwendet."/>
        </execute>
      </then>
    </if>
    <else>
      <inhibit/>
    </else>
  </mechanism>
</policy>
```

3 Schutz der Messdaten mittels Differential Privacy

Die Messdaten des Arbeitnehmers werden entsprechend seiner individuellen Einstellungen vor einer Übertragung an den Analysedienst mittels Differential Privacy verfremdet und damit zusätzlich geschützt [1][2]. Vereinfacht ausgedrückt wird bei Differential Privacy ein Rauschen auf die Daten gelegt, so dass sich die Datenwerte (mal mehr, mal weniger) verändern. Im Idealfall mittelt sich das Rauschen bei der Analyse größerer Datenbestände heraus, so dass grundsätzliche Aussagen über die Daten möglich sind, auch wenn die genutzten Daten im Detail von den Originaldaten abweichen.

Im Demonstrator bieten wir dem Arbeitnehmer daher bezüglich des Schutzes seiner Messdaten mittels Differential Privacy drei Optionen zur Auswahl an:

- OFF ⇨ keine Verfremdung der Messdaten (beste Analysequalität, kein zusätzlicher Schutz)
- LOW ⇨ leichte Verfremdung der Messdaten (mittlere Analysequalität, mittlerer Schutz)
- HIGH ⇨ starke Verfremdung der Messdaten (guter Privacy-Schutz, geringe Analysequalität)

Im Demonstrator werden die Messdaten durch den (in der App integrierten) Policy Enforcement Point (PEP) geschützt, bevor sie an den Analysedienst übermittelt werden. On-the-fly-Datenmodifikationen sind als sogenannter Modifier realisiert, die im Zuge der Richtlinienauswertung bedarfsgerecht – entsprechend den konfigurierten Richtlinien und individuellen Einstellungen des Arbeitnehmers – der Autorisierungsentscheidung hinzugefügt und anschließend vom PEP angewendet werden.

Speziell für die Option »Datenverfremdung« wurde im Rahmen des Projekts ein neuer Modifier entwickelt. Sofern eine Verfremdung nicht mit der Schalterstellung OFF deaktiviert ist, wird dieser

Modifier zur Laufzeit mit der vom Arbeitnehmer individuell gewählten Verfremdungsstufe (*LOW* oder *HIGH*) parametrisiert und verändert die Daten dann entsprechend. Mit der jeweiligen Verfremdungsstufe sind entsprechende Epsilon-Werte verknüpft, die den Grad der Datenverfremdung und damit die Stärke des Schutzes steuern. Angemessene Epsilon-Werte werden im Projektverlauf durch weiterführende Untersuchungen ermittelt.

Zum Hinzufügen des Rauschens auf die Daten verwenden wir eine an eine IBM-Lösung[1] angelehnte Implementierung, die einen Bounded-Laplace-Mechanismus für Differential Privacy [3] umsetzt. Mit dieser können numerische Werte verfremdet werden, indem Differential-Privacy-Techniken auf die Daten angewandt werden.

Das Verrauschen der Beschleunigungsdaten gestaltet sich relativ einfach, da das Rauschen direkt auf die Werte angewendet werden kann.

Das Verrauschen der Herzratendaten gestaltet sich jedoch schwieriger, da hier die Herzschläge und die exakten Zeiten zwischen diesen (sog. RR-Intervalle) aufgezeichnet und im Sekundentakt einschließlich Zeitstempel protokolliert werden (siehe Abbildung 3).

```
timestamp;HR;RR
2023-01-05T14:31:16.705+01:00;75;(894,0,0,0)
2023-01-05T14:31:17.685+01:00;74;(889,0,0,0)
2023-01-05T14:31:18.705+01:00;73;(864,874,0,0)
2023-01-05T14:31:19.681+01:00;73;(895,0,0,0)
2023-01-05T14:31:20.701+01:00;73;(911,0,0,0)
2023-01-05T14:31:21.676+01:00;72;(911,0,0,0)
2023-01-05T14:31:22.704+01:00;72;(897,0,0,0)
```

Abbildung 3 Beispiel für Wearable-Herzdaten im CSV-Format

Ein naives Verrauschen der RR-Intervalle unter Beibehaltung ihrer Position auf dem sich durch die Datenzeilen und Zeitstempel ergebenden Zeitstrahl könnte zu Inkonsistenzen führen und Rückschlüsse auf die Originaldaten zulassen. Hier ist ein komplexeres Vorgehen notwendig, um Differential Privacy korrekt umzusetzen.

Die Herzdaten werden vom Analysedienst im CSV-Format gemäß Abbildung 3 erwartet. In der ersten Spalte befindet sich der Zeitstempel, in der zweiten Spalte der Puls und in der dritten Spalte bis zu vier aufeinander folgende RR-Intervalle (in Millisekunden). Der RR-Intervall bezeichnet die Zeit, die zwischen zwei aufeinanderfolgenden Herzschlägen lag. Diese Daten werden im Sekundentakt vom Wearable an die App geliefert und dort zwischengespeichert. Alle 30 Sekunden stellt die App ein Datenpaket mit den Daten der letzten 30 Sekunden zusammen und überträgt dieses an den Analysedienst.

[1] Siehe hierzu die Differential-Privacy-Library von IBM (2019): https://github.com/IBM/differential-privacy-library/blob/9dde5b916ef70fc4854bad3ce95d2a4851c0c417/diffprivlib/mechanisms/laplace.py#L282

An dieser Stelle wurde ein PEP integriert, der die Aktion *Datenübertragung an Analysedienst* kontrolliert. Der Modifier wird bei Bedarf auf das 30-Sekunden-Datenpaket angewendet und modifiziert dieses.

Nachfolgend wird der umgesetzte Algorithmus zum Schutz der Herzdaten mittels Differential Privacy beschrieben:

1. Extrahiere alle RR-Intervalle aus dem Datenpaket in eine Liste. Dabei werden alle 0-Werte (Füllwerte im Datenformat) ausgelassen.
2. Verrausche jeden Wert in der Liste einzeln.
3. Falls durch das Rauschen die Summe der neuen RR-Intervalle kleiner ist, als bei den Originaldaten und sich eine unglaubwürdig große Lücke in den Daten ergäbe, füge plausible, ähnliche RR-Intervalle ein, um diese Lücke zu verbergen. Sollten die RR-Intervalle durch das Rauschen im Schnitt kürzer geworden sein, dann braucht es im 30 Sekunden Datenpaket gegebenenfalls mehr Herzschläge, um diese neue, verrauschte Realität glaubhaft darzustellen.
4. Erstelle die Datenzeilen unter Verwendung der ursprünglichen Zeitstempel und den neuen RR-Intervallen neu. Hierbei ist zu beachten, dass sich die durch die RR-Intervalle beschriebenen Herzschläge auf dem Zeitstrahl im Vergleich zu den Originaldaten bewegt haben können (da sich die Zeiten zwischen den Herzschlägen durch das Rauschen verändert haben) und entsprechend der neuen Datenlage auf die Zeilen zu verteilen sind. Möglicherweise wurden Herzschläge und damit die zugehörigen RR-Intervalle in der neuen, verrauschten Realität etwas früher oder später vom Sensor aufgezeichnet und entsprechend in einer anderen Sekunde / zu einem anderen Zeitstempel protokolliert, als dies bei den Originaldaten der Fall war. Falls Herzschläge / RR-Intervalle übrigbleiben, die nicht mehr in das neue 30 Sekunden Datenpaket passen, werden diese verworfen. Sollten die RR-Intervalle durch das Rauschen im Schnitt länger geworden sein, dann braucht es im 30 Sekunden Datenpaket möglicherweise weniger Herzschläge, um diese neue, verrauschte Realität glaubhaft darzustellen.
5. Für jede Datenzeile wird nun die Pulsangabe auf der Grundlage der neuen Daten berechnet. Hierzu ermitteln wir den durchschnittlichen RR-Intervall als gleitenden Mittelwert über ein Auswertungsfenster, das neben der aktuellen Datenzeile auch bis zu drei vorhergehende Datenzeilen (sofern vorhanden) berücksichtigt. Anschließend werden 60 000 Millisekunden durch den zuvor bestimmten durchschnittlichen RR-Intervall geteilt, um die Pulsangabe für die jeweilige Datenzeile zu bestimmen. Dadurch bleiben die Daten (Puls und RR-Intervalle) in sich konsistent und spiegeln die sich durch das Rauschen ergebende, neue Realität glaubhaft wider.

Die mit dieser Implementierung durchgeführten Untersuchungen und die dabei gewonnenen Erkenntnisse sind im Ergebnisbericht D6.1 [4] näher beschrieben.

4 Abläufe von kritischen Vorgängen im Gesamtsystem

Auf Grundlage der im Ergebnisbericht D1.1 [5] erhobenen Anforderungen und der in D3.1 [1] entwickelten Maßnahmen haben wir die beiden Vorgänge »Registrierung eines neuen Benutzers« und »Synchronisierung von Gesundheitsdaten« als besonders sensibel identifiziert; gleichzeitig sind diese aber auch sehr zentral für die Funktionalität des WearPrivate-Demonstrators.

Aus diesem Grund haben wir diese Vorgänge mit besonderer Sorgfalt konzipiert und dokumentieren deren Funktionsweise im Folgenden durch Sequenzdiagramme. Ausgehend von den in D1.1 definierten Rollen betrachten wir im Folgenden die nachstehenden technischen Komponenten:

- *Identity Provider:* eine vom Analysedienst betriebene Komponente, die bei der Registrierung die Eintrittskarte und ein Passwort in einen Nutzernamen umtauscht. Weiterhin tauscht sie eine gültige Kombination aus Nutzernamen und Passwort in ein Access Token um, das mit dem Datendienst verwendet werden kann.
- *Datendienst:* eine vom Analysedienst betriebene Komponente, die Gesundheitsdaten mit einem gültigen Access Token zur Verarbeitung entgegennimmt.
- *App:* eine nicht-personalisierte Smartphone-Anwendung, die gemäß der WearPrivate-Spezifikation mit den Komponenten des Analysedienstes kommuniziert. Sie kann von einem oder mehreren beliebigen Anbietern bereitgestellt werden. Die App bezieht die weiterzuleitenden Vital- und gegebenenfalls Kontextdaten von einem Wearable.
- *Wearable:* ein von einem Dritthersteller bezogenes technisches Gerät zur Erhebung von Vitaldaten.
- *Enforcement Point:* eine Komponente, welche die Wahrung der vom Nutzer definierten Policies bei der Verarbeitung von Daten sicherstellt. Im Demonstrator über REST-Endpunkte realisiert; sollte bei Produktivimplementierungen direkt in App und Cloud-Dienst integriert sein.

In unserem Demonstrator wird sowohl der *Identity Provider* als auch der *Datendienst* durch den Konsortialpartner WearHealth betrieben. Bei Produktivsystemen sollte unbedingt auf die Einhaltung dieser Trennung geachtet werden.

4.1 Registrierung eines neuen Benutzers

Um eine Zuordnung eines Benutzers im System zu einer tatsächlichen Person zu vermeiden, beginnt der Registrierungsprozess mit der Eingabe eines Registrierungscodes in die App. Dieser Code wird von der Rolle Marketing & Vertrieb produziert und zum Beispiel in Textform oder als QR-Code bereitgestellt. Alle Teilnehmer einer Gruppe sollten ihren Code anonym zugeteilt bekommen, zum Beispiel durch zufälliges Ziehen von Zetteln aus einer Urne.

Nach dem Einlesen des nur einmal verwendbaren Registrierungscodes fordert die *App* den Nutzer auf, ein Passwort einzugeben, das im Folgenden für das Nutzerkonto verwendet werden soll. Mit den beiden Informationen Registrierungscode und Passwort kann nun der Aufruf für die Registrierung eines neuen Nutzers am *Identity Provider* durchgeführt werden. Der *Identity Provider* prüft die Gültigkeit des Registrierungscodes und stellt damit sicher, dass sich nur autorisierte Nutzer registrieren können. Als Antwort auf eine gültige Registrierung erhält der Nutzer eine User-ID, die das neu angelegte Nutzerkonto repräsentiert. Die *App* stellt diese für den Nutzer dar und weist ihn wie schon zuvor bei der Passworteingabe darauf hin, dass die Zugangsdaten sicher vermerkt werden müssen. Dies ist nötig, da ohne ein identifizierendes Merkmal wie eine E-Mail–Adresse keine Authentisierung zur Wiederherstellung des Kontozugangs möglich ist, wenn die Anmeldedaten (d.h. User-ID und Passwort) vergessen wurden.

Nachdem der Registrierungsprozess abgeschlossen ist, springt die *App* zurück zum Anmeldebildschirm. Dort kann sich der Nutzer mit den nun vorhandenen Nutzerdaten in der App anmelden. Der

entsprechende Aufruf beim *Identity Provider* gibt der *App* die Access- und Refresh-Tokens zurück, mit denen die weiteren, für den Betrieb nötigen Aufrufe an den Backends getätigt werden können.

Nach der erfolgreichen Anmeldung lädt die *App* das Nutzerprofil vom *Identity Provider*, um die entsprechenden Werte (Größe, Gewicht usw.) im Einstellungsbereich korrekt anzeigen zu können. Bei einem neu angelegten Benutzer wird der *Identity Provider* auf diese Anfrage allerdings ein leeres Objekt zurückgeben (NULL), da noch keine Werte vorhanden sind. Dies löst innerhalb der App den Onboarding-Prozess für einen neuen Benutzer aus.

Der Onboarding-Prozess zeigt zunächst einige grundlegende Informationen über das System an und fragt dann auf separaten Bildschirmen zunächst das Geburtsjahr und danach das Geschlecht, die Größe und das Gewicht ab. Jedes Mal, wenn alle Informationen auf einem Bildschirm eingetragen wurden und zum nächsten gesprungen wird, werden die bisher gesammelten Profilinformationen an den *Identity Provider* übertragen. Dieser antwortet darauf mit dem aktuellen Stand des bei ihm hinterlegten Profilobjektes.

Nachdem alle Daten erfasst wurden, zeigt der Onboarding-Prozess noch einen Hinweis zur Verbindung des *Wearables* über Bluetooth an und bietet einen Sprung in die Systemeinstellungen an, um dieses ggf. zu aktivieren. Hiernach endet der Onboarding-Prozess und der Dashboard-Bildschirm der *App* wird angezeigt.

Abbildung 4 zeigt ein Sequenzdiagramm, das den gesamten Ablauf von der Nutzerregistrierung bis zum abgeschlossenen Onboarding visualisiert.

Aus Zeitgründen nicht mehr Teil des Onboardings in der aktuellen Version des Demonstrators ist die geführte Ersteinrichtung der Privacy Settings durch den Nutzer. Es bietet sich für zukünftige Weiterentwicklungen oder Neuimplementierungen an, auch dies zum Teil des Onboardings zu machen.

4.2 Übermittlung von Gesundheitsdaten

Für den korrekten Betrieb des Systems ist es notwendig, die vom *Wearable* erhobenen Gesundheitsdaten regelmäßig an den *Datendienst* zu übermitteln. Der im Folgenden beschriebene Vorgang wiederholt sich daher nach dem Start der Datenerfassung regelmäßig.

Nachdem für einen bestimmten Zeitraum (im Demonstrator: 30 Sekunden) Vitaldaten vom *Wearable* in der *App* gesammelt wurden, werden diese mit einem *enforce*-Request an den *Enforcement Point* übergeben. Je nach den dort für den Nutzer hinterlegten Einstellungen werden diese Vitaldaten mehr oder weniger (oder gar nicht) verfremdet und an die *App* zurückgegeben. Falls die Nutzer-Policy die Übertragung von Daten komplett untersagt, wird nur eine entsprechende Exception zurückgegeben.

Die Antwort enthält die angepassten Vitaldaten, die nun an den *Datendienst* übertragen werden können. Dort werden die gesendeten Vitaldaten verarbeitet.

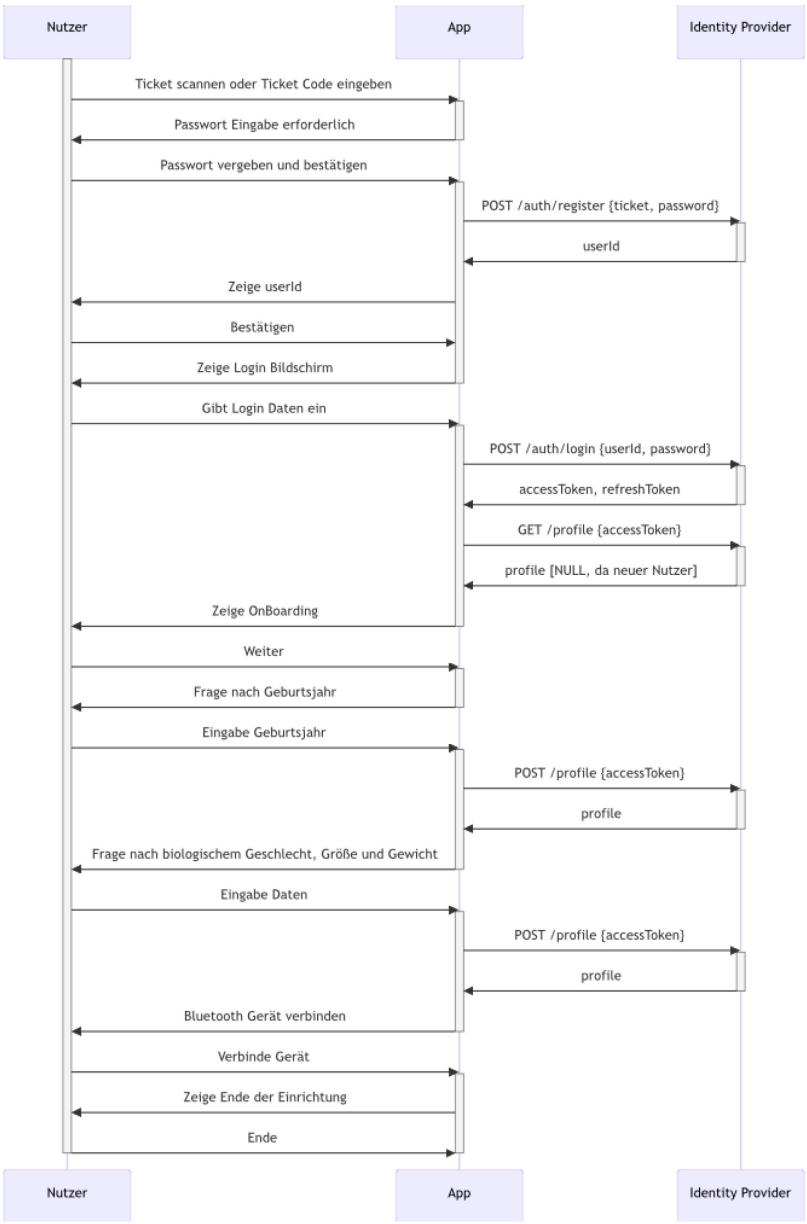

Abbildung 4 Registrierung eines Nutzers als Sequenzdiagramm

In der Antwort auf diese Datenübertragung wird ein SyncSuccess-Objekt zurückgegeben, in dem die aktuell ermittelten Analysewerte für mentale und körperliche Belastung enthalten sind. Weiterhin können optional auch Notification-Objekte enthalten sein, falls der *Datendienst* im Rahmen seiner Auswertung der gerade abgelieferten Daten eine Handlungsempfehlung für den Nutzer ermittelt hat, die von der *App* angezeigt werden soll.

Abbildung 5 zeigt ein Sequenzdiagramm, das den Ablauf der Übermittlung der Gesundheitsdaten visualisiert.

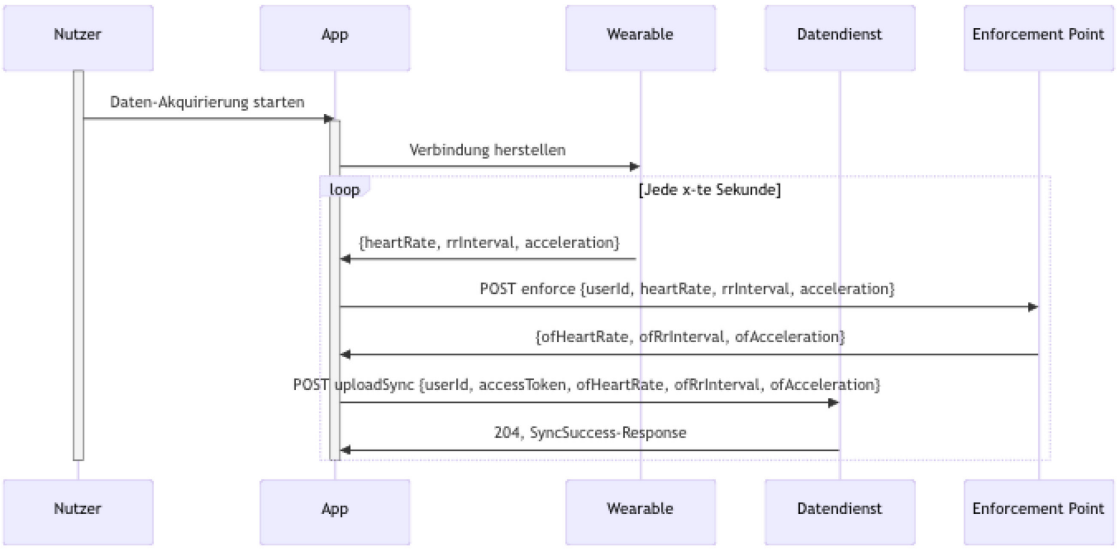

Abbildung 5 Übermittlung von Vitaldaten als Sequenzdiagramm

Quellenverzeichnis

[1] M.-S. Schröder, R. Schwarz, P. Neuschwander (2024): IT-Sicherheitsarchitektur und Datenschutzkonzept. Ergebnisbericht D3.1, WearPrivate-Projekt

[2] M.-S. Schröder, B. Steffes, P. Neuschwander (2023): Konzepte für Anonymisierung und Datennutzungskontrolle. Ergebnisbericht D3.2, WearPrivate-Projekt

[3] Naoise Holohan, Spiros Antonatos, Stefano Braghin, Pól Mac Aonghusa (2020): The Bounded Laplace Mechanism in Differential Privacy. Journal of Privacy and Confidentiality 10, no. 1, Dezember 2020 https://doi.org/10.29012/jpc.715

[4] B. Steffes, R. Schwarz, M. Schröder, J. M. Hucke (2024): Evaluationsbericht. Ergebnisbericht D6.1, WearPrivate-Projekt

[5] Svenja Polst, Philipp Neuschwander, Reinhard Schwarz, Bianca Steffes, Simone Salemi (2024): Anforderungsdokument. Ergebnisbericht D1.1, WearPrivate-Projekt